Kernel Methods

Marco Signoretto and Johan A. K. Suykens

Katholieke Universiteit Leuven, ESAT-SCD/SISTA
Kasteelpark Arenberg 10, B-3001 Leuven (BELGIUM)
{*marco.signoretto,johan.suykens*}*@esat.kuleuven.be*

Contents

1 Introduction

This chapter addresses the study of kernel methods, a class of techniques that play a major role in machine learning and non-parametric statistics. The development of kernel-based techniques [105, 103] has been an important activity within machine learning in the last two decades. In this period, a number of powerful kernel-based learning algorithms have been proposed. Among others, these methods include Support Vector Machines (SVMs) and Least Squares SVMs, Kernel Principal Component Analysis, Kernel Fisher Discriminant Analysis and Gaussian Processes. The use of kernel methods is systematic and properly motivated by statistical principles. In practical applications, kernel methods lead to flexible predictive models that often outperform competing approaches in terms of generalization performance. The core idea consists of mapping data into a high dimensional space by means of a feature map. Since the feature map is normally chosen to be nonlinear, a linear model in the feature space corresponds to a nonlinear rule in the original domain. This fact suits many real world data analysis problems that often require nonlinear models to describe their structure.

1.1 Summary of the Chapter

In the rest of this section we present historical notes and summarize the main ingredients of kernel methods. In Section 2 we present the core ideas of statistical learning and show how regularization can be employed to devise practical learning algorithms. In Section 3 we show a selection of techniques that are representative of a large class of kernel methods; these techniques — termed primal-dual methods — use Lagrange duality as the main mathematical tools. Section 4 discusses Gaussian Processes, a class of kernel methods that uses a Bayesian approach to perform inference and learning. Section 5 recalls different approaches for the tuning of parameters. In Section 6 we review the mathematical properties of different yet equivalent notions of kernels and recall a number of specialized kernels for learning problems involving structured data. We conclude the Chapter by presenting applications in Section 7. Additional information can be found on a number of existing tutorials on SVMs and Kernel Methods, including [113, 57, 88, 60, 25].

1.2 Historical Background

The study of the mathematical foundation of kernels can be traced back at least to the beginning of the nineteenth century in connection with a general theory of integral equations [82, 86]. According to [66] the theory of Reproducing Kernel Hilbert Spaces (RKHS) was first applied to detection and estimation problems by Parzen. Properties of (reproducing) kernels are thoroughly presented in [7]. A first systematic treatment in the domain of nonparametric statistics is found in [148]. Modern mathematical reviews include [18, 99]. The first use of kernel in the context of machine learning is generally attributed to [2]. The linear support vector algorithm, which undoubtedly had a prominent role in the history of kernel methods, made its first appearance in Russia in the sixties [137, 141], in the framework of *Statistical Learning Theory* developed by Vapnik and Chervonenkis [144, 138]. Later, the idea was developed in connection to kernels by Vapnik and co-workers at AT&T labs [21, 52, 53, 32]. The novel approach was rooted on a solid theoretical foundation. Additionally, studies began to report state-of-the-art performances in a number of applications, which further stimulated research on kernel-based techniques.

1.3 The Main Ingredients

Before delving into the details we now present the general setting for statistical learning problems and then briefly review the main ingredients of a substantial part of kernel methods used in machine learning.

1.3.1 Setting for Statistical Learning

The setting of learning from examples comprises three components [139]:

1. A *generator* of input data. We shall assume that data can be represented as vectors of $\mathbb{R}^D$. These vectors are independently and identically distributed (i.i.d.) according to a fixed but unknown probability distribution $p(x)$.

2. A *supervisor* that, given input data x, returns an output value y according to a conditional distribution $p(y|x)$ also fixed and unknown. Note that the supervisor might be or might not be present.

3. A *learning machine* (or *learning algorithm*) able to choose an hypothesis:

$$f(x;\theta) \ . \tag{1}$$

Note that the hypothesis f is a function of x and depends upon a parameter vector θ belonging to a set Θ. The corresponding *hypothesis space* is then:

$$\mathscr{S} = \{f(x;\theta) \ : \ \theta \in \Theta\} \tag{2}$$

which is one-to-one with the *parameter space* Θ.

When the supervisor is present the learning problem is called *supervised*. The goal is to find that hypothesis that best mimic the supervisor response. When the supervisor is not present, the learning problem is called *unsupervised*. In this case the aim is to find an hypothesis that represent the best concise representation of the data produced by the generator.

In both cases we might be interested either on the whole domain, or we might be concerned only with a specific subset of points.

1.3.2 Feature Mapping and Kernel Trick

Kernel methods are a special class of learning algorithms. Their main idea consists of mapping input points, generally represented as elements of $\mathbb{R}^D$, into an high dimensional inner product space $\mathcal{F}$, called *feature space*. The mapping is performed by means of a *feature map* ϕ:

$$\begin{array}{rccc} \phi: & \mathbb{R}^D & \rightarrow & \mathcal{F} \\ & x & \mapsto & \phi(x) \ . \end{array} \tag{3}$$

One then approaches the learning task of interest by finding a linear model in the features space according to training points $\phi(x_1), \ldots, \phi(x_N) \in \mathcal{F}$. Since the feature map is normally chosen to be nonlinear, a linear model in the feature space corresponds to a nonlinear rule in $\mathbb{R}^D$. Alternative kernel methods differ in the way the linear model in the feature space is found. Nonetheless, a common feature across different techniques is the following. If the algorithm can be expressed solely in terms of inner products, one can restate the problem in terms of evaluations of a *kernel function*:

$$\begin{array}{rccc} k: & \mathbb{R}^D \times \mathbb{R}^D & \rightarrow & \mathbb{R} \\ & (x,y) & \mapsto & k(x,y) \end{array} \tag{4}$$

by letting

$$k(x,y) = \phi(x)^\top \phi(y) \ . \tag{5}$$

This fact, usually referred to as the *kernel trick*, is of particular interest for the cases where the feature space is infinite dimensional, which prevents direct computation in the feature space. In practice, one often starts from designing a *positive definite* kernel which guarantees the existence of a feature map ϕ satisfying (42).

1.3.3 Primal-Dual Estimation Techniques

As we shall see, an important class of machine learning methods consists of *Primal-Dual* learning techniques [120, 105, 103]. In this case one starts from a *primal model* representation of the type:

$$f(x;w,b) = w^\top \phi(x) + b = \sum_i w_i \phi_i(x) + b \ . \tag{6}$$

With reference to (1) note that here we have the tuple $\theta = (w,b)$. The *primal problem* is then a mathematical optimization problem aimed at finding optimal $w \in \mathcal{F}$ and $b \in \mathbb{R}$. Notably, the right hand side of (6) is affine in $\phi(x)$; however, since ϕ is in general a nonlinear mapping, f is a nonlinear function of x.

A first approach consists of solving the primal problem. The information content of training data is absorbed into the primal model's parameters during the procedure to find optimal parameters; the evaluation of the model (6) on new patterns (*out-of-sample extension*) does no longer require the use of training data; therefore, they can be discarded after training.

A second approach relies on Lagrangian duality arguments. In this case, the solution is represented in terms of dual variables $\alpha_1, \alpha_2, \ldots, \alpha_N$ and solved in $\alpha, \in \mathbb{R}^N$ and $b \in \mathbb{R}$. The *dual model* representation is then:

$$f(x;\alpha,b) = \sum_{n=1}^{N} \alpha_n k(x_n, x) + b \tag{7}$$

and depends upon the training patterns $x_1, x_2, \ldots, x_N \in \mathbb{R}^D$. The representation in (6) is usually called *parametric* while (7) is the *non-parametric* representation [120].

2 Foundations of Statistical Learning

In this section we briefly recall the main nomenclature and give a basic introduction on statistical learning theory. Historically, statistical learning theory constituted the theoretical foundation upon which the main methods of support vector machines were grounded. The theory is similar in spirit to a number of alternative complexity criteria and bias-variance trade-off curves. Nowadays, it remains a powerful framework for the design of learning algorithms.

2.1 Supervised and Unsupervised Inductive Learning

We have already introduced the distinction between *supervised* and *unsupervised*. Three important learning tasks are found within this categorization: *regression*, *classification* and *density estimation*. In *regression* the supervisor's response takes values in the real numbers. In *classification* the supervisor's output takes values in the discrete finite set of possible labels $\mathcal{Y}$. In particular, in the *binary classification* problem $\mathcal{Y}$ consists of two elements, e.g., $\mathcal{Y} = \{-1, 1\}$. *Density estimation* is an instance of unsupervised learning: there is no supervisor. The functional relation to be learned from examples is the probability density $p(x)$ (the generator). Supervised and unsupervised learning are concerned with estimating a function (an optimal hypothesis) over the whole input domain $\mathbb{R}^D$ based upon a finite set of training points. Therefore they are *inductive* approaches aiming at the general picture.

2.2 Semi-supervised and Transductive Learning

2.2.1 Semi-supervised Inductive Learning

In supervised learning the N training data are i.i.d. pairs

$$\{(x_1, y_1),\ (x_2, y_2),\ \ldots,\ (x_N, y_N)\} \subset \mathbb{R}^D \times \mathcal{Y} \tag{8}$$

each of which is assumed to be drawn according to

$$p(x, y) = p(y|x)p(x)\ . \tag{9}$$

There is yet another inductive approach, namely *semi-supervised learning*. In semi-supervised learning one has a set of labelled pairs (8), as in supervised learning, as well as a set of unlabelled data:

$$\{x_{N+1},\ x_{N+2},\ \ldots,\ x_{N+T}\} \subset \mathbb{R}^D \tag{10}$$

i.i.d. from the generator $p(x)$, as in unsupervised learning. The purpose is the same as in supervised learning: to find an approximation of the supervisor response. However this goal is achieved by a learning algorithm that keeps into account the additional information coming from the unlabelled data. According to [30], semi-supervised learning was popularized for the first time in the mid-1970's although similar ideas appeared earlier. Alternative semi-supervised learning algorithms differ in the way they exploit the information from the unlabeled set. One popular idea is to assume that the (possibly high-dimensional) input data lie (roughly) on a low-dimensional manifold [15, 16, 14, 112].

2.2.2 Transductive Learning

In induction one seeks for the general picture with the purpose of making out-of-sample prediction. This is an ambitious goal that might be unmotivated in certain settings. What if all the (unlabeled) data are given in advance? Suppose that one is only interested in prediction at given (finitely many) points. It is expected that this less ambitious task results in simple inference problems. These ideas are reflected in the approach found in *transductive learning* formulations. As in semi-supervised learning in transductive learning one has training pairs (58) as well as test (unlabeled) data (10). However, differently from semi-supervised learning one is only interested in making predictions at the test data (10).

2.3 Bounds on Generalization Error

Transductive and inductive inference share the common goal of achieving the lowest possible error on test data. In contrast with induction, transduction assumes that input test data are given in advance and consist of a finite discrete set of patterns drawn from the same distribution as the training set. From this perspective, it is clear that both transductive and inductive learning are concerned with *generalization*. In turn, a powerful framework to study the problem of generalization is the *Structural Risk Minimization* (SRM) principle.

2.3.1 Expected and Empirical Risk

The starting point is the definition of a *loss* $L(y, f(x;\theta))$, or discrepancy, between the response y of the supervisor to a given input x and the response $f(x;\theta)$ of the learning algorithm (that can be transductive or inductive). Formally, the generalization error can be defined as the *expected risk*:

$$R(\theta) = \int L(y, f(x;\theta))p(x, y)dxdy\ . \tag{11}$$

From a mathematical perspective the goal of learning is the minimization of this quantity. However, $p(x, y)$ is unknown and one can rely only on the sample version of (89) namely the *empirical risk*:

$$R^N_{\mathrm{emp}}(\theta) = \sum_{n=1}^{N} L(y_n, f(x_n, \theta))\ . \tag{12}$$

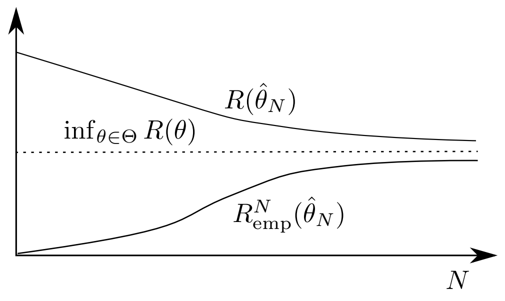


Figure 1: Consistency of ERM

A possible learning approach is based on *Empirical Risk Minimization* (ERM) and encompasses Maximum Likelihood (ML) inference [139]. It consists of finding:

$$\hat{\theta}_N := \arg\min_{\theta\in\Theta} R^N_{\text{emp}}(\theta) \ . \tag{13}$$

2.3.2 Consistency

Definition 2.1. The ERM approach is said to be *consistent* if

$$\begin{aligned} R^N_{\text{emp}}(\hat{\theta}_N) &\xrightarrow{N} \inf_{\theta\in\Theta} R(\theta) \\ R(\hat{\theta}_N) &\xrightarrow{N} \inf_{\theta\in\Theta} R(\theta) \end{aligned}$$

where $\xrightarrow{N}$ denotes convergence in probability for $N \to \infty$.

In words: the ERM is consistent if, as the number of training patterns N increases, *both* the expected risk $R(\hat{\theta}_N)$ and the empirical risk $R^N_{\text{emp}}(\hat{\theta}_N)$ converge to the minimal possible risk $\min_{\theta\in\Theta} R(\theta)$, see Figure 1 for an illustration.

It was shown in [145] that the necessary and sufficient condition for consistency is that:

$$P\left\{\sup_{\theta\in\Theta} |R(\theta) - R^N_{\text{emp}}(\theta)| \geq \epsilon\right\} \xrightarrow{N} 0, \quad \forall\epsilon > 0 \ . \tag{14}$$

In turn, the necessary and sufficient condition for (14) to hold true were established in 1968 by Vapnik [142, 143] and are based on *capacity factors*.

2.3.3 Capacity Factors

Consistency is one of the main theoretical questions in statistics. From a learning perspective, however, it does not address the most important aspect. The aspect that one should be mostly concerned with is how to control the generalization of a certain learning algorithm. Whereas consistency is an asymptotic result, we want to minimize the expected risk given that we have available only finitely many observations to train the learning algorithm. It turns out, however, that consistency is central to address also this aspect [139]. Additionally, a crucial role for answering this question is played by capacity factors that, roughly speaking, are all measures of how well the set of functions $\{f(x;\theta) \ : \ \theta \in \Theta\}$ can separate data. A more detailed description is given in the following[1]. In general, the theory states that without restricting the set of admissible functions, the ERM is not consistent. The interested reader is referred to [139, 22]

VC Entropy The first capacity factor[2] relates to the expected number of *equivalence classes*[3] according to which the training patterns divide the set of functions $\{f(x;\theta) \ : \ \theta \in \Theta\}$. We denote it by $\text{En}(p, N)$ where the symbols emphasize the dependence of the VC Entropy on the underlying joint probability p and the number of training patterns N. The condition

$$\lim_{N\to\infty} \frac{\text{En}(p, N)}{N} = 0$$

forms the necessary and sufficient condition for (14) to hold true *with respect to the fixed probability density p.*

Growth Function It corresponds to the maximal number of *equivalence classes* with respect to all the possible training samples of cardinality N. As such, it is a distribution-independent version of the VC Entropy obtained via a worst-case approach. We denote it by $\text{Gr}(N)$. The condition

$$\lim_{N\to\infty} \frac{\ln \text{Gr}(N)}{N} = 0$$

forms the necessary and sufficient condition for (14) to hold true *for all the probability densities p.*

[1]Precise definitions and formulas can be found in [139, Chapter 2].

[2]Here and below VC is used as an abbreviation for Vapnik-Chervonenkis.

[3]An equivalence class is a subset of $\{f(x;\theta) \ : \ \theta \in \Theta\}$ consisting of functions that attribute the same labels to the input pattern in the training set.

VC dimension This is the cardinality of the largest set of points that the algorithm can shatter; we denote it by $\dim_{VC}$. Note that $\dim_{VC}$ is a property of $\{f(x;\theta) : \theta \in \Theta\}$ that neither depends on N nor on p. Roughly speaking it tells how flexible is the set of functions. A finite value of $\dim_{VC}$ forms the necessary and sufficient condition for (14) to hold true *for all the probability densities* p.

The three capacities are related by the chain of inequalities [142, 143]:

$$\mathrm{En}(p,N) \leq \ln \mathrm{Gr}(N) \leq \dim_{VC}\left(\ln \frac{N}{\dim_{VC}} + 1\right) . \tag{15}$$

2.3.4 Finite-sample Bounds

One of the key results of the theory developed by Vapnik and Chervonenkis is the following probabilistic bound. With probability $1-\eta$ simultaneously for all $\theta \in \Theta$ it holds that [139]:

$$R(\theta) \leq R^N_{\mathrm{emp}}(\theta) + \sqrt{\frac{\mathrm{En}(p,2N) - \ln \eta}{N}} . \tag{16}$$

Note that the latter depends on p. The result says that, for a fixed set of functions $\{f(x;\theta) : \theta \in \Theta\}$, one can pick that $\theta \in \Theta$ that minimizes $R^N_{\mathrm{emp}}(\theta)$ and obtain in this way the best guarantee on $R(\theta)$. Now, taking into account (15) one can formulate the following bound based on the growth function:

$$R(\theta) \leq R^N_{\mathrm{emp}}(\theta) + \sqrt{\frac{\ln \mathrm{Gr}(2N) - \ln \eta}{N}} . \tag{17}$$

In the same way one has:

$$R(\theta) \leq R^N_{\mathrm{emp}}(\theta) + \sqrt{\frac{\dim_{VC}\left(\ln \frac{2N}{\dim_{VC}} + 1\right) - \ln \eta}{N}} . \tag{18}$$

Figure 2 illustrates the main idea.

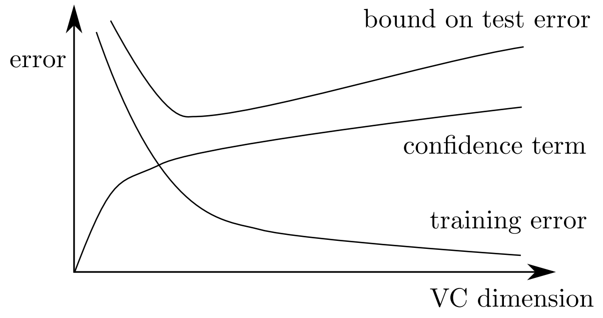


Figure 2: Illustration of the generalization bound depending on capacity factors

Note that both (17) and (18) are distribution-independent. Additionally (18) only depends upon the VC dimension (which, contrary to Gr, is independent from N). Unfortunately there is no free lunch: (17) is less tight than (16) and (18) is less tight than (17).

So far we gave a flavor of the theoretical framework in which the support vector algorithms were originally conceived. Recent research reinterpreted and significantly improved the error bounds using mathematical tools from approximation and learning theory, functional analysis and statistics. The interested reader is referred to [34] and [115]. Although tighter bounds exist, the study of sharper bounds remains a challenge for future research. In fact, existing bounds are normally too loose to lead to practical model selection techniques, i.e., strategies for tuning the parameters that controls the capacity of the model's class. Nonetheless, the theory provides important guidelines for the derivation of algorithms.

2.3.5 The Role of Transduction

It turns out that a key step in obtaining the bound (16) is based upon the *symmetrization lemma*:

$$P\left\{\sup_{\theta} |R(\theta) - R^N_{\mathrm{emp}}(\theta)| \geq \epsilon\right\} \leq 2P\left\{\sup_{\theta} |R^{N_1}_{\mathrm{emp}_1}(\theta) - R^{N_2}_{\mathrm{emp}_2}(\theta)| \geq \frac{\epsilon}{2}\right\} \tag{19}$$

where $R^{N_1}_{\mathrm{emp}_1}$ and $R^{N_2}_{\mathrm{emp}_2}$ are constructed upon two different i.i.d. samples, precisely as in transduction. More specifically, (16) comes from upper-bounding the right hand-side of (19) [140]. More generally it is apparent that for obtaining all bounds of this type the key element remains the symmetrization lemma [139]. Notably starting from the latter one can derive bounds explicitly designed for the transductive case where one of the two samples plays the role of the training set and the other of the test set. In light of this, Vapnik argues that transductive inference is a fundamental step in machine learning. Additionally, since the bounds for transduction are tighter than those for induction, the theory suggests that, whenever possible, transductive inference should be preferred over inductive inference. Practical algorithms can take advantage of this fact by implementing the *adaptive* version of the Structural Risk Minimization (SRM) principle that we discuss next.

2.4 Structural Risk Minimization and Regularization

2.4.1 The Structural Risk Minimization Principle

The structure of the bounds above suggests that one should minimize the empirical risk while controlling some measure of complexity. The idea behind the SRM, introduced in the 1970's, is to construct nested subsets of functions:

$$\mathscr{S}_1 \subset \mathscr{S}_1 \subset \cdots \subset \mathscr{S}_L = \mathscr{S} = \{f(x, \theta) \; : \; \theta \in \Theta\} \tag{20}$$

where each subset $\mathscr{S}_l$ has capacity h_l (VC entropy, Growth function or VC dimension) with $h_1 < h_2 < \cdots < h_l$ and $\mathscr{S}$ is the entire hypothesis space. Then one chooses an element of the nested subsets so that the second term in the right hand side of the bounds is kept under control; within that subset one then picks that specific function that minimizes the empirical risk. As Vapnik points out in [140]:

> *[...] to find a good solution using a finite (limited) number of training examples one has to construct a (smart) structure which reflects prior knowledge about the problem of interest.*

In practice one can use the information coming from the unlabeled data to define a smart structure to improve the learning. In other words, the side information coming from unlabeled data can serve the purpose of devising a *data-dependent* set of functions. On top of this, one should use additional side information over the structure of the problem, whenever available. Indeed using *informative representations* for the input data is also a way to construct smart set of functions. In fact, representing the data in a suitable form implies a mapping from the input space to a more convenient set of features. We will discuss this aspect more extensively in Section 6.

2.4.2 Learning through Regularization

So far we addressed the theory but we have not talked about how to practically implement it. It is understood that the essential idea of SRM is to find the best trade-off between the empirical risk and some measure of complexity (the capacity) of the hypothesis space. This ensures that the left hand side of VC bounds — the expected risk that we are interested in to achieve generalization — is minimized. In practice there are different ways to define the sets in the sequence (20). The generic set $\mathscr{S}_l$ could be the set of polynomials of degree l or a set of splines with l nodes. However, it is in connection to *regularization theory* that practical implementations of the SRM principle find their natural domain.

2.4.3 Tikhonov Theory

Regularization theory was introduced by Andrey Tikhonov [128, 129, 130] as a way to solve ill-posed problems. Ill-posed problems are problems that are not well-posed in the sense of Hadamard [54]. Consider solving in f a linear operatorial equation of the type:

$$Af = b \; . \tag{21}$$

In the general case, f is an element of a Hilbert space, A is a compact operator and b is an element of its range. Even if a solution exists, it is often observed that a slight perturbation of the right hand side b causes large deviations in the solution f. Tikhonov proposed to solve this problem by minimizing a functional of the type:

$$\|Af - b\|^2 + \lambda \Gamma(f)$$

where $\| \cdot \|$ is a suitable norm on the range of A, λ is some hyper-parameter and Γ is a regularization functional (sometimes called *stabilizer*). The theory of such an approach was developed by Tikhonov and Ivanov; in particular it was shown that there exists a strategy to choose λ depending on the accuracy of b that asymptotically leads to the desired solution $f^\star$. This was shown under the assumption that there exists $c^\star$ such that $f^\star \in \{f \; : \; \Gamma(f) \leq c^\star\}$.

According to Vapnik [138], the theory of Tikhonov regularization differs from Statistical Learning Theory in a number of ways. To begin with Tikhonov regularization considers specific structures in the nested sequence (20) (depending on the way Γ is defined); secondly it requires the solution to be in the hypothesis space; finally the theory developed by Tikhonov and Ivanov was not concerned with guarantees for a finite number of observations.

When f is an element of a Reproducing Kernel Hilbert Space (RKHS) (see Section 6), the theory is best known through the work of Wahba [148, 67].

2.4.4 SRM and Regularization in RKHSs

SVMs, and more generally, primal-dual learning algorithms, represent an important class of kernel methods. The primal-dual approach emphasizes the geometrical aspects of the problem and it is particularly insightful when (7) is used to define a discriminative rule arising in a classification problem. We will consider this class of learning algorithms in later sections. Alternatively, the setting of RKHSs provides a convenient way to define the sequence (20). When the hypothesis space $\mathscr{S}$ coincides with a RKHS of functions $\mathcal{H}$ a nested sequence can be constructed by bounding the norm in $\mathcal{H}$, used as a proxy for the complexity of models:

$$\mathscr{S}_l = \{f \in \mathcal{H} \ : \ \|f\| \leq a_l\} \ . \tag{22}$$

It turns out that there is a measure of capacity of $\mathscr{S}_l$ which is an increasing function of a_l [46]. This capacity measure can be used to derive probabilistic bounds in line with (16), (17) and (18). In practice, instead of solving the *constrained problem*:

$$\begin{array}{rl} \min_{f\in\mathcal{H}} & R_{\mathrm{emp}}^N(f) \\ \text{subject to} & \|f\| \leq a_l \end{array} \tag{23}$$

for any l, one normally solves the provably equivalent *penalized problem*:

$$\min_{f\in\mathcal{H}} R_{\mathrm{emp}}^N(f) + \lambda_l \|f\|^2 \tag{24}$$

and pick the optimal λ_l appropriately.

Note that in (23) and (24) we wrote $R_{\mathrm{emp}}^N(f)$ instead of $R_{\mathrm{emp}}^N(\theta)$, as before. In fact, in this case the solution of the learning problem is found by formulating a *convex variational problem* where the function f itself plays the role of the optimization variable θ. In practice, however, the *representer theorem* [148, 67, 101, 40] shows that a representation of the optimal f only depends upon an expansion of kernel functions centered at the training patterns. This result leads to a representation for f in line with (7). More specifically it holds that $f(x) = \sum_{n=1}^N \alpha_n k(x_n, x)$ where $\alpha \in \mathbb{R}^N$ is found solving a finite dimensional optimization problem. The latter is *convex* [23] provided that L in the empirical risk (89) is a convex loss function.

2.4.5 Abstract Penalized Empirical Risk Minimization Problems

The penalized empirical risk minimization problem was introduced in (24) in the setting of RKHS of functions. However it shall be noted that it is a very general idea. Ultimately this can be related to the generality of equation (21). The latter can either refer to infinite dimensional problems or to a finite system of linear equations involving objects living in some finite dimensional space. Therefore for the sake of generality one can consider in place of (24) the problem:

$$\min_{\theta\in\Theta} R_{\mathrm{emp}}^N(\theta) + \lambda\,\Gamma(\theta) \tag{25}$$

where Θ — which is one-to-one with the hypothesis space — either coincides with some abstract vector space, or it is a subset of it; R_{emp}^N is the empirical risk and $\Gamma : \Theta \to \mathbb{R}$ is a suitable penalty function. This, in particular, includes the situations where θ is a vector, a matrix or a higher-order tensor (i.e., a higher order array generalizing the notion of matrices).

2.5 Types of Regularization

A penalty frequently used in practice is $\Gamma(\theta) = \|\theta\|^2$ where $\|\theta\|$ is the *Hilbertian norm* defined upon the space's inner product:

$$\|\theta\|^2 = \langle\theta, \theta\rangle \ . \tag{26}$$

This choice leads to *Ridge Regression* [56, 80]. Note that, in this case, $\|\theta\| = 0$ if and only if θ is the zero vector of the space. A more general class of quadratic penalties is represented by *seminorms*. A seminorm is allowed to assign zero length to some non-zero vectors (in addition to the zero vector). They are commonly used in smoothing splines [148, 51] where the unknown is decomposed into an unpenalized parametric component and a penalized non-parametric part.

2.5.1 The LASSO and non-Hilbertian Norms

The methods that we present in the next sections are all instances of the problem class in (25); although this is not necessarily emphasized in the presentation, they all employ a simple quadratic penalty. This is central to rely on Lagrange duality theory (see, e.g., [23, 19]) that, in turn, constitutes the main technical tool for the derivation of a large class of kernel methods. However, it is important to mention that in the last decade a lot of research effort has been put on the design of alternative penalties[4]. This arises from the realization that using a certain penalty is also a

[4]Correspondingly, there has been increased interest in other notions of duality, such as *Fenchel duality*.

way to convey prior knowledge. This fact is best understood within a Bayesian setting, in light of a *Maximum a Posteriori* (MAP) interpretation of (25), see, e.g., [46]. A penalty term based on the space's inner product has been replaced with various type of *non-Hilbertian norms.* These are norms that, contrary to (26), do not arise from inner products. The LASSO (Least Absolute Shrinkage and Selection Operator, [127]) is perhaps the most prominent example of such cases. In the LASSO one considers linear functions

$$f(x;\theta) = \langle \theta, x\rangle = \sum_{d=1}^{D} \theta_d x_d$$

and uses the the l_1 norm

$$\|\theta\|_1 = \sum_{d=1}^{D} |\theta_d| \tag{27}$$

to promote the sparsity of the parameter vector θ. Note that this corresponds to define the structure (20) according to:

$$\mathscr{S}_l = \{f(x;\theta) \ : \ \|\theta\|_1 \le a_l\} \ . \tag{28}$$

Like Ridge Regression, the LASSO is a continuous shrinkage method that achieves good prediction performance via a bias-variance trade-off. Since usually the estimated coefficient vector has many entries equal to zero, the approach has the further advantage over Ridge Regression of giving rise to interpretable models.

More recently, different structure-inducing penalties have been proposed as a promising alternative, see [155, 62, 154]. The general idea is to convey structural assumption on the problem, such as grouping or hierarchies over the set of input variables, by suitably crafting the penalty. In this way the users are permitted to customize the regularization approach according to their subjective knowledge on the task. Correspondingly, as in (28), one (implicitly) forms a smart structure of nested subsets of functions, in agreement with the SRM principle.

These ideas have been generalized to the case where Θ is infinite dimensional, in particular in the framework of the Multiple Kernel Learning (MKL) problem. This was investigated both from a functional viewpoint [83, 84], and from the more pragmatic point of view of optimization [8, 70, 9].

2.5.2 Spectral Regularization

Yet another generalization of (25) arises in the context of *Multi-task learning* [13, 26, 126]. In this setting one approaches simultaneously different learning tasks under some common constraint(s). The general idea, sometimes also known as *collaborative filtering*, is that one can take advantage of shared features across tasks. In practical applications it was shown that one can significantly gain in terms of generalization performance from exploiting such prior knowledge. From the point of view of learning through regularization, a sensible approach is given in [4, 5]. Suppose one has T datasets, one for each task; the $t-$th dataset has N_t observations. Note that, in general, $N_1 \neq N_2 \neq \cdots \neq N_T$. In this setting one has to learn vectors $\theta_t, \ t = 1, 2, \ldots, T$, one per task; the parameter space is therefore a space of matrices, $\Theta = \mathbb{R}^{F\times T}$ where F is, possibly, infinity. The idea translates into penalized empirical risk minimization problems of the type:

$$\min_{\theta=[\theta_1,\theta_2,\cdots,\theta_T]\in\mathbb{R}^{F\times T}} \sum_{t=1}^{T} R_{\text{emp}}^{N_t}(\theta_t) + \lambda\|\theta\|_* \tag{29}$$

where $\|\theta\|_*$ is the *nuclear norm*:

$$\|\theta\|_* = \sum_{r=1}^{R} \sigma_r(\theta) \tag{30}$$

and $\sigma_1(\theta), \ \sigma_2(\theta), \cdots, \sigma_R(\theta)$ are the $R \le \min(T, F)$ non-zero singular values of the $F \times T$ matrix θ. Note that (30) corresponds to the l_1 norm of the vector of singular values. The definition remains valid also in the infinite dimensional case under some regularity assumptions, see [38]. The nuclear norm is the convex envelope of the rank function on the spectral-norm unit ball [48]; roughly speaking, it represents the best convex relaxation of the rank function.

The use of the nuclear norm in (29) is motivated by the assumption that the parameter vectors of related tasks should be approximately linearly dependent. This assumption is meaningful for a number of cases of interests. Other uses of the nuclear norm exist; ultimately, this is due to the fact that notions of rank are ubiquitous in the mathematical formulations stemming from real-life problems. As a consequence, the nuclear norm is a very versatile mathematical tool to impose structure on (seemingly) very diverse settings. This includes the identification of linear time-invariant systems [73, 72] and the analysis of non-stationary cointegrated systems [111]. Finally we mention that, in place of (30), one can consider *spectral penalties* [6, 1] that include the nuclear norm as a special case.

3 Primal-Dual Methods

The purpose of this section is to introduce the methods that have served as the archetypal approaches for a large class of kernel methods. In the process, we detail the Lagrange duality argument underlying general primal-dual techniques. We begin by giving a short overview of the formulations of SVMs introduced by Vapnik; successively, we discuss a number of modifications and extensions.

3.1 SVMs for Classification

3.1.1 Margin

The problem of pattern recognition amounts at finding the label $y \in \{-1, 1\}$ corresponding to a generic input point $x \in \mathbb{R}^D$. This task can be approached by assigning a label $\hat{y}$ according to the model

$$\hat{y} = \text{sign}\left[w^\top \phi(x) + b\right] \tag{31}$$

where $w^\top \phi(x)+b$ is a hyperplane, found by a learning algorithm based on training data

$$\{(x_1, y_1),\ (x_2, y_2),\ \ldots,\ (x_N, y_N)\} \subset \mathbb{R}^D \times \{-1, 1\}\ . \tag{32}$$

The concept of feature map ϕ was presented in short in Section 1.3.2. Later we will discuss the role of ϕ in more details. For now, it suffices to say that ϕ is expected to capture features that are important for the discrimination of points. In the simplest case, ϕ is the identity map, i.e., $\phi : x \mapsto x$. Note that $w^\top \phi(x) + b$ is a primal model of the type (6), with $w \in \mathcal{F}$ and $b \in \mathbb{R}$.

In general, one can see that there are several possible separating hyperplanes, see Figure 3. The solution picked by the Support Vector Classfication (SVC) algorithm is the one that separates the data with the maximal margin. More precisely, Vapnik considered a rescaling of the problem so that points closest to the separating hyperplane satisfy the normalizing condition

$$\left|w^\top \phi(x) + b\right| = 1\ . \tag{33}$$

The two hyperplanes $w^\top \phi(x) + b = 1$ and $w^\top \phi(x) + b = -1$ are called *canonical hyperplanes* and the distance between them is called the *margin*, see Figure 4.

Assuming that the classification problem is *separable*, i.e., that there exists at least a hyperplane separating the training data (58), one obtains a canonical representation (w, b) satisfying

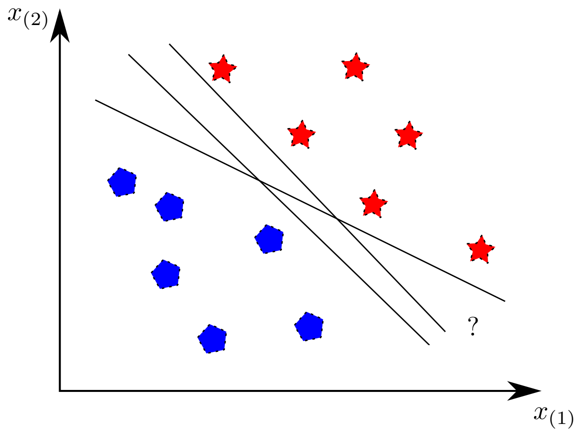


Figure 3: Several possible separating hyperplanes exist

$$y_n \left(w^\top \phi(x_n) + b\right) \geq 1,\ \ n = 1, \ldots, N\ . \tag{34}$$

Let us assume, without loss of generality, that $y_1 = 1$ and $y_2 = -1$. If the corresponding pattern x_1 and x_2 are among the closest points to the separating hyperplane, the scaling imposed by Vapnik implies:

$$\begin{array}{rcl} w^\top \phi(x_1) + b & = & 1 \\ w^\top \phi(x_2) + b & = & -1 \end{array} \tag{35}$$

which, in turn, leads to

$$w^\top (\phi(x_1) - \phi(x_2)) = 2\ . \tag{36}$$

Now the normal vector to the separating hyperplane $w^\top \phi(x) + b$, is $(1/\|w\|)w$. The margin is equal to the component of the vector $\phi(x_1) - \phi(x_2)$ along $(1/\|w\|)w$, i.e., the projection

$$(1/\|w\|)w^\top (\phi(x_1) - \phi(x_2))\ .$$

Using (36) one obtains that the margin is equal to $2/\|w\|$; correspondingly, the distance between the points satisfying (33) and the separating hyperplane, is $1/\|w\|$. By minimizing $\|w\|$, subject to the set of constraints (34), one obtain a *maximal margin classifier* that maximizes the margin between the two classes. This hyperplane, in turn, can be naturally envisioned as the simplest solution given the observed data.

3.1.2 Primal Problem

In practice, for computational reasons it is more convenient to minimize $\frac{1}{2}\|w\|^2 = \frac{1}{2}w^\top w$ rather that $\|w\|$.

Additionally, it is in general unrealistic to assume that the classification problem is separable. In practical applications, one should try to find a set of features (in fact, a feature mapping from the input domain to a more convenient representation) that allow to separate the two classes as much as possible. Nonetheless, there might be no boundary that can perfectly separate the data; therefore one should tolerate misclassifications. Keeping into account this requirement leads to the primal problem for the SVC algorithm [32]. This is the quadratic programming (QP) problem:

$$\begin{array}{ll} \min\limits_{w,b,\xi} & J_P(w,\xi) = \frac{1}{2}w^\top w + c\sum_{n=1}^N \xi_n \\ \text{subject to} & y_n(w^\top \phi(x_n)+b) \geq 1-\xi_n,\ n=1,\dots,N \\ & \xi_n \geq 0,\ n=1,\dots,N \end{array} \tag{SVC-P}$$

where $c > 0$ is a user-defined parameter. In this problem, one accounts for misclassifications by replacing the set of constraints (34), with the set of constraints:

$$y_n\left(w^\top \phi(x_n)+b\right) \geq 1-\xi_n,\ n=1,\dots,N \tag{37}$$

where $\xi_1,\ \xi_2,\dots,\xi_N$ are positive slack variables. It is clear that for higher values of c one penalizes more the violations of the conditions in (34).

3.1.3 Dual Problem

The Lagrangian corresponding to (SVC-P) is:

$$\mathcal{L}(w,b,\xi;\alpha,\nu) = J_P(w,\xi) - \sum_{n=1}^N \alpha_n\left(y_n(w^\top\phi(x_n)+b)-1+\xi_n\right) - \sum_{n=1}^N \nu_n\xi_n \tag{38}$$

with Lagrangian multipliers $\alpha_n \geq 0,\ \nu_n \geq 0$ for $n = 1,\dots,N$. The solution is given by the saddle point of the Lagrangian:

$$\max_{\alpha,\nu}\min_{w,b,\xi}\mathcal{L}(w,b,\xi;\alpha,\nu)\ . \tag{39}$$

One obtains:

$$\left\{\begin{array}{lll} \frac{\partial\mathcal{L}}{\partial w}=0 & \to & w=\sum_{n=1}^N \alpha_n y_n\phi(x_n) \\ \frac{\partial\mathcal{L}}{\partial b}=0 & \to & \sum_{n=1}^N \alpha_n y_n = 0 \\ \frac{\partial\mathcal{L}}{\partial \xi_n}=0 & \to & 0\leq\alpha_n\leq c, \quad n=1,\dots,N\ . \end{array}\right. \tag{40}$$

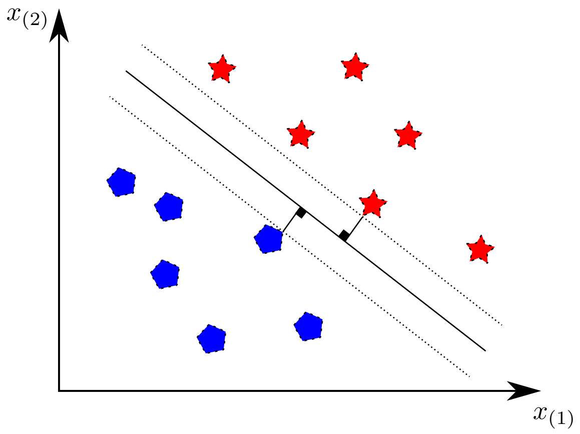


Figure 4: SVC finds the solution that maximizes the margin

The dual problem is then the QP problem:

$$\begin{array}{lll} \max\limits_{\alpha} & J_D(\alpha) & \\ \text{subject to} & \sum_{n=1}^N \alpha_n y_n = 0 & \\ & 0\leq\alpha_n\leq c, & n=1,\dots,N \end{array} \tag{SVC-D}$$

where

$$J_D(\alpha) = -\frac{1}{2}\sum_{m,n=1}^N y_m y_n k(x_m,x_n)\alpha_m\alpha_n + \sum_{n=1}^N \alpha_n \tag{41}$$

and we used the kernel trick:

$$k(x_m,x_n) = \phi(x_m)^\top\phi(x_n), \qquad m,n=1,\dots,N\ . \tag{42}$$

The classifier based on the dual model representation is:

$$\text{sign}\left[\sum_{n=1}^N \alpha_n y_n k(x,x_n)+b\right] \tag{43}$$

where α_n are positive real number obtained solving (SVC-D) and b is obtained based upon Karush–Kuhn–Tucker optimality conditions, i.e., the set of conditions which must be satisfied at the optimum of a constrained optimization problem. These are:

$$\begin{cases} \frac{\partial \mathcal{L}}{\partial w} = 0 \rightarrow w = \sum_{n=1}^{N} \alpha_n y_n \phi(x_n), \\ \frac{\partial \mathcal{L}}{\partial b} = 0 \rightarrow \sum_{n=1}^{N} \alpha_n y_n = 0, \\ \frac{\partial \mathcal{L}}{\partial \xi_n} = 0 \rightarrow c - \alpha_n - \nu_n = 0,\ n = 1, \ldots, N \\ \alpha_n \left(y_n (w^\top \phi(x_n) + b) - 1 + \xi_n \right) = 0,\ n = 1, \ldots, N \\ \nu_n \xi_n = 0,\ n = 1, \ldots, N \\ \alpha_n \geq 0,\ n = 1, \ldots, N \\ \nu_n \geq 0,\ n = 1, \ldots, N\ . \end{cases} \tag{44}$$

From these equations it is seen that, at optimum, we have:

$$y_n(w^\top \phi(x_n) + b) - 1 = 0 \ \text{ if } \ 0 < \alpha_n < c\ . \tag{45}$$

from which one can compute b.

3.1.4 SVC as a Penalized Empirical Risk Minimization Problem

The derivation so far followed the classical approach due to Vapnik; the main argument comes from geometrical insights on the pattern recognition problem. Whenever the feature space is finite dimensional, one can approach learning either by solving the primal problem or by solving the dual; when this is not the case, one can still use the dual and rely on the obtained dual representation.

Before proceeding, we highlight a different, yet equivalent problem formulation. For the primal problem this reads:

$$\min_{w,b} \quad \sum_{n=1}^{N} \left[1 - y_n(w^\top \phi(x_n) + b)\right]_+ + \lambda\, w^\top w \tag{46}$$

where we let $\lambda = 1/(2c)$ and we defined $[\cdot]_+$ by

$$[a]_+ = \begin{cases} a, & \text{if } a \geq 0 \\ 0, & \text{otherwise}\ . \end{cases} \tag{47}$$

Problem (SVC-P) is an instance of (25) obtained by letting[5] $\Theta = \mathcal{F} \times \mathbb{R}$ and taking as penalty the seminorm:

$$\Gamma : (w, b) \mapsto w^\top w\ . \tag{48}$$

This shows that (SVC-P) is essentially a regularized empirical risk minimization problem that can be analyzed in the framework of the SRM principle presented in Section 2.

[5] Note that $\mathcal{F} \times \mathbb{R}$ is naturally equipped with the inner product $\langle (w_1, b_1), (w_2, b_2) \rangle = w_1^\top w_2 + b_1 b_2$ and it is a HS.

3.1.5 VC Bounds for Classification

In Section 2.3 we have already discussed bounds on the generalization error in terms of capacity factors. In particular, (18) states a bound for the case of VC dimension. The larger this VC dimension the smaller the training error (empirical risk) can become but the confidence term (second term in the right hand side of (18)) will grow. The minimum of the sum of these two terms is then a good compromise solution. For SVM classifiers, Vapnik has shown that hyperplanes satisfying $\|w\| \leq a$ have a VC dimension h that is upper-bounded by

$$h \leq \min([r^2 a^2], N) + 1 \tag{49}$$

where $[\cdot]$ represents here the integer part and r is the radius of the smallest ball containing the points $\phi(x_1), \phi(x_2), \ldots, \phi(x_N)$ in the feature space $\mathcal{F}$.

Note that for each value of a there exists a corresponding value of λ in (46) (correspondingly, a value of c in (SVC-P) or (SVC-D)). Additionally, the radius r can also be computed solving a QP problem, see, e.g., [120]. It follows that one could compute solutions corresponding to multiple values of the hyperparameters, find the corresponding empirical risk and radius and then pick the model corresponding to the least value of the right hand side of the bound (18). As we already remarked, however, the bound (18) is often too conservative. Sharper bounds and frameworks alternative to VC theory have been derived see, e.g., [11]. In practice, however, the choice of parameters is often guided by data-driven model selection criteria, see Section 5.

3.1.6 Relative Margin and Data-Dependent Regularization

Although maximum margin classifiers have proved to be very effective, alternative notions of data separation have been proposed. The authors of [108, 107], for instance, argue that maximum margin classifiers might be mislead by direction of large variations. They propose a way to correct this drawback by measuring the margin not in an absolute sense but rather relative to the spread of data in any projection direction. Note that this can be seen as a way to conveniently crafting the hypothesis space, an important aspect that we discussed in Section 2.

3.2 SVMs for Function Estimation

In addition to classification, the support vector methodology has also been introduced for linear and

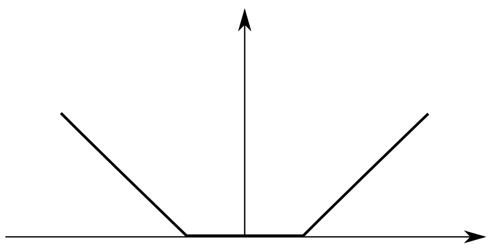


Figure 5: $\epsilon-$insensitive loss

nonlinear function estimation problems [139]. For the general non-linear case, output values are assigned according to the primal model:

$$\hat{y} = w^\top \phi(x) + b \ . \tag{50}$$

In order to estimate the model's parameter w and b, from training data consisting of N input-output pairs, Vapnik proposed to evaluate the empirical risk according to

$$R_{\text{emp}} = \frac{1}{N} \sum_{n=1}^{N} |y_n - w^\top \phi(x_n) - b|_\epsilon \tag{51}$$

with the so called Vapnik's $\epsilon-$insensitive loss function defined as

$$|y - f(x)|_\epsilon = \begin{cases} 0, & \text{if } |y - f(x)| \leq \epsilon \\ |y - f(x)| - \epsilon, & \text{otherwise .} \end{cases} \tag{52}$$

The idea is illustrated in Figure 5. The corresponding primal optimization problem is the QP problem:

$$\begin{aligned} \min_{w,b,\xi} \quad & J_P(w,\xi,\xi^*) = \tfrac{1}{2} w^\top w + c \textstyle\sum_{n=1}^{N} (\xi_n + \xi_n^*) \\ \text{subject to} \quad & y_n - w^\top \phi(x_n) - b \leq \epsilon + \xi_n, \qquad n = 1,\ldots,N \\ & w^\top \phi(x_n) + b - y_n \leq \epsilon + \xi_n^*, \qquad n = 1,\ldots,N \\ & \xi_n,\ \xi_n^* \geq 0, \qquad n = 1,\ldots,N \end{aligned} \tag{SVR-P}$$

where $c > 0$ is a user-defined parameter that determines the amount up to which deviations from the desired accuracy ϵ are tolerated. Following the same approach considered above for (SVC-P), one obtains the dual QP problem:

$$\begin{aligned} \max_{\alpha,\alpha^*} \quad & J_D(\alpha_m, \alpha_m^*) \\ \text{subject to} \quad & \textstyle\sum_{n=1}^{N} (\alpha_m - \alpha_m^*) = 0 \\ & 0 \leq \alpha_n \leq c, \quad n = 1,\ldots,N \\ & 0 \leq \alpha_n^* \leq c, \quad n = 1,\ldots,N \ . \end{aligned} \tag{SVR-D}$$

where

$$\begin{aligned} J_D(\alpha_m, \alpha_m^*) = & \\ & -\frac{1}{2} \sum_{m,n=1}^{N} (\alpha_m - \alpha_m^*)(\alpha_n - \alpha_n^*) k(x_m, x_n) \\ & - \epsilon \sum_{n=1}^{N} (\alpha_n - \alpha_n^*) + \sum_{n=1}^{N} y_n (\alpha_n - \alpha_n^*) \ . \end{aligned} \tag{53}$$

Note that, whereas (SVC-P) and (SVC-D) have tuning parameter c, in (SVR-P) and (SVR-D) one has the additional parameter ϵ.

Before continuing, we note that a number of interesting modifications of the original SVR primal-dual formulations exist. In particular, [104] proposed the $\nu-$tube support vector regression. In this method, the objective $J_P(w,\xi,\xi^*)$ in (SVR-P) is replaced by:

$$J_P(w,\xi,\xi^*,\epsilon) = \frac{1}{2} w^\top w + c \left(\nu\epsilon + \frac{1}{N} \sum_{n=1}^{N} (\xi_n + \xi_n^*) \right) \ . \tag{54}$$

In the latter, ϵ is an optimization variable rather than a hyper-parameter, as in (SVC-P); ν, on the other hand, is fixed by the user and controls the fraction of support vectors that is allowed outside the tube.

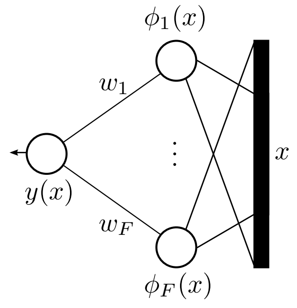
3.3 Main Features of SVMs

Here we briefly highlight the main features of support vector algorithms, making a direct comparison with classical neural networks.

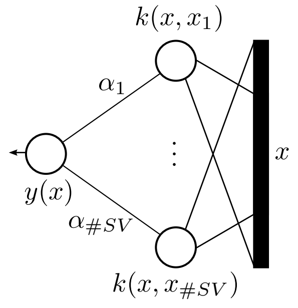
Choice of Kernel A number of possible kernels can be chosen in (42). Some examples are included in the table below.

		$k(x,y)$
linear	:	$x^\top y$
polynomial of degree $d > 0$	:	$(\tau + x^\top y)^d$, for $\tau \geq 0$
Gaussian RBF	:	$\exp(-\|x-y\|^2/\sigma^2)$

In general, it is clear from (42), that a valid kernel function must preserve the fundamental properties of inner-product. That is, for the equality to hold, the bivariate function $k : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}$ is required to be symmetric and positive definite. Note that this, in particular, imposes restriction on τ in the polynomial kernel. A more in depth discussion on kernels is postponed to Section 6.



(a) primal problem



(b) dual problem

Figure 6: Primal-dual network interpretations of SVMs [120]. The number F of hidden units in the primal weight space corresponds to the dimensionality of the feature space. The number $\#SV$ of hidden units in the dual weight space corresponds to the number of non-zero α's

Global Solution (SVC-P) and its dual (SVC-D) are convex problems. This means that any local minimum must also be global. Therefore, even though SVCs share similarities with neural network schemes (see below), they do not suffer from the well-known issue of local minima.

Sparseness The dual model is parsimonious: typically, many α's are zero at the solution with the nonzero ones located in proximity of the decision boundary. This is also desirable in all those setting were one requires fast on-line out-of-sample evaluations of models.

Neural Network Interpretation Both primal (parametric) and dual (non-parametric) problems admit neural network representations [120], see Figure 6. Note that in the dual problem the size of the QP problem is not influenced by the dimension D of the input space, nor it is influenced by the dimension of the feature space. Notably, in classical multi-layer perceptrons one has to fix the number of hidden units in advance; in contrast, in SVMs the number of hidden units follows from the QP problem and corresponds to the number of support vectors.

SVM Solvers The primal and dual formulations presented above are all QP problems. This means that one can rely on general purpose QP solvers for the training of models. Additionally, a number of specialized decomposition methods have been developed, including the Sequential Minimum Optimization (SMO) algorithm [93]. Publicly available software packages such as libSVM [29, 47] and SVMlight [64] include implementations of efficient solvers.

3.4 The Class of Least-squares SVMs

We discuss here the class of Least-Squares SVMs (LS-SVMs) obtained by simple modifications to the SVMs formulations. The arising problems relate to a number of existing methods and entail certain advantages with respect to the original SVM formulations.

3.4.1 LS-SVMs for Classification

We illustrate the idea with respect to the formulation for classification (LS-SVC). The approach, originally proposed in [123], considers the primal problem

$$\begin{array}{ll} \min\limits_{w,b,\xi} & J_P(w,\epsilon) = \frac{1}{2}w^\top w + \frac{\gamma}{2}\sum_{n=1}^N \epsilon_n^2 \\ \text{subject to} & y_n(w^\top \phi(x_n)+b) = 1-\epsilon_n,\ n=1,\ldots,N. \end{array} \tag{LS-SVC-P}$$

This formulation simplifies the primal problem (SVC-P) in two ways. First, the inequality constraints are replaced by equality constraints; the 1's on the right hand side in the constraints are regarded as target values rather than being treated as a threshold. An error ϵ_n is allowed so that misclassifications are tolerated in the case of overlapping distributions. Secondly, a squared loss function is taken for these error variables. The Lagrangian for (LS-SVC-P) is:

$$\mathcal{L}(w,b,\epsilon;\alpha,\nu) = J_P(w,\epsilon) - \sum_{n=1}^N \alpha_n \left(y_n(w^\top\phi(x_n)+b) - 1 + \epsilon_n\right) \tag{55}$$

where α's are Lagrange multipliers that can be positive or negative since the problem now only entails equality constraints. The KKT conditions for optimality yield:

$$\begin{cases} \frac{\partial\mathcal{L}}{\partial w}=0 \rightarrow & w = \sum_{n=1}^N \alpha_n y_n \phi(x_n), \\ \frac{\partial\mathcal{L}}{\partial b}=0 \rightarrow & \sum_{n=1}^N \alpha_n y_n = 0, \\ \frac{\partial\mathcal{L}}{\partial \epsilon_n}=0 \rightarrow & \gamma\epsilon_n = \alpha_n,\ n=1,\ldots,N \\ \frac{\partial\mathcal{L}}{\partial \alpha_n}=0 \rightarrow & y_n(w^\top\phi(x_n)+b) - 1 + \epsilon_n = 0,\ n=1,\ldots,N \end{cases} \tag{56}$$

By eliminating the primal variables w and b, one obtains the KKT system [90]:

$$\begin{bmatrix} 0 & y^\top \\ y & \Omega + I/\gamma \end{bmatrix} \begin{bmatrix} b \\ \alpha \end{bmatrix} = \begin{bmatrix} 0 \\ 1_N \end{bmatrix} \tag{KKT-LSSVC}$$

where $y = [y_1, y_2, \ldots, y_N]^\top$ and $1_N = [1, 1, \ldots, 1]^\top$ are $N-$dimensional vectors and Ω is defined entry-wise by:

$$(\Omega)_{mn} = y_m y_n \phi(x_m)^\top \phi(x_n) = y_m y_n k(x_m, x_n) \ . \tag{57}$$

In the latter, we used the kernel trick introduced before. The obtained dual model corresponds to (43) where α and b are now obtained solving the linear system (KKT-LSSVC), rather than a more complex QP problem, as in (SVC-D). Notably, for LS-SVMs (and related methods) one can exploit a number of computational shortcuts related to spectral properties of the kernel matrix; for instance, one can compute solutions for different values of γ at the price of computing the solution of a single problem, which cannot be done for QPs, see [92, 98, 97].

The LS-SVC is easily extended to handle multi-class problems [120]. Extensive comparisons with alternative techniques (including SVC) for binary and multi-class classification are considered in [134, 120]. The results show that, in general, LS-SVC either outperforms or perform comparably to the alternative techniques. Interestingly, it is clear from the primal problem (LS-SVC-P), that LS-SVC maximizes the margin while minimizing the within-class scattering from targets $\{+1, -1\}$. As such, LS-SVC is naturally related to Fisher discriminant analysis in the feature space [120]; see also [12, 85, 103].

3.4.2 Alternative Formulations

Besides classification, a primal-dual approach similar to the one introduced above has been considered for function estimation [120]; in this case too the dual model representation is obtained by solving a linear system of equations rather than a QP problem, as in SVR. This approach to function estimation is similar to a number of techniques including smoothing splines [148], regularization networks [46, 94], kernel ridge regression [100] and kriging [33]. LS-SVM solutions also share similarities with Gaussian Processes [79, 151] that we discuss in more details in the next Section.

Other formulations have been considered within a primal-dual framework. These include principal component analysis [122], that we discuss next, spectral clustering [3], canonical correlation analysis [136], dimensionality reduction and data visualization [117], recurrent networks [124] and optimal control [125]; see also [118]. In all these cases the estimation problem of interest is conceived at the primal level as an optimization problem with equality constraints, rather than inequality constraints as in SVMs. The constraints relate to the model which is expressed in terms of the feature map. From the KKT optimality conditions one jointly finds the optimal model representation and the model estimate. As for the case of classification the dual model representation is expressed in terms of kernel functions.

3.4.3 Sparsity and Robustness

An important difference with SVMs, is that the dual model found via LS-SVMs depends upon all the training data. Reduction and pruning techniques have been used to achieve the sparse representation in a second stage [120, 119]. A different approach which makes use of the primal-dual setting leads to *fixed-size* techniques [120], which relate to the *Nyström method* proposed in [152] but lead to estimation in the primal. Optimized versions of fixed-size LS-SVMs are currently applicable to large data sets with millions of data points for training and tuning on a personal computer [36].

In LS-SVM the estimation of the support values is only optimal in the case of a Gaussian distribution of the error variables [120]; [119] shows how to obtain robust estimates for regression by applying a weighted version of LS-SVM. The approach is suitable in the case of outliers or non-Gaussian error distributions with heavy tails.

3.5 Kernel Principal Component Analysis

Principal component analysis (PCA) is one of the most important techniques in the class of unsupervised learning algorithms. PCA linearly transforms a number of possibly correlated variables into uncorrelated features called principal components. The transformation is performed to find directions of maximal variation. Often, few principal component can account for most of the structure in the original dataset. PCA is not suitable to discover nonlinear relationships among the original variables. To overcome this limitation [102] originally proposed the idea of performing PCA in a feature space rather that in the input space.

Regardless of the space where the transformation is performed, there is a number of different ways to characterize the derivation of the PCA problem [65]. Ultimately, PCA analysis is readily performed by solving an eigenvalue problem. Here we consider a primal-dual formulation similar to the one introduced above for LS-SVC [120]. In this way the eigenvalue problem is seen to arise from optimality conditions. Notably the approach emphasizes the underlying model which is important for finding the projection of out-of-sample points along the direction of maximal variation. The analysis assumes the knowledge of N training data pairs

$$\{\ x_1,\quad x_2,\quad \ldots,\quad x_N\ \} \subset \mathbb{R}^D \tag{58}$$

i.i.d. according to the generator $p(x)$. The starting point is to define the generic *score variable* z as:

$$z(x) = w^\top (\phi(x) - \hat{\mu}_\phi)\ . \tag{59}$$

The latter represents one projection of $\phi(x) - \mu_\phi$ into the target space. Note that we considered data centered in the feature space with

$$\hat{\mu}_\phi = \frac{1}{N}\sum_{n=1}^{N}\phi(x_n) \tag{60}$$

corresponding to the center of the empirical distribution. The primal problem consists of the following constrained formulation [122]:

$$\begin{array}{rl} \max\limits_{w,z} & -\frac{1}{2}w^\top w + \frac{\gamma}{2}\sum_{n=1}^{N} z_n^2 \\ \text{subject to} & z_n = w^\top\left(\phi(x_n) - \hat{\mu}_\phi\right),\ n = 1,\ldots,N. \end{array} \tag{PCA-P}$$

where $\gamma > 0$. The latter maximizes the empirical variance of z while keeping the norm of the corresponding parameter vector w small by the regularization term. One can also include a bias term, see [120] for a derivation.

The Lagrangian corresponding to (PCA-P) is:

$$\mathcal{L}(w,z;\alpha) = -\frac{1}{2}w^\top w + \frac{\gamma}{2}\sum_{n=1}^{N} z_n^2 - \sum_{n=1}^{N}\alpha_n(z_n - w^\top(\phi(x_n) - \hat{\mu}_\phi)) \tag{61}$$

with conditions for optimality given by

$$\begin{cases} \frac{\partial\mathcal{L}}{\partial w} = 0 \rightarrow w = \sum_{n=1}^{N}\alpha_n\left(\phi(x_n) - \mu_\phi\right), \\ \frac{\partial\mathcal{L}}{\partial z_n} = 0 \rightarrow \alpha_n = \gamma z_n,\ n = 1,\ldots,N\ , \\ \frac{\partial\mathcal{L}}{\partial \alpha_n} = 0 \rightarrow z_n = w^\top\left(\phi(x_n) - \hat{\mu}_\phi\right),\ n = 1,\ldots,N\ . \end{cases} \tag{62}$$

By eliminating the primal variables w and z, one obtains for $n = 1,\ldots,N$:

$$\frac{1}{\gamma}\alpha_n - \sum_{m=1}^{N}\alpha_m\left(\phi(x_n) - \hat{\mu}_\phi\right)\left(\phi(x_m) - \hat{\mu}_\phi\right) = 0\ . \tag{63}$$

The latter is an eigenvalue decomposition that can be stated in matrix notation as:

$$\Omega_c\alpha = \lambda\alpha \tag{64}$$

where $\lambda = 1/\gamma$ and Ω_c is the centered Gram matrix defined entry-wise by:

$$[\Omega_c]_{nm} = k(x_n,x_m) - \frac{1}{N}\sum_{l=1}^{N}k(x_m,x_l) + - \frac{1}{N}\sum_{l=1}^{N}k(x_n,x_l) + \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}k(x_j,x_i)\ . \tag{65}$$

As before, one may choose any positive definite kernel; a typical choice corresponds to the Gaussian RBF kernel. By solving the eigenvalue problem (64) one finds N pairs of eigenvalues and eigenvectors

$$(\lambda_m, \alpha^m),\ m = 1,2,\ldots,N\ .$$

Correspondingly, one finds N score variables with dual representation:

$$z_m(x) = \sum_{n=1}^{N}\alpha_n^m\left(k(x_n,x) - \frac{1}{N}\sum_{l=1}^{N}k(x,x_l) - \frac{1}{N}\sum_{l=1}^{N}k(x_n,x_l) + \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}k(x_j,x_i)\right)\ , \tag{66}$$

in which α^m is the eigenvector associated to the eigenvalue λ_m. Note that all eigenvalues are positive and real because Ω_c is symmetric and positive semidefinite; the eigenvectors are mutually orthogonal, i.e., $(\alpha^l)^\top\alpha^m = 0$, for $l \neq m$. Note that, when the feature map is nonlinear, the number of score variables associated to non-zero eigenvalues might exceed the dimensionality D of the input space. Typically, one selects then the minimal number of score variables that preserve a certain reconstruction accuracy, see [65, 120, 102].

Finally, observe that by the second optimality condition in (62), one has $z_l = \lambda_l\alpha^l$ for $l = 1,2,\ldots,N$.

From this, we obtain that the score variables are empirically uncorrelated. Indeed we have, for $l \neq m$:

$$\sum_{n=1}^{N} z_l(x_n) z_m(x_n) = \sum_{n=1}^{N} \lambda_l \lambda_m (\alpha^l)^\top \alpha^m = 0 \ . \quad (67)$$

4 Gaussian Processes

So far we dealt with primal-dual kernel methods; regularization was motivated by the SRM principle, which achieves generalization by trading off empirical risk with the complexity of the model class. This is representative of a large number of procedures. However, it leaves out an important class of kernel-based probabilistic techniques that goes under the name of *Gaussian Processes*. In Gaussian processes one uses a Bayesian approach to perform inference and learning. The main idea goes back at least to the work of Wiener [149] and Kolmogorov [68] on time-series analysis.

As a first step, one poses a probabilistic model which serves as a prior. This prior is updated in the light of training data so as to obtain a predictive distribution. The latter represents a spectrum of possible answers. In contrast, in the standard SVM/LS-SVM framework one obtains only point-wise estimates. The approach, however, is analytically tractable only for a limited number of cases of interests. In the following we summarize the main ideas in the context of regression where tractability is ensured by Gaussian posteriors; the interested reader is referred to [95] for an in-depth review.

4.1 Definition

A real-valued stochastic process f is a Gaussian Process (GP) if for every finite set of indices $x_1, x_2, \ldots, x_N$ in an index set $\mathcal{X}$, the tuple

$$f_x = (f(x_1), f(x_2), \ldots, f(x_N)) \quad (68)$$

is a multivariate Gaussian random variable taking values in $\mathbb{R}^N$. Note that the index set $\mathcal{X}$ represents the set of all possible inputs. This might be a countable set, such as $\mathbb{N}$ (e.g., a discrete time index) or, more commonly in machine learning, the Euclidean space $\mathbb{R}^D$. A GP f is fully specified by a *mean function* $m : \mathcal{X} \to \mathbb{R}$ and a *covariance function* $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ defined by:

$$\begin{aligned} m(x) =& \mathbb{E}[f(x)] & (69) \\ k(x, x') =& \mathbb{E}[(f(x) - m(x))(f(x') - m(x'))] \ . & (70) \end{aligned}$$

In light of this, one writes:

$$f \sim \mathcal{GP}(m, k) \ . \quad (71)$$

Usually for notational simplicity one takes the mean function to be zero, which we consider here; however, this needs not to be the case.

Note that the specification of the covariance function implies a distribution over any finite collection of random variables obtained sampling the process f at given locations. Specifically we can write for (68):

$$f_x \sim \mathcal{N}(0, K) \quad (72)$$

which means that f_x follows a multivariate zero-mean Gaussian distribution with $N \times N$ covariance matrix K defined entry-wise by

$$[K]_{nm} = k(x_n, x_m) \ .$$

A typical use of a GP is in a regression context, that we consider next.

4.2 GPs for Regression

In regression one observes a dataset of input-output pairs $(x_n, y_n),\ n = 1, 2, \ldots N$ and wants to make prediction at one or more test points. In the following, we call y is the vector obtained staking the target observations and denoted by X the collection of input training patterns (58). In order to carry on the Bayesian inference, one needs a model for the generating process. It is generally assumed that function values are observed in noise, that is:

$$y_n = f(x_n) + \epsilon_n,\ n = 1, 2, \ldots, N \ . \quad (73)$$

One further assumes that ϵ_n are i.i.d. zero-mean Gaussian random variables independent from the process f and with variance σ^2. Under these circumstances the prior on the noisy observations is Gaussian with mean zero and covariance function:

$$c(x_m, x_n) = \mathbb{E}[y_m y_n] = k(x_m, x_n) + \sigma^2 \delta_{nm} \quad (74)$$

where the Kronecker delta function δ_{nm} is one if $n = m$ and zero otherwise.

Suppose now that we are interested in the value f_* of the process at a single test point[6] x_*. By relying on properties of Gaussian probabilities, we can readily

[6]The approach we discuss below extends to multiple test points in a straightforward manner.

write the joint distribution of the test function value and the noisy training observations. This reads:

$$\left[\begin{array}{c} y \\ f_* \end{array}\right] \sim \mathcal{N}\left(0, \left[\begin{array}{cc} K+\sigma^2 I_N & k_x \\ k_x^\top & k_* \end{array}\right]\right) \tag{75}$$

where I_N is the $N\times N$ identity matrix, $k_* = k(x_*, x_*)$ and, finally:

$$k_x = \left[\begin{array}{cccc} k(x_1, x_*), & k(x_2, x_*), & \ldots, & k(x_N, x_*) \end{array}\right]^\top . \tag{76}$$

4.2.1 Prediction with Noisy Observations

Using the conditioning rule for multivariate Gaussian distributions[7], one arrives at the key predictive equation for GP regression:

$$f_*|y, X, x_* \sim \mathcal{N}\left(m_*, \sigma_*^2\right) \tag{77}$$

where

$$m_* = k_x^\top (K+\sigma^2 I)^{-1} y \ , \tag{78}$$

$$\sigma_*^2 = k_* - k_x^\top (K+\sigma^2 I)^{-1} k_x \ . \tag{79}$$

Note that, by letting $\alpha = (K+\sigma^2 I)^{-1} y$, one obtains for the mean value:

$$m_* = \sum_{n=1}^{N} \alpha_n k(x_n, x_*) \ . \tag{80}$$

Up to the bias term b, the latter coincides with the typical[8] dual model representation (7), in which x plays the role of a test point. Therefore one sees that, in the framework of GPs, the covariance function plays the same role of the kernel function. The variance σ_*^2, on the other hand, is seen to be obtained from the prior covariance, by subtracting a positive term which accounts for the information about the process conveyed by training data.

4.2.2 Weight-space View

We have presented GPs through the so called *function space view* [95], which ultimately captures the distinctive nature of this class of methodologies. Here we illustrate a different view, which allows one to achieve three objectives: 1) it is seen that Bayesian linear models are a special instance of GPs; 2) the role of Bayes rule is highlighted; 3) one obtains additional insight on the relationship with the feature map and kernel function used before within primal-dual techniques.

Bayesian regression The starting point is to characterize f as a parametric model involving a set of basis functions $\psi_1, \psi_2, \ldots, \psi_F$:

$$f(x) = \sum_i w_i \psi_i(x) = w^\top \psi(x) \ . \tag{81}$$

Note that F might be infinity. For the special case where ψ is the identity mapping, one recognizes in (73) the standard modelling assumptions for Bayesian linear regression analysis. Inference is based on the posterior distribution over the weights, computed by Bayes rule:

$$\text{posterior} = \frac{\text{likelihood} \times \text{prior}}{\text{marginal likelihood}}, \ p(w|y,X) = \frac{p(y|X,w)p(w)}{p(y|X)} \tag{82}$$

where the marginal likelihood (a.k.a. normalizing constant) is independent on the weights:

$$p(y|X) = \int p(y|X,w)p(w)dw \ . \tag{83}$$

Explicit Feature Space Formulation To make prediction for a test pattern x_* we average over all possible parameter values with weights corresponding to the posterior probability:

$$p(f_*|y,X,x_*) = \int p(f_*|w,x_*)p(w|y,X)dw \ . \tag{84}$$

One can see that computing the posterior $p(w|y,X)$ based upon the prior:

$$p(w) = \mathcal{N}(0, \Sigma_p) \tag{85}$$

gives the predictive model:

$$f_*|y,X,x_* \sim \mathcal{N}\Big(\psi(x_*)^\top \Sigma_p \Psi A y \ , \\ \psi(x_*)^\top \Sigma_p \psi(x_*) - \psi(x_*)^\top \Sigma_p \Psi A \Psi^\top \Sigma_p \psi(x_*)\Big) \tag{86}$$

where $A = (\Psi^\top \Sigma_p \Psi + \sigma^2 I_N)^{-1}$ and we denoted by

$$\Psi = [\psi(x_1), \ \psi(x_2), \ \ldots, \ \psi(x_N)]$$

[7]
$$\left[\begin{array}{c} x \\ y \end{array}\right] \sim \mathcal{N}\left(\left[\begin{array}{c} a \\ b \end{array}\right], \left[\begin{array}{cc} A & C \\ C^\top & B \end{array}\right]\right) \Rightarrow$$
$$y|x \sim \mathcal{N}\left(b + C^\top A^{-1}(x-a), B - C^\top A^{-1} C\right)$$

[8] This holds for the case where data are not centered in the feature space

the feature representation of the training patterns. It is not difficult to see that (86) is (77) in disguise. In particular, one has:

$$k(x, y) = \psi(x)^\top \Sigma_p \psi(y) \ . \tag{87}$$

The positive definiteness of Σ_p ensures the existence and uniqueness of the square root $\Sigma_p^{1/2}$. If we now define

$$\phi(x) = \Sigma_p^{1/2} \psi(x) \tag{88}$$

we retrieve the relationship in (42). We conclude that the kernel function considered in the previous sections can be interpreted as the covariance function of a GP.

4.3 Bayesian Decision Theory

Bayesian inference is particularly appealing when prediction is intended for supporting decisions. In this case, one requires a loss function $L(f_{\text{true}}, f_{\text{guess}})$ which specify the penalty one gets by guessing f_{guess} when the true value is f_{true}. Note that the predictive distribution (77) or — equivalently — (86) was derived without reference to the loss function. This is a major difference with respect to the techniques developed within the framework of statistical learning. Indeed in the non-Bayesian framework of penalized empirical risk minimization, prediction and loss are entangled; one tackles learning in a somewhat more direct way. In contrast, in the Bayesian setting there is a clear distinction between 1) the model that generated the data and 2) capturing the consequences of making guesses[9]. In order to find the point prediction that incurs the minimal *expected* loss, one can define the merit function:

$$R\left(f_{\text{guess}} | y, X, x_*\right) = \int L(f_*, f_{\text{guess}}) p(f_* | y, X, x_*) df_* \ . \tag{89}$$

Note that, since the true value f_{true} is unknown, the latter averages with respect to the model's opinion $p(f_* | y, X, x_*)$ on what the truth might be. The corresponding best guess is:

$$f_{\text{opt}} = \arg \min_{f_{\text{guess}}} R\left(f_{\text{guess}} | y, X, x_*\right) \ . \tag{90}$$

Since $p(f_* | y, X, x_*)$ is Gaussian and hence symmetric, f_{opt} always coincides with the mean m_* whenever the loss is also symmetric. However, in many practical problems such as in safety critical applications, the loss can be asymmetrical. In these cases one has to solve the optimization problem in (90). Similar considerations hold for classification, see [95]. For an account on decision theory see [17].

[9]In light of this, [95] advises to beware from fallacious arguments like "a Gaussian likelihood implies a squared error loss".

5 Model Selection

Kernel-based models depend upon a number of parameters which are determined during training by numerical procedures. Still, one or more hyperparameters usually need to be tuned by the user. In SVC, for instance, one has to fix the value of c. The choice of the kernel function, and of the corresponding parameters, also needs to be properly addressed.

In general, performance measures used for model selection include k-fold cross-validation, leave-one-out (LOO) cross-validation, generalized approximate cross-validation (GACV), approximate span bound, VC bound, and radius-margin bound. For discussions and comparisons see [41, 10]. Another approach found in the literature is kernel-target alignment [106].

5.1 Cross-validation

In practice, model selection based on cross-validation is usually preferred over generalization error bounds. Criticism for cross-validation approaches is related to the high computational load involved; [27] presents an efficient methodology for hyper-parameter tuning and model building using LS-SVMs. The approach is based on the closed form leave-one-out (LOO) cross-validation computation for LS-SVMs, only requiring the same computational cost of one single LS-SVM training. Leave-one-out cross-validation based estimates of performance, however, generally exhibit a relatively high variance and are therefore prone to over-fitting. To amend this, [28] proposed the use of Bayesian regularization at the second level of inference.

5.2 Bayesian Inference of Hyperparameters

Many authors have proposed a full Bayesian framework for kernel-based algorithms in the spirit of the methods developed by MacKay for classical MLPs [76, 77, 78]. In particular, [120] discusses the case of LS-SVMs. It is shown that, besides leading to tuning strategies, the approach allows to take probabilistic interpretations of the outputs; [95] discusses the Bayesian model selection for GPs. In general, the Bayesian framework consists of multiple level of inference. The parameters (i.e., with reference to the primal model, w and b) are inferred at level 1. Contrary to MLPs, this usually entails the solution of a convex optimization problem or even solving a linear system, as in LS-SVMs and GPs. The regularization parameter(s) and the kernel parameter(s) are inferred at higher levels. The method progressively integrates out the parameters by using the evidence at a certain level of inference as the likelihood at the successive level.

6 More on Kernels

We have already seen that a kernel arising from an inner product can be interpreted as the covariance function of a Gaussian Process. In this section we further study the mathematical properties of different yet equivalent notions of kernels. In particular, we will discuss that positive definite kernels are *reproducing*, in a sense that we are about to clarify. We will then review a number of specialized kernels for learning problems involving structured data.

6.1 Positive Definite Kernels

Denote by $\mathcal{X}$ a nonempty index set. A symmetric function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a *positive definite kernel* if for any $N \in \mathbb{N}$ and for any tuple $(x_1, x_2, \ldots, x_N) \in \mathcal{X}^N$, the Gram matrix K defined entry-wise by $K_{nm} = k(x_n, x_m)$, satisfies[10]:

$$\alpha^\top K \alpha = \sum_{n=1}^{N} \sum_{m=1}^{N} K_{nm} \alpha_n \alpha_m \geq 0 \quad \forall \alpha \in \mathbb{R}^N \ .$$

In particular, suppose $\mathcal{F}$ is some Hilbert space[11] (HS) with inner product $\langle \cdot, \cdot \rangle$. Then for any function $\phi : \mathcal{X} \to \mathcal{F}$ one has:

$$\sum_{n=1}^{N} \sum_{m=1}^{N} \langle \phi(x_n), \phi(x_m) \rangle \alpha_n \alpha_m = \sum_{n=1}^{N} \sum_{m=1}^{N} \langle \alpha_n \phi(x_n), \alpha_m \phi(x_m) \rangle = \left\| \sum_{n=1}^{N} \alpha_n \phi(x_n) \right\|^2 \geq 0 \ . \quad (91)$$

[10] Note that, by definition, a positive definite kernel satisfies $k(x,x) \geq 0$ for any $x \in \mathcal{X}$.

[11] For an elementary introduction on HSs see, e.g., [38].

From the first line one then sees that:

$$k : (x, y) \mapsto \langle \phi(x), \phi(y) \rangle \quad (92)$$

is a positive definite kernel in the sense specified above. A continuous positive definite kernel k is often called a *Mercer kernel* [34].

Note that in Section 1.3.2 we denoted $\langle \phi(x), \phi(y) \rangle$ by $\phi(x)^\top \phi(y)$, implicitly making the assumption that the feature space $\mathcal{F}$ were a finite dimensional Euclidean space. However one can show that the feature space associated to certain positive definite kernels (such as the Gaussian RBF, [116]) is an infinite dimensional HS; in turn, the inner product in such a space is commonly denoted as $\langle \cdot, \cdot \rangle$.

6.2 Reproducing Kernels

Evaluation Functional Let $(\mathcal{H}, \langle \cdot, \cdot \rangle)$ be a HS of real-valued functions[12] on $\mathcal{X}$ equipped with the norm $\|f\| = \sqrt{\langle f, f \rangle}$. For $x \in \mathcal{X}$ we denote by L_x the evaluation functional:

$$\begin{array}{rccc} L_x : & \mathcal{H} & \to & \mathbb{R} \\ & f & \mapsto & f(x) \ . \end{array} \quad (93)$$

L_x is said to be bounded if there exists $c > 0$ such that $|L_x f| = |f(x)| \leq c\|f\|$ for all $f \in \mathcal{H}$. By the Riesz representation theorem [31] if L_x is bounded then there exists a unique $\eta_x \in \mathcal{H}$ such that, for any $f \in \mathcal{H}$:

$$L_x f = \langle f, \eta_x \rangle \ . \quad (94)$$

Reproducing Kernel A function:

$$\begin{array}{rcc} k : \mathcal{X} \times \mathcal{X} & \to & \mathbb{R} \\ (x, y) & \mapsto & k(x, y) \end{array} \quad (95)$$

is said to be a *reproducing kernel* of $\mathcal{H}$ if and only if:

$$\forall x \in \mathcal{X}, \ k(\cdot, x) \in \mathcal{H} \quad (96)$$

$$\forall x \in \mathcal{X}, \ \forall f \in \mathcal{H} \ \langle f, k(\cdot, x) \rangle = f(x) \ . \quad (97)$$

Note that by $k(\cdot, x)$ we mean the function $k(\cdot, x) : t \mapsto k(t, x)$.

The definition of reproducing kernel implies that $k(\cdot, x) = \eta_x$, i.e., $k(\cdot, x)$ is the representer of the evaluation functional L_x; (97) goes under the name of *reproducing property.* From (96) and (97) it is clear that

$$k(x, y) = \langle k(\cdot, x), k(\cdot, y) \rangle, \ \forall x, y \in \mathcal{X}; \quad (98)$$

[12] The theory of RKHSs generally deals with complex-valued functions [7, 18]. Here we stick to the real setting for simplicity.

since $\langle \cdot, \cdot \rangle$ is symmetric, it follows that $k(x,y) = k(y,x)$. A HS of functions that possesses a reproducing kernel is called a *Reproducing kernel Hilbert space* (RKHS).

Finally, notice that the r.k. of a space of a HS of functions corresponds in a one-to-one manner with the definition of inner product $\langle \cdot, \cdot \rangle$; changing the inner product product implies a change in the reproducing kernel.

Basic Properties of RKHSs Let $(\mathcal{G}, \langle \cdot, \cdot \rangle)$ be a HS of functions. If, for any x, the evaluation functional L_x is bounded, then it is clear that $\mathcal{G}$ is a RKHS with reproducing kernel

$$k(x,y) = \langle \eta_x, \eta_y \rangle \ . \tag{99}$$

Vice versa, if $\mathcal{G}$ admits a reproducing kernel k, then all evaluation functionals are bounded. Indeed, we have:

$$|f(x)| = \langle f, k(\cdot, x) \rangle \leq \|f\| \|k(\cdot, x)\| = \sqrt{k(x,x)} \|f\| \tag{100}$$

where we simply relied on the Cauchy–Schwarz inequality. Boundedness of evaluation functionals means that all the functions in the space are well-defined for all x[13].

It is immediate to prove that, in a RKHS $\mathcal{H}$, the representation of a bounded linear functional A is simply $Ak(\cdot, x)$, i.e., it is obtained by applying A to the representer of L_x. As an example, take the functional evaluating the derivative of f at x:

$$D_x : \ f \mapsto f'(x) \ .$$

If D_x is bounded on $\mathcal{H}$, then the property implies that:

$$f'(x) = \langle f, k'(\cdot, x) \rangle, \ \forall f \in \mathcal{H}$$

where $k'(\cdot, x)$ is the derivative of the function $k(\cdot, x)$.

6.3 Equivalence Between the two Notions

Moore-Aronszajn Theorem If we let:

$$\begin{array}{rccc} \phi : & \mathcal{X} & \to & \mathcal{H} \\ & x & \mapsto & k(x, \cdot) \end{array} \tag{101}$$

one sees that, in light of (91), the reproducing kernel k is also a positive definite kernel. The converse result, stating that a positive definite kernel is the reproducing kernel of a HS of functions $(\mathcal{H}, \langle \cdot, \cdot \rangle)$, is found in the Moore-Aronszajn theorem [7]. This completes the equivalence between positive definite kernels and reproducing kernels.

[13]Note that this is not the case, for instance, of the space of *square-integrable functions*.

Feature Maps and Mercer Theorem Note that (101) is a first feature map associated to the kernel function k. Correspondingly, this shows that the RKHS $\mathcal{H}$ is a possible instance of the feature space. A different feature map can be given in view of of the Mercer theorem [82, 96], which historically played a major role in the development of kernel methods. The theorem states that every positive definite kernel can be written as:

$$k(x,y) = \sum_{i=1}^{\infty} \mu_i e_i(x) e_i(y) \tag{102}$$

where the series in the right hand side of (102) converges absolutely and uniformly, $(e_i)_i$ is an orthonormal sequence of eigenfunctions and $(\mu_i)_i$ is the corresponding sequence of nonnegative eigenvalues such that for some measure ν:

$$\int k(x,y) e_i(y) d\nu(y) = \mu_i e_i(x) \quad \forall x \in \mathcal{X} \tag{103}$$

$$\int e_j(y) e_i(y) d\nu(y) = \delta_{ij} \ . \tag{104}$$

The eigenfunctions $(e_i)_i$ belong to the RKHS $(\mathcal{H}, \langle \cdot, \cdot \rangle)$ associated to k. In fact, by (103) one has:

$$e_i(x) = \frac{1}{\mu_i} \int k(x,y) e_i(y) d\nu(y) \tag{105}$$

and therefore e_i can be approximated by elements in the span of $(k_x)_{x \in \mathcal{X}}$ [34]. One can further see that $(\sqrt{\mu_i} e_i)_i$ is an orthonormal basis for $\mathcal{H}$; indeed one has:

$$\begin{aligned} &\langle \sqrt{\mu_i} e_i, \sqrt{\mu_j} e_j \rangle \overset{\text{by (103)}}{=} \\ &\quad \langle \frac{1}{\sqrt{\mu_i}} \int k(\cdot, y) e_i(y) d\nu(y), \sqrt{\mu_j} e_j \rangle = \\ &\quad \int \frac{\sqrt{\mu_j}}{\sqrt{\mu_i}} \langle k(\cdot, y), e_j \rangle e_i(y) d\nu(y) \overset{\text{by (97)}}{=} \\ &\int \frac{\sqrt{\mu_j}}{\sqrt{\mu_i}} e_j(y) e_i(y) d\nu(y) \overset{\text{by (104)}}{=} \frac{\sqrt{\mu_j}}{\sqrt{\mu_i}} \delta_{ij} \ . \end{aligned} \tag{106}$$

Note that we considered the general case in which the expansion (102) involves infinitely many terms. However, there are positive definite kernels (e.g., the polynomial kernel) for which only finitely many eigenvalues are non-zero.

In light of the Mercer theorem one can see that a different feature map is given by:

$$\begin{array}{rccl} \phi: & \mathcal{X} & \to & \mathcal{F} \\ & x & \mapsto & (\sqrt{\mu_i} e_i(x))_i \ . \end{array} \tag{107}$$

Note that ϕ maps x into an infinite dimensional vector with $i-$th entry $\phi_i(x) = \sqrt{\mu_i} e_i(x)$.

Connecting Functional and Parametric View One can see now that $\sum_i w_i \phi_i(x)$ in the primal model (6) corresponds to the evaluation in x of a function f in a RKHS. To see this we start from decomposing f according to the orthonormal basis $(\sqrt{\mu_i} e_i)_i$:

$$f = \sum_i \langle f, \sqrt{\mu_i} e_i \rangle \sqrt{\mu_i} e_i = \sum_i w_i \sqrt{\mu_i} e_i \tag{108}$$

where we let $w_i = \langle f, \sqrt{\mu_i} e_i \rangle \sqrt{\mu_i}$. Now one has:

$$L_x f = \langle f, k(\cdot, x) \rangle = \langle \sum_i w_i \sqrt{\mu_i} e_i, k(\cdot, x) \rangle = \sum_i w_i \sqrt{\mu_i} \langle e_i, k(\cdot, x) \rangle = \sum_i w_i \sqrt{\mu_i} e_i(x) = \sum_i w_i \phi_i(x) \tag{109}$$

where we applied the reproducing property on e_i and used the definition of feature map (107). Additionally, notice that one has:

$$\|f\|^2 = \langle f, f \rangle = \langle \sum_i w_i \sqrt{\mu_i} e_i, \sum_j w_j \sqrt{\mu_j} e_j \rangle = \sum_i \sum_j w_i w_j \langle \sqrt{\mu_i} e_i, \sqrt{\mu_j} e_j \rangle \overset{\text{by (106)}}{=} \sum_i \sum_j w_i w_j \frac{\sqrt{\mu_j}}{\sqrt{\mu_i}} \delta_{ij} = \sum_i w_i^2 \ . \tag{110}$$

This shows that the penalty $w^\top w$, used within the primal problems of Section 3, can be connected to the squared norm of a function. The interested reader is referred to [34, 18] for additional properties of kernels.

6.4 Kernels for Structured Data

In applications where data are well represented by vectors in an Euclidean space one usually use the Gaussian RBF kernel, which is *universal* [114]. Nonetheless, there exists an entire set of rules according to which one can design new kernels from elementary positive definite functions, see [105]. Although the idea of kernels have been around for a long time, it was only in the 90's that the machine learning community started to realize that the index set $\mathcal{X}$ does not need to be (a subset of) some Euclidean space. This significantly improved the applicability of kernel-based algorithms to a broad range of data types including sequence data, graphs and trees, XML and HTML documents [49].

Probabilistic Kernels One powerful approach consists of applying a kernel that brings generative models into a (possibly discriminative) kernel-based method [55, 61]. Generative models can deal naturally with missing data and in the case of hidden Markov models can handle sequences of varying length. A popular probabilistic similarity measure is the *Fisher kernel* [59, 131]. The key intuition behind this approach is that similar structured objects should induce similar log-likelihood gradients in the parameters of a predefined class of generative models [59]. Different instances exist, depending on the generative model of interest, see also [105].

Graph Kernels and Dynamical Systems Graphs can very naturally represent entities, their attributes, and their relationships to other entities; this makes them one of the most widely used tools for modeling structured data. Various type of kernels for graphs were proposed, see [146, 50] and reference therein. The approach can be extended to carry on recognitions and decisions for tasks involving dynamical systems; in fact, kernels on dynamical systems are related to graph kernels through the dynamics of random walks [147, 146].

Tensors and Kernels Tensors are multidimensional arrays that represent higher order generalizations of vectors and matrices. Tensor-based methods are often particularly effective in low signal-to-noise ratios and when the number of observations is small in comparison with the dimensionality of the data. They are used in domains ranging from neuroscience to vision and chemometrics, where tensors best capture the multi-way nature of the data [69]. The authors of [109] proposed a family of kernels that exploit the algebraic structure of data tensors. The approach is related to a generalization of the Singular Value Decomposition (SVD) to higher order tensors [37]. The essential idea is to measure similarity based upon a Grassmannian distance of the subspaces spanned by matrix unfolding of data tensors. It can be shown that the approach leads to perfect separation of tensors generated by different sets of rank-1 atoms, see [109]. Within this framework, [110] proposes a kernel function for multichannel signals; the idea exploits the spectral information of tensors of fourth order cross-cumulants associated to each multichannel signal.

7 Applications

Kernel methods have been shown to be successful in many different applications. In this section we mention only a few examples.

7.1 Text Categorization

Recognition of objects and handwritten digits is studied in [71, 39, 20]; natural language text categorization is discussed in [63, 42]. The task consists of classifying documents based on their content. Attribute value representation of text is used to adequately represent the document text; typically, each distinct word in a document represents a feature with value corresponding to the number of occurrences.

7.2 Time-series Analysis

The use of kernel methods for time-series prediction has been discussed in a number of papers [87, 81, 89, 45, 43] with applications ranging from electric load forecasting [44] to financial time series prediction [135]. Non-linear system identification by LS-SVMs is discussed in [120] and references therein; [110] studies the problem of training a discriminative classifier given a set of labelled multivariate time series. Applications include brain decoding tasks based on magnetoencephalography (MEG) recordings.

7.3 Bioinformatics and Biomedical Applications

Gene expression analysis performed by SVMs is discussed in [24]. Applications in metabolomics, genetics and proteomics are presented in the tutorial paper [75]; [35] discussed different techniques for the integration of side information in models based on gene expression data to improve the accuracy of diagnosis and prognosis in cancer; [153] provides an introduction to general data fusion problems using SVMs with application to computational biology problems. Detection of remote protein homologies by SVMs is discussed in [58] which combines discriminative methods with generative models. Bioengineering and bioinformatics applications are found also in [74, 91, 150], Survival analysis based on primal-dual techniques in discussed in [133, 132].

Acknowledgment

Research supported by Research Council KUL: GOA/10/09 MaNet , PFV/10/002 (OPTEC), several PhD/postdoc & fellow grants; Flemish Government: IOF: IOF/KP/SCORES4CHEM; FWO: PhD/postdoc grants, projects: G.0320.08 (convex MPC), G.0557.08 (Glycemia2), G.0588.09 (Brain-machine), G.0377.09 (Mechatronics MPC); G.0377.12 (Structured systems) research community (WOG: MLDM); IWT: PhD Grants, projects: Eureka-Flite+, SBO LeCoPro, SBO Climaqs, SBO POM, EUROSTARS SMART, iMinds 2012. Belgian Federal Science Policy Office: IUAP P7/ (DYSCO, Dynamical systems, control and optimization, 2012-2017). EU: ERNSI, FP7-EMBOCON (ICT-248940), FP7-SADCO (MC ITN-264735), ERC ST HIGHWIND (259 166), ERC AdG A-DATADRIVE-B. COST: Action ICO806: IntelliCIS.

References

[1] J. Abernethy, F. Bach, T. Evgeniou, and J.P. Vert. A new approach to collaborative filtering: Operator estimation with spectral regularization. *Journal of Machine Learning Research*, 10:803–826, 2009.

[2] M. Aizerman, E. M. Braverman, and L. I. Rozonoer. Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control*, 25:821 – 837, 1964.

[3] C. Alzate and J.A.K. Suykens. Multiway spectral clustering with out-of-sample extensions through weighted kernel PCA. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(2):335–347, 2010.

[4] A. Argyriou, T. Evgeniou, and M. Pontil. Multi-task feature learning. In *Advances in Neural Information Processing Systems 19*, pages 41–48, 2007.

[5] A. Argyriou, T. Evgeniou, and M. Pontil. Convex multi-task feature learning. *Machine Learning*, 73(3):243–272, 2008.

[6] A. Argyriou, C.A. Micchelli, and M. Pontil. On spectral learning. *Journal of Machine Learning Research*, 11:935–953, 2010.

[7] N. Aronszajn. Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68:337–404, 1950.

[8] F.R. Bach, G.R.G. Lanckriet, and M.I. Jordan. Multiple kernel learning, conic duality, and the SMO algorithm. In *Proceedings of the twenty-first International Conference on Machine Learning (ICML)*. ACM Press New York, NY, USA, 2004.

[9] F.R. Bach, R. Thibaux, and M.I. Jordan. Computing regularization paths for learning multiple kernels. In *Advances in Neural Information Processing Systems 17*, pages 41–48, 2004.

[10] P.L. Bartlett, S. Boucheron, and G. Lugosi. Model selection and error estimation. *Machine Learning*, 48(1):85–113, 2002.

[11] P.L. Bartlett and S. Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *The Journal of Machine Learning Research*, 3:463–482, 2003.

[12] G. Baudat and F. Anouar. Generalized discriminant analysis using a kernel approach. *Neural computation*, 12(10):2385–2404, 2000.

[13] J. Baxter. Theoretical models of learning to learn. In *T. Mitchell and S. Thrun (Eds.), Learning*, pages 71–94. Kluwer, 1997.

[14] M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural computation*, 15(6):1373–1396, 2003.

[15] M. Belkin and P. Niyogi. Semi-supervised learning on Riemannian manifolds. *Machine Learning*, 56(1):209–239, 2004.

[16] M. Belkin, P. Niyogi, and V. Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning Research*, 7:2399–2434, 2006.

[17] J.O. Berger. *Statistical decision theory and Bayesian analysis*. Springer, 1985.

[18] A. Berlinet and C. Thomas-Agnan. *Reproducing Kernel Hilbert Spaces in Probability and Statistics*. Kluwer Academic Publishers, 2004.

[19] D. P. Bertsekas. *Nonlinear Programming*. Athena Scientific, Belmont, MA, 1995.

[20] V. Blanz, B. Schölkopf, H. Bülthoff, C. Burges, V. Vapnik, and T. Vetter. Comparison of view-based object recognition algorithms using realistic 3d models. *Artificial Neural Networks—ICANN 96*, pages 251–256, 1996.

[21] B.E. Boser, I.M. Guyon, and V.N. Vapnik. A training algorithm for optimal margin classifiers. In D. Haussler, editor, *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory*, pages 144–152, Pittsburgh, PA, July 1992. ACM Press.

[22] O. Bousquet, S. Boucheron, and G. Lugosi. Introduction to statistical learning theory. *Advanced Lectures on Machine Learning*, pages 169–207, 2004.

[23] S.P. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[24] M.P.S. Brown, W.N. Grundy, D. Lin, N. Cristianini, C.W. Sugnet, T.S. Furey, M. Ares, and D. Haussler. Knowledge-based analysis of microarray gene expression data by using support vector machines. *Proceedings of the National Academy of Sciences*, 97(1):262–267, 2000.

[25] C. Campbell. Kernel methods: a survey of current techniques. *Neurocomputing*, 48(1):63–84, 2002.

[26] R. Caruana. *Learning to learn*, chapter Multitask Learning, pages 41–75. Springer, 1998.

[27] G.C. Cawley. Leave-one-out cross-validation based model selection criteria for weighted ls-svms. In *International Joint Conference on Neural Networks (IJCNN)*, pages 1661–1668, 2006.

[28] G.C. Cawley and N.L.C. Talbot. Preventing over-fitting during model selection via bayesian regularisation of the hyper-parameters. *Journal of Machine Learning Research*, 8:841–861, 2007.

[29] C.C. Chang and C.J. Lin. Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27, 2011.

[30] O. Chapelle, B. Schölkopf, and A. Zien. *Semi-supervised learning*. Mit Press, 2006.

[31] J.B. Conway. *A Course in Functional Analysis*. Springer, 1990.

[32] C. Cortes and V. Vapnik. Support vector networks. *Machine Learning*, 20:273–297, 1995.

[33] N. Cressie. The origins of kriging. *Mathematical Geology*, 22(3):239–252, 1990.

[34] F. Cucker and D.X. Zhou. *Learning Theory: An Approximation Theory Viewpoint (Cambridge Monographs on Applied & Computational Mathematics)*. Cambridge University Press New York, NY, USA, 2007.

[35] A. Daemen, M. Signoretto, O. Gevaert, J.A.K. Suykens, and B. De Moor. Improved microarray-based decision support with graph encoded interactome data. *PLoS ONE*, 5(4):1–16, 2010.

[36] K. De Brabanter, J. De Brabanter, J.A.K. Suykens, and B. De Moor. Optimized fixed-size kernel models for large data sets. *Computational Statistics & Data Analysis*, 54(6):1484–1504, 2010.

[37] L. De Lathauwer, B. De Moor, and J. Vandewalle. A multilinear singular value decomposition. *SIAM Journal on Matrix Analysis and Applications*, 21(4):1253–1278, 2000.

[38] L. Debnath and P. Mikusiński. *Hilbert spaces with applications*. Academic Press, 2005.

[39] D. Decoste and B. Schölkopf. Training invariant support vector machines. *Machine Learning*, 46(1):161–190, 2002.

[40] F. Dinuzzo and B. Schölkopf. The representer theorem for Hilbert spaces: a necessary and sufficient condition. In *Advances in Neural Information Processing Systems 26*, 2012.

[41] K. Duan, S.S. Keerthi, and A.N. Poo. Evaluation of simple performance measures for tuning svm hyperparameters. *Neurocomputing*, 51:41–59, 2003.

[42] S. Dumais, J. Platt, D. Heckerman, and M. Sahami. Inductive learning algorithms and representations for text categorization. In *Proceedings of the seventh international conference on Information and knowledge management*, pages 148–155. ACM, 1998.

[43] M. Espinoza, T. Falck, J.A.K. Suykens, and B. De Moor. Time series prediction using ls-svms. In *European Symposium on Time Series Prediction, ESTSP*, volume 8, pages 159–168, 2008.

[44] M. Espinoza, J.A.K. Suykens, R. Belmans, and B. De Moor. Electric load forecasting. *Control Systems, IEEE*, 27(5):43–57, 2007.

[45] M. Espinoza, J.A.K. Suykens, and B. De Moor. Short term chaotic time series prediction using symmetric ls-svm regression. In *Proc. of the 2005 International Symposium on Nonlinear Theory and Applications (NOLTA)*, pages 606–609, 2005.

[46] T. Evgeniou, M. Pontil, and T. Poggio. Regularization networks and support vector machines. *Advances in Computational Mathematics*, 13(1):1–50, 2000.

[47] R.E. Fan, P.H. Chen, and C.J. Lin. Working set selection using second order information for training support vector machines. *The Journal of Machine Learning Research*, 6:1889–1918, 2005.

[48] M. Fazel. *Matrix rank minimization with applications.* PhD thesis, Elec. Eng. Dept., Stanford University, 2002.

[49] T. Gärtner. *Kernels for structured data*, volume 72 of *Machine Perception and Artificial Intelligence.* World Scientific Publishing, 2008.

[50] T. Gärtner, P. Flach, and S. Wrobel. On graph kernels: Hardness results and efficient alternatives. *Learning Theory and Kernel Machines*, pages 129–143, 2003.

[51] C. Gu. *Smoothing spline ANOVA models.* Springer, 2002.

[52] I. Guyon, B. Boser, and V. Vapnik. Automatic capacity tuning of very large VC-dimension classifiers. In *Advances in Neural Information Processing Systems 5*, pages 147–155, 1993.

[53] I. Guyon, V. Vapnik, B. Boser, L. Bottou, and S. A. Solla. Structural risk minimization for character recognition. In *Advances in Neural Information Processing Systems 4*, pages 471–479, 1992.

[54] J. Hadamard. Sur les problèmes aux dérivées partielles et leur signification physique. *Princeton University Bulletin*, 13:49–52, 1902.

[55] D. Haussler. Convolution kernels on discrete structures. Technical report, UC Santa Cruz, 1999.

[56] A. E. Hoerl and R. W. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970.

[57] T. Hofmann, B. Schölkopf, and A.J. Smola. Kernel methods in machine learning. *The annals of statistics*, pages 1171–1220, 2008.

[58] T. Jaakkola, M. Diekhans, and D. Haussler. A discriminative framework for detecting remote protein homologies. *Journal of Computational Biology*, 7(1-2):95–114, 2000.

[59] T.S. Jaakkola and D. Haussler. Exploiting generative models in discriminative classifiers. *Advances in neural information processing systems 11*, pages 487–493, 1999.

[60] F. Jäkel, B. Schölkopf, and F.A. Wichmann. A tutorial on kernel methods for categorization. *Journal of Mathematical Psychology*, 51(6):343–358, 2007.

[61] T. Jebara, R. Kondor, and A. Howard. Probability product kernels. *The Journal of Machine Learning Research*, 5:819–844, 2004.

[62] R. Jenatton, J.Y. Audibert, and F. Bach. Structured variable selection with sparsity-inducing norms. *Technical Report - http://arxiv.org/abs/0904.3523v1*, 2009.

[63] T. Joachims. Text categorization with support vector machines: Learning with many relevant features. *Machine learning: ECML-98*, pages 137–142, 1998.

[64] T. Joachims. Making large–scale SVM learning practical. In B. Schölkopf, C. J. C. Burges, and A. J. Smola, editors, *Advances in Kernel Methods — Support Vector Learning*, pages 169–184, Cambridge, MA, 1999. MIT Press.

[65] I. Jolliffe. *Principal component analysis.* Wiley Online Library, 2005.

[66] T. Kailath. RKHS approach to detection and estimation problems–i: Deterministic signals in gaussian noise. *IEEE Transactions on Information Theory*, 17(5):530–549, 1971.

[67] G. Kimeldorf and G. Wahba. Some results on Tchebycheffian spline functions. *J. Math. Anal. Applic.*, 33:82–95, 1971.

[68] A.N. Kolmogorov. Sur l'interpolation et extrapolation des suites stationnaires. *CR Acad. Sci*, 208:2043–2045, 1939.

[69] P.M. Kroonenberg. *Applied multiway data analysis.* Wiley-Interscience, 2008.

[70] G.R.G. Lanckriet, T. De Bie, N. Cristianini, M.I. Jordan, and W.S. Noble. A statistical framework for genomic data fusion. *Bioinformatics*, 20(16):2626–2635, 2004.

[71] Y. LeCun, L.D. Jackel, L. Bottou, A. Brunot, C. Cortes, J.S. Denker, H. Drucker, I. Guyon, U.A. Muller, E. Sackinger, P. Simard, and V. Vapnik. Comparison of learning algorithms for handwritten digit recognition. In *International Conference on Artificial Neural Networks (ICANN)*, volume 2, pages 53–60, 1995.

[72] Z. Liu and L. Vandenberghe. Interior-point method for nuclear norm approximation with application to system identification. *SIAM Journal on Matrix Analysis and Applications*, 31(3):1235–1256, 2009.

[73] Z. Liu and L. Vandenberghe. Semidefinite programming methods for system realization and identification. In *Proceedings of the 48th IEEE Conference on Decision and Control (CDC)*, pages 4676–4681. IEEE, 2009.

[74] C. Lu, T. Van Gestel, J.A.K. Suykens, S. Van Huffel, D. Timmerman, and I. Vergote. Classification of ovarian tumors using bayesian least squares support vector machines. *Artificial Intelligence in Medicine*, pages 219–228, 2003.

[75] J. Luts, F. Ojeda, R. Van de Plas, B. De Moor, S. Van Huffel, and J.A.K. Suykens. A tutorial on support vector machine-based methods for classification problems in chemometrics. *Analytica Chimica Acta*, 665(2):129, 2010.

[76] D.J.C. MacKay. Bayesian interpolation. *Neural Computation*, 4:415–447, 1992.

[77] D.J.C. MacKay. The evidence framework applied to classification networks. *Neural computation*, 4(5):720–736, 1992.

[78] D.J.C. MacKay. Probable networks and plausible predictions-a review of practical bayesian methods for supervised neural networks. *Network: Computation in Neural Systems*, 6(3):469–505, 1995.

[79] D.J.C. MacKay. Introduction to gaussian processes. *NATO ASI Series F Computer and Systems Sciences*, 168:133–166, 1998.

[80] D.W. Marquardt. Generalized Inverses, Ridge Regression, Biased Linear Estimation, and Nonlinear Estimation. *Technometrics*, 12(3):591–612, 1970.

[81] D. Mattera and S. Haykin. Support vector machines for dynamic reconstruction of a chaotic system. In *Advances in kernel methods*, pages 211–241. MIT Press, 1999.

[82] J. Mercer. Functions of positive and negative type, and their connection with the theory of integral equations. *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, pages 415–446, 1909.

[83] C.A. Micchelli and M. Pontil. Learning the Kernel Function via Regularization. *Journal of Machine Learning Research*, 6:1099–1125, 2005.

[84] C.A. Micchelli and M. Pontil. Feature space perspectives for learning the kernel. *Machine Learning*, 66(2):297–319, 2007.

[85] S. Mika, G. Ratsch, J. Weston, B. Schölkopf, and KR Mullers. Fisher discriminant analysis with kernels. In *Neural Networks for Signal Processing IX, 1999. Proceedings of the 1999 IEEE Signal Processing Society Workshop.*, pages 41–48. IEEE, 1999.

[86] E.H. Moore. On properly positive hermitian matrices. *Bull. Amer. Math. Soc*, 23(59):66–67, 1916.

[87] S. Mukherjee, E. Osuna, and F. Girosi. Nonlinear prediction of chaotic time series using support vector machines. In *Neural Networks for Signal Processing [1997] VII. Proceedings of the 1997 IEEE Workshop*, pages 511–520. IEEE, 1997.

[88] K.R. Müller, S. Mika, G. Ratsch, K. Tsuda, and B. Schölkopf. An introduction to kernel-based learning algorithms. *IEEE transactions on neural networks*, 12(2):181–201, 2001.

[89] K.R. Müller, A. Smola, G. Rätsch, B. Schölkopf, J. Kohlmorgen, and V. Vapnik. Predicting time series with support vector machines. *Artificial Neural Networks—ICANN'97*, pages 999–1004, 1997.

[90] J. Nocedal and S.J. Wright. *Numerical Optimization.* Springer, 1999.

[91] F. Ojeda, M. Signoretto, R. Van de Plas, E. Waelkens, B. De Moor, and J. A. K. Suykens. Semi-supervised learning of sparse linear models in mass spectral imaging. In *Pattern Recognition in Bioinformatics (PRIB)*, pages 325–334, 2010, Nijgmegen, The Netherlands.

[92] K. Pelckmans, J. De Brabanter, J.A.K. Suykens, and B. De Moor. The differogram: Non-parametric noise variance estimation and its use for model selection. *Neurocomputing*, 69(1):100–122, 2005.

[93] J. Platt. Fast training of support vector machines using sequential minimal optimization. In B. Schölkopf, C. J. C. Burges, and A. J. Smola, editors, *Advances in Kernel Methods — Support Vector Learning*, pages 185–208, Cambridge, MA, 1999. MIT Press.

[94] T. Poggio and F. Girosi. Networks for approximation and learning. *Proceedings of the IEEE*, 78(9):1481–1497, 1990.

[95] C.E. Rasmussen and C.K.I. Williams. *Gaussian processes for machine learning*, volume 1. MIT press Cambridge, MA, 2006.

[96] F. Riesz and B.S. Nagy. *Functional Analysis.* Frederick Ungar Publishing Co., New York, 1955.

[97] R.M. Rifkin and R.A. Lippert. Notes on regularized least squares. *MIT-CSAIL-TR-2007-025 CBCL-268*, 2007.

[98] K. Saadi, G. C. Cawley, and N.L.C. Talbot. Fast exact leave-one-out cross-validation of least-square support vector machines. In *European Symposium on Artificial Neural Networks (ESANN-2002)*, 2002.

[99] S. Saitoh. *Integral transforms, reproducing kernels and their applications*, volume 369. Chapman & Hall/CRC, 1997.

[100] C. Saunders, A. Gammerman, and V. Vovk. Ridge regression learning algorithm in dual variables. In *International Conference on Machine Learning (ICML)*, pages 515–521, 1998.

[101] B. Schölkopf, R. Herbrich, and A. J. Smola. A generalized representer theorem. *Proceedings of the Annual Conference on Computational Learning Theory (COLT)*, pages 416–426, 2001.

[102] B. Schölkopf, A. Smola, and K.-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10:1299–1319, 1998.

[103] B. Schölkopf and A. J. Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond.* MIT Press, 2002.

[104] B. Schölkopf, A.J. Smola, R.C. Williamson, and P.L. Bartlett. New support vector algorithms. *Neural computation*, 12(5):1207–1245, 2000.

[105] J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis.* Cambridge University Press, 2004.

[106] N. Shawe-Taylor and A. Kandola. On kernel target alignment. In *Advances in Neural Information Processing Systems 14*, volume 1, page 367, 2002.

[107] P.K. Shivaswamy and T. Jebara. Relative margin machines. *Advances in Neural Information Processing Systems*, 21(1-8):7, 2008.

[108] P.K. Shivaswamy and T. Jebara. Maximum relative margin and data-dependent regularization. *The Journal of Machine Learning Research*, 11:747–788, 2010.

[109] M. Signoretto, L. De Lathauwer, and J. A. K. Suykens. A kernel-based framework to tensorial data analysis. *Neural networks*, 24(8):861—874, 2011.

[110] M. Signoretto, E. Olivetti, L. De Lathauwer, and J. A. K. Suykens. Classification of multichannel signals with cumulant-based kernels. *IEEE Transactions on Signal Processing*, 60(5):2304–2314, 2012.

[111] M. Signoretto and J.A.K. Suykens. Convex estimation of cointegrated var models by a nuclear norm penalty. In *Proc. of the 16th IFAC Symposium on System Identification (SYSID 2012)*, 2012.

[112] V. Sindhwani, P. Niyogi, and M. Belkin. Beyond the point cloud: from transductive to semi-supervised learning. In *International Conference on Machine Learning (ICML)*, volume 22, pages 824–831, 2005.

[113] A.J. Smola and B. Schölkopf. A tutorial on support vector regression. *Statistics and computing*, 14(3):199–222, 2004.

[114] I. Steinwart. On the influence of the kernel on the consistency of support vector machines. *The Journal of Machine Learning Research*, 2:67–93, 2002.

[115] I. Steinwart and A. Christmann. *Support vector machines.* Springer Verlag, 2008.

[116] I. Steinwart, D. Hush, and C. Scovel. An explicit description of the reproducing kernel hilbert spaces of gaussian rbf kernels. *IEEE Trans. Inform. Theory*, 52:4635–4643, 2006.

[117] J.A.K. Suykens. Data visualization and dimensionality reduction using kernel maps with a reference point. *IEEE Transactions on Neural Networks*, 19(9):1501–1517, 2008.

[118] J.A.K. Suykens, C. Alzate, and K. Pelckmans. Primal and dual model representations in kernel-based learning. *Statistics Surveys*, 4:148–183. DOI: 10.1214/09–SS052, 2010.

[119] J.A.K. Suykens, J. De Brabanter, L. Lukas, and J. Vandewalle. Weighted least squares support vector machines: robustness and sparse approximation. *Neurocomputing*, 48(1):85–105, 2002.

[120] J.A.K. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor, and J. Vandewalle. *Least squares support vector machines*. World Scientific, 2002.

[121] J.A.K. Suykens, T. Van Gestel, J. Vandewalle, and B. De Moor. A support vector machine formulation to PCA analysis and its kernel version. *Neural Networks, IEEE Transactions on*, 14(2):447–450, 2003.

[122] J.A.K. Suykens, T. Van Gestel, J. Vandewalle, and B. De Moor. A support vector machine formulation to pca analysis and its kernel version. *Neural Networks, IEEE Transactions on*, 14(2):447–450, 2003.

[123] J.A.K. Suykens and J. Vandewalle. Least squares support vector machine classifiers. *Neural Processing Letters*, 9(3):293–300, 1999.

[124] J.A.K. Suykens and J. Vandewalle. Recurrent least squares support vector machines. *Circuits and Systems I: Fundamental Theory and Applications, IEEE Transactions on*, 47(7):1109–1114, 2000.

[125] J.A.K. Suykens, J. Vandewalle, and B. De Moor. Optimal control by least squares support vector machines. *Neural Networks*, 14(1):23–35, 2001.

[126] S. Thrun. *Learning to learn*, chapter Life-long Learning Algorithms. Springer, 1998.

[127] R. Tibshirani. Regression Shrinkage and Selection via the LASSO. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1):267–288, 1996.

[128] A.N. Tikhonov. On the stability of inverse problems. In *Dokl. Akad. Nauk SSSR*, volume 39, pages 195–198, 1943.

[129] A.N. Tikhonov. Solution of incorrectly formulated problems and the regularization method. In *Soviet Math. Dokl.*, volume 5, page 1035, 1963.

[130] A.N. Tikhonov and V.Y. Arsenin. *Solutions of Ill-posed Problems*. W. H. Winston, Washington, D.C., 1977.

[131] K. Tsuda, S. Akaho, M. Kawanabe, and K.R. Müller. Asymptotic properties of the Fisher kernel. *Neural Computation*, 16(1):115–137, 2004.

[132] V. Van Belle, K. Pelckmans, S. Van Huffel, and J.A.K. Suykens. Improved performance on high-dimensional survival data by application of survival-SVM. *Bioinformatics*, 27(1):87–94, 2011.

[133] V. Van Belle, K. Pelckmans, S. Van Huffel, and J.A.K. Suykens. Support vector methods for survival analysis: a comparison between ranking and regression approaches. *Artificial Intelligence in Medicine*, 53(2):107–118, 2011.

[134] T. Van Gestel, J.A.K. Suykens, B. Baesens, S. Viaene, J. Vanthienen, G. Dedene, B. De Moor, and J. Vandewalle. Benchmarking least squares support vector machine classifiers. *Machine Learning*, 54(1):5–32, 2004.

[135] T. Van Gestel, J.A.K. Suykens, D.E. Baestaens, A. Lambrechts, G. Lanckriet, B. Vandaele, B. De Moor, and J. Vandewalle. Financial time series prediction using least squares support vector machines within the evidence framework. *IEEE Transactions on Neural Networks*, 12(4):809–821, 2001.

[136] T. Van Gestel, J.A.K. Suykens, J. De Brabanter, B. De Moor, and J. Vandewalle. Kernel canonical correlation analysis and least squares support vector machines. *Artificial Neural Networks—ICANN 2001*, pages 384–389, 2001.

[137] V. Vapnik. Pattern recognition using generalized portrait method. *Automation and Remote Control*, 24:774–780, 1963.

[138] V. Vapnik. *Estimation of Dependences Based on Empirical Data [in Russian]*. Nauka, Moscow, 1979. (English translation: Springer Verlag, New York, 1982).

[139] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer Verlag, New York, 1995.

[140] V. Vapnik. Transductive inference and semi-supervised learning. In O. Chapelle, B. Schölkopf, and A. Zien, editors, *Semi-supervised learning*, 2006.

[141] V. Vapnik and A. Chervonenkis. A note on one class of perceptrons. *Automation and Remote Control*, 25(1), 1964.

[142] V. Vapnik and A. Chervonenkis. Uniform convergence of frequencies of occurrence of events to their probabilities. *Doklady Akademii Nauk SSSR*, 181:915—-918, 1968.

[143] V. Vapnik and A. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and Its Applications*, 16(2):264—-280, 1971.

[144] V. Vapnik and A. Chervonenkis. *Theory of Pattern Recognition [in Russian]*. Nauka, Moscow, 1974. (German Translation: W. Wapnik & A. Tscherwonenkis, *Theorie der Zeichenerkennung*, Akademie-Verlag, Berlin, 1979).

[145] V. Vapnik and A. Chervonenkis. The necessary and sufficient conditions for consistency in the empirical risk minimization method. *Pattern Recognition and Image Analysis*, 1(3):283–305, 1991.

[146] S.V.N. Vishwanathan, N.N. Schraudolph, R. Kondor, and K.M. Borgwardt. Graph kernels. *Journal of Machine Learning Research*, 11:1201–1242, 2010.

[147] S.V.N. Vishwanathan, A.J. Smola, and R. Vidal. Binet-Cauchy kernels on dynamical systems and its application to the analysis of dynamic scenes. *International Journal of Computer Vision*, 73(1):95–119, 2007.

[148] G. Wahba. *Spline Models for Observational Data*, volume 59 of *CBMS-NSF Regional Conference Series in Applied Mathematics*. SIAM, Philadelphia, 1990.

[149] N. Weiner. *Extrapolation, interpolation, and smoothing of stationary time series with engineering applications.* MIT press, 1949.

[150] D. Widjaja, C. Varon, A.C. Dorado, J.A.K. Suykens, and S. Van Huffel. Application of kernel principal component analysis for single lead ecg-derived respiration. *IEEE Transactions on Biomedical Engineering*, 59(4):1169—1176, 2012.

[151] C.K.I. Williams and C.E. Rasmussen. Gaussian processes for regression. 1996.

[152] C.K.I. Williams and M. Seeger. Using the Nyström method to speed up kernel machines. *Advances in neural information processing systems 15*, pages 682–688, 2001.

[153] S. Yu, L.C. Tranchevent, B. Moor, and Y. Moreau. *Kernel-based data fusion for machine learning: methods and applications in bioinformatics and text mining*, volume 345 of *Studies in Computational Intelligence*. Springer, 2011.

[154] M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society Series B*, 68(1):49–67, 2006.

[155] P. Zhao, G. Rocha, and B. Yu. The composite absolute penalties family for grouped and hierarchical variable selection. *Annals of Statistics*, 37:3468–3497, 2009.