Machine Learning

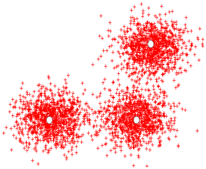
– winter term 2016/17 –

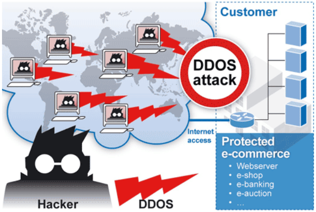
Chapter 05: Clustering

Prof. Adrian Ulges
Masters "Computer Science"
DCSM Department
University of Applied Sciences RheinMain

Unsupervised Learning = Learning without Labels images from [2], [1]

- **Clustering**: discover coherent groups of samples
- **Dimensionality reduction**: compressing samples
- **Itemset mining**: finding frequent substructures in the data
- **Anomaly detection**: detecting outliers in the data

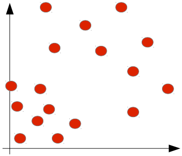


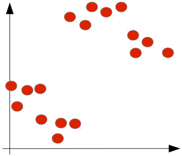


Outline

Clustering: Definition

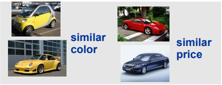
- Clustering (or *cluster analysis*) is an unsupervised learning problem *(remember:* **samples only**, *no labels)*
- The challenge is to discover coherent subgroups (or *clusters*) of samples
- **Difference to classification**: In clustering, we try to *find* the classes and assign samples to them





Challenges

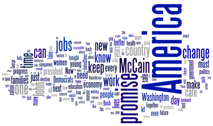
1. Often, it is unclear by which criterion to cluster *(example: cluster users, but by which demographic attributes?)*
2. Cluster granularity is unclear a priori

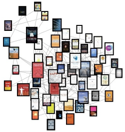


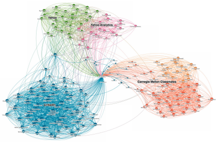
Clustering: Applications images from [4], [3]

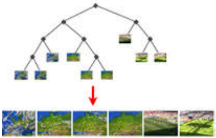
Clustering has numerours **applications** in various areas

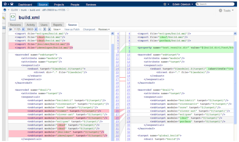
- market research
- life sciences
- information retrieval
- computer vision
- social networks
- data mining

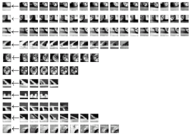


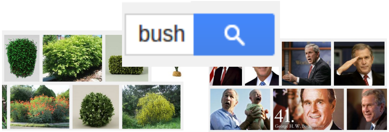




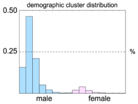




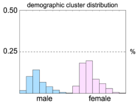




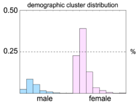
Example: Demographic Clustering on YouTube [8]



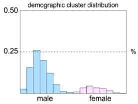
counterstrike, skateboarding, worldofwarcraft, darth-vader, simpsons, soccer



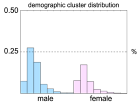
singing, cake, cooking, choir, food, baby, kitchen, cats, dancing, dogs



horse, anime, cheerleading, kiss, gymnastics, cake, riding, dancing, videoblog



obama, mccain, georgewbush, court, interview, press-conference, airplane-flying, riot



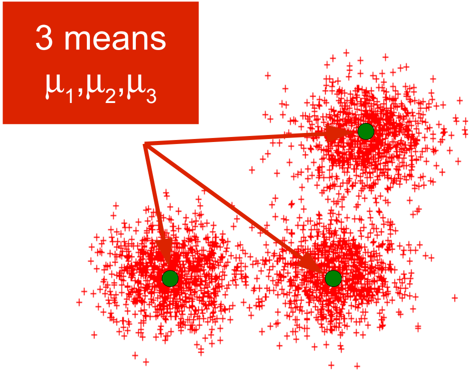
americas-got-talent, cats, cartoon, origami, piano, muppets, commercial, tornado

Outline

Clustering: K-Means

We start with the "first choice" clustering algorithm: **K-Means**

- Given: samples $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$
- We assume that samples are clustered around K centers *(the "K means")* $\mu_1, \ldots, \mu_K \in \mathbb{R}^d$
- Each sample $\mathbf{x}_i$ belongs to a mean $k(i)$
- The clusters are **spheres** of **identical size**



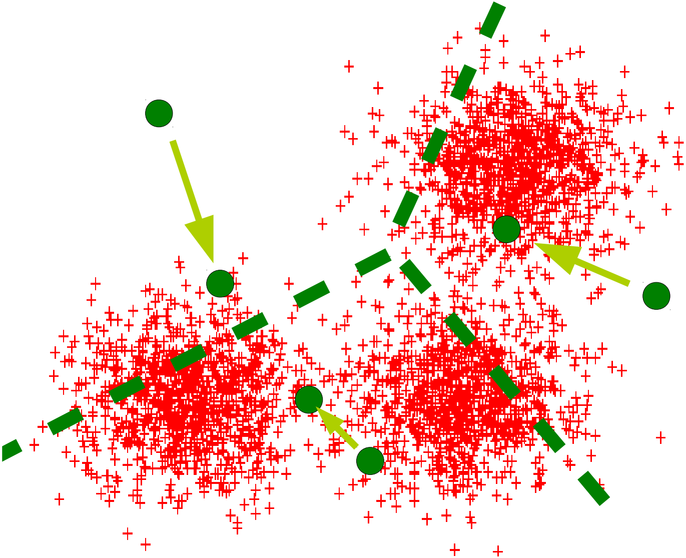
K-Means: Approach

When trying to determine the clusters / the means, we face a **chicken-egg problem**

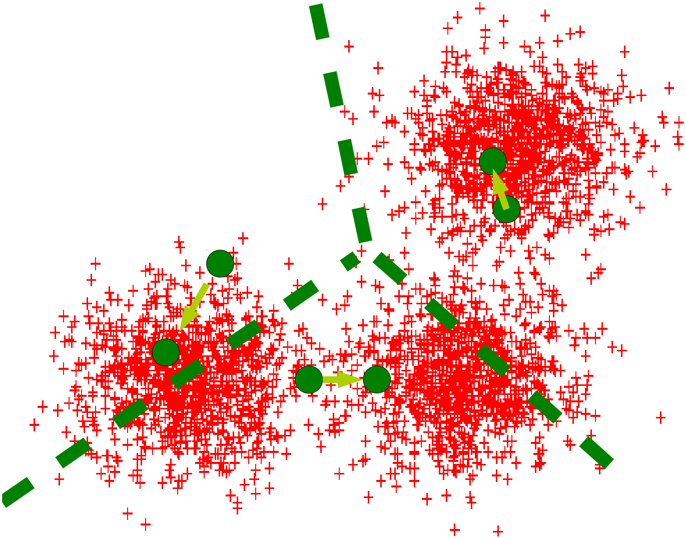
- If we knew the clusters, we could easily determine the means *(by averaging all samples of a cluster)*
- If we knew the means, we could determine the clusters *(by assigning each sample to its closest mean)*
- Approach (**interleaved optimization**): Alternately, fix the clusters/means and estimate the other

function KMEANS($\mathbf{x}_1, \ldots, \mathbf{x}_n,\ K$)
  initialize $\mu_1, \ldots, \mu_K$ by random sampling from $\mathbf{x}_1, \ldots, \mathbf{x}_n$
  repeat
    for $i = 1, .., n$: // assign each sample to its closest cluster
      $k(i) := \arg\min_{k=1,\ldots,K} \|\mathbf{x}_i - \mu_k\|$
    for $k = 1, \ldots K$: // re-estimate each cluster's mean
      $X_k := \{\mathbf{x}_i \mid k(i) = k\}$
      $\mu_k := \frac{1}{|X_k|} \sum_{\mathbf{x} \in X_k} \mathbf{x}$
  until $k(1), \ldots, k(n)$ do not change
  return $\mu_1, \ldots, \mu_K$

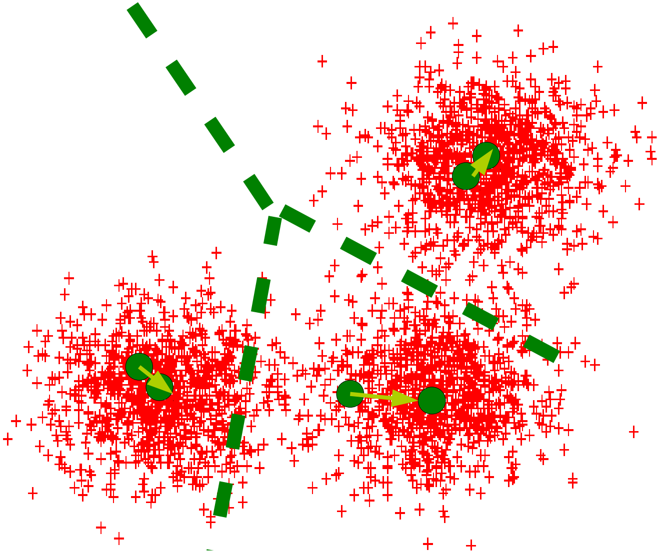
K-Means: Example (Step 1)



K-Means: Example (Step 2)



K-Means: Example (Step 3...)



K-Means: Properties

- K-Means corresponds to a local optimization of the sum of squared errors

$$E(\mu_1, ..., \mu_K) = \sum_{i=1}^{n} (\mathbf{x}_i - \mu_{k(i)})^2$$

- Computational effort: $O(K \cdot n \cdot d)$ per iteration. The number of iterations is often moderate.
- Convergence is guaranteed.

Proof of Convergence

K-Means: Properties

Proof of Convergence (cont'd)

K-Means: Properties

Proof of Convergence (cont'd)

K-Means: Properties

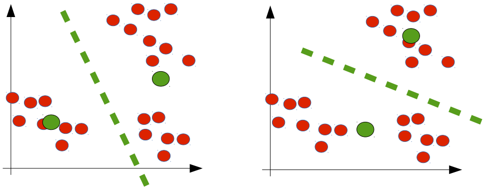
Does K-Means always lead to the same results?

No: K-Means is a local search method!

- **Problem 1**: The order of means can be permuted

$$\mu_1 = (0,0), \mu_2 = (1,1), \mu_3 = (5,3)$$
$$\mu_1 = (5,3), \mu_2 = (0,0), \mu_3 = (1,1)$$

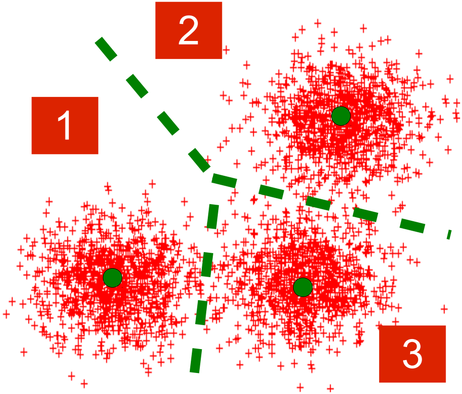
- **Problem 2**: The resulting means can be completely different
- **Approach**: Restart multiple times, and keep the result with minimal error E.
- During the algorithm, **empty clusters** may occur. **Approach**: Reinitialize the corresponding center randomly and continue.



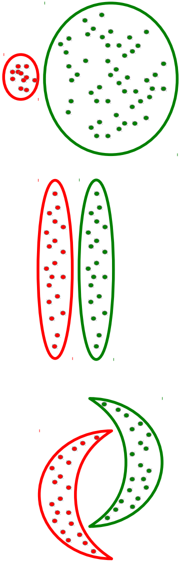
K-Means: Properties (cont'd)

Given a clustering result $\mu_1, ..., \mu_K$, we can assign new samples $\mathbf{x}$ to clusters (this is called **vector quantization**):

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



K-Means: Discussion



Outline

Choosing K: Model Selection

> *"Model selection is the task of selecting a statistical model from a set of candidate models, given data."*
>
> (en.wikipedia.org)

Here: Model Selection = Choosing K

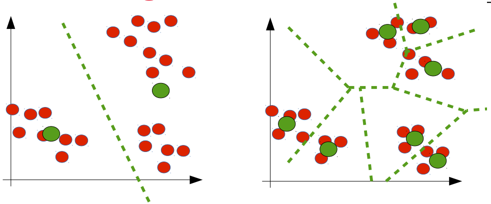
- K too small (*undersegmentation*): clusters too diverse
- K too high (*oversegmentation*): too many parameters, clusters too fine-grain
- Choosing the 'wrong' K leads to **instable results**

Approach 1: External Benchmark

- Sometimes, clustering is just one processing step of a **larger system**, and we can benchmark that larger system
- **Example**: User clustering for advertising
  (→ benchmark by click-through-rate)

Approach 2: Cluster Validation

Goal: measure a model's **goodness-of-fit** without labels



Example: The **Bayes' Information Criterion (BIC)**

1. The clusters should be **compact** *(small error E)*
2. The model should be simple, i.e. have only **few parameters**

- Let θ be the model parameters to learn, and let $\#\theta$ be their number (e.g., in K-Means: $\#\theta = K \cdot d$)
- Test different values of K, and pick this one:

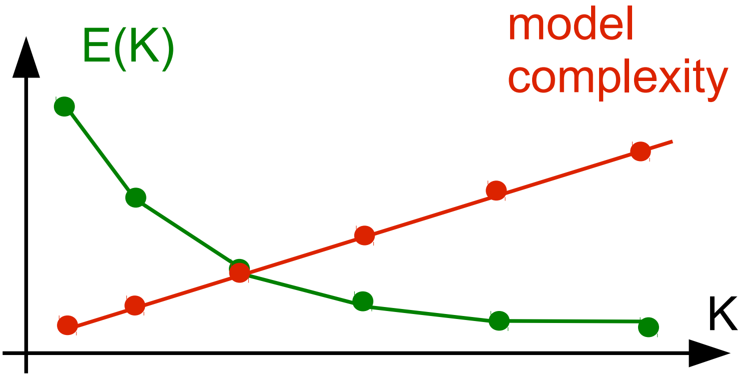
$$K^* = \arg\min_K \; -2 ln\Big(p(\mathbf{x}_1, ..., \mathbf{x}_n|\theta)\Big) + \#\theta \cdot ln(n)$$

BIC for K-Means: Derivation

BIC for K-Means: Derivation

The Bayes Information Criterion

$$K^* = \arg\min_K \underbrace{\sum_{i=1}^{n} \left(\mathbf{x}_i - \mu_{k(i)}\right)^2}_{\mathbf{E(K)}} + \underbrace{K \cdot d \cdot ln(n)}_{\text{model complexity}}$$



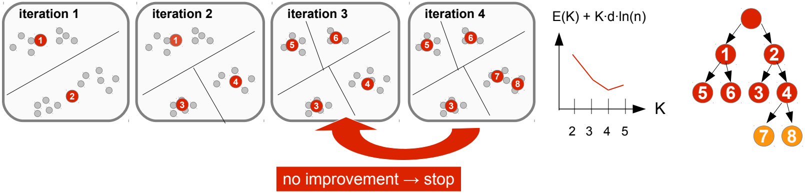
Selecting K: Search Strategies

Approach 1: **Naive**

- test values for K in a reasonable range.
- For every K, re-run clustering and evaluate (**expensive!**)

Approach 2: **Hierarchical Clustering** *(more efficient)*

- ... Iteratively, pick the largest cluster
- ... and apply K-Means to the samples in this cluster, obtaining K new clusters
- ... stop once the overall quality (e.g., BIC) stops improving
- We obtain a **tree** of clusters

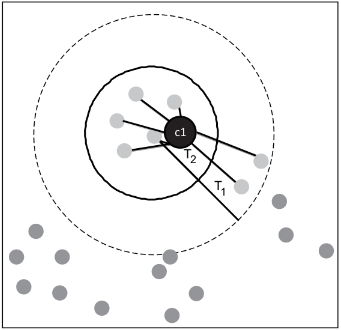


Selecting K: Canopy Clustering image from [7]

Approach 3: Canopy Clustering

- A **greedy strategy** to find (potentially suboptimal) clusters on large datasets
- We use it to estimate K and to initialize the means
- Canopy clusters can **overlap**!
- Canopy clustering uses **two thresholds**
  - T_1 (determines the number of clusters)
  - T_2 (determines the overlap of clusters) ($T_2 > T_1$)

```
function CLUSTER_CANOPY(X := {x_1, ..., x_n})
    C := {}
    while X <> {}:
        choose a random sample x ∈ X
        Y := {y ∈ X | ||y − x|| ≤ T_1}
        Z := {y ∈ X | T_1 < ||y − x|| ≤ T_2}
        C := C ∪{x}
        X := X\Y
    return C
```



Outline

Expectation Maximization (EM)

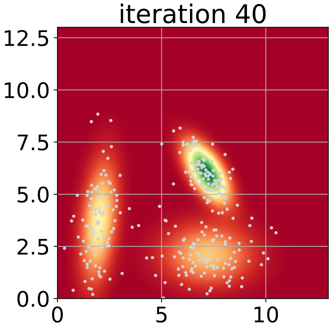
- We can overcome some of the above limitations by **generalizing K-Means**, resulting in a famous approach called **Expectation Maximization (EM)**

EM: Model

- We explain the data $\mathbf{x}_1, \ldots, \mathbf{x}_n$ by a **Gaussian mixture model**

$$\mathbf{x}_1, \ldots, \mathbf{x}_n \sim \sum_{k=1}^{K} P_k \cdot p(\mathbf{x}|\mu_k, \Sigma_k)$$

where p is the **multivariate normal density** *(Chapter 3)*, $\mu_1, \ldots, \mu_K$ are K centers, $\Sigma_1, \ldots, \Sigma_K$ are K covariance matrices (the *shapes* of the clusters), and $P_1, \ldots, P_K$ are the cluster's proportions of the data (also called *priors*).



Expectation Maximization (EM)

Remarks

- In K-Means, we would have $P_1 = P_2 = ... = P_K = \frac{1}{K}$ and

$$\Sigma_1 = \Sigma_2 = ... = \Sigma_K = \begin{pmatrix} \sigma^2 & 0 & ... & 0 \\ 0 & \sigma^2 & .... & 0 \\ 0 & ... & ... & 0 \\ 0 & ... & 0 & \sigma^2 \end{pmatrix}$$

Approach

- We **rename** the two alternating K-Means steps

E-Step Re-assigning samples to clusters $\rightarrow$ "Expectation-Step"

M-Step Re-estimating the cluster centers $\rightarrow$ "Maximization-Step"

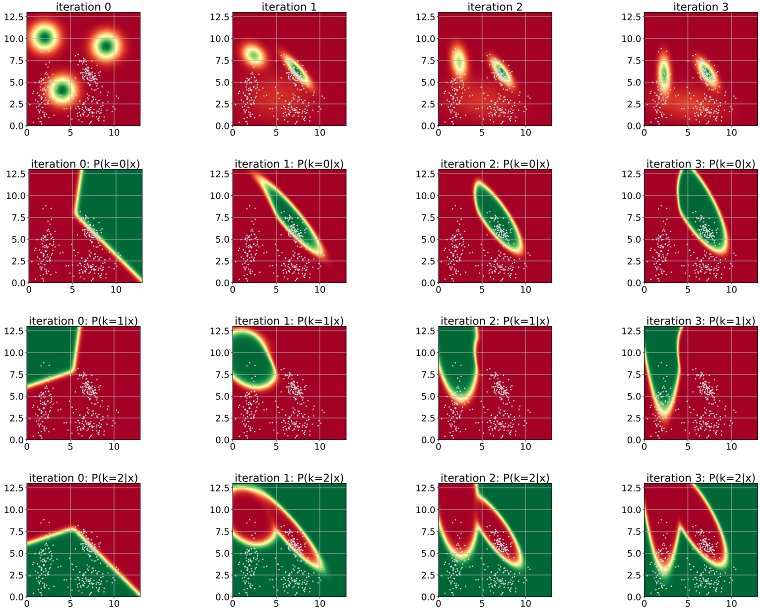
- We **modify** these steps a bit
  - **E-Step**: No hard assignment of samples to centers, but a **soft assignment** by computing the probability $P(k(i) = k \mid \mathbf{x}_i)$
  - **M-Step**: Do not only estimate the cluster *centers*, but **parameters** in general (e.g., the clusters' shape+prior)

K-Means vs. Expectation Maximization (EM)

Illustration

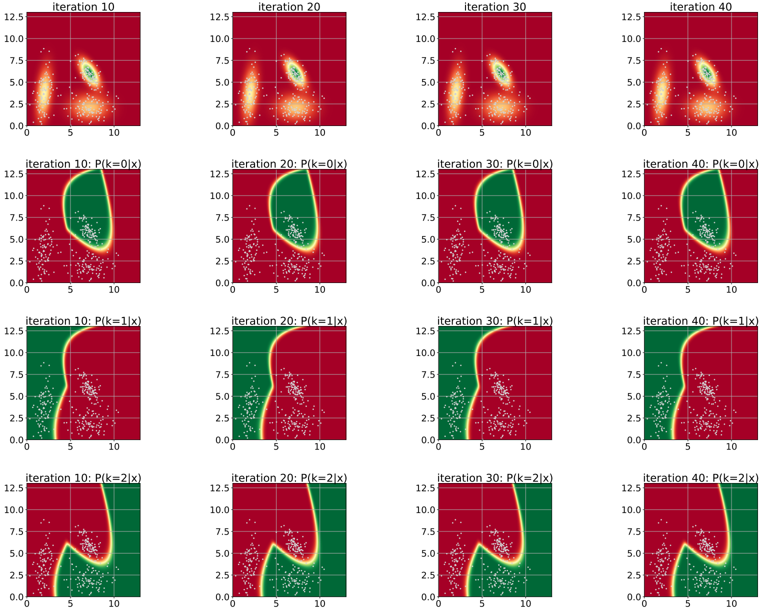
	K-Means	EM
E-Step	$k(i) := \arg\min_k \|\|\mathbf{x}_i - \mu_{k(i)}\|\|$	$w_{ki} := P(k(i) = k\|\mathbf{x}_i) = \frac{p(\mathbf{x}_i;\mu_k,\Sigma_k)}{\sum_{k'} p(\mathbf{x}_i;\mu_{k'},\Sigma_{k'})}$
M-Step	$\mu_k := \frac{\sum_{\mathbf{x}\in X_k} \mathbf{x}}{\|X_k\|}$	$\mu_k := \frac{\sum_i w_{ki}\cdot\mathbf{x}_i}{\sum_i w_{ki}}$
	—	$\Sigma_k := \frac{\sum_i w_{ki}\cdot(\mathbf{x}_i-\mu_k)(\mathbf{x}_i-\mu_k)^T}{\sum_i w_{ki}}$
	—	$P_k := \frac{\sum_i w_{ki}}{\sum_{k'}\sum_i w_{k'i}}$

EM: Example



iteration 0
iteration 1
iteration 2
iteration 3
iteration 0: P(k=0|x)
iteration 1: P(k=0|x)
iteration 2: P(k=0|x)
iteration 3: P(k=0|x)
iteration 0: P(k=1|x)
iteration 1: P(k=1|x)
iteration 2: P(k=1|x)
iteration 3: P(k=1|x)
iteration 0: P(k=2|x)
iteration 1: P(k=2|x)
iteration 2: P(k=2|x)
iteration 3: P(k=2|x)

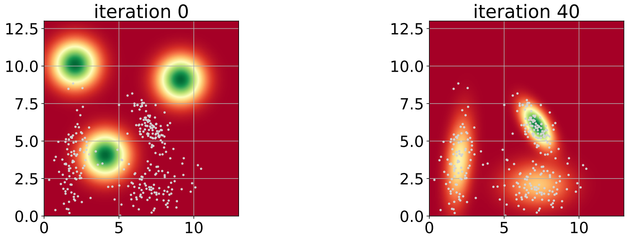
EM: Example


iteration 10
iteration 20
iteration 30
iteration 40
iteration 10: P(k=0|x)
iteration 20: P(k=0|x)
iteration 30: P(k=0|x)
iteration 40: P(k=0|x)
iteration 10: P(k=1|x)
iteration 20: P(k=1|x)
iteration 30: P(k=1|x)
iteration 40: P(k=1|x)
iteration 10: P(k=2|x)
iteration 20: P(k=2|x)
iteration 30: P(k=2|x)
iteration 40: P(k=2|x)
0.0
2.5
5.0
7.5
10.0
12.5
0
5
10

EM: Goodness-of-Fit

- Goal: restart EM many times, pick the 'best' model.
- Given an **EM model** $\Theta = (\mu_1, ..., \mu_K, \Sigma_1, ..., \Sigma_K, P_1, ..., P_K)$, we want to measure its **"goodness-of-fit"**.
- Approach: We measure the **likelihood** of the data

$$
\begin{aligned}
L(\mathbf{x}_1, ..., \mathbf{x}_n; \Theta) &= \prod_i p(\mathbf{x}_i | \Theta) \\
&= \prod_i \sum_k P_k \cdot p(\mathbf{x}_i; \mu_k, \Sigma_k)
\end{aligned}
$$



low likelihood　　high likelihood

EM: Discussion

EM as a general Learning Scheme

- EM for **Gaussian Mixture Models** is just a special case!

symbol	general EM	Gaussian Mixture Models
X	(known) input data	the features $\mathbf{x}_1, ..., \mathbf{x}_n$
Θ	parameters	means $\mu_1, ..., \mu_K$, shapes $\Sigma_1, ..., \Sigma_K$, priors $P_1, ..., P_K$
U	unknown data	the mapping from $\mathbf{x}_i$ to clusters k

EM: General Learning Scheme

```
function EM(X)
    initialize Θ randomly
    repeat
        compute P(U|X,Θ)                                   // E-step
        optimize parameters [6], obtaining a new Θ         // M-step
    until convergence
    return Θ
```

Outline

Document Clustering

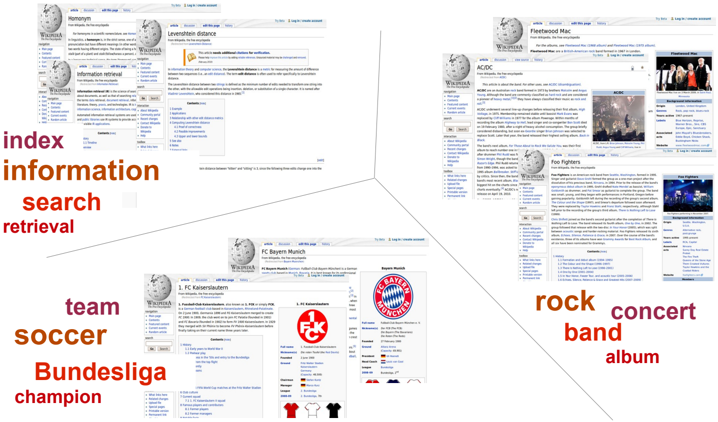
We can also **cluster text** using EM. The resulting method is called **Probabilistic Latent Semantic Analysis (PLSA)**

- Given: a set of documents with their **bag-of-words** features
- PLSA divides the set of documents into clusters of **semantically similar** documents
- The cluster centers correspond to prototypical word distributions (or **topics**)
- We also call PLSA a **topic model**

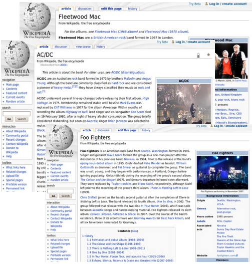
Remarks

- This is clustering, not classification! Categories/topics are not pre-defined, but PLSA discovers them by itself!

PLSA: Illustration



PLSA: Notation



rock concert
band
album

- **Input**: a collection of documents $d_1, \ldots, d_n$ and a vocabulary of terms $w_1, \ldots, w_m$
- Each document d is represented by its **bag-of-words** feature. This gives us a **probability distribution** of words $P(w|d)$.
- We assume the document collection to consist of K **topics** $z_1, \ldots, z_K$
- Each topic z has a **word distribution** $P(w|z)$ (just like a document)
- A document d can be seen a **mixture of topics**, $P(z|d)$

PLSA: Sampling Process

Words are **sampled** from a document d in **two steps**

1. Choose a random topic z' from $P(z_1|d), \ldots, P(z_K|d)$
2. Given z', pick a word from $P(w_1|z'), \ldots, P(w_m|z')$

Document d

1. choose topic $P(z|d)$

Topics z_1, z_2, z_3

2. choose term $P(w|z)$

PLSA: Derivation

PLSA Clustering estimates two **probability distributions**:

1. $P(z|d)$
   - $P(z|d)$ tells us **which topics appear in a document** (or which topics (clusters) a document *belongs to*)
   - $P(z|d)$ is a $K \times n$ probability table (covering *all topic-document combinations*)
2. $P(w|z)$
   - This distributions tells us **which words appear in a topic**
   - $P(w|z)$ is an $m \times K$ probability table (covering *all word-topic combinations*)

PLSA Approach

To estimate the above distributions, PLSA uses the EM Algorithm

- **E-Step** (*assign samples to clusters*)
  $\rightarrow$ assign **words to topics** ($=$ compute $P(z|w,d)$)
- **M-Step** (*estimate cluster parameters*)
  $\rightarrow$ estimate **topics and mixtures** ($= P(w|z), P(z|d)$)

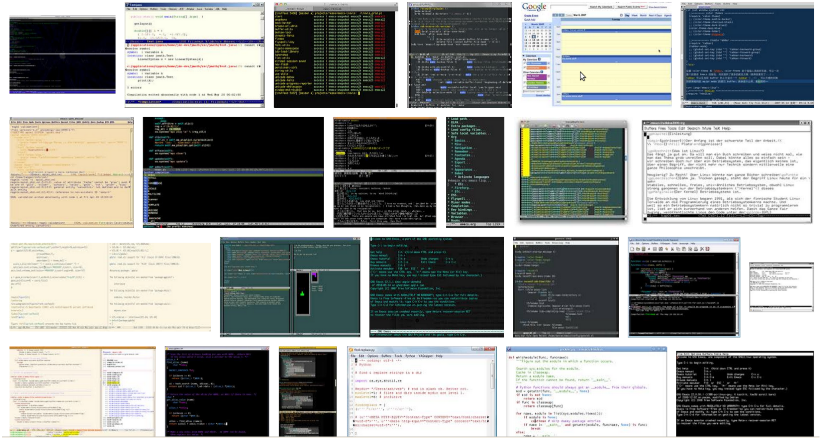
PLSA: Algorithm

- **Given**: documents $d_1, ..., d_n$, terms $w_1, ..., w_m$
- **Given**: bag-of-words $P(w|d)$, number of topics K
- Initialize $P(w|z)$, $P(z|d)$ randomly
- Repeat until convergence:

PLSA: Algorithm

PLSA: Algorithm

PLSA: Code Sample



PLSA: Discussion

Outline

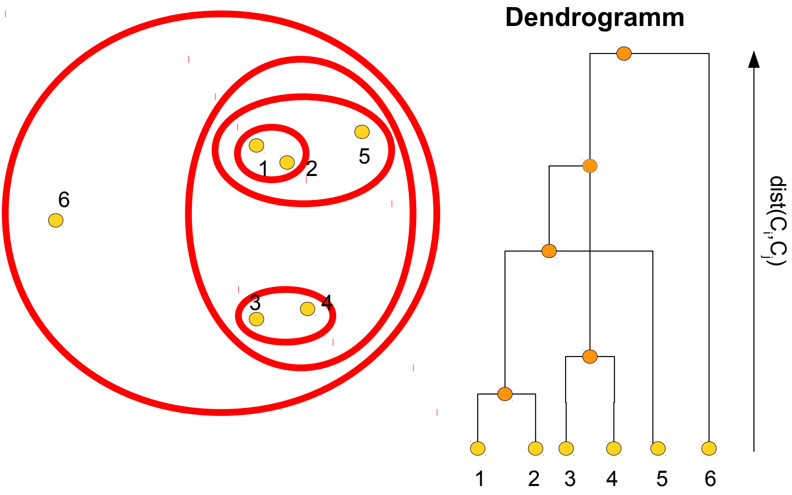
Agglomerative Clustering

- We call K-Means/EM **divisive** clustering techniques, because they divide the dataset top-down

Agglomerative clustering

- initially, each sample belongs to its own cluster (*singleton*)
- iteratively, we merge the two “most similar” clusters and obtain a new, bigger cluster
- The result can be illustrated in form of a tree-like graph, a so-called **dendrogram**

Agglomerative Clustering: Illustration



Agglomerative Clustering: Further Issues

Stopping Criterion

- heuristics (see *model selection*)

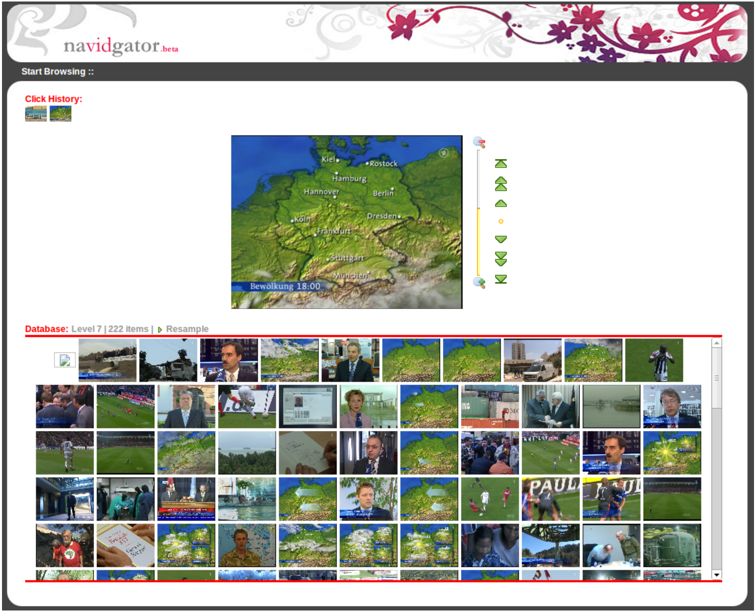
Distance Measure

- We need to define a **distance between clusters** to pick the "most similar" clusters to fuse.
- The three common alternatives (let X, Y be clusters):

single linkage	$dist(X, Y) := \min_{x \in X, y \in Y} \|\|x - y\|\|^2$
complete linkage	$dist(X, Y) := \max_{x \in X, y \in Y} \|\|x - y\|\|^2$
average linkage	$dist(X, Y) := \frac{1}{\|X\| \cdot \|Y\|} \sum_{x \in X, y \in Y} \|\|x - y\|\|^2$

Agglomerative Clustering: Discussion

Agglomerative Clustering: Application Example [5]



References I

[1] M. Dukia: DoS(Denial of Service) Attacks.
http://www.securitykiller.org/2015/12/dosdenial-of-service-attacks.html (retrieved: Nov 2016).

[2] P. Ipeirotis: A Plea to Amazon: Fix Mechanical Turk! .
http://www.behind-the-enemy-lines.com/2010/10/plea-to-amazon-fix-mechanical-turk.html (retrieved: Nov 2016).

[3] The Value of a Professional Network?
https://www.linkedin.com/pulse/value-professional-network-daniel-tunkelang (retrieved: Oct 2016).

[4] University of Vermont: Complex Networks (Course Page).
http://www.uvm.edu/pdodds/teaching/courses/2010-01UVM-303/content/pictures.html (retrieved: Nov 2016).

[5] D. Borth, C. Schulze, A. Ulges, and T. Breuel.
Navidgator - Similarity Based Browsing for Image & Video Databases.
In Proc. KI 2008, pages 22–29, September 2008.

[6] F. Dellaert.
The Expectation Maximization Algorithm.
Technical Report GIT-GVU-02-20, Georgia Institute of Technology, 2002.

[7] S. Owen, R. Anil, T. Dunning, and E. Friedman.
Mahout in Action.
Manning Publications, 2012.

[8] A. Ulges, M. Koch, and D. Borth.
Linking Visual Concept Detection with Viewer Demographics.
In ACM International Conference on Multimedia Retrieval, 6 2012.