Data Mining
(Mining Knowledge from Data)

Clustering

Marcel Jiřina, Pavel Kordík

Outline of today's lecture

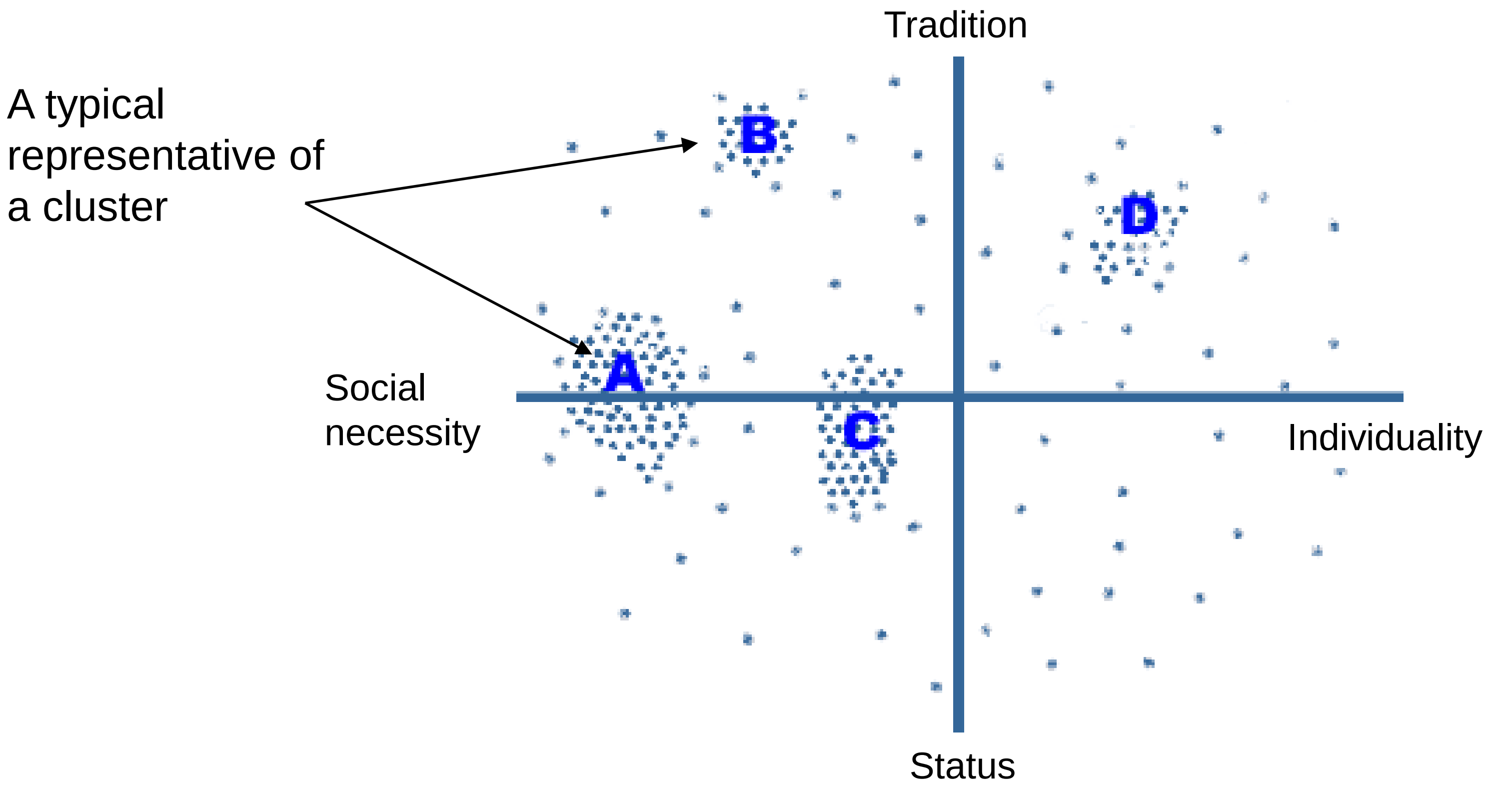
- Metrics
- Hierarchical clustering
  - Algorithms
  - Dendrograms
- K-means

Cluster analysis

- We have data, but we do not know the category (class)
- We want to find sets of similar patterns, which are also dissimilar from patterns of other sets.
- We solve an optimization problem!
- What are our unknown parameters?
  - Number of clusters
  - Assignment of data (patterns) into clusters

Clusters, representatives

- The results of the survey, why people drink alcohol



- The task of cluster analysis is to find clusters in the data, or to assign them typical representatives

Cluster analysis

- Classical cluster analysis is a tool for disjoint decomposition of a set of patterns in the input space $\boldsymbol{R}^n$ into H > 1 classes (clusters).

- Cluster analysis requires maximum similarity of patterns within a class, while the maximum dissimilarity of patterns of different classes.

- For this we need to define the similarity of two models - the **distance**

Metrics

- Metrics must meet certain conditions :
  - $d(x,y) \geq 0$
  - $d(x,y) = 0 \; iff \; x = y$
  - $d(x,y) = d(y,x)$
  - $d(x,y) \leq d(x,z) + d(z,y)$

Triangular inequality

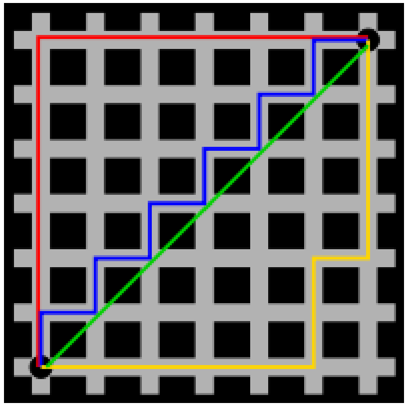
Euclidean distance

- Two points in n-dimensional space
  - $P = (p_1, p_2, \dots, pn)$
  - $Q = (q_1, q_2, \dots, qn)$

- Euclidean distance between points *P* and *Q* :
  - $e(P,Q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \cdots + (p_n - qn)^2} = \sqrt{\sum_{i=1}^{n} (p_i - qi)^2}$
- Often used $e^2(P,Q)$
  - Euclidean distance without the square root

Manhattan distance

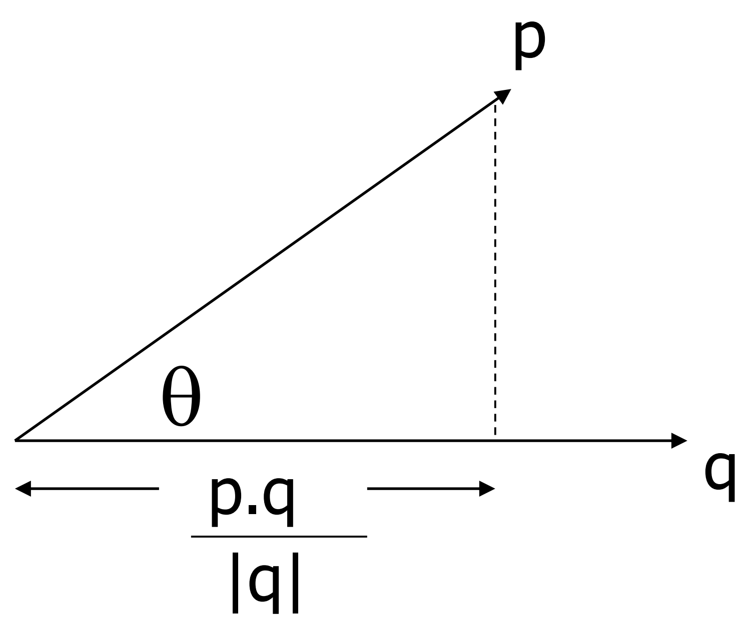
- How do we calculate the distance between two cyclists in Manhattan?




- $M(P, Q) = |p_1 - q_1| + |p_2 - q_2| + \cdots + |p_n - qn|$

Cosine distance

- It is invariant under rotation



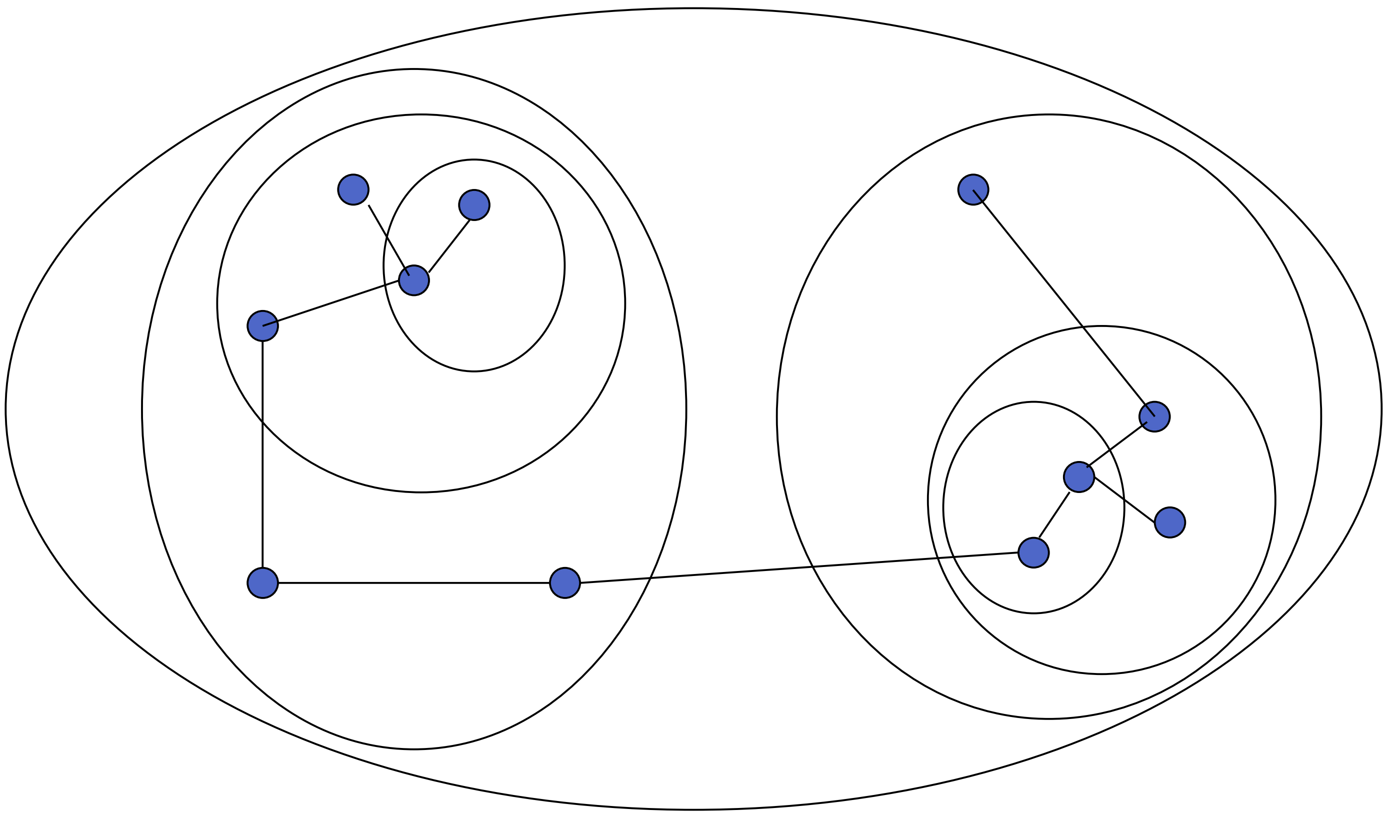
- $dist(P,Q) = \theta = \arccos\left(\frac{P \cdot Q}{\|P\| \cdot \|Q\|}\right)$

Edit distance

- E.g. to determine the distance of two words
- It is calculated as the number of deletions (insertions) of characters, needed to transform one word to another.

Hierarchical clustering

- How would you solve the problem?
- We always connect two closest vectors (points)

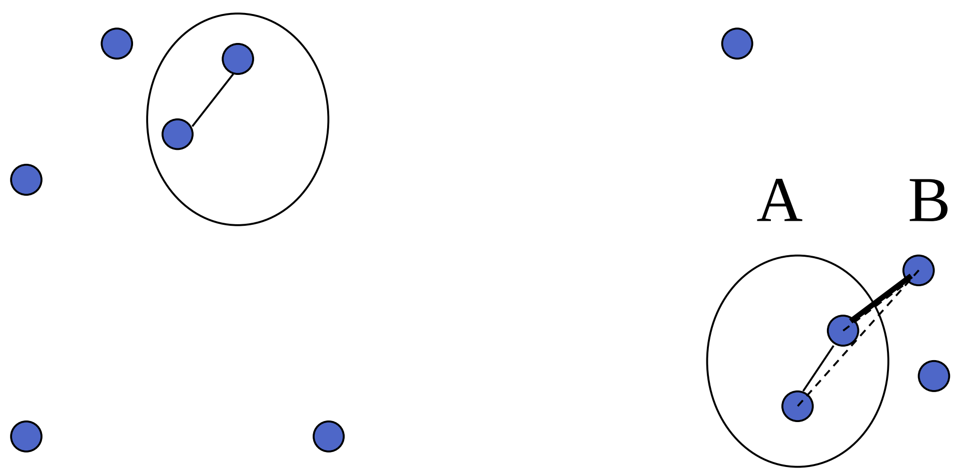


Methods for evaluating distances of clusters

- Nearest neighbor method (**single linkage**) – the distance of clusters is determined by the distance between the two closest objects (patterns) from different clusters
- Farthest neighbor method (**complete linkage**) – the distance of clusters is determined by the distance between two outermost objects from different clusters
- **Centroid linkage** – the distance of clusters is determined by the distance between their centers
- **Average linkage** – the distance of clusters is determined as the average of the distances of all pairs of objects from different clusters
- **Ward's linkage** – at each step it finds the pair of clusters that leads to minimum increase in total within-cluster variance after merging. This increase is a weighted squared distance between cluster centers.

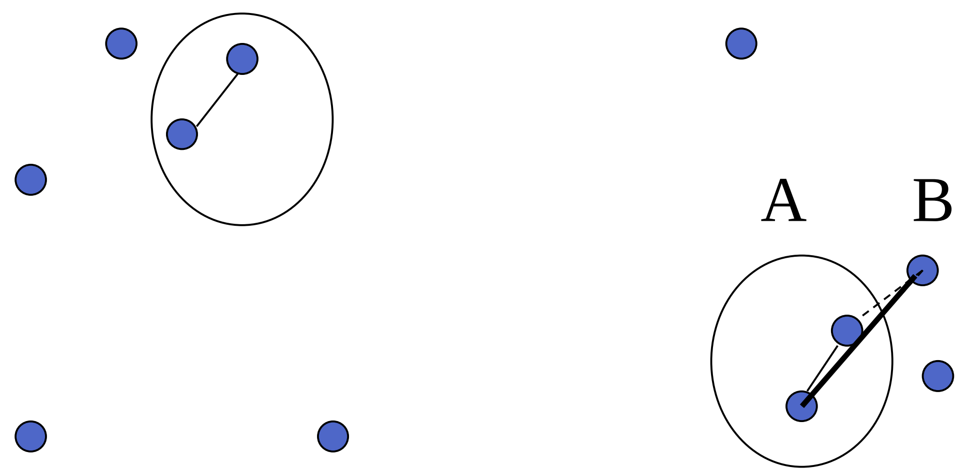
Single linkage

- The nearest pattern is always chosen from the cluster



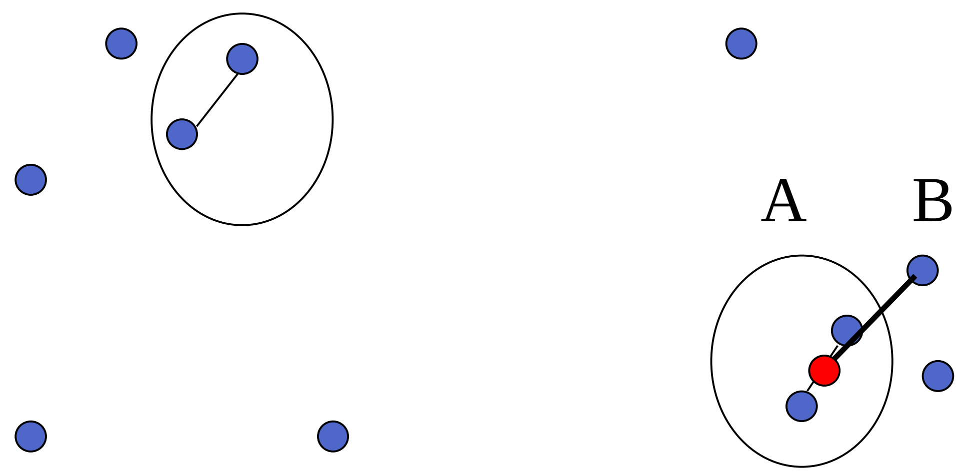
Complete linkage

- The furthest pattern is always chosen from the cluster



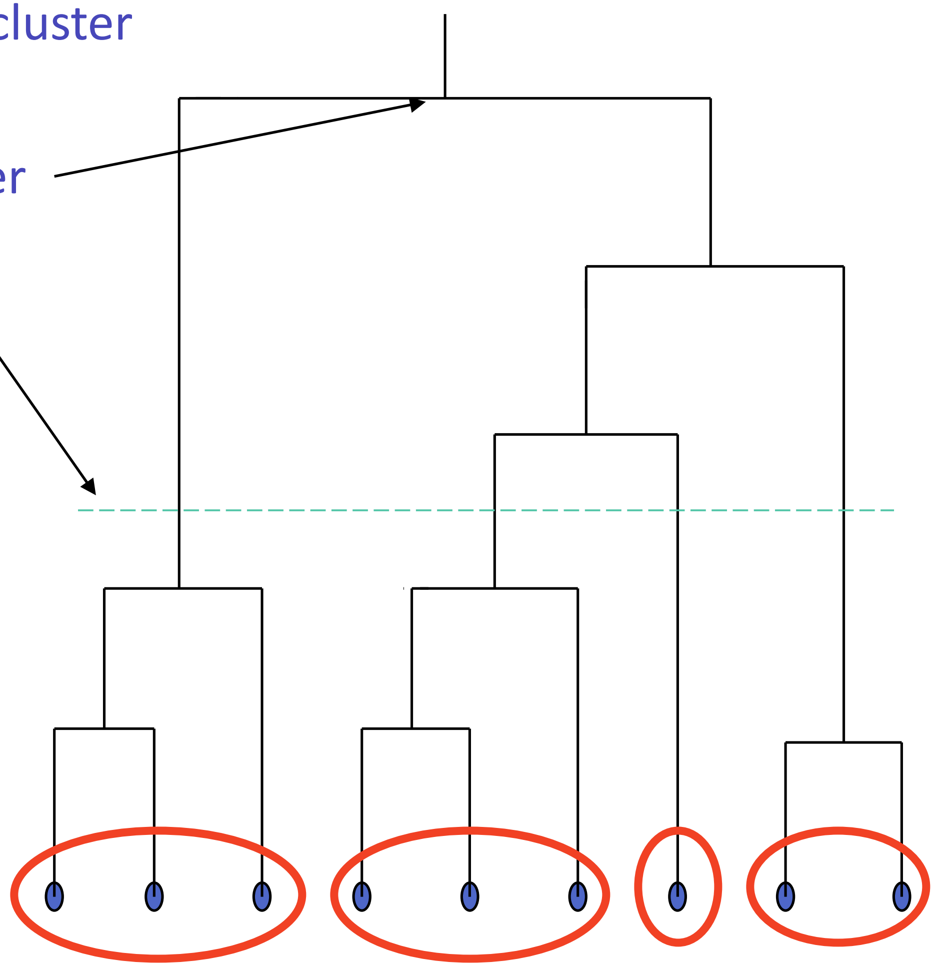
Centroid linkage

- The representative of the cluster is the centroid



How many clusters did we find?

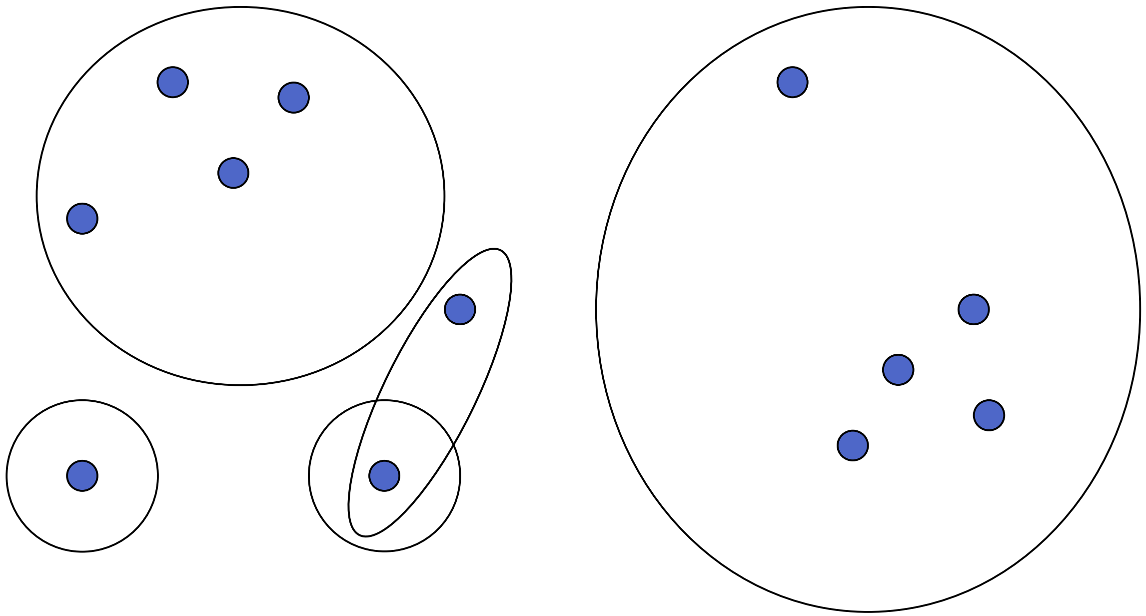
- Another view on our algorithm:
  - At the beginning each vector is a cluster
  - Linking vectors into clusters
  - At the end we get one large cluster
  - Number clusters do we have?
- The **dendrogram** =>
- The algorithm is called: **hierarchical clustering**



Just for illustration reasons, does not correspond to previous examples

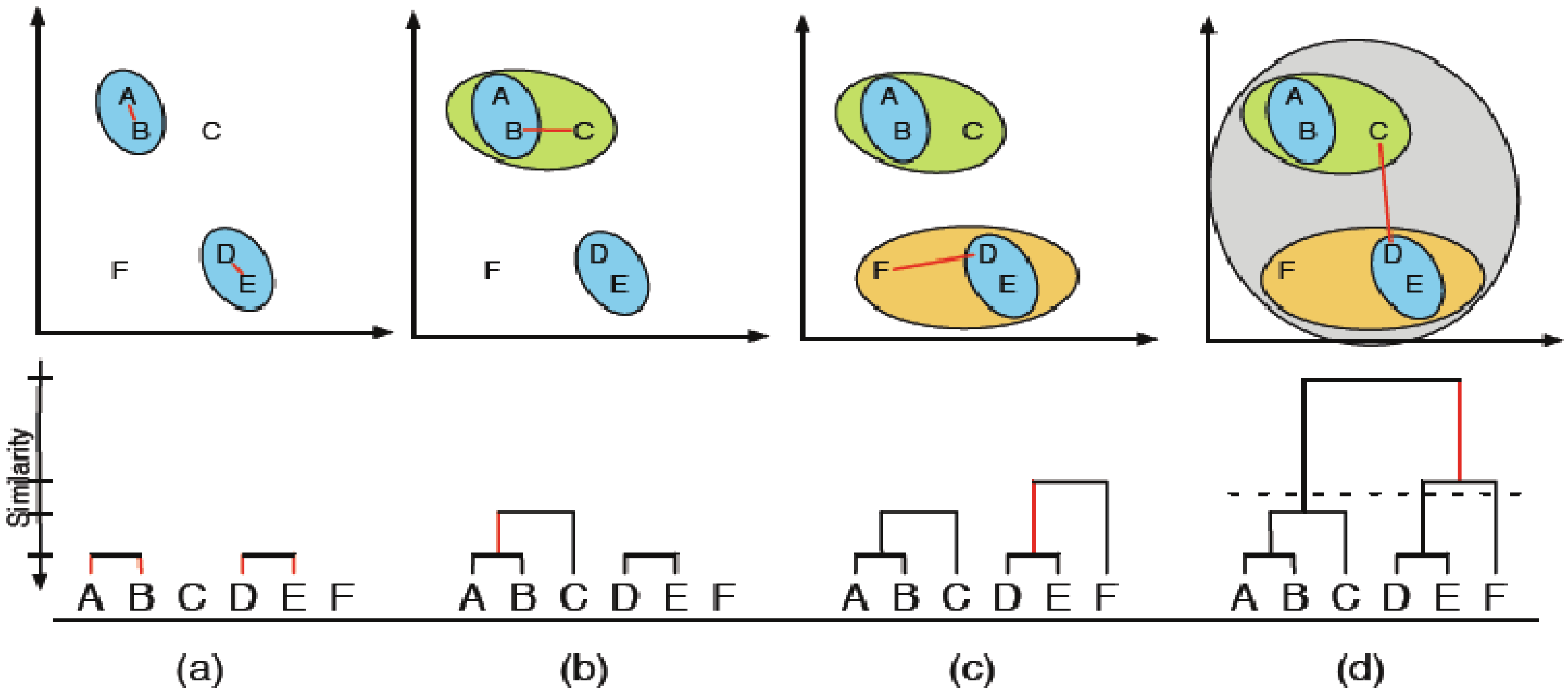
How many clusters did we find?

- It depends on where we made a “cut” of the dendrogram
- What happens when we cut the dendrogram at lower / higher level?
- Where belongs the new vector?

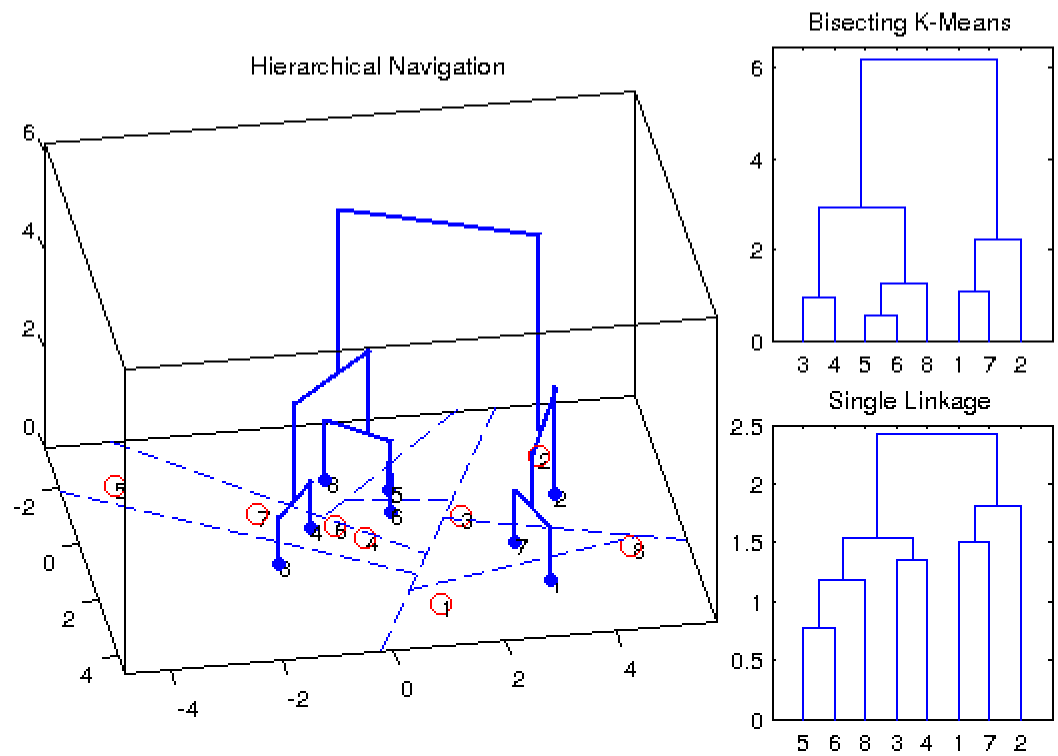


- The problem? I need to calculate the distances to all the vectors!

Hierarchical clustering

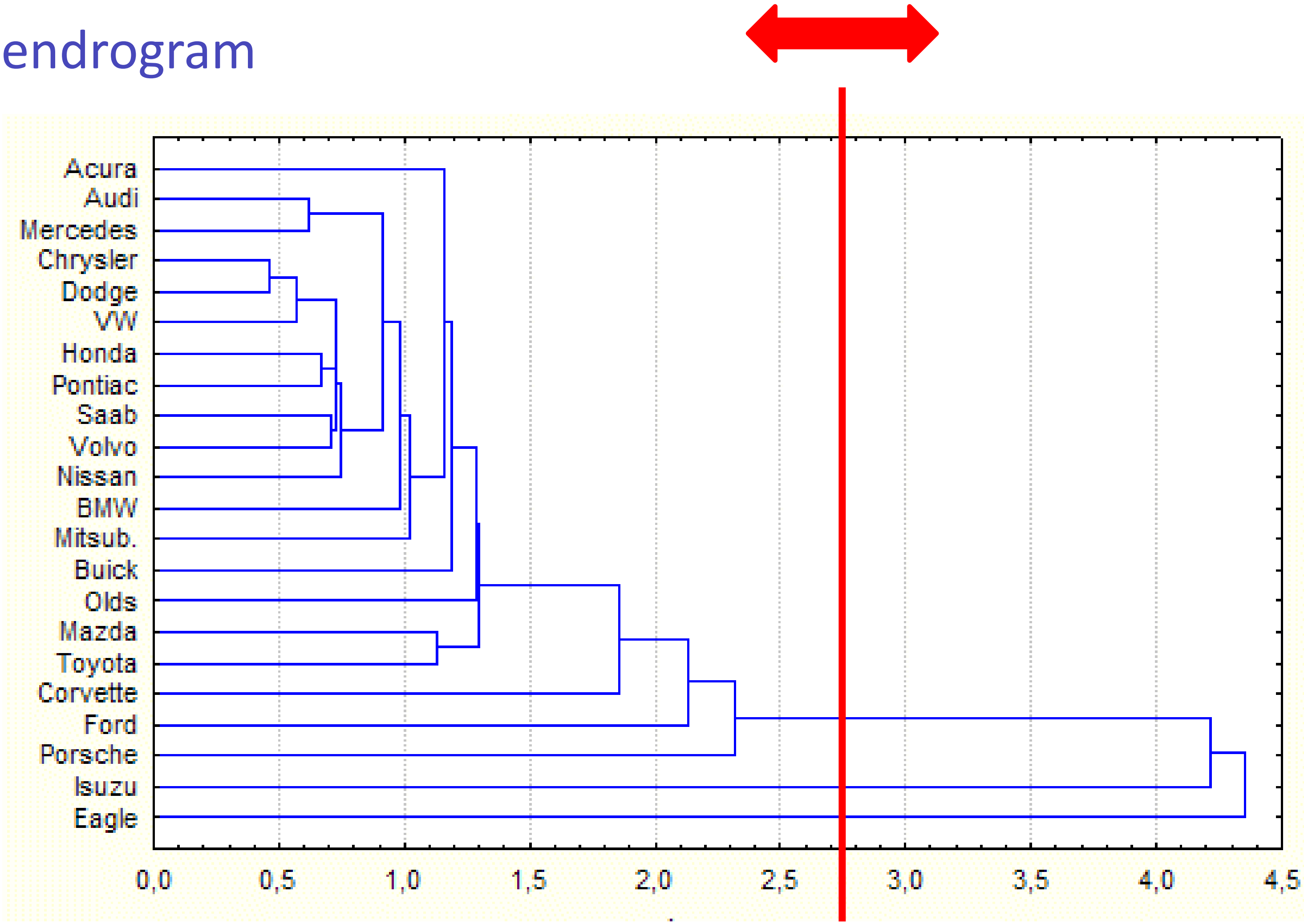


Hierarchical clustering



Hierarchical clustering

- Dendrogram



Contain the data really the clusters?

- Let’s calculate the CPCC (Cophenetic Correlation Coeffitient)
- The CPCC is a normalized covariance of distances in the original space and in the dendrogram
- If the value CPPC is less than about 0.8, all the instances belong to a single large cluster
- Generally, the higher the cophenetic coefficient of correlation, the lower is the loss of information occurring in the process of merging of objects into clusters

Hierarchical clustering

- Pseudo-code of hierarchical clustering algorithm
  - c is a required number of clusters

```
begin initialize c, c'←n, Di←{xi} i=1,...,n
   do c'←c'+1
         Calculate a matric of distances
         Find nearest clusters Di and Dj
         Merge clusters Di a Dj
   until c=c'
   return c clusters
end
```

- The procedure ends when it reached the desired number of clusters
  - when c=1, we get the dendogram
- complexity
  - $O(cn^2d)$ and typically n>>c

K-means

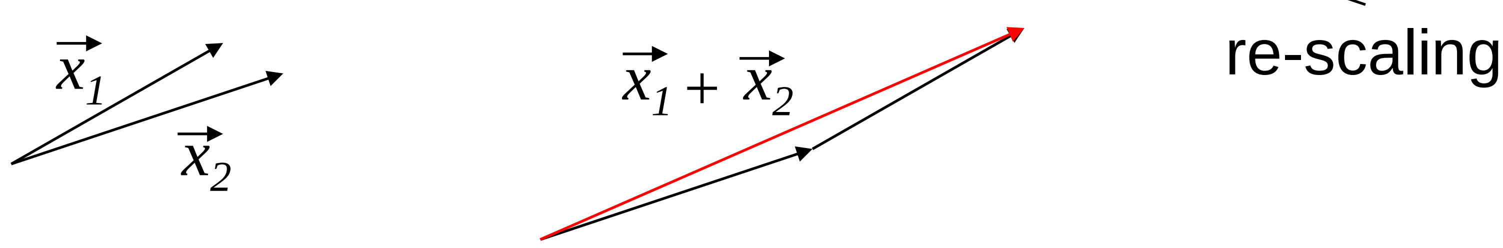
- How to avoid the calculation of all mutual distances?
- Let’s calculate distances from the representatives of clusters.
- The number of representatives is significantly smaller than the number of instances.
- Disadvantage: We have to determine the number of representatives (K) in advance.

K-means

- Representatives - here they are called centroids
- The center c of a cluster is calculated:

$$\vec{\mu}(c) = \frac{1}{|c|} \sum_{\vec{x} \in c} \vec{x}$$

  - What does this mean? How vector are summed?

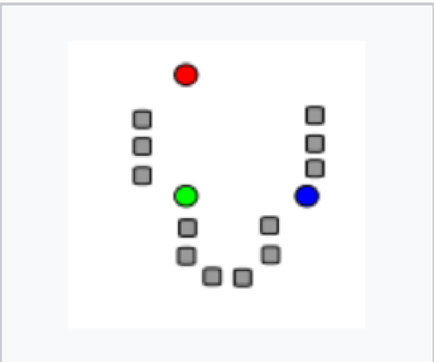


- Suppose that we know the number of clusters (centroids) and we are just looking for their position.

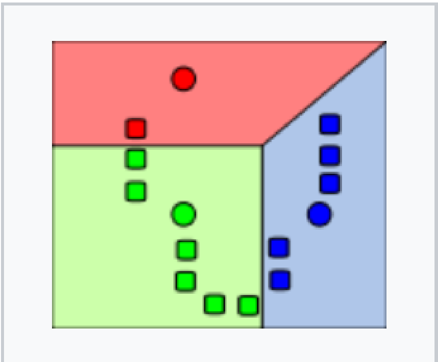
How K-means works?

- Randomly initialize k centroids. Repeat until the algorithm converges:
  - The phase of assignment of vectors: assign each vector x to cluster X_i, for which the distance from x to $\vec{\mu}_i$ (centroid X_i) is minimal
  - The phase of moving the centroids: correct positions of the centroids according to current vectors in the clusters
    - $\vec{\mu}_i(Xi) = \frac{1}{|X_i|} \sum_{\vec{x}_j \in Xi} \vec{x}_j$

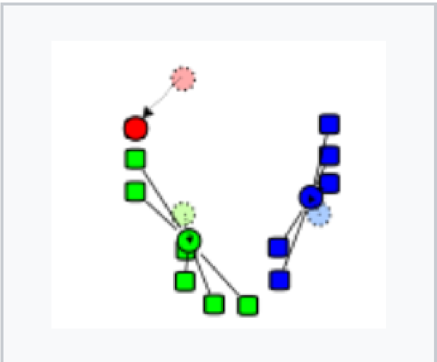
How K-means works?



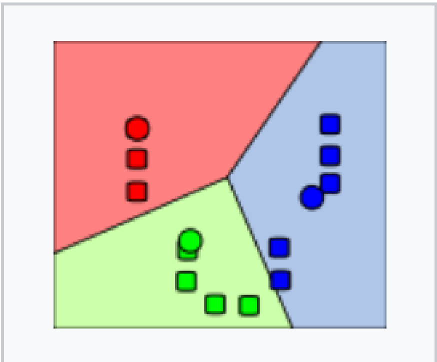
1. *k* initial "means" (in this case *k*=3) are randomly generated within the data domain (shown in color).



2. *k* clusters are created by associating every observation with the nearest mean. The partitions here represent the Voronoi diagram generated by the means.

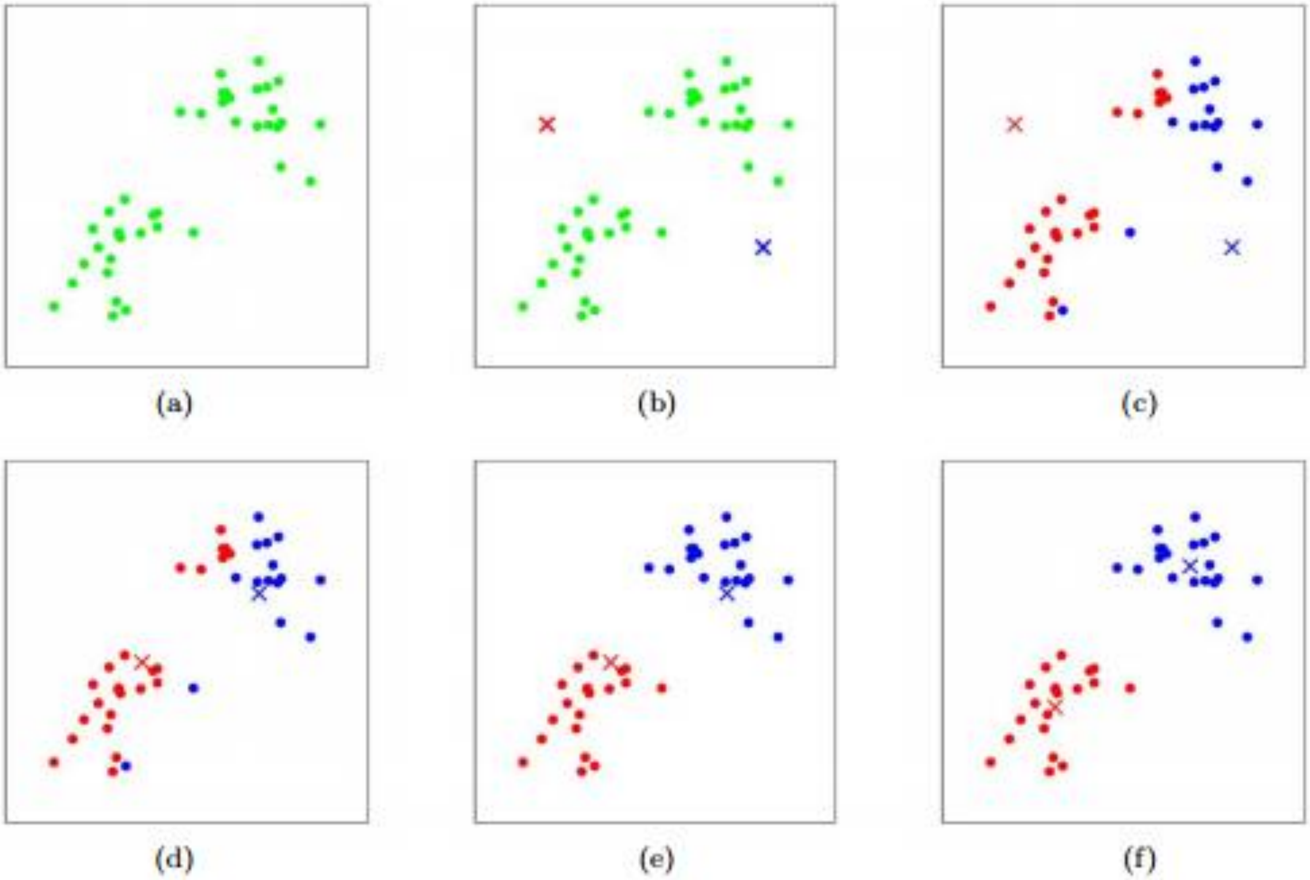


3. The centroid of each of the *k* clusters becomes the new mean.



4. Steps 2 and 3 are repeated until convergence has been reached.

How K-means works?



K-means distances

- We locally minimize energy
  - $\sum_{l=1}^{K} \sum_{xi \in Xl} ||xi - \mu l||^2$

- What does it mean?
  - For K clusters sum distances of all vectors of a given cluster from its centroid
- Does it always converge to the global minimum of the energy?
- No, it converges often to local minima. It depend on initialization of the centroids.

The K-means algorithm

- Input:
  - n patterns and a number of resulting centers c
- Output:
  - Resulting centers $\mu_1, \ldots, \mu_c$
- Algorithm:

```
1. begin initialize n, c, µ1,..., µc
2.       do classify n patterns to their nearest µi
3.          recalculate µi
4.   until no µi has changed
6.   return µ1,..., µc
7. end
```

- Complexity:
  - $O(ndcT)$
  - d is the dimension of patterns and T is a number of iterations

K-means algorithm for kids ☺

- Once there was a land with N houses...
- One day K kings arrived to this land.
- Each house was taken by the nearest king.
- But the community wanted their king to be at the center of the village, so the throne was moved there.
- Then the kings realized that some houses were closer to them now, so they took those houses, but they lost some. This went on and on… (2-3-4)
- Until one day they couldn't move anymore, so they settled down and lived happily ever after in their village...

Number of centers (clusters)

- For the K-means it is necessary to determine the K in advance – it is hard when you do not know anything about the data
- Think about an algorithm that will automatically derive the number of clusters from the data.

What criterion to use for choosing K?

- Minimum of energy?
  - $W(K) = \sum_{l=1}^{K} \sum_{xi \in Xl} \|xi - \mu l\|^2$

- Inappropriate, it decreases to zero for K = number of instances.

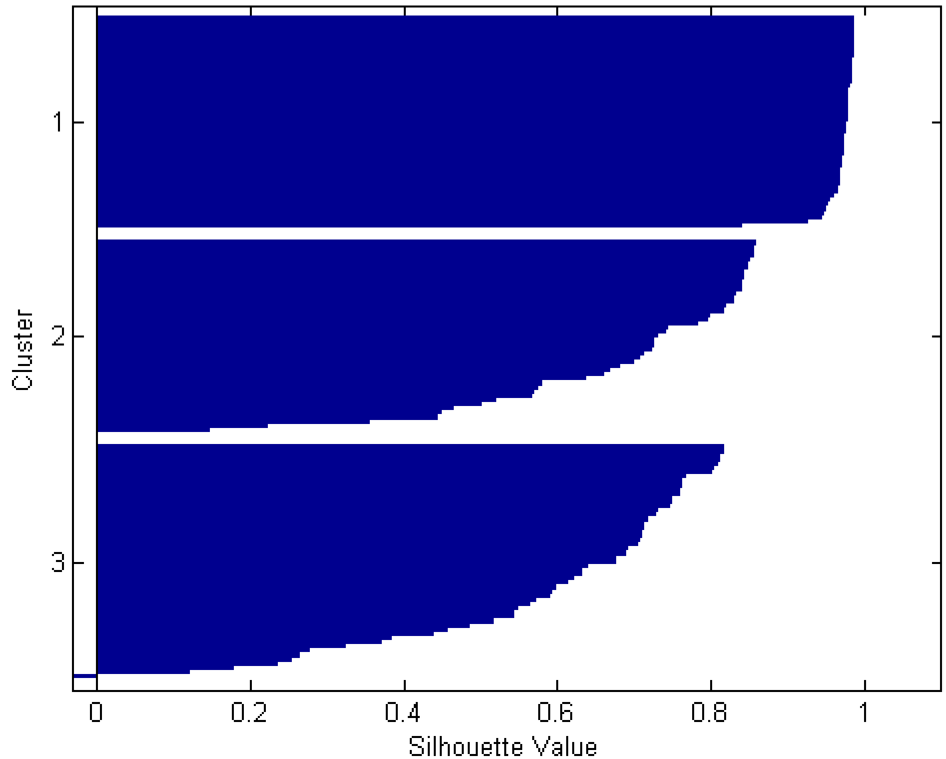
- It is better to find the maximum of function:
  - $H(K) = \frac{W(K) - W(K+1)}{W(K+1)}$

Silhouette – chart of clusters' outlines

- Iris data, for each instance calculate the certainty of its classification to cluster *s*(*i*) = <-1,1>
  - $s(i) = \frac{b(i) - a(i)}{max\{a(i), b(i)\}}$

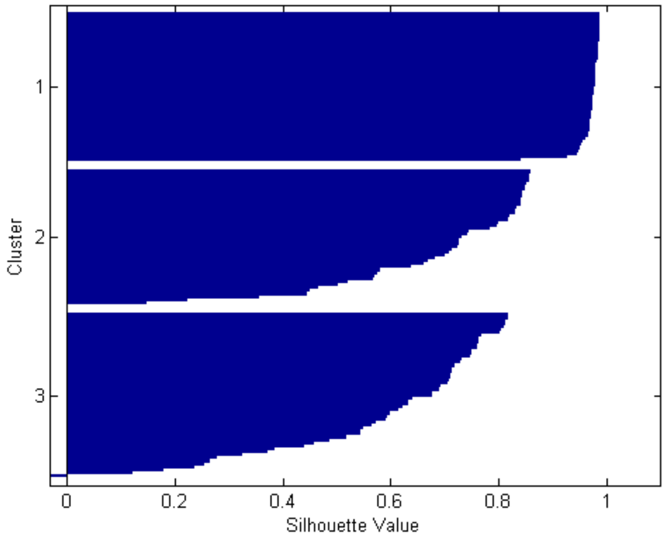
Where *a*(*i*) is an average distance of instance *i* from instances of a cluster to which it is assigned

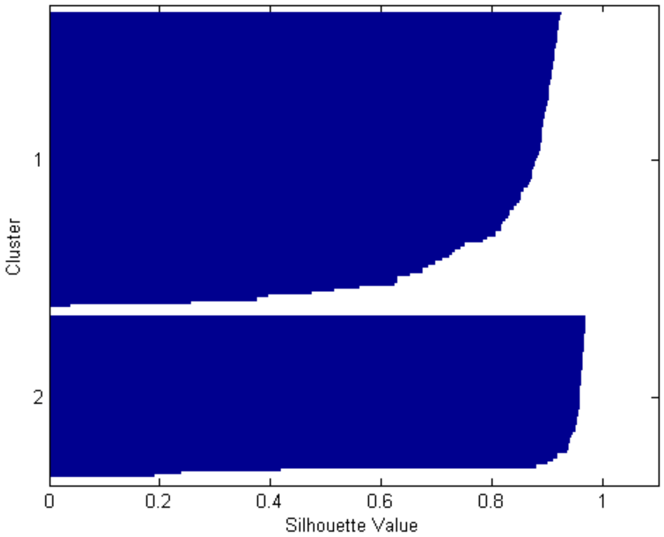
b(*i*) is an average distance of instance *i* from instances of the nearest cluster



Rating the clustering by the Silhouette chart

- Which output of the K-means is better?





- The one that has a better average value of $s(i)$ for all instances.
- Ideally on testing data.

Davies–Bouldin index

- Davies–Bouldin index (DBI) is a metric for evaluating clustering algorithms.

$$DB \equiv \frac{1}{N}\sum_{i=1}^{N} D_i \qquad D_i \equiv \max_{j \neq i} R_{i,j} \qquad R_{i,j} = \frac{S_i + S_j}{M_{i,j}}$$

1. $R_{i,j} \geqslant 0$.
2. $R_{i,j} = R_{j,i}$.
3. When $S_j \geqslant S_k$ and $M_{i,j} = M_{i,k}$ then $R_{i,j} > R_{i,k}$.
4. When $S_j = S_k$ and $M_{i,j} \leqslant M_{i,k}$ then $R_{i,j} > R_{i,k}$.

- S_i - is a measure of scatter within the cluster
- M_{ij} - is a measure of separation between cluster C_i and cluster C_j

Davies–Bouldin index

- A_i is the centroid of C_i and T_i is the size of the cluster I

$$S_i = \left(\frac{1}{T_i} \sum_{j=1}^{T_i} |X_j - A_i|^p \right)^{1/p}$$

$$M_{i,j} = ||A_i - A_j||_p = \left(\sum_{k=1}^{n} |a_{k,i} - a_{k,j}|^p \right)^{\frac{1}{p}}$$

- $a_{k,i}$ is the *k*th element of A_i, and there are n such elements in A for it is an n dimensional centroid.