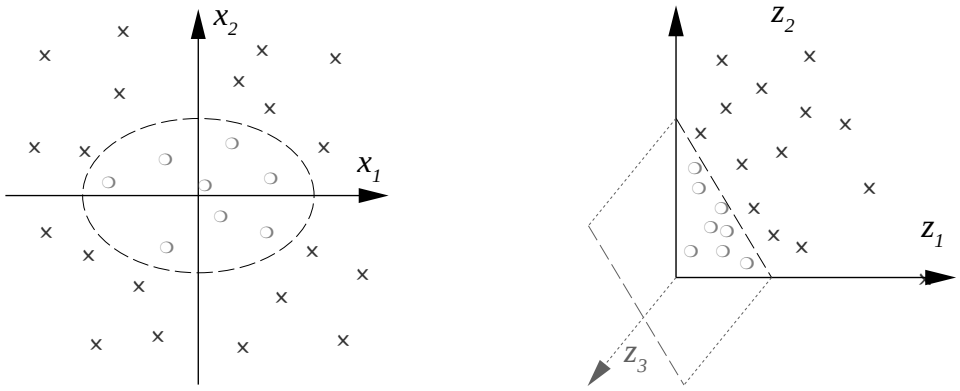

Figure 2.1 Toy example of a binary classification problem mapped into feature space. We assume that the true decision boundary is an ellipse in input space (left panel). The task of the learning process is to estimate this boundary based on empirical data consisting of training points in both classes (crosses and circles, respectively). When mapped into feature space via the nonlinear map $\Phi_2(x) = (z_1, z_2, z_3) = ([x]_1^2, [x]_2^2, \sqrt{2}\,[x]_1[x]_2)$ (right panel), the ellipse becomes a hyperplane (in the present simple case, it is parallel to the z_3 axis, hence all points are plotted in the (z_1, z_2) plane). This is due to the fact that ellipses can be written as linear equations in the entries of (z_1, z_2, z_3). Therefore, in feature space, the problem reduces to that of estimating a hyperplane from the mapped data points. Note that via the polynomial kernel (see (2.12) and (2.13)), the dot product in the three-dimensional space can be computed without computing Φ_2. Later in the book, we shall describe algorithms for constructing hyperplanes which are based on dot products (Chapter 7).

2.2 The Representation of Similarities in Linear Spaces

In what follows, we will look at things the other way round, and start with the kernel rather than with the feature map. Given some kernel, can we construct a feature space such that the kernel computes the dot product in that feature space; that is, such that (2.2) holds? This question has been brought to the attention of the machine learning community in a variety of contexts, especially during recent years [4, 152, 62, 561, 480]. In functional analysis, the same problem has been studied under the heading of *Hilbert space representations* of kernels. A good monograph on the theory of kernels is the book of Berg, Christensen, and Ressel [42]; indeed, a large part of the material in the present chapter is based on this work. We do not aim to be fully rigorous; instead, we try to provide insight into the basic ideas. As a rule, all the results that we state without proof can be found in [42]. Other standard references include [16, 455].

There is one more aspect in which this section differs from the previous one: the latter dealt with vectorial data, and the domain $\mathcal{X}$ was assumed to be a subset of $\mathbb{R}^N$. By contrast, the results in the current section hold for data drawn from domains which need no structure, other than their being nonempty sets. This generalizes kernel learning algorithms to a large number of situations where a vectorial representation is not readily available, and where one directly works

with pairwise distances or similarities between non-vectorial objects [246, 467, 154, 210, 234, 585]. This theme will recur in several places throughout the book, for instance in Chapter 13.

2.2.1 Positive Definite Kernels

We start with some basic definitions and results. As in the previous chapter, indices i and j are understood to run over $1, \ldots, m$.

Gram Matrix

Definition 2.3 (Gram Matrix) *Given a function $k : \mathcal{X}^2 \to \mathbb{K}$ (where $\mathbb{K} = \mathbb{C}$ or $\mathbb{K} = \mathbb{R}$) and patterns $x_1, \ldots, x_m \in \mathcal{X}$, the $m \times m$ matrix K with elements*

$$K_{ij} := k(x_i, x_j) \tag{2.14}$$

is called the Gram matrix *(or* kernel matrix*) of k with respect to $x_1, \ldots, x_m$.*

PD Matrix

Definition 2.4 (Positive Definite Matrix) *A complex $m \times m$ matrix K satisfying*

$$\sum_{i,j} c_i \bar{c}_j K_{ij} \geq 0 \tag{2.15}$$

for all $c_i \in \mathbb{C}$ is called positive definite.[1] *Similarly, a real symmetric $m \times m$ matrix K satisfying (2.15) for all $c_i \in \mathbb{R}$ is called* positive definite*.*

Note that a symmetric matrix is positive definite if and only if all its eigenvalues are nonnegative (Problem 2.4). The left hand side of (2.15) is often referred to as the *quadratic form* induced by K.

PD Kernel

Definition 2.5 ((Positive Definite) Kernel) *Let $\mathcal{X}$ be a nonempty set. A function k on $\mathcal{X} \times \mathcal{X}$ which for all $m \in \mathbb{N}$ and all $x_1 \ldots, x_m \in \mathcal{X}$ gives rise to a positive definite Gram matrix is called a* positive definite (pd) kernel*. Often, we shall refer to it simply as a* kernel*.*

Remark 2.6 (Terminology) *The term* kernel *stems from the first use of this type of function in the field of integral operators as studied by Hilbert and others [243, 359, 112]. A function k which gives rise to an operator T_k via*

$$(T_k f)(x) = \int_{\mathcal{X}} k(x, x') f(x')\, dx' \tag{2.16}$$

is called the kernel of T_k.

In the literature, a number of different terms are used for positive definite kernels, such as reproducing kernel*,* Mercer kernel*,* admissible kernel*,* Support Vector kernel*,* nonnegative definite kernel*, and* covariance function*. One might argue that the term* positive definite kernel *is slightly misleading. In matrix theory, the term* definite *is sometimes reserved for the case where equality in (2.15) only occurs if $c_1 = \ldots = c_m = 0$.*

1. The bar in $\bar{c}_j$ denotes complex conjugation; for real numbers, it has no effect.

Simply using the term positive kernel*, on the other hand, could be mistaken as referring to a kernel whose* values *are positive. Finally, the term positive* semi*definite kernel becomes rather cumbersome if it is to be used throughout a book. Therefore, we follow the convention used for instance in [42], and employ the term* positive definite *both for kernels and matrices in the way introduced above. The case where the value* 0 *is only attained if all coefficients are* 0 *will be referred to as* strictly *positive definite.*

We shall mostly use the term kernel*. Whenever we want to refer to a kernel $k(x, x')$ which is not positive definite in the sense stated above, it will be clear from the context.*

The definitions for positive definite kernels and positive definite matrices differ in the fact that in the former case, we are free to choose the points on which the kernel is evaluated — for every choice, the kernel induces a positive definite matrix.

Positive definiteness implies *positivity on the diagonal* (Problem 2.12),

$$k(x, x) \geq 0 \text{ for all } x \in \mathcal{X}, \tag{2.17}$$

and *symmetry* (Problem 2.13),

$$k(x_i, x_j) = \overline{k(x_j, x_i)}. \tag{2.18}$$

To also cover the complex-valued case, our definition of symmetry includes complex conjugation. The definition of symmetry of matrices is analogous; that is, $K_{ij} = \overline{K}_{ji}$.

Real-Valued Kernels

For real-valued kernels it is not sufficient to stipulate that (2.15) hold for real coefficients c_i. To get away with real coefficients only, we must additionally require that the kernel be symmetric (Problem 2.14); $k(x_i, x_j) = k(x_j, x_i)$ (cf. Problem 2.13).

It can be shown that whenever k is a (complex-valued) positive definite kernel, its real part is a (real-valued) positive definite kernel. Below, we shall largely be dealing with real-valued kernels. Most of the results, however, also apply for complex-valued kernels.

Kernels can be regarded as generalized dot products. Indeed, any dot product is a kernel (Problem 2.5); however, linearity in the arguments, which is a standard property of dot products, does not carry over to general kernels. However, another property of dot products, the Cauchy-Schwarz inequality, does have a natural generalization to kernels:

Proposition 2.7 (Cauchy-Schwarz Inequality for Kernels) *If k is a positive definite kernel, and $x_1, x_2 \in \mathcal{X}$, then*

$$|k(x_1, x_2)|^2 \leq k(x_1, x_1) \cdot k(x_2, x_2). \tag{2.19}$$

Proof For sake of brevity, we give a non-elementary proof using some basic facts of linear algebra. The 2×2 Gram matrix with entries $K_{ij} = k(x_i, x_j)$ $(i, j \in \{1, 2\})$ is positive definite. Hence both its eigenvalues are nonnegative, and so is their product, the determinant of K. Therefore

$$0 \leq K_{11}K_{22} - K_{12}K_{21} = K_{11}K_{22} - K_{12}\overline{K}_{12} = K_{11}K_{22} - |K_{12}|^2. \tag{2.20}$$

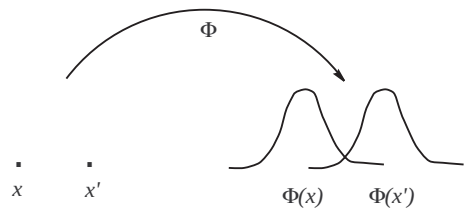

Figure 2.2 One instantiation of the feature map associated with a kernel is the map (2.21), which represents each pattern (in the picture, x or x') by a kernel-shaped *function* sitting on the pattern. In this sense, each pattern is represented by its similarity to *all* other patterns. In the picture, the kernel is assumed to be bell-shaped, e.g., a Gaussian $k(x, x') = \exp(-\|x - x'\|^2/(2\,\sigma^2))$. In the text, we describe the construction of a dot product $\langle .,. \rangle$ on the function space such that $k(x, x') = \langle \Phi(x), \Phi(x') \rangle$.

Substituting $k(x_i, x_j)$ for K_{ij}, we get the desired inequality. ■

We now show how the feature spaces in question are defined by the choice of kernel function.

2.2.2 The Reproducing Kernel Map

Feature Map

Assume that k is a real-valued positive definite kernel, and $\mathcal{X}$ a nonempty set. We define a map from $\mathcal{X}$ into the space of functions mapping $\mathcal{X}$ into $\mathbb{R}$, denoted as $\mathbb{R}^{\mathcal{X}} := \{f : \mathcal{X} \to \mathbb{R}\}$, via

$$\begin{aligned} \Phi : \mathcal{X} &\to \mathbb{R}^{\mathcal{X}} \\ x &\mapsto k(., x). \end{aligned} \tag{2.21}$$

Here, $\Phi(x)$ denotes the function that assigns the value $k(x', x)$ to $x' \in \mathcal{X}$, i.e., $\Phi(x)(.) = k(., x)$ (as shown in Figure 2.2).

We have thus turned each pattern into a function on the domain $\mathcal{X}$. In this sense, a pattern is now represented by its similarity to *all* other points in the input domain $\mathcal{X}$. This seems a very rich representation; nevertheless, it will turn out that the kernel allows the computation of the dot product in this representation. Below, we show how to construct a feature space associated with Φ, proceeding in the following steps:

1. Turn the image of Φ into a vector space,
2. define a dot product; that is, a strictly positive definite bilinear form, and
3. show that the dot product satisfies $k(x, x') = \langle \Phi(x), \Phi(x') \rangle$.

Vector Space

We begin by constructing a dot product space containing the images of the input patterns under Φ. To this end, we first need to define a vector space. This is done by taking linear combinations of the form

$$f(.) = \sum_{i=1}^{m} \alpha_i k(., x_i). \tag{2.22}$$

Here, $m \in \mathbb{N}$, $\alpha_i \in \mathbb{R}$ and $x_1, \ldots, x_m \in \mathcal{X}$ are arbitrary. Next, we define a dot product

between f and another function

$$g(.) = \sum_{j=1}^{m'} \beta_j k(., x'_j), \tag{2.23}$$

Dot Product

where $m' \in \mathbb{N}$, $\beta_j \in \mathbb{R}$ and $x'_1, \ldots, x'_{m'} \in \mathcal{X}$, as

$$\langle f, g \rangle := \sum_{i=1}^{m} \sum_{j=1}^{m'} \alpha_i \beta_j k(x_i, x'_j). \tag{2.24}$$

This expression explicitly contains the expansion coefficients, which need not be unique. To see that it is nevertheless well-defined, note that

$$\langle f, g \rangle = \sum_{j=1}^{m'} \beta_j f(x'_j), \tag{2.25}$$

using $k(x'_j, x_i) = k(x_i, x'_j)$. The sum in (2.25), however, does not depend on the particular expansion of f. Similarly, for g, note that

$$\langle f, g \rangle = \sum_{i=1}^{m} \alpha_i g(x_i). \tag{2.26}$$

The last two equations also show that $\langle \cdot, \cdot \rangle$ is bilinear. It is symmetric, as $\langle f, g \rangle = \langle g, f \rangle$. Moreover, it is positive definite, since positive definiteness of k implies that for any function f, written as (2.22), we have

$$\langle f, f \rangle = \sum_{i,j=1}^{m} \alpha_i \alpha_j k(x_i, x_j) \geq 0. \tag{2.27}$$

The latter implies that $\langle \cdot, \cdot \rangle$ is actually itself a positive definite kernel, defined on our space of functions. To see this, note that given functions $f_1, \ldots, f_n$, and coefficients $\gamma_1, \ldots, \gamma_n \in \mathbb{R}$, we have

$$\sum_{i,j=1}^{n} \gamma_i \gamma_j \langle f_i, f_j \rangle = \left\langle \sum_{i=1}^{n} \gamma_i f_i, \sum_{j=1}^{n} \gamma_j f_j \right\rangle \geq 0. \tag{2.28}$$

Here, the left hand equality follows from the bilinearity of $\langle \cdot, \cdot \rangle$, and the right hand inequality from (2.27). For the last step in proving that it qualifies as a dot product, we will use the following interesting property of Φ, which follows directly from the definition: for all functions (2.22), we have

$$\langle k(., x), f \rangle = f(x) \tag{2.29}$$

— k is the *representer of evaluation*. In particular,

$$\langle k(., x), k(., x') \rangle = k(x, x'). \tag{2.30}$$

Reproducing Kernel

By virtue of these properties, positive definite kernels k are also called *reproducing kernels* [16, 42, 455, 578, 467, 202]. By (2.29) and Proposition 2.7, we have

$$|f(x)|^2 = |\langle k(., x), f \rangle|^2 \leq k(x, x) \cdot \langle f, f \rangle . \tag{2.31}$$

Therefore, $\langle f, f \rangle = 0$ directly implies $f = 0$, which is the last property that required proof in order to establish that $\langle \cdot, \cdot \rangle$ is a dot product (cf. Section B.2).

The case of complex-valued kernels can be dealt with using the same construction; in that case, we will end up with a complex dot product space [42].

The above reasoning has shown that any positive definite kernel can be thought of as a dot product in another space: in view of (2.21), the reproducing kernel property (2.30) amounts to

$$\langle \Phi(x), \Phi(x') \rangle = k(x, x'). \tag{2.32}$$

Therefore, the dot product space $\mathcal{H}$ constructed in this way is one possible instantiation of the feature space associated with a kernel.

Kernels from Feature Maps

Above, we have started with the kernel, and constructed a feature map. Let us now consider the opposite direction. Whenever we have a mapping Φ from $\mathcal{X}$ into a dot product space, we obtain a positive definite kernel via $k(x, x') := \langle \Phi(x), \Phi(x') \rangle$. This can be seen by noting that for all $c_i \in \mathbb{R}, x_i \in \mathcal{X}, i = 1, \ldots, m$, we have

$$\sum_{i,j} c_i c_j k(x_i, x_j) = \left\langle \sum_i c_i \Phi(x_i), \sum_j c_j \Phi(x_j) \right\rangle = \left\| \sum_i c_i \Phi(x_i) \right\|^2 \geq 0, \tag{2.33}$$

due to the nonnegativity of the norm.

Equivalent Definition of PD Kernels

This has two consequences. First, it allows us to give an equivalent definition of positive definite kernels as functions with the property that there exists a map Φ into a dot product space such that (2.32) holds true. Second, it allows us to construct kernels from feature maps. For instance, it is in this way that powerful linear representations of 3D heads proposed in computer graphics [575, 59] give rise to kernels. The identity (2.32) forms the basis for the kernel trick:

Kernel Trick

Remark 2.8 ("Kernel Trick") *Given an algorithm which is formulated in terms of a positive definite kernel k, one can construct an alternative algorithm by replacing k by another positive definite kernel $\tilde{k}$.*

In view of the material in the present section, the justification for this procedure is the following: effectively, the original algorithm can be thought of as a dot product based algorithm operating on vectorial data $\Phi(x_1), \ldots, \Phi(x_m)$. The algorithm obtained by replacing k by $\tilde{k}$ then is exactly the same dot product based algorithm, only that it operates on $\tilde{\Phi}(x_1), \ldots, \tilde{\Phi}(x_m)$.

The best known application of the kernel trick is in the case where k is the dot product in the input domain (cf. Problem 2.5). The trick is not limited to that case, however: k and $\tilde{k}$ can *both* be nonlinear kernels. In general, care must be exercised in determining whether the resulting algorithm will be useful: sometimes, an algorithm will only work subject to additional conditions on the input data, e.g., the data set might have to lie in the positive orthant. We shall later see that certain kernels induce feature maps which enforce such properties for the mapped data (cf. (2.73)), and that there are algorithms which take advantage of these aspects (e.g., in Chapter 8). In such cases, not every conceivable positive definite kernel

will make sense.

Historical Remarks

Even though the kernel trick had been used in the literature for a fair amount of time [4, 62], it took until the mid 1990s before it was explicitly stated that *any* algorithm that only depends on dot products, i.e., any algorithm that is rotationally invariant, can be kernelized [479, 480]. Since then, a number of algorithms have benefitted from the kernel trick, such as the ones described in the present book, as well as methods for clustering in feature spaces [479, 215, 199].

Moreover, the machine learning community took time to comprehend that the definition of kernels on general sets (rather than dot product spaces) greatly extends the applicability of kernel methods [467], to data types such as texts and other sequences [234, 585, 23]. Indeed, this is now recognized as a crucial feature of kernels: they lead to an embedding of general data types in linear spaces.

Not surprisingly, the history of methods for representing kernels in linear spaces (in other words, the mathematical counterpart of the kernel trick) dates back significantly further than their use in machine learning. The methods appear to have first been studied in the 1940s by Kolmogorov [304] for countable $\mathcal{X}$ and Aronszajn [16] in the general case. Pioneering work on linear representations of a related class of kernels, to be described in Section 2.4, was done by Schoenberg [465]. Further bibliographical comments can be found in [42].

We thus see that the mathematical basis for kernel algorithms has been around for a long time. As is often the case, however, the practical importance of mathematical results was initially underestimated.[2]

2.2.3 Reproducing Kernel Hilbert Spaces

In the last section, we described how to define a space of functions which is a valid realization of the feature spaces associated with a given kernel. To do this, we had to make sure that the space is a vector space, and that it is endowed with a dot product. Such spaces are referred to as dot product spaces (cf. Appendix B), or equivalently as *pre-Hilbert* spaces. The reason for the latter is that one can turn them into Hilbert spaces (cf. Section B.3) by a fairly simple mathematical trick. This additional structure has some mathematical advantages. For instance, in Hilbert spaces it is always possible to define projections. Indeed, Hilbert spaces are one of the favorite concepts of functional analysis.

So let us again consider the pre-Hilbert space of functions (2.22), endowed with the dot product (2.24). To turn it into a Hilbert space (over $\mathbb{R}$), one *completes* it in the norm corresponding to the dot product, $\|f\| := \sqrt{\langle f, f\rangle}$. This is done by adding the limit points of sequences that are convergent in that norm (see Appendix B).

2. This is illustrated by the following quotation from an excellent machine learning textbook published in the seventies (p. 174 in [152]): *"The familiar functions of mathematical physics are eigenfunctions of symmetric kernels, and their use is often suggested for the construction of potential functions. However, these suggestions are more appealing for their mathematical beauty than their practical usefulness."*

RKHS

In view of the properties (2.29) and (2.30), this space is usually called a *reproducing kernel Hilbert space (RKHS)*.

In general, an RKHS can be defined as follows.

Definition 2.9 (Reproducing Kernel Hilbert Space) *Let $\mathcal{X}$ be a nonempty set (often called the index set) and by $\mathcal{H}$ a Hilbert space of functions $f : \mathcal{X} \to \mathbb{R}$. Then $\mathcal{H}$ is called a reproducing kernel Hilbert space endowed with the dot product $\langle \cdot, \cdot \rangle$ (and the norm $\|f\| := \sqrt{\langle f, f \rangle}$) if there exists a function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ with the following properties.*

Reproducing Property

1. *k has the reproducing property*[3]

$$\langle f, k(x, \cdot) \rangle = f(x) \text{ for all } f \in \mathcal{H}; \tag{2.34}$$

in particular,

$$\langle k(x, \cdot), k(x', \cdot) \rangle = k(x, x'). \tag{2.35}$$

Closed Space

2. *k spans $\mathcal{H}$, i.e. $\mathcal{H} = \overline{\text{span}\{k(x, \cdot) | x \in \mathcal{X}\}}$ where $\overline{X}$ denotes the completion of the set X (cf. Appendix B).*

On a more abstract level, an RKHS can be defined as a Hilbert space of functions f on $\mathcal{X}$ such that all evaluation functionals (the maps $f \mapsto f(x')$, where $x' \in \mathcal{X}$) are continuous. In that case, by the Riesz representation theorem (e.g., [429]), for each $x' \in \mathcal{X}$ there exists a unique function of x, called $k(x, x')$, such that

$$f(x') = \langle f, k(., x') \rangle . \tag{2.36}$$

It follows directly from (2.35) that $k(x, x')$ is symmetric in its arguments (see Problem 2.28) and satisfies the conditions for positive definiteness.

Uniqueness of k

Note that the RKHS uniquely determines k. This can be shown by contradiction: assume that there exist two kernels, say k and k', spanning the same RKHS $\mathcal{H}$. From Problem 2.28 we know that both k and k' must be symmetric. Moreover, from (2.34) we conclude that

$$\langle k(x, \cdot), k'(x', \cdot) \rangle_{\mathcal{H}} = k(x, x') = k'(x', x). \tag{2.37}$$

In the second equality we used the symmetry of the dot product. Finally, symmetry in the arguments of k yields $k(x, x') = k'(x, x')$ which proves our claim.

2.2.4 The Mercer Kernel Map

Section 2.2.2 has shown that any positive definite kernel can be represented as a dot product in a linear space. This was done by explicitly constructing a (Hilbert) space that does the job. The present section will construct another Hilbert space.

3. Note that this implies that each $f \in \mathcal{H}$ is actually a single function whose values at any $x \in \mathcal{X}$ are well-defined. In contrast, L_2 Hilbert spaces usually do not have this property. The elements of these spaces are equivalence classes of functions that disagree only on sets of measure 0; cf. footnote 15 in Section B.3.

One could argue that this is superfluous, given that any two separable Hilbert spaces are isometrically isomorphic, in other words, it is possible to define a one-to-one linear map between the spaces which preserves the dot product. However, the tool that we shall presently use, Mercer's theorem, has played a crucial role in the understanding of SVMs, and it provides valuable insight into the geometry of feature spaces, which more than justifies its detailed discussion. In the SVM literature, the kernel trick is usually introduced via Mercer's theorem.

Mercer's Theorem

We start by stating the version of Mercer's theorem given in [606]. We assume $(\mathcal{X}, \mu)$ to be a finite measure space.[4] The term *almost all* (cf. Appendix B) means *except for sets of measure zero*. For the commonly used Lebesgue-Borel measure, countable sets of individual points are examples of zero measure sets. Note that the integral with respect to a measure is explained in Appendix B. Readers who do not want to go into mathematical detail may simply want to think of the $d\mu(x')$ as a dx', and of $\mathcal{X}$ as a compact subset of $\mathbb{R}^N$. For further explanations of the terms involved in this theorem, cf. Appendix B, especially Section B.3.

Theorem 2.10 (Mercer [359, 307]) *Suppose $k \in L_\infty(\mathcal{X}^2)$ is a symmetric real-valued function such that the integral operator (cf. (2.16))*

$$\begin{aligned} T_k &: L_2(\mathcal{X}) \to L_2(\mathcal{X}) \\ (T_k f)(x) &:= \int_{\mathcal{X}} k(x, x') f(x')\, d\mu(x') \end{aligned} \tag{2.38}$$

is positive definite*; that is, for all $f \in L_2(\mathcal{X})$, we have*

$$\int_{\mathcal{X}^2} k(x, x') f(x) f(x')\, d\mu(x) d\mu(x') \geq 0. \tag{2.39}$$

Let $\psi_j \in L_2(\mathcal{X})$ be the normalized orthogonal eigenfunctions of T_k associated with the eigenvalues $\lambda_j > 0$, sorted in non-increasing order. Then

1. $(\lambda_j)_j \in \ell_1$,
2. $k(x, x') = \sum_{j=1}^{N_{\mathcal{H}}} \lambda_j \psi_j(x) \psi_j(x')$ *holds for almost all (x, x'). Either $N_{\mathcal{H}} \in \mathbb{N}$, or $N_{\mathcal{H}} = \infty$; in the latter case, the series converges absolutely and uniformly for almost all (x, x').*

For the converse of Theorem 2.10, see Problem 2.23. For a data-dependent approximation and its relationship to kernel PCA (Section 1.7), see Problem 2.26.

From statement 2 it follows that $k(x, x')$ corresponds to a dot product in $\ell_2^{N_{\mathcal{H}}}$, since $k(x, x') = \langle \Phi(x), \Phi(x') \rangle$ with

$$\begin{aligned} \Phi : \mathcal{X} &\to \ell_2^{N_{\mathcal{H}}} \\ x &\mapsto (\sqrt{\lambda_j} \psi_j(x))_{j=1,\ldots,N_{\mathcal{H}}}, \end{aligned} \tag{2.40}$$

for almost all $x \in \mathcal{X}$. Note that we use the same Φ as in (2.21) to denote the feature

4. A finite measure space is a set $\mathcal{X}$ with a σ-algebra (Definition B.1) defined on it, and a measure (Definition B.2) defined on the latter, satisfying $\mu(\mathcal{X}) < \infty$ (so that, up to a scaling factor, μ is a probability measure).

map, although the target spaces are different. However, this distinction is not important for the present purposes — we are interested in the existence of some Hilbert space in which the kernel corresponds to the dot product, and not in what particular representation of it we are using.

In fact, it has been noted [467] that the *uniform* convergence of the series implies that given any $\epsilon > 0$, there exists an $n \in \mathbb{N}$ such that even if $N_{\mathcal{H}} = \infty$, k can be approximated within accuracy ϵ as a dot product in $\mathbb{R}^n$: for almost all $x, x' \in \mathcal{X}$, $|k(x,x') - \langle \Phi^n(x), \Phi^n(x') \rangle| < \epsilon$, where $\Phi^n : x \mapsto (\sqrt{\lambda_1}\psi_1(x), \ldots, \sqrt{\lambda_n}\psi_n(x))$. The feature space can thus always be thought of as finite-dimensional within some accuracy ϵ. We summarize our findings in the following proposition.

Mercer Feature Map

Proposition 2.11 (Mercer Kernel Map) *If k is a kernel satisfying the conditions of Theorem 2.10, we can construct a mapping Φ into a space where k acts as a dot product,*

$$\langle \Phi(x), \Phi(x') \rangle = k(x,x'), \tag{2.41}$$

for almost all $x, x' \in \mathcal{X}$. Moreover, given any $\epsilon > 0$, there exists a map Φ_n into an n-dimensional dot product space (where $n \in \mathbb{N}$ depends on ϵ) such that

$$|k(x,x') - \langle \Phi^n(x), \Phi^n(x') \rangle| < \epsilon \tag{2.42}$$

for almost all $x, x' \in \mathcal{X}$.

Both Mercer kernels and positive definite kernels can thus be represented as dot products in Hilbert spaces. The following proposition, showing a case where the two types of kernels coincide, thus comes as no surprise.

Proposition 2.12 (Mercer Kernels are Positive Definite [359, 42]) *Let $\mathcal{X} = [a,b]$ be a compact interval and let $k : [a,b] \times [a,b] \to \mathbb{C}$ be continuous. Then k is a positive definite kernel if and only if*

$$\int_a^b \int_a^b k(x,x') f(x) f(x') \, dx \, dx' \geq 0 \tag{2.43}$$

for each continuous function $f : \mathcal{X} \to \mathbb{C}$.

Note that the conditions in this proposition are actually more restrictive than those of Theorem 2.10. Using the feature space representation (Proposition 2.11), however, it is easy to see that Mercer kernels are also positive definite (for almost all $x, x' \in \mathcal{X}$) in the more general case of Theorem 2.10: given any $\mathbf{c} \in \mathbb{R}^m$, we have

$$\sum_{i,j} c_i c_j k(x_i, x_j) = \sum_{i,j} c_i c_j \langle \Phi(x_i), \Phi(x_j) \rangle = \left\| \sum_i c_i \Phi(x_i) \right\|^2 \geq 0. \tag{2.44}$$

Being positive definite, Mercer kernels are thus also reproducing kernels.

We next show how the reproducing kernel map is related to the Mercer kernel map constructed from the eigenfunction decomposition [202, 467]. To this end, let us consider a kernel which satisfies the condition of Theorem 2.10, and construct

a dot product $\langle \cdot, \cdot \rangle$ such that k becomes a reproducing kernel for the Hilbert space $\mathcal{H}$ containing the functions

$$f(x) = \sum_{i=1}^{\infty} \alpha_i k(x, x_i) = \sum_{i=1}^{\infty} \alpha_i \sum_{j=1}^{N_{\mathcal{H}}} \lambda_j \psi_j(x) \psi_j(x_i). \tag{2.45}$$

By linearity, which holds for any dot product, we have

$$\langle f, k(., x') \rangle = \sum_{i=1}^{\infty} \alpha_i \sum_{j,n=1}^{N_{\mathcal{H}}} \lambda_j \psi_j(x_i) \langle \psi_j, \psi_n \rangle \lambda_n \psi_n(x'). \tag{2.46}$$

Since k is a Mercer kernel, the ψ_i ($i = 1, \ldots, N_{\mathcal{H}}$) can be chosen to be orthogonal with respect to the dot product in $L_2(\mathcal{X})$. Hence it is straightforward to choose $\langle \cdot, \cdot \rangle$ such that

$$\langle \psi_j, \psi_n \rangle = \delta_{jn} / \lambda_j \tag{2.47}$$

(using the Kronecker symbol δ_{jn}, see (B.30)), in which case (2.46) reduces to the reproducing kernel property (2.36) (using (2.45)). For a coordinate representation in the RKHS, see Problem 2.29.

Equivalence of Feature Spaces

The above connection between the Mercer kernel map and the RKHS map is instructive, but we shall rarely make use of it. In fact, we will usually *identify* the different feature spaces. Thus, to avoid confusion in subsequent chapters, the following comments are necessary. As described above, there are different ways of constructing feature spaces for any given kernel. In fact, they can even differ in terms of their dimensionality (cf. Problem 2.22). The two feature spaces that we will mostly use in this book are the RKHS associated with k (Section 2.2.2) and the Mercer ℓ_2 feature space. We will mostly use the same symbol $\mathcal{H}$ for all feature spaces that are associated with a given kernel. This makes sense provided that everything we do, at the end of the day, reduces to dot products. For instance, let us assume that Φ_1, Φ_2 are maps into the feature spaces $\mathcal{H}_1, \mathcal{H}_2$ respectively, both associated with the kernel k; in other words,

$$k(x, x') = \langle \Phi_i(x), \Phi_i(x') \rangle_{\mathcal{H}_i}, \text{ for } i = 1, 2. \tag{2.48}$$

Then it will usually *not* be the case that $\Phi_1(x) = \Phi_2(x)$; due to (2.48), however, we always have $\langle \Phi_1(x), \Phi_1(x') \rangle_{\mathcal{H}_1} = \langle \Phi_2(x), \Phi_2(x') \rangle_{\mathcal{H}_2}$. Therefore, as long as we are only interested in dot products, the two spaces can be considered identical.

An example of this identity is the so-called large margin regularizer that is usually used in SVMs, as discussed in the introductory chapter (cf. also Chapters 4 and 7),

$$\langle \mathbf{w}, \mathbf{w} \rangle, \text{ where } \mathbf{w} = \sum_{i=1}^{m} \alpha_i \Phi(x_i). \tag{2.49}$$

No matter whether Φ is the RKHS map $\Phi(x_i) = k(., x_i)$ (2.21) or the Mercer map $\Phi(x_i) = (\sqrt{\lambda_j} \psi_j(x))_{j=1,\ldots,N_{\mathcal{H}}}$ (2.40), the value of $\|\mathbf{w}\|^2$ will not change.

This point is of great importance, and we hope that all readers are still with us.

It is fair to say, however, that Section 2.2.5 can be skipped at first reading.

2.2.5 The Shape of the Mapped Data in Feature Space

Using Mercer's theorem, we have shown that one can think of the feature map as a map into a high- or infinite-dimensional Hilbert space. The argument in the remainder of the section shows that this typically entails that the mapped data $\Phi(\mathcal{X})$ lie in some box with rapidly decaying side lengths [606]. By this we mean that the range of the data decreases as the dimension index j increases, with a rate that depends on the size of the eigenvalues.

Let us assume that for all $j \in \mathbb{N}$, we have $\sup_{x\in\mathcal{X}} \lambda_j |\psi_j(x)|^2 < \infty$. Define the sequence

$$l_j := \sup_{x\in\mathcal{X}} \lambda_j |\psi_j(x)|^2. \tag{2.50}$$

Note that if

$$C_k := \sup_j \sup_{x\in\mathcal{X}} |\psi_j(x)| \tag{2.51}$$

exists (see Problem 2.24), then we have $l_j \le \lambda_j C_k^2$. However, if the λ_j decay rapidly, then (2.50) can be finite even if (2.51) is not.

By construction, $\Phi(\mathcal{X})$ is contained in an axis parallel parallelepiped in $\ell_2^{N_{\mathfrak{H}}}$ with side lengths $2\sqrt{l_j}$ (cf. (2.40)).[5]

Consider an example of a common kernel, the Gaussian, and let μ (see Theorem 2.10) be the Lebesgue measure. In this case, the eigenvectors are sine and cosine functions (with supremum one), and thus the sequence of the l_j coincides with the sequence of the eigenvalues λ_j. Generally, whenever $\sup_{x\in\mathcal{X}} |\psi_j(x)|^2$ is finite, the l_j decay as fast as the λ_j. We shall see in Sections 4.4, 4.5 and Chapter 12 that for many common kernels, this decay is very rapid.

It will be useful to consider operators that map $\Phi(\mathcal{X})$ into balls of some radius R centered at the origin. The following proposition characterizes a class of such operators, determined by the sequence $(l_j)_{j\in\mathbb{N}}$. Recall that $\mathbb{R}^{\mathbb{N}}$ denotes the space of all real sequences.

Proposition 2.13 (Mapping $\Phi(\mathcal{X})$ into ℓ_2) *Let S be the diagonal map*

$$\begin{array}{rccc} S: & \mathbb{R}^{\mathbb{N}} & \to & \mathbb{R}^{\mathbb{N}} \\ & (x_j)_j & \mapsto & S(x_j)_j = (s_j x_j)_j, \end{array} \tag{2.52}$$

where $(s_j)_j \in \mathbb{R}^{\mathbb{N}}$. If $\left(s_j\sqrt{l_j}\right)_j \in \ell_2$, then S maps $\Phi(\mathcal{X})$ into a ball centered at the origin whose radius is $R = \left\| \left(s_j\sqrt{l_j}\right)_j \right\|$.

5. In fact, it is sufficient to use the essential supremum in (2.50). In that case, subsequent statements also only hold true almost everywhere.

Proof Suppose $\left(s_j\sqrt{l_j}\right)_j \in \ell_2$. Using the Mercer map (2.40), we have

$$\|S\Phi(x)\|^2 = \sum_{j\in\mathbb{N}} s_j^2\lambda_j|\psi_j(x)|^2 \leq \sum_{j\in\mathbb{N}} s_j^2 l_j = R \tag{2.53}$$

for any $x \in \mathcal{X}$. Hence $S\Phi(\mathcal{X}) \subseteq \ell_2$. ■

The converse is not necessarily the case. To see this, note that if $\left(s_j\sqrt{l_j}\right)_j \notin \ell_2$, amounting to saying that

$$\sum_j s_j^2 \sup_{x\in\mathcal{X}} \lambda_j|\psi_j(x)|^2 \tag{2.54}$$

is not finite, then there need not always exist an $x \in \mathcal{X}$ such that $S\Phi(x) = \left(s_j\sqrt{\lambda_j}\psi_j(x)\right)_j \notin \ell_2$, i.e., that

$$\sum_j s_j^2\lambda_j|\psi_j(x)|^2 \tag{2.55}$$

is not finite.

To see how the freedom to rescale $\Phi(\mathcal{X})$ effectively restricts the class of functions we are using, we first note that everything in the feature space $\mathcal{H} = \ell_2^{N_\mathcal{H}}$ is done in terms of dot products. Therefore, we can compensate any invertible symmetric linear transformation of the data in $\mathcal{H}$ by the inverse transformation on the set of admissible weight vectors in $\mathcal{H}$. In other words, for any invertible symmetric operator S on $\mathcal{H}$, we have $\left\langle S^{-1}\mathbf{w}, S\Phi(x)\right\rangle = \langle \mathbf{w}, \Phi(x)\rangle$ for all $x \in \mathcal{X}$.

As we shall see below (cf. Theorem 5.5, Section 12.4, and Problem 7.5), there exists a class of generalization error bound that depends on the radius R of the smallest sphere containing the data. If the $(l_i)_i$ decay rapidly, we are not actually "making use" of the whole sphere. In this case, we may construct a diagonal scaling operator S which inflates the sides of the above parallelepiped as much as possible, while ensuring that it is still contained within a sphere of the original radius R in $\mathcal{H}$ (Figure 2.3). By effectively reducing the size of the function class, this will provide a way of strengthening the bounds. A similar idea, using kernel PCA (Section 14.2) to determine empirical scaling coefficients, has been successfully applied by [101].

We conclude this section with another useful insight that characterizes a property of the feature map Φ. Note that most of what was said so far applies to the case where the input domain $\mathcal{X}$ is a general set. In this case, it is not possible to make nontrivial statements about continuity properties of Φ. This changes if we assume $\mathcal{X}$ to be endowed with a notion of closeness, by turning it into a so-called topological space. Readers not familiar with this concept will be reassured to hear that Euclidean vector spaces are particular cases of topological spaces.

Continuity of Φ

Proposition 2.14 (Continuity of the Feature Map [402]) *If $\mathcal{X}$ is a topological space and k is a continuous positive definite kernel on $\mathcal{X}\times\mathcal{X}$, then there exists a Hilbert space $\mathcal{H}$ and a continuous map $\Phi : \mathcal{X} \to \mathcal{H}$ such that for all $x, x' \in \mathcal{X}$, we have $k(x, x') = \langle\Phi(x), \Phi(x')\rangle$.*

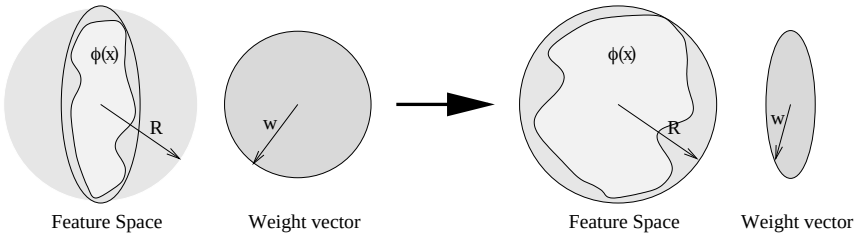

Figure 2.3 Since everything is done in terms of dot products, scaling up the data by an operator S can be compensated by scaling the weight vectors with S^{-1} (cf. text). By choosing S such that the data are still contained in a ball of the same radius R, we effectively reduce our function class (parametrized by the weight vector), which can lead to better generalization bounds, depending on the kernel inducing the map Φ.

2.2.6 The Empirical Kernel Map

The map Φ, defined in (2.21), transforms each input pattern into a function on $\mathcal{X}$, that is, into a potentially infinite-dimensional object. For any given set of points, however, it is possible to approximate Φ by only evaluating it on these points (cf. [232, 350, 361, 547, 474]):

Empirical Kernel Map

Definition 2.15 (Empirical Kernel Map) *For a given set* $\{z_1, \ldots, z_n\} \subset \mathcal{X}$, $n \in \mathbb{N}$, *we call*

$$\Phi_n : \mathbb{R}^N \to \mathbb{R}^n \text{ where } x \mapsto k(., x)|_{\{z_1,\ldots,z_n\}} = (k(z_1, x), \ldots, k(z_n, x))^\top \tag{2.56}$$

the empirical kernel map *w.r.t.* $\{z_1, \ldots, z_n\}$.

As an example, consider first the case where k is a positive definite kernel, and $\{z_1, \ldots, z_n\} = \{x_1, \ldots, x_m\}$; we thus evaluate $k(., x)$ on the training patterns. If we carry out a linear algorithm in feature space, then everything will take place in the linear span of the mapped training patterns. Therefore, we can represent the $k(., x)$ of (2.21) as $\Phi_m(x)$ without losing information. The dot product to use in that representation, however, is not simply the canonical dot product in $\mathbb{R}^m$, since the $\Phi(x_i)$ will usually not form an orthonormal system. To turn Φ_m into a feature map associated with k, we need to endow $\mathbb{R}^m$ with a dot product $\langle \cdot, \cdot \rangle_m$ such that

$$k(x, x') = \langle \Phi_m(x), \Phi_m(x') \rangle_m . \tag{2.57}$$

To this end, we use the ansatz $\langle \cdot, \cdot \rangle_m = \langle \cdot, M \cdot \rangle$, with M being a positive definite matrix.[6] Enforcing (2.57) on the training patterns, this yields the self-consistency condition [478, 512]

$$K = KMK, \tag{2.58}$$

6. Every dot product in $\mathbb{R}^m$ can be written in this form. We do not require strict definiteness of M, as the null space can be projected out, leading to a lower-dimensional feature space.

where K is the Gram matrix. The condition (2.58) can be satisfied for instance by the (pseudo-)inverse $M = K^{-1}$. Equivalently, we could have incorporated this rescaling operation, which corresponds to a Kernel PCA "whitening" ([478, 547, 474], cf. Section 11.4), directly into the map, by whitening (2.56) to get

Kernel PCA Map

$$\Phi_m^w : x \mapsto K^{-\frac{1}{2}}(k(x_1, x), \ldots, k(x_m, x)). \tag{2.59}$$

This simply amounts to dividing the eigenvector basis vectors of K by $\sqrt{\lambda_i}$, where the λ_i are the eigenvalues of K.[7] This parallels the rescaling of the eigenfunctions of the integral operator belonging to the kernel, given by (2.47). It turns out that this map can equivalently be performed using kernel PCA feature extraction (see Problem 14.8), which is why we refer to this map as the *kernel PCA map*.

Note that we have thus constructed a data-dependent feature map into an m-dimensional space which satisfies $\langle \Phi_m^w(x), \Phi_m^w(x') \rangle = k(x, x')$, i.e., we have found an m-dimensional feature space associated with the given kernel. In the case where K is invertible, $\Phi_m^w(x)$ computes the coordinates of $\Phi(x)$ when represented in a basis of the m-dimensional subspace spanned by $\Phi(x_1), \ldots, \Phi(x_m)$.

For data sets where the number of examples is smaller than their dimension, it can actually be computationally attractive to carry out Φ_m^w explicitly, rather than using kernels in subsequent algorithms. Moreover, algorithms which are not readily "kernelized" may benefit from explicitly carrying out the kernel PCA map.

We end this section with two notes which illustrate why the use of (2.56) need not be restricted to the special case we just discussed.

- *More general kernels.* When using non-symmetric kernels k in (2.56), together with the canonical dot product, we effectively work with the positive definite matrix $K^\top K$. Note that each positive definite matrix can be written as $K^\top K$. Therefore, working with positive definite kernels leads to an equally rich set of nonlinearities as working with an empirical kernel map using general non-symmetric kernels. If we wanted to carry out the whitening step, we would have to use $(K^\top K)^{-1/4}$ (cf. footnote 7 concerning potential singularities).
- *Different evaluation sets.* Things can be sped up by using expansion sets of the form $\{z_1, \ldots, z_n\}$, mapping into an n-dimensional space, with $n < m$, as done in [100, 228]. In that case, one modifies (2.59) to

$$\Phi_n^w : x \mapsto K_n^{-\frac{1}{2}}(k(z_1, x), \ldots, k(z_n, x)), \tag{2.60}$$

where $(K_n)_{ij} := k(z_i, z_j)$. The expansion set can either be a subset of the training set,[8] or some other set of points. We will later return to the issue of how to choose

7. It is understood that if K is singular, we use the pseudo-inverse of $K^{1/2}$ in which case we get an even lower dimensional subspace.
8. In [228] it is recommended that the size n of the expansion set is chosen large enough to ensure that the smallest eigenvalue of K_n is larger than some predetermined $\epsilon > 0$. Alternatively, one can start off with a larger set, and use kernel PCA to *select* the most important components for the map, see Problem 14.8. In the kernel PCA case, the map (2.60) is com-

the best set (see Section 10.2 and Chapter 18). As an aside, note that in the case of Kernel PCA (see Section 1.7 and Chapter 14 below), one does not need to worry about the whitening step in (2.59) and (2.60): using the canonical dot product in $\mathbb{R}^m$ (rather than $\langle \cdot, \cdot \rangle$) will simply lead to diagonalizing K^2 instead of K, which yields the same eigenvectors with squared eigenvalues. This was pointed out by [350, 361]. The study [361] reports experiments where (2.56) was employed to speed up Kernel PCA by choosing $\{z_1, \ldots, z_n\}$ as a subset of $\{x_1, \ldots, x_m\}$.

2.2.7 A Kernel Map Defined from Pairwise Similarities

In practice, we are given a finite amount of data $x_1, \ldots, x_m$. The following simple observation shows that even if we do not want to (or are unable to) analyze a given kernel k analytically, we can still compute a map Φ such that k corresponds to a dot product in the linear span of the $\Phi(x_i)$:

Proposition 2.16 (Data-Dependent Kernel Map [467]) *Suppose the data $x_1, \ldots, x_m$ and the kernel k are such that the kernel Gram matrix $K_{ij} = k(x_i, x_j)$ is positive definite. Then it is possible to construct a map Φ into an m-dimensional feature space $\mathcal{H}$ such that*

$$k(x_i, x_j) = \left\langle \Phi(x_i), \Phi(x_j) \right\rangle. \tag{2.61}$$

Conversely, given an arbitrary map Φ into some feature space $\mathcal{H}$, the matrix $K_{ij} = \left\langle \Phi(x_i), \Phi(x_j) \right\rangle$ is positive definite.

Proof First assume that K is positive definite. In this case, it can be diagonalized as $K = SDS^\top$, with an orthogonal matrix S and a diagonal matrix D with nonnegative entries. Then

$$k(x_i, x_j) = (SDS^\top)_{ij} = \langle S_i, DS_j \rangle = \left\langle \sqrt{D} S_i, \sqrt{D} S_j \right\rangle, \tag{2.62}$$

where we have defined the S_i as the rows of S (note that the columns of S would be K's eigenvectors). Therefore, K is the Gram matrix of the vectors $\sqrt{D_{ii}} \cdot S_i$.[9] Hence the following map Φ, defined on $x_1, \ldots, x_m$ will satisfy (2.61)

$$\Phi : x_i \mapsto \sqrt{D_{ii}} \cdot S_i. \tag{2.63}$$

Thus far, Φ is only defined on a set of points, rather than on a vector space. Therefore, it makes no sense to ask whether it is linear. We can, however, ask whether it can be *extended* to a linear map, provided the x_i are elements of a vector space. The answer is that if the x_i are linearly dependent (which is often the case), then this will not be possible, since a linear map would then typically be over-

puted as $D_n^{-1/2} U_n^\top (k(z_1, x), \ldots, k(z_n, x))$, where $U_n D_n U_n^\top$ is the eigenvalue decomposition of K_n. Note that the columns of U_n are the eigenvectors of K_n. We discard all columns that correspond to zero eigenvalues, as well as the corresponding dimensions of D_n. To *approximate* the map, we may actually discard all eigenvalues smaller than some $\epsilon > 0$.

9. In fact, every positive definite matrix is the Gram matrix of some set of vectors [46].

determined by the m conditions (2.63).

For the converse, assume an arbitrary $\boldsymbol{\alpha} \in \mathbb{R}^m$, and compute

$$\sum_{i,j=1}^{m} \alpha_i \alpha_j K_{ij} = \left\langle \sum_{i=1}^{m} \alpha_i \Phi(x_i), \sum_{j=1}^{m} \alpha_j \Phi(x_j) \right\rangle = \left\| \sum_{i=1}^{m} \alpha_i \Phi(x_i) \right\|^2 \geq 0. \tag{2.64}$$

■

In particular, this result implies that given data $x_1, \ldots, x_m$, and a kernel k which gives rise to a positive definite matrix K, it is always possible to construct a feature space $\mathcal{H}$ of dimension at most m that we are implicitly working in when using kernels (cf. Problem 2.32 and Section 2.2.6).

If we perform an algorithm which requires k to correspond to a dot product in some other space (as for instance the SV algorithms described in this book), it is possible that even though k is not positive definite in general, it still gives rise to a positive definite Gram matrix K with respect to the training data at hand. In this case, Proposition 2.16 tells us that nothing will go wrong during training when we work with these data. Moreover, if k leads to a matrix with some small negative eigenvalues, we can add a small multiple of some strictly positive definite kernel k' (such as the identity $k'(x_i, x_j) = \delta_{ij}$) to obtain a positive definite matrix. To see this, suppose that $\lambda_{\min} < 0$ is the minimal eigenvalue of k's Gram matrix. Note that being strictly positive definite, the Gram matrix K' of k' satisfies

$$\min_{\|\boldsymbol{\alpha}\|=1} \langle \boldsymbol{\alpha}, K'\boldsymbol{\alpha} \rangle \geq \lambda'_{\min} > 0, \tag{2.65}$$

where $\lambda'_{\min}$ denotes its minimal eigenvalue, and the first inequality follows from Rayleigh's principle (B.57). Therefore, provided that $\lambda_{\min} + \lambda\lambda'_{\min} \geq 0$, we have

$$\langle \boldsymbol{\alpha}, (K + \lambda K')\boldsymbol{\alpha} \rangle = \langle \boldsymbol{\alpha}, K\boldsymbol{\alpha} \rangle + \lambda \langle \boldsymbol{\alpha}, K'\boldsymbol{\alpha} \rangle \geq \|\boldsymbol{\alpha}\|^2 \left(\lambda_{\min} + \lambda\lambda'_{\min}\right) \geq 0 \tag{2.66}$$

for all $\boldsymbol{\alpha} \in \mathbb{R}^m$, rendering $(K + \lambda K')$ positive definite.

2.3 Examples and Properties of Kernels

Polynomial

For the following examples, let us assume that $\mathcal{X} \subset \mathbb{R}^N$. Besides homogeneous polynomial kernels (cf. Proposition 2.1),

$$k(x, x') = \langle x, x' \rangle^d, \tag{2.67}$$

Gaussian

Boser, Guyon, and Vapnik [62, 223, 561] suggest the usage of Gaussian radial basis function kernels [26, 4],

$$k(x, x') = \exp\left(-\frac{\|x - x'\|^2}{2\,\sigma^2}\right), \tag{2.68}$$

Sigmoid

where $\sigma > 0$, and sigmoid kernels,

$$k(x, x') = \tanh(\kappa \langle x, x' \rangle + \vartheta), \tag{2.69}$$