Introduction to Kernel Methods

Bernhard Schölkopf

Max-Planck-Institut für biologische Kybernetik
72076 Tübingen, Germany

bernhard.schoelkopf@tuebingen.mpg.de

Roadmap

1. Kernels
2. Support Vector classification
3. Further kernel algorithms: kernel PCA, kernel dependency estimation, implicit surface approximation, morphing

Learning and Similarity: some Informal Thoughts

- input/output sets $\mathcal{X}, \mathcal{Y}$
- training set $(x_1, y_1), \ldots, (x_m, y_m) \in \mathcal{X} \times \mathcal{Y}$
- "generalization": given a previously unseen $x \in \mathcal{X}$, find a suitable $y \in \mathcal{Y}$
- (x, y) should be "similar" to $(x_1, y_1), \ldots, (x_m, y_m)$
- how to measure similarity?
  - for outputs: *loss function* (e.g., for $\mathcal{Y} = \{\pm 1\}$, zero-one loss)
  - for inputs: *kernel*

Similarity of Inputs

- symmetric function

$$\begin{aligned} k : \mathcal{X} \times \mathcal{X} &\to \mathbb{R} \\ (x, x') &\mapsto k(x, x') \end{aligned}$$

- for example, if $\mathcal{X} = \mathbb{R}^N$: canonical dot product

$$k(x, x') = \sum_{i=1}^{N} [x]_i [x']_i$$

- if $\mathcal{X}$ is not a dot product space: assume that k has a representation as a dot product in a linear space $\mathcal{H}$, i.e., there exists a map $\Phi : \mathcal{X} \to \mathcal{H}$ such that

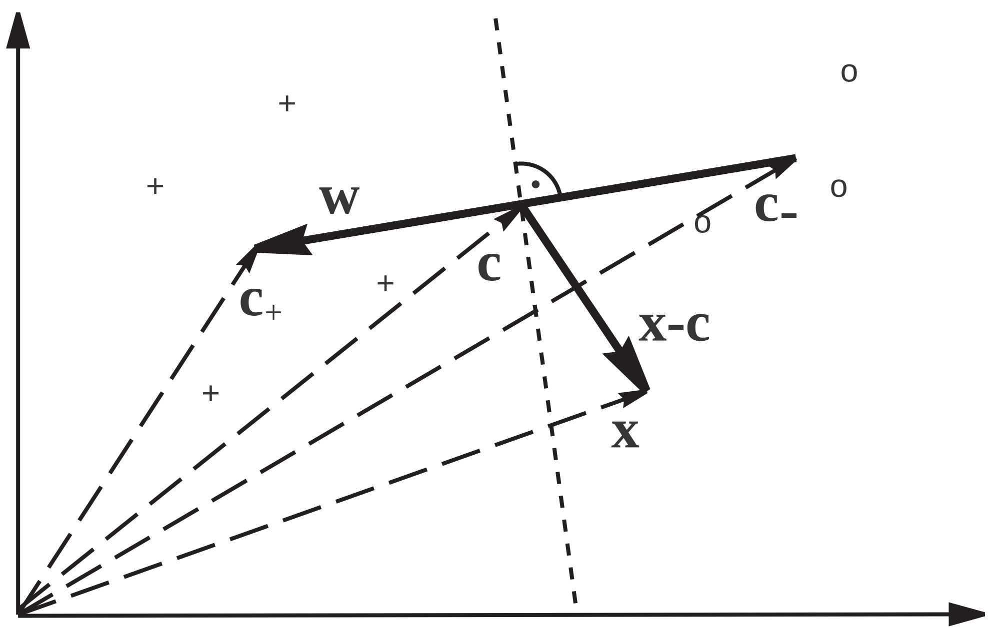
$$k(x, x') = \left\langle \Phi(x), \Phi(x') \right\rangle .$$

- in that case, we can think of the patterns as $\Phi(x), \Phi(x')$, and carry out geometric algorithms in the dot product space ("feature space") $\mathcal{H}$.

An Example of a Kernel Algorithm

Idea: classify points $\mathbf{x} := \Phi(x)$ in feature space according to which of the two class means is closer.

$$\mathbf{c}_+ := \frac{1}{m_+} \sum_{y_i=1} \Phi(x_i), \quad \mathbf{c}_- := \frac{1}{m_-} \sum_{y_i=-1} \Phi(x_i)$$



Compute the sign of the dot product between $\mathbf{w} := \mathbf{c}_+ - \mathbf{c}_-$ and $\mathbf{x} - \mathbf{c}$.

An Example of a Kernel Algorithm, ctd. [44]

$$
\begin{aligned}
f(x) &= \operatorname{sgn}\left(\frac{1}{m_+}\sum_{\{i:y_i=+1\}}\langle\Phi(x),\Phi(x_i)\rangle - \frac{1}{m_-}\sum_{\{i:y_i=-1\}}\langle\Phi(x),\Phi(x_i)\rangle + b\right) \\
&= \operatorname{sgn}\left(\frac{1}{m_+}\sum_{\{i:y_i=+1\}} k(x,x_i) - \frac{1}{m_-}\sum_{\{i:y_i=-1\}} k(x,x_i) + b\right)
\end{aligned}
$$

where

$$
b = \frac{1}{2}\left(\frac{1}{m_-^2}\sum_{\{(i,j):y_i=y_j=-1\}} k(x_i,x_j) - \frac{1}{m_+^2}\sum_{\{(i,j):y_i=y_j=+1\}} k(x_i,x_j)\right).
$$

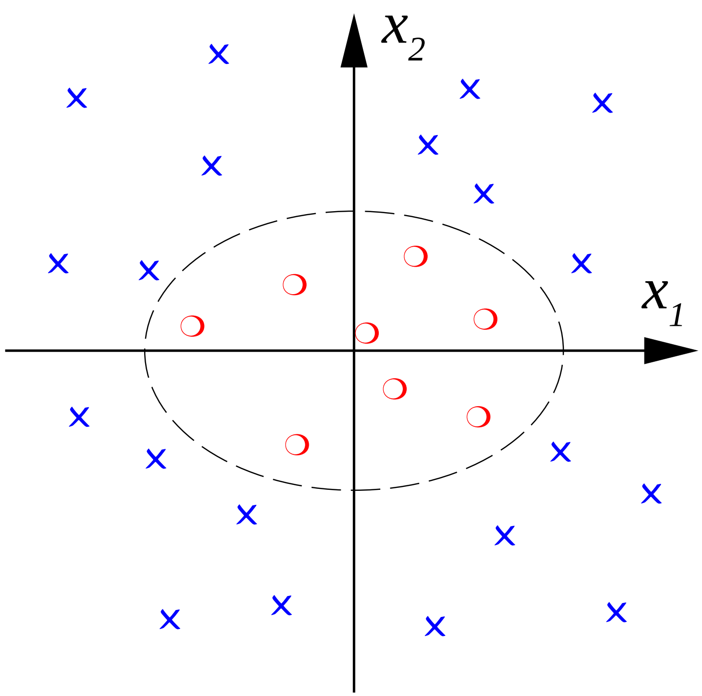
- provides a geometric interpretation of Parzen windows
- the decision function is a hyperplane

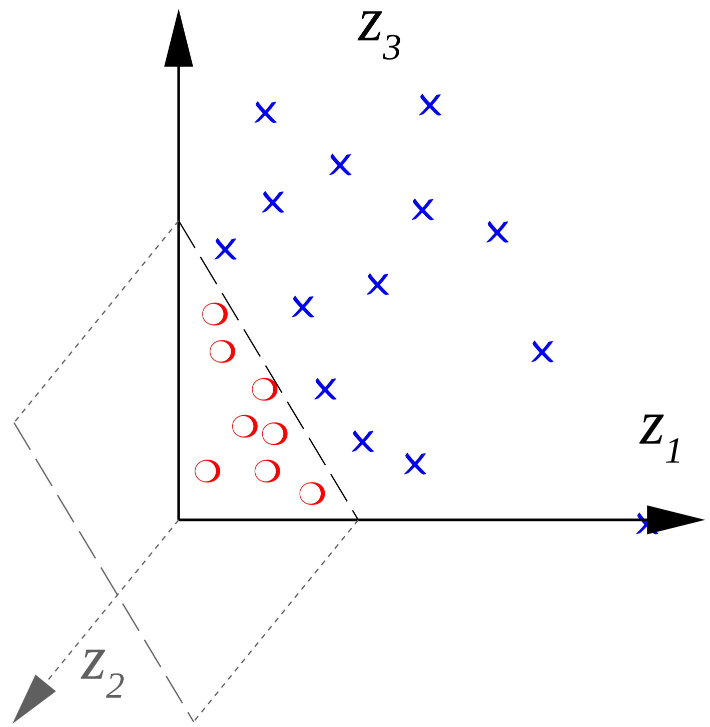
An Example of a Kernel Algorithm, ctd.

- Demo
- Exercise: derive the Parzen windows classifier by computing the distance criterion directly

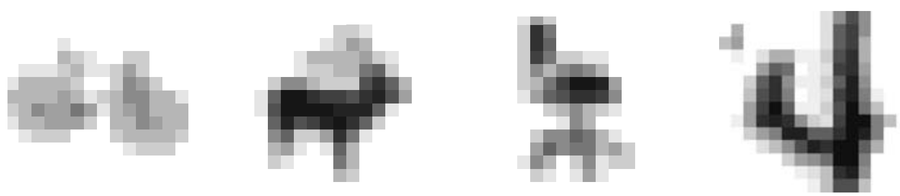
Example: All Degree 2 Monomials

$$\Phi : \mathbb{R}^2 \to \mathbb{R}^3$$
$$(x_1, x_2) \mapsto (z_1, z_2, z_3) := (x_1^2, \sqrt{2}\, x_1 x_2, x_2^2)$$





General Product Feature Space



How about patterns $x \in \mathbb{R}^N$ and product features of order d?

Here, $\dim(\mathcal{H})$ grows like N^d.

E.g. $N = 16 \times 16$, and $d = 5 \longrightarrow$ dimension 10^{10}

The Kernel Trick, $N = d = 2$

$$
\begin{aligned}
\langle \Phi(x), \Phi(x') \rangle &= (x_1^2, \sqrt{2}\, x_1 x_2, x_2^2)(x_1'^2, \sqrt{2}\, x_1' x_2', x_2'^2)^\top \\
&= \langle x, x' \rangle^2 \\
&=: k(x, x')
\end{aligned}
$$

$\longrightarrow$ the dot product in $\mathcal{H}$ can be computed in $\mathbb{R}^2$

The Kernel Trick, II

More generally: $x, x' \in \mathbb{R}^N$, $d \in \mathbb{N}$:

$$\langle x, x' \rangle^d = \left(\sum_{j=1}^{N} x_j \cdot x'_j \right)^d$$
$$= \sum_{j_1,\ldots,j_d=1}^{N} x_{j_1} \cdot \cdots \cdot x_{j_d} \cdot x'_{j_1} \cdot \cdots \cdot x'_{j_d} = \left\langle \Phi(x), \Phi(x') \right\rangle,$$

where Φ maps into the space spanned by all ordered products of d input directions

Mercer's Theorem

If k is a continuous kernel of a positive definite integral operator on $L_2(\mathcal{X})$ (where $\mathcal{X}$ is some compact space),

$$\int_{\mathcal{X}} k(x, x') f(x) f(x') \, dx \, dx' \geq 0,$$

it can be expanded as

$$k(x, x') = \sum_{i=1}^{\infty} \lambda_i \psi_i(x) \psi_i(x')$$

using eigenfunctions ψ_i and eigenvalues $\lambda_i \geq 0$ [36].

The Mercer Feature Map

In that case

$$\Phi(x) := \begin{pmatrix} \sqrt{\lambda_1}\psi_1(x) \\ \sqrt{\lambda_2}\psi_2(x) \\ \vdots \end{pmatrix}$$

satisfies $\langle \Phi(x), \Phi(x') \rangle = k(x, x')$.

Proof:

$$\langle \Phi(x), \Phi(x') \rangle = \left\langle \begin{pmatrix} \sqrt{\lambda_1}\psi_1(x) \\ \sqrt{\lambda_2}\psi_2(x) \\ \vdots \end{pmatrix}, \begin{pmatrix} \sqrt{\lambda_1}\psi_1(x') \\ \sqrt{\lambda_2}\psi_2(x') \\ \vdots \end{pmatrix} \right\rangle$$

$$= \sum_{i=1}^{\infty} \lambda_i \psi_i(x) \psi_i(x') = k(x, x')$$

The Kernel Trick — Summary

- *any* algorithm that only depends on dot products can benefit from the kernel trick
- this way, we can apply linear methods to vectorial as well as *non-vectorial data*
- think of the kernel as a nonlinear *similarity measure*
- examples of common kernels:

$$
\begin{aligned}
\text{Polynomial} \quad k(x,x') &= (\langle x,x' \rangle + c)^d \\
\text{Gaussian} \quad k(x,x') &= \exp(-\|x - x'\|^2/(2\,\sigma^2))
\end{aligned}
$$

- Kernels are studied also in the Gaussian Process prediction community (covariance functions) [61, 58, 63, 35]

Positive Definite Kernels

We will show that the admissible class of kernels coincides with the one of positive definite (pd) kernels: kernels which are symmetric (i.e., $k(x, x') = k(x', x)$), and for

- any set of training points $x_1, \dots, x_m \in \mathcal{X}$ and
- any $a_1, \dots, a_m \in \mathbb{R}$

satisfy

$$\sum_{i,j} a_i a_j K_{ij} \geq 0, \quad \text{where } K_{ij} := k(x_i, x_j).$$

K is called the *Gram matrix* or *kernel matrix*.

Elementary Properties of PD Kernels

Kernels from Feature Maps.
If Φ maps $\mathcal{X}$ into a dot product space $\mathcal{H}$, then $\langle \Phi(x), \Phi(x') \rangle$ is a pd kernel on $\mathcal{X} \times \mathcal{X}$.

Positivity on the Diagonal.
$k(x, x) \geq 0$ for all $x \in \mathcal{X}$

Cauchy-Schwarz Inequality.
$k(x, x')^2 \leq k(x, x)k(x', x')$ (Hint: compute the determinant of the Gram matrix)

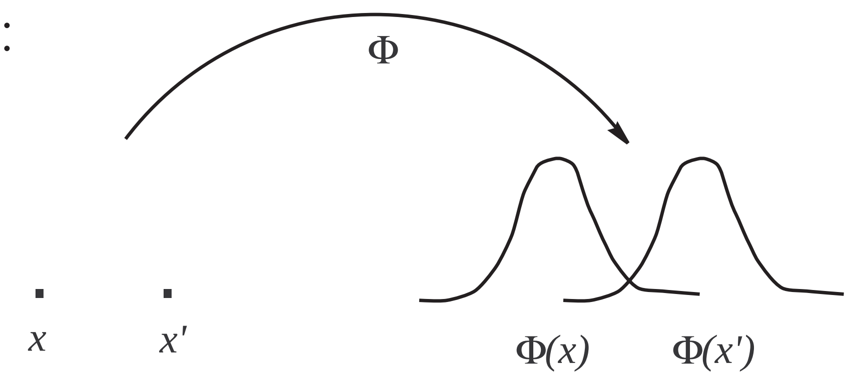
Vanishing Diagonals.
$k(x, x) = 0$ for all $x \in \mathcal{X} \Longrightarrow k(x, x') = 0$ for all $x, x' \in \mathcal{X}$

The Feature Space for PD Kernels [1, 4, 42]

- define a feature map

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

E.g., for the Gaussian kernel:



Next steps:

- turn $\Phi(\mathcal{X})$ into a linear space
- endow it with a dot product satisfying $\langle k(., x_i), k(., x_j) \rangle = k(x_i, x_j)$
- complete the space to get a *reproducing kernel Hilbert space*

Turn it Into a Linear Space

Form linear combinations

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

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

$(m, m' \in \mathbb{N},\ \alpha_i, \beta_j \in \mathbb{R},\ x_i, x'_j \in \mathcal{X})$.

Endow it With a Dot Product

$$
\begin{aligned}
\langle f, g \rangle &:= \sum_{i=1}^{m} \sum_{j=1}^{m'} \alpha_i \beta_j k(x_i, x'_j) \\
&= \sum_{i=1}^{m} \alpha_i g(x_i) = \sum_{j=1}^{m'} \beta_j f(x'_j)
\end{aligned}
$$

- This is well-defined, symmetric, and bilinear (more later).

The Reproducing Kernel Property

Two special cases:

- Assume

$$f(.) = k(., x).$$

  In this case, we have

$$\langle k(., x), g \rangle = g(x).$$

- If moreover

$$g(.) = k(., x'),$$

  we have

$$\langle k(., x), k(., x') \rangle = k(x, x').$$

k is called a *reproducing kernel*

Endow it With a Dot Product, II

- It can be shown that $\langle .,. \rangle$ is a p.d. kernel on the set of functions $\{f(.) = \sum_{i=1}^m \alpha_i k(.,x_i) | \alpha_i \in \mathbb{R}, x_i \in \mathcal{X}\}$:

$$\sum_{ij} \gamma_i \gamma_j \langle f_i, f_j \rangle = \left\langle \sum_i \gamma_i f_i, \sum_j \gamma_j f_j \right\rangle =: \langle f, f \rangle$$

$$= \left\langle \sum_i \alpha_i k(.,x_i), \sum_j \alpha_j k(.,x_j) \right\rangle = \sum_{ij} \alpha_i \alpha_j k(x_i, x_j) \geq 0$$

- furthermore, it is *strictly* positive definite:

  $$f(x)^2 = \langle f, k(.,x) \rangle^2 \leq \langle f, f \rangle \langle k(.,x), k(.,x) \rangle = \langle f, f \rangle k(x,x)$$

  hence $\langle f, f \rangle = 0$ implies $f = 0$.

- Complete the space in the corresponding norm to get a Hilbert space $\mathcal{H}_k$.

Deriving the Kernel from the RKHS

An RKHS is a Hilbert space $\mathcal{H}$ of functions f where all *point evaluation functionals*

$$\begin{aligned} p_x \colon \mathcal{H} &\rightarrow \mathbb{R} \\ f &\mapsto p_x(f) = f(x) \end{aligned}$$

exist and are continuous.

Continuity means that whenever f and f' are close in $\mathcal{H}$, then $f(x)$ and $f'(x)$ are close in $\mathbb{R}$. This can be thought of as a topological prerequisite for generalization ability *(Canu & Mary, 2002)*.

By Riesz' representation theorem, there exists an element of $\mathcal{H}$, call it r_x, such that

$$\langle r_x, f \rangle = f(x),$$

in particular,

$$\langle r_x, r_{x'} \rangle = r_{x'}(x).$$

Define $k(x, x') := r_x(x') = r_{x'}(x)$.

The Empirical Kernel Map

Recall the feature map

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

- each point is represented by its similarity to *all* other points
- how about representing it by its similarity to a *sample* of points?

Consider

$$\begin{aligned} \Phi_m : \mathcal{X} &\to \mathbb{R}^m \\ x &\mapsto k(.,x)|_{(x_1,\ldots,x_m)} = (k(x_1,x),\ldots,k(x_m,x))^\top \end{aligned}$$

ctd.

- $\Phi_m(x_1), \dots, \Phi_m(x_m)$ contain *all* necessary information about $\Phi(x_1), \dots, \Phi(x_m)$
- the Gram matrix $G_{ij} := \left\langle \Phi_m(x_i), \Phi_m(x_j) \right\rangle$ satisfies $G = K^2$ where $K_{ij} = k(x_i, x_j)$
- modify Φ_m to

$$\begin{aligned} \Phi_m^w : \mathcal{X} &\to \mathbb{R}^m \\ x &\mapsto K^{-\frac{1}{2}}(k(x_1, x), \dots, k(x_m, x))^\top \end{aligned}$$

- this whitened map ("kernel PCA map") satifies

$$\left\langle \Phi_m^w(x_i), \Phi_m^w(x_j) \right\rangle = k(x_i, x_j)$$

for all $i, j = 1, \dots, m$.

Some Properties of Kernels [44]

If $k_1, k_2, \ldots$ are pd kernels, then so are

- αk_1, provided $\alpha \geq 0$
- $k_1 + k_2$
- $k_1 \cdot k_2$
- $k(x, x') := \lim_{n\to\infty} k_n(x, x')$, provided it exists
- $k(A, B) := \sum_{x \in A, x' \in B} k_1(x, x')$, where A, B are finite subsets of $\mathcal{X}$
  (using the feature map $\tilde{\Phi}(A) := \sum_{x \in A} \Phi(x)$)

Further operations to construct kernels from kernels: tensor products, direct sums, convolutions [24].

Properties of Kernel Matrices, I [43]

Suppose we are given distinct training patterns $x_1, \dots, x_m$ (which need not live in a vector space), and a positive definite $m \times m$ matrix K.

K can be diagonalized as $K = SDS^\top$, with an orthogonal matrix S and a diagonal matrix D with nonnegative entries. Then

$$K_{ij} = (SDS^\top)_{ij} = \langle S_i, DS_j \rangle = \left\langle \sqrt{D}S_i, \sqrt{D}S_j \right\rangle,$$

where the S_i are the rows of S.

We have thus constructed a map Φ into an m-dimensional feature space $\mathcal{H}$ such that

$$K_{ij} = \langle \Phi(x_i), \Phi(x_j) \rangle.$$

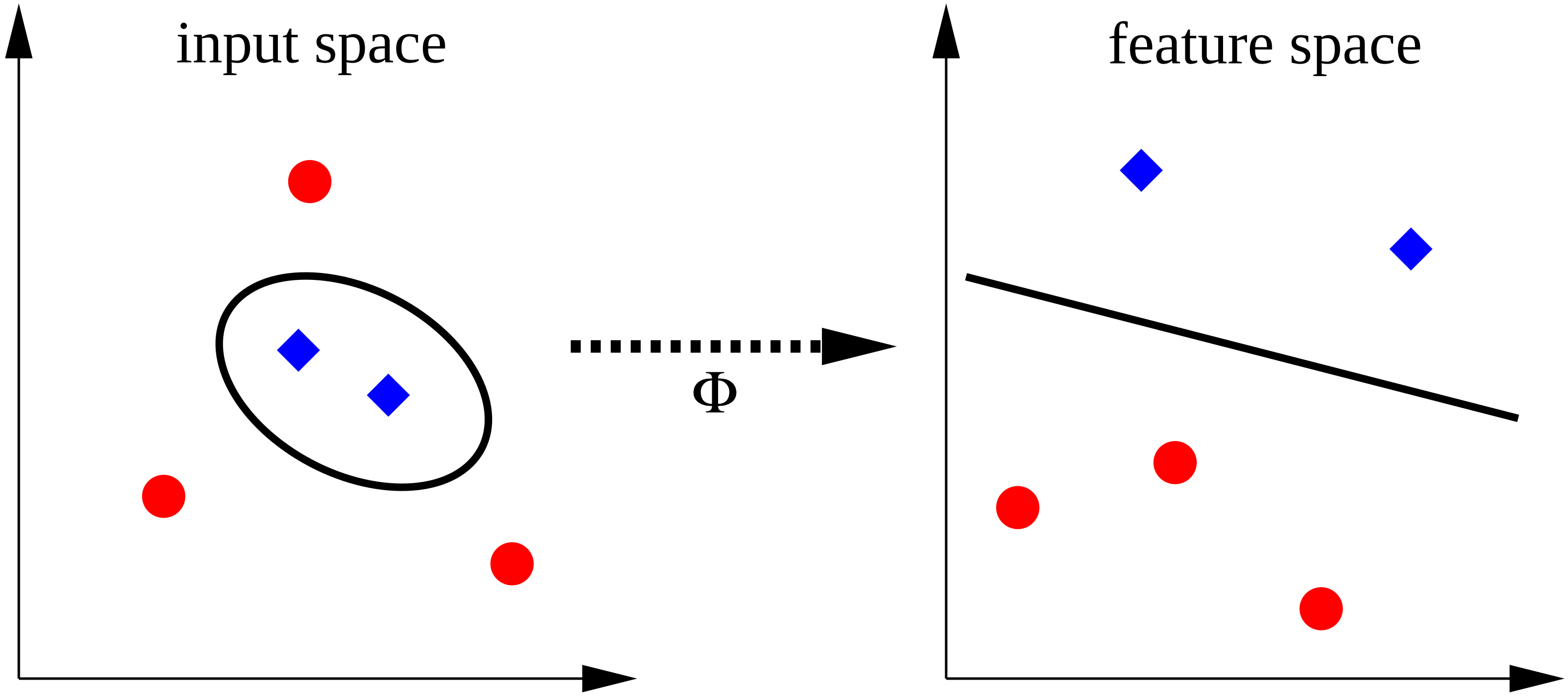
Properties, II: Functional Calculus [47]

- K symmetric $m \times m$ matrix with spectrum $\sigma(K)$
- f a continuous function on $\sigma(K)$
- Then there is a symmetric matrix $f(K)$ with eigenvalues in $f(\sigma(K))$.
- compute $f(K)$ via Taylor series, or eigenvalue decomposition of K: If $K = S^\top DS$ (D diagonal and S unitary), then $f(K) = S^\top f(D)S$, where $f(D)$ is defined elementwise on the diagonal
- can treat functions of symmetric matrices like functions on $\mathbb{R}$

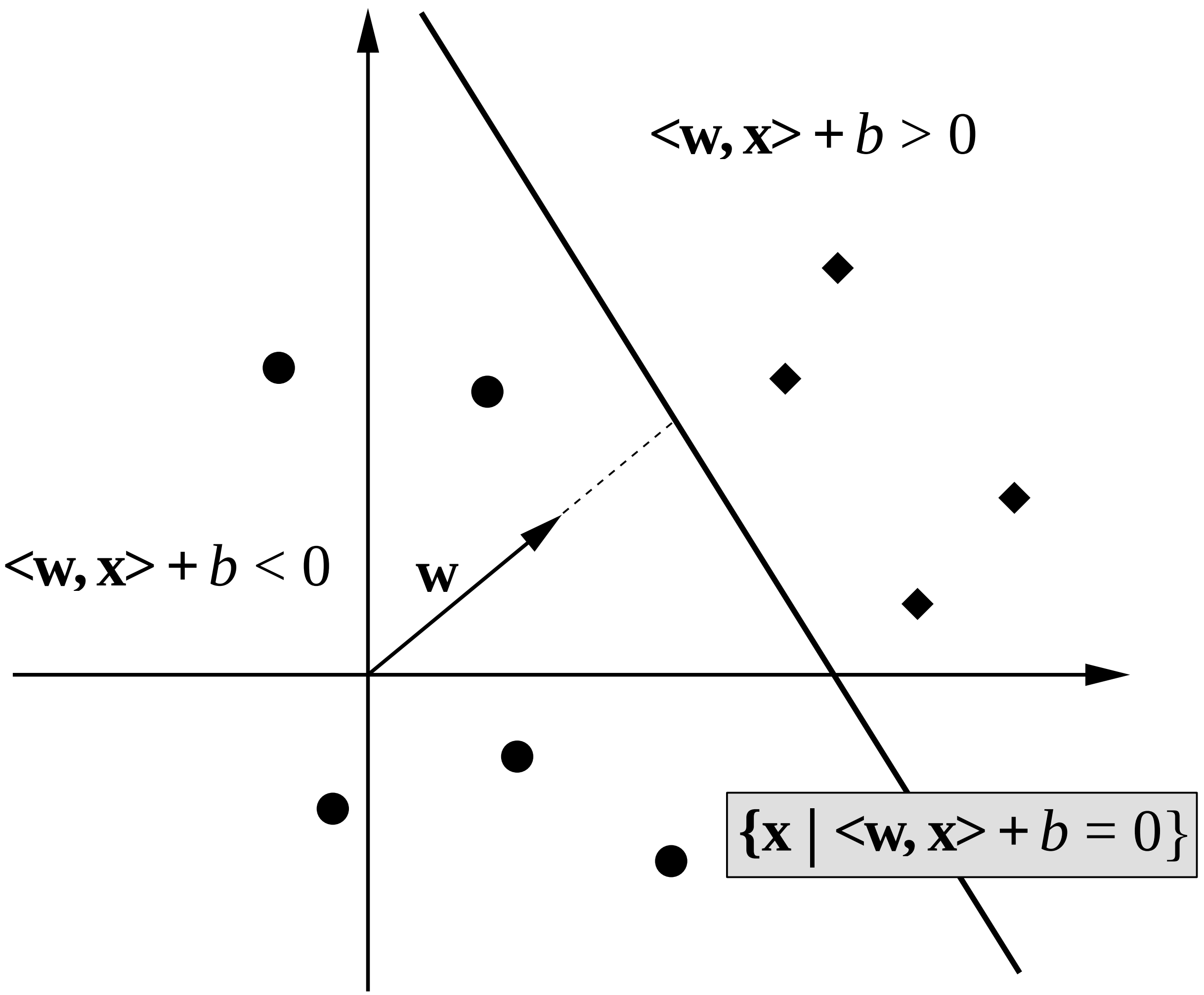
$$
\begin{aligned}
(\alpha f + g)(K) &= \alpha f(K) + g(K) \\
(fg)(K) &= f(K)g(K) = g(K)f(K) \\
\|f\|_{\infty,\sigma(K)} &= \|f(K)\| \\
\sigma(f(K)) &= f(\sigma(K))
\end{aligned}
$$

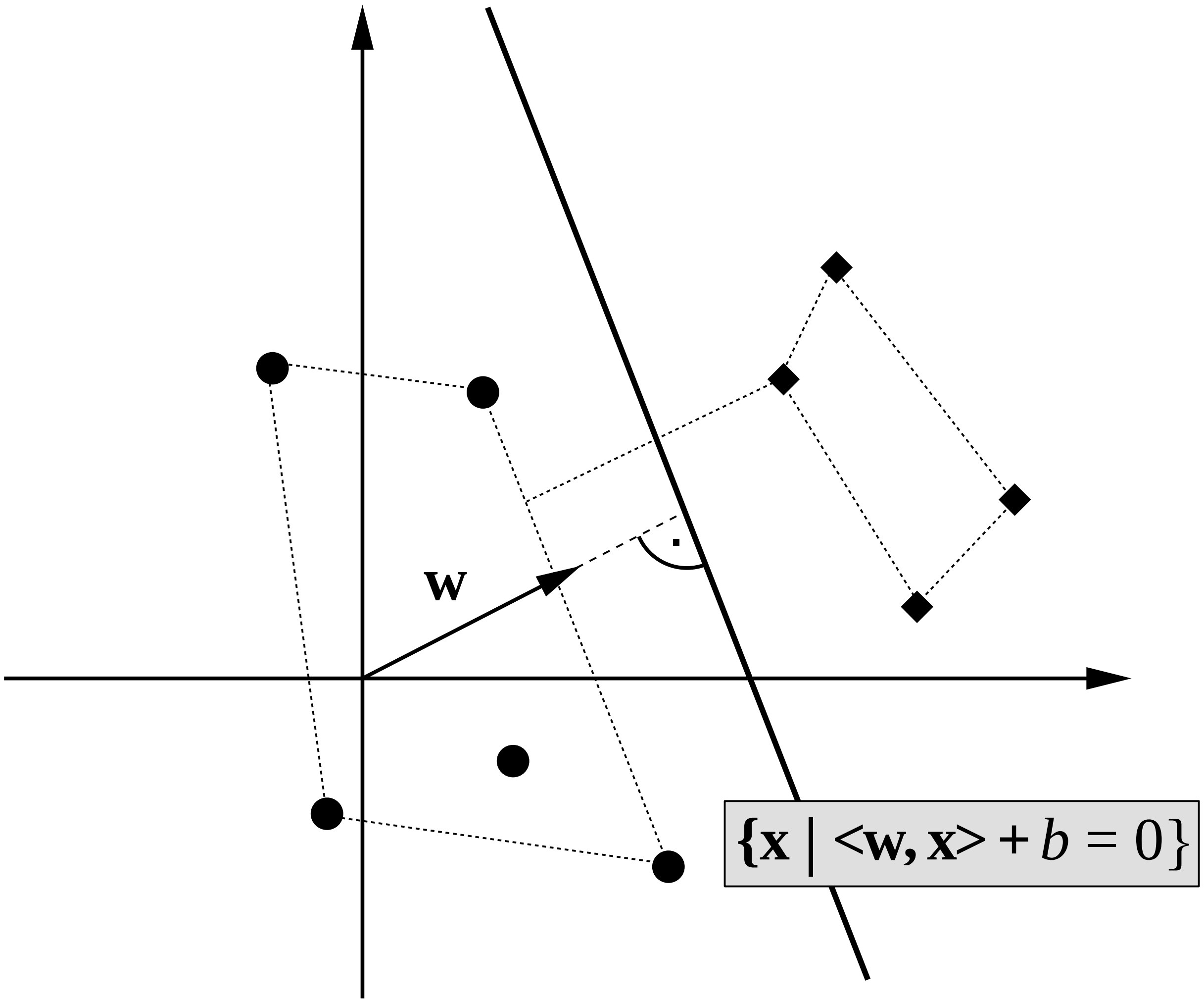
(the C^*-algebra generated by K is isomorphic to the set of continuous functions on $\sigma(K)$)

Support Vector Classifiers



Separating Hyperplane





w
{x | <w, x> + b = 0}

Eliminating the Scaling Freedom

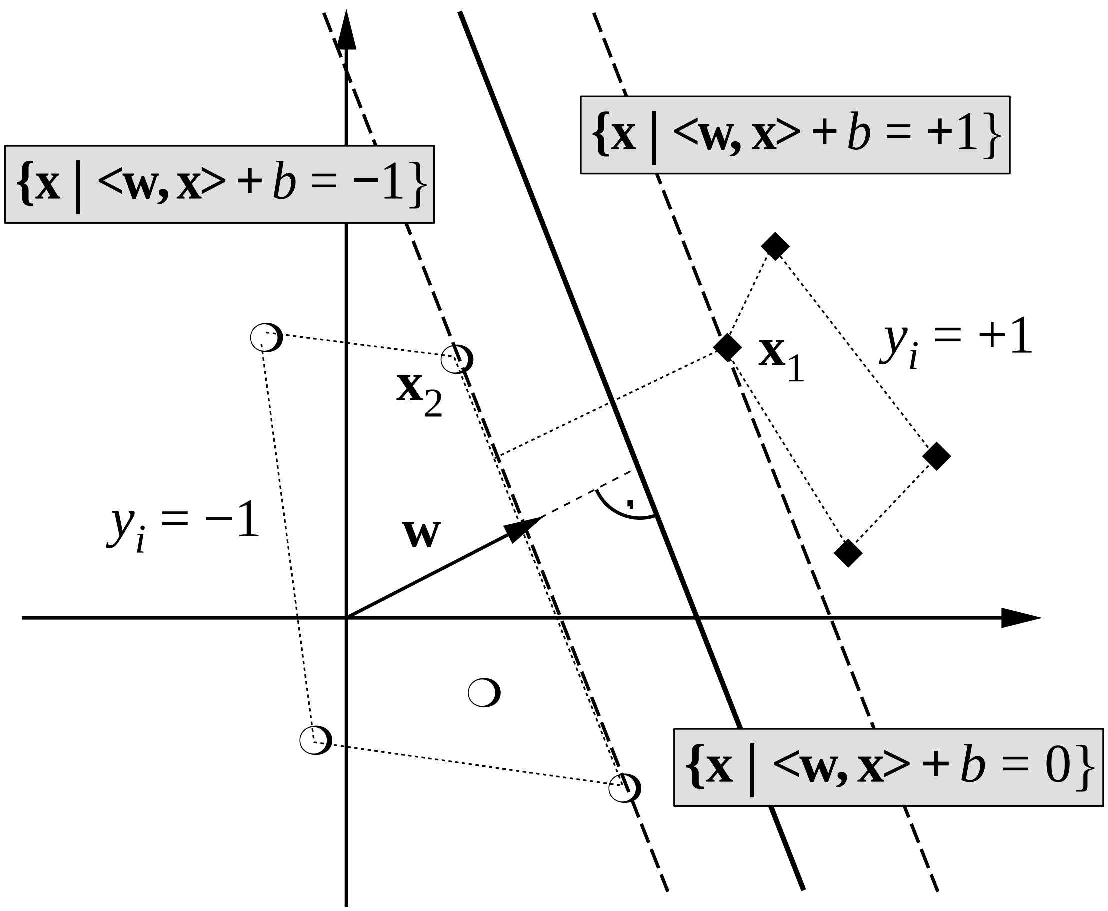
Note: if $c \neq 0$, then

$$\{\mathbf{x} | \langle \mathbf{w}, \mathbf{x} \rangle + b = 0\} = \{\mathbf{x} | \langle c\mathbf{w}, \mathbf{x} \rangle + cb = 0\}.$$

Hence $(c\mathbf{w}, cb)$ describes the same hyperplane as $(\mathbf{w}, b)$.

Definition: The hyperplane is in *canonical* form w.r.t. $X^* = \{\mathbf{x}_1, \ldots, \mathbf{x}_r\}$ if $\min_{\mathbf{x}_i \in X} |\langle \mathbf{w}, \mathbf{x}_i \rangle + b| = 1$.

Canonical Optimal Hyperplane



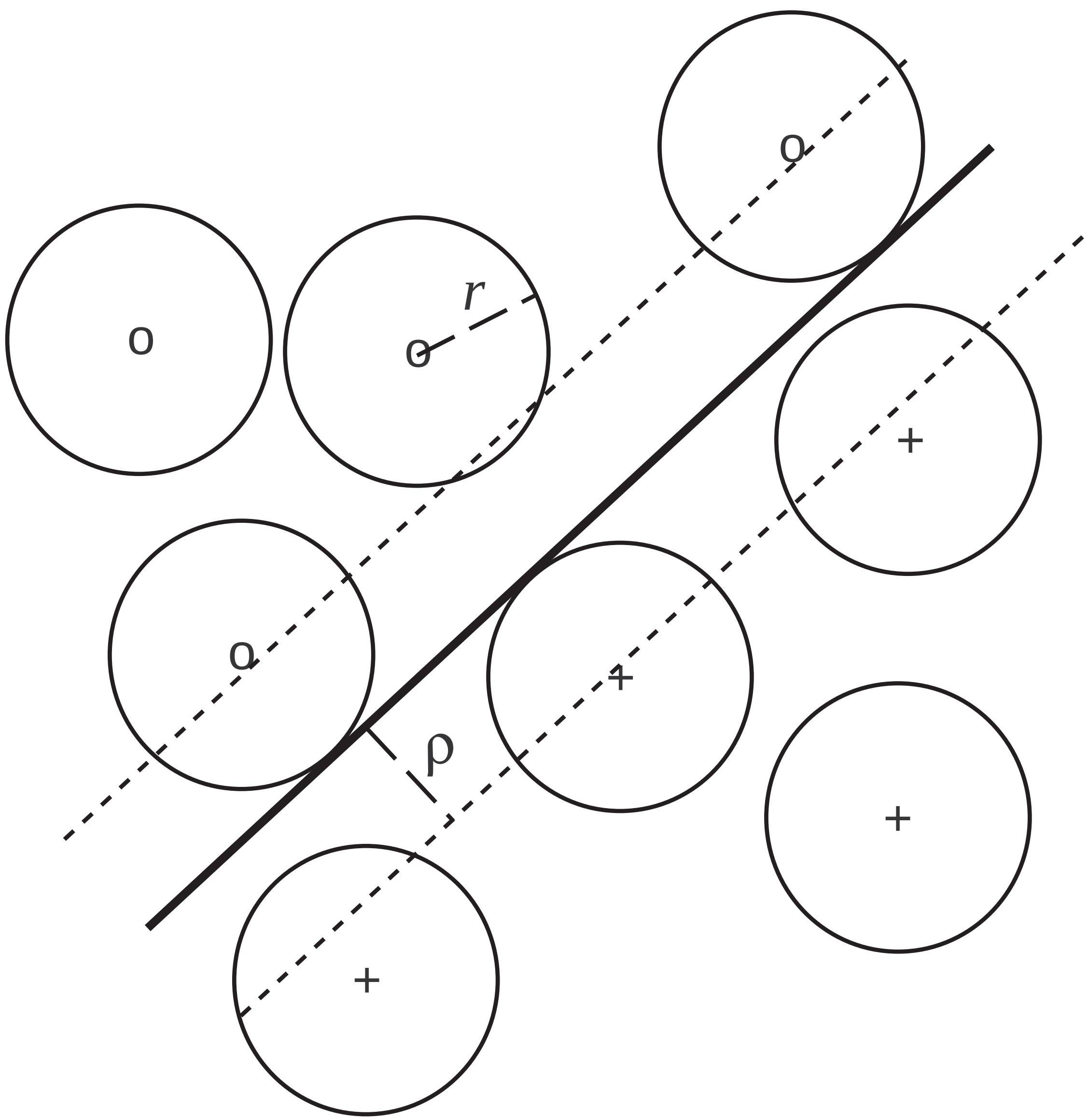
Note:

$$\langle \mathbf{w}, \mathbf{x}_1 \rangle + b = +1$$
$$\langle \mathbf{w}, \mathbf{x}_2 \rangle + b = -1$$

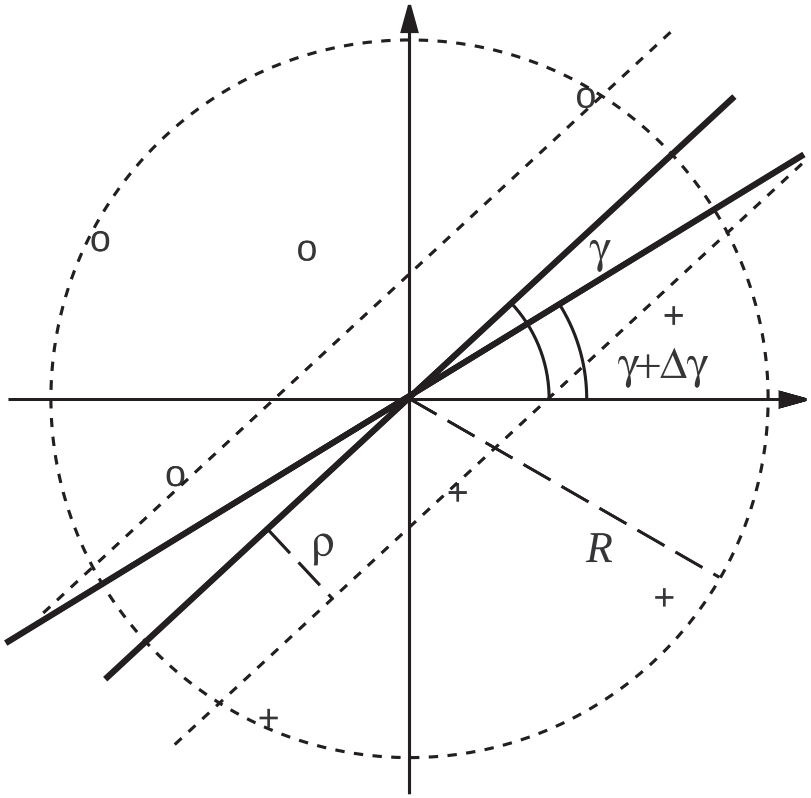
$$=> \quad \langle \mathbf{w}, (\mathbf{x}_1 - \mathbf{x}_2) \rangle = 2$$

$$=> \quad \left\langle \frac{\mathbf{w}}{\|\mathbf{w}\|}, (\mathbf{x}_1 - \mathbf{x}_2) \right\rangle = \frac{2}{\|\mathbf{w}\|}$$

Pattern Noise as Maximum Margin Regularization



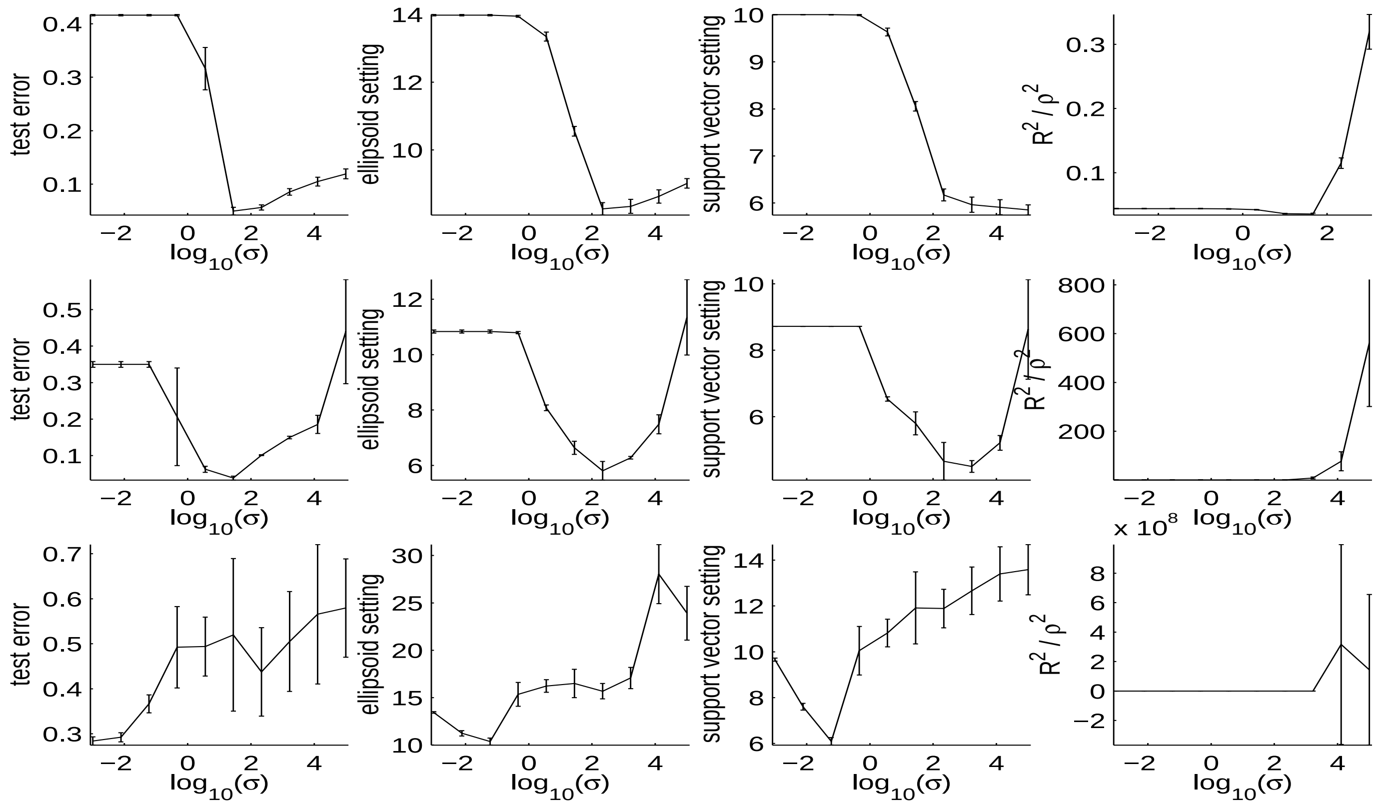
Maximum Margin vs. MDL — 2D Case



Can perturb γ by $\Delta\gamma$ with $|\Delta\gamma| < \arcsin\frac{\rho}{R}$ and still correctly separate the data.

Hence only need to store γ with accuracy $\Delta\gamma$ [44, 57].

Experiments



Datasets:

USPS ($m = 500$)

Wisconsin breast cancer ($m = 200$)

Abalone($m = 500$)

Formulation as an Optimization Problem

Hyperplane with maximum margin: minimize

$$\|\mathbf{w}\|^2$$

(recall: margin $\sim 1/\|\mathbf{w}\|$) subject to

$$y_i \cdot [\langle \mathbf{w}, \mathbf{x}_i \rangle + b] \geq 1 \quad \text{for } i = 1 \dots m$$

(i.e. the training data are separated correctly).

Lagrange Function (e.g., [5])

Introduce Lagrange multipliers $\alpha_i \geq 0$ and a Lagrangian

$$L(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{m} \alpha_i \left(y_i \cdot [\langle \mathbf{w}, \mathbf{x}_i \rangle + b] - 1\right).$$

L has to minimized w.r.t. the *primal variables* $\mathbf{w}$ and b and maximized with respect to the *dual variables* α_i

- if a constraint is violated, then $y_i \cdot (\langle \mathbf{w}, \mathbf{x}_i \rangle + b) - 1 < 0 \longrightarrow$
  - $\cdot$ α_i will grow to increase L — how far?
  - $\cdot$ $\mathbf{w}$, b want to decrease L; i.e. they have to change such that the constraint is satisfied. If the problem is separable, this ensures that $\alpha_i < \infty$.
- similarly: if $y_i \cdot (\langle \mathbf{w}, \mathbf{x}_i \rangle + b) - 1 > 0$, then $\alpha_i = 0$: otherwise, L could be increased by decreasing α_i *(KKT conditions)*

Derivation of the Dual Problem

At the extremum, we have

$$\frac{\partial}{\partial b} L(\mathbf{w}, b, \boldsymbol{\alpha}) = 0, \quad \frac{\partial}{\partial \mathbf{w}} L(\mathbf{w}, b, \boldsymbol{\alpha}) = 0,$$

i.e.

$$\sum_{i=1}^{m} \alpha_i y_i = 0$$

and

$$\mathbf{w} = \sum_{i=1}^{m} \alpha_i y_i \mathbf{x}_i.$$

Substitute both into L to get the *dual problem*

The Support Vector Expansion

$$\mathbf{w} = \sum_{i=1}^{m} \alpha_i y_i \mathbf{x}_i$$

where for all $i = 1, \ldots, m$ either

$y_i \cdot [\langle \mathbf{w}, \mathbf{x}_i \rangle + b] > 1 \quad \Longrightarrow \alpha_i = 0 \longrightarrow \mathbf{x}_i$ irrelevant

or

$y_i \cdot [\langle \mathbf{w}, \mathbf{x}_i \rangle + b] = 1$ (*on* the margin) $\longrightarrow \mathbf{x}_i$ "Support Vector"

The solution is determined by the examples on the margin.

Thus

$$\begin{aligned} f(\mathbf{x}) &= \operatorname{sgn}\left(\langle \mathbf{x}, \mathbf{w} \rangle + b\right) \\ &= \operatorname{sgn}\left(\sum_{i=1}^{m} \alpha_i y_i \langle \mathbf{x}, \mathbf{x}_i \rangle + b\right). \end{aligned}$$

A Mechanical Interpretation

Assume that each SV $\mathbf{x}_i$ exerts a perpendicular force of size α_i and sign y_i on a solid plane sheet lying along the hyperplane.

Then the solution is mechanically stable:

$$\sum_{i=1}^{m} \alpha_i y_i = 0 \quad \text{implies that the forces sum to zero}$$

$$\mathbf{w} = \sum_{i=1}^{m} \alpha_i y_i \mathbf{x}_i \quad \text{implies that the torques sum to zero,}$$

via

$$\sum_i \mathbf{x}_i \times y_i \alpha_i \cdot \mathbf{w}/\|\mathbf{w}\| = \mathbf{w} \times \mathbf{w}/\|\mathbf{w}\| = 0.$$

Dual Problem

Dual: maximize

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

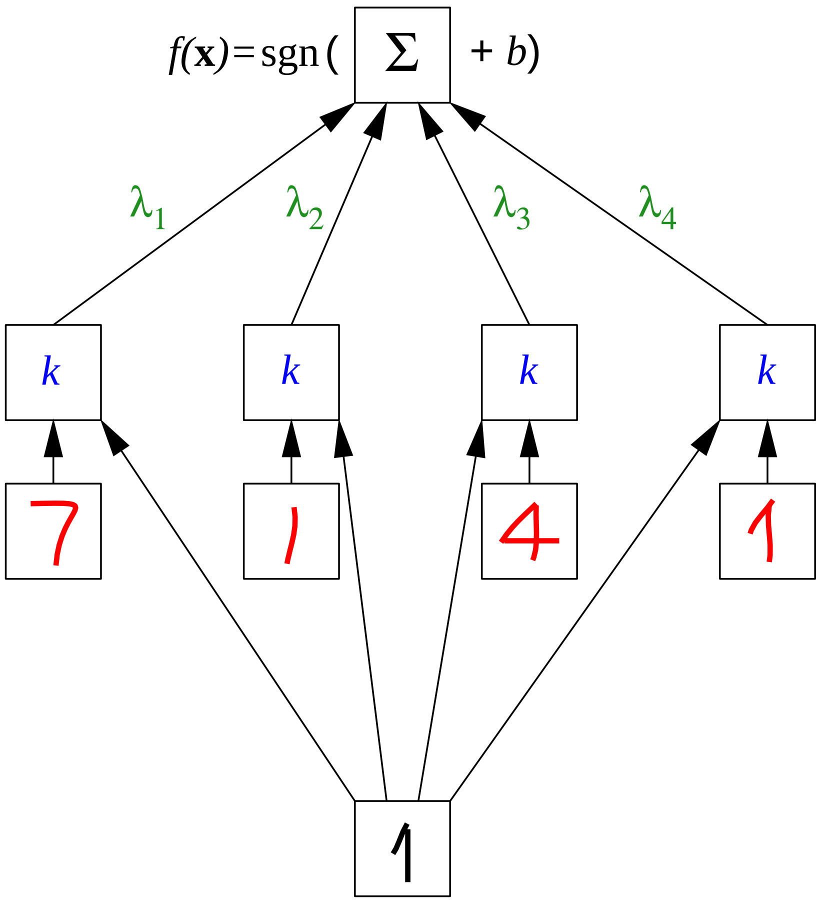
subject to

$$\alpha_i \geq 0, \quad i = 1, \ldots, m, \quad \text{and} \quad \sum_{i=1}^{m} \alpha_i y_i = 0.$$

Both the final decision function and the function to be maximized are expressed in dot products $\longrightarrow$ can use a kernel to compute

$$\langle \mathbf{x}_i, \mathbf{x}_j \rangle = \langle \Phi(x_i), \Phi(x_j) \rangle = k(x_i, x_j).$$

The SVM Architecture



classification

$f(\mathbf{x}) = \mathrm{sgn}\left(\sum \lambda_i \cdot k(\mathbf{x}, \mathbf{x}_i) + b\right)$

weights

comparison: $k(\mathbf{x}, \mathbf{x}_i)$, e.g.

$k(\mathbf{x}, \mathbf{x}_i) = (\mathbf{x} \cdot \mathbf{x}_i)^d$

$k(\mathbf{x}, \mathbf{x}_i) = \exp(-\|\mathbf{x} - \mathbf{x}_i\|^2 / c)$

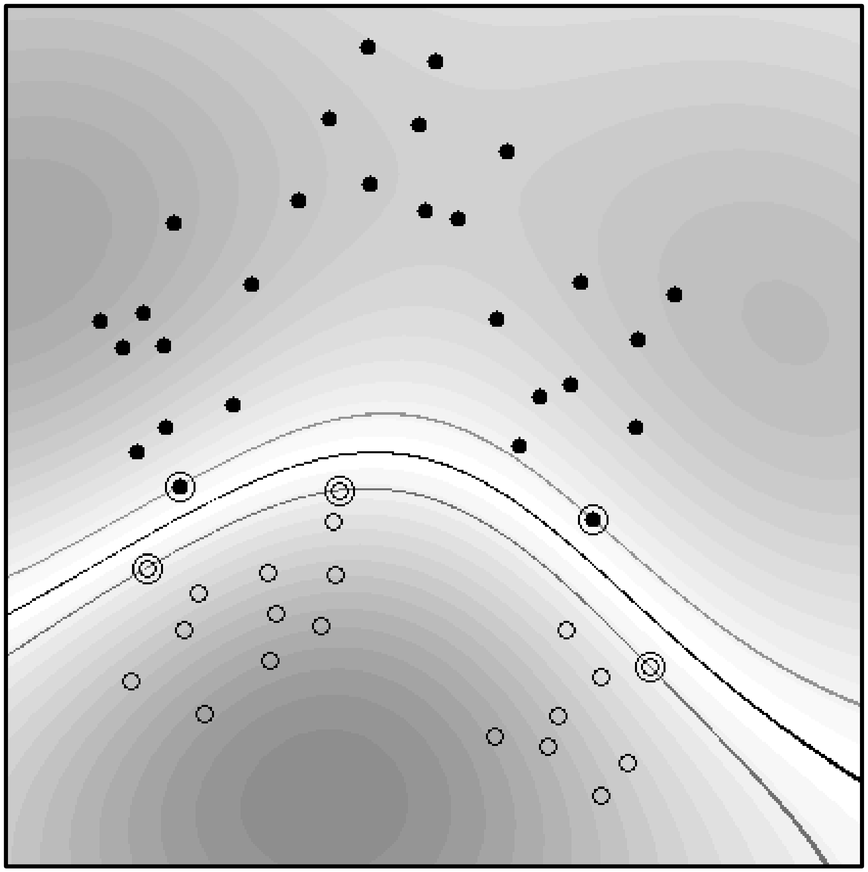
$k(\mathbf{x}, \mathbf{x}_i) = \tanh(\kappa(\mathbf{x} \cdot \mathbf{x}_i) + \theta)$

support vectors
$\mathbf{x}_1 \ldots \mathbf{x}_4$

input vector $\mathbf{x}$

Toy Example with Gaussian Kernel

$$k(x, x') = \exp\left(-\|x - x'\|^2\right)$$



Nonseparable Problems *[3, 14]*

If $y_i \cdot (\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1$ cannot be satisfied, then $\alpha_i \to \infty$.

Modify the constraint to

$$y_i \cdot (\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1 - \xi_i$$

with

$$\xi_i \geq 0$$

("*soft margin*") and add

$$C \cdot \sum_{i=1}^{m} \xi_i$$

in the objective function.

Soft Margin SVMs

C-SVM [14]: for $C > 0$, minimize

$$\tau(\mathbf{w}, \boldsymbol{\xi}) = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{m}\xi_i$$

subject to $y_i \cdot (\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1 - \xi_i, \quad \xi_i \geq 0$ (margin $2/\|\mathbf{w}\|$)

ν-SVM [46]: for $0 \leq \nu < 1$, minimize

$$\tau(\mathbf{w}, \boldsymbol{\xi}, \rho) = \frac{1}{2}\|\mathbf{w}\|^2 - \nu\rho + \frac{1}{m}\sum_{i}\xi_i$$

subject to $y_i \cdot (\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq \rho - \xi_i, \quad \xi_i \geq 0$ (margin $2\rho/\|\mathbf{w}\|$)

The ν-Property

SVs: $\alpha_i > 0$
"margin errors:" $\xi_i > 0$

KKT-Conditions $\Longrightarrow$

- All margin errors are SVs.
- Not all SVs need to be margin errors.
  Those which are *not* lie exactly on the edge of the margin.

Proposition:
1. *fraction of Margin Errors* $\leq \nu \leq$ *fraction of SVs.*
2. *asymptotically:* $\ldots = \nu = \ldots$

Duals, Using Kernels

C-SVM dual: maximize

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

subject to $0 \le \alpha_i \le C, \quad \sum_i \alpha_i y_i = 0.$

ν-SVM dual: maximize

$$W(\boldsymbol{\alpha}) = -\frac{1}{2}\sum_{ij} \alpha_i \alpha_j y_i y_j k(\mathbf{x}_i, \mathbf{x}_j)$$

subject to $0 \le \alpha_i \le \frac{1}{m}, \quad \sum_i \alpha_i y_i = 0, \quad \sum_i \alpha_i \ge \nu$

In both cases: *decision function*:

$$f(\mathbf{x}) = \operatorname{sgn}\left(\sum_{i=1}^{m} \alpha_i y_i k(\mathbf{x}, \mathbf{x}_i) + b\right)$$

The Representer Theorem

Theorem 1 *Given: a p.d. kernel k on $\mathcal{X} \times \mathcal{X}$, a training set $(x_1, y_1), \ldots, (x_m, y_m) \in \mathcal{X} \times \mathbb{R}$, a strictly monotonic increasing real-valued function Ω on $[0, \infty[$, and an arbitrary cost function $c : (\mathcal{X} \times \mathbb{R}^2)^m \to \mathbb{R} \cup \{\infty\}$*

Any $f \in \mathcal{H}$ minimizing the regularized risk functional

$$c\left((x_1, y_1, f(x_1)), \ldots, (x_m, y_m, f(x_m))\right) + \Omega\left(\|f\|\right) \tag{1}$$

admits a representation of the form

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

Remarks

- significance: many learning algorithms have solutions that can be expressed as expansions in terms of the training examples
- original form, with mean squared loss

$$c((x_1, y_1, f(x_1)), \ldots, (x_m, y_m, f(x_m))) = \frac{1}{m}\sum_{i=1}^{m}(y_i - f(x_i))^2,$$

  and $\Omega(\|f\|) = \lambda\|f\|^2$ ($\lambda > 0$): [31]
- generalization to non-quadratic cost functions: [15]
- present form: [44]

Proof

Decompose $f \in \mathcal{H}$ into a part in the span of the $k(x_i, .)$ and an orthogonal one:

$$f = \sum_i \alpha_i k(x_i, .) + f_\perp,$$

where for all j

$$\langle f_\perp, k(x_j, .)\rangle = 0.$$

Application of f to an arbitrary training point x_j yields

$$\begin{aligned} f(x_j) &= \langle f, k(x_j, .)\rangle \\ &= \left\langle \sum_i \alpha_i k(x_i, .) + f_\perp, k(x_j, .) \right\rangle \\ &= \sum_i \alpha_i \langle k(x_i, .), k(x_j, .)\rangle, \end{aligned}$$

independent of $f_\perp$.

Proof: second part of (1)

Since $f_\perp$ is orthogonal to $\sum_i \alpha_i k(x_i, .)$, and Ω is strictly monotonic, we get

$$\begin{aligned}
\Omega(\|f\|) &= \Omega\left(\|\sum_i \alpha_i k(x_i, .) + f_\perp\|\right) \\
&= \Omega\left(\sqrt{\|\sum_i \alpha_i k(x_i, .)\|^2 + \|f_\perp\|^2}\right) \\
&\geq \Omega\left(\|\sum_i \alpha_i k(x_i, .)\|\right),
\end{aligned}$$

with equality occuring if and only if $f_\perp = 0$.
Hence, any minimizer must have $f_\perp = 0$. Consequently, any solution takes the form

$$f = \sum_i \alpha_i k(x_i, .).$$

Application: Support Vector Classification

Here, $y_i \in \{\pm 1\}$. Use

$$c\left(\left(x_i, y_i, f\left(x_i\right)\right)_i\right) = \frac{1}{\lambda} \sum_i \max\left(0, 1 - y_i f\left(x_i\right)\right),$$

and the regularizer $\Omega\left(\|f\|\right) = \|f\|^2$.

$\lambda \longrightarrow 0$ leads to the hard margin SVM

Further Applications

Bayesian MAP Estimates. Identify (1) with the negative log posterior (cf. Kimeldorf & Wahba, 1970, Poggio & Girosi, 1990), i.e.

- $\exp(-c((x_i, y_i, f(x_i))_i))$ — likelihood of the data
- $\exp(-\Omega(\|f\|))$ — prior over the set of functions; e.g., $\Omega(\|f\|) = \lambda\|f\|^2$ — Gaussian process prior [63] with covariance function k
- minimizer of (1) = MAP estimate

Kernel PCA (see below) can be shown to correspond to the case of

$$c((x_i, y_i, f(x_i))_{i=1,\ldots,m}) = \begin{cases} 0 & \text{if } \frac{1}{m}\sum_i \left(f(x_i) - \frac{1}{m}\sum_j f(x_j)\right)^2 = 1 \\ \infty & \text{otherwise} \end{cases}$$

with g an arbitrary strictly monotonically increasing function.

SVM Training

- naive approach: the complexity of maximizing

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

  scales with the third power of the training set size m

- only SVs are relevant $\longrightarrow$ only compute $(k(\mathbf{x}_i, \mathbf{x}_j))_{ij}$ for SVs. Extract them iteratively by cycling through the training set in chunks [53].

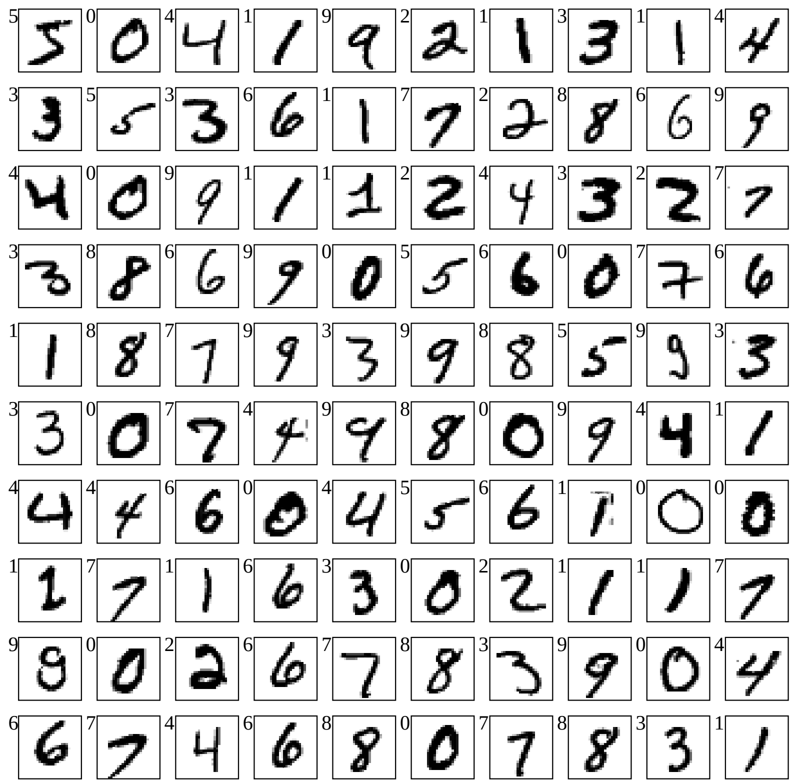
- in fact, one can use chunks which do not even contain all SVs [37]. Maximize over these sub-problems, using your favorite optimizer.

- the extreme case: by making the sub-problems very small (just two points), one can solve them analytically [39].

- http://www.kernel-machines.org/software.html

MNIST Benchmark

handwritten character benchmark (60000 training & 10000 test examples, 28×28)



MNIST Error Rates

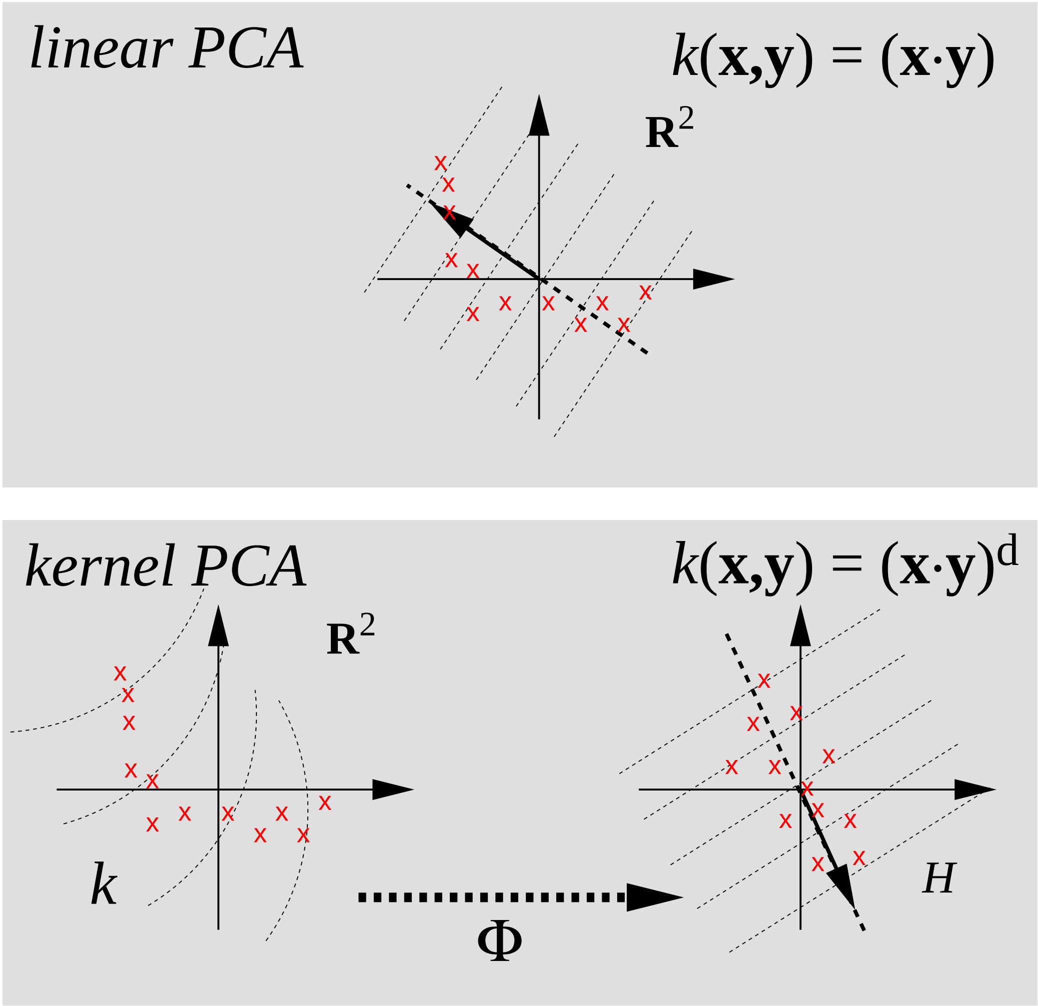
Classifier	test error	reference
linear classifier	8.4%	[7]
3-nearest-neighbour	2.4%	[7]
SVM	1.4%	[10]
Tangent distance	1.1%	[50]
LeNet4	1.1%	[33]
Boosted LeNet4	0.7%	[33]
Translation invariant SVM	0.56%	[18]

Note: the SVM used a polynomial kernel of degree 9, corresponding to a feature space of dimension $\approx 3.2 \cdot 10^{20}$.

Other successful applications: e.g., [29, 27, 25, 11, 51, 8, 65, 21, 20, 13, 19, 38, 59, 64]

Further Kernel Algorithms — Design Principles

1. "Kernel module"
   - similarity measure $k(x, x')$, where $x, x' \in \mathcal{X}$
   - data representation (in associated feature space where $k(x, x') = \langle \Phi(x), \Phi(x') \rangle$) — thus can construct geometric algorithms
   - function class ("representer theorem," $f(x) = \sum_i \alpha_i k(x, x_i)$)
2. "Learning module"
   - classification
   - quantile estimation / novelty detection
   - feature extraction
   - ...



linear PCA
$k(\mathbf{x},\mathbf{y}) = (\mathbf{x}\cdot\mathbf{y})$
$\mathbf{R}^2$
kernel PCA
$k(\mathbf{x},\mathbf{y}) = (\mathbf{x}\cdot\mathbf{y})^d$
$\mathbf{R}^2$
k
Φ
H

Kernel PCA, II

$$x_1, \ldots, x_m \in \mathcal{X}, \quad \Phi : \mathcal{X} \to \mathcal{H}, \quad C = \frac{1}{m} \sum_{j=1}^{m} \Phi(x_j)\Phi(x_j)^\top$$

Eigenvalue problem

$$\lambda \mathbf{V} = C\mathbf{V} = \frac{1}{m} \sum_{j=1}^{m} \left\langle \Phi(x_j), \mathbf{V} \right\rangle \Phi(x_j).$$

For $\lambda \neq 0$, $\mathbf{V} \in \mathrm{span}\{\Phi(x_1), \ldots, \Phi(x_m)\}$, thus

$$\mathbf{V} = \sum_{i=1}^{m} \alpha_i \Phi(x_i),$$

and the eigenvalue problem can be written as

$$\lambda \left\langle \Phi(x_n), \mathbf{V} \right\rangle = \left\langle \Phi(x_n), C\mathbf{V} \right\rangle \text{ for all } n = 1, \ldots, m$$

Kernel PCA in Dual Variables

In term of the $m \times m$ Gram matrix

$$K_{ij} := \left\langle \Phi(x_i), \Phi(x_j) \right\rangle = k(x_i, x_j),$$

this leads to

$$m\lambda K\boldsymbol{\alpha} = K^2\boldsymbol{\alpha}$$

where $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_m)^\top$.

Solve

$$m\lambda\boldsymbol{\alpha} = K\boldsymbol{\alpha}$$

$\longrightarrow (\lambda_n, \boldsymbol{\alpha}^n)$

$$\langle \mathbf{V}^n, \mathbf{V}^n \rangle = 1 \Longleftrightarrow \lambda_n \langle \boldsymbol{\alpha}^n, \boldsymbol{\alpha}^n \rangle = 1$$

thus divide $\boldsymbol{\alpha}^n$ by $\sqrt{\lambda_n}$

Feature extraction

Compute projections on the Eigenvectors

$$\mathbf{V}^n = \sum_{i=1}^{m} \alpha_i^n \Phi(x_i)$$

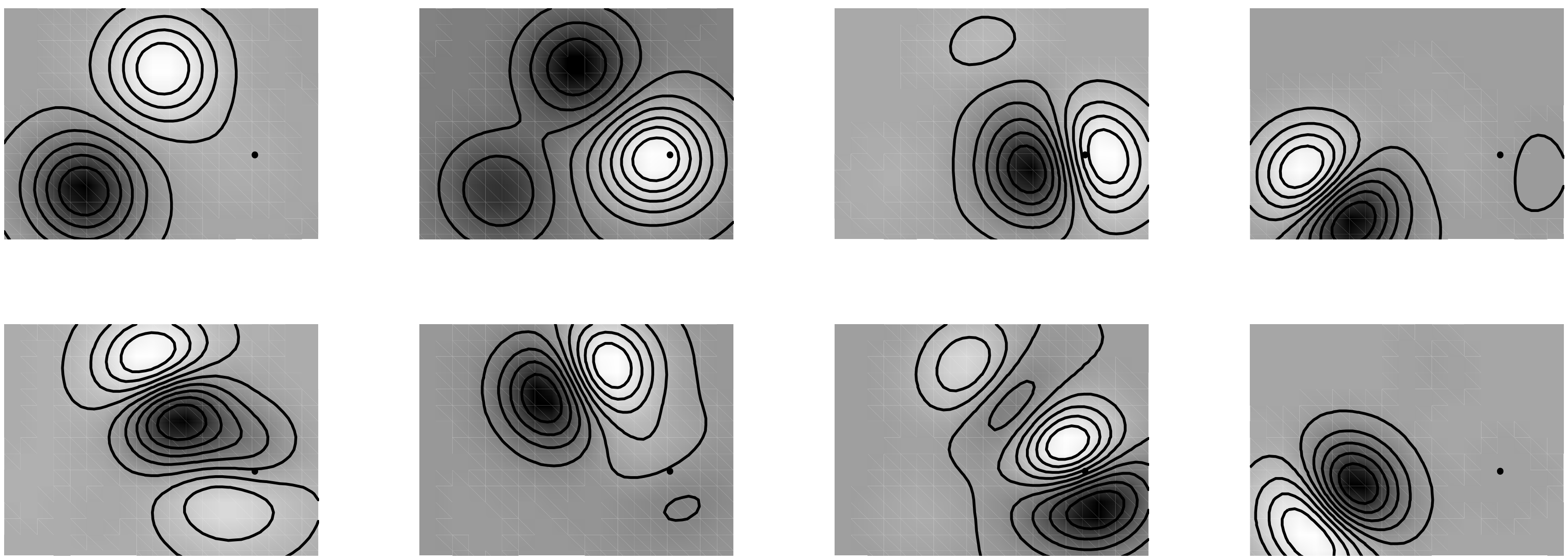
in $\mathcal{H}$:

for a test point x with image $\Phi(x)$ in $\mathcal{H}$ we get the features

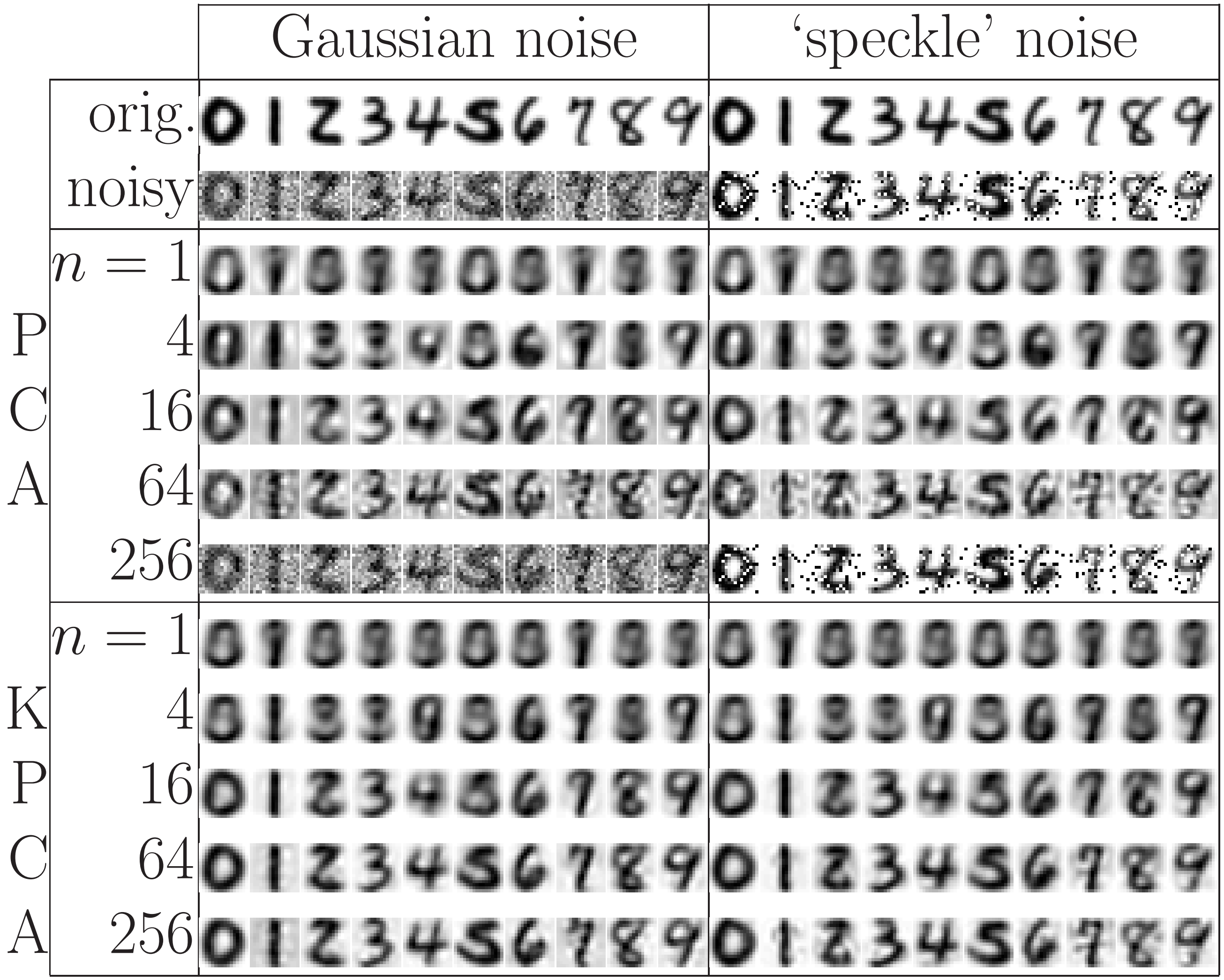
$$\begin{aligned}\langle \mathbf{V}^n, \Phi(x) \rangle &= \sum_{i=1}^{m} \alpha_i^n \langle \Phi(x_i), \Phi(x) \rangle \\ &= \sum_{i=1}^{m} \alpha_i^n k(x_i, x)\end{aligned}$$

Toy Example with Gaussian Kernel

$$k(x, x') = \exp\left(-\|x - x'\|^2\right)$$



Denoising of USPS Digits



linear PCA reconstruction

kernel PCA reconstruction

Another application: face modeling [41].

Natural Image KPCA Model

Training images of size 396×528. The 12×12 training patterns are obtained by sampling 2,500 patches at random from each image.

Super-Resolution *(Kim, Franz, & Schölkopf, 2004)*

a. original image of resolution 528 × 396

b. low resolution image (264 × 198) stretched to the original scale

c. bicubic interpolation

d. supervised example-based learning based on nearest neighbor classifier

f. unsupervised KHA reconstruction

g. enlarged portions of a-d, and f (from left to right)

Comparison between different super-resolution methods.

Kernel Dependency Estimation

Given two sets $\mathcal{X}$ and $\mathcal{Y}$ with kernels k and k', and training data (x_i, y_i).

Estimate a dependency $\mathbf{w} : \mathcal{H} \to \mathcal{H}'$

$$\mathbf{w}(\cdot) = \sum_{ij} \alpha_{ij} \Phi'(y_j) \langle \Phi(x_i), \cdot \rangle .$$

This can be evaluated in various ways, e.g., given an x, we can compute the pre-image

$$y = \mathrm{argmin}_{\mathcal{Y}} \| \mathbf{w}(\Phi(x)) - \Phi'(y) \|.$$

A convenient way of learning the α_{ij} is to work in the kernel PCA basis.

Application to Image Completion

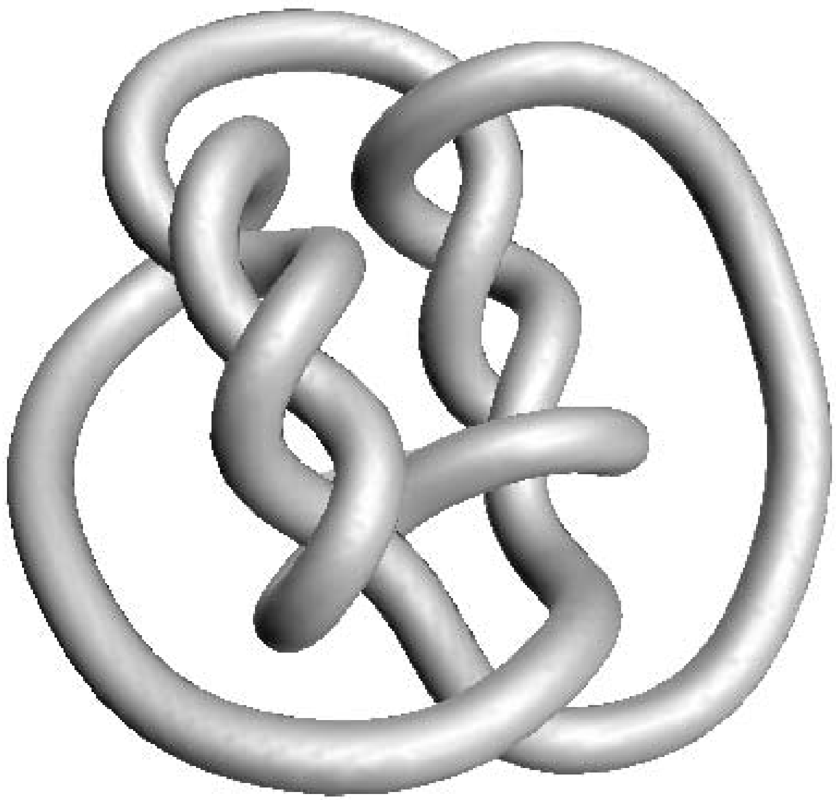
Shown are all digits where at least one of the two algorithms makes a mistake (73 mistakes for k-NN, 23 for KDE).

(from [62])

Implicit Surface Modelling

using a modified one-class SVM *(Schölkopf, Giesen, & Spalinger, 2005)*:

$$\{x \colon f(x) = 0\}$$



Next: powerpoint excursion

Kernel Machines Research

- algorithms/tasks: KDE, feature selection *(Weston et al., 2002)*, multi-label-problems *(Elisseeff & Weston, 2001)*, unlabelled data *(Szummer & Jaakkola, 2002, Zhou et al., 2004)*, ICA *[23]*, canonical correlations *(Bach & Jordan, 2002; Kuss, 2002)*
- optimization and implementation: QP, SDP *(Lanckriet et al., 2002)*, online versions, ...
- theory of empirical inference: sharper capacity measures and bounds *(Bartlett, Bousquet, & Mendelson, 2002)*, generalized evaluation spaces *(Mary & Canu, 2002)*, ...
- kernel design
  - transformation invariances *[12]*
  - kernels for discrete objects *[24, 60, 34, 17, 56]*
  - kernels based on generative models *[28, 48, 52]*
  - local kernels *[e.g., 65]*
  - complex kernels from simple ones *[24, 2]*, global kernels from local ones *[32]*
  - functional calculus for kernel matrices *[47]*
  - model selection, e.g., via alignment *[16]*
  - kernels for dimensionality reduction *[22]*

Conclusion

- crucial ingredients of SV algorithms: kernels that can be represented as dot products, and large margin regularizers
- kernels allow the formulation of a multitude of geometrical algorithms (Parzen windows, SVMs, kernel PCA,...)
- the choice of a kernel corresponds to
  - choosing a similarity measure for the data, or
  - choosing a (linear) representation of the data, or
  - choosing a hypothesis space for learning,

  and should reflect prior knowledge about the problem at hand.

For further information, cf.
`http://www.kernel-machines.org`,
`http://www.learning-with-kernels.org`,
[9, 17, 49, 26, 44].

References

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

[2] P. L. Bartlett and B. Schölkopf. Some kernels for structured data. Technical report, Biowulf Technologies, 2001.

[3] K. P. Bennett and O. L. Mangasarian. Robust linear programming discrimination of two linearly inseparable sets. *Optimization Methods and Software*, 1:23–34, 1992.

[4] C. Berg, J. P. R. Christensen, and P. Ressel. *Harmonic Analysis on Semigroups*. Springer-Verlag, New York, 1984.

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

[6] B. E. Boser, I. M. Guyon, and V. 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.

[7] L. Bottou, C. Cortes, J. S. Denker, H. Drucker, I. Guyon, L. D. Jackel, Y. LeCun, U. A. Müller, E. Säckinger, P. Simard, and V. Vapnik. Comparison of classifier methods: a case study in handwritten digit recognition. In *Proceedings of the 12th International Conference on Pattern Recognition and Neural Networks, Jerusalem*, pages 77–87. IEEE Computer Society Press, 1994.

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

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

[10] C. J. C. Burges and B. Schölkopf. Improving the accuracy and speed of support vector learning machines. In M. Mozer, M. Jordan, and T. Petsche, editors, *Advances in Neural Information Processing Systems 9*, pages 375–381, Cambridge, MA, 1997. MIT Press.

[11] O. Chapelle, P. Haffner, and V. Vapnik. SVMs for histogram-based image classification. *IEEE Transactions on Neural Networks*, 10(5), 1999.

[12] O. Chapelle and B. Schölkopf. Incorporating invariances in nonlinear SVMs. In T.G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, Cambridge, MA, 2002. MIT Press.

[13] S. Chen and C. J. Harris. Design of the optimal separating hyperplane for the decision feedback equalizer using support vector machines. In *IEEE International Conference on Acoustic, Speech, and Signal Processing*, Istanbul, Turkey, 2000.

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

[15] D. Cox and F. O'Sullivan. Asymptotic analysis of penalized likelihood and related estimators. *Annals of Statistics*, 18:1676–1695, 1990.

[16] N. Cristianini, A. Elisseeff, and J. Shawe-Taylor. On optimizing kernel alignment. Technical Report 2001-087, NeuroCOLT, 2001.

[17] N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines and other kernel-based learning methods*. Cambridge University Press, Cambridge, UK, 2000.

[18] D. DeCoste and B. Schölkopf. Training invariant support vector machines. *Machine Learning*, 46:161–190, 2002. Also: Technical Report JPL-MLTR-00-1, Jet Propulsion Laboratory, Pasadena, CA, 2000.

[19] H. Drucker, B. Shahrary, and D. C. Gibbon. Relevance feedback using support vector machines. In *Proceedings of the 18th International Conference on Machine Learning*. Morgan Kaufmann, 2001.

[20] T. S. Furey, N. Duffy, N. Cristianini, D. Bednarski, M. Schummer, and D. Haussler. Support vector machine classification and validation of cancer tissue samples using microarray expression data. *Bioinformatics*, 16(10):906–914, 2000.

[21] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46:389–422, 2002.

[22] J. Ham, D. Lee, S. Mika, and B. Schölkopf. A kernel view of the dimensionality reduction of manifolds. In *Proceedings of ICML*. 2004.

[23] S. Harmeling, A. Ziehe, M. Kawanabe, and K.-R. Müller. Kernel feature spaces and nonlinear blind source separation. In T.G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems*, volume 14. MIT Press, 2002. To appear.

[24] D. Haussler. Convolutional kernels on discrete structures. Technical Report UCSC-CRL-99-10, Computer Science Department, University of California at Santa Cruz, 1999.

[25] M. A. Hearst, B. Schölkopf, S. Dumais, E. Osuna, and J. Platt. Trends and controversies — support vector machines. *IEEE Intelligent Systems*, 13:18–28, 1998.

[26] R. Herbrich. *Learning kernel classifiers*. MIT Press, Cambridge, MA, 2002.

[27] T. S. Jaakkola, M. Diekhans, and D. Haussler. A discriminative framework for detecting remote protein homologies. *Journal of Computational Biology*, 7:95–114, 2000.

[28] T. S. Jaakkola and D. Haussler. Probabilistic kernel regression models. In *Proceedings of the 1999 Conference on AI and Statistics*, 1999.

[29] T. Joachims. Text categorization with support vector machines: Learning with many relevant features. In Claire Nédellec and Céline Rouveirol, editors, *Proceedings of the European Conference on Machine Learning*, pages 137–142, Berlin, 1998. Springer.

[30] G. S. Kimeldorf and G. Wahba. A correspondence between Bayesian estimation on stochastic processes and smoothing by splines. *Annals of Mathematical Statistics*, 41:495–502, 1970.

[31] G. S. Kimeldorf and G. Wahba. Some results on Tchebycheffian spline functions. *Journal of Mathematical Analysis and Applications*, 33:82–95, 1971.

[32] I. Kondor and J. Lafferty. Diffusion kernels on graphs and other discrete structures. In *Proceedings of ICML'2002*, 2002.

[33] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86:2278–2324, 1998.

[34] H. Lodhi, J. Shawe-Taylor, N. Cristianini, and C. Watkins. Text classification using string kernels. Technical Report 2000-79, NeuroCOLT, 2000. Published in: T. K. Leen, T. G. Dietterich and V. Tresp (eds.), *Advances in Neural Information Processing Systems 13*, MIT Press, 2001.

[35] D. J. C. MacKay. Introduction to Gaussian processes. In C. M. Bishop, editor, *Neural Networks and Machine Learning*, pages 133–165. Springer-Verlag, Berlin, 1998.

[36] J. Mercer. Functions of positive and negative type and their connection with the theory of integral equations. *Philosophical Transactions of the Royal Society, London*, A 209:415–446, 1909.

[37] E. Osuna, R. Freund, and F. Girosi. Support vector machines: Training and applications. Technical Report AIM-1602, MIT A.I. Lab., 1996.

[38] P. Pavlidis, J. Weston, J. Cai, and W. N. Grundy. Gene functional classification from heterogeneous data. In *Proceedings of the Fifth International Conference on Computational Molecular Biology*, pages 242–248, 2001.

[39] 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.

[40] T. Poggio and F. Girosi. Networks for approximation and learning. *Proceedings of the IEEE*, 78(9), September 1990.

[41] S. Romdhani, S. Gong, and A. Psarrou. A multiview nonlinear active shape model using kernel PCA. In *Proceedings of BMVC*, pages 483–492, Nottingham, UK, 1999.

[42] S. Saitoh. *Theory of Reproducing Kernels and its Applications*. Longman Scientific & Technical, Harlow, England, 1988.

[43] B. Schölkopf. *Support Vector Learning*. R. Oldenbourg Verlag, München, 1997. Doktorarbeit, Technische Universität Berlin. Available from http://www.kyb.tuebingen.mpg.de/∼bs.

[44] B. Schölkopf and A. J. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.

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

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

[47] B. Schölkopf, J. Weston, E. Eskin, C. Leslie, and W. S. Noble. A kernel approach for learning from almost orthogonal patterns. In *Proceedings of the 13th European Conference on Machine Learning (ECML'2002) and Proceedings of the 6th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD'2002), Helsinki*, volume 2430/2431 of *Lecture Notes in Computer Science*, Berlin, 2002. Springer.

[48] M. Seeger. Bayesian methods for support vector machines and Gaussian processes. Master's thesis, University of Edinburgh, Division of Informatics, 1999.

[49] J. Shawe-Taylor and N. Cristianini. *Kernel methods for pattern analysis*. Cambridge University Press, 2004.

[50] P. Simard, Y. LeCun, and J. Denker. Efficient pattern recognition using a new transformation distance. In S. J. Hanson, J. D. Cowan, and C. L. Giles, editors, *Advances in Neural Information Processing Systems 5. Proceedings of the 1992 Conference*, pages 50–58, San Mateo, CA, 1993. Morgan Kaufmann.

[51] S. Tong and D. Koller. Support vector machine active learning with applications to text classification. In P. Langley, editor, *Proceedings of the 17th International Conference on Machine Learning*, San Francisco, California, 2000. Morgan Kaufmann.

[52] K. Tsuda, M. Kawanabe, G. Rätsch, S. Sonnenburg, and K.R. Müller. A new discriminative kernel from probabilistic models. In T.G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems*, volume 14. MIT Press, 2002.

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

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

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

[56] J.-P. Vert. A tree kernel to analyze phylogenetic profiles. In *Proceedings of ISMB'02*, 2002.

[57] U. von Luxburg, O. Bousquet, and B. Schölkopf. A compression approach to support vector model selection. Technical report, Max Planck Institute for Biological Cybernetics, 2002. To appear in JMLR, 2004.

[58] G. Wahba. *Spline Models for Observational Data*, volume 59 of *CBMS-NSF Regional Conference Series in Applied Mathematics*. Society for Industrial and Applied Mathematics, Philadelphia, Pennsylvania, 1990.

[59] M. K. Warmuth, G. Rätsch, M. Mathieson, J. Liao, and C. Lemmen. Active learning in the drug discovery process. In T.G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems*, volume 14. MIT Press, 2002. To appear.

[60] C. Watkins. Dynamic alignment kernels. In A. J. Smola, P. L. Bartlett, B. Schölkopf, and D. Schuurmans, editors, *Advances in Large Margin Classifiers*, pages 39–50, Cambridge, MA, 2000. MIT Press.

[61] H. L. Weinert, editor. *Reproducing Kernel Hilbert Spaces — Applications in Statistical Signal Processing*. Hutchinson Ross, Stroudsburg, PA, 1982.

[62] J. Weston, O. Chapelle, A. Elisseeff, B. Schölkopf, and V. Vapnik. Kernel dependency estimation. Technical Report 98, Max Planck Institute for Biological Cybernetics, 2002.

[63] C. K. I. Williams. Prediction with Gaussian processes: From linear regression to linear prediction and beyond. In M. I. Jordan, editor, *Learning and Inference in Graphical Models*. Kluwer, 1998.

[64] C.-H. Yeang, S. Ramaswamy, P. Tamayo, S. Mukherjee, R. M. Rifkin, M. Angelo, M. Reich, E. Lander, J. Mesirov, and T. Golub. Molecular classification of multiple tumor types. *Bioinformatics*, 17:S316–S322, 2001. ISMB'01 Supplement.

[65] A. Zien, G. Rätsch, S. Mika, B. Schölkopf, T. Lengauer, and K.-R. Müller. Engineering support vector machine kernels that recognize translation initiation sites. *Bioinformatics*, 16(9):799–807, 2000.