

# SVM and Kernel Methods

---

*Bernhard Schölkopf*

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

*Biowulf Technologies  
305 Broadway, New York, NY 10007, USA  
bernhard.schoelkopf@tuebingen.mpg.de*

*(these slides are available from [www.kernel-machines.org](http://www.kernel-machines.org))*

# Roadmap

---

1. ideas of statistical learning theory
2. kernels and feature spaces
3. Support vector algorithms

# Statistical Learning Theory

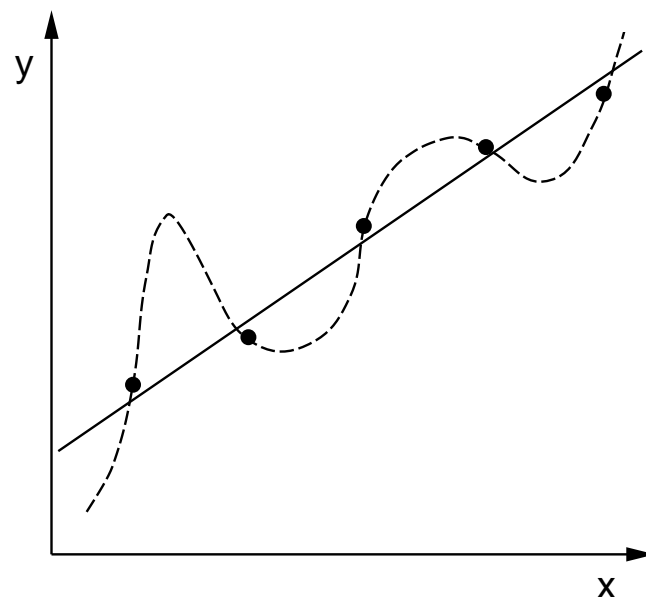
---

1. started by Vapnik and Chervonenkis in the Sixties
2. model: we observe data generated by an unknown stochastic regularity
3. *learning* = extraction of the regularity from the data
4. the analysis of the learning problem leads to notions of *capacity* of the function classes that a learning machine can implement.
5. *support vector machines* use a particular type of function class: classifiers with large “margins” in a feature space induced by a *kernel*.

[49, 50]

## Example: Regression Estimation

---



- *Data*: input-output pairs  $(x_i, y_i) \in \mathbb{R} \times \mathbb{R}$
- *Regularity*:  $(x_1, y_1), \dots, (x_m, y_m)$  drawn from  $P(x, y)$
- *Learning*: choose a function  $f : \mathbb{R} \rightarrow \mathbb{R}$  such that the error, averaged over  $P$ , is minimized.
- *Problem*:  $P$  is unknown, so the average cannot be computed — need an “*induction principle*”

# Pattern Recognition

---

Learn  $f : \mathcal{X} \rightarrow \{\pm 1\}$  from examples

$(x_1, y_1), \dots, (x_m, y_m) \in \mathcal{X} \times \{\pm 1\}$ , generated i.i.d. from  $P(x, y)$ ,

such that the expected misclassification error on a test set, also drawn from  $P(x, y)$ ,

$$R[f] = \int \frac{1}{2} |f(x) - y| dP(x, y),$$

is minimal (*Risk Minimization (RM)*).

**Problem:**  $P$  is unknown.  $\longrightarrow$  need an *induction principle*.

*Empirical risk minimization (ERM)*: replace the average over  $P(x, y)$  by an average over the training sample, i.e. **minimize the training error**

$$R_{\text{emp}}[f] = \frac{1}{m} \sum_{i=1}^m \frac{1}{2} |f(x_i) - y_i|$$

- **Regression estimation.** RM: minimize

$$R[f] = \int (f(x) - y)^2 dP(x, y)$$

— leads to the *regression*  $y(x) = \int y dP(y|x)$ .

ERM gives **least mean squares**: minimize

$$\sum_i (f(x_i) - y_i)^2$$

- **Density estimation.** RM: minimize

$$R[f] = \int (-\log p(x)) dP(x)$$

ERM gives **maximum likelihood estimation**: maximize

$$\sum_i \log p(x_i) = \log\left(\prod_i p(x_i)\right)$$

# Convergence of Means to Expectations

---

Law of large numbers:

$$R_{\text{emp}}[f] \rightarrow R[f]$$

as  $m \rightarrow \infty$ .

Does this imply that for the function  $f^m$  minimizing  $R_{\text{emp}}$ , and the function  $f^{\text{opt}}$  minimizing  $R$ , we have

$$R_{\text{emp}}[f^m] \rightarrow R[f^{\text{opt}}], \quad R[f^m] \rightarrow R[f^{\text{opt}}]$$

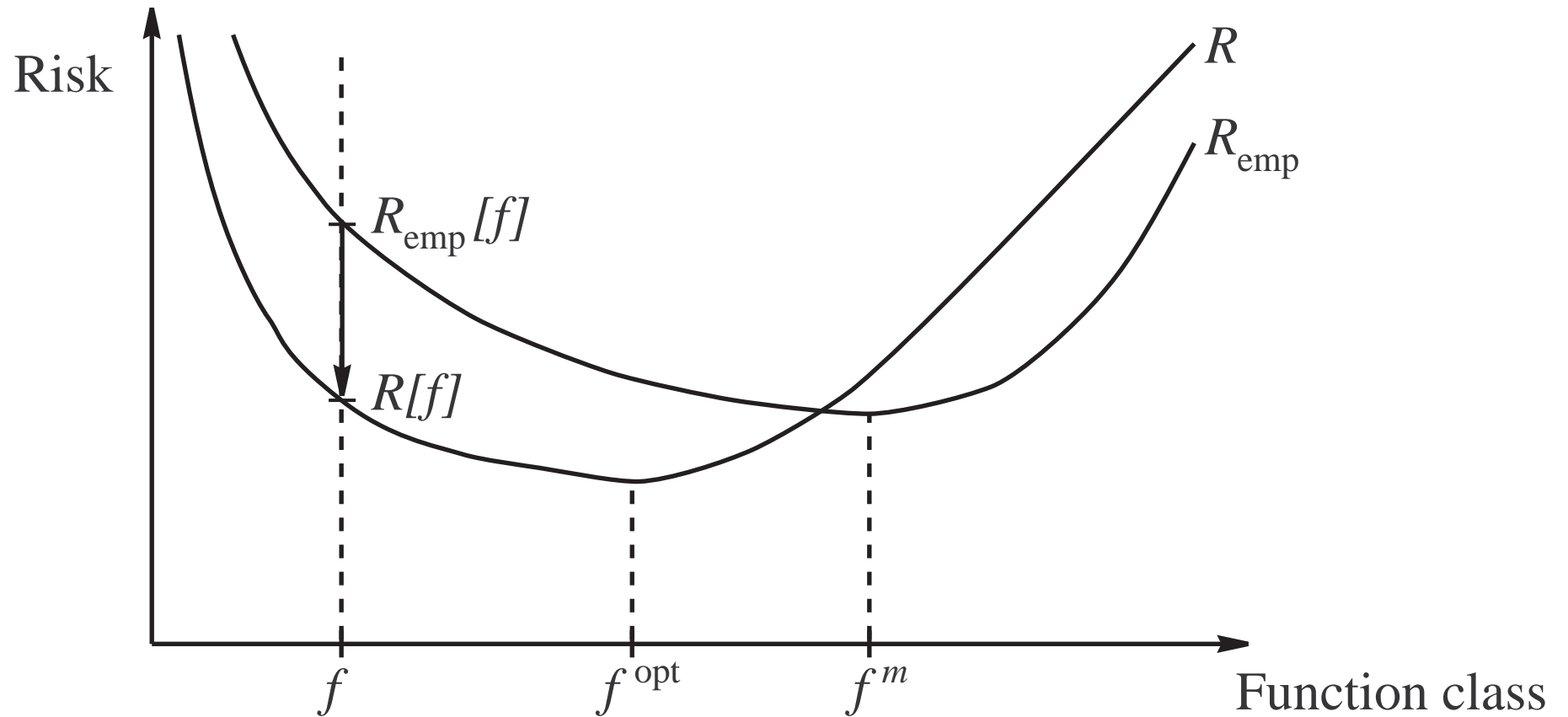
as  $m \rightarrow \infty$  (“*consistency*” of empirical risk minimization)?

**No.**

Need a *uniform* version of the law of large numbers. Uniform over *all functions that the learning machine can implement*.

# Consistency and Uniform Convergence

---





## Vapnik-Chervonenkis(VC)-Theory: Main Points

---

Necessary and sufficient conditions for **consistency of empirical risk minimization**: one-sided convergence, uniformly over all functions that can be implemented by the learning machine.

$$\lim_{m \rightarrow \infty} P\{\sup_f (R[f] - R_{emp}[f]) > \epsilon\} = 0 \text{ for all } \epsilon > 0.$$

Vapnik, Chervonenkis and others give conditions for uniform convergence in terms of **capacity concepts**, e.g.

- the VC-entropy grows sublinearly with  $m$
- the VC-dimension is finite
- the entropy numbers are well-behaved
- the classification “margin” is large

[e.g. 52, 50, 44, 61, 1]

# Conditions for Uniform Convergence

---

How to bound  $P\{\sup_{f \in \mathcal{F}} (R[f] - R_{\text{emp}}[f]) > \epsilon\}$ :

- if the function class  $\mathcal{F}$  contains only one function, then *Chernoff's bound* suffices:

$$P\{\sup_{f \in \mathcal{F}} (R[f] - R_{\text{emp}}[f]) > \epsilon\} \leq 2 \exp(-2m\epsilon^2)$$

- if there are finitely many functions, use the *union bound* to get a multiplicative constant on the RHS
- even if there are infinitely many, then *on any finite sample* there are effectively only finitely many (use *symmetrization*, [52])

- *VC entropy*: on an example  $(x, y)$ ,  $f$  causes a loss  $Q(x, y, f)$ . On a training set, different functions  $f \in \mathcal{F}$  lead to  $N^{\mathcal{F}}$  different loss vectors  $q_f = (Q(x_1, y_1, f), \dots, Q(x_m, y_m, f))$ . Define

$$H^{\mathcal{F}}(m) = \mathbf{E} \ln N^{\mathcal{F}}.$$

$H^{\mathcal{F}}(m)/m \rightarrow 0 \iff$  uniform convergence (hence consistency)

- exchange expectation and logarithm: *annealed entropy*.

$H_{ann}^{\mathcal{F}}(m)/m \rightarrow 0 \implies$  exponential convergence

$$P\left\{\sup_f (R[f] - R_{\text{emp}}[f]) > \epsilon\right\} \leq 4 \cdot \exp\left(\left(\left(H_{ann}^{\mathcal{F}}(2m)/m\right) - \epsilon^2\right) \cdot m\right).$$

- take 'max' instead of ' $\mathbf{E}$ ': *growth function*.

$G^{\mathcal{F}}(m)/m \rightarrow 0 \iff$  exponential convergence for all underlying distributions

## Structure of the Growth Function

---

**Either**  $G^{\mathcal{F}}(m) = m \cdot \ln(2)$  — this means that for any sample size  $m$  the points can be chosen such that by using functions of the learning machine, all  $2^m$  possible loss vectors can be generated (i.e., they can be separated in all possible ways — “*shattered*”).

**Or** there exists some *maximal*  $m$  for which the above is possible. Call this number the *VC-dimension*, and denote it by  $h$ . Then one can prove that for  $m > h$ ,

$$G^{\mathcal{F}}(m) \leq h \left( \ln \frac{m}{h} + 1 \right).$$

Nothing “in between” linear growth and logarithmic growth is possible [51].

## A VC Bound for Pattern Recognition

---

For any  $f \in \mathcal{F}$  and  $m > h$ , with a probability of at least  $1 - \eta$ ,

$$R[f] \leq R_{\text{emp}}[f] + \phi \left( \frac{h}{m}, \frac{\log(\eta)}{m} \right)$$

holds, where  $\phi$  is defined as

$$\phi \left( \frac{h}{m}, \frac{\log(\eta)}{m} \right) = \sqrt{\frac{h \left( \log \frac{2m}{h} + 1 \right) - \log(\eta/4)}{m}}.$$

(Derivation: in uniform convergence bounds, set RHS =  $\eta$ , and solve for  $\epsilon$  to get the confidence term.)

The study of the consistency of ERM has thus led to concepts and results which lets us formulate a better induction principle: minimize the RHS of the bound.

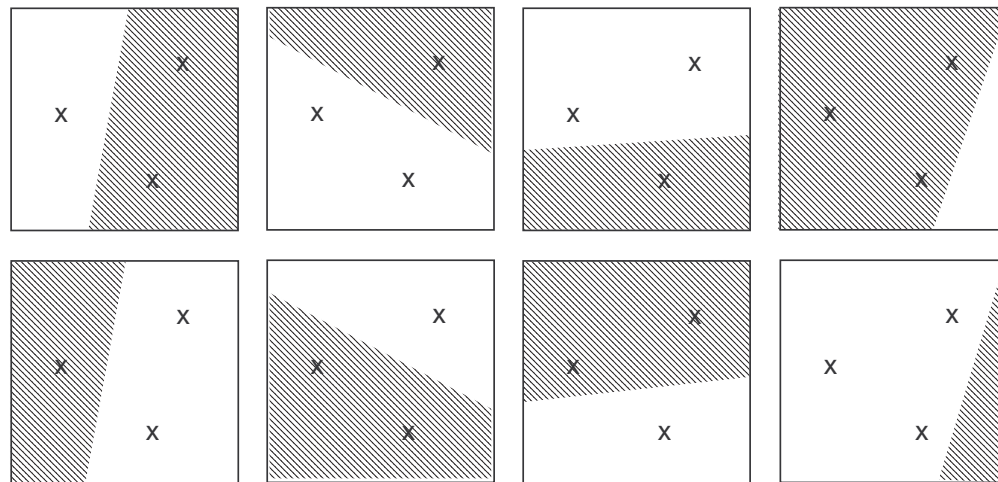
# VC-Dimension: Example

---

Half-spaces in  $\mathbb{R}^2$ :

$$f(x, y) = \text{sgn}(a + bx + cy), \quad \text{with parameters } a, b, c \in \mathbb{R}$$

- Clearly, we can shatter three non-collinear points.
- But we can never shatter four points.
- Hence the VC dimension is  $h = 3$  (in this case, equal to the number of parameters)



## VC-Dimension Example, ctd.

---

- more generally, separating hyperplanes in  $\mathbb{R}^N$  have a VC dimension of  $N + 1$ .
- hence: separating hyperplanes in high-dimensional feature spaces have extremely large VC dimension, and may not generalize well
- however, “*margin*” hyperplanes can still have a small VC dimension (see below)

# The Kernel Trick: Feature Spaces

---

Preprocess the data with

$$\begin{aligned}\Phi : \mathcal{X} &\rightarrow \mathcal{H} \\ x &\mapsto \Phi(x),\end{aligned}$$

where  $\mathcal{H}$  is a dot product space, and learn the mapping from  $\Phi(x)$  to  $y$ .

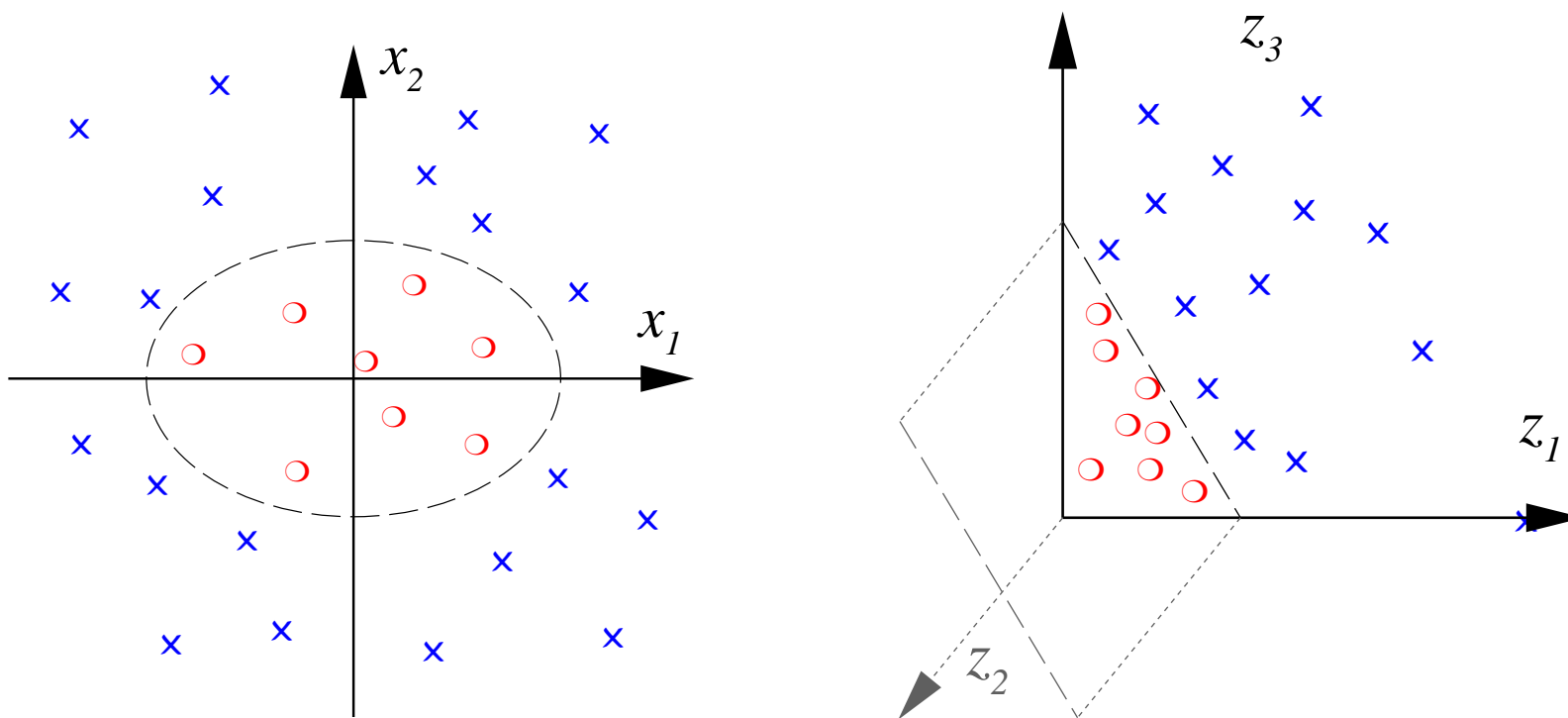
- usually,  $\dim(\mathcal{X}) \ll \dim(\mathcal{H})$
- “Curse of Dimensionality”?
- crucial issue: *capacity*, not *dimensionality*



## Example: All Degree 2 Monomials

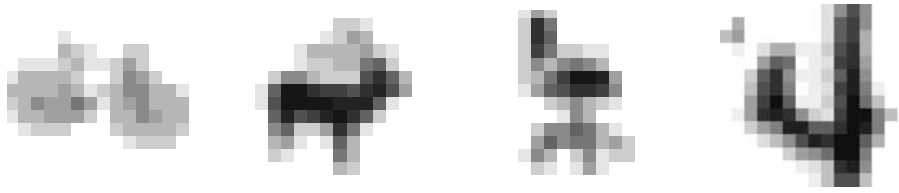
---

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



# 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, $N = d = 2$

---

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

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

## The Kernel Trick, II

---

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

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

where  $\Phi$  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  $\psi_i$  and eigenvalues  $\lambda_i \geq 0$  [34].

In that case

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

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

# Positive Definite Kernels

---

It can be shown that (modulo some details) the admissible class of kernels coincides with the one of **positive definite (pd) kernels**: kernels which are symmetric, 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

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

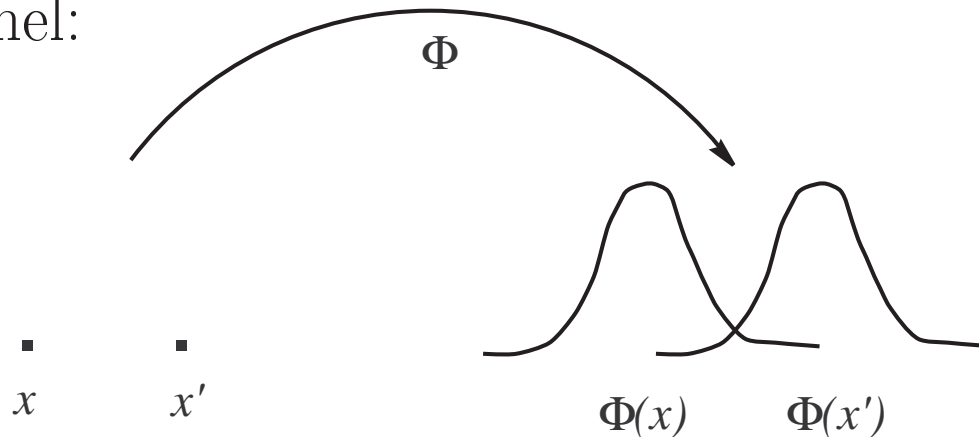
# The Feature Space for PD Kernels

[5, 2, 38]

- define a feature map

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

E.g., for the Gaussian kernel:



- turn  $\Phi(\mathcal{X})$  into a linear space,  $f(\cdot) = \sum_{i=1}^m \alpha_i k(\cdot, x_i)$ ,
- endow it with a dot product satisfying  $\langle k(\cdot, x_i), k(\cdot, x_j) \rangle = k(x_i, x_j)$
- complete the space to get a *reproducing kernel Hilbert space*

## Some Properties of Kernels

---

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

- $\alpha k_1$ , provided  $\alpha \geq 0$
- $k_1 + k_2$
- $k_1 \cdot k_2$
- $k(x, x') := \lim_{n \rightarrow \infty} k_n(x, x')$ , provided it exists

Further operations to construct kernels from kernels: tensor products, direct sums, convolutions [23].



# 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:

$$\text{Polynomial } k(x, x') = (\langle x, x' \rangle + c)^d$$

$$\text{Sigmoid } k(x, x') = \tanh(\kappa \langle x, x' \rangle + \Theta)$$

$$\text{Gaussian } k(x, x') = \exp(-\|x - x'\|^2 / (2\sigma^2))$$

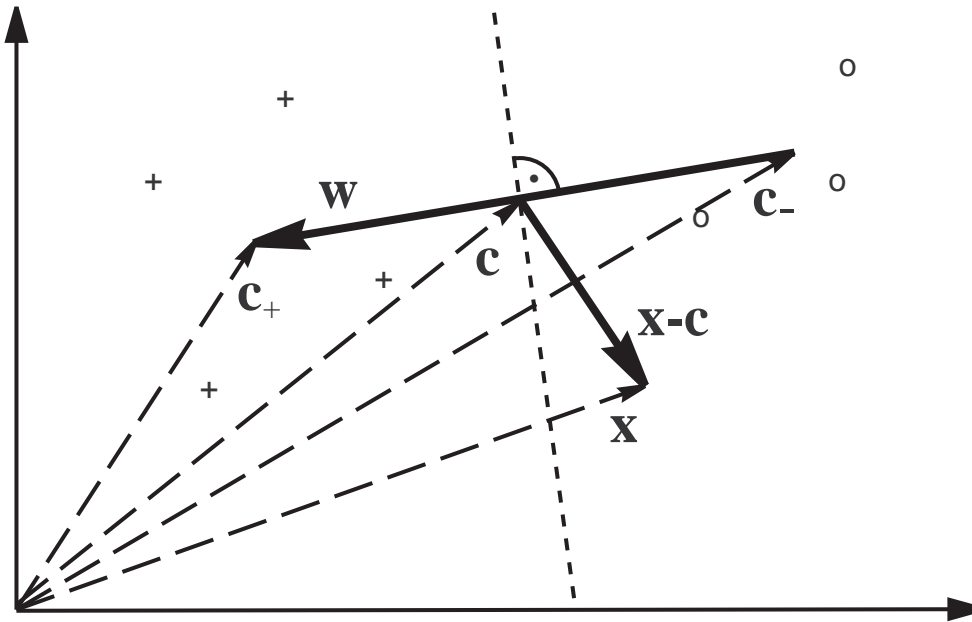
- Kernel are studied also in the Gaussian Process prediction community (covariance functions) [57, 54, 58, 33]

## 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.

$$\mathbf{c}_+ := \frac{1}{m_1} \sum_{y_i=1} \Phi(x_i), \quad \mathbf{c}_- := \frac{1}{m_2} \sum_{y_i=-1} \Phi(x_i)$$



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.

---

$$\begin{aligned} f(x) &= \operatorname{sgn} \left( \frac{1}{m_1} \sum_{\{i:y_i=+1\}} \langle \Phi(x), \Phi(x_i) \rangle - \frac{1}{m_2} \sum_{\{i:y_i=-1\}} \langle \Phi(x), \Phi(x_i) \rangle + b \right) \\ &= \operatorname{sgn} \left( \frac{1}{m_1} \sum_{\{i:y_i=+1\}} k(x, x_i) - \frac{1}{m_2} \sum_{\{i:y_i=-1\}} k(x, x_i) + b \right) \end{aligned}$$

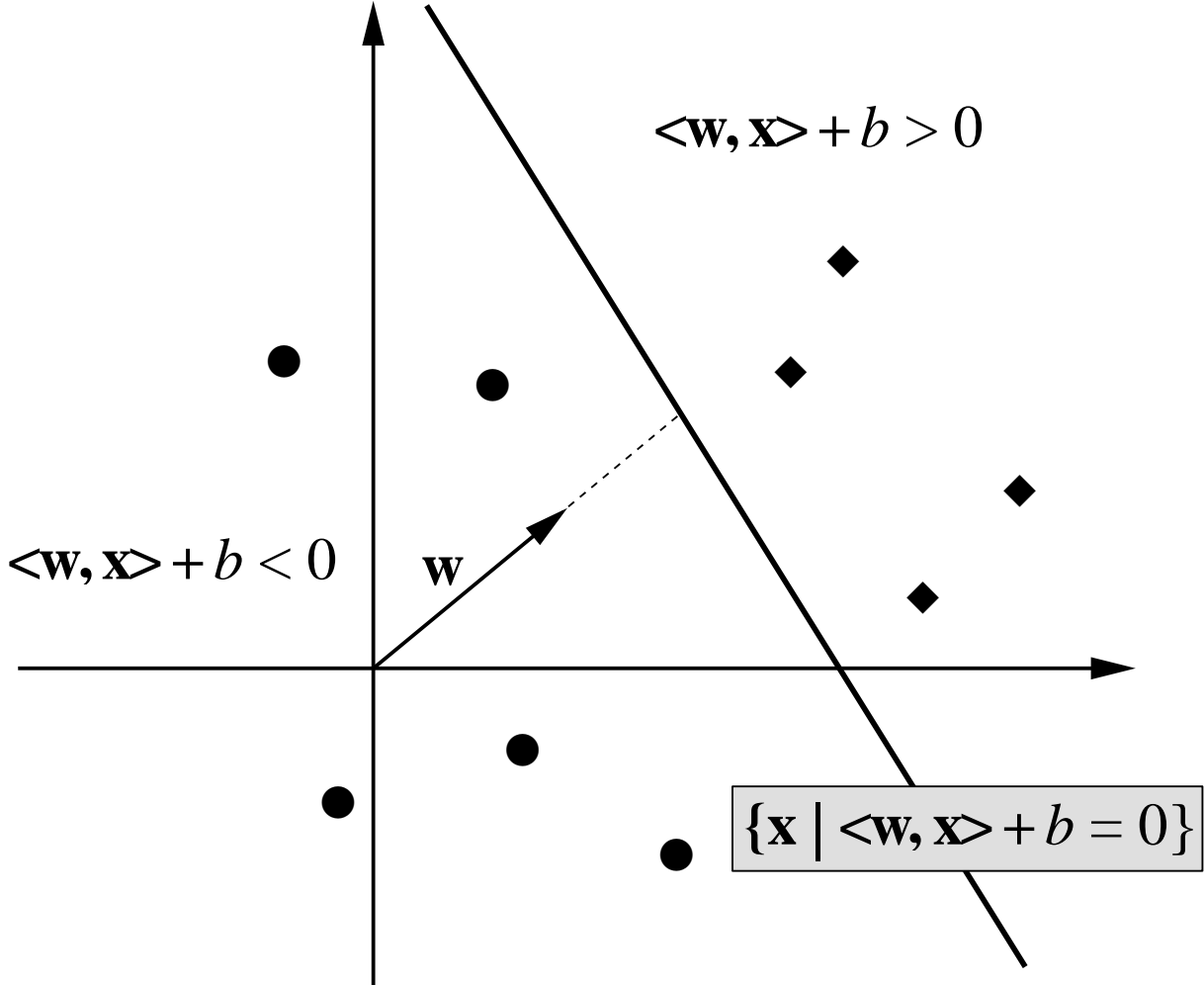
where

$$b = \frac{1}{2} \left( \frac{1}{m_2^2} \sum_{\{(i,j):y_i=y_j=-1\}} k(x_i, x_j) - \frac{1}{m_1^2} \sum_{\{(i,j):y_i=y_j=+1\}} k(x_i, x_j) \right).$$

- cf. Parzen windows
- the decision function is a hyperplane. Will it generalize well?

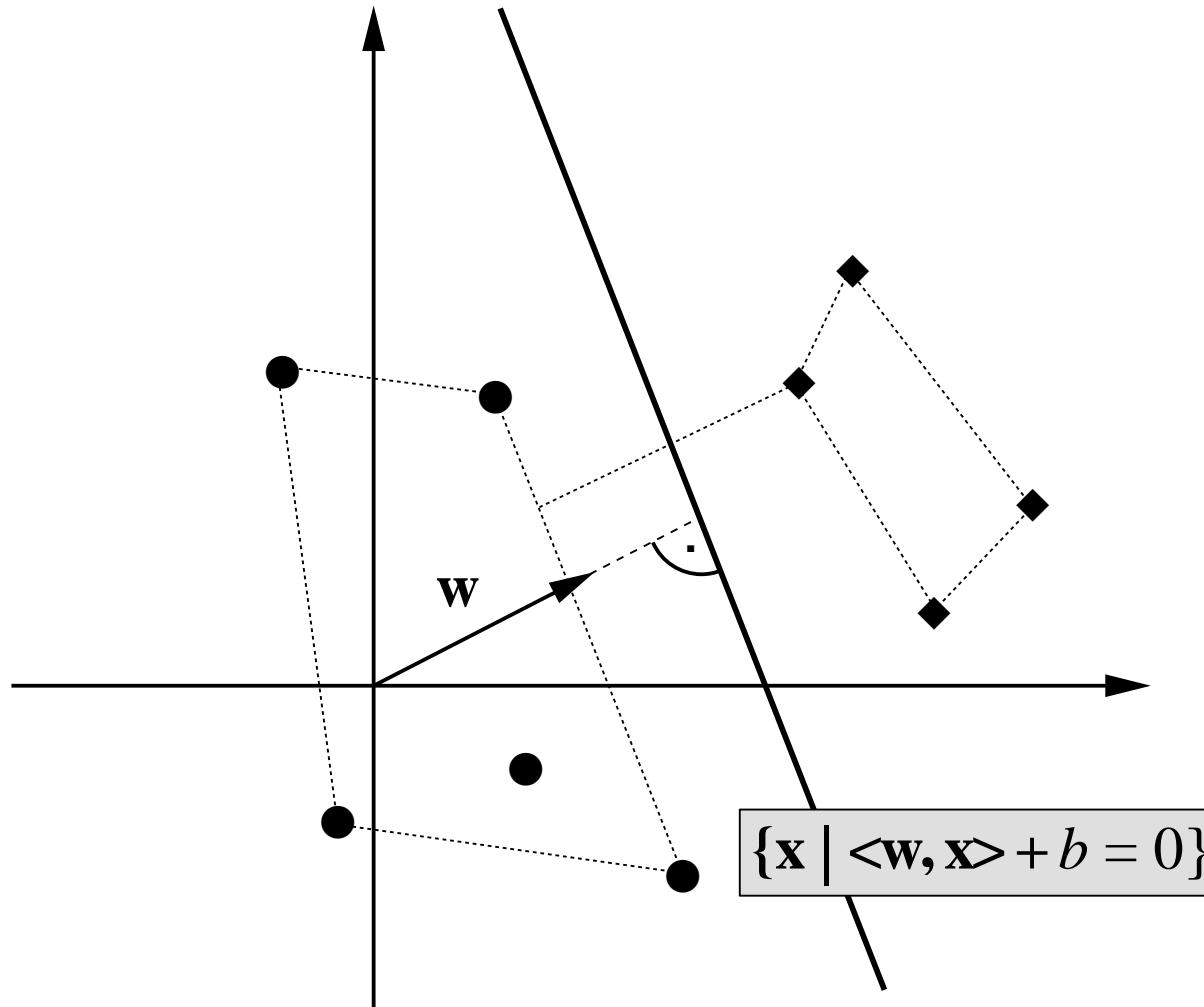
# Separating Hyperplane

---



# Optimal Separating Hyperplane

[53]



## Eliminating the Scaling Freedom

---

[49]

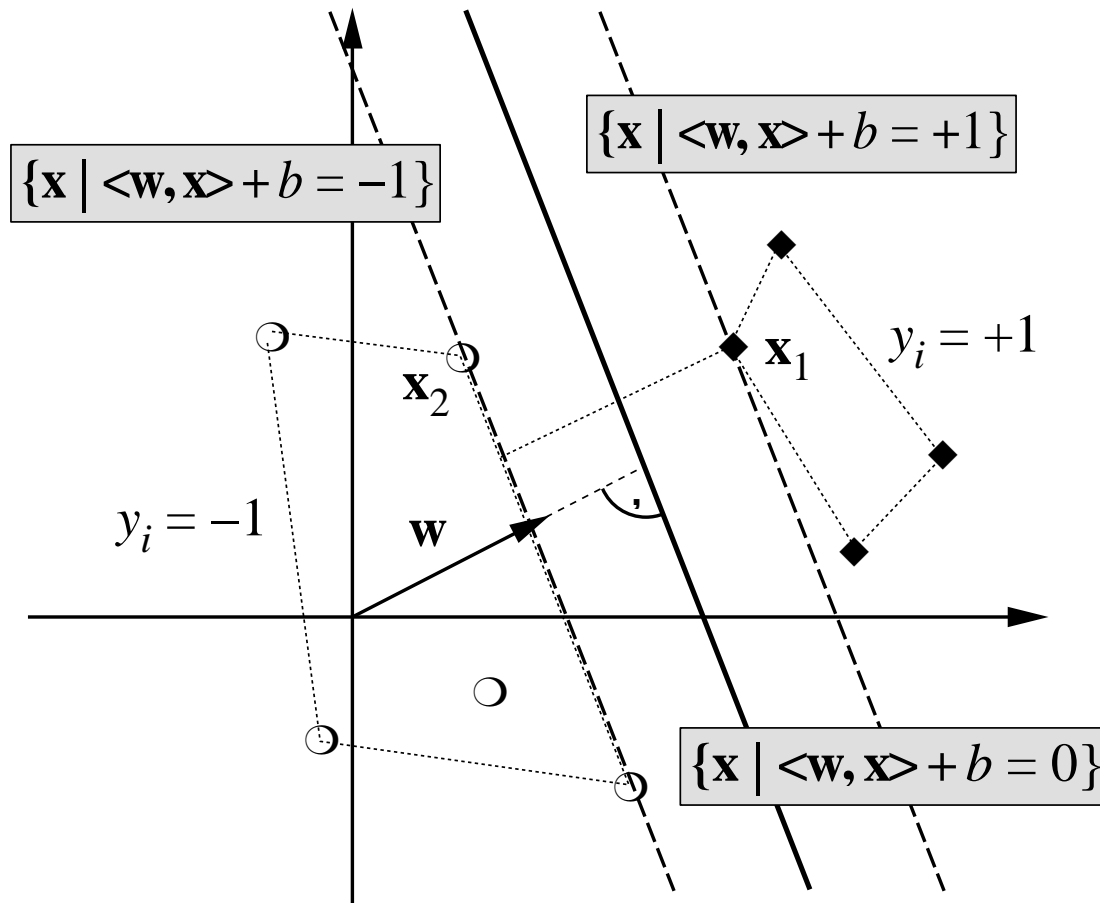
Note: if  $c \neq 0$ , then

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

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

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

# Canonical Optimal Hyperplane



Note:

$$\begin{aligned}\langle w, x_1 \rangle + b &= +1 \\ \langle w, x_2 \rangle + b &= -1\end{aligned}$$

$$\Rightarrow \langle w, (x_1 - x_2) \rangle = 2$$

$$\Rightarrow \left\langle \frac{w}{\|w\|}, (x_1 - x_2) \right\rangle = \frac{2}{\|w\|}$$

## VC Dimension of Margin Hyperplanes

---

**Theorem [48].** Consider hyperplanes  $\langle \mathbf{w}, \mathbf{x} \rangle = 0$  where  $\mathbf{w}$  is normalized such that they are in canonical form w.r.t. a set of points  $X^* = \{\mathbf{x}_1, \dots, \mathbf{x}_r\}$ , i.e.,

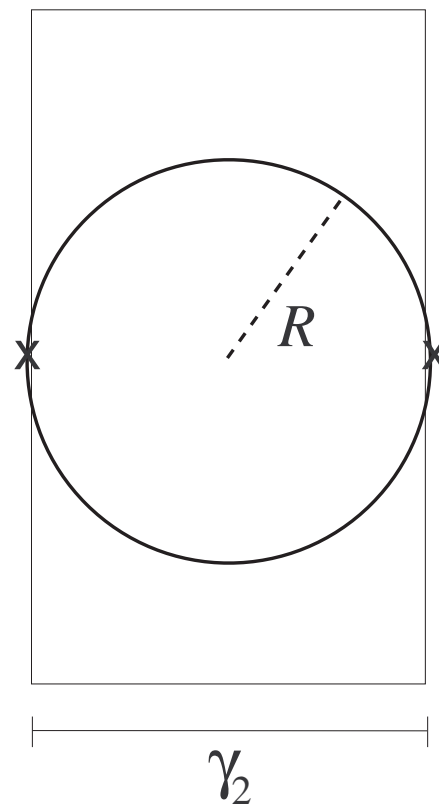
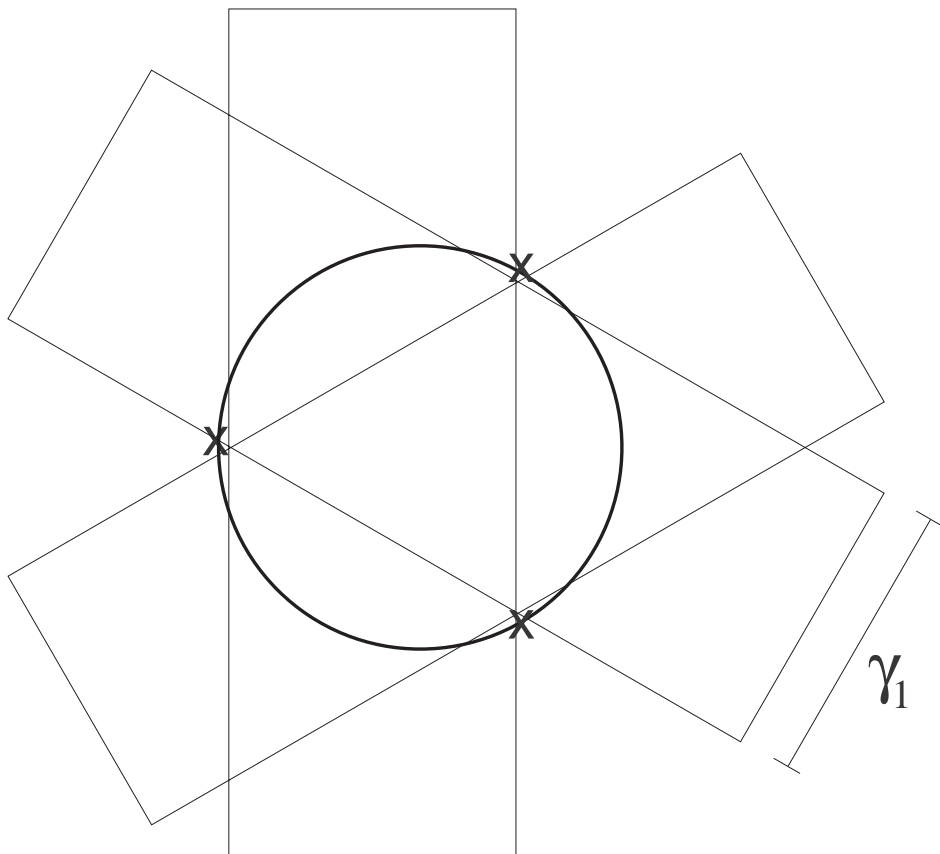
$$\min_{i=1, \dots, r} |\langle \mathbf{w}, \mathbf{x}_i \rangle| = 1.$$

The set of decision functions  $f_{\mathbf{w}}(\mathbf{x}) = \text{sgn} \langle \mathbf{x}, \mathbf{w} \rangle$  defined on  $X^*$  and satisfying the constraint  $\|\mathbf{w}\| \leq \Lambda$  has a VC dimension satisfying

$$h \leq R^2 \Lambda^2.$$

Here,  $R$  is the radius of the smallest sphere around the origin containing  $X^*$ .





# 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., [6])

---

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

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

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

- if a constraint is violated, then  $y_i \cdot (\langle \mathbf{w}, \mathbf{x}_i \rangle + b) - 1 < 0 \longrightarrow$ 
  - $\alpha_i$  will grow to increase  $L$  — how far?
  - $\mathbf{w}$ ,  $b$  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  $\alpha_i$  (*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  $i = 1, \dots, m$  either

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

or

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

The solution is determined by the examples on the margin.

Thus

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

## 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

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

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 SV Expansion in Feature Space

---

- generally, the solution of kernel algorithms corresponds to a single vector in  $\mathcal{H}$  (“Representer Theorem” [30, 39]),

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

However, there is usually no  $x \in \mathcal{X}$  such that

$$\Phi(x) = \mathbf{w},$$

i.e.,  $\Phi(\mathcal{X})$  is not closed under linear combinations — it is a nonlinear manifold (cf. [10, 40]).

- $\Phi(\mathcal{X})$  is contained in a non-isotropic shape whose sidelengths scale like the square roots of the eigenvalues of  $k$  or  $K$  [cf. 61, 60, 13, 59].

# Regularization Interpretation of Kernel Machines

---

The norm in  $\mathcal{H}$  can be interpreted as a regularization term [21, 46, 19]: if  $P$  is a regularization operator such that  $k$  is Green's function of  $P^*P$ , then

$$\|\mathbf{w}\| = \|Pf\|,$$

where

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

and

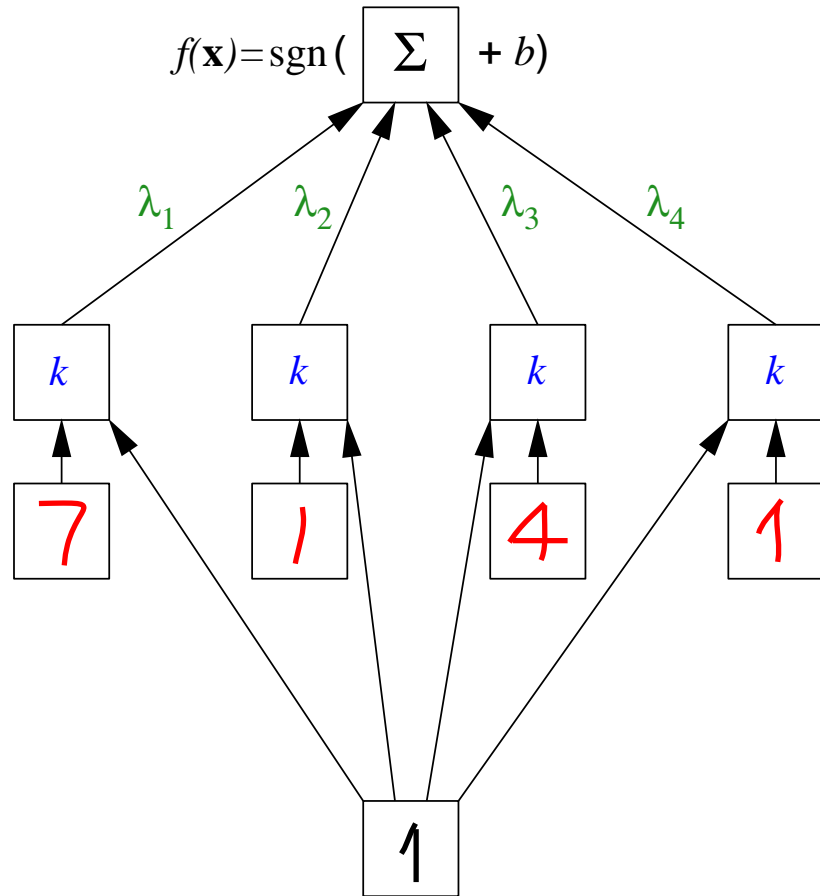
$$f(x) = \sum_i \alpha_i k(x_i, x).$$

Example: for Gaussian kernel,  $P$  is a linear combination of differential operators.

Corresponding MAP interpretation with prior  $\exp(-\lambda\|Pf\|^2)$  [29].



# The SVM Architecture



classification

weights

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

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

input vector  $\mathbf{x}$

$$f(\mathbf{x}) = \text{sgn}(\Sigma \lambda_i k(\mathbf{x}, \mathbf{x}_i) + b)$$

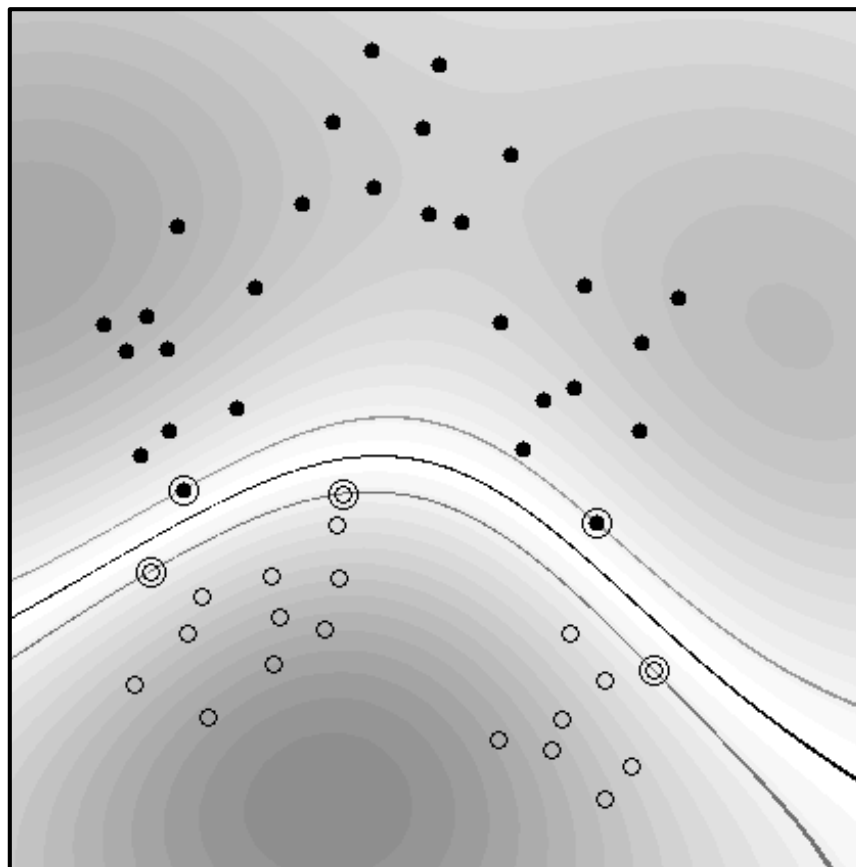
$$k(\mathbf{x}, \mathbf{x}_i) = \exp(-\|\mathbf{x} - \mathbf{x}_i\|^2 / c)$$

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

# Toy Example with Gaussian Kernel

---

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



# Nonseparable Problems

[4, 15]

---

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

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.

Same dual, with additional constraints  $\alpha_i \leq C$ .

# 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 [48].
- in fact, one can use chunks which do not even contain all SVs [35]. 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 [37].
- <http://www.kernel-machines.org/software.html>

## MNIST Error Rates

---

handwritten character benchmark (60000 training & 10000 test examples,  $28 \times 28$ )

Classifier	test error	reference
linear classifier	8.4%	[7]
3-nearest-neighbour	2.4%	[7]
SVM	1.4%	[11]
Tangent distance	1.1%	[45]
LeNet4	1.1%	[31]
Boosted LeNet4	0.7%	[31]
Translation invariant SVM	0.56%	[17]

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: [28, 26, 24, 12, 47, 8, 63, 22, 20, 14, 18, 36, 55, 62]

# Unsupervised SVM Learning

---

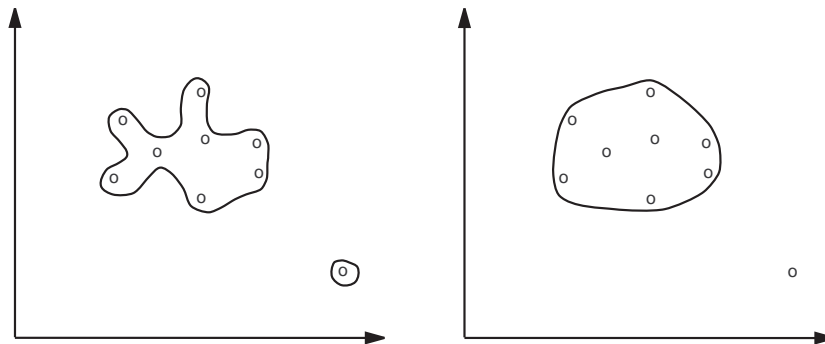
$x_1, \dots, x_m \in \mathcal{X}$  i.i.d. sample from  $P$

- extreme view: unsupervised learning = density estimation
- easier problem: for  $\alpha \in (0, 1]$ , compute a region  $R$  such that

$$P(R) \approx \alpha,$$

i.e., estimate *quantiles* of a distribution, not its density.

- becomes well-posed using a regularizer: find “smoothest” region that contains a certain fraction of the probability mass
- given only the training data, we will get a trade-off: try to enclose many training points (more than  $\alpha$ ) in a smooth region



# Multi-Dimensional Quantiles

---

- $\mathcal{C}$  a class of measurable subsets of  $\mathcal{X}$
- $\lambda$  a real-valued function on  $\mathcal{C}$
- *quantile function* with respect to  $(P, \lambda, \mathcal{C})$ :

$$U(\alpha) = \inf\{\lambda(C) \mid P(C) \geq \alpha, C \in \mathcal{C}\} \quad 0 < \alpha \leq 1.$$

- present case [41]:  $\lambda(C) \propto \frac{1}{\text{margin}^2}$ , where

$$\mathcal{C} := \{\text{half-spaces in } \mathcal{H}, \text{ not containing the origin}\}$$

## Separating Unlabelled Data from the Origin

---

One can show: if  $\Phi(x_1), \dots, \Phi(x_m)$  are separable from the origin in  $\mathcal{H}$ , then the solution of

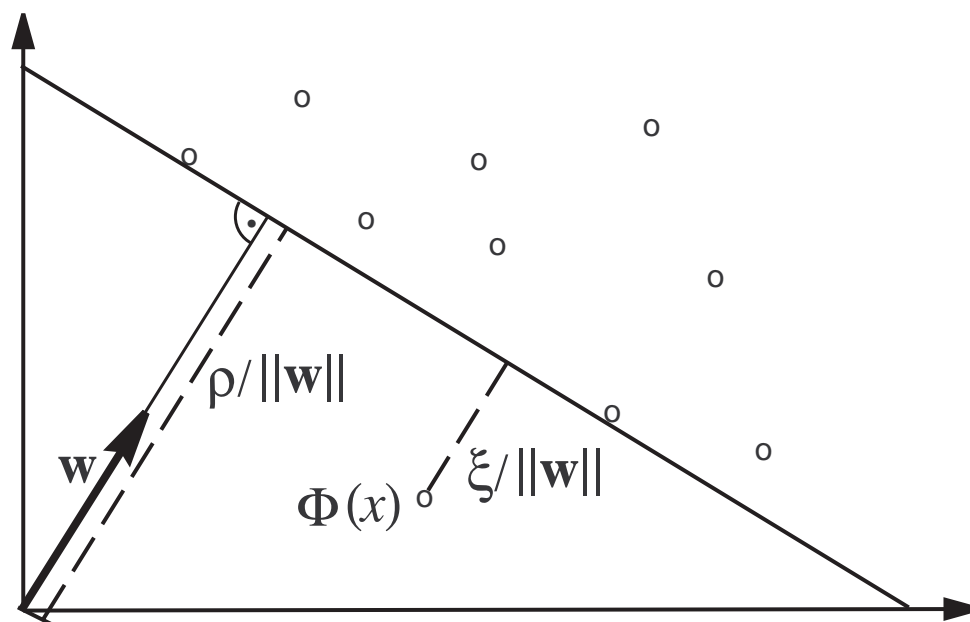
$$\min_{\mathbf{w} \in \mathcal{H}} \frac{1}{2} \|\mathbf{w}\|^2 \quad \text{subject to} \quad \langle \mathbf{w}, \Phi(x_i) \rangle \geq 1$$

is the normal vector of the hyperplane separating the data from the origin with **maximum margin**.



# $\nu$ -Soft Margin Separation

---



For  $\nu \in (0, 1]$ , compute

$$\min_{\mathbf{w} \in \mathcal{H}, \boldsymbol{\xi} \in \mathbb{R}^m, \rho \in \mathbb{R}} \frac{1}{2} \|\mathbf{w}\|^2 + \frac{1}{m} \sum_i \xi_i - \nu \rho$$

subject to  $\langle \mathbf{w}, \Phi(x_i) \rangle \geq \rho - \xi_i, \quad \xi_i \geq 0 \quad \text{for all } i.$

## Dual Problem

---

Derived using the Lagrange formalism:

$$\begin{aligned} \min_{\boldsymbol{\alpha} \in \mathbb{R}^m} \quad & \frac{1}{2} \sum_{ij} \alpha_i \alpha_j k(x_i, x_j) \\ \text{subject to} \quad & 0 \leq \alpha_i \leq \frac{1}{\nu m}, \quad \sum_i \alpha_i = 1. \end{aligned}$$

The decision function is

$$f(x) = \text{sgn} \left( \sum_i \alpha_i k(x_i, x) - \rho \right).$$

— a thresholded sparsified Parzen windows estimator

# Support Vectors and Outliers

---

$$SV := \{i | \alpha_i > 0\}; \quad OL := \{i | \xi_i > 0\}$$

The KKT-Conditions imply:

- $\xi_i > 0 \implies \alpha_i = 1/(\nu m)$ , hence  $OL \subset SV$
- $SV \setminus OL \subset \{i | \sum_j \alpha_j k(x_j, x_i) - \rho = 0\}$

## The Meaning of $\nu$

---

### Proposition.

(i)

$$\frac{|OL|}{m} \leq \nu \leq \frac{|SV|}{m}$$

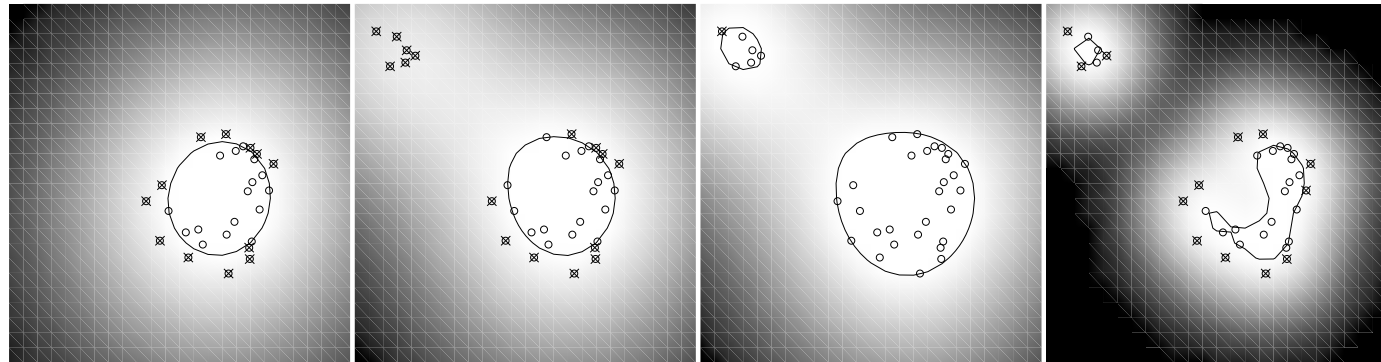
(ii) Suppose  $P$  does not contain discrete components, and the kernel is analytic and non-constant. With probability 1, asymptotically,

$$\frac{|OL|}{m} = \nu = \frac{|SV|}{m}.$$

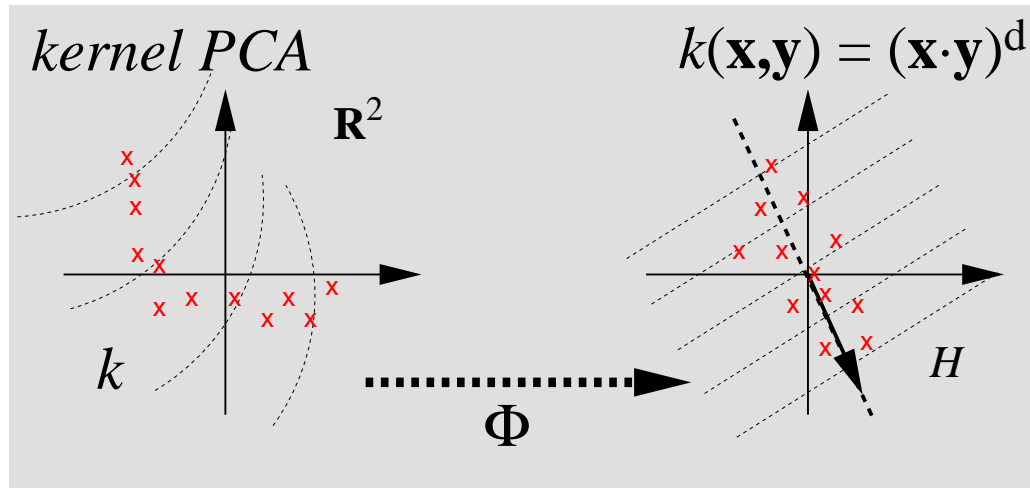
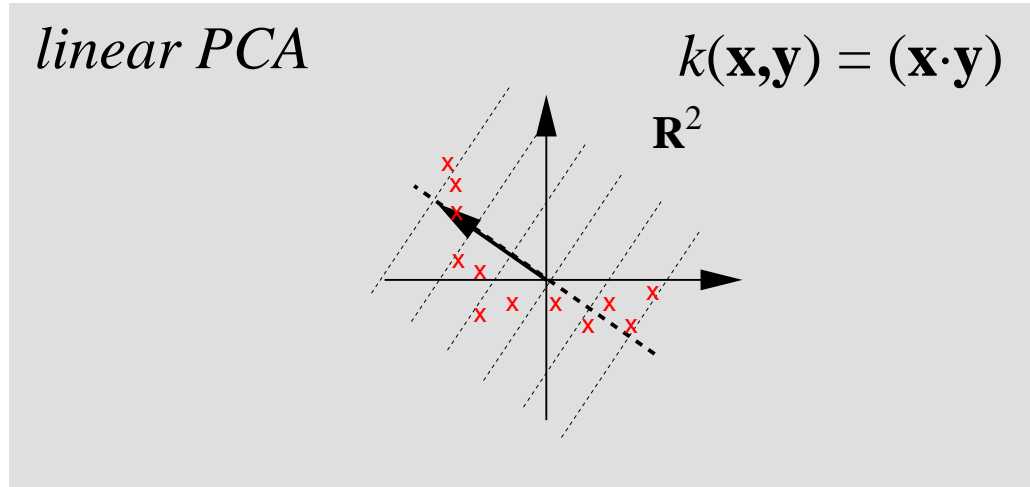
There are also  $\nu$ -versions of SV pattern recognition and SV regression.

# Toy Examples using $k(x, y) = \exp\left(-\frac{\|x-y\|^2}{c}\right)$

---



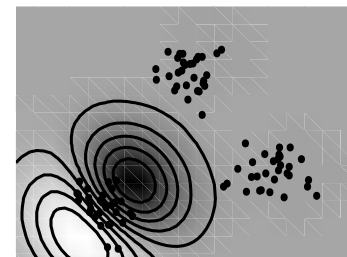
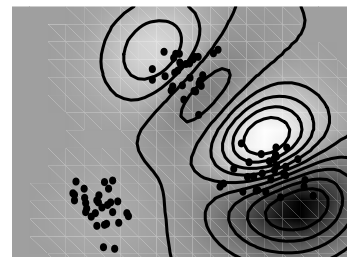
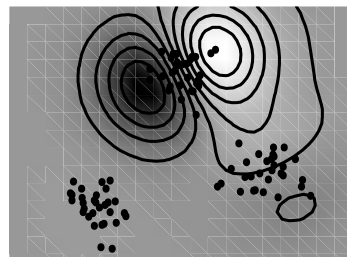
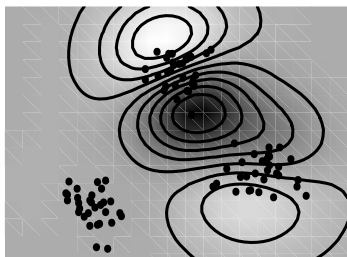
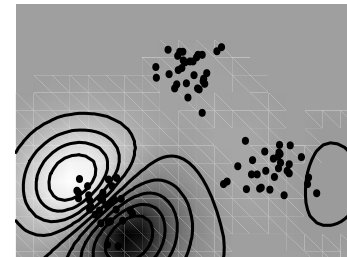
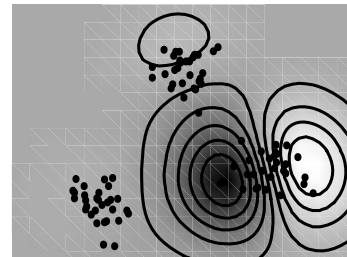
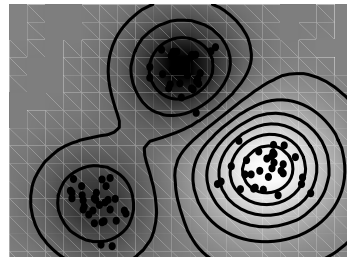
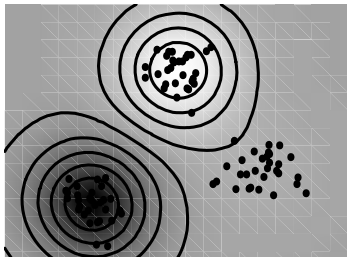
$\nu$ , width $c$	0.5, 0.5	0.5, 0.5	0.1, 0.5	0.5, 0.1
SVs/OLs	0.54, 0.43	0.59, 0.47	0.24, 0.03	0.65, 0.38



# Toy Example with Gaussian Kernel

---

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



# The Challenge: Designing Kernels

---

- transformation invariances (cf. poster of Olivier Chapelle)
- kernels for discrete objects [23, 56, 32, 3]
- kernels based on generative models: Fisher kernel [27]
- local kernels [e.g., 63]
- other sophisticated kernels: e.g., [5, 16, 42]

In general, 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.

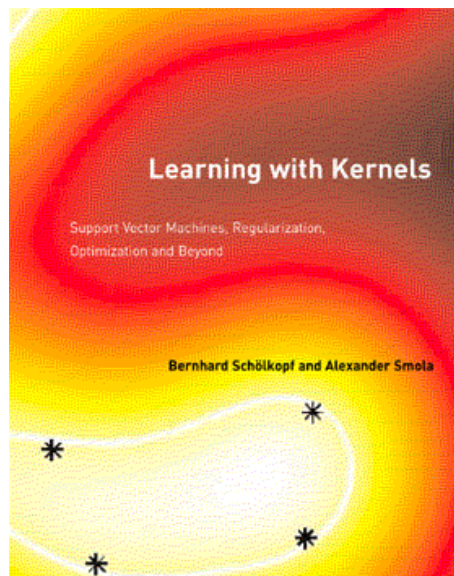
There is ‘no free lunch’ in kernel choice.



# 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, SV pattern recognition, SV quantile estimation, kernel PCA,...)
- not only do these algorithms lend themselves well to theoretical study — they also perform well in practice



For further information, cf.

<http://www.kernel-machines.org>,

<http://www.learning-with-kernels.org>,

and [9, 16, 25, 42].

---

## References

- [1] N. Alon, S. Ben-David, N. Cesa-Bianchi, and D. Haussler. Scale-sensitive dimensions, uniform convergence, and learnability. *Journal of the ACM*, 44(4):615–631, 1997.
- [2] N. Aronszajn. Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68:337–404, 1950.
- [3] P. L. Bartlett and B. Schölkopf. Some kernels for structured data. Technical report, Biowulf Technologies, 2001.
- [4] K. P. Bennett and O. L. Mangasarian. Robust linear programming discrimination of two linearly inseparable sets. *Optimization Methods and Software*, 1:23–34, 1992.
- [5] C. Berg, J. P. R. Christensen, and P. Ressel. *Harmonic Analysis on Semigroups*. Springer-Verlag, New York, 1984.
- [6] D. P. Bertsekas. *Nonlinear Programming*. Athena Scientific, Belmont, MA, 1995.
- [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. Geometry and invariance in kernel based methods. In B. Schölkopf, C. J. C. Burges, and A. J. Smola, editors, *Advances in Kernel Methods — Support Vector Learning*, pages 89–116, Cambridge, MA, 1999. MIT Press.
- [11] 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.

- [12] O. Chapelle, P. Haffner, and V. Vapnik. SVMs for histogram-based image classification. *IEEE Transactions on Neural Networks*, 10(5), 1999.
- [13] O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee. Choosing kernel parameters for support vector machines. *Machine Learning*, 2002. Forthcoming.
- [14] 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.
- [15] C. Cortes and V. Vapnik. Support vector networks. *Machine Learning*, 20:273–297, 1995.
- [16] N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines*. Cambridge University Press, Cambridge, UK, 2000.
- [17] 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.
- [18] H. Drucker, B. Shahraray, and D. C. Gibbon. Relevance feedback using support vector machines. In *Proceedings of the 18th International Conference on Machine Learning*. Morgan Kaufmann, 2001.
- [19] T. Evgeniou, M. Pontil, and T. Poggio. Regularization networks and support vector machines. In A. J. Smola, P. L. Bartlett, B. Schölkopf, and D. Schuurmans, editors, *Advances in Large Margin Classifiers*, pages 171–203, Cambridge, MA, 2000. MIT Press.
- [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] F. Girosi. An equivalence between sparse approximation and support vector machines. *Neural Computation*, 10(6):1455–1480, 1998.
- [22] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, 2002. Forthcoming. Also: Biowulf Technologies TR.
- [23] D. Haussler. Convolutional kernels on discrete structures. Technical Report UCSC-CRL-99-10, Computer Science Department, University of California at Santa Cruz, 1999.
- [24] 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.

- [25] R. Herbrich. *Learning kernel classifiers*. MIT Press, Cambridge, MA, 2002.
- [26] T. S. Jaakkola, M. Diekhans, and D. Haussler. A discriminative framework for detecting remote protein homologies. *Journal of Computational Biology*, 7:95–114, 2000.
- [27] T. S. Jaakkola and D. Haussler. Probabilistic kernel regression models. In *Proceedings of the 1999 Conference on AI and Statistics*, 1999.
- [28] 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.
- [29] 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.
- [30] G. S. Kimeldorf and G. Wahba. Some results on Tchebycheffian spline functions. *Journal of Mathematical Analysis and Applications*, 33:82–95, 1971.
- [31] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86:2278–2324, 1998.
- [32] 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.
- [33] 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.
- [34] 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.
- [35] E. Osuna, R. Freund, and F. Girosi. Support vector machines: Training and applications. Technical Report AIM-1602, MIT A.I. Lab., 1996.
- [36] 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.
- [37] 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.

- [38] S. Saitoh. *Theory of Reproducing Kernels and its Applications*. Longman Scientific & Technical, Harlow, England, 1988.
- [39] B. Schölkopf, R. Herbrich, A. J. Smola, and R. C. Williamson. A generalized representer theorem. Technical Report 2000-81, NeuroCOLT, 2000. Published in *Proceedings COLT'2001*, Springer Lecture Notes in Artificial Intelligence, 2001.
- [40] B. Schölkopf, S. Mika, C. Burges, P. Knirsch, K.-R. Müller, G. Rätsch, and A. J. Smola. Input space vs. feature space in kernel-based methods. *IEEE Transactions on Neural Networks*, 10(5):1000–1017, 1999.
- [41] B. Schölkopf, J. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson. Estimating the support of a high-dimensional distribution. *Neural Computation*, 13:1443–1471, 2001.
- [42] B. Schölkopf and A. J. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.
- [43] B. Schölkopf, A. J. Smola, and K.-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10:1299–1319, 1998.
- [44] J. Shawe-Taylor, P. L. Bartlett, R. C. Williamson, and M. Anthony. Structural risk minimization over data-dependent hierarchies. *IEEE Transactions on Information Theory*, 44(5):1926–1940, 1998.
- [45] 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.
- [46] A. J. Smola, B. Schölkopf, and K.-R. Müller. The connection between regularization operators and support vector kernels. *Neural Networks*, 11:637–649, 1998.
- [47] 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.
- [48] V. Vapnik. *Estimation of Dependences Based on Empirical Data [in Russian]*. Nauka, Moscow, 1979. (English translation: Springer Verlag, New York, 1982).
- [49] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer, NY, 1995.
- [50] V. Vapnik. *Statistical Learning Theory*. Wiley, NY, 1998.
- [51] V. Vapnik and A. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications*, 16(2):264–280, 1971.

- [52] V. Vapnik and A. Chervonenkis. *Theory of Pattern Recognition [in Russian]*. Nauka, Moscow, 1974. (German Translation: W. Wapnik & A. Tscherwonenkis, *Theorie der Zeichenerkennung*, Akademie-Verlag, Berlin, 1979).
- [53] V. Vapnik and A. Lerner. Pattern recognition using generalized portrait method. *Automation and Remote Control*, 24:774–780, 1963.
- [54] G. Wahba. *Spline Models for Observational Data*, volume 59 of *CBMS-NSF Regional Conference Series in Applied Mathematics*. SIAM, Philadelphia, 1990.
- [55] 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.
- [56] 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.
- [57] H. L. Weinert. *Reproducing Kernel Hilbert Spaces*. Hutchinson Ross, Stroudsburg, PA, 1982.
- [58] 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.
- [59] C. K. I. Williams and M. Seeger. The effect of the input density distribution on kernel-based classifiers. In P. Langley, editor, *Proceedings of the 17th International Conference on Machine Learning*, pages 1159–1166, San Francisco, California, 2000. Morgan Kaufmann.
- [60] R. C. Williamson, J. Shawe-Taylor, B. Schölkopf, and A. J. Smola. Sample-based generalization bounds. *IEEE Transactions on Information Theory*, 1999. Submitted. Also: NeuroCOLT Technical Report NC-TR-99-055.
- [61] R. C. Williamson, A. J. Smola, and B. Schölkopf. Generalization performance of regularization networks and support vector machines via entropy numbers of compact operators. Technical Report 19, NeuroCOLT, <http://www.neurocolt.com>, 1998. Accepted for publication in *IEEE Transactions on Information Theory*.
- [62] 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.
- [63] 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.