Wissensentdeckung Vorlesung

SVM – SMO, Ball SVM, Core SVM

Katharina Morik, Uwe Ligges

LS 8 Informatik
Computergestützte Statistik
Technische Universität Dortmund

4.6.2012

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 erfÂÿllen, 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 fÂÿr 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 StÂÿtzvektor.

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

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

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! Zu aufwändig.
  - 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:

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

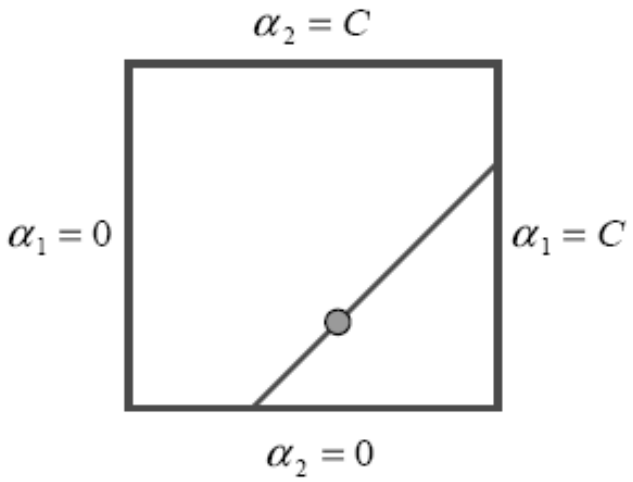
- Zulässige α_1, α_2 liegen im Bereich $[0, C] \times [0, C]$ auf der Geraden
  $W = \alpha_1 y_1 + \alpha_2 y_2$ äquivalent $\alpha_1 + s\alpha_2$ mit $s = \frac{y_2}{y_1}$
- Wir optimieren α_2
- Aus dem optimalen $\hat{\alpha}_2$ können wir das optimale $\hat{\alpha}_1$ herleiten:

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

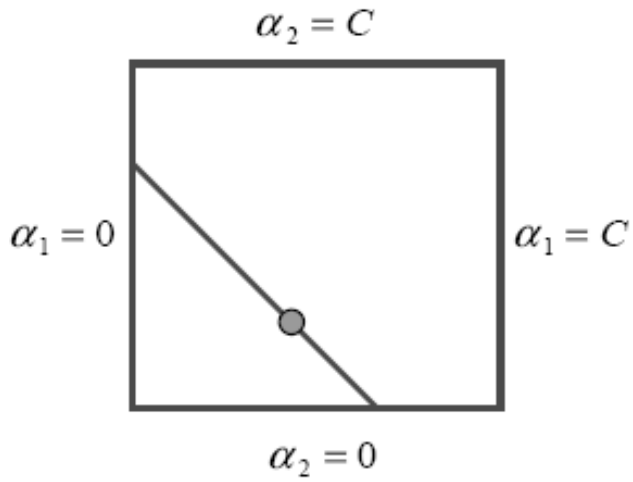
- Dann kommen die nächsten zwei α_i dran...

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...



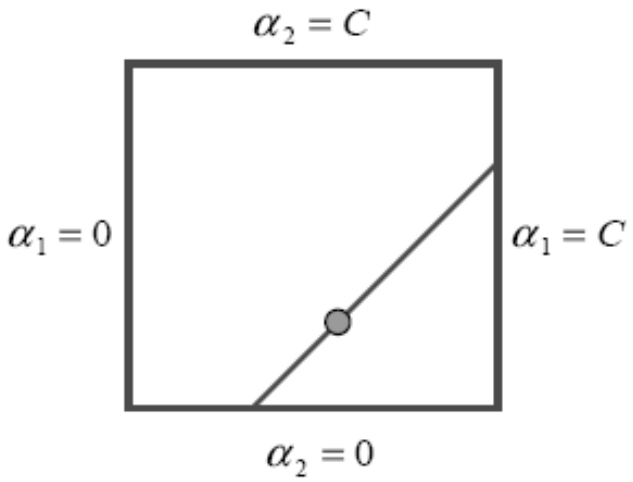
$y_1 \neq y_2 \Rightarrow \alpha_1 - \alpha_2 = k$



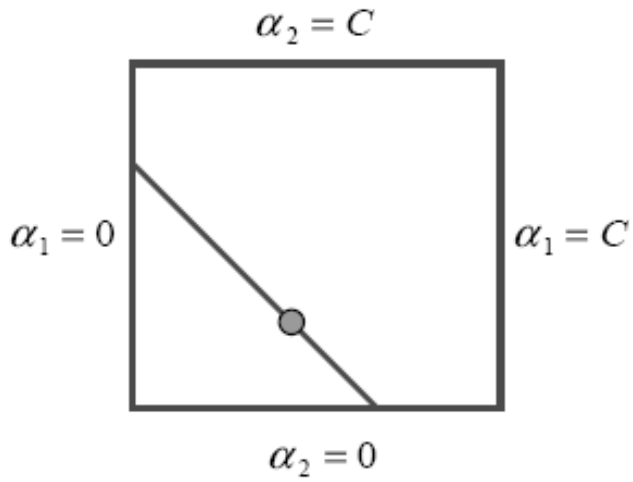
$y_1 = y_2 \Rightarrow \alpha_1 + \alpha_2 = k$

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...



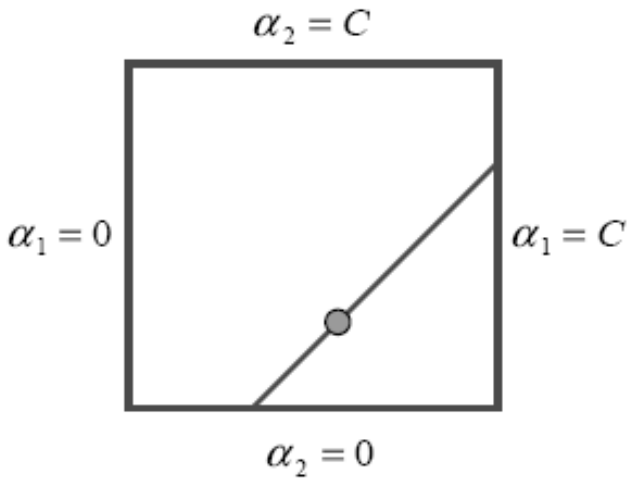
$y_1 \neq y_2 \Rightarrow \alpha_1 - \alpha_2 = k$



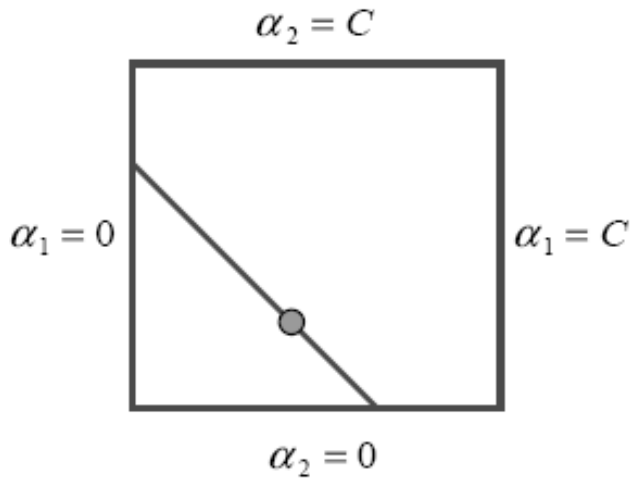
$y_1 = y_2 \Rightarrow \alpha_1 + \alpha_2 = k$

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...



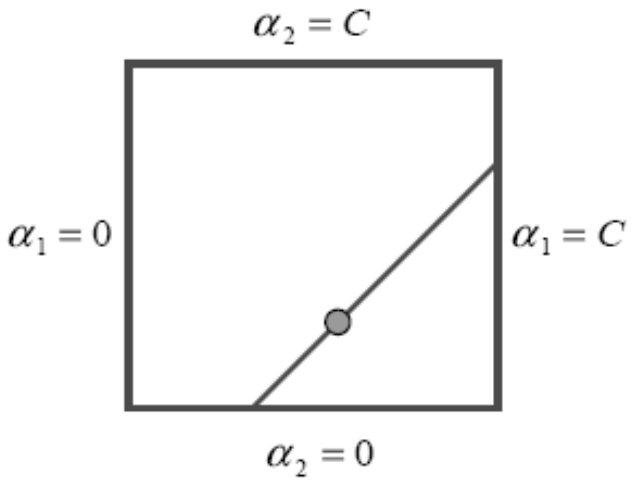
$y_1 \neq y_2 \Rightarrow \alpha_1 - \alpha_2 = k$



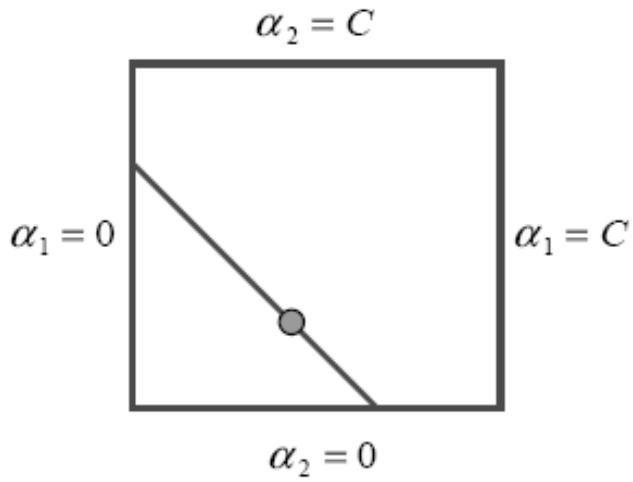
$y_1 = y_2 \Rightarrow \alpha_1 + \alpha_2 = k$

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...



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



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

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}
  - $\alpha_2^{new} = \alpha_2^{old} + \frac{y_2((f(\vec{x_1}) - y_1) - (f(\vec{x_2}) - y_2))}{\eta}$
  - $= \alpha_2^{old} + \frac{y_2(E_1 - E_2)}{\eta}$
  - $\eta = x_1^T x_1 + x_2^T x_2 - 2x_1^T x_2$

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}
  - $\alpha_2^{new} = \alpha_2^{old} + \frac{y_2((f(\vec{x}_1) - y_1) - (f(\vec{x}_2) - y_2))}{\eta}$
  - $= \alpha_2^{old} + \frac{y_2(E_1 - E_2)}{\eta}$
  - $\eta = x_1^T x_1 + x_2^T x_2 - 2x_1^T x_2$

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}
  - $\alpha_2^{new} = \alpha_2^{old} + \frac{y_2((f(\vec{x}_1) - y_1) - (f(\vec{x}_2) - y_2))}{\eta}$
  - $= \alpha_2^{old} + \frac{y_2(E_1 - E_2)}{\eta}$
  - $\eta = x_1^T x_1 + x_2^T x_2 - 2x_1^T x_2$

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}
  - $\alpha_2^{new} = \alpha_2^{old} + \frac{y_2((f(\vec{x_1}) - y_1) - (f(\vec{x_2}) - y_2))}{\eta}$
  - $= \alpha_2^{old} + \frac{y_2(E_1 - E_2)}{\eta}$
  - $\eta = x_1^T x_1 + x_2^T x_2 - 2x_1^T x_2$

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}
  - $\alpha_2^{new} = \alpha_2^{old} + \frac{y_2((f(\vec{x_1}) - y_1) - (f(\vec{x_2}) - y_2))}{\eta}$
  - $= \alpha_2^{old} + \frac{y_2(E_1 - E_2)}{\eta}$
  - $\eta = x_1^T x_1 + x_2^T x_2 - 2x_1^T x_2$

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}
  - $\alpha_2^{new} = \alpha_2^{old} + \frac{y_2((f(\vec{x_1}) - y_1) - (f(\vec{x_2}) - y_2))}{\eta}$
  - $= \alpha_2^{old} + \frac{y_2(E_1 - E_2)}{\eta}$
  - $\eta = x_1^T x_1 + x_2^T x_2 - 2x_1^T x_2$

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...

SMO und mehr

- SMO hat Laufzeit $O(N^{2,2})$.
- Der SMO nimmt jeweils ein Paar α_1, α_2.
- Man kann aber auch gleich eine Menge von α_i für die Bearbeitung auswählen.
  Eigentlich soll man die gesamte Kernmatrix für die Optimierung verwenden, man kann aber eine Auswahl von Beispielen, die die KKT-Bedingungen verletzen, in einen working set aufnehmen.
- Die Kalkulationen in diesem working set werden gespeichert (caching).
- Beispiele, die nicht mehr die KKT-Bedingunge verletzen, werden aus dem working set gelöscht (shrinking).
- Mit caching und shrinking und Heuristiken hat Thorsten Joachims die erste effiziente SVM implementiert, SVM_light.

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: $g_i = \sum \alpha_k y_k y_i \langle x_k, x_i \rangle - 1$
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: Caching von $\langle x_i, x_j \rangle$
- Wir arbeiten nicht über der gesamten $N \times N$ Kernmatrix, sondern nur über dem working set.

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 die lineare SVM mit der *cutting plane* Methode bearbeiten (Kelley 1960) (Joachims 2006)
- ...

SVM mal anders

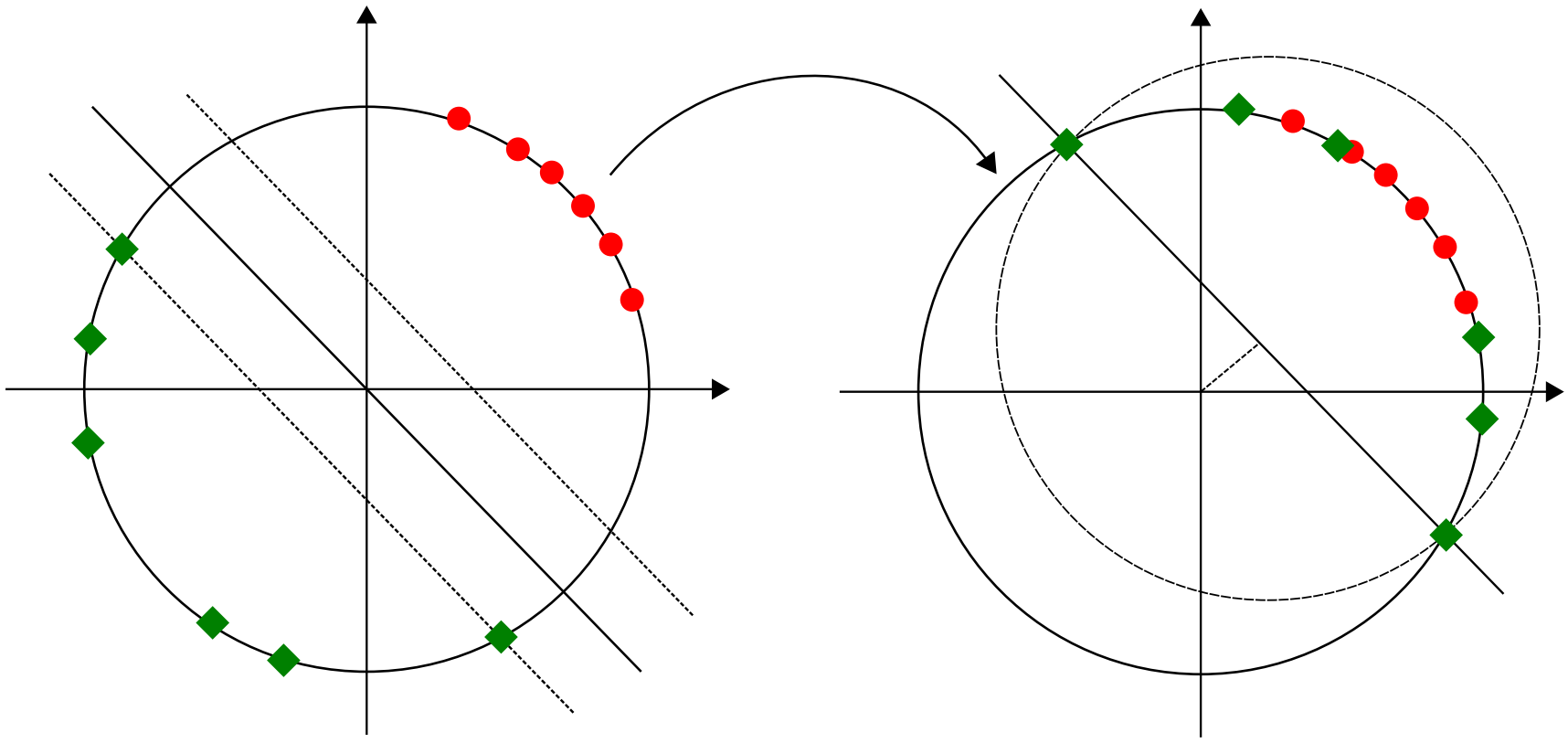
Semi-Überwachtes Lernen : Nicht alle Beispiele haben ein label y_i, aber einige. Schon die Lage, wo es Beobachtungen gibt (und wo nicht) ist informativ.

Transduktion : Nicht alle Beispiele haben ein label y_i, aber alle $\vec{x_i}$ sind gegeben. Die Klassifikationsaufgabe ist vereinfacht: sie muss nur für die gegebenen Beobachtungen gut sein. Nützlich z.B. bei großen Bestandssammlungen.

1-Klassen-SVM : Hyperebene, die alle Beispiele mit maximalem Abstand vom Ursprung trennt. Gibt einen Eindruck von der Verteilung der Daten.

SVDD zu Daten-Beschreibung : Beschreibung der Daten durch eine sie umfassende Kugel. Nützlich zur Ausreißererkennung.

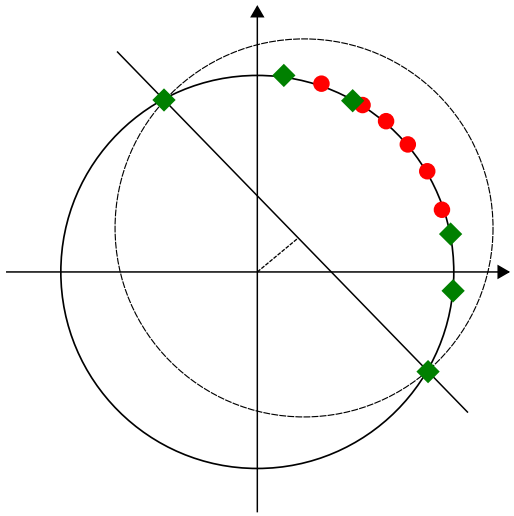
Anschaulich: 2-Klassen-SVM und 1-Klassen-SVM



Hyperebene durch den Ursprung – projizieren auf eine Seite – max. Abstand vom Ursprung.

1-Klassen-SVM und SVDD – anschaulich

Wenn die Kern-Funktion $K(x,x) = \kappa, \forall x \in S$ konstant ist, werden die Daten aus S auf einen Ball K_S abgebildet. Dann sind die SVDD und die 1-Klassen-SVM gleich.



So beim RBF-Kern, $K(\vec{x},\vec{x}) = \exp(-\gamma\|\vec{x}-\vec{x}\|^2) = 1, \forall x \in S$.
Bei einem Poly-Kern ist dies nicht der Fall. Da muss erst normalisiert werden: $K(x,y) = \frac{K(x,y)}{\sqrt{K(x,x)}\sqrt{K(y,y)}}$

Beispielrechnung

$$
\begin{aligned}
K_2(\vec{x_i}, \vec{x_i}) &= \langle \vec{x_i}, \vec{x_i} \rangle^2 \\
&= x_{i_1}^2 x_{i_1}^2 + 2 x_{i_1} x_{i_1} x_{i_2} x_{i_2} + x_{i_2}^2 x_{i_2}^2 \\
\langle (2,2),(2,2) \rangle^2 &= 2^2 \cdot 2^2 + 2^5 + 2^2 \cdot 2^2 = 64 \\
normal(\langle (2,2),(2,2) \rangle^2) &= \frac{64}{8 \cdot 8} = 1
\end{aligned}
$$

Allgemein:

$$
\begin{aligned}
\langle x, y \rangle &= ||x|| \cdot ||y|| cos\angle(x,y) \\
\frac{\langle x, y \rangle}{\sqrt{\langle x, x \rangle}\sqrt{\langle y, y \rangle}} &= \frac{||x|| \cdot ||y||}{||x|| \cdot ||y||} cos\angle(x,y) \\
\frac{\langle x, x \rangle}{\sqrt{\langle x, x \rangle}\sqrt{\langle x, x \rangle}} &= cos\angle(x,x) = 1
\end{aligned}
$$

1-Klassen-SVM – formal

Die 1-Klassen-SVM schiebt die Hyperebene so, dass +1 für (fast) alle Beispiele gilt.
ρ gibt den Abstand zum Ursprung an. Da es in der Zielfunktion aufgeführt ist, wird es (mit) optimiert.

Primales Problem der 1-Klassen-SVM

$$\min_{\beta,\rho,\xi} = \| \vec{\beta} \|^2 - 2\rho + C \sum \xi^2$$

so dass $\vec{\beta}' \phi(\vec{x_i}) \geq \rho - \xi, \forall i,$
wobei $\rho = \vec{\beta}' \phi(\vec{x_i})$

Merkmalsabbildung ϕ

- Der Abstand ρ zum Ursprung bezieht sich auf den Merkmalsraum, der durch ϕ aufgespannt wird.
- Die Kernfunktion $k(\vec{x_i}, \vec{x_j})$ ist bei der 1-Klassen-SVM erweitert zu $\tilde{k}(\vec{x_i}, \vec{x_j})$, so dass

$$\forall i = 1, ..., N : \tilde{k}(\vec{x_i}, \vec{x_i}) = \kappa + \frac{1}{C} = \tilde{\kappa}$$

- Entsprechend $\langle \tilde{\phi}(\vec{x_i}), \tilde{\phi}(\vec{x_j}) \rangle = \tilde{k}(\vec{x_i}, \vec{x_j})$ mit nicht-linearer Abbildung

$$\tilde{\phi}(\vec{x}) = \begin{bmatrix} \phi(\vec{x}) \\ \frac{1}{\sqrt{C}} \vec{e_i} \end{bmatrix}$$

wobei $\vec{e_i}$ nur an i-ter Stelle eine 1, sonst 0 hat.

SVDD – formal

Primales Problem der SVDD

$$\min_{R,c,\xi} = R^2 + C \sum_{i=1}^{N} \xi_i$$

so dass $\| \vec{x_i} - c \|^2 \leq R^2 + \xi_i, \xi_{\geq} 0, \forall i$

Duales Problem der SVDD

$$\max_{\alpha} = \sum_{i=1}^{N} \alpha_i k(\vec{x_i}, \vec{x_i}) - \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j k(\vec{x_i}, \vec{x_j}) \text{ so dass}$$

$$\alpha_i \geq 0, i = 1, ..., N$$

$$\sum \alpha_i = 1$$

SVM Betrachtung als Hyperkugel

Die 1-Klassen-SVM und die SVDD betrachten die Daten in einer umschließenden Kugel. Wenn wir diese Kugel effizient berechnen, lösen wir damit auch das Optimierungsproblem von 1-Klassen-SVM und SVDD und sogar das der 2-Klassen-SVM. Wir lernen hier zwei Methoden kennen:

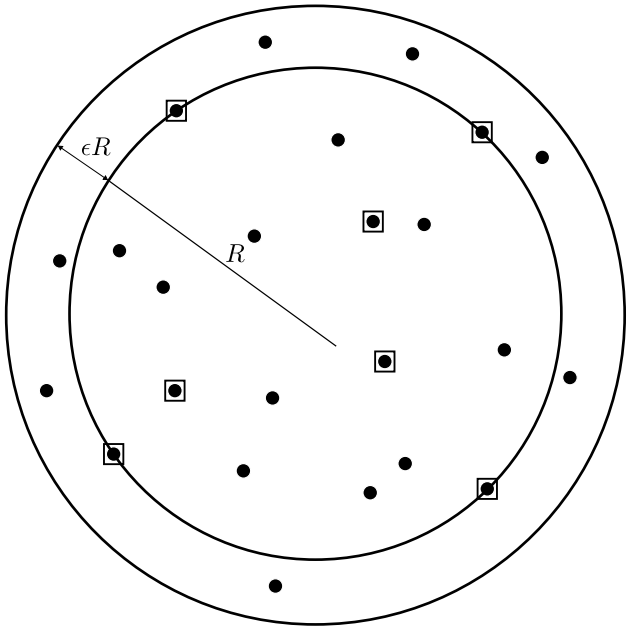
- Core Vector Machine
- Ball Vector Machine

Aber zunächst das Ausgangsproblem: Minimum Enclosing Ball.

Minimum Enclosing Ball

MEB Problem

Gegeben eine Menge $S = \{x_1, ..., x_N\}$, finde den minimum enclosing ball $MEB(S) = B(c, R)$, der alle Daten enthält, mit c Zentrum und R minimalem Radius.
$B(c, (1+\epsilon)R)$ ist eine $1+\epsilon$-Approximation, $\epsilon > 0$, wenn $R \leq MEB(S), S \subset B(c, (1+\epsilon)R)$.



Bernd Gärtner (1999)
Fast and Robust Smallest Enclosing Balls,
in: Europ. Symposion on Algorithms

Core Sets, Algorithmus – Skizze

Core Set

Eine Teilmenge aller Beispiele $Q \subseteq S$ heißt Core Set mit $MEB(Q) = B(c, r)$, wenn S in dem um ϵ erweiterten Ball enthalten ist, $S \subset B(c, (1 + \epsilon)R)$.

In jeder Iteration wird ein Punkt dem Core Set hinzugefügt, der am weitesten vom Zentrum entfernt ist, bis kein Punkt aus S mehr außerhalb $B(c, (1 + \epsilon)R)$ liegt.

Tsang, Kwok, Cheung (2005) Core Vector Machines: Fast SVM Tarining on Very Large Data Sets, JMLR 6 (http://jmlr.csail.mit.edu/)

Core Vector Machine

Die SVM wird stets approximativ gelöst (working set).
Ein Core Set ist eine Menge, mit der man ein fast so gutes Ergebnis erzielt wie mit allen Beispielen.
Die Core Vector Machine nutzt diese Approximation, um z.B. Intrusion Detection über 5.000.000 Beispielen in 1,4 Sekunden zu lernen!

Wir wollen das SVM-Problem über das MEB-Problem lösen.
Wir wollen Kernfunktionen einbeziehen.
Diese werden normalisiert (s.o. Beispielrechnung).
Kernmethode, als MEB formuliert

- Transformierte Kern-Funktion $\tilde{k}$
- Merkmalsraum $\tilde{\mathbb{F}}$ mit Featuremap $\tilde{\varphi}$
- Konstante $\tilde{\kappa} = \tilde{k}(z, z)$

Core Vector Machine - Algorithmus

Core Vector Machine Algorithmus

1. Initialisiere S_0, c_0 und R_0.
2. Terminiere, wenn es keinen Trainingspunkt z mehr gibt, der mit $\tilde{\varphi}(z)$ außerhalb der Kugel $B(c_t, (1+\epsilon)R)$ liegt.
3. Finde z, so dass $\tilde{\varphi}(z)$ am weitesten entfernt von c_t ist. Setze $S_{t+1} = S_t \cup z$.
4. Berechne den neuen $MEB(S_{t+1})$.
5. Setze $t = t + 1$ und gehe zu (2).

Schritt 1: Initalisierung

- Starte mit einem zufälligen Punkt $z \in S$ und finde den von z entferntesten Punkt $z_a \in S$.
- Finde einen anderen von z_a entferntesten Punkt $z_b \in S$.
- CoreSet $S_0 := \{z_a, z_b\}$.
- Mittelpunkt $c_0 = \frac{1}{2}(\tilde{\varphi}(z_a) + \tilde{\varphi}(z_b))$
- Radius $R_0 = \frac{1}{2}\,||\tilde{\varphi}(z_a) - \tilde{\varphi}(z_b)|| = \frac{1}{2}\sqrt{2\tilde{\kappa} - 2\tilde{k}(z_a, z_b)}$
- Bei der binären Klassifikation sollten z_a und z_b aus verschiedenen Klassen kommen.

Schritt 2, 3: Abstandsberechnung

Der Abstand $||c_t - \tilde{\varphi}(z_l)||$ muss in Schritt 2 (Terminierung) und Schritt 3 (Finden eines entferntesten Punktes) berechnet werden.

$$||\vec{c_t} - \tilde{\varphi}(z_l)||^2 = \sum_{z_i, z_j \in S_t} \alpha_i \alpha_j \tilde{k}(z_i, z_j) - 2 \sum_{z_i \in S_t} \alpha_i \tilde{k}(z_i, z_l) + \tilde{k}(z_l, z_l)$$

Random Sampling $S' \subset S$, so dass der Abstand zum Mittelpunkt nur für S' berechnet wird, beschleunigt die Laufzeit auf $O(|S_t|^2)$ in der t-ten Iteration.

Schritt 4: MEB berechnen

Wir haben in Iteration t den Core Set S_t und ermitteln aus diesem den $MEB(S_t)$.

- Die Core Vektoren S_t erfüllen die (weichen) KKT-Bedingungen.
- Die Nicht-Core-Vektoren sind innerhalb der Kugel, $\alpha = 0$, also gelten die (weichen) KKT-Bedingungen.
- Wenn die CVM terminiert, erfüllen alle Beispiele die KKT-Bedingungen mit nur minimaler Abweichung.

Wir lösen das duale Problem über dieser kleineren Menge von Beispielen mit SMO.
Mit den Core Vektoren bestimmen wir den working set.
Bei kleinem ϵ wird nähert sich die Lösung der exakten, dafÂÿr dauert es länger.
Laufzeit: $\frac{1}{\epsilon^4}$

Core Vector Machine

Die Core Vector Machine löst das Optimierungsproblem auf dem Core Set in jeder Iteration (Schritt 4):

$$\max_{\alpha} = -\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j k(\vec{x_i}, \vec{x_j}) \text{ so dass}$$

$$\alpha_i \geq 0, i = 1, ..., N$$

$$\sum \alpha_i = 1$$

Das Lösen von Optimierungsproblemen ist aufwändig.
Bei großen Datenmengen kann auch das Core Set groß werden.
Eine Heuristik kann vielleicht ähnlich gute Ergebnisse liefern?

Ball Vector Machine

Heuristischer Ansatz vereinfacht das MEB Problem durch einen festen Radius:

Enclosing Ball Problem

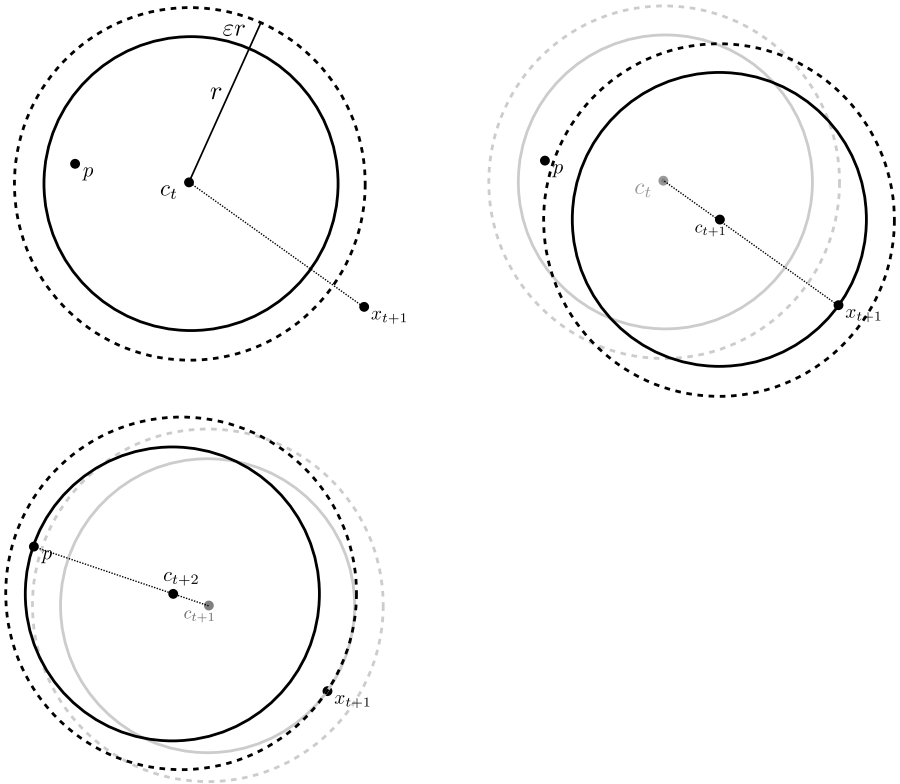
Gegeben der Radius $r > R^*$,
finde den Ball B(c, r), der alle Punkte $\phi(x) \in S$ umschließt:

$$||c - \phi(\vec{x}_i)||^2 \leq r^2$$

- Dazu gibt es einen Approximationsalgorithmus, der ausnutzt, dass der Radius fest ist. Welche Punkte liegen außerhalb von $r(1 + \epsilon)$?
- Der Radius wird gewählt als $\sqrt{\tilde{\kappa}}$.
- Der Ball wird in jeder Iteration verschoben.
  Das Zentrum wird in jeder Iteration neu berechnet:
  minimiere $\|c - c_t\|^2$ so dass $r^2 \geq \|c - \phi(x_t)\|^2$

Algorithmus anschaulich

Man findet einen Punkt außerhalb, verschiebt das Zentrum,...
Möglicherweise oszilliert der Prozess und die Lösung verbessert sich nicht.



Was wissen Sie jetzt?

- Sie kennen die Aufgabenstellungen:
  - Semi-Überwachtes Lernen
  - Transduktion
  - 1-Klassen-SVM
  - SVDD zu Daten-Beschreibun
- Sie haben eine Vorstellung, wie man SVM-Probleme als Minimum Enclosing Ball Problem sehen kann.
- Sie kennen die Methode, den working set einer SVM durch den Core Set zu bestimmen.
- Sie kennen die Heuristik, mit festem Radius das SVM-Problem durch das MEB Problem zu lösen.