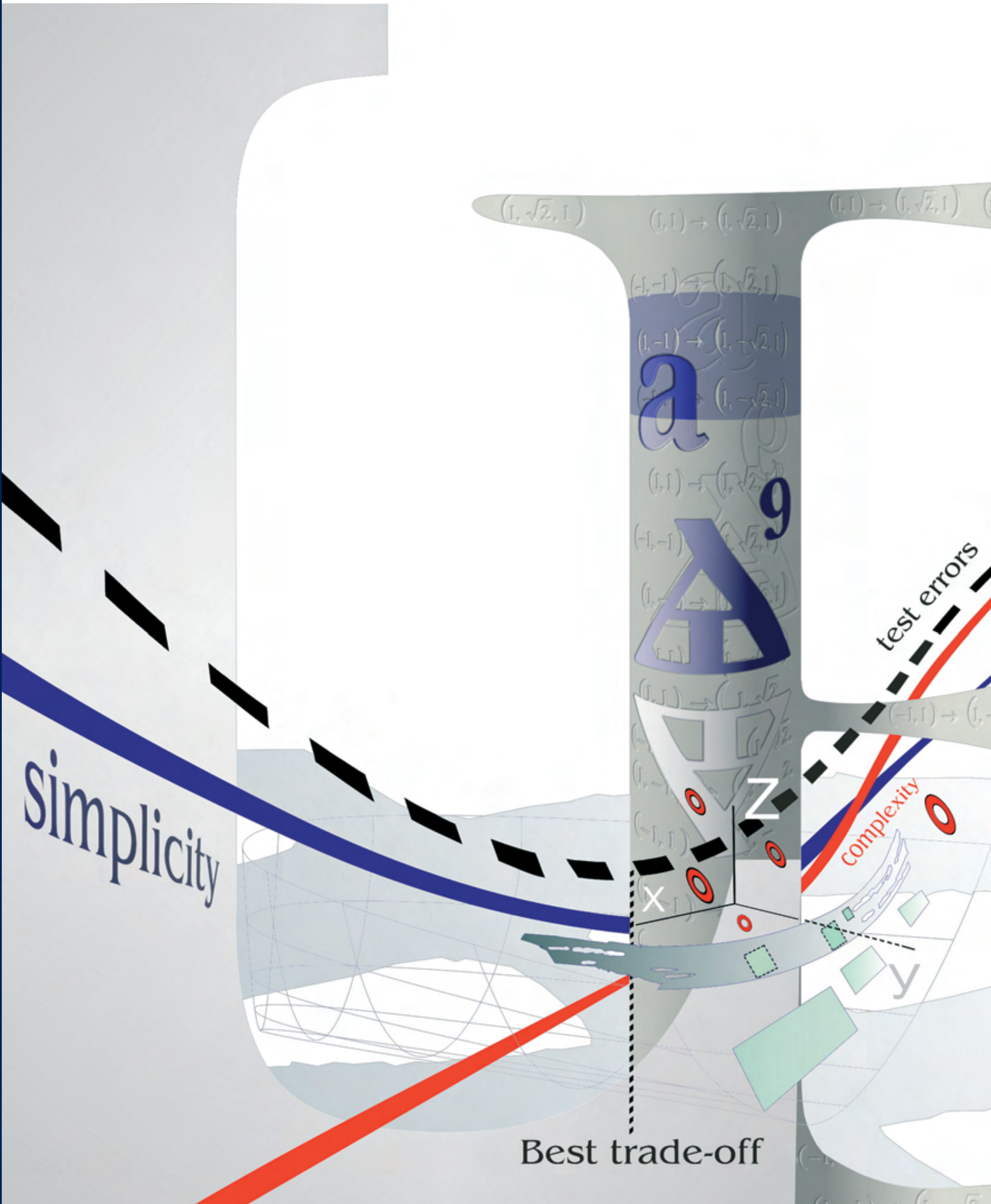
GEORGI NALBANTOV

Essays on Some Recent Penalization Methods

with Applications in Finance and Marketing



Essays on Some Recent Penalization Methods with Applications in Finance and Marketing

Essays on Some Recent Penalization Methods with Applications in Finance and Marketing

Essays over recente regularisatiemethoden en hun toepassingen in financiering en marketing

Thesis

**to obtain the degree of Doctor from the
Erasmus University Rotterdam
by command of the
rector magnificus**

Prof.dr. S.W.J. Lamberts

and according to the decision of the Doctorate Board

The public defense shall be held on

Thursday 11 September 2008 at 13:30 hrs

by

**Georgi Ilkov Nalbantov
born in Pernik, Bulgaria**

ERASMUS UNIVERSITEIT ROTTERDAM

Doctoral Committee:

Promoter:	Prof.dr. P.J.F. Groenen
Copromoter:	Dr. J.C. Bioch

Other members:

Prof.dr. P.H.B.F. Franses
Prof.dr. Eric O. Postma
Prof.dr. Albert P.M. Wagelmans

Erasmus Research Institute of Management (ERIM)
Econometric Institute
Erasmus University Rotterdam

Internet: http://www.erim.eur.nl

ERIM Electronic Series Portal: http://hdl.handle.net/1765/1

ERIM Ph.D. Series Research in Management 132

ISBN 978 - 90 - 5892 - 166 - 6

Cover illustration:
© 2008, Dimitar Kafedjiev

Acknowledgements

First and foremost, I would like to thank a person that may be caught by surprise, and who comes in here even before my relatives, friends, and people I have collaborated with. This person is Prof.dr. Hans van Mierlo from Maastricht University, who is the indispensable and irreplaceable part in a bigger system called in general bureaucracy, without whom I would never have started my university education in The Netherlands in the first place, and therefore would never have completed a PhD thesis in the present form. He was the one who signed as my so-called guarantor after knowing me for just a couple of weeks, and without whom permission to study in this country would not have been granted to me back in 1998. I can only say: a big thank you!

Of course, there are many people who have helped me grow up as an academic and inadvertently have pushed me into becoming a research-minded person. One of them is Evgueni Smirnov, who asked me about 5 years ago "What do you want to do after you graduate from Maastricht University". I answered then, without much thinking, "I would like to start a PhD". "You can do that only if you think that there is something you would like to tell to the world!", he replied. At that time I had nothing to tell, of course, but five years later I hope I have lived up to his expectations. He is the one who introduced me to the Machine Learning world, which I was blissfully unaware of, as I was studying to become an Economist or Econometrician. As is turned out, I have combined these fields in my research, thanks to – in addition to Evgueni Smirnov – Ida Sprinkhuizen-Kuyper and Rob Bauer (my Master thesis supervisors from Maastricht University), and Patrick Groenen and Cor Bioch (my supervisors from Erasmus University Rotterdam). In this respect, it would be a crime if a miss to stress that one of the most productive periods during my PhD was the trip to Stanford University on account of Trevor Hastie, to whom I am obliged for inviting me over to spend some time in the Statistical Department.

There is one special person that I would like to thank, who is not a researcher, but without whom research for me would never have been possible, as least in this form. That is Marco from Maastricht, the best cook ever, who gave a big shoulder and literary helped me survive all those student years in Maastricht.

I would also like to thank my wife, Milena, for the patience she had (and

still has, I hope) with me while I was writing the thesis. I realize this is a banal phrase that many people put in the acknowledgements, but I am sure that they know what they mean exactly. There is a whole list of people I would like to thank further on, but there is always a danger of missing somebody important. Therefore, finally, I would like to thank all my friends, relatives, colleagues and teachers who have supported me so far and who, I am confident, will continue to support me in the future.

Georgi Nalbantov
Rotterdam, September 2008

Contents

Chapter 1

Introduction

The term learning in the context of this thesis is about extracting knowledge from data. This term has penetrated deeply into fields such as Machine Learning and Statistics, where it is often referred to as function estimation. A famous example that illustrates an ubiquitous learning dilemma is the so-called "tree" example. Here, a boy and his younger sister are given several pictures. Some pictures show trees, other pictures show other items. Both the trees and the other items are known to the children. Later, a new picture with a tree is presented to the children and the task is to say whether the item shown is a tree or not. The boy, having studied the pictures extremely closely concludes (incorrectly) that the item is not a tree, since he has never seen a tree with exactly so many branches. The younger girl, who has paid little attention to the pictures, concludes (correctly) that the item in the new picture is a tree, because it is green. The typical learning dilemma illustrated here is whether, based on a finite number of examples, to find a learning rule that describes the data too well, or to construct a learning rule that is a simple one. Inevitably, the "right" balance, or trade-off, has to be sought. The idea that there is a need to search for such a trade-off is one of the main motivation pillars of this thesis. Depending on the area of research, this trade-off is referred to as bias-versus-variance, or fit-versus-complexity. In essence, the way to achieve a balance in practice is to start from a function that fits, or describes, a given data too well, and then to "punish" it for being too precise. Loosely speaking, in this way we end up with a function, or learner, that finds itself somewhere between the boy and the girl. More formally, the act of punishing is called penalization or regularization.

While a rigorous universally-acknowledged definition of the term learning has not crystallized in the literature, distinct types of learning have emerged. Among the most popular types, where the penalization idea stands out as quite relevant, are supervised and unsupervised learning. Supervised learning involves knowledge about the outcomes associated with some data. These techniques handle data with usually one output variable and many input variables. Generally speaking, a variable is something that has a magnitude and varies. The input variables are also referred to as inputs, explanatory variables, predictors, attributes, or features, depending on the area of research. In case the output variable is allowed to take real values, it is likewise also called the explained, predicted or response variable. The financial problem of predicting the expected stock market return from a linear combination of several financial input variables is an example of such a supervised input-output problem, and is generally called a regression problem, since the output variable is real valued. The bias-variance trade-off in this linear regression context reveals itself as the compromise choice between a function that fits the data too well, which is the linear function estimated by a so-called Ordinary Least Squares procedure, and a function that provides a constant value for the output variable whatever the inputs.

A second example of a supervised learning method is classification. Classification resembles regression with only one crucial difference: the output variable can take only a finite number of discrete values, or *labels*. If this variable takes only two values, then the learning process is referred to as binary classification, and sometimes even as binary, or binomial, regression. Here, the output variable is also called a group, a class, or a nominal explanatory variable. The tree example above represents a binary classification task. Another example of such a task is predicting whether the stock market is expected to go “up” or “down”.

In contrast to supervised learning methods where the outcome variable is known *a priori*, unsupervised methods seek some kind of structure in the data without a known outcome variable. A good example of an unsupervised technique is clustering. Roughly speaking, clustering is about finding a certain structure in the data based on the inputs only, whereas classification and regression are about finding relations in the data that describe the concrete way in which inputs affect outputs. In this thesis, only supervised learning tasks are addressed however, and more specifically classification and regression problems.

Thesis Topics

This thesis focuses on three major themes. The first one glides in-between the surfaces of Statistics/Econometrics and Machine Learning, without a clear priority of one over the other. Actually, it is the combination of the advances in these fields that gives rise to possible breakthroughs. The thesis combines and builds on three branches of methods, namely *instance-based*, *penalization* and *kernel* methods. The combination among these three branches, or aspects, has given rise to three novel classification methods proposed in this thesis: Support Hyperplanes (SH), Soft Nearest Neighbor (SNN) and Nearest Convex Hull (NCH) classification. To the best of my knowledge, these are the first methods that combine all of the above three aspects and benefit greatly from the valuable qualities inherent in each of them. Each of the aspects has its own merits, discussed very shortly below. Where applicable, the effect of one aspect over another is stressed as well.

Instance-based methods are those which do not provide a global prediction rule for the input-output relations, but build a model each time the output value of a test observation is needed. The term "instance-based learner" is typical of the Machine Learning literature, however it has a close counterpart in the Statistics literature, which is "local learner" or "local method". Generally speaking, instance-based methods provide flexible models that can easily adjust to the underlying data, in case such an adjustment is needed. If there is no or limited need for adjustment, then penalization comes in handy to reduce a flexible solution of an instance-based model to a simpler one, achieving in this case a better bias-variance, or fit-complexity, balance. The effect of penalization is to smooth out the solution of an intrinsically flexible model, achieving is this way a better balance between so-called bias and variance. The term "penalization" is used in both Statistics and Machine Learning, though these fields are not in complete agreement with the accompanying specifics. Penalization methods usually have a basis an intrinsically flexible, and often so-called *unbiased*, model that is subsequently penalized, or downgraded so to say, to a simpler model. In essence, penalization is the act of shrinking the predictions of a model towards each other. Equivalently stated, it is the act of "flattening" or simplifying the prediction function of a certain model. Finally, kernel methods make use of so-called kernels or kernel functions, which allow one to map the data from the original into a higher-dimensional space and to carry out necessary calculations there without explicit knowledge of this higher-dimensional space. The term kernel itself has different meanings, such as a (nonparametric) prediction-function smoother, a (nonparametric) density-estimation smoother and a means to provide an implicit expansion of the space of the inputs. In this thesis, the third meaning is used. The usage of kernels, in combination with an adequate level of penalization, has proved to enhance the performance of linear models enormously. This is mainly due to the possibility to view, so to say, the data from different angles as well as to consider the interactions between the inputs.

Special attention is paid to all of these aspects in the subsequent sections of this introduction. In addition, a good example of a kernel-based penalization technique is briefly presented: the popular Support Vector Machine (SVM) (Vapnik, 1995). This technique is suitable for illustrating the main concepts of kernelization and penalization. SVMs are not an instance-based technique however, unlike SNN, NCH and SH. It should be stressed beforehand that the proposed three novel methods are *not* extensions of SVMs, but rather successfully competing methods. If pushed to the limit, I would rather classify these three methods as conceptual extensions of the famous classification technique *1-Nearest Neighbor*.

A second theme of interest in this thesis, which chronologically comes first, concerns new applications of established penalization techniques. Significant findings on synergies between fields such as Econometrics, Machine Learning, Finance and Marketing have been reported. More precisely, this thesis contributes to an array of successful applications of kernel-based penalization tools in three main ways. In Finance, to begin with, the Support Vector Regression (SVR) technique has been found capable of providing a reliable so-called Value-versus-Growth rotation strategy used commonly in financial time-series forecasting. In marketing, secondly, SVMs have also been found to provide superior accuracy. At the same time, a way to provide a better interpretability of the nonlinear effects of (kernelized) SVMs has been put forward. Third, a prominent example of a synergy between the fields of Econometric, Machine Learning and Marketing is the application of SVRs to the so-called Market Share Attraction Model. SVRs have been found to improve the predictive ability of the Market Share Attraction Model quite substantially.

Finally, the third theme of this thesis concentrates on a new majorization algorithm, which has been developed to solve the optimization problem of the classification technique Support Vector Machines in a relatively fast way.

Main Contributions of the Thesis

As this thesis originates within the field of Econometrics, one of its overall main contributions might be an increased awareness for methods that are typical for Machine Learning. The thesis concentrates on combining research advances coming from these fields, and especially on the so-called instance-based, kernel and penalization methods. Clearly, there is an overlap between the two disciplines in this respect, as pointed out already. It is the combination of research advances put forward in these three general methods that gives rise to the key contributions of the thesis within the area of supervised learning. Concretely, these are the classification methods Soft Nearest Neighbor, Nearest Convex Hull classifier, and Support Hyperplanes. From an instance-based point of view, these methods cover almost the whole spectrum from highly local, flexible models to more conservative, smooth models. The SNN is most local, as it considers distances to points. Next in line is NCH, where the local region is the so-called convex hull of points, and finally, the SH considers an even bigger local region, which is formed by so-called Version Space of (chosen) models. An essential further contribution is that it is not only the size of the local regions that matters, but also the so-called *soft* distance to each of these regions. Each of the proposed classifiers utilizes this soft-distance approach. The SNN method, for example, extends in a soft, continuous way from one nearest neighbor to two and more neighbors, achieving in this way a smooth transition from a small local region to a bigger local region without changing the distance metric or assigning unequal weights to the observations within a certain local region. The NCH technique, on its part, computes soft distances to the convex hulls of observations that belong to different classes. And finally, SH compute the soft distances to the sets of points that are classified correctly by all models from a certain function class. This soft-distance approach is motivated from a bias-variance, or penalization, point of view (mentioned in the tree example in the beginning of the introduction), similarly to the soft margin that arises in the context of Support Vector Machines. This particular application of the penalization concept in the instance-based classification case is novel, and it has by no means been exhaustively presented in this thesis. It has long been known however that increasing the size of the so-called local region leads to smoother and less flexible models. The novelty here is not only the introduction of soft distances, but also their usage together with the so-called kernels, which has proved to enhance the performance of the instance-based models substantially. The kernels themselves have been borrowed from the literature on kernel methods, which does not so vigorously interact with the literature on instance-based methods. However, the combination of kernel and instance-based features is not complete without introducing penalization into the picture in the form of computing soft distances that change (implicitly) the size of the local region on which the classification of a test point is based.

Regarding the application part of the thesis, which is actually chronologically presented first, its main purpose and contribution is to acquaint the reader with existing established kernel-based penalization techniques, which are not instance-based however. In this part, either off-the-shelve econometric methods are reported to be improved using kernel-based penalization techniques, or existing Financial and/or Marketing tasks have been tackled in a superior way. Finally, the contribution of the proposed majorization algorithm lies in its relatively easier interpretability and in many cases in providing quite fast optimization solutions.

The rest of the introduction is organized as follows. First, special attention is paid to the three most relevant aspects of learning in this thesis: the instance-based, the penalization and the kernel aspects, and especially to the interactions between them. As these methods are not discussed in detail in the body of the thesis, the introduction has been used to serve as a selective basic presentation of these methods. Instance-based learning methods are discussed as much as needed for understanding the main concepts, advantages and disadvantages. Most of the attention is focused however on penalization. It is such a basic and powerful idea that it arises actually directly from the so-called loss function[1] that we have in mind and the fact that we have a *finite* data set available for research. A whole section is dedicated to an extensive but still rather simple example that illustrates the main motivation and effect of penalization, and introduces more formally key concepts such as bias and variance. The role of

[1] The loss function, loosely speaking, tells us how much we “lose” if our prediction is off-target by a certain amount.

kernels is discussed two times: once as a way to enhance instance-based learners, and another time more formally as a way to provide implicit decision-surfaces in a so-called higher-dimensional space. Next to the exposition of the three key relevant aspects of the thesis, the classification technique of Support Vector Machines is commented on briefly, with a view to providing a concrete example of the advantages that a (classification) penalization method provides. They are discussed from a rather non-mainstream angle though.

I would like to note that the material in the introduction is presented in a way that is as distant as possible from pre-set frameworks, such a typically Statistical or a Machine Learning framework. Arguably, this makes the introduction relatively hard to read, but in this way it may also be accessible to a larger audience.

1.1 Instance-Based Learning

First of all, it should be mentioned that an "instance" in the Machine Learning literature corresponds to an "observation" in the Econometrics and Statistics literature – it is just an observed input-output pair. Sometimes the output is not known in advance, in which case we speak about a test instance, or a test observation . Instance-based learning is the process of deriving a prediction rule for *each* single test observation separately, instead of deriving a single explicit prediction rule that could be applied to *all* test observations without any further computations. That is, any instance-based (learning) technique builds a new model each time a predicted value of a (single) test observation is needed.

Instance-based methods do not output the entire (predicted) decision surface, but just a small part of it. Actually, this part is the smallest possible: just the single point in space, where we have supplied the test observation. The simplest example of an instance-based model in the regression context is the 1-Nearest Neighbor (1NN) regression. The model here is the following rule: assign a test observation as having the output value of the nearest observation from the training data set. It is clear that this rule is formulated in such a way that it is necessary to *recompute* the rule each time we would like to predict the value of a test observation. Notice that 1NN can immediately be applied to classification tasks as well, where it is actually more popular. The 1NN rule in this case is: classify a test observation as having the class of its nearest neighbor, say in Euclidean distance sense. It is common for the instance-based methods to produce a rule for a test instance based on some area in the space around it. In this sense instance-based methods are also referred to as local methods. The fact that these methods do not output the predicted values for more than one test observation at a time has brought them the nickname "lazy" learners.

Another prominent instance-based learner is the generalized version of the 1NN method, called k-Nearest Neighbor (kNN). Here a training set is stored and each time the classification or regression value of a test observation is needed, the method looks for the k nearest observations according to some distance metric, so that the test observation receives the average value of these, nearest observations from the training data set. Other examples of instance-based learners are distance-weighted kNN, locally weighted regression, and more recently, Support Hyperplanes (SH), Nearest Convex Hull classification (NCH), and Soft Nearest Neighbor (SNN). The last three methods are the subject of Chapters 5, 6, and 7.

The difference between instance-based and (default) non-instance-based models could actually be blurred. Consider, for instance, local linear regression. This method is an instance-based method, as the output value of a test instance is determined only after a "local" region about this instance in considered. The instance-*based* nature here is hidden in the fact that the exact position of the "local" region is dependent on the (position of the) test instance. For the sake of the argument, we can imagine a local region that is so big that it actually covers all observations in the available data set. Thus, the local regression has turned into a non-local (default) one; it is non-instance-based, as the locality is no longer dependent on a particular test instance.

A big advantage of local learners is their relatively bigger adaptability to the observations from the training data set. Therefore, they are able to easily capture complex relations in the data between the inputs and the outputs. This advantage however could also be considered as a big disadvantage in case such complex relations are not inherent in the input-output generating process, since what could be captured/modeled is mostly the noise in the data. We face the usual bias-variance trade-off (discussed more deeply in Section 1.2): whether to allow a model to be more flexible, more unbiased or to artificially reduce its capability of detecting more complex relations, thereby reducing the variance of the estimated (test) output values over repeated samples from the population of input-output pairs. The so-called bias and variance are controlled by means of *penalization*, or *regularization*. A way to explicitly control for such a bias-variance trade-off is implemented in the SH, NCH and SNN methods. Such a control is especially useful when the training data set at hand is not large and/or contains many input variables.

Nonlinear Instance-Based Learning via Kernels

Being local learners, the instance-based methods mentioned so far usually produce implicit nonlinear overall decision rules already in the (original) training-data space. Nevertheless, one can map the data into a higher-dimensional space and carry out all the necessary calculations for a given method in this space. In this way, even more flexible nonlinear (implicit) decision surfaces may appear in the original data space. This mapping is usually called basis expansion, and could be carried out either explicitly, by creating new input variables out of the existing input variables or, sometimes, implicitly by using so-called kernels. The kernels make it possible to make necessary calculations in the higher-dimensional space using only knowledge of the data that is the original, non-expanded space. In case the input-output relations in the data are not very complex, the act of mapping may look like a not-so-reasonable idea. This is of course true if there is no *penalization*, or *regularization*, mechanism that smooths out the overall (implicit) decision surface in the higher-dimensional space, which corresponds to a smoother surface in the original space as well. The only remaining question is: why should we not regularize, or penalize, our method already in the original space so as to produce a smoother overall (implicit) decision surface? This is a more philosophical question and calls for experiments with real data. All our experiments point to the fact that mapping into a higher-dimensional space invariably improves the model performance, provided that a proper regularization parameter is being set (usually via a so-called cross-validation procedure). From an empirical point of view, the question is not whether we should map or not, but rather, what is a good mapping. A similar question has arisen in the early days of Support Vector Machines. It turned out that one of the best mappings is the mapping into an infinite-dimensional space via the so-called Radial Basis Function kernel, introduced in Section 1.3. This mapping allows for any possible flexible decision surface in the original space, with the concomitant fear that much of the noise in the data would be modeled, and not the true relations. However, it turned out that the regularization parameter saves it all: regularization is so powerful that it is responsible for highly non-flexible decision surfaces in the higher-dimensional space, which correspond also to highly non-flexible decision surfaces in the training-data space too. The beneficial effect of mapping lies, it seems, in the model's capability to examine the data from different angles and to capture relations that would normally be hidden in the original space. The combination of this plethora of new mapping possibilities – carried out implicitly via so-called kernel functions – and a proper level of regularization are ultimately responsible for the superior performance of the SH, NCH, and SNN methods when the data is mapped into a higher-dimensional space. Interestingly, the mapping into an infinite-dimensional space that is provided implicitly by the RBF kernel yields the best result for the SH, NCH, and SNN methods, just like the SVM case, though the philosophy and the motivation behind the instance-based group of methods is different.

Advantages of the Proposed Instance-Based Techniques

The Support Hyperplanes, Nearest Convex Hull and Soft Nearest Neighbor methods proposed in this thesis contribute to the debate on how to search for a good penalization technique, and to the literature on kernel methods, as well as support-vector, large-margin, and distance-based learning. All of these techniques should be viewed from a penalization point of view, that is as penalization methods that try to find a balance between training error and model complexity. The underlying reasons behind the beneficial effect of penalization are discussed in Sections 1.2 and 1.2.1.

Some of the advantages and disadvantages of the proposed three new techniques, SH, NCH and SNN, have already been pointed out. Being instance-based techniques, these three methods bear two main features that are common to all methods of this type. First and foremost is the feature that concerns the open debate on how flexible a model should be. As instance-based learners are local methods, they are inherently more flexible and more adaptable as compared to some methods that learn the whole decision surface at once, such as linear regression or SVMs. On the other hand, local learners are better equipped to handle cases where the true relations are rather complex. An advantage of SH, NCH and SNN is that they enjoy the flexibility of being local learners, but also are able to provide smoother relations if necessary, via in-built penalization capability. Empirical tests are in favor of some of these three methods as opposed to SVMs for example, on a variety of data sets. A more extensive view on the relative merits of these three methods vis-a-vis SVMs, as well as among themselves is given in Chapter 5. It is argued that when compared to SVMs, for example, the proposed methods could strike a better balance between bias and variance.

The biggest disadvantage of all instance-based methods from computational point of view is that they are quite slow at test time. This comes quite naturally, as a new model has to be built each time an output value for a test observation is needed. In the case of SH and NCH, quadratic optimization problems have to be solved, which makes them intrinsically slow methods. Their performance on a variety of classification tasks is however quite worthy of boasting, which suggests that they could be used in tasks where achieving higher accuracy takes precedence over speed.

1.2 To Penalize or . . . to Penalize

Incredible as it may seem at first sight, in my view this is one of the most fundamental sections of the whole thesis. Here, the beneficial effect of the notion of penalization in a broad sense is demonstrated in a straightforward and easy-to-grasp way using a tossing-a-coin experiment. In essence, penalization is defined as the act of reducing the values of the estimates (or, predictions) of a given model. In regression estimation, this would-be-penalized model usually provides *unbiased* estimates for the so-called conditional mean. Similarly, in classification this model usually provides unbiased estimates for the most probable class, conditional on a given combination of values for the inputs. Why should we penalize such a model at all? We should penalize it, because the prediction performance of the resulting penalized model on future observations *always* improves, at least in theory.

In what follows, some of the terminology is simplified for ease of exposition of the tossing-a-coin experiment. For example, it is assumed that our *a priori* chosen so-called loss function is the squared-error loss function. That is, the tossing-a-coin experiment is intrinsically a regression estimation task. The squared-error loss function is most appropriate to choose in a regression context, where the output variable takes real values. Nevertheless, in our example these values are just two. Regression estimation is actually the setup in which it is easiest to illustrate the benefits of penalization. In classification tasks, it is more appropriate to use the so-called $0-1$ loss function, for which the bias-variance interplay is more subtle and therefore it is not addressed here, even though classification penalization methods form the spine of the thesis. There exist several types of penalization, such as penalization towards zero, penalization towards the average, and a combination between these two. In this section, only penalization towards zero is discussed extensively.

What's in a Toss of a Coin?

Plenty, if a statistician is tossing the coin. What follows is a simple experimental setup with two known outcomes, the values 1 and 2, which enables us to see through the eyes of a statistician and understand the beneficial effects of penalization. First, we introduce the task, which is to estimate the expected value to be tossed, out of two equally likely outcomes. Then, we mention why we are actually interested in finding the expected value to begin with. It turns out that the answer lies in the so-called loss that we have in mind before we start the experiment, which is in this case the squared-error loss. Interestingly, we will never mention the terms loss or loss function throughout the actual exposition of the tossing-a-coin example, as they are not indispensable terms. We will however talk about a squared discrepancy, or error, between a certain real value[2] z and (future) possible outcome values, here 1 and 2. In essence, an error is the quantitative realization of the (underlying) loss function, when the latter is applied to a concrete estimate and a concrete (future) outcome. Third, having the squared error *a priori* in mind gives rise to a performance measure for any model that estimates the optimal z, where the optimal z is the expected value among all possible output values. This performance measure is the so-called expected mean squared error (EMSE). It is the EMSE that we ultimately want to minimize. The MSE is the mean of the squared errors that a prediction (for the optimal z) of our model will produce when compared to all future output values. If we re-estimate the optimal z by our model infinitely many times, and thus have infinitely many predictions, we will be able to calculate the expected

[2] It turns out that the optimal z value is actually equal to the expected value to be tossed, which in our case is equal to 1.5, as the probability to toss each of 1 and 2 is assumed to be equal to 0.5.

MSE, that is EMSE, as the average MSE. Each time we re-estimate the optimal z we should use a certain fixed amount of randomly-selected data.

The role of penalization here is to reduce the EMSE of a so-called unbiased model. We consider only penalization towards zero, as it makes most sense in the simple task at hand. The overall effect of penalization can be split into two effects on two additive parts *if* – as in our case – we measure the discrepancy between our estimated value for the expected value of the possible outcomes and any (future) possible outcome with the squared error between the two. The two additive parts affected by penalization are usually called "bias" and "variance", and it will become apparent how and why these names arise.

It should be crystal clear that the beneficial effect of penalization is *not* somehow introduced via a change of the loss function, which is here a change of the squared-error discrepancy measure. It is *not* true that when we choose to penalize the estimates of an unbiased model we are somehow changing our squared-error discrepancy (performance) measure implied by the (unchanged) squared-error loss function. On the contrary, penalization is a way to improve an unbiased model's performance in the direction implied by this very squared-error discrepancy measure, that is a decrease of EMSE.

Now we turn to a simple experiment that demonstrates the beneficial effect of penalization. Suppose we have the following experimental setup. We are given a coin with number 1 written on one of its sides, and number 2 written on the other side. These two possible outcomes are known in advance and occur with equal probability. That is, $\Pr(\text{OUTCOME} = 1) = \Pr(\text{OUTCOME} = 2) = 0.5$. Suppose we would like to find such a real value z, the optimal z, that minimizes the mean squared error (MSE), defined here as

$$\begin{aligned}\text{MSE}(z) &\equiv (1-z)^2\Pr(\text{OUTCOME}=1) + (2-z)^2\Pr(\text{OUTCOME}=2)\\ &= (1-z)^2 \times \frac{1}{2} + (2-z)^2 \times \frac{1}{2}.\end{aligned}$$

It is not hard to verify that the MSE is minimized when z equals μ, where

$$\begin{aligned}\mu &= \text{expected value to be tossed}\\ &= 1 \times \Pr(\text{OUTCOME}=1) + 2\Pr(\text{OUTCOME}=2) = \frac{1+2}{2} = 1.5. \quad (1.1)\end{aligned}$$

The corresponding minimal value for the MSE is

$$\begin{aligned}\text{MSE}(\mu) &= \text{MSE}(1.5)\\ &= (1-1.5)^2\Pr(\text{OUTCOME}=1) + (2-1.5)^2\Pr(\text{OUTCOME}=2)\\ &= \frac{(1-1.5)^2}{2} + \frac{(2-1.5)^2}{2} = 0.25.\end{aligned}$$

Our desire to minimize the MSE stems actually from the fact that we have implicitly decided to measure the discrepancy between (the single value) z and

Table 1.1: Calculation of the expected mean squared error (EMSE) of the unbiased model "estimated value for μ, $\hat{\mu}$, equals the number that appears after a single toss" in the top panel and the best possible, perfect-foresight model "estimated value for μ equals 1.5, whatever number has been tossed" in the bottom panel.

outcome after a toss	estimate of μ based on this outcome	(future) possible outcomes	squared error	probability to occur	MSE per estimate of μ	EMSE
1	1	1	0	0.5		
		2	1	0.5		
					0.5	
2	2	1	1	0.5		
		2	0	0.5		
					0.5	
						0.5

outcome after a toss	estimate of μ based on this outcome	(future) possible outcomes	squared error	probability to occur	MSE per estimate of μ	EMSE
1	1.5	1	0.25	0.5		
		2	0.25	0.5		
					0.25	
2	1.5	1	0.25	0.5		
		2	0.25	0.5		
					0.25	
						0.25

any (future) possible outcome with the *squared difference* between the two. Here, possible outcomes are 1 and 2, which occur with equal probability, and possible squared differences are $(1-z)^2$ and $(2-z)^2$. Note that μ is a so-called population parameter – it is the expected value of our outcome variable "number tossed". The z value need not *a priori* be equal to a population parameter value, but in this case it is.

Suppose that we would like to find μ, but are only allowed to estimate it on the basis of just *one* toss of the coin, and not using the set of all possible outcomes as we did in (1.1). Our model for estimating μ could be:

default model for μ:
"the estimated value of μ, $\hat{\mu}$, equals the number tossed". (1.2)

It is important to remember that the set of possible outcomes, in this experiment $\{1, 2\}$, is known in advance. Therefore, it is possible to compute the MSE if number 1 has been tossed:

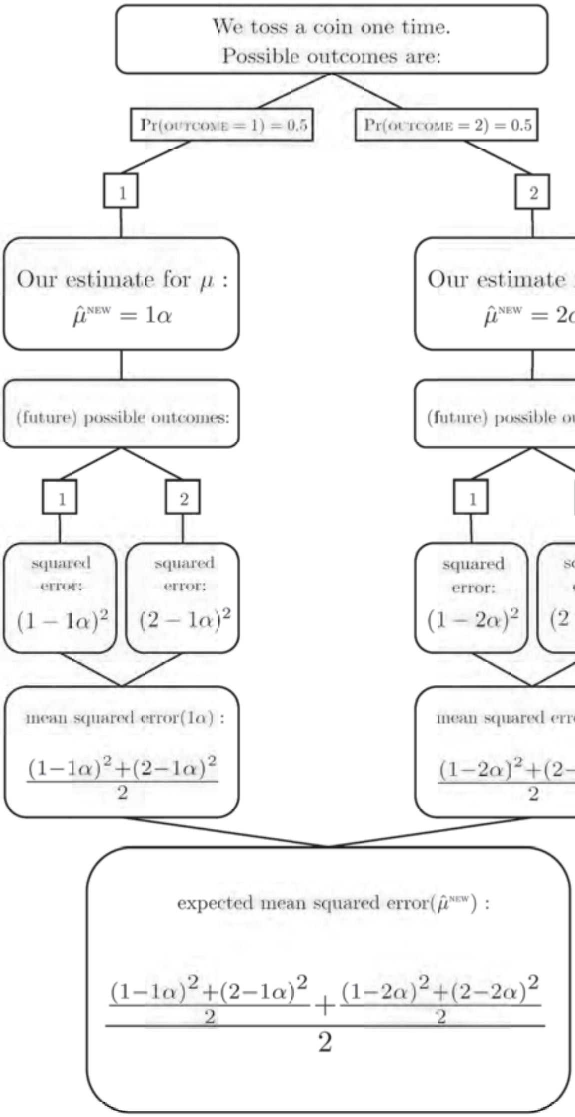

We toss a coin one time.
Possible outcomes are:
Pr(OUTCOME = 1) = 0.5
Pr(OUTCOME = 2) = 0.5
1
2
Our estimate for μ :
$\hat{\mu}^{\text{NEW}} = 1\alpha$
Our estimate for μ :
$\hat{\mu}^{\text{NEW}} = 2\alpha$
(future) possible outcomes:
(future) possible outcomes:
1
2
1
2
squared error:
$(1 - 1\alpha)^2$
squared error:
$(2 - 1\alpha)^2$
squared error:
$(1 - 2\alpha)^2$
squared error:
$(2 - 2\alpha)^2$
mean squared error(1α) :
$\frac{(1-1\alpha)^2+(2-1\alpha)^2}{2}$
mean squared error(2α) :
$\frac{(1-2\alpha)^2+(2-2\alpha)^2}{2}$
expected mean squared error($\hat{\mu}^{\text{NEW}}$) :
$\frac{\frac{(1-1\alpha)^2+(2-1\alpha)^2}{2}+\frac{(1-2\alpha)^2+(2-2\alpha)^2}{2}}{2}$

Figure 1.1: Calculation of the expected mean squared error (EMSE) of the following model for estimating the expected value μ to be tossed: "estimated value for μ, $\hat{\mu}^{\text{NEW}}$, equals the number that appears after a single toss, called an outcome, multiplied by α". Possible outcomes are 1 and 2, which occur with equal probability. Thus, the expected value to be tossed, μ, equals 1.5. An error is defined as the difference between a (future) possible outcome and a concrete value for $\hat{\mu}^{\text{NEW}}$, such as $1 \times \alpha$ or 2α.

Table 1.2: Calculation of the expected mean squared error (EMSE) of the following model for estimating the expected value μ to be tossed: "estimated value for μ, $\hat{\mu}^{\text{NEW}}$, equals the number that appears after a single toss, called an outcome, multiplied by α". Here, $\alpha = 0.9$. Possible outcomes are 1 and 2, which occur with equal probability. Thus, the expected value to be tossed, μ, equals 1.5. An error is defined as the difference between a (future) possible outcome and a concrete value for $\hat{\mu}^{\text{NEW}}$, such as $1 \times \alpha$ or 2α. The mean squared error (MSE) per concrete estimate is defined as the expected value of the squared error, which is equal to the average of two errors in our case. The EMSE is defined as the expected value of the MSE per estimate, and is equal to the average of two values for the MSE in our case.

outcome after a toss	estimate of μ based on this outcome	(future) possible outcomes	squared error	probability to occur	MSE per estimate of μ	EMSE
1	0.9	1	0.01	0.5		
		2	1.21	0.5		
					0.61	
2	1.8	1	0.64	0.5		
		2	0.04	0.5		
					0.34	
						0.475

$$\begin{aligned}
\text{MSE}(\hat{\mu} = 1) \quad &= (1 - \hat{\mu})^2 \text{Pr}(\textsc{outcome} = 1) + (2 - \hat{\mu})^2 \text{Pr}(\textsc{outcome} = 2) \\
&= (1 - 1)^2 \text{Pr}(\textsc{outcome} = 1) + (2 - 1)^2 \text{Pr}(\textsc{outcome} = 2) \\
&= \frac{(1-1)^2}{2} + \frac{(2-1)^2}{2} = 0.5.
\end{aligned}$$

If, on the other hand, number 2 has been tossed, then the MSE is

$$\begin{aligned}
\text{MSE}(\hat{\mu} = 2) \quad &= (1 - \hat{\mu})^2 \text{Pr}(\textsc{outcome} = 1) + (2 - \hat{\mu})^2 \text{Pr}(\textsc{outcome} = 2) \\
&= (1 - 2)^2 \text{Pr}(\textsc{outcome} = 1) + (2 - 2)^2 \text{Pr}(\textsc{outcome} = 2) \\
&= \frac{(1-2)^2}{2} + \frac{(2-2)^2}{2} = 0.5.
\end{aligned}$$

Since numbers 1 and 2 appear with equal probability, it is equally probable that $\hat{\mu}$ takes each of these values. However, it is not known in advance which number would be tossed. Therefore, we can compute the expected MSE (EMSE) of our estimation procedure for μ as

$$\begin{aligned}
\text{EMSE}(\hat{\mu}) \quad &= \text{MSE}(1)\text{Pr}(\hat{\mu} = 1) + \text{MSE}(2)\text{Pr}(\hat{\mu} = 2) \\
&= 0.5\text{Pr}(\hat{\mu} = 1) + 0.5\text{Pr}(\hat{\mu} = 2) \\
&= 0.5\text{Pr}(\textsc{outcome} = 1) + 0.5\text{Pr}(\textsc{outcome} = 2) = 0.5. \qquad (1.3)
\end{aligned}$$

This derivation is also shown in Table 1.1. Thus, the EMSE *of our model* is 0.5. Note that the EMSE model-performance measure is implied by the squared-difference discrepancy measure between a single predicted number, like z and $\hat{\mu}$, and any (future) possible outcome. The *minimal* possible EMSE is achieved when $\hat{\mu}$ and μ coincide, whatever number has been tossed, and is non-coincidentally equal to the minimal value of the MSE, 0.25:

$$\begin{aligned}\text{minimal EMSE}(\hat{\mu}) &= \text{MSE}(1.5)\Pr(\hat{\mu}=1)+\text{MSE}(1.5)\Pr(\hat{\mu}=2)\\ &= \text{MSE}(1.5)\left[\Pr(\hat{\mu}=1)+\Pr(\hat{\mu}=2)\right]=\text{MSE}(1.5)\\ &= \text{MSE}(\mu)=0.25. \qquad (1.4)\end{aligned}$$

This derivation is also shown in Table 1.1. Thus, it holds that *minimal* $\text{EMSE}(\hat{\mu}) = \text{EMSE}(\mu) = \text{MSE}(\mu)$. Can we improve the EMSE of our model, computed in (1.3), towards the minimal EMSE, computed in (1.4)? Can we come up with a new model that improves the EMSE of our old model, where our new model is also based on one toss only? Say, our new model is

> our new model for μ:
>
> "the estimated value for μ, $\hat{\mu}^{\text{NEW}}$, equals α times the number tossed",

where the optimal α is 0.9, as will become clear in a moment. In other words, the estimates of the new model are equal to the estimates of the default model times α, implying: $\hat{\mu}^{\text{NEW}} = \alpha\hat{\mu}$. We can compute the EMSE of the new model (using the value of 0.9 for α):

$$\begin{aligned}\text{EMSE}\left(\hat{\mu}^{\text{NEW}}\right) &= \text{MSE}\left(1\times\alpha\right)\Pr(\hat{\mu}^{\text{NEW}}=1\times\alpha)+\text{MSE}\left(2\alpha\right)\Pr(\hat{\mu}^{\text{NEW}}=2\alpha)\\ &= \text{MSE}\left(1\times\alpha\right)\Pr(\textsc{outcome}=1)+\text{MSE}\left(2\alpha\right)\Pr(\textsc{outcome}=2)\\ &= \left[\frac{(1-\alpha)^2}{2}+\frac{(2-\alpha)^2}{2}\right]\frac{1}{2}+\left[\frac{(1-2\alpha)^2}{2}+\frac{(2-2\alpha)^2}{2}\right]\frac{1}{2}\\ &= \frac{0.61}{2}+\frac{0.34}{2}=0.475. \qquad (1.5)\end{aligned}$$

The derivation of the EMSE $(\hat{\mu}^{\text{NEW}})$ is also illustrated in Figure 1.1 and Table 1.2. Note that the EMSE of the default model can be derived from the figure and the table when α is set to 1. The EMSE of the hypothetical model that achieves the minimal EMSE, that is the model "the estimate of μ equals μ whatever number has appeared after the toss", can likewise be traced. Note also that the column "(future) possible outcomes" of Table 1.2 contains two values. These values are all possible values that could occur, and each of these values occurs with equal probability. Alternatively, if we do not know how many – but still, a finite number of – values could occur and what is the probability of each one to occur, we have to consider not two, but *infinitely* many outcomes, where each outcome is a result of a random toss (that is, the outcomes should not influence each others' probability of occurrence). From these infinitely many outcomes

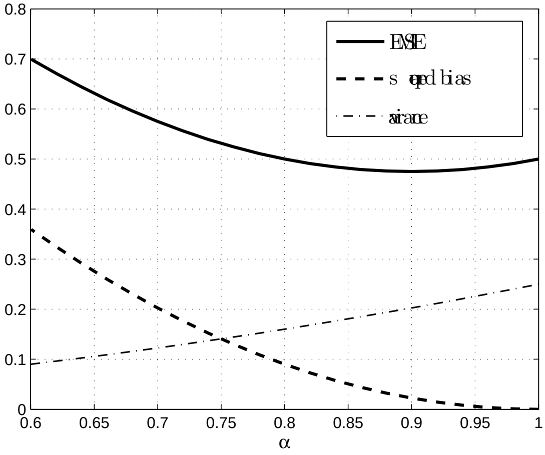


Figure 1.2: The expected mean squared error (EMSE) of the model "the estimated value for μ, $\hat{\mu}^{\text{NEW}}$, equals α times the number tossed" for different values of α, as computed by (1.5) for non-fixed λ. The minimal value for EMSE is achieved for $\alpha = 0.9$. For $\alpha = 1$ the estimator for μ, $\hat{\mu}^{\text{NEW}}$, is unbiased. Notice the interplay between the two major components of EMSE($\hat{\mu}^{\text{NEW}}$): the squared bias $\left[\mathrm{E}\left(\hat{\mu}^{\text{NEW}}\right) - \mu\right]^2$ (the downward sloping dashed curve) and the variance $\mathrm{Var}\left(\hat{\mu}^{\text{NEW}}\right)$ (the upward sloping dash-dotted curve). As long as $\alpha \in (0.8, 1)$ the EMSE is lower than the EMSE of the unbiased estimator. See (1.8) for the exact effect of the squared bias and the variance on EMSE in the squared-error discrepancy case.

we can then "compute" the probability of occurrence of each possible outcome and proceed as we have done so far.

So, the EMSE of our new model, 0.475, is better than the EMSE of the old model, 0.5, and we still base our estimate for μ on one toss only. The optimal value of α is determined from (1.5), which (more generally) is minimized for

$$\alpha^* = \frac{\mu^2}{\mu^2 + \mathrm{Var}(\hat{\mu})} = \frac{1.5^2}{1.5^2 + 0.25} = 0.9, \tag{1.6}$$

where μ is the expected value to be tossed, computed in (1.1), and $\mathrm{Var}(\hat{\mu})$ is the variance of $\hat{\mu}$, where $\hat{\mu}$ takes values as postulated by the default model (1.2). As the values that $\hat{\mu}$ could take are 1 and 2, which occur equally likely, $\mathrm{Var}(\hat{\mu})$ is computed as:

$$\begin{aligned}\mathrm{Var}(\hat{\mu}) &= \mathrm{E}(\hat{\mu} - \mathrm{E}(\hat{\mu}))^2 = \mathrm{E}(\hat{\mu} - [1\mathrm{Pr}(\hat{\mu} = 1) + 2\mathrm{Pr}(\hat{\mu} = 2)])^2 = \mathrm{E}(\hat{\mu} - 1.5)^2 \\ &= (1 - 1.5)^2\mathrm{Pr}(\hat{\mu} = 1) + (2 - 1.5)^2\mathrm{Pr}(\hat{\mu} = 2) = \frac{(1 - 1.5)^2}{2} + \frac{(2 - 1.5)^2}{2} \\ &= 0.25. \end{aligned} \tag{1.7}$$

where the E operator denotes expected value. There are several important points to be stressed here:

- Our default estimator for μ, $\hat{\mu}$, is “unbiased”, that is, the expected value of $\hat{\mu}$ is equal to μ. In our case this holds as $\mathrm{E}(\hat{\mu}) = 1 \times \mathrm{Pr}(\hat{\mu} = 1) + 2\mathrm{Pr}(\hat{\mu} = 2) = 1 \times \mathrm{Pr}(\textsc{outcome} = 1) + 2\mathrm{Pr}(\textsc{outcome} = 2) = \mu = 1.5$.

- The optimal α, with which to multiply the possible estimates $\{1, 2\}$ of the unbiased estimator $\hat{\mu}$, can be computed only if the expected value μ is known. This is unrealistic, as μ is actually the value we are looking for. Having pointed that out, consider the next note:

- The optimal α is always between 0 and 1, that is, $\alpha^* \in [0, 1]$. The optimal α would be 1 in the trivial case where $\mathrm{Var}(\hat{\mu}) = 0$.

- The optimal α is equal to 0 when $\mu = 0$, whatever the variance of an unbiased estimator for μ, such as $\mathrm{Var}(\hat{\mu})$ here, see (1.6). In this case, the penalized model estimates μ correctly as being equal to 0, whatever the training data.

Regarding the third item, the $\mathrm{Var}(\hat{\mu}) = 0$ condition can never be fulfilled in practice, as we base our model for μ on imperfect knowledge about the problem setting or, so to say, on a finite data set (of one number that has appeared after the toss). This finite data set can contain in general different values; in our case, there are two possible data sets containing one tossed number each. The main conclusion here is that even though the optimal α cannot be computed exactly, it is never in practice optimal to rely on the unbiased model, which corresponds to $\alpha = 1$, as long as the unbiased model's estimates of $\hat{\mu}$ vary over different (selected at random) training data sets. Therefore, it would be in most, if not all, of the practical cases beneficial to set α below 1. In the case at hand, it can be shown that if a model uses α in the range $(0.8, 1)$, then the EMSE of this model would be smaller than the EMSE of the unbiased model, computed in (1.3). This is illustrated in Figure 1.2.

It might appear at first sight that we have improved the EMSE of the unbiased model somehow artificially, that is, without using any additional information other than the tossed number. In fact, if we only know the tossed number, then indeed we cannot improve the EMSE. However, we *do* have additional knowledge, which is that *the output variable has some variance* by definition of the problem. That is, the output variable is a *random* variable and as such could take different values. The unbiased model never uses this extra information. The penalized model, on the other hand, *does*. The higher the variance of the output variable and the smaller the amount of data from which to build a model – which results in a higher variance for the unbiased estimator of μ, such as $\text{Var}(\hat{\mu})$ – the stronger the case for penalization. Thus, the penalized model amends an information deficiency inherent in the unbiased model. As the variance of an unbiased estimator, such as $\text{Var}(\hat{\mu})$, is not known a priori, it is not possible to provide a priori the most optimal penalization level. What is known for sure however is that as $\text{Var}(\hat{\mu}) > 0$ in practice, a penalization level of zero, here in the form $\alpha = 1$, is intrinsically suboptimal.

As a concluding remark, it should be mentioned that the act of multiplying the estimates $\hat{\mu}$ of an unbiased model with $\alpha \in [0, 1]$, where each estimate is computed from a finite data set, is called *penalization towards zero*.

It is handy at this stage to clarify explicitly the differences between three terms that have appeared so far: a model, an estimator, and an estimate:

- a model is a *relation*, for example: "$\hat{\mu} =$ number tossed";
- an estimator is a *random variable*, for example: $\hat{\mu}$;
- an estimate is a concrete *value* that an estimator may take, for example: $\hat{\mu} = 1$.

The estimator is a random variable as it can take different values, called estimates, depending on the outcome after a toss. Note that a model is *not* a random variable as it is a qualitative statement in nature, a description of how reality works. Thus, strictly speaking it is incorrect to say that "a model is unbiased". Instead, one should say "the estimator $\hat{\mu}$ (provided by a model) is unbiased", since unbiasness is a property of random variables only. Sometimes we use the term "a model is unbiased", but only as a shorthand notation for "the estimator $\hat{\mu}$ is unbiased". In the same line of thought, the estimates by themselves cannot be unbiased, as they are just concrete values, and not random variables. Finally, we should mention that the random variable $\hat{\mu}$ is referred to as an *estimator*, as this random variable tries to capture a *population parameter*, here μ.

The Bias-Variance Trade-Off

The act of penalization towards zero of the (default) unbiased model creates a bias of the new model. This bias (in absolute terms) is equal to

$$|\text{E}(\hat{\mu}^{\text{NEW}}) - \mu| = |\text{E}(\alpha^*\hat{\mu}) - \mu| = |\alpha^*\text{E}(\hat{\mu}) - \mu| = |\alpha^*\mu - \mu| = |1.35 - 1.5| = 0.15,$$

where $\alpha^* = 0.9$ (see 1.6). Of course, when $\alpha = 1$ we recover the default model, for which the bias is equal to 0. Just for the sake of the argument for the moment, we can also compute the variance of the new model as

$$\mathrm{Var}(\hat{\mu}^{\text{NEW}}) = \mathrm{Var}(\alpha^* \hat{\mu}) = (\alpha^*)^2 \, \mathrm{Var}(\hat{\mu}) = 0.9^2 \times 0.25 = 0.2025,$$

where $\mathrm{Var}(\hat{\mu})$ has been computed in (1.7). In sum, the act of penalization (towards zero) of an unbiased estimator for μ has two emblematic effects:

- it creates a bias of the new, penalized estimator for μ, that is, $|\mathrm{E}(\hat{\mu}^{\text{NEW}}) - \mu| > |\mathrm{E}(\hat{\mu}) - \mu| = 0$. Simultaneously with this however,
- it reduces the variance of the new estimator vis-a-vis the unbiased estimator, that is, $\mathrm{Var}(\hat{\mu}^{\text{NEW}}) < \mathrm{Var}(\hat{\mu})$.

Therefore, we can say that penalization brings with itself two antagonistic effects, as illustrated in Figure 1.2. One is the negative effect of creating a biased estimator. The absolute value of the bias in our case has been computed as 0.15. In words, it means that if we re-estimate μ by $\hat{\mu}^{\text{NEW}}$ using all possible tosses (here, 1 and 2), then on average we will be off-target, or off-μ, by 0.15. Let us verify this. If the number tossed is 1, then we are off-target by $|1 - 0.9 \times 1| = 0.1$. And, if the number tossed is 2, then we are off-target by $|2 - 0.9 \times 2| = 0.2$. As the probability to toss either 1 or 2 is equal to 0.5, on average we are indeed off-target by $(0.1 + 0.2)/2 = 0.15$. The second effect of penalization is a desirable one: the variance of the penalized (biased) estimator, $\mathrm{Var}(\hat{\mu}^{\text{NEW}})$, is lower than the variance of the unbiased estimator, $\mathrm{Var}(\hat{\mu})$. In words, it means that if we re-estimate μ by $\hat{\mu}^{\text{NEW}}$ using all possible tosses (here, 1 and 2), then the corresponding estimates from this model will be *closer* to each other than the estimates of the unbiased model. Let us verify this. If the number tossed is 1, then the estimate for μ is α^*. And, if the number tossed is 2, then the estimate for μ is $2\alpha^*$. In our case $\alpha^* = 0.9$, therefore the possible estimates are 0.9 and 1.8. As the probability to toss either 1 or 2 is equal to 0.5, the variance of the set of values $\{0.9, 1.8\}$ is, as already shown, 0.2025. The corresponding estimates for μ of the unbiased model form the set of values $\{1, 2\}$. The variance of these values is 0.25. It is no wonder that this variance is bigger, as 1 and 2 are further away from each other than 0.9 and 1.8 are. Thus, the biased estimator brings us the comfort that even if we had other data at our disposal (here, another realized number after a toss), our conclusions, or estimates, about μ will not be too far apart from each other.

This interplay between bias and variance is referred to as the bias-variance trade-off. Clearly, the biased model $\hat{\mu}^{\text{NEW}}$ outperforms the unbiased one, $\hat{\mu}$, in terms of EMSE by $\mathrm{EMSE}(\hat{\mu}) - \mathrm{EMSE}(\hat{\mu}^{\text{NEW}}) = 0.5 - 0.475 = 0.025$. This means that the negative effect of increased biased is more than compensated by the positive effect of reduced variance. The exact role that bias and variance play for the EMSE can be best illustrated from the following famous decomposition of EMSE:

$$\mathrm{EMSE}\left(\hat{\mu}^{\text{NEW}}\right) = \mathrm{Var}(Y) + \left[\mathrm{E}\left(\hat{\mu}^{\text{NEW}}\right) - \mu\right]^2 + \mathrm{Var}\left(\hat{\mu}^{\text{NEW}}\right). \tag{1.8}$$

For convenience, a new variable Y has been introduced, which is the random variable "number that appears after a toss", or "number tossed" for short. Thus, Y is just the *output variable* of interest in our setting. It can take only two values, 1 and 2, which appear with equal probability. The variance of Y is therefore

$$\mathrm{E}(Y-\mathrm{E}(Y))^2 = \mathrm{E}(Y-1.5)^2 = (1-1.5)^2\Pr(Y=1)+(2-1.5)^2\Pr(Y=2) = 0.25,$$

which is the same as the variance of our default model, as it can take the same values with the same (equal) probabilities. Notice that $\mathrm{Var}(Y)$ cannot be influenced by any model: it just exists even before we start thinking about a model. Thus, $\mathrm{Var}(Y)$ is the so-called *irreducible* part of EMSE of any model, or the irreducible error for short. The other two terms fall under our influence, however. The second term is the squared bias $\left[\mathrm{E}\left(\hat{\mu}^{\text{NEW}}\right)-\mu\right]^2$ between μ and our estimator of it, $\hat{\mu}^{\text{NEW}}$, which we denote as $\text{bias}^2(\mu,\hat{\mu}^{\text{NEW}})$. The last term is the variance of our estimator for μ, $\mathrm{Var}\left(\hat{\mu}^{\text{NEW}}\right)$, which was already discussed. In sum, the EMSE of the estimator $\hat{\mu}^{\text{NEW}}$ for μ from the model $\hat{\mu}^{\text{NEW}}$ = "number tossed times α" can be decomposed in general into three parts:

- *irreducible error*, here: $\mathrm{Var}(Y)$. This is the variance of the random variable whose expected value we are trying to estimate;
- *squared bias* between μ and our estimator for μ, here: $\left[\mathrm{E}\left(\hat{\mu}^{\text{NEW}}\right)-\mu\right]^2$;
- *variance of our estimator* for μ, here: $\mathrm{Var}\left(\hat{\mu}^{\text{NEW}}\right)$.

The effects of penalization-towards-zero of an unbiased estimator for μ appear clearer now. First, there is no effect whatsoever on the irreducible error. Second, the bias increases (from zero), meaning that the squared-bias term increases. This worsens the EMSE with respect to that of the unbiased estimator. However, the variance of the penalized estimator (here, $\mathrm{Var}\left(\hat{\mu}^{\text{NEW}}\right)$) decreases vis-a-vis the variance of the unbiased estimator (here, $\mathrm{Var}\left(\hat{\mu}\right)$). This, in its turn, improves the EMSE with respect to that of the unbiased estimator. The combined effect of penalization is a decrease in EMSE with respect to that of the unbiased estimator. In order to find the optimal level of penalization, the true value of μ has to be known, which is unrealistic, as this is the scalar value we are actually looking for to begin with. However, it is *not necessary* to know the optimal level of penalization in order to enjoy the effect of a decrease in EMSE. This happens since even a slight departure from the unbiased model, that is, even a slight decrease of the estimates of the unbiased estimator for μ from $\hat{\mu}$ to $\alpha\hat{\mu}$ where α is slightly below 1, is already bound to improve the EMSE of the unbiased model, as can be clearly seen in Figure 1.2. The unlikely case from a practical point of view when improvement of the EMSE of the unbiased model is impossible – that is, when $\mathrm{Var}\left(\hat{\mu}\right)=0$ – has already been addressed in Section 1.2. In this case, the EMSE equals the irreducible error, $\mathrm{Var}\left(Y\right)$, and the optimal value of the regularization parameter α is 1. The variance of the unbiased estimator, Var $(\hat{\mu})$, would be zero, for example, when we re-estimate $\hat{\mu}$ from a number of infinite series of (random) tosses. As $\hat{\mu}$ would be computed as 1.5 each time, there would be no variance among the estimates. This occurs since we (re-)estimate the population parameter of interest, here μ, from a series that contains *infinitely* many random realizations of the output variable Y. Notice that in this case, the optimal regularization parameter α would be computed as being equal to 1, implying that no regularization is needed.

1.2.1 SVMs for Classification as a Penalization Method

Up till now we have considered an example where the squared-error discrepancy measure was used as our way to assess the discrepancy between a predicted value and an output value. This measure is referred to in the literature as the squared-error loss. It has turned out that the expected-error loss implies the minimization of the Expected Mean Squared Error (EMSE) model-performance criterion, which is minimized when our prediction is the mean, or the expected value. In case the output variable is observed over many possible combinations of values for the input variables, we say that the EMSE is minimized when our prediction is the *conditional* mean, that is the mean at a particular combination of values for the inputs. In this case we are ultimately interested in minimizing the EMSE over all such possible combinations, or the minimization of the *overall* or expected EMSE. The most prominent conclusion we have arrived at is that a model that minimizes the error over the training set (in the example of the previous section, of just one toss) that is an unbiased model, is not the best prediction model that could be derived from our training set. A better model would be one that penalizes, or shrinks, the prediction for each conditional mean. Note that in the tossing-a-coin example there is just one conditional mean. This penalized model achieves a lower EMSE even though it somehow counter-intuitively does *not* minimize the empirical counterpart of EMSE, which is the sum of (squared) errors over the training data set.

A similar phenomenon occurs also in classification, where it is common to use the so-called $0-1$ loss function rather than the squared-error loss. In this case, if an observation from a given class is misclassified, the loss is 1, otherwise it is 0. For the $0-1$ loss function the expected loss cannot be decomposed into additive bias and variance terms in the same way as for the squared-error loss. Nevertheless, the bias and variance enter in a nonlinear fashion in the expression for EMSE and play a similar role. That is, both bias and variance should be reduced[3] to achieve good predictive performance. Having the $0-1$ loss in mind implies that we should maximize the *expected accuracy* in much the same way as the squared-error loss implies that we should minimize the expected

[3] It can be shown however (see Friedman, 1996; Hastie, Tibshirani, & Friedman, 2001) that it is not always optimal to trade-off bias and variance when the 0-1 loss function is used.

MSE (EMSE) criterion. We note that maximizing the expected accuracy can equivalently be stated as minimizing the *expected misclassification rate*, as the accuracy is equal to one minus the misclassification rate.

Below I consider a concrete binary classification technique, called Support Vector Machines (SVMs) (Vapnik, 1995), viewed from a rather non-mainstream angle. This technique is a penalization technique as the estimates are penalized. It is quite suitable for representing the bias-variance interplay in classification tasks and the need for penalization of the estimates in this setting. Actually, the estimates are not penalized directly, as in the tossing-a-coin example, but indirectly via shrinking of the model coefficients. Without loss of generality, we consider here linear SVMs only, and postpone the discussion of the nonlinear SVMs for the next section.

Let us consider the binary classification setup. Here, we have a data set $D = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$, where $\mathbf{x}_i \in \mathbb{R}^p$ and $y_i \in \{-1, +1\}$. That is, we have a data set of N observations for p inputs and corresponding class labels. The fact that the two possible labels of the binary output variable y are set to the values -1 and 1 and not to any other two different values is not crucial. SVMs are a classification method that, given data D, builds a model for predicting future binary outcomes y for future (or, test) values for the inputs $\mathbf{x}$. The minimization, or objective, function of SVMs for binary classification consists of two terms. One of the terms minimizes the sum of *training errors*. The second term penalizes the model coefficients towards zero, which is directly responsible for shrinking the predicted (raw) y values from D. This penalization term has a geometrical representation and is referred to as *the margin*.

At the outset, let us first focus on how the SVM model views the discrepancy between an observed y value and a predicted (raw) y value. This discrepancy is called a (training) error for short. A crucial feature of this error is that if an observation from class 1 receives a predicted value more than or equal to 1, then no error is associated with this predicted value. Similarly, if an observation from class -1 receives a predicted value less than or equal to -1, then no error is associated with this predicted value. Actually, the SVM error for class 1 is defined as $[1 - \hat{y}_i]_+$, where $[Z]_+ = \max(0, Z)$, and the error for class -1 is defined as $[1 + \hat{y}_i]_+$, where $\hat{y}_i$ is the predicted functional value for observation y_i. The predicted label for observation y_i is equal to $\text{sign}(\hat{y}_i)$. In short, the SVM error is defined as $[1 - y_i\hat{y}_i]_+$. For comparison, the squared error in this case is $[1 - y_i\hat{y}_i]^2 = [y_i - \hat{y}_i]^2$, which would be applicable in case we have a squared-error loss in mind instead of a $0 - 1$ loss.

In linear SVMs it is assumed that the predicted raw output values are a linear combination of the inputs for any $i = 1, 2, \dots, N$:

$$\hat{y}_i = \beta_0 + \boldsymbol{\beta}'\mathbf{x}_i,$$

where $\boldsymbol{\beta}$ is a $p \times 1$ vector of coefficients. The final predicted output values are subsequently given as $\text{sign}(\hat{y}_i)$. Nonlinear formulations will be discussed in the next section. In SVMs, a restriction on the overall magnitude of the $\boldsymbol{\beta}$ coefficients is imposed. Concretely, the sum $\boldsymbol{\beta}'\boldsymbol{\beta}$ is not allowed to be large, where the researcher can choose how much "large" is. The magnitude of the sum $\boldsymbol{\beta}'\boldsymbol{\beta}$ is controlled by means of a penalization parameter λ. The role of λ is to shrink the $\boldsymbol{\beta}$ coefficients towards zero, but *not* the β_0 one. Shrinking these coefficients directly translates into shrinking the predicted output values $\hat{y}_i$ (towards β_0). And reducing the predicted output values is ultimately what regularization is all about.

The objective function that SVMs minimize takes the form (see, e.g., Hastie et al., 2001, p. 380):

$$\begin{aligned} &\min_{\beta_0,\boldsymbol{\beta}} \sum_{i=1}^{N}[1 - y_i\hat{y}_i]_+ + \lambda\boldsymbol{\beta}'\boldsymbol{\beta}, \\ &\text{subject to} \quad \hat{y}_i = \beta_0 + \boldsymbol{\beta}'\mathbf{x}_i. \end{aligned} \tag{1.9}$$

The λ parameter is nonnegative and is set manually. Usually this SVM optimization problem is reformulated equivalently as:

$$\begin{aligned} &\min_{\beta_0,\boldsymbol{\beta},\boldsymbol{\xi}} C\sum_{i=1}^{N}\xi_i + \frac{1}{2}\boldsymbol{\beta}'\boldsymbol{\beta}, \\ &\text{subject to} \quad \xi_i \geq 0,\ y_i(\beta_0 + \boldsymbol{\beta}'\mathbf{x}_i) \geq 1 - \xi_i,\ i = 1, 2, \ldots N \end{aligned} \tag{1.10}$$

where the introduced ξ_i's are called slack variables and $C = 1/2\lambda$. Note that $\sum_{i=1}^{N}\xi_i = \sum_{i=1}^{N}[1 - y_i\hat{y}_i]_+$.

Let us pause for a moment and comment on the formulation of the SVM optimization problem. Three terms have appeared, each of which has a distinct meaning and usage here: "error", "loss (function)" and "objective function". Concretely,

- a (training) "error" is a real value. It is the quantitative way in which a *model* measures the discrepancy between an observed value and a predicted value. Sometimes the error in this meaning is called a loss, but we refrain from this usage here to avoid confusion. For SVMs the error is $[1 - y_i\hat{y}_i]_+$ for observed y_i and corresponding prediction $\hat{y}_i$.
- a "loss" is also a value. It is the quantitative way in which *we*, as researchers, measure the discrepancy between an observed value and a predicted value. In the classification case this is usually the $0 - 1$ loss, which means that we measure the performance of a model in terms of its *accuracy.* This comes to stress that we do not access a model based on its objective function value (which it minimizes). Using the current notation, the loss that we associate with a y_i and a prediction $\hat{y}_i$ is $[0 - y_i\text{sign}(\hat{y}_i)]_+$, which is a rather roundabout way to say that if $\text{sign}(\hat{y}_i) = y_i$, then our loss

for observation i is 0, and if $\text{sign}(\hat{y_i}) \neq y_i$, then our loss for observation i is 1.

- an "objective function" is the function, such as (1.9), that a model minimizes so as to produce a rule, such as (1.9), that will make sure that *test* observations y_j, $j = 1, 2, \ldots, J$, $J \to \infty$, are predicted with great *accuracy*, where the accuracy rate is computed as $\lim_{J\to\infty}(1/J)\sum_{j=1}^{J}[y_j\text{sign}(\hat{y_j})]_+$. That is, the minimization of the objective function aims at reducing the overall *loss* over the whole data space, which in case of $0-1$ loss is equal the overall misclassification rate. The overall misclassification rate is computed as one minus the accuracy rate, or $1 - \lim_{J\to\infty}(1/J)\sum_{j=1}^{J}[y_j\text{sign}(\hat{y_j})]_+ = \lim_{J\to\infty}(1/J)\sum_{j=1}^{J}[-y_j\text{sign}(\hat{y_j})]_+ = \lim_{J\to\infty}(1/J)\sum_{j=1}^{J}[0 - y_j\text{sign}(\hat{y_j})]_+$. The last term may seem artificial, but it is handy for comparison with the average *training* error in SVMs, which is $(1/N)\sum_{i=1}^{N}[1 - y_i\hat{y_i}]_+ = (1/N)\sum_{i=1}^{N}\xi_i$.

Notice that in order to achieve high accuracy on test data, the SVM model does *not* minimize the misclassification rate on the training data, which is $(1/N)\sum_{i=1}^{N}[0 - y_i\hat{y_i}]_+$, but minimizes a different term (actually, the sum of two terms), called objective function. The reason for this can be traced back to the tossing-a-coin example of Section 1.2. From this example it is clear that we should not actually be interested in having a zero sum of *errors* on the training set. Ultimately, we want to minimize the expected *loss* over future tosses. The quadratic squared-error loss leads to the goal of achieving a minimal expected MSE (EMSE), whereas having a $0-1$ loss in mind leads to the maximization of the *expected accuracy* criterion. That is, using the $0-1$ loss translates into a desire to maximize the accuracy that a model achieves over future observations. In the tossing-a-coin experiment, we were not keen on predicting the mean value to be tossed as the value that appeared on one (single training) toss, in which case the (training) error would of course be zero. Two forces were playing their roles in moving us away from this zero-error solution: the so-called bias and variance. The same forces play a similar role when the $0-1$ loss is used instead of the squared-error loss. Roughly speaking, the bias in case of SVMs has to do with the term $\sum_{i=1}^{N}\xi_i$, and the variance has to do with the term $\boldsymbol{\beta}'\boldsymbol{\beta}$. The manually-adjustable C parameter sets the relative preference between the two terms. Notice that the *unbiased* classification model would be minimizing just the training misclassification rate (times N):

$$\sum_{i=1}^{N}[0 - y_i\text{sign}(\hat{y_i})]_+.$$

It turns out that there are many possible solutions in this case however, which makes it rather impractical to work with. SVMs offer the term $\sum_{i=1}^{N}\xi_i =$

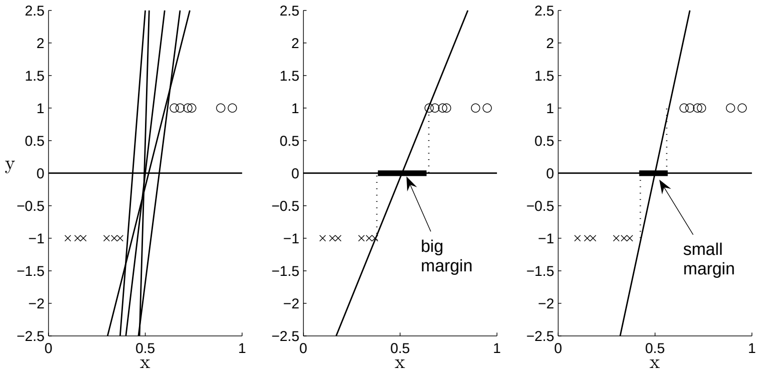


Figure 1.3: Linear solutions to a simple binary classification example in $1D$-space. Panel (a) shows many solutions that produce zero sum of training errors. Panel (b) shows the SVM solution in case of zero sum of training errors, along with the SVM margin. Panel (c) shows a random linear zero-error solution and its margin, which is smaller than the margin of the SVM line.

$\sum_{i=1}^{N}[1 - y_i\hat{y}_i]_+$ as a (rough) approximation to $\sum_{i=1}^{N}[0 - y_i\text{sign}(\hat{y}_i)]_+$, and do not suffer from this problem. In addition, the $\boldsymbol{\beta}'\boldsymbol{\beta}$ makes sure that the *predicted values* are shrunk, or *penalized.* In this particular case, we have a "penalization towards the average" or rather "penalization towards β_0", and not penalization towards zero, as the intercept term is not penalized. The $\boldsymbol{\beta}$'s play a similar role as the α in the tossing-a-coin example.

Let us comment on one final detail. It was noted in the previous paragraph that the unbiased classification model would be maximizing (just) the training misclassification rate $(1/N)\sum_{i=1}^{N}[0 - y_i\text{sign}(\hat{y}_i)]_+$. This model is unbiased in the sense that if we have an infinite amount of data at each possible combination of values for the inputs, then this model will predict the class with highest probability of occurrence at each such point. That is, if we observe at a particular point that the biggest number of observations come from one of the classes, then our prediction at this point should be this particular class. This simple rule is called the *Bayes* model. The Bayes model minimizes the expected loss in case we have decided to have a $0 - 1$ loss in mind, that is, it maximizes the expected accuracy. Actually, to reiterate, the $0-1$ loss implies the maximization of the expected accuracy criterion in much the same way as the squared-error loss implies the minimization of the expected MSE (EMSE) criterion.

A graphical representation of SVM in the one-dimensional case could come in handy. It shows that the idea of penalization in this case has a geometrical interpretation as a width of the so-called *margin*. Consider Figure 1.3, which depicts a simple two-class classification problem in a one-dimensional case. It is important to keep in mind that differences between predicted and true values count as errors only if predictions are below 1 for the positive observations with $y = 1$ and shown as circles and above -1 for the negative observations with $y = -1$ and shown as crosses. Panel (a) shows many possible linear solutions that do not induce any error, since predicted values have the correct sign and are above 1 or below -1. Panel (b) shows the SVM line, which is the *flattest* possible among all lines that induce no error. The term flattest is used to denote that the coefficients of the line (except the intercept) are close to zero. Since the SVM line is the flattest, an interesting phenomenon can be observed. Namely, the value of x for which $y = 1$, and the value of x for which $y = -1$ are most distant from each other. This distance is referred to as the margin. Thus, greater penalization in the form of increased λ results in:

- model coefficients (in general, the $\boldsymbol{\beta}$'s) being closer to zero;
- an increase in the width of the margin; and
- predicted raw output values $\hat{y}_i$ (such as those in 1.9) being closer, or shrunk, to each other.

Consider also Panel (c), which shows the (relatively smaller) margin of another line that produces zero sum of errors. All lines in Panel (a) produce zero sum of errors, also referred to as zero empirical bias or zero empirical error. However, only the flattest line among them is associated with the smallest variance among future estimates. Notice that the flattest possible line, which is a horizontal line, is associated with the minimal possible variance among future estimates, that is a zero variance. The SVM line is the flattest line (in the figure) among all the lines that correctly classify the training observations. It is possible to achieve an even flatter SVM solution, but it is bound to bring along some errors, or increased empirical bias. Depending on the manually-controllable parameter λ these errors are considered either “small” or “big”. Thus, λ controls the extent to which empirical bias could be tolerated. The SVM optimization problem will yield coefficients that produce optimal flatness (or, margin) for a given *fixed* level of empirical bias, where the level of bias is determined by λ. What the SVM optimization problem cannot do is to yield the optimal allocation between the (non-fixed) level of empirical bias and the degree of flatness of the solution, that is to find the optimal λ. The optimal λ has to be found using methods that approximate the misclassification rate over any (new) test set, such as the cross-validation method.

A Note on Regularization

The exact interpretation of regularization parameters such as the λ parameter in SVMs for classification (see 1.9) has sparkled a debate between the Statistics and

Econometrics literature on the one hand and the Machine Learning literature on the other. Rather than taking a side in this debate, I would rather present shortly and informally my viewpoint on the effect of regularization parameters, which is in many respects a consensus view. In fact, regularization can be seen as a way of shrinking predictions and/or providing a prediction function that finds itself in-between best-fit functions from different classes of functions.

Observe, first, the interesting property that for $\lambda = 0$ the optimal SVM lines in Figure 1.3, Panel (a) are infinitely many. These lines are lines of best fit, where the *fit* is measured as the sum of all training errors, here: $\sum_{i=1}^{N}[1 - y_i\hat{y}_i]_+$. In this case the fit is perfect as the sum of training errors is zero for any such line. In a hypothetical case of a quadratic-error model, which is applicable in case of a squared-error loss function, the fit is the sum of squared errors, or the sum of squared discrepancies between the training output values and the corresponding predicted output values. For $\lambda = \infty$, the optimal $\boldsymbol{\beta}$'s in SVM are zeros, and thus any predicted output value is equal to β_0. This is a penalization towards β_0 in its pure form. Thus, the predicted output value in this case is a *constant* whatever the inputs. If the intercept β_0 is penalized as well by this λ, then the predicted output value is 0 whatever the inputs, and we have a pure penalization towards zero.

This role of λ can be equivalently described using the term class of functions, or *function class* for short. Informally, the simplest function class contains just one function that outputs a predicted value of zero for the dependent variable y, whatever the inputs, that is $\hat{y}^{(0)} = 0$. Let us call this function class "class 0". Arguably, this function is in many respects more preferable than a random assigning of prediction values for the dependent variable, at least due to the fact that the variance of the predictions is zero. Next comes "class 1", which is the function class of (all) constants. When $\lambda = \infty$ and β_0 is not penalized, the SVM prediction function belongs to this class, as it is $\hat{y}^{(1)} = \beta_0^{(1)}$ whatever the inputs and $\beta_0^{(1)}$ is a constant. Here, the optimal SVM prediction function is the *constant of best fit*, as there does not exist another function from function class 1 – that is, another constant – which produces a smaller sum of training errors. This simple solution is also known as the "majority voting rule", as the optimal $\beta_0^{(1)}$ is either any value > 0 or any value < 0, depending on whether there are more positive or negative observations in the training data set. As there are (infinitely) many solutions for $\beta_0^{(1)}$ that provide the majority voting rule, there are (infinitely) many constants of best fit in the case of SVM. Notice that function class 0 is included already in function class 1, as zero is an allowable value for $\beta_0^{(1)}$. The next function class, "class 2", is the class of (all) linear functions. When $\lambda = 0$, the SVM line belongs to this class, and is expressed as $\hat{y}^{(2)} = \beta_0^{(2)} + x\beta_1^{(2)}$ for the (single) input x. This SVM line is the *line of best fit*, as there does not exist another function from class 2 – if fact, another combination of β_0 and β_1 – which produces a smaller sum of training errors. Actually, there might exist (infinitely) many lines of best fit, as can be seen in Panel (a) of Figure 1.3, where is it evident that there are infinitely many lines of perfect fit, that is lines for which the sum of training errors is zero, $\sum_{i=1}^{N}[1 - y_i\hat{y}_i]_+ = 0$. Notice also that class 2 contains in itself class 1 and therefore also class 0. That is, in a way all function classes 0, 1 and 2 can be viewed as being comprised of "lines". Nevertheless, these classes are different.

To sum up, the ultimate role of λ, for example in the context of Figure 1.3, is to provide *predictions*, $\hat{y}$'s, that are in-between the predictions of a best-fit function from function class 2, $\hat{y}^{(2)} = \beta_0^{(2)} + x\beta_1^{(2)}$, and the predictions of a best-fit function from function class 1, $\hat{y}^{(1)} = \beta_0^{(1)}$. This is achieved when λ is in-between the extremes of 0 and ∞. This interpretation is close to the statistical view on regularization. Equivalently stated, the role of λ is to provide a *prediction function* that lies in-between a prediction function of best fit from function class 2 and a prediction function of best fit from function class 1. Thus, the actual role of λ is not really to provide a new, synthetic function class that finds itself somehow in-between two (not necessarily adjacent) function classes. Rather, the usage of λ provides a compromise solution of two concrete functions from two different (not necessarily adjacent) function classes, where these two functions are functions of *best fit* from their respective function classes. For example, the SVM line of Figure 1.3, Panel (b) can be thought of as the prediction function that founds itself in-between a function of best fit from the class of all constants (class 1) and a function of best fit from the class of all lines (class 2). This interpretation is close to the Machine Learning view on regularization. Here, the introduction of λ is seen as a way to reduce, so to say, a best-fit function from function class 2 towards a best-fit function from the *simpler* function class 1. A way to quantify the extent of such simplicity, or inversely stated, *complexity*, is provided by a function-class complexity measure called the VC-dimension (Vapnik, 1995). For example, the class of constants, class 1, has a VC-dimension of 1; the class of linear functions, class 2, has a VC-dimension of 2 and so on. Notice that we speak about the class of linear functions in the input-output data space here, and not the space of the inputs only. There exist upper bounds on the expected misclassification rate over the whole domain of input values that use the VC-dimension, and are referred to as generalization-error bounds. As a matter of fact, the SVM method has been initially proposed and justified in this context. As the introduction of λ *effectively* reduces the *capacity*, or the *freedom*, of the linear functions from 2 towards 1, one can say that λ gives rise to an *effective* VC-dimension between 2 and 1 or an *effective* degrees of freedom between 2 and 1. The last sentence uses mixed terminology from Statistics/Econometrics and Machine Learning, and basically conveys the notion that these scientific areas converge here to the same ideas, but expressed in a different way. The degrees of freedom are in general equal to the number of input variables p plus one (for the intercept). In the classification case, the VC-dimension is equal to the same value if the prediction function is a hyperplane. The role λ, then, is to provide a prediction function that finds itself in-between two best-fit functions from function class p and function class 1 respectively, as the predictions are being shrunk towards β_0.

1.3 Nonlinear Learning via Duality and Kernels

In the kernel approach, nonlinear solutions to the target classification or regression function are achieved by creating new explanatory variables and then solving for a new linear target function based on the original variables *and* the new variables. Nonlinearity is observed in the space of the original variables only. In the augmented space, the solution is linear. The simplest example in this respect is the simple quadratic function $y = ax^2 + bx + c$. In the space with coordinates x, the single input, and y, the output, this function is nonlinear – that is, it is quadratic – whereas in the space with coordinates x, x^2 and y, the function is linear – that is, it is a plane in a 3D space. The introduction of such new variables is usually referred to as *extending the basis.*

Before proceeding, it is useful to point out which meaning of term kernel we will have in mind here. In the context of this thesis, a kernel is referred to the inner product or dot product of two vectors in the extended-basis, or *feature*, space. What is important is that the dot product is computable using the coordinates of the vectors in the original space only. The dot product between input vectors from $\mathbb{R}^p$ $\mathbf{x}_i$ and $\mathbf{x}_j$ in the original space is denoted as $\mathbf{x}_i'\mathbf{x}_j$, and in the feature, higher-dimensional space as $\boldsymbol{\phi}(\mathbf{x}_i)'\boldsymbol{\phi}(\mathbf{x}_j)$, where $\boldsymbol{\phi}(\mathbf{x}_i)$ are the coordinates of $\mathbf{x}_i$ mapped into the higher-dimensional, feature space. More formally, $\mathbf{x}_i$ is said to be mapped to the higher-dimensional space via mapping $\boldsymbol{\phi}$ from $\mathbb{R}^p$ to $\mathbb{R}^m$ with $m > p$. It is important to note that there are no new explanatory variables from the outside world entering into the problem. The new variables $\boldsymbol{\phi}(\mathbf{x})$ are combinations of the original variables only, that is, the new coordinates of points in the higher-dimensional space are created from the coordinates of the original points. The kernel $k(\mathbf{x}_i, \mathbf{x}_j)$ stands for the dot product between two points, $\boldsymbol{\phi}(\mathbf{x}_i)$ and $\boldsymbol{\phi}(\mathbf{x}_j)$, in feature space, that is, $k(\mathbf{x}_i, \mathbf{x}_j) = \boldsymbol{\phi}(\mathbf{x}_i)'\boldsymbol{\phi}(\mathbf{x}_j)$. The special property of kernels is that this dot product is computed using the coordinates of the original points $\mathbf{x}_i$ and $\mathbf{x}_j$, without knowing the explicit coordinates $\boldsymbol{\phi}(\mathbf{x}_i)$ and $\boldsymbol{\phi}(\mathbf{x}_j)$.

Duality refers to reexpressing an optimization problem in equivalent *dual* optimization form using Lagrange multipliers, also called dual variables. A constrained minimization problem can be reexpressed as an equivalent maximization problem using (possibly restricted) nonnegative dual variables. If the dual variables are defined as nonpositive, then this maximization problem is actually turned into a minimization problem. Dual representation is essential for all kernel methods, because it allows for efficient implementation of nonlinear solutions. Sometimes the dual representation is not only efficient but also the only one solvable in practice. For that reason, it would be illuminating to derive the SVM for classification in dual form, step by step. This line of dual derivation is also followed by the introduced methods SH, NCH, and SNN in Chapters 5, 6, and 7.

Let us start from the primal SVM formulation (1.10) with feature vectors $\boldsymbol{\phi}(\mathbf{x})$ instead of the original vectors $\mathbf{x}$, then form the corresponding Lagrangian and take its first derives with respect to the unknown variables (β_0, $\boldsymbol{\beta}$, and $\boldsymbol{\xi}$). The Lagrangian of the primal problem (1.10) with feature vectors $\boldsymbol{\phi}(\mathbf{x}_i)$ is

$$L_P(\beta_0, \boldsymbol{\beta}, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\eta}) = C\sum_{i=1}^{N}\xi_i + \frac{1}{2}\boldsymbol{\beta}'\boldsymbol{\beta} - \sum_{i=1}^{N}\alpha_i\{y_i(\beta_0\boldsymbol{\phi}(\mathbf{x}_i)'\boldsymbol{\beta}) - 1 + \xi_i\} - \sum_{i=1}^{N}\eta_i\xi_i,$$

where α_i and η_i, $i = 1, 2, \ldots, N$, are nonnegative Lagrange multipliers. Next, we take the first derivatives of the Lagrangian with respect to (β_0, $\boldsymbol{\beta}$, and $\boldsymbol{\xi}$) and set them to zero:

$$\frac{\partial L_P}{\partial \beta_0} = 0 \Leftrightarrow -\sum_{i=1}^{N}\alpha_i y_i = 0$$

$$\frac{\partial L_P}{\partial \beta_k} = 0 \Leftrightarrow \beta_k = \sum_{i=1}^{N} y_i\alpha_i\boldsymbol{\phi}(\mathbf{x}_i)_k \text{ for } k = 1, 2, \ldots, m$$

$$\frac{\partial L_P}{\partial \xi_i} = 0 \Leftrightarrow C - \alpha_i - \eta_i = 0 \text{ for } i = 1, 2, \ldots, N,$$

where m is the number of input variables in the extended, higher-dimensional space, and consequently the number of coefficients in vector $\boldsymbol{\beta}$ in this space; and, $\boldsymbol{\phi}(\mathbf{x}_i)_k$ is the k^{th} coordinate of $\boldsymbol{\phi}(\mathbf{x}_i)$ in the higher-dimensional space. Plugging back these results into the Lagrangian $L_P(\beta_0, \boldsymbol{\beta}, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\eta})$ we arrive at the SVM dual (maximization) problem

$$\min_{\boldsymbol{\alpha}} \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} y_i y_j \alpha_i \alpha_j [\boldsymbol{\phi}(\mathbf{x}_i)'\boldsymbol{\phi}(\mathbf{x}_j)], \tag{1.11}$$

$$\text{subject to } \alpha_i \geq 0,\ i = 1, 2, \ldots, N, \text{ and } \textstyle\sum_{i=1}^{N}\alpha_i y_i = 0.$$

Notice that – somewhat luckily – only $\boldsymbol{\phi}(\mathbf{x}_i)'\boldsymbol{\phi}(\mathbf{x}_j)$ for any input pair has to be known explicitly, and not $\boldsymbol{\phi}(\mathbf{x})$ explicitly. This allows for the usage of kernels, which replace the dot product $\boldsymbol{\phi}(\mathbf{x}_i)'\boldsymbol{\phi}(\mathbf{x}_j)$ with a kernel $k(\mathbf{x}_i, \mathbf{x}_j)$, and therefore for computing efficiently nonlinear decision-surface solutions in the original training-data space. Popular kernels are the polynomial kernel of degree d, $k(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i'\mathbf{x}_j + 1)^d$, the Radial Basic Function (RBF) kernel with proximity parameter γ, $k(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma \parallel \mathbf{x}_i - \mathbf{x}_j \parallel^2)$, and the linear kernel, $k(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i'\mathbf{x}_j$, which gives the linear solution in the original data space with no mapping.

A Counter Example: OLS in Dual Form

As noted in the beginning of this Section, the usage of kernels in a so-called dual problem allows for nonlinear decision-surface solutions in the original training-data space. It is *not* possible however to always use kernels in the dual. That is, kernelization and dualization do not automatically go hand in hand. The most prominent example thereof is the dual form of the pervasive Ordinary Least Squares (OLS) estimation, which happens not to allow for the usage of kernel functions.

Below, we derive the dual OLS optimization form that shows why OLS per se – that is, without penalization of the squared norm $\boldsymbol{\beta}'\boldsymbol{\beta}$ of the coefficients – cannot be kernelized. We also show that the act of penalizing this squared norm makes the usage of kernels possible. In this way, OLS estimation is turned into Ridge Regression (RR) estimation.

A linear estimation model can always be presented in dual form, but it cannot always be kernelized. The reason is that it is the quadratic penalty that gives rise to a formulation of the dual involving dot, or inner, products of the inputs of the form $\boldsymbol{\phi}(\mathbf{x}_i)'\boldsymbol{\phi}(\mathbf{x}_i)$ and not explicit linear expansions of the form $\boldsymbol{\phi}(\mathbf{x}_i)$. To illustrate this, let us derive the dual of the OLS estimation problem and then the dual of the Ridge Regression problem. The OLS dual can be derived from the (OLS) primal as follows. We start from the usual primal form:

$$\begin{aligned} &\min_{\mathbf{e},\boldsymbol{\beta}} \frac{1}{2}\mathbf{e}'\mathbf{e}, \\ &\text{subject to} \quad \mathbf{e} = \mathbf{y} - \boldsymbol{\Phi}(\mathbf{X})'\boldsymbol{\beta}, \end{aligned} \tag{1.12}$$

where p is the number of inputs, N in the number of training observations, $p < N$, $\mathbf{e}$ is a $N \times 1$ vector of errors, and $\boldsymbol{\Phi}(\mathbf{X})$ is a matrix of N row vectors $\boldsymbol{\phi}(\mathbf{x}_i)'$, $i = 1, 2, \ldots, N$. From now on we will denote $\boldsymbol{\Phi}(\mathbf{X})$ by $\boldsymbol{\Phi}$ for short. This optimization problem can be solved by forming the Lagrangian

$$L_P(\mathbf{e}, \boldsymbol{\beta}, \boldsymbol{\alpha}) \quad = \quad \frac{1}{2}\mathbf{e}'\mathbf{e} + (\mathbf{y} - \boldsymbol{\Phi}'\boldsymbol{\beta})'\boldsymbol{\alpha},$$

where $\boldsymbol{\alpha}$ is a $N \times 1$ vector of Lagrange multipliers. By taking the first derivatives with respect to $\mathbf{e}$ and the coefficients $\boldsymbol{\beta}$, and setting them equal to zero, we establish that $\mathbf{e} = \boldsymbol{\alpha}$ and $\boldsymbol{\Phi}'\boldsymbol{\alpha} = \mathbf{0}$. Plugging back the solution for the primal variables into the Lagrangian, it becomes a function that should be maximized with respect to the unknown dual variables $\boldsymbol{\alpha}$. Using the fact the $\boldsymbol{\alpha} = \mathbf{e}$ in the optimum, we establish the dual OLS formulation:

$$\begin{aligned} &\max_{\mathbf{e}} \mathbf{e}'\mathbf{y} - \frac{1}{2}\mathbf{e}'\mathbf{e}, \\ &\text{subject to} \quad \boldsymbol{\Phi}'\mathbf{e} = \mathbf{0}. \end{aligned} \tag{1.13}$$

As the $\boldsymbol{\Phi}$ matrix contains a column of ones, the constraints $\boldsymbol{\Phi}'\mathbf{e} = \mathbf{0}$ imply that $\sum_{i=1}^{N} e_i = 0$. The solution for $\mathbf{e}$ of the primal minimization problem (1.12) and the dual maximization problem (1.13) should coincide at the optima, therefore $0.5\mathbf{e}'\mathbf{e} = \mathbf{e}'\mathbf{y} - 0.5\mathbf{e}'\mathbf{e} \Leftrightarrow \mathbf{e}'\mathbf{e} = \mathbf{e}'\mathbf{y}$. Notice that the matrix $\mathbf{\Phi}$, as opposed to $\mathbf{\Phi\Phi}'$ enters in both the primal and the dual optimization problem (refer to the SVM dual problem 1.11). Therefore, $\mathbf{\Phi}$ has to be known *explicitly* in both formulations and the kernel "trick" of using a kernel (matrix) expression to replace the term $\mathbf{\Phi\Phi}'$ cannot be applied.

Now we shall see how the Ridge Regression (RR) can be expressed in dual form, where explicit knowledge only of $\mathbf{\Phi\Phi}'$ is needed, and not of $\mathbf{\Phi}$ itself. Basically, the RR dual resembles the one of OLS, with two differences. The constraint $\mathbf{\Phi}'\mathbf{e} = 0$ is removed, but the term $-(1/2\lambda)(\mathbf{\Phi}'\mathbf{e})'(\mathbf{\Phi}'\mathbf{e})$ is added to the dual objective function, where λ is a Ridge Regression manually-adjustable parameter determining the strength of the penalty. Formally, in order to derive the dual RR from the primal RR, the following steps are taken:

$$\begin{aligned} &\min_{\mathbf{e},\boldsymbol{\beta}} \frac{1}{2}\mathbf{e}'\mathbf{e} + \frac{\lambda}{2}\boldsymbol{\beta}'\boldsymbol{\beta}, \\ &\text{subject to} \quad \mathbf{e} = \mathbf{y} - \mathbf{\Phi}'\boldsymbol{\beta}, \end{aligned}$$

where $\lambda > 0$, the first column of $\mathbf{\Phi}$ is a vector of ones, and $\mathbf{e}$ is a $N \times 1$ vector of errors. We solve this problem by forming the Lagrangian

$$L(\mathbf{e},\boldsymbol{\beta},\boldsymbol{\alpha}) \quad = \quad \frac{1}{2}\mathbf{e}'\mathbf{e} + \frac{\lambda}{2}\boldsymbol{\beta}'\boldsymbol{\beta} + (\mathbf{y} - \mathbf{\Phi}'\boldsymbol{\beta})'\boldsymbol{\alpha}.$$

Taking the first derivatives with respect to $\mathbf{e}$ and $\boldsymbol{\beta}$ and setting them to zero yields $(1/\lambda)\mathbf{\Phi}'\boldsymbol{\alpha} = \boldsymbol{\beta}$, and $\mathbf{e} = \boldsymbol{\alpha}$. Plugging back these results into the Lagrangian and using the fact that $\mathbf{e} = \boldsymbol{\alpha}$ at the optimum, we arrive at the equivalent (dual) maximization RR problem

$$\max_{\mathbf{e}} = -\frac{1}{2}\mathbf{e}'\mathbf{e} + \mathbf{e}'\mathbf{y} - \frac{1}{2\lambda}\mathbf{e}'\mathbf{\Phi\Phi}'\mathbf{e}. \qquad (1.14)$$

The dual RR problem is an unconstrained problem. In many applications however it is common *not* to penalize the intercept term. This leads to an identical formulation of dual RR with the additional constraint that the sum of α's should be zero, $\sum_{i=1}^{N} \alpha_i = 0$, and $\mathbf{\Phi}$ does not contain a column of ones. Notice that $\mathbf{\Phi\Phi}'$ has to be known explicitly in (1.14), and not $\mathbf{\Phi}$, which enables the usage of kernels.

1.4 Outline of the Thesis

The thesis is structured into two main parts. The first part is more practical and concentrates on new applications of established methods. The second part is more theoretical and focuses on putting forward new classification methods.

Extensive applications of these new methods to real-data tasks and derivations of subtle theoretical properties, among others, are the subject of future research. Most of the chapters have already been published or accepted for publication. Full references are given in the chapters themselves.

Chapter 2: Estimating the Market Share Attraction Model using SVRs. This chapter is a nice example of how knowledge from three disciplines can be combined to produce a formidable result. We start with the application task: to predict the monthly market-share allocation among 36 car brands for a certain out-of-sample period. Traditionally, an Econometrics approach has been proposed to solve this task, which has its merits. Inside the Econometrics model however a regression problem has to be solved that cries, so to say, for regularization. A well-performing regularization technique can readily be borrowed from the Machine Learning literature, namely SVRs. Using SVRs to solve the Econometric Market Share Attraction Model improves the forecasted results dramatically. This chapter has been accepted for publication in Econometric Reviews (Nalbantov, Franses, Groenen, & Bioch, 2008).

Chapter 3: Equity Style Timing using Support Vector Regressions. This is yet another demonstration of the benefits that regularization techniques provide. In this case, SVRs are applied to financial series forecasting. More concretely, the task is to come up with a so-called financial rotation strategy between so-called Value and Growth stocks. Similar strategies are quite popular investment strategies. The forecasting results indicate tremendous improvement over passive models during an out-of-sample period of about 10 years. This chapter has been published as Nalbantov, Bauer, and Sprinkhuizen-Kuyper (2006).

Chapter 4: Solving and Interpreting Binary Classification Problems in Marketing with SVMs. This chapter bridges the gap that invariably arises between improved performance and decreased interpretability of results provided by SVMs in classification tasks. As SVMs are mostly nonlinear learners (that is, functions), the coefficients they output take different values over different ranges of the inputs. This makes the interpretation of the effect of the inputs hard. To make matters worse, sometimes no coefficients are output. Still, we would like to have an idea about the individual effects that the inputs have on the behavior of the output variable. This effect can be extracted using probability estimates for the predicted classes. Once these estimates are available, one of the input variables is changes, while the rest are hold fixed on their average levels. The change in class-belonging probability gives a clear idea of the (nonlinear) effect of each of the inputs, leading to a better interpretation of the effects of the inputs. This chapter has been published as Nalbantov, Bioch, and Groenen (2006b).

Chapter 5: Instance-Based Penalization Methods for Classification. This is the most important chapter of the thesis, as it combines in one place the real, new contribution of this research. Three new classification techniques are put forward, which contribute to the plethora of existing classification techniques. The proposed techniques are Support Hyperplanes, Nearest Convex Hull classification, and Soft Nearest Neighbor. The common element among these methods is, first of all, that they are regularization methods. That is, all of them try to find a balance between model complexity and sum of training errors, which is ultimately responsible for the excellent performance results. To the best of my knowledge, they are the first instance-based regularization techniques. Next to the inherent regularization capability, these techniques are able to employ kernels, which are responsible for providing nonlinear solutions by mapping the data into a higher-dimensional space. Thus, the proposed three methods can also be viewed as kernel methods. There is some sort of similarity between these techniques and SVMs, for instance. That is why a discussion on the relative merits of among all these techniques is presented. To sum up, this chapter proposes three new classification techniques, all of which fall under the general heading instance-based large-margin kernel methods.

Chapter 6: Classification with Support Hyperplanes. This chapter is a spin-off of Chapter 5. It concentrates on a new classification technique called Support Hyperplanes. This chapter has been published as Nalbantov, Bioch, and Groenen (2006a).

Chapter 7: Nearest Convex Hull Classification. This chapter is a spin-off of Chapter 5. It concentrates on a new classification technique called Nearest Convex Hull classification.

Chapter 8: SVM-Maj: A Majorization Approach to Linear Support Vector Machines with Different Hinge Errors. This chapter contributes to the debate on how to solve the optimization problem that arises in linear SVMs in a fast way. Existing solvers are shown to have a worse speed, especially so when the number of inputs is small and the data set in not so big. Next to increased speed, a big advantage of the majorization approach is that different loss functions can be employed with ease, such as the (default) linear hinge loss, the quadratic hinge loss, and the Huber hinge loss. An earlier version of this chapter has been published as Groenen, Nalbantov, and Bioch (2007).

Part I

Some Applications of SVMs and other Kernel Methods to Financial and Marketing Problems

Chapter 2

Estimating the Market Share Attraction Model using SVRs*

We propose to estimate the parameters of the Market Share Attraction Model (Cooper & Nakanishi, 1988; Fok & Franses, 2004) in a novel way by using a nonparametric technique for function estimation called Support Vector Regressions (SVR) (Vapnik, 1995; Smola, 1996). Traditionally, the parameters of the Market Share Attraction Model are estimated via a Maximum Likelihood (ML) procedure, assuming that the data are drawn from a conditional Gaussian distribution. However, if the distribution is unknown, OLS (Ordinary Least Squares) estimation may seriously fail (Vapnik, 1982). One way to tackle this problem is to introduce a linear loss function over the errors and a penalty on the magnitude of model coefficients. This leads to qualities such as robustness to outliers and avoidance of the problem of overfitting. This kind of estimation forms the basis of the SVR technique, which, as we will argue, makes it a good candidate for estimating the Market Share Attraction Model. We test the SVR approach to predict (the evolution of) the market shares of 36 car brands simultaneously and report promising results.

*This chapter has been accepted for publication as Nalbantov et al. (2008): Nalbantov, G.I., Franses, P.H.B.F., Groenen, P.J.F., & Bioch, J.C. (2008). Estimating the Market Share Attraction Model using Support Vector Regressions. *Econometric Reviews.* (Accepted)

2.1 Introduction

The Market Share Attraction Model is a popular tool for analyzing competitive structures (Cooper & Nakanishi, 1988; Fok & Franses, 2004). It is typically applied for simultaneously predicting the market shares of several brands within a given product category. The model helps to evaluate the effect of marketing-mix variables on brands' performances as well as the effect of an individual brand's own efforts while conditioning on competitors' reactions. A detailed econometric analysis of the model can be found in Fok, Franses, and Paap (2002). What makes this model rather special is the requirement that the forecasted market shares are all non-negative and sum to unity.

The traditional unrestricted Market Share Attraction Model often suffers from poor predictability, especially for the relatively larger brands. The poor performance is likely to be due to various causes, including heteroscedasticity and failure to account for a trend in the data. The huge number of coefficients to be estimated is another source of concern. A common way to address those issues is to restrict the model coefficients or to aggregate brands into categories. More fundamentally, however, one can also address the applied estimation procedure, which is Maximum Likelihood (ML) with assumed Gaussian noise model, leading to OLS (Ordinary Least Squares) estimation of model coefficients. OLS estimation is appropriate (and optimal) in cases the dependent variable has been drawn from a conditional Gaussian distribution. In cases where this is not so, the least-squares techniques are suboptimal and could lead to severely mismatched solutions for some densities (Vapnik, 1982). Then, improved coefficient estimation can be obtained in a variety of ways. One way is by using estimation methods put forward in the literature on Support Vector Machines (SVMs), and this is what we primarily address in this paper.

SVMs are a nonparametric tool that can be used for both classification and regression estimation tasks (Vapnik, 1995; Burges, 1998; Cristianini & Shawe-Taylor, 2000). They have gained considerable popularity during the last years, following a series of successful applications in areas ranging from Bioinformatics and Optical Character Recognition to Economics and Finance (see, among others, Schölkopf, Guyon, & Weston, 2001; Schölkopf, Burges, & Vapnik, 1995; Pérez-Cruz, Afonso-Rodríguez, & Giner, 2003; Tay & Cao, 2001).

The SVM technique for regression estimation is referred to as Support Vector Regression (SVR). Essentially, (linear) SVR coefficient estimation capitalizes on the utilization of a specific non-Gaussian noise model and an L_2-norm penalization of the coefficients. The utilization of a specific noise model is in itself rooted in two observations. The first one is the proposition that the linear loss function is the best error loss function of the worst model over any probability density function of the dependent variable given the independent variables (Huber, 1964). Thus, if the dependent variable is drawn from an unknown distribution, a linear loss function over the errors could be more appropriate than the common quadratic one. The second building block is the bound obtained on the test error (less than infinity) using the so-called Structural Minimization Principle (Vapnik, 1995). This bound arises when a certain error-insensitive region around the predicted value of the dependent variable is introduced. The width of this region can be made arbitrarily small however.

These two observations result in the first departure of SVR from OLS: instead of the common quadratic loss, SVR utilize the so-called ϵ-insensitive loss function (Vapnik, 1995). This robust loss function penalizes the discrepancy between a true and a corresponding target value linearly, if this discrepancy is more than (a user-defined) ϵ. Up to ϵ there is no loss incurred at all. In other words, the ϵ-insensitive loss function is: $|\varepsilon|_\epsilon \equiv |y - f(\mathbf{x})|_\epsilon \equiv \max\{0, |y - f(\mathbf{x})| - \epsilon\}$, where y is a target value and $\mathbf{x}$ is a vector of predictor variables.

To relate SVR estimation to our task of estimating the coefficients of the Market Share Attraction Model, we argue as follows. We assume that the data is generated by an underlying functional dependency plus additive noise, that is, $y = f_{true}(\mathbf{x}) + \varepsilon$. Normality tests show that for the data set we analyze, the noise is non-Gaussian. This renders the specification of a Gaussian likelihood for ML estimation (which is equivalent in this case to OLS estimation) problematic. Thus, there is room for improvement over the standard Market Share Attraction Model, which assumes a Gaussian noise scheme, i.e. the loss function is $0.5\varepsilon^2$, with corresponding density model $p(\varepsilon) = \frac{1}{\sqrt{2\pi}} \exp(-0.5\varepsilon^2)$. One way to remedy the standard Market Share Attraction Model approach is to consider more robust density noise models. SVR, for example, considers the model $p(\varepsilon) = \frac{1}{2(1+\varepsilon)} \exp(-|\varepsilon|_\epsilon)$ (see Smola & Schölkopf, 1998, 2004), which corresponds to the ϵ-insensitive loss function. Other robust noise models could also be used, like the student-t distribution model. Alternatively, the noise model itself could be (fully) estimated via, for example, a Gaussian mixture[1]. These approaches however fall out of the scope of the paper.

Next to the usage a more robust, ϵ-insensitive loss function, SVR departs from OLS in that it estimates model coefficients by including also an L_2-norm penalization term for them, as is done in Ridge Regression (RR) for instance. This puts SVR estimation close to penalized ML estimation, as RR (linear) coefficient estimates are obtained via penalized ML (Cawley & Talbot, 2002).

Nonlinear estimation is the final building block of SVR. It is achieved by mapping the predictors from their original (linear) space to a higher-dimensional space (via kernel functions). Linear estimation in this higher-dimensional space corresponds to a nonlinear regression surface in the original linear space.

A nice feature of SVRs is that they solve a quadratic programming problem to obtain a solution. Unlike competing techniques such as Neural Networks (for a general reference see (Bishop, 1995), (Hastie et al., 2001), among others), this solution is unique and does not suffer from a local minimum. This adds to the

[1] We thank an anonymous referee for pointing this out.

desirable properties of SVRs such as the ability to avoid in-sample overfitting and robustness against outliers in the data, which arise due to the ϵ-insensitive loss function and the L_2-norm penalization of model coefficients. All of these properties make SVRs very suitable for application to the standard Market Share Attraction Model.

We compare the performance of the traditional OLS solution to the Market Share Attraction model with that of SVR and RR in the experimental part of the paper. RR is a good competitor to SVR because it likewise penalizes the L_2-norm of model coefficients, and is able to employ the same type of nonlinearities as SVR via kernel functions. The nonlinear RR is referred to as Kernel Ridge Regression (KRR) (see, e.g., Saunders, Gammerman, & Vovk, 1998; Hastie et al., 2001; Shawe-Taylor & Cristianini, 2004; Pozdnoukhov, 2002). A common theme between RR and SVR is the penalization of the L_2-norm of the model coefficients. The major difference is the loss function employed: quadratic in the first case and ϵ-insensitive in the latter case. This makes the comparison of these penalization techniques quite interesting.

The paper is organized as follows. The next section introduces the Market Share Attraction Model in its traditional form. Section 2.3 outlines the SVR technique and augments it with SVR estimation and Section 2.4 discusses its nonlinear extension. In addition, we give a short account of both (linear) RR and (nonlinear) KRR. We then carry out pairwise tests for superior predictive ability (Hansen, 2005) across OLS, linear and nonlinear RR, and linear and nonlinear SVR in Section 2.5, which presents our main findings on a data set that is used to predict the evolution of market shares of 36 car brands for a certain period. The final section gives a conclusion.

2.2 The Market Share Attraction Model

The purpose of the Market Share Attraction Model is to provide an overall model for the market share $M_{i,t}$ of brand i at time t for the I brands constituting the market over a period from $t = 1$ to T. An important characteristic of a market share $M_{i,t}$ is that $0 \leq M_{i,t} \leq 1$ and that it sums over all brands to one, that is, $\sum_{i=1}^{I} M_{i,t} = 1$. The typical interval between the measurements of the market shares is a week or a month. The model uses K predictor variables with nonnegative values $x_{k,i,t}$ to predict the market shares described below. Typical predictor variables are price, distribution, advertising spending, etc. The usefulness of the model lies in its ability to describe the competitive structures and to infer cross effects of marketing-mix instruments (Fok et al., 2002).

The so-called Multiplicative Competitive Interaction (MCI) specification of a market share $M_{i,t}$ builds on the attraction $A_{i,t}$ of brand i at time t that is defined as

$$A_{i,t} = \exp(\mu_i + \varepsilon_{i,t}) \prod_{j=1}^{I} \prod_{k=1}^{K} x_{k,j,t}^{\beta_{k,j,i}} \qquad \text{for } i = 1, \ldots, I, \tag{2.1}$$

where $\beta_{k,j,i}$ is the unknown coefficient for brand i and μ_i is a brand-specific intercept term corresponding to the size of the brand. The vector of error terms $\boldsymbol{\varepsilon}_t = [\varepsilon_{1,t}, \ldots, \varepsilon_{I,t}]'$ is usually assumed to be normally distributed with zero mean and some unknown covariance matrix Σ. The market share of brand i at time t can be defined as the attraction of brand i at t divided by the sum of all attractions at t, that is,

$$M_{i,t} = \frac{A_{i,t}}{\sum_{j=1}^{I} A_{j,t}} \qquad \text{for } i = 1, \ldots, I. \tag{2.2}$$

The model in (2.1) with (2.2) is the Market Share Attraction Model. Notice that the definition of the market share of brand i at time t given in (2.2) implies that the attraction of the product category is the sum of the attractions of all brands and that equal attraction of two brands results in equal market shares.

In addition to the predictor variables $x_{k,i,t}$, one could also include lagged variables $x_{k,i,t-1}, x_{k,i,t-2}, \ldots, x_{k,i,t-P}$ and lagged market shares $M_{i,t-1}, M_{i,t-2}, \ldots, M_{i,t-P}$ as predictors. With these P lags, the attraction $A_{i,t}$ specification with a P-th order autoregressive structure becomes

$$A_{i,t} = \exp(\mu_i + \varepsilon_{i,t}) \prod_{j=1}^{I} \left(\prod_{k=1}^{K} x_{k,j,t}^{\beta_{k,j,i}} \prod_{p=1}^{P} \left(M_{j,t-p}^{\alpha_{p,j,i}} \prod_{k=1}^{K} x_{k,j,t-p}^{\beta_{p,k,j,i}} \right) \right), \tag{2.3}$$

where $\alpha_{p,j,i}$ is the effect of lagged market shares on the attraction and $\beta_{p,k,j,i}$ the effect of lagged explanatory variables. Clearly, this specification involves quite a number of parameters.

To estimate the parameters, the model is linearized in two steps. First, we choose brand I as a benchmark brand and express the market share of each of the remaining brands as a fraction of this benchmark brand, that is,

$$\frac{M_{i,t}}{M_{I,t}} = \frac{\exp(\mu_i + \varepsilon_{i,t}) \prod_{j=1}^{I} \left(\prod_{k=1}^{K} x_{k,j,t}^{\beta_{k,j,i}} \prod_{p=1}^{P} \left(M_{j,t-p}^{\alpha_{p,j,i}} \prod_{k=1}^{K} x_{k,j,t-p}^{\beta_{p,k,j,i}} \right) \right)}{\exp(\mu_I + \varepsilon_{I,t}) \prod_{j=1}^{I} \left(\prod_{k=1}^{K} x_{k,j,t}^{\beta_{k,j,I}} \prod_{p=1}^{P} \left(M_{j,t-p}^{\alpha_{p,j,I}} \prod_{k=1}^{K} x_{k,j,t-p}^{\beta_{p,k,j,I}} \right) \right)}. \tag{2.4}$$

The second step is to take the natural logarithm (denoted by log) of both sides of (2.4). Together, these two steps result in the $(I-1)$-dimensional set of equations given by

$$\begin{aligned}\log M_{i,t} - \log M_{I,t} &= (\mu_i - \mu_I) + \sum_{j=1}^{I}\sum_{k=1}^{K}(\beta_{k,j,i} - \beta_{k,j,I}) \log x_{k,j,t} \\ &+ \sum_{p=1}^{P}\sum_{j=1}^{I}(\alpha_{p,j,i} - \alpha_{p,j,I}) \log M_{j,t-p} \\ &+ \sum_{p=1}^{P}\sum_{j=1}^{I}\sum_{k=1}^{K}(\beta_{p,k,j,i} - \beta_{p,k,j,I}) \log x_{k,j,t-p} + \eta_{i,t} \quad (2.5)\end{aligned}$$

Because the μ_i parameters only appear as the difference $\mu_i - \mu_I$ with the benchmark parameter μ_I, they are not uniquely identified. However, the parameters $\tilde{\mu}_i = \mu_i - \mu_I$ are uniquely identified. Similarly, $\tilde{\beta}_{k,j,i} = \beta_{k,j,i} - \beta_{k,j,I}$, $\tilde{\beta}_{p,k,j,i} = \beta_{p,k,j,i} - \beta_{p,k,j,I}$, and $\tilde{\alpha}_{p,j,i} = \alpha_{p,j,i} - \alpha_{p,j,I}$ can also be uniquely identified. Therefore, for estimation we use $\tilde{\mu}_i, \tilde{\beta}_{k,j,i}, \tilde{\beta}_{p,k,j,i}$, and $\tilde{\alpha}_{p,j,i}$.

The errors $\eta_{i,t}$ in (2.5) are equal to $\eta_{i,t} = \varepsilon_{i,t} - \varepsilon_{I,t}$, or, equivalently, $\boldsymbol{\eta}_t = \mathbf{L}\boldsymbol{\varepsilon}_t$ with the $(I-1) \times I$ matrix $\mathbf{L} = [\mathbf{I} \mid -\mathbf{1}]$ where $\mathbf{I}$ an $(I-1)$-dimensional identity matrix and $\mathbf{1}$ is an $(I-1)$-vector of ones. Hence, given the earlier assumptions that $\boldsymbol{\varepsilon}_t$ is normally distributed with mean $\mathbf{0}$ and covariance matrix Σ, $\boldsymbol{\eta}_t$ is also normally distributed with mean $\mathbf{0}$ and a $(I-1) \times (I-1)$ covariance matrix equal to $\tilde{\Sigma} = \mathbf{L}\Sigma\mathbf{L}'$. As a consequence, out of the $I(I+1)/2$ unknown (co)variances in $\boldsymbol{\Sigma}$, we can only identify $I(I-1)/2$ values.

Using the substitution above to obtain unique estimates for the effects, the general attraction model in (2.5) can be expressed as an $(I-1)$-dimensional P-th order vector autoregression with exogenous variables, that is, by

$$\begin{aligned}\log M_{i,t} - \log M_{I,t} &= \tilde{\mu}_i + \sum_{j=1}^{I}\sum_{k=1}^{K} \tilde{\beta}_{k,j,i} \log x_{k,j,t} + \sum_{p=1}^{P}\sum_{j=1}^{I} \tilde{\alpha}_{p,j,i} \log M_{j,t-p} \\ &+ \sum_{p=1}^{P}\sum_{j=1}^{I}\sum_{k=1}^{K} \tilde{\beta}_{p,k,j,i} \log x_{k,j,t-p} + \eta_{i,t}. \quad (2.6)\end{aligned}$$

Under the assumption that the error variables are normally distributed with some unknown covariance matrix, maximum likelihood (ML) is the appropriate estimation method. In our application, the explanatory variables for each brand are the same, that is, $x_{k,1,t} = x_{k,2,t} = \ldots = x_{k,I,t}$. Under these conditions and if there are no parameter restrictions then the ordinary least squares (OLS) estimator is equal to the ML estimator (Fok et al., 2002).

If the dependent variable has not been drawn from a conditional normal distribution, then the parameters of the general Market Share Attraction Model (2.6) are not guaranteed to be optimally estimated by a least-squares technique (Vapnik, 1982). An alternative way to estimate the model parameters in this case is by means of the suggested SVR, which is outlined below.

2.3 Linear Support Vector Regression

Support Vector Regressions (SVRs) and Support Vector Machines (SVMs) are rooted in the Statistical Learning Theory, pioneered by Vapnik (1995) an coworkers. Detailed treatments of SVR and SVM can be found, for example, in Burges (1998), Smola (1996) and Smola and Schölkopf (1998). The following is a self-contained basic introduction to Support Vector Regressions (SVRs).

SVRs have two main strengths and these are good generalizability/avoidance of overfitting and robustness against outliers. Generalizability refers to the fact that SVRs are designed in such a way that they provide the most simple solution for a given, fixed amount of (training) errors. A solution is referred to as being simple if the coefficients of the predictor variables are penalized towards zero. Thus, an SVR addresses the problem of overfitting explicitly, just like many other penalization methods such as RR (Tikhonov, 1963) and Lasso (Tibshirani, 1996). The robustness property stems from considering absolute, instead of quadratic, values for the errors. As a consequence, the influence of outliers is less pronounced. More precisely, SVRs employ the so-called ϵ-insensitive error loss function, which is presented below. To put it in a nutshell, (linear) SVR departs from the classical regression in two aspects. The first one is the utilization of the ϵ-insensitive loss function instead of the quadratic one. The second aspect is the penalization of the vector of coefficients of the predictor variables.

The classical multiple regression has a well known loss function that is quadratic in the errors, $r_i^2 = (y - f(\mathbf{x}_i))^2$. The loss function employed in SVR is the ϵ-insensitive loss function

$$g(r_i) = |y_i - f(\mathbf{x}_i)|_\epsilon \equiv \max\{0, |y_i - f(\mathbf{x}_i)| - \epsilon\ \} = \max\{0, |r_i| - \epsilon\}$$

for a predetermined nonnegative ϵ, where y_i is the true target value, $\mathbf{x}_i$ is a vector of input variables and $f(\mathbf{x}_i)$ is the estimated target value for observation i. Figure 2.1 shows the resulting function for the residual. Intuitively speaking, if the absolute residual is off-target by ϵ or less, then there is no loss, that is, no penalty should be imposed, hence the name "ϵ-insensitive". However, if the opposite is true, that is $|y_i - f(\mathbf{x})| - \epsilon > 0$, then a certain amount of loss should be associated with the estimate. This loss rises linearly with the absolute difference between y and $f(\mathbf{x})$ above ϵ.

Because SVR is a nonparametric method, traditional parametric inferential statistical theory cannot be readily applied. Theoretical justifications for the SVR are instead based on statistical learning theory (Vapnik, 1995). There are two sets of model parameters in (linear) SVR: coefficients, and two manually-adjustable parameters – C and ϵ – that explicitly control the interplay between model fit and model complexity. For each value of the manually-adjustable parameters C and ϵ there is a corresponding set of optimal coefficients, which are obtained by solving a quadratic optimization problem. The C and ϵ parame-

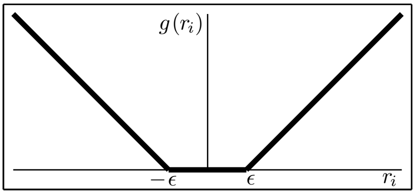


Figure 2.1: The ϵ-insensitive loss function that assigns no penalty to residuals $r_i \in [f(\mathbf{x}_i)-\epsilon, f(\mathbf{x}_i)+\epsilon]$ for point i. As $|r_i|$ gets larger than ϵ, a nonzero penalty $g(r_i)$ that rises linearly is assigned.

ters are usually tuned using a cross-validation procedure. In such a procedure, the data set is first partitioned into several mutually exclusive parts. Next, models are built on some parts of the data and other parts are used for evaluation of model performance for a particular choice of the fit-versus-complexity parameters C and ϵ. This is quite analogous to the process of adjusting the bias-versus-variance parameter in Ridge Regression, for instance. We start out the intuitive SVR exposition with assuming that C has implicitly been set to unity and ϵ has been set to 2. We later relax that assumption and give a more formal meaning of these parameters in terms of their role in the SVR optimization problem (2.7). In the nonlinear SVR case, other manually-adjustable parameters may arise. Then a cross-validation grid search over a certain range of values for C, ϵ and these parameters has to be performed in order to tune all parameters.

Let us first consider the case of simple linear regression estimation by SVR by the usual linear relation $y = \beta_1 x_1 + b$, where β_1 and b are parameters to be estimated. Figure 2.2 shows an example with three cases of possible linear functional relations. The SVR line is the solid line in Figure 2.2c, given by the equation $f(x_1) = \beta_1 x_1 + b$. The "tube" between the dotted lines in Figure 2.2 consists of points for which the inequality $|y - f(x_1)| - \epsilon \leq 0$ holds, where ϵ has been fixed arbitrarily at 2. All data points that happen to be on or inside the tubes are not associated with any loss. The rest of the points will be penalized according to the ϵ-insensitive loss function. Hence, the solutions in Panel (b) and (c) both have zero loss in ϵ-insensitive sense.

The exact position of the SVR line of Figure 2.2c is determined as follows. The starting point is that the SVR line should be as horizontal/simple/flat as possible. The extreme case of $\beta_1 = 0$ in Figure 2.2a will unavoidably yield several mistakes, as ϵ is not big enough to give zero loss for all points. This case represents a simple but quite "lousy" relationship. However, notice that the resulting region between the dotted lines, referred to as the ϵ-insensitive region, occupies the greatest possible area (for $\epsilon = 2$). It is argued in the SVR

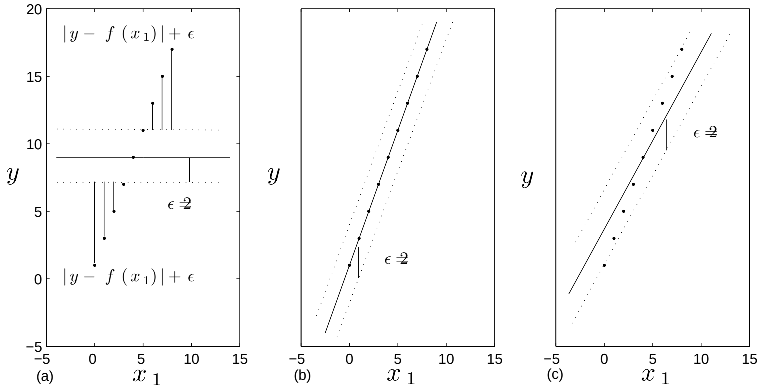


Figure 2.2: Three possible solutions to a linear regression problem with data points that lie on a line. The vertical line segments in panel (a) indicate loss per observation, which is equal to $|y - f(x_1)| - \epsilon$, for $\epsilon = 2$. In line with the ϵ-insensitive loss function, a point is not considered to induce an error if its deviation from the regression line is less than or equal to ϵ. The horizontal regression line in panel (a) is the simplest possible one since it hypothesizes that there is no relation between y and x_1, and it produces too much loss. Panel (b) gives the classical linear regression estimation, yielding zero loss. Panel (c) shows the linear SVR, which also yields zero loss but it flatter than the regression in Panel (b).

literature that this particular area can be seen as a measure of the complexity of the regression function used. Accordingly, the horizontal regression line provides the least complex functional relationship between x_1 and y, which is equivalent to no relationship at all.

Consider the next step in Figure 2.2b. Here, the solid line fits the training data extremely well. This line is the actual regression function from classical regression analysis, where the loss measured as the sum of squared errors of the estimates is being minimized. The distance between the dotted lines however has clearly diminished as compared to Figures 2.2a and 2.2c. What the SVR line of Figure 2.2c aims for is to find a balance between the amount of "flatness" (or *complexity*) and training mistakes (or *fit*). This balance is the fundamental idea behind SVR analysis. Good generalization ability is achieved when the best trade-off between function's complexity (proxied by the distance between the dotted lines) and function's accuracy on the training data is being struck. The idea that such a balance between complexity and amount of training errors should be searched has been formalized in Vapnik (1995).

To find a linear relationship between p independent variables and a single dependent variable in a data set of n observations, the mathematical formulation of the optimization problem of SVR can be derived intuitively as follows. The objective is to find a vector of p coefficients $\boldsymbol{\beta}$ and an intercept b so that the linear function $f(\mathbf{x}) = \boldsymbol{\beta}'\mathbf{x} + b$ has the best generalization ability for some fixed ϵ error insensitivity. From the "complexity" side, this linear surface should be as horizontal as possible, which can be achieved by minimizing the quadratic form $\boldsymbol{\beta}'\boldsymbol{\beta}$. From the "amount of errors" side however, a perfectly horizontal surface (obtained for $\boldsymbol{\beta} = \mathbf{0}$) will generally not be optimal since a lot of errors will typically be made in such a case. According to the ϵ-insensitive loss function, the sum of these errors is defined to be equal to $\sum_{i=1}^{n} g(r_i) = \sum_{i=1}^{n} \max\{0, |y_i - f(\mathbf{x}_i)| - \epsilon\}$. One can strike a balance between amount of errors and complexity by minimizing their sum

$$L_p(\boldsymbol{\beta}, b) \quad := \quad \frac{1}{2}\boldsymbol{\beta}'\boldsymbol{\beta} + C\sum_{i=1}^{n} \max\{0, |y_i - (\boldsymbol{\beta}'\mathbf{x}_i + b)| - \epsilon\}, \tag{2.7}$$

where C is a user-defined constant that controls the relative importance of the two terms. This minimization problem formulation is the familiar *penalty* plus *loss* minimization paradigm that arises in many domains (see, e.g., Hastie et al., 2001).

The problem can equivalently be represented by introducing the so-called slack variables $\boldsymbol{\xi}$ and $\boldsymbol{\xi}^*$. Then, minimizing $L_p(\boldsymbol{\beta}, b)$ can be represented as the constrained minimization problem

$$\begin{aligned} \text{minimize } L_p(\boldsymbol{\beta}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*) \quad &:= \quad \frac{1}{2}\boldsymbol{\beta}'\boldsymbol{\beta} + C\sum_{i=1}^{n}(\xi_i + \xi_i^*), \\ \text{subject to} \quad & \quad y_i - (\boldsymbol{\beta}'\mathbf{x}_i + b) \le \epsilon + \xi_i, \\ & \quad \boldsymbol{\beta}'\mathbf{x}_i + b - y_i \le \epsilon + \xi_i^*, \text{ and} \\ & \quad \xi_i, \xi_i^* \ge 0 \end{aligned} \tag{2.8}$$

(Vapnik, 1995; Smola & Schölkopf, 1998).

If the estimate $\boldsymbol{\beta}'\mathbf{x}_i + b$ of the i^{th} observation deviates from the target y_i by more than ϵ, then a loss is incurred. This loss is equal to either ξ_i or ξ_i^*, depending on which side of the regression surface observation i lies. It turns out that (2.8) is a convex quadratic optimization problem with linear constraints, and thus a unique solution can always be found. As already mentioned, the objective function in (2.8) consists of two terms. The first term, $\frac{1}{2}\boldsymbol{\beta}'\boldsymbol{\beta}$, captures the degree of complexity, which is proxied by the width of the ϵ-insensitive region between surfaces $y = \boldsymbol{\beta}'\mathbf{x} + b + \epsilon$ and $y = \boldsymbol{\beta}'\mathbf{x}_i + b - \epsilon$. If $\boldsymbol{\beta} = \mathbf{0}$, then complexity ($\frac{1}{2}\boldsymbol{\beta}'\boldsymbol{\beta}$) is minimal since the ϵ-insensitive region is biggest. The slack variables variables ξ_i and ξ_i^*, $i = 1, 2, \ldots, n$, are constrained to be nonnegative. All points i inside the ϵ-insensitive region have both $\xi_i = 0$ and $\xi_i^* = 0$. If a point i lies outside the ϵ-insensitive region, then either $\xi_i > 0$ and $\xi_i^* = 0$, or

$\xi_i = 0$ and $\xi_i^* > 0$. All data points that lie outside the ϵ-insensitive region (that is, for which $|y - f(\mathbf{x}_i| \geq \epsilon)$ are called "support vectors". It can be shown that the final solution for the SVR line depends only on the support vectors, and thus all other points are completely irrelevant (Smola & Schölkopf, 1998). This property is referred to as the sparse-solution property of SVR. In other words, the final formulation of the SVR function would remain the same even if all data points that are not support vectors were removed from the original data set.

Generally, it is not possible to have both terms $\frac{1}{2}\boldsymbol{\beta}'\boldsymbol{\beta}$ and $C\sum_{i=1}^{n}(\xi_i + \xi_i^*)$ equal to zero. If $\boldsymbol{\beta} = \mathbf{0}$, then the loss $C\sum_{i=1}^{n}(\xi_i + \xi_i^*)$ can be large, as depicted in Figure 2.2a. Likewise, if the sum $C\sum_{i=1}^{n}(\xi_i + \xi_i^*)$ is relatively small, then $\boldsymbol{\beta}$ will generally be large, and consequently $\frac{1}{2}\boldsymbol{\beta}'\boldsymbol{\beta}$ too. Therefore, at the minimum of the objective function in (2.8), a balance is found between $\frac{1}{2}\boldsymbol{\beta}'\boldsymbol{\beta}$ (complexity) and $C\sum_{i=1}^{n}(\xi_i + \xi_i^*)$ (fit), ensuring that neither the resulting function $f(x_1) = \boldsymbol{\beta}'\mathbf{x} + b$ fits the data too well, nor that it is too flat. The constraints in the optimization problem ensure that the degenerate solution $\boldsymbol{\beta} = \boldsymbol{\xi} = \boldsymbol{\xi}^* = \mathbf{0}$ is avoided.

2.4 Nonlinear Support Vector Regression

Another useful feature of the SVR is that nonlinear relationships can be easily included. This property may be useful in the Market Share Attraction Model if there is a nonlinear relation between the log attraction differences and the predictor variables.

2.4.1 Preliminaries

To introduce nonlinear regression solutions in the original (linear) space of the predictor variables $\mathbf{x}$, they are mapped to a higher-dimensional space via a mapping function $\mathbf{x} \rightarrow \Phi(\mathbf{x})$. Possible mappings are discussed in the next subsection. A mapping can be thought of as equivalent to extending the basis of the predictor space. As a result, optimization problem (2.8) becomes (Vapnik, 1995; Smola & Schölkopf, 1998):

$$\begin{aligned} \text{minimize } L_p(\boldsymbol{\beta}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*) \quad &:= \quad \frac{1}{2}\boldsymbol{\beta}'\boldsymbol{\beta} + C\sum_{i=1}^{n}(\xi_i + \xi_i^*), \\ \text{subject to} \quad & \quad y_i - (\boldsymbol{\beta}'\Phi(\mathbf{x}_i) + b) \leq \epsilon + \xi_i, \\ & \quad \boldsymbol{\beta}'\Phi(\mathbf{x}_i) + b - y_i \leq \epsilon + \xi_i^*, \text{ and} \\ & \quad \xi_i, \xi_i^* \geq 0. \end{aligned} \tag{2.9}$$

To cope with possibly infinite-dimensional nonlinear mappings and for greater computational efficiency, instead of the primal linear minimization problem defined in (2.9), its dual representation is used. The unknown parameters of the nonlinear SVR $\boldsymbol{\beta}$, b, ξ_i and $\xi_i^*, i = 1, 2, \ldots, n$ of the original primal (2.9) can be found as the unique solution of its dual counterpart,

$$\begin{aligned}\text{maximize } L_d(\boldsymbol{\alpha}) := \; & -\frac{1}{2}\sum_{i,j=1}^{n}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)(\Phi(\mathbf{x}_i)'\Phi(\mathbf{x}_j)) + \\ & + \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)y_i - \epsilon\sum_{i=1}^{n}(\alpha_i + \alpha_i^*) \\ \text{subject to } \; & 0 \leq \alpha_i, \alpha_i^* \leq C, i = 1, 2, \ldots, n \text{ and } \sum_{i=1}^{n}(\alpha_i - \alpha_i^*) = 0,\end{aligned}$$

where the unknowns α_i and α_i^* are the Lagrange multipliers of the primal. For a step-by-step derivation of the dual and the way to find the b parameter we refer to, for example, Vapnik (1995) and Smola (1996). The α_i and α_i^* are the weights associated with each data point i. If both α_i and α_i^* for point i are equal to zero, then this point lies inside the ϵ-insensitive region. It has a weight of zero and plays no role for the final formulation of the SVR function. Note that the specific case $\Phi(\mathbf{x}) = \mathbf{x}$ recovers the linear SVR of Section 2.3.

The SVR regression function takes the form of (Smola & Schölkopf, 1998):

$$f(\mathbf{x}) = \sum_{i=1}^{n}(\alpha_i^* - \alpha_i)(\Phi(\mathbf{x})'\Phi(\mathbf{x}_i)) + b, \tag{2.10}$$

where $\Phi(\mathbf{x})$ is a vector containing the mapped values of the independent variables for a new (test) point. Note that since the SVR regression function can be expressed as $f(\mathbf{x}) = \boldsymbol{\beta}'\Phi(\mathbf{x}) + b$, it follows that $\boldsymbol{\beta}'\Phi(\mathbf{x}) = \sum_{i=1}^{n}(\alpha_i^* - \alpha_i)(\Phi(\mathbf{x})'\Phi(\mathbf{x}_i))$ at the optimum, and therefore model coefficients are obtained as $\boldsymbol{\beta} = \sum_{i=1}^{n}(\alpha_i^* - \alpha_i)\Phi(\mathbf{x}_i)$.

2.4.2 Nonlinear SVR via Kernel Functions

We now consider nonlinear SVR estimation in greater detail. The construction of nonlinear SVR is carried out in two steps. First, the data are mapped through $\mathbf{x} \rightarrow \Phi(\mathbf{x})$ into a *higher*-dimensional space. Second, a linear regression function is constructed in the transformed space. This function corresponds to a nonlinear one in the original, non-transformed space. The optimal linear regression function in the transformed space should be, analogically to the non-transformed case, as flat as possible (Smola & Schölkopf, 1998) to ensure a good generalization ability. Due to the mapping $\mathbf{x} \rightarrow \Phi(\mathbf{x})$, the SVR estimates in the nonlinear case take the form (Smola & Schölkopf, 1998):

$$f(\mathbf{x}) = \sum_{i=1}^{n}(\alpha_i^* - \alpha_i)k(\mathbf{x}, \mathbf{x}_i) + b,$$

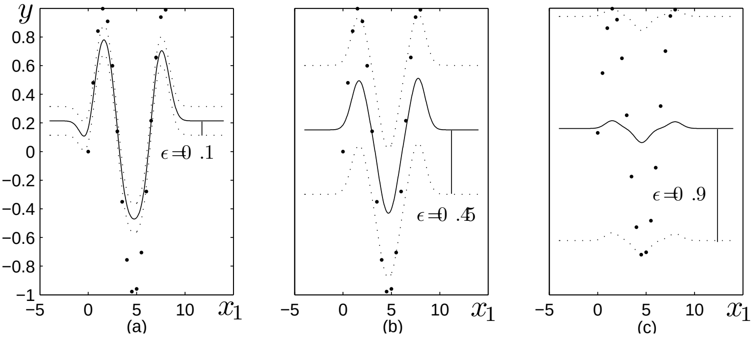


Figure 2.3: Three possible nonlinear SVR solutions to the problem of estimating the function $y = sin(x_1)$ from examples.

where $k(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i)'\Phi(\mathbf{x}_j)$ is a so-called kernel function that computes dot products in a transformed space. Often, the Gaussian kernel $k(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i)'\Phi(\mathbf{x}_j) = e^{-\gamma\|\mathbf{x}_i - \mathbf{x}_j\|^2}$ is used, where γ is a manually adjustable parameter that controls the degree of similarity between any two vectors in the transformed space. Note that coefficient estimates for nonlinear SVR are available only if the mapping $\mathbf{x} \longrightarrow \Phi(\mathbf{x})$ is carried out explicitly. The coefficients are then calculated as $\boldsymbol{\beta} = \sum_{i=1}^{n}(\alpha_i^* - \alpha_i)\Phi(\mathbf{x}_i)$. Other kernels exist but are beyond the scope of this paper.

Let us now consider Figure 2.3, which shows 17 sample points (black dots) from the function $y = sin(x_1)$. Three possible nonlinear SVR solutions to this problem are given in this figure. By construction there is no noise in the data. The nonlinear transformation of the original data is carried out via the Gaussian kernel. All SVR manually adjustable parameters are the same in all three panels, except for ϵ, which is equal to 0.1, 0.45 and 0.9 in panels (a), (b), and (c), respectively. As ϵ increases, the estimated functional relationship between y and x_1 becomes weaker (and therefore flatter); furthermore, the amount of errors reduces substantially. Notice that the estimated relationship also becomes flatter as x_1 takes on values that are farther away from the values of the original data points, which is an attractive property of SVR for extrapolation.

So far the question of how to choose the manually adjustable parameters (such as C, ϵ and γ) has been left aside. As mentioned in Section 2.3, one very common way to proceed is to use a k-fold cross-validation procedure. In a such a procedure, the data set is split in k (equally-sized) parts. Then, k models for a fixed set of values for the manually adjustable parameters are built on $k - 1$ folders and each time the one remaining folder is used for validation (or, testing). The chosen parameters are those that produce minimal mean squared error on average (over all k test parts).

For the sake of completeness, we should point out that an alternative specification of the standard ϵ-insensitive SVR exists, called ν-SVR, which minimizes the primal objective function (Schölkopf, Bartlett, Smola, & Williamson, 1998)

$$\frac{1}{2}\boldsymbol{\beta}'\boldsymbol{\beta} + C\left(\nu n\epsilon + \sum_{i=1}^{n} \max\{0, |y_i - (\boldsymbol{\beta}'\mathbf{x}_i + b)| - \epsilon\}\right)$$

instead of (2.7), where $0 \leq \nu \leq 1$ is specified a priori. The parameter ν is an upper bound on the fraction of points allowed to lie outside the tube that is asymptotically equal to the number of support vectors. Here, ϵ is treated as an unknown nonnegative parameter to optimize over and is therefore computed automatically.

Finally, we point that other (convex) loss functions could also be employed in (2.7), next to the standard ϵ-insensitive loss function, such as Laplacian, Huber, polynomial or piecewise polynomial (see, e.g., Smola & Schölkopf, 1998, 2004). Due to the convexity of these loss functions, the solution is unique.

2.4.3 Links between SVR and Classical Regression Analysis

The classical OLS approach to function estimation is to find the vector of coefficients $\boldsymbol{\beta} = \boldsymbol{\beta}^*$ and intercept term $b = b^*$, which minimize the loss $\mathrm{L}_{OLS}(\boldsymbol{\beta}, b) = \sum_{i=1}^{n}(y_i - \boldsymbol{\beta}'\mathbf{x}_i - b)^2$, where $\{y_i, \mathbf{x}_i\}$, $i = 1, 2, \ldots n$, is a data point. The RR approach extends OLS by modifying the loss to $L_{RR}(\boldsymbol{\beta}, b) = \lambda\boldsymbol{\beta}'\boldsymbol{\beta} + \sum_{i=1}^{n}(y_i - \boldsymbol{\beta}'\mathbf{x}_i - b)^2$, for $\lambda \geq 0$. Hence, the linear SVR, OLS, and RR optimization problems can be thought of special cases of the general optimization problem

$$\begin{aligned} \text{minimize } L_p^{\text{All}}(\boldsymbol{\beta}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*) \quad &:= \quad \frac{A}{2}\boldsymbol{\beta}'\boldsymbol{\beta} + \frac{C}{k}\sum_{i=1}^{n}((\xi_i)^k + (\xi_i^*)^k) \qquad (2.11) \\ \text{subject to} \quad & \quad y_i - \boldsymbol{\beta}'\mathbf{x}_i - b \leq \epsilon + (\xi_i^*)^k, \\ & \quad \boldsymbol{\beta}'\mathbf{x}_i + b - y_i \leq \epsilon + (\xi_i)^k, \text{ and} \\ & \quad \xi_i, \xi_i^* \geq 0, \\ & \quad \text{for } i = 1, 2, \ldots, n, \end{aligned}$$

where $\epsilon \geq 0, k \in \{1, 2\}, A \geq 0, C > 0$. The classical linear regression optimization problem is a special case of (2.11), where $k = 2$, $\epsilon = 0$, $C = 2$, and $A = 0$. The linear RR estimation problem is obtained for $k = 2$, $\epsilon = 0$, $C = 2$, and $A = 2\lambda$. Finally , the linear SVR estimation problem (2.8) corresponds to the restrictions $k = 1$, $\epsilon > 0$, $C > 0$, and $A = 1$.

Interestingly, linear RR can be extended to nonlinear RR via the introduction of kernel functions, akin to the transition from linear to nonlinear SVR. The resulting method is called Kernel Ridge Regression (KRR). KRR can be derived as follows. Starting from the primal minimization problem $L_{RR}(\boldsymbol{\beta}, b) = \lambda\boldsymbol{\beta}'\boldsymbol{\beta} + \sum_{i=1}^{n}(y_i - \boldsymbol{\beta}'\mathbf{x}_i - b)^2$, one introduces a mapping $\mathbf{x} \rightarrow \Phi(\mathbf{x})$, and constructs the equivalent dual (maximization) form of the nonlinear RR (minimization) optimization problem (see, e.g., Shawe-Taylor & Cristianini, 2004; Pozdnoukhov, 2002):

$$\begin{aligned} \text{maximize } L_{RRd}(\boldsymbol{\alpha}) := \; & -\frac{1}{4\lambda}\sum_{i,j=1}^{n}\alpha_i\alpha_j(\Phi(\mathbf{x}_i)'\Phi(\mathbf{x}_j)) + \sum_{i=1}^{n}\alpha_i y_i - \frac{1}{4}\sum_{i=1}^{n}\alpha_i^2 \\ \text{subject to: } & \sum_{i=1}^{n}\alpha_i = 0. \end{aligned}$$

The final step is to replace the dot product $\Phi(\mathbf{x}_i)'\Phi(\mathbf{x}_j)$ with a kernel function $k(\mathbf{x}_i, \mathbf{x}_j)$. The resulting regression function, expressed in dual form, is: $f(\mathbf{x}) = \sum_{i=1}^{n}(\alpha_i^* - \alpha_i)k(\mathbf{x}, \mathbf{x}_i) + b$.

2.5 An Illustration for New Cars

The technique of SVR might be particularly useful for the Market Share Attraction Model as it is not certain that the log of the market shares are conditionally Gaussian and also as the log transformation can create outlying data points. Here we present and compare the results of SVR and OLS estimation (which is equivalent to ML estimation with underlying Gaussian noise) of the coefficients of the Market Share Attraction Model on empirical data. Carrying out an extensive benchmark study is beyond the scope of the present paper and we refer to Pérez-Cruz et al. (2003) for a number of simulation studies. They report superior performance of SVR vis-a-vis ML coefficient estimation (with assumed Gaussian noise model) in cases where the dependent variable has not been drawn from a conditional normal distribution as well as in cases where the distribution is actually normal, but the sample size is small.

2.5.1 Description of the Data

The data are monthly sales figures per brand of new cars, in the Netherlands starting in January 1992 and ending in October 2001 obtained from Statistics Netherlands (CBS). Market shares are computed by dividing brand sales by total sales. There is a total of 36 different brands, one being 'Other' collecting all the smallest brands. The price series concerns the price of new cars. This price series is based on the best selling model per brand in a particular year. Note that we only have the prices of models for the 26 best selling brands.

The source is www.autoweek.nl. To find the price of that best selling model we consulted various annual editions of (Dutch language) Autoboek, Autovisie, and Autotest. The market shares are presented in the line plots of Figure 2.4.

2.5.2 Estimation of the Market Share Attraction Model

We now turn to the estimation of the (unrestricted) Market Share Attraction Model, applied to our data. We expect that the prices and market shares of each brand will have an effect on the market shares of all the other brands. In other words, we assume that the explanatory variables in the model are the same for each brand. For convenience, we denote with $x_{k,t}$ the k^{th} explanatory variable for any brand at time t, no matter whether it is in a lagged or other form, or it represents price or another explanatory variable. Thus, the attraction of brand i at time t, given in the general equation (2.1), becomes in our case

$$A_{i,t} = \exp(\mu_i + \varepsilon_{i,t}) \prod_{k=1}^{K} x_{k,t}^{\beta_{k,i}} \qquad \text{for } i = 1, \ldots, I, \tag{2.12}$$

with $k = 1, 2, \ldots, 88$, $t = 1, 2, \ldots, T$ and $I = 36$. The length of the time horizon T ranges from 50 to 117, since we study the evolution of market shares over time. For each T a separate Market Share Attraction Model for all brands is estimated. The first 26 explanatory variables are current prices; the next 26 variables are one month lagged prices, and the last stack of 36 variables are one month lagged market shares of all brands. Using brand I as a base brand, (2.1) translates into the market share equations for brands $1, 2, \ldots, I-1$ at time t (akin to (2.5))

$$\begin{array}{ccccccc} \log M_{1,t} - \log M_{I,t} & = & \tilde{\mu}_1 & + & \sum_{k=1}^{K} \tilde{\beta}_{k,1} z_{k,t} & + & \eta_{1,t} \\ \log M_{2,t} - \log M_{I,t} & = & \tilde{\mu}_2 & + & \sum_{k=1}^{K} \tilde{\beta}_{k,2} z_{k,t} & + & \eta_{2,t} \\ \vdots & = & \vdots & + & \vdots & + & \vdots \\ \log M_{I-1,t} - \log M_{I,t} & = & \tilde{\mu}_{I-1} & + & \sum_{k=1}^{K} \tilde{\beta}_{k,I-1} z_{k,t} & + & \eta_{I-1,t}, \end{array} \tag{2.13}$$

where $z_{k,t} = \log x_{k,t}$. For notational convenience, we denote $y_{i,t} = \log M_{i,t} - \log M_{I,t}$, $\mathbf{y}_i = (y_{i,1}, y_{i,2}, \ldots, y_{i,T})'$, $\tilde{\boldsymbol{\beta}}_i = (\tilde{\beta}_{1,i}, \tilde{\beta}_{2,i}, \ldots, \tilde{\beta}_{K,i})'$ and $\boldsymbol{\eta}_i = (\eta_{i,1}, \eta_{i,2}, \ldots, \eta_{i,T})'$. Further, we denote with $\mathbf{Z}$ the common matrix of independent variables for each brand over time $t = 1, 2, \ldots, T$. Consequently, (2.13) can be modeled in matrix form as

$$\begin{pmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \\ \vdots \\ \mathbf{y}_{I-1} \end{pmatrix} = \begin{pmatrix} \mathbf{Z} & 0 & \cdots & 0 \\ 0 & \mathbf{Z} & & \vdots \\ \vdots & & \ddots & \vdots \\ 0 & \cdots & 0 & \mathbf{Z} \end{pmatrix} \begin{pmatrix} \tilde{\boldsymbol{\beta}}_1 \\ \tilde{\boldsymbol{\beta}}_2 \\ \vdots \\ \tilde{\boldsymbol{\beta}}_{I-1} \end{pmatrix} + \begin{pmatrix} \boldsymbol{\eta}_1 \\ \boldsymbol{\eta}_2 \\ \vdots \\ \boldsymbol{\eta}_{I-1} \end{pmatrix}. \tag{2.14}$$

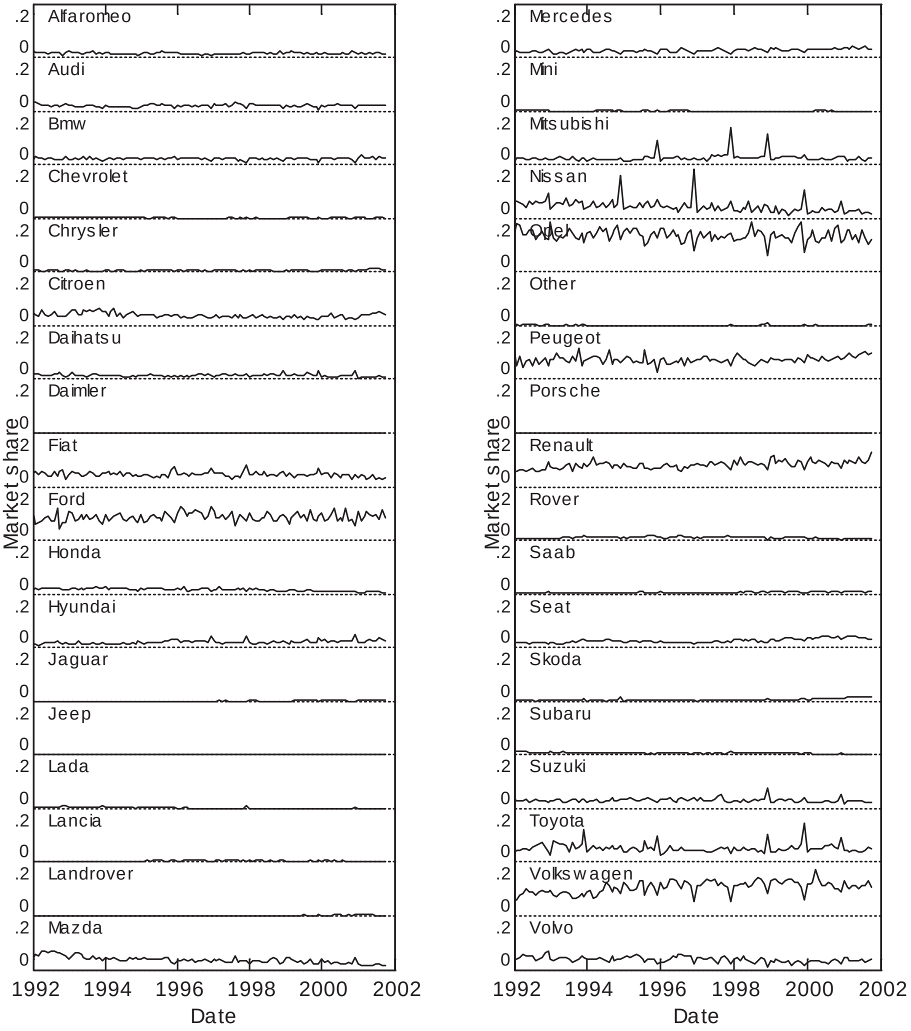


Figure 2.4: Market shares of 36 brands on the Dutch market between January 1992 and October 2001.

The coefficients of this model can now be estimated using OLS or SVR. For OLS, one estimates the model coefficients by minimizing the sum of squared errors, $\sum_{i=1}^{I-1}\sum_{t=1}^{T}\eta_{i,t}^2$. For SVR estimation, one minimizes the sum $0.5\sum_{i=1}^{I-1}\sum_{k=1}^{K}\tilde{\beta}_{k,i}+C\sum_{i=1}^{I-1}\sum_{t=1}^{T}\max\{0,|\eta_{i,t}|-\varepsilon\}$.

Because of the structure of the block diagonal matrix with blocks $\mathbf{Z}$, the OLS estimates can be computed very efficiently. The inverse $(\mathbf{Z}'\mathbf{Z})^{-1}$ only needs to be computed once and $\tilde{\boldsymbol{\beta}}_i = (\mathbf{Z}'\mathbf{Z})^{-1}\mathbf{Z}'\mathbf{y}_i$ contains the OLS optimal weights for brand i. In a similar way, the weights for the linear SVR problem can be estimated separately for each brand i. Computationally, this split will be much faster than inserting (2.14) directly into a linear SVR program. However, for nonlinear SVR, the problem cannot be split up into I smaller nonlinear SVR problems because its solution is defined in the dual where the nonlinearity is added to the full problem. Hence, splitting up the nonlinear SVR into smaller parts does not solve the full nonlinear SVR problem.

Although coefficient estimates for SVR are not always available in the nonlinear case (see Section 2.4), predicted values for the y's can always be created. Once all y's are obtained, say using values for the predictor variables at test time t^*, market shares can be derived using the relationship

$$\frac{e^{y_{i,t}}}{\sum_{i=1}^{I}e^{y_{i,t}}}=\frac{e^{(\log M_{i,t}-\log M_{I,t})}}{\sum_{i=1}^{I}e^{(\log M_{i,t}-\log M_{I,t})}}=\frac{M_{i,t}/M_{I,t}}{\sum_{i=1}^{I}M_{i,t}/M_{I,t}}=M_{i,t},$$

which uses the fact that the market shares sum up to unity.

2.5.3 Results

We estimated the coefficients of the Market Share Attraction Model given in (2.14) using both the SVR and OLS techniques. As indicated in Section 2.2, OLS is equivalent to ML estimation because (a) the dependent variable is assumed conditionally Gaussian, (b) the explanatory variables are the same for all brands, and (c) there are no parameter restrictions. The dependent variable is the log-ratio of market shares of 35 car brands and an arbitrary base brand, which we have chosen to be Volvo. The predictor variables include current prices, one period lagged prices, and one period lagged market shares. For SVR, we have used the linear SVR and the nonlinear SVR with the popular Radial Basis Function (RBF) kernel. We use an expanding window of historical in-sample data to produce a forecast for a given out-of-sample month. That is, we have estimated the market share model (2.14) 68 times, each time using slightly different, one-month-extended data. Thus, to forecast the first out-of-sample month March 1996, we use the first 50 months from January 1992 to February 1996. For each following out-of-sample month we add one month of historical data to the estimation window. In the end, we calculate the Root Mean Squared Prediction Error (RMSPE) and Mean Absolute Prediction Error (MAPE) per brand per month, where an error is defined as true brand market share minus predicted market share. Note that the market shares are always between 0 and 1.

For each of the 68 periods, we tested whether the assumption of OLS holds that the dependent variable is sampled from a Gaussian distribution. Therefore, we carried out two normality tests, that is, the large-sample Jarque-Bera and small-sample Lilliefors tests. Both tests rejected normality of the dependent variable at the 5% significance level for each of the 68 models for all samples. This result already suggests that SVR may perform better than OLS.

As noted in Section 2.4, SVR requires some parameters to be tuned, notably C and the width ϵ of the error-insensitivity region. In the case of RBF kernel, an additional γ parameter has to be tuned. The tuning is usually done via a grid parameter search, where each grid point is evaluated using a cross-validation procedure. In our case, we use a five-fold cross-validation procedure, which is carried out as follows. A given training data set is divided into five equal parts. A particular point on the grid is selected. It represents a tuple of values for the tuning parameters. Five models are trained, where each of the five parts is considered as an out-of-sample test set and the remaining four parts as a training set. The parameter combination that produces minimal squared error over the five test sets is then used to train the whole data set (consisting of all five parts) and produce an out-of-sample forecast for the following month.

The main results of the experiments are presented in Table 2.1. Overall, SVR outperforms OLS considerably and consistently in terms of RMSPE and MAPE over the 68-month out-of-sample period from March, 1996, to October, 2001. The average monthly RMSPE over all brands for the out-of-sample period is equal to 0.028839 for OLS. The corresponding figure for the linear SVR is 0.008466, and for SVR with RBF kernel the RMSPE is 0.008452. Figure 2.5 shows a detailed out-of-sample monthly RMSPE performance averaged over all brands. There are about 6 to 8 months that could visibly be considered as out-of-sample outliers, since both OLS and SVR perform relatively worse there. Clearly, OLS performs much worse, which suggests that SVR is capable of mitigating the effect of outliers and perform better in times of relative market distress.

Interestingly, both the linear SVR and the highly nonlinear SVR produce more or less the same prediction results, suggesting that there is not enough evidence in the data to favor a nonlinear relation among the dependent and independent variables. In addition, linear SVR has performed substantially better than OLS, suggesting that the robustness and penalization properties of SVR have worked out well on this particular market share prediction task. As demonstrated in Pérez-Cruz et al. (2003), factors working against OLS and in favor of SVR are the dependent variable not being sampled from a Gaussian distribution, the large amount of predictors relative to T, and the number of in-sample months.

Table 2.1: Performance results over 68 out-of-sample months (March 1996 – October 2001): Mean Absolute Prediction Error (MAPE) and Root Mean Squared Prediction Error (RMSPE) for OLS, linear SVR (Lin SVR), and nonlinear SVR with Radial Basis Function kernel (RBF SVR) models. MAPE and RMSPE represent the average of the average monthly errors over all 36 brands during the out-of-sample period.

	OLS	Lin SVR	RBF SVR	improvement of Lin SVR over OLS	improvement of RBF SVR over OLS
MAPE	0.012803	0.004882	0.004879	2.622 times	2.624 times
RMSPE	0.028839	0.008466	0.008452	3.406 times	3.412 times

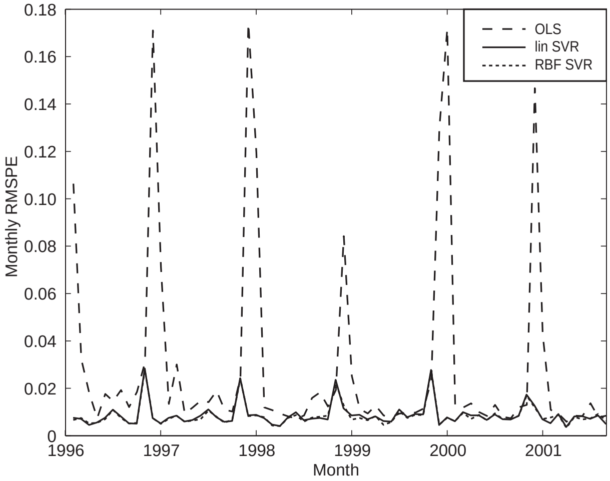


Figure 2.5: 68 monthly average RMSPE's over all brands for OLS, Linear SVR, and nonlinear RBF SVR.

It was noted above that RBF SVR gives similar forecasts to linear SVR. Adding nonlinearities to a model with a true linear relationship may lead to overfitting the training data and worse out-of-sample forecasts. In the case of RBF SVR, there is no danger of overfitting as the penalization and ϵ-insensitive loss function work in the direction of producing a linear relation, unless there is sufficient evidence in the data for the presence of nonlinearities, as argued also in Figure 2.1. What is more, Keerthi and Lin (2003) have demonstrated that there is no need to explicitly consider linear SVR since the RBF SVR is capable of discovering linear relations quite well.

Next we focus on the coefficient estimates produced by OLS and the linear SVR. Such estimates are not readily available for the nonlinear RBF SVR. As argued from the theoretical viewpoint in Sections 2.3 and 2.4, the estimated coefficients in SVR are shrunk towards zero vis-a-vis the corresponding OLS coefficients. This effect can be observed for our task as well. Figures 2.6 depict the effects of each of the 88 predictor variables on each of the 35 explained variables $\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_{I-1}$ for OLS (left column) and linear SVR (right column). The explanatory variables are divided into three groups: the first group consists of current prices, the second of the lagged prices, and the third group of lagged market shares. Each of these effects does not stand for one particular period T. Rather, it represents the average value over the 68-month out-of-sample period. The filled circles represent the average effect over the 68-month out-of-sample period of the predictor variables on $\mathbf{y}_{10}$, the log-difference of the market shares between brand 10 (Ford) and the base brand (Volvo). A key observation to make here is the striking difference between the magnitude of the OLS and linear SVR coefficients in general.

To visualize the influence of particular predictors, the sign of the effect is of less importance than the size. Therefore, we also present the absolute values of the effects of each predictor variable on $\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_{I-1}$ in Figure 2.7. For each predictor variable, the sum of absolute effects on $\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_{I-1}$ is depicted. Thus, the number of different shades of gray is $I - 1$ (or, 35 in our case). This representation allows us to observe whether OLS and linear SVR consider the same variables to be influential. For example, consider the 26 current-price variables. A striking feature, that is not easily observable from Figure 2.6, is that the variables that appear to play a key role in OLS estimation have also a relatively large impact in linear SVR estimation, most notably prices of Fiat, Ford, Mercedes, Renault, Seat, Subaru, and Volvo. In linear SVR, there are some additional variables that stand out as important: prices of Alfa Romeo, Citroen, Daihatsu, and others. Overall, the linear SVR coefficients have much lower magnitude in absolute sense, and are more evenly spread than the corresponding OLS coefficients.

To see how the three models differ in predicting the market shares, we have plotted the prediction errors for each brand. Figure 2.8 shows these plots for OLS and Figure 2.9 for linear SVR. We do not present a representation of

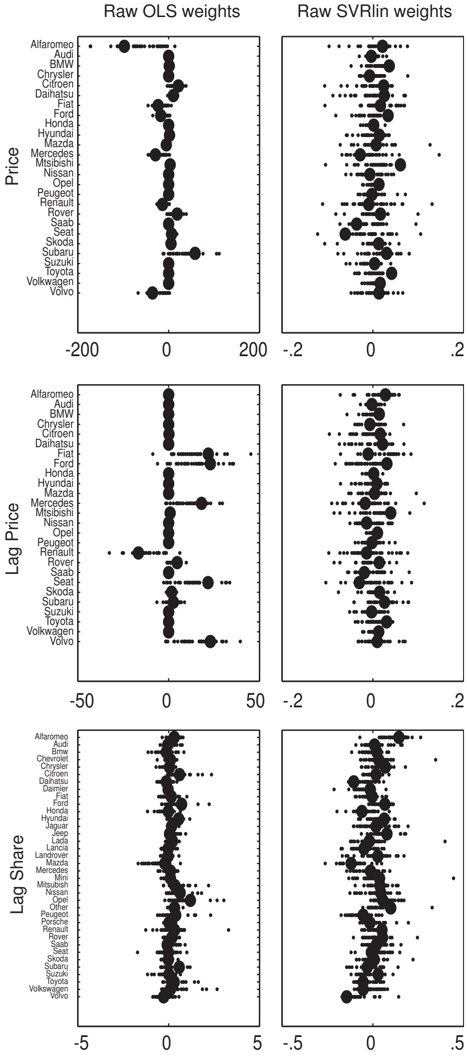


Figure 2.6: Regression weights (small dots) of the predictor variables obtained by OLS and linear SVR for each of the 35 explained variables on average, where the averaging is done over the 68 out-of-sample periods. The circles represent the average (over time) effect of all predictor variables on a concrete explanatory variable: the log-difference of the market shares of Ford and the base brand Volvo.

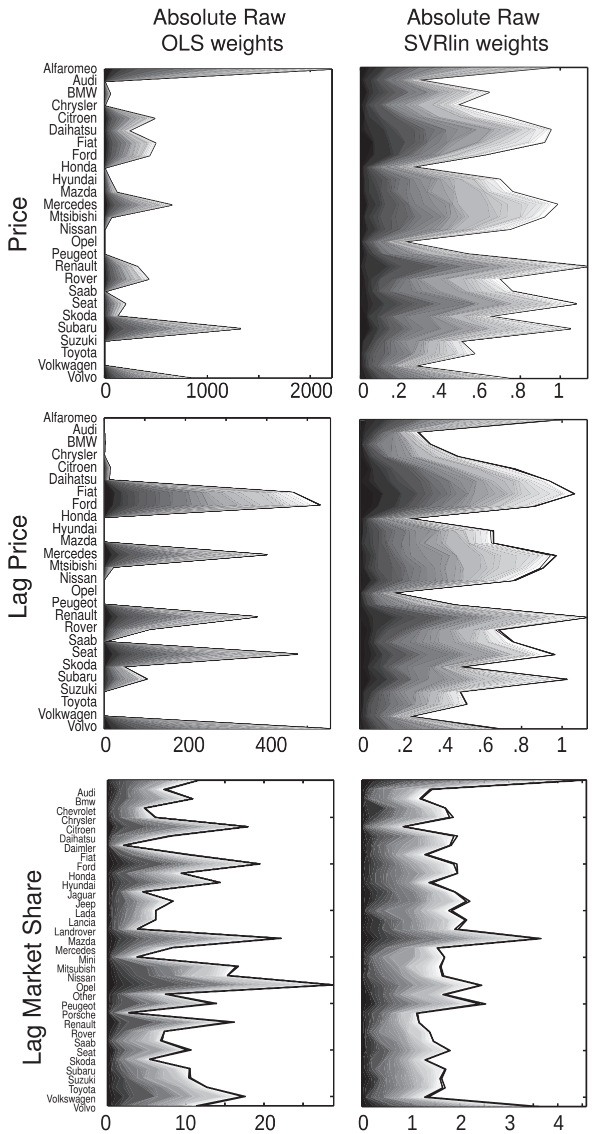


Figure 2.7: Distribution of absolute regression weights of the variables obtained by OLS and linear SVR for each of the 68 estimation periods. The darkness of the area indicates the density of the absolute regression weights.

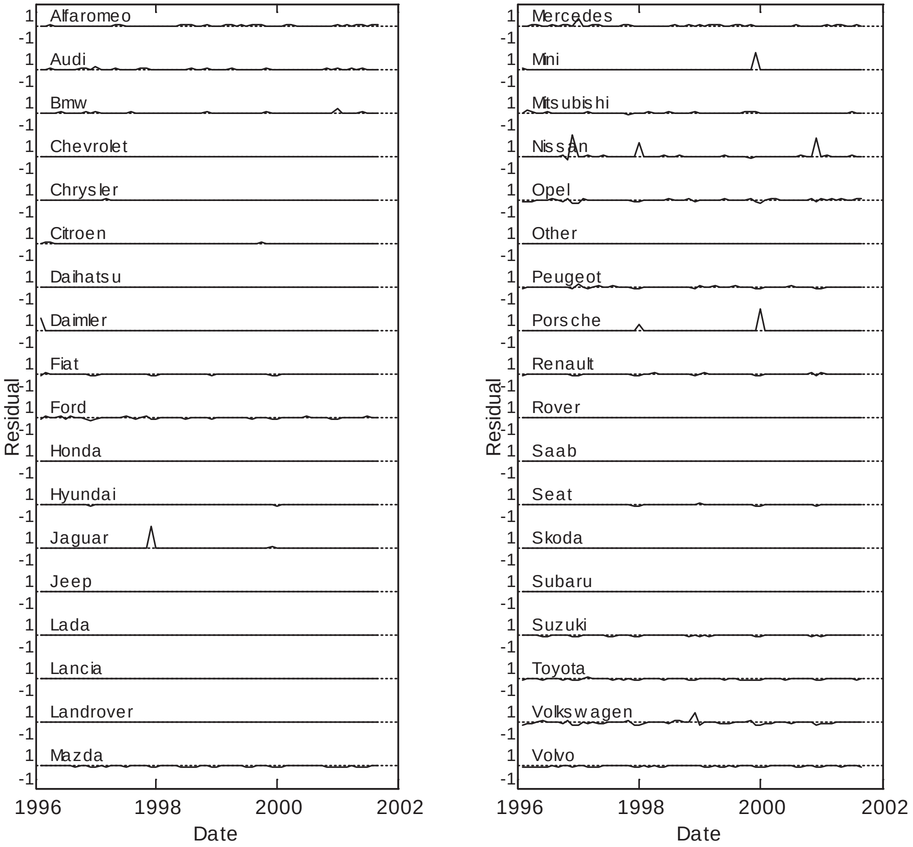


Figure 2.8: Out of sample prediction error of OLS estimation.

the errors for the nonlinear RBF SVR because there is hardly any difference with the errors of linear SVR. The most striking feature again is that OLS has several large errors whereas the largest errors of linear SVR are at least a factor 5 smaller. The error plots can be interpreted for each brand separately. To consider one case of the residual plot for linear SVR, Volkswagen had around January 1999 and again around January 2000 a positive error, whereas Toyota had simultaneously two spikes of negative errors. As a consequence, the linear SVR model apparently has underestimated the market share of Volkswagen and overestimated the market share of Toyota at these time points.

We now turn to the comparison of OLS, linear and nonlinear SVR to that of linear and Kernel Ridge Regression. As outlined in the introduction and Section

Table 2.2: Bootstrap p-values of pairwise tests for superior predictive ability (Hansen, 2005) over two performance criteria comparing OLS, linear SVR (Lin SVR), linear Ridge Regression (Lin RR), and nonlinear SVR and Kernel Ridge Regression with Radial Basis Function kernel (RBF SVR and RBF KRR). Three values for the q randomization parameter have been used in the tests: 0.1, 0.5 and 0.9. The estimation (out-of-sample) period is March 1996 – October 2001. The upper part of the table shows the results for the Mean Absolute Prediction Error (MAPE) performance criterion and the lower part for the Root Mean Squared Prediction Error (RMSPE) criterion. RMSPE (MAPE) represents the average over 68 months of the average over 36 brands monthly RMSPE's (MAPE's) during the estimation period.

	OLS	Lin SVR	RBF SVR	Lin RR	RBF KRR
MAPE	0.012803	0.004883	0.004879	0.004871	0.004868
Benchmark			Competitor		
$q = 0.1$					
OLS	N/A	0	0	0	0
Lin SVR	x	N/A	0.46291	0.37965	0.32378
RBF SVR	x	x	N/A	0.43005	0.40736
Lin RR	x	x	x	N/A	0.37254
$q = 0.5$					
OLS	N/A	0.00007	0.00009	0.00007	0.00008
Lin SVR	x	N/A	0.4759	0.41877	0.39167
RBF SVR	x	x	N/A	0.44072	0.42857
Lin RR	x	x	x	N/A	0.34727
$q = 0.9$					
OLS	N/A	0.00004	0.00002	0.00003	0.00002
Lin SVR	x	N/A	0.47851	0.42097	0.40019
RBF SVR	x	x	N/A	0.44838	0.43117
Lin RR	x	x	x	N/A	0.33249
RMSPE	0.028839	0.008466	0.008452	0.008271	0.008257
Benchmark			Competitor		
$q = 0.1$					
OLS	N/A	0	0	0	0
Lin SVR	x	N/A	0.41851	0.00062	0.00042
RBF SVR	x	x	N/A	0.00503	0.00635
Lin RR	x	x	x	N/A	0.22524
$q = 0.5$					
OLS	N/A	0.00045	0.00038	0.0004	0.00042
Lin SVR	x	N/A	0.43363	0.02285	0.01822
RBF SVR	x	x	N/A	0.04348	0.03435
Lin RR	x	x	x	N/A	0.17145
$q = 0.9$					
OLS	N/A	0.00021	0.00028	0.00027	0.00018
Lin SVR	x	N/A	0.43848	0.03456	0.029
RBF SVR	x	x	N/A	0.04866	0.03931
Lin RR	x	x	x	N/A	0.15242

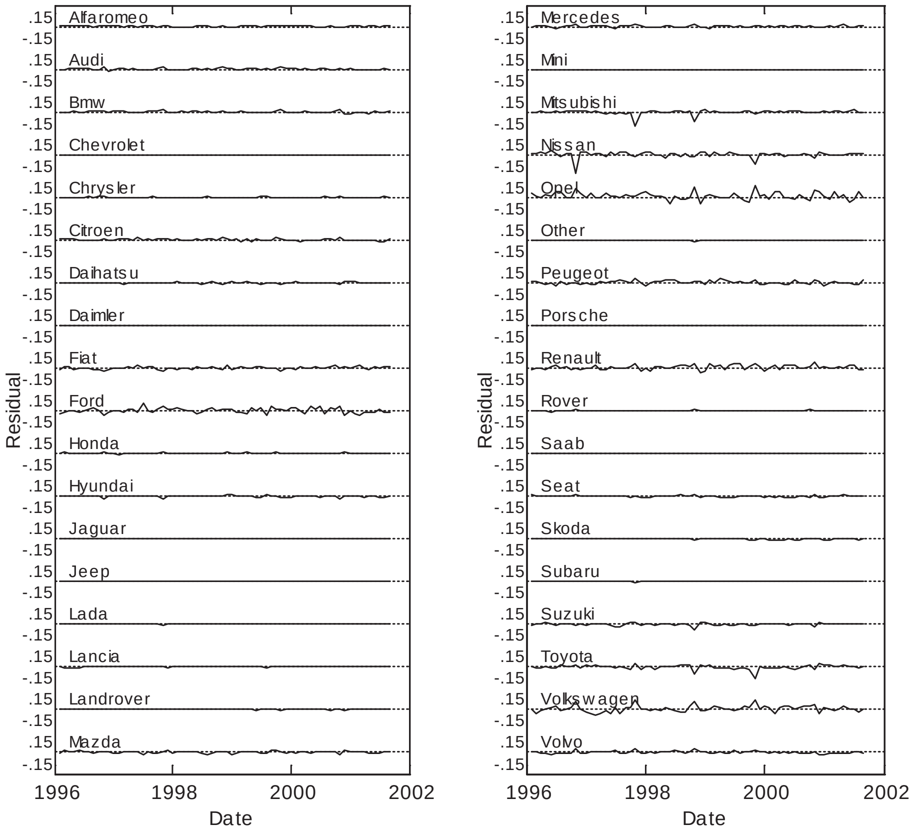


Figure 2.9: Out of sample prediction error of linear SVR estimation.

(2.4.3), RR and KRR are good competitors to linear and nonlinear SVR since all of these models share common features. In particular, these features are the penalization of the L_2 norm of the estimated coefficients and the ability to employ kernel functions to produce nonlinear regression surfaces.

As in SVR, RR and KRR require tuning of some model parameters. For RR, there is only one parameter, λ. For KRR, we use the RBF kernel, and therefore one more parameter has to be tuned – the RBF parameter γ. This tuning was performed using a five-fold cross-validation validation procedure, as in the SVR case. Before commenting on the significance of the performance differences, we stress that although the number of folds in the cross-validation procedure is equal for both SVR and KRR (in the linear and nonlinear cases), the grid for tuning the parameters is different. This grid is rougher for SVR,

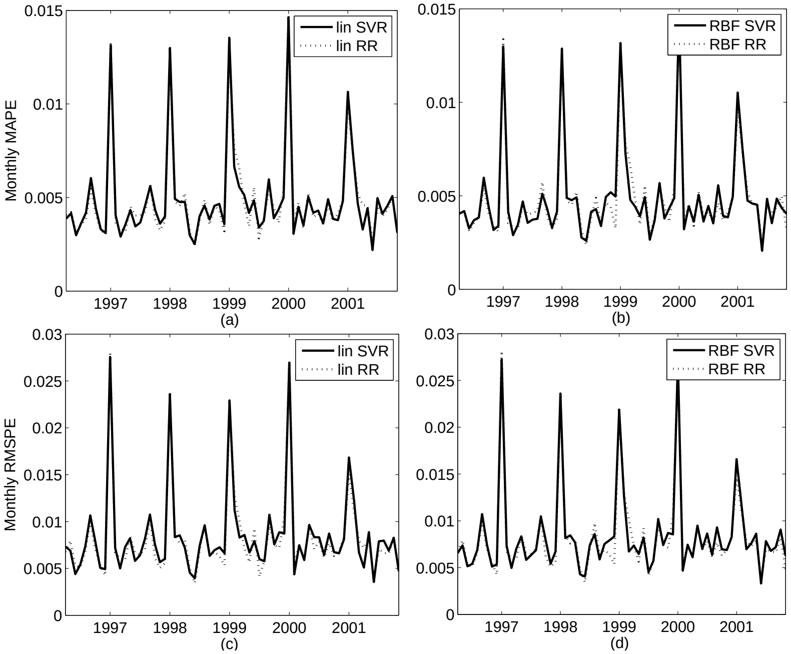


Figure 2.10: Monthly performance results in terms of RMSPE and MAPE for linear SVR and RR (panels (a) and (c)), and for RBF SVR and RBF KRR (panels (b) and (d)).

since estimation takes much longer, mostly due to the need to tune one more parameter, the width of the tube ϵ.

The two top panels of Figure 2.10 depict monthly performance results in terms of RMSPE for the linear SVR and linear RR, and for RBF SVR and RBF KRR, respectively. The two bottom panels of Figure 2.10 depict monthly performance results in terms of MAPE for the linear SVR and linear RR, and for RBF SVR and RBF KRR, respectively. What seems evident is that the majority of the good-performing and bad-performing months coincides in all four cases.

To test whether some of the applied techniques perform significantly better than others, we carry out pairwise tests for superior predictor ability, proposed by Hansen (2005), who proposed a bootstrap test, carried out essentially as follows. The input for the test are k vectors of relative performances over a benchmark model. For pairwise testing, as in our case, $k = 1$, which we assume from now on. Subsequently, B number of bootstrap samples of this vector are created, and each time the overall relative performance is recalculated. We use overall RMSPE and MAPE as performance measures and choose $B = 100000$.

The number of times that the benchmark model outperforms the contender out of B trials is the p-value of the test. There is one important detail. The bootstrap samples are not created in a completely random way, that is, once an initial point in the vector has been chosen at random, there is a specified user-defined chance that the following point will be chosen. This chance is governed by a so-called q parameter. Table 2.2 shows the p-values of the pairwise relative performance tests among all five estimation models for three different values of the q parameter.

In all three cases the results appear to be quite similar: OLS is significantly outperformed by all the other methods; linear SVR and RBF SVR perform similarly; linear RR and RBF KRR perform similarly; and SVR and (K)RR are closer to each other more in terms of MAPE rather than RMSPE. The differences in the last case may well be due to differences in the roughness of the grid over which parameter tuning has been performed. The similarity between SVR and KRR (both linear and nonlinear forms) is striking. These suggests that for our data set the penalization of model coefficients may play a decisive regularization role, rendering the further utilization of the ϵ-insensitive loss function unnecessary.

2.6 Conclusion

The Market Share Attraction Model may suffer from estimation problems when the model assumptions are not satisfied. In this paper, we have introduced SVR as an alternative estimation procedure for this model. An intuitive and self-contained introduction to SVR was provided. To test the estimation procedures, we compared OLS to linear and nonlinear SVR to empirical market share data of the Dutch automobile sales of 36 brands. It was found that the prediction by either linear or nonlinear SVR was much better than OLS. There was hardly any difference between linear and nonlinear SVR indicating that for these data it is not necessary to allow for a nonlinear prediction.

In addition, we carried out pairwise tests for superior predictive ability (proposed by Hansen, 2005) between OLS, linear and nonlinear SVR, and linear and nonlinear RR. The tests revealed that all penalization methods outperform the base OLS model in terms of RMSPE and MAPE. There is not much evidence however for differences among these methods, with the exception maybe of KRR versus SVR (both linear and nonlinear cases), when the object of comparison is RMSPE. Our conclusion is that for our data set the penalization of model coefficients is a quite effective departure from OLS estimation. The other two further departures – the introduction of nonlinearity (via the RBF kernel) and the use of a more robust loss function – do not account for a further significant improvement. These claims however are based on one data set only and cannot be generalized on the basis this data set only.

There are some remaining issues for SVR. For example, is there an optimal kernel function for the Market Share Attraction Model or is the linear SVR sufficient? What is the best procedure for tuning the manually-adjustable parameters in the SVR? Other issues are how to compute confidence intervals for the predictions and how to interpret the parameters of the nonlinear SVR model.

Clearly, more experiments have to be carried out to confirm or refute more convincingly the applicability of SVR in marketing tasks and as a competitor to traditional estimation techniques. Our empirical comparison suggests that when the OLS assumption of normality of the errors in the Market Share Attraction Model is not satisfied, SVR is a good alternative to estimate its parameters.

Chapter 3

Equity Style Timing using Support Vector Regressions*

The disappointing performance of value and small cap strategies shows that style consistency may not provide the long-term benefits often assumed in the literature. In this study we examine whether the short-term variation in the U.S. size and value premium is predictable. We document style-timing strategies based on technical and (macro-)economic predictors using a recently developed artificial intelligence tool called Support Vector Regressions (SVR). SVR are known for their ability to tackle the standard problem of overfitting, especially in multivariate settings. Our findings indicate that both premiums are predictable under fair levels of transaction costs and various forecasting horizons.

3.1 Introduction

There is no doubt about the importance of investment styles in modern portfolio management. The underlying rationale for this relates to a series of influential studies documenting the potential benefits of investing in stocks with fundamental commonalities or "styles". In the past two decades, substantial evidence surfaced suggesting that investing in portfolios of stocks with a small market capitalization and value orientation provides a premium in the long run. The "size premium" has been first reported by Banz (1981), who found a negative relation between a firm's market capitalization and its stock performance in the U.S. The extensively researched "value premium" has been documented most prominently by Fama and French (1992) and Fama and French (1998) and

*This chapter has been published as Nalbantov et al. (2006):
Nalbantov, G., Bauer, R., & Sprinkhuizen-Kuyper, I. (2006). Equity Style Timing using Support Vector Regressions. *Applied Financial Economics*, *16*(15), 1095–1111.

Lakonishok, Schleifer, and Vishny (1994). These studies showed that stocks with typical value features such as low market-to-book (M/B), low price-to-earnings (P/E) and low price-to-cash (P/C) ratios provided higher average returns than so-called "growth" stocks, with high M/B, P/E and P/C ratios. These empirical findings induced a discussion on the source and magnitude of the value and size premium. Some studies argued that this premium is a compensation for holding stocks under relative distress, see for instance Chan and Chen (1991) and Fama and French (1993). Another view, put forward in Lakonishok et al. (1994) and Haugen and Baker (1996), is that stock markets lack efficient pricing ability. A third possible explanation suggested in Lo and MacKinlay (1990) is that the obtained results are due to data snooping biases. A recent review and update for the U.S. by Chan and Lakonishok (2004) shows that value investing still generates promising returns in the long run. Dimson, Nagel, and Quigley (2003) arrive at similar conclusions for the U.K. value premium.

The rather disappointing performance of small cap and value strategies during the nineties has however pointed out that style consistency may not necessarily provide superior returns in any economic regime. A relatively small body of literature has explicitly addressed the potential benefits of style timing strategies over a style consistent approach. Although most of these papers may differ in methodology, they all rely on the notion that the cyclical behavior of investment styles is correlated with systematic economic and technical forces, which could make the value and size premium partially predictable. Cooper, Gulen, and Vassalou (2001) find sufficient predictability for size-sorted strategies in the U.S., but weaker results for value-sorted strategies.[1] Levis and Liodakis (1999) find moderate evidence in favor of small/large rotation strategies, but less evidence for value/growth rotation in the United Kingdom. Bauer, Derwall, and Molenaar (2004) find evidence for the profitability of style rotation strategies in Japan, but point out that moderate levels of transaction costs can already make these results less interesting in a practical context. The majority of rotation studies employ technical (or market-based) and (macro-)economic indicators. The dependent variables, either the value or the size premium, are constructed using well-known style index series.

In this study we will use a similar approach by constructing the value and size premium in the U.S. based on S&P style indices. The sign and magnitude of both premiums will then subsequently be forecasted using a broad set of (macro-)economic and technical predictors. In contrast to the studies mentioned above, we will not apply a standard multifactor model framework. Factor models in general suffer from deficiencies intrinsic to multiple regression techniques.

[1] Other related work includes Ahmed, Lockwood, and Nanda (2002), Arnott, Dorian, and Macedo (1992), Arnott, Rice, Kelso, Kiscadden, and Macedo (1989), Asness, Friedman, Krail, and Liew (2000), Copeland and Copeland (1999), Elfakhani (2000), Jacobs and Levy (1996), Kao and Shumaker (1999), Lucas, Van Dijk, and Kloek (2002), Mills and Jordanov (2003), and Mun, Kish, and Vasconcello (2001).

Most of the studies based on this methodology ex ante decide to construct parsimonious models to avoid the problem of overfitting. Increasing the number of factors at some point will deteriorate the out-of-sample prediction ability of the rotation models. Levis and Liodakis (1999) for instance report empirical results based on six factors for the size spread and eight factors for the value spread. Although their regression window is expanding, thereby updating the relevance of the factors through time, it does not provide the ability to add or delete economically viable factors. In most cases the "optimal" choice of independent variables is based on a set of statistical criteria, like adjusted R^2, the Akaike information criterion or the Schwarz criterion. These criteria are designed to correct the inclusion of factors for the increased model complexity. Potentially, numerous relevant variables are bound to be excluded as predictors.

A further complication arises from the fact that individual factors in a model are usually assumed to be independent. Most linear regression models however are likely to suffer from multi-collinearity, especially when the forecasting variables are numerous and closely related. One could therefore argue that factor models face two pivotal challenges: first, how to employ a large set of potentially relevant variables in a factor model without jeopardizing its predictive power, and second, how to incorporate possible interactions between individual variables in the course of the model-building process without deteriorating the quality of the model.

Support Vector Regressions (SVR) have become a popular analytical tool following a series of successful applications in fields ranging from optical character recognition to DNA analysis (Müller, Mika, Rätsch, Tsuda, & Schölkopf, 2001; Smola & Schölkopf, 1998). In essence, the SVR technique is used for function estimation based on a finite number of observations, just like the linear multiple-regression technique. Numerous potential applications of SVR in finance have been reported elsewhere.[2] The combination of three key features can justify a priori the utilization of the SVR tool in financial forecasting modeling. First, SVR behave robustly even in high-dimensional feature problems (Maragoudakis, Kermanidis, Fakotakis, & Kokkinakis, 2002), or in other words, where the explanatory variables are numerous, and in noisy, complex domains (Burbidge & Buxton, 2001). Second, SVR achieve remarkable generalization ability by striking a balance between a certain level of model accuracy on a given training data-set, and model complexity.[3] And third, SVR always find a global solution to a given problem (Smola, 1996; Vapnik, 1995), in sharp contrast with neural networks for instance. A general limitation of SVR is that they produce point estimates rather than posterior probability distributions of the obtained results, which follows from the fact that SVR are a nonparametric tool.

[2] See, e.g. Monteiro (2001), Müller et al. (1997), Rocco S. and Moreno (2003), Smola and Schölkopf (1998), and Van Gestel, Baesens, Garcia, and Van Dijcke (2003).

[3] Note that in real-world applications the presence of noise in regression estimation necessitates the search for such a balance, see Vapnik (1995) and Woodford (2001).

Some parameters however have to be estimated in advance via a standard procedure called “cross-validation”. This procedure, though quite computationally extensive, additionally ensures that model selection is based on out-of-sample rather than in-sample performance.

Using SVR we will construct models in order to predict the value and size premiums in the U.S. stock market. Our aim is to test on a preliminary level the performance of SVR, and not to engage in an extensive data-mining exercise. For that reason, we build our models on historical data of 60 months, which is a quite common horizon in the literature. Obviously, other model-building horizons can be explored, but in such a way artificially good results could emerge, falling pray to the data-mining critique. We compare the results of the rotation strategies with so-called style consistent passive strategies. Furthermore, we vary the forecast horizon (one-, three- and six-month signals), which serves as a model-stability test, and measure the impact of a wide range of transaction costs. The empirical section shows that style rotation strategies using signals created by SVR produce outstanding results for both the value and the size premiums.

The remainder of the chapter is organized as follows. Section 3.2 discusses the choice of explanatory variables and the nature of the explained variables (the proxies for the value and size premiums). Section 3.3 deals extensively with Support Vector Regressions as an analytical tool and how it can be used to predict the value and size premiums. Section 3.4 presents our main empirical findings, and Section 3.5 concludes.

3.2 Data

3.2.1 Construction of the Value and Size Premium Series

The choice of an appropriate measure to determine the value premium is crucial. Our main goal is to come up with a trading strategy, which can be easily implemented in a practical context.[4] In principle, long time series data from the Center for Research in Security Prices (CRSP) can be used. Following this venue is not well suited for a low transaction cost strategy however, since there are no readily-available instruments (e.g. futures) to exercise such a trading strategy in practice. As we expect the rotation strategies to have a considerable turnover, we conduct our analysis on the S&P Barra Value and Growth indices (the value premium). Transaction costs are expected to be relatively low as we are able to buy and sell futures on these indices.[5] Both indices are

[4] In the case of for instance the High book-to-market minus Low book-to-market (HML) series of Fama and French (1993), we can expect relatively high transaction costs as portfolios generally exhibit unacceptable liquidity features, particularly in a monthly long/short setting.

[5] In practice the maximum exposure of the trading strategy is still restricted by the liquidity features of this future.

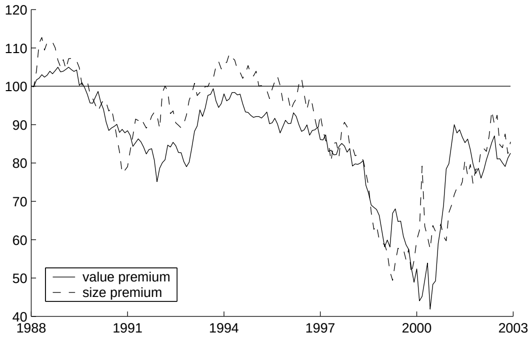


Figure 3.1: Cumulative performance of the value and size premiums (1988:01–2002:12).

constructed by dividing the stocks in the S&P 500 index according to just one single attribute: the book-to-market ratio. This procedure splits the S&P 500 index into two, mutually exclusive groups of stocks and is designed to track these accepted investment styles in the U.S. stock market. The Value index contains firms with high book-to-market ratios and conversely the Growth index firms with lower book-to-market ratios. The combination of both (market cap weighted) indices adds up to the (market cap weighted) S&P 500.

Figure 3.1 and table 3.1 show that a strategy purely based on the value premium would have witnessed some highly volatile periods. These series are the returns of a long position in the Value index and a short position in the Growth index throughout the entire sample period ranging from January 1988 to December 2002. Monthly maximum and minimum returns of this strategy are considerably high: 9.74% and -12.02%. Summary statistics (see table 3.1) reveal that the spread series exhibits excess kurtosis. The number of negative performance months of this passive value strategy is approximately 47%. The average return on an annualized basis is -0.86% with a standard deviation of 9.64%. We therefore conclude that pure and unconditional value investing in this particular sample period has not been a very attractive trading strategy. Furthermore, we indeed observe a cyclical pattern in the behavior of the premium. In some periods, like for instance in the last years of the previous decade, growth stocks persistently outperformed value stocks and in other periods value stocks clearly outperformed growth stocks. A good example of the latter is the crisis in Technology (and hence “growth”) stocks in the beginning of this century. A possible explanation for this phenomenon could be that the sign of the value premium is strongly connected with the business cycle and the economic regime. It is likely that value stocks — relative to growth stocks — gain more

Table 3.1: Summary statistics for the value and size premiums (1988:01 - 2002:12). All numbers are annual data (in %) unless stated otherwise. The spread series for the value premium are computed as returns of a long/short portfolio (long S&P Barra Value index and short S&P Barra Growth index). The spread series for the Size premium are computed as returns of a long/short portfolio (long S&P 500 index and short S&P SmallCap 600 index). Prior to the introduction of the S&P SmallCap 600 index in January 1994, the Frank Russell 1000 and Frank Russell 2000 indices have been used as inputs for the small-large calculations.

	Value premium	Size premium
Mean	-0.86	-0.91
Standard deviation	9.64	12.04
Information ratio	-0.09	-0.08
Minimum (monthly)	-12.02	-15.71
Maximum (monthly)	9.74	16.78
Skewness (monthly)	0.06	0.27
Excess kurtosis (monthly)	3.15	4.31
% negative months	47.22	51.11

from a surge of economic activity and a sharp upward revision of sentiment, see e.g. Schwob (2000). As profit expectations turn sharply and broadly positive at the bottom of the economic cycle, profitability and earnings growth become a less scarce resource. In such an environment portfolio managers start looking for stocks with typical value features. This largely explains why value stocks generally belong to cyclical industries. Moreover, value companies tend to belong to mature sectors of the economy. These sectors generally grow and shrink with the economy, whereas growth companies can offer protection during weaker periods in the economy.

Analogously, the size premium series is created by comparing the S&P 500 index (large cap) and the S&P Small Cap 600 index.[6] The passive small-large strategy has not performed satisfactorily during our sample period as well: a mean return of -0.91% (see figure 3.1 and table 3.1). Investors that have followed this strategy have experienced even greater fluctuations than those opting for the passive value-growth strategy, as revealed by the maximum (16.78%) and minimum (-15.71%) monthly returns and the higher standard deviation (12.04%). All of these findings cast serious doubt on the wisdom of persistently favoring

[6] Prior to the introduction of the S&P Small Cap 600 index in January 1994, the Frank Russell 1000 and Frank Russell 2000 indices have been used as inputs for the small-large calculations.

small stocks over large stocks in the past two decades.

3.2.2 Choice of the Forecasting Variables

We will introduce two classes of forecasting variables in this section. First, we give a brief overview of potential technical variables. Subsequently, we will address several macro-economic variables, which might shed some light on the behavior of the spread series. There appears to be a striking similarity between the chronological cumulative performance of the value and size premiums (see figure 3.1), which suggests that the behavior of both premiums might be subject to the same cyclical effects. We aim to provide a wide range of relevant forecasting variables, but we restrict ourselves to those claimed to be economically interpretable in the literature on this subject.

Good examples of technical factors are the lagged value and small cap spreads used by Levis and Liodakis (1999). Asness et al. (2000) propose two other variables of this class: the spread in valuation multiples and expected earnings growth between value portfolios and growth portfolios. Other candidates are changes in the implied volatility of the market, see Copeland and Copeland (1999), and price and earnings momentum in the market, see for instance Miller, Li, and Cox (2001), Kwon and Kish (2002) and Bernstein (2001).

The class of economic variables is mainly related to economic fundamentals, the business cycle and trends in corporate earnings. Examples of macro-economic series can be found in a variety of papers on style rotation. Kao and Shumaker (1999) document the influence of industrial production, the yield-curve spread, inflation (CPI) and the corporate credit spread on the value premium. In their view, industrial production reflects the corporate earnings cycle. In periods of high corporate earnings growth, the often highly leveraged value (and small) companies profit disproportionately. The composite leading indicators (CLI) can serve as an alternative to measure the same relationship. The interest rate environment can also have a substantial impact on the sign of the value premium. A yield spread widening between long government bonds and short term T-bills will probably hurt growth companies more than value companies as their profits are based further into the future. Growth stocks have longer durations than value stocks and are therefore more interest rate sensitive. These companies will underperform most likely in a setting with steep yield curves, which implies rising interest rates in the future. In the study of Levis and Liodakis (1999) the spread series are explained by the level of inflation, changes in the short-term interest rate and the equity risk premium respectively.[7]

In table 3.2 we list the variables actually used in our empirical analysis. In the next section we describe and discuss the nonparametric modeling tool used: Support Vector Regressions.

[7] Liew and Vassalou (2000) claim that past style performance can actually function as *a forecast* for economic growth, which brings a new dimension to this literature.

3.3 Methodology

This section describes the model-building tool (Support Vector Regression) and the construction of the SVR rotation models. Alongside, we focus on the qualities of SVR that justify their employment as a factor model tool.

3.3.1 Function Estimation with SVR

Support Vector Regressions (SVR), and Support Vector Machines (SVM) in general, are rooted in Statistical Learning Theory, pioneered by Vapnik (1995). In essence, SVR are just functions, named "learning machines", of which the basic task is to "explore" data (input-output pairs[8]) and provide optimally accurate predictions on unseen data. Extensive descriptions of SVR and SVM can be found, for example, in Burges (1998), Smola (1996), and Smola and Schölkopf (1998). Here we present a complete, but still compact and accessible representation of the basic SVR tool. The technical exposition follows mostly the descriptions in the abovementioned papers.

First, it should be mentioned that the standard loss function employed in SVR is the ϵ-insensitive loss function, which has the following form:

$$|y - f(\mathbf{x})|_\epsilon \equiv max\{0, |y - f(\mathbf{x})| - \epsilon\} \tag{3.1}$$

Here ϵ is predetermined and nonnegative, y is the true target value, $\mathbf{x}$ is a vector of input variables and $f(\mathbf{x})$ is the estimated target value. If the value of the estimate $f(\mathbf{x})$ of y is off-target by ϵ or less, then there is no "loss", and no penalty will be imposed. However, if $|y - f(\mathbf{x})| - \epsilon > 0$, then the value of the loss function rises linearly with the difference between y and $f(\mathbf{x})$ above ϵ. In practice, the actual loss associated with a given training error is equal to $C(|y - f(\mathbf{x})|_\epsilon)$, where C is a nonnegative constant. The term $|y - f(\mathbf{x})|_\epsilon$ is denoted by ξ if $y \leq f(\mathbf{x}) - \epsilon$, and by ξ^* if $y \geq f(\mathbf{x}) + \epsilon$.

Let us consider the simplest case of function estimation first, where there is only one input variable, x_1, one output variable, y, and l training data points, and a linear relationship between the input and output variables (see figure 3.2).

Notice that in the case of figure 3.2, the total amount of loss is equal to $C(\xi + \xi^*)$, since there are two training errors. The SVR algorithm estimates the parameters w_1 and b of the linear function $y = w_1x_1 + b$ for prespecified values of ϵ and C, ensuring that the resulting regression function achieves good generalization ability. It should not be too "complex", but, at the same time, it should not make too many training errors. Complexity here is defined in terms of "flatness" of the line, i.e. the smaller the slope of the line, the lower the complexity. By striking a balance between the function's complexity and

[8] The terms "inputs" and "outputs" in the machine learning domain stand for the "independent variables" and the "dependent variables" in the finance domain.

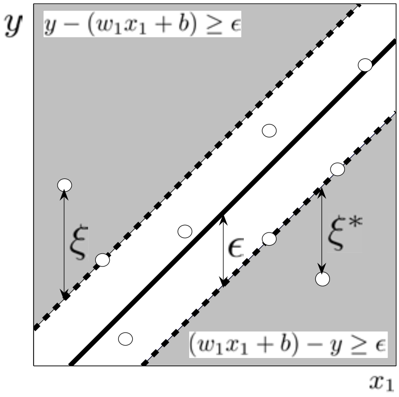


Figure 3.2: An SVR solution to the problem of estimating a relation between x_1 and y. All points inside the white region in the figure are within ϵ distance from the solid, optimal regression line $y = w_1x_1 + b$, and therefore are not penalized. However, penalties ξ_i and ξ_i^* are assigned to the two points that lie inside the shaded areas (given by $y - (w_1x_1 + b) \geq \epsilon$ and $(w_1x_1 + b) - y \geq \epsilon$). The optimal regression line is as flat as possible, and strikes a balance between the area of the white region and the amount of points that lie outside this region.

accuracy on the training data in the model-construction phase, the SVR offers a solution to the common problem of overfitting.

Figure 3.2 considers a one-dimensional input space, i.e. there is only one independent variable. If the dimension of the input space equals n, we are looking for the optimal regression function $f(\mathbf{x}) = (\mathbf{w} \cdot \mathbf{x}) + b$, with a vector of input variables $\mathbf{x} = (x_1, x_2, \ldots, x_n)$, "weight" vector $\mathbf{w} = (w_1, w_2, \ldots, w_n)$, and the inner product $(\mathbf{w} \cdot \mathbf{x}) = w_1x_1 + w_2x_2 + \ldots + w_nx_n$. Flatness in that case is defined in terms of the Euclidean norm of the weight vector: $\parallel \mathbf{w} \parallel = \sqrt{{w_1}^2 + {w_2}^2 + \ldots + {w_n}^2}$. The parameters of the linear SVR $f(\mathbf{x}) = (\mathbf{w} \cdot \mathbf{x}) + b$, i.e. $\mathbf{w}$, b, ξ_i and $\xi_i^*, i = 1, 2, \ldots, l$, can be found as the unique solution of the (convex quadratic) optimization problem:

$$\begin{aligned}
&\text{Minimize}\\
&\quad \frac{1}{2}\parallel \mathbf{w} \parallel^2 + C\sum_{i=1}^{l}(\xi_i + \xi_i^*)\\
&\text{Subject to}\\
&\quad y_i - (\mathbf{w}\cdot\mathbf{x}) - b \leq \epsilon + \xi_i\\
&\quad (\mathbf{w}\cdot\mathbf{x}) + b - y_i \leq \epsilon + \xi_i^*\\
&\quad \xi_i, \xi_i^* \geq 0\\
&\quad \text{for } i = 1, 2, \ldots, l
\end{aligned} \tag{3.2}$$

The first term of the objective (minimization) function in equation 3.2 deals with the complexity, and the second term deals with the accuracy (or, amount of training errors) of the model. In general, both terms cannot be minimal (or, close to zero) at the same time. The nonnegative parameter C determines the trade-off between the flatness of $f(\mathbf{x})$ and the amount of tolerated deviations. If C is large, some flatness could be lost in order to achieve greater training accuracy.

All points on the boundary of the ϵ-insensitive region together with the points outside that region (the training errors) are called “support vectors”. The computation of the regression is solely based on the support vectors.

The minimization problem of equation 3.2 can be represented in dual form, as a maximization problem:

$$\begin{aligned}
&\text{Maximize}\\
&\quad -\frac{1}{2}\sum_{i,j=1}^{l}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)k(\mathbf{x}_i, \mathbf{x}_j) +\\
&\quad +\sum_{i=1}^{l}(\alpha_i - \alpha_i^*)y_i - \epsilon\sum_{i=1}^{l}(\alpha_i + \alpha_i^*)\\
&\text{Subject to}\\
&\quad 0 \leq \alpha_i, \alpha_i^* \leq C, i = 1, 2, \ldots, l \text{ and}\\
&\quad \sum_{i=1}^{l}(\alpha_i - \alpha_i^*) = 0
\end{aligned} \tag{3.3}$$

where $k(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i \cdot \mathbf{x}_j)$.

The application of the kernel function $k(\mathbf{x}_i, \mathbf{x}_j)$ instead of the inner product $(\mathbf{x}_i \cdot \mathbf{x}_j)$ provides for the possibility to utilize other functional forms (see below).

In SVR, the regression estimates, which result from solving equation 3.3, take the form of:

$$f(\mathbf{x}) = \sum_{i=1}^{l} (\alpha_i^* - \alpha_i) k(\mathbf{x}, \mathbf{x}_i) + b \tag{3.4}$$

The value of b can be found from the so-called Karush-Kuhn-Tucker (KKT) conditions associated with the dual optimization problem (equation 3.3).

The training points in the series in equation 3.4 with coefficient $(\alpha_i^* - \alpha_i)$ unequal to zero are exactly the support vectors. For each training point $\mathbf{x}_i$ at most one of the two numbers α_i and α_i^* is unequal to zero. For the training points on the boundary of the ϵ-insensitive region holds either $0 < \alpha_i < C$ or $0 < \alpha_i^* < C$ and for the training errors outside the ϵ-insensitive region holds either $\alpha_i = C$ or $\alpha_i^* = C$.

Application of a kernel function transforms the original input space implicitly into a higher-dimensional input space where an optimal linear decision surface (corresponding to a nonlinear decision surface in the original input space) is found. One of the most frequently applied kernels is the so-called Radial Basis Function kernel (RBF). The dimension of the feature space for the RBF is infinite, which on first sight is counterintuitive from a complexity perspective: that should lead to overfitting. However, the literature reports very good performance of SVR using the RBF kernel (see, e.g. Burges, 1998; Chang, Chen, & Lin, 2001; Müller et al., 2001). Possible theoretical explanations thereof have been suggested in Burges (1998). Therefore, it appears that SVR with a RBF kernel are able to tackle the problems of overfitting effectively. For this reason we apply this kernel in our research.

The RBF kernel is defined as $k(\mathbf{x}_i, \mathbf{x}_j) = e^{-\gamma \| \mathbf{x}_i - \mathbf{x}_j \|^2}$, where γ is a manually adjustable parameter. The Radial Basis Function kernel is equal to 1 if $\mathbf{x}_i = \mathbf{x}_j$ and drops monotonically to zero with the Euclidean distance $\| \mathbf{x}_i - \mathbf{x}_j \|$ between the vectors $\mathbf{x}_i$ and $\mathbf{x}_j$. The greater the value of γ, the faster the function $k(\mathbf{x}_i, \mathbf{x}_j)$ decreases. So, for large values of γ the influence of a training point will be only local and the risk of overfitting will be large. So, the larger γ, the more "complex" the radial basis function is, and the smaller the number of training errors.

Summarizing, we have three parameters ϵ, C, and γ, which have to be tuned in order to find the optimal trade-off between complexity and training accuracy of the SVR. One of the ways to find the best trade-off between these parameters is via the standard cross-validation technique, which will be explained in Section 3.3.2.

3.3.2 SVR Style Timing Models

We will present the construction of the value rotation model only, since the size rotation model is constructed analogically.[9]

The input vectors for the SVR consist of the (historical) values for all 17 candidate explanatory factors as described in table 3.2. The outputs are the corresponding differences in returns between the S&P 500 Barra Value and Growth indices. Each SVR model is trained on the data for months $t-60$ till month $t-1$ in order to predict the output for month t. In order to find the optimal model parameters ϵ, C and γ we applied 5-fold cross-validation, a standard technique in machine learning (see, e.g. Stone, 1977; Weiss & Kulikowski, 1991) on the training data sets of 60 months. A k-fold cross-validation procedure is utilized as follows: a given dataset is divided into k folders of equal size; subsequently, a model is built on all possible (k) combinations of $k-1$ folders, and each time the remaining one folder is used for validation. The model that achieves minimum mean sum of squared errors on average (over the k validation folders) is considered to be the best. This best model is said to achieve minimum cross-validation mean squared error, and the parameters of this model are used in the final model for the prediction of month t.

The advantage of using a cross-validation procedure is that it ensures that model selection is based entirely on out-of-sample rather than in-sample performance. The disadvantage however is that the procedure is rather time-consuming. A tiny part of the cross-validation procedure is visualized in figure 3.3, where the vertical axis shows the cross validation minimal squared errors for $C \in (0, 35)$, while keeping ϵ and the kernel function parameter γ fixed at 1.0 and 0.007, respectively. As suggested by the figure, the value for the minimum cross-validation mean squared error is well defined.

The predicted output, i.e. the value premium for month t is used to decide on our timing rotation strategy. A positive output will result in a signal "Value" in which case we will buy the Value index and sell the Growth index, while a negative output will result in a signal "Growth" with the opposite effect. In order to avoid taking decisions based on noise, we treat an output value close to zero as a "no signal" signal.[10]

The SVR small-large strategy is defined analogically, using S&P SmallCap 600 and S&P 500 indices.

[9] The software program used throughout the analysis is LIBSVM 2.4, developed by Chang and Lin (2002).

[10] We used a range of (-0.05, 0.05) standard deviations relative to the average of the estimates over the training period.

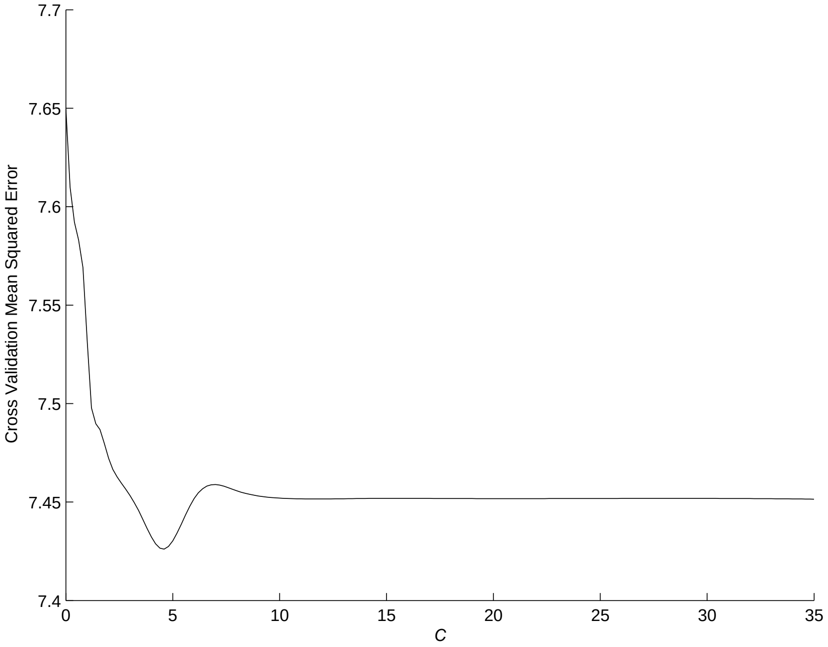


Figure 3.3: Five-fold cross validation mean squared errors associated with the penalty-on-error parameter $C \in (0, 35)$ and fixed ϵ-insensitive loss function parameter (ϵ) at 1.0 and Radial Basis Function parameter at 0.007. The to-be-predicted month here is April 2000. The "best" model is the one for which the combination of the three parameters over suitable parameter ranges produces minimal cross validation mean squared error.

3.4 Empirical Results

In this section we will present the main results from value-growth and small-large rotation strategies using SVR with different levels of transaction costs and varying forecast horizons (one-month, three-month and six-month). Additionally, we will show the output of an equally weighted combination of both strategies. Throughout this empirical section we show returns that could be achieved when we would have been able to forecast the signal correctly each month: MAX_VG (value-growth rotation) and MAX_SL (small-large rotation). The input for the SVR model consists of 60 months of data on the whole set of 17 predetermined factors. The passive style strategies are constructed in accordance with what is expected in the literature: each month a long position is taken in the Value index and a short position in the Growth index. The passive small-large strategy consistently buys the S&P SmallCap 600 index and sells the S&P 500 index.

3.4.1 Value-Growth Rotation Strategies

Detailed results of the SVR value-growth strategy can be found in the left part of table 3.3. What strikes most at first sight, is that this strategy has produced much better results than the passive strategy in the out-of-sample period starting January 1993 and ending December 2002.

Table 3.2: Variables used in the style timing models based on Support Vector Regressions.

	Technical Variables
LagVmG	Lagged Value/Growth spread
LagSmL	Lagged Small/Large spread
VOL	Volatility of the S&P 500
FPE	12-month Forward P/E of the S&P 500
MOM	6-month Momentum of the S&P 500
Profit cycle	Year on Year change in earnings per share of the S&P 500
PE dif	Price/Earnings difference between Value and Growth indices, or between S&P 500 and Small cap indices
DY dif	Difference between dividend yields on Value and Growth indices, or S&P 500 and Small cap indices
	Economic Variables
Corporate Credit Spread	The yield spread of (Lehman Aggregate) Baa over Aaa
Core inflation	The 12-month trailing change in the U.S. Consumer Price Index
Earnings-yield gap	The difference between the forward E/P ratio of the S&P 500 and the 10-year T-bond yield
Yield Curve Spread	The yield spread of 10-year T-bonds over 3-month T-bills
Real Bond Yield	The 10-year T-bond yield adjusted for the 12-month trailing inflation rate
Ind. Prod	U.S. Industrial Production Seasonally Adjusted
Oil Price	The 1-month price change
ISM (MoM)	1-month change of U.S. I.S.M. Purchasing Managers Index (Mfg Survey)
Leading Indicator	The 12-month change in the Conference Board Leading Indicator

Table 3.3: Summary performance statistics for passive rotation strategies and various Support Vector Regression strategies using a one-month forecast horizon, for the period 1993:01 to 2002:12. “VmG” and “SmL” denote passive value-growth and small-large strategies, respectively. The explanatory variables are listed in table 3.2. MAX_VG and MAX_SL denote the perfect foresight strategies. CV denotes a timing strategy based on Support Vector Regression cross validation mean squared error. All numbers are annualized data unless stated otherwise. All strategies are long/short monthly positions on the style and size indices. The overall position for month $t+1$ is based on the signal produced by the optimal model based on 60 months of historical data on all explanatory factors. Transaction costs are assumed to be 0 bp., 25 bp., and 50 bp. single trip. The row “% months in Growth/Large" has to be interpreted as “% months in Growth" for the columns considering the value-growth rotation and “% months in Large" for the columns considering the small-large rotation. The interpretation of the next row is similar.

	value-growth rotation					small-large rotation				
	VmG	CV 0 bp	CV 25 bp	CV 50 bp	MAX_VG 50 bp	SmL	CV 0 bp	CV 25 bp	CV 50 bp	MAX_SL 50 bp
Mean	0.24	10.30	8.30	6.30	21.54	-1.26	10.71	9.11	7.51	27.04
Standard deviation	10.95	9.95	9.90	9.90	7.83	13.00	10.92	10.92	10.96	8.87
Information ratio	0.02	1.04***	0.84***	0.64**	2.75***	-0.10	0.98***	0.83***	0.69**	3.05***
Z(equality)		2.15***	1.73*	1.30	5.00***		2.23**	1.93*	1.63	5.69***
Median	-0.11	0.33	0.31	0.30	0.50	0.05	0.45	0.44	0.43	0.79
Minimum (monthly)	-12.02	-5.51	-5.51	-5.51	-0.98	-15.71	-7.70	-7.70	-7.70	-0.96
Maximum (monthly)	9.74	12.02	11.77	11.52	11.02	16.78	16.78	16.53	16.28	16.78
Skewness (monthly)	0.01	1.21	1.18	1.13	1.61	0.20	0.77	0.76	0.74	2.63
Excess kurtosis (monthly)	2.40	2.66	2.52	2.34	3.41	4.49	4.74	4.55	4.30	11.74
% negative months	45.83	32.50	49.17	50.00	19.17	51.67	33.33	44.07	45.83	10.00
Largest 3-month loss	-11.55	-5.90	-6.40	-6.90	-1.99	-21.63	-8.84	-8.84	-9.21	-1.18
Largest 12-month loss	-22.86	-8.07	-11.51	-15.26	2.21	-31.85	-3.21	-5.46	-7.71	8.03
% months in Growth/Large	0.00	53.33	53.33	53.33	45.83	0.00	45.00	45.00	45.00	51.67
% months in Value/Small	100.00	28.33	28.33	28.33	54.17	100.00	45.00	45.00	45.00	48.33
% months no position	0.00	18.33	18.33	18.33	0.00	0.00	10.00	10.00	10.00	0.00

* indicates significance at the (2-tail) 10% level
** indicates significance at the (2-tail) 5% level
*** indicates significance at the (2-tail) 1% level

Under the assumption of zero transaction costs, the SVR strategy achieves an annualized mean return of 10.30%, against a modest 0.24% respective return of the passive strategy. Combining these results with the standard deviations of returns yields (annualized) information ratios of 1.04 and 0.02, respectively. Besides, even when high transaction costs of 50 basis points (bp.) (single trip) are added into the calculations, the realized SVR-model information ratio remains quite high (0.64), and statistically significant at the 5% two-tail level. When compared to other studies on the subject, for example Bauer and Molenaar (2002) in the U.S. and Levis and Liodakis (1999) in the U.K., the SVR results seem to demonstrate a significant improvement. The calculated Z(equality)-scores[11] provide further evidence (in the 0 bp. and 25 bp. transaction-cost environment) of a significant performance difference. In addition, the SVR strategy is able to capture 37.7%, 34.0%, and 29.3% of the return from the MAX_VG strategy under 0 bp., 25 bp. and 50 bp. transaction costs, respectively. Note that in table 3.3 only the results of the MAX_VG strategy under 50 bp. transaction costs are given. Table 3.3 further reveals that the largest three-month and twelve-month losses associated with the SVR value-growth model are substantially less than the respective losses incurred by the passive strategy. Summarizing, all of these findings can serve as an indication of robustness of the SVR strategy.

Figure 3.4 shows style signals associated with the SVR value-growth rotation strategy. The predominant style signal during this period is "Growth", with some notable exceptions however. "Value" signals have been produced mostly in 1993, in the beginning of 1994, and in the first half of 2001. Almost no "Value" signals have been given during the periods stretching from June 1996 till August 1998, and from June 1999 till November 2000.

Figure 3.5 presents the realized excess returns forecasted by the basic SVR style timing strategy in the 25 bp. transaction-cost scenario. It can be seen from the figure that most of the accrued returns come out of the last four years of the sample period, which actually appears to be the most volatile.

A number of further conclusions can be drawn by examining figure 3.6. Next to the cumulative returns from the passive strategy and the SVR strategy that predicts the one-month-ahead return difference under zero transaction costs, the figure reveals the cumulative returns from two more strategies: the three- and six-month-horizon SVR strategies. The latter two strategies are constructed simply by taking the (unweighted) average of the signals produced by models constructed up to three and up to six months before any predicted month, and investing according to this combined signal. Our procedure is equal to that used in Jegadeesh and Titman (1993).

The first striking feature is that most of the cumulative returns are accrued in

[11] Z(equality) measures the risk-adjusted performance difference between a switching Support Vector Regression strategy and the passive value-growth strategy. The Z(equality)-score is computed in a standard way (in line with, e.g. Glantz, 1992).

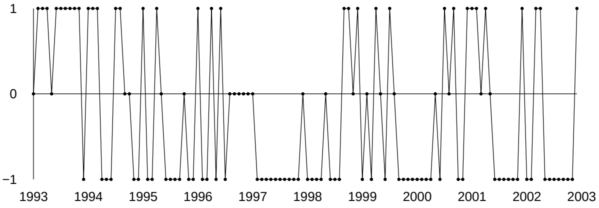


Figure 3.4: Investment signals (“value” = 1, “growth” = -1, “no signal” = 0) produced by the SVR value-growth model investment strategy.

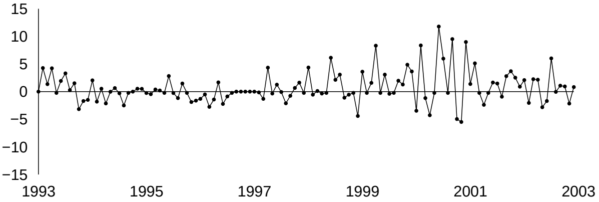


Figure 3.5: Realized excess returns forecasted by the SVR value-growth investment strategy for the 25 bp. transaction costs scenario.

times of relatively higher volatility, and especially during 1993, in the beginning of 1994, and between 1999 and 2002. The magnitude of the volatility of returns can be observed by tracking the (monthly) changes in the cumulative returns of the passive strategy. Larger shocks in these series correspond to greater volatility of the value premium. A second interesting feature is that the basic one-month-horizon SVR strategy performs better than in the case of three- and six-month forecast horizons. A potential reason for this is that forecast signals produced by models built in the more distant past become less accurate than those provided by the more recent models, which are constructed using newer information.

3.4.2 Small-Large Rotation Strategies

Detailed information on the small-large SVR strategy, the passive small-large strategy and the maximum attainable MAX_SL strategy can be found in the right part of table 3.3.

In the out-of-sample period the passive small-large rotation strategy achieves an annual return of -1.26%. The optimal MAX_SL strategy provides an annual

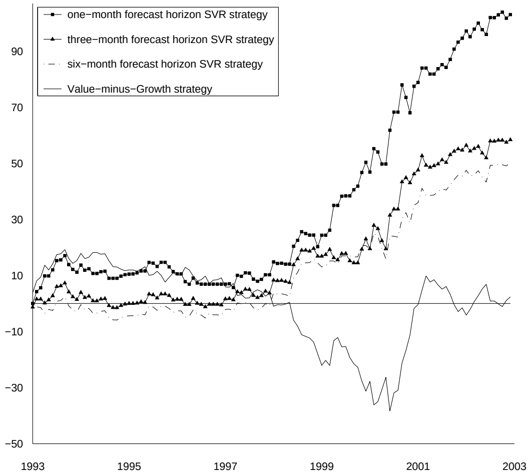


Figure 3.6: Accrued cumulative returns from the passive value-growth strategy and the Support Vector Regression (SVR) one-, three-, and six-month horizon strategies for the period January 1993 – December 2002, under no transaction costs. The one-month horizon strategy performs best, gaining most of its accumulated profits during turbulent times on the financial market. In such periods, the three- and six-month horizon models follow suit with a time lag, as logically expected. During relatively calmer periods, all strategies perform similarly.

return of 27.04% in the 50 bp. transaction-cost scenario, which is 5.50% more than the corresponding result for the MAX_VG strategy. This reveals that the potential benefit from size rotation seems to be much greater than the one from the corresponding value-growth rotation. Table 3.3 shows that this extra potential can indeed be captured. Moreover, for the zero-transaction-cost regime, for example, the one-, three- and six-month forecast horizon small-large strategies produce 10.71%, 8.03% and 7.73% annual returns, while the respective results from the SVR value-growth strategies are 10.30%, 5.84% and 5.02% respectively.[12] As in the value-growth case, the SVR size model is able to capture roughly one third of the maximum attainable cumulative returns under all considered transaction cost regimes.

As it turns out, the results from the robustness checks that were performed on the SVR value-growth model are also valid for SVR size rotation. Under the assumption of 50 bp. transaction costs, the realized information ratio of 0.69 from the SVR size model is significant at the (two-tail) 5% level. The

[12] Not all these results are presented in table 3.3. They are available upon request.

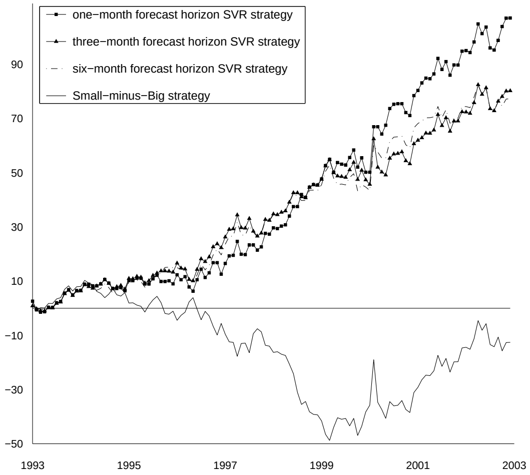


Figure 3.7: Accrued cumulative returns from the passive small-large strategy and the Support Vector Regression (SVR) one-, three-, and six-month horizon strategies for the period January 1993 – December 2002, under no transaction costs. The one-month horizon strategy performs best, gaining the predominant part of its accumulated profits during turbulent times on the financial market. The three- and six-month horizon models follow a very similar pattern, but perform slightly worse. During relatively calmer periods, all strategies perform similarly.

SVR size strategy produces significantly different results from the passive size strategy. The largest three-month and, especially, twelve-month losses from the SVR model are drastically more bearable than the ones from the passive size strategy: -8.84% vs. -21.63% and -3.21% vs. -31.85%, respectively, under zero transaction costs. Notice, additionally, that the one-month strategy again outperforms the longer horizon alternatives, consistent with the findings of the SVR value-growth strategy, see figure 3.7.

3.4.3 Simultaneous Value-Growth and Size Timing

In case investors have decided to follow both the value and size SVR strategies simultaneously at the beginning of the sample period, they would have witnessed even greater relative gains as compared to sticking only to a single type of timing (see table 3.4 and figure 3.8 for details).

Indicative of this are the realized information ratios of simultaneous style and size timing: 1.27, 1.06 and 0.84 under 0 bp., 25 bp. and 50 bp. single trip

Table 3.4: Simultaneous passive and Support Vector Regression value-growth and small-large rotation strategies. Summary statistics for following simultaneously passive value-growth (VmG) and passive small-large (SmL) strategies on the one hand, and Support Vector Regression value-growth and small-large rotation strategies using a one-month forecast horizon, on the other, for the period 1993:01 to 2002:12. The explanatory variables are listed in table 3.2. MAX denotes the perfect foresight value-growth and small-large rotation combined strategy. CV denotes the timing strategy based on Support Vector Regression cross validation mean squared error. All numbers are annualized data unless stated otherwise. All strategies are long/short monthly positions on the style and size indices. The overall position for month $t+1$ is based on the signal produced by the optimal model based on 60 months of historical data of all explanatory factors. Transaction costs are assumed to be 0 bp., 25 bp., and 50 bp. single trip.

	simultaneous rotation				
	VmG plus	CV	CV	CV	MAX
	SmL	0 bp	25 bp	50 bp	50 bp
Mean	-0.51	10.51	8.71	6.91	24.29
Standard deviation	8.71	8.25	8.23	8.23	6.76
Information ratio	-0.06	1.27***	1.06***	0.84***	3.59***
Z(equality)		2.90***	2.43**	1.96*	7.11***
Median	0.06	0.44	0.41	0.39	0.56
Minimum (monthly)	-7.26	-3.86	-4.24	-4.61	-0.94
Maximum (monthly)	8.43	12.56	12.44	12.31	12.06
Skewness (monthly)	0.13	1.51	1.49	1.45	1.87
Excess kurtosis (monthly)	1.12	5.29	5.30	5.23	6.05
% negative months	49.17	33.33	37.50	40.83	7.50
Largest 3-month loss	-13.20	-4.10	-5.10	-6.10	-1.17
Largest 12-month loss	-26.71	-3.96	-6.34	-8.82	6.77

* indicates significance at the (2-tail) 10% level
** indicates significance at the (2-tail) 5% level
*** indicates significance at the (2-tail) 1% level

transaction cost regimes, all significant at the (two-tail) 1% level. Not surprisingly, these information ratios are higher than the ones associated with either style or size timing individually, as investors actually diversify the risk associated with each timing strategy. The information ratio of the passive simultaneous timing strategy is negative (-0.06). Interestingly, the largest three-month and twelve-month losses associated with simultaneous investing turn out to be quite tolerable: -4.10% and -3.96%, assuming zero transaction costs. It appears that it pays to diversify the market timing strategies, at least as far as value and size timing are concerned.

Admittedly, we expect all of our findings to be dependent on the historical model-building horizon and on the length of the trading period. Choosing to trade for a longer period could come at the expense of incurring formidable transaction costs, as noted in the Data section. Additionally, varying the length

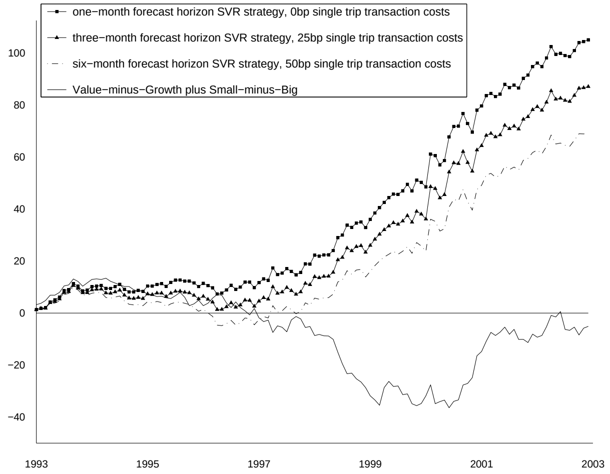


Figure 3.8: Accrued cumulative returns from investing simultaneously according to the Support Vector Regression (SVR) one-month horizon value-growth and small-large strategies on the one hand, and from investing simultaneously according to the passive value-growth and small-large strategies during the period January 1993 – December 2002, under different transaction costs.

of the model-building horizon might yield a “best” horizon that would be difficult to justify. Thus, future research could concentrate on both of these issues.

3.4.4 Discussion of Results

Overall, our findings on the predictability of the size and the value premium corroborate the results of previous studies for the US (Kao & Shumaker, 1999) and other mature markets, such as the UK (Levis & Liodakis, 1999) and Japan (Bauer et al., 2004). Our study shows that a style consistent strategy, i.e. consistently favoring value over growth and small over large, does not necessarily lead to positive returns in the long run. The proposed SVR style timing strategies, taking full advantage of information on the market and the economic cycle, are partially able to forecast the sign of both the value and the sign premium. This ability is particularly evident during volatile times, and especially during the TMT bubble and its aftermath (1998-2001). Furthermore, the transaction costs of our rotation strategies are expected to be small as futures on well-known and liquid indices are applied. This raises the possibility that institutional investors can exploit this strategy in real time.

Nonetheless, we should be careful in interpreting these results. Amihud and

Mendelson (1986) for instance argue that the bid-ask bounce possibly creates an upward bias in reported profits from trading strategies. To remedy this effect, they suggest adjusting transaction prices by one half of the bid-ask spread. This actually is equivalent to adding an extra fixed amount of transaction costs to each trade on top of what has already been assumed in our rotation strategies, provided that the bid-ask spread remains constant through time. As we use futures on well-known indices, it is likely that we grossly capture this effect by assuming relatively high transaction costs (50 bp.). Further, non-synchronous trading might induce spurious cross-autocorrelation between less frequently and more frequently traded stocks, see Campbell, Lo, and MacKinlay (1997). This particular issue might be applicable to a greater extent to the size rotation rather than the value-growth rotation strategy, since small stocks are traded less frequently in general. The use of a forecasting horizon of 3 and 6 months however yields similar results. For such long horizons the market microstructure effects mentioned above are less likely to be relevant.[13]

3.5 Conclusion

This chapter examines whether short-term directional variations in the size and value premium in the U.S. stock market are sufficiently predictable to be exploited by means of a tactical timing strategy. As a forecasting tool, we employ the so-called Support Vector Regressions (SVR). SVR have only recently been developed in the artificial intelligence field and have been rarely applied in a financial context. Using SVR, we are able to circumvent the well-known problems of overfitting, especially in multivariate settings, in an elegant way.

Our empirical findings clearly show that both premiums are highly predictable during the trading period. This comes at odds with the mainstream literature that provides evidence for the long-term superiority of returns to value vis-a-vis growth and small vis-a-vis large stocks. After adjustment for fair levels of transaction costs this result still holds. Under high transaction cost levels, expected to be relevant in a dynamic economic environment, it is difficult in practice to obtain incremental benefits over style consistent strategies. That is why it is critical to develop timing strategies that can be implemented using index futures or low-cost trading baskets like exchange traded funds. In terms of realized information ratios, a combination of both value-growth and small-large timing produces the most interesting results.

[13] Additionally, we looked at strategies where we assume higher transaction costs for small cap trades (both long and short) and short large cap trades. Results, which are available upon request, show that it is difficult to exploit the rotation strategy when using a basket of individual stocks instead of futures.

Chapter 4

Solving and Interpreting Binary Classification Problems in Marketing with SVMs*

Marketing problems often involve binary classification of customers into "buyers" versus "non-buyers" or "prefers brand A" versus "prefers brand B". These cases require binary classification models such as logistic regression, linear, and quadratic discriminant analysis. A promising recent technique for the binary classification problem is the Support Vector Machine (Vapnik, 1995), which has achieved outstanding results in areas ranging from Bioinformatics to Finance. In this chapter, we compare the performance of the Support Vector Machine against standard binary classification techniques on a marketing data set and elaborate on the interpretation of the obtained results.

4.1 Introduction

In marketing, quite often the variable of interest is dichotomous in nature. For example, a customer either buys or does not buy a product, visits or does not visit a certain shop. Some researchers and practitioners often approach such binary classification problems with traditional parametric statistical techniques,

*This chapter has been published as Nalbantov et al. (2006b):
Nalbantov, G. I., Bioch, J. C., & Groenen, P. J. F. (2006b). Solving and Interpreting Binary Classification Problems in Marketing with SVMs. In M. Spiliopoulou, R. Kruse, C. Borgelt, A. Nurnberger, & W. Gaul (Eds.), *From Data and Information Analysis to Knowledge Engineering* (pp. 566–573). Springer.

such as discriminant analysis and logistic regression (Lattin, Carroll, & Green, 2003; Franses & Paap, 2001) and others employ semiparametric and nonparametric statistical tools, like kernel regression (Van Heerde, Leeflang, & Wittink, 2001; Abe, 1991, 1995) and neural networks (West, Brockett, & Golden, 1997). Nonparametric models differ from parametric in that they make no or less assumptions about the distribution of the data. A disadvantage of nonparametric tools in general is that they are considered to be "black boxes". In many such cases, the model parameters are hard to interpret and often no direct probability estimates are available for the output binary variable. A discussion on the relative merits of both kind of techniques can be found, for instance, in Van Heerde et al. (2001) and West et al. (1997).

In this chapter, we employ the nonparametric technique of Support Vector Machine (SVM) (Vapnik, 1995; Burges, 1998; Müller et al., 2001). Some desirable features of SVM that are relevant for marketing include good generalization ability, robustness of the results, and avoidance of overfitting. One drawback of SVM is the inability to interpret the obtained results easily. In marketing, SVMs have been used by, for example, Bennett, Wu, and Auslender (1999), Cui (2003), and Evgeniou and Pontil (2004).

Our aim is to assess the applicability of SVM for solving binary marketing problems and, even more importantly, to provide for the interpretation of the results. We compare SVM with standard marketing modelling tools of linear and quadratic discriminant analysis and the logit choice model on one empirical data set. In addition, we interpret the results of the SVM models in two ways. First, we report probability estimates for the realizations of the (binary) dependent variable, as proposed by Platt (1999) and implemented by Chang and Lin (2002). Second, we use these estimates to evaluate the (possibly nonlinear) effects of some independent variables on the dependent variable of interest. In this way, we can assess the effect of manipulating some marketing instruments on the probability of a certain choice between two alternatives.

The remainder of the chapter is organized as follows. First, we describe the data used in this research. Next, we provide a brief overview of the construction of SVM for classification tasks. Sections 4.4 and 4.5 give an account of the obtained results and their interpretation and Section 4.6 gives a conclusion.

4.2 Data

We focus on a straightforward marketing problem: how to forecast holiday length on the basis of some general travelling and customer characteristics. These data have been collected by Erasmus University Rotterdam in 2003. Table 4.1 provides descriptive statistics for the data set. The dependent variable, holiday length, has been dichotomized into "not more than 14 days" and "more than 14 days". In total, there are 708 respondents. The outcome alternatives

Table 4.1: Descriptive statistics of the predictor variables for the holiday data set split by holiday length. For the categorical variables, the relative frequency is given (in %) and for numerical variables, the mean.

	Holiday length in days			Holiday length in days	
Variable	≤ 14	> 14	Variable	≤ 14	> 14
Transport			Destination		
Car	39.8	34.2	Inside Europe	87.7	66.7
Airplane	48.0	58.2	Outside Europe	12.3	33.3
Other	12.2	7.6	Accommodation		
Full board			Camping	17.5	27.9
Yes	25.7	18.3	Apartment	29.5	24.0
No	74.3	81.7	Hotel	33.6	27.6
Sunshine			Other	19.4	20.5
Important	83.9	88.5	Season		
Not important	16.1	11.5	High	38.6	43.2
Big expenses			Low	61.4	56.8
Made	26.0	26.5	Having children		
Not made	74.0	73.5	Yes	31.6	40.2
Mean no. of children	0.35	0.49	No	68.4	59.8
Mean age group	3.95	4.52	Mean income group	2.23	2.67

are quite balanced: 51.7% of the respondents have spent more than two weeks and 48.3% not more than two weeks of holidays. Eleven explanatory variables were available, some of which are categorical: destination, mode of transport, accommodation, full/nonfull board and lodging, sunshine availability, (other) big expenses, in/out of season, having/not having children, number of children, income group and age group.

4.3 Support Vector Machines for Classification

Support Vector Machines (SVM) are rooted in statistical learning theory (Vapnik, 1995) and can be applied to both classification and regression problems. We consider here the supervised learning task of separating examples that belong to two classes. Consider a data set of n explanatory vectors $\{\mathbf{x}_i\}_{i=1}^n$ from $\mathbb{R}^m$ and corresponding classification labels $\{y_i\}_{i=1}^n$, where $y_i \in \{-1, 1\}$. Thus, in the marketing data set, -1 identifies short holiday length (≤ 14 days) and 1 identifies long holiday length ($>$ 14 days). The SVM method finds the oriented hyperplane that maximizes the closest distance between observations from the two classes (the so-called "margin"), while at the same time minimizes the amount of training errors (Vapnik, 1995; Cristianini & Shawe-Taylor, 2000; Burges, 1998). In this way, good generalization ability of the resulting function is achieved, and therefore the problem of overfitting is mitigated.

The explanatory vectors $\mathbf{x}$ from the original space $\mathbb{R}^m$ are usually mapped into a higher dimensional, space, where their coordinates are given by $\Phi(\mathbf{x})$. In this case, the optimal SVM hyperplane is found as the solution of the following optimization problem:

$$\begin{array}{c} \max_\alpha \ \sum_{i=1}^n \alpha_i - \frac{1}{2}\sum_{i,j=1}^n \alpha_i\alpha_j y_i y_j k(\mathbf{x}_i, \mathbf{x}_j) \\ \text{subject to } 0 \leq \alpha_i \leq C, \ i = 1, 2, \cdots, n, \text{ and } \sum_{i=1}^n y_i\alpha_i = 0, \end{array} \tag{4.1}$$

where $k(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i)'\Phi(\mathbf{x}_j)$ is a kernel function that calculates dot products of explanatory vectors $\mathbf{x}_i$ and $\mathbf{x}_j$ in feature space. Intuitively, the kernel determines the level of proximity between any two points in the feature space. Common kernels in SVM are the linear $k(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i'\mathbf{x}_j)$, polynomial $k(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i'\mathbf{x}_j + 1)^d$ and Radial Basis Function $k(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma||\mathbf{x}_i - \mathbf{x}_j||^2)$ ones, where d and γ and manually adjustable parameters. The feature space implied by the RBF kernel is infinite-dimensional, while the linear kernel preserves the data in the original space. Maximizing the term $-\sum_{i,j=1}^n \alpha_i\alpha_j y_i y_j k(\mathbf{x}_i, \mathbf{x}_j)$ corresponds to maximizing the margin between the two classes, which is equal to the distance between hyperplanes with equations $\sum_{i=1}^n y_i\alpha_i k(\mathbf{x}_i, \mathbf{x}) + b = -1$ and $\sum_{i=1}^n y_i\alpha_i k(\mathbf{x}_i, \mathbf{x}) + b = 1$. The manually adjustable constant C determines the trade-off between the margin and the amount of training errors. The α's are the weights associated with the observations. All observations with nonzero weights are called "support vectors", as they are the only ones that determine the position of the optimal SVM hyperplane. This hyperplane consists of all points $\mathbf{x}$ which satisfy $\sum_{i=1}^n y_i\alpha_i k(\mathbf{x}_i, \mathbf{x}) + b = 0$. The b parameter is found from the so-called Kuhn-Tucker conditions associated with (4.1).

The importance of binary classification methods lies in how well they are able to predict the class of a new observation $\mathbf{x}$. To do so with SVM, the optimal separation hyperplane $\sum_{i=1}^n y_i\alpha_i k(\mathbf{x}_i, \mathbf{x}) + b = 0$ that is derived from the solution $(\{\alpha_i\}_{i=1}^n, b)$ of (4.1) is used:

$$f(\mathbf{x}) = \text{sign}(g(\mathbf{x})) = \text{sign}\left(\sum_{i=1}^n y_i\alpha_i k(\mathbf{x}_i, \mathbf{x}) + b\right),$$

where $\text{sign}(a) = -1$ if $a < 0$, $\text{sign}(a) = 1$ if $a \geq 0$.

For interpretation, it is often important to know not only the predicted binary outcome, but also its probability. One way to derive posterior probabilities for the estimated class membership $f(\mathbf{x}_i)$ of observation $\mathbf{x}_i$ has been proposed by Platt (1999). His approach is to fit a sigmoid function to all estimated $g(\mathbf{x}_i)$ to derive probabilities of the form:

$$P(y = 1|g(\mathbf{x}_i)) = p_i = (1 + \exp(a_1 g(\mathbf{x}_i) + a_2))^{-1},$$

where a_1 and a_2 are estimated by minimizing the negative log-likelihood of the training data:

$$\min_{a_1, a_2} -\sum_{i=1}^n \left(\frac{y_i + 1}{2}\log(p_i) + (1 - \frac{y_i + 1}{2})\log(1 - p_i)\right).$$

Table 4.2: Hit rates (in %) of different learning methods for the vacation data set. Approximately 85% and 15% of each data set are used for training and testing, respectively. LDA, QDA and logit stand for Linear Discriminant Analysis, Quadratic Discriminant Analysis and logit choice model.

					lin	poly	RBF
Sample		LDA	QDA	logit	SVM	SVM	SVM
Training	$\leq$ 14 days	68.2	69.2	63.3	73.0	78.9	77.5
	$>$ 14 days	63.3	67.5	66.2	60.5	59.2	61.4
	Overall	65.7	68.3	64.8	66.5	68.7	69.8
Test	$\leq$ 14 days	64.2	54.7	60.4	58.5	75.5	71.7
	$>$ 14 days	56.4	54.6	65.5	49.1	45.5	52.7
	Overall	60.2	54.6	63.0	53.7	60.2	62.0

4.4 Experiments and Results

We define a training and a test sample, corresponding to 85% and 15% of the original data set, respectively. Our experiments have been carried out with the LIBSVM software (Chang & Lin, 2002). We have constructed three SVM models, which differ in the transformation of the original data space, that is, using the linear, the polynomial of degree 2 ($d = 2$) and the RBF kernel. Table 4.2 shows detailed results of the SVM models as well as competing classification techniques in marketing such as linear discriminant analysis (LDA), quadratic discriminant analysis (QDA), and the logit choice model. The manually adjustable parameters C and γ have been estimated via a five-fold cross-validation procedure. As a result, the parameters for the linear, polynomial and RBF SVM models have been set as follows: $C = 2.5$, $C = 0.004$ and $d = 2$, $C = 3500$ and $\gamma = 0.0013$.

The overall performance of SVM on the test set is comparable to that of the standard marketing techniques. Among SVM models, the most flexible one (RBF-SVM) is also the most successful at generalizing the data. The average hit rate on the test set of all techniques considered centers at around 59%. There is no substantial distinction among the performance of all models, except for the QDA and linear SVM models, which relatively underperform. In such a setting we generally favor those models that can be better interpreted.

4.5 Interpreting the Influence of the Explanatory Variables

The classical SVM appears to lack two main interpretation aspects shared by the standard models of LDA, QDA, and logit choice model. First, for the standard models, coefficient estimates for each explanatory variable are available and can be interpreted as the direct effect of a change in one of the independent variables on the dependent variable, while keeping all other independent variables fixed. The same interpretation is possible for the linear SVM model, since the original data space is preserved, and thus individual coefficient estimates are available. For all the other types of SVM this direct variable effect can be highly nonlinear and is not directly observable. The SVM with RBF kernel, for example, implies infinitely many number of explanatory variables, and thus infinitely many coefficients for each of these variables, which makes interpretation impossible at first sight.

Second, the coefficient estimates obtained from the standard models can be used to derive the effect of each explanatory variable on the probability of a certain binary outcome. Although classical SVM does not output outcome probabilities, one can use here the proposed probability estimates by Platt (1999), discussed in Section 4.3. Interestingly, these probability estimates can help to derive individual variable effects also for the nonlinear SVM. For interpretation purposes, all that is needed is to visualize the relationship between a given explanatory variable and the probability to observe one of the two possible binary outcomes, while keeping the rest of the explanatory variables fixed. Thus, even for the SVM with RBF kernel it is not necessarily to know the coefficients for each data dimension in order to infer the influence of individual variables.

Next, we interpret the results of the SVM model with RBF kernel on the vacation data set and compare them with those from the logit model. Consider Figures 4.1 and 4.2 that show the relationships between some of the independent variables and the probability to go on a vacation for more than two weeks, for the logit and RBF-SVM models respectively. In each of the panels, the remaining explanatory variables are kept fixed at their average levels. The dashed lines denote the probability of the "average" person to go on a vacation for more than two weeks.

The first striking feature to observe is the great degree of similarity between both models. Although the RBF-SVM model is very flexible, the estimated effects for variables such as "Having children", "Big expenses", and "In season" are close to linear, just as the logit model predicts. The main difference between both techniques is best illustrated by the predicted effect of the "Age group" variable. The SVM model suggests that both relatively younger and relatively older holiday makers tend to have (on average) a higher probability to choose for the longer vacation option than the middle-aged ones, which makes sense

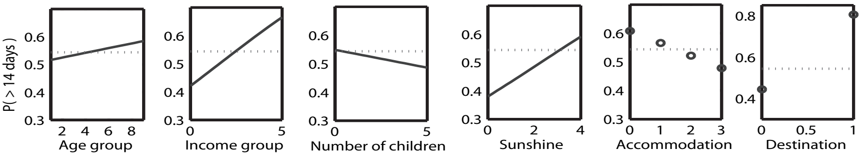


Figure 4.1: Influences of individual explanatory variables on the probability to spend more than two weeks on a vacation for the logit model.

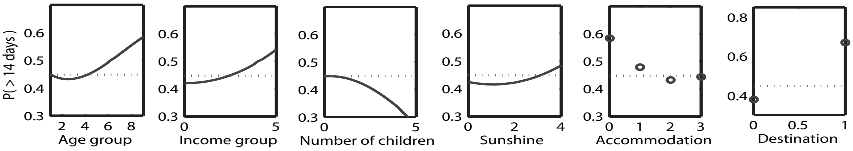


Figure 4.2: Influences of individual explanatory variables on the probability to spend more than two weeks on a vacation for the RBF-SVM model.

intuitively. The logit model cannot capture such an effect by its definition as it imposes a monotonically increasing (or decreasing) relationship between the explanatory variables and the probability of a certain outcome. The RBF-SVM model, on the other hand, is free to impose a highly nonlinear such relationship via the mapping of the original data into a higher-dimensional space. Moreover, since the SVM model does not suffer from monotonicity restrictions, it reports nonmonotonically ordered outcome probabilities for each of the ”Accommodation” variable categories (see Figure 4.2). Although one cannot conclude here that SVM is immune to the need to optimally scale the variables prior to model estimation, it is clear that it offers a better protection from arbitrary coding of unordered categorical variables than the logit model does.

The marketing implications of the results obtained by SVM can be derived directly from Figure 4.2. By considering the effects of changes in individual variables, marketeers can infer which ones are most effective and, as a result of this, streamline the advertising efforts accordingly. Thus, it seems most effective to offer longer-than-two-week vacations to customers with the following profile: relatively older, with high income, small number of children or no children at all, preferring to have sunshine available most of the time, and to a destination outside Europe.

4.6 Conclusion

We have analyzed a marketing classification problem with SVM for binary classification. We have also compared our results with those of standard marketing tools. Although the classical SVM exhibits superior performance, a general deficiency is that the results are hard to interpret, especially in the nonlinear case. To facilitate such an interpretation, we have constructed relationships between the explanatory and (binary) outcome variable by making use of probabilities for the SVM output estimates obtained from an approach proposed by Platt (1999). Ultimately, this allows for the possibility to evaluate the effectiveness of different marketing strategies under different scenarios. In terms of interpretation of the results, it appears that SVM models can give two advantages over standard techniques. First, highly nonmonotonic effects of the explanatory variables can be detected and visualized. And second, which comes as a by-product of the first, the SVM appears to model adequately the effects of arbitrarily coded unordered categorical variables.

Part II

Contributed New Instance-Based Kernel Penalization Methods

Chapter 5

Instance-Based Penalization Methods for Classification

In this chapter, several instance-based large-margin classifiers are put forward: Support Hyperplanes, Nearest Convex Hull classifier, and Soft Nearest Neighbor. The techniques are examined from a common fit-versus-complexity framework and study the links between them. Finally, the performance of these techniques is compared vis-a-vis each other as well as other standard classification methods.

5.1 Introduction

Recently, three classification methods have been introduced in the literature: Support Hyperplanes (SH) (Nalbantov et al., 2006a), Nearest Convex Hull classifier (NCH) (Nalbantov, Groenen, & Bioch, 2007) and Soft Nearest Neighbor (SNN) (Nalbantov, Bioch, & Groenen, 2008). All of them can be classified as instance-based large-margin penalization classifiers. In the following, we argue why these three techniques should perform well based on their favorable generalization qualities. We specifically look at links between Support Vector Machines (SVM), SH, NCH and SNN and approach them intuitively from a common generalization error-versus-complexity point of view. The instance-based nature of the SH, NCH, and SNN arises from the fact that these classifiers do not output an explicit formulation of a decision boundary between the classes. Rather, the classification of each test point is carried out independently of the classification of other test points.

The chapter is organized as follows. First, we briefly revise the role of penalization/capacity control for learners in general and argue that the error-versus-complexity paradigm (see e.g. Hastie et al., 2001; Vapnik, 1995; Shawe-Taylor

& Cristianini, 2004) could be applied to instance-based techniques. Second, we make an intuitive comparison between SVM, SH, NCH, and SNN in the so-called separable case, where the classes of the training data are perfectly divisible (or, separable) by a hyperplane. Finally, we present some empirical results and conclude.

5.2 Penalization in Learning

The need for penalization in learning techniques has long been discussed in both the statistical and artificial intelligence/machines learning/data mining communities. Examples of techniques that explicitly employ some kind of penalization are Ridge Regression, Lasso, Support Vector Machines, Support Vector Regression, etc. See, for example, Hastie et al. (2001) for a review of such methods. Penalization is referred to the practice of purposefully decreasing the ability of a given learner to cope with certain tasks. This ability is referred to as the learner's complexity or capacity (see e.g. Vapnik, 1995). Arguably, a decreased learner's capacity is responsible for a better prediction performance by mitigating the problem of overfitting. Data overfitting occurs when a learner fits the training data too well, producing very low amount of errors. The amount of errors is referred to as the empirical risk, the empirical error, or the loss. The main idea behind penalization techniques is that the sum empirical error plus capacity control term should be minimized to achieve good prediction results on new data, or in other words, to achieve good generalization ability. In general, if the empirical error over the training data set is rather small, implying a possible overfitting of the training data, then the capacity of the learner is expected to be high. Thus, the generalization sum – empirical error plus capacity – would be relatively high. Hence the need to put up with some increased empirical error over the training data set, which is to be more than offset by a decrease in the learner's capacity. The latter decrease could come about by explicitly penalizing in some way the class of functions to which a learner belongs.

Instance-based, or lazy classification techniques do not have an explicit rule or a decision boundary derived from the training data with which to classify all new observations, or instances. Rather, a new rule for classifying a test instance is derived each time such an instance is given to the learner. A good example of a lazy technique is k-Nearest Neighbor (kNN).

At first sight, a direct application of the idea for penalization on instance-based learners seems hard to materialize. The reason is that penalization in general is applied to a given class of functions, or learners. In the end, one optimal function out of this class should be chosen to classify *any* test observation. This optimal learner produces a minimal generalization sum. The idea for penalization can however also be applied to instance-based classifiers. In this case the function (taken from a given function class) that is used for the

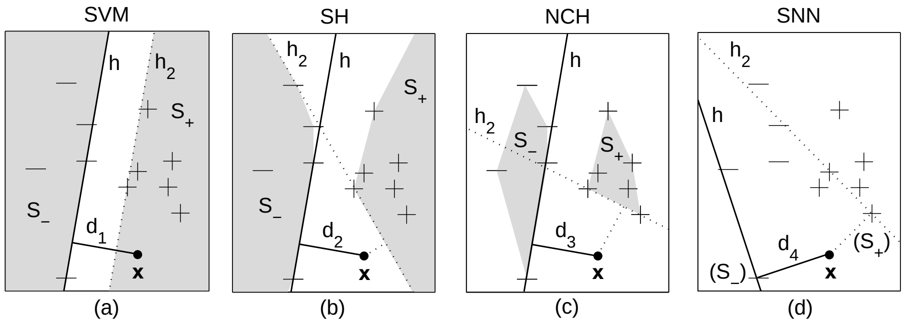


Figure 5.1: Binary classification with SVM, SH, NCH, and SNN in Panels (a), (b), (c) and (d), respectively. In all cases, the classification of test point $\mathbf{x}$ is determined using hyperplane h, which is in general different for each method. Equivalently, $\mathbf{x}$ is labeled $+1$ (-1) if it is farther away from set S_- (S_+).

classification of a *particular* test instance should be penalized.

Below we give an intuitive account of three rather new instance-based classification techniques, SH, NCH, and SNN. We approach them from a common generalization framework and discuss the links between them and SVM.

5.3 Three Instance-Based Classification Methods

Given a data set that is separable by a hyperplane and consists of positive and negative observations, let us assume that we would like to classify a new observation $\mathbf{x}$ using a hyperplane, denoted as h. There are two types of hyperplanes: (a) hyperplanes that classify correctly all training data points (called for short consistent hyperplanes) and (b) hyperplanes that do not classify correctly all training data points (called for short inconsistent hyperplanes). For the sake of clarity, we consider any hyperplane to be consistent if it does not misclassify any training points.

There are two main factors to be considered in choosing the appropriate h. First, h should not be too close to $\mathbf{x}$. Intuitively speaking, the farther h is from $\mathbf{x}$, the greater the confidence we have in the classification label h assigns to $\mathbf{x}$. Second, h should not make too many mistakes when it classifies the training data. If one chooses h to be extremely far from $\mathbf{x}$, then at one point h will misclassify either all positive or all negative observations. On the other hand, if h classifies correctly all training points, then h might be too close to $\mathbf{x}$, in which case our confidence in the label it assigns to $\mathbf{x}$ is smaller. Thus, in general one cannot have both a big distance between h and $\mathbf{x}$, and a big degree of consistency of h with respect to the training data. A balance between these two desirable properties has unavoidably to be sought. The strife to choose an h that is highly consistent with the training data is referred to as the strife to minimize the empirical risk, empirical error, or training error. The idea to demand h to be as far away from $\mathbf{x}$ as possible can be thought of as a sort of regularization or penalization: the smaller the distance between h and $\mathbf{x}$, the greater the penalty associated with the classification of $\mathbf{x}$. The intuitive assertion here is that the degree of penalization could be proxied by a certain distance. In sum, when classifying a test point $\mathbf{x}$ using a hyperplane, given a separable binary training data set, one is faced with the familiar penalty plus error paradigm (see e.g. Hastie et al., 2001; Vapnik, 1995; Shawe-Taylor & Cristianini, 2004). Below we cast four classification methods, SVM, SH, NCH, and SNN, in the light of this paradigm. The hyperplane h with which to classify a new observation $\mathbf{x}$ is in general different for each of these techniques. See Figure 5.1 for a running toy example.

The h hyperplane in Support Vector Machine classification (see Figure 1a) is defined as the farthest-away from $\mathbf{x}$ consistent hyperplane that is parallel to another consistent hyperplane, h_2, in such a way that the distance between these two hyperplanes (referred to as the "margin") is maximal. Since h is consistent with the training data, the empirical error it makes on the data is zero. The magnitude of the penalty associated with the classification of $\mathbf{x}$ can be considered to be positively related to the inverse of the distance between $\mathbf{x}$ and h ($1/d_1$ in terms of Figure 1a). The (theoretical) instance-based SVM classification algorithm can be stated as follows: first add $\mathbf{x}$ to the data set with -1 label and compute the distance d_1 to h (as defined above). Second, add $\mathbf{x}$ to the data set with $+1$ label and compute the distance d_1^* to h_2. Third, classify $\mathbf{x}$ using h (that is, as -1) if $d_1 > d_1^*$; classify $\mathbf{x}$ using h_2 (as $+1$) if $d_1 < d_1^*$; otherwise, if $d_1 = d_1^*$, the classification of $\mathbf{x}$ is undetermined.

The h hyperplane in SH classification (see Figure 1b) is defined as the farthest-away from $\mathbf{x}$ consistent hyperplane. Since h is consistent with the training data, the empirical error it makes on the data is zero. The magnitude of the penalty associated with the classification of $\mathbf{x}$ can be considered to be positively related to the inverse of the distance to h ($1/d_2$ in terms of Figure 1b). It can be shown that $d_2 \geq d_1$ always. Therefore, the sum empirical error plus penalty for SH is always smaller than the corresponding sum for SVM, suggesting that SH may possess better generalization ability than SVM. The SH classification algorithm can be stated as follows. First, add $\mathbf{x}$ to the training data set with -1 label and compute the distance d_2 to h. Note that h is consistent with both the original training data and with $\mathbf{x}$. That is, h assigns label -1 to $\mathbf{x}$. Second, add $\mathbf{x}$ to the original data set with $+1$ label and compute the distance d_2^* to h_2. In this case, h_2 is defined as the farthest-away hyperplane from $\mathbf{x}$ that is consistent with both $\mathbf{x}$ and the original training data. Third, classify $\mathbf{x}$ using h (that is, as -1) if $d_2 > d_2^*$; classify $\mathbf{x}$ using h_2 (as $+1$) if $d_2 < d_2^*$; otherwise, if $d_2 = d_2^*$, the classification of $\mathbf{x}$ is undetermined.

The h hyperplane in NCH classification (see Figure 1c) is defined as the farther of two hyperplanes. The first one is the hyperplane farthest away from $\mathbf{x}$ that is consistent with all positive observations and $\mathbf{x}$, where $\mathbf{x}$ has label -1. The second one if the hyperplane farthest away from $\mathbf{x}$ that is consistent with all the negative observations and $\mathbf{x}$, where $\mathbf{x}$ has label $+1$. Effectively, $\mathbf{x}$ is classified as $+1$ (-1) if it is closer to the convex hull of $+1$ (-1) points. The magnitude of the penalty associated with the classification of $\mathbf{x}$ is considered to be positively related to the inverse of the distance from $\mathbf{x}$ to h ($1/d_3$ in terms of Figure 1c). It can be shown that $d_3 \geq d_2 \geq d_1$ always. However, the empirical error on the training set is not guaranteed to be equal to zero. This happens because h should be consistent with at least all positive or all negative observations, and not with both all negative and all positive observations. Thus, the generalization sum training error plus penalty is not guaranteed to be smaller for NCH than for SH or SVM. The NCH classification algorithm can be stated as follows. First, add $\mathbf{x}$ to the training data set with -1 label and compute the distance d_3 to h, the hyperplane that is consistent with all $+1$ points and $\mathbf{x}$. This distance is the distance between $\mathbf{x}$ and the convex hull of the positive points. Second, add $\mathbf{x}$ to the training data set with $+1$ label and compute the distance d_3^* to h_2, the hyperplane that is consistent with all -1 points and $\mathbf{x}$. Third, classify $\mathbf{x}$ using h (that is, as -1) if $d_3 > d_3^*$; classify $\mathbf{x}$ using h_2 (as $+1$) if $d_3 < d_3^*$; otherwise, if $d_3 = d_3^*$, the classification of $\mathbf{x}$ is undetermined.

The SNN classification can also be presented along similar lines as SVM, SH, and NCH. In the separable case, SNN is equivalent to the classical First Nearest Neighbor (1NN) classifier. The h hyperplane in 1NN classification (see Figure 1d) is the farther of two hyperplanes. The first one is farthest away from $\mathbf{x}$ hyperplane that is consistent with the closest positive observation and $\mathbf{x}$, where $\mathbf{x}$ has label -1. The second hyperplane is the farthest away from $\mathbf{x}$ hyperplane that is consistent with the closest negative observation and $\mathbf{x}$, where $\mathbf{x}$ has label $+1$. Effectively, $\mathbf{x}$ is classified as $+1$ (-1) if its closest training point has label $+1$ (-1). The magnitude of the penalty associated with the classification of $\mathbf{x}$ is considered to be positively related to the inverse of the distance from $\mathbf{x}$ to h ($1/d_4$ in terms of Figure 1d). It can be shown that $d_4 \geq d_3 \geq d_2 \geq d_1$ always, suggesting (somewhat counterintuitively) that 1NN provides for the greatest penalization among the four techniques under consideration. However, the empirical error in 1NN on the training data set is certainly not guaranteed to be equal to zero. In fact, h is not even guaranteed to be consistent with either all positive or all negative points, as the case is in NCH classification, as well as in SH and SVM classification. Thus, the h hyperplane in 1NN is likely to commit the greatest amount of errors on the training data set as compared to SVM, SH and NCH. Consequently, the generalizability sum empirical error plus penalty may turn out to be the highest. Note however that it could also turn out to be the lowest for some $\mathbf{x}$, in which case 1NN exhibits the highest generalization ability. The 1NN classification algorithm can be (theoretically) stated as follows. First, add $\mathbf{x}$ to the training data set with label -1 and compute the distance d_4 to h, the hyperplane that is consistent with $\mathbf{x}$ and the closest positive point. Second, add $\mathbf{x}$ to the training data set with $+1$ label and compute the distance d_4^* to h_2, the hyperplane that is consistent with $\mathbf{x}$ and the closest negative point. Third, classify $\mathbf{x}$ using h (that is, as -1) if $d_4 > d_4^*$; classify $\mathbf{x}$ using h_2 (as $+1$) if $d_4 < d_4^*$; otherwise, if $d_4 = d_4^*$, the classification of $\mathbf{x}$ is undetermined.

5.4 Alternative Specifications

In the separable case, there is an alternative, but equivalent, formulation of the SVM, SH, NCH, and SNN techniques in terms of distances to sets as opposed to distances to hyperplanes. The corresponding sets for each technique are depicted in Figure 1 as shaded areas. A common classification rule for all methods can be defined as follows: a new point $\mathbf{x}$ should be classified as -1 if it is farther from set S_+ than from set S_-; $\mathbf{x}$ should be classified as $+1$ if it is farther from set S_- than from set S_+; otherwise, if the distance to both S_+ and S_- is the same, the class of $\mathbf{x}$ is undetermined as $\mathbf{x}$ lies on the decision boundary. Sets S_+ and S_- are defined differently for each method.

For SVM, set S_+ is defined as the set of all points that are classified as $+1$ by all hyperplanes that lie inside the SVM margin. Set S_- is similarly defined as the set of all points that are classified as -1 by all hyperplanes that lie inside the SVM margin.

For SH, set S_+ is the set of all points classified as $+1$ by all hyperplanes that are consistent with the training data. The latter include all hyperplanes that lie inside the SVM margin plus all the rest of the consistent hyperplanes. Analogically, set S_- is defined as the set of all points that are classified as -1 by all consistent hyperplanes. The collection of all consistent hyperplanes is referred to in the literature as the version space (Mitchell, 1997) of hyperplanes with respect to a given training data set. A conservative version-space classification rule is to classify a test point $\mathbf{x}$ only if all consistent hyperplanes assign one and the same classification label to it (Smirnov, Sprinkhuizen-Kuyper, Nalbantov, & Vanderlooy, 2006), or in other words if $\mathbf{x}$ belongs to either S_+ or S_-.

For the NCH classifier, set S_+ is the set of all points that are classified as $+1$ by all hyperplanes that are consistent with the positively-labeled data points. In other words, S_+ is the convex hull of the positive observations. Set S_- is defined as the set of all points that are classified as -1 by all hyperplanes that are consistent with the negatively-labeled data points. Thus, S_- is the convex hull of the negative points.

Lastly, for the 1NN classifier, which is the hard-margin version of the SNN classifier, the S_+ set consists of just one point: the closest to $\mathbf{x}$ positively-labeled point. Set S_- also consists of just one point: the closest to $\mathbf{x}$ negatively-labeled point.

5.5 Estimation

We now review the estimation of SVM, SH, NCH, and SNN. Further details can be found, e.g., in Burges (1998), Vapnik (1995), Nalbantov et al. (2006a, 2007, 2008). We examine a common setup for the four techniques: a binary classification data set $\{\mathbf{x}_i, y_i\}_{i=1}^{l}$, where each $\mathbf{x}_i$ is an n-dimensional vector of values for the predictor variables and each y_i is either a $+1$ or a -1 observation label. The classification task is: given a test point $\mathbf{x}$, output its predicted label. Each of the techniques solves an optimization problem to find an optimal hyperplane, $\mathbf{w}^{*\prime}\mathbf{x} + b^* = 0$, with which to classify the test observation in the way presented in Section 5.3. Here $\mathbf{w}$ is a vector of hyperplane coefficients, b is the intercept, and the asterisk $(^*)$ indicates optimal values.

5.5.1 Support Vector Machines

SVM solve the classification task by maximizing the so-called margin between the classes. In the separable case, the margin is equal to the distance between the convex hulls of the two classes at the optimal SVM solution (Vapnik, 1995). Formally, the margin is equal to the distance between hyperplanes $\mathbf{w}'\mathbf{x}+b = -1$ and $\mathbf{w}'\mathbf{x} + b = 1$, presented already as h and h_2 in Figure 1a. Thus, the margin equals $2/||\mathbf{w}||$. Maximizing the margin is equivalent to minimizing the term $||\mathbf{w}||^2/2 = \mathbf{w}'\mathbf{w}/2$. Formally, to find the SVM hyperplane h, one solves the following optimization problem:

$$\begin{aligned} \min_{\mathbf{w},b} \quad & \frac{1}{2}\mathbf{w}'\mathbf{w} \\ \text{s.t.} \quad & y_i(\mathbf{w}'\mathbf{x}_i + b) \geq 1, \; i = 1, 2, \ldots, l. \end{aligned} \tag{5.1}$$

If there is no hyperplane that is able to separate the classes, so-called slack variables ξ_i are introduced. This case is referred to as the nonseparable case or the class-overlapping case. Then, problem (5.1) becomes:

$$\begin{aligned} \min_{\mathbf{w},b,\boldsymbol{\xi}} \quad & \frac{1}{2}\mathbf{w}'\mathbf{w} + C\sum_{i=1}^{l}\xi_i \\ \text{s.t.} \quad & y_i(\mathbf{w}'\mathbf{x}_i + b) \geq 1 - \xi_i, \; \xi_i \geq 0, \; i = 1, 2, \ldots, l, \end{aligned} \tag{5.2}$$

where $C > 0$ is a manually adjustable constant that regulates the trade-off between the penalty term $\mathbf{w}'\mathbf{w}/2$ and the loss $\sum_{i=1}^{l}\xi_i$.

Optimization problem (5.2) can be dualized as:

$$\begin{array}{ll} \max\limits_{\boldsymbol{\alpha}} & \sum_{i=1}^{l} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{l} \alpha_i\alpha_j y_i y_j (\mathbf{x}_i'\mathbf{x}_j) \\ \text{s.t.} & 0 \leq \alpha_i \leq C,\ i = 1,2,\ldots,l, \text{ and } \sum_{i=1}^{l} y_i\alpha_i = 0, \end{array} \tag{5.3}$$

where the α_i's are the Lagrange multipliers associated with (5.2). The advantage of the dual is that different nonlinear mappings $\mathbf{x} \to \phi(\mathbf{x})$ of the data can easily handled. Thus, if one first transforms the data into a higher-dimensional space, where the coordinates of the data points are given by $\phi(\mathbf{x})$ instead of $\mathbf{x}$, then the dot product $\mathbf{x}_i'\mathbf{x}_j$ will appear as $\phi(\mathbf{x}_i)'\phi(\mathbf{x}_j)$ in the dual optimization problem. There exist so-called kernel functions $\kappa(\mathbf{x}_i, \mathbf{x}_j) \equiv \phi(\mathbf{x}_i)'\phi(\mathbf{x}_j)$ that compute this dot product efficiently, without explicitly carrying the transformation mapping. Popular kernels are the linear, $\kappa(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i'\mathbf{x}_j$, polynomial of degree d, $\kappa(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i'\mathbf{x}_j + 1)^d$ and the Radial Basis Function (RBF) kernel $\kappa(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma||\mathbf{x}_i - \mathbf{x}_j||^2)$. The mapping $\mathbf{x} \to \phi(\mathbf{x})$ when the RBF kernel is used corresponds to a mapping into an infinite-dimensional space. The manually-adjustable γ parameter of the RBF kernel determines the proximity of any two points in this infinite-dimensional space.

5.5.2 Support Hyperplanes

In the separable case, the h hyperplane in SH classification, with which to classify test point $\mathbf{x}$, can be found as the solution of the following optimization problem:

$$\begin{array}{ll} \min\limits_{\mathbf{w},b,y_{l+1}} & \frac{1}{2}\mathbf{w}'\mathbf{w} \\ \text{s.t.} & y_i(\mathbf{w}'\mathbf{x}_i + b) \geq 0,\ i = 1,2,\ldots,l \\ & y_{l+1}(\mathbf{w}'\mathbf{x} + b) = 1, y_{l+1} \in \{-1, 1\}. \end{array} \tag{5.4}$$

This problem is partially combinatorial due to the constraint that the predicted label of $\mathbf{x}$, y_{l+1}, can take on only two values. Therefore, one usually solves two separate optimization subproblems: in the first one $y_{l+1} = +1$, and in the second one $y_{l+1} = -1$. The value of y_{l+1} that minimizes the objective function in (5.4) is the predicted label of $\mathbf{x}$. Note that the distance between $\mathbf{x}$ and h is defined as $1/\sqrt{\mathbf{w}^{*\prime}\mathbf{w}^*}$ by the equality constraint $y_{l+1}(\mathbf{w}'\mathbf{x} + b) = 1$.

In the nonseparable case, SH introduce slack variables ξ_i, similarly to SVM. As a result, the nonseparable version of (5.4) becomes:

$$
\begin{aligned}
\min_{\mathbf{w},b,y_{l+1},\boldsymbol{\xi}} \quad & \frac{1}{2}\mathbf{w}'\mathbf{w} + C\sum_{i=1}^{l}\xi_i \\
\text{s.t.} \quad & y_i(\mathbf{w}'\mathbf{x}_i + b) \geq 0 - \xi_i,\ \ \xi_i \geq 0,\ \ i = 1, 2, \ldots, l \\
& y_{l+1}(\mathbf{w}'\mathbf{x}_{l+1} + b) = 1, y_{l+1} \in \{-1, 1\}.
\end{aligned}
\tag{5.5}
$$

As in (5.4), two separate optimization problems have to be solved to determine the optimal y_{l+1}. Each of these two subproblems can be dualized as:

$$
\begin{aligned}
\max_{\boldsymbol{\alpha}} \quad & \alpha_{l+1} - \tfrac{1}{2}\textstyle\sum_{i,j=1}^{l+1} \alpha_i\alpha_j y_i y_j (\mathbf{x}_i'\mathbf{x}_j) \\
\text{s.t.} \quad & 0 \leq \alpha_i \leq C,\ \ i = 1, 2, \ldots, l,\ \text{ and } \textstyle\sum_{i=1}^{l+1} y_i\alpha_i = 0.
\end{aligned}
\tag{5.6}
$$

Similarly to SVM, different kernels can be substituted for the dot product $\mathbf{x}_i'\mathbf{x}_j$.

5.5.3 Nearest Convex Hull Classifier

The optimization problem for the NCH classifier is almost identical to the SH one. The only difference is that in each of the two optimization subproblems observations from only one class are considered. This property enables NCH to handle the multi-class classification case with ease, unlike SVM and SH. In the two-class problem at hand, let us denote with S_+ the set of observations that belong to the positive class and with S_- the set of observations that belong to the negative class. Next, two optimization problems are solved, one per each class k:

$$
\begin{aligned}
\min_{\mathbf{w}_k,b_k} \quad & \frac{1}{2}\mathbf{w}_k'\mathbf{w}_k \\
\text{s.t.} \quad & \mathbf{w}_k'\mathbf{x}_i + b_k \geq 0,\ \ i \in S_k \\
& -(\mathbf{w}_k'\mathbf{x} + b_k) = 1.
\end{aligned}
\tag{5.7}
$$

The distance from $\mathbf{x}$ to the k^{th} class is defined as $1/\sqrt{\mathbf{w}_k^{*\prime}\mathbf{w}_k^*}$ by the equality constraint in (5.7). The class associated with the smallest such distance is assigned to the test point $\mathbf{x}$. Notice that this distance is inversely related to the objective function $\mathbf{w}_k'\mathbf{w}_k/2$. Therefore, the class k that achieves the maximal value for this objective function should be assigned to $\mathbf{x}$.

In the nonseparable case, each of the optimization subproblems is expressed as:

$$\begin{array}{ll} \min\limits_{\mathbf{w}_k, b_k, \boldsymbol{\xi}} & \frac{1}{2}\mathbf{w}_k'\mathbf{w}_k + C\sum\limits_{i\in S_k}\xi_i \\ \text{s.t.} & \mathbf{w}_k'\mathbf{x}_i + b_k \geq 0 - \xi_i,\ \xi_i \geq 0,\ i \in S_k \\ & -(\mathbf{w}_k'\mathbf{x} + b_k) = 1, \end{array} \tag{5.8}$$

where the ξ's are slack variables. In dual form, (5.8) becomes:

$$\begin{array}{ll} \max\limits_{\boldsymbol{\alpha}} & \alpha_{l_k+1} - \frac{1}{2}\sum_{i,j=1}^{l_k+1}\alpha_i\alpha_j y_i y_j(\mathbf{x}_i'\mathbf{x}_j) \\ \text{s.t.} & 0 \leq \alpha_i \leq C,\ i = 1,2,\ldots,l_k,\ \text{and}\ \sum_{i=1}^{l_k+1} y_i\alpha_i = 0, \end{array} \tag{5.9}$$

allowing for the employment of kernel functions, as in SVM and SH. Here $i = 1,2,\ldots,l_k$ denotes the elements of class k.

5.5.4 Soft Nearest Neighbor

In the separable case, SNN is equivalent to the 1NN classifier. Instead of computing the distances between $\mathbf{x}$ and all data points to determine the nearest neighbor of $\mathbf{x}$ however, SNN take a different approach. Observe that the distance to the nearest neighboring point is equal to the maximal radius of a (hyper)sphere with center $\mathbf{x}$ that does not contain any training data points. To find this radius r, one solves the following optimization problem:

$$\begin{array}{ll} \max & r^2 \\ \text{s.t.} & r^2 \leq ||\mathbf{x}_i - \mathbf{x}||^2,\ i = 1,2,\ldots,l. \end{array} \tag{5.10}$$

In SNN classification, one first finds the distances between $\mathbf{x}$ and the closest point from each of the two (or, in general k) classes. Point $\mathbf{x}$ is then assigned to the class, which such point is closer/closest to $\mathbf{x}$. Denoting with S_+ and S_- the sets of positive and negative observations, respectively, SNN thus solve one optimization problem per each class k, of the form:

$$\begin{array}{ll} \max & r^2 \\ \text{s.t.} & r^2 \leq ||\mathbf{x}_i - \mathbf{x}||^2,\ i \in S_k. \end{array} \tag{5.11}$$

The class that produces the minimal value for the objective function $\mathbf{R}^2$ of (5.11) is then assigned to point $\mathbf{x}$. Similarly to the SVM, SH and NCH approaches, one can introduce slack variables ξ_i. In this case (5.11) becomes:

$$\begin{aligned} \max \quad & \mathbf{R}^2 - C \sum_{i \in S_k} \xi_i \\ \text{s.t.} \quad & \mathbf{R}^2 \leq \; ||\mathbf{x}_i - \mathbf{x}||^2 + \; \xi_i, \; \xi_i \geq 0, \; i \in S_k. \end{aligned} \tag{5.12}$$

The $C > 0$ parameter controls the trade-off between the length of the radius and amount of training errors. A training error occurs if a point lies inside the hypersphere. Each of the k quadratic optimization problems (5.12) can be expressed in dual form as:

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \quad & \sum_{i \in S_k} \alpha_i (\mathbf{x}_i' \mathbf{x}_i - 2(\mathbf{x}_i' \mathbf{x}) + \mathbf{x}' \mathbf{x}) \\ \text{s.t.} \quad & 0 \leq \alpha_i \leq C, \; i \in S_k, \; \text{and} \; \sum_{i \in S_k} \alpha_i = 1. \end{aligned} \tag{5.13}$$

This formulation allows for the employment of different kernels, which can replace the dot products $\mathbf{x}_i'\mathbf{x}_i$, $\mathbf{x}_i'\mathbf{x}$ and $\mathbf{x}'\mathbf{x}$. Notice that unlike (5.12), (5.13) is a linear programming problem.

5.6 Comparison Results

The basic optimization algorithms for SH, NCH, and SNN classification, (5.6), (5.9) and (5.13) respectively, are implemented via a modification of the freely available LIBSVM software (Chang & Lin, 2006). We tested the performance of SH, NCH, and SNN on several small- to middle-sized data sets that are freely available from the SlatLog and UCI repositories (Newman, Hettich, Blake, & Merz, 1998) and have been analyzed by many researchers and practitioners (see, among others, Breiman, 1996; King, Feng, & Sutherland, 1995; Lim, Loh, & Shih, 1995; Perlich, Provost, & Simonoff, 2003): *Sonar*, *Voting*, *Wisconsin Breast Cancer* (W.B.C.), *Heart*, *Australian Credit Approval* (A.C.A.), and *Hepatitis* (Hep.). Detailed information on these data sets can be found on the web sites of the respective repositories. We stop short of carrying out an extensive experimental study, since this falls out of the main scope of the chapter. Furthermore, large data sets are harder to handle due to the instance-based nature of the SH, NCH, and SNN classifiers.

We compare the results of SH, NCH and SNN to those of several state-of-art techniques: Support Vector Machines (SVM), Linear and Quadratic Discriminant Analysis (LDA and QDA), Logistic Regression (LR), Multi-layer Perceptron (MLP), k-Nearest Neighbor (kNN), Naive Bayes classifier (NB) and two types of Decision Trees – Decision Stump (DS) and C4.5. The experiments for the NB, LR, MLP, kNN, DS and C4.5 methods have been carried out with

Table 5.1: Leave-one-out accuracy rates (in %) of the Support Hyperplanes (SH), Nearest Convex Hull (NCH) and Soft Nearest Neighbor (SNN) classifiers as well as some standard methods on several data sets. Rbf, 2p and lin stand for Radial Basis Function, second-degree polynomial and linear kernel, respectively

	Sonar	Voting	W.B.C.	Heart	A.C.A.	Hep.
SH rbf	**91.35**	96.77	**97.42**	**85.56**	**87.39**	**87.7**
SH 2p	87.98	96.31	96.85	81.90	86.70	86.45
SH lin	79.80	96.77	97.00	**85.56**	86.80	86.45
NCH rbf	**91.35**	95.85	**97.42**	**85.56**	86.38	85.16
NCH 2p	90.38	85.48	97.14	82.59	85.36	84.52
NCH lin	87.98	95.85	97.28	84.07	86.09	84.52
SNN rbf	88.46	94.47	**97.42**	85.19	85.80	87.10
SNN 2p	76.92	94.01	97.28	80.74	85.51	85.16
SNN lin	87.50	93.78	97.28	**85.56**	85.65	85.16
SVM rbf	88.94	96.54	97.00	**85.56**	**87.39**	86.45
SVM 2p	82.21	96.31	96.85	81.11	79.86	86.45
SVM lin	80.77	96.77	96.85	**85.56**	87.10	86.45
NB	67.30	90.32	95.99	82.96	77.10	83.23
LR	73.08	96.54	96.14	83.70	86.38	83.87
LDA	75.48	95.85	95.99	83.70	85.80	85.81
QDA	74.88	94.24	91.42	81.48	85.22	83.87
MLP	81.25	94.93	94.99	78.89	84.78	79.35
*k*NN	86.54	93.32	97.00	84.44	85.94	85.81
DS	73.08	95.85	92.42	76.30	85.51	79.35
C4.5	71.15	**97.00**	95.28	75.19	83.77	80.00

the WEKA learning environment using default model parameters, except for kNN. We refer to Witten and Frank (2005) for additional information on these classifiers and their implementation. We measure model performance by the leave-one-out (LOO) accuracy rate. For our purposes – comparison between the methods – LOO seems to be more suitable than the more general k-fold cross-validation (CV), because it always yields one and the same error rate estimate for a given model, unlike the CV method, because it involves a random split of the data into several parts.

Table 1 presents performance results for all methods considered. Some methods, namely kNN, SH, NCH, SNN and SVM, require tuning of model parameters. In these cases, we report only the highest LOO accuracy rate obtained by performing a grid search for tuning the necessary parameters.

Overall, the instance-based penalization classifiers SH, NCH and SNN perform quite well on all data sets. Most notably, SH achieve best accuracy rates on five data sets. NCH replicate this success three times. SVM also perform best on three data sets. The SNN classifier achieves best accuracy rate on just two data sets, but five times out of six performs better than its direct competitor, kNN. The rest of the techniques show relatively less favorable and more volatile results. For example, the C4.5 classifier performs best on the *Voting* data set, but achieves rather low accuracy rates on two other data sets – *Sonar* and *Heart*. Note that not all data sets are equally easy to handle. For instance, the performance variation over all classifiers on the *Voting* and *Breast Cancer* data sets is rather low, whereas on the *Sonar* data set it is quite substantial.

5.7 Conclusion

We have studied from a common generalization perspective three classification methods recently introduced in the literature: Support Hyperplanes, Nearest Convex Hull classifier and Soft Nearest Neighbor. In addition, we have compared them to the popular Support Vector Machines. A common theme in SH, NCH, and SNN is their instance-based nature. In addition, these methods strive to find a balance between learner's capacity and learner's fit over the training data. Last but not least, the techniques can be kernelized, which places them also in the realm of kernel methods. We have provided a rather intuitive treatment of these techniques and the generalization framework from which they are approached. Further research could concentrate on more detailed such treatment and on the derivation of theoretical test-error bounds. Extensive experiments with different loss functions, such as the quadratic one, have also to be carried out. Last but not least, ways to improve the computational speed can also be explored.

Chapter 6

Classification with Support Hyperplanes*

A new classification method is proposed, called Support Hyperplanes (SHs). To solve the binary classification task, SHs consider the set of all hyperplanes that do not make classification mistakes, referred to as semi-consistent hyperplanes. A test object is classified using that semi-consistent hyperplane, which is farthest away from it. In this way, a good balance between goodness-of-fit and model complexity is achieved, where model complexity is proxied by the distance between a test object and a semi-consistent hyperplane. This idea of complexity resembles the one imputed in the width of the so-called margin between two classes, which arises in the context of Support Vector Machine learning. Class overlap can be handled via the introduction of kernels and/or slack variables. The performance of SHs against standard classifiers is promising on several widely-used empirical data sets.

6.1 Introduction

Consider the task of separating two classes of objects from each other on the basis of some shared characteristics. In general, this separation problem is referred to as the (binary) classification task. Some well-known approaches to this task include (binary) Logistic Regression, k-Nearest Neighbor, Decision Trees, Naive Bayes classifier, Linear and Quadratic Discriminant Analysis, Neural Networks, and more recently, Support Vector Machines (SVMs).

*This chapter has been published as Nalbantov et al. (2006a):
Nalbantov, G. I., Bioch, J. C., & Groenen, P. J. F. (2006a). Classification with Support Hyperplanes. In J. Fürnkranz, T. Scheffer, & M. Spiliopoulou (Eds.), *ECML 2006: 17th European Conference on Machine Learning* (pp. 703–710). Springer Berlin/Heidelberg.

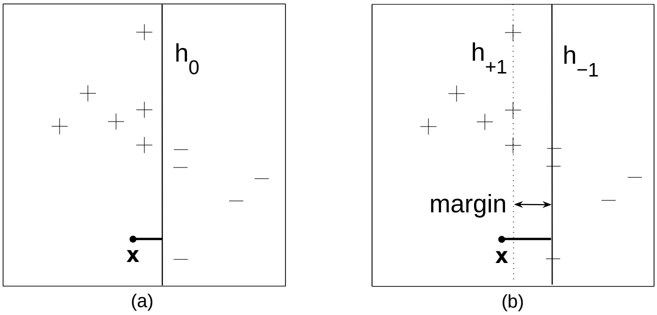


Figure 6.1: Two equivalent ways to apply the SVMs classification rule. In Panel (a), the test point $\mathbf{x}$ receives the label (+1) assigned using hyperplane h_0, $\sum_{i=1}^{l} y_i \alpha_i \kappa(\mathbf{x}_i, \mathbf{x}) + b = 0$. In Panel (b), the same test point receives the label (+1) assigned using the farthest away semi-consistent hyperplane from it (h_{-1}, $\sum_{i=1}^{l} y_i \alpha_i \kappa(\mathbf{x}_i, \mathbf{x}) + b = -1$), which is parallel to another semi-consistent hyperplane (h_{+1}, $\sum_{i=1}^{l} y_i \alpha_i \kappa(\mathbf{x}_i, \mathbf{x}) + b = 1$) in such a way that the distance between these two hyperplanes is maximal.

Support Hyperplanes (SHs) is a new instance-based large margin classification technique that provides an implicit decision boundary using a set of explicitly defined functions. For SHs, this set consists of all hyperplanes that do not misclassify any of the data objects. Each hyperplane that belongs to this set is called a *semi-consistent* hyperplane with respect to the data. We first treat the so-called separable case – the case where the classes are perfectly separable by a hyperplane. Then we deal with the nonseparable case via the introduction of kernels and slack variables, similarly to SVMs. The basic motivation behind SHs is the desire to classify a given test object with that semi-consistent hyperplane, which is most likely to classify this particular object correctly. Since for each new object there is a different such semi-consistent hyperplane, the produced decision surface between the classes is implicit.

An advantage of the SHs method is that it is robust against outliers and avoids overfitting. Further, we demonstrate empirically that the SHs decision boundary appears to be relatively insensitive to the choice of kernel applied to the data. The SHs approach is more conservative than SVMs, for instance, in the sense that the hyperplane determining the classification of a new object is more distant from it than any of the hyperplanes forming the so-called margin in SVMs. It can be argued that the SHs approach is more general than SVMs by means of a formulation of the SHs decision boundary that is nested into the formulation of the SVMs decision boundary.

6.2 Support Vector Machines for Classification

We start with an account of the SVM classifier, developed by Vapnik (Vapnik, 1995) and co-workers. SVMs for binary classification solve the following task: given training data $\{\mathbf{x}_i, y_i\}_{i=1}^{l}$ from $\mathbb{R}^n \times \{-1, 1\}$, estimate a function $f : \mathbb{R}^n \to \{-1, 1\}$ such that f will classify correctly unseen observations $\{\mathbf{x}_j, y_j\}_{j=l+1}^{l+1+m}$. In SVMs, the input vectors $\{\mathbf{x}_i\}_{i=1}^{l}$ are usually mapped from $\mathbb{R}^n$ into a higher-dimensional space via a mapping φ, in which the vectors are denoted as $\{\varphi(\mathbf{x}_i)\}_{i=1}^{l}$. In this higher-dimensional (or, feature) space, the SVM method finds the hyperplane that maximizes the closest distance between the observations from the two classes, the so-called margin, while at the same time minimizes the amount of training errors (Burges, 1998; Cristianini & Shawe-Taylor, 2000; Vapnik, 1995). The optimal SVM hyperplane is found by solving the following quadratic optimization problem:

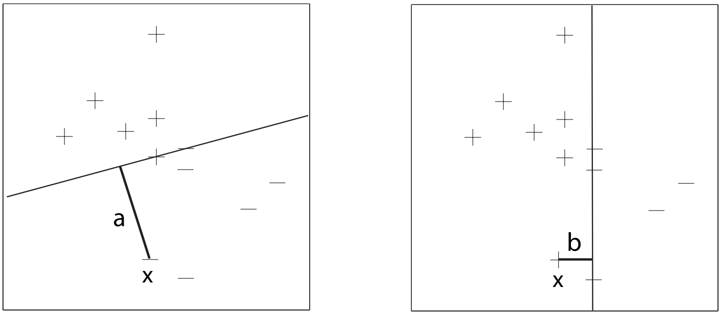
$$\begin{array}{ll} \max\limits_{\boldsymbol{\alpha}} & \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{l} \alpha_i \alpha_j y_i y_j \kappa(\mathbf{x}_i, \mathbf{x}_j) \\ \text{s.t.} & 0 \leq \alpha_i \leq C, \ i = 1, 2, \ldots, l, \text{ and } \sum_{i=1}^{l} y_i \alpha_i = 0, \end{array} \tag{6.1}$$

where $\kappa(\mathbf{x}_i, \mathbf{x}_j) = \varphi(\mathbf{x}_i)'\varphi(\mathbf{x}_j)$ is a *Mercer* kernel that calculates inner product of input vectors $\mathbf{x}_i$ and $\mathbf{x}_j$ mapped in feature space. Using the optimal α's of (6.1) the SVM hyperplane h_0, $\mathbf{w}'\varphi(\mathbf{x}) + b = 0$, can be expressed as $\sum_{i=1}^{l} y_i \alpha_i \kappa(\mathbf{x}_i, \mathbf{x}) + b = 0$. Here, $\mathbf{w}$ is a vector of hyperplane coefficients, and b is the intercept. A test observation $\mathbf{x}$ receives the class label assigned using h_0, as shown in Fig. 6.1a. Stated equivalently, $\mathbf{x}$ is classified using the farthest-away hyperplane that is semi-consistent with the training data, which is parallel to another semi-consistent hyperplane in such a way that the distance between these two hyperplanes is maximal (see Fig. 6.1b).

6.3 Support Hyperplanes

6.3.1 Definition and Motivation

Just like SVMs, the SHs address the classification task. Let us focus on the so-called linearly separable case, where the positive and negative observations of a training data set D are perfectly separable from each other by a hyperplane. Consider the set of semi-consistent hyperplanes. Formally, a hyperplane with equation $\mathbf{w}'\mathbf{x} + b = 0$ is defined to be semi-consistent with a given data set if for all data points $i = 1, 2, \ldots, l$, it holds that $y_i(\mathbf{w}'\mathbf{x}_i + b) \geq 0$; the same hyperplane is defined to be *consistent* with the data if for all data points $i = 1, 2, \ldots, l$, it holds that $y_i(\mathbf{w}'\mathbf{x}_i + b) > 0$. The basic motivation behind Support Hyperplanes (SHs) is the desire to classify a test observation $\mathbf{x}$ with that semi-consistent hyperplane, which is in some sense the most likely to assign the correct label to $\mathbf{x}$. The extent of such likeliness is assumed to be positively related to the distance



(a)

(b)

Figure 6.2: Classification with Support Hyperplanes in two steps. At stage one (Panel (a)), a test point **x** is added as class "–" to the original data set that consists of "+" and "–" labeled points, and the distance a from **x** to the farthest away semi-consistent hyperplane is computed. At stage two (Panel (b)), **x** is added to the original data set as class "+", and the distance b from **x** to the farthest away semi-consistent hyperplane is computed. If $a > b$ ($a < b$), then **x** is assigned to class "–" ("+").

between **x** and any semi-consistent hyperplane. Thus, if **x** is more distant from hyperplane h_a than from hyperplane h_b, both of which are semi-consistent with D, then h_a is considered more likely to classify **x** correctly than hyperplane h_b. This leads to the following classification rule of SHs: *a test point* **x** *should be classified using the farthest-away hyperplane from* **x** *that is semi-consistent with the training data.* Intuitively, this hyperplane can be called the "support hyperplane" since it supports its own judgement about the classification of **x** with greatest self-confidence; hence the name Support Hyperplanes for the whole method. For each test point **x** the corresponding support hyperplane is different. Therefore, the entire decision boundary between the two classes is not explicitly computed. A point is defined to lie on the SHs decision boundary if there exist two different semi-consistent hyperplanes that are farthest away from it. SHs consider the distance between a test point **x** and a semi-consistent hyperplane as a proxy for complexity associated with the classification of **x**. Under this circumstance, the best generalizability is achieved when one classifies **x** with the so-called support hyperplane: the semi-consistent hyperplane that is most distant from **x**. If one however considers the width of the margin as a proxy for complexity, then the SVM hyperplane achieves the best generalizability. Notice that by definition the support hyperplane is at least as distant from **x** as any of the two semi-consistent hyperplanes that form the margin of the optimal SVM hyperplane, which makes the SHs method relatively more conservative. Let us now argue more formally that the SH approach generalizes SVM by means of a formulation of the SHs decision boundary that is part of a formulation of the SVMs decision boundary. A point $\mathbf{x}$ is defined to lie on the implicit SHs separation surface if the following three conditions are met: (1) $\mathbf{x}$ is equally distant from two hyperplanes, (2) these two hyperplanes are semi-consistent with the training data, and (3) the distance between point $\mathbf{x}$ and any of the two hyperplanes is maximal. Next, observe that a point $\mathbf{x}$ is defined to lie on the explicit SVMs optimal hyperplane if and only if the three conditions above plus an additional fourth condition are all satisfied: (4) the two (semi-consistent) hyperplanes are parallel to each other.

6.3.2 Estimation

Given a linearly separable data set D, $\{\mathbf{x}_i, y_i\}_{i=1}^{l}$, from $\mathbb{R}^n \times \{-1, 1\}$, SHs classify a test point $\mathbf{x}_{l+1}$ using that semi-consistent hyperplane with respect to D, which is most distant from $\mathbf{x}_{l+1}$. Formally, in order to find the support hyperplane $\mathbf{w}'\mathbf{x} + b = 0$ of point $\mathbf{x}_{l+1}$, one solves the following quadratic optimization problem:

$$\begin{aligned} \min_{\mathbf{w}, b, y_{l+1}} \quad & \frac{1}{2}\mathbf{w}'\mathbf{w} \\ \text{s.t.} \quad & y_i(\mathbf{w}'\mathbf{x}_i + b) \geq 0, \ i = 1, 2, \ldots, l \\ & y_{l+1}(\mathbf{w}'\mathbf{x}_{l+1} + b) = 1 \end{aligned} \tag{6.2}$$

The distance between the support hyperplane $\mathbf{w}'\mathbf{x} + b = 0$ and $\mathbf{x}_{l+1}$ is defined as $1/\sqrt{\mathbf{w}'\mathbf{w}}$ by the last constraint of (6.2), irrespective of the label y_{l+1}. This distance is maximal when $\frac{1}{2}\mathbf{w}'\mathbf{w}$ is minimal. The role of the first l inequality constraints is to ensure that the support hyperplane is semi-consistent with the training data.

Optimization problem (6.2) is partially combinatorial, since not all variables are continuous: the label y_{l+1} can take only two discrete values. Therefore, in order to solve (6.2), two distinct optimization subproblems should we solved (see Fig. 6.2). One time (6.2) is solved when y_{l+1} equals $+1$, and another time when y_{l+1} equals -1. Each of these optimization subproblems has a unique solution, provided that the extended data set $\{\mathbf{x}_i, y_i\}_{i=1}^{l+1}$ is separable. In case the two solutions yield the same value for the objective function $\frac{1}{2}\mathbf{w}'\mathbf{w}$, the test point $\mathbf{x}_{l+1}$ lies on the SHs decision boundary and the classification label is undetermined. If the extended data set has become nonseparable when y_{l+1} is labeled, say, $+1$, then the respective optimization subproblem does not have a solution. Then, $\mathbf{x}_{l+1}$ is assigned the opposite label, here -1.

The implicit nature of SHs provides for the property that the SHs decision boundary is in general nonlinear, even in case the original data is not mapped into a higher-dimensional space. Figure 6.3 demonstrates that this property does not hold in general for SVMs. This figure also illustrates that the SHs decision boundary appears to be less sensitive to the choice of kernel and kernel parameters than the respective SVMs boundary.

We now treat the so-called (linearly) nonseparable case. A training data set is said to be nonseparable if there does not exist a single hyperplane that is consistent with it. SHs deal with the nonseparable case in the same way as SVMs: by introducing so-called slack variables. For SHs, this procedure amounts to solving the following quadratic optimization problem:

$$\begin{aligned} \min_{\mathbf{w},b,y_{l+1},\boldsymbol{\xi}} \quad & \frac{1}{2}\mathbf{w}'\mathbf{w} + C\sum_{i=1}^{l}\xi_i \\ \text{s.t.} \quad & y_i(\mathbf{w}'\mathbf{x}_i + b) \geq 0 - \xi_i,\ \xi_i \geq 0,\ i = 1, 2, \ldots, l \\ & y_{l+1}(\mathbf{w}'\mathbf{x}_{l+1} + b) = 1. \end{aligned} \tag{6.3}$$

Note that in (6.3) the points that are incorrectly classified are penalized linearly via $\sum_{i=1}^{l}\xi_i$. If one prefers a quadratic penalization of the classification errors, then the sum of squared errors $\sum_{i=1}^{l}\xi_i^2$ should be substituted for $\sum_{i=1}^{l}\xi_i$ in (6.3). One can go even further and extend the SHs algorithm in a way analogical to LS-SVM (Gestel et al., 2004) by imposing in (6.3) that constraints $y_i(\mathbf{w}'\mathbf{x}_i + b) \geq 0 - \xi_i$ hold as equalities, on top of substituting $\sum_{i=1}^{l}\xi_i^2$ for $\sum_{i=1}^{l}\xi_i$.

Each of the two primal subproblems pertaining to (6.3) can be expressed in dual form[1] as:

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & \alpha_{l+1} - \tfrac{1}{2}\textstyle\sum_{i,j=1}^{l+1}\alpha_i\alpha_j y_i y_j (\mathbf{x}_i'\mathbf{x}_j) \\ \text{s.t.} \quad & 0 \leq \alpha_i \leq C,\ i = 1, 2, \ldots, l,\ \text{and}\ \textstyle\sum_{i=1}^{l+1} y_i\alpha_i = 0, \end{aligned} \tag{6.4}$$

where the α's are the Lagrange multipliers associated with the respective subproblem. In the first subproblem $y_{l+1} = 1$, while in the second subproblem $y_{l+1} = -1$. The advantage of the dual formulation (6.4) is that different *Mercer* kernels can be employed to replace the inner product $\mathbf{x}_i'\mathbf{x}_j$ in (6.4), just like in the SVMs case. The $(l+1) \times (l+1)$ symmetric positive-definite matrix with elements $\varphi(\mathbf{x}_i)'\varphi(\mathbf{x}_j)$ on the i^{th} row and j^{th} column is called the kernel matrix.

The SHs approach can also be theoretically justified by observing that a kernel matrix used by SHs can be modified to represent the original SHs optimization problem as an SVM problem. It turns out that the theoretical underpinnings for SVMs can also be transferred to the SHs method. More details will be provided in Nalbantov, Bioch, and Groenen (2006c).

[1] The derivation of the dual problem resembles the one used in SVMs (see, e.g., Burges, 1998).

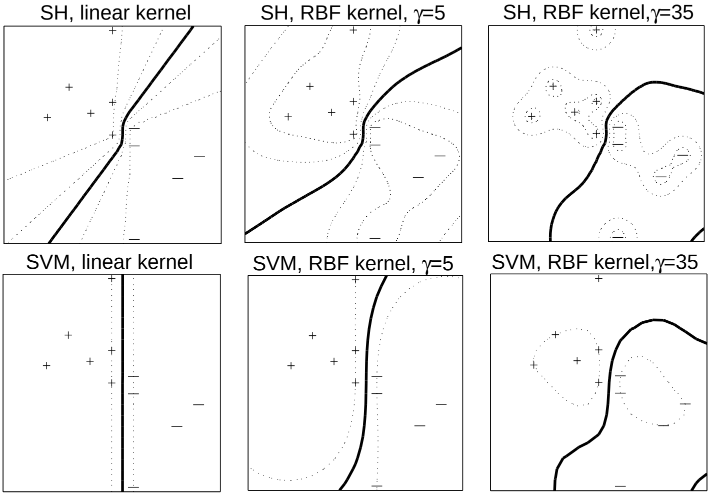


Figure 6.3: Decision boundaries for SHs and SVMs using the linear, $\kappa(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i'\mathbf{x}_j$, and the RBF, $\kappa(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma \parallel \mathbf{x}_i - \mathbf{x}_j \parallel^2)$, kernels on a linearly separable data set. The dashed contours for the SHs method are iso-curves along which the ratio of two distances is constant: the distance from a test point to the farthest semi-consistent hyperplane when it is added to the data set one time as "+", and another time as "–".

6.4 Experiments on Some UCI and SlatLog Data Sets

The basic optimization algorithm for Support Hyperplanes (6.4) is implemented via a modification of the freely available LIBSVM software (Chang & Lin, 2006). We tested the performance of Support Hyperplanes on several small- to middle-sized binary data sets that are freely available from the SlatLog and UCI repositories (Newman et al., 1998) and have been analyzed by many researchers and practitioners (e.g., among others, Breiman, 1996; King et al., 1995; Lim et al., 1995; Perlich et al., 2003): *Sonar*, *Voting*, *Wisconsin Breast Cancer* (W.B.C.), *Heart*, *Australian Credit Approval* (A.C.A.), and *Hepatitis* (Hep.). Detailed information on these data sets can be found on the web sites of the respective repositories.

We compare the results of SHs to those of several state-of-art techniques: Linear and Quadratic Discriminant Analysis (LDA and QDA), Logistic Regression (LR), Multi-layer Perceptron (MLP), k-Nearest Neighbor (kNN), Naive Bayes classifier (NB) and two types of Decision Trees – Decision Stump (DS) and C4.5. The experiments for the NB, LR, MLP, kNN, DS and C4.5 meth-

Table 6.1: Leave-one-out accuracy rates (in %) of the Support Hyperplanes classifier as well as some standard methods on several binary data sets. Rbf, 2p and lin stand for Radial Basis Function, second-degree polynomial and linear kernel, respectively

	Sonar	Voting	W.B.C.	Heart	A.C.A.	Hep.
SH rbf	**91.35**	96.77	**97.42**	**85.56**	**87.39**	**87.7**
SH 2p	87.98	96.31	96.85	81.90	86.70	86.45
SH lin	79.80	96.77	97.00	**85.56**	86.80	86.45
SVM rbf	88.94	96.54	97.00	**85.56**	**87.39**	86.45
SVM 2p	82.21	96.31	96.85	81.11	79.86	86.45
SVM lin	80.77	96.77	96.85	**85.56**	87.10	86.45
NB	67.30	90.32	95.99	82.96	77.10	83.23
LR	73.08	96.54	96.14	83.70	86.38	83.87
LDA	75.48	95.85	95.99	83.70	85.80	85.81
QDA	74.88	94.24	91.42	81.48	85.22	83.87
MLP	81.25	94.93	94.99	78.89	84.78	79.35
kNN	86.54	93.32	97.00	84.44	85.94	85.81
DS	73.08	95.85	92.42	76.30	85.51	79.35
C4.5	71.15	**97.00**	95.28	75.19	83.77	80.00

ods have been carried out with the WEKA learning environment using default model parameters, except for kNN. We refer to Witten and Frank (2005) for additional information on these classifiers and their implementation. We measure model performance by the leave-one-out (LOO) accuracy rate. For our purposes – comparison between the methods – LOO seems to be more suitable than the more general k-fold cross-validation (CV), because it always yields one and the same error rate estimate for a given model, unlike the CV method (which involves a random split of the data into several parts). Table 1 presents performance results for all methods considered. Some methods, namely kNN, SHs and SVMs, require tuning of model parameters. In these cases, we report only the highest LOO accuracy rate obtained by performing a grid search for tuning the necessary parameters. Overall, the accuracy rates of Support Hyperplanes exhibit first-rate performance on all six data sets: five times out of six the accuracy rate of SHs is the highest one. SVMs follow closely, and the rest of the techniques show relatively less favorable and more volatile results. For example, the C4.5 classifier performs best on the *Voting* data set, but achieves rather low accuracy rates on two other data sets – *Sonar* and *Heart*. Note that not all data sets are equally easy to handle. For instance, the performance variation over all classifiers on the *Voting* and *Breast Cancer* data sets is rather low, whereas on the *Sonar* data set it is quite substantial.

6.5 Conclusion

We have introduced a new technique that can be considered as a type of an instance-based large margin classifier, called Support Hyperplanes (SHs). SHs induce an implicit and generally nonlinear decision surface between the classes by using a set of (explicitly defined) hyperplanes. SHs classify a test observation using the farthest-away hyperplane from it that is semi-consistent with the data used for training. This results in a good generalization quality. Although we have treated just the binary case, the multi-class extension can easily be carried out by means of standard methods such as one-against-one or one-against-all classification. A potential weak point of SHs, also applying to SVMs, is that it is not clear a priori which type of kernel and what value of the tuning parameters should be used. Furthermore, we do not address the issue of attribute selection and the estimation of class-membership probabilities. Further research could also concentrate on the application of SHs in more domains, on faster implementation suitable for analyzing large-scale data sets, and on the derivation of theoretical test-error bounds.

A Some Properties and Interpretation of Support Hyperplanes in the Separable Case

A.1 Background

Support Hyperplanes (SH) can be viewed as a technique whereby the classification of a test point is done via computing distances to sets. This is particularly evident in the so-called linearly separable case. In this case the SH method possesses a number properties that portray it as an intuitively appealing method that outputs a classification label of a test point based on its distances to (two) well-defined sets. The separable case is also important as a platform for laying down arguments in favor of SH over SVM, as viewed from the goodness-of-fit versus complexity aspect. This aspect is also referred to in the literature as loss versus penalty or bias versus variance.

Let us introduce some notation. A two-class data set D, $\{\mathbf{x}_i, y_i\}_{i=1}^{l}$, from $\mathbb{R}^n \times \{-1, 1\}$ is referred to as being separable if there exists at least one hyperplane that is consistent with it. In other words, there exist coefficients $\mathbf{w}$ and b, such that for all $i = 1, 2, \ldots, l$, $y_i(\mathbf{w}'\mathbf{x}_i + b) > 0$. It is convenient to define three sets of points for a separable data set, as depicted in Figure 6.4. First, let $U_{(+)}$ be the set of points which are classified as "+" (or, 1) by all consistent hyperplanes. Next, let $U_{(-)}$ be the set of points which are classified as "–" (or, -1) by all consistent hyperplanes, and let the Version Space Volume (VSV) denote the rest of the points; refer to Smirnov, Sprinkhuizen-Kuyper, and Nalbantov (2004) and Smirnov et al. (2006), who use these sets in the context of the classification technique Version Space Support Vector Machines (VSSVM). For ease of exposition, we write that a hyperplane with coefficients $\mathbf{w}$ and b "crosses" or "intersects" a given set of points if there exist at least two points $\mathbf{x}_i$ and $\mathbf{x}_j$ from this set such that $\mathbf{w}'\mathbf{x}_i + b > 0$ and $\mathbf{w}'\mathbf{x}_j + b < 0$. Also, a hyperplane "touches" a set of points if there exists at least one point p from this set for which $\mathbf{w}'\mathbf{x}_p + b = 0$ and there do not exist two points $\mathbf{x}_i$ and $\mathbf{x}_j$ from this set such that $\mathbf{w}'\mathbf{x}_i + b > 0$ and $\mathbf{w}'\mathbf{x}_j + b < 0$. The smallest distance between objects A and B is denoted by distance(A,B). The SVM separating hyperplane is denoted by h_{SVM}, while the hyperplanes that form the SVM margin are denoted by h_{m1} and h_{m2}. Finally, $CH_{(+)}$ and $CH_{(-)}$ stand for the convex hulls of the "+" and "–" points, respectively.

A.2 Properties

Lemma 1. *Let D be a separable data set. Then, $U_{(+)}$ and $U_{(-)}$ are nonempty and convex.*

Proof. Suppose $U_{(-)}$ is not convex. Then, there exist $t \in (0, 1)$ and points $\mathbf{x}_B \in U_{(-)}$ and $\mathbf{x}_C \in U_{(-)}$ such that $\mathbf{x}_A = t\mathbf{x}_B + (1 - t)\mathbf{x}_C \notin U_{(-)}$. Since

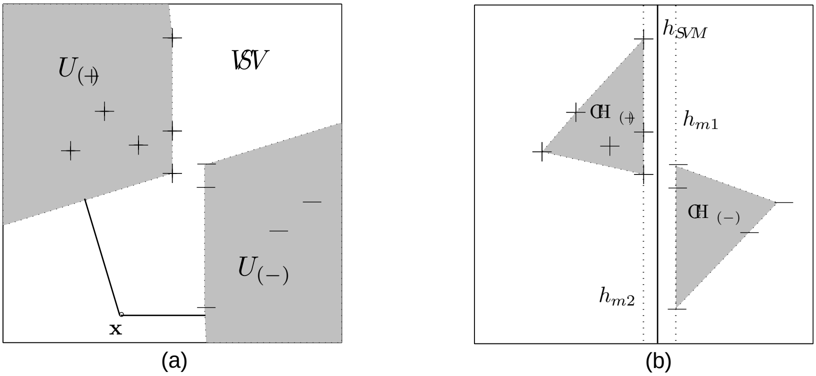


Figure 6.4: Panel (a): sets $U_{(+)}$, $U_{(-)}$, and *VSV*, as well as a test point $\mathbf{x}$ with its distances to $U_{(+)}$ and $U_{(-)}$. Panel (b): The convex hulls of "+" and "–" points, the SVM hyperplane (h_{SVM}), and the two hyperplanes that form the SVM margin, h_{m1} and h_{m2}.

$\mathbf{x}_A \notin U_{(-)}$, $\mathbf{x}_B \in U_{(-)}$, and $\mathbf{x}_C \in U_{(-)}$, there exists a semi-consistent hyperplane $\mathbf{w}'\mathbf{x} + b = 0$ such that $\mathbf{w}'\mathbf{x}_B + b \leq 0$, $\mathbf{w}'\mathbf{x}_C + b \leq 0$ and $\mathbf{w}'\mathbf{x}_A + b > 0$. This, however, is impossible since $\mathbf{w}'\mathbf{x}_A + b = \mathbf{w}'(t\mathbf{x}_B + (1-t)\mathbf{x}_C) + tb + (1-t)b$ $= t(\mathbf{w}'\mathbf{x}_B + b) + (1-t)(\mathbf{w}'\mathbf{x}_C + b) \leq 0$. Therefore, $U_{(-)}$ is convex. (Note that $\mathbf{x}_A \notin U_{(+)}$). □

Lemma 2. *Let D be a separable data set. Then, the support hyperplane touches either $U_{(+)}$, or $U_{(-)}$, or both.*

Proof. Suppose that the support hyperplane (sh) does not touch either $U_{(+)}$ or $U_{(-)}$ (that is, sh is strictly consistent with respect to $U_{(+)}$ and $U_{(-)}$). Then, there exists a set of hyperplanes parallel to sh, within $\epsilon > 0$ distance from sh, which are also consistent with $U_{(+)}$ and $U_{(-)}$. More concretely, there exists a semi-consistent hyperplane sh', which is a better candidate for a farthest-away semi-consistent hyperplane than the original sh, which contradicts the definition of sh. Therefore, sh touches either $U_{(+)}$, or $U_{(-)}$, or both. Note that sh cannot strictly intersect $U_{(+)}$ or $U_{(-)}$ by definition. □

Lemma 3. *Let D be a separable data set and let (1) hyperplane h_1 touch $U_{(+)}$ at point $\mathbf{x}_A$ (that is, $\mathbf{w}'\mathbf{x}_A = 0$); (2) h_1 cross $U_{(-)}$; and (3) point $\mathbf{x}_B \in U_{(-)}$ be a point for which $\mathbf{w}'\mathbf{x}_B + b = 0$. Then, h_1 crosses $U_{(+)}$ (that is, there exists at least one point $\in h_1$ that is strictly inside $U_{(+)}$).*

Proof. Consider point $\mathbf{x}_C$, such that point $\mathbf{x}_A$ is between point $\mathbf{x}_C$ and point $\mathbf{x}_B$. In other words, $\mathbf{x}_C = (1/t)\mathbf{x}_A - ((1-t)/t)\mathbf{x}_B$, $t \in (0,1)$. Since $\mathbf{x}_C$ is a linear combination of $\mathbf{x}_A$ and $\mathbf{x}_B$, and both $\mathbf{x}_A$ and $\mathbf{x}_B \in h_1$, it follows that $\mathbf{x}_C \in h_1$. Now, for each semi-consistent hyperplane "i" we have:

$$\left| \begin{array}{l} \mathbf{w}_i\mathbf{x}_A + b_i \geq 0 \\ \mathbf{w}_i\mathbf{x}_B + b_i < 0 \end{array} \right. \Rightarrow \mathbf{w}_i((1/t)\mathbf{x}_A - ((1-t)/t)\mathbf{x}_B) + b_i > 0 \Rightarrow \mathbf{w}_i\mathbf{x}_C + b_i > 0.$$

Therefore, $\mathbf{x}_C$ is strictly inside $U_{(+)}$. Conclusion: h_1 crosses $U_{(+)}$. □

Theorem 1. *Let D be a separable data set and $\mathbf{x}_A$ be a point in the version space volume (*VSV*). Then, distance(*$\mathbf{x}_A$*,sh) is equal to the distance between point $\mathbf{x}_A$ and the farthest of the sets $U_{(+)}$ and $U_{(-)}$ from point $\mathbf{x}_A$.*

Proof. Suppose distance(sh,$\mathbf{x}_A$) < distance($U_{(+)}$,$\mathbf{x}_A$). The case with $U_{(-)}$ is analogous. Since $U_{(+)}$ is convex, the line segment representing the distance($U_{(+)}$,$\mathbf{x}_A$) is perpendicular to a hyperplane (h_1) that touches $U_{(+)}$ at a point $\mathbf{x}_B$ (but does not cross $U_{(+)}$). By Lemma 3 it follows that h_1 is semi-consistent with $U_{(-)}$ (because if it crosses $U_{(-)}$, it will be inconsistent (=cross) with $U_{(+)}$). Therefore, h_1 is a better candidate for a farthest-away semi-consistent hyperplane than the current (*true*) support hyperplane. This is a contradiction. Thus, distance(sh,$\mathbf{x}_A$) $\geq$ distance($U_{(+)}$,$\mathbf{x}_A$). By analogy, distance(sh,$\mathbf{x}_A$) $\geq$ distance($U_{(-)}$,$\mathbf{x}_A$).

Assume, without loss of generality, that distance($U_{(+)}$,$\mathbf{x}_A$) > distance($U_{(-)}$,$\mathbf{x}_A$) > 0. Suppose distance(sh,$\mathbf{x}_A$) > distance($U_{(+)}$,$\mathbf{x}_A$). Now, let us show that sh cannot touch only $U_{(-)}$ (that is, it must also touch $U_{(+)}$). If sh touches $U_{(-)}$ only, then the solution of the support hyperplanes optimization problem is the distance from point $\mathbf{x}_A$ to the convex hull of the negative observations, and point $\mathbf{x}_A$ is labelled as "positive" in the solution. Thus, $\mathbf{x}_A$ and $U_{(-)}$ are on different sides of h_1. Therefore, distance($\mathbf{x}_A$,sh) $\leq$ distance($\mathbf{x}_A$,$U_{(-)}$), which is a contradiction (distance(sh,$\mathbf{x}_A$) > distance($U_{(+)}$,$\mathbf{x}_A$) $\geq$ distance(sh,$\mathbf{x}_A$)). So, h_1 touches $U_{(+)}$. Therefore, distance(sh,$\mathbf{x}_A$) $\leq$ distance($U_{(+)}$,$\mathbf{x}_A$).

Overall conclusion: distance($U_{(+)}$,$\mathbf{x}_A$) = distance(sh,$\mathbf{x}_A$). □

Theorem 1 allows us to view the SHs as a conceptual extension of the famous Nearest-Neighbor technique, the difference being that SHs compute distances to sets rather than to individual points.

Lemma 4. *Let D be a separable data set and let point $\mathbf{x}_A \in$ convex hull of the negative observations. Then $\mathbf{x}_A \in U_{(-)}$.*

Proof. For each negative point in the data set we have that all semi-consistent hyperplanes do not misclassify them. Therefore, all (convex combinations of) points from the convex hull of the negative observations will likewise be not misclassified. Thus, the convex hull of the negative observations is a subset of $U_{(-)}$. □

Lemma 5. *Let D be a separable data set. Then, all points from $U_{(+)}$ and $U_{(-)}$ lie outside the SVM margin.*

Proof. Assume that point $\mathbf{x}_A \in U_{(-)}$, but it is also inside the margin. All hyperplanes $\in$ SVM margin are semi-consistent by definition, including h_1 and h_2 (the hyperplanes that form the margin). So, one of the two marginal hyperplanes classifies $\mathbf{x}_A$ as "positive". Thus, not all semi-consistent hyperplanes classify $\mathbf{x}_A$ as "negative", which is a contradiction with the assumption that point $\mathbf{x}_A \in U_{(-)}$. Conclusion: $\mathbf{x}_A$ lies outside the margin. □

Lemma 6. *Let D be a separable data set. Then, the SVM hyperplane separates $U_{(+)}$ and $U_{(-)}$ with the widest possible margin.*

Proof. The two SVM hyperplanes that form the margin touch the convex hulls of the negative and positive examples (as shown, e.g., in Bennett & Bredensteiner, 2000). Moreover, the convex hulls of the positive and negative observations are subsets of $U_{(+)}$ and $U_{(-)}$, respectively. Therefore, the SVM hyperplane cannot separate $U_{(+)}$ and $U_{(-)}$ with a bigger margin than that between the convex hulls of the negative and positive observations. These two margins can be equal, however, as can be seen from Lemma 5. Since the SVM hyperplane is unique, it follows that it separates $U_{(+)}$ and $U_{(-)}$ with the widest possible margin. □

Lemma 7. *Let D be a separable data set that consists of $l+1$ points. The Support Hyperplanes optimization problem can be represented as an SVM optimization problem, where a special (Mercer) kernel is being applied.*

Proof. The SVM optimization problem can be stated as:

$$\begin{array}{ll} \max\limits_{\boldsymbol{\alpha}} & \sum_{i=1}^{l+1} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{l+1} \alpha_i \alpha_j y_i y_j (\mathbf{x}_i' \mathbf{x}_j) \\ \text{s.t.} & 0 \leq \alpha_i,\ i = 1, 2, \ldots, l+1, \text{ and } \sum_{i=1}^{l+1} y_i \alpha_i = 0, \end{array} \tag{6.5}$$

The SH optimization problem can be stated as:

$$\begin{array}{ll} \max\limits_{\boldsymbol{\alpha}} & \alpha_{l+1} - \frac{1}{2} \sum_{i,j=1}^{l+1} \alpha_i \alpha_j y_i y_j (\mathbf{x}_i' \mathbf{x}_j) \\ \text{s.t.} & 0 \leq \alpha_i,\ i = 1, 2, \ldots, l+1, \text{ and } \sum_{i=1}^{l+1} y_i \alpha_i = 0. \end{array} \tag{6.6}$$

In order to convert the SVM formulation into an SH one, the main idea is to add a positive number to each diagonal entry of the kernel matrix XX', which will absorb the sum $\sum_{i=1}^{l} \alpha_i$ in the SVM optimization formulation. So, $\sum_{i=1}^{l} \alpha_i$ will be incorporated in XX'. This can be done only if all alpha's are known, in the following way: $2/\alpha_i$ has to be added to the i^{th} diagonal element of XX' in the SVM formulation in case $\alpha_i \neq 0$ and $i \neq l+1$.

The SVM high-dimensional vector dot product thus becomes:

$$(\mathbf{x}_{i1}, \mathbf{x}_{i2}, \ldots, \mathbf{x}_{in}, \sqrt{2}\tfrac{\delta_{ij}}{\sqrt{\alpha_i}})'(\mathbf{x}_{j1}, \mathbf{x}_{j2}, \ldots, \mathbf{x}_{jn}, \sqrt{2}\tfrac{\delta_{ij}}{\sqrt{\alpha_j}}),$$

where $\delta_{ij} = 1$ if $i = j$, and $\delta_{ij} = 0$ if $i \neq j$. □

Chapter 7

Nearest Convex Hull Classification

Consider the classification task of assigning a test object to one of two or more possible groups, or classes. An intuitive way to proceed is to assign the object to that class, to which the distance is minimal. As a distance measure to a class, we propose here to use the distance to the convex hull of that class. Hence the name Nearest Convex Hull (NCH) classification for the method. Convex-hull overlap is handled through the introduction of slack variables and kernels. In spirit and computationally the method is therefore close to the popular Support Vector Machine (SVM) classifier. Advantages of the NCH classifier are its robustness to outliers, good regularization properties and relatively easy handling of multi-class problems. We compare the performance of NCH against state-of-art techniques and report promising results.

7.1 Introduction

There are many approaches to the classification task of separating two or more groups of objects on the basis of some shared characteristics. Existing techniques range from Linear Discriminant Analysis (LDA), Quadratic Discriminant Analysis (QDA) and Binary Logistic Regression to Decision Trees, Neural Networks, Support Vector Machines (SVM), etc. Many of those classifiers make use of some kind of a distance metric (in some n-dimensional space) to derive classification rules. Here, we propose to use another such classifier, called Nearest Convex Hull (NCH) classifier.

As the name suggests, the so-called hard-margin version of the NCH classifier assigns a test object $\mathbf{x}$ to that group of training objects, which convex hull is closest to $\mathbf{x}$. This involves solving an optimization problem to find the distance to each class. Algorithms for doing so have been proposed in the literature under the general heading of finding the minimum distance between convex sets (see, e.g., Vapnik, 1995; Bennett & Bredensteiner, 2000). We confer also to Luxburg and Bousquet (2004) for a more general discussion on distance-based classification. Existing off-the-shelf algorithms however cannot be directly applied for classification tasks where a mixture of a soft-margin and a hard-margin approaches is required. In the separable, hard-margin case, a problem arises if $\mathbf{x}$ lies inside the convex hulls of two or more groups, since its distance to these convex hulls is effectively equal to zero and the classification of $\mathbf{x}$ is undetermined. To deal with this problem, we introduce a soft-margin version of the NCH classifier, where convex-hull overlap between $\mathbf{x}$ and a given class is penalized linearly. The difference with the soft-margin SVM approach lies in the requirement that the soft approach is applied to all data points except the test point $\mathbf{x}$. As an alternative solution to convex-hull overlaps, one could map the training data from the original space into a higher-dimensional space where convex-hull overlap can be avoided. A combination of both approaches is also possible.

The linear (and not, for example, quadratic) penalization of the errors gives rise to the robustness-to-outliers property of NCH. Another advantage of NCH in terms of computational speed arises in the context of multi-class classification tasks. This occurs because only same-class objects are considered in the estimation of a (soft) distance to a convex hull, and not the whole data set. The decision surface of the NCH classifier is not explicitly computed because the classification process for each test point is independent of the classification process for other test points. That is why the classification process is instance-based in nature. In sum, the NCH method can be considered as a type of instance-based large-margin classifier.

The chapter is organized as follows. First we provide some intuition behind the NCH classifier and a formal definition of it. Next, we discuss the technical aspects of the classifier – derivation and implementation. Finally, we show some experimental results on popular data sets and then conclude.

7.2 Nearest Convex Hull Classifier: Definition and Motivation

At the outset, consider a binary data set of positive and negative objects $\langle I^+, I^- \rangle$ from $\mathbb{R}^n$. Formally, the task is to separate the two classes of objects with a decision surface that performs well on a test data set. This task is formalized as finding a (target) function $f : \mathbb{R}^n \to \{-1, 1\}$ such that f will classify correctly unseen observations. The extension to the multi-class case is straightforward. The decision rule of the NCH classifier is the following: *a test point* $\mathbf{x}$ *should be assigned to that class, which convex hull is closest to* $\mathbf{x}$.

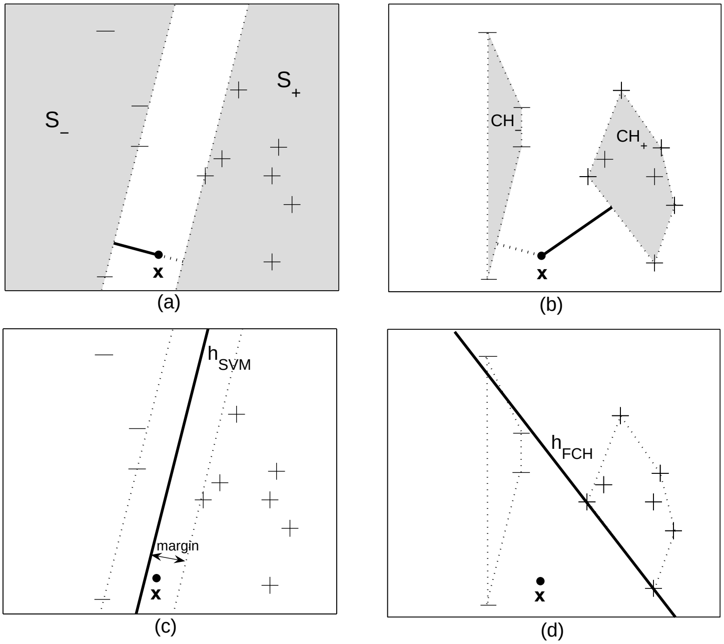


Figure 7.1: Classification of a test point $\mathbf{x}$ with SVM in Panels (a) and (c), and NCH in Panels (b) and (d) on a binary data set. In Panel (a), the white band has the largest possible width, which is equal to twice the margin, shown in Panel (c). The points to the left and to the right of the band form shaded sets S_- and S_+, respectively. Test point $\mathbf{x}$ receives label $+1$ since it is farther from S_- than S_+. In Panel (b) point $\mathbf{x}$ is classified as -1 since it is farther from the convex hull of the positive points, CH_+, than from the convex hull of the negative points, CH_-.

Let us consider the so-called separable case where the classes are separable by a hyperplane and draw an intuitive comparison between NCH and the popular SVM classifier. See Figure 7.1 for an illustrative binary classification example. Panels (a) and (c) refer to SVM classification, and Panels (b) and (d) refer to NCH classification. In SVM classification, the target function is a hyperplane of the form $\mathbf{w}'\mathbf{x} + b = 0$, where $\mathbf{w}$ is a vector of coefficients and b in the intercept. The SVM hyperplane $\mathbf{w}^{*\prime}\mathbf{x}+b^* = 0$ (denoted as h_{SVM}) is the one that separates the classes with the widest margin, where a margin is defined as the distance between a (separating) hyperplane and the closest point to it from the training data set. In terms of Figure 7.1, Panel (a), the width of white band is equal to twice the margin, which is shown in Panel (c). The closest point to h_{SVM} is defined to lie on the hyperplane $\mathbf{w}^{*\prime}\mathbf{x}+b^* = 1$ if this point is positively labeled, or on the hyperplane $\mathbf{w}^{*\prime}\mathbf{x} + b^* = -1$ if this closest point is negatively labeled. For all points $\mathbf{x}$ that lie outside the margin it holds that either $\mathbf{w}^{*\prime}\mathbf{x} + b^* < -1$ or $\mathbf{w}^{*\prime}\mathbf{x} + b^* > 1$. The former set of points is defined as S_-, and the latter set of points is defined as S_+. For any test point $\mathbf{x}$, the SVM classification rule can be formulated as follows: a test point $\mathbf{x}$ should be classified as -1 if it is farther away from set S_+ than from set S_-; otherwise $\mathbf{x}$ receives label $+1$.

It has been argued (see, e.g., Burges, 1998; Vapnik, 1995) that SVM classification searches for a balance between empirical error (or, the goodness-of-fit over the training data) and complexity, where complexity is proxied by the distance between sets S_+ and S_- (that is, twice the margin). In the separable case at hand, the empirical error of h_{SVM} is zero since it fits the data perfectly. Also, complexity and margin width are inversely related: the larger the margin, the lower the associated complexity. The balance between empirical error and complexity can intuitively be approached from an instance-based viewpoint as well. In this case, complexity is imputed in the classification of each separate test object/instance. Thus, the larger the distance from a test object $\mathbf{x}$ to the farther one of the two sets S_+ and S_-, the lower the complexity associated with the classification of $\mathbf{x}$.

The NCH classifier can also be considered from a fit-versus-complexity standpoint. Let us denote by CH_+ and CH_- the set of points that form the convex hulls of the positive and negative objects, respectively (see Figure 7.1, Panel (b)). Somewhat similarly to SVM, in NCH classification one considers the distance to the farther one of the two convex hulls CH_+ and CH_- as a proxy for the complexity associated with the classification of $\mathbf{x}$. Quite interestingly, this distance is always as big as or bigger than the distance from $\mathbf{x}$ to the farther of sets S_+ and S_-. This property holds since the convex hull of the $+1$ (-1) points is a subset of S_+ (S_-), as can be seen in Figure 7.1. Therefore, if one considers the distance to the farther-away convex hull as a proxy for complexity associated with the classification of $\mathbf{x}$, then NCH classification is characterized by a lower complexity than SVM classification. However, the fit over the training data of NCH may turn out to be inferior to SVM in some cases. Let h_{FCH} denote the hyperplane that is tangent to the farther-away convex hull of same-class training data points, and is perpendicular to the line segment that represents the distance between $\mathbf{x}$ and this convex hull, as in Figure 7.1, Panel (d). Thus, the distance between $\mathbf{x}$ and h_{FCH} equals the distance between $\mathbf{x}$ and the farther convex hull. Effectively, in NCH classification $\mathbf{x}$ is classified using h_{FCH}. Notice that by definition h_{FCH} separates without an error either the positive or the negative observations, depending on which convex hull is farther from $\mathbf{x}$. Thus, h_{FCH} is not guaranteed to have a perfect fit over the whole data set that consists of both positive and negative points, as illustrated in Figure 7.1, Panel (d). As a consequence, it is not clear a priory whether NCH or SVM will strike a better balance between fit and complexity in the classification of a given point $\mathbf{x}$: there is a gain for NCH coming from decreased complexity (in the form of an increased distance) vis-a-vis SVM on the one hand, accompanied by a potential loss arising from a possible increased empirical error of h_{FCH} over the whole training data set, on the other.

NCH has the property that the extent of proximity to a given class is determined without taking into consideration objects from other classes. This property contrasts with the SVM approach, where the sets S_+ and S_- are not created independently of each other. A similar parallel can be drawn between LDA and QDA methods. In LDA, one first determines the Mahalanobis distances from $\mathbf{x}$ to the centers of the classes using a common pooled covariance matrix and then classifies $\mathbf{x}$ accordingly. In QDA, one uses a separate covariance matrix for each class. Analogically, the NCH classifier first determines the Euclidean distance from $\mathbf{x}$ to the convex hulls of each of the classes and then classifies $\mathbf{x}$ accordingly. In sum, loosely speaking one may think of the shift from SVM to NCH as resembling the shift from LDA to QDA.

7.3 Estimation

7.3.1 Separable Case

Consider a data set of l objects from k different groups, or classes. Let l_k denote the number of objects in the k^{th} class. According to NCH, a test point $\mathbf{x}$ is assigned to that class, to which the distance is minimal. In the separable case, the distance to a class is defined as the distance to the convex hull of the objects from that class. The algorithm for classifying $\mathbf{x}$ can be described as follows (see Figure (7.2)): first, compute the distance from $\mathbf{x}$ to the convex hull of each of the k classes; second, assign to $\mathbf{x}$ the label of the closest class. Formally, to find the distance from a test point $\mathbf{x}$ to the convex hull of the nearest class, the following quadratic optimization problem has to be solved for each class k:

$$\begin{aligned} \min_{\mathbf{w}_k, b_k} \quad & \frac{1}{2}\mathbf{w}_k'\mathbf{w}_k \\ \text{such that} \quad & \mathbf{w}_k'\mathbf{x}_i + b_k \geq 0, \; i = 1, 2, \ldots, l_k \\ & -(\mathbf{w}_k'\mathbf{x} + b_k) = 1 \end{aligned} \tag{7.1}$$

The distance between hyperplane $\mathbf{w}_k'\mathbf{x}+b_k = 0$ and $\mathbf{x}$ is defined as $1/\sqrt{\mathbf{w}_k'\mathbf{w}_k}$ by the last constraint of (7.1). This distance is maximal when $\frac{1}{2}\mathbf{w}_k'\mathbf{w}_k$ is minimal. At the optimum, it represents the distance from $\mathbf{x}$ to the convex hull of class k. The role of the first l_k inequality constraints is to ensure that the hyperplane classifies correctly each point that belongs to class k. Effectively, for each of the k classes, the l_k same-class objects are assigned label 1, and the test point is assigned label -1. Eventually, $\mathbf{x}$ is assigned to that class to which the distance is minimal, that is, which corresponding value for the objective function in (7.1) is maximal.

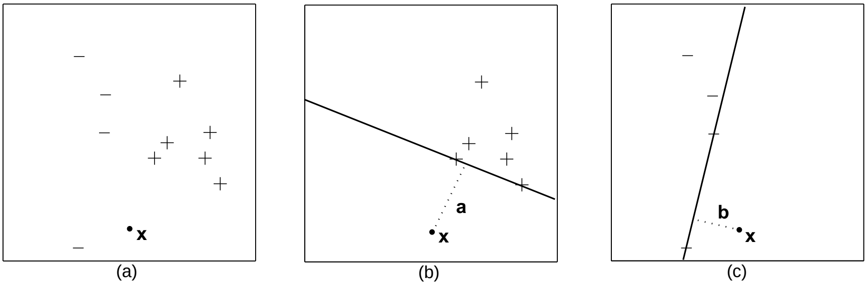


Figure 7.2: Classification of a test point $\mathbf{x}$ with NCH on the binary data set in Panel (a) in two steps. At stage one (Panel (b)), a test point $\mathbf{x}$ is added to a data set that contains only the positive class, and the distance $\mathbf{a}$ from $\mathbf{x}$ to the convex hull of this class is computed. At stage two (Panel (c)), $\mathbf{x}$ is added to a data set that contains only the negative class, and the distance $\mathbf{b}$ from $\mathbf{x}$ to the convex hull of this class is computed. If $\mathbf{a} > \mathbf{b}$ ($\mathbf{a} < \mathbf{b}$), then $\mathbf{x}$ is assigned to the negative (positive) class.

7.3.2 Nonseparable Case

Optimization problem (7.1) can be solved for each k only if the test point lies outside the convex hull of each class k. A further complication arises if some of the convex hulls overlap. Then a test point could lie simultaneously in two or more convex hulls and its classification label would be undetermined. To cope with these situations, so-called slack variables can be introduced, similarly to the SVM approach. Consequently, the nonseparable version of optimization problem (7.1) that has to be solved for each class k becomes:

$$\begin{aligned} \min_{\mathbf{w}_k, b_k, \boldsymbol{\xi}} \quad & \frac{1}{2}\mathbf{w}_k'\mathbf{w}_k + C\sum_{i=1}^{l_k} \xi_i \qquad (7.2) \\ \text{such that.} \quad & \mathbf{w}_k'\mathbf{x}_i + b_k \geq 0 - \xi_i,\ \ \xi_i \geq 0,\ \ i = 1, 2, \ldots, l_k \\ & -(\mathbf{w}_k'\mathbf{x} + b_k) = 1. \end{aligned}$$

Note that in (7.2) the points that are incorrectly classified are penalized linearly via the term $\sum_{i=1}^{l_k} \xi_i$. If one prefers a quadratic penalization of the classification errors, then the sum of squared errors $\sum_{i=1}^{l_k} \xi_i^2$ should be substituted for $\sum_{i=1}^{l_k} \xi_i$ in (7.2). One can go even further and extend the NCH algorithm in a way analogical to LS-SVM (Gestel et al., 2004) by imposing in (7.2) that constraints $\mathbf{w}_k'\mathbf{x}_i + b_k \geq 0 - \xi_i$ hold as equalities, on top of substituting $\sum_{i=1}^{l_k} \xi_i^2$ for $\sum_{i=1}^{l_k} \xi_i$.

Each of the k (primal) optimization problems pertaining to (7.2) can be expressed in dual form[1] as:

$$\begin{array}{ll} \max\limits_{\boldsymbol{\alpha}} & \alpha_{l_k+1} - \frac{1}{2}\sum_{i,j=1}^{l_k+1} \alpha_i\alpha_j y_i y_j (\mathbf{x}_i'\mathbf{x}_j) \\ \text{such that} & 0 \leq \alpha_i \leq C,\ i = 1, 2, \ldots, l_k,\ \text{and}\ \sum_{i=1}^{l_k+1} y_i\alpha_i = 0, \end{array} \tag{7.3}$$

where the α's are the Lagrange multipliers associated with the respective k^{th} primal problem. Here α_{l_k+1} is the Lagrange multiplier associated with the equality constraint $-(\mathbf{w}_k'\mathbf{x}+b_k) = 1$. In each problem $y_i = 1, i = 1, 2, \ldots, l_k$ and $y_{l_k+1} = -1$. The advantage of the dual formulation (7.3) is that different *Mercer* kernels can be employed to replace the inner product $\mathbf{x}_i'\mathbf{x}_j$ in (7.3) in order to obtain nonlinear decision boundaries, just like in the SVM case. Three popular kernels are linear $\kappa(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i'\mathbf{x}_j$, polynomial of degree d $\kappa(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i'\mathbf{x}_j+1)^d$ and the Radial Basis Function (RBF) kernel $\kappa(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma \parallel \mathbf{x}_i - \mathbf{x}_j \parallel^2)$, where the manually-adjustable γ parameter determines the proximity between $\mathbf{x}_i$ and $\mathbf{x}_j$.

A total of k NCH optimization problems have to be solved to determine the class of any test point $\mathbf{x}$. This property provides for the fact that the NCH decision boundary is in general implicit and nonlinear, even in case the original data is not mapped into a higher-dimensional space via a kernel. Figure 7.3 demonstrates that this property does not hold in general for Support Vector Machines, for instance. This figure also illustrates that the NCH decision boundary appears to be less sensitive to the choice of kernel and kernel parameters than the respective SVM boundary.

Technically speaking, in case the convex hulls do not overlap, NCH could be solved using the standard SVM optimization formulation (see, e.g., Vapnik, 1995; Burges, 1998). In this case one searches for the widest margin between each of the k classes and a test point $\mathbf{x}$. This margin represents the distance from $\mathbf{x}$ to the convex hull of the k^{th} class. The class for which the margin is smallest is the winning one. The standard nonseparable-case SVM formulation cannot however be automatically applied to the nonseparable NCH case, since the equality constraint in (7.2) will not be satisfied in general.

7.4 Experiments on Some UCI and SlatLog Data Sets

The basic optimization algorithm for Nearest Convex Hull classification (7.3) is implemented via a modification of the freely available LIBSVM software (Chang & Lin, 2006). We tested the performance of NCH on several small- to middle-sized data sets that are freely available from the SlatLog and UCI repositories

[1] The derivation of the dual problem resembles the one used in SVM (see, e.g., Burges, 1998).

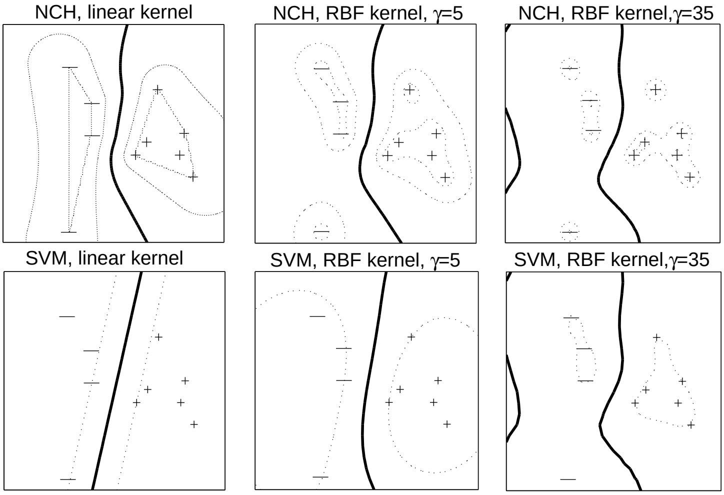


Figure 7.3: Decision boundaries for NCH and SVM using the linear and RBF kernels on a linearly separable data set. The dashed contours for the NCH method are iso-curves along which the ratio of the distances to the two convex hulls is constant.

(Newman et al., 1998) and have been analyzed by many researchers and practitioners (e.g., among others, Breiman, 1996; King et al., 1995; Lim et al., 1995; Perlich et al., 2003): *Sonar*, *Voting*, *Wisconsin Breast Cancer* (W.B.C.), *Heart*, *Australian Credit Approval* (A.C.A.), and *Hepatitis* (Hep.). Detailed information on these data sets can be found on the web sites of the respective repositories.

We compare the results of NCH to those of several state-of-art techniques: Support Vector Machines (SVM), Linear and Quadratic Discriminant Analysis (LDA and QDA), Logistic Regression (LR), Multi-layer Perceptron (MLP), k-Nearest Neighbor (kNN), Naive Bayes classifier (NB) and two types of Decision Trees – Decision Stump (DS) and C4.5. The experiments for the NB, LR, MLP, kNN, DS and C4.5 methods have been carried out with the WEKA learning environment using default model parameters, except for kNN. We refer to Witten and Frank (2005) for additional information on these classifiers and their implementation. We measure model performance by the leave-one-out (LOO) accuracy rate. Because we aim at comparing several methods, LOO seems to be more suitable than the more general k-fold cross-validation (CV), because it always yields one and the same error rate estimate for a given model, unlike the CV method (which involves a random split of the data into several parts).

Table 1 presents performance results for all methods considered. Some methods, namely kNN, NCH and SVM, require tuning of model parameters. In these

Table 7.1: Leave-one-out accuracy rates (in %) of the Nearest Convex Hull classifier as well as some standard methods on several data sets. Rbf, 2p and lin stand for Radial Basis Function, second-degree polynomial and linear kernel, respectively

	Sonar	Voting	W.B.C.	Heart	A.C.A.	Hep.
NCH rbf	**91.35**	95.85	**97.42**	**85.56**	86.38	85.16
NCH 2p	90.38	85.48	97.14	82.59	85.36	84.52
NCH lin	87.98	95.85	97.28	84.07	86.09	84.52
SVM rbf	88.94	96.54	97.00	**85.56**	**87.39**	86.45
SVM 2p	82.21	96.31	96.85	81.11	79.86	86.45
SVM lin	80.77	96.77	96.85	**85.56**	87.10	86.45
NB	67.30	90.32	95.99	82.96	77.10	83.23
LR	73.08	96.54	96.14	83.70	86.38	83.87
LDA	75.48	95.85	95.99	83.70	85.80	85.81
QDA	74.88	94.24	91.42	81.48	85.22	83.87
MLP	81.25	94.93	94.99	78.89	84.78	79.35
*k*NN	86.54	93.32	97.00	84.44	85.94	85.81
DS	73.08	95.85	92.42	76.30	85.51	79.35
C4.5	71.15	**97.00**	95.28	75.19	83.77	80.00

cases, we report only the highest LOO accuracy rate obtained by performing a grid search for tuning the necessary parameters. Overall, the NCH classifier performs quite well on all data sets, and achieves best accuracy rates on three data sets. SVM also perform best on three data sets. The rest of the techniques show relatively less favorable and more volatile results. For example, the C4.5 classifier performs best on the *Voting* data set, but achieves rather low accuracy rates on two other data sets – *Sonar* and *Heart*. Note that not all data sets are equally easy to handle. For instance, the performance variation over all classifiers on the *Voting* and *Breast Cancer* data sets is rather low, whereas on the *Sonar* data set it is quite substantial.

7.5 Conclusion

We have introduced a new technique that can be considered as a type of an instance-based large-margin classifier, called Nearest Convex Hull classifier (NCH). NCH assigns a test observation to the class, which convex hull is closest. Convex-hull overlap is handled via the introduction of slack variables and/or kernels. NCH induces an implicit and generally nonlinear decision surface between the classes. One of the advantages of NCH is that an extension from binary to multi-class classification tasks can be carried out in a straightforward way. Others are its alleged robustness to outliers and good generalization qualities. A potential weak point of NCH, which also holds for SVM, is that it is not clear a priori which type of kernel and what value of the tuning parameters should be used. Furthermore, we do not address the issue of attribute selection and the estimation of class-membership probabilities. Further research could also concentrate on the application of NCH in more domains, on faster implementation suitable for analyzing large-scale data sets, and on the derivation of theoretical test-error bounds.

Chapter 8

SVM-Maj: A Majorization Approach to Linear Support Vector Machines with Different Hinge Errors*

Support vector machines (SVM) are becoming increasingly popular for the prediction of a binary dependent variable. SVMs perform very well with respect to competing techniques. Often, the solution of an SVM is obtained by switching to the dual. In this paper, we stick to the primal support vector machine (SVM) problem, study its effective aspects, and propose varieties of convex loss functions such as the standard for SVM with the absolute hinge error as well as the quadratic hinge and the Huber hinge errors. We present an iterative majorization algorithm that minimizes each of the adaptations. In addition, we show that many of the features of an SVM are also obtained by an optimal scaling approach to regression. We illustrate this with an example from the literature and do a comparison of different methods on several empirical data sets.

8.1 Introduction

An increasingly more popular technique for the prediction of two groups from a set of predictor variables is the support vector machines (SVM, see, e.g.,

*This chapter has been published as Groenen, Nalbantov, and Bioch (2008): Groenen, P.J.F., Nalbantov, G. & Bioch, J.C. (2008). SVM-Maj: a majorization approach to linear support vector machines with different hinge errors. In *Advances in Data Analysis and Classification* (Vol. 2, pp. 149–162). Berlin/Heidelberg.

Vapnik, 2000). Although alternative techniques such as linear and quadratic discriminant analysis, neural networks, and logistic regression can also be used to analyze this data analysis problem, the prediction quality of SVMs seems to compare favorably with respect to these competing models. Another advantage of SVMs is that they are formulated as a well-defined optimization problem that can be solved through a quadratic program. A second valuable property of the SVM is that the derived classification rule is relatively simple and can be readily applied to new, unseen samples. A potential disadvantage is that the interpretation in terms of the predictor variables in nonlinear SVM is not always possible. In addition, the usual dual formulation of an SVM may not be so easy to grasp.

In this paper, we restrict our focus to linear SVMs. We believe that this paper makes several contributions on three themes. First, we offer a nonstandard way of looking at linear SVMs that makes the interpretation easier. To do so, we stick to the primal problem and formulate the SVM in terms of a loss function that is regularized by a penalty term. From this formulation, it can be seen that SVMs use robustified errors. Apart from the standard SVM loss function that uses the absolute hinge error, we advocate two other hinge errors, the Huber and quadratic hinge errors, and show the relation with ridge regression. Note that recently, Rosset and Zhu (2007) also discusses the use of different errors in SVMs including these two hinge errors.

The second theme of this paper is to show the connection between optimal scaling regression and SVMs. The idea of optimally transforming a variable so that a criterion is being optimized has been around for more than 30 years (see, for example, Young, 1981; Gifi, 1990). We show that optimal scaling regression using an ordinal transformation with the primary approach to ties comes close to the objective of SVMs. We discuss the similarities between both approaches and give a formulation of SVM in terms of optimal scaling.

A third theme is to develop and extend the majorization algorithm of Groenen et al. (2007) to minimize the loss for any of the hinge errors. We call this general algorithm SVM-Maj. The advantage of majorization is that each iteration is guaranteed to reduce the SVM loss function until convergence is reached. As the SVM loss functions with convex hinge errors such as the quadratic and Huber hinge errors are convex, the majorization algorithm stops at a minimum after a sufficient number of iterations. For the case of the Huber and quadratic hinge, the SVM-Maj algorithm turns out to yield computationally very efficient updates amounting to a single matrix multiplication per iteration. Through SVM-Maj, we contribute to the discussion on how to approach the linear SVM quadratic problem.

Finally, we provide numerical experiments on a suite of 14 empirical data sets to study the predictive performance of the different errors in SVMs and compare it to optimal scaling regression. We also compare the computational efficiency of the majorization approach for the SVM to several standard SVM solvers.

Note that this paper is a significantly extended version of Groenen et al. (2007).

8.2 The SVM Loss Function

Here, we present a rather non-mainstream view on explaining how SVM work. There is a quite close relationship between SVM and regression. We first introduce some notation. Let the matrix of quantitative predictor variables be represented by the $n \times m$ matrix $\mathbf{X}$ of n objects and m variables. The grouping of the objects into two classes is given by the $n \times 1$ vector $\mathbf{y}$, that is, $y_i = 1$ if object i belongs to class 1 and $y_i = -1$ if object i belongs to class -1. The exact labeling -1 and 1 to distinguish the classes is not important. The weights used to make a linear combination of the predictor variables is represented by the $m \times 1$ vector $\mathbf{w}$. Then, the predicted value q_i for object i is

$$q_i = c + \mathbf{x}_i'\mathbf{w}, \tag{8.1}$$

where $\mathbf{x}_i'$ is row i of $\mathbf{X}$ and c is an intercept. As an illustrative example, consider Figure 8.1a with a scatterplot of two predictor variables, where each row i is represented by a point labeled '+' for the class 1 and 'o' for class -1. Every combination of w_1 and w_2 defines a direction in this scatter plot. Then, each point i can be projected onto this line. The main idea of the SVM is to choose this line in such a way that the projections of the points of class 1 are well separated from those of class -1. The line of separation is orthogonal to the line with projections and the intercept c determines where exactly it occurs. The length $\|\mathbf{w}\|$ of $\mathbf{w}$ has the following significance. If $\mathbf{w}$ has length 1, that is, $\|\mathbf{w}\| = (\mathbf{w}'\mathbf{w})^{1/2} = 1$, then Figure 8.1a explains fully the linear combination (8.1). If $\mathbf{w}$ does not have length 1, then the scale values along the projection line should be multiplied by $\|\mathbf{w}\|$. The dotted lines in Figure 8.1a show all those points that project to the lines at $q_i = -1$ and $q_i = 1$. These dotted lines are called the margin lines in SVMs. With three predictor variables, the objects are points in a three dimensional space, $\mathbf{w}$ still defines a direction, but all points that project on $\mathbf{w}$ at the same locations now form a plane and in higher dimensionality form a hyperplane. Thus, with more than two predictor variables, there will be a separation hyperplane and the margins are also hyperplanes. Summarizing, the SVM has three sets of parameters that determine its solution: (1) the weights normalized to have length 1, that is, $\mathbf{w}/\|\mathbf{w}\|$, (2) the length of $\mathbf{w}$, that is, $\|\mathbf{w}\|$, and (3) the intercept c.

An error is counted in SVMs as follows. Every object i from class 1 that projects such that $q_i \geq 1$ yields a zero error. However, if $q_i < 1$, then the error is linear with $1 - q_i$. Similarly, objects in class -1 with $q_i \leq -1$ do not contribute to the error, but those with $q_i > -1$ contribute linearly with $q_i + 1$. Thus,

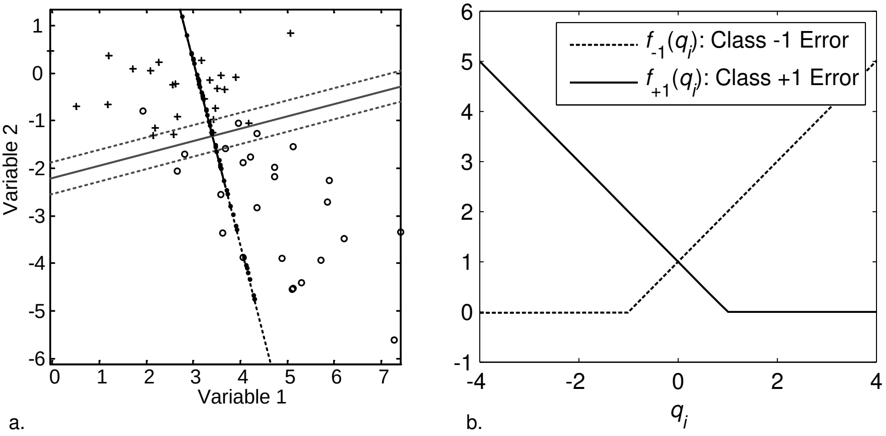


Figure 8.1: Panel a. gives projections of the observations in groups 1 (+) and −1 (o) onto the line given by w_1 and w_2. Panel b. shows the absolute hinge error function $f_1(q_i)$ for class 1 objects (solid line) and $f_{-1}(q_i)$ for class −1 objects (dashed line).

objects that project on the wrong side of their margin contribute to the error, whereas objects that project on the correct side of their margin yield zero error. Figure 8.1b shows the error functions for the two classes. Because of its hinge form, we call this error function the *absolute hinge* error.

As the length of **w** controls how close the margin lines are to each other, it can be beneficial for the number of errors to choose the largest $\|\mathbf{w}\|$ possible, so that fewer points contribute to the error. To control the $\|\mathbf{w}\|$, a penalty term that is dependent on $\|\mathbf{w}\|$ is added to the loss function. The penalty term also avoids overfitting of the data.

Let G_1 and G_{-1} respectively denote the sets of class 1 and −1 objects. Then, the SVM loss function can be written as

$$\begin{array}{llllll} L_{\text{SVM}}(c, \mathbf{w}) & & & & & (8.2) \\ = & \sum_{i \in G_1} \max(0, 1 - q_i) & + & \sum_{i \in G_{-1}} \max(0, q_i + 1) & + & \lambda \mathbf{w}'\mathbf{w} \\ = & \sum_{i \in G_1} f_1(q_i) & + & \sum_{i \in G_{-1}} f_{-1}(q_i) & + & \lambda \mathbf{w}'\mathbf{w} \\ = & \text{Class 1 errors} & + & \text{Class } -1 \text{ errors} & + & \text{Penalty for} \\ & & & & & \text{nonzero } \mathbf{w}, \end{array}$$

where $\lambda > 0$ determines the strength of the penalty term. In this notation, the arguments $(c, \mathbf{w})$ indicate that $L_{\text{SVM}}(c, \mathbf{w})$ needs to be minimized with respect to the arguments c and $\mathbf{w}$. For similar expressions, see Hastie et al. (2001) and

Vapnik (2000). Note that (8.3) can also be expressed as

$$L_{\mathrm{SVM}}(c, \mathbf{w}) = \sum_{i=1}^{n} \max(0, 1 - y_i q_i) + \lambda \mathbf{w}'\mathbf{w},$$

which is closer to the expressions used in the SVM literature.

Once a solution c and $\mathbf{w}$ is found that minimizes (8.3), we can determine how each object contributes to the error. Each object i that projects on the correct side of its margin contributes with zero error to the loss. Therefore, these objects could be removed from the analysis without changing the minimum of (8.3) and the values of c and $\mathbf{w}$ where this minimum is reached. The only objects determining the solution are those projecting on or at the wrong side of their margin thereby inducing error. Such objects are called support vectors as they form the fundament of the SVM solution. Unfortunately, these objects (the support vectors) are not known in advance and, therefore, the analysis needs to be carried out with all n objects present in the analysis. It is the very essence of the SVM definition that error free data points have no influence on the solution.

From (8.3) it can be seen that any error is punished linearly, not quadratically. Therefore, SVMs are more robust against outliers than a least-squares loss function. The idea of introducing robustness by absolute errors is not new. For more information on robust multivariate analysis, we refer to Huber (1981), Vapnik (2000), and Rousseeuw and Leroy (2003). In the next section, we discuss two other error functions, one of which is robust.

In the SVM literature, the SVM loss function is usually presented as follows (Burges, 1998):

$$\begin{aligned} L_{\mathrm{SVMClas}}(c, \mathbf{w}, \xi) \quad &= \quad C\sum_{i \in G_1} \xi_i + C\sum_{i \in G_2} \xi_i + \frac{1}{2}\mathbf{w}'\mathbf{w}, && (8.3)\\ \text{subject to} \quad & \quad 1 + (c + \mathbf{w}'\mathbf{x}_i) \le \xi_i \text{ for } i \in G_{-1} && (8.4)\\ & \quad 1 - (c + \mathbf{w}'\mathbf{x}_i) \le \xi_i \text{ for } i \in G_1 && (8.5)\\ & \quad \xi_i \ge 0, && (8.6) \end{aligned}$$

where C is a nonnegative parameter set by the user to weigh the importance of the errors represented by the so-called slack variables ξ_i. If object i in G_1 projects at the correct side of its margin, that is, $q_i = c + \mathbf{w}'\mathbf{x}_i \ge 1$, then $1 - (c + \mathbf{w}'\mathbf{x}_i) \le 0$ so that the corresponding ξ_i can be chosen as 0. If i projects on the wrong side of its margin, then $q_i = c + \mathbf{w}'\mathbf{x}_i < 1$ so that $1 - (c + \mathbf{w}'\mathbf{x}_i) > 0$. Choosing $\xi_i = 1 - (c + \mathbf{w}'\mathbf{x}_i)$ gives the smallest ξ_i satisfying the restrictions in (8.4), (8.5), and (8.6). Therefore, $\xi_i = \max(0, 1 - q_i)$ and is a measure of error. For class -1 objects, a similar derivation can be made. Note that in the SVM literature (8.3) and (8.6) are often expressed more compactly as

$$\begin{aligned} L_{\mathrm{SVMClas}}(c, \mathbf{w}, \xi) &= C\sum_{i=1}^{n} \xi_i + \frac{1}{2}\mathbf{w}'\mathbf{w}, \\ \text{subject to} \quad & y_i(c + \mathbf{w}'\mathbf{x}_i) \leq 1 - \xi_i \text{ for } i = 1, \ldots, n \\ & \xi_i \geq 0. \end{aligned}$$

If we choose C as $(2\lambda)^{-1}$ then

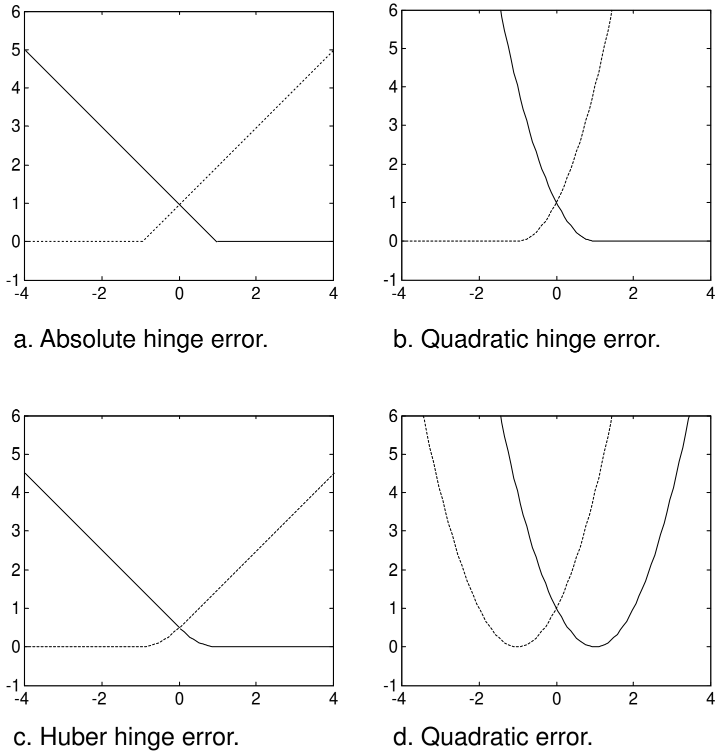
$$\begin{aligned} & L_{\mathrm{SVMClas}}(c, \mathbf{w}, \xi) \\ &= (2\lambda)^{-1}\left(\sum_{i \in G_1} \xi_i + \sum_{i \in G_{-1}} \xi_i + 2\lambda\frac{1}{2}\mathbf{w}'\mathbf{w}\right) \\ &= (2\lambda)^{-1}\left(\sum_{i \in G_1} \max(0, 1 - q_i) + \sum_{i \in G_{-1}} \max(0, q_i + 1) + \lambda\mathbf{w}'\mathbf{w}\right) \\ &= (2\lambda)^{-1}L_{\mathrm{SVM}}(c, \mathbf{w}). \end{aligned}$$

showing that the two formulations (8.3) and (8.3) are exactly the same up to a scaling factor $(2\lambda)^{-1}$ and yield the same c and $\mathbf{w}$. The advantage of (8.3) lies in that it can be interpreted as a (robust) error function with a penalty. This quadratic penalty term is used for regularization much in the same way as in ridge regression, that is, to force the w_j to be close to zero. The penalty is particularly useful to avoid overfitting. Furthermore, it can be easily seen that $L_{\mathrm{SVM}}(c, \mathbf{w})$ is a convex function in c and $\mathbf{w}$ as all three terms are convex in c and $\mathbf{w}$. The minimum of $L_{\mathrm{SVM}}(c, \mathbf{w})$ must be a global one as the function is convex and bounded below by zero. Note that the formulation in (8.3) allows the problem to be treated as a quadratic program. However, in Section 8.5, we optimize (8.3) directly by the method of iterative majorization.

8.3 Other Error Functions

An advantage of clearly separating error from penalty is that it is easy to apply other error functions. Instead of the absolute hinge error in Figure 8.2a, we can use different definitions for the errors $f_1(q_i)$ and $f_{-1}(q_i)$. A straightforward alternative for the absolute hinge error is the *quadratic hinge* error, see Figure 8.2b. This error simply squares the absolute hinge error, yielding the loss function

$$L_{\mathrm{Q-SVM}}(c, \mathbf{w}) = \sum_{i \in G_1} \max(0, 1 - q_i)^2 + \sum_{i \in G_{-1}} \max(0, q_i + 1)^2 + \lambda\mathbf{w}'\mathbf{w}, \quad (8.7)$$



a. Absolute hinge error.

b. Quadratic hinge error.

c. Huber hinge error.

d. Quadratic error.

Figure 8.2: Four error functions: a. the absolute hinge error, b. the quadratic hinge error, c. the Huber hinge error, and d. the quadratic error.

see also,Vapnik (2000) and Cristianini and Shawe-Taylor (2000). It uses the quadratic error for objects that have prediction error and zero error for correctly predicted objects. An advantage of this loss function is that both error and penalty terms are quadratic. In Section 8.5, we see that the majorizing algorithm is very efficient because in each iteration a linear system is solved very efficiently. A disadvantage of the quadratic hinge error is that outliers can have a large influence on the solution.

An alternative that is smooth and robust is the *Huber hinge* error, see Figure 8.2c. This hinge error was called "Huberized squared hinge loss" by Rosset and Zhu (2007). Note that Chu, Keerthi, and Ong (2003) proposed a similar function for support vector regression. The definition of the Huber hinge is

Table 8.1: Definition of error functions that can be used in the context of SVMs.

Error	$f_{-1}(q_i)$	
Absolute hinge	$\max(0, q_i + 1)$	
Quadratic hinge	$\max(0, q_i + 1)^2$	
Huber hinge	$h_{-1}(q_i) = (1/2)(k+1)^{-1}\max(0, q_i + 1)^2$	if $q_i \leq k$
	$h_{-1}(q_i) = q_i + 1 - (k+1)/2$	if $q_i > k$
Quadratic	$(q_i + 1)^2$	
	$f_{+1}(q_i)$	
Absolute hinge	$\max(0, 1 - q_i)$	
Quadratic hinge	$\max(0, 1 - q_i)^2$	
Huber hinge	$h_{+1}(q_i) = 1 - q_i - (k+1)/2$	if $q_i \leq -k$
	$h_{+1}(q_i) = (1/2)(k+1)^{-1}\max(0, 1 - q_i)^2$	if $q_i > -k$
Quadratic	$(1 - q_i)^2$	

found in Table 8.1 and the corresponding SVM problem is defined by

$$L_{\mathrm{H-SVM}}(c, \mathbf{w}) = \sum_{i \in G_1} h_{+1}(q_i) + \sum_{i \in G_{-1}} h_{-1}(q_i) + \lambda \mathbf{w}'\mathbf{w}. \tag{8.8}$$

The Huber hinge error is characterized by a linearly increasing error if the error is large, a smooth quadratic error for errors between 0 and the linear part, and zero for objects that are correctly predicted. The smoothness is governed by a value $k \geq -1$. The Huber hinge approaches the absolute hinge for $k \downarrow -1$, so that the Huber hinge SVM loss solution can approach the classical SVM solution. If k is chosen too large, then the Huber hinge error essentially approaches the quadratic hinge function. Thus, the Huber hinge error can be seen as a compromise between the absolute and quadratic hinge errors. As we will see in Section 8.5, it is advantageous to choose k sufficiently large, for example, $k = 1$, as is done in Figure 8.2c. A similar computational efficiency as for the quadratic hinge error is also available for the Huber hinge error.

In principle, any robust error can be used. To inherit as much of the nice properties of the standard SVM it is advantageous that the error function has two properties: (1) if the error function is convex in q_i (and hence in $\mathbf{w}$), then the total loss function is also convex and hence has a global minimum that can be reached, (2) the error function should be asymmetric and have the form of a hinge so that objects that are predicted correctly induce zero error.

In Figure 8.2d the quadratic error is used, defined in Table 8.1. The quadratic error alone simply equals a multiple regression problem with a dependent variable $y_i = -1$ if $i \in G_{-1}$ and $y_i = 1$ if $i \in G_1$, that is,

$$\begin{aligned}
L_{\mathrm{MReg}}(c, \mathbf{w}) &= \sum_{i \in G_1} (1 - q_i)^2 + \sum_{i \in G_{-1}} (1 + q_i)^2 + \lambda \mathbf{w}'\mathbf{w} \\
&= \sum_{i \in G_1} (y_i - q_i)^2 + \sum_{i \in G_{-1}} (y_i - q_i)^2 + \lambda \mathbf{w}'\mathbf{w} \\
&= \sum_i (y_i - c - \mathbf{x}_i'\mathbf{w})^2 + \lambda \mathbf{w}'\mathbf{w} \\
&= \|\mathbf{y} - c\mathbf{1} - \mathbf{X}\mathbf{w}\|^2 + \lambda \mathbf{w}'\mathbf{w}. \qquad (8.9)
\end{aligned}$$

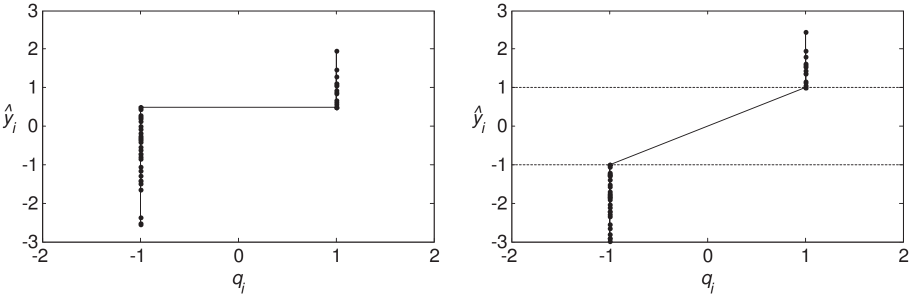
Note that for $i \in G_{-1}$ we have the equality $(1 + q_i)^2 = ((-1)(1 + q_i))^2 = (-1 - q_i)^2 = (y_i - q_i)^2$. $L_{\mathrm{MReg}}(c, \mathbf{w})$ has been extensively discussed in Suykens, Van Gestel, De Brabanter, De Moor, and Vandewalle (2002). To show that (8.9) is equivalent to ridge regression, we center the columns of $\mathbf{X}$ and use $\mathbf{JX}$ with $\mathbf{J} = \mathbf{I} - n^{-1}\mathbf{1}\mathbf{1}'$ being the centering matrix. Then (8.9) is equivalent to

$$\begin{aligned}
L_{\mathrm{MReg}}(c, \mathbf{w}) &= \|\mathbf{y} - c\mathbf{1} - \mathbf{JXw}\|^2_{n^{-1}\mathbf{11}'} + \|\mathbf{y} - c\mathbf{1} - \mathbf{JXw}\|^2_{\mathbf{J}} + \lambda \mathbf{w}'\mathbf{w} \\
&= \|\mathbf{y} - c\mathbf{1}\|^2_{n^{-1}\mathbf{11}'} + \|\mathbf{Jy} - \mathbf{JXw}\|^2 + \lambda \mathbf{w}'\mathbf{w}, \qquad (8.10)
\end{aligned}$$

where the norm notation is defined as $\|\mathbf{Z}\|^2_{\mathbf{A}} = \operatorname{tr} \mathbf{Z}'\mathbf{AZ} = \sum_{i=1}^n \sum_{j=1}^n \sum_{k=1}^K a_{ij} z_{ik} z_{jk}$. Note that (8.10) is a decomposition in three terms with the intercept c appearing alone in the first term so that it can be estimated independently of $\mathbf{w}$. The optimal c in (8.10) equals $n^{-1}\mathbf{1}'\mathbf{y}$. The remaining optimization of (8.10) in $\mathbf{w}$ simplifies into a standard ridge regression problem. Hence, the SVM with quadratic errors is equivalent to ridge regression. As the quadratic error has no hinge, even properly predicted objects with $q_i < -1$ for $i \in G_{-1}$ or $q_i > 1$ for $i \in G_1$ can receive high error. In addition, the quadratic error is nonrobust, hence can be sensitive to outliers. Therefore, ridge regression is more restrictive than the quadratic hinge error and expected to give worse predictions in general.

8.4 Optimal Scaling and SVM

Several ideas that are used in SVMs are not entirely new. In this section, we show that the application of optimal scaling known since the 1970s has almost the same aim as the SVM. Optimal scaling in a regression context goes back to the models MONANOVA (Kruskal, 1965), ADDALS (Young, De Leeuw, & Takane, 1976a), MORALS (Young, De Leeuw, & Takane, 1976b), and, more recently, CatREG (Van der Kooij, Meulman, & Heiser, 2006; Van der Kooij, 2007). The main idea of optimal scaling regression (OS-Reg) is that a variable $\mathbf{y}$ is replaced by an optimally transformed variable $\widehat{\mathbf{y}}$. The regression loss function is not only optimized over the usual weights, but also over the optimally scaled



a. Optimal scaling transformation by primary approach to ties.

b. Optimal scaling transformation for SVMs.

Figure 8.3: Optimal scaling transformation $\widehat{\mathbf{y}}$ of the dependent variable $\mathbf{y}$. Panel a. shows an example transformation for the OS-Reg, Panel b. for SVM.

variable $\widehat{\mathbf{y}}$. Many transformations are possible, see, for example, Gifi (1990). However, to make OS-Reg suitable for the binary classification problem, we use the so-called ordinal transformation with the primary approach to ties that allows to untie the tied data. This transformation was proposed in the context of multidimensional scaling to optimally scale the ordinal dissimilarities. As we are dealing with two groups only, this means that the only requirement is to constrain all $\widehat{y}_i$ in G_{-1} to be smaller than or equal to all $\widehat{y}_j$ in G_1. An example of such a transformation is given in Figure 8.3a.

OS-Reg can be formalized by minimizing

$$L_{\text{OS-Reg}}(\widehat{\mathbf{y}}, \mathbf{w}) = \sum_{i=1}^{n} (\widehat{y}_i - \mathbf{x}_i'\mathbf{w})^2 + \lambda \mathbf{w}'\mathbf{w} = \|\widehat{\mathbf{y}} - \mathbf{X}\mathbf{w}\|^2 + \lambda \mathbf{w}'\mathbf{w} \qquad (8.11)$$

subject to $\widehat{y}_i \leq \widehat{y}_j$ for all combinations of $i \in G_{-1}$ and $j \in G_1$ and $\widehat{\mathbf{y}}'\widehat{\mathbf{y}} = n$. The latter requirement is necessary to avoid the degenerate zero-loss solution of $\widehat{\mathbf{y}} = \mathbf{0}$ and $\mathbf{w} = \mathbf{0}$. Without loss of generality, we assume that $\mathbf{X}$ is column centered here. In the usual formulation, no penalty term is present in (8.11), but here we add it because of ease of comparison with SVMs.

The error part of an SVM can also be expressed in terms of an optimally scaled variable $\widehat{\mathbf{y}}$. Then, the SVM loss becomes

$$L_{\text{SVM-Abs}}(\widehat{\mathbf{y}}, \mathbf{w}, c) = \sum_{i=1}^{n} |\widehat{y}_i - \mathbf{x}_i'\mathbf{w} - c| + \lambda \mathbf{w}'\mathbf{w} \qquad (8.12)$$

subject to $\widehat{y}_i \leq -1$ if $i \in G_{-1}$ and $\widehat{y}_i \geq 1$ if $i \in G_1$. Clearly, for $i \in G_{-1}$ a zero error is obtained if $\mathbf{x}_i'\mathbf{w} + c \leq -1$ by choosing $\widehat{y}_i = \mathbf{x}_i'\mathbf{w} + c$. If $\mathbf{x}_i'\mathbf{w} + c > -1$, then the restriction $\widehat{y}_i \leq -1$ becomes active so that $\widehat{y}_i$ must be chosen as -1. Similar reasoning holds for $i \in G_1$, where $\widehat{y}_i = \mathbf{x}_i'\mathbf{w} + c$ if $\mathbf{x}_i'\mathbf{w} + c \geq 1$ (yielding zero error) and $\widehat{y}_i = 1$ if $\mathbf{x}_i'\mathbf{w} + c < 1$.

Just as the SVM, OS-Reg also has a limited number of support vectors. All objects i that are below or above the horizontal line yield zero error. All objects i that are have a value $\widehat{y}_i$ that is on the horizontal line generally give error, hence are support vectors.

The resemblances of SVM and OS-Reg is that both can be used for the binary classification problem, both solutions only use the support vectors, and both can be expressed in terms of an optimal scaled variable $\widehat{\mathbf{y}}$. Although, the SVM estimates the intercept c, OS-Reg implicitly estimates c by leaving the position free where the horizontal line occurs, whereas the SVM attains this freedom by estimating c. One of the main differences is that OS-Reg uses squared error whereas SVM uses the absolute error. Also, in its standard form $\lambda = 0$ so that OS-Reg does not have a penalty term. A final difference is that OS-Reg solves the degenerate zero loss solution of $\widehat{\mathbf{y}} = \mathbf{0}$ and $\mathbf{w} = \mathbf{0}$ by imposing the length constraint $\widehat{\mathbf{y}}'\widehat{\mathbf{y}} = n$ whereas the SVM does this through setting a minimum difference of 2 between $\widehat{y}_i$ and $\widehat{y}_j$ if i and j are from different groups.

In some cases with $\lambda = 0$, we found occasionally OS-Reg solutions where one of the groups collapsed at the horizontal line and the some objects of the other group were split into two points: one also at the horizontal line, the other at a distinctly different location. In this way, the length constraint is satisfied, but it is hardly possible to distinguish the groups. Fortunately, these solutions do not occur often and they never occurred with an active penalty term ($\lambda > 0$).

8.5 SVM-Maj: A Majorizing Algorithm for SVM with Robust Hinge Errors

The SVM literature often solves the SVM problem by changing to the dual of (8.3) and expressing it as a quadratic program that subsequently is solved by special quadratic program solvers. A disadvantage of these solvers is that they may become computationally slow for large number of objects n (although fast specialized solvers exist). Here, we use a different minimization approach based on iterative majorization (IM) algorithm applied to the primal SVM problem. One of the advantages of IM algorithms is that they guarantee descent, that is, in each iteration the SVM loss function is reduced until no improvement is possible. As the resulting SVM loss function for each of the three hinge errors is convex, the IM algorithm will stop when the estimates are sufficiently close to the global minimum. The combination of these properties forms the main strength of the majorization algorithm. In principle, a majorization algorithm can be derived for any error function that has a bounded second derivative as most robust errors have.

The general method of iterative majorization can be understood as follows. Let $f(\mathbf{q})$ be the function to be minimized. Then, iterative majorization makes use of an auxiliary function, called the majorizing function $g(\mathbf{q}, \overline{\mathbf{q}})$, that is dependent on $\mathbf{q}$ and the previous (known) estimate $\overline{\mathbf{q}}$. There are requirements on the majorizing function $g(\mathbf{q}, \overline{\mathbf{q}})$: (1) it should touch f at the supporting point $\mathbf{y}$, that is, $f(\overline{\mathbf{q}}) = g(\overline{\mathbf{q}}, \overline{\mathbf{q}})$, (2) it should never be below f, that is, $f(\mathbf{q}) \leq g(\mathbf{q}, \overline{\mathbf{q}})$, and (3) $g(\mathbf{q}, \overline{\mathbf{q}})$ should be simple, preferably linear or quadratic in $\mathbf{q}$. Let $\mathbf{q}^*$ be such that $g(\mathbf{q}^*, \overline{\mathbf{q}}) \leq g(\overline{\mathbf{q}}, \overline{\mathbf{q}})$, for example, by choosing $\mathbf{q}^* = \arg\min_{\mathbf{q}} g(\mathbf{q}, \mathbf{q}^*)$. As the majorizing function is never below the original function, we obtain the so called sandwich inequality

$$f(\mathbf{q}^*) \leq g(\mathbf{q}^*, \overline{\mathbf{q}}) \leq g(\overline{\mathbf{q}}, \overline{\mathbf{q}}) = f(\overline{\mathbf{q}}).$$

This chain of inequalities shows that the update $\mathbf{q}^*$ obtained by minimizing the majorizing function never increases f and usually decreases it. This constitutes a single iteration. By repeating these iterations, a monotonically nonincreasing (generally a decreasing) series of loss function values f is obtained. For convex f and after a sufficient number of iterations, the IM algorithm stops at a global minimum. More detailed information on iterative majorization can be found in De Leeuw (1994), Heiser (1995), Lange, Hunter, and Yang (2000), Kiers (2002), and Hunter and Lange (2004) and an introduction in Borg and Groenen (2005).

An additional property of IM is useful for developing the algorithm. Suppose we have two functions, $f_1(\mathbf{q})$ and $f_2(\mathbf{q})$, and each of these functions can be majorized, that is, $f_1(\mathbf{q}) \leq g_1(\mathbf{q}, \overline{\mathbf{q}})$ and $f_2(\mathbf{q}) \leq g_1(\mathbf{q}, \overline{\mathbf{q}})$. Then, the function $f(\mathbf{q}) = f_1(\mathbf{q}) + f_2(\mathbf{q})$ can be majorized by $g(\mathbf{q}) = g_1(\mathbf{q}, \overline{\mathbf{q}}) + g_2(\mathbf{q}, \overline{\mathbf{q}})$ so that the following majorizing inequality holds:

$$f(\mathbf{q}) = f_1(\mathbf{q}) + f_2(\mathbf{q}) \leq g_1(\mathbf{q}, \overline{\mathbf{q}}) + g_2(\mathbf{q}, \overline{\mathbf{q}}) = g(\mathbf{q}, \overline{\mathbf{q}}).$$

For notational convenience, we refer in the sequel to the majorizing function as $g(\mathbf{q})$ without the implicit argument $\overline{\mathbf{q}}$.

To find an algorithm, we need to find a majorizing function for (8.3). For the moment, we assume that a quadratic majorizing function exists for each individual error term of the form

$$\begin{aligned} f_{-1}(q_i) &\leq a_{-1i}q_i^2 - 2b_{-1i}q_i + c_{-1i} = g_{-1}(q_i) && (8.13) \\ f_1(q_i) &\leq a_{1i}q_i^2 - 2b_{1i}q_i + c_i = g_1(q_i). && (8.14) \end{aligned}$$

Then, we combine the results for all terms and come up with the total majorizing function that is quadratic in c and $\mathbf{w}$ so that an update can be readily derived. In the next subsection, we derive the SVM-Maj algorithm for general hinge errors assuming that (8.13) and (8.14) are known for the specific hinge error. In the appendix, we derive $g_{-1}(q_i)$ and $g_1(q_i)$ for the absolute, quadratic, and Huber hinge error SVM.

8.5.1 The SVM-Maj Algorithm

Equation (8.3) was derived with the absolute hinge error in mind. Here, we generalize the definitions of the error functions $f_{-1}(q)$ and $f_1(q)$ in (8.3) to be any of the three hinge errors discussed above so that $L_{\mathrm{Q-SVM}}$ and $L_{\mathrm{H-SVM}}$ become special cases of L_{SVM}. For deriving the SVM-Maj algorithm, we assume that (8.13) and (8.14) are known for these hinge losses. Figure 8.4 shows that this is the case indeed. Then, let

$$\begin{aligned} a_i &= \begin{cases} \max(\delta, a_{-1i}) & \text{if } i \in G_{-1}, \\ \max(\delta, a_{1i}) & \text{if } i \in G_1, \end{cases} && (8.15) \\ b_i &= \begin{cases} b_{-1i} & \text{if } i \in G_{-1}, \\ b_{1i} & \text{if } i \in G_1, \end{cases} && (8.16) \\ c_i &= \begin{cases} c_{-1i} & \text{if } i \in G_{-1}, \\ c_{1i} & \text{if } i \in G_1. \end{cases} && (8.17) \end{aligned}$$

Summing all the individual terms leads to the majorization inequality

$$L_{\mathrm{SVM}}(c, \mathbf{w}) \leq \sum_{i=1}^{n} a_i q_i^2 - 2\sum_{i=1}^{n} b_i q_i + \sum_{i=1}^{n} c_i + \lambda \sum_{j=1}^{m} w_j^2. \qquad (8.18)$$

It is useful to add an extra column of ones as the first column of $\mathbf{X}$ so that $\mathbf{X}$ becomes $n \times (m+1)$. Let $\mathbf{v}' = [c\ \mathbf{w}']$ so that $q_i = c + \mathbf{x}_i'\mathbf{w}_i$ can be expressed as $\mathbf{q} = \mathbf{X}\mathbf{v}$. Then, (8.3) can be majorized as

$$\begin{aligned} L_{\mathrm{SVM}}(\mathbf{v}) &\leq \sum_{i=1}^{n} a_i(\mathbf{x}_i'\mathbf{v})^2 - 2\sum_{i=1}^{n} b_i \mathbf{x}_i'\mathbf{v} + \sum_{i=1}^{n} c_i + \lambda \sum_{j=2}^{m+1} v_j^2 \\ &= \mathbf{v}'\mathbf{X}'\mathbf{A}\mathbf{X}\mathbf{v} - 2\mathbf{v}'\mathbf{X}'\mathbf{b} + c_m + \lambda\mathbf{v}'\mathbf{P}\mathbf{v} \\ &= \mathbf{v}'(\mathbf{X}'\mathbf{A}\mathbf{X} + \lambda\mathbf{P})\mathbf{v} - 2\mathbf{v}'\mathbf{X}'\mathbf{b} + c_m, \qquad (8.19) \end{aligned}$$

where $\mathbf{A}$ is a diagonal matrix with elements a_i on the diagonal, $\mathbf{b}$ is a vector with elements b_i, and $c_m = \sum_{i=1}^{n} c_i$, and $\mathbf{P}$ is the identity matrix except for element $p_{11} = 0$. If $\mathbf{P}$ were $\mathbf{I}$, then the last line of (8.19) would be of the same form as a ridge regression. Differentating the last line of (8.19) with respect to $\mathbf{v}$ yields the system of equalities linear in $\mathbf{v}$

$$(\mathbf{X}'\mathbf{A}\mathbf{X} + \lambda\mathbf{P})\mathbf{v} = \mathbf{X}'\mathbf{b}. \qquad (8.20)$$

The update $\mathbf{v}^+$ solves this set of linear equalities, for example, by Gaussian elimination, or, less efficiently, by

$$\mathbf{v}^+ = (\mathbf{X}'\mathbf{A}\mathbf{X} + \lambda\mathbf{P})^{-1}\mathbf{X}'\mathbf{b}. \qquad (8.21)$$

Because of the substitution $\mathbf{v}' = [c\ \mathbf{w}']$, the update of the intercept is $c^+ = v_1$ and $w_j^+ = v_{j+1}^+$ for $j = 1, \ldots, m$. The update $\mathbf{v}^+$ forms the heart of the majorization algorithm for SVMs.

Extra computational efficiency can be obtained for the quadratic and Huber hinge errors for which $a_{-1i} = a_{1i} = a$ for all i and this a does not depend on $\overline{\mathbf{q}}$. In these cases, (8.21) simplifies into

$$\mathbf{v}^+ = (a\mathbf{X}'\mathbf{X} + \lambda\mathbf{P})^{-1}\mathbf{X}'\mathbf{b}.$$

Thus, the $m \times n$ matrix $\mathbf{S} = (a\mathbf{X}'\mathbf{X}+\lambda\mathbf{P})^{-1}\mathbf{X}'$ can be computed once and stored in memory, so that the update (8.21) simply amounts to setting $\mathbf{v}^+ = \mathbf{Sb}$. In this case, a single matrix multiplication of the $m \times n$ matrix $\mathbf{S}$ with the $n \times 1$ vector $\mathbf{b}$ is required to obtain an update in each iteration. Therefore, SVM-Maj for the Huber and quadratic hinge will be particularly efficient, even for large n as long as m is not too large.

The SVM-Maj algorithm for minimizing the SVM loss function in (8.3) is summarized in Algorithm 1. Note that SVM-Maj handles the absolute, quadratic, and Huber hinge errors. The advantages of SVM-Maj are the following. First, SVM-Maj approaches the global minimum closer in each iteration. In contrast, quadratic programming of the dual problem needs to solve the dual problem completely to have the global minimum of the original primal problem. Secondly, the progress can be monitored, for example, in terms of the changes in the number of misclassified objects. If no changes occur, then the iterations can be stopped. Thirdly, the computational time could be reduced, for example, by using smart initial estimates of c and $\mathbf{w}$ available from a previous cross validation run. Note that in each majorization iteration a ridge regression problem is solved so that the SVM-Maj algorithm can be seen as a solution to the SVM problem via successive solutions of ridge regressions.

A visual illustration of a single iteration of the SVM-Maj algorithm is given in Figure 8.5 for the absolute hinge SVM. We fixed c at its optimal value and the minimization is done only over $\mathbf{w}$, that is, over w_1 and w_2. Therefore, each point in the horizontal plane represents a combination of w_1 and w_2. In the same horizontal plane, the class 1 points are represented as open circles and the class -1 points as closed circles. The horizontal plane also shows the separation line and the margins corresponding to the current estimates of w_1 and w_2. It can be seen that the majorization function is indeed located above the original function and touches it at the dotted line, that is, at the current w_1 and w_2. At the location (w_1, w_2) where this majorization function finds its minimum, $L_{\text{SVM}}(c, \mathbf{w})$ is lower than at the previous estimate, so $L_{\text{SVM}}(c, \mathbf{w})$ has decreased.

```
Algorithm: SVM-Maj
input  : $\mathbf{y}, \mathbf{X}, \lambda, \epsilon$, Hinge, $k$
output: $c_t, \mathbf{w}_t$
$t = 0$;
Set $\epsilon$ to a small positive value;
Set $\mathbf{w}_0$ and $c_0$ to random initial values;
if Hinge = Huber or Quadratic then
    if Hinge = Quadratic then $a = 1$;
    if Hinge = Huber then $a = (1/2)(k+1)^{-1}$;
    $\mathbf{S} = (a\mathbf{X}'\mathbf{X} + \lambda\mathbf{P})^{-1}\mathbf{X}'$;
end
Compute $L_{\mathrm{SVM}}(c_0, \mathbf{w}_0)$ according to (8.3);
while $t = 0$ or $(L_{t-1} - L_{\mathrm{SVM}}(c_t, \mathbf{w}_t))/L_{\mathrm{SVM}}(c_t, \mathbf{w}_t) > \epsilon$ do
    $t = t + 1$;
    $L_{t-1} = L_{\mathrm{SVM}}(c_{t-1}, \mathbf{w}_{t-1})$;
    Comment:Compute A and b for different hinge errors
    if Hinge = Absolute then
        Compute $a_i$ by (8.22) if $i \in G_{-1}$ and by (8.25) if $i \in G_1$;
        Compute $b_i$ by (8.23) if $i \in G_{-1}$ and by (8.26) if $i \in G_1$;
    else if Hinge = Quadratic then
        Compute $b_i$ by (8.29) if $i \in G_{-1}$ and by (8.32) if $i \in G_1$;
    else if Hinge = Huber then
        Compute $b_i$ by (8.35) if $i \in G_{-1}$ and by (8.38) if $i \in G_1$;
    end
    Make the diagonal matrix $\mathbf{A}$ with elements $a_i$;
    Comment:Compute update
    if Hinge = Absolute then
        Find $\mathbf{v}$ by that solves (8.20): $(\mathbf{X}'\mathbf{A}\mathbf{X} + \lambda\mathbf{P})\mathbf{v} = \mathbf{X}'\mathbf{b}$;
    else if Hinge = Huber or Quadratic then
        $\mathbf{v} = \mathbf{S}\mathbf{b}$;
    end
    Set $c_t = v_1$ and $w_{tj} = v_{j+1}$ for $j = 1, \ldots, m$;
end
```

Algorithm 1: The SVM majorization algorithm SVM-Maj.

Table 8.2: Information on the 14 data sets used in the experiments. n_1 and n_{-1} are the number of observations with $y_i = 1$ and $y_i = -1$, respectively. Sparsity equals the percentage of zeros in the data set. A data set with scaled attributes has maximum values +1 and minimum values −1.

Dataset	Source	n	n_1	n_{-1}	m	Sparsity	Notes
Australian	UCI	690	307	383	14	20.04	
Breast_cancer_w	UCI	699	458	241	9	0.00	
Heart_statlog	UCI	270	120	150	13	0.00	Standardized data
Hepatitis	UCI	155	123	32	19	39.86	
Sonar	UCI	208	97	111	60	0.07	
Voting	UCI	434	167	267	16	45.32	
Liver-disorders	LibSVM	345	200	145	6	0.00	
Liver-disorders2	LibSVM	345	200	145	6	0.92	Scaled attributes
Diabetes	LibSVM	768	500	268	8	0.00	
Diabetes2	LibSVM	768	500	268	8	0.15	Scaled attributes
Ionosphere	LibSVM	351	225	126	34	11.59	Scaled attributes
German.number	LibSVM	1000	300	700	24	0.00	
German.number2	LibSVM	1000	300	700	24	4.16	Scaled attributes
Splice	LibSVM	1000	517	483	60	0.00	

8.6 Experiments

To investigate the performance of the various variants of SVM algorithms, we report experiments on several data sets from the UCI repository (Newman et al., 1998) and the home page of LibSVM software (Chang & Lin, 2006). These data sets cover a wide range of characteristics such as extent of being unbalanced (one group is larger than the other), number of observations n, ratio of observations to attributes m/n, and sparsity (the percentage of nonzero attribute values x_{ij}). More information on the data sets are given in Table 8.2.

In the experiments, we applied the standard absolute hinge (ϵ-insensitive), the Huber hinge and quadratic hinge SVM loss functions. All experiments have been carried out in Matlab 7.2, on a 2.8Ghz Intel processor with 2GB of memory under Windows XP. The performance of the majorization algorithms is compared to those of the off-the-shelf programs LibSVM, BSVM (Hsu & Lin, 2006), SVM-Light (Joachims, 1999), and SVM-Perf (Joachims, 2006). Although these programs can handle nonlinearity of the predictor variables by using special kernels, we limit our experiments to the linear kernel. Note that not all of these SVM-solvers are optimized for the linear kernel. In addition, no comparison between majorization is possible for the Huber hinge loss function as it is not supported by these solvers.

The numerical experiments address several issues. First, how well are the different hinge losses capable of predicting the two groups? Second, we focus on the performance of the majorization algorithm with respect to its competitors. We would like to know how the time needed for the algorithm to converge scales with the number of observations n, the strictness of the stopping criterion, and with λ; what is a suitable level for the stopping criterion.

To answer these questions, we consider the following measures. First, we define convergence between two steps as the relative decrease in loss between two subsequent steps, that is, by $L_{\text{diff}} = (L_{t-1} - L_t)/L_t$. The error rate in the training data set is defined as the number of misclassified cases. To measure how well a solution predicts, we define the accuracy as the percentage correctly predicted out-of-sample cases in 5-fold cross validation.

8.6.1 Predictive Performance for the Three Hinge Errors

It is interesting to compare the performance of the three hinge loss functions. Consider Table 8.3, which compares the 5-fold cross-validation accuracy for the three different loss function. For each data set, we tried a grid of λ values ($\lambda = 2^p$ for $p = -15, -14.5, -14, \ldots, 7.5, 8$ where $2^{-15} = 0.000030518$ and $2^8 = 256$). Alongside are given the values of the optimal λ's and times to convergence (stop whenever $L_{\text{diff}} < 3 \times 10^{-7}$). From the accuracy, we see that there is no one best loss function that is suitable for all data sets. The absolute hinge is best in 5 of the cases, the Huber hinge is best in 4 of the cases, and the quadratic hinge is best in 7 of the cases. The total number is greater than 14 due to equal accuracies. In terms of computational speed, the order invariably is: absolute hinge is the slowest, Huber hinge is faster, and the quadratic hinge is the fastest.

The implementation of optimal scaling regression was also done in MatLab, but the update in each iteration for $\widehat{\mathbf{y}}$ by monotone regression using the primary approach to ties was calculated by a compiled Fortran subroutine. Therefore, the CPU time is not comparable to those of the other SVM methods that were solely programmed in MatLab. Optimal scaling regression performs well on three data sets (Breast cancer, Diabetes and Diabetes2) where the accuracy is better than the three SVM methods. On the remaining data sets, the accuracy is worse or much worse when compared to the SVM methods. It seems that in some cases OS regression can predict well, but its poor performance for the majority of the data sets makes it hard to use it as a standard method for the binary classification problem. One of the reasons for the poor performance could be due to solutions where all $\hat{y}_i$s of one of the two classes collapses in the inequality constraint and the $\hat{y}_i$s of the other class remain to have variance. In a transformation plot like Figure 8.3 this situation means that the vertical scatter of either the -1 or 1 class collapses into a single point. By definition, the SVM transformation cannot suffer from this problem. More study is needed to understand if the collapse is the only reason for bad performance of OS-Reg, and, if possible, provide adaptations that make it work better for more data sets.

Table 8.3: Performance of SVM models for the three hinge loss functions (Abs., Hub., and Quad.) and optimal scaling regression (OS). The optimal $\lambda = 2^p$ is computed by 5 fold cross validation. CPU-time to convergence for the optimal λ and the prediction accuracy (in %) is obtained for the 14 different test data sets from Table 8.2.

	Optimal p				CPU time in sec.				5-fold CV accuracy			
Data set	Abs.	Hub.	Quad.	OS	Abs.	Hub.	Quad.	OS	Abs.	Hub.	Quad.	OS
Australian	-0.5	2.0	3.0	-15.0	0.20	0.12	0.14	0.14	85.4	86.7	86.7	18.1
Breast_cancer_w	7.5	6.0	8.0	-15.0	0.13	0.11	0.04	0.49	96.7	96.6	96.7	97.7
Heart_statlog	0.0	5.5	7.0	5.5	0.03	0.01	0.01	0.04	84.4	84.4	84.4	9.3
Hepatitis	0.0	0.0	2.0	-8.5	0.04	0.02	0.01	0.41	85.8	87.1	86.5	69.0
Sonar	0.5	1.5	1.5	-2.0	0.06	0.02	0.02	0.15	77.4	76.9	78.4	13.5
Voting	-5.5	-1.5	-0.5	-0.5	0.46	0.11	0.10	0.35	97.0	96.8	97.0	6.2
Liver-disorders	3.0	8.0	2.5	8.0	0.05	0.02	0.01	0.07	68.7	68.1	66.1	24.3
Liver-disorders2	-7.0	-3.0	1.0	-2.5	0.07	0.02	0.01	0.07	68.4	68.1	66.4	24.6
Diabetes	1.0	0.5	3.5	7.5	0.24	0.04	0.01	0.07	77.3	78.0	77.6	79.0
Diabetes2	-2.0	-2.5	4.0	8.0	0.13	0.04	0.01	0.03	77.6	78.3	77.9	80.1
Ionosphere	-5.0	2.5	-0.5	-8.0	4.03	0.10	0.17	5.62	90.3	89.5	90.6	25.1
German.number	-0.5	-4.0	3.5	3.5	1.71	0.17	0.03	0.18	77.2	77.0	77.3	26.2
German.number2	2.5	2.0	-0.5	1.5	0.81	0.16	0.03	0.37	77.2	77.0	77.2	33.0
Splice	5.0	7.0	-0.5	3.5	3.21	0.34	0.14	0.61	80.7	81.4	81.1	13.0

8.6.2 Computational Efficiency of SVM-Maj

To see how computationally efficient the majorization algorithms are, two types of experiments were done. In the first experiment, the majorization algorithm is studied and tuned. In the second, the majorization algorithm SVM-Maj for the absolute hinge error is compared with several off-the-shelf programs that minimize the same loss function.

As the majorization algorithm is guaranteed to improve the $L_{\text{SVM}}(c, \mathbf{w})$ in each iteration by taking a step closer to the final solution, the computational efficiency of SVM-Maj is determined by its stopping criterion. The iterations of SVM-Maj stop whenever $L_{\text{diff}} < \epsilon$. It is also known that majorization algorithms have a linear convergence rate (De Leeuw, 1994), which can be slow especially for very small ϵ. Therefore, we study the relations between four measures as they change during the iterations: (a) the difference between present and final loss, $L_t - L_{\text{final}}$, (b) the convergence L_{diff}, (c) CPU time spent sofar, and (d) the difference between current and final within sample error rate.

Figure 8.6 shows the relationships between these measures for three exemplary data sets: Liver disorders, Sonar and Australian. Note that in Figures 8.6c and 8.6d the direction of the horizontal axis is reversed so that in all four panels the right side of the horizontal axis means more computational investment. Figure 8.6a draws the relationship between CPU-time and $L_t - L_{\text{final}}$, with L_{final} the objective function values obtained at convergence with $\epsilon = 3 \times 10^{-7}$. Notice that in most of the cases the first few iterations are responsible for the bulk of the decreases in the objective function values and most of the CPU time is spent to obtain small decreases in loss function values. Figure 8.6b shows the relationship between $L_t - L_{\text{final}}$ and the convergence L_{diff} that is used as a stopping criterion. The two lower panels show the development of the within sample error rate and CPU time (Figure 8.6c) and convergence L_{diff} (Figure 8.6d). To evaluate whether it is worthwhile using a looser stopping criterion, it is in instructive to observe the path of the error rate over the iterations (the lower right panel). It seems that the error rate stabilizes for values of L_{diff} below 10^{-6}. Nevertheless, late-time changes sometimes occur in other data sets. Therefore, it does not seem recommendable to stop the algorithm much earlier, hence our recommendation of using $\epsilon = 3 \times 10^{-}7$.

The analogues of Figures 8.6 and 8.7 were also produced for the Huber hinge and quadratic hinge loss functions. Overall, the same patterns as for the absolute hinge function can be distinguished, with several differences: the objective function decreases much faster (relative to CPU time), and the error rate stabilizes already at slightly greater values for the convergence criterion. In addition, the number of iterations until convergence by and large decline (vis-a-vis the absolute hinge function).

Figure 8.7 investigates how sensitive the speed of SVM-Maj is relative to changes in the values of λ for four illustrative datasets (Splice, German-number with scaled attributed, Ionosphere, and Sonar). As expected, the relationship appears to be decreasing. Thus, for large λ the penalty term dominates L_{SVM} and SVM-Maj with the absolute hinge does not need too many iterations to converge. Note that the same phenomenon is in general observed for the other SVM-solvers as well so that, apparently, the case for large λ is an easier problem to solve.

8.6.3 Comparing Efficiency of SVM-Maj with Absolute Hinge

The efficiency of SVM-Maj can be compared with off-the-shelf programs for the absolute hinge error. As competitors of SVM-Maj, we use LibSVM, BSVM, SVM-Light, and SVM-Perf. Note that the BSVM loss function differs from the standard SVM loss function by additionally penalizing the intercept. Nevertheless, we keep BSVM in our comparison to compare its speed against the others. We use the same 14 data sets as before. As SVM-Maj, LibSVM, SVM-Light, and SVM-Perf minimize exactly the same loss function L_{SVM} they all should have the same global minimum. In addition to L_{SVM}, the methods are compared on speed (CPU-time in seconds) at optimal levels of the $\lambda = 2^p$ (or equivalent) parameter. Note that the optimal levels of λ could differ slightly between methods as the off-the-shelf programs perform their own grid search for determining the optimal λ, that could be slightly different from those reported in Table 8.3. We note that the relationship between the λ parameter in SVM-Maj and the C parameter in LibSVM and SVM-light is given by $\lambda = 0.5/C$. For SVM-Maj, we choose three stopping criteria, that is, the algorithm is stopped whenever L_{diff} is respectively smaller than $10^{-4}, 10^{-5}$, and 10^{-6}.

For some data sets, it was not possible to run the off-the-shelf programs, sometimes because the memory requirements were too large, sometimes because no convergence was obtained. Such problems occurred for three data sets with SVM-Perf and two data sets with SVM-Light. Table 8.4 shows the results. Especially for $\epsilon = 10^{-6}$, SVM-Maj gives solutions that are close to the best minimum found. Generally, Lib-SVM and SVM-Light obtain the lowest L_{SVM}. SVM-Maj performs well with $\epsilon = 10^{-6}$, but even better values can be obtained by a stronger convergence criterion. Even though the loss function is slightly different, BSVM finds proper minima but is not able to handle all data sets. In terms of speed SVM-Maj is faster than its competitors in almost all cases. Of course, a smaller ϵ increases the CPU-time of SVM-Maj. Nevertheless, even for $\epsilon = .0001$ good solutions can be found in a short CPU-time.

These results are also summarized in Figure 8.8, where SVM-Maj is used with the default convergence criterion of $\epsilon = 3 \times 10^{-7}$. As far as speed is concerned (see Figure 8.8a), SVM-Maj ranks consistently amongst the fastest method. The quality of SVM-Maj is also consistently good as it has the same loss function as the global minimum with differences occurring less then 0.01.

Note that SVM-Perf finds consistently much higher loss function values than SVM-Maj, LibSVM and SVM-Light. Generally, the best quality solutions are obtained by LibSVM and SVM-Light although they tend to use more CPU time reaching it.

8.7 Conclusions and Discussion

We have discussed how linear SVM can be viewed as a the minimization of a robust error function with a regularization penalty. The regularization is needed to avoid overfitting in the case when the number of predictor variables increases. We provided a new majorization algorithm for the minimization of the primal SVM problem. This algorithm handles the standard absolute hinge error, the quadratic hinge error, and the newly proposed Huber hinge error. The latter hinge is smooth everywhere yet is linear for large errors. The majorizing algorithm has the advantage that it operates on the primal, is easy to program, and can easily be adapted for robust hinge errors. We also showed that optimal scaling regression has several features in common with SVMs. Numerical experiments on fourteen empirical data sets showed that there is no clear difference between the three hinge errors in terms of cross validated accuracy. The speed of SVM-Maj for the absolute hinge error is similar or compares favorably to the off-the-shelf programs for solving linear SVMs.

There are several open issues and possible extensions. First, the SVM-Maj algorithm is good for situations where the number of objects n is (much) larger than the number of variables m. The reason is that each iteration solves an $(m+1) \times (m+1)$ linear system. As m grows, each iteration becomes slower. Other majorization inequalities can be used to solve this problem yielding fast iterations at the cost of making (much) smaller steps in each iteration. A second limitation is the size of n. Eventually, when n gets large, the iterations will become slow. The good thing about SVM-Maj is that each iteration is guaranteed to improve the SVM-Loss. The bad thing is that at most linear convergence can be reached so that for large n one has to be satisfied with an approximate solution only. However, for m not too large and even reasonably large n, SVM-Maj should work fine and be competitive. The SVM-Maj for the quadratic and Huber hinge are computationally more efficient than the absolute hinge, so they result in a faster algorithm, even for reasonably large n.

Second, this paper has focussed on linear SVMs. Nonlinearity can be brought in in two ways. In Groenen et al. (2007), we proposed to use optimal scaling for the transformation of the predictor variables. Instead of using kernels, we propose to use I-splines to accommodate nonlinearity in the predictor space. The advantage of this approach is that it can be readily applied in any linear SVM algorithm. The standard way of introducing nonlinearity in SVMs is by using kernels. We believe that this is also possible for SVM-Maj and intend to study this possibility in future publications.

SVMs can be extended to problems with more than two classes in several ways. If the extension has error terms of the form $f_1(q)$ or $f_{-1}(q)$ then the present majorization results can be readily applied for an algorithm. We believe that applying majorization to SVMs is a fruitful idea that opens new applications and extensions to this area of research.

A Majorizing the Hinge Errors

Here we derive the quadratic majorizing functions for the three hinge functions.

A.1 Majorizing the Absolute Hinge Error

Consider the term $f_{-1}(q) = \max(0, q + 1)$. For notational convenience, we drop the subscript i for the moment. The solid line in Figure 8.2a shows $f_{-1}(q)$. Because of its shape of a hinge, we have called this function the absolute hinge function. Let $\overline{q}$ be the known error q of the previous iteration. Then, a majorizing function for $f_{-1}(q)$ is given by $g_{-1}(q, \overline{q})$ at the supporting point $\overline{q} = 2$. We want $g_{-1}(q)$ to be quadratic so that it is of the form $g_{-1}(q) = a_{-1}q^2 - 2b_{-1}q + c_{-1}$. To find a_{-1}, b_{-1}, and c_{-1}, we impose two supporting points, one at $\overline{q}$ and the other at $-2 - \overline{q}$. These two supporting points are located symmetrically around -1. Note that the hinge function is linear at both supporting points, albeit with different gradients. Because $g_{-1}(q)$ is quadratic, the additional requirement that $f_{-1}(q) \leq g_{-1}(q)$ is satisfied if $a_{-1} > 0$ and the derivatives at the two supporting points of $f_{-1}(q)$ and $g_{-1}(q)$ are the same. More formally, the requirements are that

$$\begin{aligned} f_{-1}(\overline{q}) &= g_{-1}(\overline{q}), \\ f'_{-1}(\overline{q}) &= g'_{-1}(\overline{q}), \\ f_{-1}(-2 - \overline{q}) &= g_{-1}(-2 - \overline{q}), \\ f'_{-1}(-2 - \overline{q}) &= g'_{-1}(-2 - \overline{q}), \\ f_{-1}(q) &\leq g_{-1}(q). \end{aligned}$$

It can be verified that the choice of

$$a_{-1} = \tfrac{1}{4}|\overline{q} + 1|^{-1}, \tag{8.22}$$

$$b_{-1} = -a_{-1} - \tfrac{1}{4}, \tag{8.23}$$

$$c_{-1} = a_{-1} + \tfrac{1}{2} + \tfrac{1}{4}|\overline{q} + 1|, \tag{8.24}$$

satisfies all these requirements. Figure 8.4a shows the majorizing function $g_{-1}(q)$ with supporting points $\overline{q} = 1.5$ as the dotted line.

For Class 1, a similar majorizing function can be found for $f_1(q) = \max(0, 1 - q)$. However, in this case, we require equal function values and first derivative at $\overline{q}$ and at $2 - \overline{q}$, that is, symmetric around 1. The requirements are

$$\begin{aligned} f_1(\overline{q}) &= g_1(\overline{q}), \\ f'_1(\overline{q}) &= g'_1(\overline{q}), \\ f_1(2 - \overline{q}) &= g_1(2 - \overline{q}), \\ f'_1(2 - \overline{q}) &= g'_1(2 - \overline{q}), \\ f_1(q) &\leq g_1(q). \end{aligned}$$

Choosing

$$\begin{aligned} a_1 &= \tfrac{1}{4}|1-\overline{q}|^{-1} && (8.25)\\ b_1 &= a_1 + \tfrac{1}{4} && (8.26)\\ c_1 &= a_1 + \tfrac{1}{2} + \tfrac{1}{4}|1-\overline{q}| && (8.27) \end{aligned}$$

satisfies these requirements. The functions $f_1(q)$ and $g_1(q)$ with supporting points $\overline{q} = 2$ or $\overline{q} = 0$ are plotted in Figure 8.4a.

Note that a_{-1} is not defined if $\overline{q} = -1$. In that case, we choose a_{-1} as a small positive constant δ that is smaller than the convergence criterion ϵ (introduced below). Strictly speaking, the majorization requirements are violated. However, by choosing δ small enough, the monotone convergence of the sequence of $L_{\text{SVM}}(\mathbf{w})$ will be no problem. The same holds for a_1 if $\overline{q} = 1$.

A.2 Majorizing the Quadratic Hinge Error

The majorizing algorithm for the SVM with the quadratic hinge function is developed along the same lines as for the absolute hinge function. However, because of its structure, each iteration boils down to a matrix multiplication of a fixed $m \times n$ matrix with an $n \times 1$ vector that changes over the iterations. Therefore, the computation of the update is of order $O(nm)$ which is more efficient than the majorizing algorithm for the absolute hinge error.

To majorize the term $f_{-1}(q) = \max(0, q+1)^2$ is relatively easy. For $\overline{q} > -1$, $f_{-1}(q)$ coincides with $(q+1)^2$. Therefore, if $\overline{q} > -1$, $(q+1)^2$ can be used to majorize $\max(0, q+1)^2$. Note that $(q+1)^2 \geq 0$ so that $(q+1)^2$ also satisfies the majorizing requirements for $q < 1$. For the case $\overline{q} \leq -1$, we want a majorizing function that has the same curvature as $(q+1)^2$ but touches at $\overline{q}$, which is obtained by the majorizing function $(q+1-(\overline{q}+1))^2 = (q-\overline{q})^2$. Therefore, the majorizing function $g_{-1} = a_{-1}q^2 - 2b_{-1}q + c_{-1}$ has coefficients

$$\begin{aligned} a_{-1} &= 1, && (8.28)\\ b_{-1} &= \begin{cases} \overline{q} & \text{if } \overline{q} \leq -1 \\ -1 & \text{if } \overline{q} > -1 \end{cases}, && (8.29)\\ c_{-1} &= \begin{cases} 1 - 2(\overline{q}+1) + (\overline{q}+1)^2 & \text{if } \overline{q} \leq -1 \\ 1 & \text{if } \overline{q} > -1 \end{cases}. && (8.30) \end{aligned}$$

Similar reasoning can be held for $f_1(q) = \max(0, 1-q)^2$ which has majorizing function $g_1 = a_1q^2 - 2b_1q + c_1$ and coefficients

$$\begin{aligned} a_1 &= 1, && (8.31)\\ b_1 &= \begin{cases} 1 & \text{if } \overline{q} \leq 1 \\ \overline{q} & \text{if } \overline{q} > 1 \end{cases}, && (8.32)\\ c_1 &= \begin{cases} 1 & \text{if } \overline{q} \leq 1 \\ 1 - 2(1-\overline{q}) + (1-\overline{q})^2 & \text{if } \overline{q} > 1 \end{cases}. && (8.33) \end{aligned}$$

Again, a_i, b_i, and c_i are defined as in (8.15), (8.16), and (8.17), except that δ in (8.15) can be set to 0, so that $a_i = 1 = a$ for all i.

A.3 Majorizing the Huber Hinge Error

The majorizing algorithm of the Huber hinge error function shares a similar efficiency as for the quadratic hinge: the coefficients a_1 and a_{-1} are the same for all i, so that again an update boils down to a matrix multiplication of a matrix of order $m \times n$ with an $n \times 1$ vector.

To majorize $h_{-1}(q)$ we use the fact that the second derivative of $h_{-1}(q)$ is bounded. For $q \geq k$, $h_{-1}(q)$ is linear with first derivative $h'_{-1}(q) = 1$, so that its second derivative $h''_{-1}(q) = 0$. For $q \leq -1$, $h_{-1}(q) = 0$, so that here too $h''_{-1}(q) = 0$. Therefore, $h''_{-1}(q) > 0$ only exists for $-1 < q < k$, where $h''_{-1}(q) = 1$. Therefore, for $-1 < q < k$, the quadratic majorizing function is equal to $h_{-1}(q)$, for $q \leq -1$ and $q \geq k$, a quadratic majorizing function with the same second derivative of $(1/2)(k+1)^{-1}$ is produced that touches at the current estimate $\overline{q}$. Let the majorizing function $g_{-1} = a_{-1}q^2 - 2b_{-1}q + c_{-1}$ has coefficients

$$a_{-1} = (1/2)(k+1)^{-1}, \tag{8.34}$$

$$b_{-1} = \begin{cases} a_{-1}\overline{q} & \text{if } \overline{q} \leq -1 \\ -a_{-1} & \text{if } -1 < \overline{q} < k \\ a_{-1}\overline{q} - 1/2 & \text{if } \overline{q} \geq k \end{cases}, \tag{8.35}$$

$$c_{-1} = \begin{cases} a_{-1}\overline{q}^2 & \text{if } \overline{q} \leq -1 \\ a_{-1} & \text{if } -1 < \overline{q} < k \\ 1 - (k+1)/2 + a_{-1}\overline{q}^2 & \text{if } \overline{q} \geq -k \end{cases}. \tag{8.36}$$

It may be verified for any $\overline{q}$ from the three intervals that $h_{-1}(\overline{q}) = g_{-1}(\overline{q})$ and $h'_{-1}(\overline{q}) = g'_{-1}(\overline{q})$ hold. In addition, $g''_{-1}(q) = (1/2)(k+1)^{-1} \geq h''_{-1}(q)$ for all q (as long as $k > -1$) so that the second derivative $d''_{-1}(q)$ of the difference function $d_{-1}(q) = g_{-1}(q) - h_{-1}(q)$ equals $g''_{-1}(q) - h''_{-1}(q) \geq 0$ indicating that $d_{-1}(q)$ is convex. As $g_{-1}(q)$ touches $h_{-1}(q)$ at $\overline{q}$, $d_{-1}(\overline{q}) = 0$, so that, combined with convexity of $d_{-1}(q)$ the inequality $d_{-1}(q) \geq 0$ must hold implying the majorizing inequality $h_{-1}(q) \leq g_{-1}(q)$ for all q with equality at $\overline{q}$.

For $h_1(q)$ similar reasoning can be held. Let the majorizing function $g_1 = a_{-1}q^2 - 2b_{-1}q + c_{-1}$ has coefficients

$$a_1 = (1/2)(k+1)^{-1}, \tag{8.37}$$

$$b_1 = \begin{cases} 1/2 + a_1\overline{q} & \text{if } \overline{q} \leq -k \\ a_1 & \text{if } -k < \overline{q} < 1 \\ a_1\overline{q} & \text{if } \overline{q} \geq 1 \end{cases}, \tag{8.38}$$

$$c_1 = \begin{cases} 1 - (k+1)/2 + a_{-1}\overline{q}^2 & \text{if } \overline{q} \leq -k \\ a_1 & \text{if } -k < \overline{q} < 1 \\ a_1\overline{q}^2 & \text{if } \overline{q} \geq 1 \end{cases}. \tag{8.39}$$

Note that a_{-1} and a_1 are exactly the same and both independent of $\overline{q}$. Therefore, the curvature of the majorizing functions for all Huber hinge errors is the same. This property is exploited in the simple update derived from (8.22).

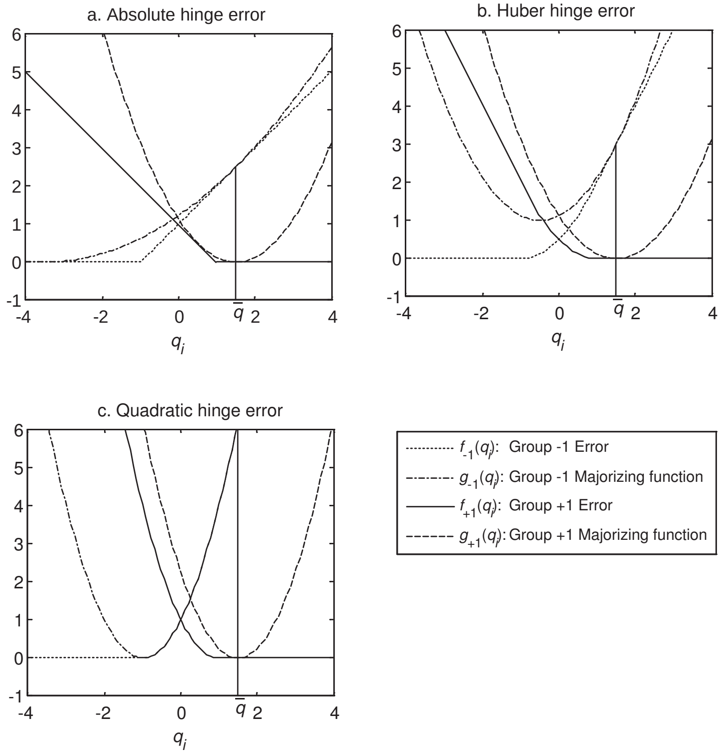


Figure 8.4: Quadratic majorization functions for (a) the absolute hinge error, (b) the Huber hinge error, and (c) the quadratic hinge error. The supporting point is $\overline{q} = 1.5$ both for the Group -1 and 1 error so that the majorizing functions touch at $q = \overline{q} = 1.5$.

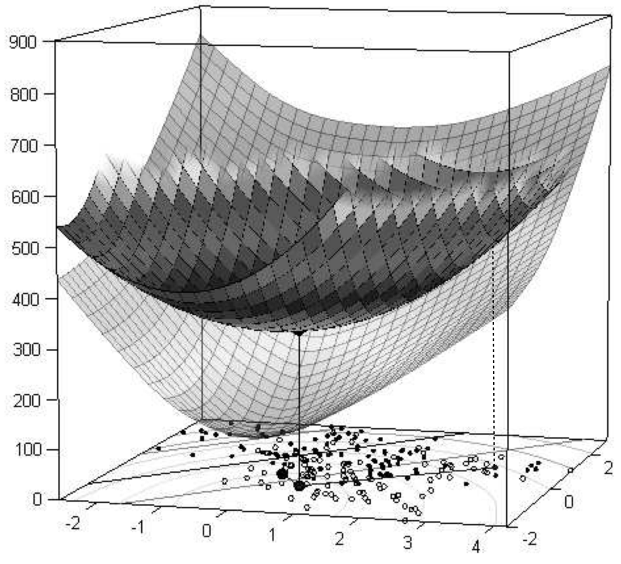


Figure 8.5: Illustrative example of the iterative majorization algorithm for SVMs in action where c is fixed and w_1 and w_2 are being optimized. The majorization function touches $L_{\mathrm{SVM}}(c, \mathbf{w})$ at the previous estimates of $\mathbf{w}$ (the dotted line) and a solid line is lowered at the minimum of the majorizing function showing a decrease in $L_{\mathrm{SVM}}(c, \mathbf{w})$ as well.

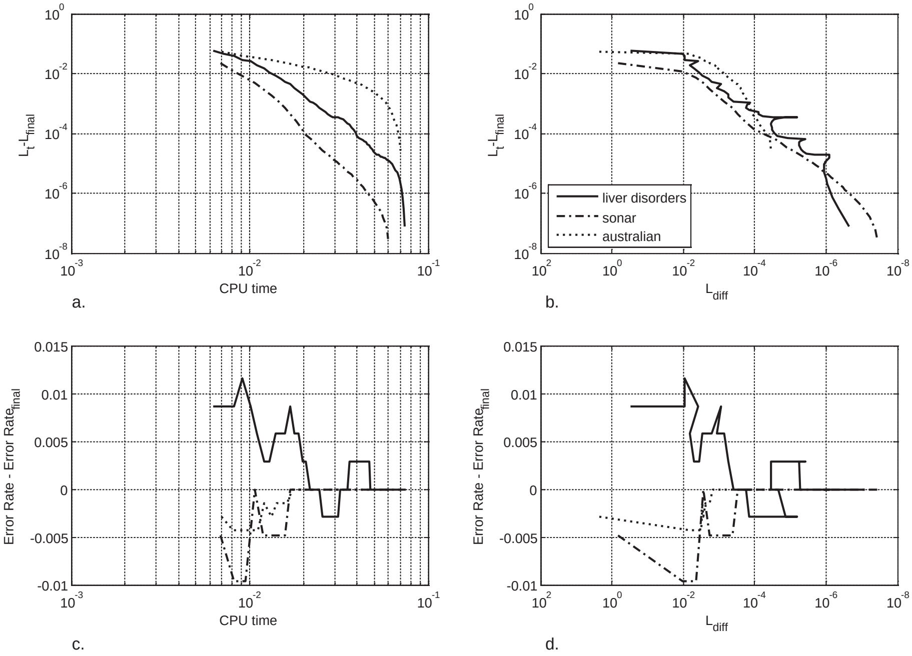


Figure 8.6: The evolution of several statistics (see text for details) of three datasets: Australian (dotted lines), Sonar (dash-dot lines), and Liver Disorders (scaled, solid lines). Values of λ's are fixed at optimal levels for each dataset. Loss function used: absolute hinge.

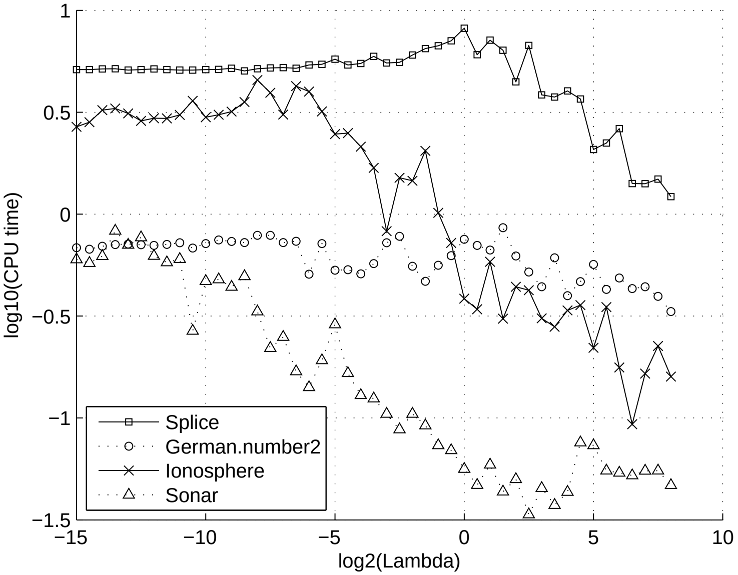


Figure 8.7: The effect of changing λ on CPU time taken by SVM-Maj to converge. Loss function used: absolute hinge.

Table 8.4: Comparisons between SVM solvers: time to convergence in CPU sec. and objective values. The values of $\lambda = 2^p$'s are fixed at levels close to the optimal ones of Table 8.3.

		Time to converge, in CPU sec.							L_{SVM}						
		SVMMaj			Lib-		SVM-	SVM-	SVMMaj			Lib-		SVM-	SVM-
Dataset	p	10^{-4}	10^{-5}	10^{-6}	SVM	BSVM	Light	Perf	10^{-4}	10^{-5}	10^{-6}	SVM	BSVM	Light	Perf
Australian	0	0.07	0.08	0.11	6395.27	0.30	76.80	–	202.67	202.69	202.66	220.81	203.17	202.07	–
Breast_cancer_w	6	0.03	0.06	0.10	0.06	0.18	0.09	0.89	58.21	58.05	58.02	58.03	205.03	58.03	205.03
Heart_statlog	0	0.02	0.02	0.02	0.08	0.14	0.10	1.43	91.50	91.49	91.49	91.48	91.52	91.48	91.52
Hepatitis	0	0.01	0.02	0.03	0.05	0.09	0.06	62.14	46.31	46.29	46.28	46.28	48.43	46.28	48.43
Sonar	0	0.02	0.04	0.05	0.13	0.30	0.15	31.08	114.54	114.51	114.51	114.51	116.83	114.51	116.83
Voting	-5	0.06	0.11	0.11	0.09	0.13	0.12	5.15	26.46	25.76	25.76	25.76	25.76	25.76	25.76
Liver-disorders	3	0.04	0.06	0.06	0.49	1.25	–	–	248.56	248.43	248.42	248.42	253.03	–	–
Liver-disorders2	-6	0.03	0.04	0.07	0.11	0.15	–	–	249.67	249.62	249.59	249.59	249.63	–	–
Diabetes	1	0.06	0.09	0.15	33.05	1.88	10.30	407.81	396.73	396.60	396.57	396.81	444.18	396.57	444.18
Diabetes2	-2	0.05	0.07	0.12	0.14	0.23	0.26	77.35	399.81	399.69	399.66	399.66	399.67	399.66	399.67
Ionosphere	-5	0.12	0.15	0.38	0.24	0.67	0.88	233.19	55.63	55.51	55.33	55.32	56.56	55.32	56.56
German.number	0	0.13	0.35	0.55	15.21	0.54	1.87	87.73	522.37	521.94	521.89	521.88	524.33	521.88	524.33
German.number2	3	0.14	0.22	0.38	0.27	0.39	0.23	82.49	539.51	539.44	539.40	539.39	539.63	539.39	539.63
Splice	5	0.55	1.01	1.55	0.47	0.87	0.33	11.50	427.37	427.21	427.19	427.18	433.68	427.18	433.68

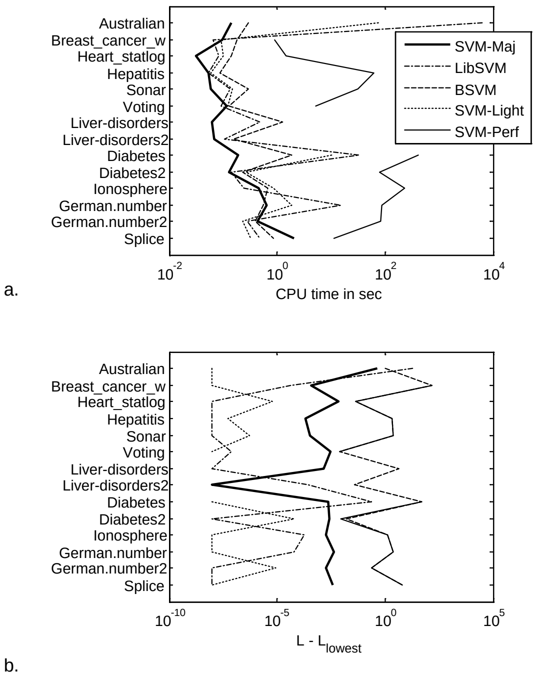


Figure 8.8: Difference in performance of SVM algorithms with absolute hinge and SVM-Maj using $\epsilon = 3\times 10^-7$. Panel a. shows the CPU time used in seconds and Panel b. shows the difference of L and the lowest L amongst the methods.

Epilogue

A number of topics have not been explicitly addressed in this thesis or have just been mentioned in passing. This has been done with a view to keeping the subject matter more focused and compendious. Future research could, for instance, concentrate on further developments of the proposed techniques, finding more links with established methods, applications to more areas and real-life data sets, etc. On the other hand, a number of conclusions can be made from this thesis. One of the main ones is that penalization is a powerful means to improve model performance, where the model under consideration is usually considered to be unbiased or, at least, very flexible to changes in the data. This result is by no means new, but since it is not really universally acknowledged, there may still appear more pieces of academic research (like this one) before it becomes so. The penalization aspect is not new, but its application to instance-based classification methods is. This has proved to be a pretty happy "marriage", at least given the initial positive results. In this case, the act of penalization materializes as the change in the *soft distance* to (local) sets, where the *soft* size of the sets varies. So, the bigger the size of the set, which generally translates into a smaller soft distance to the set from a test point of interest, the greater the function-smoothing effect of penalization. An additional contribution here is the introduction of kernels into the proposed instance-based penalization techniques. This idea has been borrowed from the Support Vector Machines literature, but there is no *a priori* reason as to why it should be possible to apply it in the instance-based case. In the end, it turns out that the introduction of kernels into the picture improves the model performance dramatically.

Leaving aside the instance-based aspect for moment, another major conclusion is that established kernel-based penalization methods can be allied to both readily-available application tasks and to enhance the performance of existing non-penalization estimation methods. For example, Support Vector Regressions have successfully been applied to handle a Financial investment strategy. Support Vector Machines have been shown to perform quite well on a Marketing data set. A way to improve the interpretability of the individual effects of the inputs on the output variable has also been proposed here. Finally, Support Vector Regressions have been implanted inside a state-of-art econometric model for predicting market shares, which has improved the results of this model tremendously.

Last but not least, the proposed majorization algorithm opens the door for non-experts who would like to solve the Support Vector Machine optimization problem in a fast and easy-to-follow way. Potentially, this algorithm can be further enhanced to handle nonlinear cases and similar optimization problems. Ideally, it might be possible to generalize the algorithms to handle a vast number of penalization techniques using one common approach. This could prove quite handy and popularize research into the area on the side of both experts and, especially, non-experts.

A number of topics have been left out of the main discussion line of the thesis. The fact that some topics are missing is not so crucial, as it is not meant to be used as a reference guide, but nevertheless I would like to comment on the next-in-line points of interest. A quite important venue that has not been followed is pursuing explicitly the forces that affect the interplay between bias and variance, or if one likes, error and complexity, for instance-based classification methods in general, and in particular for the instance-based methods in this thesis. There is also no discussion on the difference between inductive bias, or model bias, and estimation bias. Arguably, these two combine forces to give rise to the “bias” as used in the sense when we speak about the bias-variance trade-off. Next, possible extensions to unsupervised tasks have also been left out. Also, throughout the thesis the cross-validation technique has been used for model selection, or parameter-tuning, without a deep explanation as to why this approach deserves to be so ubiquitous. Other approaches to model selection exist, some of which are based, for example, on (assumed) knowledge of the underlying noise in the output variable. Prominent such approaches are the Bayesian approach (or, Bayesian information criterion), Akaike information criterion, the Vapnik-Chervonenkis dimension, the parametric and nonparametric bootstrap method and others. For some of these approaches it is obvious how to apply in our framework, for others it is not. Actually, cross-validation can be thought of as a simplified bootstrap method for model selection.

In addition, there is none or almost none reference to Bayesian Econometrics or maximum-likelihood estimation. The latter is particularly applicable in case we would like to adjust our model as much as possible to the underlying noise of the dependent variable. However, if we have *a priori* a loss function in mind, such as the squared-error loss function, then we would ultimately like to minimize this loss rather than the loss implied by the distribution of the noise. Interestingly, sometimes it is the penalized maximum likelihood estimation that produces better out-of-sample performance. For example, the optimal parameter in Ridge Regression estimation on finite data with a known Gaussian noise of the dependent variable and inherently linear input-output relation is *not* zero. The case where it is zero corresponds to OLS and maximum-likelihood estimation. The non-zero parameter value case corresponds to Ridge Regression and penalized maximum-likelihood estimation. It is hard to believe at first sight that the maximum-likelihood estimation is not optimal in the case of Gaussian noise and linear relation between the explained and explanatory variables. The reason as to why this happens is that we have a *finite* data set from which to estimate our (linear) model. As they say in Economics, we are in a "second-best scenario". That is, if we have an infinite data set here, then we would use maximum-likelihood estimation, as it is the optimal estimation method. However, in case we do not have an infinite data set, then we cannot any more expect that this method would still be optimal. Therefore, we find ourselves in a situation, where a method that is not optimal in the ideal setting turns out to be more preferable than the method that is optimal in the *ideal* case. To give a simple example from Economics, imagine a market for a product in which there is perfect competition, which is an ideal case scenario. The competition drives the price down to a level at which each firm sells the product at a price at which it does not make any profits or losses. Setting such a price is the best strategy for any firm. If a firm sets a slightly higher price, then it will loose all of its (potential) clients and incur losses equal, at least, to the cost of production. If we move away from the perfect-competition case, then setting a price such that the production costs can just be covered may *not* be optimal. Say, we have a firm that is a monopolist, that is, it is the only one that produces a certain product. Clearly, it is not optimal for a monopolist to set the price that is optimal in the perfect-competition case! The optimal monopolistic price would definitely be relatively higher. This extensive example comes also to show that it is not only Statistics and Machine Learning that have realized that some kind of regularization, or change of method, is more optimal in non-ideal cases.

The list of lightly-touched topics is by no means finished. Sometimes it is quite important not only to provide a certain prediction value or class label, but also to have a way to say how confident a method is in the prediction it makes. The majority of the methods described in this thesis fail to output confidence intervals for the estimates. This happens as they are nonparametric techniques. However, if some "parametrics" are assumed, like a concrete parametric distribution for the noise of the output variable, then it is possible to construct confidence internals as well. Alternatively, if the noise is unknown, then one could consult the so-called typicalness framework (see Melluish, Saunders, Nouretdinov, & Vovk, 2001) or the version space framework (see Smirnov, Sprinkhuizen-Kuyper, Nalbantov, & Vanderlooy, 2006) for a possible insight on prediction confidence.

There is little mentioning in the thesis of related penalization techniques and links with competing regression/classification methods such as Neural Networks, Decision Trees, Linear and Quadratic Discriminant Analysis, Logistic Regression, Classification and Regression Trees, and many, many others. Ways to reduce the number of inputs like (Kernel) Principal Component Analysis, which is arguably a good idea in some settings, have also not been discussed.

Last but not least, no bounds on the test, or generalization, error have been provided for the proposed techniques. The difficulty to provide such bounds stems from the instance-based nature of these techniques. All of the above raised points demand attention and, I hope, will be the subject of future research.

Summary

This thesis combines in a novel way three research areas, namely instance-based, kernel and penalization methods. These areas are more popular in Machine Learning rather than in Statistics or Econometrics, where they have been gaining momentum in recent years. The combination has lead to new research models based on a solid foundation that exhibit outstanding performance results. The underlying justification for the models comes mainly from penalization, or regularization, and has to do with the so-called bias-variance or fit-complexity trade-off. The idea that such a balance has to be sought is part of the main motivation of the thesis.

Recent years have witnessed a wave of research that gravitates around the so-called penalization aspect of learning, especially in supervised and unsupervised tasks. Accordingly, a number of tools have been proposed or further developed especially in the fields of Statistics, Econometrics, and Machine Learning. Basically, the idea behind penalization is that under non-ideal conditions, that is if we do not have perfect knowledge about the data, or the *population*, at hand, methods that provide best solutions for the ideal case do not perform as good as other methods. These latter methods may underperform in the ideal-case scenario, but outperform in the non-ideal case. Basically, we are in a non-ideal situation when the data we have is *finite*, which invariably happens in practice. Thus, the imperative need for penalization has become evident. What is penalized are the models that are suitable in the ideal-case scenario, sometimes also called the asymptotic case. In any case, if we have an infinite amount of data to begin with, we can (in theory) compute a population parameter of interest as opposed to estimating it using a model.

Simultaneously with the penalization methods, the kernel methods have gained considerable momentum. In this thesis, the term kernel is used to denote a way to efficiently make calculations of data that is mapped into a space, which dimension is higher than the dimension of the original data space. Sometimes the calculations in the higher-dimensional space can not only be carried out more efficiently, but a kernel actually provides the only possible means to deal with the data in this space. On the face of it, kernelization allows for greater model flexibility and consequently for a greater chance of overfitting,

which is not a desirable property. It is well known that functions that fit the training data too well do not perform well on test data in general. However, it is the penalization that plays a crucial role in this respect. The kernel allows for greater flexibility, which effect is immediately stifled, so to say, by the act of penalization. The end effect is one of increased potential flexibility, which is materialized in case there exist complex relations in the data. That is one of the main reasons why the combination of kernels and penalization has produced such powerful results: in general, if there are complex relations in the data, they will be detected, and if the input-output relation is close to linear, then a right level of penalization will make sure that this linear relationship is found. There are many approaches as to how to detect the "right" penalization level, and the one used in the thesis is the popular cross-validation.

The final main research area considered are the instance-based methods. Arguably, instance-based methods have gained popularity due to the intuitive way in which they operate. In classification, for example, the proximity of a test observation, called also a test instance, is computed to other instances within a certain local region, and a final decision is made based on that information. Instance-based methods are known for there flexibility and ability to adjust to changes in the data. This can be viewed as both advantage and a disadvantage, similarly to the situation with the kernels. One disadvantage is that instead of relations in the data, there is danger of modeling mainly the noise in the data (if there is noise). Again, however, penalization comes in handy and shields us from this possibility. In the instance-based context of classification, penalization is proportional to the size of the local region on which the classification of a test observation is based. Actually, a material revelation of penalization can be found in the computed so-called soft distance to local sets. These sets are a point (or points), the convex hull of points, and the set of points classified correctly by all the functions from a given function class. To each of these local sets corresponds a proposed method: Soft Nearest Neighbor, Nearest Convex Hull classifier, and Support Hyperplanes, respectively.

The purpose of this thesis is twofold. First, a number of real life applications have been examined, where the performance of penalized and non-penalized (default) models have been compared. Usually, the penalized models significantly outperform conventional methods. In practice, this happens even if the only obstacle in front of the ideal-case model is the fact that the data set at hand is finite, that is, even if all the other ideal-case conditions are satisfied, such as normality of the noise, independence of the observations, etc. Concretely, established penalization methods such as Support Vector Regression, Ridge Regression and Support Vector Machines have be shown to outperform non-penalized models in tasks from the areas of Finance and Marketing. Chapters 2 to 4 are devoted to this application-driven theme. Chapter 2 is a demonstration of how synergies between an established econometric model, the Market Share Attraction Model, and a kernel-based penalization model, Support Vector Regressions can be exploited. In particular, the econometric-based model is taken as given, and consequently its parameters are estimated in a more robust way. The end result is a tremendous improvement in prediction performance. Chapter 3 is a successful application of Support Vector Regressions to a Financial investment strategy, and Chapter 4 discusses the application of Support Vector Machines to a Marketing classification task. Next to relatively superior results, the latter chapter also reports ways to provide better interpretability of the individual effects of each of the inputs on the output variable.

The second purpose of the thesis is to put forward three new instance-based kernel penalization techniques. These include Soft Nearest Neighbor (discussed in part in Chapter 5), Support Hyperplanes (Chapter 6), Nearest Convex Hull classification (Chapter 7). A common theme in these methods is that they are all instance-based in nature. Next to the potential concomitant advantages and disadvantages, this means that even if they outperform existing methods, they are bound to work quite slowly. It turns out that two (quadratic) optimization problems have to be solved in order to output the predicted value of a single test point, which hinders for the time being the application of these methods to large data sets. Nevertheless, the speed of computing machines is constantly increasing, which is a positive sign in this respect. This means that instance-based approaches may become even more popular in the future. Another positive sign is that the prediction performance of the new techniques seems excellent so far. Arguably, this is mainly due to the combination of the three elements: *kernel*, *penalization*, and *instance-based*. Chapter 5 is the chapter that discusses the three proposed techniques at one place and points out their similarities and differences. Alongside, these techniques are compared to the popular Support Vector Machines classifier, which is not an instance-based method. It turns out that some of the new methods, especially Support Hyperplanes, perform consistently better than the rest. One of the main reasons for this could be hidden in the relatively better balance that it strikes between fit and complexity, or bias and variance. This trade-off has extensively been discussed in the chapter. Further research in this respect could concentrate on deriving so-called test-error, or generalization, bounds, which arguably would point out exactly the magnitude of the bias and variance forces that interweave to provide superior test accuracies. In addition, more links could be established between existing state-of-the-art techniques and the proposed ones in this thesis.

Finally, the last chapter of the thesis, Chapter 8, puts forward a new iterative majorization algorithm to solve the linear Support Vector Machines in the so-called primal optimization formulation. Some advantages of it are that it is rather straightforward to implement, it is fast, and it can accommodate easily several convex error functions, such as the SVM absolute-hinge, the Huber-hinge, and, the quadratic-hinge error.

Nederlandse samenvatting (Summary in Dutch)

De onderwerpen van dit proefschrift vallen zowel binnen het vakgebied econometrie als artificiele intelligentie. Via de centrale thema's statistische regressie en classificatie draagt dit proefschrift bij aan de literatuur doordat er nieuwe worden classificatiemethoden worden voorgesteld en nieuwe toepassingen gegeven worden van bestaande methoden op het gebied van financiering en marketing.

Het grootste gedeelte van deze dissertatie richt zich op de zogenaamde loss-versus-penalty classificatietechnieken. Dergelijke technieken bouwen voort op de aanname dat een model niet te ingewikkeld maar ook niet te simpel zou moeten zijn om goed te kunnen voorspellen. Een belangrijke reden voor het gebruiken van een penalty is dat in de praktijk niet alle mogelijke data punten, maar een eindig aantal beschikbaar zijn. De nieuwe technieken die voorgesteld worden zijn Support Hyperplanes, Nearest Convex Hull classificatie en Soft Nearest Neighbor.

Naast deze nieuwe technieken, bespreekt dit proefschift ook nieuwe toepassingen van enkele standaard loss-versus-penalty methoden. Zo worden Support Vector Machines (SVMs) toegepast voor classificatie op financiele tijdreeksen. In een andere toepassing wordt het marktaandeel van automerken over de tijd voorspeld met Support Vector Regression (SVR) toegepast op het Market Share Attraction model. In de laatste toepassing wordt SVM gebruikt op een classificatieprobleem in marketing.

Een ander deel van het proefschrift richt zich op nieuwe en efficiente oplossingen van SVMs met het zogenaamde majorizatiealgoritme SVM-Maj. Dit algoritme geeft de mogelijkheid om verschillende zogenaamde loss-functies te gebruiken binnen SVM-achtige methoden.

Резюме (Summary in Bulgarian)

Предметът на този научен труд е в сферата на иконометрията и изкуствения интелект и по-конкретно, регресионния и класификационния статистически анализ. Предложени са нови класификационни подходи, редом с приложения на традиционни такива в областта на финансите и маркетинга.

В по-голямата си част, дисертацията фокусира върху развитието на стандартни методи, известни с общото наименование балансионни методи. Тези методи имат в основата си схващането, че модел, който има на разположение ограничен набор от данни не трябва да е нито твърде сложен, нито твърде прост, за да притежава добра предсказателна способност. В дисертацията са предложени три нови балансионни подхода: поддържащи хиперравнини, класификация според най-близката конвексна обвивка, и псевдо най-близък пример.

Заедно с новите подходи, дисертацията предлага и нови приложения на някои традиционни балансионни методи. Конкретно, това са приложения на метода на поддържащите вектори за класификационен и регресионен анализ върху финансово прогнозиране, решаване на така-наречения атракционен модел на пазарния дял, и решаване и интерпретиране на бинарни класификационни задачи в маркетинга.

Освен това, този научен труд предлага един нов, ефективен начин за решаване на оптимизационния проблем присъщ на метода на поддържащите вектори, чрез така-наречения мажоризиращ алгоритъм. Този алгоритъм позволява да се използват множество така-наречени функции на загубата при оптимизирането на по-общи методи, подобни на този на поддържащите вектори.

Bibliography

Abe, M. (1991). A moving ellipsoid method for nonparametric regression and its application to logit diagnostics with scanner data. *Journal of Marketing Research*, *28*, 339–346.

Abe, M. (1995). A nonparametric density estimation method for brand choice using scanner data. *Marketing Science*, *14*, 300–325.

Ahmed, P., Lockwood, L., & Nanda, S. (2002). Multistyle rotation strategies. *Journal of Portfolio Management*, *28*, 17–29.

Amihud, Y., & Mendelson, H. (1986). Asset pricing and the bid-ask spread. *Journal of Financial Economics*, *17*, 223–249.

Arnott, R., Dorian, J., & Macedo, R. (1992). Style management: The missing element in equity portfolios. *Journal of Investing*, *1*, 13–21.

Arnott, R., Rice, D., Kelso, C., Kiscadden, S., & Macedo, R. (1989). Forecasting factor returns: An intriguing possibility. *Journal of Portfolio Management*, *16*, 28–35.

Asness, C., Friedman, J., Krail, R., & Liew, J. (2000). Style timing: Value versus growth. *Journal of Portfolio Management*, *26*, 51–60.

Banz, R. (1981). The relationship between return and market value of common stocks. *Journal of Financial Economics*, *9*, 3–18.

Bauer, R., Derwall, J., & Molenaar, R. (2004). The real-time predictability of the size and value premium in Japan. *Pacific-Basin Finance Journal*, *12*, 503–523.

Bauer, R., & Molenaar, R. (2002). *Is the value premium predictable in real time?* (LIFE working Paper 02-003)

Bennett, K. P., & Bredensteiner, E. J. (2000). Duality and geometry in SVM classifiers. In *ICML '00: Proceedings of the seventeenth international conference on machine learning* (pp. 57–64). San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Bennett, K. P., Wu, S., & Auslender, L. (1999). On support vector decision trees for database marketing. *IEEE International Joint Conference on Neural Networks (IJCNN '99)*, *2*, 904–909.

Bernstein, R. (2001). *Navigate the noise: Investing in the new age of media and hype.* New York: John Wiley and Sons.

Bishop, C. M. (1995). *Neural networks for pattern recognition.* Oxford University Press.

Borg, I., & Groenen, P. J. F. (2005). *Modern multidimensional scaling: Theory and applications (2nd edition).* New York: Springer.

Breiman, L. (1996). Bagging predictors. *Machine Learning*, *24*, 123-–140.

Burbidge, R., & Buxton, B. (2001). An introduction to support vector machines for data mining. In M. Sheppee (Ed.), *Keynote papers, young OR12* (pp. 3–15). University of Nottingham: Operational Research Society.

Burges, C. (1998). A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, *2*, 121–167.

Campbell, J., Lo, A., & MacKinlay, A. (1997). *The Econometrics of Financial Markets.* Princeton University Press.

Cawley, G. C., & Talbot, N. L. C. (2002). Reduced rank kernel ridge regression. *Neural Processing Letters*, *16*(3), 293-302.

Chan, K., & Chen, N.-F. (1991). Structural and return characteristics of small and large firms. *Journal of Finance*, *46*, 1467-1484.

Chan, L., & Lakonishok, J. (2004). Value and growth investing: Review and update. *Financial Analysts Journal*, *60*(1), 71–86.

Chang, C.-C., & Lin, C.-J. (2002). *LIBSVM: a library for support vector machines.*

Chang, C.-C., & Lin, C.-J. (2006). *LIBSVM: a library for support vector machines.* (Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`)

Chang, M.-W., Chen, B.-J., & Lin, C.-J. (2001). *EUNITE network competition: Electricity load forecasting.* (Winner of EUNITE world wide competition on electricity load prediction)

Chu, W., Keerthi, S., & Ong, C. (2003). Bayesian trigonometric support vector classifier. *Neural Computation*, *15*(9), 2227–2254.

Cooper, L. G., & Nakanishi, M. (1988). *Market share analysis: Evaluating competitive marketing effectiveness.* Boston, MA: Kluwer Academic Publishers.

Cooper, M., Gulen, H., & Vassalou, M. (2001). *Investing in size and book-to-market portfolios using information about the macroeconomy: some new trading rules.* (mimeo, Columbia University, New York)

Copeland, M., & Copeland, T. (1999). Market timing: style and size rotation using the VIX. *Financial Analyst Journal*, *55*, 73–81.

Cristianini, N., & Shawe-Taylor, J. (2000). *An introduction to support vector machines.* Cambridge University Press.

Cui, D. (2003). *Product selection agents: A development framework and preliminary application.* Unpublished doctoral dissertation, University of Cincinnati, Business Administration: Marketing, Ohio. (Retrieved April 5, 2005, from http://www.ohiolink.edu/etd/send-pdf.cgi?ucin1054824718)

De Leeuw, J. (1994). Block relaxation algorithms in statistics. In H.-H. Bock, W. Lenski, & M. M. Richter (Eds.), *Information systems and data analysis* (pp. 308–324). Berlin: Springer.

Dimson, E., Nagel, S., & Quigley, G. (2003). Capturing the value premium in the United Kingdom. *Financial Analysts Journal*, *59*(6), 35–45.

Elfakhani, S. (2000). Short positions, size effect, and the liquidity hypothesis: implications for stock performance. *Applied Financial Economics*, *10*, 105–116.

Evgeniou, T., & Pontil, M. (2004). *Optimization conjoint models for consumer heterogeneity.* (INSEAD Working Paper, Serie No. 2004/10/TM)

Fama, E., & French, K. (1992). The cross-section of expected stock returns. *Journal of Finance*, *47*, 427–465.

Fama, E., & French, K. (1993). Common risk factors in the returns on stocks and bonds. *Journal of Financial Economics*, *33*, 3–53.

Fama, E., & French, K. (1998). Value versus growth: the international evidence. *Journal of Finance*, *53*, 1975–1999.

Fok, D., Franses, P., & Paap, R. (2002). Advances in econometrics. In P. Franses & A. Montgomery (Eds.), (Vol. 16, pp. 223–256). Elsevier Science.

Fok, D., & Franses, P. H. (2004). Analyzing the effects of a brand introduction on competitive structure using a market share attraction model. *International Journal of Research in Marketing*, *21 (2)*, 159–177.

Franses, P. H., & Paap, R. (2001). *Quantitative models in marketing research.* Cambridge: Cambridge University Press.

Friedman, J. H. (1996). *On bias, variance, 0/1-loss, and the curse of dimensionality* (Technical Report, Deparment of Statistics,Stanford University).

Gestel, T. V., Suykens, J. A. K., Baesens, B., Viaene, S., Vanthienen, J., Dedene, G., Moor, B. D., & Vandewalle, J. (2004). Benchmarking least squares support vector machine classifiers. *Machine Learning*, *54*(1), 5–32.

Gifi, A. (1990). *Nonlinear multivariate analysis.* Chichester: Wiley.

Glantz, S. (1992). *Primer of biostatistics* (Third ed.). McGraw-Hill.

Groenen, P. J. F., Nalbantov, G. I., & Bioch, J. C. (2007). Nonlinear support vector machines through iterative majorization and I-splines. In R.Decker & H.-J. Lenz (Eds.), *Advances in data analysis* (pp. 149–162). Berlin: Springer.

Groenen, P. J. F., Nalbantov, G. I., & Bioch, J. C. (2008). SVM-Maj: a majorization approach to linear support vector machines with different hinge errors. In *Advances in data analysis and classification* (Vol. 2, pp. 17–43). Springer Berlin/Heidelberg.

Hansen, P. R. (2005). A test for superior predictive ability. *Journal of Business & Economic Statistics*, 365–380. (available at `http://ideas.repec.org/a/bes/jnlbes/v23y2005p365-380.html`)

Hastie, T., Tibshirani, R., & Friedman, J. H. (2001). *The elements of statistical learning: Data mining, inference, and prediction.* Springer-Verlag New York, Inc.

Haugen, R., & Baker, N. (1996). Commonality in the determinants of expected stock returns. *Journal of Financial Economics*, *41*, 401–439.

Heiser, W. J. (1995). Convergent computation by iterative majorization: Theory and applications in multidimensional data analysis. In W. J. Krzanowski (Ed.), *Recent advances in descriptive multivariate analysis* (pp. 157–189). Oxford: Oxford University Press.

Hsu, C.-W., & Lin, C.-J. (2006). *BSVM: bound-constrained support vector machines.* (Software available at `http://www.csie.ntu.edu.tw/~cjlin/bsvm/index.html`)

Huber, P. J. (1964). Robust estimation of a location parameter. *Annals of Mathematical Statistics*, *35*, 73–101.

Huber, P. J. (1981). *Robust statistics.* New York: Wiley.

Hunter, D. R., & Lange, K. (2004). A tutorial on MM algorithms. *The American Statistician*, *39*, 30–37.

Jacobs, B., & Levy, K. (1996). High definition style rotation. *Journal of Investing*, *5*, 14–23.

Jegadeesh, N., & Titman, S. (1993). Returns to buying winners and selling losers: Implications for market efficiency. *Journal of Finance*, *48*, 65–91.

Joachims, T. (1999). Making large-scale SVM learning practical. In B. Schölkopf, C. Burges, & A. Smola (Eds.), *Advances in kernel methods - support vector learning.* MIT-Press. (`http://www-ai.cs.uni-dortmund.de/DOKUMENTE/joachims_99a.pdf`)

Joachims, T. (2006). Training linear SVMs in linear time. In *Proceedings of the ACM conference on knowledge discovery and data mining (KDD).* (`http://www.cs.cornell.edu/People/tj/publications/joachims_06a.pdf`)

Kao, D.-L., & Shumaker, R. (1999). Equity style timing. *Financial Analysts Journal*, *55*, 37–48.

Keerthi, S. S., & Lin, C.-J. (2003). Asymptotic behaviors of support vector machines with gaussian kernel. *Neural Computation*, *15*, 1667–1689.

Kiers, H. A. L. (2002). Setting up alternating least squares and iterative majorization algorithms for solving various matrix optimization problems. *Computational Statistics and Data Analysis*, *41*, 157–170.

King, R. D., Feng, C., & Sutherland, A. (1995). Statlog: comparison of classification algorithms on large real-world problems. *Applied Artificial Intelligence*, *9(3)*, 289–334.

Kruskal, J. B. (1965). The analysis of factorial experiments by estimating monotone transformations of the data. *Journal of the Royal Statistical Society, Series B*, *27*, 251–263.

Kwon, K.-Y., & Kish, R. J. (2002). Technical trading strategies and return predictability: NYSE. *Applied Financial Economics*, *12*, 639–653.

Lakonishok, J., Schleifer, A., & Vishny, R. (1994). Contrarian investment, extrapolation and risk. *Journal of Finance*, *49*, 1541–1578.

Lange, K., Hunter, D. R., & Yang, I. (2000). Optimization transfer using surrogate objective functions. *Journal of Computational and Graphical Statistics*, *9*, 1–20.

Lattin, J., Carroll, J., & Green, P. (2003). *Analyzing multivariate data.* Belmont, CA: Duxbury Press.

Levis, M., & Liodakis, M. (1999). The profitability of style rotation strategies in the United Kingdom. *Journal of Portfolio Management*, *25*(1), 73–86.

Liew, J., & Vassalou, M. (2000). Can book-to-market, size and momentum be risk factors that predict economic growth? *Journal of Financial Economics*, *57*, 221–245.

Lim, T. S., Loh, W. Y., & Shih, Y. S. (1995). A comparison of prediction accuracy, complexity, and training time for thirty–three old and new classification algorithms. *Machine Learning*, *40*, 203—228.

Lo, A., & MacKinlay, A. (1990). Data-snooping biases in tests of financial asset pricing models. *Review of Financial Studies*, *3*, 431–468.

Lucas, A., Van Dijk, R., & Kloek, T. (2002). Stock selection, style rotation, and risk. *Journal of Empirical Finance*, *9*, 1–34.

Luxburg, U. von, & Bousquet, O. (2004). Distance–based classification with lipschitz functions. *Journal of Machine Learning Research*, *5*, 669-695.

Maragoudakis, M., Kermanidis, K., Fakotakis, N., & Kokkinakis, G. (2002). Combining bayesian and support vector machines learning to automatically complete syntactical information for HPSG-like formalisms. In *Lrec 2002, 3rd international conference on language resources and evaluation, las palmas, spain* (Vol. 1, pp. 93–100).

Melluish, T., Saunders, C., Nouretdinov, I., & Vovk, V. (2001). *The typicalness framework: a comparison with the Bayesian approach* (Technical Report CLRC-TR-01-05). Royal Holloway University of London.

Miller, K., Li, H., & Cox, D. (2001). *U.S. style rotation model.* (Industry Note, Salomon, Smith Barney)

Mills, T. C., & Jordanov, J. V. (2003). The size effect and the random walk hypothesis: evidence from the London Stock Exchange using Markov Chains. *Applied Financial Economics*, *13*, 807–815.

Mitchell, T. M. (1997). *Machine learning.* McGraw-Hill, New York, NY.

Monteiro, A. (2001). *Interest rate curve estimation: A support vector regression application to finance.* (Working paper)

Müller, K.-R., Mika, S., Rätsch, G., Tsuda, K., & Schölkopf, B. (2001). An introduction to kernel-based learning algorithms. *IEEE Transactions on Neural Networks*, *12*(2), 181–201.

Müller, K.-R., Smola, A., Rätsch, G., Schölkopf, B., Kohlmorgen, J., & Vapnik, V. (1997). Predicting time series with support vector machines. In W. Gerstner, A. Germond, M. Hasler, & J.-D. Nicoud (Eds.), *Proceedings of the international conference on artificial neural networks* (Vol. 1327, pp. 999–1004). Springer.

Mun, J. C., Kish, R. J., & Vasconcello, G. M. (2001). The contrarian investment strategy: additional evidence. *Applied Financial Economics*, *11*, 619–640.

Nalbantov, G., Bauer, R., & Sprinkhuizen-Kuyper, I. (2006). Equity style timing using support vector regressions. *Applied Financial Economics*, *16*(15), 1095–1111.

Nalbantov, G. I., Bioch, J. C., & Groenen, P. J. F. (2006a). Classification with support hyperplanes. In J. Fürnkranz, T. Scheffer, & M. Spiliopoulou (Eds.), *ECML 2006: 17th European Conference on Machine Learning* (pp. 703–710). Springer Berlin / Heidelberg.

Nalbantov, G. I., Bioch, J. C., & Groenen, P. J. F. (2006b). Solving and interpreting binary classification problems in marketing with SVMs. In M. Spiliopoulou, R. Kruse, C. Borgelt, A. Nurnberger, & W. Gaul (Eds.), *From Data and Information Analysis to Knowledge Engineering* (pp. 566–573). Springer.

Nalbantov, G. I., Bioch, J. C., & Groenen, P. J. F. (2006c). *Three notes on binary classification with support hyperplanes* (Econometric Institute Technical Report). Erasmus University Rottedam. (To Appear)

Nalbantov, G. I., Bioch, J. C., & Groenen, P. J. F. (2008). *Soft nearest neighbor* (Econometric Institute Technical Report). Econometric Institute and Erasmus Research Institute of Management. (To Appear)

Nalbantov, G. I., Franses, P. H., Groenen, P. J. F., & Bioch, J. C. (2008). Estimating the Market Share Attraction Model using Support Vector Regressions. *Econometric Reviews.* (Accepted)

Nalbantov, G. I., Groenen, P. J. F., & Bioch, J. C. (2007). *Nearest convex hull classification* (Econometric Institute Technical Report No. EI 2006-50). Econometric Institute and Erasmus Research Institute of Management.

Newman, D. J., Hettich, S., Blake, C. L., & Merz, C. J. (1998). *UCI repository of machine learning databases.* (`http://www.ics.uci.edu/~mlearn/MLRepository.html` University of California, Irvine, Dept. of Information and Computer Sciences)

Pérez-Cruz, F., Afonso-Rodríguez, J., & Giner, J. (2003). Estimating GARCH models using support vector machines. *Quantitative Finance*, *3*, 163–172.

Perlich, C., Provost, F., & Simonoff, J. S. (2003). Tree induction vs. logistic regression: A learning-curve analysis. *Journal of Machine Learning Research*, *4*, 211–255.

Platt, J. (1999). Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In A. Smola, P. Bartlett, B. Schölkopf, D. Schuurmans, editors, *Advances in Large Margin Classifiers.* Cambridge, MA: MIT Press.

Pozdnoukhov, A. (2002). *The analysis of kernel ridge regression learning algorithm* (IDIAP research report IDIAP-RR 02-54).

Rocco S., C., & Moreno, J. (2003). A support vector machine model for currency crises discrimination. In S. Chen & P. Wang (Eds.), *Computational intelligence in economics and finance* (pp. 171–81). Springer-Verlag.

Rosset, S., & Zhu, J. (2007). Piecewise linear regularized solution paths. *The Annals of Statistics*, *35*, 1012–1030.

Rousseeuw, P. J., & Leroy, A. M. (2003). *Robust regression and outlier detection.* New York: Wiley.

Saunders, C., Gammerman, A., & Vovk, V. (1998). Ridge regression learning algorithm in dual variables. In *ICML '98: Proceedings of the fifteenth international conference on machine learning* (pp. 515–521). San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Schölkopf, B., Bartlett, P., Smola, A., & Williamson, R. (1998). Support vector regression with automatic accuracy control. In L. Niklasson, M. Bodén, & T. Ziemke (Eds.), *Proceedings of the international conference on artificial neural networks* (pp. 111–116). Berlin: Springer.

Schölkopf, B., Burges, C., & Vapnik, V. (1995). Extracting support data for a given task. In U. M. Fayyad & R. Uthurusamy (Eds.), *First international conference on knowledge discovery and data mining.* AAAI Press, Menlo Park, CA.

Schölkopf, B., Guyon, I., & Weston, J. (2001). *Statistical learning and kernel methods in bioinformatics.* (Available at http://citeseer.ist.psu.edu/509446.html)

Schwob, R. (2000). Style and style analysis from a practitioner's perspective: What is it and what does it mean for european equity investors. *Journal of Asset Management*, *1*, 39–59.

Shawe-Taylor, J., & Cristianini, N. (2004). *Kernel methods for pattern analysis.* New York, NY, USA: Cambridge University Press.

Smirnov, E., Sprinkhuizen-Kuyper, I., & Nalbantov, G. (2004). Unanimous voting using support vector machines. In *Proceedings of the sixteenth belgian-dutch conference on artificial intelligence (BNAIC-04)* (pp. 43–50).

Smirnov, E., Sprinkhuizen-Kuyper, I., Nalbantov, G., & Vanderlooy, S. (2006). Version space support vector machines. In A. P. G. Brewka, S. Coradeschi & P. Traverso (Eds.), *Proceedings of the 17th european conference on artificial intelligence* (pp. 809–810). IOS Press, Amsterdam, The Netherlands.

Smirnov, E. N., Sprinkhuizen-Kuyper, I. G., Nalbantov, G. I., & Vanderlooy, S. (2006). Version space support vector machines. In G. Brewka, S. Coradeschi, A. Perini, & P. Traverso (Eds.), *Proceedings of the 17th european conference on artificial intelligence (ECAI)* (pp. 809–810). IOS Press.

Smola, A. (1996). *Regression estimation with support vector learning machines.* Master's thesis, Technische Universität München.

Smola, A., & Schölkopf, B. (1998). *A tutorial on support vector regression* (NeuroCOLT2 Technical Report No. NC-TR-98-030). University of London, UK.

Smola, A. J., & Schölkopf, B. (2004). A tutorial on support vector regression. *Statistics and Computing*, *14*(3), 199–222.

Stone, M. (1977). Asymptotics for and against cross-validation. *Biometrika*, *64*, 29–35.

Suykens, J. A. K., Van Gestel, T., De Brabanter, J., De Moor, B., & Vandewalle, J. (2002). *Least squares support vector machines.* Singapore: World Scientific.

Tay, F., & Cao, L. (2001). Application of support vector machines in financial time series forecasting. *Omega: The International Journal of Management Science*, *29 (4)*, 309–317.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, *58*, 267–288.

Tikhonov, A. N. (1963). Solution of incorrectly formulated problems and the regularization method. *Soviet Math Dokl*, *4*, 1035–1038. (English translation of Dokl Akad Nauk SSSR 151, 1963, 501-504)

Van der Kooij, A. J. (2007). *Prediction accuracy and stability of regression with optimal scaling transformations.* Unpublished doctoral dissertation, Leiden University.

Van der Kooij, A. J., Meulman, J. J., & Heiser, W. J. (2006). Local minima in Categorical Multiple Regression. *Computational Statistics and Data Analysis*, *50*, 446–462.

Van Gestel, T., Baesens, B., Garcia, J., & Van Dijcke, P. (2003). A support vector machine approach to credit scoring. *Bank en Financiewezen*, *2*, 73–82.

Van Heerde, H., Leeflang, P., & Wittink, D. (2001). Semiparametric analysis to estimate the deal effect curve. *Journal of Marketing Research*, *38*, 197—215.

Vapnik, V. N. (1982). *Estimation of dependences based on empirical data.* Berlin: Springer.

Vapnik, V. N. (1995). *The nature of statistical learning theory.* Springer-Verlag New York, Inc. (2nd edition, 2000)

Vapnik, V. N. (2000). *The nature of statistical learning theory.* New York: Springer.

Weiss, S., & Kulikowski, C. (1991). *Computer systems that learn.* Morgan Kaufman.

West, P. M., Brockett, P. L., & Golden, L. L. (1997). A comparative analysis of neural, networks and statistical methods for predicting consumer choice. *Marketing Science*, *16*, 370–391.

Witten, I. H., & Frank, E. (2005). *Data mining: Practical machine learning tools and techniques.* Morgan Kaufman, San Francisco. (2nd edition)

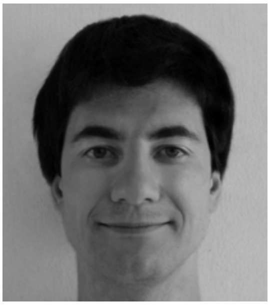
Woodford, B. J. (2001). Comparative analysis of the EFuNN and the support vector machine models for the classification of horticulture data. In N. Kasabov & B. Woodford (Eds.), *Proceedings of the fifth biannual conference on artificial neural networks and expert systems* (pp. 70–75). University of Otago Press.

Young, F. W. (1981). Quantitative analysis of qualitative data. *Psychometrika*, *46*, 357–388.

Young, F. W., De Leeuw, J., & Takane, Y. (1976a). Additive structure in qualitative data: An alternating least squares method with optimal scaling features. *Psychometrika*, *41*, 471–503.

Young, F. W., De Leeuw, J., & Takane, Y. (1976b). Regression with qualitative and quantitative variables: An alternating least squares method with optimal scaling features. *Psychometrika*, *41*, 505–529.

About the Author



Georgi Nalbantov was born in Pernik, Bulgaria, on 17 June 1977. He graduated the English language school there and headed for the capital to study Economics at Sofia University. About two years later, he won a scholarship and went to study International Business in Utrecht, The Netherlands, for about six months. Shortly afterwards, he moved to Maastricht University, enrolled in the program International Economic Studies and graduated with a master degree in 2003. Then he continued his academic carrier at the Econometric Institute of Erasmus University Rotterdam as a PhD candidate up until 2008. During his time at Erasmus University Rotterdam he published articles in the areas of Marketing, Finance, Econometrics and Machine Learning in journals such as Applied Financial Economics, Econometric Reviews and others. In 2006, he was awarded the Chikio Hayashi Award at the International Federation of Classification Societies conference in Slovenia. In addition, he attended and presented papers at various other conferences, such as the European Conference on Machine Learning and the European Conference on Artificial Intelligence, and spent some research time at the statistical department of Stanford University in 2007. Currently, he works on financial and computer science-related projects at Maastricht University.

ERASMUS RESEARCH INSTITUTE OF MANAGEMENT (ERIM)

ERIM PH.D. SERIES
RESEARCH IN MANAGEMENT

ERIM Electronic Series Portal: http://hdl.handle.net/1765/1

Althuizen, N.A.P., Analogical Reasoning as a Decision Support Principle for Weakly Structured Marketing Problems, Promotor: Prof. dr. ir. B. Wierenga, EPS-2006-095-MKT, ISBN: 90-5892-129-8, http://hdl.handle.net/1765/8190

Alvarez, H.L., Distributed Collaborative Learning Communities Enabled by Information Communication Technology, Promotor: Prof. dr. K. Kumar, EPS-2006-080-LIS, ISBN: 90-5892-112-3, http://hdl.handle.net/1765/7830

Appelman, J.H., Governance of Global Interorganizational Tourism Networks: Changing Forms of Co-ordination between the Travel Agency and Aviation Sector, Promotors: Prof. dr. F.M. Go & Prof. dr. B. Nooteboom, EPS-2004-036-MKT, ISBN: 90-5892-060-7, http://hdl.handle.net/1765/1199

Baquero, G, On Hedge Fund Performance, Capital Flows and Investor Psychology, Promotor: Prof. dr. M.J.C.M. Verbeek, EPS-2006-094-F&A, ISBN: 90-5892-131-X, http://hdl.handle.net/1765/8192

Berens, G., Corporate Branding: The Development of Corporate Associations and their Influence on Stakeholder Reactions, Promotor: Prof. dr. C.B.M. van Riel, EPS-2004-039-ORG, ISBN: 90-5892-065-8, http://hdl.handle.net/1765/1273

Berghe, D.A.F. van den, Working Across Borders: Multinational Enterprises and the Internationalization of Employment, Promotors: Prof. dr. R.J.M. van Tulder & Prof. dr. E.J.J. Schenk, EPS-2003-029-ORG, ISBN: 90-5892-05-34, http://hdl.handle.net/1765/1041

Berghman, L.A., Strategic Innovation Capacity: A Mixed Method Study on Deliberate Strategic Learning Mechanisms, Promotor: Prof. dr. P. Mattyssens, EPS-2006-087-MKT, ISBN: 90-5892-120-4, http://hdl.handle.net/1765/7991

Bijman, W.J.J., Essays on Agricultural Co-operatives: Governance Structure in Fruit and Vegetable Chains, Promotor: Prof. dr. G.W.J. Hendrikse, EPS-2002-015-ORG, ISBN: 90-5892-024-0, http://hdl.handle.net/1765/867

Bispo, A., Labour Market Segmentation: An investigation into the Dutch hospitality industry, Promotors: Prof. dr. G.H.M. Evers & Prof. dr. A.R. Thurik, EPS-2007-108-ORG, ISBN: 90-5892-136-9, http://hdl.handle.net/1765/10283

Blindenbach-Driessen, F., Innovation Management in Project-Based Firms, Promotor: Prof. dr. S.L. van de Velde, EPS-2006-082-LIS, ISBN: 90-5892-110-7, http://hdl.handle.net/1765/7828

Boer, C.A., Distributed Simulation in Industry, Promotors: Prof. dr. A. de Bruin & Prof. dr. ir. A. Verbraeck, EPS-2005-065-LIS, ISBN: 90-5892-093-3, http://hdl.handle.net/1765/6925

Boer, N.I., Knowledge Sharing within Organizations: A situated and Relational Perspective, Promotor: Prof. dr. K. Kumar, EPS-2005-060-LIS, ISBN: 90-5892-086-0, http://hdl.handle.net/1765/6770

Boer-Sorbán, K., Agent-Based Simulation of Financial Markets: A modular, Continuous-Time Approach, Promotor: Prof. dr. A. de Bruin, EPS-2008-119-LIS, ISBN: 90-5892-155-0, http://hdl.handle.net/1765/10870

Boon, C.T., HRM and Fit: Survival of the Fittest!?, Promotors: Prof. dr. J. Paauwe & Prof. dr. D.N. den Hartog, EPS-2008-129-ORG, ISBN: 978-90-5892-162-8, http://hdl.handle.net/1765/1

Brito, M.P. de, Managing Reverse Logistics or Reversing Logistics Management? Promotors: Prof. dr. ir. R. Dekker & Prof. dr. M. B. M. de Koster, EPS-2004-035-LIS, ISBN: 90-5892-058-5, http://hdl.handle.net/1765/1132

Brohm, R., Polycentric Order in Organizations: A Dialogue between Michael Polanyi and IT-Consultants on Knowledge, Morality, and Organization, Promotors: Prof. dr. G. W. J. Hendrikse & Prof. dr. H. K. Letiche, EPS-2005-063-ORG, ISBN: 90-5892-095-X, http://hdl.handle.net/1765/6911

Brumme, W.-H., Manufacturing Capability Switching in the High-Tech Electronics Technology Life Cycle, Promotors: Prof. dr. ir. J.A.E.E. van Nunen & Prof. dr. ir. L.N. Van Wassenhove, EPS-2008-126-LIS, ISBN: 978-90-5892-150-5, http://hdl.handle.net/1765/1

Campbell, R.A.J., Rethinking Risk in International Financial Markets, Promotor: Prof. dr. C.G. Koedijk, EPS-2001-005-F&A, ISBN: 90-5892-008-9, http://hdl.handle.net/1765/306

Chen, H., Individual Mobile Communication Services and Tariffs, Promotor: Prof. dr. L.F.J.M. Pau, EPS-2008-123-LIS, ISBN: 90-5892-158-1, http://hdl.handle.net/1765/11141

Chen, Y., Labour Flexibility in China's Companies: An Empirical Study, Promotors: Prof. dr. A. Buitendam & Prof. dr. B. Krug, EPS-2001-006-ORG, ISBN: 90-5892-012-7, http://hdl.handle.net/1765/307

Damen, F.J.A., Taking the Lead: The Role of Affect in Leadership Effectiveness, Promotor: Prof. dr. D.L. van Knippenberg, EPS-2007-107-ORG, http://hdl.handle.net/1765/10282

Daniševská, P., Empirical Studies on Financial Intermediation and Corporate Policies, Promotor: Prof. dr. C.G. Koedijk, EPS-2004-044-F&A, ISBN: 90-5892-070-4, http://hdl.handle.net/1765/1518

Delporte-Vermeiren, D.J.E., Improving the Flexibility and Profitability of ICT-enabled Business Networks: An Assessment Method and Tool, Promotors: Prof. mr. dr. P.H.M. Vervest & Prof. dr. ir. H.W.G.M. van Heck, EPS-2003-020-LIS, ISBN: 90-5892-040-2, http://hdl.handle.net/1765/359

Derwall, J.M.M., The Economic Virtues of SRI and CSR, Promotor: Prof. dr. C.G. Koedijk, EPS-2007-101-F&A, ISBN: 90-5892-132-8, http://hdl.handle.net/1765/8986

Dijksterhuis, M., Organizational Dynamics of Cognition and Action in the Changing Dutch and US Banking Industries, Promotors: Prof. dr. ir. F.A.J. van den Bosch & Prof. dr. H.W. Volberda, EPS-2003-026-STR, ISBN: 90-5892-048-8, http://hdl.handle.net/1765/1037

Elstak, M.N., Flipping the Identity Coin: The Comparative Effect of Perceived, Projected and Desired Organizational Identity on Organizational Identification and Desired Behavior, Promotor: Prof. dr. C.B.M. van Riel, EPS-2008-117-ORG, ISBN: 90-5892-148-2, http://hdl.handle.net/1765/10723

Fenema, P.C. van, Coordination and Control of Globally Distributed Software Projects, Promotor: Prof. dr. K. Kumar, EPS-2002-019-LIS, ISBN: 90-5892-030-5, http://hdl.handle.net/1765/360

Fleischmann, M., Quantitative Models for Reverse Logistics, Promotors: Prof. dr. ir. J.A.E.E. van Nunen & Prof. dr. ir. R. Dekker, EPS-2000-002-LIS, ISBN: 35-4041-711-7, http://hdl.handle.net/1765/1044

Flier, B., Strategic Renewal of European Financial Incumbents: Coevolution of Environmental Selection, Institutional Effects, and Managerial Intentionality, Promotors: Prof. dr. ir. F.A.J. van den Bosch & Prof. dr. H.W. Volberda, EPS-2003-033-STR, ISBN: 90-5892-055-0, http://hdl.handle.net/1765/1071

Fok, D., Advanced Econometric Marketing Models, Promotor: Prof. dr. Ph.H.B.F. Franses, EPS-2003-027-MKT, ISBN: 90-5892-049-6, http://hdl.handle.net/1765/1035

Ganzaroli, A., Creating Trust between Local and Global Systems, Promotors: Prof. dr. K. Kumar & Prof. dr. R.M. Lee, EPS-2002-018-LIS, ISBN: 90-5892-031-3, http://hdl.handle.net/1765/361

Gilsing, V.A., Exploration, Exploitation and Co-evolution in Innovation Networks, Promotors: Prof. dr. B. Nooteboom & Prof. dr. J.P.M. Groenewegen, EPS-2003-032-ORG, ISBN: 90-5892-054-2, http://hdl.handle.net/1765/1040

Ginkel, W.P. van, The Use of Distributed Information in Decision Making Groups: The Role of Shared Task Representations, Promotor: Prof. dr. D. van Knippenberg, EPS-2007-097-ORG, http://hdl.handle.net/1765/8424

Govers, R., Virtual Tourism Destination Image: Glocal Identities Constructed, Perceived and Experienced, Promotors: Prof. dr. F.M. Go & Prof. dr. K. Kumar, EPS-2005-069-MKT, ISBN: 90-5892-107-7, http://hdl.handle.net/1765/6981

Graaf, G. de, Tractable Morality: Customer Discourses of Bankers, Veterinarians and Charity Workers, Promotors: Prof. dr. F. Leijnse & Prof. dr. T. van Willigenburg, EPS-2003-031-ORG, ISBN: 90-5892-051-8, http://hdl.handle.net/1765/1038

Groot, E.A. de, Essays on Economic Cycles, Promotors: Prof. dr. Ph.H.B.F. Franses & Prof. dr. H.R. Commandeur, EPS-2006-091-MKT, ISBN: 90-5892-123-9, http://hdl.handle.net/1765/8216

Gutkowska, A.B., Essays on the Dynamic Portfolio Choice, Promotor: Prof. dr. A.C.F. Vorst, EPS-2006-085-F&A, ISBN: 90-5892-118-2, http://hdl.handle.net/1765/7994

Hagemeijer, R.E., The Unmasking of the Other, Promotors: Prof. dr. S.J. Magala & Prof. dr. H.K. Letiche, EPS-2005-068-ORG, ISBN: 90-5892-097-6, http://hdl.handle.net/1765/6963

Halderen, M.D. van, Organizational Identity Expressiveness and Perception Management: Principles for Expressing the Organizational Identity in Order to Manage the Perceptions and Behavioral Reactions of External Stakeholders, Promotor: Prof. dr. S.B.M. van Riel, EPS-2008-122-ORG, ISBN: 90-5892-153-6, http://hdl.handle.net/1765/10872

Hartigh, E. den, Increasing Returns and Firm Performance: An Empirical Study, Promotor: Prof. dr. H.R. Commandeur, EPS-2005-067-STR, ISBN: 90-5892-098-4, http://hdl.handle.net/1765/6939

Hermans. J.M., ICT in Information Services; Use and Deployment of the Dutch Securities Trade, 1860-1970, Promotor: Prof. dr. drs. F.H.A. Janszen, EPS-2004-046-ORG, ISBN 90-5892-072-0, http://hdl.handle.net/1765/1793

Heugens, P.P.M.A.R., Strategic Issues Management: Implications for Corporate Performance, Promotors: Prof. dr. ir. F.A.J. van den Bosch & Prof. dr. C.B.M. van Riel, EPS-2001-007-STR, ISBN: 90-5892-009-7, http://hdl.handle.net/1765/358

Heuvel, W. van den, The Economic Lot-Sizing Problem: New Results and Extensions, Promotor: Prof. dr. A.P.L. Wagelmans, EPS-2006-093-LIS, ISBN: 90-5892-124-7, http://hdl.handle.net/1765/1805

Hoedemaekers, C.M.W., Performance, Pinned down: A Lacanian Analysis of Subjectivity at Work, Promotors: Prof. dr. S. Magala & Prof. dr. D.H. den Hartog, EPS-2008-121-ORG, ISBN: 90-5892-156-7, http://hdl.handle.net/1765/10871

Hooghiemstra, R., The Construction of Reality: Cultural Differences in Self-serving Behaviour in Accounting Narratives, Promotors: Prof. dr. L.G. van der Tas RA & Prof. dr. A.Th.H. Pruyn, EPS-2003-025-F&A, ISBN: 90-5892-047-X, http://hdl.handle.net/1765/871

Hu, Y., Essays on the Governance of Agricultural Products: Cooperatives and Contract Farming, Promotors: Prof. dr. G.W.J. Hendrkse & Prof. Dr. B. Krug, EPS-2007-113-ORG, ISBN: 90-5892-145-1, http://hdl.handle.net/1765/10535

Huij, J.J., New Insights into Mutual Funds: Performance and Family Strategies, Promotor: Prof. dr. M.C.J.M. Verbeek, EPS-2007-099-F&A, ISBN: 90-5892-134-4, http://hdl.handle.net/1765/9398

Huurman, C.I., Dealing with Electricity Prices, Promotor: Prof. dr. C.D. Koedijk, EPS-2007-098-F&A, ISBN: 90-5892-130-1, http://hdl.handle.net/1765/9399

Iastrebova, K, Manager's Information Overload: The Impact of Coping Strategies on Decision-Making Performance, Promotor: Prof. dr. H.G. van Dissel, EPS-2006-077-LIS, ISBN: 90-5892-111-5, http://hdl.handle.net/1765/7329

Iwaarden, J.D. van, Changing Quality Controls: The Effects of Increasing Product Variety and Shortening Product Life Cycles, Promotors: Prof. dr. B.G. Dale & Prof. dr. A.R.T. Williams, EPS-2006-084-ORG, ISBN: 90-5892-117-4, http://hdl.handle.net/1765/7992

Jansen, J.J.P., Ambidextrous Organizations, Promotors: Prof. dr. ir. F.A.J. Van den Bosch & Prof. dr. H.W. Volberda, EPS-2005-055-STR, ISBN: 90-5892-081-X, http://hdl.handle.net/1765/6774

Jong, C. de, Dealing with Derivatives: Studies on the Role, Informational Content and Pricing of Financial Derivatives, Promotor: Prof. dr. C.G. Koedijk, EPS-2003-023-F&A, ISBN: 90-5892-043-7, http://hdl.handle.net/1765/1043

Keizer, A.B., The Changing Logic of Japanese Employment Practices: A Firm-Level Analysis of Four Industries, Promotors: Prof. dr. J.A. Stam & Prof. dr. J.P.M. Groenewegen, EPS-2005-057-ORG, ISBN: 90-5892-087-9, http://hdl.handle.net/1765/6667

Kijkuit, R.C., Social Networks in the Front End: The Organizational Life of an Idea, Promotor: Prof. dr. B. Nooteboom, EPS-2007-104-ORG, ISBN: 90-5892-137-6, http://hdl.handle.net/1765/10074

Kippers, J., Empirical Studies on Cash Payments, Promotor: Prof. dr. Ph.H.B.F. Franses, EPS-2004-043-F&A, ISBN: 90-5892-069-0, http://hdl.handle.net/1765/1520

Kole, E., On Crises, Crashes and Comovements, Promotors: Prof. dr. C.G. Koedijk & Prof. dr. M.J.C.M. Verbeek, EPS-2006-083-F&A, ISBN: 90-5892-114-X, http://hdl.handle.net/1765/7829

Kooij-de Bode, J.M., Distributed Information and Group Decision-Making: Effects of Diversity and Affect, Promotor: Prof. dr. D.L. van Knippenberg, EPS-2007-115-ORG, http://hdl.handle.net/1765/10722

Knapp, S., The Econometrics of Maritime Safety: Recommendations to Enhance Safety at Sea, Promotor: Prof. dr. Ph.H.B.F. Franses, EPS-2007-096-ORG, ISBN: 90-5892-127-1, http://hdl.handle.net/1765/7913

Koppius, O.R., Information Architecture and Electronic Market Performance, Promotors: Prof. dr. P.H.M. Vervest & Prof. dr. ir. H.W.G.M. van Heck, EPS-2002-013-LIS, ISBN: 90-5892-023-2, http://hdl.handle.net/1765/921

Kotlarsky, J., Management of Globally Distributed Component-Based Software Development Projects, Promotor: Prof. dr. K. Kumar, EPS-2005-059-LIS, ISBN: 90-5892-088-7, http://hdl.handle.net/1765/6772

Kuilman, J., The Re-Emergence of Foreign Banks in Shanghai: An Ecological Analysis, Promotor: Prof. dr. B. Krug, EPS-2005-066-ORG, ISBN: 90-5892-096-8, http://hdl.handle.net/1765/6926

Langen, P.W. de, The Performance of Seaport Clusters: A Framework to Analyze Cluster Performance and an Application to the Seaport Clusters of Durban, Rotterdam and the Lower Mississippi, Promotors: Prof. dr. B. Nooteboom & Prof. drs. H.W.H. Welters, EPS-2004-034-LIS, ISBN: 90-5892-056-9, http://hdl.handle.net/1765/1133

Le Anh, T., Intelligent Control of Vehicle-Based Internal Transport Systems, Promotors: Prof. dr. M.B.M. de Koster & Prof. dr. ir. R. Dekker, EPS-2005-051-LIS, ISBN: 90-5892-079-8, http://hdl.handle.net/1765/6554

Le-Duc, T., Design and Control of Efficient Order Picking Processes, Promotor: Prof. dr. M.B.M. de Koster, EPS-2005-064-LIS, ISBN: 90-5892-094-1, http://hdl.handle.net/1765/6910

Leeuwen, E.P. van, Recovered-Resource Dependent Industries and the Strategic Renewal of Incumbent Firm: A Multi-Level Study of Recovered

Resource Dependence Management and Strategic Renewal in the European Paper and Board Industry, Promotors: Prof. dr. ir. F.A.J. Van den Bosch & Prof. dr. H.W. Volberda, EPS-2007-109-STR, ISBN: 90-5892-140-6, http://hdl.handle.net/1765/10183 Lentink, R.M., Algorithmic Decision Support for Shunt Planning, Promotors: Prof. dr. L.G. Kroon & Prof. dr. ir. J.A.E.E. van Nunen, EPS-2006-073-LIS, ISBN: 90-5892-104-2, http://hdl.handle.net/1765/7328

Liang, G., New Competition: Foreign Direct Investment and Industrial Development in China, Promotor: Prof. dr. R.J.M. van Tulder, EPS-2004-047-ORG, ISBN: 90-5892-073-9, http://hdl.handle.net/1765/1795

Liere, D.W. van, Network Horizon and the Dynamics of Network Positions: A Multi-Method Multi-Level Longitudinal Study of Interfirm Networks, Promotor: Prof. dr. P.H.M. Vervest, EPS-2007-105-LIS, ISBN: 90-5892-139-0, http://hdl.handle.net/1765/10181

Loef, J., Incongruity between Ads and Consumer Expectations of Advertising, Promotors: Prof. dr. W.F. van Raaij & Prof. dr. G. Antonides, EPS-2002-017-MKT, ISBN: 90-5892-028-3, http://hdl.handle.net/1765/869

Londoño, M. del Pilar, Institutional Arrangements that Affect Free Trade Agreements: Economic Rationality Versus Interest Groups, Promotors: Prof. dr. H.E. Haralambides & Prof. dr. J.F. Francois, EPS-2006-078-LIS, ISBN: 90-5892-108-5, http://hdl.handle.net/1765/7578

Maas, A.A., van der, Strategy Implementation in a Small Island Context: An Integrative Framework, Promotor: Prof. dr. H.G. van Dissel, EPS-2008-127-LIS, ISBN: 978-90-5892-160-4, http://hdl.handle.net/1765/1

Maeseneire, W., de, Essays on Firm Valuation and Value Appropriation, Promotor: Prof. dr. J.T.J. Smit, EPS-2005-053-F&A, ISBN: 90-5892-082-8, http://hdl.handle.net/1765/6768

Mandele, L.M., van der, Leadership and the Inflection Point: A Longitudinal Perspective, Promotors: Prof. dr. H.W. Volberda & Prof. dr. H.R. Commandeur, EPS-2004-042-STR, ISBN: 90-5892-067-4, http://hdl.handle.net/1765/1302

Meer, J.R. van der, Operational Control of Internal Transport, Promotors: Prof. dr. M.B.M. de Koster & Prof. dr. ir. R. Dekker, EPS-2000-001-LIS, ISBN: 90-5892-004-6, http://hdl.handle.net/1765/859

Mentink, A., Essays on Corporate Bonds, Promotor: Prof. dr. A.C.F. Vorst, EPS-2005-070-F&A, ISBN: 90-5892-100-X, http://hdl.handle.net/1765/7121

Meyer, R.J.H., Mapping the Mind of the Strategist: A Quantitative Methodology for Measuring the Strategic Beliefs of Executives, Promotor:

Prof. dr. R.J.M. van Tulder, EPS-2007-106-ORG, ISBN: 978-90-5892-141-3, http://hdl.handle.net/1765/10182

Miltenburg, P.R., Effects of Modular Sourcing on Manufacturing Flexibility in the Automotive Industry: A Study among German OEMs, Promotors: Prof. dr. J. Paauwe & Prof. dr. H.R. Commandeur, EPS-2003-030-ORG, ISBN: 90-5892-052-6, http://hdl.handle.net/1765/1039

Moerman, G.A., Empirical Studies on Asset Pricing and Banking in the Euro Area, Promotor: Prof. dr. C.G. Koedijk, EPS-2005-058-F&A, ISBN: 90-5892-090-9, http://hdl.handle.net/1765/6666

Mol, M.M., Outsourcing, Supplier-relations and Internationalisation: Global Source Strategy as a Chinese Puzzle, Promotor: Prof. dr. R.J.M. van Tulder, EPS-2001-010-ORG, ISBN: 90-5892-014-3, http://hdl.handle.net/1765/355

Mom, T.J.M., Managers' Exploration and Exploitation Activities: The Influence of Organizational Factors and Knowledge Inflows, Promotors: Prof. dr. ir. F.A.J. Van den Bosch & Prof. dr. H.W. Volberda, EPS-2006-079-STR, ISBN: 90-5892-116-6, http://hdl.handle.net/1765

Mulder, A., Government Dilemmas in the Private Provision of Public Goods, Promotor: Prof. dr. R.J.M. van Tulder, EPS-2004-045-ORG, ISBN: 90-5892-071-2, http://hdl.handle.net/1765/1790

Muller, A.R., The Rise of Regionalism: Core Company Strategies Under The Second Wave of Integration, Promotor: Prof. dr. R.J.M. van Tulder, EPS-2004-038-ORG, ISBN: 90-5892-062-3, http://hdl.handle.net/1765/1272

Ning, H., Hierarchical Portfolio Management: Theory and Applications, Promotor: Prof. dr. J. Spronk, EPS-2007-118-F&A, ISBN: 90-5892-152-9, http://hdl.handle.net/1765/10868

Noeverman, J., Management Control Systems, Evaluative Style, and Behaviour: Exploring the Concept and Behavioural Consequences of Evaluative Style, Promotors: Prof. dr. E.G.J. Vosselman & Prof. dr. A.R.T. Williams, EPS-2007-120-F&A, ISBN: 90-5892-151-2, http://hdl.handle.net/1765/10869

Oosterhout, J., van, The Quest for Legitimacy: On Authority and Responsibility in Governance, Promotors: Prof. dr. T. van Willigenburg & Prof.mr. H.R. van Gunsteren, EPS-2002-012-ORG, ISBN: 90-5892-022-4, http://hdl.handle.net/1765/362

Paape, L., Corporate Governance: The Impact on the Role, Position, and Scope of Services of the Internal Audit Function, Promotors: Prof. dr. G.J. van der Pijl & Prof. dr. H. Commandeur, EPS-2007-111-MKT, ISBN: 90-5892-143-7, http://hdl.handle.net/1765/10417

Pak, K., Revenue Management: New Features and Models, Promotor: Prof. dr. ir. R. Dekker, EPS-2005-061-LIS, ISBN: 90-5892-092-5, http://hdl.handle.net/1765/362/6771

Pattikawa, L.H, Innovation in the Pharmaceutical Industry: Evidence from Drug Introduction in the U.S., Promotors: Prof. dr. H.R.Commandeur, EPS-2007-102-MKT, ISBN: 90-5892-135-2, http://hdl.handle.net/1765/9626

Peeters, L.W.P., Cyclic Railway Timetable Optimization, Promotors: Prof. dr. L.G. Kroon & Prof. dr. ir. J.A.E.E. van Nunen, EPS-2003-022-LIS, ISBN: 90-5892-042-9, http://hdl.handle.net/1765/429

Pietersz, R., Pricing Models for Bermudan-style Interest Rate Derivatives, Promotors: Prof. dr. A.A.J. Pelsser & Prof. dr. A.C.F. Vorst, EPS-2005-071-F&A, ISBN: 90-5892-099-2, http://hdl.handle.net/1765/7122

Popova, V., Knowledge Discovery and Monotonicity, Promotor: Prof. dr. A. de Bruin, EPS-2004-037-LIS, ISBN: 90-5892-061-5, http://hdl.handle.net/1765/1201

Pouchkarev, I., Performance Evaluation of Constrained Portfolios, Promotors: Prof. dr. J. Spronk & Dr. W.G.P.M. Hallerbach, EPS-2005-052-F&A, ISBN: 90-5892-083-6, http://hdl.handle.net/1765/6731

Prins, R., Modeling Consumer Adoption and Usage of Value-Added Mobile Services, Promotors: Prof. dr. Ph.H.B.F. Franses & Prof. dr. P.C. Verhoef, EPS-2008-128-MKT, ISBN: 978/90-5892-161-1, http://hdl.handle.net/1765/1

Puvanasvari Ratnasingam, P., Interorganizational Trust in Business to Business E-Commerce, Promotors: Prof. dr. K. Kumar & Prof. dr. H.G. van Dissel, EPS-2001-009-LIS, ISBN: 90-5892-017-8, http://hdl.handle.net/1765/356

Quak, H.J., Sustainability of Urban Freight Transport: Retail Distribution and Local Regulation in Cities, Promotor: Prof. dr.M.B.M. de Koster, EPS-2008-124-LIS, ISBN: 978-90-5892-154-3, http://hdl.handle.net/1765/11990

Rinsum, M. van, Performance Measurement and Managerial Time Orientation, Promotor: Prof. dr. F.G.H. Hartmann, EPS-2006-088-F&A, ISBN: 90-5892-121-2, http://hdl.handle.net/1765/7993

Romero Morales, D., Optimization Problems in Supply Chain Management, Promotors: Prof. dr. ir. J.A.E.E. van Nunen & Dr. H.E. Romeijn, EPS-2000-003-LIS, ISBN: 90-9014078-6, http://hdl.handle.net/1765/865

Roodbergen, K.J., Layout and Routing Methods for Warehouses, Promotors: Prof. dr. M.B.M. de Koster & Prof. dr. ir. J.A.E.E. van Nunen, EPS-2001-004-LIS, ISBN: 90-5892-005-4, http://hdl.handle.net/1765/861

Rook, L., Imitation in Creative Task Performance, Promotor: Prof. dr. D.L. van Knippenberg, EPS-2008-125-ORG, http://hdl.handle.net/1765/11555

Schramade, W.L.J., Corporate Bonds Issuers, Promotor: Prof. dr. A. De Jong, EPS-2006-092-F&A, ISBN: 90-5892-125-5, http://hdl.handle.net/1765/8191

Schweizer, T.S., An Individual Psychology of Novelty-Seeking, Creativity and Innovation, Promotor: Prof. dr. R.J.M. van Tulder, EPS-2004-048-ORG, ISBN: 90-5892-07-71, http://hdl.handle.net/1765/1818

Six, F.E., Trust and Trouble: Building Interpersonal Trust Within Organizations, Promotors: Prof. dr. B. Nooteboom & Prof. dr. A.M. Sorge, EPS-2004-040-ORG, ISBN: 90-5892-064-X, http://hdl.handle.net/1765/1271

Slager, A.M.H., Banking across Borders, Promotors: Prof. dr. R.J.M. van Tulder & Prof. dr. D.M.N. van Wensveen, EPS-2004-041-ORG, ISBN: 90-5892-066-6, http://hdl.handle.net/1765/1301

Sloot, L., Understanding Consumer Reactions to Assortment Unavailability, Promotors: Prof. dr. H.R. Commandeur, Prof. dr. E. Peelen & Prof. dr. P.C. Verhoef, EPS-2006-074-MKT, ISBN: 90-5892-102-6, http://hdl.handle.net/1765/7438

Smit, W., Market Information Sharing in Channel Relationships: Its Nature, Antecedents and Consequences, Promotors: Prof. dr. ir. G.H. van Bruggen & Prof. dr. ir. B. Wierenga, EPS-2006-076-MKT, ISBN: 90-5892-106-9, http://hdl.handle.net/1765/7327

Sonnenberg, M., The Signalling Effect of HRM on Psychological Contracts of Employees: A Multi-level Perspective, Promotor: Prof. dr. J. Paauwe, EPS-2006-086-ORG, ISBN: 90-5892-119-0, http://hdl.handle.net/1765/7995

Speklé, R.F., Beyond Generics: A closer Look at Hybrid and Hierarchical Governance, Promotor: Prof. dr. M.A. van Hoepen RA, EPS-2001-008-F&A, ISBN: 90-5892-011-9, http://hdl.handle.net/1765/357

Teunter, L.H., Analysis of Sales Promotion Effects on Household Purchase Behavior, Promotors: Prof. dr. ir. B. Wierenga & Prof. dr. T. Kloek, EPS-2002-016-MKT, ISBN: 90-5892-029-1, http://hdl.handle.net/1765/868

Tims, B., Empirical Studies on Exchange Rate Puzzles and Volatility, Promotor: Prof. dr. C.G. Koedijk, EPS-2006-089-F&A, ISBN: 90-5892-113-1, http://hdl.handle.net/1765/8066

Tuk, M.A., Empirical Studies on Exchange Rate Puzzles and Volatility, Promotors: Prof. dr.ir. A. Smidts & Prof.dr. D.H.J. Wigboldus, EPS-2008-130-MKT, ISBN: 978-90-5892-164-2, http://hdl.handle.net/1765/1

Valck, K. de, Virtual Communities of Consumption: Networks of Consumer Knowledge and Companionship, Promotors: Prof. dr. ir. G.H. van Bruggen

& Prof. dr. ir. B. Wierenga, EPS-2005-050-MKT, ISBN: 90-5892-078-X, http://hdl.handle.net/1765/6663

Valk, W. van der, Buyer-Seller Interaction Patterns During Ongoing Service Exchange, Promotors: Prof. dr. J.Y.F. Wynstra & Prof. dr. ir. B. Axelsson, EPS-2007-116-MKT, ISBN: 90-5892-146-8, http://hdl.handle.net/1765/10856

Verheul, I., Is There a (Fe)male Approach? Understanding Gender Differences in Entrepreneurship, Prof. dr. A.R. Thurik, EPS-2005-054-ORG, ISBN: 90-5892-080-1, http://hdl.handle.net/1765/2005

Vis, I.F.A., Planning and Control Concepts for Material Handling Systems, Promotors: Prof. dr. M.B.M. de Koster & Prof. dr. ir. R. Dekker, EPS-2002-014-LIS, ISBN: 90-5892-021-6, http://hdl.handle.net/1765/866

Vlaar, P.W.L., Making Sense of Formalization in Interorganizational Relationships: Beyond Coordination and Control, Promotors: Prof. dr. ir. F.A.J. Van den Bosch & Prof. dr. H.W. Volberda, EPS-2006-075-STR, ISBN 90-5892-103-4, http://hdl.handle.net/1765/7326

Vliet, P. van, Downside Risk and Empirical Asset Pricing, Promotor: Prof. dr. G.T. Post, EPS-2004-049-F&A, ISBN: 90-5892-07-55, http://hdl.handle.net/1765/1819

Vlist, P. van der, Synchronizing the Retail Supply Chain, Promotors: Prof. dr. ir. J.A.E.E. van Nunen & Prof. dr. A.G. de Kok, EPS-2007-110-LIS, ISBN: 90-5892-142-0, http://hdl.handle.net/1765/10418

Vries-van Ketel E. de, How Assortment Variety Affects Assortment Attractiveness: A Consumer Perspective, Promotors: Prof. dr. G.H. van Bruggen & Prof. dr. ir. A. Smidts, EPS-2006-072-MKT, ISBN: 90-5892-101-8, http://hdl.handle.net/1765/7193

Vromans, M.J.C.M., Reliability of Railway Systems, Promotors: Prof. dr. L.G. Kroon, Prof. dr. ir. R. Dekker & Prof. dr. ir. J.A.E.E. van Nunen, EPS-2005-062-LIS, ISBN: 90-5892-089-5, http://hdl.handle.net/1765/6773

Vroomen, B.L.K., The Effects of the Internet, Recommendation Quality and Decision Strategies on Consumer Choice, Promotor: Prof. dr. Ph.H.B.F. Franses, EPS-2006-090-MKT, ISBN: 90-5892-122-0, http://hdl.handle.net/1765/8067

Waal, T. de, Processing of Erroneous and Unsafe Data, Promotor: Prof. dr. ir. R. Dekker, EPS-2003-024-LIS, ISBN: 90-5892-045-3, http://hdl.handle.net/1765/870

Watkins Fassler, K., Macroeconomic Crisis and Firm Performance, Promotors: Prof. dr. J. Spronk & Prof. dr. D.J. van Dijk, EPS-2007-103-F&A, ISBN: 90-5892-138-3, http://hdl.handle.net/1765/10065

Wennekers, A.R.M., Entrepreneurship at Country Level: Economic and Non-Economic Determinants, Promotor: Prof. dr. A.R. Thurik, EPS-2006-81-ORG, ISBN: 90-5892-115-8, http://hdl.handle.net/1765/7982

Wielemaker, M.W., Managing Initiatives: A Synthesis of the Conditioning and Knowledge-Creating View, Promotors: Prof. dr. H.W. Volberda & Prof. dr. C.W.F. Baden-Fuller, EPS-2003-28-STR, ISBN: 90-5892-050-X, http://hdl.handle.net/1765/1042

Wijk, R.A.J.L. van, Organizing Knowledge in Internal Networks: A Multilevel Study, Promotor: Prof. dr. ir. F.A.J. van den Bosch, EPS-2003-021-STR, ISBN: 90-5892-039-9, http://hdl.handle.net/1765/347

Zhang, X., Strategizing of Foreign Firms in China: An Institution-based Perspective, Promotor: Prof. dr. B. Krug, EPS-2007-114-ORG, ISBN: 90-5892-147-5, http://hdl.handle.net/1765/10721

Zhu, Z., Essays on China's Tax System, Promotors: Prof. dr. B. Krug & Prof. dr. G.W.J. Hendrikse, EPS-2007-112-ORG, ISBN: 90-5892-144-4, http://hdl.handle.net/1765/10502

Erasmus Research Institute of Management - ERIM

ESSAYS ON SOME RECENT PENALIZATION METHODS WITH APPLICATIONS IN FINANCE AND MARKETING

The subject of this PhD research is within the areas of Econometrics and Artificial Intelligence. More concretely, it deals with the tasks of statistical regression and classification analysis. New classification methods have been proposed, as well as new applications of established ones in the areas of Finance and Marketing.

The bulk of this PhD research centers on extending standard methods that fall under the general term of loss-versus-penalty classification techniques. These techniques build on the premises that a model that uses a finite amount of available data to be trained on should neither be too complex nor too simple in order to possess a good forecasting ability. New proposed classification techniques in this area are Support Hyperplanes, Nearest Convex Hull classification and Soft Nearest Neighbor.

Next to the new techniques, new applications of some standard loss-versus-penalty methods have been put forward. Specifically, these are the application of the so-called Support Vector Machines (SVMs) for classification and regression analysis to financial time series forecasting, solving the Market Share Attraction model and solving and interpreting binary classification tasks in Marketing.

In addition, this research focuses on new efficient solutions to SVMs using the so-called majorization algorithm. This algorithm provides for the possibility to incorporate various so-called loss functions while solving general SVM-like methods.

ERIM

The Erasmus Research Institute of Management (ERIM) is the Research School (Onderzoekschool) in the field of management of the Erasmus University Rotterdam. The founding participants of ERIM are RSM Erasmus University and the Erasmus School of Economics. ERIM was founded in 1999 and is officially accredited by the Royal Netherlands Academy of Arts and Sciences (KNAW). The research undertaken by ERIM is focussed on the management of the firm in its environment, its intra- and interfirm relations, and its business processes in their interdependent connections.

The objective of ERIM is to carry out first rate research in management, and to offer an advanced doctoral programme in Research in Management. Within ERIM, over three hundred senior researchers and PhD candidates are active in the different research programmes. From a variety of academic backgrounds and expertises, the ERIM community is united in striving for excellence and working at the forefront of creating new business knowledge.

ERIM PhD Series
Research in Management

Rotterdam School of Management
Erasmus University
Erasmus School of Economics
P.O. Box 1738, 3000 DR Rotterdam
The Netherlands

Tel.	+31 10 408 11 82
Fax	+31 10 408 96 40
E-mail	info@erim.eur.nl
Internet	www.erim.eur.nl

Design & layout: B&T Ontwerp en advies (www.b-en-t.nl) Print: Haveka (www.haveka.nl)