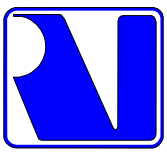
Support Vector Machines - Backgrounds and Practice

Panu Erästö

Academic Dissertation for the Degree of
Licentiate of Philosophy



Rolf Nevanlinna Institute

Helsinki 2001

Contents

1 Introduction

In many fields of science, computer science in particular, automatic learning from examples is a long-standing goal. Also, in recent years the amount of information that has to be processed has exploded and there is a growing need to extract structure from the data instead of just storing it. Moreover, if one is able to capture some dependence in the data this knowledge can be used to predict future situations. To model the structure in the data it is usually important to know the causality of the data generating process. Usually a factor causing a change in some other factor is called a variable and the factor which changes due to the change of the variable is called a response variable. In many cases there are several variables affecting the response variable and in this work we use symbol $\mathbf{x}$ to denote a vector that consists of variables $x_1, x_2, \ldots, x_d$ and the symbol y is used to denote a response variable.

There are two main types of learning from examples. In *regression* the response variable y can have infinitely many values while in *classification* it can have only a finite number of values. Another interpretation is that in the regression case one is learning a function and in the classification case one is learning sets and the number of sets is finite. In this work we deal with classification which is usually simpler type of learning. In particular, we examine a classification algorithm called the *support vector machine*, which has many good practical and theoretical properties. One of the best practical properties is its simplicity which can be very important when dealing with large datasets, even if very powerful computers are available.

History of the support vector machines is ambiguous: it is difficult to name one single paper that introduced the concept. The fundamental theory of linear classifiers, which also includes support vector machines, dates back to the 1930's while the paper by Rosenblatt in 1956 introduced the perceptron of which the support vector machines are one very special case. Linear large margin classifiers which are the simplest form of the support vector machines, have also been invented by several people in various fields of research, and similar ideas are presented in many articles, of which we will mention two: the paper by Vapnik and Lerner [82] in 1963 and the paper by Mangasarian [47] in 1965. Generalization of linear large margin classifiers to the nonlinear case was introduced in [11], which is the first paper that presents the current support vector machine methodology, but due to earlier work it can not be said that it is the essential paper that constitutes the principles of the support vector machines. Theoretical properties of the support vector machines have been studied in conjunction with general research on the the-

oretical properties of machine learning. Theory on which the support vector machines are based has changed over time. The first explanation was given by the Vapnik-Chervonenkis theory[1], followed by the theory of large margin classifiers, but these explanations have still left something to hope for. The latest theory, called data dependent structural risk minimization, and, in particular, the luckiness framework, appears to remedy the lack of theory, and it also combines the VC-theory and some parts of the theory of large margin classifiers within the framework of one single theory. It should be pointed out that in computer science the VC-theory and parts of the theory of large margin classifiers is called PAC-theory (Probably Approximately Correct), but we won't use this term.

The present work is organized as follows. In Section 2 we will give a short introduction to the learning problem in classification. Section 3 is devoted to some general results of the most common approach to solve the learning problem. In the end of this section, some important results of the VC-theory are presented, and one application of these, called structural risk minimization is shortly discussed in Section 4. In Section 5 the theory of large margin classifiers is presented, with results similar to those in the VC-theory. The state-of-the-art theory, which gives the justification for support vector machines, is given in Section 6. In Section 7, we will present the support vector methodology with some of its most common modifications. Generalization of this to nonlinear cases is discussed in Section 8. Conclusions are drawn in Section 9 and the optimization theory needed in support vector machines is given in the Appendix.

[1]Abbreviated as the VC-theory.

2 Learning problem in classification

Suppose we are given a training set $Z = (X, Y)$, where $X = (\mathbf{x}_1, \ldots, \mathbf{x}_n) \in \{\mathbb{R}^d\}^n$ and $Y = (y_1, \ldots, y_n) \in \{-1, 1\}^n$. Suppose also that the training set is drawn i.i.d. according to some unknown (joint) probability measure P defined over pairs $(\mathbf{x}, \mathsf{y}) \in \{\mathbb{R}^d\}^n \times \{-1, 1\}$, where the response variable y characterizes a certain property, or label, of the d-dimensional vector $\mathbf{x}$. In real life situations, getting such a set of examples can be expensive and it is therefore reasonable to assume that n is finite and usually small. In short, our learning problem is: predict the value of y from $\mathbf{x}$ based on the training set and a loss function, where the loss function characterizes the goodness of prediction. More formally: our goal is to find a measurable function $t : \mathbb{R} \to \{-1, 1\}$ based on Z and a loss function $L : \{-1, 1\} \times \{-1, 1\} \to \{0, 1\}$. In our case we define the loss function as

$$L(y, t(\mathbf{x})) = \begin{cases} 0 & : \quad \text{if } y = t(\mathbf{x}) \\ 1 & : \quad \text{if } y \neq t(\mathbf{x}), \end{cases} \tag{2.1}$$

where $t(\mathbf{x})$ is the prediction of y. In the following we will also use the terms *classifier* and *classification rule* for the function t, since we are classifying observations. In the literature, t is also often called a decision function or a decision rule. We use the symbol t_n to denote a classifier which is selected based on a training set of size n and a loss function L so, using the above notations, $t_n = t_n(Z, L)$.

Note that we still have not completely formulated the learning problem but only defined the ingredients of it. Our way of using the ingredients of the learning problem is the following: find a measurable function which minimizes the following risk functional

$$\mathrm{e}(t) = \int L(\mathsf{y}, t(\mathbf{x}))d\,\mathrm{P} = \mathrm{P}\{(\mathbf{x}, y) : t(\mathbf{x}) \neq y\}, \tag{2.2}$$

which quantifies the error probability of t and therefore is a natural measure of performance. It is usually practical (e.g. for computational reasons) to set some limitations on the classifier t. We can, for example, require that t is a linear classifier. In the following, we denote by $\mathcal{C}$ the class of (measurable) classifiers $t : \mathbb{R}^d \to \{-1, 1\}$ which satisfy all the additional requirements. Thus, our problem is to find a classifier t_n such that $t \in \mathcal{C}$, where $\mathcal{C}$ is fixed.

The procedure or algorithm which selects t_n from $\mathcal{C}$ based on the training set Z of size n and loss function L is usually called a *learning machine* $\mathcal{L}$, which can be considered as mapping $\mathcal{L} : \{\mathbb{R}^d, \{-1, 1\}\}^n \to \mathcal{C}$ where $\mathcal{C}$ is the

class of classifiers used. According to some researchers, the learning machine also gives label y' to observed vector $\mathbf{x}'$ (based on Z and L). In this case the learning machine is a mapping $\mathcal{L} : \{\{\mathbb{R}^d \times \{-1, 1\}\}^n, \mathbb{R}^d\} \to \{-1, 1\}$.

3 Empirical risk minimization

As it was assumed in the previous section the distribution P is unknown and we therefore cannot directly minimize (2.2). One approach would be to estimate the unknown probability distribution P, but it is known that estimation of the unknown distribution from a finite data is an ill-posed problem. It is also recommended to avoid making too big a generalization step based on a finite or a small number of examples [19, 80]. The most often used and well-known way of minimizing (2.2) is minimizing the *empirical risk* which avoids solving the more general problem of distribution estimation as an intermediate step. The empirical risk [2] of t based on n observations, denoted by $\hat{\mathrm{e}}_n(t)$, is defined as

$$\hat{\mathrm{e}}_n(t) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}_{\{t(\mathbf{x}_i) \neq y_i\}},$$

which is an empirical estimate of $\mathrm{e}(t) = \mathrm{P}\{t(\mathbf{x}) \neq y\}$. In the literature, the empirical risk is also called the *training error* of the classifier t based on the sample Z. Above we used subscript n to emphasize that the quantity is based on n observations, since we assumed that the training set Z has n observations. The most common approach to solving the learning problem is that, based on the sample Z, the learning machine selects the empirically optimal classifier from $\operatorname{argmin}_{t \in \mathcal{C}} \hat{\mathrm{e}}_n(t)$ the set of minimizers of $\hat{\mathrm{e}}_n(t)$. This approach is called *empirical risk minimization*, abbreviated commonly as ERM. We denote the solution to the argmin rule by $\hat{t}_n$, where the subscript n emphasizes that the selection is based on n observations. To get a good approximation to $\mathrm{e}(\hat{t}_n)$, that is the true error probability of the selected classifier $\hat{t}_n$ based on Z and the ERM principle, we hope that the bias of empirical risk functional is small, that is,

$$|\hat{\mathrm{e}}_n(\hat{t}_n) - \mathrm{e}(\hat{t}_n)|$$

is small. This is important especially in estimating the performance of the classifier $\hat{t}_n$. It is a well-known mathematical fact that $\lim_{n\to\infty} \mathrm{P}\{|\hat{\mathrm{e}}_n(\hat{t}_n) -$

[2] $\hat{\mathrm{e}}_n(t)$ is also often called *empirical risk functional*, but we use the shorthand term *empirical risk*.

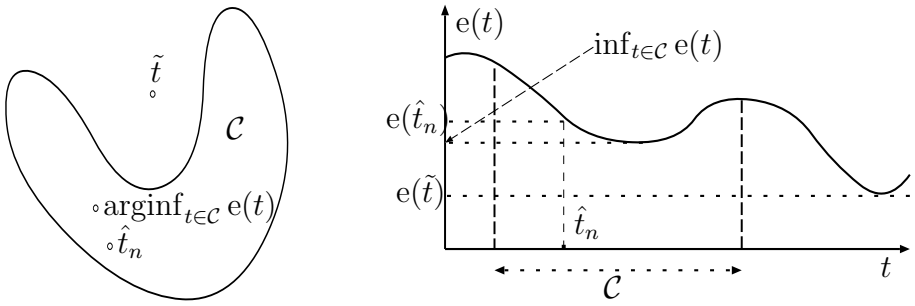


Figure 1: Error decomposition into approximation error and estimation error.

$\mathrm{e}(\hat{t}_n)| > \epsilon\} \to 0$ but since our training set is small we are dealing with non-asymptotic theory and therefore we do not have any guarantee that the bias is small. When we select $\hat{t}_n$ based on the argmin rule above, we also hope that $\mathrm{e}(\hat{t}_n)$ is close to the optimal error probability in the class of approximating functions $\mathcal{C}$, that is,

$$\mathrm{e}(\hat{t}_n) - \inf_{t\in\mathcal{C}} \mathrm{e}(t)$$

is small. This is important especially in classifier selection. However, even if the true risk of $\hat{t}_n$ is close to $\inf_{t\in\mathcal{C}} \mathrm{e}(t)$ the best classifier $\hat{t}_n$ based on Z and the ERM principle can be still far away from the overall best classifier $\tilde{t}$ (not restricted to class $\mathcal{C}$) known as the *Bayes classifier*. We can write the difference between the risk functionals of $\hat{t}$ and $\tilde{t}$ in the following form

$$\mathrm{e}(\hat{t}_n) - \mathrm{e}(\tilde{t}) = \left(\mathrm{e}(\hat{t}_n) - \inf_{t\in\mathcal{C}} \mathrm{e}(t)\right) + \left(\inf_{t\in\mathcal{C}} \mathrm{e}(t) - \mathrm{e}(\tilde{t})\right). \tag{3.1}$$

The former term on the right hand side of (3.1) is called *estimation error* and the latter *approximation error*, correspondingly (cf. Figure 1). We see that the size of $\mathcal{C}$ plays a crucial role in the above equation: if $|\mathcal{C}|$ is small we have not very much hope of making the approximation error small while the estimation error is small with high probability. On the contrary, if $|\mathcal{C}|$ is large, the approximation error is probably small, while the estimation error is probably large [46]. Therefore, when choosing $\mathcal{C}$, there is a tradeoff between a large $|\mathcal{C}|$ and a small $|\mathcal{C}|$. If, for example, $\mathcal{C}$ is the class of all (measurable) classifiers, we can always find a t such that $\hat{\mathrm{e}}_n(t) = 0$, but t is not a good classifier since it performs perfectly on Z but poorly on future samples. Our goal is that the selected classifier learns some general structure from the data, and thus will have a much lower error probability than a random guess, that is, t_n should perform some *generalization*. This is, of course, difficult or even

impossible to achieve if the distribution behaves very badly and therefore when dealing with small samples, we usually make an implicit assumption of a well behaved distribution. Term *overfitting* is commonly used in situations where the selected decision rule t_n does not perform "enough" generalization. In general, it is also natural to require that the error probability of the selected classifier t_n decreases when the number of observations is increased with previously unseen examples.

Next, we will define the consistency of a learning process.

Definition 1 (Consistency of the ERM principle) . *A learning process based on the ERM principle is said to be consistent if the following two sequences converge to the same limit in probability*

$$\begin{aligned} \mathrm{e}(\hat{t}_n) &\xrightarrow[n\to\infty]{\mathrm{P}} \inf_{t\in\mathcal{C}} \mathrm{e}(t) \\ \hat{\mathrm{e}}_n(\hat{t}_n) &\xrightarrow[n\to\infty]{\mathrm{P}} \inf_{t\in\mathcal{C}} \mathrm{e}(t). \end{aligned}$$

In other words, if the learning process is consistent, then both the true risk and the empirical risk of the classifier $\hat{t}_n$ converge towards the same value, the smallest possible value of risk in $\mathcal{C}$, $\inf_{t\in\mathcal{C}} \mathrm{e}(t)$. The following theorem gives bounds on the difference between the true error of a classifier $\hat{t}_n$ and the minimal error probability over classifiers of $\mathcal{C}$, that is $\inf_{t\in\mathcal{C}} \mathrm{e}(t)$, and also a bound on the bias of the risk functional.

Theorem 1 *([22], Lemma 8.2).*

$$\mathrm{e}(\hat{t}_n) - \inf_{t\in\mathcal{C}} \mathrm{e}(t) \leq 2 \sup_{t\in\mathcal{C}} |\hat{\mathrm{e}}_n(t) - \mathrm{e}(t)| \tag{3.2}$$

$$|\hat{\mathrm{e}}_n(\hat{t}_n) - \mathrm{e}(\hat{t}_n)| \leq \sup_{t\in\mathcal{C}} |\hat{\mathrm{e}}_n(t) - \mathrm{e}(t)|. \tag{3.3}$$

Proof ([22]).

$$\begin{aligned} \mathrm{e}(\hat{t}_n) - \inf_{t\in\mathcal{C}} \mathrm{e}(t) &= \mathrm{e}(\hat{t}_n) - \hat{\mathrm{e}}_n(\hat{t}_n) + \hat{\mathrm{e}}_n(\hat{t}_n) - \inf_{t\in\mathcal{C}} \mathrm{e}(t) \\ &\leq \mathrm{e}(\hat{t}_n) - \hat{\mathrm{e}}_n(\hat{t}_n) + \sup_{t\in\mathcal{C}} |\hat{\mathrm{e}}_n(t) - \mathrm{e}(t)| \\ &\leq 2 \sup_{t\in\mathcal{C}} |\hat{\mathrm{e}}_n(t) - \mathrm{e}(t)|, \end{aligned}$$

where the first inequality follows from the fact that $\hat{\mathrm{e}}_n(\hat{t}_n) \leq \hat{\mathrm{e}}_n(t)$, $\forall t \in \mathcal{C}$. The inequality (3.3) is trivially true. □

3.1 Performance bounds in the finite case

From the previous theorem we can see that by bounding $\sup_{t\in\mathcal{C}} |\hat{\mathrm{e}}_n(t) - \mathrm{e}(t)|$, we can control two things simultaneously: the estimation error and the bias of empirical risk functional in the given sample. The next theorem gives a bound on $\sup_{t\in\mathcal{C}} |\hat{\mathrm{e}}_n(t) - \mathrm{e}(t)|$, where $|\mathcal{C}|$ is finite.

Theorem 2 *([22], Theorem 8.3). Assume that the cardinality of $\mathcal{C}$ is finite, and $|\mathcal{C}| \leq N \leq \infty$. Then for all $\epsilon > 0$,*

$$\mathrm{P}\{\sup_{t\in\mathcal{C}} |\hat{\mathrm{e}}_n(t) - \mathrm{e}(t)| > \epsilon\} \leq 2Ne^{-2n\epsilon^2}.$$

Proof ([22]).

$$\mathrm{P}\{\sup_{t\in\mathcal{C}} |\hat{\mathrm{e}}_n(t) - \mathrm{e}(t)| > \epsilon\} \leq \sum_{t\in\mathcal{C}} \mathrm{P}\{|\hat{\mathrm{e}}_n(t) - \mathrm{e}(t)| > \epsilon\} \leq 2Ne^{-2n\epsilon^2},$$

by Hoeffding's inequality ([35]) and the fact that the random variable $n\hat{\mathrm{e}}_n(t)$ is binomially distributed with parameters n and $\mathrm{e}(t)$. □

Let us still consider the case where the cardinality of $\mathcal{C}$ is finite and suppose also that there exists a $t_0 \in \mathcal{C}$ such that $\mathrm{e}(t_0) = 0$, which implies that $\hat{\mathrm{e}}_n(\hat{t}_n) = 0$ with probability one. Then we have the following theorem for the performance of the true risk of the classifier $\hat{t}_n$.

Theorem 3 *([22], Theorem 12.1). Assume that $\min_{t\in\mathcal{C}} \mathrm{e}(t) = 0$ and that $|\mathcal{C}| < \infty$. Then for every n and $\epsilon > 0$ the following two inequalities hold:*

$$\begin{aligned} \mathrm{P}\{\mathrm{e}(\hat{t}_n) > \epsilon\} &\leq |\mathcal{C}|e^{-n\epsilon}, \\ \mathrm{E}\{\mathrm{e}(\hat{t}_n)\} &\leq \frac{1 + \log|\mathcal{C}|}{n}. \end{aligned} \tag{3.4}$$

Proof ([22]).

$$\begin{aligned} \mathrm{P}\{\mathrm{e}(\hat{t}_n) > \epsilon\} &\leq \mathrm{P}\left\{\max_{t\in\mathcal{C}:\hat{\mathrm{e}}_n(t)=0} \mathrm{e}(t) > \epsilon\right\} \\ &= \mathrm{E}\left\{\mathbb{1}_{\{\max_{t\in\mathcal{C}:\hat{\mathrm{e}}_n(t)=0} \mathrm{e}(t)>\epsilon\}}\right\} \\ &= \mathrm{E}\left\{\max_{t\in\mathcal{C}} \mathbb{1}_{\{\hat{\mathrm{e}}_n(t)=0\}}\mathbb{1}_{\{\mathrm{e}(t)>\epsilon\}}\right\} \\ &\leq \sum_{t\in\mathcal{C}:\mathrm{e}(t)>\epsilon} \mathrm{P}\{\hat{\mathrm{e}}_n(t) = 0\} \\ &\leq |\mathcal{C}|(1-\epsilon)^n, \end{aligned}$$

due to the fact that if $\mathrm{P}\{(\mathbf{x}, y) : t(\mathbf{x}) \neq y\} > \epsilon$ then the probability that no $(\mathbf{x}_i, y_i)$ pair falls in the set $\{(\mathbf{x}, y) : t(\mathbf{x}) \neq y\}$ is less than $(1-\epsilon)^n$. In the end we use the simple inequality $1 - x \leq \mathrm{e}^{-x}$. Next, note that for any $s > 0$

$$\begin{aligned}
\mathrm{E}\{\mathrm{e}(\hat{t}_n)\} &\leq \int_0^\infty \mathrm{P}\{\mathrm{e}(\hat{t}_n) > t\}dt \\
&\leq s + \int_s^\infty \mathrm{P}\{\mathrm{e}(\hat{t}_n) > t\}dt \\
&\leq s + |\mathcal{C}| \int_s^\infty e^{-nt}dt \\
&= s + \frac{|\mathcal{C}|}{n} e^{-ns}.
\end{aligned}$$

We are free to select s and if we select $s = \log|\mathcal{C}|/n$, we obtain the desired inequality. □

The drawback of the above theorem is the assumption of zero error on the training sample, which is rather restrictive and not realistic in many practical problems.

Next, we will go back to the general case, i.e., to the case of nonzero training error, and write some parts of the above theory in terms of some measure ν on $\mathbb{R}^d \times \{-1, 1\}$ and begin to study uniform deviations of relative frequencies of errors from their probabilities with the help of standard probability theory.

Thus, suppose we have a probability measure ν of $(\mathbf{x}, \mathsf{y})$ on $\mathbb{R}^d \times \{-1, 1\}$, which was earlier denoted as P, and denote by ν_n the corresponding empirical measure based on the sample Z so that $\nu(B) = \mathrm{P}\{(\mathbf{x}, y) \in B\}$ and $\nu_n(B) = \frac{1}{n}\sum_{i=1}^n \mathbb{1}_{\{(\mathbf{x}_i, y_i) \in B\}}$, where B is a measurable set $B \subset \mathbb{R}^d \times \{-1, 1\}$. With these definitions we can formulate $\mathrm{e}(t)$ as the ν-measure of the set

$$\{\{\mathbf{x} : t(\mathbf{x}) = 1\} \times \{0\}\} \bigcup \{\{\mathbf{x} : t(\mathbf{x}) = 0\} \times \{1\}\}, \quad t \in \mathcal{C}, \tag{3.5}$$

and define $\hat{\mathrm{e}}_n(t)$ as the empirical ν-measure,

$$\hat{\mathrm{e}}_n(t) = \nu_n(\{(\mathbf{x}, y) : t(\mathbf{x}) \neq y\}).$$

Now we have

$$\sup_{t \in \mathcal{C}} |\hat{\mathrm{e}}_n(t) - \mathrm{e}(t)| = \sup_{B \in \mathcal{B}} |\nu_n(B) - \nu(B)|,$$

where $\mathcal{B}$ is the collection of all sets of the form (3.5). If the set B is of finite cardinality we get from Theorem 2

$$\mathrm{P}\{\sup_{B \in \mathcal{B}} |\nu_n(B) - \nu(B)| > \epsilon\} \leq 2|\mathcal{B}|e^{-2n\epsilon^2}. \tag{3.6}$$

3.2 Performance bounds in the infinite case

In the previous subsection, we shortly investigated the bounds on $\sup_{t\in\mathcal{C}} |\hat{\mathrm{e}}_n(t) - \mathrm{e}(t)|$ in the case where the cardinality of $\mathcal{C}$ is finite. Now we will move to the general case of infinite cardinality of $\mathcal{C}$ and give bounds on the random variable $\sup_{B\in\mathcal{B}} |\nu_n(B) - \nu(B)|$. Let us first formulate one of the most fundamental theorems related to the convergence of empirical distributions.

Theorem 4 (Glivenko-Cantelli theorem) *([22], Theorem 12.4). Let $\mathsf{Z}_1, \ldots, \mathsf{Z}_n$ be i.i.d. real-valued random variables with distribution function $F(z) = \mathrm{P}\{\mathsf{Z}_1 \le z\}$. Denote the empirical distribution function by*

$$F_n(z) = \frac{1}{n}\sum_{i=1}^{n} I_{\{\mathsf{Z}_i \le z\}}.$$

$$\textit{Then} \qquad \mathrm{P}\{\sup_{z\in\mathbb{R}} |F_n(z) - F(z)| > \epsilon\} \;\le\; 8(n+1)e^{-n\epsilon^2/32}.$$

Thus, by the Borel-Cantelli lemma we get in particular that

$$\lim_{n\to\infty} \sup_{z\in\mathbb{R}} |F(z) - F_n(z)| = 0 \text{ with probablity one.}$$

Proof. Proof is given in many textbooks, for example in [22].

Next, we will give some definitions needed in the generalization of the Glivenko-Cantelli theorem to the case of infinite cardinality, where the problem of uniform deviations of relative frequencies from their probabilities is formulated in terms of combinatorial quantities. The theory, which gives performance bounds based on the following quantities is usually called *VC-theory* and it has an elegant combinatorial interpretation.

Definition 2 (Shatter coefficient or Covering Number) . *Let $\mathcal{B}$ be a collection of measurable subsets of $\mathbb{R}^d$. For $(\mathbf{x}'_1, \ldots, \mathbf{x}'_n) \in \{\mathbb{R}^d\}^n$ denote by $N_{\mathcal{B}}(\mathbf{x}'_1, \ldots, \mathbf{x}'_n)$ the number of different sets of the form*

$$\{\mathbf{x}'_1, \ldots, \mathbf{x}'_n\} \bigcap B, B \in \mathcal{B}.$$

The n-th shatter coefficient of $\mathcal{B}$ is

$$s(\mathcal{B}, n) = \sup_{(\mathbf{x}'_1, \ldots, \mathbf{x}'_n)\in\{\mathbb{R}^d\}^n} N_{\mathcal{B}}(\mathbf{x}'_1, \ldots, \mathbf{x}'_n). \tag{3.7}$$

In other words, $s(\mathcal{B}, n)$ is the maximal number of different subsets of n points that can be picked out by the class of sets $\mathcal{B}$. One can think of $s(\mathcal{B}, n)$ as the maximum number of different vertices of the n-dimensional cube obtained on the basis of a class of classifiers and a sample $(\mathbf{x}'_1, \ldots, \mathbf{x}'_n)$ and thus $s(\mathcal{B}, n)$ can be thought as a measure of richness of the class. Note that in the literature the term *covering number* is often used for the shatter-coefficient [33, 74].

If for some sample $(\mathbf{x}'_1, \ldots, \mathbf{x}'_n)$ the shatter coefficient $s(\mathcal{B}, n) = 2^n$ it is said that $\mathcal{B}$ shatters $(\mathbf{x}'_1, \ldots, \mathbf{x}'_n)$. It is clear that $s(\mathcal{B}, n) \leq 2^n$ and also that if for some integer V shatter coefficient $s(\mathcal{B}, V) < 2^V$, then $s(\mathcal{B}, n) < 2^n$ for all $n > V$.

Definition 3 (Vapnik-Chervonenkis dimension) . *Let $\mathcal{B}$ be a collection of measurable subsets of $\mathbb{R}^d$ and assume that $|\mathcal{B}| \geq 2$. The largest integer $V_{\mathcal{B}} \geq 1$ for which $s(\mathcal{B}, V_{\mathcal{B}}) = 2^{V_{\mathcal{B}}}$ is called the Vapnik-Chervonenkis dimension* [3] *of the class $\mathcal{B}$. VC-dimension of a class $\mathcal{C}$, denoted as $V_{\mathcal{C}}$, is defined as*

$$\max \left\{ \begin{array}{c} m \mid \exists \mathbf{x}_1, \ldots, \mathbf{x}_m \in \mathbb{R}^d \\ s.t.\ \forall y \in \{-1, 1\}^m\ \exists t \in \mathcal{C}\ s.t.\ t(\mathbf{x}_i) = y_i, i = 1, \ldots, m \end{array} \right\} \tag{3.8}$$

i.e. maximum cardinality of a subset of $\{\mathbb{R}^d\}^n$ that can be shattered by the class of classifiers $\mathcal{C}$. If $s(\mathcal{B}, n) = 2^n$ for all n then, by definition, $V_{\mathcal{B}} = \infty$.

It is easy to show that the VC-dimension of the class of linear classifiers $\{\mathbf{x} \mapsto \text{sgn}(\mathbf{x}^T\mathbf{w} + b),\ \mathbf{w} \in \mathbb{R}^d, b \in \mathbb{R}\}$ is $d + 1$. More details on shatter coefficients and VC-dimensions with examples are given in [2, 22, 83, 85].

In the case of infinite cardinality of $\mathcal{B}$, the trick is that in (3.6) $|\mathcal{B}|$ is replaced by the shatter coefficient $s(\mathcal{B}, n)$. This is stated more formally in the following theorem known also as the Vapnik-Chervonenkis inequality.

Theorem 5 *([22], Theorem 12.5). For any probability measure ν, a class of sets $\mathcal{B}$, and any n and $\epsilon > 0$*

$$\text{P}\{\sup_{B \in \mathcal{B}} |\nu_n(B) - \nu(B)| > \epsilon\} \leq 8 s(\mathcal{B}, n) e^{-n\epsilon^2/32}. \tag{3.9}$$

Proof. A proof can be found in [22], while the original proof given in [81] was formulated with the exponent $-n\epsilon^2/8$.

This bound is completely distribution free and depends only on the combinatorial variable $s(\mathcal{B}, n)$. The bound is useful when the shatter coefficient

[3] Abbreviated as VC-dimension.

does not increase too quickly with n. Moreover, it is clear that if $|\mathcal{B}|$ is small, then also $s(\mathcal{B}, n)$ is small. It is also possible to improve the bounds of subsection 3.1 by using bounds of type (3.9) that are formulated in terms of the shatter coefficients, instead of the cardinality of $\mathcal{C}$. One remark should be given: the supremum in the above theorem is not always measurable and the measurability must be verified for every family $\mathcal{B}$ [22]. In fact, this note holds throughout the present study.

It is also possible to give a tighter distribution dependent bound for $\mathrm{P}\{\sup_{B\in\mathcal{B}} |\nu_n(B) - \nu(B)| > \epsilon\}$. The following bound is proved in [22].

$$\mathrm{P}\{\sup_{B\in\mathcal{B}} |\nu_n(B) - \nu(B)| > \epsilon\} \leq 8\,\mathrm{E}\{N_{\mathcal{B}}(\mathbf{x}_1, \ldots, \mathbf{x}_n)\} e^{-n\epsilon^2/32}, \tag{3.10}$$

where $\mathbf{x}_1, \ldots, \mathbf{x}_n$ are i.i.d. random variables with probability measure ν and the expectation is taken over a random sample of size n.

Next, we will move back from the general probability theory to investigate the uniform convergence of losses of classifiers and to formulate the bound of Theorem 2 in terms of shatter coefficients of the class of classifiers $\mathcal{C}$. Let $\mathcal{C}$ now be a class of classifiers and $\mathcal{B}$ the collection of sets defined in (3.5). We define the n-th shatter coefficient of $\mathcal{C}$ as $s(\mathcal{C}, n) = s(\mathcal{B}, n)$ and furthermore VC-dimension of $\mathcal{C}$ as $V_{\mathcal{C}} = V_{\mathcal{B}}$.

Theorem 6 *([22], Theorem 12.6). Let $\hat{t}_n$ be the classifier that minimizes $\hat{\mathrm{e}}_n(t) = \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}_{\{t(\mathbf{x}_i)\neq y_i\}}$ over the class $\mathcal{C}$. Then*

$$\mathrm{P}\{\sup_{t\in\mathcal{C}} |\hat{\mathrm{e}}_n(t) - \mathrm{e}(t)| > \epsilon\} \leq 8 s(\mathcal{C}, n) e^{-n\epsilon^2/32},$$

and

$$\mathrm{P}\{\mathrm{e}(\hat{t}_n) - \inf_{t\in\mathcal{C}} \mathrm{e}(t) > \epsilon\} \leq 8 s(\mathcal{C}, n) e^{-n\epsilon^2/128} \tag{3.11}$$

Proof. The first inequality is a direct application of Theorem 5. The second follows from (3.2) and from Theorem 5 by dividing ϵ by 2.

If it happens that $\inf_{t\in\mathcal{C}} \mathrm{e}(t) = 0$, we have the following theorem.

Theorem 7 *([22], Theorem 12.7). Let $\mathcal{C}$ be a class of classifiers and $\hat{t}_n = \mathrm{argmin}_{t\in\mathcal{C}}\, \hat{\mathrm{e}}_n(t)$. Suppose that $\inf_{t\in\mathcal{C}} \mathrm{e}(t) = 0$, i.e., the Bayes classifier is in $\mathcal{C}$. Then*

$$\mathrm{P}\{\mathrm{e}(\hat{t}_n) \geq \epsilon\} < 2 s(\mathcal{C}, 2n) 2^{-n\epsilon/2}. \tag{3.12}$$

Proof. Theorem follows from the proof of Theorem 11 (see equation 3.23).

One should in particular notice the exponent in the above theorem. In Theorem 7 the exponent is $c\epsilon$, while in Theorem 6 the exponent is $c'\epsilon^2$, where c and c' are constants. Thus, the the convergence in the zero error case is remarkably faster than in the general case. It is also possible to interpolate between the general distribution free case (3.11) and the zero error case (3.12) [44].

Let us next go back one step to the case of the tighter bound (3.10), which is based on the expected value of $\mathrm{E}\{N_{\mathcal{C}}(\mathbf{x}_1, \ldots, \mathbf{x}_n)\}$. The next theorem is proved in [80].

Theorem 8 *([80], Theorem 4.1). The following inequality holds true*

$$\mathrm{P}\{\sup_{t\in\mathcal{C}} |\hat{\mathrm{e}}_n(t) - \mathrm{e}(t)| > \epsilon\} \leq 4 \exp\left\{ n \left(\frac{\ln \mathrm{E}\{N_{\mathcal{C}}(\mathbf{x}_1, \ldots, \mathbf{x}_{2n})\}}{n} - \left(\epsilon - \frac{1}{n}\right)^2 \right) \right\}. \tag{3.13}$$

The drawback of this theorem is that it is usually difficult to find the value of $\ln \mathrm{E}\{N_{\mathcal{C}}(\mathbf{x}_1, \ldots, \mathbf{x}_n)\}$ since it is a distribution dependent variable (P is unknown) and it can be easily calculated only in some very special settings, see for example [53]. In the literature the term annealed entropy is used for $\ln \mathrm{E}\{N_{\mathcal{C}}(\mathbf{x}_1, \ldots, \mathbf{x}_n)\}$ [79], while we use term *annealed VC-entropy*[4] to emphasize the difference to Shannon's entropy used in information theory. Annealed VC-entropy can further be bounded in a distribution free manner in the following way

$$\ln \mathrm{E}\{N_{\mathcal{C}}(\mathbf{x}_1, \ldots, \mathbf{x}_n)\} \leq \ln \sup_{(\mathbf{x}_1, \ldots, \mathbf{x}_n) \in \{\mathbb{R}^d\}^n} N_{\mathcal{C}}(\mathbf{x}_1, \ldots, \mathbf{x}_n) = \ln s(\mathcal{C}, n),$$

where $\ln s(\mathcal{C}, n)$ is usually called the *growth function* and denoted as $G(\mathcal{C}, n)$.

Let us now return to the consistency of the ERM principle (see Definition 1). The following theorem gives necessary and sufficient conditions for the consistency of the ERM principle.

Theorem 9 *([80], Theorem 3.1) For consistency of the ERM method it is necessary and sufficient that uniform one-sided convergence of the means to their mathematical expectation takes place, i.e.,*

$$\lim_{n\to\infty} \mathrm{P}\left\{ \sup_{t\in\mathcal{C}} (\mathrm{e}(t) - \hat{\mathrm{e}}_n(t)) > \epsilon \right\} = 0, \quad \forall \epsilon > 0.$$

[4]Some authors use only term entropy.

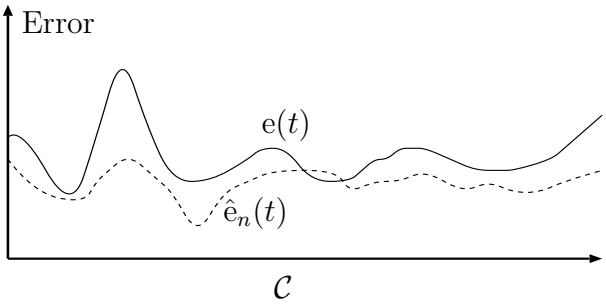


Figure 2: Empirical error and true error as a function of $t \in \mathcal{C}$.

Proof. Proof is given in [80].

This theorem states that the consistency is guaranteed if one-sided uniform convergence takes place and, thus, we do not necessarily need two-sided uniform convergence. This, however, does not shift the main focus of our study away from uniform two-sided convergence, since two-sided convergence is a stronger property and it implies one-sided convergence. Another motivation for two-sided convergence is that we may also want to bound the bias of estimation error $|\hat{\mathrm{e}}_n(t) - \mathrm{e}(t)|$ for an arbitrary classifier $t \in \mathcal{C}$, see Figure 2. In addition, the conditions for two-sided convergence are usually simpler (conditions for one-sided uniform converge are given in [80]). The following theorem gives necessary and sufficient conditions for uniform two-sided convergence.

Theorem 10 *([80], p.120) Necessary and sufficient condition for uniform two-sided convergence of the ERM principle for any probability measure is*

$$\lim_{n\to\infty} \frac{G(\mathcal{C}, n)}{n} = 0. \tag{3.14}$$

If this condition holds, then also the rate of convergence is fast, that is, for any $n > n_0$ the following bound holds

$$\mathrm{P}\left\{\sup_{t\in\mathcal{C}} (\mathrm{e}(t) - \hat{\mathrm{e}}_n(t)) > \epsilon\right\} < e^{-cn\epsilon^2},$$

where c is some positive constant.

Proof. Proof follows from Theorems 3.8 and 15.2 of [78].

This condition does not depend on the distribution, since the growth function does not depend on the probability measure used. In [80] an even stronger

theorem is proved, namely, if (3.14) is satisfied then almost sure two sided uniform convergence takes place, i.e.,

$$\sup_{t\in\mathcal{C}} |\,\mathrm{e}(t) - \hat{\mathrm{e}}_n(t)| \xrightarrow[n\to\infty]{\text{a.s.}} 0.$$

In other words, uniform convergence of empirical means implies their almost sure convergence [85].

3.3 Performance bounds based on VC-dimension

Performance bounds of the previous subsection are based on shatter coefficients and, thus, they are in a sense non-constructive bounds as it can be very difficult or even impossible to calculate the value of the shatter coefficient for some $\mathcal{C}$. In this subsection we present bounds that are based on the VC-dimension, the value of which is potentially much easier to calculate. We start by studying the growth function defined at the end of the previous subsection. Although the growth function is a distribution-free quantity there is no general way to easily calculate its value. But what saves our day is the fact that under very general assumptions we can bound the growth function by the VC-dimension. To show this we first give the following important lemma proved in [61].

Lemma 1 *(Sauer's Lemma) Suppose $\mathcal{C}$ has finite VC-dimension $V_\mathcal{C}$. Then the following holds*

$$s(\mathcal{C},n)\begin{cases} = 2^n & : \quad \textit{if} \quad n \le V_\mathcal{C} \\ \le \sum_{i=0}^{V_\mathcal{C}} \binom{n}{i} & : \quad \textit{if} \quad n > V_\mathcal{C}. \end{cases} \tag{3.15}$$

Furthermore, we can bound the value of the sum of binomial coefficients for $n > V_\mathcal{C}$ as follows:

$$\begin{aligned}\sum_{i=0}^{V_\mathcal{C}} \binom{n}{i} &< \left(\frac{n}{V_\mathcal{C}}\right)^{V_\mathcal{C}} \sum_{i=0}^{V_\mathcal{C}} \binom{n}{i} \left(\frac{V_\mathcal{C}}{n}\right)^i < \left(\frac{n}{V_\mathcal{C}}\right)^{V_\mathcal{C}} \sum_{i=0}^{n} \binom{n}{i} \left(\frac{V_\mathcal{C}}{n}\right)^i \\ &= \left(\frac{n}{V_\mathcal{C}}\right)^{V_\mathcal{C}} \left(1 + \frac{V_\mathcal{C}}{n}\right)^n < \left(\frac{n}{V_\mathcal{C}}\right)^{V_\mathcal{C}} e^{V_\mathcal{C}} = \left(\frac{en}{V_\mathcal{C}}\right)^{V_\mathcal{C}}.\end{aligned}$$

Recall now that earlier we defined the growth function as $G(\mathcal{C},n) = \ln s(\mathcal{C},n)$. Taking logarithms in the above bound it can be seen that the growth function as a function of n is either linear or after some point n_0 bounded by a logarithmic function, where the point n_0 by definition is the VC-dimension. The logarithmic bound is

$$G(\mathcal{C},n) \le V_\mathcal{C}\left(\ln\frac{en}{V_\mathcal{C}}\right), \quad n > V_\mathcal{C}. \tag{3.16}$$

If $G(\mathcal{C}, n) = n \ln 2$, for all n, then the VC-dimension is infinite and some collection of n different points can be splitted in all 2^n ways and consequently $\mathcal{C}$ does not perform any generalization. With the help of the above logarithmic bound we can write (3.13) in terms of a finite VC-dimension as

$$\mathrm{P}\{\sup_{t\in\mathcal{C}} |\,\mathrm{e}(t) - \hat{\mathrm{e}}_n(t)| > \epsilon\} \leq 4 \exp\left\{ n \left(\frac{V_{\mathcal{C}}(\ln(\frac{2en}{V_{\mathcal{C}}}))}{n} - \left(\epsilon - \frac{1}{n}\right)^2 \right) \right\}. \quad (3.17)$$

Usually the bounds of the above type are used in such a way that we set the left hand side of the inequality to some value η referred to as the confidence, and solve ϵ in terms of the VC-dimension, the sample size and the confidence. In particular, if we set left hand side of (3.17) equal to η and solve for ϵ, we get the following corollary.

Corollary 1 *With probability at least $1 - \eta$ it holds simultaneously for all functions $t \in \mathcal{C}$, including the function $\hat{t}_n$ which minimizes $\hat{\mathrm{e}}_n(t) = \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}_{\{t(\mathbf{x}_i)\neq y_i\}}$ that*

$$\mathrm{e}(t) \leq \hat{\mathrm{e}}_n(t) + \sqrt{\frac{V_{\mathcal{C}}(\ln(\frac{2en}{V_{\mathcal{C}}})) - \ln\frac{\eta}{4}}{n}} + \frac{1}{n}. \quad (3.18)$$

In the case of zero training error we can use the following theorem

Theorem 11 *Let the class of classifiers $\mathcal{C}$ have VC-dimension $V_{\mathcal{C}}$. For any probability distribution P over n random examples, any classifier $t \in \mathcal{C}$ having $\hat{\mathrm{e}}_n(t) = 0$ has with probability $1 - \eta$ the following upper bound for error probability*

$$\mathrm{e}(t) \leq \frac{2}{n}\left(V_{\mathcal{C}} \log_2 \frac{2en}{V_{\mathcal{C}}} + \log\frac{2}{\eta} \right), \quad (3.19)$$

provided that $n \geq V_{\mathcal{C}}$ and $n > 2/\epsilon$.

In order to prove this theorem we need the following lemma proved first in [81].

Lemma 2 (Symmetrization by ghost sample) . *For all probability measures P and all subsets $\mathcal{U}$ of the σ-algebra Σ over the sample space $\mathbb{R}^d \times \{-1, 1\}$, for $n\epsilon > 2$ we have that*

$$\mathrm{P}\left(Z_n^1 : \sup_{A\in\mathcal{U}:\mathrm{P}(A)>\epsilon} \nu_{Z_n^1}(A) = 0 \right) < 2\,\mathrm{P}\left(Z_n^1 Z_n^2 : \sup_{A\in\mathcal{U}:\nu_{Z_n^1}(A)=0} \nu_{Z_n^2}(A) > \frac{\epsilon}{2} \right), \quad (3.20)$$

where Z_n^1 and Z_n^2 are two independent random samples of size n, $\nu_{Z_n^1}$ and $\nu_{Z_n^2}$ are the corresponding empirical measures and $Z_n^1 Z_n^2$ is the concatenated sample.

Proof of Theorem 11. Using the previous lemma to bound the left hand side of (3.19), i.e. the probability

$$\mathrm{P}\left\{Z_n : \exists t \in \mathcal{C} : \mathrm{err}_{Z_n}(t) = 0, \mathrm{e}(t) > \epsilon\right\},$$

where $\mathrm{err}_{Z_n}(t)$ stands for the number of errors made by t on Z_n, gives the inequality

$$\begin{aligned} &\mathrm{P}\left\{Z_n : \exists t \in \mathcal{C} : \mathrm{err}_{Z_n}(t) = 0, \mathrm{e}(t) > \epsilon\right\} \\ &\leq 2\,\mathrm{P}\left\{Z_n^1 Z_n^2 : \exists t \in \mathcal{C} : \mathrm{err}_{Z_n^1}(t) = 0, \mathrm{err}_{Z_n^2}(t) > \epsilon n/2\right\}. \end{aligned} \tag{3.21}$$

At this point we note that the event $\{\mathrm{err}_{Z_n^1}(t) = 0, \mathrm{err}_{Z_n^2}(t) > \epsilon n/2\}$ is defined via a (sample, classifier) -pair $(Z_n^1 Z_n^2, t)$. To bound the right hand side of the above inequality we use the "swapping group" idea presented in [58]. Let us fix a classifier t and a $2n$-sample Z_{2n} for which the number of errors $\mathrm{err}_{Z_{2n}}(t)$ is fixed and the order of the vectors $\mathbf{z}_i = (\mathbf{x}_i, y_i)$, $i = 1, \ldots, 2n$ is otherwise arbitrary. We denote by $T(t, n)$ the number of different orders in which the points might have been chosen while keeping all the errors with respect to classifier t in the second sample Z_n^2 and set $A := \{\mathrm{err}_{Z_n^1}(t) = 0, \mathrm{err}_{Z_n^2}(t) > \epsilon n/2\ \}$ Since each order of permutation is equally likely the ratio $T(t, n)/(2n)!$ gives an upper bound for

$$\mathrm{P}(\mathrm{err}_{Z_n^1}(t) = 0, \mathrm{err}_{Z_n^2}(t) > \epsilon n/2 \mid Z_{2n}, t, \mathcal{A}), \tag{3.22}$$

where $\mathcal{A}$ is permutation invariant σ-algebra w.r.t. sample Z_{2n}. If $l = (\mathrm{err}_{Z_n^1}(t) + \mathrm{err}_{Z_n^2}(t)) > n\epsilon/2$ then $T(t, n)/(2n)! \leq \binom{n}{l}/\binom{2n}{l} \leq 2^{-l} < 2^{-n\epsilon/2}$. Since the upper bound on (3.22) is deterministic we can use the union bound $\mathrm{P}(a_1 \bigcup \cdots \bigcup a_n) \leq \sum_{i=1}^{n} \mathrm{P}(a_i)$ to give an upper bound for the probability of a union, that is the right hand side of (3.21), while every element in the sum has the same value $2^{-n\epsilon/2}$. So, all we need is the number of elements in the sum, where the events a_i are of the form A. It is clear that there cannot be infinitely many pairs (t, Z_{2n}) [5] and thus what we should consider is the shatter coefficient $s(\mathcal{C}, 2n)$ (see Definition 2), which gives the number dichotomies $\mathcal{C}$ is capable of on a $2n$-sample. Now, applying the union bound gives

$$\mathrm{P}\left\{Z_1^n : \exists t \in \mathcal{C} : \mathrm{err}_{Z_n}(t) = 0, \mathrm{e}(t) > \epsilon\right\} \leq 2s(\mathcal{C}, 2n)2^{-\epsilon n/2},$$

[5] The maximum number is 2^{2n}.

and using Lemma 1 we can write

$$\mathrm{P}\left\{Z_n : \exists t \in \mathcal{C} : \mathrm{err}_{Z_n}(t) = 0, \mathrm{e}(t) > \epsilon\right\} \leq 2\left(\frac{2en}{V_\mathcal{C}}\right)^{V_\mathcal{C}} 2^{-\epsilon n/2}. \tag{3.23}$$

Setting the right hand side of this inequality to η and solving for ϵ we get the result. □

Note that in the zero error case (Theorem 11) the convergence of the true risk of the selected classifier $\hat{t}_n$ towards zero is of the order $O(\ln(n)/n)$, while in the general case (Corollary 1) the rate of convergence is of the order $O(\sqrt{\ln(n)/n})$.

To bound the difference $\hat{\mathrm{e}}_n(t) - \mathrm{e}(t)$, and in particular $\hat{\mathrm{e}}_n(t^*) - \mathrm{e}(t^*)$ where $t^* = \mathrm{arginf}_{t\in\mathcal{C}}\, \mathrm{e}(t)$ [6] we use the Chernoff inequality (see e.g. [85], p.22), which implies

$$\mathrm{P}\left\{\hat{\mathrm{e}}_n(t^*) - \mathrm{e}(t^*) > \epsilon\right\} \leq \exp(-2n\epsilon^2). \tag{3.24}$$

Furthermore, this implies that with probability $1 - \eta$ the inequality

$$\mathrm{e}(t^*) \geq \hat{\mathrm{e}}_n(t^*) - \sqrt{\frac{-\ln\eta}{2n}} \tag{3.25}$$

holds true. Combining this and (3.18) gives the following theorem on the performance of the classifier $\hat{t}_n$ compared to t^*.

Theorem 12 *With probability at least $1 - 2\eta$ the following inequality holds for the function $\hat{t}_n$ which minimizes the empirical risk :*

$$\mathrm{e}(\hat{t}_n) - \mathrm{e}(t^*) \leq \sqrt{\frac{V_\mathcal{C}(\ln(\frac{2en}{V_\mathcal{C}})) - \ln\frac{\eta}{4}}{n}} + \sqrt{\frac{-\ln\eta}{2n}} + \frac{1}{n}. \tag{3.26}$$

3.4 Lower bounds

Up to now, we have only presented upper bounds for classifier performance since usually they are more interesting than lower bounds. Now we will briefly discuss the question of lower bounds. This problem is studied in more detail in [23, 80]. In particular, we consider lower bounds for

$$\sup \mathrm{P}\{\mathrm{e}(t_n) - \mathrm{e}(t^*) \geq \epsilon\}, \tag{3.27}$$

[6] We assume that arginf is well defined.

where the supremum is taken over all possible random pairs $(\mathbf{x}, \mathbf{y})$ and $\mathrm{e}(t_n) = \mathrm{P}\{t_n(\mathbf{x}) \neq y\}$ and where the classifier $t_n = t_n(Z_n)$ can be an arbitrary function of the training set Z_n so we are not restricted to the ERM principle. The bounds are very pessimistic since we are dealing with worst case analysis but, they can be interesting in themselves. First we give the theorem proved in [23] that deals with the simpler case of $\mathrm{e}(t^*) = 0$.

Theorem 13 *([23], Theorem 14.2) Let $\mathcal{C}$ be a class of classifiers with $V_{\mathcal{C}} \geq 2$, $\mathcal{Z}'$ be the set of all random variables $(\mathbf{x}, \mathbf{y})$ for which $\mathrm{e}(t^*) = 0$ and assume that $\epsilon \leq 1/4$, $n \geq V_{\mathcal{C}} - 1$. Then for every classifier t_n based on a training set Z_n,*

$$\sup_{(\mathbf{x},\mathbf{y})\in\mathcal{Z}'} \mathrm{P}\{\mathrm{e}(t_n) \geq \epsilon\} \geq \frac{1}{e\sqrt{\pi V_{\mathcal{C}}}} \left(\frac{2ne\epsilon}{V_{\mathcal{C}} - 1}\right)^{(V_{\mathcal{C}}-1)/2} e^{-4n\epsilon/(1-4\epsilon)}, \tag{3.28}$$

and if it furthermore holds that $n \leq (V_{\mathcal{C}} - 1)/(4\epsilon)$, then

$$\sup_{(\mathbf{x},\mathbf{y})\in\mathcal{Z}'} \mathrm{P}\{\mathrm{e}(t_n) \geq \epsilon\} \geq \frac{1}{2}. \tag{3.29}$$

The next theorem deals with the more general case, where we have no classifier in the class $\mathcal{C}$ with zero error, i.e., $\mathrm{e}(t^*) > 0$.

Theorem 14 *([22], Theorem 14.6) Let $\mathcal{C}$ be a class of classifiers with $V_{\mathcal{C}} \geq 2$ and let $\mathcal{Z}'$ be the set of all random variables $(\mathbf{x}, \mathbf{y})$ for which $\mathrm{e}(t^*) \in (0, 1/4]$. Then for every classifier t_n based on a training set Z_n, and any $\epsilon \leq \mathrm{e}(t^*)$*

$$\sup_{(\mathbf{x},\mathbf{y})\in\mathcal{Z}'} \mathrm{P}\{\mathrm{e}(t_n) - \mathrm{e}(t^*) \geq \epsilon\} \geq \frac{1}{4} e^{-4n\epsilon^2/\mathrm{e}(t^*)}. \tag{3.30}$$

Proof. Proof is given in [22]

We will give one more theorem which gives a lower bound for the performance of a classifier obtained from any learning algorithm, not necessarily the ERM principle.

Theorem 15 *([7]) Let $\mathcal{C}$ be a class of classifiers with finite $V_{\mathcal{C}} \geq 1$. Then for any classifier t_n there exists distributions for which the error probability of t_n over n random samples is at least*

$$\max\left(\frac{V_{\mathcal{C}} - 1}{32n}, \frac{1}{n} \ln \frac{1}{\eta}\right) \tag{3.31}$$

with probability at least η.

Proof. Proof follows from Theorem 8.6.1 of [3]

One must note that this lower bound holds for particular distributions, while upper bounds of the previous subsection hold for all distributions.

4 Structural risk minimization

We can see from the right hand side of (3.18) that the bound for the true error of a classifier is an increasing function of the VC-dimension and, on the other hand, the empirical error is a decreasing function of the VC-dimension. It could be said that there is a tradeoff between the empirical error and the complexity of a classifier and therefore it is necessary to make the VC-dimension a controlling variable. It should be pointed out that the approach of making the VC-dimension a controlling variable is well justified when the results of the form (3.18) are thought of as equalities or tight inequalities.

Suppose we have a nested sequence of classifiers $\mathcal{C}^{(m)}$,

$$\mathcal{C}^{(1)} \subset \mathcal{C}^{(2)} \subset \cdots \subset \mathcal{C}^{(m)} \cdots , \tag{4.1}$$

and suppose that the VC-dimension of each class $\mathcal{C}^{(m)}$ is finite so that $V_{\mathcal{C}^{(1)}} \leq V_{\mathcal{C}^{(2)}} \leq \cdots \leq V_{\mathcal{C}^{(m)}} \leq \cdots$. Although the VC-dimension of each class $\mathcal{C}^{(m)}$ is finite, the VC-dimension of the class $\mathcal{C} = \bigcup_m \mathcal{C}^{(m)}$ can be infinite. It is clear that the number of errors $\mathrm{err}_{Z_n}(\hat{t}_n^{(m)})$, i.e., number of errors made by the optimal classifier in the subclass $\mathcal{C}^{(m)}$ over the training set Z, decreases with n,

$$\mathrm{err}_{Z_n}(\hat{t}_n^{(1)}) \geq \mathrm{err}_{Z_n}(\hat{t}_n^{(2)}) \geq \ldots \geq \mathrm{err}_{Z_n}(\hat{t}_n^{(m)}) \ldots . \tag{4.2}$$

The problem is to select the subclass $\mathcal{C}^{(m)}$ that provides the lowest value of right hand side of (3.18)[7]. The problem is that, in general, there is no rule of thumb to directly select the optimal subset of classifiers $\mathcal{C}^{(m)}$ based on the formula (3.18). What can be done, however, is to do the minimization by fixing the subset of classifiers $\mathcal{C}^{(m)}$ and then selecting the classifier $\hat{t}_n^{(m)}$ from each class $\mathcal{C}^{(m)}$. Finally, we select the optimal subset of classifiers $\mathcal{C}^{(\hat{m})}$ which gives the lowest upper bound (3.18). This approach is called *structural risk minimization*, abbreviated as SRM. This principle takes into account the complexity of the approximating function and thus avoids overfitting the data. The SRM penalizes the high complexity of a classifier and thus performs complexity regularization, which is just one formulation of the general Occam's razor principle. Penalizing complexity is a fundamental idea in many statistical approaches, for example in the minimum description length (MDL) principle [59]. The next theorem gives consistency conditions for the SRM principle.

[7]One must note that the VC-dimension is an integer and thus the upper bound (3.18) does not vary smoothly.

Theorem 16 *([22], Theorem 18.2). Let* $\mathcal{C}^{(1)} \subset \mathcal{C}^{(2)} \subset \cdots \subset \mathcal{C}^{(m)} \cdots$ *be a sequence of classes of classifiers with finite VC-dimensions satisfying*

$$\sum_{m=1}^{\infty} e^{-V_{\mathcal{C}^{(m)}}} < \infty \tag{4.3}$$

and suppose that for any distribution it holds that

$$\lim_{m\to\infty} \inf_{t\in\mathcal{C}^{(m)}} \mathrm{e}(t) = \mathrm{e}(t^*). \tag{4.4}$$

Then the classification rule t *based on structural risk minimization is strongly universally consistent, that is,*

$$\lim_{n\to\infty} \mathrm{e}(t_n) = \mathrm{e}(\tilde{t}) \quad w.p.\ 1$$

for any distribution of the pair $(\mathbf{x}, y)$.

Proof. Proof is given in [22].

It should be pointed out that according to Vapnik who introduced the SRM principle in [78], the subsets should be constructed a priori without seeing the data first [79, 80]. The assumption of a priori construction of the hierarchy can be problematic, because in real life situations it can be difficult to construct such a hierarchy without seeing the data first. In practice, using (3.18) to choose the subset $\mathcal{C}^{(m)}$, it is necessary to calculate or estimate the VC-dimension of each class of classifiers and this can be extremely difficult. Shawe-Taylor et al. [70] presented a method to overcome the problem of a priori hierarchy by allowing the construction of subsets of classifiers to be dependent on the data. This method is called data-dependent or data-sensitive SRM. In this approach, the data is, in a sense, used twice: first to decide the hierarchy of the classes and then to find the best classifier in each subset. The drawback of data-dependent SRM is that the bounds can be looser than in standard SRM [14]. More details concerning data dependent structural risk minimization is given in Section 6.

5 Large margin classifiers

The bounds based on the VC-dimension presented in Section 2 also have a few drawbacks. The most important one is that they essentially neglect some information in the data. Consider for example the two classification problems shown in Figure 3. The bounds based on the VC-dimension give

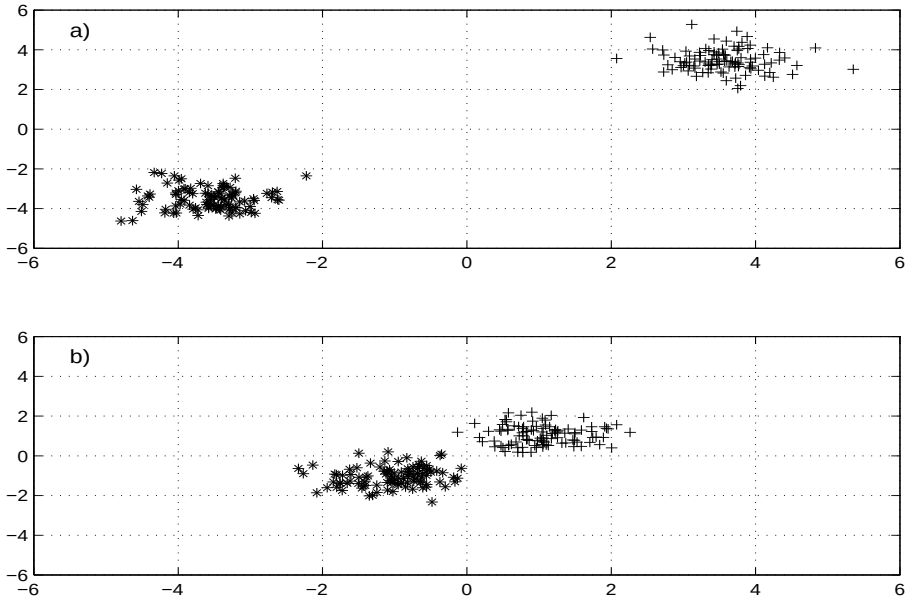


Figure 3: Two classification problems with equal class variances. In a) two Gaussian clouds with unit variances and the difference between the means is 10. In b) clouds have unit variance and the difference between the means is one.

the same performance bounds for both classification problems, without taking advantage of the fact that the first classification problem is essentially easier. Another drawback is that there exists a class of classifiers with infinite VC-dimension that nevertheless generalizes well [4, 14, 74]. It can be said that in some cases bounds based on the VC-dimension are too loose. Moreover, finding the VC-dimension analytically is simple only for a simple classes of functions and it can be extremely difficult for non-linear functions [2, 83]. In this section we give bounds which overcome the above difficulties and also provide tighter upper bounds. This is done by shortening the chain of inequalities that was used to bound the fundamental quantities like covering number.

5.1 Basic definitions of large margin classifiers

Suppose that our classifier is a real-valued function under a threshold function T_θ, $t = T_\theta \circ f$ and $\mathcal{C} = \{T_\theta \circ f : f \in \mathcal{F}\}$, where $\mathcal{F}$ is a class of real-valued functions and in the following we use $T_\theta(f)$ to denote classifier t. Function f under a threshold function is usually referred as *discriminant function*. We

define the threshold function T_θ as

$$T_\theta(f(\mathbf{x}) = \begin{cases} 1 & : \quad \text{if} \quad f(\mathbf{x}) \geq \theta \\ -1 & : \quad \text{if} \quad f(\mathbf{x}) < \theta. \end{cases} \tag{5.1}$$

We use $\theta = 0$, if not otherwise stated. Clearly, when $\theta = 0$ the threshold function is the same as the sgn-function.

Next, we will define an important property of a real-valued function f.

Definition 4 (Margin) *The margin γ_i with respect to f of an example $(\mathbf{x}_i, y_i) \in Z$ is defined to be*

$$\gamma_i = y_i f(\mathbf{x}_i). \tag{5.2}$$

If the value $\theta = 0$ is used then the thresholded classification of $\mathbf{x}_i$ is correct if and only if $\gamma_i > 0$. The minimum value of margin with respect to the training set Z, that is,

$$\min\{y_i f(\mathbf{x}_i) : (\mathbf{x}_i, y_i) \in Z\}$$

is called *the margin* of f and we denote it by $m_Z(f)$. Let us still give a few definitions that we will be needed later.

Definition 5 (Covering number for real-valued function classes) *Let $(\mathcal{S}, d)$ be a (pseudo-)metric space, let $\mathcal{F}$ be a subset of $\mathcal{S}$ and suppose $\gamma > 0$. A set $G \subseteq \mathcal{S}$ is a γ-cover for $\mathcal{F}$ if, for every $a \in \mathcal{F}$, there exists $g \in G$ such that $d(a, g) < \gamma$. The minimum cardinality of a γ-cover of $\mathcal{F}$ is the γ-covering number of $\mathcal{F}$ and it is denoted by $\mathcal{N}_d(\gamma, \mathcal{F})$ when it is finite, if not, then we define the γ-covering number of $\mathcal{F}$ is infinite. See Figure 4 for a γ-cover of a set.*

For the metric d we use the l^∞-distance over a finite sample $X = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$ and in the following we use the symbol $\mathcal{N}(\gamma, \mathcal{F}, X_n)$ for the γ-covering number of $\mathcal{F}$ with the l^∞-pseudo-metric induced by a finite sample X of size n.

Earlier in this study we have defined the VC-dimension for classes of binary valued function, and now we will give a generalization of the VC-dimension to classes of real-valued function.

Definition 6 (VC-dimension for classes of real-valued function) *The VC-dimension of a class $\mathcal{F}$, known also as the pseudo-dimension or Pollard dimension, is denoted by $V_\mathcal{F}^p$ and defined as*

$$\max \left\{ \begin{array}{c} m \mid \quad \exists\, \mathbf{c} \in \mathbb{R}^m \ s.t. \ \forall \mathbf{y} \in \{-1, 1\}^m \\ \exists f \in \mathcal{F} \ s.t. \ \operatorname{sgn}(f(\mathbf{x}'_i) - c_i) = y_i, i = 1, \ldots, m \end{array} \right\}. \tag{5.3}$$

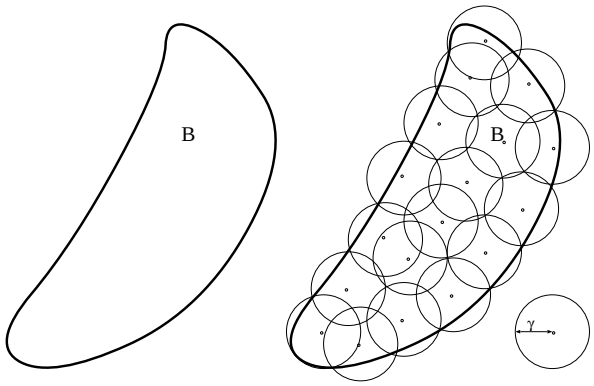


Figure 4: γ-cover of a set B is the set of centers of spheres of radius γ. Covering number of the set appears to be 18.

Next, we will give the definition of a scale sensitive version of pseudo-dimension.

Definition 7 (fat-shattering dimension) *Let $\mathcal{F}$ be a class of real-valued functions. We say that a set of points H is γ-shattered by $\mathcal{F}$ if there are real numbers c_h indexed by $h \in H$, such that for all binary vectors $\mathbf{b} = (b_1, \ldots, b_{\mathrm{card}(H)})$, there is a function $f_{\mathbf{b}} \in \mathcal{F}$ for which*

$$f_{\mathbf{b}}(h) \begin{cases} \geq c_h + \gamma & : \quad \text{if} \quad b_h = 1 \\ \leq c_h - \gamma & : \quad \text{otherwise.} \end{cases} \tag{5.4}$$

The fat shattering dimension $\mathrm{fat}(\mathcal{F}, \gamma)$ of the class $\mathcal{F}$ is a function $\mathbb{R}_+ \to \mathbb{N}$ which maps a value γ to the size of the largest γ-shattered set if this is finite, and to ∞ if the set is not finite.

Fat shattering dimension can be regarded as a generalization of the VC-dimension where the function outputs are above (below) the threshold θ by the amount γ for positive (negative) classification. The analogy of the fat-shattering dimension and the VC-dimension can easily be seen from the following equivalent definition of fat-shattering dimension

$$\max \left\{ \begin{array}{c} m \mid \quad \exists\, \mathbf{c} \in \mathbb{R}^m \text{ s.t. } \forall \mathbf{y} \in \{-1, 1\}^m \\ \exists f \in \mathcal{F} \text{ s.t. } y_i(f(\mathbf{x}_i') - c_i) \geq \gamma, i = 1, \ldots, m \end{array} \right\}. \tag{5.5}$$

It can be thought that the fat shattering dimension measures the richness of the function class up to a certain resolution. In other words, changes have to occur at least at the scale γ while the VC-dimension measures the changes with infinite resolution. By construction, $\lim_{\gamma \to 0} \mathrm{fat}(\mathcal{F}, \gamma)) = V_{\mathcal{F}}^p$. Due to this property the fat shattering dimension is also called fat VC-dimension or scale sensitive (version of) the VC-dimension. Using the fat shattering dimension instead of the VC-dimension it is possible to obtain convergence results at the specific scale one is interested in.

5.2 Minimum margin based performance bounds for large margin classifiers

In this subsection, we give performance bounds based on the minimum margin, covering numbers and the fat-shattering dimension of a real valued function class $\mathcal{F}$. The idea of replacing the original space by a γ-cover with $N(\gamma, \mathcal{F}, X_n)$ elements is similar to replacing $|\mathcal{B}|$ by a shatter coefficient (covering number) $s(\mathcal{B}, n)$ in the case of a binary valued function class. The next two theorems give bounds on the error probability in terms of the fat shattering dimension both in the zero training error and in the general case.

Theorem 17 *([70]) Consider a class of real-valued function $\mathcal{F}$ with fat shattering dimension bounded above by a function* $\text{afat} : \mathbb{R}^+ \to \mathbb{N}$ *which is continuous from the right. Fix $\theta \in \mathbb{R}$. If a classifier $T_\theta(f)$ classifies correctly n independently generated samples and $\gamma = \min |f(\mathbf{x}_i) - \theta|$, then with probability at least $1 - \eta$ the error probability of $T_\theta(f)$ is no more than*

$$\frac{2}{n}\left(h \log_2\left(\frac{8en}{h}\right)\log_2(32n) + \log_2\left(\frac{8n}{\eta}\right)\right), \tag{5.6}$$

where $h = \text{afat}_{\mathcal{F}}(\gamma/8)$.

Proof. Proof is given in [70].

The next theorem deals with the general case, that is, when all the observations cannot be shattered with margin at least γ.

Theorem 18 *([5]) Suppose that the training set Z is chosen by n independent draws from* P. *Then with probability at least $1 - \eta$ every classifier $T_0(f)$ where $f \in \mathcal{F}$ has error probability no more than*

$$\frac{b}{n} + \sqrt{\frac{2}{n}\left(h \log_2\left(\frac{34en}{h}\right)\log_2(578n) + \log_2\left(\frac{4}{\eta}\right)\right)}, \tag{5.7}$$

where $h = \text{fat}_{\mathcal{F}}(\gamma/16)$ and b is the number of training examples with margin less than γ.

Proof. Proof is given in [5].

The bounds in the above two theorems can be considerably tighter than the bounds based on the VC-dimension since fat-shattering dimension as a measure of function class richness can have a much lower value than the VC-dimension. Note again that in the zero training error case convergence is

essentially faster.

The following theorem gives a bound on the error probability in terms of the covering number when all observations have a margin at least γ.

Theorem 19 *([21],Theorem 4.9) Suppose we have a thresholded real-valued function space $T_0(\mathcal{F})$ and fix $\gamma \in \mathbb{R}_+$. Given any distribution* P*, with probability $1-\eta$ any function $f \in \mathcal{F}$ that has minimum margin $m_Z(f) \geq \gamma$ on Z has the error probability upper bound*

$$\mathrm{e}(T_\theta(f)) \leq \frac{2}{n}\left(\log_2 \mathcal{N}(\mathcal{F}, \gamma/2, X_{2n}) + \log_2 \frac{2}{\eta}\right) \tag{5.8}$$

provided that $n > 2/\epsilon$.

Proof ([21]). Using Lemma 2 we can bound the probability

$$\mathrm{P}\left(Z_n^1 : \exists f \in \mathcal{F} : \mathrm{err}_{Z_n^1}(T_0(f)) = 0, m_{Z_n^1}(f) \geq \gamma, \mathrm{e}(T_0(f)) > \epsilon\right)$$

as

$$\begin{aligned} &\mathrm{P}\left(Z_n^1 : \exists f \in \mathcal{F} : \mathrm{err}_{Z_n^1}(T_0(f)) = 0, m_{Z_n^1}(f) \geq \gamma, \mathrm{e}(T_0(f)) > \epsilon\right) \quad (5.9) \\ \leq\ & 2\,\mathrm{P}\left(Z_n^1 Z_n^2 : \exists f \in \mathcal{F} : \mathrm{err}_{Z_n^1}(T_0(f)) = 0, m_{Z_n^1}(f) \geq \gamma, \mathrm{err}_{Z_n^2}(T_0(f)) > \frac{\epsilon n}{2}\right). \end{aligned}$$

Consider now a $\gamma/2$-cover $B(\gamma/2, \mathcal{F})$ of $\mathcal{F}$ with respect to the sample $Z_n^1 Z_n^2$. Suppose that $g \in B$ is such that g is within distance $\gamma/2$ of f. Since we assumed that $m_{Z_n^1}(f) \geq \gamma$ it follows that $m_{\mathsf{Z}_n^1}(g) > \gamma/2$, so that $\mathrm{err}_{Z_n^1}(T_0(g)) = 0$. Now, if $T_0(f)$ makes an error on some point $\mathbf{x}' \in Z_n^2$, then $y' g(\mathbf{x}') < \gamma/2$. We by denote $N(\mathsf{Z}_n^2, g, \gamma/2)$ the number of points in Z_n^2 for which g has margin less than $\gamma/2$. Now we have the bound

$$\begin{aligned} 2\,\mathrm{P}\left(Z_n^1 Z_n^2 : \exists f \in \mathcal{F} : \mathrm{err}_{Z_n^1}(T_0(f)) = 0, m_{Z_n^1}(f) \geq \gamma, \mathrm{err}_{Z_n^2}(T_0(f)) > \frac{n\epsilon}{2}\right) \\ \leq 2\,\mathrm{P}\left(Z_n^1 Z_n^2 : \exists g \in B(\gamma/2, \mathcal{F}) : \mathrm{err}_{Z_n^1}(T_0(g)) = 0, \right. \\ \left. m_{Z_n^1}(g) > \gamma/2, N(Z_n^2, g, \gamma/2) > \frac{n\epsilon}{2}\right). \end{aligned}$$

Fix the pair (f, Z_{2n}) in the same way as in the proof of Theorem 11 we get the bound $2^{n\epsilon/2}$ for

$$\mathrm{P}\left(\mathrm{err}_{Z_n^1}(T_0(g)) = 0, m_{Z_n^1}(g) > \gamma/2, N(Z_n^2, g, \gamma/2) > \frac{n\epsilon}{2} \mid Z_{2n}, t, \mathcal{A}\right),$$

and the number of different (sample, discriminant function) -pairs for which the event $A := \{\mathrm{err}_{Z_n^1}(T_0(g)) = 0, m_{Z_n^1}(g) > \gamma/2, N(Z_n^2, g, \gamma/2) > \frac{n\epsilon}{2}\}$ takes

place is bounded above by the covering number $\mathcal{N}(\mathcal{F}, \gamma/2, X_{2n})$. Now we can write

$$\mathrm{P}\left(Z_n^1 : \exists f \in \mathcal{F} : H\right) \leq 2\mathcal{N}(\mathcal{F}, \gamma/2, X_{2n})2^{-\epsilon n/2}.$$

Setting the right hand side of this inequality to η and solving for ϵ gives the result. □

It is clear that as the margin grows the covering number decreases and so the number of classifiers with the margin γ decreases. It should also be clear that the margin in a sense represents confidence in our classification and therefore instead of considering the infinitely small margin case (VC-dimension) we are motivated to study some scale sensitive case.

It is also possible to give tighter bounds that are more data dependent. In some cases it can be more advantageous to base our approximation of error probability on the whole set of margins $\boldsymbol{\gamma}$, instead of using only margin. Then we would use all the available information, instead of basing the bounds on the locations of a possibly small proportion of the training vectors. Such results are given in [71–73].

The practical drawback of the above bounds based on the fat-shattering dimension $\mathrm{fat}_{\mathcal{F}}(\gamma)$ and the γ-covering number $\mathcal{N}(\gamma, \mathcal{F}, X_n)$ is that these quantities should be known in advance. This drawback also holds for bounds based on the VC-dimension. In other words: one should have a priori information about the fundamental quantities on which the bounds are based and this can be difficult. Theory which avoids this difficulty by basing the bounds on a posteriori quantities is discussed briefly in the next section.

6 Luckiness framework

Here we will introduce a generalization of the SRM principle called data dependent structural risk minimization, and in particular discuss the luckiness framework which was first introduced in [69] and with more details in [70].

Recall now that the bounds based on standard VC theory presented in subsection 3.3 (and some bounds of subsection 5) are of the form:

$$\text{error bound} = \text{empirical error} + \text{complexity term},$$

where the second term on the right hand side depends on some complexity related quantity assumed to be known a priori on the basis of the worst

case properties of the class $\mathcal{C}$ of classifiers considered. In standard SRM the error bounds are similar for each subclass $\mathcal{C}^{(m)}$, where the decomposition into subclasses is done a priori and independently of the training sample. In short: the standard VC-theory and SRM ignore completely the data in evaluating the complexity term - the only quantity which depends on the data is $\mathrm{err}_{Z^n}(t)$, or $\hat{\mathrm{e}}_n(t) = \mathrm{err}_{Z^n}(t)/n$. Luckiness framework uses information from the training sample to make an assumption about some property that measures the goodness of classifier based on observed data and then encodes this assumption in to a real valued function $\Lambda : \mathcal{X}^n \times \mathcal{C} \mapsto \mathbb{R}^+$, luckiness function, whose value tells to what extent our assumption is satisfied for a particular classifier and a particular sample. This approach is much more reasonable for practical problems since we investigate in some sense the "real" situation and not the crude worst case scenario. Given a training set $Z = (X, Y)$ we define a *level* l of a classifier $t \in \mathcal{C}$ relative to X (and Λ) as

$$l(X,t) = |\{\mathbf{b} \in \{-1,1\}^n : \exists g \in \mathcal{C}, g(\mathbf{x}_i) = b_i, \mathbf{x}_i \in X, \Lambda(X,g) \geq \Lambda(X,t)\}|. \tag{6.1}$$

Using the loss function defined in (2.1) the level $l(X,t)$ effectively counts the number of functions $g \in \mathcal{C}$ that give different value of the loss function (i.e. equivalence classes) w.r.t. zero-one loss and which are at least as lucky as t. We can also consider a unluckiness function $\Upsilon : \mathcal{X}^n \times \mathcal{C} \mapsto \mathbb{R}^+$ instead of luckiness function Λ. In this case the level of a classifier $t \in \mathcal{C}$ relative to X (and Υ) is defined by

$$l(X,t) = |\{\mathbf{b} \in \{-1,1\}^n : \exists g \in \mathcal{C}, g(\mathbf{x}_i) = b : i, \mathbf{x}_i \in X, \Upsilon(X,g) \leq \Upsilon(X,t)\}|. \tag{6.2}$$

The definitions of the level functions (6.1) and (6.2) are essentially the same and the choice between the two is just a matter of convenience.

In the subsequent theorem we need the following two definitions which are rather technical in nature.

Definition 8 (α-subsequence) *An α-subsequence of a vector $\mathbf{x}$ is a vector $\mathbf{x}'$ obtained by deleting a fraction $\lfloor \alpha \rfloor$ of coordinates. We write $\mathbf{x}' \subseteq_\alpha \mathbf{x}$ and $\mathbf{x}'\mathbf{z}' \subseteq_\alpha \mathbf{x}\mathbf{z}$ for a concatenation [8] of the vectors $\mathbf{x}$ and $\mathbf{z}$.*

Definition 9 (Probable smoothness of a luckiness function) *A luckiness function $\Lambda(X_n^1, t)$ defined on a function class $\mathcal{C}$ is defined to be probably*

[8] By concatenation of vectors $\mathbf{x}$ and $\mathbf{z}$ we mean the vector $(x_1, \ldots, x_m, z_1, \ldots, z_{m'})$

smooth with respect to the functions $\alpha(n, \Lambda, \eta)$ *and* $\phi(n, \Lambda, \eta)$ *if, for all distributions* P *and all* $\eta \in (0, 1]$

$$\mathrm{P}\left(\mathsf{Z}_n^1\mathsf{Z}_n^2 : \exists t \in \mathcal{C} : \mathrm{err}_{\mathsf{Z}_1^n}(t) = 0 \ \wedge \ \forall \mathsf{X}_n^{1\prime}\mathsf{X}_n^{2\prime} \subseteq_\alpha \mathsf{X}_n^1\mathsf{X}_n^2 : \right.$$
$$\left. l(\mathsf{X}_n^{1\prime}\mathsf{X}_n^{2\prime}, t) > \phi(n, \Lambda(\mathsf{X}_n^1, t), \eta)\right) \leq \eta,$$

where $\alpha = \alpha(n, \Lambda(\mathsf{X}_n^1, t), \eta)$.

If the probable smoothness of a luckiness function holds then luckiness can be estimated from the first half of the sample with high confidence. In other words, when probable smoothness holds then with probability at most η, there exists more than $\phi(n, \Lambda(\mathsf{X}, t), \eta)$ equivalence classes (measured by l and zero-one loss) which are luckier than t on the double sample and for which it holds $\mathrm{err}_{\mathsf{Z}_1^n}(t) = 0$. Note that this rather technical looking requirement is similar to Lemma 2. Replacing the luckiness function Λ by an unluckiness function Υ in Definition 9 leads us to the definition of probable smoothness of an unluckiness function.

At this point we give the following error probability bound based on the luckiness framework.

Theorem 20 *([33]) Suppose that* Λ *is a luckiness function for a function class* $\mathcal{C}$ *and that* Λ *is probably smooth with respect to functions* α *and* ϕ. *Then for any probability measure* P, *any* $0 < \eta < 1/2$, *and any* $d \in \mathbb{N}$, *with probability at least* $1 - \eta$ *over training set* Z *of size* n *chosen according to* P, *if one can find a classifier* $t \in \mathcal{C}$ *which satisfies* $\mathrm{err}_Z(t) = 0$ *and* $\phi(n, \Lambda(X, t), \eta/4) \leq 2^d$, *then the error probability of* t *has the following bound*

$$\mathrm{e}(t) \leq \frac{2}{n}\left(d + \log_2\left(\frac{4}{\eta}\right)\right) + 4\alpha\left(n, \Lambda(X, t), \frac{\eta}{4}\right)\log_2(4n). \qquad (6.3)$$

Proof. Proof is given in [33].

This result allows the sample to determine the decomposition of $\mathcal{C}$ into subclasses, that is, the complexity of the class is not fixed and furthermore depends on data. The obvious drawback is that there is no guarantee of good performance before the data is seen. First theorem of this type was given by Shawe-Taylor et. al in [70] in a slightly more general situation.

In the following three examples of luckiness functions are given. The examples are taken from [33, 70]. From these examples it can be seen that the

luckiness framework subsumes the standard VC-theory and structural risk minimization as well as some parts of the theory of large margin classifiers. More examples of luckiness functions can be found in [70].

Example 1 (Effective VC dimension luckiness) . *Suppose that in addition to $s(\mathcal{C}, n)$, that is, the maximal number of equivalence classes of classifiers in $\mathcal{C}$ on a sample of size n, we know $s(\mathcal{C}, X)$, that is, the number of dichotomies that $\mathcal{C}$ can generate w.r.t a given training sample $Z = (X, Y)$. For $s(\mathcal{C}, X)$ there is a bound analogous to (3.16), that is, there exists $V_{\mathcal{C}}^{eff}$ such that*

$$s(\mathcal{C}, X) \leq \left(\frac{2en}{V_{\mathcal{C}}^{eff}}\right)^{V_{\mathcal{C}}^{eff}}.$$

It can be shown [33], that the unluckiness function

$$\Upsilon(X, t) = V_{\mathcal{C}}^{eff}$$

is probably smooth w.r.t the functions

$$\alpha(n, \Upsilon(X, t), \eta) = 0,$$

$$\phi(n, \Upsilon(X, t), \eta) = \left(\frac{en}{2\Upsilon(X, t) - \ln(\eta)}\right)^{4\Upsilon(X,t) - \ln(\eta)},$$

where $n \in \mathbb{N}$, $\eta \in (0, 1)$.

There is a strong motivation to calculate the effective VC dimension, since a class of classifiers can have good performance bounds based on low empirical VC dimension, while the VC dimension of the class can be infinite [4, 14, 74]. Unfortunately, calculating the effective VC dimension is a very difficult problem although it can be done in some cases [9].

Example 2 (VC dimension luckiness) . *The unluckiness function*

$$\Upsilon(X, t) = V_{\mathcal{C}}$$

is probably smooth w.r.t the functions

$$\alpha(n, \Upsilon(X, t), \eta) = 0$$

$$\phi(n, \Upsilon(X, t), \eta) = \left(\frac{2en}{\Upsilon(X, t)}\right)^{\Upsilon(X,t)},$$

where $n \in \mathbb{N}$, $\eta \in (0, 1)$ and $t \in \mathcal{C}$.

[9] More details on calculating empirical quantities that measures the complexity of function classes are given in [6, 45].

This example clearly shows that the luckiness framework subsumes the standard VC-theory presented in subsection 3.3. Using this unluckiness function with Theorem 20 leads, up to some constants, to the same bound that was given in Theorem 11. Note that in this example, the unluckiness function is independent of the sample Z and depends only on the class of classifiers $\mathcal{C}$.

Example 3 (Margin luckiness) . *Suppose that $\mathcal{C}$ is a class of linear classifiers in canonical form, $\mathcal{C} = \{\mathbf{x} \mapsto \operatorname{sgn}(\mathbf{w}^T\mathbf{x} + b), \mathbf{w} \in \mathbb{R}^n, b \in \mathbb{R}$ and $\min_{1\leq i\leq n} |\mathbf{w}^T\mathbf{x}_i + b| = 1\}$. The unluckiness function*

$$\Upsilon(X,t) = \max_{1\leq i\leq n} \left(\|\mathbf{x}_i\|^2 \|\mathbf{w}\|^2\right)$$

is probably smooth w.r.t the functions

$$\alpha(n, \Upsilon(X,t), \eta) = \left(\frac{2en}{9\Upsilon(X,t)}\right)^{9\Upsilon(X,t)}$$

$$\phi(n, \Upsilon(X,t), \eta) = \frac{1}{2n}\left(k\log_2\left(\frac{8en}{k}\right)\log_2(32n(4n+2)) + 2\log\left(\frac{2}{\eta}\right)\right),$$

where $n \in \mathbb{N}$, $\eta \in (0,1)$ and $k = \lfloor 1297\Upsilon(X,t)\rfloor$.

In the next section it will be shown that if $\mathcal{C}$ is a class of linear classifiers in canonical form then minimizing the norm of the weight vector is equivalent to maximizing the minimum margin.

Example 4 (Boundary luckiness) . *Consider again the class of linear classifiers in canonical form. Now, if linear threshold function t is a maximal margin hyperplane in canonical form, w.r.t sample Z of size n, defined by parameters $(\hat{\mathbf{w}}, \hat{b})$, then it is possible to show that unluckiness function*

$$\Upsilon(X,t) = |\{\mathbf{x}_i \in X : |\hat{\mathbf{w}}^T\mathbf{x}_i + \hat{b}| = 1\}| \tag{6.4}$$

is probably smooth [70].

Compared to Example 3, this example suggests a different criterion to measure the performance of the linear classifier where one should try to minimize the number of examples on the boundary. Note that the unluckiness function (6.4) effectively discards all training points $\mathbf{x}_i$ except those on the boundary. This approach will be briefly discussed in subsection 7.4 in connection with a specific optimization problem.

Suppose then that we can construct a classifier t (a general classifier which is possibly nonlinear) in such a way that it is based only on n' sample points, $n' < n$, and that it correctly classifies the whole training set Z of size n. Such a method of using only a subset of observations to construct a classifier is sometimes called a compression scheme [27]. The following theorem suggest that we should use maximal compression, i.e., find the smallest subset $Z' \subset Z$ for which the constructed classifier correctly classifies Z. Note the connection to (6.4).

Theorem 21 (A compression scheme) *([21], Theorem 4.25). Consider a classifier t which uses only n' observations and which classifies the whole training set of size n correctly. Then with probability $1 - \eta$*

the classifier $t = t_{n'}$ has error probability no more than

$$\frac{1}{n - n'} \left(n' \log_2 \frac{en}{n'} + \log_2 \frac{n}{\eta} \right). \tag{6.5}$$

Proof. Proof is given in [21].

We conclude this subsection by noting that currently there exist results for luckiness framework only under the assumption that there exists at least one function $t \in \mathcal{C}$ such that $\hat{\mathrm{e}}(t) = 0$.

7 SVM classification

In the following two sections, we will introduce a learning algorithm for classification called the support vector machine (SVM[10]), with many good properties, both theoretical and practical. The use of the SVM can be partially justified by the theory of large margin classifiers and the VC-theory because the SVM is based on a linear discriminant function and consequently it has a very low VC- and fat-shattering dimension. In the following we will also show the connection of the SVM to the luckiness framework, especially with the luckiness function of Example 3. This shows the essential connection between the SVM and the theory presented earlier. One of the most important practical properties of the SVM is that it performs well in real-world situations, see for example [31, 41, 63, 64, 80]. Another important property is its simplicity: it is easy to implement and to understand. The SVM is also robust [75]. A major advantage over many other methods is that the SVM has a global solution, while other methods usually have many local solutions (see for example [10]). In many real-life situations it is unrealistic to assume that the data can be separated without an error using a linear discriminant function, but in the following we will show that the SVM can be generalized to cope also with such situations.

In this section the standard SVM algorithm and some of its extensions are presented. The next section is devoted to the important extension of the SVM to nonlinear discriminant functions. In the Appendix we examine the optimization theory needed in the SVM algorithm in more detail.

7.1 The linearly separable case

Consider first the case where we have a linearly separable training set $Z = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$, that is, there is a linear discriminant function of the form

$$\mathbf{x} \mapsto \mathbf{w}^T\mathbf{x} + b, \ \mathbf{w} \in \mathbb{R}^d, b \in \mathbb{R} \tag{7.1}$$

for which the corresponding decision function $t = \operatorname{sgn}(\mathbf{w}^T\mathbf{x} + b)$ has the property $\hat{\mathrm{e}}_n(t) = 0$. Of course, there can be infinitely many linear classifiers that separate the training set without errors and, consequently our task is to choose the best one. One principled criteria in selecting the classifier is the margin. In particular, we are interested in maximizing the minimal margin of

[10]Some authors also use terms SVM-C or SVC to emphasize that the SVM algorithm is used in classification.

the linear discriminant function with respect to the training set Z [11]. Recall the definition of a functional margin (see Definition 4). One can see that using linear a function of the form (7.1) and values $(\alpha\mathbf{w}, \alpha b)$ instead of $(\mathbf{w}, b)$ increases the functional margin by a factor α and therefore we normalize the minimum margin by requiring that $\mathbf{w}$ and b satisfy

$$\min_{\mathbf{x}_i \in X} |\mathbf{w}^T \mathbf{x}_i + b| = 1. \tag{7.2}$$

Hyperplanes of this form are usually called *canonical hyperplanes*. In the following, instead of the functional margin of Definition 4, we will use the *geometric margin* $\bar{\gamma}$ defined as

$$\bar{\gamma}_i = \frac{y_i(\mathbf{w}^T \mathbf{x}_i + b)}{\|\mathbf{w}\|_2}. \tag{7.3}$$

Now, since we imposed restriction (7.2) on the numerator of (7.3) [12] it can be seen that maximizing the minimal geometric margin reduces to minimizing the norm of the weight vector. The hyperplane that maximizes the geometric minimum margin and satisfies

$$y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1, \ i = 1, \dots, n \tag{7.4}$$

is called the *optimal separating hyperplane*.

The problem of finding the optimal separating hyperplane can be written in the form

$$\begin{aligned} &\min_{\mathbf{w},b} \quad \tfrac{1}{2}\|\mathbf{w}\|_2^2 \\ \text{s.t.} \quad &y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1, \quad i = 1, \dots, n. \end{aligned}$$

We can, instead of solving the above minimization problem, solve the *dual* maximization problem (see the Appendix for details)

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & \textstyle\sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i,j=1}^n \alpha_i \alpha_j y_i y_j (\mathbf{x}_i^T \mathbf{x}_j) \\ \text{s.t.} \quad & \alpha_i \geq 0, \quad i = 1, \dots, n, \\ & \textstyle\sum_{i=1}^n \alpha_i y_i = 0. \end{aligned} \tag{7.5}$$

Note that the dual problem is independent of the dimension of $\mathbf{x}$ and instead scales with the number of observations. This is a very good property, especially when $\mathbf{x}$ is high dimensional and the number of observations remains

[11] Intuitively, this corresponds to minimizing the error probability of the linear classifier since we are maximizing the margin of the "worst" classification in the training set.

[12] Without the constraint there would not exist solution for $\text{argmax}_{(\mathbf{w},b)} \min_i y_i(\mathbf{w}^T \mathbf{x}_i + b)$.

small. This is also essential when using kernels (to be introduced in Section 8) where the dimension of the weight vector can be infinite.

Solving the constrained optimization problem (7.5) gives a solution $\hat{\boldsymbol{\alpha}}$ in terms of which the optimal weight vector $\hat{\mathbf{w}}$ is

$$\hat{\mathbf{w}} = \sum_{i:\hat{\alpha}_i>0} \hat{\alpha}_i \mathbf{x}_i y_i, \tag{7.6}$$

and the optimal bias term $\hat{b}$ is given by

$$\hat{b} = -\frac{1}{2}\left[\min_{y_i=1}(\hat{\mathbf{w}}^T\mathbf{x}_i) + \max_{y_i=-1}(\hat{\mathbf{w}}^T\mathbf{x}_i)\right].$$

For numerical reasons, it is often better to use more support vectors for calculating $\hat{b}$ and in the case where the number of support vectors from classes -1 and 1 are equal we can set

$$\hat{b} = -\frac{1}{2|\{i\ |\hat{\alpha}_i > 0\}|}\sum_{i:\hat{\alpha}_i>0} \hat{\mathbf{w}}^T\left[\mathbf{x}_i\mathbb{1}(y_i=1) + \mathbf{x}_i\mathbb{1}(y_i=-1)\right]$$

(see the Appendix for details). Training vectors $\mathbf{x}_i$ for which the $\hat{\alpha}_i$'s are strictly positive are called *support vectors*[13]. In the following we use shorthand notation SV for the set of indices i for which the training vectors $\mathbf{x}_i$ are support vectors.

From the Kuhn-Tucker complementary condition (see the Appendix) it follows that the points $\mathbf{x}_i$ for which the α_i are positive must satisfy

$$y_i(\hat{\mathbf{w}}^T\mathbf{x}_i\hat{b}) = 1$$

so that all support vectors lie on a hyperplane at functional distance 1 from the optimal separating hyperplane (see Figure 5). Because of this property, the number of support vectors can be very small. Another important property of support vectors is the following: if we remove all the data points $(\mathbf{x}_i, y_i)$ for which the $\mathbf{x}_i$'s are not support vectors, we get the same solution $(\hat{\mathbf{w}}, \hat{b})$. Thus, the support vectors give, in some sense, a compact representation of the data. The SVM ignores non-informative data and considers only informative data points from point of view of the optimal hyperplane, that is, points lying on the hyperplanes which have unit functional distance from the optimal separating hyperplane. The idea of ignoring data without

[13]Sometimes called *extreme vectors*.

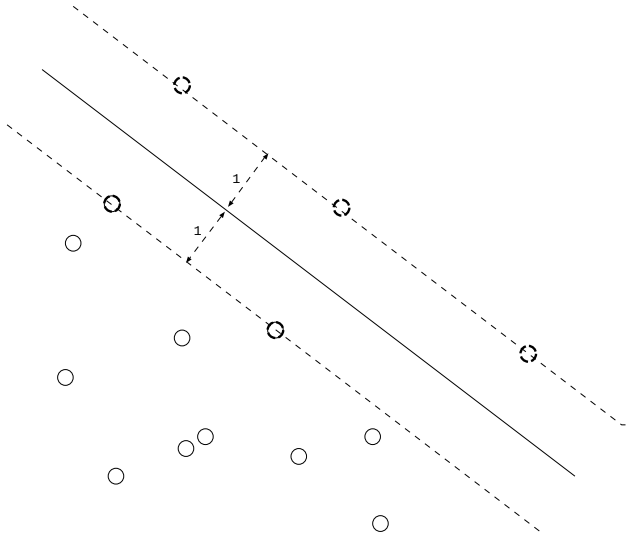


Figure 5: The optimal separating hyperplane is the solid line. Support vectors (highlighted with an extra circle) are the points which lie on hyperplanes (the dashed lines) that have unit (functional) distance to the optimal separating hyperplane.

losing the quality of the estimate is very useful, especially when very large datasets are considered. Of course, this property can be used to classify a of previously unseen vector $\mathbf{x}'$ only after the optimal hyperplane has been found and training must be done using the whole dataset.

After we have found the optimal parameters $\hat{\mathbf{w}}, \hat{b}$, the classifier t is

$$t(\mathbf{x}) = \mathrm{sgn}(\hat{\mathbf{w}}^T\mathbf{x} + \hat{b}). \tag{7.7}$$

Another possibility is to use the optimal parameters as follows,

$$t(\mathbf{x}) = h(\hat{\mathbf{w}}^T\mathbf{x} + \hat{b}), \text{ where } h(s) = \begin{cases} -1 & if \quad s \leq -1, \\ t & if \quad -1 \leq \ s, \leq 1 \\ 1 & if \quad s \geq 1, \end{cases},$$

which can be interpreted to give a naive posterior probability estimate of classification.

7.2 The linearly non-separable case (soft margin)

Here we assume that one cannot separate the data without a misclassification error using the class of linear classifiers but we still try to find a linear

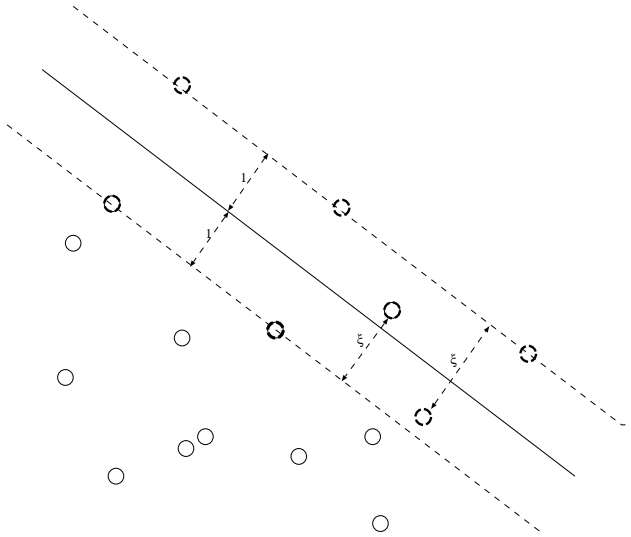


Figure 6: The optimal separating hyperplane in a linearly non-separable case.

discriminant function. From now on we mean by the error of observation ξ the amount by which the discriminant function fails to reach the (functional) unit margin, see also Figure 6. Formally,

$$\xi_i := \max\{0, 1 - y_i(\hat{\mathbf{w}}^T \mathbf{x}_i + \hat{b})\}, \tag{7.8}$$

and misclassification takes place when $\xi_i > 1$.

Using a linear discriminant function in the linearly non-separable case is reasonable at least in two cases. First, if errors are considered noise and the true underlying feature is still linear and, second, if the classes overlap and the optimal decision rule is still linear. For the first case see Figure 6 and for the second case see Figure 9. In the SVM literature classification in the linearly non-separable case is usually called classification with a *soft margin*. In Section 8 we will deal with nonlinear discriminant functions for situations where it is not reasonable to try to find a linear discriminant function. In practice, when there are errors present, one should have some (prior) knowledge about which approach to use, linear or nonlinear discriminant functions.

In the linearly non-separable case the optimal hyperplane is defined to be the hyperplane which maximizes the geometric margin and minimizes some functional $\Theta(\boldsymbol{\xi})$ of the errors. The functional is usually of the form

$$\Theta(\boldsymbol{\xi}) = \sum_{i=1}^{n} \xi_i^{\sigma},$$

where σ is some small positive constant. Usually the value $\sigma = 1$ is used since the corresponding dual does not involve $\boldsymbol{\xi}$ and therefore offers a simpler optimization problem[14]. Also value $\sigma = 2$ has been used [21, 80]. The constraint (7.4) now assumes the form

$$y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i, \ i = 1, \ldots, n, \tag{7.9}$$

where $\xi_i \geq 0$. Selecting $\sigma = 1$ the optimization problem is

$$\begin{aligned} &\min_{\mathbf{w},b,\boldsymbol{\xi}} && \tfrac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{i=1}^n \xi_i && \\ &\text{s.t.} && y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i, && i = 1, \ldots, n, \\ &&& \xi_i \geq 0, && i = 1, \ldots, n. \end{aligned} \tag{7.10}$$

Here C is a positive trade-off parameter which intuitively defines how important it is to avoid misclassification errors. Selection of C is briefly discussed in Section 8.3. It was suggested in [15] that every observation has its own trade-off parameter C_i, the second term in the objective being $\sum_{i=1}^n C_i\xi_i$. In [84] different trade-off parameters for the different labels -1 and 1 are proposed. Again, instead of solving direct optimization problem (7.10) we consider the corresponding dual problem with the objective function

$$W(\boldsymbol{\alpha}) = \sum_{i=1}^n \alpha_i - \frac{1}{2}\sum_{i,j=1}^n \alpha_i\alpha_j y_i y_j (\mathbf{x}_i^T\mathbf{x}_j),$$

to be maximized with the constraints

$$\begin{aligned} &0 \leq \alpha_i \leq C, \quad i = 1, \ldots, n, \\ &\textstyle\sum_{i=1}^n \alpha_i y_i = 0. \end{aligned} \tag{7.11}$$

The only difference between the duals in the linearly separable and the non-separable cases is that in the non-separable case the coefficients α_i have an upper bound C. From the Kuhn-Tucker complementary condition it also follows that $\hat{\alpha}_i = C$ if and only if $\xi_i > 0$ and thus the vectors $\mathbf{x}_i$ with $\xi_i > 0$ are support vectors.

7.3 Semi-supervised support vector machines

Suppose that we are given a training set Z and a test set $X^* = (\mathbf{x}_1^*, \ldots, \mathbf{x}_m^*)$ which is to be classified using labels -1 and 1. In this subsection we will

[14]While this optimization problem is simpler, the choice $\sigma > 1$ produces solution with better properties, see the Appendix for more.

present an approach where the optimal hyperplane is selected based not only on the training set but also on the test set. Taking into account also the test set it is hoped that fewer errors are made as we know how the estimated labeling rule is going to be used, that is, the points on which the classifier is going to be used are known. Of course, there are no guarantees that this approach will work well for totally unknown points, i.e., for points that are not in the test set. In [78] this approach of considering also the test set is called overall risk minimization (ORM). The approach where the ORM principle is applied to the SVM methodology is called in [8] the *semi-supervised* SVM.

Earlier the problem of estimating the values of the unknown function at given points has been solved by moving from particular to general (induction) and then from general to particular (deduction). In the first stage the unknown function is estimated and in the second stage the values of the functions are estimated. The inductive problem of the first stage, i.e, moving from particular to general is in general a difficult problem, especially if we have only a finite or a small training set. Moreover, to guarantee low error probability requires the function to be estimated with high precision for which large or infinite training set is needed. The approach of moving from particular to particular without solving the general problem as an intermediate problem is called transductive inference in [8, 18, 79].

The test set can be regarded as a source of a priori information concerning the points at which the values of the unknown functions are of interest. The semi-supervised support vector machines can also be considered a generalization of the usual support vector approach. If the test set is empty the method becomes the standard SVM and if the training sets is empty the situation is a form of unsupervised learning.

In the semi-supervised SVM the optimization problem in the linearly separable case is

$$\begin{aligned} & \min_{\mathbf{w},b} \quad \tfrac{1}{2}\|\mathbf{w}\|_2^2 \\ \text{s.t.} \quad & y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1, \quad i = 1, \ldots, n, \\ & y_j^*(\mathbf{w}^T\mathbf{x}_j^* + b) \geq 1, \quad j = 1, \ldots, m, \end{aligned}$$

where y_j^* is the unknown label of $\mathbf{x}_j^*$. The last condition for the test set is analogous to the canonical condition that was required earlier only for the training set.

In the linearly non-separable case the optimization problem is

$$\begin{aligned} \min_{\mathbf{w},b,\xi,\xi^*} \quad & \tfrac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}\xi_i + C^*\sum_{j=1}^{m}\xi_j^* \\ \text{s.t.} \quad & y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i, \qquad i = 1, \ldots, n, \\ & y_j^*(\mathbf{w}^T\mathbf{x}_j^* + b) \geq 1 - \xi_j^*, \qquad j = 1, \ldots, m, \end{aligned} \tag{7.12}$$

where C and C^* are pre-specified trade-off parameters.

With a fixed $\mathbf{y}^* = (y_1^*, \ldots, y_m^*)$ the optimization problem in the linearly non-separable case leads to the following objective function α, α^*

$$\begin{aligned} W_{\mathbf{y}^*}(\alpha, \alpha^*) = & \sum_{i=1}^{n}\alpha_i + \sum_{j=1}^{m}\alpha_j^* - \frac{1}{2}\sum_{i,r=1}^{n} y_i y_r \alpha_i \alpha_r (\mathbf{x}_i^T \mathbf{x}_r) \\ & -\frac{1}{2}\sum_{j,r=1}^{m} \alpha_j^* \alpha_r^* y_j^* y_r^* (\mathbf{x}_j^{*T} \mathbf{x}_r^*) - \sum_{i=1}^{n}\sum_{r=1}^{m} y_i y_r^* \alpha_i \alpha_r^* (\mathbf{x}_i^T \mathbf{x}_r^*) \end{aligned}$$

which is to be maximized over variables α, α^* with constraints

$$\begin{aligned} 0 \leq \alpha_i \leq C, \quad & i = 1, \ldots, n, \\ 0 \leq \alpha_j^* \leq C^*, \quad & j = 1, \ldots, m, \\ \sum_{i=1}^{l} y_i \alpha_i + \sum_{j=1}^{m} y_j^* \alpha_j^* &= 0. \end{aligned}$$

The above optimization problem is based on a fixed $(y_1^*, \ldots, y_m^*)$ and the final solution requires finding the solution for all 2^m optimization problems (all possible fixed labelings for $(y_1^*, \ldots, y_m^*)$) and selecting the labeling that has the smallest value of (7.12). This can be very demanding computationally and usually only small test sets can be used. The final classifier is

$$f(\mathbf{x}) = \operatorname{sgn}\Big[\sum_{i=1}^{n}\hat{\alpha}_i y_i (\mathbf{x}^T \mathbf{x}_i) + \sum_{j=1}^{m}\hat{\alpha}_j \hat{y}_j^* (\mathbf{x}^T \mathbf{x}_j^*) + \hat{b}\Big],$$

where $\hat{\mathbf{y}}^* = (\hat{y}_1^*, \ldots, \hat{y}_m^*)$ is the vector of optimal labels for the corresponding test set X^*. The final classifier is analogous to the standard SVM, the only difference is that there are two sums instead of one inside the sign-function.

7.4 Linear approximation of the SVM

Sometimes it might be useful to have an approximation for the solution of the optimal hyperplane for instance when finding the optimal solution is

computationally expensive. In the following two different linear approximation approaches are presented, where the quadratic optimization problem of standard SVM is replaced by linear optimization problem.

7.4.1 Minimizing the number of support vectors

In Section 6 it was shown that the compression scheme (Theorem 21) is an example of a luckiness function. From (7.6) one can see that the expansion of the optimal weight vector $\mathbf{w}$ is in terms of training vectors and thus minimizing the number of support vector belongs to the luckiness framework. Minimizing directly the number of support vectors so that (7.4) or (7.9) holds leads to a very complex optimization problem but this solution can be approximated by minimizing $\sum_{i=1}^{n} \alpha_i$ instead of the number of the positive α_i's [21]. Note that the optimization problem with the target function $\sum_{i=1}^{n} \alpha_i$ such that (7.4) or (7.9) holds is linear. In the linearly non-separable case the optimization problem is

$$\begin{aligned} \min_{\boldsymbol{\alpha},\boldsymbol{\xi}} \quad & \textstyle\sum_{i=1}^{n} \alpha_i + C \sum_{i=1}^{n} \xi_i & \\ \text{s.t.} \quad & y_i(\textstyle\sum_{j=1}^{n} \alpha_i \mathbf{x}_i^T \mathbf{x}_j + b) \geq 1 - \xi_i, & i = 1, \ldots, n \\ & \alpha_i \geq 0, \quad \xi_i \geq 0 & i = 1, \ldots, n. \end{aligned} \tag{7.13}$$

Note also that this optimization problem is formulated already in the dual space.

7.4.2 Robust Linear Programming

In robust linear programming (RLP) the goal is to minimize only the sum of errors under the canonical condition (7.9) [12]. The optimization problem is

$$\begin{aligned} & \min_{\mathbf{w},b,\boldsymbol{\xi}} \quad \textstyle\sum_{i=1}^{n} \xi_i & \\ \text{s.t.} \quad & y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1 - \xi_i, & i = 1, \ldots, n, \\ & \xi_i \geq 0, & i = 1, \ldots, n. \end{aligned}$$

This linear approximation performs well on many real-world problems [9, 12] and it is computationally much cheaper than the standard SVM because of its linearity and because very efficient and robust algorithms such as the simplex method can be used. Note that by adding the term $\|\mathbf{w}\|_2/2$ to the objective function of RLP we get essentially the same optimization problem that we had in the linearly non-separable case (7.10). The drawback of RLP is that it cannot be generalized to nonlinear discriminant functions in as a powerful a way as the standard SVM. Generalization of the standard SVM

to nonlinear discriminant functions will be discussed later.

One can also minimize the l_1-norm of $\mathbf{w}$ instead of the l_2-norm [49]. Setting bounds s_i to the absolute values of w_i and minimizing the sum of these bounds leads to the optimization problem

$$\begin{aligned} & \min_{\mathbf{w},b,\boldsymbol{\xi},\mathbf{s}} \quad \sum_{i=1}^{n} s_i + C\sum_{i=1}^{n}\xi_i & \\ \text{s.t.} \quad & y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i, & i = 1,\dots,n, \\ & s_i \geq w_i \geq -s_i, & i = 1,\dots,n, \\ & \xi_i \geq 0, & i = 1,\dots,n. \end{aligned}$$

More details concerning linear approximation of the SVM are given in [42, 50, 51, 76].

7.5 The SVM for multiclass classification

The standard form of the SVM can be used in two-way classification. However, in real-life situations it is often necessary to separate more than two classes at the same time. The well-known example is the classification of handwritten characters. In this section we explore how the standard SVM can be extended from the binary two-class problem to classification tasks with $k > 2$ classes. Let us first consider some simple extensions of a general binary classification procedure and slight modifications of the standard SVM. After this some more advanced, state-of-the-art methods are investigated. New theoretical foundations of multiclass classification systems are discussed in [25, 29, 30].

7.5.1 One-versus-all

The simplest extension of the SVM to a k-class problem is to separate the observations from class j from the rest for every $j = 1,\dots,k$. Here the "rest" means that all the observations from other classes than j are combined to form one class. The optimal hyperplane that separates samples from the class j and the combined class is found by using the standard SVM approach. We denote the optimal separating hyperplane discriminating the class j and the combined class as

$$\mathbf{x}^T\hat{\mathbf{w}}^j + \hat{b}^j, \quad j = 1,\dots,k,$$

where the superscript in $\hat{\mathbf{w}}^j$ stands for the class which should be separated from the other observations. The decision rule t^j that assigns the vector $\mathbf{x}$

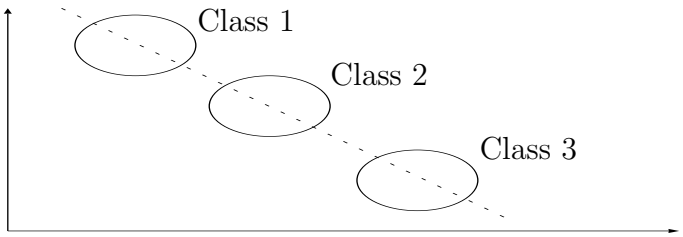


Figure 7: Three classes aligned approximately along a line.

to the class j or to the combined class is

$$t^j(\mathbf{x}) = \text{sgn}\left(g^j(\mathbf{x})\right), \tag{7.14}$$

where $g^j(\mathbf{x}) = \mathbf{x}^T\hat{\mathbf{w}}^j + \hat{b}^j$. After all the k optimal separating hyperplanes defined by $(\hat{\mathbf{w}}^j, \hat{b}^j)$, $j = 1, \ldots, k$ have been found the final classifier $\mathbf{t}_k$ [15] is

$$\mathbf{t}_k(\mathbf{x}) = \text{argmax}_j\left(t^j(\mathbf{x})\right). \tag{7.15}$$

This approach of assigning the class label using argmax-rule is usually called *voting*. In this case voting is performed by giving every classifier a vote of size one and the unknown label is decided to be the index of the classifier that gives the only positive vote. If there are no positive votes or if there is more than one classifier t^j with positive vote, then no decisions about the class label is made.

The one-versus-all approach has drawbacks, that can be rather serious. In (7.15) the argmax-rule is not well-defined since there is not always a unique solution. The main difficulty in this approach is that the outputs of the classifiers t_j are binary values, e.g., $t^j(\mathbf{x}) = t^h(\mathbf{x})$ even if $g^j(\mathbf{x}) \gg g^h(\mathbf{x}) > 0$. The usual way handle this problem is to ignore the sign-operator in (7.14) and to use the argmax-rule in (7.15) for the $g^j(\mathbf{x})$'s. In this approach the index of the largest component of the discriminant vector $(g^1(\mathbf{x}), \ldots, g^k(\mathbf{x}))$ is assigned to the vector $\mathbf{x}$. In [41] this approach is called *winner-takes-all*. Even with this modification the one-versus-all approach has problems which occur for example when the classes are approximately aligned along a line. Figure 7 illustrates such a situation where the class 2 cannot be reasonably separated.

[15] We use here the symbol $\mathbf{t}_k$ instead of t since t was used earlier to denote a binary classifier. The subscript k of $\mathbf{t}_k$ signifies the number of classes.

7.5.2 Scaling the outputs of the SVM

One can scale the outputs g^j and apply the argmax-rule to the scaled outputs. In [52] four different ways to normalize the g^j's before applying the argmax-rule are presented. The normalization factor for the output $g^j(\mathbf{x}) = \mathbf{x}^T\hat{\mathbf{w}}^j + \hat{b}^j$ is denoted by π^j. The decision rule gets the form

$$\mathbf{t}_k(\mathbf{x}) = \text{argmax}_j \left(\frac{g^j}{\pi^j} \right). \tag{7.16}$$

The proposed normalizations are

1. $\pi^j = \|\hat{\mathbf{w}}^j\|$. The outputs $g^j(\mathbf{x})$ are divided by the Euclidean norm of $\hat{\mathbf{w}}^j$. This has an interesting geometrical interpretation as $\frac{g^j(\mathbf{x})}{\pi^j}$ is the Euclidean distance from $\mathbf{x}$ to the hyperplane.

2. $\pi^j = \sqrt{\text{Var}(g^j(\mathbf{x}))}$. The normalization term is the sample standard deviation of $g^j(\mathbf{x})$. The variance is computed over $\mathbf{x}_i$, $i = 1, \ldots, n$.

3. $\pi^j = \frac{1}{n}\sum_{i=1}^n y_i g^j(\mathbf{x}_i)$. With this normalization $\frac{1}{n}\sum_{i=1}^n \frac{y_i g^j(\mathbf{x}_i)}{\pi^j} = 1$ for all $j = 1, \ldots, k$.

4. $\pi^j = \frac{\sum_{i=1}^n y_i g^j(\mathbf{x}_i)}{\sum_{i=1}^n g^j(\mathbf{x}_i)}$. This normalization is motivated by the fact that now $\sum_{i=1}^n (\frac{g^j(\mathbf{x}_i)}{\pi^j} - y_i)^2$ is minimized.

A more general way to use scaling is to consider a decision rule

$$\mathbf{t}_k(\mathbf{x}) = \text{argmax}_j((\mathbf{M}\mathbf{g})_j),$$

where $\mathbf{M}$ is a $k \times k$-mixture matrix, $\mathbf{g} = (g^1(\mathbf{x}), \ldots, g^k(\mathbf{x}))$, and $(\mathbf{z})_j$ is the j-th component of a vector $\mathbf{z}$. When $\mathbf{M}$ is the $k \times k$-identity matrix the problem reduces to the standard $\text{argmax}_j(\mathbf{x}^T\hat{\mathbf{w}}^j + \hat{b}^j)$ -rule and if $\mathbf{M} = \text{Diag}(\pi^1, \ldots, \pi^k)^{-1}$ the problem reduces to (7.16).

7.5.3 Pairwise classification

In pairwise classification the problem is to find the optimal separating hyperplane that discriminates vectors of classes i and j, $i \neq j$. We denote such a discriminant function by t^{ij}. Since the discriminant function t^{ij} is symmetric, i.e., $t^{ji} = -t^{ij}$, the total number of binary classifiers is $\binom{k}{2}$. The final classifier then assigns the label $\text{argmax}_j\, t^j(\mathbf{x})$ to the vector $\mathbf{x}$, where $t^j(\mathbf{x}) = \sum_{\substack{i=1 \\ i\neq j}}^k t^{ji}(\mathbf{x})$ and $t^{ij}(\mathbf{x}) = \text{sgn}\left(\mathbf{x}^T\hat{\mathbf{w}}^{ij} + \hat{b}^{ij}\right)$. The optimal separating hyperplane determined by parameters $(\hat{\mathbf{w}}^{ij}, \hat{b}^{ij})$ that discriminates vectors of

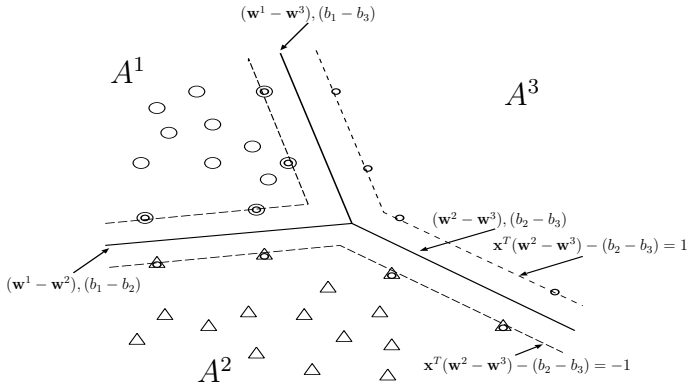


Figure 8: Illustration of piecewise linear separability when $k = 3$. The optimal piecewise separating hyperplanes are the solid lines and piecewise hyperplanes having unit functional distance to corresponding optimal hyperplane are depicted by the dashed lines. Support vectors are denoted with extra circles.

classes i and j is found by the standard SVM ignoring observations which are not from classes i and j. The maximum number of positive votes $\max_j t^j(\mathbf{x})$ is $k-1$. Possible tie situations where a unique maximum is not found can be solved for example by ignoring the sgn-operator and using the argmax rule in regions of the $\mathbf{x}$-space where decisions cannot be made [41]. More advanced multi-class methods based on pairwise classification are given in [28, 32, 57].

7.5.4 M-SVM with piecewise linearly separable classes

All the above approaches involved several independent discrimination steps. In the one-versus-all approach k optimization problems were solved to separate one class from the others, and in pairwise classification $\binom{k}{2}$ optimizations were needed. A more natural and principled way is to perform the classification over all k classes simultaneously. In the following we present one such approach called called M-SVM [12]. Let us first give the definition of piecewise linear separability.

Definition 10 (Piecewise linear separability) . *The sets of points given as rows of the matrices $\mathbf{A}^j \in \mathbb{R}^{n_j \times d}$, $j = 1, \ldots, k$, are piecewise linearly separable if $\exists\ \mathbf{w}^j \in \mathbb{R}^d$ and $b^j \in \mathbb{R}$, $j = 1, \ldots, k$, such that*

$$\mathbf{A}^j \mathbf{w}^j + b^j \mathbf{e}^j > \mathbf{A}^j \mathbf{w}^l + b^l \mathbf{e}^j, \quad j, l = 1, \ldots, k, \quad j \neq l, \tag{7.17}$$

where $\mathbf{e}^j$ is the $n_j \times 1$ vector of ones and the inequality is by components. See Figure 8 for an illustration when $k = 3$.

Consider now the case where the k classes are piecewise linearly separable, i.e., there exist k pairs $(\mathbf{w}^j, b^j)$ such that (7.17) holds. We denote by $\mathbf{A}^j$ the $n_j \times d$-matrix, which includes all the $\mathbf{x}$-vectors from the class j, i.e., $\mathbf{A}^j = ({\mathbf{x}_1^j}^T, \ldots, {\mathbf{x}_{n_j}^j}^T)$, where $\mathbf{x}^j \in \mathbb{R}^d$, $j = 1, \ldots, k$, and $\sum_{j=1}^k n_j = n$. Note that under the assumption of piecewise linear separability there exist infinitely many pairs $(\mathbf{w}^j, b^j)$ satisfying the condition (7.17) and the goal is to find the optimal ones by simultaneous optimization over all $(\mathbf{w}^j, b^j)$, $j = 1, \ldots, k$. To avoid this problem we add the unit vector $\mathbf{e}^j$ to the right hand side of (7.17) and require that

$$\mathbf{A}^j\mathbf{w}^j + b^j\mathbf{e}^j \geq \mathbf{A}^j\mathbf{w}^l + b^l\mathbf{e}^j + \mathbf{e}^j, \quad j, l = 1, \ldots, k, \quad j \neq l.$$

The above inequality can further be written as

$$\mathbf{A}^j(\mathbf{w}^j - \mathbf{w}^l) + (b^j - b^l)\mathbf{e}^j \geq \mathbf{e}^j \quad j, l = 1, \ldots, k, \quad j \neq l. \tag{7.18}$$

From this form one should see the analogy with hyperplanes in the standard SVM (see (7.4)): now the weight vector is of the form $(\mathbf{w}^j - \mathbf{w}^l)$ and the bias term $(b^j - b^l)$.

Let us consider separation between the classes i and j. The maximum margin of separation between the classes i and j satisfying

$$\begin{aligned}\mathbf{A}^i(\mathbf{w}^i - \mathbf{w}^j) + (b^i - b^j)\mathbf{e}^i &\geq \mathbf{e}^i \\ \mathbf{A}^j(\mathbf{w}^j - \mathbf{w}^i) + (b^j - b^i)\mathbf{e}^j &\geq \mathbf{e}^j\end{aligned}$$

is found by maximizing $2/\|\mathbf{w}^i - \mathbf{w}^j\|$ with the constraint (7.18), that is, the norm of the vector separating the observations of classes i and j (see also Figure 8). In addition to this, the norm of the weight vectors $\mathbf{w}^i$ and $\mathbf{w}^j$ must be controlled [12]. The optimization problem to find the best hyperplanes simultaneously is therefore

$$\min_{\mathbf{w}^i, b^i} \quad \tfrac{1}{2}\sum_{i=1}^k \sum_{j=1}^{i-1} \|\mathbf{w}^i - \mathbf{w}^j\|^2 + \tfrac{1}{2}\sum_{i=1}^k \|\mathbf{w}^i\|^2 \tag{7.19}$$

$$\text{s.t.} \quad \mathbf{A}^i(\mathbf{w}^i - \mathbf{w}^j) - \mathbf{e}^i(b^i - b^j) - \mathbf{e}^i \geq 0, \; i, j = 1, \ldots, k, \quad i \neq j. \tag{7.20}$$

There are also other propositions to generalize the two-class SVM to the multi-class case. In [30, 88] a slightly different objective function for the optimization problem in multi-class SVM is presented and it is suggested that

$$\frac{1}{2}\sum_{i=1}^k \sum_{j=1}^{i-1} \|\mathbf{w}^i - \mathbf{w}^j\|^2$$

is minimized w.r.t $\mathbf{w}^i, \mathbf{w}^j$ and b^i, b^j in such a way (7.20) holds. In this case the optimization problem is essentially the same as in the standard SVM, only the formula for the hyperplane is in a different form and the separation is done piecewise.

Using the following matrix notation we can write the optimization problem (7.19) in the dual space, although the definitions of matrices are quite laborious. Let

$$\bar{\mathbf{A}} = \begin{bmatrix} \mathbf{A}^1 & -\mathbf{A}^1 & 0 & 0 & \cdots & 0 \\ \mathbf{A}^1 & 0 & -\mathbf{A}^1 & 0 & \cdots & 0 \\ \vdots & \vdots & 0 & \ddots & 0 & \vdots \\ \vdots & \vdots & 0 & 0 & \ddots & \vdots \\ \mathbf{A}^1 & 0 & \dots & \dots & 0 & -\mathbf{A}^1 \\ -\mathbf{A}^2 & \mathbf{A}^2 & 0 & 0 & \dots & 0 \\ 0 & \mathbf{A}^2 & -\mathbf{A}^2 & 0 & \dots & 0 \\ \vdots & \vdots & 0 & \ddots & 0 & \vdots \\ \vdots & \vdots & \vdots & 0 & \ddots & 0 \\ 0 & \mathbf{A}^2 & 0 & \dots & \dots & -\mathbf{A}^2 \\ \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ -\mathbf{A}^k & 0 & \dots & \dots & 0 & \mathbf{A}^k \\ 0 & -\mathbf{A}^k & 0 & \dots & 0 & \mathbf{A}^k \\ \vdots & 0 & \ddots & 0 & \vdots & \vdots \\ \vdots & \vdots & 0 & \ddots & 0 & \vdots \\ 0 & \dots & \dots & 0 & -\mathbf{A}^k & \mathbf{A}^k \end{bmatrix}, \tag{7.21}$$

where $\mathbf{A}^i \in \mathbb{R}^{n_i \times d}$. The matrix $\bar{\mathbf{A}}$ is a $((k-1)n) \times kd$ -matrix. Let $\bar{\mathbf{C}}$ be the $\left(d \sum_{i=2}^{k} (i-1)\right) \times kd$-matrix

$$\bar{\mathbf{C}} = \begin{bmatrix} \mathbf{I} & -\mathbf{I} & 0 & 0 & \cdots & 0 \\ \mathbf{I} & 0 & -\mathbf{I} & 0 & \cdots & 0 \\ \vdots & \vdots & 0 & \ddots & 0 & \vdots \\ \vdots & \vdots & 0 & 0 & \ddots & 0 \\ \mathbf{I} & 0 & \dots & \dots & 0 & -\mathbf{I} \\ 0 & \mathbf{I} & -\mathbf{I} & 0 & \dots & 0 \\ \vdots & \vdots & 0 & \ddots & 0 & \vdots \\ \vdots & \vdots & \vdots & 0 & \ddots & 0 \\ 0 & \mathbf{I} & 0 & \dots & 0 & -\mathbf{I} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \dots & 0 & \mathbf{I} & -\mathbf{I} & 0 \\ 0 & \dots & 0 & \mathbf{I} & 0 & -\mathbf{I} \\ 0 & \dots & \dots & 0 & \mathbf{I} & -\mathbf{I} \end{bmatrix}, \tag{7.22}$$

where $\mathbf{I}$ is a $d \times d$ matrix of ones. Let also

$$\bar{\mathbf{E}} = \begin{bmatrix} -\mathbf{e}^1 & \mathbf{e}^1 & 0 & 0 & \cdots & 0 \\ -\mathbf{e}^1 & 0 & \mathbf{e}^1 & 0 & \cdots & 0 \\ \vdots & \vdots & 0 & \ddots & 0 & \vdots \\ \vdots & \vdots & 0 & 0 & \ddots & \vdots \\ -\mathbf{e}^1 & 0 & \dots & \dots & 0 & \mathbf{e}^1 \\ \mathbf{e}^2 & -\mathbf{e}^2 & 0 & 0 & \dots & 0 \\ 0 & -\mathbf{e}^2 & \mathbf{e}^2 & 0 & \dots & 0 \\ \vdots & \vdots & 0 & \ddots & 0 & \vdots \\ \vdots & \vdots & \vdots & 0 & \ddots & 0 \\ 0 & -\mathbf{e}^2 & 0 & \dots & \dots & \mathbf{e}^2 \\ \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \\ \mathbf{e}^k & 0 & \dots & \dots & 0 & -\mathbf{e}^k \\ 0 & \mathbf{e}^k & 0 & \dots & 0 & -\mathbf{e}^k \\ \vdots & 0 & \ddots & 0 & \vdots & \vdots \\ \vdots & \vdots & 0 & \ddots & 0 & \vdots \\ 0 & \dots & \dots & 0 & \mathbf{e}^k & -\mathbf{e}^k \end{bmatrix}, \tag{7.23}$$

where $\mathbf{e}^i$ is the $n_i \times 1$-vector of ones. Here $\bar{\mathbf{E}}$ is a $((k-1)n) \times k$-matrix. With this notation the optimization problem can be rewritten as

$$\begin{aligned} \min_{\mathbf{w},\mathbf{b}} \quad & \frac{1}{2}\|\bar{\mathbf{C}}\mathbf{w}\|^2 + \frac{1}{2}\|\mathbf{w}\|^2 \\ \text{s.t.} \quad & \bar{\mathbf{A}}\mathbf{w} + \bar{\mathbf{E}}\mathbf{b} \geq \mathbf{e}, \end{aligned} \tag{7.24}$$

where $\mathbf{w} = [\mathbf{w}^{1^T}, \mathbf{w}^{2^T}, \ldots, \mathbf{w}^{k^T}]^T$, and $\mathbf{b} = [b^1, b^2, \ldots, b^k]^T$. The dual[16] of this optimization problem can be written as

$$\begin{aligned} \max_{\mathbf{u},\mathbf{w},\mathbf{b}} \quad & \tfrac{1}{2}\|\bar{C}\mathbf{w}\|^2 + \tfrac{1}{2}\|\mathbf{w}\|^2 - \mathbf{u}^T(\bar{\mathbf{A}}\mathbf{w} + \bar{\mathbf{E}}\mathbf{b} - \mathbf{e}) \\ \text{s.t.} \quad & (I + \bar{\mathbf{C}}^T\bar{\mathbf{C}})\mathbf{w} = \bar{\mathbf{A}}^T\mathbf{u}, \\ & -\bar{\mathbf{E}}^T\mathbf{u} = 0, \\ & \mathbf{u} \geq 0, \end{aligned} \tag{7.25}$$

where $\mathbf{u} = [\mathbf{u}^{12^T}, \mathbf{u}^{13^T}, \ldots, \mathbf{u}^{1k^T}, \mathbf{u}^{21^T}, \mathbf{u}^{23^T}, \ldots, \mathbf{u}^{k(k-1)^T}]^T$ is a vector of Lagrange multipliers (see the Appendix) of size $((k-1)n) \times 1$, and where $\mathbf{u}^{ij} \in \mathbb{R}^{n_i \times 1}$. Since $\bar{\mathbf{C}}^T\bar{\mathbf{C}}$ is positive semi-definite and, consequently, $(I + \bar{\mathbf{C}}^T\bar{\mathbf{C}})$ is positive definite, one can solve $\mathbf{w}$ from (7.25),

$$\mathbf{w} = (I + \bar{\mathbf{C}}^T\bar{\mathbf{C}})^{-1}\bar{\mathbf{A}}^T\mathbf{u} = \frac{1}{k+1}\bar{\mathbf{A}}^T\mathbf{u}. \tag{7.26}$$

After a simple manipulation the dual can written as

$$\begin{aligned} \max_{\mathbf{u}} \quad & \mathbf{e}^T\mathbf{u} - \tfrac{1}{2(k+1)}\mathbf{u}^T\bar{\mathbf{A}}\bar{\mathbf{A}}^T\mathbf{u} \\ \text{s.t} \quad & \bar{\mathbf{E}}^T\mathbf{u} = 0, \\ & \mathbf{u} \geq 0. \end{aligned}$$

This optimization problem is concave since the matrix $\bar{\mathbf{A}}\bar{\mathbf{A}}^T$ is positive semi-definite and symmetric and thus the Hessian matrix is negative semi-definite. The above optimization problem can also be written in summation notation as

$$\begin{aligned} \max_{\mathbf{u}} \quad & \sum_{i=1}^{k}\sum_{\substack{j=1 \\ j\neq i}}^{k}\sum_{l=1}^{n_i} u_l^{ij} - \frac{1}{2(k-1)}\sum_{i=1}^{k}\sum_{\substack{j=1 \\ j\neq i}}^{k}\sum_{\substack{l=1 \\ l\neq j}}^{k}\Big[\sum_{p=1}^{n_i}\sum_{q=1}^{n_i} u_p^{ij}u_q^{il}\mathbf{A}_p^i{\mathbf{A}_q^i}^T \\ & -2\sum_{p=1}^{n_j}\sum_{q=1}^{n_i} u_p^{ji}u_q^{il}\mathbf{A}_p^j{\mathbf{A}_q^i}^T + \sum_{p=1}^{n_j}\sum_{q=1}^{n_l} u_p^{ji}u_q^{li}\mathbf{A}_p^j{\mathbf{A}_q^l}^T\Big] \end{aligned}$$

[16]This is not the same dual formulation which was used earlier in the standard SVM.

with the constraints

$$-\sum_{\substack{j=1\\j\neq i}}^{k}\sum_{l=1}^{n_i} u_l^{ij} + \sum_{\substack{j=1\\j\neq i}}^{k}\sum_{l=1}^{n_j} u_l^{ji} = 0, \quad i = 1,\dots,k$$
$$u_l^{ij} \geq 0 \quad i,j = 1,\dots,k, i \neq j \text{ and } l = 1,\dots,n_i,$$

where $\mathbf{A}_q^i$ stands for the q-th row of the matrix $\mathbf{A}^i$. The final classifier is then $\mathbf{t}_k(\mathbf{x}) = \text{argmax}_i\, g^i(\mathbf{x})$, $i = 1,\dots,k$, where $g^i(\mathbf{x}) = \mathbf{x}^T\hat{\mathbf{w}}^i + \hat{b}^i$. Since we have found that $\mathbf{w} = \frac{1}{k+1}\bar{\mathbf{A}}^T\mathbf{u}$ we can write

$$g^i(\mathbf{x}) = \sum_{\substack{j=1\\j\neq i}}^{k}\sum_{p=1}^{n_i} \hat{u}_p^{ij}\mathbf{x}^T{\mathbf{A}_p^i}^T - \sum_{\substack{j=1\\j\neq i}}^{k}\sum_{p=1}^{n_j} \hat{u}_p^{ji}\mathbf{x}^T{\mathbf{A}_p^j}^T - \hat{b}^i.$$

7.5.5 M-SVM with piecewise linearly non-separable classes

In this case, the classes are not piecewise linearly separable. The approach to moving from the standard M-SVM to the case of nonzero errors is analogous to the generalization of the standard SVM to the linearly non-separable case - a penalty term $\boldsymbol{\xi}$ (cf. (7.8)), is added to (7.24) and the constraints are updated. The optimization problem that allows classification errors is then

$$\begin{aligned}\min_{\mathbf{w},\mathbf{b},\boldsymbol{\xi}} \quad & \tfrac{1}{2}\|\bar{\mathbf{C}}\mathbf{w}\|^2 + \tfrac{1}{2}\|\mathbf{w}\|^2 + C\mathbf{e}^T\boldsymbol{\xi} \\ \text{s.t.} \quad & \bar{\mathbf{A}}\mathbf{w} + \bar{\mathbf{E}}\mathbf{b} \geq \mathbf{e} - \boldsymbol{\xi}, \\ & \boldsymbol{\xi} \geq 0,\end{aligned}$$

where C is a pre-specified trade-off parameter. The expansion for $\mathbf{w}$ given in (7.26) still holds in the linearly non-separable case and we can write the dual in the form

$$\begin{aligned}\max_{\mathbf{u}} \quad & \mathbf{e}^T\mathbf{u} - \tfrac{1}{2(k+1)}\mathbf{u}^T\bar{\mathbf{A}}\bar{\mathbf{A}}^T\mathbf{u} \\ \text{s.t} \quad & \bar{\mathbf{E}}^T\mathbf{u} = 0, \\ & 0 \leq \mathbf{u} \leq C.\end{aligned}$$

The only difference to the linearly separable case is that the dual variables are bounded by C, as was the case with the standard SVM in the linearly non-separable case (cf. constraint (7.12)).

8 Kernels

In this section we consider a situation where the two classes cannot be reasonably separated with a linear discriminant function and nonlinear discriminant function must be used. Figure 9 illustrates two linearly non-separable situations. In a) it is clear that a classifier with a linear discriminant function performs poorly while in b) the classes overlap and the optimal discriminant function is at least roughly linear. In practice, most real-world classification problems are at least linearly non-separable but, in addition to this, the optimal discriminant function is often nonlinear. Note that using a nonlinear discriminant function of course does not guarantee zero training error.

We will map the vectors $\mathbf{x}_i$, $i = 1, \ldots, n$, into a new space in the hope that the optimal separating hyperplane in new space performs better classifications than the optimal hyperplane in the original space. The mapping is $\Phi : \mathbb{R}^d \to \mathcal{H}$, where $dim(\mathcal{H}) \geq d$ and possibly $dim(\mathcal{H}) = \infty$ [14, 65, 79, 80]. More specifically the mapping that will be considered is of the form

$$\Phi(\mathbf{x}) = \left(\sqrt{\lambda_1}\psi_1(\mathbf{x}), \sqrt{\lambda_2}\psi_2(\mathbf{x}), \ldots \right),$$

where λ_i and ψ_i are the eigenvalues and the normalized eigenfunctions of an integral operator $\mathfrak{L}_K$: $f \mapsto \int K(\cdot, \mathbf{v}) f(\mathbf{v}) d\mathbf{v}$. In the SVM literature the space $\mathcal{H}$ is often called a feature space and the $\Phi(\mathbf{x}_i)$'s are called feature vectors. Calculating the feature vectors can be computationally expensive, or even impossible, if the dimension of feature space is high or infinite. It should be noted that in the SVM algorithm all the calculations involving the $\Phi(\mathbf{x}_i)$'s appear as inner products (see the next subsection). Instead of explicitly mapping the vectors into a high dimensional feature space and computing the inner product there it is under certain condition possible to use a function $K(\mathbf{u}, \mathbf{v})$ whose value directly gives the inner product between two vectors $\Phi(\mathbf{u})$ and $\Phi(\mathbf{v})$. Direct consequence is that using K the inner products can be computed roughly the same time in the feature space and in the original space. In the literature the function $K(\cdot, \cdot)$ is usually called a *kernel*.

The conditions under which a continuous and symmetric kernel $K(\mathbf{u}, \mathbf{v})$ corresponds to an inner product in some feature space is given by Mercer's theorem.

Theorem 22 (Mercer) *([34], Theorem 17). Let C be a compact subset of $\mathbb{R}^n$. Suppose K is a continuous and symmetric function such that the integral*

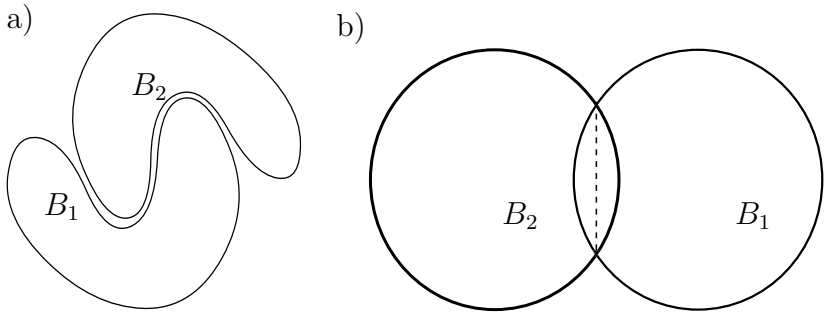


Figure 9: In a) the optimal discriminant function is nonlinear while the optimal classifiers has no errors. In b) the optimal discriminant function is linear while the classes overlap and thus the optimal classifier in not error-free.

operator $\mathfrak{L}_K : L_2(C) \to L_2(C)$,

$$(\mathfrak{L}_K f)(\cdot) = \int_C K(\cdot, \mathbf{v}) f(\mathbf{v}) d\mathbf{v},$$

is positive, that is,

$$\int_{C\times C} K(\mathbf{u}, \mathbf{v}) f(\mathbf{u}) f(\mathbf{v}) d\mathbf{u} d\mathbf{v} \geq 0, \tag{8.1}$$

for all $f \in L_2(C)$. *We can then expand* $K(\mathbf{u}, \mathbf{v})$ *in a uniformly convergent series on* $C \times C$ *in terms of eigenfunctions* $\psi_j \in L_2(C)$ *of* $\mathfrak{L}_K$, $\|\psi_j\|_{L_2} = 1$, *and the associated positive eigenvalues* $\lambda_j > 0$ *as*

$$K(\mathbf{u}, \mathbf{v}) = \sum_{j=1}^{N_F} \lambda_j \psi_j(\mathbf{u}) \psi_j(\mathbf{v}), \tag{8.2}$$

where $N_F \leq \infty$ *stands for the dimension of the feature space.*

Proof For a proof we refer to [34].

The feature space $\mathcal{H}$ can be taken to be the Hilbert space l^2 of square summable sequences of real numbers because $\|\Phi(\mathbf{x})\|^2 = \sum_{j=1}^{N_F} [\sqrt{\lambda_j}\psi_j(\mathbf{x})]^2 = K(\mathbf{x}, \mathbf{x}) < \infty$, by definition.

For any finite subset of C the following condition is equivalent to condition (8.1): K is continuous and symmetric and for all finite $\{\mathbf{u}_1, \ldots, \mathbf{u}_n\} \in C$ the matrix

$$\mathbf{K} = (K(\mathbf{u}_i, \mathbf{u}_j))_{i,j=1}^n$$

is positive semi-definite [60].

It should be pointed out that the class of mathematical functions, that can be used as kernels is very large, see for example [36, 37].

8.1 Kernels in the SVM algorithm

Using a kernel K in the nonlinearly separable case, all the calculations of subsections 7.1 and 7.2 hold when the inner products $\mathbf{x}_i^T\mathbf{x}_j$ are replaced by $K(\mathbf{x}_i, \mathbf{x}_j)$. It follows that the dual object function in the linearly separable case is

$$W(\boldsymbol{\alpha}) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j) \tag{8.3}$$

which is to be maximized under the constraints

$$\alpha_i \geq 0, \ i = 1, \ldots, n,$$
$$\sum_{i=1}^{n} \alpha_i y_i = 0.$$

It is straightforward to generalize the above optimization problem to the case where errors are present. In this case multipliers α_i also have upper bound, $C \geq \alpha_i \geq 0$, $i = 1, \ldots, n$. Generalization of the multiclass SVM to nonlinear discriminant functions is analogous: one replaces $\mathbf{x}_i^T\mathbf{x}_i$ by $K(\mathbf{x}_i, \mathbf{x}_i)$.

Note that since the optimal $\mathbf{w}$ is a linear combination of the $\mathbf{x}_i$'s, $i \in \text{SV}$, the final classifier $f(\mathbf{x})$ is expressed in terms of $K(\mathbf{x}_i, \mathbf{x})$,

$$f(\mathbf{x}) = \text{sgn}\left(\sum_{SV} y_i \hat{\alpha}_i K(\mathbf{x}_i, \mathbf{x}) + \hat{b} \right),$$

while the optimal $\hat{\mathbf{w}}$ usually cannot be expressed in a closed form using (7.6). Thus, we still try to find the optimal hyperplane but now in the feature space. Moreover, the final decision function is linear in the feature space but nonlinear in the original space unless $K(\mathbf{u}, \mathbf{v}) = \mathbf{u}^T\mathbf{v}$.

Some authors use the term SVM only when some nontrivial kernel is used and give no specific name to the learning algorithm which finds the optimal

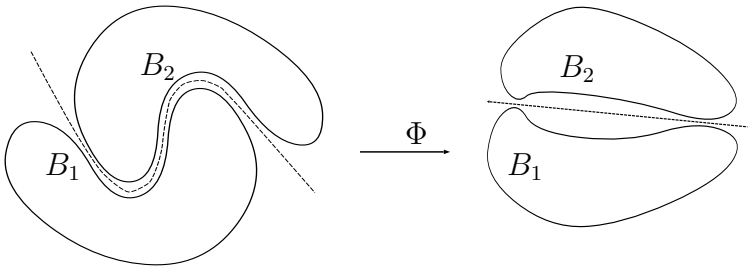


Figure 10: Using a kernel to map the original nonlinearly separable data into a linearly separable data in the feature space. The optimal discriminant function is depicted by the dashed curve.

separating hyperplane only in the original space. Since the mapping can also be the identity map, we use term SVM also when the optimization is done in the original space.

8.2 Some well-known kernels

There are also some difficulties associated with the mapping Φ and the kernel K. Usually it is very difficult or even impossible to find a mapping that corresponds to a particular kernel and, vice versa, it is difficult to find a kernel that corresponds to some particular mapping. The selection of a kernel function is an important problem in applications although there is no theory to tell which kernel to use. Moreover, it can be difficult to check that some particular kernel satisfies Mercer's conditions, since these conditions must hold for every $f \in L_2(C)$. In the following some well-known and widely used kernels are presented. Selection of the kernel, perhaps from among the presented kernels, is usually based on experience and knowledge about the classification problem at hand, and also on theoretical considerations. The problem of choosing the kernel on the basis of theoretical considerations is discussed in subsection 8.3.

8.2.1 Polynomial kernel

The polynomial kernel of degree q is of the form

$$K(\mathbf{u}, \mathbf{v}) = (\mathbf{u}^T\mathbf{v} + c)^q, \tag{8.4}$$

where c is some non-negative constant, usually $c = 1$. Using of a generalized inner-product instead of the standard inner-product is proposed in [17]. In

this case the kernel is

$$K(\mathbf{u}, \mathbf{v}) = \left(\sum_i \frac{u_i v_i}{\sigma_i} + c \right)^q,$$

where the vector $\boldsymbol{\sigma}$ is such that the function satisfies the Mercer's condition. When c is positive the kernel is called inhomogeneous and, correspondingly, homogeneous when $c = 0$. The inhomogeneous kernel avoids problems with the Hessians becoming zero in numerical calculations. To prove that this kernel is a Mercer kernel, one must show that

$$\int \left(\sum_{i=1}^{d'} u_i v_i \right)^q f(\mathbf{u}) f(\mathbf{v}) d\mathbf{u} d\mathbf{v} \geq 0, \tag{8.5}$$

where d' is the dimension of the vectors and where we have for simplicity set $c = 0$. Every element in the multinomial term $\left(\sum_{i=1}^{d'} u_i v_i \right)^q$ can be written as $\frac{q!}{p_1! p_2! \cdots p_{d'}!} u_1^{p_1} v_1^{p_1} u_2^{p_2} v_2^{p_2} \cdots u_{d'}^{p_{d'}} v_{d'}^{p_{d'}}$, where $\sum_{i=1}^{d'} p_i = q$, $p_i \geq 0$, and thus every element in (8.5) is of the form

$$\frac{q!}{p_1! p_2! \cdots p_{d'}!} \int u_1^{p_1} u_2^{p_2} \cdots u_{d'}^{p_{d'}} v_1^{p_1} v_2^{p_2} \cdots v_{d'}^{p_{d'}} f(\mathbf{u}) f(\mathbf{v}) d\mathbf{u} d\mathbf{v}.$$

Factorizing this gives

$$\frac{q!}{p_1! p_2! \cdots p_{d'}!} \left(\int u_1^{p_1} u_2^{p_2} \cdots u_{d'}^{p_{d'}} f(\mathbf{u}) d\mathbf{u} \right)^2 \geq 0,$$

which completes the proof.

8.2.2 Sigmoid-function

The sigmoid kernel is of the form

$$K(\mathbf{u}, \mathbf{v}) = \tanh(v(\mathbf{u}^T \mathbf{v}) + a), \tag{8.6}$$

and it satisfies the Mercer condition only for certain values of the parameters v and a. Currently there are no theoretical results on the parameter values that satisfy the Mercer condition and proper values are found by empirical means. When the sigmoid kernel is used with the SVM one can regard it as a two-layer neural network. In two-layer neural network the vector $\mathbf{x}$ is mapped by the first layer into the vector $\mathbf{F} = (F_1, \ldots, F_N)$, where

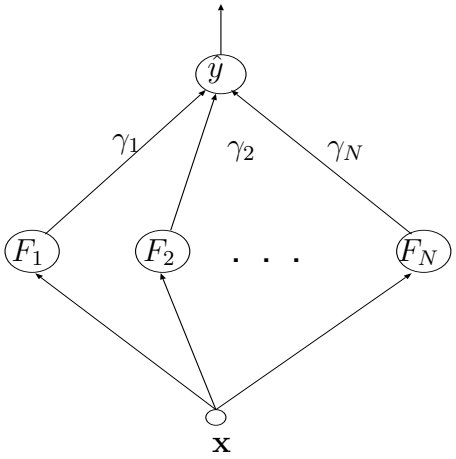


Figure 11: A two-layer neural network. Outputs of the first layer are of the form $F_{n'} = \tanh(v(\mathbf{z}_{n'}^T\mathbf{x}) + a), \quad n' = 1, \ldots, N$. Output of the network is $\hat{y} = \text{sgn}\left(\sum_{n'=1}^{N} y_{n'}\gamma_{n'}F_{n'} - b\right)$.

$F_{n'} = \tanh(v(\mathbf{z}_{n'}^T\mathbf{x}) + a)$, $n' = 1, \ldots, N$, and the dimension of this vector is called the number of hidden units. In the second layer the sign of weighted sum of elements of $\mathbf{F}$ is calculated by using weights $\gamma_{n'}$, see Figure 11 for an illustration. The major difference between the SVM and a two-layer neural network is in different optimization criterion: in the SVM the goal is to find the optimal hyperplane that maximizes the margin (in the feature space), while in a two-layer neural network the optimization criterion usually is to minimize the empirical risk associated with some loss function, typically the mean squared error. It should be pointed out that quite often in neural networks the optimal network architecture is unknown. If one uses the sigmoid-function with SVM then such problems are avoided, since the number of hidden units ($\hat{N} = |\{\mathbf{x}_i \in X | \hat{\alpha}_i > 0\}|$, the number of support vectors), the centers in the hidden layer (weights $\hat{\mathbf{z}}_i = \mathbf{x}_i \mathbb{1}(\hat{\alpha}_i > 0)$, that is, support vectors) and the vector of weights in the output layer ($\hat{\gamma}_{n'} = \hat{\alpha}_{n'} y_{n'}$) are all determined automatically in the linearly separable case.

8.2.3 Radial basis function

The Gaussian kernel, known also as the radial basis function, is of the form

$$K(\mathbf{u}, \mathbf{v}) = \exp\big(-\frac{\|\mathbf{u} - \mathbf{v}\|^2}{\sigma^2}\big), \tag{8.7}$$

where σ stands for a window width. It is also possible to have different window widths for different vectors, that is, to use a vector $\boldsymbol{\sigma}$ [17]. In the

following it is shown that (8.7) is a kernel. We can factorize the Gaussian function (8.7) as

$$\exp\left(\frac{2\mathbf{u}^T\mathbf{v}}{\sigma^2}\right)\exp\left(\frac{-\mathbf{u}^T\mathbf{u}}{\sigma^2}\right)\exp\left(\frac{-\mathbf{v}^T\mathbf{v}}{\sigma^2}\right). \tag{8.8}$$

The first factor $\exp\left(\frac{2\mathbf{u}^T\mathbf{v}}{\sigma^2}\right)$ is kernel since it can be approximated arbitrarily well by polynomials with positive coefficients and it was already proved that a function of the form (8.4) is a kernel. To show that the last two factors of (8.8) together form a kernel, suppose that we have a finite set of points $U = \{\mathbf{u}_1, \ldots, \mathbf{u}_n\}$. Now, by definition, the positive semidefiniteness condition of the kernel matrix is

$$\begin{aligned}\sum_{i,j=1}^{n}\beta_i\beta_j K(\mathbf{u}_i,\mathbf{u}_j) &= \sum_{i=1,j}^{n}\beta_i\beta_j\exp\left(\frac{-\mathbf{u}_i^T\mathbf{u}_i}{\sigma^2}\right)\exp\left(\frac{-\mathbf{u}_j^T\mathbf{u}_j}{\sigma^2}\right)\\ &= \left(\sum_{i=1}^{n}\beta_i\exp\left(\frac{-\mathbf{u}_i^T\mathbf{u}_i}{\sigma^2}\right)\right)^2 \geq 0,\end{aligned}$$

for all $\beta_1, \ldots, \beta_n$, and thus $\exp\left(-\mathbf{u}_i^T\mathbf{u}_i/\sigma^2\right)\exp\left(-\mathbf{u}_j^T\mathbf{u}_j/\sigma^2\right)$ is a kernel. It is now enough to show that the product of kernels is a kernel, but this is clear, because product of positive semidefinite matrices is positively semidefinite. This completes the proof.

The final classifier with this kernel is

$$f_{SVM}(\mathbf{x}) = \operatorname{sgn}\left(\sum_{i=1}^{n} y_i\hat{\alpha}_i\exp\left(-\frac{\|\mathbf{x}_i-\mathbf{x}\|^2}{\sigma^2}\right)+b\right), \tag{8.9}$$

which has almost the same form as the well-known RBF-network [10], where the decision function is

$$f_{RBF}(\mathbf{x}) = \operatorname{sgn}\left(\sum_{i=1}^{N}\alpha_i\exp\left(-\frac{\|\mathbf{c}_i-\mathbf{x}\|^2}{\sigma^2}\right)+b\right), \tag{8.10}$$

and N is the number of centers $\mathbf{c}_i$ employed. Using the Gaussian kernel (8.7) with the SVM, the number of basis functions (the number of support vectors), the centers (the $\mathbf{x}_i$'s corresponding to the nonzero Lagrangian multipliers, i.e., support vectors) and the weights in the output layer ($y_i\hat{\alpha}_i$) of the RBF-network are all determined automatically. Furthermore, in some situations it can be useful to use the centers given by the SVM in an RBF-network if no other information is available for an optimal placing of the centers [63, 80]. It should be pointed out again that the RBF-SVM and the RBF-network use the different optimization criteria.

8.3 Selecting the kernel and the parameters

When a kernel is used it is often unclear what the properties of the mapping and the feature space are. It is always possible to make a mapping into a potentially very high dimensional space and to produce a classifier with no classification errors on the training set. However, then the performance of the classifier can be poor. On the other hand, it is possible that a classifier with an infinite dimensional feature space performs well. Thus, the dimension of the feature space is not the essential quantity when choosing the right kernel. This is opposite to the usual curse of dimensionality problem (cf. [24]).

One could try to select the kernel on the basis of some functional analytic criteria, say using covering numbers that were shown to be important in the upper bounds of the error probability in the previous sections. Many other theoretical results based on functional analysis are given in [1, 13, 20, 62, 66, 67, 77, 89, 90].

On the other hand, Vapnik argued in [80] that on the basis of experiments the choice between the kernels presented in the previous subsection does not make a big difference in empirical performance. The more important, and usually the more difficult problem is the selection of the parameters of kernel function. This problem could be solved using a (leave-one-out) cross-validation procedure but quite often with real-world sized training sets this is computationally very costly or even impossible since the quadratic optimization problem of the SVM algorithm is computationally rather demanding. One approach would be to use the linear approximation of subsection 7.4 in the cross-validation to make the parameter selection faster. Recently, some more advanced approaches have been proposed. In [16, 17] various kernel-dependent upper bounds are given on the leave-one-out error of the SVM. These upper bounds are then differentiated with respect to kernel parameters and then, by using some optimization algorithm (for example Newton-Rhapson -method), the best values for a kernel are found. In the following some of these upper bounds are given.

Jaakkola-Haussler bound
Jaakkola and Haussler [37] proposed the bound

$$\frac{1}{n}\sum_{i=1}^{n}\Psi\left(\hat{\alpha}_i K(\mathbf{x}_i,\mathbf{x}_i)-1\right), \tag{8.11}$$

where $\hat{\alpha}_i$ is the optimal value of α_i trained with the whole training set and Ψ is a step-function: $\Psi(t)=1$, if $t>0$ and $\Psi(t)=0$ otherwise.

Support vector count
Vapnik ([80], Theorem 10.5) has proved the following upper bound on the error probability of the SVM

$$\frac{|\{\mathbf{x}_i \in X : \hat{\alpha}_i > 0\}|}{n}, \tag{8.12}$$

that is, the fraction of the support vectors in the training set. Note the direct connection to Theorem 21.

Radius-margin bound
In [17] the upper bound

$$\frac{nR^2}{\|\mathbf{w}\|^2}, \tag{8.13}$$

is given for the SVM with no training errors, where R is the minimal enclosing sphere of the collection of the vectors $\Phi(\mathbf{x}_i)$, $i = 1, \ldots, n$. This upper bound suggests that one should try to find the minimal enclosing sphere in the feature space. The radius R is easily found from the minimization problem

$$\min R^2 \tag{8.14}$$

$$\text{subject to} \quad \|\Phi(\mathbf{x}_i) - \mathbf{C}\| \leq R^2 \qquad \forall \mathbf{x}_i \in X, \tag{8.15}$$

where $\mathbf{C}$ is the center of the minimal enclosing sphere in feature space that should also be minimized. The solution is easily found by the Lagrangian method (see the Appendix).

Some of the given upper bounds are not differentiable with respect to the parameters, but it is possible to use some differentiable function which approximates the upper bounds well. A more detailed presentation on finding the optimal kernel parameters using different upper bounds is given in [16, 17].

The question of the trade-off parameter of (7.10) still remains quite an open issue. Some propositions have been made to answer the question of a proper choice of C [16, 86, 87]. One could also choose C subjectively because in some sense the value of C determines the trade-off between how much one is giving weight to the minimization of estimation error and how much to the minimization of approximation error. Setting $C = \infty$ corresponds to minimizing only the empirical error and setting C to some small positive value leads

to a large margin and small approximation error but possibly large training error. A recent proposition given in [68] suggests that one should consider the problem

$$\begin{aligned} \min_{\mathbf{w},b,\boldsymbol{\xi}} \quad & \tfrac{1}{2}\|\mathbf{w}\|_2^2 + \nu\rho + \textstyle\sum_{i=1}^n \xi_i & \\ \text{s.t.} \quad & y_i(\mathbf{w}^T\mathbf{x}_j + b) \geq \rho - \xi_i, & i = 1, \ldots, n \\ & \xi_i \geq 0, & i = 1, \ldots, n \\ & \rho \geq 0. & \end{aligned}$$

It can be shown [68] that ν is then a lower bound on the fraction of support vectors in the training set.

9 Conclusions

As already pointed out, the SVM-algorithm usually works very well in practice but the theoretical reason for this has been somewhat unclear. In [79] Vapnik who introduced the SVM gives two theoretical motivations. The first one is based on the small number of support vectors which corresponds to the compression scheme (cf. Theorem 21). The second argument is based on the following Theorem proved in [80].

Theorem 23 *([80], Theorem 10.3) The VC-dimension of the class $\mathcal{C}'$ of canonical hyperplanes of the form (7.2) defined on $\mathbb{R}^d$ and satisfying the constraints*

$$\|\mathbf{x}_i - \hat{\mathbf{C}}\| \leq R, \qquad \forall \mathbf{x}_i \in \mathbb{R}^d \tag{9.1}$$

$$\|\mathbf{w}\| \leq A \tag{9.2}$$

satisfies the inequality

$$V_{\mathcal{C}'} \leq \min([R^2A^2], n) + 1 \tag{9.3}$$

where $[\cdot]$ denotes the integer part of the argument and $\hat{\mathbf{C}}$ is the center of the minimal enclosing sphere containing all the training vectors $\mathbf{x}_i$.

Taking this theorem into account with structural risk minimization suggests that one should maximize the minimum margin of the classifier. But the above theorem can be used only after seeing the data and using this theorem together with SRM violates the assumption of a priori hierarchy of classes. Moreover, the bounds based on the VC-theory and the theory of large margin classifiers presented in subsections 3.3 and 5.2 cannot completely motivate

the SVM since they are based on a priori assumption of a classifier. Nevertheless the bounds given by the VC-theory and the theory of large margin classifiers are interesting and should not by any means be ignored. It should just borne in mind that they have their limitations from a practical point of view. Finally, the luckiness framework of the data dependent structural risk minimization principle presented in Section 6 gives the justification for maximizing the minimum the margin. This framework can also be used to give bounds similar to the VC-theory and the theory of large margin classifiers. Since the luckiness framework also includes the compression scheme it can be said that it is this theory that justifies the use of the SVM and explains its performance. At the moment the luckiness framework has two drawbacks, the first one being that the bounds given by data dependent structural risk minimization are still quite loose. However, some tighter bounds are already being introduced. The other drawback is the assumption of zero training error but we are optimistic that this problem will be solved in the near future.

The fact that there are two different quantities that provide different kinds of bounds leads to the question as to which one to use in the SVM context. In other words, should one maximize the minimal margin or minimize the number of support vectors? There is no simple solution to this question since both approaches provide good bounds of which neither is generally better than the other. The usual criterion is the margin, perhaps because it was presented earlier than the optimization problem minimizing the number of support vectors. Furthermore, the optimization problem (7.13) is only an approximation of the true optimization problem minimizing the number of support vectors.

One important drawback of the SVM is that it is computationally demanding because of the quadratic optimization problem involved. Fortunately, there are very good optimization packages available, such as MINOS, LOQO and CPLEX. Nevertheless, working with large datasets can be very time consuming, even with state-of-the-art optimization packages and therefore the number of papers on memory handling and practical questions of solving the optimization problem is increasing. For excellent papers on this issue see [38, 39, 54–56].

Support vector machines have shown their great promise in many different areas and in some cases they have outperformed other methods. There is also an increasing number of modifications of the SVM, one of the most important generalization being the use support vector methodology in regression, see [74] e.g. for a very good presentation. The SVMs have also been

especially tailored to solve successfully various practical problems, ranging from economics to genetics, see for example [21] and especially the references therein.

A Appendix1: Optimization

Suppose that we have an optimization problem

$$\min f(\mathbf{x}) \tag{A.1}$$

$$\text{subject to} \quad \mathbf{x} \in \{\mathbf{x} \mid c_i(\mathbf{x}) \geq 0, \quad i = 1, 2, \ldots, n\}, \tag{A.2}$$

where the functions c_i are concave on $\mathbb{R}^d$, that is, for all c_i, $i = 1, \ldots, n$,

$$c_i\left((1-\theta)\mathbf{x}_0 + \theta\mathbf{x}_1\right) \geq (1-\theta)c_i(\mathbf{x}_0) + \theta c_i(\mathbf{x}_1), \quad \forall\theta \in [0,1] \quad \mathbf{x}_0, \mathbf{x}_1 \in \mathbb{R}^d.$$

We use term *objective function* for the function f to be minimized. Since the functions c_i are concave on $\mathbb{R}^d$ they define a convex region [26], that is, a region K for which it holds that for all $\mathbf{x}_0$, $\mathbf{x}_1 \in K \subset \mathbb{R}^d$, $\theta \in [0,1]$ it follows that $\mathbf{x}_\theta \in K$, where

$$\mathbf{x}_\theta = (1-\theta)\mathbf{x}_0 + \theta\mathbf{x}_1.$$

The linear combination $\mathbf{x}_\theta$ of $\mathbf{x}_0$ and $\mathbf{x}_1$ is called a *convex combination*. Suppose now that f also is a convex function on

$$K = \{\mathbf{x} \mid c_i(\mathbf{x}) \geq 0, \quad i = 1, 2, \ldots, n\},$$

that is,

$$f(\mathbf{x}_\theta) \leq (1-\theta)f(\mathbf{x}_0) + \theta f(\mathbf{x}_1), \quad \forall\theta \in [0,1], \quad \mathbf{x}_0, \mathbf{x}_1 \in K.$$

The lagrangian function $\mathcal{L}(\mathbf{x}, \boldsymbol{\lambda})$ of the optimization problem (A.1) with the constraints (A.2) is defined as

$$\mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}) = f(\mathbf{x}) - \sum_{i=1}^{n} \lambda_i c_i(\mathbf{x}), \tag{A.3}$$

where $\lambda_i \in \mathbb{R}$ are called Lagrangian multipliers. The motivation for studying the Lagrangian function becomes clear from the following theorem of Karush and Kuhn-Tucker.

Theorem 24 (Karush-Kuhn-Tucker) *If $\mathbf{x}^*$ is a local minimizer of the problem (A.1) and the c_i's are C^1 functions, then there exist Lagrange multipliers $\boldsymbol{\lambda}^*$ such that $\mathbf{x}^*, \boldsymbol{\lambda}^*$ satisfy the following system*

$$\begin{aligned} \nabla_{\mathbf{x}}\mathcal{L}(\hat{\mathbf{x}}, \hat{\boldsymbol{\lambda}}) &= 0, \\ c_i(\hat{\mathbf{x}}) &\geq 0, \\ \hat{\lambda}_i &\geq 0, \\ \hat{\lambda}_i c_i(\hat{\mathbf{x}}) &= 0. \end{aligned}$$

Proof. Proof is given in [26].

The last condition of this theorem is usually called the KKT complementary condition. In this theorem the objective function doesn't have to be convex. If the objective is convex the following theorem can be used. This theorem also states an important property of a convex optimization problem.

Theorem 25 *Every local solution* $\mathbf{x}^*$ *to a convex optimization problem is a global solution and the set of global solutions is convex.*

Proof. Proof is given in [26].

A.1 The optimization problem of the SVM

Now we move to the optimization problem that arises in defining the optimal separating hyperplane of the SVM. Recall from Section 7 the optimization problem with linearly separable data:

$$\begin{aligned} \min_{\mathbf{w},b} \quad & \tfrac{1}{2}\|\mathbf{w}\|^2 \\ \text{s.t.} \quad & y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1, \quad i = 1, \dots, n. \end{aligned} \tag{A.4}$$

In the following we will in particular consider the optimization problem in the linearly non-separable case since it is more general problem. The optimization problem in this case is

$$\begin{aligned} \min_{\mathbf{w},b,\boldsymbol{\xi}} \quad & \tfrac{1}{2}\|\mathbf{w}\|^2 + C\textstyle\sum_{i=1}^n \xi_i \\ \text{s.t.} \quad & y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i, \quad i = 1, \dots, n, \\ & \xi_i \geq 0, \qquad\qquad i = 1, \dots, n. \end{aligned} \tag{A.5}$$

The Lagrangian $\mathcal{L}(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta})$ of this problem is

$$\frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^n \xi_i - \sum_{i=1}^n \alpha_i(y_i((\mathbf{w}^T\mathbf{x}_i) + b) - 1 + \xi_i) - \sum_{i=1}^n \beta_i\xi_i, \tag{A.6}$$

where $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are vectors of Lagrangian multipliers. The Karush-Kuhn-Tucker conditions of Theorem 24 are

$$\begin{aligned}
\nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w}, b, \boldsymbol{\alpha}, \boldsymbol{\xi}, \boldsymbol{\beta}) = \mathbf{w} - \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i &= 0, && \text{(A.7)}\\
\nabla_{\boldsymbol{\xi}}\mathcal{L}(\mathbf{w}, b, \boldsymbol{\alpha}, \boldsymbol{\xi}, \boldsymbol{\beta}) = C\mathbf{e} - \boldsymbol{\alpha} - \boldsymbol{\beta} &= 0, && \text{(A.8)}\\
\frac{\partial \mathcal{L}(\mathbf{w}, b, \boldsymbol{\alpha}, \boldsymbol{\xi}, \boldsymbol{\beta})}{\partial b} = -\boldsymbol{\alpha}^T \mathbf{y} &= 0, && \text{(A.9)}\\
\alpha_i(y_i((\mathbf{w}^T \mathbf{x}_i) + b) - 1 + \xi_i) &= 0, && \text{(A.10)}\\
\beta_i \xi_i &= 0, && \text{(A.11)}\\
\alpha_i &\geq 0, && \text{(A.12)}\\
\beta_i &\geq 0. && \text{(A.13)}
\end{aligned}$$

The following optimization problem is called the *dual* problem of (A.5) while the problem (A.5) is called the *primal* problem:

$$\begin{aligned}
&\max_{\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\beta}, \boldsymbol{\alpha}} \mathcal{L}(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta}) && \text{(A.14)}\\
&\text{s.t. } (A.8), (A.9), (A.12), (A.13).
\end{aligned}$$

Now, using (A.8) and (A.9) it is easy to show that the dual does not involve b, $\boldsymbol{\xi}$ and $\boldsymbol{\beta}$. Thus, the only variables in the dual are $\mathbf{w}$ and $\boldsymbol{\alpha}$.

The next theorem which is essentially an application of Dorn's duality theorem [48] justifies the approach of solving the dual problem instead of the primal.

Theorem 26 *If $(\hat{\mathbf{w}}, \hat{\boldsymbol{\alpha}})$ solves the dual problem (A.14) then there exists a $\hat{\boldsymbol{\xi}}$ such that $(\hat{\mathbf{w}}, \hat{\boldsymbol{\xi}})$ solves the primal problem (A.5), and $\hat{\boldsymbol{\alpha}}$ is the corresponding Lagrange multiplier vector.*

Proof Proof can be found in [48].

In this theorem the variables b and $\boldsymbol{\beta}$ are omitted since they can be uniquely determined from the other variables and from the KKT complimentary condition (A.10).

Note that when using (A.7),(A.8) and (A.9), we can write the dual in the following form which includes only $\boldsymbol{\alpha}$:

$$\begin{aligned}
\max_{\boldsymbol{\alpha}} \quad & \textstyle\sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j (\mathbf{x}_i^T \mathbf{x}_j) && \text{(A.15)}\\
\text{s.t.} \quad & 0 \leq \alpha_i \leq C, \qquad i = 1, \ldots, n,\\
& \textstyle\sum_{i=1}^{n} \alpha_i y_i = 0.
\end{aligned}$$

From the solution $\hat{\boldsymbol{\alpha}}$ of this problem we get

$$(A.7) \quad \Rightarrow \quad \hat{\mathbf{w}} = \sum_{i=1}^{n} \hat{\alpha}_i y_i \mathbf{x}_i \tag{A.16}$$

$$(A.8) \quad \Rightarrow \quad \hat{\boldsymbol{\beta}} = C - \hat{\boldsymbol{\alpha}}$$

$$(A.8), (A.10), (A.11) \quad \Rightarrow$$

$$\hat{b} = -\frac{1}{2|\{i \mid 0 < \hat{\alpha}_i < C\}|} \sum_i \hat{\mathbf{w}}^T [\mathbf{x}_i \mathbb{1}(i \in \mathcal{S}_1) + \mathbf{x}_i \mathbb{1}(i \in \mathcal{S}_2)], \tag{A.17}$$

where $\mathcal{S}_1 = \{i | 0 < \hat{\alpha}_i < C, y_i = -1\}$ and $\mathcal{S}_2 = \{i | 0 < \hat{\alpha}_i < C, y_i = 1\}$. Finally,

$$(A.8), (A.10), (A.11), (A.16), (A.17) \Rightarrow$$

$$\hat{\xi}_i = \begin{cases} 0 & : \quad 0 \leq \hat{\alpha}_i < C \\ -y_i(\hat{\mathbf{w}}^T \mathbf{x}_i + \hat{b} - 1) & : \quad \hat{\alpha}_i = C \end{cases}.$$

On the basis of the last equation one can see that the vectors $\mathbf{x}_i$ with $\xi_i > 0$ are support vectors.

A.2 Existence and uniqueness

Next, we will focus more thoroughly on the question of existence and uniqueness of the solution of the optimization problems (A.4) and (A.5). It is clear that in the linearly separable case (A.4), the target function of the primal is strictly convex on $\{\mathbf{w} \mid y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1 \ , \forall i\}$. Furthermore, the region is convex since it is an intersection of convex regions. Thus, if the primal is solvable it has a unique solution, where by solvable we mean that there exists a solution for the optimization problem. It is clear that the primal of a linearly separable optimization problem is not solvable if and only if the data are linearly nonseparable. In this case one can consider instead the linearly non-separable optimization problem (A.5) which is solvable for all $C > 0$ [43].

However, strict convexity of the primal does not imply that the corresponding dual is strictly convex and having thus a unique solution. For example, the objective function of the dual will only be convex and thus can have multiple solutions whenever the size of the training set is greater than the dimension of the vectors in the input space [15]. The following theorem proved in [43] deals with the existence of solutions to problems (A.4) and (A.5).

Theorem 27 *For problems (A.5) and (A.4), only one of the following two alternatives are possible:*

(A.4) is solvable and the corresponding dual is solvable
(A.4) is infeasible and the corresponding dual is unbounded

If $C > 0$ then (A.5) and the corresponding dual are both solvable.

Now, since there exists a solution for (A.5), we move to the question of uniqueness of the solution. In the case of linearly non-separable data it is possible to have the following four different situations

1. Both the primal and the dual have unique solutions.
2. The solution of the primal is unique but the solution of the dual is not.
3. The solution of the primal is not unique but the solution of the dual is unique.
4. Both the primal and the dual do not have unique solutions.

In the case of linearly separable data only (1) and (2) are possible. In case (2) the expansion of the optimal weight vector (A.7) still holds - all the different solutions $\hat{\boldsymbol{\alpha}}$ give exactly the same $\hat{\mathbf{w}}$. If in (A.5) one selects $p > 1$ in the functional of errors

$$\sum_{i=1}^{n} \xi^p$$

instead of the standard selection $p = 1$, then the corresponding primal is strictly convex and thus has a unique solution because the linear constraints create a convex region. For $p = 1$ a necessary condition for the solution of the primal to be non-unique is that

$$\sum_i \mathbb{1}(i \in \text{SV} \mid y_i = -1) = \sum_i \mathbb{1}(i \in \text{SV} \mid y_i = 1),$$

that is, the number of support vectors from both classes should be equal [15].

Next we will consider the problem (A.5) and assume that in case $p = 1$ we have two solutions $\mathbf{w}_1$ and $\mathbf{w}_2$. Since $\mathbf{w}^T\mathbf{w}/2$ is strictly convex we find that $\mathbf{w}_1 = \mathbf{w}_2$. Now it follows that the solution to the primal is not unique if and only if $\hat{b}$ is not unique. In other words, the optimal weight vector $\mathbf{w}$ is

always unique but the whole solution $(\hat{\mathbf{w}}, \hat{b}, \hat{\boldsymbol{\xi}})$ of the primal is not necessarily so. From (A.17) one can see that this is the case for example if the set of indices $\mathcal{S}_1 \bigcup \mathcal{S}_2$ is empty. This case is studied in [15]. In general, when $\hat{b}$ is not unique one could try to find a solution that minimizes the Bayes error. Another approach is to set $\hat{b}(1) = \lim_{p \to 1+} \hat{b}(p)$, where $\hat{b}(p)$ is the optimal solution with parameter p.

Let us investigate briefly the connection between the solutions of (A.4) and (A.5). The following theorem proved in [43] gives an interesting result when the trade-off parameter C is large enough.

Theorem 28 *If (A.4) is solvable (and thus unique), then there exists a C^* such that for all $C \geq C^*$ any solution $(\hat{\mathbf{w}}, \hat{b})$ of (A.4) is a solution of (A.5).*

Suppose now that the previous theorem holds for some C^* and $(\hat{\mathbf{w}}, \hat{b})$. It follows from previous theorem that $(\hat{\mathbf{w}}, \hat{b}, 0)$ is also an optimal solution of (A.5). Solving (A.5) with $C \geq C^*$ gives a solution $(\hat{\mathbf{w}}', \hat{b}', \hat{\boldsymbol{\xi}}')$ but since we already stated that the weight vectors are unique, then also $\hat{\mathbf{w}}' = \hat{\mathbf{w}}$. Now

$$\frac{1}{2}\hat{\mathbf{w}}'^T\hat{\mathbf{w}}' + C\sum_{i=1}^{n}\hat{\xi}_i' \leq \frac{1}{2}\hat{\mathbf{w}}^T\hat{\mathbf{w}},$$

from which it follows that $\hat{\boldsymbol{\xi}}' = 0$ and $(\hat{\mathbf{w}}', \hat{b}')$ is an optimal solution of (A.4). We can state this result in the form of following theorem.

Theorem 29 *If (A.4) is solvable (and thus unique), then there exists a C^* such that for all $C \geq C^*$ any solution $(\hat{\mathbf{w}}, \hat{b}, \hat{\boldsymbol{\xi}})$ of (A.5) satisfies $\hat{\boldsymbol{\xi}} = 0$ and $(\hat{\mathbf{w}}, \hat{b})$ is a solution of (A.4).*

Let us end this subsection by noting that when the solution of the dual is unique, one can use the following duality theorem.

Theorem 30 (Wolfe's duality theorem) *If $\hat{\mathbf{w}}$ solves the convex optimization problem (A.5), then there exists a $\hat{\boldsymbol{\alpha}}$ such that $(\hat{\mathbf{w}}, \hat{\boldsymbol{\alpha}})$ solves the corresponding dual and the optimal values of the dual and the primal are equal.*

Proof Proof can be found in [48].

This theorem essentially says that if one has found a solution for the primal one can find a solution for the dual as well. However, if the solution of the dual is not unique this theorem cannot be used in same way as Theorem 26, that is, to move back from the dual solution to the primal solution [40].

In the literature Wolfe's theorem is often used to justify the use of the dual but because there are no guarantees that the solution of the dual is unique one should use Dorn's duality theorem (Theorem 26) instead. In the SVM literature the duals of the form (A.15) are usually called Wolfe's duals but since the use of Wolfe's duality theorem is questionable we use only the term dual.

A.3 Optimization with kernels

Suppose now that we have performed some mapping Φ of the vectors $\mathbf{x}_i$, $i = 1, \ldots, n$, and therefore are performing optimization in the feature space. It is hoped that the data are linearly separable in the feature space but there are no guarantees that this is so and, therefore, we consider the approach where the variable is $\boldsymbol{\xi}$ is used. We can write the optimization problem (A.5) in the form

$$\begin{aligned} \min_{\mathbf{w},b,\boldsymbol{\xi}} \quad & \tfrac{1}{2}\langle \mathbf{w}, \mathbf{w}\rangle + C\textstyle\sum_{i=1}^{n} \xi_i \\ \text{s.t.} \quad & y_i(\langle \mathbf{w}, \Phi(\mathbf{x}_i)\rangle + b) \geq 1 - \xi_i \quad i = 1, \ldots, n \\ & \xi_i \geq 0 \qquad\qquad\qquad\quad i = 1, \ldots, n. \end{aligned} \tag{A.18}$$

where $\langle\ ,\ \rangle$ denotes the inner product of two possibly infinite dimensional vectors in the l^2-space. By using the notation $\langle\ ,\ \rangle$ it is emphasized that the inner product is in l^2-space, while the notation $\mathbf{x}^T\mathbf{x}$ is used to indicate a finite dimensional inner product. Now, since the primal is an optimization problem in a high, possibly infinite dimensional space, it is much more practical to solve the corresponding dual problem which scales in size with the number of training samples. Using the KKT conditions with the same analogy that was done in the original space we can write the dual as a function of $\boldsymbol{\alpha}$ as

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & \mathbf{e}^T\boldsymbol{\alpha} - \tfrac{1}{2}\boldsymbol{\alpha}^T\mathbf{Q}\boldsymbol{\alpha} \\ \text{s.t.} \quad & 0 \leq \alpha_i \leq C \qquad i = 1, \ldots, n, \\ & \textstyle\sum_{i=1}^{n} \alpha_i y_i = 0, \end{aligned} \tag{A.19}$$

where $\mathbf{Q}$ is a positive semi-definite matrix with $\mathbf{Q}_{ij} = y_i y_j \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j)\rangle$ and $\mathbf{e}$ is a vector of ones. It is now possible to use the kernel K to calculate the value $\langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j)\rangle$. The problem of strict convexity (and thus uniqueness of solution) of the dual reduces to problem of positive semi-definitiness of the matrix $\mathbf{Q}$. It is also possible for the dual to have a unique solution even if the matrix $\mathbf{Q}$ is only positive semi-definite [49].

References

[1] Shun-Ichi Amari and Si Wu. Improving support vector machine classifiers by modifying kernel functions. *Neural Networks*, 12(6), 1999.

[2] Martin Anthony and Peter Bartlett. *Neural Network Learning: Theoretical Foundations*. Cambridge University Press, 1999.

[3] Martin Anthony and Norman Biggs. *Computational Learning Theory*, volume 30 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 1992.

[4] Martin Anthony and John Shawe-Taylor. A sufficient condition for polynomial distribution-dependent learnability. *Discrete Applied Mathematics*, 77(1), 1997.

[5] Peter Bartlett. The sample complexity of pattern classification with neural networks: The size of the weights is more important than the size of the network. *IEEE Transactions on Information Theory*, 44(2), 1998.

[6] Peter Bartlett, Stéphane Boucheron, and Gábor Lugosi. Model selection and error estimation. In *Proceedings of 13th ACM, Conference on Learning Theory*, 2000.

[7] Peter Bartlett and John Shawe-Taylor. Generalization performance of support vector machines and other pattern classifiers. In Bernhard Schölkopf, Christopher Burges, and Alex Smola, editors, *Advances in Kernel Methods: Support Vector Learning*. MIT press, 1998.

[8] Kristin Bennet and Ayhan Demiriz. Semi-supervised support vector machines. To appear in: M. S. Kearns, S. A. Solla, and D. A. Cohn, editors, Advances in Neural Information Processing Systems 11, MIT Press, Cambridge, MA, 1999.

[9] Kristin Bennett and Erin Bredensteiner. Geometry in learning. In C. Gorini, editor, *Geometry at Work*. Mathematical Association of America, 1996.

[10] Christopher Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.

[11] Bernhard Boser, Isabelle Guyon, and Vladimir Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of 5th Annual ACM Workshop on Computational Learning Theory*, 1992.

[12] Erin Bredensteiner and Kristin Bennett. Multicategory classification by support vector machines. *Computational Optimizations and Applications*, pages 53–79, 1999.

[13] Christopher Burges. Geometry and invariance in kernel based methods. In Bernhard Schölkopf, Christopher Burges, and Alex Smola, editors, *Advances in Kernel Methods: Support Vector Learning*. MIT press, 1998.

[14] Christopher Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, Volume 2(2), 1998.

[15] Christopher Burges and David Crisp. Uniqueness of the SVM solution. In *Proceedings of 12th Conference on Neural Information Processing Systems*, 1999.

[16] Olivier Chapelle and Vladimir Vapnik. Model selection for support vector machines. *Advances in Neural Information Processing Systems*, Volume 12, 1999.

[17] Olivier Chapelle, Vladimir Vapnik, Olivier Bousquet, and Sayan Mukherjee. Choosing kernel parameters for support vector machines. to appear in Machine Learning - Special Issue on Support Vector Machines, 2000.

[18] Olivier Chapelle, Vladimir Vapnik, and Jason Weston. Transductive inference for estimating values of functions. *Advances in Neural Information Processing Systems*, Volume 12, 1999.

[19] Vladimir Cherkassky and Filip Mulier. *Learning from Data: Concepts, Theory and Methods*. John Wiley & Sons, 1998.

[20] Nello Cristianini, Colin Campbell, and John Shawe-Taylor. Dynamically adapting kernels in support vector machines. Technical Report NC2-TR-1998-017, NeuroCOLT, 1998.

[21] Nello Cristianini and John Shawe-Taylor. *An introduction to Support Vector Machines and other Kernel-Based Learning Methods*. Cambridge University Press, 2000.

[22] Luc Devroye, László Györfi, and Gábor Lugosi. *A Probabilistic Theory of Pattern Recognition*. Springer, 1996.

[23] Luc Devroye and Gábor Lugosi. Lower bounds in pattern recognition and learning. *Machine Learning*, 28(7), 1995.

[24] David Donoho. Aide memoire: High dimensional data-analysis: The curses and blessings of dimensionality, 2000. Lecture in the Conference Math Challenges of the 21st Century by The American Math. Society.

[25] André Elisseeff, Yann Guermeur, and Hélène Paugam-Moisy. Margin error and generalization capabilities of multiclass discriminant systems. Technical Report TR-051, NeuroCOLT2, 1999.

[26] Roger Fletcher. *Practical Methods of Optimization*. John Wiley & Sons, 2nd edition, 1989.

[27] Sally Floyd and Manfred Warmuth. Sample compression, learnability, and the Vapnik-Chervonenkis dimension. *Machine Learning*, 21(3), 1995.

[28] Jerome Friedman. Another approach to polychotomous classification. Technical report, Standford University, 1996.

[29] Yann Guermeur, André Elisseeff, and Hélène Paugam-Moisy. Estimating the sample complexity of multi-class discriminant model. In *Proceedings of IJCNN'99*, 1999.

[30] Yann Guermeur, André Elisseeff, and Hélène Paugam-Moisy. A new multi-class SVM based on a uniform convergence result. In *Proceedings of IJCNN'00*, 2000.

[31] Steve Gunn. Support vector machines for classification and regression. Technical report, Image Speech & Intelligent Systems Group, University of Southampton, 1998.

[32] Trevor Hastie and Robert Tibshirani. Classification by pairwise coupling. *The Annals of Statistics*, 26(2), 1998.

[33] Ralf Herbrich. *Learning Linear Classifiers - Theory and Algorithms*. PhD thesis, Berlin Technical University, 2000.

[34] Harry Hochstadt. *Integral Equations*. John Wiley & Sons, 1973.

[35] Wassily Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of American Statistical Association*, 58:13–30, 1963.

[36] Tommi Jaakkola, Mark Diekhans, and David Haussler. Using the fisher kernel method to detect remote protein homologies. In *Proceedings of Seventh Internation Conference on Intelligent Systems for Molecular Biology*, 1999.

[37] Tommi Jaakkola and David Haussler. Probabilistic kernel regression models. In *Proceedings of the 1999 Conference on AI and Statistics*, 1999.

[38] Thorsten Joachims. Making large-scale support vector machine learning practical. In Bernhard Schölkopf, Christopher Burges, and Alex Smola, editors, *Advances in Kernel Methods: Support Vector Learning*. MIT press, 1998.

[39] Linda Kaufman. Solving the quadratic programming problem arising in support vector classification. In Bernhard Schölkopf, Christopher Burges, and Alex Smola, editors, *Advances in Kernel Methods: Support Vector Learning*. MIT press, 1998.

[40] Sathiya Keerthi. On the use of Wolfe's duality theorem in support vector machines, 2000.

[41] Ulrich H.-G. Kreßel. Pairwise classification and support vector machines. In Bernhard Schölkopf, Christopher Burges, and Alex Smola, editors, *Advances in Kernel Methods: Support Vector Learning*. MIT press, 1998.

[42] Yuh-Jey Lee and Olvi Mangasarian. SSVM: A smooth support vector machine for classification. Technical Report 99-03, University of Wisconsin, 1999.

[43] Chih-Jen Lin. Formulations of support vector machines: a note from an optimization point of view. Technical report, Department of Computer Science and Information Engineering, National Taiwan University, 1999.

[44] Gábor Lugosi. Lectures is statistical learning theory, 2000. Presented in The 22nd Finnish Summer School on Probability Theory.

[45] Gábor Lugosi and Andrew B. Nobel. Adaptive model selection using empirical complexities. *Annals of Statistics*, 27(6), 1995.

[46] Gábor Lugosi and Kenneth Zeger. Nonparametric estimation via empirical risk minimization. *IEEE Transactions on information theory*, 41(3), 1995.

[47] Olvi Mangasarian. Linear and nonlinear separation of patterns by linear programming. *Operations Research*, 13:444–452, 1965.

[48] Olvi Mangasarian. *Nonlinear programming*. Tata McGraw-Hill Publishing Company, 1969.

[49] Olvi Mangasarian. Generalized support vector machines. In Alex Smola, Peter Bartlett, Bernhard Schölkopf, and Dale Schuurmans, editors, *Advances in Large Margin Classifiers*. MIT press, 2000.

[50] Olvi Mangasarian and David Musicant. Successive overrelaxation for support vector machines. *IEEE Transactions on Neural Networks*, 10(5), 1999.

[51] Davide Mattera, Francesco Palmieri, and Simon Haykin. Simple and robust methods for support vector expansions. *IEEE Transactions on Neural Networks*, 10(5), 1999.

[52] Eddy Mayoraz and Ethem Alpaydin. Support vector machines for multiclass classification. In *Proceedings of International Workshop on Artificial Neural Networks (IWANN'99)*, 1999.

[53] Manfred Opper. On the annealed vc entropy for margin classifiers: A statistical mechanics study. In Bernhard Schölkopf, Christopher Burges, and Alex Smola, editors, *Advances in Kernel Methods: Support Vector Learning*. MIT press, 1998.

[54] Edgar Osuna, Robert Freund, and Federico Girosi. An improved training algorithm for support vector machines. In *IEEE NNSP'97*, 1997.

[55] Edgar Osuna and Federico Girosi. Reducing the run-time complexity of support vector machines. In *ICPR'98*, 1998.

[56] John Platt. Fast training of support vector machines using sequential minimal optimization. In Bernhard Schölkopf, Christopher Burges, and Alex Smola, editors, *Advances in Kernel Methods: Support Vector Learning*. MIT press, 1998.

[57] John Platt, Nello Cristianini, and John Shawe-Taylor. Large margin DAGs for multiclass classification. In *Advances in Neural Information Processing Systems 12*, 2000.

[58] David Pollard. *Convergence of Stochastic Processes*. Springer-Verlag, 1984.

[59] Jorma Rissanen. Modeling by shortest data description. *Automatica*, 14, 1978.

[60] Saburou Saitoh. *Theory of Reproducing Kernels and its Applications*. Longman Scientific, 1998.

[61] Norbert Sauer. On the density of families of sets. *Journal of Combinatorial theory (A)*, 13, 1972.

[62] Bernhard Schölkopf, Sebastian Mika, Christopher Burges, Philipp Knirsch, Klaus-Robert Müller, Gunnar Rätsch, and Alex Smola. Input space vs. feature space in kernel-based methods. *IEEE Transactions on Neural Networks*, 10(5), 1999.

[63] Bernhard Schölkopf, Kah-Kay Sung, Christopher Burges, Federico Girosi, Partha Niyogi, Tomaso Poggio, and Vladimir Vapnik. Comparing support vector machines with radial basis function classifiers. *IEEE Transactions on Signal Processing*, 45(11), 1997.

[64] Bernhard Schölkopf. *Support Vector Learning*. R. Oldenbourg Verlag, 1997.

[65] Bernhard Schölkopf, Christopher Burges, and Alex Smola, editors. *Advances in Kernel Methods: Support Vector Learning*, chapter Introduction to Support Vector Learning. MIT press, 1998.

[66] Bernhard Schölkopf, John Shawe-Taylor, Alex Smola, and Robert Williamson. Generalization bounds via eigenvalues of the gram matrix. Technical Report NC-TR-99-035, NeuroCOLT, 1999.

[67] Bernhard Schölkopf, John Shawe-Taylor, Alex Smola, and Robert Williamson. Kernel-dependent support vector error bounds. In *Proceedings of ICANN'99*, 1999.

[68] Bernhard Schölkopf, Alex Smola, Robert Williamson, and Peter Bartlett. New support vector algorithms. To appear in Neural Computation.

[69] John Shawe-Taylor, Peter Bartlett, Robert Williamson, and Martin Anthony. A framework for structural risk minimization. Technical Report NC-TR-96-032, NeuroCOLT, 1996.

[70] John Shawe-Taylor, Peter Bartlett, Robert Williamson, and Martin Anthony. Structural risk minimization over data-dependent hierarchies. *IEEE Transactions on Information Theory*, 44(5), 1998.

[71] John Shawe-Taylor, Peter Bartlett, Robert Williamson, and Martin Anthony. Further results on the margin distribution. In *Proceedings of COLT99*, pages 278–285, 1999.

[72] John Shawe-Taylor and Nello Cristianini. Margin distribution bounds on generalization. Technical Report NC2-TR-1998-020, NeuroCOLT, 1998.

[73] John Shawe-Taylor and Nello Cristianini, editors. *Advances in Large Margin Classifiers*, chapter Margin Distribution and Soft Margin. MIT press, 2000.

[74] Alex Smola. *Learning with Kernels*. PhD thesis, Technische Universität Berlin, 1998.

[75] Alex Smola, Peter Bartlett, Bernhard Schölkopf, and Dale Schuurmans, editors. *Advances in Large Margin Classifiers*, chapter Introduction to Large Margin Classifiers. MIT press, 2000.

[76] Alex Smola, Olvi Mangasarian, and Bernhard Schölkopf. Sparse kernel feature analysis. Technical Report 99-04, University of Wisconsin, 1999.

[77] Alex Smola, Bernhard Schölkopf, and Klaus-Robert Müller. The connection between regularization operators and support vector kernels. *Neural Networks*, 11, 1998.

[78] Vladimir Vapnik. *Estimation Dependences Based on Empirical Data*. Springer-Verlag, 1982.

[79] Vladimir Vapnik. *The Nature of Statistical Learning Theory*. Springer, 1995.

[80] Vladimir Vapnik. *Statistical Learning Theory*. John Wiley & Sons, 1998.

[81] Vladimir Vapnik and Alexey Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications*, 16, 1971.

[82] Vladimir Vapnik and A. Lerner. Pattern recognition using generalized portrait method. *Automation and Remote Control*, 24, 1963.

[83] Vladimir Vapnik, Esther Levin, and Yann LeCun. Measuring the VC-dimension of a learning machine. *Neural Computation*, 6(5), 1994.

[84] Konstantinos Veropoulos, Colin Campbell, and Nello Cristianini. Controlling the sensitivity of support vector machines. In *Proceedings of the International Joint Conference on Artificial Intelligence*, 1999.

[85] Mathukumalli Vidyasagar. *A Theory of Learning and Generalization*. Springer, 1997.

[86] Grace Wahba. Support vector machines, reproducing kernel hilbert spaces and randomized GACV. In Bernhard Schölkopf, Christopher Burges, and Alex Smola, editors, *Advances in Kernel Methods: Support Vector Learning*. MIT press, 1998.

[87] Jason Weston. Leave-one-out support vector machines. In *Proceedings of the International Joint Conference on Artificial Intelligence*, 1999.

[88] Jason Weston and Charles Watkins. Multi-class support vector machines. Technical Report CSD-TR-98-04, Royal Holloway, University of London, 1998.

[89] Robert Williamson, Alex Smola, and Bernhard Schölkopf. Entropy numbers, operators and support vector kernels. Technical Report NC2-TR-1998-023, NeuroCOLT 2, 1998.

[90] Robert Williamson, Alex Smola, and Bernhard Schölkopf. Generalization performance of regularization networks and support vector machines via entropy numbers of compact operators. Technical Report NC2-TR-1998-019, NeuroCOLT 2, 1998.