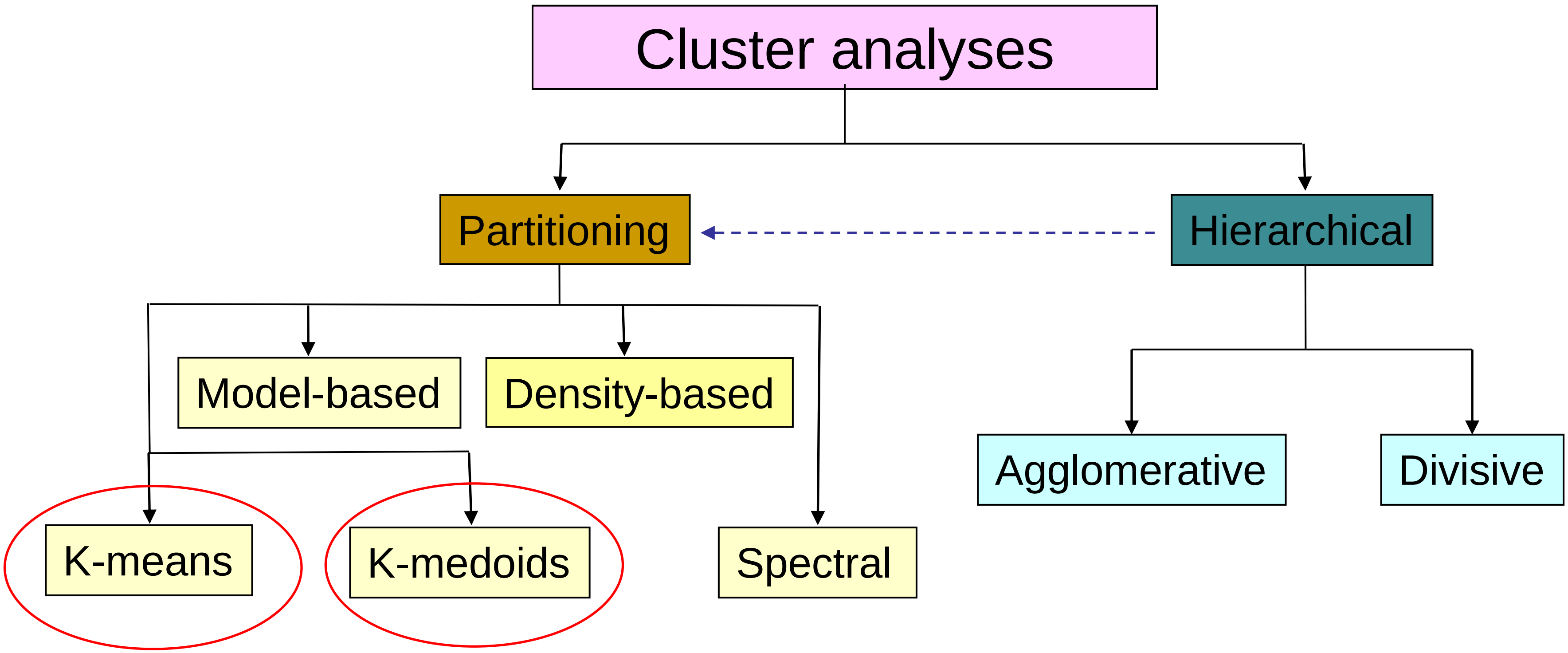
Cluster analysis I

Introduction, k-means, k-medoids, k-medians

Radoslav Harman
KAMS FMFI UK

Structure of cluster analyses



Note: partitioning methods can be „hard“ or „soft (fuzzy)“

Applications: Image segmentation, Recommender systems, Anomaly detection, Identification of groups in social networks, Market research, Medical imagining, Categorization of astronomic objects,…

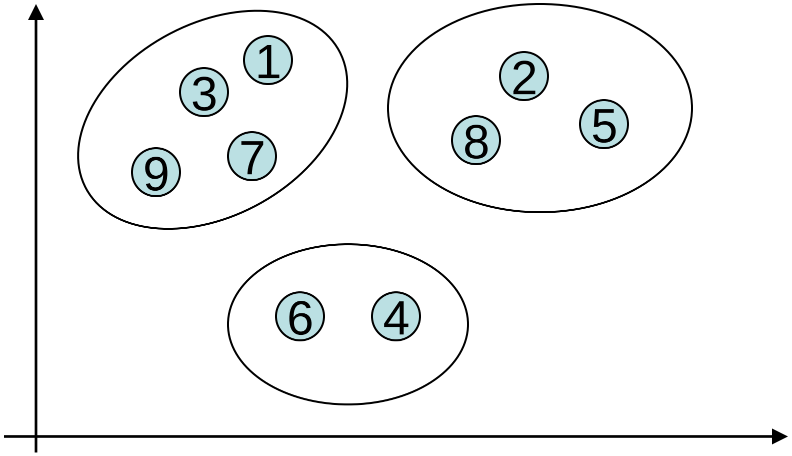
Partitioning cluster analysis

Finds a decomposition of objects 1,...,n into k disjoint clusters $C_1,\ldots,C_k$ of „similar“ objects:

$$C_1 \cup \ldots \cup C_k = \{1,\ldots,n\},\ i \neq j \Rightarrow C_i \cap C_j = \emptyset$$

The objects are (mostly) characterized by „vectors of features“ $x_1,\ldots,x_n \in \Re^p$

p=2
k=3
n=9



$$C_1 = \{1,3,7,9\} \quad |C_1| = 4$$

$$C_2 = \{2,5,8\} \quad |C_2| = 3$$

$$C_3 = \{4,6\} \quad |C_3| = 2$$

How do we understand „decomposition into clusters of similar objects“?

How is this decomposition calculated?

Many different principles and algorithms...

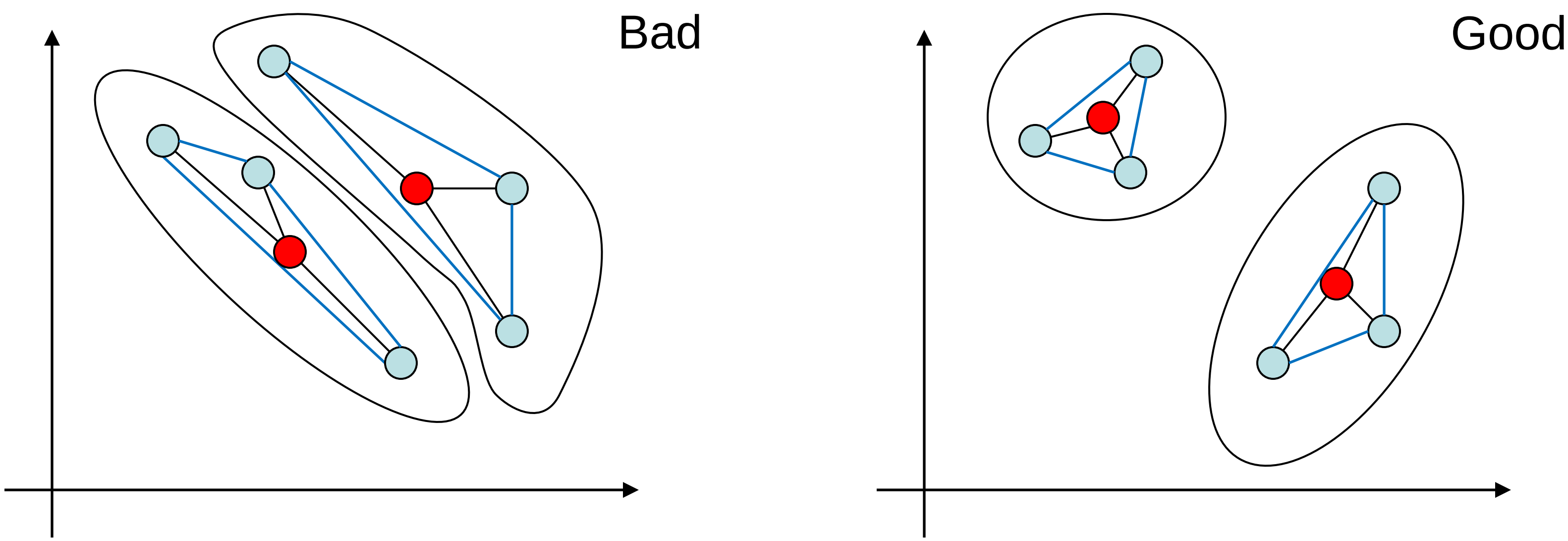
K-means clustering

The objective function to be minimized with respect to the selection of clusters is the „**within-cluster sum of squares**“:

$$\sum_{i=1}^{k}\sum_{r\in C_i}\rho^2(x_r,c_i) \quad \text{where} \quad c_i=\frac{1}{|C_i|}\sum_{r\in C_i}x_r \quad \text{is the centroid of } C_i,$$

ρ is the Euclidean distance. Equivalent objective function is the „**sum of average pairwise squared deviations**“:

$$\sum_{i=1}^{k}\frac{1}{|C_i|}\sum_{x,y\in C_i}\rho^2(x,y)$$



K-means clustering

Computing the clustering that minimizes the k-means objective function is a difficult problem. Nevertheless, there are many efficient heuristics able to find a „good“ (not always optimal) solution, such as:

Lloyd’s Algorithm

- Create a random initial clustering $C_1,...,C_k$.
- Until a maximum number of iterations is reached, or no reassignment of objects occurs do:
  - Calculate the centroids $c_1,...,c_k$ of clusters.
  - For every i=1,...,k :
    - Form the new cluster C_i from all the points that are closer to c_i than to any other centroid.

Illustration of the Lloyds' algorithm

Choose an initial clustering

p=2

k=3

n=11

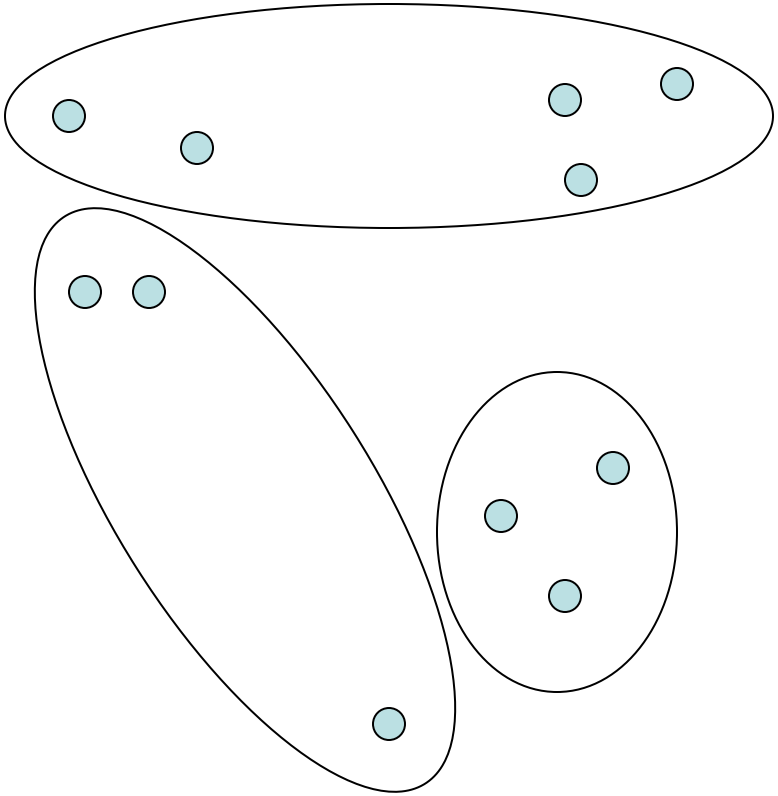


Illustration of the Lloyds' algorithm

Calculate the centroids of clusters

p=2

k=3

n=11

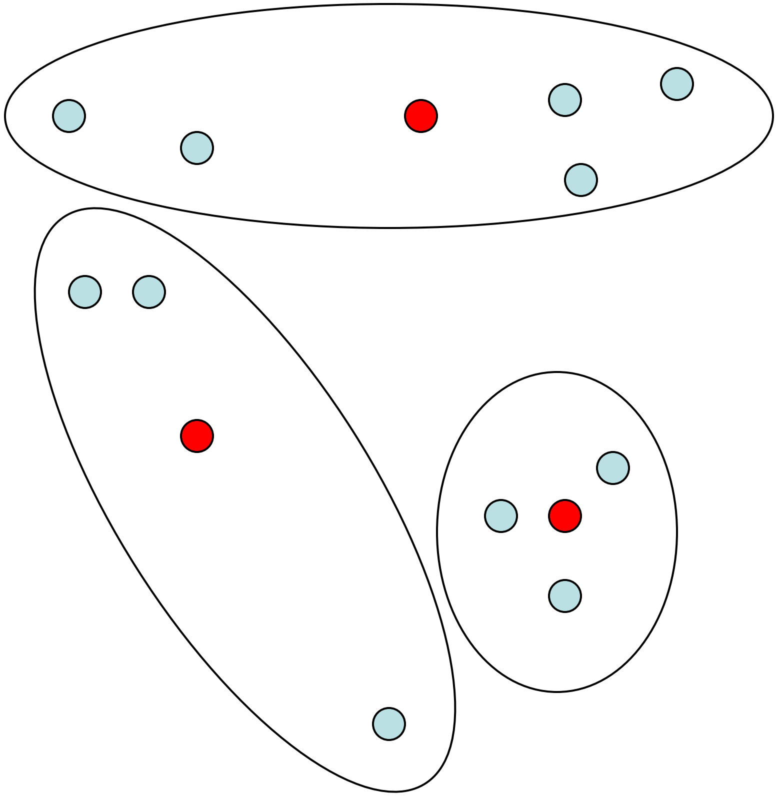


Illustration of the Lloyds' algorithm

Calculate the centroids of clusters

p=2

k=3

n=11

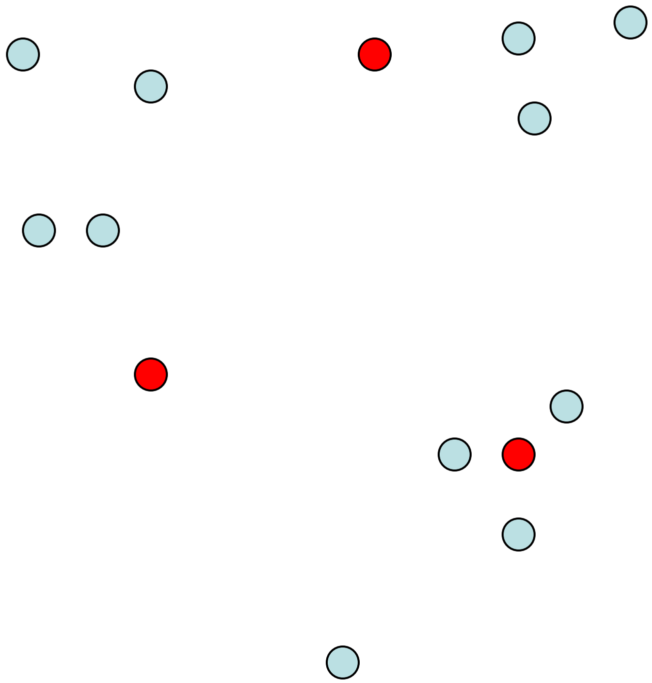


Illustration of the Lloyds' algorithm

Assign the points to the closest centroids

p=2

k=3

n=11

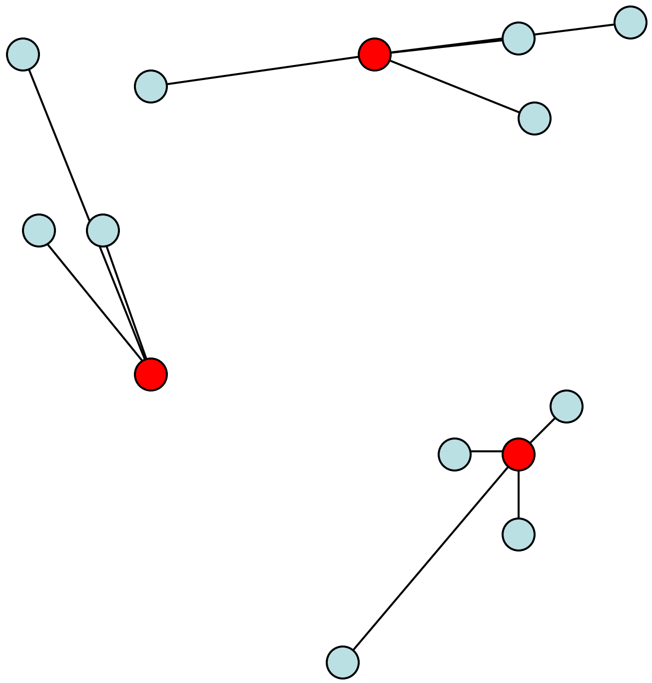


Illustration of the Lloyds' algorithm

Create the new clustering

p=2

k=3

n=11

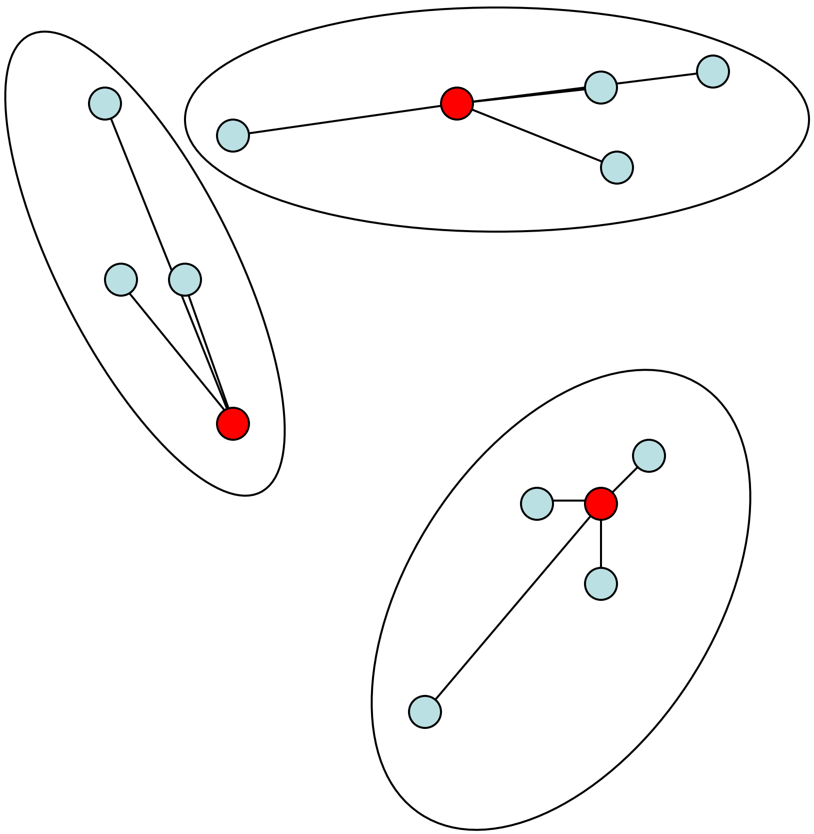


Illustration of the Lloyds' algorithm

Create the new clustering

p=2

k=3

n=11

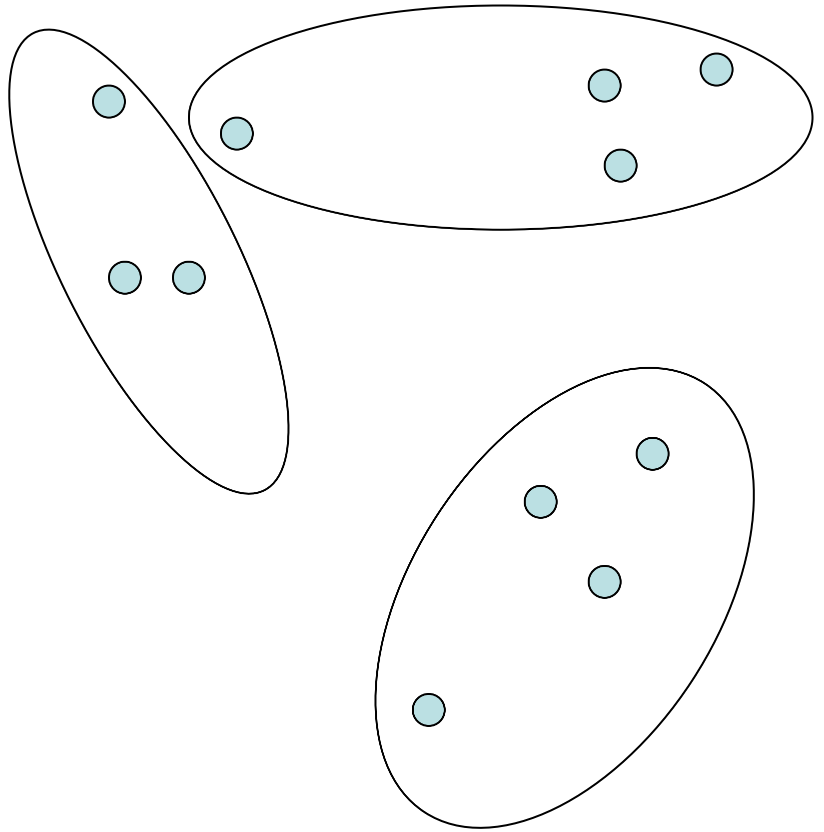


Illustration of the Lloyds' algorithm

Calculate the new centroids of clusters

p=2
k=3
n=11

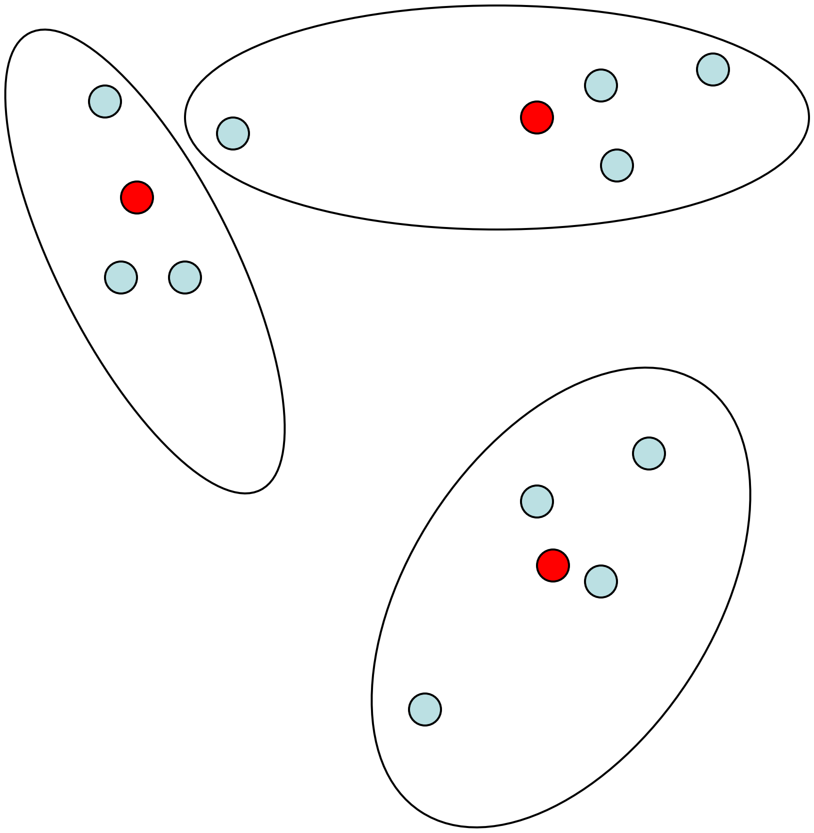


Illustration of the Lloyds' algorithm

Calculate the new centroids of clusters

p=2

k=3

n=11

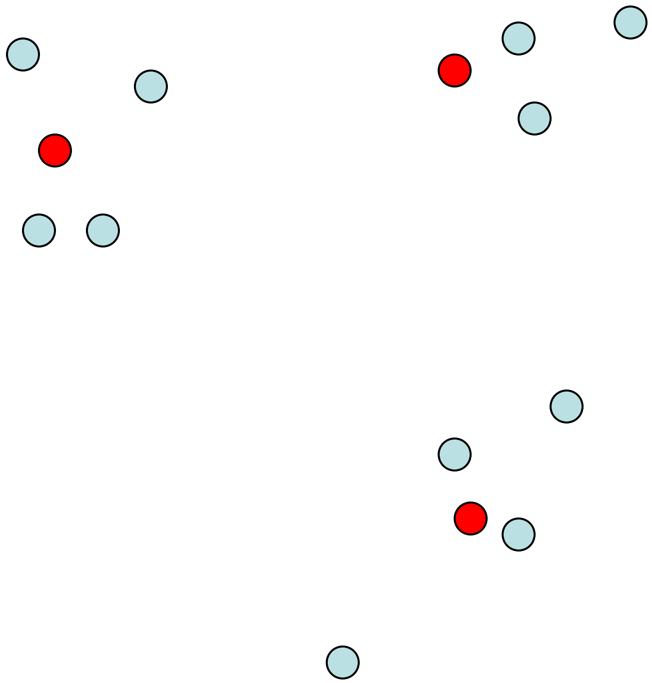


Illustration of the Lloyds' algorithm

Assign the points to the closest centroids

p=2

k=3

n=11

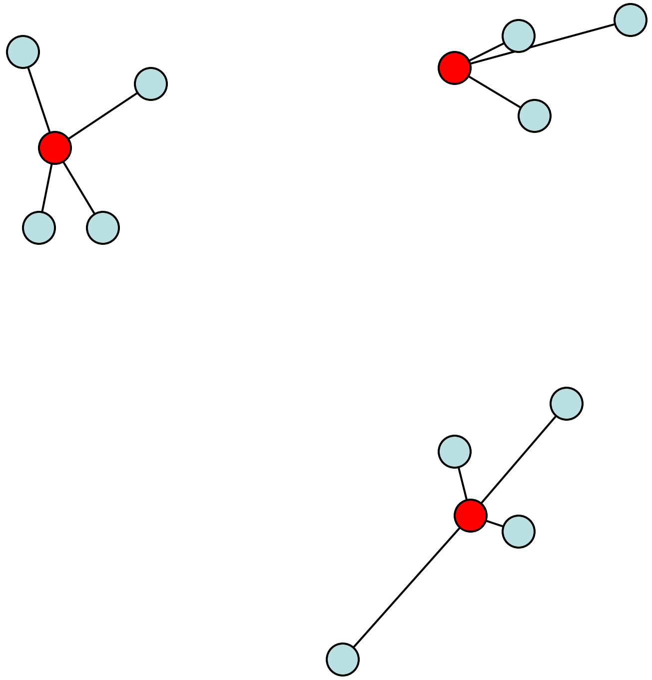


Illustration of the Lloyds' algorithm

Create the new clustering

p=2

k=3

n=11

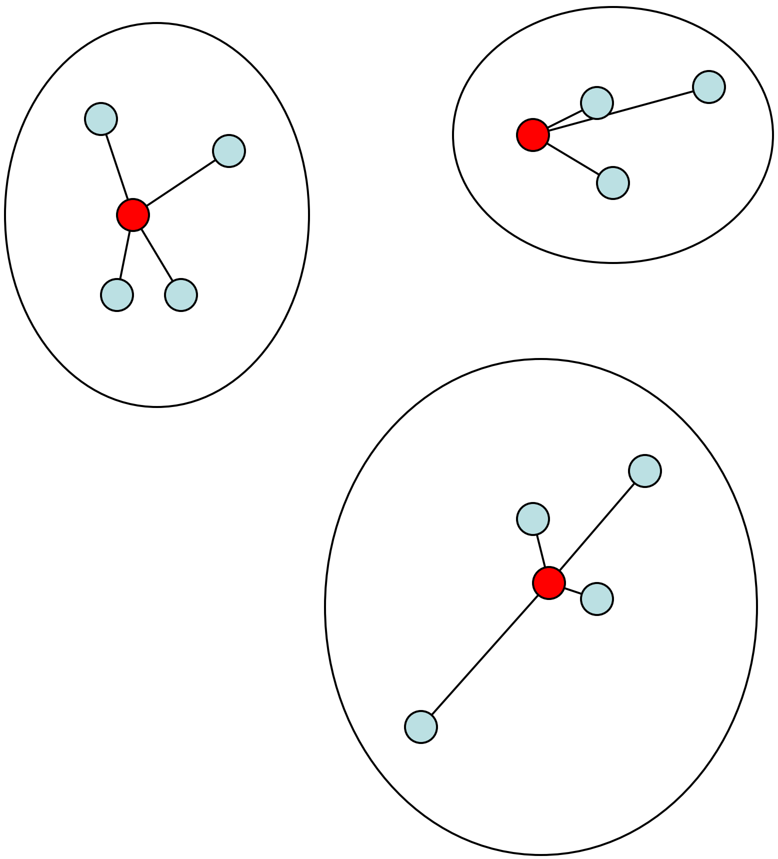


Illustration of the Lloyds' algorithm

Create the new clustering

p=2

k=3

n=11

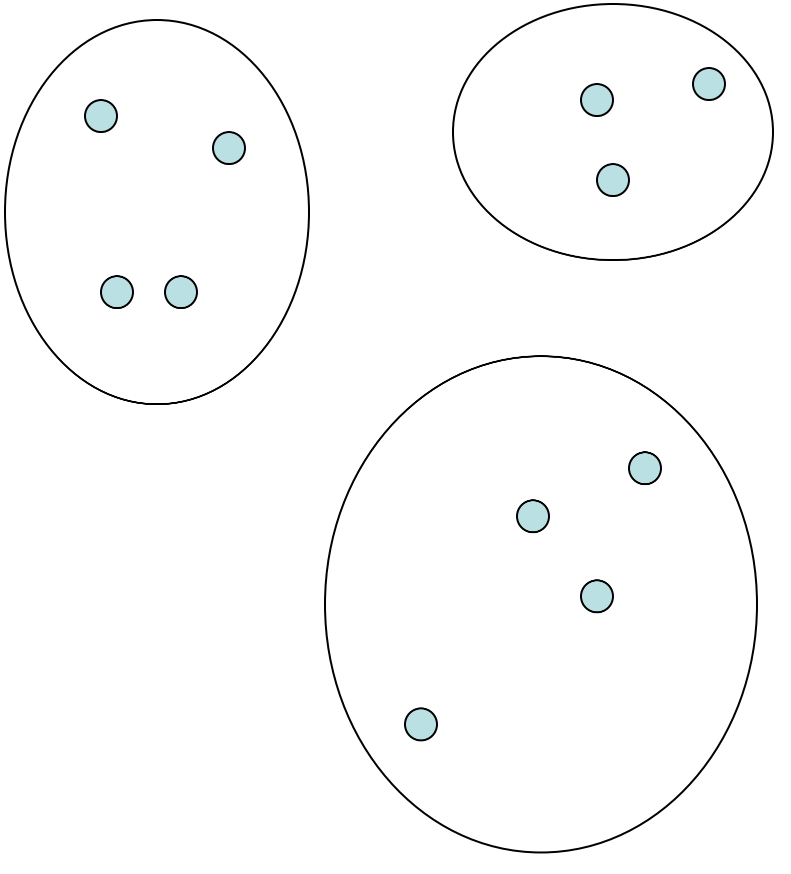


Illustration of the Lloyds' algorithm

Calculate the new centroids of clusters

p=2
k=3
n=11

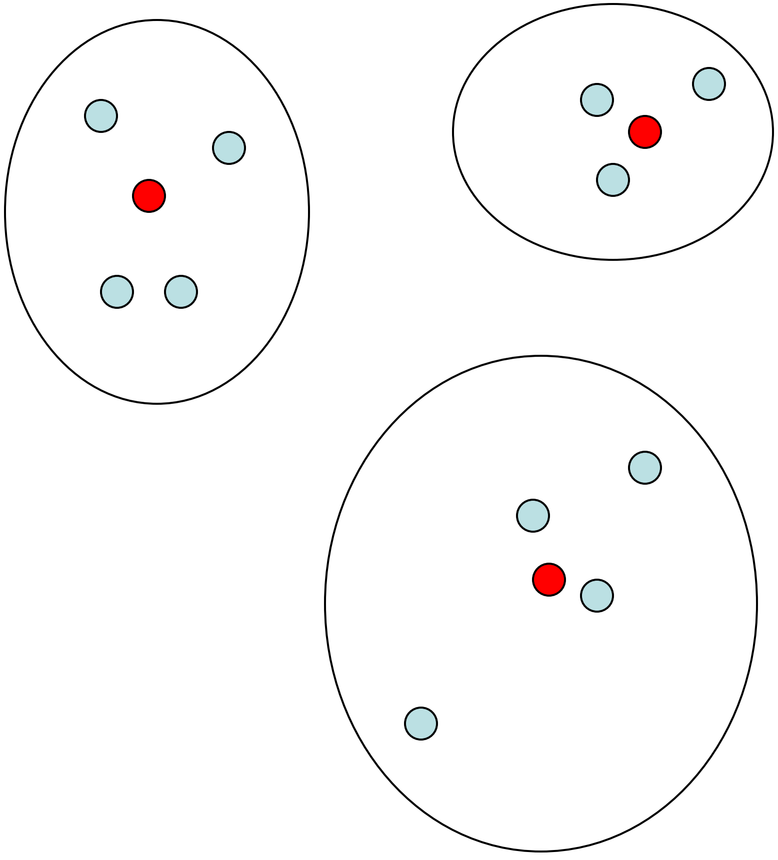


Illustration of the Lloyds' algorithm

Calculate the new centroids of clusters

p=2

k=3

n=11

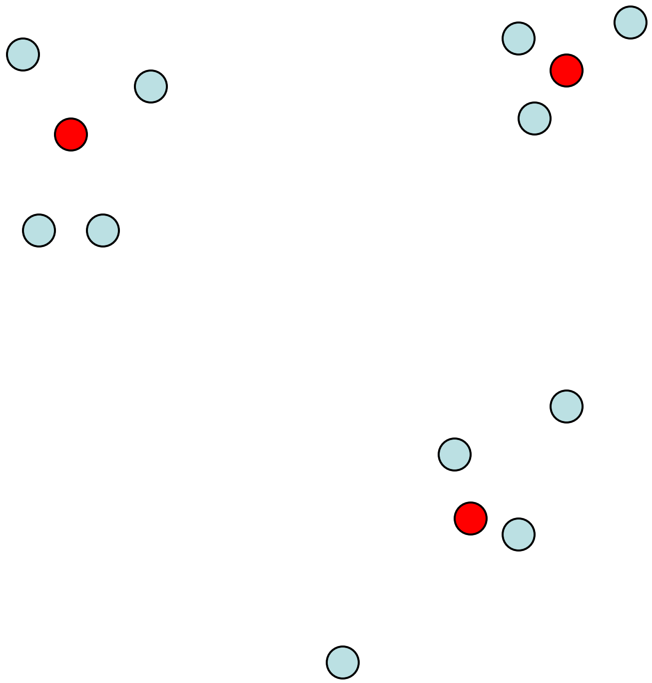


Illustration of the Lloyds' algorithm

Assign the points to the closest centroids

p=2

k=3

n=11

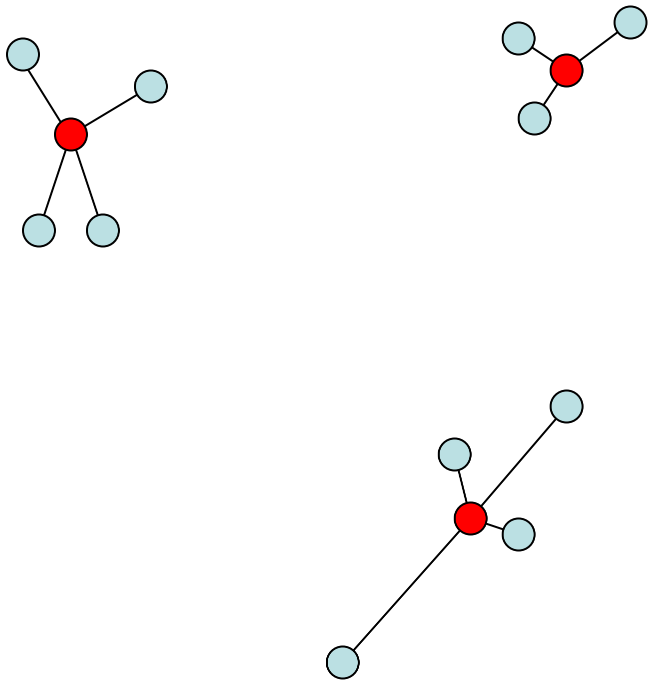


Illustration of the Lloyds’ algorithm

Create the new clustering

p=2

k=3

n=11

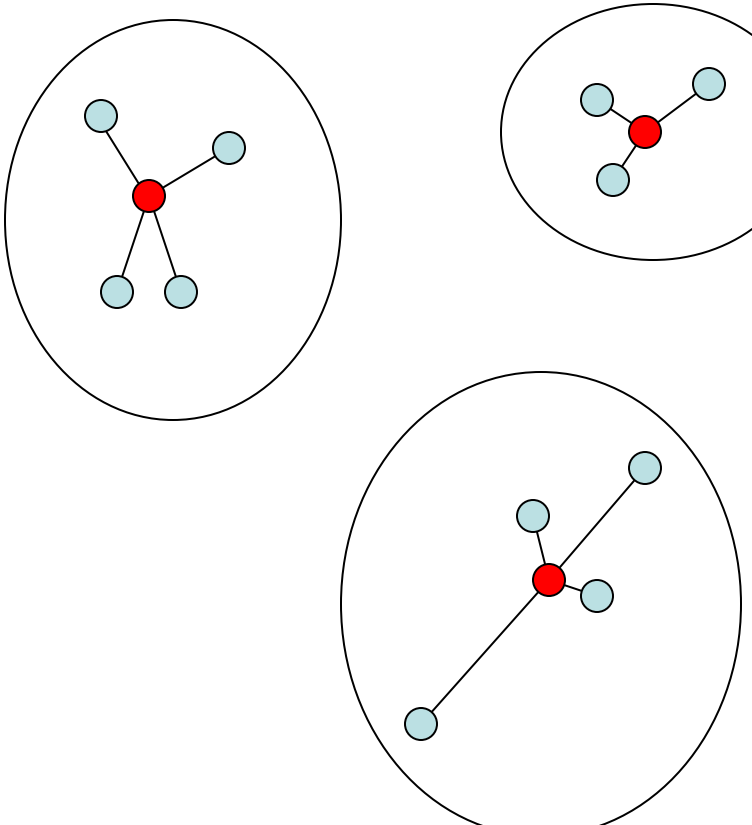
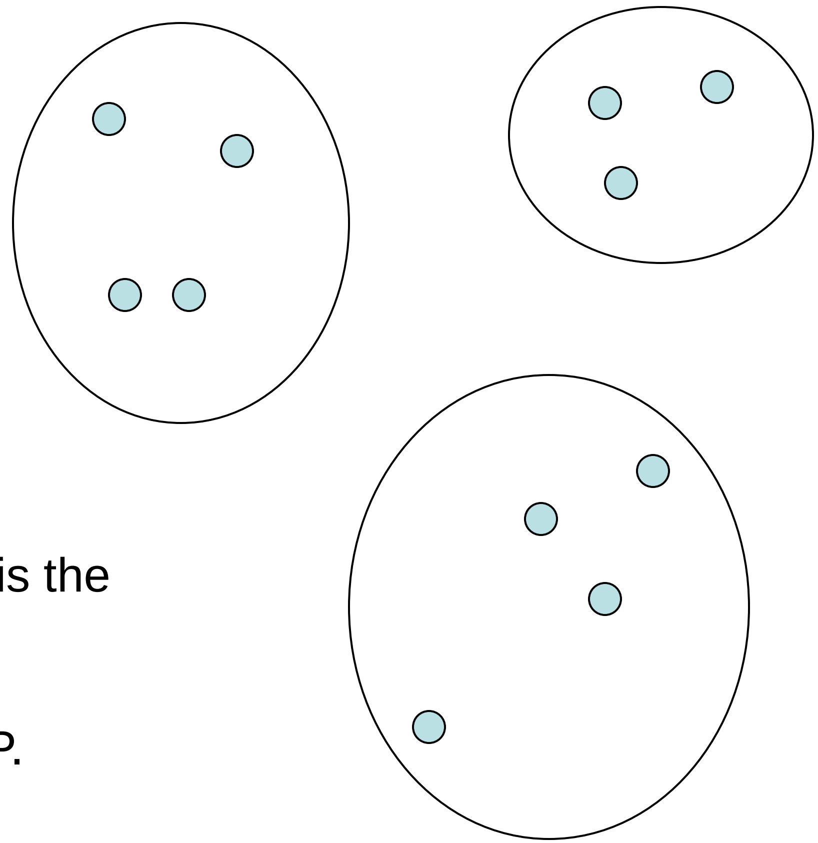


Illustration of the Lloyds' algorithm

Create the new clustering

p=2
k=3
n=11



The clustering is the same as in the previous step, therefore STOP.

Properties of the k-means as a method

+ The principle is simple to understand
+ Many efficient heuristic methods (some better than the Lloyds')

- The number k of clusters must be given in advance
- The resulting clustering depends on the units of measurement
- The variables must be real vectors („dissimilarities" are not enough)
- Not suitable for finding clusters with nonconvex shapes
- The optimization problem itself is very difficult (NP hard)

Properties of the Lloyds' algorithm

+ Simple to understand and the naive implementation is trivial
+ Reasonably fast in practice (always convergent in a finite number of steps)
+ Usually converges to a "good" solution in practice

- Worst case super-polynomial in the input size
- Different initial clusterings can lead to different final clusterings. The result can be arbitrarily bad compared to the true optimum. This is ameliorated by the **k-means++** initialization

Computation of k-means in R

In R (library `stats`):

```
kmeans(x, centers, iter.max, nstart, algorithm)
```

Dataframe of real vectors of features

The number of clusters or a set of centers

Maximum number of iterations

Number of restarts

The method used ("Hartigan-Wong", "Lloyd", "Forgy", "MacQueen")

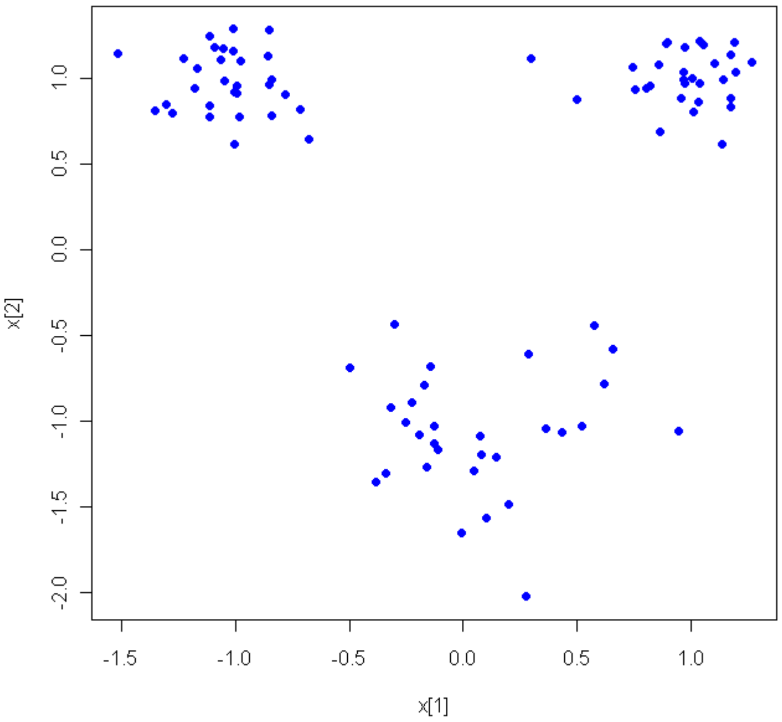
Many R packages deal with clustering, e.g.: `cluster`, `clusterR`, `flexclust`

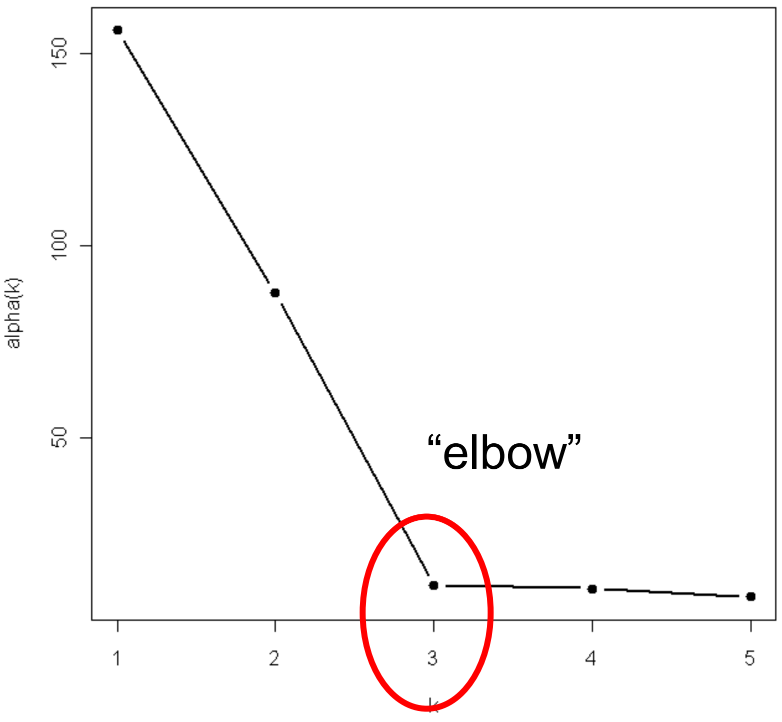
The “elbow” method to determine k

$$\alpha(k) = \sum_{i=1}^{k} \sum_{r \in C_i^{(k)}} \rho^2\left(x_r - c_i^{(k)}\right)$$

$C_1^{(k)}, \ldots, C_k^{(k)}$ … optimal clustering obtained by assuming k clusters

$c_1^{(k)}, \ldots, c_k^{(k)}$ … corresponding centroids





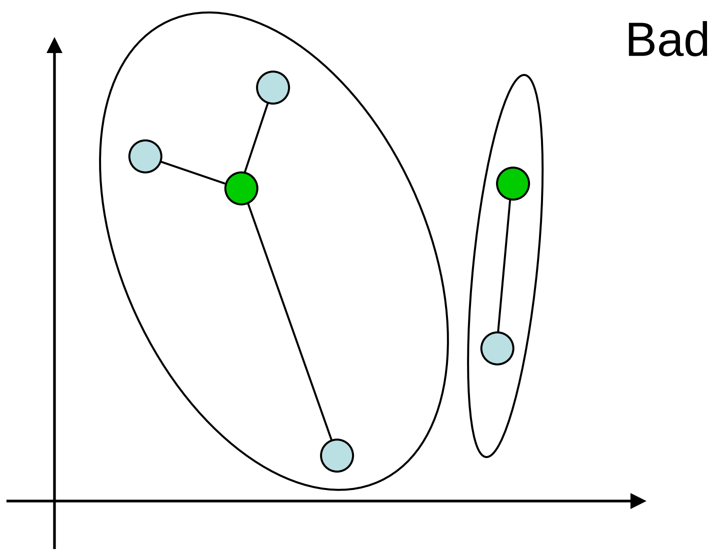
K-medoids clustering

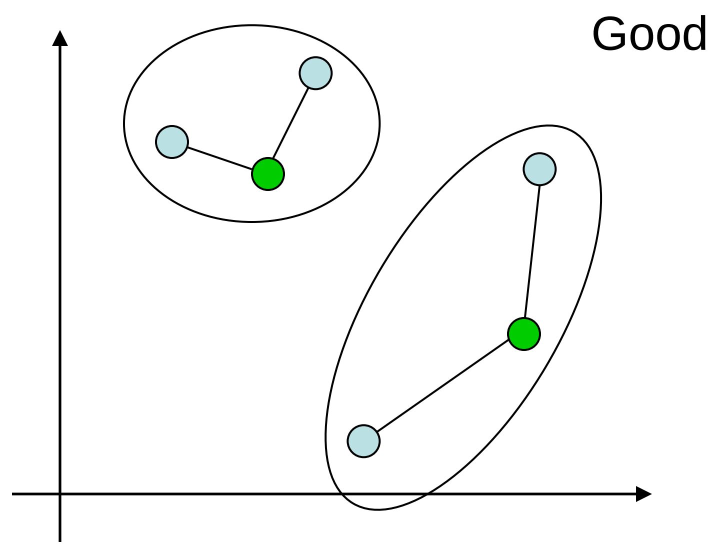
Instead of centroids uses „**medoids**“ – the most central objects (the „best representatives“) of each cluster.

This allows using only „**dissimilarities**“ $d(r,s)$ of all pairs (r,s) of the objects.

The aim is to find the clusters $C_1,\ldots,C_k$ that minimize the objective function:

$\sum_{i=1}^{k}\sum_{r\in C_i} d(r,m_i)$ where for each i the medoid m_i minimizes $\sum_{r\in C_i} d(r,m_i)$





K-medoids algorithm

k-medoids as an optimization problem is difficult. There are many efficient heuristics that find a „good“ (although not always optimal) solution. Example:

„**Partitioning around medoids**“ (PAM)

- (BUILD) „Greedily“ select k objects $m_1,...,m_k$ as initial medoids.
- (SWAP) Until the maximum number of iterations is reached or no improvement of the target function has been found do:
  1. Calculate the clustering based on $m_1,...,m_k$ by associating each point to the nearest medoid and calculate the value of the target function.
  2. For all pairs (m_i , x_s), where x_s is a non-medoid point, try to improve the target function by taking x_s to be a new medoid point and m_i to be a non-medoid point.
  3. Stop, if no exchange from Step 2 improves the objective function
  4. Realize the best possible exchange from Step 2

Alternative algorithm: “**Voronoi iteration**” (similar to the Lloyd’s method)

Comparison of k-medoids to k-means

Many general properties of k-medoids are the same as k-means (see the list of properties for k-means). Differences of k-medoids and k-means include:

+ k-medoids allows using general dissimilarities d of objects
+ If d is the Euclidean distance, k-medoids is less sensitive to outliers
+ The result is a list of medoids, i.e., a list of „representative objects"

- Fewer algorithms for computing, less available theory

Computational issues of k-medoids

In R (library `cluster`):

```
pam(x, k, diss, metric, medoids, stand,…)
```

- `x`: Dataframe of real vectors of features or a matrix of dissimilarities
- `k`: The number of clusters
- `diss`: Is `x` a dissimilarity matrix? (`TRUE`, `FALSE`)
- `metric`: Metrics used (`euclidean`, `manhattan`)
- `medoids`: Vector of initial medoids
- `stand`: Standardize data? (`TRUE`, `FALSE`)

The silhouette

$a(r)$ … the average dissimilarity of the object r and the objects of the same cluster

$b(r)$ … the average dissimilarity of the object r and the objects of the "neighboring" cluster

"Silhouette" of the object r … the measure of "how well" is r "clustered"

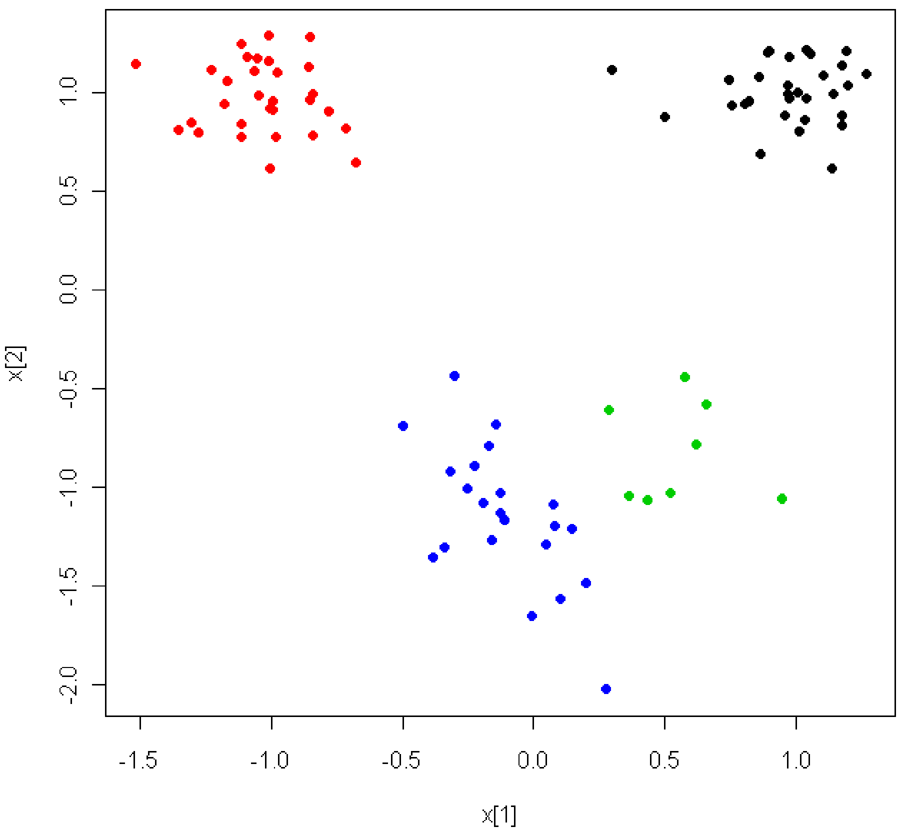
$$s(r) = \frac{b(r) - a(r)}{\max\left(b(r), a(r)\right)} \in [-1,1]$$

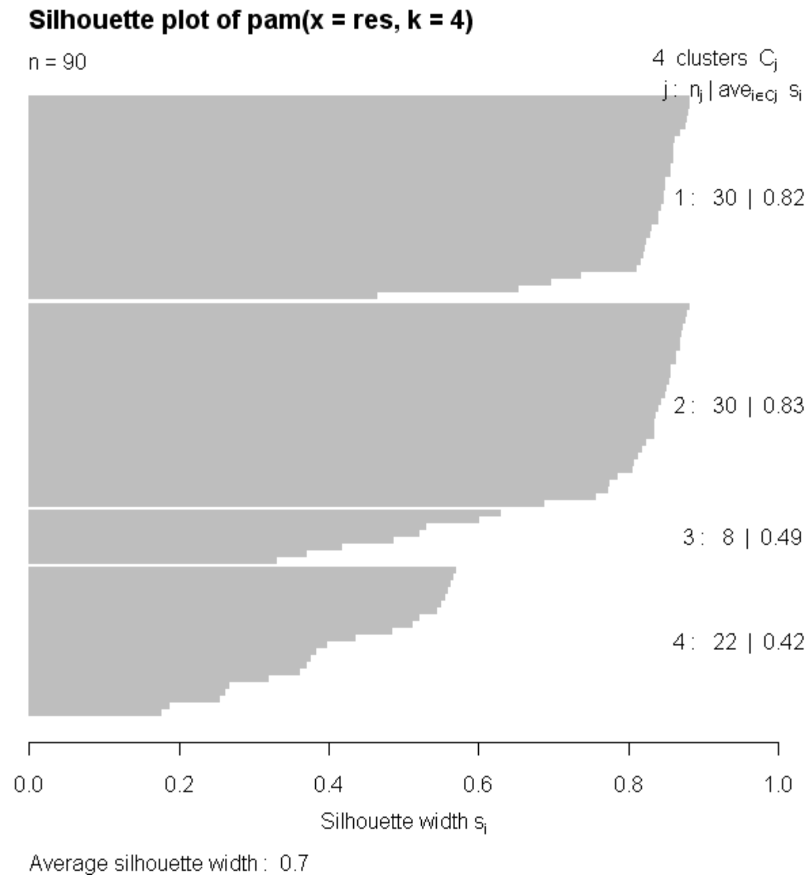
$s(r)$ close to 1 … the object r is well clustered

close to 0 … the object r is at the boundary of clusters

less than 0 … the object r is probably placed in a wrong cluster

The silhouette





K-medians clustering

k-medians is a rarely used approach (which is moreover understood differently in different sources). In general, it minimizes the objective

$$\sum_{i=1}^{k} \sum_{r \in C_i} \rho(x_r, m_i)$$ where m_i is „a“ median of C_i, ρ is „a“ distance

k-medians can be specified for instance as:

A) Geometric median + Euclidean distance
B) Manhattan („taxicab“, „l1“) median + Manhattan distance