# 1 SVM, Adaboost

## 1.1 Lineární SVM

SVM je lineární klasifikátor minimalizující strukturální risk. SVM řeší kvadratickou optimalizační úlohu:

$$minimize \ \frac{1}{\|w\|^2} \tag{1}$$

$$s.t.: \ y_i(w \cdot x_i + b) \geq 1, i \in \{1, ..., N\} \tag{2}$$

Duální úloha:

$$maximize \ L(\mathbf{h}) = \sum_{i=1}^{N} h_i - 0.5 \cdot \mathbf{h} \cdot \mathbf{D} \cdot \mathbf{h} \tag{3}$$

$$s.t.: \ \mathbf{h} \cdot \mathbf{y} = 0 \tag{4}$$

$$h \geq 0 \tag{5}$$

Kde $\mathbf{h}$ je vektor lagrangeových multiplikátorů a $D = y_i y_j x_i \cdot x_j$ ($x_i$ a $x_j$ jsou vektory, $D$ matice). Vektor $\mathbf{h}$ má nenulové prvky jenom pro support vectory.

Když nejsou data lineárně separovatelné dodáme slackové proměnné $\sigma$, které nám dovolí relaxovat problém zanedbáním některých pozorování. Formulace problému se změní následovně:

$$minimize \ \frac{1}{\|w\|^2} + C \sum_{i=1}^{N} \sigma_i \tag{6}$$

$$s.t.: \ y_i(w \cdot x_i + b) \geq 1 - \sigma_i, i \in \{1, ..., N\} \tag{7}$$

$$\sigma_i \geq 0, i \in \{1, ..., N\} \tag{8}$$

a duální:

$$maximize \ L(\mathbf{h}) = \sum_{i=1}^{N} h_i - 0.5 \cdot \mathbf{h} \cdot \mathbf{D} \cdot \mathbf{h} \tag{9}$$

$$s.t.: \ \mathbf{h} \cdot \mathbf{y} = 0 \tag{10}$$

$$0 \leq \mathbf{h} \leq C \tag{11}$$

Shrnutí v Figure 1.

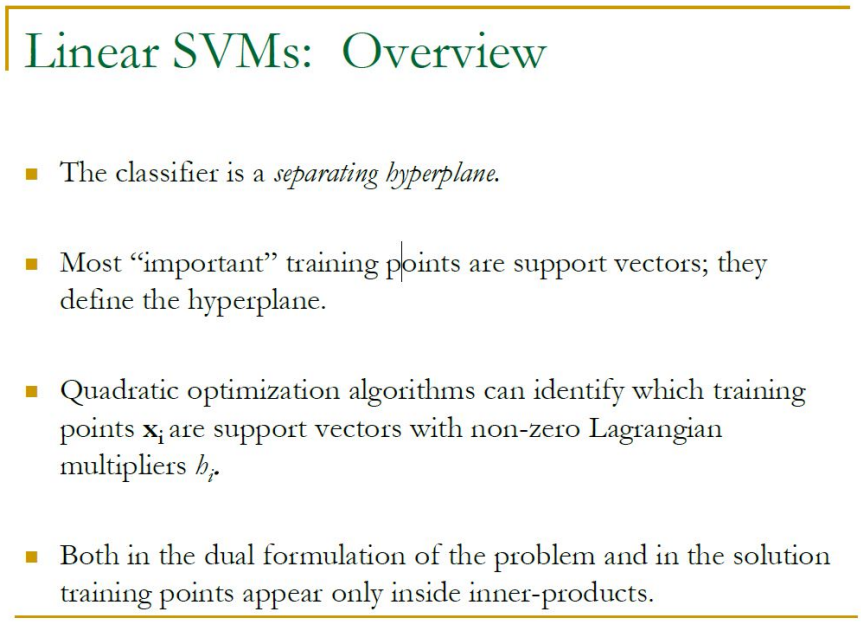



Figure 1: SVM shrnutí

### 1.2 Nelineární SVM

Základní myšlenka je že namapujeme originální, nadrovinou nerozdělitelná data do prostoru větší dimenze, kde už to půjde. Příklad ve Figure 2. Pro tuto akci využijeme vlastnost, že SVM závisí pouze na skalárním součinu $x_i \cdot x_j$. Tento součin nahradíme jádrovou funkcí která bude mapovat naše data do prostoru s vyšší dimenzí. Příklady jádrových funkcí ve Figure 3. Celkové shrnutí SVM v 4. Formulace problému tentokrát už jenom v duálu:

$$maximize\ L(\mathbf{h}) = \sum_{i=1}^{N} h_i - 0.5 \cdot \mathbf{h} \cdot \mathbf{D} \cdot \mathbf{h} \tag{12}$$

$$s.t.: \ \mathbf{h} \cdot \mathbf{y} = 0 \tag{13}$$

$$0 \leq \mathbf{h} \leq C \tag{14}$$

$$\mathbf{D} = y_i y_j K(\mathbf{x}_i, \mathbf{x}_j) \tag{15}$$

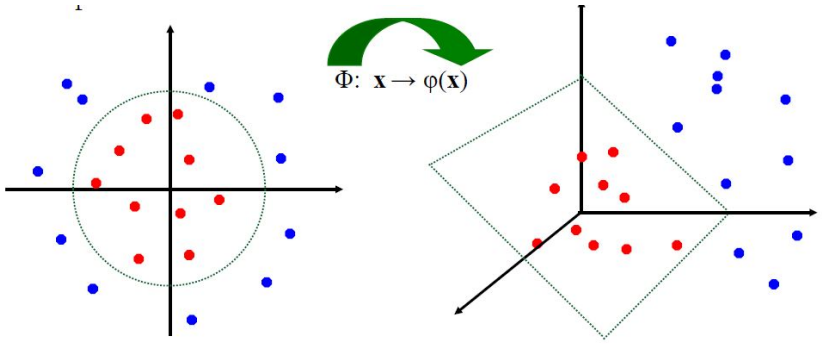



Figure 2: Nonlinear SVM example

## Examples of Kernel Functions

- Linear kernel: $K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i \cdot \mathbf{x}_j$
- Polynomial kernel of power *p*: $K(\mathbf{x}_i, \mathbf{x}_j) = (1 + \mathbf{x}_i \cdot \mathbf{x}_j)^p$
- Gaussian kernel: $K(\mathbf{x}_i, \mathbf{x}_j) = e^{-\|\mathbf{x}_i - \mathbf{x}_j\|^2 / 2\sigma^2}$
  - In the form, equivalent to RBFNN, but has the advantage of that the center of basis functions, i.e., support vectors, are optimized in a supervised.
- Two-layer perceptron: $K(\mathbf{x}_i, \mathbf{x}_j) = \tanh(\alpha \mathbf{x}_i \cdot \mathbf{x}_j + \beta)$

Figure 3: Jádrové funkce

## SVM Overviews

- Main features:
  - By using the kernel trick, data is mapped into a high-dimensional feature space, without introducing much computational effort;
  - Maximizing the margin achieves better generation performance;
  - Soft-margin accommodates noisy data;
  - Not too many parameters need to be tuned.

Figure 4: Shrnutí SVM

# 2 Adaboost

Úvod k algoritmu v 5, popis algoritmu v Figure 6 a výhody a nevýhody v Figure 7.

AdaBoost is an algorithm for constructing a "strong" classifier as linear combination

$$f(x) = \sum_{t=1}^{T} \alpha_t h_t(x)$$

of "simple" "weak" classifiers $h_t(x)$.

**Terminology**

- $h_t(x)$ ... "weak" or basis classifier, hypothesis, "feature"
- $H(x) = sign(f(x))$ ... "strong" or final classifier/hypothesis

**Interesting properties**

- AB is a linear classifier with all its desirable properties.
- AB output converges to the logarithm of likelihood ratio.
- AB has good generalisation properties.
- AB is a feature selector with a principled strategy (minimisation of upper bound on empirical error).
- AB close to sequential decision making (it produces a sequence of gradually more complex classifiers).

Figure 5: Úvod k adaboost

Given: $(x_1, y_1), \ldots, (x_m, y_m); x_i \in \mathcal{X}, y_i \in \{-1, +1\}$
Initialise weights $D_1(i) = 1/m$.
For $t = 1, ..., T$:

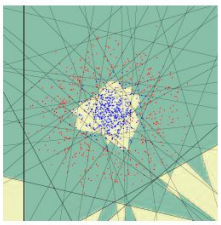
- Find $h_t = \arg\min_{h_j \in \mathcal{H}} \epsilon_j = \sum_{i=1}^{m} D_t(i)[\![y_i \neq h_j(x_i)]\!]$
- If $\epsilon_t \geq 1/2$ then stop
- Set $\alpha_t = \frac{1}{2}\log(\frac{1-\epsilon_t}{\epsilon_t})$
- Update

$$D_{t+1}(i) = \frac{D_t(i)exp(-\alpha_t y_i h_t(x_i))}{Z_t}$$

Output the final classifier:

$$H(x) = sign\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$

$t = 40$



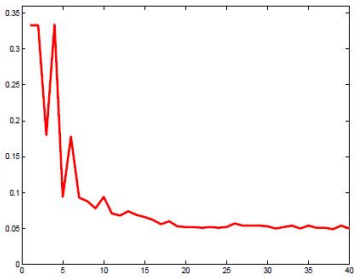



Figure 6: Shrnutí adaboostu

**Advantages**

- Very simple to implement
- General learning scheme - can be used for various learning tasks
- Feature selection on very large sets of features
- Fairly good generalisation

**Disadvantages**

- Suboptimal solution (greedy learning)
- Can overfit in presence of noise (has been addressed recently)

Figure 7: Výhody a nevýhody