



**Prof. Dr. Adrian Ulges**

**Empolis Workshop “Machine Learning”**

# **Clustering**

**Hochschule RheinMain**

Department DCSM (*Design, Computer Science, Media*)

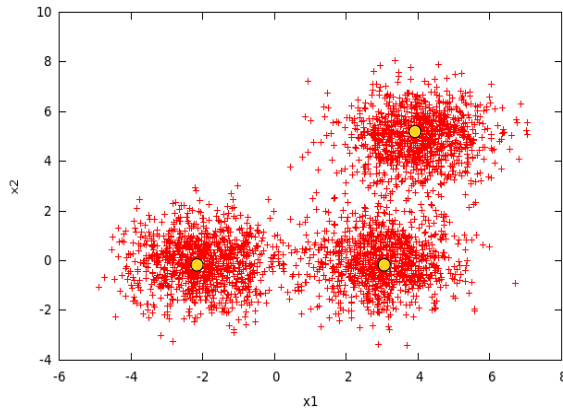
## ► Clustering

- Clustering im Vektorraum: K-Means, EM
- Model Selection (→ Anzahl der Cluster)
- Agglomeratives Clustering, Topic Modeling
- **Python-Beispiel: News-Clustering**

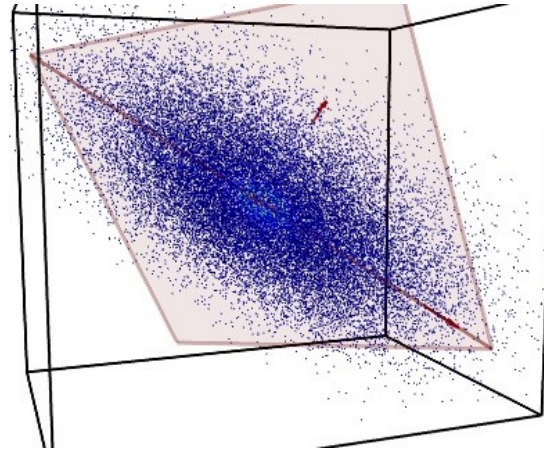
# Unüberwachtes Lernen

Prof. Adrian Ulges

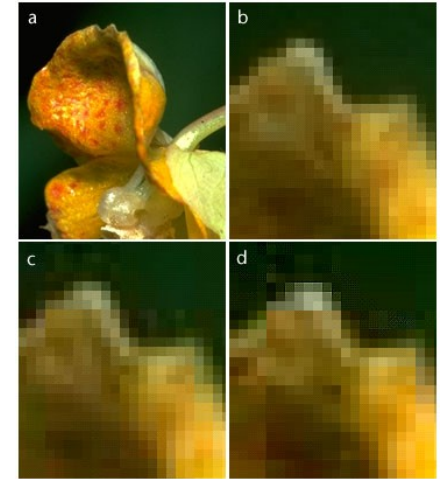
Fachbereich DCSM / Informatik  
Hochschule RheinMain



**Clustering:** Teile die Daten in kohärente Gruppen ein



**Dimensionality Reduction:**  
Komprimiere die Daten



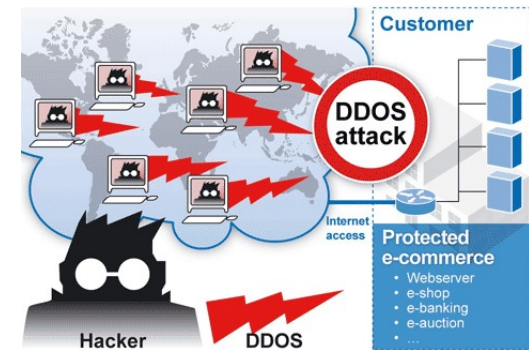
amazon.com

Recommended for You

Amazon.com has new recommendations for you based on [items](#) you purchased or told us you own.



**Itemset Mining:** Finde häufige Substrukturen in den Daten.

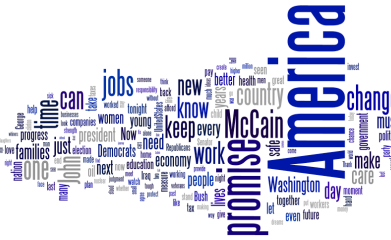


**Anomaly Detection:** Finde Outlier / Ausreißer in den Daten

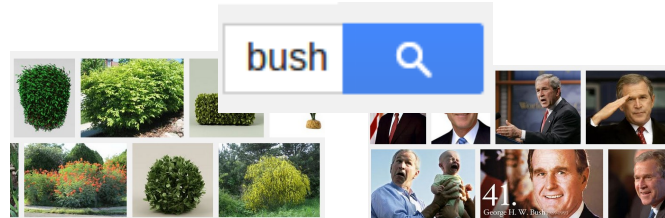
# Clustering: Anwendungen

**Prof. Adrian Ulges**  
Fachbereich DCSM / Informatik  
Hochschule RheinMain

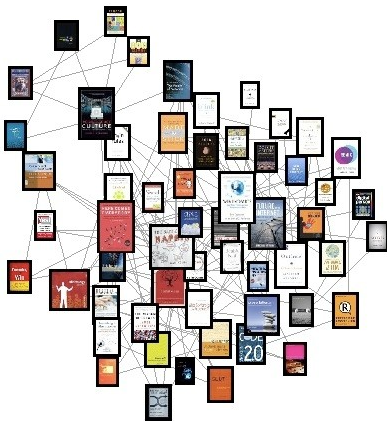
- Clustering-Verfahren finden zahlreiche Anwendungen in den **verschiedensten Gebieten**
  - Marktforschung
  - Information Retrieval
  - Computer Vision
  - Social Networks
  - Data Mining



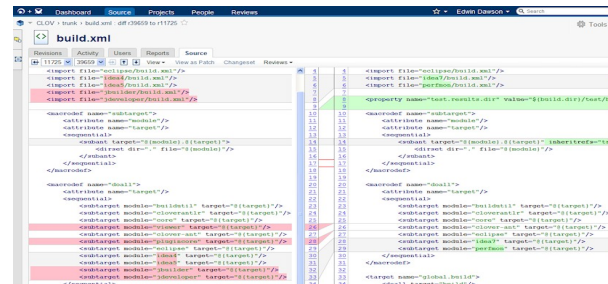
# Text clustern



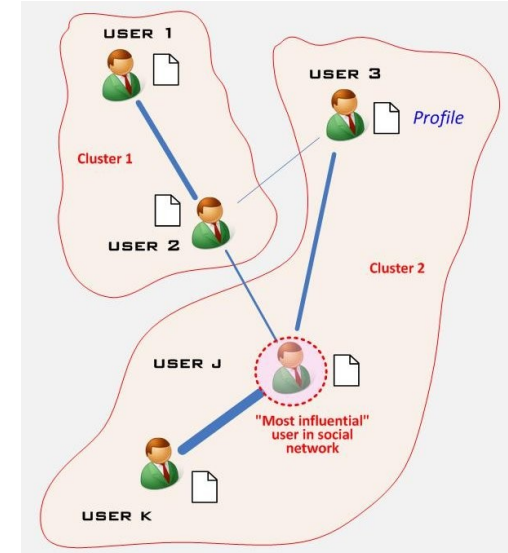
## Suchergebnisse clustern



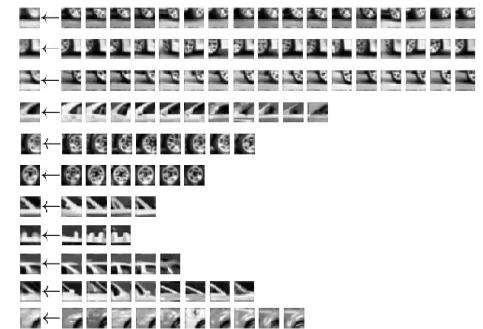
## Produkte clustern



## Sourcecode clustern



## Benutzer clustern



## Bildstücke clustern

**Grundlagen**

**K-Means**

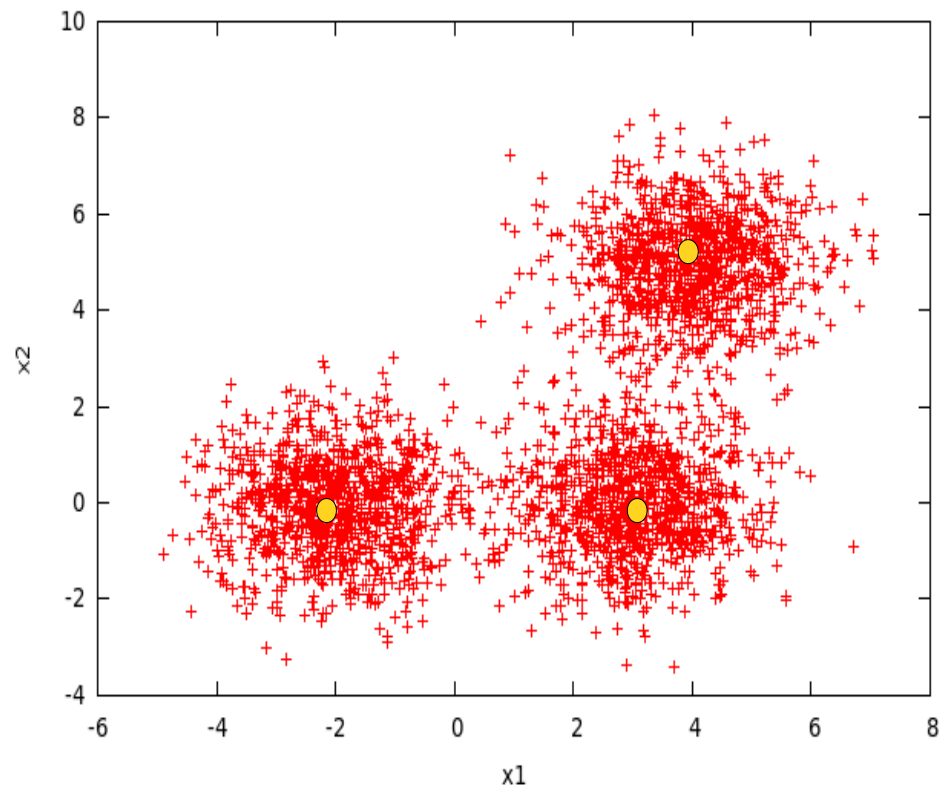
**Expectation  
Maximization**

**Model  
Selection**

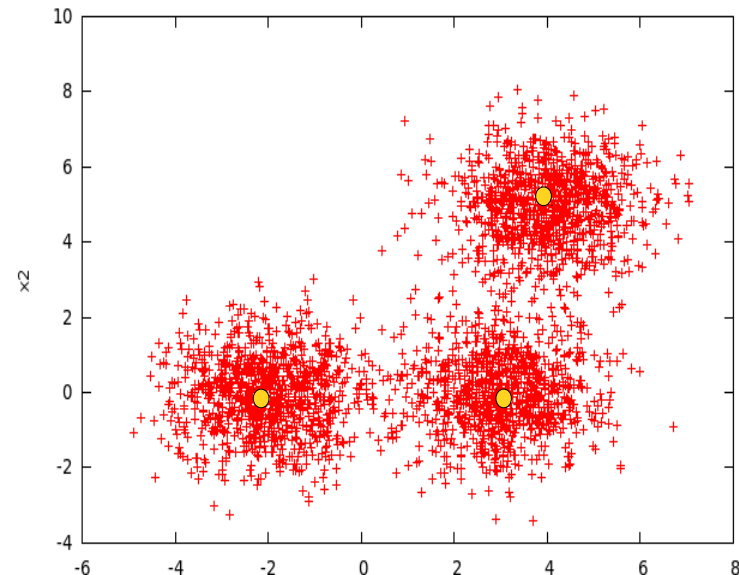
**Agglomeratives  
Clustering**

**Topic Models**

- **K-Means** ist ein sehr einfaches und gängiges Verfahren zur Cluster-Analyse
- **Gegeben:** Samples  $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathbb{R}^m$
- **Grundannahmen:**
  - Es existieren **K** Zentren  $\mu_1, \dots, \mu_K \in \mathbb{R}^m$  („**K means**“)
  - Jedes Sample  $\mathbf{x}_i$  ist einem Zentrum **k(i)** zugeordnet
  - Die Verteilung zu jedem Zentrum ist **sphärisch**



- Es liegt ein **Henne-Ei-Problem** vor
- Wüssten wir die **Cluster-Zuordnung**  $k(i)$  jedes Samples  $\mathbf{x}_i$ , könnten wir die Zentren ermitteln (z.B. mittels **ML-Schätzung**)
- Wüssten wir die **Zentren**, könnten wir die Cluster-Zuordnung  $k(i)$  ermitteln (wir wählen zu jedem Datenpunkt das **nächstgelegene Zentrum**)
- **Ansatz**: Wir **fixieren** jeweils einen Aspekt und **optimieren** den anderen!



Gegeben: Samples  $x_1, \dots, x_n$

1. Initialisiere  $\mu_1, \dots, \mu_K$  mit zufälligen Samples
2. Iteriere bis zur Konvergenz

(a) Für  $i = 1, \dots, n$ :

$$k(i) := \arg \min_k \|x_i - \mu_k\|_2$$

(b) Für  $k = 1, \dots, K$ :

$$X_k := \{x_i \mid k(i) = k\}$$

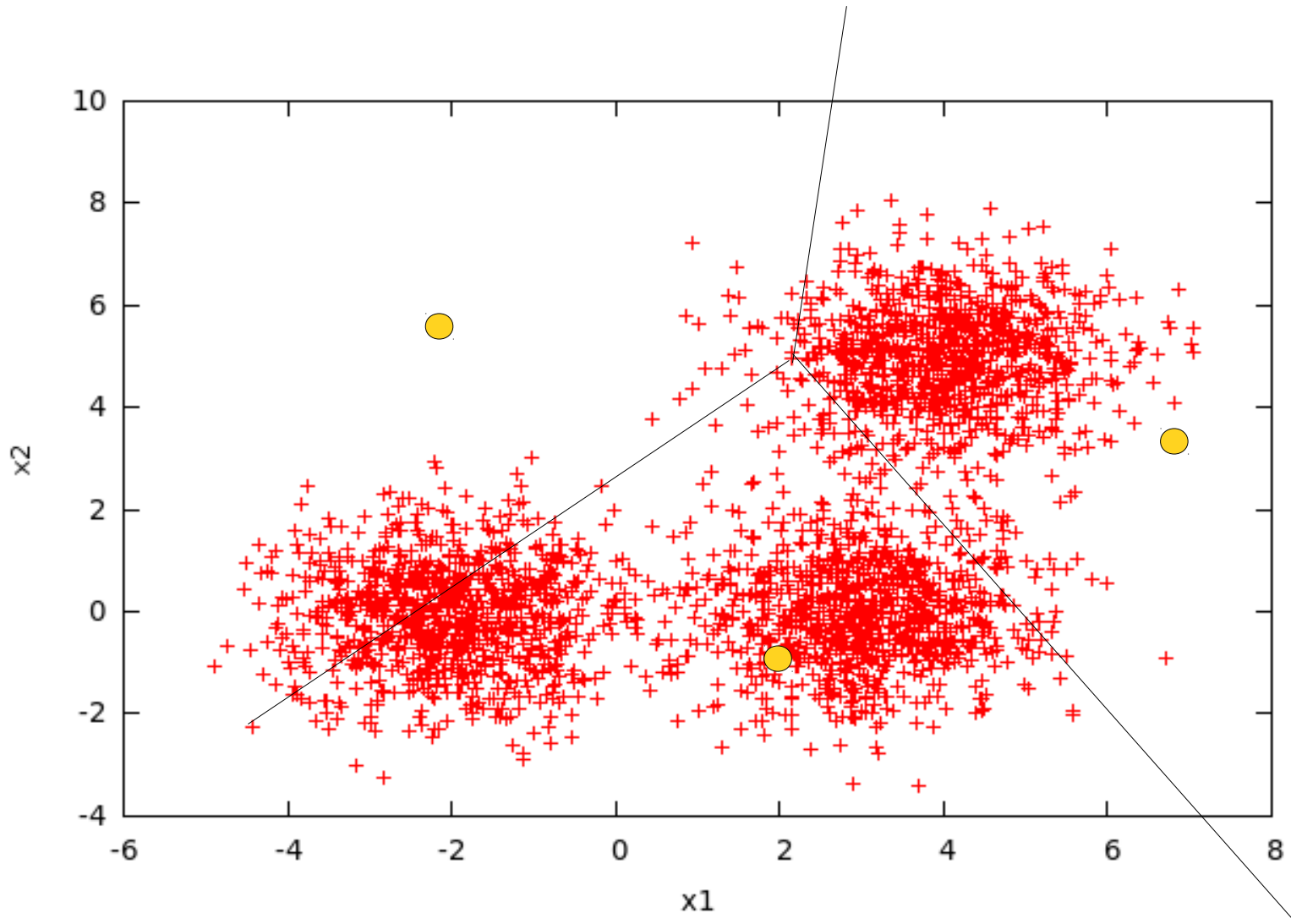
$$\mu_k := \frac{1}{|X_k|} \cdot \sum_{x \in X_k} x$$



# K-Means: Beispiel

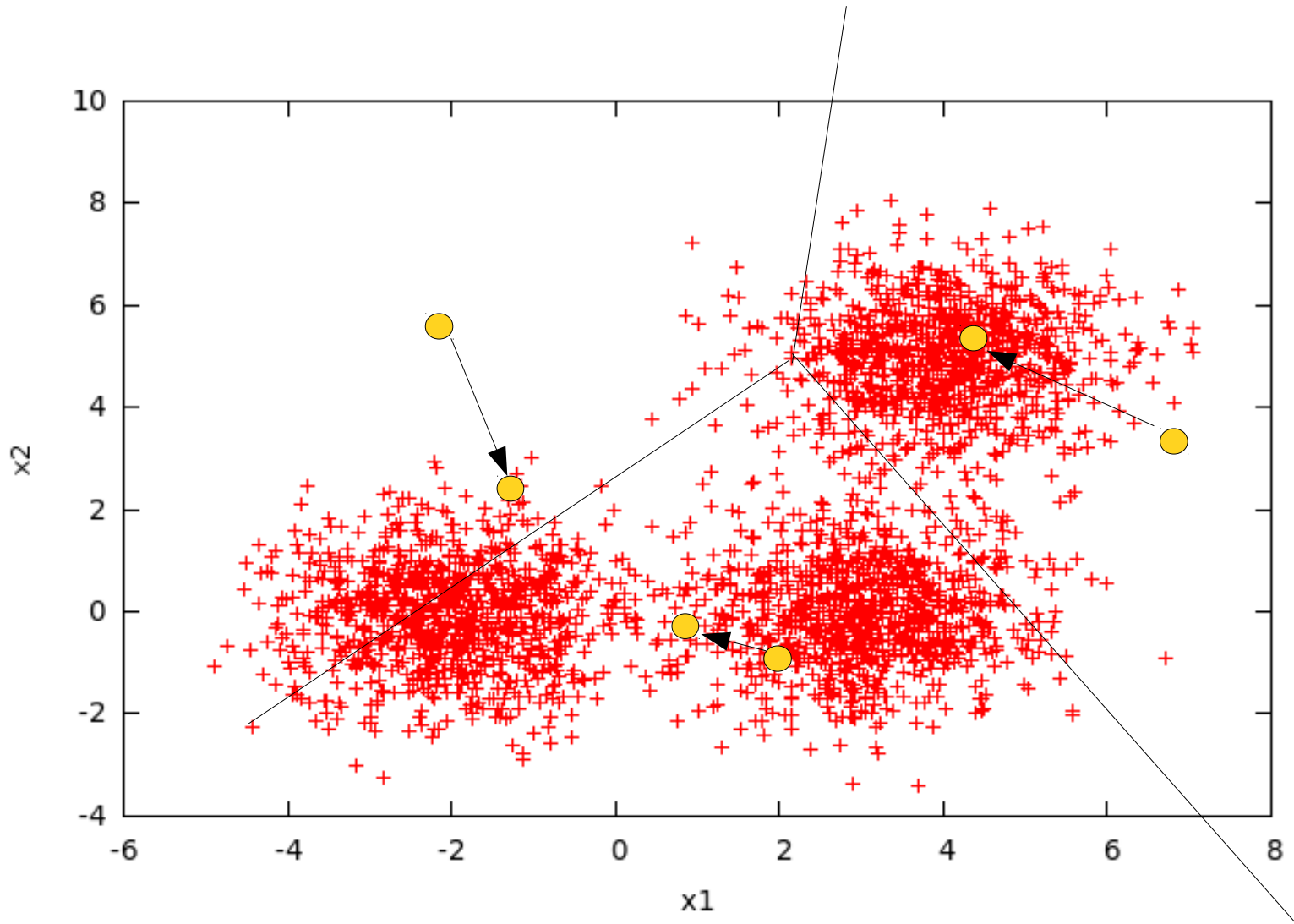
**Prof. Adrian Ulges**

Fachbereich DCSM / Informatik  
Hochschule RheinMain



# K-Means: Beispiel

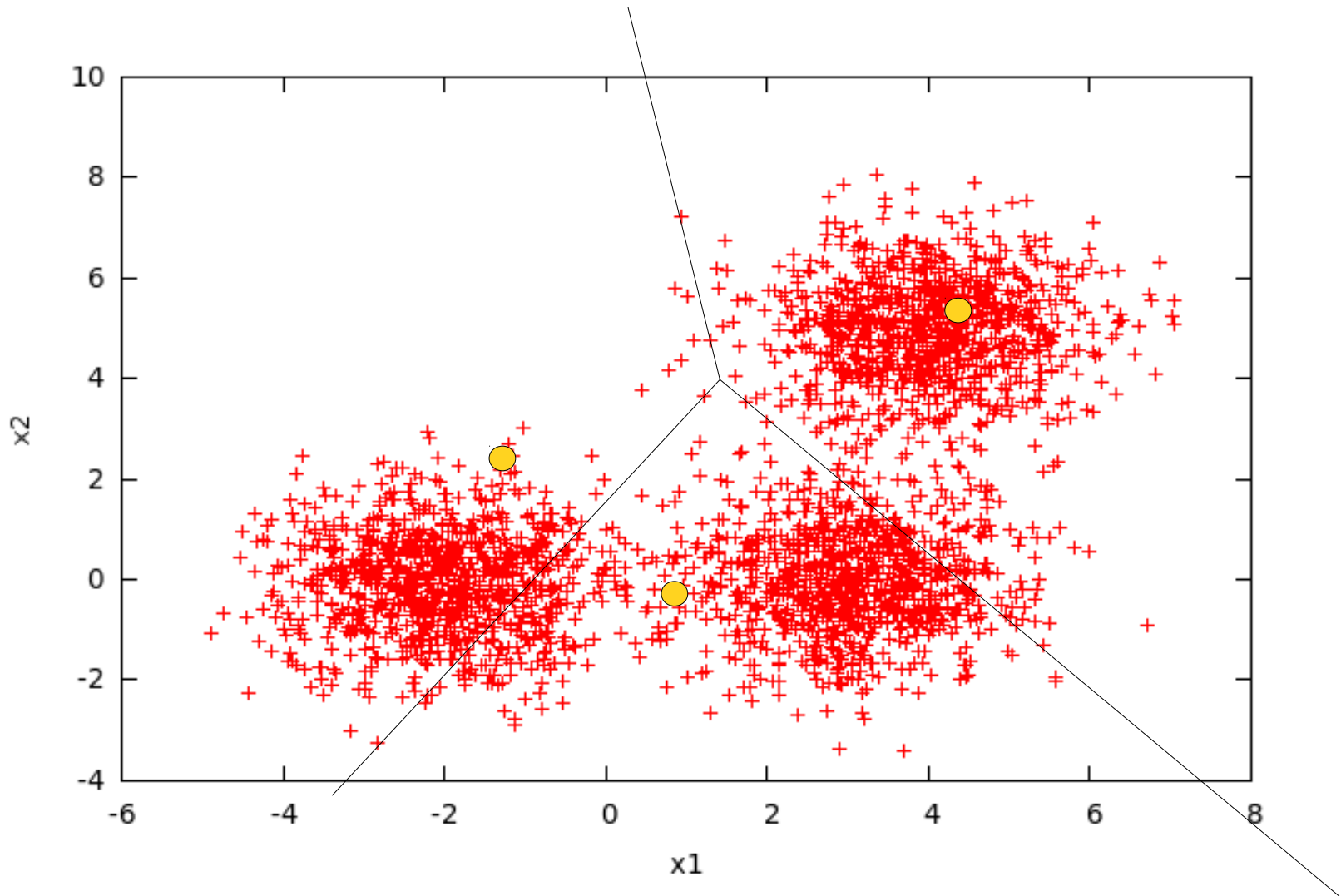
**Prof. Adrian Ulges**  
Fachbereich DCSM / Informatik  
Hochschule RheinMain



# K-Means: Beispiel

**Prof. Adrian Ulges**

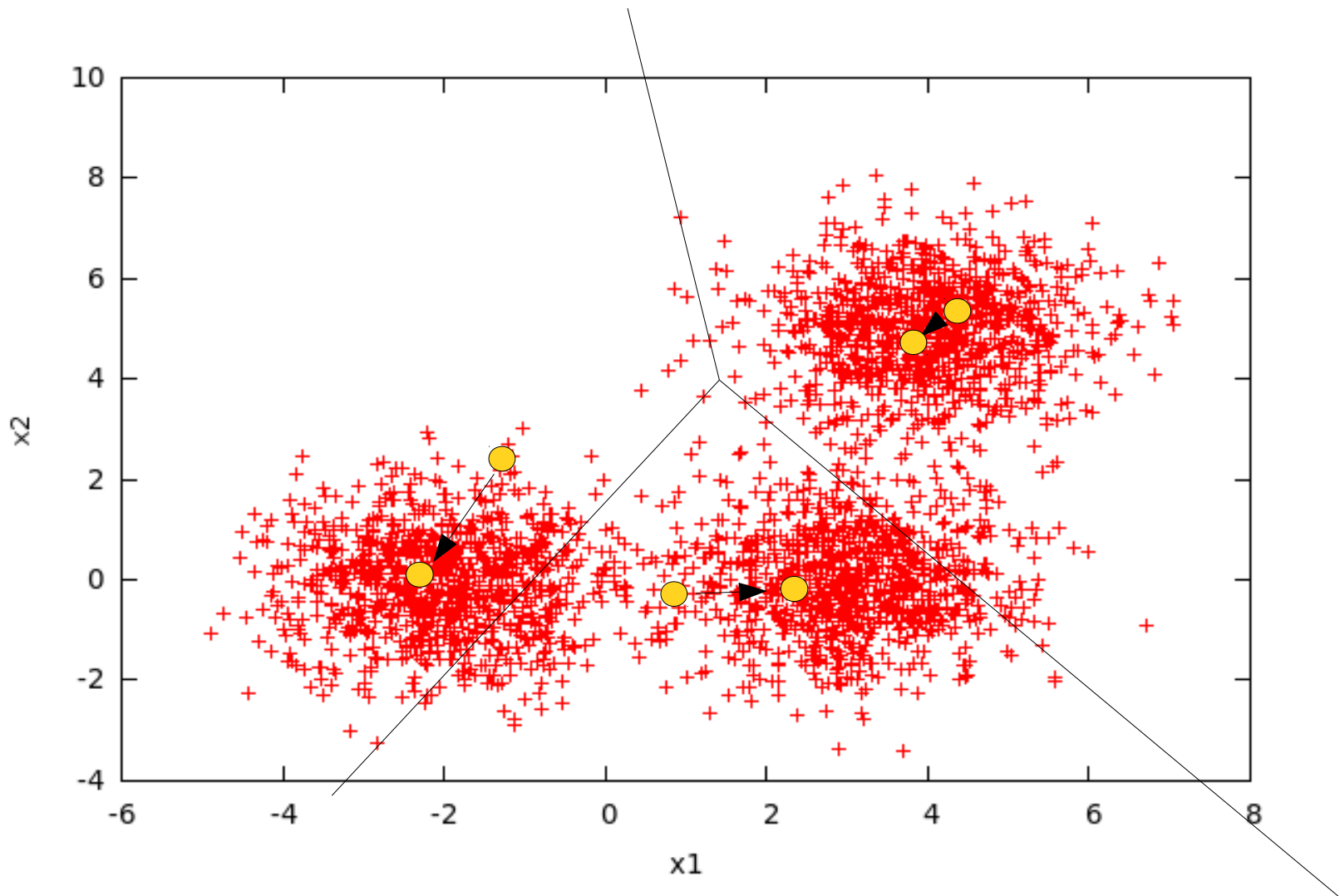
Fachbereich DCSM / Informatik  
Hochschule RheinMain



# K-Means: Beispiel

**Prof. Adrian Ulges**

Fachbereich DCSM / Informatik  
Hochschule RheinMain

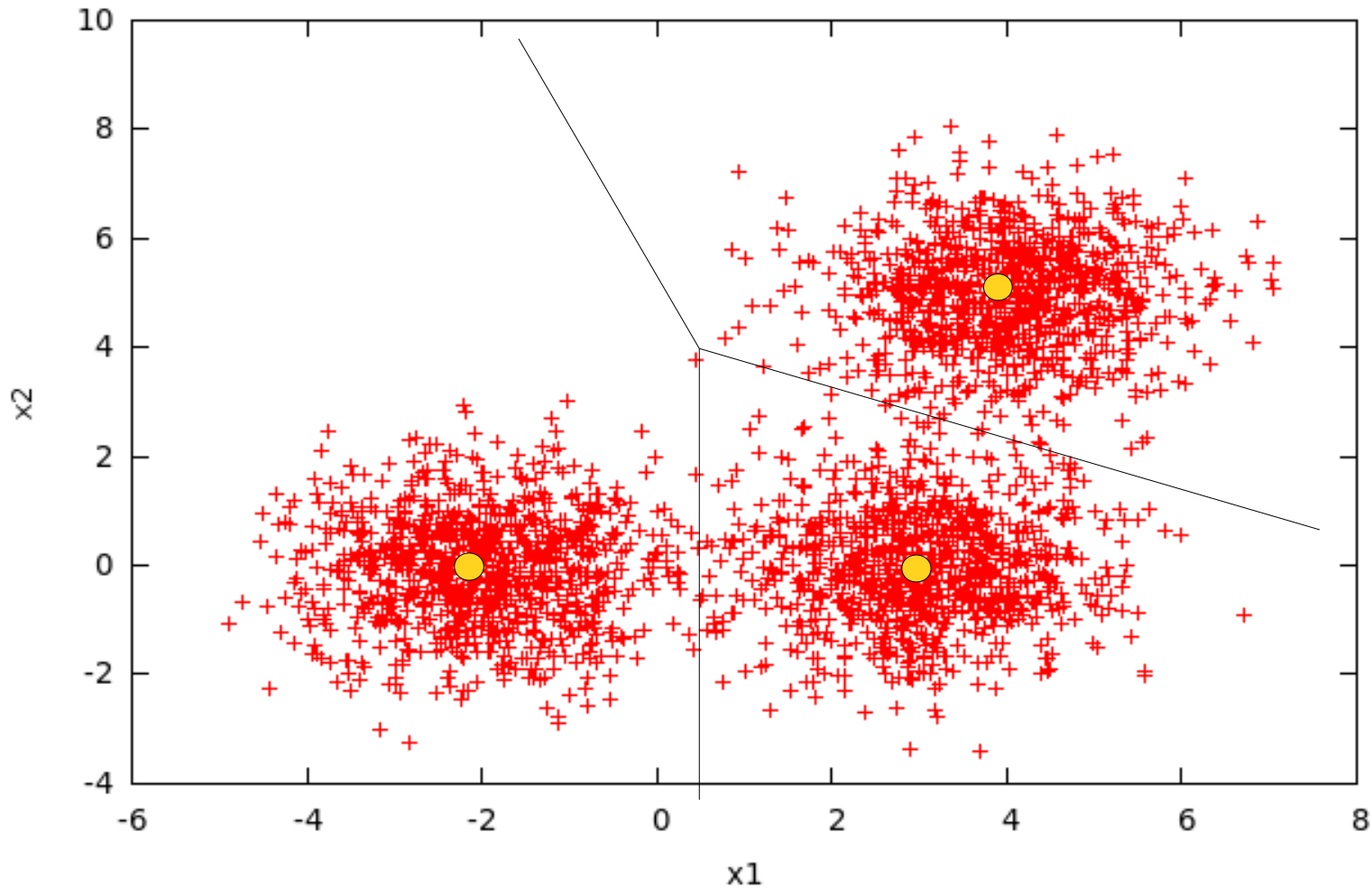


...

# K-Means: Beispiel

**Prof. Adrian Ulges**

Fachbereich DCSM / Informatik  
Hochschule RheinMain



- K-Means entspricht einer Minimierung des **quadratischen Fehlers  $E$**
- **Berechnungsaufwand?**
  - Pro Iteration:  **$O(k \cdot n \cdot m)$**
  - Die **Anzahl** der benötigten Iterationen ist üblicher Weise moderat
- **Konvergiert** K-Means immer?
  - Ja. **Begründung:** Die Folge der Fehlerwerte pro Iteration,  **$E_0, E_1, E_2, \dots$**  ist ...
    - (a) ... monoton fallend
    - (b) ... nach unten beschränkt ( $\geq 0$ )
  - Also ist die **Folge** (und somit das **Verfahren**) **konvergent**

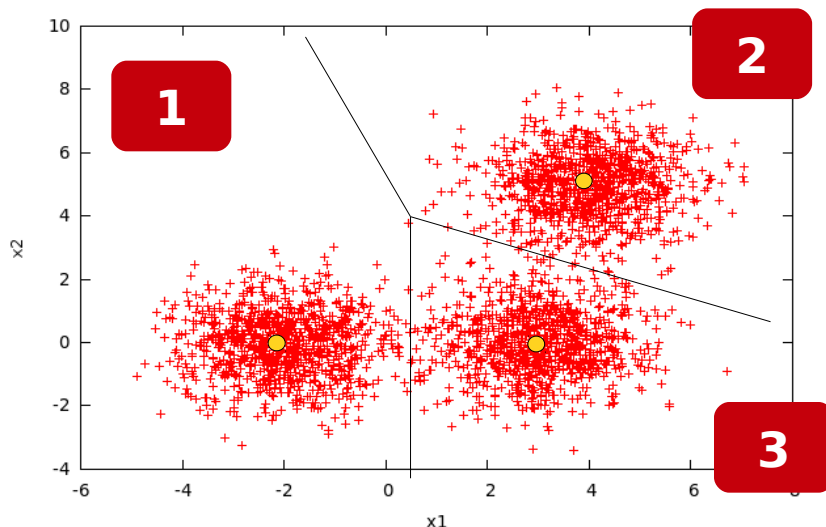
$$E = \sum_{i=1}^n \left( x_i - \mu_{k(i)} \right)^2$$

- Gibt K-Means immer **dasselbe Ergebnis**?
- Nein – K-Means ist ein lokales Suchverfahren (ähnlich dem **Gradientenabstieg**)
- **Problem 1:** Die Means können **vertauscht** sein:
  - $\mu_1=(0,0)$ ,  $\mu_2=(1,1)$ ,  $\mu_3=(5,3)$
  - $\mu_1=(5,3)$ ,  $\mu_2=(0,0)$ ,  $\mu_1=(1,1)$
- **Problem 2:** Die Mittelwerte können **komplett andere** sein (**lokales Minimum**)
- **Ansatz:** Starte iterativ neu, behalte das Ergebnis mit minimalem quadratischen Fehler **E**
- Außerdem können während des Algorithmus **leere Cluster** auftreten. **Ansatz:** initialisiere das zugehörige Zentrum zufällig neu und iteriere weiter



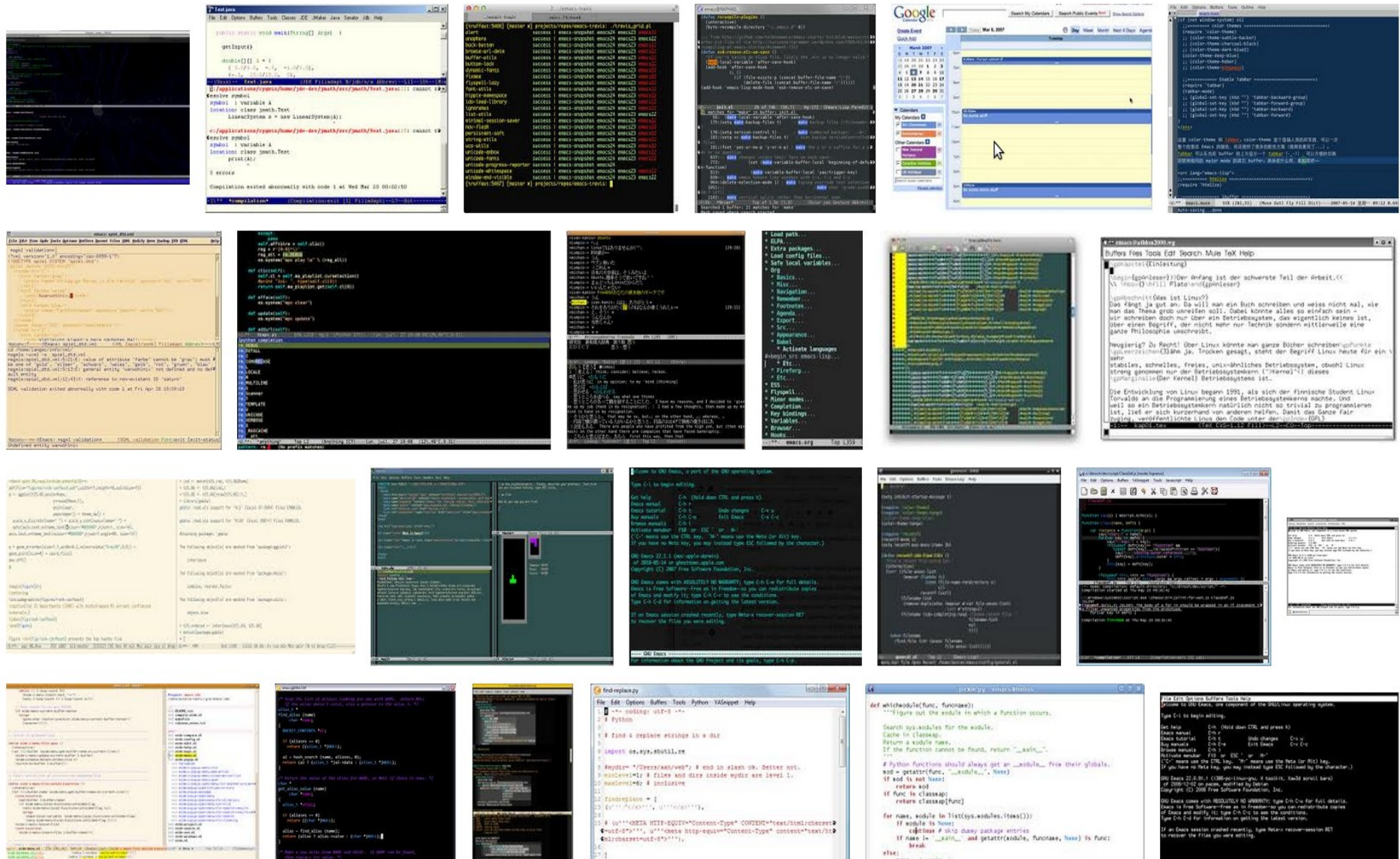
- **Nach der Clusteranalyse** können wir den Mittelwerten auch **neue Samples  $x$**  zuordnen, die nicht zur Eingabe der Clusteranalyse gehörten (**Vektor-Quantisierung**)

$$k(x) = \arg \min_k ||x - \mu_k||$$



# K-Means: Code-Beispiel

Prof. Adrian Ulges  
Fachbereich DCSM / Informatik  
Hochschule RheinMain



**Grundlagen**

**K-Means**

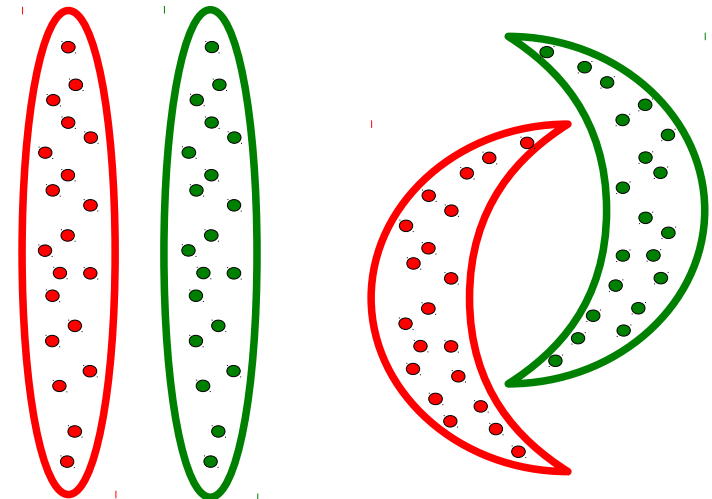
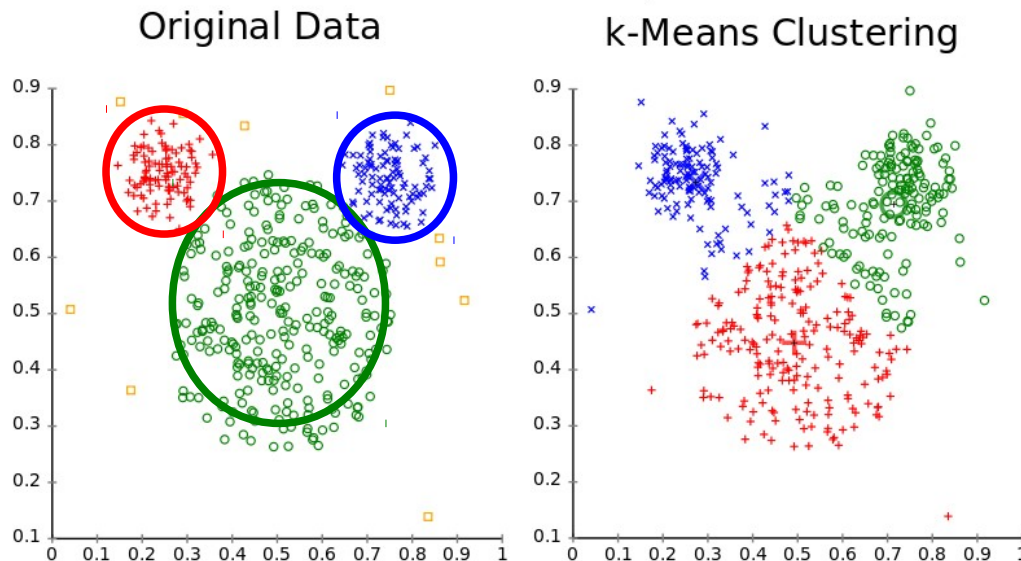
**Expectation  
Maximization**

**Model  
Selection**

**Agglomeratives  
Clustering**

**Topic Models**

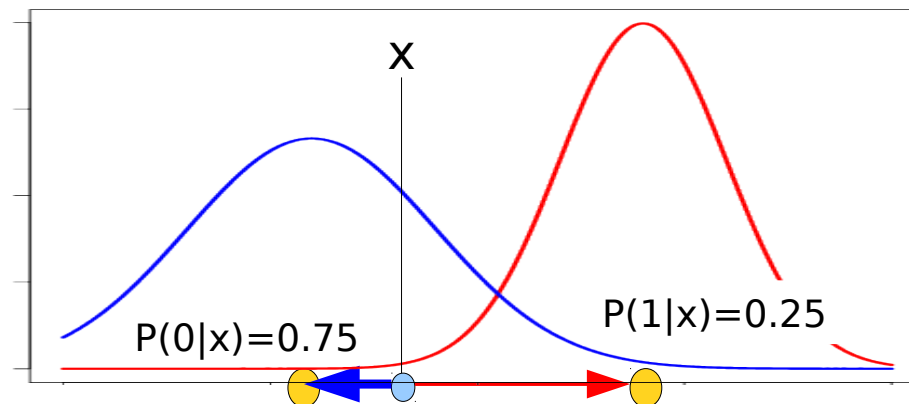
- **Einschränkungen:** Welche Daten können wir mit K-Means nicht richtig clustern?
  - Cluster mit unterschiedlicher **Varianz**
  - Cluster mit **anisotroper** Verteilung
  - **Nicht normalverteilte** Cluster



# Von K-Means zu Expectation Maximization

Prof. Adrian Ulges  
Fachbereich DCSM / Informatik  
Hochschule RheinMain

- Wir können einige der vorgenannten Schwächen durch eine **Verallgemeinerung des K-Means**-Ansatzes beheben
- Wie bei K-Means **alternieren wir zwei Schritte**:
  - Zuweisung von Samples zu Clustern („**E-Schritt**“)
  - Parameterschätzung der Cluster („**M-Schritt**“)
- Der resultierende Algorithmus heißt **Expectation Maximization**
- **Unterschiede zu K-Means**:
  - „**E-Schritt**“: Keine „harte“ Zuweisung von Samples zu Zentren, sondern Schätzung einer **Wahrscheinlichkeit**  $P(k|x_i)$
  - „**M-Schritt**“: Neben dem Zentrum wird auch die Varianz / Kovarianz geschätzt



## „E-Schritt“

$$P(k|x_i) = \frac{\mathcal{N}(x_i; \mu_k, \sigma_k)}{\sum_{k'} \mathcal{N}(x_i; \mu_{k'}, \sigma_{k'})}$$

EM

$$k(i) = \arg \min_k \|x_i - \mu_k\|^2$$

K-Means

## „M-Schritt“

$$\mu_k := \frac{\sum_i P(k|x_i) \cdot x_i}{\sum_i P(k|x_i)}$$

$$\sigma_k^2 := \frac{\sum_i P(k|x_i) \cdot (x_i - \mu_k)^2}{\sum_i P(k|x_i)}$$

EM

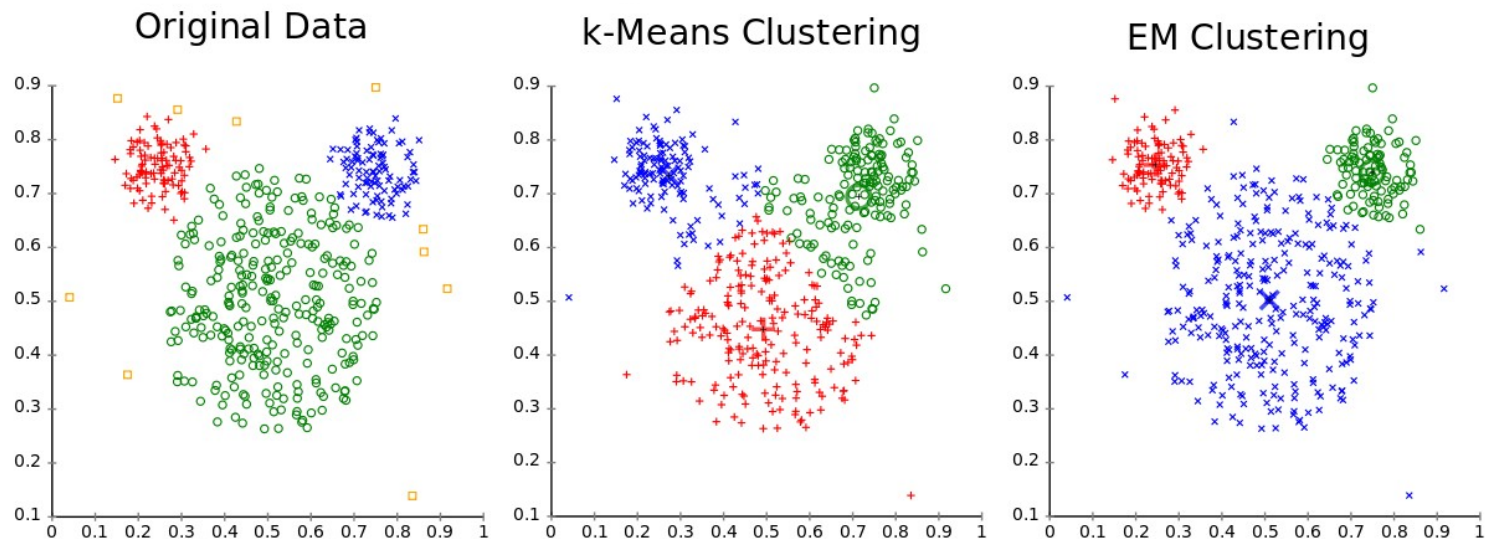
$$\mu_k^{t+1} := \frac{\sum_i \mathbf{1}_{k(i)=k} \cdot x_i}{\sum_i \mathbf{1}_{k(i)=k}}$$

—

K-Means



- Expectation Maximization kann Normalverteilungen mit **beliebigen Varianzen** lernen
- **Konvergenz** ist garantiert!
- Aber: EM ist immer noch ein **lokales Suchverfahren**
- Das heisst: Verschiedenen Neustarts können immer noch zu unterschiedlichen (potenziell suboptimalen) Ergebnissen führen.



**Grundlagen**

**K-Means**

**Expectation  
Maximization**

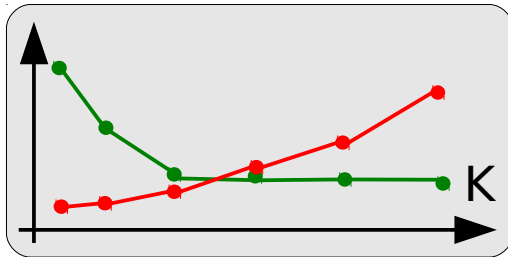
**Model  
Selection**

**Agglomeratives  
Clustering**

**Topic Models**



- **Offene Frage: Anzahl der Cluster  $K$ ?**
  - **$K$  zu klein**: Höherer Fehler, Untersegmentierung
  - **$K$  zu groß**: Komplexes Modell, instabile Optimierung, Übersegmentierung
- **Ansatz: Bayes'sches Informationskriterium**

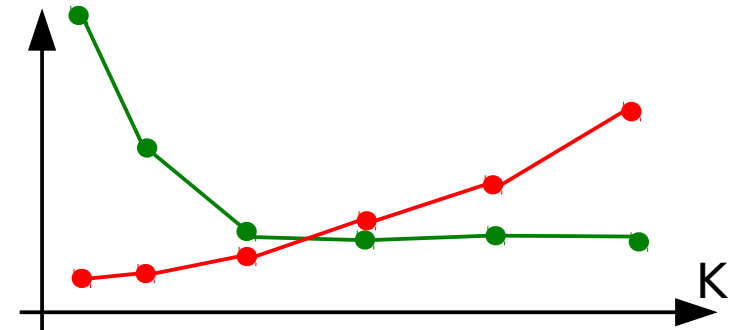
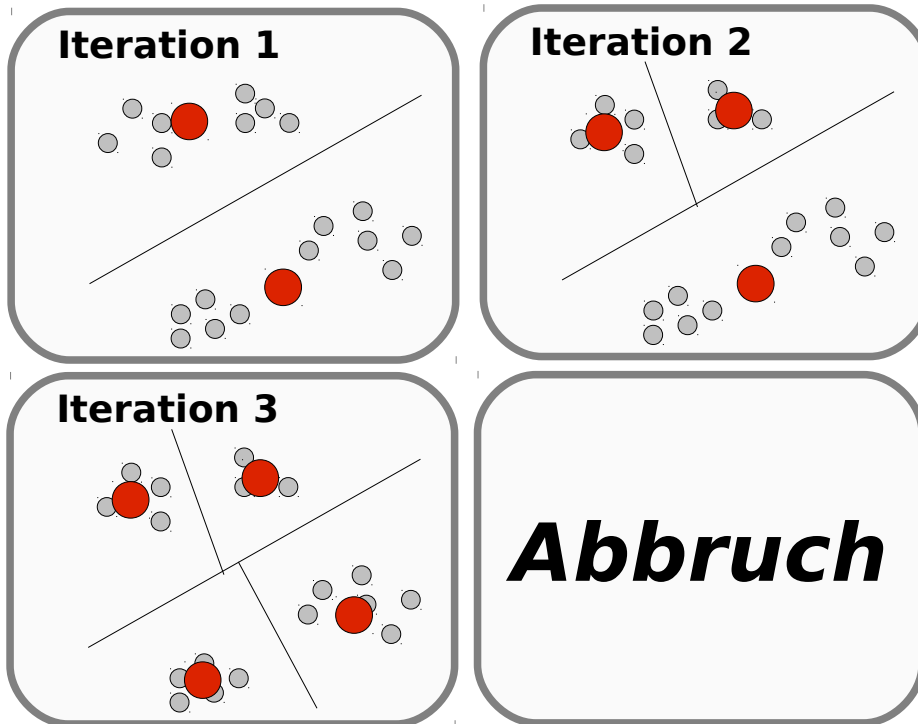


$$K^* = \arg \min_K \underbrace{\sum_{i=1}^n \left( x_i - \mu_{k(i)} \right)^2}_{E(k)} + \underbrace{m \cdot K \cdot \log(n)}_{\text{"Modell-Komplexität"}}$$

Dimension der Daten  $\nearrow$   $m$

$\nearrow$  #Samples  $n$

- **Hierarchischer Algorithmus: Wiederhole ...**
  - Füge ein neues Zentrum hinzu (z.B. durch Split des größten Clusters)
  - Wende K-Means rekursiv auf den einzelnen Clustern an
- **Breche ab**, wenn die Güte sich nicht mehr verbessert

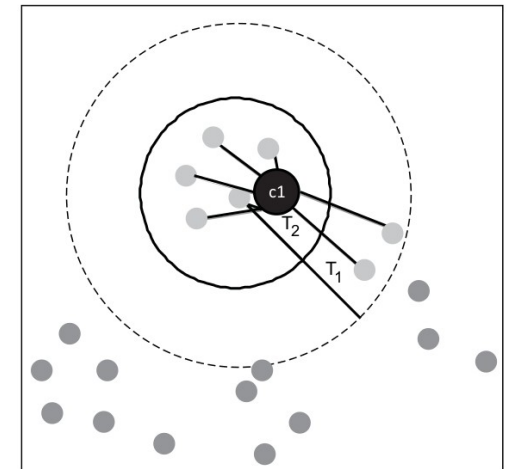


## • Ansatz 3: **Canopy Clustering**

- Eine **Greedy-Strategie**, um auf großen Datenmengen (potenziell suboptimale) Cluster zu finden
- **Anwendung**: Schätzer für **K** und Initialisierung für K-Means
- Die entstehenden Cluster können überlappen!
- Zwei **Schwellwerte**:  $T_1$  (bestimmt die Anzahl der Cluster),  $T_2$  (bestimmt die Überlappung)
- Oft einfaches Distanzmaß  $d(\cdot)$  als Approximation (Aufwand  $O(n^2)$ )

Gegeben: Menge von Samples  $X$

- (1)  $C \leftarrow \{\}$  // Menge von Zentren/„canopies“
- (2) Wähle zufällig ein Sample  $x \in X$
- (3)  $Y \leftarrow \{y \mid d(y-x) \leq T_1\}$  // „sehr ähnliche“ Samples
- (4)  $Z \leftarrow \{z \mid T_1 < d(z-x) \leq T_2\}$  // „ähnliche“ Samples
- (5)  $C \leftarrow C \cup \{x\}$
- (6)  $X \leftarrow X \setminus Y$
- (7) Falls  $X \neq \{\}$ : gehe zu (2)
- (8) return  $C$



**Grundlagen**

**K-Means**

**Expectation  
Maximization**

**Model  
Selection**

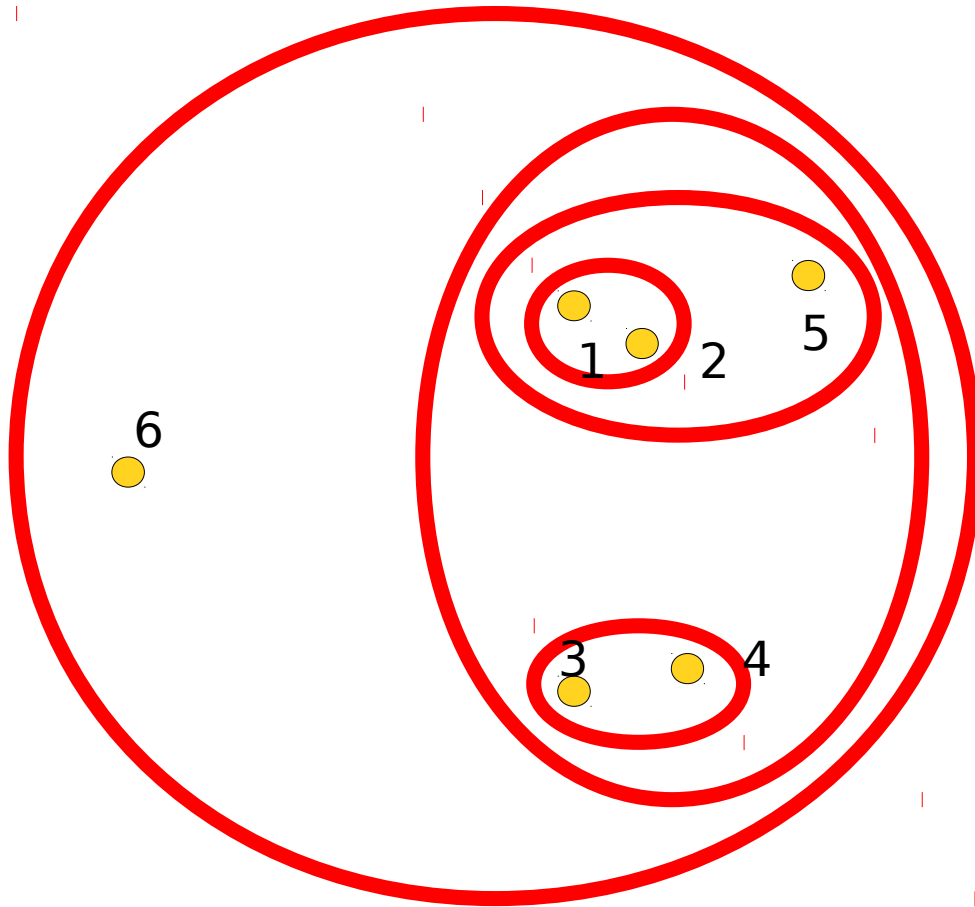
**Agglomeratives  
Clustering**

**Topic Models**

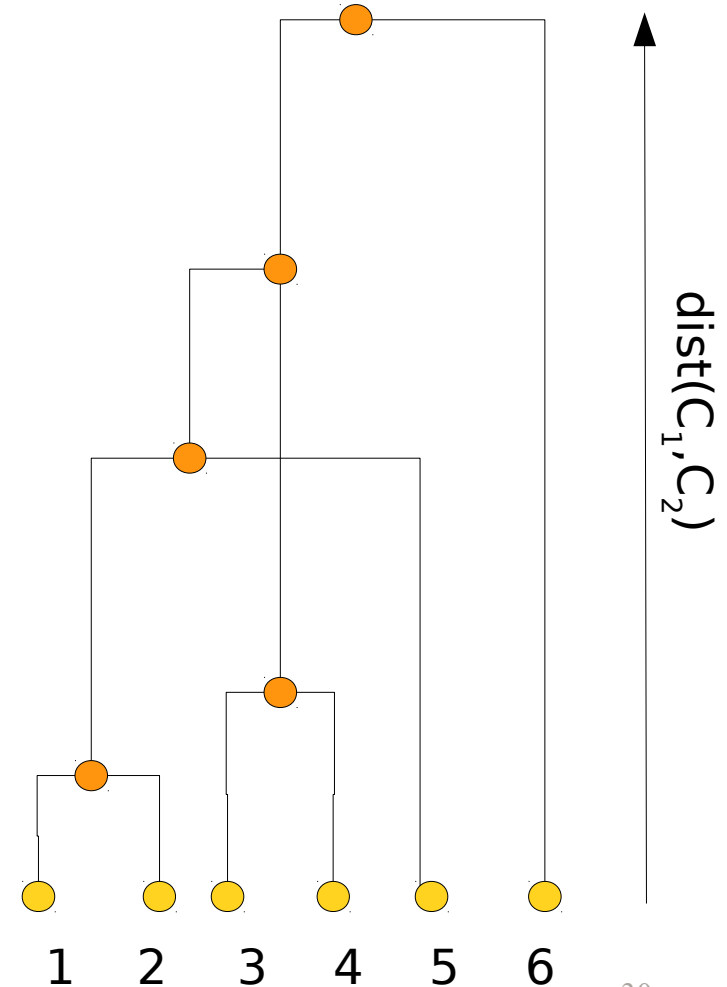
- Wir bezeichnen K-Means/EM als **divisive Clustering-Verfahren**, weil sie (**top-down**) die Datenmenge **unterteilen**.
- Alternative: **Agglomeratives Clustering**
  - **Startzustand**: Jedes Sample befindet sich in seinem „eigenen“ Cluster („**singletons**“)
  - **Iterativ**: **Mische** die beiden „ähnlichsten“ Cluster zu einem neuen Cluster
- Das Ergebnis kann mittels eines sogenannten **Dendogramms** in Baumform illustriert werden.

# Agglomeratives Clustering: Beispiel

Prof. Adrian Ulges  
Fachbereich DCSM / Informatik  
Hochschule RheinMain



## Dendrogramm



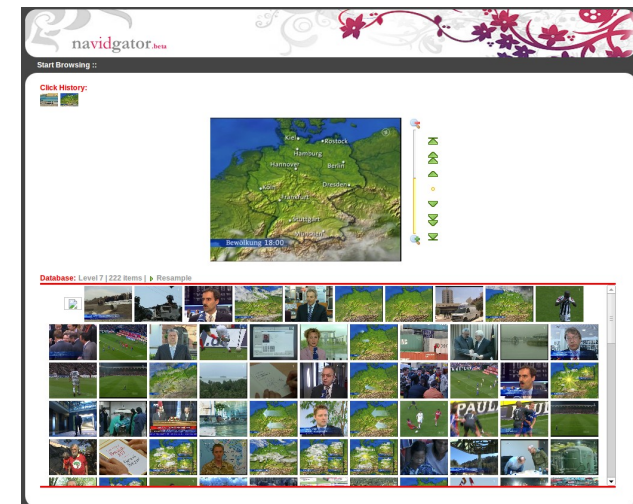
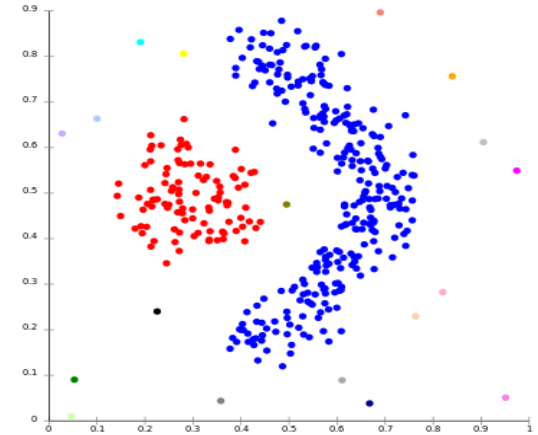
- Zwei offene **Fragen**:
  1. Distanzmaß
  2. Abbruchbedingung
- **Abbruchbedingung**: Heuristiken  
(siehe „Model Selection“ / K-Means)
- **Distanzmaß**: Drei gängige Alternativen  
(seien  $X, Y$  Cluster)

<b>Single Linkage:</b>	$dist(X, Y) = \min_{x \in X, y \in Y} \ x - y\ _2$
<b>Complete Linkage:</b>	$dist(X, Y) = \max_{x \in X, y \in Y} \ x - y\ _2$
<b>Average Linkage:</b>	$dist(X, Y) = \frac{1}{ X  \cdot  Y } \sum_{x \in X, y \in Y} \ x - y\ _2$

# Agglomeratives Clustering: Diskussion

Prof. Adrian Ulges  
Fachbereich DCSM / Informatik  
Hochschule RheinMain

- **Nachteil: Berechnungsaufwand**
  - Aufbau Ähnlichkeitsmatrix:  $O(n^2)$
  - Anzahl der Fusion-Schritte:  $O(n)$
  - **Pro Fusion-Schritt**: Neuberechnung der Ähnlichkeit des entstehenden Clusters zu den anderen Clustern:  $O(n)$  (oder noch teurer)
    - Insgesamt:  $O(n^2)$  :-)
  - Häufig grobes **Pre-Clustering** und Anwendung agglomerativer Verfahren in den einzelnen Clustern
- **Vorteile**
  - Auch geeignet für nicht-normalverteilte Cluster





**Grundlagen**

**K-Means**

**Expectation  
Maximization**

**Model  
Selection**

**Agglomeratives  
Clustering**

**Topic Models**

- Ein weiteres Verfahren zum Clustering von Text heißt **PLSA** (*Probabilistic Latent Semantic Analysis*).
- Wir modellieren Dokumente als „**bag-of-words**“
- EM liefert zwei interessante Resultate:
  - **Cluster** von Dokumenten
  - Die Cluster-Zentren entsprechen prototypischen Schlagwort-Verteilungen (oder „**topics**“), in denen jeweils bestimmte Terme besonders häufig vorkommen.
- Wir nennen Modelle dieser Art „**topic models**“.

# EM Anwendung: Text Clustering

Prof. Adrian Ulges  
Fachbereich DCSM / Informatik  
Hochschule RheinMain

The collage displays five Wikipedia article snippets, each demonstrating a different concept related to text clustering or information retrieval:

- Homonym**: Explains that in linguistics, a homonym is a word that has the same pronunciation but different meanings. It provides examples like "bank" (financial institution vs. riverbank) and "bat" (mammal vs. insect).
- Levenshtein distance**: Describes a metric for measuring the difference between two sequences of characters, often used in spell checking and natural language processing. It is named after the Russian mathematician Vladimir Levenshtein.
- Information retrieval**: Discusses the science of searching for information, particularly in the context of databases and the Internet. It mentions various search engines and the challenges of finding relevant information.
- AC/DC**: A music band article. It mentions that the band is an Australian rock band formed in 1973 by brothers Malcolm and Angus Young. It also notes their classification as hard rock and their consistent use of the name AC/DC.
- Fleetwood Mac**: A music band article. It states that the band is a British-American rock band formed in 1967 in London. The sidebar provides detailed background information, including the band's origin (London, United Kingdom), genres (Rock, pop rock, blues-rock), years active (1967-present), labels (Blue Horizon, Reprise, Warner Bros., Sire, CBS, EMI, Epic, Sanctuary), associated acts (John Mayall's Bluesbreakers, Eddie Boyd, Chicken Shack, Buckingham Nicks), and website (www.fleetwoodmac.com).

# EM Anwendung: Text Clustering

Prof. Adrian Ulges  
Fachbereich DCSM / Informatik  
Hochschule RheinMain



index  
information  
search  
retrieval

Levenshtein distance

From Wikipedia, the free encyclopedia  
(Redirected from Levenshtein Distance)

This article needs additional citations for verification.  
Please help improve this article by adding reliable references. Unsourced material may be challenged and removed.  
February 2010

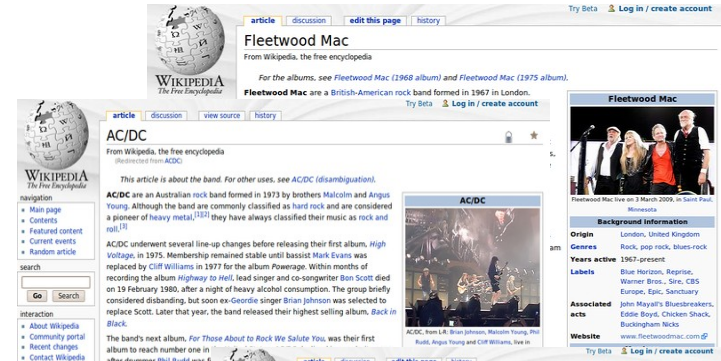
In information theory and computer science, the **Levenshtein distance** is a metric for measuring the amount of difference between two sequences (i.e., an **edit distance**). The term **edit distance** is often used to refer specifically to Levenshtein distance.

The Levenshtein distance between two strings is defined as the minimum number of edits needed to transform one string into the other, with the allowable edit operations being insertion, deletion, or substitution of a single character. It is named after Vladimir Levenshtein, who considered this distance in 1965.<sup>[1]</sup>

**Contents** [hide]

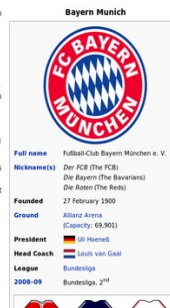
- 1 Example
- 2 Applications
- 3 Relationship with other edit distance metrics
- 4 Computing Levenshtein distance
- 4.1 Proof of correctness
- 4.2 Possible improvements
- 4.3 Upper and lower bounds
- 5 See also
- 6 Notes
- 7 External links

**tein distance between "kitten" and "sitting" is 3, since the following three edits change one into the**



rock concert  
band  
album

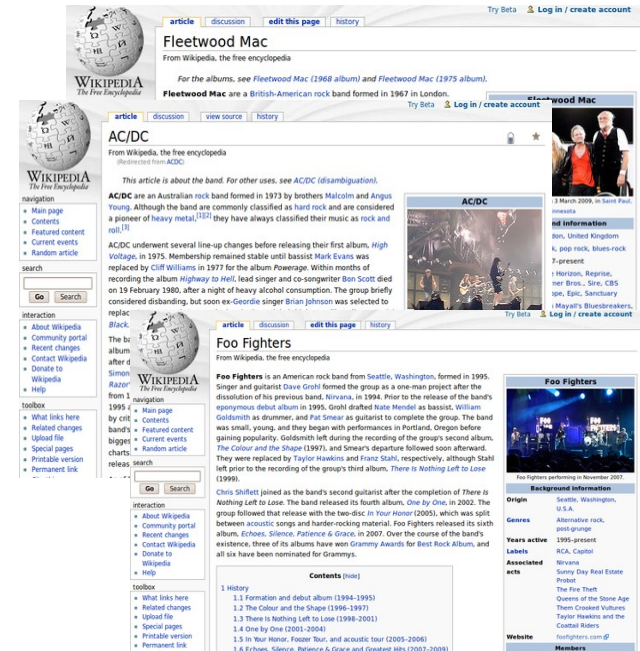
team  
soccer  
Bundesliga  
champion



# Das PLSA-Modell

Prof. Adrian Ulges  
Fachbereich DCSM / Informatik  
Hochschule RheinMain

- Eingabe: **Dokumente**  $d_1, \dots, d_M$
- Jedes Dokument  $d$  wird repräsentiert durch sein **Bag-of-Words** Feature:  
Für jedes Wort  $w$  kennen wir die Wahrscheinlichkeit  $P(w|d)$ .
- Diese Wahrscheinlichkeiten speichern wir in einem **Vektor** (siehe IR).
- **PLSA** nimmt außerdem an, dass in der Dokumentsammlung  $K$  **Themen (topics)**  $z_1, \dots, z_k$  auftreten.
- Jedes Topic hat wieder eine **Wortverteilung**  $P(w|z)$ .
- Die Topics sollen mittels unüberwachten Lernens **automatisch ermittelt** werden!



rock  
concert  
band  
album

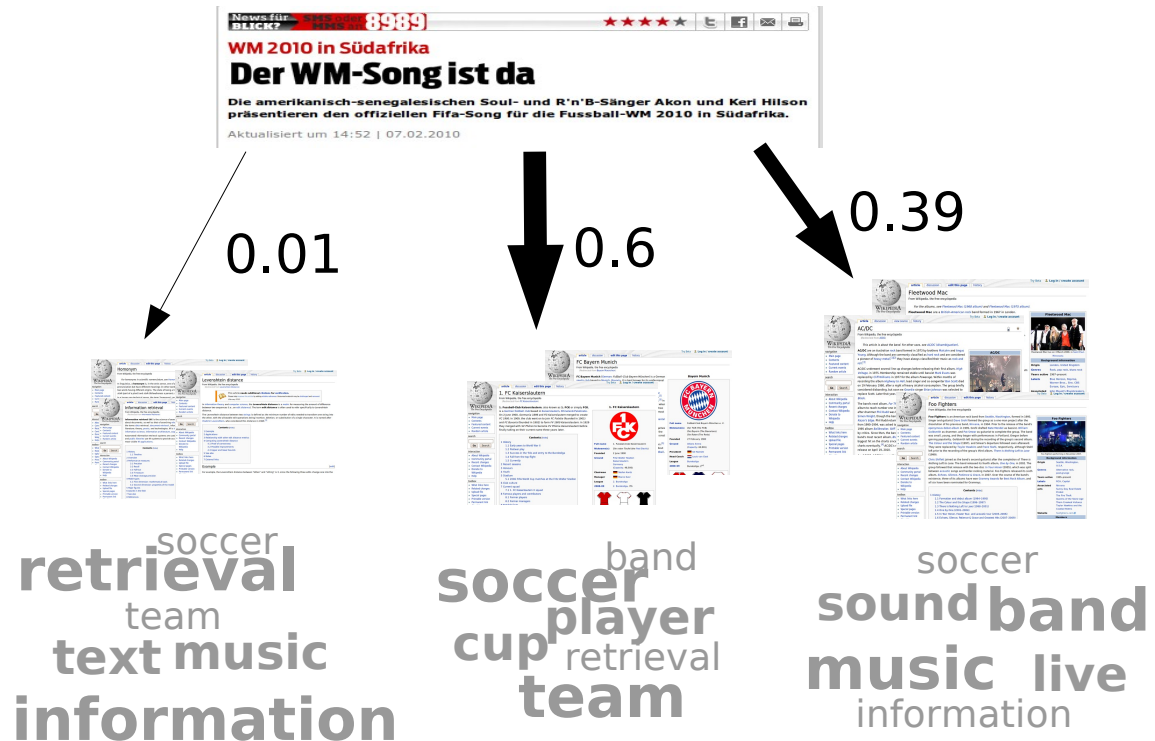


- **Annahme 1:** Ein Dokument ist eine **Mischung aus Topics**.
- **Annahme 2:** Wörter des Dokumentes werden generiert, indem wir (a) ein Topic wählen, und (b) ein zufälliges Wort des gewählten Topics wählen.

1. Wähle topic:  
 $P(z|d)$

topics

2. Wähle Wort:  
 $P(w|z)$



- **Ziel des PLSA-Clusterings:** Ermittle...
  - Die **Topics**:  $P(w|z)$
  - Die **Verteilung der Topics** auf Dokumente:  $P(z|d)$
- **Erinnerung (EM):** Zwei abwechselnde Schritte
  1. Ordne jedem Trainingssample ein Clustern zu.
  2. Adaptiere Cluster-Parameter (Form der Cluster).
- **EM für PLSA: Dieselben Schritte.**
  1. Ordne jedem Wort in jedem Dokument ein Topic zu  
→  $P(z|w,d)$ .
  2. Adaptiere Topics und Topicverteilung  
→  $P(w|z), P(z|d)$ .

given: documents  $d_1, \dots, d_M$ , words  $w_1, \dots, w_N$ .

given:  $P(w|d)$ , number of topics  $K$ .

1. initialize  $P(w|z)$ ,  $P(z|d)$  randomly

2. until convergence:

(a) fit word-topic correspondences:

for  $m = 1, \dots, M$ ,  $n = 1, \dots, N$ ,  $k = 1, \dots, K$ :

$$P(z_k | w_n, d_m) = \frac{P(w_n | z_k) \cdot P(z_k | d_m)}{\sum_{k'} P(w_n | z_{k'}) \cdot P(z_{k'} | d_m)}$$

(b) re-estimate parameters:

for  $m = 1, \dots, M$ ,  $n = 1, \dots, N$ ,  $k = 1, \dots, K$ :

$$P(z_k | d_m) = \frac{\sum_n n(w_n, d_m) \cdot P(z_k | w_n, d_m)}{n(d_m)}$$

$n(w_n, d_m)$  Anzahl der Vor-  
kommen von  $w_n$  in  $d_m$ .

$$P(w_n | z_k) = \frac{\sum_m n(w_n, d_m) \cdot P(z_k | w_n, d_m)}{\sum_{n'} \sum_m n(w_{n'}, d_m) \cdot P(z_k | w_{n'}, d_m)}$$



- PLSA ist ein EM-Algorithmus, also ein **lokales Suchverfahren**. Wir erhalten (wieder) unterschiedliche Resultate bei unterschiedlichen Startwerten.
- PLSA löst **Homonyme** auf.
  - „bush“ [Garten] vs. „bush“ [Politik]
- PLSA „**komprimiert**“ Dokumente:
  - Vorher:  $\mathbf{P}(\mathbf{w}|\mathbf{d})$ , hochdimensional
  - Hinterher:  $\mathbf{P}(\mathbf{z}|\mathbf{d})$ , niedrigdimensional



$$\mathbf{P}(\mathbf{z}|\mathbf{d}) = ( 0.01, \quad 0.6, \quad 0.39 )$$