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 $X = \{\boldsymbol{x}_i, ..., \boldsymbol{x}_n\}$ 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, $f_a(\boldsymbol{x}_i) \approx y_i \; \forall i = (1, ..., i)$, 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 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 $g(x)$: $X \rightarrow Y$ 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_{0j} 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 paradigm 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]).

$$\text{max.} \quad W(\lambda)=\sum_{i=1}^{l}\lambda_i-\frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}\lambda_i\lambda_j y_i y_j\left(\boldsymbol{x}_i\cdot\boldsymbol{x}_j\right)$$

$$\text{s.t.} \quad 0\le\lambda_i\le C \ ; \ \sum_{i=1}^{l}\lambda_i y_i=0 \ (i=1,\dots,l) \qquad (1)$$

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 $\Phi(x)\cdot\Phi(x_i)$ in a high-dimensional space can be avoided by introducing a kernel function $K(\boldsymbol{x},\boldsymbol{x}_i)=\Phi(x)\cdot\Phi(x_i)$ [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 I

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.

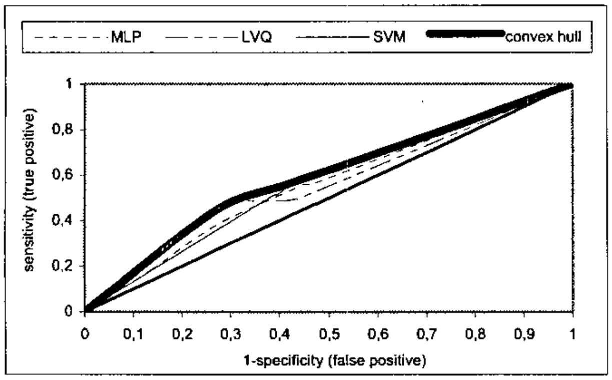

Fig. 1. ROC-Chart of SVM, MLP and LVQ performance in experiment A.1 to C.3, including the resulting convex hull.

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.