

Fourier-Transformation, Fourier-Reihen

► Fourier-Reihenzerlegung

→ periodisches Signal → Fourier Reihenzerlegung in Koeffizienten

$$C_k = \frac{1}{2L} \int_{-L}^{+L} f(x) e^{-i\omega_0 k x} dx$$

← "Gewicht"

Komplexe Fourierreihe:

$$f(x) = \sum_{k=-\infty}^{+\infty} C_k e^{ikx}$$

← mit C_k

$$e^{ix} = \cos(x) + i \sin(x)$$

Reelle Fourierreihe:

$$f(x) = \frac{a_0}{2} + \sum_{k=1}^{+\infty} (a_k \cos(\frac{2\pi}{T} k x) + b_k \sin(\frac{2\pi}{T} k x))$$

$$a_k = \frac{1}{L} \int_{-L}^{+L} f(x) \cos(kx) dx$$

$$b_k = \frac{1}{L} \int_{-L}^{+L} f(x) \sin(kx) dx$$

mit
 a_k, b_k

► Fourier-Transformation ω -Domäne \rightarrow t-Domäne

$$F(\omega) = \int_{-\infty}^{\infty} f(t) e^{-i\omega t} dt$$

Frequenz
↳ inverse FT

$$f(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} F(\omega) e^{i\omega t} d\omega$$

↳ Betrag Fourier-Spektrum

$$|F(\omega)| = \sqrt{R(F(\omega))^2 + I(F(\omega))^2}$$

↳ Phase des Spektrums

$$\varphi(F(\omega)) = \arctan \frac{I(F(\omega))}{R(F(\omega))}$$

↳ Reellanteil
↳ Imaginär-
anteil

► Eigenschaften FT

↳ Linearität

$$F(c_1 f_1 + c_2 f_2) = c_1 F(f_1) + c_2 F(f_2)$$

↳ Differentiation

$$F(f^{(n)}) = (i\omega)^n F(f)$$

↳ Verschiebung

$$F(f(t-T)) = e^{-i\omega T} F(f) \quad \text{um Zeit}$$

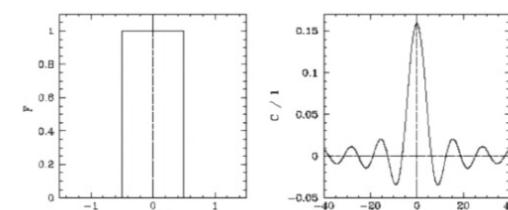
$$F(e^{-i\omega_0 t} f(t)) = F(\omega - \omega_0) \quad \text{um Frequenz}$$

► typische FT

$$a \sin(\omega_0 x) \Leftrightarrow F(\omega) = i a \pi \delta(\omega - \omega_0) - i a \pi \delta(\omega + \omega_0)$$

$$a \cos(\omega_0 x) \Leftrightarrow F(\omega) = a \pi \delta(\omega - \omega_0) + a \pi \delta(\omega + \omega_0)$$

$$\Leftrightarrow \text{sqr} \Leftrightarrow \text{sinc}$$



$$\Leftrightarrow f(x) = \sum_{v=-\infty}^{\infty} \delta(x - vT) \Leftrightarrow F(\omega) = \frac{2\pi}{T} \sum_{v=-\infty}^{+\infty} \delta(\omega - \frac{v2\pi}{T})$$

Grund
für
Aliasing
(nächste Seite)

► Discrete FT

→ E: diskret → Fenster ausschneiden

→ A: diskret

Faltung

► kontinuierlich:

$$(f * g)(t) = \int_{-\infty}^{\infty} f(t-\tau) g(\tau) d\tau$$

► diskret

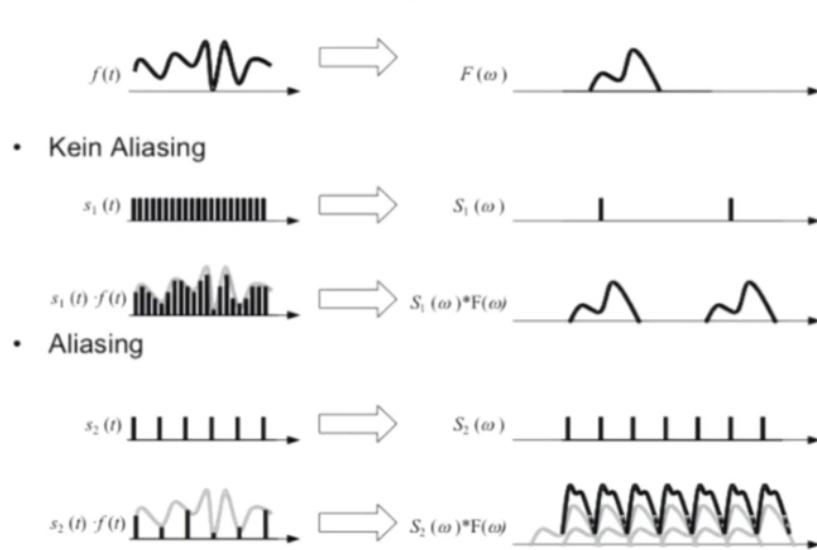
$$(f * g)[i] = \sum_{j=-\infty}^{\infty} f[i-j] g[j]$$

Wie bekommt man die charakteristische Funktion eines Systems?

-> Signal mit Impuls falten (oder FTransformierte multiplizieren)

Aliasing

Aliasing



→ schnell genug abtasten! // $\Delta x < \frac{1}{2\omega}$

→ Signal muss bandbegrenzt sein → z.B. Tiefpassfilter

→ Rekonstruktion $F \rightarrow f$ nur möglich, wenn

Abtasttheorem eingehalten!

► Faltung:

$$F(f_1(t) * f_2(t)) = F_1(\omega) * F_2(\omega)$$

► Multiplikation:

$$F(f_1(t) \cdot f_2(t)) = F_1(\omega) * F_2(\omega)$$

! Filtern: auf der spektralen Seite: Multiplikation
auf der Eingabeseite: Falten

Korrelation

► 1d-Kreuzkorrelation

$$R_{fg}(m) = \sum_i f(i)g(i+m)$$

Unterschied zu Faltung: Vorzeichen.
aus den Folien; für diskrete Funktionen
im Internet habe ich die Formel mit „+“ gefunden, auf den Folien steht sie mit „-“
mit „-“ ist das eig. die Formel für Faltung und nicht für Korrelation?

► 2d-Korrelation

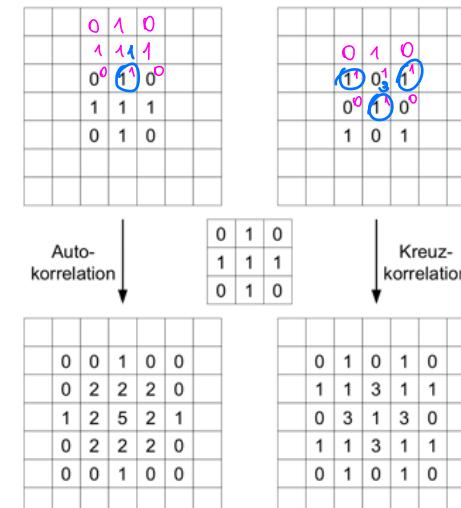
$$R_{fg}(m,n) = \sum_i \sum_j f(i,j) g(i+m, j+n)$$

► Autokorrelation

► Anwendung

↳ Autokorrelation: Bestimmung einer Grundfrequenz

↳ 2d Auto & Kreuzkorrelation: Kantenentdeckung
(heutzutage „Neuronale Netze“)



Pattern Matching

Ziel: Muster erkennen anhand von einem gespeicherten Beispiel

Maß Übereinstimmung: $M_{fig}(m,n) = \sum_i \sum_j |f(i,j)g(i+m, j+n)|$

Abstand: $E_{fig}(m,n) = (R_{fg}(m,n))^2$

⊖ teuer → Schablonengenerierung aufwändig

→ Model bauen → generalisieren

Klassifikation

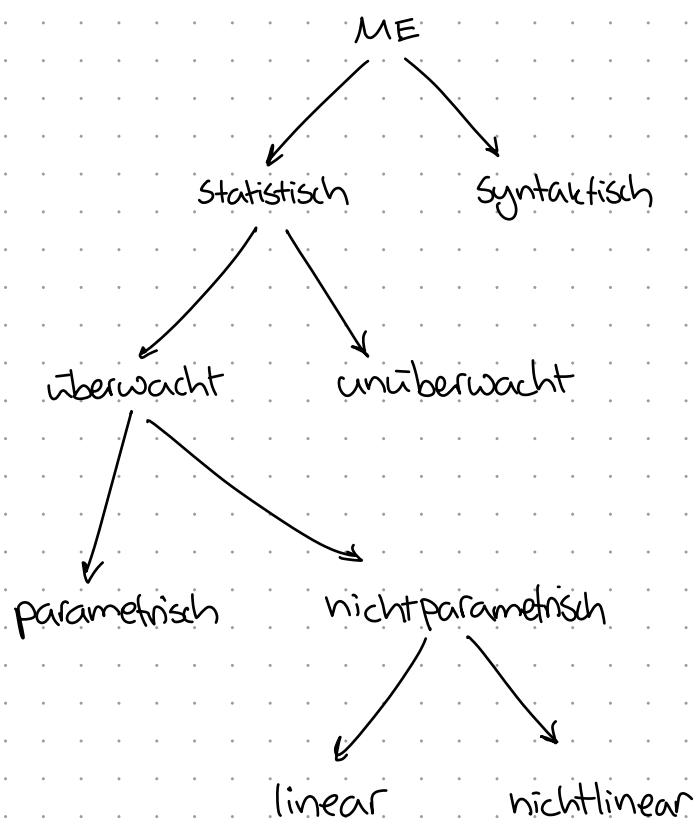
- Einordnen in die Welt:

- gesellschaftlich: Buchstaben, menschliche Artefakte, Kredite
- biologisch: Katze, Hund ... natürlich

→ Regeln: sehr komplex → automatisch lernen

Mustererkennung

► Klassifikation



↳ überwacht: Klasse bekannt für jeden Sample der Trainingsdaten
nicht gelabte Daten

→ Vorgehensweise:

- Merkmale: bekannt
- Klassen: gesucht

↳ nicht überwacht: Klasse muss automatisch erlernt werden

→ Vorgehensweise:

- ① Strukturen finden
- ② Clustern → wie finde ich sie?
→ wie viele gibt es?

↳ überwacht, parametrisiert

- Wahrscheinlichkeitsverteilung bestimmen
 - ↳ z.B. Gaussverteilung
- Parameter der Verteilung bestimmen
 - ↑ Bayes

↳ überwacht, nicht parametrisiert

- von keiner Verteilung ausgehen
 - ↑ kNN, Perzeptron

Bayes' Entscheidungstheorie

► Bayes' Regel

$$P(\omega_i | x) = \frac{P(x | \omega_i) P(\omega_i)}{P(x)}$$

Klasse Bedachtung (Vektor)

(E16)?

Verteilung der Daten im gesamten Datensatz

a-posterior W-keit

geg. w.

a-priori w. ohne Bedachtung

$$P(x) = \sum_j p(x | \omega_j) P(\omega_j)$$

* Entscheidungsregionen

↳ Entscheidung für eine Klasse:

Vgl:	$P(\omega_1 x)$	mit	$P(\omega_2 x)$
	$p(x \omega_1) P(\omega_1)$	mit	$p(x \omega_2) P(\omega_2)$
mehrere Kategorien:			
	$p(x \omega_j) P(\omega_j)$	mit	$p(x \omega_j) P(\omega_j); j \neq i$

↳ Diskriminierungsfunktion

$$g_i = P(\omega_i | x), \text{ aber auch}$$

$$g_i = p(x | \omega_i) P(\omega_i)$$

$$g_i = \log(p(x | \omega_i)) + \log(P(\omega_i))$$

⊕ Score anstatt W-keit

⊕ Kann mit sehr kleinen Werten besser umgehen

Gaussverteilung (parametrischer Klassifikator) ⊖ oft anders verteilt → nicht param. Ansatz

Univariante Verteilung:

$$p(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right]$$

$N(\mu, \sigma^2)$

(F31)?

Multivariate Verteilung:

$$p(\vec{x}) = \frac{1}{(2\pi)^{d/2} |\Sigma|^{1/2}} \exp\left[-\frac{1}{2} (\vec{x} - \vec{\mu})^t \Sigma^{-1} (\vec{x} - \vec{\mu})\right]$$

$N(\mu, \Sigma)$

Mehalanobis-Distanz → $g_i(\vec{x}) = \frac{1}{2} (\vec{x} - \vec{\mu}_i)^t \Sigma_i^{-1} (\vec{x} - \vec{\mu}_i) - \frac{1}{2} \log(2\pi) - \frac{1}{2} \log |\Sigma_i| + \log P(\omega_i)$

► Maximum Likelihood Estimation

↳ Multivariate Verteilung:

$$\mu_i = \frac{1}{N} \sum_{k=1}^N \vec{x}_k$$

$$\Sigma = \frac{1}{N} \sum_{k=1}^N (\vec{x}_k - \vec{\mu}_i)(\vec{x}_k - \vec{\mu}_i)^T$$

→ transponiert
→ Ergebnis ist eine Matrix

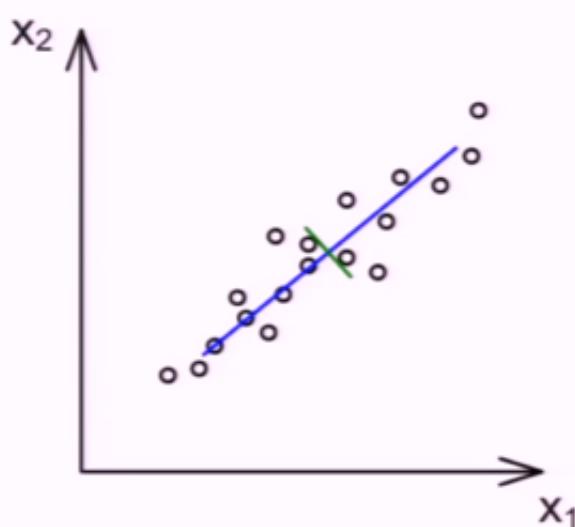
Curse of Dimensionality

→ mehr Features, selbe Menge an Daten: schlechtere Performance

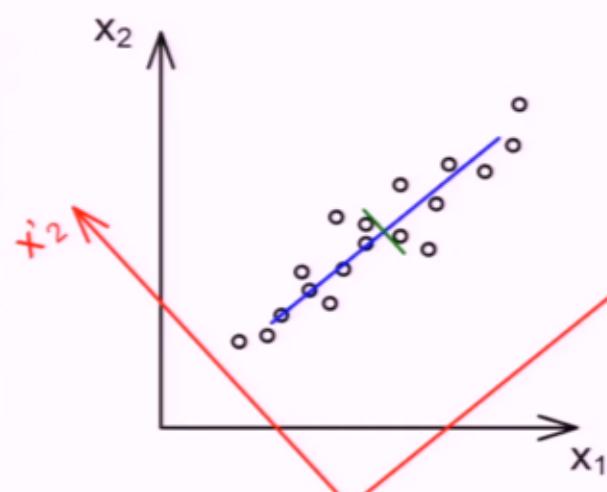
► Principal Component Analysis

⊕ (meistens) Verbesserung der Ergebnisse

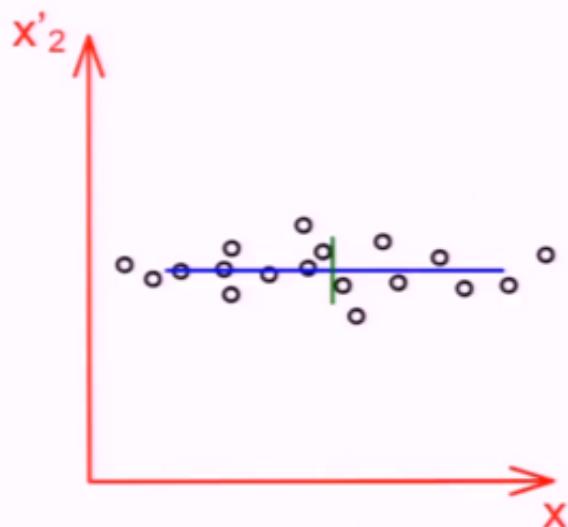
⊖ Immerhin Informationsverlust, der sich auf die Ergebnisse auswirken kann



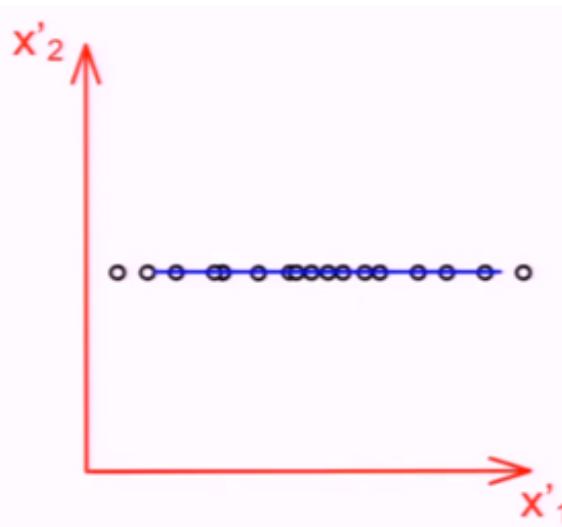
Find the axis along the highest variance



Rotate the space along the axis



Dimensions are uncorrelated now



Remove dimensions with low variance
=> Reduction of dimensionality with minimum loss of information

Risk Analysis

→ Kosten jeder Entscheidung berechnen

Zustände der Natur: $\Omega = \{\omega_1, \omega_2, \dots\}$

Mögliche Aktionen: $A = \{a_1, \dots, a_s\}$

⇒ Verlustfunktion

$$\gamma(a_i | \omega_j) = \gamma_{ij}$$

↳ erwarteter Verlust:

$$R(a_i | \vec{x}) = \sum_{j=1}^s \gamma(a_i | \omega_j) P(\omega_j | \vec{x})$$

↳ Entscheidung für eine Klasse

- ω_1 , wenn: $R(a_1 | \vec{x}) < R(a_2 | \vec{x})$

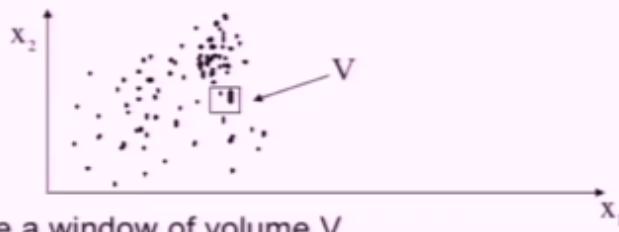
- wenn: $(\gamma_{21} - \gamma_{11}) P(\omega_2 | \vec{x}) > (\gamma_{12} - \gamma_{22}) P(\omega_1 | \vec{x})$

- wenn: $(\gamma_{21} - \gamma_{11}) P(\vec{x} | \omega_1) P(\omega_1) > (\gamma_{12} - \gamma_{22}) P(\vec{x} | \omega_2) P(\omega_2)$

$$\text{, wenn: } \frac{P(\vec{x} | \omega_1)}{P(\vec{x} | \omega_2)} > \frac{\gamma_{12} - \gamma_{22}}{\gamma_{21} - \gamma_{11}} \cdot \frac{P(\omega_2)}{P(\omega_1)}$$

Parzen Windows (nicht parametrischer Klassifikator)

- No Assumptions about the distribution are made
estimate $p(x)$ directly from data



- Choose a window of volume V
- Count the number of samples that fall inside the window.

$$p(x) \approx \frac{k/n}{V} \quad \begin{array}{l} \text{: k = count} \\ \text{: n = number of samples} \end{array}$$

→ Histogramm

⇒ zu große Kästchen: Auflösung gering

⇒ zu kleine Kästchen: ungünstige Schätzung

→ Parzen(x) $\rightarrow p(x)$, wenn: $\lim_{n \rightarrow \infty} V_n = 0$

$$\lim_{n \rightarrow \infty} k_n = \infty$$

$$\lim_{n \rightarrow \infty} k_n/n = 0$$

günstig, wenn man unendlich viele Daten hat

K-Nearest Neighbors

① k nearest neighbors finden

② Klasse der am öftesten auftretenden Beispiele herausfinden (Vote)

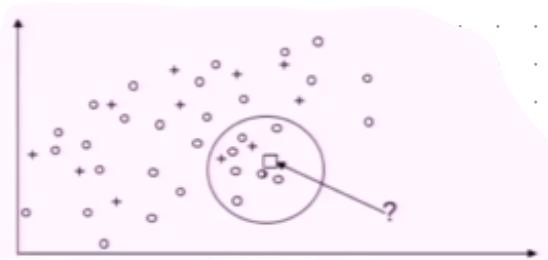
③ x dieser Klasse zuweisen

*größere k: zuverlässiges Ergebnis

*kleinere k: sicherstellen, dass andere Ergebnisse in der euklidischen Nähe liegen

→ lieber mehr Daten

$$\begin{aligned} k &= 9 \\ 7 &\circ \\ 2 &+ \\ \Rightarrow &\text{classify o} \end{aligned}$$



Decision Function

→ „Trennlinie(-oberfläche) im Raum“

► **Diskriminierungsfunktion:** entscheidet sich für eine bestimmte Klasse

- $g(\vec{x}) > 0 \Rightarrow$ Class A
- $g(\vec{x}) < 0 \Rightarrow$ Not class A
- $g(\vec{x}) = 0 \Rightarrow$ No decision

$$g(\vec{x}) = \sum_{i=1}^n w_i x_i + w_0 = \vec{w}^T \vec{x} + w_0$$

← Linearkombination der Merkmale

wird klassifiziert

- $\vec{x} = (x_1, \dots, x_n)^T$ Feature vector
- $\vec{w} = (w_1, \dots, w_n)^T$ Weight vector
- w_0 Threshold weight

→ kompensiert, wenn Trennlinie nicht durch 0-Pkt. geht

Linearkombination: Vektor, der sich durch gegebene Vektoren unter Verwendung der Vektoraddition und der skalaren Multiplikation ausdrücken lässt

↳ Schätzung der Diskriminierungsfunktion

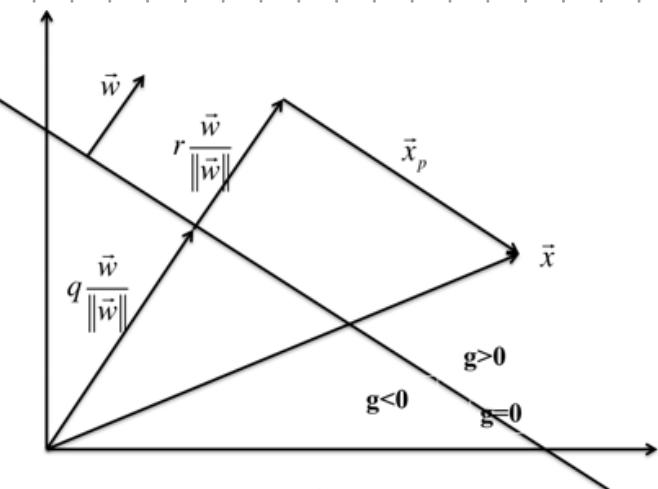
→ anhand von Datenklassen Gewichte schätzen

→ w_0 „reinziehen“ um es uns einfacher zu machen: $x_0 = 1$

Discriminant function:

$$g(x) = w_0 + \sum_{i=1}^n w_i x_i = \sum_{i=0}^n w_i x_i; \quad x_0 = 1$$

→ Diskriminierungsfunktion im Vektorraum:



$$g(\vec{x}) = \vec{w}^T q \frac{\vec{w}}{\|\vec{w}\|} + \vec{w}^T r \frac{\vec{w}}{\|\vec{w}\|} + \vec{w}^T \vec{x}_p + w_0 = -w_0 + r \|\vec{w}\| + w_0$$

$$\text{On Hyperplane H: } g(\vec{x}) = \sum_{i=1}^n w_i x_i + w_0 = \vec{w}^T \vec{x} + w_0 = 0$$

In geometry a **hyperplane** is a subspace of one dimension less than its ambient space. If a space is 3-dimensional then its hyperplanes are the 2-dimensional planes, while if the space is 2-dimensional, its hyperplanes are the 1-dimensional lines.

- aus geometrischer Sicht: Messung der Distanz zur Hyperebene

↳ Fisher Linear Discriminant Analysis: Projektion von mehrdimensionalen Pkt. auf die Linie

$$\begin{aligned} y &= \vec{w}^T \vec{x} \\ \vec{w} &= S_w^{-1} (\vec{m}_1 - \vec{m}_2) \\ S_w &= S_1 + S_2 \\ S_i &= \sum_{x \in X_i} (x - \vec{m}_i)(x - \vec{m}_i)^T \end{aligned}$$

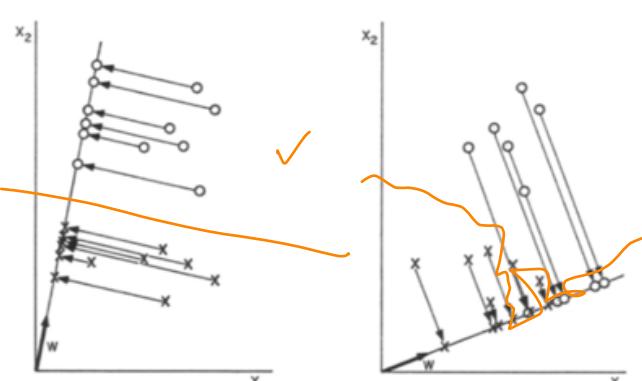
analog Kovarianzmatrizen; sa. MLE Gaus multivariat Varianz (S4ZF)

→ Unterschiedliche Projektion der Punkte auf einer Dimension

↳ Punkte auf unterschiedliche Linien projizieren
↳ ... bis die Punkte voneinander trennbar werden

$$g(x) = \frac{|\vec{m}_1 - \vec{m}_2|}{\sqrt{S_1 + S_2}}$$

← Mittelwert beider Klassen möglichst weit voneinander entfernt
Streuung innerhalb einer Klasse möglichst klein



Goal: find an orientation of the line, were the projected samples are well separated.

$$\vec{m}_i = \frac{1}{n_i} \sum_{y \in Y_i} y$$

Mittelwert einer Klasse

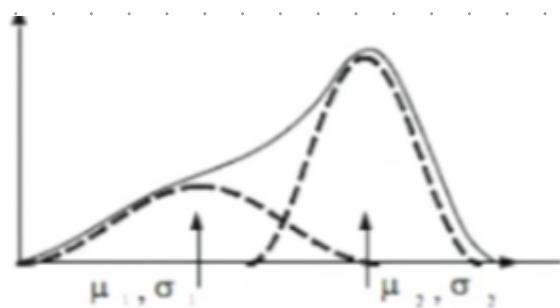
$$\tilde{S}_i = \sum_{y \in Y_i} (y - \vec{m}_i)^2$$

Wie verstreut sind die Daten innerhalb von einer Klasse voneinander?

Unüberwachte Klassifikation

► Mixture Densities Annahme Samples kommen aus mehreren Klassen; k sind gaussverteilt

↳ k-Means-Verfahren / top-down



- 1) initiale μ_1, \dots, μ_n bestimmen / zufällig/
 - 2) Daten klassifizieren (anhand von nächstem Mean)
 - 3) Daten werden zu den neuen Mittelpunkten zugeordnet: neuen Mittelpunkt aus Datum & Mittelpunkt berechnen
 - 4) haben sich die Mittelpunkte geändert?
- $j = 1$
n: stop ✓

(!) am wichtigsten bei k-Means: Wahl der Anzahl der Klassen

Evaluation der Wahl von k : Distortion: Streuung der Daten innerhalb einer Klasse

↳ Hierarchical Clustering / bottom-up

- 1) alle Beispiele: eigenes Cluster
- 2) Paar mit kleinsten Distanz finden
- 3) Paar clustern
- 4) Wiederhole von 2), es sei denn threshold Cstop ist erreicht

Perzeptron

- Gewichte
- Merkmale } Summe + Threshold

Annahmen/Vereinfachungen

↳ Threshold in der Summe

$$g(x) = \omega_0 + \sum_{i=1}^n \omega_i x_i \rightarrow g(x) = \sum_{i=1}^n \omega_i x_i ; x_0 = 1$$

- einfacher
- Linearkombination geht somit durch den Nullpunkt

↳ Beispiele aus ω_2 werden gespiegelt

$$\vec{x} \rightarrow -\vec{x}$$

$\Rightarrow \vec{\omega} \cdot \vec{x} > 0$, wenn korrekt klassifiziert

Perceptron Criterion Function

$$J_p(\vec{\omega}) = \sum_{\vec{x} \in \text{ex}} (-\vec{\omega} \cdot \vec{x}) / \text{größer, je mehr Fehler gemacht wurden}$$

↳ Gradienten von J_p nach $\vec{\omega}$ berechnen (Ableitung)

$$\nabla J_p = \sum_{\vec{x} \in \text{ex}} (-\vec{x}) / \text{Summe der Distanzen aller falsch klassifizierten Beispiele}$$

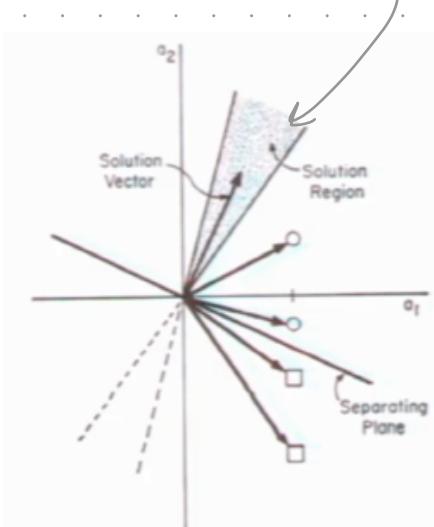
\Rightarrow Gewichte müssen in Richtung von diesem Gradienten geändert werden

Gradient descent:

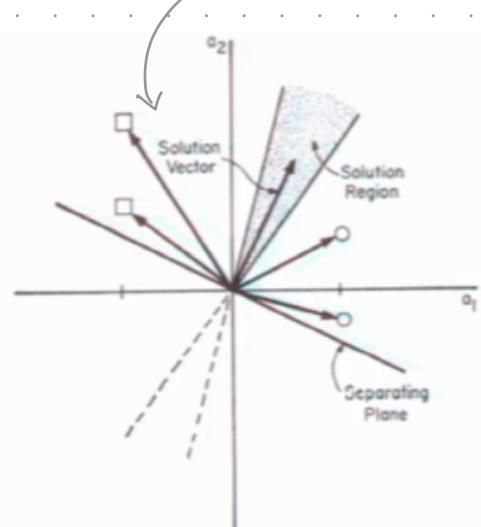
$$\vec{\omega}_{k+1} = \vec{\omega}_k + S_k \sum_{\vec{x} \in \text{ex}} \vec{x}$$

neues, besseres Gewicht
falsch klassifizierte Tokens (Δ_{j_p})
Skalierfaktor ("learning rate")

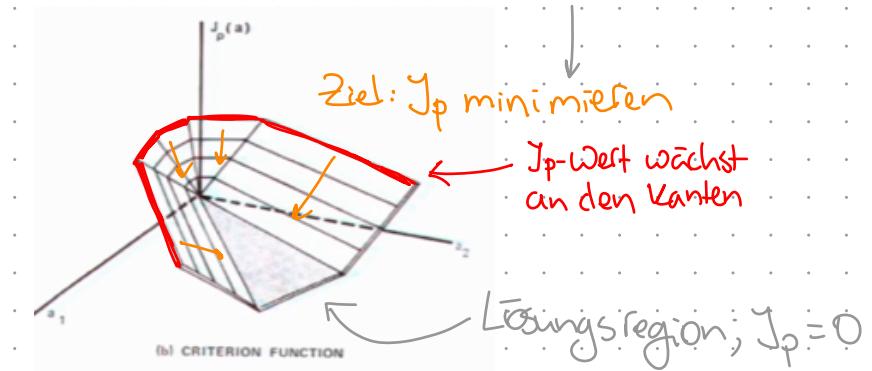
"Spielraum" für den Lösungsvector



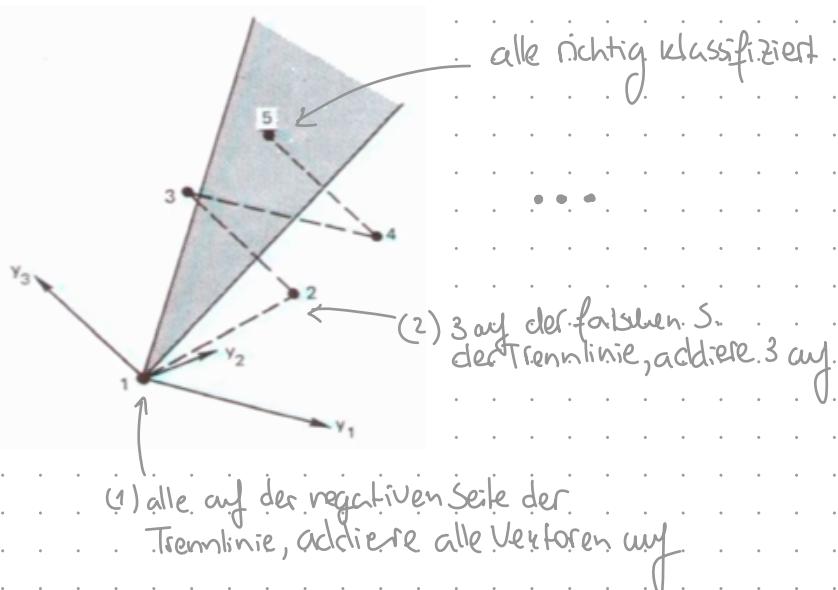
Normalisierte Vektoren
der zweiten Klasse



Grundlage für die
Perzepton-Lernregel



↳ Lösungsraum anhand von Gradient Descent finden



- Varianten

- Relaxation

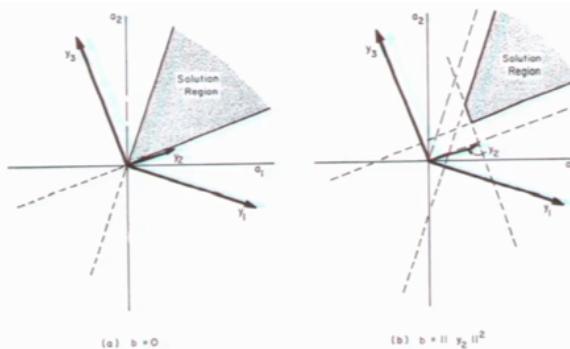
$$J_q(\vec{w}) = \sum_{\vec{x} \in X} (\vec{w} \cdot \vec{x})^2$$

quadriert => glattere Oberflächen → höhere Bestrafung, wenn zu weit weg von der Lösung
→ Oberfläche ist glatter

- Margin

$$J(\vec{w}) = \frac{1}{2} \sum \frac{(\vec{x} \cdot \vec{w} - b)^2}{\|\vec{x}\|^2}$$

Margin



- Min Square Error (MSE)

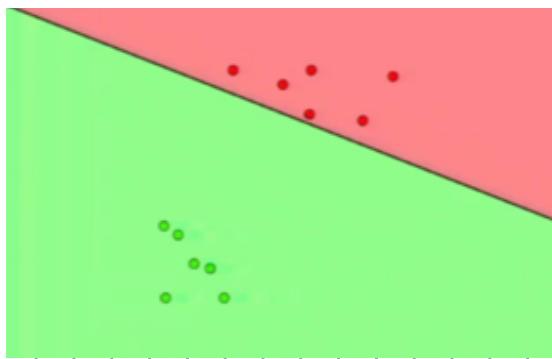
$$J_b(\vec{w}) = \sum_{\vec{x} \in X} (\vec{w} \cdot \vec{x}_j - b)^2$$

Margin

- eignet sich auch für nichtseparierbare Daten

-Probleme

→ Trennlinie nicht immer optimal



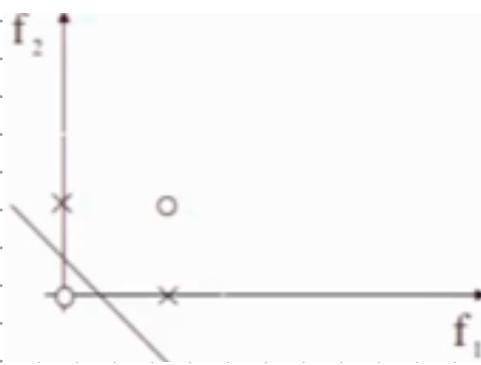
→ Linear nicht separierbare Daten



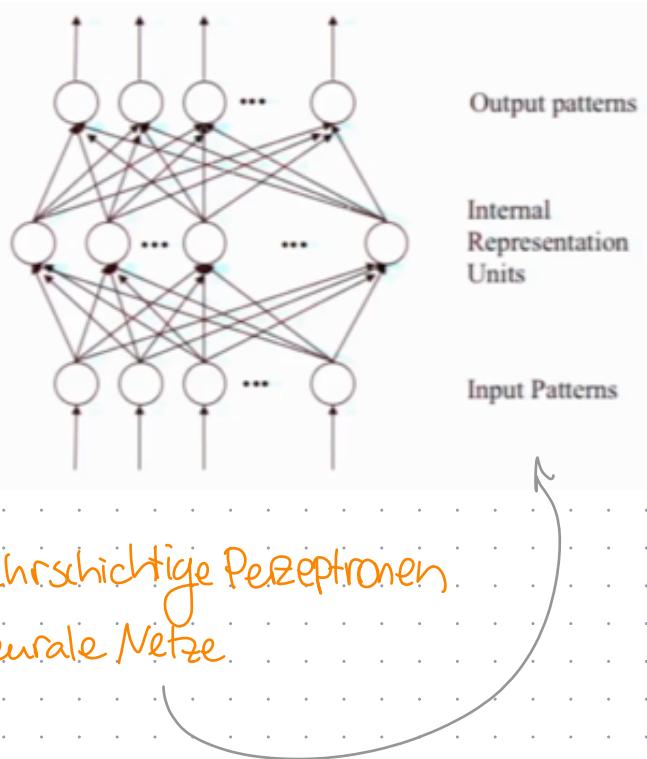
- Lernrate kleiner machen → bessere Ergebnisse

Algorithmus terminiert nicht

→ XOR-Problem nicht erlernbar



Lösungen nicht linear separierbar → mehrschichtige Perzeptronen



Neural Networks

