Universität Potsdam
Institut für Informatik
Lehrstuhl Maschinelles Lernen

Zusammenfassung: Lernprobleme, Bayes‘sches Lernen, Evaluierung

Christoph Sawade/Niels Landwehr/Paul Prasse
Silvia Makowski
Tobias Scheffer

Überblick

- Lernprobleme
- Bayes‘sches Lernen
- Evaluierung

Problemstellung Lernen

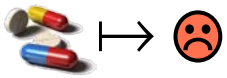
- Eingabe Lernproblem: Trainingsdaten.
  - $L = \langle (\mathbf{x}_1, y_1), ..., (\mathbf{x}_N, y_N) \rangle$
  - $\mathbf{x}_i = \begin{pmatrix} x_{i1} \\ ... \\ x_{im} \end{pmatrix}$

$\mathbf{x} \in \mathcal{X}, \quad y \in \mathcal{Y}$

$\mathcal{Y}$ diskret (Klassifikation) oder kontinuierlich (Regression)

- Ausgabe: Hypothese, Modell

$f : \mathcal{X} \rightarrow \mathcal{Y}$



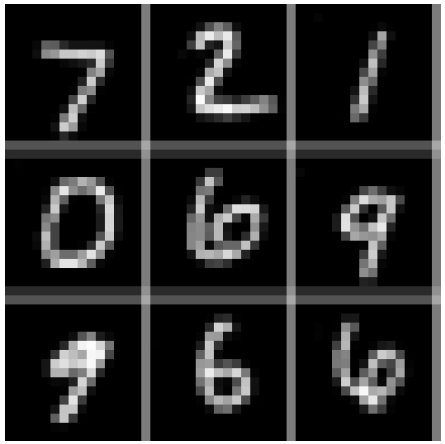
$$f(\mathbf{x}) = \begin{cases} \text{☹}, & \text{wenn } x_1 = 1 \wedge x_3 = 0 \wedge x_6 = 1 \\ \text{☺}, & \text{sonst} \end{cases}$$

Klassifikation: Beispiel

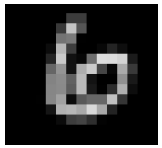
- Eingabe: Instanz (Objekt) $\mathbf{x} \in \mathcal{X}$.

  $\mathcal{X}$ = Menge aller 16x16 Pixel Bitmaps

Attribute	Instanz $\mathbf{x}$	
Grauwert Pixel 1	$\begin{pmatrix} 0.1 \\ 0.3 \\ 0.45 \\ \dots \\ 0.65 \\ 0.87 \end{pmatrix}$	256 Pixelwerte
...		
Grauwert Pixel 256		



- Ausgabe: $y \in \mathcal{Y} = \{0,1,2,3,4,5,6,7,8,9\}$: erkannte Ziffer



→ Klassifikator → "6"

Klassifikation: Beispiel

- Eingabe: Instanz (Objekt) $\mathbf{x} \in \mathcal{X}$.

$\mathcal{X}$ = Menge aller möglichen Email-Texte

Attribute

Wort 1 kommt vor?

…

Wort N kommt vor?

$N \approx 100000$

Instanz $\mathbf{x}$

$$\begin{pmatrix} 0 \\ 1 \\ 0 \\ \dots \\ 1 \\ 0 \end{pmatrix}$$

Alternative

Beneficiary

Friend

…

Sterling

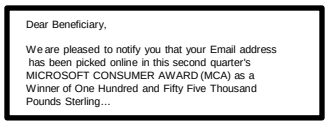
Zoo

Email

Dear Beneficiary,

your Email address has been picked online in this years MICROSOFT CONSUMER AWARD as a Winner of One Hundred and Fifty Five Thousand Pounds Sterling…

- Ausgabe: $y \in \mathcal{Y} = \{spam, ok\}$



→ Klassifikator → „Spam“

Lernen durch Berechnung des Version Space

- Version Space: alle mit den Daten konsisten Hypothesen

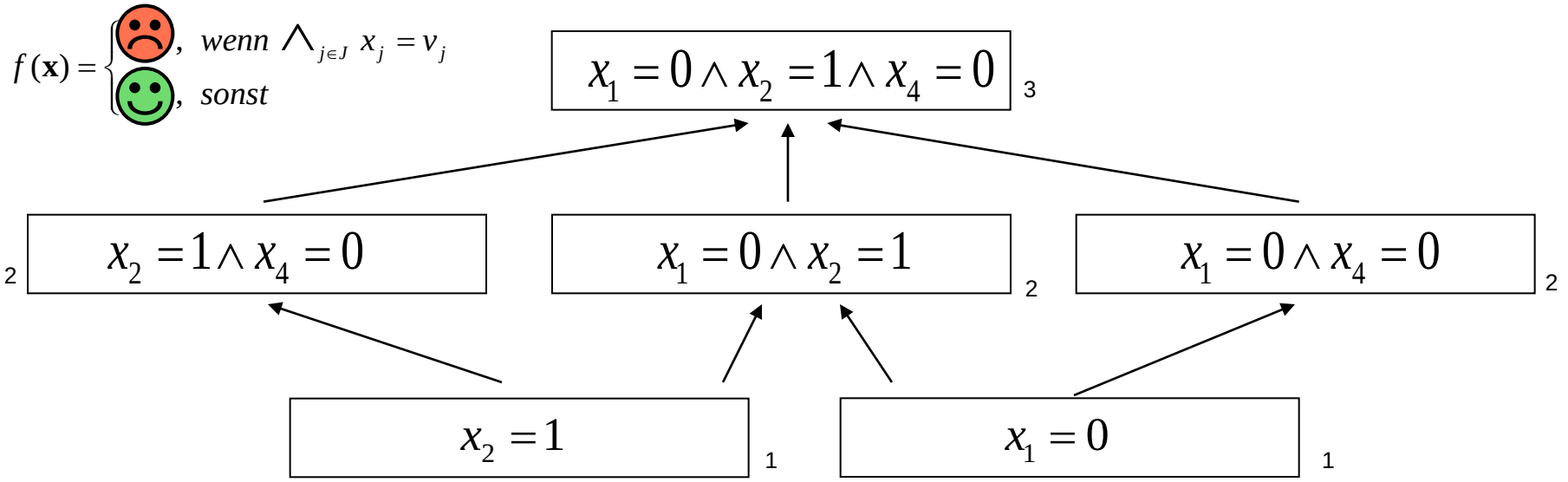
Medikamente in der Kombination

Beispiel-Kombinationen

	x_1	x_2	x_3	x_4	x_5	x_6	y
$\mathbf{x}_1$	0	1	0	0	1	1	☹
$\mathbf{x}_2$	0	1	1	0	1	1	☹
$\mathbf{x}_3$	1	0	1	0	1	0	☺
$\mathbf{x}_4$	0	1	1	0	0	0	☹

$$L = \langle (\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), (\mathbf{x}_3, y_3), (\mathbf{x}_4, y_4) \rangle$$

$$f(\mathbf{x}) = \begin{cases} ☹, & \text{wenn } \bigwedge_{j \in J} x_j = v_j \\ ☺, & \text{sonst} \end{cases}$$



Unsicherheit

- Version Space-Ansatz in der Praxis problematisch
  - Der Hypothesenraum ist meist unendlich groß.
  - Der Version Space ist dann meist auch unendlich groß, oder leer.

Alternativer Ansatz: Lernen als *Optimierungsproblem*

Verlustfunktion, Optimierungskriterium

- Alternativer Ansatz: Lernen als *Optimierungsproblem*

$$\hat{R}(\mathbf{w}, L) = \sum_i \ell(f_{\mathbf{w}}(\mathbf{x}_i), y_i) + \lambda |\mathbf{w}|^2$$

Verlust auf Trainingsdaten

Regularisierer

  - *Verlustfunktion* misst, wie gut Modell zu Trainingsdaten passt
  - *Regularisierungsfunktion* misst, ob das Modell nach unserem Vorwissen *wahrscheinlich* ist.
  - *Optimierungskriterium* ist Summe aus Verlust und Regularisierer.
  - Suchen Minimum des Optimierungskriteriums

Verlustfunktion

- Verschiedene Verlustfunktionen (anwendungsspezifisch)
- Beispiel: Verlustfunktionen in Form von Kostenmatrix

$$\ell(f(\mathbf{x}), y) = \begin{cases} f(\mathbf{x}) = +1 \\ f(\mathbf{x}) = -1 \end{cases}$$

	$y = +1$	$y = -1$
$f(\mathbf{x}) = +1$	0	c_{FP}
$f(\mathbf{x}) = -1$	c_{FN}	0

- Zum Beispiel diagnostische Klassifikationsprobleme, übersehene Erkrankungen (False Negatives) schlimmer als False Positives.

Regularisierer

- Unterschiedliche Regularisierer möglich

- Häufig wird die Annahme ausgedrückt, dass wenige der Attribute für ein gutes Modell ausreichen.
  - Anzahl der Attribute, L_0-Regularisierung
  - Betrag der Attribut-Gewichtungen, L_1-Regularisierung
  - Quadrat der Attribut-Gewichtungen, L_2-Regularisierung.

Optimierungsproblem

- Rechtfertigung für Optimierungskriterium?
- Mehrere Rechtfertigungen und Herleitungen.
  - **Wahrscheinlichste Hypothese (*MAP-Hypothese*).**
  - Hypothese, die Daten am stärksten komprimiert (*Minimum Description Length*).
  - Niedrige obere Schranke für Fehler auf zukünftigen Daten abhängig von $|\mathbf{w}|$. (*SRM*).

Überblick

- Lernprobleme
- Bayes‘sches Lernen
- Evaluierung

Diskrete und kontinuierliche Zufallsvariablen

- Diskrete Zufallsvariablen
  - Endliche, diskrete Menge von Werten D möglich
  - Verteilung beschrieben durch diskrete Wahrscheinlichkeiten

$$p(x) \in [0,1] \quad \text{mit} \quad \sum_{x \in D} p(x) = 1$$

- Kontinuierliche Zufallsvariablen
  - Kontinuierliche, unendliche Menge von Werten möglich
  - Verteilung beschrieben durch kontinuierliche Dichtefunktion

$$p(x) \in \mathbb{R}_{\geq 0} \qquad \text{mit} \quad \int_{-\infty}^{\infty} p(x)dx = 1$$

$$\text{Wahrscheinlichkeit für } x \in [a,b]\text{: } p(x \in [a,b]) = \int_{a}^{b} p(x)dx$$

Diskrete Zufallsvariablen

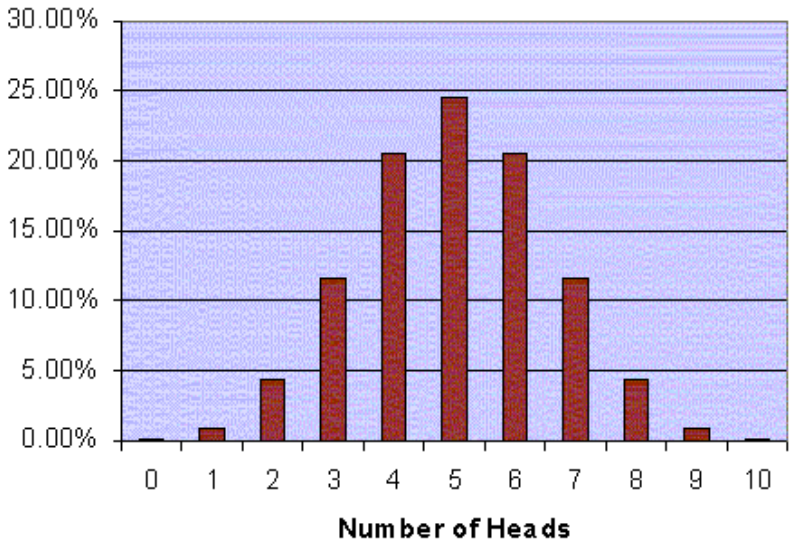
- Binomialverteilung: Anzahl „Köpfe“ bei *N* Münzwürfen

$$X_i \sim \text{Bern}(X_i \mid \mu)$$

$$X = \sum_{i=1}^{N} X_i, \qquad X \in \{0, \ldots, N\}$$

$$X \sim \text{Bin}(X \mid N, \mu)$$

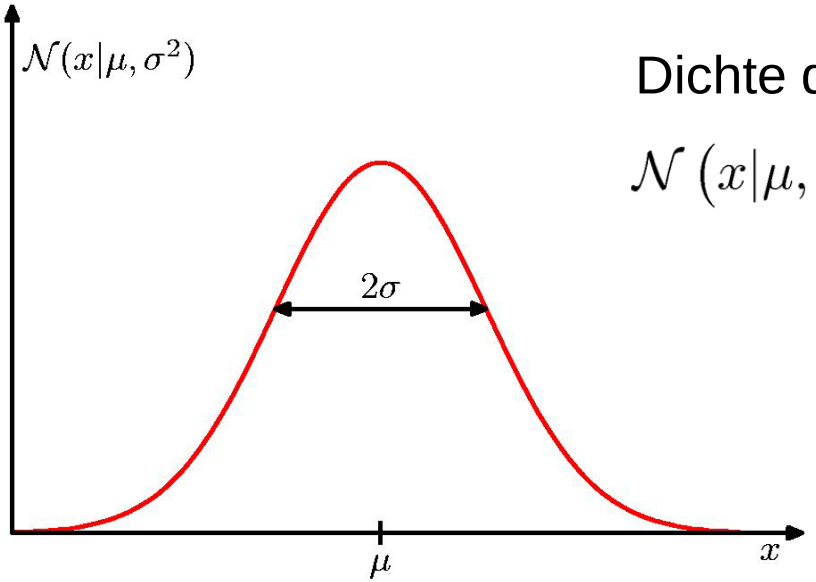
$$\text{Bin}(X \mid N, \mu) = \binom{N}{X} \mu^X (1-\mu)^{N-X}$$



$$N = 10, \quad \mu = 0.5$$

Kontinuierliche Zufallsvariablen

- Normalverteilung (Gaußverteilung)
- Beispiel: Körpergröße X
  - Annähernd normalverteilt: $X \sim \mathcal{N}(x \mid \mu, \sigma^2)$



Dichte der Normalverteilung

$$\mathcal{N}\left(x|\mu,\sigma^2\right) = \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left\{-\frac{1}{2\sigma^2}(x-\mu)^2\right\}$$

$$\mathcal{N}(x|\mu,\sigma^2) > 0$$

$$\int_{-\infty}^{\infty} \mathcal{N}\left(x|\mu,\sigma^2\right)\,\mathrm{d}x = 1$$

z.B. $\mu = 170, \sigma = 10$

Erwartungswert und Varianz

- Erwartungswert einer Zufallsvariable
  - Gewichteter Mittelwert

$$E(X) = \sum_x xp(X = x) \qquad X \text{ diskrete ZV}$$

$$E(X) = \int xp(x)dx \qquad X \text{ kontinuierliche ZV mit Dichte } p(x)$$

- Varianz einer Zufallsvariable
  - Erwartete quadrierte Abweichung der Zufallsvariable von ihrem Erwartungswert
  - Mass für die Stärke der Streuung

$$Var(X) = E((X - E(X))^2)$$

Verschiebungssatz: $Var(X) = E(X^2) - E(X)^2$

Beispiel: Erwartungswert Normalverteilung

- Erwartungswert Normalverteilung

$$X \sim \mathcal{N}(x \mid \mu, \sigma^2) \qquad \mathcal{N}\left(x|\mu,\sigma^2\right) = \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left\{-\frac{1}{2\sigma^2}(x-\mu)^2\right\}$$

$$E(X) = \int_{-\infty}^{\infty} x\mathcal{N}(x \mid \mu, \sigma^2)dx \qquad \int_{-\infty}^{\infty} \mathcal{N}\left(x|\mu,\sigma^2\right)\,\mathrm{d}x = 1$$

$$= \int_{-\infty}^{\infty} x \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left(-\frac{1}{2\sigma^2}(x-\mu)^2\right) dx$$

$z = x - \mu$

$$= \int_{-\infty}^{\infty} (z+\mu) \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left(-\frac{1}{2\sigma^2}z^2\right) dz$$

$$= \mu \underbrace{\int_{-\infty}^{\infty} \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left(-\frac{1}{2\sigma^2}z^2\right) dz}_{=1} + \underbrace{\int_{-\infty}^{\infty} z \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left(-\frac{1}{2\sigma^2}z^2\right) dz}_{=0} = \mu$$

Bayessche Lernen

- Bayes‘sches Lernen: Anwendung probabilistischer Überlegungen auf Modelle, Daten, und Vorhersagen
- Bayes‘sche Regel: Modellwahrscheinlichkeit gegeben Daten und Vorwissen

Likelihood: Wie hoch ist die Wahrscheinlichkeit, bestimmte Daten zu sehen, unter der Annahme dass Modell das korrekte Modell ist?

$$p(Modell \mid Daten) = \frac{p(Daten \mid Modell)\, p(Modell)}{p(Daten)}$$

A-priori Verteilung über Modelle: Vorwissen

Posterior: wie ist die Wahrscheinlichkeit für Modelle, gegeben Evidenz der Trainingsdaten?

Wahrscheinlichkeit der Daten, unabhängig von Modell

Parameter von Verteilungen schätzen

- Erste Anwendung Bayes‘scher Überlegungen: Parameterschätzung in Münzwurfexperimenten

- Münze mit unbekanntem Parameter wird N Mal geworfen
- Daten L in Form von N_K Kopfwürfen und N_Z Zahlwürfen
- Was sagen uns diese Beobachtungen über den echten Münzwurfparameter θ?
- Wissen über echten Münzwurfparameter abgebildet in a-posteriori Verteilung $p(\theta \mid L)$

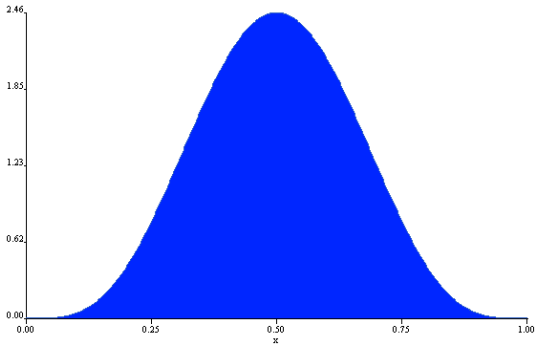
- Ansatz mit Bayesscher Regel:

$$p(\theta \mid L) = \frac{p(L \mid \theta)\, p(\theta)}{p(L)}$$

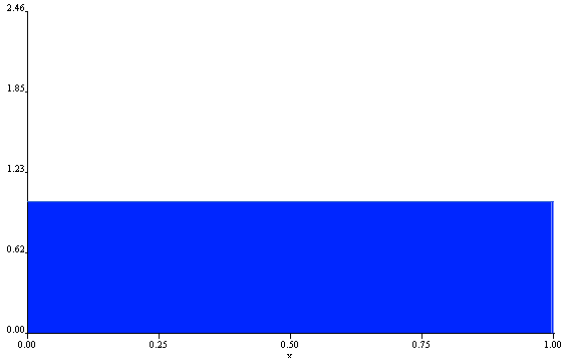
Parameterschätzungen von Wahrscheinlichkeitsverteilungen: Binomial

- Geeignete Prior-Verteilung: Beta-Verteilung
- Kontinuierliche Verteilung über θ
- Konjugierter Prior: a-posteriori Verteilung wieder Beta

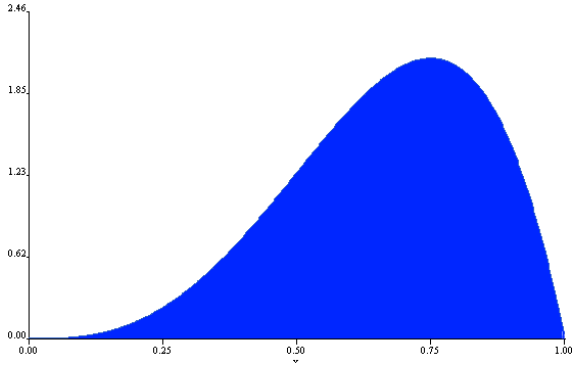
$\alpha_K = 5, \quad \alpha_Z = 5$



$\alpha_K = 1, \quad \alpha_Z = 1$



$\alpha_K = 4, \quad \alpha_Z = 2$

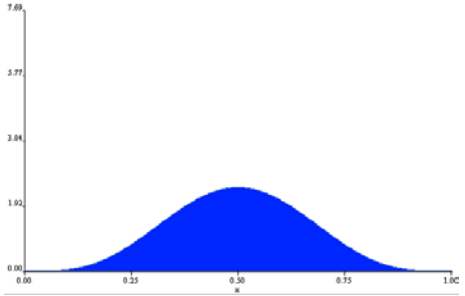


$$p(\theta) = Beta(\theta \mid \alpha_k, \alpha_z)$$
$$= \frac{\Gamma(\alpha_k + \alpha_z)}{\Gamma(\alpha_k)\Gamma(\alpha_z)} \theta^{\alpha_k - 1}(1-\theta)^{\alpha_z - 1}$$

$$\int_0^1 Beta(\theta \mid \alpha_K, \alpha_Z) d\theta = 1$$

Parameterschätzungen von Wahrscheinlichkeitsverteilungen: Binomial

- Mit Hilfe der Bayes‘schen Regel werden Vorwissen (Beta-Verteilung) und Beobachtungen (N_K , N_Z) zu neuem Gesamtwissen $P(\theta \mid L)$ integriert



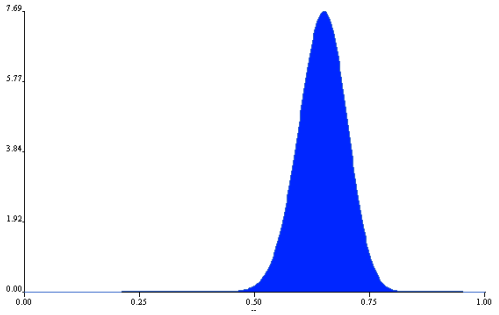
- A-priori-Verteilung Beta

$$P(\theta) = Beta(\theta \mid 5,5)$$

- Daten L: N_K=50x Kopf, N_z=25x Zahl

- Konjugierter Prior: A-posteriori Verteilung wieder Beta

$$P(\theta \mid L) = Beta(\theta \mid 55,30)$$



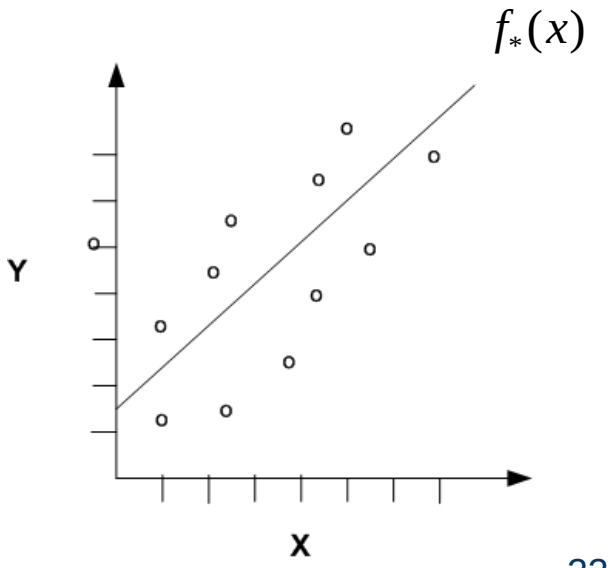
Bayessche Lineare Regression

- Bayes‘sche Lineare Regression: Bayes‘sches Modell für Regressionsprobleme
- Lineares Modell

$$f_{\mathbf{w}}(\mathbf{x}) = \mathbf{w}^T \mathbf{x} = w_0 + \sum_{i=1}^{m} w_i x_i$$

- Annahme über datengenerierenden Prozess: echtes lineares Modell $f_*(x)$ plus Gauß‘sches Rauschen

$$y_i = f_*(\mathbf{x}_i) + \varepsilon \text{ mit } \varepsilon \sim \mathcal{N}(\varepsilon \mid 0, \sigma^2)$$

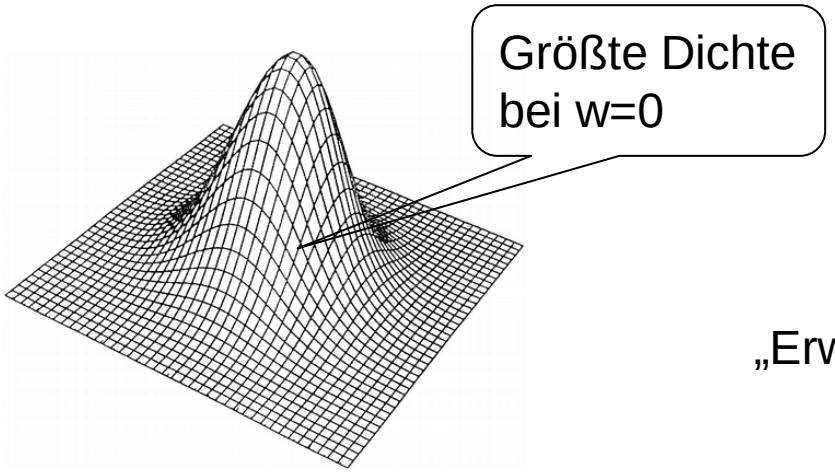


Bayessche Lineare Regression: Prior

- Ziel zunächst Berechnung der a-posteriori Verteilung mit Bayes‘scher Regel

$$P(\mathbf{w} \mid L) = \frac{P(L \mid \mathbf{w}) P(\mathbf{w})}{p(L)}$$

- Geeignete (konjugierte) Prior-Verteilung: Normalverteilung über Parametervektoren $\mathbf{w}$



„Erwarten kleine Attributgewichte“

Bayessche Lineare Regression: Posterior

- Posterior-Verteilung über Modelle gegeben Daten

$$P(\mathbf{w} \mid L) = \frac{1}{Z} P(L \mid \mathbf{w}) P(\mathbf{w})$$ Bayessche Regel

$$= \frac{1}{Z} \mathcal{N}(\mathbf{y} \mid X^{\mathrm{T}}\mathbf{w}, \sigma^2 I) \cdot \mathcal{N}(\mathbf{w} \mid 0, \Sigma_p)$$

$$= \mathcal{N}(\mathbf{w} \mid \bar{\mathbf{w}}, A^{-1})$$

mit $\bar{\mathbf{w}} = \sigma^{-2} A^{-1} X\mathbf{y}$ $\quad A = \sigma^{-2} XX^{\mathrm{T}} + \Sigma_p^{-1}$

- Posterior ist wieder normalverteilt, mit neuem Mittelwert $\bar{\mathbf{w}}$ und Kovarianzmatrix A^{-1}

Bayes‘sche Vorhersage

- Nicht auf MAP-Modell festlegen, solange noch Unsicherheit über Modelle besteht
- Stattdessen Bayes‘sche Vorhersage: berechne direkt wahrscheinlichstes Label für Testinstanz

Neue Testinstanz $\mathbf{x}$

Vorhersage $y_* = \arg\max_y p(y \mid \mathbf{x}, L)$

$$= \arg\max_y \int p(y \mid \mathbf{w}, \mathbf{x})\, p(\mathbf{w} \mid L)\, d\mathbf{w}$$

Bayesian Model Averaging

Vorhersage, gegeben Modell

Modell gegeben Trainingsdaten

- Bayes‘sche Vorhersage:
  - Mitteln der Vorhersage über alle Modelle.
  - Gewichtung: wie gut passt Modell zu Trainingsdaten.

Bayes‘sche Vorhersage für die Lineare Regression

- Für die Bayes‘sche lineare Regression kann die Bayes-optimale Vorhersage direkt berechnet werden:

$$P(y \mid \mathbf{x}, L) = \int P(y \mid \mathbf{x}, \mathbf{w}) P(\mathbf{w} \mid L) d\mathbf{w}$$

$$= \int \mathcal{N}(y \mid \mathbf{x}^T \mathbf{w}, \sigma^2) \mathcal{N}(\mathbf{w} \mid \bar{\mathbf{w}}, A^{-1}) d\mathbf{w}$$

$$= \mathcal{N}\left(y \mid \mathbf{x}^T \bar{\mathbf{w}}, \mathbf{x}^T A^{-1} \mathbf{x}\right)$$

$$\text{mit } \bar{\mathbf{w}} = \sigma^{-2} A^{-1} X \mathbf{y} \qquad A = \sigma^{-2} X X^{\mathrm{T}} + \Sigma_p^{-1}$$

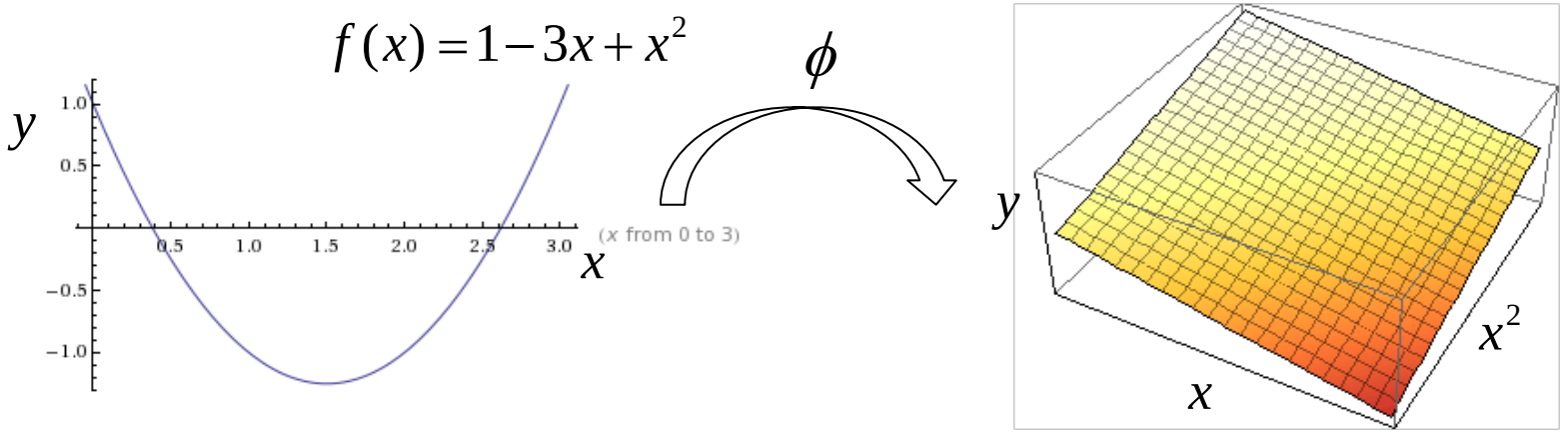
- Vorhersageverteilung wieder normalverteilt

Lineare Regression auf nichtlinearen Basisfunktionen

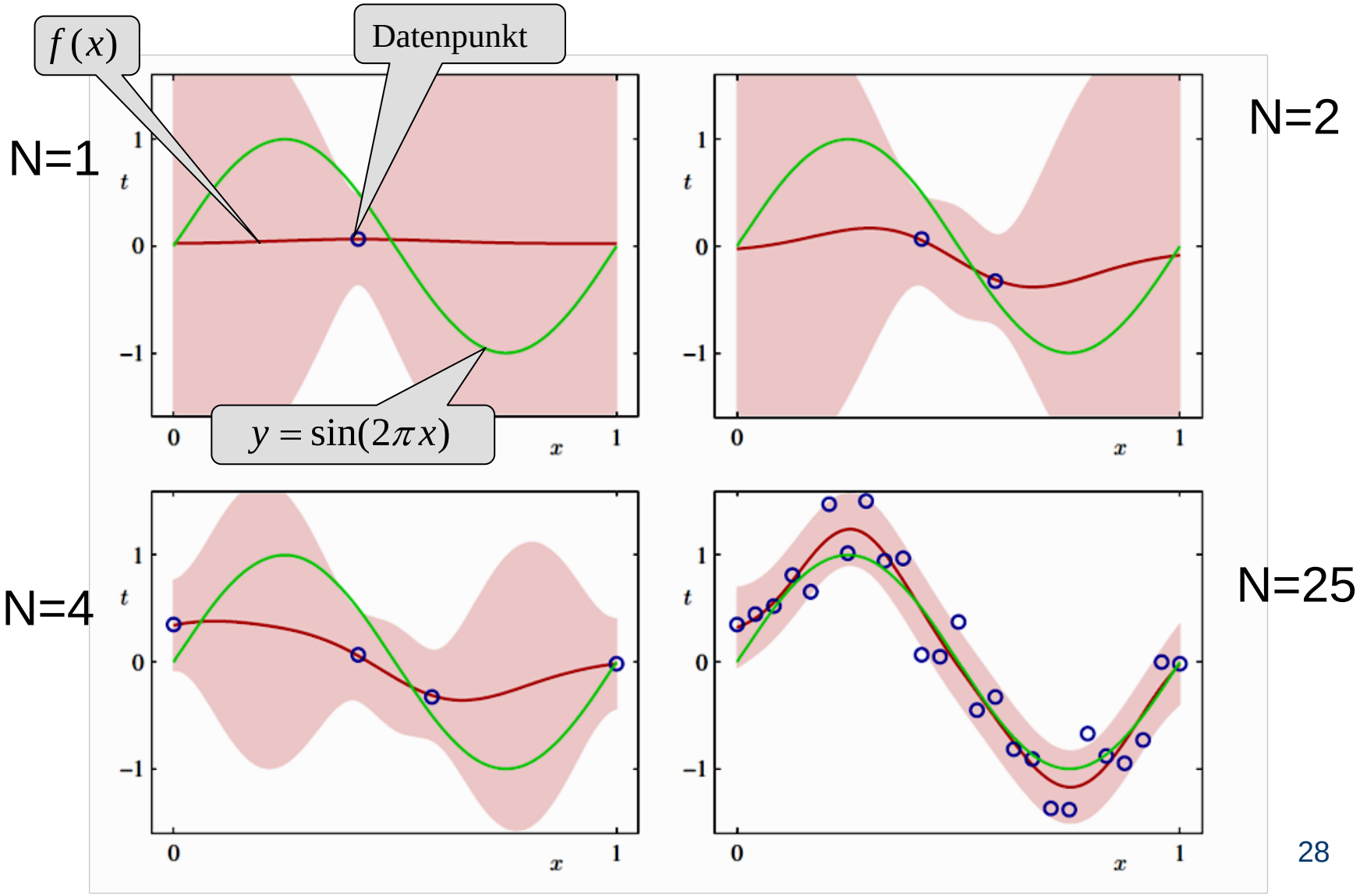
- Nichtlineare Zusammenhänge in Daten darstellbar durch lineare Regression in nichtlinearen Basisfunktionen

$$f(\mathbf{x}) = \mathbf{w}^T \phi(\mathbf{x}) = w_0 + \sum_{i=1}^{d} w_i \phi_i(\mathbf{x})$$

  - ϕ : Abbildung von $\mathcal{X}$ in höherdimensionalen Raum $\phi(\mathcal{X})$
  - Lineare Regression in $\phi(\mathcal{X})$ entspricht nichtlinearer Regression in $\mathcal{X}$.

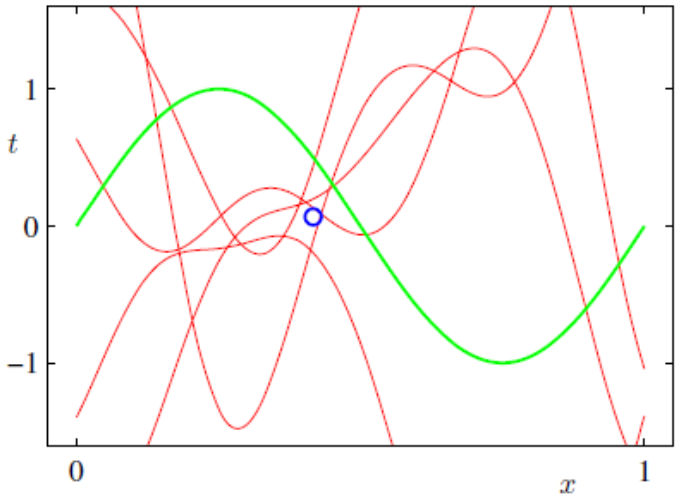


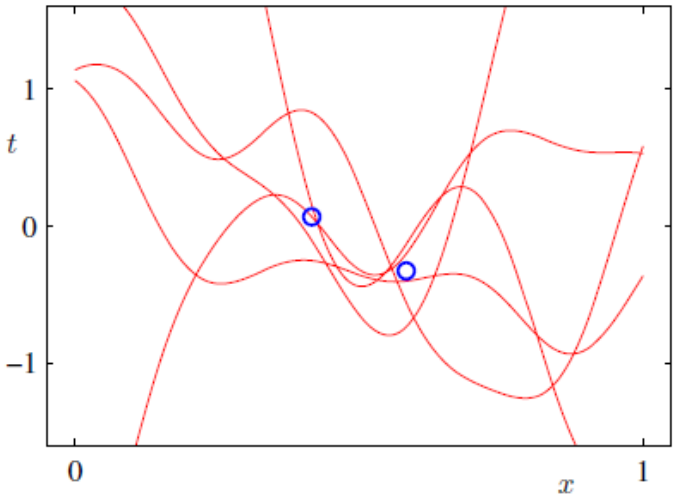
Beispiel nichtlineare Regression: Vorhersageverteilung



Beispiel nichtlineare Regression: Samples aus dem Posterior

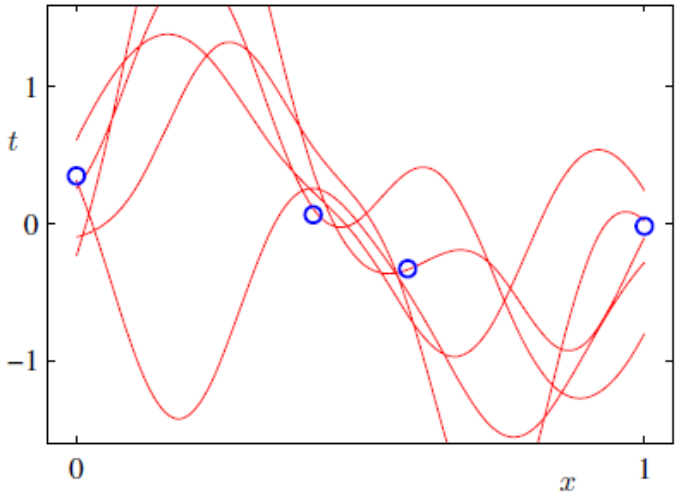
N=1

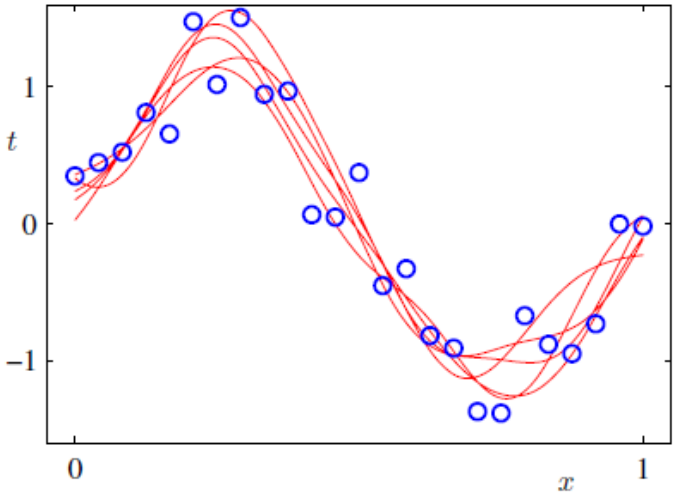




N=2

N=4





N=25

Naive Bayes Klassifikator

- Naive Bayes Klassifikator als einfaches probabilistisches Klassifikationsmodell

- Keine Bayes‘sche Vorhersage möglich, aber Berechnung des MAP-Modells

$$\theta_{MAP} = \arg\max_{\theta} P(\theta) P(\mathbf{y}, X \mid \theta)$$

Naive Bayes Klassifikator

- Naive Unabhängigkeitsannahme

$$P(\mathbf{x} \mid y, \theta) = \prod_{i=1}^{m} P(x_i \mid y, \theta)$$

„Attribute x_i unabhängig gegeben die Klasse y“

- Likelihood zerfällt in Produkt von Bernoulliverteilungen

$$P(L \mid \theta) = \prod_{j=1}^{N} P(\mathbf{x}_{j,} y_j \mid \theta)$$

Unabhängigkeit Instanzen

$$= \prod_{j=1}^{N} P(y_j \mid \theta) P(\mathbf{x}_j \mid y_j, \theta)$$

Produktregel

$$= \prod_{j=1}^{N} P(y_j \mid \theta^y) \prod_{i=1}^{m} P(x_{ji} \mid y_j, \theta^{x_i \mid y_j})$$

Unabhängigkeit Attribute

Münzwurf für Klasse

Münzwurf für Attribut, gegeben Klasse

- Konjugierter Prior Betaverteilung, Lösung für MAP Parameter wie bei Münzwurfexperimenten

Naive Bayes: Lernalgorithmus

- Eingabe: $L = \langle (\mathbf{x}_1, y_1), ..., (\mathbf{x}_N, y_N) \rangle$
- Schätze Klassenverteilung:

Zähle

N_1: Anzahl Beispiele mit Klasse 1 in L

N_0: Anzahl Beispiele mit Klasse 0 in L

$$\theta^{y}_{MAP} = \frac{N_1 + \alpha_1 - 1}{N_0 + \alpha_0 + N_1 + \alpha_1 - 2}$$

Standardlösung für Münzwürfe

- Für Klassen *y=0* und *y=1*, für alle Attribute *i*:

Zähle

$N_{x_i|y}$: Anzahl Beispiele mit $x_i = 1$ und Klasse y in L

$N_{\bar{x}_i|y}$: Anzahl Beispiele mit $x_i = 0$ und Klasse y in L

$$\theta^{x_i|y}_{MAP} = \frac{N_{x_i|y} + \alpha_{x_i|y} - 1}{N_{x_i|y} + \alpha_{x_i|y} + N_{\bar{x}_i|y} + \alpha_{\bar{x}_i|y} - 2}$$

Standardlösung für Münzwürfe

- Alle Modellparameter gelernt!

Naive Bayes: Beispiel

- Gelernte Parameter/Hypothese
  - Merkmalsverteilungen $P(x_i \mid y)$

x_1	$P(x_1 \mid y=0)$
0	2/3
1	1/3

x_1	$P(x_1 \mid y=1)$
0	1/4
1	3/4

x_2	$P(x_2 \mid y=0)$
0	1/3
1	2/3

x_2	$P(x_2 \mid y=1)$
0	2/4
1	2/4

  - Klassenprior $P(y)$

y	$P(y)$
0	2/5
1	3/5

Naive Bayes: Eigenschaften

- Einfach zu implementieren, effizient, populär.
- Funktioniert ok, wenn die Attribute wirklich unabhängig sind.
- Das ist aber häufig nicht der Fall.
- Unabhängigkeitsannahme und modellbasiertes Training führen häufig zu schlechten Ergebnissen.
- Logistische Regression, Winnow, Perzeptron sind meist besser.

Überblick

- Lernprobleme
- Bayes‘sches Lernen
- Evaluierung

Hypothesenbewertung

- Problem der Risikoschätzung für gelernte Hypothesen
- Risiko einer Hypothese: erwarteter Verlust auf Testbeispielen

$$R(f) = E[\ell(y, f(\mathbf{x}))] = \int \ell(y, f(\mathbf{x}))\, p(\mathbf{x}, y) d\mathbf{x} d$$

$$(\mathbf{x}, y) \sim p(\mathbf{x}, y) \qquad \text{Testbeispiel}$$

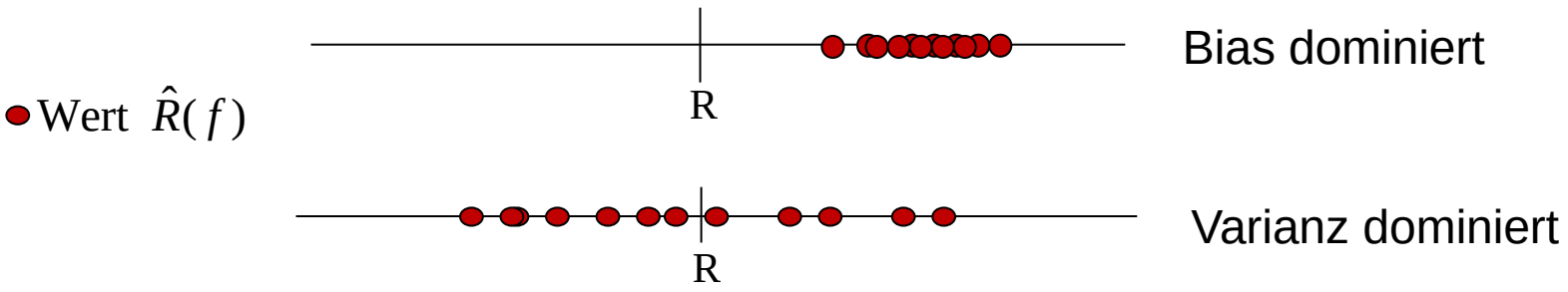
- Risiko lässt sich aus Testdaten schätzen

$$\hat{R}(f) = \frac{1}{m} \sum_{j=1}^{m} \ell(y_j, f(\mathbf{x}_j)) \qquad \text{"Risikoschätzer, empirisches Risiko"}$$

$$T = \left\langle (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m) \right\rangle \qquad (\mathbf{x}_i, y_i) \sim p(\mathbf{x}, y)$$

Der Risikoschätzer als Zufallsvariable

- Risikoschätzer ist Zufallsvariable
- Charakterisiert durch Bias und Varianz: systematischer Fehler und zufällige Streuung



- Fehler des Schätzers lässt sich zerlegen in Bias und Varianz

$$
\begin{aligned}
E[(\hat{R}(f)-R)^2] &= E[\hat{R}(f)^2 - 2R\hat{R}(f) + R^2] \\
&= E[\hat{R}(f)^2] - 2RE[\hat{R}(f)] + R^2 \\
&= E[\hat{R}(f)]^2 - 2RE[\hat{R}(f)] + R^2 + E[\hat{R}(f)^2] - E[\hat{R}(f)]^2 \\
&= (E[\hat{R}(f)]-R)^2 + Var[\hat{R}(f)] \\
&= Bias[\hat{R}(f)]^2 + Var[\hat{R}(f)]
\end{aligned}
$$

Risikoschätzung auf Trainingsdaten?

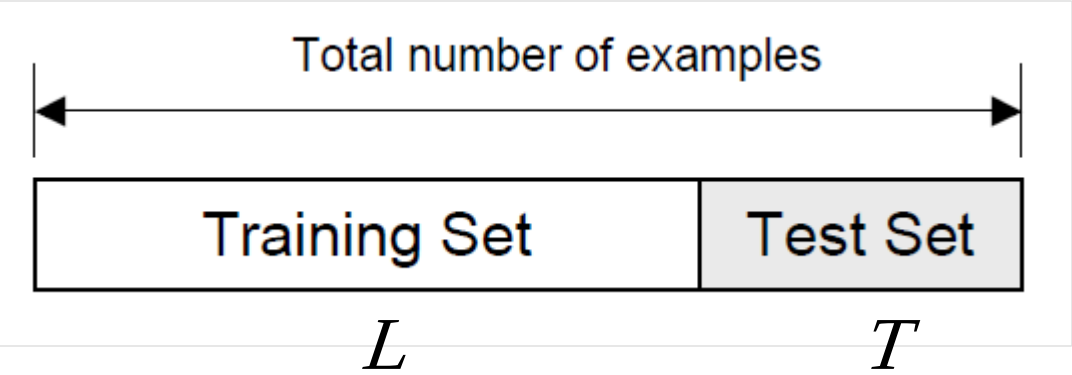
- Risikoschätzung kann nicht auf Trainingsdaten erfolgen: optimistischer Bias

$$E[\hat{R}_L(f)] < R(f)$$

- Problem ist die Abhängigkeit von gewählter Hypothese und zur Fehlerschätzung verwendeten Daten
- Ansatz: Testdaten verwenden, die von den Trainingsdaten unabhängig sind.

Holdout-Testing

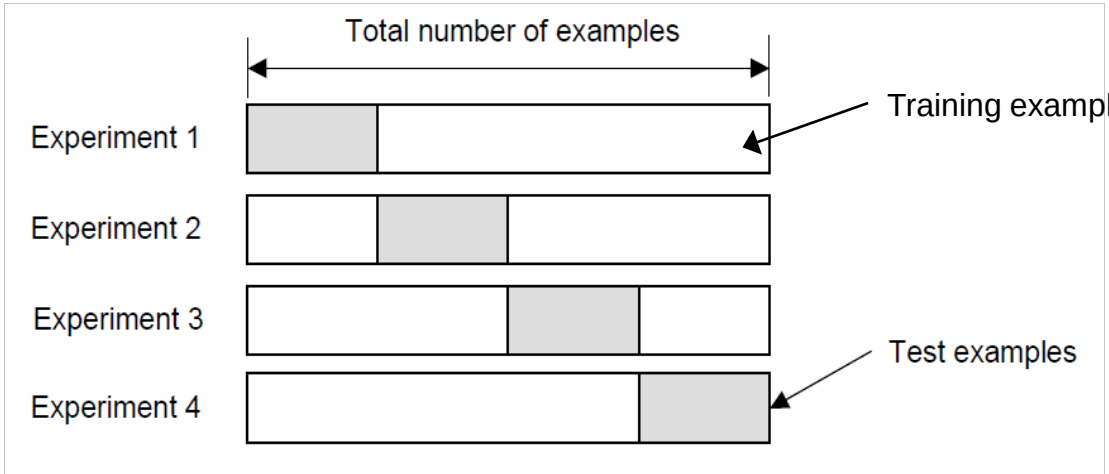
- Gegeben Daten $D = \langle (\mathbf{x}_1, y_1), ..., (\mathbf{x}_d, y_d) \rangle$
- Teile Daten auf in Trainingsdaten $L = \langle (\mathbf{x}_1, y_1), ..., (\mathbf{x}_m, y_m) \rangle$ und Testdaten $T = \langle (\mathbf{x}_{m+1}, y_{m+1}), ..., (\mathbf{x}_d, y_d) \rangle$
- Starte Lernalgorithmus mit Daten L, gewinne so Hypothese f_L.
- Ermittle empirisches Risiko $\hat{R}_T(f_L)$ auf Daten T.
- Starte Lernalgorithmus auf Daten D, gewinne so Hypothese f_D.
- Ausgabe: Hypothese f_D , benutze $\hat{R}_T(f_L)$ als Schätzer für das Risiko von f_D



Cross-Validation

- Gegeben: Daten $D = \langle (\mathbf{x}_1, y_1), ..., (\mathbf{x}_d, y_d) \rangle$
- Teile D in n gleich große Blöcke $D_1, ..., D_n$
- Wiederhole für i=1..n
  - Trainiere f_i mit $L_i = D \setminus D_i$.
  - Bestimme empirisches Risiko $\hat{R}_{D_i}(f_i)$ auf D_i.
  - Fehlerschätzung $\overline{R} = \frac{1}{n} \sum_{i=1}^{n} \hat{R}_{D_i}(f_i)$

D_1	D_2	D_3	D_4



Bias und Varianz in Holdout-Testing und Cross-Validation

- Bias und Varianz der Fehlerschätzung aus Holdout-Testing und Cross-Validation?
- Fehlerschätzungen aus Holdout-Testing und Cross-Validation jeweils leicht pessimistisch
- Aber im Gegensatz zum Trainingsfehler in der Praxis brauchbar
- Cross-Validation hat geringere Varianz als Holdout-Testing, weil wir über mehrere Holdout-Experimente mitteln

Überblick

- Lernprobleme
- Bayes‘sches Lernen
- Evaluierung

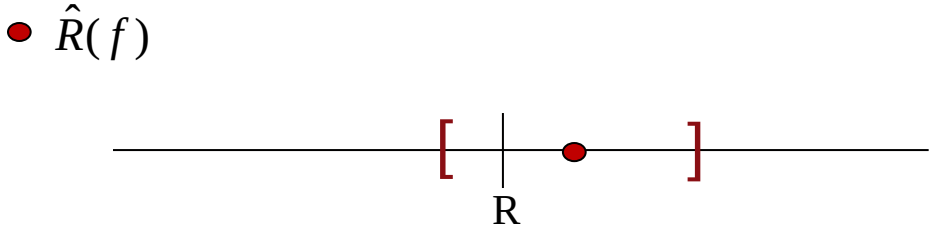
Überblick

- Hypothesenbewertung
  - Konfidenzintervalle
  - ROC-Analyse
- Entscheidungsbäume
- Lineare Klassifikatoren

Konfidenzintervalle

- Idee Konfidenzintervall:
  - Intervall um den geschätzten Fehler $\hat{R}(f)$ angeben, so dass der echte Fehler „meistens“ im Intervall liegt
  - Quantifiziert Unsicherheit der Schätzung

- Weg zum Konfidenzintervall: Analyse der Verteilung der Zufallsvariable $\hat{R}(f)$



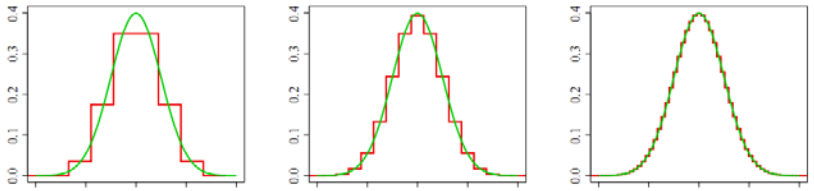
Normalverteilung

- Empirisches Risiko annähernd normalverteilt:

$$\hat{R}_T(f) \sim N\left(\hat{R}_T(f) \mid r, \sigma_{\hat{r}}^2\right)$$

[approximativ, für große m]

$$\sigma_{\hat{r}}^2 = \frac{r(1-r)}{m}$$



- Für die weitere Analyse betrachten wir das standardisierte Risiko, dieses ist standardnormalverteilt:

$$\frac{\hat{R}_T(f) - R(f)}{\sigma_{\hat{r}}} \sim N\left(\frac{\hat{R}_T(f) - R(f)}{\sigma_{\hat{r}}} \mid 0,1\right)$$

[approximativ, für große m]

- Schätzen der Varianz des empirischen Risikos:

$$\sigma_{\hat{r}}^2 \approx s_{\hat{r}}^2 \qquad s_{\hat{r}}^2 = \frac{\hat{r}(1-\hat{r})}{m-1}, \qquad \hat{r} = \hat{R}_T(f)$$

Konfidenzintervalle

- Ermitteln des einseitigen 1-δ-Konfidenzintervalls:

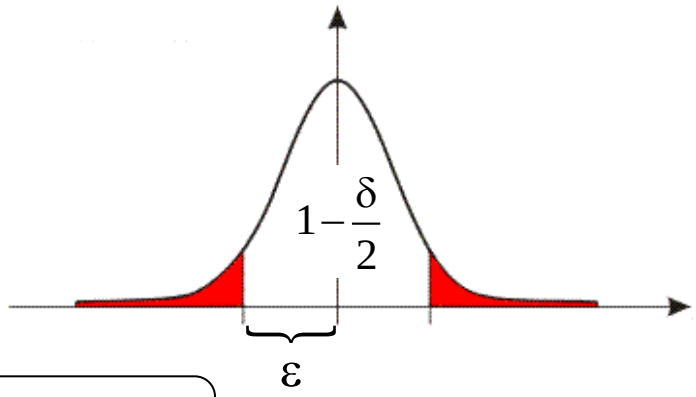
$$P\left(R(f) \le \hat{R}_T(f) + \varepsilon\right) \ge 1 - \delta$$

$$\tilde{\Leftrightarrow} \; \Phi\left(\frac{\varepsilon}{s_{\hat{r}}}\right) \ge 1 - \delta$$

$$\Leftrightarrow \frac{\varepsilon}{s_{\hat{r}}} \ge \Phi^{-1}(1-\delta)$$

$$\Leftrightarrow \varepsilon \ge s_{\hat{r}} \Phi^{-1}(1-\delta)$$

$$\Phi(z) = \int_{-\infty}^{z} \frac{1}{\sqrt{2\pi}} \exp\left(-x^2/2\right) dx$$



- Konfidenzintervall ist $[\hat{R}_T(f) - \varepsilon, \hat{R}_T(f) + \varepsilon]$ (Konfidenzlevel 1-2δ)

ROC-Analyse

- Entscheidungsfunktion + Schwellwert = Klassifikator

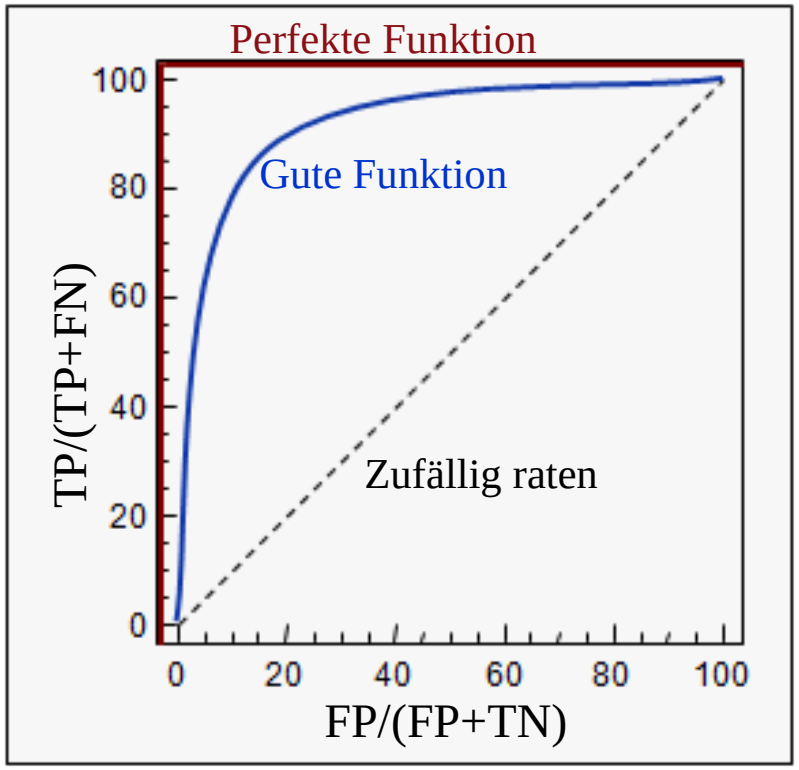
$$\text{Vorhersage} = \begin{cases} +1 : f(\mathbf{x}) \geq \theta \\ -1 : \text{sonst} \end{cases}$$

  - Fehler hängen vom Schwellwert ab
  - Großer Schwellwert: Mehr positive Bsp falsch.
  - Kleiner Schwellwert: Mehr negative Bsp falsch.

- ROC-Analyse: Bewertung der Entscheidungsfunktion unabhängig vom konkreten Schwellwert.
- Charakterisieren das Verhalten des Klassifikators für alle möglichen Schwellwerte.

ROC-Kurven

- Rate der „False Positives“ und „True Positives“ in Abhängigkeit des Schwellwertes
  - X-Achse: „False Positive Rate“
  - Y-Achse: „True Positive Rate“
- Theorem: AUC = $P(f(x_+) > f(x_-))$.



	f(x) = +1	f(x) = -1
y = +1	TP	FN
y = -1	FP	TN

Precision / Recall

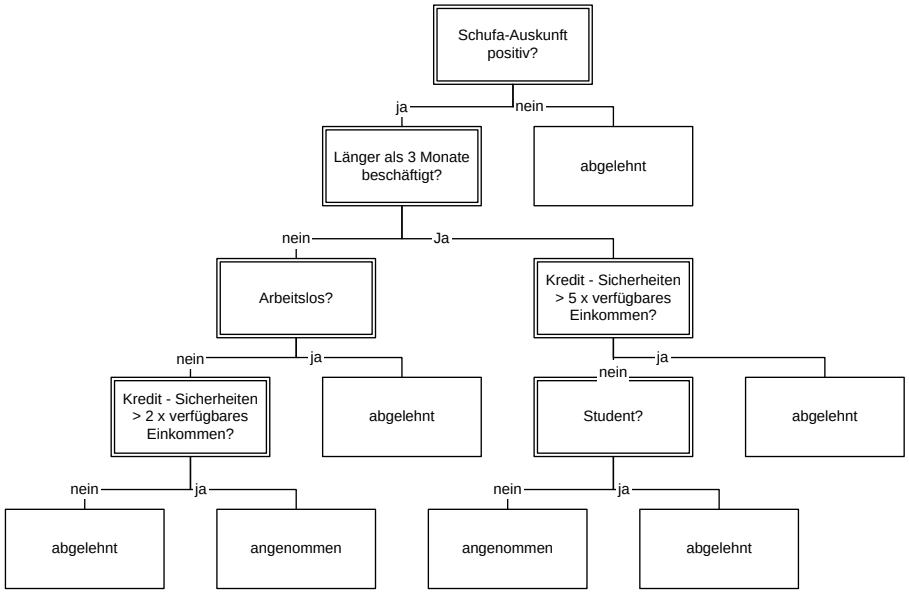
- Evaluierungsmaße für schiefe Klassenverteilung:
  - $\text{Precision} = \frac{TP}{TP + FP}$ ← Alle Instanzen mit Vorhersage „+“
  - $\text{Recall} = \frac{TP}{TP + FN}$ ← Alle Instanzen mit echtem Label „+“
  - $\text{F-measure} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$
  - Breakeven-Point

Überblick

- Hypothesenbewertung
  - Konfidenzintervalle
  - ROC-Analyse
- Entscheidungsbäume
- Lineare Klassifikatoren

Entscheidungsbäume

- Testknoten: Testen, ob der Wert eines Attributs eine Bedingung erfüllen; bedingte Verzweigung in einen Ast.
- Terminalknoten: Liefern einen Wert als Ausgabe.



Anwendung von Entscheidungsbäumen

- Entscheidungsbäume sinnvoll, wenn:
  - Instanzen durch Attribut-Werte-Paare beschrieben werden.
  - Zielattribut hat einen diskreten Wertebereicht.
    - Bei Erweiterung auf Modellbäume auch kontinuierlicherer Wertebereich möglich.
  - Interpretierbarkeit der Vorhersage gewünscht ist.

- Anwendungsgebiete:
  - Medizinische Diagnose
  - Kreditwürdigkeit

Lernen von Entscheidungsbäumen

- Eleganter Weg: Unter den Bäumen, die mit den Trainingsdaten konsistent sind, wähle einen möglichst kleinen Baum (möglichst wenige Knoten).
- Kleine Bäume sind gut, weil:
  - sie leichter zu interpretieren sind;
  - sie in vielen Fällen besser generalisieren.
  - Es gibt mehr Beispiele pro Blattknoten. Die Klassenentscheidungen in den Blättern stützen sich so auf mehr Beispiele.

Lernen von Entscheidungsbäumen

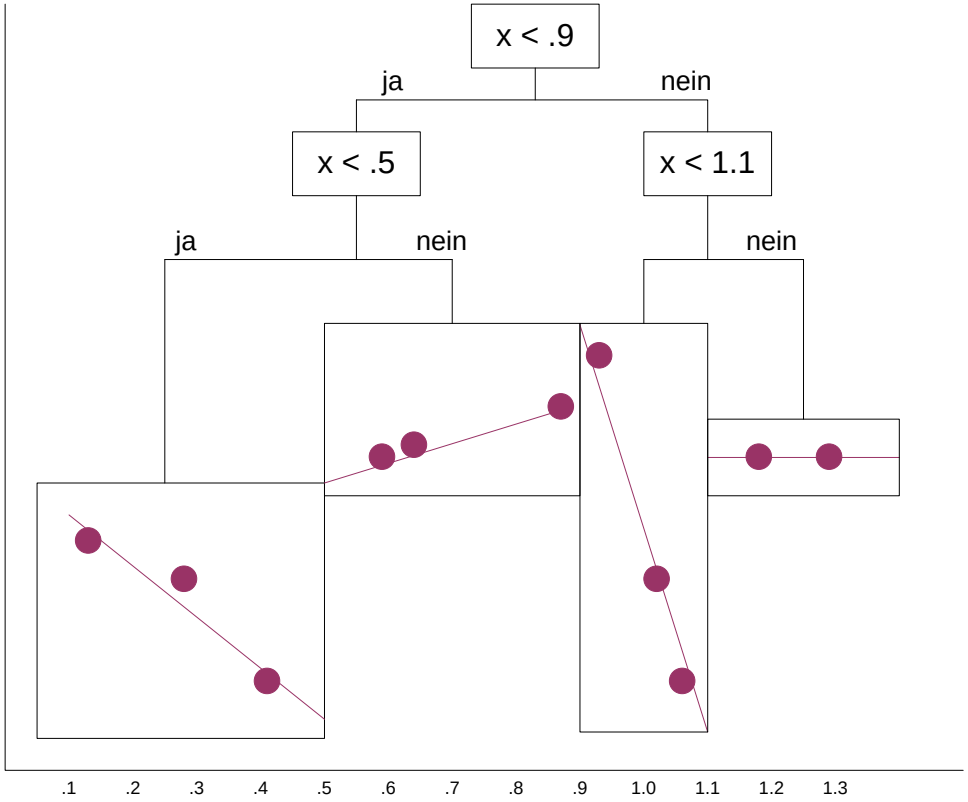
- Idee: rekursiver Algorithmus.
  - Wähle das Attribut, welches die Unsicherheit / Fehler bzgl. der Zielklasse maximal verringert.
  - Aufteilen der Trainingsdaten nach diesem Kriterium und rekursiver Aufruf für Teilbaum.

- Algorithmen
  - Klassifikation: ID3, C4.5, SLIQ
  - Regression: CART

- Verschiedene Erweiterungen:
  - Fehlende Attribute
  - Attribute mit Kosten

Modellbäume

- Entscheidungsbaum, aber lineares Regressionsmodell in Blattknoten.



Pruning mit Schwellwert

- Für alle Blattknoten: Wenn weniger als r Trainingsbeispiele in den Blattknoten fallen
  - Entferne darüberliegenden Testknoten.
  - Erzeuge neuen Blattknoten, sage Mehrheitsklasse vorher.

- Regularisierungsparameter r.
- Einstellung mit Cross Validation.

Entscheidungsbaum: Pro / Kontra

- Vorteile:
  - Einfach zu interpretieren.
  - Können effizient aus vielen Beispielen gelernt werden.

- Nachteile:
  - Nicht robust gegenüber Rauschen
  - Tendenz zum Overfitting
  - Instabil

Überblick

- Hypothesenbewertung
  - Konfidenzintervalle
  - ROC-Analyse
- Entscheidungsbäume
- Lineare Klassifikatoren

Lineare Klassifikatoren

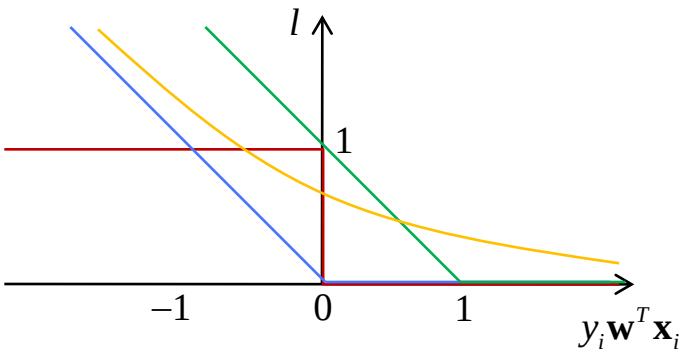
- Rocchio: $\mathbf{w}^* = \bar{\mathbf{x}}_{+1} - \bar{\mathbf{x}}_{-1}$
- Fisher-Diskr.: $\mathbf{w}^* = \mathbf{S}_W^{-1}(\bar{\mathbf{x}}_{+1} - \bar{\mathbf{x}}_{-1})$
- Perceptron: $\mathbf{w}^* \in \arg\min_{\mathbf{w}} \sum_i \max\left(0, -y_i \mathbf{w}^T \mathbf{x}_i\right)$
- SVM: $\mathbf{w}^* = \arg\min_{\mathbf{w}} \sum_i \max\left(0, 1 - y_i \mathbf{w}^T \mathbf{x}_i\right) + C\|\mathbf{w}\|^2$
- LogReg: $\mathbf{w}^* = \arg\min_{\mathbf{w}} \sum_i \log\left(1 + \exp\left(-y_i \mathbf{w}^T \mathbf{x}_i\right)\right) + C\|\mathbf{w}\|^2$

Geschlossene Lösung

Wenn linear-separierbar

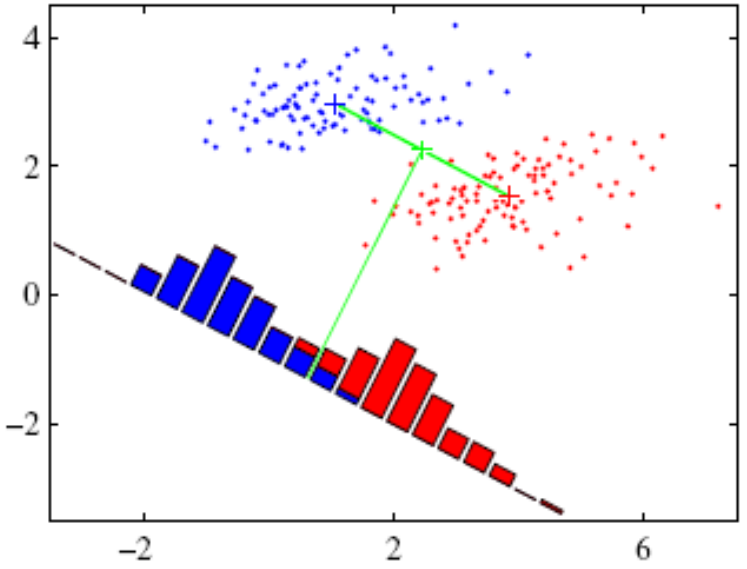
Lösung ist sparse

$\sigma\left(-y_i \mathbf{w}^{*T} \mathbf{x}_i\right)$ ist kalibrierte Fehlerwahrscheinlichkeit



Rocchio

- Trennebenen hat maximalen Abstand von den Mittelpunkten der Klassen.
- Trainingsbeispiele können falsch klassifiziert werden.
- **Problem:** Differenz der Mittelwerte kann schlechter Normalenvektor für Diskrimination sein.



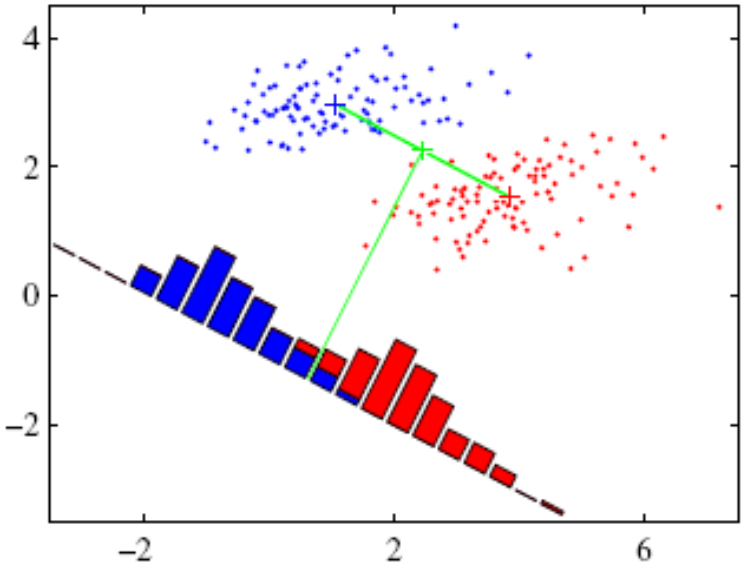
Fisher-Diskriminante

- Fisher-Optimierungskriterium:

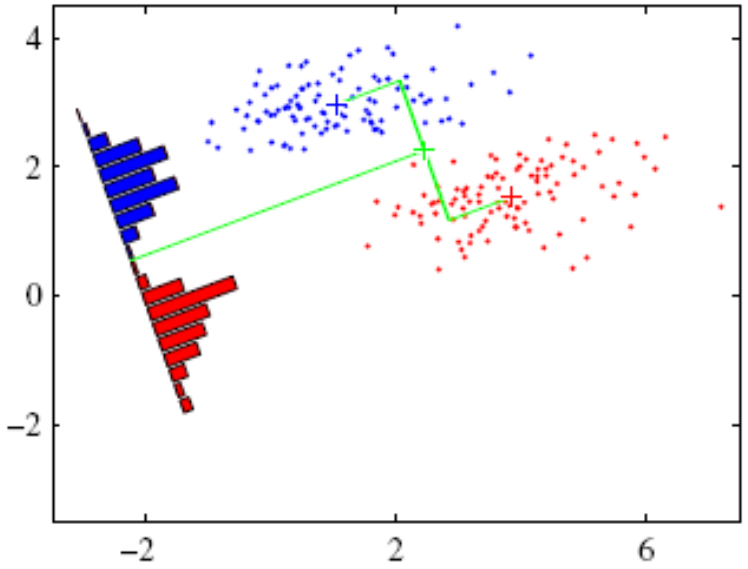
$$J(\mathbf{w}) = \frac{\mathbf{w}^{\mathrm{T}}\mathbf{S}_B\mathbf{w}}{\mathbf{w}^{\mathrm{T}}\mathbf{S}_W\mathbf{w}}$$

$f(\mathbf{x})$ für Mittelwerte möglichst unterschiedlich

Geringe Varianz von $f(\mathbf{x})$ innerhalb der Klassen



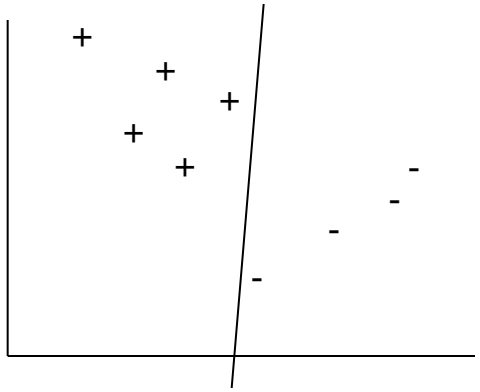
Rocchio



Fisher

Perzeptron

- Lineares Modell:
  - $f(\mathbf{x}) = \mathbf{w}^{\mathrm{T}}\mathbf{x}$
- Perzeptron-Optimierungskriterium:
  - $J_P(\mathbf{w}) = \sum_{(\mathbf{x}_i, y_i) \in L} \max\left\{-y_i \mathbf{w}^{\mathrm{T}}\mathbf{x}_i, 0\right\}$
- Subgradient für Beispiel $(\mathbf{x}_i, y_i)$:
  - $\nabla_i J_P(\mathbf{w}) = \begin{cases} 0, & \text{wenn } y_i \mathbf{w}^{\mathrm{T}}\mathbf{x}_i > 0 \\ -y_i \mathbf{x}_i & \text{sonst} \end{cases}$
- Gradientenabstieg: Wiederhole, für alle Beispiele mit $y_i \mathbf{w}^{\mathrm{T}}\mathbf{x}_i \leq 0$
  - $\mathbf{w} \leftarrow \mathbf{w} + y_i \mathbf{x}_i$



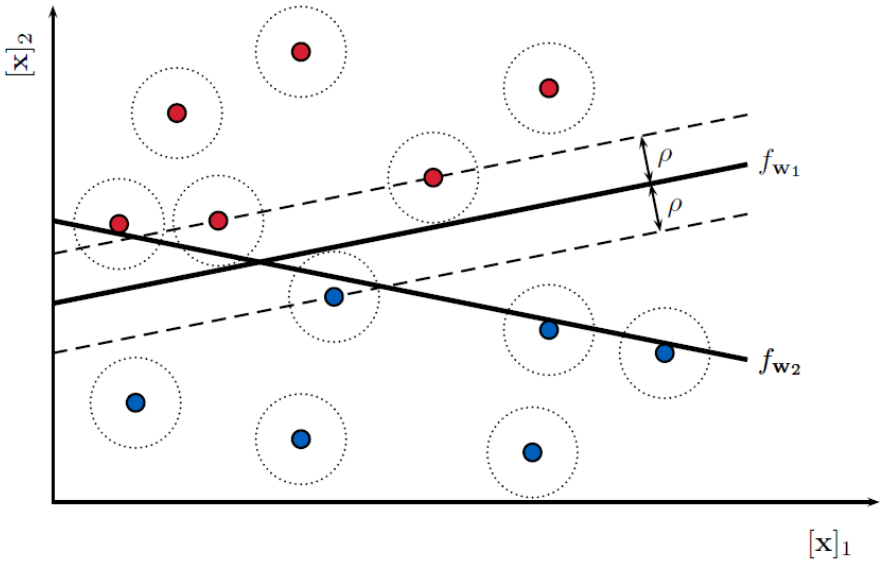
Support Vector Machine (hard-margin)

- Finde Ebene, die alle Beispiele mindestens ρ von Ebene entfernt: $y_i \left(\frac{\mathbf{w}^{\mathrm{T}}}{|\mathbf{w}|} \mathbf{x}_i \right) > \rho$
- Für den größtmöglichen Wert ρ .

- Optimierungsproblem:

$$\min_{\mathbf{w}} \frac{1}{2} \|\mathbf{w}\|^2$$

$$\text{wobei } \forall i : y_i \left(\mathbf{w}^{\mathrm{T}} \mathbf{x}_i \right) > 1$$

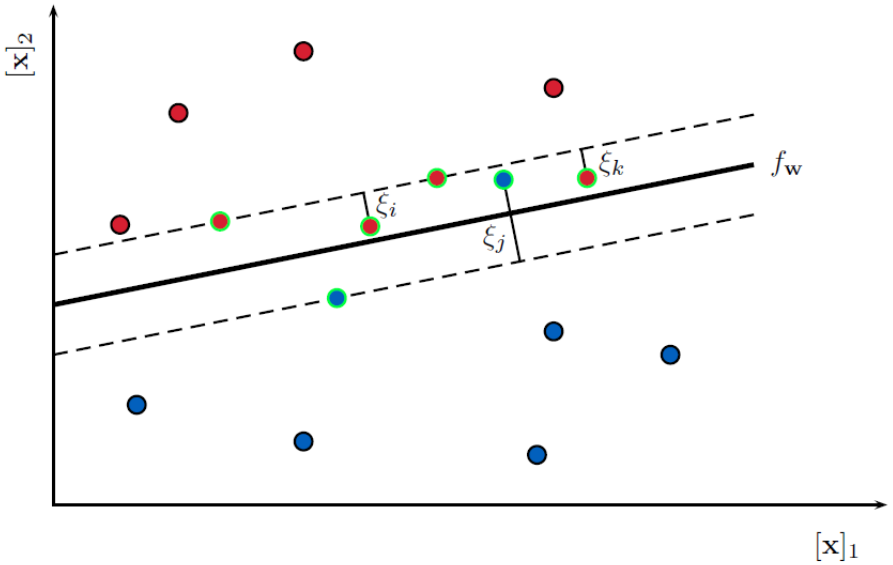


Support Vector Machine (soft-margin)

- Finde Ebene, die alle Beispiele mindestens ρ von Ebene entfernt: $y_i \left(\frac{\mathbf{w}^{\mathrm{T}}}{|\mathbf{w}|} \mathbf{x}_i \right) > \rho$
- Für den größtmöglichen Wert ρ .
- Zusätzlich: Fehlerterm ξ für Marginverletzung
- Optimierungsproblem:

$$\min_{\mathbf{w}} \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_i \xi_i$$

$$\text{wobei } \forall i : y_i \left(\mathbf{w}^{\mathrm{T}} \mathbf{x}_i \right) > 1 - \xi_i$$



Logistische Regression

- SVM: großer Entscheidungsfunktionswert ~ hohe Sicherheit der Vorhersage.
- Aber: beim Lernen nicht auf korrekte Kalibrierung der Klassenwahrscheinlichkeiten optimiert.
- $f(\mathbf{x})=18.3 \rightarrow$ Risiko eines Fehlers?

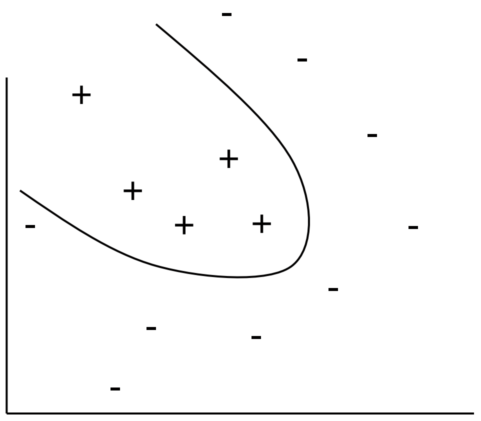
- Logistische Regression: Vorhersage der Klassenwahrscheinlichkeit.

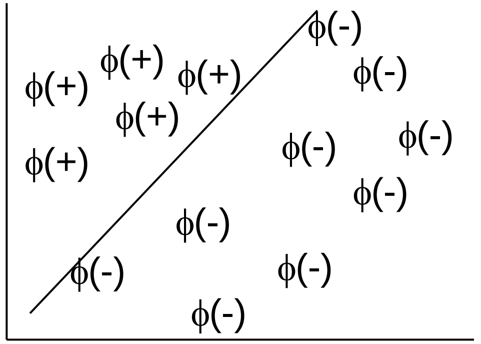
Logistische Regression

- Prior über Parameter:
  - Normalverteilung, $\mathbf{w} \sim \mathcal{N}\left(\mathbf{0}, \tau^2 \mathbf{I}\right)$.
- Posterior:
  - $$P(\mathbf{w} \mid L) \propto \prod_{i=1}^{N} p(y_i \mid \mathbf{x}_i, \mathbf{w}) p(\mathbf{w})$$
    $$= \prod_{i=1}^{N} \sigma(\mathbf{w}^{\mathrm{T}} \mathbf{x}_i)^{[[y_i=+1]]} (1-\sigma(\mathbf{w}^{\mathrm{T}} \mathbf{x}_i))^{[[y_i=-1]]} p(\mathbf{w})$$
- Verlustfunktion+Regularisierer:
  - $$E(\mathbf{w}, L) = -\log p(\mathbf{w} \mid L)$$
    $$= -\sum_{i=1}^{N} [[y_i = +1]] \log \sigma(\mathbf{w}^{\mathrm{T}} \mathbf{x}_i) + [[y_i = -1]] \log(1-\sigma(\mathbf{w}^{\mathrm{T}} \mathbf{x}_i)) - \frac{\mathbf{w}^{\mathrm{T}} \mathbf{w}}{2\tau^2}$$

Kernel

- Lineare Klassifikatoren:
  - Oft adäquat, aber nicht immer.
- Idee: Beispiele in anderen Raum abbilden, in dem sie linear klassifizierbar sind.
- Abbildung
  - $\mathbf{x} \mapsto \varphi(\mathbf{x})$
- Zugehöriger Kernel
  - $k(\mathbf{x}_i, \mathbf{x}_j) = \varphi(\mathbf{x}_i)^{\mathrm{T}} \varphi(\mathbf{x}_j)$
- Kernel = Inneres Produkt = Ähnlichkeit der Beispiele.





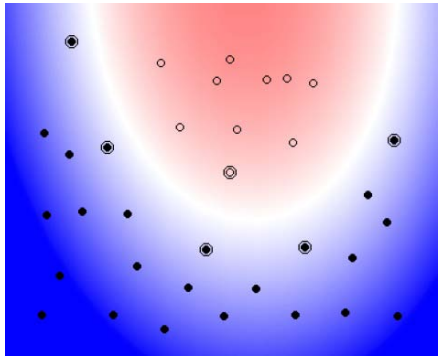
Representer Theorem

- Gegeben $L = \langle (\mathbf{x}_1, y_1), ..., (\mathbf{x}_N, y_N) \rangle$
- Abbildung $\varphi(\mathbf{x})$ mit dazu analogem Kernel $k(\mathbf{x}_i, \mathbf{x}_j) = \varphi(\mathbf{x}_i)^{\mathrm{T}} \varphi(\mathbf{x}_j)$
- Trennebenen $f(\mathbf{x}) = \mathbf{w}^{\mathrm{T}} \varphi(\mathbf{x}) + w_0$
- Die Trennebene $\mathbf{w}^*$, die $|\mathbf{w}| + C \sum_i \max\{0, 1 - y_i(\mathbf{w}^{\mathrm{T}} \varphi(\mathbf{x}_i) + w_0)\}$ minimiert, lässt sich repräsentieren als
  - $f(\mathbf{x}) = \mathbf{w}^{*\mathrm{T}} \varphi(\mathbf{x}) + w_0 = \sum_{j=1}^{N} \alpha_j y_j k(\mathbf{x}, \mathbf{x}_j) + w_0$
- Primale Sicht:
  - Hypothese $f(\mathbf{x}) = \mathbf{w}^{\mathrm{T}} \varphi(\mathbf{x}) + w_0$ so viele Parameter $\mathbf{w}_i$ wie Dimensionen hat.
- Duale Sicht:
  - Hypothese $f(\mathbf{x}) = \sum_{i=1}^{N} \alpha_j y_j k(\mathbf{x}, \mathbf{x}_j) + w_0$ hat so viele Parameter α_j wie Beispiele existieren.

Kernel-Funktionen

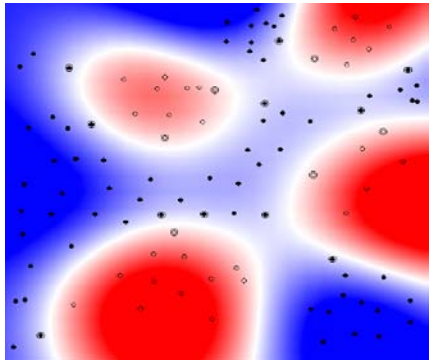
- Polynomielle Kernels

$$k_{\text{poly}}(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i^{\mathrm{T}} \mathbf{x}_j + 1)^d$$



- Radiale Basisfunktion

$$k_{\text{RBF}}(\mathbf{x}_i, \mathbf{x}_j) = e^{-\gamma(\mathbf{x}_i - \mathbf{x}_j)^2}$$



- Weitere Literatur:
  B.Schölkopf, A.J.Smola: „*Learning with Kernels*". 2002

Kernelisierte Verfahren

- Umformung des Optimierungsproblem, so dass Beispiele nur im paarweisen Skalarprodukt

$$f(\mathbf{x}_i) = \sum_{j=1}^{m} \alpha_j y_j k(\mathbf{x}_j, \mathbf{x}_i)$$

- Perzeptron
- SVM

$$\min_\alpha \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N} \alpha_i \alpha_j y_i y_j k(\mathbf{x}_i, \mathbf{x}_j)$$

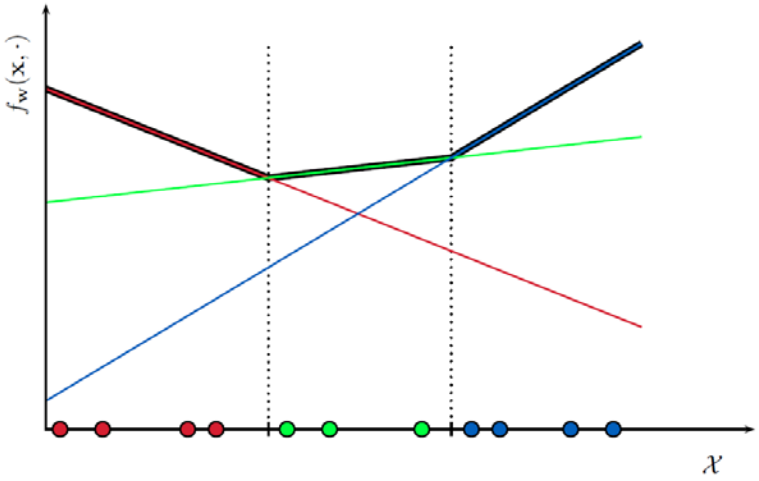
- Logistische Regression
- …

Multiklassen SVM

- Klassifikation bei mehr als zwei Klassen:
  - $y^* = \arg\max_y \; f(\mathbf{x}, y) = \arg\max_y \left\langle \mathbf{w}_y, \mathbf{x} \right\rangle$
- Gewichtsvektors für jede mögliche Klasse

- Optimierungsproblem:
  - $\min \sum_{i=1}^{k} \|\mathbf{w}_i\|^2 + C \sum_{i=1}^{n} \xi_i$
  - Mit Nebenbedingungen

$$\forall y \neq y_i : f(\mathbf{x}_i, y_i) \geq f(\mathbf{x}_i, y) + 1 - \xi_i$$



- Originalreferenz:

J.Weston, C.Watkins: „ *Support vector machines for multi-class pattern recognition*”. 1999

Feature-Mapping

- Entscheidungsfunktion bekommt zwei Parameter:

$$f(\mathbf{x}, y) = \mathbf{w}^{\mathrm{T}} \Phi(\mathbf{x}, y)$$

- Gemeinsame Repräsentation von Ein- und Ausgabe:

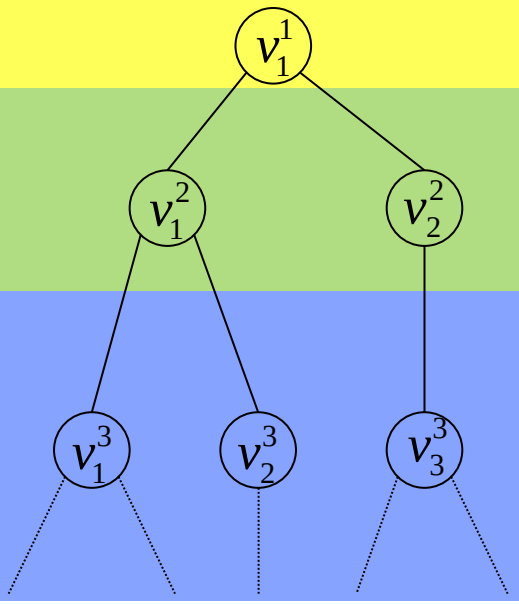
$$\Phi(\mathbf{x}, y) = \begin{pmatrix} [\![y=1]\!]\mathbf{x} \\ [\![y=2]\!]\mathbf{x} \\ [\![y=3]\!]\mathbf{x} \\ [\![y=4]\!]\mathbf{x} \\ [\![y=5]\!]\mathbf{x} \\ [\![y=6]\!]\mathbf{x} \\ \vdots \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ \mathbf{x} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \vdots \end{pmatrix} \begin{matrix} (1) \\ (2) \\ (3) \\ (4) \\ (5) \\ (6) \end{matrix}$$

- Gleicher Ansatz für Taxonomien, Sequenz- und Struktur-Lernen, Ranking, Ordinale Regression,…

Klassifikation mit Taxonomien

- $\mathbf{x}$ kodiert z.B. ein Dokument
- $\mathbf{y} = \left(v_1^1, v_2^2, v_3^3\right)^T$ ist ein Pfad z.B. in einem Themenbaum

$$\Phi(\mathbf{x}, \mathbf{y}) = \begin{pmatrix} [\![y^1 = v_1^1]\!]\mathbf{x} \\ [\![y^2 = v_1^2]\!]\mathbf{x} \\ [\![y^2 = v_2^2]\!]\mathbf{x} \\ [\![y^3 = v_1^3]\!]\mathbf{x} \\ [\![y^3 = v_2^3]\!]\mathbf{x} \\ [\![y^3 = v_3^3]\!]\mathbf{x} \\ \vdots \end{pmatrix} = \begin{pmatrix} \mathbf{x} \\ \mathbf{0} \\ \mathbf{x} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{x} \\ \vdots \end{pmatrix}$$



Strukturierte Ein-/Ausgaben

- Ausgaberaum Y beinhaltet komplexe Objekte
  - Wortart- und Eigennamenerkennung
  - Natural Language Parsing
  - Sequence Alignment
  - …
- Mehrstufenverfahren propagieren Fehler
- Warum ist das kein einfaches Multiklassen-Problem?
  - Exponentielle Anzahl von Parametern
  - Effektive Vorhersage
  - Effektives Lernen

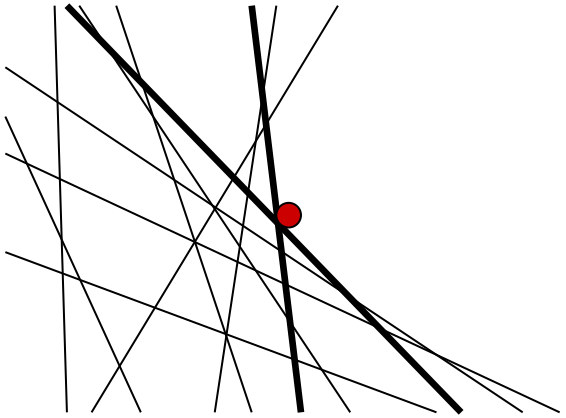
Lernen mit strukturierten Ausgaben

- Optimierungsproblem
  - $\min \|\mathbf{w}\|^2 + C\sum_{i=1}^{n} \xi_i$
  - Mit Nebenbedingungen

    $\forall i = 1 \ldots n \forall \bar{\mathbf{y}} \neq \mathbf{y}_i : \mathbf{w}^T \left(\Phi\left(\mathbf{x}_i, \mathbf{y}_i\right) - \Phi\left(\mathbf{x}_i, \bar{\mathbf{y}}\right) \right) \geq 1 - \xi_i$

    $\forall i = 1 \ldots n : \xi_i \geq 0$

- Iteratives Training.
  - Negative Constraints werden hinzugefügt, wenn beim Training Fehler auftritt.



Lernen mit strukturierten Ausgaben
Schnittebenenalgorithmus

- Gegeben: $L = \{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_n, y_n)\}$
- Wiederhole bis alle Sequenzen korrekt vorhergesagt werden.
  - Iteriere über alle Beispiele ($\mathbf{x_i}$, $\mathbf{y_i}$).
    - Bestimme $\bar{\mathbf{y}} = \arg\max_{\mathbf{y} \neq \mathbf{y}_i} \mathbf{w}^{\mathrm{T}} \Phi(\mathbf{x}_i, \mathbf{y})$
    - Wenn $\mathbf{w}^{\mathrm{T}} \Phi(\mathbf{x}_i, \mathbf{y}_i) < \mathbf{w}^{\mathrm{T}} \Phi(\mathbf{x}_i, \bar{\mathbf{y}}) + 1$ (Margin-Verletzung) dann füge Constraint $\mathbf{w}^{\mathrm{T}} \Phi(\mathbf{x}_i, \mathbf{y}_i) \geq \mathbf{w}^{\mathrm{T}} \Phi(\mathbf{x}_i, \bar{\mathbf{y}}) + 1 - \xi_i$ dem Working Set hinzu.
    - Löse Optimierungsproblem für Eingabe $\mathbf{x_i}$, Ausgabe $\mathbf{y_i}$, und negative Pseudo-Beispiele $\bar{\mathbf{y}}$ (working set).
- Liefere **w** zurück.

Halbüberwachtes Lernen

- Verschiedene Mechanismen, mit denen Informationen aus ungelabelten Daten genutzt werden.
  - Transduktion
  - Graph-Laplacians
  - Co-Lernen