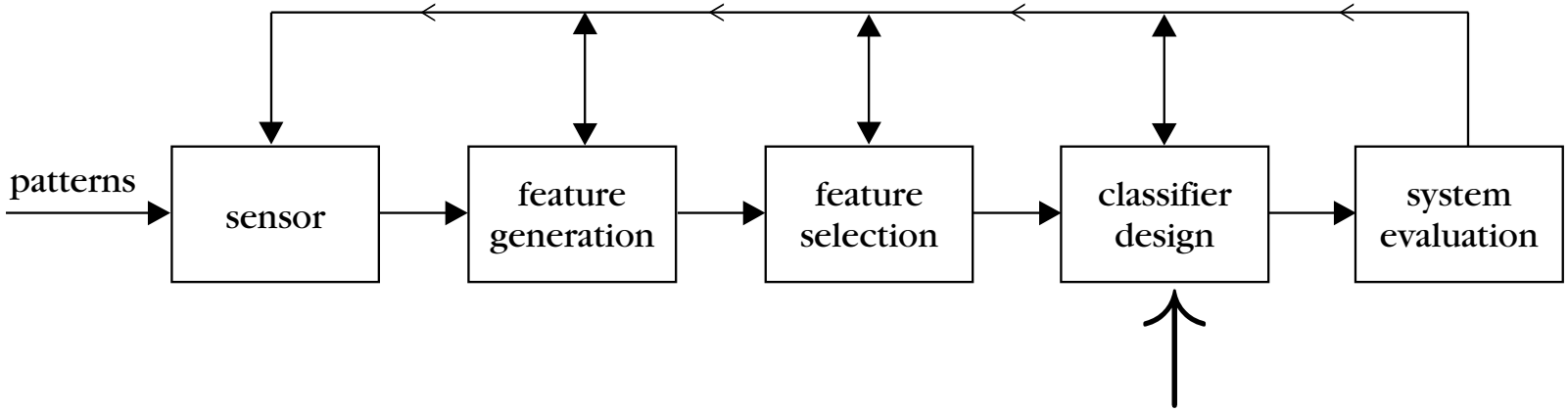
Pattern Recognition Lecture
Linear Classifiers

Prof. Dr. Marcin Grzegorzek

Research Group for Pattern Recognition
Institute for Vision and Graphics
University of Siegen, Germany

Pattern Recognition Chain



Overview

Overview

Introducing Example

Known

- A two-class problem $\Omega = \{\omega_1, \omega_2\}$ in a 2D feature space $\mathbf{x} = [x_1, x_2]^{\mathsf{T}}$ is considered.
- The classifier is given by

$$y = 2x_1 + x_2$$

and

$$\begin{cases} y > 5 & \Rightarrow \quad i = 1 \\ y \leq 5 & \Rightarrow \quad i = 2 \end{cases}$$

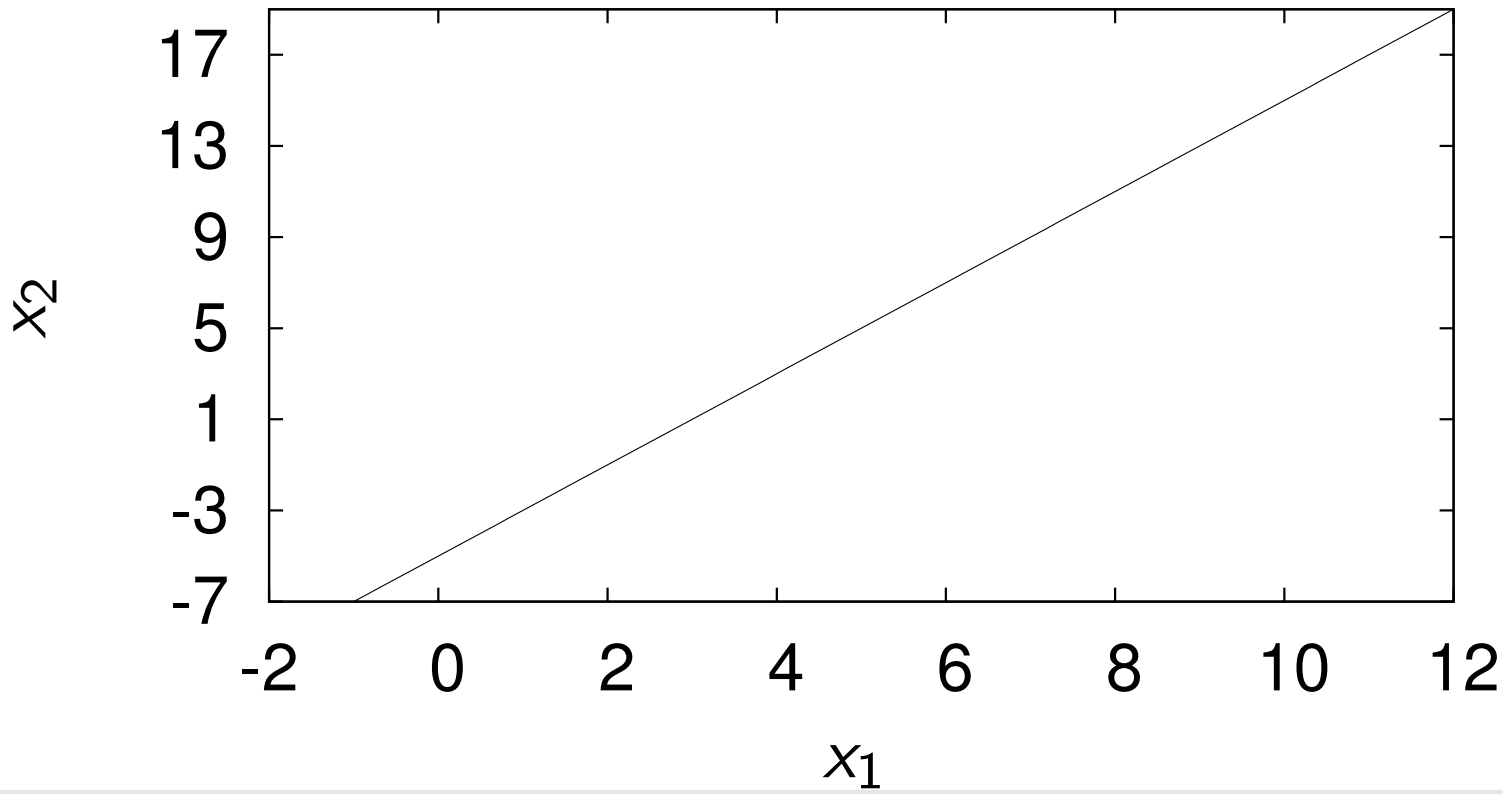
Task

- Find the decision line!

Solution

Yes, it is that simple as it sounds. The decision line is just given by

$$x_2 = 2x_1 - 5$$



Decision Hyperplanes for l-Dimensions (1)

- Let us focus on the two-class problem and consider linear discriminant functions. The decision hypersurface in the l-dimensional feature space is then given by

$$\mathbf{w}^{\mathsf{T}}\mathbf{x} = 0$$

- The dimensionality problem ($\mathbf{w} \in \mathbb{R}^{l+1}$, but feature vectors have l elements) is overcome by increasing the dimensionality of each feature vector, so that

$$\mathbf{x} = [x_1, x_2, \ldots, x_l, 1]^{\mathsf{T}}$$

  This does not change anything in the linear classification process.

Decision Hyperplanes for l-Dimensions (2)

- If $\mathbf{x}_1$ and $\mathbf{x}_2$ are two points on the decision hyperplane, then the following is valid

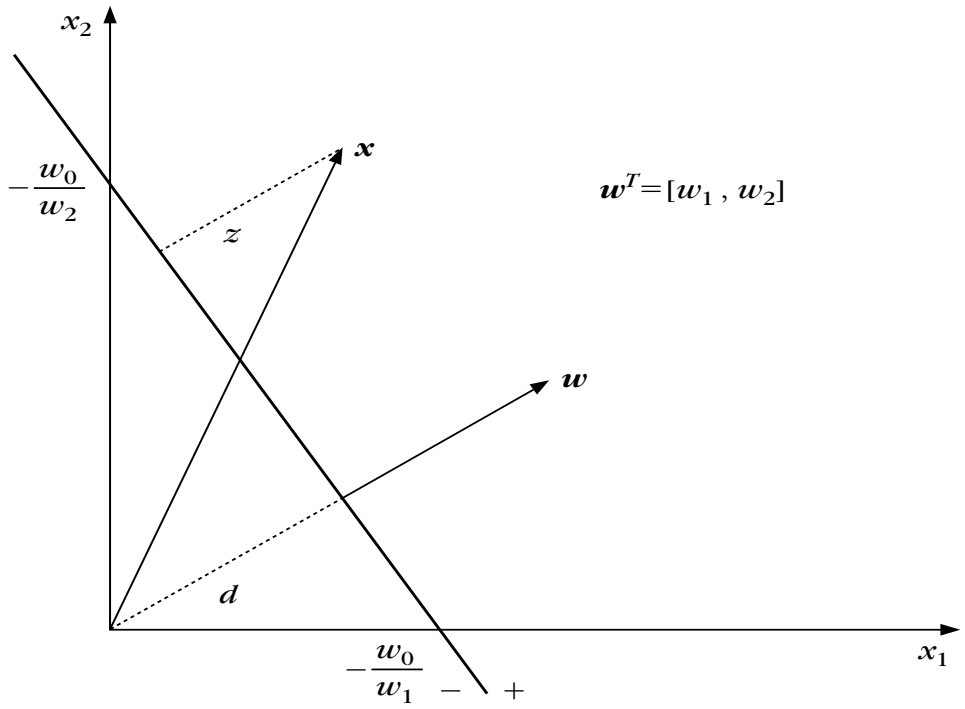
$$\mathbf{w}^{\mathsf{T}}\mathbf{x}_1 = \mathbf{w}^{\mathsf{T}}\mathbf{x}_2 = 0$$

$$\Updownarrow$$

$$\mathbf{w}^{\mathsf{T}}(\mathbf{x}_1 - \mathbf{x}_2) = 0$$

- Since the difference vector $\mathbf{x} = \mathbf{x}_1 - \mathbf{x}_2$ obviously lies on the decision hyperplane, it is apparent that the weight vector $\mathbf{w}$ is orthogonal to the decision hyperplane.

Decision Hyperplanes for l-Dimensions (3)



$$d = \frac{|w_0|}{\sqrt{w_1^2 + w_2^2}} \qquad z = \frac{|g(\mathbf{x})|}{\sqrt{w_1^2 + w_2^2}}$$

Overview

Problem Statement

Problem

How to compute the unknown parameters $w_1, \ldots, w_l, w_0$?

Assumptions

The two classes ω_1 and ω_2 are linearly separable, i. e., there exist a hyperplane $\widehat{\mathbf{w}}$ such that

$$\widehat{\mathbf{w}}^{\mathsf{T}}\mathbf{x} > 0; \qquad \forall \mathbf{x} \in \omega_1$$

$$\widehat{\mathbf{w}}^{\mathsf{T}}\mathbf{x} < 0; \qquad \forall \mathbf{x} \in \omega_2$$

Approach

The problem will be solved as an optimisation task. Therefore, we need:

- an appropriate cost function
- an algorithmic scheme to optimise it

Perceptron Cost Function - Definition

- As cost function the perceptron cost will be used:

$$J(\mathbf{w}) = \sum_{\mathbf{x} \in Y} (\delta_x \mathbf{w}^{\mathsf{T}} \mathbf{x})$$

- Y - subset of training vectors misclassified by the hyperplane $\mathbf{w}$

- The variable δ_x is chosen so that:

$$\begin{cases} \mathbf{x} \in \omega_1 & \Rightarrow \quad \delta_x = -1 \\ \mathbf{x} \in \omega_2 & \Rightarrow \quad \delta_x = +1 \end{cases}$$

Perceptron Cost Function - Properties

- The perceptron cost is not negative. It becomes zero when $Y = \emptyset$, that is, if there are no misclassified vectors $\mathbf{x}$
- Indeed, if $\mathbf{x} \in \omega_1$ and it is misclassified, then $\mathbf{w}^T\mathbf{x} < 0$ and $\delta_x < 0$. Thus, the product is positive
- The perceptron cost function is continuous and piecewise linear

Minimisation of the Perceptron Cost Function (1)

- The iterative minimisation works according to:

$$\mathbf{w}(t+1) = \mathbf{w}(t) - \rho_t \left. \frac{\partial J(\mathbf{w})}{\partial \mathbf{w}} \right|_{\mathbf{w}=\mathbf{w}(t)}$$

- $\mathbf{w}$ is the weight vector at the iteration step no. t

- ρ_t is a positive real number chosen manually.

Minimisation of the Perceptron Cost Function (2)

- From the perceptron definition and the points where this is valid, we get

$$\frac{\partial J(\mathbf{w})}{\partial \mathbf{w}} = \sum_{\mathbf{x} \in Y} \delta_x \mathbf{x}$$

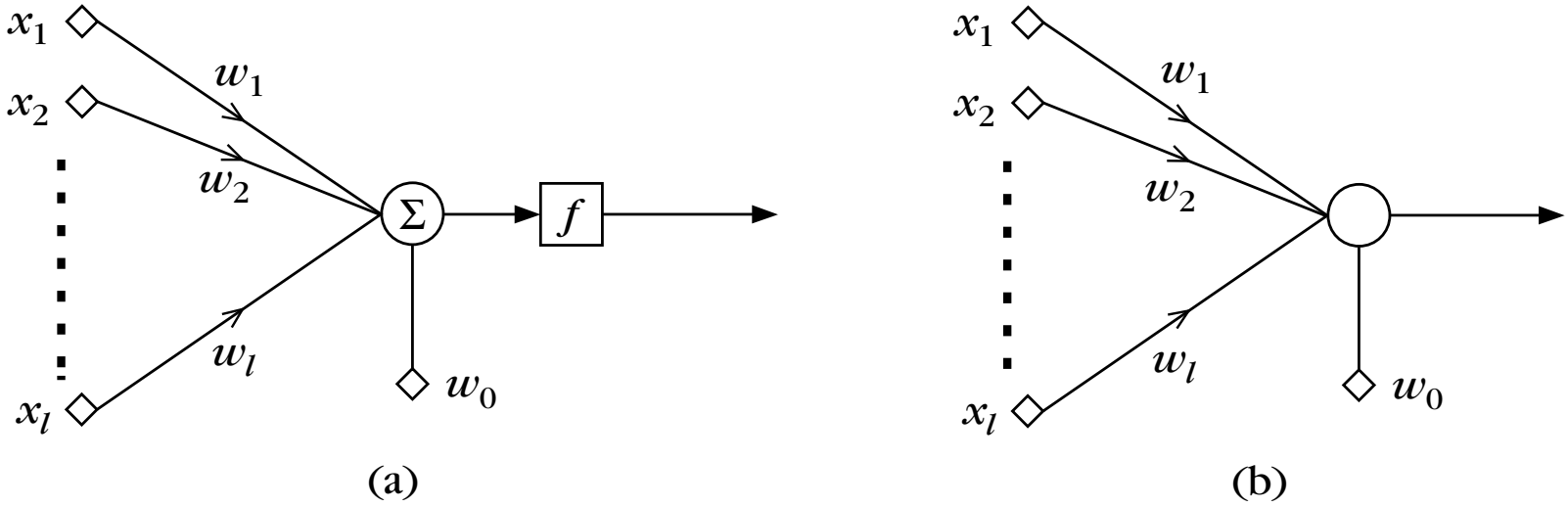
- Thus, the iterative minimisation of the cost function from the previous slide can be written as

$$\mathbf{w}(t+1) = \mathbf{w}(t) - \rho_t \sum_{\mathbf{x} \in Y} \delta_x \mathbf{x}$$

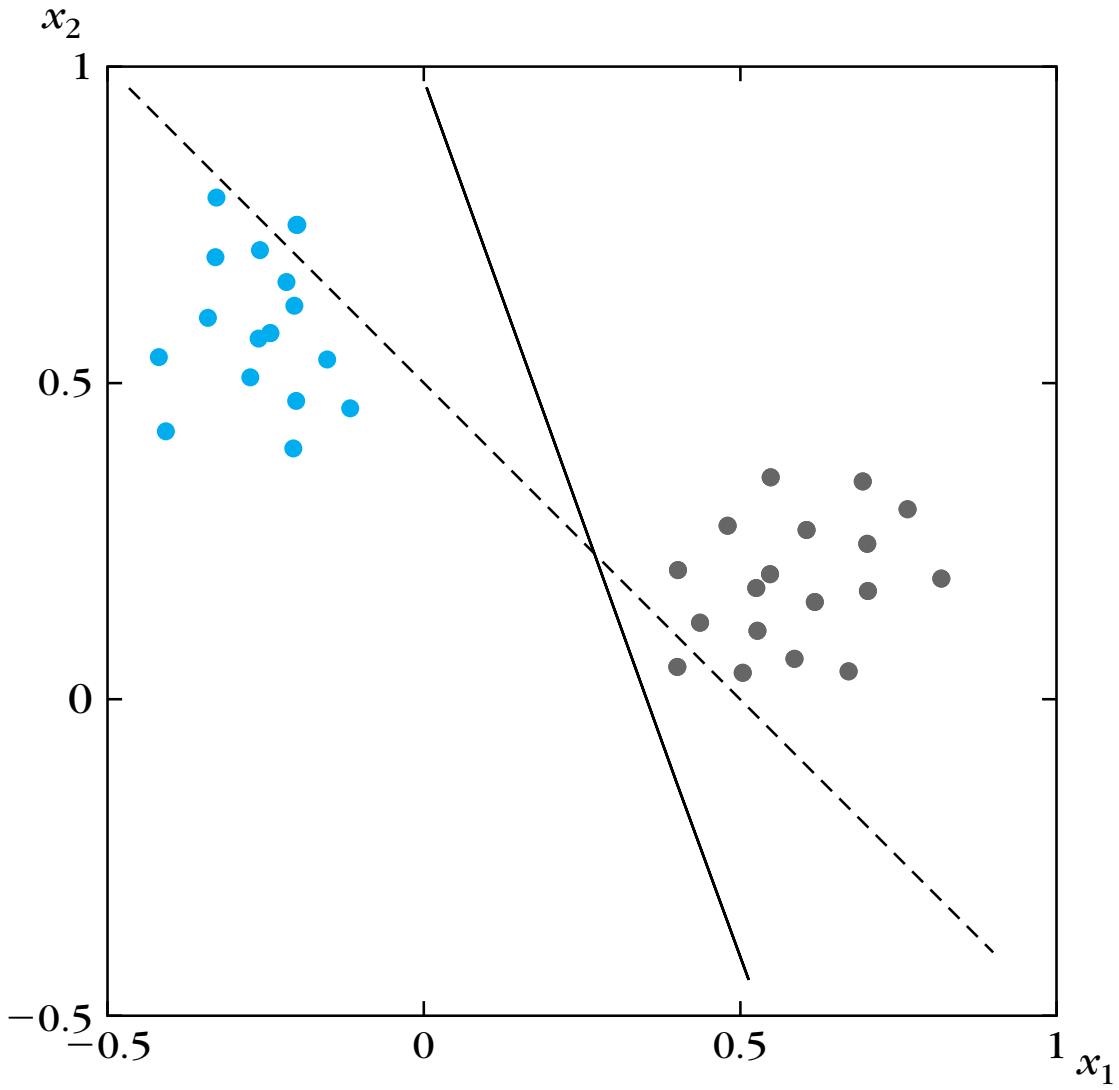
The Perceptron Algorithm - Pseudocode

- Choose $\mathbf{w}(0)$ randomly
- Choose ρ_0
- $t = 0$
- Repeat
  - Set $Y = \emptyset$
  - For $j = 1$ to K
    - If $\delta_{x_j} \mathbf{w}(j)^{\mathsf{T}} \mathbf{x}_j \geq 0$ then $Y = Y \cup \{\mathbf{x}_j\}$
  - End For
  - $\mathbf{w}(t+1) = \mathbf{w}(t) - \rho_t \sum_{\mathbf{x} \in Y} \delta_x \mathbf{x}$
  - Adjust ρ_t
  - Iterate $t = t + 1$
- Until $Y = \emptyset$

The Basic Perceptron Model



Example for the Perceptron Algorithm (1)



Example for the Perceptron Algorithm (2)

Known

- Decision line after the iteration no. t is given by

$$x_1 + x_2 - 0.5 = 0 \quad \Leftrightarrow \quad \mathbf{w}(t) = [1, 1, -0.5]^{\mathsf{T}}$$

- With $\rho_t = 0.7$
- Vectors misclassified: $[0.4, 0.05]^{\mathsf{T}}$ and $[-0.2, 0.75]^{\mathsf{T}}$

Unknown

- The decision line after the iteration no. $t + 1$:

$$\mathbf{w}(t+1) = \begin{bmatrix} w_1(t+1) \\ w_2(t+1) \\ w_0(t+1) \end{bmatrix} = ?$$

Example for the Perceptron Algorithm (3)

$$\mathbf{w}(t+1) = \begin{bmatrix} 1 \\ 1 \\ -0.5 \end{bmatrix} - 0.7(-1) \begin{bmatrix} 0.4 \\ 0.05 \\ 1 \end{bmatrix} - 0.7(+1) \begin{bmatrix} -0.2 \\ 0.75 \\ 1 \end{bmatrix}$$

$$\Updownarrow$$

$$\mathbf{w}(t+1) = \begin{bmatrix} 1.42 \\ 0.51 \\ -0.5 \end{bmatrix}$$

Note that the dimensionality of the misclassified vectors has been increased by one!

Overview

Mean Square Error Estimation

- Linear classifiers are fast, thus, they sometimes are applied even for classes that are not linearly separable.
- In this case, the desired output of a classifier $y(\mathbf{x}) = y$ is sometimes not equal to the real output $\mathbf{w}^{\mathsf{T}}\mathbf{x}$.
- The cost function expresses the mean square error (MSE) between the desired and the true outputs

$$J(\mathbf{w}) = E[|y - \mathbf{x}^{\mathsf{T}}\mathbf{w}|^2]$$

- To find the optimal separating hyperplane $\widehat{\mathbf{w}}$, the cost function is minimised with regard to $\mathbf{w} = [w_1, \ldots, w_l, w_0]^{\mathsf{T}}$

$$\widehat{\mathbf{w}} = \underset{\mathbf{w}}{\operatorname{argmin}}\, J(\mathbf{w})$$

Sum of Error Squares Estimation (1)

- Two-class problem with not separable classes is considered.
- The cost function here is the sum of error squares

$$J(\mathbf{w}) = \sum_{i=1}^{N} (y_i - \mathbf{x}^{\mathsf{T}}{}_i \mathbf{w})^2$$

- $y_i \in \{-1, 1\}$ is the desired output of the classifier for $\mathbf{x}_i$
- $\mathbf{x}^{\mathsf{T}}{}_i \mathbf{w}$ is the real output of the classifier for $\mathbf{x}_i$
- In order to find the optimal separating hyperplane $\widehat{\mathbf{w}}$, the cost function has to be minimised with respect to $\mathbf{w}$

$$\frac{\partial J(\mathbf{w})}{\partial \mathbf{w}} = 0 \quad \Leftrightarrow \quad \sum_{i=1}^{N} \mathbf{x}_i (y_i - \mathbf{x}^{\mathsf{T}}{}_i \widehat{\mathbf{w}}) = 0 \qquad (1)$$

Sum of Error Squares Estimation (2)

- The minimisation term (1) can be rewritten as follows:

$$\left(\sum_{i=1}^{N} \mathbf{x}_i \mathbf{x}^{\mathsf{T}}{}_i\right) \widehat{\mathbf{w}} = \sum_{i=1}^{N} (\mathbf{x}_i y_i) \tag{2}$$

- For the sake of formulation let us define

$$X = \begin{bmatrix} \mathbf{x}^{\mathsf{T}}{}_1 \\ \vdots \\ \mathbf{x}^{\mathsf{T}}{}_N \end{bmatrix} = \begin{bmatrix} x_{1,1} & \dots & x_{1,l} & 1 \\ \vdots & \ddots & \vdots & \vdots \\ x_{N,1} & \dots & x_{N,l} & 1 \end{bmatrix}, \mathbf{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix} \tag{3}$$

- X contains all training feature vectors for both classes, and $\mathbf{y}$ is a vector consisting of the corresponding desired responses $y_i \in \{-1, 1\}$.

Sum of Error Squares Estimation (3)

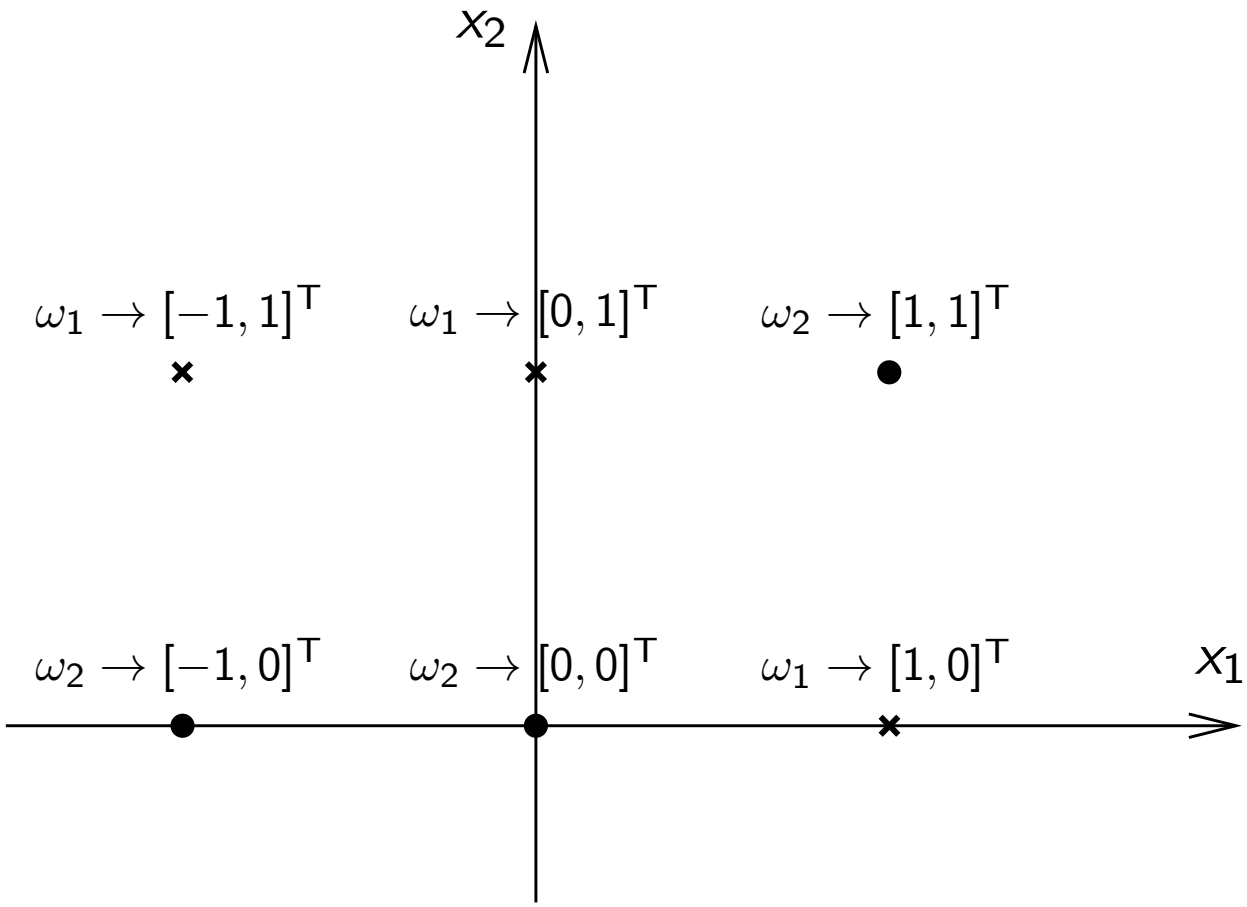
- Using both, (2) and (3) the following is true

$$(X^{\mathsf{T}}X)\widehat{\mathbf{w}} = X^{\mathsf{T}}\mathbf{y}$$

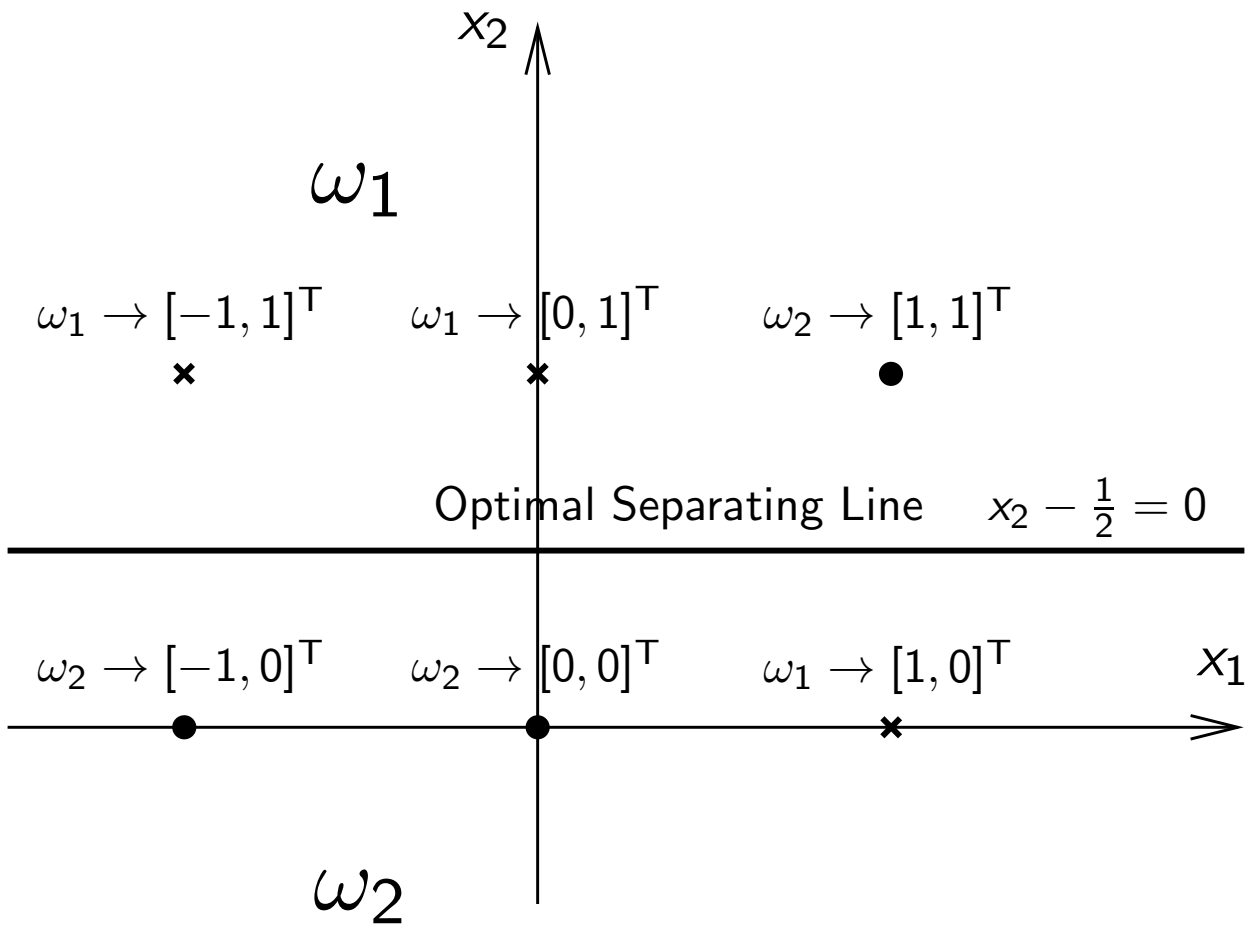
- Finally, the optimal separating hyperplane is given by

$$\widehat{\mathbf{w}} = (X^{\mathsf{T}}X)^{-1}X^{\mathsf{T}}\mathbf{y}$$

Sum of Error Squares Estimation - Example



Sum of Error Squares Estimation - Example



Overview

SVMs for Linearly Separable Classes (1)

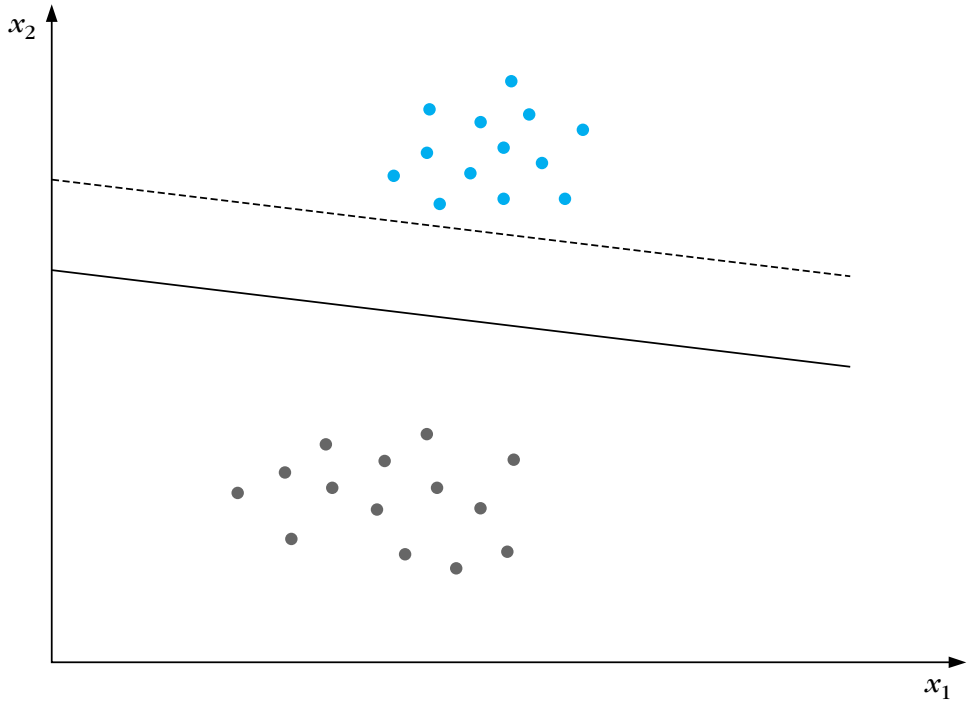
- A two-class problem $\Omega = \{\omega_1, \omega_2\}$
- $\mathbf{x}_{i=1,\ldots,N}$ are all training feature vectors
- The goal, once more, is to design a hyperplane[1]

$$g(\mathbf{x}) = \mathbf{w}^{\mathsf{T}}\mathbf{x} + w_0 = 0$$

that classifies correctly all the training feature vectors.

[1] Note that $\mathbf{w} = [w_1, \ldots, w_l]^{\mathsf{T}}$ and w_0 are treated separately here.

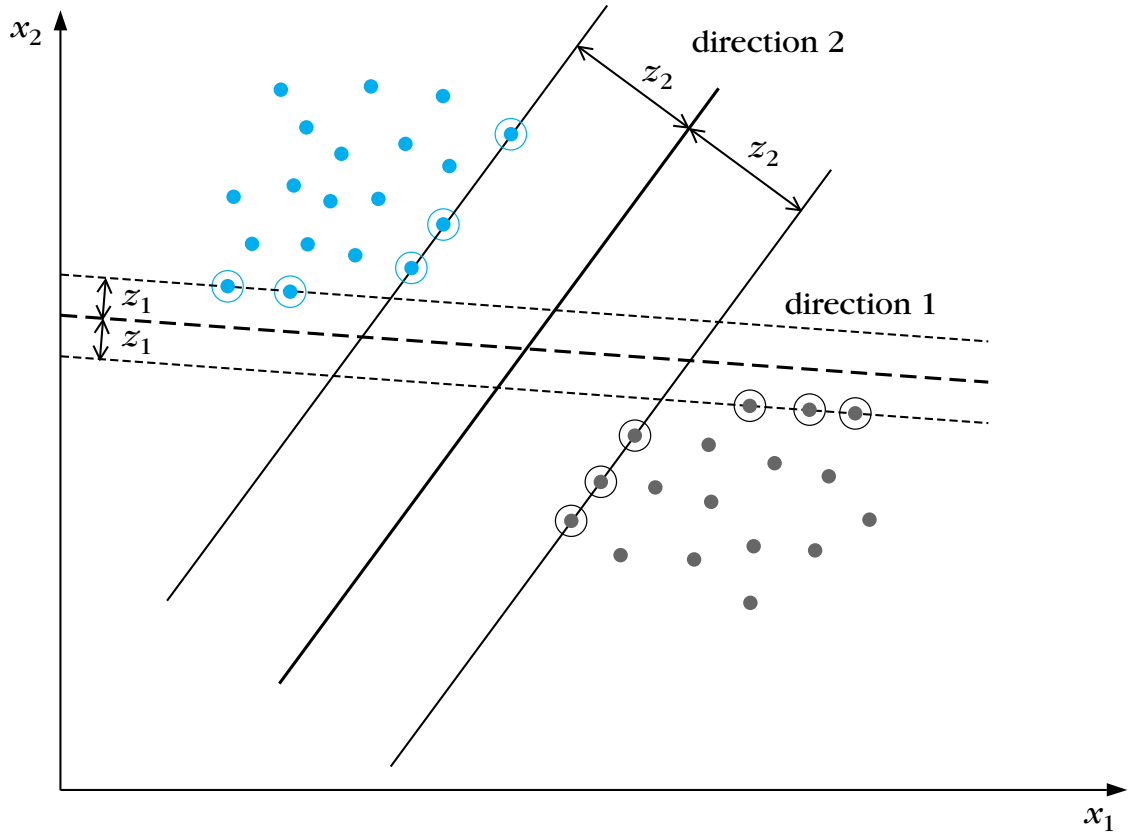
SVMs for Linearly Separable Classes (2)



- As we have seen for the perceptron algorithm, such a hyperplane is not unique.
- However, the full-line secures higher generalisation performance of the classifier, because it leaves the maximum margin from both classes.

SVMs for Linearly Separable Classes (3)

- The goal is to search for the direction that gives the maximum possible margin.



SVMs for Linearly Separable Classes (4)

- The distance of a point from a hyperplane is given by

$$z = \frac{|g(\mathbf{x})|}{||\mathbf{w}||}$$

- $\mathbf{w}$ and w_0 are now scaled so that the value $|g(\mathbf{x})|$ at the nearest points in both classes is equal to 1:

$$\begin{cases} \mathbf{w}^T\mathbf{x} + w_0 \geq 1 & \forall \mathbf{x} \in \omega_1 \\ \mathbf{w}^T\mathbf{x} + w_0 \leq -1 & \forall \mathbf{x} \in \omega_2 \end{cases}$$

- In this case, the margin is equal to

$$\frac{1}{||\mathbf{w}||} + \frac{1}{||\mathbf{w}||} = \frac{2}{||\mathbf{w}||}$$

SVMs for Linearly Separable Classes (5)

- In order to make the margin maximum, the following cost function has to be minimised

$$J(\mathbf{w}, w_0) = \frac{1}{2}||\mathbf{w}||^2$$

subject to

$$y_i(\mathbf{w}^{\mathsf{T}}\mathbf{x}_i + w_0) \geq 1; \quad \forall i = 1, 2, \ldots, N$$

SVMs for Linearly Separable Classes (6)

- Using the so called Lagrange function $\mathcal{L}(\mathbf{w}, w_0, \boldsymbol{\lambda})$ the Karush-Kuhn-Tucker (KKT) conditions have to be satisfied to minimise the cost function

(i) $\frac{\partial}{\partial \mathbf{w}} \mathcal{L}(\mathbf{w}, w_0, \boldsymbol{\lambda}) = \mathbf{0}$

(ii) $\frac{\partial}{\partial w_0} \mathcal{L}(\mathbf{w}, w_0, \boldsymbol{\lambda}) = 0$

(iii) $\lambda_i \geq 0; \quad \forall i = 1, \ldots, N$

(iv) $\lambda_i [y_i (\mathbf{w}^\mathsf{T} \mathbf{x}_i + w_0) - 1] = 0; \quad \forall i = 1, \ldots, N$

SVMs for Linearly Separable Classes (7)

- The Lagrange function itself is defined as

$$\mathcal{L}(\mathbf{w}, w_0, \boldsymbol{\lambda}) = \frac{1}{2}\mathbf{w}^{\mathsf{T}}\mathbf{w} - \sum_{i=1}^{N} \lambda_i [y_i(\mathbf{w}^{\mathsf{T}}\mathbf{x}_i + w_0) - 1]$$

- Applying the KKT criteria (i) and (ii) for the Lagrange function

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

$$\sum_{i=1}^{N} \lambda_i y_i = 0$$

SVMs for Linearly Separable Classes - Discussion

The Lagrange multipliers can be either zero or positive. Thus, the vector $\mathbf{w}$ of the optimal solution is a linear combination of $N_s \leq N$ feature vectors that are associated with $\lambda_i \neq 0$.

$$\mathbf{w} = \sum_{i=1}^{N_s} \lambda_i y_i \mathbf{x}_i$$

These are known as **support vectors** and the optimum hyperplane classifier as **support vector machine**.