Extracting Rules from Support Vector Machines

Klaus B. Schebesch and Ralf Stecking

Institut für Konjunktur- und Strukturforschung,
Universität Bremen, D-28359 Bremen, Germany

Abstract. Support Vector Machines (SVM) from statistical learning are very powerful methods which can be used as (e.g.binary) classifiers or discriminators in a wide range of applications. Advantages of SVM are that weak prior assumptions about both model and data suffice. Moreover, optimization of the SVM essentially regularizes the emerging data model by restricting the model to special data points, the support vectors, usually a small subset from the training data. In our paper we discuss ways of detecting informative and typical subsets from SVM solutions, with the aim of extracting simple rules.

1 Introduction

Classification of labeled high dimensional data is an important step within computational decision support in many industries. Classification rules obtained from large data sets can often be used as decision rules. An example is the decision of whether to accept or to reject a new credit applicant by a bank. Using personal records on many past credit applicants as a training set, a method from Discriminant Analysis extracts a classification rule, which is used to forecast whether a new credit applicant is likely to default. Among the many methods for separating data into classes we mention Linear Discriminant Analysis (LDA), which is computationally cheap and most suitable for linearly separable data but also more expensive methods like Support Vector Machines (SVM), which can "lift" arbitrarily complex data into high dimensional feature spaces until all nonlinearities are disentangled such that linear separation becomes possible [9]. SVM identify special data points (the support vectors) from the set of input data, which describe the boundaries between the cases. However, as an essentially geometrical concept in high dimensional space they are not of any direct use to the practitioner.

There are several alternative approaches which express classification by a small set of conjunctions / disjunctions or other standardized "linguistic" rules [5], or which link fuzzy decision rules with certain non-linear SVM [2]. However, many of these approaches are "open ended" as they employ evolutionary search and genetic programming to discover useful rules from the vast number of feasible rules [3] and, hence, they also tend to be unacceptable for practitioners.

Classification and regression trees (CART) in the sense of [1] are a widely accepted procedure for building decision rules based on recursively placing

split points on the values of single input variables. This recursive partitioning is simple and inexpensive, but compared to general rule systems its expressive power is limited.

While using SVM has been shown to be useful for credit scoring [10], [8] one would still like to use the attractively simple features of recursive decision trees as a back end for practitioners. In the remaining part of the paper we therefore propose to use the support vectors of an SVM solution as subset preselection for computation of the decision trees. After a short description of the SVM mechanism and different types of support vectors, we first show the reasonability of the approach with some two dimensional "toy" examples and then proceed with the results on real life credit data.

2 SVM and different types of support vectors

Following [8] we give an abridged exposition of the SVM mechanism. Given $N > 0$ training cases $\{x_i, y_i\}_{i=1}^N$, with input data $x_i \in X \subset \mathbb{R}^m$ (e.g. the description of a credit applicant) and the associated class labels $y_i \in \{-1, +1\}$ (e.g. the recorded defaulting behavior), assume a feature space $\mathcal{F}$, with $\dim(\mathcal{F}) \geq \dim(X)$, i.e. at least as expressive as input space X. There should also exist a map which transform points x into points u. Now imagine $u \in \mathcal{F}$ being such that a given, but arbitrarily distributed data set from X can now be linearly separated in $\mathcal{F}$. The central ingredient of SVM is to linearly separate points in $\mathcal{F}$ by a **maximal margin**, i.e. points form one class described by $\langle u_{(-1)}, w\rangle + b \leq -1$ should be as much apart as possible from the points of the other class $\langle u_{(+1)}, w\rangle + b \geq 1$, which finally amounts to maximize $2/||w||$. While this would minimize the expected misclassification rate, in practice, hard margins are replaced by soft margins by minimizing $||w||$ or

$$C\sum_{i=1}^{N}\zeta_i \; + \; \min_{w,b}\frac{1}{2}\langle w, w\rangle \quad \text{s.t.} \;\; y_i[\langle u_i, w_i\rangle + b] \geq 1 - \zeta_i, \quad i = 1, ..., N,$$

with slacks $\zeta_i \geq 0$. Capacity $C > 0$ now controls the misclassification rate which is used to avoid over-fitting to the data. The associated dual SVM, which disposes of the use of unknown space $\mathcal{F}$, reads

$$\max_{\alpha} \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j k(x_i, x_j) \quad \text{s.t.} \quad \sum_{i=1}^{N} y_i\alpha_i = 0, \quad C \geq \alpha_i \geq 0.$$

The "kernels" $k(.,.)$ are just scalar products of the inputs $\langle x_i, x_j\rangle$ in the case of linear SVM, i.e. if $u_i = x_i$ or $(X = \mathcal{F})$. In general $(X \neq \mathcal{F})$, they are chosen from some set of nonlinear functions like e.g. $\exp(s||x_i - x_j||^2)$, $s > 0$.

A data point x_i is called a **support vector** if $\alpha_i > 0$ (i.e. the point i is a binding inequality in the primal). Note that there are at least two kinds of