Support Vector Machines

Michael Reiter
Vienna University of Technology

July 7, 2009

"There is nothing so practical as a good theory" (K. Lewin)

SVM Overview

- follows a principled approach rooted in statistical learning theory: **structural risk minimization**.
- good generalization performance
- without incorporating problem-domain knowledge.
- use of kernels allows to construct RBF-NN, 2-Layer-MLP

Empirical risk minimization

Many training algorithms rely on the (known) empirical risk (given a training set)

$$E_{emp}(\mathbf{w}) = \frac{1}{N}\sum_{i=1}^{N}(\hat{f}_{\mathbf{w}}(\mathbf{x}_i) - t_i)^2. \tag{1}$$

as an estimate for the true (unknown) expected risk

$$E(\mathbf{w}) = \int (\hat{f}_{\mathbf{w}}(\mathbf{x}) - f(\mathbf{x}))^2 p(\mathbf{x}) d\mathbf{x} = \int (\hat{f}_{\mathbf{w}}(\mathbf{x}) - y)^2 p(\mathbf{x}, y) d\mathbf{x} d\mathbf{y}. \tag{2}$$

and chose $\mathbf{w}$ to minimize $E_{emp}(\mathbf{w})$. Note that generally $p(\mathbf{x}, y)$ is unknown.

This approach is known as principle of **empirical risk minimization** (ERM). We assume that $E_{emp}(\mathbf{w})$ is close to $E(\mathbf{w})$.
VC theory (also referred to as **statistical learning theory**) provides principles for learning based on bounds on the difference of $E_{emp}(\mathbf{w})$ and $E(\mathbf{w})$. These bounds explicitly take into account **model complexity** and the **size of the training set**.

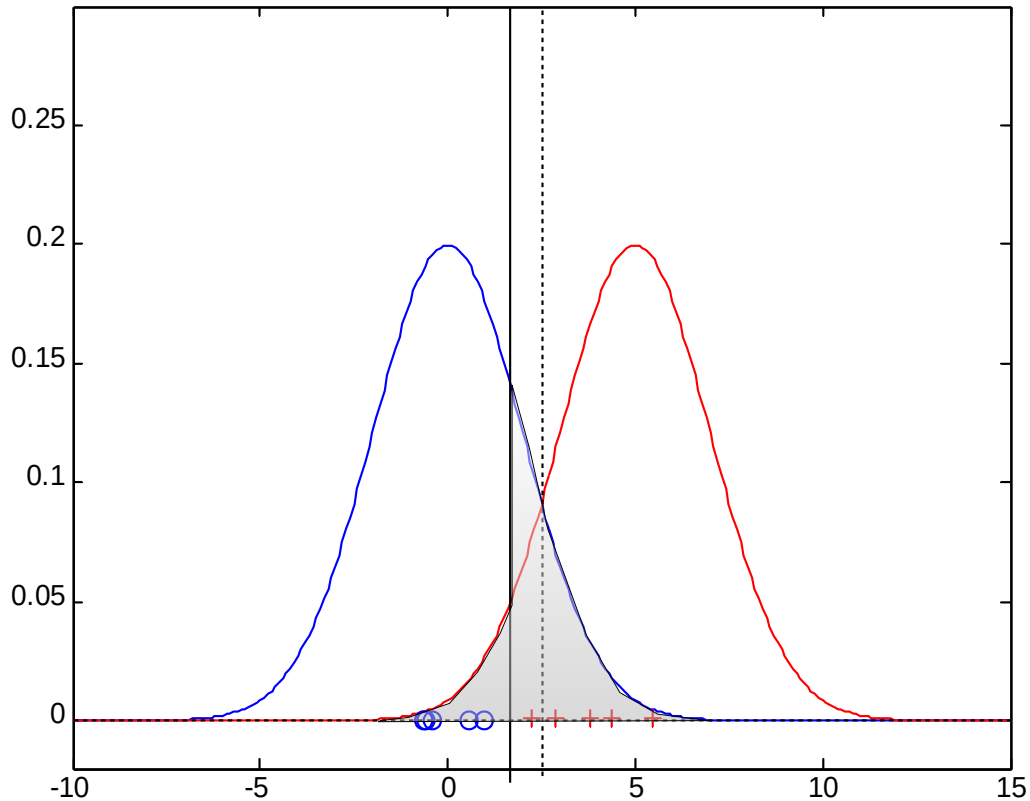


Figure: Binary classification example. The training error here is the ratio of missclassified data points and overall number of points. The theoretical error is determined by unknown class-conditional probabilty density functions.

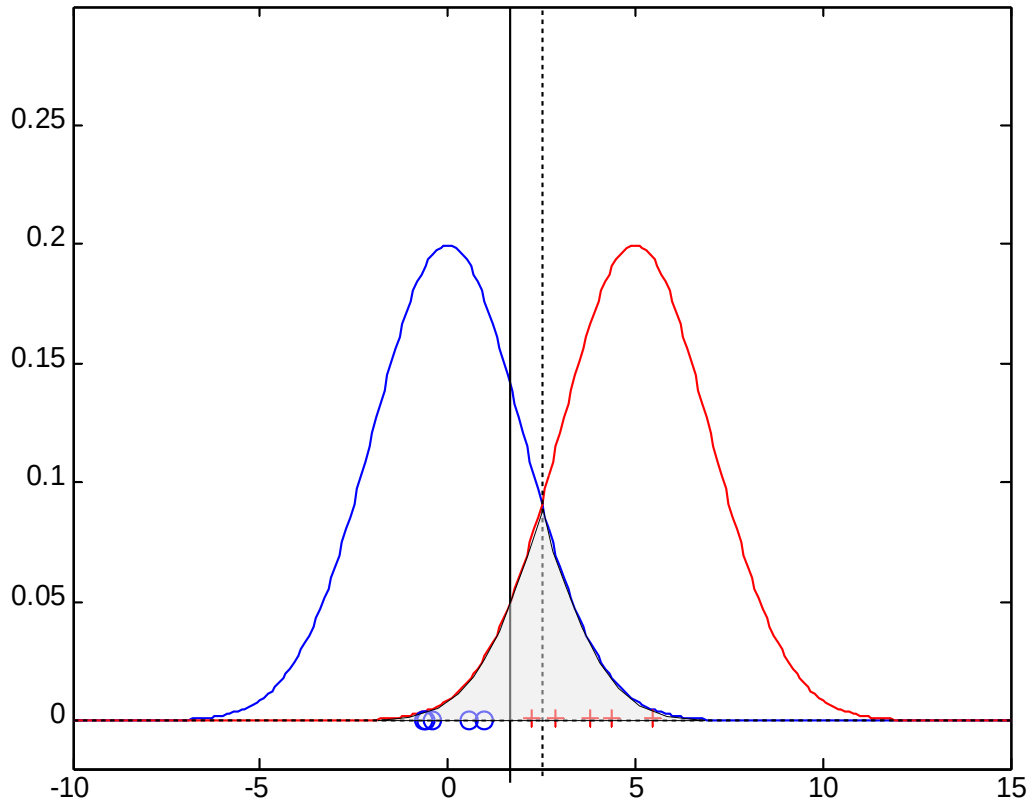


Figure: Binary classification example. The training error here is the ratio of missclassified data points and overall number of points. The theoretical error is determined by unknown class-conditional probabilty density functions.

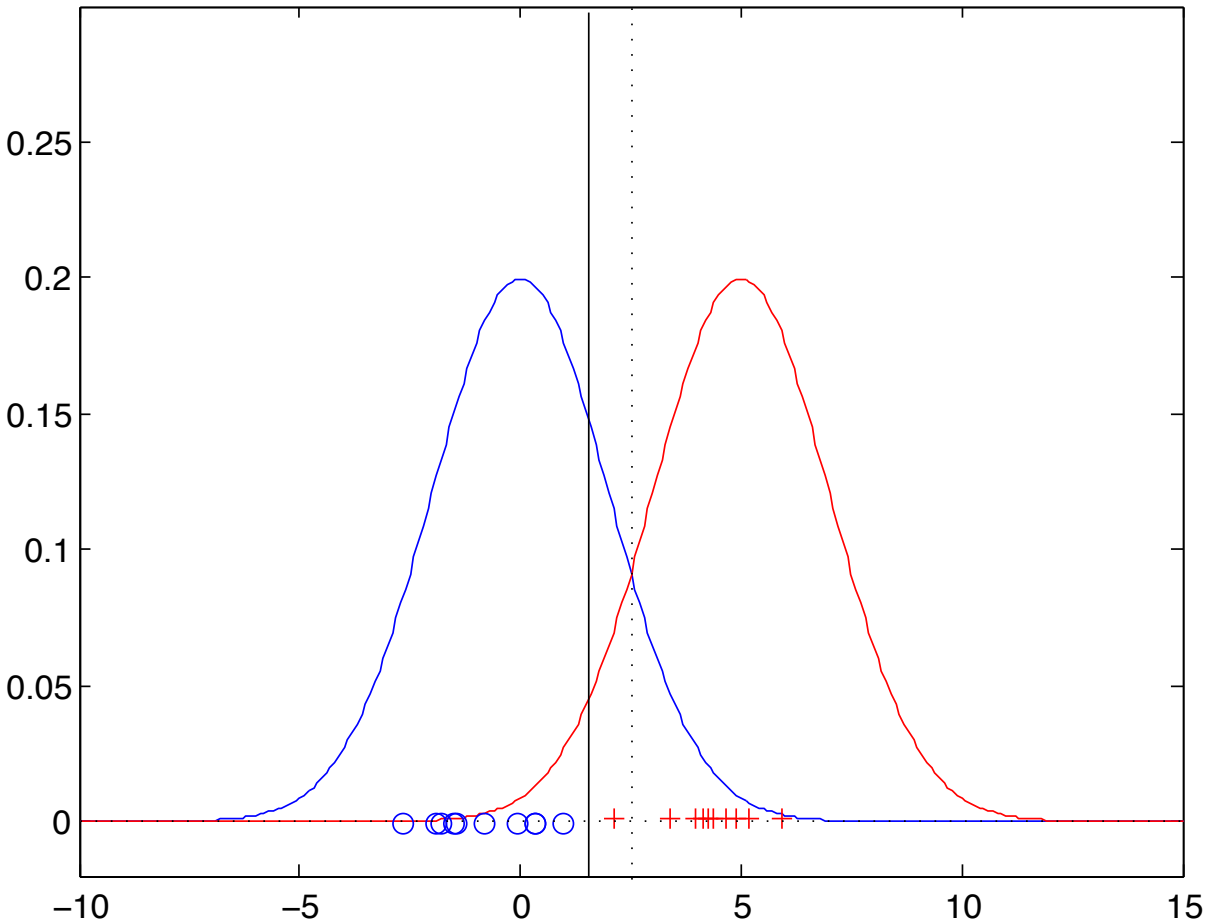


Figure: Increasing number of training points N. The solid line depicts the decision boundary $\mathbf{w}^*$ minimizing the empirical risk (training error). The dotted line depicts the optimal decision boundary minimizing the true risk.

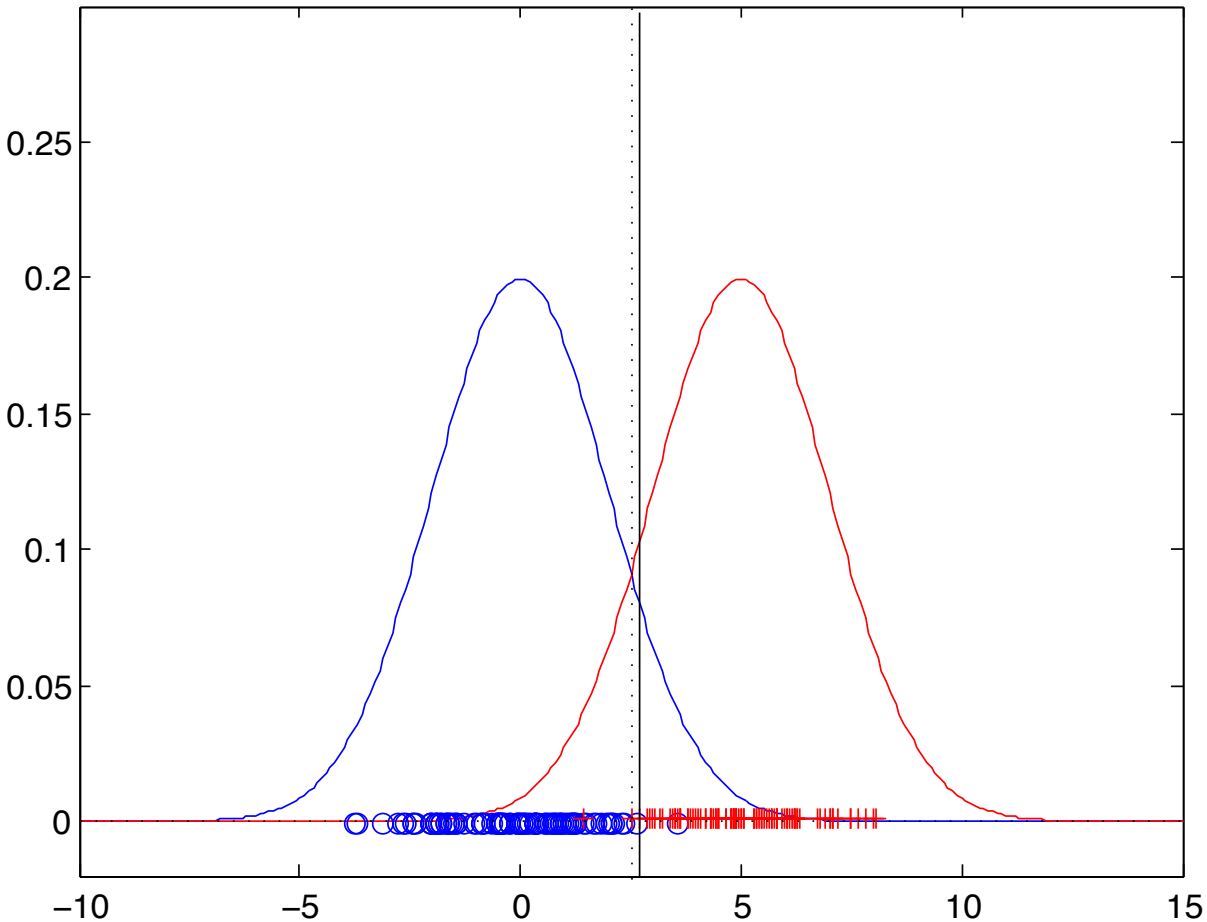


Figure: Increasing number of training points N. The solid line depicts the decision boundary $\mathbf{w}^*$ minimizing the empirical risk (training error). The dotted line depicts the optimal decision boundary minimizing the true risk.

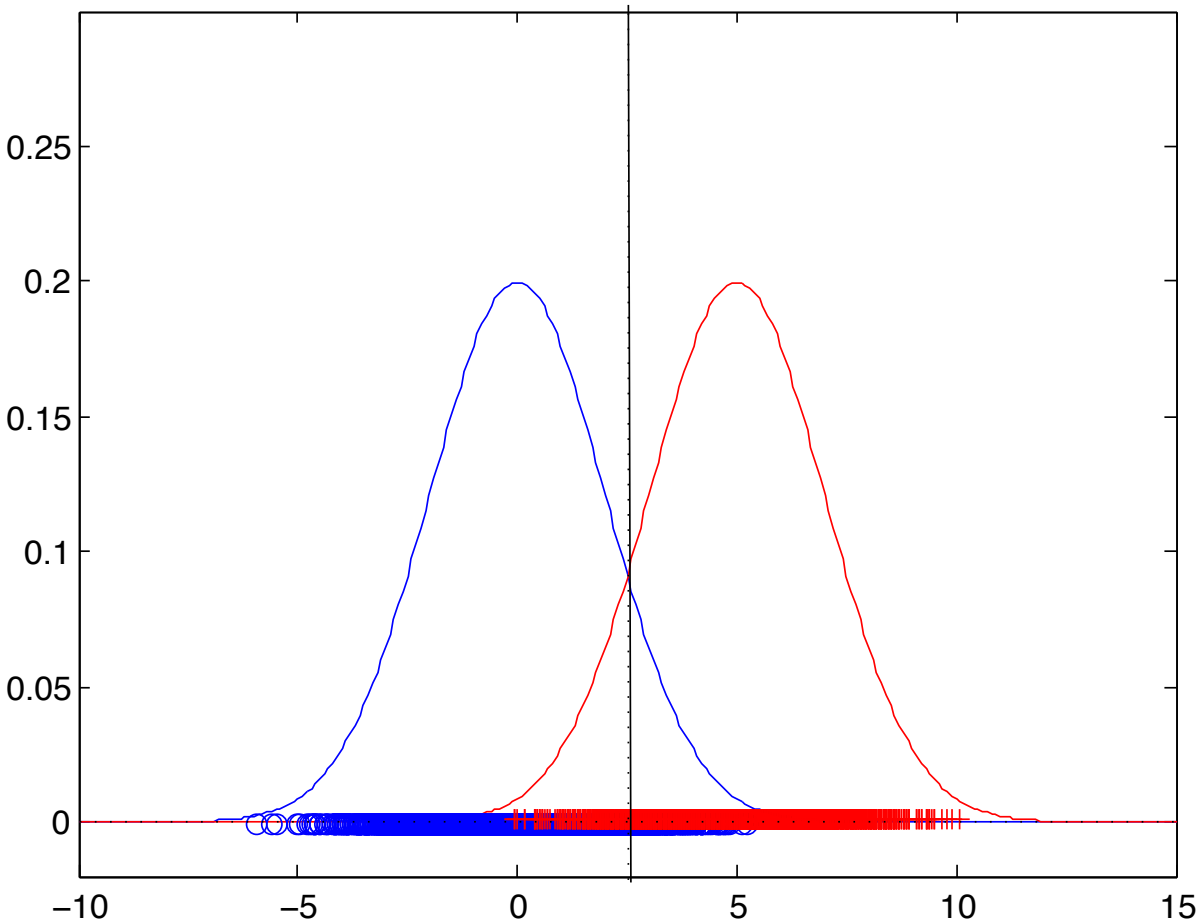


Figure: Increasing number of training points N. The solid line depicts the decision boundary $\mathbf{w}^*$ minimizing the empirical risk (training error). The dotted line depicts the optimal decision boundary minimizing the true risk.

ERM consistency

Let $\mathcal{F} = \{\hat{f}_{\mathbf{w}}() : w \in \mathcal{W}\}$ be the set of functions which can be implemented by the learning machine (e.g. a neural network), and $\mathcal{W}$ is the set of valid parameter vectors.
Iff $E_{emp}(\mathbf{w})$ approximates $E(\mathbf{w})$ *uniformly* in $\mathbf{w} \in \mathcal{W}$, i.e.,

$$\lim_{N \to \infty} P\Big(\sup_{\mathbf{w} \in \mathcal{W}} |E(\mathbf{w}) - E_{emp}(\mathbf{w})| > \epsilon \Big) = 0, \tag{3}$$

the ERM principle is said to be **consistent** (Vapnik, 1982).

- Note that the consistency is determined by the **"worst-case w"** $\in \mathcal{W}$.
- Worst-case analysis is necessary for deriving bounds for the true risk $E(\mathbf{w})$ that are **independent of the distribution** of the data (density p).

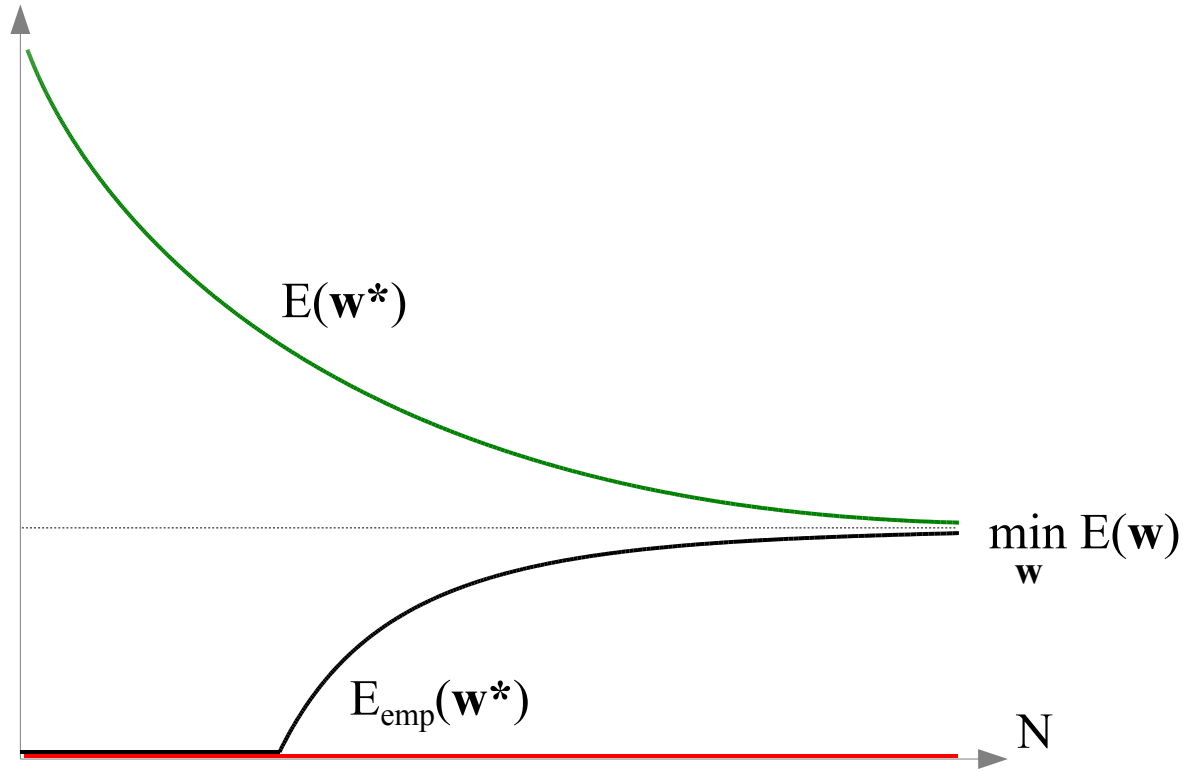


Figure: In the case of consistency, proximity between $E(\mathbf{w}^*)$ and $E_{emp}(\mathbf{w}^*)$, where $\mathbf{w}^* = \arg\min E_{emp}(\mathbf{w})$ is guaranteed.

VC-Dimension

The **Vapnik-Chervonenkis Dimension** (VC-dimension) is a measure for the capacity (expressive power) of a set of functions $\mathcal{F}$. The VC-dimension is defined as the size of the largest set of data samples which can be **shattered** by $\mathcal{F}$.
A set of data samples $\mathcal{S}$ is said to be **shattered** by $\mathcal{F}$ if it can be devided in all possible $2^{|\mathcal{S}|}$ ways by functions $\hat{f}_w \in \mathcal{F}$.

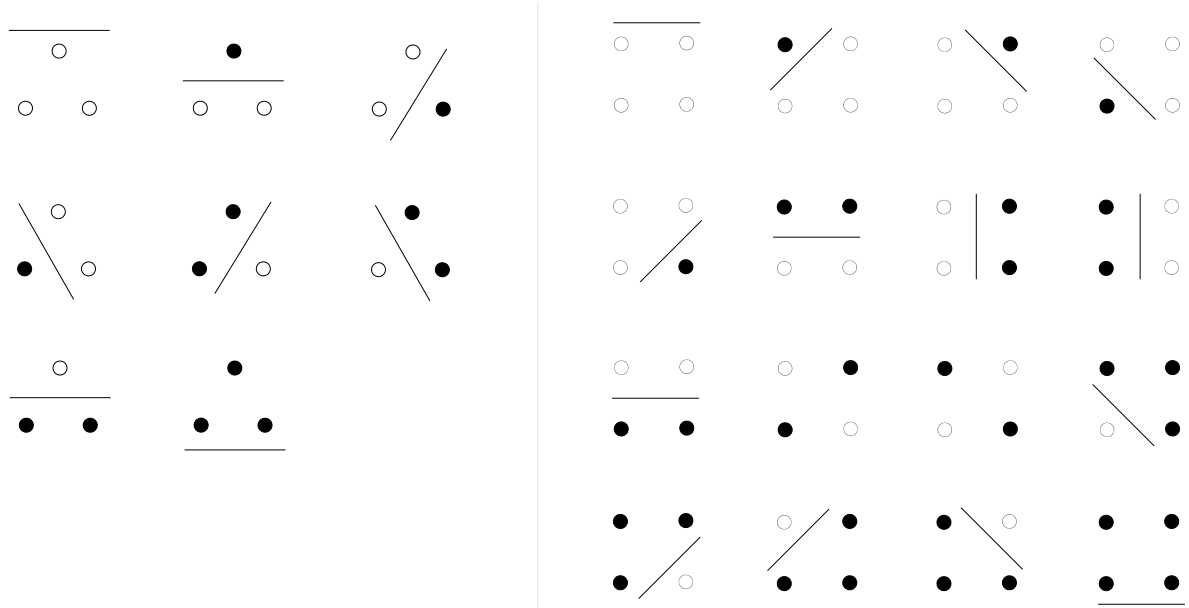


Figure: Three 2d-points in general position can be shattered by the set of linear functions. The definition of VC-dimension **requires only one such set of points to exist**. Because there exists no such set with 4 points, the VC-dimension for bevariate linear functions is 3. In higher dimensions, the VC-dimension of linear functions with d input variables is $d + 1$.

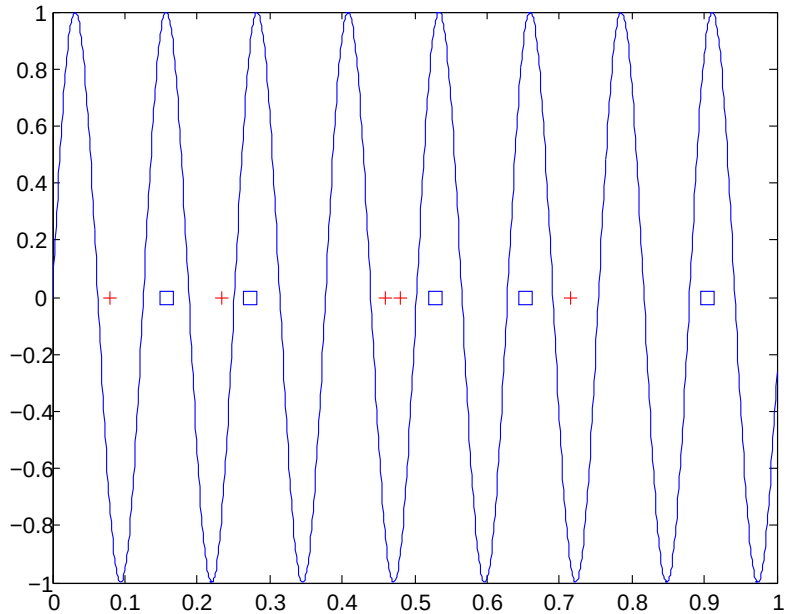


Figure: Example: Although the function $f(x) = I(sin(w * x) > 0)$, where $I()$ is the indicator function, has only one adjustable parameter w, the VC-dimension is infinite.

For the empirical risk minimization principle to be consistent, the VC-dimension d has to be finite (e.g., a set of functions with infinite VC-dimension will always have $E_{emp}(w) = 0$, regardless of the size N of the training set).

Bounds on the Classification Error

It can be shown (Vapnik) that the following bound holds with probability of at least $1 - \eta$ (for classification problems):

$$E(\mathbf{w}) \leq E_{emp}(\mathbf{w}) + \sqrt{\frac{h(\ln \frac{2}{h} + 1) - \ln \frac{\eta}{4}}{N}}, \tag{4}$$

for all $\mathbf{w} \in \mathcal{W}$ (worst-case analysis) where h is the VC-dimension of the set of functions $\mathcal{F}$ implemented by the parameterized model $\hat{f}_{\mathbf{w}}()$.
It follows from Eq. 4 that the ERM approach is only justified when the ratio of N/h is large (e.g. ≥ 20), otherwise it cannot be guaranteed, that the overall risk is "close" to the empirical risk. In the case of low N/h the size of $\mathcal{W}$ (i.e., the flexibility of the model) has to be adjusted to the available data (*occam's razor* principle).

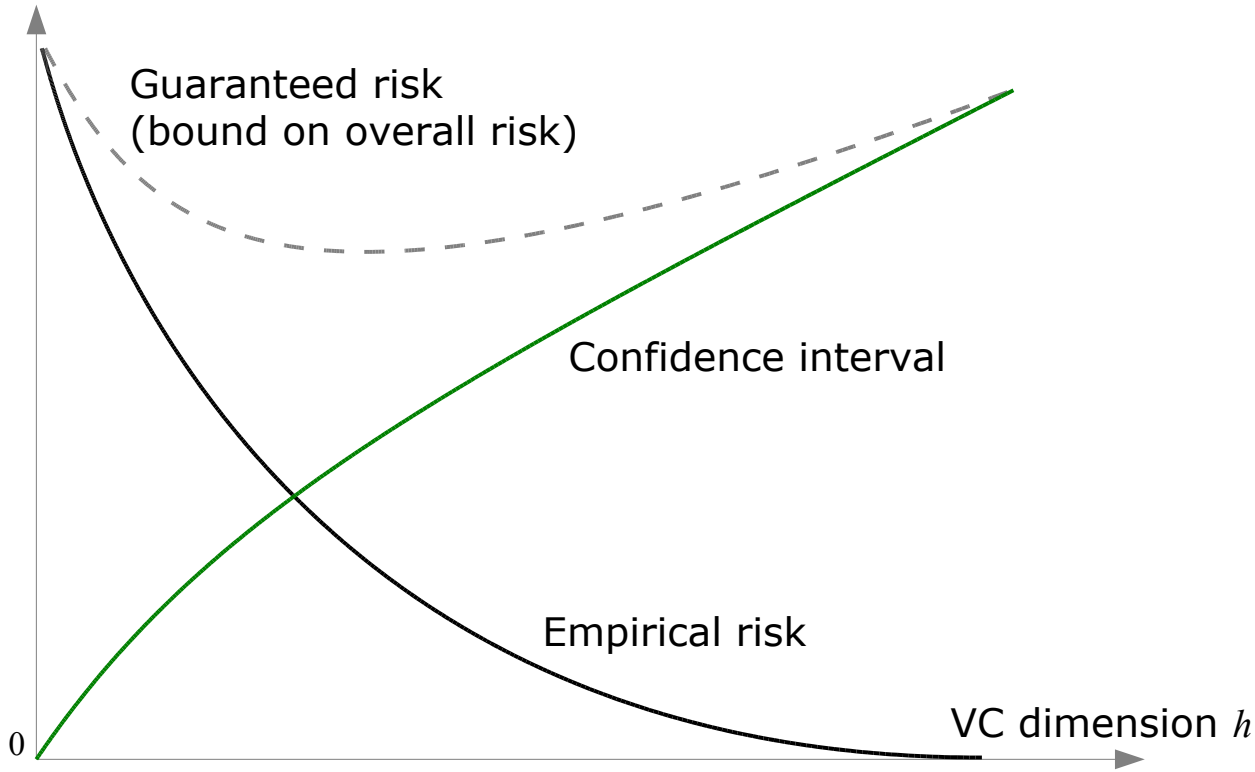
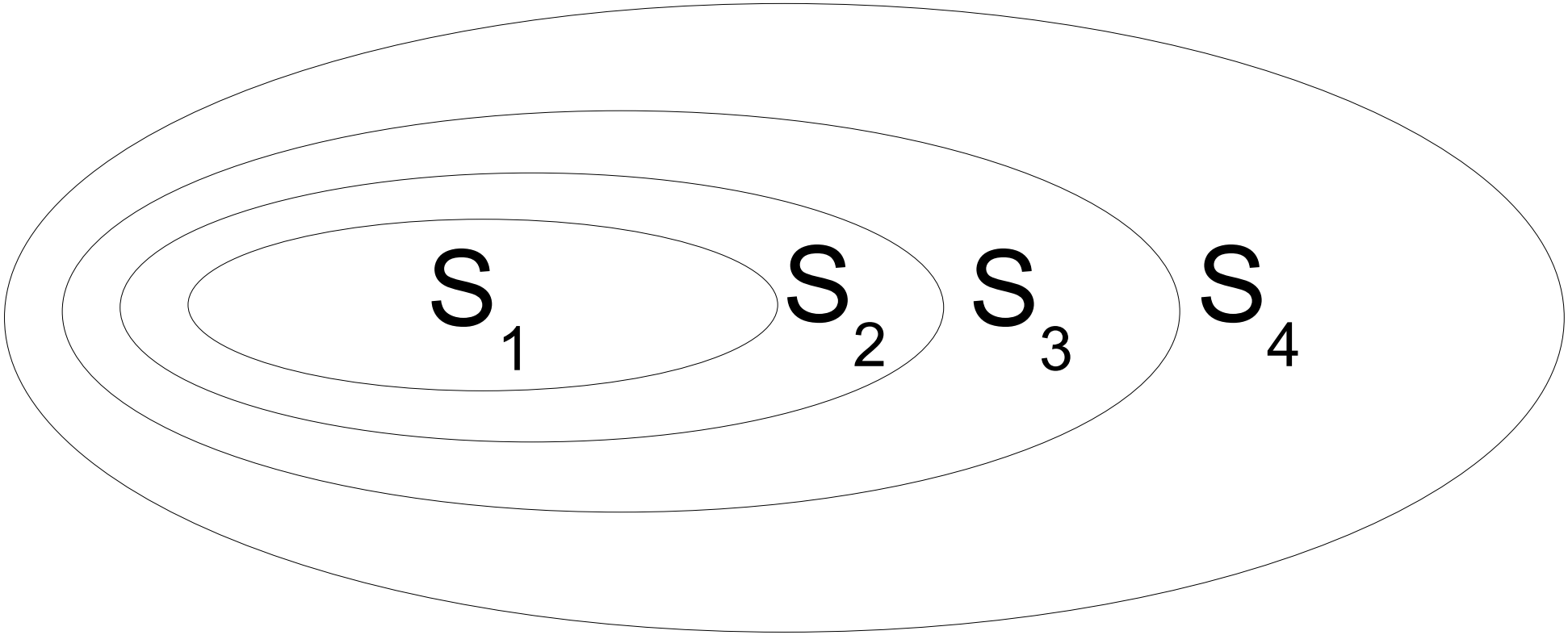


Figure: The confidence interval given in Eq. 4 shown graphically in dependence of the VC-dimension h.

Stuctural Risk Minimization (SRM) Principle



$$S_1 \subseteq S_2 \subseteq S_3 \subseteq S_4$$

$$h(S_1) < h(S_2) < h(S_3) < h(S_4)$$

Find the optimal subset (i.e., model complexity) such that the **risk bound** is minimized.

linear discriminant functions

$$\text{choose } \mathbf{w} \text{ such that } \operatorname{sgn}(\mathbf{w}^T\mathbf{x} + w_0) = \begin{cases} -1 & \text{if } \mathbf{x} \in \text{class } 1(+) \\ +1 & \text{if } \mathbf{x} \in \text{class } 2(*) \end{cases}$$

Given a training set of N data points with corresponding target values $(\mathbf{x}_i, t_i), i = 1, \ldots, N$ we aim to find $\mathbf{w}$ such that

$$t_i = \operatorname{sgn}(\mathbf{w}^T\mathbf{x}_i + w_0)$$

This is possible if the set is **linearly separable**.

LD: $t_i = \text{sgn}(\mathbf{w}^T\mathbf{x}_i + w_0)$

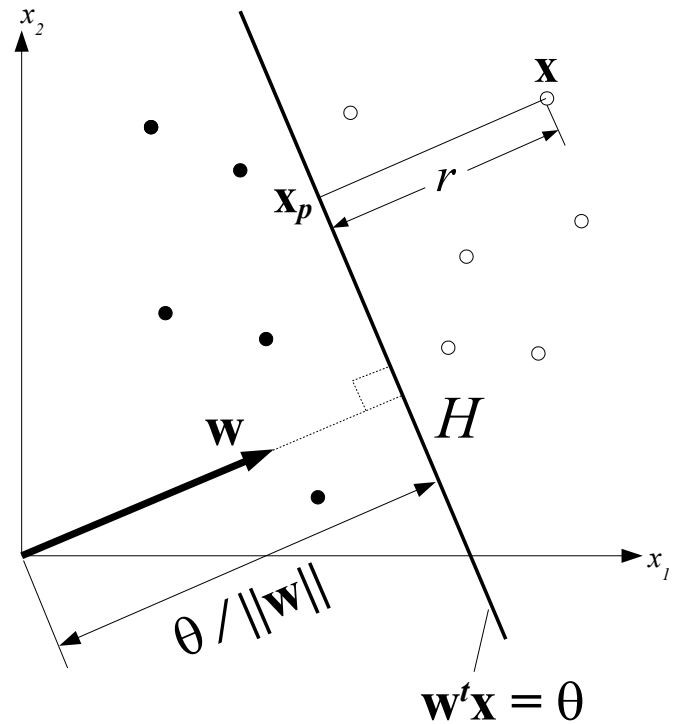


Figure: $r = \frac{\mathbf{w}^t\mathbf{x}+w_0}{\|\mathbf{w}\|}$ gives an algebraic measure of the distance of a point $\mathbf{x}$ from H. r is positive if $\mathbf{w}^t\mathbf{x} + w_0 > 0$, 0 if the point lies on the hyperplane and negative otherwise. If $r > 0$, $\mathbf{x}$ is said to lie on the *positive side* of H. In particular, the normal distance of the origin to the hyperplane is given by $r_0 = \frac{w_0}{\|\mathbf{w}\|}$

.

XOR-Problem

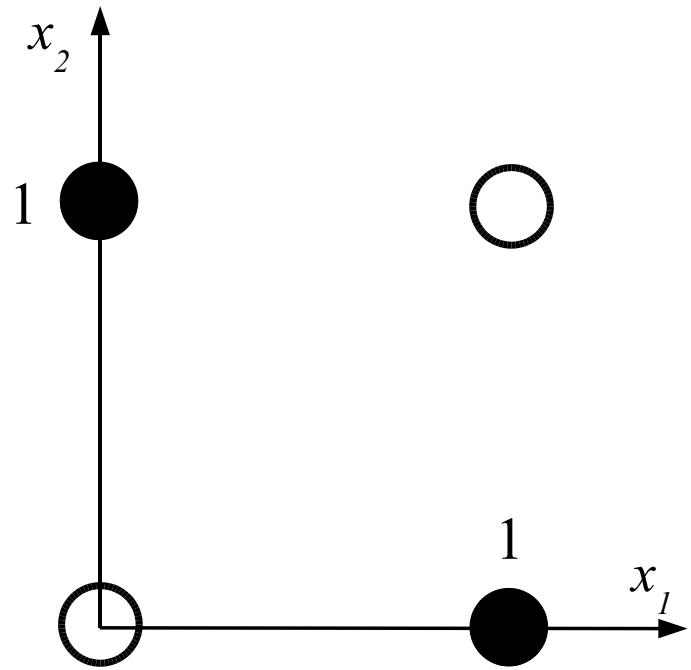
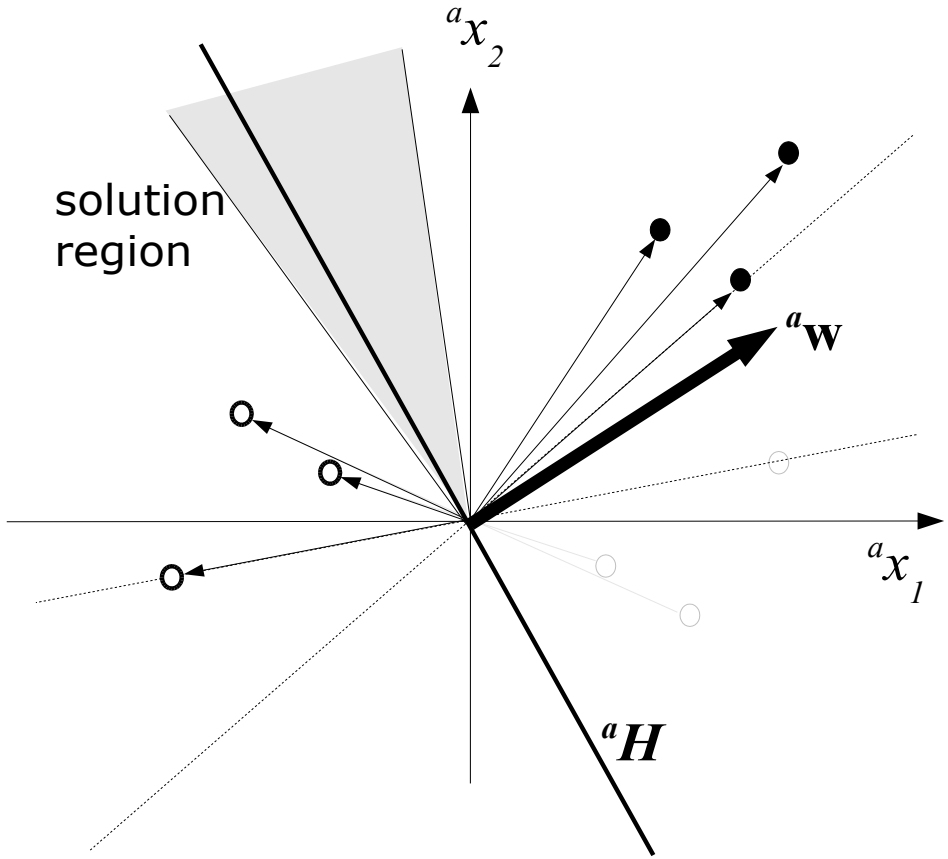


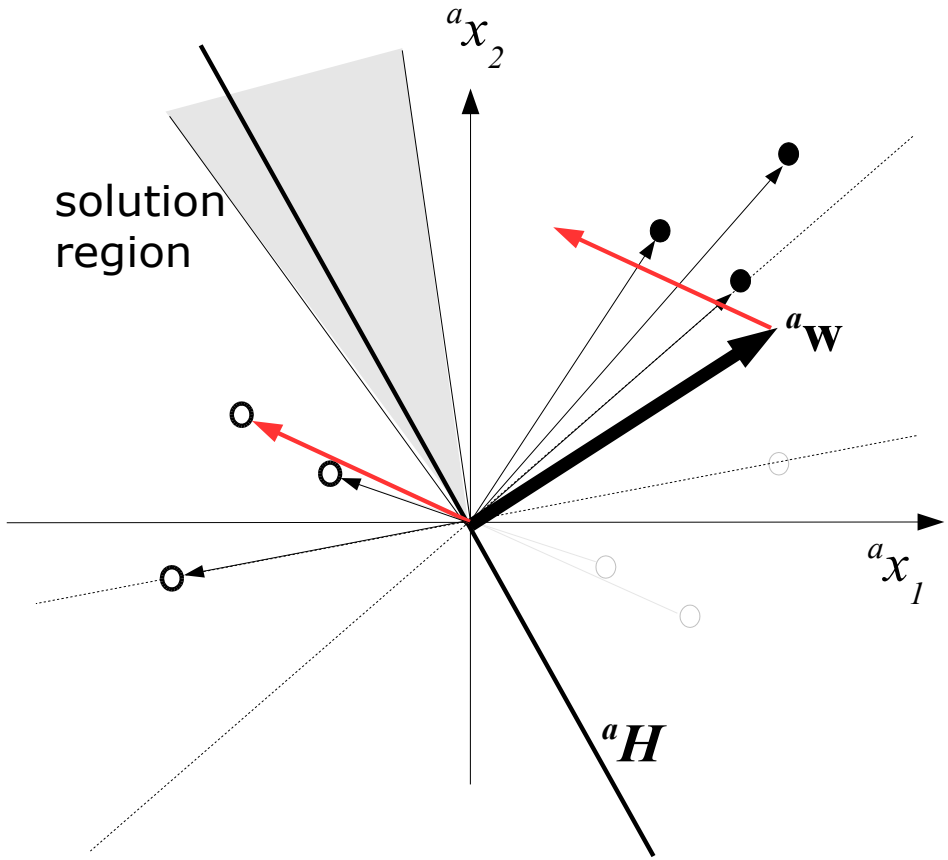
Figure: The exclusive-OR problem is a simple example with 2-d binary input vectors which is not linearly saparable.

The perceptron



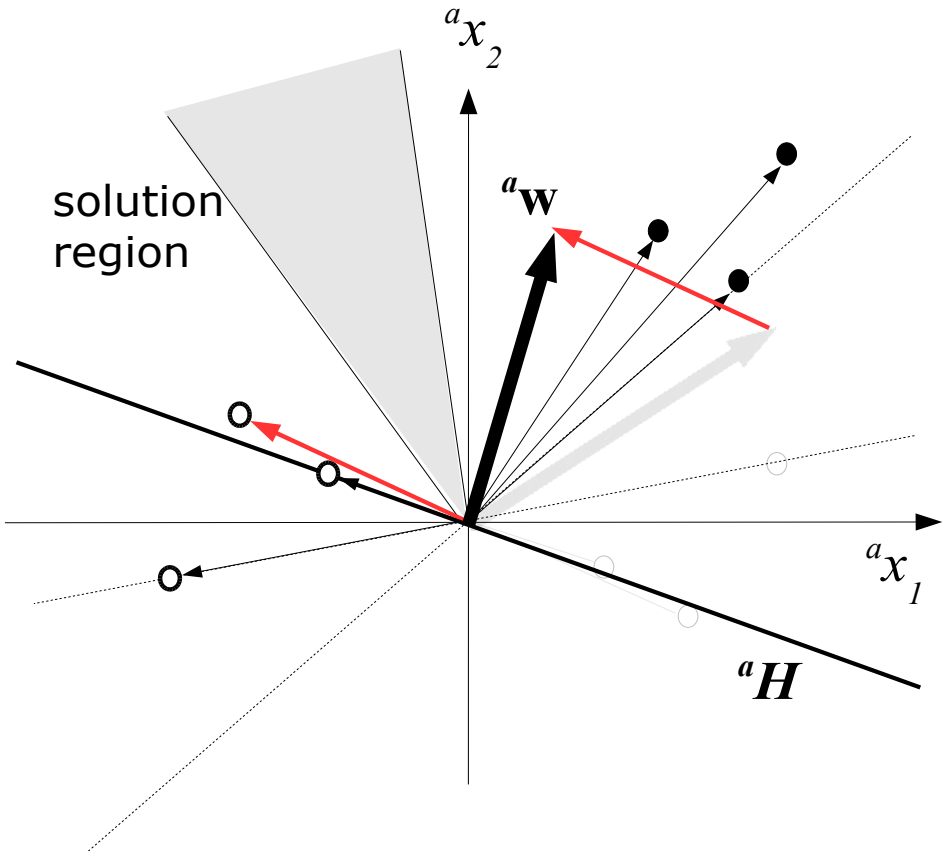
$$\mathbf{w} \leftarrow \mathbf{w} + \alpha t_i \mathbf{x}_i$$

The perceptron



$$\mathbf{w} \leftarrow \mathbf{w} + \alpha t_i \mathbf{x}_i$$

The perceptron



$$\mathbf{w} \leftarrow \mathbf{w} + \alpha t_i \mathbf{x}_i$$

Let $\mathcal{S}_{Tr} = \{(\mathbf{x}_1, t_1), \ldots, (\mathbf{x}_N, t_N)\}$ denote a linearly separable set of N (non-augmented) training vectors $\mathbf{x}_i \in \mathbf{R}^d$ and corresponding class labels $t_i \in \{1, -1\}$.
We have used the perceptron training algorithm to find weight vector $\mathbf{w}$ and bias w_0 of a decision function

$$d(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + w_0, \tag{5}$$

i.e. a separating hyperplane satisfying

$$d(\mathbf{x}_i)t_i > 0, \quad 1 \leq i \leq N. \tag{6}$$

For a linearly separable set of training data, there exists an infinite number of separating hyperplanes (cf. figure 9) represented by Eq. 6.

We are now looking for a solution $\mathbf{w}^*, w_0$ with maximum geometric margin satifying Eq. 6.
Recall that the geometric margin of a hyperplane equals the euclidean distance of the pattern that lies closest to the hyperplane. A separating hyperplane with maximal geometrical margin is called **optimal** or **maximal margin** hyperplane.
To introduce a margin τ we modify Eq. 6

$$\frac{d(\mathbf{x}_i)t_i}{\|\mathbf{w}\|} \geq \tau, \quad 1 \leq i \leq N. \tag{7}$$

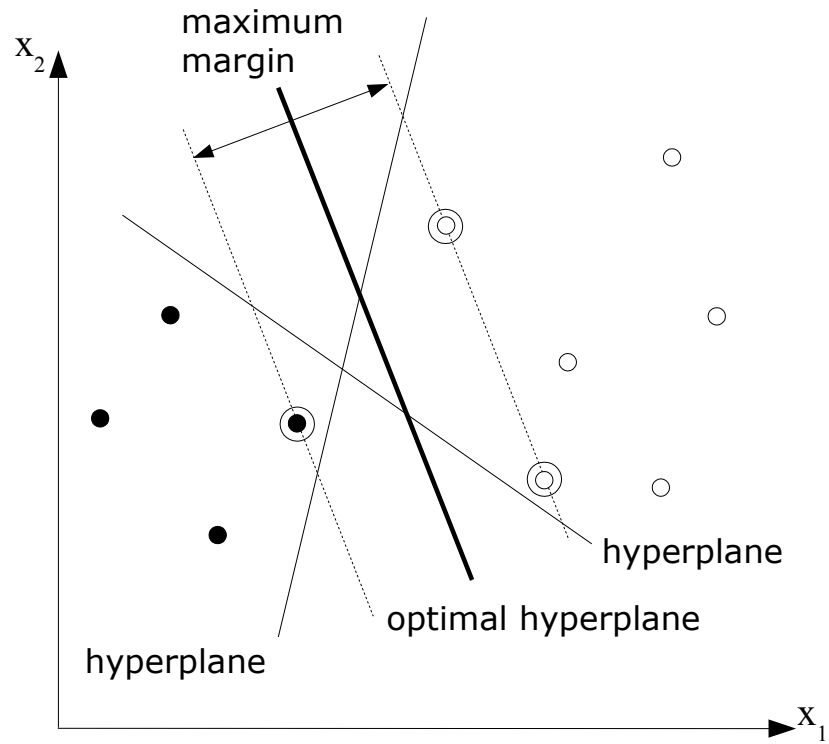


Figure: Among all separating hyperplanes the optimal hyperplane is defined by the subset of input vectors (support vectors) with smallest euclidean distance to the hyperplane.

Recall that $\mathbf{w}$ and w_0 can be scaled by the same positive factor $\alpha \in \mathbb{R}^+$ leaving the decision boundary unchanged, thus there exists an infinite number of solutions $\alpha\mathbf{w}^*, \alpha w_0^*$ corresponding to the same decision boundary.
To obtain a unique solution we choose

$$\tau = \frac{1}{\|\mathbf{w}\|}. \tag{8}$$

so that Eq. 7 can be written as

$$d(\mathbf{x}_i)t_i \geq \tau\|\mathbf{w}\| = 1, \quad 1 \leq i \leq N. \tag{9}$$

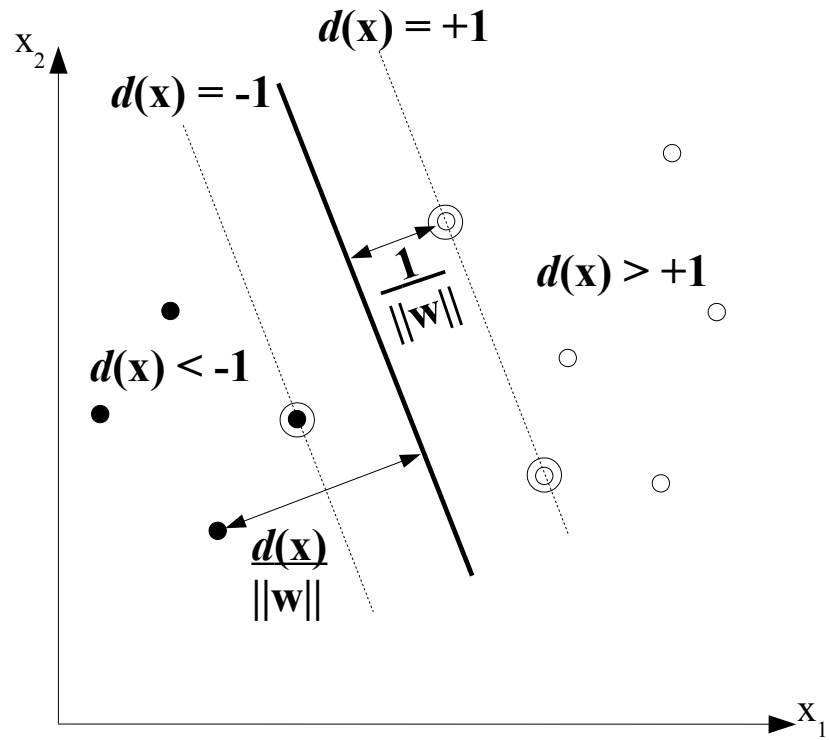


Figure: The decision boundary is defined by $d(\mathbf{x}) = 0$. The distance between a support vector and the decision boundary is $\frac{1}{\|\mathbf{w}\|}$, which defines the geometric margin τ of the optimal hyperplane.

It follows from Eq. 8 that maximizing the margin of $\mathbf{w}$ (given by $\tau = \frac{1}{\|\mathbf{w}\|}$) is equivalent to minimizing the norm of $\mathbf{w}$.
Thus an optimal hyperplane is one which satisfies conditions given by Eq. 9 and additionally minimizes the **(quadratic) criterion function**

$$J(\mathbf{w}) = \frac{1}{2}\|\mathbf{w}\|^2 \tag{10}$$

Dual representation

Recall that without loss of generality for perceptron training we have initialized $\mathbf{w} = \mathbf{0}$ so that the solution vector is a linear combination of training data $\mathbf{x}_i t_i$ misclassified during training and can be written as

$$\mathbf{w} = \sum_{i=1}^{N} \alpha_i \mathbf{x}_i t_i, \quad \alpha_i \in \mathbf{N}_0 \tag{11}$$

i.e., given a fixed training set and using Eq. 11, the decision rule can be written in terms of the α_i (**dual representation**)

$$d(\mathbf{x}) = (\mathbf{w}^T \mathbf{x} + w_0) = \sum_{i=1}^{N} \alpha_i t_i \mathbf{x}_i{}^T \mathbf{x} + w_0, \quad 1 \leq i \leq N. \tag{12}$$

Notice that in this formulation the data is given only in terms of inner products $\mathbf{x}_i^T \mathbf{x}$. This property has important consequences (see below). In particular, it allows to define a separating hyperplane in high dimensional feature space, without having to compute the high dimensional mapping explicitly

Review of Lagrange Multipliers

Consider a constrained optimization problem, i.e. to find the minimum

$$\mathbf{x}^* = \arg\min_{\mathbf{x}} f(\mathbf{x}) \tag{13}$$

subject to the contraints given by k functions

$$h_i(\mathbf{x}) = 0, \quad 1 \leq i \leq m. \tag{14}$$

To simplify the notation we write

$$\mathbf{h}(\mathbf{x}) = (h_1(\mathbf{x}), \ldots, h_m(\mathbf{x}))^T = \mathbf{0}. \tag{15}$$

Example:

$$f(\mathbf{x}) = x_1^2 + x_2^2 \tag{16}$$

$$h(\mathbf{x}) = x_1 + x_2 - 1 = 0 \tag{17}$$

Allthough in this example $h(x_1, x_2) = 0$ could be used to substitute either x_1 or x_2 (e.g. use $x_2 = 1 - x_1$ to give a function $f(\mathbf{x}) = f(x_1) = 2x_1^2 + 2x_1 + 1$), it is not always possible to find a simple analytic solution of the contraint equation (e.g., $h(x_1, x_2) = (x_1 - 1)^2 + (x_2 - 1)^2$).

A more elegant, and often simpler, approach is based in the introduction of new parameters $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_m)^T$ (**Lagrange multipliers**). We formulate the constraint optimization problem using the **Lagrangian function**

$$L(\mathbf{x}, \boldsymbol{\beta}) = f(\mathbf{x}) + \sum_{i=1}^{m} \beta_i h_i(\mathbf{x}) \tag{18}$$

$$= f(\mathbf{x}) + \boldsymbol{\beta}^T \mathbf{h}(\mathbf{x}) \tag{19}$$

Note that since $h_i(\mathbf{x}^*) = 0$ the value of the Lagrangian function at the optimal point is

$$L(\mathbf{x}^*, \boldsymbol{\beta}^*) = f(\mathbf{x}^*) \tag{20}$$

A necessary condition for $\mathbf{x}^*$ to be a minimum of $f(\mathbf{x})$ subject to $h_i(\mathbf{x}) = 0, \quad 1 \leq i \leq m$ is (cf. figures 11-12)

$$\frac{\partial L(\mathbf{x}, \boldsymbol{\beta})}{\partial \mathbf{x}}|_{(\mathbf{x}^*, \beta^*)} = \mathbf{0} \tag{21}$$

$$\frac{\partial L(\mathbf{x}, \boldsymbol{\beta})}{\partial \boldsymbol{\beta}}|_{(\mathbf{x}^*, \beta^*)} = \mathbf{0} \tag{22}$$

These conditions are sufficient if $L(\mathbf{x}, \boldsymbol{\beta}^*)$ is a convex function of $\mathbf{x}$. Note that while Eq. 21 gives a new system of equations, Eq. 22 is equivalent to the constraint equations given by Eq. 14. A geometrical explanation of condition Eq. 21 is given in figure 12.

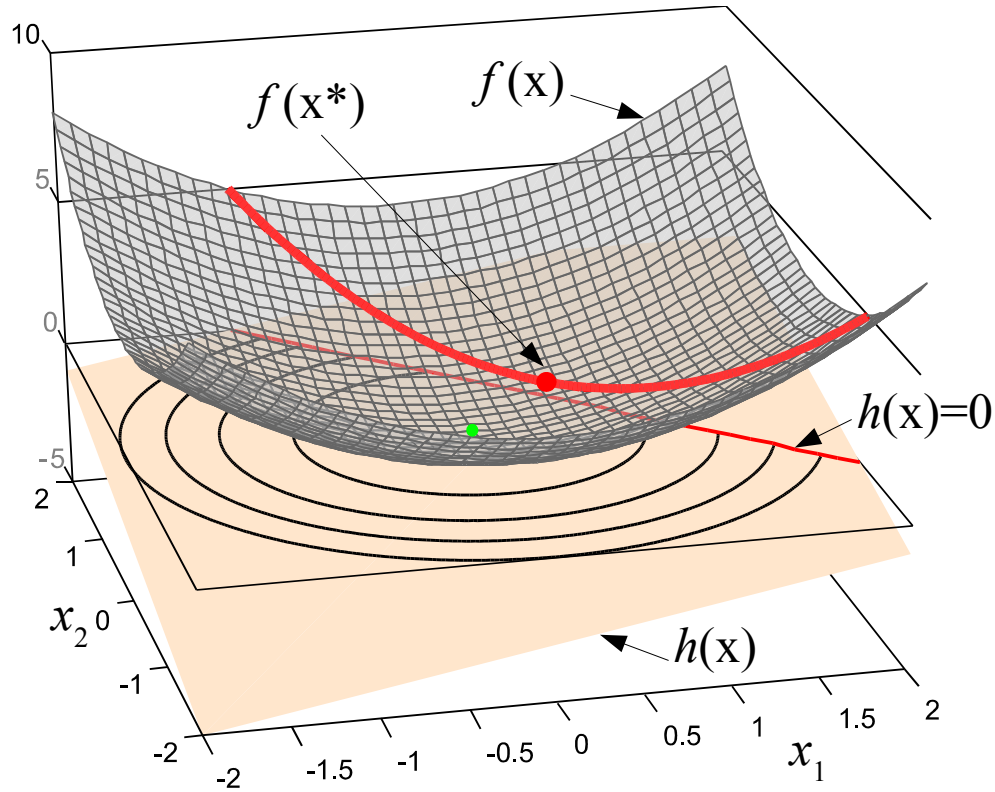


Figure: Geometrical interpretation of the contrained optimization problem given by Eqs. 16 and 17. The point at $(0, 0)$ is the optimum of the unconstrained problem $\arg\min_x f(\mathbf{x})$. The solution x^* of the contrained problem is forced to lie on the line $h(\mathbf{x}) = 0$.

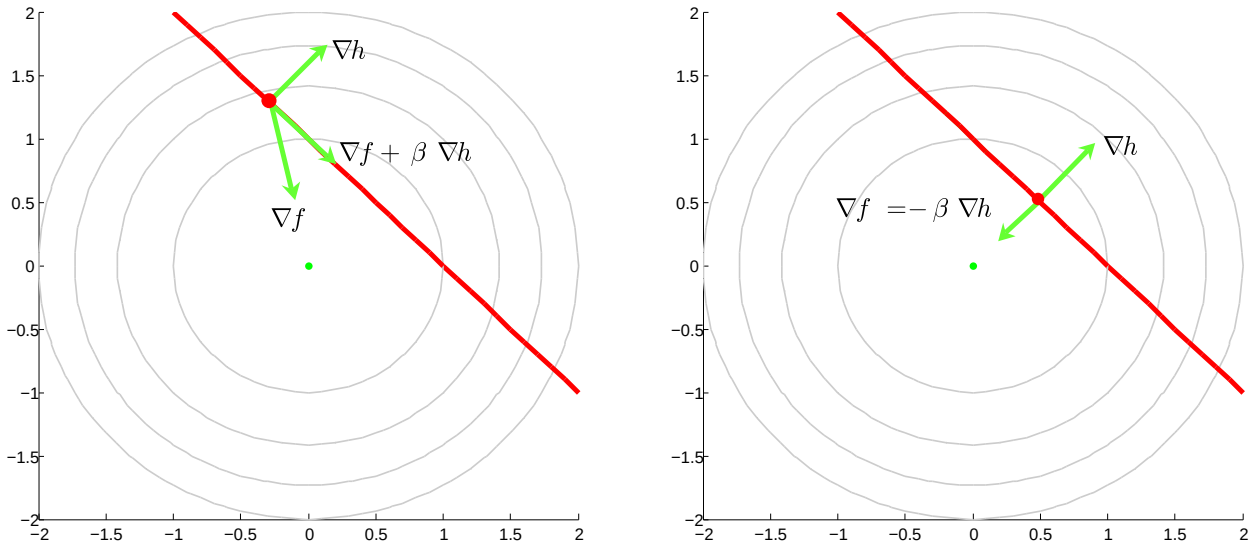


Figure: The gradient of the Lagrangian function is given by ∇f. We wish to find the component of this vector lying within the constraint surface (line) $h(\mathbf{x}) = 0$. This component is given by the gradient of the Lagrangian function $\nabla f(\mathbf{x}) + \beta \nabla h(\mathbf{x})$ and has to be zero at the optimum (condition Eq. 21).

General Case

We now consider a more general case where the optimization problem contains both equality and inequality constraints:

$$\begin{aligned} \text{minimize} \quad & f(\mathbf{x}) && (23) \\ \text{subject to} \quad & g_i(\mathbf{x}) \leq 0, \quad 1 \leq i \leq k. && (24) \\ & h_i(\mathbf{x}) = 0, \quad 1 \leq i \leq m. && (25) \end{aligned}$$

We define the **generalized Lagrangian function** as

$$\begin{aligned} L(\mathbf{x}, \boldsymbol{\alpha}, \boldsymbol{\beta}) &= f(\mathbf{x}) + \sum_{i=1}^{k} \alpha_i g_i(\mathbf{x}) + \sum_{i=1}^{k} \beta_i h_i(\mathbf{x}) && (26) \\ &= f(\mathbf{x}) + \boldsymbol{\alpha}^T \mathbf{g}(\mathbf{x}) + \boldsymbol{\beta}^T \mathbf{h}(\mathbf{x}) && (27) \end{aligned}$$

The region of the domain where $f(\mathbf{x})$ is defined and all constraints are satisfied is called **feasible region**.

If the optimization problem is *convex* and g_i,h_i are *affine* functions (i.e. $\mathbf{g}(\mathbf{x})$ and $\mathbf{h}(\mathbf{x})$ are of the form $\mathbf{Ax} - \mathbf{b}$) the necessary and sufficient conditions for $\mathbf{x}^*$ to be an optimum are the existence of $\boldsymbol{\alpha}^*$,$\boldsymbol{\beta}^*$ such that

$$\frac{\partial L(\mathbf{x}, \boldsymbol{\alpha}, \boldsymbol{\beta})}{\partial \mathbf{x}}|_{(\mathbf{x}^*, \alpha^*, \beta^*)} = \mathbf{0}, \tag{28}$$

$$\frac{\partial L(\mathbf{x}, \boldsymbol{\alpha}, \boldsymbol{\beta})}{\partial \boldsymbol{\beta}}|_{(\mathbf{x}^*, \alpha^*, \beta^*)} = \mathbf{0}, \tag{29}$$

$$\alpha_i^* g_i(\mathbf{x}) = 0, \quad 1 \leq i \leq k, \tag{30}$$

$$g_i(\mathbf{x}) \leq 0, \quad 1 \leq i \leq k, \tag{31}$$

$$\alpha_i^* \geq 0, \quad 1 \leq i \leq k. \tag{32}$$

Eq. 30 is referred to as **Karush-Kuhn-Tucker complementarity condition**, which will be discussed in the following simple examples:

Example 1:

minimize $f(x) = x^2$, subject to $g(x) = x + 1 \leq 0$

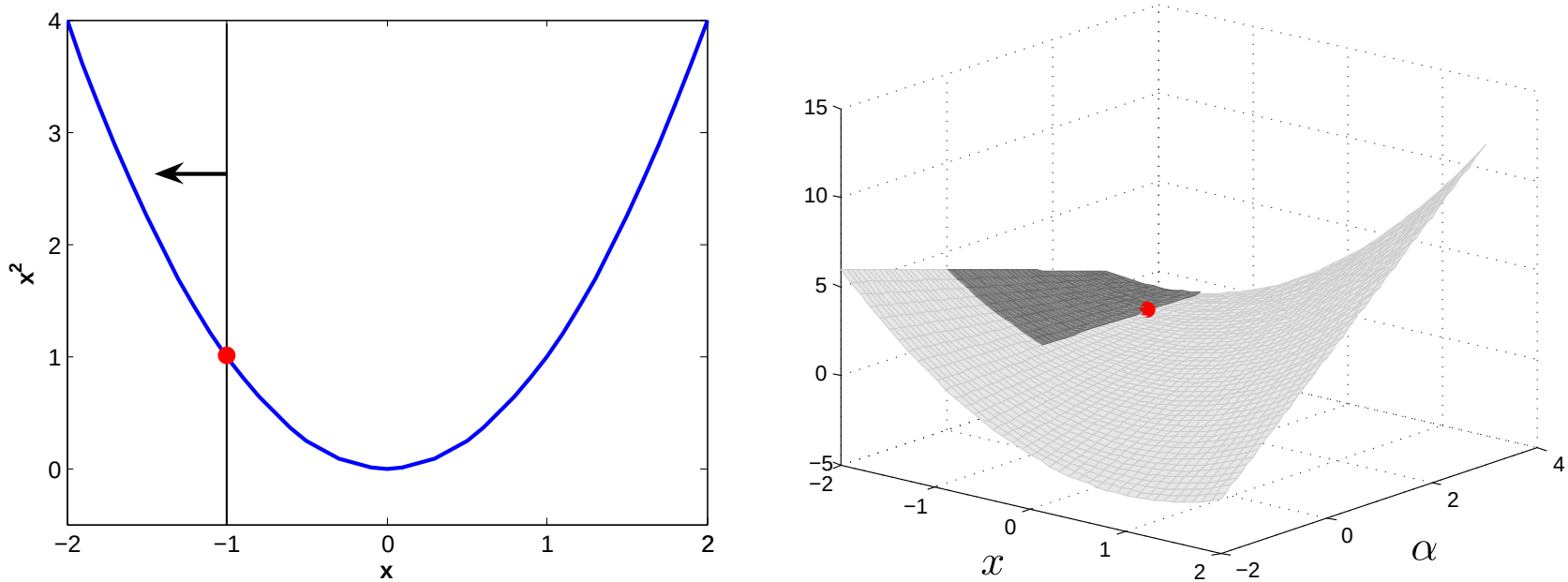


Figure: The optimum $(x^* = -1, \alpha^* = 2)$ lies on the boundary $x = -1$ of the feasible region of x (the dark patch marks the "combined" feasible region).

In the example of figure , $g(x) = 0$ follows from Eq. 30 because $\alpha > 0$ (g is said to be **active**).
Note that substituting $x = -\frac{\alpha}{2}$ (condition Eq. 28) into the Lagrangian function leads to another optimization problem:

$$\text{maximize} \quad -\frac{\alpha^2}{4} + \alpha, \text{subject to} \quad \alpha \geq 0. \tag{33}$$

This optimization problem is called the **dual optimization problem**. It is defined only in terms of α (in general in terms of k dual variables $\alpha_i, \quad 1 \leq i \leq k$). Eq.30 implies that the "dual contraint" $\alpha \geq 0$ can only be active (i.e., be satisfied with equality $\alpha = 0$), if the primal contraint is inactive ($g(x) < 0$) and vice versa.
For a convex criterion function and affine contraints primal and dual problem lead to the same solution for $\mathbf{x}^*, \boldsymbol{\alpha}^*$.

Example 2:

minimize $f(x) = x^2$, subject to $g(x) = x - 1 \leq 0$

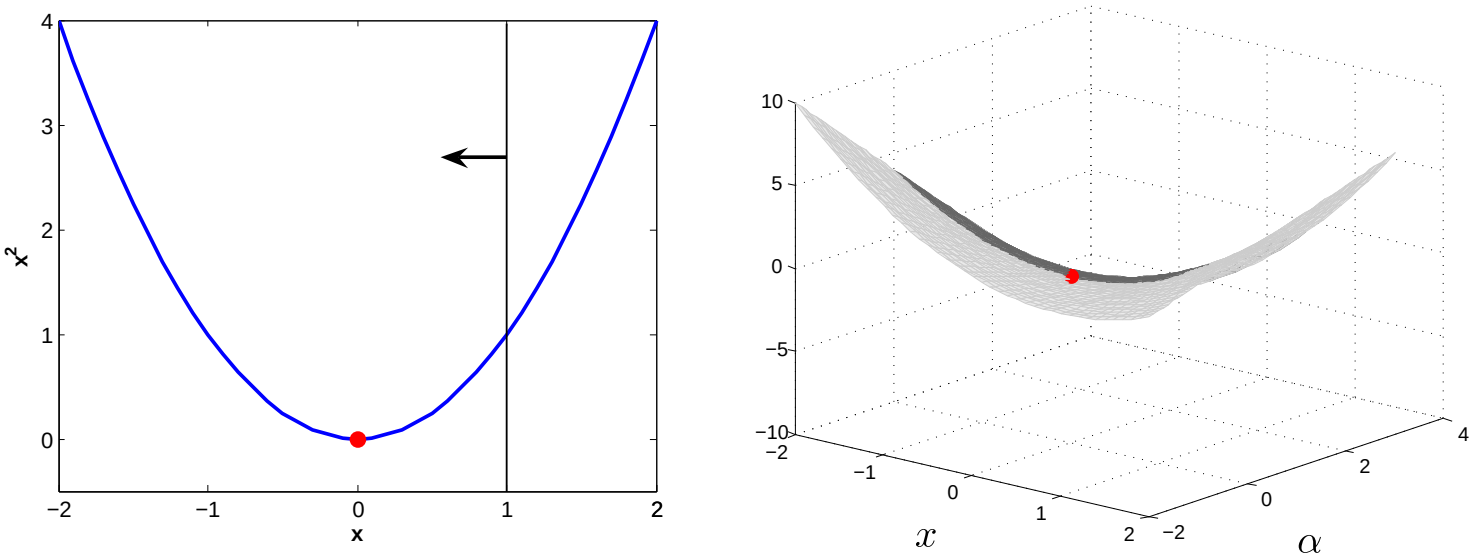


Figure: The contraint $g(x) \leq 0$ is inactive leading to a dual variable $\alpha = 0$ (Eq. 30). The solution is $x^* = 0, \alpha = 0$ and lies on the boundary $\alpha = 0$ of the "dual" feasible region.

SVM Training as constrained optimization

We now return to the problem of training an SVM. As we have seen above the search for an optimal hyperplane can be stated as the following *convex* optimization problem: Minimize

$$J(\mathbf{w}) = \frac{1}{2}\|\mathbf{w}\|^2 \qquad \text{(quadratic)} \tag{34}$$

subject to the *affine* constraints

$$(\mathbf{w}^T\mathbf{x}_i + w_0)t_i \geq 1, \quad 1 \leq i \leq N. \tag{35}$$

We build the functional

$$L(\mathbf{w}, w_0, \boldsymbol{\alpha}) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{N} \alpha_i(\mathbf{w}^T \mathbf{x}_i t_i + w_0 t_i - 1), \tag{36}$$

minimize it w.r.t. $\mathbf{w}$ while at the same time maximize it w.r.t. the Lagrange multipliers $\alpha_i \geq 0$, i.e. solve

$$\max_{\alpha \geq 0} \left(\min_{\mathbf{w}, w_0} \left(L(\mathbf{w}, w_0, \boldsymbol{\alpha}) \right) \right). \tag{37}$$

We use condition (cf. Eq. 28)

$$\frac{\partial L(\mathbf{w}^*, w_0^*, \boldsymbol{\alpha}^*)}{\partial \mathbf{w}} = 0 \tag{38}$$

$$\frac{\partial L(\mathbf{w}^*, w_0^*, \boldsymbol{\alpha}^*)}{\partial w_0} = 0 \tag{39}$$

for solutions $\mathbf{w}^*, w_0^*, \boldsymbol{\alpha}^*$ so that we can write

$$\mathbf{w}^* = \sum_{i=1}^{N} \alpha_i^* \mathbf{x}_i t_i, \qquad \alpha_i^* \geq 0, \quad 1 \leq i \leq N. \tag{40}$$

and

$$\sum_{i=1}^{N} \alpha_i^* t_i = 0 \tag{41}$$

Note that Eq. 40 is in fact the same **dual representation** as given by Eq. 11.

In addition the Kuhn-Tucker theorem states that $\alpha_i > 0$ only if $\mathbf{x}_i t_i$ satisfies Eq. 35 with equality (see Eq. 30).
This means that the actual number of parameters involved may be significantly fewer than the full training set size.
The data points for which the (dual variables) $\alpha_i > 0$ are called **support vectors**.
The support vectors are marked in figures 9 and 10 by circles.

Next by substituting Eq. 40 into Eq. 36 we obtain

$$L(\boldsymbol{\alpha}) = -\frac{1}{2}\sum_{i,j=1}^{N}\alpha_i\alpha_j t_i t_j(\mathbf{x}_i^T\mathbf{x}_j) + \sum_{i=1}^{N}\alpha_i. \tag{42}$$

which is the criterion function of the **dual optimization problem**. It has to be maximized w.r.t. $\alpha_1, \ldots, \alpha_N$ subject to the constraints

$$\sum_{i=1}^{N}\alpha_i t_i = 0, \alpha_i \geq 0, \quad 1 \leq i \leq N. \tag{43}$$

Note that in the dual formulation there are equality constraints making the problem considerably easier to solve (by standard quadratic programming methods).

The kernel trick

Consider we use a nonlinear mapping $\phi : \mathbf{R}^2 \mapsto \mathbf{R}^6$ given by

$$\phi(\mathbf{x}) = \phi(x_1, x_2) = \begin{pmatrix} 1 \\ \sqrt{2}x_1 \\ \sqrt{2}x_2 \\ \sqrt{2}x_1x_2 \\ x_1^2 \\ x_2^2 \end{pmatrix} \tag{44}$$

The kernel trick

Now consider the inner product of two feature vectors (measurements)

$$\phi(\mathbf{x}) = \begin{pmatrix} 1 \\ \sqrt{2}x_1 \\ \sqrt{2}x_2 \\ \sqrt{2}x_1x_2 \\ x_1^2 \\ x_2^2 \end{pmatrix} \qquad \phi(\bar{\mathbf{x}}) = \begin{pmatrix} 1 \\ \sqrt{2}\bar{x}_1 \\ \sqrt{2}\bar{x}_2 \\ \sqrt{2}\bar{x}_1\bar{x}_2 \\ \bar{x}_1^2 \\ \bar{x}_2^2 \end{pmatrix}$$

$$\phi(\mathbf{x})^T\phi(\bar{\mathbf{x}}) = 1 + 2x_1\bar{x}_1 + 2x_2\bar{x}_2 + 2x_1x_2\bar{x}_1\bar{x}_2 + x_1^2\bar{x}_1^2 + x_2^2\bar{x}_2^2$$

The kernel trick

The inner product can be computed without having to compute $\phi(\mathbf{x})$ and $\phi(\bar{\mathbf{x}})$ explicitly:

$$\begin{aligned}\phi(\mathbf{x})^T\phi(\bar{\mathbf{x}}) &= 1 + 2x_1\bar{x}_1 + 2x_2\bar{x}_2 + 2x_1x_2\bar{x}_1\bar{x}_2 \\ &\quad + x_1^2\bar{x}_1^2 + x_2^2\bar{x}_2^2 \\ &= (x_1\bar{x}_1 + x_2\bar{x}_2 + 1)^2 \\ &= (\mathbf{x}^T\bar{\mathbf{x}} + 1)^2,\end{aligned}$$

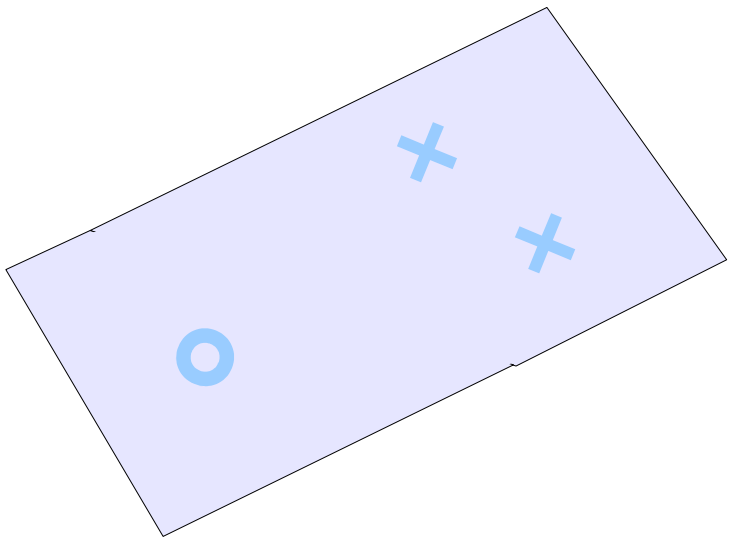
i.e., we have a way of computing the inner product directly as a function

$$\phi(\mathbf{x})^T\phi(\bar{\mathbf{x}}) = K(\mathbf{x}, \bar{\mathbf{x}})$$

We call such a *direct* computation method a **kernel function**.

The kernel trick

Assume that the solution vector $\mathbf{w}$ can be written in terms of the training data (the solution $\mathbf{w}$ can be chosen to lie in the span of the training data)



Example (Classification)

training set of $N = 3$ data points classified according to

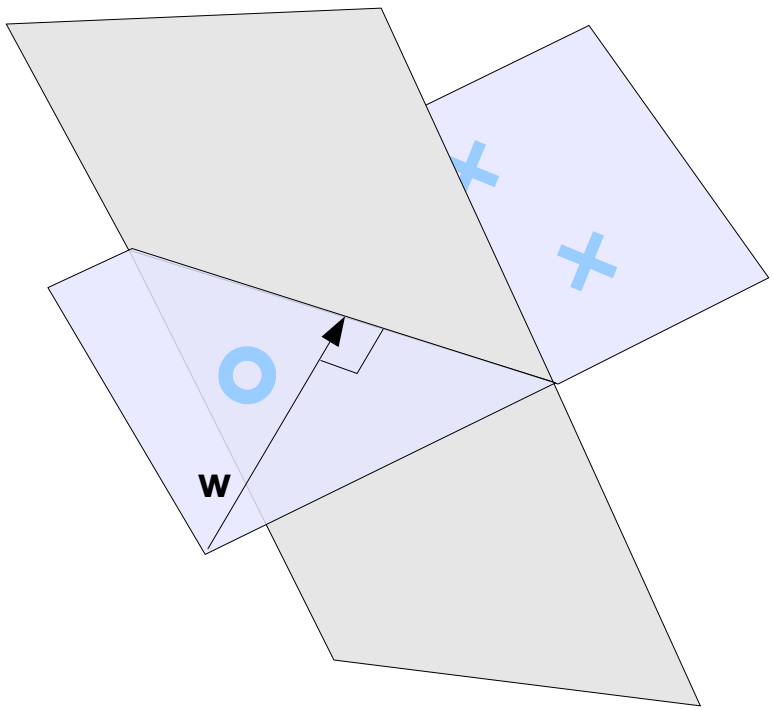
$$\text{sgn}(\mathbf{w}^T\mathbf{x} + d)$$

with

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

The kernel trick

Assume that the solution vector $\mathbf{w}$ can be written in terms of the training data (the solution $\mathbf{w}$ can be chosen to lie in the span of the training data)



Example (Classification)

training set of $N = 3$ data points classified according to

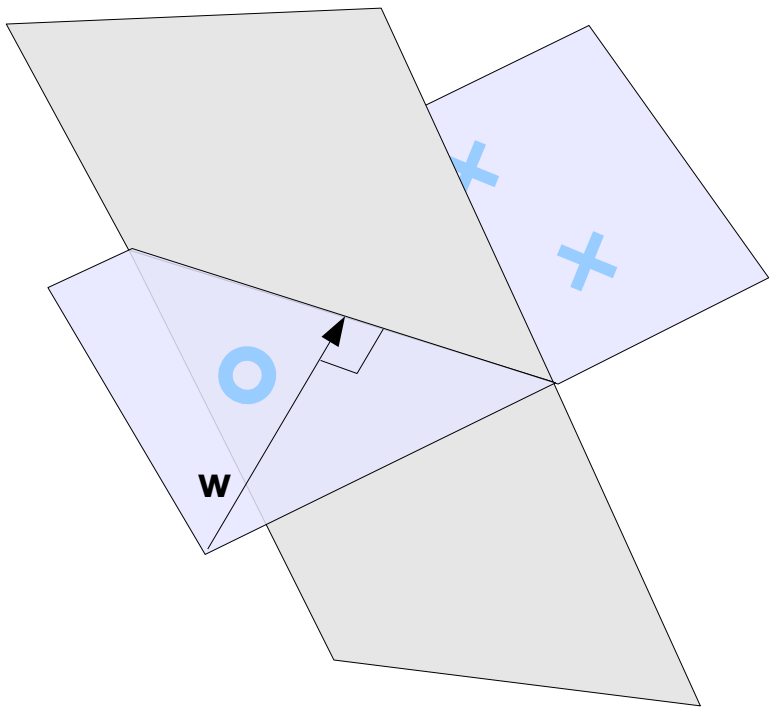
$$\text{sgn}(\mathbf{w}^T\mathbf{x} + d)$$

with

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

The kernel trick

Assume that the solution vector $\mathbf{w}$ can be written in terms of the training data (the solution $\mathbf{w}$ can be chosen to lie in the span of the training data)



Example (Classification)

$N = 3$ data points classified according to

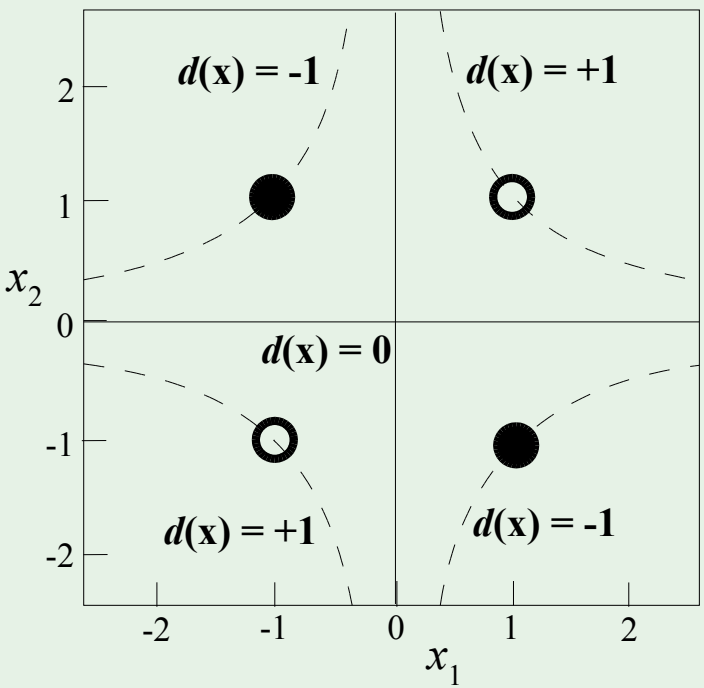
$$\text{sgn}(\sum_{i=1}^{N} \alpha_i \mathbf{x}_i^T \mathbf{x} + d)$$

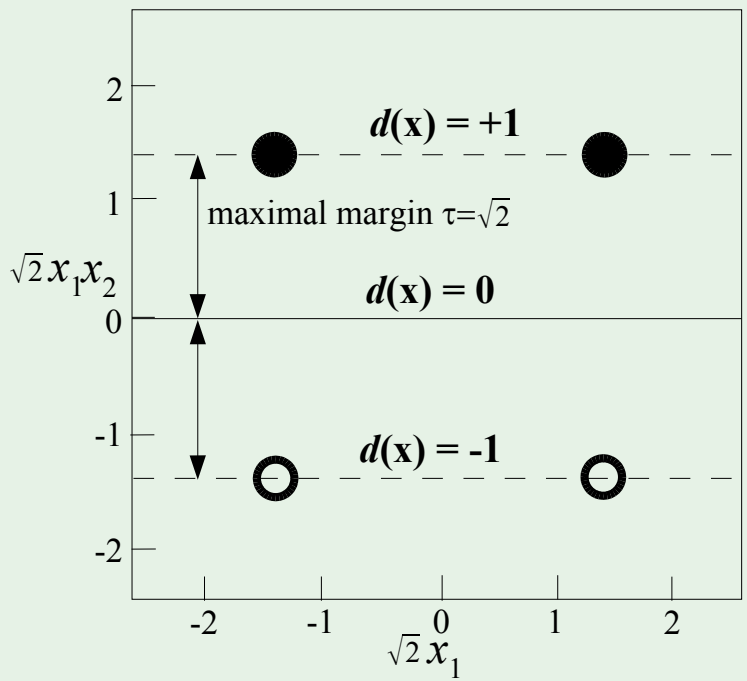
The kernel trick

We now have a generalized linear discriminant function

$$d(\mathbf{x}) = \text{sgn}(\sum_{i=1}^{N} \alpha_i K(\mathbf{x}_i, \mathbf{x}) + d)$$

Example (Maximal margin classifier with polynomial kernel)





Inner product kernels

Positive integral operator kernels can be associated with (sometimes infinite) hilbert spaces of non-linear functions from which ϕ is chosen. Other examples of kernel functions are:

- polynomials of degree d: $K(\mathbf{x}, \mathbf{y}) = (\mathbf{x}^T \mathbf{y} + 1)^d$
- radial basis functions: $K(\mathbf{x}, \mathbf{y}) = \exp(-\frac{\|\mathbf{x}-\mathbf{y}\|^2}{\sigma^2})$ (infinite dimensional)
- multilayer perceptrons: $K(\mathbf{x}, \mathbf{y}) = \tanh(v(\mathbf{x}^T \mathbf{y}) + a)$, where v, a are chosen to satisfy the positivity conditions.

Capacity of an SVM

For a separating hyperplane with $\|\mathbf{w}\|^2 \leq c$, the VC-dimension is bounded by

$$h \leq \min(r^2 c, d) + 1 \tag{45}$$

where r is the radius of the smallest sphere containing all training input vectors $\mathbf{x}_1, \ldots, \mathbf{x}_N$. Note that by choosing c it is possible to control the complexity of the hyperplane *independent of the dimensionality of the input vector space* d.

SVM for classification - XOR example

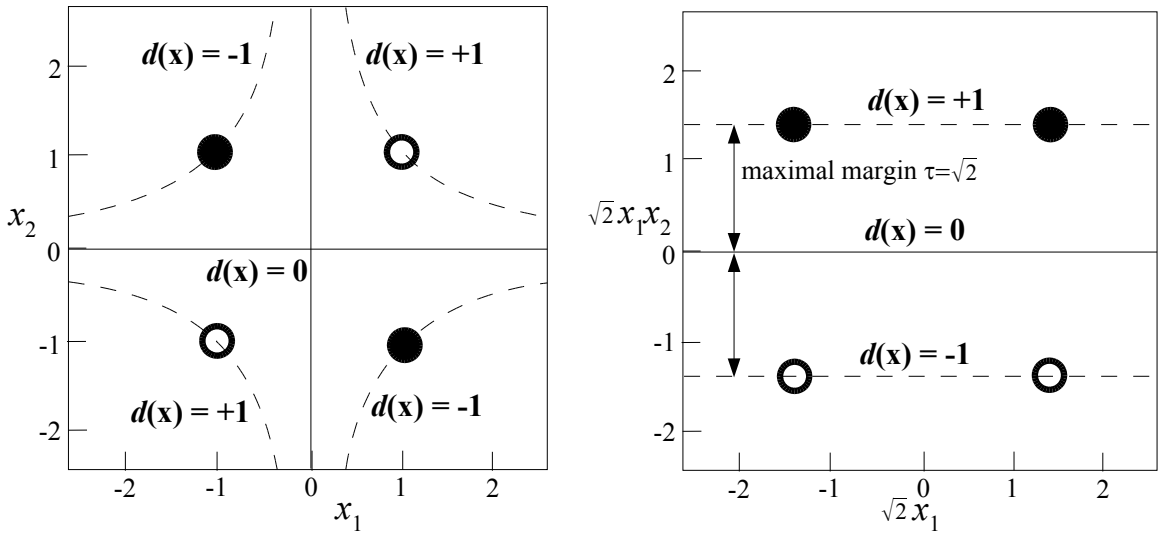


Figure: It is not possible to solve the XOR-Problem with a linear decision boundary. However, a polynomial decision boundary of order 2 can seperate these data points.

The inner product kernel for polynomials of order two is

$$K(\mathbf{x}, \mathbf{y}) = (\mathbf{x}^T \mathbf{y} + 1)^2. \tag{46}$$

This expression corresponds to the set of basis functions $\phi(\mathbf{x}) = (1, \sqrt{2}x_1, \sqrt{2}x_2, \sqrt{2}x_1x_2, x_1^2, x_2^2)^T$ (cf. Eq. 44).
The inner product kernel is given by the matrix

$$K = \begin{pmatrix} 9 & 1 & 1 & 1 \\ 1 & 9 & 1 & 1 \\ 1 & 1 & 9 & 1 \\ 1 & 1 & 1 & 9 \end{pmatrix} \tag{47}$$

with elements k_{ij} computed using Eq. 46.

To determine the decision boundary in the transformed space, we must solve the following optimization problem: Maximize (cf. Eq. 42)

$$L(\boldsymbol{\alpha}) = \alpha_1 + \alpha_2 + \alpha_3 + \alpha_4 - \frac{1}{2}\sum_{i=1}^{4}\sum_{j=1}^{4}\alpha_i\alpha_j t_i t_j k_{ij}, \tag{48}$$

subject to

$$\sum_{i=1}^{4} t_i\alpha_i = \alpha_1 - \alpha_2 + \alpha_3 - \alpha_4 = 0, \tag{49}$$

$$0 \leq \alpha_1, \tag{50}$$

$$0 \leq \alpha_2, \tag{51}$$

$$0 \leq \alpha_3, \tag{52}$$

$$0 \leq \alpha_4. \tag{53}$$

The solution is $\alpha_1 = \alpha_2 = \alpha_3 = \alpha_4 = 0.125$ indicating that all four data points are support vectors. The decision function in the inner product representation is

$$d(\mathbf{x}) = \sum_{i=1}^{N} \alpha_i^* t_i K(\mathbf{x}_i, \mathbf{x}) = 0.125 \sum_{i=1}^{4} t_i (\mathbf{x}_i^T \mathbf{x} + 1)^2. \tag{54}$$

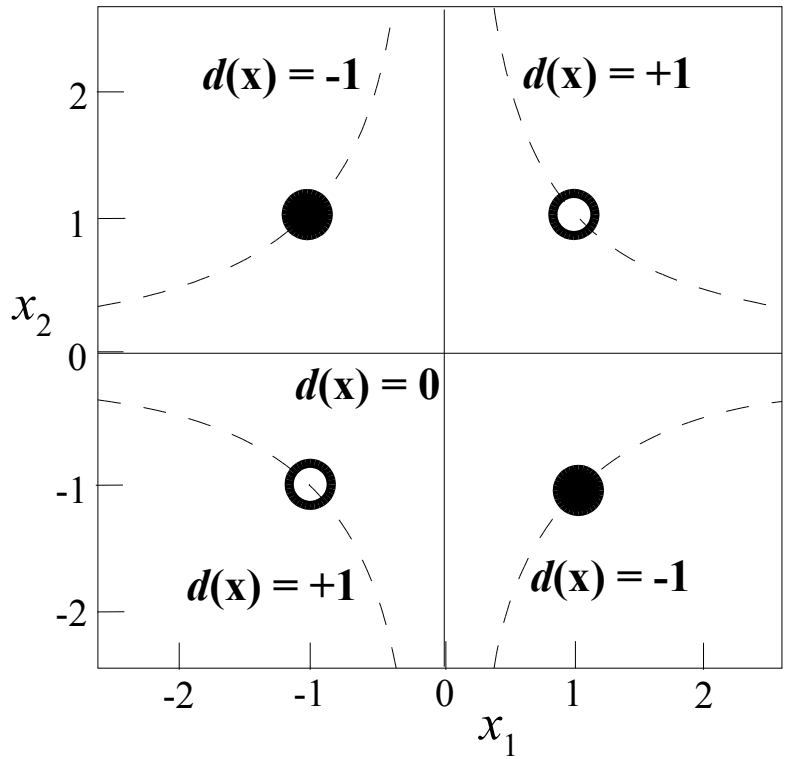


Figure: Decision function determined by the support vector machine with a feature space of second order polynomials. In the original 2-dimensional input space, the decision function is nonlinear.

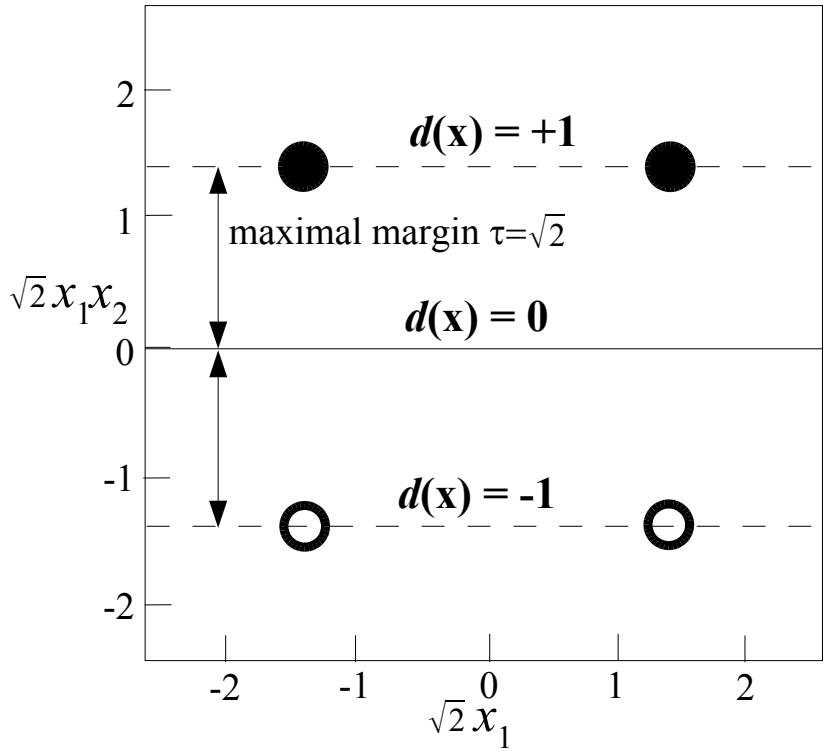


Figure: In the 6-dimensional feature space the decision function is linear with maximal margin.

Soft margin

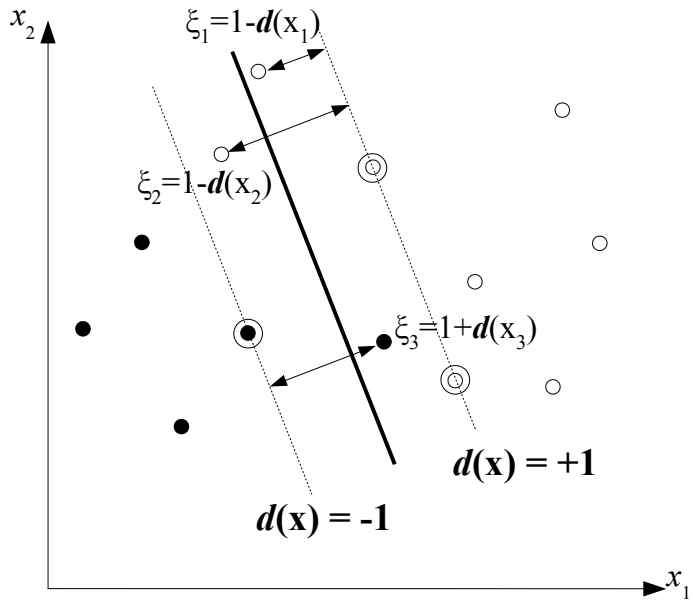


Figure: For training data that cannot be separated without error the number of errors can be minimized using slack variables ξ_i, which are positive for nonseparable and greater than one for misclassified data points.

The optimization problem becomes: minimize

$$\sum_{i=1}^{N} \xi_i^p, \tag{55}$$

subject to

$$t_i(\mathbf{w}^T\mathbf{x} + w_0) \geq 1 - \xi_i^p, \tag{56}$$

using the structure

$$\mathcal{S}_k = \{\mathbf{w}^T\mathbf{x} + w_0 : \|\mathbf{w}\|^2 \leq c_k\}, \tag{57}$$

where c_k selects the trade-off between complexity (cf. Eq. 45) and proportion of nonseparable data (number of $\xi_i > 0$).
p is a small positive constant, which is usually set to 1, to make the problem tractable.

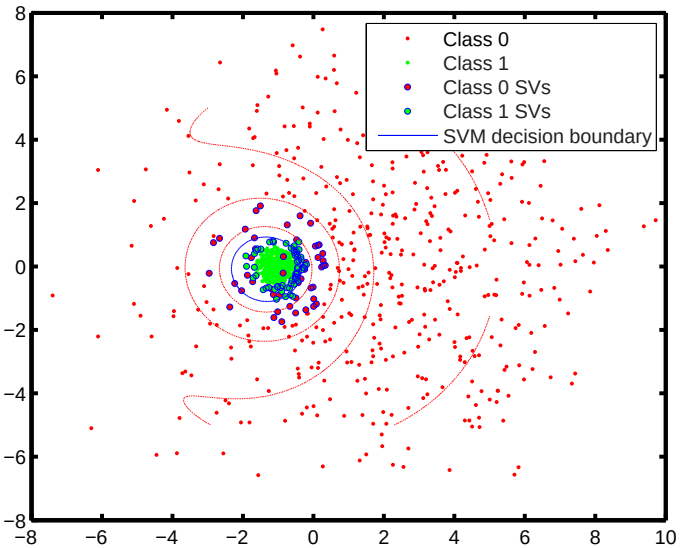
The dual form of the optimization problem is: Maximize

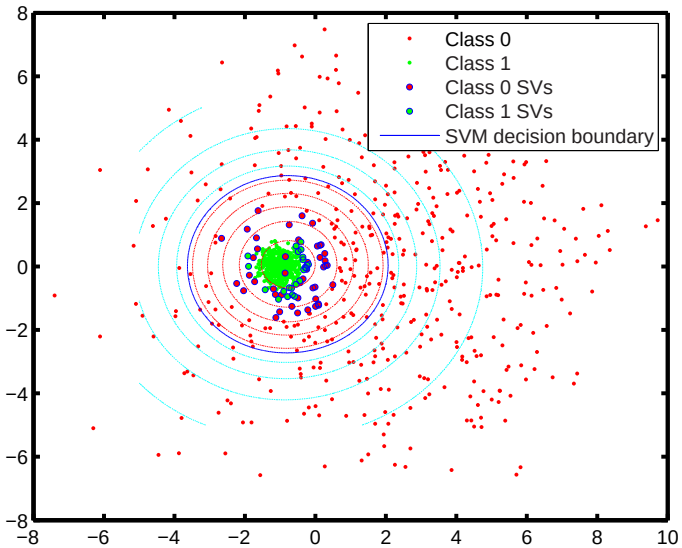
$$L(\boldsymbol{\alpha}) = -\frac{1}{2}\sum_{i,j=1}^{N}\alpha_i\alpha_j t_i t_j(\mathbf{x}_i^T\mathbf{x}_j) + \sum_{i=1}^{N}\alpha_i, \tag{58}$$

subject to

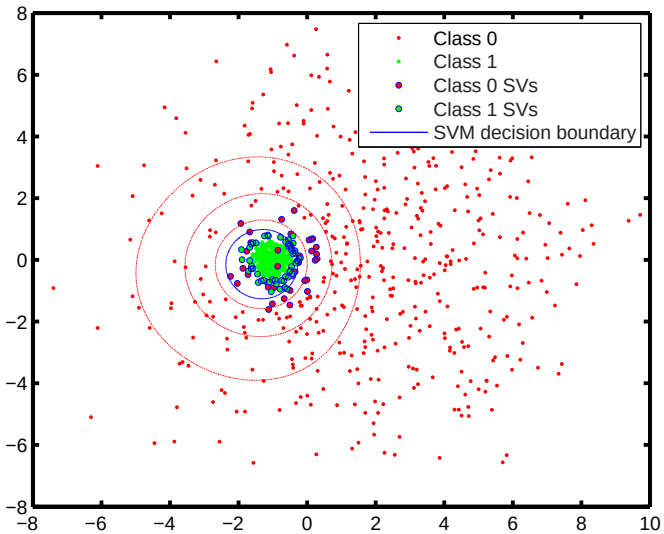
$$\sum_{i=1}^{N} t_i\alpha_i = 0, 0 \leq \alpha_i \leq \frac{C}{N}, i = 1, \ldots, N. \tag{59}$$

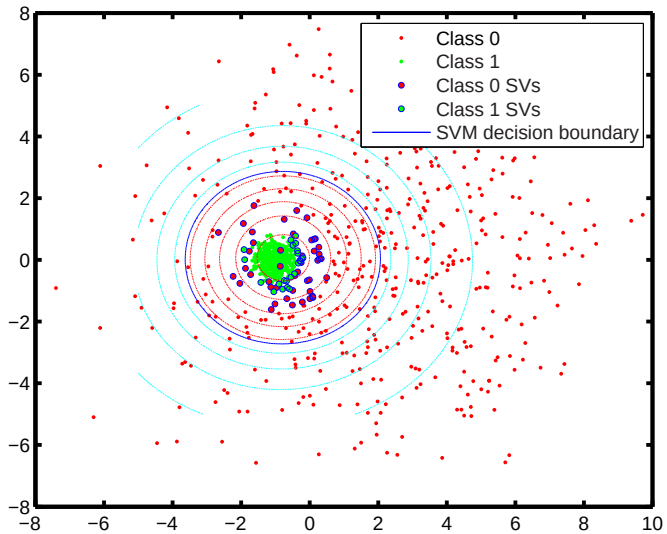
influence of the kernel width σ





Influence of the regularization parameter C





SVM Overview

- follows a principled approach rooted in statistical learning theory: **structural risk minimization**.
- uses worst-case error bound instead of empirical risk
- good generalization performance
- without incorporating problem-domain knowledge (but **no free lunch**).
- kernels: non-linear learning machine