Pattern Recognition

Lecture 4 : Support Vector Machine

Dr. Andreas Fischer

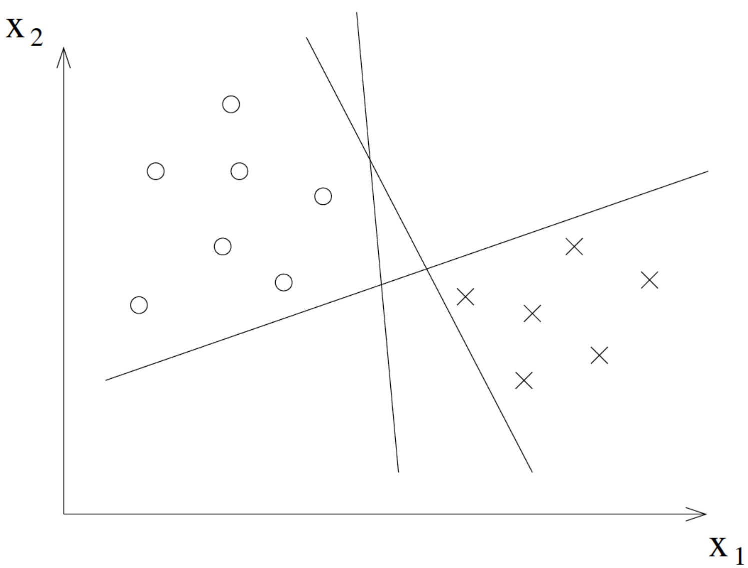
andreas.fischer@unifr.ch

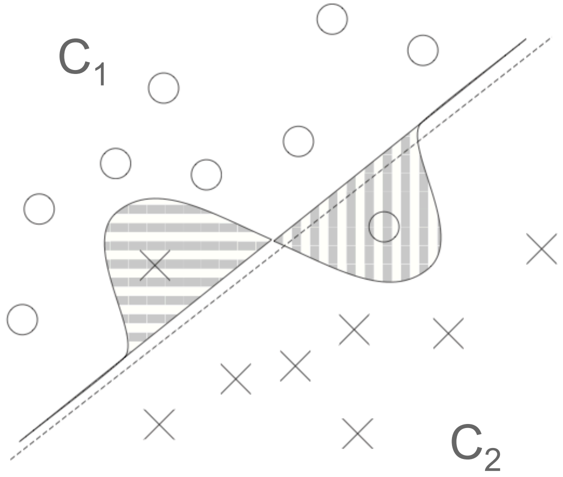
UNIVERSITÉ DE FRIBOURG
UNIVERSITÄT FREIBURG

MASTER IN COMPUTER SCIENCE

Support Vector Machine (SVM)

- Discriminative classifier
- Statistical representation: $X = R^n$
- SVM assumes linear boundaries and optimizes hyperplanes.
- In the following, we will distinguish the following cases:
  - Two linearly separable classes
  - Two non-linearly separable classes
  - Multi-Class SVM

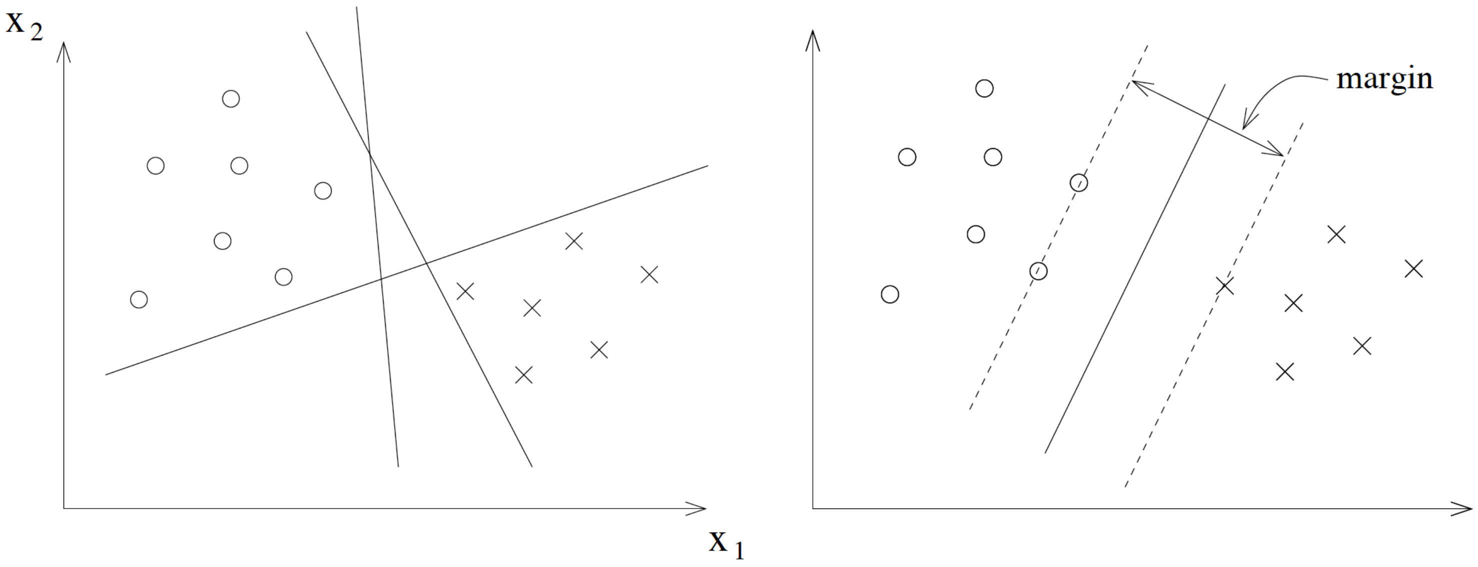




Two Linearly Separable Classes

Goal of SVM

- Consider two classes that are linearly separable.
- The goal of SVM is to find the hyperplane that:
  - Has the same distance to both classes
  - Maximizes the *margin*, that is the distance to both classes
- Link to Bayes classifier: hyperplane boundary was optimal for normal distribution and two classes with equal covariance.



Hyperplane Properties

- We consider a hyperplane in R^n:

$$w'x + b = \sum_{i=1}^{n} w_i x_i + b = 0$$

- Distance of the hyperplane to a vector x:

$$d_{(w,b)}(x) = \frac{|w'x + b|}{\|w\|}$$

- Two parameters (w,b):
  - w is the normal vector orthogonal to the hyperplane with length:

$$\|w\| = \sqrt{\sum_{i=0}^{n} w_i^2}$$

  - b corresponds with the distance of the hyperplane to the origin:

$$d_{(w,b)}(0) = \frac{|b|}{\|w\|}$$

Same Distance To Both Classes

- Training set S = $\{(x_1,y_1),\ldots,(x_N,y_N)\}$ with

$$y_i = \begin{cases} -1 \Leftrightarrow x_i \in C_1 \\ +1 \Leftrightarrow x_i \in C_2 \end{cases}$$

- Classify all samples correctly:

$$y_i(w' x_i + b) \geq 0$$

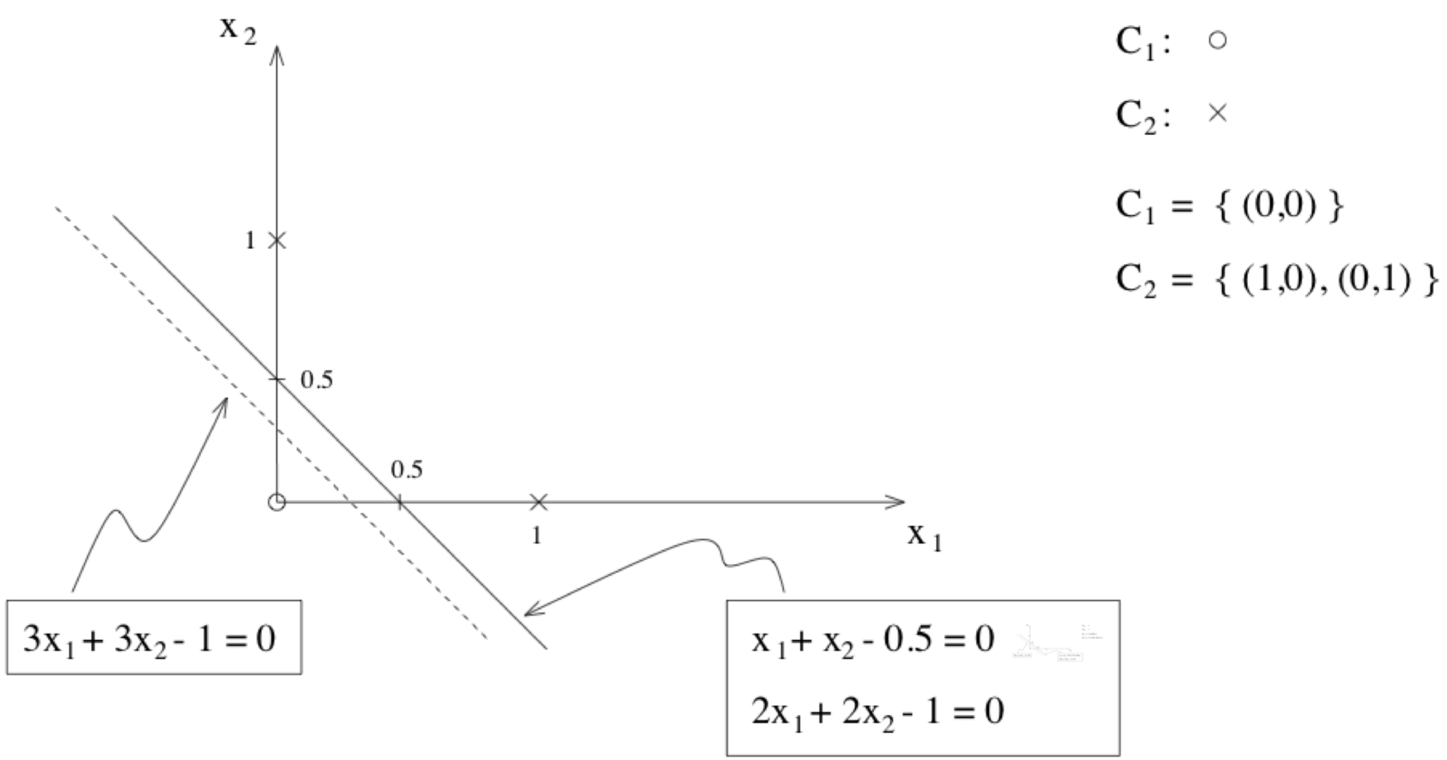
- Same distance to both classes in *canonical form*:

$$\min_{x_i \in C_1} \left| w' x_i + b \right| = \min_{x_i \in C_2} \left| w' x_i + b \right| = 1$$

$$\Rightarrow y_i(w' x_i + b) \geq 1$$

Example

- h_1: $x_1+x_2-0.5=0$ / h_2: $2x_1+2x_2-1=0$ / h_3: $3x_1+3x_2-1=0$
- Both h_1 and h_2 have the same distance from both classes. The distance to class C_1 (origin) is: $\frac{|b|}{\|w\|} = \frac{0.5}{\sqrt{2}} = \frac{1}{\sqrt{8}}$
- However, only h_2 is in canonical form.



Support Vectors and Margin

- Select vectors $x^{(-1)} \in C_1$ and $x^{(+1)} \in C_2$ with minimum distance of the hyperplane, so-called *support vectors*.
  - Support vectors $w'x^{(-1)} + b = -1$ lie in the hyperplane

$$h_1 : w' x^{(-1)} + (b+1) = 0$$

  - Support vectors $w'x^{(+1)} + b = 1$ lie in the hyperplane

$$h_2 : w' x^{(+1)} + (b-1) = 0$$

- The *margin* is the distance between h_1 and h_2. For example, the distance of h_2 to the support vector $x^{(-1)}$:

$$d_{(w,b-1)}(x^{(-1)}) = \frac{\left|wx^{(-1)} + b - 1\right|}{\|w\|} = \frac{|-1-1|}{\|w\|} = \frac{2}{\|w\|}$$

Maximum Margin Hyperplane

- Accordingly, the problem of SVM can be stated as:
  - Find the hyperplane (w,b) that maximizes the margin

$$\frac{2}{\|w\|}$$

  - Under the condition that, for $1 \le i \le N$:

$$y_i(w' x_i + b) \ge 1$$

- Can be solved, for example, by means of quadratic programming (not shown in this lecture). Let $x^{(-1)}$ and $x^{(+1)}$ be arbitrary support vectors. Then the optimal parameters (w*,b*) are:

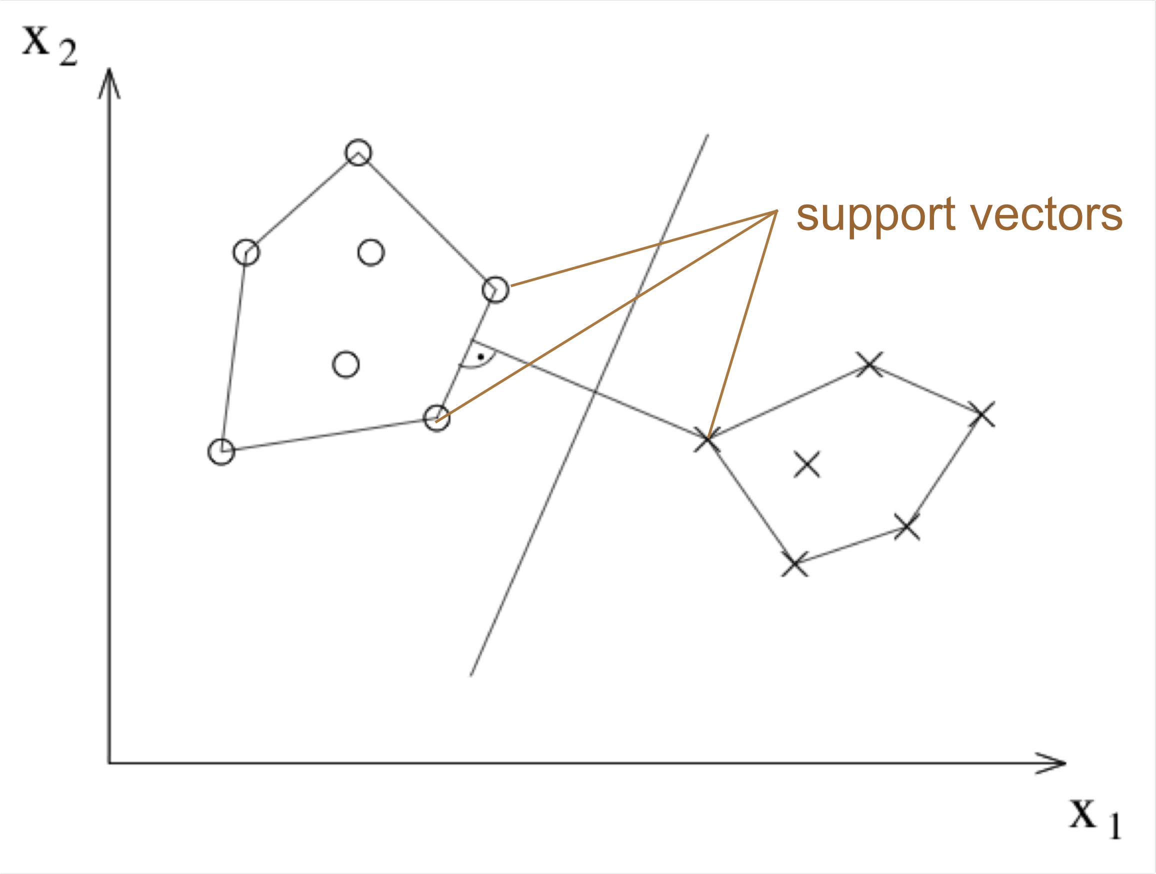
$$w^* = \sum_{i=1}^{N} \alpha_i y_i x_i$$

$$b^* = -\frac{1}{2} w^* \left(x^{(-1)} + x^{(+1)} \right)$$

- The coefficients $\alpha_i \ge 0$ found by the optimization method are non-negative Lagrange multipliers with $\sum \alpha_i y_i = 0$. They are non-zero only for support vectors, hence only support vectors are relevant for the solution.

Geometric Interpretation

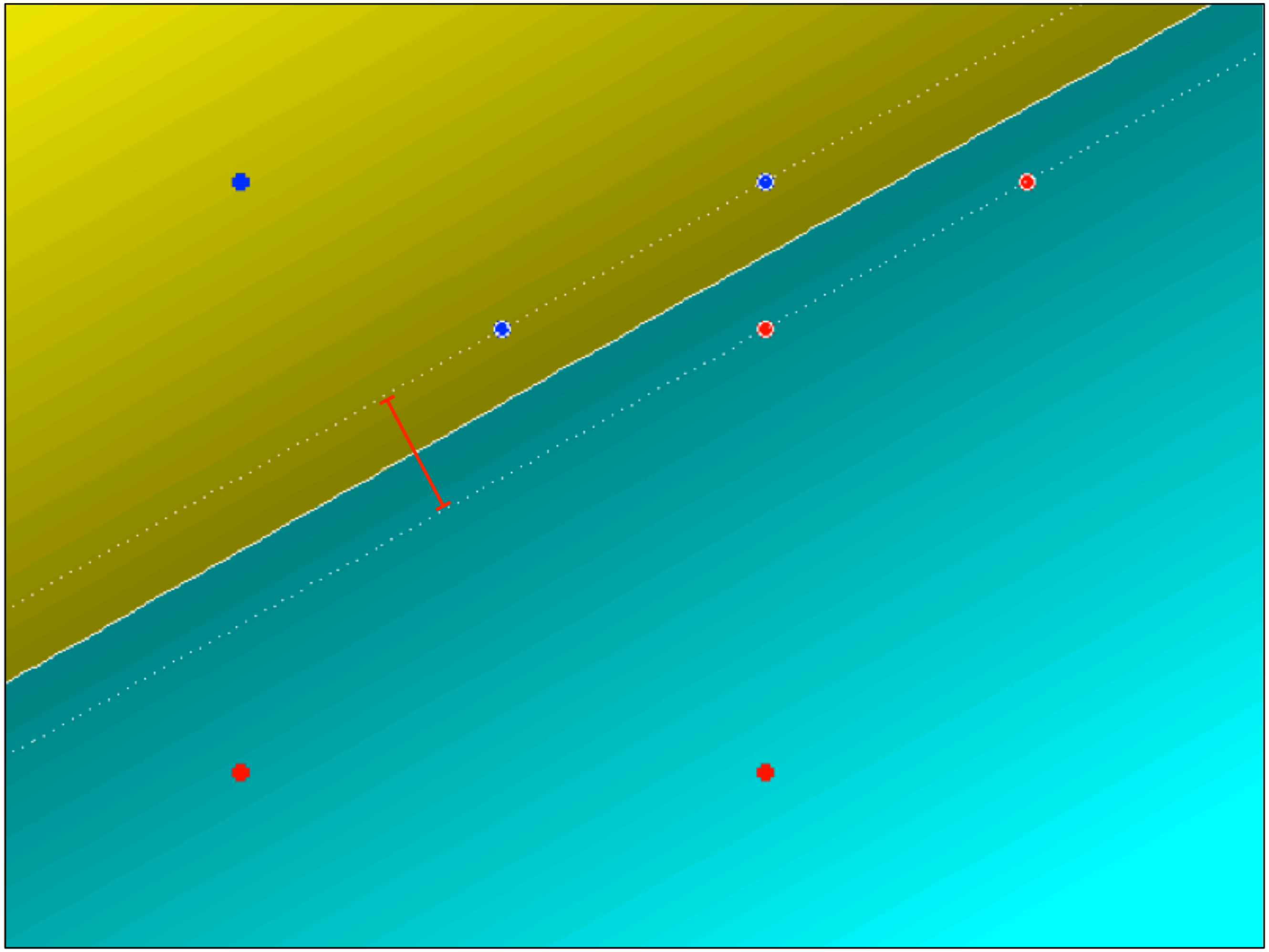
- Find the convex hull for both classes.
- Find a straight segment with minimum length between the convex hulls.
- The maximum margin hyperplane cuts this segment in the middle and is orthogonal to it.



Example

- 4 support vectors (57%). The same result would be obtained when removing all non-support vectors from the training set.

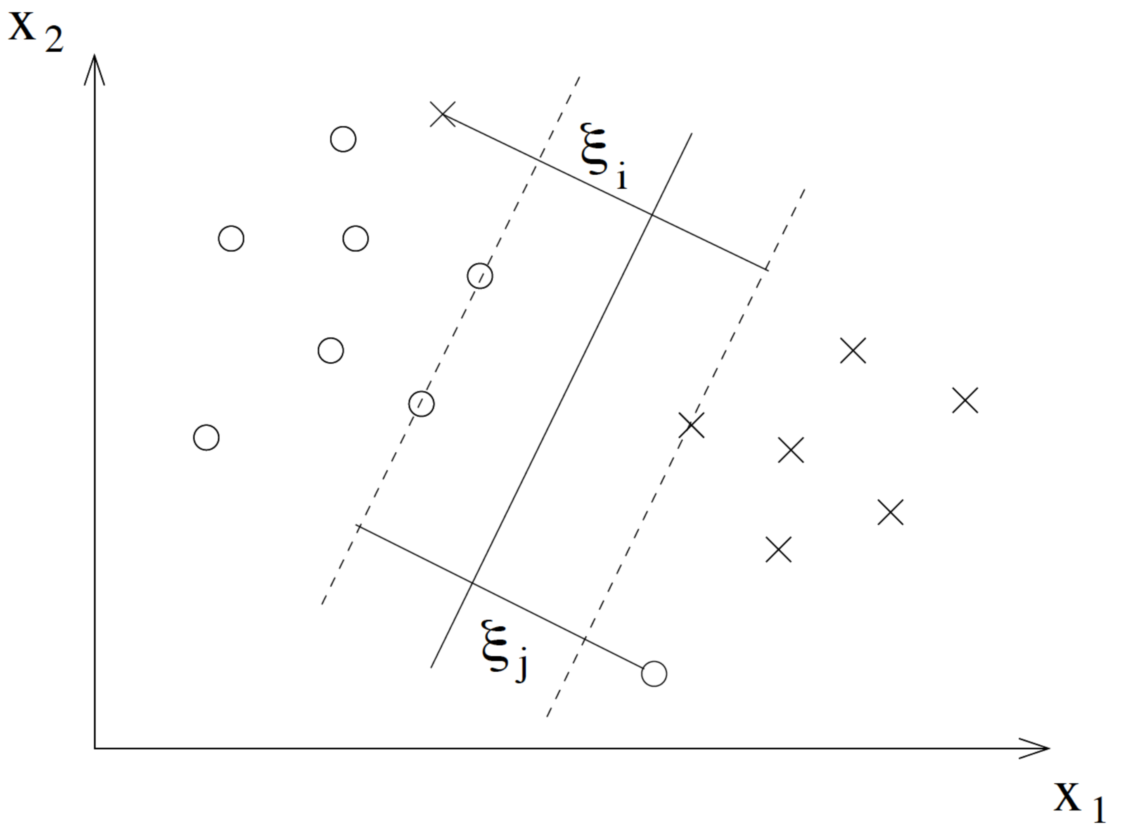
x_1	x_2	y
1	1	-1
3	3	1
1	3	1
3	1	-1
2	2.5	1
3	2.5	-1
4	3	-1



Two Non-Linearly Separable Classes

Slack Variables

- Introduce so-called slack variables $\xi_i \geq 0$ (i=1,…,N) that correspond to the misclassification error.
- $\xi_i > 1$ if the sample x_i of the training set is misclassified.



General SVM

- In this general case, the problem of SVM can be stated as:
  - Find the hyperplane (w,b) that minimizes

$$\frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N}\xi_i$$

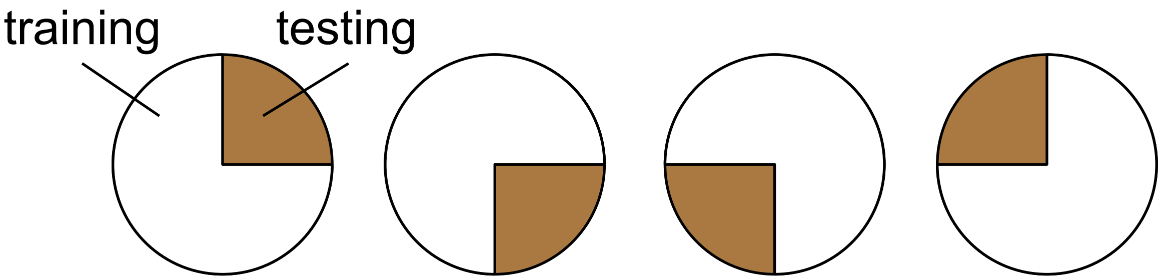
  - Under the condition that, for $1 \leq i \leq N$:

$$y_i(w'x_i + b) \geq 1 - \xi_i$$

- It can be solved with the same optimization methods.
  - Minimization of $||w||^2 / 2$ corresponds to maximization of the margin.
  - Additionally, the classification error $\sum\xi_i$ is minimized.
- The parameter $C \geq 0$ balances the two criteria. It is often optimized experimentally with respect to the classification accuracy achieved on independent validation samples.

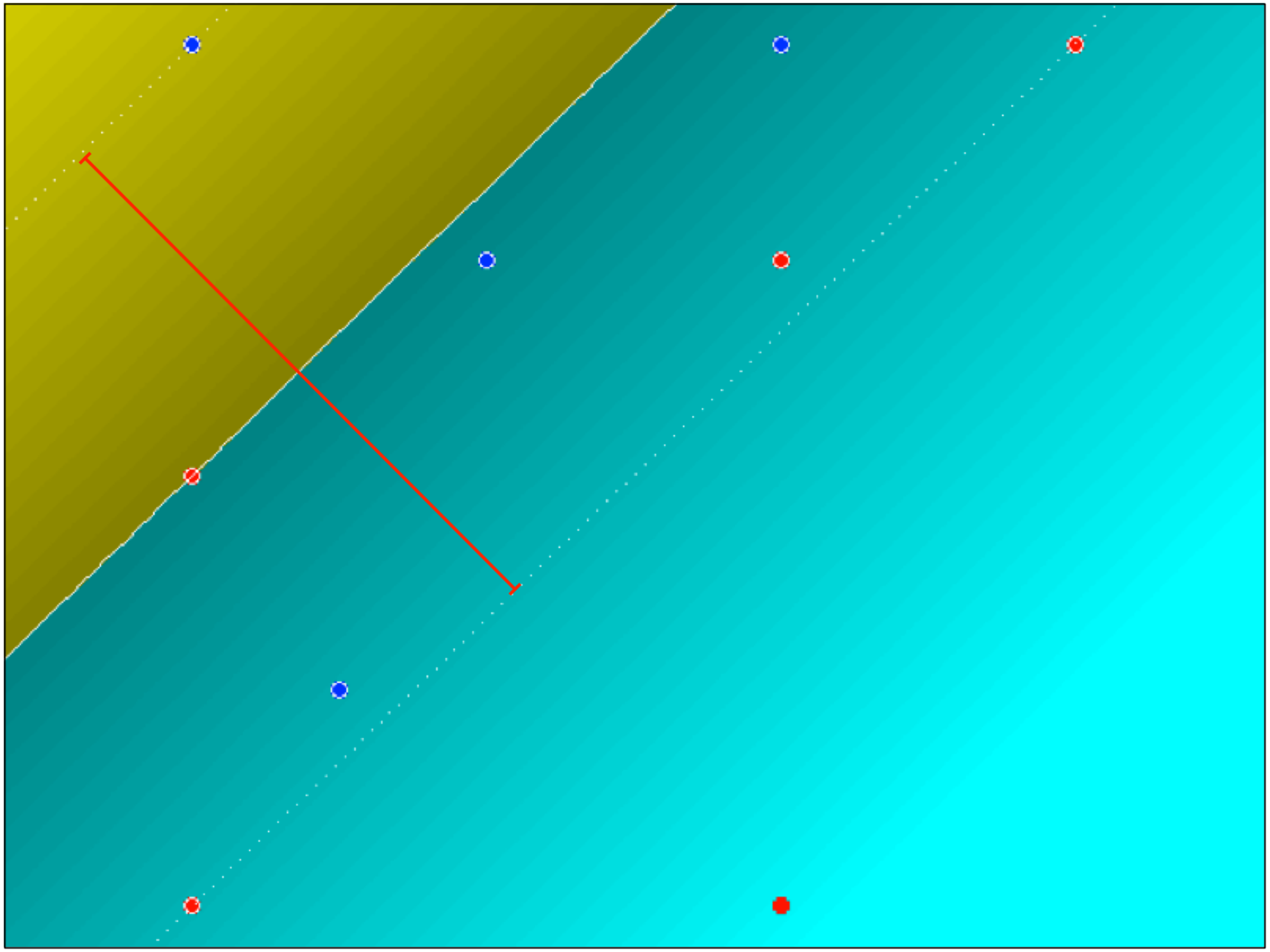
Cross-Validation

- To avoid an overfitting to the training samples, which reduces the generalization capability of a classifier, cross-validation is often used to optimize system parameters such as K for KNN and C for SVM.
- *K-fold cross-validation*:
  - Split the training samples into K independent parts and use each part once for testing; compute the average accuracy.
  - Experiment with different values for the system parameter and choose the one that achieves the best average accuracy.
  - *Leave-one-out* method: K equals the number of training samples. This method is particularly interesting for small data sets.



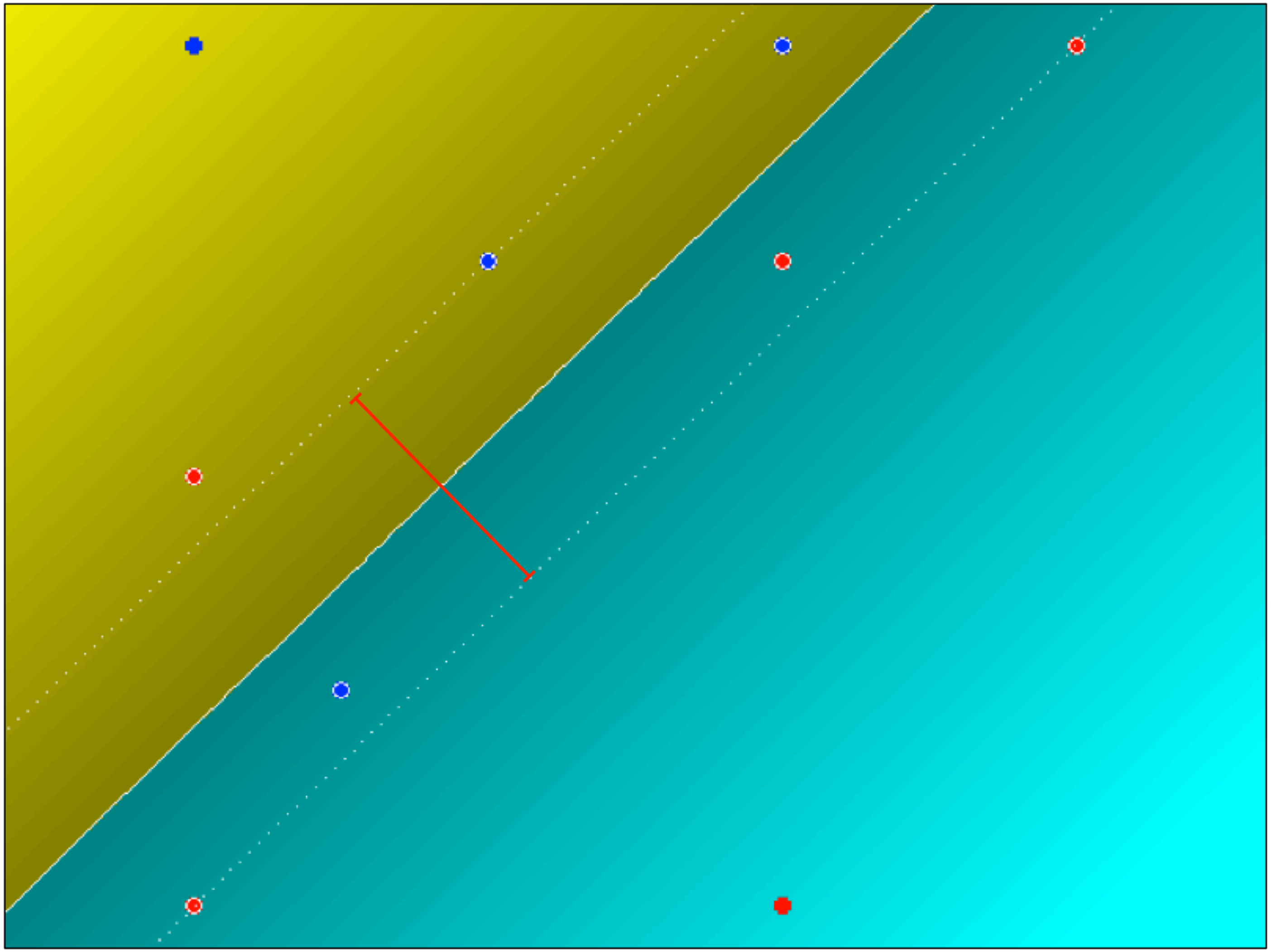
Example 1

- Parameter C=1: 3 misclassifications.



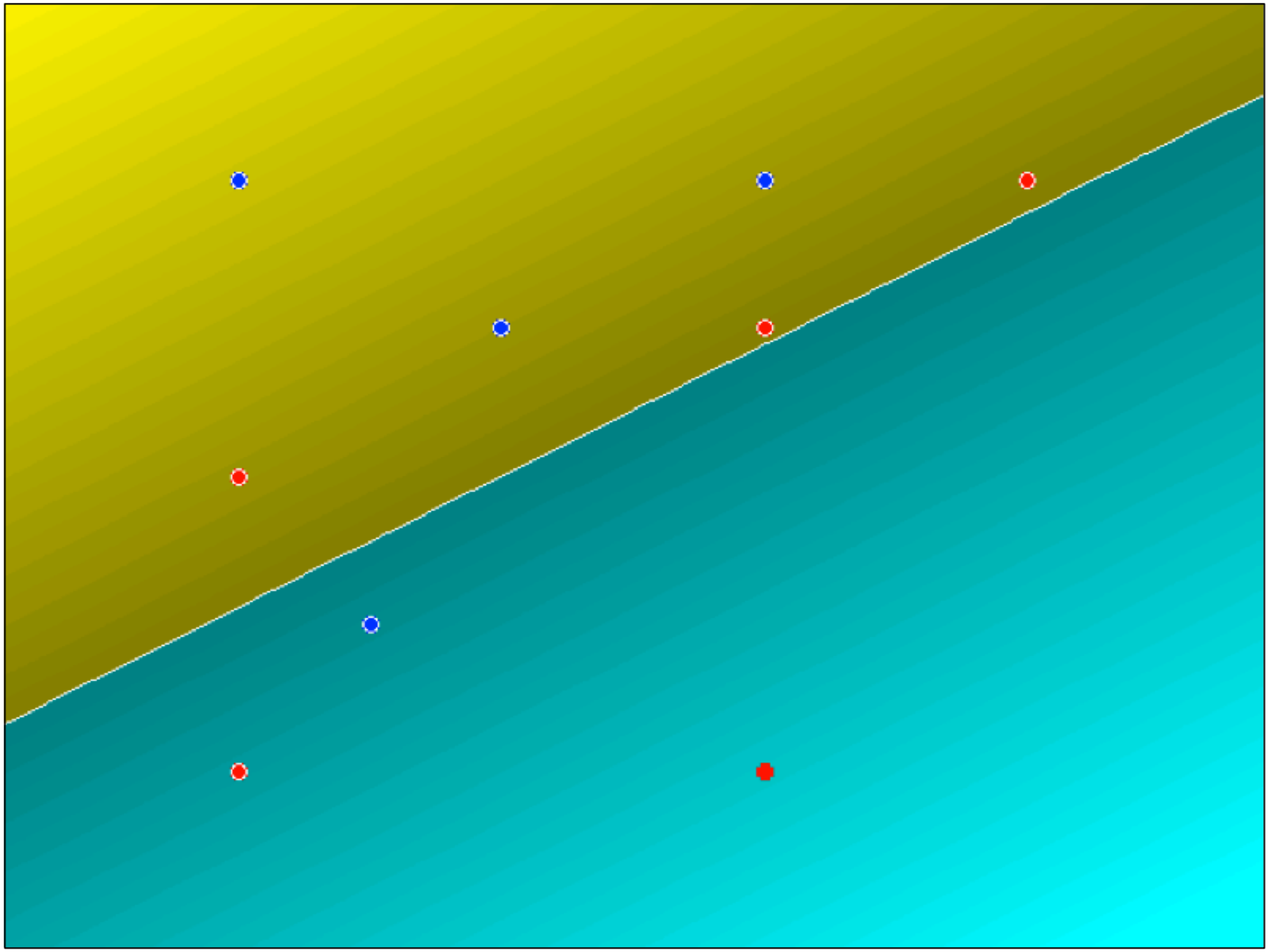
Example 2

- Parameter $C=10^5$: 2 misclassifications.



Example 3

- Parameter $C=10^{-8}$: 4 misclassifications.



Multi-Class SVM

One vs One

- Each pair of classes is separated by means of a hyperplane. That is, compute m(m-1) / 2 hyperplanes $d_{ij}(x)$ such that

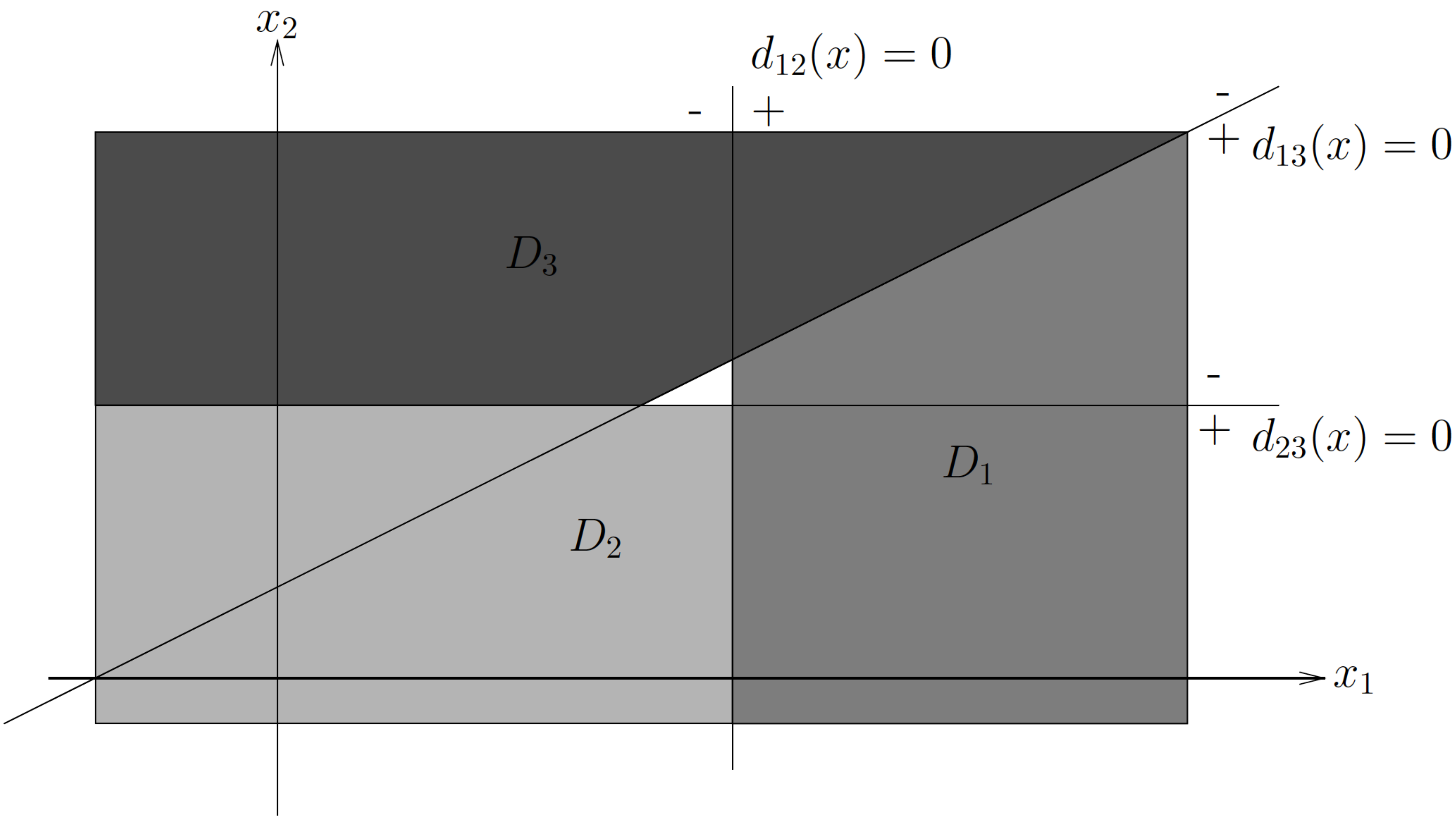
$$x \in \begin{cases} C_i \Leftrightarrow d_{ij}(x) > 0 \\ C_j \Leftrightarrow d_{ij}(x) < 0 \end{cases}$$

- The classification rule is:

$$x \in C_i \Leftrightarrow d_{ij}(x) > 0 \text{ for all } j = 1, \ldots, m;\ j \neq i$$

- However, there are regions that are not assigned to a class:
  - If $d_{ij}(x) > 0$ for some but not for all other classes.
- A possible resolution is to select the class with the most votes among the m(m-1) / 2 decisions.

Example



One vs All (with Rejection)

- Separate each class from all others. That is, compute m hyperplanes $d_i(x)$ such that:

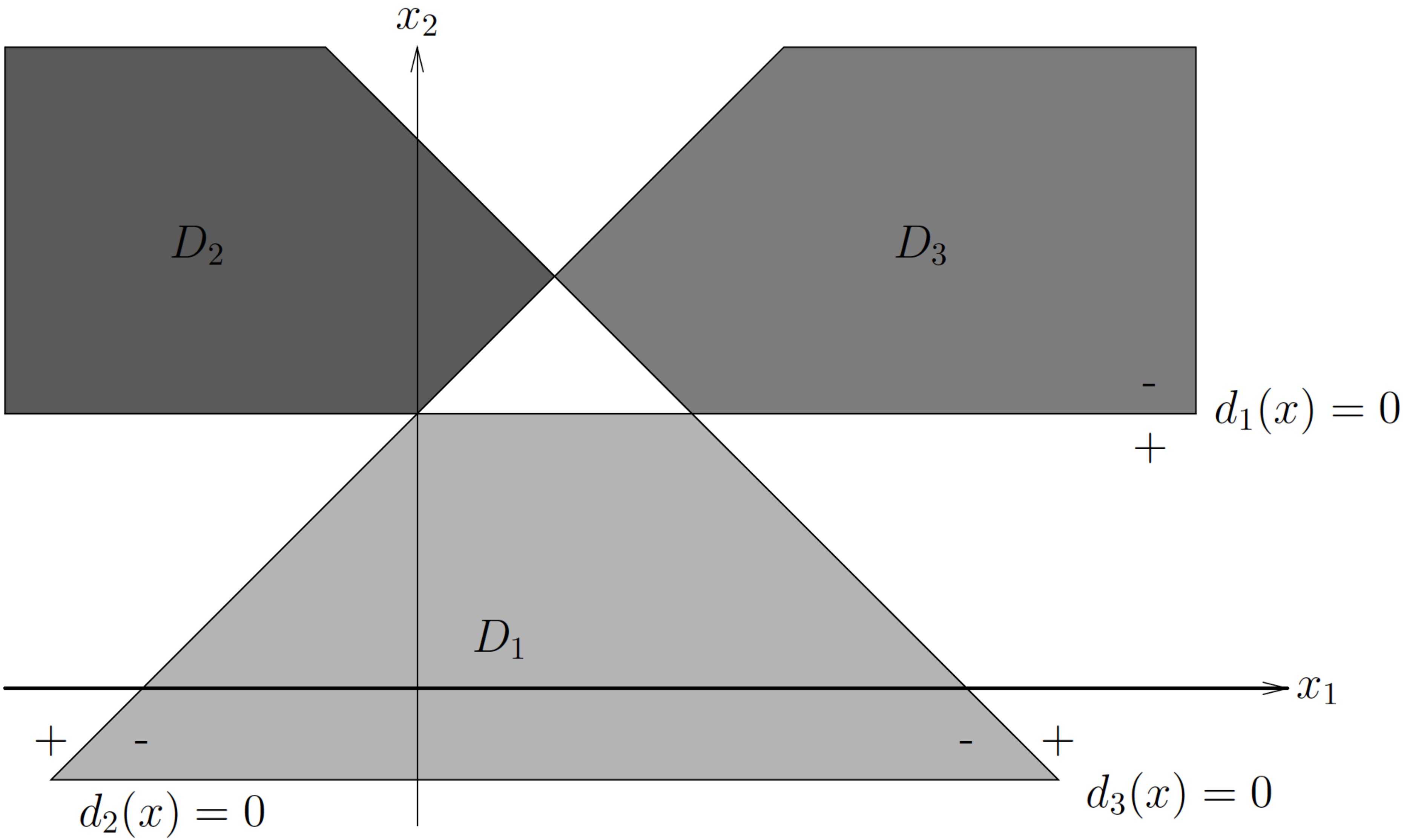
$$x \in \begin{cases} C_i \Leftrightarrow d_i(x) > 0 \\ \bar{C}_i \Leftrightarrow d_i(x) < 0;\ \bar{C}_i = \{C_1, \ldots, C_m\} \setminus C_i \end{cases}$$

- A possible classification rule is:

$$x \in C_i \Leftrightarrow d_i(x) > 0 \text{ and } d_j(x) < 0 \text{ for all } j = 1, \ldots, m;\ j \neq i$$

- However, there are regions that are not assigned to a class:
  - If $d_i(x) < 0$ for all classes.
  - If $d_i(x) > 0$ for more than one class.

Example



One vs All (without Rejection)

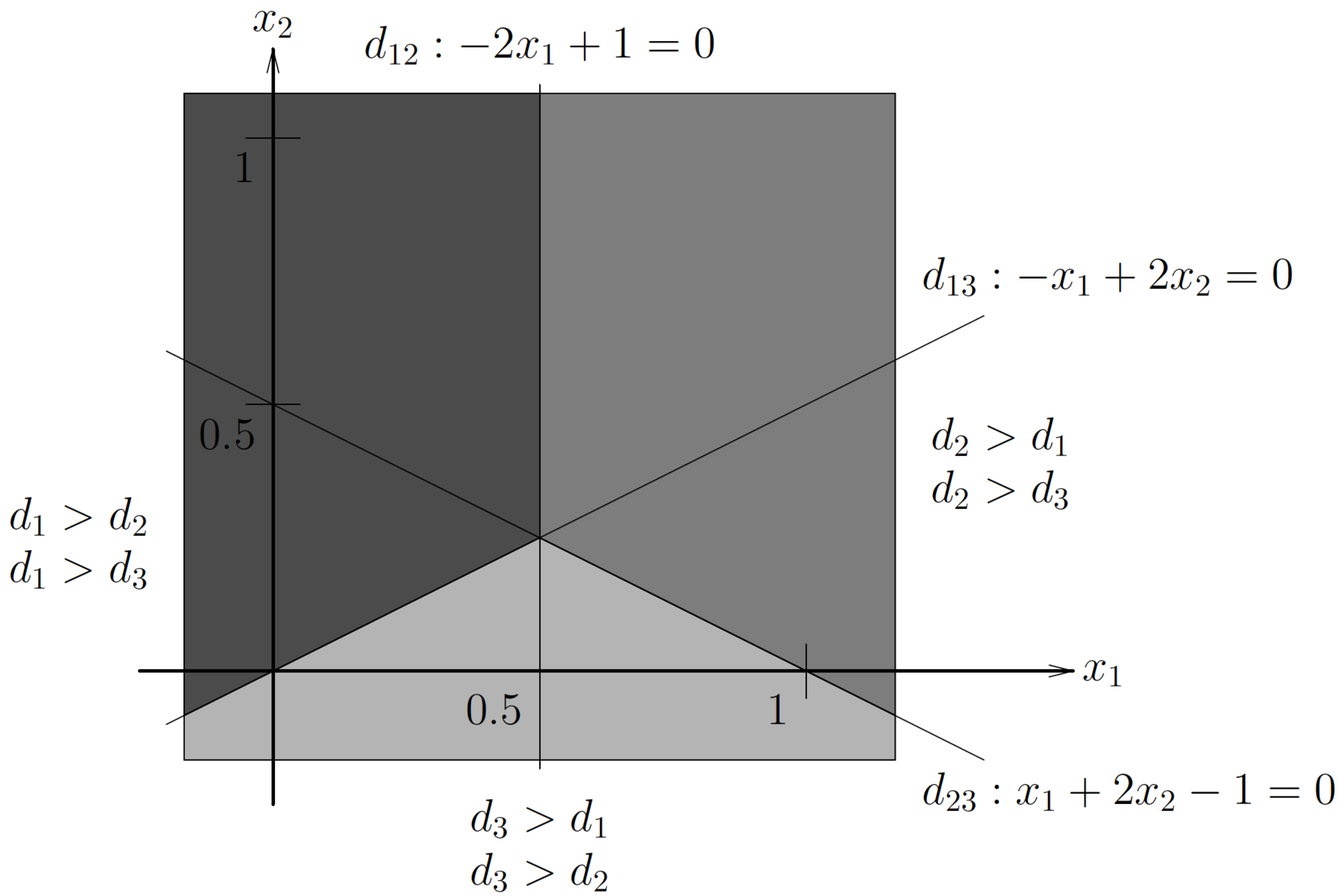
- A unique assignment is obtained with the maximum rule:

$$x \in C_i \Leftrightarrow d_i(x) > d_j(x) \text{ for all } j = 1,\ldots,m;\, j \neq i$$

- This decision rule is a special case of the One vs One rule with

$$d_{ij}(x) = d_i(x) - d_j(x)$$

Example



x_2
$d_{12} : -2x_1 + 1 = 0$
1
0.5
$d_{13} : -x_1 + 2x_2 = 0$
$d_2 > d_1$
$d_2 > d_3$
$d_1 > d_2$
$d_1 > d_3$
x_1
0.5
1
$d_{23} : x_1 + 2x_2 - 1 = 0$
$d_3 > d_1$
$d_3 > d_2$

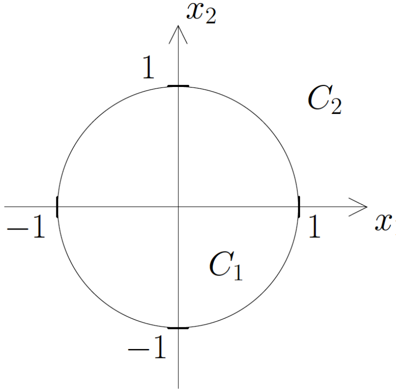
Kernel SVM

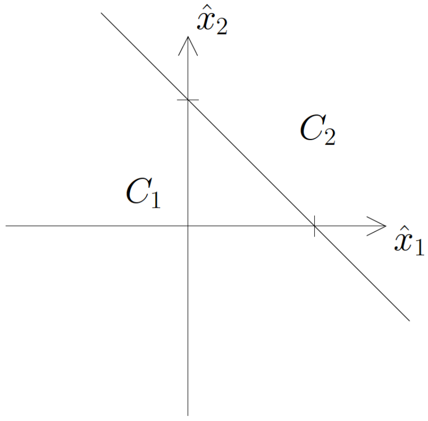
Brief Introduction to Kernel Methods

- For a comprehensive introduction, see for example:
  J. Shawe-Taylor and N. Cristianini. Kernel Methods for Pattern Analysis. Cambridge University Press, 2004.
- Idea: Problem might be simpler to solve in a different, possibly higher dimensional feature space.
- Example in R^2:

$$d(x) = -x_1^2 - x_2^2 + 1 = 0$$

$$x \in \begin{cases} C_1 \Leftrightarrow d(x) \geq 0 \\ C_2 \Leftrightarrow d(x) < 0 \end{cases}$$





- Map the features $\varphi : R^2 \to R^2$

$$\varphi(x) = (x_1^2, x_2^2)$$

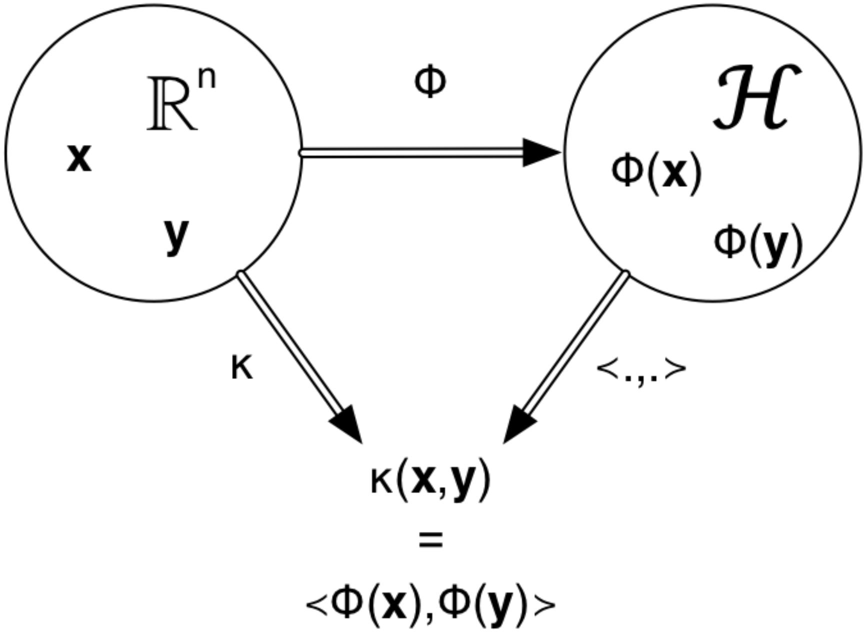
$$\hat{d}(x) = -\hat{x}_1 - \hat{x}_2 + 1 = 0$$

- In the new feature space, the classes are linearly separable. Note that in this example the dimension of the new feature space is still 2.

Kernel Trick

- Avoid an explicit, possibly costly mapping into the new feature space.
- Instead, calculate only the dot product:

$$\kappa(x, y) = \langle \varphi(x), \varphi(y) \rangle$$



Kernelization

- Theorem: For every *valid* kernel function such a feature space exists.
- Replace standard dot product with any valid kernel to solve the problem implicitly in a different feature space.
- Algorithms that can be expressed in terms of dot products only are called *kernelizable*.
- KNN is kernelizable:

$$\left\|\varphi(x)-\varphi(y)\right\|^2 = \left\langle \varphi(x)-\varphi(y), \varphi(x)-\varphi(y) \right\rangle$$

$$= \left\langle \varphi(x), \varphi(x) \right\rangle + \left\langle \varphi(y), \varphi(y) \right\rangle - 2\left\langle \varphi(x), \varphi(y) \right\rangle$$

$$= \kappa(x,x) + \kappa(y,y) - 2\kappa(x,y)$$

Kernel SVM

- SVM is kernelizable:

$$w\varphi(x)+b=\left(\sum_{i=1}^{N}\alpha_i y_i \varphi(x_i)\right)\varphi(x)+b$$

$$=\sum_{i=1}^{N}\alpha_i y_i \left\langle \varphi(x_i),\varphi(x)\right\rangle + b$$

$$=\sum_{i=1}^{N}\alpha_i y_i \kappa(x_i,x)+b$$

Common Kernels

- Common kernels include:

$$\kappa(x,y) = \langle x,y \rangle \text{ linear kernel}$$

$$\kappa(x,y) = \left(\gamma\langle x,y \rangle + r\right)^d, \gamma > 0 \text{ polynomial kernel}$$

$$\kappa(x,y) = \exp\left(-\gamma\left\|x-y\right\|^2\right), \gamma > 0 \text{ RBF (Gaussian) kernel}$$

$$\kappa(x,y) = \tanh\left(\gamma\langle x,y \rangle + r\right) \text{ sigmoid kernel}$$

- One of the most frequently used kernels for SVM is the radial basis function (RBF) kernel. Note that the kernel parameter $\gamma > 0$ has to be optimized carefully together with the SVM parameter C.
- Fundamental property of non-linear kernel SVM: The linear class boundary in the implicit feature space H corresponds with a non-linear class boundary in the original feature space R^n.

Example