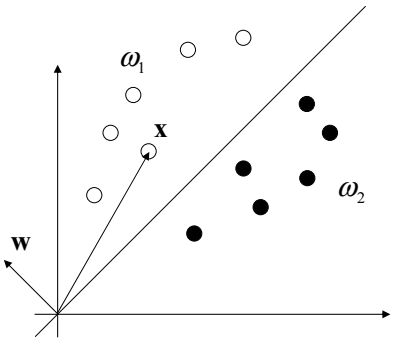
Linear separierbare Klassen – der Perceptron-Algorithmus

Ziel ist, ein Algorithmus zur *automatischen* Anpassung der unbekannten Gewichtsmatrizen $\mathbf{W}^i$ eines Perceptrons an ein Klassifikationsproblem, gegeben eine endliche Stichprobe $\{(\mathbf{x}_i,\omega_k)\}$.
Für den Fall linear separierbarer Klassen gab Rosenblatt einen iterativen Algorithmus zur Lösung dieses Problems an. Es wird angenommen, dass eine lineare Trennfläche, definiert durch die Gleichung $\mathbf{w}^{*T}\mathbf{x}=0$ für das Zweiklassenproblem existiert, derart dass:

$$\mathbf{w}^{*T}\,\mathbf{x} > 0 \quad \forall \mathbf{x} \in \omega_1$$
$$\mathbf{w}^{*T}\,\mathbf{x} < 0 \quad \forall \mathbf{x} \in \omega_2$$

Dies schließt auch den Fall ein, dass die Trennfläche nicht wie hier durch den Ursprung läuft, nämlich $\mathbf{w}^{*T}\mathbf{x}+b^*=0$, da dieser durch Erweiterung von $\mathbf{x}$ auf die obige Formulierung zurückgeführt werden kann:

$$\mathbf{x}'^T = \left[1, \mathbf{x}^T\right] \quad \mathbf{w}'^{*T} = \left[b^*, \mathbf{w}^{*T}\right]$$
$$\Rightarrow \mathbf{w}'^{*T}\mathbf{x}' = \mathbf{w}^{*T}\mathbf{x} + b^*$$



Linear separierbares Problem

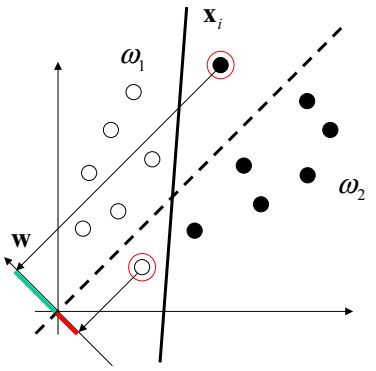
Gelöst wird das Problem durch die Wahl eines geeigneten Gütekriteriums und die Angabe eines Optimierungsalgorithmus. Das zu minimierende Perceptron-Gütemaß sei gegeben durch:

$$J(\mathbf{w}) = \sum_{\mathbf{x}\in\mathbb{Y}} (\delta_x \mathbf{w}^T\mathbf{x}) = \text{—} + \text{—}$$

mit: $\delta_x = +1$ für $\mathbf{x} \in \omega_1$

und: $\delta_x = -1$ für $\mathbf{x} \in \omega_2$

wobei $\mathbb{Y}$ die Menge der Trainingsvektoren enthält, welche durch die gegebene Hyperfläche *falsch* klassifiziert werden.
Offensichtlich ist obige Summe immer positiv und sie wird Null, wenn $\mathbb{Y}$ die leere Menge ist, d.h. alle Elemente *richtig* zugeordnet werden. Nämlich für den Fall eines falsch zugeordneten Vektors $\mathbf{x}\in\omega_1$ erhalten wir mit $\mathbf{w}^T\mathbf{x} > 0$ und $\delta_x > 0$ ein positives Produkt der beiden Terme. Das gleiche gilt für Vektoren der Klasse ω_2 . Das Gütekriterium nimmt seinen Minimalwert Null an, falls alle Stichproben richtig klassifiziert werden.



Das Gütemaß ist *stetig und stückweise linear*. Falls wir den Gewichtsvektor wenig verändern, ändert sich $J(\mathbf{w})$ linear bis zu dem Punkt, wo sich die Zahl der falsch zugeordneten Vektoren ändert. An dieser Stelle ist der Gradient von J nicht definiert und der Gradient von J ist unstetig.
Die iterative Minimierung der Kostenfunktion geschieht durch einen Algorithmus, welcher dem Gradientenabstieg ähnlich ist:

$$\mathbf{w}_{i+1} = \mathbf{w}_i - \rho_i \left. \frac{\partial J(\mathbf{w})}{\partial \mathbf{w}} \right|_{\mathbf{w}=\mathbf{w_i}}$$

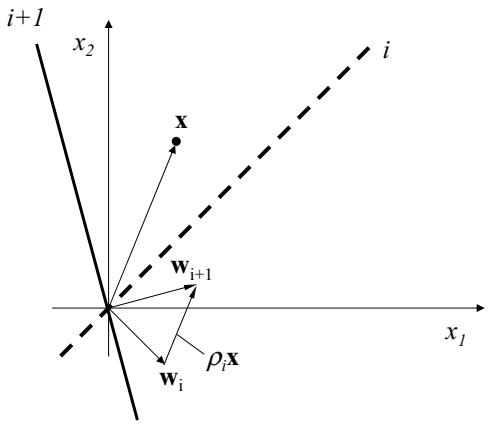
Dabei ist $\mathbf{w}_i$ der Gewichtsvektor für die i-te Iteration und ρ_i eine Sequenz von positiven Zahlen.
Vorsicht ist geboten an den Unstetigkeitsstellen! Aus dem Gütekriterium ergibt sich:

$$\frac{\partial J(\mathbf{w})}{\partial \mathbf{w}} = \sum_{\mathbf{x}\in\mathbb{Y}} \delta_x \mathbf{x} \quad \text{wegen:} \quad \frac{\partial \langle \mathbf{w}, \mathbf{x} \rangle}{\partial \mathbf{w}} = \mathbf{x}$$

und damit:

$$\boxed{\mathbf{w}_{i+1} = \mathbf{w}_i - \rho_i \sum_{\mathbf{x}\in\mathbb{Y}} \delta_x \mathbf{x}}$$ Perceptron-Algorithmus

Die Iteration ist fortzusetzen bis der Algorithmus konvergiert.
Das folgende Bild vermittelt einen Eindruck der Funktionsweise. Es wird angenommen, dass beim Iterationsschritt *i* nur ein **x** falsch klassifiziert wird ($\mathbf{w}^T\mathbf{x} > 0$) und es sei ρ_i=0,5. Der Perceptron-Algorithmus korrigiert den Gewichtsvektor *in die Richtung von* **x**. Damit dreht sich die Trennlinie und **x** wird korrekt zugeordnet ($\mathbf{w}^T\mathbf{x} > 0$) .



In dem folgenden Beispiel wird angenommen, dass wir beim vorletzten Iterationsschritt angekommen sind und lediglich noch zwei Merkmalsvektoren falsch zugeordnet werden:

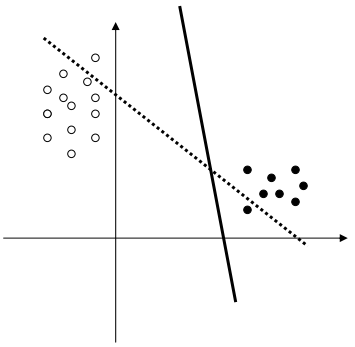
Trennlinie: $x_1 + x_2 - 0,5 = 0 \Rightarrow$ Gewichtsvektor: $\mathbf{w}^T = \begin{bmatrix} 1 & 1 & -0,5 \end{bmatrix}^T$

Falsch klassifizierte Vektoren: $\begin{bmatrix} 0,4 & 0,05 \end{bmatrix}^T$ $\begin{bmatrix} -0,2 & 0,75 \end{bmatrix}^T$

Mit dem Perceptron-Algorithmus ergibt sich der nächste Gewichtsvektor mit ρ_i=0,5 zu:

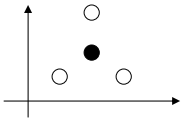
$$\mathbf{w}_{i+1} = \mathbf{w}_i - 0,7(-1)\begin{bmatrix} 0,4 \\ 0,05 \\ 1 \end{bmatrix} - 0,7(+1)\begin{bmatrix} -0,2 \\ 0,75 \\ 1 \end{bmatrix} = \begin{bmatrix} 1,42 \\ 0,51 \\ -0,5 \end{bmatrix}$$

Die neue Trennlinie: $1,42x_1 + 0,51x_2 - 0,5 = 0$ trennt alle Vektoren korrekt. Der Algorithmus terminiert.



Eigenschaften des Perceptron-Algorithmus

- Man kann beweisen, dass der Algorithmus in einer endlichen Anzahl von Iterationen zu einer Lösung konvergiert (bei geeignet gewählten Werten von ρ_i, was bei Gradientenalgorithmen immer ein Problem ist), falls die Voraussetzung erfüllt ist, dass die Klassen linear trennbar sind.
- Die gefundene Lösung ist allerdings nicht eindeutig.
- Für nicht linear-separierbare Probleme erhält man keine Lösung. Die Korrektur in der Iteration hat immer von Null verschiedene Beiträge und kann somit nicht terminieren.



- Später werden wir Multilagen-NN kennenlernen, welche beliebige Klassifikationsprobleme (auch nichtlinear
Backpropagation-Algorithmus kann verw
Netzwerke zu trainieren.

Demo aus Matlab: NN-Toolbox

-Perceptron Learning rule

-Linearly non-separable vectors