Data Mining mit der Support Vektor Maschine

Zur Erlangung des Grades

Doktor der Wirtschaftswissenschaften

(Dr. rer. pol.)

am

Department Wirtschaftswissenschaften der Universität Hamburg

eingereichte

kumulative Dissertation

von Stefan Lessmann

Mitglieder der Promotionskommission:

Vorsitzender: Prof. Dr. Hartmut Stadtler

Erstgutachter: Prof. Dr. Dr. h.c. Dieter B. Preßmar

Zweitgutachter: Prof. Dr. Stefan Voß

Das wissenschaftliche Gespräch fand am 21. Dezember 2007 statt.

Hiermit erkläre ich an Eides statt, dass ich die vorliegende Arbeit selbstständig und ohne fremde Hilfe nur unter Verwendung der angeführten Literatur angefertigt habe.

Stefan Lessmann

Erklärung zum Promotionsvorhaben

Hiermit erkläre ich, dass ich zuvor noch keiner Promotionsprüfung unterzogen wurde sowie ich mich noch um keine Zulassung an der Universität Hamburg bzw. einer anderen Universität beworben habe. Weiterhin habe ich noch keiner Universität oder ähnlichen Einrichtung eine Dissertation vorgelegt.

_____

Stefan Lessmann

– Meinen Eltern –

Data Mining mit der Support Vektor Maschine

Die kumulative Dissertation entstand während meiner Tätigkeit als wissenschaftlicher Mitarbeiter und später als Lehrkraft für besondere Aufgaben am Institut für Wirtschaftsinformatik der Universität Hamburg. Sie beschreibt in 14 Fachartikeln die sogenannte Support Vektor Maschine als eine aktuell diskutierte Methodik zur Lösung betriebswirtschaftlicher Klassifikationsprobleme. Die Klassifikation wird dabei als eine Aufgabenstellung des Data Minings verstanden, welches die Aufdeckung nichttrivialer, geschäftsrelevanter Muster und Zusammenhänge in großen Datenbeständen zum Ziel hat. Die Verfügbarkeit großer Datenmengen in der betrieblichen Praxis folgt unmittelbar aus dem umfassenden Einsatz von Informations- und Kommunikationssystemen in sämtlichen Unternehmensbereichen. Damit steigt die Bedeutung von Data Mining, um diese Daten zu analysieren und im Sinne einer Wissensentdeckung zur Verbesserung von Geschäftsprozessen oder allgemein der Erzielung von Wettbewerbsvorteilen zu nutzen. Von besonderer Relevanz ist in diesem Zusammenhang das Kundenbeziehungsmanagement, welches sich als Konzept zur Reaktion auf verschärfte Wettbewerbsbedingungen sowie voranschreitende Marktsättigung und tendenziell abnehmende Kundenloyalität etabliert hat. Eine wesentliche Zielsetzung dieser Managementphilosophie besteht in dem Aufbau, beziehungsweise dem Ausbau, langfristiger und profitabler Kundenbeziehungen. Data Mining liefert hierzu das nötige analytische Instrumentarium, um Kundenwünsche und -potentiale zu erkennen, zu verstehen und in geeignete Produkte und Dienstleistungen umzusetzen.

Vor diesem Hintergrund entstammen die in der vorliegenden Arbeit zur Evaluation des Support Vektor Verfahrens untersuchten Planungs- und Entscheidungsprobleme vornehmlich dem Kundenbeziehungsmanagement. Hierzu gehören beispielsweise Klassifikationsprobleme in den Bereichen Direktmarketing, Kreditwürdigkeitsprüfung, Stornoprophylaxe und Betrugserkennung sowie die durch diesen Kontext erforderlichen algorithmischen Modifikationen und Erweiterungen der Support Vektor Maschine. Das Potential dieser Modifikationen konnte im Rahmen von breit angelegten empirischen Studien bestätigt werden.

Zusammenfassend ergibt sich ein wesentlicher Beitrag zum derzeitigen Stand der Forschung durch den Einsatz eines noch wenig beachteten Verfahrens zur Lösung praxisrelevanter Planungs- und Entscheidungsprobleme sowie methodischer Erweiterungen, welche aus den besonderen Anforderungen der betrachteten Anwendungen abgeleitet wurden. Ferner ist der prozessorientierte Data Mining Analyseansatz in dieser Form innerhalb der Literatur zu Support Vektor Maschinen neuartig. Im Speziellen stellt der Entwurf eines ganzheitlichen, methodisch konsistenten Vorgehensmodells zur Lösung betriebswirtschaftlicher Klassifikationsprobleme mittels Support Vektor Verfahren einen signifikanten wissenschaftlichen Erkenntnisgewinn dar.

Mit der kumulativen Dissertation eingereichte Fachartikel

Veröffentlichungen in Zeitschriften

- S. Lessmann, M.-C. Sung und J. E. V. Johnson. Identifying winners of competitive events: A SVM-based classification model for horserace prediction. *European Journal of Operational Research,* – Zur Veröffentlichung angenommen (doi: 10.1016/j.ejor.2008.03.018) – (2008)
- S. F. Crone, S. Lessmann und R. Stahlbock. The impact of preprocessing on data mining: An evaluation of classifier sensitivity in direct marketing. *European Journal of Operational Research*, 173(3), 781–800 (2006)
- S. Lessmann. Customer relationship management. *WISU - das Wirtschaftsstudium*, 32(2), 190–192 (2003)

Beiträge im Begutachtungsprozess

- S. Lessmann und S. Voß. A framework for customer-centric data mining with support vector machines. *European Journal of Operational Research* – unter Begutachtung – (2007)
- S. Lessmann, B. Baesens, C. Mues und S. Pietsch. Benchmarking classification models for software defect prediction: A proposed framework and novel findings. *IEEE Transactions on Software Engineering* – unter Begutachtung – (2007)

Beiträge in Konferenz- und Sammelbänden

- S. Lessmann, N. Li und S. Voß. A Case Study of Core Vector Machines in Corporate Data Mining. In: *Proc. of the 41st Hawaii Intern. Conf. on System Sciences (HICSS'08)*, Hawaii, USA, IEEE Computer Society, 1–9 (2008)
- S. Lessmann, S. F. Crone, R. Stahlbock und N. Zacher. An Evaluation of Discrete Support Vector Machines for Cost-Sensitive Learning. In: *Proc. of the Intern. Joint Conf. on Neural Networks (IJCNN'06)*, Vancouver, Britisch-Kolumbien, Kanada, IEEE Computer Society, 347–354 (2006)
- S. Lessmann, R. Stahlbock und S. F. Crone. Genetic Algorithms for Support Vector Machine Model Selection. In: *Proc. of the Intern. Joint Conf. on Neural Networks (IJCNN'06)*, Vancouver, Britisch-Kolumbien, Kanada, IEEE Computer Society, 3063–3069 (2006)
- S. F. Crone, S. Lessmann und S. Pietsch. Forecasting with Computational Intelligence – An Evaluation of Support Vector Regression and Artificial Neural Networks for Time Series Prediction. In: *Proc. of the Intern. Joint Conf. on Neural Networks (IJCNN'06)*, Vancouver, Britisch-Kolumbien, Kanada, IEEE Computer Society, 3159–3166 (2006)
- S. Lessmann, S. F. Crone und R. Stahlbock. Genetically Constructed Kernels for Support Vector Machines. In: H. D. Haasis, H. Kopfer und J. Schönberger (Hrsg.) *Operations Research Proceedings 2005*, Berlin: Springer, 257–262 (2005)
- S. Lessmann und R. Stahlbock. Support Vektor Klassifikatoren im analytischen Kundenbeziehungsmanagement. In: H. Rommelfanger (Hrsg.) *Neue Anwendungen von Fuzzy-Logik und Künstlicher Intelligenz*, Aachen: Shaker Verlag, 113–124 (2005)
- S. F. Crone, S. Lessmann und R. Stahlbock. Empirical Comparison and Evaluation of Classifier Performance for Data Mining in Customer Relationship Management. In: *Proc. of the Intern. Joint Conf. on Neural Networks (IJCNN'04)*, Budapest, Ungarn, IEEE Computer Society, 443–448 (2004)
- S. Lessmann. Solving Imbalanced Classification Problems with Support Vector Machines. In: *Proc. of the Intern. Conf. on Artificial Intelligence (IC-AI'04)*, Las Vegas, Nevada, USA, CSREA Press, 214–220 (2004)

Arbeitspapier:

- R. Stahlbock und S. Lessmann. Potential von Support Vektor Maschinen im analytischen Customer Relationship Management. Arbeitspapier, Universität Hamburg (2003).

Ko-Autorenschaft bei den eingereichten Fachartikeln

Professoren

- Johnnie Johnson, University of Southampton
- Stefan Voß, Universität Hamburg

Assistenzprofessoren und wissenschaftliche Mitarbeiter

- Bart Baesens, University of Southampton
- Sven F. Crone, University of Lancaster
- Christophe Mues, University of Southampton
- Robert Stahlbock, Universität Hamburg
- Ming Chien Sung, University of Southampton

Studierende

- Ning Li, Universität Hamburg
- Swantje Pietsch, Universität Hamburg
- Nico Zacher, Universität Hamburg

Inhaltsverzeichnis

Teil I

Begründung des thematischen Zusammenhangs

Kapitel 1

Data Mining mit der Support Vektor Maschine

1.1 Thematische Einordnung

Der Begriff Data Mining beschreibt allgemein die Analyse umfangreicher Datenbestände zur Aufdeckung nicht-trivialer, geschäftsrelevanter und verständlicher Muster in Daten. Data Mining kann als ein Paradigma zur Generierung von Hypothesen über Zusammenhänge in Daten verstanden werden, welches sich durch einen höheren Automationsgrad auszeichnet, als klassische, beispielsweise statistische Analyseansätze. Das aktuelle Interesse an Data Mining in der betrieblichen Praxis ergibt sich zum einen aus der kostengünstigen Verfügbarkeit leistungsfähiger, komplexe Datenanalysen ermöglichender Computerhardware sowie zum anderen aus dem Vorhandensein großer Datenbestände in Unternehmensdatenbanken. Das Bestreben, diese Daten im Sinne von Data Mining zur Generierung von Wissen zu nutzen, ist somit eine logische Konsequenz des stetig voranschreitenden Einsatzes von Informationssystemen zur Unterstützung oder Automation administrativer und dispositiver Tätigkeiten in sämtlichen Unternehmensbereichen.

Von besonderer Bedeutung ist in diesem Zusammenhang das Kundenbeziehungsmanagement (engl. Customer Relationship Management), welches eine Managementphilosophie beschreibt, die den Kunden in den Mittelpunkt unternehmerischer Aktivitäten rückt, um veränderten Umweltbedingungen besser Rechnung tragen zu können. So bedingen Globalisierung und Deregulierung, flankiert von Transaktionskosten reduzierenden, technologischen Innovationen wie zum Beispiel Electronic Commerce, eine erhebliche Wettbewerbsintensivierung. Diese wird durch eine auf vielen Konsumgütermärkten zu beobachtende Marktsättigung sowie tendenziell abnehmende Kundenloyalität weiter verschärft. Letztere kann ebenfalls durch technischen Fort-

schritt erklärt werden, da die Markttransparenz und damit Vergleichbarkeit alternativer Produkte durch das Internet spürbar zugenommen hat, Produkte sich parallel in ihren funktionalen Eigenschaften immer weiter angleichen (sogenannte Produkthomogenisierung) und auch für Konsumenten Transaktionskosten, zum Beispiel beim Wechsel eines Anbieters, erheblich abgenommen haben. Ausgehend von der Annahme, dass die Erhaltung eines Kunden, beziehungsweise der Ausbau einer bestehenden Kundenbeziehung im Sinne von „Cross/Up-Selling“, mit signifikant geringeren Kosten verbunden ist als die Gewinnung von Neukunden, sollen im Rahmen des Kundenbeziehungsmanagements langfristige und profitable Kundenbeziehungen aufgebaut werden. Charakteristisch ist dabei ein umfassender Einsatz von Informations- und Kommunikationssystemen zur Unterstützung kundennaher Prozesse in Marketing, Vertrieb und Service.

Data Mining ist ein wesentlicher Eckpfeiler dieses Managementkonzeptes: Zum einen kann Data Mining unmittelbar zur Lösung einer Reihe operativer Planungs- und Entscheidungsprobleme im Kundenbeziehungsmanagement beitragen. Weiterhin sollen die durch Analysen gewonnenen Erkenntnisse im Sinne einer kontinuierlichen Verbesserung eingesetzt werden und den Aufbau eines logischen Kreislaufsystems ermöglichen, welches die Erfassung von Daten an Kundenkontaktpunkten, deren Speicherung und Harmonisierung sowie anschließende Auswertung in einem analytischen Back-End und eine darauf aufbauende Überprüfung, beziehungsweise, bei Bedarf, Anpassung oder gar Neugestaltung betroffener Geschäftsprozesse umfasst.

Das Kundenbeziehungsmanagement bildet entsprechend den betriebswirtschaftlichen Rahmen der vorliegenden Arbeit und ein Großteil der empirisch untersuchten Fragestellungen entstammt diesem Anwendungsfeld. Dabei werden ausschließlich operative Planungsaufgaben betrachtet, die als Klassifikationsproblem modelliert werden können. Das heißt, eine Entscheidung ist jeweils durch die Einordnung eines Objekts, beispielsweise eines Kunden, in eine von mehreren vordefinierten Gruppen, zum Beispiel „hohes Risiko“ und „geringes Risiko“ in der Kreditwürdigkeitsprüfung, repräsentiert. Diese Eigenschaft ermöglicht den Einsatz einer entsprechenden Klasse von Data Mining Methoden, zu denen auch die sogenannte Support Vektor Maschine gehört.

Ferner bedingt der operative Charakter der untersuchten Problemstellungen einen hohen Automationsgrad, welcher die wirtschaftsinformatische Relevanz der Themenstellung dokumentiert. Die ursprünglich von Mertens vorgeschlagene „sinnhafte Vollautomation“ ist heute als ein Globalziel der Wirtschaftsinformatik anerkannt und umfasst damit auch planerische Aufgabenstellungen im Kundenbeziehungsmanagement. Dabei kann ein Data Mining Modell, je nach betriebswirtschaftlicher Bedeutung des betrachteten Entscheidungsproblems, eine angemessene Handlungsalternative entweder vollautomatisch auswählen oder eine Unterstützungsfunktion bei der Lösung schlecht strukturier Entscheidungsprobleme ausüben. Die mangelnde Strukturiertheit ergibt sich dabei häufig aus der hohen Dimensionalität der dem Entscheidungsproblem zugrundeliegenden Daten. Beispielsweise liegen über Kunden häufig sehr detaillierte Daten vor (demografische Daten, mikrografische Daten, Daten aus der Transaktionshistorie, etc.) die in ihrer Gesamtheit eine Größe wie das Kündigungsrisiko beeinflussen mögen, aber unmöglich durch einen menschlichen Entscheidungsträger simultan verarbeitet werden können. In solchen Situationen bietet das Data Mining geeignete Techniken an, um entscheidungsrelevante Zusammenhänge anhand von Vergangenheitsdaten selbstständig zu erlernen und in einer aggregierten Größe, beispielsweise einer geschätzten Kündigungswahrscheinlichkeit, zu verdichten.

Die in der Arbeit untersuchte Support Vektor Maschine repräsentiert eine solche Technik. Das maschinelle Lernen basiert dabei auf der mathematischen Optimierung. Im Falle der Support Vektor Maschine wird ein funktionaler Zusammenhang zwischen vorliegenden Beispieldaten und einer zu modellierenden diskreten Zielgröße angenommen. Die freien Parameter des resultierenden Prognosemodells werden durch die Lösung eines konvexen, quadratischen Programms bestimmt. Optimierungsprobleme dieser Art werden traditionell in der Mathematik, und, insbesondere vor dem Hintergrund eines konkreten Anwendungsproblems, im Operations Research betrachtet. Entsprechend bestehen zwischen den Bereichen Data Mining und Operations Research erhebliche Synergien. Dabei können Methoden des Operations Research nicht nur zur Lösung der im Zusammenhang mit dem maschinellen Lernen auftretenden Optimierungsprobleme eingesetzt werden, sondern auch in der Datenvorverarbeitung wertvolle Beiträge liefern. Die dem eigentlichen Data Mining vorgelagerten Datenaufberei-

tungsschritte sind häufig kombinatorischer Natur. Als Beispiel sei hier die Merkmalsselektion angeführt, welche die Auswahl einer optimalen Menge an Attributen zur Repräsentation eines zu verarbeitenden Objektes anstrebt. Die Lösung solcher kombinatorischer Planungsprobleme mittels exakter Verfahren oder intelligenter Heuristiken gehört zu den Kernkompetenzen des Operations Research. Schlussendlich wird die Relevanz von Data Mining im Operations Research auch durch die große Anzahl entsprechender Publikationen in einschlägigen Zeitschriften dokumentiert.[1]

Die kumulative Dissertation besitzt gemäß der vorangehenden Darstellung einen interdisziplinären Charakter. Es sollen betriebswirtschaftliche Fragestellungen als Klassifikationsproblem abgebildet und durch Einsatz von Techniken des Operations Research, beziehungsweise des maschinellen Lernens, gelöst werden. Entsprechend dem Kerngedanken der Wirtschaftsinformatik wird dabei ein prozessorientierter Ansatz verfolgt und versucht, die Belastung des eigentlichen Entscheiders, sei es hinsichtlich des Umgangs mit der Planungstechnik oder der Nachbearbeitung der gelieferten Ergebnisse, durch einen hohen Automationsgrad strikt zu begrenzen.

1.2 Zielsetzung und Motivation

Im Mittelpunkt der Arbeit steht die Support Vektor Maschine, welche eine aktuell diskutierte Methodik zur Klassifikation repräsentiert und auf ihre Eignung zur Unterstützung ausgewählter betriebswirtschaftlicher Planungs- und Entscheidungsprobleme untersucht wird.

Support Vektor Maschinen gehören zu den Prognoseverfahren und ermöglichen die Vorhersage einer Gruppenzugehörigkeit auf der Basis vorliegender Beispieldatensätze. Fragestellungen dieser Art sowie entsprechende Lösungsmethoden werden in der Statistik und dem Operations Research schon seit vielen Jahren untersucht. Beispielsweise formulierte Mangasarian bereits 1965 ein mathematisches Programm, welches formal eine gewisse Ähnlichkeit mit dem der Support Vektor Maschine zugrundeliegendem

[1] Beispielhaft seien hier aktuelle Spezialausgaben dreier führender Operations Research Zeitschriften genannt, die ausschließlich dem Data Mining gewidmet sind: *Computer & OR (2006)*, *Annals of Operations Research (2007)* und *Journal of the Operational Research Society (2007)*.

Optimierungsproblem aufweist. Eine Besonderheit des 1992 erstmalig von Boser, Guyon und Vapnik vorgestellten Support Vektor Verfahrens ergibt sich jedoch daraus, dass es unmittelbar auf den theoretischen Erkenntnissen der statistischen Lerntheorie aufsetzt und diese in algorithmischer Form implementiert. Die nach ihren Erfindern auch als Vapnik-Chervonenkis Theorie bezeichnete statistische Lerntheorie untersucht die formalen Voraussetzungen für das maschinelle Lernen. Dies beinhaltet eine Analyse, unter welchen Bedingungen ein aus Beispieldaten erlerntes Prognosemodell zutreffende Vorhersagen auf unbekannten Daten liefern wird, das heißt, in der Lage ist zu generalisieren. Aufbauend auf den gewonnenen Erkenntnissen wurde die Support Vektor Maschine so konstruiert, dass das Risiko einer Fehlprognose auf unbekannten Daten minimiert wird.[2] Dieses theoretische Fundament motiviert eine empirische Validierung der Leistungsfähigkeit des Verfahrens und führte zu zahlreichen Anwendungen in unterschiedlichen Disziplinen. Dabei wurden vornehmlich Fragestellungen aus dem Bereich der medizinischen Diagnostik sowie der Text- und Bilderkennung oder der Dokumentenklassifikation im Information Retrieval untersucht. Betriebswirtschaftliche Szenarien werden dagegen nur nachrangig betrachtet.

Eine wesentliche Motivation der Arbeit besteht folglich in einer Untersuchung, in wie weit Support Vektor Maschinen auch zur Lösung ausgewählter Klassifikationsprobleme aus der Betriebswirtschaft zielführend eingesetzt werden können. Die betrachteten Aufgabenstellungen entstammen dabei vornehmlich dem Kundenbeziehungsmanagement und repräsentieren operative Planungs- und Entscheidungsprobleme. Beispielhaft seien hier die Kreditwürdigkeitsprüfung im Finanzdienstleistungsbereich (engl. credit scoring), die Zielgruppenbestimmung im Direktmarketing (engl. repeat purchase modelling), die Prognose von Kündigungswahrscheinlichkeiten (engl. customer attrition analysis oder churn prediction), zum Beispiel bei Mobilfunk- oder Internetnutzungsverträgen sowie die Identifikation betrügerischer Geschäftstrans-

[2] Diese Fokussierung auf Fragestellungen der Prognose repräsentiert einen der wichtigsten Unterschiede zwischen Support Vektor Maschinen und frühen Arbeiten aus dem Operations Research. Erstere lösen ein mathematisches Programm zur Konstruktion eines generalisierbaren Klassifikators, während letztere sich auf die Formulierung eines – mathematisch ähnlichen – Programms zur Separation zweier Mengen konzentrieren. Im Sinne der statistischen Lerntheorie kommt dies lediglich einer Betrachtung der empirischen Risikominimierung gleich.

aktionen (engl. fraud detection) angeführt. Die betrachteten Fragestellungen weisen zum Teil erhebliche strukturelle Unterschiede gegenüber medizinischen oder informatischen Anwendungen auf, welche die Erstellung von Vorhersagemodellen erschweren und folglich geeignet behandelt werden müssen. So sind die mit einer fehlerhaften Prognose assoziierten Kosten typischer Weise asymmetrisch. Ein Beispiel ist die Kreditwürdigkeitsprüfung: Wird ein Antragsteller fälschlicher Weise abgelehnt obwohl er den gewünschten Kredit ordnungsgemäß zurückgezahlt hätte, ergeben sich die Fehlklassifikationskosten aus dem entgangenen Zinsgewinn. Der komplementäre Fehler, die Vergabe eines Kredits, der nicht zurückgezahlt wird, ist offenkundig mit erheblich höheren Kosten verbunden. Dementsprechend ist beim Einsatz eines Klassifikationsverfahrens in besonderem Maße darauf zu achten, dass dieser Fehlertypus vermieden wird. Weiterhin ist die betriebswirtschaftlich relevante Gruppe häufig gegenüber einer entsprechenden Alternativgruppe stark unterrepräsentiert. Besonders deutlich zeigt sich dieser Effekt bei der Betrugserkennung, wo eine typischer Weise kleine Anzahl betrügerischer Vorgänge einer sehr großen Menge an regulären Geschäftstransaktionen gegenübersteht.

Vor dem Hintergrund solcher Besonderheiten sind positive Ergebnisse aus anderen Domänen nicht unmittelbar auf die hier betrachteten Fragestellungen übertragbar, sondern bedürfen einer empirischen Validierung. Diese beinhaltet neben der reinen Anwendung der Support Vektor Maschine auch einen komparativen Vergleich mit etablierten Alternativen wie zum Beispiel der logistischen Regression oder Entscheidungsbaumverfahren und das dazugehörige Experimentdesign inklusive statistischer Testverfahren. Diese anwendungsorientierte Potentialanalyse der Support Vektor Maschine steht in engem Zusammenhang mit der methodischen Dimension der vorliegenden Arbeit. Hier soll gezeigt werden, wie die besonderen Anforderungen des Anwendungsfeldes durch Erweiterungen und Modifikationen der Support Vektor Maschine, beziehungsweise deren Integration mit anderen Data Mining/Operations Research Methoden im Sinne einer Hybridisierung, geeignet erfüllt werden können. Dabei wird besonderer Wert auf eine ganzheitliche Betrachtung gelegt, welche über die eigentliche Prognose hinausgeht und auch die im Sinne eines Prozesses zur Wissensentdeckung in Datenbanken (engl. knowledge discovery in databases) vor- und nach-

gelagerten Analyseschritte berücksichtigt. Diese prozessorientierte Sichtweise dient unter anderem dem wirtschaftsinformatischen Automationsgedanken und soll nicht zuletzt eine stärkere Verbreitung von Support Vektor Maschinen in der betrieblichen Praxis begünstigen.

Zusammenfassend bietet die Arbeit einen wissenschaftlichen Erkenntnisgewinn, der sich aus dem Einsatz der Support Vektor Maschine zur Lösung von bisher nur nachrangig mit diesem Verfahren betrachteten Planungs- und Entscheidungsproblemen und dem Entwurf entsprechender methodischer Erweiterungen sowie der prozessorientierten Perspektive ergibt. Dabei wird ein empirisch-induktiver Forschungsansatz verfolgt, welcher von einer konkreten Problemstellung ausgeht, mittels geeigneter Experimente spezifische Ergebnisse liefert und diese – gegebenenfalls – zu verallgemeinerungsfähigen Erkenntnissen generalisiert.

Im Folgenden werden die im Rahmen der kumulativen Promotion eingebrachten Fachartikel vorgestellt und in eine thematische Reihenfolge gebracht. Diese Einordnung erfolgt entlang dreier Dimensionen, die das Potential von Support Vektor Maschinen in ausgewählten Problemstellungen, methodische Erweiterungen sowie den prozessorientierten Analyseansatz repräsentieren. Dabei werden grundlegende Zielsetzungen und Resultate angeführt, aber nicht im Detail erläutert; eine vollständige Reproduktion der entsprechenden Aufsätze findet sich in Teil III.

1.3 Anwendung von Support Vektor Maschinen

Im Rahmen der kumulativen Dissertation wird die Support Vektor Maschine zur Lösung ausgewählter Klassifikationsprobleme eingesetzt. Die Klassifikation wird dabei als eine Ausprägung von (prognostischem) Data Mining verstanden. Von besonderer Bedeutung ist dabei das Kundenbeziehungsmanagement, welches das wohl wichtigste Einsatzfeld von Data Mining Techniken in der Betriebswirtschaftslehre darstellt. Vor diesem Hintergrund erfolgt in [1] eine Charakterisierung dieser Managementphilosophie sowie dessen operativer, kollaborativer und analytischer Dimension. Jeder dieser drei logischen Teilbereiche bedarf entsprechender Informationssysteme zur Unterstützung der jeweiligen operativen, kollaborativen und analytischen Geschäftsprozesse. Um das dem Kundenbeziehungsmanagement inhärente Grundsatzziel eines organisa-

tionalen Lernens zu erreichen, ist darüber hinaus eine Integration dieser Teilsysteme erforderlich, welche eine einheitliche Sicht auf sämtliche kundenrelevanten Daten für alle betroffenen Abteilungen gewährleistet. Dies illustriert auch die Bedeutung des Kundenbeziehungsmanagements als Forschungsgegenstand der Wirtschaftsinformatik.

Eine vertiefende Betrachtung des Kundenbeziehungsmanagements findet sich daher in [2], wobei insbesondere auf die Rolle von Data Mining eingegangen wird und analytische Aufgabenstellungen im Detail erklärt werden. Es wird gezeigt, dass vor allem klassifikatorische Fragestellungen von erheblicher Bedeutung sind, da eine Vielzahl operativer Entscheidungsprobleme als Klassifikationsproblem modelliert werden können. Grundsätzlich werden Kunden dabei über einen Vektor repräsentiert, dessen Komponenten die über den betreffenden Kunden verfügbaren Daten beinhalten. Die Zielvariable einer Klassifikationsanalyse wird dann entsprechend der zugrundeliegenden Fragestellung gebildet und liefert für jeden „Kundenvektor“ eine zugehörige Kategorie; beispielsweise hohes/niedriges Risiko bei der Kreditwürdigkeitsprüfung. Aus der grundsätzlichen Bedeutung von Klassifikation ergibt sich auch die Relevanz der Support Vektor Maschine als möglichem Lösungsverfahren. Um der geringen Bekanntheit dieser Methodik in der betriebswirtschaftlichen Forschung Rechnung zu tragen, bietet [2] ferner eine ausführliche Beschreibung der zugrundeliegenden Mathematik.

In nachfolgenden Arbeiten werden die im Kundenbeziehungsmanagement anzutreffenden Herausforderungen wie das Problem asymmetrischer Klassenverteilungen oder die kostensensitive Klassifikation zunächst isoliert betrachtet. In diesem Zusammenhang wird in [3] gezeigt, dass Support Vektor Maschinen ungleiche Klassenverteilungen durch eine entsprechende Einstellung der Verfahrensparameter korrigieren können. Ein weiteres Ergebnis ist, dass die Methode darüber hinaus auch mit verfahrensunabhängigen, externen Ausgleichungstechniken hervorragend zusammenarbeitet. Dieses Resultat wird in [4] auf das Problem der kostensensitiven Klassifikation übertragen. Die Ergebnisse dieser Arbeit weisen ebenfalls darauf hin, dass eine adäquate Berücksichtigung asymmetrischer Fehlerkosten über eine einfache Parametrisierungsheuristik der Support Vektor Maschine erreicht werden kann.

Ferner stellt die Masse der im Kundenbeziehungsmanagement prinzipiell verfügbaren Daten eine grundsätzliche Herausforderung für jedes Analyseverfahren dar. Dementsprechend wird in [5] eine modifizierte Support Vektor Maschine untersucht, die speziell für die Verarbeitung sehr großer Datenmengen entwickelt wurde. Die Ergebnisse belegen, dass die bemerkenswerten Laufzeitergebnisse dieser sogenannten Core Vector Maschine auf Klassifikationsprobleme im Kundenbeziehungsmanagement übertragen werden können. Für die Verarbeitung eines repräsentativen Datensatzes von 300.000 Kunden werden beispielsweise weniger als zwei Minuten benötigt.

Aufbauend auf den vorangegangenen Ergebnissen wird in [6] ein ganzheitliches Vorgehensmodell zur Lösung klassifikatorischer Fragestellungen aus dem Kundenbeziehungsmanagement auf Basis der Support Vektor Maschine konzipiert und in [7] erweitert, implementiert und empirisch validiert. Zunächst erfolgt dabei eine Auswahl unabhängiger Variablen, das heißt der einen Kunden beschreibenden Merkmale, durch eine rekursive Merkmalsselektionsheuristik. Die Verwendung einer modifizierten Support Vektor Maschine ermöglicht dabei den Einsatz eines besonders leistungsfähigen Optimierungsalgorithmus und folglich die Verarbeitung sehr großer Datenmengen. Die zweite Phase strebt eine Verbesserung der Prognosegüte an, wobei die betreffende Problemstellung durch die Wiederverwendung von Ergebnissen aus dem vorangegangenen Schritt substanziell vereinfacht werden kann. Das entworfene Referenzmodell zeichnet sich durch ein hohes Maß an Modularität und Automatisierbarkeit sowie methodische Konsistenz aus. Letztere dient nicht zuletzt einer besseren Verständlichkeit des Verfahrens und sollte die Adaption in der betrieblichen Praxis begünstigen. Die Leistungsfähigkeit des Ansatzes wird in einer breit angelegten empirischen Studie untersucht, welche beachtliche Verbesserungen gegenüber etablierten Data Mining Verfahren dokumentiert.

Neben Problemstellungen aus dem Kundenbeziehungsmanagement werden in der kumulativen Dissertation auch ausgewählte Prognoseprobleme aus anderen Bereichen betrachtet. So wird in [8] das Potential verschiedener Klassifikationsverfahren zur Identifikation fehlerhafter Softwarekomponenten untersucht. Im Kern dieser Anwendung steht das klassische betriebswirtschaftliche Problem einer effizienten Ressourcenallokation; ein Klassifikator soll die Fehlerhaftigkeit eines Softwaremoduls prognos-

tizieren, damit wertvolle Prüfungsressourcen so verteilt werden können, dass die Bausteine mit der höchsten Fehlerwahrscheinlichkeit besonders intensiv getestet werden. Die betrachteten Klassifikationsprobleme sind ebenfalls durch asymmetrische Klassenverteilungen und Fehlerkosten charakterisiert, so dass strukturelle Ähnlichkeiten zu den im analytischen Kundenbeziehungsmanagement untersuchten Fragestellungen bestehen. Vor diesem Hintergrund dokumentiert das gute Abschneiden von Support Vektor Maschinen in einem empirischem Vergleich von 19 Klassifikationsverfahren einmal mehr das Potential dieser Methode.

Ein analoges Fazit erlauben auch die in [9] erzielten Ergebnisse. Hier werden Support Vektor Maschinen im Rahmen eines zweistufigen, hybriden Prognosemodells zusammen mit klassischen statistischen Methoden eingesetzt, um die Entwicklung eines Finanzmarktes vorherzusagen. Ziel der Analyse ist es, Rückschlüsse zu ziehen, in wie weit öffentlich verfügbare Marktdaten von Akteuren effizient im Rahmen ihrer Entscheidungsfindung genutzt werden. Die Motivation für den Einsatz von Support Vektor Maschinen ergibt sich daraus, dass diese aufgrund ihres Ursprungs in der statistischen Lerntheorie in besonderem Maße dazu geeignet sind, eine Vielzahl von Informationen zu verarbeiten. Andererseits ist zu erwarten, dass gewöhnliche Markteilnehmer lediglich einen Bruchteil der verfügbaren Informationen nutzen, beziehungsweise nicht in der Lage sind, hochdimensionale Datenstrukturen zu verarbeiten. Die resultierende Hypothese, dass Informationen im betrachteten Wettmarkt nicht effizient eingesetzt werden, konnte im Rahmen eines entsprechenden Experiments bestätigt werden. Auch hier konnte die Prognosegüte des entworfenen Modells durch den Einsatz der Support Vektor Maschine in bemerkenswerter Weise gesteigert werden.

1.4 Erweiterungen und Modifikationen der Support Vektor Maschine

Im Rahmen der kumulativen Dissertation werden neben dem ursprünglichen Support Vektor Verfahren auch methodische Erweiterungen und Modifikationen evaluiert, beziehungsweise selbst entworfen. Dieses dient in erster Line einer verbesserten Abbildung der sich aus einem konkreten Anwendungsproblem ergebenden Anforderungen. Dementsprechend wird in [4] eine sogenannte diskrete Support Vektor Ma-

schine untersucht, welche eine direkte Minimierung von Fehlklassifikationskosten erlaubt und damit für Fragestellungen des Kundenbeziehungsmanagements besonders geeignet ist. Zur Konstruktion dieses Klassifikators wird ein heuristisches Suchverfahren, eine sogenannte Tabu-Suche, implementiert, um das resultierende gemischt-ganzzahlige Optimierungsproblem zu lösen. Eine ähnliche Anwendung von Meta-Heuristiken findet sich ferner in [10], wo ein Genetischer Algorithmus eingesetzt wird, um die freien Parameter einer Support Vektor Maschine zu konfigurieren. Gewöhnlich muss diese Aufgabe vom Anwender selbst übernommen werden, so dass die Verwendung eines Suchverfahrens zum einen, über eine adäquatere Parametereinstellung, eine Verbesserung der Prognosequalität und zum anderen einen allgemein höheren Automationsgrad ermöglicht. Ein Problem ergibt sich allerdings aus dem hohen Rechenbedarf der Kombination aus Genetischem Algorithmus und Support Vektor Maschine. Daher wird in [11] untersucht, in wie weit das die heuristische Parametersuche steuernde Selektionskriterium durch eine effizient zu berechnende Schranke des Generalisierungsfehlers substituiert werden kann. Insgesamt bietet das als GA-SVM bezeichnete Verfahren eine interessante Alternative zu derzeit gebräuchlichen Parametrisierungsstrategien.

Als Modifikation der Support Vektor Maschine kann auch die in [12] verwendete Support Vektor Regression verstanden werden, welche im Gegensatz zur Klassifikation die Vorhersage kontinuierlichen Zielgrößen ermöglicht. Damit ist dieses Verfahren auch für die Zeitreihenanalyse einsetzbar, was in [12] exemplarisch am Beispiel der Absatzprognose gezeigt wird.

Methodische Erweiterungen finden sich ferner in den Arbeiten [9] und [7]. Analog zu GA-SVM steht dabei die Integration von Support Vektor Maschinen und anderen Planungsmethoden im Vordergrund. So wird in [9] ein Klassifikationsproblem betrachtet, bei dem die zu klassifizierenden Objekte in einem Wettbewerbszusammenhang stehen. Das heißt, die Klassenzugehörigkeit eines Objekts wird nicht nur über die dieses Objekt beschreibenden Merkmale sondern auch über die Eigenschaften bestimmter Konkurrenten beeinflusst. Um diesen Zusammenhang geeignet abzubilden, wird ein zweistufiges Verfahren entworfen, welches die Support Vektor Maschine zum Verarbeiten

einer großen Menge an Eingabedaten[3] nutzt und anschließend die Wettbewerbsbeziehungen über ein statistisches Prognosemodell berücksichtigt.

Ein zweistufiges Verfahren wird auch in [7] im Zusammenhang mit dem Entwurf eines umfassenden Referenzmodells für den Einsatz von Support Vektor Maschinen im kundenbezogenen Data Mining entwickelt. Im Vordergrund steht dabei die Integration verschiedener aktueller Erweiterungen des Support Vektor Algorithmus. So wird zunächst ein neues Optimierungsverfahren zur Verarbeitung großer Datenmengen eingesetzt, um die Menge der Ausgangsdaten für eine nachfolgende, die Prognosequalität optimierende Phase, zu reduzieren. Diese Reduktion ergibt sich einerseits hinsichtlich der Zahl der zu verarbeitenden Attribute durch Elimination wenig relevanter Eigenschaften mittels einer rekursiven Merkmalsselektionsheuristik und zum anderen durch eine Löschung redundanter Datensätze auf der Basis eines Support Vektor-basierten Filtermechanismus. Weiterhin wird gezeigt, wie Teile des in dem resultierenden Prognosemodell verborgenen Wissens expliziert werden können.

1.5 Support Vektor Maschinen im Data Mining Prozess

Der Prozess zur Wissensentdeckung in Datenbanken lässt sich im Wesentlichen durch die Teilschritte Problemdefinition, Datenvorverarbeitung, Datenanalyse und Evaluation beschreiben. Die Datenvorverarbeitung beinhaltet eine Transformation der zu untersuchenden Daten, um eine Repräsentation zu erzeugen, die durch mathematische Lernalgorithmen verarbeitet werden kann. Im zweiten Schritt erfolgt die Anwendung einer oder mehrerer Data Mining Methoden, welche zuvor auf der Basis des betrachteten Anwendungsproblems (zum Beispiel Regression, Klassifikation, Segmentierung oder Assoziation) ausgewählt wurden. Es folgt eine Evaluation der Methoden und die endgültige Auswahl eines Modells. Alternativ kann zu einer der Vorphasen zurückgegangen werden, um durch inkrementelle Änderungen, beispielsweise eine andere Form der Datenvorverarbeitung, die Gesamtqualität der Analyse zu verbessern.

---

[3] Mit Eingabedaten sind hier die ein Objekt beschreibenden Merkmale gemeint. Das heißt, die Support Vektor Maschine wird in erster Linie aufgrund ihrer Fähigkeit zum Lernen in hochdimensionalen Merkmalsräumen eingesetzt.

Diese prozessorientierte Sichtweise wird auch in der vorliegenden Arbeit verfolgt, so dass die einzelnen Aufsätze der kumulativen Dissertation entlang der genannten Teilschritte des Data Mining Prozesses thematisch eingeordnet werden können.

1.5.1. Problemdefinition

Eine Erläuterung des zugrundeliegenden Anwendungsproblems ist grundsätzlich Bestandteil eines jeden Aufsatzes der vorliegenden kumulativen Dissertation. Für das vornehmlich betrachtete Feld des Kundenbeziehungsmanagements erfolgt dies insbesondere in den Arbeiten [1] und [2]. Darüber hinaus werden die sich aus der Domäne ergebenden Anforderungen in [6] vertieft.

Sofern in Arbeiten andere betriebswirtschaftliche Fragestellungen als das Kundenbeziehungsmanagement betrachtet werden, erfolgt eine ausführliche Problemdefinition unmittelbar in dem jeweiligen Aufsatz; so beispielsweise im Zusammenhang mit der Prognose fehlerhafter Softwarekomponenten in [8], der Modellierung eines Finanzmarktes in [9] oder der Absatzprognose in [12].

1.5.2. Datenvorverarbeitung

Die Datenvorverarbeitung bietet zahlreiche Freiheitsgrade, so dass Datentransformationen, neben der reinen Ermöglichung einer mathematischen Auswertung, auch hinsichtlich ihrer Wirkung auf die Prognosegüte untersucht werden sollten. Dieses Thema wird im Zusammenhang mit Methoden zur Kompensation asymmetrischer Klassenverteilungen in [3] aufgegriffen, um die Eignung bestimmter Methoden zur Stichprobenziehung zu untersuchen. So kann der Anteil einer unterrepräsentierten Klasse durch zufälliges Kopieren der entsprechenden Datensätze künstlich erhöht werden (engl. Oversampling). Alternativ besteht die Möglichkeit, zufällig Datensätze der zahlenmäßig dominierenden Klasse zu löschen, bis ein gewünschtes Verhältnis zwischen beiden Klassen erreicht ist (engl. Undersampling). Für die in [3] betrachtete Fallstudie zeigt sich ein vergleichbarer Einfluss von Over-/Undersampling auf die Prognosegüte. Angesichts des geringeren Rechenaufwands empfiehlt sich daher das zufällige Löschen von Datensätzen der Mehrheitsklasse. Allerdings setzt diese Schlussfolgerung voraus, dass die absolute Anzahl an Datensätzen hinreichend groß ist.

Vertiefende Untersuchungen des Einflusses alternativer Vorverarbeitungstechniken auf die Prognosegüte von Support Vektor Maschinen finden sich in [9] und [13]. Diese Arbeiten zeigen im Rahmen einer Fallstudie aus dem Direktmarketingbereich, dass die durch alternative Vorverarbeitungstechniken induzierte Variabilität der Prognosegüte in etwa derjenigen entspricht, die durch unterschiedliche Parametrisierungen des Klassifikationsverfahrens hervorgerufen wird. Dieser Zusammenhang zeigt sich für Support Vektor Maschinen, Künstliche Neuronale Netzwerke und, in abgeschwächter Form, ebenfalls für Entscheidungsbaumverfahren. Während das Problem der Parameterauswahl in der Literatur große Aufmerksamkeit genießt, werden unterschiedliche Transformationen, zum Beispiel der Effekt einer statistischen Standardisierung gegenüber einer linearen Intervallskalierung, in der Regel nicht betrachtet. In Anbetracht von [13] ist davon auszugehen, dass durch diese Vorgehensweise erhebliche Potentiale zur Prognoseverbesserung ungenutzt bleiben und sich eine stärkere Wahrnehmung der Datenvorverarbeitung empfiehlt.

1.5.3. Datenanalyse

Die Datenanalyse beschreibt das Data Mining im engeren Sinne, also die eigentliche Anwendung einer Data Mining Methode sowie die dazu notwendigen Arbeitsschritte. Dementsprechend besitzen sämtliche im Rahmen der kumulativen Dissertation eingereichten Fachartikel einen engen Bezug zu diesem Prozessschritt.

Grundsätzlich setzt der Einsatz von Support Vektor Maschinen die Einstellung spezifischer Verfahrensparameter voraus, was als Modellselektion bezeichnet wird. Ein weit verbreiteter Ansatz zur Modellselektion besteht in einer empirischen, vollenumerativen Suche über einen a priori definierten, diskreten Parameterraum; sogenannte Grid-Suche. Dieses Prinzip wird in [4, 5, 8, 9, 12, 14] angewendet. Da der hohe Rechenaufwand dieser Strategie bei großen Datensätzen zu Laufzeitproblemen führen kann, wurden in der Literatur verschiedene Alternativen vorgeschlagen. In diesem Sinne wird in [8] eine rekursive Verfeinerung der Grid-Suche eingesetzt, um einerseits eine große Bandbreite von Parameterwerten zu betrachten und andererseits viel versprechende Regionen des Parameterraums intensiv zu durchsuchen. Alternative Suchstrategien werden insbesondere in [7] implementiert und miteinander verglichen. Weitere Bezüge zum Problem der Modellselektion und Parametrisierungsheuristiken fin-

den sich ferner in den Beiträgen [3] und [4] im Zusammenhang mit der Einstellung von Verfahrensparametern bei Vorliegen asymmetrischer Klassen- und/oder Kostenverteilungen. Ein grundsätzlich anderer Ansatz wird hingegen in [10] und [11] verfolgt. Hier werden sämtliche Parameter autonom durch einen Genetischen Algorithmus eingestellt.

1.5.4. Auswertung und Evaluation

Besondere Anforderungen bezüglich der Auswertung von Klassifikationsmethoden ergeben sich im Kundenbeziehungsmanagement durch die beschriebenen Charakteristika der Anwendungsdomäne. So kommt eine klassische, auf Klassifikationsfehlern basierende Evaluation, das heißt der Einsatz entsprechender Metriken, nicht in Betracht, wenn asymmetrische Klassenverteilungen vorliegen. Beispielsweise könnte, bei einer Klassenverteilung von 95 : 5 ein Prognoseverfahren problemlos eine exzellente Trefferrate von 95% erreichen, indem grundsätzlich alle zu klassifizierenden Objekte der Mehrheitsklasse zugeordnet werden. Betriebswirtschaftlich wäre dieser Klassifikator offensichtlich wertlos.

Sofern die notwendigen Informationen vorliegen, empfiehlt sich eine unmittelbare Bewertung der mit einem Prognosefehler verbundenen Kosten. Andernfalls stellt die Fläche unter der „Receiver-Operating-Characteristics" Kurve ein geeignetes Gütekriterium dar; siehe zum Beispiel [3, 4, 13]. Dieses wird in [8] ausführlich beschrieben und im Rahmen eines empirischen Vergleichs von 19 verschiedenen Klassifikationsverfahren eingesetzt.

Eine Gegenüberstellung verschiedener Verfahren findet sich auch in [12] im Zusammenhang mit dem Problem der Absatzprognose. Hier wird gezeigt, dass Support Vektor Maschinen auch kontinuierliche Vorhersagen ermöglichen und im Vergleich zu Künstlichen Neuronalen Netzwerken und klassischen Verfahren der Zeitreihenprognose gute Ergebnisse liefern. Diese Arbeit dokumentiert ferner, dass die Interpretation komparativer Studien gewisse Risiken birgt und sich zum Beispiel durch Einsatz unterschiedlicher Gütekriterien widersprüchliche Ergebnisse ergeben können. Vor diesem Hintergrund werden für das in [8] vorgenommene Experiment statistische Test-

verfahren implementiert, die für den Vergleich von Klassifikationsverfahren besonders geeignet sind.

Die eigentliche Prognose erweiternd, greift [7] die Fragestellung auf, wie, ausgehend von einem einsatzfähigen Support Vektor Klassifikator, Rückschlüsse auf die aus den Beispieldaten erlernten Zusammenhänge, etwa das Kundenverhalten, gezogen werden können. Dieser Aspekt findet sich ferner in [9]. Hier kann im Rahmen einer Untersuchung von Marktmechanismen über den Entwurf einer profitablen Handelsstrategie nachgewiesen werden, dass, entgegen der in der Literatur vorherrschenden Meinung, Marktteilnehmer öffentlich verfügbare Informationen nicht effizient zur Entscheidungsfindung einsetzen.

1.6 Konklusion

Im Rahmen der Promotion erfolgte eine umfassende Evaluation von Support Vektor Maschinen hinsichtlich ihrer Eignung zur Lösung betriebswirtschaftlicher Klassifikationsprobleme. Dabei standen Fragestellungen aus dem Kundenbeziehungsmanagement im Mittelpunkt. Um die Dimension der Evaluation weiter zu vergrößern, wurden ferner ausgewählte Klassifikationsprobleme aus anderen Anwendungsfeldern, beispielsweise die Absatzprognose, die Prognose fehlerhafte Softwarekomponenten oder die Marktpreisprognose, untersucht. Die einzelnen Teilschritte eines dem Data Mining Ansatz folgenden Analyseprozesses wurden individuell betrachtet und entsprechende Handlungsempfehlungen für einen effektiven Verfahrenseinsatz ausgesprochen. Diese wurden anschließend integriert, um ein ganzheitliches Vorgehensmodell zur Lösung betriebswirtschaftlicher Klassifikationsprobleme mittels Support Vektor Verfahren abzuleiten. Es ist die erklärte Hoffnung des Verfassers, dass dieses Referenzmodell, hinausgehend über einen rein wissenschaftlichen Erkenntnisgewinn, auch einen wertvollen Beitrag für die betriebliche Praxis leistet und durch die weitreichende Automatisierbarkeit sowie vergleichsweise intuitive Verständlichkeit, die Verbreitung von (prognostischem) Data Mining insgesamt begünstigt.

1.7 Literaturverzeichnis

[1] S. Lessmann. Customer relationship management. *WISU - das Wirtschaftsstudium* 32(2), 190–192 (2003)

[2] R. Stahlbock und S. Lessmann. Potential von Support Vektor Maschinen im analytischen Customer Relationship Management. Arbeitspapier, Universität Hamburg (2003)

[3] S. Lessmann. Solving Imbalanced Classification Problems with Support Vector Machines. In: *Proc. of the Intern. Conf. on Artificial Intelligence (IC-AI'04)*, Las Vegas, Nevada, USA, CSREA Press, 214–220 (2004)

[4] S. Lessmann, S. F. Crone, R. Stahlbock und N. Zacher. An Evaluation of Discrete Support Vector Machines for Cost-Sensitive Learning. In: *Proc. of the Intern. Joint Conf. on Neural Networks (IJCNN'06)*, Vancouver, Britisch-Kolumbien, Kanada, IEEE Computer Society, 347–354 (2006)

[5] S. Lessmann, N. Li und S. Voß. A Case Study of Core Vector Machines in Corporate Data Mining. In: *Proc. of the 41st Hawaii Intern. Conf. on System Sciences (HICSS'08)*, Hawaii, USA, IEEE Computer Society, 1–9 (2008)

[6] S. Lessmann und R. Stahlbock. Support Vektor Klassifikatoren im analytischen Kundenbeziehungsmanagement. In: H. Rommelfanger (Hrsg.) *Neue Anwendungen von Fuzzy-Logik und Künstlicher Intelligenz*, Aachen: Shaker Verlag, 113–124 (2005)

[7] S. Lessmann und S. Voß. A framework for customer-centric data mining with support vector machines. *European Journal of Operational Research* – unter Begutachtung – (2007)

[8] S. Lessmann, B. Baesens, C. Mues und S. Pietsch. Benchmarking classification models for software defect prediction: A proposed framework and novel findings. *IEEE Transactions on Software Engineering* – unter Begutachtung – (2007)

[9] S. Lessmann, M.-C. Sung und J. E. V. Johnson. Identifying winners of competitive events: A SVM-based classification model for horserace prediction. *European Journal of Operational Research* – Zur Veröffentlichung angenommen (doi: 10.1016/j.ejor.2008.03.018) – (2008)

[10] S. Lessmann, S. F. Crone und R. Stahlbock. Genetically Constructed Kernels for Support Vector Machines. In: H. D. Haasis, H. Kopfer und J. Schönberger (Hrsg.) *Operations Research Proceedings 2005*, Berlin: Springer, 257–262 (2005)

[11] S. Lessmann, R. Stahlbock und S. F. Crone. Genetic Algorithms for Support Vector Machine Model Selection. In: *Proc. of the Intern. Joint Conf. on Neural Networks (IJCNN'06)*, Vancouver, Britisch-Kolumbien, Kanada, IEEE Computer Society, 3063–3069 (2006)

[12] S. F. Crone, S. Lessmann und S. Pietsch. Forecasting with Computational Intelligence – An Evaluation of Support Vector Regression and Artificial Neural Networks for Time Series Prediction. In: *Proc. of the Intern. Joint Conf. on Neural Networks (IJCNN'06)*, Vancouver, Britisch-Kolumbien, Kanada, IEEE Computer Society, 3159–3166 (2006)

[13] S. F. Crone, S. Lessmann und R. Stahlbock. The impact of preprocessing on data mining: An evaluation of classifier sensitivity in direct marketing. *European Journal of Operational Research* 173(3), 781–800 (2006)

[14] S. F. Crone, S. Lessmann und R. Stahlbock. Empirical Comparison and Evaluation of Classifier Performance for Data Mining in Customer Relationship Management. In: *Proc. of the Intern. Joint Conf. on Neural Networks (IJCNN'04)*, Budapest, Ungarn, IEEE Computer Society, 443–448 (2004)

Kapitel 2

Kumulative Promotion

2.1 Drei thematisch zusammenhängende Fachartikel

Kapitel 1 positioniert die im Rahmen dieser kumulativen Promotion eingereichten Fachartikel in einem gemeinsamen Themenzusammenhang. Der zentrale Fokus des Promotionsvorhabens liegt auf der Anwendung von Support Vektor Maschinen zur Lösung betriebswirtschaftlicher Klassifikationsprobleme sowie dazugehöriger methodischer Erweiterungen. Dazu wurde ein prozessorientierter Ansatz verfolgt, der sich an den Phasen der Wissensentdeckung in Datenbanken orientiert. Dieser interdisziplinäre Charakter ermöglicht verschiedene Einordnungen der einzelnen Beiträge.

So ist eine Fokussierung auf ein bestimmtes Anwendungsproblem und das Design eines geeigneten Lösungsverfahrens insbesondere in:

- S. Lessmann und S. Voß. A framework for customer-centric data mining with support vector machines. *European Journal of Operational Research* – unter Begutachtung – (2007),
- S. Lessmann, M.-C. Sung und J. E. V. Johnson. Identifying winners of competitive events: A SVM-based classification model for horserace prediction. *European Journal of Operational Research*, – Zur Veröffentlichung angenommen (doi: 10.1016/j.ejor.2008.03.018) – (2008),
- S. Lessmann, S. F. Crone, R. Stahlbock und N. Zacher. An Evaluation of Discrete Support Vector Machines for Cost-Sensitive Learning. In: *Proc. of the Intern. Joint Conf. on Neural Networks (IJCNN'06)*, Vancouver, Britisch-Kolumbien, Kanada, IEEE Computer Society, 347–354 (2006),

gegeben.

Darüber hinaus erfolgt eine Konzentration auf diejenigen Anforderungen, welche sich speziell aus dem Kundenbeziehungsmanagement ergeben, repräsentiert durch asymmetrische Klassenverteilungen, klassenspezifische Fehlklassifikationskosten, die Notwendigkeit, über eine reine Klassifikation hinaus, auch das in einem Prognosemodell gekapselte Wissen zu extrahieren sowie algorithmische Anforderungen, welche sich aus der Menge der im Kundenbeziehungsmanagement zu verarbeitenden Daten ergeben, in den Beiträgen:

- S. Lessmann. Solving Imbalanced Classification Problems with Support Vector Machines. In: *Proc. of the Intern. Conf. on Artificial Intelligence (IC-AI'04)*, Las Vegas, Nevada, USA, CSREA Press, 214–220 (2004),
- S. Lessmann, S. F. Crone, R. Stahlbock und N. Zacher. An Evaluation of Discrete Support Vector Machines for Cost-Sensitive Learning. In: *Proc. of the Intern. Joint Conf. on Neural Networks (IJCNN'06)*, Vancouver, Britisch-Kolumbien, Kanada, IEEE Computer Society, 347–354 (2006),
- S. Lessmann und S. Voß. A framework for customer-centric data mining with support vector machines. *European Journal of Operational Research* – unter Begutachtung – (2007),
- S. Lessmann, N. Li und S. Voß. A Case Study of Core Vector Machines in Corporate Data Mining. In: *Proc. of the 41st Hawaii Intern. Conf. on System Sciences (HICSS'08)*, Hawaii, USA, IEEE Computer Society, 1–9 (2008)

Losgelöst von der Verwendung des Support Vektor Verfahrens, welche grundsätzlich in fast allen Arbeiten gegeben ist,[4] besteht ferner ein enger methodischer Zusammenhang zwischen den Beiträgen:

- S. Lessmann, R. Stahlbock und S. F. Crone. Genetic Algorithms for Support Vector Machine Model Selection. In: *Proc. of the Intern. Joint Conf. on Neural Networks (IJCNN'06)*, Vancouver, Britisch-Kolumbien, Kanada, IEEE Computer Society, 3063–3069 (2006),
- S. Lessmann, S. F. Crone, R. Stahlbock und N. Zacher. An Evaluation of Discrete Support Vector Machines for Cost-Sensitive Learning. In: *Proc. of the Intern. Joint Conf. on Neural Networks (IJCNN'06)*, Vancouver, Britisch-Kolumbien, Kanada, IEEE Computer Society, 347–354 (2006),
- S. Lessmann, S. F. Crone und R. Stahlbock. Genetically Constructed Kernels for Support Vector Machines. In: H. D. Haasis, H. Kopfer und J. Schönberger (Hrsg.) *Operations Research Proceedings 2005*, Berlin: Springer, 257–262 (2005),

da jeweils eine Kombination zwischen der ursprünglichen Support Vektor Maschine und sogenannten Meta-Heuristiken entworfen und eingesetzt wird. Mit gewissen Einschränkungen lässt sich behaupten, dass all drei Arbeiten damit dem Bereich *Computational Intelligence* zuzurechnen sind.[5]

Abschließend zeichnet sich die kumulative Dissertation durch den dem Data Mining entsprechenden prozessorientierten Analyseansatz aus, welcher vor allem in den Arbeiten:

---

[4] Einzige Ausnahme ist der Beitrag S. Lessmann. Customer relationship management. *WISU - das Wirtschaftsstudium*, 32(2), 190–192 (2003).

[5] Streng genommen umfasst der Begriff Computational Intelligence lediglich naturanaloge Verfahren wie Künstliche Neuronale Netzwerke, Fuzzy-Verfahren oder Evolutionäre Algorithmen, sowie insbesondere deren Kombination im Sinne einer Hybridisierung. Allerdings bestehen erhebliche Ähnlichkeiten zwischen Support Vektor Maschinen und Neuronalen Netzwerken. Analog existieren Übereinstimmungen zwischen der Tabu-Suche und Evolutionären Algorithmen. Auch die im Computational Intelligence betonte Integration solcher Verfahren ist in allen drei Aufsätzen gegebenen, weswegen die getroffene Einordnung gerechtfertigt erscheint. Dieses Verständnis entspricht auch der wissenschaftlichen Praxis, da in Zeitschriften/Proceedingsbänden zum Thema Computational Intelligence regelmäßig Beiträge zu Support Vektor Maschinen und/oder der Tabu-Suche veröffentlicht werden.

- R. Stahlbock und S. Lessmann. Potential von Support Vektor Maschinen im analytischen Customer Relationship Management. Arbeitspapier, Universität Hamburg (2003).
- S. F. Crone, S. Lessmann und R. Stahlbock. The impact of preprocessing on data mining: An evaluation of classifier sensitivity in direct marketing. *European Journal of Operational Research* 173(3), 781–800 (2006),
- S. Lessmann und S. Voß. A framework for customer-centric data mining with support vector machines. *European Journal of Operational Research* – unter Begutachtung – (2007),
- S. Lessmann, C. Mues, B. Baesens und S. Pietsch. Benchmarking classification models for software defect prediction: A proposed framework and novel findings. *IEEE Transactions on Software Engineering* – unter Begutachtung – (2007),

zum Ausdruck kommt, die jeweils einen der wesentlichen Teilschritte des Prozesses zur Wissensentdeckung in Datenbanken (Problemdefinition, Datenvorverarbeitung, Datenanalyse, Auswertung und Evaluation) in den Mittelpunkt stellen.

2.2 Veröffentlichung von Fachartikeln

Die Veröffentlichung und damit Bereitstellung von Forschungsergebnissen ist eine elementare Notwendigkeit in der wissenschaftlichen Forschung und Lehre. Im Rahmen der vorliegenden Arbeit wurde eine heterogene Verteilung zwischen Zeitschriften und Konferenzen angestrebt, um dem interdisziplinären Charakter der Wirtschaftsinformatik gerecht zu werden. Während wissenschaftliche Zeitschriften in der Betriebswirtschaftslehre als Publikationsmedium präferiert werden, wird in der Informatik auch die Veröffentlichung in Konferenzbänden forciert. Die der Arbeit beigefügten Aufsätze sind wie folgt veröffentlicht, beziehungsweise zur Veröffentlichung eingereicht:

Veröffentlichungen in Zeitschriften

- S. Lessmann, M.-C. Sung und J. E. V. Johnson. Identifying winners of competitive events: A SVM-based classification model for horserace prediction. *European Journal of Operational Research*, – Zur Veröffentlichung angenommen (doi: 10.1016/j.ejor.2008.03.018) – (2008)
- S. F. Crone, S. Lessmann und R. Stahlbock. The impact of preprocessing on data mining: An evaluation of classifier sensitivity in direct marketing. *European Journal of Operational Research*, 173(3), 781–800 (2006)
- S. Lessmann. Customer relationship management. *WISU - das Wirtschaftsstudium*, 32(2), 190–192 (2003)

Beiträge im Begutachtungsprozess

- S. Lessmann und S. Voß. A framework for customer-centric data mining with support vector machines. *European Journal of Operational Research* – unter Begutachtung – (2007)
- S. Lessmann, B. Baesens, C. Mues und S. Pietsch. Benchmarking classification models for software defect prediction: A proposed framework and novel findings. *IEEE Transactions on Software Engineering* – unter Begutachtung – (2007)

Beiträge in Konferenz- und Sammelbänden

- S. Lessmann, N. Li und S. Voß. A Case Study of Core Vector Machines in Corporate Data Mining. In: *Proc. of the 41st Hawaii Intern. Conf. on System Sciences (HICSS'08)*, Hawaii, USA, IEEE Computer Society, 1–9 (2008)
- S. Lessmann, S. F. Crone, R. Stahlbock und N. Zacher. An Evaluation of Discrete Support Vector Machines for Cost-Sensitive Learning. In: *Proc. of the Intern. Joint Conf. on Neural Networks (IJCNN'06)*, Vancouver, Britisch-Kolumbien, Kanada, IEEE Computer Society, 347–354 (2006)
- S. Lessmann, R. Stahlbock und S. F. Crone. Genetic Algorithms for Support Vector Machine Model Selection. In: *Proc. of the Intern. Joint Conf. on Neural Networks (IJCNN'06)*, Vancouver, Britisch-Kolumbien, Kanada, IEEE Computer Society, 3063–3069 (2006)
- S. F. Crone, S. Lessmann und S. Pietsch. Forecasting with Computational Intelligence – An Evaluation of Support Vector Regression and Artificial Neural Networks for Time Series Prediction. In: *Proc. of the Intern. Joint Conf. on Neural Networks (IJCNN'06)*, Vancouver, Britisch-Kolumbien, Kanada, IEEE Computer Society, 3159–3166 (2006)
- S. Lessmann, S. F. Crone und R. Stahlbock. Genetically Constructed Kernels for Support Vector Machines. In: H. D. Haasis, H. Kopfer und J. Schönberger (Hrsg.) *Operations Research Proceedings 2005*, Berlin: Springer, 257–262 (2005)
- S. Lessmann und R. Stahlbock. Support Vektor Klassifikatoren im analytischen Kundenbeziehungsmanagement. In: H. Rommelfanger (Hrsg.) *Neue Anwendungen von Fuzzy-Logik und Künstlicher Intelligenz*, Aachen: Shaker Verlag, 113–124 (2005)
- S. F. Crone, S. Lessmann und R. Stahlbock. Empirical Comparison and Evaluation of Classifier Performance for Data Mining in Customer Relationship Management. In: *Proc. of the Intern. Joint Conf. on Neural Networks (IJCNN'04)*, Budapest, Ungarn, IEEE Computer Society, 443–448 (2004)
- S. Lessmann. Solving Imbalanced Classification Problems with Support Vector Machines. In: *Proc. of the Intern. Conf. on Artificial Intelligence (IC-AI'04)*, Las Vegas, Nevada, USA, CSREA Press, 214–220 (2004)

Arbeitspapier:

- R. Stahlbock und S. Lessmann. Potential von Support Vektor Maschinen im analytischen Customer Relationship Management. Arbeitspapier, Universität Hamburg (2003)

Ein Ranking der Medien variiert in Abhängigkeit der Quellen, Kriterien der Untersuchung sowie zum Teil subjektiven Nuancen. Die nachfolgende Tabelle 1 illustriert die Bewertung der Zeitschriften *European Journal of Operational Research (EJOR)*, *IEEE Transactions on Software Engineering (IEEE TSE)* und *WISU das Wirtschaftsstudium* gemäß der Tabelle von A.-W. Harzing[6], welche mehrere Zeitschriftenrankings zusammenfasst.[7] Eine weitere anerkannte Quelle für die Bewertung von Zeitschriften ist der ISI Journal

[6] Vgl. http://www.harzing.com/download/jql.zip.

[7] Die Zeitschrift WISU wird in der Tabelle von A.-W. Harzing nicht berücksichtigt. Mir ist lediglich eine Evaluation durch den Verband der Hochschullehrer für Betriebswirtschaft bekannt (vgl. http://www.v-h-b.de). Diese ist in Tabelle 1 mit aufgeführt.

Citation Report[8]. Einen Auszug aus den Jahren 2002 – 2006 für die drei relevanten Zeitschriften bietet Tabelle 2.

Tabelle 1: Zeitschriftenrankings nach A.-W. Harzing

Ranking	Bewertungsskala (in aufsteigender Qualität)	EJOR	IEEE TSE	WISU
US98	0,01 – 1	0,43		
Wie01	D; C; B; A; A+	A	A	
UQ03	5, 4, 3, 2, 1	3	1	
VHB03	E, D, C, B, A, A+	A	(B)	E
BJM04	1; 1,5 – 6,5; 7	6		
CNRS04	1; 2; 3; 4; 5	3		
Ess05	4; 3; 2; 1; 0	1	1	
Hkb05	B-; B; B+; A	B	B+	
Theo05	1,2; 1,3 – 94,9; 95	5,23	34,17	
Ast06	1; 2; 3	3	3	
Cra06	1; 2; 3; 4	3		
EJL06	SD; S; P A, P, STAR	P		
ABS07	0; 1; 2; 3; 4	3		

Tabelle 2: ISI Journal Citation Report

	Jahr	Anzahl Artikel	Anzahl Zitationen	Impact Factor	Immediacy Index	Cited Half-life
EJOR	2006	651	8732	0,918	0,237	8,4
	2005	447	6742	0,824	0,201	8,2
	2004	467	6251	0,828	0,137	7,8
	2003	374	4904	0,605	0,11	7,6
	2002	375	4394	0,553	0,083	7,5
IEEE TSE	2006	57	3203	2,132	0,158	>10,0
	2005	67	3165	1,967	0,149	>10,0
	2004	69	3088	1,503	0,333	>10,0
	2003	83	3241	1,730	0,205	>10,0
	2002	76	2479	1,170	0,237	>10,0

[8] Vgl. http://scientific.thomson.com/products/jcr/.

Fachkonferenzen werden in analoger Weise hinsichtlich ihrer Relevanz, Qualität und Reichweite durch wissenschaftliche Institutionen evaluiert. Die im Rahmen der kumulativen Dissertation eingereichten Beiträge wurden auf der *International Joint Conference on Neural Networks (IJCNN)*, der *Hawaii International Conference on System Sciences* (*HICSS*), der *International Conference on Artificial Intelligence (ICAI)* sowie der Jahrestagung der deutschen Gesellschaft für Operations Research (*GOR*) präsentiert. Erstere wird im Rahmen des Computer Science Conference Rankings[9] im Teilgebiet Künstliche Intelligenz und Maschinelles Lernen mit 0,76 von 1,0 Punkten bewertet. Die ICAI erhält hier 0,62 Punkte. Das Ranking der University of Alberta[10] stuft die IJCNN als „Second Tier Conference" (Rang 2) ein; die ICAI wird in diesem Ranking nicht bewertet.

Für die HICSS konnte lediglich eine ältere Einstufung aus dem Jahre 2003 über die Scientific Literature Digital Library (CiteSeer) gefunden werden.[11] Die Konferenz erhält hier 0,33 Punkte, was einer Platzierung in den Top 62,57% entspricht. Allerdings ist zu berücksichtigen, dass diese Quelle Publikationsmedien allgemein, das heißt auch wissenschaftliche Zeitschriften, bewertet.

Für die GOR Tagung liegen keine Bewertung vor.

2.3 Ko-Autorenschaft

Die beigefügten Fachartikel repräsentieren Ergebnisse von Forschungsprojekten und sind auf Grund dessen mit dem Namen aller beteiligten Personen unabhängig des Status (Student, wissenschaftlicher Mitarbeiter, Professor) veröffentlicht beziehungsweise eingereicht worden. Gemäß der in der Promotionsordnung genannten Berechnungsvorschrift (2/(n+1) mit n= Anzahl der Autoren) ergibt sich für die Promotionspunkte (PP) ein Wert von 8,31. Dieser setzt sich gemäß Tabelle 3 wie folgt zusammen:

[9] Vgl. http://www.cs-conference-ranking.org/conferencerankings/topicsii.html.
[10] Vgl. http://www.cs.ualberta.ca/~zaiane/htmldocs/ConfRanking.html.
[11] Vgl. http://citeseer.ist.psu.edu/impact.html.

Tabelle 3: Promotionspunkte

Nr.	Titel	Anzahl Autoren	PP
1.	The impact of preprocessing on data mining: An evaluation of classifier sensitivity in direct marketing	3	0,5
2.	Customer Relationship Management	1	1,0
3.	A framework for customer-centric data mining with support vector machines.	2	0,67
4.	Benchmarking classification models for software defect prediction: A proposed framework and novel findings	4	0,4
5.	Identifying winners of competitive events: Comparing conditional logit and support vector machine classification	3	0,5
6.	A Case Study of Core Vector Machines in Corporate Data Mining	3	0,5
7.	An Evaluation of Discrete Support Vector Machines for Cost-Sensitive Learning	4	0,4
8.	Genetic Algorithms for Support Vector Machine Model Selection	3	0,5
9.	Forecasting with Computational Intelligence - An Evaluation of Support Vector Regression and Artificial Neural Networks for Time Series Prediction	3	0,5
10.	Genetically Constructed Kernels for Support Vector Machines	3	0,5
11.	Support Vektor Klassifikatoren im analytischen Kundenbeziehungsmanagement	2	0,67
12.	Empirical Comparison and Evaluation of Classifier Performance for Data Mining in Customer Relationship Management	3	0,5
13.	Solving Imbalanced Classification Problems with Support Vector Machines	1	1,0
14.	Potential von Support Vektor Maschinen im analytischen Customer Relationship Management	2	0,67
		Summe:	**8,31**

2.4 Substantieller Beitrag des Doktoranden

Die hier eingereichten Fachartikel stellen einen wesentlichen Bestandteil meiner wissenschaftlichen Forschung dar und wurden so ausgewählt, dass ein substantieller eigener Beitrag durchgängig gegeben ist.[12] Dieser wird formal auch durch die überwiegende Erst-Autorenschaft repräsentiert und bezieht sich unter anderem auf die Initiation des Forschungsvorhabens, die Implementierung entsprechender Applikationen und Durchführung empirischer Studien sowie den Anteil an der Verfassung des Aufsatzes.[13] Neben den allein veröffentlichten Beiträgen seien nachfolgend drei Arbeiten angeführt, bei denen sich dieser Anteil besonders abzeichnet:

- S. Lessmann und S. Voß. A framework for customer-centric data mining with support vector machines. *European Journal of Operational Research* – unter Begutachtung – (2007)
- S. Lessmann, R. Stahlbock und S. F. Crone. Genetic Algorithms for Support Vector Machine Model Selection. In: *Proc. of the Intern. Joint Conf. on Neural Networks (IJCNN'06)*, Vancouver, Kanada, IEEE Computer Society, S. 3063–3069 (2006)
- S. Lessmann und R. Stahlbock. Support Vektor Klassifikatoren im analytischen Kundenbeziehungsmanagement. In: H. Rommelfanger (Hrsg.) *Neue Anwendungen von Fuzzy-Logik und Künstlicher Intelligenz*, Aachen: Shaker Verlag, 113–124 (2005)

Keiner der hier eingereichten Beiträge ist zum aktuellen Zeitpunkt Bestandteil eines laufenden oder abgeschlossenen Promotionsvorhabens.

[12] Eine vollständige Publikationsliste findet sich im beigefügten Lebenslauf.

[13] Die Leistung und Qualifikation der Ko-Autoren soll dabei in keiner Weise in Frage gestellt werden.

Teil II

Lebenslauf und Zeugnisse

CURRICULUM VITAE

PROFIL

Stefan Lessmann

Geburtsdatum:	18. März 1975
Familienstand:	verheiratet
Nationalität:	deutsch
Postanschrift:	Alte Königstr. 19 22767 Hamburg
Telefon:	+49.172.4034753
Email:	lessmann@econ.uni-hamburg.de
Homepage	http://iwi.econ.uni-hamburg.de/slessm/

AUSBILDUNG UND STUDIUM

2002 – 2007 PROMOTIONSSTUDIUM, UNIVERSITÄT HAMBURG
Department Wirtschaftswissenschaften
Institut für Wirtschaftsinformatik, Prof. Dr. Dr. h.c. Preßmar

Abschluss: Dr. rer. pol.
Thema: Data Mining mit der Support Vektor Maschine
Note: Summa cum Laude

4/1996 – 9/2001 STUDIUM DER BETRIEBSWIRTSCHAFTSLEHRE, UNIVERSITÄT HAMBURG

Abschluss: Diplom-Kaufmann
Note: 1.55
Schwerpunkte: Wirtschaftsinformatik, Operations Research, Industriebetriebslehre

9/1995 – 3/1996 STUDIUM DER BIOCHEMIE UND MOLEKULARBIOLOGIE, UNIVERSITÄT HAMBURG

7/1994 – 6/1995 WEHRDIENST 2./ PANZERGRENADIERBATAILLON 72

8/1985 – 6/1994 IMMANUEL KANT GYMNASIUM, HAMBURG

Abschluss: Abitur
Note: 1.2

AKADEMISCHE POSITIONEN

6/2006 & 6/2007 & 7/2008	CONFERENCE AND PROGRAM CO-CHAIR, International Conference on Data Mining Las Vegas, Nevada, USA
3/2006 & 9/2006 & 8/2007 & 9/2008	VISITING SCHOLAR, UNIVERSITY OF SOUTHAMPTON School of Management Centre for Risk Research, Prof. Johnson
seit 4/2005	LEHRKRAFT FÜR BESONDERE AUFGABEN, UNIVERSITÄT HAMBURG Department Wirtschaftswissenschaften Institut für Wirtschaftsinformatik, Prof. Voß
1/2002 – 4/2005	WISSENSCHAFTLICHER MITARBEITER, UNIVERSITÄT HAMBURG Department Wirtschaftswissenschaften Institut für Wirtschaftsinformatik, Prof. Voß
10/1999 – 2/2000	TUTOR FÜR MIKROÖKONOMISCHE THEORIE, UNIVERSITÄT HAMBURG Department Wirtschaftswissenschaften Institut für Mikroökonomische Theorie, Prof. Hasenkamp

ERFAHRUNG IN FORSCHUNG UND LEHRE

PUBLIKATIONEN

2008

1. S. Lessmann, M.-C. Sung und J. E. V. Johnson. Identifying winners of competitive events: A SVM-based classification model for horserace prediction. *European Journal of Operational Research* (doi: 10.1016/j.ejor.2008.03.018) (2008)
2. N. Martin, S. Lessmann, S. Voß. Crowdsourcing: Systematisierung praktischer Ausprägungen und verwandter Konzepte. In: M. Bichler, T. Hess, H. Krcmar, U. Lechner, F. Matthes, A. Picot, B. Speitkamp, P. Wolf (Hrsg.) Multikonferenz Wirtschaftsinformatik 2008, Berlin: Gito, 1251–1263 (2008).
3. S. Lessmann, N. Li, S. Voß. A Case Study of Core Vector Machines in Corporate Data Mining. In: *Proc. of the Hawaii Intern. Conf. on System Sciences (HICSS'08),* Waikoloa, Hawaii, USA, IEEE Computer Society, S. 1–9 (2008)

2007

4. S. Lessmann, B. Baesens, C. Mues und S. Pietsch. Benchmarking classification models for software defect prediction: A proposed framework and novel findings. *IEEE Transactions on Software Engineering* – unter Begutachtung – (2007)
5. S. Lessmann und S. Voß. A framework for customer-centric data mining with support vector machines. *European Journal of Operational Research* – unter Begutachtung – (2007)
6. S. Lessmann, M.-C. Sung und J.E.V. Johnson. Adapting Least-Square Support Vector Regression Models to Forecast the Outcome of Horseraces, *Journal of Prediction Markets* 1(3), 169-187 (2007)
7. S. F. Crone, S. Lessmann und R. Stahlbock (Hrsg.). *DMIN'07 – Proceedings of International Conference on Data Mining*. Las Vegas: CSREA Press, 2007.
8. J.E.V. Johnson, S. Lessmann, M.-C. Sung. A new Method for Predicting the Outcome of Speculative Events. Arbeitspapier CRR-07-03, Centre for Risk Research, Universität Southampton (2007)

2006

9. S. F. Crone, S. Lessmann und R. Stahlbock. The impact of preprocessing on data mining: An evaluation of classifier sensitivity in direct marketing. *European Journal of Operational Research* 173(3), 781–800 (2006)
10. S. F. Crone, S. Lessmann und R. Stahlbock (Hrsg.). *DMIN'06 – Proceedings of International Conference on Data Mining*. Las Vegas: CSREA Press, 2006.
11. S. F. Crone, S. Lessmann und S. Pietsch. Forecasting with Computational Intelligence – An Evaluation of Support Vector Regression and Artificial Neural Networks for Time Series Prediction. In: *Proc. of the Intern. Joint Conf. on Neural Networks (IJCNN'06)*, Vancouver, Britisch-Kolumbien, Kanada, IEEE Computer Society, S. 3159–3166 (2006)
12. S. F. Crone, S. Lessmann und S. Pietsch. Parameter Sensitivity of Support Vector Regression and Neural Networks for Forecasting. In: *Proc. of the Intern. Conf. on Data Mining (DMIN'06)*, Las Vegas, Nevada, USA, CSREA Press, 396–402 (2006)
13. S. Lessmann, S. F. Crone, R. Stahlbock und N. Zacher. An Evaluation of Discrete Support Vector Machines for Cost-Sensitive Learning. In: *Proc. of the Intern. Joint Conf. on Neural Networks (IJCNN'06)*, Vancouver, Britisch-Kolumbien, Kanada, IEEE Computer Society, 347–354 (2006)
14. S. Lessmann, R. Stahlbock und S. F. Crone. Genetic Algorithms for Support Vector Machine Model Selection. In:

Proc. of the Intern. Joint Conf. on Neural Networks (IJCNN'06), Vancouver, Britisch-Kolumbien, Kanada, IEEE Computer Society, 3063–3069 (2006)

15. S. Lessmann und S. Voß. Solving discrete support vector machines with tabu search. In: *INFORMS 2006 Workshop of Artificial Intelligence and Data Mining*, Pittsburg, Pennsylvania, USA, INFORMS (2006)

2005

16. S. Lessmann, S. F. Crone und R. Stahlbock. Genetically Constructed Kernels for Support Vector Machines. In: H. D. Haasis, H. Kopfer und J. Schönberger (Hrsg.) *Operations Research Proceedings 2005*, Berlin: Springer, 257–262 (2005)
17. S. Lessmann und R. Stahlbock. Support Vektor Klassifikatoren im analytischen Kundenbeziehungsmanagement. In: H. Rommelfanger (Hrsg.) *Neue Anwendungen von Fuzzy-Logik und Künstlicher Intelligenz*, Aachen: Shaker Verlag, 113–124 (2005)
18. S. F. Crone, S. Lessmann und R. Stahlbock. Support Vector Machines versus Artificial Neural Networks - New Potential in Data Mining for Customer Relationship Management? In: D. Wang und N. K. Lee (Hrsg.) *Neural Networks Applications in Information Technology and Web Engineering*, Sarawak: Borneo Publishing Co., 80–93 (2005)
19. S. Lessmann und S. Voß. Electronic-Procurement. In: S. G. Häberle (Hrsg.) *Lexikon der Betriebswirtschaftslehre.* – zur Veröffentlichung angenommen – (2005)
20. S. F. Crone, S. Lessmann und R. Stahlbock. Utility Based Data Mining for Time Series Analysis – Cost Sensitive Learning for Neural Network Predictors. In: *Proc. of the 1st ACM SIGKDD Workshop on Utility based Data Mining (UBDM@KDD'05)*, Chicago, Illinois, USA, ACM Press, 59–68 (2005)
21. S. Lessmann, R. Stahlbock und S. F. Crone. Optimizing Hyperparameters of Support Vector Machines by Genetic Algorithms. In: *Proc. of the Intern. Conf. on Artificial Intelligence (IC-AI'05)*, Las Vegas, Nevada, USA, CSREA Press, 74–80 (2005)
22. R. Stahlbock, S. Lessmann und S. F. Crone. Evolutionary Neural Classification Approaches for Strategic and Operational Decision Support in Retail Store Planning. In: *Proc. of the Intern. Conf. on Artificial Intelligence (IC-AI'05)*, Las Vegas, Nevada, USA, CSREA Press, 60–66 (2005)

2004

23. S. F. Crone, S. Lessmann und R. Stahlbock. Empirical Comparison and Evaluation of Classifier Performance for Data Mining in Customer Relationship Management. In: *Proc. of the Intern. Joint Conf. on Neural Networks (IJCNN'04)*,

Budapest, Ungarn, IEEE Computer Society, 443–448 (2004)

24. S. Lessmann. Solving Imbalanced Classification Problems with Support Vector Machines. In: *Proc. of the Intern. Conf. on Artificial Intelligence (IC-AI'04)*, Las Vegas, Nevada, USA, CSREA Press, 214–220 (2004)

2003

25. S. Lessmann. Customer relationship management. *WISU - das Wirtschaftsstudium* 32(2), 190–192 (2003)
26. R. Stahlbock und S. Lessmann. Potential von Support Vektor Maschinen im analytischen Customer Relationship Management. Arbeitspapier, Universität Hamburg (2003)

KONFERENZVORTRÄGE

2007

1. Benchmarking classification algorithms for software defect prediction. *Operations Research 2007*, 05. – 07. September, Saarbrücken, Deutschland.
2. Repeat purchase modelling with transductive support vector machines. *EURO XXII*, 08. – 11. Juli, Prag, Tschechien.

2006

3. Genetic algorithms for support vector machine model selection. *OR 48*, 11. – 13. September, Bath, England.
4. Forecasting with computational intelligence. *OR 48*, 11. – 13. September, Bath, England.
5. An evaluation of discrete support vector machines for cost-sensitive learning. *Intern. Joint Conf. on Neural Networks*, 16. – 21. Juli, Vancouver, Britisch-Kolumbien, Kanada.
6. Discrete support vector machines. *EURO XXI*, 02. – 05. Juli, Reykjavik, Island.
7. Modelling classification analysis for competitive events with applications to sports betting. *Workshop on Virtual Environments for Advanced Modelling*, 06. – 07. Juni, Hamburg, Deutschland.
8. Some steps towards a reference model for support vector machine based decision support in customer relationship management. *CORMSIS - Centre for OR, Management Science and Information Systems*, 30. März, Southampton, England.

2005

9. Genetically constructed kernels for support vector machines. *Operations Research 2005*, 06. – 09. September, Bremen, Deutschland.
10. Support Vektor Klassifikatoren im analytischen Kundenbeziehungsmanagement. *GOR-Workshop Fuzzy Sets, Neuronale Netze und Künstliche Intelligenz*, 21. Februar,

Frankfurt a. M., Deutschland.

2004

11. Empirical comparison and evaluation of classifier performance for data mining in customer relationship management. *Intern. Joint Conf. on Neural Networks*, 25. – 29. Juli, Budapest, Ungarn.
12. Support vector machines and artificial neural networks. *EURO Summer Institute*, 09. – 23. Juli 2004, Ankara, Türkei.
13. Solving imbalanced classification problems with support vector machines. *Intern. Conf. on Artificial Intelligence*, 21. – 25. Juni, Las Vegas, Nevada, USA.

GUTACHTERTÄTIGKEIT

- INFORMS Journal on Computing
- European Journal of Operational Research
- IEEE Transactions on Knowledge and Data Engineering
- Annals of Operations Research
- Empirical Software Engineering
- Journal on Data and Knowledge Engineering
- Netnomics
- Soft Computing Journal
- Intern. Conf. on Data Mining (DMIN'07)
- Intern. Joint Conf. on Neural Networks (IJCNN'07)
- Hawaiian Intern. Conf. on System Science (HICSS'07)
- Intern. Conf. on Data Mining (DMIN'06)
- Intern. Conf. on Artificial Intelligence (ICAI'05)
- Hawaiian Intern. Conf. on System Science (HICSS'05)
- Intern. Conf. on Artificial Intelligence (ICAI'04)

MITGLIEDSCHAFT IM ORGANISATIONSKOMITEE WISSENSCHAFTLICHER KONFERENZEN

- 2008 Inter. Conf. on Data Mining
- 18th Intern. Conf. on Information Resources Management
- 2007 Inter. Conf. on Data Mining
- 2007 Portuguese Conf. on Artificial Intelligence
- 2006 Intern. Conf. on Data Mining

MITGLIEDSCHAFTEN IN WISSENSCHAFTLICHEN VERBÄNDEN

GOR – Gesellschaft für Operations Research

GI – Gesellschaft für Informatik

OR Society

INFORMS

INFORMS Section on Data Mining

IEEE Data Mining Technical Committee

IEEE Computational Intelligence Society

BETREUUNG UND INITIIERUNG VON FORSCHUNGSKOOPERATIONEN

2007 STRATEGISCHES VERTRIEBSCONTROLLING MIT DER BSC
MPC Capital AG — Projekbetreuung.

Entwicklung eines Kennzahlensystems für das strategische Vertriebscontrolling mit der Balanced Scorecard

2005 IT-VALUE MANAGEMENT
Detecon Consulting— Projektleitung.

Entwicklung eines Rahmenwerks für das IT-Portfoliomanagement auf der Basis des *Economic value edit (EVA)*

2004 DATA MINING ZUR STORNOPROPHYLAXE IM VERSICHERUNGSWESEN
Hamburg-Manheimer— Projektleitung.

Bewertung verschiedener Data Mining Verfahren sowie korrespondierender Standardsoftware zur Prognose von Kündigungswahrscheinlichkeiten von Lebensversicherungen.

2003 ZIELGRUPPENSELEKTION IM DIREKTMARKETING
Gruner & Jahr AG— BI-Berater.

Studie zur Bewertung innovativer Data Mining Methoden im Zeitschriftenmarketing einschließlich Kosten-/Nutzenanalyse und Vergleich mit Standardsoftwaresystemen.

STORNOANALYSE IM INTERNET SERVICE PROVIDING
AOL Deutschland— Projektabwicklung und -dokumentation.

Evaluation von *Soft-Computing* Verfahren zur Prognose kundenindividueller Abwanderungswahrscheinlichkeiten.

2002 KUNDENBEZIEHUNGSMANAGEMENT IM FINANZDIENSTLEISTUNGSSEKTOR

IBM Business Consulting Services— Projektleitung.

Durchführung einer empirischen Studie zur Erhebung und Bewertung von Kundenbeziehungsmanagementaktivitäten bei Banken und Versicherungen.

LEHRERFAHRUNG

seit 2002 *Durchgeführte Lehrveranstaltungen im Grundstudium:*

- Rechnerpraktikum für Wirtschaftswissenschaftler
- Objektorientierte Programmierung mit Visual Basic .Net

Unterstützung von Vorlesungen und Seminaren im Hauptstudium:

- Computergestützte Planung
- Informationsmanagement
- Planung und Entwurf betrieblicher Informationssysteme
- Produktion und Supply Chain Management
- Seminar zum Informationsmanagement
- Seminar zum Innovationsmanagement
- Seminar zur ökonomischen Bewertung von Informationssystemen
- Seminar zur ökonomischen Evaluation wissenschaftlichen Wirkens
- Seminar zur Wirtschaftsinformatik

Selbstständig konzipierte und durchgeführte Lehrveranstaltungen:

- Entwicklung webbasierter Anwendungssysteme
- Projektseminar: Entwicklung von Web-Anwendungen mit ASP.Net

Unabhängige Betreuung von Studierenden im Hauptstudium:

- Entwurf und Korrektur von 50+ Hausarbeiten und Seminarvorträgen
- Entwurf, Betreuung und Begutachtung von 15+ Diplom-/ Studienarbeiten in Betriebswirtschaftslehre und Wirtschaftsinformatik
- Konzeption und Korrektur von Klausuren in den o.g. Lehrveranstaltungen

PRAXISERFAHRUNG

BERUFSERFAHRUNG

seit 2003	BI³S LAB Gründungsmitglied und Geschäftsführer des *Business Intelligence Laboratory* (Spin-off des Instituts für Wirtschaftsinformatik für Technologietransfer und Praxiskooperationen). Aufgabengebiete: - Data Mining und Business Intelligence Beratung - Vorträge auf Fachkonferenzen - Erstellung von Studien und Gutachten
1997 – 2002	IT - SERVICE LESSMANN Tätigkeit als selbstständiger Softwareentwickler und IT-Berater mit Schwerpunkt Arbeitseinsatzplanung. Kunden (u.a.): - Initions AG - Pro-Medisoft GmbH - Pharma Card Beratungs GmbH
1997 – 1999	GRUNER & JAHR AG Stipendiat im Bereich IT-Anwendungen. Aufgabengebiete: - Design und Erstellung von Webauftritten - System- und Anwendungsprogrammierung - Entwicklung und Betreuung WiSo Börsenspiel

SOFTWAREKENNTNISSE UND SONTIGE FÄHIGKEITEN

- Anwendungssysteme:
  - Betriebswirtschaftliche SSW: SAP /R3, Navision
  - Office Produkte: komplette MS Office Reihe
  - Back Office: SQL-Server, BizTalk Server
  - DW und OLAP: Cognos, Analysis Services
  - Data Mining: SAS Enterprise Miner, SPSS Clementine, WEKA, YALE, Matlab
- BI und Data Mining:
  - Methoden: Support Vektor Maschinen, Neuronale Netze, Entscheidungsbaumverfahren, Logistische Regression, Ensembles
  - Prozesse: Vorverarbitung, Modellselektion, Parametrisierung, Modellevaluation und -vergleich
- Softwareentwicklung:
  - Modellierung: OO-Design, UML, ER-Modellierung
  - Sprachen: C#, Java, Perl, Visual Basic, VB.Net
  - Web-Entwicklung: ASP.Net, HTML, Java Script
- Sprachkenntnisse:
  - Deutsch: Muttersprache
  - Englisch: verhandlungssicher
  - Französisch: Grundkenntnisse

Hamburg, den 23. April 2008

AKADEMISCHE REFERENZEN

DERZEITIGER ARBEITGEBER

Prof. Dr. S. Voß
Institut für Wirtschaftsinformatik
Department für Wirtschaftswissenschaften
Universität Hamburg
Von-Melle-Park 5, D-20146 Hamburg
Deutschland
Email: Stefan.voss@uni-hamburg.de

DOKTORVATER

Prof. Dr. Dr. h.c. D. B. Preßmar
Institut für Wirtschaftsinformatik
Department für Wirtschaftswissenschaften
Universität Hamburg
Von-Melle-Park 5, D-20146 Hamburg
Deutschland
Email: pressmar@econ.uni-hamburg.de

KOOPERATIONSPARTNER

Prof. Dr. J. E. V. Johnson
Centre for Risk Research
School of Management
University of Southampton
Highfield, Southampton SO17 1BJ
United Kingdom
Email: jej@soton.ac.uk

UNIVERSITÄT HAMBURG

Zwischenprüfung

für Studierende der Fachrichtung Betriebswirtschaftslehre

PRÜFUNGSZEUGNIS

Stefan Lessmann, geboren am 18.03.1975 in Hamburg,
hat sich gemäß der Diplom-Prüfungsordnung für den Studiengang Betriebswirtschaftslehre vom 28. November 1984 in der Fassung vom 29.11.1995 der Zwischenprüfung unterzogen und die Prüfung bestanden.
Folgende Leistungsnachweise über die erfolgreiche Teilnahme an Klausurarbeiten von jeweils insgesamt vierstündiger Dauer sind vorgelegt worden:

Prüfungsgebiete:	Note:
1. Grundzüge des betrieblichen Rechnungswesens	gut
2. Grundzüge der Betriebswirtschaftslehre	gut
3. Grundzüge der Volkswirtschaftslehre	gut
4. Recht der Wirtschaft	sehr gut
5. Mathematik für Wirtschaftswissenschaftler	befriedigend
6. Statistik	sehr gut

Hamburg, den 18.12.1997

Universität Hamburg Lessmann

Der Vorsitzende des Prüfungsausschusses
für Diplom-Kaufleute

Notensklala: 1 = sehr gut; 2 = gut; 3 = befriedigend; 4 = ausreichend

UNIVERSITÄT HAMBURG

Prüfungszeugnis

Stefan Lessmann

geboren am 18.03.1975 in Hamburg

hat die

Diplomprüfung für Kaufleute

mit der Gesamtnote **gut (1,55)** am 14.09.2001 bestanden.

In den einzelnen Prüfungsfächern wurden folgende Noten erzielt:

Allg. Betriebswirtschaftslehre Prof. Dr. Hansmann	sehr gut (1,4)
Volkswirtschaftslehre Prof. Dr. Stahlecker	gut (1,9)
Industriebetriebslehre Prof. Dr. Hansmann	sehr gut (1,4)
Betriebsw. Datenverarbeitung Prof. Dr. Preßmar	sehr gut (1,0)
Unternehmensforschung Prof. Dr. Hansen	befriedigend (2,6)

Thema der Diplomarbeit:

Integration von Geschäftsanwendungen mit Hilfe von XML-Technologie und Biztalk Server

Prüfer: Prof. Dr. Preßmar
Note: sehr gut (1,0)

Hamburg, den 07.01.2002

Der Vorsitzende des Prüfungsausschusses für Diplom-Kaufleute

Lessmann

Universität Hamburg

Notenskala: bis 1,5 sehr gut, über 1,5 bis 2,5 gut, über 2,5 bis 3,5 befriedigend, über 3,5 bis 4,0 ausreichend, über 4,0 nicht ausreichend.

UNIVERSITÄT HAMBURG

DIPLOM

Stefan Lessmann

geboren am 18.03.1975 in Hamburg

hat am 14.09.2001 die Diplomprüfung für Kaufleute gemäß Diplom-Prüfungsordnung bestanden.

Auf Grund dieser Prüfung wird ihm der akademische Grad

Diplom-Kaufmann

verliehen.

HAMBURG, den 07.01.2002

Universität Hamburg

Der Vorsitzende des Prüfungsausschusses
für Diplom-Kaufleute

FREIE UND HANSESTADT HAMBURG

IMMANUEL-KANT-GYMNASIUM

ZEUGNIS
DER ALLGEMEINEN HOCHSCHULREIFE

Stefan Lessmann

geboren am 18. 03. 1975 in Hamburg

hat nach dem Besuch der gymnasialen Oberstufe die Abiturprüfung abgelegt.

Dem Zeugnis liegen zugrunde:

Die Ausbildungs- und Prüfungsordnung der gymnasialen Oberstufe vom 15.5.1990 (Hamburgisches Gesetz- und Verordnungsblatt Seite 93) in der jeweils geltenden Fassung,

die „Vereinbarung zur Neugestaltung der gymnasialen Oberstufe in der Sekundarstufe II" (Beschluß der Kultusministerkonferenz vom 11.4.1988) in der jeweils geltenden Fassung,

die „Vereinbarung über die Abiturprüfung der neugestalteten gymnasialen Oberstufe in der Sekundarstufe II (gemäß Vereinbarung der Kultusministerkonferenz vom 7.7.1972)" (Beschluß der Kultusministerkonferenz vom 13.12.1973) in der jeweils geltenden Fassung.

I. Leistungen in den Kursen der Studienstufe

Stefan Lessmann
Name

Fach	Bewertung Punktzahlen der einzelnen Kurse in einfacher Wertung			
	1. Halbjahr	2. Halbjahr	3. Halbjahr	4. Halbjahr
Sprachlich-literarisch-künstlerisches Aufgabenfeld				
Deutsch	13	12	12	14
Fremdsprachen (weitergeführt) Englisch	12	13	13	12
	--	--	--	--
Fremdsprache (neu aufgenommen)	--	--	--	--
Bildende Kunst	--	--	--	--
Musik	--	--	--	--
Darstellendes Spiel	13	14	13	15
	--	--	--	--
Gesellschaftswissenschaftliches Aufgabenfeld				
Gemeinschaftskunde	12	13	13	14
Erdkunde	--	--	--	--
Geschichte	(09)	(10)	11	(10)
Wirtschaft	--	--	--	--
Religion	--	--	--	--
Philosophie	(09)	13	14	13
	--	--	--	--
	--	--	--	--
Mathematisch-naturwissenschaftlich-technisches Aufgabenfeld				
Mathematik LF	11	14	15	14
Biologie	--	--	--	--
Chemie LF	15	15	15	15
Physik	(10)	11	--	--
Informatik	--	--	--	--
	--	--	--	--
	--	--	--	--
	--	--	--	--
	--	--	--	--
Sport	13	15	(12)	14

(Leistungsfächer werden mit „ LF " gekennzeichnet. Die Bewertungen von Kursen, die nicht in die Gesamtqualifikation eingehen, sind in Klammern gesetzt.)

Stefan Lessmann
Name

II. Leistungen in der Abiturprüfung

	Prüfungsfach	Punktzahlen in einfacher Wertung	
		schriftliche Prüfung	mündliche Prüfung [1]
1. LF	Mathematik	11	--
2. LF	Chemie	14	--
3.	Gemeinschaftskunde	14	--
4.	Englisch		13

III. Berechnung der Gesamtqualifikation und der Durchschnittsnote

Gesamtpunktzahl aus 22 Grundkursen in einfacher Wertung:	286	mindestens 110 höchstens 330 Punkte
Gesamtpunktzahl aus 6 Leistungskursen in doppelter Wertung (1. – 3. Halbjahr) und aus 2 Leistungskursen in einfacher Wertung (4. Halbjahr):	199	mindestens 70 höchstens 210 Punkte
Gesamtpunktzahl aus den Prüfungen in vierfacher Wertung und aus den Kursen der Prüfungsfächer im 4. Halbjahr in einfacher Wertung:	263	mindestens 100 höchstens 300 Punkte
Summe der Gesamtpunktzahlen:	748	mindestens 280 höchstens 840 Punkte
Durchschnittsnote:	1,2	

IV. Fremdsprachen

(ohne Arbeitsgemeinschaften)

In der ersten Fremdsprache Englisch und in der zweiten Fremdsprache Französisch ist Unterricht in dem für den Erwerb der allgemeinen Hochschulreife erforderlichen Umfang besucht worden.

In der dritten Fremdsprache -- ist Unterricht in/von Klasse/Jahrgangsstufe -- bis -- besucht worden. [2]

In der vierten Fremdsprache -- ist Unterricht in/von Klasse/Jahrgangsstufe -- bis -- besucht worden. [2]

Dieses Zeugnis enthält das Latinum / Großes Latinum und das Graecum. [2]

[1] Falls Sport Prüfungsfach ist, auch die Punktzahl der praktischen Prüfung.
[2] Nichtzutreffendes bitte streichen.

V. Bemerkungen

Stefan Lessmann
Name

VI. ~~Frau~~/ Herr Lessmann

hat mit der Ablegung der Abiturprüfung die Befähigung zum Studium an einer Hochschule in der Bundesrepublik Deutschland erworben.

Hamburg, den 14. Juni 94

Immanuel-Kant-Gymnasium Siegel Hamburg

Prüfungsbeauftragte/r

Schulleiter / in / Abteilungsleiter / in

Für die Umrechnung der Noten in Punkte gilt der folgende Schlüssel:

Noten	sehr gut 1			gut 2			befriedigend 3			ausreichend 4			mangelhaft 5			ungenügend 6
	+		–	+		–	+		–	+		–	+		–	
Punkte	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Teil III

Literatur

Identifying winners of competitive events: A SVM-based classification model for horserace prediction

Stefan Lessmann[a,*], Ming-Chien Sung[b], Johnnie E.V. Johnson[b]

[a] *Institute of Information Systems, University of Hamburg, Von-Melle-Park 5, 20146 Hamburg, Germany*

[b] *Centre for Risk Research, School of Management, University of Southampton, Highfield, Southampton, SO17 1BJ, UK*

Abstract

The aim of much horserace modelling is to appraise the informational efficiency of betting markets. The prevailing approach involves forecasting the runners' finish positions by means of discrete or continuous response regression models. However, theoretical considerations and empirical evidence suggest that the information contained within finish positions might be unreliable, especially among minor placings. To alleviate this problem, a classification-based modelling paradigm is proposed which relies only on data distinguishing winners and losers. To assess its effectiveness, an empirical experiment is conducted using data from a UK racetrack. The results demonstrate that the classification-based model compares favourably with state-of-the-art alternatives and confirm the reservations of relying on rank ordered finishing data. Simulations are conducted to further explore the origin of the model's success by evaluating the marginal contribution of its constituent parts.

Keywords: Forecasting, Decision analysis, Finance, Horseracing, Support Vector Machines

1. Introduction

The rationality of traders' collective decisions in financial markets is explored by assessing the extent to which they discount information in market prices. However, financial markets are complex and, in order to shed light on investors' use of information, researchers often turn to simpler financial markets where the pricing problem is reduced. In particular, many studies explore horserace betting markets because they share many features in common with wider financial markets, including a large number of participants and a wide range of factors which can influence a horse's (asset's) prospects (Hausch and Ziemba, 1985; Johnson et al., 2006; Law and Peel, 2002; Levitt, 2004; Sauer, 1998; Schnytzer and Shilony, 1995; Vaughan Williams, 1999). In addition, betting markets offer an important advantage over wider financial markets; namely, they generate an unequivocal outcome (a winner) and an associated rate of

[*] Corresponding author: Tel.: +49-40-42838-4706, Fax: +49-40-42838-5535.

E-mail addresses: lessmann@econ.uni-hamburg.de; {jej; ms9}@soton.ac.uk

return within a finite time frame (Law and Peel, 2002), and hence provide an objective benchmark against which to measure the quality of an investment decision (i.e. a bet). 'As a result, wagering markets can provide a clear view of pricing issues which are more complicated elsewhere' (Sauer, 1998 p. 2021) and the value of studying bettors' decisions is reinforced by the fact that these markets are, in themselves, important. For example, the turnover of the UK horserace betting market in 2006 was £15,500 million.

Predictive modelling is often employed when assessing the degree to which bettors efficiently use information when making their investment decisions. In particular, models, incorporating variables based on publicly available information, are employed to estimate horses' chances of winning. If these estimates enable profitable betting over a number of future races it may be concluded that bettors do not fully discount information concerning the attributes contained in the model (e.g., Benter, 1994; Bolton and Chapman, 1986; Johnson et al., 2006; Sung et al., 2005).

It has been shown that in forecasting the winner of a race it is important to account for the relative strength of competitors; referred to as within-race competition (Bolton and Chapman, 1986). Conditional logit models (McFadden, 1973) have been proposed for this task since, unlike ordinary logistic regression which considers each horse in isolation from the race, conditional logit (CL) models a race as an entity and consequently maintains the relationship among the competing runners (see, e.g., Bolton and Chapman, 1986; Chapman, 1994; Gu, Huang and Benter, 2003). Recently, Edelman (2007) showed that the predictive accuracy of such models can be further improved if they are used in conjunction with modern machine learning methods. His approach is based on Benter's (1994) *two-stage philosophy* and utilises Support Vector Regression (SVR) to model the relationship between (a) fundamental variables which are associated with horses' recent performances and factors relating to the current race (e.g., prize money, weight carried), and (b) horses' finish position. The resulting forecasts are combined with the horses' final odds by means of CL in a second step. CL and SVR complement each other in the sense that the latter accounts for within-race competition whereas SVR accommodates a large number of potentially correlated variables with low risk of overfitting and *automatically* models complex non-linear relationships between attributes in a data-driven manner.

This paper develops a forecasting model which adopts the two-stage modelling approach. However, whereas previous work in horserace forecasting focuses predominantly on regression methods (e.g., Benter, 2003; Edelman, 2007), the model proposed here embodies support vector machines (SVMs) for classifying race results. As explained later in the paper, theoreti-

cal considerations as well as empirical results (Sung and Johnson, 2007) cast doubt on the reliability of a key ingredient of regression-based modelling, namely, rank order finishing data. Taking a classification approach, modelling focuses on distinguishing winning and non-winning horses and avoids excessive use of potentially corrupted rank orderings, especially among minor placings. In addition, a novel data pre-processing method is suggested to introduce some notion of within-race competition in the first modelling stage.

The major objectives of the paper are, therefore, to examine the effectiveness of the proposed classification-based methodology for horserace prediction and to shed light on the marginal contribution of the elements of this complex two-stage model. To that end, an empirical evaluation is conducted to contrast the predictive performance of the proposed SVM-based modelling technique with highly competitive benchmarks (i.e. Edelman, 2007; Sung et al., 2005). Subsequently, the components of the forecasting model (the hierarchical two-step approach, the novel data pre-processing technique, and non-linear modelling) are evaluated individually to confirm their appropriateness.

The remainder of the paper is organised as follows: The theory of SVMs is reviewed before describing the particularities of horserace modelling and the details of the two-stage classification-based approach. Subsequently, results of the empirical evaluation are presented and conclusions are drawn.

2. Support vector machines for classification

The SVM is a machine learning technique that facilitates linear and non-linear binary classification. Given a sample, $S = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{M}$, with $\boldsymbol{x}_i \in \mathrm{X} \subseteq R^N$ being a vector of N measurements, $y_i \in \{-1, +1\}$ the corresponding class label, and M denoting the number of observations, SVMs infer (*learn*) from the data a functional model, $f_\Lambda(\boldsymbol{x}) : X \mapsto \{-1, +1\}$. This enables estimation of the class membership of novel examples (i.e. observations not contained in S). The vector Λ includes the parameters of the classifier which are fitted on S in a model building stage (*classifier training*).

SVMs are inspired by statistical learning theory (Vapnik, 1995). To derive a classification model from S, they implement the concept of a maximal margin separation. That is, they strive to maximise the distance between examples that are closest to a linear decision surface separating the two classes (Cristianini and Shawe-Taylor, 2000). It can be shown that by maximising this margin, a bound on the generalisation error, i.e. the error on future data, is minimised (Vapnik, 1995).

To construct a linear classifier with maximal margin, the norm of the corresponding hyperplane's weight vector, $\boldsymbol{w}$, has to be minimised, subject to the constraint that training examples of each class reside on opposite sides of the separating surface (see Figure 1). With $y_i \in \{-1,+1\}$, this constraint can be formulated as (e.g., Burges, 1998):

$$y_i((\boldsymbol{w}\cdot\boldsymbol{x}_i)+b)\geq 1,\ \ i=1,...,M\ . \tag{1}$$

Examples which satisfy (1) with equality are called support vectors as they define the orientation of the resulting hyperplane.

[Figure 1 about here]

To account for misclassifications (i.e. examples violating (1)), the soft margin formulation (e.g., Cristianini and Shawe-Taylor, 2000) introduces continuous slack variables, ξ_i. Hence, to build a maximal margin SVM classifier, the following convex quadratic programming problem has to be solved:

$$\begin{aligned} &\min_{w,b,\xi} \frac{1}{2}\|\boldsymbol{w}\| + C\sum_{i=1}^{M}\xi_i \\ &s.t.:\ y_i((\boldsymbol{w}\cdot\boldsymbol{x}_i)+b)\geq 1-\xi_i,\ \ i=1,...,M. \end{aligned} \tag{2}$$

The primal decision variables $\boldsymbol{w}$ and b define the separating hyperplane, so that the resulting classifier takes the form:

$$f_{\Lambda=\{\boldsymbol{w},b\}}(\boldsymbol{x}) = sign\left((\boldsymbol{w}^*\cdot\boldsymbol{x})+b^*\right), \tag{3}$$

where $\boldsymbol{w}^*$ and b^* represent the solution of (2).

To construct more general non-linear decision surfaces, SVMs map the input data into a high-dimensional feature space via an a priori chosen mapping function Φ. Constructing a separating hyperplane in this feature space leads to a non-linear decision boundary in the input space (Vapnik, 1995). The capability of SVMs to disclose non-linear relationships among input variables by projecting the data into a feature space of higher dimension has been demonstrated on several well known benchmarking datasets (e.g., Van Gestel et al., 2004). For example, standard non-linear classification tasks like the XOR problem, the 2-spiral problem or the classification of a chess board into black and white regions are solved with SVMs (see Cui and Curry, 2005; Suykens and Vandewalle, 1999; Cristianini and Shawe-Taylor, 2000).

The mapping of the data is accomplished implicitly to avoid resource intensive calculations in the transformed feature space. Consider the dual of (2), with α_i denoting the Lagrangian multipliers (e.g., Burges, 1998; Vapnik, 1995):

$$\begin{aligned} &\max_{\alpha} = \sum_{i=1}^{M} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{M} \alpha_i \alpha_j y_i y_j (\boldsymbol{x}_i \cdot \boldsymbol{x}_j) \\ &s.t. \quad \sum_{i=1}^{M} \alpha_i y_i = 0 \quad ; \quad 0 \le \alpha_i \le C \quad \forall i = 1,...,M. \end{aligned} \tag{4}$$

As (4) contains the input data only in form of scalar products a so-called kernel function, *K*, can be employed to compute the scalar product of the transformed vectors directly in the input space:

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \Phi(\boldsymbol{x}_i) \cdot \Phi(\boldsymbol{x}_j) . \tag{5}$$

The kernel can be regarded as a proximity function measuring the distance between two input vectors in the non-linearly transformed feature space. The resulting classifier is

$$\begin{aligned} &f_{\Lambda=\{\boldsymbol{\alpha},b\}}(\boldsymbol{x}) = sign\left(\left(\sum_{i \in SV} \alpha_i y_i K(\boldsymbol{x}_i, \boldsymbol{x}) \right) + b \right) \\ &\text{with } SV = \left\{ i \mid \alpha_i > 0 \right\}, \end{aligned} \tag{6}$$

where the set *SV* contains the support vectors.

In this paper, the Gaussian radial basis function (RBF) kernel

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\gamma \left\| \boldsymbol{x}_i - \boldsymbol{x}_j \right\|^2 \right) \tag{7}$$

is employed, since it has been shown that the RBF kernel includes other kernel functions as special cases, and is at least as good as other alternatives, such as the sigmoid kernel (Hsu et al., 2003; Keerthi and Lin, 2003; Lin and Lin, 2003). In addition, it exhibits less numerical difficulties because output values of the Gaussian function lie between zero and one, whereas that of other kernels (e.g., polynomial kernels) range between zero and infinity (Coussement and Van den Poel, 2008). Finally, the choice of the RBF kernel is motivated by previous work in horserace modelling (Edelman, 2007).

In order to solve a classification task with a RBF-SVM, two free parameters have to be determined: The regularisation parameter, *C*, which controls the trade-off between maximising the margin and classifying the training set without error; and the smoothing parameter, γ, which determines the width of the Gaussian function and therewith the sensitivity of the distance measurement. These parameters are generally tuned by means of a grid-search ap-

proach, which involves selecting different candidate values for C and γ and empirically evaluating all possible combinations (Hsu et al., 2003; Van Gestel et al., 2004).

3. Forecasting the outcome of horseracing events

3.1. Background

Predictive modelling helps to scrutinise the efficiency of horserace betting markets. The market participants' view of a horse's chance of winning is called the "track probability", q_i^j, and can be obtained from the closing odds, u_i^j, of horse i in race j via $q_i^j = 1/1+u_i^j$. A market is informationally efficient if market participants account for all available information. The odds represent the market's best estimate of a horse's chances and should reflect the *true* probability of this horse winning the respective race. Profitable betting becomes possible only if the track probabilities are inaccurate. Consequently, the modelling objective is to accurately assess winning probabilities based on publicly available information. If it is shown that betting on the basis of these probabilities yields a profit, then it can be concluded that the model succeeded in distilling knowledge from publicly available information that was not (fully) discounted in market prices and that the market is informationally inefficient.

Decision making in a horserace betting context can be modelled as a discrete choice process. The CL model (McFadden, 1973) has emerged as a popular approach to study consumer preference among choice candidates. Unlike ordinary logit regression which treats each data point (i.e. each horse) individually, CL maintains the connections within the alternatives of a choice set (i.e. between runners in a given race). This enables the identification of information which affects the choice of each subject (i.e. which horse wins). Consequently, CL enables the winning probability of one horse to be estimated in conjunction with those of its competitors, thus accounting for within-race competition. This ability explains the ongoing popularity of CL in horserace prediction (e.g., Edelman, 2007; Figlewski, 1979; Johnson et al., 2006; Sung et al., 2005).

The aim of a CL horserace forecasting model is to predict a vector of winning probabilities $\boldsymbol{p}^j = \left(p_1^j, p_2^j, ..., p_{m_j}^j \right)$ for race j, where component, p_i^j, represents the estimated model probability of horse i winning race j, and m_j denotes the number of runners in race j. To achieve this, a *winningness index*, W_i^j, is defined as follows:

$$W_i^j = \boldsymbol{\beta} \cdot \boldsymbol{x}_i^j + \varepsilon_i^j, \tag{8}$$

where $\boldsymbol{\beta}$ is a vector of coefficients which measure the relative contribution of input variables contained in the vector, $\boldsymbol{x}_i^j$, describing runner *i* in race *j*. The error term, ε_i^j, represents unperceived information. If W_i^j is defined such that the horse with the highest value of the winningness index wins race *j*, then it can be shown that, if errors are independent and distributed according to the double exponential distribution, the probability of horse *i* winning race *j* is given by the following CL function (McFadden, 1973):

$$p_i^j = \frac{\exp\left(\boldsymbol{\beta} \cdot \boldsymbol{x}_i^j\right)}{\sum_{i=1}^{m_j} \exp\left(\boldsymbol{\beta} \cdot \boldsymbol{x}_i^j\right)}. \tag{9}$$

As noted by Johnson et al. (2006), this choice of model allows the exponent $\beta \cdot \boldsymbol{x}_i^j$ to be interpreted directly as the ability of horse *i*. The models' coefficients, $\boldsymbol{\beta}$, are estimated by means of maximum likelihood procedures. In particular, given a training dataset of *R* races, the joint likelihood $L = L(\boldsymbol{\beta})$ is the probability of observing the respective results, assuming the p_i^j are as above. Therefore,

$$\boldsymbol{\beta} \leftarrow \max_{\boldsymbol{\beta}} L(\boldsymbol{\beta}) = \prod_{j=1}^{R} p_{i*}^j = \prod_{j=1}^{R} \frac{\exp\left(\boldsymbol{\beta} \cdot \boldsymbol{x}_{i*}^j\right)}{\sum_{i=1}^{m_j} \exp\left(\boldsymbol{\beta} \cdot \boldsymbol{x}_i^j\right)}, \tag{10}$$

whereby $\boldsymbol{x}_{i*}^j$ represents the winner of race *j*.

3.2. Two-stage antecedents of the proposed model

Previous studies have demonstrated that track probabilities are a very good predictor of race results (Bruce and Johnson, 2000). Therefore, it has been suggested that it might be prejudicial to utilise track probability alongside fundamental variables describing a horse's ability in a single forecasting model (Benter, 1994; Edelman, 2007). In particular, the dominating impact of the former may mask the impact of other variables and unduly influence the model (Sung and Johnson, 2007).

To alleviate this problem and to capture the true influence of fundamental variables a two-step forecasting procedure has been proposed. In one such model, Benter (2003) develops a first stage model which predicts a runner's finishing position by means of multivariate linear regression (MLR) using only fundamental variables. Track probabilities are not considered in this step. The estimated finishing positions are interpreted as an assessment of a runner's ability, based on its previous performances captured by the fundamental variables. Subsequently,

this ability score is pooled with track probabilities using CL to estimate model-based winning probabilities.

Let D represent a database of R past races with an overall number of M runners. Let D_1 and D_2 be disjoint sub-samples of D containing R_1 and R_2 races with M_1 and M_2 runners, respectively. Denote by $\boldsymbol{x}_i^j$ the vector of fundamental variables describing horse i in race j and by y_i^j its respective finishing position, the two-stage procedure is then given as follows:

$$\begin{aligned} &\text{Stage one:} \quad f_{MLR}(\boldsymbol{x}) = \hat{\boldsymbol{w}} \cdot \boldsymbol{x} + \hat{b} \\ &\qquad \left\{\hat{\boldsymbol{w}}, \hat{b}\right\} \leftarrow \min_{\boldsymbol{w},b} \sum_{i=1}^{M_1} \left(y_i - (\boldsymbol{w} \cdot \boldsymbol{x}_i + b)\right)^2 \\ &\text{Stage two:} \quad p_i^j = \frac{\exp\left(\beta_1 f_{MLR}\left(\boldsymbol{x}_i^j\right) + \beta_2 q_i^j\right)}{\sum_{i=1}^{m_j} \exp\left(\beta_1 f_{MLR}\left(\boldsymbol{x}_i^j\right) + \beta_2 q_i^j\right)}; \qquad i = 1,...,M_2;\ j = 1,...R_2, \end{aligned} \tag{11}$$

with $\boldsymbol{w}$ and b representing the slope and intercept of a linear regression function, and $\hat{\boldsymbol{w}}$, $\hat{b}$ their respective ordinary least square estimates, calculated over the first stage training dataset. Since the fundamental variables are processed in stage one and are summarized in f_{MLR}, the second stage CL models incorporates only two inputs with respective coefficients β_1 and β_2.

Note that the index j is dropped in stage one because linear regression is unable to exploit information concerning race context. That is, all runners are considered as independent and their finishing position is estimated solely from their respective fundamental variables. One way to overcome this restriction and take a runner's competitors into account is to replace the linear regression in stage one with a CL regression step:

$$\begin{aligned} &\text{Stage one:} \quad f_{CL}\left(\boldsymbol{x}_i^j\right) = \frac{\exp\left(\hat{\boldsymbol{\alpha}} \cdot \boldsymbol{x}_i^j\right)}{\sum_{i=1}^{m_j} \exp\left(\hat{\boldsymbol{\alpha}} \cdot \boldsymbol{x}_i^j\right)} \\ &\qquad \hat{\boldsymbol{\alpha}} \leftarrow \max \prod_{i=1}^{M_1} \frac{\exp\left(\boldsymbol{\alpha} \cdot \boldsymbol{x}_i^j\right)}{\sum_{i=1}^{m_j} \exp\left(\boldsymbol{\alpha} \cdot \boldsymbol{x}_i^j\right)} \qquad j = 1,...R_1 \\ &\text{Stage two:} \quad p_i^j = \frac{\exp\left(\beta_1 f_{CL}\left(\boldsymbol{x}_i^j\right) + \beta_2 q_i^j\right)}{\sum_{i=1}^{m_j} \exp\left(\beta_1 f_{CL}\left(\boldsymbol{x}_i^j\right) + \beta_2 q_i^j\right)} \quad j = 1,...R_2. \end{aligned} \tag{12}$$

This approach has been successfully applied in Sung et al. (2005) and is shown to outperform a corresponding one-step model in Sung and Johnson (2007).

Edelman (2007) modifies this two-stage model to overcome some algorithmic limitations of CL and MLR, respectively. In particular, these techniques infer a model by minimising the forecasting error on training data. Consequently, they are prone to model not only the struc-

ture but also the noise within the data (i.e. overfit the data), especially if a large number of fundamental variables is processed (Vapnik, 1995). Furthermore, they are unable to account for non-linear interactions among the variables unless these are pre-defined by the modeller. Therefore, Edelman (2007) adjusts the two-stage model shown in (11), by using SVR (Smola and Schölkopf, 2004) instead of MLR. In addition, he modifies the original SVR procedure to allow for multiple intercept terms. That is, a b_j rather than b is introduced in (11), which results in an effective stratified analysis by race (Edelman, 2007).

The three approaches outlined above differ only in the first stage, whereas the general idea of first modelling fundamental variables and, subsequently, combining the output of the first stage model with track probabilities using the CL model, is identical. Therefore, the notation of first stage model/second stage model will be used throughout the rest of the paper to refer to different procedures. For example, MLR/CL refers to the original two-stage model (11), whereas CL/CL represents (12) and SVR/CL the approach of Edelman (2007).

It should be noted that the unmodified SVR algorithm rather than the stratified one of Edelman (2007) is used as benchmark in this study to obtain a clearer view on the competitive performance of classification-based versus regression-based modelling and, thereby, the reliability of rank ordered finishing data.

3.3. A two-stage SVM-based classification model

The two-step forecasting model developed in this paper builds upon Edelman (2007). It differs from Edelman's model in that in stage one he conducts a regression of horses' finishing positions whereas the model proposed here uses SVMs to derive a classification model which strives to identify a race's winner. This is motivated by the view that, in a horseracing context, minor placings do not necessarily carry informational value. The rules of racing require jockeys to continue riding in a manner to secure the horse's best possible finish position, but there is little incentive for them to do this when it becomes clear that they are not going to secure a prize. In fact, there are good reasons for jockeys to secure a *poorer* finish position on non-winning horses than they might be able to achieve. This will have the effect of reducing the public's perception of the ability of the horse, which will result in higher odds being available on the horse in subsequent races; offering the prospect of sizeable betting gains to the owners. Consequently, the reliability of rank order finishing data, which constitutes the core of regression-based modelling, is questionable. This view is supported by Sung and Johnson's (2007) empirical findings that rank order finish data beyond position two cannot be relied upon. This problem is alleviated in classification because the modelling focuses on distinguishing win-

ning and non-winning horses and does not use potentially unreliable rank orderings (associated with minor placings).

The proposed SVM/CL model is given as follows, where *y* denotes a binary win/non-win indicator variable rather than a finishing position in stage one:

$$\begin{aligned}
&\text{Stage one:} \quad f_{SVM}(\boldsymbol{x}) = \left(\sum_{i \in SV} \hat{\alpha}_i y_i \exp\left(-\gamma \left\| \boldsymbol{x}_i - \boldsymbol{x} \right\|^2\right)\right) + \hat{b} \\
&\qquad \left\{\hat{\boldsymbol{\alpha}}, \hat{b}\right\} \leftarrow \max_{\alpha} = \sum_{i=1}^{M_1} \alpha_i - \frac{1}{2} \sum_{i,k=1}^{M_1} \alpha_i \alpha_k y_i y_k \exp\left(-\gamma \left\| \boldsymbol{x}_i - \boldsymbol{x}_k \right\|^2\right) \\
&\qquad\qquad s.t. \sum_{i=1}^{M_1} \alpha_i y_i = 0; \quad 0 \le \alpha_i \le C \qquad \forall i = 1, ..., M_1 \\
&\text{Stage two:} \quad p_i^j = \frac{\exp\left(\beta_1 f_{SVM}\left(\boldsymbol{x}_i^j\right) + \beta_2 q_i^j\right)}{\sum_{i=1}^{m_j} \exp\left(\beta_1 f_{SVM}\left(\boldsymbol{x}_i^j\right) + \beta_2 q_i^j\right)} \qquad i = 1, ..., M_2;\ j = 1, ...R_2.
\end{aligned} \tag{13}$$

The objective of the first stage model is to estimate the likelihood of a given runner being a winner. Therefore, the sign-function, see (6), is removed to obtain a continuous output from the SVM. The resulting value, $f_{SVM}(\boldsymbol{x})$, is proportional to the distance of a data point (a horse) to the separating hyperplane (between winners and losers) and, therefore, represents a confidence that a point (horse) belongs to a particular class (Vapnik, 1995) (i.e. is a winner or loser). That is, a horse assigned a higher SVM score is more likely to be a winner.

As in most previous models, information concerning which horses compete against each other and prior knowledge that each race has a unique winner is lost in stage one, SVMs are unable to account for within-race competition. However, it has to be emphasised that the overall objective of the forecasting model is not to maximise the number of correct winner predictions but to maximise profit. Profitable betting requires an accurate estimate of horses' winning probabilities. The first stage output serves only as a summary of a horse's ability (based on previous performances), whereas the second stage accounts for information on within-race competition.

Nonetheless, it is expected that enabling SVM to capture some elements of within-race competition in stage one will improve final estimates. To that end, a race-wise standardisation procedure is proposed to augment the data. Continuous variables are commonly standardised to zero mean and standard deviation of one before applying a forecasting model to avoid numerical difficulties with different value ranges (Bishop, 1995). This is accomplished by subtracting from a variable its mean value over the dataset and dividing by the respective standard deviation. This pre-processing is enhanced to account, to some extent, for within-race

competition. In particular, additional predictors are derived from the raw variables by standardising the data *within* a race as follows:

$$\tilde{x}_{it}^{j} = \frac{x_{it}^{j} - \overline{x}_{t}^{j}}{\sigma_{t}^{j}} \forall i = 1,...,m_{j}, \qquad (14)$$

where $\tilde{x}_{it}^{j}$ (x_{it}^{j}) denotes the new (original) value of attribute *t* of runner *i* in race *j* and the mean ($\overline{x}_{t}^{j}$) as well as the standard deviation (σ_{t}^{j}) are calculated over the runners in race *j*. To illustrate the intuition behind race-wise standardisation, consider two four-runner races with horses' class as the single input. Table 1 shows that standardisation across the database maintains the relative class differences between races whereas race-wise standardisation enables relative class differences between horses in a race to be compared across races.

[Table 1 about here]

4. Empirical evaluation of the SVM/CL forecasting model

4.1. Data and variables

The data on which the empirical analysis of this study is based was provided by Raceform Ltd. and relates to races run at Goodwood racetrack in the UK between May 1995 and August 2000. This period was deliberately chosen since the biggest online betting exchange, Betfair, was first advertised in Oct 2000. It is widely accepted that the advent of Betfair considerably increased the competition between bookmakers (since individual Betfair customers can act as bookmakers) and increased the number of professional bettors in the market. These changes are likely to have changed the market ecology, and, in particular, to have increased the degree to which information is discounted in final odds. The advantage of using data prior to September 2000 is that this enables us to reference our empirical results to that of other seminal studies conducted on markets before the advent of Betfair (e.g., Bolton and Chapman, 1986: 200 races run prior to 1986; and Edelman, 2007: 300 races run in 1995).

The data consists of 556 races with 5,947 runners. The 400 races (4,296 horses) run before May 1999 are used to develop the forecasting models whereas the remaining 156 races run after May 1999 are preserved to conduct out-of-sample testing.

In order to set a difficult task for the forecasting model, the fundamental variables included in the first-step model are limited to those included in Bolton and Chapman (1986) (see Table 2). These variables were in the public domain for 9 years prior to 1995, and it is there-

fore likely that bettors would have attempted to discount these in final odds; suggesting that it would be difficult for a model to produce estimates which could be used to make profits.

[Table 2 about here]

4.2. Experimental setup

As a first step, continuous variables within the dataset are normalised by the database- and race-wise standardisation procedures indicated in section 3.3. This process results in 20 continuous variables, plus two binary indicator variables which are not pre-processed.

A sub-sample of 200 races (from the 400 training races run prior to May 1999) is used to construct a SVM classifier with RBF-kernel (see (13)). The free parameters C and γ are determined by means of five-fold cross-validation (Stone, 1974). That is, the 200 races are split into five equal-sized partitions and a SVM model is recursively built on four partitions and assessed on the remaining one. The resulting five performance values are averaged to provide an estimated out-of-sample performance of the respective parameter setting; performance is measured in terms of the number of accurately predicted winners at this stage. 441 different parameter settings are considered from a grid of $\log(C) = \{-3, -2 \ldots, 17\}$ and $\log(\gamma) = \{-20, -19, \ldots, 0\}$ (Hsu et al., 2003). The parameter values which lead to the highest number of correctly identified winners during cross-validation are retained and a final SVM classifier with this setting is built on the whole 200 races.

Subsequently, the resulting SVM classification model is used to score the remaining 200 training set races, providing an ability index concerning the relative strength of each horse which is based solely on fundamental variables. The SVM ability index is then pooled with the track probability in a stage two using CL (13).

In order to appraise the profitability of the forecasting model, a *Kelly wagering strategy* (Kelly, 1956) is implemented. The Kelly strategy identifies how much to bet on each horse: Let r_i^j be the return on a bet of one pound if horse i wins race j and let b_i^j be the fraction of current wealth that is bet on horse i, respectively. Given that horse h wins race j, the current wealth increases by a factor

$$1 - \sum_{i=1}^{m_j} b_i^j + b_h^j \cdot r_h^j . \qquad (15)$$

The Kelly strategy determines bets to maximise the expected log payoff across all potential winners h using the model-based winning probabilities, p_i^j, (see (11)-(13)). It has been shown to be optimal in the sense that it maximises the asymptotic rate of growth for wealth, with zero probability of ruin (Breiman, 1961).

$$\max_{b_h^j} \sum_{h=1}^{m_j} p_h^j \cdot \ln\left(1 - \sum_{i=1}^{m_j} b_i^j + b_h^j \cdot r_h^j\right). \qquad (16)$$

Consequently, if the proposed methodology produces a higher positive return than that achieved by models previously employed, this will be taken as evidence that the SVM/CL approach adds value (and, in passing, that the horserace betting market is not informationally efficient).

4.3. Benchmarking the proposed two-stage model

The empirical evaluation examines the effectiveness of the proposed SVM/CL model with RBF kernel function (which includes database- and race-wise normalised variables). The ability of this model to discover relationships in the underlying data which are not yet discounted in market prices is confirmed when examining the performance of the holdout sample bets: A Kelly wagering strategy (without reinvestment) based on the predicted winning probabilities of the proposed model yields a return of 30.58 per cent (see Table 3).

In order to set this result in context, a two-step CL/CL model (Sung et al., 2005) and a SVR/CL procedure (Edelman, 2007) are considered as benchmarks. Applying these techniques to the same 156 races yields a rate of return of 1.74 per cent and 17.50 per cent, respectively (see Table 3). Similar comparisons can be observed when permitting reinvestment of winnings. The two support vector-based methods, SVM/CL and SVR/CL, produce a significant increase in wealth (642.65 per cent, and 211.55 per cent, respectively) over the holdout races, whereas wealth decreases by 16.53 per cent when using the CL/CL model. These results are depicted in Figure 2 which plots the development of the natural logarithm of cumulative return over the 156 holdout races for all three models.

[Table 3 about here]

[Figure 2 about here]

The proposed SVM/CL method compares favourably to its two competitors, providing significantly higher profits using Kelly betting with and without reinvestment of winnings over the holdout sample races. The results of applying Kelly without reinvestment are a more reliable indicator of the models' relative success, since the profits achieved with reinvestment can arise from fortunate selection of the order in which winners and losers occur in the holdout sample. Consequently, in subsequent analysis we focus on the 'without reinvestment' results.

It might also be argued that relying solely on the profitability of a particular model overlooks other key performance indicators for a model. Therefore, Table 3 summarises additional performance indicators to aid further comparison of the three methods.

The R^2 of the SVM/CL model (0.132) exceeds those of its two competitors, indicating that the winning probabilities generated by the former capture more useful information contained in the fundamental variables. This is confirmed when examining the values of the *t*-statistic of β_1, the CL coefficient associated with the output of stage one ((11)-(13)). The ability index obtained by processing the fundamental variables in stage one with a SVM has the highest *t*-value and may, consequently, be regarded as being most informative.

A model's discriminative power in terms of its *area under a receiver-operating-characteristics curve* (AUC) is also considered as performance indicator (Fawcett, 2006). The AUC is a popular metric for assessing classifiers. For this application, it represents the probability that a model assigns a higher winning probability to a winning horse than to a loser. Consequently, practical AUC values range between 0.5 and 1 with higher values representing higher discriminative power (an AUC of 0.5 represents a classifier which randomly guesses a class; see Fawcett, 2006). The performance differences in terms of AUC among all three models are minor (see Table 3) and it will be shown that this trend continues in subsequent experiments. These results suggest that although SVM/CL is only slightly better in terms of identifying winners, it excels in producing accurate winning probabilities (and hence achieving profits).

The major difference between the approach proposed here and Edelman's (2007) seminal work is the usage of rank ordered finishing data during model building. Consequently, the higher profitability of the SVM/CL model over a respective regression-based model observed in this study indicates that placings beyond the winner do not contain valuable information. Therefore, the results suggest that classification may be more reliable than regression for horseracing data. This view is supported when conducting a formal test to scrutinise if races for the second place (i.e. these races are artificially manufactured by excluding the ultimate winner from all training races) follow the same distribution as the races to finish first (Chapman and Staelin, 1982; Watson and Westin, 1975). The respective test statistic ($\chi^2_{13} = 21.20$) indicates that this hypothesis should be rejected at the 7 per cent level, providing further evidence of the unreliability of the rank ordered finishing positions.

4.4. Examining the origin of profit

Despite the appealing performance of the SVM/CL model, SVMs are inherently 'black-box' methods that provide no explanation of the relationships discovered in the data. Consequently, additional simulations are required to shed light on the origin of the profit observed within the Kelly-based betting simulation. Beside the fact that SVM classification, rather than CL or regression (SVR or MLR), is used in stage one, three main factors can be identified which affect the performance of the SVM/CL model; namely, implementing a two-step modelling procedure which postpones usage of track probabilities, employing a special data pre-processing approach to capture some information on within-race competition, and using a non-linear model (i.e. RBF kernel function), to distinguish between winners and losers. The results of these experiments are summarised in Table 4.

4.4.1 One stage versus two stage models

With respect to the two-stage modelling approach, Sung and Johnson (2007) have shown that it is superior to single stage models when using CL. Their result is confirmed for the dataset employed in this study: A one-stage CL model yields a loss of 0.46 per cent over the holdout races when applying the Kelly strategy without reinvestment. However, a single stage SVM model performs much worse, resulting in a loss of 22.57 per cent over the same races (Table 4). Note that Platt's (2000) procedure is used to obtain probability estimates from the SVM classifier. Consequently, the winning probabilities produced by single stage CL or SVM models are significantly inferior to those of respective two-stage models.

The inappropriateness of the single stage SVM model can be explained since the classification approach is ad-hoc, in the sense that it estimates winning probabilities that do not sum to one across a race; the binary win/loss target variable within each race is considered as independent, which it clearly is not as only one horse can win. SVM is unable to take relationships among individual runners (data points) into account and the SVM/CL method, consequently, relies heavily upon the second stage CL model. On the other hand, the single stage CL does not suffer from this shortcoming as it naturally models within-race competition. Therefore, it outperforms the one-stage SVM model. Finally, the inferiority of a one-stage CL model to a respective two-step CL/CL model originates from the fact that the former processes fundamental variables and track probabilities simultaneously. Due to the dominant influence of track probabilities, subtle relationships among fundamental variables are missed when following this approach (Sung and Johnson, 2007).

4.4.2 Database-wise versus race-wise normalisation

Another ingredient of the proposed method is a novel data pre-processing technique (i.e. race-wise normalisation), to provide the first stage SVM model with some information concerning within-race competition. To demonstrate the effectiveness of including database- *and* race-wise normalised variables two additional SVM/CL models are built which utilise only database- *or* race-wise normalised variables. The performance of these models is summarised in Table 4 and a comparison of Kelly returns without reinvestment, R^2, and the *t*-statistic of β_1, all suggest that these two transformations complement each other and capture different aspects of racing data. Combining these transformations enhance the predictive accuracy of SVM-based horserace prediction models.

It is important to emphasise that the raw variables in all three models are the same. Consequently, the two sets of variables (database- and race-wise normalised) are highly correlated. Nonetheless, SVM is able to capitalise on the enlarged input set. This can be interpreted as further confirmation of Edelman's (2007) argument that SVM-type procedures are well suited for horserace modelling because of their ability to process high-dimensional, correlated inputs.

4.4.3 Linear versus non-linear models

The proposed SVM/CL model embodies a RBF kernel function to account for non-linear relationships among the fundamental variables. The superior performance of the SVM/CL over a CL/CL model, which accommodates only linear relationships, indicates that the relationship between independent variables and race outcome is non-linear. However, to obtain a clearer view on this issue the performance of a SVM/CL model with a *linear* kernel is computed (last row of Table 4). A betting simulation of this model over the 156 holdout races reveals that the linear SVM/CL model produces an inferior return (7.35 per cent without reinvestment) to that produced by the non-linear SVM/CL model (30.58 per cent). Similarly, all other performance indicators demonstrate the superiority of the non-linear model. Given that the kernel function is the only difference between these two models, it can be concluded that the non-linear relationships which exist among independent variables should be taken into account when modelling race outcome.

[Table 4 about here]

4.5. Discussion

In assessing the significance of the results presented in the previous sections, it is important to remember that the only variables used in this study are those included in Bolton and Chap-

man's (1986) seminal paper. It was expected that, since they have been in the public domain for many years, the public is likely to have fully discounted information contained in these variables in market odds. Despite this, the CL/CL model manages to generate a small profit over the 156 race holdout sample (1.74 per cent) if winnings are not reinvested. Given that this approach is much championed as a technique for extracting information from variables in horse races the return obtained here is a difficult benchmark. However, both methods building upon support vector methodology achieve a significant improvement in terms of rate of return (SVM/CL: 30.58 per cent and SVR/CL: 17.50 per cent) and in terms of other performance indicators. This result confirms Edelman's (2007) previous findings that embedding modern machine learning methods in a two-step model enables information from fundamental variables, that have not yet been taken into account by the betting public, to be distilled. In addition, the proposed classification-based approach, SVM/CL, has been demonstrated to enable significant further improvements over a SVR/CL approach. Consequently, the suspicion that a regression-based forecasting of runners' finish positions may suffer from unreliable rank orderings is confirmed.

Despite the appealing empirical performance, using classification has some theoretical drawbacks. In particular, the binary win/loss target variable is not independent across runners. A multi-nominal SVM formulation (see, e.g., Hsu and Lin, 2002) could be considered as an alternative. This would involve defining a runners' finish position as a discrete target variable and building a SVM model which distinguishes the horses that finish first, second, etc. However, such an approach has major disadvantages. In particular, races may include a large number of runners and one class is needed for each possible finishing position. For example, the largest number of runners within one race is 30 for the dataset employed here. However, the main argument against multi-class classification is that it, again, makes use of rank ordered finishing data. A key motivation of this study is the suspicion (confirmed) that this type of information may be unreliable and (binary) classification is, thus, more robust. On the other hand, if some prior knowledge is available which indicates that it is safe to extract information from finishing positions, the natural approach to employ is regression-based modelling; and, in view of Edelman's (2007) results, SVR would be the obvious candidate. Consequently, multi-class classifiers do not seem ideally suited for horserace modelling.

A key issue in predicting the outcome of racing events is within-race competition. From a methodological point of view, the only procedure currently available capable of accommodating relationships among runners is CL. On the other hand, the results of Benter (1994), Edelman (2007) and those presented here indicate that ordinary forecasting techniques which con-

sider each example as independent are well suited, if they are combined with CL within a second stage. In fact, the combination with CL is crucial as is demonstrated in section 4.4.1. Consequently, further improvements can be expected when it becomes possible to also model relationships among horses in a race in stage one. Race-wise normalisation represents an attempt along this line and approaches the problem by means of data pre-processing.

An orthogonal approach could focus on algorithmic modifications of the techniques employed in stage one (see also Edelman's (2007) modifications of SVR). A common feature of MLR, SVR and SVM, as well as many other techniques, is the way they model inaccuracy. In particular, the empirical loss over the training records (possibly together with regularisation considerations) is optimised during model building. The particular loss function differs for classification and regression models. Other models like ordinal SVMs (Herbrich et al., 1999) or kernel logistic regression (Keerthi et al., 2005) could be considered which follow the same ideas underlying SVMs, but embody different types of loss functions. However, all these procedures measure *loss over individual examples* and aggregate these values to form an overall empirical error measure. This is the step where dependencies among examples (i.e. race context) is lost. Recent advances in the field of structural SVMs could offer an alternative by allowing more complex loss functions that are not restricted to individual examples. For example, Joachims (2005) develops a SVM which optimised AUC directly. This technique appears to be a well suited candidate for (first-stage) horserace modelling to be assessed in future work.

5. Conclusion

A two-stage methodology for forecasting results of competitive events has been proposed and its value in assessing how traders in financial markets use information has been explained. The proposed model differs from other two-stage models by considering *classification* rather than *regression* in the first stage to avoid problems with unreliable rank orderings of the horses. Instead, a within-race attribute standardisation procedure has been undertaken to provide the SVM-based model with some information on within-race competition. The empirical results have demonstrated the merit of each of the model's components, as well as the effectiveness of the overall model in offering sizeable accuracy improvements over competitive alternatives.

The results indicate that, although horserace betting models using similar fundamental variables have been in the public domain for many years (Benter, 1994; Bolton and Chapman, 1986; Chapman, 1994), the betting public still does not fully discount the information content

of these variables in market odds. This reflects the complexity of the relationship between the fundamental variables and race outcome, which, in view of the observed results, is likely to include non-linear interactions and might remain opaque to those populating the markets. In future work, techniques for extracting rules from trained SVM classifiers (e.g., Martens et al., 2007) could be applied to explore the nature of the relationships amongst the variables and improve understanding of the information which individuals in these markets fail to discount. In addition, it would be interesting to conduct further experiments using data after the advent of Betfair. Comparing such results to the ones presented here could help to quantify the degree to which the internet has changed the ecology of horserace betting markets.

References

Benter W 1994. Computer Based Horse Race Handicapping and Wagering Systems: A Report. In: Hausch DB, Lo VSY, Ziemba WT (Eds.), Efficiency of Racetrack Betting Markets. Academic Press: London; 1994. pp. 183-198.

Benter W. Advances in the Mathematical Modelling of Horse Race Outcomes. 12th International Conference on Gambling and Risk Taking. Vancouver, British Columbia, Canada; 2003.

Bishop CM. Neural Networks for Pattern Recognition. Oxford University Press: Oxford; 1995.

Bolton RN, Chapman RG. Searching for positive returns at the track: A multinomial logit model for handicapping horse races. Management Science 1986;32(8); 1040-1060.

Breiman L 1961. Optimal Gambling Systems for Favourable Games. In: Neyman J (Ed.), Proceedings of the 4th Berkeley Symposium on Mathematical Statistics and Probability. University California Press: Berkeley; 1961. pp. 63-68.

Bruce AC, Johnson JEV. Investigating the roots of the favourite-longshot bias: An analysis of supply and demand side agents in parallel betting markets. The Journal of Behavioral Decision Making 2000;13(4); 413-430.

Burges CJC. A tutorial on support vector machines for pattern recognition. Data Mining and Knowledge Discovery 1998;2(2); 121-167.

Chapman RG 1994. Still Searching for Positive Returns at the Track: Empirical Results from 2000 Hong Kong Races. In: Hausch DB, Lo VSY, Ziemba WT (Eds.), Efficiency of Racetrack Betting Markets. Academic Press: New York; 1994. pp. 173-181.

Chapman RG, Staelin R. Exploiting rank ordered choice set data within the stochastic utility model. Journal of Marketing Research 1982;19(3); 288-301.

Coussement K, Van den Poel D. Churn prediction in subscription services: An application of support vector machines while comparing two parameter-selection techniques. Expert Systems with Applications 2008;34(1); 313-327.

Cristianini N, Shawe-Taylor J. An Introduction to Support Vector Machines and other Kernel-based Learning Methods. Cambridge University Press: Cambridge; 2000.

Cui D, Curry D. Predictions in marketing using the support vector machine. Marketing Science 2005;24(4); 595-615.

Edelman D. Adapting support vector machine methods for horserace odds prediction. Annals of Operations Research 2007;151(1); 325-336.

Fawcett T. An introduction to ROC analysis. Pattern Recognition Letters 2006;27(8); 861-874.

Figlewski S. Subjective information and market efficiency in a betting market. Journal of Political Economy 1979;87(1); 75-89.

Gu MG, Huang C, Benter W. Multinomial Probit Models for Competitive Horse Racing. Working paper, Chinese University of Hong Kong 2003.

Hausch DB, Ziemba WT. Transactions costs, market inefficiencies and entries in a racetrack betting model. Management Science 1985;31(4); 381-94.

Herbrich R, Graepel T, Obermayer K. Support Vector Learning for Ordinal Regression. 9th International Conference on Artificial Neural Networks. Edinburgh, Scotland, UK; 1999.

Hsu C-W, Chang C-C, Lin C-J. A Practical Guide to Support Vector Classification. Technical report, Department of Computer Science and Information Engineering, National Taiwan University 2003.

Hsu C-W, Lin C-J. A comparison of methods for multi-class support vector machines. IEEE Transactions on Neural Networks 2002;13(2); 415-425.

Joachims T. A Support Vector Method for Multivariate Performance Measures. 22nd International Conference on Machine Learning. Bonn, Germany; 2005.

Johnson JEV, Jones O, Tang L. Exploring decision makers' use of price information in a speculative market. Management Science 2006;52(6); 897-908.

Keerthi SS, Lin C-J. Asymptotic behaviors of support vector machines with gaussian kernel. Neural Computation 2003;15(7); 1667-1689.

Keerthi SS, Duan KB, Shevade SK, Poo AN. A fast dual algorithm for kernel logistic regression. Machine Learning 2005;61(1-3); 151-165.

Kelly JL. A new interpretation of information rate. The Bell System Technical Journal 1956;35; 917-926.

Law D, Peel DA. Insider trading, herding behaviour and market plungers in the British horse-race betting market. Economica 2002;69(274); 327-238.

Levitt SD. Why are gambling markets organised so differently from financial markets? The Economic Journal 2004;114(495); 223-246.

Lin H-T, Lin C-J. A Study on Sigmoid Kernels for SVM and the Training of Non-PSD Kernels by SMO-type Methods. Technical report, Department of Computer Science and Information Engineering, National Taiwan University 2003.

Martens D, Baesens B, Van Gestel T, Vanthienen J. Comprehensible credit scoring models using rule extraction from support vector machines. European Journal of Operational Research 2007;183(3); 1466-1476.

McFadden D. Conditional Logit Analysis of Qualitative Choice Behavior. In: Zarembka P (Ed.), Frontiers in Econometrics. Academic Press: New York; 1973. pp. 105-142.

Platt JC. Probabilities for Support Vector Machines. In: Smola AJ, Bartlett P, Schölkopf B, Schuurmans D (Eds.), Advances in Large Margin Classifiers. MIT Press: Cambridge; 2000. pp. 61-74.

Sauer RD. The economics of wagering markets. Journal of Economic Literature 1998;36(4); 2021-2064.

Schnytzer A, Shilony Y. Inside information in a betting market. The Economic Journal 1995;105(431); 963-971.

Smola AJ, Schölkopf B. A tutorial on support vector regression. Statistics and Computing 2004;14(3); 199-222.

Stone M. Cross-validatory choice and assessment of statistical predictions. Journal of the Royal Statistical Society (Series B) 1974;36(2); 111-147.

Sung M-C, Johnson JEV, Bruce AC 2005. Searching for Semi-Strong Form Inefficiency in the UK Racetrack Betting Market. In: Vaughan Williams L (Ed.), Information Efficiency in Financial and Betting Markets. Cambridge University Press: Cambridge; 2005. pp. 179-192.

Sung M, Johnson JEV. Comparing the effectiveness of one- and two-step conditional logit models for predicting outcomes in a speculative market. Journal of Prediction Markets 2007;1(1); 43-59.

Suykens JAK, Vandewalle J. Least square support vector machine classifiers. Neural Processing Letters 1999;9(3); 293-300.

Van Gestel T, Suykens JAK, Baesens B, Viaene S, Vanthienen J, Dedene G, De Moor B, Vandewalle J. Benchmarking least squares support vector machine classifiers. Machine Learning 2004;54(1); 5-32.

Vapnik VN. The Nature of Statistical Learning Theory. Springer: New York; 1995.

Vaughan Williams L. Information efficiency in betting markets: A survey. Bulletin of Economic Research 1999;51(1); 1-39.

Watson PL, Westin RB. Transferability of disaggregated mode choice models. Regional Science and Urban Economics 1975;5; 227-249.

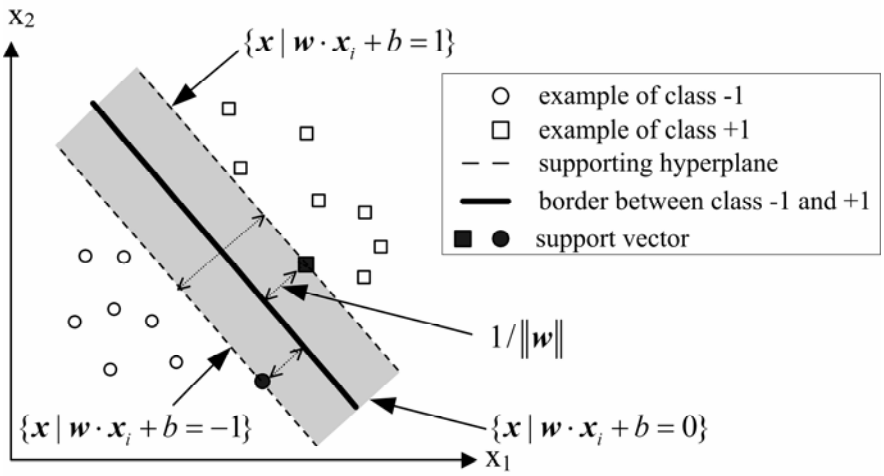


Figure 1: Linear separation of two classes -1 and +1 in two-dimensional space with a SVM classifier.

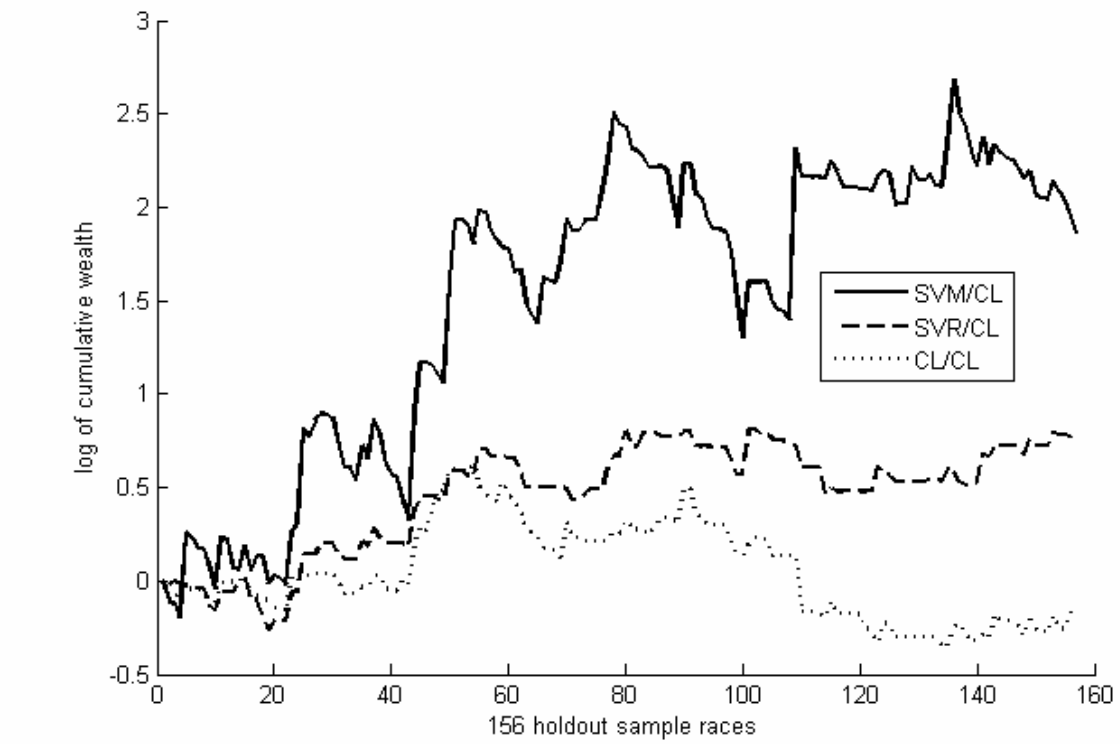


Figure 2: Wealth as a result of applying a Kelly-wagering strategy to holdout sample races.

Table 1: Comparison between race-wise and database-wise standardisation

		Horse class*	Database-wise standardisation	Race-wise standardisation
Race 1	Horse 1	20	-0.234	-1.162
	Horse 2	40	0.435	-0.387
	Horse 3	60	1.104	0.387
	Horse 4	80	1.773	1.162
Race 2	Horse 1	1	-0.870	-1.162
	Horse 2	3	-0.803	-0.387
	Horse 3	5	-0.736	0.387
	Horse 4	7	-0.669	1.162

*Let *horse class* be an abstract measure of a horse's ability with higher class value indicating better horses.

Table 2: Definitions of the independent variables employed in the empirical evaluations

Independent variable	Variable definitions
Market-related variable	
$\ln(q_i^j)$	The natural logarithm of the normalised track probabilities.
Fundamental variables	
pre_s_ra	Speed rating for the previous race in which the horse ran.
avgsr4	The average of a horse's speed rating in its last 4 races; value of zero when there is no past run.
disavesr	The average speed rating of the past runs of each horse at this distance; value of zero when no previous run.
go_avesr	The average speed rating of all past runs of the horse on this going; value of zero when no previous run.
draw	Post-position in current race.
eps	Total prize money earnings (finishing first, second or third) to date/Number of races entered.
newdis	1 indicates a horse that ran three or four of its last four races at a distance of 80% less than current distance, and 0 otherwise.
weight	Weight carried by the horse in current race.
win_run	The percentage of the races won by the horse in its career.
jnowin	The number of wins by the jockey in career to date of race.
jwinper	The winning percentage of the jockey in career to date of race.
jst1miss	1 indicates when the other jockey variables are missing; 0 otherwise.

Table 3: Empirical comparison of different two-stage models over 156 holdout races

	Rate of Return		R^2	*t*-value		AUC
	Without reinvestment	With reinvestment		β_1	β_2	
SVM/CL	30.58%	642.65%	0.1323	4.85	10.32	0.762
SVR/CL	17.50%	211.55%	0.1238	2.86	9.57	0.757
CL/CL	1.74%	-16.53%	0.1231	2.64	10.53	0.759

Table 4: Assessing the components of the SVM/CL model

	Rate of Return		R^2	*t*-value		AUC
	Without reinvestment	With reinvestment		β_1	β_2	
Proposed reference model						
SVM/CL	30.58%	642.65%	0.1323	4.85	10.32	0.762
One-stage models[*]						
CL	-0.46	-48.37%	-	-	-	0.737
SVM	-22.57	-100%	-	-	-	0.761
Two-step models employing different input variables						
Only DB[**]	5.86%	116.33%	0.1200	1.14	11.87	0.756
Only RW[**]	3.46%	-0.04%	0.1250	3.26	10.72	0.756
Two-stage SVM-based model with linear kernel						
SVM_{linear}/CL	7.35%	120.73%	0.1193	0.188	13.20	0.756

[*]Note that some performance indicators are not available when using single stage models.
[**]RW=race-wise normalisation, DB=database-wise normalisation.

Available online at www.sciencedirect.com
SCIENCE DIRECT®
European Journal of Operational Research 173 (2006) 781–800
EUROPEAN JOURNAL OF OPERATIONAL RESEARCH
www.elsevier.com/locate/ejor

The impact of preprocessing on data mining: An evaluation of classifier sensitivity in direct marketing

Sven F. Crone [a], Stefan Lessmann [b,*], Robert Stahlbock [b]

[a] Department of Management Science, Lancaster University, Lancaster LA1 4YX, United Kingdom
[b] Institute of Information Systems, University of Hamburg, Von-Melle-Park 5, 20146 Hamburg, Germany

Received 15 November 2004; accepted 18 July 2005
Available online 15 November 2005

Abstract

Corporate data mining faces the challenge of systematic knowledge discovery in large data streams to support managerial decision making. While research in operations research, direct marketing and machine learning focuses on the analysis and design of data mining algorithms, the interaction of data mining with the preceding phase of data preprocessing has not been investigated in detail. This paper investigates the influence of different preprocessing techniques of attribute scaling, sampling, coding of categorical as well as coding of continuous attributes on the classifier performance of decision trees, neural networks and support vector machines. The impact of different preprocessing choices is assessed on a real world dataset from direct marketing using a multifactorial analysis of variance on various performance metrics and method parameterisations. Our case-based analysis provides empirical evidence that data preprocessing has a significant impact on predictive accuracy, with certain schemes proving inferior to competitive approaches. In addition, it is found that (1) selected methods prove almost as sensitive to different data representations as to method parameterisations, indicating the potential for increased performance through effective preprocessing; (2) the impact of preprocessing schemes varies by method, indicating different 'best practice' setups to facilitate superior results of a particular method; (3) algorithmic sensitivity towards preprocessing is consequently an important criterion in method evaluation and selection which needs to be considered together with traditional metrics of predictive power and computational efficiency in predictive data mining.

Keywords: Data mining; Neural networks; Data preprocessing; Classification; Marketing

1. Introduction

In competitive consumer markets, data mining faces the growing challenge of systematic knowledge discovery in large datasets to achieve

* Corresponding author. Tel.: +49 40 42838 5500; fax: +49 40 42838 5535.
E-mail addresses: s.crone@lancaster.ac.uk (S.F. Crone), lessmann@econ.uni-hamburg.de (S. Lessmann), stahlboc@econ.uni-hamburg.de (R. Stahlbock).

0377-2217/$ - see front matter
doi:10.1016/j.ejor.2005.07.023

operational, tactical and strategic competitive advantages. As a consequence, the support of corporate decision making through data mining has received increasing interest and importance in operational research and industry. As an example, direct marketing campaigns aiming to sell products by means of catalogues or mail offers [1] are restricted to contacting a certain number of customers due to budget constraints. The objective of data mining is to select the customer subset most likely to respond in a mailing campaign, predicting the occurrence or probability of purchase incident, purchase amount or interpurchase time for each customer [2,3] based upon observable customer attributes of varying scale. Traditionally, response modelling has utilised transactional data consisting of continues variables to predict purchase incident focusing on the recency of the last purchase, the frequency of purchases and the overall monetary purchase amount, referred to as recency, frequency and monetary value (RFM)-analysis [2]. The continuous scale of these attributes together with their limited number has facilitated the use of conventional statistical methods, such as logistic regression.

Recently, progress in computational and storage capacity has enabled the accumulation of ordinal, nominal, binary and unary demographic and psychographic customer centric data, inducing large, rich datasets of heterogeneous scales. On the one hand, this has advanced the application of data driven methods like decision trees (DT) [4], artificial neural networks (NN) [2,5,6], and support vector machines (SVM) [7], capable of mining large datasets. On the other hand, the enhanced data has created particular challenges in transforming attributes of different scales into a mathematically feasible and computationally suitable format. Essentially, each customer attribute may require special treatment for each algorithm, such as discretisation of numerical features, rescaling of ordinal features and encoding of categorical ones. Applying a variety of different methods, the phase of data preprocessing (DPP) represents a complex prerequisite for data mining in the process of knowledge discovery in databases [8].

Aiming to maximise the predictive accuracy of data mining, research in management science and machine learning is largely devoted to enhancing competing classifiers and the effective tuning of algorithm parameters. Classification algorithms are routinely tested in extensive benchmark experiments, evaluating the impact on predictive accuracy and computational efficiency, using preprocessed datasets; e.g. [9–11]. In contrast to this, research in DPP focuses on the development of algorithms for particular DPP tasks. While feature selection [12–14], resampling [15,16] and the discretisation of continuous attributes [17,18] are analysed in some detail, few publications investigate the impact of data projection for categorical attributes and scaling [19,20]. More importantly, interactions on predictive accuracy in data mining are not been analysed in detail, especially not within the domain of corporate direct marketing.

To narrow this gap in research and practice, we seek to investigate the potential of DPP in a real world scenario of response modelling, predicting purchase incident to identify those customers most likely to respond to a mailing campaign in the publishing industry. We analyse the impact of different DPP schemes across a selection of established data mining methods. Due to the questionable usefulness of traditional statistical techniques in large scale data mining settings [21,22] and mixed scaling levels of customer attributes, we confine our analysis to data driven methods of C4.5 DT, NN and SVM.

The remainder of the paper is organised as follows: We begin with a short overview of the classification methods of DT, NN and SVM used. Next, the task of DPP for competing methods for scaling, sampling and coding is discussed in Section 3. Conducting a structured literature review, we exemplify that the influence of DPP is widely overlooked to motivate our further analysis. This is followed by the case study setup of purchase incident modelling for direct marketing in Section 4 and the experimental results providing empirical evidence for the significant impact of DPP on classification performance in Section 5. Conclusions are given in Section 6.

2. Classification algorithms for data mining

2.1. Multilayer perceptrons

NN represent a class of statistical methods capable of universal function approximation, learning non-linear relationships between independent and dependent variables directly from the data without previous assumptions about the statistical distributions [23]. Multilayer perceptrons (MLP) represent a prominent class of NN [24–26], implementing a paradigm of supervised learning methods which is routinely used in academic and empirical classification and data mining tasks [27–29].

The architecture of a MLP, as shown in Fig. 1, consists of several layers of nodes u_j fully interconnected through weighted acyclic arcs w_{ij} from each preceding layer to the following, without lateral connections or feedback [27]. The information is processed from left to right, using nodes in the input layer to forward input vector information to the hidden layer. Each hidden node j calculates a weighted linear combination $\boldsymbol{w}^{\mathrm{T}}\boldsymbol{o}$ of its input vector $\boldsymbol{o}$, weighting each input activation o_i of node i in the preceding layer with the transposed matrix $\boldsymbol{w}^{\mathrm{T}}$ of the trainable weights w_{ij} including a trainable constant θ_j. The linear combination is transformed by means of a bounded, non-decreasing, non-linear activation functions in each node [21] to model different network behaviour. The processed results are forwarded to the nodes in the output layer, which compute an output vector of the classification results for each presented input pattern.

MLP learn to separate classes directly from presented data, approximating a function $g(\boldsymbol{x})$: $X \rightarrow Y$ by iteratively adapting $\boldsymbol{w}$ after presentation of an input pattern to minimise a given objective function $e(\boldsymbol{x})$ using a learning algorithm. Each node forms a linear hyperplane that partitions feature space into two half-spaces, whereby the non-linear activation function models a graded response of indicated class membership depending on the distance of $\boldsymbol{x}$ to each node hyperplane [27]. Nodes in successive hidden layers form convex regions as intersections of these hyperplanes. Output units form unisons of the convex regions into arbitrarily shaped, convex, non-convex or disjoint regions. The successive combination creates a complex decision boundary that separates feature space into polyhedral sets or regions, each one being assigned to a different class of Y. The desired output of class membership may be coded using a single output node $y_i = \{0; 1\}$ or using n nodes for multiple classifications $y_i = \{(0,1);(1,0)\}$, respectively. Moreover, the choice of the output function allows the prediction of binary class memberships as well as the more suitable conditional probability of class membership to rank each customer instance (see Section 4.3).

Being universal approximators, NN should theoretically be capable of processing any continuous input data or categorical attributes of ordinal, nominal, binary or unary scale [19] to learn any

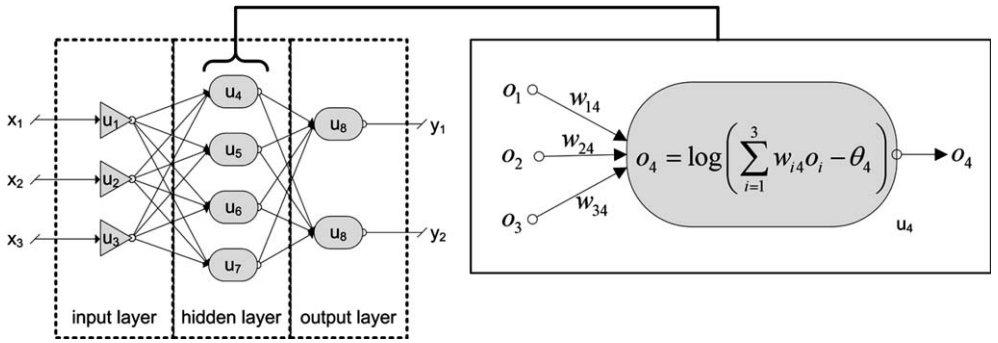


Fig. 1. Three layered MLP showing the information processing within a node, using a weighted sum as input function, the logistic function as sigmoid activation function and an identity output function.

non-linear decision boundary to a desired degree of accuracy. However, best practices suggest scaling of continuous and categorical input to $[-1; 1]$, output data to match the range of the activation functions, i.e. $[0; 1]$ or $[-1; 1]$, and avoidance of ordinal coding [19] to facilitate learning speed and robustness. Despite their significant attention and application, only limited research on the impact of DPP decisions of scaling, coding and sampling on data mining performance exists.

2.2. Decision trees

DT are intuitive methods for classifying a pattern through a sequence of rules or questions, in which the next question depends on the answer on a current question. They are particularly useful for categorical data, as rules do not require any notion of metric. A variety of different DT paradigms exists, such as ID3, C4.5, CART or CHAID. A popular approach to DT modelling induces decision trees based on the information theoretical concept of entropy [30]. Depending upon the proportion of examples of class -1 and $+1$ in the sample, a tree is split into nodes on the attribute which maximises the expected reduction of entropy. The tree is constructed with recursive partitioning of successive splits. A rule set can be formulated by derivation of a rule for each path from the tree's root to a leaf node. Due to the recursive growing strategy, DT tends to overfit the training data, constructing a complex structure of many internal nodes. Consequently, overfitting is controlled through retrospective pruning procedures for deleting redundant parts of rules [30,31]. Extending the case of binary classification, DT permit the prediction of a conditional probability of class membership using the concentration of class $+1$ records within a node as a ranking criterion. DT are robust to continuous or categorical attributes in the sense that appropriate split criteria for each scaling type exist [31].

2.3. Support vector machines

The original SVM can be characterised as a supervised learning algorithm capable of solving linear and non-linear binary classification problems. Given a training set with m patterns $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^{m}$, where $\boldsymbol{x}_i \in X \subseteq \Re^n$ is an input vector and $y_i \in \{-1, +1\}$ its corresponding binary class label, the idea of support vector classification is to separate examples by means of a maximal margin hyperplane [32]. That is, the algorithm strives to maximise the distance between examples that are closest to the decision surface. It has been shown that maximising the margin of separation improves the generalisation ability of the resulting classifier [33]. To construct such a classifier one has to minimise the norm of the weight vector $\boldsymbol{w}$ under the constraint that the training patterns of each class reside on opposite sides of the separating surface (see Fig. 2). Since $y_i \in \{-1, +1\}$ we can formulate this constraint as

$$y_i((\boldsymbol{w} \cdot \boldsymbol{x}_i) + b) \geqslant 1, \quad i = 1, \ldots, m. \tag{1}$$

Examples which satisfy (1) with equality are called support vectors since they define the orientation of the resulting hyperplane.

To account for misclassifications, that is examples where constraint (1) is not met, the so called soft margin formulation of SVM introduces slack variables ξ_i [32]. Hence, to construct a maximal margin classifier one has to solve the convex quadratic programming problem (2).

$$\begin{aligned} &\min_{w,b,\xi} \frac{1}{2}\|\boldsymbol{w}\| + C\sum_{i=1}^{m}\xi_i \\ &\text{s.t.: } y_i((\boldsymbol{w} \cdot \boldsymbol{x}_i) + b) \geqslant 1 - \xi_i, \quad i = 1, \ldots, m. \end{aligned} \tag{2}$$

C is a tuning parameter which allows the user to control the trade off between maximising the mar-

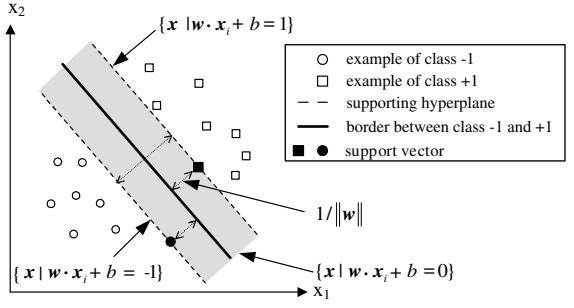


Fig. 2. Linear separation of two classes -1 and $+1$ in two-dimensional space with SVM classifier [34].

gin (first term in the objective) and classifying the training set without error. The primal decision variables $\boldsymbol{w}$ and b define the separating hyperplane, so that the resulting classifier takes the form

$$y(\boldsymbol{x}) = sgn((\boldsymbol{w}^* \cdot \boldsymbol{x}) + b^*), \quad (3)$$

where $\boldsymbol{w}^*$ and b^* are determined by (2).

To construct more general non-linear decision surfaces SVM implement the idea to map the input vectors into a high-dimensional feature space via an a priori chosen non-linear mapping function Φ. Constructing a separating hyperplane in this feature space leads to a non-linear decision boundary in the input space. Expensive calculation of dot products $\Phi(\boldsymbol{x}_i)\cdot\Phi(\boldsymbol{x}_j)$ in a high-dimensional space can be avoided by introducing a kernel function $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \Phi(\boldsymbol{x}_i)\cdot\Phi(\boldsymbol{x}_j)$ [32].

SVM requires specific postprocessing to model conditional class membership probabilities; see e.g. [35]. However, a ranking of customer instances, as is usually required in direct marketing, can be produced by removing the sign function in (3). This gives the distance of an example to the separating hyperplane which is directly related to the confidence of correct classification [35]. Therefore, customer instances that are further apart from the separating surfaces receive a higher ranking.

Research of SVM in conjunction with DPP focuses mainly on data reduction and feature selection in particular, e.g. [36–38]. While some work on the influence of scaling and discretisation of continuous attributes [39–41] exists, the effect of coding of categorical attributes has to our best knowledge not been investigated.

3. Data preprocessing for predictive classification

3.1. Current research in data preprocessing

The application of each data mining algorithm requires the presence of data in a mathematically feasible format, achieved through DPP. Consequently, DPP represents a prerequisite phase for data mining in the process of knowledge discovery in databases. DPP tasks are distinguished in data reduction, aiming at decreasing the size of the dataset by means of instance selection and/or feature selection, and data projection, altering the representation of the data, e.g. mapping continuous variables to categories or encoding nominal attributes [8]. While some of these are imperative for the valid application of a method, such as scaling for NN, others appear to be more general to facilitate method performance in general.

To evaluate the impact of DPP methods on classification accuracy and to derive best practices within the domain, we conduct a structured literature review of publications in corporate data mining applications of classification within the related domains of target selection in direct marketing, including case-based analyses as well as comparative papers evaluating various algorithms on multiple datasets [9]. We analyse each publication regarding the methods applied, whether parameter tuning was conducted, and which DPP methods of data reduction and projection could be observed. The results of our analysis are presented in Table 1.

Our review documents the emphasis on evaluating and tuning competing classification algorithms in a particular data mining task or dataset. In addition, it shows only limited documentation and almost no competitive evaluation of DPP issues within data mining applications. Only 47% of all studies use and document data reduction approaches while only 64% consider data projection in general. Only a single publication provides information on the treatment of categorical attributes, although categorical variables are used and documented in 71% of all studies and commonly encountered in the application and the data mining domain in general. In contrast, information on the respective procedures for parameter tuning is provided in 16 out of 19 publications. Most strikingly, across all surveys only a single DPP technique is applied, ignoring possible alternatives without evaluation or justification. In data projection, only [10,6] evaluate models incorporating discretised as well as standardised alternatives of continuous attributes in their study. Standardisation of continuous attributes are routinely included in experimental setups [10], particularly of NN, their use appears scarce. While the necessity of DPP for data reduction is motivated by the size of the individual dataset, all three authors

Table 1
Data preprocessing activities within publications on corporate data mining

	Input type[a,b]	Methods[c]	Parameter tuning	Data reduction[d]		Data projection		
				FS	RS	Continuous attributes		Categories
						Standardisation	Discretisation	Coding
[2]	2	BMLP, LR, LDA, QDA	X			X		
[42]	1	MLP, LR, CHAID	X			X		
[43]	2	MLP, RBF, LR, GP, CHAID	X		X			
[44]	3	MLP, LR, LDA	X	X				
[4]	2	CHAID, CART			X			
[6]	2	MLP, LR	X	X	X	X	X	
[9]	2	LVQ, RBF, 22 DT, 9 SC	X					X
[45]	2	LDA, LR, KNN, KDE, CART, MLP, RBF, MOE, FAR, LVQ	X				X	
[3]	1	MLP		X		X		
[7]	2	LSSVM	X	X		X		
[11]	2	LR, LS-SVM, KNN, NB, DT	X			X	X	
[10]	1	LDA, QDA, LR, BMLP, DT, SVM, LSSVM, TAN, LP, KNN	X				X	
[46]	2	LR, MLP, BMLP	X	X				
[47]	2	LSSVM, SVM, DT, RL, LDA, QDA, LR, NB, IBL	X			X		
[48]	1	DT, MLP, LR, FC	X					
[49]	1	FC	X			X		

[a] Type 1: only continuous; 2: continuous and categorical; 3: only categorical.

[b] Some publications provide no detailed information about the type or scaling level of their variables. Considering the fact that demographic customer data consist mostly of categorical variables, we assume that any experiment that includes demographic customer information together with transaction oriented data has to deal with continuous as well as categorical variables. Binary variables are considered as categorical ones.

[c] BMLP: Bayesian learning MLP, CART: classification and regression tree, CHAID: Chi-square automatic interaction detection, FAR: fuzzy adaptive resonance, FC: fuzzy classification, GP: genetic programming, IBL: instance based learning, KDE: kernel density estimation, KNN: K-nearest neighbor, LDA: linear discriminant analysis, LP: linear programming, LR: logistic regression, LVQ: learning vector quantisation, MLP: multilayer perceptron, MOE: mixture of experts, NB: Naïve Bayes, QDA: quadratic discriminant analysis, RBF: radial basis function NN, RL: rule learner, SC: statistical classifiers (e.g. LDA, LR, etc.), LSSVM: least squares SVM, TAN: tree augmented Naïve Bayes.

[d] FS: feature selection; RS: resampling.

that make use of instance selection techniques evaluate only one single procedure.

As the choices of DPP depend on the individual dataset used, the lack of DPP may be contributed to the use of ready preprocessed, 'toy' datasets. However, we may conclude that the potential impact of DPP decisions on the predictive performance of classification methods has neither been analysed nor systematically exploited. Particular recommendations exist for selected algorithm classes, which must not hold for other methods. However, only a single DPP scheme is utilised to compare classifier performance, possibly biasing the evaluation results. Consequently, the suitability of different DPP approaches for different methods within a specific task, as well as the sensitivity of data mining algorithms towards DPP in general, requires further investigation. We present an overview of the relevant methods in data reduction and data projection for DPP, which will later be evaluated in a comprehensive experimental setup.

3.2. Data reduction

Data reduction is performed by means of feature selection and/or instance selection. Feature selection aims at identifying the most relevant, explanatory input variables within a dataset [14]. In addition to improving the performance of the predictors, feature selection facilitates a better understanding of the underlying process that generated the data. Also, reducing the feature-vector condenses the size of the dataset, accelerating the task of training a classifier and thereby increasing computational efficiency [13]. Feature selection methods are categorised as wrappers and filters [50]. While filters make use of designated methods for feature evaluation and construction, e.g. principal component analysis [51] and factor analysis [52], wrappers utilise the particular learning algorithm to assess selected feature subsets heuristically by means of the resulting prediction accuracy. In general, wrapper-based approaches have proven more popular for direct marketing applications; see e.g. [3,7,12]. Feature selection appears to be well researched and established in data mining practice as for enhancing individual methods [13,14]. Therefore we limit our experiments on the effects of less analysed DPP choices, disregarding the impact of feature selection from further analysis.

The selection of data instances through resampling techniques often represents a prerequisite for data mining, establishing computational feasibility on large datasets or ensuring unbiased classification on imbalanced datasets. Particularly in empirical domains of corporate response modelling, such as direct marketing, fraud detection, etc., the number of instances in the interesting minority class is significantly smaller than of the majority class. For example, the number of customers who respond to a mail offer is usually very small compared to the overall size of a solicitation [4,5,46] so that the target class distributions are highly skewed. These imbalances obstruct classification methods by biasing the classifier towards the majority class [53] requiring specific DPP treatment to diminish negative effects. Popular approaches to account for imbalances without modifying the classifier are random oversampling of the minority class or random undersampling of the majority class, respectively [54,55]. Additionally, sophisticated techniques have recently been proposed, e.g. the removal of noisy, borderline and redundant training instances of the majority class [16] or the creation of new members of the minority class as a mixture of two adjacent class members [15].

3.3. Data projection

Data projection aims at transforming raw data into a feasible, beneficial representation for a particular classification algorithm. It comprises techniques of value transformation, e.g. mapping of categorical variables and discretisation or scaling of continuous ones. Working with large attribute sets of mixed scale, data mining routinely encounters mixtures of categorical and continuous attributes. Consequently, the combination of different data projection approaches offers vast degrees of freedom in the DPP stage.

Continuous attributes may be preprocessed using various forms of discretisation or standardisation, of which we present the most common variants. Discretisation or binning represents a transformation of continuous attributes into a limited set of values (bins), thereby suppressing noise and removing outlier values. Each raw value x_i is uniquely mapped to a particular symbol s_i, e.g. $s_i = 1$ for $x_{\min} < x_i \leqslant x_{c1}$, $s_i = 2$ for $x_{c1} < x_i \leqslant x_{c2}$, $s_i = 3$ for $x_{c2} < x_i \leqslant x_{\max}$, thus deriving a set of artificially created ordinal attributes from metric variables. With a higher quantity of used symbols, more details of the original attributes are captured in the transformed dataset. Obviously, the resulting dataset depends on the definition of the critical boundaries x_c between two adjacent symbols. As an unfavourable choice of values may lead to a loss of meaningful information [40,41], the DPP choice of discretisation is not without risk. Popular variants of discretisation are analysed [18], confirming their relevance for classifier performance. Alternatively, standardisation of continuous attributes (4) ensures that all scaled attributes values $\hat{x}_i$ reside in a similar numerical range [21]:

$$\hat{x}_i = \frac{x_i - \bar{x}_i}{\sigma_{x_i}} \tag{4}$$

Table 2
Schemes for encoding categorical attributes

Ordinal raw value	N encoding			$N-1$ encoding		Thermometer encoding			Ordinal encoding
	x_1	x_2	x_3	x_1	x_2	x_1	x_2	x_3	x_1
High	1	0	0	0	0	1	0	0	1
Medium	0	1	0	1	0	1	1	0	2
Low	0	0	1	1	1	1	1	1	3

with mean $\bar{x}_i$ and standard deviation σ_{x_i} of all realisations of attribute x_i, this approach is sensitive to outlier values but avoids the creation of additional features that increase the dimensionality of the dataset.

While variants for data projection of continuous attributes receive selected attention, variants for numerical mapping of categorical attributes or data conversion are largely neglected. Several encoding schemes are feasible, which are exemplified in Table 2 for three ordinal values on a N encoding, $N-1$ encoding, thermometer code and ordinal encoding scheme using one to three binary (dummy) variables [8,19,56].

After mapping original data by means of reasonable transformation rules and encoding schemes, scaling procedures transform values of each variable into an interval being appropriate to a particular classification algorithm. Typical intervals are $[-1;1]$ and $[0;1]$, either with binary values only or with real values, depending on the encoding scheme.

4. Case study of data preprocessing in direct marketing

4.1. Experimental setup

We analyse the impact of individual DPP choices on classification performance in a structured experiment, based upon the characteristics of an empirical dataset from a previous direct mailing campaign conducted in the publishing industry. The objective is to evaluate customers for cross-selling, identifying those most likely to buy an additional magazine subscription from all customers already subscribed to at least one periodical. The original campaign contacted 300,000 customers, of which 4019 ordered a new subscription. The response rate of 1.4% is considered representative for the application domain. The dataset characterises each customer instance by 28 attributes of nominal scale, e.g. flags identifying email, previous merchandising treatment, etc., categorical scale, such as age group, order channel, etc., and continuous scaling level, including the total number of subscriptions, number of cancellations, overall revenue, etc. The binary target variable identifies a customer as one of the 4019 responders (1) or as a non-responder (-1). The significantly skewed target class distribution and the mixed scaling level of potentially valuable customer attributes poses particular challenges to be addressed using DPP. Therefore, projection of categorical attributes, discretisation or scaling of continuous ones as well as resampling are of primary importance. Regarding the moderate number of attributes, the wealth of previous research and the scope of our analysis, we omit feature selection from our study.

An explorative analysis reveals the presence of outlier values in some of the continuous attributes, e.g. customer instances with 253 inactive subscriptions in contrast to and average of 0.8. As binning may diminish the effect of outliers while scaling remains sensitive to extreme values, we create two sets of experiments implementing discretisation as in [18] versus standardisation. For categorical attributes we consider the four encoding schemes of Table 2. To evaluate possible effects of scaling into different intervals, we run two sets of experiment setups, scaling all attributes to $[0;1]$ and $[-1;1]$, respectively. Finally, we evaluate the impact of over- and undersampling [54] to counter class imbalance between responders and

Table 3
Identification of experimental setups—sampling, encoding and scaling of attributes

	Oversampling								Undersampling							
	N		$N-1$		Temperat.		Ordinal		N		$N-1$		Temperat.		Ordinal	
	0	−1	0	−1	0	−1	0	−1	0	−1	0	−1	0	−1	0	−1
Experiment #ID																
Discretisation	#1	#2	#3	#4	#5	#6	#7	#8	#9	#10	#11	#12	#13	#14	#15	#16
Standardisation	#17	#18	#19	#20	#21	#22	#23	#24	#25	#26	#27	#28	#29	#30	#31	#32
No. of attributes[a]																
Discretisation	117	117	90	90	117	117	29	29	117	117	90	90	117	117	29	29
Standardisation	88	88	70	70	88	88	29	29	88	88	72	72	88	88	29	29

[a] Varying attribute numbers result from applying different encoding schemes (see Table 2).

non-responders, aiming to increase classifier sensitivity for the economically relevant minority class 1.

The resulting 32 experiments (Table 3) are evaluated applying a hold-out method, requiring three disjoint datasets for training, validation and testing. While training data is used to parameterise each classifier, the second set is used for model selection and to prevent overfitting through early stopping for NN. The trained and selected classifiers are tested out-of-sample on an unknown hold-out set to evaluate their classification performance as an indication of their ability to generalise on unknown data. To ensure comparability all datasets contain the same records over all experiments, differing only in data representation according to the respective DPP treatment. To separate balanced datasets, we randomly select 65,000 records for the test set, leading to a statistically representative asymmetric class distribution of 1.4% responders (912 class 1) to 98.6% non-responders (64,088 class −1). In order to facilitate full usage of the remaining 3107 responders, 66.6% (2072) are randomly assigned to the training set with 33.3% (1035) assigned to the validation set. Using strategies of oversampling versus undersampling, different sizes of the training and validation datasets are created through resampling of responders and non-responders until equally distributed class sizes are achieved. In undersampling, 2072 records of non-responders are randomly chosen for the training set until their number equals that of responding customers, with 1035 records for the validation set, respectively. For oversampling, 20,000 and 10,000 records of inactive customers are randomly chosen for the training and validation set, while responders are randomly duplicated to equal the number of non-responders in each set. The size of the individual data subsets is chosen to balance the objective of learning to accurately predict responders from the training set while keeping datasets computationally feasible. The resulting datasets are summarised in Table 4.

Table 4
Dataset size and structure for the empirical simulation—over-/undersampling approaches

Data subset	Data partition (number of records)			
	Oversampling		Undersampling	
	Class 1	Class −1	Class 1	Class −1
Training set	20,000	20,000	2072	2072
Validation set	10,000	10,000	1035	1035
Test (hold-out) set	912	64,088	912	64,088

4.2. Method parameterisation

Each experimental setup is evaluated using different parameterisations for each classifier to account for possible interactions between method tuning and the effects of the multifactorial design of sampling, coding and scaling on predictive performance.

With regard to the large degrees of freedom and the considerable computational time of over 3 hours for MLP training, we conduct a pre-experimental sensitivity analysis to heuristically identify a suitable subset of parameters from hidden nodes,

activation functions, learning algorithms, etc. We limit the experiments to architectures using $n_i = 25$ hidden nodes and two sets of activation function in the hidden layer $\text{act}_j = \{\tanh, \log\}$, using a softmax output-function on the two nodes in the output layer to model the conditional probability of class membership for each pattern in order to rank each customer instance according to its probability of belonging to class 1. Each NN is initialised four times and trained up to a maximum of 10,000,000 iterations, evaluating the performance on the validation set after every epoch for early stopping. We apply the Delta–Bar–Delta learning rule, using autoadaptive learning parameters for each weight w_{ij} to further limit the degrees of freedom. For SVM modelling, we consider alternative regularisation parameters C in the range $\log(C) = \{-3, -2, -1, 0\}$ and kernel parameters $\log(\sigma^2) = \{-3, -2\}$, derived from a previous grid search for a Gaussian kernel function. The selection of the Gaussian kernel is motivated by previous results [57] and a pre-experimental analysis, indicating computational infeasibility of polynomial kernels with training times of over 72 hours on the oversampled datasets. Degrees of freedom in C4.5 parameterisation are mainly concerned with pruning, to guide the process of cutting back a grown tree for better generalisation. We consider the standard pruning procedure together with reduced-error pruning and vary the confidence threshold in the range of $\{0.1, 0.2, 0.25, 0.3\}$ [58].

We compute a total of 768 classifiers for each data subset, relating to 256 results per NN, SVM and DT each, and corresponding to 32 groups of 8 observations per dataset and method, i.e. 384 results for each scaling effect, 384 experiments per sampling effect, 192 experiments per coding effect of categorical attributes and 384 experiments of coding continuous variables. This leads to a total of 2304 classification results evaluated across three performance measures in order to test the effect of factors and factor combinations independent of method parameterisation. All experiments are carried out on 3.6 GHz Pentium IV workstation with 4GB main memory. The WEKA software library [58] is used to model tree classifiers, taking an average of 4 minutes to build a DT. In contrast, parameterising SVM takes on average 20 minutes per experiment for undersampling and 2 hours for oversampling using the LIBSVM package [59]. MLP are trained using Neural Works Professional II+, taking 25 minutes for undersampling and on average 3 hours, depending on the early stopping of each initialisation. In total, experimental runtime consists of 34 days excluding pre-experiments, setup and evaluation.

4.3. Performance metrics for method evaluation

A variety of performance metrics exists in data mining, direct marketing and machine learning, permitting an evaluation of DPP effects by alternative performance metrics. As certain metrics provide biased results for imbalanced classification [60], we limit potential biases by evaluating the impact of DPP on three alternative performance metrics established in business classification problems [57]. Classifier performance is routinely assessed using a confusion matrix of the predicted and actual class memberships (see Table 5).

Performance metrics calculate means of the correctly classified records within each class to obtain a single measure of performance such as arithmetic (AM) or geometric mean (GM) classification rates

$$\text{AM} = \frac{1}{2}\left(\frac{h_{00}}{h_{0.}} + \frac{h_{11}}{h_{1.}}\right); \quad \text{GM} = \sqrt{\left(\frac{h_{00}}{h_{0.}} \cdot \frac{h_{11}}{h_{1.}}\right)}. \quad (5)$$

While these performance metrics assess only the capability of a binary classifier to separate the classes without error, they do not take a classifier's ability to rank instances by their probability of class membership into consideration. As direct marketing applications need to identify customers ranked by the highest propensity to buy, given a

Table 5
Confusion matrix for binary classification problem with output domain $\{-1, +1\}$

		Predicted class		$\sum$
		-1	$+1$	
Actual class	-1	h_{00}	h_{01}	$h_{0.}$
	$+1$	h_{10}	h_{11}	$h_{1.}$
	$\sum$	$h_{.0}$	$h_{.1}$	L

varying constraint of the size of a possible mailing campaign, a lift analysis reflects a more appropriate approach to evaluate response models [53,61,62]. Using a classifier to score customers according to their responsiveness from most likely to least likely buyers, the lift reflects the redistribution of responders after the ranking, with superior classifiers showing a high concentration of actual buyers in the upper quantiles of the ranked list. Hence, the lift evaluates a classifier's capability to identify potential responders and measures the improvement over selecting customers for a campaign at random. Given a ranked list of customers S with known class membership a lift index is calculated as

$$\text{Lift} = (1.0 \cdot S_1 + 0.9 \cdot S_2 + \cdots + 0.1 \cdot S_{10}) \Big/ \sum_{i=1}^{10} S_i \tag{6}$$

with S_i denoting the number of responders in the ith decile of the ranked listed. An optimal lift provides a value of 1 with $S_1 = \sum_i S_i < 10\%$, while a random selection of customers would result in a lift of 50% [53].

We evaluate the impact of DPP on classifier performance using the performance metrics of AM, GM and lift index. As individual classifiers use particular error metric to guide their parameterisation processes, such as early stopping of NN on AM, or the selection of a best parameterisation on the validation set, this may induce an additional bias if evaluated on a inconsistent metric. To confirm the robustness of our experiments and the appropriateness of analysing the results using a single performance metric, we analyse Spearman's rho non-parametric correlations between the individual metrics across all experiments and all datasets. The analysis reveals consistent, positive correlations significant at a 0.01 level, indicating a mean correlation of 0.775 between GM, AM and lift index across all datasets of training, validation and test for each method. Consequently, the use of an arbitrary performance metric seems feasible, utilising the AM for parameterisation where the lift metric is inapplicable as an objective function. The lift is used for out of sample evaluation across all methods to reflect the business objective. In order to adhere to space restrictions and to present results in a coherent manner for both the direct marketing and the machine learning domains, unless otherwise stated we provide results using the out-of-sample lift index. However, all presented results on the impact of DPP upon the classification performance also hold for alternative performance metrics.

5. Experimental results

5.1. Impact of data preprocessing across classification methods

We calculate the lift index of SVM, NN and DT across 32 experimental designs of different DPP variants and across three datasets of training, validation and test data, visualised in Fig. 3.

To quantify the impact and significance of each DPP candidate on the classification performance of different methods, we conduct a multifactorial analysis of variance with extended multi comparison tests of estimated marginal means across all methods and for each of the three methods separately. The experimental setup assures a balanced factorial design, modelling each DPP variant as different factor treatment of equal cell sizes. Sampling, scaling, coding of continuous attributes, coding of categorical attributes and the method are modelled as fixed main effects to test whether the factor levels show different linear effects on the dependent variables, the classification lift index on the training, validation and test datasets. In addition, we investigate ten 2-fold, ten 3-fold, five 4-fold and one 5-fold non-linear interaction effects between factors. We consider factor effects as relevant if they prove consistently significant at a 0.01 level of significance using Pillai's trace statistic across all datasets. In addition, a factor needs to prove significant for the individual test set to indicate an consistent out-of-sample impact independent of the data sample. We disregard a significant Box's test of equality and a significant Levene statistic of indifferent group variances due to the large dataset, equal cell sizes across all factor-level-combinations and ex postanalysis of the residuals revealing no violations of the

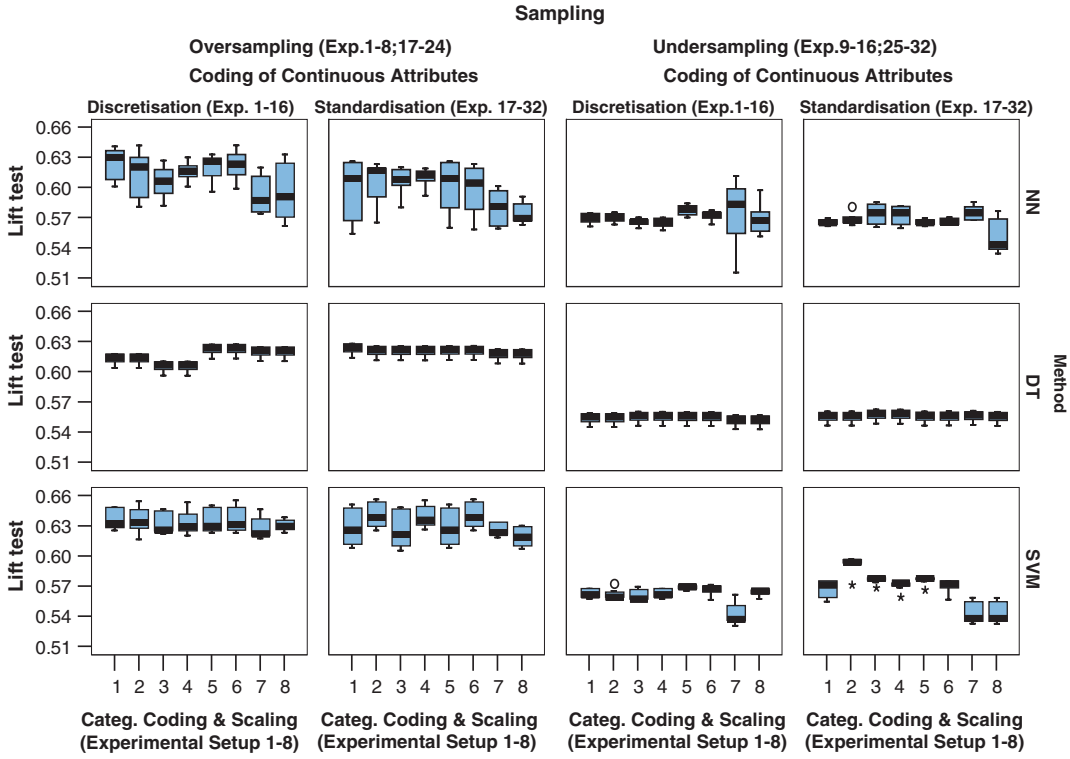


Fig. 3. Boxplots of lift performance on the test sets for NN, DT and SVM across 32 experimental setups of sampling, scaling, coding of categorical and coding of continuous attributes. Boxplots provide median and distributional information, additional symbols of stars and circles indicate outliers and extreme values. Higher lift values indicate increased accuracy.

underlying assumptions. The individual contribution of each main factor and their interactions to explaining a proportion of the total variation is measured by a partial eta squared statistic (η), with larger values relating to higher relative importance. To contrast the impact of each factor levels within each factor we conduct a set of posthoc multi comparison tests using Tamhane's T2 statistics, accounting for unequal variances in the factor cells. This evaluates the positive or negative impact of each factor level on the classification accuracy of lift across the data subsets by estimated marginal means, $\text{mm}_i = \{\text{training; validation, test}\}$, with positive impacts indicating increased accu-

Table 6
Significance of DPP main effects by individual datasets and individual methods using Pillai's trace

Factors	Significance by dataset				Significance by method		
	All	Train	Valid	Test	NN	SVM	DT
Method	0.000**	0.000**	0.000**	0.000**	–	–	–
Scaling	0.077	0.011*	0.092	0.343	No	No	No
Sampling	0.000**	.000**	0.000**	0.000**	Yes	Yes	Yes
Continuous coding	.000**	0.000**	0.000**	0.153	Yes	No	Yes
Categorical coding	0.000**	0.000**	0.000**	0.000**	Yes	Yes	Yes

* Significant at the 0.05 level (2-tailed).
** Highly significant at the 0.01 level (2-tailed).

racy and vice versa. Table 6 presents a summary of the findings by dataset across all methods and for each method individually.

The main factors of sampling ($\eta = 0.958$), method choice ($\eta = 0.392$) and coding of categorical attributes ($\eta = 0.108$) prove significant at a 0.01 level in the order of their relative impact, while the effect of scaling and the coding of continuous attributes prove just insignificant. In addition, all two-way interactions of the significant main effects led by sampling * method ($\eta = 0.404$) and one three-way interaction of method * sampling * categorical prove significant. This confirms a significant impact of DPP through different levels of sampling, coding of categorical attributes and coding of continuous attributes on out-of sample model performance for the case study dataset. In addition, the significant impact proves consistent across alternative methods. However, no significant impact of different scaling ranges for continuous and categorical variables can be validated.

In order to determine the size and positive or negative direction of each DPP choice upon classification performance, we analyse the treatments of the significant factors in more detail. In addition, the analysis indicates interaction effects between the used classification methods and selected DPP factor levels of varying significance and impact. As this indicates method specific reactions to individual DPP factor levels, we need to analyse the impact of the factor effects in separate multifactorial ANOVA analyses for each method.

5.2. Impact of sampling on method performance

To further investigate the significant impact of over- versus undersampling we analyse the estimated marginal means of the classification performance for NN, SVM and DT separately. Regarding undersampling, the results across NN, SVM and DT are consistent and confirm an increased performance across training and validation datasets and a severely decreased performance on the test set. The impact of undersampling versus oversampling for NN is estimated at $mm_{NN} = \{0.088; 0.081; -0.035\}$, indicating a -3.5% drop in lift accuracy, for SVM at $mm_{SVM} = \{0.071; 0.078; -0.068\}$ and for DT at $mm_{DT} = \{0.035; 0.033; -0.063\}$. As already a 1% increase in out-of-sample accuracy is regarded as economically relevant due to the highly asymmetric costs in the problem domain, the use of undersampling would induce a significant monetary loss. In addition, the marginal means in Fig. 4 indicate a stronger impact of undersampling on SVM and DT than on NN.

Our analysis clearly identifies undersampling as suboptimal to oversampling across all methods, leading to significantly increased yet irrelevant in-sample performance at the cost of decreased out-of-sample performance regardless

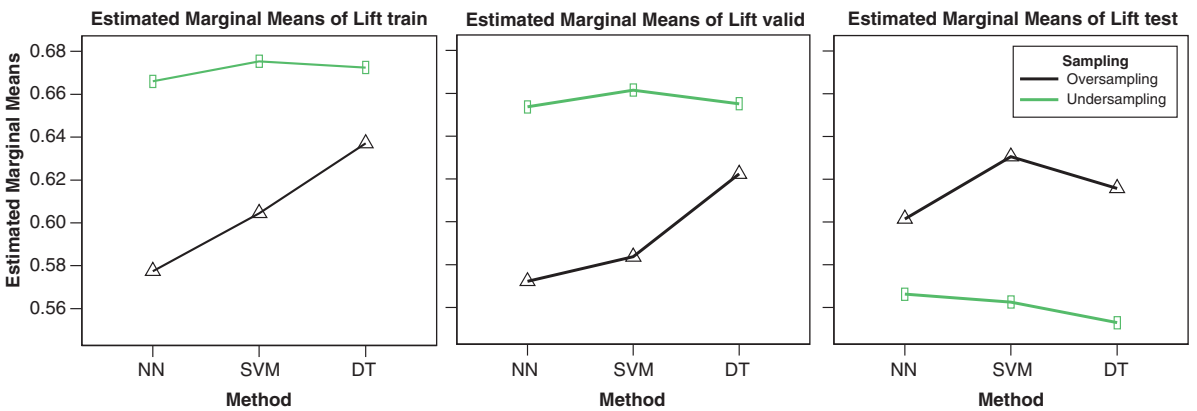


Fig. 4. Estimated marginal means plots of the test set performance of two sampling factor treatments of oversampling (△) and undersampling (□) across different classification methods of NN, SVM and DT.

Table 7
Spearman's rho non-parametric correlation coefficients between datasets for sampling variants

Spearman's rho		NN correlations			SVM correlations			DT correlations		
		Train	Valid	Test	Train	Valid	Test	Train	Valid	Test
Oversampling	Train	1.000	0.912**	0.858**	1.000	0.594**	0.762**	1.000	0.778**	0.775**
	Valid	0.912**	1.000	0.786**	0.594**	1.000	0.803**	0.778**	1.000	0.671**
	Test	0.858**	0.786**	1.000	0.762**	0.803**	1.000	0.775**	0.671**	1.000
Undersampling	Train	1.000	0.985**	−0.307**	1.000	0.878**	−0.540**	1.000	0.970**	−0.626**
	Valid	0.985**	1.000	−0.329**	0.878**	1.000	−0.631**	0.970**	1.000	−0.639**
	Test	−0.307**	−0.329**	1.000	−0.540**	−0.631**	1.000	−0.626	−0.639	1.000

* Correlation is significant at the 0.05 level (2-tailed).
** Correlation is highly significant at the 0.01 level (2-tailed).

of the classification method. The selective increase on in-sample performance indicates overfitting instead of learning to generalising for unseen instances from the training data. Regardless of any computational advantages of undersampling due to the reduced sample size, undersampling seems inapplicable in contrast to the time demanding oversampling for the case study dataset. In addition to the inferior accuracy, undersampling induces inconsistencies in selecting 'best' candidate parameterisations for each method. A correlation analysis confirms high correlations between training, validation and test performance for oversampling in contrast to a negative correlation on the out of sample test set for undersampling, see Table 7.

Consequently, classifiers with a high performance on out-of-sample data cannot reliably be selected based upon superior in-sample performance, indicating undersampling as unsuitable for the given imbalanced classifications problem. In contrast, oversampling promises a valid and reliable selection of favourable SVM, NN or DT parameterisations on the validation set to facilitate a high out of sample performance. Considering the lack of generalisation and suboptimal results, we exclude undersampling from further analysis.

5.3. Impact of coding on method performance

After eliminating the dominating factor level of undersampling from the analysis design, we evaluate the effects of coding of categorical and continuous variables across the three methods. Only the coding of categorical variables remains significant for SVM ($\eta = 0.066$). A multiple comparison test confirms a negative impact of ordinal encoding on SVM lift performance of $mm_{SVM} = \{-0.014; -0.002; -0.009\}$ in contrast to a homogeneous subset of all other categorical coding schemes of N, $N-1$ and temperature showing no significant impact. This seems particularly surprising, considering the induced multicollinearity through N encoding. Considering the insignificant differences on classification performance by discretisation or standardisation of continuous attributes, we derive that SVM perform indifferent of binning of metric variables, scaling in different intervals, and N, $N-1$ or temperature encoding of categorical attributes on the given dataset.

In contrast to SVM, both the coding of continuous attributes ($\eta = 0.173$) and the coding of categorical attributes ($\eta = 0.131$) have a significant impact on NN out-of-sample accuracy at a 0.01 level, while no interaction of both coding schemes is observed. An analysis of the marginal means reveals a negative impact of standardisation of continuous variables $mm_{NN} = \{-0.011; -0.009; -0.014\}$ in contrast to discretisation. As with SVM, a multiple comparison test of individual factor levels of categorical coding reveals two homogeneous subsets and a significant, negative impact of ordinal encoding on lift accuracy of $mm_{NN} = \{-0.013; -0.006; -0.024\}$. The negative impact of ordinal coding is considerably larger than for SVM, confirming NN sensitivity to ordinal coding [19]. The impacts of all other factor levels of N, $N-1$ and temperature coding prove

insignificant. Scaling of variables remains insignificant for NN performance. These results seem interesting, considering the frequent assumption that NN learning may benefit from metric variables, and that the limited research conducted by [19] indicates the benefits of scaling to $[-1;1]$ intervals. More specifically, it indicates a dataset specific need for analysis of DPP choices in using NN.

For DT only categorical coding of attributes ($\eta = 0.350$) and its interaction with different continuous codings ($\eta = 0.280$) prove significant, while the main effects of continuous coding or scaling are not significant. In contrast to SVM and NN, an analysis of the marginal means provides inconsistent results, indicating a small but significant decrease in performance of $N-1$ coding of $\mathrm{mm}_{\mathrm{DT}} = \{-0.004; -0.001; -0.004\}$ in contrast to N-coding, a significant increase in performance of temperature encoding of $\mathrm{mm}_{\mathrm{DT}} = \{0.003; 0.004; 0.004\}$ in contrast to N-coding and no significant impact of ordinal encoding. This is attributed to an observed interaction effect of categorical with continuous encoding, as apparent in Fig. 5 at method DT. While no impact is apparent for standardised continuous attributes, a strong negative effect of N and $N-1$ encoding becomes visible for discretised continuous attributes, contrasted by a strong positive effect on the accuracy using temperature or ordinal coding.

In contrast, the plots of marginal means show no interaction between coding categorical and continuous attributes for NN and SVM, with consistently inferior classification results of standardisation for NN but not for SVM. While the impact of scaling remains statistically insignificant for all methods, our analysis indicates that scaling to the interval $[0;1]$ consistently improves out of sample accuracy across NN and SVM, while leaving DT unaffected. However, these results are just insignificant at a 0.05 level. In addition, interactions of scaling, continuous coding and categorical coding emerge for NN. For all standardised and discretised attributes of interval scale, all categorical coding schemes improve test lift when scaled to $[0,1]$. However, N encoding of discretised attributes displays pre-eminent performance when scaled to $[-1;1]$, while scaling to $[0,1]$ decreases out of sample accuracy by 1.5%. In contrast, SVM and DT are generally unaffected by these interaction effects.

5.4. Implications of data preprocessing impact on method performance

As a conclusion from the analysis across various alternative architectures and parameterisations, we determine undersampling to be inferior DPP alternative for NN, SVM and DT. Ordinal coding of categorical variables appears to be a

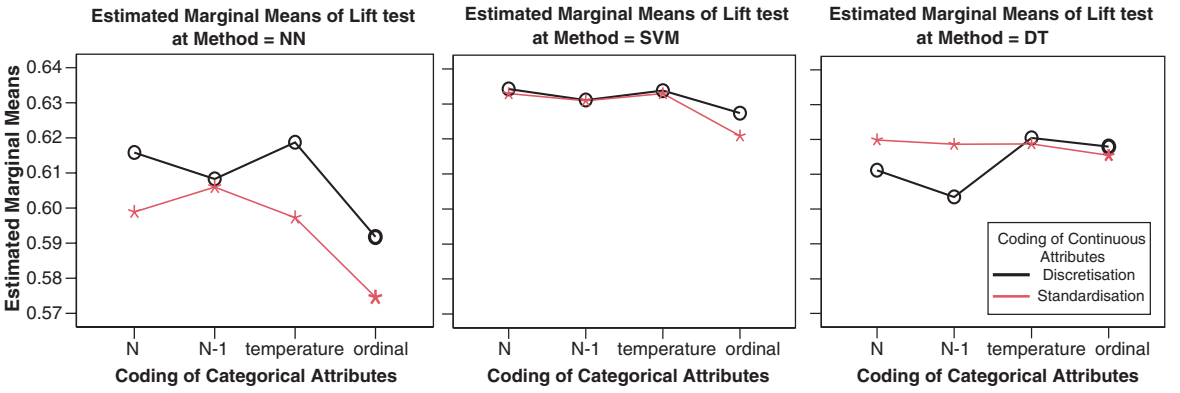


Fig. 5. Plots of the estimated marginal means of lift performance on the test set resulting from continuous coding schemes of discretisation (○) and standardisation (∗) across different categorical coding schemes of N, $N-1$, temperature and ordinal encoding, for each method of NN, SVM and DT.

suboptimal DPP choice for SVM and NN but has no effect on DT classification. Standardisation of continuous attributes is inferior to discretisation for NN, given the case study dataset induced by outliers in the data. As neither temperature scaling, N nor $N-1$ coding of categorical attributes show a significant impact on classification performance across datasets and methods, we propose the use of $N-1$ encoding. $N-1$ encoding reduces the size of the input vector, resulting in a lower dimensional classification domain and increased computationally efficiency through reduced training time. Accordingly, we propose standardisation of continuous attributes to reduce input vector length in the lack of negative effect on SVM or DT performance, but not for NN. On the contrary, discretisation of attributes paired with $N-1$ encoding should be avoided for DT. While scaling to $[0,1]$ generally suggests slightly increased performance across all methods and other DPP choices, this in combination with the computationally motivated preference of $N-1$ encoding would simultaneously avoid significantly decreased NN-performance resulting from the interaction effect with scaling for discretised attributes. To summarise, NN provide best results on the given dataset when continuous data is discretised to categorical scale, N-encoded and scaled to $[-1;1]$ using oversampling. In contrast, SVM benefit from standardised continuous attributes, $N-1$ encoding of categorical attributes and scaling to $[0,1]$ while DT are indifferent and may use the same scheme as SVM.

We conclude that in avoiding undersampling and ordinal coding, SVM as NN offer a robust out-of-sample performance equal or better to DT, which is not significantly influenced by preprocessing through different coding or scaling of variables. However, these findings suggest method specific best practices in using DPP to facilitate out of sample performance for different classification methods. Moreover, it implies that different learning classifiers may produce suboptimal results if they are all evaluated on a single, identical dataset with a single, implicit decision for DPP. Therefore, we eliminate the impact of different method parameterisations and evaluate DPP impact on a selected 'best' architecture for NN, SVM and DT.

5.5. Impact of data preprocessing on best classifier architectures

After analysing the effect of DPP across different parameterisations of each method, we omit the impact of modelling decisions from our analysis by selecting a single 'best' architecture for NN, SVM and DT. We select the method setup from the experiments 1–6 and 17–22, avoiding biased results from suboptimal DPP methods of undersampling and single number encoding found in our preceding analysis. In addition, we identify a single architecture setup for each method based upon the highest mean lift performance on the validation data subset. For NN, we select a topology of 25 hidden nodes in a single hidden layer using a hyperbolic tangent activation function. We apply a DPP scheme from experiment setup #2, discretising continuous variables and scaling all $N-1$ encoded attributes to $[-1,1]$, leading to a lift performance of 0.640 on the test set. For SVM, we select DPP scheme #19, standardising continuous variables, encoding all categorical as $N-1$ and scaling them to $[0,1]$. For DT we apply the same DPP scheme #19, resulting in an out-of-sample lift of 0.619. SVM demonstrate best performance, achieving a lift of 0.645 on the test set.

However, these results are based upon our preceding analysis of different DPP variants across all methods and the individual matching of DPP to method. To relate our findings to the effects of DPP on the validity and reliability of results provided in incomplete case studies from our literature analysis, we need to simulate the effect of choosing a single, arbitrary DPP combination of scaling and coding. Consequently, we analyse the lift performance of the 12 dominant DPP setups for SVM, NN and DT across all three data subsets. A successive multivariate ANOVA reveals limited differences of the classification performance between SVM, NN and DT at a 0.05 level. Although an average SVM lift of 0.634 outperforms the mean NN lift of 0.627 by 0.7% and a DT mean lift of 0.616 by 1.8% on the out-of-sample test set, these results prove not significant. An analysis of estimated marginal mean reveals two homogeneous subgroups. DT perform significantly inferior on out-of-sample than NN or SVM, with $\mathrm{mm}_{\mathrm{DT}} =$

$\{0.049; 0.043; -0.011\}$ and $mm_{DT} = \{0.021; 0.042; -0.018\}$, respectively. While the mean performances of SVM and NN are significantly different across training and validation datasets, no significant difference can be confirmed in out-of-sample accuracy (see Fig. 6).

We conclude that SVM and NN significantly outperform DT on the case study dataset, representing a valuable monetary benefit considering the costs attributed to the imbalanced classes in the case study domain. However, neither SVM nor NN significantly outperform each other across different choices of coding of continuous attributes, coding of categorical attributes or scaling. The lack of significant differences between SVM and NN accuracy seems unsurprising in the light of recent publications inconsistently identifying one method as superior over the other, presenting a different winner from one empirical case study to the next. Our experiments indicate one potential influence: the variance induced by different DPP choices towards the out-of-sample performance of NN and SVM. An analysis of the variance of the out-of-sample performances of each method induced by DPP reveals a significant difference, confirmed by Levene's test of equality at a 5% level. While NN provide a reduce mean performance, they also show a reduced variance of the classification performance across competing DPP, indicating more robust results in comparison with increased DPP sensitivity of SVM. SVM provide not only a larger variance of the results, but also promise a higher maximum performance against the risk of a lower minimum performance than NN. Two thirds of the 95% interval of NN lift ranges, from 0.622 to 0.633, overlap with the SVM results from 0.629 to 0.640. Therefore, SVM incorporate all potential NN performances and most mean performances within their range of results, depending on an individual DPP choice. In contrast, the DT interval of 0.611–0.622 clearly proves inferior considering not only mean performance but also robustness of performance across DPP choices. The results prove consistent across different performance metrics of lift, arithmetic mean classification accuracy and geometric mean classification accuracy, provided in Fig. 6. This implies that comparing in-sample and out-of-sample performance between SVM and NN based upon a particular, arbitrarily motivated DPP choice of coding and scaling on a given dataset may lead to arbitrary results of superior performance of a method, favouring either SVM

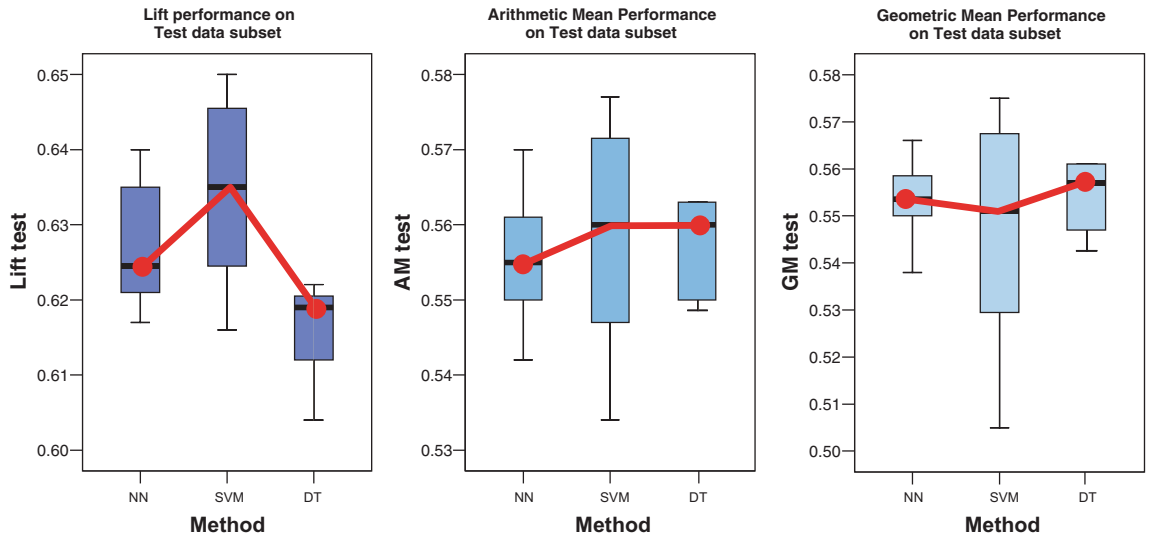


Fig. 6. Boxplots of performances on test data subset for different methods of NN, SVM and DT, displaying mean, across performance measures of lift, AM and GM (from left to right). The estimated marginal means are connected across boxes to highlight mixed patterns of method superiority across performance metrics.

or NN. Although these results are not valid across all possible datasets, they support the importance of DPP decisions with regard to model evaluation. As a consequence, the individual performance of SVM or NN may be increased by evaluating alternative coding, scaling and novel sampling schemes.

Moreover, the variation induced by DPP choices for each classification method is larger than the differences between the methods mean performance. In particular, the impact of DPP on NN and SVM accounts for 50–70% of the variation in accuracy induced by selecting optimal NN architectures, with an average increase of 0.016 through selecting the correct activation function, or SVM parameters, with the impact of selecting significant σ- and C-parameters between 0.004 and 0.021. Considering the variability of performances for SVM and NN depending on adequate DPP, an analysis of alternative preprocessing methods may prove more beneficial in increasing classifier performance than the evaluation of alternative classification methods also sensitive to preprocessing decisions. It is generally accepted within data mining as in operational research, that to derive sound classification results on empirical datasets, alternative candidate methods need to be evaluated, as no single method may be considered generally superior. In addition, our experimental results suggest that avoiding the evaluation of different DPP variants in the experimental designs may limit the validity and reliability of results regarding method performances, possibly leading to an arbitrary method preference.

6. Conclusions

We investigate the impact of different DPP techniques of attribute scaling, sampling, coding of categorical and continuous attributes on classifier performance of NN, SVM and DT in a case-based evaluation of a direct marketing mailing campaign. Supported by a multifactorial analysis of variance, we provide empirical evidence that DPP has a significant impact on predictive accuracy. While certain DPP schemes of undersampling prove consistently inferior across classification methods and performance metrics, others have a varying impact on the predictive accuracy of different algorithms.

Selected methods of NN and SVM prove almost as sensitive to different DPP schemes as to the evaluated method parameterisations. In addition, the differences in mean out-of-sample performance between both methods prove small and insignificant in comparison to the variance induced by evaluating different DPP schemes within each method. This indicates the potential for increased algorithmic performance through effective, method specific preprocessing. Furthermore, an analysis of DPP approaches may not only increase classifier performance of SVM and NN, it may even indicate a higher marginal return in analysing the individual classifiers regarding different DPP alternatives than the conventional approach of evaluation competing classification methods on a single, preprocessed candidate dataset of DPP. Consequently, the choice of a 'superior' algorithm may be supported or even replaced by the evaluation of a 'best' preprocessing approach. Additionally, the performance of NN and SVM across DPP schemes falls within a similar range of predictive accuracy. This suggests that if a dataset is preprocessed in a particular way to facilitate performance of a specific classifier, the results of other classifiers may be negatively biased or produce arbitrary results of method performance. If arbitrary DPP schemes are selected, method evaluation may exemplify the superiority of an arbitrary algorithm, lacking validity and reliability and leading to inconsistent research findings. If however different DPP schemes are evaluated to facilitate the performance of a favoured classifier, the results may even be biased towards prove of his dominance.

The single case-based analysis of DPP prohibits generalised conclusions of enhanced method performance. Considering the almost prohibitive runtime of our experiments on a single dataset, the evaluation on a variety of dissimilar datasets may be infeasible. Additional research may extend the analysis towards a larger set of DPP schemes for selected methods and across different artificial and empirical datasets. However, the significant impact on this representative case raises questions for the validity and reliability of current method

selection practices. The presented results justify the structured analysis of competing sampling, coding and scaling methods—currently neglected from systematic analysis—in order to derive valid and reliable results of the performance of classification methods.

References

[1] E.L. Nash, The Direct Marketing Handbook, second ed., McGraw-Hill, New York, 1992.

[2] B. Baesens, S. Viaene, D. Van den Poel, J. Vanthienen, G. Dedene, Bayesian neural network learning for repeat purchase modelling in direct marketing, European Journal of Operational Research 138 (1) (2002) 191–211.

[3] S. Viaene, B. Baesens, D. Van den Poel, G. Dedene, J. Vanthienen, Wrapped input selection using multilayer perceptrons for repeat-purchase modeling in direct marketing, International Journal of Intelligent Systems in Accounting, Finance and Management 10 (2) (2001) 115–126.

[4] D. Haughton, S. Oulabi, Direct marketing modeling with CART and CHAID, Journal of Direct Marketing 11 (4) (1999) 42–52.

[5] J. Zahavi, N. Levin, Issues and problems in applying neural computing to target marketing, Journal of Direct Marketing 11 (4) (1999) 63–75.

[6] J. Zahavi, N. Levin, Applying neural computing to target marketing, Journal of Direct Marketing 11 (4) (1999) 76–93.

[7] S. Viaene, B. Baesens, T. Van Gestel, J.A.K. Suykens, D. Van den Poel, J. Vanthienen, B. De Moor, G. Dedene, Knowledge discovery in a direct marketing case using least squares support vector machines, International Journal of Intelligent Systems 16 (9) (2001) 1023–1036.

[8] D. Pyle, Data Preparation for Data Mining, Morgan Kaufmann, San Francisco, 1999.

[9] T.-S. Lim, W.-Y. Loh, Y.-S. Shih, A comparison of prediction accuracy, complexity, and training time of thirty-three old and new classification algorithms, Machine Learning 40 (3) (2000) 203–228.

[10] B. Baesens, T. Van Gestel, S. Viaene, M. Stepanova, J. Suykens, J. Vanthienen, Benchmarking state-of-the-art classification algorithms for credit scoring, Journal of the Operational Research Society 54 (6) (2003) 627–635.

[11] S. Viaene, R.A. Derrig, B. Baesens, G. Dedene, A comparison of state-of-the-art classification techniques for expert automobile insurance claim fraud detection, Journal of Risk and Insurance 69 (3) (2002) 373–421.

[12] Y.S. Kim, W.N. Street, G.J. Russell, F. Menczer, Customer targeting: A neural network approach guided by genetic algorithms, Management Science 51 (2) (2005) 264–276.

[13] S. Piramuthu, Evaluating feature selection methods for learning in data mining applications, European Journal of Operational Research 156 (2) (2004) 483–494.

[14] J. Yang, S. Olafsson, Optimization-based feature selection with adaptive instance sampling, Computers and Operations Research, in press.

[15] N.V. Chawla, K.W. Bowyer, L.O. Hall, W.P. Kegelmeyer, SMOTE: Synthetic minority over-sampling technique, Journal of Artificial Intelligence Research 16 (2002) 321–357.

[16] M. Kubat, S. Matwin, Addressing the curse of imbalanced training sets: One-sided selection, in: Proceedings of the 14th International Conference on Machine Learning, 1997.

[17] P. Berka, I. Bruha, Empirical comparison of various discretization procedures, International Journal of Pattern Recognition and Artificial Intelligence 12 (7) (1998) 1017–1032.

[18] U.M. Fayyad, K.B. Irani, On the handling of continuous-valued attributes in decision tree generation, Machine Learning 8 (1) (1992) 87–102.

[19] W.S. Sarle, Neural Network FAQ, 2004, Downloadable from website ftp://ftp.sas.com/pub/neural/FAQ.html.

[20] S. Zhang, C. Zhang, Q. Yang, Data preparation for data mining, Applied Artificial Intelligence 17 (5/6) (2003) 375–381.

[21] C.M. Bishop, Neural Networks for Pattern Recognition, Oxford University Press, Oxford, 1995.

[22] J.A.K. Suykens, J. Vandewalle, Nonlinear Modeling: Advanced Black-box Techniques, Kluwer, Dordrecht, 1998.

[23] K.A. Smith, J.N.D. Gupta, Neural networks in business: Techniques and applications for the operations researcher, Computers and Operations Research 27 (11–12) (2000) 1023–1044.

[24] K.A. Krycha, U. Wagner, Applications of artificial neural networks in management science: A survey, Journal of Retailing and Consumer Services 6 (1999) 185–203.

[25] B.K. Wong, V.S. Lai, J. Lam, A bibliography of neural network business applications research: 1994–1998, Computers and Operations Research 27 (11–12) (2000) 1045–1076.

[26] B.K. Wong, T.A. Bodnovich, Y. Selvi, Neural network applications in business: A review and analysis of the literature (1988–1995), Decision Support Systems 19 (4) (1997) 301–320.

[27] R.D. Reed, R.J. Marks, Neural Smithing: Supervised Learning in Feedforward Artificial Neural Networks, MIT Press, Cambridge, 1999.

[28] J.P. Bigus, Data Mining with Neural Networks: Solving Business Problems from Application Development to Decision Support, McGraw-Hill, New York, 1996.

[29] M.W. Craven, J.W. Shavlik, Using neural networks for data mining, Future Generation Computer Systems 13 (2–3) (1997) 211–229.

[30] J.R. Quinlan, C4.5: Programs for Machine Learning, Morgan Kaufmann, San Mateo, 1993.

[31] R.O. Duda, P.E. Hart, D.G. Stork, Pattern Classification, second ed., Wiley, New York, 2001.

[32] N. Cristianini, J. Shawe-Taylor, An Introduction to Support Vector Machines and Other Kernel-based Learn-

ing Methods, Cambridge University Press, Cambridge, 2000.
[33] V.N. Vapnik, The Nature of Statistical Learning Theory, Springer, New York, 1995.
[34] C.J.C. Burges, A tutorial on support vector machines for pattern recognition, Data Mining and Knowledge Discovery 2 (2) (1998) 121–167.
[35] J.C. Platt, Probabilities for support vector machines, in: A. Smola, P. Bartlett, B. Schölkopf, D. Schuurmans (Eds.), Advances in Large Margin Classifiers, MIT Press, 1999, pp. 61–74.
[36] J. Weston, S. Mukherjee, O. Chapelle, M. Pontil, T. Poggio, V. Vapnik, Feature selection for SVMs, in: Proceedings of the Annual Conference on Neural Information Processing Systems, 2000.
[37] G. Fung, O.L. Mangasarian, Data selection for support vector machine classifiers. in: Proceedings of the 6th International Conference on Knowledge Discovery and Data Mining, 2000.
[38] H. Fröhlich, A. Zell, Feature subset selection for support vector machines by incremental regularized risk minimization, in: Proceedings of the International Joint Conference on Neural Networks, 2004.
[39] C. Edwards, B. Raskutti, The effect of attribute scaling on the performance of support vector machines, in: 17th Australian Joint Conference on Artificial Intelligence, 2004.
[40] R. Kumar, A. Kulkarni, V.K. Jayaraman, B.D. Kulkarni, Symbolization assisted SVM classifier for noisy data, Pattern Recognition Letters 25 (4) (2004) 495–504.
[41] R. Kumar, V.K. Jayaraman, B.D. Kulkarni, An SVM classifier incorporating simultaneous noise reduction and feature selection: Illustrative case examples, Pattern Recognition 38 (1) (2005) 41–49.
[42] R. Potharst, U. Kaymak, W. Pijls, Neural networks for target selection in direct marketing, Technical Report ERS-2001-14-LIS, Erasmus Research Institute of Management (ERIM), Erasmus University Rotterdam, Rotterdam, 2001, Downloadable from website http://ideas.repec.org/p/dgr/eureri/200177.html.
[43] A.E. Eiben, T.J. Euverman, W. Kowalczyk, E. Peelen, F. Slisser, J.A.M. Wesseling, Comparing adaptive and traditional techniques for direct marketing, in: 4th European Congress on Intelligent Techniques and Soft Computing, 1996.
[44] P.M. West, P.L. Brockett, L.L. Golden, A comparative analysis of neural networks and statistical methods for predicting consumer choice, Marketing Science 16 (4) (1997) 370–391.
[45] D. West, Neural network credit scoring models, Computers and Operations Research 27 (11–12) (2000) 1131–1152.
[46] G. Cui, M.L. Wong, Implementing neural networks for decision support in direct marketing, International Journal of Market Research 46 (2) (2004) 235–254.
[47] T. van Gestel, J.A.K. Suykens, B. Baesens, S. Viaene, J. Vanthienen, G. Dedene, B. de Moor, J. Vandewalle, Benchmarking least squares support vector machine classifiers, Machine Learning 54 (1) (2004) 5–32.
[48] S. Madeira, J.M. Sousa, Comparison of target selection methods in direct marketing, in: European Symposium on Intelligent Technologies, Hybrid Systems and their implementation on Smart Adaptive Systems, 2002.
[49] J.M. Sousa, U. Kaymak, S. Madeira, A comparative study of fuzzy target selection methods in direct marketing, in: International Conference on Fuzzy Systems, 2002.
[50] R. Kohavi, G.H. John, Wrappers for feature subset selection, Artificial Intelligence 97 (1–2) (1997) 273–324.
[51] I.T. Jolliffe, Principal Component Analysis, second ed., Springer, Berlin, 2002.
[52] R.L. Gorsuch, Factor Analysis, second ed., L. Erlbaum Associates, Hillsdale, 1983.
[53] C.X. Ling, C. Li, Data mining for direct marketing: Problems and solutions, in: Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining, 1998.
[54] N. Japkowicz, S. Stephen, The class imbalance problem: A systematic study, Intelligent Data Analysis 6 (5) (2002) 429–450.
[55] G.M. Weiss, Mining with rarity: A unifying framework, ACM SIGKDD Explorations Newsletter 6 (1) (2004) 7–19.
[56] M. Smith, Neural Networks for Statistical Modeling, International Thomson Computer Press, London, 1996.
[57] S. Lessmann, Solving imbalanced classification problems with support vector machines, in: Proceedings of the International Conference on Artificial Intelligence, 2004.
[58] I.H. Witten, F. Eibe, Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations, Morgan Kaufmann, San Francisco, 1999.
[59] C.-C. Chang, C.-J. Lin, LIBSVM—A Library for Support Vector Machines, 2001, Downloadable from website http://www.csie.ntu.edu.tw/~cjlin/libsvm.
[60] F. Provost, T. Fawcett, R. Kohavi, The case against accuracy estimation for comparing induction algorithms, in: Proceedings of the 5th International Conference on Machine Learning, 1998.
[61] J. Banslaben, Predictive modelling, in: E.L. Nash (Ed.), The Direct Marketing Handbook, second ed., McGraw-Hill, New York, 1992.
[62] M.J.A. Berry, G. Linoff, Data Mining Techniques: For Marketing, Sales and Customer Relationship Management, second ed., Wiley, New York, 2004.

Literaturempfehlungen:

Frame, J.D.: The New Project Management. 2. Aufl., San Francisco 2002.

Kor, A.: ERP-Systeme zur Verbesserung der Geschäftsprozesse. In: WISU, 31. Jg. (2002), S. 1521 - 1524.

Kor, A.: Ein Projektmanagement-Ansatz zur rechtzeitigen Eskalation bei Abweichungen vom Projektplan. In: International Conference on Operations Research. Gerhard-Mercator-Universität Duisburg, September 2001, Book of Abstracts, S. 139.

Kerzner, H.: Project Management. 6. Aufl., New York 1998.

Kupper, H.: Die Kunst der Projektsteuerung — Qualifikationen und Aufgaben eines Projektleiters. 9. Aufl., München 2001.

Meredith, J.R./Mantel S.J.: Project Management — A Managerial Approach. 4. Aufl., New York 2000.

Project Management Institute (Hrsg.): A Guide to the Project Management Body of Knowledge. PMBOK Guide. 2000 Edition. Newtown Square 2000 (zitiert als: PMBOK 2000).

Zehnder, C.A.: Informatik-Projektentwicklung. 3. Aufl., Zürich 2001.

BASISWISSEN WIRTSCHAFTSINFORMATIK

Customer Relationship Management

In den letzten Jahren haben sich die Umweltbedingungen und insbesondere die Situation auf den jeweiligen Absatzmärkten für viele Unternehmen nachhaltig verändert. Dazu gehören etwa

- das höhere Erwartungs- und Informationsniveau der Kunden,
- der steigende Wettbewerbsdruck durch die Globalisierung und zunehmend gesättigte Absatzmärkte,
- der erhöhte Kostendruck durch höhere Markttransparenz und
- eine abnehmende Kundenloyalität.

Hinzu kommt, dass sich die Produkte verschiedener Hersteller in ihren funktionalen und qualitativen Eigenschaften immer stärker angleichen, so dass die Differenzierung des eigenen Angebots sehr oft über Zusatzleistungen erfolgen muss. Deshalb müssen geeignete Strategien entwickelt werden, um die langfristige Überlebensfähigkeit der Unternehmen zu gewährleisten.

Unter **Customer Relationship Management** (CRM) versteht man eine Management-Philosophie, die den Aufbau und die Pflege **langfristiger** und **profitabler Kundenbeziehungen** zum Ziel hat. Dazu gehört der Einsatz spezialisierter Informationssysteme **(CRM-Systeme)**, die alle kundennahen Prozesse in Marketing, Verkauf und Service unterstützen, die anfallenden Daten sammeln und integriert bereitstellen (vgl. Fink et al. 2001). Die Analyse dieser Daten liefert das nötige Wissen zur kontinuierlichen Verbesserung der Kundengewinnung, Kundenbindung sowie zur Wirtschaftlichkeit und Qualität aller Interaktionen mit den Kunden.

Langfristige Kundenbeziehungen

Die Gewinnung neuer Kunden in zunehmend **gesättigten Märkten** wird immer schwieriger. Dies verdeutlicht die Bedeutung der **Kundenbindung** als Erfolgsstrategie. Grundsätzlich ist von einem Zusammenhang zwischen Kundenbindung und Kundenwert auszugehen. Diese Annahme lässt sich durch Beobachtungen begründen (vgl. Berson et al. 1999 sowie Raab/Lorbacher 2002):

- Die Gewinnung neuer Kunden verursacht höhere Kosten als die Betreuung vorhandener Kunden.
- Die Wiedergewinnung eines Kunden verursacht höhere Kosten als eine von Anfang an zufrieden stellende Betreuung.
- Ein neues Produkt lässt sich einem vorhandenen Kunden leichter verkaufen als einem Neukunden.
- Stammkunden weisen eine geringere Preisempfindlichkeit auf.
- Marketing- und Vertriebskosten sinken mit der Dauer der Kundenbeziehung.

Die Kundenbindung ist für eine profitable Kundenbeziehung also äußerst bedeutsam.

Profitable Kundenbeziehungen

Die Profitabilität eines Kunden lässt sich durch eine Reihe von Maßnahmen steuern. Allerdings ist die Bindungsbereitschaft eines Kunden kaum direkt beeinflussbar, da sie sich aus der Gesamtheit seiner Erfahrungen mit den bisherigen Interaktionen ergibt (vgl. Raab/Lorbacher 2002). Abb. 1 verdeutlicht elementare Wirkungsketten innerhalb einer Kundenbeziehung und deren Auswirkung auf die Profitabilität.

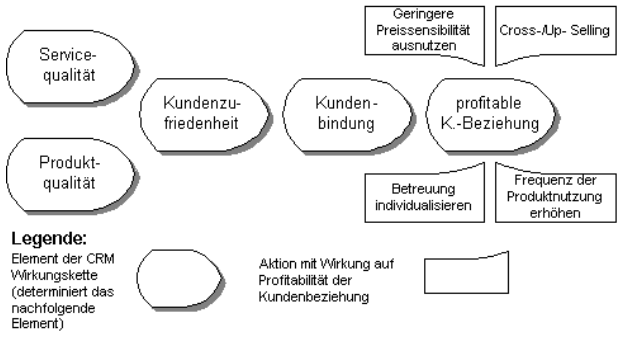


Abb. 1: CRM-Wirkungsketten

Die Gleichbehandlung des gesamten Kundenstamms ist aus betriebswirtschaftlicher Sicht nicht sinnvoll. Zum einen sind die Anforderungen und Erwartungen der Kunden sehr unterschiedlich, zum anderen divergieren Werte und Potenziale der Kunden aus Unternehmenssicht. Die Zuteilung von Ressourcen für Marketing und Kundenbetreuung sollte daher am Wert des Kunden ausgerichtet werden. Im Rahmen des CRM wird dieses Vorgehen als **profitorientierte Segmentierung** bezeichnet.

Relevanz der Kundenbewertung

Wenn der Kundenwert bzw. die Profitabilität eines Kunden als strategischer Aktionsparameter eingesetzt werden soll, ist zu klären, inwiefern eine solche Größe operationalisierbar ist. In Literatur und Praxis haben sich eine Reihe verschiedener Techniken und Modelle zur Kundenbewertung herausgebildet, die von einfachen

ABC-Analysen und Scoring-Modellen über Ansätze aus der Prozesskostenrechnung bis zu komplexen Modellen reichen, die — unter Verwendung statistischer Techniken und/oder intelligenter Planungsverfahren — eine Bewertung aus verschiedenen quantitativen und qualitativen Merkmalen (z.B. Referenzpotenzial) ableiten (vgl. Raab/Lorbacher 2002).

Grundsätzlich sind **detaillierte Kenntnisse** über die eigenen Kunden (Bedürfnisse, Präferenzen, Potenziale) Voraussetzung für jegliche Form der Kundenbewertung. Dies gilt auch gilt für die Personalisierung der Kundenbetreuung.

Aufgaben und Unterstützung durch Informationssysteme

Integraler Bestandteil von CRM ist die Zusammenführung aller kundenrelevanten Daten in einer zentralen Datenbank, einem so genannten **Customer Data Warehouse** (vgl. Berson et al. 1999).

Diese Konsolidierung kundenbezogener Daten ist ausgesprochen komplex. Die relevanten Daten sind im Unternehmen typischerweise auf viele historisch gewachsene Insellösungen zur Unterstützung von Marketing und Verkauf verteilt (z.B. Computer Aided Selling, Online-Datenbanken, Sales-Force-Automation-Systeme, Call Center etc.). Deshalb ist eine Integration mit betriebswirtschaftlichen Standardsoftwaresystemen (Enterprise Resource Planning, Supply Chain Management) erforderlich, welche ebenfalls wichtige Daten und Informationen enthalten können. Durch diese Maßnahmen können alle Unternehmensbereiche auf eine **logische Kundendatenbank** zugreifen, außerdem wird eine ganzheitliche Sicht auf einzelne Kunden oder Kundengruppen gewährleistet.

Neben diesen **integrativen Aufgaben** müssen CRM-Systeme auch **operative Geschäftsprozesse** unterstützen bzw. (teil-)automatisieren. Üblich ist dabei eine Unterscheidung in operatives, kollaboratives und analytisches CRM (vgl. Abb. 2).

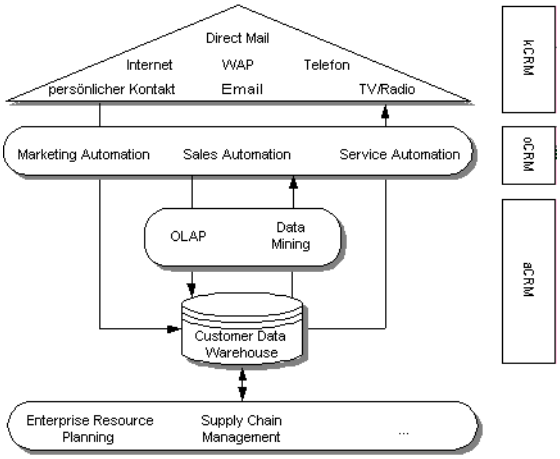


Abb. 2: CRM-Architektur (Quelle: Hippner/Wilde 2002)

— Das **operative CRM (oCRM)** umfasst Lösungen zur Abwicklung und Abstimmung sämtlicher Aktivitäten an den zentralen **Customer Touch Points** Marketing, Verkauf und Service. Der Dialog zwischen Kunde und Unternehmen sowie entsprechende Geschäftsprozesse werden unterstützt. In diesen Bereich fallen z.B. Verkaufsgespräche oder die Beantwortung von Kundenanfragen zu Lieferterminen oder Produktverfügbarkeiten. Zur Bewältigung dieser Aufgaben ist eine Integration mit den so genannten Back-Office-Systemen (Enterprise Resource Planning, Supply Chain Management) eines Unternehmens erforderlich (vgl. Fink et al. 2001).

— **Kollaboratives CRM (kCRM)** beinhaltet die Bereitstellung, Steuerung und Synchronisation verschiedener Kommunikationskanäle zum Kunden (Telefon, Fax, E-Mail etc.). Ziel ist die Sicherung konsistenter Informationen und ein einheitlicher Service-Level bei allen Kanälen (vgl. Fink et al. 2001).

— Die systematische Aufzeichnung und Auswertung aller Kundenkontakte und -reaktionen ist Gegenstand des **analytischen CRM (aCRM)**. Die Daten, welche im Rahmen operativer Tätigkeiten anfallen und im Customer Data Warehouse konsolidiert werden, werden durch OLAP (Online Analytical Processing) und Data Mining in entsprechendes Wissen über den Kunden transformiert.

OLAP Systeme bilden betriebswirtschaftlich relevante Maßgrößen (Umsatz, Absatzzahlen, Kosten) in Form eines multidimensionalen Datenwürfels ab. Die Dimensionen dieses Würfels werden durch betriebswirtschaftlich relevante Gliederungskriterien (Produktgruppe, Kundengruppe, Vertriebsregionen) gebildet (vgl. Hippner/Wilde 2002). Eine typische Fragestellung wäre z.B. „Wie hoch war der Absatz von Produkt X im Zeitraum Y in der Vertriebsregion Z“. Die Antwort entspricht einer Zelle in einem dreidimensionalen OLAP-Würfel mit den Kanten Produkt, Zeit und Region.

Ist der Anwender lediglich in der Lage Hypothesen zu formulieren, ohne eine genaue Kenntnis über Wirkungszusammenhänge zu besitzen (z.B. „der Wert eines Kunden wird durch die Merkmale Alter, Geschlecht und Einkommen beeinflusst“), kann **Data Mining** zur Aufdeckung geschäftsrelevanter Muster in den Daten genutzt werden. Unter Data Mining wird eine (semi-)automatisierte Auswertung großer Datenbestände mittels intelligenter Algorithmen verstanden (vgl. Voß/Gutenschwager 2001). Im Rahmen von CRM kommen verschiedene Analysen zum Einsatz:

- Prognose von Kundenabwanderungswahrscheinlichkeiten
- Prognose von Cross-/Up-Selling-Potenzialen
- Auswahl einer Zielgruppe mit überdurchschnittlicher Reaktionswahrscheinlichkeit für eine Katalogsendung/ein Mailing (Response-Optimierung)

Weitere Einsatzfelder sind z.B. bei Berson et al. 2002 beschrieben.

Das im Rahmen des analytischen CRM gewonnene Wissen fließt anschließend auf die operative Ebene zurück, um dort zur Verbesserung kundenbezogener Geschäftsprozesse beizutragen. Damit ergibt sich ein Regelkreislauf **(Closed Loop Architecture)**, in dem sämtliche Kundenreaktionen genutzt werden, um die Kommunikation mit dem Kunden, die Produkte bzw. Dienstleistungen des Unternehmens und die Servicequalität kontinuierlich zu verbessern und differenziert auf die Kundenbedürfnisse abzustimmen.

Es werden verschiedene CRM-Systeme angeboten, die jedoch stark in ihrem Funktionsumfang variieren. In der Regel wird eine entsprechende Software auch dann als CRM-System deklariert, wenn nur ein kleiner Teil des gesamten Funktionsspektrums (z.B. Kundenkontaktverwaltung) unterstützt wird.

Was ist neu an CRM?

Es zeigt sich, dass CRM viele Ansätze enthält, die in der betrieblichen Praxis schon seit längerem etabliert sind (Konzepte zur Kundenbindung und -bewertung, operative Systeme für Verkauf und Marketing etc.). Die Zusammenfassung dieser Teilbereiche in einem Gesamtkonzept und deren durchgängige Softwareunterstützung stellt das eigentliche Novum von CRM dar. Ermöglicht wurde dies erst durch technologische Innovationen. Neben dem Internet sind hier vor allem Fortschritte bei der Integration von Unternehmensanwendungen **(Enterpri-**

se Application Integration, EAI) und Data Mining sowie deren Umsetzung in Standardsoftware-Produkte zu nennen.

Dipl.-Kfm. Stefan Lessmann, Hamburg

Literaturempfehlungen:

Bensberg, F./Schultz, M.B.: Data Mining. In: WISU, 30. Jg. (2001), S. 474 ff.
Berson, A./Smith, S./Thearling, K.: Building Data Mining Applications for CRM. New York 1999.
Fink,A./Schneidereit, G./Voß, S.: Grundlagen der Wirtschaftsinformatik. Heidelberg 2001.
Hippner, H./Wilde, K.: CRM — Ein Überblick. In: Helmke, S./Uebel, M./Dangelmaier, W. (Hrsg.): Effektives Customer Relationship Management. Wiesbaden 2002, S. 3 - 38.
Hettich, S./Hippner, H./Wilde, K.D.: Customer Relationship Management (CRM). In WISU, 29. Jg. (1999), S. 1346 ff.
Raab, G./Lorbacher, N.: Customer Relationship Management. Heidelberg 2002.
Voß, S./Gutenschwager, K.: Informationsmanagement. Berlin 2001.

BASISWISSEN VWL

Adverse Selektion

Der erste Hauptsatz der Wohlfahrtstheorie besagt, dass Wettbewerbsmärkte zu einer pareto-optimalen Güterallokation führen. Der Marktmechanismus findet — von Adam Smiths unsichtbarer Hand gelenkt — diejenigen Preise, bei denen Güterangebot und Güternachfrage übereinstimmen. Das Marktgleichgewicht ist effizient: Kein Marktteilnehmer kann besser gestellt werden, ohne die Situtation eines anderen zu verschlechtern.

Die Gültigkeit des ersten Hauptsatzes ist allerdings an eine Reihe von Voraussetzungen geknüpft. Insbesondere dürfen keine **Informationsdefizite** vorliegen. Sie treten vor allem dann auf, wenn Information asymmetrisch verteilt ist, wenn also eine Marktseite besser über die Qualität eines Guts informiert ist als die andere. Es kann dann zu **adverser Selektion** kommen. Unter adverser Selektion versteht man eine Negativauslese: Schlechte Qualitäten verdrängen gute Qualitäten vom Markt, im Extrem bricht der Markt sogar vollständig zusammen.

Ein Beispiel

George Akerlof (1970) hat das Problem adverser Selektion am Beispiel des Gebrauchtwagenmarktes erstmalig beschrieben. 2001 erhielt er dafür den Nobelpreis für Wirtschaftswissenschaften. Die Idee ist recht einfach: Man stelle sich einen Gebrauchtwagenmarkt vor, auf dem nur zwei Qualitäten gehandelt werden — Autos „guter“ und Autos „schlechter“ Qualität. Nur die Verkäufer kennen die Qualität ihrer Autos, die Käufer können den Gebrauchtwagen die Qualität hingegen nicht ansehen. Diese Information ist also asymmetrisch verteilt, die Verkäufer sind besser informiert als die Käufer.

Nehmen wir weiter an, dass gute Autos 15.000 Euro und schlechte 5.000 Euro wert sind. Wären Käufer und Verkäufer gleichmaßen gut informiert, würden die guten Autos für 15.000 Euro und die schlechten für 5.000 Euro gehandelt werden. Durch den Informationsvorteil der Verkäufer stellt sich die Situation jedoch anders dar: Wenn es aus der Sicht eines Käufers gleichermaßen wahrscheinlich ist, ein gutes oder ein schlechtes Auto zu erwischen, ist er grundsätzlich bereit, 10.000 Euro für einen Gebrauchtwagen zu zahlen, denn im Mittel bekommt er ja ein Auto, das 10.000 Euro wert ist (0,5·15.000 + 0,5·5.000 = 10.000). Zu einem Preis von 10.000 Euro werden jedoch nur schlechte Autos angeboten. Denn würde der Eigentümer eines guten Autos dieses für 10.000 Euro verkaufen, hätte er einen Verlust von 5.000 Euro.

Gehen die Käufer davon aus, dass nur schlechte Autos angeboten werden, sind sie nicht mehr bereit, 10.000 Euro für einen Wagen zu zahlen. Damit stellt sich ein Gleichgewichtspreis von 5.000 Euro ein, bei dem ausschließlich Gebrauchtwagen mit schlechter Qualität gehandelt werden.

Das Modell

Akerlofs Argument soll nun etwas genauer unter die Lupe genommen werden. Dabei hilft uns ein einfaches Modell. Dem interessierten Leser sei darüber hinaus die Lektüre von Hillier (1997), Molho (1997), Kreps (1994, Kap. 17) und Varian (1996, Kap. 35) empfohlen.

In unserem Modell gibt es zwei (risikoneutrale) Haushalte. Einer der beiden Haushalte — der Verkäufer — besitzt ein Auto, das er abgeben will, sofern er einen hinreichend hohen Preis dafür erzielen kann. Der andere Haushalt — der Käufer — ist grundsätzlich bereit, das Auto zu erwerben, allerdings muss der Preis akzeptabel sein. Der Nutzen des Käufers lautet

$$(1)\quad U^K = C^K + 3/2 \cdot Q \cdot n,$$

wobei Q für die Qualität des Autos steht und n eine Entscheidungsvariable ist, die den Wert 1 annimmt, wenn man das Auto besitzt, und den Wert 0 hat, wenn man das Auto nicht besitzt. C^K repräsentiert die noch verbleibenden Konsummöglichkeiten des Käufers. Mit anderen Worten: Mit dem Auto (n = 1) beträgt der Nutzen des Käufers $U^K = C^K + 3/2 \cdot Q$, ohne das Auto (n = 0) ist sein Nutzen $U^K = C^K$ hoch. Wie üblich muss der Haushalt eine Budgetbeschränkung einhalten. Sie lautet in unserem Beispiel: $Y^K = C^K + P \cdot n$. Y^K ist dabei das Einkommen des Käufers und P bezeichnet den Preis des Autos. Wird das Auto nicht gekauft (n = 0), kann das gesamte Einkommen ausgegeben werden, d.h. $C^K = Y^K$. Beim Erwerb des Gebrauchtwagens (n = 1) reduzieren sich die verbleibenden Konsummöglichkeiten dagegen auf $C^K = Y^K - P$. Der Käufer maximiert seinen Nutzen gemäß Gleichung (1) unter Einhaltung seiner Budgetrestriktion. Wenn wir die Budgetgleichung nach C^K auflösen und anschließend in die Nutzenfunktion einsetzen, erhalten wir

$$(2)\quad U^K = Y^K + (3/2 \cdot Q - P) \cdot n.$$

Klar: Falls $3/2 \cdot Q \geq P$ ist, wird das Auto gekauft. Der Käufer wählt n = 1. Andernfalls (bei $3/2 \cdot Q < P$) wird es

A Framework for Customer-Centric Data Mining with Support Vector Machines

Stefan Lessmann (corresponding author)[1] and Stefan Voß[2]

***Abstract*—**Supervised classification is an important part of data mining for customer relationship management. The paper proposes a hierarchical reference model for support vector machine based classification within this discipline. The approach balances the conflicting goals of transparent yet accurate models and compares favourably to reference classifiers in a large scale empirical evaluation in real-world customer relationship management applications. Recent advances in support vector machine research are incorporated to approach feature, instance and model selection in a unified framework.

Keywords: Marketing, Data Mining, Customer Relationship Management, Support Vector Machines

1. INTRODUCTION

Data mining is an essential part of customer relationship management (CRM) to analyse large data stores and gain insight into customer behaviour, needs and preferences. Such knowledge facilitates the design of customer-centric business processes as well as personalized marketing and service activities, which, in turn, help to leverage customer loyalty and maintain competitiveness in globalized and saturated consumer markets.

This paper is concerned with applications of predictive data mining in CRM, such as response modelling for direct marketing (Baesens et al., 2002; Kim et al., 2005; Viaene et al., 2001), churn analysis to prevent customer defection by proactive marketing (Buckinx and Van den Poel, 2005; Buckinx et al., 2007; Hung et al., 2006; Van den Poel and Lariviere, 2004), evaluating credit risk

[1] S. Lessmann, Institute of Information Systems, University of Hamburg, Von-Melle-Park 5, D-20146 Hamburg, Germany (telephone:+49.40.42838.4706, fax: +49.40.42838.5535, email: lessmann@econ.uni-hamburg.de).

[2] S. Voß, Institute of Information Systems, University of Hamburg, Von-Melle-Park 5, D-20146 Hamburg, Germany (telephone:+49.40.42838.3064, fax: +49.40.42838.5535, email: stefan.voss@uni-hamburg.de).

in consumer lending (Baesens et al., 2003a; Mues et al., 2004; Thomas et al., 2005), as well as identifying fraudulent business transactions (Fawcett and Provost, 1997; Viaene et al., 2002). Despite task-specific particularities, supervised classification is the predominant modelling approach to support decision-making in these fields.

Support vector machine (SVM) classifiers have received considerable attention in the machine learning literature and are appreciated because of their strong theoretical foundations and appealing predictive performance. Successful applications in CRM-related fields have been reported in, e.g., (Baesens et al., 2003b; Cheung et al., 2003; Crone et al., 2006; Cui and Curry, 2005; Shin and Cho, 2006). However, the awareness of SVMs in corporate practice is yet limited. According to Coussement and Van den Poel (2008), this may be explained with the lack of a holistic metamodel for adapting SVMs to specific tasks.

The objective of this paper is to overcome this obstacle and to develop a reference model that offers general guidance on applying SVMs effectively within a CRM-context. Considering the aforementioned applications, predictive accuracy and comprehensibility as well as scalability are (with varying degrees) important requirements candidate classifiers have to fulfil. Accuracy is crucial since even small variations can induce significant financial consequences (Baesens et al., 2003b; Baesens et al., 2002; Buckinx and Van den Poel, 2005), whereas comprehensibility of the classification decision, i.e. the way information is processed to generate a class prediction and which factors are most influential, is a prerequisite to satisfy the overall data mining objective of distilling knowledge from data and, eventually, facilitate enhancements of the concerned business processes. Furthermore, data mining applications require scalable algorithms capable of handling large (and high-dimensional) data streams.

The proposed reference model strives to satisfy these requirements. It consists of two steps and incorporates procedures for feature, instance and model selection to embrace the whole forecasting process. The first step discloses the relevance and effect of input variables, i.e. customer characteristics, to satisfy transparency constraints and provides a comparable degree of comprehensibility as established techniques like logistic regression (LogReg) or decision trees. The final classifier is constructed in step two and may involve nonlinear modelling to optimize predictive accuracy. This step exploits results of the preceding stage to reduce the computational burden associated with constructing a SVM classifier.

The reference model incorporates recent advancements in SVM-oriented research (i.e., Guyon et al., 2002; Keerthi and DeCoste, 2005; Keerthi and Lin, 2003), which have to the best of our knowledge not been considered in management applications before. Each technique has been developed independently to enhance an individual modelling step, e.g., the task of feature selection, model selection or classifier training. We identify synergies between these modifications/extensions of the original SVM and integrate them into a holistic framework that offers efficient and effective decision support. Exhaustive empirical evaluations are conducted to assess individual components of the framework and contrast the overall model against reference classifiers. The experimental design strives to achieve a maximal degree of representativeness for real-world classification problems in CRM. To that end, the study includes several large real-world datasets that represent challenging corporate data mining problems.

The paper is organized as follows: The basics of SVM theory are reviewed in Section 2 before the reference model is designed in Section 3. Section 4 describes the design and results of the empirical evaluation. Conclusions are drawn in Section 5.

2. SUPPORT VECTOR MACHINES FOR CLASSIFICATION

A SVM is a supervised learning algorithm that implements the principles of statistical learning theory (Vapnik, 1995) and can solve linear as well as nonlinear binary classification problems. Let S be a dataset with M observations, $S = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{M}$, where $\boldsymbol{x}_i \in R^N$ denotes an input vector and $y_i \in \{-1, +1\}$ its corresponding binary class label. The goal of classification is to infer a predictive model, i.e. a classifier, $y(\boldsymbol{x})$, from S, which accurately predicts the class membership of novel examples. Within a CRM-context, $\boldsymbol{x}$ is usually representing a customer, characterized by attributes x_t with t=1,…N, whereas y encodes some behavioural trait, e.g., whether or not the customer has defaulted on a dept.

The principle of SVM classification is to separate examples of opposite categories by means of a maximal margin hyperplane (Cristianini and Shawe-Taylor, 2000). That is, the algorithm maximizes the distance between examples of opposite classes which are closest to the separating hyperplane; see Fig. 1. It has been shown that maximizing the margin minimizes a bound of the generalization error, i.e. improves the model's ability to accurately classify future examples (Vapnik, 1995).

The constraint that examples of opposite classes have to reside on different sides of a linear hyperplane can be formulated as:

$$y_i((\boldsymbol{w}\cdot\boldsymbol{x}_i)+b)\geq 0,\ \ i=1,...,M\ , \tag{1}$$

where $\boldsymbol{w}$ denotes the plane's normal and b the intercept.

[Fig. 1 about here]

To maximize the margin of separation, the following quadratic program has to be solved:

$$\begin{aligned}&\min_{w,b,\xi}\frac{1}{2}\|\boldsymbol{w}\|_2+C\sum_{i=1}^{M}\xi_i\\&s.t.:\ y_i((\boldsymbol{w}\cdot\boldsymbol{x}_i)+b)+\xi_i\geq 1,\ \ i=1,...,M.\end{aligned} \tag{2}$$

The slack variables ξ_i account for misclassifications when the problem is not linearly separable, and C is a regularization parameter (also called hyperparameter) to control the trade-off between the conflicting goals of maximizing the margin and classifying the training set without error.

Examples which satisfy the constraint with equality are called support vectors (SVs). They define the orientation of the separating hyperplane and suffice to completely describe the dataset. That is, the solution to (2), i.e. the classifier, would not change if all other examples were discarded from S (Vapnik, 1995). To see this, consider the SVM dual program shown in (3). The Lagrangian multipliers α_i are zero for all non-SVs and the final classifier (4) is constructed by means of a SV-expansion.

$$\begin{aligned}&\max_{\alpha}=\sum_{i=1}^{M}\alpha_i-\frac{1}{2}\sum_{i,j=1}^{M}\alpha_i\alpha_j y_i y_j(\boldsymbol{x}_i\cdot\boldsymbol{x}_j)\\&s.t.\quad\sum_{i=1}^{M}\alpha_i y_i=0\ \ ;\ \ 0\leq\alpha_i\leq C\quad\forall i.\end{aligned} \tag{3}$$

$$y(\boldsymbol{x})=sgn\left(\left(\sum_{i\in SV}\alpha_i y_i(\boldsymbol{x}_i\cdot\boldsymbol{x})\right)+b\right). \tag{4}$$

The dual contains the input data only in form of inner products, which enables an extension of SVMs to nonlinear classification. This is accomplished by mapping the input vectors into a high-dimensional feature space via an a priori chosen mapping function Φ. Constructing a separating

hyperplane in this feature space leads to a nonlinear decision boundary in the input space. As only inner products within the transformed space are required, the mapping can be computed implicitly by means of a so called kernel function *K*:

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \Phi(\boldsymbol{x}_i) \cdot \Phi(\boldsymbol{x}_j). \quad (5)$$

K may be regarded as a proximity function that measures the distance between two input vectors in the nonlinearly transformed feature space. Note that algorithms for solving (3) may remain unchanged if the inner products are replaced with a respective kernel. The Gaussian radial basis function (RBF) (6) is a popular choice and most widely used in SVM applications.

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\gamma \left\| \boldsymbol{x}_i - \boldsymbol{x}_j \right\|^2\right). \quad (6)$$

The smoothing parameter γ determines the sensitivity of the distance measurement, i.e. the width of the Gaussian function.

This paper considers only the RBF kernel function because it possesses some desirable properties, i.e., being at least as good as other kernels or including other kernels as special cases (Keerthi and Lin, 2003; Lin and Lin, 2003), and avoids numerical difficulties associated with very large numbers. That is, values of the RBF function range between zero and one whereas those of a, e.g., polynomial kernel function range between zero and infinity (Coussement and Van den Poel, 2008). In order to apply a SVM with RBF kernel, the hyperparameters *C* and γ have to be determined to adapt the classifier to the given task, which is referred to as model selection.

3. A TWO-STAGE REFERENCE MODEL FOR SUPPORT VECTOR MACHINE-BASED CLASSIFICATION

The proposed reference model for CRM-related classification tasks consists of two major stages: First, a ranking of all attributes according to their relevance is produced by means of the recursive feature elimination (RFE) algorithm of Guyon et al. (2002). This step is implemented with a modified, linear SVM formulation that facilitates the application of an extremely fast Newton method developed by Keerthi and DeCoste (2005). The RFE-based feature ranking sheds light on the mechanisms underlying the linear first stage classifier and enables discarding less informative attributes to decrease the size of the data. The second stage aims at improving predictive accuracy and relies upon RBF-SVM to account for nonlinear interactions among attributes. To integrate the two stages and improve computational efficiency, we suggest initializing RBF-SVM model

selection with the optimal hyperparameter of the linear first stage classifier utilizing the line-search heuristic of Keerthi and Lin (2003). In addition, all non-SVs are discarded to further reduce the size of the second stage training dataset. Subsequently, these steps are explained in detail to motivate each design decision and provide a discussion of possible alternatives.

3.1. Feature selection with support vector machines

Feature selection aims at identifying and discarding attributes which are of minor importance, or, eventually detrimental, for the predictive model. Motivations for feature selection include: 1) decreasing the risk of over-fitting, 2) reducing the time for training and applying a classifier as well as 3) the costs for gathering the respective data and 4) improving comprehensibility of the classification model (Guyon and Elisseeff, 2003).

Several procedures have been proposed for SVM-based feature selection. For example, minimizing the L1-norm instead of the L2-norm of $\boldsymbol{w}$ in (2) yields a linear program that implicitly discards features by forcing several components of $\boldsymbol{w}$ to zero (Bradley et al., 1998; Bradley and Mangasarian, 1998). Alternatively, an individual scaling factor for each attribute may be incorporated into the RBF kernel function to enable weighting each feature individually (Chapelle et al., 2002; Keerthi et al., 2007).

The approach taken here is based on an attribute's contribution to the margin of separation, i.e. its weight vector coefficient $\boldsymbol{w}_t$ in the linear case. Intuitively, attributes with low coefficient have less influence on the class decision (4), indicating that they are not important for a classification purpose and can thus be discarded (Brank et al., 2002; Sindhwani et al., 2001). Following this reasoning, Guyon et al. (2002) propose RFE as an iterative backward-elimination procedure for SVM-based feature ranking: A SVM classifier is trained using the full feature set and all attributes are assessed by means of their margin contribution. The attribute with the lowest value is removed and the classifier is trained again with the modified feature set. The procedure continues until all features are removed, therewith providing a ranking of attributes by means of their coefficients in $\boldsymbol{w}$ and time of removal.

Although RFE is applicable with nonlinear SVMs, a simple linear model is preferable at this stage for the following reasons: Linear SVMs require less training time as the optimization can be carried out with fast Newton algorithms instead of quadratic programming (see below). Further efficiency gains originate from a reduction of model selection activities as only one hyperparameter, *C*, has to be tuned. Consequently, the reference model implements RFE with a linear SVM.

The resulting ranking enables appraising the relevance of individual attributes but does not dictate the optimal number of attributes to be discarded. Consequently, an auxiliary performance criterion is needed to utilize RFE for feature selection (Guyon et al., 2002). The empirical results presented in Section 4.2 motivate usage of a simple selection heuristic: Attributes are removed according to their rank until the performance of the classifier on validation data deteriorates.

3.2. Implementing recursive feature elimination with a modified finite Newton method

It is important to note that Guyon et al. (2002) proposed RFE while analysing a cancer classification problem that was linearly separable down to just a few features and consequently insensitive towards the regularization parameter; see (2). On the other hand, the performance of RFE to select meaningful features depends substantially upon an appropriate choice of C when class distributions overlap (Huang et al., 2006). However, no clear standard for organizing feature selection in conjunction with model selection has emerged (Rakotomamonjy, 2003). The approach taken here considers tuning C as an initial step, prior to utilizing the linear SVM for RFE. Adopting the standard practice in SVM model selection (e.g., Hsu et al., 2003), a set of candidate choices is assessed by means of cross-validation using the full feature set and the setting with highest predictive performance is selected.

RFE requires training multiple classifiers on feature sets of decreasing size. Such iterative schemes are costly, especially for large datasets. However, we can alleviate difficulties by using a novel training algorithm for linear SVMs, recently proposed by Keerthi and DeCoste (2005). It reformulates the SVM program (2) by introducing a least-square loss-function, i.e. minimizing the L2-norm of the slack, and measuring the margin with respect to normal $\boldsymbol{w}$ and intercept b. We refer to (7) as L2-SVM.

$$\begin{aligned} &\min_{w,b} \frac{1}{2C}\left\|\boldsymbol{w},b\right\|_2^2 + \frac{1}{2}\sum_{i\in I}\left(\left((\boldsymbol{w}\cdot\boldsymbol{x}_i)+b\right)-y_i\right)^2 \\ &I \coloneqq \left\{i : y_i((\boldsymbol{w}\cdot\boldsymbol{x}_i)+b) < 1\right\}. \end{aligned} \tag{7}$$

The inclusion of $b^2/2$ in the objective has little effect on the classifier and induces strong convexity, so that (7) has a unique minimizer (Mangasarian and Musicant, 2001).

An advantage of L2-SVM stems from its similarity to regularized least-squares problems (e.g., Hastie et al., 2002), which enables the application of fast algorithms specifically designed for this type of programs. In particular, the method of Keerthi and DeCoste (2005) begins with an initial

solution $\left(\boldsymbol{w}^0, b^0\right)$ and solves a regularized least-squares problem consisting only of data points misclassified by $\left(\boldsymbol{w}^0, b^0\right)$. Employing a specialized conjugate gradient procedure (Frommer and Maaß, 1999), a Newton point $\left(\bar{\boldsymbol{w}}, \bar{b}\right)$ is obtained and used to decrease the full objective (7) via the updating rule (8), whereby δ denotes the step-size which is determined by means of an exact line-search on the ray from the current solution to the Newton point.

$$\left(\boldsymbol{w}^1, b^1\right) = \left(\boldsymbol{w}^0, b^0\right) + \delta\left(\left(\bar{\boldsymbol{w}}, \bar{b}\right) - \left(\boldsymbol{w}^0, b^0\right)\right). \tag{8}$$

The reader is referred to Keerthi and DeCoste (2005) for algorithmic details and convergence proofs.

3.3. Optimizing accuracy in stage two

In order to maintain transparency as well as computational efficiency, the first stage relies upon a linear L2-SVM classifier. The objective of stage two is to optimize predictive accuracy and scrutinize if classification performance can be improved by means of nonlinear RBF-SVMs. However, building nonlinear SVMs in large-scale CRM-settings is a time-consuming endeavour as training algorithms are less efficient and a careful selection of the hyperparameters *C* and γ requires intensive model selection (see, e.g., Baesens et al., 2003b; Hsu et al., 2003; Van Gestel et al., 2004). The framework accounts for these challenges by reusing results of stage one to reduce the amount of data and simplify model selection.

On the one hand, feature selection decreases the data by discarding less informative attributes. Furthermore, we restrict the nonlinear modelling to the SVs of stage one, i.e. the set *I* after solving (7). This can be seen as an 'intelligent sampling' and is inspired by the definition of SVs as the set of points which suffice to completely describe the data (Coussement and Van den Poel, 2008; Vapnik, 1995). One may object that this understanding requires methodological consistency whereas we suggest reusing the SVs of L2-SVM for training a nonlinear RBF-SVM. However, Keerthi and DeCoste (2005) point out that L2-SVM usually yields a larger number of SVs than a standard SVM. Hence, the risk that the proposed SV-sampling discards 'relevant' data points and inadequately affects the nonlinear stage seems negligible. In addition, one may understand this sampling as an attempt to reweight the training data towards difficult examples in the sense of boosting (Freund and Schapire, 1996).

Regarding SVM model selection, two philosophies can be distinguished within the literature. An empirical approach appraises different parameter settings by means of cross-validation, bootstrapping or a similar scheme to estimate generalization error, whereas theoretical model selection procedures minimize bounds on the generalization error with respect to the hyperparameters. The intuition behind the latter is that the computation of these bounds may be cheaper than repetitive training of SVMs with varying parameter values (see, e.g., Chung et al., 2003; Lee et al., 2004; Vapnik and Chapelle, 2000). However, previous empirical results suggest that cross-validation approximates the generalization more accurately than theoretical criteria (Duan et al., 2003). Furthermore, empirical model selection is independent of the employed performance metric, whereas most theoretical work is restricted to classification error. Consequently, we adopt an empirical model selection procedure for our framework.

Grid-search is the most popular procedure and involves predefining a range of candidate settings for each hyperparameter and empirically evaluating all possible combinations. For example, Hsu et al. (2003) recommend a grid of $\log_2(C)$=[-5, -4, …, 15] and $\log_2(\gamma)$=[-15, -14, …, 3] for RBF-SVM. Using a log-scale is standard practice to explore a large region of C, γ values. To reduce the number of evaluations, Van Gestel et al. (2004) start with a coarse grid which is subsequently refined in promising regions of the parameter space. A related yet less known approach to search the parameter space more efficiently is the pattern search algorithm (Dennis and Torczon, 1994), which has been used successfully in conjunctions with support vector regression (Momma and Bennett, 2002). Finally, Keerthi and Lin (2003) investigate the hyperparameter space of RBF-SVMs to determine good and bad (over-/under-fitting) regions and derive a simple model selection heuristic. Based on a theoretical analysis of the asymptotic behaviour as $C, \gamma \to 0, \infty$, they identify patterns within this space and conclude that promising parameter settings are arranged along the line (9), where $\tilde{C}$ denotes the optimal setting of a linear SVM.

$$\log_2(\gamma) = \log_2(C) - \log_2(\tilde{C}). \qquad (9)$$

The idea is that RBF-SVMs and linear SVMs behave similarly when $\gamma \to 0$. Assuming that a linear classifier already gives a reasonably good performance, $\tilde{C}$ will reside in the lower region of the promising C, γ area. Hence, the heuristic starts with a linear classifier and strives to improve it by introducing nonlinearity through the RBE kernel searching along a line with unit-slope that cuts through the promising region of the parameter space (Keerthi and Lin, 2003).

This heuristic integrates fluently with the proposed framework as the optimal solution of a linear SVM model, L2-SVM, has already been obtained in stage one. Consequently, the reference model employs (9) for RBF-SVM model selection and assigns the optimal hyperparameter value of L2-SVM to $\tilde{C}$.

It should be noted that the reference model naturally produces two estimates of generalization error when following the proposed procedures for model building: One for the linear L2-SVM in the first stage and one for RBF-SVM is stage two. If these estimates suggest that L2-SVM is better suited for the respective datasets, the reference model can smoothly switch to the linear classifier and use it for classifying hold-out data.

4. Empirical evaluation of the SVM-based reference model

In designing the reference model, several design decisions have been made. Consequently, an empirical validation is required to confirm the effectiveness of individual framework components and assess the potential of the overall model.

Comparing the predictive performance of classification models involves selecting an appropriate accuracy indicator and deciding upon a scheme for estimating the model's performance on future data. The studies objectives are: 1) Assessing the effectiveness of design decisions made in developing the reference model and contrasting the overall model against alternative classifiers. To that end, the experimental setup is as follows: 2/3 of a dataset are randomly sampled for model building and the remaining 1/3 is used as test set facilitating out-of-sample comparisons. This procedure is repeated ten times to decrease the variance of the resulting performance estimates and avoid bias because of a 'lucky sample'. Model selection for benchmark classifiers (see below) is conducted by means of ten-fold cross-validation on training data.

In addition, an extended setup is used for the reference model to facilitate a robust assessment of individual components without affecting the hold-out comparisons against other classifiers. Therefore, each iteration's training set is once more randomly partitioned into a learning and validation set, using a ratio of 2/3 : 1/3. We denote this setting as a nested two-level split-sample setup. Yielding ten randomly sampled test and validation sets, it facilitates robust benchmarks against competing classifiers (test-set) and assessments of internal framework components (validation set). However, each of the ten iterations may require additional performance estimates, e.g. for guiding the search for predictive hyperparameter settings or select informative features.

Within our experiments, such estimates are produced by inner cross-validation (ten-fold) on learning data.

The area under a receiver operating characteristic curve (AUC) is considered as accuracy indicator (Bradley, 1997). AUC measures the probability that a classifier ranks a randomly chosen positive instance higher than a randomly chosen negative one, which is equivalent to the Wilcoxon test of ranks (Fawcett, 2006). AUC is a general measure of predictiveness and decouples classifier assessment from class/cost distributions which makes it an ideal tool for benchmarking experiments. Another motivation for using AUC in this study stems from its close relationship to lift and gain chart analysis which are standard tools in direct marketing (see, e.g., Huang and Ling, 2005; Ling and Li, 1998).

4.1. Dataset description

The study incorporates three standard datasets from the UCI machine learning repository (Newman et al., 1998). Australian credit (AC) and German credit (GC) constitute credit scoring tasks that aim at categorizing customers into good and bad risks, whereas the Adult database comprises census data to classify whether the annual income of individuals exceed 50,000$ per year.

In addition, six datasets from the annual Data Mining Cup (DMC) competition are utilized. The DMC datasets represent challenging real-world business problems in CRM-related applications and can be obtained from the contest organizer prudsys AG.[3] The datasets DMC 2000 and 2001 stem from the mail-order business and involve response modelling, i.e. scoring households according to their likelihood of responding to direct-mail. DMC 2002 considers a case of customer attrition analysis in energy markets, whereas DMC 2004 requires an analysis of customers' tendency to return ordered items in a direct marketing setting. DMC 2005 is the only non-real-world dataset within this collection and simulates a case of fraud-detection in online-selling. It involves estimating a customer's risk of defaulting to differentiate the offered payment type. Finally, internet auctions are analysed in DMC 2006. Based upon auction configuration (duration, starting price, one-click-buy price etc.) as well as textual data form the offer's title, a model should classify if the settlement will be above the average price of the respective product category. Note that DMC 2003 has been excluded from the comparison because it represents a text

[3] http://www.data-mining-cup.com/

classification problem (junk-email filtering) which is not representative for the field considered here.

The DMC datasets are available in raw format, including non-numerical data and missing values. It has been shown that the particular way of pre-processing the data influences predictive performance (Crone et al., 2006). However, this paper focuses on prediction and consequently employs standard techniques like dummy-encoding of categorical attributes and standardization of numerical values (Crone et al., 2006), whereas the evaluation of more sophisticated methods is left to further research. Summary statistics for each dataset are given in Table 1.

[Table 1 about here]

4.2. Feature ranking and selection by means of recursive feature elimination

A key objective of the first modelling stage is to appraise the relevance of individual attributes to provide some insight into the classification model and discard less informative features. In the following, the Adult dataset is used to illustrate the respective procedures.

RFE produces one feature set per iteration which predictive power can be traced to produce a vector of performance estimates. Fig. 2 displays the performance distribution over individual RFE-iterations. A boxplot is used to depict the variance over different validation sets and the solid line represents the average AUC performance.

[Fig. 2 about here]

Fig. 2 reveals that this particular dataset contains several features with little predictive value. The classification performance in terms of AUC remains roughly at the same level when removing the five lowest ranked attributes and using only the last three features for model building still achieves a respectable AUC of 0.88. On the one hand, this may be seen as evidence for the importance of (rigorous) feature selection. However, within a CRM context, a common view is that minor performance improvements – or reductions – can induce significant financial consequences (e.g., Baesens et al., 2003b; Baesens et al., 2002; Coussement and Van den Poel, 2008). Consequently, a conservative feature selection strategy seems preferable for this domain. Therefore, the

reference model utilizes AUC to guide feature selection, i.e. the attribute set yielding the maximal performance is considered in subsequent modelling steps.

To confirm the appropriateness of this heuristic, the following benchmark is considered: The data is augmented with random gauge attributes and all (true) attributes that receive a lower rank than the highest ranked random attribute are dismissed (Bi et al., 2003; Stoppiglia et al., 2003). Table 2 depicts the results of these two competitors by means of AUC.

[Table 2 about here]

Augmenting the data with gauge attributes is conceptually elegant and avoids the need to trace predictive performance. However, Table 2 reveals that it is almost consistently inferior to selecting the best performing feature set for the considered datasets, i.e. achieves the same or smaller AUC at a larger number of attributes. The results on DMC 2004 are completely misleading and cause significant performance deterioration.

The results of Table 2 suggest that feature selection is not required to improve predictive accuracy. On the contrary, the benchmark setting with all attributes included constitutes an upper bound across all datasets. This is consistent with the opinion that SVMs are robust towards large feature sets (Vapnik, 1995). However, another merit of feature selection is sought in reducing the dataset size to speed-up nonlinear modelling in the subsequent stage. Therefore, the results are taken as further confirmation for the decision to rely upon predictive performance for selecting features.

The major objective of applying RFE in stage one is, however, to improve the comprehensibility of the classification model. This is achieved by exploiting the RFE-based ranking, which sheds light on the relevance of individual attributes and their influence on the classification decision. A distribution of each attribute's coefficient in $\boldsymbol{w}$ is obtained over ten random validation sets, which can be illustrated by means of a boxplot. This is shown for the case of Adult in Fig. 3.[4]

[Fig. 3 about here]

[4] Note that the ordering of attributes differs from the original Adult dataset. Here, the order is: age, fnlwgt, education-num, capital-gain, capital-loss, hours-per-week, sex, workclass, education, marital-status, occupation, relationship, race, native-country; see ftp://ftp.ics.uci.edu/pub/machine-learning-databases/adult/adult.names.

The examination of individual attributes' influence is crucial to satisfy the overall data mining objective of discovering patterns in data which, eventually, enable improving customer-centric processes. For example, some attributes' impact on the classifier is consistent over different iterations whereas others exhibit a high degree of variation. This could indicate the presence of subpopulations within the data and possibly opportunities for cross-selling activities. In addition, Fig. 3 allows appraising the classification model's consistence with human experience. For example, considering the task represented by Adult, i.e. predicting if a person's gross income exceeds 50,000$, it is not surprising that feature four (the individual's capital gain) exhibits a strong positive correlation with the target variable. In other words, Fig. 3 confirms that the model's view of this attribute is consistent with human domain knowledge. Considering the general scepticism towards (complex) data mining models in corporate practice the possibility of conducting such simple checks is of vital importance to improve their acceptance.

Clearly, the perceivability of visualizations like Fig. 3 benefits from a smaller number of attributes. This may be taken as further support for the decision to incorporate feature selection – rather than only feature ranking – into the framework. Note that the view that models using a smaller number of features are easier to interpret has also been advanced in previous studies (e.g., Kim et al., 2005)

It should be reemphasized that the reference model employs a linear SVM for feature ranking in stage one. One may object that the effect of certain features could be different in stage two, when using a nonlinear RBF-SVM. Clearly, linear classifiers are unable to detect nonlinear interactions, which is precisely the motivation for using RBF-SVM in the second stage. However, considering the asymptotic behaviour of RBF-SVM (Keerthi and Lin, 2003) and the employed model selection procedure, it is reasonable to assume that the importance of individual attributes varies only moderately when moving to nonlinear classification. In addition, we argue that disclosing linear relationships between attributes and the target variable suffices to satisfy the general data mining objective of deriving knowledge from data. For example, knowing that a specific attribute, e.g. capital gain in the previous example, or a set of attributes are key drivers for the classification decision enables refining marketing activities and/or enhancing business processes. Consequently, the results of Fig. 3 remain informative even if a nonlinear RBF-SVM is used to

produce the final predictions and provide valuable insight into an otherwise opaque SVM-based classification model.

Alternatively, it would be possible to extract rules from the second stage RBF-SVM classifier (see, e.g., Barakat and Diederich, 2005; Barakat and Bradley, 2007; Martens et al., 2007); but at the expense of incorporating additional algorithms into the framework and higher computational costs. In particular, the model selection procedure considered here implicitly requires solving a linear SVM before applying RBF-SVM. Thus, results of a linear model are naturally available before moving to stage two.

4.3. Instance reduction by means of support vector sampling

Feature selection helps to reduce the computational burden associated with building a non-linear RBF-SVM classifier in stage two. In addition, the overall size of the data may be further reduced by restricting the second stage learning set to the SVs of stage one which, by definition, suffice to characterize the data. The effect of this mechanism is summarized in Table 3, which contrasts the results of applying the reference model with and without SV-sampling.

[Table 3 about here]

Table 3 confirms the observation of Keerthi and DeCoste (2005) that L2-SVM produces a larger number of support vectors. Consequently, only a moderate decrease of dataset size is achieved. On the other hand, a noteworthy advantage of this reduction procedure is that it does not deteriorate predictive performance: The influence of SV-sampling on AUC is negligible. Therefore, the proposed sampling may be seen as a ‘safe’ option for data reduction, discarding some examples while maintaining the predictive performance of the classification model.

In addition, Table 3 suggests that the class distribution among support vectors is less skewed than within the overall training data, i.e. the prior of positive instances is consistently higher in the modified training set. This is an interesting pattern since class imbalances generally impede predictive modelling (Japkowicz and Stephen, 2002) and are commonly encountered within a CRM-context because important customers, e.g. those who respond to direct-mail or exhibit a high risk to abandon their relationship with a company, naturally represent minorities. A possible explanation for the balancing effect of SV-sampling is that the distribution of majority class examples could exhibit a higher dispersion, which, in turn, would naturally produce a larger number

of instances far away from the decision boundary, i.e. non-SVs; see also Fig. 1. However, further research is needed to clarify the origin and scrutinize the persistence of this pattern.

4.4. Model selection by means of line-search

We suggest organizing model selection for RBF-SVM by means of the line-search heuristic of Keerthi and Lin (2003). This enables reusing the setting of the regularization parameter *C* of the previous stage and thereby further decreases the computational effort of building the classifier. Table 4 depicts the empirical results of this model selection procedure to illustrate its effectiveness in comparison to pattern-search (Momma and Bennett, 2002).

[Table 4 about here]

Considering the 399 iterations of a conventional search over the reference grid of Hsu et al. (2003), line-search and pattern-search achieve a significant gain in efficiency. Clearly, a standard grid-search would be infeasible for datasets of the size considered here. The number of line-search iterations depends on the range of *C* candidate values, whereas pattern-search explores the two-dimensional parameter space until finding a local optimum. Line-search generally requires less iterations than pattern-search and is consequently superior in terms of computing times. The observed differences are particularly large for the datasets Adult as well as DMC 2000 and 2005.

Noteworthy, this improvement does not sacrifice accuracy. Overall, both techniques give similar results in terms of AUC, with line-search being slightly better on GC, Adult, DMC 2001, 2002, 2005 and slightly inferior on the others. This is a respectable result and confirms the findings of Keerthi and Lin (2003) regarding the structure of the C, γ parameter space.

4.5. Predictive accuracy on hold-out data

The previous sections have confirmed the appropriateness of individual components of the proposed reference model. Subsequently, empirical comparisons are conducted to contrast its predictive performance against established benchmarks. LogReg, C4.5 (Quinlan, 1993) and CART (Breiman et al., 1984) are considered as reference classifiers because of their popularity within corporate data mining.

Model selection for decision trees involves deciding upon a pruning strategy and varying confidence levels of [0.1, 0.2, 0.25, 0.3] are evaluated as candidate settings, each time with and with-

out Laplacian smoothing (see Mingers, 1989; Provost and Domingos, 2003 for details). LogReg exhibits no auxiliary hyperparameters to be tuned by means of model selection. However, multicollinearities within the data prohibit a direct application of this method. Thus, model selection for LogReg corresponds to selecting a feature subset which is accomplished by means of backward elimination. Furthermore, the comparison also includes the original RBF-SVM which, in view of previous results from the literature (Baesens et al., 2003b; Van Gestel et al., 2004), represents a highly challenging benchmark. Pattern search (Momma and Bennett, 2002) with an initial setting of C=1 and $\gamma = 1/N$ is used for RBF-SVM model selection.

All experiments are conducted within the Matlab environment. The CART experiments use the Matlab Statistics toolbox and external packages are employed for the other algorithms (Chang and Lin, 2001; Kiefte, 1999; Sindhwani and Keerthi, 2007; Weston et al., 2006). Results are presented in Table 5. Note that this is the only evaluation which is based on hold-out testing data, whereas all previous results have been derived from validation data.

[Table 5 about here]

The proposed model compares favourably to the considered benchmarks, yielding the highest AUC on five out of nine datasets, whereby the first rank on the Adult dataset is shared with RBF-SVM and LogReg. In particular, promising results are obtained for the challenging DMC datasets which is appealing since these are deemed most representative for real-world data mining applications in CRM. As explained above, the design of the reference model enables estimating whether L2-SVM or RBF-SVM is better suited for a particular task, i.e. comparing in-sample cross-validation performance, and a respective classifier is selected automatically. The effectiveness of this feature is confirmed by the fact that the first rank on DMC2002, 2004 and 2005 is indeed obtained with the linear L2-SVM classifier. Consequently, the results provide strong evidence for the appropriateness of granting linear methods a major role within the reference model.

The key motivation for selecting the particular datasets of this study has been their representativeness for the field of corporate data mining (considering their size, AC and GC do not fully qualify as data mining tasks but have been chosen because of their popularity in the literature). Therefore, the competitive performance of linear classifiers is interesting. It may suggest that these datasets exhibit only moderate nonlinearities. On the other hand, considering the fact that

RBF-SVM includes linear SVMs as a special case (Keerthi and Lin, 2003), another explanation could be that the hyperparameter space has not been searched sufficiently during model selection. Clearly, line-search and pattern search are heuristics which might produce local optima. Overall, the hold-out results confirm the previous finding of line-search being as effective as pattern search: The reference model is competitive, sometimes superior to standard RBF-SVM. In assessing this result it is important to remember that the reference model is significantly cheaper in terms of computing times, due to feature/instance selection and the more efficient model selection procedure, whereas the additional cost of RFE is negligible thanks of the high efficiency of L2-SVM.[5] It falls behind standard SVM only on DMC 2000. In view of the results presented in Tables 2–4, it is likely that this result is caused by line-search determining less effective hyperparameter values.

However, model selection on large dataset is a major challenge. For example, one could assume that RBF-SVM should be able to outperform LogReg on AC and GC when conducting excessive model selection. Though, even an exponentially refined grid-search, which is computationally much more expensive than the procedures considered here, failed to outperform LogReg on these datasets in previous experiments (Baesens et al., 2003b; Van Gestel et al., 2004). Utilizing such techniques for datasets of the size considered here would be computationally infeasible. Therefore, the reference model would benefit from future research to develop yet more effective and efficient model selection procedures. For the time being, the integration of the respective merits of linear and nonlinear SVMs, which is at the core of the proposed reference model, has proven its potential to provide accurate predictions on challenging benchmarking problems. Classifiers that are purely linear or nonlinear might outperform the reference model on a particular task, e.g. AC and GC or DMC 2000, but in general, the proposed two stage classifier produces competitive results across all datasets and achieves the highest degree of consistency, i.e. highest average AUC and lowest average standard deviation over all tasks. Therefore, it may be concluded that it is a promising candidate for predictive data mining and has the potential to enhance modelling standards in corporate practice.

---

[5] The overall size of the numerical study required using multiple workstations with varying hard- and software configuration. Therefore, we refrain from reporting runtimes since these are not comparable across machines. As an indication, we refer the reader to Table 4 which depicts runtimes for the comparison of line-search versus pattern-search for model selection, i.e., the most expensive modelling step.

5. Conclusion

A reference model for SVM-based classification in CRM-contexts has been proposed to overcome the obstacle of a missing meta-theory for applying SVMs to particular decision problems. The model demonstrates how recent advancs in SVM-oriented research can be integrated into a well structured framework that offers holistic guidance for approaching typical challenges in corporate data mining with SVM-based technology. Methodological consistency has been a major design principle and is expected to improve the comprehensibility of the modelling paradigm, which, in turn, may avail dispersion in corporate practice. Furthermore, the empirical results indicate that the approach compares favourably to established benchmarks and can be applied 'off-the-shelf' to disclose relationships among attributes and gain insight into their relevance for the classification decision, discard less informative features and reduce the dataset size, select suitable hyperparameters and determine if a linear or nonlinear SVM is better suited for the given task.

The latter feature exemplifies that important modelling decision can be made in a purely data-driven manner, which is regarded as a particular merit of the proposed technique. It is planned to incorporate further extensions along this line in feature research; i.e. additional components for specific modelling challenges. For example, imbalanced class distributions are commonly encountered within a CRM context and algorithms like kernel boundary alignment (Wu and Chang, 2005) could help to further improve predictive accuracy under such circumstances. The availability of related independent components for specific problems, together with mechanisms for activating or deactivating them in a data-driven manner improves the usability of a forecasting procedure in general and may help to proceeds some steps into the direction of automated modelling.

References

Baesens B, Setiono R, Mues C, Vanthienen J. Using neural network rule extraction and decision tables for credit-risk evaluation. *Management Science* 2003a;49(3); 312-329.

Baesens B, Van Gestel T, Viaene S, Stepanova M, Suykens J, Vanthienen J. Benchmarking state-of-the-art classification algorithms for credit scoring. *Journal of the Operational Research Society* 2003b;54(6); 627-635.

Baesens B, Viaene S, Van den Poel D, Vanthienen J, Dedene G. Bayesian neural network learning for repeat purchase modelling in direct marketing. *European Journal of Operational Research* 2002;138(1); 191-211.

Barakat N, Diederich J. Eclectic rule-extraction from support vector machines. *International Journal of Computational Intelligence* 2005;2(1); 59-62.

Barakat NH, Bradley AP. Rule extraction from support vector machines: A sequential covering approach. *IEEE Transactions on Knowledge and Data Engineering* 2007;19(6); 729-741.

Bi J, Bennett KP, Embrechts M, Breneman C, Song M. Dimensionality reduction via sparse support vector machines. *Journal of Machine Learning Research* 2003;3; 1229-1243.

Bradley AP. The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition* 1997;30(7); 1145-1159.

Bradley P, Mangasarian O, Street W. Feature selection via mathematical programming. *INFORMS Journal on Computing* 1998;10(2); 209-217.

Bradley PS, Mangasarian OL 1998. Feature Selection via Concave Minimization and Support Vector Machines. In: Shavlik JW (Ed.), *Proc. of the 15th Intern. Conf. on Machine Learning.* Morgan Kaufmann: San Francisco. pp. 82-90.

Brank J, Grobelnik M, Milic-Frayling N, Mladenic D 2002. Feature Selection Using Support Vector Machines. In: *Proc. of the 3rd Intern. Conf. on Data Mining Methods and Databases for Engineering, Finance, and Other Fields*. Bologna, Italy.

Breiman L, Friedman JH, Olshen R, Stone C. *Classification and Regression Trees*. Wadsworth: Belmont; 1984.

Buckinx W, Van den Poel D. Customer base analysis: Partial defection of behaviourally loyal clients in a non-contractual FMCG retail setting. *European Journal of Operational Research* 2005;164(1); 252-268.

Buckinx W, Verstraeten G, Van den Poel D. Predicting customer loyalty using the internal transactional database. *Expert Systems with Applications* 2007;32(1); 125-134.

Chang C-C, Lin C-J, LIBSVM - A Library for Support Vector Machines, 2001 (www.csie.ntu.edu.tw/~cjlin/libsvm).

Chapelle O, Vapnik V, Bousquet O, Mukherjee S. Choosing multiple parameters for support vector machines. *Machine Learning* 2002;46(1-3); 131-159.

Cheung K-W, Kwok JT, Law MH, Tsui K-C. Mining customer product ratings for personalized marketing. *Decision Support Systems* 2003;35(2); 231-243.

Chung K-M, Kao W-C, Wang L-L, Lin C-J. Radius margin bounds for support vector machines with RBF kernel. *Neural Computation* 2003;15(11); 2643-2681

Coussement K, Van den Poel D. Churn prediction in subscription services: An application of support vector machines while comparing two parameter-selection techniques. *Expert Systems with Applications* 2008;34(1); 313-327.

Cristianini N, Shawe-Taylor J. *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods*. Cambridge University Press: Cambridge; 2000.

Crone SF, Lessmann S, Stahlbock R. The impact of preprocessing on data mining: An evaluation of classifier sensitivity in direct marketing. *European Journal of Operational Research* 2006;173(3); 781-800.

Cui D, Curry D. Predictions in marketing using the support vector machine. *Marketing Science* 2005;24(4); 595-615.

Dennis JE, Torczon V 1994. Derivative-Free Pattern Search Methods for Multidisciplinary Design Problems, *Proc. of the 5th AIAA/ USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization.* AIAA: Washington. pp. 922-932.

Duan K, Keerthi SS, Poo AN. Evaluation of simple performance measures for tuning SVM hyperparameters. *Neurocomputing* 2003;51; 41-59.

Fawcett T. An introduction to ROC analysis. *Pattern Recognition Letters* 2006;27(8); 861-874.

Fawcett T, Provost F. Adaptive fraud detection. *Data Mining and Knowledge Discovery* 1997;1(3); 291-316.

Freund Y, Schapire RE 1996. Experiments with a New Boosting Algorithm. In: Saitta L (Ed.), *Proc. of the 13th Intern. Conf. on Machine Learning.* Morgan Kaufmann: San Francisco. pp. 148-156.

Frommer A, Maaß P. Fast CG-based methods for Tikhonov-Phillips regularization. *SIAM Journal of Scientific Computing* 1999;20(5); 1831-1850.

Guyon I, Weston J, Barnhill S, Vapnik V. Gene selection for cancer classification using support vector machines. *Machine Learning* 2002;46(1-3); 389-422.

Guyon IM, Elisseeff A. An introduction to variable and feature selection. *Journal of Machine Learning Research* 2003;3; 1157-1182.

Hastie T, Tibshirani R, Friedman J. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* Springer: New York; 2002.

Hsu C-W, Chang C-C, Lin C-J 2003, A Practical Guide to Support Vector Classification, Working paper, *Department of Computer Science and Information Engineering, National Taiwan University*, 2003 (www.csie.ntu.edu.tw/~cjlin/guide/guide.pdf).

Huang J, Ling CX. Using AUC and accuracy in evaluating learning algorithms. *IEEE Transactions on Knowledge and Data Engineering* 2005;17(3); 299-310.

Huang T-M, Kecman V, Kopriva I. *Kernel based Algorithms for Mining Huge Data Sets: Supervised, Semi-Supervised, and Unsupervised Learning*. Springer: Berlin; 2006.

Hung S-Y, Yen DC, Wang H-Y. Applying data mining to telecom churn management. *Expert Systems with Applications* 2006;31(3); 515-524.

Japkowicz N, Stephen S. The class imbalance problem: A systematic study. *Intelligent Data Analysis* 2002;6(5); 429-450.

Keerthi SS, DeCoste D. A modified finite Newton method for fast solution of large scale linear SVMs. *Journal of Machine Learning Research* 2005;6; 341-361.

Keerthi SS, Lin C-J. Asymptotic behaviors of support vector machines with Gaussian kernel. *Neural Computation* 2003;15(7); 1667-1689.

Keerthi SS, Sindhwani V, Chapelle O 2007. An Efficient Method for Gradient-Based Adaptation of Hyperparameters in SVM Models. In: Schölkopf B, Platt JC, Hoffman T (Eds.), *Advances in Neural Information Processing Systems 19*. MIT Press: Cambridge; 2007. pp. 217-224.

Kiefte M, Discriminant Analysis Toolbox 1999 (http://www.mathworks.com/matlabcentral/fileexchange/).

Kim YS, Street WN, Russell GJ, Menczer F. Customer targeting: A neural network approach guided by genetic algorithms. *Management Science* 2005;51(2); 264-276.

Lee MMS, Keerthi SS, Ong CJ, DeCoste D. An efficient method for computing leave-one-out error in support vector machines with Gaussian kernels. *IEEE Transactions on Neural Networks* 2004;15(3); 750-757.

Lin H-T, Lin C-J 2003, A Study on Sigmoid Kernels for SVM and the Training of Non-PSD Kernels by SMO-Type Methods, Technical report *Department of Computer Science and Information Engineering, National Taiwan University.* , 2003 (http://www.csie.ntu.edu.tw/~cjlin/papers/tanh.pdf).

Ling CX, Li C 1998. Data Mining for Direct Marketing: Problems and Solutions. In: Agrawal R, Stolorz P (Eds.), *Proc. of the 4th Intern. Conf. on Knowledge Discovery and Data Mining.* AAAI Press: Menlo Park. pp. 73-79.

Mangasarian OL, Musicant DR. Lagrangian support vector machines. *Journal of Machine Learning Research* 2001;1; 161-177.

Martens D, Baesens B, van Gestel T, Vanthienen J. Comprehensible credit scoring models using rule extraction from support vector machines. *European Journal of Operational Research* 2007;183(3); 1466-1476.

Mingers J. An empirical comparison of pruning methods for decision tree induction. *Machine Learning* 1989;4(2); 227-243.

Momma M, Bennett KP 2002. A Pattern Search Method for Model Selection of Support Vector Regression. In: *Proc. of the 2nd SIAM Intern. Conf. on Data Mining.* Arlington, VA, USA.

Mues C, Baesens B, Files CM, Vanthienen J. Decision diagrams in machine learning: An empirical study on real-life credit-risk data. *Expert Systems with Applications* 2004;27(2); 257.

Newman DJ, Hettich S, Blake CL, Merz CJ, UCI Repository of Machine Learning Databases, 1998 (www.ics.uci.edu/~mlearn/MLRepository.html).

Provost F, Domingos P. Tree induction for probability-based ranking. *Machine Learning* 2003;52(3); 199-215.

Quinlan JR. *C4.5 : Programs for Machine Learning*. Morgan Kaufmann: San Mateo; 1993.

Rakotomamonjy A. Variable selection using SVM based criteria. *Journal of Machine Learning Research* 2003;3; 1357-1370.

Shin H, Cho S. Response modeling with support vector machines. *Expert Systems with Applications* 2006;30(4); 746-760.

Sindhwani V, Bhattacharyya P, Rakshit S 2001. Information Theoretic Feature Crediting in Multiclass Support Vector Machines. In: *Proc. of the 1st SIAM Intern. Conf. on Data Mining.* Chicago, IL, USA.

Sindhwani V, Keerthi SS 2007. Newton Methods for Fast Solution of Semi-Supervised Linear SVMs. In: Bottou L, Chapelle O, DeCoste D, Weston J (Eds.), *Large Scale Kernel Machines*. MIT Press: Cambridge; 2007. pp. 155-174.

Stoppiglia H, Dreyfus G, Dubois R, Oussar Y. Ranking a random feature for variable and feature selection. *Journal of Machine Learning Research* 2003;3; 1399-1414.

Thomas LC, Oliver R, Hand DJ. A survey of the issues in consumer credit modelling research. *Journal of the Operational Research Society* 2005;56(9); 1006-1015.

Van den Poel D, Lariviere B. Customer attrition analysis for financial services using proportional hazard models. *European Journal of Operational Research* 2004;157(1); 196-217.

Van Gestel T, Suykens JAK, Baesens B, Viaene S, Vanthienen J, Dedene G, De Moor B, Vandewalle J. Benchmarking least squares support vector machine classifiers. *Machine Learning* 2004;54(1); 5-32.

Vapnik VN. *The Nature of Statistical Learning Theory*. Springer: New York; 1995.

Vapnik VN, Chapelle O. Bounds on error expectation for support vector machines. *Neural Computation* 2000;12(9); 2013-2036.

Viaene S, Baesens B, Van Gestel T, Suykens JAK, Van den Poel D, Vanthienen J, De Moor B, Dedene G. Knowledge discovery in a direct marketing case using least squares support vector machines. *International Journal of Intelligent Systems* 2001;16(9); 1023-1036.

Viaene S, Derrig RA, Baesens B, Dedene G. A comparison of state-of-the-art classification techniques for expert automobile insurance claim fraud detection. *Journal of Risk & Insurance* 2002;69(3); 373-421.

Weston J, Elisseeff A, BakIr G, Sinz F, The Spider, 2006 (http://www.kyb.tuebingen.mpg.de/bs/people/spider/main.html).

Wu G, Chang EY. KBA: Kernel boundary alignment considering imbalanced data distribution. *IEEE Transactions on Knowledge and Data Engineering* 2005;17(6); 786-795

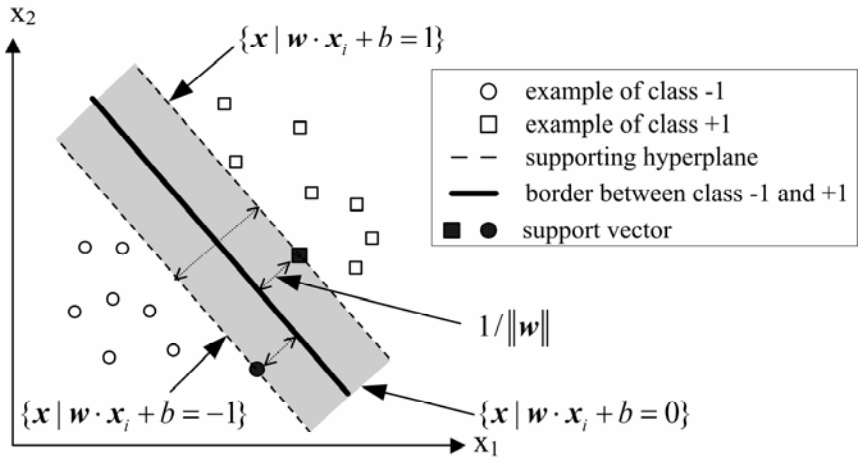


Fig. 1: Linear separation of two classes -1 and +1 in two-dimensional space with SVM classifier.

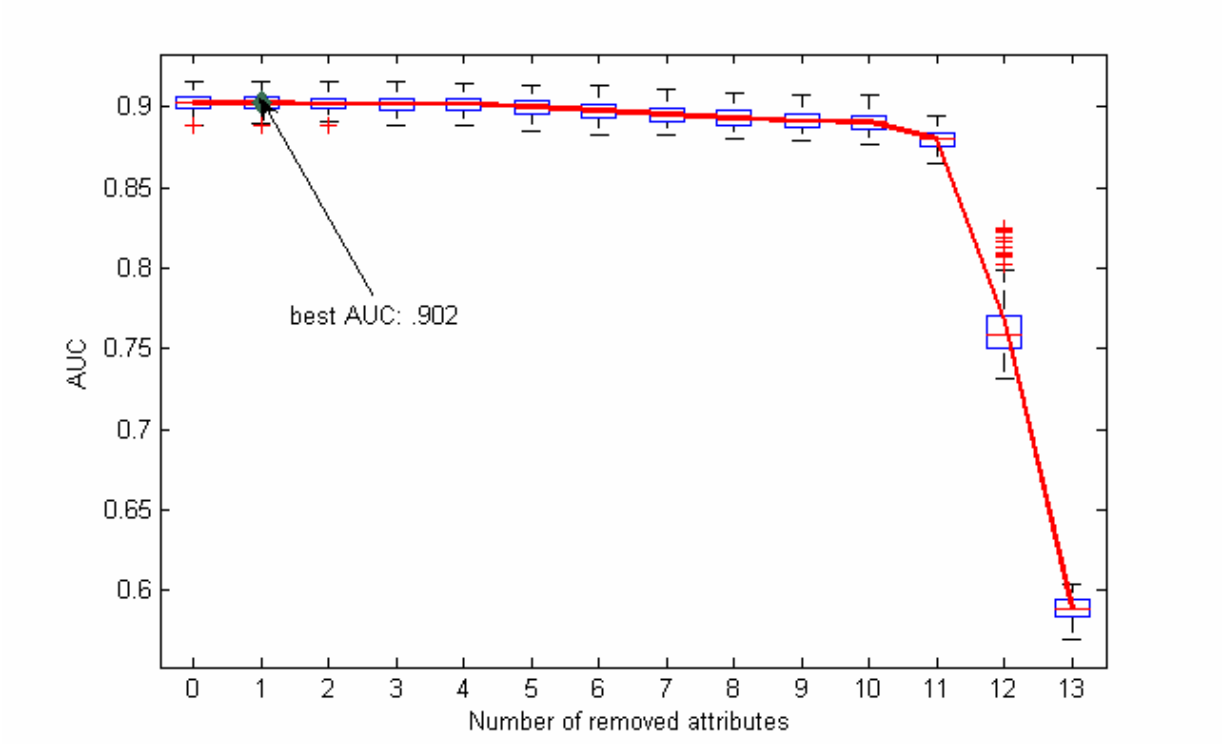


Fig. 2: Development of predictive performance during iterative attribute removal for the Adult dataset.

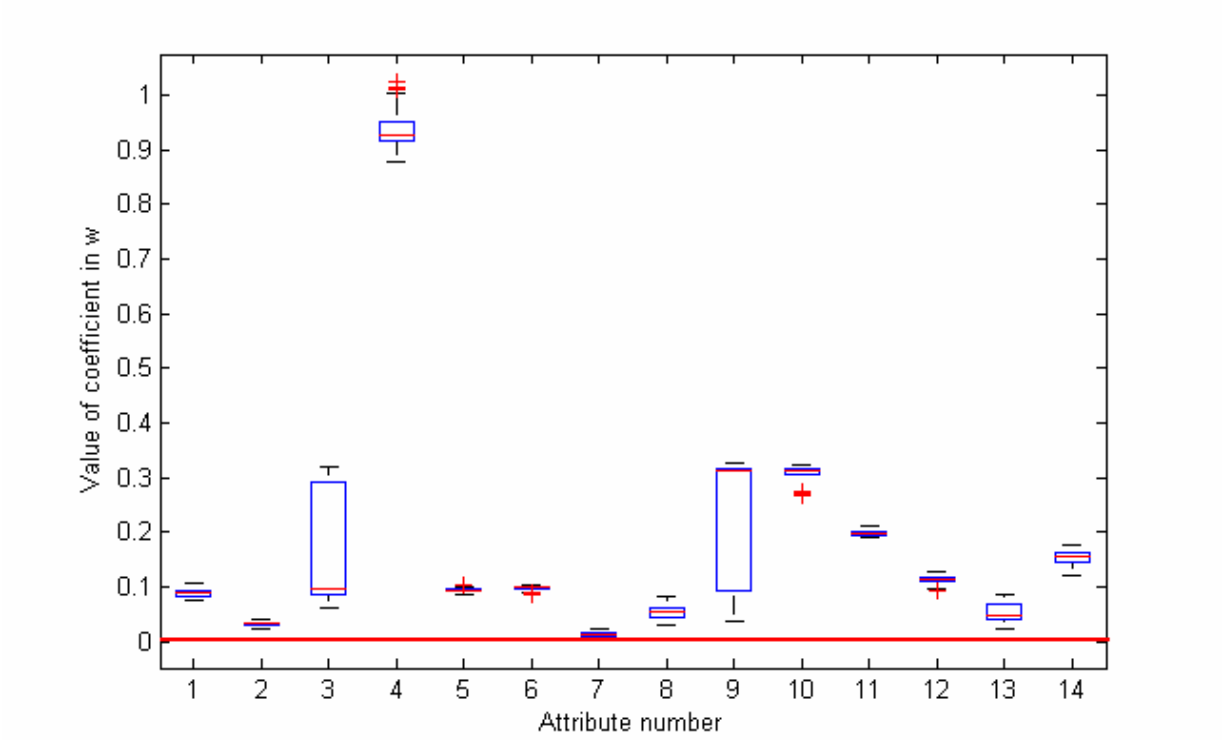


Fig. 3: Distribution of coefficients in the SVM weight vector ***w*** for the Adult dataset.

TABLE 1: DESCRIPTION OF THE DATASET USED FOR EMPIRICAL EVALUATIONS

	No. examples	No. attributes*	Prior class +1	Prior class -1
AC	690	14	44.49	55.51
GC	1000	24	30.00	70.00
Adult	48842	14	23.93	76.07
DMC 2000	38890	96	5.87	94.13
DMC 2001	28128	106	50.00	50.00
DMC 2002	20000	101	10.00	90.00
DMC 2004	40292	107	20.43	79.57
DMC 2005	50000	119	5.80	94.20
DMC 2006	16000	24	47.71	52.29

*Note that high dimensionality in some problems originates from dummy encoding of categorical attributes (Crone et al., 2006).

TABLE 2: COMPARATIVE RESULTS OF TWO FEATURE SELECTION MECHANISMS

		AC	GC	Adult	DMC 2000	DMC 2001	DMC 2002	DMC 2004	DMC 2005	DMC 2006
Full	AUC	.92 (.02)	.79 (.01)	.90 (.00)	.81 (.01)	.66 (.00)	.66 (.01)	.85 (.00)	.67 (.01)	.60 (.01)
	$\|N\|$	14	24	14	96	106	101	107	119	24
MFS	AUC	.92 (.02)	.78 (.01)	.90 (.00)	.81 (.01)	.66 (.01)	.65 (.00)	.85 (.00)	.67 (.01)	.60 (.01)
	$\|N\|$	6.7/52%	16.3/32%	13.8/1%	55.6/42%	25.1/76%	47.9/53%	52.2/51%	76.9/35%	20.7/14%
RFS	AUC	.92 (.02)	.77 (.02)	.90 (.00)	.81 (.01)	.66 (.01)	.65 (.01)	.76 (.27)	.66 (.01)	.59 (.01)
	$\|N\|$	11.3/19%	17.3/28%	12.4/1%	64.3/33%	48.5/54%	66.2/35%	64.6/40%	82/31%	19.8/18%

The second row gives the average number of attributes selected by the respective strategy. The format is: No. of attributes/percent reduction compared to the full attribute set. Values in square brackets denote the standard deviation of AUC over ten random validation datasets. MFS refers to selecting the attribute set with maximal performance whereas RFS represents the usage random gauge attributes for feature selection.

TABLE 3: IMPACT OF SV-SAMPLING ON DATASET SIZE AND PREDICTIVE PERFORMANCE

	Without SV-sampling			With SV-sampling		
	Instances	Prior +1	AUC	Instances*	Prior +1	AUC
AC	460	44.7%	0.919 (.02)	322/30.0%	50.0%	0.920 (.02)
GC	667	30.7%	0.785 (.02)	611/8.3%	33.7%	0.792 (.03)
Adult	32562	23.9%	0.903 (.00)	19421/40.4%	35.7%	0.903 (.00)
DMC 2000	25927	5.9%	0.865 (.01)	18241/29.6%	8.3%	0.854 (.01)
DMC 2001	18752	50.1%	0.664 (.01)	18442/1.7%	50.9%	0.665 (.01)
DMC 2002	13334	10.0%	0.667 (.01)	13078/1.9%	10.1%	0.660 (.01)
DMC 2004	26862	20.5%	0.849 (.00)	18898/29.6%	27.7%	0.849 (.00)
DMC 2005	33334	5.8%	0.677 (.01)	30645/8.1%	6.3%	0.674 (.00)
DMC 2006	10667	47.7%	0.735 (.02)	10650/0.2%	48.5%	0.725 (.03)

* Format: Number of support vectors/percent reduction compared to using all examples. AUC estimates are based upon validation data, whereby values in square brackets denote the standard deviation over ten randomly selected validation sets.

TABLE 4: COMPARISON OF LINE-SEARCH VERSUS PATTERN-SEARCH FOR RBF-SVM MODEL SELECTION

	Line-search heuristic			Pattern search heuristic		
	AUC	Iter.*	Runtime in sec.	AUC	Iter.	Runtime in sec.
AC	.902 (.019)	19	19 (17.3)	.913 (.022)	21.7 (5.6)	5 (2.9)
GC	.777 (.015)	19	15 (0.7)	.776 (.028)	23.5 (7.1)	16 (5.0)
Adult	.767 (.031)	19	6055 (4589.1)	.730 (.075)	24.3 (2.8)	17638 (40854.4)
DMC 2000	.767 (.022)	19	3247 (575.5)	.786 (.050)	30.6 (8.8)	28395 (10945.0)
DMC 2001	.663 (.008)	19	3745 (514.6)	.659 (.003)	22.9 (3.6)	4401 (1102.2)
DMC 2002	.643 (.017)	19	3435 (6773.5)	.591 (.030)	27.9 (10.9)	4308 (4728.7)
DMC 2004	.828 (.023)	19	16256 (5698.5)	.847 (.003)	35.9 (7.6)	16690 (3845.1)
DMC 2005	.655 (.011)	19	9188 (2823)	.578 (.032)	30.7 (7.0)	83034 (42212)
DMC 2006	.708 (.019)	19	2768 (4450.7)	.746 (.003)	20.0 (2.4)	2996 (1306.3)

* The parameter range for line-search has been decreased to [-3, -2, 15] in comparison to Keerthi and Lin (2003) because very small *C*, and consequently large γ values (9) generally give poor results for the considered data. Note that large runtimes originate from using a rigorous ten-fold cross-validation on learning data to assess an individual setting of the SVM hyperparameters. AUC estimates are based upon validation data, whereby values in square brackets denote the standard deviation over ten randomly selected validation sets.

TABLE 5: COMPARATIVE ASSESSMENT OF THE PROPOSED REFERENCE MODEL BY MEANS OF AUC

	Ref.-model	RBF-SVM	LogReg	C 4.5	CART
AC	.911 (.011)	.916 (.020)	**.924 (.014)**	.837 (.032)	.888 (.030)
GC	.790 (.031)	.791 (.020)	**.799 (.022)**	.669 (.048)	.653 (.107)
Adult	**.903 (.002)**	**.903 (.002)**	**.903 (.002)**	.881 (.005)	.860 (.008)
DMC 2000	.807 (.006)	**.873 (.006)**	.552 (.007)	.724 (.019)	.711 (.011)
DMC 2001	**.665 (.004)**	.664 (.004)	.648 (.004)	.574 (.004)	.652 (.012)
DMC 2002	***.659 (.010)***	.597 (.010)	.651 (.010)	.555 (.009)	.500 (.000)
DMC 2004	***.848 (.003)***	.846 (.003)	.843 (.017)	.675 (.006)	.731 (.024)
DMC 2005	***.672 (.009)***	.587 (.005)	.647 (.004)	.519 (.012)	.500 (.000)
DMC 2006	.727 (.031)	.745 (.005)	.603 (.004)	**.747 (.007)**	.723 (.018)
Average AUC	**.776 (.100)**	.769 (.128)	.730 (.138)	.687 (.124)	.691 (.135)

Bold font indicates the highest average AUC for a particular dataset. Italic font is used for the reference model to indicate situations, in which it was automatically decided to use the linear L2-SVM for prediction rather than RBF-SVM. AUC estimates are based upon hold-out testing data, whereby values in square brackets denote the standard deviation over ten randomly selected test sets.

Benchmarking classification models for software defect prediction: A proposed framework and novel findings

Stefan LESSMANN, Bart BAESENS, Christophe MUES, and Swantje PIETSCH

S. Lessmann (corresponding author) is with the Institute of Information Systems, University of Hamburg, Von-Melle-Park 5, D-20146 Hamburg, Germany (telephone: +49.40.42838.4706, e-mail: lessmann@econ.uni-hamburg.de).
B. Baesens is with the Faculty of Economic and Applied Economic Sciences, K.U.Leuven; Vlerick Leuven Ghent Management School, and the School of Management, University of Southampton (e-mail:Bart.Baesens@econ.kuleuven.ac.be).
C. Mues is with the School of Management, University of Southampton, SO17 1BJ, Southampton, UK (e-mail: c.mues@soton.ac.uk).
S. Pietsch is with the Institute of Information Systems, University of Hamburg, Von-Melle-Park 5, D-20146 Hamburg, Germany (e-mail: mailing@swantje-pietsch.de).

Abstract— **Software defect prediction strives to improve software quality and testing efficiency by constructing predictive classification models from code attributes to enable a timely identification of fault-prone modules. Several classification models have been evaluated for this task. However, due to inconsistent findings regarding the superiority of one classifier over another and the usefulness of metric-based classification in general, more research is needed to improve convergence across studies and further advance confidence in experimental results. We consider three potential sources for bias: comparing classifiers over one or a small number of proprietary datasets, relying on accuracy indicators that are conceptually inappropriate for software defect prediction and cross-study comparisons, and finally, limited use of statistical testing procedures to secure empirical findings. To remedy these problems, a framework for comparative software defect prediction experiments is proposed and applied in a large-scale empirical comparison of 22 classifiers over ten public domain datasets from the NASA Metrics Data repository. Our results indicate that the importance of the particular classification algorithm may have been overestimated in previous research since no significant performance differences could be detected among the top-17 classifiers.**

Index Terms—**Complexity measures, data mining, formal methods, statistical methods, software defect prediction**

I. Introduction

THE development of large and complex software systems is a formidable challenge, and activities to support software development and project management processes are an important area of research. This paper considers the task of identifying error prone software modules by means of metric-based classification, referred to as *software defect prediction*. It has been observed that the majority of a software system's faults are contained in a small number of modules [1, 20]. Consequently, a timely identification of these modules facilitates an efficient allocation of testing resources and may enable architectural improvements by suggesting a more rigorous design for high-risk segments of the system (e.g., [4, 8, 19, 33, 34, 44, 51, 52]).

Classification is a popular approach for software defect prediction and involves categorizing modules, represented by a set of software metrics or code attributes, into fault-prone (fp) and non fault-prone (nfp) by means of a classification model derived from data of previous development projects [57]. Various types of classifiers have been applied to this task, including statistical procedures [4, 28, 47], tree-based methods [24, 30, 43, 53, 58], neural networks [29, 31], and analogy-based approaches [15, 23, 32]. However, as noted in [48, 49, 59], results regarding the superiority of one method over another or the usefulness of metric-based classification in general are not always consistent across different studies. Therefore, "*we need to develop more reliable research procedures before we can have confidence in the conclusion of comparative studies of software prediction models*." [49].

We argue that the size of the study, the way predictive performance is measured, as well as the type of statistical test applied to secure conclusions have a major impact on cross-study comparability and may have produced inconsistent findings. In particular, several (especially early) studies in software defect prediction had to rely upon a small number of, commonly proprietary,

datasets, which naturally constrains the generalizability of observed results as well as replication by other researchers (see also [44]). Furthermore, different accuracy indicators are used across studies, possibly leading to contradictory results [49], especially if these are based on the number of misclassified fp and nfp modules. Finally, statistical hypothesis testing has only been applied to a very limited extent in the software defect prediction literature. As indicated in [44, 49], it is standard practice to derive conclusions without checking significance.

In order to remedy these problems, we propose a framework for organizing comparative classification experiments in software defect prediction and conduct a large-scale benchmark of 22 different classification models over ten public-domain datasets from the NASA Metrics Data (MDP) repository [10] and the PROMISE repository [56]. Comparisons are based on the *area under the receiver operating characteristics curve (AUC).* As argued later in the paper, the AUC represents the most informative and objective indicator of predictive accuracy within a benchmarking context. Furthermore, we apply state-of-the-art hypothesis testing methods [12] to validate the statistical significance of performance differences among different classification models. Finally, the benchmarking study assesses the competitive performance of several established and novel classification models so as to appraise the overall degree of accuracy that can be achieved with (automated) software defect prediction today, investigate whether certain types of classifiers excel and, thereby, support the (pre-)selection of candidate models in practical applications. In this respect, our study can also be seen as a follow-up to Menzies et al.'s recent paper [44] on defect predictions, providing additional results as well as suggestions for a methodological framework.

The paper is organized as follows: Section II first reviews accuracy indicators for classification and discusses the distinctive merits of receiver operating characteristic (ROC) analysis, after

which statistical testing procedures for model comparisons are presented. Section III is devoted to the benchmarking experiment and discusses the respective setup, findings, as well as limitations. Conclusions are given in Section IV.

II. Components Of The Benchmarking Framework

In this section, we present the two major components of our framework. First, we discuss the difficulties associated with assessing a classification model in software defect prediction and advocate the use of the AUC to improve cross-study comparability. Subsequently, the statistical testing procedures applied within the benchmarking experiment are introduced.

A. Accuracy indicators for assessing binary classification models

The task of (binary) classification can be defined as follows: Let $S = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{N}$ be a training dataset of *N* examples, where $\boldsymbol{x}_i \in \Re^M$ represents a software module that is characterized by *M* software metrics and $y_i \in \{\text{nfp}, \text{fp}\}$ denotes its binary class label. A classification model is a mapping from instances $\boldsymbol{x}$ to predicted classes *y*: $f(\boldsymbol{x}) : \Re^M \mapsto \{\text{nfp}, \text{fp}\}$.

Binary classifiers are routinely assessed by counting the number of correctly predicted modules over hold-out data. This procedure has four possible outcomes: If a module is fp and is classified accordingly, it is counted as true positive (TP); if it is wrongly classified as nfp, it is counted as false negative (FN). Conversely, a nfp module is counted as true negative (TN) if it is classified correctly, or as false positive (FP) otherwise. El-Eman et al. describe a large number of performance indicators which can be constructed from these four basic figures [15].

A defect prediction model should identify as many fp modules as possible while avoiding false alarms. Therefore, classifiers are predominantly evaluated by means of their true positive rate

(TPR), also known as sensitivity, rate of detection, or hit rate, and by their false positive rate (FPR) or false alarm rate (e.g., [24, 32, 44, 67]).

$$\mathrm{TPR} = \mathrm{TP}/(\mathrm{FN}+\mathrm{TP})\,;\ \mathrm{FPR} = \mathrm{FP}/(\mathrm{TN}+\mathrm{FP})\,. \tag{1}$$

We argue that such error-based metrics, although having undoubted practical value, are conceptually inappropriate for empirical comparisons of the competitive performance of classification algorithms. This is because they are constructed from a discrete classification of modules into fp and nfp. Most classifiers do not produce such crisp classifications but probability estimates or confidence scores, which represent the likelihood that a module belongs to a particular class. Consequently, threshold values have to be defined for converting such continuous predictions into discrete classifications [17]. The Bayes rule of classification guides the choice of threshold value: Let $p(\mathrm{fp})$ and $p(\mathrm{nfp})$ denote the prior probabilities of fp and nfp modules, respectively. The objective of software defect classification is to estimate the a posteriori probability of a module with characteristics $\boldsymbol{x}$ to be fp, which we denote by $p(y = \mathrm{fp} \mid \boldsymbol{x})$, with analogous meaning for $p(y = \mathrm{nfp} \mid \boldsymbol{x})$. Let C_{FP} denote the cost of conducting a false positive error, i.e. classifying a nfp module incorrectly as fp, and C_{FN} the cost of a false negative error (misclassifying a fp module). Then Bayes rule (e.g., [27]) states that modules should be classified as fp, if:

$$\frac{p(\boldsymbol{x} \mid y = \mathrm{fp})}{p(\boldsymbol{x} \mid y = \mathrm{nfp})} > \frac{p(\mathrm{nfp}) \cdot C_{FP}}{p(\mathrm{fp}) \cdot C_{FN}}, \tag{2}$$

whereby $p(\boldsymbol{x} \mid y = \mathrm{fp})$ and $p(\boldsymbol{x} \mid y = \mathrm{nfp})$ represent the so called class conditional probabilities, which are related to the a posteriori probabilities via Bayes theorem.

The Bayes optimal threshold, i.e. the right hand side of (2), depends on prior probabilities and misclassification costs, or their respective ratios. However, within a benchmarking context, classifiers should be compared over several datasets from several different software releases and/or

projects (see also [9, 44, 52]) and it is extremely unlikely that information on class and cost distributions are available for every dataset. Consequently, the necessary information to determine meaningful and objective threshold values is usually missing. This problem can be alleviated by relying on default values or estimating settings from the data [33]. However, two studies that use the same classifiers and datasets could easily come to different conclusions just because different procedures for determining classification thresholds are employed. Furthermore, it should be noted that detailing the concrete strategy for determining thresholds is not standard practice in the defect prediction literature. Consequently, comparing algorithms by means of discrete classifications leaves considerable room for bias and may cause inconsistencies across studies. Our key point is that this risk can easily be avoided if defect predictors are assessed independently from thresholds, i.e. over all possible combinations of misclassification costs and prior probabilities of fp and nfp modules. Receiver operating characteristic (ROC) analysis is a tool that realizes such an evaluation.

The ROC-graph is a 2-dimensional illustration of TPR on the Y-axis versus FPR on the X-axis (Fig. 1). A ROC-curve is obtained by varying the classification threshold over all possible values [17]. Thereby, each ROC-curve passes through the points (0,0), representing a classifier that always predicts nfp, and (1,1), the opposite case [44]. The ideal point is the upper-left corner (0,1) since such a classifier accurately identifies all fp modules (TPR=1) while making no error (FPR=0). Hence, points towards the north-west are preferable, i.e. achieve high hit rate with low FPR. Advantages of ROC-analysis are its robustness towards imbalanced class distributions and to varying and asymmetric misclassification costs [54]. Therefore, it is particularly well suited for software defect prediction tasks which naturally exhibit these characteristics [33, 44].

To compare different classifiers, their respective ROC-curves are drawn in ROC-space. Fig. 1 provides an example of three classifiers C_1, C_2, and C_3. C_1 is a dominating classifier because its ROC-curve is always above that of its competitors, i.e. it achieves a higher TP rate for all FP rates.

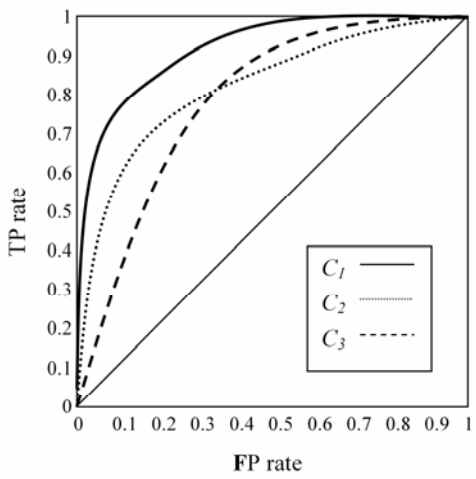


Fig. 1: Exemplary ROC curve of three classifiers with dominating classifier C_1.

As ROC-curves of different classifiers may intersect (e.g., curves C_2 and C_3) one often calculates the area under the ROC curve (abbreviated by AUC) as a single scalar measure of expected performance [6]. Higher AUC values indicate that the classifier is on average more to the upper-left region of the graph.

The AUC has the potential to significantly improve convergence across empirical experiments in software defect prediction because it separates predictive performance from operating conditions, i.e. class and cost distributions, and thus represents a general measure of predictiveness. The importance of such a general indicator in comparative experiments is reinforced when considering the discussion following Menzies et al.'s paper [44] about whether the accuracy of their models is or is not sufficient for practical applications and whether method A is or is not better than method B [42, 66]. Furthermore, the AUC has a clear statistical interpretation: It measures

the probability that a classifier ranks a randomly chosen fp module higher than a randomly chosen nfp module, which is equivalent to the Wilcoxon test of ranks [17]. Consequently, any classifier achieving AUC well above 0.5 is demonstrably effective for identifying fp modules and gives valuable advice which modules should receive particular attention in software testing.

B. Statistical comparison of classification models

Few reported studies in software defect prediction make use of statistical inference. For example, analysis of variance (ANOVA) is applied in [33, 34, 58] to determine if observed performance differences between candidate methods are statistically significant. However, as indicated by [44, 49], the prevailing approach is to derive conclusions solely from empirical results without applying formal hypothesis tests. As will be shown later, this practice may be misleading and consequently represents another possible source for inconsistency across experiments.

In a recent article, Demšar reviews the problem of benchmarking classifiers and offers valuable guidance on how to organize such comparisons in a statistically sound manner [12]. Subsequently, we summarize his recommendations for the comparison of multiple algorithms over multiple datasets which we deem most relevant for software defect prediction.[1]

The null hypothesis, H_0, being tested in this setting is that all algorithms perform alike. That is, it is assumed that performance differences observed within an empirical experiment are just due to random chance. Performance may be measured by means of an arbitrary accuracy indicator, e.g., the AUC. Testing the significance of differences between multiple means, i.e. mean accuracies across different datasets, is a well known statistical problem and ANOVA is specifically designed for this purpose. However, Demšar explicitly discourages usage of ANOVA for comparing classifiers because it is based on assumptions that are most likely violated within this setting

[1] Note that dedicated tests are applicable for comparing only two classifiers over a single or multiple datasets [12].

[12]. In particular, ANOVA assumes that: (1) performance differences are distributed normally, which can be taken for granted only if the sample size is large, i.e. the algorithms are compared over many datasets (~ 30); (2) all classifiers exhibit the same variance in predictive performance over all datasets (homogeneity of variance); (3) the variance in performance differences across two classifiers is identical for all possible pairs of classifiers (sphericity assumption) [65]. On the one hand, the validity of these assumptions is difficult to check when the number of samples (i.e. datasets) is limited. On the other hand, violations, especially with respect to non-sphericity, have been shown to be highly detrimental to ANOVA and especially to the subsequently performed post-hoc tests [55]. Consequently, Demšar recommends the Friedman test for classifier comparisons, which is a non-parametric alternative to ANOVA and relies on less restrictive assumptions [12].

Friedman's test is based on ranked performances rather than actual performance estimates and is thereby less susceptible to outliers. All classifiers are ranked according to their performance in ascending order for each dataset and the mean rank of a classifier *i*, AR_i, is computed across all datasets. With *K* representing the overall number of datasets, *L* the number of classifiers and r_j^i the rank of classifier *i* on dataset *j*, the test statistic of the Friedman test is calculated as:

$$\chi_F^2 = \frac{12K}{L(L+1)}\left[\sum_{i=1}^{L} AR_i^2 - \frac{L(L+1)^2}{4}\right], \qquad AR_i = \frac{1}{K}\sum_{j=1}^{K} r_j^i \tag{3}$$

and is distributed according to the Chi-Square distribution with *L*-1 degrees of freedom [65].

If the value of the test statistic is large enough to reject the null hypothesis, it may be concluded that performance differences among classifiers are non-random. In this case, a so called post-hoc test can be applied to detect which specific classifiers differ significantly. Demšar rec-

ommends the test of Nemenyi for this task [12]. For all pairs of classifiers, it tests the null hypothesis that their respective mean ranks are equal, which may be rejected if the difference between their mean ranks exceeds the critical difference CD:

$$CD = q_{a,\infty,L}\sqrt{\frac{L(L+1)}{12K}}. \quad (4)$$

The value $q_{a,\infty,L}$ is based on the Studentized range statistic and is tabulated in standard statistical textbooks.[2]

III. Empirical Evaluation Of Candidate Classifiers On NASA MDP Data

In this section, we describe the setup of the benchmarking study and elaborate on the experimental design. Subsequently, the empirical results are presented in detail, together with a discussion of possible limitations and threats to validity.

A. Dataset characteristics

The data used in this study stems from the NASA MDP repository [10]. Ten software defect prediction datasets are analyzed including the eight sets used in [44] as well as two additional datasets (JM1 and KC1, see also Table 1). Each dataset comprises several software modules together with their number of faults and characteristic code attributes. After preprocessing, modules that contain one or more errors were labeled as fp, whereas error-free modules were categorized as nfp. Beside LOC-counts, the NASA MDP datasets include several Halstead attributes as well as McCabe complexity measures. The former estimate reading complexity by counting operators and operands in a module, whereas the latter are derived from a module's flow graph. The reader is referred to [26, 41, 44] for a more detailed description of code attributes or the ori-

[2] Note that more powerful post-hoc tests are available if one is interested in the performance of one particular classifier, e.g., to test if a novel technique performs significantly better than an established benchmark (see [11] for details).

gin of the MDP datasets. Individual attributes per dataset together with some general statistics are given in Table 1.

TABLE 1: CODE ATTRIBUTES WITHIN THE MDP DATASETS

		NASA MDP dataset									
		CM1	KC1	KC3	KC4	MW1	JM1	PC1	PC2	PC3	PC4
LOC counts	LOC_total	X	X	X	X	X	X	X	X	X	X
	LOC_blank	X	X	X		X	X	X	X	X	X
	LOC_code_and_comment	X	X	X		X	X	X	X	X	X
	LOC_comments	X	X	X		X	X	X	X	X	X
	LOC_executable	X	X	X		X	X	X	X	X	X
	Number_of_lines	X		X		X		X	X	X	X
Halstead attributes	content	X	X	X		X	X	X	X	X	X
	difficulty	X	X	X		X	X	X	X	X	X
	effort	X	X	X		X	X	X	X	X	X
	error_est	X	X	X		X	X	X	X	X	X
	length	X	X	X		X	X	X	X	X	X
	level	X	X	X		X	X	X	X	X	X
	prog_time	X	X	X		X	X	X	X	X	X
	volume	X	X	X		X	X	X	X	X	X
	num_operands	X	X	X		X	X	X	X	X	X
	num_operators	X	X	X		X	X	X	X	X	X
	num_unique_operands	X	X	X		X	X	X	X	X	X
	num_unique_operators	X	X	X		X	X	X	X	X	X
McCabe attributes	cyclomatic_complexity	X	X	X	X	X	X	X	X	X	X
	cyclomatic_density	X		X	X	X		X	X	X	X
	design_complexity	X	X	X	X	X	X	X	X	X	X
	essential_complexity	X	X	X	X	X	X	X	X	X	X
Miscellaneous	branch_count	X	X	X	X	X	X	X	X	X	X
	call_pairs	X		X	X	X		X	X	X	X
	condition_count	X		X		X		X	X	X	X
	decision_count	X		X		X		X	X	X	X
	decision_density	X		X		X		X	X	X	X
	design_density	X		X	X	X		X	X	X	X
	edge_count	X		X	X	X		X	X	X	X
	essential_density	X		X	X	X		X	X	X	X
	parameter_count	X		X		X		X	X	X	X
	maintenance_severity	X		X	X	X		X	X	X	X
	modified_condition_count	X		X		X		X	X	X	X
	multiple_condition_count	X		X		X		X	X	X	X
	global_data_complexity			X							
	global_data_density			X							
	normalized_cyclomatic_compl.	X		X	X	X		X	X	X	X
	percent_comments	X		X		X		X	X	X	X
	node_count	X		X	X	X		X	X	X	X
Number of code attributes		37	21	39	13	37	21	37	37	37	37
Number of modules		505	1571	458	125	403	9537	1059	4505	1511	1347
Number of fp modules		48	319	43	61	31	1777	76	23	160	178
Percentage of fp modules		9.50	20.31	9.39	48.80	7.69	18.63	7.18	0.51	10.59	13.21

B. Experimental design

The benchmarking experiment aims at contrasting the competitive performance of several classification algorithms. To that end, an overall number of 22 classifiers are selected, which may be grouped into the categories of statistical approaches, nearest-neighbor methods, neural networks, support vector machines, tree-based methods, and ensembles. The selection aims at achieving a balance between established techniques such as Naïve Bayes, decision trees, or logistic regression, and novel approaches that have not yet found widespread usage in defect prediction (e.g., different variants of support vector machines, logistic model trees, or random forests). The classifiers are sketched in Table 2, together with a brief description of their underlying paradigms. A detailed description of most methods can be found in general textbooks like [14, 27]; specific references are given for less known/novel techniques.

The merit of a particular classifier (in terms of the AUC) is estimated on a randomly selected hold-out test set (so called split-sample setup). More specifically, all datasets are randomly partitioned into training and test set using 2/3 of the data for model building and 1/3 for performance estimation. Besides providing an unbiased estimate of a classifier's generalization performance, the split-sample setup offers the advantage of enabling easy replication, which constitutes an important part of empirical research [2, 19, 49, 50]. Furthermore, its choice is motivated by the fact that the split-sample setup is the prevailing approach to assess predictive accuracy in software defect prediction [15, 16, 23, 28, 32, 33, 34, 37].

Several classification models exhibit adjustable parameters, also termed hyperparameters, which enable an adaptation of the algorithm to a specific problem. It is known that a careful tuning of such hyperparameters is essential to obtain a representative assessment of the classifier's potential (see, e.g., [3, 63]). For example, neural network models require specification of net-

work architecture (number of hidden layers, number of nodes per layer), whereas a pruning strategy has to be defined for tree-based classifiers. We adopt a grid-search approach to organize this model selection step. That is, a set of candidate values are defined for each hyperparameter and all possible combinations are evaluated empirically by means of 10-fold cross validation on the training data. The parameter combination with maximal cross-validation performance is retained and a respective classification model is constructed on the whole training dataset. Since we advocate using the AUC for classifier comparison, the same metric is used during model selection to guide the search towards predictive parameter settings. The respective candidate values are described in Appendix I to enable a replication of our experiments.

TABLE 2: CLASSIFICATION MODELS EMPLOYED IN THE COMPARATIVE EXPERIMENT

Classification model		Philosophy
Statistical classifiers		Strive to construct a Bayes optimal classifier by estimating either posterior probabilities directly (LogReg), or class-conditional probabilities (LDA, QDA, NB, BayesNet) which are subsequently converted into posterior probabilities using Bayes' theorem. LDA/QDA assume a multivariate Gaussian density function, whereas NB is based on the assumption that attributes are conditionally independent, so that class-conditional probabilities can be estimated individually per attribute. BayesNet extends NB by explicitly modeling statements about independence and correlation among attributes. LARS adopts a different approach and consists of a multivariate linear regression model and heuristics to shrink the number of features. RVM has been proposed as an extension of the SVM (see below) which avoids the need to tune certain hyperparameters and may incorporate kernel functions SVMs are unable to process.
Linear Discriminant Analysis[2, 3]	(LDA)	
Quadratic Discriminant Analysis[2, 3]	(QDA)	
Logistic Regression[2, 3]	(LogReg)	
Naïve Bayes[1]	(NB)	
Bayesian Networks[1]	(BayesNet)	
Least-Angle Regression[2]	(LARS)	
Relevance Vector Machine[2] [62]	(RVM)	
Nearest neighbor methods		Belong to the group of analogy-based methods which classify a module by considering the *k* most similar examples. The definition of similarity differs among algorithms. An Euclidian distance is used in *k*-NN whereas K* employs an entropy-based distance function.
k-Nearest Neighbor[1]	(k-NN)	
K-Star[1] [11]	(K*)	
Neural Networks		Mathematical representations inspired by the functioning of the human brain. They depict a network structure which defines a concatenation of weighting, aggregation and thresholding functions that are applied to a software module's attributes to obtain an approximation of its posterior probability of being fp. The study includes two
Multi-Layer Perceptron[2, 4]	(MLP)	

Radial Basis Function Network[1]	(RBF net)	types of MLP classifiers which incorporate different approaches to avoid overfitting the training data, i.e. weight decay and Bayesian Learning.
Support vector machine-based classifiers		Utilize mathematical programming to optimize a linear decision function that discriminates between fp and nfp modules. A kernel function enables more complex decision boundaries by means of an implicit, nonlinear transformation of attribute values. This kernel function is polynomial for the VP classifier, whereas SVM and LS-SVM consider a radial basis function. L-SVM and LP are linear classifiers.
Support Vector Machine[2]	(SVM)	
Lagrangian SVM[2] [40]	(L-SVM)	
Least Squares SVM[2] [61]	(LS-SVM)	
Linear Programming[2]	(LP)	
Voted Perceptron[1] [22]	(VP)	
Decision tree approaches		Recursively partition the training data by means of attribute splits. The algorithms differ mainly in the splitting criterion which determines the attribute used in a given iteration to separate the data. C4.5 induces decision trees based on the information-theoretical concept of entropy, whereas CART uses the Gini criterion. ADT distinguishes between alternating splitter and prediction nodes. A prediction is computed as the sum over all prediction nodes an instance visits while traversing the tree.
C 4.5 Decision Tree[1]	(C 4.5)	
Classification and Regression Tree[2]	(CART)	
Alternating Decision Tree[1] [21]	(ADT)	
Ensemble methods		Meta-learning schemes that embody several base-classifiers. These are built independently and participate in a voting procedure to obtain a final class prediction. RndFor incorporates CART as base learner, whereas LMT utilizes LogReg. Each base learner is derived from a limited number of attributes. These are selected at random within the RndFor procedure, whereby the user has to predefine their number. LMT considers only univariate regression models, i.e. uses one attribute per iteration, which is selected automatically.
Random Forest[1] [7]	(RndFor)	
Logistic Model Tree[1] [36]	(LMT)	

[1] Classifier is implemented using the YALE workbench [45].
[2] Classifier is implemented using the MATLAB environment.
[3] These classifiers fail to produce a classification model if all attributes are used. Therefore, they are trained in conjunction with a backward-feature elimination heuristic [25] (see also Appendix I).
[4] Subsequently, we use the abbreviation MLP-1 to refer to a multi-layer perceptron neural network which has been trained with a weight decay penalty to prevent overfitting, whereas MLP-2 represents a network which uses a Bayesian learning paradigm (see also Appendix I).

C. Experimental results

Next, we present the results of the empirical comparison in terms of the AUC. The last column of Table 3 reports the mean rank AR_i (3) of each classifier over all MDP datasets, which constitutes the basis of the Friedman test. The classifier yielding the best AUC for a particular dataset is highlighted in bold face. Note that all figures are based on hold-out test data; results on training data are omitted for brevity.

TABLE 3: HOLD-OUT TEST SET RESULTS OF 22 CLASSIFICATION ALGORITHMS OVER TEN NASA MDP DATASETS IN TERMS OF THE AUC

	CM1	KC1	KC3	KC4	MW1	JM1	PC1	PC2	PC3	PC4	*AR*
Statistical classifiers											
LDA	0.77	**0.78**	0.62	0.73	0.82	0.73	0.82	0.87	0.82	0.88	9.7
QDA	0.70	**0.78**	0.74	0.80	**0.83**	0.70	0.70	0.80	0.78	0.86	13.1
LogReg	0.80	0.76	0.61	0.74	0.82	0.73	0.82	0.86	0.82	0.89	10.0
NB	0.72	0.76	0.83	0.68	0.80	0.69	0.79	0.85	0.81	0.85	12.9
Bayes Net	0.79	0.75	0.83	0.80	0.82	0.73	0.84	0.85	0.80	0.90	8.7
LARS	0.84	0.75	0.80	0.76	0.74	0.72	0.70	0.30	0.79	0.90	13.3
RVM	0.82	0.76	0.74	0.74	0.75	0.72	0.84	0.91	0.82	0.89	10.4
Nearest neighbor methods											
k-NN	0.70	0.70	0.82	0.79	0.75	0.71	0.82	0.77	0.77	0.87	14.5
K*	0.76	0.68	0.71	0.81	0.71	0.69	0.72	0.62	0.74	0.83	17.1
Neural networks											
MLP-1	0.76	0.77	0.79	0.80	0.77	0.73	0.89	**0.93**	0.78	0.95	6.9
MLP-2	0.82	0.77	0.83	0.76	0.76	0.73	**0.91**	0.84	0.81	0.94	6.9
RBF net	0.58	0.76	0.68	0.73	0.65	0.69	0.64	0.79	0.78	0.79	17.8
Support vector machine-based classifiers											
SVM	0.70	0.76	**0.86**	0.77	0.65	0.72	0.80	0.85	0.77	0.92	13.0
L-SVM	0.80	0.76	0.82	0.76	0.76	0.73	0.86	0.83	**0.84**	0.92	7.7
LS-SVM	0.75	0.77	0.83	0.81	0.60	0.74	0.90	0.85	0.83	0.94	6.8
LP	**0.90**	0.75	0.74	0.83	0.74	0.72	0.73	0.88	0.82	0.92	9.3
VP	0.72	0.76	0.74	0.73	0.73	0.54	0.75	0.50	0.74	0.83	18.2
Decision tree approaches											
C4.5	0.57	0.71	0.81	0.76	0.78	0.72	0.90	0.84	0.78	0.93	11.6
CART	0.74	0.67	0.62	0.79	0.67	0.61	0.70	0.68	0.63	0.79	19.3
ADT	0.78	0.69	0.74	0.81	0.76	0.73	0.85	0.70	0.76	0.94	11.8
Ensemble methods											
RndFor	0.81	**0.78**	0.86	**0.85**	0.81	**0.76**	0.90	0.82	0.82	**0.97**	**4.0**
LMT	0.81	0.76	0.78	0.80	0.71	0.72	0.86	0.83	0.80	0.92	10.4

Most classifiers achieve promising AUC results of 0.7 and more, i.e. rank deficient modules higher than accurate ones with probability >70%. Overall, this level of accuracy confirms Menzies et al.'s conclusion that "*defect predictors are demonstrably useful*" for identifying fp modules and guiding the assignment of testing resources [44]. Furthermore, one observes a concentration of novel and/or sophisticated classifiers like RndFor, LS-SVMs, MLPs and Bayesian networks among the best performing algorithms. Whereas, e.g., analogy-based classification is a popular tool for software defect prediction and has been credited for its accuracy in several stud-

ies (e.g., [15, 23, 32, 34, 38, 60]), Table 3 seems to suggest that analogy-based approaches (*k*NN and K*) are outperformed when compared against these state-of-the-art competitors.

However, to evaluate individual classification models and verify if some are generally superior to others, it is important to test whether the differences in AUC are significant. This is confirmed when conducting the Friedman test: Its p-value of 2.1E-009 indicates that it is very unlikely that the observed performance differences among classifiers are just random. Consequently, one may proceed with a post-hoc test to detect which particular classifiers differ significantly. This is accomplished by applying Nemenyi's post hoc test ($\alpha = 0.05$), i.e. conducting all pairwise comparisons between different classifiers and checking which models' performance differences exceed the critical difference (4). The results of the pairwise comparisons are depicted in Fig. 2, utilizing a modified version of Demšar's significance diagrams [12]: The diagram plots classifiers against mean ranks, whereby all methods are sorted according to their ranks. The line segment to the right of each classifier represents its corresponding critical difference. That is, the right end of the line indicates from which mean rank onwards another classifier is outperformed significantly. For illustrative purpose, this threshold is highlighted with a vertical dotted line in three cases. The left most vertical line is associated with RndFor. Therefore, all classifiers right to this line perform significantly worse than RndFor. The second line separates the MLP-1 classifier from RBF net, VP and CART. Hence, these are significantly inferior to MLP-1 and any better-ranked method. Finally, the third line indicates that the Bayes net classifier is significantly better than CART.

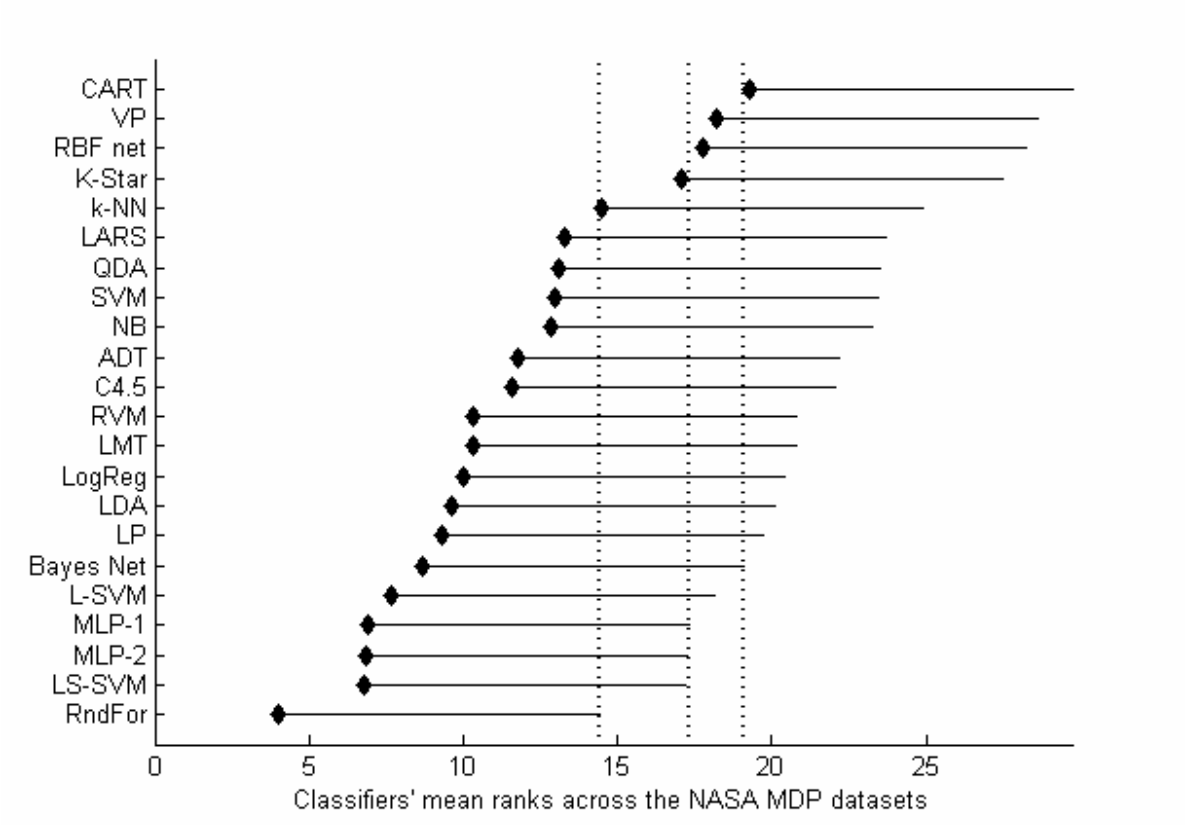


Fig. 2: Results of the pairwise comparisons of all classifiers using Nemenyi's post-hoc test with $\alpha = 0.5$.

The statistical comparison reveals an interesting finding: Despite noteworthy differences in terms of the AUC among competing classifiers, all methods – with few exceptions – do not differ significantly. This result may be explained as follows: The relationship between the code attributes and the dependent variable $y \in \{\text{fp} \mid \text{nfp}\}$ is clearly present, but limited (e.g., AUC ~0.7). This relationship is disclosed by almost all classifiers, whereas the remaining differences across methods are just random.[3] This view is reinforced when considering that relatively simple classifiers like LP, LogReg, LDA, and especially L-SVM provide respectable results. These techniques separate fp and nfp modules by means of a linear decision function and are consequently restricted to merely account for linear dependencies among code attributes. In other words, their competitive performance indicates that the degree of nonlinearity within the MDP datasets is

limited. Following this reasoning, one may conclude that the choice of classification modeling technique is less important than generally assumed and that practitioners are free to choose from a broad set of candidate models when building defect predictors.

However, it should be noted that Nemenyi's test checks the null hypothesis that two classifiers give equal performance. Failing to reject this H_0, does not guarantee that it is true. For example, Nemenyi's test is unable to reject the null hypothesis that RndFor and LARS have the same mean rank. This can mean that the performance differences between these two are just due to chance. But, the result could also be caused by a Type II error: Possibly, the Nemenyi test does not have enough power to detect a significant difference at $\alpha = 0.05$. In other words, only rejecting H_0 allows the conclusion that it is very likely (with probability $1-\alpha$) that two classifiers differ significantly.

With the former in mind, a general conclusion that may be drawn from the benchmarking experiment is that predictive performance alone does not suffice to appraise the merit of a classification model and has to be augmented by other criteria. For example, Vandecruys et al. [64] argue in favor of comprehensible classifiers and propose an Ant-Colonony optimization based detection system. Similarly, Menzies et al. point out that their preferred classifier, a Naïve Bayes model, is easy to interpret as well as computationally efficient [44]. Clearly, computational efficiency and transparency are desirable features of candidate classifiers, and it appears to be a promising area for future research to formalize these concepts, e.g., by developing a multi-dimensional classifier assessment system. Meanwhile, the results observed here confirm previous findings regarding the effectiveness of RndFor for software defect prediction [24] and allow recommending this classifier for future experiments or practical applications. It is fast to train

[3] The authors would like to thank an anonymous reviewer for suggesting this interpretation.

and requires only moderate parameter tuning, i.e. is robust towards parameter settings. Furthermore, RndFor naturally assesses the relevance of individual code attributes (see [7]) and thereby provides not just an accurate but also an understandable model.

D. Threats to validity

When conducting an empirical study it is important to be aware of potential threats to the validity of the obtained results and derived conclusions. A possible source of bias relates to the data used, e.g., its measurement accuracy and representativeness if results are to be generalized. Using public domain data secures the results in so far that they can be verified by replication and compared with findings from previous experiments. Also, several authors have argued in favor of the appropriateness and representativeness of the NASA MDP repository and/or used some of its datasets for their experiments (e.g., [24, 35, 44, 64, 67]). Therefore, we are confident that the obtained results are relevant for the software defect prediction community.

Despite general suitability of the data, the sampling procedure might bias results and prevent generalization. We consider a split-sample setup with randomly selected test records (1/3 of the available dataset). This is a well established approach for comparative classification experiments and the size of the MDP datasets seems large enough to justify this setting. Compared to cross-validation or bootstrapping, the split sample setup saves a considerable amount of computation time, which, in turn, can be invested into model selection to ensure that the classifiers are well tuned to each dataset. It would be interesting to quantify possible differences between a split-sample setup and cross-validation/bootstrapping setups by means of empirical experimentation. However, this step is left for future research.

The selection of classifiers is another possible source of bias. Given the variety of available learning algorithms, there are still others that could have been considered. Our selection is

guided by the aim to find a meaningful balance between established techniques and novel approaches. We believe that the most important representatives of different domains (statistics, machine learning, etc.) are included.

Finally, it should be noted that classification is only a single step within a multi-stage data mining process [18]. Especially data pre-processing or engineering activities such as the removal of non-informative features or the discretization of continuous attributes may improve the performance of some classifiers (see, e.g., [13, 25]). For example, Menzies et al. report that their Naïve Bayes classifier benefits from feature selection and a log-filter pre-processor [44]. Such techniques have an undisputed value. However, a wide range of different algorithms for feature selection, discretization, scaling, etc. has been proposed in the data mining literature. A thorough assessment of several candidates seems computationally infeasible when considering a large number of classifiers at the same time. That is, each added individual pre-processing algorithm would multiply the computational effort of the whole study. Our view is that especially simple classifiers like Naïve Bayes or decision trees would benefit from additional pre-processing activities (see [13]), whereas sophisticated techniques are well prepared to cope with, e.g., large and correlated feature sets through inbuilt regularization facilities [7, 27, 61]. As our results indicate that most simple classifiers are already competitive to more sophisticated approaches, i.e. not significantly inferior, it seems unlikely that pre-processing activities would alter our overall conclusion that most methods do not differ significantly in terms of predictive accuracy.

IV. Conclusions

In this paper, we reported on a large scale empirical comparison of 22 classification models over 10 public domain software development datasets from the NASA MDP repository. The area under the receiver operating characteristic curve was recommended as the primary accuracy in-

dicator for comparative studies in software defect prediction since it separates predictive performance from class and cost distributions, which are project-specific characteristics that may be unknown or subject to change. Therefore, the AUC-based evaluation has the potential to significantly improve convergence across studies. Another contribution along this line was the discussion and application of statistical testing procedures which are particularly appropriate for contrasting classification models.

The overall level of predictive accuracy across all classifiers confirmed the general appropriateness of defect prediction to identify fp software modules and guide the assignment of testing resources [44]. In particular, previous findings regarding the efficacy of RndFor for defect prediction [24] were confirmed.

However, where the statistical comparison of individual models is concerned, the major conclusion is that the predictive accuracy of most methods does not differ significantly according to a Nemenyi post hoc test ($\alpha = 0.05$). This suggests that the importance of the classification model may have been over-estimated in previous research, hence illustrating the relevance of statistical hypothesis testing. Given that basic models, and especially linear ones such as LogReg, LP and LDA, give similar results to more sophisticated classifiers, it is evident that most datasets are fairly well linearly separable. In other words, simple classifiers suffice to model the relationship between static code attributes and software defect.

Consequently, the assessment and selection of a classification model should not be based on predictive accuracy alone, but comprise several additional criteria like computational efficiency, ease of use, and especially comprehensibility. Comprehensible models reveal the nature of detected relationships and help to improve our overall understanding of software failures and their sources, which, in turn, may enable the development of novel predictors of fault-proneness. In

fact, efforts to design new software metrics and other explanatory variables appear to be a particularly promising area for future research and have the potential to achieve general accuracy improvements across all types of classifiers. We hope that the proposed framework will offer valuable guidance for appraising the potential of respective advancements.

APPENDIX I: MODEL SELECTION METHODOLOGY

The following Section reports hyperparameter settings which have been considered for individual classifiers during model selection. These settings may be useful for other researchers when trying to replicate the results observed within this study. It should be noted that, since a hold-out test set of 1/3 is randomly selected and removed from the overall dataset, we employ 10-fold cross validation during model selection to assess individual candidate hyperparameter settings, to avoid bias because of a small training sample. The overall experimental setup has been motivated in Section III.B and is summarized in Fig. 3.

```
D = List of datasets
C = List of classifiers
P = Dictionary of hyperparameter settings per classifier

For Each d in D
  train = randomly select 2/3 of d
  test  = d - train
  For Each c in C
    p_opt   = ModelSel(train, c, P[c])
    model   = BuildClassifier(train, c, p_opt)
    auc[c,d]= ApplyClassifier(test, model)
output auc
#-----------------------------------------------------
ModelSel(data, classifier, hyperparameters)
  crossval = generate 10 bins from data
  For i=1 to 10
    validate = crossval[i]
    learn = crossval - validate
    For Each p in hyperparameters
      model = BuildClassifier(learn, classifier, p)
      cv_auc[p,i] = ApplyClassifier(validate, model)
  auc = compute mean performance over cross-validation bins
  return hyperparameters( Max(auc) )

BuildClassifier(data, classifier, para)
  #Train classifier on data with hyperparameters = para

ApplyClassifier(data, model)
  #Compute AUC of model on data
```

Fig. 3: Outline of the experimental evaluation of 22 classifiers over ten NASA MDP datasets.

In general, most statistical classifiers do not require additional model selection and are estimated directly from the training data. This approach has been adopted for LARS, NB, RVM. However, some methods (LDA, QDA and LogReg) suffer from correlations among the attributes and require additional feature selection to produce a valid classification model. Consequently, model selection for these classifiers consist of identifying a suitable set of attributes by means of a backward feature-elimination heuristic [25].

The BayesNet classifier is a directed acyclic graph that represents the joint probability distribution of code attributes and target variable, i.e. each node in the graph represents an attribute and each arc a correlation or dependency. Thus, learning a BayesNet can be considered an optimization problem where a quality measure of the network structure has to be maximized. Therefore, different search techniques (K2, simulated annealing, tabu search, hill climbing, tree augmented naïve Bayes) implemented in the YALE machine learning workbench [45] have been evaluated.

The K* classifier does not require model selection and the number of neighbors has been varied in the range [1,3,5, …, 15] for *k*-NN.

Model selection for neural networks requires defining the number of hidden layers as well as nodes per layer. A single hidden layer of [4,5,…,28] nodes has been considered for MLP networks, whereby each individual architecture is assessed with different weight decay parameters of 0.1 and 0.2 to limit the influence of non-informative features [5]. In addition, a Bayesian learning paradigm towards neural network construction (MLP-2) has been appraised [39]. Finally, the number of cluster centers per class has been varied from 1 to 10 for RBFnet.

The major degrees of freedom of a SVM type model are the kernel function as well as a regularization parameter, commonly denoted by *C.* A radial basis function kernel has been consid-

ered for SVM and LS-SVM, which is the most popular choice in the literature. Consequently, the width of the kernel function and *C* have been tuned by means of a multi-level grid search with exponentially refined parameter grids to achieve a broad coverage of the parameter space as well as an intensive exploration of promising regions [63]. L-SVM is a linear classifier without kernel function and requires tuning of the regularization parameter. A range from $\log(C) = [-6, -5, ..., 20]$ has been evaluated. The LP classifier exhibits no additional parameters and does not require model selection, whereas VP incorporates a polynomial kernel function which degree has to be determined. Values of 1 to 6 have been studied.

Model selection for C4.5 and CART involves deciding upon a pruning strategy. We have considered unpruned trees as well as pruned trees with varying confidence level (0.05, 0.1, …, 0.7); each time with and without Laplacian smoothing [46] and subtree raising. The ADTree classifier is trained by a boosting-based algorithm offering the number of iterations as tuning parameter. Following [21], settings of 10 to 50 iterations have been evaluated.

With respect to ensemble classifiers, LMT generally requires determination of the number of boosting iterations. However, it has been reported that this setting is irrelevant if the final classifier is augmented by pruning [36]. Consequently, we have used the default pruning strategy with an overall number of 100 boosting iterations. Two hyperparameters have been considered for RndFor, namely the number of trees as well as the number of attributes used to grow each individual tree. A range of [10, 50, 100, 250, 500, 1000] trees has been assessed, as well as three different settings for the number of randomly selected attributes per tree $\left(\left[0.5;1,2\right]\cdot\sqrt{M}\right)$, whereby *M* denotes the number of attributes within the respective dataset (see also [7]).

REFERENCES

[1] C. Andersson, "A replicated empirical study of a selection method for software reliability growth models," *Empirical Software Engineering*, 12(2), pp. 161-182, 2007.

[2] C. Andersson and P. Runeson, "A replicated quantitative analysis of fault distributions in complex software systems " *IEEE Transactions on Software Engineering*, 33(5), pp. 273-286, 2007.

[3] B. Baesens, T. Van Gestel, S. Viaene, M. Stepanova, J. Suykens, and J. Vanthienen, "Benchmarking state-of-the-art classification algorithms for credit scoring," *Journal of the Operational Research Society*, 54(6), pp. 627-635, 2003.

[4] V. R. Basili, L. C. Briand, and W. L. Melo, "A validation of object-oriented design metrics as quality indicators," *IEEE Transactions on Software Engineering*, 22(10), pp. 751-761, 1996.

[5] C. M. Bishop, *Neural Networks for Pattern Recognition*. Oxford: Oxford University Press, 1995.

[6] A. P. Bradley, "The use of the area under the ROC curve in the evaluation of machine learning algorithms," *Pattern Recognition*, 30(7), pp. 1145-1159, 1997.

[7] L. Breiman, "Random forests," *Machine Learning*, 45(1), pp. 5-32, 2001.

[8] L. C. Briand, V. R. Basili, and C. J. Hetmanski, "Developing interpretable models with optimized set reduction for identifying high-risk software components," *IEEE Transactions on Software Engineering*, 19(11), pp. 1028-1044, 1993.

[9] L. C. Briand, W. L. Melo, and J. Wüst, "Assessing the applicability of fault-proneness models across object-oriented software projects " *IEEE Transactions on Software Engineering*, 28(7), pp. 706-720, 2002.

[10] M. Chapman, P. Callis, and W. Jackson,"Metrics Data Program," *NASA IV & V Facility*, 2004. Available at http://mdp.ivv.nasa.gov/.

[11] J. G. Cleary and L. E. Trigg, "K*: An Instance-based Learner Using an Entropic Distance Measure," *Proc. of the 12th Intern. Conf. on Machine Learning*, Tahoe City, CA, USA, 1995.

[12] J. Demšar, "Statistical comparisons of classifiers over multiple data sets " *Journal of Machine Learning Research*, 7, pp. 1-30, 2006.

[13] J. Dougherty, R. Kohavi, and M. Sahami:, "Supervised and Unsupervised Discretization of Continuous Features," *Proc. of the 12th Intern. Conf. on Machine Learning*, Tahoe City, CA, USA, 1995.

[14] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, 2 ed. New York: Wiley, 2001.

[15] K. El-Emam, S. Benlarbi, N. Goel, and S. N. Rai, "Comparing case-based reasoning classifiers for predicting high-risk software components," *Journal of Systems and Software*, 55(3), pp. 301-320, 2001.

[16] K. El-Emam, W. Melo, and J. C. Machado, "The prediction of faulty classes using object-oriented design metrics," *Journal of Systems and Software*, 56(1), pp. 63-75, 2001.

[17] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognition Letters*, 27(8), pp. 861-874, 2006.

[18] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth, "From data mining to knowledge discovery in databases: An overview," *AI Magazine*, 17(3), pp. 37-54, 1996.

[19] N. Fenton and M. Neil, "A critique of software defect prediction models," *IEEE Transactions on Software Engineering*, 25(5), pp. 675-689, 1999.

[20] N. E. Fenton and N. Ohlsson, "Quantitative analysis of faults and failures in a complex software system," *IEEE Transactions on Software Engineering*, 26(8), pp. 797-814, 2000.

[21] Y. Freund and L. Mason, "The Alternating Decision Tree Learning Algorithm," *Proc. of the 16th Intern. Conf. on Machine Learning* Bled, Slovenia, 1999.

[22] Y. Freund and R. E. Schapire, "Large margin classification using the perceptron algorithm " *Machine Learning*, 37(3), pp. 277-296, 1999.

[23] K. Ganesan, T. M. Khoshgoftaar, and E. B. Allen, "Case-based software quality prediction," *International Journal of Software Engineering and Knowledge Engineering*, 10(2), pp. 139-152, 2000.

[24] L. Guo, Y. Ma, B. Cukic, and H. Singh, "Robust Prediction of Fault-Proneness by Random Forests," *Proc. of the 15th Intern. Symposium on Software Reliability Engineering*, Saint-Malo, Bretagne, France, 2004.

[25] M. A. Hall and G. Holmes, "Benchmarking attribute selection techniques for discrete class data mining," *IEEE Transactions on Knowledge and Data Engineering* 15(6), pp. 1437-1447, 2003.

[26] M. H. Halstead, *Elements of Software Science*. New York: Elsevier, 1977.

[27] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York: Springer, 2002.

[28] T. M. Khoshgoftaar and E. B. Allen, "Logistic regression modeling of software quality," *International Journal of Reliability, Quality and Safety Engineering*, 6(4), pp. 303-317, 1999.

[29] T. M. Khoshgoftaar, E. B. Allen, J. P. Hudepohl, and S. J. Aud, "Application of neural networks to software quality modeling of a very large telecommunications system," *IEEE Transactions on Neural Networks*, 8(4), pp. 902-909, 1997.

[30] T. M. Khoshgoftaar, E. B. Allen, W. D. Jones, and J. P. Hudepohl, "Classification-tree models of software-quality over multiple releases," *IEEE Transactions on Reliability*, 49(1), pp. 4-11, 2000.

[31] T. M. Khoshgoftaar, A. S. Pandya, and D. L. Lanning, "Application of neural networks for predicting faults," *Annals of Software Engineering*, 1(1), pp. 141-154, 1995.

[32] T. M. Khoshgoftaar and N. Seliya, "Analogy-based practical classification rules for software quality estimation," *Empirical Software Engineering* 8(4), pp. 325-350, 2003.

[33] T. M. Khoshgoftaar and N. Seliya, "Comparative assessment of software quality classification techniques: An empirical case study," *Empirical Software Engineering*, 9(3), pp. 229-257, 2004.

[34] T. M. Khoshgoftaar, N. Seliya, and N. Sundaresh, "An empirical study of predicting software faults with case-based reasoning," *Software Quality Journal*, 14(2), pp. 85-111, 2006.

[35] A. G. Koru and H. Liz, "An investigation of the effect of module size on defect prediction using static measures," *Proc. of the 2005 Workshop on Predictor Models in Software Engineering* St. Louis, Missouri, USA, 2005.

[36] N. Landwehr, M. Hall, and F. Eibe, "Logistic model trees," *Machine Learning*, 59(1), pp. 161-205, 2005.

[37] F. Lanubile and G. Visaggio, "Evaluating predictive quality models derived from software measures: Lessons learned " *Journal of Systems and Software*, 38(3), pp. 225-234, 1997.

[38] J. Li, G. Ruhe, A. Al-Emran, and M. Richter, "A flexible method for software effort estimation by analogy," *Empirical Software Engineering*, 12(1), pp. 65-106, 2007.
[39] D. J. C. MacKay, "The evidence framework applied to classification networks," *Neural Computation*, 4(5), pp. 720-736, 1992.
[40] O. L. Mangasarian and D. R. Musicant, "Lagrangian support vector machines," *Journal of Machine Learning Research*, 1, pp. 161-177, 2001.
[41] T. J. McCabe, "A complexity measure," *IEEE Transactions on Software Engineering*, 2(4), pp. 308-320, 1976.
[42] T. Menzies, A. Dekhtyar, J. Distefano, and J. Greenwald, "Problems with Precision: A Response to Comments on 'Data Mining Static Code Attributes to Learn Defect Predictors " *IEEE Transactions on Software Engineering*, 33(9), pp. 637-640, 2007.
[43] T. Menzies, J. DiStefano, A. Orrego, and R. Chapman, "Assessing Predictors of Software Defects," *Proc. of the 2004 Workshop on Predictive Software Models*, Chicago, USA, 2004.
[44] T. Menzies, J. Greenwald, and A. Frank, "Data mining static code attributes to learn defect predictors," *IEEE Transactions on Software Engineering*, 33(1), pp. 2-13, 2007.
[45] I. Mierswa, M. Wurst, R. Klinkenberg, M. Scholz, and T. Euler, "YALE: Rapid Prototyping for Complex Data Mining Tasks," *Proc. of the 12th ACM SIGKDD Intern. Conf. on Knowledge Discovery and Data Mining*, Philadelphia, PA, USA, 2006.
[46] J. Mingers, "An empirical comparison of pruning methods for decision tree induction," *Machine Learning*, 4(2), pp. 227-243, 1989.
[47] J. C. Munson and T. M. Khoshgoftaar, "The detection of fault-prone programs," *IEEE Transactions on Software Engineering*, 18(5), pp. 423-433, 1992.
[48] I. Myrtveit and E. Stensrud, "A controlled experiment to assess the benefits of estimating with analogy and regression models " *IEEE Transactions on Software Engineering*, 25(4), pp. 510-525, 1999.
[49] I. Myrtveit, E. Stensrud, and M. Shepperd, "Reliability and validity in comparative studies of software prediction models " *IEEE Transactions on Software Engineering*, 31(5), pp. 380-391, 2005.
[50] M. C. Ohlsson and P. Runeson, "Experience from replicating empirical studies on prediction models," *Proc.of the 8th Intern. Software Metrics Symposium*, Ottawa, Canada, 2002.
[51] N. Ohlsson and H. Alberg, "Predicting fault-prone software modules in telephone switches," *IEEE Transactions on Software Engineering* 22(12), pp. 886-894, 1996.
[52] N. Ohlsson, A. C. Eriksson, and M. Helander, "Early risk-management by identification of fault prone modules," *Empirical Software Engineering*, 2(2), pp. 166-173, 1997.
[53] A. A. Porter and R. W. Selby, "Evaluating techniques for generating metric-based classification trees," *Journal of Systems and Software* 12(3), pp. 209-218, 1990.
[54] F. Provost and T. Fawcett, "Robust classification for imprecise environments," *Machine Learning*, 42(3), pp. 203-231, 2001.
[55] J. B. Robert, "A priori tests in repeated measures designs: Effects of nonsphericity," *Psychometrika*, 46(3), pp. 241-255, 1981.
[56] J. Sayyad Shirabad and T. J. Menzies,"The PROMISE Repository of Software Engineering Databases," *School of Information Technology and Engineering, University of Ottawa, Canada*, 2005. Available at http://promise.site.uottawa.ca/SERepository
[57] N. F. Schneidewind, "Methodology for validating software metrics," *IEEE Transactions on Software Engineering*, 18(5), pp. 410-422, 1992.
[58] R. W. Selby and A. A. Porter, "Learning from examples: Generation and evaluation of decision trees for software resource analysis," *IEEE Transactions on Software Engineering*, 14, pp. 1743-1756, 1988.
[59] M. Shepperd and G. Kadoda, "Comparing software prediction techniques using simulation," *IEEE Transactions on Software Engineering*, 27(11), pp. 1014-1022, 2001.
[60] M. Shepperd and C. Schofield, "Estimating software project effort using analogies " *IEEE Transactions on Software Engineering*, 23(11), pp. 736-743, 1997.
[61] J. A. K. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," *Neural Processing Letters*, 9(3), pp. 293-300, 1999.
[62] M. E. Tipping, "The Relevance Vector Machine," in *Advances in Neural Information Processing Systems 12*, S. A. Solla, T. K. Leen, and K.-R. Müller, Eds. Cambridge: MIT Press, 2000, pp. 652–658.
[63] T. Van Gestel, J. A. K. Suykens, B. Baesens, S. Viaene, J. Vanthienen, G. Dedene, B. De Moor, and J. Vandewalle, "Benchmarking least squares support vector machine classifiers," *Machine Learning*, 54(1), pp. 5-32, 2004.
[64] O. Vandecruys, D. Martens, B. Baesens, C. Mues, M. D. Backer, and R. Haesen, "Mining software repositories for comprehensible software fault prediction models " *Journal of Systems and Software*, (doi:10.1016/j.jss.2007.07.034), 2007.
[65] J. H. Zar, *Biostatistical Analysis*, 4th ed. Upper Saddle River: Prentice Hall, 1999.
[66] H. Zhang and X. Zhang, "Comments on "Data mining static code attributes to learn defect predictors" " *IEEE Transactions on Software Engineering*, 33(9), pp. 635-637, 2007.
[67] S. Zhong, T. M. Khoshgoftaar, and N. Seliya, "Analyzing software measurement data with clustering techniques," *IEEE Intelligent Systems*, 19(2), pp. 20-27, 2004.

A Case Study of Core Vector Machines in Corporate Data Mining

Stefan Lessmann, Ning Li, Stefan Voß
Institute of Information Systems, University of Hamburg

Abstract

The core vector machine (CVM) has been introduced as an extremely fast classifier which is demonstrably superior to standard support vector machines (SVMs) on very large datasets. However, only limited information regarding the suitability of CVM for supporting corporate planning is available so far. In this paper, we strive to overcome this deficit. In particular, we consider customer-centric data mining which commonly involves classification in medium-sized settings. CVMs are compared to SVMs within the scope of an empirical benchmarking study to clarify whether previous findings regarding the competitiveness of CVMs generalize to business applications. To that end, representative real-world datasets are employed. In addition, the study aims at scrutinizing the behavior of CVM during model selection. Following a standard grid-search based approach we find some evidence for CVM being more sensitive towards parameter settings than SVMs.

1. Introduction

Data mining has become an important tool to support customer-centric planning tasks in, e.g., response modeling [5, 10], customer attrition analysis [13, 18], credit scoring [23, 28] or fraud detection [16, 30]. Such applications are commonly approached by means of supervised classification and SVMs have proven their suitability for respective decision problems [4, 8, 19, 22].

Recently, Tsang et al. [24, 25] have introduced the CVM as a novel classifier. CVMs possess substantial similarities with traditional SVMs but are more efficient for mining very large datasets. In particular, the quadratic program underlying SVMs is reformulated as a minimum-enclosing-ball problem which solution can be approximated by means of a fast, iterative algorithm. For example, CVMs have been shown to construct a classifier on datasets of up to five million examples and approximately 100 variables within seconds on contemporary hardware without sacrificing predictive accuracy [24]. This is an exceptional result and exceeds the computational capabilities of traditional SVMs by far. However, Tsang et al. [24, 25] demonstrate that the latter can be more efficient on small datasets. Consequently, the current body of knowledge regarding CVMs, e.g. [1, 2, 17, 24, 25, 26, 27], naturally raises question which method to apply in medium sized settings. Contributing towards answering this question from a perspective of corporate data mining is one of the aims of this paper.

Corporate data mining tasks commonly involve datasets of medium size. On the one hand, customer-centric data is collected in almost any business transaction due to extensive usage of information systems. On the other hand, the predominant approach to model customer behavior involves mapping one customer to one example, i.e. one record in the dataset to be mined. Therefore, the size of such datasets is naturally bounded by the number of a company's customers. In addition, the supervised learning paradigm imposes further constraints on the availability of useable training data by requiring detailed label information, i.e., a specific value for the dependant variable for each customer.

Therefore, the paper strives to appraise CVMs potential for customer-centric classification tasks. In particular, we conduct an empirical experiment to contrast CVMs and SVMs (as reference model) on representative datasets. Amending traditional measures of comparison like predictive accuracy and computational efficiency, a classifier's sensitivity towards parameter settings is considered as an additional quality indicator.

Both methods exhibit the same free parameters and thus require model selection techniques to determine suitable values. This task is predominantly approached by means of empirical procedures that repetitively evaluate different

1530-1605/08 $25.00 © 2008 IEEE

candidate values. Consequently, parameter sensitivity increases the number of evaluation and thereby the overall training time of the classifier. Furthermore, higher sensitivity elevates the risk of selecting a suboptimal setting which produces inferior out-of-sample accuracy. A standard argument within the corporate data mining community is that small deviations in predictive accuracy can have substantial financial consequences [4, 5, 8]. Therefore, a more robust method might be preferable despite computational inferiority.

To appraise the parameter sensitivity of CVM and SVM, we propose a worst-case analysis as well as an analysis based on the fourth statistical moment of the respective performance distributions. Following this approach, the model selection results presented here provide some evidence for CVMs being more sensitive towards parameter settings than SVMs. In other words, the latter may be considered appropriate even if constructing a single classifier on a given dataset turns out to be more time consuming than conducting the same task with the CVM.

The paper is organized as follows. We briefly review the basics of SVMs in Section 2 before discussing the reformulation considered in CVMs. Sections 3 and 4 present the empirical results of the benchmarking study, and conclusions are drawn in Section 5.

2. Classification algorithms

In the sequel, we review the theory of SVM- and CVM-based classification. Formally, the task of classification can be stated as follows: Let S be a dataset containing M examples, $S = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{M}$, where $\boldsymbol{x}_i \in R^N$ denotes an input vector and y_i its corresponding discrete class label. The goal of classification is to infer a predictive model, i.e. a classifier, $y(\boldsymbol{x})$ from S, which accurately predicts the class membership of novel examples. Here, we consider the case of binary classification where $y_i \in \{-1, +1\}$.

2.1. Support vector machines

SVMs can be characterized as linear classifiers. That is, predictions are based on a separation of the data by means of a linear hyperplane (1), with normal $\boldsymbol{w}$ and intercept b:

$$y(\boldsymbol{x}) = sign\left(\boldsymbol{w} \cdot \boldsymbol{x} + b\right). \tag{1}$$

For SVMs, the parameters, $\boldsymbol{w}$ and b, are determined by means of mathematical programming. Thus, the construction of the classifier, commonly referred to as classifier training, corresponds to solving the convex program (2) to optimality, whereby ξ_i represents a slack variable which is greater than zero only if a training example $\boldsymbol{x}_i \in S$ is misclassified.

$$\begin{aligned} &\min_{\boldsymbol{w},b,\xi} \frac{1}{2}\boldsymbol{w} \cdot \boldsymbol{w} + C\sum_{i=1}^{M}\xi_i \\ &s.t.:\ y_i((\boldsymbol{w} \cdot \boldsymbol{x}_i) + b) + \xi_i \geq 1,\ \ i = 1,...,M. \end{aligned} \tag{2}$$

Program (2) is inspired by statistical learning theory [29] and minimizes the sum of two terms which measure the distance between the examples of opposite classes which are closets to the hyperplane defined by $\boldsymbol{w}$ and b, referred to as the margin of separation, and the number of misclassifications, respectively.

The margin can be related to the model's capability of producing predictions that generalize to future data. Roughly speaking, SVMs strive to discriminate the training data accurately, i.e. without error, with a model as simple as possible, i.e. a model with large margin; see [9] for details. The parameter C allows controlling the trade-off between these two conflicting goals and has to be specified by the user prior to classifier training. Subsequently, we refer to C as the penalty parameter.

A mapping function is employed to produce more complex, nonlinear classifiers. That is, the classification model (3) is considered instead of (1), whereby φ is a nonlinear mapping that projects $\boldsymbol{x}$ into a higher dimensional feature space.

$$y(\boldsymbol{x}) = sign\left(\boldsymbol{w} \cdot \varphi(\boldsymbol{x}) + b\right). \tag{3}$$

By constructing the linear classifier in this nonlinearly transformed space, a nonlinear separation of the data in the input space R^N is obtained. Due to the structure of SVMs, it is not necessary to explicitly compute this transformation. Consider the dual of (2) and let α_i denote the Lagrangian multipliers:

$$\begin{aligned} &\max_{\boldsymbol{\alpha}} = \sum_{i=1}^{M}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{M}\alpha_i\alpha_j y_i y_j (\boldsymbol{x}_i \cdot \boldsymbol{x}_j) \\ &s.t. \quad \sum_{i=1}^{M}\alpha_i y_i = 0 \quad 0 \leq \alpha_i \leq C \quad \forall i. \end{aligned} \tag{4}$$

As the input data enters the dual (4) only in form of scalar products, a so called kernel func-

tion K (5) can be employed to compute the scalar products of the transformed vectors directly:

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \varphi(\boldsymbol{x}_i) \cdot \varphi(\boldsymbol{x}_j). \quad (5)$$

Thus, the final SVM classifier is given as:

$$y(\boldsymbol{x}) = sign\left(\sum_{i \in SV} \alpha_i y_i K(\boldsymbol{x}_i, \boldsymbol{x}) + b \right), \quad (6)$$

where SV represents the set of support vectors, i.e examples with non-zero Lagrangian multipliers.

The kernel (5) defines a measure of proximity between examples in the transformed feature space. Integration of a kernel into (4) is straightforward and does not affect the overall algorithm. This may be considered a particular merit of the SVM classifier which leads to increased flexibility, e.g. by developing special purpose kernels for text or multi-media mining tasks or incorporating prior knowledge. However, the radial basis function (RBF) kernel (7) is most popular in practical applications of corporate data mining and has been shown to posses some desirable properties [15]. Therefore, this kernel is used later in the benchmarking experiment.

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\gamma \left\| \boldsymbol{x}_i - \boldsymbol{x}_j \right\|^2 \right). \quad (7)$$

2.2. Core vector machines

CVMs have initially been proposed in [24, 25]. Extensions to the task of support vector regression and classification with class-dependant penalties are proposed in [26, 27]. Furthermore, CVMs have been considered in conjunction with clustering algorithms [1] and multi-class classification [2]. A classifier closely related to the CVM has also been proposed in [17].

As solving (4) involves quadratic programming, SVM learning may become infeasible in large-scale settings when datasets comprise several hundred thousand examples. Observing that practical algorithms for SVM learning, e.g. [20], do not solve (4) to optimality but impose a tolerance parameter on the Karush-Kuhn-Tucker conditions, Tsang et al. [25] propose to reformulate (4) as an equivalent minimum-enclosing-ball (MEB) problem which solution can be approximated by means of a fast iterative algorithm using the concept of core sets [24].

Given a set of points, e.g. $\boldsymbol{x}_i \in S$, the MEB is defined as the smallest ball which contains all points. Let r denote the radius and c the center of a ball, the problem of finding an MEB in the feature space can be stated as follows (8):

$$\begin{aligned} &\min_{r,\boldsymbol{c}} \; r^2 \\ &s.t. \quad \left\| \varphi(\boldsymbol{x}_i) - \boldsymbol{c} \right\| \le r^2 \quad \forall i = 1,...,M. \end{aligned} \quad (8)$$

The corresponding dual is given as:

$$\begin{aligned} &\max_{\boldsymbol{\alpha}} \; \sum_{i,j=1}^{M} \alpha_i \alpha_j K(\boldsymbol{x}_i, \boldsymbol{x}_j) \\ &\quad - \sum_{i=1}^{M} \alpha_i K(\boldsymbol{x}_i, \boldsymbol{x}_i) \\ &s.t. \quad \sum_{i=1}^{M} \alpha_i = 1 \quad \alpha_i \ge 0 \quad \forall i = 1,...,M, \end{aligned} \quad (9)$$

If:

$$K(\boldsymbol{x}_i, \boldsymbol{x}_i) = \kappa, \text{ a constant}, \quad (10)$$

one may discard the second term in the objective to obtain the final MEB problem (11).

$$\begin{aligned} &\max_{\boldsymbol{\alpha}} \; \sum_{i,j=1}^{M} \alpha_i \alpha_j K(\boldsymbol{x}_i, \boldsymbol{x}_j) \\ &s.t. \quad \sum_{i=1}^{M} \alpha_i = 1 \quad \alpha_i \ge 0 \quad \forall i = 1,...,M, \end{aligned} \quad (11)$$

Note that (10) holds for many practical kernel functions, including the RBF kernel. However, a generalization of the CVM [27] does not require this constraint anymore and enables arbitrary kernels.

As is shown in [2, 24, 25], a slight modification of the original SVM program (2) yields a dual similar to (11). In particular, considering the L2-norm of the slack variable, in other words using a squared-error loss function, produces the dual (12), with δ_{ij} being the Kronecker delta:

$$\begin{aligned} &\max_{\boldsymbol{\alpha}} = \sum_{i,j=1}^{M} \alpha_i \alpha_j \left(y_i y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j) + y_i y_j + \frac{\delta_{ij}}{C} \right) \\ &s.t. \quad \sum_{i=1}^{M} \alpha_i = 1 \qquad \alpha_i \ge 0 \qquad \forall i. \end{aligned} \quad (12)$$

Now, to obtain (11), set:

$$\tilde{K}(\boldsymbol{x}_i, \boldsymbol{x}_j) = y_i y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j) + y_i y_j + \frac{\delta_{ij}}{C} \quad (13)$$

Hence, redefining the kernel by (13) allows formulating the SVM with L2-loss as a MEB problem.

The computational advantage of solving the MEB problem with an approximation algorithm

stems from the concept of core sets. Given a set of points $\boldsymbol{x}_i \in S$, a subset $Q \subseteq S$ is a core set of S if an expansion by a factor $(1+\varepsilon)$ of its MEB contains S [24], where ε is a small positive number. Tsang et al. [25] employ the algorithm of Bădoiu and Clarkson [3] to obtain such an ε-approximation of (11): Let $B_t(\boldsymbol{c}_t, r_t)$ denote the MEB of the core set Q at iteration t. Then, the algorithm adds to Q the furthest point outside the ball $B(\boldsymbol{c}_t, (1+\varepsilon)r_t)$. This step is repeated until all points in S are covered by $B(\boldsymbol{c}_t, (1+\varepsilon)r_t))$; see [24] for details.

CVMs efficiency on large datasets can be attributed to the fact that the size of the final core set depends only on ε but not on M or N [25].

The calculation of class predictions using the CVM differs from (6) only in the sense that the modified kernel $\tilde{K}$ is considered instead of K which also encodes label information y_i; namely:

$$y(\boldsymbol{x}) = sign\left(\sum_{i \in Q} \alpha_i \tilde{K}(\boldsymbol{x}_i, \boldsymbol{x}) + b\right). \qquad (14)$$

Note that we can always remove the sign-function in (1), (3), (6) and (14) to obtain a continuous prediction which represents the confidence of the classifier [29].

3. Comparing the computational efficiency of CVM versus SVM

One motivation for evaluating the CVM as a candidate technique for business classification is its remarkable computational efficiency. Therefore, we begin the empirical evaluation with a small runtime comparison of CVMs versus SVMs to replicate the results of [24, 25] in a setting of corporate data mining. In particular, we consider a case of direct marketing in the publishing industry. The respective dataset represents a marketing campaign aiming at cross-selling an additional magazine subscription to customers of the publisher. Each customer is characterized by 95 numerical as well as categorical attributes and 300,000 examples are given. The binary target variable indicates if a contacted customer has responded to the campaign, i.e. subscribed to one or more periodicals; see [10] for details.

Increasing numbers of examples are randomly sampled to scrutinize the evolution of training times. The LibSVM library [7] is employed as SVM implementation and both classifiers are configured with their respective default parameters. Table 1 depicts the results of this comparison in terms of training time and number of identified core vectors and support vectors, respectively.

Table 1: Efficiency comparison of CVM versus SVM

	Training set size in 1000 examples				
	60	120	180	240	300
	Runtime in sec.				
CVM	52	60	78	85	96
SVM	292	1,736	4,910	7,858	14,101
	Number of core/support vectors				
CVM	1,150	1,368	1,612	1,735	1,793
SVM	2,263	4,179	6,319	8,226	10,139

The results confirm previous findings [24, 25] and further emphasize CVMs efficiency on large datasets. Besides significantly lower training times, the number of core vectors is considerably smaller than the respective figure for SVM. Therefore, CVMs are significantly faster at classifying novel examples than SVMs for this task; see also (14) and (6), respectively.

4. Comparing predictive accuracy and parameter sensitivity of CVM versus SVM

4.1. Experimental setup

The previous results demonstrate that 60,000 training examples may suffice to give CVMs a computational advantage over SVMs. Therefore, subsequent experiments consider smaller datasets to enhance our understanding when to use which classifier. To that end, we employ four datasets from the Data Mining Cup, which is an annual competition organized by prudsys AG [21]. The considered data stems from the years 2000 to 2002 as well as 2005 (DMC00, DMC01, DMC02, DMC05) and represent classification tasks in direct marketing, churn prediction and fraud detection; see [19] for details. We deem these datasets representative for the domain considered here and summarize their characteristics in Table 2.

Table 2. Dataset characteristics

	DMC00	DMC01	DMC02	DMC05
#Train	10,000	10,000	10,000	30,000
#Test	28,890	18,128	10,000	20,000
%Pos	5.7%	50%	10%	5.9%
%MV	5.6%	22.6%	24%	84%
#CA	24	9	13	84
#MA	19	24	19	8

#Train/#Test: the number of records used during building/evaluating the classification model.
%Pos/%MV: the percentage of class 1 records and records that contain at least one missing value, respectively.
#CA/#MA: the given number of categorical and numerical attributes within the datasets.

The partitioning of examples into training/test records is taken from the particular challenge. With respect to the study's focus on predictive performance, standard pre-processing techniques are utilized; e.g., mean replacement of missing values, normalization to zero mean and standard deviation of numerical variables as well as dummy-variable-based encoding of categories; see, e.g., [10].

A model's predictive performance is measured by means of the area under a receiver-operating-characteristics-curve (AUC) [6]. The AUC is a general indicator of predictiveness and is selected because of its robustness towards imbalanced class distributions. Class imbalance is present in DMC01, DMC03 as well as DMC05, and may be considered characteristic for most customer-centric decision problems. In particular, AUC appraises the ranking capabilities of a model, i.e. the probability that a classifier ranks a randomly selected positive example higher than a randomly selected negative one and is thus equivalent to the Wilcoxon test of ranks [11].

The tolerance parameter ε is not considered in this study but left on its default setting for the CVM and the SVM. This leaves two free parameters that have to be specified prior to applying a CVM and a SVM classifier, respectively. These are the penalty parameter, C, as well as the width of the RBF kernel function, γ. We organize parameter determination as a grid-search over candidate values of $\log_2(C)$ = [-5,-3,-1,…, 15] and $\log_2(\gamma)$ = [-15,-13,-11,…,1]; e.g. [12]. Each of the resulting 99 parameter combinations is evaluated by means of 10-fold cross-validation on the training set to estimate the predictive power of the resulting classifier. The best setting is retained and a respective classifier is constructed on the whole training set to predict the test set.

4.2. Results of the model selection stage

The primary objective of analyzing the detailed grid-search results is to appraise the risk of model misspecification when applying the novel CVM classifier. As the likelihood of selecting suboptimal parameter values increases with the classifier's parameter sensitivity, i.e. the variation in predictive accuracy induced by different parameter settings, we utilize the latter as a proxy of misspecification risk.

As a first step towards a deeper understanding of CVMs' behavior during model selection, we consider a worst-case perspective. In particular, we may ask how the worst possible CVM model, with respect to the abovementioned parameter grid, compares to the worst SVM model. This idea is implemented by drawing the sorted cross-validation based AUC estimates over all parameter combinations for CVM and SVM (Figure 1). Hence, the abscissa of Figure 1 gives the rank value of a particular parameter setting, whereby the best setting obtains the highest rank.

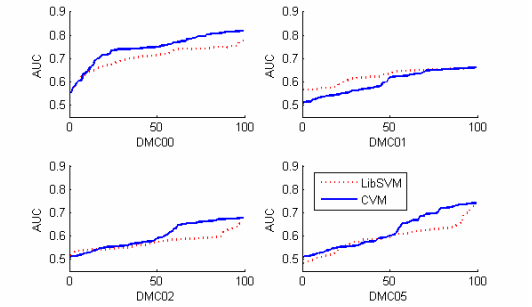


Figure 1. Ordered AUC for CVM and SVM per parameter setting

Figure 1 suggests that CVMs compare favorably to SVMs in the sense that an AUC estimate with given rank is commonly at the same level, or above, a respective SVM result. The situation is different on DMC01 where the ~70% least good parameter settings produce a lower AUC as in the SVM case. However, with respect to the ultimate goal of model selection, i.e. identifying promising parameter values, special consideration should be given to the top ranked parameters. Consequently, we may conclude that CVM is at least not inferior to SVM on the datasets employed here.

The best parameter values per dataset are reported in Table 3.

Table 3. Optimal parameter values per dataset and classifier by means of 10-fold cross-validation AUC

	DMC00	DMC01	DMC02	DMC05
	CVM classifier			
$\log_2(C)$	-1	5	1	-1
$\log_2(\gamma)$	-5	-11	-9	-7
AUC	0.82	0.66	0.68	0.74
	SVM classifier			
$\log_2(C)$	13	13	15	13
$\log_2(\gamma)$	-15	-15	-15	-15
AUC	0.78	0.66	0.66	0.73

Noteworthy, the parameters selected by SVM are more consistent and identical on three datasets. Considering CVM, higher variation of the penalty parameter C could be attributed to the fact that a L2-loss function is considered which might be more sensitive to outliers, see Section 2.2, whereas higher variation of γ is yet unexplained.

To gain further insight into CVMs' parameter sensitivity, Figure 2 depicts the distribution of AUC-values over the 99 parameter combinations per cross-validation fold and classifier across all datasets by means of box-plots. Datasets are ordered consecutively starting with DMC00 (first row).

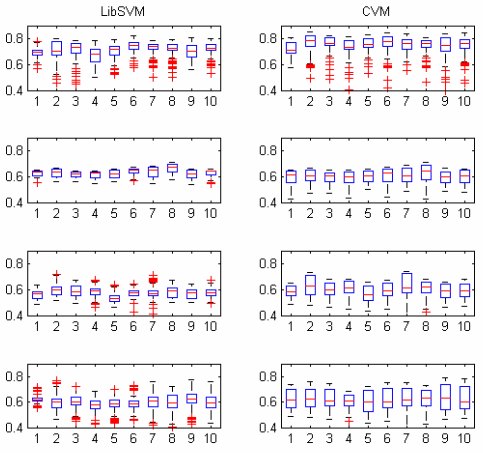


Figure 2. Distribution of AUC values per classifier and cross-validation partition

Clearly, both classifiers exhibit considerable variation which illustrates their parameter sensitivity and demonstrates the importance of model selection in general. For example, the median AUC value on DMC00 is 0.75 for CVM with a maximal (minimal) value of 0.81 (0.55) and respective figures of 0.71, 0.78 and 0.56 for SVM. Considering the results on the last three datasets, i.e. larger box height, one may speculate if CVMs' parameter sensitivity exceeds those of SVM. To further scrutinize this suspicion, we consider the fourth moment of the AUC distributions, the kurtosis, as depicted in Table 4.

Table 4. Kurtosis of AUC distributions per dataset and classifier

	DMC00	DMC01	DMC02	DMC05
CVM	1.155	-1.451	-1.525	-1.488
SVM	0.771	-0.956	-0.011	-0.108

The kurtosis is a measure of "peakedness" (or "tailedness") of a distribution compared to the normal distribution which has a kurtosis of zero [31]. The fact that CVMs' AUC distributions have smaller kurtosis on three out of four datasets indicates that the same parameter settings produce performance distributions with smaller peak and thus larger dispersion. This may be seen as evidence for the initial suspicion of CVM being more sensitive towards parameter settings.

A more elaborate approach, yet leading to the same conclusion, is as follows: Model selection strives to identify accurate models. Therefore, we may discard inappropriate parameter settings and focus on analyzing good results. This is done in Figure 3, which illustrates the development of kurtosis, when only the top N-percent of parameter combinations are considered. That is, we extract the 50%, 40%, etc. best parameter settings for both classifiers and compute the kurtosis for the resulting AUC distributions.

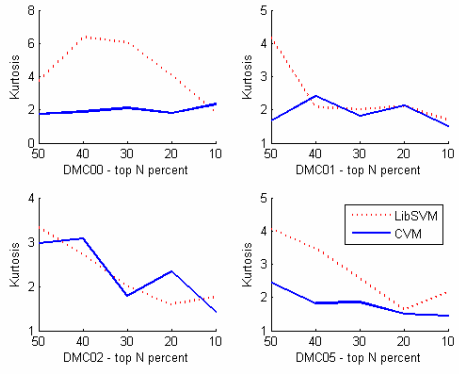


Figure 3. Kurtosis of upper percentiles of the AUC distribution across classifiers and datasets

The SVM shows demonstrably higher kurtosis on DMC00 and DMC05, whereas this pattern can be observed on DMC01 only up to the 40% best parameter settings. Mixed results are obtained on DMC02. Overall, the results provide some empirical evidence for CVMs' increased parameter sensitivity.

Note that the scope of the empirical evaluation requires distributing computations across multiple workstations with varying hardware configurations. Consequently, we refrain from presenting detailed runtimes for the model selection stage. However, we are able to report that CVM model selection consumes significantly more time than conducting the respective task for LibSVM for the considered datasets. For example, processing the 990 SVM models (99 parameter combinations * 10-fold cross validation) for DMC00 takes 93,778 sec., whereas CVM requires 496,098 sec. (The same hardware has been used for these two experiments, i.e. a Windows XP PC with 1.75Ghz CPU and 1GB RAM. As indicated by Tsang et al. [24, 25], SVM is more efficient for small sized problems because of sophisticated heuristics to speed-up classifier training. In addition, CVM tends to select a larger number of core vectors in such settings which, in turn, increases the time for classifier evaluation; e.g. the final SVM classifier for DMC00 includes 1,204 support vectors, whereas the CVM model comprises 6,880 core vectors. A similar yet less extreme pattern could be observed on DMC05. Combining this observation with the results of Table 1, it can be assumed that the CVM requires at least 40,000 to 50,000 examples to offer a computational advantage over the SVM.

4.3. Hold-out set results

Finally, Table 5 concludes the empirical comparison and depicts the predictive accuracy of the final CVM and SVM classification model on training and hold-out testing data. The results are roughly at the same level across both classifiers, with CVM giving slightly higher accuracy on DMC00. This confirms previous results of Tsang et al. [24, 25] regarding the competitive performance of CVM and demonstrates that they generalize to the datasets considered here. Furthermore, the overall experience with the CVM classifier in this study, as well as in previous experiments [1, 2, 24, 25, 27], further secures the initial conclusion of [24], "*that it* [CVM] *is as accurate as existing SVM implementations*" in terms of hold-out test set performance. However, due to amplified parameter sensitivity, model selection results might be less stable, leading to an increased risk of model misspecification. Although no case of predictive inferiority has been observed so far, this issue should be kept in mind before applying the CVM classifier.

Table 5. Training and test set results of the final classifiers by means of AUC

	DMC00	DMC01	DMC02	DMC05
	Final CVM classifier			
Train	0,82	0,66	0,68	0,74
Test	0,82	0,66	0,67	0,59
#CV	6,880	9,569	9,649	24,679
	Final SVM classifier			
Train	0,78	0,66	0,66	0,73
Test	0,79	0,66	0,66	0,59
#SV	1,204	8,264	2,037	3,591

5. Conclusions

Following an empirical research paradigm, we have evaluated a novel classification model, the CVM classifier, as a tool for corporate data mining. Our experiments replicate previous findings regarding the potential of CVMs and demonstrate that it is a promising approach for large-scale business classification tasks.

It is well known that the classification performance of a SVM model heavily depends upon a suitable selection of parameter values. Analyzing CVMs model selection behavior we have found some evidence for CVM being even more sensitive towards parameterization than SVM. In particular, parameter-induced performance variability of CVM exceeds that of a SVM classifier. Consequently, results of model selection might be less stable, leading to an increased risk of model-misspecification. However, no respective case has been observed empirically so far. On the contrary, we could replicate previous findings of CVM being at least competitive to SVM in terms of hold-out test set performance. Therefore, the question how severe practical applications are affected by slightly higher parameter variability requires further research. On the one hand, parameter sensitivity is not problematic as long as the employed model selection procedure, e.g. grid-search, selects "the right" configuration, i.e. parameter values that yield accurate hold-out predictions. On the other hand, training data used during model selection is always just a sample

and might give a biased picture of the stochastic process which has generated the data in the first place. In this sense, (higher) parameter dependency is undesirable. Furthermore, higher variability requires more extensive model selection, i.e. evaluating more parameter combinations, thereby decreasing CVMs computational advantage to some extend.

In this sense, we may conclude that CVM amend SVM and offer a capable alternative when the volume of the data to be processed prohibits application of the later. This is also evident from the fact that CVM can be considerably slower than SVM on smaller sized datasets [24, 25]. In medium-sized settings, users have to decide between both techniques. Our results suggest that the time for constructing a single classifier is a misleading indicator in such settings. Even if the size of the respective dataset suffices to give CVM a computational advantage over SVM, the former might still require a larger number of parameter evaluation to arrive at the same level of stability. Conversely, SVM facilitates using a coarser parameter grid and thereby regain some efficiency compared to CVM.

However, this does not depreciate the remarkable potential of CVMs. They enable classification in scenarios where the SVM can no longer be applied directly. One may object that it is not necessary to utilize all available data in large-scale settings but could employ SVMs in conjunction with sampling procedures. While true, we emphasize that each additional component, e.g. a sampling algorithm, adds to the overall complexity of the data mining process and thereby hinders a wider adoption in corporate practice.

As classification performance depends so heavily upon appropriate parameter values, the development of more sophisticated model selection procedures seems a promising field for future research. Substantial achievements have been made in the SVM community, e.g. by using gradient-based techniques [14]. On the other hand, this is the first study that considers CVM model selection in some detail. Gradient-based optimization of free parameters might be an option if they scale up to very large datasets where CVM unfold their full potential. Considering the approximate nature of the approach tuning heuristics like evolutionary algorithms appear to be another capable direction for future research.

Acknowledgments

The authors would like to express their gratitude to Ivor W. Tsang, James T. Kwok and Pak-Ming Cheung for making available the CVM executables. In particular we are grateful to James T. Kwok for continuous assistance and providing several valuable comments.

References

[1] S. Asharaf, M. N. Murty, and S. K. Shevade, "Cluster Based Core Vector Machine," *Proc. of the 6th IEEE Intern. Conf. on Data Mining*, Hong Kong, China, 2007, pp. 1038-1042.

[2] S. Asharaf, M. N. Murty, and S. K. Shevade, "Multiclass Core Vector Machine," *Proc. of the 24th Intern. Conf. on Machine Learning*, Corvallis, OR, USA, 2007 (to appear).

[3] M. Bădoiu and K. L. Clarkson, "Optimal Core Sets for Balls," in *Proc. of the DIMACS Workshop on Computational Geometry*. Piscataway, NJ, USA, 2002 (http://cm.bell-labs.com/who/clarkson/coresets1.pdf).

[4] B. Baesens, T. Van Gestel, S. Viaene, M. Stepanova, J. Suykens, and J. Vanthienen, "Benchmarking state-of-the-art classification algorithms for credit scoring," *Journal of the Operational Research Society*, 54(6), pp. 627-635, 2003.

[5] B. Baesens, S. Viaene, D. Van den Poel, J. Vanthienen, and G. Dedene, "Bayesian neural network learning for repeat purchase modelling in direct marketing," *European Journal of Operational Research*, 138(1), pp. 191-211, 2002.

[6] A. P. Bradley, "The use of the area under the ROC curve in the evaluation of machine learning algorithms," *Pattern Recognition*, 30(7), pp. 1145-1159, 1997.

[7] C.-C. Chang and C.-J. Lin, "LIBSVM - A Library for Support Vector Machines," 2001. (www.csie.ntu.edu.tw/~cjlin/libsvm)

[8] K. Coussement and D. Van den Poel, "Churn prediction in subscription services: An application of support vector machines while comparing two parameter-selection techniques," *Expert Systems with Applications*, 34(1), pp. 313-327, 2008.

[9] N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines and other Kernel-based Learning Methods*. Cambridge: Cambridge University Press, 2000.

[10] S. F. Crone, S. Lessmann, and R. Stahlbock, "The impact of preprocessing on data mining: An evaluation of classifier sensitivity in direct marketing," *European Journal of Operational Research*, 173(3), pp. 781-800, 2006.

[11] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognition Letters*, 27(8), pp. 861-874, 2006.

[12] C.-W. Hsu, C.-C. Chang, and C.-J. Lin, "A practical guide to support vector classification," Working paper, *Department of Computer Science and Information Engineering, National Taiwan University*, 2003.

[13] Y. Hur and S. Lim, "Customer Churning Prediction Using Support Vector Machines in Online Auto Insurance Service," *Proc. of the 2nd Intern. Symposium on Neural Networks*, Chongqing, China, 2005, pp. 928-933.

[14] S. Keerthi, V. Sindhwani, and O. Chapelle, "An Efficient Method for Gradient-Based Adaptation of Hyperparameters in SVM Models," in *Advances in Neural Information Processing Systems 19*, B. Schölkopf, J. C. Platt, and T. Hoffman, Eds. Cambridge: MIT Press, 2007, pp. 217-224.

[15] S. S. Keerthi and C.-J. Lin, "Asymptotic behaviors of support vector machines with Gaussian kernel," *Neural Computation*, 15(7), pp. 1667-1689, 2003.

[16] E. Kirkos, C. Spathis, and Y. Manolopoulos, "Data mining techniques for the detection of fraudulent financial statements," *Expert Systems with Applications*, 32(4), pp. 995-1003, 2007.

[17] P. Kumar, J. S. B. Mitchell, and E. A. Yildirim, "Approximate minimum enclosing balls in high dimensions using core-sets," *ACM Journal of Experimental Algorithmics*, 8, 2003. (http://doi.acm.org/10.1145/996546.996548).

[18] B. Lariviere and D. Van den Poel, "Predicting customer retention and profitability by using random forests and regression forests techniques," *Expert Systems with Applications*, 29(2), pp. 472-484, 2005.

[19] S. Lessmann and S. Voß, "A framework for customer-centric data mining with support vector machines," Working paper, *Institute of Information Systems, University of Hamburg*, 2007.

[20] J. C. Platt, "Fast Training of Support Vector Machines using Sequential Minimal Optimization," in *Advances in Kernel Methods: Support Vector Learning*, B. Schölkopf, C. J. C. Burges, and A. J. Smola, Eds. Cambridge: MIT Press, 1999, pp. 185-208.

[21] Prudsys, "The Data Mining Cup," 2007. (www.data-mining-cup.com)

[22] H. Shin and S. Cho, "Response modeling with support vector machines," *Expert Systems with Applications*, 30(4), pp. 746-760, 2006.

[23] L. C. Thomas, R. Oliver, and D. J. Hand, "A survey of the issues in consumer credit modelling research," *Journal of the Operational Research Society*, 56(9), pp. 1006-1015, 2005.

[24] I. W. Tsang, J. T. Kwok, and P.-M. Cheung, "Core vector machines: Fast SVM training on very large data sets" *Journal of Machine Learning Research*, 6, pp. 363-392, 2005.

[25] I. W. Tsang, J. T. Kwok, and P.-M. Cheung, "Very Large SVM Training Using Core Vector Machines," *Proc. of the 10th Intern. Workshop on Artificial Intelligence and Statistics*, Barbados, 2005, pp. 349-356.

[26] I. W. Tsang, J. T. Kwok, and K. T. Lai, "Core Vector Regression for Very Large Regression Problems," *Proc. of the 22nd Intern. Conf. on Machine learning* Bonn, Germany, 2005, pp. 912-919.

[27] I. W. H. Tsang, J. T. Y. Kwok, and J. M. Zurada, "Generalized core vector machines," *IEEE Transactions on Neural Networks*, 17(5), pp. 1126-1140, 2006.

[28] T. Van Gestel, B. Baesens, J. A. K. Suykens, D. Van den Poel, D.-E. Baestaens, and M. Willekens, "Bayesian kernel based classification for financial distress detection," *European Journal of Operational Research*, 172(3), pp. 979-1003, 2006.

[29] V. N. Vapnik, *The Nature of Statistical Learning Theory*. New York: Springer, 1995.

[30] S. Viaene, R. A. Derrig, B. Baesens, and G. Dedene, "A comparison of state-of-the-art classification techniques for expert automobile insurance claim fraud detection," *Journal of Risk & Insurance*, 69(3), pp. 373-421, 2002.

[31] J. H. Zar, *Biostatistical Analysis*, 4th ed. Upper Saddle River: Prentice Hall, 1999.

2006 International Joint Conference on Neural Networks
Sheraton Vancouver Wall Centre Hotel, Vancouver, BC, Canada
July 16-21, 2006

An Evaluation of Discrete Support Vector Machines for Cost-Sensitive Learning

Stefan Lessmann, Sven F. Crone, Robert Stahlbock, Nikolaus Zacher

Abstract— **The problem of cost-sensitive learning involves classification analysis in scenarios where different error types are associated with asymmetric misclassification costs. Business applications and problems of medical diagnosis are prominent examples and pattern recognitions techniques are routinely used to support decision making within these fields. In particular, support vector machines (SVMs) have been successfully applied, e.g. to evaluate customer credit worthiness in credit scoring or detect tumorous cells in bio-molecular data analysis. However, ordinary SVMs minimize a continuous approximation for the classification error giving similar importance to each error type. While several modifications have been proposed to make SVMs cost-sensitive the impact of the approximate error measurement is normally not considered. Recently, Orsenigo and Vercellis introduced a discrete SVM (DSVM) formulation [1] that minimize misclassification errors directly and overcomes possible limitations of an error proxy. For example, DSVM facilitates explicit cost minimization so that this technique is a promising candidate for cost-sensitive learning. Consequently, we compare DSVM with a standard procedure for cost-sensitive SVMs and investigate to what extent improvements in terms of misclassification costs are achievable. While the standard SVM performs remarkably well DSVM is found to give yet superior results.**

I. Introduction

THE support of decision making by means of classification analysis has received considerable attention in research and practice. Classification involves the prediction of a discrete class membership on the basis of observable/measurable attributes. For example, the task of credit scoring [2] consists of estimating whether a customer is credit worthy or not using attributes like the applicants age, income, occupation etc. It is generally believed that the costs of granting credit to a bad risk, e.g. a defaulting customer, is significantly greater than the cost of denying credit to a good risk candidate [3]. The same problem arises in medical diagnosis where a false alarm is usually not as severe as a missed correct alarm; e.g. missing a positive result when detecting tumors from magnetic resonance imaging scans might induce dramatic consequences while a small number of false alarms is tolerable if the scans are subsequently re-screened by medical personal; e.g. [4].

The SVM, introduced by Vapnik an co-workers [5, 6], is a state-of-the-art classification algorithm that has been used successfully to support managerial and medical decision making; e.g. [7-10]. SVM training involves the minimization of a continuous approximation of the classification error giving similar importance to each error type. While the problem of using SVM for cost-sensitive classification has received some attention in the literature, [4, 11-14], the impact of the error approximation is usually not considered.

Recently, Orsenigo and Vercellis proposed a discrete SVM formulation that minimizes misclassification errors in a more intuitive manner. Emphasizing on classification errors and facilitating an explicit cost-minimization this approach is a promising candidate for cost-sensitive learning. Consequently, we compare DSVM with a standard procedure for cost-sensitive SVMs and evaluate to what extent improvements in terms of misclassification costs are achievable.

The remainder of the paper is organized as follows. A brief introduction to SVMs is given in Section II before we review previous work on cost-sensitive SVMs in Section III. We present the DSVM formulation in Section IV and describe a tabu search (TS) heuristic to solve the resulting optimization problem. The results of our empirical comparisons with the standard SVM can be found in Section V. Conclusions are given in Section VI.

II. Support Vector Machines

The original SVM can be characterized as a supervised learning algorithm capable of solving linear and non-linear binary classification problems. Given a training set with m patterns $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^{m}$, where $\boldsymbol{x}_i \in \mathrm{X} \subseteq \Re^n$ is an input vector and $y_i \in \{-1,+1\}$ its corresponding binary class label, the idea of support vector classification is to separate examples by means of a maximal margin hyperplane [15]. That is, the algorithm strives to maximize the distance between examples that are closest to the decision surface. It has been shown that maximizing the margin of separation improves the generalization ability of the resulting classifier [6]. To construct such a classifier one has to minimize the norm of the weight vector $\boldsymbol{w}$ under the constraint that the training patterns of each class reside on opposite sides of the

S. Lessmann (corresponding author), R. Stahlbock, N. Zacher, Institute of Information Systems, University of Hamburg, Von-Melle-Park 5, D-20146 Hamburg, Germany (phone: 0049-40-42838-5500; fax: 0049-40-42838-5535; e-mail: [lessmann, stahlbock]@econ.uni-hamburg.de, zacher.nikolaus@googlemail.com.

S. F. Crone, Department of Management Science, Lancaster University, Lancaster LA1 4YX, United Kingdom (e-mail: s.crone@lancaster.ac.uk).

0-7803-9490-9/06/$20.00/©2006 IEEE

separating surface; see Fig. 1. Since $y_i \in \{-1,+1\}$ we can formulate this constraint as:

$$y_i((\boldsymbol{w}\cdot\boldsymbol{x}_i)+b)\geq 1,\ i=1,...,m. \quad (1)$$

Examples which satisfy (1) with equality are called support vectors since they define the orientation of the resulting hyperplane.

To account for misclassifications, that is examples where constraint (1) is not met, the so called soft margin formulation of SVM introduces slack variables $\xi_i \in \Re$ [15]. Hence, to construct a maximal margin classifier one has to solve the convex quadratic programming problem (2):

$$\min_{w,b,\xi} \frac{\beta}{2}\|\boldsymbol{w}\|+(1-\beta)\sum_{i=1}^{m}\xi_i$$
$$s.t.:\ y_i((\boldsymbol{w}\cdot\boldsymbol{x}_i)+b)\geq 1-\xi_i,\ i=1,...,m. \quad (2)$$

β is a tuning parameter which allows the user to control the trade off between maximising the margin and classifying the training set without error. The primal decision variables $\boldsymbol{w}$ and b define the separating hyperplane, so that the resulting classifier takes the form:

$$y(\boldsymbol{x})=sgn\left((\boldsymbol{w}^*\cdot\boldsymbol{x})+b^*\right), \quad (3)$$

where $\boldsymbol{w}^*$ and b^* are determined by (2).

To construct more general non-linear decision surfaces SVMs implement the idea to map the input vectors into a high-dimensional feature space via an a priori chosen non-linear mapping function Φ. Constructing a separating hyperplane in this feature space leads to a non-linear decision boundary in the input space. Expensive calculation of dot products $\Phi(\boldsymbol{x}_i)\cdot\Phi(\boldsymbol{x}_j)$ in a high-dimensional space can be avoided by introducing a kernel function K, see (4). The structure of SVMs allows this kernel integration without affecting the overall algorithms or training procedure [15].

$$K(\boldsymbol{x}_i,\boldsymbol{x}_j)=\Phi(\boldsymbol{x}_i)\cdot\Phi(\boldsymbol{x}_j). \quad (4)$$

Prominent candidates for the kernel function are the linear, radial and polynomial kernel; e.g. [15].

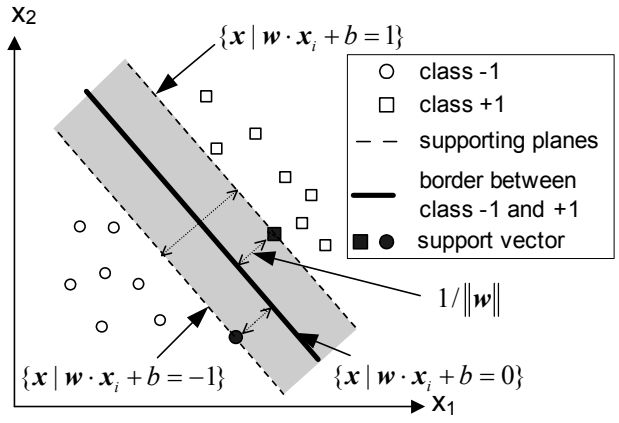


Fig. 1: Linear separation of two classes -1 and +1 in two-dimensional space with SVM classifier [16]

III. Approaches for Cost-Sensitive SVM

The problem of cost-sensitive learning is well established in the literature; e.g.[17, 18]. Let the entries c(i,j) of a loss-matrix $\boldsymbol{C}$ denote the cost of predicting class i when the true class is j. If we assume $\boldsymbol{C}$ to be asymmetric so that certain error types are more severe or costly than others, cost-sensitive learning refers to comprising this cost-information into the process of classifier induction. Approaches for cost-sensitive learning include algorithmic modifications to make individual learners cost-sensitive, e.g. [4, 14, 19] and meta-strategies designed to work with a broad variety of standard error based learners [20, 21]. Note that there is a strong connection between cost-sensitive learning and learning from imbalanced data sets so that these problems are commonly considered in a mutual framework [22, 23].

Following, we briefly review algorithmic modifications to make SVM cost-sensitive, and/or robust to skewed class distributions.

Considering the SVM classifier (2) and (3) there are three links to incorporate cost-sensitivity into SVM: The weight vector $\boldsymbol{w}$, the threshold b and the kernel K. The easiest way to bias SVM towards a minority and/or more important class is to manipulate the threshold b. This approach has been proposed by [14] and is also known as boundary movement [19] since the learned optimal separating hyperplane is altered as a post-processing step. The new resulting decision function is shown in (5) where the modifier Δb can be determined by ROC analysis [24].

$$y(\boldsymbol{x})=sgn\left((\boldsymbol{w}^*\cdot\boldsymbol{x})+b^*+\Delta b\right) \quad (5)$$

A well established approach to consider imbalanced class/cost distributions during SVM learning is to modify the SVM objective using individual error weights for each class [4, 13].

$$\min_{w,b,\xi} \frac{\beta}{2}\|\boldsymbol{w}\|+(1-\beta)\left(c^+\sum_{\{i|y_i=+1\}}\xi_i+c^-\sum_{\{i|y_i=-1\}}\xi_i\right)$$
$$s.t.:\ y_i((\boldsymbol{w}\cdot\boldsymbol{x}_i)+b)\geq 1-\xi_i,\ i=1,...,m. \quad (6)$$

In (6), the weights c^+ and c^- measure the severity of a misclassification of positive and negative examples. It has been shown that this modification affects the weight vector $\boldsymbol{w}$ in the SVM decision function, e.g. [15]. We identify (6) to be the standard version of cost-sensitive SVM and will refer to it as csSVM in the following.

Adjustments of the kernel to overcome class imbalance problems have been suggested in [25, 26]. The authors propose a kernel boundary alignment algorithm that directly alters the kernel matrix increasing its values in the vicinity of the decision boundary and decreasing them in non boundary areas. This magnifies the spatial resolution of the training examples near the boundary, particularly in the area close to the minority class, and is shown to allow a purer data separation.

All approaches focus on effectiveness, e.g. classification accuracy, and do not explicitly consider costs. However, the most accurate classifier is not necessarily the most cost efficient one and to obtain a deeper integration of cost and decision making aspects, we propose a discrete SVM in the following.

IV. Discrete Support Vector Machines

A. Motivation and mathematical formulation

The original SVM utilizes the distance of a misclassified point to the separating hyperplane to measure classification error. That is, the discrete classification error is replaced by the continuous proxy ξ_i for computational convenience; see (2). DSVM [1] reverses this simplification and replaces ξ_i by $\theta_i \in [0,1]$ to account for asymmetric misclassification costs in a more intuitive manner. In addition, we substitute the L2-norm in SVMs' objective by the L1-norm. The L1-norm forces more elements of the weight vector to zero and therewith increases the interpretability of the model [27, 28]. Since model comprehensibility and transparency are deemed important in business and medical areas we believe the L1-norm to beneficial for these domains. Further more, using the L1-norm facilitates the usage of fast algorithms for solving linear programs as a sub-step within our tabu search heuristic; see IV.B for details.

The resulting formulation is given in (7) where u_j denotes a primal decision variable that controls the value of the weight vector $\boldsymbol{w}$ and Q is a sufficient large number.

$$\begin{aligned} \min_{w,b,\theta,u} \ & \frac{\beta}{2}\sum_{j=1}^{n} u_j + (1-\beta)\left(c^{+}\sum_{\{i|y_i=+1\}} \theta_i + c^{-}\sum_{\{i|y_i=-1\}} \theta_i \right) \\ s.t.: \ & y_i((\boldsymbol{w}\cdot\boldsymbol{x}_i)+b) \geq 1 - Q\theta_i, && i=1,\ldots,m \\ & -u_j \leq w_j \leq u_j && j=1,\ldots,n \\ & \theta_j \in [0,1] && j=1,\ldots,n \\ & u_j \geq 0 && j=1,\ldots,n. \end{aligned} \tag{7}$$

Note that DSVM allows a different interpretation of c^{+} and c^{-} than csSVM (6). From a decision making point of view these values serve as an abstract measure of error importance in csSVM that is used to weight a proxy for the classification error. As a rule of thumb the reciprocal of the class prior is a prominent choice for these parameters [27, 29]. On the contrary, in DSVM we can interpret c^{+} and c^{-} a real cost values, e.g. in an economical sense. Consider for example the case of credit scoring. A false positive error, e.g. predicting a defaulting customer as credit worthy, is associated with a certain cost and while we could directly incorporate such values or respective estimates into (7) their usage in (6) is questionable due to the multiplication with a continuous distance. Hence, even if a user has an idea about class specific misclassification costs, there is no straightforward rule how to translate them into a suitable setting for c^{+} and c^{-}. Furthermore, errors of the same type have in general varying impact on the objective depending on ξ_i. In fact, this is another reason why DSVM might be sounder for cost-sensitive learning.

Problem (7) is a linear program with continuous (w,b,u) and discrete (θ) decision variables.

As in the original SVM the objective includes a margin maximization part and an (empirical) error minimization part and therefore implement Vapnik's principle of structural risk minimization [6]. Additionally, we can interpret (7) as an approach to optimize generalization performance and misclassification cost in parallel.

Obviously, DSVM is computationally much more expensive than (6) due to the integer constraint. Since standard SVM optimization techniques are no longer applicable we develop a tabu search (TS) algorithm to be described in the following.

B. A tabu search heuristic for DSVM

To solve (7) we start with considering a relaxation of DSVM where the integer constraint for θ is dropped.

$$\begin{aligned} \min_{w,b,\theta,u} \ & \frac{\beta}{2}\sum_{j=1}^{n} u_j + (1-\beta)\left(c^{+}\sum_{\{i|y_i=+1\}} \theta_i + c^{-}\sum_{\{i|y_i=-1\}} \theta_i \right) \\ s.t.: \ & y_i((\boldsymbol{w}\cdot\boldsymbol{x}_i)+b) \geq 1 - Q\theta_i, && i=1,\ldots,m \\ & -u_j \leq w_j \leq u_j && j=1,\ldots,n \\ & 0 \leq \theta_j \leq 1 && j=1,\ldots,n \\ & u_j \geq 0 && j=1,\ldots,n \end{aligned} \tag{8}$$

The linear program (8) provides an upper bound for the optimal solution of DSVM and can be solved efficiently using the simplex method; e.g. [30].

In order to solve the discrete SVM problem (7) we developed a tabu search (TS) algorithm. TS is a meta-heuristic to solve combinatorial optimization problems. The idea is to find a feasible solution and search its neighborhood for better candidates using local hill-climbing strategies. Here, better means higher/lower objective values for maximization/minimization problems. However, the TS objective does not necessarily coincide with the MIP's objective [1]. The name TS originates from the fact that the algorithms incorporates some heuristics which prohibit certain moves (tabu moves) to avoid cycling and stops at suboptimal points [31]; see [32-34] for details.

Our TS implementation is based on the observations that feasible solutions, and consequently also the optimal solution, for the zero-one problem (7) can be found in an extreme point of the relaxation (8); see e.g. [35]. Therefore, the extreme points of the polyhedral constraint region defined by (8) form a natural neighbourhood for TS and each extreme point is a basic feasible solution (BFS). The general structure of such an extreme point tabu search (EPTS) [35, 36] is as follows: 1) Use the simplex method to

find an extreme point e for (8) and use it as an initial solution. 2) Examine adjacent solution in the neighbourhood of e. These are all solution that could be obtained by ordinary simplex pivot operations, e.g. exchanging a current basis variable for a non-basis variable. 3) Select the move that results in the largest improvement of the objective value and is not contained in the tabu list. 4) Execute the selected move and update the tabu list using information on the time a variable is pivoted (recency information) and its overall numbers of pivots (frequency information). To transform the generic EPTS schema in a concrete algorithm one has to define the strategy for screening the candidate list in step 2), the move evaluation function and the rules and memory structures for the tabu list.

Note that each TS move can increase/decrease the current objective value z and increase/decrease the current amount of integer infeasibility ii; that is the amount a given solution fails to fulfil the integer constraint. Therefore, every pivot operation belongs to one for four elementary types, e.g. increase z and decrease ii (best moves) or decrease z and increase ii (worst moves) [37].

Our TS implementation evaluates each move within the candidate list according to its move type. To resolve situations in which one can either increase z on the cost of increasing ii or decrease ii on the cost of decreasing z we incorporate a strategic oscillation component [35] so that the algorithms strives to improve z for a given number of iterations, then switches to decreasing ii, then switched back to z improvements, etc. This trade-off is illustrated in Fig. 2.

While the tabu status in our implementation is solely based on recency and frequency information (see above) we use an aspiration criterion to allow moves that lead to a new best feasible MIP solution even if they are currently tabu.

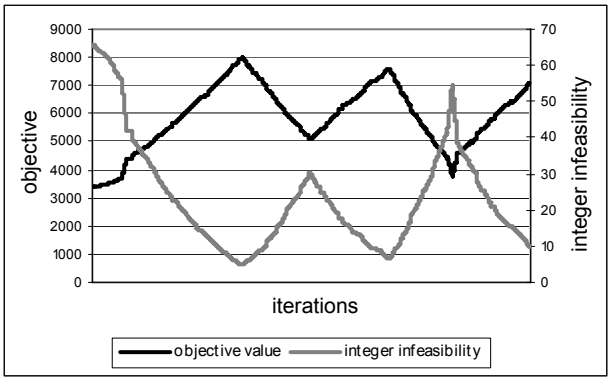


Fig. 2: Trade-off between TS moves that lower ii and improve objective value.

C. SVM decision tree

Introducing an integer constraint into the SVM formulations hinders application of the kernel trick to construct non-linear classifiers. In particular, the relaxed primal constraint $0 \le \theta_j \le 1$ in (8) results in a non-linear constraint in the dual problem. Consequently, solving the dual formulation is complicated. Note that this constraint is independent of the norm of the weight vector so that the same problem arises when using the L2-norm.

To overcome this limitation the construction of a hierarchical SVM decision tree has been proposed in [1]. We adopt this idea and recursively partition the data by linear DSVMs until some predefined conditions are met. To avoid over-fitting or the necessity to prune the SVM decision tree the following rules have to be regarded during the tree generation progress: 1) the tree deep, e.g. the number of levels, must not exceed a specified value, 2) splitting a node is impossible if the ratio of minority class examples falls below a specified threshold, and 3) any node must contain a specified minimum number of instances to remain divisible.

V. Empirical Study

A. Overview

The empirical evaluation of DSVM strives to explore the potential of DSVM and csSVM in cost-sensitive scenarios. We consider four data sets from the Statlog project [38] and the UCI machine learning library [39] as a case study. The two credit scoring data sets Australian credit (ac) and German credit (gc) serve as examples from the field of managerial decision making while heart-disease (hrt) and Wisconsin breast cancer (wbc) exemplify cases of medical diagnosis. A brief description of data set characteristics is given in Table 1. For detailed information on data set origin, task and variable description the reader is referred to [38, 40].

Table 1:
Data set characteristics*

	#Cases	#Features	#Class -1	#Class +1
ac	690	14	307	383
gc	1000	24	700	300
hrt	270	13	150	120
wbc	683	10	239	444

* We use the pre-processed data sets that are available via the LIBSVM homepage [41].

The data sets have been partitioned into 2/3 training set for model building and 1/3 test set for out-of-sample evaluation.

B. Predictive accuracy of DSVM

Preceding an analysis of asymmetric misclassification costs we have to verify the predictive accuracy of our DSVM implementation for standard situations. Therefore, we compare DSVM versus csSVM with linear (lin), radial (rad) and polynomial (poly) kernel function. Since DSVM is a linear classifier, we use the decision tree extension (Section IV.C) to construct hierarchical classifiers with a tree deep of two (DSVM-DT2) and three (DSVM-DT3) levels. The balanced error rate (BER) is used to measure predictive accuracy. BER is calculated as:

$$BER = 0.5\left(\frac{FP}{N_+} + \frac{FN}{N_-}\right). \quad (9)$$

Here, *FP* denotes the number of false positive cases, e.g. the number of false alarms while *FN* (false negatives) measures the number of missed true alarms. We use N_+ / N_- to represent the overall number of positive/negative examples with in data set.

Hyperparameters, e.g. β and kernel parameters have been determined by cross-validation (CV) adopting a grid search strategy [7, 42]. For each classifier, the parameter setting providing the lowest 10-fold CV BER is selected and evaluated on the test set. Results are given in Table 2.

TABLE 2:
BALANCED ERROR RATE OF CSSVM AND DSVM

	Training[1]			Test		
	BER	FP	FN	BER[2]	FP	FN
ac						
SVM (lin)	0.13	43	18	0.15	26	10
SVM (rad)	0.14	48	17	**0.14**	29	6
SVM (poly)	0.09	15	24	0.20	16	28
DSVM	0.13	46	15	0.19	31	8
DSVM-DT2	0.10	26	20	0.19	25	17
DSVM-DT3	0.09	26	15	0.19	28	14
gc						
SVM (lin)	0.32	46	98	0.34	13	66
SVM (rad)	0.32	30	63	0.34	17	63
SVM (poly)	0.32	30	72	0.32	14	61
DSVM	0.30	37	106	0.33	17	70
DSVM-DT2	0.28	37	106	**0.31**	14	65
DSVM-DT3	0.28	37	106	**0.31**	14	65
hrt						
SVM (lin)	0.14	12	14	0.21	6	10
SVM (rad)	0.15	11	17	0.22	4	12
SVM (poly)	0.13	10	14	**0.20**	5	10
DSVM	0.12	8	15	**0.20**	8	8
DSVM-DT2	0.08	8	7	0.21	7	10
DSVM-DT3	0.05	5	5	0.21	7	10
wbc						
SVM (lin)	0.03	5	8	0.04	3	3
SVM (rad)	0.03	4	9	**0.03**	2	5
SVM (poly)	0.02	4	4	0.07	7	4
DSVM	0.02	10	4	0.04	7	2
DSVM-DT2	0.02	5	5	0.04	4	3
DSVM-DT3	0.02	5	4	0.04	5	3

[1] Results on the training set are calculated by means of 10-fold CV.
[2] We use bold face to highlight the classifier that provides the lowest BER for the respective data set.

Table 2 reveals that DSVM's predictions are competitive to standard SVM for all data sets. In addition, the quality of DSVM is further confirmed by comparing our results with other benchmarking studies [1, 7, 43] that have used the same data. Having verified the predictive accuracy of our DSVM implementation we consider cost-sensitivity in the following section.

C. Cost-efficiency of DSVM

DSVM provides a more explicit integration of misclassification errors avoiding the use of a continuous error proxy. Therefore, we can expect DSVM to outperform standard SVM in scenarios involving asymmetric misclassification costs since it facilitates direct cost-minimization. In order to confirm this assumption we compare DSVM versus csSVM (6) under different cost-distributions contrasting their capability to derive cost-efficient predictions.

We consider applications where a false alarm is less severe than missing a correct one. Let C^+ denote the cost for a missed alarm, e.g. a bad credit risk, and C^- the costs for false alarm respectively. With no loss of generality we can set C^- to one and scale C^+ accordingly. Obviously, there is some point $\bar{C}^+ \gg C^-$ where a classifier completely avoids the more expensive error type *FN*. A pre-test revealed that this point is reached at latest at a ratio of $C^+ : C^- = 50:1$ for csSVM. Consequently our study incorporates cost distributions of $C^+ : C^- = 2:1$ to 50:1.

Aiming at a proximate translation of cost values into classifier parameters we considered a fixed setting for the c^+ and c^- parameters in csSVM and DSVM, see (6) and (7), directly using the respective $C^+ : C^-$ ratio. This decreases the number of free parameters and reflects the previously developed understanding of these parameters for DSVM. Subsequently, the remaining free parameter (kernel parameters and the trade-off parameter β) are determined by means of 10-fold CV using the resulting misclassification costs as selection criterion. That is, the classifier providing minimal misclassification costs within 10-fold CV is selected and evaluated on the test set. This procedure is repeated for each $C^+ : C^-$ ratio. Results are presented in Table 3.

TABLE 3:
RESULTS FOR CSSVM AND DSVM UNDER DIFFERENT COST-DISTRIBUTIONS WITH FIXED COST PARAMETERIZATION

	Results for csSVM				Results for DSVM			
Data set	Training		Test		Training		Test	
Cost ratio	FP	FN	FP	FN	FP	FN	FP	FN
ac								
2:1-5:1	40	10	32	7	46	15	31	8
6:1-10:1	50	10	36	6	49	11	34	7
11:1-15:1	100	0	57	3	102	6	58	3
16:1-20:1	120	0	68	3	141	3	74	3
25:1-50:1*	130	0	70	2	142	2	74	2
gc								
2:1-5:1	180	40	80	24	238	23	116	20
6:1-10:1	320	10	153	5	377	3	188	4
11:1-15:1	410	0	202	1	395	3	185	3
16:1-20:1	450	0	219	0	400	2	185	3
25:1-50:1*	450	0	221	0	400	2	185	3
hrt								
2:1-5:1	20	1	13	7	24	6	13	7
6:1-10:1	40	0	23	3	39	2	17	6
11:1-15:1	50	0	26	1	49	2	22	3
16:1-20:1	50	0	28	0	50	2	23	3
25:1-50:1*	60	0	28	0	63	1	30	1
wbc								
2:1-5:1	10	0	7	3	8	4	6	2
6:1-10:1	20	0	10	1	10	4	6	2
11:1-15:1	20	0	12	1	10	4	7	2
16:1-20:1	30	0	13	2	13	4	7	2
25:1-50:1*	40	0	15	1	22	2	11	1

* We used a step size of 5 to increase asymmetry in cost distributions from a ratio of 20:1 onwards. Consequently, this group contains approximately the same number of elements as the other groups.

We aggregate the different cost ratios into five groups and report the number of false positive and false negative predictions instead of one unique misclassification cost. Further more, csSVMs with linear, polynomial and radial kernel as well as DSVM with various tree levels are averaged for brevity of presentation.

The number of "expensive" false negative predictions is consistently decreased with increasing asymmetry of the $C^+ : C^-$ ratio for both classifiers. In fact, this type of error almost vanishes in highly asymmetric settings. Surprisingly, such results are not only obtained for DSVM but for csSVM as well. For the hrt data set, csSVM even outperforms DSVM on the test set in terms of FN errors. Regarding the less severe FP error type (false alarm) DSVM is slightly better than csSVM giving superior predictions in 13 out of 20 cases. While these results support our initial assumption that a direct integration of misclassification costs is suitable for DSVM, the competitive performance of csSVM has not been expected. On the one hand, this finding justifies the common application of csSVM for cost-sensitive learning. Further more, the overall low error rates actually motivate an adoption of our strategy for csSVM parameterization. The parameters c^+ and c^- are routinely set to the reciprocal of the prior probability for the positive / negative class [27, 29] and our results suggest that this rule of thumb can be extended to situations involving asymmetric misclassification costs.

In order to evaluate the implications of such a tuning heuristic, we conduct a second line of experiments allowing any possible setting for c^+ and c^- to obtain a cost minimal classifier for a given $C^+ : C^-$ ratio. That is, we allow usage of a csSVM or DSVM classifier with parameters settings of e.g. $c^+ = 2$ and $c^- = 1$ even if the assumed application domain involves a cost ratio of $C^+ : C^- = 50 : 1$. Consequently, the number of free parameters is increased dramatically. Results are given in Table 4.

Comparing Table 3 and Table 4 the larger number of free parameters has not helped to further improve csSVM results in general. We can observe a purer separation of the training data due to a larger number of degrees of freedom. Regarding to test set performance the number of FP errors has decreased at the cost of committing additional more severe FN errors. For the considered cost ratios this would induce overall higher misclassification costs. Consequently, the larger number of free parameters hinders effective model selection and leads to slightly deteriorate test results. This confirms that a proximate integration of misclassification costs is actually a good strategy for the considered data sets. In view of the fact, that there is to our best knowledge no broadly accepted procedure for SVM parameter setting in cost-sensitive scenarios, e.g. a cost-sensitive grid search, our results can be seen as a first step to develop a sophisticated tuning heuristic for csSVM.

TABLE 4:
RESULTS FOR CSSVM AND DSVM UNDER DIFFERENT COST-DISTRIBUTIONS WITH FREE COST PARAMETERIZATION

	Results for csSVM				Results for DSVM			
Data set	Training		Test		Training		Test	
Cost ratio	FP	FN	FP	FN	FP	FN	FP	FN
ac								
2:1-5:1	50	10	33	7	50	10	34	7
6:1-10:1	80	0	46	6	73	7	45	5
11:1-15:1	80	0	46	6	113	2	63	3
16:1-20:1	80	0	46	6	113	2	63	3
25:1-50:1	80	0	46	6	142	1	76	2
gc								
2:1-5:1	170	4	76	26	238	23	116	20
6:1-10:1	350	0	168	2	377	3	188	4
11:1-15:1	360	0	175	1	395	3	185	3
16:1-20:1	360	0	175	1	400	2	185	3
25:1-50:1	360	0	175	1	400	2	185	3
hrt								
2:1-5:1	20	0	12	7	27	4	13	7
6:1-10:1	20	0	13	6	34	2	16	6
11:1-15:1	20	0	13	6	45	1	21	3
16:1-20:1	20	0	13	6	60	0	28	0
25:1-50:1	20	0	13	6	60	0	28	0
wbc								
2:1-5:1	0	0	4	4	10	4	6	2
6:1-10:1	0	0	4	4	21	2	10	2
11:1-15:1	0	0	4	4	29	1	14	1
16:1-20:1	0	0	4	4	29	1	14	1
25:1-50:1	0	0	4	4	29	1	14	1

Results for the DSVM classifier have not changed significantly. This is explained by the fact that DSVM does not provide a large number of free parameters beside c^+ and c^-. While the novel setting allows a broader value range for these parameters, this flexibility is obviously not exploited so that the results remain stable. This verifies that the previous setting, e.g. direct translation of misclassification costs into parameter values, is indeed appropriate for DSVM and that further improvements are hardly achievable.

VI. CONCLUSIONS

We considered the case of cost-sensitive learning using SVM classifiers. DSVM appeared to be a very promising candidate for such scenarios due to its accurate error measurement. On the other hand, csSVM is the standard formulation for cost-sensitive classification with SVMs. However, the usage of a continuous approximation of misclassification error puts csSVMs appropriateness into perspective. Therefore, we compared csSVM and DSVM within an empirical experiment to evaluate their capabilities to derive cost-efficient predictions.

Our results confirmed that DSVM indeed provides highly accurate predictions when the distributions of misclassification error are uneven. In addition, we found csSVM to give surprisingly good results as well. In spite of its algorithmic treatment of classification errors, a direct translation of cost values into parameter settings emerged to be an efficient approach confirming and extending tuning heuristics for csSVM.

REFERENCES

[1] C. Orsenigo and C. Vercellis, "Discrete Support Vector Decision Trees via Tabu Search," *Computational Statistics & Data Analysis*, vol. 47, pp. 311-322, 2004.

[2] L. C. Thomas, "A Survey of Credit and Behavioral Scoring; Forecasting financial risk of lending to consumers," *International Journal of Forecasting*, vol. 16, pp. 149-172, 2000.

[3] D. West, "Neural network credit scoring models," *Computers & Operations Research*, vol. 27, pp. 1131-1152, 2000.

[4] Veropoulos, N. Cristianini, and C. Campbell, "Controlling the Sensitivity of Support Vector Machines," *Proc. of the 16th Intern. Joint Conf. on Artificial Intelligence*, Stockholm, Sweden, 1999.

[5] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A Training Algorithm for Optimal Margin Classifiers," *Proc. of the 5th Annual Workshop on Computational Learning Theory*, Pittsburgh, Pennsylvania, USA, 1992.

[6] V. N. Vapnik, *The Nature of Statistical Learning Theory*. New York: Springer, 1995.

[7] B. Baesens, T. Van Gestel, S. Viaene, M. Stepanova, J. Suykens, and J. Vanthienen, "Benchmarking state-of-the-art classification algorithms for credit scoring," *Journal of the Operational Research Society*, vol. 54, pp. 627-635, 2003.

[8] H. Shin and S. Cho, "Response modeling with support vector machines," *Expert Systems with Applications*, vol. In Press, Corrected Proof.

[9] Y. Wang, I. V. Tetko, M. A. Hall, E. Frank, A. Facius, K. F. X. Mayer, and H. W. Mewes, "Gene selection from microarray data for cancer classification - a machine learning approach," *Computational Biology and Chemistry*, vol. 29, pp. 37-46, 2005.

[10] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik, "Gene Selection for Cancer Classification using Support Vector Machines," *Machine Learning*, vol. 46, pp. 389-422, 2002.

[11] G. Fumera and F. Roli, "Cost-sensitive learning in Support Vector Machines," *Proc. of the Workshop on Machine Learning, Methods and Applications*, Siena, Italy, 2002.

[12] P. Geibel, U. Brefeld, and F. Wysotzki, "Perceptron and SVM learning with generalized cost models," *Intelligent Data Analysis*, vol. 8, pp. 439-455, 2004.

[13] Y. Lin, Y. Lee, and G. Wahba, "Support Vector Machines for Classification in Nonstandard Situations," *Machine Learning*, vol. 46, pp. 191-202, 2002.

[14] G. Karakoulas and J. Shawe-Taylor, "Optimizing classifiers for imbalanced training sets," *Advances in Neural Information Processing Systems*, 1999.

[15] N. Cristianini and J. Shawe-Taylor, *An introduction to support vector machines and other kernel-based learning methods*. Cambridge: Cambridge University Press, 2000.

[16] C. J. C. Burges, "A Tutorial on Support Vector Machines for Pattern Recognition," *Data Mining and Knowledge Discovery*, vol. 2, pp. 121-167, 1998.

[17] C. Elkan, "The Foundations of Cost-Sensitive Learning," *Proc. of the 7th Intern. Joint Conf. on Artificial Intelligence*, Seattle, Washington, USA, 2001.

[18] S. Viaene and G. Dedene, "Cost-sensitive learning and decision making revisited," *European Journal of Operational Research*, vol. 166, pp. 212-220, 2004.

[19] G. Wu and E. Y. Chang, "KBA: Kernel Boundary Alignment considering Imbalanced Data Distribution," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, pp. 786-795, erscheint 2005.

[20] P. Domingos, "MetaCost: a general method for making classifiers cost-sensitive," *Proc. of the 5th Intern. Conf. on Knowledge Discovery and Data Mining*, San Diego, CA, USA, 1999.

[21] W. Fan, S. J. Stolfo, J. Zhang, and P. K. Chan, "AdaCost: Misclassification Cost-Sensitive Boosting," *Proc. of the 16th Intern. Conf. on Machine Learning*, Bled, Slovenia, 1999.

[22] N. Japkowicz and S. Stephen, "The Class Imbalance Problem: A Systematic Study," *Intelligent Data Analysis*, vol. 6, pp. 429-450, 2002.

[23] G. M. Weiss, "Mining with rarity: a unifying framework," *ACM SIGKDD Explorations Newsletter*, vol. 6, pp. 7-19, 2004.

[24] E. Carrizosa and B. Martin-Barragan, "Two-group classification via a biobjective margin maximization model," *European Journal of Operational Research*, appear 2006.

[25] G. Wu and E. Y. Chang, "Class-Boundary Alignment for Imbalanced Dataset Learning," in *ICML Workshop on Learning from Imbalanced Data Sets*. Washington DC, USA, 2003.

[26] G. Wu and E. Y. Chang, "KBA: Kernel Boundary Alignment considering Imbalanced Data Distribution," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, pp. 786-795, 2005.

[27] K. P. Bennett, S. Wu, and L. Auslender, "On support vector decision trees for database marketing," *Proc. of the Intern. Joint Conf. on Neural Networks*, Washington D.C., USA, 1999.

[28] P. S. Bradley and O. L. Mangasarian, "Feature Selection via Concave Minimization and Support Vector Machines," *Proc. of the 15th Intern. Conference on Machine Learning*, Madison, Wisconsin, 1998.

[29] C.-W. Hsu, C.-C. Chang, and C.-J. Lin, "A Practical Guide to Support Vector Classification," Department of Computer Science and Information Engineering National Taiwan University, Taipei, Taiwan 2003.

[30] K. G. Murty, *Linear programming*. New York: Wiley, 1983.

[31] F. Glover, "Tabu search - wellsprings and challenges," *European Journal of Operational Research*, vol. 106, pp. 221-225, 1998.

[32] F. Glover and M. Laguna, *Tabu search*. Boston: Kluwer, 1997.

[33] F. Glover, "Tabu search: Part 1," *ORSA Journal on Computing*, vol. 1, pp. 190-206, 1989.

[34] F. Glover, "Tabu search: Part 2," *ORSA Journal on Computing*, vol. 2, pp. 4-32, 1990.

[35] A. Lokketangen and F. Glover, "Solving zero-one mixed integer programming problems using tabu search," *European Journal of Operational Research*, vol. 106, pp. 624, 1998.

[36] J. A. Blue and K. P. Bennett, "Hybrid extreme point tabu search," *European Journal of Operational Research*, vol. 106, pp. 676, 1998.

[37] A. Lokketangen and F. Glover, "Candidate List and Exploration Strategies for Solving 0/1 MIP Problems using a Pivot Neighborhood," in *Meta-Heuristics. Advances and Trends in Local Search Paradigms for Optimization*, S. Voß, S. Martello, I. H. Osman, and C. Roucairol, Eds. Boston: Kluwer, 1998.

[38] D. Michie, D. J. Spiegelhalter, and C. C. Taylor, *Machine learning, neural and statistical classification*. New York: Horwood, 1994.

[39] D. J. Newman, S. Hettich, C. L. Blake, and C. J. Merz, "UCI Repository of machine learning databases," Department of Information and Computer Science, University of California, Irvine, CA, 1998.

[40] T.-S. Lim, W.-Y. Loh, and Y.-S. Shih, "A Comparison of Prediction Accuracy, Complexity, and Training Time of Thirty-Three Old and New Classification Algorithms," *Machine Learning*, vol. 40, pp. 203-228, 2000.
[41] C.-C. Chang and C.-J. Lin, "LIBSVM - A Library for Support Vector Machines," Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm, 2001.
[42] T. van Gestel, J. A. K. Suykens, B. Baesens, S. Viaene, J. Vanthienen, G. Dedene, B. de Moor, and J. Vandewalle, "Benchmarking Least Squares Support Vector Machine Classifiers," *Machine Learning*, vol. 54, pp. 5-32, 2004.
[43] Y. Lin, "Support Vector Machines and the Bayes Rule in Classification," *Data Mining and Knowledge Discovery*, vol. 6, pp. 259-275, 2002.

2006 International Joint Conference on Neural Networks
Sheraton Vancouver Wall Centre Hotel, Vancouver, BC, Canada
July 16-21, 2006

Genetic Algorithms for Support Vector Machine Model Selection

Stefan Lessmann, Robert Stahlbock, Sven F. Crone

Abstract— **The support vector machine is a powerful classifier that has been successfully applied to a broad range of pattern recognition problems in various domains, e.g. corporate decision making, text and image recognition or medical diagnosis. Support vector machines belong to the group of semi-parametric classifiers. The selection of appropriate parameters, formally known as model selection, is crucial to obtain accurate classification results for a given task. Striving to automate model selection for support vector machines we apply a meta-strategy utilizing genetic algorithms to learn combined kernels in a data-driven manner and to determine all free kernel parameters. The model selection criterion is incorporated into a fitness function guiding the evolutionary process of classifier construction. We consider two types of criteria consisting of empirical estimators or theoretical bounds for the generalization error. We evaluate their effectiveness in an empirical study on four well known benchmark data sets to find that both are applicable fitness measures for constructing accurate classifiers and conducting model selection. However, model selection focuses on finding one best classifier while genetic algorithms are based on the idea of re-combining and mutating a large number of good candidate classifiers to realize further improvements. It is shown that the empirical estimator is the superior fitness criterion in this sense, leading to a greater number of promising models on average.**

I. Introduction

THE support vector machine (SVM) is a prominent classifier that has been introduced by Vapnik and co-workers in 1992 [1, 2]. In subsequent years the technique has received considerable attention in various application domains. Promising results have been obtained for e.g. medical diagnosis [3, 4], text and image recognition [5, 6] or the support of corporate decision making [7, 8].

SVMs are supervised learners that construct a model from available training data with known classification. In order to obtain accurate class predictions SVMs provide a number of free parameters that have to be tuned to reflect the requirements of the given task. We will use the term model to refer to a specific classifier, e.g. a SVM with specified kernel and kernel parameters.

The process of parameter fitting is known as model selection aiming at finding a model which will give minimum prediction error when being applied to classify unseen examples that originate from the same source as the training data. Since this true generalization performance is inaccessible we have to rely on appropriate estimators.

Within the scope of SVM model selection we can distinguish two major methodologies. The empirical approach to model selection involves estimating the generalization error by re-sampling techniques such as disjoint hold-out sets or cross-validation (CV) while theoretical approaches consist of constructing and minimizing algebraic bounds for the generalization error.

In this work, we propose a meta-strategy utilizing a genetic algorithm (GA) for model selection striving to determine all properties of the classifier in a solely data-driven manner. A particular classifier is assessed on the basis of its fitness that reflects arbitrary model selection criteria. Consequently, the fitness is the proxy for generalization error and is used to guide the evolutionary process of SVM model construction. We consider the CV performance as a popular empirical estimator for generalization error and the ratio of support vectors and data instances as a classical algebraic bound. Their effectiveness is contrasted in an empirical study using four well known benchmark data sets.

The remainder of the paper is organized as follows. Section II provides an introduction to SVMs while we review previous work on SVM model selection in Section III. Our GA based approach is presented in Section IV. The numerical results of an experimental study are described in Section V. Conclusions are given in Section VI.

II. Support Vector Machines

The SVM can be characterized as a supervised learning algorithm capable of solving linear and non-linear binary classification problems. Given a training set with m patterns $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^{m}$, where $\boldsymbol{x}_i \in \mathrm{X} \subseteq \Re^n$ is an input vector and $y_i \in \{-1,+1\}$ its corresponding binary class label, the idea of support vector classification is to separate examples by means of a maximal margin hyperplane [9]. Therefore, the algorithm strives to maximize the distance between examples that are closest to the decision surface. The margin of separation is related to the so called Vapnik-Chervonenkis dimension (VCdim) which measures the complexity of a learning machine [10]. The VCdim is used in several bounds for the generalization error of a learner and it is known that margin maximization is beneficial for the generalization ability of the resulting classifier [11]. To construct the SVM classifier one has to minimize the norm of the weight vector $\boldsymbol{w}$ under the constraint that the training patterns of each class reside on opposite sides of the separating surface; see Fig. 1.

S. Lessmann, R. Stahlbock (corresponding author), Institute of Information Systems, University of Hamburg, Von-Melle-Park 5, D-20146 Hamburg, Germany (phone: 0049-40-42838-3063; fax: 0049-40-42838-5535; e-mail: [lessmann, stahlbock] @econ.uni-hamburg.de).

S. F. Crone, Department of Management Science, Lancaster University, Lancaster LA1 4YX, United Kingdom (e-mail: s.crone@lancaster.ac.uk).

0-7803-9490-9/06/$20.00/©2006 IEEE

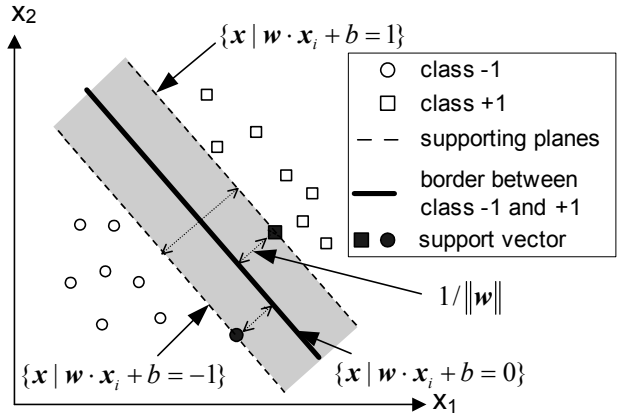


Fig. 1: Linear separation of two classes -1 and +1 in two-dimensional space with SVM classifier [12].

Since $y_i \in \{-1,+1\}$ we can formulate this constraint as

$$y_i((\boldsymbol{w} \cdot \boldsymbol{x}_i) + b) \geq 1, \ i = 1,...,m. \tag{1}$$

Examples which satisfy (1) with equality are called support vectors since they define the orientation of the resulting hyperplane.

To account for misclassifications, e.g. examples where constraint (1) is not met, the soft margin formulation of SVM introduces slack variables $\xi_i \in \Re$ [9]. Hence, to construct a maximal margin classifier one has to solve the convex quadratic programming problem (2):

$$\begin{aligned} &\min_{w,b,\xi} \frac{1}{2}\|\boldsymbol{w}\| + C\sum_{i=1}^{m}\xi_i \\ &s.t.: \ y_i((\boldsymbol{w} \cdot \boldsymbol{x}_i) + b) \geq 1 - \xi_i, \ i = 1,...,m. \end{aligned} \tag{2}$$

C is a tuning parameter which allows the user to control the trade off between maximizing the margin (first term in the objective) and classifying the training set without error. The primal decision variables $\boldsymbol{w}$ and b define the separating hyperplane, so that the resulting classifier takes the form

$$y(\boldsymbol{x}) = sgn\left((\boldsymbol{w}^* \cdot \boldsymbol{x}) + b^*\right), \tag{3}$$

where $\boldsymbol{w}^*$ and b^* are determined by (2).

Instead of solving (2) directly, it is common practice to solve its dual (4):

$$\begin{aligned} \max_a = \quad & \sum_{i=1}^{m} a_i - \frac{1}{2}\sum_{i,j=1}^{m} a_i a_j y_i y_j \boldsymbol{x}_i \boldsymbol{x}_j \\ s.t.: \quad & \sum_{i=1}^{m} a_i y_i = 0 \\ & 0 \leq a_i \leq C \ \ \forall i. \end{aligned} \tag{4}$$

In (4), a_i denotes the Lagrange variable for the i^{th} constraint of (1). Since the input vectors enter the dual only in form of dot products the algorithm can be generalized to non-linear classification by mapping the input data into a high-dimensional feature space via an a priori chosen non-linear mapping function Φ. Constructing a separating hyperplane in this feature space leads to a non-linear decision boundary in the input space. Expensive calculation of dot products $\Phi(\boldsymbol{x}_i) \cdot \Phi(\boldsymbol{x}_j)$ in a high-dimensional space can be avoided by introducing a kernel function K (5):

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \Phi(\boldsymbol{x}_i) \cdot \Phi(\boldsymbol{x}_j). \tag{5}$$

We obtain the general SVM classifier (6) with decision function (7):

$$\begin{aligned} \max_a = \quad & \sum_{i=1}^{m} a_i - \frac{1}{2}\sum_{i,j=1}^{m} a_i a_j y_i y_j \boldsymbol{K}\left(\boldsymbol{x}_i, \boldsymbol{x}_j\right) \\ s.t.: \quad & \sum_{i=1}^{m} a_i y_i = 0 \\ & 0 \leq a_i \leq C \ \ \forall i \end{aligned} \tag{6}$$

$$y(\boldsymbol{x}) = \text{sgn}\left(\sum_{i=1}^{m} \boldsymbol{a}_i y_i K(\boldsymbol{x}_i, \boldsymbol{x}) + b\right). \tag{7}$$

This kernel trick makes SVM flexible allowing the construction of special purpose kernels, e.g. for text classification [13].

III. Approaches for SVM Model Selection

Regarding the final SVM formulation (6), the free parameters of SVMs to be determined within model selection are given by the regularization parameter C and the kernel, together with additional parameters of the respective kernel function.

A generic approach to model selection, applicable with any learning algorithm, involves cross-validating a parameterized classifier on a sub-sample of available data that has not been used for training. Repetitive evaluation of a model on k disjoint sub-samples while the union of the remaining k-1 sub-samples is used to form the training set gives the well known CV estimate of generalization performance. We obtain the leave-one-out estimate [14] as a special case of CV by setting $k = m$-1. While being computationally expensive the leave-one-out estimator is appealing since it uses the largest possible amount of training data for model building.

For SVMs, CV-based model selection is popular in conjunction with previously determined kernels. In particular, when considering only Gaussian kernels (Table 1) the number of free parameters reduces to two (regularization parameter C and kernel width). These are routinely determined by means of a grid-search varying the parameter settings with a fixed step-size through a wide range of values and assessing the performance of every combination [7, 15]. To reduce the potentially large number of parameter combinations, Keerthi and Lin proposed a heuristic that starts with a linear kernel to determine C and subsequently executes a line search to find promising candidates for the parameters of a Gaussian SVM [16].

Due to extensive re-sampling and re-training of the classifier, these empirical techniques, and the calculation of the leave-one-out estimate in particular, are expensive. A computationally more feasible alternative is to construct algebraic bounds for the generalization error, or the leave-one out estimate respectively, which are easier to calculate. Using this approach, model selection is accomplished by as-

sessing a classifier's capability to minimize these bounds.

For SVMs, the task of developing classifier specific bounds has received considerable attention in the literature; e.g. [1, 17-19], see [20] for comparisons. For example, (8) describes a simple bound T for the leave-one-out error, given by the ratio of support vectors ($\#SV$) to the number of training examples [11]:

$$T = \frac{\#SV}{m}. \tag{8}$$

This bound is inspired by the idea that removing a non-support vector from the training set does not change the optimal solution of (6) and leaves the resulting classifier unchanged [21].

By calculating the derivatives of such bounds with respect to the free parameter one can develop efficient search techniques for finding high quality parameterizations, e.g. [21-23]. However, these bounds usually depend on certain assumptions, e.g. they are valid only for a specific kernel or require a separation of the training set without error. Therefore, meta-heuristics as generic search procedures have been proposed as an alternative facilitating the use of arbitrary, non-differentiable model selection criteria [24, 25].

IV. Genetic Algorithms for SVM Model Selection

A. Genetic algorithms

GA are meta-heuristics that imitate the long-term optimization process of biological evolution for solving mathematical optimization problems. They are based upon Darwin's principle of the 'survival of the fittest'. Problem solutions are abstract 'individuals' in a population. Each solution is evaluated by a fitness function. The fitness value expresses survivability of a solution, i.e. the probability of being a member of the next population and generating 'children' with similar characteristics by handing down genetic information via evolutionary mechanisms like reproduction, variation and selection, respectively. Reproduction and variation is achieved by mutation of genes and crossover. The latter combines characteristics of two solutions for deriving two new solutions. The coding of the problem into a genetic representation, e.g. the sequence of the phenotype's parameters on a genotype, is crucial to the performance of GA. Moreover, the fitness function has great impact on performance. The reader is referred to [26, 27] for more detailed information regarding GA.

B. Data driven construction of SVM kernels

Meta-heuristics like GA have been used in conjunction with SVM in several ways, e.g. for feature selection [28], optimizing SVM's parameters (assuming a fixed kernel) [29], and kernel construction [24, 25].

We believe that the task of feature selection resides more in the realms of data pre-processing than within model selection and discard it from further analysis. While GA can be used to tune the parameters of a specific SVM with fixed kernel, a data driven kernel construction is obviously more flexible so that we follow this approach.

It has been shown that if *K1* and *K2* are kernels, we can derive a new valid kernel $\tilde{K}$ by $\tilde{K} = K1 + K2$ and $\tilde{K} = K1 \bullet K2$, respectively [9]. Consequently, we can use any number of base kernels and combine them to build a combined kernel. This idea has been proposed by [25, 30, 31] and we implement it by using the basic kernels of Table 1.

TABLE 1:
BASIC KERNELS FOR CONSTRUCTION OF COMBINED KERNEL

Radial (K_{rad})	$K_{rad}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\alpha \left\| \boldsymbol{x}_i - \boldsymbol{x}_j \right\|^2\right)$
Polynomial (K_{poly})	$K_{poly}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \left(\alpha(\boldsymbol{x}_i \cdot \boldsymbol{x}_j) + \beta\right)^d$
Sigmoidal (K_{sig})	$K_{sig}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \tanh\left(\alpha(\boldsymbol{x}_i \cdot \boldsymbol{x}_j) + \beta\right)$
Anova (K_{anova})	$K_{anova}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \left(\sum_j \exp\left(-\alpha\left(\boldsymbol{x}_i - \boldsymbol{x}_j\right)\right)^2\right)^d$
Inverse multi-quadratic (K_{imq})	$K_{imq}(\boldsymbol{x}_i, \boldsymbol{x}_j) = 1/\sqrt{\left\| \boldsymbol{x}_i - \boldsymbol{x}_j \right\|^2 + \beta^2}$

Therewith, we obtain the combined kernel $\tilde{K}$ (9) with $\otimes_j \in \{+;\cdot\}\ \forall j = 1,...,4$:

$$\tilde{K} = K_{poly}^{\kappa_1} \otimes_1 K_{rad}^{\kappa_2} \otimes_2 K_{sig}^{\kappa_3} \otimes_3 K_{imq}^{\kappa_4} \otimes_4 K_{anova}^{\kappa_5}. \tag{9}$$

C. Genetic representation of SVM's combined kernel

In order to facilitate a data driven determination of the combined kernel (9) by means of GA we have to define a genotype encoding for the free parameters. This is accomplished by using five integer genes for the kernel exponents $(\kappa_1,...,\kappa_5)$, four binary genes for the kernel combination operators $(\otimes_1,...,\otimes_4)$, fifteen real valued genes for individual kernel parameters, e.g. (α, β, d) in Table 1, and one additional real valued gene for the regularization parameter. The overall genotype structure is shown in Fig. 2.

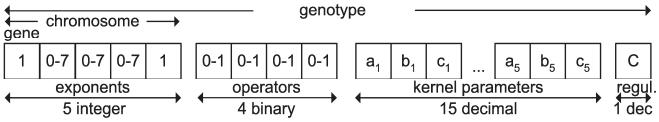


Fig. 2: Genotype encoding of SVM's combined kernel

We restrict the acceptable values for kernel exponent genes for computational reasons. In addition, these genes are superficial for polynomial and anova kernels that provide a kernel exponent as individual kernel parameter. Consequently, these genes have been set to one.

D. GA-based model selection

The GA-based development of SVMs is an iterative process starting with an initial population of randomly generated genotypes. Subsequently, SVMs are constructed by transferring the genotype's genetic code into a phenotype, i.e. a

SVM with a well defined combined kernel. After learning and (cross-)validation, each SVM is evaluated by the fitness function. Genetic operations use this quality information for building a new population of SVMs, which are trained and evaluated again. Thus, the whole learning process can be seen as subdivided into a microscopic cycle for learning of a SVM and a macroscopic evolutionary one; see Fig. 3.

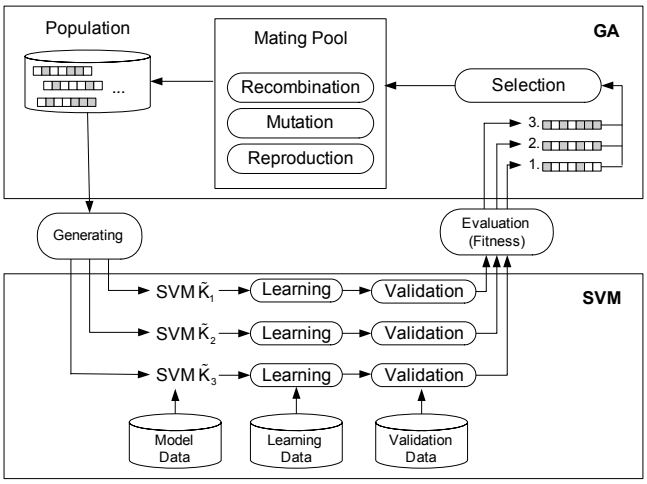


Fig. 3: Evolution of SVM by means of GA. Decoding of genotype into SVM is accomplished using the relationship between (9) and Fig. 2; here denoted as model data.

The fitness function is an important factor for evaluation and evolution of SVMs providing satisfactory and stable results in real-world applications. The fitness function guides the superordinated evolutionary learning process determining the probability that an individual can hand down genetic information to the subsequent population. Therefore, it should express the user's objective and should favour SVMs with satisfactory generalization ability in order to select useful classifiers systematically instead of accidentally. Consequently, the fitness function effectively conducts model selection and we can incorporate arbitrary model selection criteria as fitness measure.

Whereas the fitness function selects solutions for reproduction, the reproduction itself is conducted by means of mutation and crossover. The selection is implemented as tournament selection with a tournament size of two. Furthermore, an elitist mechanism is applied in order to ensure that the best SVM is member of the next generation.

The crossover operator is implemented as uniform crossover, i.e. all genes between two random points within a chromosome are interchanged between two genotypes representing parents for the resulting two new genotypes. Crossover is potentially applied to chromosomes for kernel aggregation and kernel exponent, whereas mutation can be applied to all chromosomes. The actual application of a genetic operation depends on user-defined rates. A high rate for crossing over and low rate for mutation are recommended. We set the crossover rate to 0.7 and the mutation rate for one gene to 0.3; see e.g. [26, 32]. Mutation is implemented as a stepwise increment or decrement with specific step size resulting in a new value within minimum and maximum limits. Binary genes are mutated by flipping 0 to 1 and vice versa.

V. Empirical Evaluation of GA-based Model Selection for SVM

A. Overview

We evaluate four data sets from the Statlog project and the UCI machine learning library. The data sets Australian credit (ac) and German credit (gc) exemplify a case of corporate credit scoring, e.g. classifying if an applicant is a good/bad credit risk. As examples for medical diagnosis we consider the data sets heart-disease (hrt), and Wisconsin breast cancer (wbc) each of which require a classification if a patient suffers from a certain disease or not. All sets are cases of binary classification so that examples either belong to a class +1 or a class -1 respectively. A brief description of each data set's characteristic is given in Table 2. For detailed information the reader is referred to [33-35].

Table 2:
Data set characteristics*

	#cases	#features	#class -1	#class +1
ac	690	14	307	383
gc	1000	20	700	300
hrt	270	13	150	120
wbc	683	10	239	444

* We use the pre-processed versions of the data sets available via the LIBSVM homepage [35].

The data sets have been partitioned into 2/3 training set for model building and 1/3 test set for out-of-sample evaluation. For each data set, the GA is used to construct a population of 50 individual SVMs. The evolutionary process of classifier assessment and fitness based recombination is run for 50 generations resulting in an overall number of 2,500 learned and evaluated SVMs per data set.

To consider empirical model selection procedures and algebraic bounds in a mutual framework we evaluated two different fitness criteria. In GA-1 fitness is measured by means of 10-fold CV balanced classification accuracy (*bca*) (10) whereas the bound (8) is used in GA-2. The *bca* is calculated as:

$$bca = \frac{1}{2}\left(\frac{\pi^-}{m^-} + \frac{\pi^+}{m^+}\right), \tag{10}$$

where m^- denotes the number of class -1 records in the data set and π^- the number of class -1 records that have been classified correctly with similar meanings for π^+ and m^+.

Results for GA-1 and GA-2 are contrasted with standard SVMs with linear, radial and polynomial kernel. Model selection for the standard SVMs is accomplished by means of extensive grid search, see Table 3.

TABLE 3:
PARAMETER RANGE FOR GRID SEARCH WITH STANDARD SVM *

	log(C)	d	log(α)	log(β)
Linear kernel	{-2,-1,…,3}	-	-	-
Radial kernel	{-2,-1,…,3}	-	{-2,-1,…,3}	-
Polynomial kernel	{-2,-1,…,2}	{2,3,4,5}	{-1,0,1}	{0,1,2}

*All parameters except the kernel exponent d for the polynomial kernel are varied on log scale. A minus sign indicates that the respective parameter is not present for the particular kernel.

B. Experimental Results

Following the idea of GA-based SVM model selection one chooses the individual with maximum overall fitness for future use on unseen data. To simulate this scenario, we assessed the performance by means of *bca* of the respective SVMs, e.g. the fittest member in the population, on the hold-out test set. To consider dynamical aspect of the GA, like the evolution of fitness and test performance, we report results on an aggregated generation level in Table 4 for GA-1 and Table 5 for GA-2 respectively.

TABLE 4:
RESULTS FOR GA-1 BASED MODEL SELECTION *

GA-1		GA		Standard SVM		Deviation
	Gen.	Best fitness	*bca* on test	Best fitness	*bca* on test	between GA and standard SVM
ac	10	0.8878	***0.8376***	0.8761	0.7993	4.79%
	25	0.8903	*0.8376*			4.79%
	50	0.8903	*0.8376*			4.79%
gc	10	0.6719	0.5752	0.6794	0.6629	-13.23%
	25	0.6853	***0.6784***			2.34%
	50	0.6903	0.5611			-15.36%
wbc	10	0.9753	***0.9743***	0.9750	0.9659	0.87%
	25	0.9758	*0.9743*			0.87%
	50	0.9767	*0.9743*			0.87%
hrt	10	0.8592	0.7785	0.8611	**0.7997**	-2.65%
	25	0.8647	0.7810			-2.34%
	50	0.8770	0.7744			-3.16%

* Results are provided on an aggregated generation level. That is, the fittest individual within the first 10, 25, and 50 generations is selected and evaluated on the test set simulating a scenario where the GA is stopped after the respective number of iterations. We use bold letters to denote the classifier that performs best on test data (with lower number of iterations, if performances are equal). In addition, italic letters indicate that SVMs with a combined kernel outperform standard SVM.

Results for standard SVM are given for comparison purpose. These have been computed using the grid search approach of Table 3 and selecting the model within maximum overall performance. Here, performance is defined in the sense of 10-fold CV *bca* on training data (Table 4) and bound (8) (Table 5) mimicking the behaviour of GA-1 and GA-2.

Using the algebraic bound (8) as fitness criterion, the GA-based SVM outperforms standard SVM on all considered data sets whereas it fails to find a superior model on the heart data set when using the empirical estimator. Similarly, the deviation between test performance of GA-based SVMs and standard SVMs appears more favorable for GA-2. However, differences between GA-1 and GA-2 in absolute performance values on test data are minor so that we conclude that both are appropriate fitness criteria for GA.

TABLE 5:
RESULTS FOR GA-2 BASED MODEL SELECTION *

GA-2		GA		Standard SVM		Deviation
	Gen.	Best fitness	*bca* on test	Best fitness	*bca* on test	between GA and standard SVM
ac	10	0.7957	***0.8034***	0.7782	0.7292	10.18%
	25	0.8065	*0.7401*			1.49%
	50	0.8152	*0.7344*			0.71%
gc	10	0.6712	***0.6236***	0.6441	0.5853	6.54%
	25	0.6787	0.6192			5.79%
	50	0.6922	0.5480			-6.37%
wbc	10	0.9186	***0.9694***	0.9269	0.9646	0.50%
	25	0.9457	0.9528			-1.22%
	50	0.9520	0.9444			-2.09%
hrt	10	0.7937	***0.7810***	0.6825	0.7400	5.54%
	25	0.7937	*0.7810*			5.54%
	50	0.7937	*0.7810*			5.54%

* see Table 4.

Noteworthy, for both GA-1 and GA-2 we observe a trend to overfit the data when running for a large number of generations. Due to our elitist selection the fitness increases monotonically from generation to generation. Though, selecting a model after 50 generations is always equal or inferior, in the sense of final performance achieved on test data, to selecting a model in an earlier stage of the evolutionary process. While these differences are negligible for the medical data sets the performance drop-off is serious for ac (6.9% for GA-2) and gc (11.7% GA-1). To clarify on this issue we analyse the relationship between fitness and performance on hold-out data in more detail using generalization diagrams as shown exemplary for GA-1 on ac in Fig. 4.

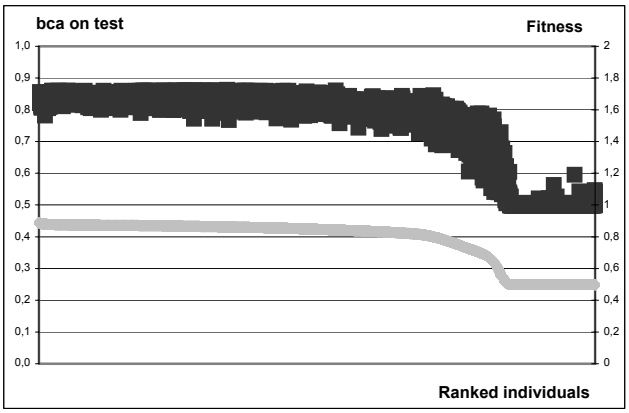


Fig. 4: Generalization diagram for GA-1 on ac showing all individual SVMs over all generations ranked by their fitness (grey squares) with according *bca* on test set (black squares). Note that fitness and test performance are scaled differently on individual axis to improve readability.

The diagram reveals that GA-1 provides excellent model selection capabilities for this particular data set. Individuals with high fitness exhibit similarly high test set performance

so that fitness based model selection will produce reliable classifiers with good generalization performance. Conducting this analysis over all data set revealed that GA-1 exceeds GA-2 in terms of correlation between fitness and generalization performance on average.

At the right side of Fig. 4 we observe a clear fitness drop-off. The test performance reaches a constant level of 0.5. This is explained by the fact, that the respective classifiers become naïve, predicting only one class for all instances. We refrained from incorporating prior knowledge into the GA, e.g. what kernel types/parameters to avoid for a given data set, range of the regularization parameter, etc., striving for a generic model selection mechanism. Equipping the algorithm with maximum flexibility allowed the construction of accurate and generalizable classifiers but at the cost that a certain amount of the derived models become futile. While extensive grid search usually leads to a number of naïve predictors as well, we analyze the ratio of naïve SVMs to overall SVMs for the GA and grid search in Fig. 5 to find that the number of ineffective models is in fact larger for the GA-based approach and GA-2 in particular.

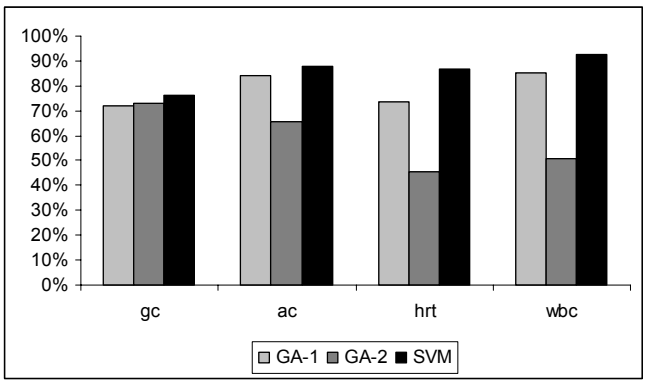


Fig. 5: Ratio of non-naïve models for GA-1, GA-2 and standard SVM.

This analysis explains our previous finding regarding the superiority of GA-1 in terms of generalization ability. While GA-1 and GA-2 are both promising for the task of selecting one best model out of a large candidate list, GA-1 is superior for steering the process of SVM kernel construction leading to a larger number of suitable classifiers on average.

VI. Conclusions

In this paper, we developed a GA-based approach to automate the task of model selection for the SVM classifier. This involved the construction of a combined kernel and the tuning of all resulting parameters. Requiring an appropriate fitness criterion for the GA we evaluated the well known CV performance on training data as an empirical model selection criterion. On the other hand, the minimization of algebraic bounds is well established within the SVM community facilitating model selection without re-sampling and re-training. Comparing these two model selection measures in the context of GA-based SVM parameterization we found that both are appropriate to choose a classifier that will generalize well to unknown data. However, model selection aims at finding only one classifier and from a GA perspective the empirical estimate of generalization performance is the better choice to guide the evolutionary process of SVM construction. Using the support vector bound (8) as fitness criterion delivered a larger number of futile classifiers decreasing reliability on average. To overcome this shortcoming, partly present in GA-1 as well, we will develop GAs that incorporate prior knowledge regarding SVM kernels and parameters, e.g. tuning heuristics like [16], in further research. However, such approaches will come at the cost of sacrificing generality and dissociate from the appealing vision of automatic model selection.

References

[1] N. E. Ayat, M. Cheriet, and C. Y. Suen, "Automatic model selection for the optimization of SVM kernels," *Pattern Recognition*, vol. 38, pp. 1733-1745, 2005.

[2] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A Training Algorithm for Optimal Margin Classifiers," *Proc. of the 5th Annual Workshop on Computational Learning Theory*, Pittsburgh, Pennsylvania, USA, 1992.

[3] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik, "Gene Selection for Cancer Classification using Support Vector Machines," *Machine Learning*, vol. 46, pp. 389-422, 2002.

[4] J. Zhang and Y. Liu, "Cervical Cancer Detection Using SVM Based Feature Screening," *Proc. of the 7th Medical Image Computing and Computer-Assisted Intervention*, Saint-Malo, France, 2004.

[5] T. Joachims, "Text Categorization with Support Vector Machines: Learning with Many Relevant Features," *Proc. of the 10th European Conf. on Machine Learning*, Chemnitz, Germany, 1998.

[6] G. Guo, S. Z. Li, and K. L. Chan, "Support vector machines for face recognition," *Image and Vision Computing*, vol. 19, pp. 631-638, 2001.

[7] B. Baesens, T. Van Gestel, S. Viaene, M. Stepanova, J. Suykens, and J. Vanthienen, "Benchmarking state-of-the-art classification algorithms for credit scoring," *Journal of the Operational Research Society*, vol. 54, pp. 627-635, 2003.

[8] S. Viaene, B. Baesens, T. Van Gestel, J. A. K. Suykens, D. Van den Poel, J. Vanthienen, B. De Moor, and G. Dedene, "Knowledge discovery in a direct marketing case using least squares support vector machines," *International Journal of Intelligent Systems*, vol. 16, pp. 1023-1036, 2001.

[9] N. Cristianini and J. Shawe-Taylor, *An introduction to support vector machines and other kernel-based learning methods*. Cambridge: Cambridge University Press, 2000.

[10] V. N. Vapnik, *Estimation of Dependences Based on Empirical Data*. New York: Springer, 1982.

[11] V. N. Vapnik, *The Nature of Statistical Learning Theory*. New York: Springer, 1995.

[12] C. J. C. Burges, "A Tutorial on Support Vector Machines for Pattern Recognition," *Data Mining and Knowledge Discovery*, vol. 2, pp. 121-167, 1998.

[13] H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, and C. Watkins, "Text Classification using String Kernels," *Journal of Machine Learning Research*, vol. 2, pp. 419-444, 2002.

[14] A. Lunts and V. Brailovskiy, "Evaluation of attributes obtained in statistical decision rules," *Engineering Cybernetics*, vol. 3, pp. 98-109, 1967.

[15] T. van Gestel, J. A. K. Suykens, B. Baesens, S. Viaene, J. Vanthienen, G. Dedene, B. de Moor, and J. Vandewalle, "Benchmarking Least Squares Support Vector Machine Classifiers," *Machine Learning*, vol. 54, pp. 5-32, 2004.

[16] S. S. Keerthi and C.-J. Lin, "Asymptotic Behaviors of Support Vector Machines with Gaussian Kernel," *Neural Computation*, vol. 15, pp. 1667-1689, 2003.

[17] T. Joachims, "Estimating the Generalization Performance of an SVM Efficiently," *Proc. of the 17th Intern. Conf. on Machine Learning*, Stanford, CA, USA, 2000.
[18] O. Chapelle and V. Vapnik, "Model selection for support vector machines," *Proc. of the 13th Annual Conference on Neural Information Processing Systems*, Denver, CO, USA, 2000.
[19] K.-M. Chung, W.-C. Kao, L.-L. Wang, and C.-J. Lin, "Radius Margin Bounds for Support Vector Machines with RBF kernel," *Neural Computation*, vol. 15, pp. 2643-2681, 2003.
[20] K. Duan, S. S. Keerthi, and A. N. Poo, "Evaluation of simple performance measures for tuning SVM hyperparameters," *Neurocomputing*, vol. 51, pp. 41-59, 2003.
[21] O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee, "Choosing Multiple Parameters for Support Vector Machines," *Machine Learning*, vol. 46, pp. 131-159, 2002.
[22] S. S. Keerthi, "Efficient tuning of SVM hyperparameters using radius/margin bound and iterative algorithms," *IEEE Transactions on Neural Networks*, vol. 13, pp. 1225-1229, 2002.
[23] S. Boughorbel, J. P. Tarel, and N. Boujemaa, "The LCCP for Optimizing Kernel Parameters for SVM," *Prof. of the 15th Intern. Conf. on Artificial Neural Networks*, Warsaw, Poland, 2005.
[24] F. Friedrichs and C. Igel, "Evolutionary Tuning of multiple SVM parameters," *Neurocomputing*, vol. 64, pp. 107-117, 2005.
[25] H.-N. Nguyen, S.-Y. Ohn, and W.-J. Choi, "Combined Kernel Function for Support Vector Machine and Learning Method Based on Evolutionary Algorithm," *Proc. of the 11th Intern. Conf. on Neural Information Processing*, Calcutta, India, 2004.
[26] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. Reading: Addison-Wesley, 1989.
[27] J. H. Holland, *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*, 6 ed. Cambridge: MIT Press, 2001.
[28] L. Li, W. Jiang, X. Li, K. L. Moser, Z. Guo, L. Du, Q. Wang, E. J. Topol, Q. Wang, and S. Ra, "A robust hybrid between genetic algorithm and support vector machine for extracting an optimal feature gene subset," *Genomics*, vol. 85, pp. 16-23, 2005.
[29] B. Samanta, "Gear fault detection using artificial neural networks and support vector machines with genetic algorithms," *Mechanical Systems and Signal Processing*, vol. 18, pp. 625-644, 2004.
[30] S.-Y. Ohn, H.-N. Nguyen, and S.-D. Chi, "Evolutionary Parameter Estimation Algorithm for Combined Kernel Function in Support Vector Machine," *Proc. of the Advanced Workshop on Content Computing*, ZhenJiang, JiangSu, China, 2004.
[31] S.-Y. Ohn, H.-N. Nguyen, D. S. Kim, and J. S. Park, "Determining optimal decision model for support vector machine by genetic algorithm," *Proc. of the 1st Intern. Symposium on Computational and Information Science*, Shanghai, China, 2004.
[32] S. Bhattacharyya, "Direct Marketing Response Models using Genetic Algorithms," *Fourth International Conference on Knowledge Discovery and Data Mining*, New York, 1998.
[33] D. Michie, D. J. Spiegelhalter, and C. C. Taylor, *Machine learning, neural and statistical classification*. New York: Horwood, 1994.
[34] D. J. Newman, S. Hettich, C. L. Blake, and C. J. Merz, "UCI Repository of machine learning databases," Department of Information and Computer Science, University of California, Irvine, CA, 1998.
[35] C.-C. Chang and C.-J. Lin, "LIBSVM - A Library for Support Vector Machines," software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm, 2001.

2006 International Joint Conference on Neural Networks
Sheraton Vancouver Wall Centre Hotel, Vancouver, BC, Canada
July 16-21, 2006

Forecasting with Computational Intelligence - An Evaluation of Support Vector Regression and Artificial Neural Networks for Time Series Prediction

Sven F. Crone, Stefan Lessmann and Swantje Pietsch

Abstract— **Recently, novel algorithms of Support Vector Regression and Neural Networks have received increasing attention in time series prediction. While they offer attractive theoretical properties, they have demonstrated only mixed results within real world application domains of particular time series structures and patterns. Commonly, time series are composed of a combination of regular patterns such as levels, trends and seasonal variations. Thus, the capability of novel methods to predict basic time series patterns is of particular relevance in evaluating their initial contribution to forecasting. This paper investigates the accuracy of competing forecasting methods of NN and SVR through an exhaustive empirical comparison of alternatively tuned candidate models on 36 artificial time series. Results obtained show that SVR and NN provide comparative accuracy and robustly outperform statistical methods on selected time series patterns.**

I. Introduction

Support Vector regression (SVR) and artificial neural networks (NN) have found increasing consideration in forecasting theory, leading to successful applications in time series and explanatory forecasting in various application domains, including business and management science [1, 2]. Methods form computational intelligence promise attractive features to business forecasting, being data driven learning machines, permitting universal approximation of arbitrary linear or nonlinear functions from examples without a priori assumptions on the model structure, often outperforming conventional statistical approaches of ARMA-, ARIMA- or exponential smoothing-methods [3]. As a consequence, significant effort has been invested in developing forecasting methods from computational intelligence [4] to reduce forecasting error.

Despite their theoretical capabilities, NN as SVR are not an established forecasting method in business practice. Recently, substantial theoretical criticism of NN has raised questions to their ability to forecast even simple time series patterns of seasonality or trends without prior data preprocessing [5]. While all novel methods must ultimately be evaluated in an objective experiment using a number of empirical time series, adequate error measures and multiple origins of evaluation [6], the fundamental questions to their ability to approximate and generalise basic time series patterns must be evaluated beforehand. Time series can generally be characterized by the combination of basic regular patterns: level, trend, season and residual errors. For trend, a variety of linear, progressive, degressive and regressive patterns are feasible. For seasonality, an additive or multiplicative combination with level and trend further determines the shape of the empirical time series. Consequently, we evaluate SVR and NN using a consistent methodology [3] in comparison to a benchmark statistical forecasting expert system using Exponential Smoothing and ARIMA-models on a set of artificially created time series derived from previous publications. We evaluate the comparative forecasting accuracy of each method on alternative error measures to avoid evaluation biases in order to reflect their ability of learning and forecasting 12 fundamental time series patterns relevant to empirical forecasting tasks under 3 levels of increasing random noise. In total, we evaluate 500,000 NN and 2,900,000 SVR candidate models for their predictive accuracy.

This paper is organized as follows: first, we provide a brief introduction to SVR and NN in forecasting time series. Section three provides an overview of the experimental design including the artificially generated time series. This is followed by the experimental results and their discussion. Conclusions are given in section 4.

II. Computational Intelligence for Forecasting

A. Multilayer Perceptrons

NNs represent a class of mathematical models originally motivated by the information processing in biological neural systems [7-10]. They promise a number of attractive features of arbitrary input–output mapping from examples without a priori assumptions on the model structure, being a semi-parametric, data driven universal approximator, which make them well suited for time series prediction tasks.

Forecasting with non-recurrent NNs may encompass prediction of a dependent variable $\hat{y}$ from lagged realizations of the predictor variable y_{t-n}, 1 or i explanatory variables x_i of metric, ordinal or nominal scale as well as lagged realizations thereof, $x_{i,t-n}$. Therefore, NNs offer large degrees of freedom towards the forecasting design, permitting explanatory or causal forecasting through relationship of the form $\hat{y} = f(x_1, x_2, ..., x_z)$, as well a

Sven F. Crone (corresponding author), Department of Management Science, Lancaster University Management School, Lancaster LA1 4YX, United Kingdom (phone +44.1524.5-92991; e-mail: sven.f.crone@crone.de).

Stefan Lessmann, Swantje Pietsch, Institute of Information Systems, University of Hamburg, 20146 Hamburg, Germany (e-mail: Lessmann@econ.uni-hamburg.de; mailing@swantje-pietsch.de).

0-7803-9490-9/06/$20.00/©2006 IEEE

$\hat{y} = f(x_1, x_2, ..., x_z)$, as well a general transfer function models and simple time series prediction. Following, we present a brief introduction to modelling ANNs for time series prediction; a general discussion is given in [11, 12]. Forecasting time series with ANN is generally based on modeling the network in analogy to an non-linear autoregressive AR(p) model [1, 13]. At a point in time t, a one-step ahead forecast $\hat{y}_{t+1}$ is computed using $p=n$ observations $y_t, y_{t-1}, \ldots, y_{t-n+1}$ from n preceding points in time *t, t-1, t-2, ..., t-n+1*, with n denoting the number of input units of the ANN. This models a time series prediction of the form

$$\hat{y}_{t+1} = f(y_t, y_{t-1}, ..., y_{t-n+1}). \quad (1)$$

In this study, a special class of NN, the well researched multilayer Perceptron (MLP) is applied. MLPs are hetero-associative, feed forward neural network which are typically composed of several layers of nodes with nonlinear signal processing [14] and trained by a derivative of the back propagation algorithm [14]. Applying a standard summation as the input unction and using an arbitrary nonlinear activation a MLP with a single layer of hidden nodes may be written as [15]

$$\hat{y}_t = f_{act}\left(w_{co} + \sum_h w_{ho} f_{act}\left(w_{ch} + \sum_i w_{ih} y_{t-j}\right)\right). \quad (2)$$

The architecture of a MLP is displayed in figure 1.

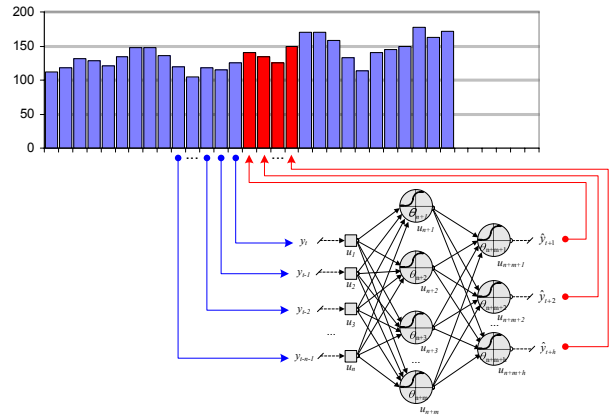


Fig. 1. Autoregressive MLP application to time series forecasting with a MLP of arbitrary topology, using n input neurons for observations in *t, t-1, t-2, ..., t-n-1*, m hidden units, h output units for time periods *t+1, t+2, ..., t+h* and a two layers of trainable weights. The bias node is not displayed.

For a time series forecasting problem, training data is provided in form of vectors of $n=p$ time lagged observations [1, 8] in form of a sliding window over the time series observations [16]. The task of the MLP is to model the underlying generator of the data during training, so that a valid forecast is made when the trained ANN network is subsequently presented with a new input vector value [5].

Although the network paradigm of MLP offers extensive degrees of freedom in modeling for prediction tasks, it must be noted that they do not utilize recurrent feedback of their own output or previous errors and are therefore incapable of modeling moving average processes required to approximate data generating process of seasonal ARMA or ARIMA $(p,d,q)(P,D,Q)_s$ structure. For topologies without hidden nonlinear nodes, MLPs are equivalent to a linear AR(p) models [9]. For a detailed discussion of these issues and the ability of NN to forecast univariate time series see [1].

B. Support Vector Regression

Recently, SVR has been applied to time series prediction. SVR represents another method from computational intelligence related to NN and methodically based upon the statistical learning theory developed by Vapnik [2, 17, 18]. In this study we consider the ε-SVR, which approximates a function $f(\mathbf{x})$ to provide a maximum of ε-deviation from all target values y_i in the training dataset $((\mathbf{x}_1, y_1), ..., (\mathbf{x}_\ell, y_\ell)) \subseteq (\mathbf{X} \times Y)^\ell$ and is as flat as possible [19-21]. Unlike the NN, the training problem of the SVR is a convex optimization problem without local minima [2] For a simple linear problem this function is of the form $f(x) = \langle \mathbf{w}, \mathbf{x} \rangle + b$ with $\mathbf{w} \in \mathbf{X}, b \in \Re$ and $\langle \mathbf{w}, \mathbf{x} \rangle$ denotes the dot product in the space of the input patterns $\mathbf{x}$ [17, 19, 22]. The support vectors are those data points used to describe the searched function [23]. In removing those training patterns which are not support vectors, the solution is unchanged and hence a fast method for validation is available when the support vectors are sparse [2, 24]. As noise exists, it is useful to work with a soft margin, as known from Support Vector Machines (SVM). This is realized by slack variables $\xi_i^+, \xi_i^- \geq 0$ which extend the mathematical formulation of the convex optimization problem [2],

$$\frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{\ell}(\xi_i^+ + \xi_i^-), \quad (3)$$

which has to be minimized by $\|\mathbf{w}\|^2 = \langle \mathbf{w}, \mathbf{w} \rangle$ with the constraints $(\mathbf{w} \cdot \mathbf{x_i}) + b - y_i = \varepsilon + \xi_i^-$, $y_i - (\mathbf{w} \cdot \mathbf{x_i}) - b \leq \varepsilon + \xi_i^+$ to ensure flatness [23, 25]. The constant C determines the trade-off between flatness and the amount of outliers of the ε-tube, which is handled in this study with the ε-intensive loss function [26]

$$|\xi|_\varepsilon := \begin{cases} 0 & if \ |\xi| \leq \varepsilon \\ |\xi| - \varepsilon & otherwise \end{cases}. \quad (4)$$

For this particular cost function the Lagrange multipliers are sparse [2, 24] and only data points outside the ε-tube contribute to costs. To assure that the training data appear in the form of dot products between the vectors and to better handle the constraints, the problem is transformed to a Lagrangian formulation [2]:

$$L(\mathbf{w}, b, \xi_i, \xi_i^*) := \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{\ell}(\xi_i^+ + \xi_i^-) - \sum_{i=1}^{\ell}(\eta_i^+ \xi_i^+ + \eta_i^- \xi_i^-)$$
$$-\sum_{i=1}^{\ell}\alpha_i^+(\varepsilon + \xi_i^+ - y_i + (\mathbf{w} \cdot \mathbf{x_i}) + b) \quad (5)$$
$$-\sum_{i}^{\ell}\alpha_i^-(\varepsilon + \xi_i^- + y_i - (\mathbf{w} \cdot \mathbf{x_i}) - b).$$

This represents the precondition for nonlinear problems. Here L is the Lagrangian function and $\eta_i^\pm$ and $\alpha_i^\pm$ are positive and the Lagrange multipliers. To receive the dual optimization problem,

$$-\frac{1}{2}\sum_{i,j=1}^{\ell}\left(\alpha_i^+-\alpha_i^-\right)\left(\alpha_i^+-\alpha_j^-\right)\left(\mathbf{x_i}\cdot\mathbf{x_j}\right) \quad , \qquad -\varepsilon\sum_{i=1}^{\ell}\left(\alpha_i^++\alpha_i^-\right)+\sum_{i=1}^{\ell}y_i\left(\alpha_i^+-\alpha_i^-\right)\rightarrow \max! \tag{6}$$

with subject to $\sum_{i=1}^{\ell}\left(\alpha_i^+-\alpha_i^-\right)=0$ and $\alpha_i^+,\alpha_i^-\in[0,C]$, the partial derivatives of L with respect to the primal variables $\mathbf{w},b$ and $\xi_i^{\pm}$ are vanished and substituted to the primal function [2]. With the condition $\partial_{\xi_i^+}L=C-\alpha_i^+-\eta_i^+=0$ and $\partial_{\xi_i^-}L=C-\alpha_i^--\eta_i^-=0$ the dual variables $\xi_i^{\pm}$ can be eliminated and thus the dual optimization problem reformulated as Support Vector (SV) expansion $f(\mathbf{x}_i)=\sum_{i=1}^{\ell}\left(\alpha_i^+-\alpha_i^-\right)\left(\mathbf{x_i}\cdot\mathbf{x}_j\right)+b$, which is a linear combination of the training patterns[27]. The coefficients $\alpha_i^{\pm}$ are the parameters to be adjusted by training and $\mathbf{x_i}$ are the training patterns. The choice of the bias b gives rise to several variants [28] In this study the Karush–Kuhn–Tucker (KKT) conditions are used [2, 24, 26]. This method base on the idea that the variables $\alpha_i^{\pm}$, for those the prediction error can be determined are uniquely. This means for the ε-intensive case, to select the data dots on the margin as here the exact value of the prediction error is known and calculate for the according data dot the threshold b [26]. To guarantee stability, b is calculated for all dots on the margin and the average is used as threshold [26].

Nonlinearity can be created by nonlinear mapping ϕ the data into a high dimensional feature space F and do linear regression in this space, thus this corresponds to nonlinear regression in a low dimensional input space [2]. As mapping all data to space can easily become computationally infeasible for polynomial features of higher order and higher dimensionality [23]. To avoid this, kernel functions $k(\mathbf{x},\mathbf{x}')=\langle\phi(\mathbf{x}),\phi(\mathbf{x}')\rangle$ are used, that enable operations to be performed in the input space rather than the potentially high dimensional feature space, hence the inner product does not need to be evaluated in the feature space[29]. All kernel functions, those correspond to the inner product of some feature space, must satisfy Mercer's condition. This study uses the Gausian Radial Basis Function (RBF)

$$k(\mathbf{x},\mathbf{x}')=\exp\left(-\gamma\left|\mathbf{x}-\mathbf{x}'\right|^2\right),\gamma>0 \quad . \tag{7}$$

which represents the most commonly used kernel for regression problems [2, 26] and corresponds to minimizing the specific cost function with a regularization operator and satisfies the Mercer conditions, as any symmetric kernel function [23, 26, 28]. Finally in this study the quadratic programming problem is defined as minimize

$$\frac{1}{2}\sum_{i,j=1}^{\ell}\left(\alpha_i^+-\alpha_i^-\right)\left(\alpha_j^+-\alpha_j^-\right)k\left(\mathbf{x_i},\mathbf{x_j}\right) \quad , \qquad -\varepsilon\sum_{i=1}^{\ell}\left(\alpha_i^++\alpha_i^-\right)+\sum_{i=1}^{\ell}y_i\left(\alpha_i^+-\alpha_i^-\right) \tag{8}$$

with subject to $\sum_{i=1}^{\ell}\alpha_i^+-\alpha_i^-=0$, $\alpha_i^+,\alpha_i^-\in[0,C]$ and $i=1,\ldots,\ell$ [26]. As the RBF kernel function is used in the experiments, the output weights as well as the RBF centers and variances are adjusted by back-propagation [30].

III. Experimental Design

A. Experimental Data

In order to evaluate the ability of SVR and MLP to forecast a benchmark subset of common time series patterns, we develop a set of archetype time series derived from decomposing monthly retail sales in [16]. Time series patterns are composed of overlaying components of a general level *L* of the time series, seasonality *S* within a calendar year, trends *T* in the form of long term level shifts and random noise *E* as a remaining error component. Through combination of the regular patterns of linear, progressive, degressive or regressive trends with additive or multiplicative seasonality we derive 12 artificial time series following the patterns motivated from Pegel's classification framework, later extended by Gardner to incorporate degressive trends [31]. In particular, we create time series following a stationary pattern L+E denoted as (E), additive seasonality without trend L+S_A+E (S_A), multiplicative seasonality without trend but increasing with time L+S_M*t+E (S_M), linear trend L+T_L+E (T_L), linear trend with

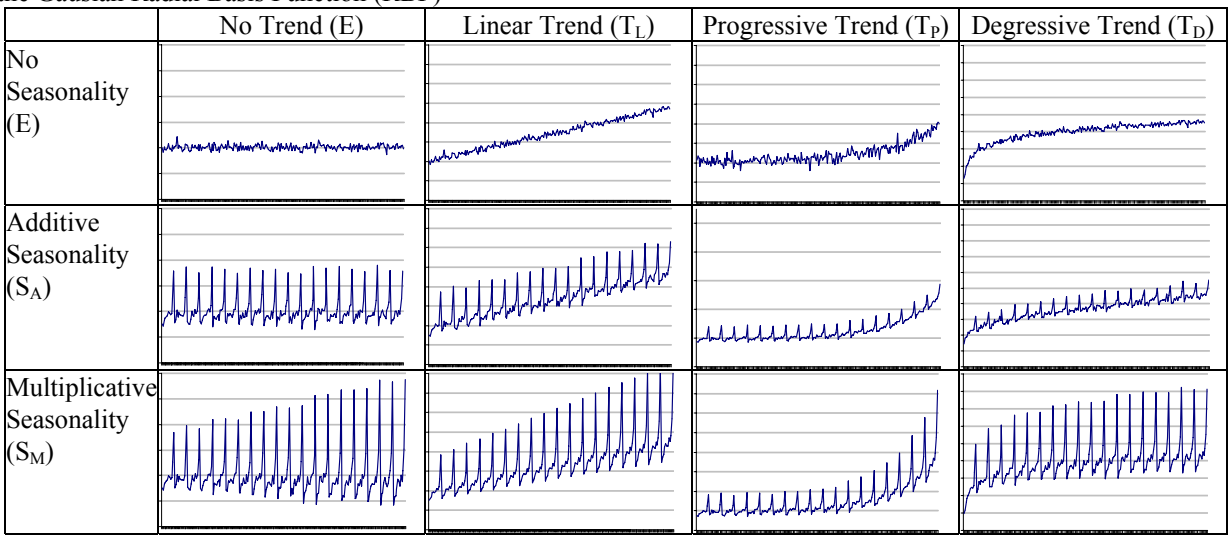


Fig. 2. Basic time series patterns of artificial time derived from the Pegels- and Gardner-classification, combining Level, Trend and Seasonality with a medium level of additive noise.

additive seasonality $L+T_L+S_A+E$ (T_LS_A) and linear trend with multiplicative seasonality depending on the level of the time series $L+T_L*S_M+E$ (T_LS_M). The functional form of these basic time series patterns is visualized in the left six quadrants of Fig. 1. In addition, we model similar combinations of degressive and progressive trend (T_P) with additive and multiplicative seasonality to T_DS_A, T_DS_M, T_PS_A and T_PS_M displayed in the six right quadrants of Fig.1.

Each time series is overlaid with tree levels of low, medium and high additive random noise $\sigma^2 = 1,\ 25,\ 100$ following a Gaussian distribution $N(0,\sigma^2)$, thereby creating a total of 36 time series of 228 monthly observations [8]. The time series may be distinguished in linear versus nonlinear patterns, with the patterns of E, T_L, S_A and T_L+S_A relating to linear model forms and all other combinations to nonlinear models. Consequently, we can subsequently analyze the experimental results of forecasting accuracy of competing methods using multiple hypotheses of varying noise and different time series structure.

Each time series is split into training set, validation set and test set using a proportion of [60%, 20%, 20%] in accordance with [32]. As the size of the test set affects the validity of the forecasting results [33], but very long time series often do not reflect empirical data availability, a test set size of 48 observations data serves as a sufficient and acceptable trade-off. For the statistical benchmark methods, which do not require the use of a validation set, both training and validation set are used for parameterization, with an identical out-of-sample test set used for all methods

B. General Experimental Setup

We determine a number of identical input variables for both NN and SVR. Each time series may be characterized by a different autoregressive lag structure and require a different number of input nodes. As a consequence, we identified suitable lag-structures for inclusion in the input vector following the approach by Lattermacher and Fuller using the linear autocorrelation function (ACF) and partial autocorrelation functions (PACF) as is common practice in ARIMA-modeling [16, 34]. In particular, we generate an input vector length using the last statistically significant PACF lag of a time series successively differenced until stationary [8, 35].

All data for NN and SVR was linearly scaled to avoid numerical difficulties and to speed up the training process [3, 8], using

$$z_t = AF_{\min} + (AF_{\max} - AF_{\min}) \cdot \frac{(x_t - x_{t\,\min})}{(x_{t\,\max} - x_{t\,\min})}, \qquad (9)$$

with z_t the scaled data used for training and $x_{t\ max}$ and $x_{t\ min}$ the maximum or minimum observed value on the training and validation set of each time series [3]. In order to avoid saturation effects close to the asymptotic limits of nonlinear activation function [-1;1] through non-stationary time series with consistent trends or seasonality, we applied an additional 50% headroom AF_{max}=0,5 and AF_{min}=-0.5, effectively scaling the data into the interval [-0.5; 0.5].

As the relative performance of each forecasting method is influenced by the selection of the evaluation criteria [16, 36, 37], we evaluate the forecasting accuracy using a set of five established accuracy measures: mean absolute error (MAE), mean absolute percentage error (MAPE), median absolute percentage error (MdAPE), Root mean squared error (RMSE) and Theil's U-statistic (TU), which are discussed in detail in [16]. Although RMSE and MAPE provide a strong bias in over-penalizing large deviations or sensitivity to the scale of the errors, their information is provided to allow comparisons with alternative studies frequently applying these inefficient error statistics. The TU statistic provides a relative accuracy measure in comparison to the accuracy of a naïve forecast using the last observation as a prediction, with values TU<1 demonstrating superior performance then a naïve method and values TU> 1 indicating inferior accuracy [31]. All error measures are calculated as mean errors across an out-of-sample test set. In addition, we calculate ordinal performance metrics. Each forecasting method is ranked by each error measure, with a 1 indicating highest performance and a 3 documenting the lowest. The means of these ranks across all time series are calculated to demonstrate relative performance robust to influences from outliers. The use of ordinal error measures based upon rank-information omits information on the distance between individual methods. As a consequence, we propose an additional distance measure, with the worst error measure setting an origin of zero percent and the optimum of e=0 setting 100%. In relation to this, the percentage distances of the other two methods were calculated. Thus, the higher the percentage of the distance the closer the method performs to the optimum [3]. These distances may be accumulated across different error measures and time series in order to further analyze the differences in accuracy of the forecasting methods.

C. Setup of Forecasting Models

The accuracy of each forecasting method is determined by its specific architecture. Both NN [2] and SVR [2] offer a large amount of degrees of freedom in customizing and parameterising models to a particular forecasting task. Thus we need to evaluate a number of candidate models to determine a suitable MLP and SVR architecture.

For the ε-SVR with RBF kernel function, the model accuracy depends on the parameters ε, C and γ [35]. We evaluate a variety of parameter combinations through a systematic grid search with exponentially growing sequences as proposed by Hsu [28, 38]. First, a coarse grid of $C=[2^{-5}, 2^{-4.5}, \ldots, 2^{15}]$, $\gamma=[2^{-23.0}, 2^{-22.5}, \ldots, 2^{0}]$ and $\varepsilon=[2^{-12}, 2^{-11.5}, \ldots, 2^{3}]$ is used. The parameter combination with highest validation accuracy is picked and its region successively analyzed applying a refined grid using an exponential reduced sequence of step sizes from 0.5, 0.05, 0.005 unto 0.0005. As a consequence, the initial grid evaluates 59,737 parameter combinations and each successive refined grid a further 8,000 parameter combinations. Using this shrinking technique we aim to reduce the total training time in considering only a subset of

free variables [39]. Of all SVR candidates, the one with the lowest error on the validation dataset was selected.

In order to determine an effective MLP topology, a set of 70 different NN topologies using 240 different parameter combinations is evaluated, resulting in 16,800 different MLP candidate models for each time series. A maximum of 30 nodes are distributed across a maximum of 3 layers of hidden nodes, evaluating every combination of hidden layer [1,…,3] and nodes [0,…,30] in steps of 2 nodes, limiting the candidates to pyramidal topologies with the number of nodes in successive hidden layers equal or smaller to the preceding ones in order to limit the design complexity [10, 32, 40]. The maximum of 30 nodes was set to reflect the number of free parameters in relation to the training patterns. All predictions are computed as iterative one-step ahead predictions *t+1*, a single output node is used. The number of input nodes is determined ex ante through the analysis of the autocorrelation structure of each time series, resulting in a total of 70 topologies for each successive variation of model parameters. For information processing within the nodes, the established hyperbolic tangent activation function TanH is applied in all hidden nodes [4, 39] and a linear activation function in the single output node [41], using a simple summation as the input function in all nodes. To allow for randomized starting points each MLP is randomly initialized 20 times using three different initializations intervals of [-0.88;0.88], [-0.55;0.55] and [0.22;0.22]. Each MLP candidate is trained for 1000 epochs using four different initial learning rates of [0.05; 0.35; 0.65; 1] which are reduced by a cooling factor of 0.99 after each epoch of presenting all data patterns in the training set to the input nodes in random order. During training, the NN with the lowest error on the validation set is saved, applying early stopping if the MSE on the validation set has not decreased for 100 epochs. The MLP candidate showing the lowest MSE validation error is selected for forecasting. Each MLP was simulated using a NN software simulator "Intelligent Forecaster" developed for large scale empirical evaluations of NN by the authors.

To serve as a benchmark, all time series are evaluated using an established expert forecasting system ForecastPro, which evaluates ARIMA-models and various forms of Exponential Smoothing-methods using an automatic model selection technique [42], allowing robust prediction of stationary, seasonal, trended and trend-seasonal time series patterns. The superior performance of the forecasting software has been demonstrated sufficiently in outperforming other software and human experts in the M3-competition [35].

IV. Experimental results

The following tables provide the results of the forecasting performance of the SVR and NN models in comparison to the statistical methods. The time series results are separated into patterns of nonlinear trends in Table 1 versus time series of linear patterns in Table 2. For each time series, we computed a total of 16,800 MLP candidates and 91,737 SVR candidates, resulting into a total evaluation of 537,600 NN models and 2,935,584 SVR models to determine suitable candidate models for each time series pattern. Although results for all error measures are provided, information on the relative performance of each method should not be derived from the biased error metrics of RMSE or MAPE.

On non-linear time series it is evident that SVR and NN significantly outperform the statistical benchmark methods applied by ForecastPro on most performance criteria. Both methods demonstrate the ability to robustly learn and extrapolate all of the provided time series patterns, stationary or instationary, without any in data preprocessing through detrending or deseasonalisation. Their general ability to forecast is documented through a TU significantly smaller than 1, indicating higher performance than a Naïve forecast and therefore their general applicability in forecasting basic time series patterns.

TABLE 1
Out Of Sample Performance For Non linear Time Series on three Noise Levels

		LOW NOISE LEVEL					MEDIUM NOISE LEVEL					HIGH NOISE LEVEL				
Type	Method	MAE	MdAPE	MAPE	TU	RMSE	MAE	MdAPE	MAPE	TU	RMSE	MAE	MdAPE	MAPE	TU	RMSE
T_E	SVR	**0.41**	**2.64**	**4.74**	**0.34**	**0.52**	**2.17**	**13.05**	56.39	**0.39**	**2.85**	5.01	**19.15**	**65.03**	0.42	6.16
	MLP	0.43	2.67	5.54	0.36	0.54	2.61	19.63	95.61	0.44	3.24	**4.70**	24.31	125.04	**0.40**	**5.87**
	Stat. M.	2.17	4.92	7.18	1.58	2.76	3.95	16.05	**49.18**	0.69	4.92	5.47	19.32	71.65	0.45	6.81
T_D	SVR	**0.42**	**0.16**	**0.21**	**0.36**	**0.54**	2.19	0.92	1.08	0.38	**2.74**	**4.09**	**1.56**	**1.98**	**0.36**	**5.22**
	MLP	0.50	0.20	0.26	0.42	0.63	**2.17**	**0.87**	**1.07**	**0.38**	2.76	4.33	1.70	2.16	0.37	5.38
	Stat. M.	1.22	0.54	0.58	1.02	1.44	3.30	1.38	1.56	0.55	3.96	4.15	1.60	2.03	0.37	5.32
T_P S_A	SVR	**0.69**	**0.22**	**0.25**	**0.02**	**0.85**	**3.43**	**1.09**	**1.23**	**0.12**	**4.40**	**5.29**	**1.83**	**2.01**	**0.17**	**6.37**
	MLP	1.17	0.23	0.35	0.05	1.79	3.77	1.18	1.38	0.13	4.70	6.31	1.94	2.32	0.20	7.88
	Stat. M.	6.18	1.21	1.52	0.23	8.28	10.26	2.45	2.79	0.35	12.80	10.81	2.87	3.13	0.35	13.67
T_P S_M	SVR	**1.98**	**0.65**	**0.86**	**0.04**	**3.32**	**4.84**	**1.80**	**1.95**	**0.09**	**7.71**	**6.62**	**2.87**	**3.73**	**0.13**	**8.83**
	MLP	2.10	0.71	0.89	0.05	3.50	6.01	2.06	2.42	0.12	9.77	7.20	3.44	3.92	0.14	9.63
	Stat. M.	6.05	1.55	1.95	0.10	8.85	18.62	3.74	6.25	0.27	24.63	11.19	4.23	4.85	0.19	14.92
T_D S_A	SVR	0.90	0.18	26.26	0.04	1.52	2.58	**0.49**	0.66	0.09	3.29	5.03	1.07	1.26	0.16	6.32
	MLP	**0.88**	**0.18**	**24.81**	**0.03**	**1.13**	2.88	0.61	0.74	0.10	3.66	5.24	1.15	1.33	0.17	6.54
	Stat. M.	1.01	0.21	25.55	0.04	1.27	**2.45**	0.57	**0.63**	**0.08**	**2.99**	**4.96**	**0.96**	**1.24**	**0.16**	**6.29**
T_D S_M	SVR	**0.71**	**0.27**	**0.37**	**0.01**	**0.94**	2.90	1.24	1.45	0.05	3.70	4.47	1.68	2.21	0.08	5.84
	MLP	1.26	0.57	0.65	0.02	1.50	3.29	1.38	1.68	0.06	4.12	4.53	1.92	2.23	0.08	**5.70**
	Stat. M.	0.98	0.36	0.49	0.02	1.31	**2.38**	**0.96**	**1.22**	**0.04**	**3.03**	**4.41**	**1.65**	**2.21**	**0.08**	5.73

TABLE 2
OUT OF SAMPLE PERFORMANCE FOR LINEAR TIME SERIES ON THREE NOISE LEVELS

		LOW NOISE LEVEL					MEDIUM NOISE LEVEL					HIGH NOISE LEVEL				
Type	Method	MAE	MdAPE	MAPE	TU	RMSE	MAE	MdAPE	MAPE	TU	RMSE	MAE	MdAPE	MAPE	TU	RMSE
	SVR	0.38	24.28	690	**0.35**	0.48	1.96	43.67	487.84	0.35	2.50	**3.80**	**47.31**	428.05	0.33	4.87
E	MLP	**0.38**	**23.91**	830	0.36	**0.48**	**1.95**	**43.37**	449.85	**0.34**	**2.49**	3.83	47.93	417.17	0.34	4.94
	Stat. M.	0.39	25.27	**607**	0.36	0.49	1.96	44.62	**414.20**	0.35	2.51	3.81	48.18	**154.52**	**0.33**	**4.85**
	SVR	0.42	0.20	0.24	0.36	0.53	2.07	0.98	1.18	0.37	2.70	4.14	1.84	2.30	0.36	5.23
T_L	MLP	0.43	0.21	0.25	0.37	0.55	2.17	1.03	1.24	0.38	2.76	4.52	2.08	2.57	0.39	5.60
	Stat. M.	**0.40**	**0.18**	**0.23**	**0.34**	**0.50**	**1.98**	**0.91**	**1.12**	**0.35**	**2.54**	**3.86**	**1.62**	**2.16**	**0.34**	**4.99**
	SVR	**0.41**	**0.35**	**0.43**	**0.02**	**0.53**	2.13	1.92	2.21	0.07	2.68	**3.89**	**2.64**	**4.23**	**0.13**	**5.16**
S_A	MLP	0.43	0.36	0.44	0.02	0.54	2.17	1.83	2.26	0.08	2.75	4.34	3.49	4.59	0.14	5.52
	Stat. M.	0.53	0.46	0.55	0.02	0.67	**2.02**	**1.77**	**2.09**	**0.07**	**2.59**	4.02	2.90	4.34	0.14	5.25
	SVR	0.58	0.53	0.65	0.01	0.73	**2.48**	2.29	**2.83**	**0.05**	**3.03**	**4.15**	**3.44**	**4.95**	**0.10**	**5.43**
S_M	MLP	0.58	0.50	0.64	0.01	0.72	2.52	**2.16**	2.86	0.05	3.16	4.81	4.18	5.79	0.10	5.95
	Stat. M.	**0.52**	**0.42**	**0.57**	**0.01**	**0.70**	2.81	2.54	3.05	0.06	3.69	5.11	4.05	6.00	0.11	6.53
T_L S_A	SVR	0.56	0.27	0.32	0.02	0.70	2.37	1.21	130.03	0.08	2.98	5.81	2.81	3.23	0.19	7.22
	MLP	0.57	0.27	0.32	0.02	0.70	2.39	1.08	132.39	0.08	2.96	5.48	2.67	3.05	0.18	6.83
	Stat. M.	**0.43**	**0.19**	**0.24**	**0.02**	**0.54**	**2.10**	**0.97**	**117.08**	**0.07**	**2.67**	**4.04**	**1.60**	**2.27**	**0.14**	**5.27**
T_L S_M	SVR	0.51	0.23	29.28	0.01	0.66	2.62	1.07	150.12	0.05	3.41	5.20	2.25	297.09	0.10	6.76
	MLP	0.50	0.23	28.84	0.01	0.63	2.61	1.15	154.00	0.05	3.36	5.31	2.42	301.56	0.10	6.70
	Stat. M.	**0.43**	**0.20**	**24.99**	**0.01**	**0.54**	**2.16**	**0.98**	**125.95**	**0.04**	**2.76**	**4.23**	**1.63**	**242.72**	**0.08**	**5.50**

SVR slightly outperform MLP on three of the six series for all noise levels, with statistical methods outperforming SVR and NN on two series with higher noise levels and MLPs showing only inconsistent performance. However, the differences between SVR and NN performance do not appear to be significant, with NNs always providing the second best performance across all series. Moreover, SVR and NN show robust performance regardless of time series pattern, while the statistical benchmark performs worse then naïve methods on selected time series. The results are largely consistent across error measures, with slight inconsistencies only for T_E and T_D S_A patterns of medium noise level, showing robustness of the solution.

While we may conclude that SVR shows great promise in forecasting basic nonlinear time series patterns, their performance on linear patterns given in Table 2 is not as dominant. For linear time series patterns, NN and the statistical benchmark methods outperform SVR on all but one time series consistently across all error measures. In particular for simple linear patterns, the established statistical methods of Exponential Smoothing and ARIMA outperform both NN and SVR. Again, all methods show superior performance to the Naïve method, documenting the general ability of all three approaches to forecast all of the 12 basic time series patterns without data preprocessing except a simple scaling technique applying headroom.

In order to derive more general results, we calculate the mean out of sample errors for each method across all time series patterns on the three levels of noise in Table 3, using only the unbiased error measures of MAE, MdAPE and TU with the best performance of a method indicated in bold.

TABLE 3
MEAN OUT OF SAMPLE PERFORMANCE ACROSS ALL TIME SERIES PATTERNS

	Low Noise			Medium Noise			High Noise		
	MAE	MdAPE	TU	MAE	MdAPE	TU	MAE	MdAPE	TU
SVR	**0,66**	**2,50**	**0,13**	**2,65**	**5,81**	**0,17**	**4,79**	**7,37**	**0,21**
MLP	0,77	2,50	0,14	2,88	6,36	0,18	5,05	8,10	0,22
Stat.M.	1,69	2,96	0,31	4,50	6,41	0,24	5,51	7,55	0,23

SVR clearly outperform statistical methods closely followed by MLP, although their enhanced performance in comparison to MLPs does not prove to be statistically different. Interestingly, the differences between forecasting methods decrease with an increasing level of noise, indicating extended complications in determining patterns.

A similar picture is derived in averaging the performance metrics further across all time series and noise levels in Table 4.

TABLE 4
MEAN PERFORMANCE ACROSS ALL TIME SERIES AND NOISE LEVELS

	MAE	MdAPE	TU
SVR	**2,70**	**5,23**	**0,17**
MLP	2,90	5,66	0,18
Statistical Methods.	3,90	5,64	0,26

Again, the results indicate the preeminent accuracy of SVR against statistical methods as well as MLPs. To extend this analysis, we compute a distance based accuracy to evaluate relative method performance for different noise levels and linear versus nonlinear patterns in Fig. 3.

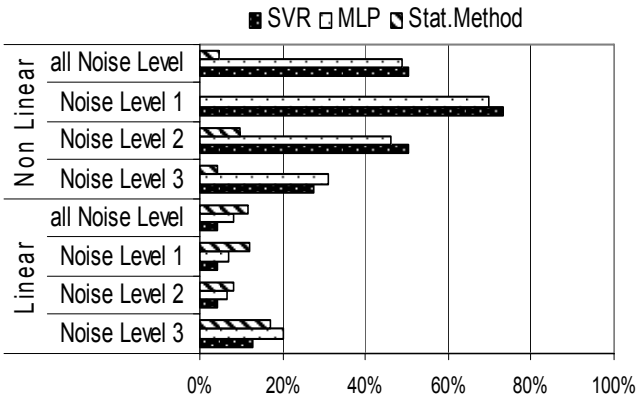


Fig. 3. The distance measure visualizes the difference of the accuracy between the single forecasting methods, with noise level 1 denoting low noise, level 2 medium noise and level 3 high noise.

The distance measures again indicate that SVR performs best on all non linear time series closely followed by NN and at a significant distance form the statistical methods. For non-linear time series on a low noise level the statistical benchmark does not appear since its forecasts were always the worst on each evaluation criterion. For linear time series the statistical methods perform best, again closely followed by the NN. It must be noted, that the level of differences in performance between the methods on linear time series is much smaller then on the nonlinear time series, in particular for noise level one and two. Therefore, the ordinal rank based accuracy measures provided in table 3 may suggest a slight bias in the evaluation of methods.

TABLE 3
ACCUMULATED RELATIVE ORDINAL MEAN RANKS

Mean Ranks	All Noise Levels	Noise Level 1	Noise Level 2	Noise Level 3
Non Linear Time Series				
SVR	**1.42**	**1.07**	**1.40**	**1.67**
MLP	1.92	1.93	2.27	1.87
Stat. Method	2.66	3.00	2.33	2.47
Linear Time Series				
SVR	2.40	2.23	2.50	2.47
MLP	1.87	2.03	**1.70**	1.87
Stat. Method	**1.73**	**1.73**	1.80	**1.67**
All Time Series				
SVR	1.91	**1.65**	**1.95**	2,07
MLP	**1.89**	1.98	1.98	**1,87**
Stat .Method	2.19	2.36	2.06	2,07

The ranking illustrate more clearly the ability of SVR to predict non linear time series, while their performance deteriorates for linear time series. Statistical methods perform best on linear time series and worst on nonlinear patterns. NN perform second best on both types of time series, always coming in close second place also with regard to other non-ordinal error measures. In summarizing over all time series, NN show the best performance, allowing valid and reliable forecasting of linear as well as nonlinear time series patterns.

V. CONCLUSIONS

We analyze the performance of competing forecasting methods of SVR and MLP from computational intelligence versus established benchmarks of univariate statistical forecasting methods. In order to derive the general ability of SVR and MLP to predict the most common time series patterns, we combine various forms of seasonal and trended time series patterns to create a benchmark dataset of 36 time series, consisting of 12 basic patterns overlaid with three levels of noise. In order to facilitate future comparisons, all time series are published at the website www.neural-forecasting.com.

The results are evaluated using five established error measures on out-of-sample accuracy. He experiments clearly indicate the ability of SVR as well as MLP to robustly forecast various forms of stationary, trended, seasonal and trend-seasonal time series without prior detrending or deseasonalisation of the data. SVR and MLPs demonstrate preeminent accuracy in comparison to statistical methods on non linear time series patterns. While statistical methods outperform SVR and NN on basic linear patterns, the differences in accuracy are not substantial. Therefore, SVR show a generally superior forecasting performance closely followed by MLP on mean accuracy measures, and MLP showing a robust forecasting accuracy using an evaluation on rank based accuracy measures.

Similar to other empirical studies, we do not attempt to demonstrate a general superiority of a particular time series method for all potential forecasting applications and time series. However, in the light of recent criticism that NN are incapable of forecasting even basic time series patterns, we provide a strong indication that MLP as well as SVR may indeed be applied successfully for time series prediction of various trend-seasonal time series without prior data analysis and iterative data preprocessing. Moreover, both SVR and MLPs validate their semi-parametric ability to learn an adequate model form and the corresponding parameters directly from the presented data, avoiding issues of conventional model selection of statistical forecasting methods. However, this evaluation has certain limitations. Even if a substantial variety of SVR parameters and NN architectures was evaluated, the evaluation took only a single methodology into consideration, which was based upon a refined simple grid search and a linearly motivated estimation of adequate lag structures. Also, not all potential NN architectures, activation functions or SVR kernel functions were evaluated. In particular, recurrent NN which are theoretically capable of approximating nonlinear AR(p) as well as nonlinear ARIMA(p,d,q)-processes should be evaluated. It may be possible, that alternative SVR and NN models with better forecasting accuracy or robustness exit even for the time series in question. In particular, the reduced SVR performance on the linear time series may be

attributed to the use of an RBF kernel function, which would further support the need for extended experimentation.

For future evaluations we also seek to extend the experiments to linear and polynomial kernel functions and to analyze the resulting forecasting accuracy with regard to the increased complexity of the modeling process. In addition, we need to extend our evaluation towards multiple time origins trough different sampling proportions, multiple step ahead forecasts and different forecasting horizons as well as empirical time series of a given application domain, in order to assure a valid and reliable evaluation on the ability of SVR and MLP to enhance future forecasting research and practice.

REFERENCES

[1] G. Zhang, B. E. Patuwo, and M. Y. Hu, "Forecasting with artificial neural networks: The state of the art," *International Journal of Forecasting*, pp. 35-62, 1998.

[2] A. J. Smola; and B. Schölkopf, "A Tutorial on Support Vector Regression," Australian National University / Max-Planck-Institut für biologische Kypernetik, Canberra / Thübingen 2003.

[3] K.-P. Liao; and R. Fildes, "The accuracy of a procedural approach to specifying feedforward neural networks for forecasting," *Computers & Operations Research*, pp. 2121-2169, 2005.

[4] G. Zhang, "Linear and Nonlinear Time Series Forecasting with Artifical Neural Networks," vol. Doctor of Philosophy: Kent State Graduate School of Management, 1998, pp. 152.

[5] G. P. Zhang and M. Qi, "Neural network forecasting for seasonal and trend time series," *European Journal Of Operational Research*, vol. 160, pp. 501-514, 2005.

[6] L. J. Tashman, "Out-of-sample tests of forecasting accuracy: an analysis and review," *International Journal of Forecasting*, vol. 16, pp. 437-450, 2000.

[7] A. Zell, *Simulation neuronaler Netze*, vol. 1. Aufl. Bonn: Addison - Wesley Verlag, 1994.

[8] S. F. Crone, "Stepwise Selection of Artificial Neural Networks Models for Time Series Prediction," University of Lancaster, Lancaster (UK) 2004.

[9] V. N. Vapnik, "An Overview of Statistical Learning Theory," *IEE Transactions on Neural Networks*, pp. 988-999, 1999.

[10] R. Callan, *Neuronale Netze im Klartext*. München: Pearson Studium, 2003.

[11] C. M. Bishop, *Neural networks for pattern recognition*. Oxford New York: Clarendon Press;Oxford University Press, 1995.

[12] S. S. Haykin, *Neural networks: a comprehensive foundation*, 2nd ed. Upper Saddle River, N.J.: Prentice Hall, 1999.

[13] A. Lapedes and R. Farber, "How neural nets work," in *Neural Information Processing Systems*, D. Z. Anderson, Ed. New York: American Institute of Physics, 1988, pp. 442-456.

[14] S. D. Balkin; and J. K. Ord, "Automatic neural network modelling for univariate time series," *International Journal of Forecasting*, pp. 509-515, 2000.

[15] J. V. Hansen; and R. D. Nelson, "Neural Networks and Traditional Time Series Methods: A Synergistic Combination in State Economic Forecasts," *IEEE TRANSACTIONS ON NEURAL NETWORKS*, vol. 8, 1997.

[16] S. Makridakis, S. C. Wheelwright, and R. J. Hyndman, *Forecasting Methods and Applications*, 3rd Edition ed. New York: John Wiley & Sons, 1998.

[17] N. Cristianini; and J. Shawe-Taylor, *An introduction to Support Vector Machines and other kernel-based learning methods*. Cambridge: Cambridge University Press, 2000.

[18] M. Anthony; and N. Biggs, *Computational Learning*. Cambridge: Cambridge University Press, 1992.

[19] R. Stahlbock; and S. Lessmann, "Potential von Support Vektor Maschinen im analytischen Customer Relationship Management," Universität Hamburg, Hamburg, Arbeitspapier 2004.

[20] M. Welling, "Support Vector Regression," Department of Computer Science, University of Toronto, Toronto (Kanada) 2004.

[21] J. Bi; and K. P. Bennett, "A Geometric Approach to Support Vector Regression," Department of Mathematical Sciences, Rensselaer Polytechnic Institute, Troy, NY 12180, New York 2003.

[22] B. Schölkopf, *Support Vektor Learning*. Berlin: GMD - Forschungszentrum Informationstechnik, 1997.

[23] S. R. Gunn, "Support Vector Machines for Classification and Regression," Faculty of Engineering, Science and Mathematics, School of Electronics and Computer Science, University of Southampton 1998.

[24] C. J. C. Burges, "A Tutorial on Support Vector Machines for Pattern Recognition," in *Data Mining and Knowledge Discovery*, U. Fayyad, Ed. Boston: Kluwer Academic Publishers, 1998, pp. 121–167.

[25] A. Smola, "Regression Estimation with Support Vector Learning Machines," Technische Universität München, 1996.

[26] K.-R. Müller, A. J. Smola, G. Rätsch, B. Schölkopf, J. Kohlmorgen, and V. Vapnik, "Predicting Time Series with Support Vector Machines," in *Advances in Kernel Methods — Support Vector Learning*, B. Schölkopf, C. J. C. Burges, and A. J. Smola, Eds. Cambridge: MIT Press, 1999, pp. 243-254.

[27] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," presented at Annual Conference on Computational Learning Theory, Pittsburgh (U.S.A.), 1992.

[28] C.-C. Chang; and C.-J. Lin, "LIBSVM: a Library for Support Vector Machines," National Science Council of Taiwan, Taiwan 2005.

[29] C.-H. Wu, C.-C. Wei, D.-C. Su, M.-H. Chang, and J.-M. Ho, "Travel Time Prediction with Support Vector Regression," Institute of Information Science, Academia Sinica, Taipei, Taiwan 2003.

[30] G. P. Zhang; and M. Qi, "Computing, Artificial Intelligence and Information Technology - Neural network forecasting for seasonal and trend time series," *European Journal of Operation Research*, pp. 501 - 514, 2003.

[31] S. Pietsch, "Computational Intelligence zur Absatzprognose - Eine Evaluation von Künstlichen Neuronalen Netzen und Support Vector Regression zur Zeitreihenprognose," in *Institut für Wirtschaftsinformatik*, vol. Diplomarbeit. Hamburg: Universität Hamburg, 2006.

[32] G. P. Zhang, B. E. Patuwo, and M. Y. Hu, "A simulation study of artificial neural networks for nonlinear time-series forecasting," *Computers & Operations Research*, pp. 381-396, 2001.

[33] G. E. P. Box and G. M. Jenkins, *Time series analysis: forecasting and control*. San Francisco: Holden-Day, 1970.

[34] R. Schlittgen; and B. H. J. Streitberg, *Zeitreihenanalyse*, 8. Auflage ed. München; Wien: Oldenburg: Oldenburg Verlag, 1999.

[35] C.-W. Hsu, C.-C. Chang, and C.-J. Lin, "A Practical Guide to Support Vector Classification," Department of Computer Science and Information Engineering - National Tawain University, Taipei (Taiwan) 2003.

[36] J. S. Armstrong; and F. Collopy, "Error Measures For Generalizing About Forecasting Methods: Empirical Comparisons," *International Journal of Forecasting*, pp. 69-80, 1992.

[37] K.-W. Hansmann, *Kurzlehrbuch Prognoseverfahren*. Wiesbaden: Gabler, 1983.

[38] C.-W. Hsu; and C.-J. Lin, "A Comparison of Methods for Multiclass Support Vector Machines," presented at IEEE Transactions on Neural Networks, 2002.

[39] S. F. Crone, H. Kausch, and D. Preßmar, "Prediction of the CATS benchmark using a Business Forecasting Approach to Multilayer Perceptron Modelling," presented at IJCNN'04, Budapest (Hungary), 2004.

[40] J. Faraway; and C. Chatfield, "Time Series Forecasting with Neural Networks: A Case Study," University of Bath, Research Report 1995.

[41] R. L. Goodrich, "The Forecast Pro methodology," *International Journal of Forecasting*, vol. 16, pp. 533-535, 2000.

[42] S. Makridakis and M. Hibon, "The M3-Competition: results, conclusions and implications," *International Journal Of Forecasting*, vol. 16, pp. 451-476, 2000.

Genetically Constructed Kernels for Support Vector Machines

Stefan Lessmann[a], Robert Stahlbock[a], Sven Crone[b]

[a] University of Hamburg, Inst. of Information Systems, Von-Melle-Park 5, 20146 Hamburg, Germany
[b] Lancaster University Management School, Dept. of Management Science, Lancaster, LA1 4YX, United Kingdom

Abstract

Data mining for customer relationship management involves the task of binary classification, e.g. to distinguish between customers who are likely to respond to direct mail and those who are not. The support vector machine (SVM) is a powerful learning technique for this kind of problem. To obtain good classification results the selection of an appropriate kernel function is crucial for SVM. Recently, the evolutionary construction of kernels by means of meta-heuristics has been proposed to automate model selection. In this paper we consider genetic algorithms (GA) to generate SVM kernels in a data driven manner and investigate the potential of such hybrid algorithms with regard to classification accuracy, generalisation ability of the resulting classifier and computational efficiency. We contribute to the literature by: (1) extending current approaches for evolutionary constructed kernels; (2) investigating their adequacy in a real world business scenario; (3) considering runtime issues together with measures of classification effectiveness in a mutual framework.

1 Introduction

The support of managerial decision making in marketing applications is a common task for corporate data mining with classification playing a key role in this context [2]. The SVM [9] is a reliable classifier that has been successfully applied

to marketing related decision problems, e.g. [1; 10]. Like other learning algorithms such as neural networks, the SVM algorithm offers some degrees of freedoms that have to be determined within the data mining process. The selection of suitable parameters is crucial for effective classification. Therefore, we propose a data driven heuristic to determine the SVM parameters without manual intervention.

The remainder of this paper is organised as follows: Following a brief introduction to SVM theory we present our combination of GA and SVM (GA-SVM) in Section 3. The potential of GA-SVM is evaluated in a real world scenario of direct marketing in Section 4. Conclusions are given in Section 5.

2 Support Vector Machines

The SVM is a supervised learning machine to solve linear and non-linear classification problems. Given a training set $S = \{\boldsymbol{x}_i; y_i\}_{i=1}^{m}$ where $\boldsymbol{x}_i$ is a n-dimensional real vector and $y_i \in \{-1,+1\}$ its corresponding class label, the task of classification is to learn a mapping $\boldsymbol{x}_i \mapsto y_i$ from *S*, that allows the classification of new examples with unknown class membership.

The SVM is a linear classifier of the form

$$y(\boldsymbol{x}) = sgn(\boldsymbol{w} \cdot \boldsymbol{x} + b), \tag{1}$$

which strives to maximise the margin of separation between the two classes [9]. The parameters $\boldsymbol{w}$ and b realising such a maximal margin hyperplane can be found by solving a quadratic optimisation problem with inequality constraints; e.g. [3].

In order to derive more general, non-linear decision surfaces SVMs implement the idea to map the input data into a high-dimensional feature space via an a priori chosen non-linear mapping function. Due to the fact, that the SVM optimisation problem contains the input patterns only as dot products, such a mapping can be accomplished implicitly by introducing a kernel function [3; 9]

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \Phi(\boldsymbol{x}_i) \cdot \Phi(\boldsymbol{x}_j). \tag{2}$$

Beside the selection of an appropriate kernel and its corresponding kernel parameters, see Section 3, the SVM classifier offers one additional regularisation parameter *C* which controls the trade off between maximising the margin of separation and classifying the training set without error.

3 Genetic algorithms for SVM model selection

The classification performance of SVM depends heavily on the choice of a suitable kernel function and an adequate setting of the regularisation parameter *C*.

Consequently, we develop a data driven approach to determine the kernel *K* and its corresponding kernel parameters together with *C* by means of GA. Using the five basic kernels of Table 1, we construct a combined kernel function as

$$K_{poly}^{1} \otimes K_{rad}^{\alpha} \otimes K_{sig}^{\beta} \otimes K_{imq}^{\gamma} \otimes K_{anova}^{1} \quad , \tag{3}$$

with $\otimes \in \{+;\cdot\}$, where we exploit the fact that if K_1 and K_2 are kernels, $K_1 + K_2$ and $K_1 \cdot K_2$ are valid kernels as well [3].

Table 1. Basic SVM kernel functions

Polynomial kernel	$K_{poly}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \left(a(\boldsymbol{x}_i \cdot \boldsymbol{x}_j) + b\right)^c$
Radial kernel	$K_{rad}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-a\left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\|^2\right)$
Sigmoidal kernel	$K_{sig}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \tanh\left(a(\boldsymbol{x}_i \cdot \boldsymbol{x}_j) + b\right)$
Inverse multi-quadratic kernel	$K_{imq}(\boldsymbol{x}_i, \boldsymbol{x}_j) = 1/\sqrt{\left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\|^2 + b^2}$
Anova kernel	$K_{anova}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \left(\sum_j \exp\left(-a\left(\boldsymbol{x}_i - \boldsymbol{x}_j\right)\right)^2\right)^c$

To encode (3) into a structure suitable for GA based optimisation we use five integer genes for the kernel exponents in (3), four binary genes for the kernel combination operator $\otimes$ and sixteen real-valued genes for the specific kernel parameters (three per kernel) as well as the regularisation parameter *C*. The complete structure is given in Fig. 1. This coding is inspired by [7] and extends their approach to five kernels and the inclusion of C into the GA based optimisation.

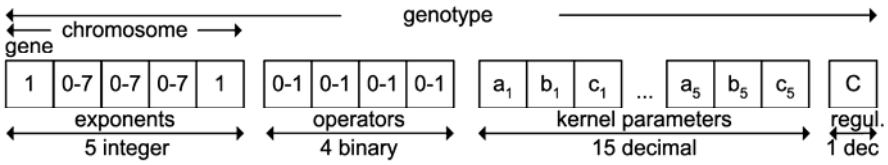


Fig. 1. Structure of the genotype for SVM kernel construction

The GA is implemented in accordance with [8] and utilises a uniform crossover for the five kernel exponent genes. That is, all genes between two random points within this string are interchanged between two genotypes representing parents for the resulting two new genotypes. The mutation operator is implemented as a simple bit swap for the four kernel combination genes and a random increment or decrement for all integer and real value genes. Crossover and mutation probabilities have been determined through pre-tests to 0.7 and 0.3 respectively.

4 Empirical evaluation

4.1 Experimental setup

The simulation experiment aims at comparing genetically constructed SVM with conventional ones to assess capabilities of GA to support SVM model selection.

We consider the case of repeat purchase modelling in a direct marketing setting, see e.g. [1; 10], using real world data from a German publishing house. The data set consists of 300,000 customer records that have been selected for a past mailing campaign to cross-sell an additional magazine subscription to customers that have subscribed to at least one periodical. Each customer is described by a 28-dimensional vector of 9 numerical and 19 categorical attributes describing transactional and demographic customer properties. The number of subscriptions sold in this campaign is given with 4,019, leading to a response rate of 1.35% which is deemed to be representative for the application domain. An additional target variable indicates the class membership of each customer (class 1 for subscribers and class -1 for non subscribers) facilitating the application of supervised learning algorithms to model a relationship between customer attributes and likelihood of responding to direct mail.

Classifiers are evaluated applying a hold-out method of three disjoint datasets to control over-fitting and for out-of-sample evaluation. While training data is used for learning, i.e. determining the decision variables $\boldsymbol{w}$ and b, see (1), a validation set is used to steer the GA. That is, a classifier's performance on the validation set represents its fitness and is used to select items for the mating pool within the GA [4]. The trained and selected classifiers are finally tested on an unknown hold-out set to evaluate their generalisation ability on unknown data.

In order to assure computational feasibility and with regard to the vast imbalance between class 1 and class -1 membership within our data set, we apply an undersampling approach [11] to obtain a training and validation data set of 4,144 and 2,070 records respectively with equal class distributions. The test set consists of 65,000 records containing 912 class 1 customers, reflecting the original unequal distribution of the target variable.

4.2 Experimental results

In order to deliver good results GA usually require a large population size that ensures sufficient variability within the elements in the gene pool [8]. For GA-SVM we select a population size of 50 and monitor the progress in classification quality for 15 generations. Thus, 750 individual SVMs with genetic kernel are constructed on the training set, assessed on the validation set and finally evaluated on the test set. Since the skewed class distribution of the target variable prohibits the application of standard performance metrics of classification accuracy [11], we used the G-metric instead [6]. Striving to maximise the class individual accuracies while keeping them balanced the G-metric is calculated as the geometric mean between

class individual accuracies. Consequently, higher values indicate improved predictive accuracy.

Results at the generation level are given in Table 2 where each value is calculated on the basis of the 50 individual GA-SVM classifiers within a generation.

Table 2. Results of GA-SVM at the generation level over 15 generations

	Mean runtime per SVM [min]		SVM performance by means of G-metric on training set		validation set		test set	
Generation	mean	std.dev.	mean	std.dev.	mean	std.dev.	mean	std.dev.
0	91.3	53.4	0.596	0.306	0.544	0.277	0.444	0.225
1	71.2	37.0	0.731	0.158	0.661	0.145	0.534	0.111
2	78.1	38.8	0.687	0.236	0.633	0.215	0.496	0.168
3	77.8	27.9	0.754	0.158	0.685	0.142	0.528	0.110
4	79.6	31.1	0.736	0.192	0.668	0.172	0.516	0.132
5	76.0	27.8	0.759	0.158	0.684	0.142	0.527	0.110
6	68.8	16.6	0.786	0.025	0.713	0.019	0.549	0.013
7	77.3	31.9	0.785	0.030	0.714	0.015	0.547	0.012
8	67.8	22.8	0.775	0.114	0.703	0.102	0.537	0.078
9	65.1	21.7	0.768	0.115	0.696	0.105	0.539	0.079
10	67.8	25.0	0.784	0.034	0.711	0.027	0.552	0.012
11	64.2	11.2	0.795	0.008	0.721	0.012	0.551	0.009
12	62.2	12.5	0.796	0.008	0.720	0.015	0.552	0.009
13	59.6	12.5	0.791	0.014	0.716	0.019	0.553	0.010
14	59.4	12.6	0.789	0.014	0.720	0.015	0.553	0.008

Our results show a generally increasing average performance from generation to generation over all data sets. However, vast improvements are obtained only when moving from generation 0 to 1, indicating that a saturation level is reached early in the evolutionary process. In fact, while a oneway analysis of variance confirmed a highly significantly difference in mean performance over all data sets at the 0.001 level, a Tukey post hoc test revealed that only the generations 0 and 2 differ from the remaining ones significantly at the 0.01 level.

The decrease in standard deviation is more explicit and illustrates a higher similarity within the gene pool. Interestingly, the average runtimes decrease tremendously, meaning that the high quality kernels of later generations are also computationally more efficient. The best kernel was found in generation 14 with a test set G-value of 0.585 incorporating all base kernels but the anova kernel.

To compare our approach with standard SVM we calculate solutions for the radial and polynomial SVM classifier, conducting an extensive grid search [5] in the range $\log(C) = \{-4; 4\}$ and $\log(a) = \{-4; 4\}$ with a step size of one for the radial kernel and $\log(C) = \{-2; 3\}$, $\log(a) = \{-2; -1\}$, $b = \{0; 1\}$, $c = \{2; 7\}$ for the polynomial kernel to obtain and average G-value of $G_{radial} = (0.70; 0.58; 0.53)$ and $G_{polynomial} = (0.71; 0.65; 0.54)$ on training, validation and test sets. As expected, the higher flexibility of the combined kernel in GA-SVM allows a purer separation of the training set. Regarding generalisation, GA-SVM consistently outperforms classical SVM in later generations, providing superior results on the validation set from generation 3 and on the test set from generation 10 onwards.

5 Conclusions

We investigated the potential of SVMs with GA-optimised kernel functions in a real world scenario of corporate decision making in marketing. Solving more than 750 evolutionary constructed SVMs, the GA proved to be a promising tool for kernel construction, enhancing the predictive power of the resulting classifier. However, the vastly increased computational cost might be the main obstacle for practical applications. Most radial SVMs needed less than a minute to construct a solution and the runtime of polynomial SVMs ranged from 12 to 60 minutes. In contrast, we observed average GA-SVM runtimes of 60 to 90 minutes.

Since the task of model selection shifts from setting SVM parameters to determining the parameters of the utilised search heuristic, the proposed GA is a promising candidate for SVM tuning, offering only four degrees of freedom on its own (crossover and mutation probabilities, population size, termination criterion e.g. number of generations).

Further research involves the application of GA-SVM to other data sets as well as a detailed analysis and comparison of the constructed kernels per generation.

References

[1] Baesens B, Viaene S, Van den Poel D, Vanthienen J, Dedene G (2002) Bayesian neural network learning for repeat purchase modelling in direct marketing. European Journal of Operational Research 138(1):191-211

[2] Berry MJA, Linoff G (2004) Data mining techniques: for marketing, sales and customer relationship management, 2. edn. Wiley, New York

[3] Cristianini N, Shawe-Taylor J (2000) An introduction to support vector machines and other kernel-based learning methods. Cambridge University Press, Cambridge

[4] Goldberg DE (1989) Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley, Reading

[5] Keerthi SS, Lin C-J (2003) Asymptotic Behaviours of Support Vector Machines with Gaussian Kernel. Neural Computation 15(7):1667-1689

[6] Kubat M, Holte RC, Matwin S (1998) Machine Learning for the Detection of Oil Spills in Satellite Radar Images. Machine Learning 30(2-3):195-215

[7] Nguyen H-N, Ohn S-Y, Choi W-J (2004) Combined Kernel Function for Support Vector Machine and Learning Method Based on Evolutionary Algorithm. In: Pal NR, Kasabov N, Mudi RK (eds) Proc. of the 11th Intern. Conf. on Neural Information Processing, Calcutta, India, pp 1273-1278

[8] Stahlbock R (2002) Evolutionäre Entwicklung künstlicher neuronaler Netze zur Lösung betriebswirtschaftlicher Klassifikationsprobleme. WiKu, Berlin

[9] Vapnik VN (1995) The Nature of Statistical Learning Theory. Springer, New York

[10] Viaene S, Baesens B, Van Gestel T, Suykens JAK, Van den Poel D, Vanthienen J, De Moor B, Dedene G (2001) Knowledge discovery in a direct marketing case using least squares support vector machines. International Journal of Intelligent Systems 16(9):1023-1036

[11] Weiss GM (2004) Mining with rarity: a unifying framework. ACM SIGKDD Explorations Newsletter 6(1):7-19

Support Vektor Klassifikatoren im analytischen Kundenbeziehungsmanagement

Stefan Lessmann, Robert Stahlbock
Institut für Wirtschaftsinformatik, Universität Hamburg
Von-Melle-Park 5, 20146 Hamburg
[lessmann, stahlboc]@econ.uni-hamburg.de

Abstract: Eine gezielte Auswertung kundenorientierter Datenbestände mittels Data Mining ist ein wichtiger Teilbereich des analytischen Kundenbeziehungsmanagement. Prognostische Fragestellungen werden dabei häufig als Klassifikationsproblem formuliert, so dass entsprechenden Lösungsverfahren wie Künstlichen Neuronalen Netzen oder Entscheidungsbäumen eine große Bedeutung zukommt. Eines der leistungsfähigsten bekannten Klassifikationsverfahren, die sog. Support Vektor Maschine, wurde, trotz zahlreicher viel versprechender Ergebnisse in verwandten Anwendungsgebieten, bisher kaum für das analytische Kundenbeziehungsmanagement in Betracht gezogen. Die vorliegende Arbeit zeigt das Potential dieser Technik und verdeutlicht, wie die besonderen Anforderungen des Anwendungsfeldes durch ein spezialisiertes Vorgehensmodell geeignet berücksichtigt werden können. Hinausgehend über die reine Klassifikationsleistung wird eine effiziente Entscheidungsunterstützung erreicht.

Schlüsselwörter: Support Vektor Maschinen, Klassifikation, analytisches Kundenbeziehungsmanagement, Data Mining

1 Einleitung

Der Begriff Kundenbeziehungsmanagement (KBM) beschreibt eine kundenorientierte Managementphilosophie, die den Aufbau und die Pflege langfristiger und profitabler Kundenbeziehungen verfolgt und für die kontinuierliche Verbesserung kundenbezogener Geschäftsprozesse einen ganzheitlichen Einsatz von Informations- und Kommunikationstechnologie vorsieht [12].

Die Struktur und Beschaffenheit der eigenen Absatzmärkte hat sich für viele Unternehmen in den vergangenen Jahren zum Teil erheblich verändert. Hierunter fallen eine durch Globalisierung verschärfte Wettbewerbssituation, abnehmende Kundenloyalität, generell erhöhte Markttransparenz und Kundenerwartungen sowie eine voranschreitende Homogenisierung von Produkten. KBM kann als Strategie zur Reaktion auf derart veränderte Umweltbedingungen verstanden werden [18]. In der beschriebenen Situation wettbewerbsintensiver und gesättigter Märkte, ist die Gewinnung neuer Kunden mit hohen Investitionen verbunden, da potentielle Neukunden zumeist von Konkurrenten abgeworben werden müssen. Die Bestrebung Beziehungen zu bestehenden Kunden langfristig zu erhalten und profitabel auszugestalten ergibt sich als logische Konsequenz.

Die Methoden und Werkzeuge, die den Aufbau der hierzu erforderlichen Wissensbasis (Kundenpräferenzen und -potentiale, bevorzugte Kommunikationskanäle, etc.) unterstützen, werden unter dem Begriff analytisches Kundenbeziehungsmanagement

(aKBM) zusammengefasst [12]. Hierunter fällt insbesondere auch eine gezielte Auswertung kundenorientierter Datenbestände mittels Data Mining [13]. Es zeigt sich, dass viele der typischen Data Mining Fragestellungen, die im Rahmen von aKBM betrachtet werden (Zielgruppenselektion, Cross-/Up-Selling, Stornoanalysen, Betrugserkennung und andere [13]), als Klassifikationsproblem formuliert werden können, so dass entsprechenden Lösungsverfahren eine große Bedeutung zukommt.

Während Entscheidungsbaumverfahren, Neuronalen Netze und Methoden der multivariaten Statistik zur Lösung von aKBM-Anwendungen weit verbreitet sind, wurden Support Vektor Maschinen (SVM) [26] bisher kaum eingesetzt. SVM lieferten in anderen Domänen bereits viel versprechende Ergebnisse, so dass deren Potential für aKBM-Fragestellungen in der vorliegenden Arbeit untersucht und ein an das Anwendungsgebiet angepasstes Vorgehensmodell entwickelt werden soll.

Dazu werden im nächsten Kapitel die Grundprinzipien von Support Vektor Klassifikatoren dargestellt. Der anschließende Teil drei verdeutlicht typische Anforderungen, die im aKBM an Data Mining Methoden zu stellen sind und zeigt, wie diese durch SVM geeignet behandelt werden können. In diesem Zusammenhang wird ein integriertes Referenzmodell zur Anwendung von SVM im aKBM entworfen. Im Teil vier werden wesentliche Erkenntnisse der Arbeit zusammengefasst.

2 Support Vektor Maschinen

SVMs gehören zu den überwacht lernenden Verfahren zur Klassifikation und wurden in ihrer ursprünglichen Form Mitte der neunziger Jahre von Vapnik und seinen Mitarbeitern vorgestellt [26].

Das Ausgangsproblem einer Klassifikationsanalyse lässt sich wie folgt beschreiben: Gegeben Sei eine Menge von Lernbeispielen S, die jeweils aus einem Merkmalsvektor $\boldsymbol{x}$ und einer diskreten Klassenzugehörigkeitsvariable y bestehen.

$$S = \{(\boldsymbol{x}_1, y_1), ..., (\boldsymbol{x}_m, y_m)\} \; ; \boldsymbol{x}_i \in X \subseteq \mathbb{R}^n; \; y_i \in \{1, 2, ..., K\} \qquad (1)$$

Ein Spezialfall ist die binäre Klassifikation mit $y_i \in \{-1, +1\} \; \forall i$.[1] Es wird unterstellt, dass ein Zusammenhang zwischen den in $\boldsymbol{x}$ zusammengefassten Merkmalsausprägungen eines Objektes und seiner Klassenzugehörigkeit besteht. Dieser ist entweder unbekannt oder zu komplex, als dass er explizit modelliert werden könnte und soll daher anhand von Beispieldatensätzen durch ein Lernverfahren geschätzt werden [25]. Der in dieser Lernphase kalibrierte Klassifikator kann anschließend zur Prognose neuer Objekte mit unbekannter Klassenzugehörigkeit verwendet werden.

[1] Da der Mehrklassenfall immer auf mehrere Zwei-Klassen Klassifikationen zurückgeführt werden kann, wird im Folgenden nur noch die binäre Klassifikation betrachtet.

SVMs gehören zu den linearen Klassifikatoren, was bedeutet, dass sie die Lerndaten durch Konstruktion einer Hyperebene separieren. Die Klasse neuer Objekte kann dann durch die Entscheidungsfunktion

$$e(\boldsymbol{x}) = sgn\left(\langle \boldsymbol{w}, \boldsymbol{x} \rangle + b\right) \qquad (2)$$

geschätzt werden. Diese bestimmt den Abstand eines Objektes zu der durch $\boldsymbol{w}$ und b charakterisierten, trennenden Hyperebene und weist Objekten die Klasse +1 zu, sofern dieser Abstand größer als Null ist. Andere Objekte werden in die Klasse -1 eingeordnet. Ausgehend von der Idee, eine maximal trennende Hyperebene zu konstruieren [6], werden die Parameter $\boldsymbol{w}$ und b durch ein quadratisches Optimierungsproblem bestimmt; zur ausführlichen Herleitung vgl. [6; 26].

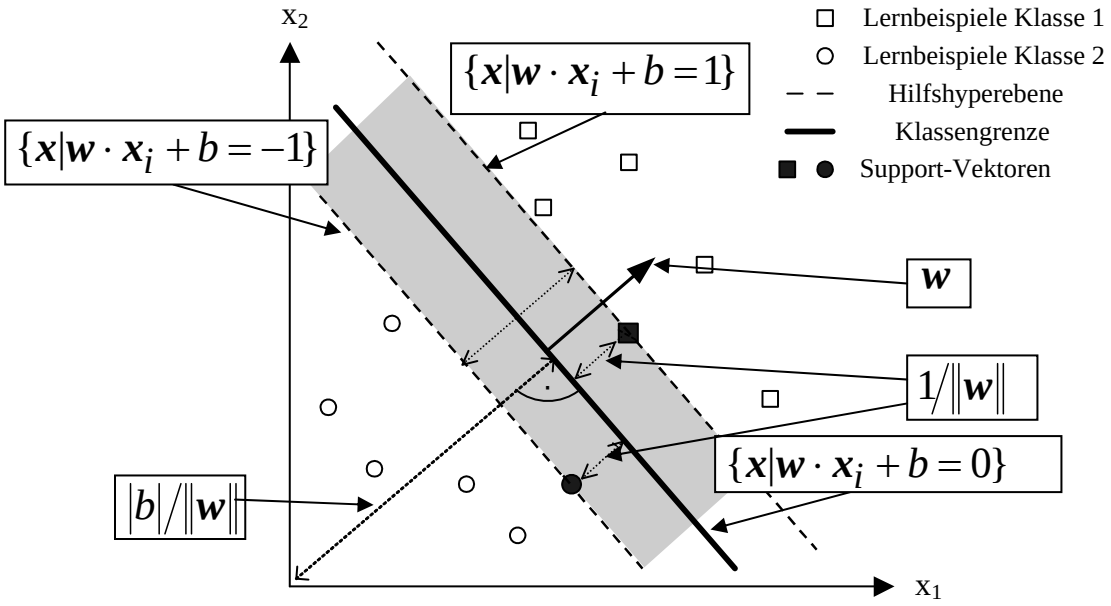


Abb. 1: Klassengrenze, Hilfshyperebenen und Trennungsgürtel einer linearen SVM, in Anlehnung an [3].

Gemäß Abb. 1 lässt sich die Breite des Trennungsgürtels (engl. Margin of separation) maximieren, wenn $\|\boldsymbol{w}\|$ minimiert wird. Dazu werden zwei Hilfshyperebenen parallel von der Trennlinie weg verschoben, bis sie die angrenzenden Datenpunkte berühren. Die der Klassengrenze am Nächsten liegenden Datenpunkte werden als Support Vektoren bezeichnet. Aus diesen Überlegungen ergibt sich das folgende Optimierungsproblem, wobei Schlupfvariablen ξ_i eingeführt wurden, um Fehlklassifikationen zu erlauben [6].

$$\min_{w,b} \frac{1}{2}\|\boldsymbol{w}\| + C\sum_{i=1}^{m}\xi_i$$
$$N.B.:\ y_i(\langle \boldsymbol{w}, \boldsymbol{x}_i\rangle + b) \geq 1 - \xi_i \quad \forall i \qquad (3)$$

In dem zu (3) korrespondierenden dualen Problem (4), erscheinen die Eingabevektoren $\boldsymbol{x}$ ausschließlich aus Skalarprodukt, was eine einfache Verallgemeinerung des Algorithmus für den Fall der nichtlinearen Klassifikation erlaubt.

$$\max_{\lambda} = \sum_{i=1}^{m}\lambda_i - \frac{1}{2}\sum_{i,j=1}^{m}\lambda_i\lambda_j y_i y_j \langle \boldsymbol{x}_i, \boldsymbol{x}_j\rangle$$
$$N.B.:\quad \sum_{i=1}^{m}\lambda_i y_i = 0 \qquad (4)$$
$$0 \leq \lambda_i \leq C \quad \forall i$$

Dazu werden die Eingabedaten mittels einer nichtlinearen Abbildung φ in einen Merkmalsraum höherer Dimension transformiert und eine lineare Trennlinie in diesem Raum konstruiert. Dies kommt einer nichtlinearen Trennung im Eingaberaum gleich. Um die Abbildung nicht explizit berechnen zu müssen, wird eine sog. Kernfunktion K eingeführt, die das Skalarprodukt zweier Vektoren im hochdimensionalen Abbildungsraum implizit berechnet:

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \left\langle \varphi(\boldsymbol{x}_i), \varphi(\boldsymbol{x}_j)\right\rangle \qquad (5)$$

Der Algorithmus bleibt davon weitestgehend unberührt, so dass sich die endgültige Form einer SVM zu

$$\max_{\lambda} = \sum_{i=1}^{m}\lambda_i - \frac{1}{2}\sum_{i,j=1}^{m}\lambda_i\lambda_j y_i y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j)$$
$$N.B.:\quad \sum_{i=1}^{m}\lambda_i y_i = 0 \qquad (6)$$
$$0 \leq \lambda_i \leq C \ \forall i$$

mit der Entscheidungsfunktion

$$e(\boldsymbol{x}) = \operatorname{sgn}\left(\sum_{i=1}^{m}\lambda_i y_i K(\boldsymbol{x}_i, \boldsymbol{x}) + b\right) \qquad (7)$$

ergibt.

Die häufig eingesetzten Standardkernfunktionen sind in der folgenden Tab. 1 zusammengefasst.

Linearer Kern	$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \left\langle \boldsymbol{x}_i, \boldsymbol{x}_j \right\rangle$
Polynom des Grades *d*	$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = (a \cdot \boldsymbol{x}_i \cdot \boldsymbol{x}_j + b)^d,\ d \in \mathrm{N}$
Radiale Basisfunktion (RBF)	$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(\frac{-\| \boldsymbol{x}_i - \boldsymbol{x}_j \|^2}{2\sigma^2} \right)$
Sigmoide Kernfunktion	$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \tanh \ \ (a \cdot \boldsymbol{x}_i \cdot \boldsymbol{x}_j - b)$

Tab. 1: Typische Kernfunktionen für SVM

3 Vorgehensmodell zur Anwendung von SVM im aKBM

Nach einer aktuellen Umfrage von *KDnuggets* [23] sind SVM als Data Mining Methode bisher kaum verbreitet, obwohl ihre Leistungsfähigkeit hinlänglich empirisch nachgewiesen wurde, vgl. z.B. [1; 5; 7; 17]. Da gute Klassifikationsergebnisse allein offenbar nicht für einen verstärkten Einsatz von SVM in der betrieblichen Praxis ausreichen, soll im Folgenden der Data Mining Prozess [11] genauer betrachtet werden. Hierbei zeigt sich, dass typische Anforderungen des aKBM durch SVM sehr gut erfüllt werden und der Analyseprozess von methodischen Eigenschaften des Verfahrens profitiert.

3.1 Anforderungen des Anwendungsgebiets

Theoretisch können SVM für jede Art von Klassifikationsproblem eingesetzt werden. Im aKBM finden sich entsprechende Data Mining Fragestellung z.B. bei der Selektion von Adressen für eine Mailingkampagne. Je nachdem, ob Kunden bei einer vergangenen Aktionen eine Reaktion, z.B. im Sinne einer Bestellung, gezeigt haben, lassen sich die Klassen Reagierer und Nicht-Reagierer unterscheiden. Ein Klassifikator soll nun anhand von demografischen und transaktionsorientierten Kundenmerkmalen sowie den Informationen aus zurückliegenden Aktionen diejenigen Kunden ermitteln, die bei einer zukünftigen Kampagne voraussichtlich reagieren werden.

Ein grundsätzliches Problem ist hierbei die Auswahl geeigneter Kundenmerkmale (Alter, Geschlecht, Anzahl bisheriger Bestellungen, etc.), sog. Prediktoren, anhand derer das Lernverfahren zwischen Kunden der einen und solchen der anderen Klasse unterscheiden soll. Durch den Einsatz von Informationssystemen zur Unterstüt-

zung/Automation operativer Geschäftsprozesse verfügen Unternehmen gewöhnliche über eine Vielzahl kundenbezogener Daten, die als potentielle Prediktoren in Frage kommen. Die Verwendung aller möglichen Merkmale in einem Data Mining Modell ist jedoch problematisch. Zum einen wird die Rechenzeit zum Trainieren eines Klassifikators direkt erhöht. Weiterhin wird die Gefahr der Überanpassung, also des schlichten Auswendiglernens der Trainingsdaten, durch eine große Zahl von Prediktoren vergrößert [2]. Um diesen Effekt zu kompensieren, muss die Zahl der Lernbeispiele erhöht werden, was zu einem erneuten Ansteigen der Trainingszeit führt. In Folge dessen wird eine Form der Merkmalsbewertung benötigt, mit deren Hilfe Prediktoren, die die Klassenprognose nur wenig begünstigen, eliminiert werden können.

Im Prozess der Wissensentdeckung in Datenbanken wird diese Merkmalsselektion der Datenaufbereitung zugerechnet und ist einem Data Mining im engeren Sinne vorgestellt [8]. Neben der Wahl eines konkreten Verfahrens, müssen in der eigentlichen Data Mining Phase auch die jeweiligen Verfahrensparameter festgelegt werden [11]. Dies erfolgt in einem iterativen Prozess, indem ein Modell mit einer bestimmten Parameterkonstellation entwickelt und anschließend auf sog. Validierungsdaten, d.h. Daten die nicht in den Trainingsprozess eingeflossen sind, evaluiert wird. Im Sinne eines zeiteffizienten Data Mining sollte diese Aufgabe durch ein verfahrensspezifisches Vorgehensmodell unterstützt werden, welches in wenigen Schritten zu guten Parametereinstellungen führt.

Die Zielsetzung eines aKBM orientierten Data Minings ist stets die Lösung eines betrieblichen Entscheidungsproblems; z.B. die Auswahl von Kunden für eine Mailingkampagne. Um eine effektive und effiziente Entscheidungsunterstützung zu ermöglichen, müssen in Frage kommende Lösungsverfahren in der Lage sein, die mit ihrer Prognose einhergehenden Kosten zu berücksichtigen. Bezogen auf Klassifikationsverfahren ergibt sich hieraus die Anforderung, dass keine ausschließliche Fokussierung auf abstrakte Gütekriterien wie Treffer- oder Fehlerraten erfolgen sollte, sondern betriebswirtschaftlich relevante Zielgrößen wie die Minimierung der Klassifikationskosten im Data Mining Prozess berücksichtigt werden müssen.

Im Folgenden wird gezeigt, wie diese Anforderungen durch SVM erfüllt werden.

3.2 Merkmalsbewertung und -selektion

Techniken zur Merkmalsbewertung und –selektion lassen sich in die Gruppen Filter und Wrapper unterteilen [21]. Der Filteransatz ist dadurch gekennzeichnet, dass eine Untermenge aussagekräftiger Merkmale unabhängig vom Klassifikationsverfahren durch dedizierte Methoden ermittelt wird. Erfolgt hingegen die Merkmalsauswahl direkt unter Verwendung des Klassifikators, entspricht dies dem Wrapperansatz. Die

Prognosekraft eines Merkmals kann von der Art und Weise wie ein konkretes Verfahren Klassifikationsentscheidungen trifft[2] abhängen, so dass ein Vorteil von Wrappern darin zu sehen ist, dass die Merkmalsauswahl unmittelbar durch die Klassifikationsgüte gesteuert wird [21]. Aus Sicht der praktischen Anwendbarkeit sind die Verständlichkeit und methodische Einheitlichkeit von Wrappern zu begrüßen.

Zu den bekanntesten Verfahren für eine SVM basierte Merkmalsauswahl nach dem Wrapper Paradigma zählt die rekursive Merkmalselimination nach Guyon et al., welche eine Merkmale nach ihrem Beitrag zum „Margin of separation" bewertet [10]. Sei λ^* die optimale Lösung von (6), dann lässt sich unter der Annahme, dass die Menge der Support Vektoren sich durch die Entfernung eines Merkmals nicht wesentlich ändert, die Veränderung des Margins bei Elimination eines Merkmals gemäß der Formel

$$\Delta W_t = \sum_{i,j}^{n} \lambda_i^* \lambda_j^* y_i y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j) - \sum_{i,j}^{n} \lambda_i^* \lambda_j^* y_i y_j K(\boldsymbol{x}_i^{-t}, \boldsymbol{x}_j^{-t}) \qquad (8)$$

Berechnen, wobei x^{-t} einen Eingabevektor beschreibt, aus dem das Merkmal t entfernt wurde. Es ergibt sich eine Sortierung der Merkmale nach absteigendem Marginbeiträgen ΔW_t, auf deren Basis eine Merkmalsselektion ermöglicht wird.

Für den Fall einer linearen SVM vereinfacht sich Sortierung und kann direkt aus dem Gewicht eines Merkmals im Normalenvektor w abgeleitet werden [10].

$$\Delta W_t = \left(w_t\right)^2 \qquad (9)$$

Über eine einfache Sortierung hinausgehend können diese Gewicht gemäß der Entscheidungsfunktion einer linearen SVM, vgl. (2), inhaltlich interpretiert und verglichen werden, was die Transparenz des Verfahrens erhöht.

Andererseits reichen lineare Modelle zur Abbildung der komplexen, häufig nichtlinearen Zusammenhänge in aKBM-orientierten Klassifikationsproblemen nicht aus. Dies ist in einem Schritt der Datenvorverarbeitung aber auch nicht erforderlich, so dass die Vorteile einer linearen SVM hinsichtlich Merkmalsselektion und Erklärungsfähigkeit im Rahmen eines Vorgehensmodells für SVM in aKBM genutzt werden sollten.

3.3 Festlegung der Modellparameter

In der eigentlichen Data Mining Phase sind die freien Modellparameter einer SVM festzulegen. Dies betrifft die Auswahl einer Kernfunktion sowie deren Kernelparame-

[2] In Frage kommen sog. Dichte- oder Verteilungsschätzer, grenzbildende Klassifikatoren oder Schätzer von a-posteriorie-Wahrscheinlichkeiten [25].

tern und der sog. Regularisierungskonstanten C; vgl. (6). Da Standardkernfunktionen (siehe Tab. 1) lediglich ein bis drei Kernelparameter aufweisen, gestaltet sich die Parametrisierung einer SVM, z.B. im Vergleich zu Multi Layer Perceptrons, eher einfach. Angesichts der Größe praktischer Problemstellungen und der damit verbundenen Rechenzeit ist eine Vollenumeration aber trotzdem unmöglich. Während RBFs sich auch bei großen Datensätzen bewährt haben, kommen polynomiale Kernfunktionen aufgrund zu langer Rechzeiten häufig nicht in Frage [7; 19]. Der einzige Parameter einer RBF ist der Glättungsparameter σ, so dass bei Wahl dieses Kernels mit C und σ insgesamt lediglich zwei Freiheitsgrade festzulegen sind. Hierfür schlagen Keerthi und Lin ein Referenzmodell vor, das von der Verwendung einer linearen SVM ausgeht, den in diesem Schritt ermittelten Wert für C fixiert und in einer zweiten Phase die Parametrisierung einer SVM mit radialem Kern iterativ gemäß

$$\log(\sigma^2) = \log(C) - \log(\tilde{C}) \qquad (10)$$

vollzieht [15]. $\tilde{C}$ kennzeichnet dabei den Wert der Regularisierungskonstanten, mit dem die beste Klassifikationsleistung für die lineare SVM beobachtet wurde. Das Vorgehen wird durch eine Betrachtung des asymptotischen Verhaltens von SVM mit RBF Kern gerechtfertig und verringert die Anzahl der Iterationsschritte gegenüber herkömmlichen gridbasierten Suchstrategien [15].

Eine Untersuchung, ob diese Heuristik auch für andere Kernfunktionen gültig ist, steht derzeit noch aus. Allerdings ist deren Notwendigkeit auch zu bezweifeln, da RBFs unter den üblichen Kernen als dominant angesehen werden können und komplexere Kernfunktionen wie verallgemeinerte RBFs [9] oder kombinierte Kerne [22] spezialisierte Verfahren zur Parametereinstellung benötigen.[3]

Für SVM im aKBM empfehlen wir daher vorerst die Verwendung von SVM mit RBF Kernfunktion und die in (10) beschriebenen Heuristik zur Parametereinstellung. Dieser Ansatz fügt sich zudem sehr gut in unser Vorgehensmodell ein, da auch dieses mit einer linearen SVM beginnt. D.h., nach durchlaufen der unter 3.2 beschriebenen Phase zur Merkmalsauswahl, steht bereits ein geeigneter Wert $\tilde{C}$ zur Verfügung, bzw. wird parallel zu der Untermenge relevanter Merkmale bestimmt, so dass im Rahmen der eigentlichen Modellselektion nach (10) vorgegangen werden kann.

[3] RBF-Kerne und die vorgeschlagene Heuristik erscheinen angesichts der methodischen Eleganz mächtigerer, kombinierter Kernfunktionen; z.B. [9; 22], unterlegen. Allerdings ist zu beachten, dass letztere den Nachweis ihrer praktischen Anwendbarkeit im Data Mining, insbesondere hinsichtlich Laufzeitverhalten, bisher nicht erbringen konnten.

3.4 Bestimmung der kostenoptimalen Klassifikationsschwelle

Ausgangspunkt einer sog. kostensensitiven Klassifikation ist eine Verlustmatrix, die für jede mögliche Kombination aus geschätzter und tatsächlicher Klasse die mit einem Fehler assoziierten Kosten enthält [25].

SVM unterstützen in der Basisversion nur den binären Zwei-Klassenfall, so dass die Berücksichtigung von Fehlklassifikationskosten auf den trade-off zwischen der Sensitivität und der Spezifität eines Klassifikators zurückgeführt werden kann [27]. Für SVM kann eine solche explizite Berücksichtigung der beiden möglichen Fehlertypen sehr einfach durch die Einführung klassenspezifischer Regularisierungsparameter erreicht werden [20; 27]. Als Alternative schlagen Wu und Chang [28] eine Modifikation der Kernfunktion vor, um der relevanteren Klassen stärkeres Gewicht zu verleihen. Beide Ansätze versuchen durch algorithmische Modifikationen den Trainingsprozess einer SVM zu verändern, um Fehlklassifikationskosten bereits in dieser Phase zu berücksichtigen.

Eine komplementäre Möglichkeit besteht darin, eine kostenoptimale Klassifikationsschwelle τ erst im Anschluss an die Modellerstellung zu bestimmen. Dazu muss lediglich die SVM Entscheidungsfunktion gemäß (11) modifiziert werden.

$$e(\boldsymbol{x}) = \begin{cases} +1, \text{ wenn } \sum_{i=1}^{m} \lambda_i y_i K(\boldsymbol{x}_i, \boldsymbol{x}) + b \geq \tau \\ -1, \text{ sonst} \end{cases} \qquad (11)$$

Offenkundig entspricht dies einer Variation des Entscheidungsparameters b, wie sie erstmalig von Karakoulas und Shawe-Taylor [14] vorgeschlagen wurde. Carrizosa und Martin-Barragan zeigen, dass diese Vorgehensweise eine pareto-optimale Lösung hinsichtlich des trade-off zwischen Sensitivität und Spezifität gewährleistet [4]. Diese Form der Kostenberücksichtigung führt keine zusätzlichen Freiheitsgrade in die Trainingsphase ein und lässt das oben beschriebene Problem der Modellselektion unberührt. Ein kostenoptimaler Wert für τ bzw. b kann über eine ROC-Analyse [24] sehr einfach und schnell ermittelt werden.

Sollen Fehlklassifikationskosten direkt im Training berücksichtig werden, empfiehlt sich trotzdem die nachträgliche Justierung von b, um die Anpassung des Prognosemodells an die betriebliche Entscheidungssituation weiter zu verbessern.

3.5 Zusammenfassung

Entsprechend der Darstellung unter 3.1 – 3.3 empfiehlt sich für SVM im aKBM ein dreistufiges Vorgehensmodell. Zunächst wird mittels einer linearen SVM eine Un-

termenge relevanter Merkmale ermittelt. Der einzige einzustellende Methodenparameter C kann über Standardverfahren wie Kreuzvalidierung oder Bootstrapping [16] einfach bestimmt werden. Dieser Parameterwert sowie die Untermenge an relevanten Merkmalen bilden den Input für die zweite Stufe. Hier wird ein mächtigerer Klassifikator mit radialer Kernfunktion trainiert, der nichtlineare Zusammenhänge adäquat abbilden kann. Durch die im ersten Schritt erreichte Merkmalsreduktion und die Fixierung eines freien Methodenparameters kann der Zeitaufwand zur Ermittlung eines geeigneten Klassifikationsmodells erheblich reduziert werden. Im letzten Schritt wird eine kostenoptimale Klassifikationsschwelle über die ROC-Analyse eingestellt, um die Charakteristika der betrieblichen Entscheidungssituation bestmöglich widerzuspiegeln.

4 Schlussbetrachtung

Das aKBM ist eines der wichtigsten Anwendungsfelder für betriebliches Data Mining, wobei eine große Zahl analytischer Fragestellungen als Klassifikationsproblem modelliert werden können. SVM wurden zur Lösung von Klassifikationsproblemen unter anderem in der medizinischen Diagnostik [10; 17] sehr erfolgreich eingesetzt, so dass das Potential dieses Verfahrens für betriebswirtschaftliche Anwendungen zu untersuchen ist. In der vorliegenden Arbeit wurde ein Vorgehensmodell für die Anwendung solcher Support Vektor Klassifikatoren im aKBM vorgestellt. Ausgehend von typischen Problemstellungen im Data Mining Prozess (Merkmalsselektion, Modellauswahl und -evaluation) wurde ein dreistufiger Analyseprozess entworfen, der sich unmittelbar an den Erfordernissen der Anwendungsdomäne orientiert. Die gute Abdeckung der Anforderungen sowie eine weitreichende methodische Einheitlichkeit und damit einhergehende Verständlichkeit des Ansatzes, sollten auch einen verstärkten Einsatz von SVM in der betrieblichen Praxis begünstigen.

5 Literatur

1. Baesens, B.; Van Gestel, T.; Viaene, S.; Stepanova, M.; Suykens, J.; Vanthienen, J. (2003) Benchmarking state-of-the-art classification algorithms for credit scoring. *Journal of the Operational Research Society* 54(6): 627-635
2. Bishop, C.M. (1995) *Neural networks for pattern recognition*. Oxford University Press Oxford
3. Burges, C.J.C. (1998) A Tutorial on Support Vector Machines for Pattern Recognition. *Data Mining and Knowledge Discovery* 2(2): 121-167
4. Carrizosa, E.; Martin-Barragan, B. (to appear 2005) Two-group classification via a biobjective margin maximization model. *European Journal of Operational Research*

5. Chang, C.-C.; Lin, C.-J. (2001) IJCNN 2001 Challenge: Generalization Ability and Text Decoding. *Proceedings of the International Joint Conference on Neural Networks*, 1031-1036. IEEE Press Piscataway, New York

6. Cristianini, N.; Shawe-Taylor, J. (2000) *An introduction to support vector machines and other kernel-based learning methods*. Cambridge University Press Cambridge

7. Crone, S.F.; Lessmann, S.; Stahlbock, R. (2004) Empirical Comparison and Evaluation of Classifier Performance for Data Mining in Customer Relationship Management. In: Wunsch, D (Ed) *Proceedings of the International Joint Conference on Neural Networks*, 443-448. IEEE Press New York

8. Fayyad, U.; Piatetsky-Shapiro, G.; Smyth, P. (1996) From Data Mining to Knowledge Discovery in Databases: an overview. *AI Magazine* 17(3): 37-54

9. Friedrichs, F.; Igel, C. (erscheint 2005) Evolutionary Tuning of multiple SVM parameters. *Neurocomputing*

10. Guyon, I.; Weston, J.; Barnhill, S.; Vapnik, V. (2002) Gene Selection for Cancer Classification using Support Vector Machines. *Machine Learning* 46(1-3): 389-422

11. Hippner, H.; Wilde, K. (2001) Der Prozess des Data Mining im Marketing. In: Hippner, H; Küsters, U; Meyer, M; Wilde, K (Ed) *Handbuch Data Mining im Marketing: Knowledge Discovery in Marketing Databases*, 22-94. Vieweg Braunschweig, Wiesbaden

12. Hippner, H.; Wilde, K.D. (2002) CRM - Ein Überblick. In: Helmke, S; Uebel, M; Dangelmaier, W (Ed) *Effektives Customer Relationship Management*, 3-38. Gabler Wiesbaden

13. Hippner, H.; Wilde, K.D. (2002) Data Mining im CRM. In: Helmke, S; Uebel, M; Dangelmaier, W (Ed) *Effektives Customer Relationship Management*, 211-232. Gabler Wiesbaden

14. Karakoulas, G.; Shawe-Taylor, J. (1999) Optimizing classifiers for imbalanced training sets. In: Kearns, M; Solla, S; Cohn, D (Ed) *Advances in Neural Information Processing Systems*, 253-259. MIT Press Cambridge, Mass.

15. Keerthi, S.S.; Lin, C.-J. (2003) Asymptotic Behaviours of Support Vector Machines with Gaussian Kernel. *Neural Computation* 15(7): 1667-1689

16. Kohavi, R. (1995) A study of cross-validation and bootstrap for accuracy estimation and model selection. In: Mellish, Cs (Ed) *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, 1137-1143. Morgan Kaufmann

17. Kowalczyk, A.; Raskutti, B. (2002) One Class SVM for Yeast Regulation Prediction. *SIGKDD Explorations Newsletter* 4(2): 99-100

18. Lessmann, S. (2003) Customer Relationship Management. *WISU - das Wirtschaftsstudium* 32(2): 190-192

19. Lessmann, S. (2004) Solving imbalanced classification Problems with Support Vector Machines. In: Arabnia, Hr (Ed) *Artificial Intelligence*, 214 - 220. CSREA Press

20. Lin, Y.; Lee, Y.; Wahba, G. (2002) Support Vector Machines for Classification in Nonstandard Situations. *Machine Learning* 46(1-3): 191-202

21. Liu, H.; Motoda, H. (1998) *Feature selection for knowledge discovery and data mining*. Kluwer Boston

22. Nguyen, H.-N.; Ohn, S.-Y.; Choi, W.-J. (2004) Combined Kernel Function for Support Vector Machine and Learning Method Based on Evolutionary Algorithm. In: Pal, Nr; Kasabov, N; Mudi, Rk (Ed) *Proceedings of the 11th International Conference on Neural Information Processing*. Springer Verlag Heidelberg

23. O.V.:*Data Mining Techniques*, http://www.kdnuggets.com/polls/2005/data_mining_techniques.htm. Letzter Abruf: 15.03.05.

24. Provost, F.; Fawcett, T. (1997) Analysis and Visualization of Classifier Performance: Comparison under Imprecise Class and Cost Distributions. In: Heckerman, D; Mannila, H; Pregibon, D; Uthurusamy, R (Ed) *Proceedings of the Third International Conference on Knowledge Discovery and Data Mining*, 43-48. AAAI Press Menlo Park, Calif.

25. Schürmann, J. (1996) *Pattern classification: a unified view of statistical and neural approaches*. Wiley & Sons New York

26. Vapnik, V.N. (1995) *The Nature of Statistical Learning Theory*. Springer New York

27. Veropoulos; Cristianini, N.; Campbell, C. (1999) Controlling the Sensitivity of Support Vector Machines. In: Dean, T (Ed) *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*, 55-60. Morgan Kaufmann San Francisco, Calif.

28. Wu, G.; Chang, E.Y. (erscheint 2005) KBA: Kernel Boundary Alignment considering Imbalanced Data Distribution. *IEEE Transactions on Knowledge and Data Engineering*

Empirical Comparison and Evaluation of Classifier Performance for Data Mining in Customer Relationship Management

Sven F. Crone
Dep. of Management Science
Lancaster University, England
E-mail: sven.f.crone@crone.de

Stefan Lessmann
Inst. of Business Information Systems
University of Hamburg, Germany
E-mail: lessmann@econ.uni-hamburg.de

Robert Stahlbock
Inst. of Business Information Systems
University of Hamburg, Germany
E-mail: stahlboc@econ.uni-hamburg.de

Abstract— **In competitive consumer markets, data mining for customer relationship management faces the challenge of systematic knowledge discovery in large data streams to achieve operational, tactical and strategic competitive advantages. Methods from computational intelligence, most prominently artificial neural networks and support vector machines, compete with established statistical methods in the domain of classification tasks. As both methods allow extensive degrees of freedom in the model building process, we analyse their comparative performance and sensitivity towards data pre-processing in a real-world scenario.**

I. Introduction

The customers of a company are regarded as valuable business resources in competitive markets, leading to efforts to systematically prolong and exploit existing customer relations. Consequently, the strategies and techniques of customer relationship management (CRM) have received increasing attention in management science.

CRM features data mining as a technique to gain knowledge about customer behaviour and preferences. Various paradigms of artificial neural networks (ANN) and support vector machines (SVM) have found consideration in the CRM area, promising effective and efficient solutions for managerial problems in similar domains. However, both classes and especially ANN allow severe degrees of freedom in the model-building process through extensive parameters, making broad adoption in the CRM area difficult. In addition, different variations of data pre-processing through scaling, encoding etc. raise degrees of freedom prior to the actual data mining phase even further.

Following, we conduct an experimental evaluation of the competing methods in the domain of analytic CRM (aCRM), striving to exemplify the adequacy and performance of ANN versus SVM for the task of response optimization based upon an empirical, numerical experiment from an ongoing project with a large publishing house.

Following a brief introduction to data mining within CRM, section 3 assesses the competing approaches of different ANN paradigms and SVM to classification tasks, highlighting the degrees of freedom in the modelling process. This is followed by an experimental evaluation of their competitive performance on an empirical dataset in section 4. Conclusions are given in section 5.

II. Data Mining in Customer Relationship Management

In an increasingly competitive market, caused by inconsistent consumer behaviour, escalating globalization and the extending possibilities to conduct business over the internet in a recessive global economy, the customers of a company are regarded as key business resources [1]. Consequently, aCRM has received increasing attention in management science as a systematic approach to strategically prolong and exploit these valuable customer relations, providing the tools and infrastructure to record and analyze customer centred information in order to build up longer lasting and more profitable customer relationships [2]. The analytical process of collecting, assembling and understanding the profound knowledge about customer behaviour and preferences is referred to as knowledge discovery in databases (KDD).

KDD may be regarded as various, iterative and interdependent phases, such as data selection, data preprocessing and cleaning as well as a data transformation stage that ensures a mathematical feasible data format for the proceeding application of a specific data mining algorithm [3].

Utilising the processed and transformed data set, the stage of data mining consist of selecting and applying a suitable data mining method in order to identify hidden patterns in the data relevant to business decisions through a partially automated analysis [3]. The results must be evaluated not only regarding precision and statistical significance but also economical relevance.

Data mining problems in the aCRM domain, such as response optimization to distinguish between customers who will react to a mailing campaign or not, churn prediction, in the form of classifying customers for churn probability, cross-selling, or up-selling are routinely modeled as classification tasks, predicting a discrete, of-

0-7803-8359-1/04/$20.00 ©2004 IEEE

ten binary feature using empirical, customer centered data of past sales, amount of purchases, demographic or psychographic information etc.

Recently, various architectures from computational intelligence and machine learning, such as artificial neural networks (ANN) and support vector machines (SVM) have found increasing consideration in practice, promising effective and efficient solutions for classification problems in real-world applications through robust generalization in linear and non-linear classification problems, deriving relationships directly from the presented sample data without prior modeling assumptions.

Following, we will give a brief discussion on the different classification approaches of the competing soft computing methods.

III. Neural Networks and Support Vector Machines for Classification

A. Soft Computing Methods for Classification

Data driven methods from computational intelligence, share a common approach of learning machines in classification for data mining [4].

Let all relevant and measurable attributes of an object, e.g. a customer, be combined in a vector $\boldsymbol{x}$ and the set $X = \{\boldsymbol{x}_1,...,\boldsymbol{x}_n\}$ denotes the input space with n objects. Each object belongs to a discrete class $y \in Y$ and we will refer to a pair $(\boldsymbol{x}, y)$ as an example of our classification problem. Presuming that it is impossible to model the relationship between attribute vector $\boldsymbol{x}$ and class membership y directly, either because it is unknown, to complex or the data is corrupted by noise, and that a sufficient large set of examples $S = ((\boldsymbol{x}_1, y_1),...,(\boldsymbol{x}_l, y_l)) \subseteq (X \times Y)^l$ is available, we can incorporate a machine to learn the mapping between $\boldsymbol{x}$ and y. The learning machine is actually defined by a set of possible mappings $x \rightarrow f(\boldsymbol{x}, \boldsymbol{\alpha})$, where the functions $f(\boldsymbol{x}, \boldsymbol{\alpha})$ themselves are labeled by the adjustable parameter vector $\boldsymbol{\alpha}$ [5]. The objective is to modify the free parameters $\boldsymbol{\alpha}$ to find a specific learning machine which captures the relationships in the training examples, $f_{\alpha}(\boldsymbol{x}_i) \approx y_i \ \forall i = (1,...,i)$, incrementally minimizing a given objective function and generalizing the problem structure within to allow correct estimation of unseen objects on the basis of their attribute values $\boldsymbol{x}_i$.

Following, we outline the specific modeling-properties for classification for alternative network paradigms. For a comprehensive discussion readers are referred to [4-7].

B. Multilayer Perceptrons

Multilayer perceptrons (MLPs) represent the most prominent and well researched class of ANNs in classification, implementing a feedforward and supervised paradigm. MLPs consist of several layers of nodes u_j fully interconnected through weighted acyclic arcs w_{ij} from each preceding layer to the following, without lateral connections or feedback [8]. Each node output calculates a transformed weighted linear combination of its inputs of the form $f_{act}(\boldsymbol{w}^T\boldsymbol{o})$, with $\boldsymbol{o}$ the vector of output activations o_j from the preceding layer, $\boldsymbol{w}^T$ the transposed column vector of weights w_{ij}, and f_{act} a bounded non-decreasing non-linear function, such as the linear threshold or the sigmoid, with one of the weights w_{oj} acting as a trainable bias θ_j connected to a constant input $o_0 = 1$ [6].

The desired output as a binary class membership is often coded with one output node $y_i = \{0;1\}$ or for multiple classifications n nodes with $f_i = \{(0,1);(1,0)\}$ respectively. For pattern classification, MLPs partition the input space through linear hyperplanes. To separate distinct classes, MLPs approximate a function $g(\boldsymbol{x})$: $X \rightarrow Y$ through adapting the free parameters $\boldsymbol{w}$ to minimize an objective function $e(\boldsymbol{x})$ on the training data, which partitions the X space into polyhedral sets or regions, each one being assigned to one class out of Y. Each node has an associated hyperplane to partition the input space into two half-spaces. The combination of the linear node-hyperplanes in additional layers allows a stepwise separation of complex regions in the input space, generating a decision boundary to separate the different classes. The orientation of the node hyperplanes is determined by $\boldsymbol{w}$ including threshold θ_j modeled as an adjustable weight w_{oj} to offset the node hyperplane along $\boldsymbol{w}$ for a distance $d = \theta_j \|\boldsymbol{w}\|$ from the origin for a more flexible separation. The node non-linearity f_{act} determines the output change as the distance from $\boldsymbol{x}$ to the node hyperplane [8].

The representational capabilities of a MLP are determined by the range of mappings it may implement through weight variation. MLPs with three layers are capable to approximate any desired bounded continuous function. The units in the first hidden layer generate hyperplanes to divide the input space in half-spaces. Units in the second hidden layer form convex regions as intersections of these hyperplanes. Output units form unisons of the convex regions into arbitrarily shaped, convex, non-convex or disjoint regions.

Given a sufficient number of hidden units, a MLP can approximate any complex decision boundary to divide the input space with arbitrary accuracy, producing a (0) when the input is in one region and an output of (1) in the other. This property, known as a universal approximation capability, poses the essential problems of adequate model complexity in depth and size, i.e. the number of nodes and layers, and controlling the network training process to prevent over-fitting.[8, 9]

C. Learning Vector Quantization

Learning Vector Quantization (LVQ), a supervised version of vector quantization, represent another para-

digm of feedforward, heter-associative ANNs, related to self-organizing maps (SOM) [10] and existing in various extensions (see, e.g., [11-13]. They are regularly applied in pattern recognition, multi-class classification and data compression tasks. LVQs are multi-layered, with only one hidden layer of Kohonen neurons.

The weight vector of the weights between all input neurons and a hidden Kohonen neuron is called a codebook vector (CV). In training, the weights are changed in accordance with adapting rules, changing the position of a CV in the feature space. The basic LVQ algorithm rewards correct classifications by moving the 'winner' – the CV which is nearest to the presented input vector *x(t)* – towards *x(t)*, whereas incorrect classifications are punished by moving the CV in opposite direction.

LVQs define class boundaries based on prototypes, a nearest-neighbor rule and a winner-takes-it-all paradigm by covering the feature space of samples with 'codebook vectors' (CVs), each representing a region labeled with a class. A CV can be seen as a prototype of a class member, localized in the centre of a class or decision region ('Voronoi cell') in the feature space. As a result the space is partitioned by a 'Voronoi net' of hyperplanes perpendicular to the linking line of two CVs (mid-planes of the lines forming the 'Delaunay net').

A class can be represented by an arbitrarily number of CVs, but one CV represents one class only. Since class boundaries are built piecewise-linearly as segments of the mid-planes between CVs of neighboring classes, the class boundaries are adjusted during the learning process. The tessellation induced by the set of CVs is optimal if all data within one cell indeed belong to the same class. Classification after learning is based on a presented sample's vicinity to the CVs: the classifier assigns the same class label to all samples that fall into the same tessellation: the label of the cell's prototype, equal to the CV nearest to the sample. The core of the heuristics is based on a distance function, e.g. the Euclidean distance, for comparison between an input vector with the class representatives. The distance expresses the degree of similarity between presented input vector and CVs. Small distance corresponds with a high degree of similarity and a higher probability for the presented vector to be a member of the class represented by the nearest CV. Therefore, the definition of class boundaries by LVQ is strongly dependent on the distance function, the start positions of CVs and their adjustment rules and the pre-selection of distinctive input features.

D. Support Vector Machines

The original support vector machine (SVM) can be characterized as a supervised learning algorithm capable of solving linear and non-linear classification problems. The main building blocks of SVMs are structural risk minimization, non-linear optimization and duality and kernel induced features spaces, underlining the technique with an exact mathematical framework [7].

The idea of support vector classification is to separate examples with a linear decision surface and maximize the margin between the two different classes. This leads to the convex quadratic programming problem (1) (the primal form was omitted for brevity, see for example [7]).

$$\max. \quad W(\lambda) = \sum_{i=1}^{l} \lambda_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \lambda_i \lambda_j y_i y_j \left(\boldsymbol{x}_i \cdot \boldsymbol{x}_j \right)$$

$$\text{s.t.} \quad 0 \le \lambda_i \le C \; ; \; \sum_{i=1}^{l} \lambda_i y_i = 0 \; (i = 1,...,l) \tag{1}$$

The examples for which the Lagrange multiplier λ_i, is positive are called (bounded) support vectors as they define the separating hyperplane. C is a constant cost parameter, enabling the user to control the trade-off between learning error and model complexity, regarded by the margin of the separating hyperplane [5]. As complexity is considered directly during the learning stage, the risk of over-fitting the training data is less severe for SVM.

For constructing more general non-linear decision functions than hyperplanes, SVMs implement the idea to map the input vectors into a high-dimensional feature space Ψ via an a priori chosen non-linear mapping function $\Phi : X \to \Psi$. The construction of a separating hyperplane in the features space leads to a non-linear decision boundary in the original space. Expensive calculation of dot products $\Phi(\boldsymbol{x}) \cdot \Phi(\boldsymbol{x}_i)$ in a high-dimensional space can be avoided by introducing a kernel function $K(\boldsymbol{x}, \boldsymbol{x}_i) = \Phi(\boldsymbol{x}) \cdot \Phi(\boldsymbol{x}_i)$ [5]. Leaving the algorithms almost unchanged, this reduces numerical complexity significantly and allows efficient support vector learning for up to hundreds of thousand examples.

Degrees of freedom are significantly smaller for SVM, compared to MLP. The main freedom is the choice of a kernel function and the corresponding kernel parameters, influencing the speed of convergence and the quality of results. Furthermore, the choice of the cost parameter C is vital to obtain good classification results.

IV. Simulation Experiment of Soft Computing Classifiers

A. Objective

The main goal of the empirical simulation experiment is the evaluation of soft computing classification algorithms implemented as SVM, MLP and LVQ in a real world scenario of aCRM. An important objective for a large publishing house is to sell a second subscription to a customer, who has already subscribed one magazine in order to make extra profit ('cross selling'). Therefore, special offers are posted to those customers ('mailing

campaign') in order to take advantage of cross selling potential.

One main factor for profit is the response quote (the number of new subscriptions divided by the number of sales letters). By means of response optimization a presumable optimal group of addresses with as much responses as possible is chosen for the campaign. From the point of aCRM and data mining the problem is to identify a high probability of a second subscription based on attributes of customers with one subscription, e.g. the type of journal already subscribed.

In general, classification algorithms are capable of solving this kind of problem, but it's unclear, which method and which parameterization is best suited. Further more, no algorithm can directly operate on raw data and the necessary pre-processing stage offers an even larger variety of degrees of freedom making the overall task even more complicated for business users.

The empirical simulation delivers valuable hints about an appropriate classification technique and its sensitivity with regard to parameterization and pre-processing issues. Of special interest is the question, if SVMs - quite new to new to the area of data mining and, due to the smaller number of parameters easier to manage - can compete with or even outperform well established techniques like neural networks.

B. Experimental Design

Following, a description of the selected free modeling parameters for all methods used in the comparative experiments is given. A hold-out method, dividing the data into three separate sets was chosen to control over-fitting and allow out-of-sample evaluation.

The available data consisted of 300,000 customer records, which were selected for a previous mailing campaign. The number of subscriptions sold in this campaign was given with 4,019, resulting in a response quote of 1.24%. Handling the extreme dissymmetry in class distributions turned out to be a major challenge of our analysis. Usual approaches to deal with asymmetric class distributions include algorithmic modifications/extensions and resampling strategies. As sampling was inevitable due to the large data set size and because MLP and LVQ do not support asymmetric cost functions natively the latter approach was chosen.

As we are ultimately interested in the minority class of customers who responded in the last mailing, a stratified sampling technique was incorporated to increase the learning machines sensibility for that class. However, stratified sampling introduces another degree of freedom to the experiment, as an appropriate class distribution has to be chosen for the training set (the hold-out set was created by random sampling, ensuring a realistic performance evaluation). A pre-testing stage revealed, that the best classification results where obtained, if positive and negative examples in the training set where evenly distributed. To create data sets of reasonable size, oversampling has been applied to create three disjoint data sets, described in Table 1, which formed the basis for all following experiments.

TABLE 1
DATA SET SIZE AND STRUCTURE FOR THE EMPIRICAL SIMULATION

data set label	data partition		data set usage
training set	20,000	class 1	Data sample for the learning algorithm to build a concrete classifier
	20,000	class 0	
validation set	15,000	class 1	Used for model/parameter selection
	15,000	class 0	
generalisation set	1,011	class 1	Hold-out set for out-of-sample evaluation of classifier performance
	73,989	class 0	

Among the vast degrees of freedom in the preprocessing stage, the encoding of categorical attributes, present in almost every aCRM related analysis, and the selection of eligible input variables are most relevant. Therefore, the experimental set-up consists of the combination of three commonly used encoding schemes (N encoding, N-1 encoding and using a single number per categorical attribute) with input and instance selection techniques; see Table 2.

Fixing the general experimental framework, several parameterizations for MLP, LVQ and SVM where evaluated and their corresponding performance compared on the generalization set.

An iterative heuristic approach to determine appropriate architectures (e.g., number of hidden neurons) was selected for ANN. Each network was randomly initialized with 5 to 10 different random seeds to account for alternative starting weights. We selected an early stopping approach, evaluating each network's mean classification rate on a validation set after r iterations and stopping the learning process after no increase for s iterations (with variations in r and s). For the MLP, the weighted sum was chosen as the input function and a hyperbolic tangent activation function in all hidden nodes. The output layer used a 1-of-n-code to present two different classes, using a softmax output function with linear activation function.

Using a SVM classifier the choice of a network architecture is replaced by selecting an appropriate kernel function [5] and we utilized the LIBSVM [14] package for our experiment. The application of SVMs to database marketing problems like the one described above is an ongoing research topic and no kind of prior knowledge was available to give hints which kernel would best suit the data. Hence, we selected an iterative approach, evaluating the standard linear, polynomial and Gaussian kernels with a broad range of common parameter settings as well as symmetric and asymmetric cost functions. Later,

we excluded polynomial kernels from the analysis as their computational performance was to low, leading to execution times of 24 hours and more. We followed the suggestions of [15] to determine the value of the spread parameter in the gaussian kernel.

TABLE 2
EXPERIMENTAL SET-UP INTRODUCING DIFFERENT APPROACHES FOR DATA ENCODING AND INPUT SELECTION TECHNIQUES

label	main group	sub group	resulting number of attributes
A.1	single number encoding for categorical attributes	all attributes included	68
A.2		input selection	44
A.3		input selection & outliner filtering	44
B.1	N-1 encoding for categorical attributes	all attributes included	147
B.2		input selection	84
B.3		input selection & outliner filtering	84
C.1	N encoding for categorical attributes	all attributes included	165
C.2		input selection	89
C.3		input selection & outliner filtering	89

C. *Visualization*

The influence of pre-processing techniques on classification results is compared in classification accuracy, derived from a confusion matrix (a cross-classification of the predicted class against the true class) as calculation of the ratio between correctly classified examples and all examples. However, accuracy based analysis suffers from certain deficits when the underlining class and cost distributions are imbalanced which is the case for most practical problems [16].

Combining a confusion matrix with case dependant misclassification cost is straightforward, leading to a cost-sensitive measure of classification performance. However, the technique of receiver operating characteristics (ROC), provides a more reliable way to compare classification performance [16].

ROC charts are based on the sensitivity *se* and specificity *sp* of a classifier, which can be derived from the confusion matrix as class dependant accuracies. A point (*se*, 1-*sp*) forms one point in ROC-space and evaluating different parameterizations and the corresponding confusion matrixes leads to a ROC-graph which optimal point is the upper left corner. A classifier realizing this point has no errors on the evaluation data set. To enable single number comparison of classifier performance we calculate the geometric mean (G) between *se* and *sp* which strives to maximize the accuracies of each individual class while keeping them balanced and is directly related to a point in ROC-space [17].

D. *Experimental Results on Classifier Performance*

The consolidated main results of the computational experiments are presented in Table 3, comparing the performance of MLP, LVQ and SVM on the generalization set.

For the case of response optimization the sensitivity is of primarily importance, as it measures the amount of correctly classified respondents. The sensitivity of SVM was always higher than 50% and rates of 58% can be regarded as very good for the application domain. For some MLPs and of almost all LVQs the sensitivity is below 50%. The geometric mean exemplifies the dominance of the SVM classifier for almost all experiments. The apparently superior LVQ results on *C.2* and *C.3* are due to a high specificity and therefore inferior to SVM in an economical sense. However, this indicates a possible disadvantage of G as sacrificing specificity to obtain higher sensitivity can be economically sensible while the reverse cannot.

TABLE 3
MAIN RESULTS (CLASSIFICATION RATES ON HOLD-OUT SET [%])

		Group A			Group B			Group C		
		A.1	*A.2*	*A.3*	*B.1*	*B.2*	*B.3*	*C.1*	*C.2*	*C.3*
MLP	sensitivity	49,4	44,8	50,2	51,8	56,0	56,6	18,2	73,0	55,7
	specificity	58,0	62,0	36,5	55,5	55,0	52,9	86,5	38,0	55,4
	G	53,5	52,6	42,8	53,6	55,5	54,7	39,7	52,7	55,6
LVQ	sensitivity	49,8	47,0	53,0	50,0	48,8	39,4	45,5	48,6	34,9
	specificity	55,9	59,1	52,5	55,8	58,9	66,0	72,3	63,7	70,7
	G	52,8	49,3	52,7	52,8	53,6	51,0	**57,4**	**55,6**	49,7
SVM	sensitivity	51,6	51,7	50,9	57,5	58,1	54,2	51,0	52,0	55,6
	specificity	60,5	60,4	61,4	56,4	55,6	58,6	56,5	55,9	57,6
	G	**55,9**	**55,9**	**55,9**	**56,9**	**56,8**	**56,4**	53,6	53,9	**56,6**

Drawing the best SVM, MLP and LVQ classifier for every experiment in ROC-space; see Fig. 1, this dominance is mostly confirmed. For any class and cost distribution the optimal classifier has to lie on the north-west boundary of the convex hull [16]. However, to be economically relevant a classifier has to provide sensitivity higher than 0.5. This region of the convex hull is completely determined by SVM results.

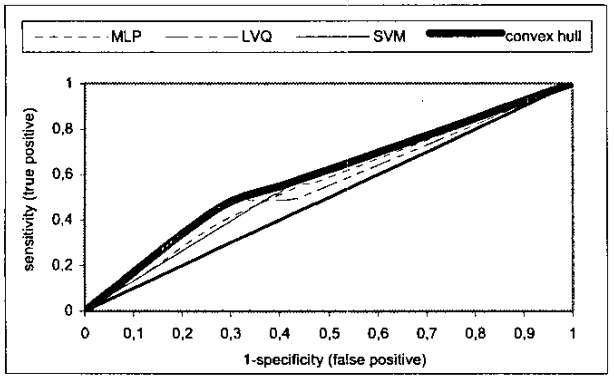


Fig. 1. ROC-Chart of SVM, MLP and LVQ performance in experiment A.1 to C.3, including the resulting convex hull.

The classification performance varies from experiment to experiment, proving the considerable influence of pre-processing issues. Again it is the SVM classifier, which shows the smallest variance between each subgroup and even between different experiments. This robustness to pre-processing issues is a major advantage in business environments since the time and consequently cost to find an appropriate configuration can be reduced significantly.

V. Conclusion

Various different parameter setting has been used, both for ANN and SVM. Our numerical results show, that ANN and SVM are both suitable for the task of response optimization, leading to classification rates that can be considered as very good for practical problems.

Preliminary results with various architectures and data pre-processing configurations show severe differences in performance, especially for MLP and LVQ. SVM seem to dominate in the simulation, concurrently delivering stable results among different architectures and pre-processing configurations. This robustness makes SVM best suited for users who are less experienced in data mining and model building, which is not untypical in business environments. Consequently, we recommend the integration of SVM in standard data mining software packages like SPSS Clementine or SAS Enterprise Miner as the technique is easy to manage and provides competitive results with less parameterization. Verifying the influence of pre-processing issues, further research is needed to find robust data preparation techniques, suitable for aCRM related classification tasks in general.

References

[1] S. Lessmann, "Customer relationship Management," WISU - das Wirtschaftsstudium, vol. 32, pp. 190-192, 2003.

[2] S. F. Crone, "Künstliche neuronale Netze zur betrieblichen Entscheidungsunterstützung," WISU - das Wirtschaftsstudium, vol. 32, pp. 452-458, 2003.

[3] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth, "From Data Mining to Knowledge Discovery in Databases : an overview," AI Magazine, vol. 17, pp. 37-54, 1996.

[4] S. S. Haykin, Neural networks : a comprehensive foundation, 2nd ed. Upper Saddle River, N.J.: Prentice Hall, 1999.

[5] V. N. Vapnik, The Nature of Statistical Learning Theory. New York: Springer, 1995.

[6] C. M. Bishop, Neural networks for pattern recognition. Oxford: Oxford University Press, 1995.

[7] N. Cristianini and J. Shawe-Taylor, An introduction to support vector machines : and other kernel-based learning methods. Cambridge: Cambridge University Press, 2000.

[8] R. D. Reed and R. J. Marks, Neural smithing : supervised learning in feedforward artificial neural networks. Cambridge, Mass.: The MIT Press, 1999.

[9] D. S. Levine, Introduction to neural and cognitive modeling, 2nd ed. Mahwah, N.J.: Lawrence Erlbaum Associates Publishers, 2000.

[10] T. Kohonen, Self-Organizing Maps, 2 ed. Berlin: Springer, 1997.

[11] D. DeSieno, "Adding a Conscience to Competitive Learning," presented at IEEE International Conference on Neural Networks (ICNN '88), San Diego, CA, 1988.

[12] M.-T. Vakil-Baghmisheh and N. Pavesic, "Premature clustering phenomenon and new training algorithms for LVQ," Pattern Recognition, vol. 36, pp. 1901-1912, 2003.

[13] R. Stahlbock, Evolutionäre Entwicklung künstlicher neuronaler Netze zur Lösung betriebswirtschaftlicher Klassifikationsprobleme. Berlin: WiKu, 2002.

[14] C.-C. Chang and C.-J. Lin, "LIBSVM - A Library for Support Vector Machines," 2.6 ed: Software available at http://www.csie.ntu.edu.tw/\verb"~"cjlin/libsvm, 2000.

[15] S. S. Keerthi and C.-J. Lin, "Asymptotic behaviors of support vector machines with Gaussian kernel," Neural Computation, vol. 15, pp. 1667-1689, 2003.

[16] F. Provost and T. Fawcett, "Analysis and Visualization of Classifier Performance: Comparison under Imprecise Class and Cost Distributions," presented at Proceedings of the Third International Conference on Knowledge Discovery and Data Mining, 1997.

[17] M. Kubat and S. Matwin, "Addressing the Curse of Imbalanced Training Sets: One-Sided Selection.," presented at Proceedings of the 14th International Conference on Machine Learning, ICML'97, Nashville, TN, U.S.A., 1997.

SOLVING IMBALANCED CLASSIFICATION PROBLEMS WITH SUPPORT VECTOR MACHINES

Stefan Lessmann
Inst. of Business Information Systems
University of Hamburg, Germany
E-mail: Lessmann@econ.uni-hamburg.de

***Abstract*—The Support Vector Machine (SVM) is a powerful learning mechanism and promising results have been obtained in the field of medical diagnostics and text-categorization. However, successful applications to business oriented classification problems are still limited. Most real world data sets exhibit vast class imbalances and an accurate identification of the economical relevant minority class is a major challenge within this domain. Based upon an empirical experiment, we evaluate the adequacy of SVMs to identify the respondents of a mailing campaign, massively underrepresented in our data set finding SVM to be capable of handling class imbalances in an internal manner providing robust and competitive results when compared to re-sampling methods which are commonly used to account for class imbalances. Consequently, the overall process of data pre-processing is simplified when applying a SVM classifier leading to less time consuming and more cost-efficient analysis.**

Keywords: support vector machine, sampling, imbalanced classification, data mining

I. INTRODUCTION

As technical innovations like the internet have led to a higher market transparency and increasing competition in the last years we observe a shift from the classical, mainly transaction oriented marketing towards a more customer relationship oriented one [1]. In a competitive consumer market customers are regarded as precious business resources and as a result the concept of Customer Relationship Management (CRM) [2], providing companies with techniques and tools to retain and exploit customer relationships, has found increasing consideration in management science.

At the core of CRM we have an analytical component, recording customer centered data in Data Warehouses and analyzing these data stores with mathematical methods originating from various scientific domains like statistics, artificial intelligence and machine learning. A common task within this domain is the prediction of a customer group membership, formally known as classification.[1]

Lately, SVMs [3] have found increasing consideration in the CRM community, providing effective and efficient solutions for managerial problems in similar domains. However, as most CRM related classification tasks involve the prediction of an – often heavily – underrepresented class of interest we evaluate the sensibility of SVM towards class imbalances, striving to exemplify their adequacy for the task of response optimization based upon an empirical, numerical experiment from an ongoing project with a large publishing house.
Following a brief introduction to analytical CRM (aCRM) and the relevance of classification in this domain section 3 assesses the problem of imbalanced class distributions and introduces techniques to overcome this issue. The principles of support vector classification are given in chapter 4 where we focus on SVM's inbuilt capabilities for modeling asymmetric misclassification costs. The suitability of different approaches to deal with imbalanced class distributions for the SVM classifier is evaluated within a numerical experiment in section 5. Conclusions are given in section 6.

II. CLASSIFICATION FOR ANALYTICAL CUSTOMER RELATIONSHIP MANAGEMENT

Whereas the CRM front-office includes techniques and tools for campaign management, sales force automation and service management the analytical back-end consist mainly of a data warehouse to record customer centered transaction and analytical components like online analytical processing and data mining, aiming at the detection of economically relevant information within the data masses in order to achieve operational, tactical and strategic competitive advantages [4].

Among the typical tasks in the field of data mining for aCRM, including regression, classification, segmentation and association, classification analysis is of primary importance with logistic regression and decision trees most widely used in practical applications. Typically, we have a specific class of interest, e.g. a group of customers with high probability of responding to direct mail, and strive to accurately identify these customers among all of our customers using a classifier which has learned a mapping between e.g. demographic and transaction-oriented customer data and the corresponding group membership, previously. Common aCRM tasks like

[1] Throughout this paper, we focus on concept-learning problems in which one class represents the concept at hand (positive class denoted by *B*) while the other represent counter-examples of the concept (negative class denoted by *A*).

response optimization, fraud detection, churn prediction and cross-selling [5] can be cast in this framework. Major challenges within this field are:

- a large number of attributes,
- asymmetric misclassification costs and
- highly imbalanced class distributions.

Having huge amounts of data at hand the problem of large attribute numbers is not that serious, especially as efficient algorithms for subset selection are available and modern classification techniques like SVM are able to provide stable results in high dimensional feature spaces [3] .

However, in order to obtain useful and profitable classification results the fact that the economically relevant group of customers is usually underrepresented in the data has to be considered. In our real world data set, further discussed in section 5, the proportion of relevant customers was for example below two percent, which is absolutely not untypical for this kind of problems. Such imbalances hinder classification and need to be addressed in an appropriate manner.

III. CLASSIFICATION WHEN CLASS DISTRIBUTIONS ARE IMBALANCED

A. Handling class imbalances

When class distributions are imbalanced traditional classification algorithms can be biased towards the majority class due to its over-prevalence [6]. This problem has been observed in various applications, as different as the identification of fraudulent telephone calls [7] and the detection of oil spills in satellite radar images [8]. We can categorize the proposed approaches to deal with imbalanced class distributions in internal ones, which modify existing algorithms to take the class imbalance into consideration, e.g. [8], and external ones that use unmodified existing algorithms, but resample the data in order to diminish the negative effect of the imbalance [9, 10].

Since the sensibility of a classifier for a specific class can be increased by assigning a higher cost of misclassification to this class [11], approaches for cost-sensitive classification are most prominent among the former category. As will be shown in the following section SVM supports this methodology by nature.

On the other hand, resampling aims at the elimination of the over-prevalence of one class, presenting only a selected subset of the available data to the classifier. Basically, this is accomplished either by randomly removing instances of the majority class population (under-sampling), or by randomly duplicating instances from the minority class (over-sampling), until some specified ratio between majority and minority examples is reached. A possible drawback of over-sampling is the fact that the decision region of the minority class becomes very specific, which can lead to over-fitting problems. Consequently, more advanced resampling procedures have been introduced that create synthetic examples of the minority class in the decision space [10] or form a hybrid resampling system by combining over- and under-sampling [12].

B. Measuring classifier performance in imbalanced environments

A major problem caused by class imbalances and ubiquitous in the aCRM area is to find an appropriate indicator to measure classifier performance. It is common practice to visualize the results of a classification analysis by means of a confusion matrix, as show in Table 1.

TABLE 1
Confusion matrix for binary classification problem with output domain *{A,B}*

		predicted		
		A	*B*	**Σ**
actual	*A*	h_{00}	h_{01}	$h_{0.}$
	B	h_{10}	h_{11}	$h_{1.}$
	Σ	$h_{.0}$	$h_{.1}$	*L*

Ordinary performance measures can be derived directly or indirectly from the confusion matrix with accuracy, given as $\frac{h_{00}+h_{11}}{L}$ being the indicator most widely used in practical applications. However, accuracy is known to be inappropriate whenever class and/or cost distributions are highly imbalanced, as it is trivial to obtain a low error rate by simply ignoring the minority class completely, thereby achieving classification accuracy as high as the prior probability of the majority class [13].

Receiver operating characteristics (ROC) analysis is a powerful tool to compare induction algorithms in such imprecise environments and offers the possibility to determine a cost-optimal classifier [14]. A major drawback of ROC analysis is the fact, that it does not deliver a single, easy to use performance measure like accuracy directly. An alternative is to use the area under the ROC curve (AUC) for single number evaluation [15] or the geometric mean of sensitivity and specificity [16] instead. This leads to the measure $G = \sqrt{sensitivity * specificity} = \sqrt{\frac{h_{11}}{h_{1.}} * \frac{h_{00}}{h_{0.}}}$, which strives to maximize the accuracies of each individual class while keeping them balanced and is directly related to a point in ROC-space [9].

Other prominent performance metrics include:

Precision $P = \frac{h_{11}}{h_{.1}}$; corresponding to the proportion of examples classified as positive that are truly positive and Recall $R = \frac{h_{11}}{h_{1.}}$; giving the proportion of truly positives examples that were correctly classified and being identical to sen-

sitivity. P and R are both desirable and typically trade off against each other so that it is convenient to combine them in a single measure called the F-measure (F) [17] which is calculated as $F = \frac{2PR}{(P+R)}$.[2] Being closely related F and G are both not influenced by imbalanced class distributions and thus generally applicable in our domain. However, we expect F to be the more important indicator, as it focuses more directly on one particular class, e.g. class B, which is consistent with typical demands in the field of aCRM related classification.

IV. SUPPORT VECTOR MACHINES FOR CLASSIFICATION

The original SVM can be characterized as a supervised learning algorithm capable of solving linear and non-linear classification problems. The main building blocks of SVM are structural risk minimization, non-linear optimization and duality as well as kernel induced features spaces, underlining the technique with an exact mathematical framework [18].

The main idea of support vector classification is to separate examples with a linear decision surface and maximize the margin of separation between the two different classes.

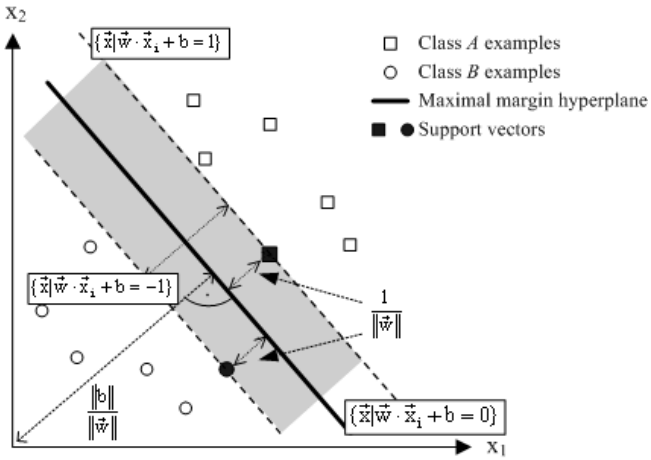


Fig. 1: Maximal margin hyperplane for discriminating between two classes [19]

The idea to construct a separating hyperplane with maximal margin leads to the well known soft-margin optimization problem [18].

$$\begin{aligned} & min_{\vec{w},\xi,b} \frac{1}{2}\|\vec{w}\|^2 + C\sum_{i=1}^{L}\xi_i \\ \text{s.t.} \quad & k_i(\vec{w}\cdot\vec{x}_i + b) \geq 1 - \xi_i \quad \forall i = 1,...,L \\ & \xi_i \geq 0 \quad \forall i = 1,...,L \end{aligned} \tag{1}$$

where L denotes the number of training examples, $\vec{x}_i$ represents the attribute vector of example i, $k_i \in \{0,1\}$ is the class label of example i and C is a constant cost parameter, enabling the user to control the trade-off between learning error and model complexity, given by the margin of the separating hyperplane [3]. The slack variables ξ_i accounts for the fact, that the training data is not necessarily linearly separable, such that some examples will be misclassified by a linear discriminant function.

Data points closest to the maximal margin hyperplane, that is points satisfying $k_i(\vec{w}\cdot\vec{x}_i + b) = 0$, are called (bounded) support vectors as they define the position of the separating plane; see Fig. 1. Consequently, the solution of a support vector classifier depends only on a (possibly very) small number of training examples, the support vectors, and removing all other instances from the training set would leave the solution unchanged. From this understanding of a support vector we could expect SVM to be insensitive to imbalanced class distributions since there should always be a sufficient number of examples from each class to form a reasonable support vector set [11]. However, our experiment reveals that this assumption is not true.

Problem (1) forms the basis for SVM classification and an internal modification to account for imbalanced class distributions by means of asymmetric misclassification cost is straightforward. A simple revision of the objective function gives

$$\begin{aligned} & min_{\vec{w},\xi,b} \frac{1}{2}\|\vec{w}\|^2 + C_+\sum_{k_i=1}\xi_i + C_-\sum_{k_i=0}\xi_i \\ \text{s.t.} \quad & k_i(\vec{w}\cdot\vec{x}_i + b) \geq 1 - \xi_i \quad \forall i = 1,...,L \\ & \xi_i \geq 0 \quad \forall i = 1,...,L \end{aligned} \tag{2}$$

providing two independent cost parameters while leaving the overall algorithm almost unchanged. Formulation (2) is the one incorporated in the SVM solver LIBSVM [20] which we used for our study.

For constructing more general non-linear decision surfaces than hyperplanes, SVMs implement the idea to map the input vectors into a high-dimensional feature space Ψ via an a priori chosen non-linear mapping function $\Phi: X \to \Psi$. The construction of a separating hyperplane in this features space leads to a non-linear decision boundary in the original space; see Fig. 2. Expensive calculation of dot products $\Phi(\vec{x}_i)\cdot\Phi(\vec{x}_j)$ in a high-dimensional space can be avoided by introducing a kernel function $K(\vec{x}_i,\vec{x}_j) = \Phi(\vec{x}_i)\cdot\Phi(\vec{x}_j)$ [3].

Therewith, SVM enable a considerably easier parameterization when compared to other learning machines like for example multi-layer perceptron neural networks [21]. The only degrees of freedom are the selection of a kernel function together with corresponding kernel parameters and the choice of the cost parameter C or C_+ and C_-, respectively.

[2] Although not widely used, the geometric mean of P and R is a suitable performance metric as well.

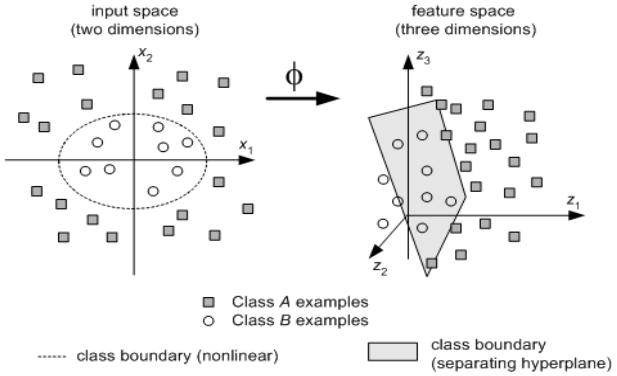


Fig. 2. Non-linear -mapping from two-dimensional input space with non-linear class boundaries into a linear separable feature space

V. SIMULATION EXPERIMENT OF SUPPORT VECTOR SENSIBILITY TO CLASS IMBALANCES

A. Objective

A broader adoption of SVM in the field of aCRM related problems is just beginning and in order to become a major classification technique within this particularly difficult domain SVM has to proof empirically its capability of handling highly imbalanced data sets. For the SVM classifier the question if imbalances have to be adjusted and which method, e.g. internal or external approaches, is preferable has to our best knowledge not been answered by now as most research in this field is based on decision trees or artificial neural networks [6, 9, 11, 12]. Thus, we evaluate SVM's capabilities to address class imbalances internally in comparison to external balancing by resampling within an empirical, numerical study.

In our experiment, we consider the case of response optimization as a representative example for aCRM related classification. The goal of response optimization is to identify a subset of customers who exhibit a substantially higher probability of reacting to a certain offer than the average customer, based on experiences from past campaigns. Here, the cost of making an offer to a person who does not respond is typically small compared to the cost of not contacting a customer, who would otherwise have ordered an item. The imbalance is introduced as usually only a very small group of people who were contacted purchase a product.

B. Experimental setup

Our data is based on a mailing campaign which included 300,000 addresses and aims at selling an additional magazine subscription to customers who have already subscribed to at least one periodical. The response rate of this campaign was 1.3%, meaning that only 4019 customers showed a positive reaction.

In order to discriminate these economically relevant customers from all others, the data contains 50 numerical as well as categorical attributes, which provide demographic and transactional information about each customer. While numerical attribute value were scaled to the interval [-1;1] using a linear transformation we applied one-of-N remapping to account for discrete attributes [22].

Following, we randomly selected 100,000 records as a hold-out set to enable out of sample validation on unseen data. The remaining 200,000 customer records formed the training data and were used in five different training scenarios, as is described in Table 2 where the class label B denotes the group of customers, who responded to a previous campaign.

Experiment 1 consists of a randomly selected sub-sample of 10,000 records of the available training data. Here, it is left to the SVM to adjust the imbalance internally. Under-sampling leads to experiment 2 and 3 where all class *B* records of the training data base together with some randomly selected class *A* records were used, so that we obtain a class *B* to *A* ratio of 1:2 and 1:1, respectively.[3] Within the remaining two experiments over-sampling was used to achieve the same class ratios of 1:2 and 1:1. That is, the 2,693 class *B* records within the available training data where randomly duplicated until the respective target ratios between class *B* and class *A* records were reached.

TABLE 2
Setup for numerical evaluation of SVM's sensibility towards imbalanced class distributions

Experiment No.	training data partitioning				
	records class *A*		records class *B*		records
	number	percent	number	percent	total
1	9885	98,85	115	1,15	10000
2	5368	66,67	2963	33,33	8079
3	2693	50,00	2693	50,00	5368
4	9885	66,67	4942	33,33	14827
5	9885	50,00	9885	50,00	19770

Concerning SVM parameterization, we refused to use polynomial kernel functions as several pre-test revealed their computational inefficiency and incorporated linear and gaussian kernels instead [18]. It is common practice to use the same value for C_+ and C_- when class distributions are balanced and we will denote this as symmetric costing (SC). A correspondingly parameterized SVM will be called symmetric costing support vector machine (SC-SVM). We expect SC to provide competitive performance when class imbalances are externally adjusted through resampling and consequently included according classifiers in our study varying *log(C)* stepwise from -3 to 4. If on the other hand imbalances are not

[3] To ensure comparability, the class A records were fixed throughout all experiments. That is, experiment 2 and 3 used a randomly selected sub-sample of the class A records that were used in experiment 1, 4 and 5.

externally adjusted, as is done in experiment 1, we can hardly expect SC-SVM to deliver reasonable classification results. Therefore, we incorporated SVM with asymmetric costing (ASC-SVM) as well and evaluated 20 parameter settings for C_- in the range of 0.001 to 0.02 while leaving C_+ fixed at 1.

Since the kernel width σ can have a crucial impact on the classification ability of the gaussian SVM [23] we evaluated six different settings ($\sigma = \{0.05; 0.075; 0.1; 0.125; 0.3; 0.5\}$) for any cost parameterization.

Combining all parameter settings for the linear and the gaussian SVM, we obtain a total number of more than 130 classifiers which were evaluated for every experiment.

C. Results

Our study revealed that differences in classifier performance between individual experiments are not severe. However, we have to keep in mind that even a small difference can have a noticeable monetary impact in economical environments. The maximal observed performance within each experiment by means of F and G is given in Table 3.

TABLE 3
Maximal observed performance by means of F and G

Exp. No.	best observed results			
	F	rank	G	rank
1	**0,0306**	**1**	0,5126	3
2	0,0294	2	0,5207	2
3	0,0281	4	0,4631	5
4	0,0286	3	**0,5213**	**1**
5	0,0279	5	0,4966	4

The unadjusted experiment 1 delivered quite competitive results indicating that SVM is indeed capable of handling imbalanced data sets by assigning different cost parameters to each class. This is a promising result as re-sampling complicates the overall data mining process and is therefore time- and cost-consuming. Detailed results for experiment 1 are given in Fig. 3

Surprisingly, the linear SVM seems to dominate non-linear classifiers with gaussian kernel when G is used as performance indicator but this result is not confirmed by the probably more important F-measure. Regarding F we find that only a small range of C_- between 0.012 and 0.015 delivers utilizable classification results indicating the particular challenge of analyzing complex real world data sets.

For the adjusted data sets (experiment 2-5) ASC leads to naive classifiers where all instances were classified as belonging to class B. This is probably due to the considered ratio between C_+ and C_-. However, the idea of re-sampling is to avoid internal balancing so that we compare different re-sampling techniques for the SC-SVM.[4]

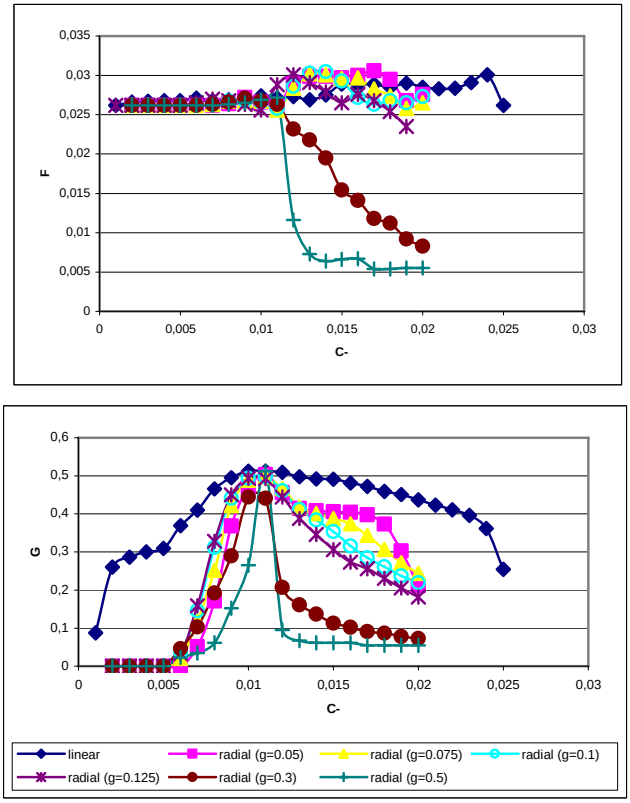


Fig. 3: Results for different linear and radial ASC-SVM in experiment 1 by means of F and G

Regarding the poor performance of experiment 1 SC is obviously inappropriate when class imbalances are not adjusted. This proofs that SVM is indeed sensitive to imbalanced class distributions and contrasts the results of [11]. Surly, this is due their univariate experiment design which is not representative for real world aCRM problems.

While for experiment 3 and 5 with completely balanced class distributions we can select very small values for the cost parameter C this leads to naïve classification in all remaining experiments with imbalanced class distributions. Considering the SVM objective function (1) a low value for C results in a classifier which focuses primary on margin maximization instead of accuracy. Hence, if data similarity and the risk of over-fitting is increased, e.g. through over-sampling, SVM naturally compensates this by enabling lower settings for C leading to robust classifiers with large margin of separation and improved generalization ability.

[4] SVM with linear kernel showed the same trend over all experiments on a slightly lower performance level and were therefore excluded for clarity. We report results for the radial SVM with σ =0.05 as it consistently delivered superior performance.

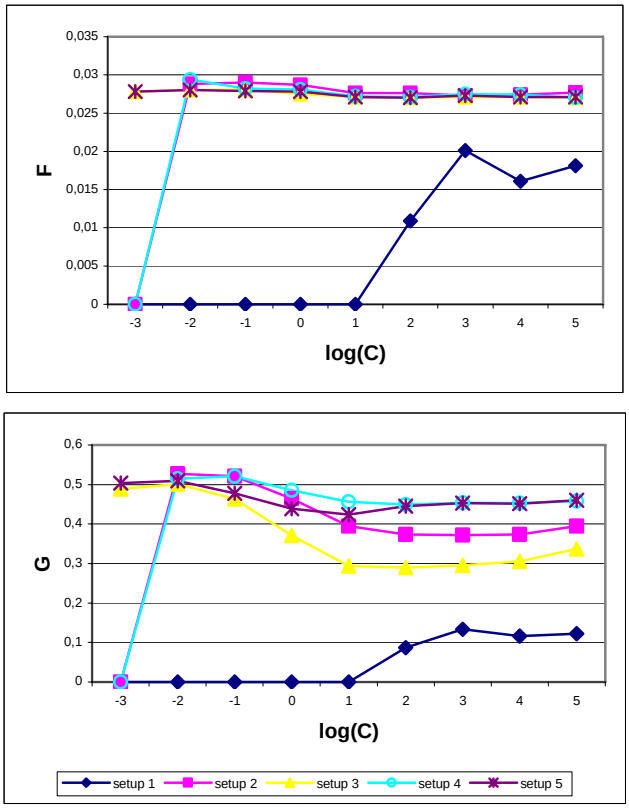


Fig. 4: Performance of radial SVM by means of F and G for all experiments

VI. CONCLUSION

We analyzed the problem of imbalanced class distributions in the field of aCRM related classification exemplifying the need to regard this issue during classification analysis by means of internal or external data adjustments and suitable performance metrics.

Our experiment revealed that the SVM classifier is able to account for class imbalances in an internal manner through according parameterization within the model selection stage. Consequently, the data pre-processing phase, preceding any data mining analysis, can be simplified significantly when the SVM classifier is used.

On the other hand, internal modifications are usually not reusable among different classification algorithms [12] and therefore complicate the comparison of different methods. If such a comparison is desirable, e.g. to determine a superior algorithm for a specific problem, it is wiser to account for imbalances externally through re-sampling. Our experiment revealed that SVM is robust towards re-sampling methods, working with under- and over-sampling alike. However, when applied in conjunction with under-sampling SVMs provide competitive results while using considerably less records leading to an increased computational efficiency.

We restricted our analysis to basic re-sampling techniques randomly downsizing the majority class and randomly upsizing the minority class, respectively. More elaborate approaches are proposed in [9, 10, 12] and the question if ASC-SVM is still competitive to re-sampling when such techniques are applied and if the potential gain in classification performance would justify the additional sampling effort under economical considerations needs further research.

We used F and G to measure classification performance in imprecise environments which is consistent with other research conduced in this field [6, 11, 12, 24]. Though both measures are not influenced by class distributions it is questionable if they are ideal for the field of aCRM where the minority class is generally of primary importance. Hence, it can be economically sensible to sacrifice precision in order to achieve a higher recall and the F-measure will give poor advice on classifier selection. The same argumentation holds for G so that it seems worthwhile to investigate the question of an economical performance metric for classification analysis in future research.

REFERENCES

[1] M. Bruhn, *Relationship marketing : management of customer relationships*. Harlow [u.a.]: Financial Times Prentice Hall, 2002.

[2] S. Lessmann, "Customer Relationship Management," *WISU - das Wirtschaftsstudium*, vol. 32, pp. 190-192, 2003.

[3] V. N. Vapnik, *The Nature of Statistical Learning Theory*. New York: Springer, 1995.

[4] A. Berson, S. Smith, and K. Thearling, *Building Data Mining Applications for CRM*. New York: McGraw Hill, 1999.

[5] H. Hippner and K. D. Wilde, "Data Mining im CRM," in *Effektives Customer Relationship Management*, S. Helmke, M. Uebel, and W. Dangelmaier, Eds., 2 ed. Wiesbaden: Gabler, 2002, pp. 211-232.

[6] N. V. Chawla, "C4.5 and Imbalanced Datasets: Investigating the effect of sampling method, probabilistic estimate, and decision tree structure," presented at ICML Workshop on Learning from Imbalanced Datasets II, Washington DC, 2003.

[7] T. Fawcett and F. J. Provost, "Adaptive Fraud Detection," *Data Mining and Knowledge Discovery*, vol. 1, pp. 291-316, 1997.

[8] M. Kubat, R. C. Holte, and S. Matwin, "Machine Learning for the Detection of Oil Spills in Satellite Radar Images," *Machine Learning*, vol. 30, pp. 195-215, 1998.

[9] M. Kubat and S. Matwin, "Addressing the Curse of Imbalanced Training Sets: One-Sided Selection.," presented at Proceedings of the 14th International Conference on Machine Learning, ICML'97, Nashville, TN, U.S.A., 1997.

[10] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic Minority Over-sampling Technique," *Journal of Artificial Intelligence Research*, vol. 16, pp. 321-357, 2002.

[11] N. Japkowicz and S. Stephen, "The class Imbalance Problem: A Systematic Study," *Intelligent Data Analysis*, vol. 6, pp. 429-450, 2002.

[12] A. Estabrooks, T. Jo, and N. Japkowicz, "A Multiple Resampling Method for Learning from Imbalanced Data

Sets," *Computational Intelligence*, vol. 20, pp. 18-36, 2004.

[13] S. Lawrence, I. Burns, A. D. Back, A. C. Tsoi, and C. L. Giles, "Neural Network Classification and Prior Class Probabilities," in *Neural Networks: Tricks of the Trade*, vol. 1524, *Lecture Notes in Computer Science*, G. B. Orr and K.-R. Müller, Eds. Heidelberg: Springer, 1998, pp. 299-313.

[14] F. Provost and T. Fawcett, "Analysis and Visualization of Classifier Performance: Comparison under Imprecise Class and Cost Distributions," presented at Proceedings of the Third International Conference on Knowledge Discovery and Data Mining, 1997.

[15] A. P. Bradley, "The use of the area under the ROC curve in the evaluation of machine learning algorithms," *Pattern Recognition*, vol. 30, pp. 1145-1159, 1997.

[16] M. Kubat, R. C. Holte, and S. Matwin, "Learning when Negative Examples Abound," presented at Proceedings of the 9th European Conference on Machine Learning ECML'97, Prague, Czech Republic, 1997.

[17] C. J. Van Rijsbergen, *Information retrieval*, 2d ed. London ; Boston: Butterworths, 1979.

[18] N. Cristianini and J. Shawe-Taylor, *An introduction to support vector machines : and other kernel-based learning methods*. Cambridge: Cambridge University Press, 2000.

[19] C. J. C. Burges, "A Tutorial on Support Vector Machines for Pattern Recognition," *Data Mining and Knowledge Discovery*, vol. 2, pp. 121-167, 1998.

[20] C.-C. Chang and C.-J. Lin, "LIBSVM - A Library for Support Vector Machines," 2.6 ed: Software available at http://www.csie.ntu.edu.tw/\verb"~"cjlin/libsvm, 2001.

[21] S. F. Crone, S. Lessmann, and R. Stahlbock, "Empirical Comparison and Evaluation of Classifier Performance for Data Mining in Customer Relationship Management," presented at Proceedings of the IEEE 2004 International Joint Conference on Neural Networks (IJCNN), Budapest, Hungary, 2004.

[22] D. Pyle, *Data preparation for data mining*. San Francisco, Calif.: Morgan Kaufmann Publishers, 1999.

[23] S. S. Keerthi and C.-J. Lin, "Asymptotic behaviors of support vector machines with Gaussian kernel," *Neural Computation*, vol. 15, pp. 1667-1689, 2003.

[24] F. Provost, T. Fawcett, and R. Kohavi, "The Case Against Accuracy Estimation for Comparing Induction Algorithms," presented at Proceedings of the Fifteenth International Conference on Machine Learning, San Francisco, 1998.

Robert Stahlbock
Stefan Lessmann

Potential von Support Vektor Maschinen im analytischen Customer Relationship Management

1 Zusammenfassung

Globalisierung und die technischen Möglichkeiten des Internet haben den Wettbewerbsdruck für viele Unternehmen spürbar erhöht. Hinzu kommt, dass sich Produkte in ihren funktionalen und qualitativen Eigenschaften, zumindest was die Wahrnehmung der Konsumenten betrifft, immer stärker angleichen, so dass eine Differenzierung gegenüber Wettbewerbern fast ausschließlich über Zusatzleistungen zu erfolgen hat. Das Konzept des Customer Relationship Management (CRM) bietet in dieser Situation geeignete Handlungsstrategien an, um den Wert einer Kundenbeziehung für Unternehmen und Kunde zu erhöhen. Die Wirksamkeit von CRM-Aktivitäten hängt dabei maßgeblich von einem detaillierten Wissen über die Präferenzen und Interessen aktueller sowie potentieller Kunden ab. Zum Aufbau einer solchen Wissensbasis bedarf es eines leistungsfähigen analytischen Instrumentariums, dessen Aufgabe die Integration, Konsolidierung und Auswertung aller kundenrelevanten Datenbestände ist. Die Bereitstellung einer abteilungsübergreifenden kundenbezogenen Datenbank wird durch Werkzeuge aus dem Data Warehouse Bereich bereits umfassend unterstützt. In Ermangelung geeigneter Analysemethoden unterbleibt eine gezielte Auswertung dieser Datenbestände jedoch oftmals und Verbesserungspotentiale bleiben ungenutzt. Vor diesem Hintergrund ist es das Ziel der vorliegenden Arbeit ein neues, viel versprechendes Verfahren, die so genannte Support Vektor Maschine, vorzustellen, welches nach Auffassung der Autoren einen erheblichen Beitrag zur Ausschöpfung dieser Potentiale leisten könnte.

2 Analytisches Customer Relationship Management

2.1 Überblick

Der Begriff Customer Relationship Management beschreibt eine kundenorientierte Managementphilosophie, welche den Aufbau und die Pflege langfristiger und profitabler Kundenbeziehungen verfolgt und für die kontinuierliche Verbesserung kundenbezogener Geschäftsprozesse einen ganzheitlichen Einsatz von Informations- und Kommunikationstechnologie vorsieht.[1]

---

[1] Alternative Definitionen finden sich z. B. bei: SCHULZE (2000), S. 18; SCHMID (2001), S. 11 f.; RAAB+ (2002), S. 11; HIPPNER+ (2002a), S. 6; HOLLAND+ (2001), S. 20.

Das CRM-Konzept ist als Strategie zur Reaktion auf veränderte Umweltbedingungen entstanden, mit denen sich Unternehmen in der heutigen Zeit, insbesondere auf ihren Absatzmärkten, konfrontiert sehen. Als Beispiele sind hier tendenziell höhere Kundenerwartungen bei abnehmender Kundenloyalität sowie ein steigender Wettbewerbs- und Kostendruck zu nennen.[1] Hieraus ergibt sich die elementare Zielsetzung, Kundenbeziehungen langfristig und profitabel zu gestalten.[2] Eine stärkere Individualisierung der Kundenbeziehung etabliert sich zunehmend als ein Mittel, um dies zu erreichen.[3] Der Kunde profitiert dabei von einem maßgeschneiderten Leistungsangebot, was sich positiv auf seine Zufriedenheit und Loyalität auswirken sollte. Voraussetzung für eine solche Individualisierung ist ein detailliertes Wissen über die eigenen Kunden, welches Unternehmen ihrerseits für eine Kundenbewertung und korrespondierende Zuteilung knapper Marketing- und Betreuungsressourcen verwenden können. Eine solche profitorientierte Kundensegmentierung entspricht den Grundgedanken von CRM.

Ein wesentliches Merkmal von CRM ist der Versuch, ein in sich geschlossenes logisches Kreislaufsystem zu implementieren, welches abteilungsübergreifend sämtliche kundenrelevanten Geschäftsprozesse integriert und in dessen Rahmen Daten in Informationen bzw. Wissen[4] transformiert und daraus Handlungsstrategien abgeleitet werden. Dabei werden die drei Ebenen kollaboratives, operatives und analytisches CRM unterschieden.[5] Deren Zusammenspiel zeigt Abbildung 1.

---

1 HOLLAND+ (2001), S. 14 ff.; SCHULZE (2002), S. 235 f.

2 Die Langfristigkeit kann dabei als Subziel aufgefasst werden, da allgemein eine positive Korrelation zwischen Kundenbindungsdauer und -profitabilität unterstellt wird. KOTLER+ (1995), S. 74 ff.; BERSON+ (1999), S. 42; RAAB+ (2002), S. 14 f.

3 In diesem Zusammenhang wird auch von Customized Marketing, Mass Customization oder One-to-One Marketing gesprochen. Vgl. z. B. SCHULZE (2000), S. 14; LINK+ (1997), S. 17; BERSON+ (1999), S. 301.

4 Unter Daten wird hierbei eine Folge von Zeichen verstanden. In einem bestimmten Kontext interpretiert werden diese zu Informationen. Informationen dienen der Bildung von Wissen, dessen wesentliche Eigenschaft die Zweckorientiertheit ist. Vgl. VOSS+ (2001), S. 24; FINK+ (2001), S. 68 f.

5 HIPPNER+ (2002a), S. 14 f.; SCHULZE (2002), S. 237 ff.; BERSON+ (1999), S. 45.

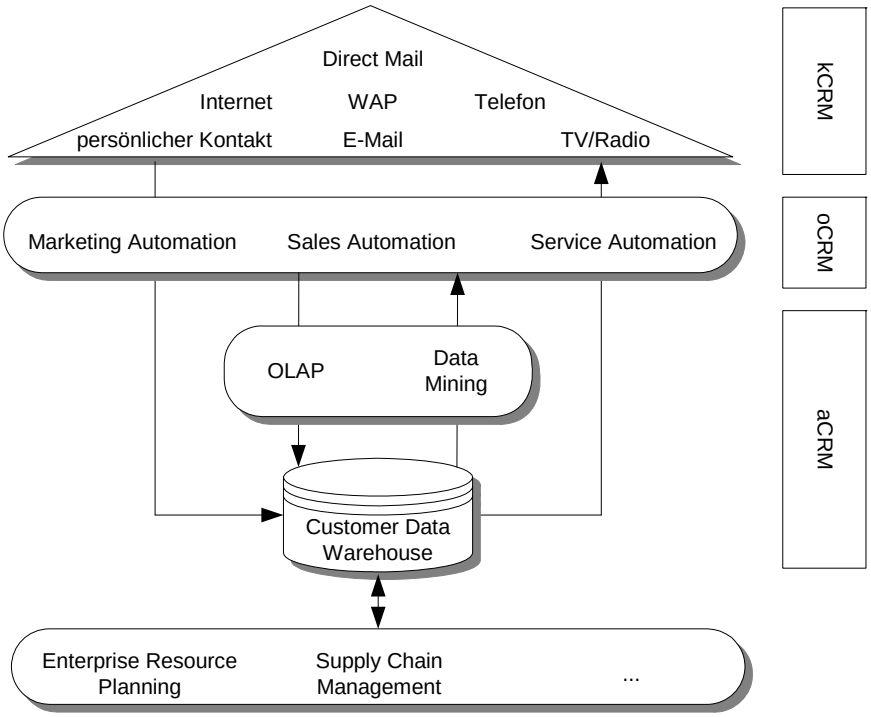


Abbildung 1: CRM-Architektur
(in Anlehnung an HIPPNER+ (2002b), S. 213)

Das operative CRM (oCRM) umfasst Lösungen zur abteilungsübergreifenden Abwicklung und Abstimmung sämtlicher Aktivitäten an den zentralen Customer Touch Points Marketing, Verkauf und Service. Der Dialog zwischen Kunde und Unternehmen sowie entsprechende Geschäftsprozesse werden unterstützt. In diesen Bereich fallen z. B. Verkaufsgespräche oder die Beantwortung von Kundenanfragen zu Lieferterminen oder Produktverfügbarkeiten.

Kollaboratives CRM[1] (kCRM) beinhaltet die Bereitstellung, Steuerung und Synchronisation verschiedener Kommunikationskanäle zum Kunden (Telefon, Fax, E-Mail etc.). Ziel ist die Sicherung konsistenter Informationen und einheitlicher Servicelevel quer über alle Kanäle.[2]

Die systematische Aufzeichnung und Auswertung aller Kundenkontakte und -reaktionen ist Gegenstand des analytischen CRM (aCRM).[3] Die Daten, welche im Rahmen operativer Tätigkeiten anfallen und konsolidiert werden, sollen durch Anwendung von Online Analytical Processing (OLAP) und Data Mining in kundenbezogenes Wissen transformiert werden.

---

1 Synonym wird z.T. auch von kommunikativen CRM gesprochen.
2 FINK+ (2001), S. 210.
3 HIPPNER+ (2002a), S. 15.

2.2 Aufgaben und Komponenten

Ziel von aCRM ist die Generierung von Wissen über eigene und potentielle Kunden, welches zur Verbesserung operativer kundennaher Geschäftsprozesse verwendet werden kann. Eine wesentliche Aufgabe besteht in der Zusammenführung sämtlicher kundenrelevanter Daten in einer integrierten Datenbank, einem so genannte Customer Data Warehouse.[1] Diese Konsolidierung ist ausgesprochen komplex, da die benötigten Daten typischerweise auf viele historisch gewachsene Insellösungen zur Unterstützung von Marketing, Verkauf und Service (z. B. Computer Aided Selling, Online-Datenbanken, Sales Force Automation-Systeme, Call Center etc.) verteilt sind. Weiterhin ist eine Integration mit betriebswirtschaftlichen Standardsoftwaresystemen (Enterprise Resource Planning, Supply Chain Management) erforderlich, welche ebenfalls wichtige Daten enthalten.[2] Typische Inhalte eines Customer Data Warehouse sind beispielsweise[3]:

- Kundenstammdaten (Adressdaten, Demographie, Mikrogeographie etc.)
- Kaufhistorien: Wann wurde was von wem wie oft gekauft?
- Aktionsdaten: Wann wurde wer auf welche Art kontaktiert?
- Reaktionsdaten: Wer hat wie auf einen Kontakt reagiert? Wer hat sich worüber beschwert?

Erst diese Datenintegration ermöglicht eine ganzheitliche Sicht auf einzelne Kunden bzw. Kundengruppen, da alle Unternehmensbereiche nur noch auf eine logische Datenbank zugreifen.

Die strukturierte Datenspeicherung ist der Ausgangspunkt für weitergehende Analysen, wobei im Rahmen von aCRM hauptsächlich OLAP und Data Mining eingesetzt werden.[4] OLAP-Systeme bilden betriebswirtschaftlich relevante Maßgrößen (Umsatz, Absatzzahlen, Kosten) in Form eines multidimensionalen Datenwürfels ab.[5] Die Dimensionen dieses Würfels werden durch betriebswirtschaftlich relevante Gliederungskriterien (Produktgruppe, Kundengruppe, Vertriebsregionen) gebildet.[6] Eine typische Fragestellung wäre z. B. „Wie hoch war der Ab-

---

1 Synonym werden auch die Begriffe Customer Database beziehungsweise Customer Centered Database verwendet. BERSON$^+$ (1999), S. 46; KOTLER$^+$ (1997), S. 495.
2 FINK$^+$ (2001), S. 210.
3 RAPP$^+$ (1999), S. 257; HIPPNER$^+$ (2002a), S. 15.
4 Zu den Komponenten von aCRM vgl. auch Abbildung 1.
5 VOSS$^+$ (2001), S. 266 ff.
6 HIPPNER$^+$ (2002a), S. 16 f.

satz von Produkt X im Zeitraum Y in der Vertriebsregion Z?“. Die Antwort entspricht einer Zelle in einem dreidimensionalen OLAP-Würfel mit den Kanten Produkt, Zeit und Region.[1] Ist der Anwender lediglich in der Lage Hypothesen zu formulieren, ohne eine genaue Kenntnis über Wirkungszusammenhänge zu besitzen (z. B. „der Wert eines Kunden wird durch die Merkmale Alter, Geschlecht und Einkommen beeinflusst“), kann Data Mining zur Aufdeckung geschäftsrelevanter Muster in den Daten verwendet werden.

2.3 Untersuchungsgegenstände

Es sind eine Vielzahl von Untersuchungen denkbar, die im Sinne von aCRM interessante Erkenntnisse über einzelne Kunden oder Kundengruppen liefern und nach der Art bzw. Phase der Beziehung zwischen Kunde und Unternehmen gegliedert werden können.[2] Die meisten der nachfolgenden Untersuchungen werden mit Methoden und Techniken aus dem Bereich Data Mining vorgenommen.[3] Grundsätzlich werden dabei historische Daten genutzt, um gewisse Verhaltensmerkmale eines Kunden (z. B. die Neigung zur Reaktion auf Direct Mail oder die Kreditwürdigkeit) zu modellieren. Die in eine solche Analyse einfließenden Daten werden auch als Merkmale, Variablen oder Attribute bezeichnet.[4]

Die Akquisition von Neukunden kann durch Data Mining z. B. dergestalt unterstützt werden, dass für vergangene Kampagnen untersucht wird, welche Zielgruppen überproportional häufig reagiert haben (so genannte Responseanalyse).[5] Werden Werbungsaktionen im Folgenden ausschließlich auf entsprechende Gruppen beschränkt, kann die Effizienz von Akquisitionskampagnen deutlich gesteigert werden. Die Abgrenzung einer geeigneten Zielgruppe kann auch durch eine Gruppierung des aktuellen Kundenstamms unterstützt werden. Dabei wird verstärkt versucht, Kunden zu gewinnen, die beispielsweise hinsichtlich demographischer Merkmale eine große Ähnlichkeit mit Bestandskunden eines besonders profitablen Segments aufweisen.[6] Ebenfalls dieser Phase zuzurechnen sind Modelle zur Bewertung von Interessenten bzw. potentiellen Kunden. Typische Beispiele sind hier das Antragsscoring bei Versicherungen oder Bonitätsprüfungen bei der Kreditvergabe.

---

1 Für Details zu OLAP vgl. z. B. CHAMONI (2001).

2 Im Wesentlichen lassen sich die Phasen potentieller Kunde (Interessent), Neukunde, Bestandskunde und verlorener/zurück gewonnener Kunde unterscheiden. Vgl. z. B. HUNSEL$^+$ (2000), S. 116 und Abbildung 2.

3 Vgl. auch Kapitel 3, insbesondere Abschnitt 3.3.

4 HAND$^+$ (2001), S. 4.

5 HIPPNER$^+$ (2002b), S. 224.

6 Hierfür kommen segmentierende und clusterbildende Verfahren in Betracht, die auf S. 12 näher beschrieben werden.

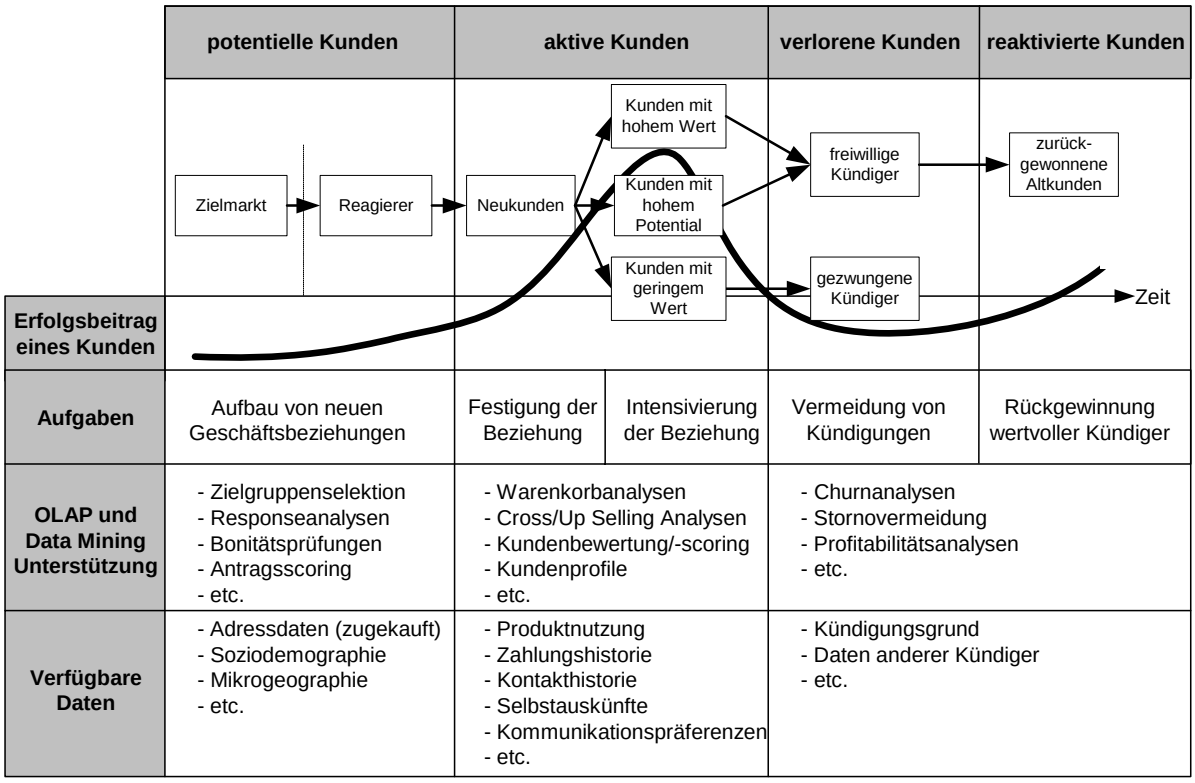


Abbildung 2: Analytisches CRM im Lebenszyklus von Kundenbeziehungen
(in Anlehnung an HIPPNER[+] (2002b), S. 222)

Profitabilitätsanalysen sind auch für Bestandskunden relevant, z. B. um eine potentialadäquate Zuteilung von Marketingressourcen vornehmen zu können. Hierfür kommen klassische Methoden der Kundenbewertung wie ABC-Analysen, Scoringverfahren und Kundenportfolios ebenso in Betracht wie umfassendere deskriptive oder statistische Modelle.[1] Kundenbewertungen stellen die Grundlage für kundenspezifische Marketing-, Vertriebs- und Servicekonzepte dar.[2] Von großer Bedeutung sind in diesem Zusammenhang auch so genannte Cross und Up Selling Analysen, welche das Produktnutzungsverhalten von Kunden untersuchen.[3] Werden z. B. zwei Produkte häufig gemeinsam erworben, kann Käufern des einen Produktes gezielt ein Angebot für das andere unterbreitet werden (Cross Selling).[4] Ziel des Up Selling ist es, die Nutzungsfrequenz eines Produktes zu erhöhen oder dem Kunden ein höherwertiges Produkt zu verkaufen. Cross und Up Selling Analysen liefern somit eine Information über das zukünftig zu erwartende Potential eines Kunden. Solche Aspekte sollten bei der Kundenbe-

1 KRAFFT (2002); RAAB[+] (2002); SMIDT[+] (2001). Einen guten Überblick liefert BEYER (2003).
2 HIPPNER[+] (2002b), S. 226.
3 BERSON[+] (1999), S. 264; RUD (2001), S. 10.
4 Typisch ist dies z. B. für den Versicherungsbereich. Hier kann häufig beobachtet werden, dass Kunden mehrere Policen bei ein und demselben Versicherer abschließen.

wertung berücksichtigt werden, auch wenn eine exakte Quantifizierung zumeist nicht möglich ist.[1]

Beobachtungen des Kundenverhaltens zeigen, dass Kunden heutzutage verstärkt bereit sind, eingegangene Geschäftsbeziehungen aufzukündigen und Anbieter zu wechseln. Angesichts der Investitionen, die im Rahmen von Akquisitionsmaßnahmen in einen Kunden getätigt werden und der positiven Korrelation zwischen Kundenbindungsdauer und Profitabilität,[2] ist der Einsatz von Präventionsmaßnahmen sinnvoll. Hier setzen so genannte Churn- oder Stornoanalysen an, welche versuchen, abwanderungswillige Kunden innerhalb des Kundenstamms zu identifizieren.[3] Anschließend können gezielt Maßnahmen initiiert werden, um einer tatsächlichen Abwanderung vorzubeugen (Sonderangebote, verbesserte Vertragsbedingungen etc.). Wesentliche Determinante für das Ausmaß solcher Aktionen ist wiederum der individuelle Kundenwert. Neben attraktiven Konkurrenzangeboten, ist die Unzufriedenheit mit einem Produkt oder einer Serviceleistung eine der Hauptursachen für Kunden, eine Geschäftsbeziehung abzubrechen. Um Kündigungen aufgrund von Unzufriedenheit vorzubeugen, kommt dem Beschwerdemanagement eine große Bedeutung zu. Zum einen können hier wertvolle Informationen über eventuelle Unzulänglichkeiten eines Produkts gewonnen und, im Sinne einer kontinuierlichen Verbesserung, zur Weiterentwicklung genutzt werden. Weiterhin bietet es die Chance, die Loyalität des Kunden durch eine überzeugende Abwicklung der Beschwerde und Beseitigung seiner Kritikpunkte wieder herzustellen.

3 Data Mining und Knowledge Discovery

3.1 Überblick

Der Begriff Data Mining beschreibt eine (semi-)automatisierte Auswertung großer Datenbestände mit dem Ziel, geschäftsrelevante Zusammenhänge in den Daten zu entdecken.[4] Dabei kommen anspruchsvolle Verfahren aus unterschiedlichen Wissenschaftsdisziplinen wie z. B. Statistik, Datenbanken, Künstliche Intelligenz und maschinelles Lernen zum Einsatz.[5] Allerdings wird schnell deutlich, dass nützliche Erkenntnisse kaum ohne das domänenspezifische

---

1 Viele Techniken der Kundenbewertung arbeiten allerdings noch rein vergangenheitsorientiert, wie z. B. ABC-Analysen und die meisten Scoring Modelle.
2 RAAB+ (2002), S. 14 f.; KOTLER+ (1995), S. 74 ff.
3 BERSON+ (1999), S. 277 ff.; RUD (2001), S. 10 f. und 257 ff.
4 HAND+ (2001) S. 1; BERRY+ (1997), S. 5.
5 VOSS+ (2001), S. 349; HIPPNER+ (2002b), S. 216.

Fachwissen des Anwenders gewonnen werden können. An die Stelle einer vollständig automatisierten Auswertung tritt damit ein interaktiver und iterativer Analyseprozess, in dessen Rahmen der Analytiker die Aufgabendefinition und Datenaufbereitung übernimmt, Data Mining Algorithmen auswählt und anwendet und deren Ergebnisse evaluiert.[1] Dieser Prozess wird als „Knowledge Discovery in Databases (KDD)“ bezeichnet und integriert Data Mining als einen Prozessschritt.[2] Zum Teil werden die Begriffe Data Mining und KDD auch synonym verwendet oder es wird vom Data Mining Prozess gesprochen.[3]

3.2 Prozess des Knowledge Discovery in Databases

Grundsätzlich beinhaltet KDD die folgenden Phasen:[4]

1. Aufgabendefinition – *Formulierung der betriebswirtschaftlichen Zielsetzung und Ableitung von analytischen Zielen für das Data Mining*
   Aufbauend auf den Zielvorgaben kann eine Vorselektion der anwendbaren Methoden[5] erfolgen und ein Projektplan erstellt werden.

2. Datenselektion – *Katalogisierung und Bewertung verfügbarer Datenquellen*
   Es ist zu entscheiden, ob eine Anreicherung interner Datenbestände durch extern von Partnern oder Marketingdienstleistern erhältliche Daten sinnvoll ist. Alle in die Untersuchung einzubeziehenden Merkmale sind in dieser Phase auszuwählen.

3. Datenvorverarbeitung – *Überführung der Ausgangsdaten in ein analysefähiges Format*
   Dieser Schritt beinhaltet die Behandlung fehlender und fehlerhafter Werte, welche z. B. mittels explorativer Datenanalyse[6] identifiziert werden können. Zur Erhöhung der rechentechnischen Effizienz eines Algorithmus kann eine Zufallsstichprobe aus den Ausgangsdaten gezogen werden.

---

1 HIPPNER+ (2002b), S. 216.
2 FAYYAD+ (1996), S. 40.
3 Vgl. z. B. CABENA+ (1997), S. 12; KÜSTERS (2001), S. 97; WITTEN+ (2001), S. 3.
4 KÜSTERS (2001), S. 97 ff.; HIPPNER+ (2001), S. 22 ff.; FAYYAD+ (1996), S. 42 f. Vgl. auch Abbildung 3.
5 Unter einer Methode wird hier eine generelle Beschreibung einer Vorgehensweise verstanden (HIPPNER+ (2002b), S. 217). Die Begriffe Methode und Verfahren werden im Folgenden synonym verwendet.
6 HAND+ (2001), S.53 ff.

4. Datentransformation – *Re-Kodierung der Merkmale und eventuell Reduktion der Variablenanzahl*

   In Abhängigkeit der anzuwendenden Data Mining Methoden werden bestimmte Anforderungen an das Skalenniveau[1] der Merkmale gestellt. Einige Verfahren verarbeiten z. B. nur kategoriale Variablen, andere nur metrische, so dass gegebenenfalls entsprechende Skalentransformationen vorzunehmen sind. Wird mit sehr vielen Einflussfaktoren gearbeitet, können Verfahren wie die Faktoren- oder Hauptkomponentenanalyse eingesetzt werden, um eine Teilmenge besonders relevanter Merkmale zu extrahieren.[2]

5. Data Mining – *Auswahl und Anwendung des Data Mining Algorithmus*

   In Abhängigkeit der betriebswirtschaftlichen Fragestellung werden aus einer geeigneten Verfahrensklasse[3] ein oder mehrere konkrete Algorithmen ausgewählt und auf die transformierten Daten angewendet. Da eine geeignete Parametrisierung der Methode nicht a priori bekannt ist, werden zumeist verschiedene Konfigurationen evaluiert und die entstandenen Modelle[4] miteinander verglichen. Alternativ kann die Wahl der Parametrisierung auch automatisiert werden, indem z. B. ein evolutionärer Algorithmus eingesetzt wird, um systematisch eine Vielzahl verschiedener Parameterkonstellationen zu testen und eine geeignete Konfiguration zu ermitteln.[5]

6. Evaluation – *Aufbereitung, Interpretation und Bewertung der Ergebnisse des KDD-Prozess unter betriebswirtschaftlichen Gesichtspunkten*

   Es ist zu entscheiden, in wie weit die identifizierten Muster geschäftsrelevante Informationen darstellen und welche Konsequenzen aus den Erkenntnissen gezogen werden.

---

1 BACKHAUS+ (2000), S. XVIII; HAND+ (2001), S. 25.
2 FREITAS (2002), S. 64 ff.; BACKHAUS+ (2000), S. 252 ff.; HAND+ (2001), S. 74 ff.
3 Vgl. auch Abschnitt 3.3.
4 In diesem Zusammenhang beschreibt ein Modell das Ergebnis der Anwendung einer Methode auf einen konkreten Datenbestand. HIPPNER+ (2002b), S. 217.
5 BÄCK+ (2001); FREITAS (2002); STAHLBOCK (2002).

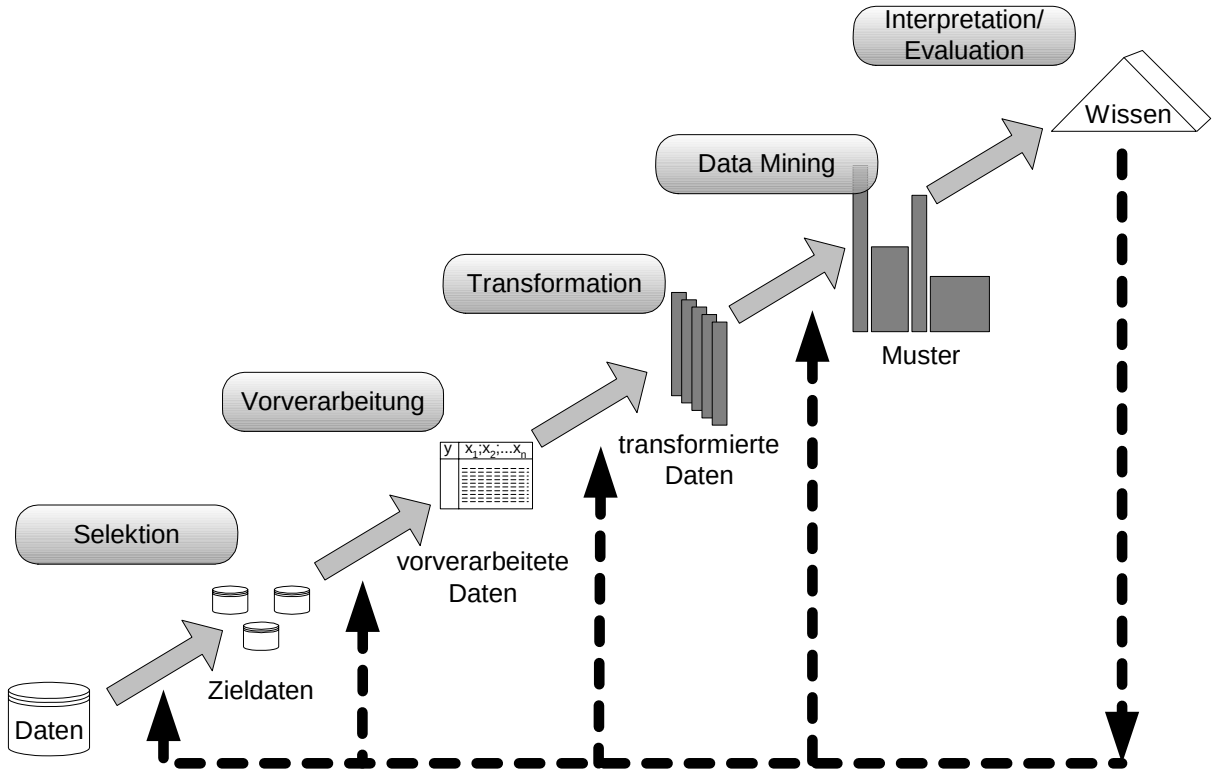


Abbildung 3: Der Prozess des Knowledge Discovery in Databases
(in Anlehnung an FAYYAD+ (1996), S. 41)

Zur Unterstützung des KDD-Prozess stehen eine Reihe von Softwaresystemen zur Verfügung. Dabei kann zwischen Werkzeugen unterschieden werden, die nahezu alle Prozessphasen unterstützen und solchen, die lediglich einzelne Teilschritte abbilden. Zu den letzteren gehören insbesondere Programme, die einen speziellen Algorithmus implementieren. Integrierte Data Mining Pakete mit umfangreicher Unterstützung für sämtliche Phasen sind z. B. der SAS Enterprise Miner und Clementine von SPSS.

3.3 Data Mining Modelle und Methoden

Welche Verfahren zu den Data Mining Methoden gerechnet werden, wird in der Literatur kontrovers diskutiert. So zählen z. B. einige Autoren Methoden der multivariaten Statistik nicht zu den Data Mining Verfahren, da diese weder autonom noch auf großen Datenbeständen eingesetzt werden können.[1] Andere zählen diese ebenso dazu, wie klassische zeitreihenanalytische Verfahren, Methoden der explorativen Datenanalyse und Visualisierungstechniken.[2] Im Folgenden soll daher eine Auswahl an Methoden und Problemklassen vorgestellt

1 RAPP+ (1999), S. 250.
2 VOSS+ (2001), S. 351; KÜSTERS (2001).

werden, deren Zugehörigkeit zum Data Mining Kontext praktisch unumstritten ist, die aber keinen Anspruch auf Vollständigkeit erhebt.

Grundsätzlich kann zwischen Beschreibungsproblemen und Prognoseproblemen unterschieden werden.[1] Gegenstand von Beschreibungsproblemen ist die Aufdeckung interpretierbarer und handlungsrelevanter Muster in den Daten (z. B. häufig gemeinsam gekaufte Produkte). Für Prognosen ist hingegen die Vorhersage einer unbekannten oder zukünftigen Größe, der so genannten Zielvariable, charakteristisch. Dabei soll die Zielvariable, z. B. die Bonität eines Neukunden, aus bekannten Attributen eines Datensatzes, z. B. demographischen Merkmalen, abgeleitet werden.[2] Eine klare Trennung zwischen Prognose- und Beschreibungsmodellen ist allerdings nicht immer möglich, da z. B. Ergebnisse von Prognosen durchaus auch erklärende Komponenten beinhalten können und die in Beschreibungsmodellen identifizierten Wirkungszusammenhänge sich zum Teil für Vorhersagen verwenden lassen.[3]

Auf niedrigerem Abstraktionsniveau lassen sich die nachfolgenden Fragestellungen identifizieren, die von Data Mining adressiert werden:

- Regression – *Untersuchung von Beziehungen zwischen einer abhängigen und einer oder mehreren unabhängigen Variablen*[4]

  Ziel ist die Schätzung der Parameter einer Funktion, welche eine Abbildung der abhängigen Variablen auf die Prognosevariable vornimmt. Je nach der Form dieser Funktion und dem Skalenniveau der Variablen können verschiedene Formen der Regressionsanalyse unterschieden werden (z. B. lineare versus nicht-lineare Regression).[5] Ebenfalls für diese Fragestellung geeignet sind künstliche neuronale Netze, wie sie z. B. in HERTZ$^+$ (1991), PATTERSON (1996) oder ZELL (2000) näher beschrieben werden.

  Ein typisches Anwendungsfeld für Regressionsanalysen ist z. B. die Prognose des Umsatzes eines Kunden in der nächsten Saison anhand vergangener Bestellungen.

---

1 FAYYAD$^+$ (1996), S. 44.
2 HIPPNER$^+$ (2002b), S. 218. Dabei wird unterstellt, dass die verwendeten Merkmale einen signifikanten Einfluss auf die Zielgröße ausüben.
3 FAYYAD$^+$ (1996), S. 44.
4 BACKHAUS$^+$ (2000), S. 2.
5 KÜSTERS (2001), S. 106.

- Segmentierung – *Untergliederung der Datensätze in einzelne Gruppen, die durch gemeinsame Merkmalsausprägungen beschrieben sind*[1]

  Während die Elemente einer Gruppe hinsichtlich ihrer Merkmalsausprägungen möglichst ähnlich sein sollen, soll zwischen den einzelnen Gruppen eine möglichst große Heterogenität herrschen.[2] Zum Einsatz kommen vor allem so genannte Clusterverfahren[3], die methodisch aus zwei Schritten bestehen:[4]

  1. Es wird die Ähnlichkeit zwischen allen Datenobjekten auf der Basis der jeweiligen Merkmalsausprägungen mittels eines festzulegenden Distanzmaßes (Proximitätsmaßes) berechnet.

  2. Aufgrund der so ermittelten Ähnlichkeiten werden die Objekte durch einen Fusionsalgorithmus in Gruppen zusammengefasst. Einige Algorithmen versuchen dabei nicht nur die Homogenität innerhalb der Gruppen zu maximieren, sondern auch die Heterogenität zwischen den Gruppen. Neben klassischen Clusterverfahren können auch bestimmte Varianten künstlicher neuronaler Netze, so genannte Selbstorganisierende Karten nach KOHONEN, eingesetzt werden.[5]

  Segmentierungsverfahren werden z. B. zur Gruppierung von Kunden eingesetzt, um Marketingaktionen zielgruppenspezifisch zu gestalten.

- Assoziation – *Beschreibung von Abhängigkeiten zwischen Merkmalen von Datensätzen*

  Der Zweck einer Assoziationsanalyse besteht darin, bestimmte Datenelemente zu identifizieren, die das Auftreten anderer Elemente implizieren.[6] Wird eine solche Beziehung aufgedeckt, kann sie als Regel „Wenn Element A auftritt, dann tritt (mit einer gewissen Wahrscheinlichkeit) auch Element B auf“ formuliert und für Prognoseaufgaben verwendet werden.[7] Mathematisch basiert die Suche nach solchen Assoziationen auf der Häufigkeitsbetrachtung von Attributkombinationen.[8] In diesen Bereich gehören auch so genannte Sequenzanalysen, welche ebenfalls Abhängigkeiten zwischen Elementen beschreiben. Im

---

1 HIPPNER$^+$ (2002b), S. 219.
2 RAPP$^+$ (1999), S. 249.
3 Details zu clusterbildenden Methoden können z.B. GRABMEIER (2001) entnommen werden.
4 KÜSTERS (2001), S. 112.
5 KOHONEN (2001).
6 CABENA$^+$ (1997), S. 80.
7 HETTICH$^+$ (2001), S. 427.
8 RAPP$^+$ (1999), S. 249.

Gegensatz zu den zeitpunktbezogenen Assoziationsmethoden berücksichtigen diese jedoch die zeitliche Reihenfolge des Auftretens verschiedener Zustände.[1]

Ein klassisches Beispiel für Assoziationsprobleme sind Warenkorbanalysen, die untersuchen, welche Produkte häufig gemeinsam erworben werden. Diese Erkenntnisse können dann zur Planung der Warenplatzierung genutzt werden. Sequenzanalysen können eingesetzt werden, um die Navigationspfade von Websitebesuchern zu untersuchen. Die nacheinander betrachteten Seiten bilden dabei das Sequenzmuster.

- Klassifikation – *Zuordnung von Datensätzen zu a priori definierten und durch bestimmte Merkmale beschriebenen Klassen*

  Viele der unter Abschnitt 2.3 vorgestellten Analysen gehören in diese Kategorie. Typisch ist dies z. B. für Responseanalysen, bei denen die Klassen „Reagierer“ und „Nicht-Reagierer“ unterschieden werden. Sinngemäß gilt dies auch für alle anderen Vorhersagemodelle, deren Prognosevariable nur diskrete Zustände annehmen kann.[2] Dies schließt Cross und Up Selling Probleme sowie Churnanalysen mit ein, die ebenfalls als Klassifikationsproblem modelliert werden können.[3] Aufgrund dieser zentralen Bedeutung von Klassifikationsanalysen für Problemstellungen des aCRM sollen ihre charakteristischen Merkmale detaillierter im nächsten Abschnitt 3.4 beschrieben werden. Neben den unter Abschnitt 4 vorgestellten Support Vektor Maschinen können z. B. künstliche neuronale Netze, Entscheidungsbaumverfahren, logistische Regressionen oder Diskriminanzanalysen zur Lösung klassifikatorischer Fragestellungen eingesetzt werden.[4]

3.4 Klassifikation

Mit dem Begriff Klassifikation werden der Prozess und das Ergebnis einer Einteilung von Objekten in Klassen bezeichnet. Objekte einer Klasse sollen sich ähnlich sein, Objekte verschiedener Klassen sollen sich möglichst weitgehend in ihren Merkmalsausprägungen unterscheiden. Ein Objekt kann als Muster aufgefasst werden, welches die Objektmerkmale, also seine messbaren physikalischen Eigenschaften (zusammengefasst im Vektor $\vec{x}$, dessen Komponenten x_i die einzelnen beobachteten Merkmalsausprägungen durch Messwerte ausdrü-

---

1 HETTICH$^+$ (2001), S. 441.
2 Beispielsweise Antragsscoring und Bonitätsprüfung. Als Zustände kommen „kreditwürdig“ und „nicht kreditwürdig“ in Betracht.
3 RUD (2001), S. 259 f.; TIETZ$^+$ (2001), S. 767 ff.
4 BISHOP (1995); WITTEN$^+$ (2001), S. 95 ff.; HAND$^+$ (2001), S. 145 ff.; BACKHAUS$^+$ (2000), S. 104 ff.

cken[1], beispielsweise Alter und Einkommen eines Objekts „Kunde“) und die zugehörige, durch das Objekt repräsentierte Klasse k (z. B. „Reagierer“, „Nicht-Reagierer“), in einem Paar ($\vec{x}$, k) zusammenfasst. Die Dimension N von $\vec{x}$ entspricht der Anzahl der gemessenen Objekteigenschaften. Bezeichnet man den diskreten, beschränkten Konzeptraum, der alle problemrelevanten Klassen enthält, mit **K**, seine Größe mit K, den Musterraum mit **X** und seine Größe mit X, lässt sich ein Klassifikationsverfahren formal zusammenfassen:

Die Menge **X** aller X durch einen Datengenerator unabhängig und gleichverteilt generierten Objekte ist gegeben mit:

$$\mathbf{X} = \{ \vec{x}_1, ..., \vec{x}_X \}.$$

Ein Objekt $\vec{x}$ wird durch N Messwerte x beschrieben, die im Merkmalsvektor $\vec{x}$ zusammengefasst sind:

$$\vec{x} = \{x_1, ..., x_N\}.$$

Jedes Objekt $\vec{x}$ gehört einer der K Klassen $\mathrm{k} \in \mathbf{K}$ an. Die Menge **K** der Klassen ist also gegeben mit:

$$\mathbf{K} = \{\mathrm{k}_1, ..., \mathrm{k}_K\}.$$

Ein Klassifikator ordnet Eingabedaten eine Klasse durch Bildung einer Diskriminanzfunktion zu, so dass folgende Entscheidungsregel entsteht:

$$e(\vec{x}) : \mathbf{X} \rightarrow \mathbf{K}.$$

Ein Klassifikationsproblem besteht darin, ein neues Objekt mit unbekannter Klasse aufgrund seiner beobachtbaren Eigenschaften einer Klasse zuzuordnen. Das Problem wird durch Zufälligkeiten (sog. Rauschen) in den Daten erschwert. Mittels einer Lernmaschine kann versucht werden, dieses Problem zu lösen.[2] Ziel eines überwachten Lernens[3] ist das Ableiten eines allgemeinen, generalisierten (funktionalen) Zusammenhangs zwischen Objekteigenschaften und Klasse aus Beispieldaten[4], die in Form von Eingabe- (Ist) und Ausgabewerten (Soll) vorliegen. Durch diese Kenntnis über klassenspezifische Verteilungen soll ein möglichst geringer durchschnittlicher Fehler für neue unabhängige Anwendungsdaten – zufällig aus gleicher

---

[1] Die Messwerte können von unterschiedlichem Zahlentyp sein. Typisch sind Binärwerte aus den Mengen {0, 1} bzw. {-1, +1} und ganzzahlige oder reelle Skalare.

[2] Dabei wird davon ausgegangen, dass der Zusammenhang zwischen Objektmerkmalen und Klasse nicht explizit modelliert werden kann – entweder, weil er nicht genau bekannt ist oder weil die Zusammenhänge zu komplex sind –, denn ansonsten könnte das Problem „direkt“, d.h. ohne Lernvorgang, gelöst werden.

[3] Daneben gibt es unüberwachtes Lernen und verstärkendes Lernen. Das überwachte Lernen ist mit der Diskriminanzanalyse der multivariaten Statistik vergleichbar, unüberwachtes Lernen mit der Clusteranalyse. BACKHAUS$^+$ (2000), S. 328 ff.; SCHÜRMANN (1996), S. 7 f.

[4] Auch: „Trainingsdaten“, „Lerndaten“.

Verteilung (also gleichem Anwendungsgebiet) ausgewählt – erzielt werden. Eine Lernmaschine ist definiert durch eine Menge möglicher Abbildungen $\vec{x} \to f(\vec{x}, \alpha)$, wobei α Element des Parameterraums A ist. Durch die Wahl eines bestimmten α aus A wird die Lernmaschine festgelegt.[1] Von zentraler Bedeutung für den Lernprozess ist die Festlegung des Hypothesenraums, der Menge aller Funktionen, aus welcher der Lernalgorithmus durch Festlegung von α-Werten in Abhängigkeit von den Lerndaten eine Funktion auswählt. Im Folgenden wird zunächst von einer binären Klassifikation mit **K** = {+1, -1} ausgegangen.

4 Support Vektor Maschinen zur Lösung von Klassifikationsproblemen

4.1 Überblick

Die Support Vektor Maschine (SVM) ist ein überwacht lernendes Verfahren zur Klassifikation und – mit Erweiterungen – Punktprognose (Regression). Sie wurde in ihrer ursprünglichen Form an den AT&T Bell Laboratories Anfang bis Mitte der 1990er Jahre von VAPNIK und seinen Mitarbeitern entwickelt. Sie basiert auf der statistischen Lerntheorie, die in den letzten drei Jahrzehnten von VAPNIK, CHERVONENKIS und anderen entwickelt wurde.[2] Grundlage ist der verallgemeinerte, um nichtlineare Zusammenhänge erweiterte „Generalised Portrait"-Algorithmus[3]. Das SVM-Verfahren ist somit theoretisch gut fundiert. Mittlerweile sind eine Vielzahl von Veröffentlichungen zu SVM und ihren Erweiterungen erschienen. Es gibt allerdings noch vergleichsweise wenig dokumentierte Anwendungen, insbesondere aus dem betriebswirtschaftlichen Bereich.[4] Die aus der Theorie abgeleiteten hohen Erwartungen an die Ergebnisqualität und behaupteten Leistungen müssen in diesem Bereich noch bestätigt werden.

Mit Hilfe des SVM-Lernverfahrens können lineare und nichtlineare Probleme gelöst werden. Das Verfahren kann effektiv insbesondere bei zahlreichen Objektattributen eingesetzt werden, bei denen der Einsatz anderer Lernverfahren, z. B. künstlicher neuronaler Netze, eher proble-

---

1 Ein künstliches neuronales Netz mit festgelegter Architektur ist z. B. eine Lernmaschine, bei der α die Gewichte im Netzwerk beschreibt. (BURGES (1998), S. 3).

2 VAPNIK (1982); VAPNIK (1995); VAPNIK (1998). Eine gute Einführung gibt z. B. BURGES (1998).

3 1960er Jahre, Russland.

4 Aktuelle Informationen sind im Internet z. B. über `http://www.kernel-machines.org` zu beziehen.

matisch ist. Ein weiterer Nachteil neuronaler Netze, die Gefahr des Überlernens[1] von gegebenen Daten, ist bei SVM nicht in dem Maße gegeben. Überanpassung hemmt die gewünschte Generalisierungsleistung, d. h. das Erreichen guter Ergebnisse auf neuen, nicht gelernten Daten. Bei einer SVM kann die Kapazität der Lernalgorithmen und damit das Potential einer Überanpassung, kontrolliert werden. Ferner sind SVM-Ergebnisse wegen des zugrunde liegenden exakten mathematischen Kalküls leicht reproduzierbar. Sie unterliegen keinen Zufallsschwankungen innerhalb des Verfahrens, da das Lernen einer SVM durch die Minimierung eines konvexen, mathematischen Programms erfolgt und es folglich keine lokalen Optima gibt.[2]

4.2 Risikominimierung und Überanpassung

Erwartetes Risiko

Ein Lernalgorithmus zielt darauf ab, den erwarteten Fehler eines Klassifikators, das Risiko

$$\mathrm{R}(\alpha) = \int \frac{1}{2} \left| \mathrm{k} - \mathrm{f}(\vec{x}, \alpha) \right| \mathrm{dP}(\vec{x}, \mathrm{k}) \tag{1}$$

zu minimieren. Dieses Risiko lässt sich allerdings aufgrund der Unbekanntheit der Verteilung $\mathrm{P}(\vec{x}, \mathrm{k})$ nicht direkt bestimmen.

Empirische Risikominimierung

Der Fehler für die *L* Lerndaten, in denen sämtliche vorhandene Informationen enthalten sind, wird durch das empirische Risiko

$$\mathrm{R}_{\mathrm{emp}}(\alpha) = \frac{1}{L} \sum_{i=1}^{L} \left| k_i - f\left(\vec{x}_i, \alpha\right) \right| \tag{2}$$

ausgedrückt, welches unabhängig von der Verteilung $\mathrm{P}(\vec{x}, \mathrm{k})$ ist. Für ein festes α konvergiert nach dem Gesetz der großen Zahl, also bei ausreichend umfangreicher Lerndatenmenge ($\mathrm{L} \to \infty$), das empirische Risiko gegen das erwartete Risiko. Man hofft entsprechend, dass durch eine Funktion, die das empirische Risiko minimiert, auch das erwartete Risiko minimiert wird.

Konsistenz und Überanpassung

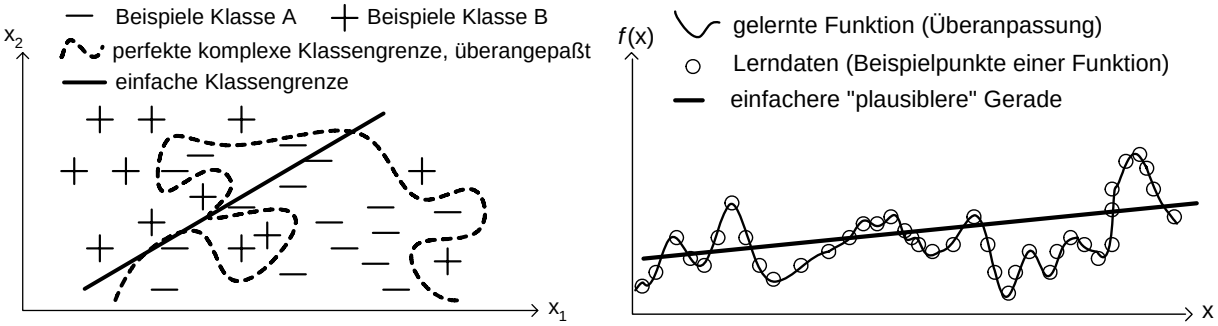
Enthält der Hypothesenraum mindestens eine Funktion, die keinen Lernfehler produziert, also die Lerndaten korrekt klassifiziert, bezeichnet man den Hypothesenraum als konsistent. Es

---

1 Auch: „Auswendiglernen“, „Überanpassung“, „Overlearning“, „Overfitting“.

2 BENNETT+ (2000), S. 1.

kann schwierig sein, eine konsistente Menge zu finden, sei es, weil Daten verrauscht sind, sei es, weil die Darstellung entsprechender Funktionen zu schwierig ist. Aber selbst eine Minimierung des empirischen Risikos hat nicht zwingend ein Minimum des Fehlers für ungelernte Daten, also für den Anwendungsfall, zur Folge. Diese Fähigkeit, gute Ergebnisse auf Lerndaten auch auf unbekannte Testdaten übertragen zu können, wird mit Generalisierungsfähigkeit bezeichnet. Generalisierungsfähigkeit wird durch Überanpassung an Lerndaten gehemmt.[1] Abbildung 4 (a) zeigt Überanpassung einer perfekten Klassifikationsgrenze an gegebene Datenpunkte mit empirischem Risiko Null im Vergleich zu einer einfachen Klassengrenze, die zwar höheres empirisches Risiko hat, aber dafür einfacher (und „plausibler“) ist. Abbildung 4 (b) zeigt entsprechend eine Überanpassung an einen Funktionsverlauf durch exaktes Lernen von Funktionspunkten im Vergleich zu einer einfachen Gerade.



(a) Überanpassung an Klassengrenze (b) Überanpassung an Funktionsverlauf

Abbildung 4: Überanpassung eines Lernalgorithmus an Beispieldaten
(in Anlehnung an STAHLBOCK (2002) S. 67)

Es besteht eine Wechselwirkung zwischen Lern- und Generalisierungsergebnis, denn je reicher die Funktionsklasse, die dem Lernalgorithmus zur Verfügung steht, an flexiblen (komplexen) Funktionen ist, desto eher können konsistente Funktionen gefunden werden. Mit einer ausdrucksstarken Funktionsmenge kann zwar das empirische Risiko stark verringert werden, aber es wird mit steigender Komplexität eine deutlich größere Anzahl an Beispieldaten benötigt, um möglichst kleine Abweichungen zwischen dem Lernfehler und dem Generalisierungsfehler zu garantieren. Das zentrale Problem der statistischen Lerntheorie ist die Beantwortung der Frage, wann ein niedriger Lernfehler zu einem niedrigen „echten“ Fehler führt. Die Wahrscheinlichkeit der Überanpassung steigt mit der Komplexität der Funktionen. Es besteht also eine entgegen gerichtete Wirkung der Genauigkeit, mit der Lerndaten korrekt klassifiziert werden und einer möglichst geringen Komplexität der Lernmaschine. Ein möglicher Weg aus diesem Dilemma ist, nach dem Ockhams-Razor-Prinzip die Funktionsklasse a priori zu be-

[1] BISHOP (1995); VAPNIK (1995), S. 14; SCHÖLKOPF (1997), S. 22.

schränken („so einfach wie möglich, so genau und komplex wie nötig“), bzw. bei sonst gleichen Bedingungen einfache Modelle den komplexen Modellen vorzuziehen. SVM verwirklichen dazu das Prinzip der so genannten strukturellen Risikominimierung, in dem für den erwarteten Fehler effektive obere Schranken, bestehend aus dem empirischen Fehler und einem Konfidenzintervall, konstruiert bzw. minimiert werden.[1]

Obere Schranke für das erwartete Risiko

VAPNIK leitet aufgrund des empirischen Risikos die Gültigkeit einer probabilistischen oberen Schranke für das erwartete Risiko, die mit einer Wahrscheinlichkeit $1-\eta \quad (0 \le \eta \le 1)$ eingehalten wird, wie folgt ab:

$$\mathrm{R}(\alpha) \le \mathrm{R}_{\mathrm{emp}}(\alpha) + \sqrt{\left(\frac{h(\log(2L/h) - \log(\eta/4)}{L}\right)}, \tag{3}$$

wobei $h \ge 0$ die so genannte VAPNIK-CHERVONENKIS-Dimension (VC-Dimension[2]), ein Maß für die Komplexität der zugrunde liegenden Funktionsklasse, also für die Komplexität des Hypothesenraums, ist.[3] Die Schranke ist unabhängig von einer bestimmten Verteilungsfunktion $\mathrm{P}(\vec{x},\mathrm{k})$. Sie ist abhängig von der VC-Dimension und der Anzahl an Lerndaten L. Sie sinkt monoton mit sinkender VC-Dimension (also mit zunehmender Beschränkung des Hypothesenraums) und mit guter Trennbarkeit der Lerndaten (d.h. geringem empirischen Risiko). Mit großem L sinkt die Schranke ebenfalls, daher werden bei großen Lerndatenmengen gute Ergebnisse erzielt.[4] [5]

Strukturelle Risikominimierung

Grundidee der strukturellen Risikominimierung[6] (SRM) ist, für einen Lernalgorithmus die VC-Dimension der eingesetzten Funktionsklasse als Parameter zu benutzen. Dazu wird der Hypothesenraum $\mathrm{S} = \{\mathrm{f}(\vec{x},\alpha);\ \alpha \in \mathrm{A}\}$ in geschachtelte Teilmengen $\mathrm{S}_\mathrm{n} = \{\mathrm{f}(\vec{x},\alpha);\ \alpha \in \mathrm{A}_n\}$ geteilt, so dass $\mathrm{S}_1 \subset \mathrm{S}_2 \subset ... \subset \mathrm{S}_\mathrm{n}...$ mit $\mathrm{h}_1 \le \mathrm{h}_2 \le ... \le \mathrm{h}_\mathrm{n}...$ gilt. Für gegebene Lerndaten wird die Teilmenge S_k als Hypothesenraum ausgewählt, die die Schranke (rechte Seite von (3)) für

---

1 Die strukturelle Risikominimierung geht zurück auf VAPNIK (1995).

2 VAPNIK (1995), S. 76–78.

3 Der zweite Summand auf der rechten Seite wird VC-Konfidenz genannt. (BURGES (1998), S. 3).

4 BURGES (1998), S. 2–6.

5 Allerdings kann die für ein geringes empirisches Risiko benötigte VC-Dimension so groß sein, dass die VC-Konfidenz zu groß wird. Mit Hilfe der Schranke kann aber die methodologische Aussage gemacht werden, dass mit hoher Wahrscheinlichkeit richtig gelernt wurde, wenn mit niedriger VC-Dimension die Daten erklärt werden können. Ferner ist zu beachten, dass die Schranke oft pessimistisch ist, da sie für sämtliche möglichen Verteilungsfunktionen gilt. (SCHÖLKOPF$^+$ (1999b), S. 156).

6 VAPNIK (1982), S. 232–236.

das Risiko (1) minimiert. Die Kontrolle über das Risiko erfolgt also durch Angabe einer möglichst kleinen oberen Schranke, dadurch, dass in verschiedenen beschränkten Funktionsmengen S_k eine empirische Risikominimierung durchgeführt und schließlich die Funktion f^* aus der Menge S^* gewählt wird, die die obere Schranke als Summe aus Lernfehler und VC-Konfidenz gemäß Formel (3) minimiert.[1] Es wird die Funktionsklasse mit geringstmöglicher Kapazität gewählt. Abbildung 5 zeigt das Prinzip der strukturellen Risikominimierung durch Kapazitätskontrolle. Durch Support Vektor Maschinen wird dieses Prinzip umgesetzt.

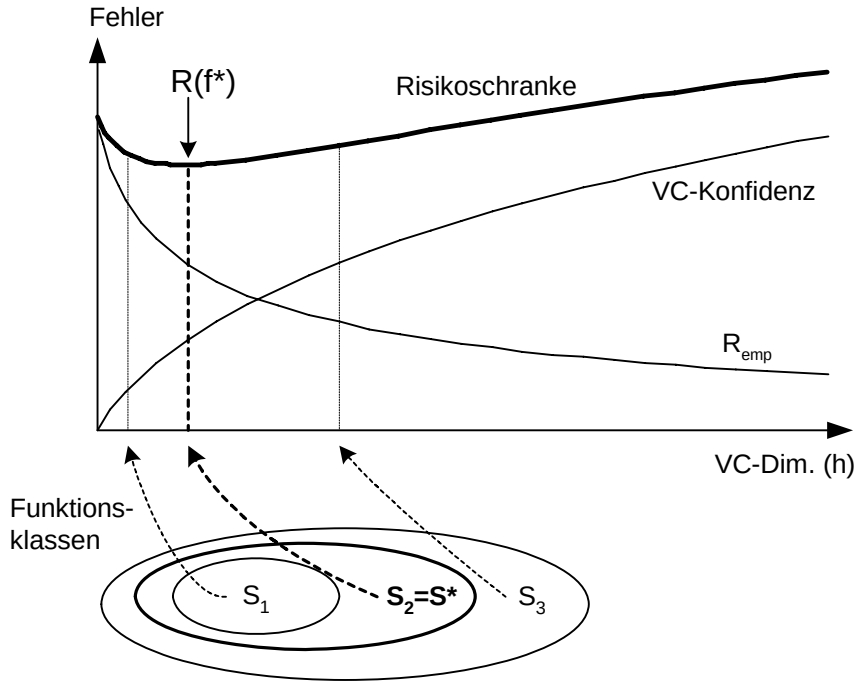


Abbildung 5: Prinzip der strukturellen Risikominimierung
(in Anlehnung an VAPNIK (1992), S. 92; SCHÖLKOPF (1997), S. 24)

4.3 Perfekte lineare Klassifikation

Gegeben sind zweidimensionale Vektoren, je nach zugehöriger Klasse als Quadrat oder Kreis dargestellt. Ziel ist, in dem Punkteraum eine trennende Linie (im allgemeinen Fall bei höherer Dimensionalität: eine Hyperebene) zu finden, die alle Datenpunkte korrekt separiert und somit klassifiziert. Zur Konstruktion werden zwei parallele Hilfslinien bzw. Hilfshyperebenen eingefügt, deren maximaler Abstand eine maximale Trenngüte durch die in der Mitte zwischen ihnen parallel liegende Klassengrenze zur Folge hat. Die Hilfsebenen werden gedreht und so weit auseinander geschoben, bis sie jeweils erstmals Datenpunkte berühren. Diejenigen Vektoren, die auf diesen Hilfsebenen liegen und somit die Trennebene und maximale

[1] SCHÖLKOPF (1997), S. 23 f.

Trenngüte festlegen, werden als Support-Vektoren bezeichnet.[1] Die Lösung des Klassifikationsproblems (die Lage der Trennebene) hängt nur von den Support-Vektoren ab, alle anderen Lerndaten sind für die Lage der Klassengrenze irrelevant. Abbildung 6 zeigt eine Konstellation mit zwei Hilfshyperebenen mit maximalem Abstand sowie einer alternativen schlechteren Lösung.

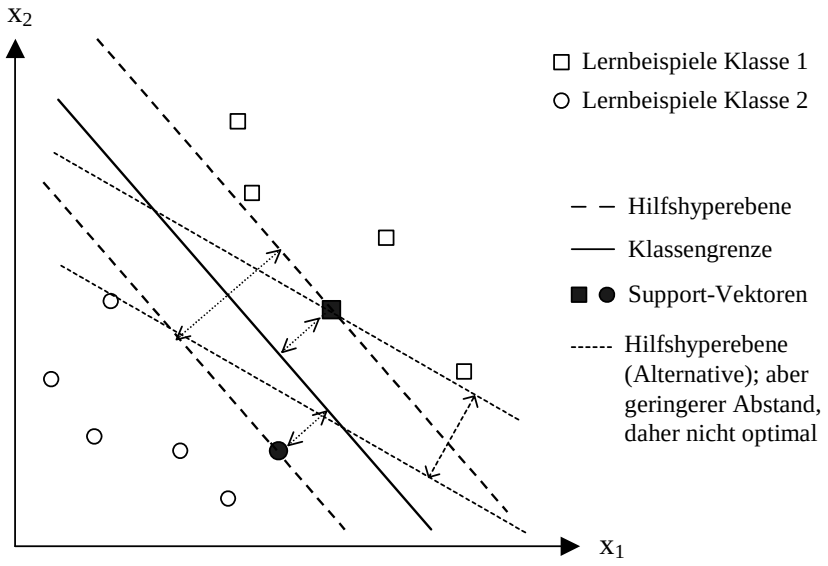


Abbildung 6: Hilfshyperebenen und Klassengrenze im linear perfekt trennbaren Zweiklassenfall
(in Anlehnung an BENNETT$^+$ (2000), S. 3)

4.4 Ableitung des Optimierungsproblems

Im linear perfekt trennbaren Zweiklassenfall kann die optimal trennende Hyperebene durch zwei parallele Hilfshyperebenen gefunden werden, zwischen denen kein Datenpunkt liegen darf und deren Abstand maximal ist. Erfolgt eine Skalierung, so dass die zur trennenden Hyperebene nächstliegenden Vektoren, die auf den Hilfshyperebenen liegenden Support-Vektoren, aus beiden Klassen einen Abstand von jeweils $1/\|\vec{w}\|$ zur Klassengrenze haben, erhält man die linearen Bedingungen

$$\vec{w} \cdot \vec{x}_i + b \geq 1 \qquad \text{für } k_i = +1,\ i=1, \ldots L \tag{4}$$

$$\text{und} \quad \vec{w} \cdot \vec{x}_i + b \leq -1 \qquad \text{für } k_i = -1,\ i=1, \ldots L, \tag{5}$$

welche von den Lerndaten erfüllt werden müssen. Die Punkte, die (4) und (5) mit Gleichheit erfüllen, liegen auf den oben genannten Hyperebenen. Zusammenfassen kann man (4) und (5) zu der Bedingung

---

[1] BENNETT$^+$ (2000), S. 1 f.

$$k_i(\vec{w}\cdot\vec{x}_i + b) \geq 1 \qquad \text{für } k_i \in \{-1,+1\},\ i=1, \ldots L, \tag{6}$$

Zusammenfassen, die durch die Support-Vektoren mit Gleichheit, d. h. $k_i(\vec{w}\cdot\vec{x}_i + b) = 1$ erfüllt wird. Eine optimal trennende Hyperebene, gebildet aus allen Punkten, die $\vec{w}\cdot\vec{x}_i + b = 0$ erfüllen, ist eindeutig bestimmt. Das Auffinden von ihr ist gleichbedeutend mit der Bestimmung eines Vektors $\vec{w}$ und einer Konstanten *b*, so dass Bedingung (6) erfüllt ist und $\vec{w}$ minimale Euklidische Norm $\|\vec{w}\| = \sqrt{\vec{w}\cdot\vec{w}}$ hat. Die Hilfshyperebenen haben senkrechte Abstände zum Ursprung in Höhe von $(1-b)/\|\vec{w}\|$ bzw. $(-1-b)/\|\vec{w}\|$, die trennende Ebene dazwischen hat senkrechten Ursprungsabstand von $|b|/\|\vec{w}\|$. Der Normalenvektor $\vec{w}$ bestimmt also den Abstand vom Ursprung und die Richtung der trennenden Hyperebene. Zwischen den durch die Support-Vektoren festgelegten Hilfshyperebenen bildet sich ein Trennungsgürtel der Breite $2/\|\vec{w}\|$. Die folgende Abbildung zeigt den Zusammenhang.[1]

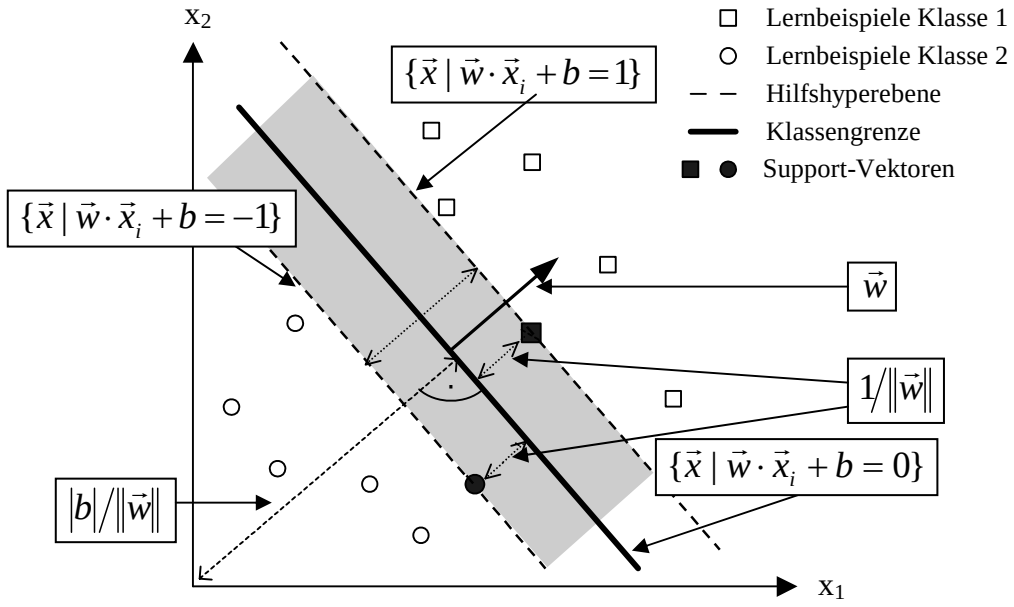


Abbildung 7: Klassengrenze, Hilfshyperebenen und Trennungsgürtel im perfekt trennbaren Fall (in Anlehnung an SCHÖLKOPF (1997), S. 35; BURGES (1998), S. 9)

Eine Maximierung des Abstandes zwischen den beiden Hilfsebenen entspricht einer Minimierung von $\frac{1}{2}\|\vec{w}\|^2$, so dass sich zur Konstruktion der optimalen trennenden Hyperebene das quadratische Optimierungsproblem

[1] BURGES (1998), S. 8 f.

$$\tau(\vec{w}) = \tfrac{1}{2}\|\vec{w}\|^2 \quad \rightarrow \min! \tag{7}$$

unter der Nebenbedingung $\quad k_i(\vec{w}\cdot\vec{x}_i + b) \geq 1 \qquad$ für $k_i \in \{-1,+1\}$, i=1, ..., L,

ergibt.[1] Einem zu klassifizierenden Objekt kann dann durch die Entscheidungsfunktion

$$e(\vec{x}) = \operatorname{sgn}\left((\vec{w}\cdot\vec{x}) + b\right) \tag{8}$$

die richtige Klasse zugeordnet werden.

Die primale Formulierung (7) des Problems lässt sich in eine Lagrange-Funktion mit Verwendung von λ_i als nicht-negative Lagrange-Multiplikatoren für jede Nebenbedingung[2] überführen:

$$LF(\vec{w},b,\lambda) = \frac{1}{2}\|\vec{w}\|^2 - \sum_{i=1}^{L}\lambda_i\left(k_i\left(\vec{w}\cdot\vec{x}_i + b\right) - 1\right) + \sum_{i=1}^{L}\lambda_i \qquad \text{mit } \lambda_i \geq 0\,. \tag{9}$$

Die Funktion *LF* muss minimiert werden bezüglich der (primalen) Variablen $\vec{w}$ und b bzw. maximiert in Bezug auf die Dualvariablen λ_i.[3] An der Stelle, an der die partiellen Ableitungen von *LF* gleich Null sind, befindet sich ein Sattelpunkt als lokales Minimum, welches aufgrund der Konvexität des Problems gleichzeitig ein globales Minimum ist.[4] Die Sattelpunkt-Bedingungen lauten also:

$$\begin{aligned} &\frac{\partial}{\partial b}LF = 0 \Rightarrow -\sum_{i=1}^{L}\lambda_i k_i = 0 \\ &\frac{\partial}{\partial \vec{w}}LF = 0 \;\Rightarrow\; \vec{w} - \sum_{i=1}^{L}\lambda_i k_i \vec{x}_i = 0 \;\Rightarrow\; \vec{w} = \sum_{i=1}^{L}\lambda_i k_i \vec{x}_i \end{aligned} \tag{10}$$

Der die optimale Hyperebene bestimmende Vektor $\vec{w}_{opt}$ kann also als Linearkombination von Lernmustern gebildet werden. Die Sattelpunkt-Bedingungen sind bei positiven Dualvariablen nur dann erfüllt, wenn die Nebenbedingungen des primalen Modells mit Gleichheit erfüllt sind. Unmittelbar sichtbar wird dieser Zusammenhang in den KUHN-TUCKER-Bedingungen[5]:

$$\lambda_i\left(k_i\left(\vec{w}\cdot\vec{x}_i + b\right) - 1\right) = 0 \qquad \text{für alle } i\text{=1, ..., } L. \tag{11}$$

Die zu einem Support-Vektor $\vec{x}_i$ gehörende Dualvariable λ_i ist größer Null, für alle anderen Vektoren der Lernmenge gleich Null. Das Ergebnis der Sattelpunkt-Bedingungen (10) des Dualproblems kann in das Primalproblem eingesetzt werden, in dem dann die Primalvariablen

---

1 SCHÖLKOPF$^+$ (1999a), S. 4.
2 Auch: Dualvariablen.
3 BURGES (1998), S. 8 f.; SCHÖLKOPF$^+$ (1999a), S. 4.
4 Zur Lösung eines quadratischen Optimierungsproblems durch Auffinden eines Sattelpunktes einer Lagrange-Funktion: siehe z. B. GROSSMANN$^+$ (1997).
5 Auch: KUHN-TUCKER-Komplementär-Bedingungen oder KARUSH-KUHN-TUCKER-Bedingungen (KKT).

verschwinden. Die Zielfunktion ist nur noch von den Dualvariablen abhängig. Es können Standardverfahren der mathematischen Optimierung, für das zu (7) gehörende duale quadratische Optimierungsproblem nach WOLFE[1]

$$\mathrm{T}\ (\lambda) = \sum_{i=1}^{L} \lambda_i - \frac{1}{2} \sum_{i=1}^{L}\ \sum_{j=1}^{L} \lambda_i \lambda_j \mathrm{k_i} \mathrm{k_j} \left(\vec{x}_i \cdot \vec{x}_j \right) \rightarrow \max! \tag{12}$$

unter den Nebenbedingungen $\lambda_\mathrm{i} \geq 0$ für i=1, ..., L und $\sum_{i=1}^{L} \lambda_i \mathrm{k_i} = 0$

eingesetzt werden.[2] Für eine Klassifizierung ergibt sich aus (7) mit $\vec{w} = \sum_{i=1}^{L} \lambda_i \mathrm{k_i} \vec{x}_i$ (aus (10)) die Entscheidungsfunktion für eine Klassifikation

$$e(\vec{x}) = \mathrm{sgn}\left(\sum_{i=1}^{L} \lambda_i \mathrm{k_i} \vec{x}_i \cdot \vec{x} + b \right). \tag{13}$$

Aus den KUHN-TUCKER-Bedingungen (11), umgeformt zu

$$\lambda_i \left(\left(\sum_{j=1}^{L} \lambda_j \mathrm{k_j} \vec{x}_j \cdot \vec{x}_i + b \right) - 1 \right) = 0 \qquad \text{für } i\text{=1, ..., } L, \tag{14}$$

lässt sich aus jedem Support-Vektor der Wert für b bestimmen.[3]

Die duale Formulierung des Problems ist vorteilhaft, weil die Nebenbedingungen einfacher handhabbar sind als im primalen Problem. Bemerkenswert ist, dass die Lerndaten nur als Skalarprodukte in die Problemlösung eingehen. Dadurch ist auch eine Verallgemeinerung für den nichtlinearen Fall möglich.[4]

4.5 Verallgemeinerung für linear nicht perfekt trennbare Fälle

Im Allgemeinen ist nicht davon auszugehen, dass Daten linear perfekt trennbar sind. Es sind Fehlklassifikationen zwar hinzunehmen, diese sind aber bei optimaler Trennung kleinstmöglich. Zur Bestimmung der optimalen Hyperebene werden Schlupfvariablen $\xi_i \geq 0$ eingeführt,

---

1 Siehe z. B. KÜNZI$^+$ (1962), S. 113 ff.

2 Mit Hilfe der *interior point method* kann z.B. ein globales Zielfunktionsminimum in polynomieller Zeit gefunden werden. (SMOLA (1998)) Außerdem kann eine Dekomposition, eine Zerlegung in mehrere kleinere Optimierungsprobleme, von Nutzen sein. (zu Algorithmen und deren Verbesserung siehe z. B. JOACHIMS (1999); OSUNA$^+$ (1999); PLATT (1999)).

3 Es kann aus numerischen Gründen günstig sein, zunächst für alle Support-Vektoren die einzelnen Werte für b zu berechnen und dann b als arithmetisches Mittel daraus zu berechnen. (BURGES (1998), S. 10 f., 15).

4 Graphisch kann eine Dualität des Problems veranschaulicht werden einerseits durch die Maximierung der Breite des Trennungsgürtels, andererseits die Teilung zwischen den nächstliegenden Punkten der durch die Datenpunkte beider Klassen gebildeten konvexen Hüllen. (BENNETT$^+$ (2000), S. 1 f.).

die Fehlklassifikationen, also Muster, die auf der „falschen“ Seite der (Hilfs-)Trennebene liegen, erlauben. In Analogie zu (6) ergibt sich durch Ergänzung

$$k_i(\vec{w}\cdot\vec{x}_i+b)\geq 1-\xi_i \qquad \text{für } k_i\in\{-1,+1\},\ \xi_i\geq 0,\ i=1,\ldots L. \tag{15}$$

Es wird nun einerseits die Kapazität bzw. Trenngüte durch $\vec{w}$ betrachtet, andererseits der Lernfehler als Abweichung von optimaler Trenngüte bei perfekt trennbaren Daten. Dieser Lernfehler kann in der Zielfunktion mittels einer Kostenfunktion, die Fehler bestraft, berücksichtigt werden, z.B. durch die mit υ parametrisierte Summe aller Schlupfvariablen $\sum_{i=1}^{L}\xi_i^{\upsilon}$, wobei mit steigendem υ die Komplexität der Berechnungen ansteigt. Typisch sind Werte $\upsilon=1$ (einfache Summe) oder $\upsilon=2$ (quadratische Kosten).[1] Ergänzt werden kann die Modellformulierung ferner durch einen vom Anwender wählbaren Kostenfaktor *C*, mit dem Lernfehler gewichtet werden können, um das Verhältnis von gewünschter Trenngüte und akzeptablem Lernfehler einzustellen.[2] Mit $C/L\sum_{i=1}^{L}\xi_i$ als gewichtetem Strafterm sind das primale und das duale Modell in Anlehnung an (7) und (12) formulierbar:

primal[3]:
$$\tau\ (\vec{w},b,\xi)=\frac{1}{2}\|\vec{w}\|^2+\frac{C}{L}\sum_{i=1}^{L}\xi_i\ \rightarrow \min! \tag{16}$$

unter den Nebenbedingungen
$$k_i(\vec{w}\cdot\vec{x}_i+b)\geq 1-\xi_i \qquad \text{für } k_i\in\{-1,+1\},\ i=1,\ldots,L$$
$$\xi_i\geq 0 \qquad \text{für } i=1,\ldots,L$$

dual:
$$T(\lambda)=\sum_{i=1}^{L}\lambda_i-\frac{1}{2}\sum_{i=1}^{L}\sum_{j=1}^{L}\lambda_i\lambda_j k_i k_j\left(\vec{x}_i\cdot\vec{x}_j\right)\ \rightarrow \max! \tag{17}$$

unter den Nebenbedingungen
$$0\leq\lambda_i\leq C \qquad \text{für } i=1,\ldots,L$$
$$\sum_{i=1}^{L}\lambda_i k_i=0 \qquad \text{für } i=1,\ldots,L$$

Im dualen Problem erscheinen die Schlupfvariablen aufgrund ihrer Linearität in der primalen Zielfunktion nicht mehr. Ferner haben die Dualvariablen eine durch *C* bestimmte obere Schranke, *C* bestimmt also die Anzahl an Support-Vektoren. Die Entscheidungsfunktion ist

---

1 VAPNIK (1995), S. 132 ff.; BURGES (1998), S. 14.

2 Mit höher werdendem *C* werden Fehler stärker bestraft, also weniger akzeptiert.

3 Aus der Zielfunktion ergibt sich der Bezug zur Risikoschranke (3), da aus der Norm von $\vec{w}$ die Schranke für die VC-Dimension abgeleitet werden kann (VAPNIK (1995), S. 128 f.). Somit wird in (16) die Summe aus empirischem Fehler und dem Komplexitätsmaß minimiert.

wieder durch (13) gegeben. Mit Hilfe der KUHN-TUCKER-Bedingungen kann *b* aus jedem Support-Vektor aus der Menge SV aller Support-Vektoren errechnet werden:[1]

$$b = \frac{1-\xi_i}{\mathrm{k_i}} - \vec{w} \cdot \vec{x}_i \tag{18}$$

bzw. als Durchschnitt

$$b = \frac{1}{|SV|} \sum_{i \in SV} \left(\frac{1-\xi_i}{\mathrm{k_i}} - \vec{w} \cdot \vec{x}_i \right), \; SV = \{ i \mid \lambda_i \neq 0 \} \; . \tag{19}$$

4.6 Nichtlineare Trennbarkeit

Die gezeigten Lösungsansätze zum Auffinden optimal trennender Hyperebenen können verallgemeinert werden, so dass auch Entscheidungsfunktionen gefunden werden können, die nicht linear von den Lerndaten abhängen. Gerade wirtschaftswissenschaftliche Probleme sind häufig nichtlinearer Natur. Um auch nichtlineare Funktionen bzw. nichtlineare Klassengrenzen lernen zu können, werden die Lerndaten mit einer Transformationsfunktion $\Phi : \mathbf{X} \rightarrow \boldsymbol{\Psi}$ aus dem Eingaberaum **X** in einen höherdimensionalen Merkmalsraum $\boldsymbol{\Psi}$ überführt und dort mit einer zu lernenden linearen Funktion separiert. In Abbildung 8 ist dieses Prinzip dargestellt.

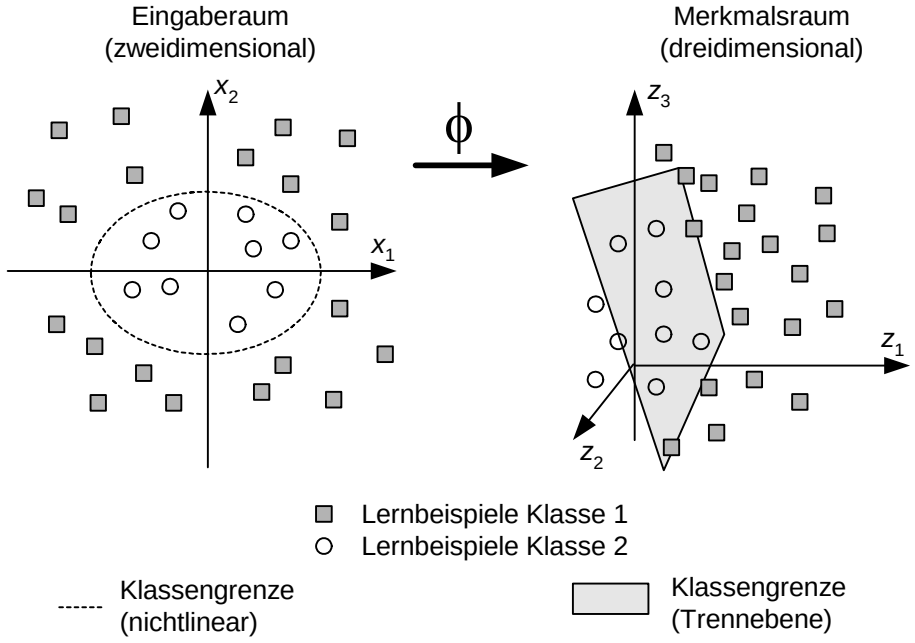


Abbildung 8: Φ-Transformation aus dem zweidimensionalen Eingaberaum mit nichtlinearer Klassengrenze in einen dreidimensionalen Merkmalsraum mit linearer Trennung durch eine Hyperebene
(in Anlehnung an SCHÖLKOPF (1997), S. 41)

[1] BURGES (1998), S. 13 ff.

Den ursprünglichen Musterattributen werden also weitere „künstliche“ Attribute, die sich durch nichtlineare Funktionen aus den Ursprungsdaten ableiten, zugefügt[1]. Wie in Abschnitt 4.4 gezeigt, geht in den Lernalgorithmus für je zwei Punkte $\vec{x}, \vec{y} \in \mathbf{X}$ nur das Skalarprodukt dieser Vektoren ein. Das duale Problem mit Berücksichtigung der Datentransformation

$$\mathrm{T}\ (\lambda) = \sum_{i=1}^{L} \lambda_i - \frac{1}{2} \sum_{i=1}^{L}\ \sum_{j=1}^{L} \lambda_i \lambda_j \mathrm{k_i} \mathrm{k_j} \Phi(\vec{x}_i) \cdot \Phi(\vec{x}_j)\ \rightarrow \max! \tag{20}$$

unter den Nebenbedingungen $\lambda_i \geq 0$ für i=1, ..., L

$$\sum_{i=1}^{L} \lambda_i \mathrm{k_i} = 0 \quad \text{für } i\text{=1, ..., } L$$

führt zur gelernten (im Eingaberaum nichtlinearen) Entscheidungsfunktion

$$e(\vec{x}) = \mathrm{sgn}\left(\sum_{i=1}^{L} \lambda_i \mathrm{k_i} \Phi(\vec{x}_i) \cdot \Phi(\vec{x}) + b \right) \tag{21}$$

für die es auch genügt, den Wert des Skalarprodukts $\Phi(\vec{x}_i) \cdot \Phi(\vec{x})$ zu kennen. Entscheidend für die Anwendung von SVM ist, dass die Transformation nicht tatsächlich durchgeführt werden muss, sondern dass es genügt, eine Kernel- oder Kernfunktion k zu kennen, für die eine Transformation $\Phi : \mathbf{X} \rightarrow \boldsymbol{\Psi}$ existiert, durch welche die Gleichung

$$k(\vec{x}, \vec{y}) = \Phi(\vec{x}) \cdot \Phi(\vec{y}) \quad \text{für alle } \vec{x}, \vec{y} \in \mathbf{X} \tag{22}$$

erfüllt wird. Bei Berechnung der Transformation würde insbesondere bei hochdimensionalen Daten durch die exponentiell ansteigende Dimensionalität des Merkmalsraums $\boldsymbol{\Psi}$ zum einen die Gefahr der Überanpassung, zum anderen der hohe Rechenaufwand problematisch werden. In SVM werden diese Probleme durch Verwendung von Kernelfunktionen umgangen. Aus den Bedingungen von MERCER[2] lassen sich Kernelfunktionen, die diese Bedingung erfüllen, so genannte MERCER-Kernel, ableiten. Beispiele und in Implementationen von SVM gängige Kernelfunktionen, die sich in einem Merkmalsraum $\boldsymbol{\Psi}$ als Skalarprodukt ausdrücken lassen, sind[3] [4]:

---

[1] Beispielsweise könnten Muster eines zweidimensionalen Eingaberaums $\{x_1, x_2\}$, die dort nicht linear zu trennen sind, in einen fünfdimensionalen Merkmalsraum als $\{x_1, x_2, x_1 x_2, x_1^2, x_2^2\}$ überführt und dort linear getrennt werden (BENNETT$^+$ (2000), S.4).

[2] VAPNIK (1995), S. 135 f.; SCHÖLKOPF (1997), S. 27 f., 156 ff.; BURGES (1998), S. 18.

[3] VAPNIK (1995), S. 137–141; BURGES (1998), S. 21; BENNETT$^+$ (2000), S. 5.

[4] Die VC-Dimension für SVM mit Polynomkernen wächst mit steigender Dimensionsanzahl d schnell an, für RBF-Kerne wird sie sogar unendlich. Insofern wirkt die Schranke (3) zur Risikoabschätzung nicht aussagekräftig genug. Dennoch sind zahlreiche empirische Ergebnisse bislang gut und sprechen für Anwendung und weitere Erforschung der Methode. Zu dieser Problematik und Lösungsansätzen wie „Margin Percentile

Polynom des Grades *d*	$k(\vec{x},\vec{y}) = (\vec{x}\cdot\vec{y}+1)^d,\ d \in \mathrm{N}$
Radiale Basisfunktion (RBF)	$k(\vec{x},\vec{y}) = \exp\left(\frac{-\|\vec{x}-\vec{y}\|^2}{2\sigma^2}\right), \sigma \in \Re$
sigmoides neuronales Netz (eine verdeckte Schicht)	$k(\vec{x},\vec{y}) = \tanh(\kappa\ \vec{x}\cdot\vec{y}-\delta)$ (für bestimmte κ und δ zur Einhaltung der MERCER-Kriterien). Satt tanh sind auch andere sigmoide Funktionen möglich.

Um eine nichtlineare Klassifikation durchzuführen, bedarf es also keiner algorithmischen Änderung des linearen Klassifikators, sondern es wird lediglich das Skalarprodukt der Vektoren durch eine geeignete Kernelfunktion ersetzt. Eine Berechnung erfolgt nicht explizit für die Transformationsfunktion $\Phi(\vec{x})$, sondern nur für die Kernelfunktion.

5 Bewertung des Verfahrens

Die Methode der SVM hat zahlreiche Vorteile, aber auch einige Nachteile oder Aspekte, die zumindest für konkrete Anwendungsfälle oder auch allgemein noch der Klärung bedürfen.[1] Die Relevanz von SVM für aCRM ergibt sich schon aus ihrer Eigenschaft als Klassifikator, denn viele der im aCRM Kontext anzutreffenden Fragestellungen, können als Klassifikationsproblem interpretiert werden.[2] Gegenüber den in der betrieblichen Praxis eingesetzten Verfahren (z. B. Entscheidungsbaumverfahren, künstliche neuronale Netze), bietet SVM folgende Vorteile:

- SVM sind durch die statistische Lerntheorie theoretisch gut fundiert. Über die Kapazitätskontrolle kann Überlernen eingeschränkt oder vermieden und somit eine gute Generalisierungsfähigkeit erreicht werden.
- Es gibt nur wenige Modellparameter, die vom Anwender festgelegt werden können, beziehungsweise müssen: eine Kostenfunktion sowie eine Kernelfunktion und ihre(n) Parameter. Eine komplizierte Parametrisierung, wie sie z. B. bei künstlichen neuronalen Netzen erforderlich ist, entfällt somit. Dadurch wird die Anwendung des Verfahrens durch Mitarbeiter entsprechender Fachabteilungen begünstigt.

Bounds“ und „Soft Margin Bounds“ siehe z. B. SHAWE-TAYLOR$^+$ (1998); BARTLETT$^+$ (1999); CRISTIANINI$^+$ (2002); SCHÖLKOPF$^+$ (2002).

1 BURGES (1998), S. 35; BENNETT$^+$ (2000), S. 9 f.

2 Zur Relevanz der Klassifikation siehe S. 13.

- SVM sind linear im Merkmalsraum, also konzeptionell und rechnerisch einfach; sie erzeugen aber nichtlineare Trennung im Eingaberaum.[1]
- Das Verfahren ist sehr flexibel. Allein die Auswahl eines neuen Kernels erzeugt einen neuen Klassifikator, ohne dass weitere Änderungen vorgenommen werden müssen. Durch die Wahl einer bestimmten Kernelfunktion können verschiedene Architekturen, die teilweise Ähnlichkeiten mit neuronalen Netzen haben, erzeugt werden, z. B. polynomiale Klassifikatoren, Radiale-Basisfunktionen-Klassifikatoren oder Multilayer Perceptrons.
- Mit SVM wird ein Mittelweg zwischen künstlichen neuronalen Netzen, die beliebig nichtlineare Zusammenhänge aufdecken können und Entscheidungsbaumverfahren, die intuitiv zu interpretieren, aber auf die Erkennung linearer Zusammenhänge beschränkt sind,[2] beschritten. Werden bei SVM lineare Kernfunktionen verwendet, kann die Relevanz eines Merkmals unmittelbar aus dem Gewichtungsvektor $\vec{w}$ abgeleitet werden.[3] Soll auf diese Transparenz verzichtet werden, können nichtlineare Zusammenhänge durch die Wahl einer entsprechend mächtigen Kernfunktion aufgedeckt werden.[4]
- Es liegt ein quadratisches Optimierungsproblem vor, das ein ermittelbares globales Optimum hat, welches mit bekannten, robusten Optimierungsmethoden gefunden werden kann. Es gibt keine algorithmisch bedingten Zufälligkeiten (wie teilweise bei neuronalen Netzen), sondern stabile, reproduzierbare Ergebnisse, unabhängig vom verwendeten Optimierungsalgorithmus oder von Initialisierungswerten.
- Es sind verschiedene Optimierungsalgorithmen speziell für verschiedene Problemstrukturen entwickelt worden.
- Es sind nicht nur symmetrische, sondern auch asymmetrische Kostenfunktionen ohne Effizienzverlust implementierbar.
- Es existieren diverse Erweiterungen, die eine noch flexiblere Einsetzbarkeit von SVM ermöglichen. Neben der vergleichsweise etablierten Support Vektor Regression sind hier Ansätze zu nennen, welche die Idee der maximal trennenden Hyperebene aufgreifen, um ein hybrides Entscheidungsbaumverfahren oder Segmentierungsmethoden zu konstruie-

---

1 SCHÖLKOPF$^+$ (1999b), S. 157.

2 Einige Entscheidungsbaumverfahren sind ferner auf die Erkennung univariater Muster beschränkt.

3 In GUYON$^+$ (2002) wird diese Transparenz z. B. ausgenutzt, um eine rekursive Elimination von weniger relevanten Inputmerkmalen vorzunehmen.

4 Durch einen Verzicht auf Nichtlinearität, kann die gleiche Transparenz auch bei neuronalen Netzen erreicht werden. Durch den Ansatz der maximalen Trennungsgüte und der impliziten Berücksichtigung der Generalisierungsfähigkeit wären lineare SVM solchen Architekturen aber überlegen.

ren.[1] Die Erzeugung eines probabilistischen Outputs ist ebenfalls möglich.[2] Die Methode der SVM kann auch für den Fall mit mehr als zwei Klassen eingesetzt werden, z. B. durch Kaskadierung mehrerer binärer SVM oder durch angepasste Modellformulierungen und Optimierungsalgorithmen.[3] Diese Ansätze unterstreichen noch einmal die Praxistauglichkeit von SVM. Neben der Abbildung von mehreren Klassen und der Forderung nach einer transparenten, nachvollziehbaren Lösung, ist die Möglichkeit, klassifizierte Objekte zu reihen, für praktische Problemstellungen besonders wichtig.[4] Genau dies wird durch probabilistische Outputs, die als Wahrscheinlichkeit interpretiert werden können, erreicht.

- Es gibt zahlreiche Anwendungen, die als „erfolgreich“ bezeichnet werden, z. B. in der Teile-, Handschriften-, Gesichtserkennung, Bioinformatik oder Textkategorisierung.

Weiterer Forschungsbedarf besteht aber noch, um unter anderem folgende Fragen zu beantworten:

- Sind SVM besser als die beste „handeingestellte“ Methode für ein bestimmtes Problem?
- Wie gut ist die Rechengeschwindigkeit und Ergebnisqualität bei sehr hoher Anzahl (Millionen) an Datensätzen und/oder Dimensionen und kann sie verbessert werden?
- Gibt es Anhaltspunkte aus dem Anwendungsfall für die Wahl einer gut geeigneten oder besten Kernelfunktion und ihrer Parameterwerte?[5]
- Wie ist domänenspezifisches Wissen in das Verfahren einzubinden?
- Wie wirken sich Skalierungen von Attributwerten auf die Ergebnisse (und die Rechenzeit) aus?
- Werden die theoretisch fundierten hohen Erwartungen in weiteren Anwendungsgebieten, speziell aus der Betriebswirtschaftslehre mit Nichtlinearitäten und hochdimensionalen Daten, erfüllt? Wie empfindlich sind Ergebnisse hinsichtlich der Anzahl an Datensätzen; sind Ergebnisse auch bei wenigen Datensätzen, bei denen der Einsatz neuronaler Netze problematisch ist, noch gut?
- Werden SVM-Algorithmen in Standardsoftwarepakete zum Data Mining integriert?[6]

---

1 BENNET+ (1998); BENNET + (1997); BEN-HUR+ (2001).
2 PLATT (2000).
3 Siehe z. B. ALLWEIN+ (2000); HSU+ (2001); SCHÖLKOPF+ (2002), S. 214.
4 Zum Beispiel Auswahl der 1000 besten Kunden für eine Direktmarketingkampagne.
5 Siehe z. B. CRISTIANINI+ (1998).
6 Die einzig bekannte kommerzielle Implementierung erfolgte im Rahmen des „KXEN Analytical Frameworks“ der Firma KXEN (URL: http://www.kxen.com).

Insgesamt konnte gezeigt werden, dass SVM viele Eigenschaften haben, die eine stärkere Verbreitung in der Praxis wünschenswert erscheinen lassen. Als Hürde könnte sich insbesondere die zurzeit noch unzureichende Softwareunterstützung erweisen. Bisherige Implementierungen sind vorwiegend zu Forschungszwecken erfolgt und für den praktischen Einsatz noch nicht geeignet.

Für das CRM bieten sich SVM besonders an, da – zusammen mit methodischen Erweiterungen (Regression, Cluster) – fast alle analytischen Fragestellungen mit einem Verfahren behandelt werden können. Die Grundidee der maximal trennenden Hyperebene bleibt dabei stets erhalten, so dass Anwender sich das nötige methodische Basiswissen relativ einfach aneignen können. Dies stellt einen großen Vorteil gegenüber der aktuellen Situation dar, die häufig durch den Einsatz individueller Verfahren für spezifische Anwendungsdomänen gekennzeichnet ist.

Literatur

ALLWEIN+ (2000) Allwein, Erin L.; Schapire, Robert E.; Singer, Yoram: Reducing multiclass to binary: A unifying approach for margin classifiers. *Journal of Machine Learning Research*, 1 (2000), S. 113–141.

BÄCK+ (2001) Bäck, Thomas; Schütz, Martin: Evolutionäre Algorithmen im Data Mining. In: Hippner, Hajo; Küsters, Ulrich; Meyer, Mathias; Wilde, Klaus (Hrsg.): *Handbuch Data Mining im Marketing: Knowledge Discovery in Marketing Databases*. Wiesbaden: Vieweg, 2001, S. 403–428.

BACKHAUS+ (2000) Backhaus, Klaus; Erichson, Bernd; Plinke, Wulff; Weiber, Rolf: *Multivariate Analysemethoden : Eine anwendungsorientierte Einführung*. 9. überarb. Aufl., Berlin: Springer, 2000.

BARTLETT+ (1999) Bartlett, Peter; Shawe-Taylor, John: Generalization Performance of Support Vector Machines and other Pattern Classifiers. In: Schölkopf, Bernhard; Burges, Christopher J.C.; Smola, Alexander J. (Hrsg.): *Advances in Kernel Methods: Support Vector Learning*. Cambridge, Mass.: MIT Press, 1999, S. 43–54.

BEN-HUR+ (2001) Ben-Hur, Asa; Horn, David; Siegelmann, Hava T.; Vapnik, Vladimir: Support vector clustering. *Journal of Machine Learning Research*, 2 (2001), S. 125–137.

BENNETT+ (1997) Bennet, Kristin P. ; Blue, Jennifer A.: *A Support Vector Approach to Decision Trees*. Arbeitspapier, Rensselaer Polytechnic Institute, 1997.

BENNETT+ (1998) Bennet, Kristin P.; Wu, Donghui; Auslender, Leonardo: On *Support Vector to Decision Trees for Database Marketing*. Arbeitspapier, Rensselaer Polytechnic Institute, 1998.

BENNETT+ (2000) Bennett, Kristin P.; Campbell, Colin: Support vector machines: Hype or hallelujah? *SIGKDD Explorations*, 2 (2000), Nr. 2, S. 1–13.

BERRY+ (1997) Berry, Michael J.A.; Linoff, Gordon: *Data Mining Techniques for Marketing,Sales and Customer Support*. New York: Wiley, 1997.

BERSON+ (1999) Berson, Alex; Smith, Stephen; Thearling, Kurt: Building Data Mining Applications for CRM. New York: McGraw Hill, 1999.

BEYER (2003) Beyer, Thomas C.: *Kennen Sie Ihre wertvollsten Kunden*? Online im Internet. (`http://www.phil.uni-erlangen.de/economics/bwl/bpract/kuwert/kuwert.pdf`). Von Verfassern zuletzt geladen am 23.01.2003.

BISHOP (1995) Bishop, Christopher M.: *Neural Networks for Pattern Recognition*. Oxford: Clarendon Press, 1995.

BURGES (1998) Burges, Christopher J. C.: A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery* 2 (1998), Nr. 2, S. 121–167.

CABENA+ (1997) Cabena, Peter; Hadjnian, Pablo; Stadler, Rolf; Verhees, Jaap; Zanasi, Alessandro: *Discovering Data Mining: From Concept to Implementation.* London: Prentice Hall, 1997.

CHAMONI (2001) Chamoni, Peter: On-Line Analytical Processing (OLAP). In: Hippner, Hajo; Küsters, Ulrich; Meyer, Mathias; Wilde, Klaus (Hrsg.): *Handbuch Data Mining im Marketing: Knowledge Discovery in Marketing Databases*. Wiesbaden: Vieweg, 2001, S. 543–556.

CRISTIANINI+ (1998) Cristianini, Nello; Campbell, Colin; Shawe-Taylor, John: *Dynamically Adapting Kernels in Support Vector Machines*. In: *Advances in Neural Information Processing Systems 11: Proceedings of the 12th Annual Conference on Neural Information Processing Systems (NIPS)*, Cambridge, Mass.: MIT Press (1999), S. 204–210.

CRISTIANINI+ (2002) Cristianini, Nello; Shawe-Taylor, John: *An Introduction to Support Vector Machines and other Kernel-based Learning Methods*. Cambridge: Cambridge University Press, 2002.

FAYYAD (1996) Fayyad, Usama; Piatetsky-Shapiro, Gregory; Smyth, Padhraic: From data mining to knowledge discovery in databases: An overview. *AI Magazine*, 17 (1996), Nr. 3, S. 37–54.

FINK+ (2001) Fink, Andreas; Schneidereit, Gabriele; Voß, Stefan: Grundlagen der Wirtschaftsinformatik. Heidelberg: Physica-Verlag, 2001.

FREITAS (2002) Freitas, Alex A.: *Data Mining and Knowledge Discovery with Evolutionary Algorithms*. Berlin: Springer 2002.

GRABMEIER (2001) Grabmeier, Johannes: Segmentierende und Clusterbildene Methoden. In: Hippner, Hajo; Küsters, Ulrich; Meyer, Mathias; Wilde, Klaus (Hrsg.): *Handbuch Data Mining im Marketing: Knowledge Discovery in Marketing Databases*. Wiesbaden: Vieweg, 2001, S. 299–362.

GROSSMANN+ (1997) Großmann, Christian; Terno, Johannes: *Numerik der Optimierung*. 2. durchges. Aufl., Stuttgart: Teubner, 1997.

GUYON+ (2002) Guyon, Isabelle; Weston, Jason; Barnhill, Stephen; Vapnik, Vladimir: *Gene selection for cancer classification using support vector machines*. Machine Learning, 46 (2002), Nr. 1-3, S. 389–422.

HAND+ (2001) Hand, David; Mannila, Heikki; Smyth, Padhraic: Principles of Data Mining. Cambridge, Mass.: MIT Press, 2001.

HERTZ+ (1991) Hertz, John A.; Krogh, Anders S.; Palmer, Richard G.: *Introduction to the Theory of Neural Computation*. Redwood City: Addison-Wesley, 1991.

HETTICH+ (2001) Hettich, Stefanie; Hippner, Hajo: Assoziationsanalyse. In: Hippner, Hajo; Küsters, Ulrich; Meyer, Mathias; Wilde, Klaus (Hrsg.): *Handbuch Data Mining im Marketing: Knowledge Discovery in Marketing Databases*. Wiesbaden: Vieweg, 2001, S. 427–464.

HIPPNER+ (2001) Hippner, Hajo; Wilde, Klaus: Der Prozess des Data Mining im Marketing. In: Hippner, Hajo ; Küsters, Ulrich ; Meyer, Mathias ; Wilde, Klaus (Hrsg.): *Handbuch Data Mining im Marketing : Knowledge Discovery in Marketing Databases*. Wiesbaden : Vieweg, 2001, S. 22–94.

HIPPNER+ (2002a) Hippner, Hajo; Wilde, Klaus D.: CRM - Ein Überblick. In: Helmke, Stefan; Uebel, Matthias; Dangelmaier, Wilhelm (Hrsg.): *Effektives Customer Relationship Management*. 2. überarb. und erweiterte Aufl., Wiesbaden: Gabler, 2002, S. 3–38.

HIPPNER+ (2002b) Hippner, Hajo; Wilde, Klaus D.: Data Mining im CRM. In: Helmke, Stefan; Uebel, Matthias; Dangelmaier, Wilhelm (Hrsg.): *Effektives Customer Relationship Management*. 2. überarb. und erweiterte Aufl., Wiesbaden: Gabler, 2002, S. 211–232.

HOLLAND+ (2001) Holland, Heinrich; Huldi, Christian; Kuhfuß, Holger; Nitsche, Martin: *CRM im Direktmarketing : Kunden gewinnen durch interaktive Prozesse*. Wiesbaden: Gabler, 2001.

HSU+ (2001) Hsu, Chih-Wie; Lin, Chih-Jen: *A Comparison of Methods for Multi-class Support Vector Machines*. Arbeitspapier, Department of Computer Science and Information Engineering, National Taiwan University, 2001.

HUNSEL+ (2000) Hunsel, Lothas; Zimmer, Sabine: Kundenwert und Kundenloyalität. In: Hofmann, Markus; Mertiens, Markus (Hrsg.): *Customer-Lifetime-Value Management: Kundenwert schaffen und erhöhen: Konzepte, Strategien, Praxisbeispiele*. Wiesbaden: Gabler, 2000, S. 115–128.

JOACHIMS (1999) Joachims, Thorsten: Making Large-Scale Support Vector Machine Learning Practical. In: Schölkopf, Bernhard; Burges, Christopher J.C.; Smola, Alexander J. (Hrsg.): *Advances in Kernel Methods: Support Vector Learning*. Cambridge, Mass.: MIT Press, 1999, S. 169–184.

KOHONEN (2001) Kohonen, Teuvo: *Self-organizing Maps*. 3. Aufl., Berlin: Springer, 2001.

KOTLER+(1997) Kotler, Philip; Bliemel, Friedhelm: *Marketing-Management : Analyse, Planung, Umsetzung und Steuerung*. 8. vollständig neu bearbeitete und erweiterte Aufl., Stuttgart: Schäffer-Poeschel, 1995.

KRAFFT (2002) Krafft, Manfred: *Kundenbindung und Kundenwert*. Heidelberg: Physica-Verlag, 2002.

KÜNZI+ (1962) Künzi, Hans Paul; Krelle, Wilhelm; Oettli, Werner: *Nichtlineare Programmierung*. Berlin: Springer, 1962.

KÜSTERS (2001) Küsters, Ulrich: Data Mining Methoden: Einordnung und Überblick. In: Hippner, Hajo; Küsters, Ulrich; Meyer, Mathias; Wilde, Klaus (Hrsg.): *Handbuch Data Mining im Marketing: Knowledge Discovery in Marketing Databases*. Wiesbaden: Vieweg, 2001, S. 95–130.

LINK+ (1997) Link, Jörg; Hildebrand, Volker: Grundlagen des Database Marketing. In: Link, Jörg; Brändli, Dieter; Schleuning, Christian; Kehl, Roger E. (Hrsg.): Handbuch Database Marketing. Ettlingen: IM Fachverlag Marketing-Forum, 1997, S. 15–38.

OSUNA+ (1999) Osuna, Edgar E.; Girosi, Federico: Reducing the Run-time Complexity in Support Vector Machines. In: Schölkopf, Bernhard; Burges, Christopher J.C.; Smola, Alexander J. (Hrsg.): *Advances in Kernel Methods: Support Vector Learning*. Cambridge, Mass.: MIT Press, 1999. S. 271–283.

PATTERSON (1996) Patterson, Dan W.: *Artificial Neural Networks: Theory and Applications*. Singapur: Prentice Hall, 1996.

PLATT (2000) Platt, John C.: Probabilities for SV Machines. In: Smola, Alexander J.; Bartlett, Peter J.; Schölkopf, Bernhard; Schuurmans, Dale (Hrsg.): *Advances in Large Margin Classiers*. Cambridge, Mass.: MIT Press, 2000, S. 61–74.

PLATT (1999) Platt, John C.: Fast Training of Support Vector Machines Using Sequential Minimal Optimization. In: Schölkopf, Bernhard; Burges, Christopher J.C.; Smola, Alexander J. (Hrsg.): *Advances in Kernel Methods: Support Vector Learning*. Cambridge, Mass.: MIT Press, 1999, S. 185–208.

RAAB+ (2000) Raab, Gerhard; Lorbacher, Nicole: Customer Relationship Management. Heidelberg: Sauer-Verlag, 2002.

RAPP+ (1999) Rapp, Reinhold; Guth, Sebastian: Data Mining Anwendungen im Relationship Marketing. In: Payne, Adrian; Rapp, Reinhold (Hrsg.): *Handbuch Relationship Marketing*. München: Vahlen, 1999, S. 245–260.

RUD (2001) Rud, Olivia Pad: *Data Mining Cookbook*. New York: Wiley, 2001.

SCHMID (2001) Schmid, Roland: *Architektur für das Customer Relationship Management und Prozessportale bei Banken*. St. Gallen, Hochschule für Wirtschafts-, Rechts- und Sozialwissenschaften (HSG), Fachbereich Wirtschaftswissenschaften, Dissertation, April 2001. Bamberg: Difo-Druck, 2001.

SCHÖLKOPF (1997) Schölkopf, Bernhard: *Support Vector Learning*. München: Oldenbourg, 1997.

SCHÖLKOPF+ (1999a) Schölkopf, Bernhard; Burges, Christopher J.C.; Smola, Alexander J.: Introduction to Support Vector Learning. In: Schölkopf, Bernhard; Burges, Christopher J.C.; Smola, Alexander J. (Hrsg.): *Advances in Kernel Methods: Support Vector Learning*. Cambridge, Mass.: MIT Press, 1999, S. 1–15.

SCHÖLKOPF+ (1999b) Schölkopf, Bernhard; Müller, Klaus-Robert; Smola, Alexander J.: Lernen mit Kernen: Support-Vektor-Methoden zur Analyse hochdimensionaler Daten. *Informatik Forschung und Entwicklung*, 14 (1999), Nr. 3, S. 154–163.

SCHÖLKOPF+ (2002) Schölkopf, Bernhard; Smola, Alex J.: *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond.* Cambridge, Mass.: MIT Press, 2002.

SCHÜRMANN (1996) Schürmann, Jürgen: *Pattern Classification: a Unified View of Statistical and Neural Approaches*. New York: Wiley, 1996.

SCHULZE (2000) Schulze, Jens: *Prozessorientierte Einführungsmethode für das Customer Relationship Management.* St. Gallen, Hochschule für Wirtschafts-, Rechts- und Sozialwissenschaften (HSG), Fachbereich Wirtschaftswissenschaften, Dissertation, April 2000. Bamberg: Difo-Druck, 2000.

SCHULZE (2002) Schulze, Thomas: Erfolgsorientiertes Customer Relationship Marketing (CRM) auf der Basis von Business Intelligence (BI)-Lösungen. In: Helmke, Stefan; Uebel, Matthias; Dangelmaier, Wilhelm (Hrsg.): *Effektives Customer Relationship Management*. 2. überarb. und erweiterte Aufl., Wiesbaden: Gabler, 2002, S. 234–255.

SHAWE-TAYLOR+ (1998) Shawe-Taylor, John; Bartlett, Peter L.; Williamson, Robert C.; Anthony, Martin: Structural risk minimization over data-dependent hierarchies. *IEEE Transactions on Information Theory* 44 (1998), Nr. 5, S. 1926–1940.

SMIDT+ (2001) Smidt, Wolfhart; Marzian, Sieghard: *Brennpunkt Kundenwert: Mit dem Customer Equity Kundenpotentiale erhellen, erweitern und ausschöpfen*. Berlin: Springer, 2001.

SMOLA (1998) Smola, Alexander Johannes: *Learning with Kernels*. Berlin, Technische Universität, Fachbereich Informatik, Dissertation, November 1998.

STAHLBOCK (2002) Stahlbock, Robert: *Evolutionäre Entwicklung künstlicher neuronaler Netze zur Lösung betriebswirtschaftlicher Klassifikationsprobleme*. Berlin: WiKu, 2002.

TIETZ+ (2001) Tietz, Christiane; Poscharsky, Nikolaus; Erichson, Bernd; Müller, Holger: Ein Vergleich von Data Mining-Methoden zur Cross-Selling-Optimierung von Finanzprodukten. In: Hippner, Hajo; Küsters, Ulrich; Meyer, Mathias; Wilde, Klaus (Hrsg.): *Handbuch Data Mining im Marketing: Knowledge Discovery in Marketing Databases*. Wiesbaden: Vieweg, 2001, S. 767–786.

WITTEN+ (2001) Witten, Ian H.; Frank, Eibe: *Data Mining: Praktische Werkzeuge und Techniken für das maschinelle Lernen.* München: Hanser, 2001.

VAPNIK (1982) Vapnik, Vladimir N.: *Estimation of Dependences Based on Empirical Data.* New York: Springer, 1982.

VAPNIK (1995) Vapnik, Vladimir N.: *The Nature of Statistical Learning Theory.* New York: Springer, 1995.

VAPNIK (1998) Vapnik, Vladimir N.: *Statistical Learning Theory*. New York: Wiley, 1998.

VOSS+(2001) Voß, Stefan; Gutenschwager, Kai: *Informationsmanagement*. Berlin: Springer, 2001.

ZELL (2000) Zell, Andreas: Simulation Neuronaler Netze. 3. unveränd. Nachdr., München: Oldenbourg, 2000.