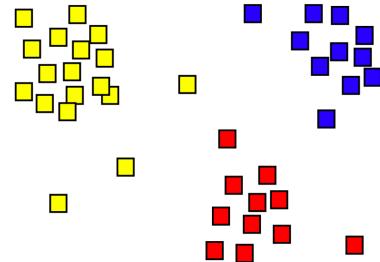
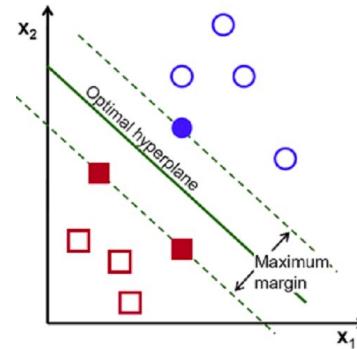
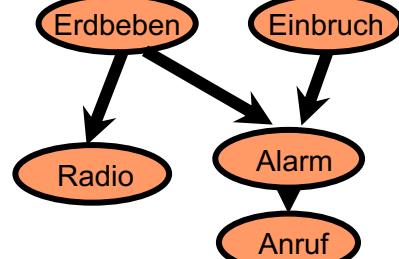
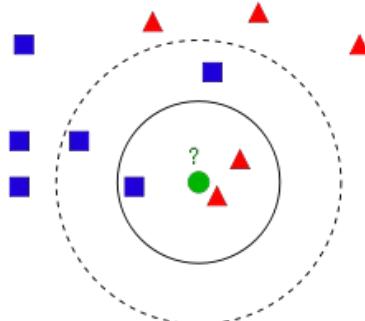


# Data Mining und Maschinelles Lernen

## Weitere Verfahren des Maschinellen Lernen: Probabilistische Graphische Modelle, Stützvektormethode und Clustering



Basierende auf Folien von Katharina Morik, Uwe Ligges, Claus Weihs, Lutz Plümer und viele anderen.  
Danke fürs Offenlegen ihrer Folien

# Bayes-Optimaler Klassifikator

Eigentlich ist das Ziel der Klassifikation die Bestimmung einer insofern **optimalen Klassifikationsregel**, als dass die Fehlerwahrscheinlichkeit dieser Regel minimal ist

$$P(y_{\text{Regel}}(x) \neq y_{\text{wahr}}(x)) \rightarrow \min!$$

# Erkennung von „Schrott“ in der Produktion: Klettenlabkraut in Winterweizen



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

Hohe Ertragsverluste, niedrige Schadsschwelle, Behinderung der Erntearbeiten, hohe Lebensdauer des Samens



# Datenvorverarbeitung / Segmentierung



## 1. Datenvorverarbeitung

- Erfassung im Rot- und Infrarot-Bereich

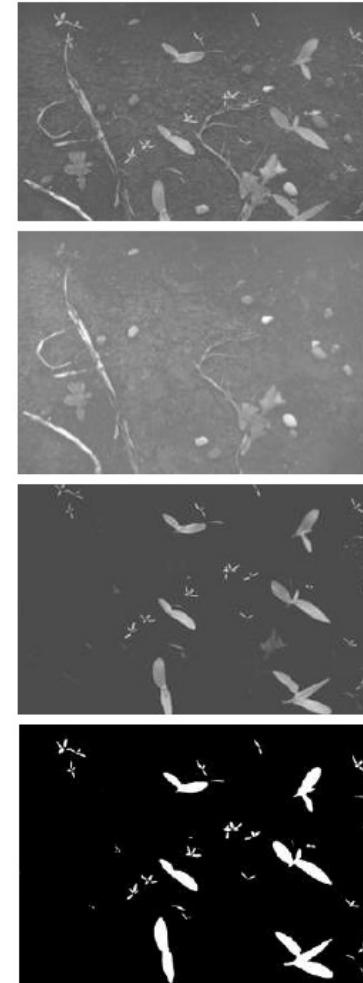
2. Durch Ausnutzung der Temperaturdifferenz  
gelingt die Trennung von Pflanze und  
Untergrund/Boden

3. Resultiert in einem Differenzbild

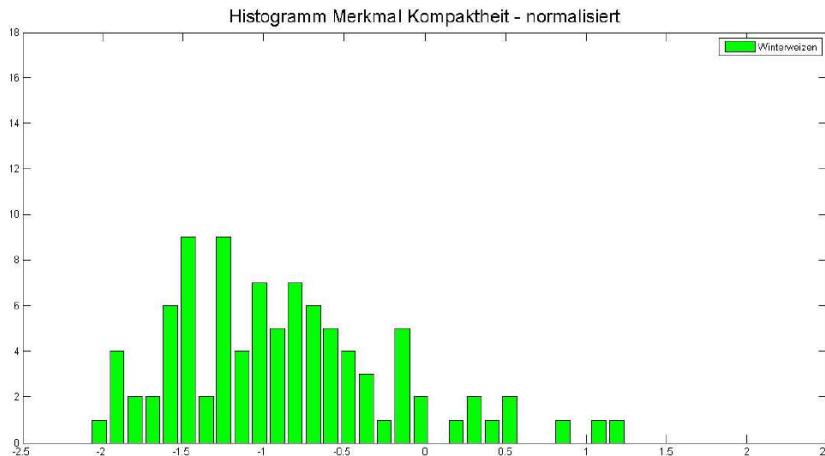
- Binärbild isoliert die Pflanzen

4. Bildverarbeitung liefert Formparameter

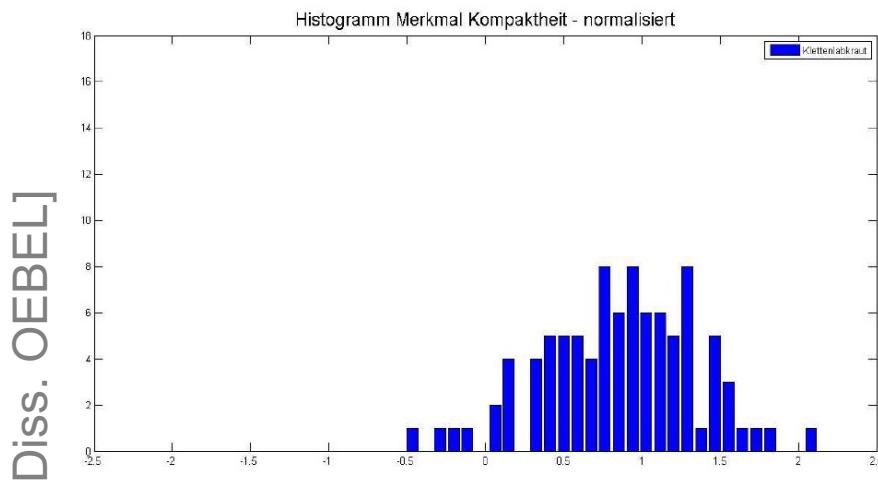
- **Kompaktheit**
- **Schlankheit**



# Histogramme für Kompaktheit

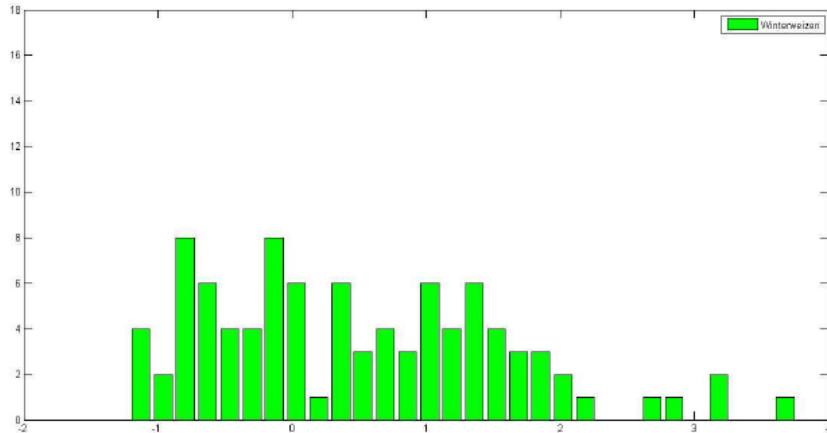


Winterweizen

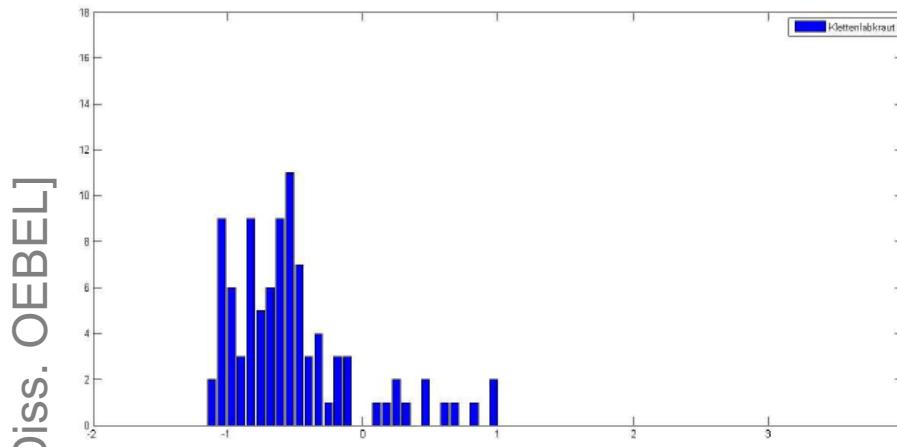


Klettenkleblaub

# Histogramme für Schlankheit



Winterweizen



Klettenkleblaub

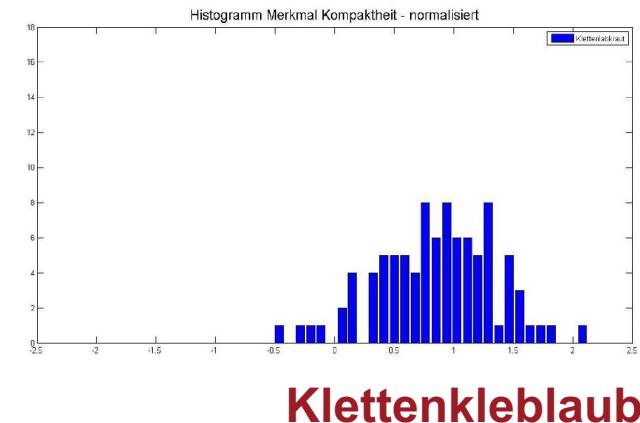
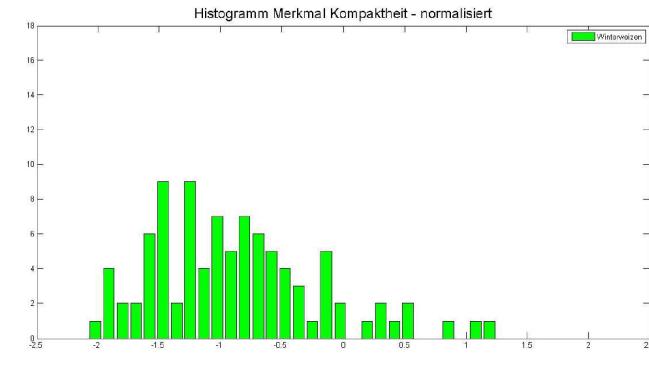
# Bedingte Wahrscheinlichkeit (für das Merkmal „Kompaktheit“)



Das Merkmal „Kompaktheit“ hat offensichtlich unterschiedliche Verteilungen für Winterweizen und Klettenkleblaub

**P(A|B) ist die Wahrscheinlichkeit von A unter der Bedingung B**

hier: die Wahrscheinlichkeit der Beobachtung eines bestimmten Werts der Kompaktheit unter der Bedingung z.B. „Klettenlabkraut“



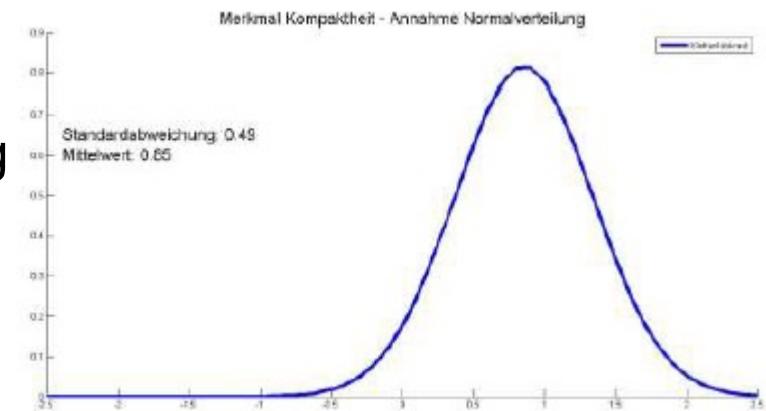
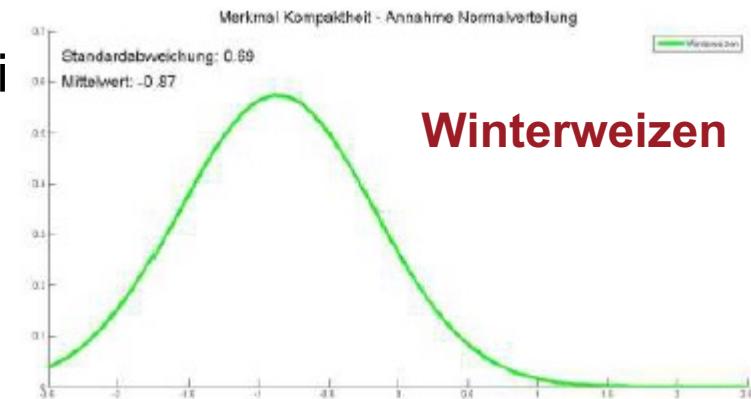
# Annahme: Kompaktheit „normalverteilt“



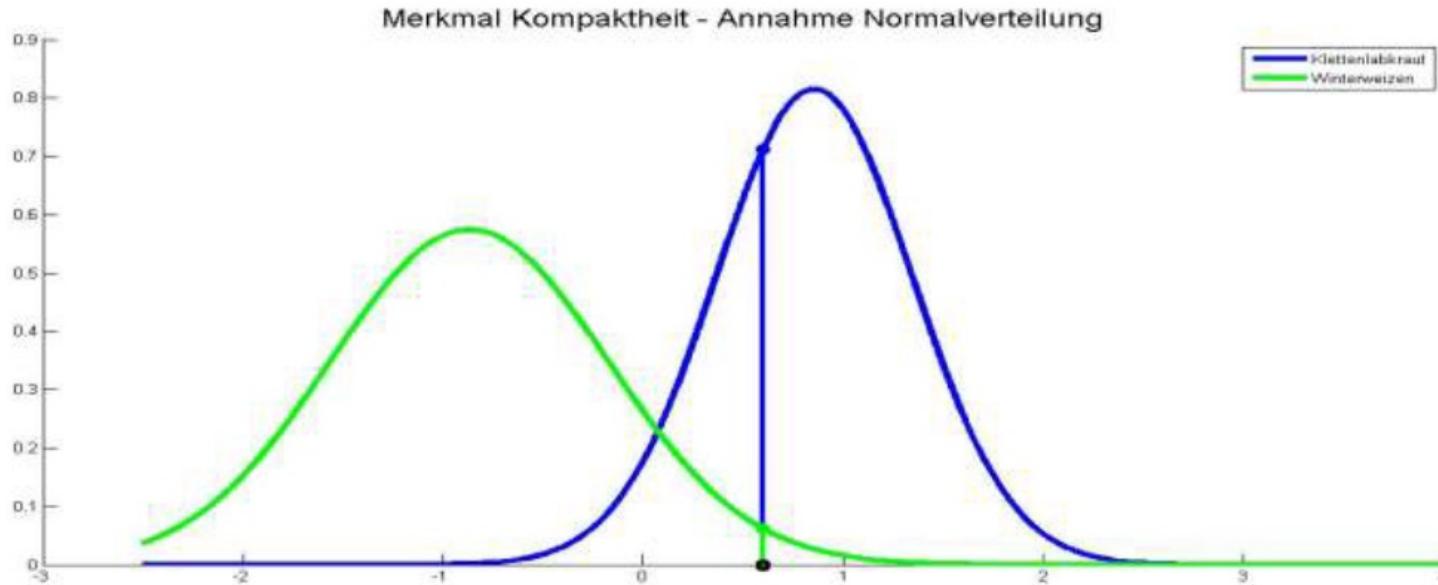
Machen wir die vereinfachende Annahme, das Merkmal Kompaktheit sei „normalverteilt“

Dann wird jede der beiden Verteilungen durch die beiden Parameter **Mittelwert** und **Standardabweichung** eindeutig beschrieben

Die Wahrscheinlichkeit der Beobachtung einer bestimmten Ausprägung des Merkmals Komapktheit beschreibt die Wahrscheinlichkeitsdichte-Funktion pdf („Probability Density Function) – hier auch „Likelihood“ genannt



# Maximum-Likelihood-Klassifikator

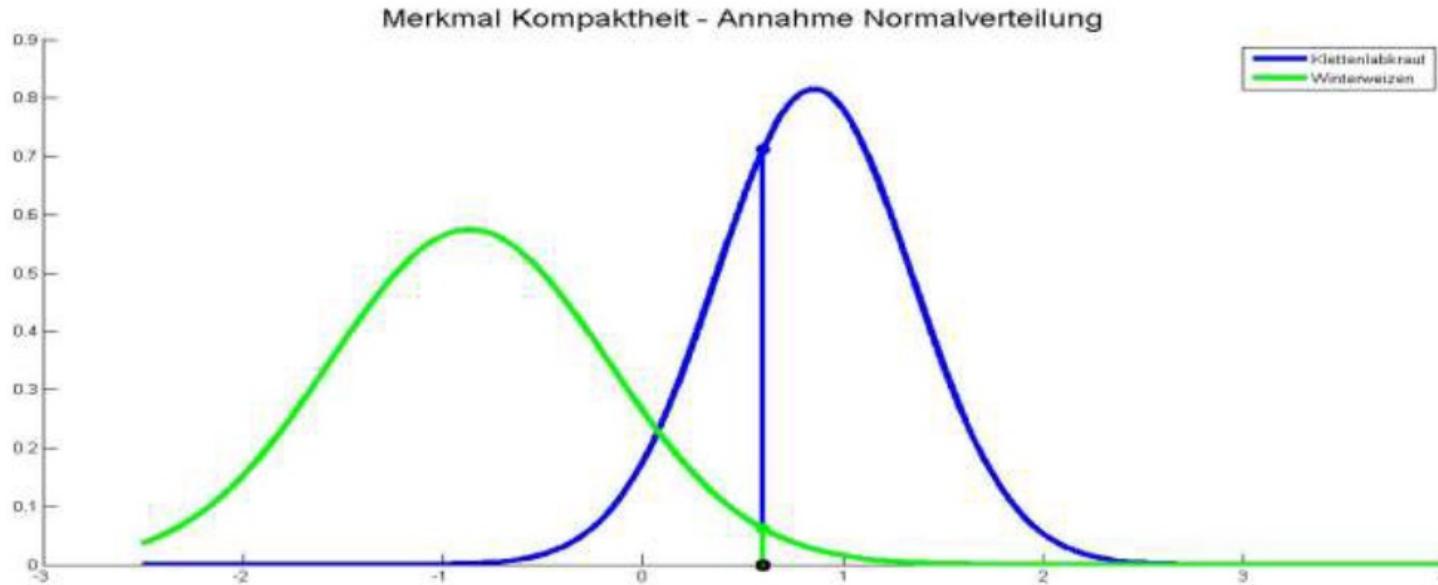


$$p_{\text{Klettenlabkraut}} = p(\text{kompaktheit} \mid \text{Klettenlabkraut})$$

$$p_{\text{Winterweizen}} = p(\text{kompaktheit} \mid \text{Winterweizen})$$

Der Likelihood-Klassifikator wählt die Klasse mit der höchsten Likelihood

# Zahlenbeispiel



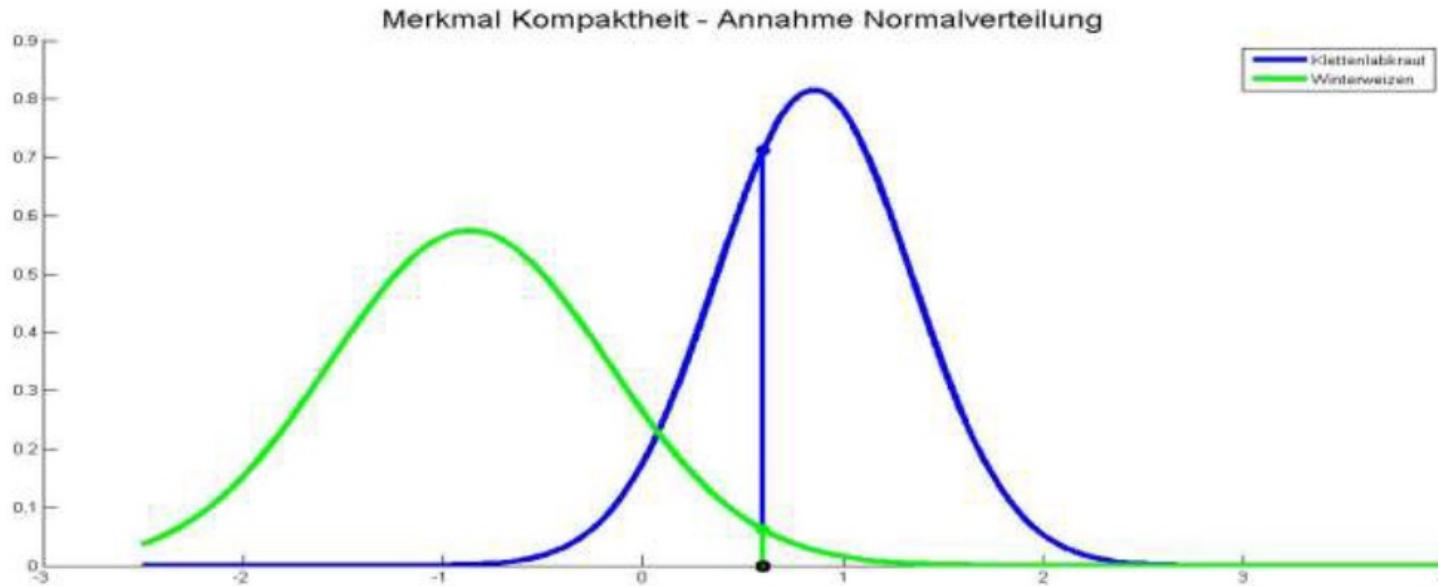
Sei die Kompaktheit = 0.6

$$p(0.6 | \text{Klettenlabkraut}) = 0.7119$$

$$p(0.6 | \text{Winterweizen}) = 0.0614$$

Vorhersage/Klassifikation: **Klettenlabkraut**

# „A Priori“ Gleichverteilt



Achtung! Wir haben eine (vielleicht unrealistische) Annahme getroffen:  
die „a priori“ Wahrscheinlichkeiten beider Klassen seien gleich  
 $p(\text{Klettenlabkraut}) = p(\text{Winterweizen}) = 0.5$

Dann muss die Summe beider Wahrscheinlichkeiten 1 ergeben. Durch „Normierung“ erhalten wir dann aus den beiden Likelihoods die Wahrscheinlichkeit der Klassifikation:  $p(\text{Klettenlabkraut}) = 0.7119 / (0.0614 + 0.7119) = 0.92$

# Im Allgemeinen: Satz von Bayes

$$P(K|M) = \frac{P(M|K) * P(K)}{P(M)}$$

wenn  $P(M) > 0$



$$P(\text{Klasse}|Merkmal) = \frac{\text{Likelihood} * \text{Prior}}{P(M)}$$

wenn  $P(M) > 0$

# Bayes'cher Klassifikator (I)



Likelihood: bedingte Wahrscheinlichkeit des Merkmals

Prior: Vorwissen über die Häufigkeit des Auftretens der Klasse (des Labels)

- $p(\text{Winterweizen}) = 0.95$  und  $p(\text{Klettenlabkraut}) = 0.05$
- Der „Prior“ für **W** ist wesentlich höher als für **K**

$$P(\text{Klasse}|\text{Merkmal}) = \frac{\text{Likelihood} * \text{Prior}}{P(M)} \text{ wenn } P(M) > 0$$

Aber wie geht das an?  $P(\text{Merkmal})$  also die Verteilung der Daten ist doch in den meisten Fällen nicht bekannt?! Da wir maximieren, und  $P(\text{Merkmal})$  für alle Klassen gleich ist, ist das kein Problem.

$$\text{Klasse}^* = \arg \max_{\text{Klasse}} \text{Likelihood}(\text{Klasse}) * \text{Prior}(\text{Klasse})$$

Wenn wir die wirkliche „a posteriori“ Wahrscheinlichkeit berechnen wollen, können wir ausnutzen, dass  $P(\text{Merkmal})$  eine „normalisierende Konstante“. Dabei hilft die Einsicht, dass die Gesamtwahrscheinlichkeit 1 ergeben muss

# Zahlenbeispiel



$$p(0.6 \mid \text{Klettenlabkraut}) = 0.7119$$

$$p(0.6 \mid \text{Winterweizen}) = 0.0614$$

$$p(\text{Winterweizen}) = 0.95$$

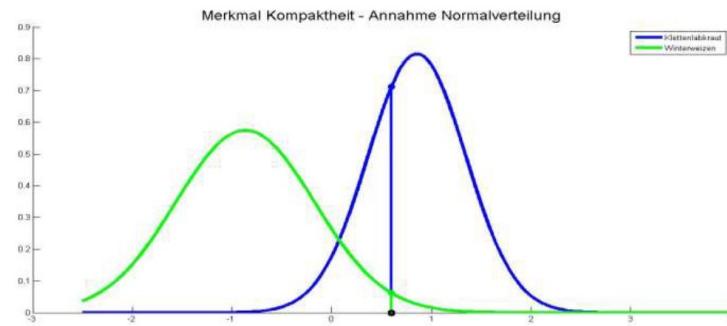
$$p(\text{Klettenlabkraut}) = 0.05$$

$$p(\text{Winterweizen} \mid 0.6) = 0.0614 * 0.95 / p(0.6) = 0.0583 / p(0.6)$$

$$p(\text{Klettenlabkraut} \mid 0.6) = 0.7119 * 0.05 / p(0.6) = 0.0356 / p(0.6)$$

$$p(\text{Winterweizen} \mid 0.6) = 0.0583 / (0.0583 + 0.0356) = 0.6212$$

$$p(\text{Klettenlabkraut} \mid 0.6) = 0.0356 / (0.0583 + 0.0356) = 0.3788$$



# Bayes'cher Klassifikator (II)



Der Bayes'sche Klassifikator setzt voraus, daß wir zwei Verteilungen kennen, den **Likelihood  $P(M|K)$**  und den **Prior  $P(K)$**

Das "Lernen" des Klassifikators besteht in dem Fall im **Schätzen dieser beiden Verteilungen**

Z.B. bei einer **Gaußverteilung (Normalverteilung)** müssen wir **Mittelwert** und **Standardabweichung** schätzen

# Bayes'cher Klassifikator (III)

Wir haben hier den **univariaten** Fall diskutiert (nur Kompaktheit)  
Es geht aber auch bei mehreren Merkmalen (**multivariater** Fall).  
Dann brauchen wir die **Kovarianzmatrix**  
Oder wir treffen eine Unabhängigkeitsannahme (**Naive Bayes**):

$$P(M|k) = P(M_1, M_2, \dots, M_l | k) = \prod_{i=1}^l P(M_i | k)$$

$$\frac{P(M|k_1)*P(k_1)}{P(M|k_1)*P(k_1)+P(M|k_2)*P(k_2)}$$

# Bayes'cher Klassifikator (IV)



Wir haben hier den **univariaten** Fall diskutiert (nur Kompaktheit)  
Es geht aber auch bei mehreren Merkmalen (**multivariater** Fall).  
Dann brauchen wir die **Kovarianzmatrix** (siehe vorherige Vorlesungen)

Der Bayes'che Klassifikator hat **drei wesentliche Vorteile**:

- Er kann “Vorwissen” in Form des Priors nutzen. Zum Beispiel die Tatsache, dass in einem Schlag “a priori” Klettenlabkraut seltener ist als Weizen
- Er liefert optimale Ergebnisse (unter den getroffenen Annahmen,)
- Er kann die Güte der Klassifikation in Form einer Wahrscheinlichkeit angeben:

$$\frac{P(M|k_1) * P(k_1)}{P(M|k_1) * P(k_1) + P(M|k_2) * P(k_2)}$$

# Was wissen wir jetzt?

---

Bayes-Klassifikatoren betrachten die Verteilung, die die Daten generiert haben

Sie treffen „optimale“ Entscheidungen mit Hilfe der Bayes-Regel

Der Naive Bayes Klassifikator nimmt an, dass alle möglichen Teilmengen von Merkmalen voneinander unabhängig sind, geben das Klassenlabel



Viele tägliche Situationen sind von Unsicherheiten und Ungenauigkeiten geprägt.



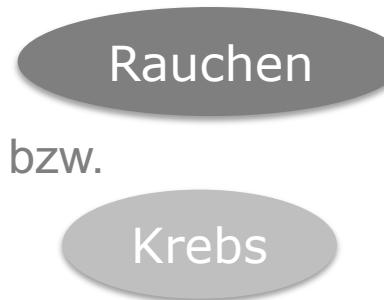
Viele tägliche Situationen sind von Unsicherheiten und Ungenauigkeiten geprägt. Mit denen muss auch ein Maschine umgehen können.



# Wie wir gerade schon gesehen haben, kann man unsicheres Wissen mittels Wahrscheinlichkeiten beschreiben



Wahrscheinlichkeiten  
dafür, dass **Rauchen=r** bzw.  
**Krebs=k** eintrifft



Nein	Gelegentlich	Stark
0.800	0.150	0.050
Nein	Benigne	Maligne
0.935	0.046	0.019

Diese “Einzel”-Wahrscheinlichkeiten (Marginals) reichen aber nicht, um z.B. die **Wahrscheinlichkeit** zu beschreiben, **mit der Rauchen zu Krebs führt**.

I. A. sind wir an der **Verbundwahrscheinlichkeit  $P(R,K)$**  interessiert, dass **Rauchen=r und Krebs=k** gemeinsam eintreffen

Rauchen

Die Werte sind **mindestens 0, höchstens 1** und **summieren sich zu 1 auf**

		Krebs		
		Nein	Benigne	Maligne
Rauchen	Nein	0.768	0.024	0.008
	Gelegentlich	0.132	0.012	0.006
	Stark	0.035	0.010	0.005

**Im Allgemeinen, wollen die Verbundwahrscheinlichkeit darstellen , weil sich aus ihr alles ablesen lässt.**



### Marginalisierung

$$P(K) = \sum_{i=1}^n P(K, R = r_i)$$

Krebs

summiere

Rauchen	Krebs			TOTAL
	Nein	Benigne	Maligne	
Nein	0.768	0.024	0.008	0.800
Gelegentlich	0.132	0.012	0.006	0.150
Stark	0.035	0.010	0.005	0.050
TOTAL	0.935	0.046	0.019	

P(Rauchen)

P(Krebs)

### Bedingte Wahrscheinlichkeiten

$$P(K|R) = \frac{P(K, R)}{P(R)}$$

Wir hatten ja schon  $P(R, K)$  und  $P(K)$ . Also teilen wir sie entsprechend.

	Nein	Benigne	Maligne
Nein	0.768/0.800	0.024/0.800	0.008/0.800
Gelegentlich	0.132/0.150	0.012/0.150	0.006/0.150
Stark	0.035/0.050	0.010/0.050	0.005/0.050

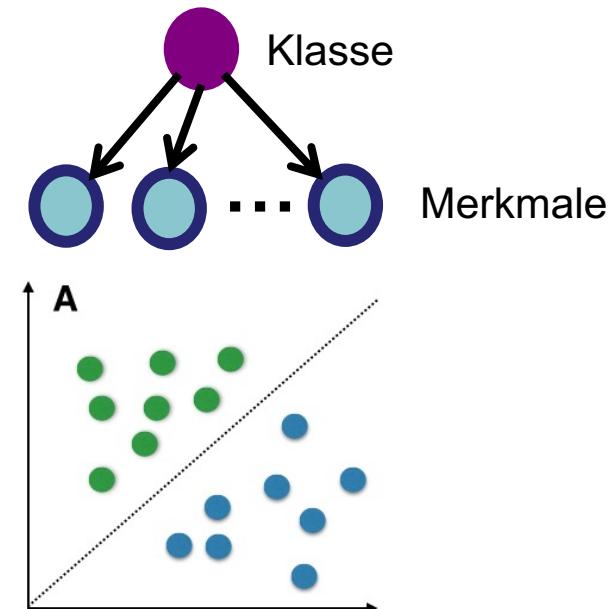
Wahrscheinlichkeiten  $P(\text{Krebs}|R)$ , mit der Rauchen zu Krebs führt

**Exponentielle Platz- und Laufzeitkomplexität !**

# Graphische Modelle bieten einen kompakten Umgang mit Verbundwahrscheinlichkeiten.

## Sind zu Arbeitstieren des Maschinellen Lernens geworden

- > **Informationsdienste, Suche, Kollaboratives Filtern, Expressionsanalyse, Sprachverarbeitung, Bioinformatik, Frage-Antwort Systeme, ... und viele mehr**
- > Warum? Daten und Wissen sind verrauscht, also unsicher



Judea Pearl erhielt den ACM Turing Award 2012 für seine Arbeiten zum probabilistischen und kausalen Schlussfolgern.

# Als ein Beispiel: Bayes'sche Netzwerke (BNs)



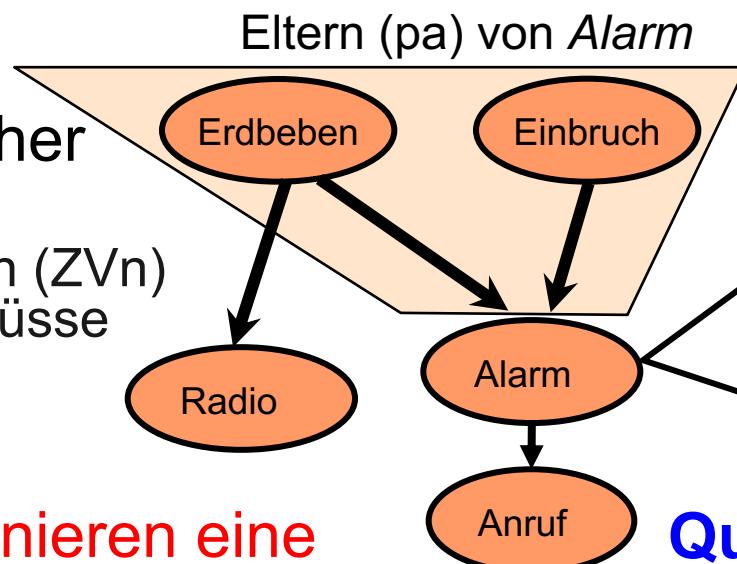
Sie sind eine kompakte Darstellung von Verteilungen durch Ausnutzen von Unabhängigkeiten: das Eintreten des einen Ereignisses beeinflusst die Wahrscheinlichkeit des Eintretens des anderen Ereignisses nicht  $P(X,Y|Z)=P(X|Z)P(Y)$

## Qualitativer Teil:

Gerichteter, azyklischer Graph (DAG)

Knoten – Zufallsvariablen (ZVn)

Kanten – gerichtete Einflüsse



$E_r$	$E_i$	$P(A_l   E_r, E_i)$	
e	b	0.9	0.1
e	$\bar{b}$	0.2	0.8
$\bar{e}$	b	0.9	0.1
$\bar{e}$	$\bar{b}$	0.01	0.99

Beide zusammen definieren eine (faktorierte) Wahrscheinlichkeitsverteilung (\*) : Produkt aller bedingten Wahrscheinlichkeitsverteilungen

Quantitativer Teil:  
Menge von bedingten Wahrscheinlichkeitsverteilungen

$$P(E_r, E_i, R, A_l, A_n) = P(E_r)P(E_i)P(R|E_r)P(A_l|E_r, E_i)P(A_n|A_l)$$

# Wie berechnen wir Wahrscheinlichkeiten mittels Bayesschen Netzwerken?



Wir wollen die Verteilung über  $Z$  bestimmen, wenn wir die Evidenz  $e$  beobachtet haben:  $P(Z|e) = P(Z, e)/P(e)$

Bis auf Normalisierung  $P(Z|e) = \propto P(Z, e)$  die Verbundwahrscheinlichkeit von  $Z$  und  $e$ .

Allgemein für eine Menge  $\mathbf{Y}$  von Zufallsvariablen:

$$P(\mathbf{Y}) = \sum_{X_i \notin \mathbf{Y}} \prod_{i=1}^n P(X_i | \text{Pa}(X_i))$$

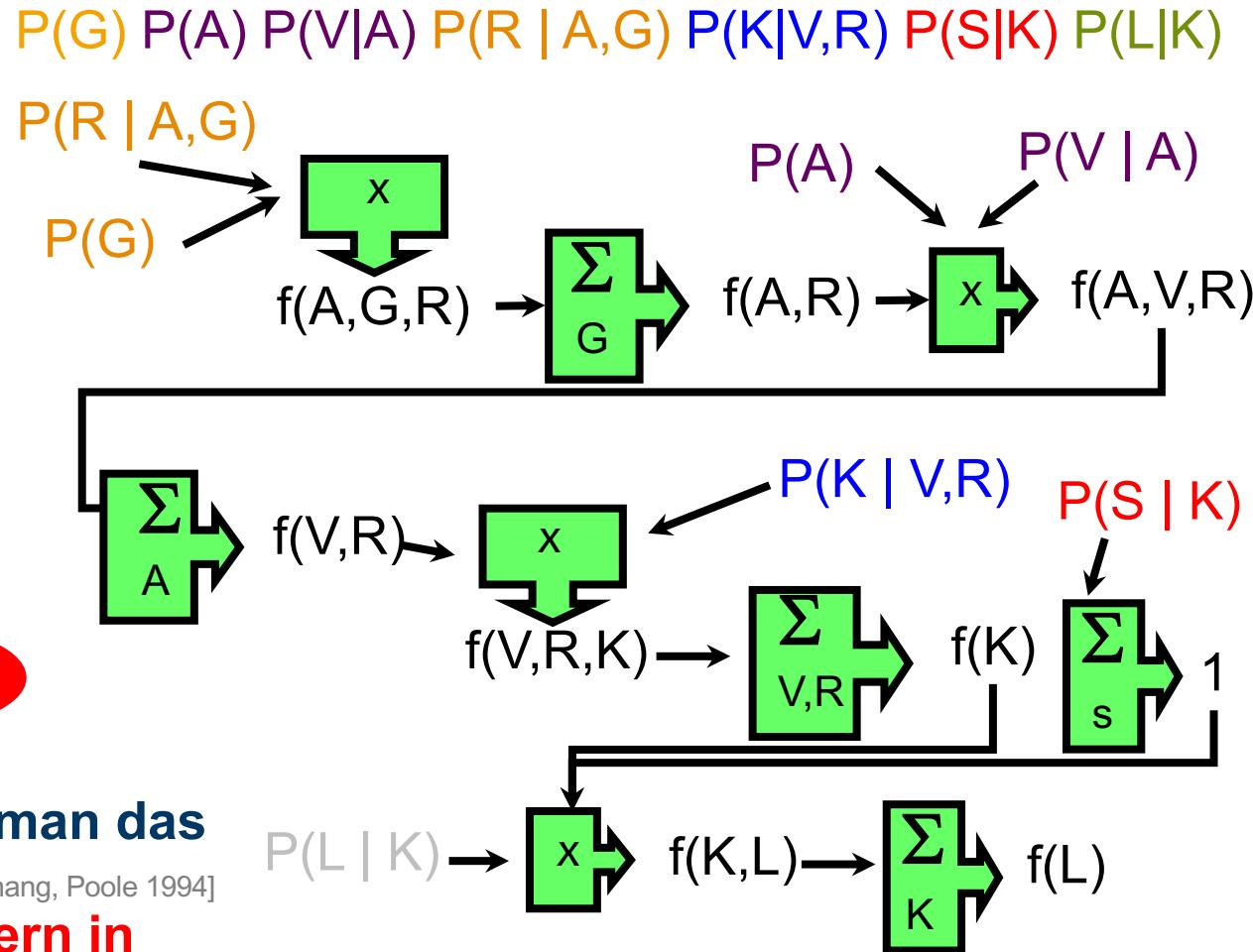
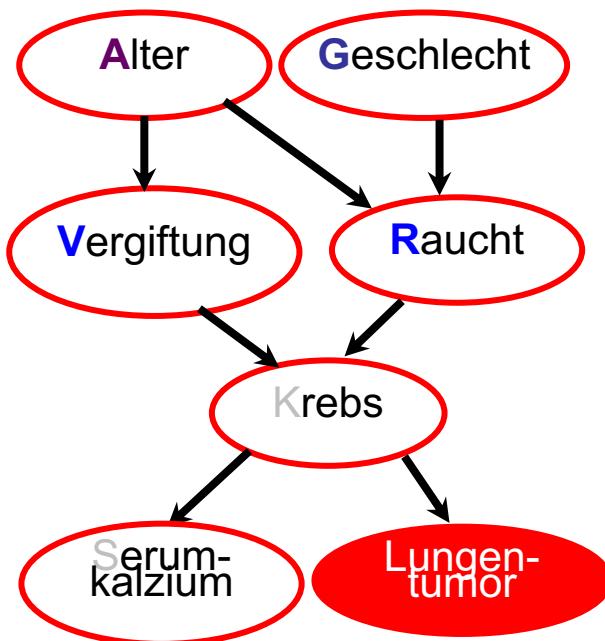
BN Faktorisierung (\*)

Marginalisierung (Eliminierung von Zufallsvariable)

**Zentrale Einsicht:  $\Sigma$  und  $\prod$  vertauschbar**

$$\sum_a (P_1 \times P_2) = (\sum_a P_1) \times P_2 \text{ falls A nicht in } P_2$$

# Ein komplexeres Beispiel bei dem wir nacheinander G,A,V,R,S und K “eliminieren”, um P(Lungentumor) zu berechnen



Als Algorithmus nennt man das Variablen-Elimination [Zhang, Poole 1994]  
I.A. ist das Schlussfolgern in Bayes'schen Netzwerken NP-schwierig (#P)

Nach Normalisierung (so dass sich die Werte in der letzten Tabelle  $f(L)$  zu 1 aufsummieren) ist das  $P(\text{Lungentumor})$

◆ What do when we find a problem that looks hard...



I couldn't find a polynomial-time algorithm; I guess I'm too dumb.

◆ Sometimes we can prove a strong lower bound... (but not usually)



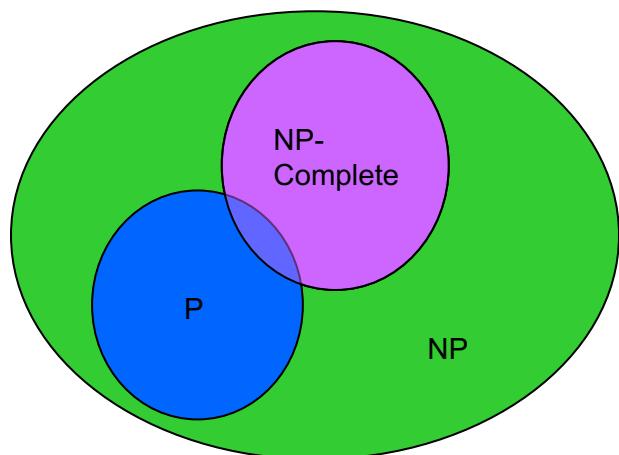
I couldn't find a polynomial-time algorithm, because no such algorithm exists!

◆ NP-completeness lets us show collectively that a problem is hard.

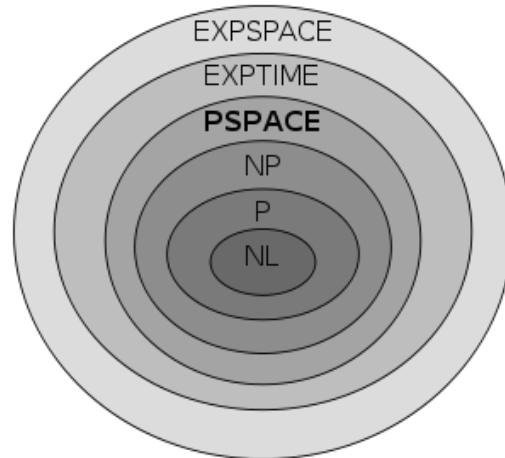
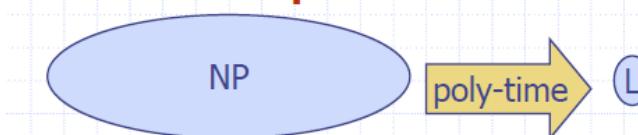


I couldn't find a polynomial-time algorithm, but neither could all these other smart people.

- $P = \{ L \mid L \text{ is accepted by a deterministic Turing Machine in polynomial time} \}$
- $NP = \{ L \mid L \text{ is accepted by a non-deterministic Turing Machine in polynomial time} \}$



- ◆ A problem (language)  $L$  is **NP-hard** if every problem in NP can be reduced to  $L$  in polynomial time.
- ◆ That is, for each language  $M$  in NP, we can take an input  $x$  for  $M$ , **transform** it in polynomial time to an input  $x'$  for  $L$  such that  $x$  is in  $M$  if and only if  $x'$  is in  $L$ .
- ◆  $L$  is **NP-complete** if it's in NP and is NP-hard.

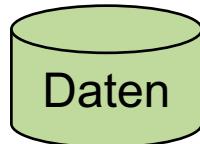


# Aber woher kommen die Netzwerke?



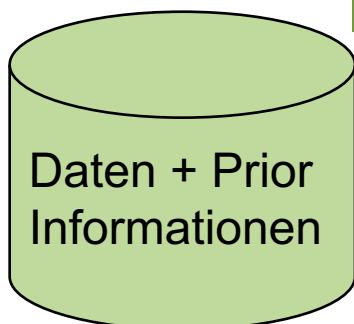
## “Knowledge acquisition bottleneck”

Es ist teuer, den Experten ihr Wissen aus der “Nase zu ziehen”, und meistens haben wir gar keinen Experten zu Hand!



Daten

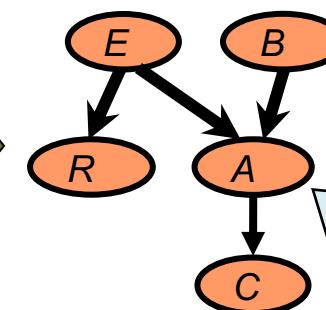
**Daten sind oftmals billiger zu bekommen**



Daten + Prior  
Informationen



Maschineller  
Lernalgorithmus

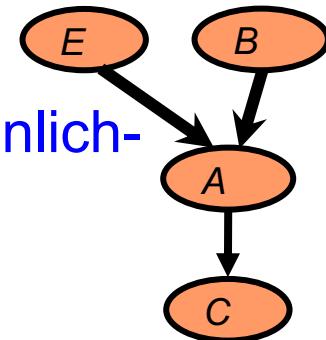


E	B	$P(A   E, B)$	
e	b	.9	.1
e	—	.7	.3
—	b	.8	.2
—	—	.99	.01
e	b		

# Einfachste Variante\*: Parameterschätzung bei vollständigen Daten



Qualitativer Teil eines Netzwerkes ist gegeben und wir wollen den quantitativen Teil (die bedingten Wahrscheinlichkeitsverteilung assoziiert mit jedem Knoten) aus einer gegebenen Menge von Trainingsbeispielen schätzen



## Zufallsvariablen

$$D = \begin{bmatrix} E[1] & B[1] & A[1] & C[1] \\ E[2] & B[2] & A[2] & C[2] \\ \vdots & \vdots & \vdots & \vdots \\ E[M] & B[M] & A[M] & C[M] \end{bmatrix}$$

Trainingsbeispiele  
(Konfigurationen  
der Zufallsvariable)

\*Es gibt auch Methoden zur Selektion des qualitativen Teils eines Netzwerkes. Diese werden hier nicht behandelt, beruhen aber auf den einfacheren Varianten.

# Maximum-Likelihood Schätzung (MLE) für multinomialen Zufallsvariable (hier ohne Herleitung)



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

Bei vollständigen Daten **zählen wir, wie häufig**

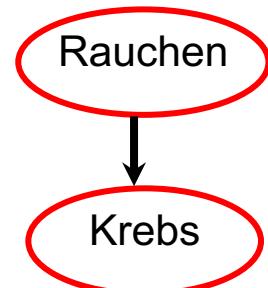
**Zufallsvariablen in einem Zustand sind**  $N(x_i, pa_i)$

und schätzen die Parameter als

$$\hat{\theta}_{x_i | pa_i} = \frac{N(x_i, pa_i)}{N(pa_i)}$$

Das ist **asymptotisch konsistent** und kann auch online (also für große Datenmengen) berechnet werden  
Wir haben 6-mal malignen Krebs bei 10 starken Rauchern aus 20 Menschen beobachtet:

$$\theta_{R=S} = 10 / 20 = 0.5 \text{ und } \theta_{K=m|R=S} = 6 / 10 = 0.6$$



# Allerdings sind Daten oft unvollständig!

Einige Variable sind niemals beobachtet  
(gekennzeichnet durch "?").

$$D = \begin{bmatrix} E[1] & ? & A[1] & C[1] \\ E[2] & ? & A[2] & ? \\ \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots \\ E[M] & ? & A[M] & C[M] \end{bmatrix}$$

Einige Zufallsvariablen sind in einigen Trainingsinstanzen nicht beobachtet.

# Expectation Maximization (EM)

---

Allgemeines Schema zum Schätzen von  
Wahrscheinlichkeitsverteilungen mit unvollständigen Daten

## Intuition:

Wenn wir die wahren Zählungen hätten, könnten wir direkt die Parameter bestimmen.

Wenn aber Werte fehlen, dann sind die Zählungen unvollständig.

Wir vervollständigen die Zählungen durch probabilistisches Schlussfolgern im aktuellen Modell.

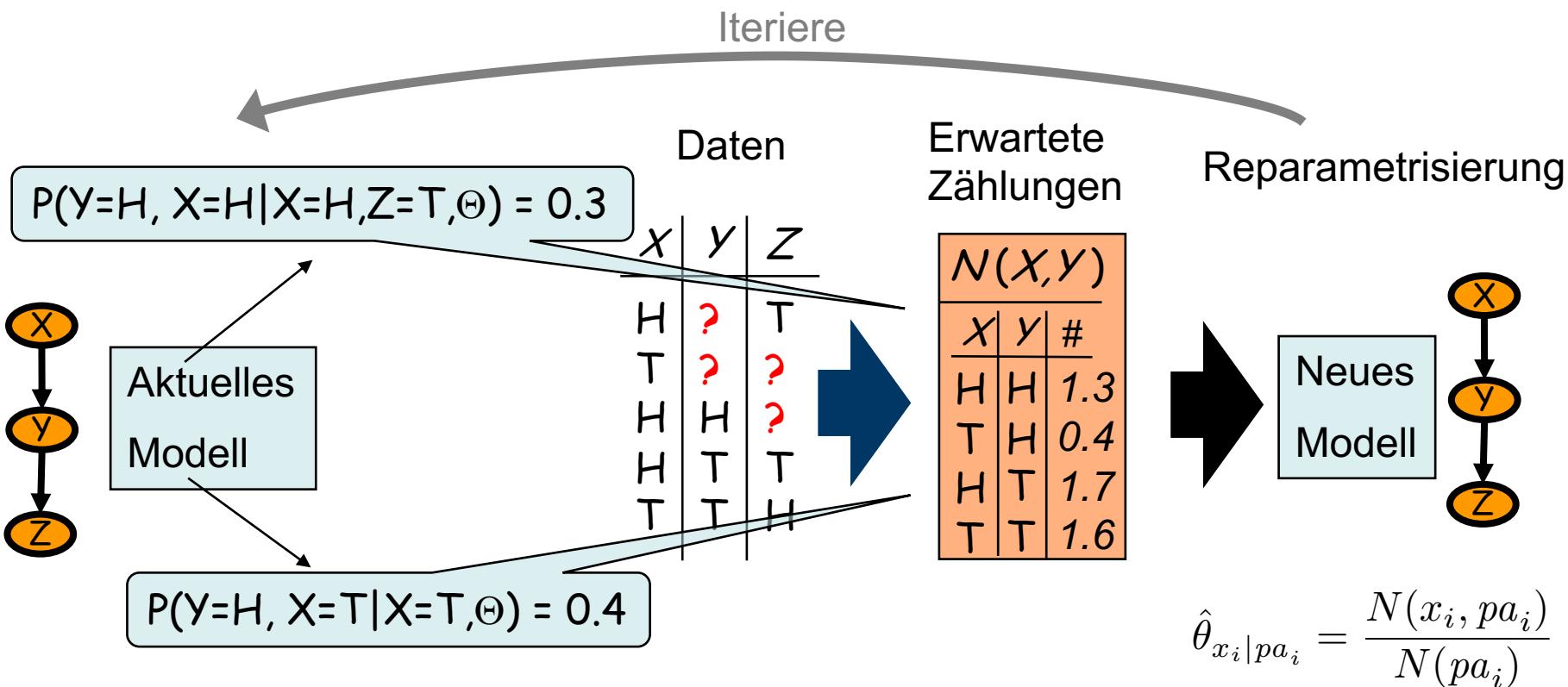
Wir benutzen die vervollständigten Zählungen, als ob sie die echten Zählungen wären.

# Expectation Maximization (EM)

am Beispiel von Bayes'schen Netzwerken mit multinomialen Zufallsvariablen



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT



Allerdings machen wir ja etwas falsch. Daher iterieren wir das, bis wir konvergieren.

# Was haben Sie kennengelernt?



- Graphische Modelle sind Arbeitstiere der Datenwissenschaften.
- Sie spezifizieren Wahrscheinlichkeitsverteilungen und erlauben das Schlussfolgern unter Unsicherheit.
- Unvollständige Daten machen die Parameterschätzung schwierig, weil die Likelihood-Funktion multimodal ist und keine geschlossene Form hat.
- In diesem Falle können Sie die ML-Parameter mittels Expectation-Maximization (EM) schätzen.
- Dazu müssen wir schlussfolgern. Das ist (NP-)schwierig!

# Von Bayes'schen zu Diskriminativen Ansätzen



---

Der Bayes'sche Ansatz:

Falls  $P(x|k_1) \cdot P(k_1) > P(x|k_2) \cdot P(k_2)$  wähle k1, sonst k2

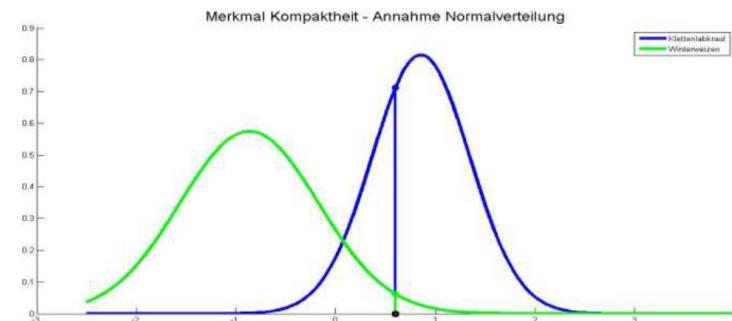
Das motiviert die Suche nach einem **Schwellwert  $x_0$**  mit

$$P(x_0|k_1) \cdot P(k_1) = P(x_0|k_2) \cdot P(k_2)$$

Dieser Schwellwert ist ein **(lineare) Diskriminator**

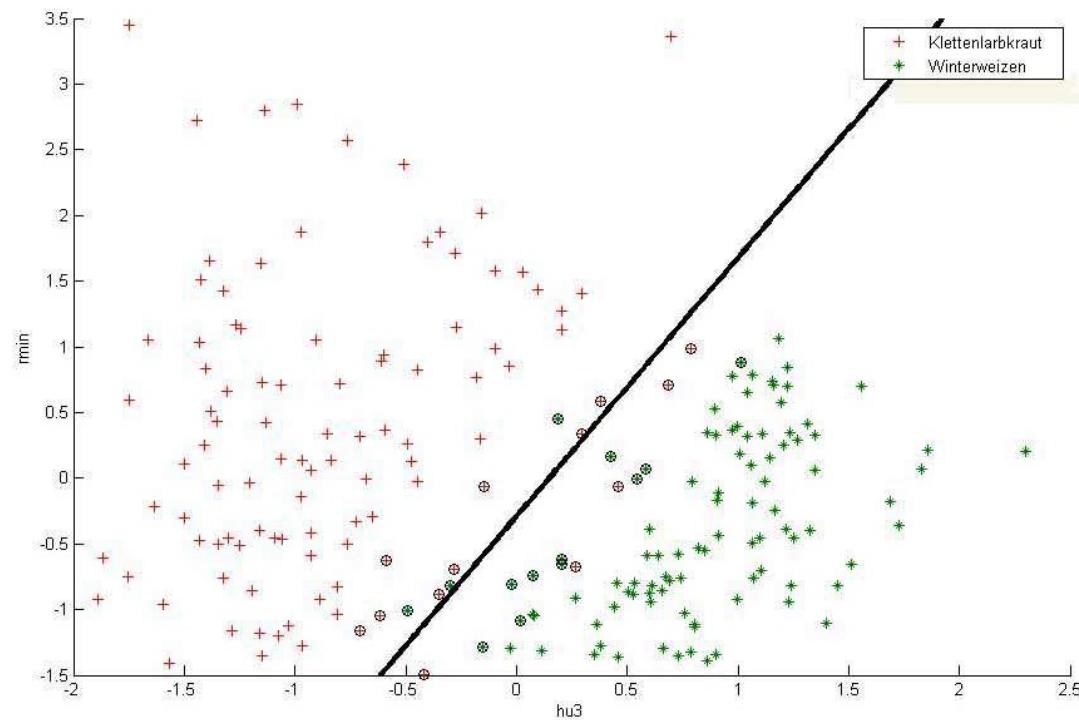
Im Allgemeinen, sind das:

- Ein Merkmal: Schwellwert
- Zwei Merkmale: Gerade
- Drei Merkmale: Ebene
- N Merkmale: (Lineare) Hyperebene



# Lineare Trennung

Kombination beider Merkmale: die zwei Klassen sind durch eine Gerade offenbar recht gut trennbar



# Lineare Trennung: Minimierung des beobachteten Fehlers



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

Funktionslernaufgabe nicht direkt lösbar. Problem:

- Die tatsächliche Funktion  $f(X)$  ist unbekannt.
- Die zugrunde liegenden Wahrscheinlichkeiten sind unbekannt.

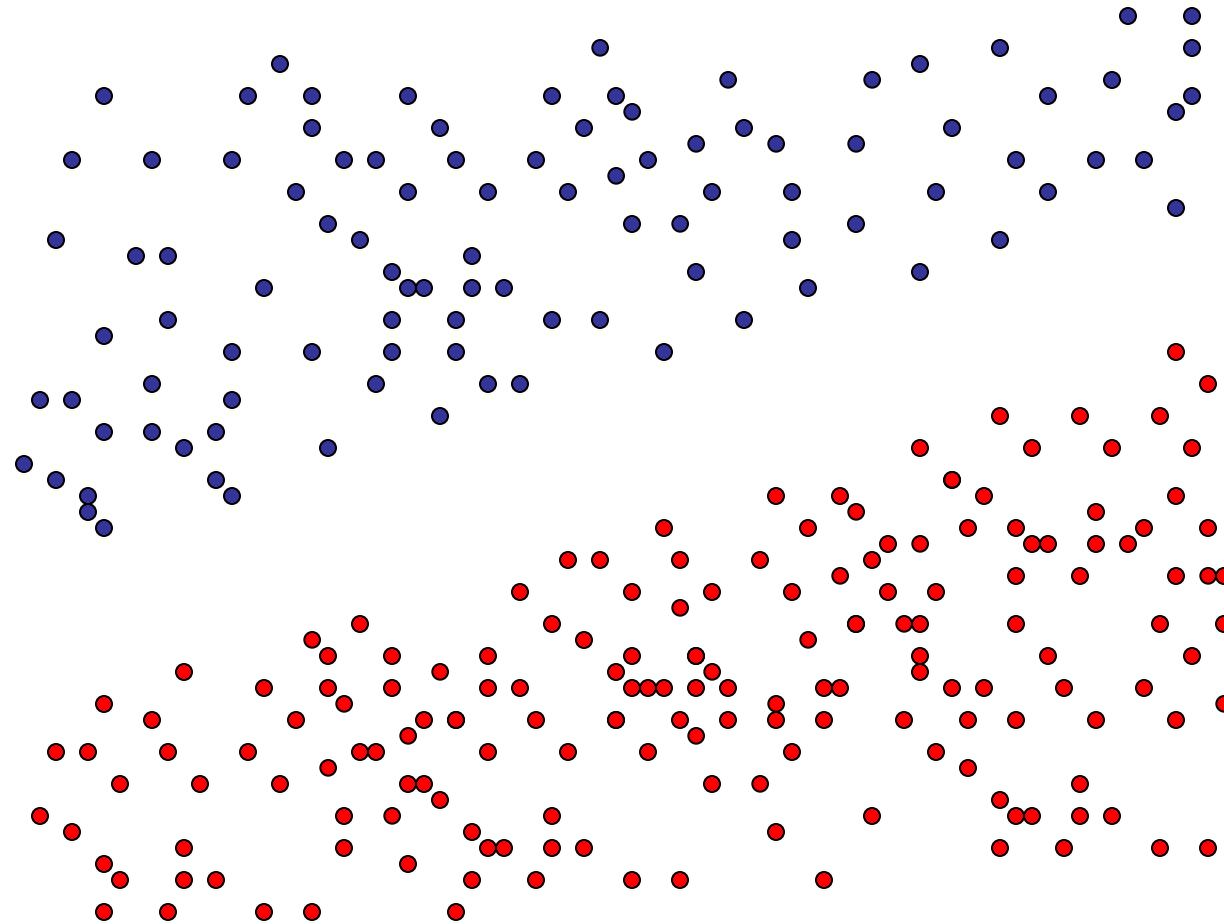
## Ansatz der Empirical Risk Minimization (ERM):

Einfach gesagt: eine hinreichend große Lernmenge nehmen und für diese den Fehler minimieren.

# Lineare Trennung: Minimierung des beobachteten Fehlers — Beispiel 1



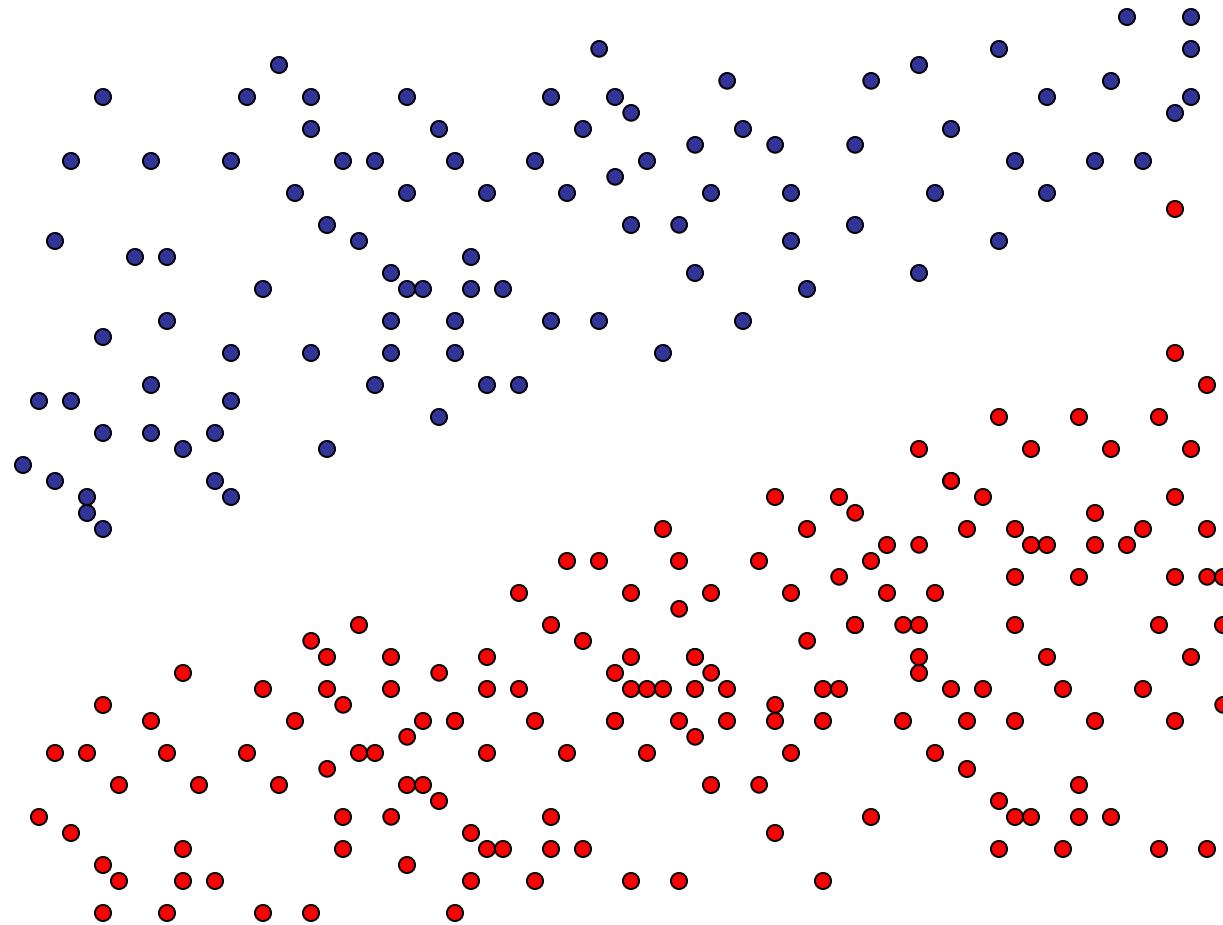
TECHNISCHE  
UNIVERSITÄT  
DARMSTADT



# Lineare Trennung: Minimierung des beobachteten Fehlers — Beispiel 2



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT



# Probleme der ERM

---

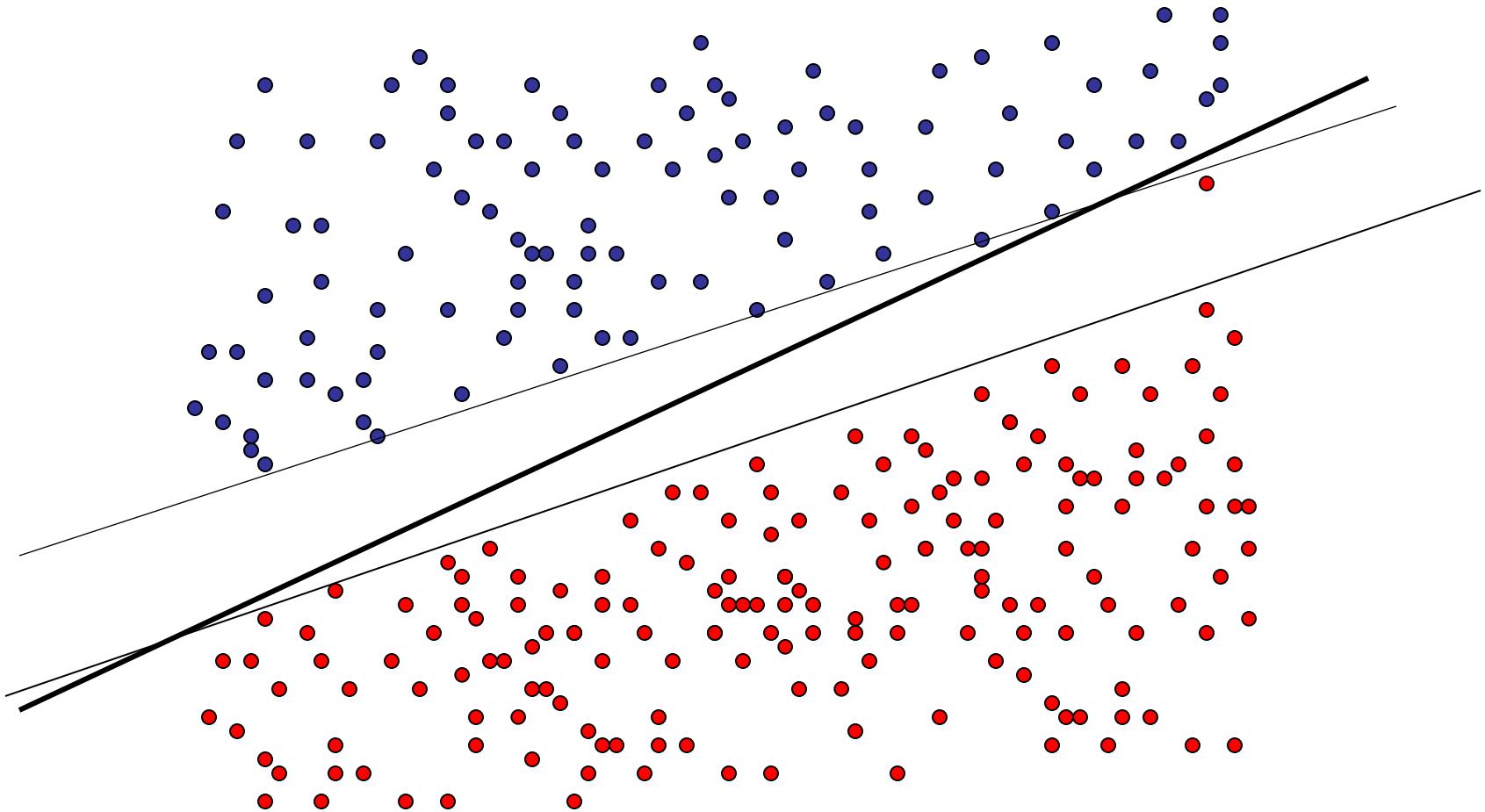
Aufgabe ist nicht eindeutig beschrieben: Mehrere Funktionen mit minimalem Fehler existieren. Welche wählen?

Overfitting: Verrauschte Daten und zu wenig Beispiele führen zu falschen Ergebnissen.

# Lineare Trennung: Minimierung des beobachteten Fehlers — Beispiel 3



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT



# Stützvektormethode: Grundlagen



## Zwei-Klassen-Problem:

- Trainingsdaten  $(x_1, y_1), \dots, (x_m, y_m)$ ,  $x_m \in X$ ,  $y_m \in \{+1, -1\}$
- Ähnlichkeit eines neuen  $x_i$  bestimmt  $y_i$
- Ähnlichkeitsmaß  $k: X \times X \rightarrow \mathbb{R}$

$$(x, x') \rightarrow k(x, x')$$

z.B. Skalarprodukt (auch inneres Produkt)

$$x^*x' := \sum [x]_i [x']_i$$

# Stützvektormethode: Grundlagen



**Skalarprodukt**  $x^*y$ : Seien  $x$  und  $y$  Vektoren aus  $\mathbb{R}^p$

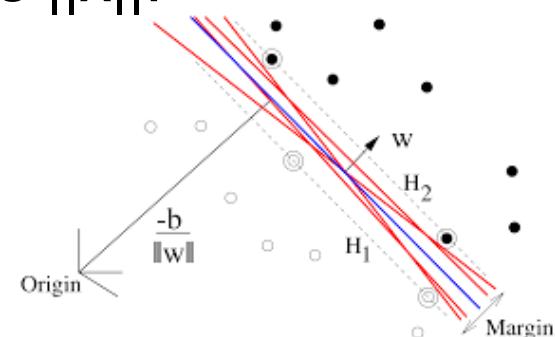
$$x^*y = \sum_{i=1}^p [x]_i [y]_i$$

**Euklidische Länge** (Betrag) eines Vektors  $\|x\|$ :

$$\|x\| = \sqrt{x^*x} = \left( \sum_{i=1}^p [x]_i^2 \right)^{\frac{1}{2}}$$

**Hyperebene H**: Sei  $w \neq 0$  der Normalenvektor und  $b \in \mathbb{R}$  der Bias

$$H(w, b) = \{x \mid w^*x + b = 0\}$$



Der **Normalenvektor  $w$**  beschreibt eine Gerade durch den Koordinatenursprung. Senkrecht zu dieser Geraden verlaufen Hyperebenen. Jede schneidet die Gerade in einer bestimmten Entfernung  $b/\|w\|$  vom Ursprung (gemessen in der Gegenrichtung zu  $w$ ). Diese Entfernung nennt man **Bias**.

# Warum Skalarprodukt?

- Cosinus des Winkels zwischen  $x$  und  $x'$ , wenn beide Vektoren auf die Länge 1 normiert sind.
- Abstand zwischen  $x$  und  $x'$  ist Länge des Differenzvektors.
- Voraussetzung: Beispiele sind Vektoren.
- Überführung in einen Raum mit Skalarprodukt  $\Phi : X \rightarrow \mathcal{H}$
- Wenn  $X$  bereits ein Raum mit Skalarprodukt ist, kann **nicht-lineare Abbildung**  $\Phi$  auch sinnvoll sein.

# Einfachster Lernalgorithmus:

„Assign a new point  $x$  to the class whose mean is closest“



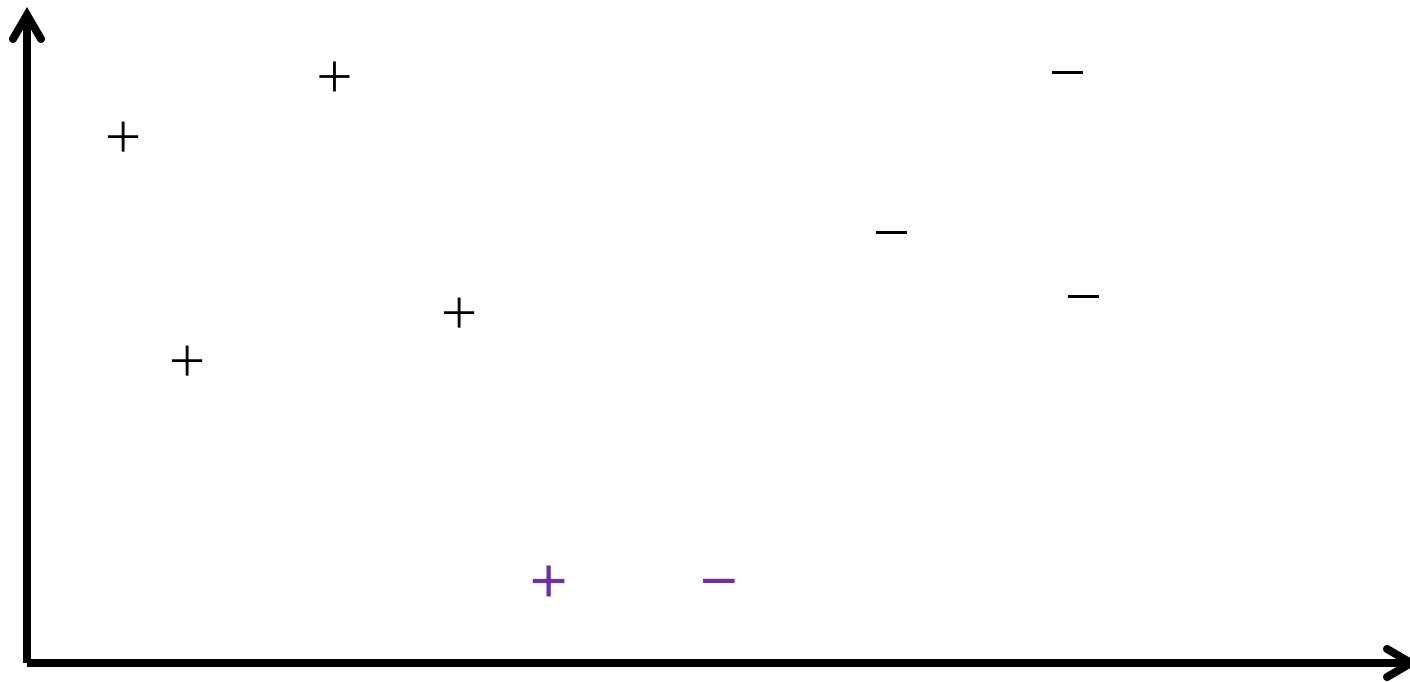
TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

- Annahme: Beispiele in einem Raum mit Skalarprodukt.
- Durchschnitt einer Klasse:  $c_+ = \frac{1}{m_+} \sum_{\{i|y_i=+1\}} x_i$  Anzahl positiver Beispiele:  $m_+$
- $c_- = \frac{1}{m_-} \sum_{\{i|y_i=-1\}} x_i$  Anzahl negativen Beispiele:  $m_-$
- In der Mitte liegt Punkt  $c := (c_+ + c_-)/2$
- Der Vektor  $\mathbf{x}-\mathbf{c}$  verbindet neue Beobachtung  $\mathbf{x}$  und  $\mathbf{c}$
- Ähnlichkeit zum Durchschnitt einer Klasse:  
**Winkel zwischen  $\mathbf{w} := \mathbf{c}_+ - \mathbf{c}_-$  und  $\mathbf{x}-\mathbf{c}$**
- Berechnen über Skalarprodukt!

# Lernalgorithmus im Bild



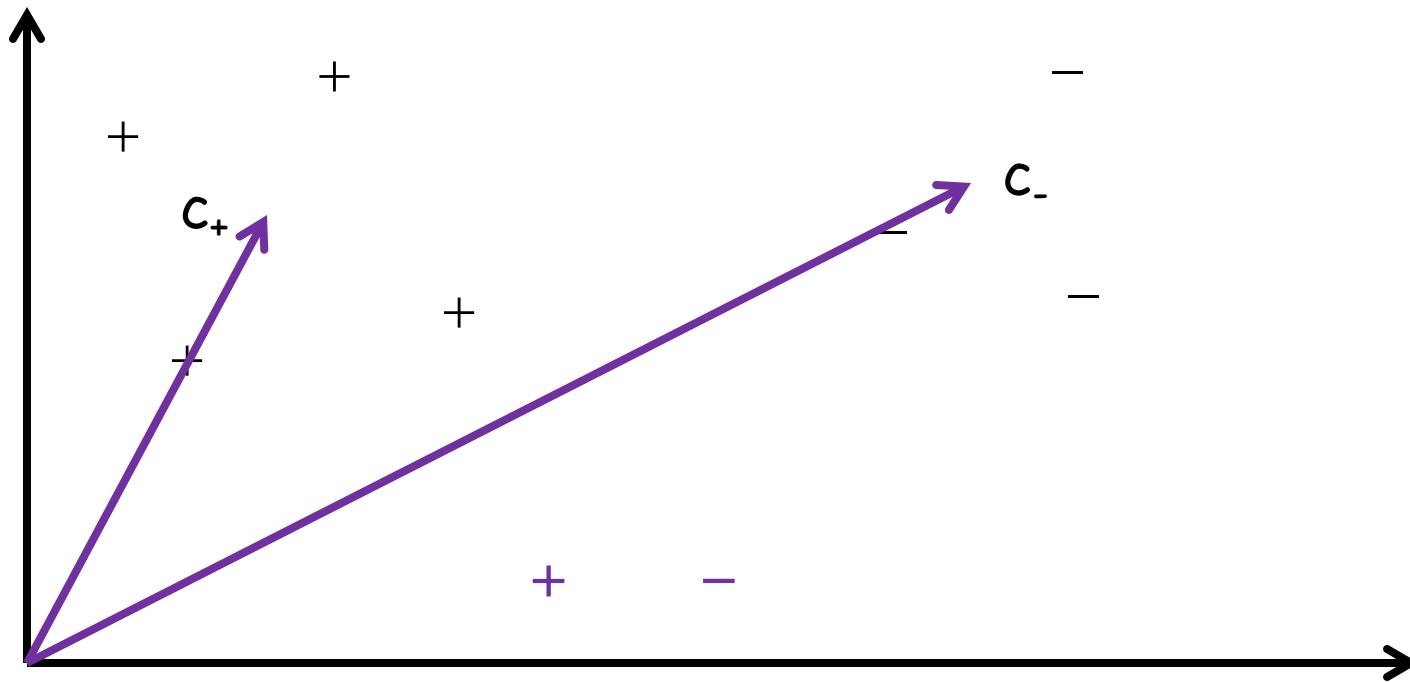
TECHNISCHE  
UNIVERSITÄT  
DARMSTADT



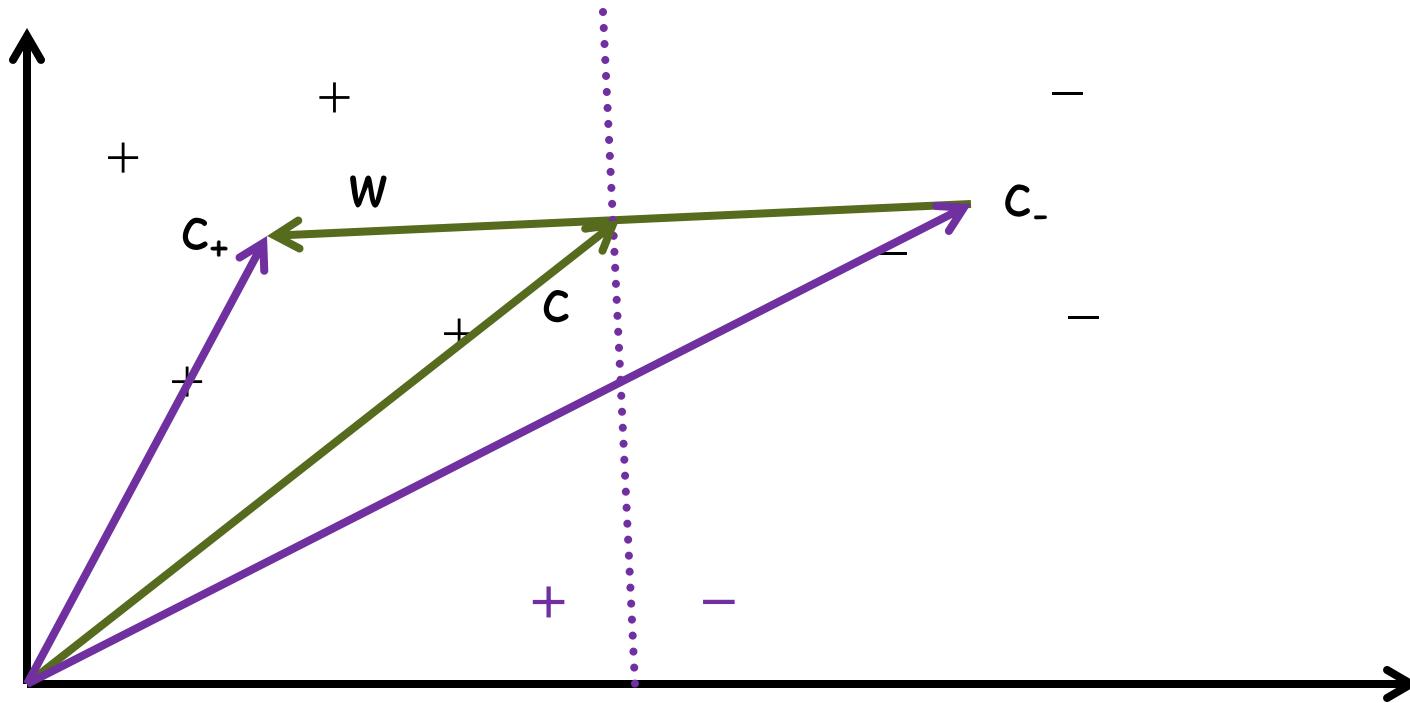
# Lernalgorithmus im Bild



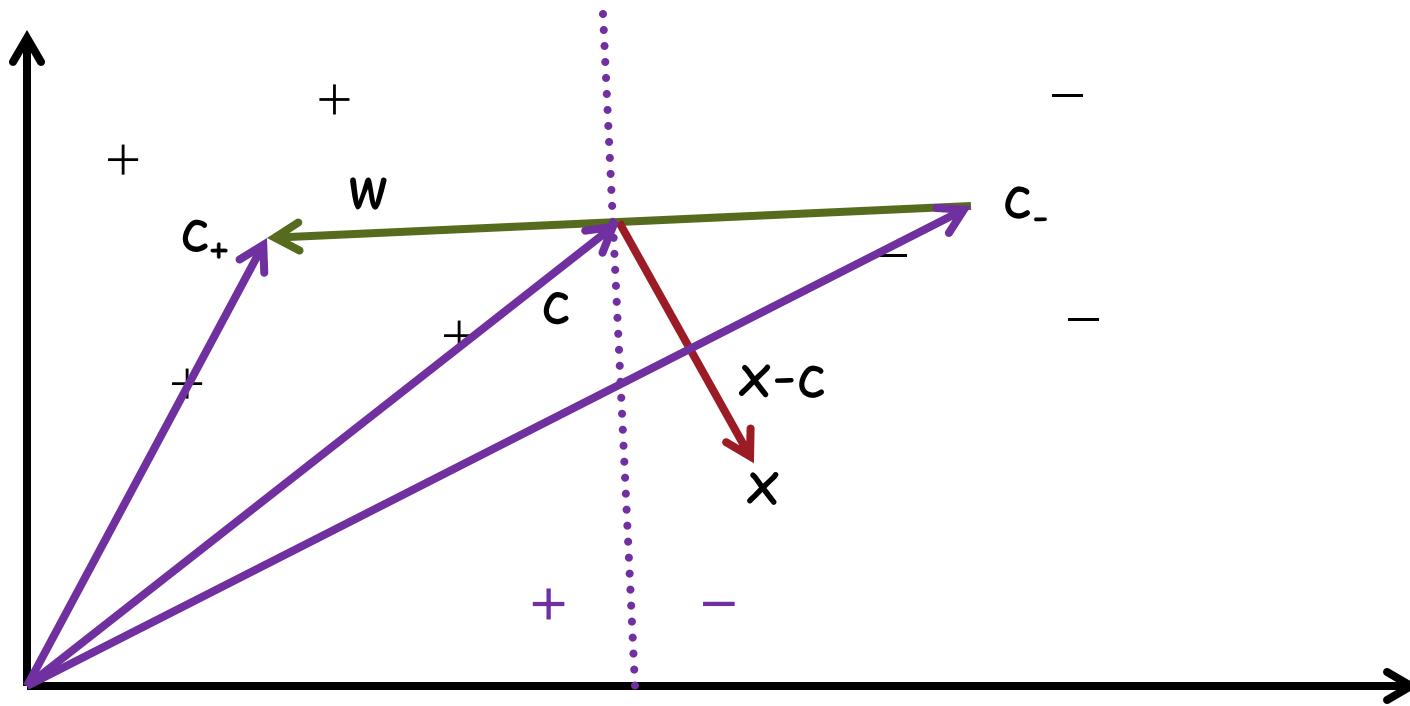
TECHNISCHE  
UNIVERSITÄT  
DARMSTADT



# Lernalgorithmus im Bild



# Lernalgorithmus im Bild



# Lernalgorithmus in Formeln

$$\begin{aligned}y &= \text{sign}((x - c) * w) \\&= \text{sign}((x - (c_+ + c_-)/2) * (c_+ - c_-)) \\&= \text{sign}\left((x * c_+) - (x * c_-) - \frac{1}{2}c_+^2 - \frac{1}{2}c_+ * c_- + \frac{1}{2}c_-^2 + \frac{1}{2}c_+ * c_-\right) \\&= \text{sign}\left((x * c_+) - (x * c_-) + \frac{1}{2}(\|c_-\|^2 - \|c_+\|^2)\right) \\&= \text{sign}((x * c_+) - (x * c_-) + b)\end{aligned}$$

Wir setzen nun die Mittelwerte für  $c_+$  und  $c_-$  ein:

$$\begin{aligned}y &= \text{sign} \left( \frac{1}{m_+} \sum_{\{i|y_i=+\}} x^* x_i - \frac{1}{m_-} \sum_{\{i|y_i=+\}} x^* x_i + b \right) \\&= \text{sign} \left( \frac{1}{m_+} \sum_{\{i|y_i=+\}} k(x, x_i) - \frac{1}{m_-} \sum_{\{i|y_i=+\}} k(x, x_i) + b \right)\end{aligned}$$

**Die neue Beobachtung  $x$  wird also mit allen Trainingsbeispielen  $x_i$  verglichen.**

„Assign a new point  $x$  to the class whose mean is closest“

**Fast ...**

**... wäre das schon die Stützvektormethode.**



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

**Aber:**

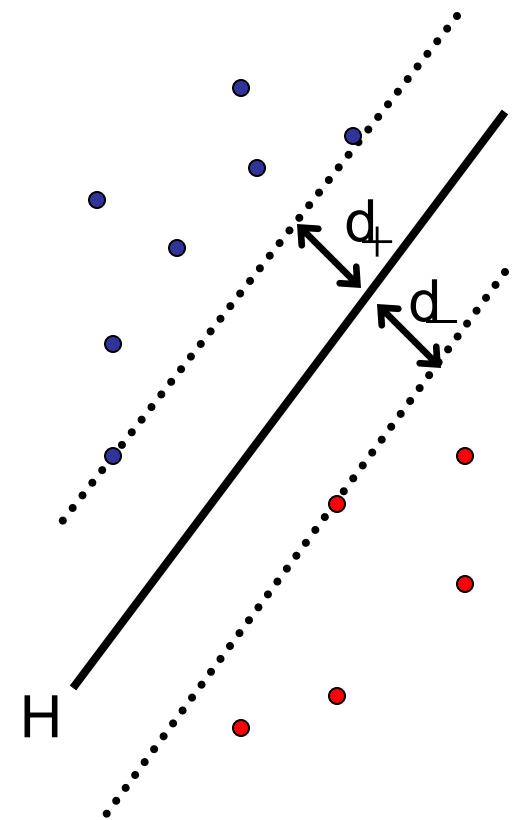
- Nur den Mittelpunkt der Beispiele einer Klasse zu berechnen ist zu einfach, um ein ordentliches  $w$  zu bekommen, das gut funktioniert.
- Man erhält so nicht die **optimale** Hyperebene.

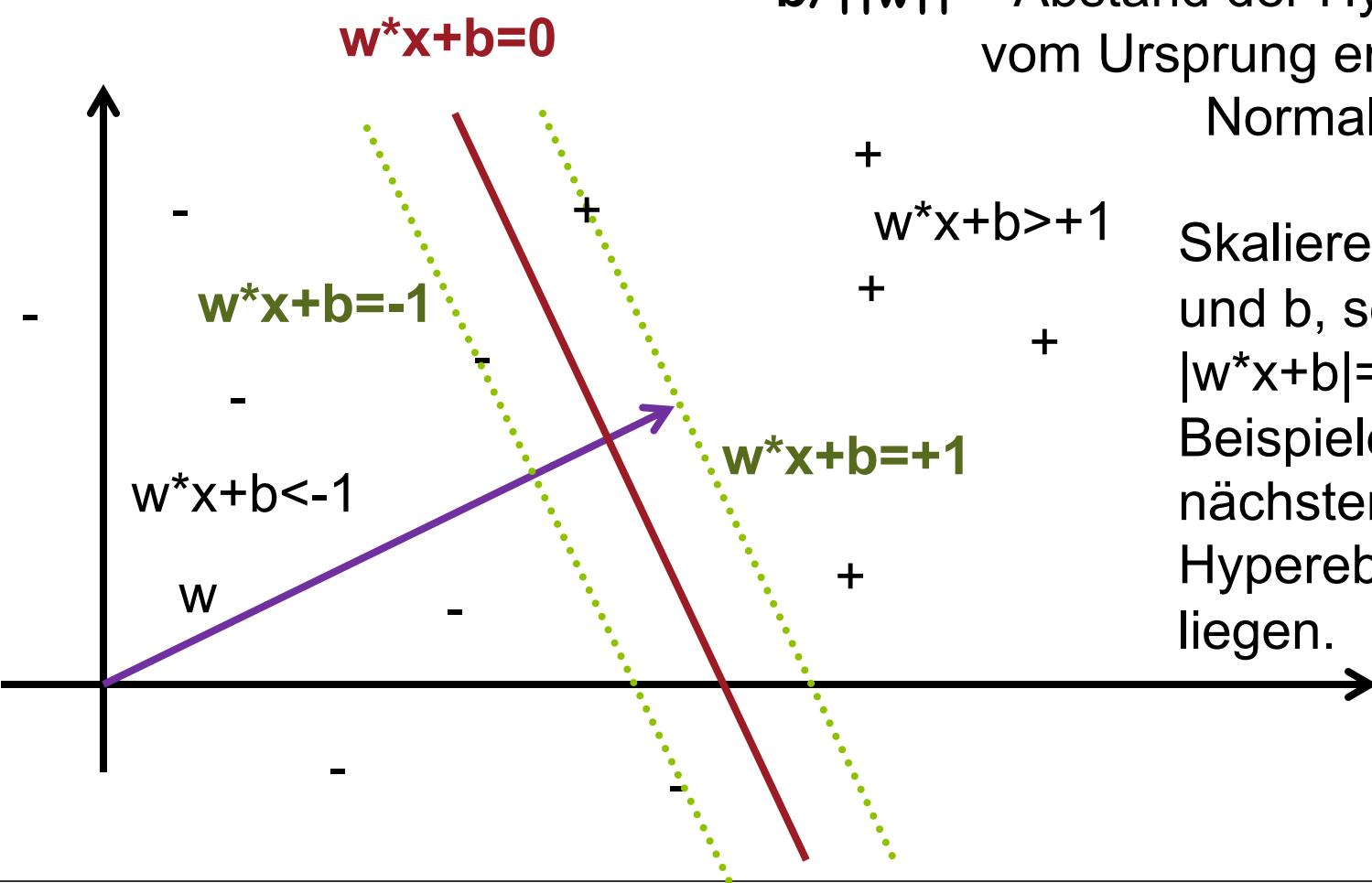
# Aber wann ist eine Hyperebene denn optimal?

- Beispiele heißen **linear trennbar**, wenn es eine Hyperebene  $H$  gibt, die die positiven und negativen Beispiele voneinander trennt.
- $H$  heißt **optimale Hyperebene**, wenn ihr Abstand  $d$  zum nächsten positiven und zum nächsten negativen Beispiel **maximal** ist.
- **Satz:** Es existiert eine eindeutig bestimmte optimale Hyperebene.

Der Normalenvektor steht senkrecht auf allen Vektoren der Hyperebene. Es gilt:

$$w^* x + b \begin{cases} > 0 & \text{falls } x \text{ im positiven Raum} \\ = 0 & \text{falls } x \text{ auf } H \\ < 0 & \text{falls } x \text{ im negativen Raum} \end{cases}$$





# Wie breit ist die Margin?

## Die Margin und der Normalenvektor

- Sein  $x_0$  ein Punkt in der Hyperebene  $w^*x+b=-1$ .
- Per constructionem,  $w/\|w\|$  ist ein Normalenvektor der Länge 1 für die Hyperebene  $w^*x+b=-1$ .
- Ebenso (per constructionem) muss der Punkt  $x_0+d*w/\|w\|$  in der Hyperebene  $w^*x+b=1$  liegen.
- Daher:

$$\begin{aligned} w(x_0+d*w/\|w\|)+b &= 1 \\ \Rightarrow w^*x_0 + d*w^*w/\|w\|+b &= 1 \\ \Rightarrow w^*x_0 + d*\|w\|^2/\|w\|+b &= 1 \\ \Rightarrow w^*x_0 + d*\|w\| +b &= 1 \\ \Rightarrow w^*x_0 + b &= 1 - d*\|w\| \\ \Rightarrow -1 &= 1 - d * \|w\| \\ \Rightarrow d &= 2 / \|w\| \end{aligned}$$

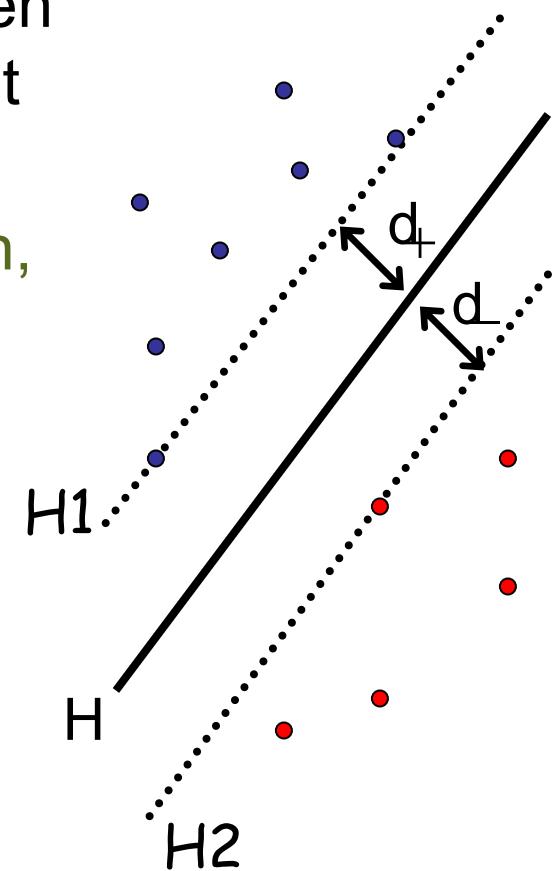
**Satz:  $\|w\| = 2/d$ , d = Breite der optimalen Margin bzgl. der Beispiele.**

# Optimal Margin = Maximale Margin

- H1 und H2 sind parallel, haben denselben Normalenvektor w. Per Konstruktion liegt kein Beispiel zwischen H1 und H2.
- Um die Margin  $d=2 / \|w\|$  zu maximieren, müssen wir  $\|w\|$  minimieren.

Achtung, wir wollen die richtigen Klassifikationen für unsere Trainingsdaten.  
Daher haben wir **Nebenbedingungen**, die eingehalten werden müssen :

$$\forall i : y_i(x_i * w + b) - 1 \geq 0$$



# Stützvektormethode: Optimierungsaufgabe



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

Minimiere  $\|w\|^2$

so dass für alle  $i$  gilt:

$$f(x_i) = w^*x_i + b \geq 1 \quad \text{für } y_i = 1 \text{ und}$$

$$f(x_i) = w^*x_i + b \leq -1 \quad \text{für } y_i = -1$$

Äquivalente Nebenbedingungen:  $y_i^*f(x_i) - 1 \geq 0$

Das ist ein konvexes, quadratisches Optimierungsproblem, das eindeutig in  $O(n^3)$  für  $n$  Trainingsbeispiele lösbar ist.

# Stützvektormethode: Primales Problem



Die Nebenbedingungen  $g_i$  sind gegeben durch

$$g_i(\vec{\beta}, \beta_0) = y_i (\langle \vec{x}_i, \vec{\beta} \rangle + \beta_0) - 1 \geq 0 \quad \forall \vec{x}_i \in \mathbf{X}$$

Die Formulierung des Optimierungsproblems nach Lagrange wird auch als **Primales Problem** bezeichnet:

## Primales Problem

Die Funktion

$$L_P(\vec{\beta}, \beta_0, \vec{\alpha}) = \frac{1}{2} \|\vec{\beta}\|^2 - \sum_{i=1}^N \alpha_i (y_i (\langle \vec{x}_i, \vec{\beta} \rangle + \beta_0) - 1)$$

soll  $L_P$  bezüglich  $\vec{\beta}$  und  $\beta_0$  *minimiert* und bezüglich  $\vec{\alpha}$  *maximiert* werden!

Es reicht also die Zielfunktion bzgl. der betas zu minimieren, weil dann durch das Vorzeichen “-” vor der Summe alpha maximiert wird.

# SVM Optimierungsproblem

(mittels Lagrange/KKT und Dualität, hier nicht angesprochen)



Achtung, wir behandeln die KKT hier nicht im Detail. Sie sind eine Verallgemeinerung der "Gradient=0" Bedingung ohne Nebenbedingungen und der Lagrange-Multiplikatoren, also von Optimierungsproblemen mit Nebenbedingungen

Durch die partiellen Ableitung nach  $\vec{\beta}$  und  $\beta_0$  erhalten wir

$$\frac{\partial}{\partial \vec{\beta}} L_P(\vec{\beta}, \beta_0, \vec{\alpha}) = \vec{\beta} - \sum_i \alpha_i y_i \vec{x}_i \quad \text{und} \quad \frac{\partial}{\partial \beta_0} L_P(\vec{\beta}, \beta_0, \vec{\alpha}) = - \sum_i \alpha_i y_i$$

Nullsetzen der Ableitungen und die Berücksichtigung der Nebenbedingungen führt zu den KKT-Bedingungen für eine Lösung für  $L_P$ :

Herüberziehen von beta

$$\vec{\beta} = \sum_{i=1}^N \alpha_i y_i \vec{x}_i \quad \text{und} \quad \sum_{i=1}^N \alpha_i y_i = 0$$

Vorzeichen weg, weil wir mit "-1" multiplizieren können

$$(*) \quad \alpha_i \geq 0 \quad \forall i = 1, \dots, N$$

$$\alpha_i (y_i (\langle \vec{x}_i, \vec{\beta} \rangle + \beta_0) - 1) = 0 \quad \forall i = 1, \dots, N$$

**Sattelpunkt-Bedingung laut KKT:** Punkte mit  $\alpha_i > 0$  liegen direkt auf dem Rand, sie sind also Stützvektoren. Alle restlichen Trainingspunkte haben keinen Einfluss, weil  $\alpha_i = 0$

## SVM Optimierungsproblem

(mittels Lagrange/KKT und Dualität, hier nicht angesprochen)



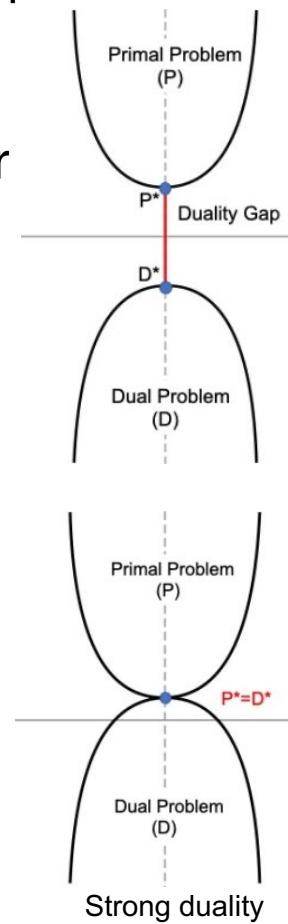
Das primale Problem soll bezüglich  $\vec{\beta}$  und  $\beta_0$  minimiert und bezüglich  $\vec{\alpha}$  maximiert werden.

Setzen wir die Bedingungen an  $\frac{\partial L_P}{\partial \vec{\beta}}$  und  $\frac{\partial L_P}{\partial \beta_0}$  in  $L_p$  ein, so erhalten wir den so genannten dualen Lagrange-Ausdruck  $L_D(\vec{\alpha})$

- Der duale Lagrange-Ausdruck  $L(\vec{\alpha})$  soll maximiert werden.
- Das Minimum des ursprünglichen Optimierungsproblems tritt genau bei den selben Werten von  $\vec{\beta}, \beta_0, \vec{\alpha}$  auf wie das Maximum des dualen Problems.

Normalerweise wird das duale Problem in der Praxis optimiert.

Warum? **Das duale Problem erlaubt zum Beispiel die einfache Benutzung von Kernfunktionen**, wie wir sie später kennenlernen werden.





# SVM Optimierungsproblem

(mittels Lagrange/KKT und Dualität, hier nicht angesprochen)

Maximiere

unter  $0 \leq \alpha_i$  für alle i und  $\sum \alpha_i y_i = 0$

Duales Problem

$$L(\alpha) = \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n y_i y_j \alpha_i \alpha_j (x_i * x_j)$$

Für jedes Beispiel gibt es ein  $\alpha$  in der Lösung.

- $0 = \alpha_i$  heißt, dass das Beispiel  $x_i$  im passenden Halbraum liegt.
- $0 < \alpha_i$  heißt, dass das Beispiel  $x_i$  auf H1 oder H2 liegt (Stützvektor).

Es gilt  $w = \sum \alpha_i y_i x_i$  und somit  $f(x) = \sum \alpha_i y_i (x_i * x) + b$ . Also ist der beste Normalenvektor  $w$  eine Linearkombination von Stützvektoren ( $\alpha_i \neq 0$ ).

# Was wissen wir jetzt?

Maximieren der Margins einer Hyperebene ergibt eine eindeutige Festlegung der optimalen trennenden Hyperebene.

Dazu minimieren wir die Länge des Normalenvektors  $w$ .

- Formulierung als Lagrange-Funktion
- Formulierung als duales Optimierungsproblem

Das Lernergebnis ist eine Linearkombination von Stützvektoren.

Mit den Beispielen müssen wir nur noch das Skalarprodukt rechnen.

# Was wissen wir jetzt?

Das Lagrange-Optimierungs-Problem ist definiert als:

$$L_P = \frac{1}{2} \|\vec{\beta}\|^2 - \sum_{i=1}^N \alpha_i \left[ y_i (\langle \vec{x}_i, \vec{\beta} \rangle + \beta_0) - 1 \right]$$

mit den *Lagrange-Multiplikatoren*  $\alpha_i \geq 0$ .

Notwendige Bedingung für ein Minimum liefern die Ableitungen nach  $\vec{\beta}$  und  $\beta_0$

$$\frac{\partial L_P}{\partial \vec{\beta}} = \vec{\beta} - \sum_{i=1}^N \alpha_i y_i \vec{x}_i \quad \text{und} \quad \frac{\partial L_P}{\partial \beta_0} = \sum_{i=1}^N \alpha_i y_i$$

Diese führen zum *dualen Problem*

$$L_D = \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i=1}^N \sum_{i'=1}^N \alpha_i \alpha_{i'} y_i y_{i'} \langle \vec{x}_i, \vec{x}_{i'} \rangle$$

# Nicht linear trennbare Daten



In der Praxis sind linear trennbare Daten selten.

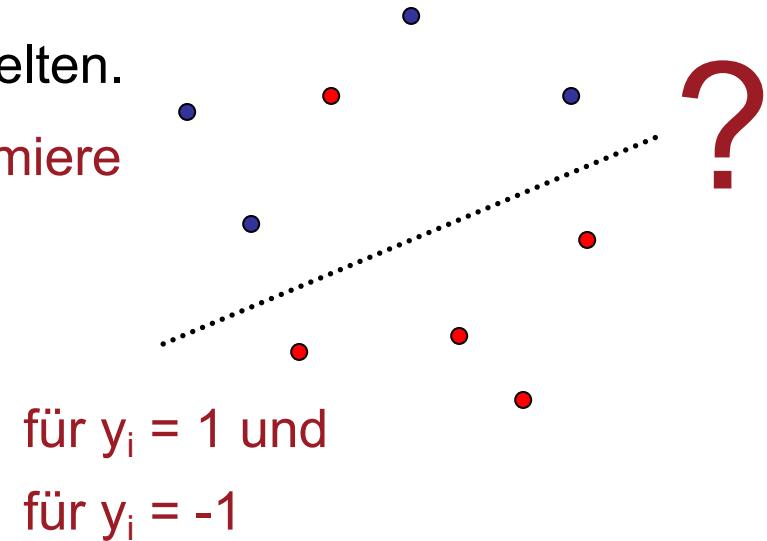
Wähle Kosten  $C \in \mathbb{R}_{>0}$  für Fehler und minimiere

$$\|w\|^2 + C \sum_{i=1}^n \xi_i$$

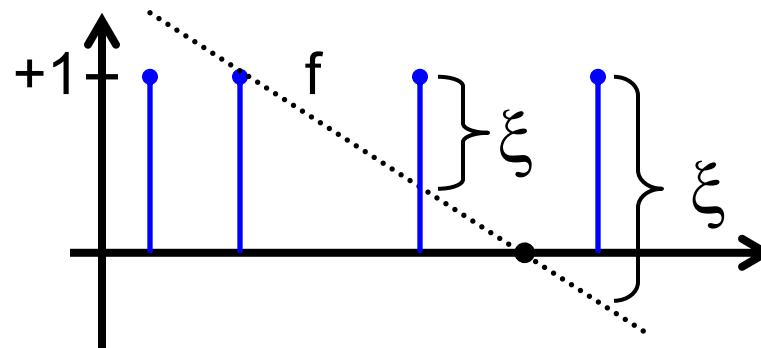
so dass für alle  $i$  gilt:

$$f(x_i) = w^*x_i + b \geq 1 - \xi_i$$

$$f(x_i) = w^*x_i + b \leq -1 + \xi_i$$

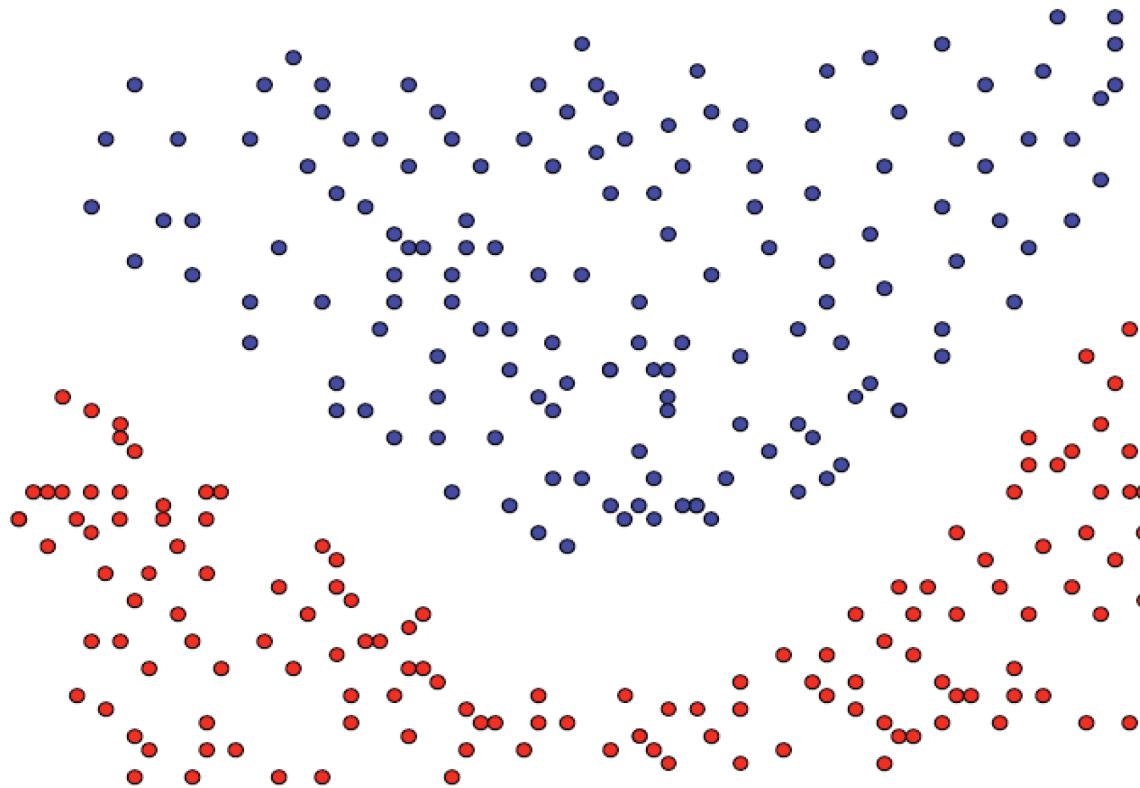


Äquivalent:  $y_i * f(x_i) \geq 1 - \xi_i$

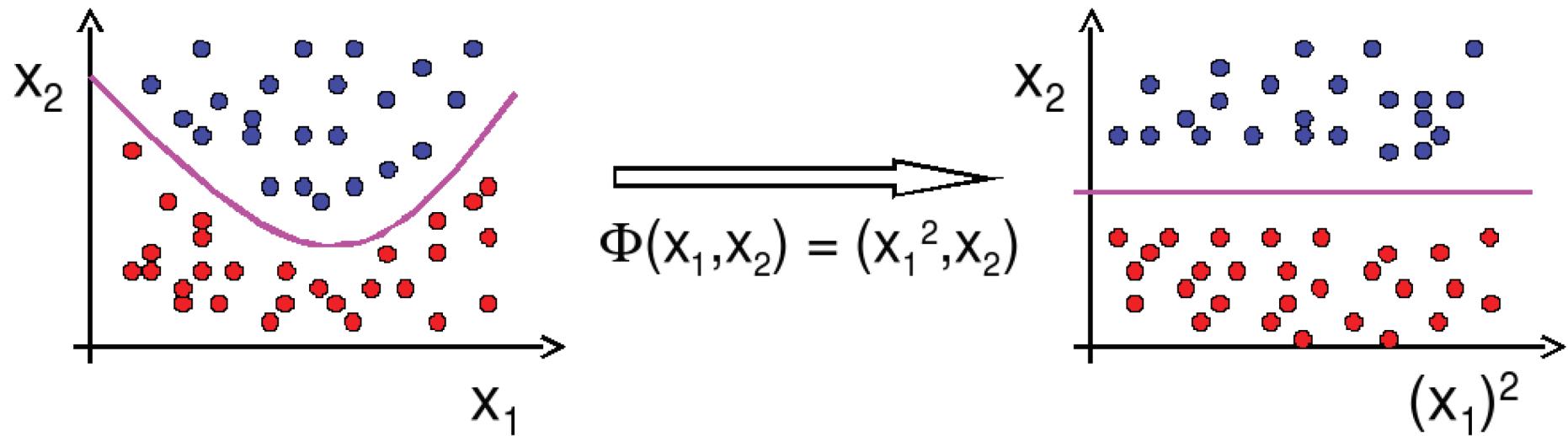


# Reichen Hyperebenen im Merkmalsraum?

Nein! Nicht-lineare Daten. Eine (weiche) trennende Hyperebene im Merkmalsraum reicht hier nicht aus.



# Transformation in ein lineares Problem



Erinnerung, die SVM benötigt nur das Skalarprodukt.

Idee: Ersetze Transformation  $\Phi$  und Skalarprodukt durch Kernfunktion

$$K(\vec{x}_1, \vec{x}_2) := \langle \Phi(\vec{x}_1), \Phi(\vec{x}_2) \rangle$$

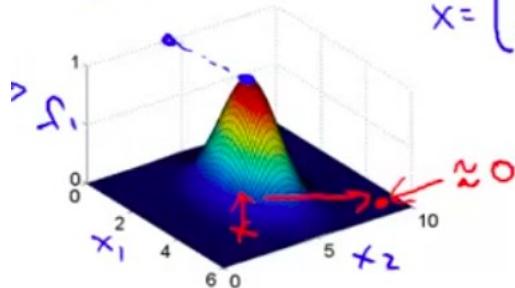
# Transformation in ein lineares Problem



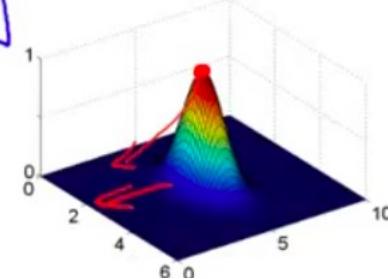
**Example:**

$$l^{(1)} = \begin{bmatrix} 3 \\ 5 \end{bmatrix}, \quad f_1 = \exp\left(-\frac{\|x-l^{(1)}\|^2}{2\sigma^2}\right)$$

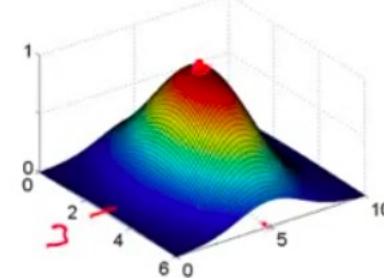
$$\rightarrow \sigma^2 = 1$$



$$x = \begin{bmatrix} 3 \\ 5 \end{bmatrix} \quad \sigma^2 = 0.5$$



$$\sigma^2 = 3$$



Idee: Ersetze Transformation  $\Phi$  und Skalarprodukt durch Kernfunktion

$$K(\vec{x}_1, \vec{x}_2) := \langle \Phi(\vec{x}_1), \Phi(\vec{x}_2) \rangle$$

# Kern-Trick



Aber ist das nicht zu aufwendig? Wollen wir wirklich explizit in einem hoch-dimensionalen Innenproduktraum arbeiten?

Angabe von  $\Phi$  nicht nötig, einzige Bedingung:

*Kernmatrix  $\mathbf{K} = (K(\vec{x}_i, \vec{x}_j))_{i,j=1\dots N}$  muss symmetrisch und positiv semi-definit sein, d.h.  $\vec{z}^T \mathbf{K} \vec{z} \geq 0$ .*

Manchmal nennt man die Kernmatrix auch Gram-Matrix.

Man muss lediglich bei der Berechnung des Skalarprodukts die Kernfunktion berücksichtigen.

$$L_D(\alpha) = \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N y_i y_j \alpha_i \alpha_j K(\vec{x}_i, \vec{x}_j)$$

$$\alpha_i \geq 0 \quad \forall i = 1, \dots, m \text{ und } \sum_{i=1}^m \alpha_i y_i = 0$$

# Was wissen wir jetzt?



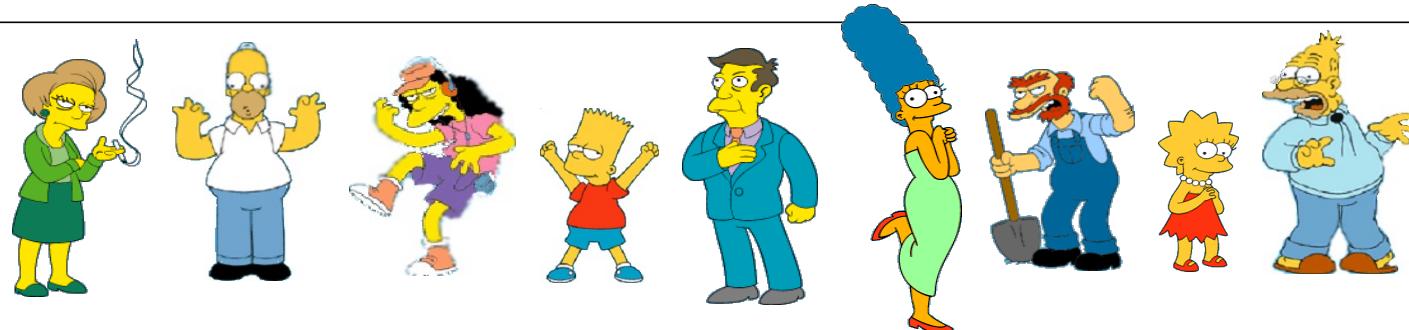
TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

- Weiche Margin mittels Strafterme (Slack)
- Kernfunktionen - eine Transformation, die man nicht erst durchführen und dann mit ihr rechnen muss, sondern bei der nur das Skalarprodukt gerechnet wird.

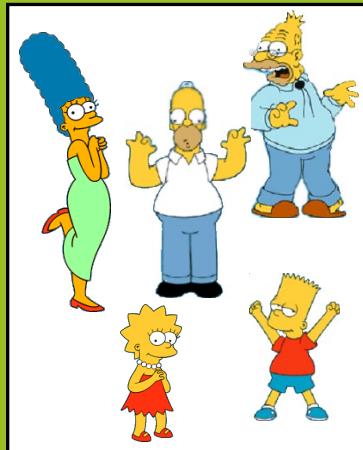
# Cluster-Analyse: Wie würden Sie die Simpsons gruppieren?



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT



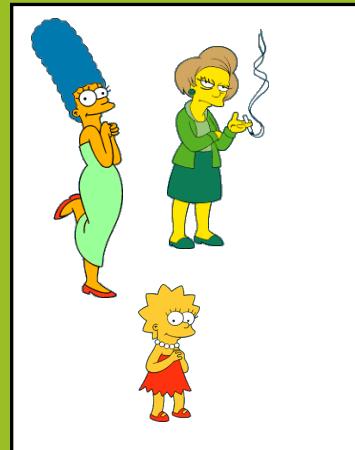
## Clusteranalyse ist subjektiv



Die Simpsons



Schulangestellte



Weiblich



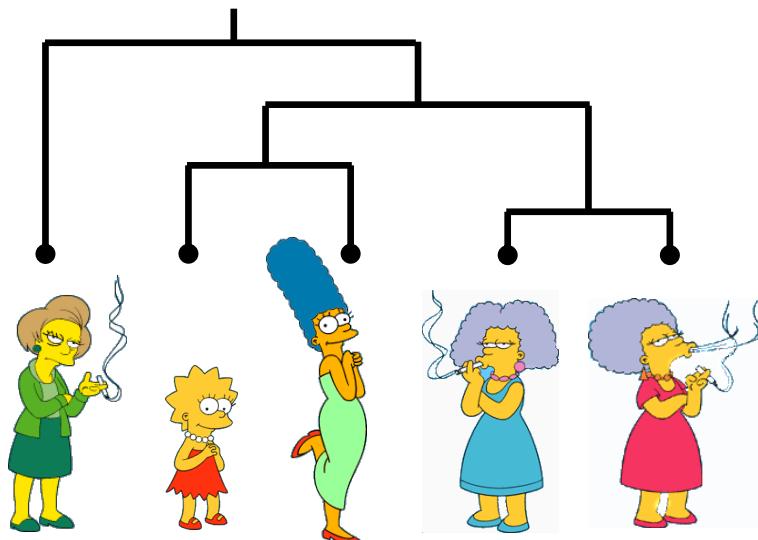
Männlich

# Zwei Arten der Clusteranalyse

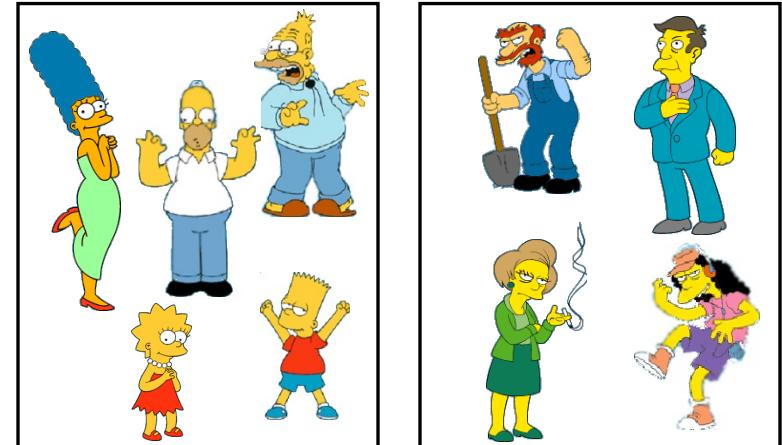
**Partitionierungsansätze:** Man konstruiert Partitionierungen (Aufteilungen) der Daten und bewertet sie mittels einer Bewertungsfunktion

**Hierarchische Ansätze:** Konstruiere eine hierarchische Aufteilung der Daten anhand eines Kriteriums

## Hierarchisch



## Partition



Beide Arten benötigen im Wesentlichen eine Distanzfunktion:

- $D(A,B) = D(B,A)$  *Symmetrie*
- $D(A,A) = 0$  *Konstanz der Selbstähnlichkeit*
- $D(A,B) = 0$  If  $A = B$  *Positiv*
- $D(A,B) \leq D(A,C) + D(B,C)$  *Dreiecksunähnlichkeit*

*“We know it when we see it”*

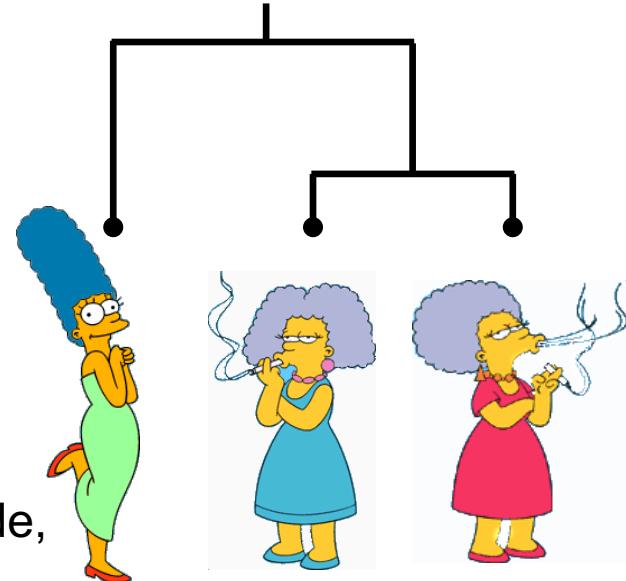
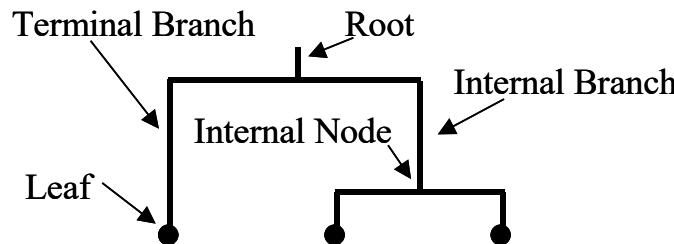


# Dendrogramme



Ähnlichkeit können wir auch durch einen Baum darstellen:

Die Ähnlichkeit zweier Objekte wird in einem Dendrogram durch die Höhe (von den Blättern ausgesehen) des niedrigsten internen Knoten ausgedrückt, den beide Objekte gemeinsam haben



Wenn zwei Cluster auf Höhe  $x$  verschmolzen werde,  
dann war der Abstand zwischen den Clustern  $x$

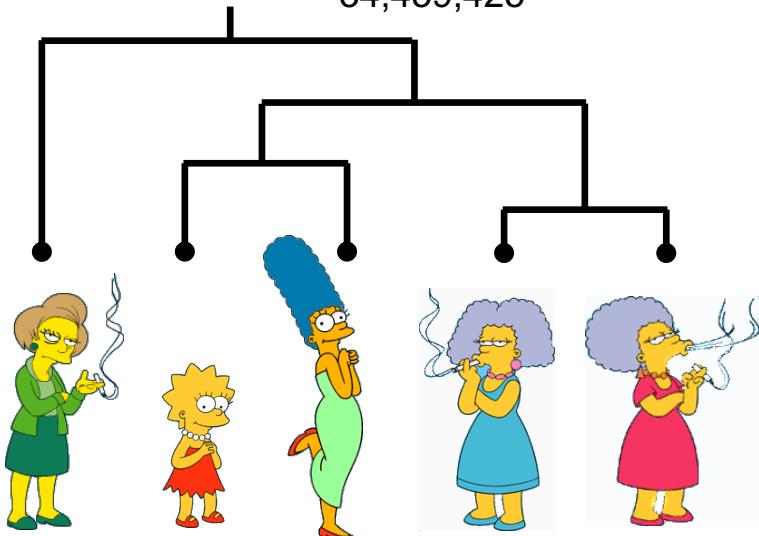
# Hierarchisches Clustering



Murtagh: Counting dendograms: A survey. Discrete Applied Mathematics 7(2)=:191-199 1984

Die Zahl der Dendrogramme mit  $n$  Blättern  
 $= (2n - 3)! / [(2^{n-2}) (n - 2)!]$

Zahl der Blätter	Zahl der möglichen Dendrogramme
2	1
3	3
4	15
5	105
...	...
10	34,459,425



Zahl der Dendrogramme steigt sehr schnelle. Weil wir nicht alle durchtesten können, müssen wir uns auf Heuristiken beschränken:

**Bottom-Up (Agglomerativ):** Anfangs ist jedes Objekt sein eigenes Cluster. Finde die beiden Cluster, die sich am ähnlichsten sind, und vereinige (merge) sie. Wiederhole das solange, bis es nur noch ein Cluster gibt.

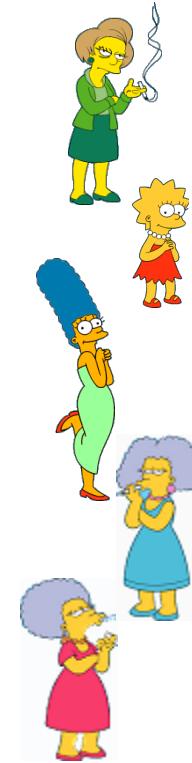
**Top-Down (Aufteilend):** Anfangs sind alle Objekte in einem Cluster. Finde den besten Split und führe diesen aus. Wiederhole das so oft, bis die Cluster nur noch aus einem Objekt bestehen

# Beispiel Bottom-Up

Wir haben eine Distanzmatrix,  
die alle paarweisen Distanzen  
enthält

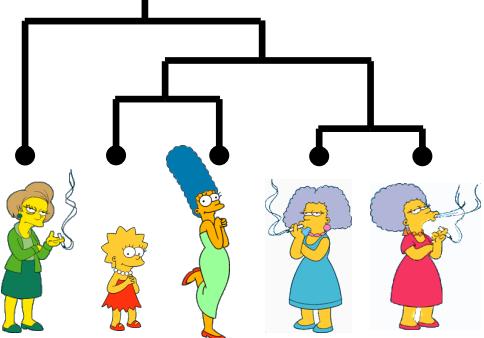
$$D(\text{Marge, Lisa}) = 8$$

$$D(\text{Marge, Marge}) = 1$$



0	8	8	7	7
	0	2	4	4
		0	3	3
			0	1
				0

(4)

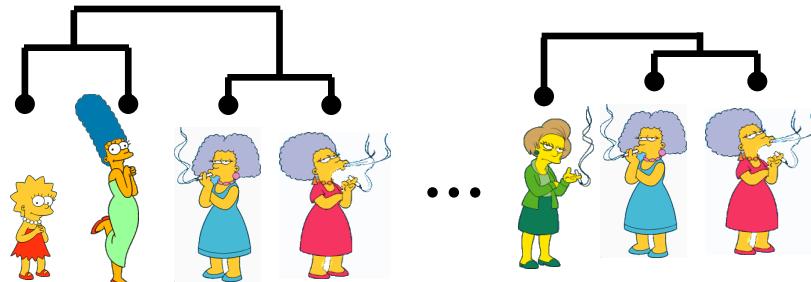


## Beispiel Bottom-Up

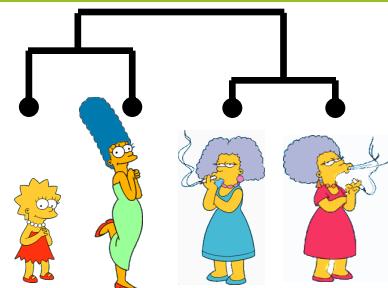
Anfangs ist jedes Objekt sein eigenes Cluster. (1) Finde die beiden Cluster, die sich am ähnlichsten sind, und vereinige (merge) sie. Wiederhole (2,3) das solange, bis es nur noch ein Cluster gibt (4).

(3)

Betrachte alle möglichen Vereinigungen

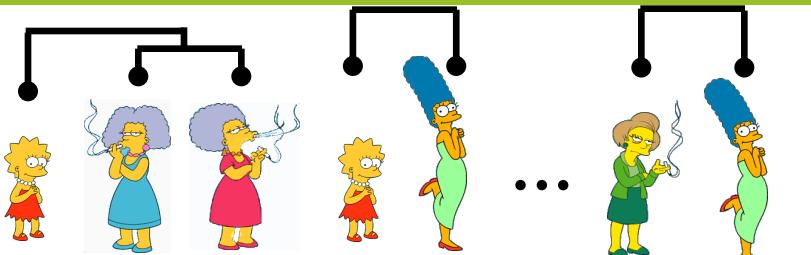


Wähle die beste

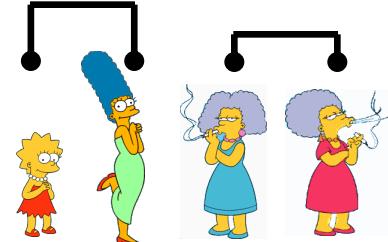


(2)

Betrachte alle möglichen Vereinigungen



Wähle die beste



(1)

Betrachte alle möglichen Vereinigungen



Wähle die beste





# Dendograms / Hierarchiches Clustering

## Pedro

Petros (Greek), Peter (English), Piotr (Polish), Peadar (Irish), Pierre (French), Peder (Danish), Peka (Hawaiian), Pietro (Italian), Piero (Italian Alternative), Petr (Czech), Pyotr (Russian)

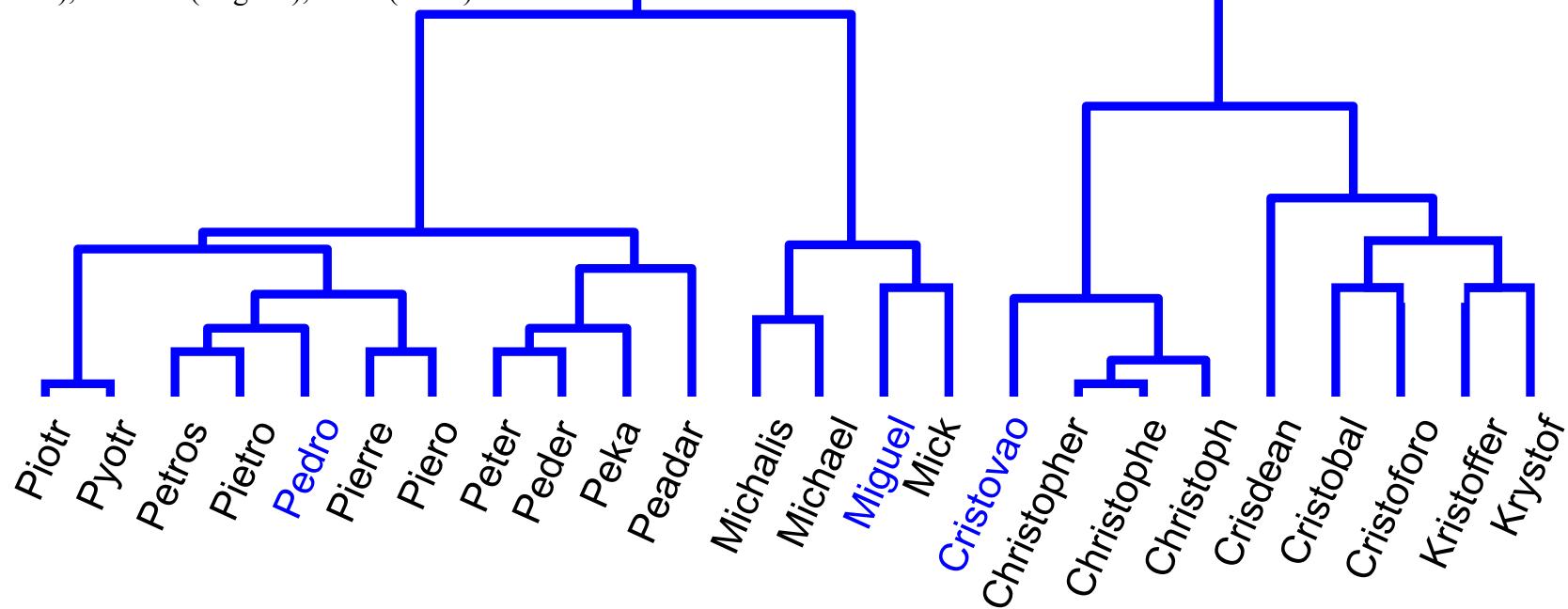
## Cristovao

Christoph (German), Christophe (French), Cristobal (Spanish), Cristoforo (Italian), Kristoffer (Scandinavian), Krystof (Czech), Christopher (English)

## Miguel

Michalis (Greek), Michael (English), Mick (Irish!)

Distanz = Wieviele Editieroperationen brauchen wir um String A in String B zu überführen ?

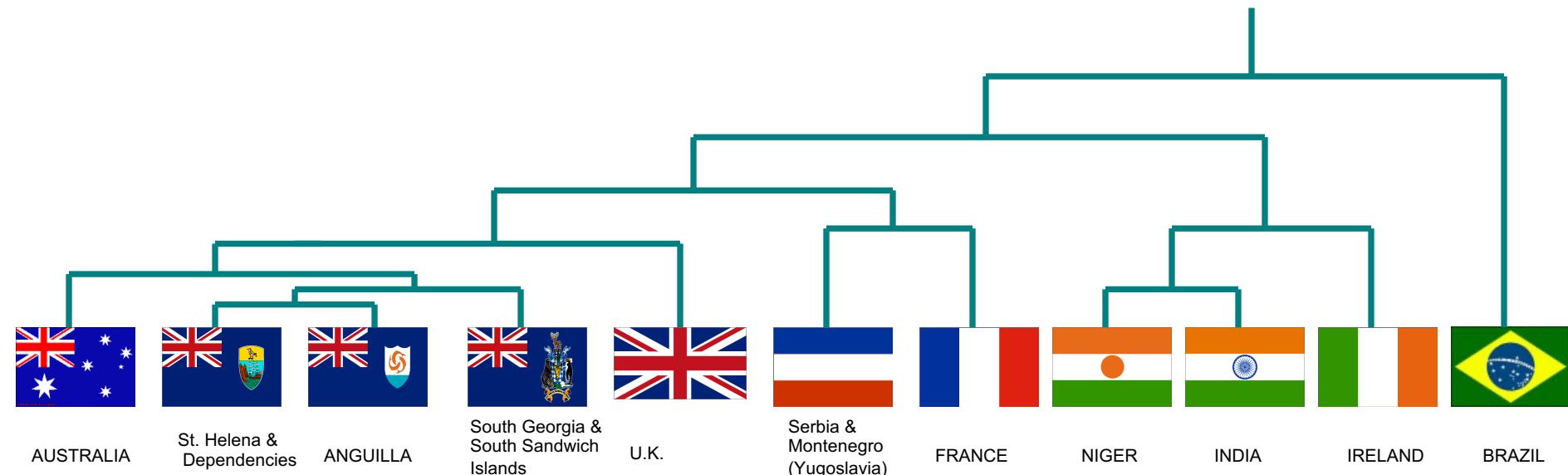


# Hierarchien können Strukturen aufdecken, aber auch vortäuschen



Die dichte Gruppe um Australien macht Sinn. Es sind alles Ländern aus den ehemaligen Britischen Kolonien.

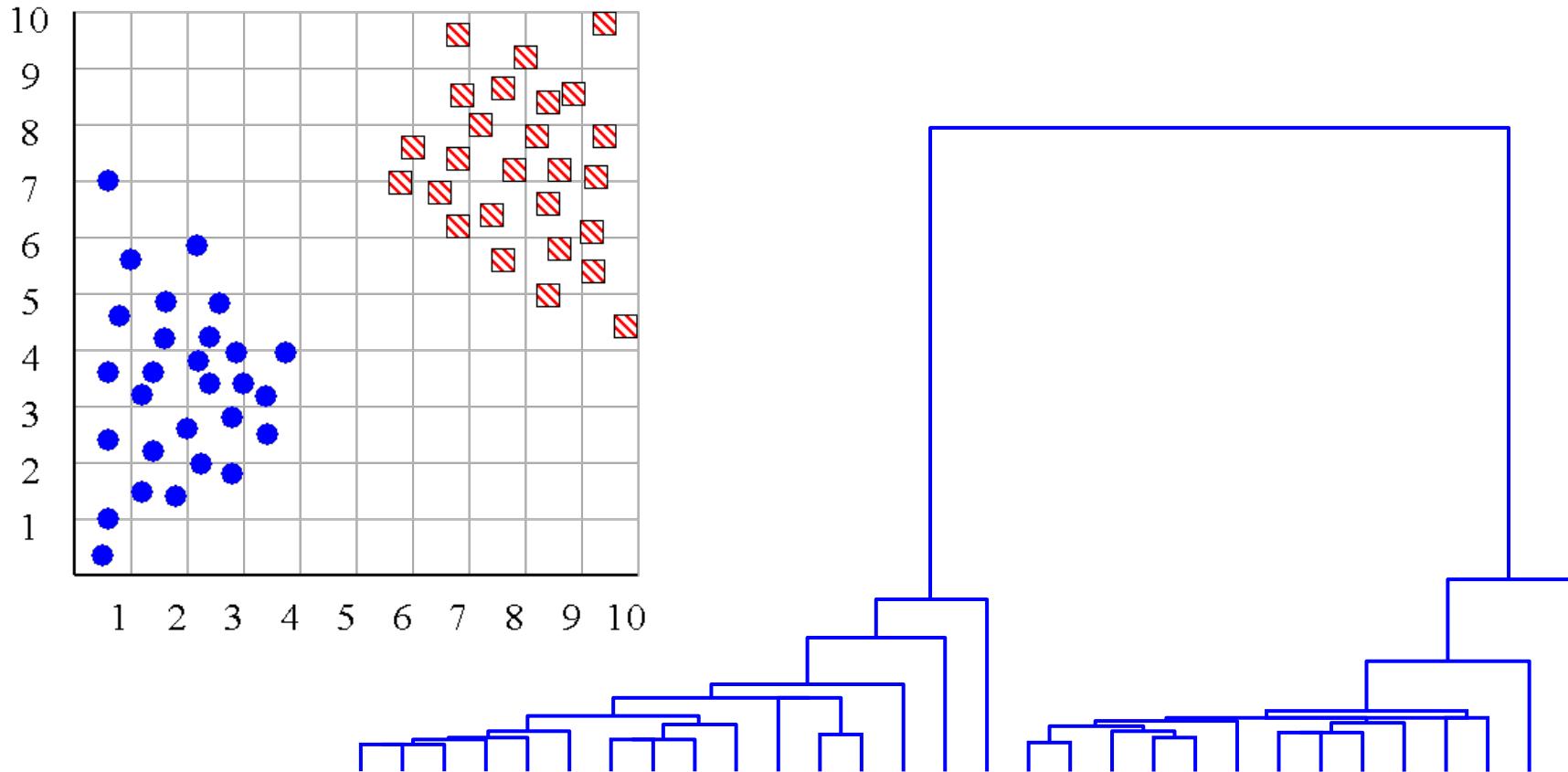
Aber Nigeria und Indien (von Irland wollen wir mal gar nicht erst sprechen) haben nicht viel mit einander am Hut.



# Dendrogramme können uns manchmal auch die richtige Zahl der Cluster sagen



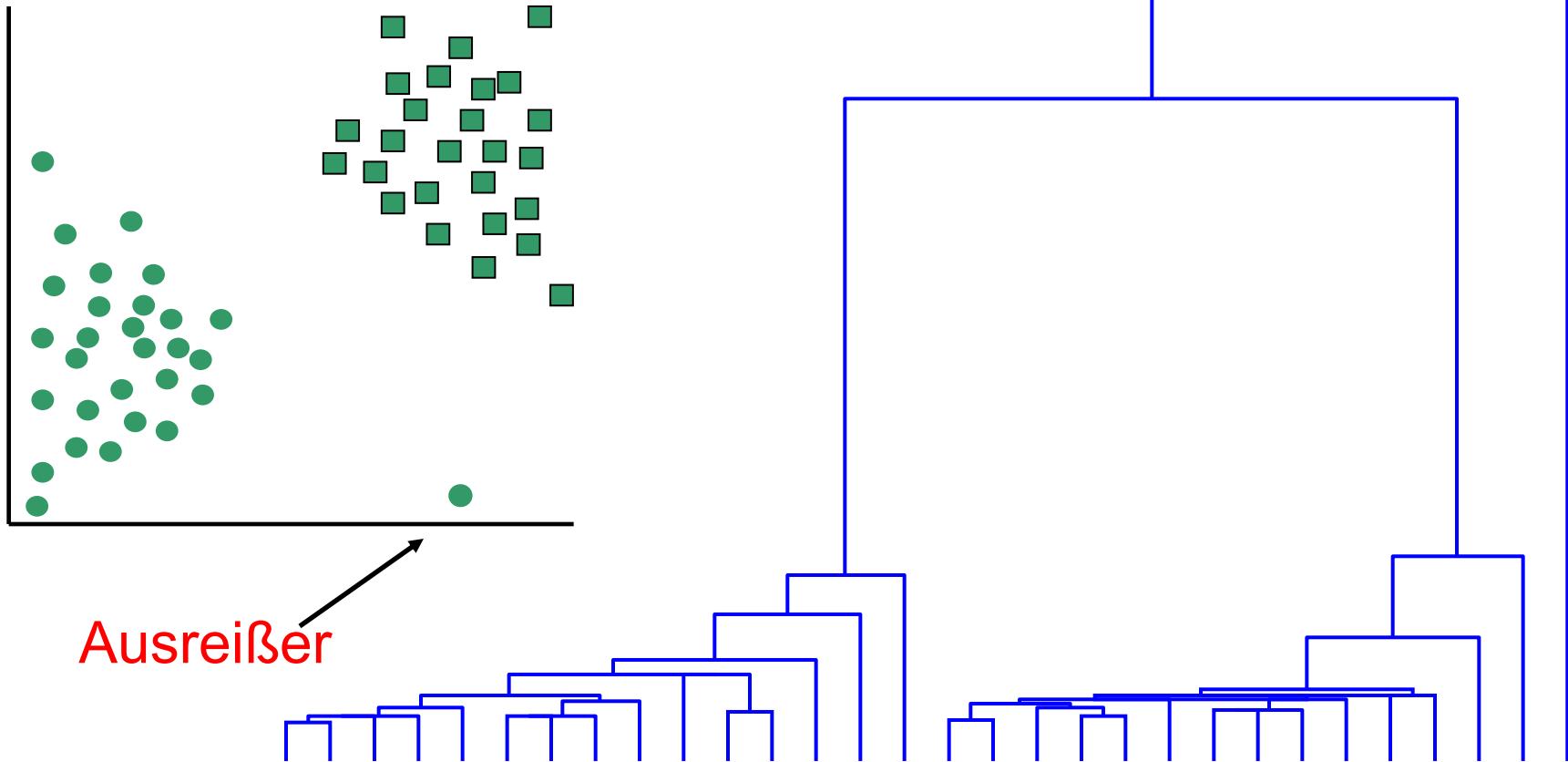
Die zwei stark getrennten Teilbäume legen es nahe, dass es zwei Cluster gibt. Normalerweise ist das aber nicht so klar!



# Sie können auch Ausreißer feststellen



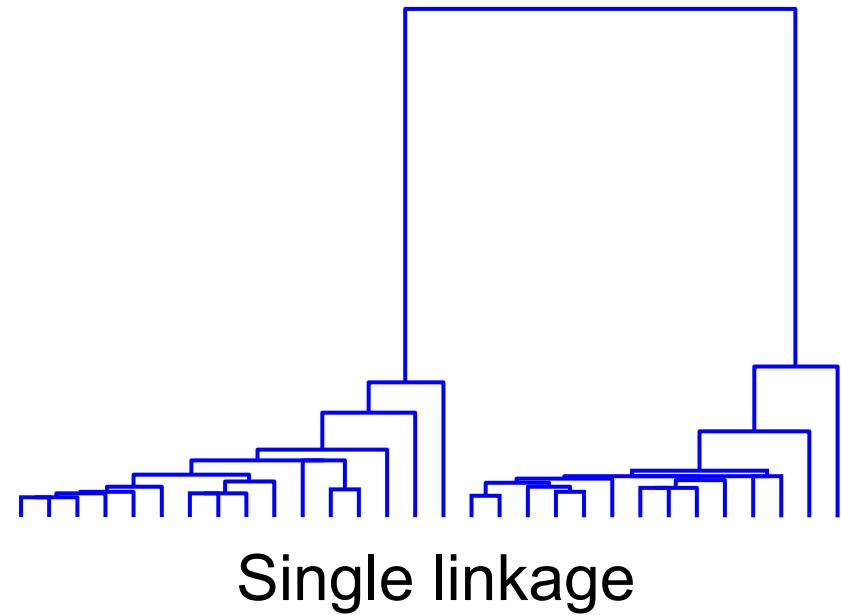
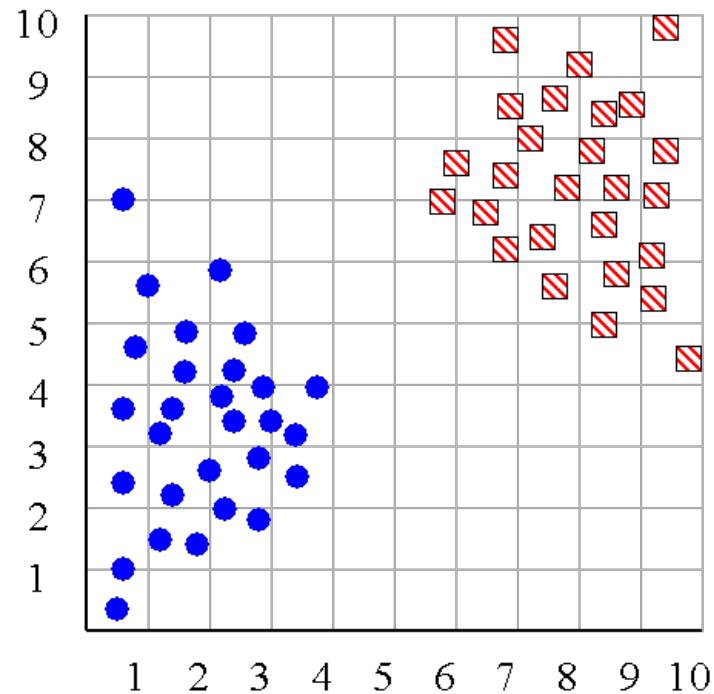
Dieser einzelne Ast legt es nahe, dass es sich um einen Ausreißer handelt.



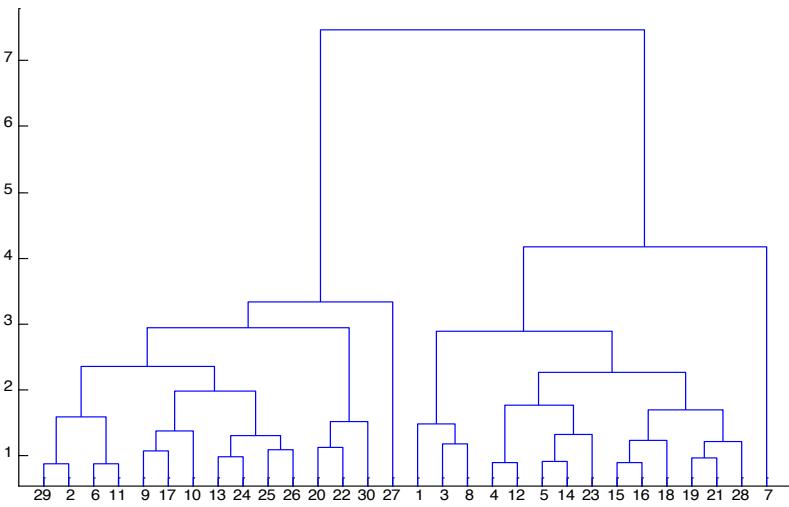
# Clustering ist eine Kunst

Selbst wenn wir eine gute Distanzfunktion habe, ist es nicht selbstverständlich, wie wir eine Distanz zwischen einem Objekt und einem Cluser bzw, zwischen Clustern definieren.

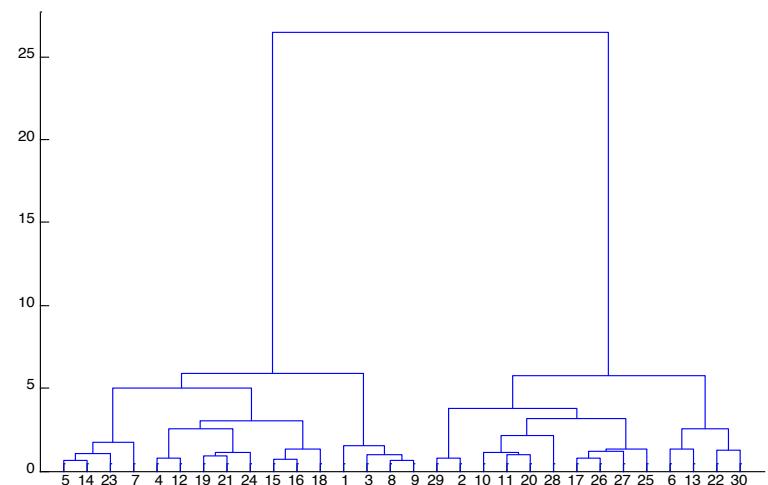
- **Single linkage (nearest neighbor):** Die Distanz ist die Distanz zwischen den beiden nächsten Nachbarn in den beiden unterschiedlichen Clustern.
- **Complete linkage (furthest neighbor):** Die Distanz ist die Distanz zwischen den beiden entferntesten Objekten
- **Group average linkage:** Durchschnitt aller paarweisen Distanzen
- **Wards Linkage:** Man versucht die Varianz zwischen den Clustern zu minimieren.



Single linkage



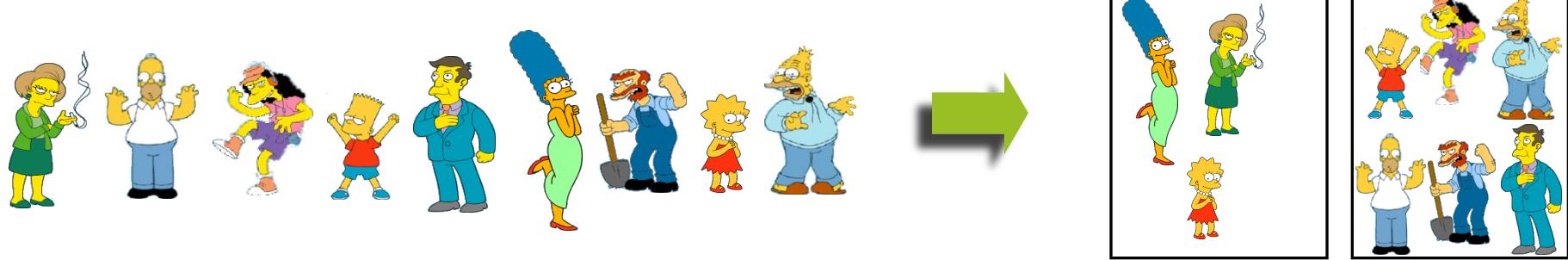
Average linkage



Wards linkage

# Clustering mittels Partitionierungen

Keine Hierarchie. Jedes Objekt gehört zu genau einem Cluster. Die Cluster überlappen nicht



# K-Means Algorithmus

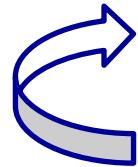


$K =$  Zahl der Clusters (**das gibt ihr an**)

Ein “Mittelwert” pro Cluster

(1) **Initialisiere die Mitterlwert** (z.B. in dem ihr K Datenpunkte zufällig wählt)

Jetzt, wiederholen wir die folgenden zwei Schritte bis zur Konvergenz:



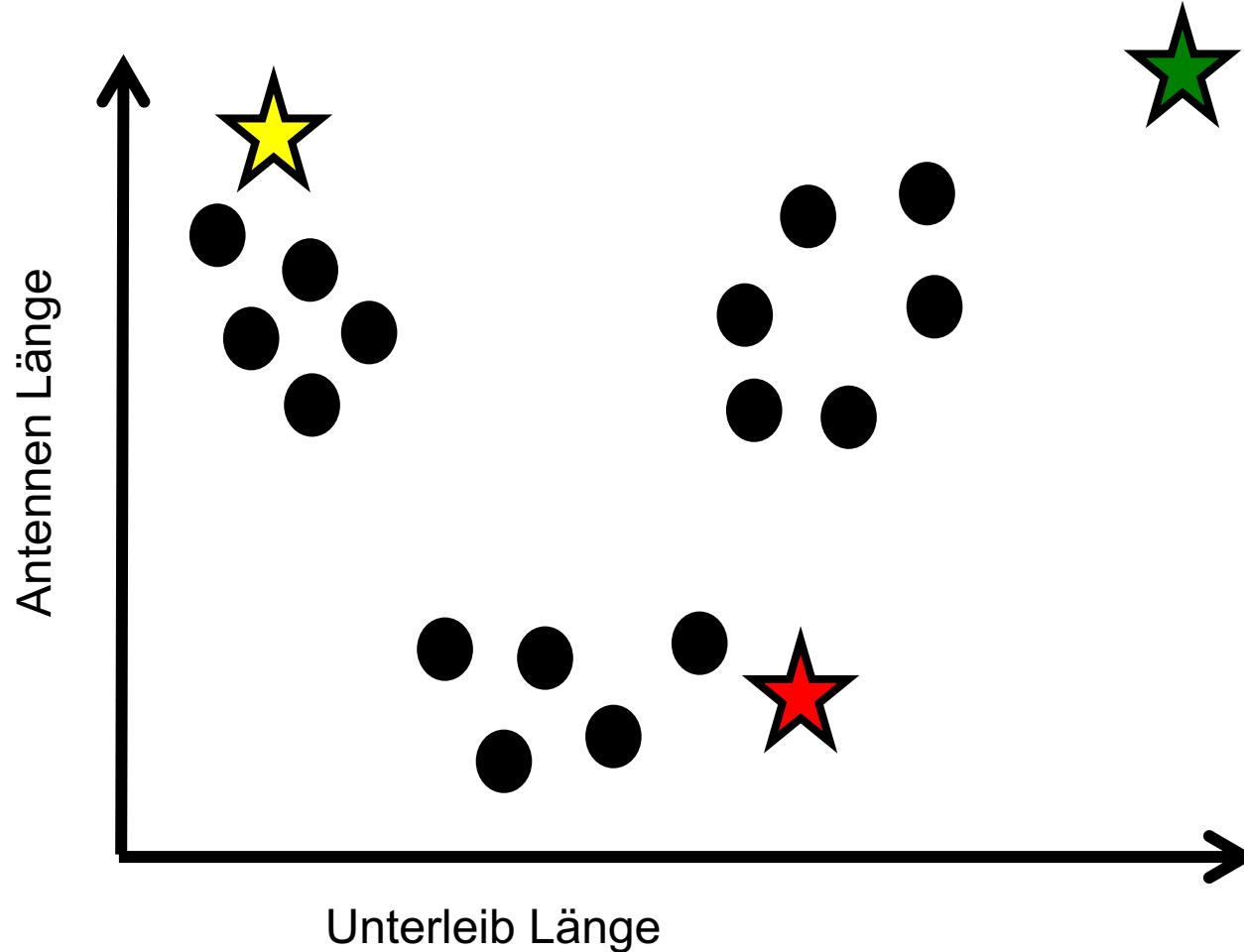
(2) **Ordne jeden Datenpunkt seinem nächsten Mittelwert zu**

(3) **Ersätze jeden Mittelwerte auf den Mittelwert seines Clusters**  
“mean” to center of its cluster

# Schritt 1 : Initialisierung der Mittelwerte



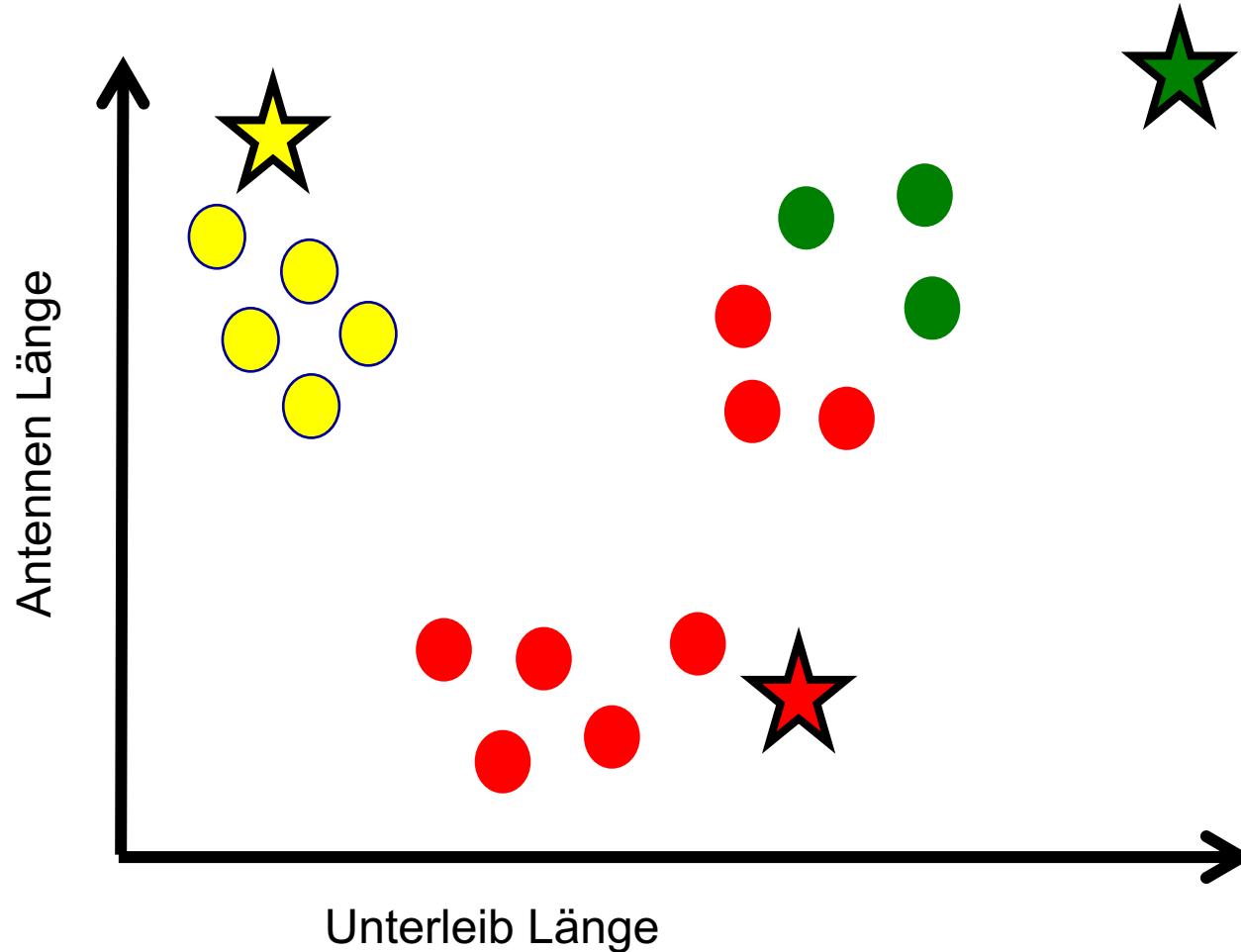
TECHNISCHE  
UNIVERSITÄT  
DARMSTADT



## Schritt 2 : Nächster Mittelwert



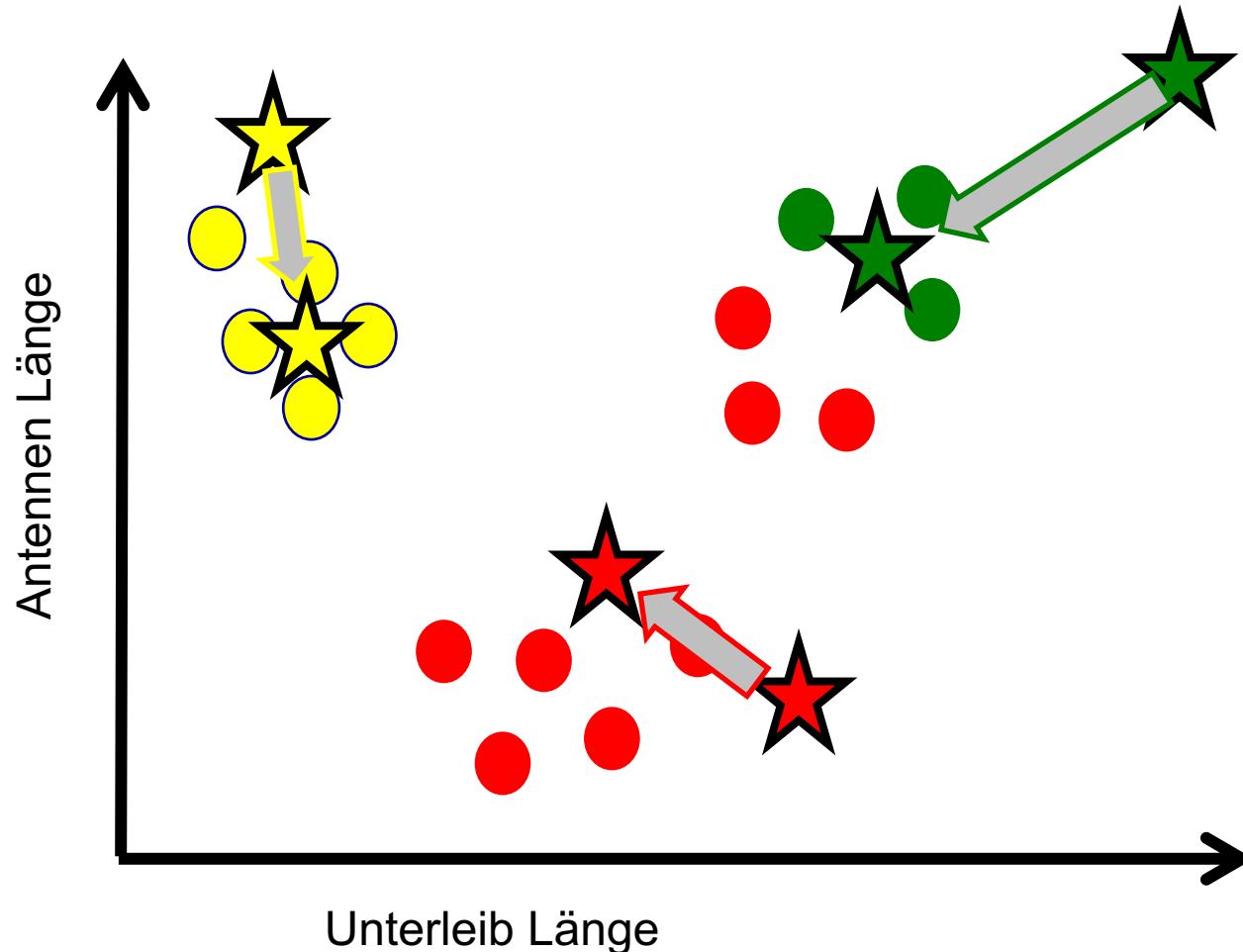
TECHNISCHE  
UNIVERSITÄT  
DARMSTADT



# Schritt 3 : Mittelwert=Mittelwert des Clusters



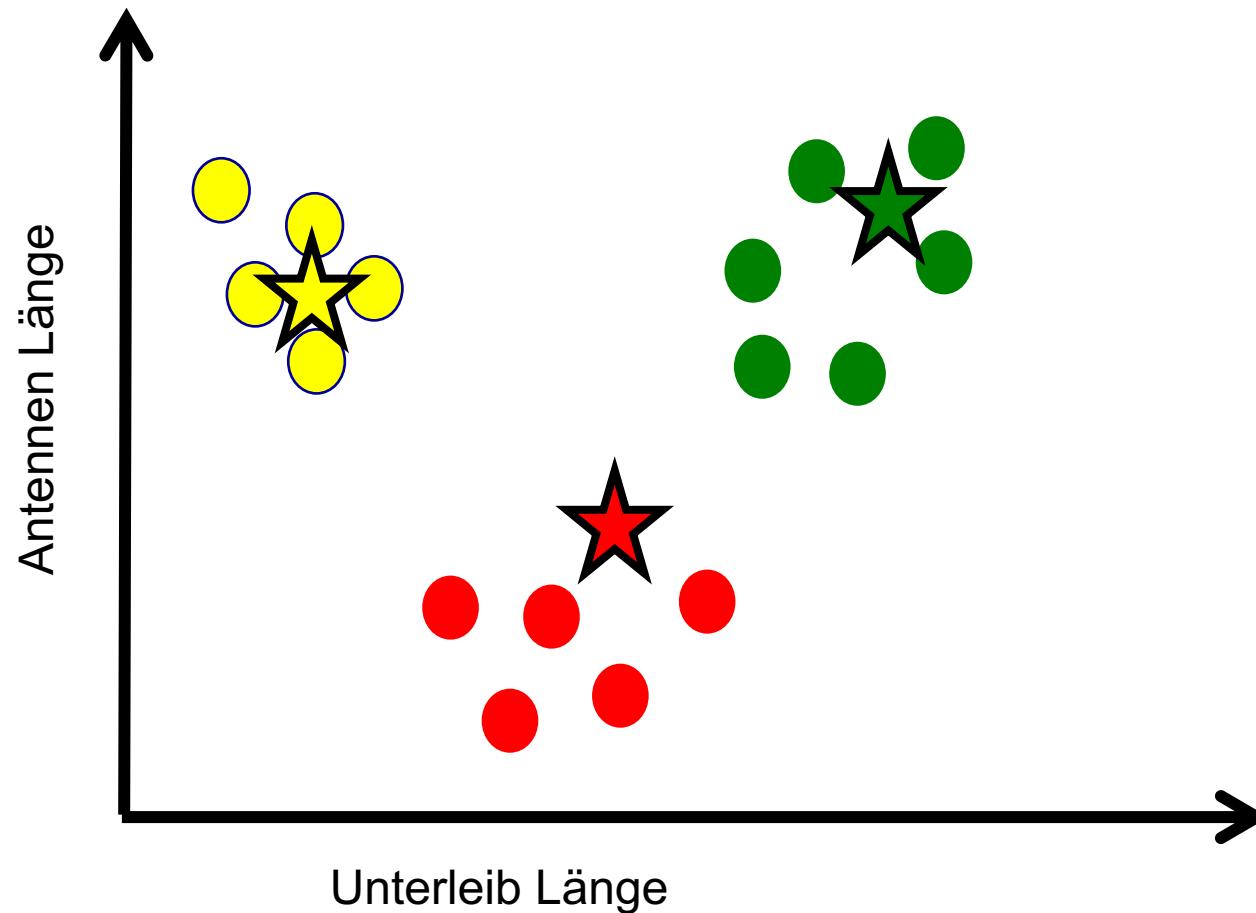
TECHNISCHE  
UNIVERSITÄT  
DARMSTADT



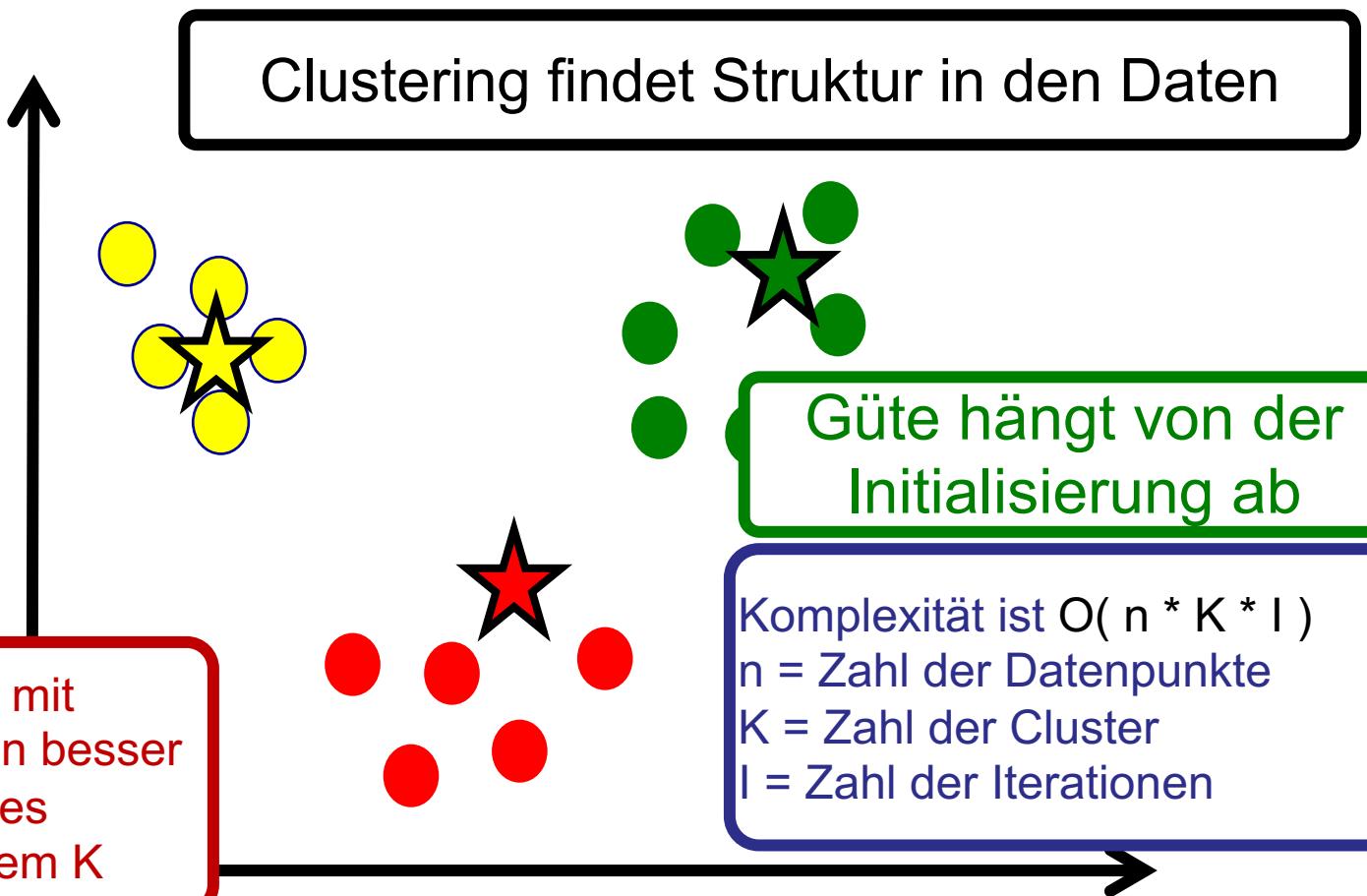
## Schritt 2 : Nächster Mittelwert



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT



# Schritt 3 : Mittelwert=Mittelwert des Clusters



# Nachteile des kMeans-Verfahrens



kMeans bekommt Probleme, wenn die Cluster unterschiedliche

- Größe und
- Dichten

haben und/oder nicht „kugel-förmig“ sind

Weitere Probleme können sich aus

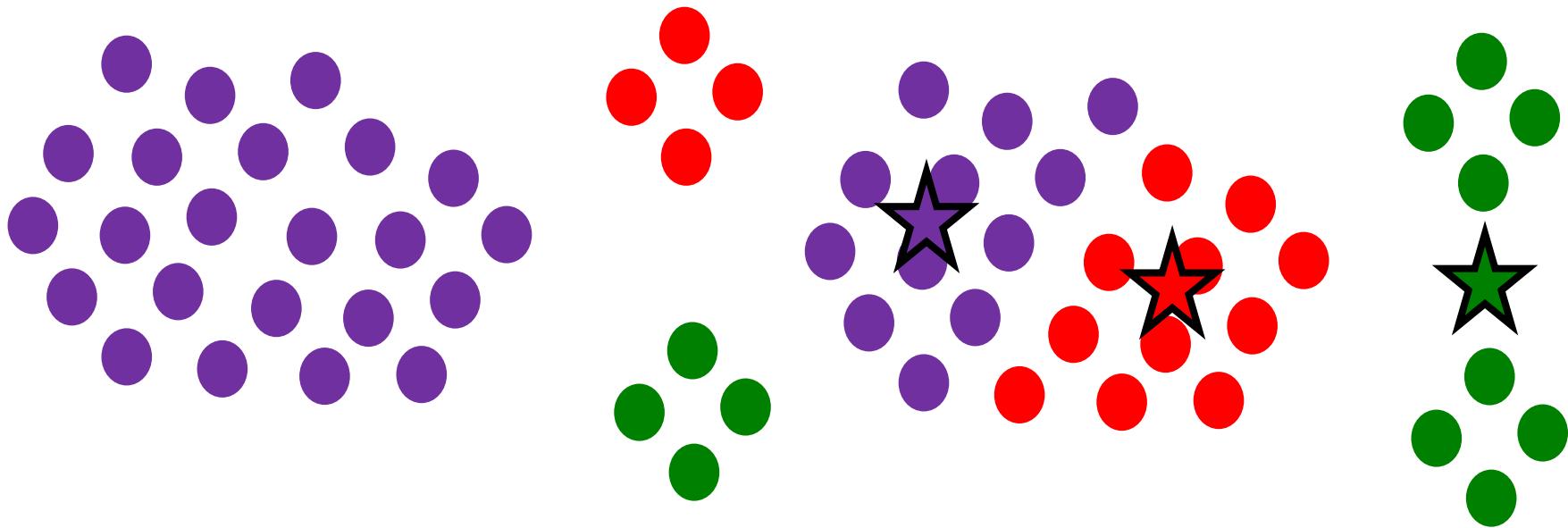
- Ausreißern,
- leere Cluster, und
- hoch-dimensionalen Daten (**Fluch der hohen Dimensionen**)

ergeben

# Nachteile: Unterschiedliche Größen



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT



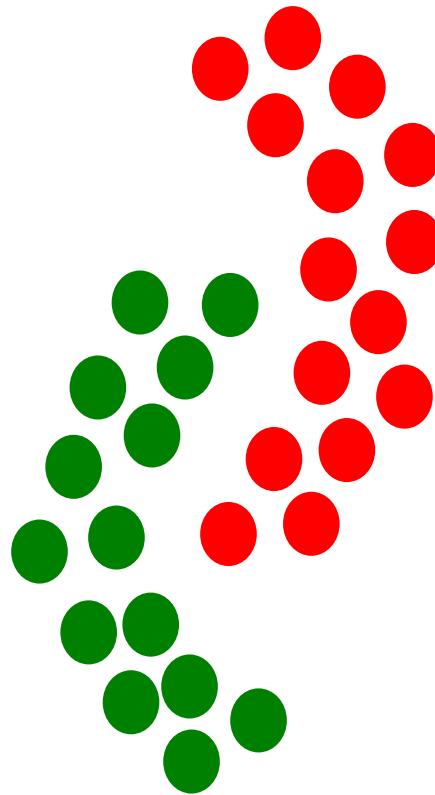
Daten

kMeans (3 Cluster)

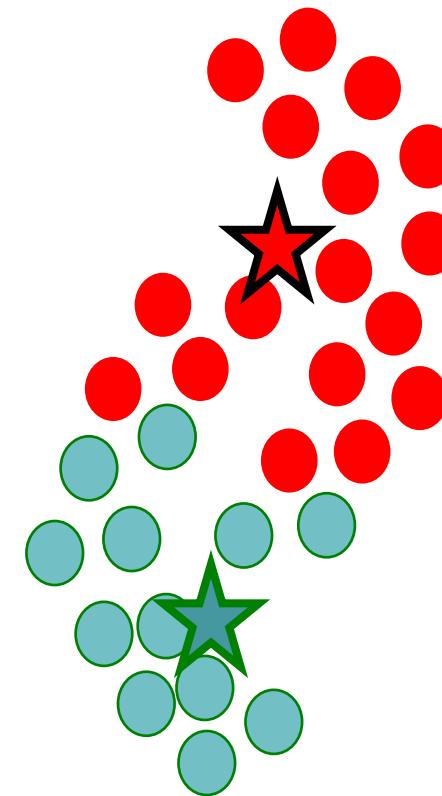
# Nachteile: Nicht-kugelförmig



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT



Daten



kMeans (2 Cluster)

# Strategien zum Umgehen der Nachteile

- Mehrfaches Durchführen + Behalten des besten Ergebnisses
- „Over-Clustering“ + Nachverarbeitung
- Probabilistische oder kernelized Varianten
- Andere Verfahren zur Clusteranalyse wie z.B. Spectral clustering, Random Projections, Clusteranalyse mit Randbedingungen, DBSCAN, bi-clustering, LDA, ...
- ...

# Was wissen wir jetzt?

---

Die Lernaufgabe Clustering kennen Sie

Wir haben zwei Klassen von Methoden gesehen:

- hierarchisches Clustering,
- k-Means.

Die Wahl des Abstandmaßes ist entscheidend für das Clustering.