

Stützvektor Methoden

Support Vector Methoden

Kernel Methoden

Grundlagen und Anwendungen

Prof Dr.-Ing. J. Marius Zöllner



Forschungszentrum Karlsruhe
in der Helmholtz-Gemeinschaft



Universität Karlsruhe (TH)
Forschungsuniversität • gegründet 1825

Grundsatz



William of Occam

(1280/88-1347/49)

Franziskaner

- Anklage wegen Ketzerei
- Entzug der Lehrerlaubnis

Razor principle (Rasiermesser - Prinzip) :

Entia non sunt multiplicanda sine necessitate
(Entities should not be multiplied beyond necessity)



Löse nie ein Problem komplizierter als nötig,
denn die einfachste, richtige Erklärung ist die Beste

Überwachtes Lernen aus Beispielen (Wdh.)

Lernen = Schätzen einer Abbildung (Hypothese) :

$$h_{\alpha}(\vec{x}) = y, \quad \alpha - Parameter$$

mit Hilfe von (bekannten) Beispielen (Lernbeispiele)

$$(\vec{x}_1, y_1), \dots, (\vec{x}_n, y_n) \in X \times Y$$

generiert von einem (nicht bekannten) System
mit der Wahrscheinlichkeit

$$P(\vec{x}, y)$$

Lernproblem = definiert durch $X \times Y$

$\{Attr_1, \dots, Attr_n\} \times \{true, false\}$ - Konzept

$R^N \times \{Klasse_1, \dots, Klasse_n\}$ - Klassifikation (numerisch)

$R^N \times R$ - Regression (numerisch)

Abschätzung des Testfehlers (VC - Dim.)

Nach Vapnik gilt mit Wahrscheinlichkeit $1 - \eta$

$$E(h_\alpha) \leq E_{emp}(h_\alpha) + \sqrt{\dots \frac{VC(h_\alpha)}{N} \dots}$$

wobei:

$VC(h_\alpha)$ – VC-Dimension der lernenden Maschine

N – Anzahl der Lernbeispiele

$E_{emp}(h_\alpha)$ – abhängig von $VC(h_\alpha)$ und N

Lernerfolg ist abhängig von:

- Kapazität der lernenden Maschine (so gering wie nötig)
- Optimierungsmethode (so gut wie möglich)
- Lernbeispiele (repräsentativ, so viele wie möglich)

Lösungsansatz: Structural Risk Minimization

Ziel: finde eine Lösung für

$$\min_{H_n} (E_{emp}(h_\alpha) + \sqrt{.... \frac{VC(h_\alpha)}{N}})$$

↔ finde $VC(h_\alpha)$ („Maschine“), N („Beispiele“),
und α („Minimum des emp. Fehlers“)

Ideale Lösung:

- Minimiere Summe (nicht jeweilige Summanden)
- Strukturiere den Hypothesenraum

$$H^1 \subset H^2 \subset \dots \subset H^n, \quad VC(h_\alpha^i) \leq VC(h_\alpha^{i+1})$$

- Suche Optimum: das Minimum für $E(h_\alpha)$

Support Vektor Methode

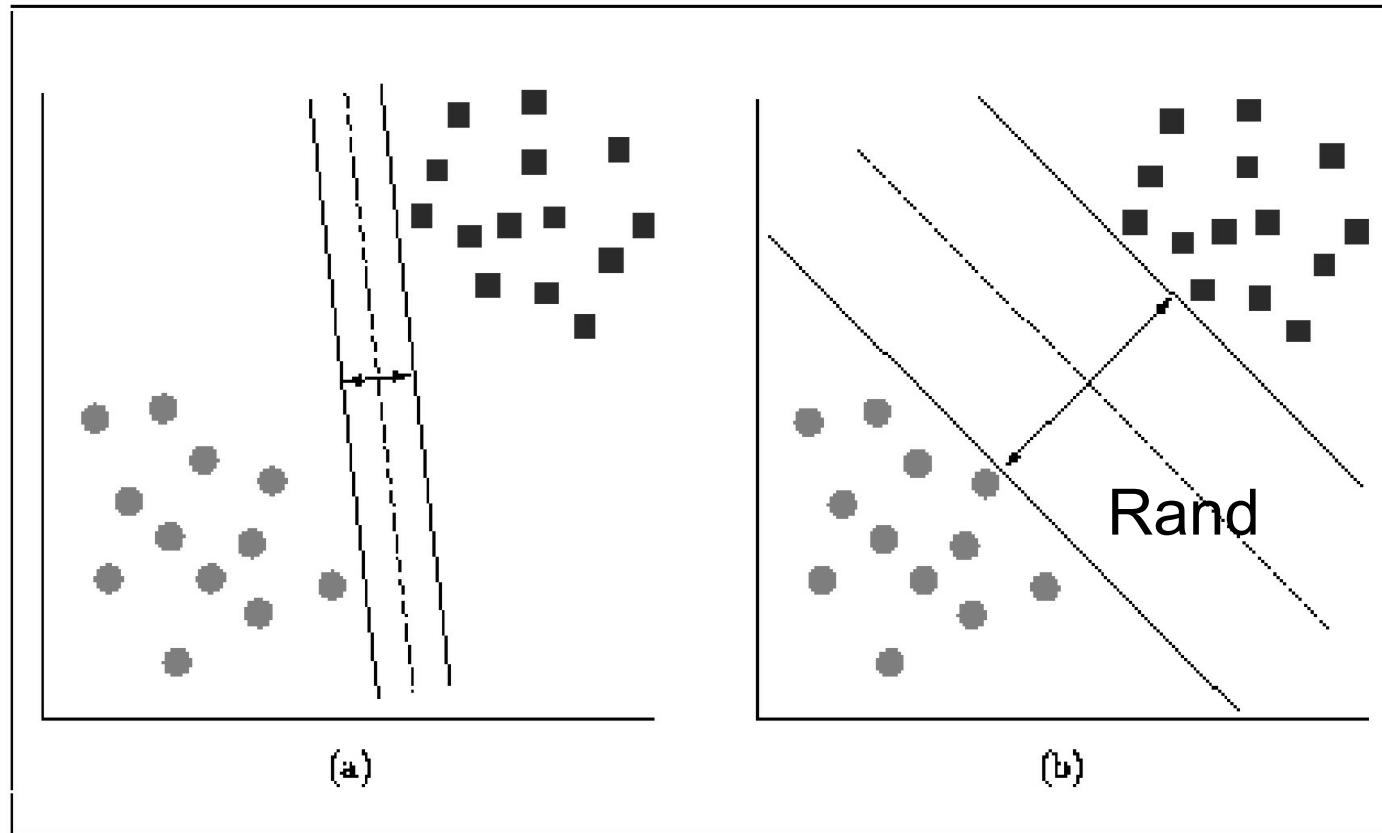
Lineare Support Vektor Methode
geometrische Interpretation / Formalisierung
Lernen = Optimierungsproblem lösen

Nichtlineare Support Vektor Methode
Soft-margin SVM
Kernel – Trick

Architektur
Support Vektoren

Optimierungen
Erweiterungen

Lineare Support Vektor Methode



Problem:

Klassifikation, Trenne die beiden Mengen

Lösung:

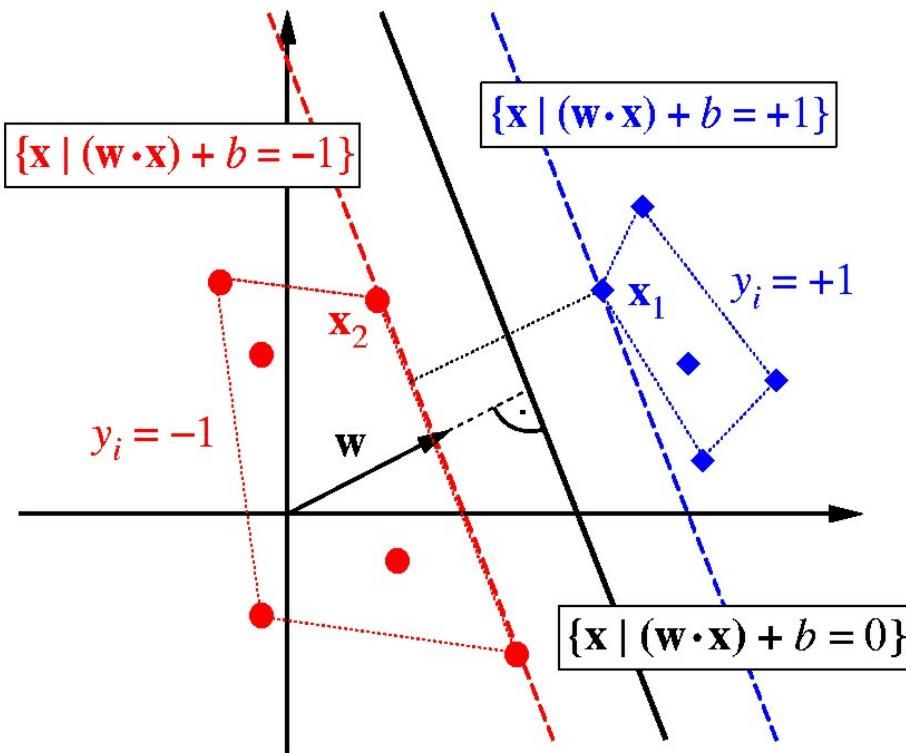
Finde die beste Trenn-Gerade

Intuition:

Größe des Randes (margin) \leftrightarrow Generalisierung

SV – Methode, Trennebene

Finde die Hyperebene $\{\vec{x} \in S : \vec{w}\vec{x} + b = 0, (\vec{w}, b) \in S \times R\}$



Abstand $\frac{\mathbf{w} \cdot (\mathbf{x}_1 - \mathbf{x}_2)}{\|\mathbf{w}\|}$

$$(\mathbf{w} \cdot \mathbf{x}_1) + b = +1$$
$$(\mathbf{w} \cdot \mathbf{x}_2) + b = -1$$
$$\Rightarrow (\mathbf{w} \cdot (\mathbf{x}_1 - \mathbf{x}_2)) = 2$$
$$\Rightarrow \left(\frac{\mathbf{w}}{\|\mathbf{w}\|} \cdot (\mathbf{x}_1 - \mathbf{x}_2) \right) = \frac{2}{\|\mathbf{w}\|}$$

$\vec{x}, \mathbf{x}_1, \mathbf{x}_2, \mathbf{w}$ - Vektoren

Bedingung für die optimale Hyperebene

$$\min_{i=1 \dots n} |\vec{w}\vec{x}_i + b| = 1, \quad \vec{x}_i - \text{Lerndaten}$$

→ Der nächste Punkt hat den Abstand $\frac{1}{\|\vec{w}\|}$

→ Der Abstand zw. den 2 Klassen $\frac{2}{\|\vec{w}\|}$

→ Die Entscheidungsfunktion eines Hyperebenen - Klassifikators

$$f_{\vec{w}, b}(\vec{z}) = \text{sign}(\vec{w}\vec{z} + b)$$

Hyperebene mit maximalem Abstand (margin)

$$\text{Maximiere } \frac{2}{\|\vec{w}\|} \quad \rightarrow \quad \text{Minimiere } \|\vec{w}\|^2$$

Unter den Bedingungen

$$y_i(\vec{w}\vec{x}_i + b) \geq 1 \quad i = 1 \dots n$$

d.h. die Daten werden korrekt klassifiziert

→ Laut Vapnik die Lernmaschine mit der kleinsten möglichen VC-Dimension (falls die Klassen linear trennbar sind)

Äquivalentes Problem (*Primäres Optimierungsproblem*):

Finde den (eindeutigen) Sattelpunkt der Funktion

$$L_P = L(\vec{w}, b, \vec{\alpha}) = \frac{1}{2} \|\vec{w}\|^2 - \sum_{i=1}^N \alpha_i (y_i (\vec{w} \cdot \vec{x}_i + b) - 1)$$

und $\alpha_1, \alpha_2, \dots, \alpha_N \geq 0$ (Lagrange-Multiplikatoren)

Sattelpunkt - Bedingungen:

Minimum von L bzgl. \vec{w}, b

$$\frac{\partial L}{\partial b} = \sum_{i=1}^N y_i \alpha_i = 0 \quad \frac{\partial L}{\partial \vec{w}} = \vec{w} - \sum_{i=1}^N \alpha_i y_i \vec{x}_i = 0$$

Maximum von L bzgl. $\alpha_1, \dots, \alpha_n$

→ Berechnung von \vec{w} und b direkt aus α_i, x_i, y_i

Minimierung: Lagrange - Methode

Duale Lagrange - Gleichung:

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

Duales Optimierungsproblem: $\max_{\alpha} \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i,j=1}^m \alpha_i \alpha_j y_i y_j \vec{x}_i \cdot \vec{x}_j$
unter den Bedingungen

$$\sum_{i=1}^m \alpha_i y_i = 0$$
$$\alpha_i \geq 0 \quad i = 1, \dots, m$$

Vorteile:

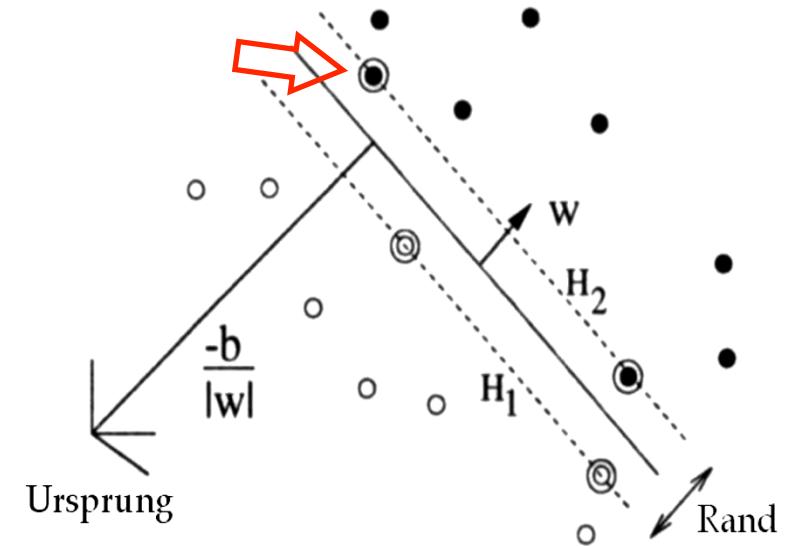
- Nur noch von α abhängig
- Einfacher zu lösen

Support Vektoren

Die meisten Bed. sind erfüllt

→ die meisten $\alpha_i = 0$
 (Sattelpunkt-Bedingung)

Support-Vektoren: \vec{x}_i mit $\alpha_i > 0$



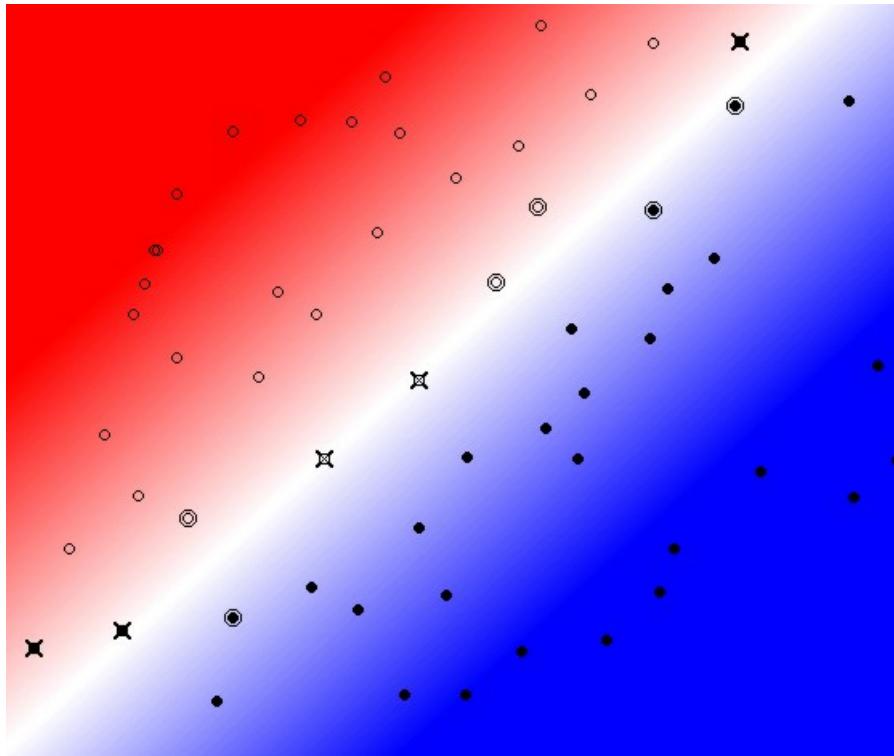
\vec{w} ist eine Linearkombination weniger Vektoren \vec{x}_i
 (Support Vektoren)

$$\vec{w} = \sum_{i=1}^N \alpha_i y_i \vec{x}_i$$

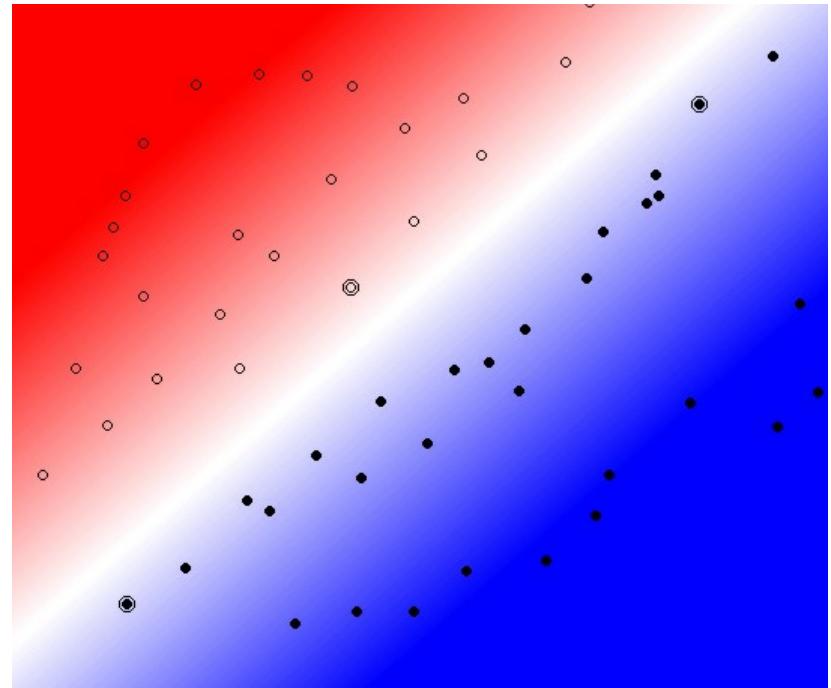
Support Vektoren liegen am nächsten zur Trennebene

Beispiel

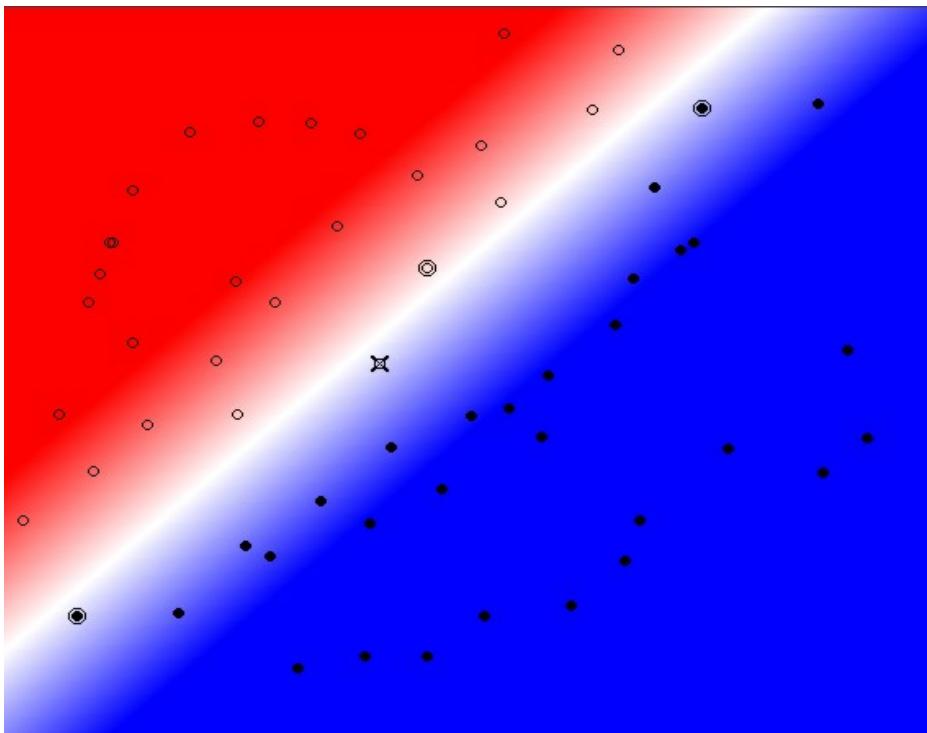
1 Lineare trennbare Daten



2 Nicht linear trennbare Daten



Soft Margin - Hyperebene



Idee:
Erlaube eine geringe Zahl
von Missklassifikationen
→ Höhere Generalisierung

Änderung der Randbedingungen

$$y_i(\vec{w}\vec{x}_i + b) \geq 1 - \xi_i \quad i = 1 \dots n, \xi_i \geq 0$$

Generalisierte optimale Hyperebene

Minimiere

$$\min_{\vec{w}, b, \xi_i} \frac{1}{2} \|\vec{w}\|^2 + C \left(\sum_{i=1}^m \xi_i \right)^p$$

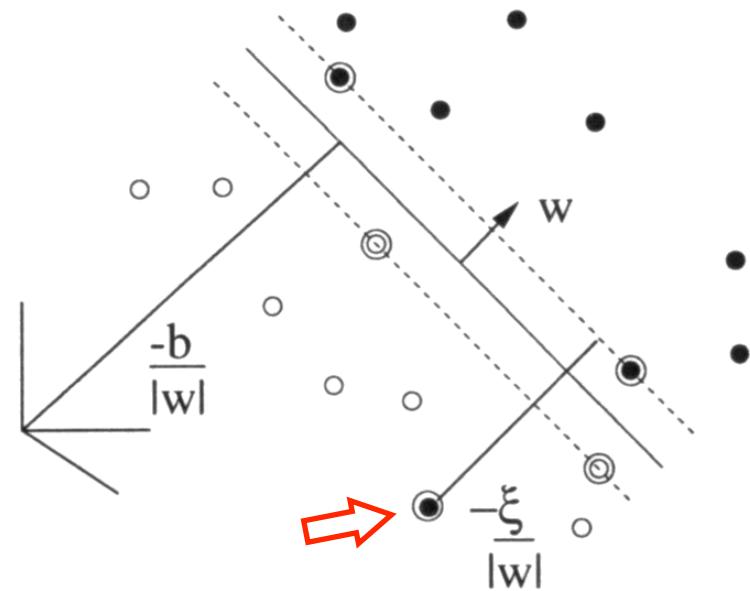
Bedingungen

$$y_i (\vec{w} \vec{x}_i + b) \geq 1 - \xi_i$$
$$\xi_i \geq 0$$

Lösung: Lagrange - Methode

Rolle von C

- Regulierungsparameter
- C – groß → wenig Missklassifikationen
- C – klein → maximale Margins



Support Vektoren

Support-Vektoren: alle Vektoren \vec{x}_i mit $\alpha_i > 0$

$$\alpha_i < C$$

Support Vektor liegt am Rand, Abstand
(margin – vector)

$$\frac{1}{\|\vec{w}\|}$$

$$\alpha_i = C$$

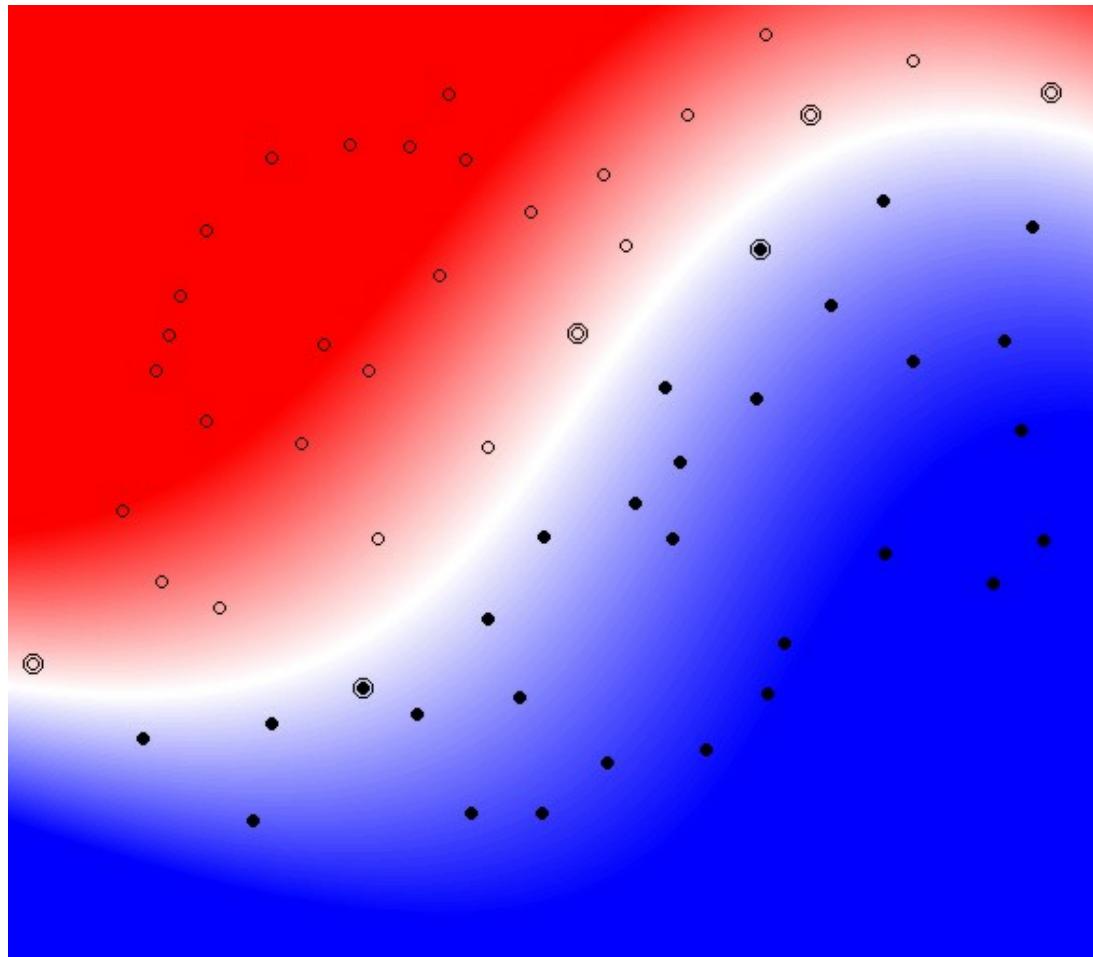
$\xi_i > 1$ Fehlklassifikation

$0 < \xi_i \leq 1$ Richtig,
geringer Abstand = *margin error*

$\xi_i = 0$ *margin vector*

Beispiel

nichtlinear trennbare Daten



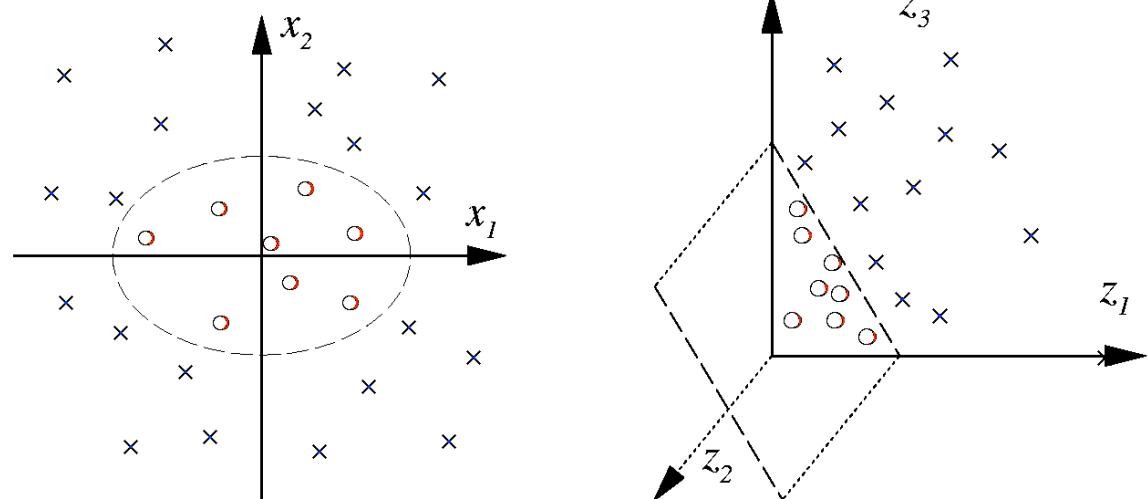
Nichtlineare Kernel-Methoden (SVM)

Idee: Transformiere die Daten in einen anderen Raum
 Bei Klassifikation: Lineare Trennung in diesem Raum

Beispiel $\Phi : R^n \rightarrow R^m$, meistens $m > n$

$$\begin{aligned}\Phi : \mathbb{R}^2 &\rightarrow \mathbb{R}^3 \\ (x_1, x_2) &\mapsto (z_1, z_2, z_3) := (x_1^2, \sqrt{2}x_1x_2, x_2^2)\end{aligned}$$

Monom-
Transformation



Problem: Transformation + Rechnen in hoch-dimensionalen Räumen = rechenintensiv

Kernel – Trick

Einige Form in der Daten vorkommen = Skalarprodukt

→ Kernelfunktion: $K(\vec{x}, \vec{y}) = \Phi(\vec{x}) \cdot \Phi(\vec{y})$

Berechnung soll einfach sein

alle bisherigen Ergebnisse bleiben gültig

Beispiel:

$$\begin{aligned}\Phi(\vec{x}) \cdot \Phi(\vec{y}) &= (x_1^2, \sqrt{2}x_1x_2, x_2^2)(y_1^2, \sqrt{2}y_1y_2, y_2^2) \\ &= (\vec{x} \cdot \vec{y})^2 \\ &=: K(\vec{x}, \vec{y})\end{aligned}$$

Kernel Funktionen

Skalarprodukt

$$K(\vec{x}, \vec{y}) = \vec{x} \cdot \vec{y}$$

Polynomial (Vovk)

$$K(\vec{x}, \vec{y}) = (\vec{x} \cdot \vec{y} + c)^d$$

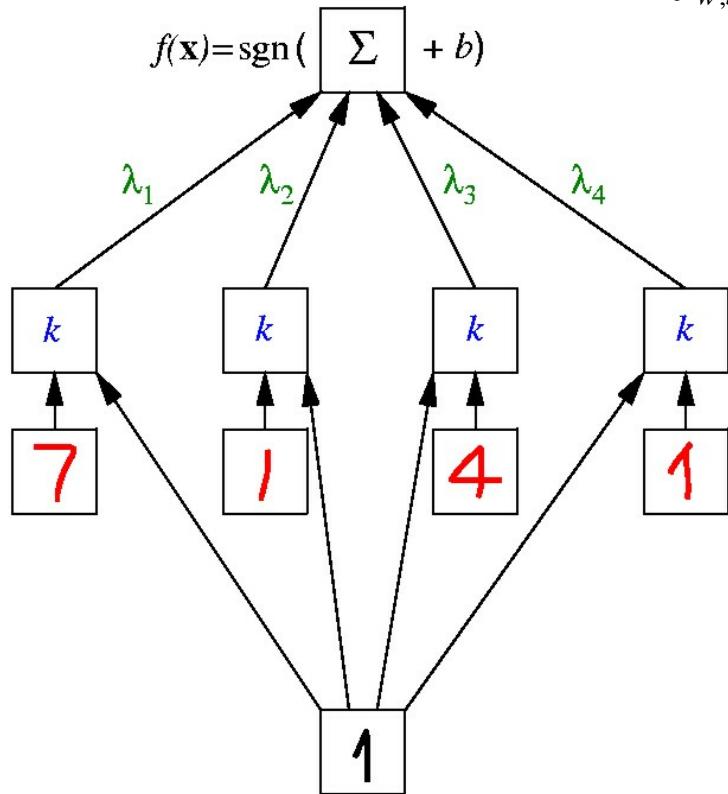
Radiale Basis-Funktionen (RBF)

$$K(\vec{x}, \vec{y}) = e^{-\frac{\|\vec{x}-\vec{y}\|^2}{2\sigma^2}}$$

Sigmoid (Ähnlichkeit mit FF - Neuronalen Netzen)

$$K(\vec{x}, \vec{y}) = \tanh(\kappa(\vec{x} \cdot \vec{y}) + \theta)$$

Ähnlichkeit mit einschichtigen Neuronalen Netzten ?!



$$f_{\vec{w},b}(\vec{x}) = \text{sgn}(\vec{w}\vec{x} + b) \quad \vec{w} = \sum_{i=1}^N \alpha_i y_i \vec{x}_i \quad \lambda_i = \alpha_i y_i$$

$$f(\mathbf{x}) = \text{sgn}(\Sigma \lambda_i k(\mathbf{x}, \mathbf{x}_i) + b)$$

weights

comparison: $k(\mathbf{x}, \mathbf{x}_i)$, e.g. $k(\mathbf{x}, \mathbf{x}_i) = (\mathbf{x} \cdot \mathbf{x}_i)^d$

$$k(\mathbf{x}, \mathbf{x}_i) = \exp(-\|\mathbf{x} - \mathbf{x}_i\|^2 / c)$$

$$k(\mathbf{x}, \mathbf{x}_i) = \tanh(\kappa(\mathbf{x} \cdot \mathbf{x}_i) + \theta)$$

support vectors
 $\mathbf{x}_1 \dots \mathbf{x}_4$

input vector \mathbf{x}

Im Bsp.: Klassifikation „Ziffer = 1“ (Ausgabe: ja / nein)

Beispiel

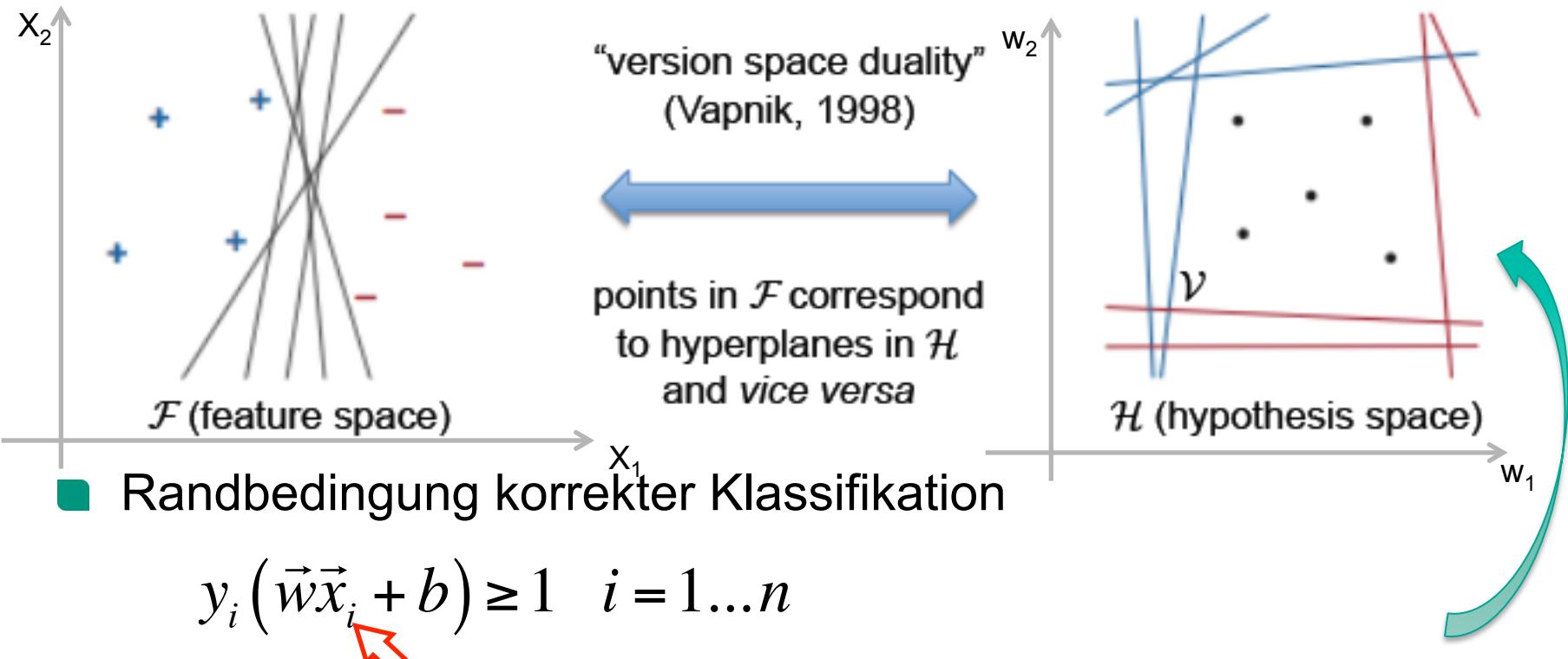
Verschiedene Kernels

z.B. LIBSVM

<http://www.csie.ntu.edu.tw/~cjlin/libsvm/>

Version Space für SVM

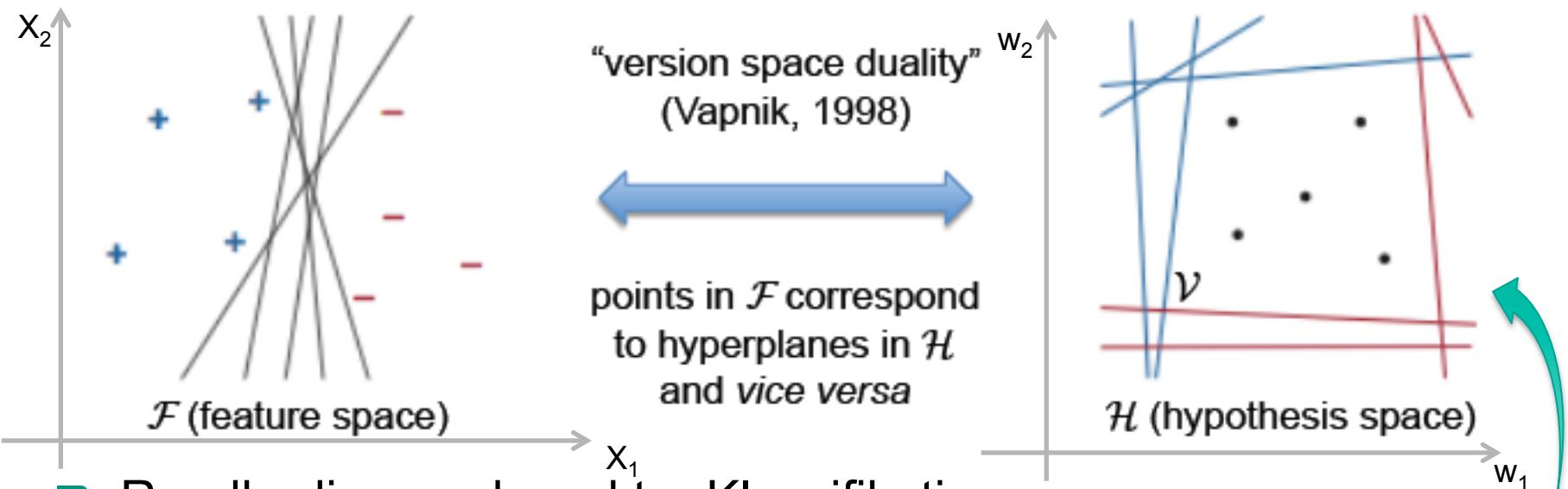
- Besondere Eigenschaft: Dualität Merkmal- u. Hypothesenraum



- Ungleichung kann nach \vec{w} im Merkmalsraum und nach \vec{x} im Hypothesenraum interpretiert werden

Version Space für SVM

- Besondere Eigenschaft: Dualität Merkmal- u. Hypothesenraum



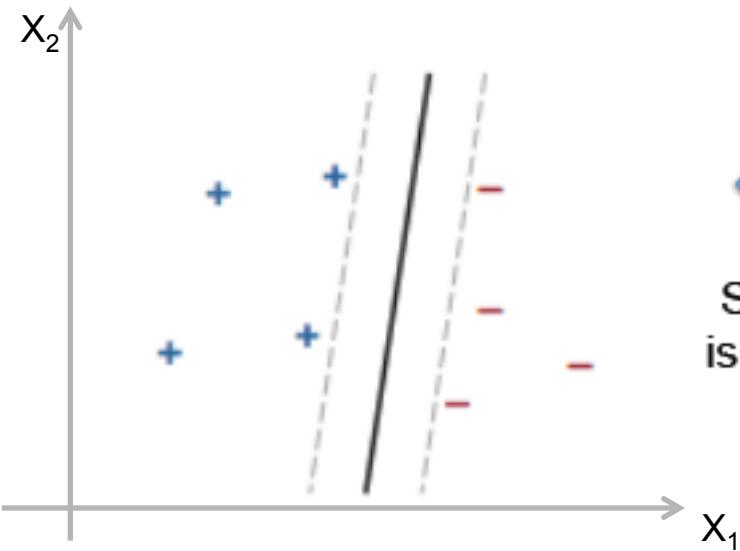
- Randbedingung korrekter Klassifikation

$$y_i (\vec{w} \vec{x}_i + b) \geq 1 \quad i = 1 \dots n$$

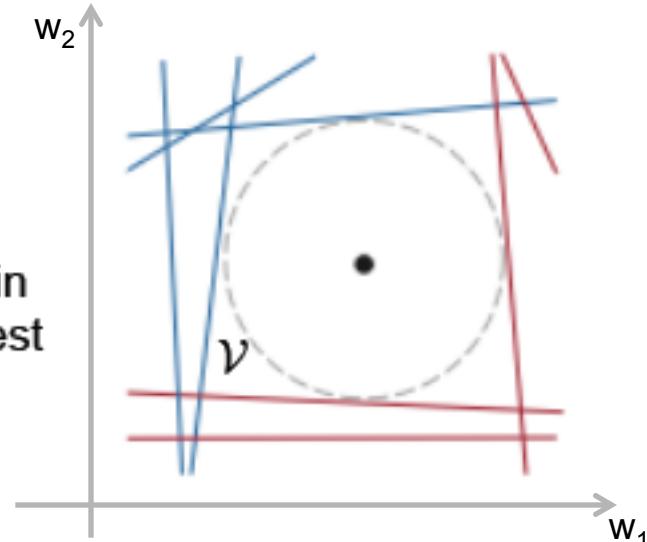
- D.h. aber auch, dass jeder Datenpunkt $x_i \quad i = 1 \dots n$ eine Hyperebene im Hypothesenraum definiert s.d. gültige \vec{w} jeweils in einem Halbraum = reduzierter Hypothesenraum sind

Version Space für SVM

■ Größter Rand

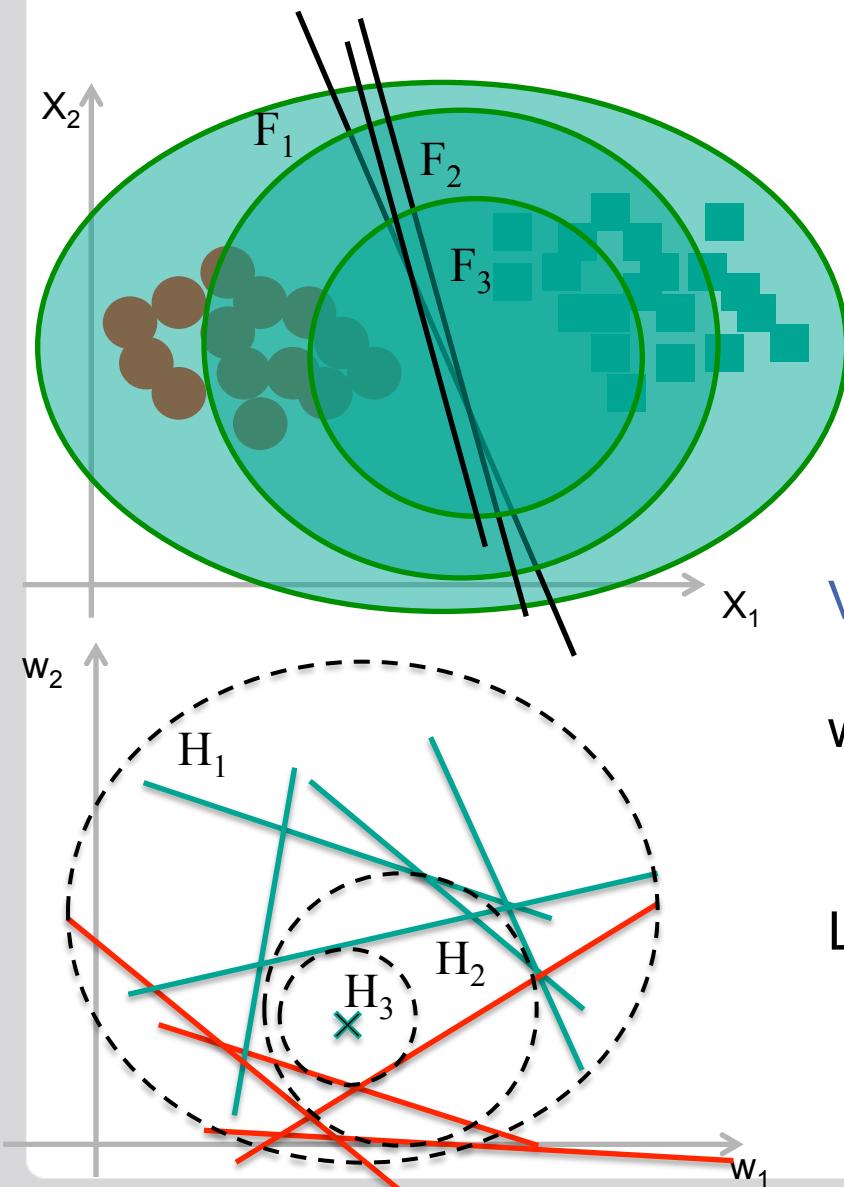


SVM with largest margin
is the center of the largest
hypersphere in \mathcal{V}



- Heißt gleichzeitig dass wir nach dem \vec{w} suchen, dass den maximalen Abstand zu allen Hyperebenen der Datenpunkte hat → Mittelpunkt der Hyperkugel

Wieso SRM bei SVM ?



- Suchraum für die Trennhyperebene wird während der Sattelpunkt – Suche eingeschränkt
- Irrelevante Daten fallen nicht mehr ins Gewicht

$$\text{VC- Dimension } h - \text{ fällt stetig: } h \leq D^2 |w|^2$$

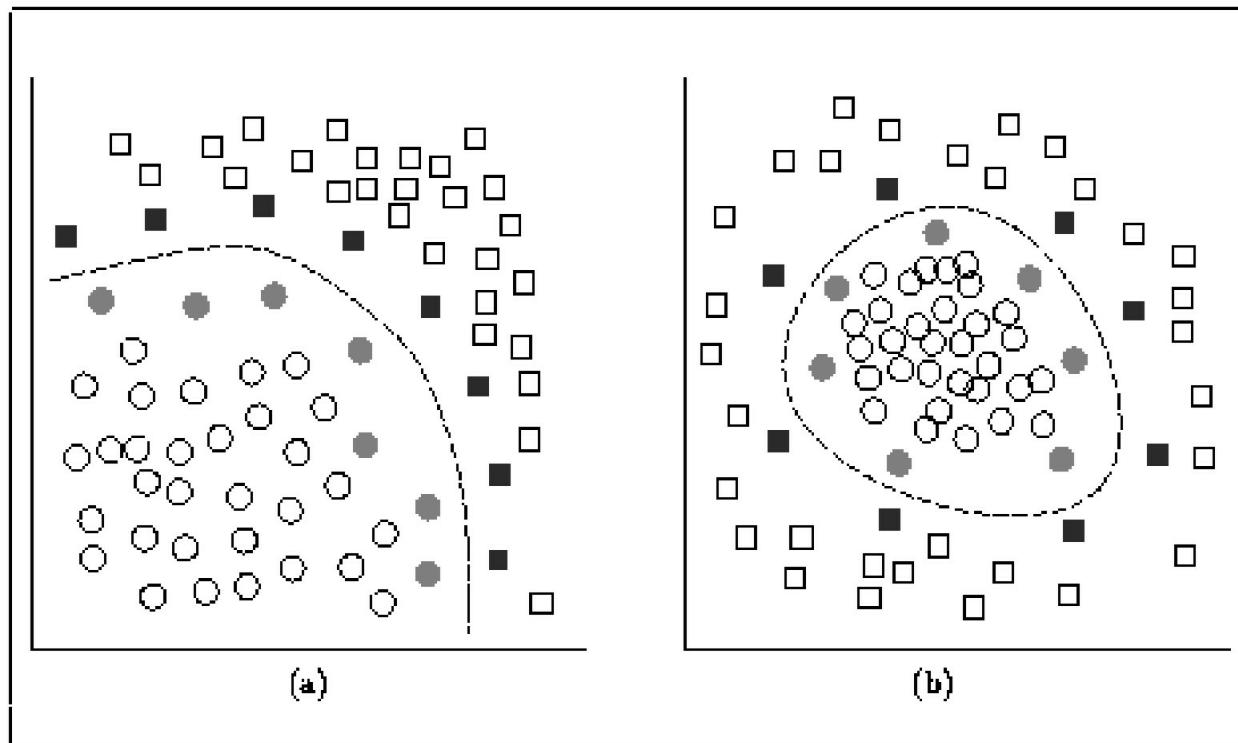
wobei: D – Radius der Hyperkugel gegeben

durch relevante Daten F im Hypothesenraum

Lösung = Mittelpunkt der Hyperkugel im Hypothesenraum

$$H_3 \subset H_2 \subset H_1$$

Eigenschaften der SVM

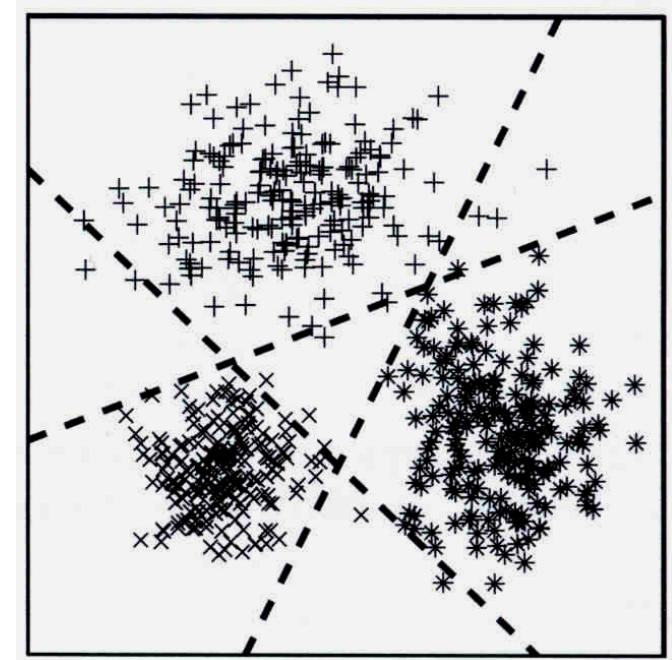


Kleine Anzahl (regulierbar) von Support Vektoren
Entscheidung wird anhand der Randregionen gefällt

(Erweiterung reduced - RSVM)

Multiple Klassifikation (k Klassen)

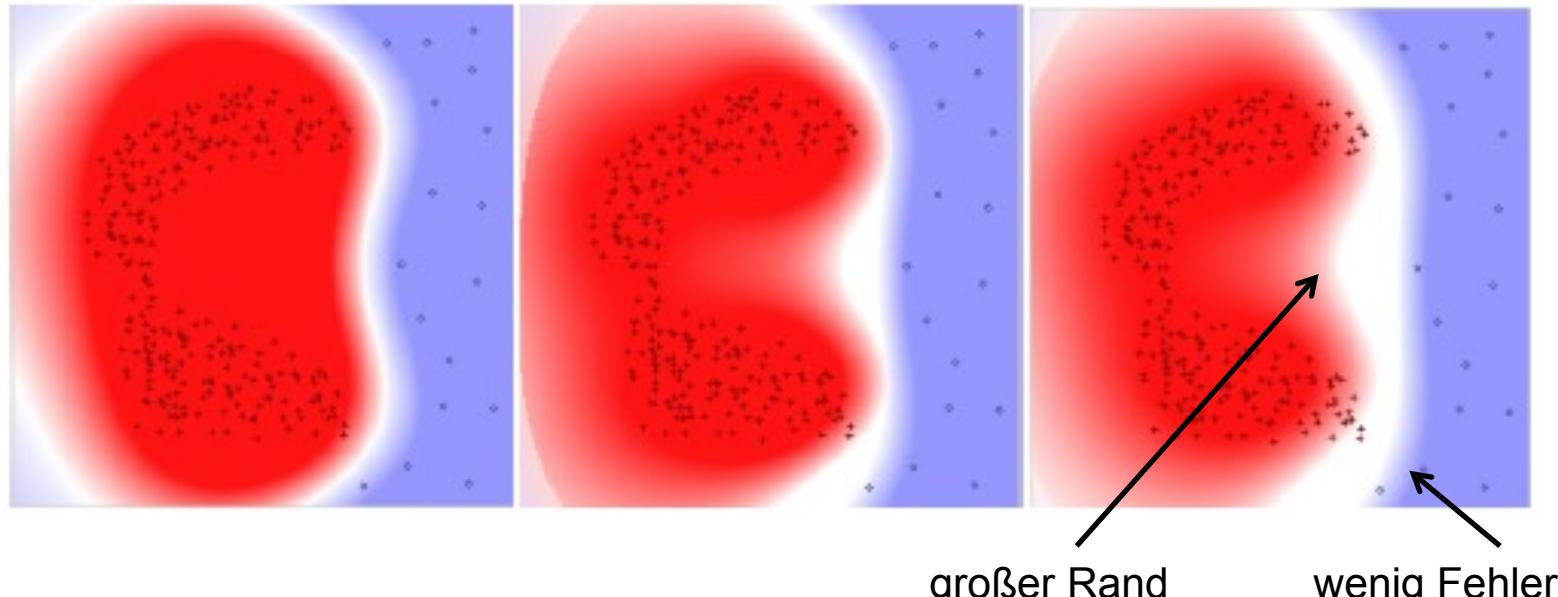
- Einer – gegen – Alle:
 k SVM's (für jede Klasse eine)
Abstimmungsverfahren
- Einer – gegen – Einen:
 $k(k-1)/2$ SVM's
Abstimmungsverfahren
- Mehrfachzugehörigkeit (Multiple)
 k SVM's (für jede Klasse eine)
Abstimmungsverfahren
- k – class SVM von Watkins
kein Abstimmungsverfahren



Gewichtete SVM

$$\text{Minimiere } \min_{\vec{w}, b, \xi_i} \frac{1}{2} \|\vec{w}\|^2 + C_+ \left(\sum_{y_i=+1, i=1}^m \xi_i \right)^p + C_- \left(\sum_{y_i=-1, i=1}^m \xi_i \right)^p$$

je ein C - für „positive und negative“ Klasse



$$(C_-, C_+) = (10, 10)$$

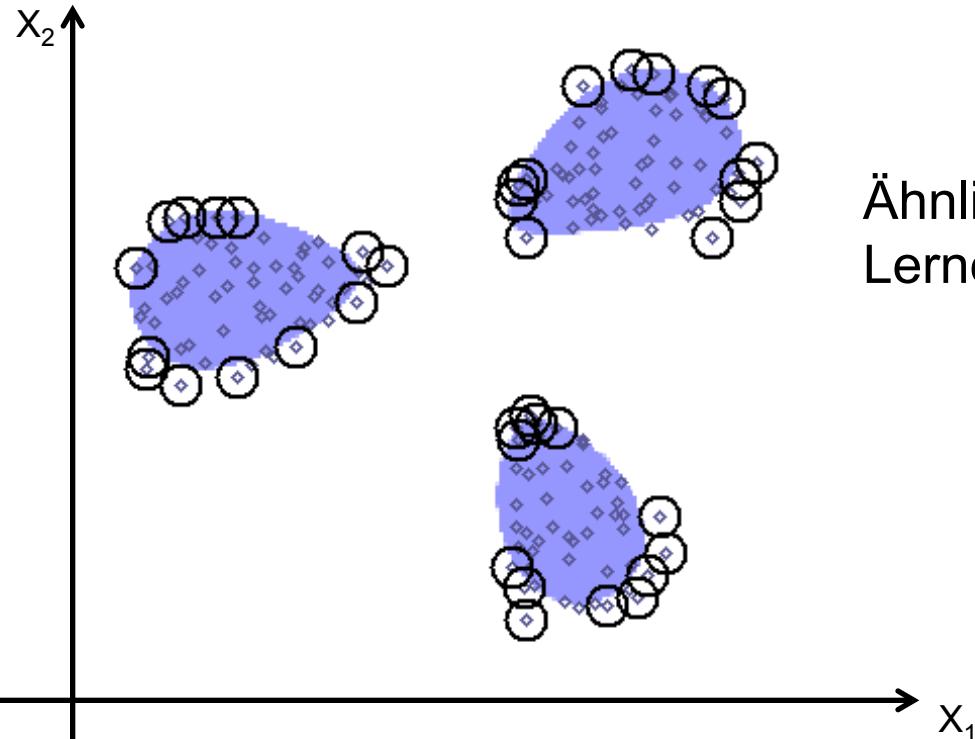
$$(1, 10)$$

$$(0.1, 10)$$

Dichte – Träger Schätzung

Gesucht:

- Eine Funktion f , die für eine „kleine“ Region, welche die meisten Lernbeispiele enthält, den Wert >0 und sonst den Wert 0 (oder <0) annimmt.



Ähnlichkeit mit unüberwachtem
Lernen (Clustering)

SVM Dichte – Träger Schätzung

Idee:

- Trennung der Lernbeispiele vom Ursprung im transformierten Merkmalsraum
- Entspricht nicht-linearem Schätzer im Eingaberaum

Lösung:

- Primäres Optimierungsproblem:

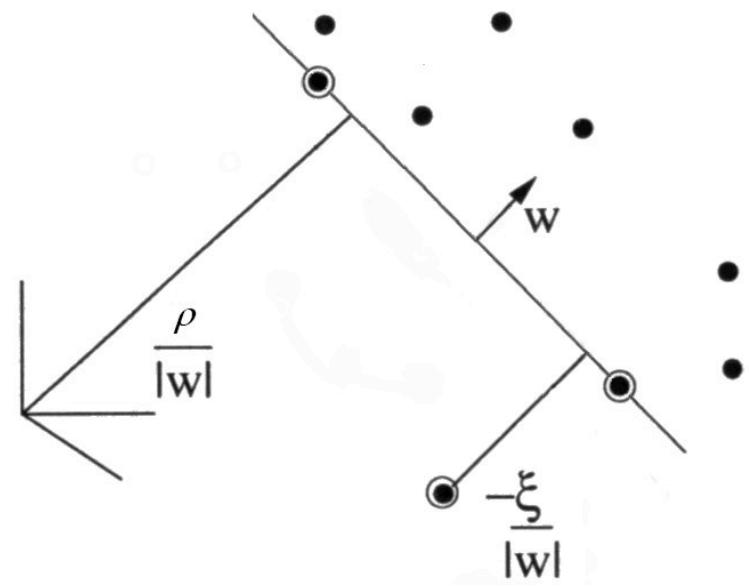
$$\min_{\vec{w}, b, \xi_i} \frac{1}{2} \|\vec{w}\|^2 + \frac{1}{vm} \sum_{i=1}^m \xi_i - \rho$$

unter den Bedingungen

$$\vec{w} \cdot \mathbf{x}_i \geq \rho - \xi_i \quad i = 1, \dots, m$$

$$\xi_i \geq 0 \quad i = 1, \dots, m$$

- Weitere Lösung analog zur Klassifikation



Weitere Kernel Methoden

- Einfacher Algorithmus kann oft als Kernel Methode realisiert werden
- Gegeben Vektoren \vec{x}, \vec{y} und eine Transformation $\phi(\vec{x}), \phi(\vec{y})$ in einen anderen Raum
- Dann ist ein Kernel:
$$K(\vec{x}, \vec{y}) = \langle \phi(\vec{x}), \phi(\vec{y}) \rangle$$
- Damit lassen sich oft mit einfachen Methoden im transformierten Raum komplexe Probleme im Ursprungsraum lösen

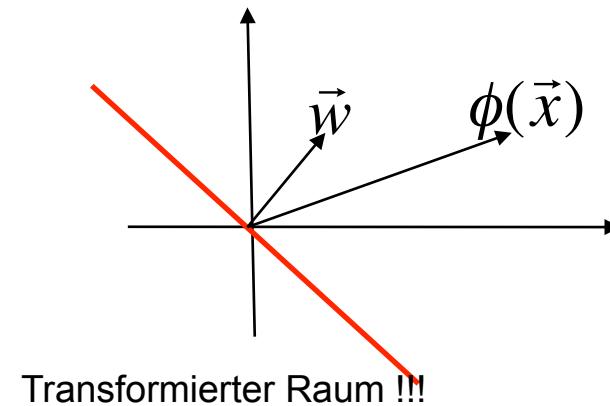
Weitere Kernel Methoden

- Einfacher Algorithmus kann oft als Kernel Methode realisiert werden

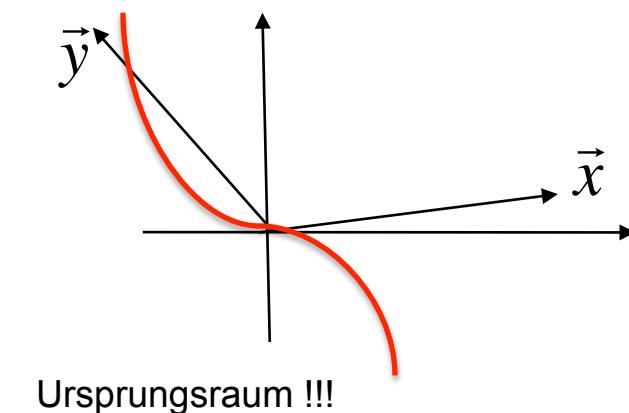
Perzepton Algorithmus

- Erinnerung: realisiert Trennung an Hyperebene

$$h(\vec{x}) = \text{sign} \langle \vec{w}, \phi(\vec{x}) \rangle$$



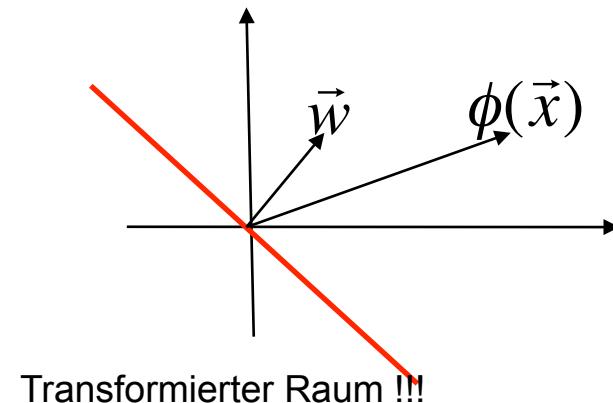
- Lineare Trennung im transformierten Raum führt zu komplexer Trennung im Ursprungsraum



Weitere Kernel Methoden

- Einfacher Algorithmus kann oft als Kernel Methode realisiert werden

Perzepton Algorithmus



- Erinnerung: realisiert Trennung an Hyperebene

$$h(\vec{x}) = \text{sign} \langle \vec{w}, \phi(\vec{x}) \rangle$$

- Lernen (im transformierten Raum)

- falsch klassif. Beispiele $(\vec{x}_i, y_i) \rightarrow$ update (Addieren/Subtrahieren) :

$$\vec{w}_{t+1} = \vec{w}_t + y_i \phi(\vec{x}_i) \Rightarrow \vec{w}_t = \sum_{i=1}^l \alpha_i y_i \phi(\vec{x}_i), \quad \alpha_i \in N, y_i = \pm 1$$

Kernel Perceptron

- Gegeben eine Kernel - Funktion K ergibt sich der einfache Algorithmus:

$$\vec{\alpha} = 0, i = 0$$

repeat

$$i = i + 1$$

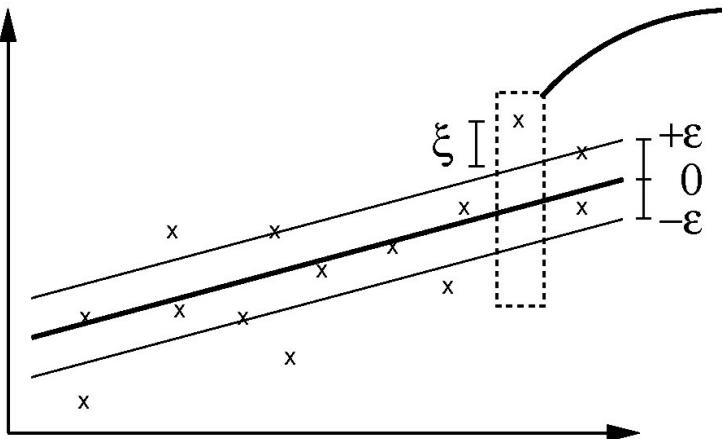
$$\text{if } \text{sign}\left(\sum_{j=1}^l \alpha_j y_j K(\vec{x}_j, \vec{x}_i)\right) \neq y_i$$

$$\alpha_i = \alpha_i + 1$$

until finished

$$f(\vec{x}) = \sum_{j=1}^l \alpha_j y_j K(\vec{x}_j, \vec{x})$$

Weitere Anwendungen der SVM



Lineare – Regression

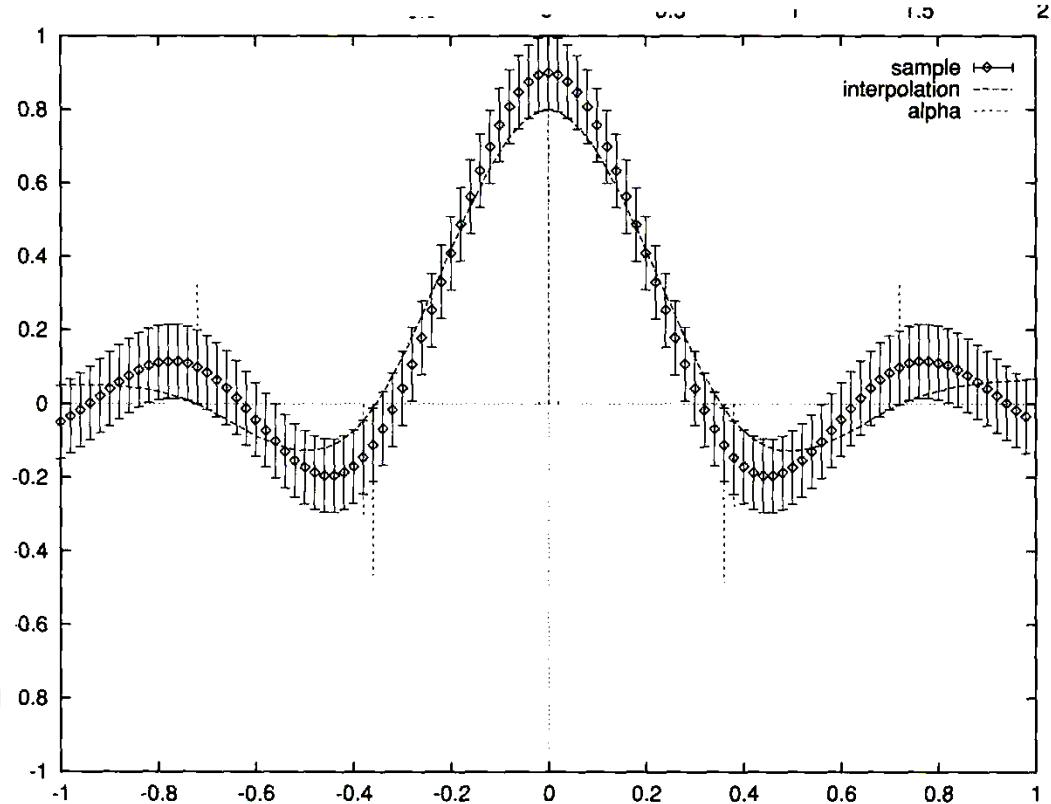
+

Randbedingung = Epsilon-schlauch

+

Transformation

SVM – Regression



Weitere Anwendungen der SVM

Nichtlineare Kernel - PCA

	Gaussian noise	'speckle' noise
orig.		
noisy		
$n = 1$		
4		
16		
64		
256		
$n = 1$		
4		
16		
64		
256		

linear PCA
reconstruction

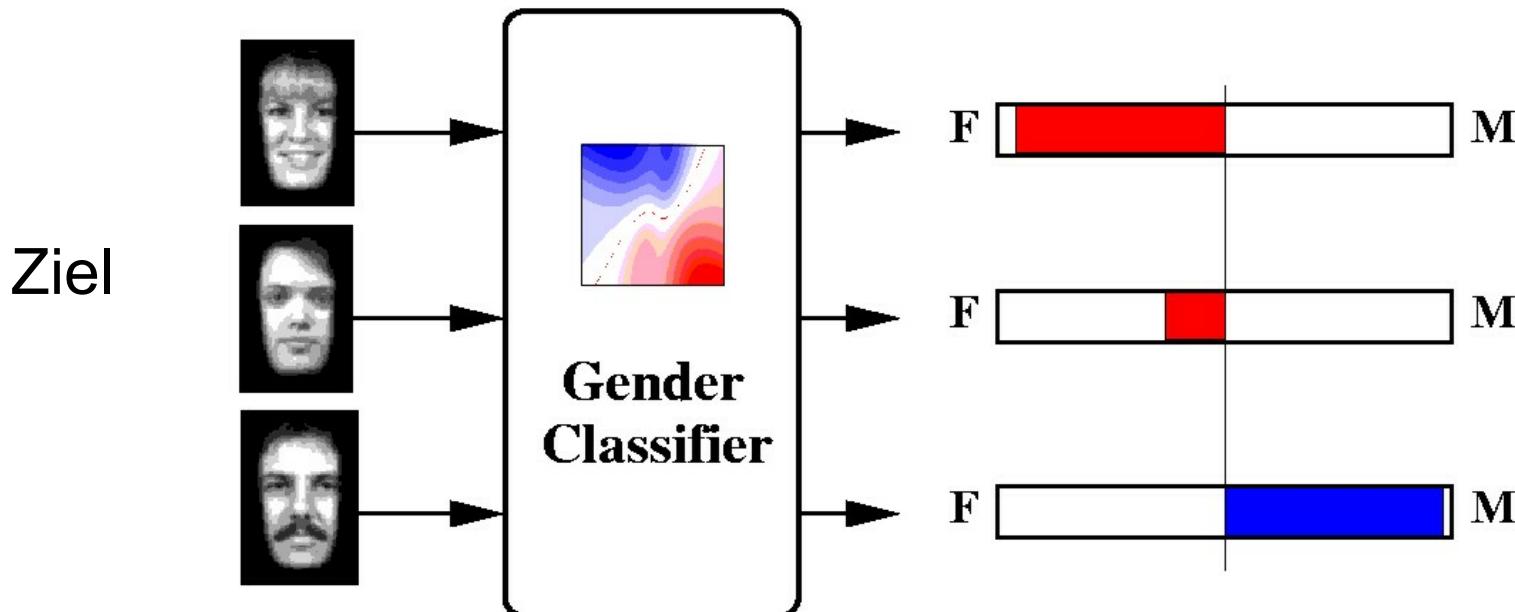
kernel PCA
reconstruction

Another application: face modelling (Romdhani et al., 1999).

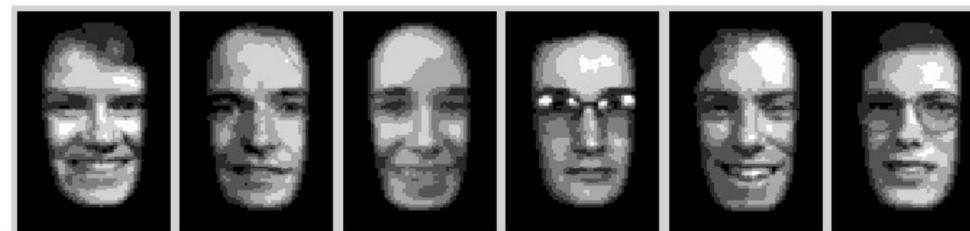
Gesichtserkennung
Gender Classification (einfache Bsp.)
Fahrerassistenz
Robotik

Pro & Kontra SVM
Offene Fragen bei SVM

SVM – Gender Classification



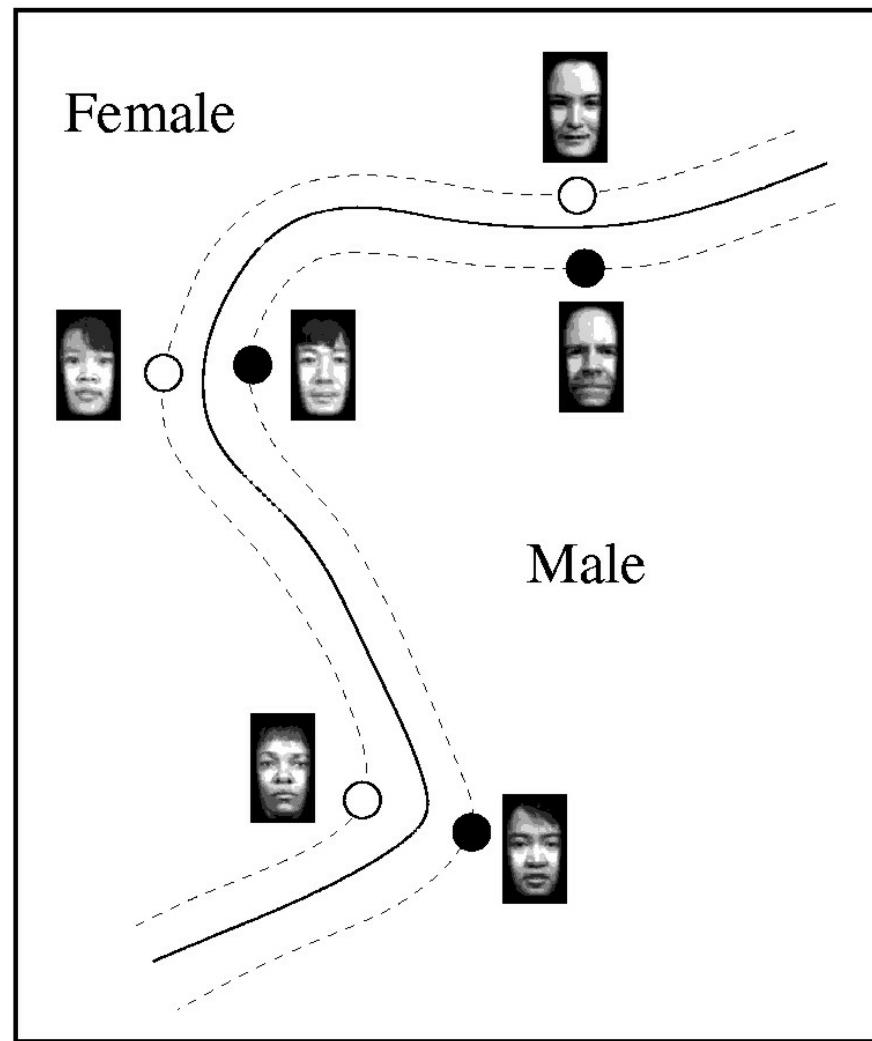
Lerndaten



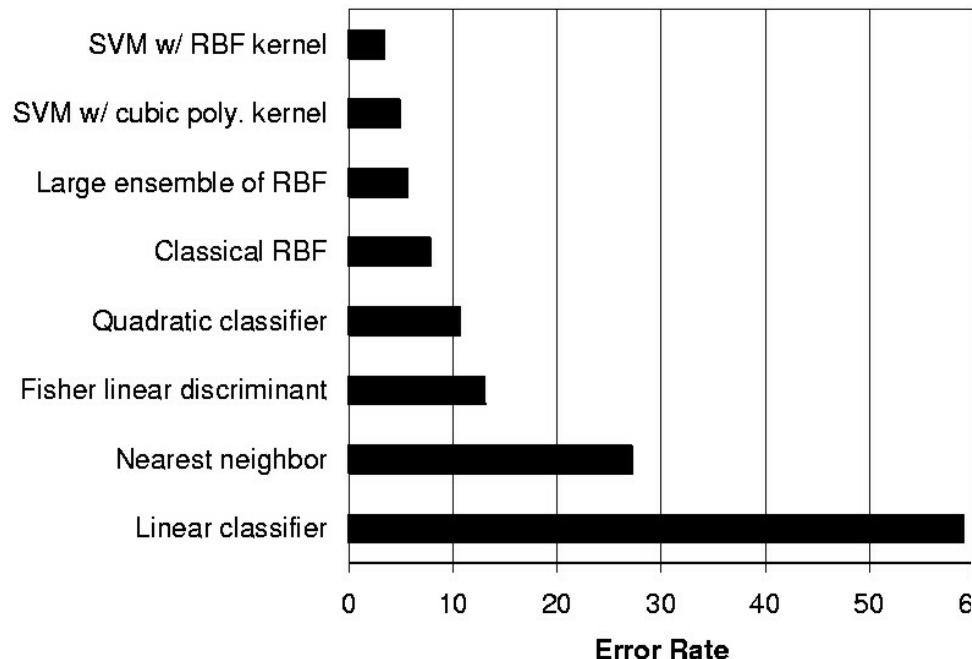
SVM – Gender Classification

Support – Vektoren

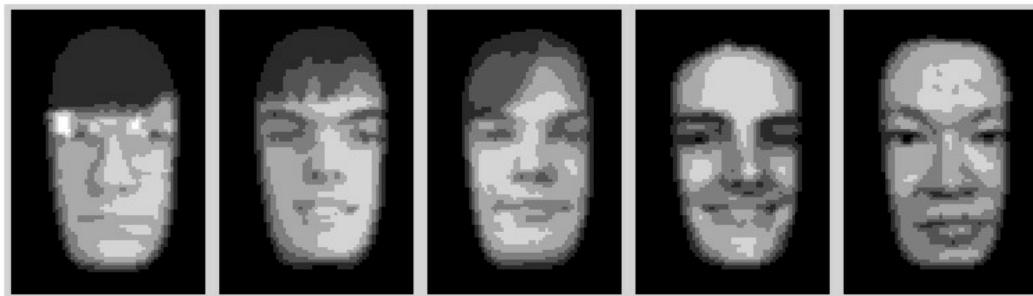
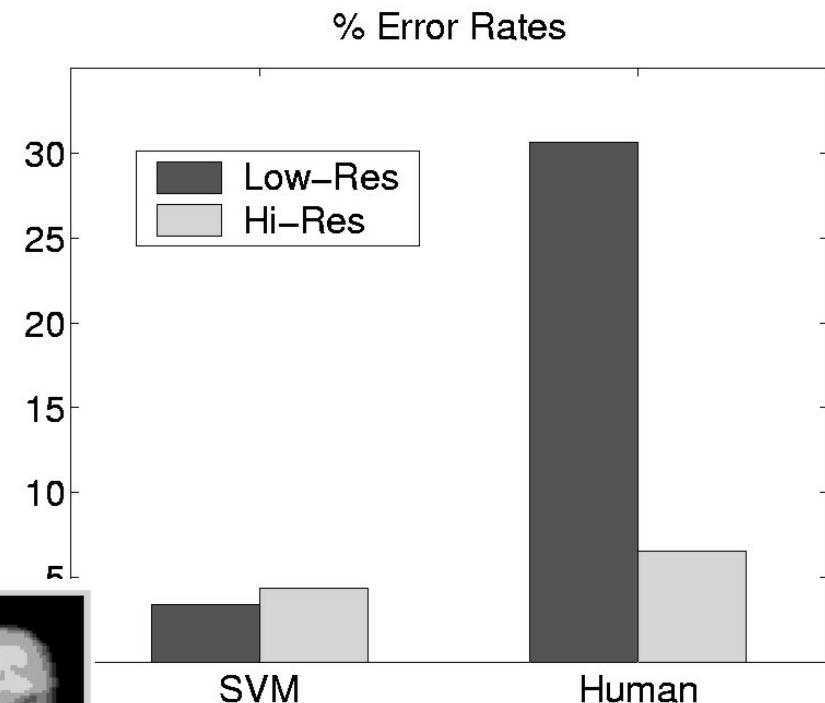
Entsprechend ihrem Abstand
von der Trenneben



SVM – Gender Classification



Ergebnisse



Fehler: Mensch bei hohem Rauschanteil schlechter

SVM - Gesichtserkennung

Ziel: Erkennen von Gesichter bzw. Teile davon in einem Bild

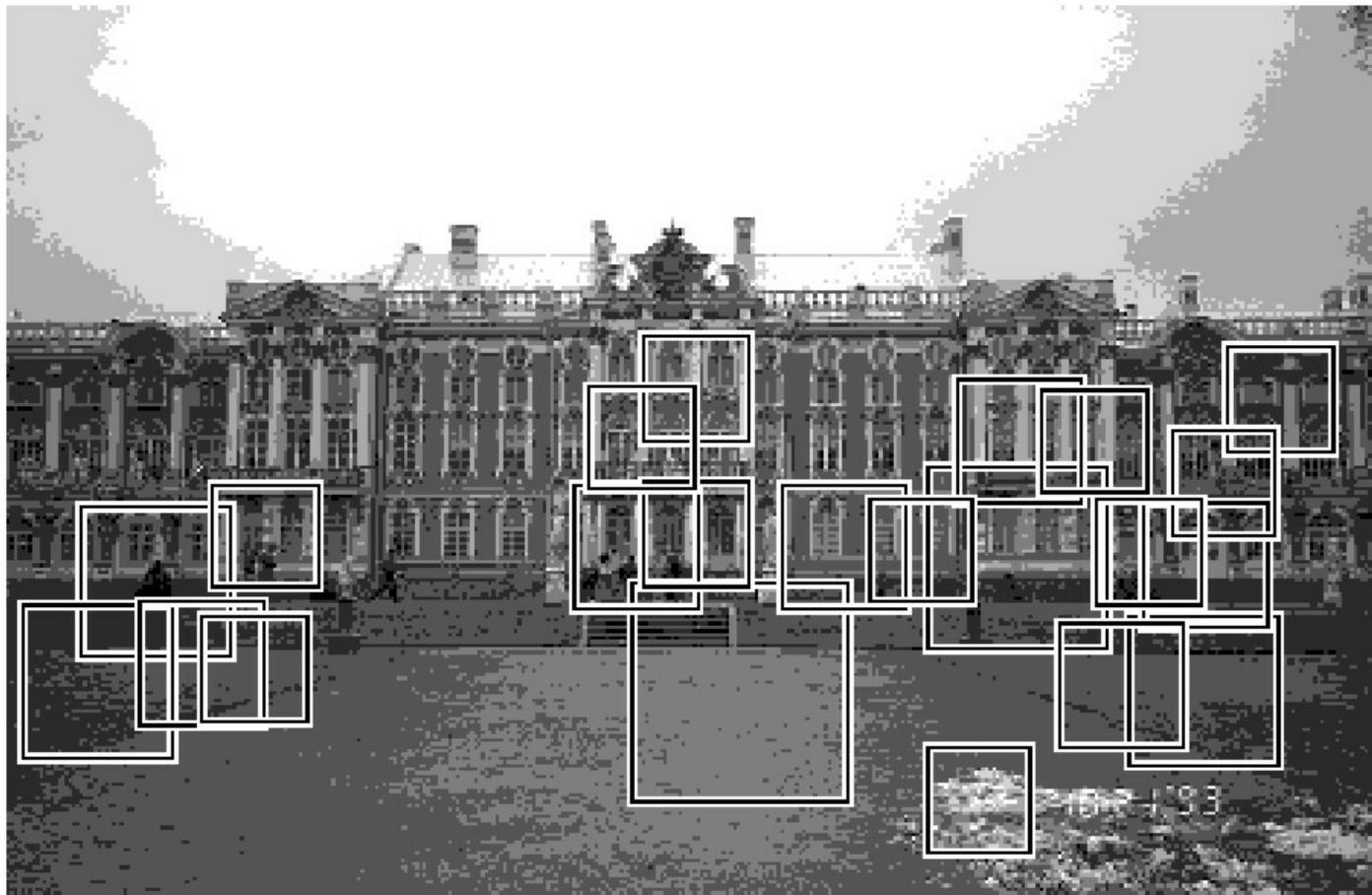
Lerndaten als 19x19 Pixmaps



Verrauschte Lerndaten



SVM - Gesichtserkennung



Lerndaten, keine Gesichter

SVM - Gesichtserkennung



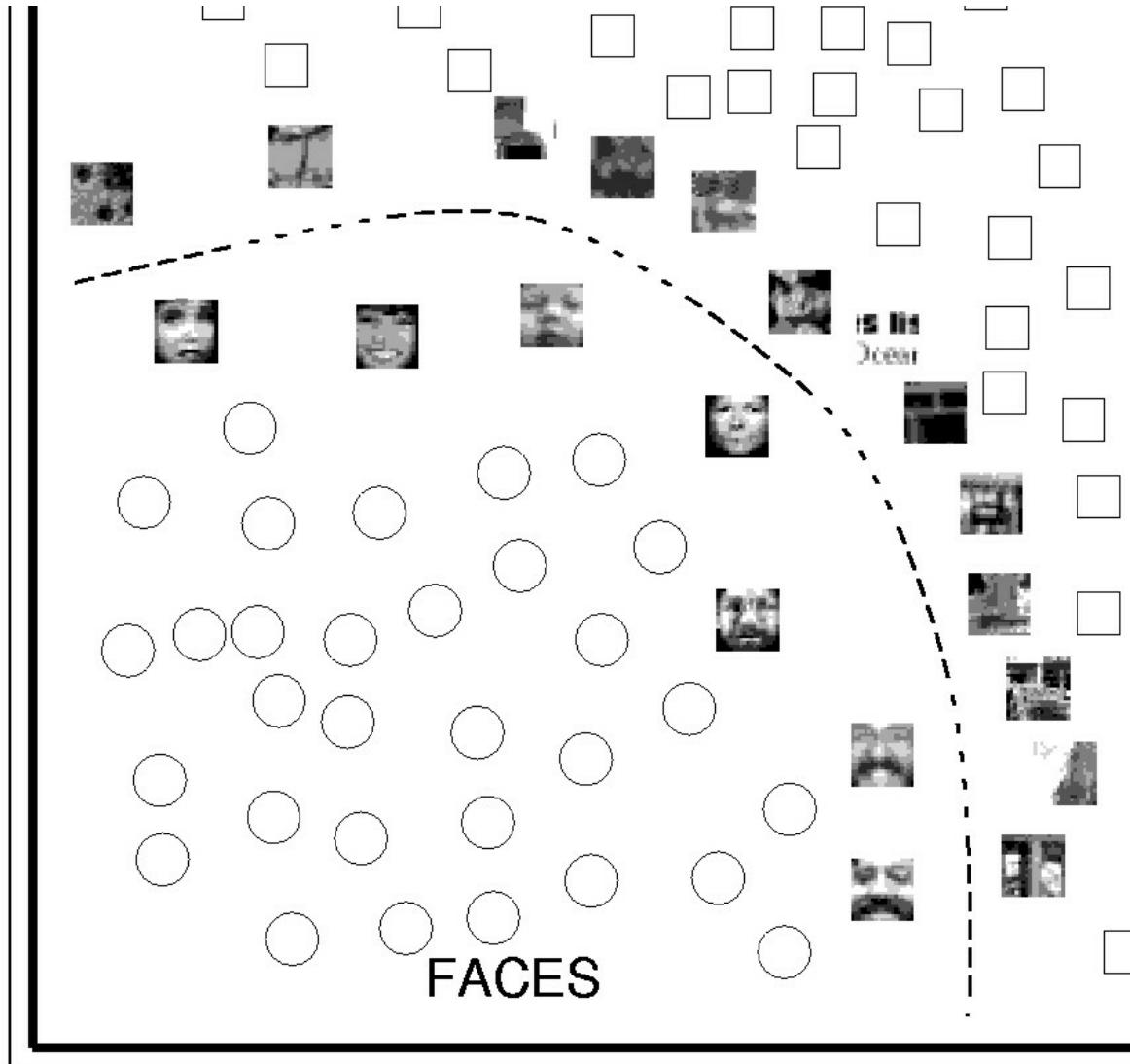
Erkennung von
gedrehten
Gesichtern
→ entspr. Lerndaten
müssen erzeugt
werden

SVM - Gesichtserkennung



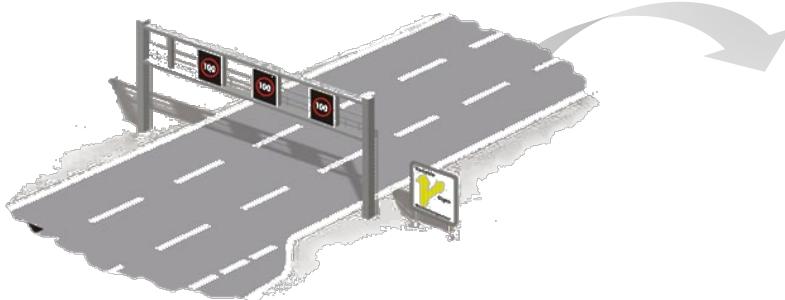
Ergebnisse

SVM - Gesichtserkennung



Support Vektoren
Entsprechend ihrer
Distanz zur Trenn-
ebene eingezeichnet

Erkennung v. Verkehrszeichen



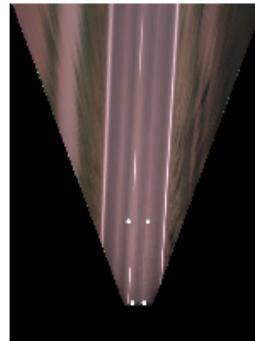
- Vorverarbeitung
 - Farbverbesserung
- Segmentierung
 - Hough-Transformation
 - Entfernung von Flackern
- Tracking
- Klassifikation
 - Vorverarbeitung
 - SVM mit RBF Kernel, ca. 20 Klassen
- Temporale Fusion
- Erkennungsrate >> 95%



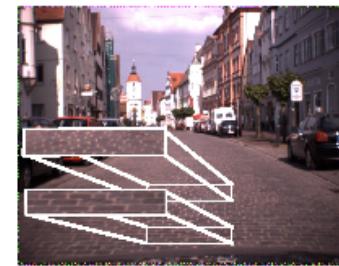
Erkennung von Verkehrsflächen



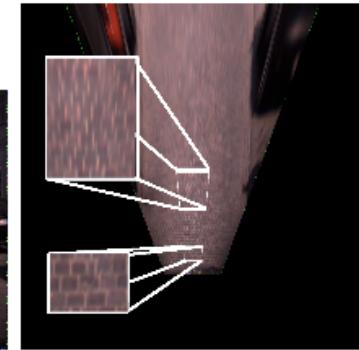
(a) Original Farbbild



(b) V



(c)



(d)

Bildanpassung

- Transformation
- Segmentierung

Merkmalsberechnung

- Farb- und
- Texturmerkmale

2-5 Klassen

- Strasse
- Hintergrund
- Urban
- Teer
- Natur

Güte: 70 - 95%



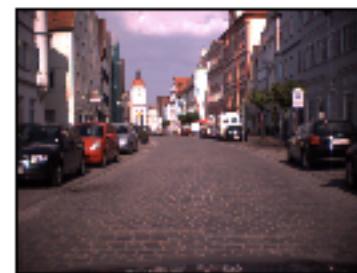
(a) Kamerabild einer Überlandfahrt



(b) Klassifikationsergebnis bei Verwendung von 5 Klassen



(c) Klassifikationsergebnis bei Verwendung von 2 Klassen



(d) Kamerabild einer Stadtszene mit Kopfsteinpflaster



(e) Klassifikationsergebnis bei Verwendung von 5 Klassen



(f) Klassifikationsergebnis bei Verwendung von 2 Klassen

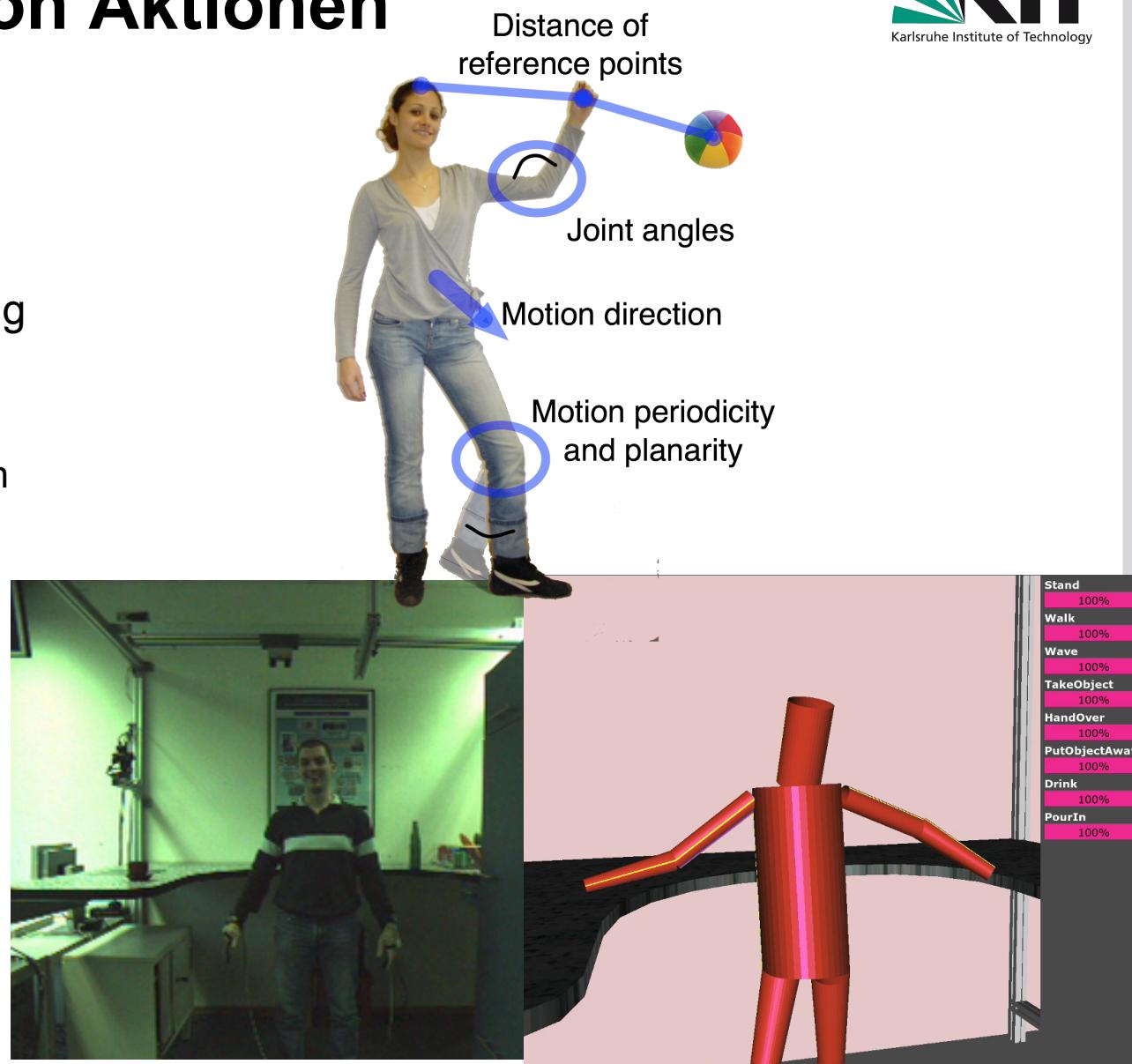
Wetterklassifikation

- Vorverarbeitung
- Merkmalsberechnung
 - Intensität / Sättigung
 - Kontrast
 - Schärfe
 - ...
- Klassifikation
 - 5 Klassen
 - Gutes Wetter
 - Regen
 - Nebel
 - ...
 - Güte: ca 85%
- Bildkorrektur



Erkennung von Aktionen

- Modellierung
- Tracking
- Merkmalsberechnung
 - Zeitfenster
 - Gelenkwinkel
 - Geschwindigkeiten
 - ...
- Klassifikation
 - 20 Klassen
 - Güte : ca. 80-90%



Pro und Kontra SVM

Pro:

- Optimale Hyperebene → gute Lernergebnisse
- Finden der optimalen VC-Dimension → korrektes Lernen
- Verarbeitung hochdimensionaler Daten
- Anwendungsspezifische Kernels (mit Datenverarbeitung)
- Schnelle Auswertung
- Viele Anwendungen: Klassifikation, Regression, PCA
- Probabilistische Sicht!? → siehe ML II
- Semi-überwachtes Lernen → siehe ML II
- Aktives Lernen → siehe ML II

Kontra:

- Vorverarbeitung extern (kein „tiefes“ Lernen)
- Finden des optimalen Kernels – aktuelle Forschung
- Parametrisierung des Kernels – aktuelle Forschung
- Speicher und Rechenaufwand (Trainieren)
- Anzahl der SV abh. von Problem und Parameter

Literatur

- *Libsvm*: <https://www.csie.ntu.edu.tw/~cjlin/libsvm/>
- *V.N. Vapnik*: **Statistical Learning Theory**. Wiley, 1998.
- *B. Schölkopf*: **Support Vector Learning**.
- Webseite: <http://www.kernel-machines.org> . (leider seit längerem nicht mehr gepflegt)
- B. Moghaddam and M.-H. Yang. Gender Classification with Support Vector Machines. Proceedings of the International Conference on Automatic Face and Gesture Recognition(FG), 2000