

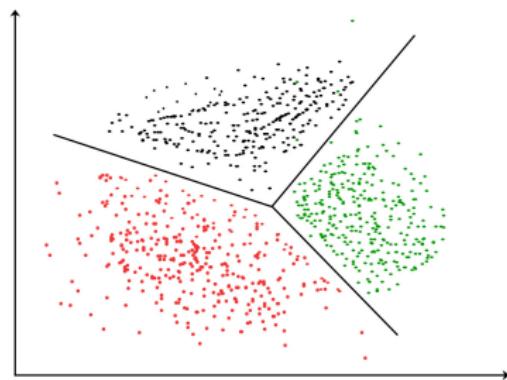
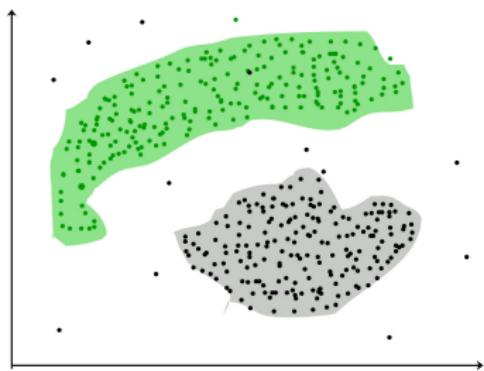
# Artificial Intelligence - Data Science

## Klassifikation und Clustering Algorithmen

Alexandra Posekany

WS 2020

# Mustererkennung



# Mustererkennung

Mustererkennung ist die Quintessenz von Machine Learning \[0.2cm]

- ▶ **Zuordnung** von Beobachtungen zu definierten **Klassen** (Klassifikation)

logistische Regression und Verallgemeinerungen, lineare und quadratische Diskriminanzanalyse (LDA, QDA), k-nächste Nachbarn (k-nearest neighbours KNN)

- ▶ **Erkennen von Strukturen** in Datenmengen (**pattern recognition**)

Clustering, Textmining, ...

neuronale Netzwerke

# Arten des maschinellen Lernens

- ▶ **supervised learning**

auf interpretierbaren Modellen und klar strukturierten Algorithmen basiertes maschinelles Lernen

jeder Teil ist interpretierbar und nachvollziehbar (supervised)

kleinste Quadrate Schätzer wie lineare, nichtlineare Regression; Regression/Decision Trees, Random Forests; Klassifikation durch lineare und quadratische Diskriminanzanalyse, logistische Regression, kNN

- ▶ **unsupervised learning**

maschinelles Lernen, das unbekannte Muster und existierende Klassifizierung sucht ohne auf vorgegebene Strukturen beschränkt zu sein  
k-means Clustering, verteilungsbasiertes Clustering; Anomaly detection; neural networks, deep learning Algorithmen

- ▶ **semi-supervised learning**

wenige gelabelte Trainingsdaten, viele Testdaten ohne bekannte Labels

# Mustererkennung - wichtigste Arten von Klassifikation

- ▶ **k-nearest neighbours** kNN Klassifikation

es wird jedem Punkt das Label gegeben, das die Mehrheit seiner  $k$  nächstgelegenen Nachbarn haben, wobei die Abstände (Distanzen) unterschiedlich gemessen werden können

- ▶ **lineare und quadratische** Diskriminanzanalyse

es wird eine (oder mehrere) lineare Funktion(en) (allgemein Hyperebenen) bzw. quadratische Funktion(en) angepasst, die den Raum so auftrennt (auftrennen), dass auf der einen Seite alle Punkte das eine Label (Klasse) und auf der anderen Seite die Punkte ein anderes Label haben

braucht normalverteilte Daten (analog zu ANOVA)

- ▶ verallgemeinerte lineare etwa **logistische Regression**

ein (lineares) Modell berechnet den erwarteten Wert bei gegebenen unabhängigen erklärenden Variablen; je nachdem, ob eine Schwelle überschritten wird, trennt man zwischen je 2 Labels (Klassen) auf

# Mustererkennung - wichtigste Arten von Klassifikation

- ▶ **Support Vector Machines**

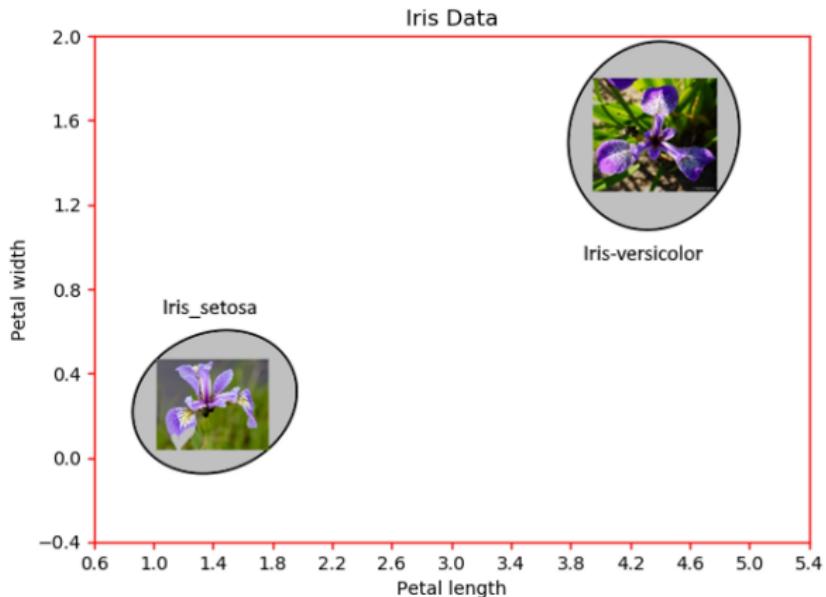
unterteilt Punkte in Klassen durch Projektion auf unterschiedliche Raumrichtungen, sodass um die Klassengrenzen herum ein möglichst breiter Bereich frei von nicht zur Klasse gehörenden Punkten bleibt

- ▶ **Perceptron**

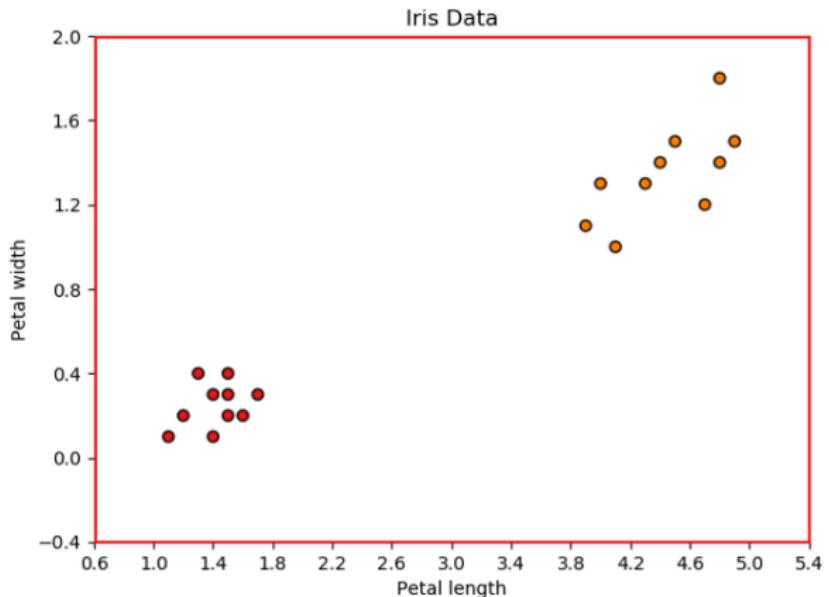
supervised learning Algorithmus verwandt mit neuronalen Netzen  
Basierend auf Gewichten, die von den lernenden Neuronen kommen, werden für alle Punkte Werte ermittelt, die mit einer linearen Schwelle verglichen werden

Je nach Überschreiten oder Unterschreiten der Schwelle wird eine Kategorie zugeordnet

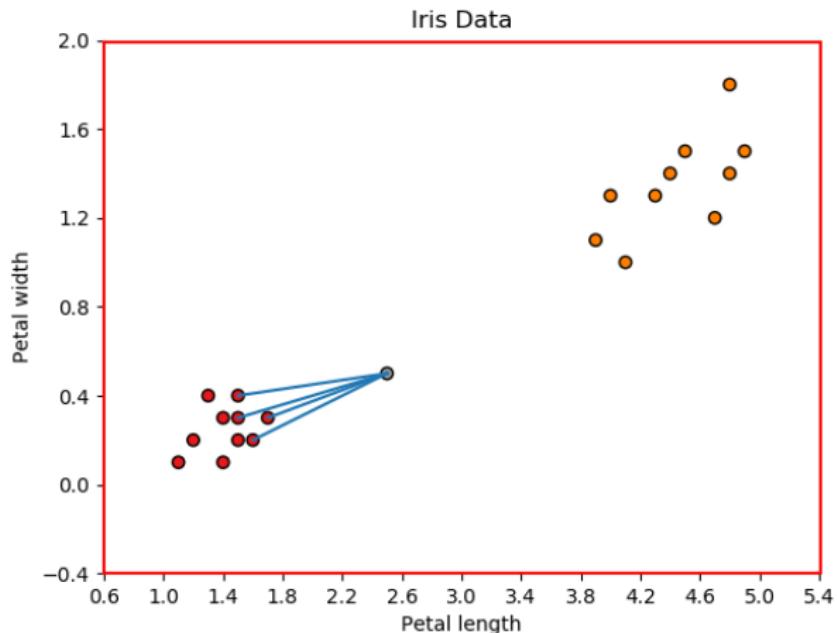
# Konzept des Klassifizierens



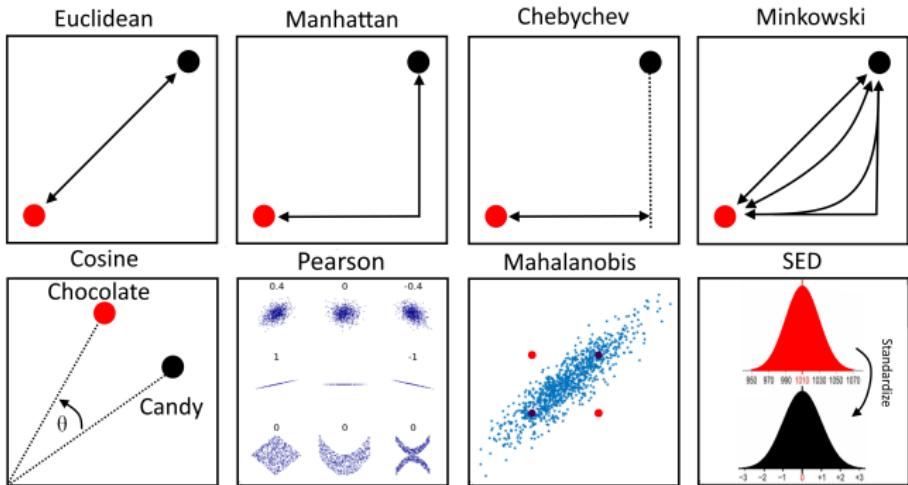
## Konzept des Klassifizierens



# Konzept des Klassifizierens

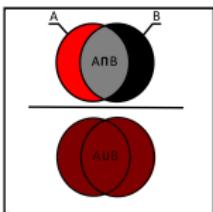


# Distanzmaße

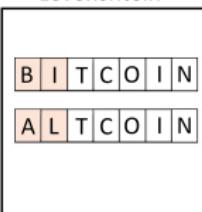


# Distanzmaße

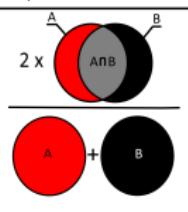
Jaccard



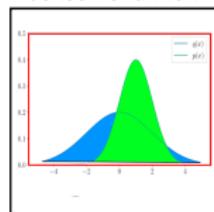
Levenshtein



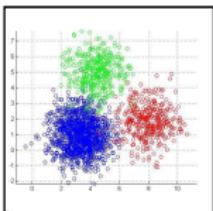
Sørensen–Dice



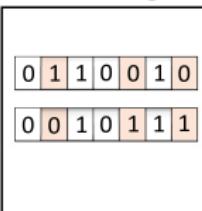
Jensen-Shannon



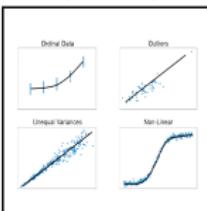
Canberra



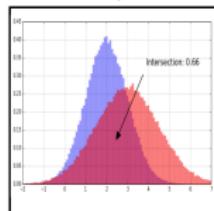
Hamming



Spearman



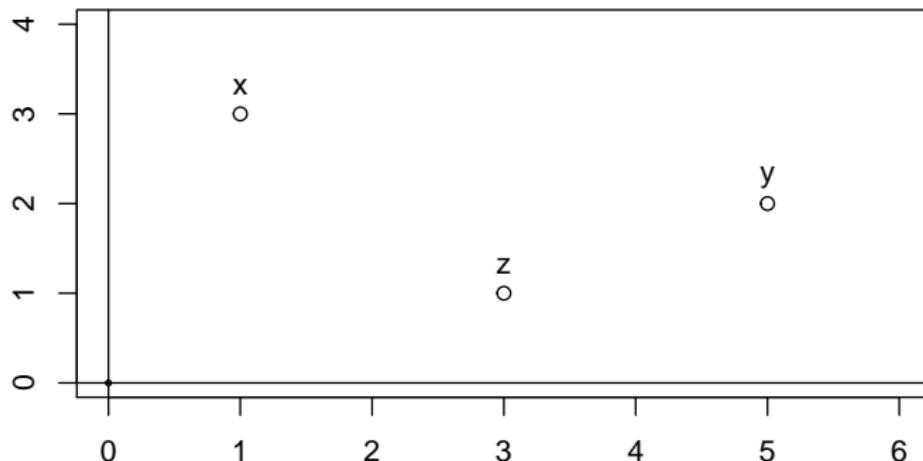
Chi-Square



# Mathematik hinter Distanzen

Eine **Metrik** (Distanzmaß) ist eine Abbildung  $d: X \times X \rightarrow \mathbb{R}$ , die folgende Eigenschaften erfüllt:

1. **Positiv Definitheit:**  $d(x, y) \geq 0$  und  $d(x, y) = 0 \Leftrightarrow x = y$
2. **Symmetrie:**  $d(x, y) = d(y, x)$
3. **Dreiecksungleichung**  $d(x, y) \leq d(x, z) + d(z, y)$



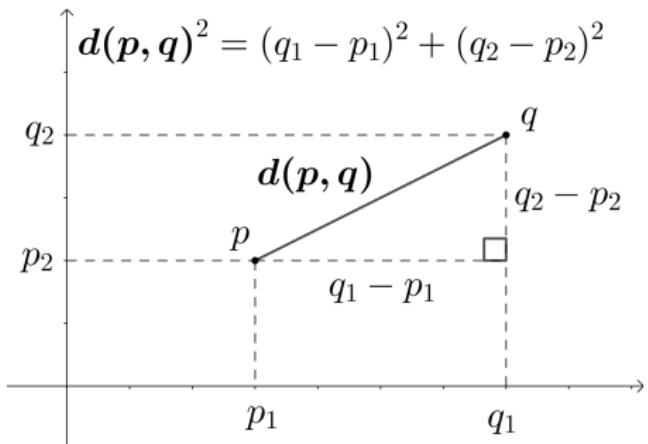
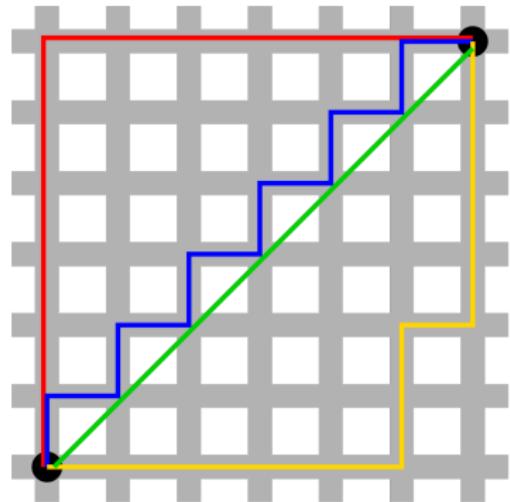
# Distanzmaße

Für zwei Punkte  $a = (a_1 | a_2 | \dots | a_d)$  und  $b = (b_1 | b_2 | \dots | b_d)$  im  $d$ -dimensionalen Raum  $\mathbb{R}^d$  bzw. für Zeichenketten  $x$  und  $y$  der Länge  $d$  kann der Abstand zwischen den beiden Punkten auf unterschiedliche Arten gemessen werden:

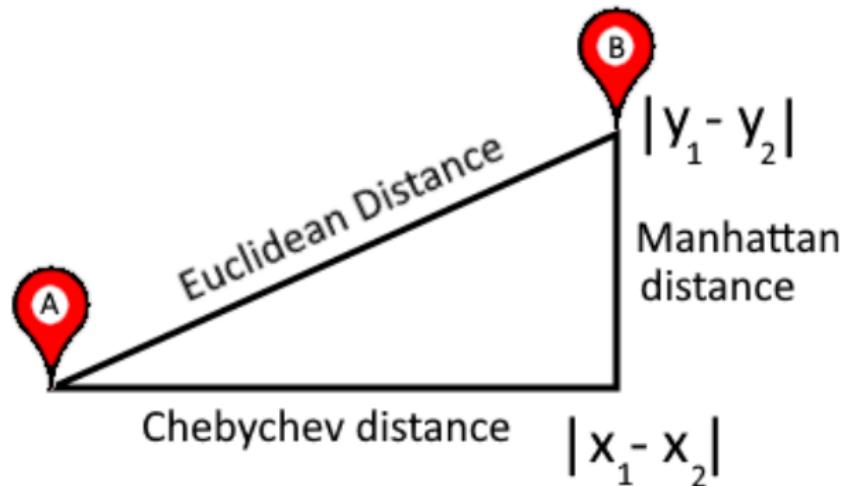
Euklidische Distanz	$\ a - b\ _2 = \sqrt{\sum_{i=1}^d (a_i - b_i)^2}$
Manhattan Distanz City-Block-Metrik	$\ a - b\ _1 = \sum_{i=1}^d  a_i - b_i $
Maximaldistanz Chebychevdistanz	$\ a - b\ _\infty = \max_{i=1, \dots, d}  a_i - b_i $
Mahalanobisdistanz	$\ a - b\ _{Mahalanobis} = \sqrt{(a - b)^\top COV^{-1}(a - b)}$
Hamming-Distanz	$\ x - y\ _{Hamming} =  \{j \in \{1, \dots, d\} \mid x_j \neq y_j\} $

# Distanzmaße

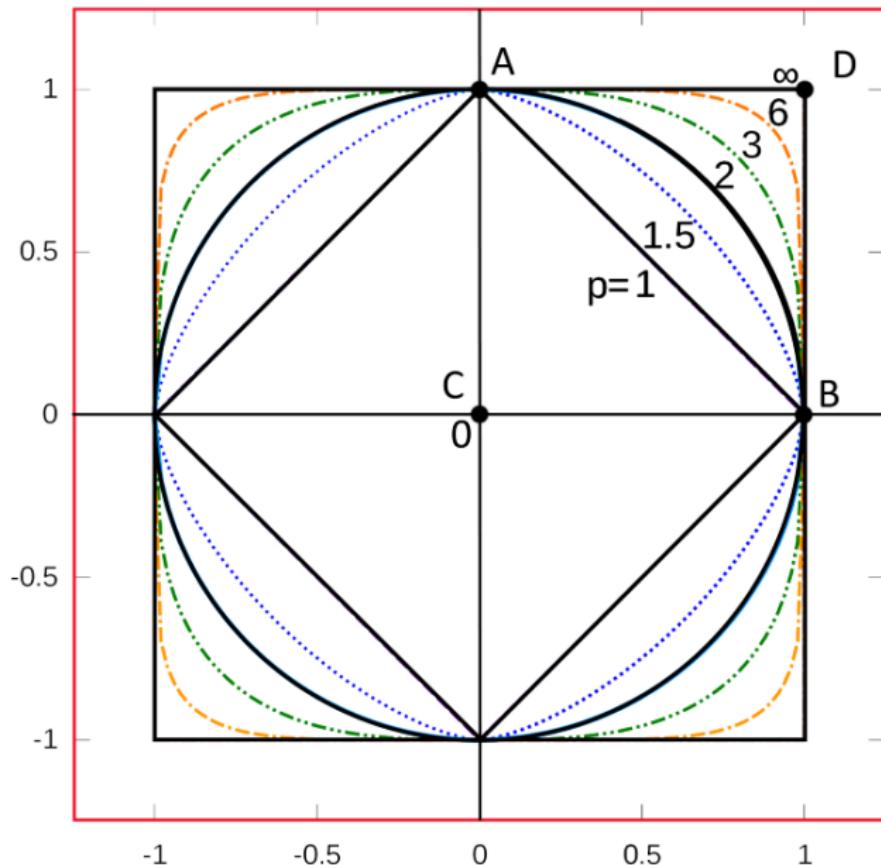
Euklidische Distanz vs. Manhattan Distanz vs. Maximaldistanz



## Distanzmaße



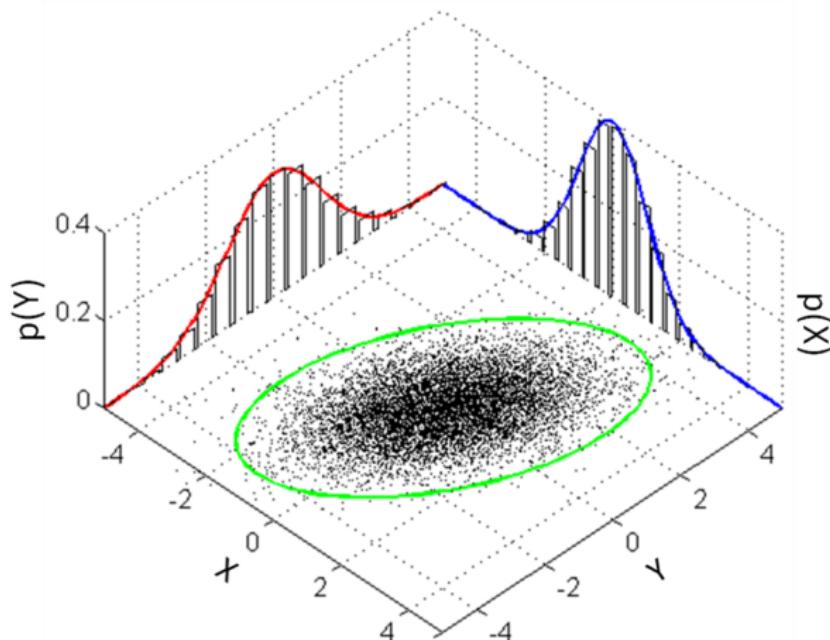
# Distanzmaße



# Distanzmaße

Was bedeutet die Mahalanobisdistanz?

Sie skaliert um den Mittelwert und die Kovarianz auf unkorrelierte Normalverteilungen in den Rändern



# Korrelation - Mahalanobisdistanz

Hauptkomponentenanalyse basiert auf der Eigenvektorzerlegung der Datenmatrix, sodass Korrelationen verschwinden \[0.1cm]

## Kovarianz und Korrelation

Der Pearson Korrelationskoeffizient

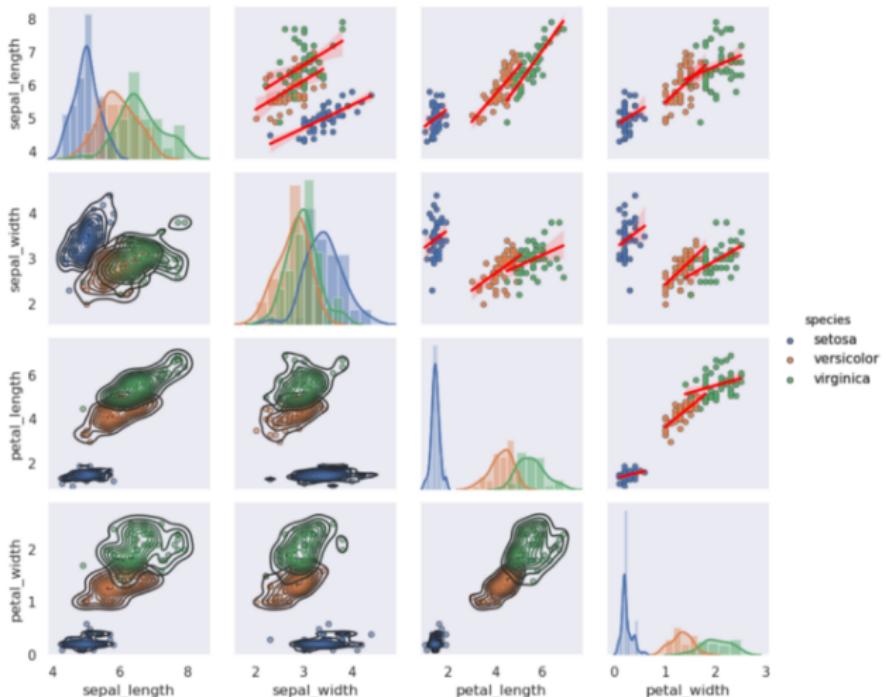
$$r = r(X, Y) = \frac{\widehat{\text{cov}}(X, Y)}{s(X) \cdot s(Y)},$$

misst den linearen Zusammenhang zwischen 2 metrischen Variablen  $X$  und  $Y$ , der sich auf Basis der Kovarianz

$$\widehat{\text{cov}}(X, Y) = \frac{1}{n-1} \sum_{i=1}^n (y_i - \bar{y})(x_i - \bar{x}).$$

berechnet.

# Korrelation und Klassifikation



# Distanzmaße

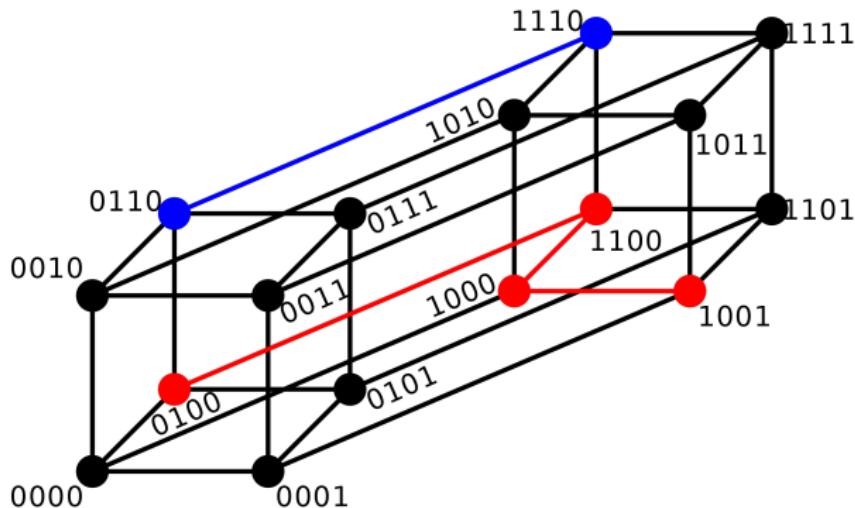
Was bedeutet die Hamming-Distanz?

Sie zählt die Anzahl der unterschiedlichen Zeichen

11~~0~~01 und 11~~1~~00 haben Hamming-Distanz 2

N~~O~~ST und N~~E~~ST haben Hamming-Distanz 1

In einem Graphen zählt sie die Anzahl der Kanten um von einem Knoten zum anderen zu kommen



# Distanzmaße

Die Levenshtein-Distanz (Editierdistanz) zählt die Mindestanzahl von Operationen

1. Einfügen
2. Ersetzen
3. Löschen

Machen wir "Klara" zu "Caro":

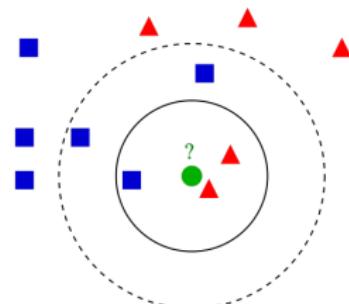
1. Löschen von "l": "Kara"
2. Ersetzen von "K": "Cara"
3. Ersetzen von "o": "Caro"

"Klara" und "Caro" haben Levenshtein-Distanz 3, da 3 Transformationsschritte mindestens notwendig sind.

# Klassifikation - k nearest neighbours

## Ablauf des Algorithmus

1. Initialisiere die Labels auf den Trainingsdaten
2. Klassifiziere, indem
  - 2.1 die Distanzen aller Testdatenpunkte zu allen Trainingsdatenpunkten ermittelt werden
  - 2.2 für alle Punkte aus dem Testdatensatz ermittle die  $k$  am nächsten gelegenen Punkte der Trainingsdaten und ihre Labels
  - 2.3 für alle Punkte aus dem Testdatensatz weise das am häufigsten bei seinen Nachbarn vorkommende Label dem Punkt zu
3. Validiere, indem die wahren Labels und die vom Algorithmus zugewiesenen Labels verglichen werden



## Klassifikation - k nearest neighbours

Wie wählt man die Anzahl der Klassen?

**k muss so gewählt werden, dass es immer eine eindeutige Mehrheit geben kann**

für 2 Klassen, muss daher k immer ungerade sein

bei n Klassen, ist k idealerweise (ein Vielfaches von  $n + 1$ ), wodurch jedenfalls eine Klasse mehr Zugehörigkeiten als die anderen haben muss

n Klassen	k Nachbarn
2	3, 5, 7, ...
3	7, 13, 19 ...
4	5, 9, 13, ...
:	:
n	$(c \cdot n + 1)$ , $c \in \mathbb{N}$ und ungerade

# Klassifikation - k nearest neighbours

Optionen für die Klassifikation:

- ▶ alle k Nachbarn erhalten **dasselbe Gewicht**

Vorteil: zufällige Schwankungen bei der Lage der Punkte haben weniger Einfluss

Nachteil: "wichtige" und "weniger wichtige" Punkte werden nicht unterschieden

- ▶ die k Nachbarn erhalten **Gewichte, abhängig vom Abstand zum Punkt**

Vorteil: nahe gelegene Punkte haben mehr Einfluss, da sie besser dazupassen;

weiter entfernt gelegene Punkte haben geringeren Einfluss, da sie weniger gut passen  
Nachteil: zufällige Schwankungen bei der Lage der Punkte haben einen Einfluss auf ihr Gewicht

# Qualitätsmaße für Klassifikation

Wir starten von der Konfusionsmatrix (Confusion matrix) bei der die wahren Labels den prädiktierten Labels gegenübergestellt werden.

- ▶ **Missklassifikationsrate und Accuracy**

Anteil der falsch oder richtig klassifizierten Kategorien der Test- oder Validierungsdaten über alle Klassen hinweg

- ▶ **Sensitivität und Spezifität**

True Positive Rate und True Negative Rate jeder einzelnen Klassen gegen alle anderen Klassen gepoolt  
darstellbar als separate ROC Kurve je Klasse

# Fluch der Dimensionalität - Curse of dimensionality

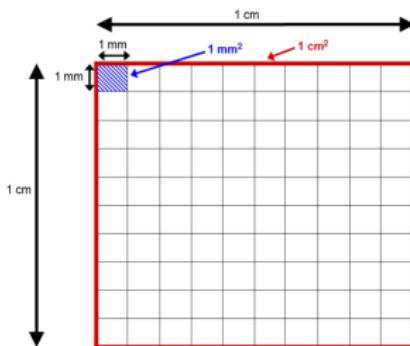
je mehr Dimensionen (Raumrichtungen = Variablen) berücksichtigt werden, um so Beobachtungspunkt sind notwendig, um denselben Informationsgehalt zu bekommen

Bsp:

$$1m = 100cm$$

$$1m^2 = 100^2 cm^2 = 10000 cm^2$$

$$1m^3 = 100^3 cm^3 = 1000000 cm^3$$



Um dieselbe Information wie durch 100 Punkte in 1 Dimension zu erhalten, braucht man 10000 Punkte in 2 Dimensionen und 1000000 Punkte in 3 Dimensionen.

Dieser **Informationsverlust durch höhere Dimensionen** heißt  
**“Fluch der Dimensionalität” (Curse of Dimensionality)**.

# Fluch der Dimensionalität - Curse of dimensionality

Was tun, wenn wir verflucht sind?

Es gibt Methoden zur Dimensionsreduktion, die die wichtigste Information auf wenige Dimensionen reduzieren

Dimensionsreduktion

- ▶ reduziert die Laufzeit und den Speicheraufwand,
- ▶ entfernt Multikollinearität und verbessert die Interpretierbarkeit von erklärenden Variablen in (Regressions-) Modellen,
- ▶ ermöglicht, dass hochdimensionale Daten wieder zwei- oder dreidimensional visualisiert werden können,
- ▶ verhindert den Fluch der Dimensionalität.

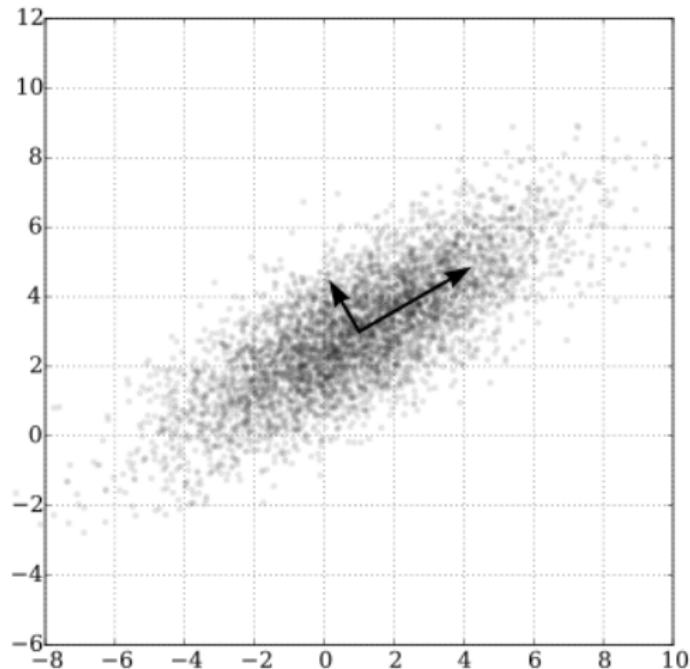
# Dimensionsreduktion

Methoden zur Dimensionsreduktion, die die wichtigste Information auf wenige Dimensionen reduzieren

- ▶ **Hauptkomponentenanalyse (Principal Component Analysis, PCA)** und Varianten davon  
bekannte Variante: **Faktorenanalyse (Factor Analysis, FA)**
- ▶ lineare und verallgemeinerte **Diskriminanzanalyse (LDA, GDA)**
- ▶ **Non-negative Matrix Factorisation (NMF)**
- ▶ **T-distributed Stochastic Neighbor Embedding (t-SNE)**
- ▶ **Uniform Manifold Approximation and Projection (UMAP)**

# Hauptkomponentenanalyse - Principal Component Analysis

Idee: Erfassen der Raumrichtung(en) mit der meisten Variation = Information



# Mathematik hinter Hauptkomponentenanalyse

Hauptkomponentenanalyse basiert auf der Eigenvektorzerlegung der Datenmatrix, sodass Korrelationen verschwinden \[0.1cm]

## Eigenvektoren und Eigenwerte

Ein Vektor  $\vec{v}$  heißt **Eigenvektor** einer quadratischen Matrix  $A \in \mathbb{R}^{n \times n}$  und ein Wert  $\lambda$  **Eigenwert**, wenn sie die Gleichung

$$\underbrace{A}_{\text{Matrix}} \cdot \underbrace{\vec{v}}_{\text{Eigenvektor}} = \underbrace{\lambda}_{\text{Eigenwert}} \cdot \underbrace{\vec{v}}_{\text{Eigenvektor}}$$

erfüllen.

## Spektralsatz

Für eine symmetrische Matrix gilt, dass sie eine Orthonormalbasis aus Eigenvektoren besitzt.

Die Eigenvektoren skaliert auf Länge 1 lassen sich daher in einer Matrix, deren Transponierte ihre eigene Inverse ist (orthonormal), der **Spektralmatrix**  $S$  anschreiben:

$$\underbrace{S^T}_{=S^{-1}} \cdot A \cdot S = \begin{pmatrix} \lambda_1 & 0 & \dots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & \lambda_n \end{pmatrix}$$

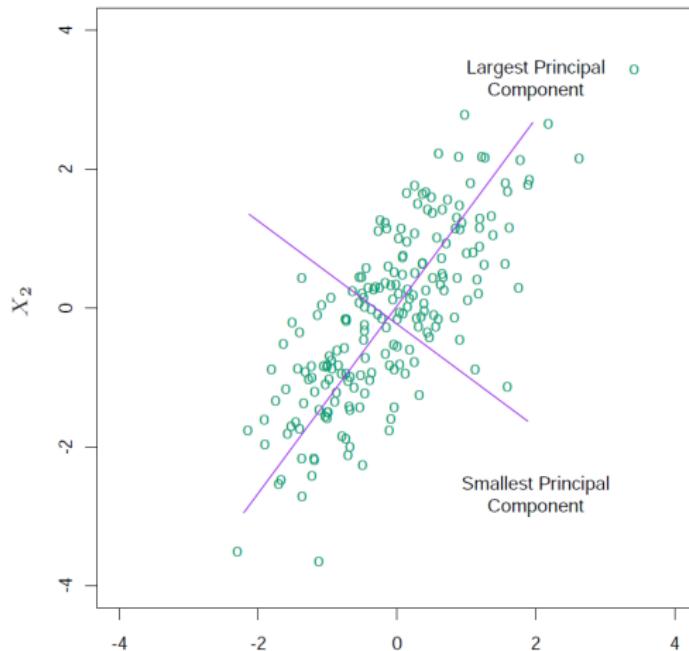
Wie der Name Orthonormalbasis erahnen lässt, stehen die Vektoren im rechten Winkel aufeinander und spannen den gesamten Vektorraum auf.

Die **Skalierungslängen** der Vektoren bei einer positiv (semi)-definiten Matrix wie der Kovarianzmatrix sind die **Eigenwerte**.

# Hauptkomponentenanalyse - Principal Component Analysis

{Die Raumrichtungen sind die beiden Orthonormalvektoren = Eigenvektoren = Hauptkomponenten. }

{Anhand der Eigenwerte, weiß man, welche Raumrichtungen wieviel Information tragen: Je kleiner der Eigenwerte, desto weniger Information in Richtung des Eigenvektors = Hauptkomponente. }

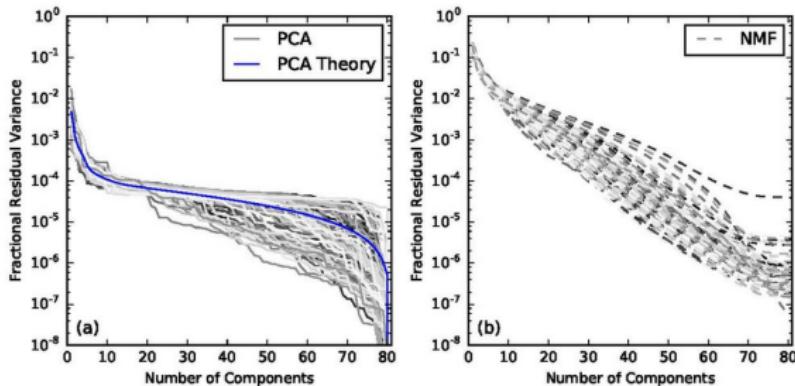


# Non-negative Matrix Factorisation

Ähnlich wie die PCA wird eine Raumbasis eines kleineren Teilraums gewählt, aber so, dass alle Matrizeeinträge nichtnegativ sind.

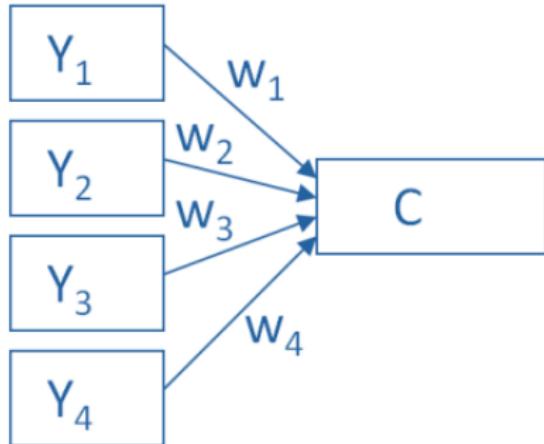
Dadurch ist die Raumbasis nicht orthogonal, hat also keine so schöne Interpretierbarkeit als Ellipsoid wie die PCA Orthonormalbasis.

Dafür ist NMF effizienter als PCA bei mittleren Anzahlen von Komponenten.



# Hauptkomponentenanalyse vs. Faktorenanalyse

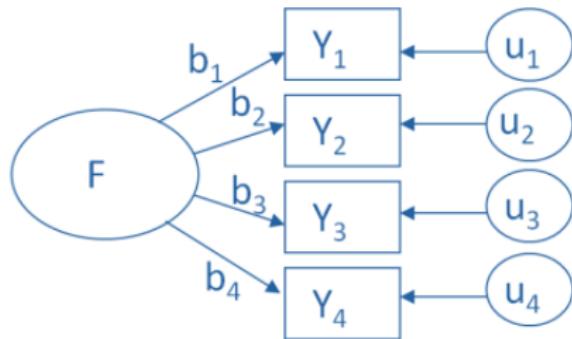
## PCA



Eine Komponente  $C$  setzt sich durch die folgende Gleichung aus 4 Variablen (Dimensionen) mit Gewichten  $w_i$  zusammen:

$$C = w_1(Y_1) + w_2(Y_2) + w_3(Y_3) + w_4(Y_4)$$

## FA



Ein Faktor entsteht durch das Gleichungssystem von linearen Regressionsgleichungen, die den Faktor als Regressor haben:

$$Y_1 = b_1 \cdot F + u_1$$

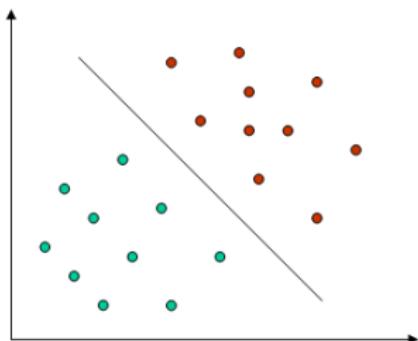
$$Y_2 = b_2 \cdot F + u_2$$

$$Y_3 = b_3 \cdot F + u_3$$

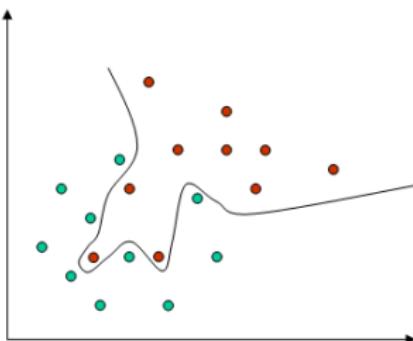
$$Y_4 = b_4 \cdot F + u_4$$

# Klassifikation - Diskriminanzfunktion

Die Verfahren Diskriminanzanalyse und Support Vector Machine benutzen **lineare oder nichtlineare Diskriminanzfunktionen**, die die **Grenze zwischen den Klassen** beschreiben



linear trennbar



nicht linear trennbar

# Klassifikation - lineare Diskriminanzanalyse

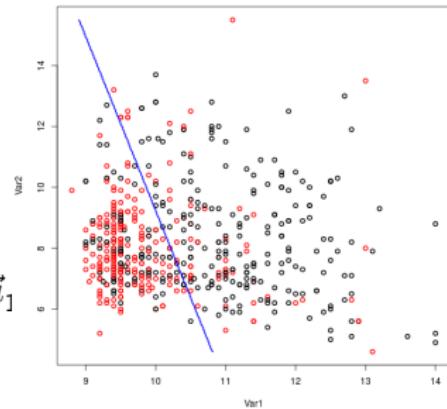
## Ablauf des Algorithmus

1. Überprüfe die Voraussetzung der **Normalverteilung der unabhängigen Variablen** innerhalb der einzelnen Klassen innerhalb der unterschiedlichen Klassen müssen dieselben Streuungen (Kovarianzen) herrschen
2. Klassifizierte, indem

### 2.1 Berechne den Klassifikationsterm

$$\Sigma^{-1}(\vec{\mu}_1 - \vec{\mu}_0) \cdot \vec{x} > \frac{1}{2} (\vec{\mu}_0^T \Sigma^{-1} \vec{\mu}_0 + \vec{\mu}_1^T \Sigma^{-1} \vec{\mu}_1 - \vec{\mu}_0^T \Sigma^{-1} \vec{\mu}_1 - \vec{\mu}_1^T \Sigma^{-1} \vec{\mu}_0)$$

und ordne je nach Über- oder Unterschreiten der Schwelle das eine oder andere Label zu



3. Validiere, indem die wahren Labels und die vom Algorithmus zugewiesenen Labels verglichen werden

# Klassifikation - quadratische Diskriminanzanalyse

## Ablauf des Algorithmus

1. Überprüfe die Voraussetzung der **Normalverteilung der unabhängigen Variablen** innerhalb der einzelnen Klassen innerhalb der unterschiedlichen Klassen dürfen verschiedene Streuungen (Kovarianzen) herrschen
2. Klassifizierte, indem

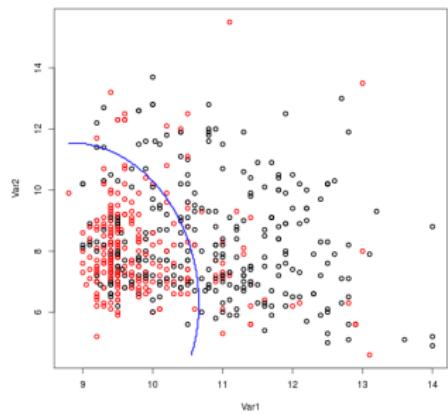
### 2.1 berechne den Klassifikationsterm

$$(\vec{x} - \vec{\mu}_0)^T \Sigma_0^{-1} (\vec{x} - \vec{\mu}_0) + \ln |\Sigma_0| -$$

$$(\vec{x} - \vec{\mu}_1)^T \Sigma_1^{-1} (\vec{x} - \vec{\mu}_1) - \ln |\Sigma_1| > T$$

und ordne je nach Über- oder Unterschreiten der Schwelle das eine oder andere Label zu

3. Validiere, indem die wahren Labels und die vom Algorithmus zugewiesenen Labels verglichen werden



# Mathematik hinter linearer und quadratischer Diskriminanzanalyse

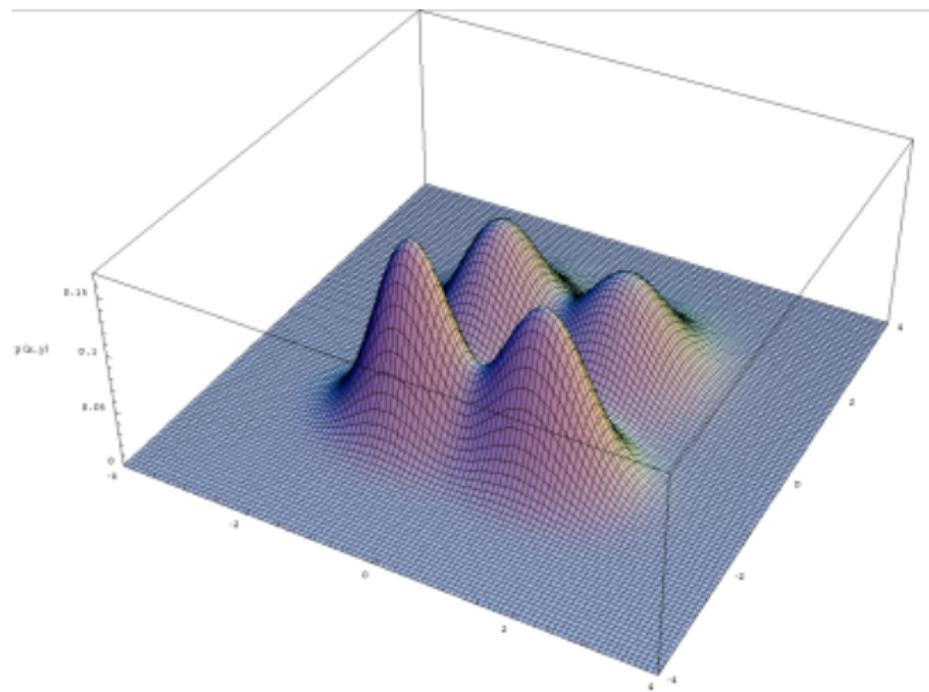
Die Wahrscheinlichkeit, dass eine Beobachtung  $x$  als die Klasse  $k$  klassifiziert wird, lautet

$$\mathbb{P}(G = k|x) = \frac{h_k(x) \cdot \pi_k}{\sum_{l=1}^K h_l(x) \cdot \pi_l},$$

wobei  $h_k$  oft als multivariate Normalverteilung wird:

$$h_k(\mathbf{x}) = \frac{\exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu})\right)}{\sqrt{(2\pi)^k |\boldsymbol{\Sigma}|}}$$

# Mathematik hinter linearer und quadratischer Diskriminanzanalyse



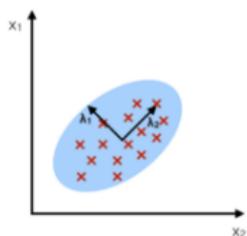
# Mathematik hinter linearer und quadratischer Diskriminanzanalyse

$$\begin{aligned} \log \frac{\mathbb{P}(G = k_j|x)}{\mathbb{P}(G = k_l|x)} &= \log \frac{h_k(x) \cdot \pi_k}{h_l(x) \cdot \pi_l} = \log \frac{h_k(x)}{h_l(x)} + \log \frac{\pi_k}{\pi_l} = \\ &- \frac{1}{2} \cdot \left( (\mathbf{x} - \boldsymbol{\mu}_k)^T \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}_k) - (\mathbf{x} - \boldsymbol{\mu}_l)^T \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}_l) \right) + \log \frac{\pi_k}{\pi_l} \\ &= \underbrace{\left( -\frac{1}{2} \cdot ((\boldsymbol{\mu}_k + \boldsymbol{\mu}_l)^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{\mu}_k + \boldsymbol{\mu}_l)) \right)}_{\text{Unterschied der Mittelwerte}} + \underbrace{\mathbf{x}^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{\mu}_k + \boldsymbol{\mu}_l)}_{\text{linear in } \mathbf{x}} + \underbrace{\log \frac{\pi_k}{\pi_l}}_{\text{log-Odds Ratio der Klassen}} \end{aligned}$$

# PCA vs. LDA

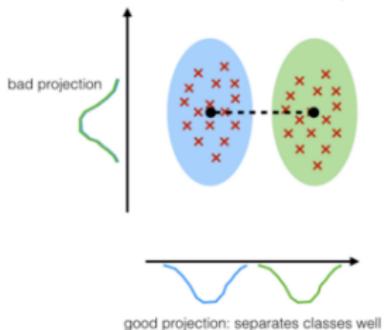
## PCA:

component axes that maximize the variance



## LDA:

maximizing the component axes for class-separation



# Mathematik hinter Diskriminanzanalyse erklärt

Das bedeutet, dass Diskriminanzanalyse die **Wahrscheinlichkeiten**, in unterschiedliche “Glocken” zu gehören, miteinander **vergleicht** und den Wert in die Klasse mit der größten Wahrscheinlichkeit klassifiziert.

Bei **linearer** Diskriminanzanalyse wird immer **dieselbe Kovarianzstruktur  $\Sigma$**  angenommen, bei **quadratischer** Diskriminanzanalyse eine **unterschiedliche** für jede “Glocke”.

# Klassifikation - Support Vector Machine

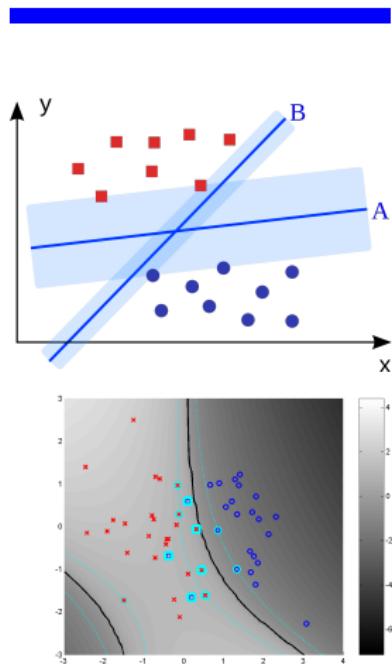
## Ablauf des Algorithmus

1. Bestimme auf den Trainingsdaten eine lineare oder nichtlineare Funktion, die die Daten optimal trennt
2. Klassifizierte, indem
  - 2.1 die Normalabstände aller Testdatenpunkte zur Trennlinie modifiziert um einen Biasterm  $b$  ermittelt werden

$$\mathbf{w} \cdot \mathbf{x} + b = 0$$

dieser Ausdruck ist positiv, negativ oder gleich 0 (auf der Trennlinie)

- 2.2 für alle Punkte aus dem Testdatensatz weise das Label je nach Vorzeichen zu
3. Validiere, indem die wahren Labels und die vom Algorithmus zugewiesenen Labels verglichen werden



(türkise Punkte und Linien = Diskriminanzfunktion der Trainingsdaten)  
(Trennlinie = schwarze Linie)

# Geradengleichungen in Ebene und Vektorraum

**Normalvektorform:**  $w \cdot x + b = 0$  bedeutet

$$\begin{pmatrix} w_1 \\ w_2 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + b = 0 \text{ oder } \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = -b$$

Der Vektor  $\begin{pmatrix} w_1 \\ w_2 \end{pmatrix}$  ist also die Normalrichtung des Trägervektors (support vector).

**implizite inhomogene Geradengleichung:**  $w_1 \cdot x + w_2 \cdot y = -b$

**explizite inhomogene Geradengleichung:**  $y = k \cdot x + d$  mit

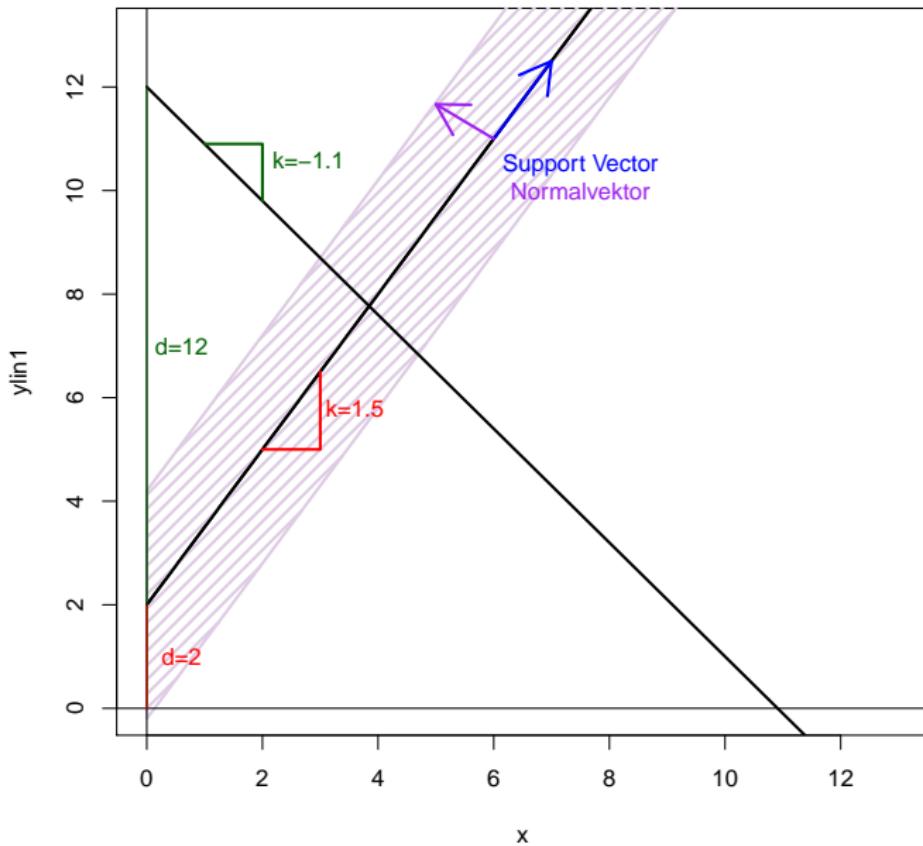
$$k = \frac{w_2}{w_1} \text{ und } d = -b$$

Der Biasterm  $b$  ist also der negative Achsenabschnitt der Geraden, welche den Trägervektor (Supportvektor) beinhaltet.

**Parameterdarstellung:**  $X = \begin{pmatrix} 0 \\ d \end{pmatrix} + \lambda \cdot \begin{pmatrix} w_2 \\ -w_1 \end{pmatrix}$

# Was bedeuten w und b graphisch?

Lineare Diskriminanzfunktion = Gerade im Vektorraum



# Klassifikation - Perceptron

## Ablauf des Algorithmus

1. Initialisiere die Gewichte der Neuronen (z.B. alle = 0)

2. Klassifizierte, indem iterativ

2.1 die Thresholdfunktion

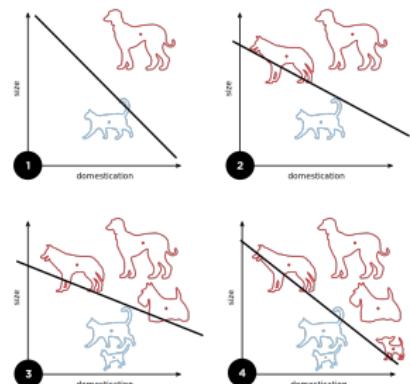
$$f(x) = \begin{cases} 1 & \text{if } \mathbf{w} \cdot \mathbf{x} + b > 0, \\ 0 & \text{otherwise} \end{cases}$$

berechnet wird

2.2 für alle Punkte aus dem Trainingsdatensatz  
ermittle, ob die Klassifikation stimme

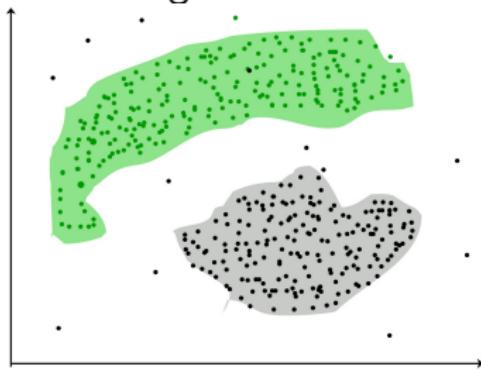
2.3 passe die Gewichte an, sodass  $\sum_{i=1}^m w_i x_i$   
entsprechend erhöht oder verringert wird,  
um eine andere Klassifikation zu erzielen

3. Validiere, indem du die Klassifikation mit den finalen Gewichten  
auf Testdaten anwendest und die wahren Labels und die vom  
Algorithmus zugewiesenen Labels verglichen werden



# Mustererkennung - Clustering

- ▶ die Anzahl der Klassen/Gruppen ist unbekannt und muss erst geschätzt werden
- ▶ eindimensionale Variablen: Multimodalität
- ▶ Was bedeutet Clustering im mehrdimensionalen Raum?



2-dimensionales Clustering ist noch graphisch darstellbar und z.B. in Bilderkennung in Verwendung

# Mustererkennung - wichtigste Arten von Clustering

- ▶ **k-means** clustering

es werden k Mittelwerte angepasst, sodass die Summe der quadratischen Abstände minimiert wird (**kleinste Quadrate Methode**)

- ▶ **hierarchical** clustering

es werden die **Abstände (Distanzen)** zwischen den Punkten und Werten gemessen und anhand der Abstände zusammengehörende Punkte zu Clustern zusammengefasst

- ▶ **distribution-based** clustering

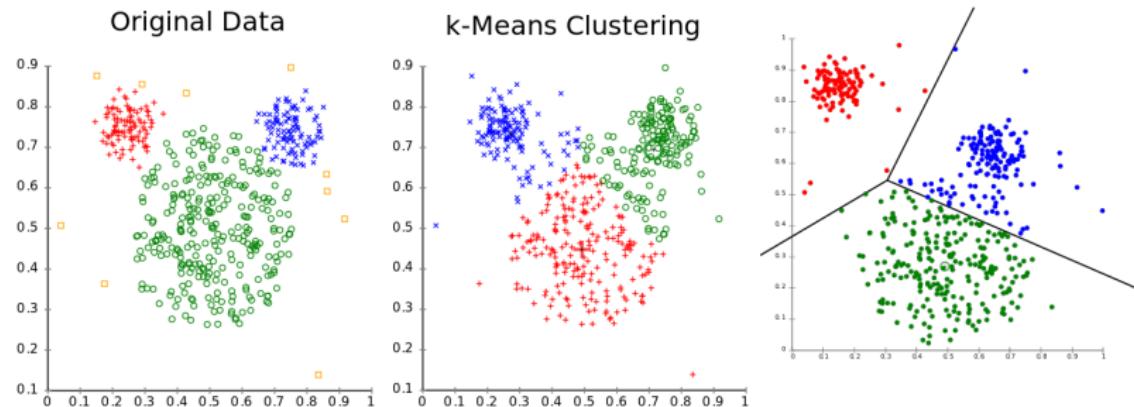
es werden k (multivariate) **Normalverteilungen** angepasst und deren Mittelwerte und (Ko-)Varianzen geschätzt  
Maximum Likelihood, Bayesianische Schätzung

- ▶ **density-based** clustering

anhand von **Punktdichten** werden die Punkte Clustern zugeordnet

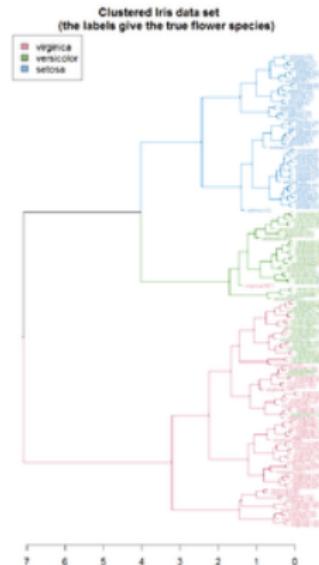
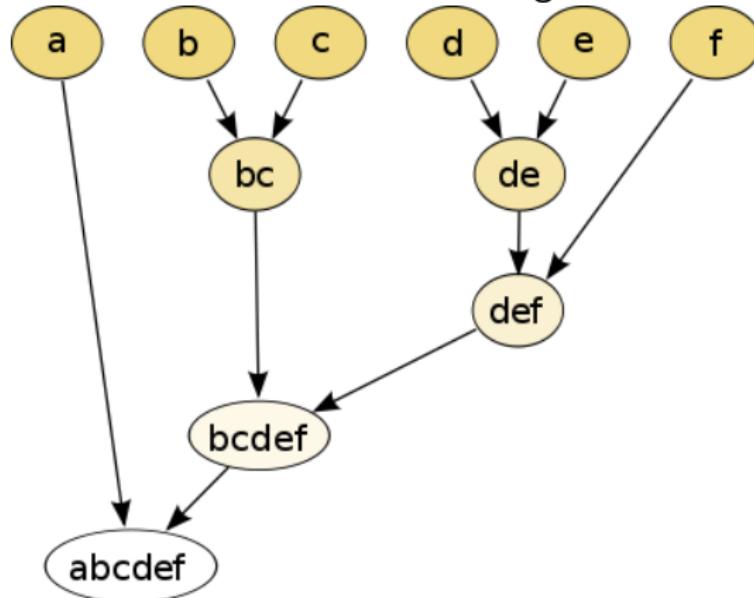
# Mustererkennung - k-means Clustering

es werden  $k$  Mittelwerte angepasst, sodass die Summe der quadratischen Abstände minimiert wird (**kleinste Quadrate Methode**)



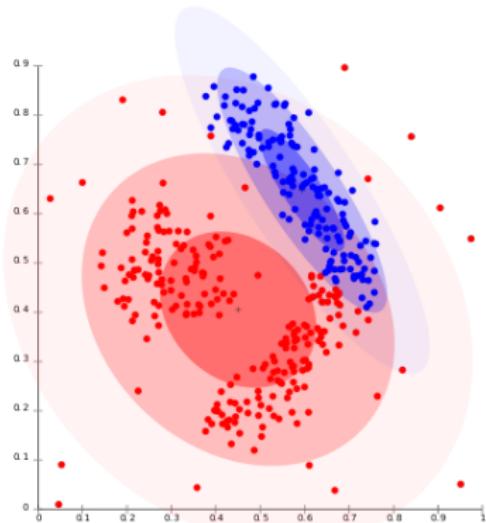
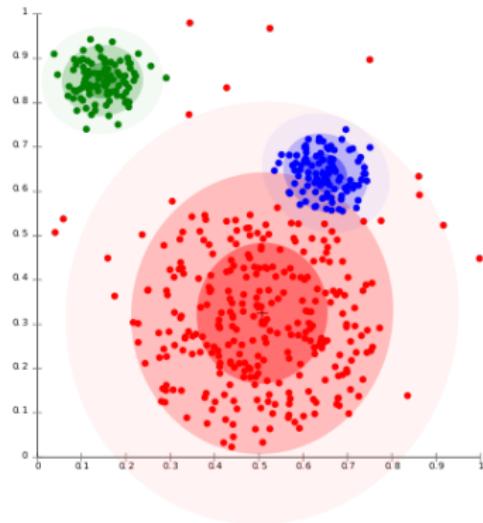
# Mustererkennung - hierarchical Clustering

es werden die **Abstände (Distanzen)** zwischen den Punkten und Werten gemessen und anhand der Abstände zusammengehörende Punkte zu Clustern zusammengefasst



# Mustererkennung - distribution-based Clustering

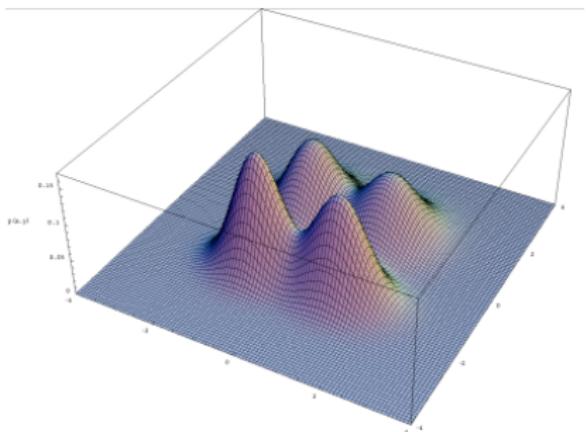
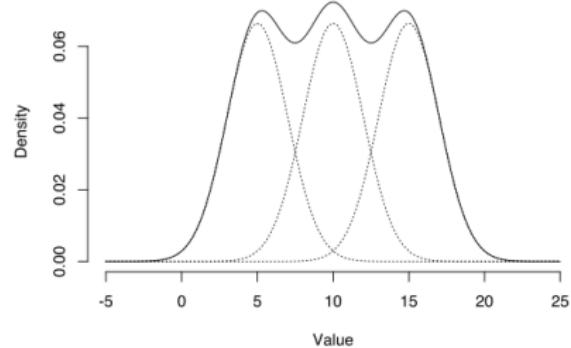
es werden **k** (multivariate) **Normalverteilungen** angepasst und deren Mittelwerte und (Ko-)Varianzen geschätzt



# Mustererkennung - distribution-based Clustering

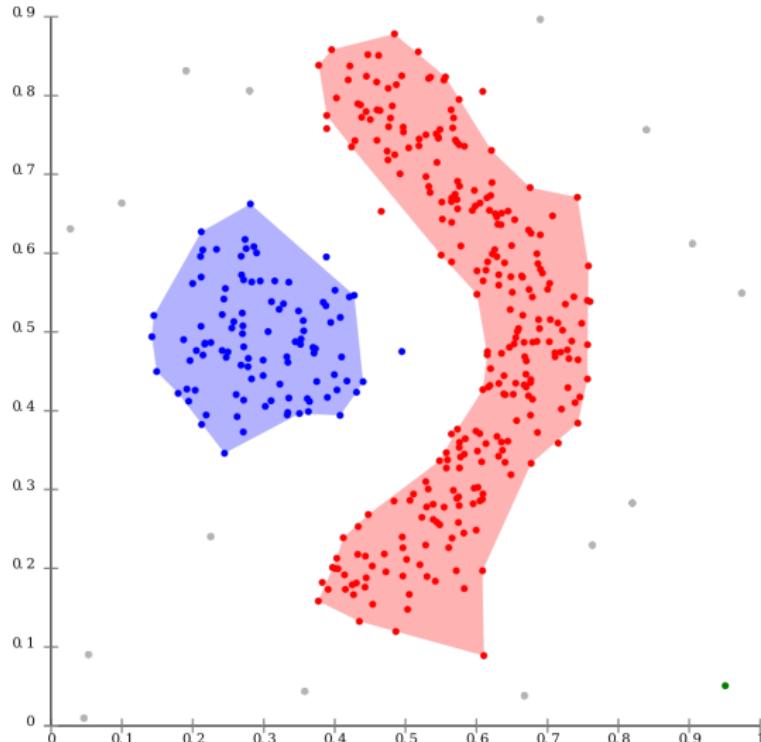
was bedeutet Mischen von Normalverteilungen?

Modellierung von Multimodalität



# Mustererkennung - density-based Clustering

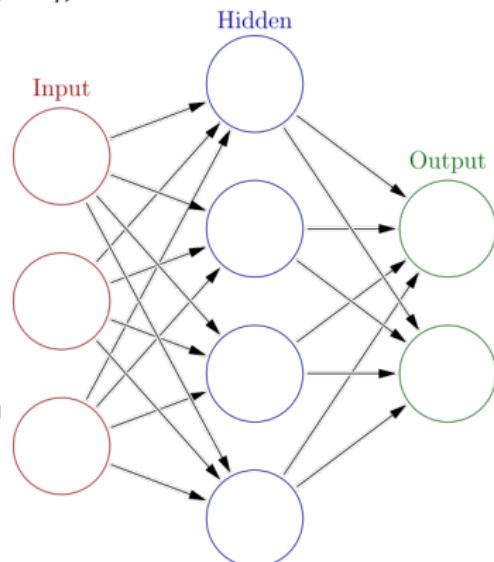
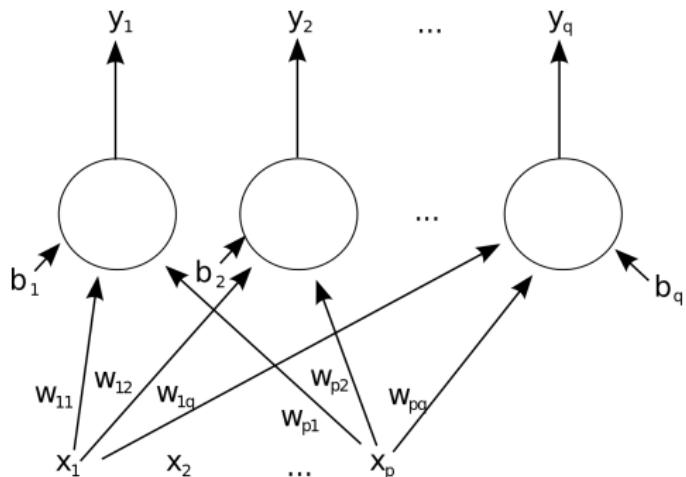
anhand von **Punktdichten** werden die Punkte Clustern zugeordnet



# Mustererkennung - neural networks

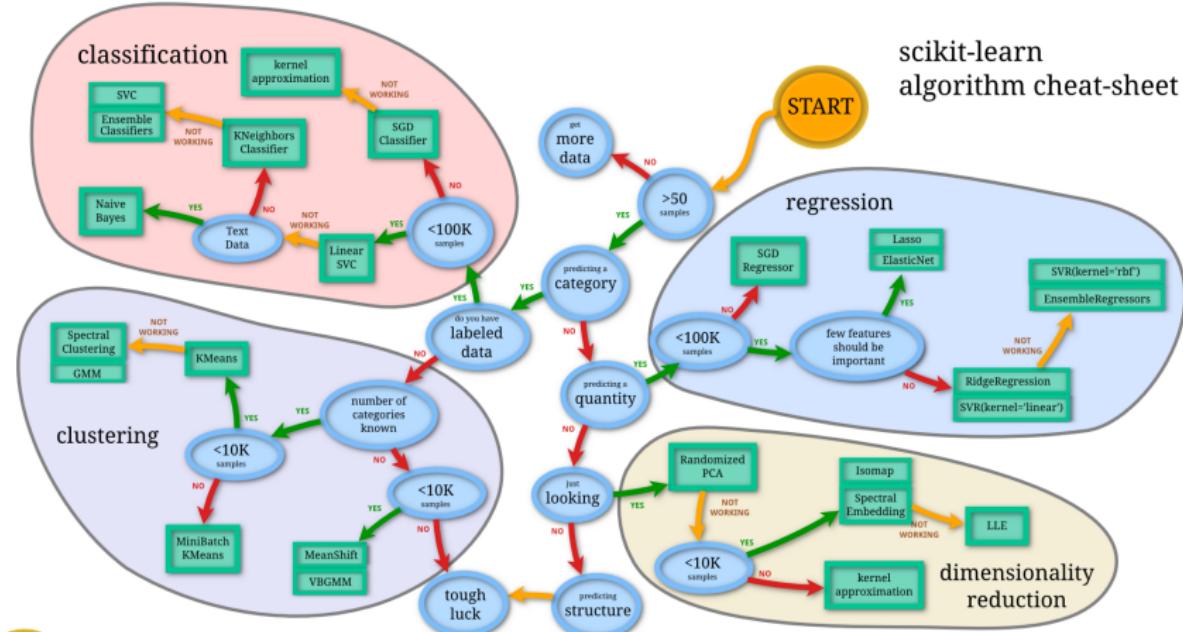
**unsupervised learning** mit Graphen, die **Hierarchien (layers)** enthalten

wie Perceptron ermittle  $y_q = K * (\sum (x_i * w_{iq}) - b_q)$



# Überblick über Algorithmen anhand des Python scikit

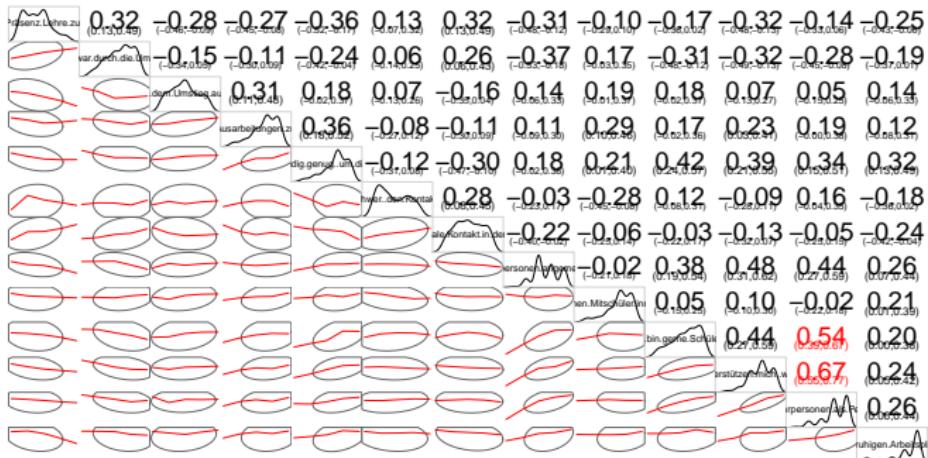
scikit-learn  
algorithm cheat-sheet



# Fallbeispiel - Faktorenanalyse

Umfrage von DOLD im Frühjahr 2021 zu Covid19 und Distance Learning

```
feedbackges<-read.csv("~/TGM/Projekte/FIE 2021/Feedback_MEDT_DOLD_gesamt.csv")
corrgram(feedbackges[,3:15], lower.panel=panel.ellipse,
         upper.panel=mypanel.conf, diag.panel=panel.density)
```



# Begleitung der Abteilung und des Schulversuchs Lernbüro durch Umfragen



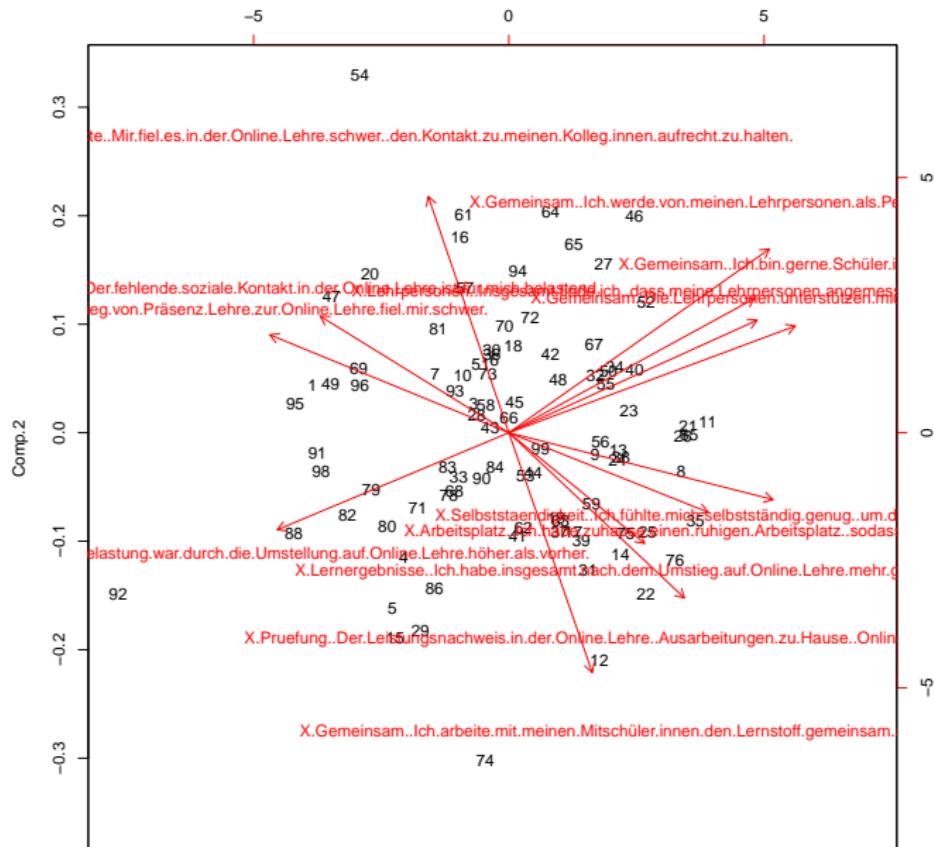
# Hauptkomponentenanalyse

```
options(width = 80)
mydata=na.omit(feedbackges[,3:15])
fit <- princomp(mydata, cor=TRUE)
summary(fit)

## Importance of components:
##                               Comp.1    Comp.2    Comp.3    Comp.4    Comp.5
## Standard deviation      1.9507546 1.3247319 1.1115038 1.08626506 0.89560997
## Proportion of Variance 0.2927264 0.1349934 0.0950339 0.09076706 0.06170132
## Cumulative Proportion   0.2927264 0.4277199 0.5227538 0.61352082 0.67522215
##                               Comp.6    Comp.7    Comp.8    Comp.9    Comp.10
## Standard deviation      0.87007915 0.81626517 0.80005207 0.76086893 0.71503471
## Proportion of Variance 0.05823367 0.05125299 0.04923718 0.04453243 0.03932882
## Cumulative Proportion   0.73345582 0.78470881 0.83394598 0.87847841 0.91780723
##                               Comp.11   Comp.12   Comp.13
## Standard deviation      0.66220964 0.60942626 0.50851162
## Proportion of Variance 0.03373243 0.02856926 0.01989108
## Cumulative Proportion   0.95153966 0.98010892 1.00000000
```

# Grafik zu Hauptkomponentenanalyse

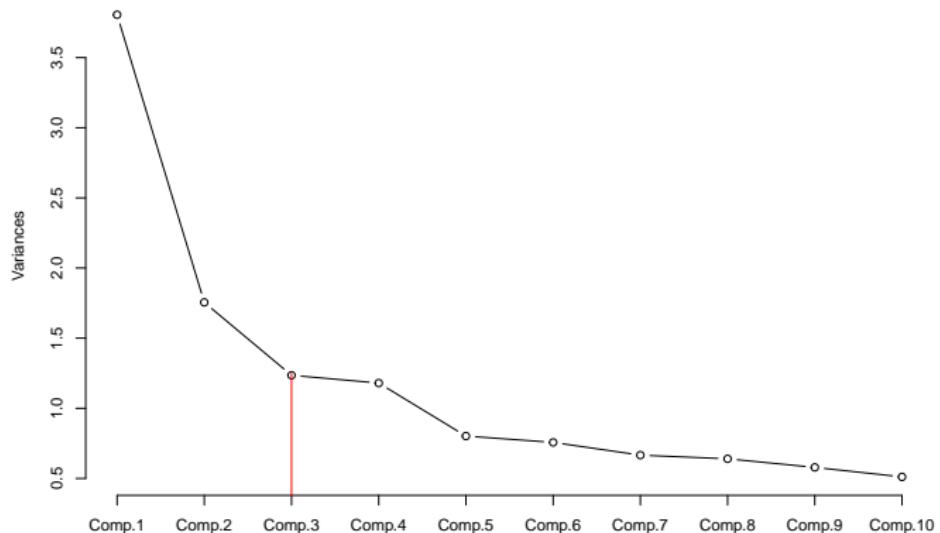
## biplot(fit)



# Grafik zu Hauptkomponentenanalyse - Wahl der Dimension

```
plot(fit,type="lines");lines(c(3,3),c(0,1.25),col=2,lwd=1)
```

fit



# Faktorenanalyse

```
fa3komp<-factanal(mydata,3, rotation="varimax")
fa3komp$loadings
```

```
## Loadings:
## 
## X.Umstieg..Der.Umstieg.von.Präsenz.Lehre.zur.Online.Lehre.fiel.mir.schwer. Factor1
## X.Arbeitsbelastung..Die.Arbeitsbelastung.war.durch.die.Umstellung.auf.Online.Lehre.höher.als.vorher. -0.200
## X.Lernergebnisse..Ich.habe.insgegant.nach.dem.Umstieg.auf.Online.Lehre.mehr.gelernt.als.ins.Präsenz. -0.360
## X.Prufung..Der.Leistungsnachweis.in.der.Online.Lehre..Ausarbeitungen.zu.Hause..Online.Tests.....gefieel.mir.besser.als.in.Präsenz. 0.171
## X.Selbststaendigkeit..Ich.fühle.sich.selbststaendig.genug..um.die.geforderten.Aufgaben.eigenstaendig.zu.loesen. 0.395
## X.Kontakte..Mir.fiel.es.in.der.Online.Lehre.schwer..den.Kontakt.zu.seinen.Kolleg.innen.aufrecht.zu.halten. 0.105
## X.KontakteFehlen..Der.fehlende.soziale.Kontakt.in.der.Online.Lehre.ist.für.mich.belastend. -0.136
## X.Lehrpersonen..Insgesamt.finde.ich..dass.meine.Lehrpersonen.angemessen.auf.den.Umstieg.auf.Online.Lehre.reagiert.haben. 0.495
## X.Gemeinsam..Ich.arbeite.mit.meinen.Mitschüler.innen.den.Lernstoff.gemeinsam.aus. 0.558
## X.Gemeinsam..Ich.bin.gerne.Schüler.in.dieser.Schule. 0.733
## X.Gemeinsam..Die.Lehrpersonen.unterstützen.mich..wenn.ich.etwas.nicht.verstanden.habe. 0.918
## X.Gemeinsam..Ich.werde.von.meinen.Lehrpersonen.als.Person.wahrgenommen.und.wertgeschätzt. 0.280
## X.Arbeitsplatz..Ich.habe.zuhause.einen.ruhigen.Arbeitsplatz..sodass.ich.ungestört.arbeiten.kann. 0.265
## 
## X.Umstieg..Der.Umstieg.von.Präsenz.Lehre.zur.Online.Lehre.fiel.mir.schwer. Factor2
## X.Arbeitsbelastung..Die.Arbeitsbelastung.war.durch.die.Umstellung.auf.Online.Lehre.höher.als.vorher. -0.621
## X.Lernergebnisse..Ich.habe.insgegant.nach.dem.Umstieg.auf.Online.Lehre.mehr.gelernt.als.ins.Präsenz. -0.507
## X.Prufung..Der.Leistungsnachweis.in.der.Online.Lehre..Ausarbeitungen.zu.Hause..Online.Tests.....gefieel.mir.besser.als.in.Präsenz. 0.330
## X.Selbststaendigkeit..Ich.fühle.sich.selbststaendig.genug..um.die.geforderten.Aufgaben.eigenstaendig.zu.loesen. 0.373
## X.Kontakte..Mir.fiel.es.in.der.Online.Lehre.schwer..den.Kontakt.zu.seinen.Kolleg.innen.aufrecht.zu.halten. -0.282
## X.KontakteFehlen..Der.fehlende.soziale.Kontakt.in.der.Online.Lehre.ist.für.mich.belastend. -0.502
## X.Lehrpersonen..Insgesamt.finde.ich..dass.meine.Lehrpersonen.angemessen.auf.den.Umstieg.auf.Online.Lehre.reagiert.haben. 0.352
## X.Gemeinsam..Ich.arbeite.mit.meinen.Mitschüler.innen.den.Lernstoff.gemeinsam.aus. 0.191
## X.Gemeinsam..Ich.bin.gerne.Schüler.in.dieser.Schule. 0.211
## X.Gemeinsam..Die.Lehrpersonen.unterstützen.mich..wenn.ich.etwas.nicht.verstanden.habe. 0.284
## X.Gemeinsam..Ich.werde.von.meinen.Lehrpersonen.als.Person.wahrgenommen.und.wertgeschätzt. 0.265
## X.Arbeitsplatz..Ich.habe.zuhause.einen.ruhigen.Arbeitsplatz..sodass.ich.ungestört.arbeiten.kann. 0.232
## 
## X.Umstieg..Der.Umstieg.von.Präsenz.Lehre.zur.Online.Lehre.fiel.mir.schwer. Factor3
## X.Arbeitsbelastung..Die.Arbeitsbelastung.war.durch.die.Umstellung.auf.Online.Lehre.höher.als.vorher. 0.172
## X.Lernergebnisse..Ich.habe.insgegant.nach.dem.Umstieg.auf.Online.Lehre.mehr.gelernt.als.ins.Präsenz. 0.186
## X.Prufung..Der.Leistungsnachweis.in.der.Online.Lehre..Ausarbeitungen.zu.Hause..Online.Tests.....gefieel.mir.besser.als.in.Präsenz. 0.405
## X.Selbststaendigkeit..Ich.fühle.sich.selbststaendig.genug..um.die.geforderten.Aufgaben.eigenstaendig.zu.loesen. 0.294
## X.Kontakte..Mir.fiel.es.in.der.Online.Lehre.schwer..den.Kontakt.zu.seinen.Kolleg.innen.aufrecht.zu.halten. -0.367
## X.KontakteFehlen..Der.fehlende.soziale.Kontakt.in.der.Online.Lehre.ist.für.mich.belastend. -0.102
## X.Lehrpersonen..Insgesamt.finde.ich..dass.meine.Lehrpersonen.angemessen.auf.den.Umstieg.auf.Online.Lehre.reagiert.haben. 0.805
## X.Gemeinsam..Ich.arbeite.mit.meinen.Mitschüler.innen.den.Lernstoff.gemeinsam.aus. 0.111
## X.Gemeinsam..Ich.bin.gerne.Schüler.in.dieser.Schule. 0.232
## X.Gemeinsam..Die.Lehrpersonen.unterstützen.mich..wenn.ich.etwas.nicht.verstanden.habe. 0.185
## X.Gemeinsam..Ich.werde.von.meinen.Lehrpersonen.als.Person.wahrgenommen.und.wertgeschätzt. 0.121
## X.Arbeitsplatz..Ich.habe.zuhause.einen.ruhigen.Arbeitsplatz..sodass.ich.ungestört.arbeiten.kann. 0.093
## Cumulative Var 0.185 0.306 0.399
```

# Factor-Loadings und Interpretation

```
loadtab<-as.table((factanal(mydata,3, rotation="varimax"))$loadings)
rownames(loadtab)<-c(paste0("F",1:13))
print.xtable(xtable((loadtab)),comment = F)
```

	Factor1	Factor2	Factor3
F1	-0.20	-0.62	-0.16
F2	-0.36	-0.51	0.17
F3	0.06	0.33	0.19
F4	0.17	0.27	0.41
F5	0.39	0.37	0.29
F6	0.11	-0.28	-0.37
F7	-0.14	-0.50	-0.10
F8	0.49	0.35	-0.04
F9	0.02	-0.04	0.81
F10	0.56	0.19	0.05
F11	0.73	0.21	0.11
F12	0.92	-0.02	-0.01
F13	0.28	0.28	0.23

# Fallbeispiel - Klassifikation in R mit kNN

```
data(diamonds); dia <- data.frame(diamonds); head(dia)

##   carat      cut color clarity depth table price     x     y     z
## 1 0.23    Ideal    E    SI2  61.5    55  326 3.95 3.98 2.43
## 2 0.21  Premium    E    SI1  59.8    61  326 3.89 3.84 2.31
## 3 0.23     Good    E    VS1  56.9    65  327 4.05 4.07 2.31
## 4 0.29  Premium    I    VS2  62.4    58  334 4.20 4.23 2.63
## 5 0.31     Good    J    SI2  63.3    58  335 4.34 4.35 2.75
## 6 0.24 Very Good    J   VVS2  62.8    57  336 3.94 3.96 2.48

## Normalisierung als Funktion definieren
nor <-function(x) { (x -min(x))/(max(x)-min(x)) }
```

```
# die erklärenden Spalten werden normalisiert
dia_norm <- as.data.frame(lapply(dia[,c(1,5,6,7,8,9,10)], nor))
summary(dia_norm)
```

```
##       carat           depth          table         price
##  Min.   :0.00000   Min.   :0.0000   Min.   :0.0000   Min.   :0.00000
##  1st Qu.:0.04158   1st Qu.:0.5000   1st Qu.:0.2500   1st Qu.:0.03374
##  Median :0.10395   Median :0.5222   Median :0.2692   Median :0.11218
##  Mean   :0.12431   Mean   :0.5208   Mean   :0.2780   Mean   :0.19499
##  3rd Qu.:0.17464   3rd Qu.:0.5417   3rd Qu.:0.3077   3rd Qu.:0.27022
##  Max.   :1.00000   Max.   :1.0000   Max.   :1.0000   Max.   :1.00000
##             x           y           z
##  Min.   :0.0000   Min.   :0.00000   Min.   :0.00000
##  1st Qu.:0.4385   1st Qu.:0.08014   1st Qu.:0.09151
##  Median :0.5307   Median :0.09694   Median :0.11101
##  Mean   :0.5336   Mean   :0.09736   Mean   :0.11128
##  3rd Qu.:0.6089   3rd Qu.:0.11104   3rd Qu.:0.12704
##  Max.   :1.0000   Max.   :1.00000   Max.   :1.00000
```

# Trainingsdaten und Testdaten

```
# Zufällige Aufteilung in Training und Testdaten
set.seed(457314)
ran <- sample(1:nrow(dia), 0.9 * nrow(dia))
dia_train <- dia_norm[ran,] ##extract testing set
dia_test <- dia_norm[-ran,]
dia_target <- as.factor(dia[ran,2]) # Zielvektor
test_target <- as.factor(dia[-ran,2])

library(class)
pr <- knn(dia_train,dia_test,cl=dia_target,k=20)
```

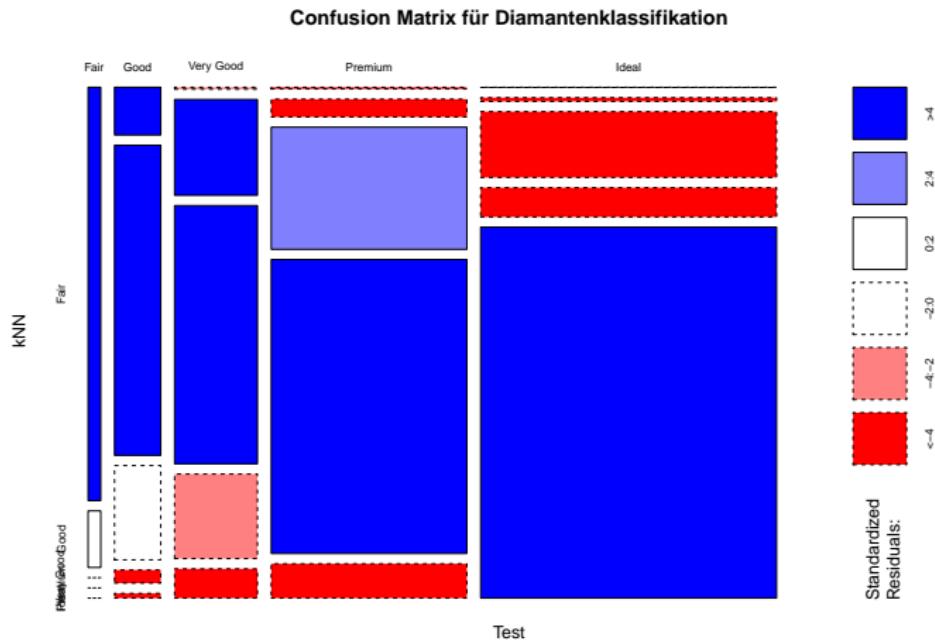
# Confusion Matrix

```
options(width = 100); (conf<-confusionMatrix(data=pr,reference = test_target))
```

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction  Fair Good Very Good Premium Ideal
##   Fair        95   13      0      0      0
##   Good        40  260     79     11      4
##   Very Good    3  144    388    127     44
##   Premium      6   63    433   1042    122
##   Ideal        1   20    353    157   1989
##
## Overall Statistics
##
##           Accuracy : 0.6997
##           95% CI  : (0.6872, 0.7119)
##   No Information Rate : 0.4003
##   P-Value [Acc > NIR] : < 2.2e-16
##
##           Kappa : 0.5702
##
## McNemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##           Class: Fair Class: Good Class: Very Good Class: Premium Class: Ideal
## Sensitivity       0.65517   0.52000     0.30966   0.7794   0.9213
## Specificity       0.99752   0.97262     0.92321   0.8462   0.8359
## Pos Pred Value    0.87963   0.65990     0.54958   0.6255   0.7893
## Neg Pred Value    0.99054   0.95200     0.81549   0.9209   0.9408
## Prevalence        0.02688   0.09270     0.23230   0.2479   0.4003
## Detection Rate    0.01761   0.04820     0.07193   0.1932   0.3687
## Detection Prevalence 0.02002   0.07304     0.13089   0.3089   0.4672
## Balanced Accuracy  0.82635   0.74631     0.61643   0.8128   0.8786
```

# Visualisierung der Confusion Matrix

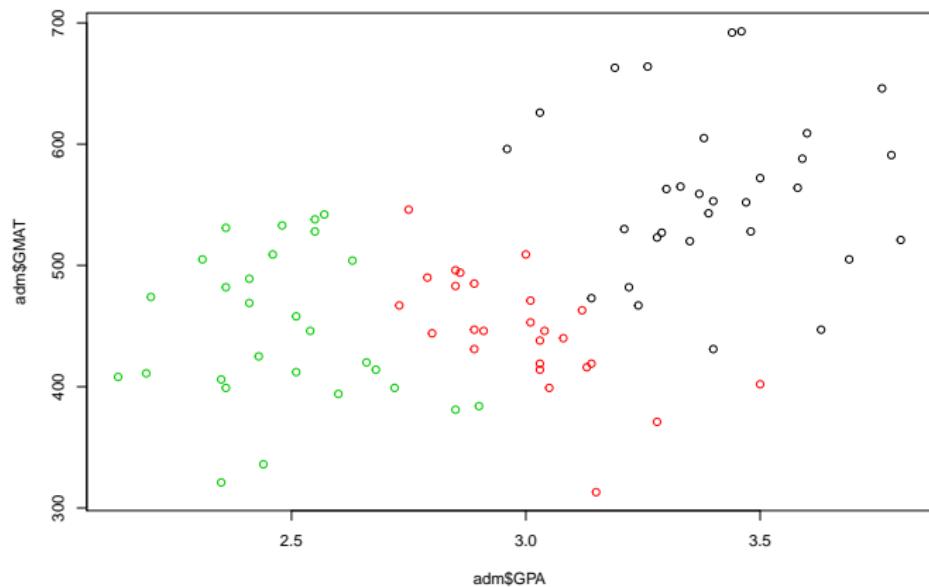
```
plot(conf$table, main="Confusion Matrix für Diamantenklassifikation",  
      xlab="Test", ylab="kNN", shade=T)
```



# Fallbeispiel - Diskriminanzanalyse

Grade Point Average (GPA) und Graduate Management Admission Test (GMAT)  
und tatsächliche Aufnahme (admission)

```
url <- 'http://www.biz.uiowa.edu/faculty/jledolter/DataMining/admission'
admit <- read.csv(url)
adm=data.frame(admit)
plot(adm$GPA,adm$GMAT,col=adm$De)
```



LD<sup>A</sup>

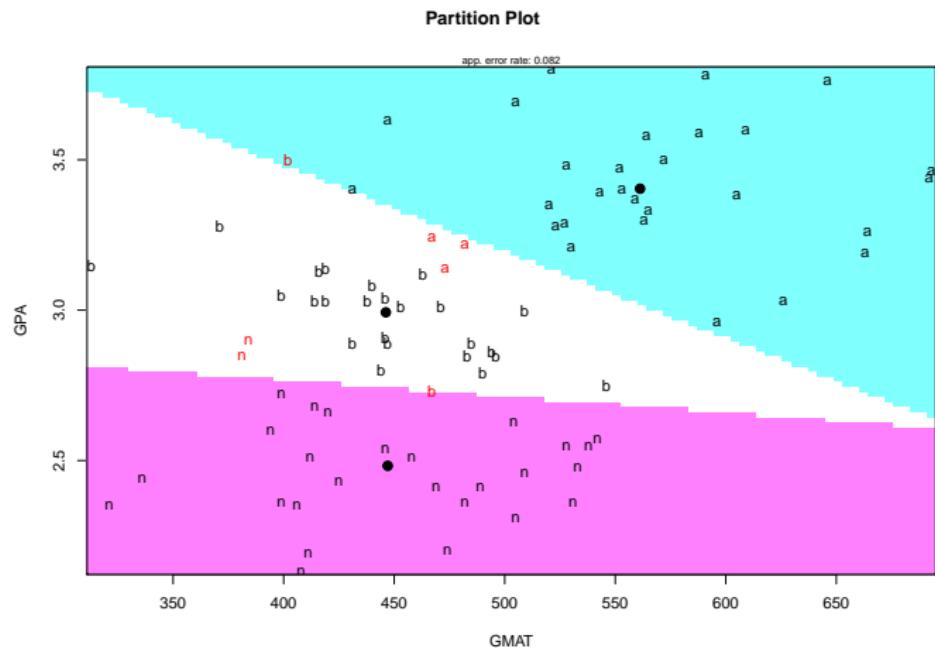
```
m1=MASS:::lda(De~,adm)
m1

## Call:
## lda(De ~ ., data = adm)
##
## Prior probabilities of groups:
##      admit    border  notadmit
## 0.3647059 0.3058824 0.3294118
##
## Group means:
##          GPA      GMAT
## admit   3.403871 561.2258
## border  2.992692 446.2308
## notadmit 2.482500 447.0714
##
## Coefficients of linear discriminants:
##          LD1       LD2
## GPA  5.008766354  1.87668220
## GMAT 0.008568593 -0.01445106
##
## Proportion of trace:
##      LD1       LD2
## 0.9673 0.0327
predict(m1,newdata=data.frame(GPA=3.21,GMAT=497))
```

```
## $class
## [1] admit
## Levels: admit border notadmit
##
## $posterior
##      admit    border    notadmit
## 1 0.5180421 0.4816015 0.0003563717
##
## $x
```

# Visualisierung von LDA

```
klaR:::partimat(De^~., data=adm, method="lda")
```



# QDA

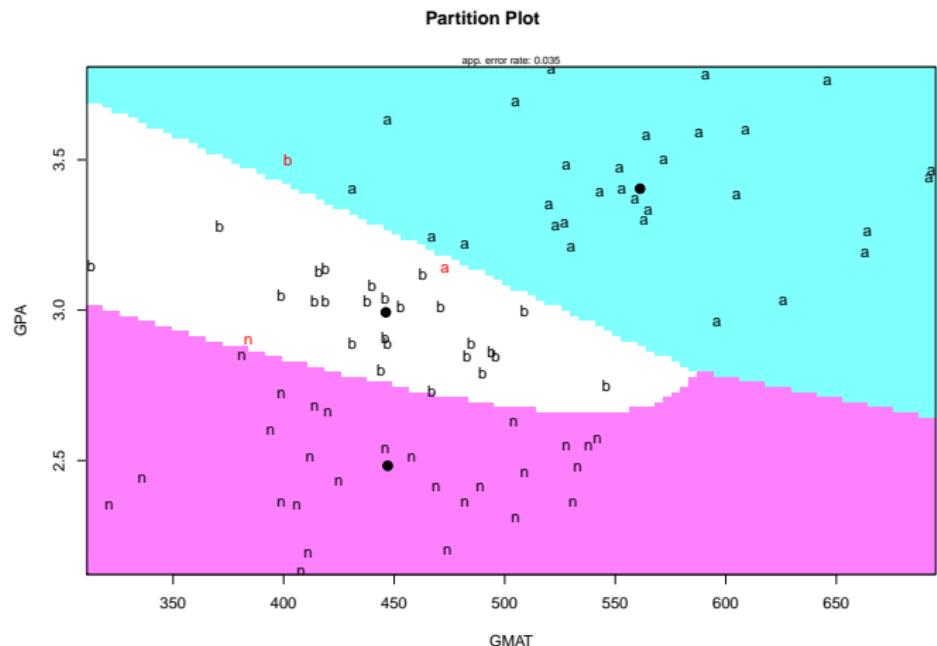
```
m2=MASS::qda(De~,adm)
m2

## Call:
## qda(De ~ ., data = adm)
##
## Prior probabilities of groups:
##      admit    border  notadmit
## 0.3647059 0.3058824 0.3294118
##
## Group means:
##          GPA      GMAT
## admit 3.403871 561.2258
## border 2.992692 446.2308
## notadmit 2.482500 447.0714
predict(m2,newdata=data.frame(GPA=3.21,GMAT=497))

## $class
## [1] admit
## Levels: admit border notadmit
##
## $posterior
##      admit    border  notadmit
## 1 0.9226763 0.0768693 0.0004544468
```

# Visualisierung von QDA

```
klaR:::partimat(De^~., data=adm, method="qda")
```



# Fallbeispiel - k Means Clustering in R

```
options(width = 100)
wineUrl <- 'http://archive.ics.uci.edu/ml/machine-learning-databases/wine/wine.data'
wine <- read.table(wineUrl, header=FALSE, sep=",",
                   stringsAsFactors=FALSE,
                   col.names=c('Cultivar', 'Alcohol', 'Malic.acid',
                               'Ash', 'Alcalinity.of.ash',
                               'Magnesium', 'Total.phenols',
                               'Flavanoids', 'Nonflavonoid.phenols',
                               'Proanthocyanin', 'Color.intensity',
                               'Hue', 'OD280.OD315.of.diluted.wines',
                               'Proline'
))
summary(wine)
```

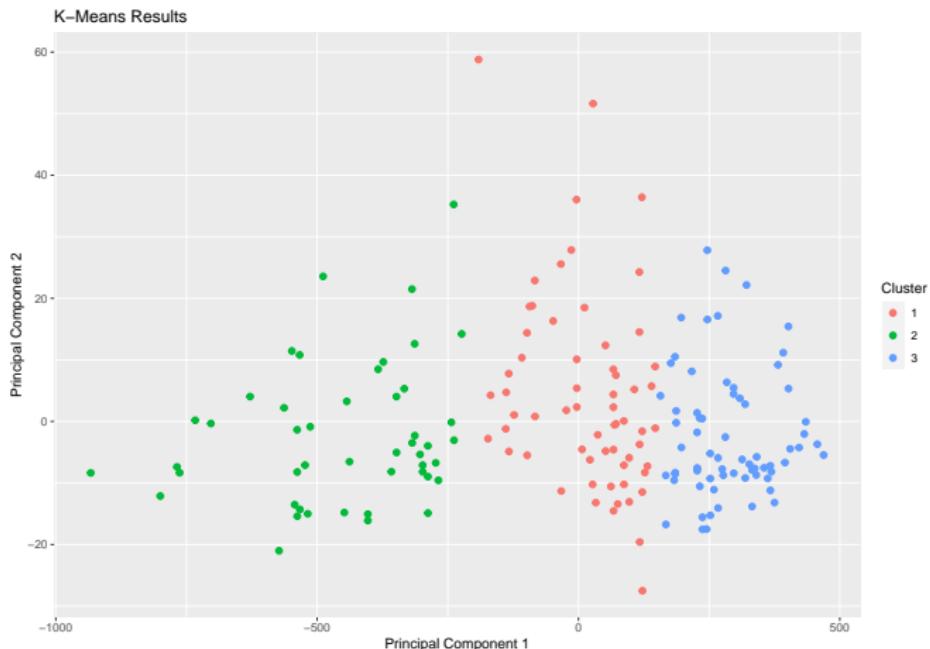
  

```
##      Cultivar       Alcohol      Malic.acid        Ash Alcalinity.of.ash   Magnesium
## Min. :1.000  Min. :11.03  Min. :0.740  Min. :1.360  Min. :10.60  Min. : 70.00
## 1st Qu.:1.000  1st Qu.:12.36  1st Qu.:1.603  1st Qu.:2.210  1st Qu.:17.20  1st Qu.: 88.00
## Median :2.000  Median :13.05  Median :1.865  Median :2.360  Median :19.50  Median : 98.00
## Mean  :1.938  Mean  :13.00  Mean  :2.336  Mean  :2.367  Mean  :19.49  Mean  : 99.74
## 3rd Qu.:3.000  3rd Qu.:13.68  3rd Qu.:3.083  3rd Qu.:2.558  3rd Qu.:21.50  3rd Qu.:107.00
## Max. :3.000  Max. :14.83  Max. :5.800  Max. :3.230  Max. :30.00  Max. :162.00
##      Total.phenols    Flavanoids Nonflavonoid.phenols Proanthocyanin Color.intensity
## Min. :0.980  Min. :0.340  Min. :0.1300  Min. :0.410  Min. : 1.280
## 1st Qu.:1.742  1st Qu.:1.205  1st Qu.:0.2700  1st Qu.:1.250  1st Qu.: 3.220
## Median :2.355  Median :2.135  Median :0.3400  Median :1.555  Median : 4.690
## Mean  :2.295  Mean  :2.029  Mean  :0.3619  Mean  :1.591  Mean  : 5.058
## 3rd Qu.:2.800  3rd Qu.:2.875  3rd Qu.:0.4375  3rd Qu.:1.950  3rd Qu.: 6.200
## Max. :3.880  Max. :5.080  Max. :0.6600  Max. :3.580  Max. :13.000
##      Hue          OD280.OD315.of.diluted.wines     Proline
## Min. :0.4800  Min. :1.270          Min. : 278.0
## 1st Qu.:0.7825 1st Qu.:1.938          1st Qu.:500.5
## Median :0.9650  Median :2.780          Median : 673.5
## Mean  :0.9574  Mean  :2.612          Mean  : 746.9
## 3rd Qu.:1.1200 3rd Qu.:3.170          3rd Qu.:985.0
## Max. :1.7100  Max. :4.000          Max. :1680.0
```

## Anwendung von k Means Clustering

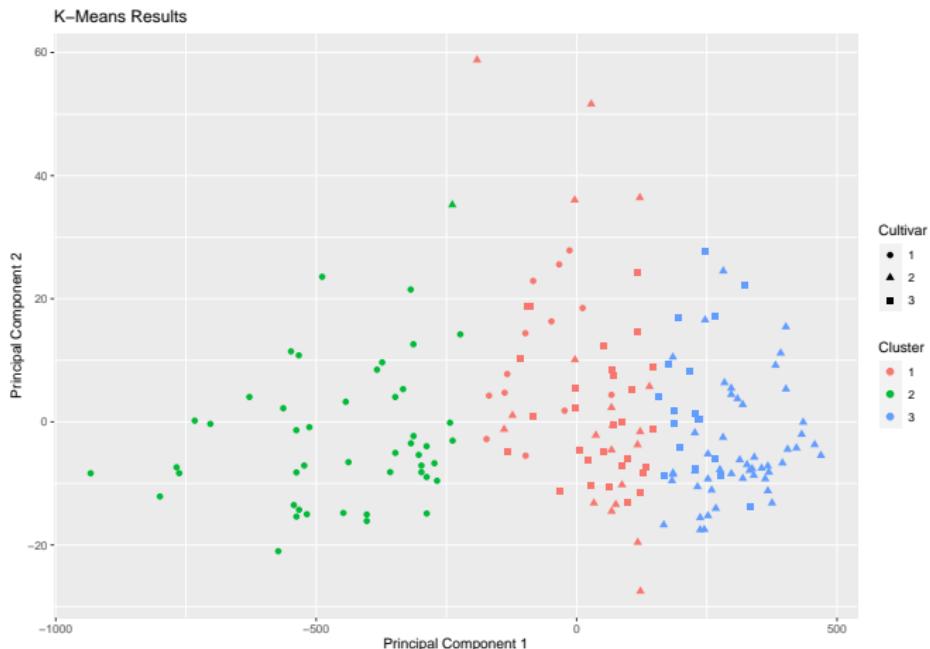
# Grafische Darstellung von k Means Clustering

```
library(useful) # lädt den plot-Befehl  
plot.kmeans(wineK3, data=wineTrain)
```



## Prädiktion mit neuen Daten

```
plot.kmeans(wineK3, data=wine, class="Cultivar")
```

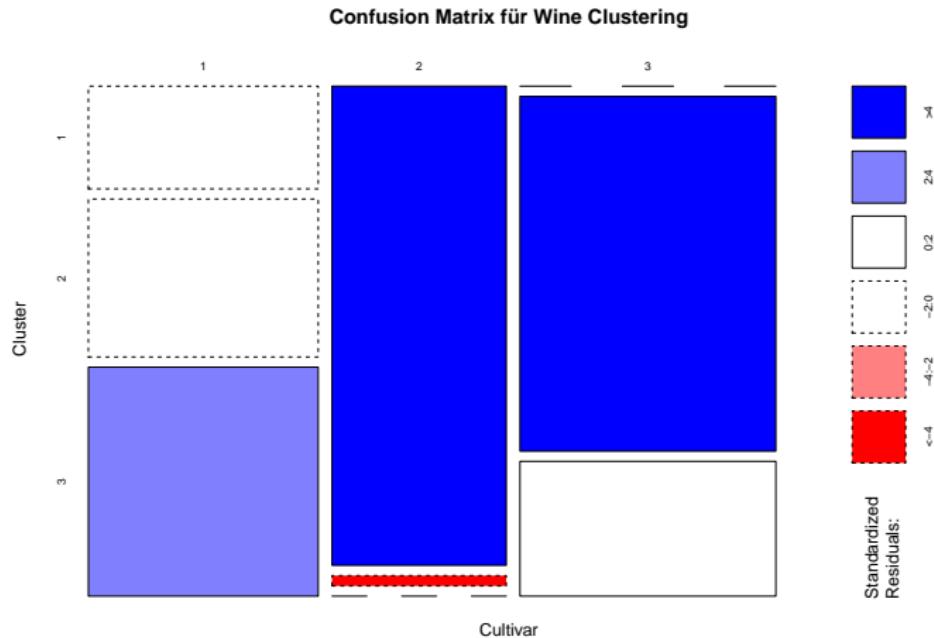


# Prädiktion mit neuen Daten

```
(conf<-confusionMatrix(reference=factor(wine$Cultivar), data=factor(wineK3$cluster)))  
  
## Confusion Matrix and Statistics  
##  
##          Reference  
## Prediction 1 2 3  
##          1 13 20 29  
##          2 46  1  0  
##          3  0 50 19  
##  
## Overall Statistics  
##  
##          Accuracy : 0.1854  
##          95% CI : (0.1312, 0.2504)  
##          No Information Rate : 0.3989  
##          P-Value [Acc > NIR] : 1  
##  
##          Kappa : -0.2074  
##  
## McNemar's Test P-Value : <2e-16  
##  
## Statistics by Class:  
##  
##          Class: 1 Class: 2 Class: 3  
## Sensitivity      0.22034 0.014085  0.3958  
## Specificity       0.58824 0.570093  0.6154  
## Pos Pred Value    0.20968 0.021277  0.2754  
## Neg Pred Value    0.60345 0.465649  0.7339  
## Prevalence        0.33146 0.398876  0.2697  
## Detection Rate    0.07303 0.005618  0.1067  
## Detection Prevalence 0.34831 0.264045  0.3876  
## Balanced Accuracy  0.40429 0.292089  0.5056
```

# Visualisierung der Confusion Matrix

```
plot(conf$table,  
      main="Confusion Matrix für Wine Clustering",  
      xlab="Cultivar", ylab="Cluster", shade=T)
```



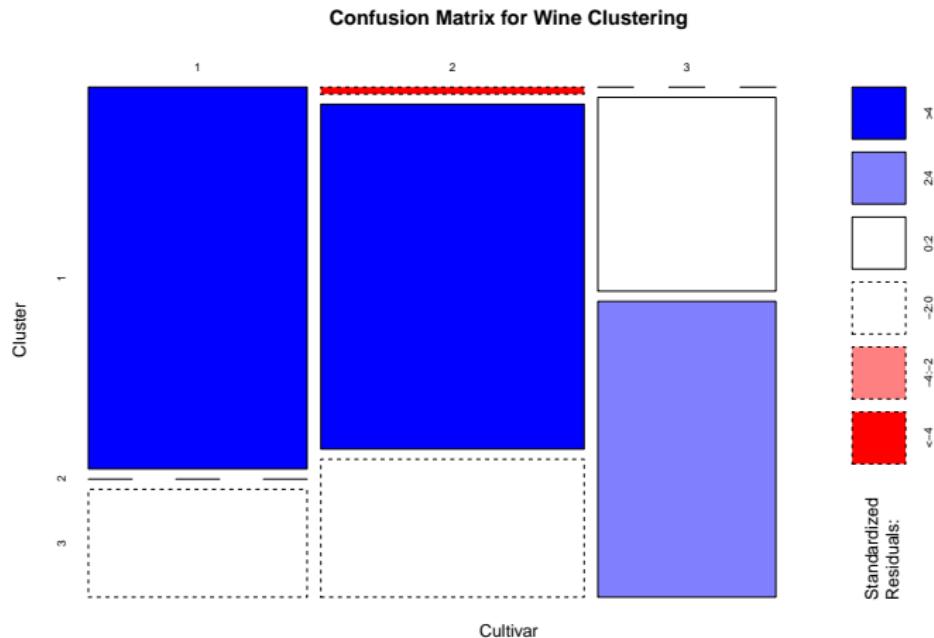
# Problem: Label Switching

```
clusterlabelsunswitched<-wineK3$cluster
clusterlabelsunswitched[wineK3$cluster==1]<-3
clusterlabelsunswitched[wineK3$cluster==2]<-1
clusterlabelsunswitched[wineK3$cluster==3]<-2
confusionMatrixunswitched<-confusionMatrix(
  reference=factor(wine$Cultivar), data=factor(clusterlabelsunswitched))
confusionMatrixunswitched

## Confusion Matrix and Statistics
##
##          Reference
## Prediction 1 2 3
##           1 46 1 0
##           2 0 50 19
##           3 13 20 29
##
## Overall Statistics
##
##          Accuracy : 0.7022
##                 95% CI : (0.6293, 0.7683)
##   No Information Rate : 0.3989
## P-Value [Acc > NIR] : 2.627e-16
##
##          Kappa : 0.5515
##
## McNemar's Test P-Value : 0.00287
##
## Statistics by Class:
##
##          Class: 1 Class: 2 Class: 3
## Sensitivity      0.7797  0.7042  0.6042
## Specificity       0.9916  0.8224  0.7462
## Pos Pred Value    0.9787  0.7246  0.4677
## Neg Pred Value    0.9008  0.8073  0.8362
## Prevalence        0.3315  0.3989  0.2697
## Detection Rate    0.2584  0.2809  0.1629
## Detection Prevalence 0.2640  0.3876  0.3483
## Balanced Accuracy  0.8856  0.7633  0.6752
```

# Visualisierung der Confusion Matrix

```
plot(table(wine$Cultivar,clusterlabelsunswitched),  
     main="Confusion Matrix for Wine Clustering",  
     xlab="Cultivar", ylab="Cluster", shade=T)
```



# Suchen der besten Clusteranzahl

```
wineBest <- FitKMeans(wineTrain, max.clusters=20, nstart=25, seed=278613)
wineBest
```

```
##      Clusters   Hartigan AddCluster
## 1          2  505.429310     TRUE
## 2          3  160.411331     TRUE
## 3          4  135.707228     TRUE
## 4          5   78.445289     TRUE
## 5          6   71.489710     TRUE
## 6          7   97.582072     TRUE
## 7          8   46.772501     TRUE
## 8          9   33.198650     TRUE
## 9         10   33.277952     TRUE
## 10        11   33.465424     TRUE
## 11        12   17.940296     TRUE
## 12        13   33.268151     TRUE
## 13        14    6.434996    FALSE
## 14        15    7.833562    FALSE
## 15        16   46.783444     TRUE
## 16        17   12.229408     TRUE
## 17        18   10.261821     TRUE
## 18        19  -13.576343    FALSE
## 19        20   56.373939     TRUE
```

# Grafische Suche der besten Clusteranzahl

PlotHartigan(wineBest)

