Vorlesung Wissensentdeckung in Datenbanken

SVM — Kerne und Optimierung

Kristian Kersting, (Katharina Morik), Claus Weihs

LS 8 Informatik
Computergestützte Statistik
Technische Universität Dortmund

03.06.2014

Was bisher bei der SVM geschah

Das primale Optimierungsproblem ist definiert als:

$$L_P = \frac{1}{2}\|\vec{\beta}\|^2 - \sum_{i=1}^{N} \alpha_i \left[y_i(\left\langle \vec{x}_i \, , \, \vec{\beta} \right\rangle + \beta_0) - 1 \right]$$

mit den *Lagrange-Multiplikatoren* $\vec{\alpha}_i \geq 0$.
Notwendige Bedingung für ein Minimum liefern die Ableitungen nach $\vec{\beta}$ und β_0

$$\frac{\partial L_P}{\partial \vec{\beta}} = \vec{\beta} - \sum_{i=1}^{N} \alpha_i y_i \vec{x}_i \quad \text{und} \quad \frac{\partial L_P}{\partial \beta_0} = \sum_{i=1}^{N} \alpha_i y_i$$

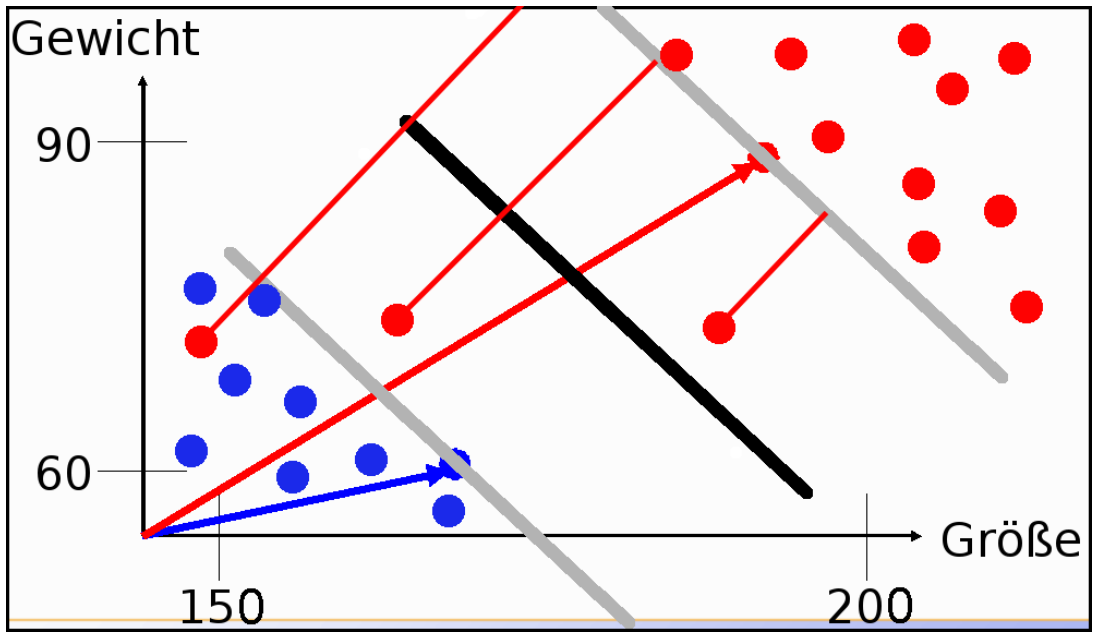
Diese führen zum *dualen Optimerierungsproblem*

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{i'=1}^{N} \alpha_i \alpha_{i'} y_i y_{i'} \left\langle \vec{x}_i \, , \, \vec{x}_{i'} \right\rangle$$

Gliederung

SVM mit Ausnahmen

- Was passiert, wenn die Beispiele nicht komplett trennbar sind?



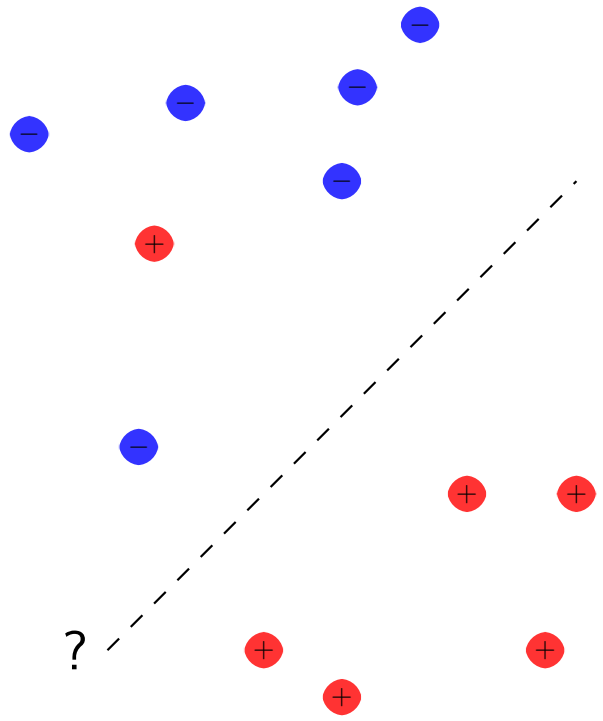
Nicht linear trennbare Daten

Bisher haben wir angenommen, dass die Daten linear trennbar sind.

In der Praxis sind linear trennbare Daten aber selten

Eine erste Idee: Entferne eine minimale Menge von Datenpunkten, so dass die Daten linear trennbar werden (minimale Fehlklassifikation).

Problem: Algorithmus wird kombinatorisch (also wahrscheinlich exponentiell).



SVM mit Ausnahmen via Relaxierung

Ein anderer Ansatz basiert wieder auf einer Relaxation, d.h. wir lockern einige der Nebenbedingungen in dem mathematischen Programm:

- Punkte, die nicht am Rand oder auf der richtigen Seite der Ebene liegen, bekommen einen Strafterm $\xi_j > 0$.
- Korrekt klassifizierte Punkte erhalten eine Variable $\xi_j = 0$.

Dies führt zu folgender Zielfunktion

$$\frac{1}{2}\|\vec{\beta}\|^2 + C\sum_{j=1}^{N}\xi_j \quad \text{für ein festes } C \in \mathbb{R}_{>0}\ , \tag{1}$$

Der Parameter $C \in \mathbb{R}$ bestimmt unsere Gewichtung der beiden Zwillingsziele (1) $\|\vec{\beta}\|^2$ klein (Margin groß) und (2) Strafterme klein.

Weich trennende Hyperebene: das Optimierungsproblem

Relaxiertes primales Optimierungsproblem

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

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

unter den Nebenbedingungen

$$\begin{aligned} \left\langle \vec{x}_i \, , \, \vec{\beta} \right\rangle + \beta_0 &\geq +1 - \xi_i \quad \text{für } \vec{y}_i = +1 \\ \left\langle \vec{x}_i \, , \, \vec{\beta} \right\rangle + \beta_0 &\leq -1 + \xi_i \quad \text{für } \vec{y}_i = -1 \\ \xi_i &\geq 0 \quad \text{für allle } i \end{aligned}$$

Durch Umformung erhalten wir wieder Bedingungen für die Lagrange-Optimierung:

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

Weich trennende Hyperebene

Jetzt können wir wieder mittels Lagrange-Multiplikatoren das duale Problem bestimmen[1]:

Relaxiertes duals Optimierungsproblem

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

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

unter den Bedingungen

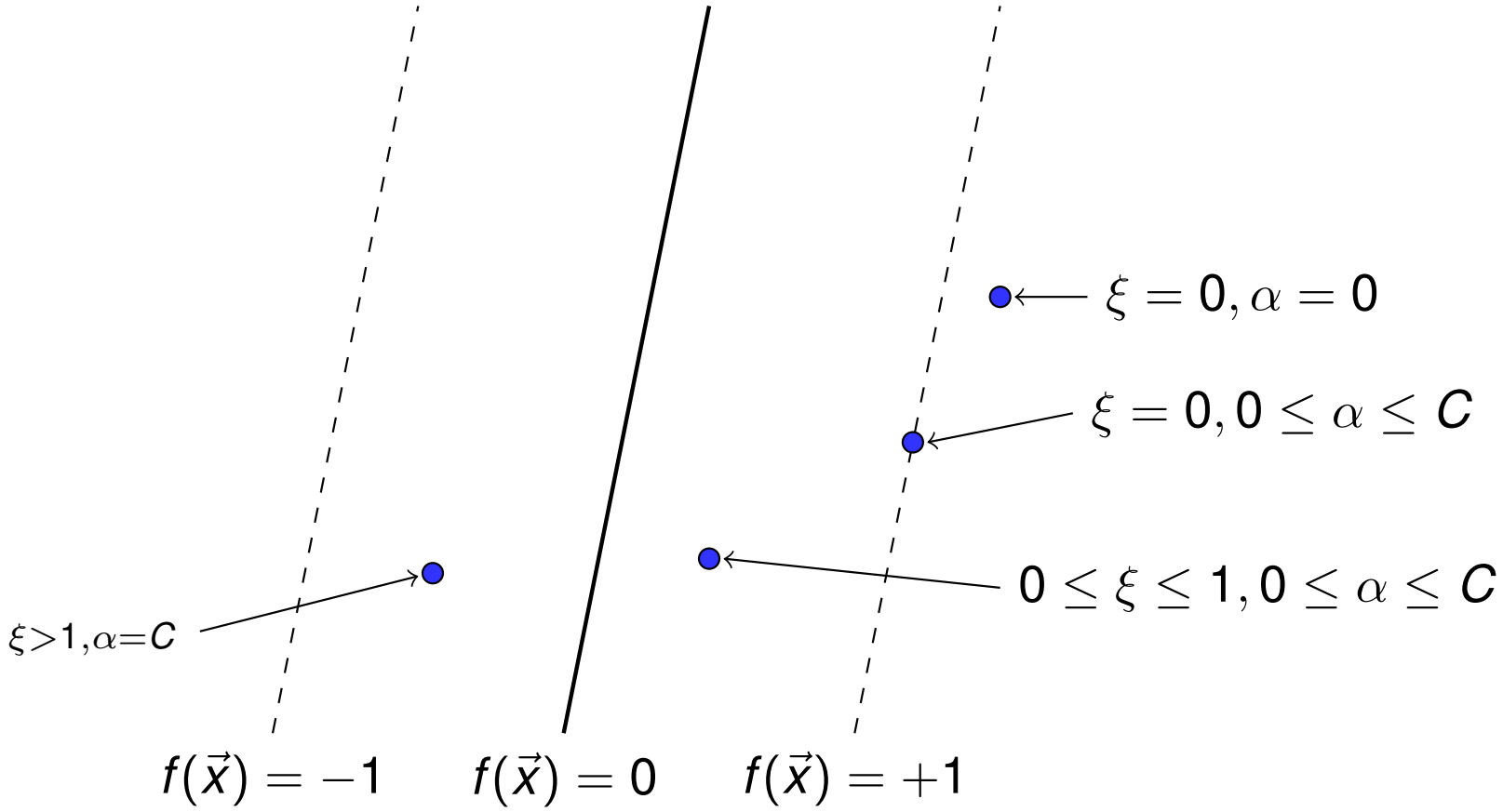
$$0 \leq \alpha_i \leq C \; \forall \, i = 1, \ldots, N \quad \text{und} \quad \sum_{i=1}^{N} \alpha_i y_i = 0$$

Überraschung: einzige Änderung ist $0 \leq \alpha_i \leq C$.

[1] Herleitung nicht gezeigt. Nach dem "gleich 0"-Setzen der partiellen Ableitungen von $\vec{\beta}$ und β_0 wie zuvor, dem Einsetzen der Ergebnisse und einigen Vereinfachungen erhalten wir das hier angegebene duale Problem.

Bedeutung von ξ und $\vec{\alpha}$

Die Berechnung von β_0 (hier nicht gezeigt) und vor Allem die Interpretation der α_is ändern sich!



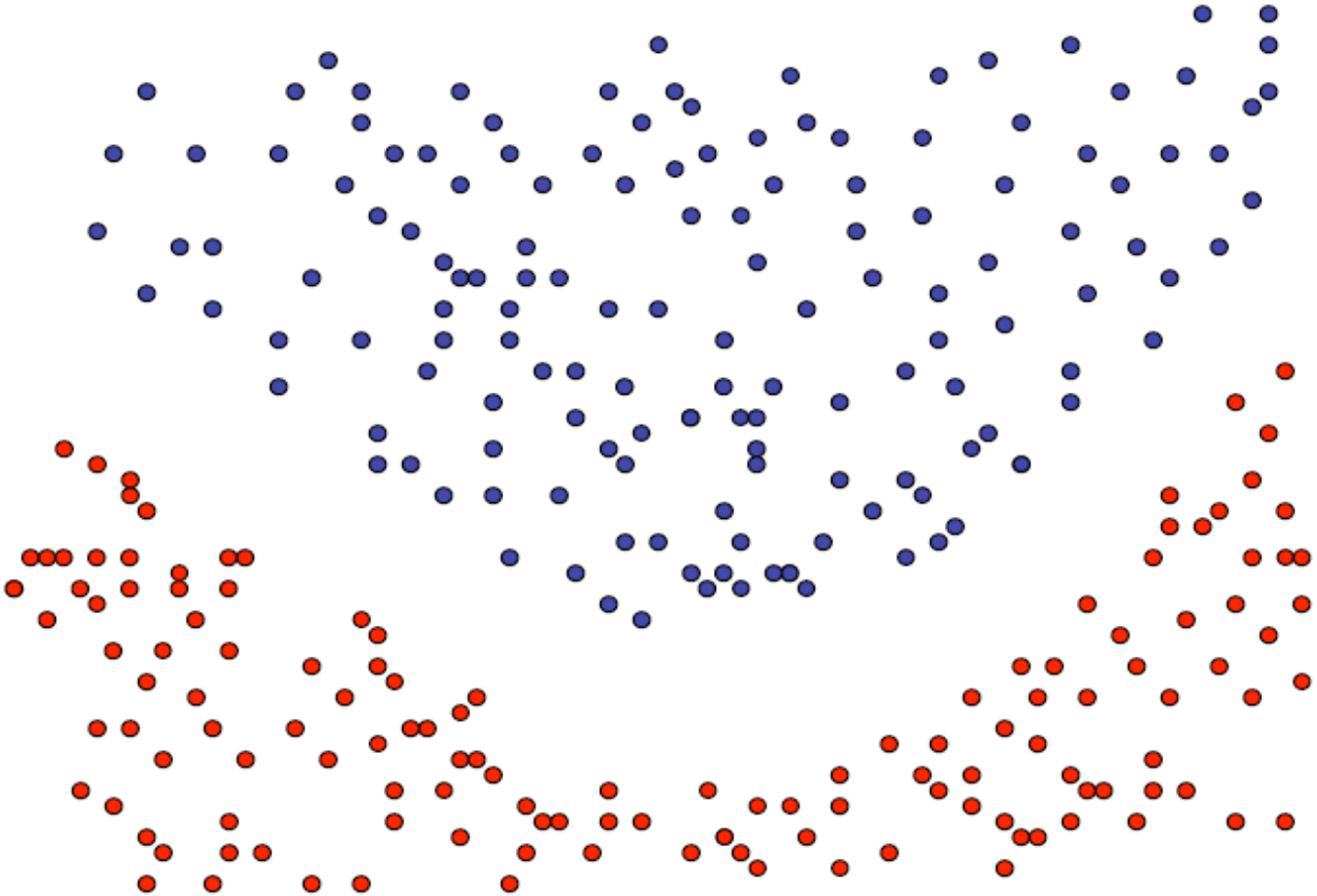
Beispiele $\vec{x}_i$ mit $\alpha_i > 0$ sind Stützvektoren.

Wo stehen wir jetzt?

- Maximieren der Breite einer separierenden Hyperebene (maximum margin method) ergibt eindeutige, optimale trennende Hyperebene.
- Relaxierung erlaubt eine lineare Trennung von nicht linear trennbaren Klassen (weiche Hyperbene).

Bisher gehen wir aber davon aus, dass die Entscheidungsfunktion eine Hyperebene im Merkmalsraum ist. Reicht das?

Nein!!! Nicht-lineare Daten



Eine (weich) trennende Hyperebene im Merkmalsraum reicht hier nicht aus!

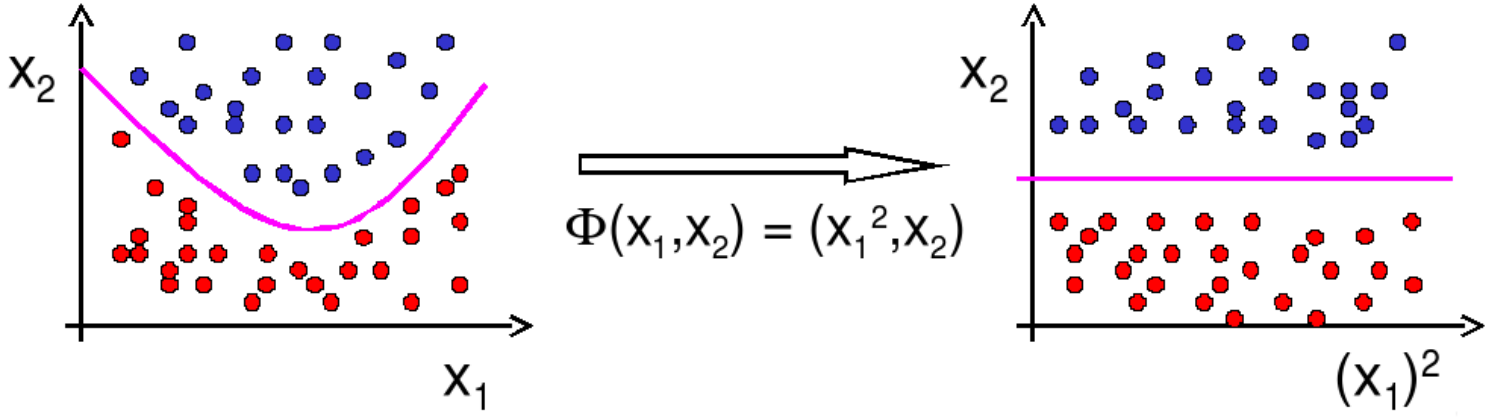
Nicht-lineare Daten

Aber wie machen wir es dann?

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

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

- Ja genau, aber erst nach Transformation Φ in ein lineares Problem!



Kernfunktionen

- Erinnerung duales Problem (ohne Slack):

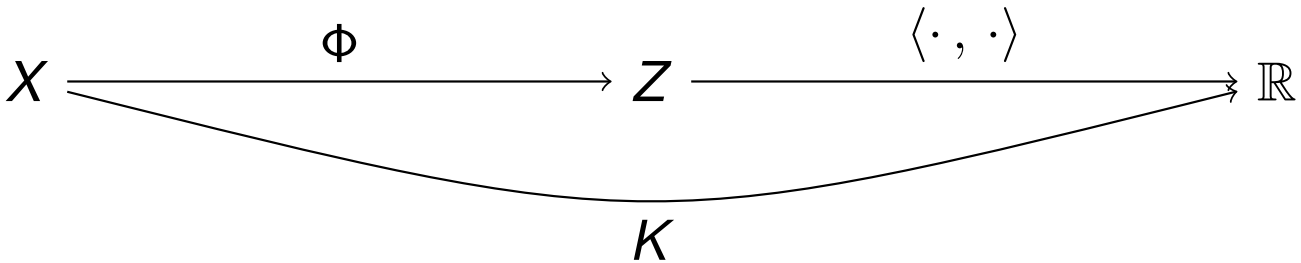
$$L_D(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} y_i y_j \alpha_i \alpha_j \left\langle \vec{x_i}, \vec{x_j} \right\rangle$$

$$\alpha_i \geq 0 \; \forall i = 1, \ldots, m \text{ und } \sum_{i=1}^{m} \alpha_i y_i = 0$$

- Wie sehen, dass die SVM von $\vec{x}$ nur über Skalarprodukt $\left\langle \vec{x}, \vec{x'} \right\rangle$ abhängt.

Idee: Ersetze Transformation Φ und Skalarprodukt durch Kernfunktion

$$K(\vec{x_1}, \vec{x_2}) := \left\langle \Phi(\vec{x_1}), \Phi(\vec{x_2}) \right\rangle$$



Houston, do we have a problem?

Aber ist das nicht viel zu aufwendig? Wollen wir wirklich explizit in einem hoch-dimensionalen Innenproduktraum operieren?

Der Kern-Trick

Angabe von Φ nicht nötig, einzige Bedingung:

> *Kernmatrix* $\mathbf{K} = (K(\vec{x}_i, \vec{x}_j))_{i,j=1\ldots N}$ *muss symmetrisch und positiv semi-definit sein, d.h.* $\vec{z}^T \mathbf{K} \vec{z} \geq 0$.

Manchmal nennt man die Kernmatrix auch Gram-Matrix.

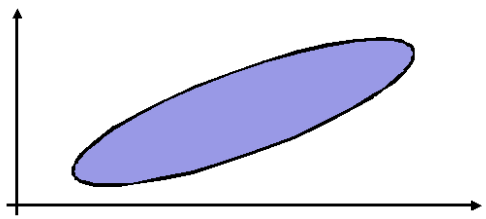
$$\begin{aligned}
\vec{z}^T \mathbf{K} \vec{z} &= \sum_i \sum_j z_i K_{ij} z_j = \sum_i \sum_j z_i \Phi(\vec{x}_i)^T \Phi(\vec{x}_j) z_j \\
&= \sum_i \sum_j z_i \sum_k \Phi_k(\vec{x}_i)^T \Phi_k(\vec{x}_j) z_j = \sum_k \sum_i \sum_j z_i \Phi_k(\vec{x}_i)^T \Phi_k(\vec{x}_j) z_j \\
&= \sum_k \sum_i \left(z_i \Phi_k(\vec{x}_i) \right)^2 \geq 0
\end{aligned}$$

Einige Beispiele für Kernfunktionen

- Polynom: $K(\vec{x}_i, \vec{x}_j) = \left\langle \vec{x}_i \, , \, \vec{x}_j \right\rangle^d$
- Radial-Basisfunktion (RBF): $K(\vec{x}_i, \vec{x}_j) = \exp(-\gamma \|\vec{x}_i - \vec{x}_j\|^2)$
- Neuronale Netze: $K(\vec{x}_i, \vec{x}_j) = \tanh(\left\langle \alpha \vec{x}_i \, , \, \vec{x}_j \right\rangle + b)$
- Konstruktion von Spezialkernen durch Summen und Produkte von Kernfunktionen, Multiplikation mit positiver Zahl, Weglassen von Attributen. Das ist z.B. spannend, um Kernfunktionen für Graphen zu design.

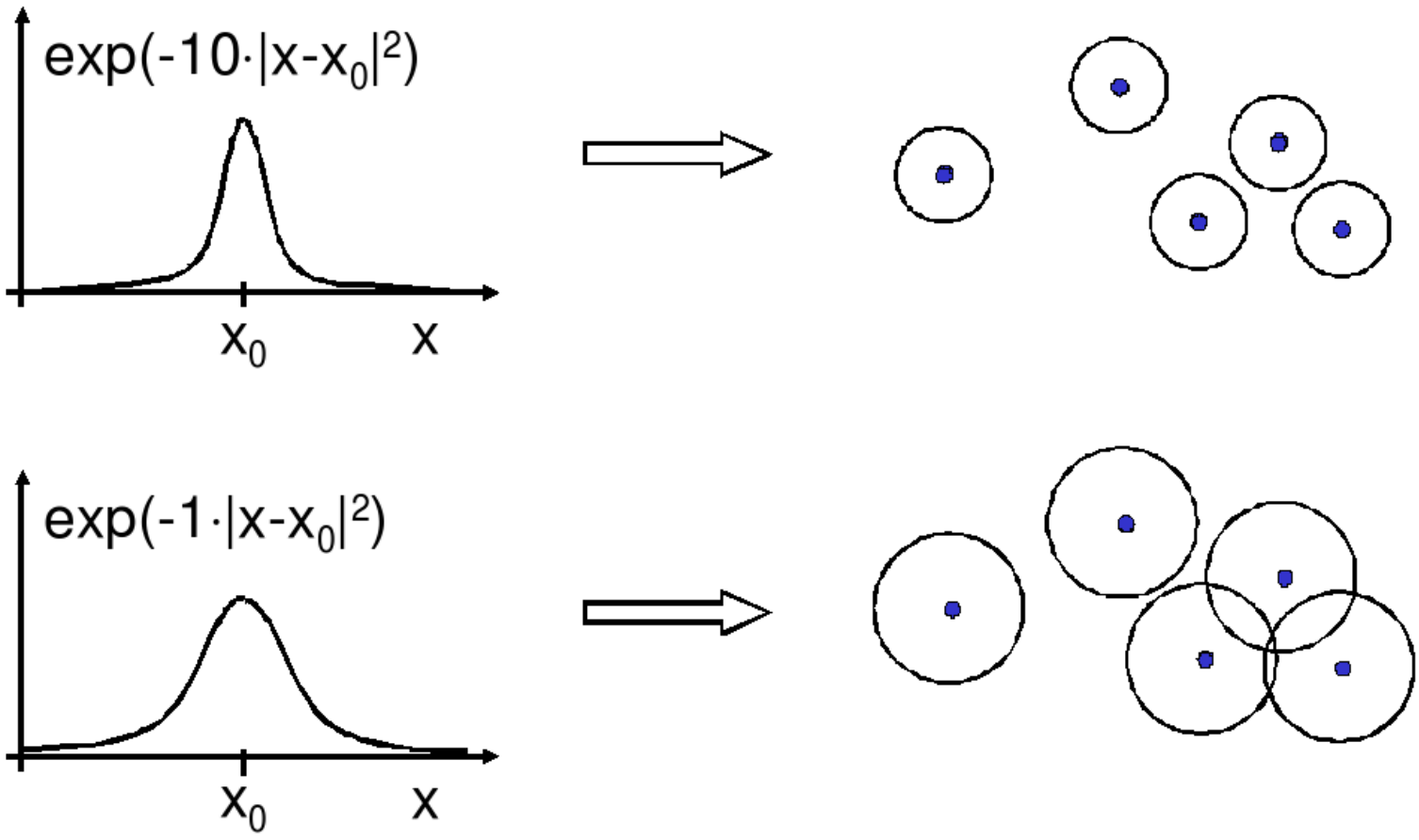
Beispiel: Polynom-Kernfunktionen

- $K_d(\vec{x}_i, \vec{x}_j) = \left\langle \vec{x}_i \, , \, \vec{x}_j \right\rangle^d$
- Beispiel: $d = 2, \vec{x}_i, \vec{x}_j \in \mathbb{R}^2$.



$$
\begin{aligned}
K_2(\vec{x}_i, \vec{x}_j) &= \left\langle \vec{x}_i \, , \, \vec{x}_j \right\rangle^2 \\
&= (x_{i_1} x_{j_1} + x_{i_2} x_{j_2})^2 \\
&= x_{i_1}^2 x_{j_1}^2 + 2 x_{i_1} x_{j_1} x_{i_2} x_{j_2} + x_{i_2}^2 x_{j_2}^2 \\
&= \left\langle (x_{i_1}^2, \sqrt{2} x_{i_1} x_{i_2}, x_{i_2}^2) \, , \, (x_{j_1}^2, \sqrt{2} x_{j_1} x_{j_2}, x_{j_2}^2) \right\rangle \\
&=: \left\langle \Phi(\vec{x}_i) \, , \, \Phi(\vec{x}_j) \right\rangle
\end{aligned}
$$

Beispiel: RBF-Kernfunktion



SVM mit Kernfunktionen

- Die Kernfunktionen werden nicht als Vorverarbeitungsschritt durchgeführt.
- Man muss lediglich bei der Berechnung des Skalarprodukts die Kernfunktion berücksichtigen.

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

$$\alpha_i \geq 0 \; \forall i = 1, \ldots, m \text{ und } \sum_{i=1}^{m} \alpha_i y_i = 0$$

- Allerdings kann $\vec{\beta}$ — ermittelt aus der dualen Lösung — jetzt nicht mehr so einfach interpretiert werden als Bedeutung der Variablen (Merkmale) X_i.

Interpretation der SVM mit Kernfunktionen

Wenn das Skalarprodukt als Kernfunktion gewählt wird — der lineare Kern — entspricht jede Komponente des $\vec{\beta}$ einem Gewicht des Merkmals und jedes α dem Gewicht eines Beispiels $\vec{x}$, $\phi(\vec{x}) = \vec{x}$.

Warum ist das so. Dazu betrachten wir die Entscheidungsfunktion f der SVM:

$$\begin{aligned} f(\vec{x}) &= \sum_{i=1}^{N} \alpha_i K(\vec{x}_i, \vec{x}) = \sum_{i=1}^{N} \alpha_i \left\langle \Phi(\vec{x}_i), \Phi(\vec{x}) \right\rangle \\ &= \left\langle \sum_{i=1}^{N} \alpha_i \Phi(\vec{x}_i), \Phi(\vec{x}) \right\rangle = \left\langle \vec{\beta}, \Phi(\vec{x}) \right\rangle \end{aligned}$$

Für eine linear SVM — $\Phi(\vec{x}) = x$ — kann β also direkt als die Gewichtung der Merkmale gesehen werden.

Aber wie sieht das für den nicht-linearen Fall aus? Wie finden wir zu jedem $\phi(\vec{x})$ den Ursprung $\vec{x}$?

Diese Frage führt zu dem Pre-Image Problem

Mika, Schölkopf, Smola, Müller, Scholz, Rätsch (1998) Kernel PCA and de-noising in feature spaces, in: NIPS, vol 11.

Rüping (2006) Learning Interpretable Models, Diss. TU Dortmund

Pre-Image Problem

Gegeben die Abbildung $\Phi : X \to \mathcal{X}$ und ein Element aus dem Merkmalsraum, $\vec{\beta} \in \mathcal{X}$, **finde** ein $\vec{x} \in X$, so dass $\Phi(\vec{x}) = \vec{\beta}$.

Pre-Images müssen aber nicht immer existieren. Zwar hat $RBF(\vec{x}_i)$ hat $\vec{x}_i$ als Mittelpunkt, aber man kann eine Gauss'sche Funktion nicht als Linearkombination Gauss'scher Funktionen, die um andere $\vec{x}_j$ zentriert sind, schreiben .

Wir müssen das Problem relaxieren

Approximatives Pre-Image Problem

Gegeben die Abbildung $\phi : X \rightarrow \mathcal{X}$ und ein Element aus dem Merkmalsraum, $\vec{\beta} \in \mathcal{X}$, **finde** ein $\vec{x} \in X$ mit minimalem Fehler $\| \vec{\beta} - \phi(\vec{x}) \|^2$.

Den Ursprung im Merkmalsraum suchen

Weil wir die Abbildung Φ normalerweise nicht kennen, müssen wir den quadratischen Fehler im Merkmalsraum minimieren, um $\vec{x}$ zu finden.

$$\begin{aligned} \vec{x} &= \mathit{argmin} \parallel \vec{\beta} - \Phi(\vec{x}) \parallel^2 \\ &= \mathit{argmin} \left\langle \vec{\beta}, \vec{\beta} \right\rangle - \left\langle 2\vec{\beta}, \Phi(\vec{x}) \right\rangle + \left\langle \Phi(\vec{x}), \Phi(\vec{x}) \right\rangle \\ &= \mathit{argmin} \left\langle \vec{\beta}, \vec{\beta} \right\rangle - 2f(\vec{x}) + K(\vec{x}, \vec{x}) \end{aligned} \tag{3}$$

Minimum von $K(\vec{x}, \vec{x}) - 2f(\vec{x})$ (z.B. mittels Gradientenabstieg) kann das Pre-Image von $\vec{\beta}$ liefern (oder ein lokales Minimum).

Pre-Images lernen!

Wenn wir häufiger für den selben Merkmalsraum den Ursprung $\vec{x}$ von $\Phi(\vec{x})$ bestimmen wollen, dann lohnt es sich, die umgekehrte Abbildung $\Gamma : \mathcal{X} \rightarrow X$ zu lernen.

Allerdings müssen wir dann für den Merkmalsraum eine geeignete Basis finden, z.B. durch eine Hauptkomponentenanalyse mit Kernfunktion.

Auf dieser Basis wird dann für eine kleinere Menge $\vec{x_i}$ die Abbildung Γ approximiert.

Bakir, Weston, Schölkopf (2003) Learning to find pre-images, in: NIPS, vol. 16

Reduced Set Problem

Reduced Set Problem

Gegeben die Abbildung $\Phi : X \rightarrow \mathcal{X}$ und eine natürliche Zahl s, **finde** $\vec{z_1}, ..., \vec{z_s} \in X$ und Koeffizienten $\gamma_1, ..., \gamma_s$ so, dass $\| \vec{\beta} - \sum_{i=1}^{s} \gamma_i \phi(\vec{z_i}) \|^2$ minimal ist.

Das gelernte $\vec{\beta} = \sum_{i=1}^{N} \alpha_i \phi(\vec{x_i})$ ist eine Linearkombination der Stützvektoren. Diese sind die erste Lösung des Problems.

Wir wollen aber nicht alle Daten bearbeiten, sondern nur $s << N$!

Wir wollen $\vec{\gamma}$ aus weniger Beispielen lernen. Das ist möglich, weil hier nicht die Nebenbedingungen gelten wie bei dem Optimierungsproblem der SVM.

Neues Optimierungsproblem

SVM liefert $\vec{\beta} = \sum_i^N \alpha_i \phi(\vec{x}_i)$.

Wir wollen die Euklidische Distanz der Approximation zur originalen SVM minimieren:

$$\| \sum_{i=1}^{N} \alpha_i \phi(x_i) - \sum_{i=1}^{N} \gamma_i \phi(x_i) \|^2 + \lambda \sum |\gamma_i|$$

und γ soll spärlich besetzt sein (l_1-Regularisierung). $\lambda > 0$ gewichtet die Spärlichkeit gegen die Präzision.

Schölkopf, Mika, Burges, Knirsch, Müller, Rätsch, Smola (1999) Input space versus feature space in kernel-based methods. IEEE Neural Networks Vol.10, No. 5

Iterativer Algorithmus – Skizze

Das können wir auch über eine aktive Menge Z_k iterativ approximieren:

1. k=:0; $Z_k =: \{\}$
2. **Finde** neue z_{k+1}. Dazu lösen wir das "1D"-Problem (**??**) für

$$\beta_\Delta = \sum_{i=1}^{N} \alpha_i \phi(x_i) - \sum_{i=1}^{k} \beta_i \phi(z_i)$$

3. **Setze** $Z_{k+1} =: Z_k \cup z_{k+1}$; k=: k+1;
4. Für eine gegebene aktive Menge Z_k können wir die optimalen γs berechnen:

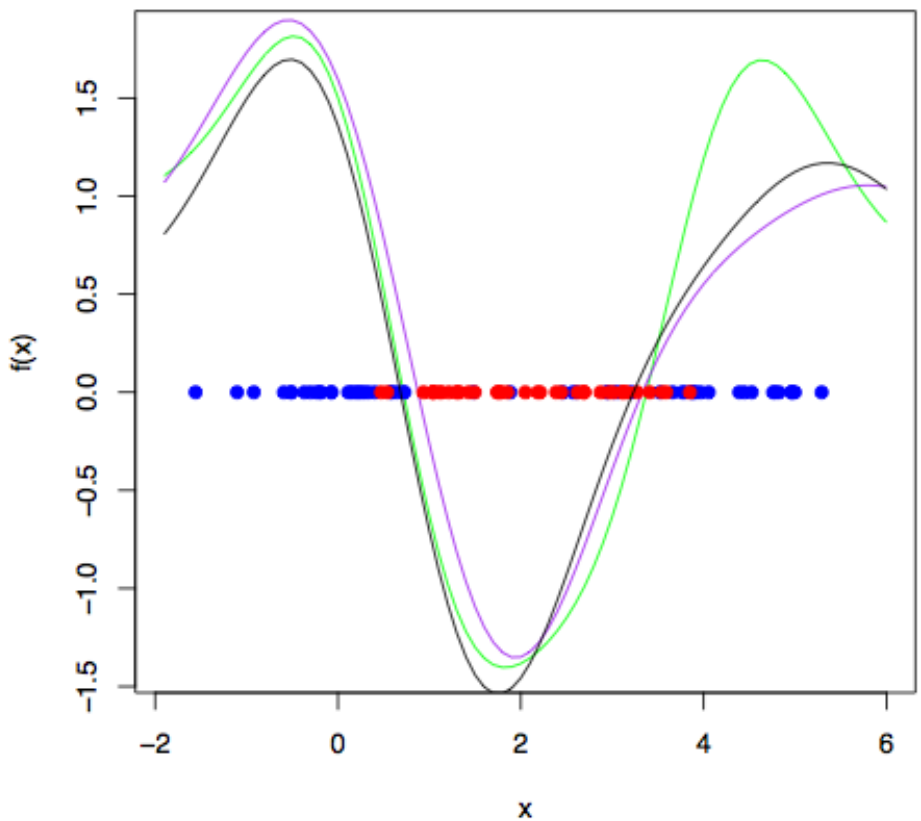
$$\gamma = (\mathbf{K}^z)^{-1} \mathbf{K}^{zx} \alpha$$

   weil $\mathbf{K}^{zx}\alpha = \mathbf{K}^z\gamma$ gilt (hier nicht gezeigt) mit $\mathbf{K}^{zx} = \left\langle \phi(\vec{z_i}), \phi(\vec{x_i}) \right\rangle$ und $\mathbf{K}^z = \left\langle \phi(\vec{z_i}), \phi(\vec{z_j}) \right\rangle$

5. **Wenn** $\| \sum_{i=1}^{N} \alpha_i \phi(x_i) - \sum_{i=1}^{k} \gamma_i \phi(z_i) \|^2 + \lambda \sum |\gamma_i| < \theta$, stop, sonst Schritt 2!

Reduced Set Approximation – Bild

Bei einem eindimensionalen Datensatz mit Klassen blau und rot, sieht man die Funktionswerte der tatsächlichen Funktion (grün), die Approximation lt. Schölkopf et al (1999) (lila) und die Approximation lt. Rüping (2006) (schwarz):



Was wissen Sie jetzt?

- Weiche Margin mittels Strafterme (Slack)
- Kernfunktionen - eine Transformation, die man nicht erst durchführen und dann mit ihr rechnen muss, sondern bei der nur das Skalarprodukt gerechnet wird.
- Lineare SVM sind leicht zu interpretieren: α gewichtet Beispiele, β gewichtet Merkmale.
- Bei nicht-linearen SVM wissen wir i.A. für gegebenen Wert $\phi(\vec{x})$ nicht, welches $\vec{x}$ dahinter steht.
- Lösung: zu einer nicht-linearen SVM noch eine Approximation der SVM lernen!
  - Die gelernte SVM klassifiziert mit max margin.
  - Die Approximation gibt eine Vorstellung von der Funktion.
  - Das Reduced Set Problem findet eine Approximation für wenige Beispiele mit γ statt β auf der Grundlage eines gelernten Modells.

Lösung des SVM-Optimierungsproblems

Aber wie lösen wir denn nun das eigentliche Optimierungsproblem der SVM?

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...

- Also, jetzt 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 $0 \leq \alpha_i \leq C \; \forall i = 1, \ldots, N$

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

Wie machen wir das?

Generell verfolgen wir wieder einmal einer Gradientensuche!

1. 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 gerade m Komponenten, nach denen es optimiert werden muss.

Sollen wir wirklich alle Komponenten von α auf einmal optimieren?
Das sind m^2 Terme, die wir aufgrund des quadratischen Optimierungsproblems zu beachten haben! Zu aufwändig.

Wie machen wir das?

Wenn wir nicht alles auf einmal schaffen können, dann können wir ja vielleicht nur einige Komponenten optimieren. Wenn nur einige Komponenten optimiert werden, während die anderen festgehalten werden, dann nennt man das ein koordiniertes Gradientenverfahren!

2. Also nur eine Komponente von α ändern, wobei die anderen fest gehalten werden? Geht leider nicht weil dann die "summiere zu 0"-Nebenbedingung verletzt wird. Wenn wir etwas ändern müssen wir auch etwas anderes ändern
3. Daher also zwei Komponenten α_1, α_2 im Bereich $[0, C] \times [0, C]$ verändern!

Sequential Minimal Optimization (SMO)

Platt, John. Fast Training of Support Vector Machines using Sequential Minimal Optimization. Advances in Kernel Methods - Support Vector Learning, B. Scholkopf, C. Burges, A. Smola, eds., MIT Press (1998).

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

$$\alpha_1 y_i + \alpha_2 y_2 = -\sum_{i=3}^{m} \alpha_i y_i = \xi \ (const.)$$

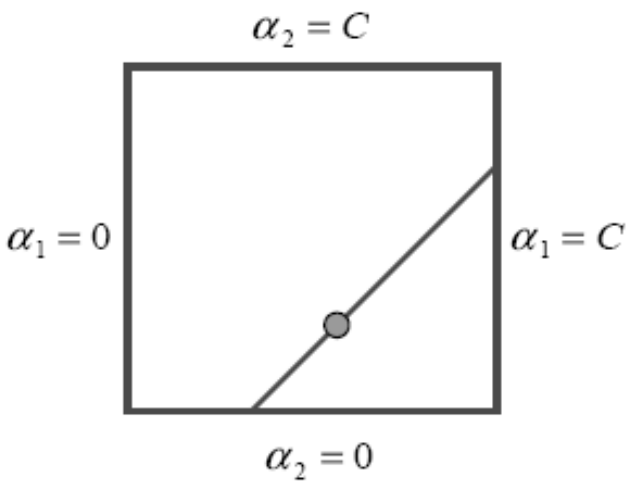
- Zulässige α_1, α_2 liegen im Bereich $[0, C] \times [0, C]$ auf der Geraden
- Ausserdem ist $\xi = \alpha_1 y_1 + \alpha_2 y_2$ äquivalent zu $\alpha_1 + s\alpha_2 = \frac{\xi}{y_1} = \xi y_1$ mit $s = \frac{y_2}{y_1} = y_1 y_2$ da $y_i \in \{-1, 1\}$. Daher optimieren wir zunächst α_2 (weil wir dann α_1 zunächst als fix annehmen können).
- Aus dem optimalen $\hat{\alpha}_2$ können wir dann das optimale $\hat{\alpha}_1$ bestimmen:

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

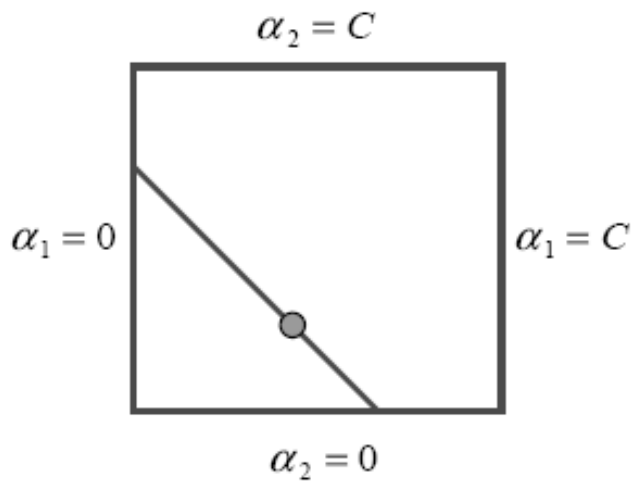
- Dann kommen die nächsten zwei α_i dran...

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...



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



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

α_2 optimieren

- Maximum der Funktion $L'_D(\alpha)$ entlang der Geraden $s\alpha_2 + \alpha_1 = \xi y_1$.
- 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

- Ableiten des dualen Problems nach α_2 ergibt das Optimum für α_2^{new}
  - $\alpha_2^{new} = \alpha_2^{old} + \frac{y_2((f(\vec{x_1}) - y_1) - (f(\vec{x_2}) - y_2))}{\eta} = \alpha_2^{old} + \frac{y_2(E_1 - E_2)}{\eta}$
  - $\eta = k(x_1, x_1) + k(x_2, x_2) - 2k(x_1, x_2)$
- Allerdings gibt es aufgrund der Optimierungsquadrate untere und obere Schranken für α_2 bestimmen:
  - $y_1 = y_2 : L = max(0, \alpha_1^{old} + \alpha_2^{old} - C)\ H = min(C, \alpha_1^{old} + \alpha_2^{old})$
  - $y_1 \neq y_2 : L = max(0, \alpha_2^{old} - \alpha_1^{old})\ H = min(C, C + \alpha_2^{old} - \alpha_1^{old})$
- Daher beschränken wir uns auf diesen Bereich.

Optimales α_2

- Somit wählen wir zum Update:

$$\hat{\alpha}_2 = \begin{cases} H & \text{wenn } \alpha_2^{new} \geq H \\ \alpha_2^{new} & \text{wenn } L < \alpha_2^{new} < H \\ L & \text{wenn } \alpha_2^{new} \leq L \end{cases}$$

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

SMO

- SMO hat ein Laufzeit zwischen $O(N)$ und $O(N^2)$ für viele Probleme. SMO's Laufzeit ist durch die Evaluierung der SVM dominiert, so dass SMO am schnellsten für lineare SVMs und spärliche Daten ist.
- Ein Standardsolver wie z.B. Projected Conjugate Gradient (PCG) zwischen $O(N)$ und $O(N^3)$.
- Der SMO nimmt jeweils ein Paar α_1, α_2 (koordinierter Abstieg). Im Vergleich zu z.B. PCG werden keine komplexen Matrix-Operationen benötigt, so dass SMO einfach zu implementieren ist.

SMO und mehr

- 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 sich gemerkt (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}.

Optimierungscheme wie in SVM^{light}

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.
- Er folgt einem koordinierten Gradientenabstieg.
- Man kann auch z.B. per Evolutionsalgorithmus optimieren (Mierswa 2006).
- Oder die lineare SVM mit der *cutting plane* Methode bearbeiten (Kelley 1960) (Joachims 2006)
- ...