Support Vector Machines

Kai-Uwe Kühnberger
University of Osnabrück
15.01.2021

UNIVERSITÄT OSNABRÜCK

Machine Learning Methodology: Schedule

- Overview Machine Learning Sessions

  - Basics (last week)
    - Machine Learning
    - Important Concepts
    - Clustering Methods
    - Properties of Hypotheses

  - Classification Methods (today)
    - Support Vector Machines
    - Example: Document Classification

  - Classification Methods (next week)
    - Decision Trees
    - Random Forests

  - Literature

Idea

- Support Vector Machines (SVMs)
- Motivation
  - Single-layer neural networks have efficient learning algorithms but can only learn linear functions
  - Multi-layer neural networks can learn non-linear functions, but are hard to train
  - Idea: SVMs map an input space into a space of higher dimensionality:

    Then separability is possible with linear functions

UNIVERSITÄT OSNABRÜCK

Support Vector Machines

If no simple linear hyperplane exists: map to a higher dimensional space

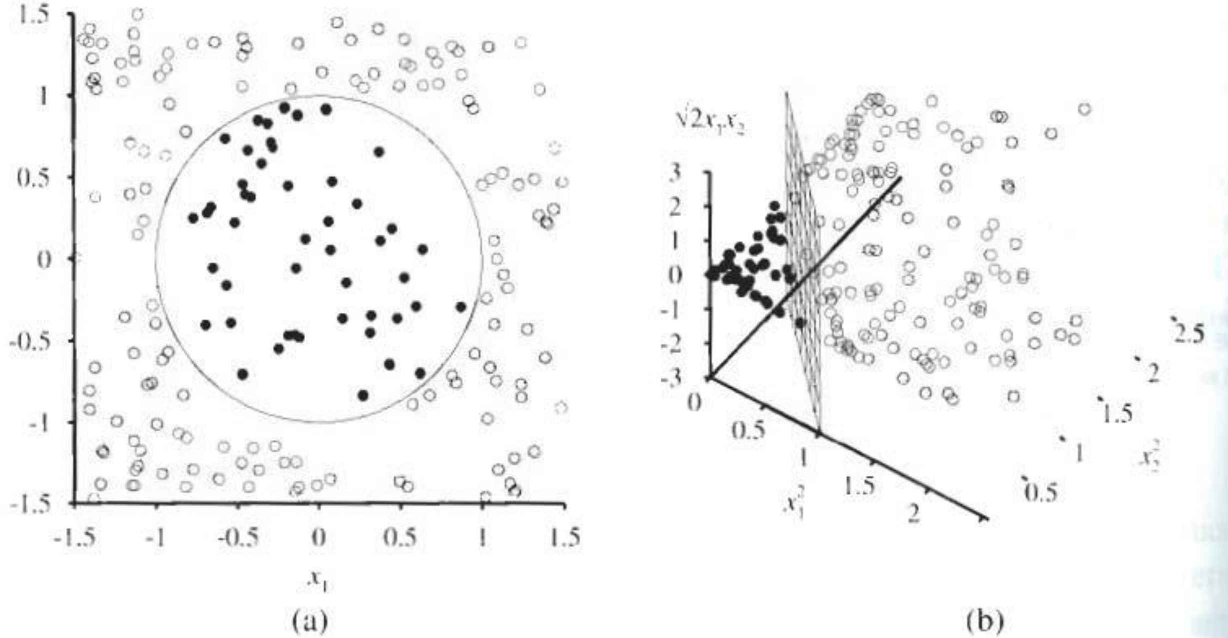


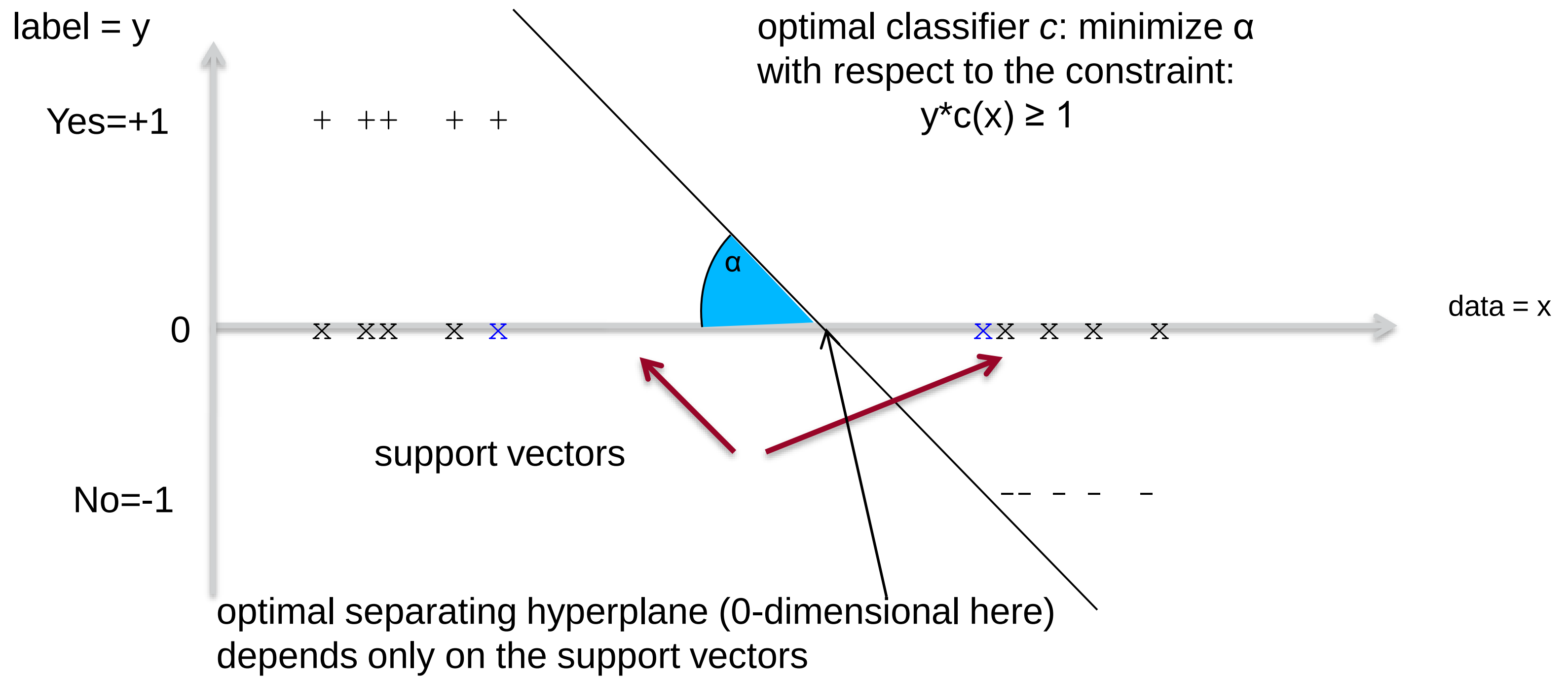
Figure 20.27 (a) A two-dimensional training with positive examples as black circles and negative examples as white circles. The true decision boundary, $x_1^2 + x_2^2 \leq 1$, is also shown. (b) The same data after mapping into a three-dimensional input space $(x_1^2, x_2^2, \sqrt{2}x_1x_2)$. The circular decision boundary in (a) becomes a linear decision boundary in three dimensions.

Russell & Norvig, p. 747

UNIVERSITÄT OSNABRÜCK

Support Vector Machines

- Basic Ideas: optimal classifiers in case of linear separable data
- Simplest case: one-dimensional data



Support Vector Machines

- Remarks on support vector machines:
  - Supervised learning model for classification
  - Mathematical foundations were proposed by Vapnik and Chervonenkis in the early 1960ies.
  - SVM were introduced by Vapnik and Lerner 1963 (special case of support vector machine algorithm).
  - Later SVMs were extended in various directions, e.g. to cover also non-linear classifiers.
- SVMs are a de facto standard in machine learning.
  - They were applied to a broad variety of domains and showed often excellent results.
  - Examples: face detection, text categorization, bioinformatics, character recognition, prediction of financial time series, prediction of health risks etc.
    - For some of these applications deep learning approaches are now outperforming SVMs, e.g. face detection, character recognition
    - For others the situation is not as clear.

Basics

- Assume data is given by vectors of dimensionality *n*.
  - We presuppose there is an *n*-dimensional vector space given.
- Find a hyperplane that separates positive from negative examples
  - A hyperplane in an *n*-dimension space is an *n*-1 dimensional subspace.
- If this is possible, we can use a linear classifier.
- Because there are potentially many such hyperplanes, we choose the one that maximizes the distance between the hyperplane and the nearest positive and negative examples.

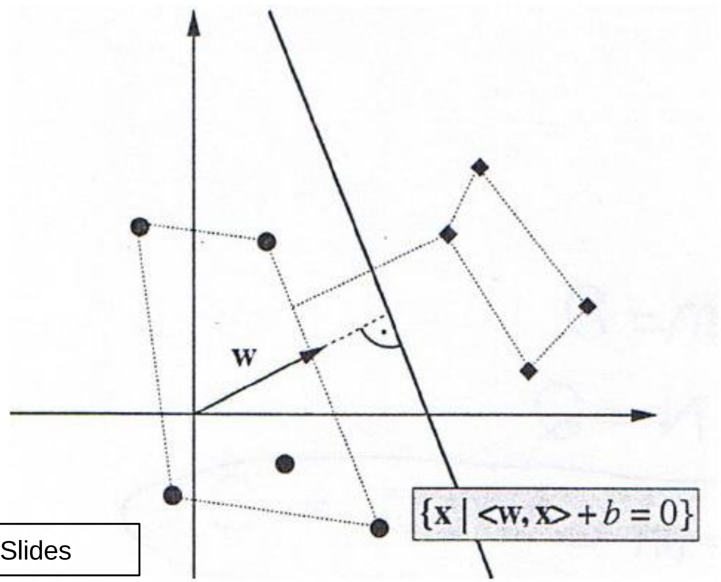
UNIVERSITÄT OSNABRÜCK

Basics

- Formally we have:
  - ($\mathbf{x}_1$,$\mathbf{y}_1$), ($\mathbf{x}_2$,$\mathbf{y}_2$),…,($\mathbf{x}_n$,$\mathbf{y}_n$) training data – class membership pairs
  - $y_i \in \{+1, -1\}$
  - Find a function f: $\Re^n \rightarrow \{+1,-1\}$, such that $f(\boldsymbol{x}_i) = y_\mathrm{i}$
  - Unseen data is then mapped via f.
- Assume again $\Re^n$ is given. A separating hyperplane H is defined as follows:

$$H = \{\boldsymbol{x} \in \Re^n \mid < \boldsymbol{w}, \boldsymbol{x} > + b = 0\}$$

- Here: $\boldsymbol{w} \in \Re^n$ orthogonal to H and $b \in \Re$.



Jasmin Fischer; Ulm, Online Slides

UNIVERSITÄT OSNABRÜCK

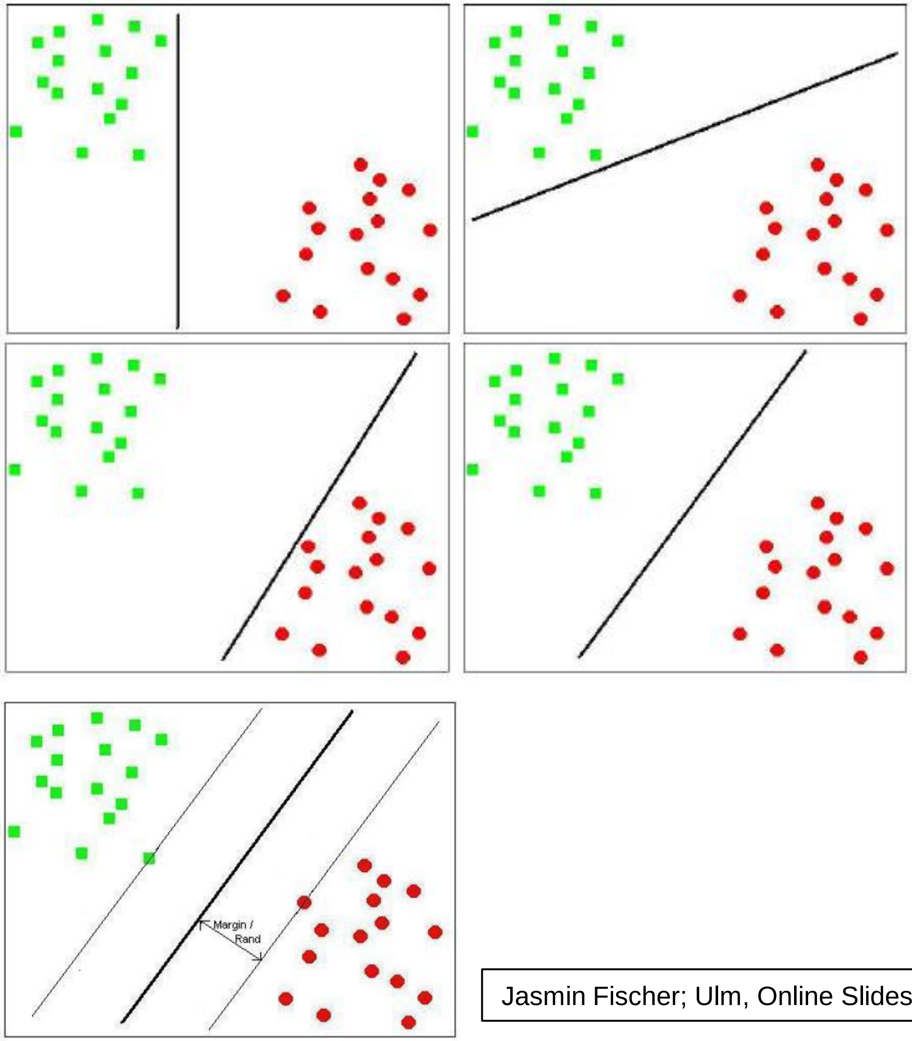
Generalized Portrait

- The generalized portrait version by Vapnik & Lerner, 1963:
- A(n) (affine) hyperplane in H is defined by a weight vector $\mathbf{w} \in H$ and a bias $b \in \Re$ with the condition that $\langle \mathbf{w},\mathbf{x} \rangle + b = 0$.
  - All $\mathbf{x} \in X$ that satisfy the condition above are on the hyperplane.
  - If $\langle \mathbf{w},\mathbf{x} \rangle + b < 0$, $\mathbf{x}$ is a negative example, if $\langle \mathbf{w},\mathbf{x} \rangle + b > 0$, $\mathbf{x}$ is a positive example.
- The original generalized portrait algorithm computes the optimal hyperplane in H that separates positive and negative examples (if they are linearly separable).
  - Task: Maximize the distance between the hyperplane and the closest points (margin).

UNIVERSITÄT OSNABRÜCK

Generalized Portrait

- Assume training data is linearly separable
- What is an optimal hyperplane separating the positive and negative examples?
- Optimality means here maximizing the distance between the hyperplane and the closest points (margin).



Jasmin Fischer; Ulm, Online Slides

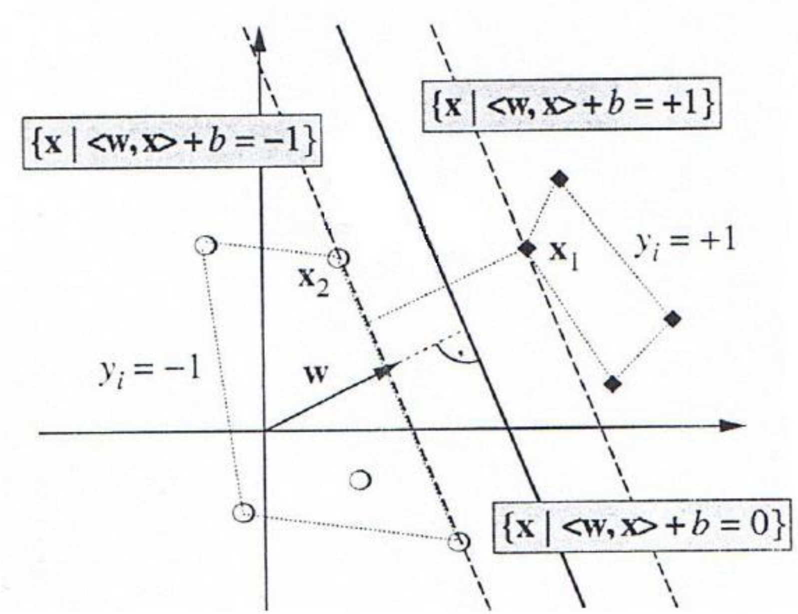
Generalized Portrait

- If the distance of two parallel hyperplanes that separate the classes is maximal, then the region between these hyperplanes is the margin.
- These hyperplanes can be specified as follows:

$$< \boldsymbol{w}, \boldsymbol{x_1} > + b = +1$$
$$< \boldsymbol{w}, \boldsymbol{x_2} > + b = -1$$

$$\Rightarrow < \boldsymbol{w}, (\boldsymbol{x_1} - \boldsymbol{x_2}) > = 2$$

$$\Rightarrow \; < \frac{\boldsymbol{w}}{||\boldsymbol{w}||}, (\boldsymbol{x_1} - \boldsymbol{x_2}) > = \frac{2}{||\boldsymbol{w}||}$$



- Margin is specified: 2/||***w***||

UNIVERSITÄT OSNABRÜCK

Generalized Portrait

- Maximizing the margin is equivalent to minimizing the following expression.

  $$\tau(\mathbf{w}) = ½ \cdot ||\mathbf{w}||^2 \text{ with condition } y_i \cdot (<\boldsymbol{x_i},\boldsymbol{w}> + b) \geq 1$$

  - If the objective function is found and the constraint holds that all training examples are classified correctly, finding the optimal hyperplane can be phrased as a constraint **optimization problem**.
- Such optimization problems can be solved by so-called Lagrange multipliers.
  - Formally: Consider the simplified optimization problem:

    maximize $f(x,y)$
    subject to the condition $g(x,y) = 0$.
  - We assume that both f and g have continuous first partial derivatives. We introduce a new variable λ called a **Lagrange multiplier** and study the **Lagrange function** (or **Lagrangian**) defined by

    $$L(x,y,\lambda) = f(x,y) - \lambda \cdot g(x,y)$$

Generalized Portrait

- Transferring Lagrange multipliers and the Lagrange function to the case of support vector machines, we get the following:

$$L(w,b,\alpha) = ½ \cdot ||\mathbf{w}||^2 - \sum_N \alpha_i \cdot (y_i \cdot (<\boldsymbol{x_i},\boldsymbol{w}> + b) - 1)$$

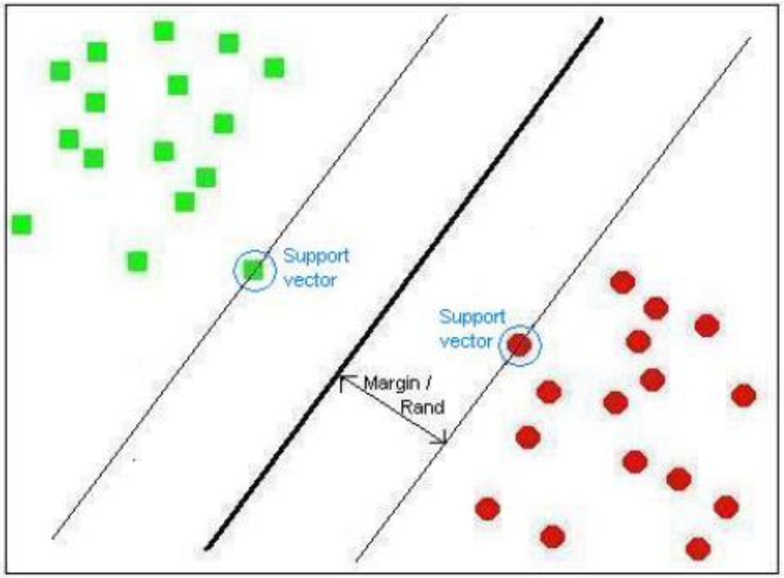
  with Lagrange multipliers $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_N)$

- Maximizing Lagrange multipliers α_i while minimizing $\boldsymbol{w}$ and b, yields the following equation:
  - $\mathbf{w} = \sum_{i \in N} \alpha_i \cdot y_i \cdot \mathbf{x_i}$
  - Intuitively: the weight vector can be expressed as a linear combination (sum) of the training examples.

UNIVERSITÄT OSNABRÜCK

Generalized Portrait

- In fact, there is a rather strong consequence:
  - According to the Karush-Kuhn-Tucker saddle point condition: $\alpha_i \cdot (y_i \cdot (<\mathbf{x_i},\boldsymbol{w}> + b) - 1) = 0$
  - All points where $\alpha_i > 0$ specify the optimal hyperplane *H* (points closest to the hyperplane, i.e. support vectors).
  - The rest of the points ($\alpha_i = 0$) do not have an influence.
- In fact, the support vectors (points that are closest to the hyperplane) suffice to compute the weight vector as the linear combination above: $\mathbf{w} = \sum_{i \in N:\boldsymbol{xi} \text{ Support Vector}} \alpha_i \cdot y_i \cdot \mathbf{x_i}$



UNIVERSITÄT OSNABRÜCK

Support Vector Machines

- The new decision function can now be expressed as follows:

$$f(\mathbf{x}) = sgn(\textstyle\sum_i \alpha_i \cdot y_i \cdot \langle \mathbf{x},\mathbf{x_i} \rangle + b)$$

- Support vector machines allow to replace the dot product by a kernel (kernel trick):

$$f(\mathbf{x}) = sgn(\textstyle\sum_i \alpha_i \cdot y_i \cdot k(\mathbf{x},\mathbf{x_i}) + b)$$

- The kernel function $k: X \times X \rightarrow \Re: (x,x') \rightarrow k(x,x')$ can be used to map a low-dimensional input space X to a feature space H with higher dimensionality:
  - An appropriate choice of the feature space H allows to linearly separate examples that are not linearly separable in the input space X.
  - A kernel allows to compute the decision boundary without computing the whole function but only the support vectors.

Kernel

- Again: The idea is to use a similarity measure (kernel)

  $$k: X \times X \rightarrow \Re: (x,x') \rightarrow k(x,x')$$

- Formal definition: A mapping $k: X \times X \rightarrow \Re$ is called *kernel*, if there is a product space $(H,\langle\cdot,\cdot\rangle)$ and a mapping $\Phi: X \rightarrow H$, such that $k(x,x') = \langle\Phi(x),\Phi(x')\rangle$.
- Remarks:
  - X can be any set on which a kernel can be defined.
  - Training data is given by pairs (x_i,y_i) for $x_i \in X$ and $y_i \in \{0,1\}$
  - An example for a kernel is the dot product $\langle\boldsymbol{a},\boldsymbol{b}\rangle$ in a product space: $\langle\boldsymbol{a},\boldsymbol{b}\rangle = a_1b_1 + a_2b_2 + \ldots + a_nb_n$
    - Take, for example, a vector space for the dot product space, and vectors $\boldsymbol{a}$ and $\boldsymbol{b}$ as factors.

UNIVERSITÄT OSNABRÜCK

Kernels

- There are many possibilities to define kernels.
- The choice of the kernel is dependent on the learning task and the input domain.
- Here are three important kernels:

Kernel	Definition	Parameter
Linear Kernel	$k(x, x') = < x, x' >$	none
Polynomial Kernel	$k(x, x') = < x, x' >^d$	Polynomial degree *d*
Gaussian Kernel / RBF Kernel	$k(x, x') = exp(-\frac{\lvert x-x' \rvert^2}{2\sigma^2})$	Free parameter σ

Klaus-Michael Lux, Bachelor Thesis

- For linear separable problems, non-linear kernels do not increase performance.
- For various applications new kernels were invented.

Support Vector Machines

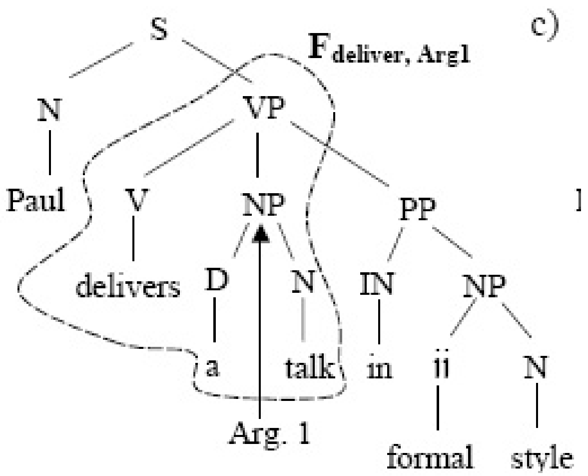
- Support vector machines are defined on binary classification problems.
  - How can multi-class classification problems be addressed?
- Here are two methods that have been proposed:
  - One-against-all
    - For k classes, k binary SVM models are trained. The ith SVM is trained using the examples in the ith class as positive examples and all other examples as negative examples.
    - For classifying x, feed x it into all k models. Select the model for which the decision function is maximal.
  - One-against-one
    - For k classes, $k \cdot (k\text{-}1)/2$ SVM models are trained. Each is trained to decide between two particular classes, leaving out all examples from other classes.
    - Each SVM votes for one class. The class with the most votes is selected.

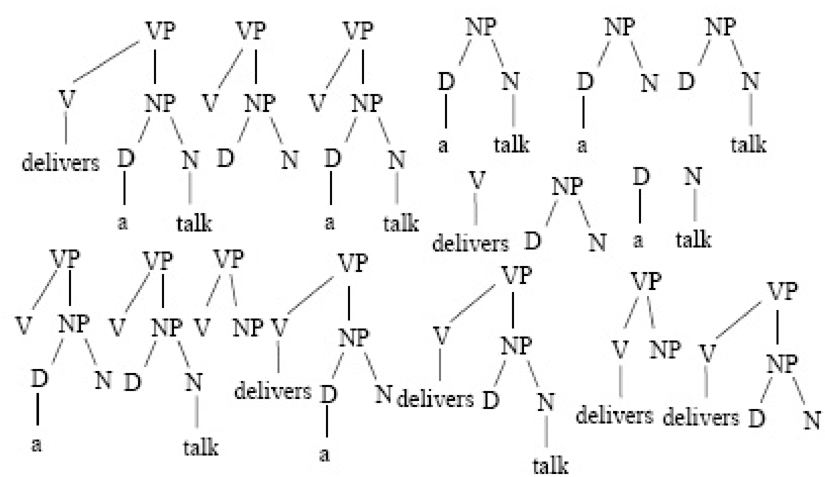
Example

Document Classification

Example (based on Geibel et al., 2007)

- We want to classify DOM (document object model) trees of documents
  - DOM trees are rooted, labeled, ordered trees.
  - Inner nodes are labeled with XML tags, leaves might be labeled with sentences (maybe also represented as trees).
  - Approach works for every tree-like presentation.
- The Parse Tree Kernel (Collins & Duffy)
  - A tree is described by all possible intermediate parse trees t_i resulting in a feature vector $(\varphi_{t1}, . . . , \varphi_{tk})$.





UNIVERSITÄT OSNABRÜCK

Example

- The Parse Tree Kernel (Collins & Duffy):
  - Assume $\Delta(v,v')$ is defined as the number of isomorphic mappings of partial parse trees rooted in nodes *v* and *v'*.
  - A kernel can be computed recursively by $k(T,T') = \sum_{v \in V, v' \in V} \Delta(v,v')$ such that
    - $\Delta(v,v') = 0$ if the productions applied in *v* and *v'* are different.
    - $\Delta(v, v') = 1$, if the productions in v and v' are identical and both nodes are pre-terminals.
    - For other non-terminals with identical productions (v_i is the *i*th child of *v* and $n(v)$ denotes the number of children of *v*):

$$\Delta(v, v') = \prod_{i=1}^{n(v)} (1 + \Delta(v_i, v_i'))$$

- This idea can now be applied to DOM trees.

UNIVERSITÄT OSNABRÜCK

Example

- The Simple Tree Kernel SimTK for DOM trees:
  - Incorporate kernel k^{Σ} operating on pairs of node labels (tags, attributes, text).
  - If there are either no children, or the number of children differs we set:

$$\Delta_{\text{SimTK}}(v,v') = \lambda \cdot k^{\Sigma}(\alpha(v), \alpha(v'))$$

  - Else: $$\Delta_{\text{SimTK}}(v,v') = \lambda \cdot k^{\Sigma}(\alpha(v), \alpha(v'))(1 + \prod_{i=1}^{n(v)} \Delta_{\text{SimTK}}(v_i, v'_i))$$

α is a mapping from nodes to node labels Σ, λ is a parameter.

- Pros of the SimTK:
  - No grammar is presupposed.
  - We can include complex node labels, e.g. text in leave nodes.
- • Shortcomings of the SimTK:
  - If the number of children differs, then the children are not compared.
  - If the number of children is not different, then they are only compared in the original order.

UNIVERSITÄT OSNABRÜCK

Example

- The Left Aligned Tree Kernel (compares just as many children as possible, if the number of children differ)
- Recursive case:

$$\Delta(v, v') = \lambda \cdot k^{\Sigma}(\alpha(v), \alpha(v'))\left(1 + \sum_{k=1}^{\min(n(v), n'(v'))} \prod_{i=1}^{k} \Delta(v_i, v_i')\right)$$

- Shortcomings: Trees occurring more on the left have a higher influence than trees occurring on the right and no permutations are allowed.
- The Set Tree Kernel (treat children as a set)

$$\Delta(v, v') = \lambda \cdot k^{\Sigma}(\alpha(v), \alpha(v'))\left(1 + \sum_{i=1}^{n(v)} \sum_{i'=1}^{n'(v')} \Delta(v_i, v_{i'}')\right)$$

- Shortcomings: no information about ordering retained at all.

UNIVERSITÄT OSNABRÜCK

Example

- The Soft Tree Kernel
- Idea: use a fuzzy / soft comparison of node positions using an RBF kernel:

$$k_\gamma(x, y) = e^{-\gamma(x-y)^2}$$

- Recursion:

$$\Delta(v, v') = \lambda \cdot k^{\Sigma}(\alpha(v), \alpha(v')) \cdot k_\gamma(\mu(v), \mu'(v')) \cdot \Big(1 + \sum_{i=1}^{n(v)} \sum_{i'=1}^{n'(v')} \Delta(v_i, v'_{i'})\Big)$$

- $\mu(v_i) = i$ specifies the position of child v_i of some node v, γ is a parameter, x and y are positions of children of a node.
- Pros: Has everything that is necessary.

UNIVERSITÄT OSNABRÜCK

Example

- Apply these kernels to artificial data and real data.
  - The class 1 examples all have a left-aligned subtree of the form g(a, b(e), c).
  - The class 2 examples all have a general ordered subtree of the form g(c, b, e(a)), where gaps are allowed but the ordering of the subtrees c, b and e(a) has to be preserved.
  - The class 3 examples contain subtrees of the form g(c, b, a(e)), where the child trees c, b and a(e) are allowed to occur reordered and gaps might have been inserted, too.

Table: Optimal F-Measures

	Class 1	Class 2	Class 3
TagTK	0.727	0.6	0.736
LeftTK	0.909	0.363	0.44
SetTK	0.952	1.00	1.00
SoftTK	1.0	1.0	1.0
StringTK	1.0	1.0	1.0

UNIVERSITÄT OSNABRÜCK

Example

Class 1:
f(n,h(m),g(a,b(e),c))
f(h(m,g(a,b(e),c)))
f(g(a,b(e),c,n),h(m))
f(g(a,b(e),c,n,m))
f(a,m,m,h(h(g(a,b(e),c))),n)
f(g(a,b(e),c),g(e(a),c,a))
f(h(m,c),g(a,b(e),c),g(m,n,m))
f(g(a,b(e),c,h(h(m),n,b)))
f(h(h(g(a,b(e),c))))
f(g(a,b(e),c),a)

Class 2:
f(n,h(n),g(c,m,b,n,e(a)))
f(h(h(m),n,b),g(c,b,n,m,e(a)))
f(h(g(h(n),c,b,e(a))),b)
f(h(g(n,c,b,e(a))),h(m,b,h(a,n)))
f(g(b(e),c,a),g(c,n,b,e(a)))
f(g(c,h(c,n),b,h(h(h(b))),e(a)))
f(g(c,b,e(a)),g(m,b,n,a,b(e),n,c))
f(b,g(c,b,e(h(b,m,a),a)))
f(g(g(a),c,d,m),g(c,b,h(n),e(a)))
f(g(m,c,b,e(a),h(a)))

Class 3:
f(n,g(c,n,b,m,a(e)),h(m))
f(m,e,b,h(g(b,n,c,m,a(e))))
f(h(h(a(e),m,b,n,n,c)),b,e)
f(e,b,g(m,c,a,b,e,e,a(e)))
f(b,e,g(a(e),m,b,h(a),m,c))

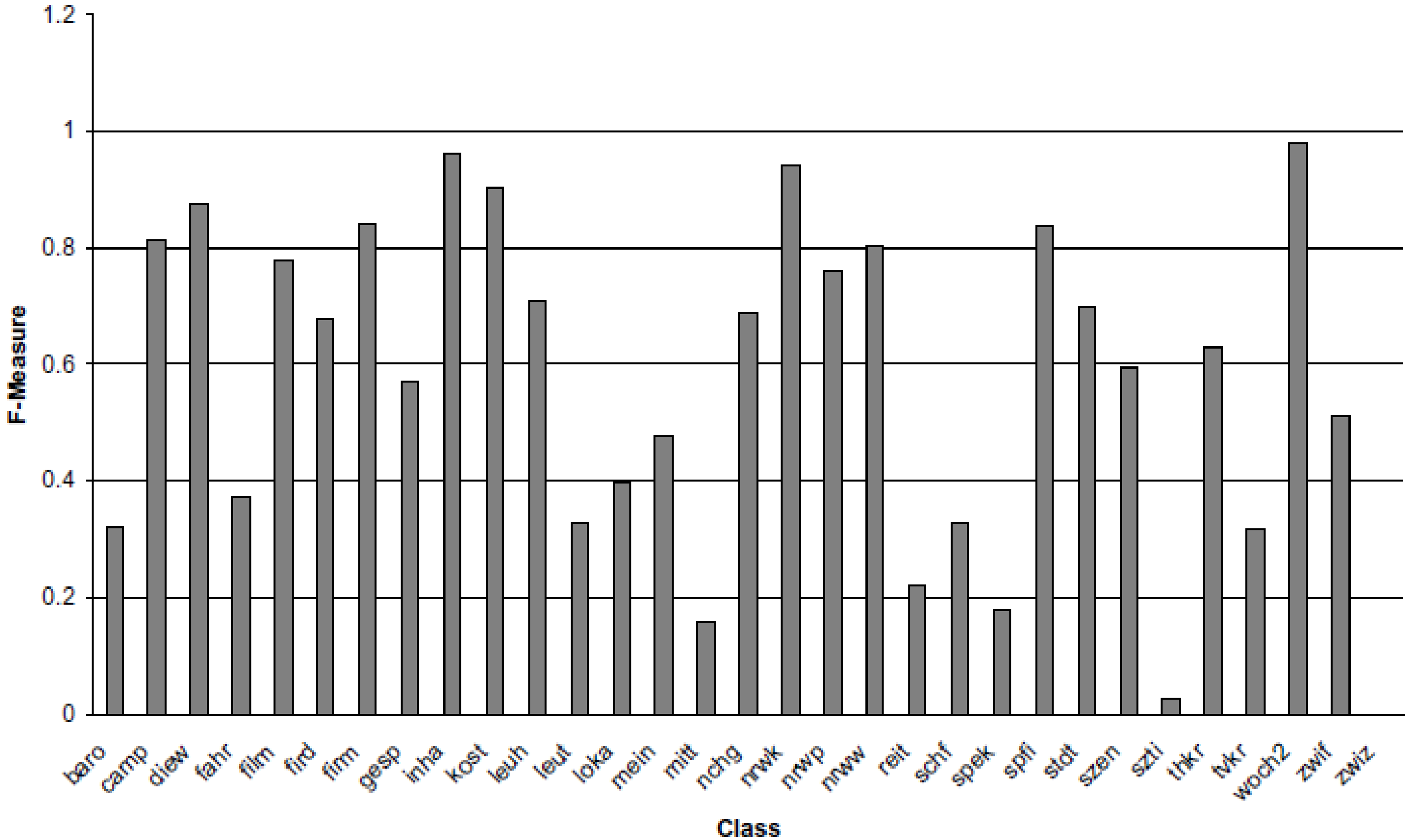
Class 3 (continued):
f(g(h(m,e),a(e),h(b),c,a,b))
f(g(b,h(m),c,h(n),a(b,e,a)))
f(g(m,h(n,h(n)),g(b,e,c,a(e))))
f(g(a(e),g(h(n)),b,c),b,n)
f(h(h(e),h(n),b,h(n),m),g(c,m,n,b,a(e)))

Example

- Real world example:
- Approx. 35,000 short texts from Süddeutsche Zeitung were considered.
  - 31 Classes:
    - Bühnentip (theater)
    - Hochschulnachrichten (university news)
    - Fragen und Antworten (questions and answers)
    - Inhalt (content)
    - Wochenchronik (chronicle of the week)
    - Etc.
- Document DOM tree + class

UNIVERSITÄT OSNABRÜCK

Example



- Result for LeftTK (left aligned TK) are depicted.
- Many classes are learnable, some still have problems.
- SoftTK and SetTK: rather bad results.

UNIVERSITÄT OSNABRÜCK

Bibliography

- Support Vector Machines
  - Schölkopf, Smola: *Learning with Kernels*, MIT Press, 2001.
  - Vapnik and Lerner: Pattern Recognition Using Generalized Portrait Method. *Automation and Remote Control*, Vol. 24, 1963, pp. 774-780.
  - Vapnik, Vladimir N (2000). *The Nature of Statistical Learning Theory*. Springer.
  - Geibel, Gust, Kühnberger (2007): Variants of Tree Kernels for XML Documents. In: A. Gelbukh and A.F. Kuri Morales (Eds.): *MICAI 2007*, LNAI 4827, pp. 850–860, 2007.