Data Mining

Descriptive Modelling

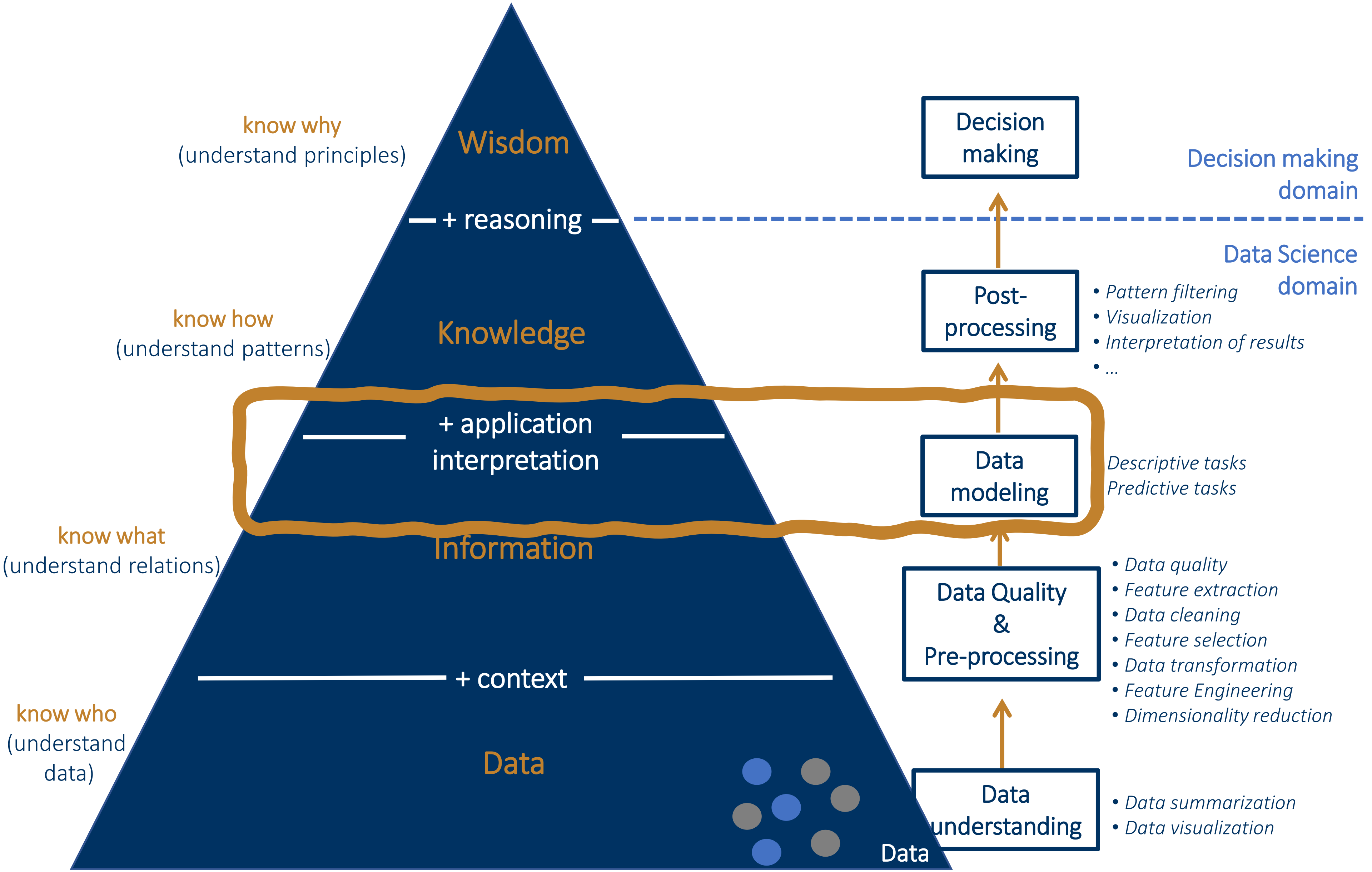
Raquel Sebastião

Departamento de Eletrónica, Telecomunicações e Informática

Universidade de Aveiro

raquel.sebastiao@ua.pt

2022/2023


know why
(understand principles)
Wisdom
+ reasoning
know how
(understand patterns)
Knowledge
+ application
interpretation
know what
(understand relations)
Information
+ context
know who
(understand
data)
Data
Data
Decision making
Decision making domain
Data Science domain
Post-processing
• Pattern filtering
• Visualization
• Interpretation of results
• ...
Data modeling
Descriptive tasks
Predictive tasks
Data Quality & Pre-processing
• Data quality
• Feature extraction
• Data cleaning
• Feature selection
• Data transformation
• Feature Engineering
• Dimensionality reduction
Data understanding
• Data summarization
• Data visualization

Contents

Descriptive Analytics

Goals:

- Describe/summarize or discover structure in collections of data
  - Data summarization and visualization are simple forms of descriptive analytics
  - Cluster analysis is frequently used for discovering **structure/groups** in data
  - Clustering the data into similar groups helps greatly in summarizing the data and understanding it

Contents

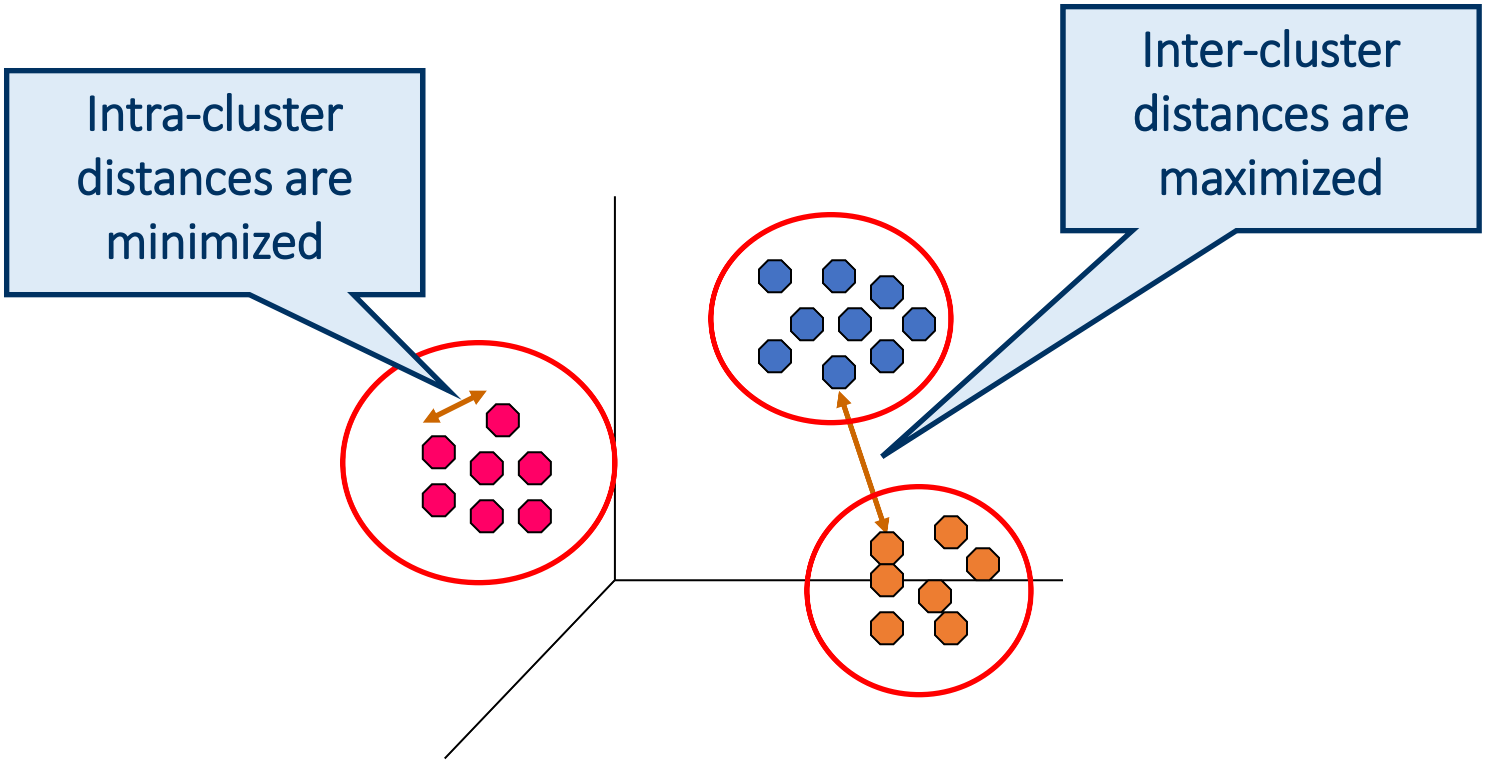
- Descriptive analytics
- Cluster analysis
- Main categories of clustering methods
- Clustering validation
- Summary

Cluster analysis (clustering)

- Process of grouping data objects (or observations) into subsets
  - exploitation of similarities (or differences) between objects(or observations, data points)
  - Objects within a group are similar (or related) to one another and different from (or unrelated to) the objects in other groups
- Unsupervised technique - no labeled data
  - finds groups based only on information in the data that describes the objects and their relationships

Cluster analysis (clustering)

- Process of grouping data objects (or observations) into subsets
  - Objects within a group are similar (or related) to one another and different from (or unrelated to) the objects in other groups



Cluster analysis (clustering): goals

- Find some structure on the data set (obtaining “natural” groups)
- Provide some abstraction of the found groups
  - representation of their main features
  - a prototype for each group
- Gain novel insights of data

Cluster analysis (clustering): motivation

- Data reduction
  - All objects within a cluster/group are substituted (represented) by the corresponding cluster representative
- Hypothesis generation
- Hypothesis testing
- Prediction based on groups
  - a cluster/group of data objects can be treated as an implicit class
  - clustering is a form of **learning by observation**, rather than *learning by examples*

Cluster analysis (clustering): applications

- Business and Marketing
  - group clients with similar buying behavior
  - describe different market segments from a set of potential clients
  - group stocks with similar price fluctuations
- Medical
  - find patients with similar symptoms
  - provide treatment recommendations based on groups of similiar patients
  - identify groups of diseases
  - Group diagnostic imaging techniques with similar characteristics

Cluster Analysis (clustering): applications

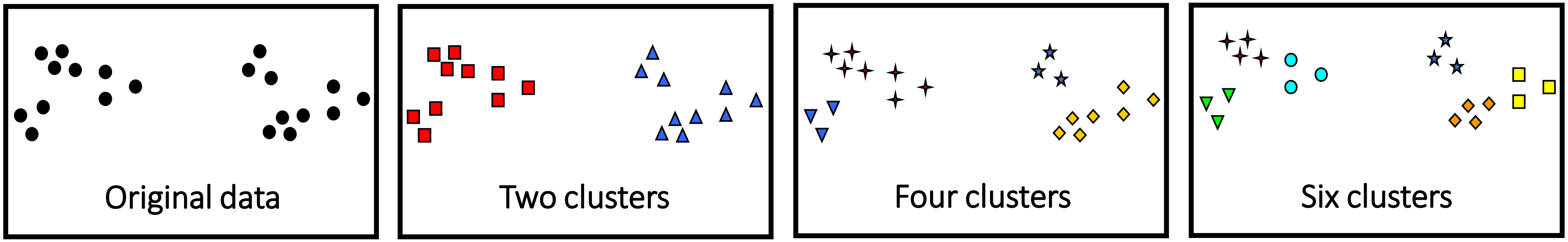
- Document retrieval
  - find documents with similar contents (travels, economy, ...).
- Image retrieval
  - find images with similar contents (sport, landscapes, ...)
- Biology
  - describe spatial and temporal communities of organisms
  - group genes or proteins that have similar functionality
- Web Mining
  - find communities in social networks
  - build recommendation systems

Cluster analysis (clustering): subjectivity

Different clusters may result, depending on

- the proximity measure
- the clustering criterion
- the clustering algorithm

How many clusters?



Contents

- Descriptive analytics
- Cluster analysis
- Main categories of clustering methods
- Clustering validation
- Summary

Clustering: main categories

Partitional: divide the observations in *k* partitions according to some criterion

- Representative based: identify cluster representatives (centroids)
- Density based[1]: locate regions of high density in the feature space
- Model based: assume a probability model for the data
- Grid based: discretize the data into *p* intervals (typically equal-width)
- Neural-Network Based: Self Organizing Maps (SOM) - consider an underlying "topology" that relates cluster centroids to one another

Hierarchical: successive development of clusters by generating a hierarchy of groups

- Agglomerative: generate a hierarchy from bottom-up (from n to 1 group)
- Divisive: create a hierarchy in a top-down way (from 1 to n groups)

[1] can be considered two-level hierarchical agglomerative

Clustering: task stages

- Proximity measure
  - quantifies the term similar or dissimilar
- Clustering criterion
  - cost function or some type of rules
- Clustering algorithm
  - steps followed to reveal the structure, based on the proximity measure and the adopted criterion
- Validation of the results
- Interpretation of the results

Clustering: partitional methods

Representative based

Discovering the groupings in the data by optimizing a specific objective function and iteratively improving the quality of partitions

- Cluster compactness
  - how similar are objects within the same cluster
- Cluster separation
  - how far is the cluster from the other clusters

- A clustering solution assigns all the objects to a cluster
  - hard clustering: an object belongs to a single cluster
  - fuzzy clustering: each object has a probability of belonging to each cluster

K-means clustering

- Partitional clustering method
- Number of clusters, **K**, must be specified
  - Methods to determine the "best" **K**
- Each cluster is represented by the centroid/representative (center point)
- Each object is assigned to the cluster with the closest centroid
- Different kind of proximity measures can be used
  - Manhattan distance (L_1 norm), Euclidean distance (L_2 norm), Cosine similarity, Correlation
- Simple iterative algorithm

K-means clustering

Consider the cluster $C_k = \{x_1, x_2, \ldots, x_{n_k}\}$, the centroid of C_k is given by

$$c_k = \frac{1}{n_k} \sum_{x_i \in C_k} x_i$$

<u>Goal</u>: obtain a set of clusters C that minimize the criterion

$$h(C) = \sum_{j=1}^{K} \sum_{x_i \in C_j} d(x_i, c_j)$$

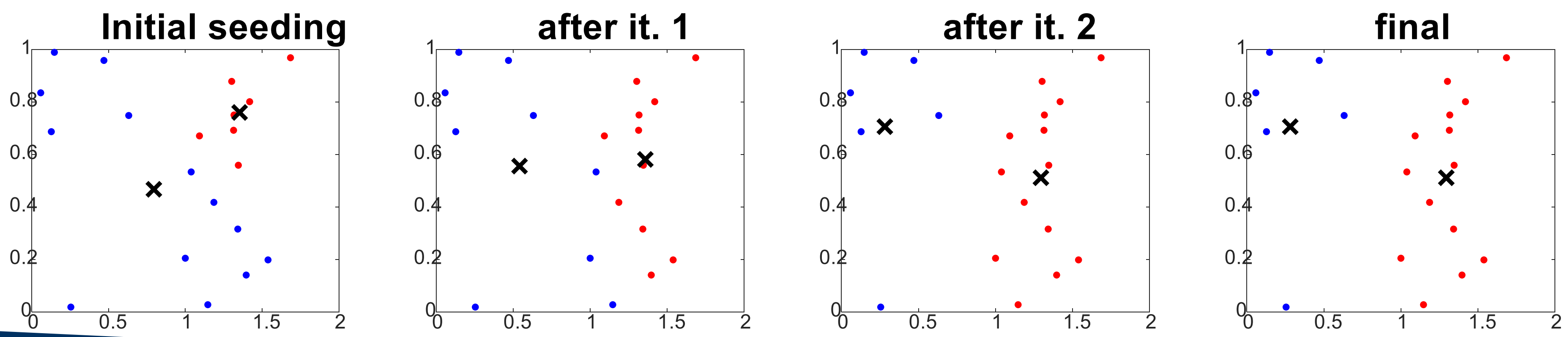
Usual criteria for numerical data

- Sum of square errors (SSE): $d(x_i, c_j) = (x_i - c_j)^2$
- L1 measure: $d(x_i, c_j) = |x_i - c_j|$
- Cosine: $d(x_i, c_j) = 1 - \frac{x_i \bullet c_j}{||x_i|| \times ||c_j||}$

K-means clustering

Execution of the K-means clustering algorithm:

- Select **K** points as the initial centroids/representatives (often randomly chosen)
- Repeat
  - assign each object/observation to the group with the nearest centroid
  - re-compute cluster centroids (i.e., **mean** point) of each cluster
- Until convergence criterion is satisfied (i.e., the centroids stop changing)



K-means clustering: details

- **Initial centroids** are often chosen randomly
- minimize intra-cluster distance and maximize inter-cluster distances

Advantages:

- Stochastic approach that frequently works well. It tends to identify local minima
- Most of the convergence happens in the first few iterations
  - Often the stopping condition is changed to 'Until relatively few points change clusters'
- Complexity: $O(n*K*I*d)$ where n: # of objects, K: # of clusters, I: # of iterations, and d: # attributes
  - Normally, K, $I << n$; thus, an efficient method

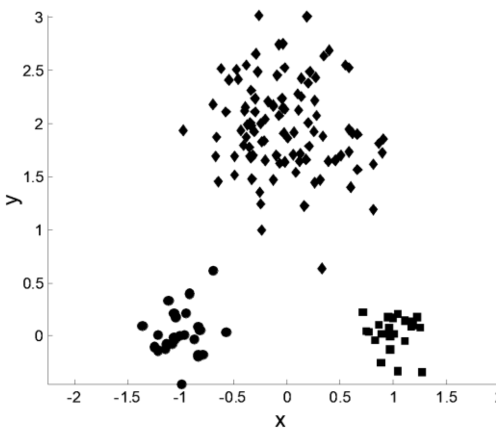
K-means clustering: details

<u>Disadvantages:</u>

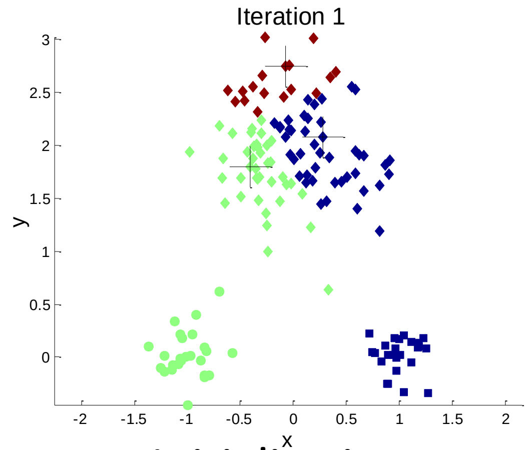
- Different initializations may generate rather different final clusters
- Sensitive to **noisy** data and **outliers**
  - Possible solution: remove outliers prior to clustering
  - Alternatives: **K-medians**, **K-medoids**, …
- K-means is applicable only to numerical data
  - Alternatives: **K-modes** (*categorical data*)
- Not suitable to discover clusters:
  - with different sizes
  - with different densities
  - with non-convex/non-globular shapes
  - Alternatives: **kernel *K*-means**, **density-based** clustering, …

K-means limitations: poor initialization

- Different initializations may generate rather different final clusters

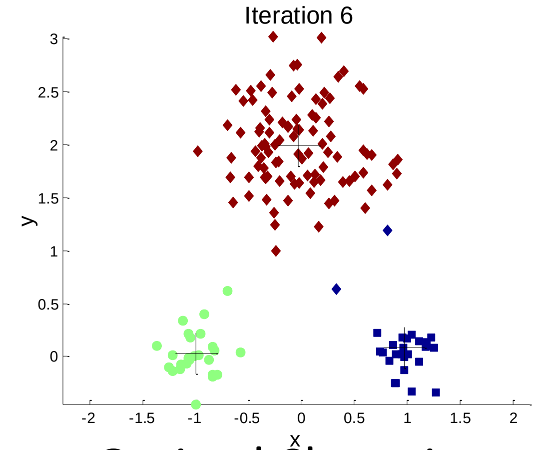


Original Data

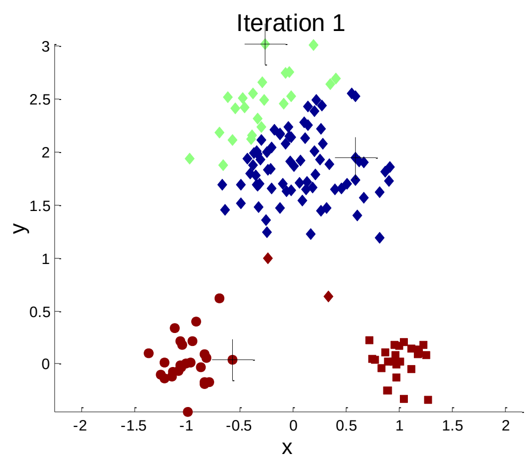


Initialization

(...)

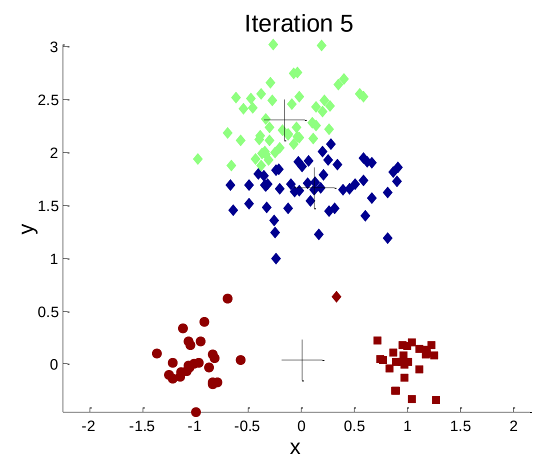


Optimal Clustering



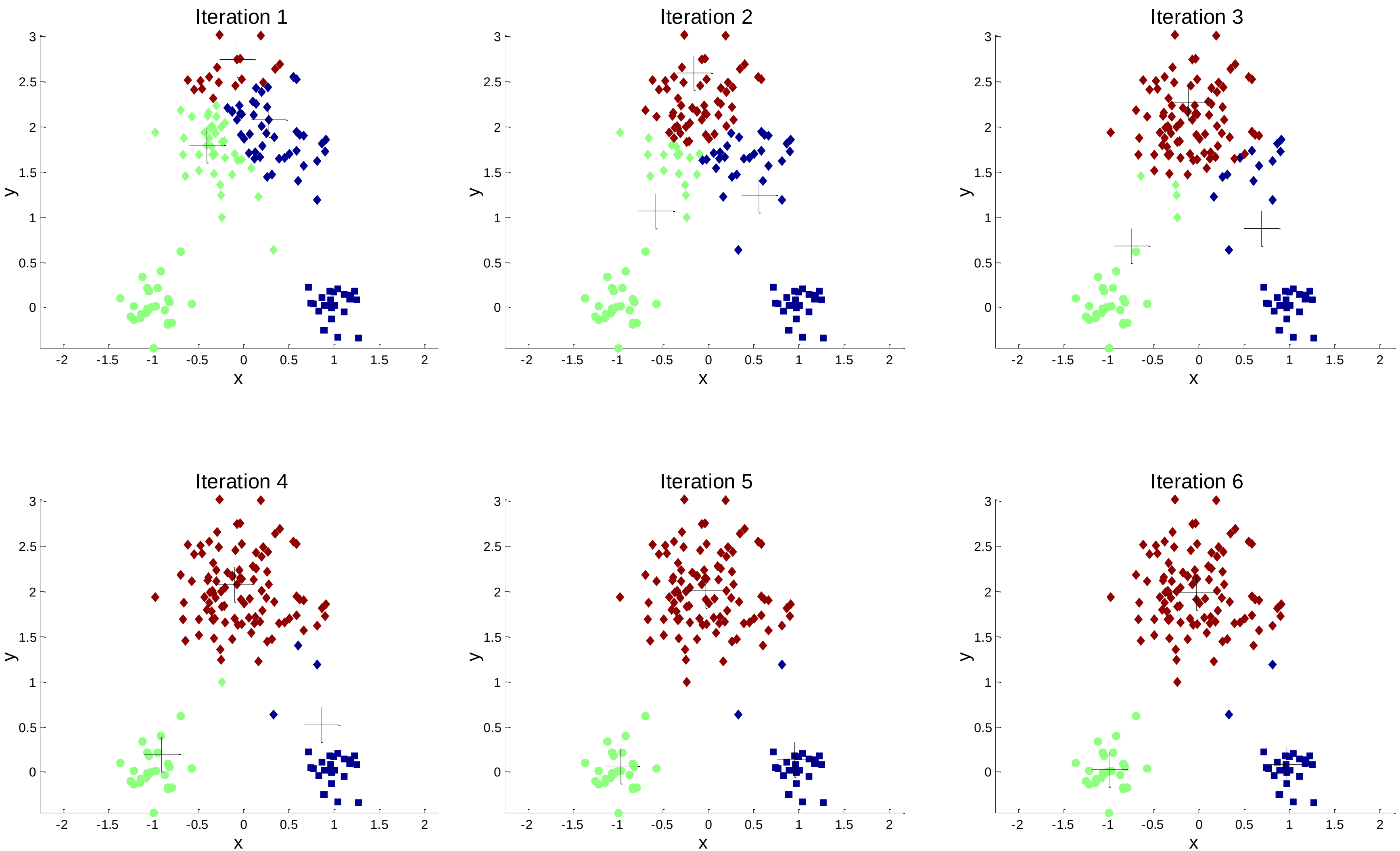
Initialization

(...)

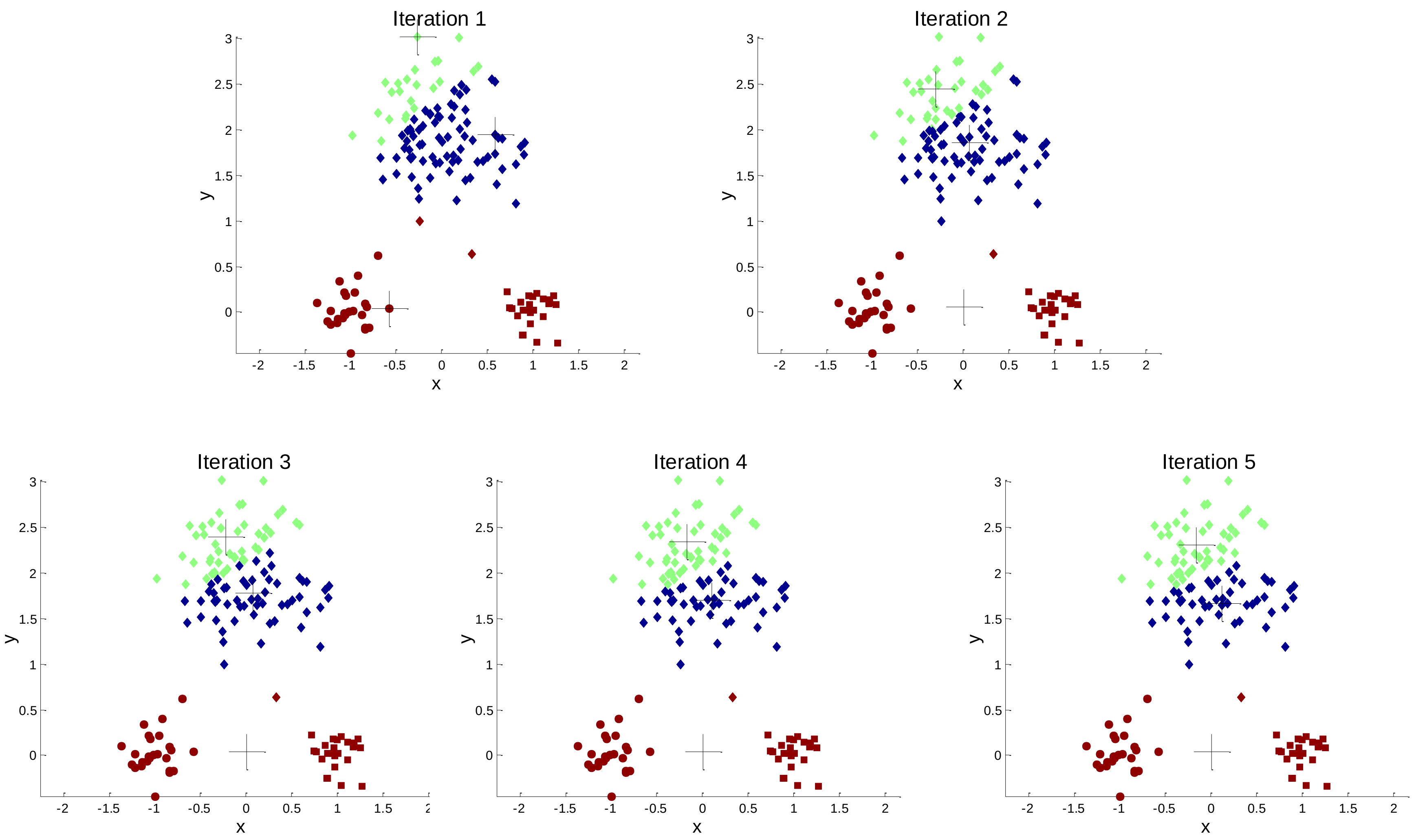


Sub-optimal Clustering

K-means limitations: poor initialization



K-means limitations: poor initialization



K-means limitations: poor initialization

Solutions

- Multiple runs with K seeds randomly selected
  - Helps, but probability is not on your side
- K-means++ : select the most widely separated centroids
  - The first centroid is selected at random
  - The next centroid selected is the one that is farthest from the currently selected (selection is based on a weighted probability score)
  - The selection continues until K centroids are obtained
- Use hierarchical clustering to determine initial centroids
- Bisecting K-means
  - Not as susceptible to initialization issues

K-medians clustering: handling outliers

Medians are **less sensitive to outliers** than means

K-medians: Instead of taking the **mean** value of the object in a cluster as a reference point, **medians** are used

Execution of the K-medians clustering algorithm:

- Select **K** points as the initial centroids/representatives (i.e., as initial **K** medians)
- Repeat
  - assign each object/observation to the group with the nearest centroid
  - re-compute cluster centroids (i.e., **median** point) of each cluster
- Until convergence criterion is satisfied (i.e., the centroids stop changing)

K-medoids clustering: handling outliers

K-medoids: Instead of taking the **mean** value of the object in a cluster as a representative/centroid, **medoids** are used, which is the **most centrally** located object in a cluster

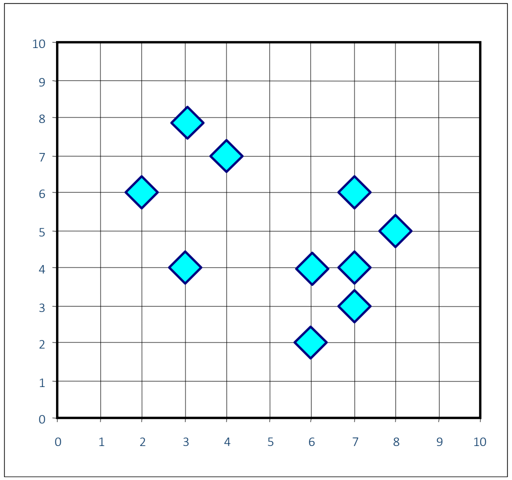
Is more robust to the presence of outliers because it uses original objects as centroids instead of averages that may be subject to the effects of outliers

K-medoids clustering: handling outliers

Execution of the K-medoids clustering algorithm:

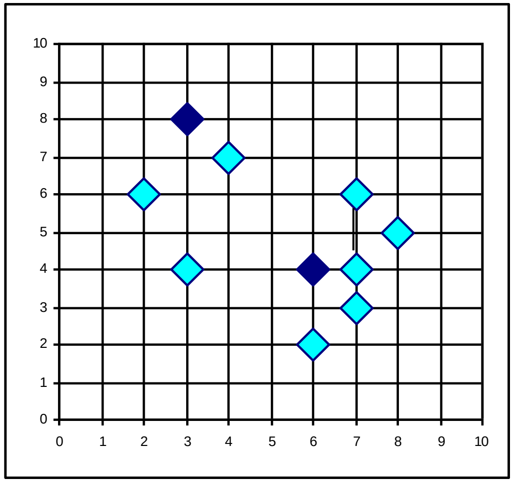
- Select **K** points as the initial centroids/representatives (i.e., as initial **K** medoids)
- Repeat
  - assign each object/observation to the group with the nearest medoid
  - Randomly select a non-representative object o_i
  - Compute the total cost S of swapping the medoid m with o_i
  - If $S < 0$, then swap m with o_i to form the new set of medoids
- Until convergence criterion is satisfied (i.e., the centroids stop changing)

K-medoids clustering: PAM (Partitioning Around Medoids)

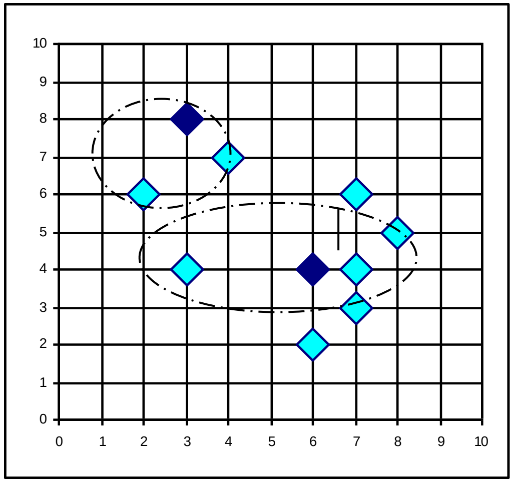


$K = 2$

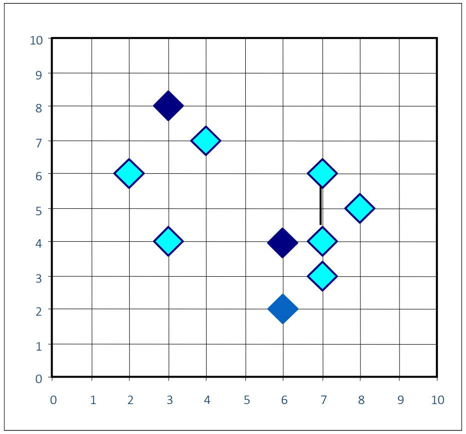
Arbitrary choose K object as initial medoids



Assign each remaining object to nearest medoid (m)



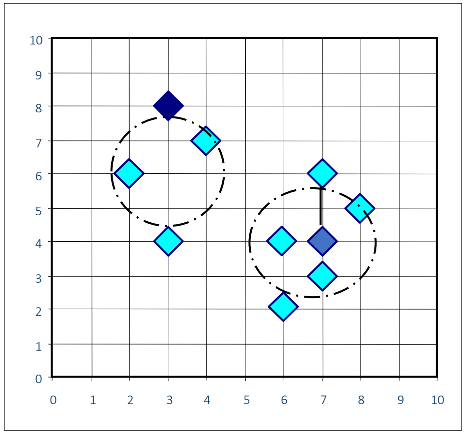
Randomly select a non-medoid object, O_{ramdom}



Compute total cost of swapping

If quality is improved:

swap m and O_{ramdom}



Select initial K *medoids* randomly

Repeat

Object re-assignment

If quality is improved

Swap medoid m with o_i

Until convergence criterion is satisfied

K-medoids clustering: discussion

PAM (Partitioning Around Medoids)

- Computational complexity: $O(K(n-K)^2)$ (quite expensive!)
- PAM works effectively for small data sets but does not scale well for large data sets (due to the computational complexity)

Efficient improvements on PAM

- CLARA (Clustering Large Applications)
  - PAM on samples; $O(Ks^2 + K(n-K))$, s is the sample size
- CLARANS (Clustering Large Applications based on RANdomized Search)
  - randomized re-sampling
  - ensure efficiency & quality

K-modes clustering: categorical data

- K-Means **cannot** handle non-numerical (categorical) data
  - Mapping categorical value to 1/0 cannot generate quality clusters
- *K-Modes*: an extension to **K-Means** by replacing means of clusters with ***modes*** as representatives/centroids
- Dissimilarity measure for categorical data: **frequency-based**

Algorithm is still based on iterative *object cluster assignment* and *centroid update*

kernel K-means clustering

K-Means can only detect clusters that are linearly separable

Kernel K-Means can be used to detect non-convex clusters

- A region is **convex** if it contains all the line segments connecting any pair of its points. Otherwise, it is **concave**

- **Idea**: **Project data** onto the **high-dimensional kernel space**, and then perform ***K-Means*** clustering

kernel K-means clustering

- **Idea**: Project data onto the high-dimensional kernel space, and then perform ***K-Means*** clustering
  - Map data points in the input space onto a high-dimensional feature space using the kernel function
  - Perform *K-Means* on the mapped feature space
- Computational complexity is higher than K-Means
  - Need to compute and store n x n kernel matrix generated from the kernel function on the original data, where n is the number of points
- ***Spectral clustering*** can be considered as a variant of Kernel K-Means clustering

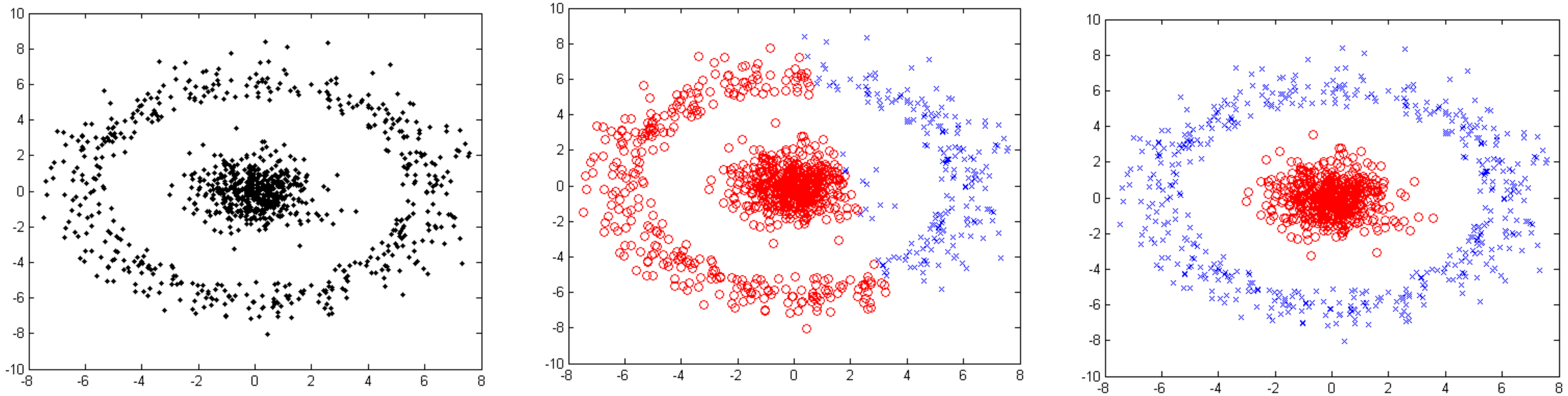
kernel K-means clustering

- **Typical Kernel functions**
  - Polynomial kernel of degree
  - Gaussian radial basis function (RBF) kernel:
  - Sigmoid kernel
- The formula for **kernel matrix K** for any two points $x_i, x_j \in C_k$:

$$K_{x_i x_j} = \phi(x_i) \bullet \phi(x_j)$$

- All steps of K-means are based on dot product, but the centroid is never explicitly computed
- Clustering can be performed without the actual individual projections $\phi(x_i)$ and $\phi(x_j)$ for the data points $x_i, x_j \in C_k$

kernel K-means clustering


The original data set — K-Means clustering — Gaussian Kernel K-Means clustering

- This data set cannot generate quality clusters by **K-Means** since it contains non-convex clusters

- Kernel transformation maps data to a kernel matrix **K** for any two points x_i, x_j :

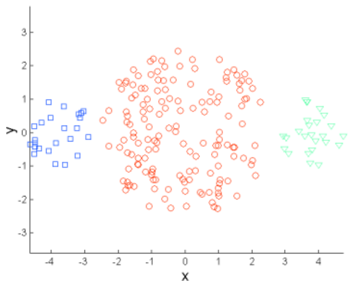
$$K_{x_i x_j} = \phi(x_i) \bullet \phi(x_j)$$

  - Gaussian kernel: $K(\boldsymbol{X}_i, \boldsymbol{X}_j) = e^{-\frac{||X_i - X_j||^2}{2\sigma^2}}$

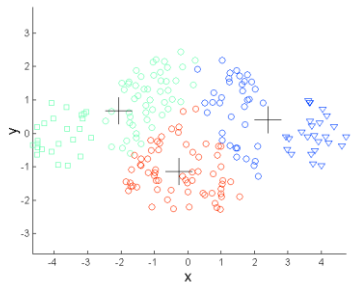
- K-Means clustering is conducted on the mapped data, generating quality clusters

K-means “variations” limitations

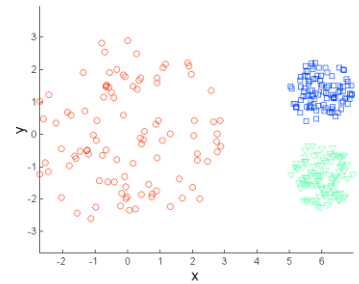
- Clusters of **diferente sizes**, **densities** and with **non-convex/non-globular shapes**



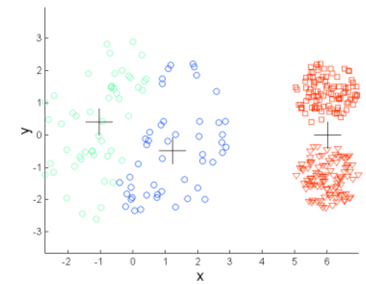
Original Data



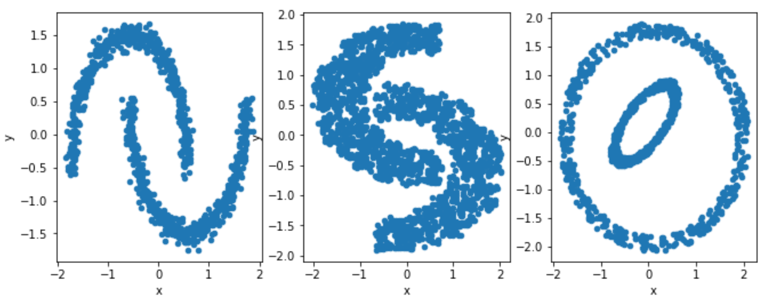
K-means (3 Clusters)



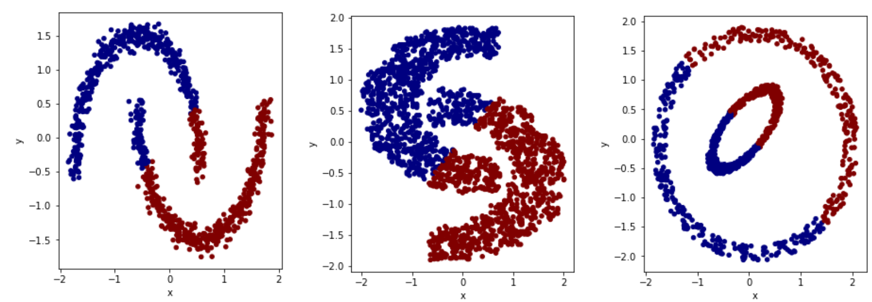
Original Data



K-means (3 Clusters)



Original Data



K-means (2 Clusters)

- Data with **outliers/noise**

Density based clustering

DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

- **Clusters** are regions of high density that are separated from one another by regions on low density
- The density of a single observation is **estimated by the number of observations that are within a specified radius** (**eps** - parameter of the method)

- It does not require the user to specify the **number of groups**
- Input the radius (**eps**) and the minimum number of points (**MinPts**)

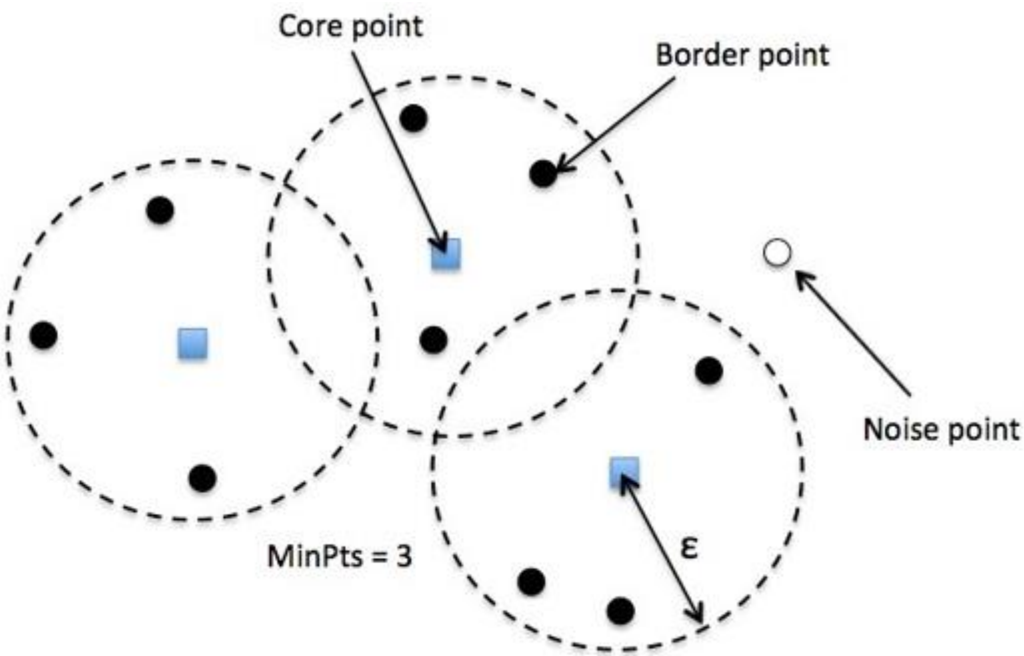
Density based clustering

Observations are identified as:

- core point: if it has at least a specified number of points (**MinPts**) within **eps** radius
  - These are points that are at the interior of a cluster
  - Counts the point itself
- border point: if the number of observations within its radius does not reach the threshold but it is within the radius of a core point
- noise point: does not have enough observations within their radius, nor is sufficiently close to any core point (any point that is not a core point or a border point)

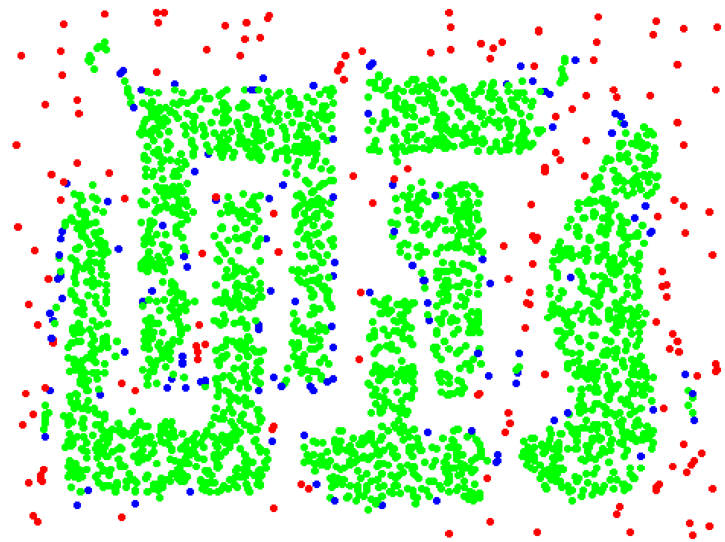
Density based clustering: DBSCAN

Core, border and noise points



Density based clustering: DBSCAN

Original Points



Point types: core, border and noise

Eps = 10, MinPts = 4

Density based clustering: DBSCAN

Form clusters using core points, and assign border points to one of its neighboring clusters

- Identify all points as core, border, or noise points
- Eliminate noise points
- Put an edge between all core points within a distance ***Eps*** of each other
- Make each group of connected core points into a separate cluster
- Assign each border point to one of the clusters of its associated core points

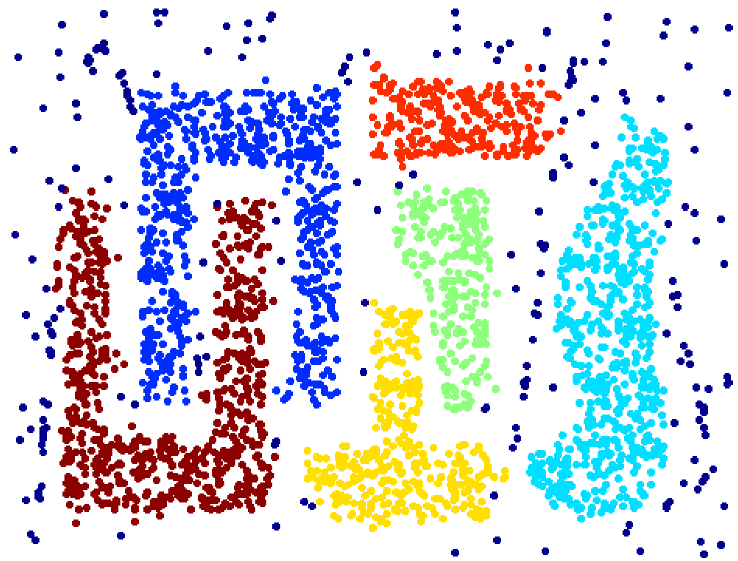
Density based clustering: DBSCAN

Advantages:

- Can handle clusters with different shapes and sizes
- Resistant to noise

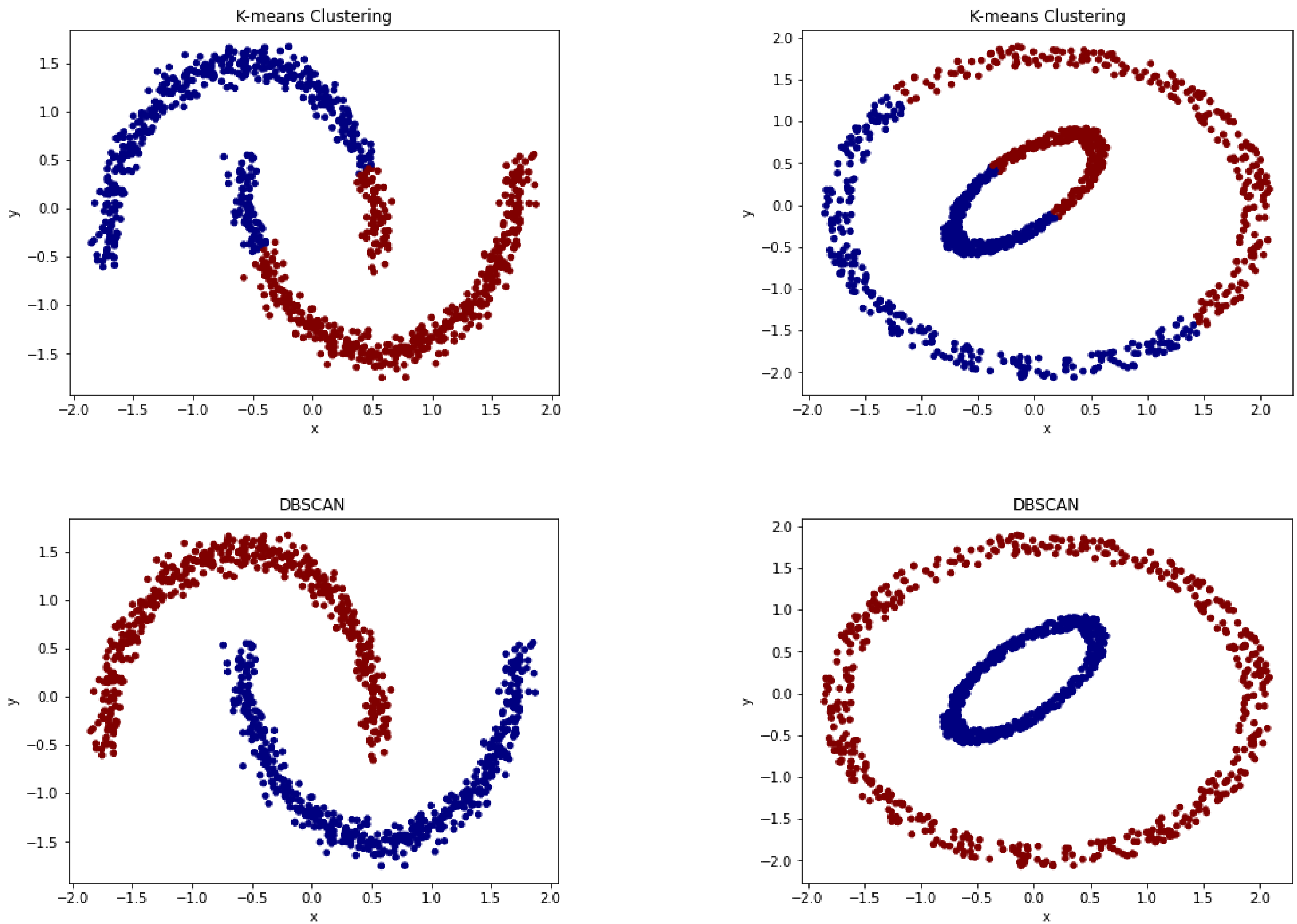
Disadvantages:

- Varying densities
- High-dimensional data



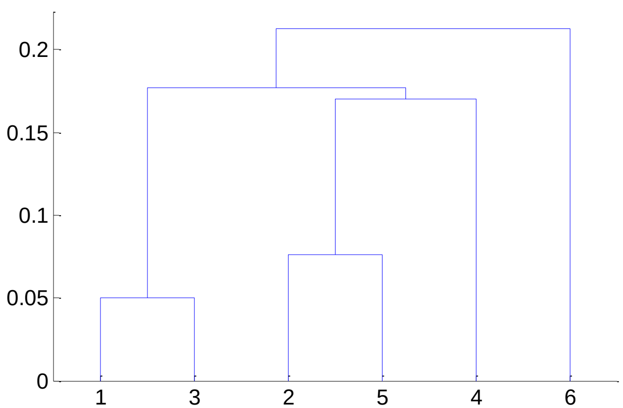
Clusters (dark blue points indicate noise)

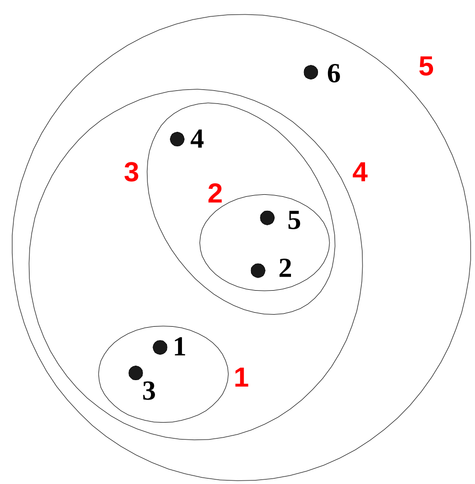
Density based clustering: DBSCAN

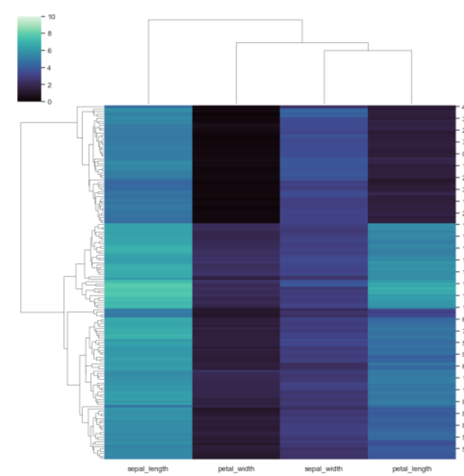


Hierarchical clustering

- Obtain a set of nested clusters organized as a hierarchical tree
  - each level represents a possible solution with *x* groups
- Can be visualized as a dendrogram
  - A tree like diagram that records the sequences of merges or splits
- Additional visualization: combine the dendogram with heat maps





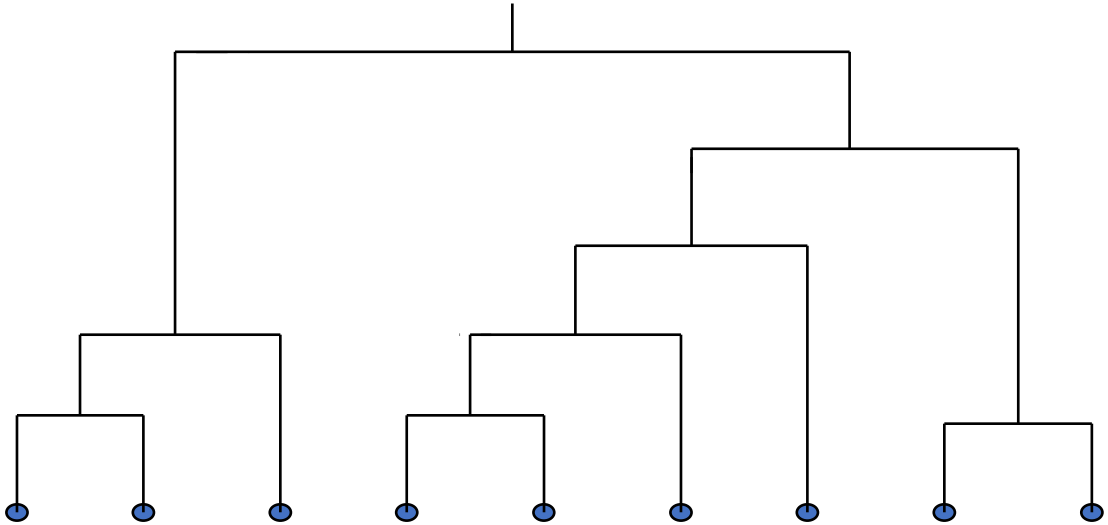


Hierarchical clustering

- Do not have to assume a pre-defined number of clusters
  - Any desired number of clusters can be obtained by ‘cutting’ the dendrogram at the proper level (the user can specify the desired number of clusters)
- More deterministic
- No iterative refinement
- Two categories of algorithms: Agglomerative and Divisive
- Key operation is the computation of the proximity of two clusters
- Different approaches to defining the distance between clusters distinguish the different algorithms

Hierarchical clustering

- Dendrogram: Decompose a set of data objects into a tree of clusters by multi-level nested partitioning
- A clustering of the data objects is obtained by cutting the dendrogram at the desired level, and each connected component forms a cluster



Hierarchical clustering generates a dendrogram (a hierarchy of clusters)

Hierarchical clustering

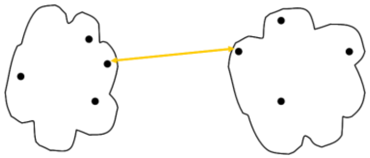
Two main types of hierarchical clustering

- Agglomerative: bottom-up, from n to 1 group
  - Start with as many clusters as there are objects
  - At each step, merge the closest pair of clusters into a single cluster
    - The chosen pair is formed by the groups that are more similar
  - Until only one cluster (or k clusters) left
- Divisive: top-down (less used), from 1 to n groups
  - Start with a single, all-inclusive, cluster
  - At each step, split a cluster into two
    - The selected cluster is the one with smallest uniformity
  - Until each cluster contains only one object (or there are k clusters)

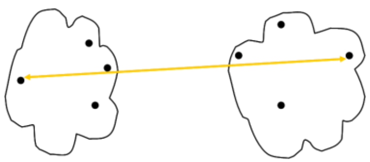
Traditional hierarchical algorithms use a proximity matrix

- Merge or split one cluster at a time

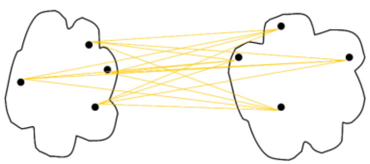
Hierarchical clustering: Inter-cluster proximity



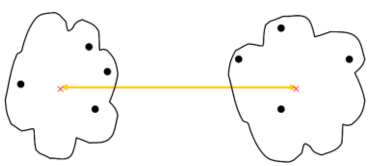
- **Single Linkage** (MIN): $d(C_1, C_2) = \min_{x_i \in C_1, x_j \in C_2} d(x_i, x_j)$
  - dissimilarity between the two most similar objects of the two clusters



- **Complete Linkage** (MAX): $d(C_1, C_2) = \max_{x_i \in C_1, x_j \in C_2} d(x_i, x_j)$
  - dissimilarity between the two most dissimilar objects of the two clusters



- **Average Linkage**: $d(C_1, C_2) = \frac{1}{n_1 n_2} \sum_{x_i \in C_1, x_j \in C_2} d(x_i, x_j)$
  - dissimilarity between all pairs of objects of the two clusters



- **Distance between centroids**

- **Ward's method**: takes into account the number of objects of the clusters

Hierarchical clustering: agglomerative methods

Algorithm:

- Compute the proximity matrix: matrix with value of proximity measure between pairs of points
- Let it each data point be a cluster
- **Repeat**
  - Merge the two closest clusters
  - Update the proximity matrix to reflect the proximity between the new cluster and original clusters
- **Until** only a single cluster remains

Hierarchical clustering: agglomerative methods

Example: consider the following distance matrix

- Use Agglomerative Hierarchical Clustering to obtain the single-link dendogram

	A	B	C	D	E	F
A	**0**					
B	4	**0**				
C	25	21	**0**			
D	24	20	1	**0**		
E	9	5	16	15	**0**	
F	7	3	18	17	2	**0**

Distance Matrix - Stage 0

	A	B	CD	E	F
A	**0**				
B	4	**0**			
CD	24	20	**0**		
E	9	5	15	**0**	
F	7	3	17	2	**0**

Distance Matrix - Stage 1

- New cluster with two "objects“
- Updating the proximity matrix
  - the distance to the remaining is the shortest distance from any member i

    Single linkage:

    $\min(d_{i,C}, d_{i,D})$

	A	B	CD	EF
A	**0**			
B	4	**0**		
CD	24	20	**0**	
EF	7	3	15	**0**

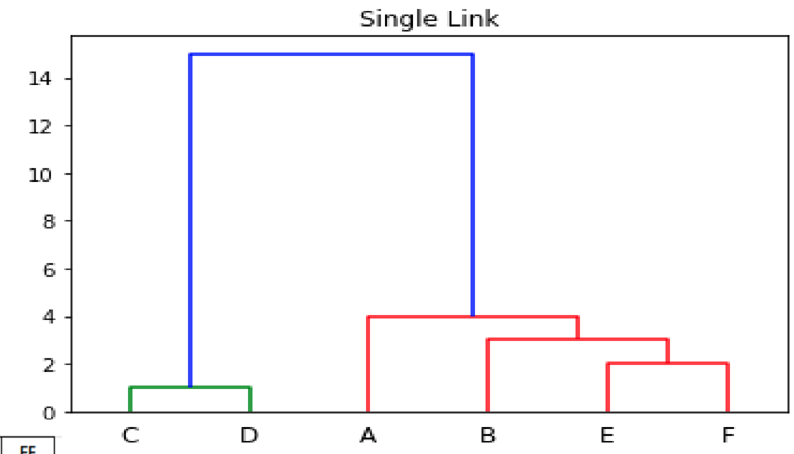
Distance Matrix - Stage 2

	A	BEF	CD
A	**0**		
BEF	4	**0**	
CD	24	15	**0**

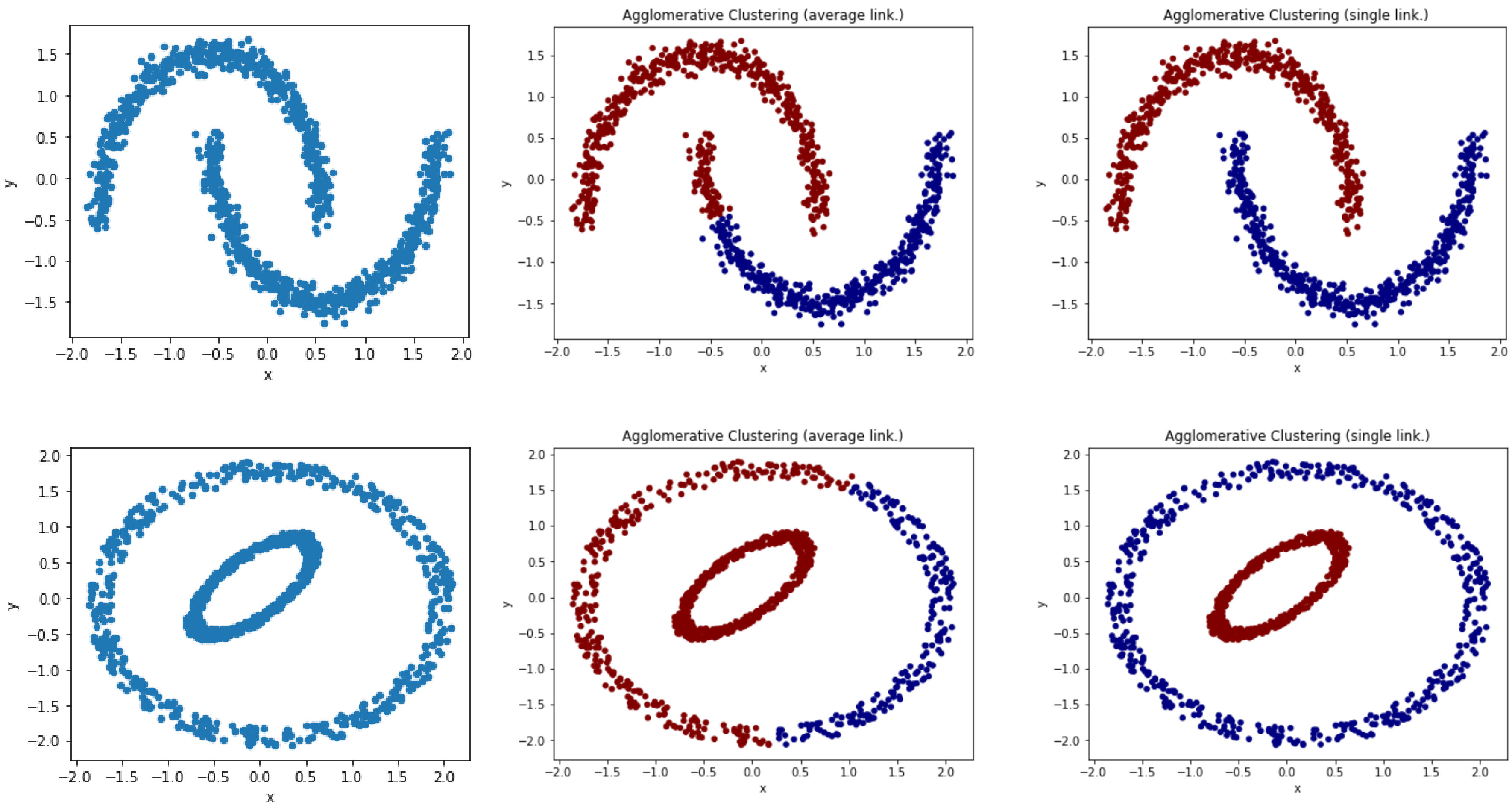
Distance Matrix - Stage 3

	ABEF	CD
ABEF	**0**	
CD	15	**0**

Distance Matrix - Stage 4

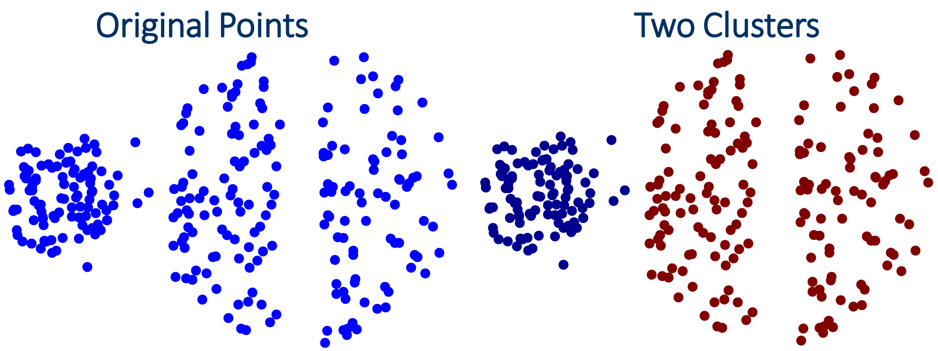


Hierarchical clustering: agglomerative methods



Hierarchical clustering

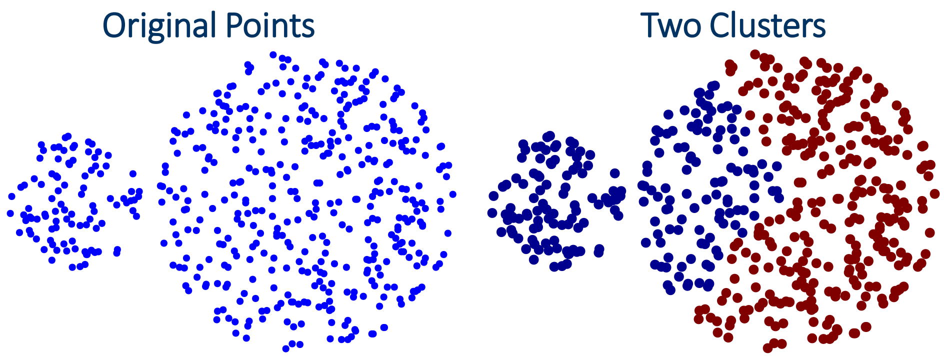
Different inter-cluster proximity schemes yield to **different types of clusters**



- Single-linkage
  - follows chains on the data
  - can handle non-elliptical shapes
  - uses a local merge criterion
  - distant parts of the cluster and the clusters' overall structure are not taken into account
  - sensitive to noise

Hierarchical clustering

Different inter-cluster proximity schemes yield to **different types of clusters**



- Complete-linkage
  - Tends to break large groups in data
  - biased towards globular clusters
  - uses a non-local merge criterion
  - chooses the pair of clusters whose merge has the smallest diameter
  - the similarity of two clusters is the similarity of their most dissimilar members
  - sensitive to outliers
  - less susceptible to noise

Hierarchical clustering

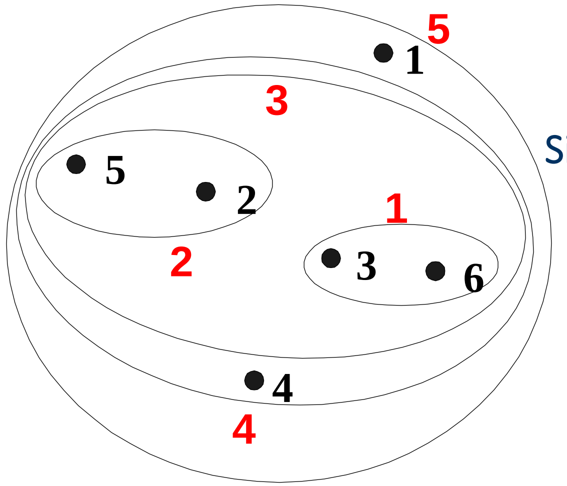
Different inter-cluster proximity schemes yield to **different types of clusters**

- Average-linkage
  - it is a compromise between **single** and **complete linkage**
  - biased towards globular clusters
  - less susceptible to noise

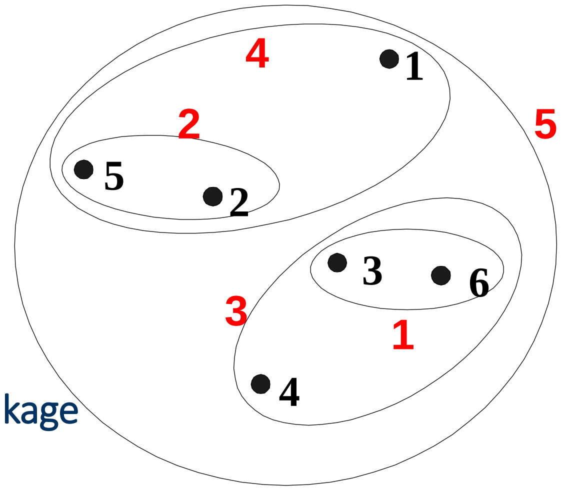
- Complete and Average linkage
  - lead to compact clusters

Hierarchical clustering: problems and limitations

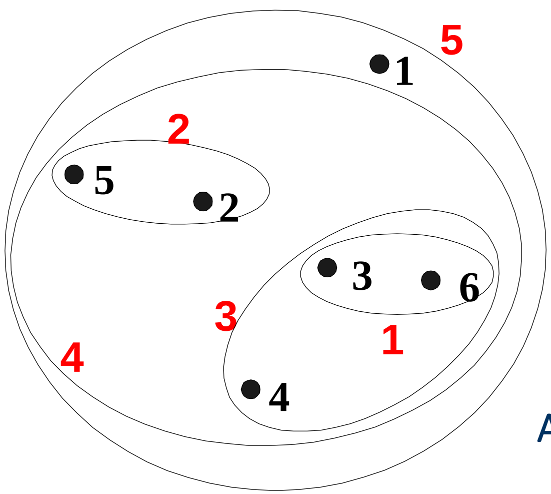
Different proximity measures yield to **different types of clusters**



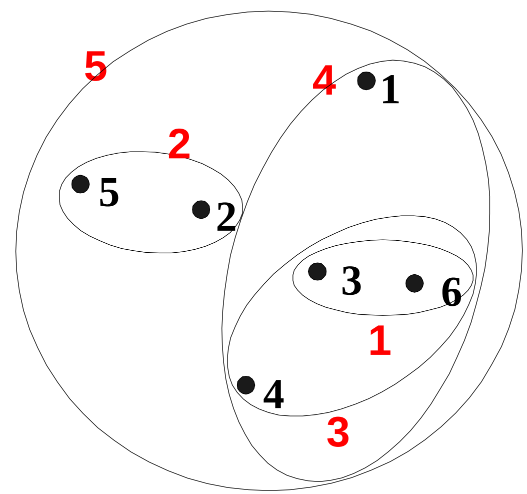
Single-linkage



Complete-linkage



Average linkage



Ward's Method

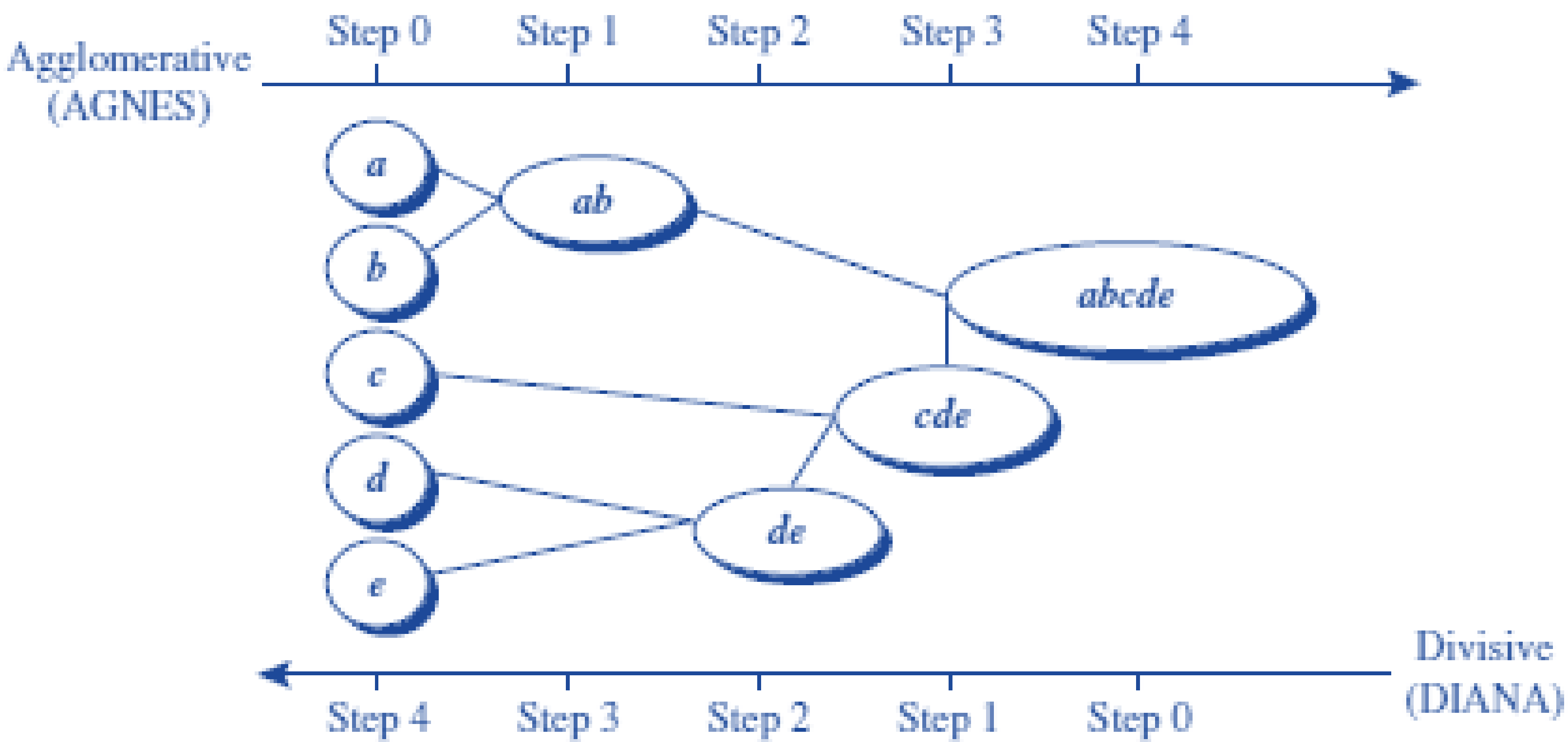
Hierarchical clustering: divisive methods

Algorithm:

- Compute the proximity matrix: matrix with value of proximity measure between pairs of points
- Start with a single cluster that contains all data points
- **Repeat**
  - choose a cluster based on a pre-defined criterion
  - use flat-algorithm ***A***[1] to split the cluster into **L** clusters
- **Until** each data point constitutes a cluster

[1] algorithm *A* can be any arbitrary clustering algorithm, and not just a distance-based one

Hierarchical clustering



Hierarchical clustering: remarks and constraints

- Time and space requirements:
  - Storage: **O(n^2)** space, **n** is the number of points
  - Time: **O(n^3)**, there are **n** steps and at each step the proximity matrix (with size **n^2**) must be updated and searched
- Once a decision is made to **merge two clusters** or **divide one cluster**, it **cannot be undone**
- No objective function is directly minimized
- **Different schemes have problems** with one or more of the following:
  - Sensitivity to noise
  - Difficulty handling clusters of different sizes and non-globular shapes
  - Breaking large clusters

Contents

- Descriptive analytics
- Cluster analysis
- Main categories of clustering methods
- **Clustering validation**
- Summary

Clustering validation

How to validate/evaluate/compare the results obtained by some clustering method?

1. **Clustering tendency**
    - Assess the suitability of clustering, i.e., whether the data has any inherent grouping structure
        - Is the found grouping structure random?
2. **Clustering stability**
    - Understand the sensitivity of the clustering result to various algorithm parameters, e.g., # of clusters
        - What is the “correct” number of clusters?
3. **Clustering evaluation**
    - Evaluating the goodness of clustering results

Clustering validation:
types of evaluation measures

3. **Clustering evaluation**: evaluating the goodness of clustering results

- Evaluating the entire clustering or just individual clusters

3.1. Unsupervised

How to evaluate the result of a clustering algorithm without external information?

3.2. Supervised

How to compare the results obtained by different methods when external information exists (such as class labels)?

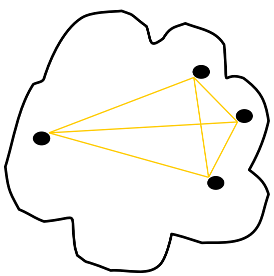
3.3 Relative (supervised or unsupervised)

How to compare clustering or clusters?

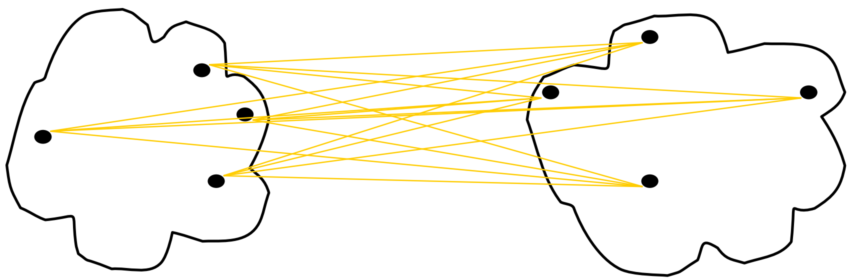
Clustering validation: unsupervised

Measure the quality of the clustering **without any external information**

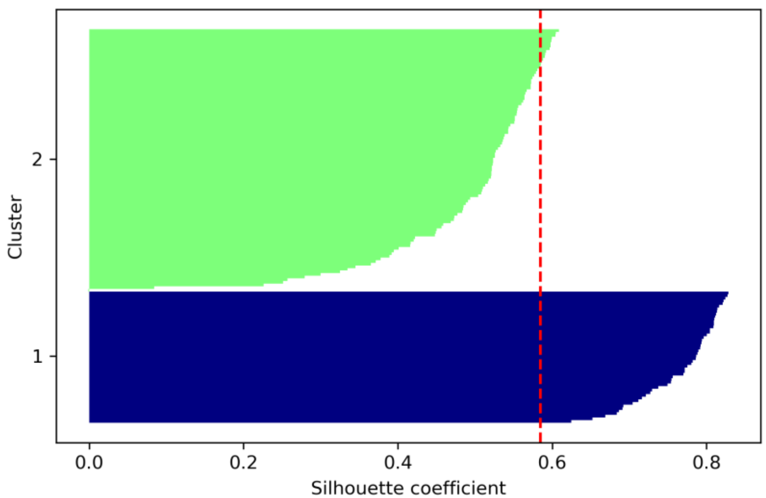
- Cluster Cohesion
- Cluster Separation
- Silhouette coefficient

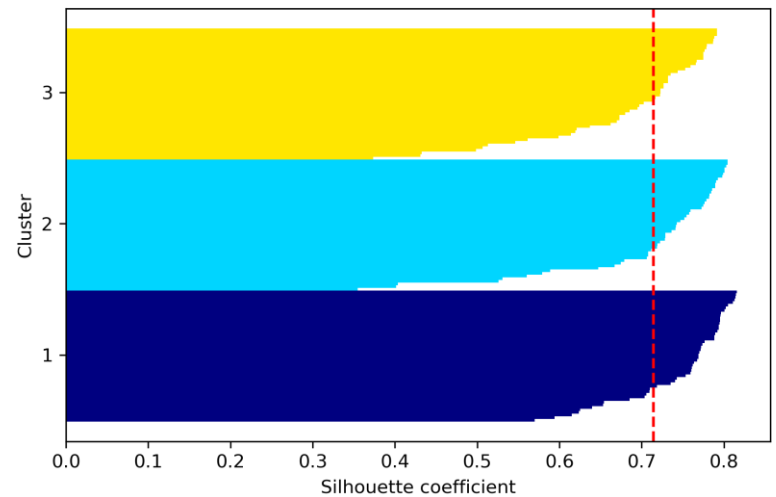


cohesion



separation

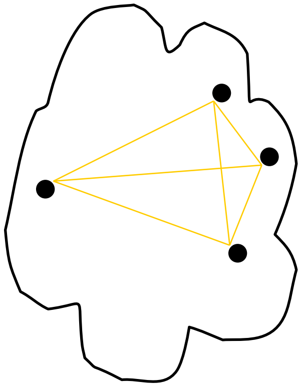




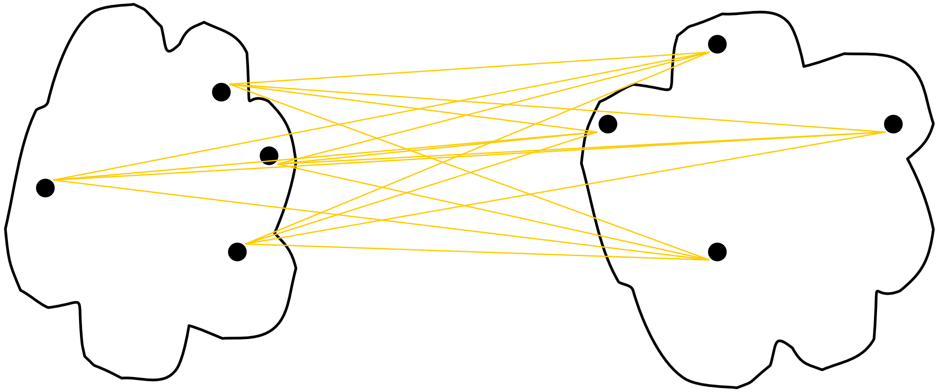
Clustering validation: unsupervised
Cohesion and separation

- Cluster Cohesion: Measures how cohesive/close/compact are the elements in a cluster (Intra-cluster distances)

- Cluster Separation: Measure how distinct or well-separated a cluster is from other clusters (Inter-cluster distances)



cohesion



separation

Clustering validation: unsupervised
Cohesion and separation

Example: Squared error (***SSW + SSB*** is constant)

- Cluster Cohesion: is measured by the within cluster sum of squares

$$SSW = \sum_{i=1}^{n} \sum_{x \in C_j} (x_i - m_j)^2$$

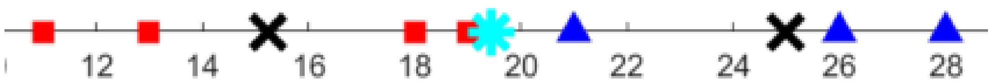
- Cluster Separation: is measured by the between cluster sum of squares

$$SSB = \sum_{j=1}^{K} |C_j| (m - m_j)^2$$

where m is the mean of all points, m_j is the center/centroid of cluster C_j , and $|C_j|$ is the size of cluster C_j

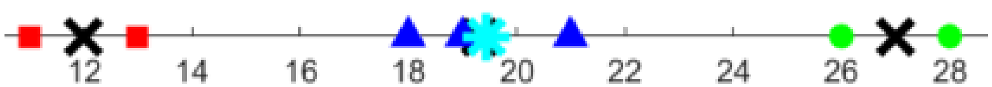
Clustering validation: unsupervised
Example: Cohesion and separation

- K=2 clusters



  - Centroids: *m1 = 15.25, m2 = 25*
  - Mean of all points: *m = 19.43*
  - *SSW = 70.75, SSB = 162.97, SST = 233.71**

- K=3 clusters



  - Centroids: *m1 = 12, m2 = 19.3, m3 = 27*
  - Mean of all points: *m = 19.43*
  - *SSW = 8.7, SSB = 225.05, SST = 233.71**

Obj.	K=2	K=3
11	1	1
13	1	1
18	1	2
19	1	2
21	2	2
26	2	3
28	2	3

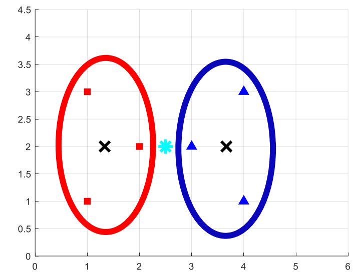
The case *K*=3 is better:

- higher cohesion (SSW is lower) & higher separation (SSB is higher)

* the sum SST = SSW + SSB is constant

Clustering validation: unsupervised
Example: Cohesion and separation

- K=2 clusters

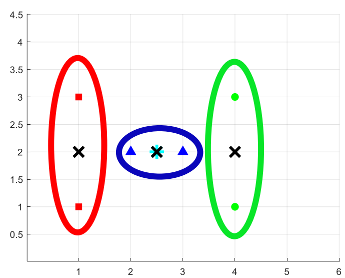


Centroids: *m1 = (4/3, 2), m2 = (11/3, 2)*

Mean of all points: *m = (2.5, 2)*

SSW = 5.333, ***SSB = 8.16,*** ***SST = 13.5****

- K=3 clusters



Centroids: *m1 = (1, 2), m2 = (2.5, 2), m3 = (4,2)*

Mean of all points: *m = (2.5, 2)*

SSW = 4.5, ***SSB = 9,*** ***SST = 13.5****

The case *K*=3 is better:

- higher cohesion (SSW is lower) & higher separation (SSB is higher)

* the sum SST = SSW + SSB is constant

Clustering validation: unsupervised
Silhouette coefficient

Silhouette coefficient: check cluster cohesion and separation

- For each object $\boldsymbol{x}_i$:
  - compute $\boldsymbol{a}_i$ = average distance of $\boldsymbol{x}_i$ to the points in its cluster
  - compute $\boldsymbol{b}_i$ = min (average distance of $\boldsymbol{x}_i$ to the points in other clusters)
  - the silhouette coefficient is $\boldsymbol{s}_i = (\boldsymbol{b}_i - \boldsymbol{a}_i) / \max(\boldsymbol{a}_i, \boldsymbol{b}_i)$
- Silhouette coefficient (*SC*) is the mean values of $\boldsymbol{s}_i$ across all the objects:

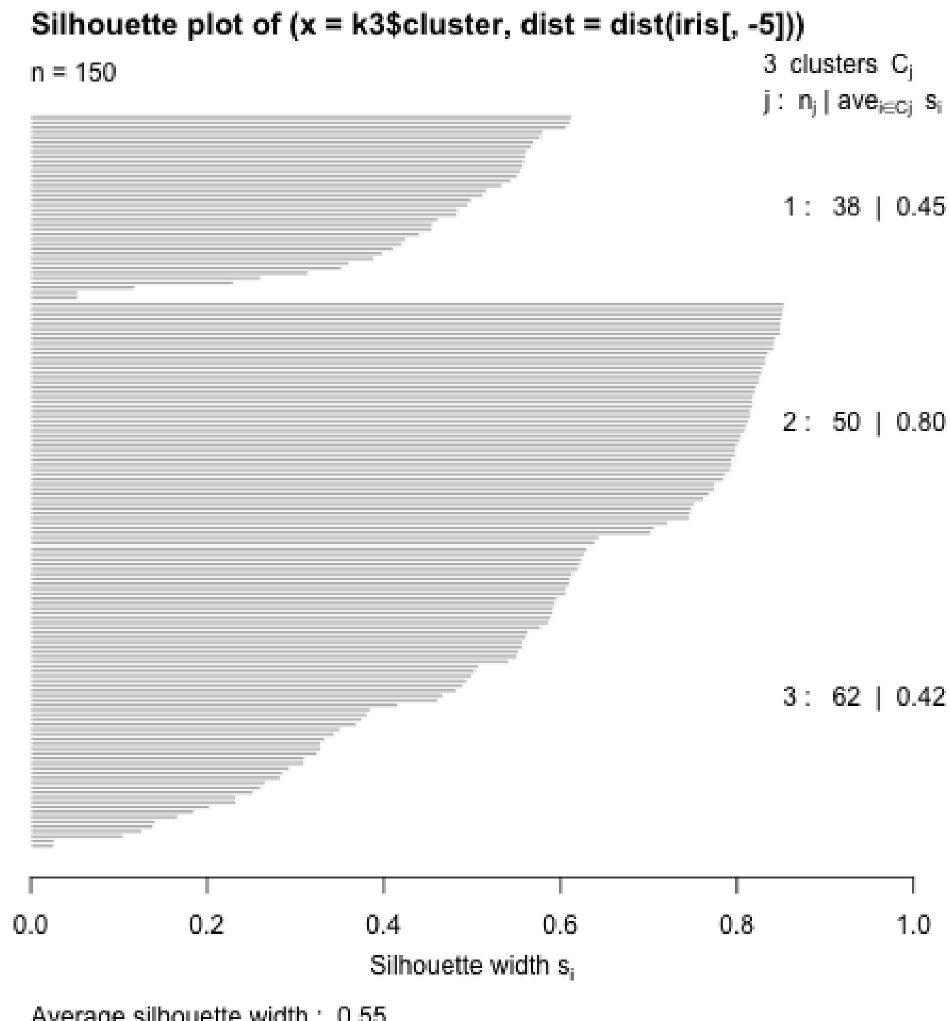
$$SC = \frac{1}{n}\sum_{i=1}^{n} s_i$$

- *SC* can vary between -1 and 1
  - close to +1 signifies good clustering: objects are close to their own clusters but far from other clusters

Clustering validation: unsupervised Silhouette coefficient

Example: iris data set silhouette coefficients s_i with $k = 3$ clusters

- Large $\boldsymbol{s_i}$ (almost 1) means that the object is very well clustered
- Small $\boldsymbol{s_i}$ (around 0) means that the object lies between two clusters
- Negative $\boldsymbol{s_i}$ means that the object is probably placed in the wrong cluster

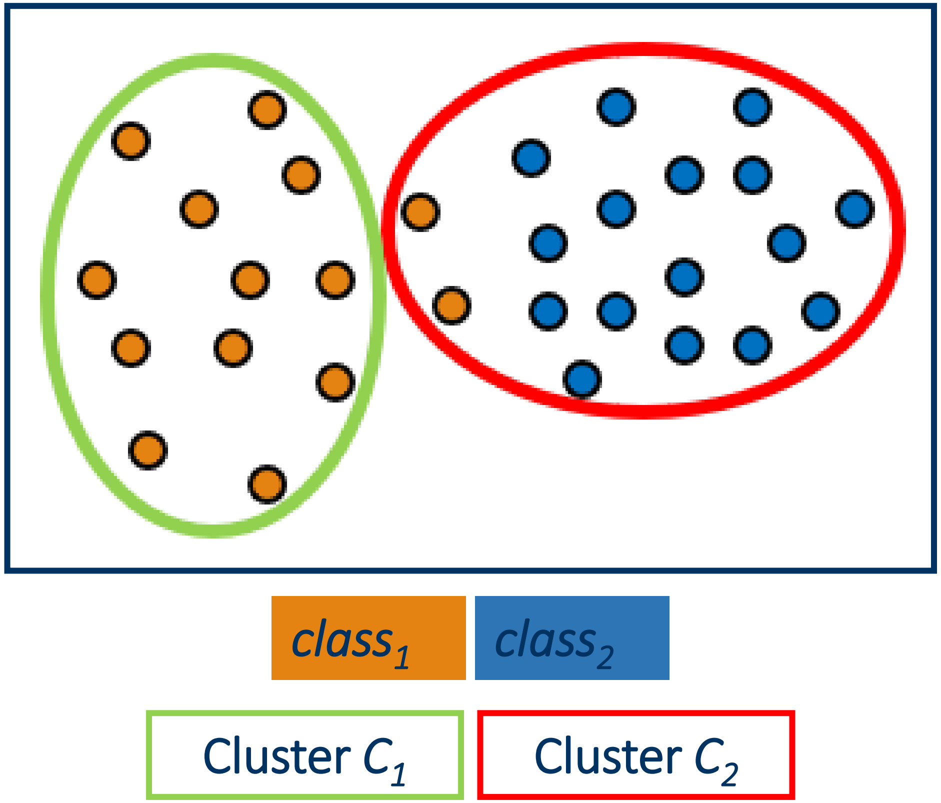


The closer SC to 1, the better

Clustering validation: supervised

Compare the results obtained by different methods when external information exists

- Pairwise measures
- Matching-based measures
- Entropy-Based Measures
- Correlation measures



Clustering validation: supervised
Pairwise measures

- f_{00} = number of pairs of objects having a different class and a different cluster
- f_{01} = number of pairs of objects having a different class and the same cluster
- f_{10} = number of pairs of objects having the same class and a different cluster
- f_{11} = number of pairs of objects having the same class and the same cluster

Two-way contingency table for determining whether pairs of objects are in the same class and same cluster

	Same cluster	Different cluster
Same class	f_{11}	f_{10}
Different class	f_{01}	f_{00}

Clustering validation: supervised Matching-based measures

- **Precision:** the fraction of a cluster that consists of objects of a specified class
- **Recall:** the extent to which a cluster contains all objects of a specified class
- ***F*-measure:** A combination of both precision and recall that measures the extent to which a cluster contains *only* objects of a particular class and *all* objects of that class

Clustering validation: best number of clusters

How to select the right *K* for k-means?

- An inappropriate choice of *K* can result in a clustering with poor performance
- What happens when selecting a *K* that is too high? When the *K* is too low?

Ideally: some a priori knowledge on the real structure of the data

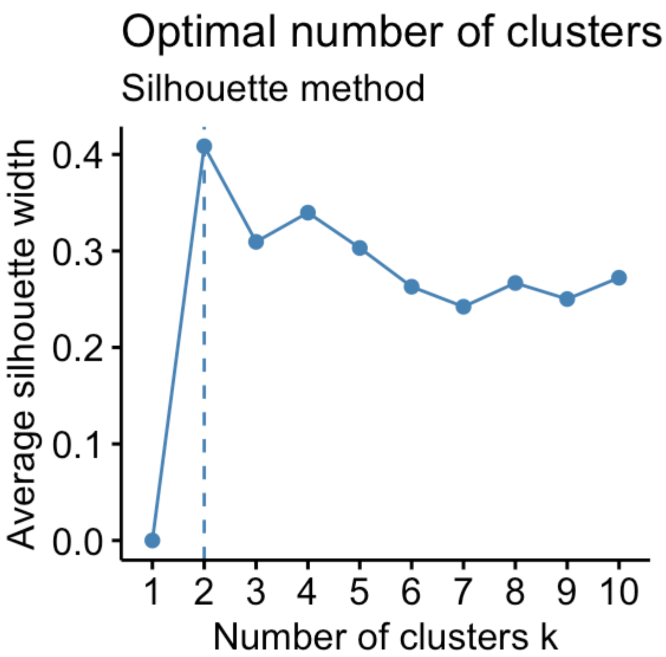
- If no a priori value is known start with $K=\sqrt{n/2}$ as a rule of thumb, where n is the number of attributes.

Clustering validation: best number of clusters

Silhouette coefficient method

For several possible number of clusters *K*:

- Calculate the **SC** and choose the ***K*** that yields to the highest value

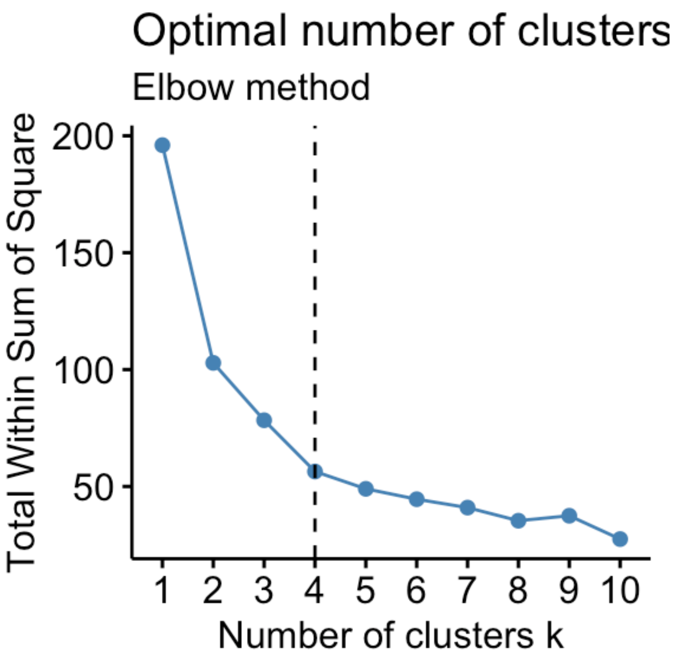


Clustering validation: best number of clusters

Elbow method

For several possible number of clusters *K*:

- Calculate the **SSE** (Sum of Squared Error), also called distortion, and choose the *K* so that adding another cluster doesn't yield to a much smaller SSE.



Clustering validation: tendency

Assess if the data has any inherent grouping structure

- Cluster the data set
  - Use multiple algorithms and evaluate the quality of the resulting clusters
  - If the clusters are uniformly poor, then this may indeed indicate that there are no clusters in the data

Focus of measures of clustering tendency

- try to evaluate whether a data set has clusters without clustering
  - use statistical tests for spatial randomness
  - However, choosing the correct model, estimating the parameters, and evaluating the statistical significance of the hypothesis that the data is non-random can be quite challenging
  - many approaches have been developed, most of them for points in low-dimensional Euclidean space
    - Hopkins statistic

Clustering validation: tendency

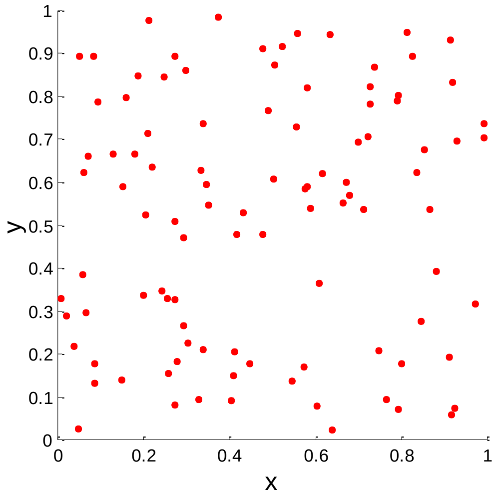
Hopkins statistic H

- generate ***p*** points randomly distributed across the data space
- sample ***p*** actual data points from the data set
- For both sets of points find the distance to the nearest neighbor in the original data set
  - ***ui*** - nearest neighbor distances of the artificially generated points
  - ***wi*** - nearest neighbor distances of the samples from the original data set

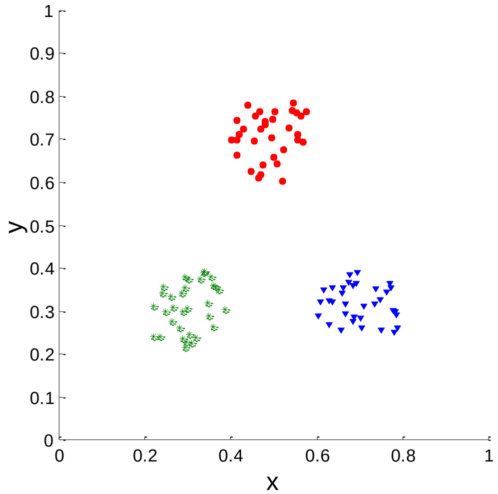
$$H = \frac{\sum_{i=1}^{p} w_i}{\sum_{i=1}^{p} u_i \sum_{i=1}^{p} w_i}$$

Clustering validation: Hopkins statistic *H*

- *H ~ 0.5 :* randomly generated points and the samples of the data set have roughly the same nearest neighbor distances
- *H ~* 0 : whole data (random + sampled) is highly clustered
- *H ~ 1 :* whole data is regularly distributed in the data space



H: 0.56 ± 0.03
(p=20, p=100)



H: 0.95 ± 0.006
(p=20, p=100)

Contents

Summary

- **Descriptive analytics**
- **Cluster analysis**
- **Main categories of clustering methods**
  - Partitional
    - Representative based
    - Density based
  - Hierarchical
    - Agglomerative
    - Divisive
- **Clustering validation**

What is good clustering?

A good clustering method will produce high quality clusters which should have

- **High intra-class similarity:** Cohesive within clusters
- **Low inter-class similarity:** Distinctive between clusters

Clustering methods: comparison

Overall, we can compare clustering methods w.r.t

- Algorithm:
  - complexity and scalability
  - proximity measures that can be employed
  - robustness to noise
  - it is able to find clusters on sub-spaces
  - different runs lead to different results
  - it is incremental

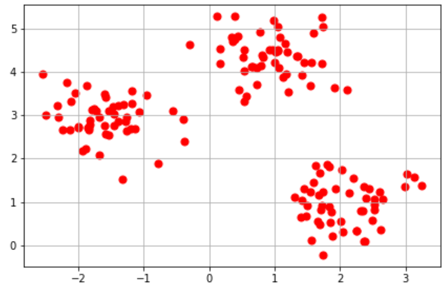
Clustering methods: comparison

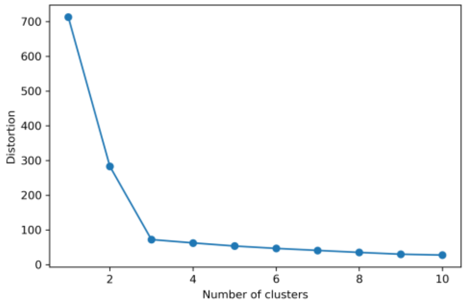
- Data:
  - it is able to handle different types of data (continuous, categorical, binary)?
  - is there dependency on the order of data points?
- Domain:
  - does the algorithm finds the number of clusters, or needs it as input?
  - how many parameters are necessary?
  - what is the required domain knowledge for that?
- Results:
  - shape of clusters that is able to find
  - interpretability

Clustering: toy example

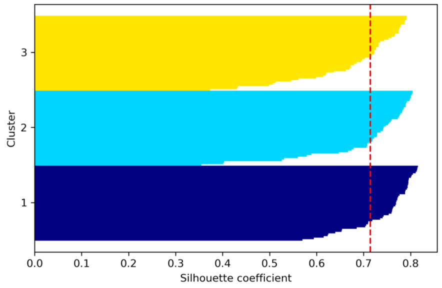
Two versus three clusters

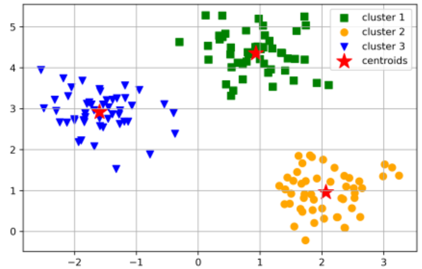
- The optimal number of clusters according Elbow method is three

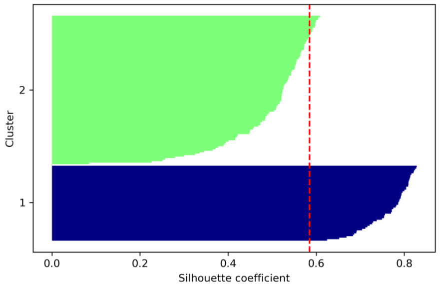


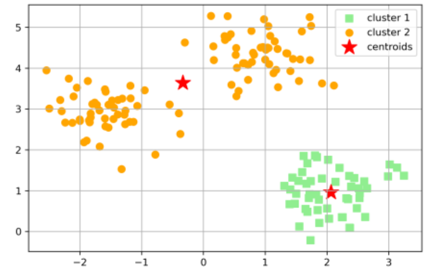


- The silhouette values and clusters

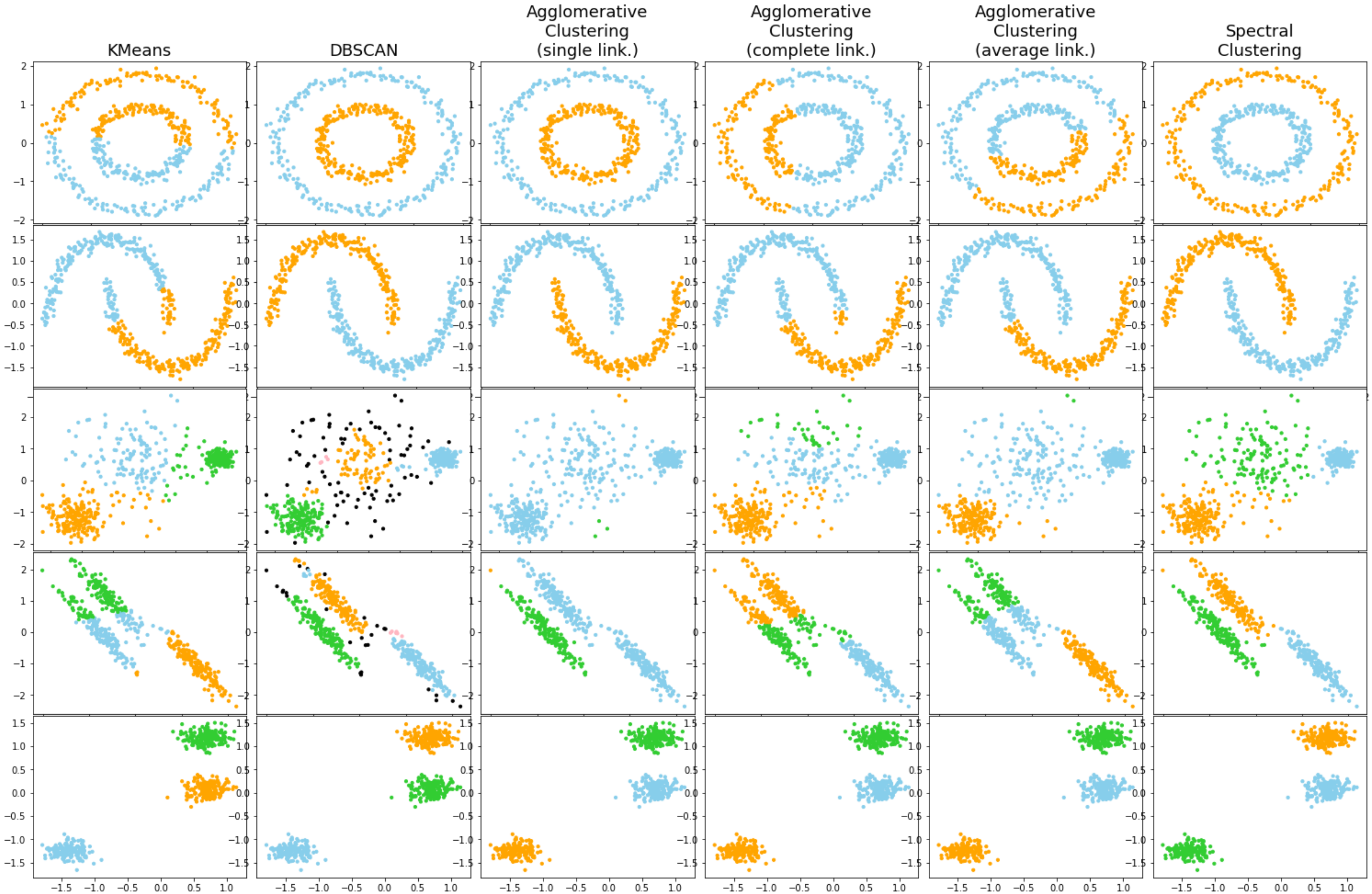








Toy exemple: K-mean, DBSCAN, Hierarchical



Bibliography

Introduction to Data Mining, Pang-Ning Tan, Michael Steinbach, Anuj Karpatne, Vipin Kumar, *Pearson*, 2019 (chap 5)

Data Mining, the Textbook, Charu C. Aggarwal, *Springer*, 2015 (chap 6)

XindongWu, et al.. Top 10 algorithms in data mining. *Knowl Inf Syst*, 14:1-37, 2008

R. Ng and J. Han. **Efficient and Effective Clustering Method for Spatial Data Mining**. *VLDB'94*, 1994

B. Schölkopf, A. Smola, and K. R. Müller. **Nonlinear Component Analysis as a Kernel Eigenvalue Problem**. *Neural computation*, 10(5):1299–1319, 1998

I. S. Dhillon, Y. Guan, and B. Kulis. **Kernel K-Means: Spectral Clustering and Normalized Cuts**. *KDD'04*, 2004