Knowledge Discovery in Databases

SS 2016

Chapter 4: Clustering

Lecture: Prof. Dr. Thomas Seidl

Tutorials: Julian Busch, Evgeniy Faerman,
Florian Richter, Klaus Schmid

Contents

1) Introduction to Clustering
2) Partitioning Methods
   - K-Means
   - Variants: K-Medoid, K-Mode, K-Median
   - Choice of parameters: Initialization, Silhouette coefficient
3) Probabilistic Model-Based Clusters: Expectation Maximization
4) Density-based Methods: DBSCAN
5) Hierarchical Methods
   - Agglomerative and Divisive Hierarchical Clustering
   - Density-based hierarchical clustering: OPTICS
6) Evaluation of Clustering Results
7) Further Clustering Topics
   - Ensemble Clustering
   - Discussion: an alternative view on DBSCAN

What is Clustering?

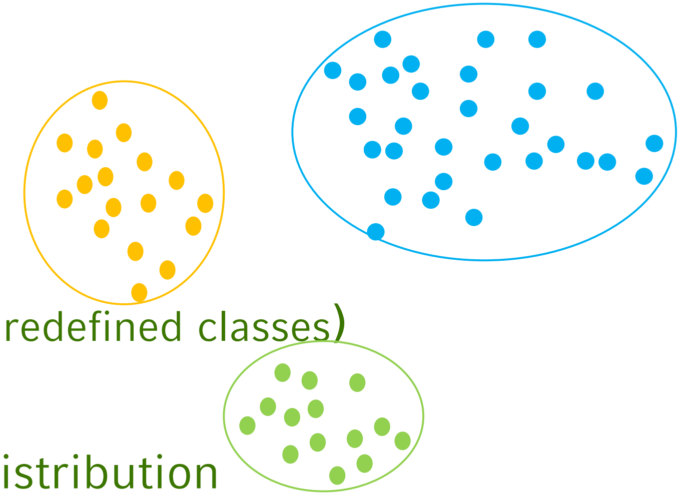
Grouping a set of data objects into clusters

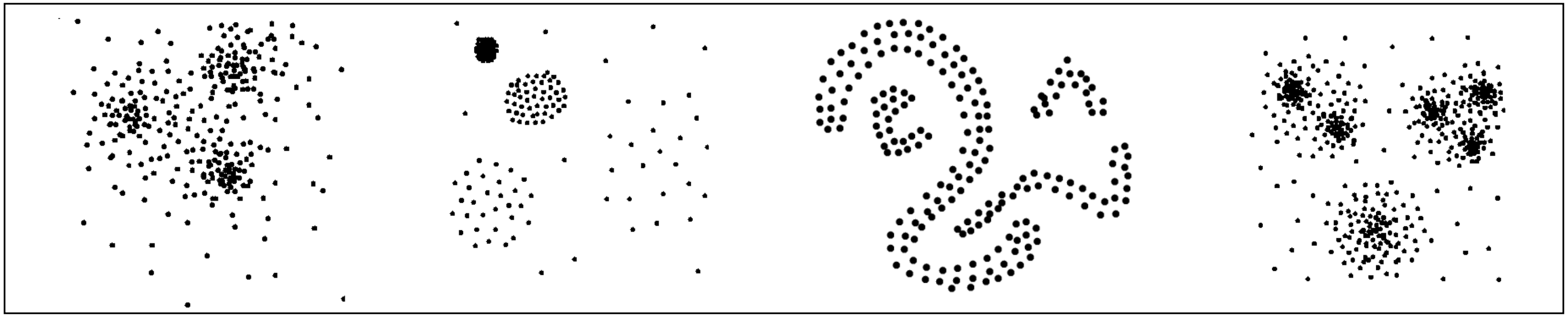
- Cluster: a collection of data objects
  1) *Similar* to one another within the same cluster
  2) *Dissimilar* to the objects in other clusters

Clustering = ***unsupervised "classification"*** (no predefined classes)

Typical usage

- As a *stand-alone tool* to get insight into data distribution
- As a *preprocessing step* for other algorithms





General Applications of Clustering

Preprocessing – as a data reduction (instead of sampling)

- Image data bases (color histograms for filter distances)
- Stream clustering (handle endless data sets for offline clustering)

Pattern Recognition and Image Processing

Spatial Data Analysis

- create thematic maps in Geographic Information Systems by clustering feature spaces
- detect spatial clusters and explain them in spatial data mining

Business Intelligence (especially market research)

WWW

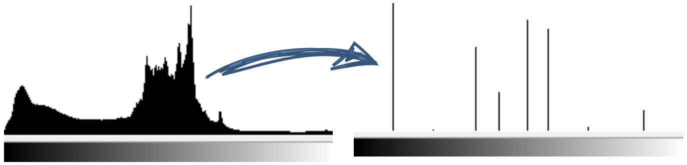
- Documents (Web Content Mining)
- Web-logs (Web Usage Mining)

Biology

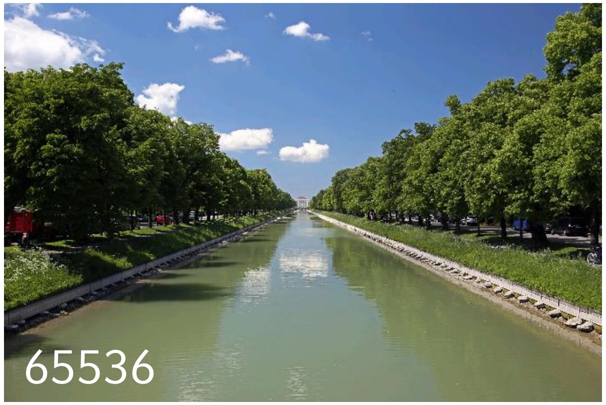
- Clustering of gene expression data

An Application Example: Downsampling Images

- Reassign color values to k distinct colors
- Cluster pixels using color difference, not spatial data

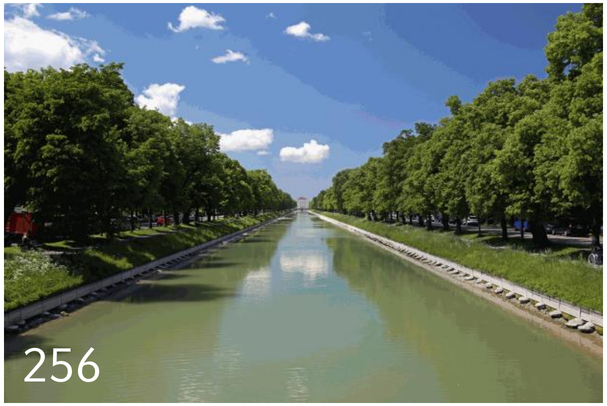


58483 KB



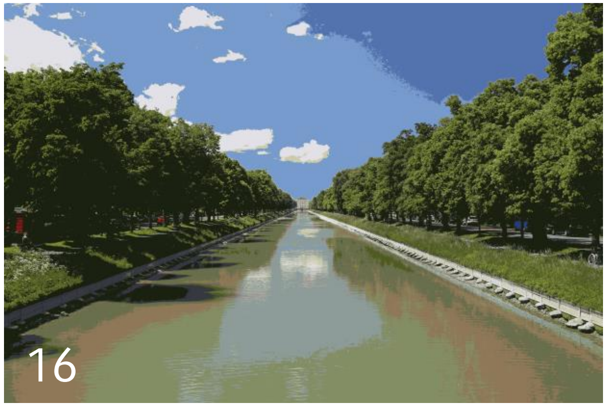
65536

19496 KB

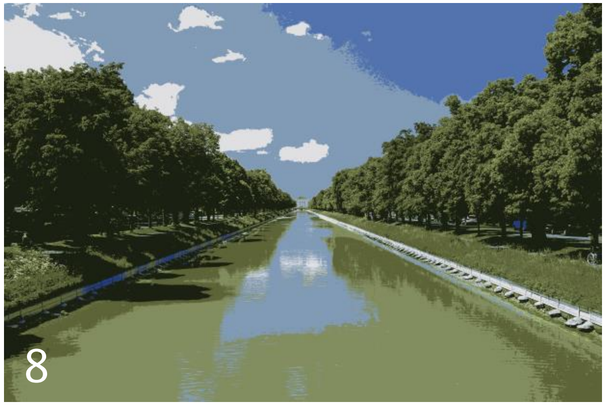


256

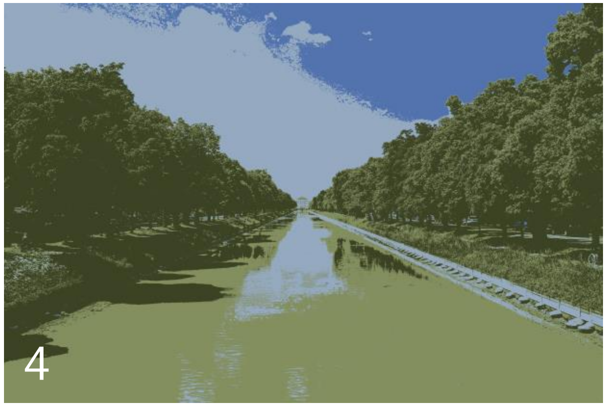
9748 KB



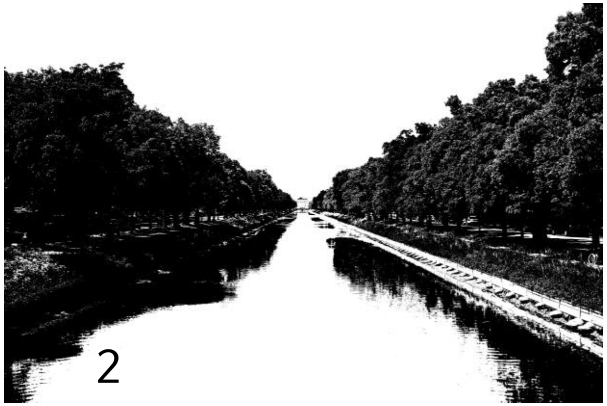
16



8



4



2

Major Clustering Approaches

Partitioning algorithms

- Find k partitions, minimizing some objective function

Probabilistic Model-Based Clustering (EM)

Density-based

- Find clusters based on connectivity and density functions

Hierarchical algorithms

- Create a hierarchical decomposition of the set of objects

Other methods

- Grid-based
- Neural networks (SOM's)
- Graph-theoretical methods
- Subspace Clustering
- ...

Contents

1) Introduction to clustering

2) Partitioning Methods
   - K-Means
   - K-Medoid
   - Choice of parameters: Initialization, Silhouette coefficient

3) Expectation Maximization: a statistical approach

4) Density-based Methods: DBSCAN

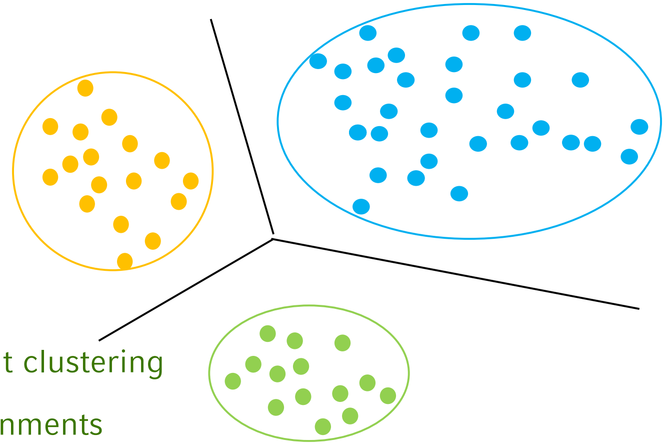
5) Hierarchical Methods
   - Agglomerative and Divisive Hierarchical Clustering
   - Density-based hierarchical clustering: OPTICS

6) Evaluation of Clustering Results

7) Further Clustering Topics
   - Ensemble Clustering
   - Discussion: an alternative view on DBSCAN
   - Outlier Detection

Partitioning Algorithms: Basic Concept

- Goal: Construct a partition of a database D of n objects into a set of k $(k < n)$ clusters $C_1, \dots, C_k$ $(\mathrm{C_i} \subset D, C_i \cap C_j = \emptyset \Leftrightarrow C_i \neq C_j, \bigcup C_i = D)$ minimizing an objective function.
  - Exhaustively enumerating all possible partitions into *k* sets in order to find the global minimum is too expensive.



- Popular heuristic methods:
  - Choose *k* representatives for clusters, e.g., randomly
  - Improve these initial representatives iteratively:
    - Assign each object to the cluster it “fits best” in the current clustering
    - Compute new cluster representatives based on these assignments
    - Repeat until the change in the objective function from one iteration to the next drops below a threshold

- Examples of representatives for clusters
  - *k*-means: Each cluster is represented by the center of the cluster
  - *k*-medoid: Each cluster is represented by one of its objects

Contents

1) Introduction to clustering

2) Partitioning Methods
   - <u>K-Means</u>
   - Variants: K-Medoid, K-Mode, K-Median
   - Choice of parameters: Initialization, Silhouette coefficient

3) Probabilistic Model-Based Clusters: Expectation Maximization

4) Density-based Methods: DBSCAN

5) Hierarchical Methods
   - Agglomerative and Divisive Hierarchical Clustering
   - Density-based hierarchical clustering: OPTICS

6) Evaluation of Clustering Results

7) Further Clustering Topics
   - Scaling Up Clustering Algorithms
   - Outlier Detection

K-Means Clustering: Basic Idea

Idea of K-means: find a clustering such that the *within-cluster variation* of each cluster is small and use the *centroid* of a cluster as representative.

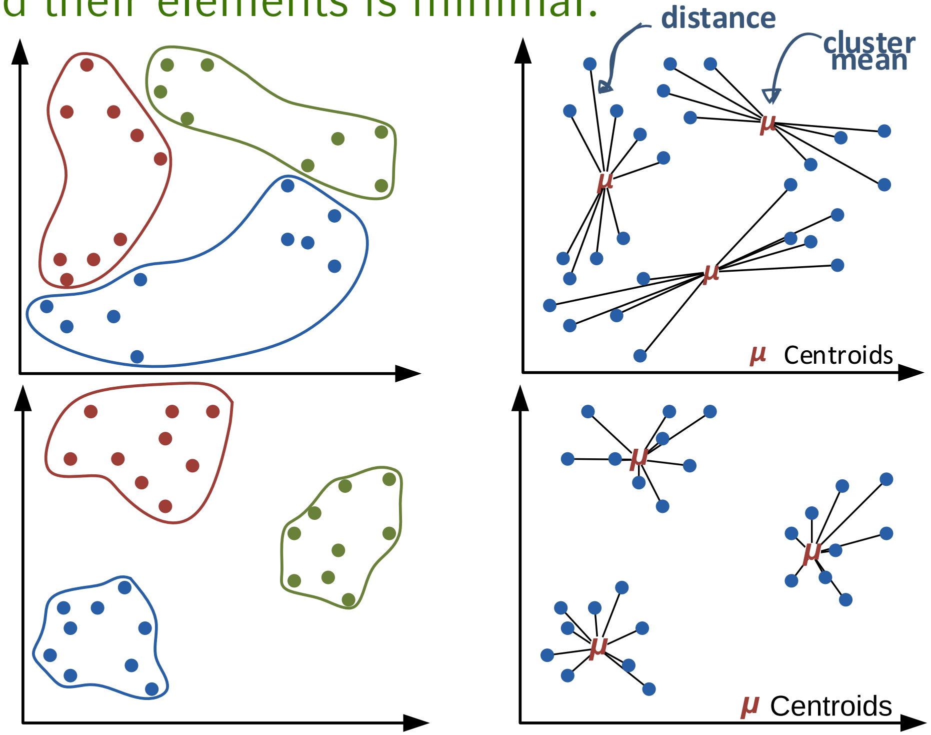
Objective: For a given *k*, form *k* groups so that the sum of the (squared) distances between the mean of the groups and their elements is minimal.

Poor Clustering

(large sum of distances)

Optimal Clustering

(minimal sum of distances)



S.P. Lloyd: Least squares quantization in PCM. In IEEE Information Theory, 1982 (original version: technical report, Bell Labs, 1957)
J. MacQueen: *Some methods for classification and analysis of multivariate observation*, In Proc. of the 5th Berkeley Symp. on Math. Statist. and Prob., 1967.

K-Means Clustering: Basic Notions

Objects $p = (p_1, \dots, p_d)$ are points in a d-dimensional vector space

(the mean μ_S of a set of points S must be defined: $\mu_S = \frac{1}{|S|}\sum_{p\in S} p$)

Measure for the compactness of a **cluster** C_j (sum of squared errors):

$$SSE(C_j) = \sum_{p\in C_j} dist\left(p, \mu_{C_j}\right)^2$$

Measure for the compactness of a **clustering** $\mathcal{C}$**:**

$$SSE(\mathcal{C}) = \sum_{C_j\in\mathcal{C}} SSE(C_j) = \sum_{p\in DB} dist\left(p, \mu_{C(p)}\right)^2$$

Optimal Partitioning: $\underset{\mathcal{C}}{\operatorname{argmin}}\, SSE(\mathcal{C})$

Optimizing the within-cluster variation is computationally challenging (NP-hard) → use efficient heuristic algorithms

K-Means Clustering: Algorithm

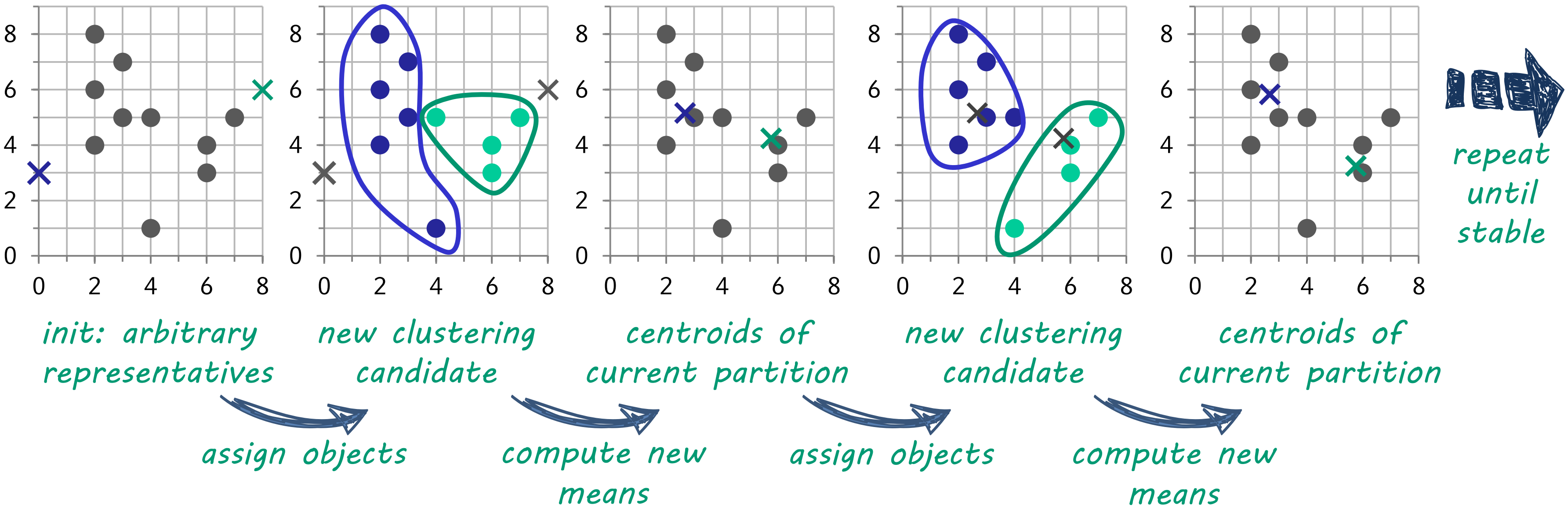
k-Means algorithm (Lloyd’s algorithm):

Given *k*, the *k*-means algorithm is implemented in 2 main steps:

Initialization: Choose *k* arbitrary representatives

Repeat until representatives do not change:

1. Assign each object to the cluster with the nearest representative.
2. Compute the centroids of the clusters of the current partitioning.



K-Means Clustering: Discussion

Strengths

- Relatively efficient: $O(tkn)$, where n = # objects, k = # clusters, and t = # iterations
- Typically: k, $t << n$
- Easy implementation

Weaknesses

- Applicable only when mean is defined
- Need to specify *k,* the number of clusters, in advance
- Sensitive to noisy data and outliers
- Clusters are forced to convex space partitions (Voronoi Cells)
- Result and runtime strongly depend on the initial partition; often terminates at
  a *local optimum* – however: methods for a good initialization exist

Several variants of the *k*-means method exist, e.g., ISODATA

- Extends *k*-means by methods to eliminate very small clusters, merging and split of clusters; user has to specify additional parameters

Contents

1) Introduction to clustering

2) Partitioning Methods
   - K-Means
   - Variants: K-Medoid, K-Mode, K-Median
   - Choice of parameters: Initialization, Silhouette coefficient

3) Probabilistic Model-Based Clusters: Expectation Maximization

4) Density-based Methods: DBSCAN

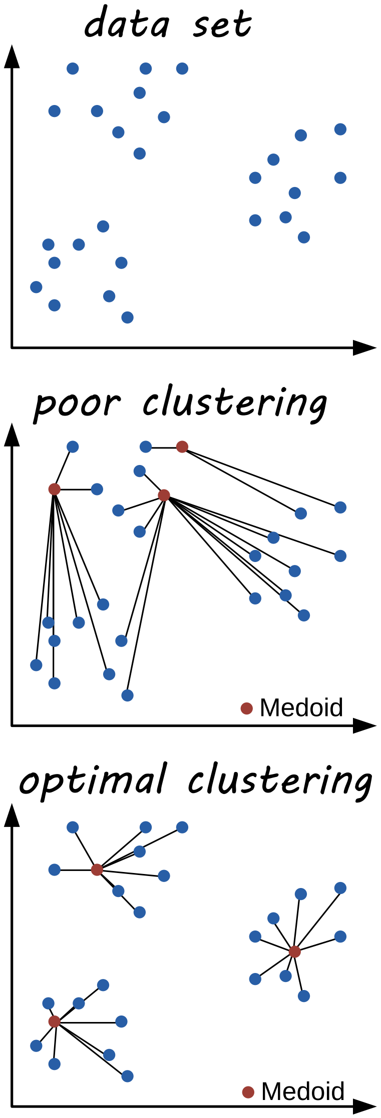
5) Hierarchical Methods
   - Agglomerative and Divisive Hierarchical Clustering
   - Density-based hierarchical clustering: OPTICS

6) Evaluation of Clustering Results

7) Further Clustering Topics
   - Scaling Up Clustering Algorithms
   - Outlier Detection

K-Medoid, *K*-Modes, *K*-Median Clustering: Basic Idea

- Problems with K-Means:
  - Applicable only when mean is defined (vector space)
  - Outliers have a strong influence on the result
- The influence of outliers is intensified by the use of the *squared* error → use the absolute error (total distance instead): $TD(C) = \sum_{p\in C} dist(p, m_{c(p)})$ and $TD(\mathcal{C}) = \sum_{C_i\in\mathcal{C}} TD(C_i)$
- Three alternatives for using the Mean as representative:
  - *Medoid*: representative object "in the middle"
  - *Mode*: value that appears most often
  - *Median*: (artificial) representative object "in the middle"
- Objective as for k-Means: Find *k* representatives so that, the sum of the distances between objects and their closest representative is minimal.



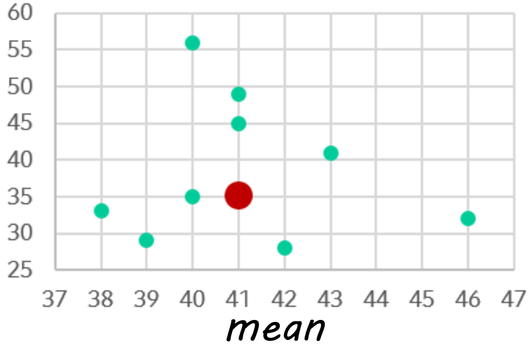
K-Median Clustering

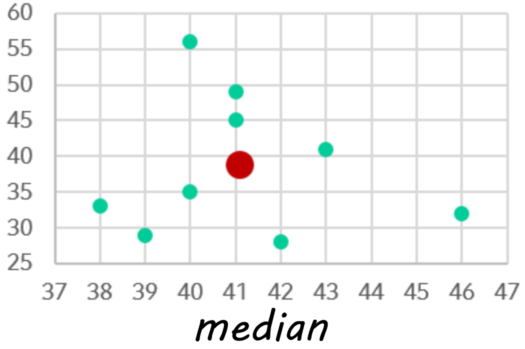
Problem: Sometimes, data is not numerical

Idea: If there is an ordering on the data $X = \{x_1, x_2, x_3, \dots, x_n\}$, use median instead of mean

$$Median(\{x\}) = x$$
$$Median(\{x, y\}) \in \{x, y\}$$
$$Median(X) = Median(X - \min X - \max X), \qquad if\ |X| > 2$$

- A median is computed in each dimension independently and can thus be a combination of multiple instances
  → median can be efficiently computed for ordered data
- Different strategies to determine the “middle” in an array of even length possible





K-Mode Clustering: First Approach [Huang 1997]

Given: $X \subseteq \Omega = A_1 \times A_2 \times \cdots \times A_d$ is a set of n objects, each described by d categorical attributes $A_i \;\; (1 \leq i \leq d)$

Mode: a mode of X is a vector $M = [m_1, \mathrm{m}_2, \cdots, m_d] \in \Omega$ that minimizes

$$d(M, X) = \sum_{x_i \in X} d(x_i, M)$$

where d is a distance function for categorical values (e.g. Hamming Dist.)

→ Note: M is not necessarily an element of X

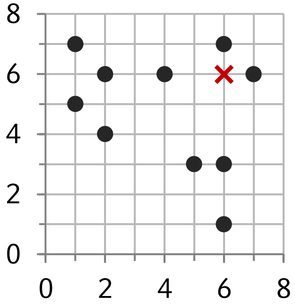
Theorem to determine a Mode: let $f(c, j, X) = \frac{1}{n} \cdot |\{x \in X | \, x[j] = c\}|$ be the relative frequency of category c of attribute A_j in the data, then:

$$d(M, X) \text{ is minimal} \Leftrightarrow \forall j \in \{1, \ldots, d\}\colon \forall c \in A_j\colon f\left(\mathrm{m_j}, \mathrm{j}, \mathrm{X}\right) \geq f(c, j, X)$$

→ this allows to use the k-means paradigm to cluster categorical data without loosing its efficiency

→ Note: the mode of a dataset might be not unique

K-Modes algorithm proceeds similar to k-Means algorithm



Huang, Z.: *A Fast Clustering Algorithm to Cluster very Large Categorical Data Sets in Data Mining*, In DMKD, 1997.

K-Mode Clustering: Example

Employee-ID	Profession	Household Pets
#133	Technician	Cat
#134	Manager	None
#135	Cook	Cat
#136	Programmer	Dog
#137	Programmer	None
#138	Technician	Cat
#139	Programmer	Snake
#140	Cook	Cat
#141	Advisor	Dog

Profession: (**Programmer: 3**, Technician: 2, Cook: 2, Advisor: 1, Manager:1)
Household Pet: (**Cat: 4**, Dog: 2, None: 2, Snake: 1)

Mode is (Programmer, Cat)
Remark: (Programmer, Cat) $\notin DB$

K-Medoid Clustering: PAM Algorithm

Partitioning Around Medoids [Kaufman and Rousseeuw, 1990]

- Given *k*, the *k*-medoid algorithm is implemented in 3 steps:
  - Initialization: Select *k* objects arbitrarily as initial medoids (representatives)
  - assign each remaining (non-medoid) object to the cluster with the nearest representative
  - compute $\text{TD}_{\text{current}}$

- Problem of PAM: high complexity ($O(tk(n-k)$^2))

Kaufman L., Rousseeuw P. J., *Finding Groups in Data: An Introduction to Cluster Analysis*, John Wiley & Sons, 1990.

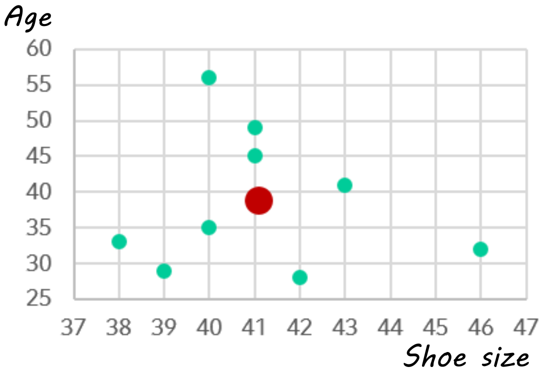
Algorithmus PAM

```
PAM(Punktmenge D, Integer k)
    Initialisiere die k Medoide;
    TD_Änderung := $-\infty$;
    while TD_Änderung < 0 do
        Berechne für jedes Paar (Medoid M, Nicht-Medoid N)
          den Wert $TD_{N \leftrightarrow M}$;
        Wähle das Paar (M, N), für das der Wert
          TD_Änderung := $TD_{N \leftrightarrow M} - TD$ minimal ist;
        if TD_Änderung < 0 then
            ersetze den Medoid M durch den Nicht-Medoid N;
            Speichere die aktuellen Medoide als die bisher beste
              Partitionierung;
    return Medoide;
```

Algorithmus CLARANS

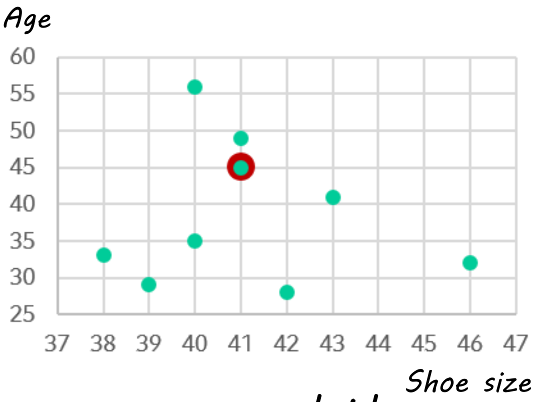
```
CLARANS(Punktmenge D, Integer k,
        Integer numlocal, Integer maxneighbor)
   for r from 1 to numlocal do
      wähle zufällig k Objekte als Medoide; i := 0;
      while i < maxneighbor do
         Wähle zufällig (Medoid M, Nicht-Medoid N);
         Berechne TD_Änderung := $TD_{N\leftrightarrow M}$ – TD;
         if TD_Änderung < 0 then
           ersetze M durch N;
           TD := $TD_{N\leftrightarrow M}$; i := 0;
         else i:= i + 1;
      if TD < TD_best then
         TD_best := TD; Speichere aktuelle Medoide;
   return Medoide;
```

K-Means/Medoid/Mode/Median overview

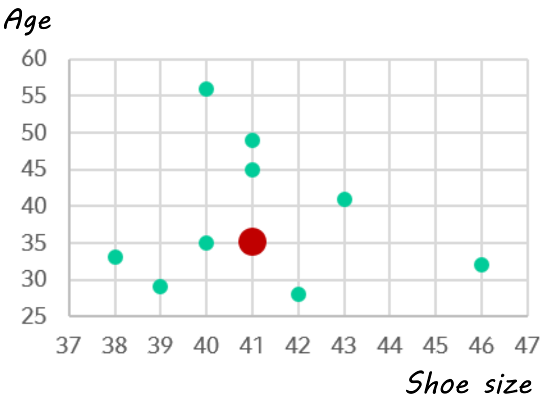


mean

Employee-ID	Profession	Shoe size	Age
#133	Technician	42	28
#134	Manager	41	45
#135	Cook	46	32
#136	Programmer	40	35
#137	Programmer	41	49
#138	Technician	43	41
#139	Programmer	39	29
#140	Cook	38	33
#141	Advisor	40	56



medoid



median

Profession: Programmer
Shoe size: 40/41
Age: n.a.

mode

K-Means/Median/Mode/Medoid Clustering: Discussion

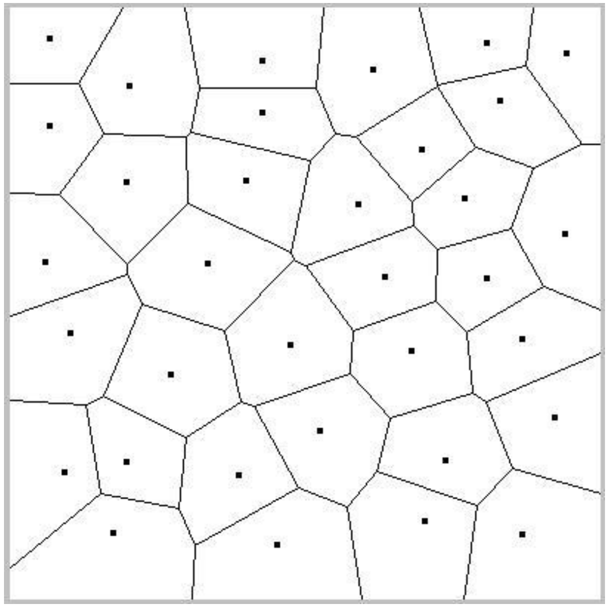
	k-Means	*k*-Median	K-Mode	K-Medoid
data	numerical data (mean)	ordered attribute data	categorical attribute data	metric data
efficiency	high $O(tkn)$	high $O(tkn)$	high $O(tkn)$	low $O(tk(n-k)^2)$
sensitivity to outliers	high	low	low	low

- Strength
  - Easy implementation (→ many variations and optimizations in the literature)
- Weakness
  - Need to specify *k*, the number of clusters, in advance
  - Clusters are forced to convex space partitions (Voronoi Cells)
  - Result and runtime strongly depend on the initial partition; often terminates at a *local optimum* – however: methods for a good initialization exist

Voronoi Model for convex cluster regions

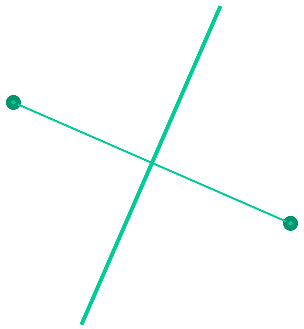
Definition: Voronoi diagram

- For a given set of points $P = \{p_i \mid i = 1 \dots k\}$ (here: cluster representatives), a Voronoi diagram partitions the data space in Voronoi cells, one cell per point.
- The cell of a point $p \in P$ covers all points in the data space for which p is the nearest neighbors among the points from P.



Observations

- The Voronoi cells of two neighboring points $p_i, p_j \in P$ are separated by the perpendicular hyperplane („Mittelsenkrechte") between p_i and p_j.
- As Voronoi cells are intersections of half spaces, they are convex regions.



Contents

1) Introduction to clustering

2) Partitioning Methods
   - K-Means
   - Variants: K-Medoid, K-Mode, K-Median
   - Choice of parameters: Initialization, Silhouette coefficient

3) Probabilistic Model-Based Clusters: Expectation Maximization

4) Density-based Methods: DBSCAN

5) Hierarchical Methods
   - Agglomerative and Divisive Hierarchical Clustering
   - Density-based hierarchical clustering: OPTICS

6) Evaluation of Clustering Results

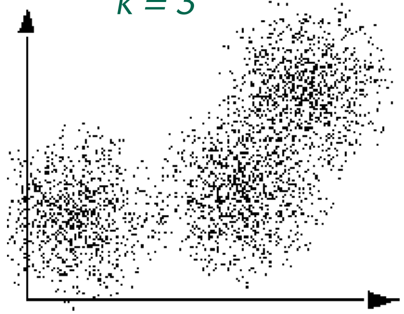
7) Further Clustering Topics
   - Scaling Up Clustering Algorithms

Initialization of Partitioning Clustering Methods

Just two examples:

[naïve]

- Choose sample A of the dataset
- Cluster the sample and use centers as initialization



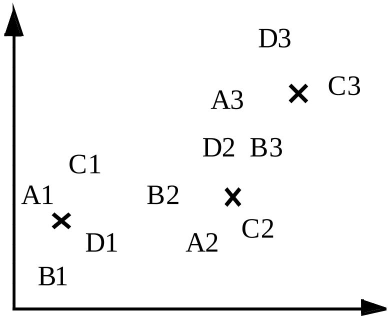
whole dataset
k = 3

[Fayyad, Reina, and Bradley 1998]

- Choose m different (small) samples $A, \ldots, M$ of the dataset
- Cluster each sample to get m estimates for k representatives $A = (A_1, A_2, \ldots, A_k)$, $B = (B_1, \ldots, B_k)$, ..., $M = (M_1, \ldots, M_k)$
- Then, cluster the set $DS = A \cup B \cup \ldots \cup M$ m times. Each time use the centers of A, B, ..., M as respective initial partitioning
- Use the centers of the best clustering as initialization for the partitioning clustering of the whole dataset

m = 4 samples A, B, C, D

× true cluster *centers*



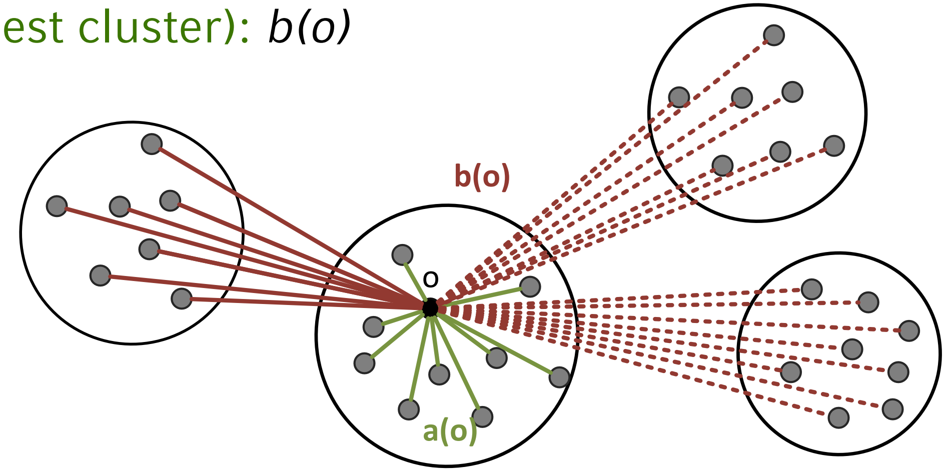
Fayyad U., Reina C., Bradley P. S., „Initialization of Iterative Refinement Clustering Algorithms“, *In KDD 1998)*, pp. 194—198.

Choice of the Parameter *k*

- Idea for a method:
  - Determine a clustering for each $k = 2, \dots, n$-1
  - Choose the “best” clustering
- But how to measure the quality of a clustering?
  - A measure should not be monotonic over *k*.
  - The measures for the compactness of a clustering SSE and TD are monotonously decreasing with increasing value of *k*.
- Silhouette-Coefficient [Kaufman & Rousseeuw 1990]
  - Measure for the quality of a *k*-means or a *k*-medoid clustering that is not monotonic over *k*.

The Silhouette coefficient (1)

- Basic idea:
  - How good is the clustering = how appropriate is the mapping of objects to clusters
  - Elements in cluster should be „similar“ to their representative
    → measure the average distance of objects to their representative: *a(o)*
  - Elements in different clusters should be „dissimilar“
    → measure the average distance of objects to alternative clusters (i.e. second closest cluster): *b(o)*



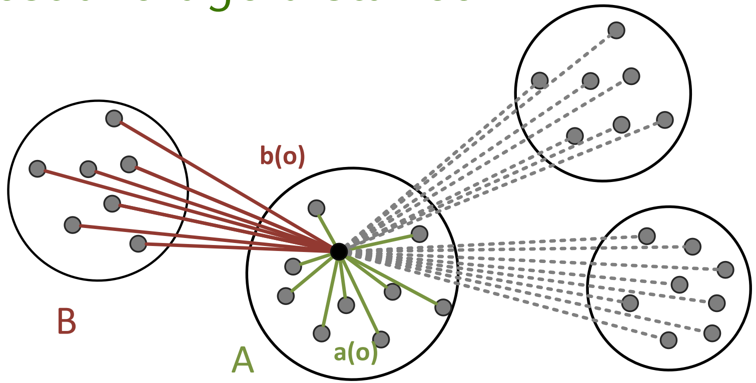
The Silhouette coefficient (2)

- $a(o)$: average distance between object o and the objects in its cluster A

$$a(o) = \frac{1}{|C(o)|} \sum_{p \in C(o)} dist(o,p)$$

- $b(o)$: for each other cluster C_i compute the average distance between o and the objects inC_i. Then take the smallest average distance

$$b(o) = \min_{C_i \neq C(o)} \left(\frac{1}{|C_i|} \sum_{p \in C_i} dist(o,p) \right)$$



- The silhouette of o is then defined as

$$s(o) = \begin{cases} 0 & if\ a(o) = 0, e.g. |C_i| = 1 \\ \frac{b(o) - a(o)}{\max\{a(o), b(o)\}} & else \end{cases}$$

- The values of the silhouette coefficient range from –1 to +1

The Silhouette coefficient (3)

- The silhouette of a cluster C_i is defined as:

$$silh(C_i) = \frac{1}{|C_i|} \sum_{o \in C_i} s(o)$$

- The silhouette of a clustering $\mathcal{C} = (C_1, \dots, C_k)$ is defined as:

$$silh(\mathcal{C}) = \frac{1}{|D|} \sum_{o \in D} s(o)\,,$$

where D denotes the whole dataset.

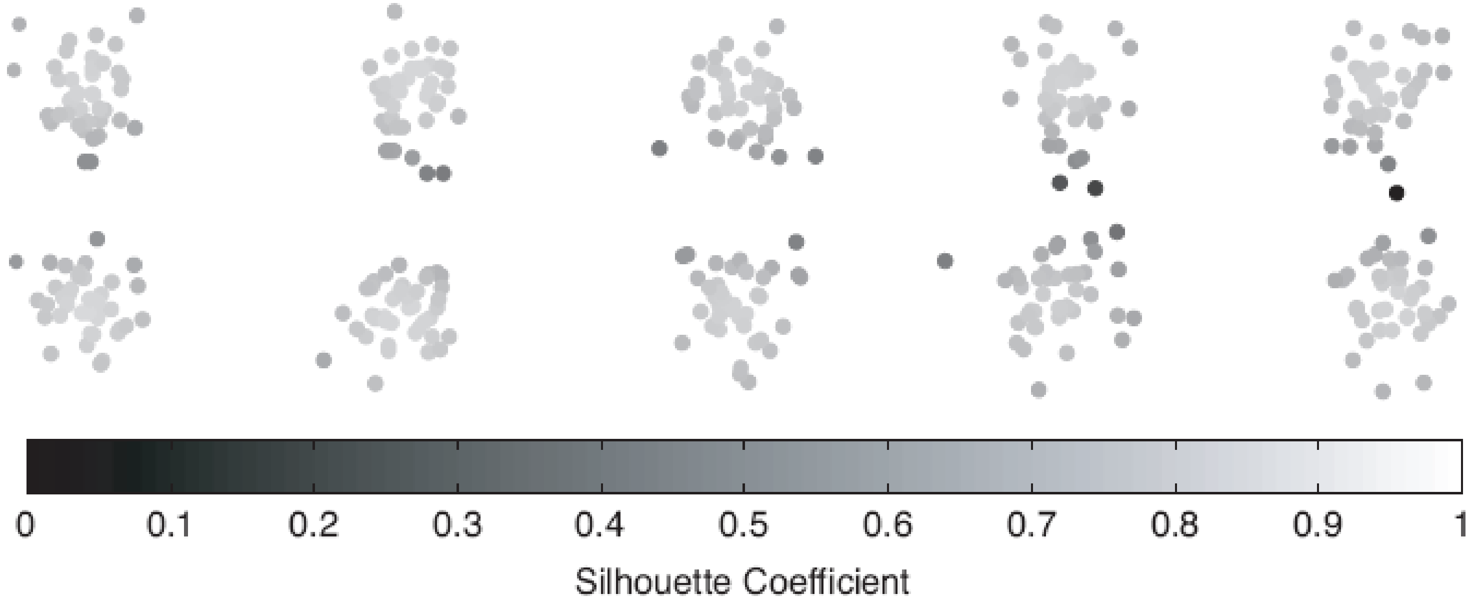
The Silhouette coefficient (4)

- „Reading“ the silhouette coefficient:
  Let $a(o) \neq 0$.
  - $b(o) \gg a(o) \Rightarrow s(o) \approx 1$: good assignment of *o* to its cluster *A*
  - $b(o) \approx a(o) \Rightarrow s(o) \approx 0$: o is in-between *A* and *B*
  - $b(o) \ll a(o) \Rightarrow s(o) \approx -1$: bad, on average o is closer to members of *B*

- Silhouette Coefficient $s_{\mathcal{C}}$ of a clustering: average silhouette of all objects
  - $0.7 < s_C \leq 1.0$ strong structure, $0.5 < s_C \leq 0.7$ medium structure
  - $0.25 < s_C \leq 0.5$ weak structure, $s_C \leq 0.25$ no structure

The Silhouette coefficient (5)

Silhouette Coefficient for points in ten clusters



in: Tan, Steinbach, Kumar: Introduction to Data Mining (Pearson, 2006)