

# Lecture 10: Support Vector Machines

## Cognitive Systems - Machine Learning

### **Part II: Special Aspects of Concept Learning**

**SVM, kernel trick, linear separability**

last change: 2. Dezember 2014

# Motivation

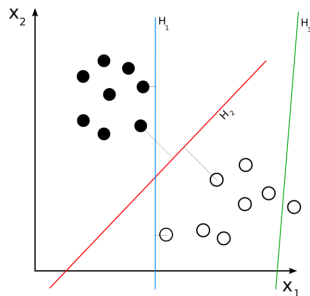
- linear classifiers (e.g. perceptrons): efficiently trainable, but low capacity and only for non-symbolic instances
- non-linear classifiers (e.g. NNs with hidden layers): high capacity, but high time complexity, local optima, overfitting, only numerical data
- idea of kernel methods: non-linearly embed instance space in high dimensional feature space with dot-product where mapped training data is linearly separable
  - high capacity
  - applicable for symbolical data
  - time efficient training (polynomial with sample size)
  - global optimum
  - prevents overfitting

# Outline

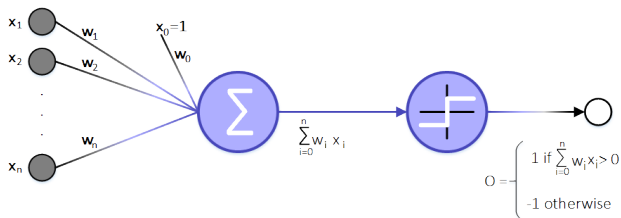
- Perceptron Revised
- Support Vector Machines
- Soft Margin
- The Kernel Trick
- Parameter Optimization
- Class Learning

# Perceptrons' Representational Power

- a perceptron represents a **hyperplane decision surface** in the  $n$ -dimensional space of instances
- some sets of examples cannot be separated by any hyperplane, those that can be separated are called **linearly separable**

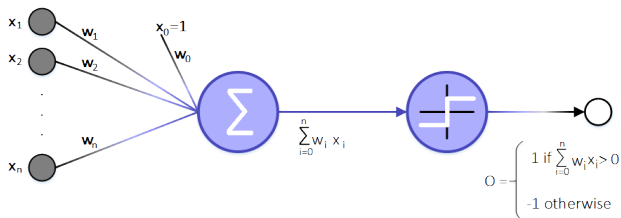


# Perceptrons Revised



- takes a vector of real-valued inputs  $(x_1, \dots, x_n)$  weighted with  $(w_1, \dots, w_n)$
- calculates the linear combination of these inputs
  - $\sum_{i=0}^n w_i x_i = w_0 x_0 + w_1 x_1 + \dots + w_n x_n$
  - $-w_0$  denotes a threshold value, i.e. that value which must be reached by the linear combination of inputs to cause the perceptron to output 1
  - $x_0$  is always 1
- outputs **1** if the result is greater than 0, otherwise **-1**

# Perceptrons Revised



- takes a vector of real-valued inputs  $(x_1, \dots, x_n)$  weighted with  $(w_1, \dots, w_n)$
- calculates the linear combination of these inputs
  - $\sum_{i=0}^n w_i x_i = w_0 x_0 + w_1 x_1 + \dots + w_n x_n$
  - $-w_0$  denotes a threshold value, i.e. that value which must be reached by the linear combination of inputs to cause the perceptron to output 1
  - $x_0$  is always 1
- outputs 1 if the result is greater than 0, otherwise -1

# Alternate Representations

## Vector Representation

$$o(\mathbf{x}) = \begin{cases} 1 & \text{if } w\mathbf{x} - b > 0 \\ -1 & \text{otherwise} \end{cases}$$

## Constraint Representation

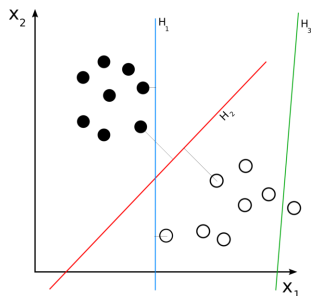
$w\mathbf{x}_i - b > 0$  for positive instances  $\mathbf{x}_i$

$w\mathbf{x}_i - b < 0$  for negative instances  $\mathbf{x}_i$

## Unified Constraint Representation

$$c(\mathbf{x}_i)(w\mathbf{x}_i - b) > 0$$

## Initial Idea (Vladimir Vapnik)

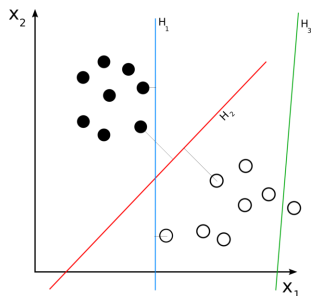


## Hypothesis' Language

$$c(\mathbf{x}_i) \cdot (w\mathbf{x}_i - b) > 0$$



## Initial Idea (Vladimir Vapnik)



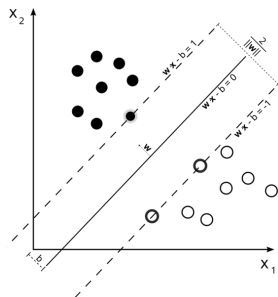
## Hypothesis' Language

$$c(\mathbf{x}_i) \cdot (w\mathbf{x}_i - b) > 0$$

## Issue

The hyperplane not unique!

# Transforming a Hyperplane

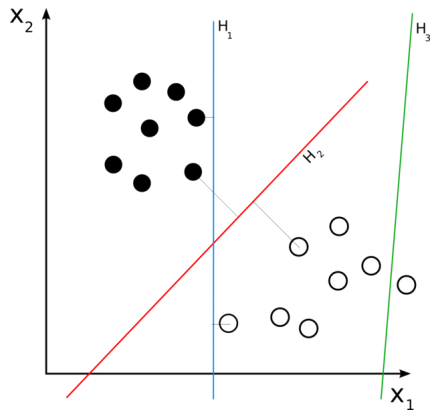


- moving parallel:  

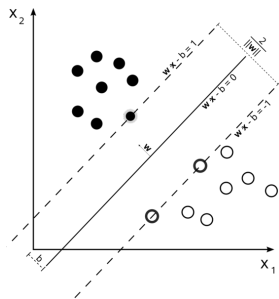
$$c(\mathbf{x}_i) \cdot (w\mathbf{x}_i - (b + \nu)) = 0$$
- scaling:  

$$\beta \cdot c(\mathbf{x}_i) \cdot (w\mathbf{x}_i - b) = 0$$

# The Best Hyperplane



# The Best Hyperplane



## Best Hyperplane

The hyperplane separating all instances while being as far apart from the instances as possible.

## Corridor

Define a Hyperplane Corridor such that  $|(w\mathbf{x}_i) - b| = 1$  for the  $\mathbf{x}_i$  closest to the hyperplane. The distance between the two hyperplanes is  $\frac{2}{\|w\|}$ .

# Initial Idea (cont'd)

## Support Vector Machine

A hyperplane  $w\mathbf{x}_i - b$  such that

- $\frac{2}{\|w\|}$  is maximal and
- $c(\mathbf{x}_i)(w\mathbf{x}_i - b) \geq 1$  for all instances  $\mathbf{x}_i$ .

## Learning a SVM

- Swarm Algorithms
- Genetic Algorithms
- Analytic Solutions

# Alternate Problem Formulation

## Idee

Minimize  $\frac{1}{2} \|w\|^2$  such that  $c(\mathbf{x}_i)(w\mathbf{x}_i - b) \geq 1$  for all  $\mathbf{x}_i$ :

$$\frac{1}{2} \|w\|^2 - \sum_i \mu_i (1 - c(\mathbf{x}_i)(w\mathbf{x}_i - b))$$

## Support Vectors

The solution can be expressed as

$$w = \sum_{i=0}^n \alpha_i c(\mathbf{x}_i) \mathbf{x}_i$$

Only few  $\alpha_i$  are not zero. The corresponding  $\mathbf{x}_i$  are called the support vectors. The support vectors satisfy  $c(\mathbf{x}_i)(w\mathbf{x}_i - b) = 1$ , thus lie on the two parallel hyperplanes.

# Soft Margins (Corinna Cortes and Vladimir Vapnik)

- data may contain noise
- constraint is relaxed:  
 $c(\mathbf{x}_i)(w\mathbf{x}_i - b) \geq 1 - \xi_i$
- relaxation is penalized
- $\xi_i$  are called slack variables

# Soft Margins (Corinna Cortes and Vladimir Vapnik)

- data may contain noise
- constraint is relaxed:  
 $c(\mathbf{x}_i)(w\mathbf{x}_i - b) \geq 1 - \xi_i$
- relaxation is penalized
- $\xi_i$  are called slack variables

## Soft Margin SVM

A hyperplane  $w\mathbf{x}_i - b$  such that

- $\frac{1}{2} \|w\|^2 - C \sum_{i=0}^n \xi_i$  is minimal and
- $c(\mathbf{x}_i)(w\mathbf{x}_i - b) \geq 1 - \xi_i$  for all  $\mathbf{x}_i$ .

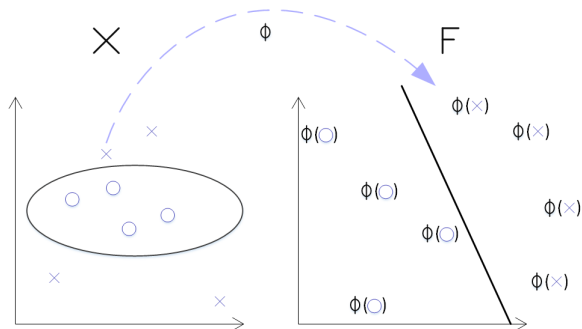


# The Kernel Trick

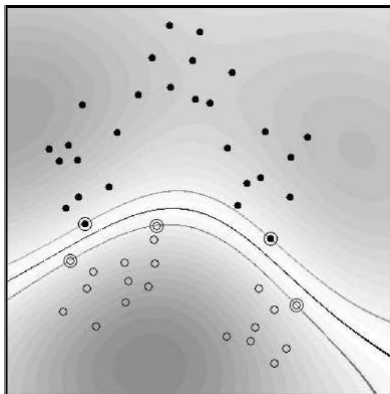
(Bernhard Boser, Isabelle Guyon and Vladimir Vapnik)

- the data may not be linearly separable at all
- the solution is to transform the feature space:  $\mathbf{x}_i \rightarrow \Phi(\mathbf{x}_i)$

- for example  $\Phi \left( \begin{pmatrix} a \\ b \end{pmatrix} \right) = \begin{pmatrix} a^2 \\ 2ab \\ b^2 \end{pmatrix}$



# The Kernel Trick (cont'd)



- as every vector occurs only inside a dot product it is not necessary to give  $\Phi$  explicitly
- every dot-product is replaced by an application of a non-linear kernel function  $\mathcal{K}(x_i, x_j)$

# The Kernel Trick (cont'd)

## Common Kernels

- Polynomial:  $\mathcal{K}(x_i, x_j) = (x_i \cdot x_j)^d$
- Gaussian Radial Basis Function (RBF):  $\mathcal{K}(x_i, x_j) = e^{-\gamma \|x_i - x_j\|^2}$
- Hyperbolic Tangent:  $\mathcal{K}(x_i, x_j) = \tanh(\kappa x_i \cdot x_j + c)$

## Kernel SVM

A hyperplane  $\Phi(w)\Phi(\mathbf{x}_i) - b$  such that

- $\frac{1}{2} \|w\|^2$  is minimal
- $c(\mathbf{x}_i)(\Phi(w) \cdot \Phi(\mathbf{x}_i) - b) \geq 1$  for all  $\mathbf{x}_i$

Or, equivalently  $c(\mathbf{x}_i)(\mathcal{K}(w, \mathbf{x}_i) - b) \geq 1$  for all  $\mathbf{x}_i$

# Parameter Optimization

- free parameters are:
  - the cost factor  $C$ : the higher the more accurate instances are classified (during training)
  - the kernel function  $\mathcal{K}$
  - the parameters of the kernel
- normally powers of 2 are used for  $C$ :  $C \in \{2^{-5}, \dots, 2^{15}\}$
- the same holds for  $\gamma$  when using a Gaussian RBF:  $\gamma \in \{2^{-15}, \dots, 2^3\}$

# Class Learning

- support vector machines are designed for concept learning
- there is ongoing research how to handle class learning
- a single SVM approach (called multi-class SVM) tries to solve the optimization problem directly
- other approaches use more than one SVM in a “divide and conquer” manner
  - 1-against-1,
  - 1-against-all,
  - error correcting output codes (ECOC)

# Divide and Conquer

## 1-against-1

- for each pair of classes one SVM is learned
- only examples of the two classes are used
- final classification is assigned by vote

## 1-against-all

- for each class one SVM is trained
- the concept is whether the example belongs to the class or not
- the class of the SVM with the highest output ( $w\mathbf{x} - b$ ) is assigned as final classification

A similar meta learner is available (e.g. for perceptrons).

# Divide and Conquer (cont'd)

## ECOCs

- for each class a bit-string  $s_j$  is created
- every bit-string is of length  $L$
- the Hamming-Distance between all bit-strings is maximized
- $L \leq 2^{(k-1)} - 1$  for  $k$  classes
- for each bit an SVM is trained
- final classification is assigned to the class with the closest bit-string

# Learning Terminology

## Support Vector Machine

<b>Supervised Learning</b>	unsupervised learning
----------------------------	-----------------------

Approaches:

<b>Concept / Classification</b>	Policy Learning
symbolic	<b>statistical</b> / neuronal network
<b>inductive</b>	analytical

Learning Strategy: Data: Target Values:	<b>learning from examples</b> <b>metric features</b> /any (with the right kernel) <b>concept</b>
---	--