Supportvektormaschinen bei nicht trennbaren Daten

Vorlesung 9, Maschinelles Lernen

Dozenten: Prof. Dr. M. O. Franz, Prof. Dr. O. Dürr

HTWG Konstanz, Fakultät für Informatik

Übersicht

1. Soft Margin SVM

2. Ein Optimierungsalgorithmus für die SVM

Übersicht

1. Soft Margin SVM

2. Ein Optimierungsalgorithmus für die SVM

Wiederholung: Supportvektoralgorithmus als Optimierungsproblem

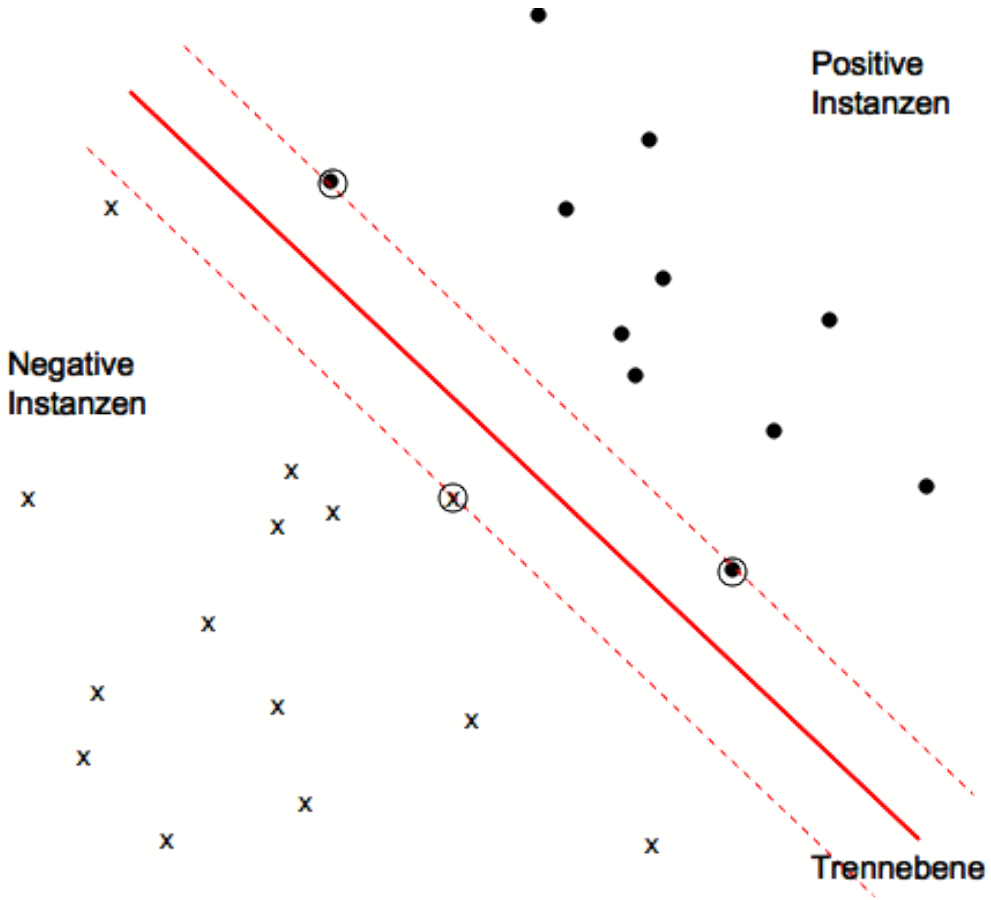
SVM-Optimierung

Für einen gegebenen Trainingsdatensatz
$S = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_\ell, y_\ell)\} \subseteq (X \times Y)^\ell$

- minimiere
$$f(\mathbf{w}) = \langle \mathbf{w} \cdot \mathbf{w} \rangle, \ \mathbf{w} \in \mathbb{R}^d$$
- unter ℓ Nebenbedingungen, $i = 1, \ldots, \ell$
$$y_i(\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b) \geq 1$$

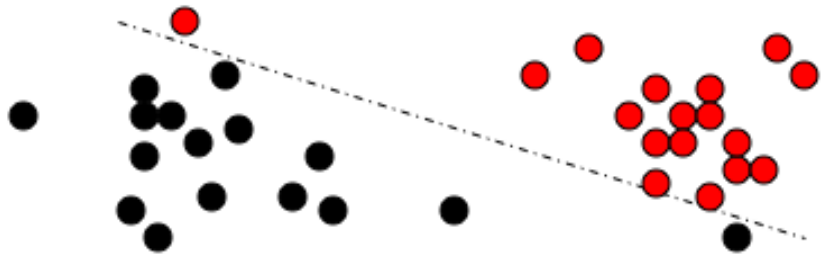
Der SVM-Algorithmus ist ein konvexes Optimierungsproblem, d.h. jedes lokale Minimum ist eine Lösung. Falls das Problem linear trennbar ist, ist die Existenz einer Lösung ist garantiert. Durch das Kriterium der maximalen Trennbreite ist die SVM auch auf hochdimensionale Probleme anwendbar.

Ergebnis: Trennebene mit maximaler Trennbreite



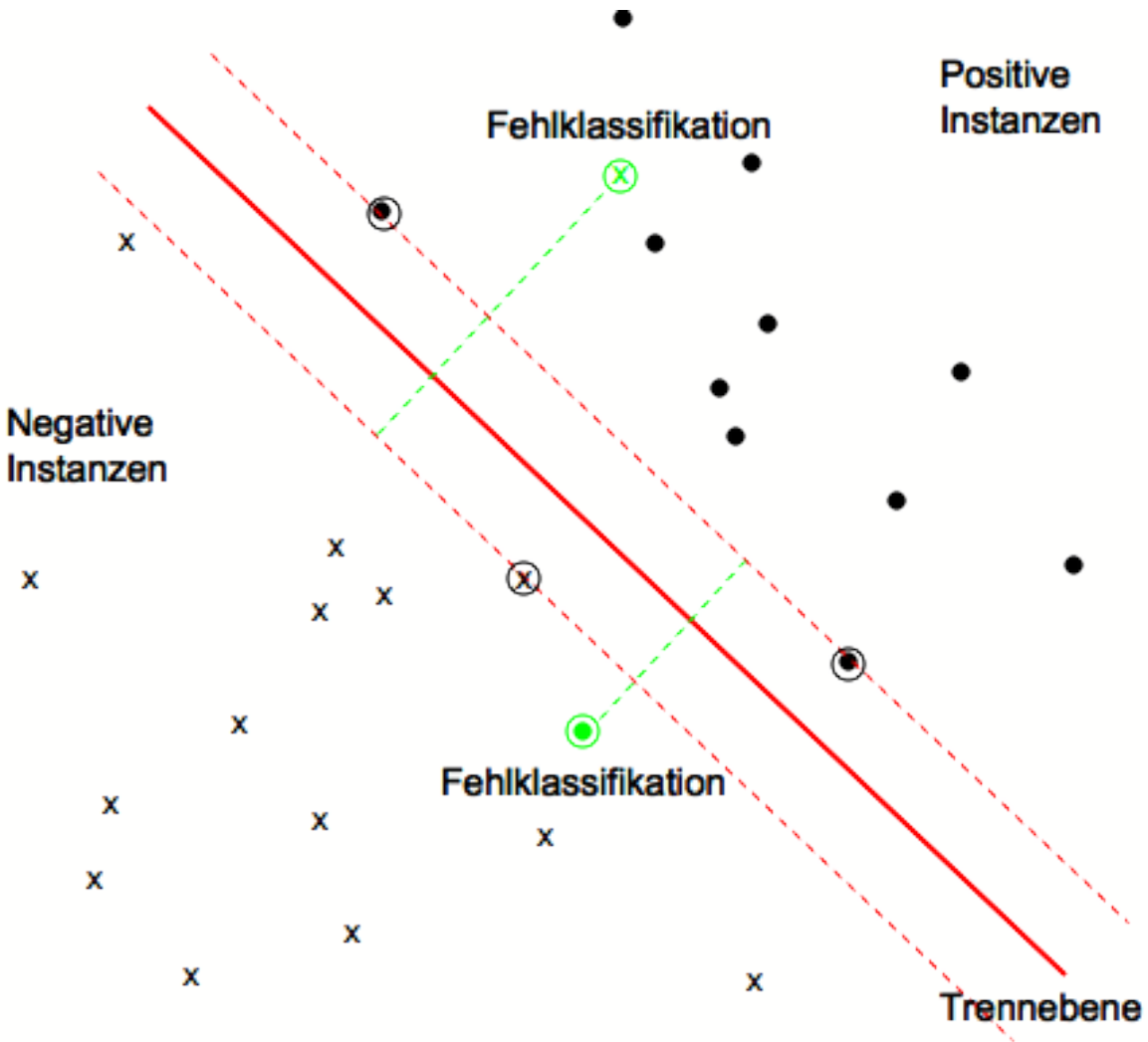
- Zur Lösung des quadratischen Optimierungsproblems existieren leistungsfähige Programmpakete (z.B. quadprog in Matlab), aber meist ist es besser, spezialisierte SVM-Pakete zu verwenden (z.B. svmlib).
- Die Lage der Trennebene wird nur durch die nächstliegenden Datenpunkte, den Supportvektoren, bestimmt. Alle anderen Datenpunkte spielen keine Rolle.
- Aber: wir sind noch nicht am Ziel...

Nicht linear trennbare Daten



- Wendet man die bisher beschriebene SVM-Optimierung auf linear nicht trennbare Daten an, so ist der zulässige Bereich des Problems die leere Menge, so daß keine zulässige Lösung gefunden werden kann.
- Selbst bei linear trennbaren Problemen kann es dennoch besser sein, einige Ausnahmen zuzulassen (s.o.).
- Ansatz: man erlaubt den Datenpunkten, zur Not auch innerhalb des Margins oder sogar auf der falschen Seite der Trennebene zu liegen, solange der Abstand zur Trennebene nicht zu groß ist.

Schlupfvariablen (1)



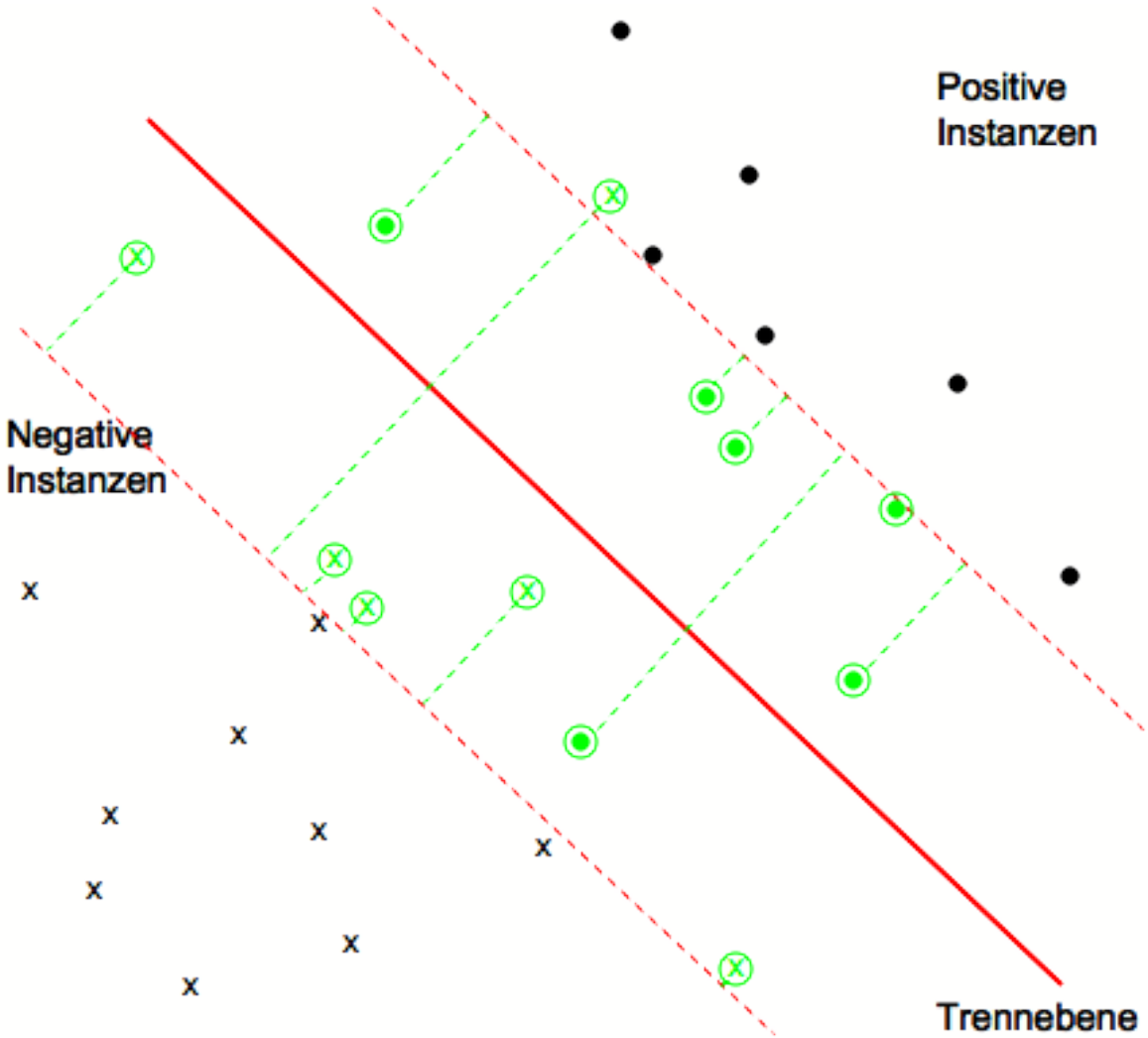
Jede Nebenbedingung erhält eine Schlupfvariable ξ_i, die angibt, wie stark die Nebenbedingung verletzt wird:

$$y_i = +1 : \langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b \geq 1 - \xi_i$$

$$y_i = -1 : \langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b \leq -1 + \xi_i$$

mit $\xi_i \geq 0$.

Schlupfvariablen (2)



Jede Nebenbedingung erhält eine Schlupfvariable ξ_i, die angibt, wie stark die Nebenbedingung verletzt wird:

$$y_i = +1 : \langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b \geq 1 - \xi_i$$

$$y_i = -1 : \langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b \leq -1 + \xi_i$$

mit $\xi_i \geq 0$.

Tradeoff: Je größer die Trennbreite, desto mehr Fehler $\Rightarrow$ erfordert sinnvollen Kompromiß.

Zielfunktion bei nicht linear trennbaren Daten

- Die Gesamtgröße der Schlupfvariablen kann man z.B. dadurch charakterisieren, dass man sie ähnlich wie beim Perzeptron einfach aufaddiert:

$$\sum_{i=1}^{\ell} \xi_i$$

  Da die $\xi_i \geq 0$ sind, können sie sich nicht gegenseitig aufheben.

- Es ist sinnvoll, $\sum_i \xi_i$ möglichst klein und gleichzeitig die Trennbreite möglichst groß zu halten.

- Zielfunktion (ist immer noch quadratisch und konvex!):

$$f(\mathbf{w}, \boldsymbol{\xi}) = \tfrac{1}{2}\langle \mathbf{w} \cdot \mathbf{w} \rangle + C \sum_i \xi_i$$

  Der Regularisierungsparameter $C > 0$ kontrolliert den relativen Einfluß beider Größen.

- Bei stark verrauschten und schlecht trennbaren Klassifikationsproblemen ist ein kleineres C tendentiell besser.

Soft Margin SVM

SVM-Optimierung (1-Norm Soft margin)

- minimiere

$$f(\mathbf{w}, \boldsymbol{\xi}) = \tfrac{1}{2}\langle \mathbf{w} \cdot \mathbf{w} \rangle + C \sum_{i=1}^{\ell} \xi_i, \quad \mathbf{w} \in \mathbb{R}^d, \boldsymbol{\xi} \in \mathbb{R}^\ell$$

- unter den 2ℓ Nebenbedingungen, $i = 1, \ldots, \ell$

$$y_i(\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b) \geq 1 - \xi_i$$

$$\xi_i \geq 0$$

Übersicht

1 Soft Margin SVM

2 Ein Optimierungsalgorithmus für die SVM

Umwandlung in eine Optimierung ohne Nebenbedingungen

Die ℓ Nebenbedingungen

$$y_i(\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b) \geq 1 - \xi_i$$

können mithilfe der Entscheidungsfunktion $f(\mathbf{x}_i) = \langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b$ der SVM umgeschrieben werden in

$$y_i f(\mathbf{x}_i) \geq 1 - \xi_i.$$

Zusammen mit $\xi_i \geq 0$ ist dies äquivalent zu

$$\xi_i \geq \max(0, 1 - y_i f(\mathbf{x}_i)).$$

Am Minimum selbst muss Gleichheit gelten, sonst könnte man die Zielfunktion durch entsprechende Wahl von ξ_i weiter vermindern.

Zielfunktion ohne Nebenbedingungen (ξ_i sind eliminiert):

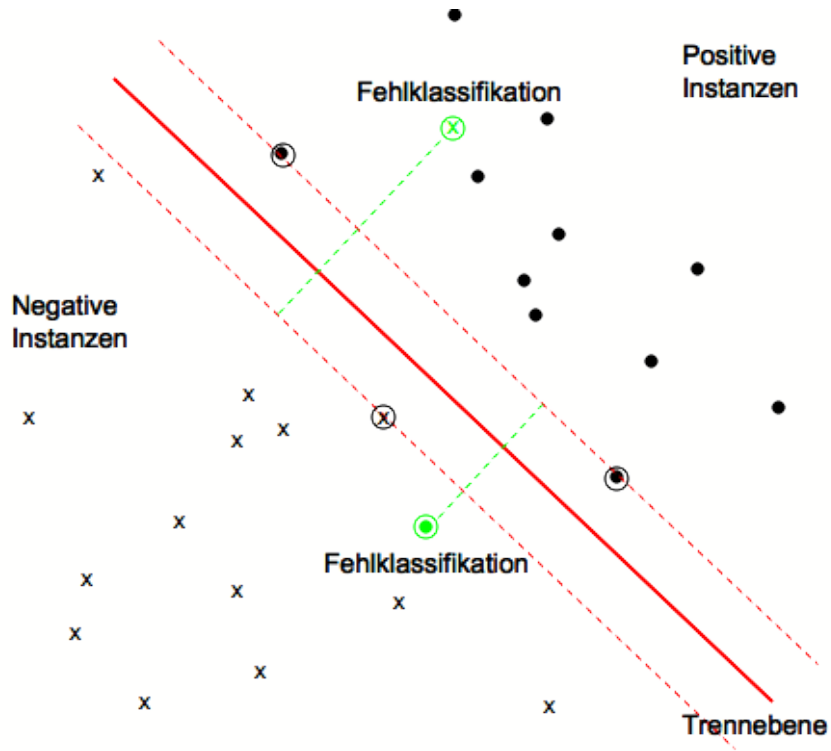
$$f(\mathbf{w}) = \tfrac{1}{2}\langle \mathbf{w} \cdot \mathbf{w} \rangle + C \sum_{i=1}^{\ell} \max(0, 1 - y_i f(\mathbf{x}_i)), \quad \mathbf{w} \in \mathbb{R}^d$$

Verlustfunktion

Der Ausdruck

$$\mathcal{L}(\mathbf{w}) = \sum_{i=1}^{\ell} \max(0, 1 - y_i f(\mathbf{x}_i))$$

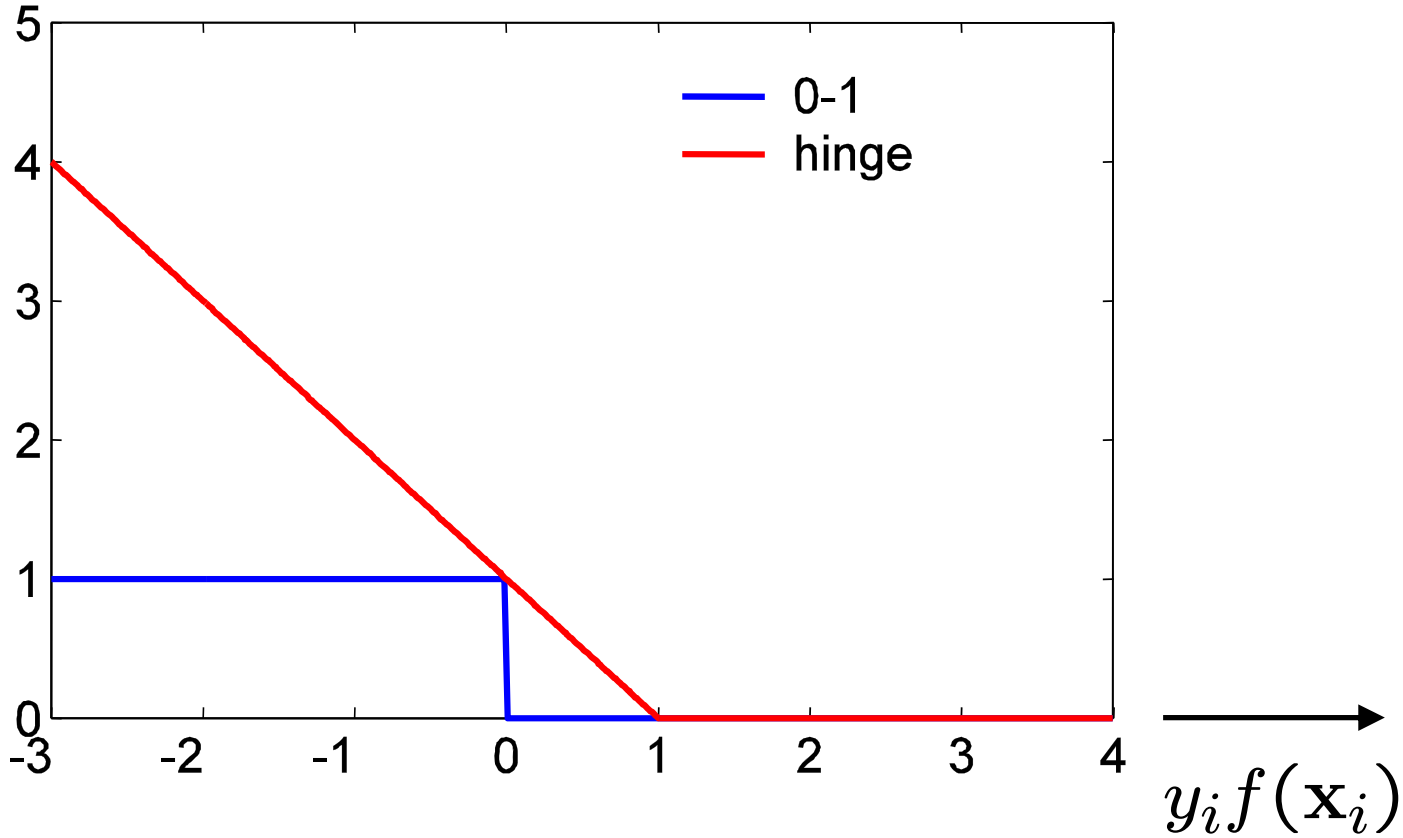
stellt eine Verlustfunktion dar.



3 Fälle:

1. $y_i f(\mathbf{x}_i) > 1$: Punkt ist außerhalb des Margins, trägt nichts zum Verlust bei.
2. $y_i f(\mathbf{x}_i) = 1$: Punkt sitzt genau auf dem Margin (Supportvektor), trägt ebenfalls nichts zum Verlust bei.
3. $y_i f(\mathbf{x}_i) < 1$: Punkt verletzt den Margin (ebenfalls Supportvektor), trägt zum Verlust bei.

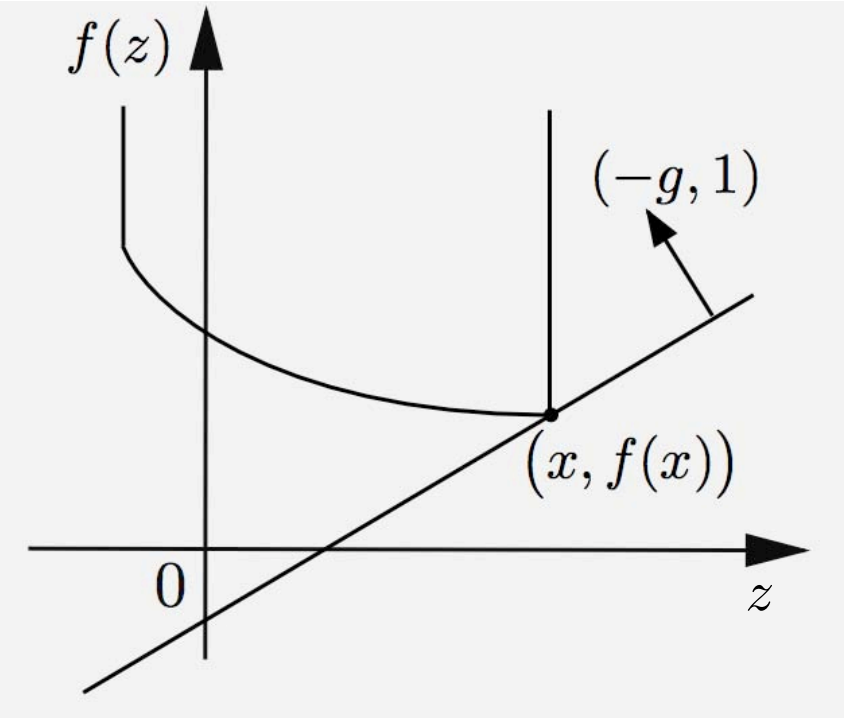
Hinge Loss



[Zisserman]

Der Hinge Loss (engl. ”Scharnierverlust”) ist eine obere Schranke an den 0-1-Verlust. Die Verlustfunktion ist immer noch konvex, aber **nicht mehr differenzierbar**!

Subgradient



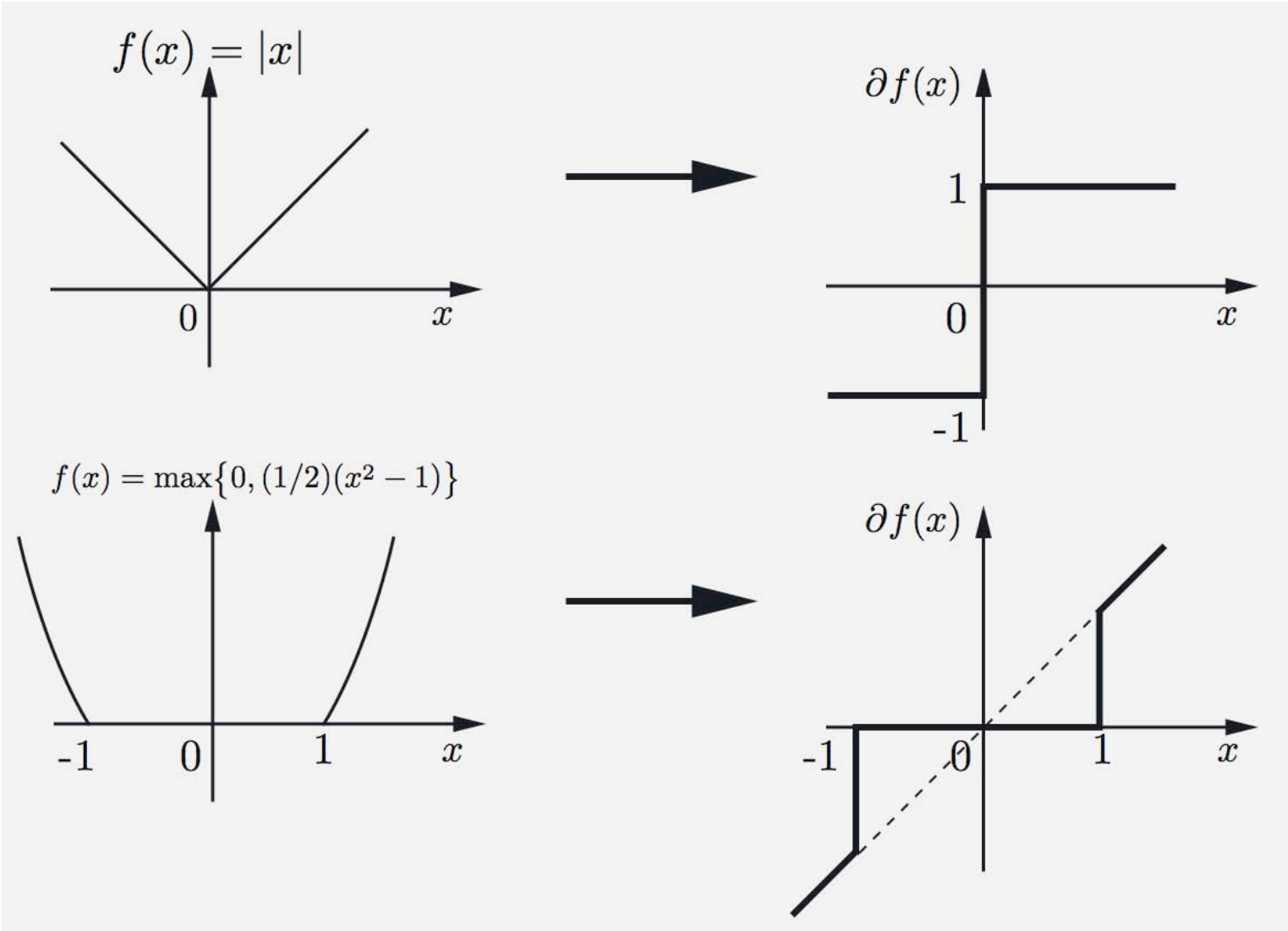
[Athena Scientific]

Subgradient einer konvexen Funktion $f(x)$ an Punkt x ist definiert als Steigung einer Geraden (oder Ebenen), die $f(x)$ in x berührt und unterhalb von $f(x)$ verläuft, d.h. es gilt

$$f(z) \geq f(x) + g^\top (z - x) \quad \forall z \in \mathbb{R}.$$

- An allen Punkten, an denen $f(x)$ differenzierbar ist, ist der Subgradient gleich der Steigung der Tangentialebene.
- An allen Punkten, an denen $f(x)$ nicht differenzierbar ist, ist der Subgradient eine Menge, nämlich die Steigung aller Tangentialebenen durch x, die unterhalb von $f(x)$ verlaufen.

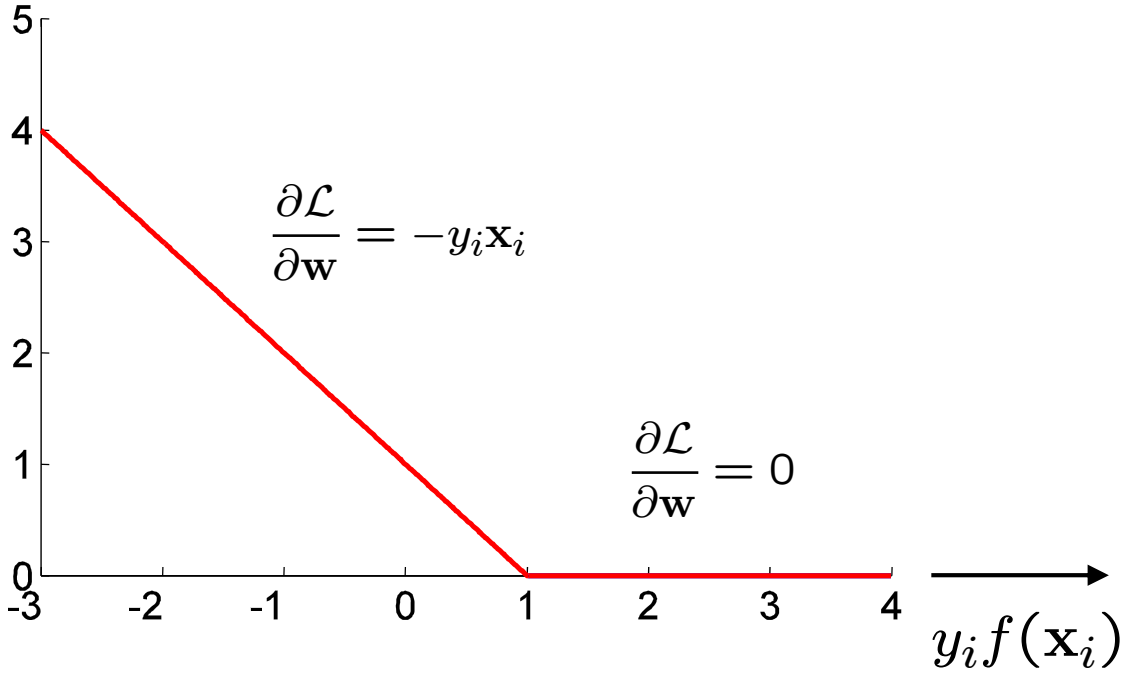
Subgradient (Beispiele)



[Athena Scientific]

Subgradient Hinge Loss

Verlustfunktion: $\mathcal{L}(\mathbf{w}) = \sum_{i=1}^{\ell} \max(0, 1 - y_i f(\mathbf{x}_i))$ mit $f(\mathbf{x}_i) = \langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b$



[Zisserman]

An Knickpunkten nimmt der Subgradient alle Zwischenwerte zwischen 0 und $-\sum_{i \text{ mit } y_i f(\mathbf{x}_i) < 1} y_i \mathbf{x}_i$ an.

Subgradientenmethode

Sei $f : \mathbb{R}^n \to \mathbb{R}$ konvex. Zur Minimierung von f nutzt die Subgradientenmethode die Iteration

$$x^{(k+1)} = x^{(k)} - \eta g^{(k)},$$

wobei $g^{(k)}$ eine **beliebiger Subgradient** von f an $x^{(k)}$ ist. Dies entspricht dem normalen Gradientenabstieg, nur laufen wir stattdessen in Richtung des negativen Subgradienten.

Im Unterschied zum normalen Gradientenabstieg ist nicht garantiert, dass die Zielfunktion bei jedem Schritt kleiner wird. Dennoch kann man beweisen, dass die Methode zum Minimum hin konvergiert, wenn die Schrittweite asymptotisch gegen 0 geht.

Subgradientenmethode für die SVM

Für die SVM wählen wir einfach 0, solange wir an einem Knickpunkt sind. Somit ergibt sich die Iteration

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta \left(\mathbf{w}_t - C \sum_{i \text{ mit } y_i f(\mathbf{x}_i) < 1} y_i \mathbf{x}_i \right)$$

Statt eines Updates über den gesamten Datensatz machen wir wie beim Perzeptron eine Einzelbeispielkorrektur (d.h. einen stochastischen (Sub-)Gradientenabstieg):

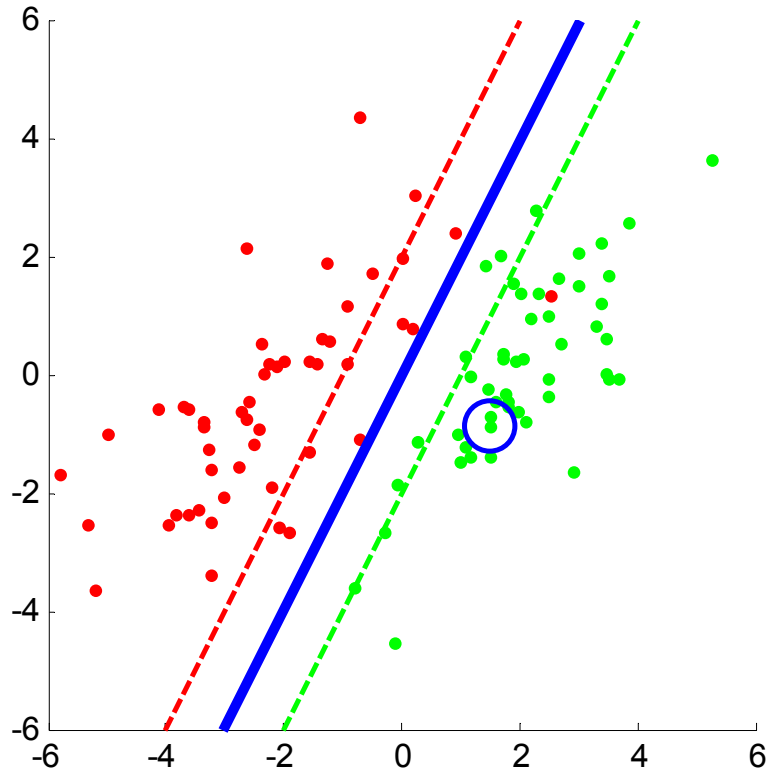
Pegasos: Primal Estimated sub-GrAdient SOlver for SVM

$$\mathbf{w}_{t+1} = \begin{cases} \mathbf{w}_t - \eta_t(\mathbf{w}_t - C y_i \mathbf{x}_i) & \text{falls } y_i f(\mathbf{x}_i) < 1 \\ \mathbf{w}_t - \eta_t \mathbf{w}_t & \text{sonst} \end{cases}$$

mit der zeitabhängigen Lernrate $\eta_t = 1/t$.

Beispiel Pegasos





[Zisserman]