Vorlesung Wissensentdeckung

SVM – SMO, Kernfunktionen, Anwendungen

Katharina Morik, Uwe Ligges

LS 8 Informatik
Computergestützte Statistik
Technische Universität Dortmund

15.6.2010

Gliederung

Optimierungsproblem der SVM

Die Lösung $\vec{\alpha}^*$ des dualen Problems

$$L_D(\vec{\alpha}) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} y_i y_j \alpha_i \alpha_j \langle \vec{x}_i, \vec{x}_j \rangle$$

muss die KKT-Bedingungen erfüllen, d.h. es gilt unter anderem

$$\alpha_i \left(y_i \left(\langle \vec{x}_i, \vec{\beta} \rangle + \beta_0 \right) - 1 \right) = 0 \; \forall \, i = 1, \ldots, N$$

$\vec{\alpha}^*$ enthält für jedes Beispiel $\vec{x}_i$ genau ein α_i mit

$\alpha_i = 0$, falls $\vec{x}_i$ im richtigen Halbraum liegt

$\alpha_i > 0$, falls $\vec{x}_i$ auf der Hyperebene H_1 oder H_2 liegt

Ein Beispiel $\vec{x}_i$ mit $\alpha_i > 0$ heißt Stützvektor.

Optimierungsproblem für weiche Trennung

Sei $C \in \mathbb{R}$ mit $C > 0$ fest. Minimiere

$$||\vec{\beta}||^2 + C \sum_{i=1}^{N} \xi_i$$

unter den Nebenbedingungen

$$\begin{aligned} \langle \vec{x}_i, \vec{\beta} \rangle + \beta_0 &\geq +1 - \xi_i \quad \text{für } \vec{y}_i = +1 \\ \langle \vec{x}_i, \vec{\beta} \rangle + \beta_0 &\leq -1 + \xi_i \quad \text{für } \vec{y}_i = -1 \end{aligned}$$

Optimierungsproblem zur Minimierung

- Erst minimierten wir $\vec{\beta}$ (primales Problem), dann maximierten wir α (duales Problem), jetzt minimieren wir das duale Problem, indem wir alles mit -1 multiplizieren...
- Minimiere $L'_D(\alpha)$

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

unter den Nebenbedingungen $0 \leq \alpha_i \leq C$

$$\sum_{i=1}^{m} y_i\alpha_i = 0$$

Algorithmus?

- Berechnen wir $L'_D(\alpha)$ durch Gradientensuche!
  - Naiver Ansatz berechnet Gradienten an einem Startpunkt und sucht in angegebener Richtung ... Bis kleinster Wert gefunden ist. Dabei wird immer die Nebenbedingung eingehalten. Bei m Beispielen hat α m Komponenten, nach denen es optimiert werden muss. Alle Komponenten von α auf einmal optimieren? m^2 Terme!
  - Eine Komponente von α ändern? Nebenbedingung verletzt.
  - Zwei Komponenten α_1, α_2 im Bereich $[0, C] \times [0, C]$ verändern!

Sequential Minimal Optimization

- Wir verändern α_1, α_2 , lassen alle anderen α_i fest. Die Nebenbedingung wird zu:

$$\alpha_1 y_i + \alpha_2 y_2 = -\sum_{i=3}^{m} \alpha_i y_i$$

- Zulässige α_1, α_2 liegen im Bereich $[0, C] \times [0, C]$ auf der Geraden
  $W = \alpha_1 y_1 + \alpha_2 y_2$ äquivalent $\alpha_1 + s\alpha_2$ mit $s = \frac{y_2}{y_1}$
- Wir optimieren α_2
- Aus dem optimalen $\hat{\alpha}_2$ können wir das optimale $\hat{\alpha}_1$ herleiten:

$$\hat{\alpha}_1 = \alpha_1 + y_1 y_2 (\alpha_2 - \hat{\alpha}_2)$$

- Dann kommen die nächsten zwei α_i dran...

α_2 optimieren

- Maximum der Funktion $L'_D(\alpha)$ entlang der Geraden $s\alpha_2 + \alpha_1 = d$.
- Wenn $y_1 = y_2$ ist $s = 1$, also steigt die Gerade. Sonst $s = -1$, also fällt die Gerade.
- Schnittpunkte der Geraden mit dem Bereich $[0, C] \times [0, C]$:
  - Falls s steigt: $\max(0; \alpha_2 + \alpha_1 - C)$ und $\min(C; \alpha_2 + \alpha_1)$
  - Sonst: $\max(0; \alpha_2 - \alpha_1)$ und $\min(C; \alpha_2 - \alpha_1 + C)$
  - Optimales α_2 ist höchstens $\max$-Term, mindestens $\min$-Term.

Bestimmen der αs

- $k = \alpha_1^{old} + s\alpha_2^{old} = \alpha_1^{new} + s\alpha_2^{new}$
- Mit Hilfe der Optimierungsquadrate lassen sich untere und obere Schranken für α_2 bestimmen:
  - $y_1 = y_2 : L = max(0, \alpha_1^{old} + \alpha_2^{old} - C)\ H = min(C, \alpha_1^{old} + \alpha_2^{old})$
  - $y_1 \neq y_2 : L = max(0, \alpha_2^{old} - \alpha_1^{old})\ H = min(C, C + \alpha_2^{old} - \alpha_1^{old})$
- Ableiten des Dualen Problems nach α_2 ergibt das Optimum für α_2^{new}
  - $\alpha_2^{new} = \alpha_2^{old} + \frac{y_2((f(\vec{x_1}) - y_1) - (f(\vec{x_2}) - y_2))}{\eta}$
  - $= \alpha_2^{old} + \frac{y_2(E_1 - E_2)}{\eta}$
  - $\eta = x_1^T x_1 + x_2^T x_2 - 2x_1^T x_2$

Optimales α_2

- Sei $\alpha = (\alpha_1, \ldots, \alpha_N)$ eine Lösung des Optimierungsproblems. Wir wählen zum update:

$$\hat{\alpha}_2 = \alpha_2 + \frac{y_2 \left((f(x_1) - y_1) - (f(x_2) - y_2) \right)}{K(x_1, x_1) - 2K(x_1, x_2) + K(x_2, x_2)}$$

- Optimales $\hat{\alpha}_1 = \alpha_1 + y_1 y_2 \left(\alpha_2 - \hat{\alpha}_2 \right)$
- Prinzip des Optimierens: Nullsetzen der ersten Ableitung...

Optimierungsalgorithmus

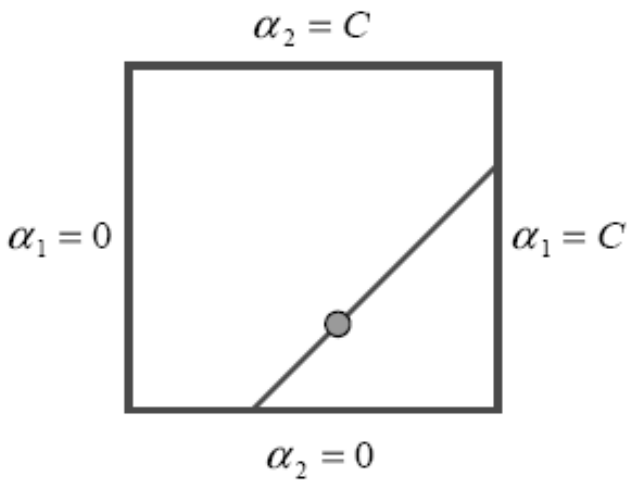
1: g = Gradient von $L'_D(\alpha)$
2: WHILE nicht konvergiert(g)
3: WS=working set(g)
4: α'=optimiere(WS)
5: g=aktualisiere(g, α')

1: $g_i = \sum \alpha_k y_k y_i (x_k * x_i) - 1$
2: auf ϵ genau
3: suche k “gute” Variablen
4: k neue α-Werte (update)
5: g = Gradient von $L'_D(\alpha')$

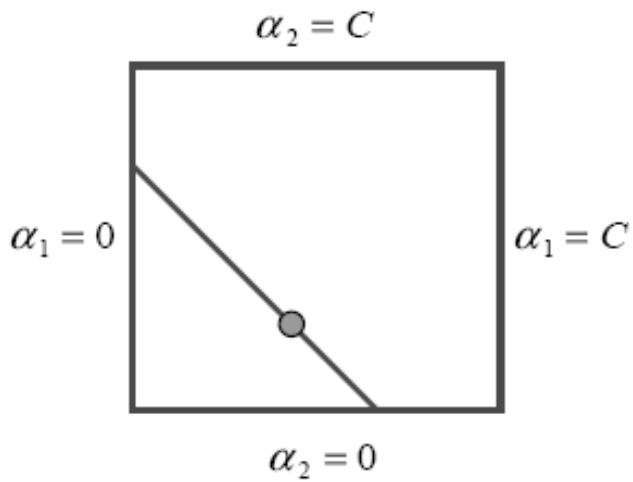
- Gradientensuchverfahren
- Stützvektoren allein definieren die Lösung
- Tricks: Shrinking und Caching von $x_i * x_j$

Ermitteln der αs im Bild

- Alle αs zu optimieren ist zu komplex.
- Nur ein α zur Zeit zu optimieren, verletzt $0 = \sum_{i=1}^{N} \alpha_i y_i$
- Also: zwei αs gleichzeitig optimieren!
- Man optimiert beide innerhalb eines Quadrates...



$y_1 \neq y_2 \Rightarrow \alpha_1 - \alpha_2 = k$

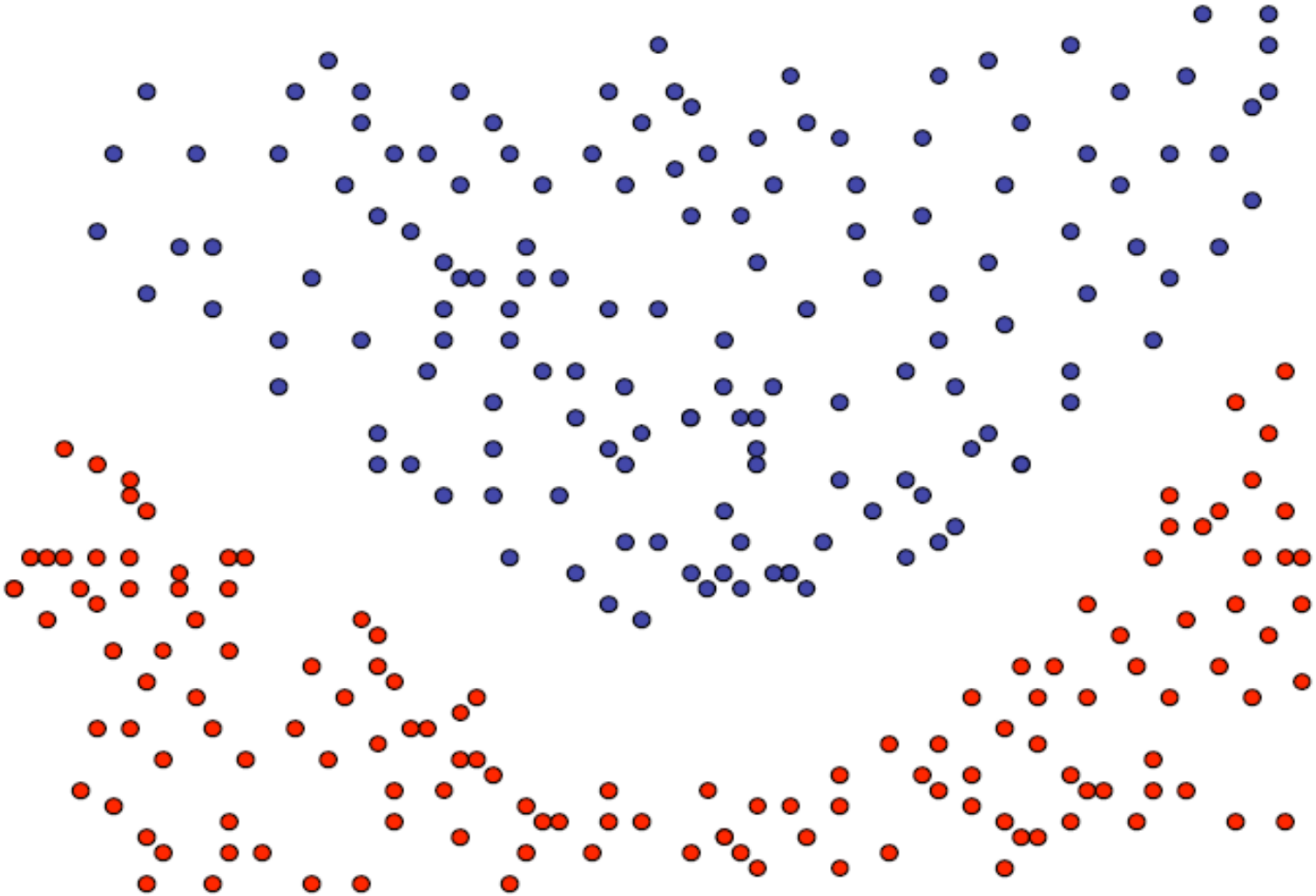


$y_1 = y_2 \Rightarrow \alpha_1 + \alpha_2 = k$

Was wissen wir jetzt?

- Der SMO-Algorithmus ist einer der Optimierungsalgorithmen für das duale Problem.
- Man kann auch z.B. per Evolutionsalgorithmus optimieren (Mierswa 2006).
- Oder mit der *cutting plane* Methode (Kelley 1960) (Joachims 2006)
- ...

Nicht-lineare Daten

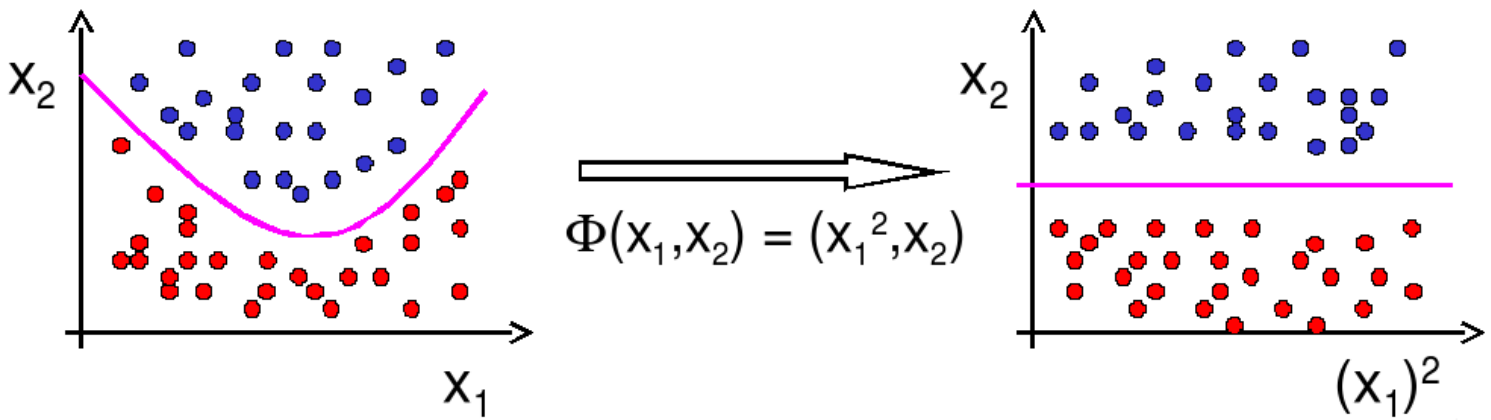


Nicht-lineare Daten

- Neue SVM-Theorie entwickeln? (Neeee!)
- Lineare SVM benutzen?

  If all you've got is a hammer, every problem looks like a nail

- Transformation in lineares Problem!



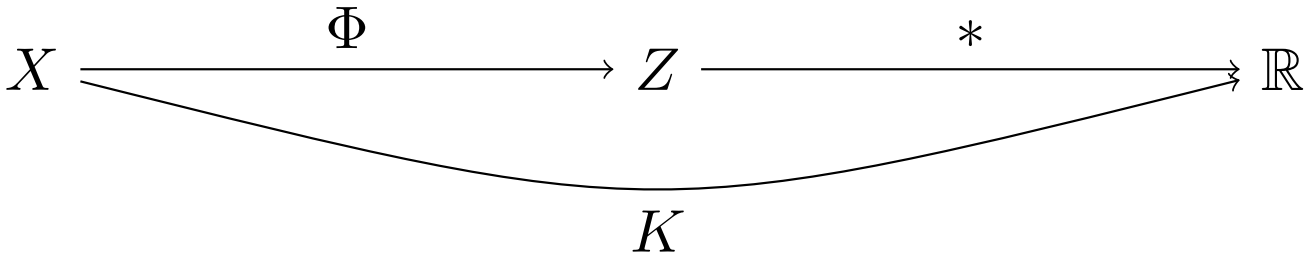
Kernfunktionen

- Erinnerung:

$$L_D(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} y_i y_j \alpha_i \alpha_j \left(\vec{x_i} * \vec{x_j}\right)$$

$$f(\vec{x}) = \sum \alpha_i y_i \left(\vec{x_i} * \vec{x}\right) + \beta_0$$

- SVM hängt von $\vec{x}$ nur über Skalarprodukt $\vec{x} * \vec{x'}$ ab.
- Ersetze Transformation Φ und Skalarprodukt $*$ durch Kernfunktion $K(\vec{x_1}, \vec{x_2}) = \Phi(\vec{x_1}) * \Phi(\vec{x_2})$

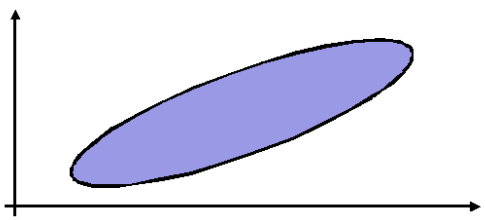


Kernfunktionen II

- Angabe von ϕ nicht nötig, einzige Bedingung: Kernmatrix $(K(\vec{x_i}, \vec{x_j}))_{i,j=1\ldots N}$ muss positiv definit sein.
- Radial-Basisfunktion: $K(\vec{x_i}, \vec{x_j}) = \exp(-\gamma \|\vec{x_i} - \vec{x_j}\|^2)$
- Polynom: $K(\vec{x_i}, \vec{x_j}) = (\vec{x_i} * \vec{x_j})^d$
- Neuronale Netze: $K(\vec{x_i}, \vec{x_j}) = \tanh(\alpha \vec{x_i} * \vec{x_j} + b)$
- Konstruktion von Spezialkernen durch Summen und Produkte von Kernfunktionen, Multiplikation mit positiver Zahl, Weglassen von Attributen

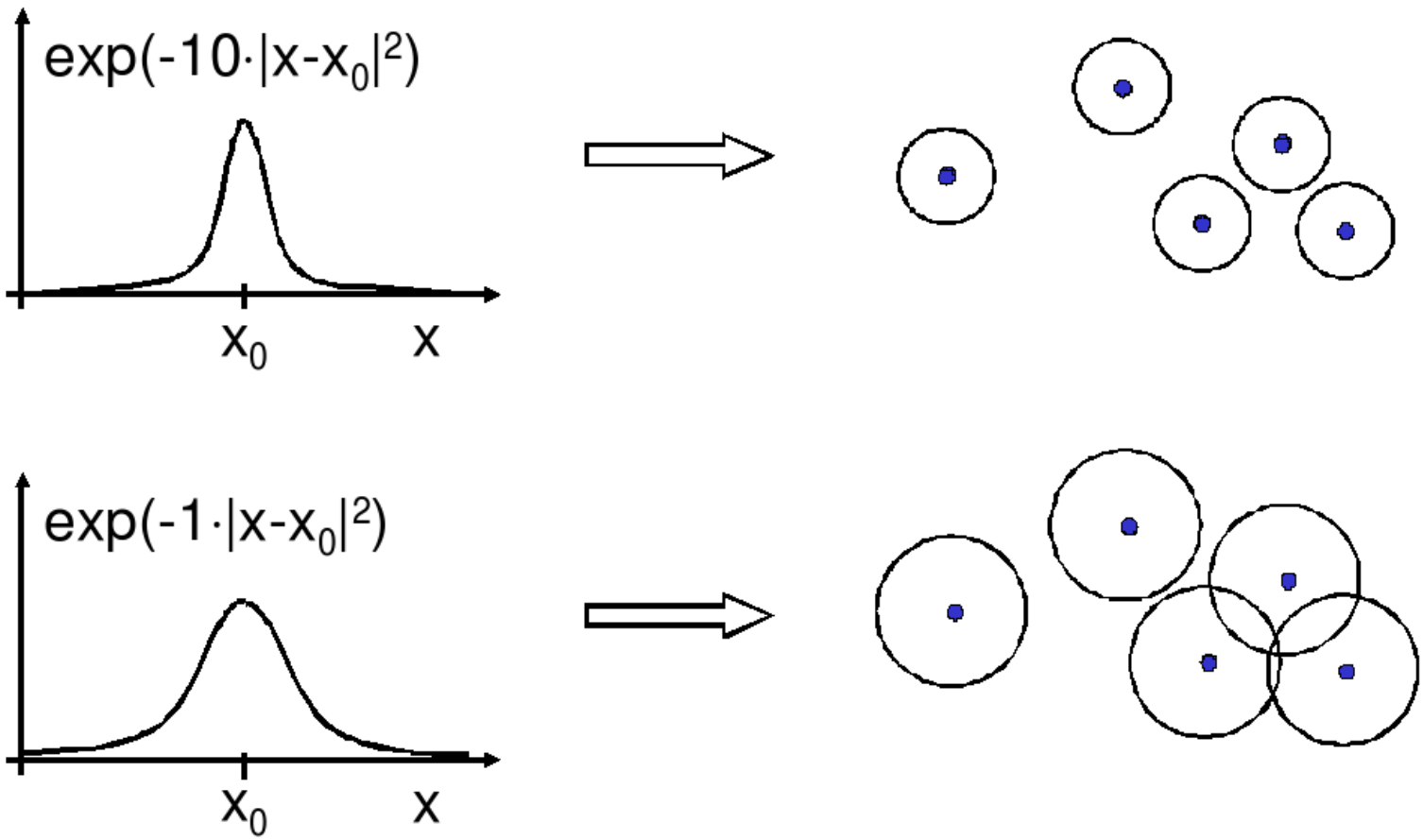
Polynom-Kernfunktionen

- $K_d(\vec{x_i}, \vec{x_j}) = (\vec{x_i} * \vec{x_j})^d$
- Beispiel: $d = 2, \vec{x_i}, \vec{x_j} \in \mathbb{R}^2$.



$$
\begin{aligned}
K_2(\vec{x_i}, \vec{x_j}) &= (\vec{x_i} * \vec{x_j})^2 \\
&= ((x_{i_1}, x_{i_2}) * (x_{j_1}, x_{j_2}))^2 = (x_{i_1} x_{j_1} + x_{i_2} x_{j_2})^2 \\
&= x_{i_1}^2 x_{j_1}^2 + 2 x_{i_1} x_{j_1} x_{i_2} x_{j_2} + x_{i_2}^2 x_{j_2}^2 \\
&= (x_{i_1}^2, \sqrt{2} x_{i_1} x_{i_2}, x_{i_2}^2) * (x_{j_1}^2, \sqrt{2} x_{j_1} x_{j_2}, x_{j_2}^2) \\
&=: \phi(\vec{x_i}) * \phi(\vec{x_j})
\end{aligned}
$$

RBF-Kernfunktion



Kernfunktionen – Basisexpansionen

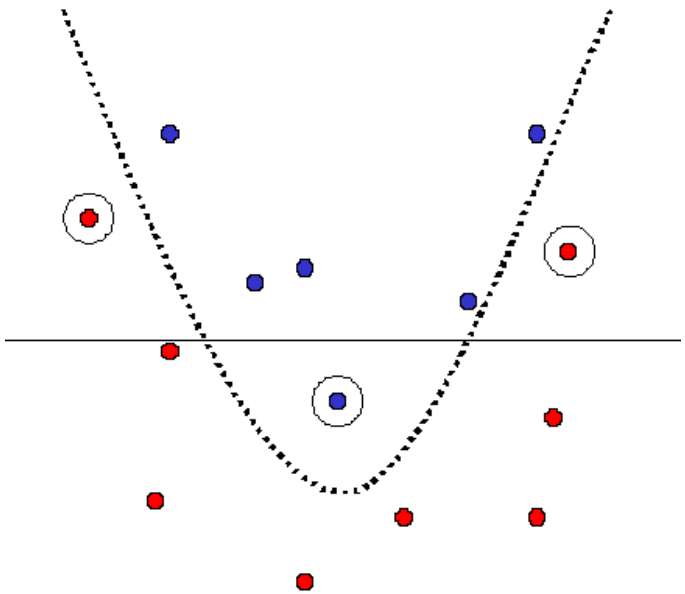
- Die Basisexpansionen waren ein tatsächlicher Schritt der Vorverarbeitung.
- Die Kernfunktionen werden nicht als Vorverarbeitungsschritt durchgeführt.
- Man muss lediglich bei der Berechnung des Skalarprodukts die Kernfunktion berücksichtigen.
- Allerdings kann $\vec{\beta}$ jetzt nicht mehr so einfach interpretiert werden als Bedeutung der Variablen (Merkmale) X_i.

Was ist gutes Lernen?

- Fauler Botaniker:
  “klar ist das ein Baum - ist ja grün.”
  - Übergeneralisierung
  - Wenig Kapazität
  - Bias
- Botaniker mit fotografischem Gedächtnis:
  “nein, dies ist kein Baum, er hat 15 267 Blätter und kein anderer hatte genau so viele.”
  - Overfitting
  - Viel Kapazität
  - Varianz
- Kontrolle der Kapazität!

Bias-Varianz-Problem

- Zu kleiner Hypothesenraum: Zielfunktion nicht gut genug approximierbar (Bias)
- Zu großer Hypothesenraum: Zuviel Einfluss zufälliger Abweichungen (Varianz)
- Lösung: Minimiere obere Schranke des Fehlers:
  $R(\alpha) \leq_\eta R_{emp}(\alpha) + Var(\alpha)$



Risikoschranke nach Vapnik

Strukturelles Risiko

Gegeben eine unbekannte Wahrscheinlichkeitsverteilung $P(\vec{x}, y)$, nach der Daten gezogen werden. Die Abbildungen $\vec{x} \to f(\vec{x}, \vec{\alpha})$ werden dadurch gelernt, dass $\vec{\alpha}$ bestimmt wird. Mit einer Wahrscheinlichkeit $1 - \mu$ ist das Risiko $R(\vec{\alpha})$ nach dem Sehen von N Beispielen beschränkt:

$$R(\vec{\alpha}) \leq R_{emp}(\vec{\alpha}) + \underbrace{\sqrt{\frac{\eta\left(\log\left(\frac{2N}{\eta}\right)+1\right) - \log\left(\frac{\mu}{4}\right)}{N}}}_{\text{VC confidence}}$$

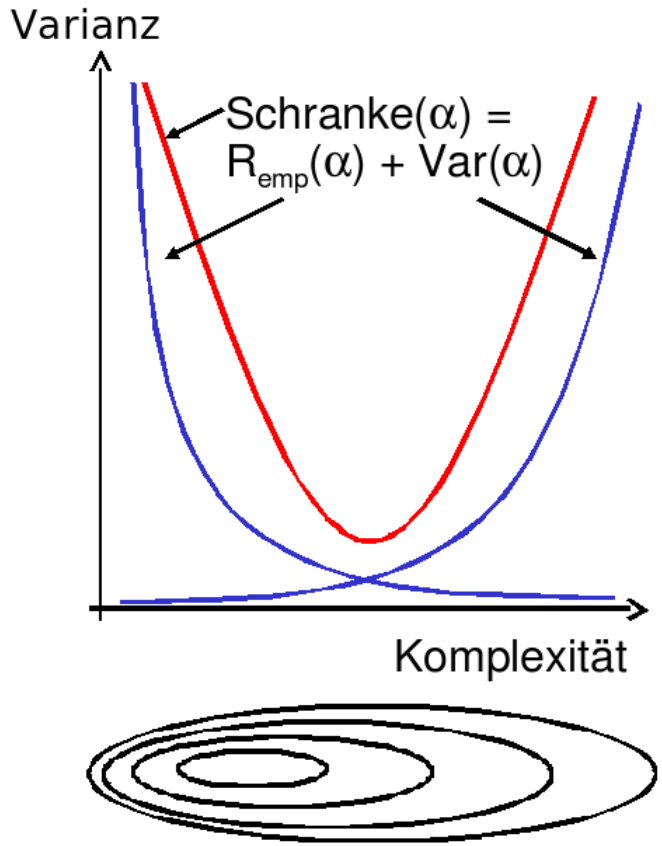
Bevor wir η ergründen (Vapnik-Chervonenkis-Dimension), erst einmal festhalten, was die Bedeutung dieser Schranke ist!

Strukturelle Risikoschranke

- Unabhängig von einer Verteilungsannahme. Alles, was die Schranke braucht, ist, dass Trainings- und Testdaten gemäß der selben Wahrscheinlichkeitsverteilung gezogen werden.
- Das tatsächliche Risiko können wir nicht berechnen.
- Die rechte Seite der Ungleichung können wir berechnen, sobald wir η kennen, die Vapnik-Chervonenkis-Dimension.
- Gegeben eine Menge Hypothesen für $f(\vec{x}, \vec{\alpha})$, wähle immer die mit dem niedrigsten Wert für die rechte Seite der Schranke (R_{emp} oder VC confidence niedrig).

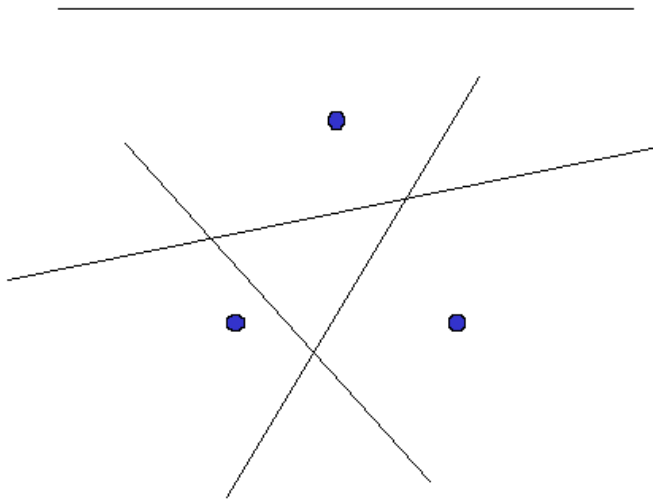
Strukturelle Risikominimierung

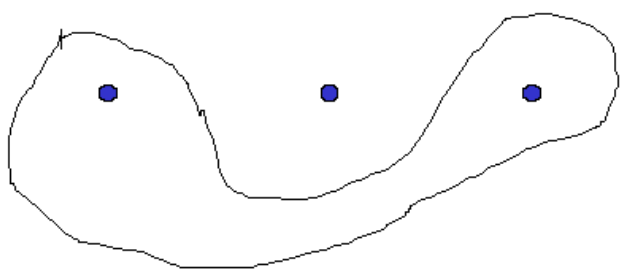
1. Ordne die Hypothesen in Teilmengen gemäß ihrer Komplexität.
2. Wähle in jeder Teilmenge die Hypothese mit dem geringsten empirischen Fehler.
3. Wähle insgesamt die Hypothese mit minimaler Risikoschranke.



Vapnik-Chervonenkis-Dimension

- Definition: Eine Menge H von Hypothesen zerschmettert eine Menge E von Beispielen, wenn jede Teilmenge von E durch ein $h \in H$ abgetrennt werden kann.
- Definition: Die VC-Dimension einer Menge von Hypothesen H ist die maximale Anzahl von Beispielen E, die von H zerschmettert wird.
- Eine Menge von 3 Punkten kann von geraden Linien zerschmettert werden, keine Menge von 4 Punkten kann von geraden Linien zerschmettert werden.





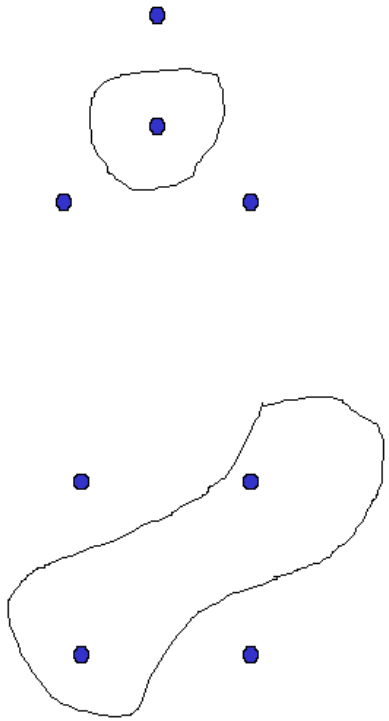
ACHTUNG

- Für eine Klasse von Lernaufgaben gibt es mindestens eine Menge E, die zerschmettert werden kann - NICHT jede Menge E kann zerschmettert werden!
- Zum Beweis der VC Dimension n muss man also zeigen:
  - Es gibt eine Menge E aus n Punkten, die von H zerschmettert werden kann. $VCdim(H) \geq n$
  - Es kann keine Menge E' aus $n+1$ Punkten geben, die von H zerschmettert werden könnte. $VCdim(H) \leq n$

VC-Dimension von Hyperebenen

Satz: Die VC-Dimension der Hyperebenen im R^p ist $p+1$.
Beweis:

- $VCdim(R^p) \geq p+1$: Wähle $\vec{x_0} = 0$ und $\vec{x_i} = (0, \ldots, 0, 1, 0, \ldots 0)$. Für eine beliebige Teilmenge A von $(\vec{x_0}, \ldots, \vec{x_n})$ setze $y_i = 1$, falls $\vec{x_i} \in A$, sonst $y_i = -1$.
  Definiere $\vec{\beta} = \sum y_k \vec{x_k}$ und $\beta_0 = \frac{y_0}{2}$.
  Dann gilt $\vec{\beta}\vec{x_0} + \beta_0 = \frac{y_0}{2}$ und $\vec{\beta}\vec{x_i} + \beta_0 = y_i + \frac{y_0}{2}$.
  Also: $\vec{\beta}\vec{x} + \beta_0$ trennt A.
- $VCdim(R^p) \leq p+1$: Zurückführen auf die beiden Fälle rechts.



VCdim misst Kapazität

- Eine Funktion mit nur 1 Parameter kann unendliche $VCdim$ haben: H kann Mengen von n Punkten zerschmettern, egal wie groß n ist.
- H kann unendliche $VCdim$ haben und trotzdem kann ich eine kleine Zahl von Punkten finden, die H nicht zerschmettern kann.
- $VCdim$ ist also nicht groß, wenn die Anzahl der Parameter bei der Klasse von Funktionen H groß ist.

VC-Dimension der SVM

- Gegeben seien Beispiele $\vec{x_1}, \ldots, \vec{x_N} \in \mathcal{R}^p$ mit $\|\vec{x_i}\| < D$ für alle i. Für die VC-Dimension der durch den Vektor $\vec{\beta}$ gegebenen optimalen Hyperebene H gilt:

$$VCdim(H) \leq \min\left\{D^2\|\vec{\beta}\|^2, p\right\} + 1$$

- Die Komplexität einer SVM ist auch durch die Struktur der Lösung begrenzt!
- Die SVM minimiert nicht nur das empirische Risiko, sondern auch das strukturelle.

Zusicherungen

- Strukturelle Risikominimierung garantiert, dass die einfachste Hypothese gewählt wird, die noch an die Daten anpassbar ist.
- Strukturelle Risikominimierung kontrolliert die Kapazität des Lernens (weder fauler noch fotografischer Botaniker).
- Die Strukturen von Klassen von Funktionen werden durch die $VCdim$ ausgedrückt. Große $VCdim \rightarrow$ große VC-confidence.
- Wir haben nun also ein Verfahren, das ohne zusätzlichen Aufwand die Komplexität regularisiert, wie wir es bei der Modellselektion für lineare und lokale Modelle mal wollten.

Performanzschätzer

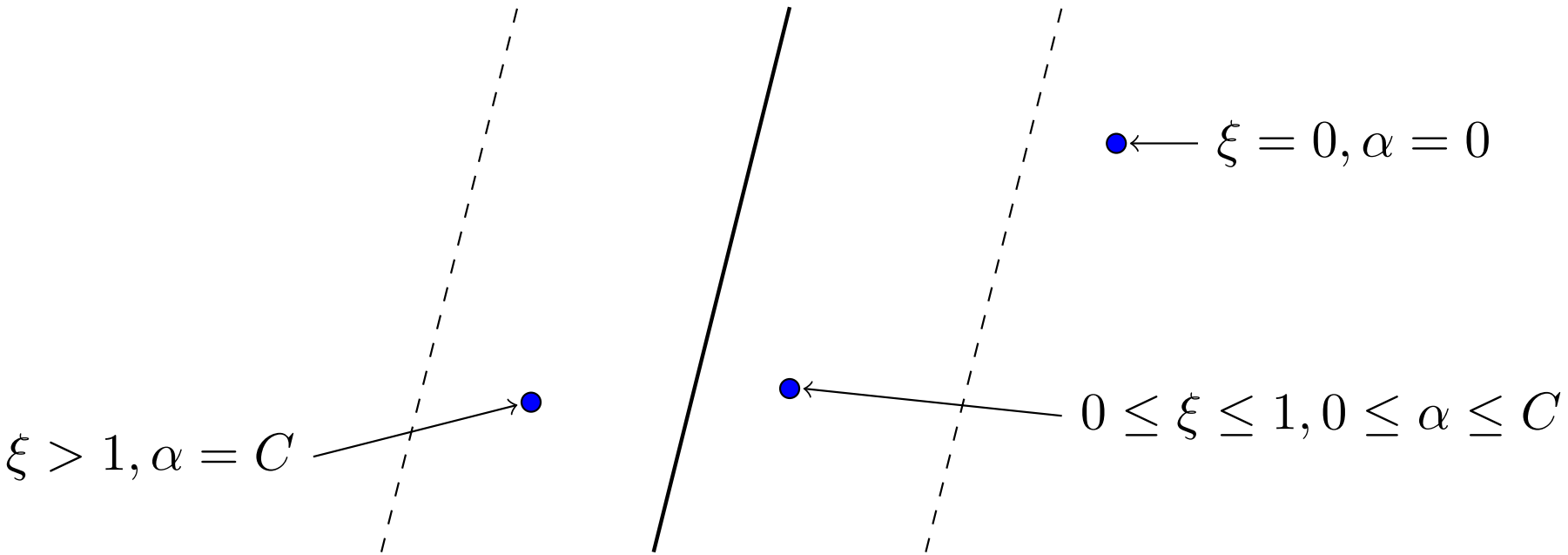
- Welches erwartete Risiko $R(\alpha)$ erreicht SVM?
- $R(\vec{\alpha})$ selbst nicht berechenbar
- Trainingsfehler (zu optimistisch - Overfitting)
- Obere Schranke mittels VC-Dimension (zu locker)
- Kreuzvalidierung / Leave-One-Out-Schätzer (ineffizient)

Performanzschätzer II

- Satz: Der Leave-One-Out-Fehler einer SVM ist beschränkt durch $R_{l1o} \leq \frac{|SV|}{N}$
- Beweis (Skizze):
  - Falsch klassifizierte Beispiele werden Stützvektoren (SV).
  - Also: Nicht-Stützvektoren werden korrekt klassifiziert. Weglassen eines Nicht-Stützvektors ändert die Hyperebene nicht, daher wird es auch beim $l1o$-Test richtig klassifiziert.
  - Nur der Anteil der Stützvektoren an den Beispielen macht den Fehler aus.

Performanzschätzer III

- Satz: Der Leave-One-Out-Fehler einer SVM ist beschränkt durch $R_{l1o} \leq \frac{|\{i:(2\alpha_i D^2+\xi_i)\geq 1\}|}{N}$ (D = Radius des Umkreises um die Beispiele im transformierten Raum).
- Beweis: Betrachte folgende drei Fälle:



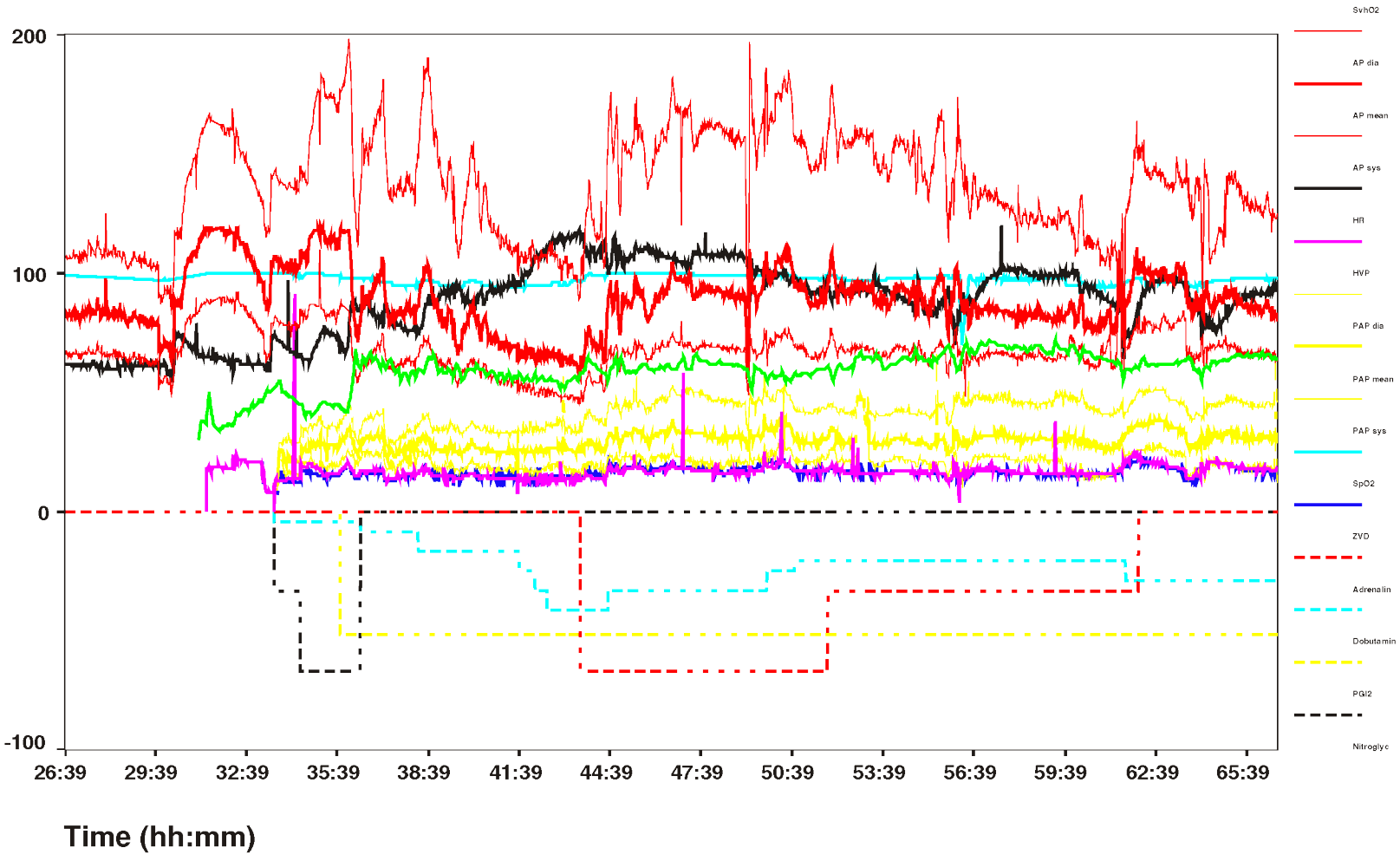
Was wissen wir jetzt?

- Kernfunktionen - eine Transformation, die man nicht erst durchführen und dann mit ihr rechnen muss, sondern bei der nur das Skalarprodukt gerechnet wird.
- Idee der strukturellen Risikominimierung (SRM):
  - obere Schranke für das Risiko
  - Schrittweise Steigerung der Komplexität
- Formalisierung der Komplexität: VC-Dimension
- SRM als Prinzip der SVM
- Garantie für die Korrektheit der Lernstrategie

Fallstudie Intensivmedizin

- Städtische Kliniken Dortmund, Intensivmedizin 16 Betten, Prof. Dr. Michael Imhoff (Ruhr-Universität Bochum)
- Hämodynamisches Monitoring, minütliche Messungen
  - Diastolischer, systolischer, mittlerer arterieller Druck
  - Diastolischer, systolischer, mittlerer pulmonarer Druck
  - Herzrate
  - Zentralvenöser Druck
- Therapie, Medikamente:
  - Dobutamine, adrenaline, glycerol trinitrate, noradrenaline, dopamine, nifedipine

Patient G.C., male, 60 years old - Hemihepatektomie right



Wann wird Medikament gegeben?

- Mehrklassenproblem in mehrere 2-Klassen-Probleme umwandeln:
  - Für jedes Medikament entscheide, ob es gegeben werden soll oder nicht.
  - Positive Beispiele: alle Minuten, in denen das Medikament gegeben wurde
  - Negative Beispiele: alle Minuten, in denen das Medikament nicht gegeben wurde

  Parameter: Kosten falscher Positiver = Kosten falscher Negativer

  Ergebnis: Gewichte der Vitalwerte $\vec{\beta}$, so dass positive und negative Beispiele maximal getrennt werden (SVM).

Beispiel: Intensivmedizin

$$f(\vec{x}) = \left[\begin{pmatrix} 0.014 \\ 0.019 \\ -0.001 \\ -0.015 \\ -0.016 \\ 0.026 \\ 0.134 \\ -0.177 \\ \vdots \end{pmatrix} \begin{pmatrix} artsys = 174.00 \\ artdia = 86.00 \\ artmn = 121.00 \\ cvp = 8.00 \\ hr = 79.00 \\ papsys = 26.00 \\ papdia = 13.00 \\ papmn = 15.00 \\ \vdots \end{pmatrix} - 4.368 \right]$$

Wie wird ein Medikament dosiert ?

- Mehrklassenproblem in mehrere 2 Klassenprobleme umwandeln: für jedes Medikament und jede Richtung (increase, decrease, equal), 2 Mengen von Patienten-daten:
  - Positive Beispiele: alle Minuten, in denen die Dosierung in der betreffenden Richtung geändert wurde
  - Negative Beispiele: alle Minuten, in denen die Dosierung nicht in der betreffenden Richtung geändert wurde.

Steigern von Dobutamine

Vektor $\vec{\beta}$ für p Attribute

$ARTEREN$:	-0.05108108119
$SUPRA$:	0.008928075386579 73
$DOBUTREX$:	-0.100650806786886
$WEIGHT$:	-0.039353180104626 5
AGE :	-0.003788286810714 17
$ARTSYS$:	-0.323407537252192
$ARTDIA$:	-0.039456533301949 3
$ARTMN$:	-0.180425080906375
HR :	-0.100104052643 06
$PAPSYS$:	-0.025264118853173 1
$PAPDIA$:	0.045484333711276 5
$PAPMN$:	0.004295049637365 22
$PULS$:	-0.031350123639988 1

Anwendung des Gelernten für Dobutamin

- Patientwerte
  pat46, artmn 95, min. 2231
  ...
  pat46, artmn 90, min. 2619

- Gelernte Gewichte β_i:
  $artmn - 0,18$
  ...

$$svm\_calc = \sum_{i=1}^{p} \beta_i x_i$$

$$decision = sign(svm\_calc + \beta_0)$$

- $svm\_calc(pat46, dobutrex, up, min.2231, 39)$
- $svm\_calc(pat46, dobutrex, up, min.2619, 25)$
- $\beta_0 = -26$, i.e. increase in minute 2231, not increase in minute 2619.

Steigern von Glyceroltrinitrat (nitro)

$$f(x) = \left[\begin{pmatrix} 0.014 \\ 0.019 \\ -0.001 \\ -0.015 \\ -0.016 \\ 0.026 \\ 0.134 \\ -0.177 \\ -9.543 \\ -1.047 \\ -0.185 \\ 0.542 \\ -0.017 \\ 2.391 \\ 0.033 \\ 0.334 \\ 0.784 \\ 0.015 \end{pmatrix} \begin{pmatrix} artsys = 174.00 \\ artdia = 86.00 \\ artmn = 121.00 \\ cvp = 8.00 \\ hr = 79.00 \\ papsys = 26.00 \\ papdia = 13.00 \\ papmn = 15.00 \\ nifedipine = 0 \\ noradrenaline = 0 \\ dobutamie = 0 \\ dopamie = 0 \\ glyceroltrinitrate = 0 \\ adrenaline = 0 \\ age = 77.91 \\ emergency = 0 \\ bsa = 1.79 \\ broca = 1.02 \end{pmatrix} - 4.368 \right]$$

- Jedes Medikament hat einen Dosierungsschritt. Für Glyceroltrinitrat ist es 1, für $Suprarenin$ (adrenalin) 0,01. Die Dosis wird um einen Schritt erhöht oder gesenkt.
- Vorhersage: $pred\_interv(pat49, min.32, nitro, 1, 0)$

Evaluierung

- Blind test über 95 noch nicht gesehener Patientendaten.
  - Experte stimmte überein mit tatsächlichen Medikamentengaben in 52 Fällen
  - SVM Ergebnis stimmte überein mit tatsächlichen Medikamentengaben in 58 Fällen

Dobutamine	*Actual up*	*Actual equal*	*Actual down*
Predicted up	**10 (9)**	12 (8)	0 (0)
Predicted equal	7 (9)	**35 (31)**	9 (9)
Predicted down	2 (1)	7 (15)	**13 (12)**

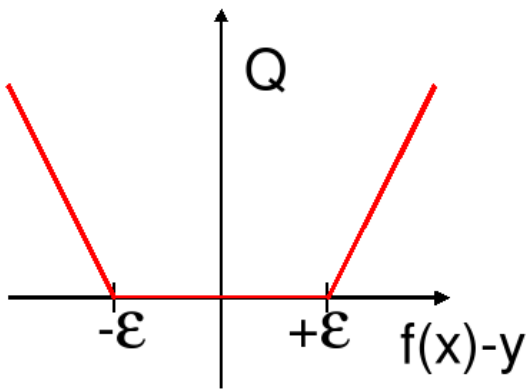
SVMs für Regession

Durch Einführung einer anderen *Loss-Funktion* läßt sich die SVM zur Regression nutzen. Sei $\varepsilon \in \mathbb{R}_{>0}$ und

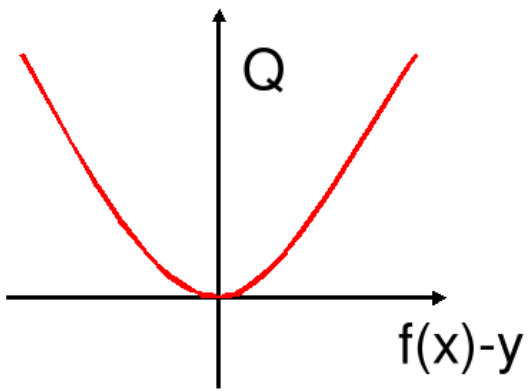
$$L_k(y, f(\vec{x}, \alpha)) = \begin{cases} 0 & \text{, falls } y - f(\vec{x}, \alpha) \leq \varepsilon \\ (y - f(\vec{x}, \alpha) - \varepsilon)^k & \text{, sonst} \end{cases}$$

Die *Loss-Funktion* L_1 gibt den Abstand der Funktion f von den Trainingsdaten an, alternativ quadratische Loss-Funktion L_2:

lineare Verlustfunktion



quadratische Verlustfunktion



SVMs für Regression

Dadurch ergibt sich das Optimierungsproblem:

Regressions-SVM

Minimiere

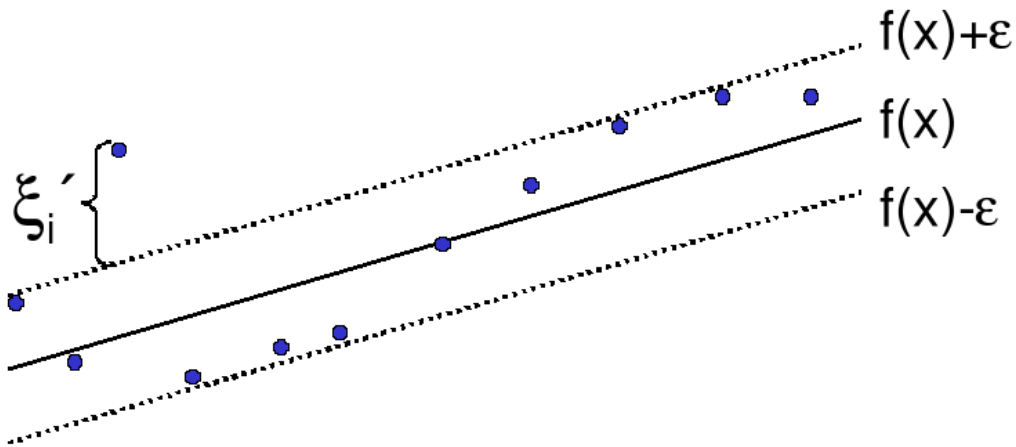
$$\|\vec{\beta}\|^2 + C\left(\sum_{i=1}^{N}\xi_i + \sum_{i=1}^{N}\xi_i'\right)$$

unter den Nebenbedingungen

$$\begin{aligned} f(\vec{x_i}) &= \langle\vec{\beta}, \vec{x_i}\rangle + \beta_0 \leq y_i + \epsilon + \xi_i' \\ f(\vec{x_i}) &= \langle\vec{\beta}, \vec{x_i}\rangle + \beta_0 \geq y_i - \epsilon - \xi_i \end{aligned}$$

SVMs für Regression

Die ξ_i bzw. ξ_i' geben für jedes Beispiel Schranken an, innerhalb derer der vorhergesagte Funktionswert für jedes Beispiel liegen soll:



Bei der Lösung des Optimierungsproblems mit Lagrange führt dies zu *zwei* α-Werten je Beispiel!

SVMs für Regression

Das duale Problem enthält für jedes $\vec{x}_i$ je zwei α-Werte α_i und α_i', je einen für ξ_i und ξ_i', d.h.

Duales Problem für die Regressions-SVM

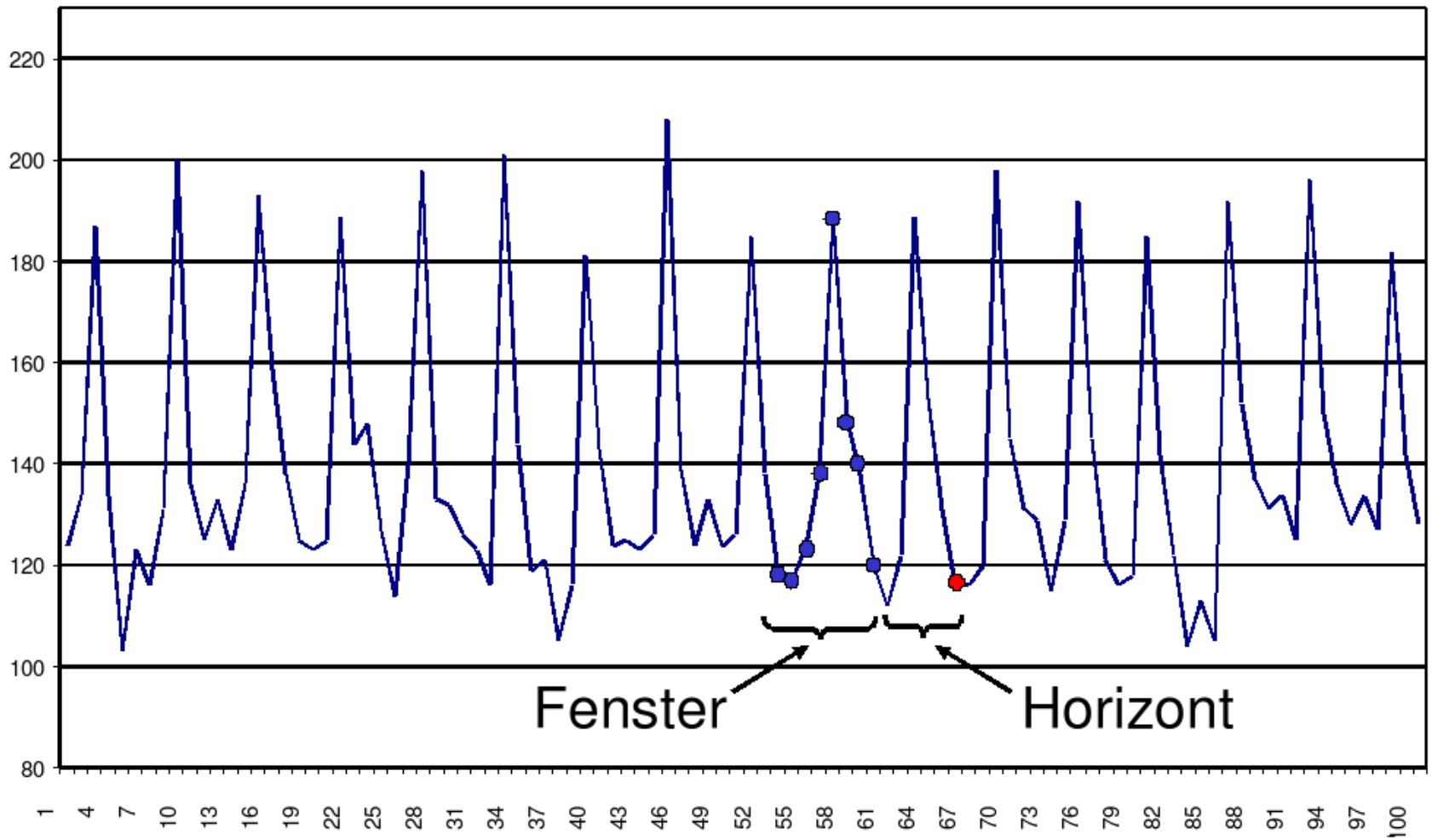
Maximiere

$$L_D(\vec{\alpha}, \vec{\alpha}') = \sum_{i=1}^{N} y_i \left(\alpha_i' - \alpha_i\right) - \epsilon \sum_{i=1}^{N} y_i \left(\alpha_i' - \alpha_i\right) - \frac{1}{2} \sum_{i,j=1}^{n} y_i \left(\alpha_i' - \alpha_i\right) \left(\alpha_j' - \alpha_j\right) K(\vec{x_i}, \vec{x_j})$$

unter den Nebenbedingungen

$$0 \leq \alpha_i, \alpha_i' \leq C \,\forall\, i = 1, \ldots, N \quad \text{und} \quad \sum_{i=1}^{N} \alpha_i' = \sum_{i=1}^{N} \alpha_i$$

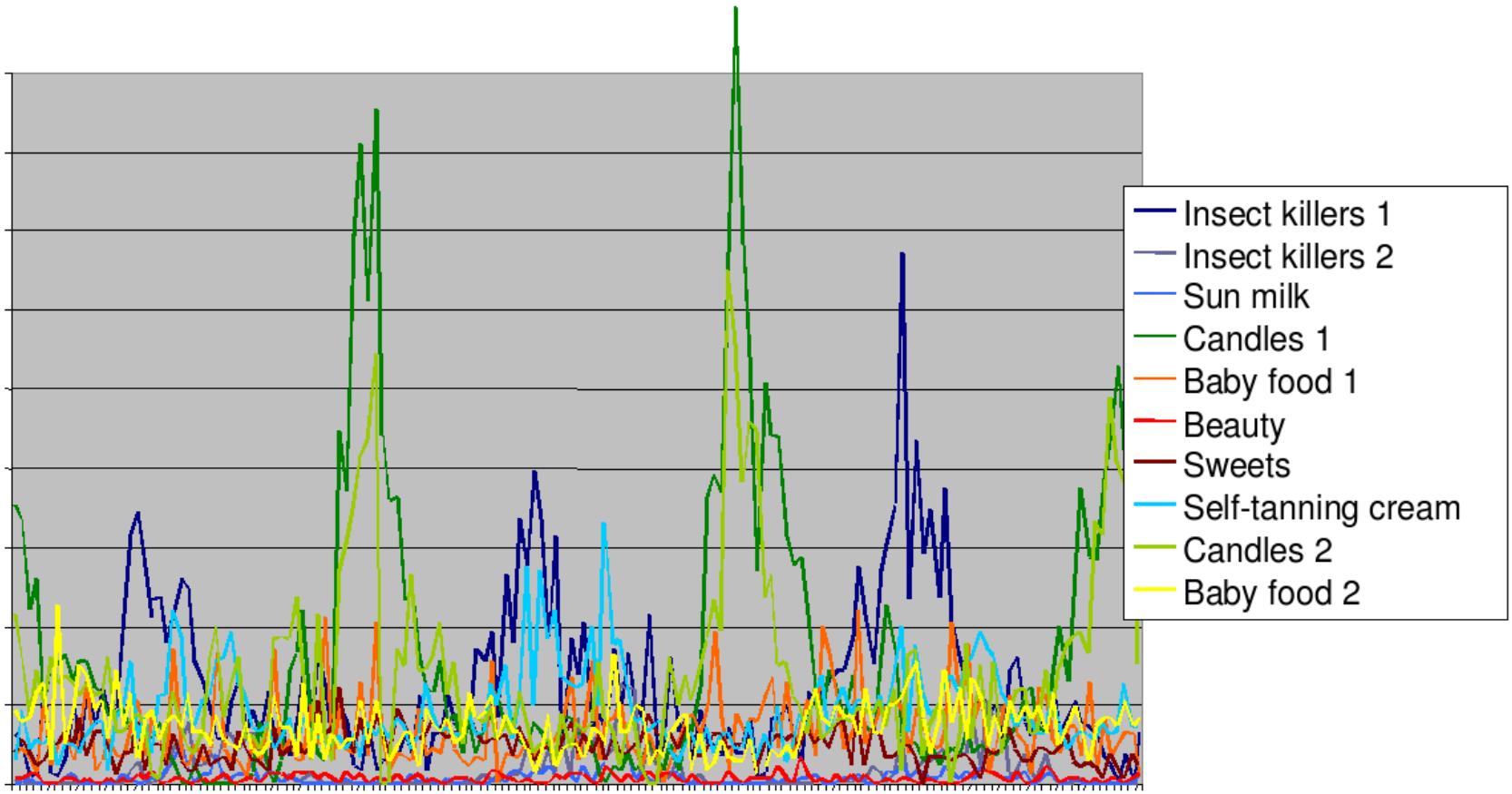
Beispiel: Prognose von Zeitreihen



Prognose von Zeitreihen

- Trend
- Zyklen
- Besondere Ereignisse (Weihnachten, Werbung, ...)
- Wieviel vergangene Beobachtungen?
- Ausreißer

Abverkauf Drogerieartikel



Vorhersage Abverkauf

Gegeben Verkaufsdaten von 50 Artikeln in 20 Läden über 104 Wochen

Vorhersage Verkäufe eines Artikels, so dass

- Die Vorhersage niemals den Verkauf unterschätzt,
- Die Vorhersage überschätzt weniger als eine Faustregel.

Beobachtung 90% der Artikel werden weniger als 10 mal pro Woche verkauft.

Anforderung Vorhersagehorizont von mehr als 4 Wochen.

Verkaufsdaten – multivariate Zeitreihen

Shop	Week	Item1	...	Item50
Dm1	1	4	...	12
Dm1	...	...	...	...
Dm1	104	9	...	16
Dm2	1	3	...	19
...	...	...	...	...
Dm20	104	12	...	16

Vorverarbeitung: multivariat nach univariat

Quasi-SQL:
For all shops for all items: Create view Univariate as
Select shop, week, $item_i$
Where shop="dm_j"
From Source;

- Multiples Lernen für alle univariaten Zeitreihen

Shop_Item	Week	Sale	Week	Sale
Dm1_Item1	1	4...	104	9
...				
Dm1_Item50	1	12...	104	16
...				
Dm20_Item50	1	14...	104	16

Vorverarbeitung II

- Problem: eine Zeitreihe ist nur 1 Beispiel!
- Das ist für das Lernen zu wenig.
- Lösung: Viele Vektoren aus einer Reihe gewinnen durch Fenster der Breite (Anzahl Zeitpunkte) w, bewege Fenster um m Zeitpunkte weiter.

Shop_Item_Window	Week	Sale	Week	Sale
Dm1_Item1_1	1	4...	5	7
Dm1_Item1_2	2	4...	6	8
...	...	...	...	...
Dm1_Item1_100	100	6...	104	9
...	...	...	...	...
Dm20_Item50_100	100	12...	104	16

SVM im Regressionfall

- Multiples Lernen:
  für jeden Laden und jeden Artikel, wende die SVM an. Die gelernte Regressionsfunktion wird zur Vorhersage genutzt.
- Asymmetrische Verlustfunktion :
  - Unterschätzung wird mit 20 multipliziert, d.h. 3 Verkäufe zu wenig vorhergesagt – 60 Verlust
  - Überschätzung zählt unverändert, d.h. 3 Verkäufe zu viel vorhergesagt – 3 Verlust

  (Diplomarbeit Stefan Rüping 1999)

Vergleich mit Exponential Smoothing

Horizont	SVM	exp. smoothing
1	56.764	52.40
2	57.044	59.04
3	57.855	65.62
4	58.670	71.21
8	60.286	88.44
13	59.475	102.24

Verlust, nicht normiert auf $[0, 1]$!

Was wissen wir jetzt?

- Anwendung der SVM für die Medikamentenverordnung
- Idee der Regressions-SVM
- Anwendung der SVM für die Verkaufsvorhersage
  - Umwandlung multivariater Zeitreihen in mehrere univariate
  - Gewinnung vieler Vektoren durch gleitende Fenster
  - Asymmetrische Verlustfunktion