Lecture 10: Kernel Methods / Support Vector Machines

Cognitive Systems II - Machine Learning

WS 2005/2006

Part II: Special Aspects of Concept Learning

Support Vector Machines, optimal canonical hyperplane, kernel trick

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 seperable
  - high capacity
  - applicable for symbolical data
  - time efficient training (polynomial with sample size)
  - global optimum
  - prevents overfitting

Support Vector Machines – Overview

- Kernel: $\mathcal{K}(x, x') = (\Phi(x) \cdot \Phi(x'))$ where $x, x' \in X, \Phi : X \to H$, H feature space
  - problem specific!
- SVM-Training:
  - finds separating hyperplane with maximal margin in H
  - constructs hyperplane based on kernel function in X
    $\rightsquigarrow$ kernel trick

Optimal Separating Hyperplane

- hyperplane: $(w \cdot x) + b = 0 \quad w \in \mathbb{R}^N, b \in \mathbb{R}$
  corresponding decision function: $f(x) = \text{sgn}((w \cdot x) + b)$
- *optimal* hyperplane (maximal margin):
  $\max_{w,b} \min_{i=1,\ldots,m} \{\|x - x_i\| : x \in \mathbb{R}^N, (w \cdot x) + b = 0\}$
- with $y_i \in \{-1, +1\}$ holds: $y_i \cdot ((w \cdot x_i) + b) > 0$ for all $i = 1, \ldots, m$
  - w and b not unique!
  - w and b can be scaled, so that $|(w \cdot x_i) + b| = 1$ for the x_i closest to the hyperplane
    $\rightsquigarrow$ *canonical* form (w, b) of hyperplane, now holds:
    $y_i \cdot ((w \cdot x_i) + b) \geq 1$ for all $i = 1, \ldots, m$
- margin of optimal hyperplane in canonical form equals $\frac{2}{\|w\|}$

Constructing Optimal Hyperplane

- minimize $\frac{1}{2}\|w\|^2$ subject to $y_i \cdot ((w \cdot x_i) + b) \geq 1, \quad i = 1, \ldots, m$
- dealt with by introduce Langrangian:

$$L(w, b, \alpha) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{m} \alpha_i (y_i \cdot ((w \cdot x_i) + b) - 1)$$

  - L has to be minimized w.r.t. the *primal variables* w and b and maximized w.r.t. the *dual variables* α_i
  - conditions $\frac{\delta}{\delta b} L(w, b, \alpha) = 0$ and $\frac{\delta}{\delta w} L(w, b, \alpha) = 0$ lead to

$$\sum_{i=1}^{m} \alpha_i y_i = 0 \quad \text{and} \quad w = \sum_{i=1}^{m} \alpha_i y_i x_i$$

  - $\alpha_i = 0$ for all i except of those with x_i closest to optimal hyperplane (lying on the margin, *Support Vectors*!)

Dual Problem

- inserting the results from the derivatives into the Lagrangian leads to the *dual problem*: maximize

$$W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j (x_i \cdot x_j)$$

subject to $\alpha_i \geq 0, i = 1, \ldots, m$, and $\sum_{i=1}^{m} \alpha_i y_i = 0$

- the hyperplane decision function can now be written as

$$f(x) = \text{sgn} \left(\sum_{i=1}^{m} y_i \alpha_i \cdot (x \cdot x_i) + b \right)$$

where b is computed by $\alpha_i \cdot (y_i((x_i \cdot w) + b) - 1) = 0, \quad i = 1, \ldots, m$
(Karush-Tucker complementarity conditions)

Support Vectors

- Support Vectors are the points closest to the optimal hyperplane, i.e. lying on the margin
- only support vectors define optimal hyperplane, since $\alpha_i \neq 0$ only for support vectors, i.e. $w = \sum_{i=1}^{\#sv} \alpha_i y_i x_i^{sv}$
- all other training instances can be discarded after training

Kernel Functions and Kernel Trick

- Φ maps training data into a higher dimensional feature space, where an optimal separating hyperplane is constructed
- the dual problem becomes: maximize

$$W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j (\Phi(x_i) \cdot \Phi(x_j))$$

  subject to $\alpha_i \geq 0, i = 1, \ldots, m$, and $\sum_{i=1}^{m} \alpha_i y_i = 0$
- the hyperplane decision function becomes

$$f(x) = \text{sgn} \left(\sum_{i=1}^{m} y_i \alpha_i \cdot (\Phi(x) \cdot \Phi(x_i)) + b \right)$$

Kernel Functions and Kernel Trick Cont.

- Kernel Function specified by:
  $\mathcal{K} : X \times X \rightarrow \mathbb{R}, (x, x') \mapsto (\Phi(x) \cdot \Phi(x'))$

- since all feature vectors only occure in dot-products, the problem can be solved by using the kernel function, i.e. *without* explicitly carrying out Φ
  (substitute dot-productes by kernel function!)

- Example: $X = \mathbb{R}^2, \Phi : (x_1, x_2) \mapsto (x_1^2, x_1 x_2 \sqrt{2}, x_2^2)$
  $\rightsquigarrow (\Phi(x) \cdot \Phi(x')) = (x \cdot x')^2 =_{def} \mathcal{K}(x, x')$, i.e. $\mathcal{K}(x, x')$ computes $(\Phi(x) \cdot \Phi(x'))$ without computing $\Phi(x)$ and $\Phi(x')$

- Φ might actually be unknown and dimension of feature space might be infinite!