Lineare Klassifikatoren

Volker Tresp

Klassifikatoren

- Klassifikation ist die zentrale Aufgabe in der Mustererkennung
- Sensoren liefern mir Informationen über ein Objekt
- Zu welcher Klasse gehört das Objekt: Hund, Katze, ...
- Welches Objekt ist es? Fiffi?

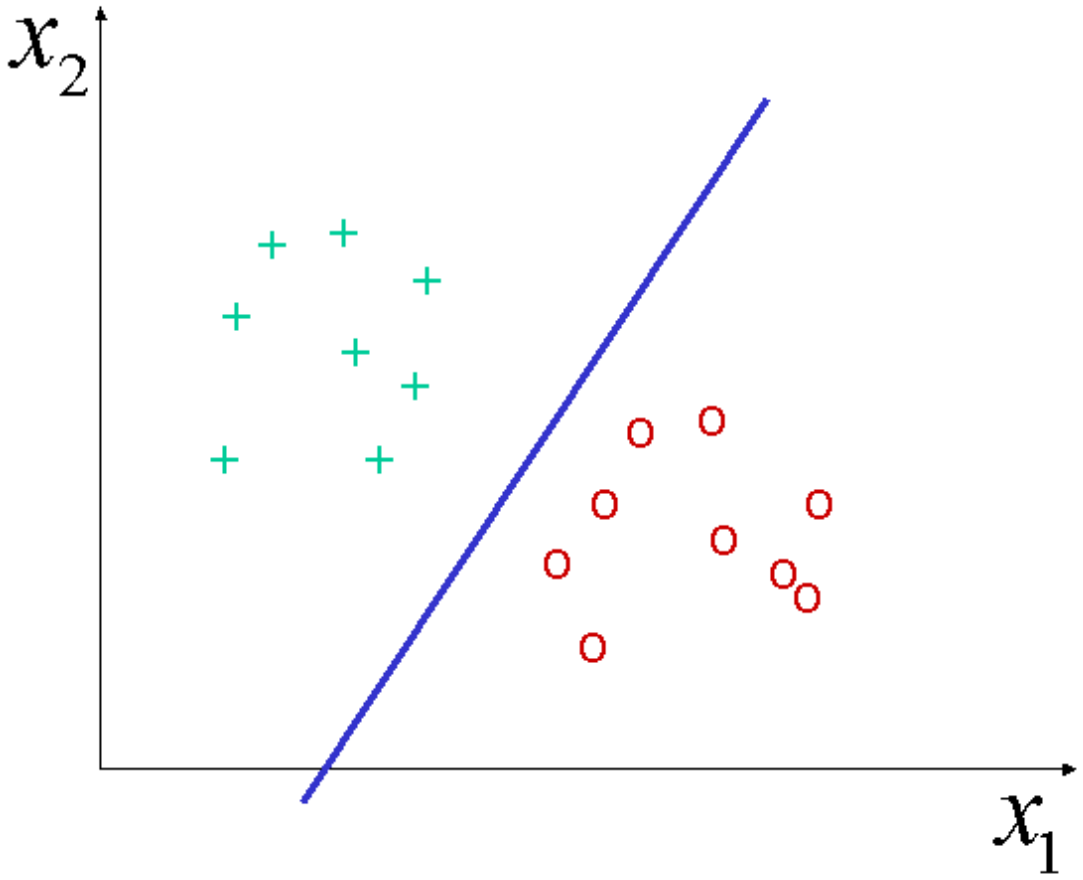
Lineare Klassifikatoren

- Lineare Klassifikatoren trennen Klassen durch eine lineare Hyperebene (genauer: affine Menge)
- In hoch dimensionalen Problemen trennt schon eine lineare Trennebene die Klassen
- Lineare Klassifikatoren können nicht das *exclusive-or* Problem lösen
- Im Zusammenhang mit Basisfunktionen oder Kernfunktionen können jedoch beliebige Trennflächen modelliert werden

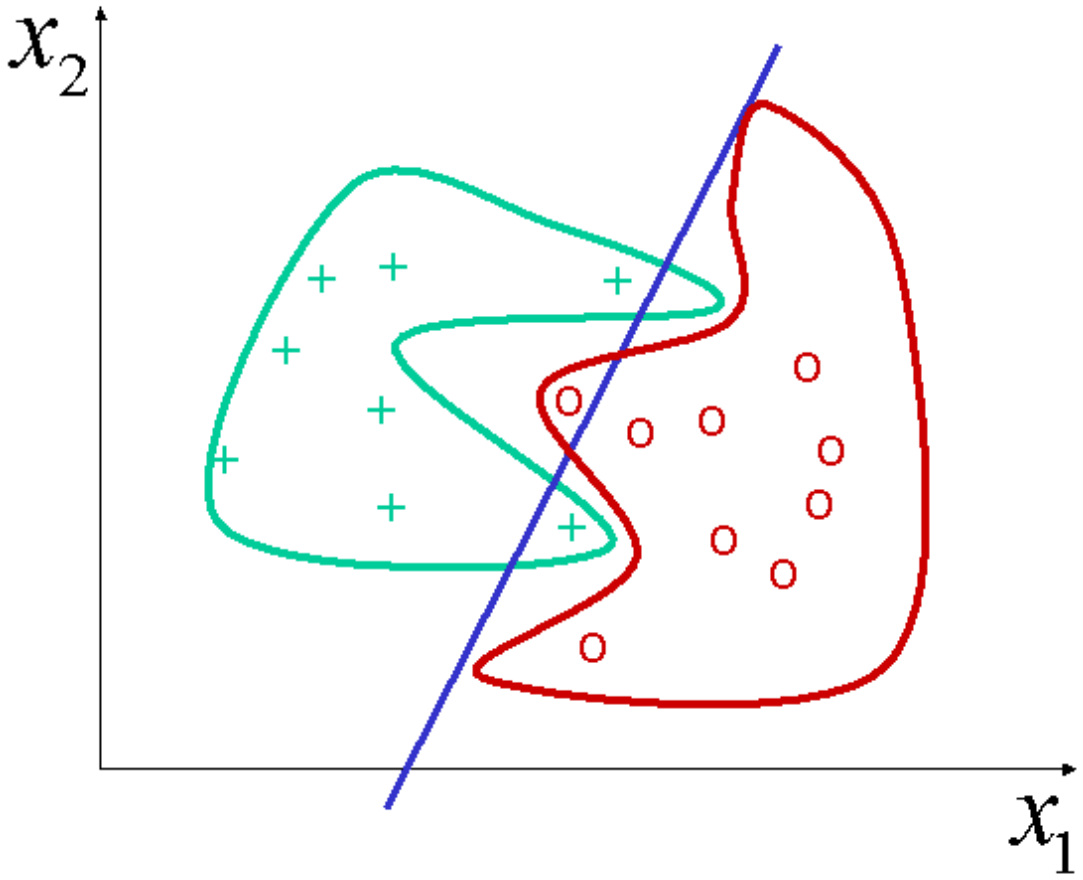
Einführung (2)

- Wir werden uns zunächst auf Klassifikatoren konzentrieren, die zwei Klassen $y_i = 1$ und $y_i = 0$ (oder $y_i = 1$ und $y_i = -1$) voneinander trennen
- Das *Perzeptron* hatten wir bereits kennengelernt. Wir behandeln hier folgende weitere Ansätze
  - I. Generatives Modell zur Klassifikation
  - II. Logistische Regression
  - III. Klassifikation durch Regression

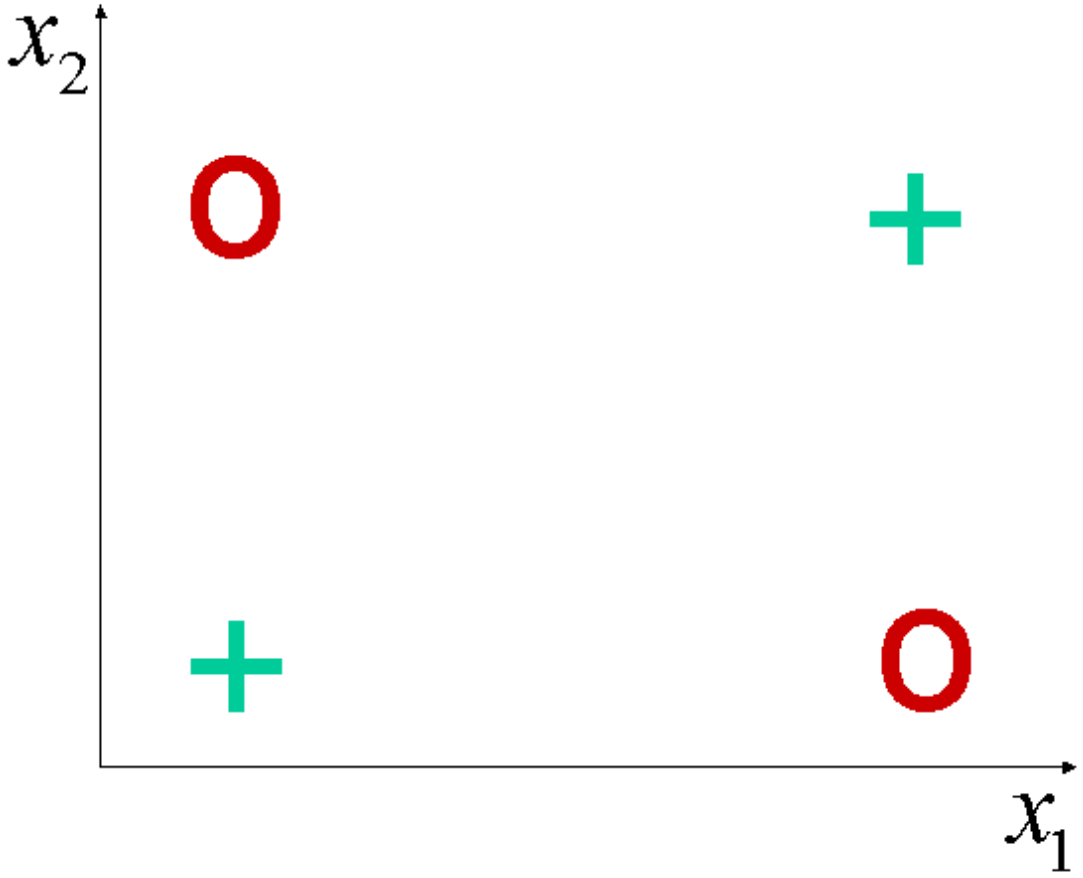
Zwei linear-separierbare Klassen



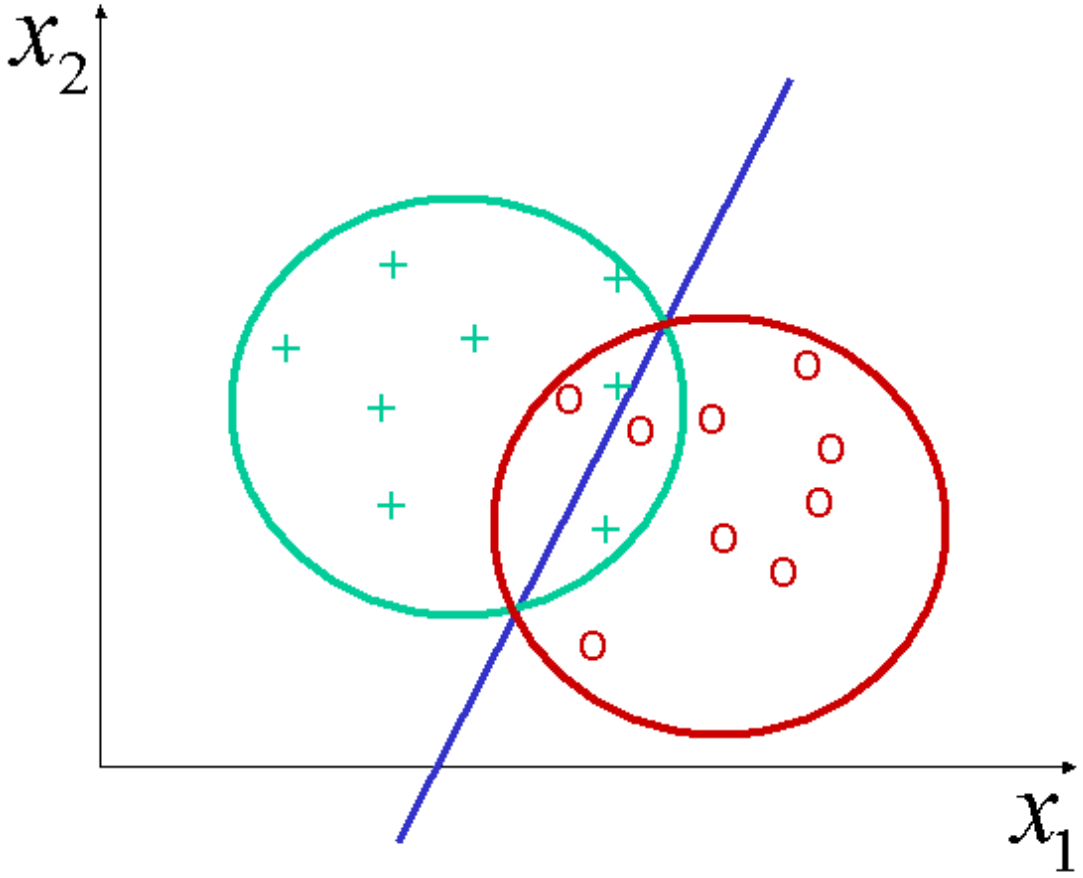
Zwei Klassen, die nicht linear separierbar sind



Das klassische Beispiel nicht-separierbarer Klassen: XOR



Separierbarkeit ist kein Ziel an sich: überlappende Klassen



I. Generatives Modell zur Klassifikation

- In einem generativen Modell modelliert man einen angenommenen datengenerierenden Prozess
- Wir nehmen an, dass die beobachtete Klasse y_i mit Wahrscheinlichkeit

$$P(y_i = 1) = \kappa_1 \qquad P(y_i = 0) = \kappa_0 = 1 - \kappa_1$$

  generiert wurde. Anschließend wurde nach $P(\mathbf{x}_i|y_i)$ ein Datenpunkt $\mathbf{x}_i$ erzeugt
- (Beachte, dass nur hier $\mathbf{x}_i = (x_{i,1}, \ldots, x_{i,M})^T$, das heißt $\mathbf{x}_i$ enthält nicht den Bias $x_{i,0}$)

Satz von Bayes

- Wollen wir nun einen Datenpunkt $\mathbf{x}_i$ klassifizieren für den y_i unbekannt ist, so bemühen wir den Satz von Bayes und erhalten

$$P(y_i|\mathbf{x}_i) = \frac{P(\mathbf{x}_i|y_i)P(y_i)}{P(\mathbf{x}_i)}$$

$$P(\mathbf{x}_i) = P(\mathbf{x}_i|y_i = 1)P(y_i = 1) + P(\mathbf{x}_i|y_i = 0)P(y_i = 0)$$

- Maximum-likelihood Schätzer für die Klassenwahrscheinlichkeiten sind

$$\widehat{P}(y_i = 1) = \widehat{\kappa}_1 = N_1/N$$

und

$$\widehat{P}(y_i = 0) = \widehat{\kappa}_0 = N_0/N = 1 - \widehat{\kappa}_1$$

wobei N_1 und N_0 die Anzahl der Trainingsmuster der Klasse 1 b.z.w. der Klasse 0 sind

Klassenspezifische Verteilungen

- Zur Modellierung von $P(\mathbf{x}_i|y_i)$ kann man nun problemangepasste Parametrierungen wählen
- Eine mögliche Wahl sind multivariate Gauß-Verteilungen

$$P(\mathbf{x}_i|y_i = l) = \mathcal{N}(\mathbf{x}_i; \mu^{(l)}, \Sigma)$$

mit

$$\mathcal{N}\left(\mathbf{x}_i; \mu^{(l)}, \Sigma\right) = \frac{1}{(2\pi)^{M/2}\sqrt{|\Sigma|}} \exp\left(-\frac{1}{2}\left(\mathbf{x}_i - \mu^{(l)}\right)^T \Sigma^{-1}\left(\mathbf{x}_i - \mu^{(l)}\right)\right)$$

- Beachte, dass beide Gauss-Verteilungen die gleiche Kovarianzmatrix aber andere Zentren besitzen (diese Wahl ist nicht zwingend, hat sich aber als oft sinnvoll erwiesen)

Maximum-likelihood Schätzer für Zentren und Kovarianzen

- Man erhält als maximum-likelihood Schätzer für die Zentren

$$\widehat{\mu}^{(l)} = \frac{1}{N_l} \sum_{i:y_i=l} \mathbf{x}_i$$

- Man erhält als unverzerrte (*unbiased*) Schätzer für die Kovarianzmatrix

$$\widehat{\Sigma} = \frac{1}{N-M} \sum_{l=0}^{1} \sum_{i:y_i=l} (\mathbf{x}_i - \widehat{\mu}^{(l)})(\mathbf{x}_i - \widehat{\mu}^{(l)})^T$$

Abgeleitete a posteriori Verteilung

- Es folgt

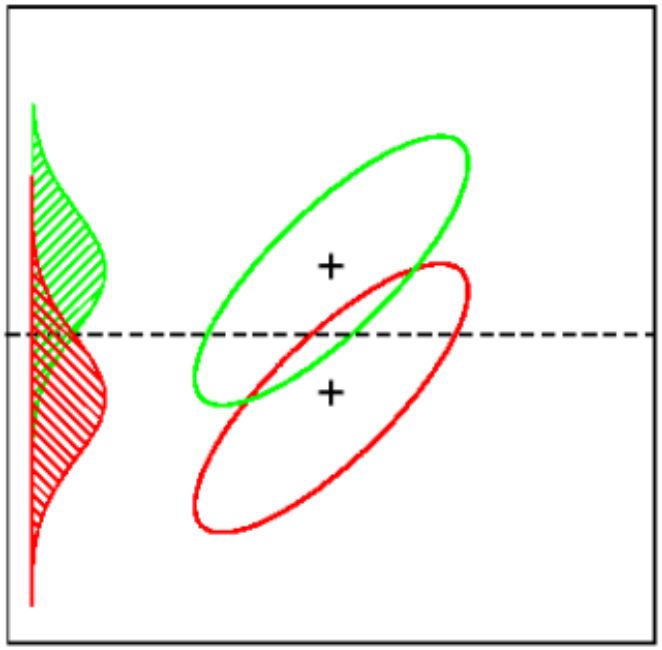
$$P(y_i = 1|\mathbf{x}_i) = \frac{P(\mathbf{x}_i|y_i = 1)P(y_i = 1)}{P(\mathbf{x}_i|y_i = 1)P(y_i = 1) + P(\mathbf{x}_i|y_i = 0)P(y_i = 0)}$$

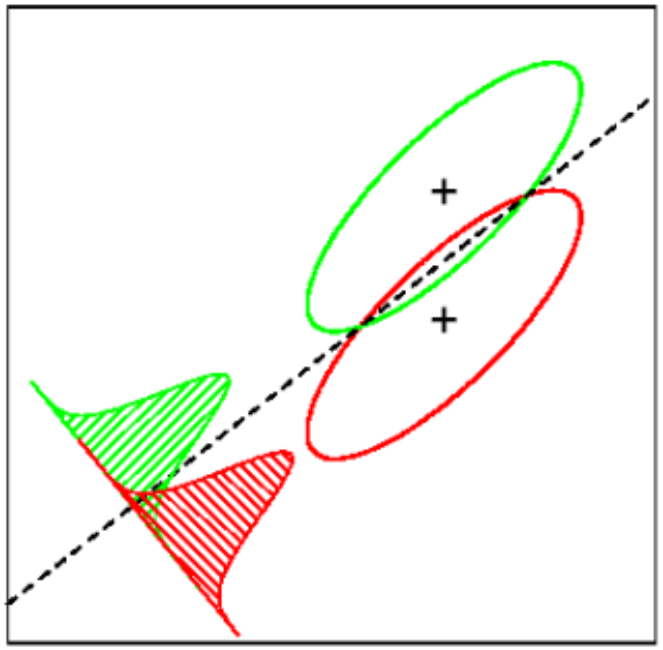
$$= \frac{1}{1 + \frac{\kappa_0}{\kappa_1} \exp\left((\mu^{(0)} - \mu^{(1)})^T \Sigma^{-1} \mathbf{x}_i + \frac{1}{2}\mu^{(0)^T}\Sigma^{-1}\mu^{(0)} - \frac{1}{2}\mu^{(1)^T}\Sigma^{-1}\mu^{(1)}\right)}$$

$$= \text{sig}\left(\begin{pmatrix} 1 \\ \mathbf{x}_i \end{pmatrix}^T \begin{pmatrix} w_0 \\ \mathbf{w} \end{pmatrix}\right)$$

$$\mathbf{w} = \Sigma^{-1}\left(\mu^{(1)} - \mu^{(0)}\right)$$

$$w_0 = \log \kappa_1/\kappa_0 - \frac{1}{2}\mu^{(0)^T}\Sigma^{-1}\mu^{(0)} + \frac{1}{2}\mu^{(1)^T}\Sigma^{-1}\mu^{(1)}$$





Kommentare

- Der generative Klassifikator führte zu einer linearen Trennebene
- Der vorgestellte Ansatz ist analog zu Fishers linearer Diskriminantenanalyse, bei der eine Projektion der Daten auf eine Dimension gesucht, so dass die Daten der gleichen Klasse eine kleine Varianz besitzen und gleichzeitig der Abstand der Zentren der beiden Klassen in der Projektion maximal ist
- Das heißt man bekommt das gleiche Ergebnis aus einem Optimalitätskriterium ohne Gauss-Verteilungen annehmen zu müssen

Spezialfall: Naive Bayes

- Wenn die Kovarianzmatrizen als diagonal angenommen werden erhält man einen sogenannten *Naive-Bayes* Klassifikator

$$P(\mathbf{x}_i|y_i = l) = \prod_{j=1}^{M} \mathcal{N}(\mathbf{x}_{i,j}; \mu_j^{(l)}, \sigma_j^2)$$

- Der naive Bayes Klassifikator hat wesentlich weniger Parameter aber ignoriert vollständig die Korrelation zwischen den Merkmalen, was manchmal als "naive" angesehen wird

- Appendix: Naive Bayes Klassifikatoren sind recht beliebt in der Textanalyse mit oft mehr als 10000 Merkmalen (Termen).

II. Logistische Regression

- Das generative Modell motivierte

$$P(y_i = 1|\mathbf{x}_i) = \text{sig}\left(\mathbf{x}_i^T \mathbf{w}\right)$$

(jetzt haben wir wieder $\mathbf{x}_i^T = (x_{i,0} = i, 1, x_{i,1}, \ldots, x_{i,M-1})^T$). sig() wie vorher definiert (logistische Funktion).

- Es ist nun nahe liegend, dass man die Likelihood des bedingten Modells optimiert

$$L(\mathbf{w}) = \prod_{i=1}^{N} \text{sig}\left(\mathbf{x}_i^T \mathbf{w}\right)^{y_i} \left(1 - \text{sig}\left(\mathbf{x}_i^T \mathbf{w}\right)\right)^{1-y_i}$$

Log-Likelihood

- Log-Likelihood-Funktion

$$l = \sum_{i=1}^{N} y_i \log\left(\text{sig}\left(\mathbf{x}_i^T \mathbf{w}\right)\right) + (1 - y_i) \log\left(1 - \text{sig}\left(\mathbf{x}_i^T \mathbf{w}\right)\right)$$

$$l = \sum_{i=1}^{N} y_i \log\left(\frac{1}{1 + \exp(-\mathbf{x}_i^T \mathbf{w})}\right) + (1 - y_i) \log\left(\frac{1}{1 + \exp(\mathbf{x}_i^T \mathbf{w})}\right)$$

$$= -\sum_{i=1}^{N} y_i \log(1 + \exp(-\mathbf{x}_i^T \mathbf{w})) + (1 - y_i) \log(1 + \exp(\mathbf{x}_i^T \mathbf{w}))$$

Adaption

- Die Ableitung der Log-Likelihood nach den Gewichten

$$\frac{\partial l}{\partial \mathbf{w}} = \sum_{i=1}^{N} y_i \frac{\mathbf{x}_i \exp(-\mathbf{x}_i^T \mathbf{w})}{1 + \exp(-\mathbf{x}_i^T \mathbf{w})} - (1 - y_i) \frac{\mathbf{x}_i \exp(\mathbf{x}_i^T \mathbf{w})}{1 + \exp(\mathbf{x}_i^T \mathbf{w})}$$

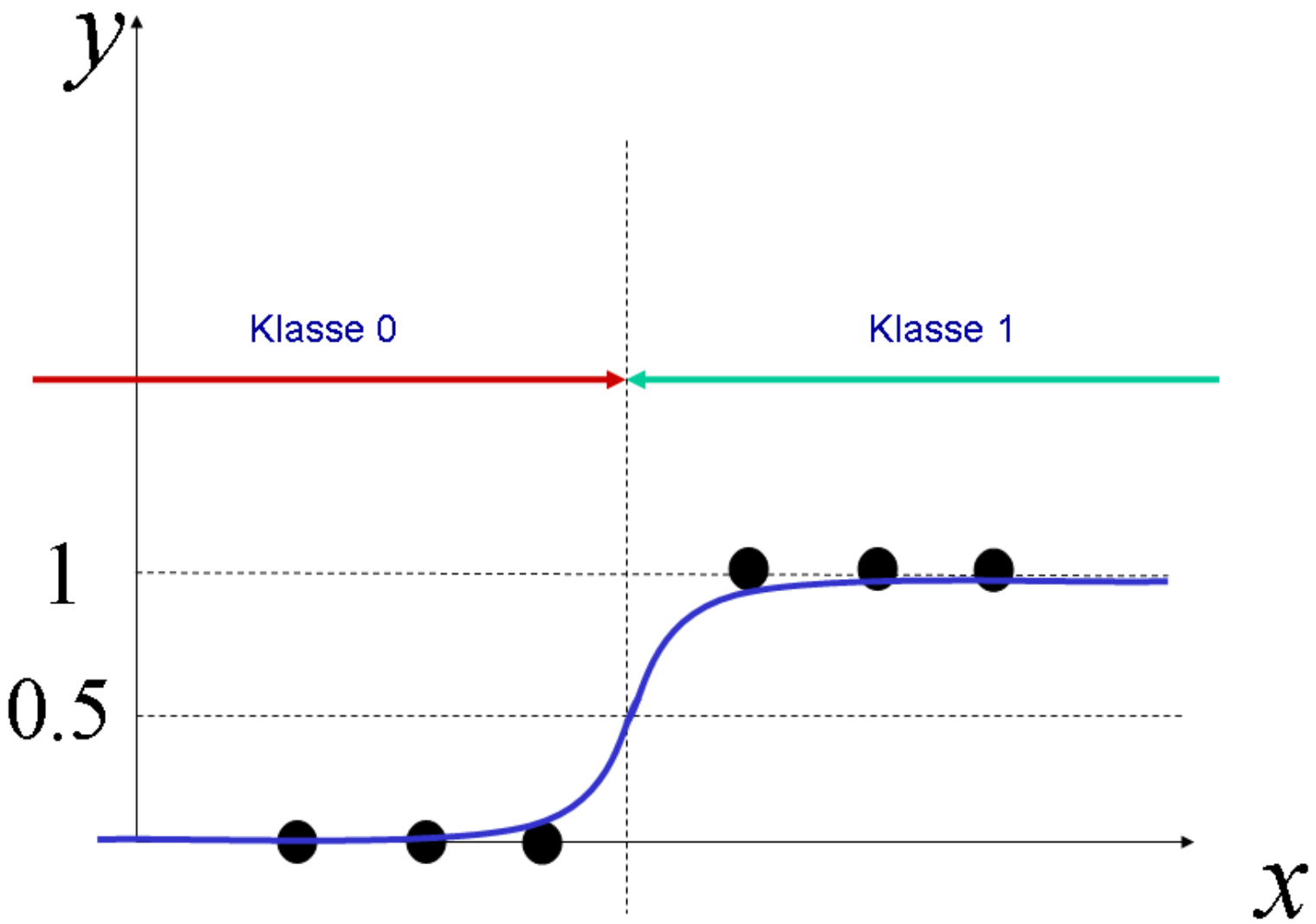
$$= \sum_{i=1}^{N} y_i \mathbf{x}_i (1 - \mathrm{sig}(\mathbf{x}_i^T \mathbf{w})) - (1 - y_i) \mathbf{x}_i \mathrm{sig}(\mathbf{x}_i^T \mathbf{w})$$

$$= \sum_{i=1}^{N} (y_i - \mathrm{sig}(\mathbf{x}_i^T \mathbf{w})) \mathbf{x}_i$$

- Gradientenbasierte Optimierung der Parameter (zur *Maximierung* der Log-Likelihood)

$$\mathbf{w} \longleftarrow \mathbf{w} + \eta \frac{\partial l}{\partial \mathbf{w}}$$

- Üblicherweise wird jedoch ein Newton-Raphson Verfahren zur Optimierung benutzt


y
Klasse 0
Klasse 1
1
0.5
x

Log-Likelihood und Log-Odds

- Mit

$$P(y_i = 1|\mathbf{x}_i) = \text{sig}\left(\mathbf{x}_i^T \mathbf{w}\right)$$

-

$$\log \frac{P(y_i = 1|\mathbf{x}_i)}{P(y_i = 0|\mathbf{x}_i)} = \log \frac{1}{1 + \exp(-\mathbf{x}_i^T \mathbf{w})} \frac{1 + \exp(-\mathbf{x}_i^T \mathbf{w})}{\exp(-\mathbf{x}_i^T \mathbf{w})}$$

$$= \log \frac{1}{\exp(-x)} = \mathbf{x}_i^T \mathbf{w}$$

- Eine andere Sicht: Die Log-Odds Funktion ist linear

Logistische Regression: Exponentialfamilie und Generalisierte Lineare Modelle

- Generatives Likelihood Model: Man kann auch folgendes Experiment annehmen. Zu einem $\mathbf{x}_i$ berechne ich ein $0 \leq \theta(\mathbf{x}_i) \leq 1$. Ich nehme eine unfaire Münze (biased coin) und wähle Klasse 1, wenn die Münze Kopf zeit und sonst Klasse 0 (Bernoulli Experiment). D.h. ich habe

$$P(y_i = 1|\mathbf{x}_i) = \theta(\mathbf{x}_i)$$

- Die Nebenbedingung $0 \leq \theta(\mathbf{x}_i) \leq 1$ sind unpraktisch

- Die Bernoulliverteilung gehört zur Exponentialfamilie und kann daher bequem reparametrisiert werden mit dem natürlichen Parameter

$$\eta = \log \frac{\theta}{1-\theta}$$

und die inverse Parameterabbildung ist dann

$$\theta = \frac{\exp \eta}{1 + \exp \eta} = \frac{1}{1 + \exp -\eta} = \text{sig}(\eta)$$

- Modellieren wir η als lineare Funktion des Eingangs erhalten wir die logistische Regression. Die logistische Regression gehört zur Familie der Generalisierten Linearen Modelle.

- Eine große Klasse wichtiger Verteilungen (Bernoulli, Gauss, Binomial, multinomial, Poisson, ...) gehören zur Exponentialfamilie und erlauben eine einheitliche Parametrisierung und die Herleitung entsprechender Generalisierter Linearer Modelle.

$$P(x|w) = \frac{1}{Z(w)} \exp\left(w^T F(x)\right)$$

$$F(x) = (1, T_1(x), T_2(x), ...)^T$$

  wobei $T_1(x), T_2(x), ...$ eine suffiziente Statistik (erschöpfende Statistik) der Verteilung ist.

Logistische Kostenfunktion (logistic loss) (cross-entropy cost function)

- Wir nehmen die negative Log-Likelihood als Kostenfunktion. Dann kann man schreiben für $y_i \in \{0, 1\}$

$$cost = \sum_{i=1}^{N} y_i \log(1 + \exp(-\mathbf{x}_i^T \mathbf{w})) + (1 - y_i) \log(1 + \exp(\mathbf{x}_i^T \mathbf{w}))$$

$$= \sum_{i=1}^{N} \left(1 + \exp\left((1 - 2y_i)\mathbf{x}_i^T \mathbf{w}\right)\right)$$

- und für $y_i \in \{-1, 1\}$

$$cost = \sum_{i=1}^{N} \left(1 + \exp\left(-y_i \mathbf{x}_i^T \mathbf{w}\right)\right)$$

III. Klassifikation durch Regression

- Lineare Regression:

$$f(\mathbf{x}_i, \mathbf{w}) = w_0 + \sum_{j=1}^{M-1} w_j x_{i,j}$$

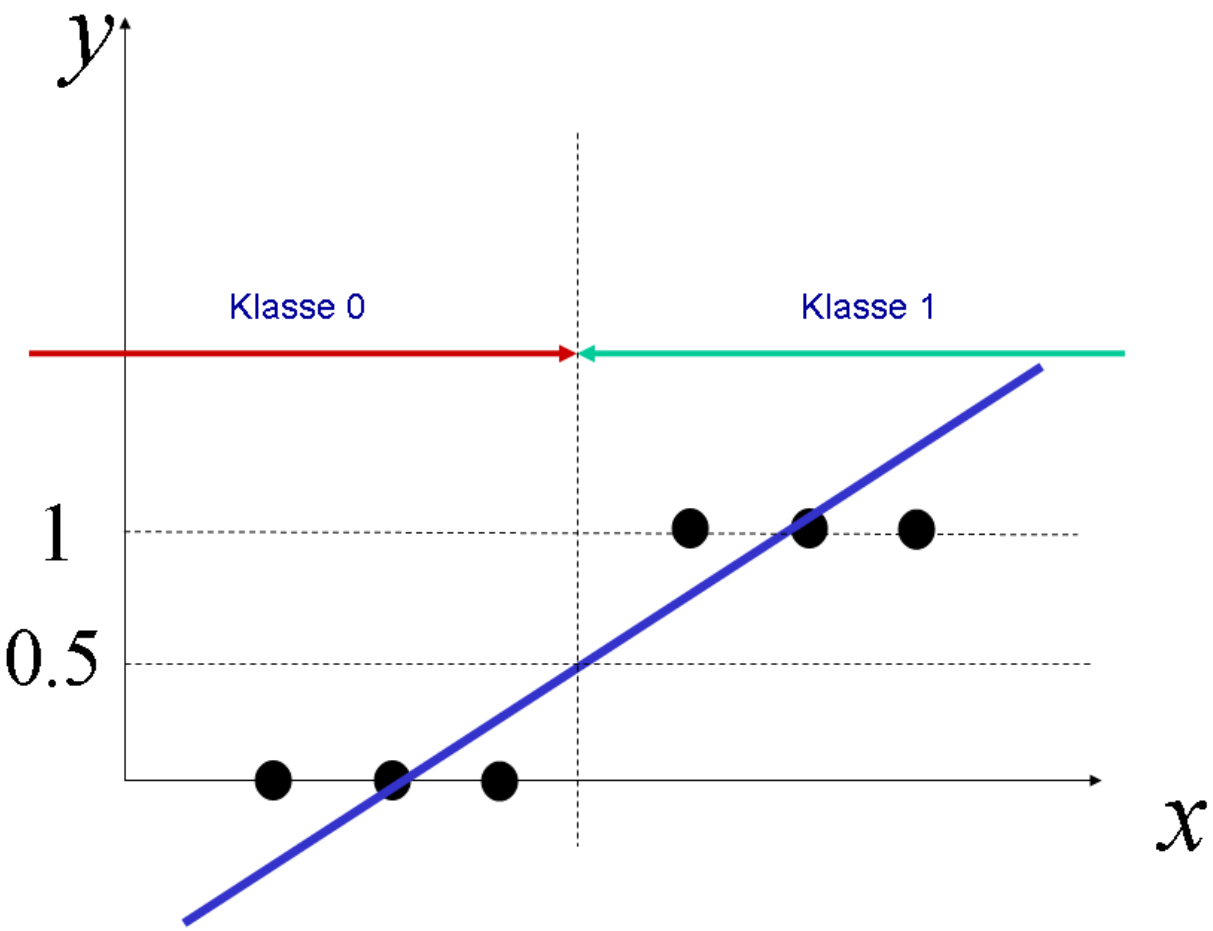
$$= \mathbf{x}_i^T \mathbf{w}$$

- Wir definieren als Zielgröße $y_i = 1$ falls Muster $\mathbf{x}_i$ zu Klasse 1 gehört und $y_i = 0$ falls Muster $\mathbf{x}_i$ zu Klasse 0 gehört
- Wir berechnen Gewichte $\mathbf{w}_{LS} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$ als LS-Lösung, genau wie in der linearen Regression
- Für einen neues Muster $\mathbf{z}$ berechnen wir $f(\mathbf{z}) = \mathbf{z}^T\mathbf{w}_{LS}$ und ordnen das Muster Klasse 1 zu falls $f(\mathbf{z}) > 1/2$; ansonsten ordnen wir das Muster Klasse 0 zu

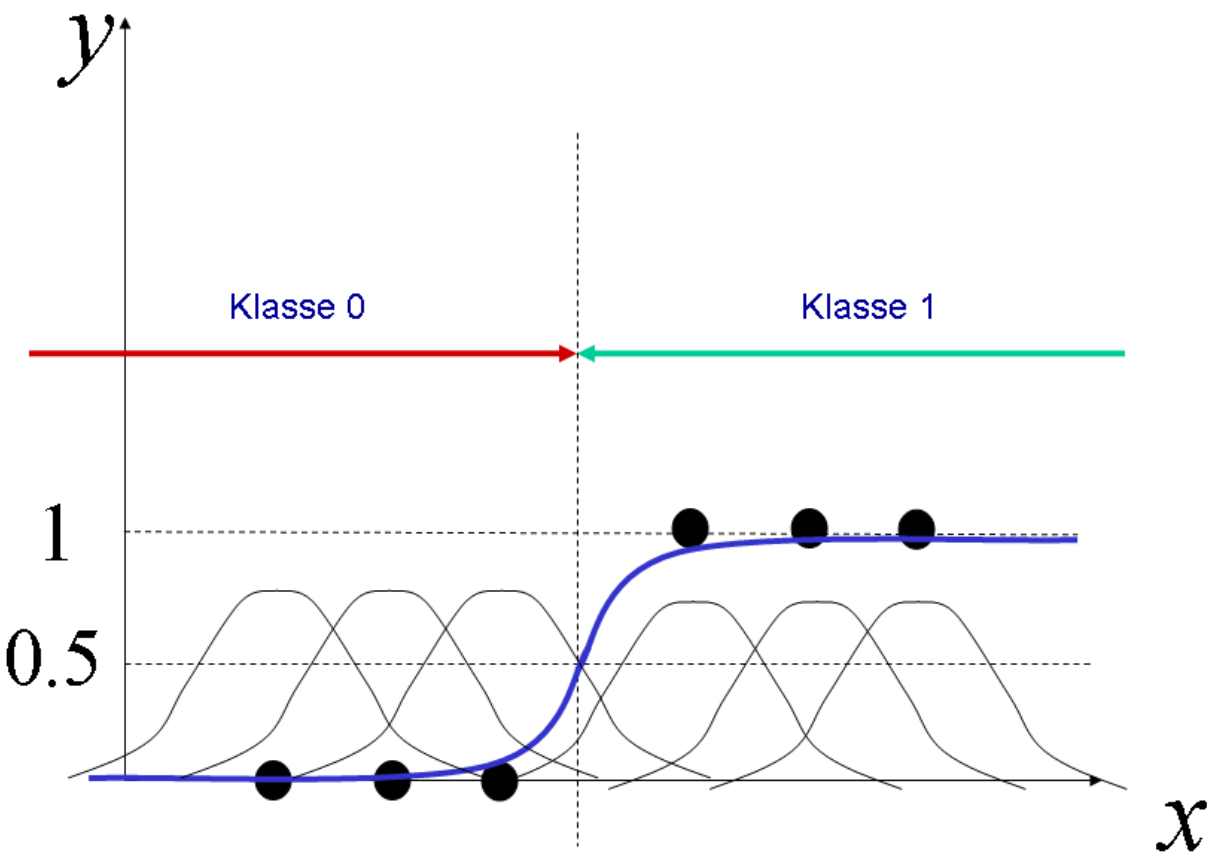
Bias

- Asymptotisch konvergiert eine LS-Lösung zur Klassenwahrscheinlichkeit $P(c = 1|\mathbf{x})$; allerdings ist eine lineare Funktion in der Regel nicht fähig, die Klassenwahrscheinlichkeit zu repräsentieren (Bias!); dennoch kann die resultierende Klassifikationsentscheidung durchaus sinnvoll sein
- Man kann gute Klassifikationsentscheidungen in hohen Dimensionen erwarten und im Zusammenhang mit Basisfunktionen und Kernfunktionen; in manchen Fällen erreicht man dann Konsistenz

Klassifikation durch Regression mit linearen Funktionen



Klassifikation durch Regression mit radialen Basisfunktionen



Performanz

- Obwohl der Ansatz für lineare Klassifikation etwas simplistisch erscheint, ist die Performanz in Kombination mit Basisfunktionen oder Kernfunktionen durchaus sehr gut! Dort wegen der großen Einfachheit und guten Performanz sehr populär

Vergleich: Musterbasiertes Lernen

- Perzeptron

$$w_j \longleftarrow w_j + \eta \left(y(t) - \text{sign} \left(x(t)^T w \right) \right) x_j(t)$$

$y(t) \in \{-1, 1\}$. Beachte, dass die Klammer Null ist, wenn richtig klassifiziert. Ansonsten ist der Term entweder gleich 2 oder gleich -2.

- Logistic regression

$$w_j \longleftarrow w_j + \eta \left(y(t) - \text{sig} \left(x(t)^T w \right) \right) x_j(t)$$

Die "natürliche" kontinuierliche Verallgemeinerung; $y(t) \in \{-1, 1\}$

- Neuronale Netze

$$w_j \longleftarrow w_j + \eta \left(y(t) - \text{sig} \left(x(t)^T w \right) \right) \text{ sig}' \left(x(t)^T w \right) \; x_j(t)$$

Wird ein Muster mit hoher Sicherheit falsch klassifiziert, ist der Gradient nahezu Null!

- Regression (ADALINE)

$$w_j \longleftarrow w_j + \eta \left(y(t) - \left(x(t)^T w \right) \right) x_j(t)$$

Wichtige Gütemaße für Klassifikation

- Sei auf ungesehenen Daten y das wahre Klassenlabel und $pred$ das vorhergesagte. Dann wird definiert

	$y = 1$	$y = 0$
$pred = 1$	tp (true positive)	fp (false positive)
$pred = 0$	fn (false negative)	tn (true negative)

Wobei $tp + fp + fn + tn = K$ (Anzahl der Testdaten).

Beispiel: Sichelanämie in Schwarzafrika mit perfektem Klassifikator

	SA	keine SA
$pred = 1$	40	0
$pred = 0$	0	9960

Sichelanämie in Schwarzafrika mit schlechtem Klassifikator

	SA	keine SA
$pred = 1$	2	1
$pred = 0$	38	9959

Accuracy

- **Accuracy** (allg. Gütemaß, die Wahrscheinlichkeit, richtig zu klassifizieren):

$$P(pred = 1, y = 1) + P(pred = 0, y = 0) \rightarrow \frac{tp + tn}{tp + tn + fn + fp}$$

  Die Fehlerwahrscheinlichkeit ist dann (1-Accuracy).

  Wenn Klassen unballanziert sind schlecht zu interpretieren. Im Beispiel hat der schlechte Klassifikator eine Accurycy von 0,996 und der perfekte Klassifikator von 1,0. Eine Suchmaschine, die nie eine Seite zurückliefert hat eine Accuracy nahe 1. Eine Suchmaschine, die immer alle Seiten zurückliefert hat eine Accuracy nahe 0.

Recall

- **Recall** (Sensitivität, sensitivity, true positive rate):

$$P(pred = 1|y = 1) \rightarrow \frac{tp}{tp + fn}$$

(Eine Suchmaschine sollten alle relevanten Dokumente finden; ein Feuermelder sollte jedes Feuer melden)

Im Beispiel hat der schlechte Klassifikator einen Recall von 0.05 und der perfekte Klassifikator von 1,0. Eine Suchmaschine, die nie eine Seite zurückliefert hat einen Recall nahe 0. Eine Suchmaschine, die immer alle Seiten zurückliefert hat einen Recall nahe 1.

Precision

- **Precision** (Relevanz, Wirksamkeit, Genauigkeit)

$$P(y = 1|pred = 1) \rightarrow \frac{tp}{tp + fp}$$

(Eine Suchmaschine sollten nur relevante Dokumente finden; wenn der Feuermelder anschlägt, sollte es auch ein Feuer geben)

Im Beispiel hat der schlechte Klassifikator einen Recall von 0,66 und der perfekte Klassifikator von 1,0. Eine Suchmaschine, die nie eine Seite zurückliefert hat einen undefinierten Recall. Eine Suchmaschine, die immer alle Seiten zurückliefert hat eine Precision nahe 0.

Specifity

- **Specificity** (Spezifität, true negative rate)

$$P(pred = 0|y = 0) \rightarrow \frac{tn}{tn + fp}$$

(Wenn es kein Feuer gibt, sollte der Feuermelder nicht anschlagen)

Im Beispiel hat der schlechte Klassifikator einen Recall von 0,9999 und der perfekte Klassifikator von 1,0. Eine Suchmaschine, die nie eine Seite zurückliefert hat einen Specificy nahe 1. Eine Suchmaschine, die immer alle Seiten zurückliefert hat eine Specificy nahe 0.

F-Measure

- **F-measure** (F-Maß)

$$F = 2\frac{Precision \times Recall}{Precision + Recall}$$

Im Beispiel hat der schlechte Klassifikator einen Recall von 0,093 und der perfekte Klassifikator von 1,0. Eine Suchmaschine, die nie eine Seite zurückliefert hat eine F-measure nahe 0. Eine Suchmaschine, die immer alle Seiten zurückliefert hat eine F-measure nahe 0.

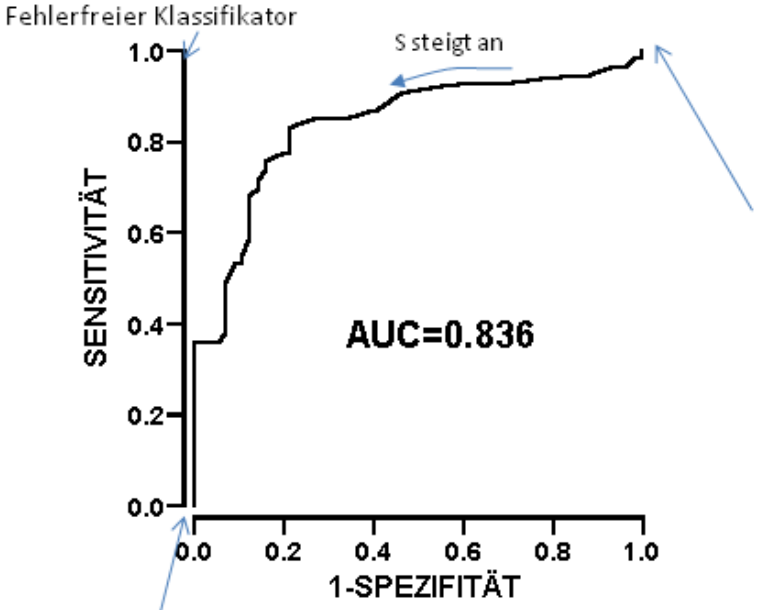
ROC, AUC-ROC, AUC-PR

Betrachte als Entscheidung die Funktion $\mathbf{sign}(f(x) + \alpha)$ für $-\infty < \alpha < \infty$, d.h. wir variieren die Schwelle für die Klassifikation. Man kann Recall beliebig verbessern mit $\alpha \to -\infty$ bzw Precision beliebig verbessern mit $\alpha \to \infty$. Wichtig ist es, einen Anwendungsspezifischen optimalen Kompromiss zu treffen.

- ROC (Receiver operating characteristic). Für die ROC-Kurve wird α variiert und Recall (y-Achse) gegen (1-Specific) (x-Achse) geplottet. (1-Specific) wird auch as *false-positive rate* bezeichnet.
- AUC-ROC: Area under ROC curve. Numerische Integration der ROC Kurve. Ein perfekter Klassifikator ergibt AUC-ROC $= 1$, eine Zufallsklassifikation eine AUC-ROC $=$ 0.5. Zum Beispiel mit Trapezregel berechnet.
- Ebenso populär ist die AUC-PR curve. Hier wird Precision als Funktion von Recall aufgetragen und die Funktion numerisch integriert.

Die Receiver Operating Characteristic (ROC) – Kurve

•Gibt mein Klassifikator eine Klassenwahrscheinlichkeit aus, dann entscheide ich mich für Klasse 0, wenn dieser Wert unter einem Schwellwert S ist und ansonsten entscheide ich mich für Klasse 1
•(0,0): S=1 ($\alpha=-\infty$) (1,1): S= ist 0 ($\alpha=\infty$) (0.3, 0.85): S=0.5 (Beispiel)



- Ich sage nur Klasse 1 voraus
- Mein Klassifikator sagt, dass jede Webseite ein Treffer ist
- Alle Patienten sind krank (Klasse 1)
- Spezifität: P(pred = 0 | y = 0) = 0 Sensitivität: P(pred = 1 | y = 1) = 1

- Ich sage nur Klasse 0 voraus
- Mein Klassifikator sagt, dass keine Webseite ein Treffer ist
- Alle Patienten sind gesund (Klasse 0)
- Spezifität: P(pred = 0 | y = 0) = 1 Sensitivität: P(pred = 1 | y = 1) = 0

•Das Integral unter der Kurve (area under curve, AUC-ROC) ist bei perfekter Klassifikation gleich 1 und bei Zufallsklassifikation gleich 0.5

Optimierung der entsprechenden Gütemaße

- Einige der Gütemaße können im Training direkt optimiert werden, durch Wahl geeigneter Kostenfunktionen
- Beispiele: F-Measure, AUC

Appendix

Naive Bayes multinomiales Modell

- Modell eines "unfairen" Würfels (mehrere Würfelwürfe)
- Multinomiales Modell mit $x_j \in \{1, \ldots, K_j\}$

$$P(\mathbf{x}|y = l) = Z \prod_{j=1}^{M} \gamma_{j,x_j,l} \quad \textbf{mit} \quad \gamma_{j,x_j,l} \geq 0, \sum_{k=1}^{K_j} \gamma_{j,k,l} = 1$$

- Für Texte ist $l \in \{1, \ldots, C\}$ die Klasse des Dokumentes, M ist die Länge des Dokumentes und $\gamma_{j,x_j,l} = \gamma_{x_j,l}$ ist die Wahrscheinlichkeit das Wort j zu beobachten. Dann können wir auch schreiben

$$P(\mathbf{x}|y = l) = Z \prod_{j=1}^{W} \gamma_{x_j,l}^{n_j}$$

wobei W die Größe des Vokabulars ist und n_j angibt wie häufig Wort j im Dokument vorkommt.

- Z ist ein für die Klassifikation unwichtiger Faktor, der die verschiedenen Permutationen berücksichtigt.

Naive Bayes Bernoulli Modell

- Modell einer "unfairen" Münze (ein Münzwurf)
- Bernoulli Modell mit $x_j \in \{0, 1\}$

$$P(\mathbf{x}|y = l) = \prod_{j=1}^{M} \gamma_{j,l}^{x_j} \, (1 - \gamma_{j,l})^{(1-x_j)} \quad 0 \leq \textbf{mit} \quad \gamma_{j,x_j,l} \leq 1$$

- Im Falle von zwei Klassen ergibt sich als a posteriori Verteilung wieder ein linearer Klassifikator mit sigmoider Aktivierungsfunktion
- Für Texte ist $x_j = 1$, wenn das Wort j im Text vorkommt und sonst $x_j = 0$. Das heißt,

$$P(\mathbf{x}|y = l) = \prod_{j=1}^{W} \gamma_{j,l}^{\delta_j} \, (1 - \gamma_{j,l})^{(1-\delta_j)}$$

wobei W die Größe des Vokabulars ist und $\delta_j = 1$ wenn Wort j im Dokument vorkommt und sonst $\delta_j = 0$.

Gauss'sche Mischverteilungen, probabilistisches Clustern, EM

- Flexible Verteilungen kann man durch eine Summe von elementaren Verteilungen modellieren,

$$P(\mathbf{x}_i) = \sum_{h=1}^{H} \alpha_h P(\mathbf{x}_i|h)$$

- $P(\mathbf{x}_i|h)$ können zum Beispiel Bernoulli. multinomiale oder Poisson Verteilungen sein. Sehr beliebt sind Gauß'sche Mischdichten mit

$$P(\mathbf{x}_i|h) = \mathcal{N}(\mathbf{x}_i; \mu^{(h)}, \Sigma^{(h)})$$

  wobei man meist eine diagonale Kovarianzmatrix wählt.

- Die Parameter im Modell können durch den iterativen EM-Algorithmus bestimmt werden (Expectation-Maximization-Algorithmus)

- Für Gauß'sche Verteilungen kann man die Zentren der Verteilungen $\mu^{(1)}, ..., \mu^{(H)}$ als repräsentative Muster interpretieren (Clusterzentren)