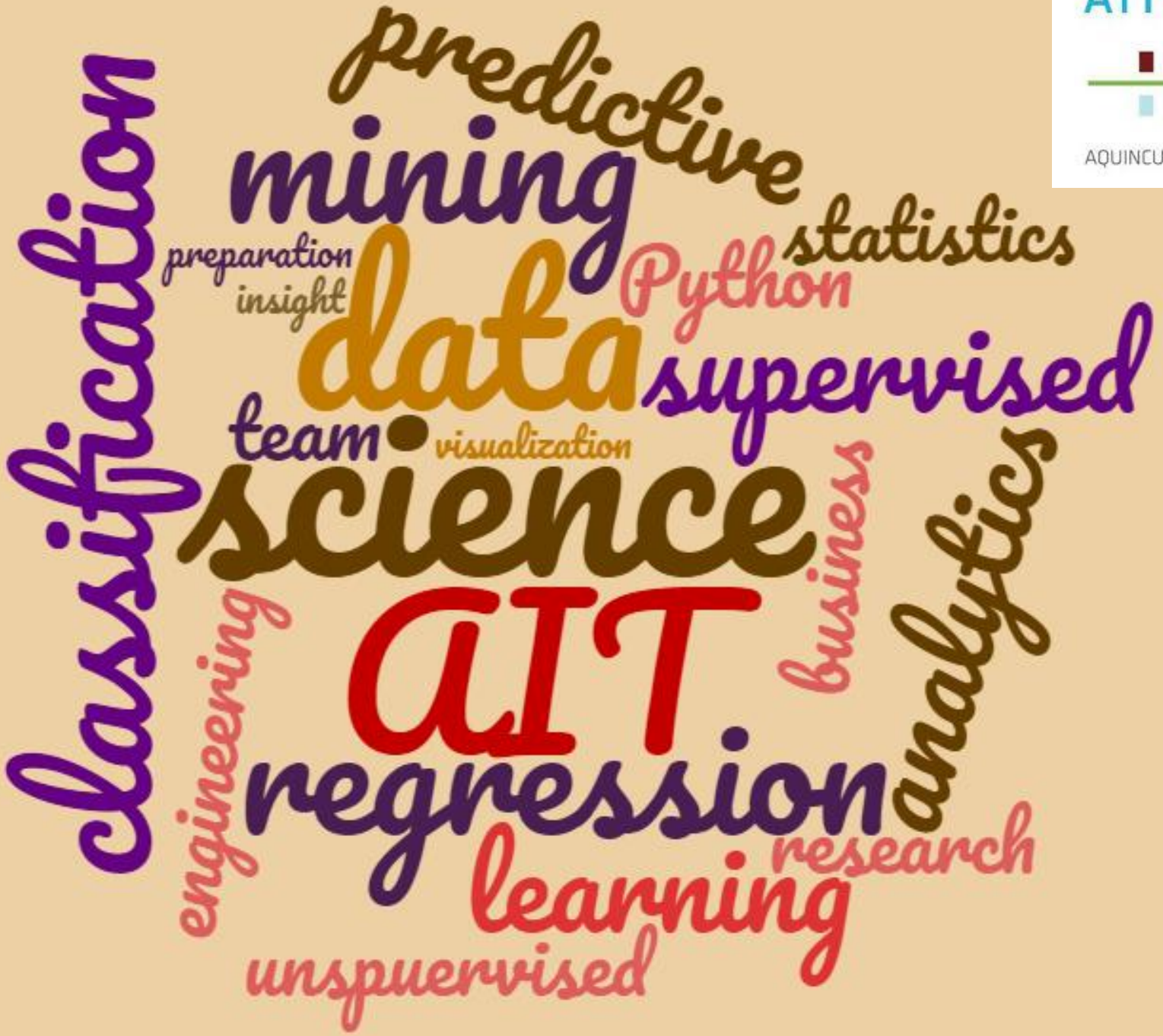
# Data Science

May 5, 2023.
Clustering




AIT-BUDAPEST
AQUINCUM INSTITUTE OF TECHNOLOGY

Dr. Roland Molontay

# Schedule of the semester

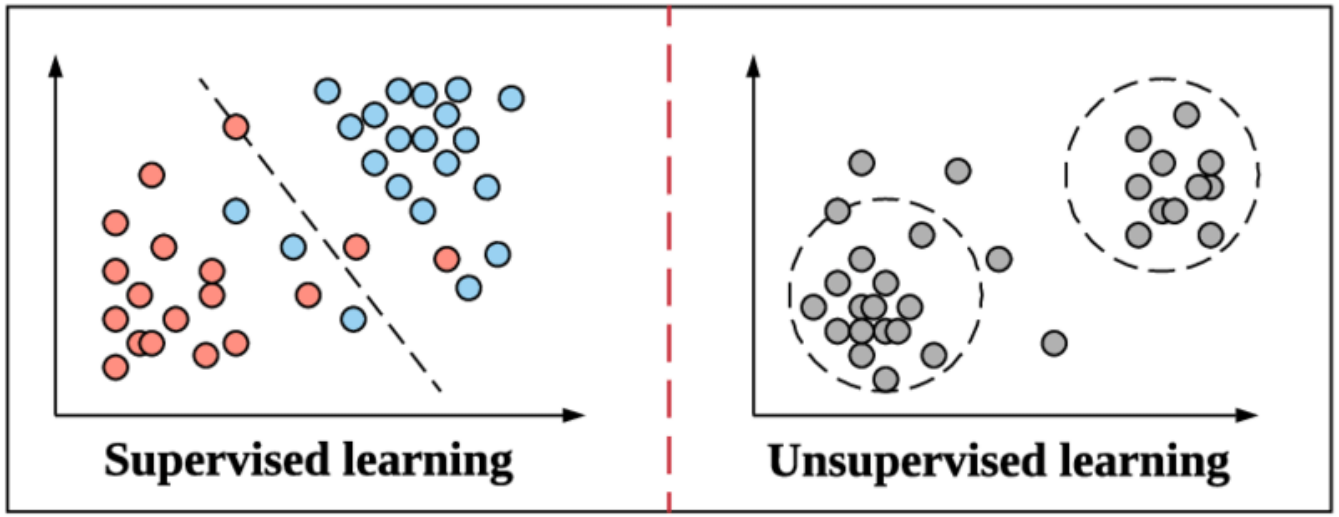
| | *Monday midnight* | *Tuesday class* | *Friday class* |
|---|---|---|---|
| *W1 (02/06)* | | | |
| *W2 (02/13)* | | HW1 out | |
| *W3 (02/20)* | | | |
| *W4 (02/27)* | HW1 deadline + TEAMS | HW2 out | |
| *W5 (03/06)* | | | PROJECT PLAN |
| *W6 (03/13)* | HW2 deadline | HW3 out | |
| *W7 (03/20)* | | | MIDTERM |
| SPRING BREAK | | SPRING BREAK | SPRING BREAK |
| *W8 (04/03)* | HW3 deadline | | GOOD FRIDAY |
| *W9 (04/10)* | MILESTONE 1 | | |
| *W10 (04/17)* | | HW4 out | |
| *W11 (04/24)* | | | |
| *W12 (05/01)* | HW4 deadline | | |
| *W13 (05/08)* | MILESTONE 2 | | |
| *W14 (05/15)* | | FINAL | -~~PROJECT presentations~~ |
| *W15 (05/22)* | | PROJECT presentations | |

# Milestone 2

- Three-page long report including:
  - Reviewing the related works
  - Data understanding and data preparation steps
  - Data analysis steps, implementing some models and evaluating them

# Unsupervised learning

- The target (label) is not known for any records (latent labels)
- Aim: to associate useful labels to the records based on the attributes
- In many cases our aim is to gain better understanding of the data or visualize the data

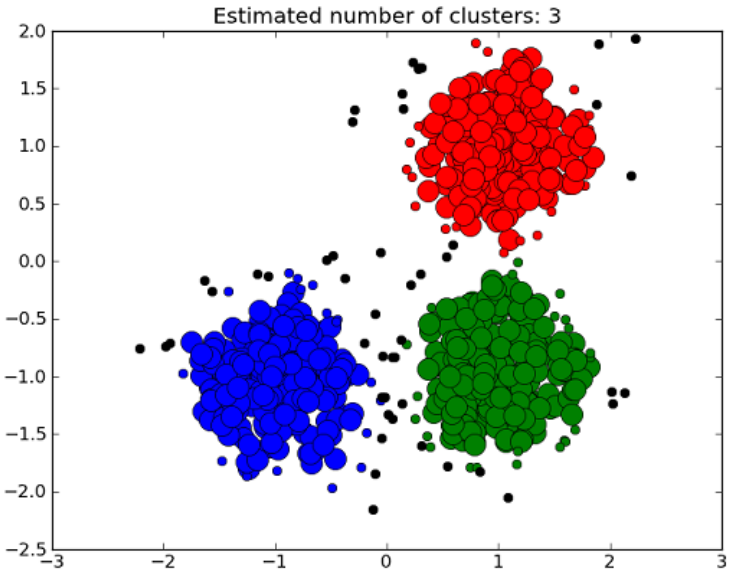


# Clustering

- Unsupervised learning
- Grouping similar objects together
  - Aim: objects within a group should be more similar to each other compared to objects from different groups
- How to measure similarity? ➔ similarity measures
- Challenges: What features is the clustering based on? How to measure similarity? How many clusters do we want? How to evaluate a clustering? How to visualize it?

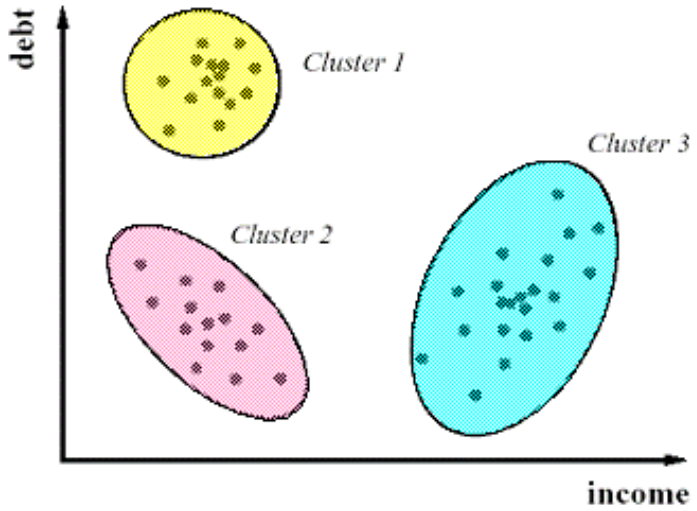
# Goal of clustering

- For some data science projects the goal is to solve a clustering task (group similar objects together)
- Sometimes clustering is part of the explanatory analysis
  - Recognize some (hidden) pattern in the data
  - Visualization, gain a better understanding
  - Reducing the complexity of the data by clustering
- The evaluation of clustering and the (optimal) number of clusters depend on the application domain



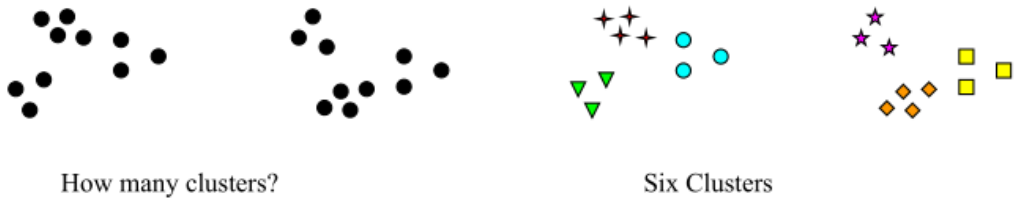
# Clustering - examples

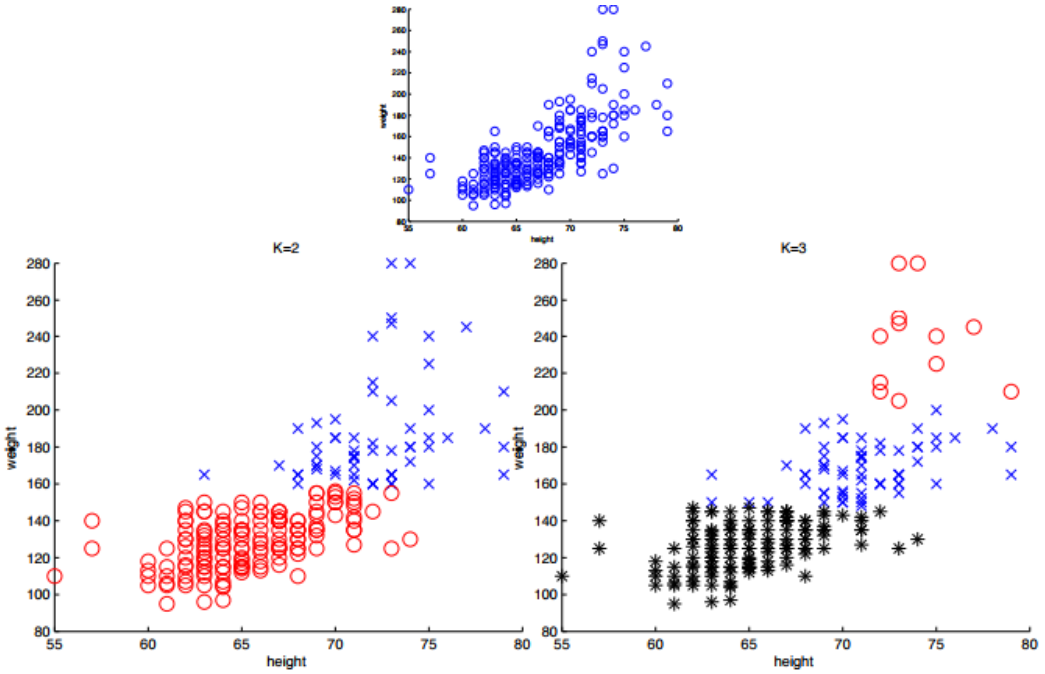
- Customer segmentation
  - Features:  Customer data (sex, age, address, profession, …), purchase history
- Grouping documents based on their content
  - Features: words, n-grams appearing in the text
- Grouping pictures
  - Features: extracted from the pixels



# Evaluation of clustering

- The evaluation is ambiguous, there are no obvious evaluation methods
- It is difficult to judge how good a clustering is (there is no ground truth)

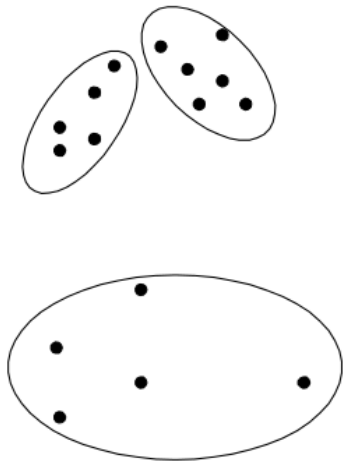


# Types of clustering algorithms

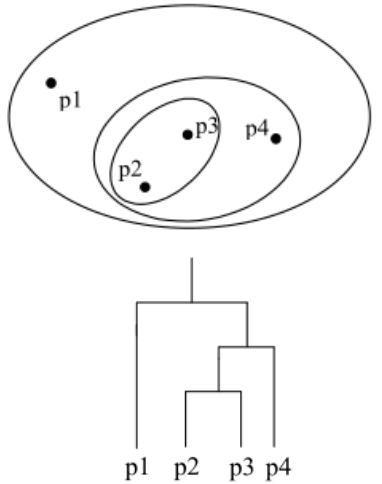
## Partitional

- We partition the records into non-overlapping subsets (clusters)
- Each record is in exactly one cluster



## Hierarchical

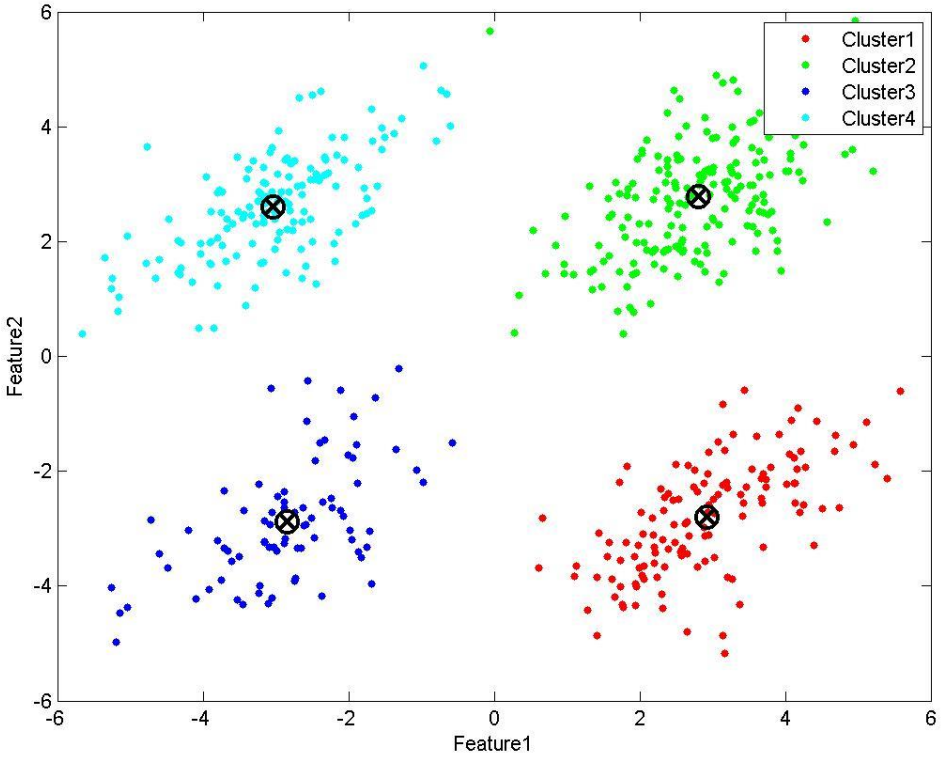
- The clusters are nested
- A record corresponds to a nested hierarchy of clusters




dendrogram

# Center-based clustering

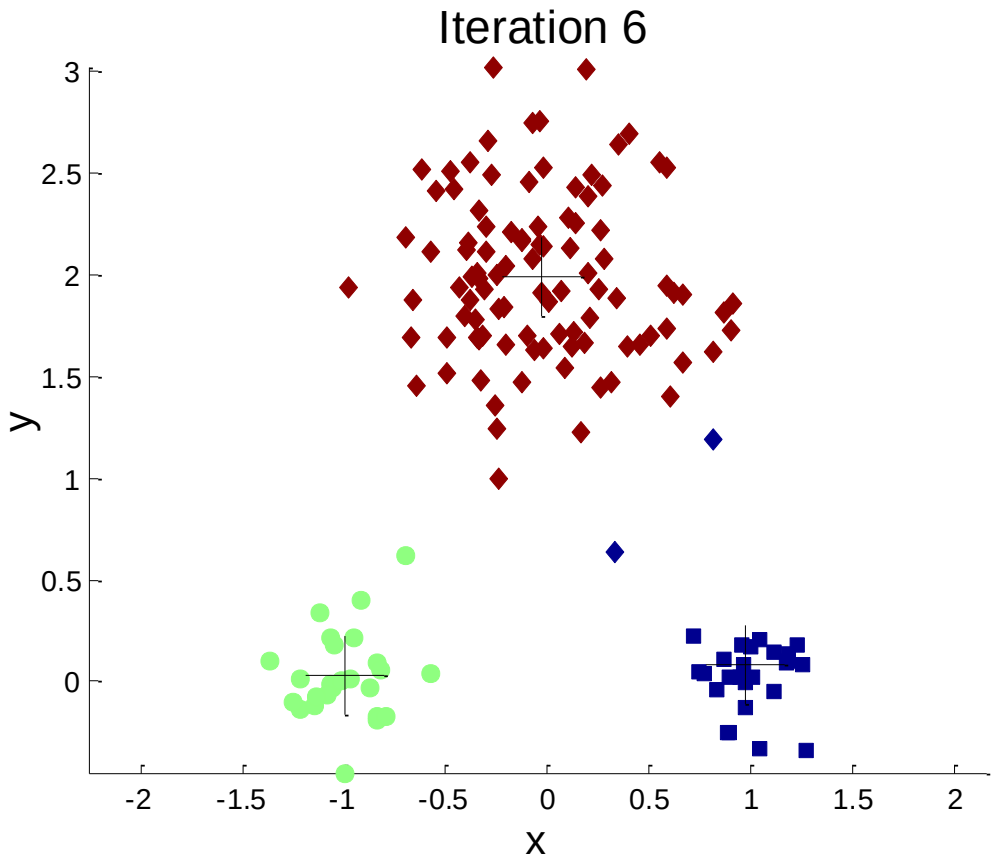
- Center-based or prototype-based
- Every cluster has a representative center point
- Every record corresponds to the cluster whose center point is the closest to the record
  - The center point can be the centroid (the average of the records) or the medoid (the most „representative” datapoint)



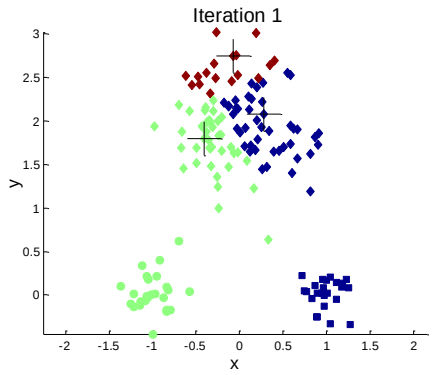
# K-means algorithm

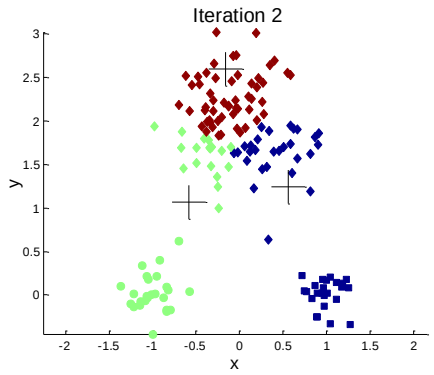
- Partitional, center-based clustering algorithms that creates a given *K* number of clusters
- For a given *K* number
  1. Choose *K* initial centroids in the *n*-dimensional space (we have *n* attributes), the centroids are not necessarily data points themselves
  2. Assign each record to the closest centroid to form clusters
  3. Calculate the new centroids (means of the records) of the clusters
  4. Repeat point 2 and 3 until convergence (when the assignments no longer change)

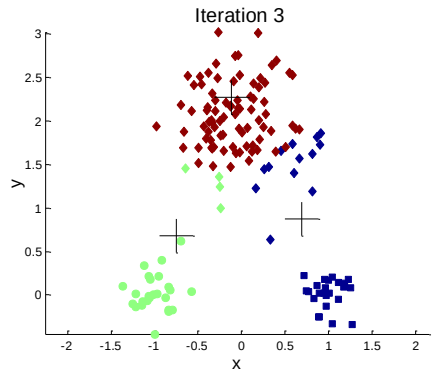
# K-means algorithm - iterations

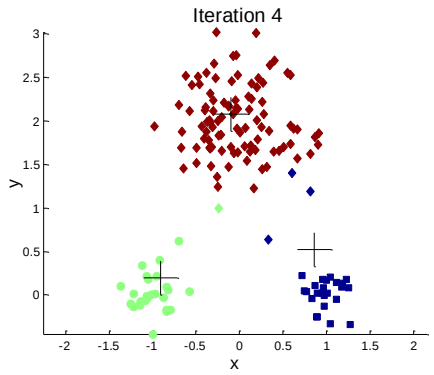


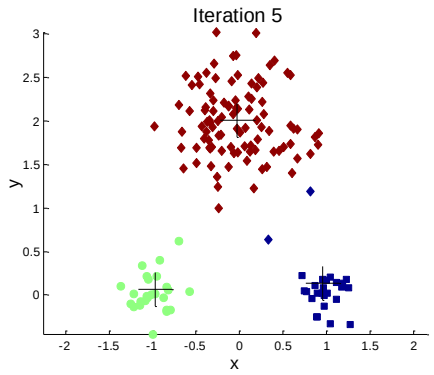
# K-means algorithm – iterations II.

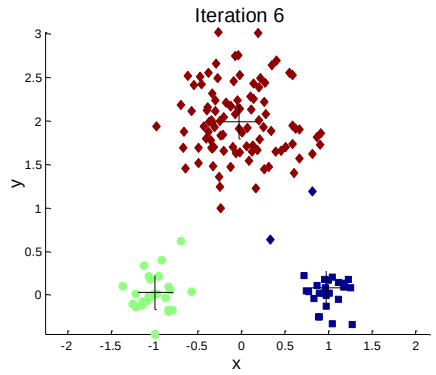


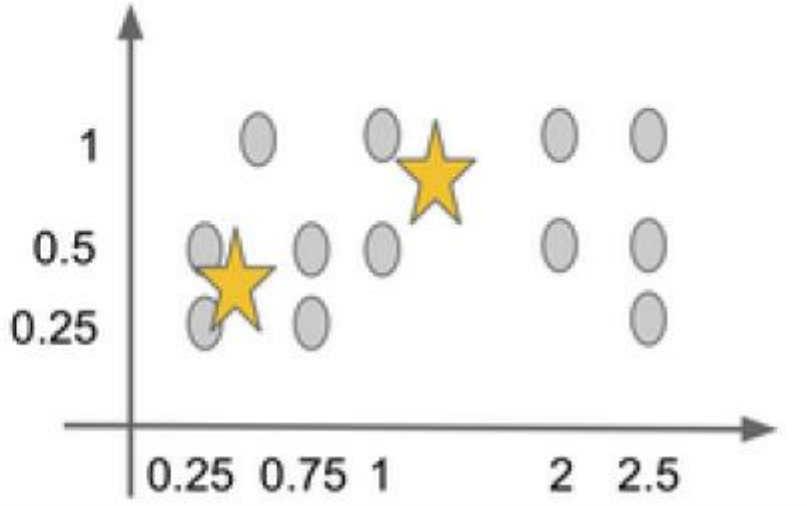



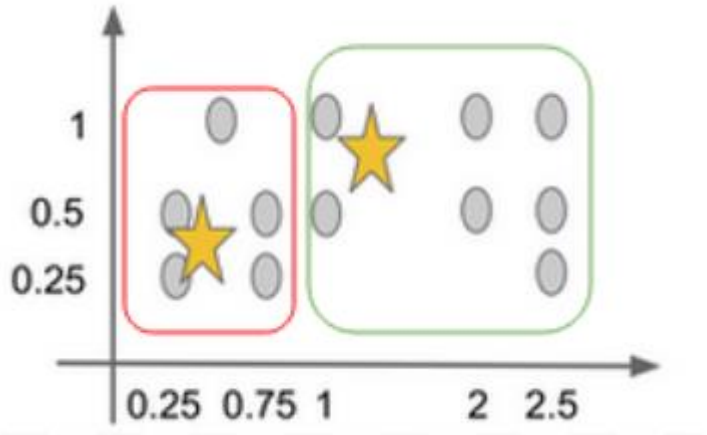
# Problem

We initialize k-means clustering algorithm with the centroids marked by stars in the figure. Perform an iteration step! Calculate the positions of the new centroids!



# Solution

We assign each data point to the nearest centroid. Then we calculate the mean of the coordinates of the data points in each cluster. The means of the coordinates are the new coordinates of the centroids.




Right (green) box:

$$x = \frac{1+1+2+2+2.5+2.5+2.5}{7} = 1.9$$
$$y = \frac{0.25+0.5+0.5+0.5+1+1+1}{7} = 0.68$$

The corresponding centroid: (1.92, 0.68)

Left (red) box:

$$x = \frac{0.25+0.25+0.5+0.75+0.75}{5} = 0.5$$
$$y = \frac{0.25+0.5+1+0.25+0.5}{5} = 0.5$$

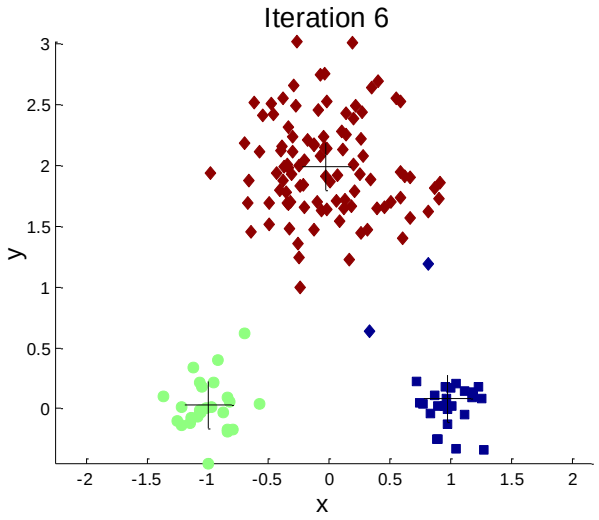
The corresponding centroid: (0.5. 0.5)

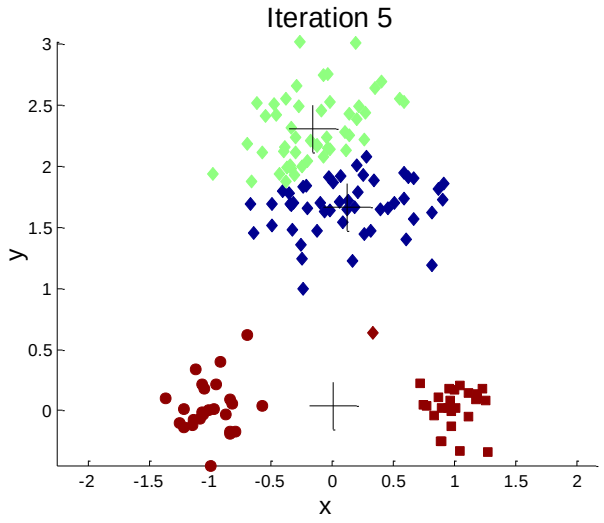
# K-means - specification

- What (dis)similarity measure to use?
  - Depends on the data, choose the most suitable (dis)similarity
  - Most cases: Euclidean distance
- How to calculate the new centroids?
  - Usually the aim is to minimize the squared distance between records and their centroids (sum of squared „errors”)
  - The mean minimizes the squared error
- Does it converge?
  - Usually yes, for Euclidean distance always
  - The convergence is usually fast
  - Does not guarantee to find the optimum (just local optimum)

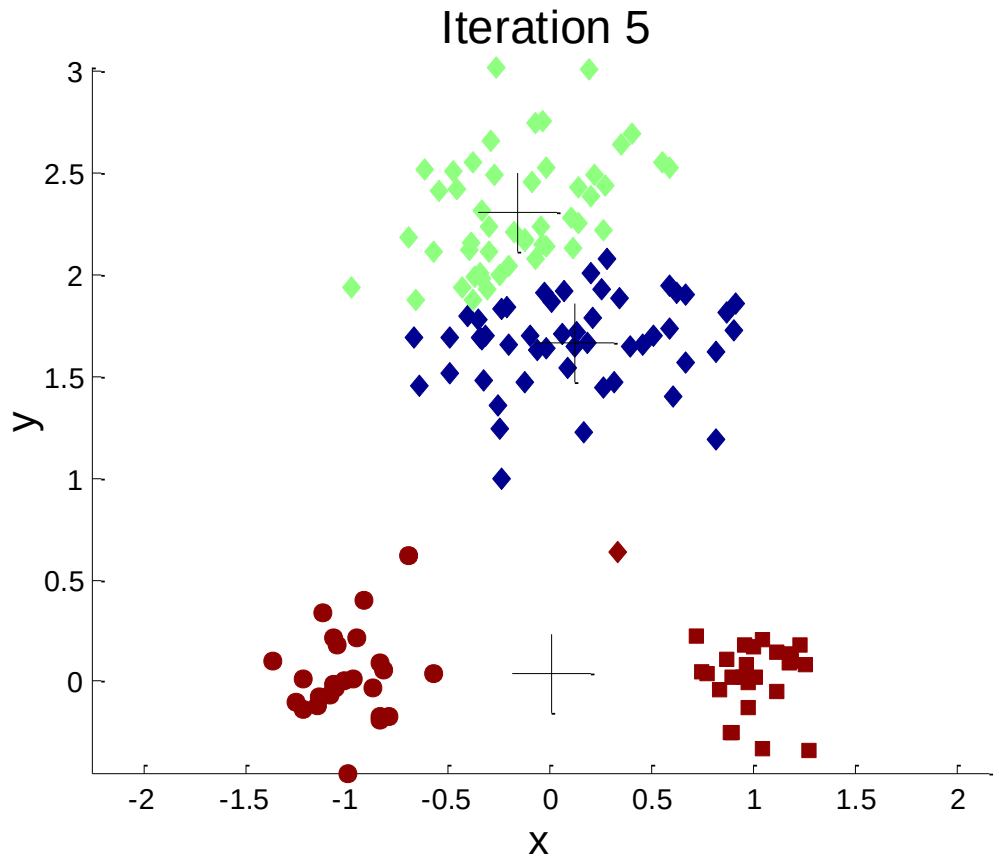
# Choosing initial centroids

- The final clustering also highly depends on the choice of the initial centroids
- With a bad first initialization we can obtain bad clusters even if there were nice natural clusters

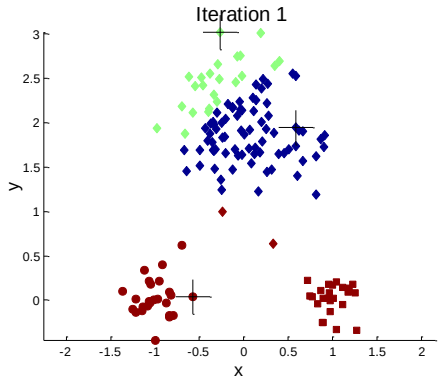


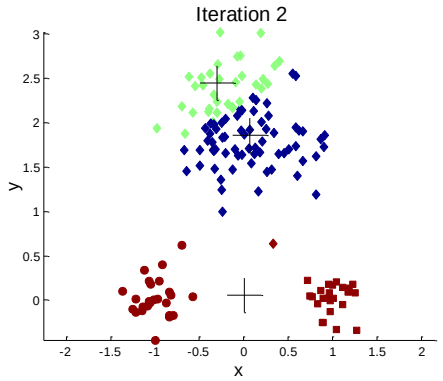



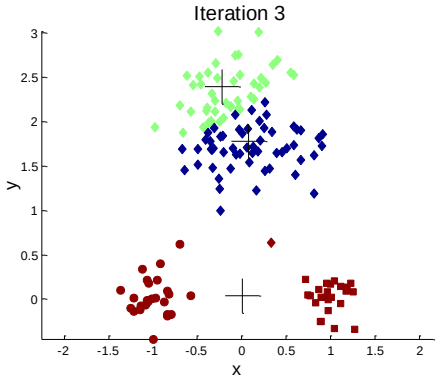
# Importance of the initialization

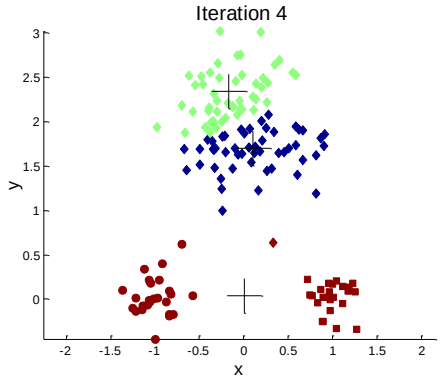


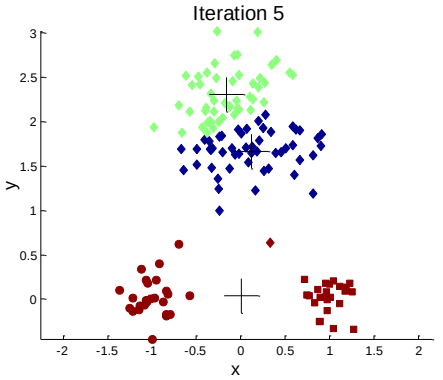
# Importance of the initialization II.



# Choosing initial data points randomly

- Running the algorithms more times with random initializations
  - Choose the final clustering the lowest sum of squared errors (SSE):
    - SSE or also called total within-cluster sum of squares (WSS)

  - $C_i$ is the *i*th cluster with centroid $c_i$

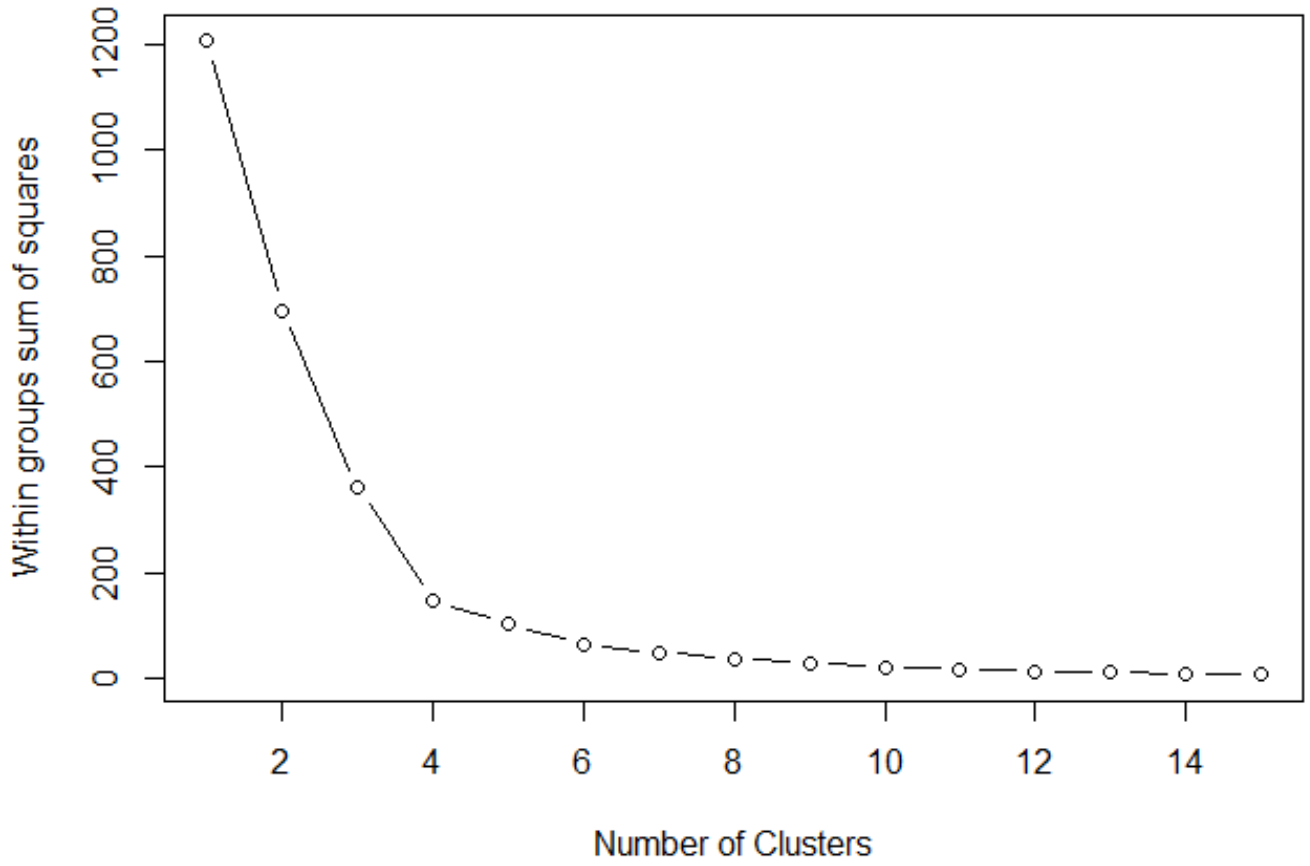
$$\text{SSE} = \sum_{i=1}^{K} \sum_{x \in C_i} dist(x, c_i)^2$$

# Determining the optimal number of clusters

- The optimal value of *K* depends on the application domain
- We can try several *K* values (each with more random initializations)
- We choose the *K* that gives the most desirable result
- If we solely decide the K value based on the SSE
  - Be careful! The more clusters we have the less the SSE value is
  - If *K*=*N*, then SSE=0, but it is meaningless
  - The „elbow” method: choose a number of clusters so that adding another cluster doesn’t improve much better the SEE

# The elbow method

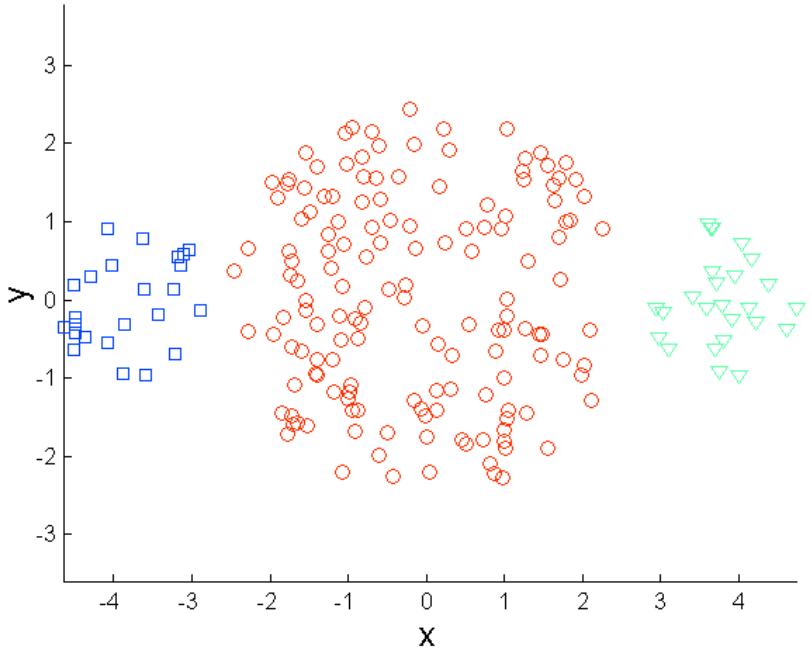
- Scree plot



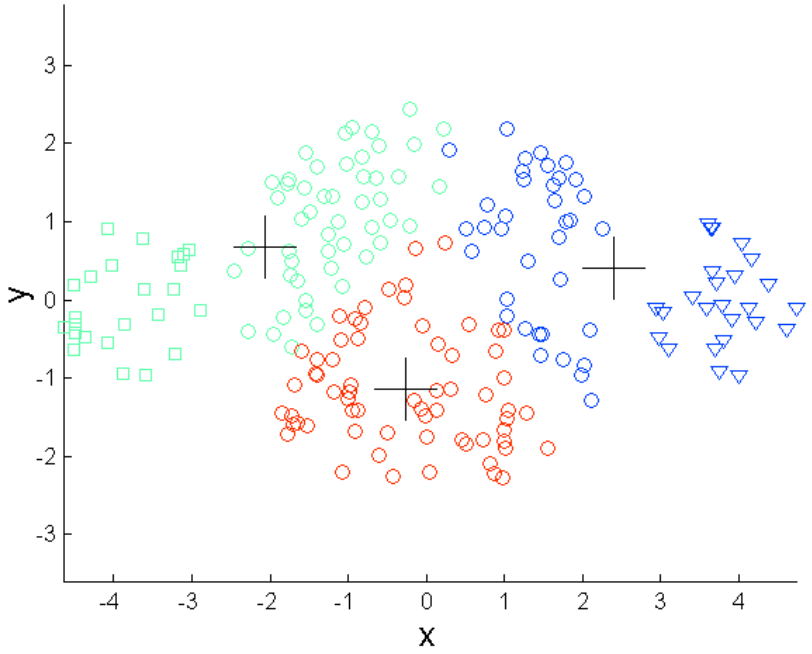
# Drawbacks of K-means algorithm

- One should decide on the number of clusters before
- Initial seeds have a strong impact on the final result
- Always construct spherical shapes around the centroids
  - If the natural clusters are not spherical, K-means will not perform well
- K-means cluster tend to be of the same size
  - If the natural clusters differ in size, it will not perform well
- K-means algorithm does not perform well for clusters with differing density

# Clusters with differing size

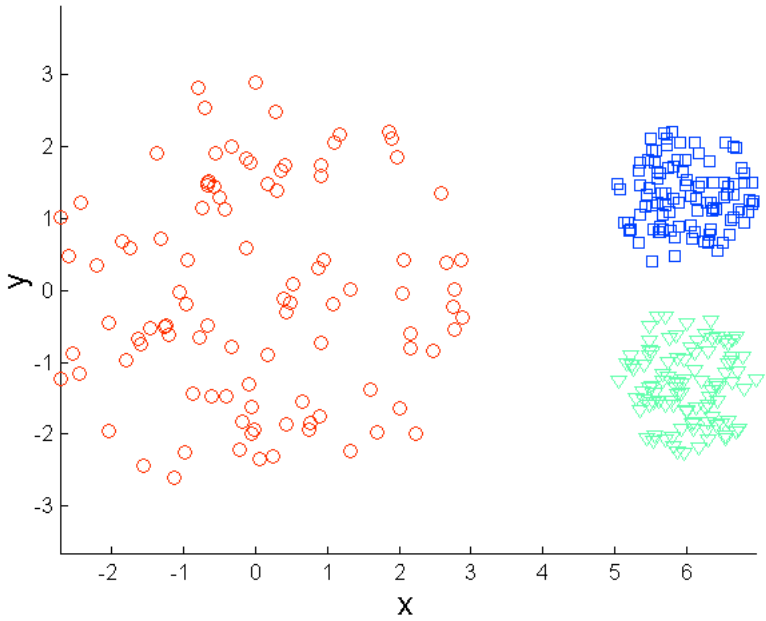



Natural clusters

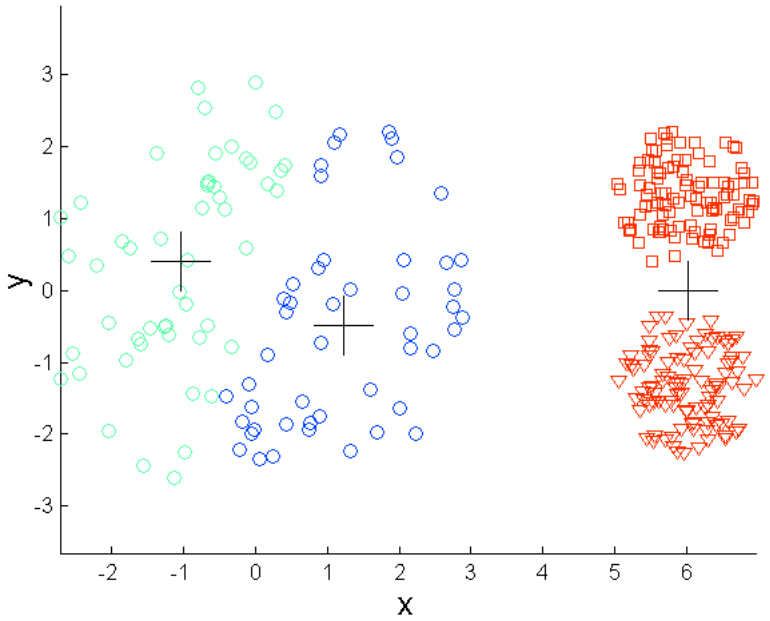



Result of K-means algorithm (K=3)

# Clusters with differing density

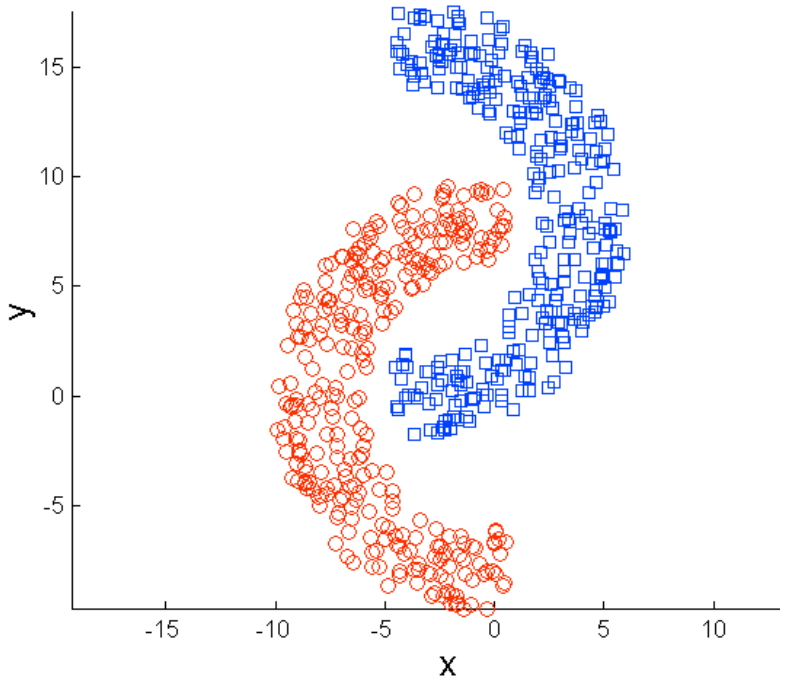



Natural clusters

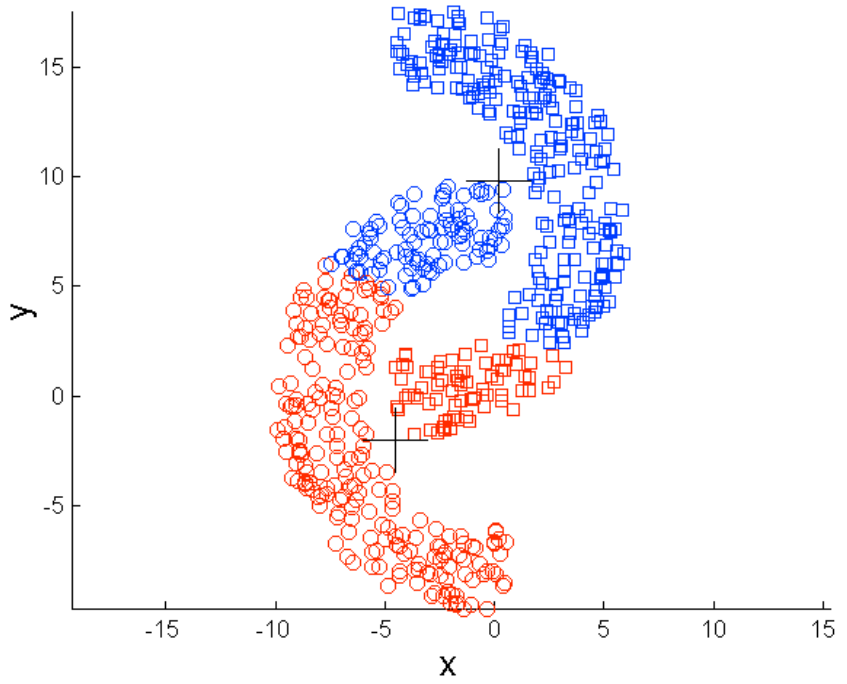



Result of K-means algorithm (K=3)

# Non-spherical clusters



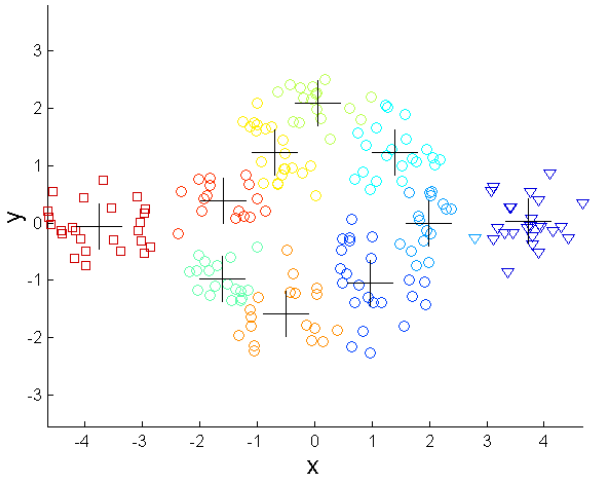

Natural clusters

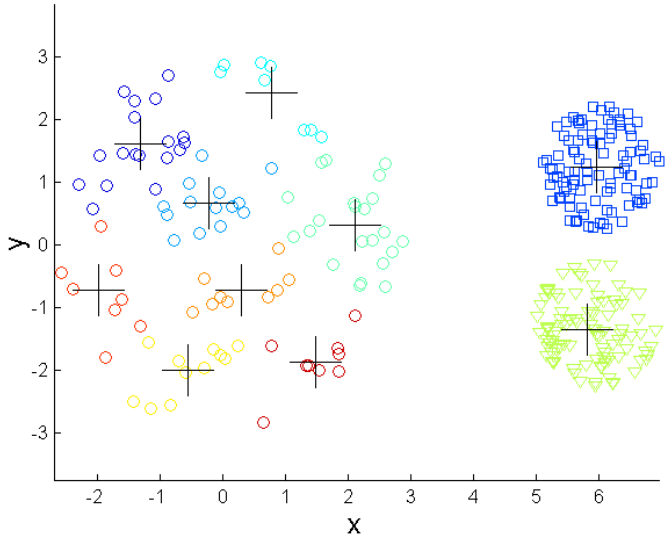


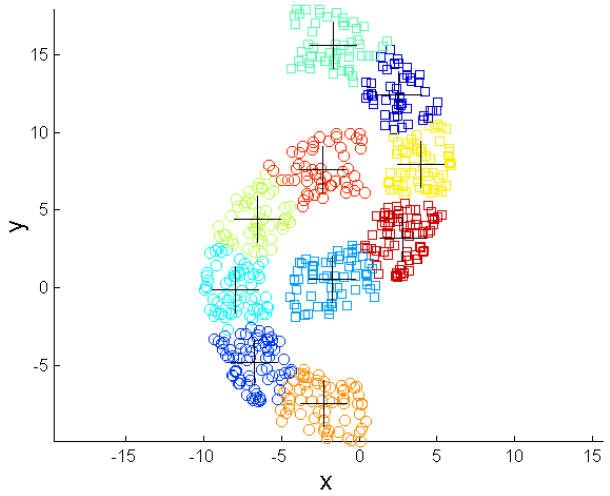

Result of K-means algorithm (K=2)

# Advantages of K-means

- Easy to implement
- Computationally relatively fast (for small K)
- For the presented drawbacks,  a possible solution can be to choose higher K, then the natural clusters will be the union of small clusters

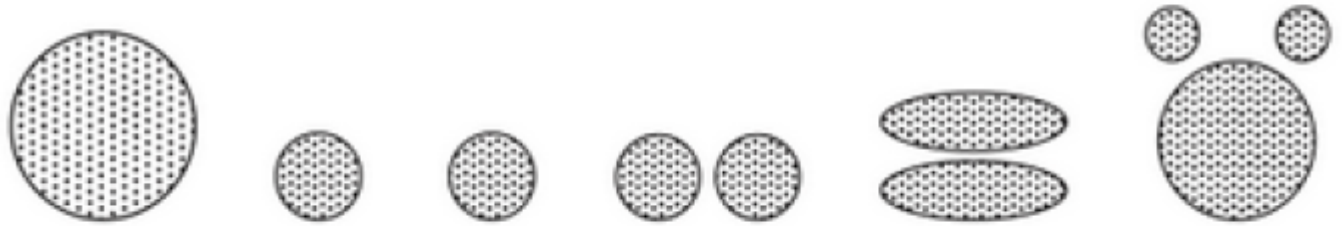


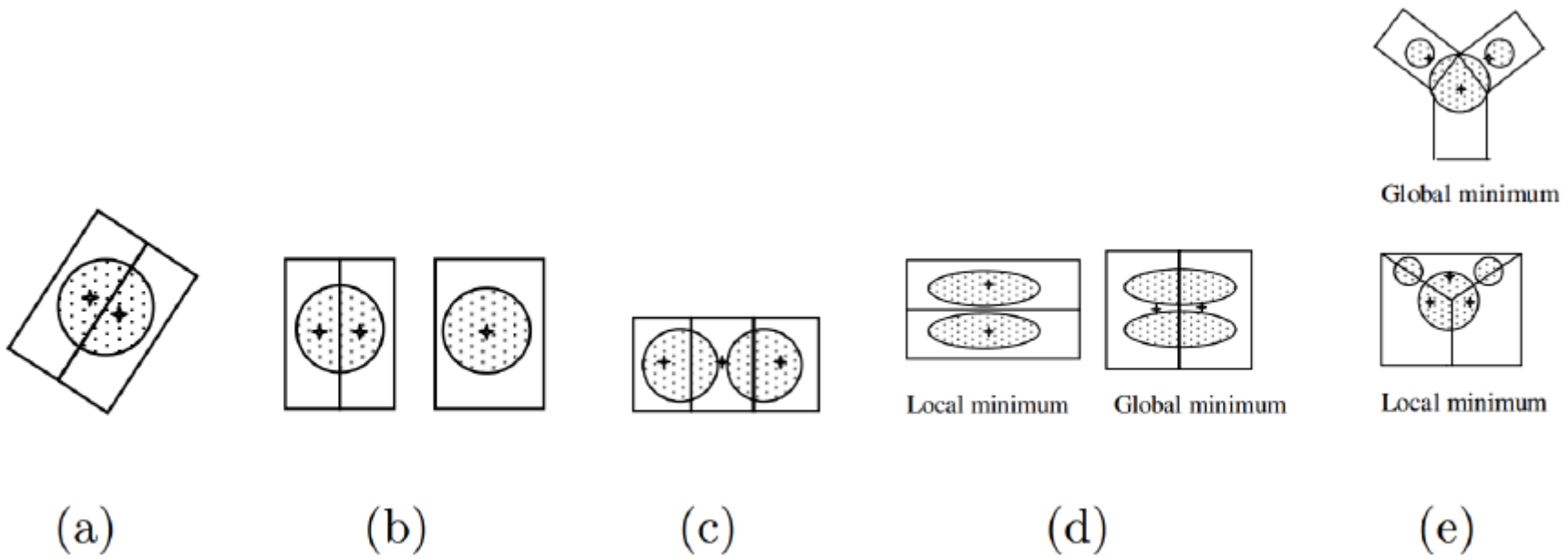
# Problem

Consider the following sets of two-dimensional points. For the given number of clusters, provide a sketch of the resulting clusters found by k-means algorithm, also indicate the positions of centroids. Assume that Euclidean distance is used and the points are uniformly distributed. If you think that there are more than one possible solutions, then indicate whether a solution is a global or local minimum (i.e. in which case the SSE is smaller) (a) $k = 2$ (b) $k = 3$ (c) $k = 3$ (d) $k = 2$ (e) $k = 3$ (Note that the label of each diagram below matches the corresponding part of this question.)



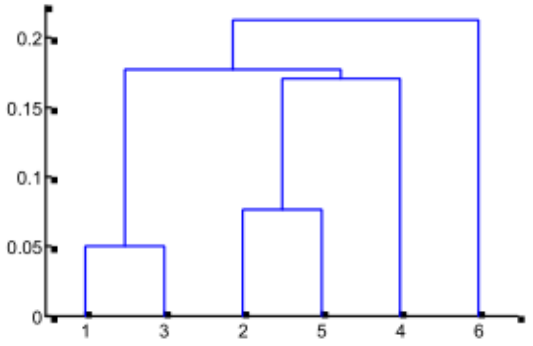
(a) (b) (c) (d) (e)

# Solution

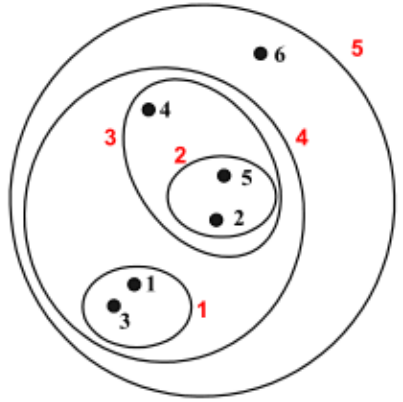

Global minimum
Local minimum
Global minimum
Local minimum
(a)
(b)
(c)
(d)
(e)

# Hierarchical clustering

- Construct nested hierarchy of structures
- One doesn’t have to determine the number of clusters before, after running the algorithm it is enough to decide on the number of clusters (with the help of the dendrogram)
- Two main strategies
  - Agglomerative
    - „Bottom-up”
  - Divisive
    - „Top-down”



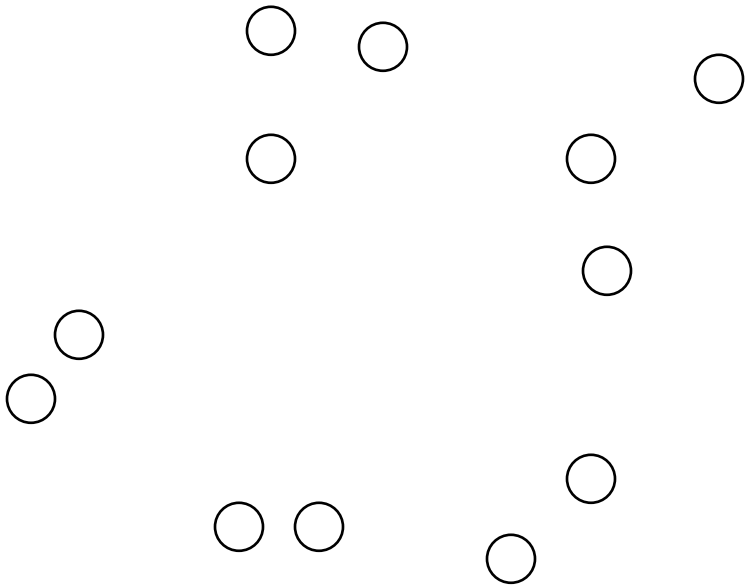

dendrogram



# Hierarchical agglomerative clustering

- Each observation starts in its own cluster
- Pairs of clusters are merged as one moves up the hierarchy
  - Until all the observations are merged into one cluster
- We need
  - (dis)similarity measure between observations
  - (dis)similarity measure between clusters
    - How to define the (dis)similarity (or distance) between an observation and a cluster or between two clusters?

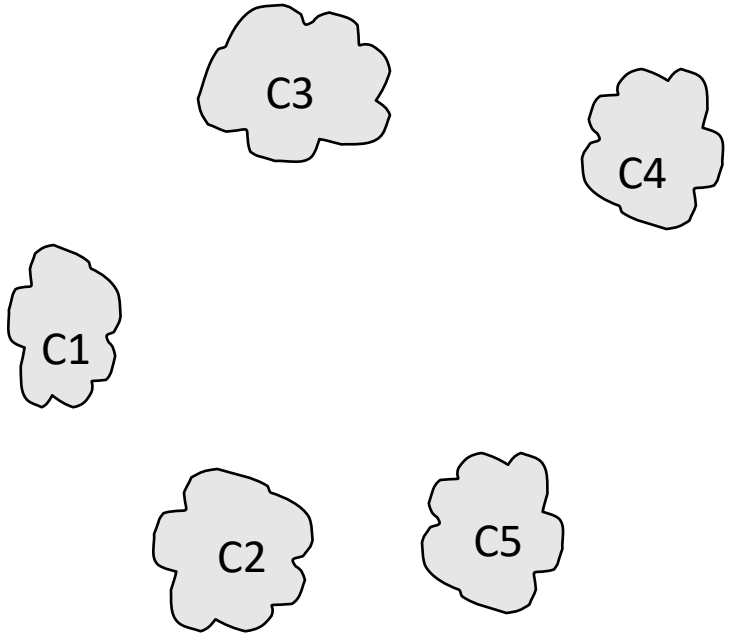
# Example



| | p1 | p2 | p3 | p4 | p5 | . . . |
|---|---|---|---|---|---|---|
| p1 | | | | | | |
| p2 | | | | | | |
| p3 | | | | | | |
| p4 | | | | | | |
| p5 | | | | | | |
| . | | | | | | |
| . | | | | | | |
| . | | | | | | |

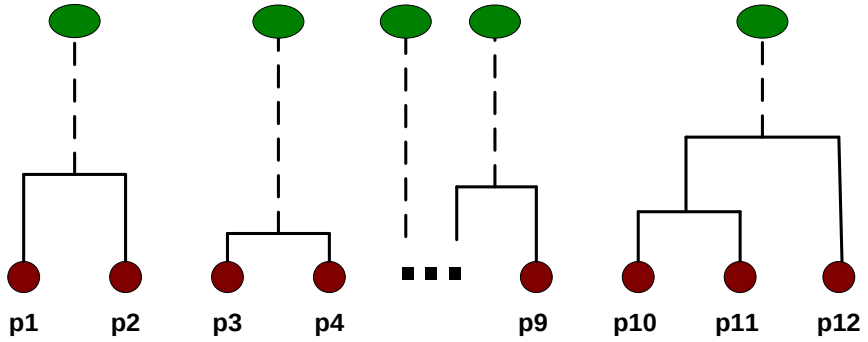
Proximity Matrix

# After some merging steps



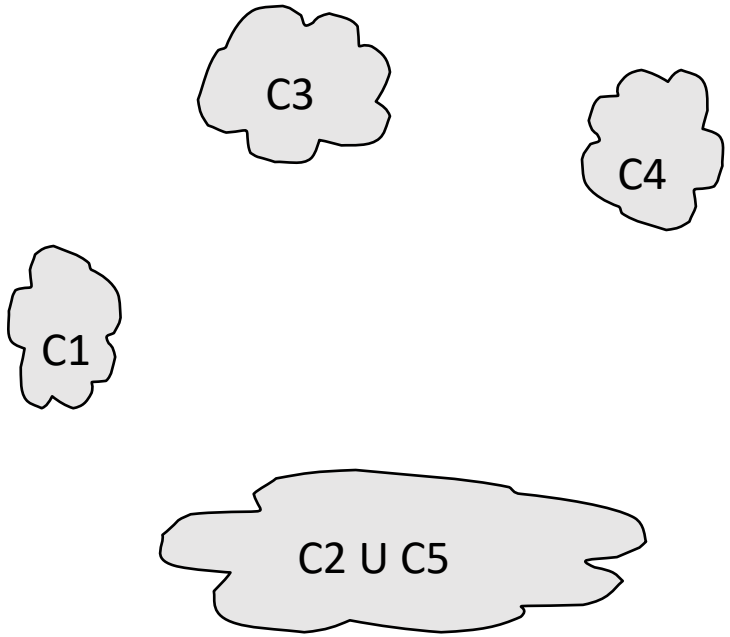

| | C1 | C2 | C3 | C4 | C5 |
|---|---|---|---|---|---|
| C1 | | | | | |
| C2 | | | | | |
| C3 | | | | | |
| C4 | | | | | |
| C5 | | | | | |

Proximity Matrix



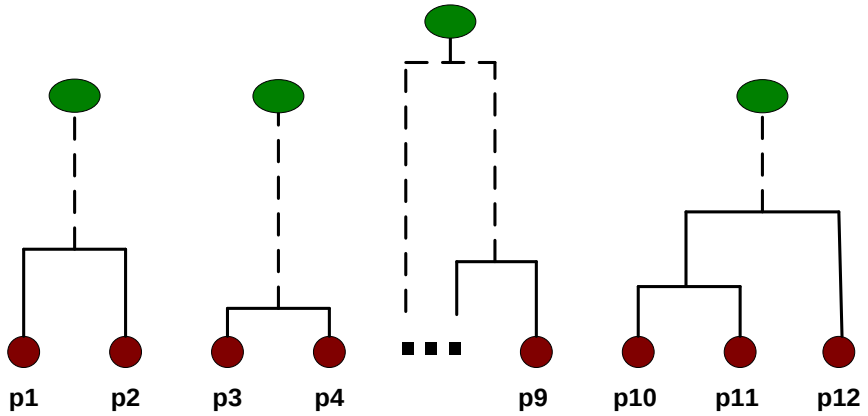
# Merging clusters

- How to define the new (dis)similarities?



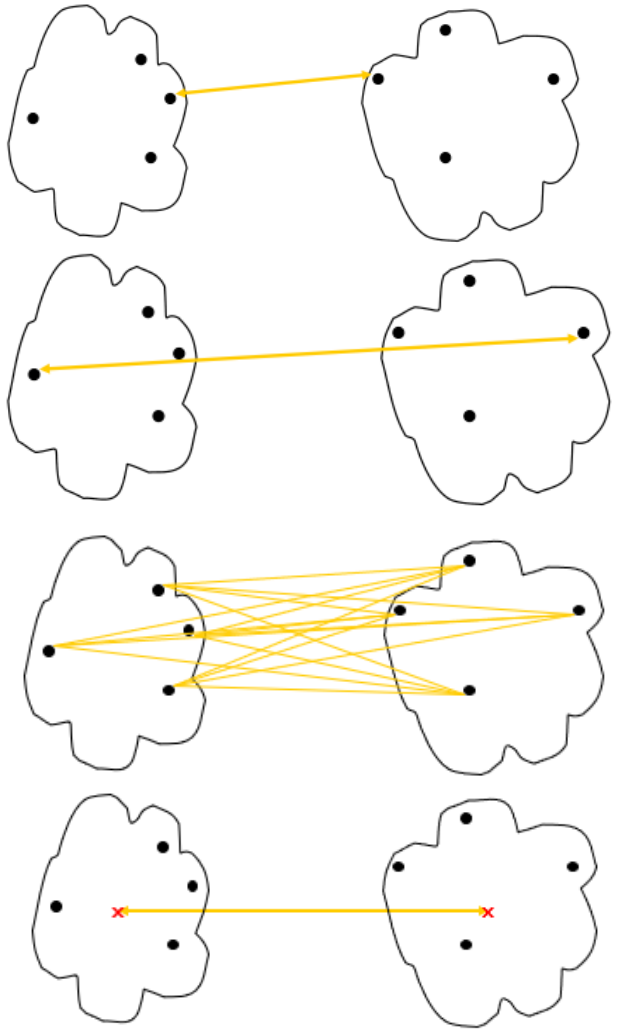

| | C1 | C2 U C5 | C3 | C4 |
|---|---|---|---|---|
| C1 | | ? | | |
| C2 U C5 | ? | ? | ? | ? |
| C3 | | ? | | |
| C4 | | ? | | |

Proximity Matrix



# Distance of clusters

- Single linkage (MIN): the distance of two clusters is the minimum pairwise distance of their data points
- Complete linkage (MAX): the distance of two clusters is the maximum pairwise distance of their data points
- Average linkage: taking the average of the distance values between pairs of cases
- Centroid method: the distance between two clusters is the distance between their centroids



# Example (single vs. complete linkage)

- Be careful! Here these matrices are similarity matrices not distance matrices! Similarity and dissimilarity (or distance) work oppositely!

| | I1 | I2 | I3 | I4 | I5 |
|---|---|---|---|---|---|
| I1 | 1.00 | 0.90 | 0.10 | 0.65 | 0.20 |
| I2 | 0.90 | 1.00 | 0.70 | 0.60 | 0.50 |
| I3 | 0.10 | 0.70 | 1.00 | 0.40 | 0.30 |
| I4 | 0.65 | 0.60 | 0.40 | 1.00 | 0.80 |
| I5 | 0.20 | 0.50 | 0.30 | 0.80 | 1.00 |

Single

| | I1 | I2 | I3 | I4 | I5 |
|---|---|---|---|---|---|
| I1 | 1.00 | 0.90 | 0.10 | 0.65 | 0.20 |
| I2 | 0.90 | 1.00 | 0.70 | 0.60 | 0.50 |
| I3 | 0.10 | 0.70 | 1.00 | 0.40 | 0.30 |
| I4 | 0.65 | 0.60 | 0.40 | 1.00 | 0.80 |
| I5 | 0.20 | 0.50 | 0.30 | 0.80 | 1.00 |

Complete

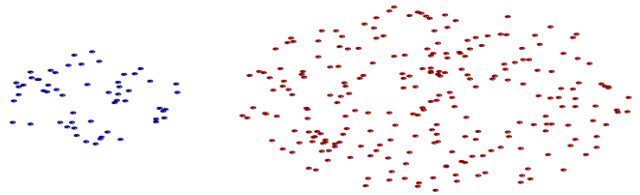
# Problem

Use the following similarity matrix to perform single and complete linkage hierarchical clustering by drawing two dendrograms!

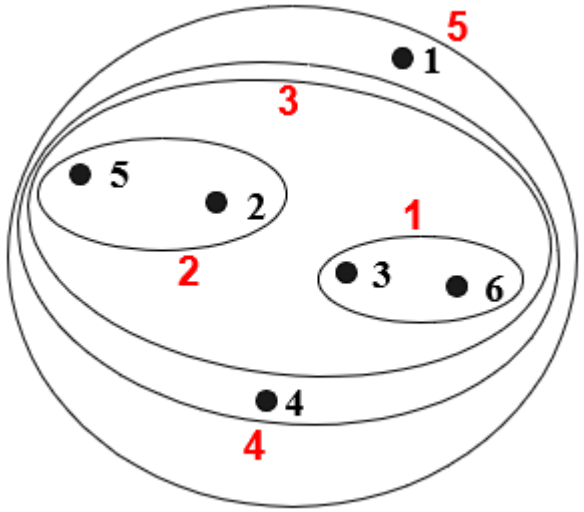
| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 1 | 0.15 | 0.6 | 0.15 | 0.95 |
| 2 | 0.15 | 1 | 0.5 | 0.2 | 0.2 |
| 3 | 0.6 | 0.5 | 1 | 0.05 | 0.7 |
| 4 | 0.15 | 0.2 | 0.05 | 1 | 0.85 |
| 5 | 0.95 | 0.2 | 0.7 | 0.85 | 1 |

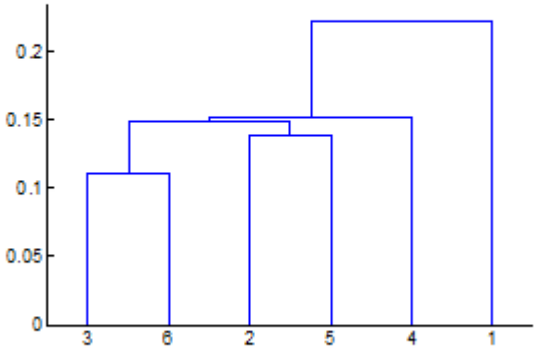
# Evaluating single linkage

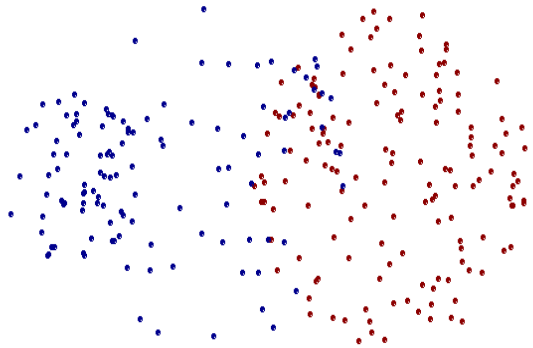
- Handle elliptical shapes well
- Sensitive for noise and outliers



Two clusters



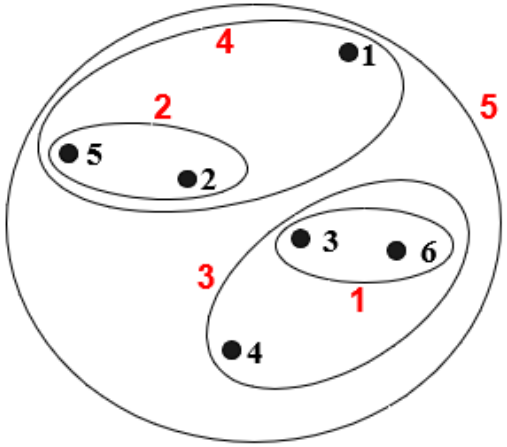



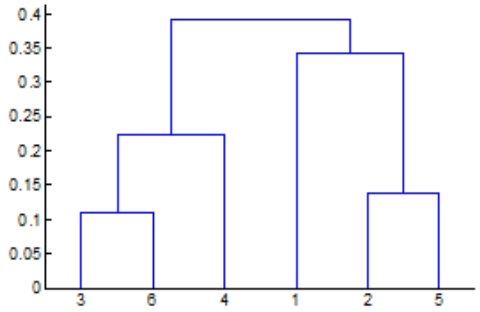
Two clusters

# Evaluating complete linkage

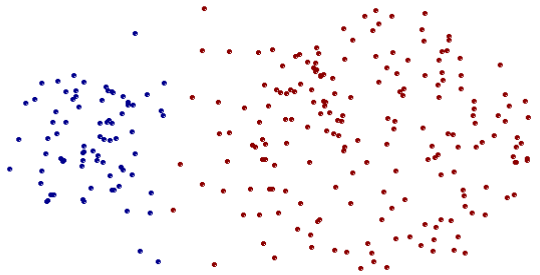
- Less sensitive for noise and outliers
- Tend to divide large clusters
- Tend to construct spherical clusters



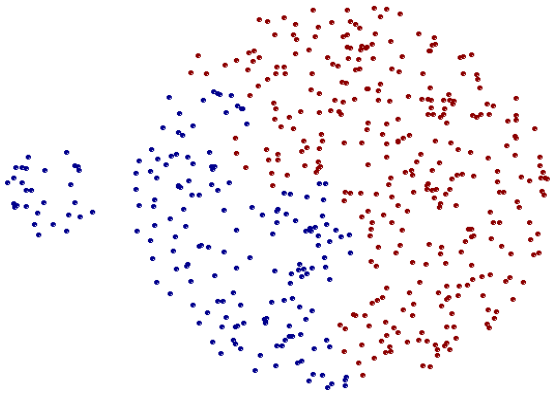

Nested Clusters




Dendrogram



Two clusters



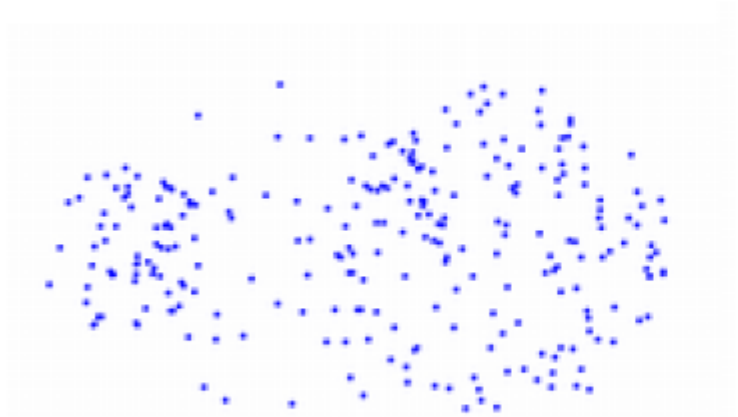
Two clusters

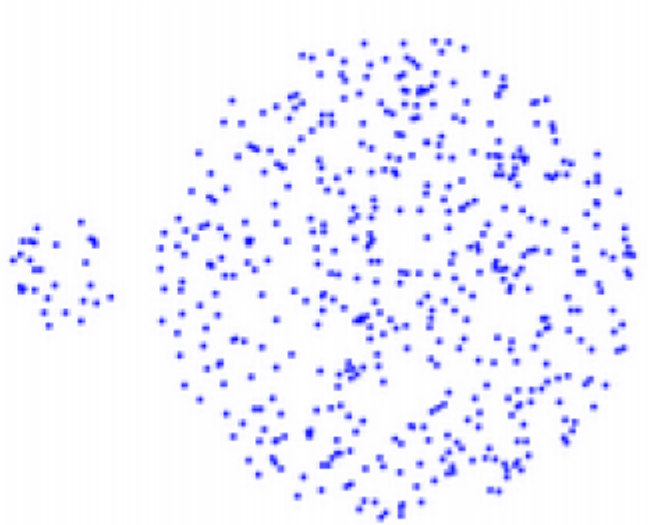
# Evaluating hierarchical clustering

- If two clusters are merged during the algorithm, they remain merged: sensitive for noise and outliers
- Complexity: for *n* datapoints $O(n^3)$, since we have at most *n* steps with complexity $O(n^2)$ (determining the new distances)
- It outputs a hierarchy or taxonomy that is more informative than the unstructured set of flat clusters (returned by K-means)
- Easier to decide on the number of clusters by looking at the dendrogram
- Easy to implement

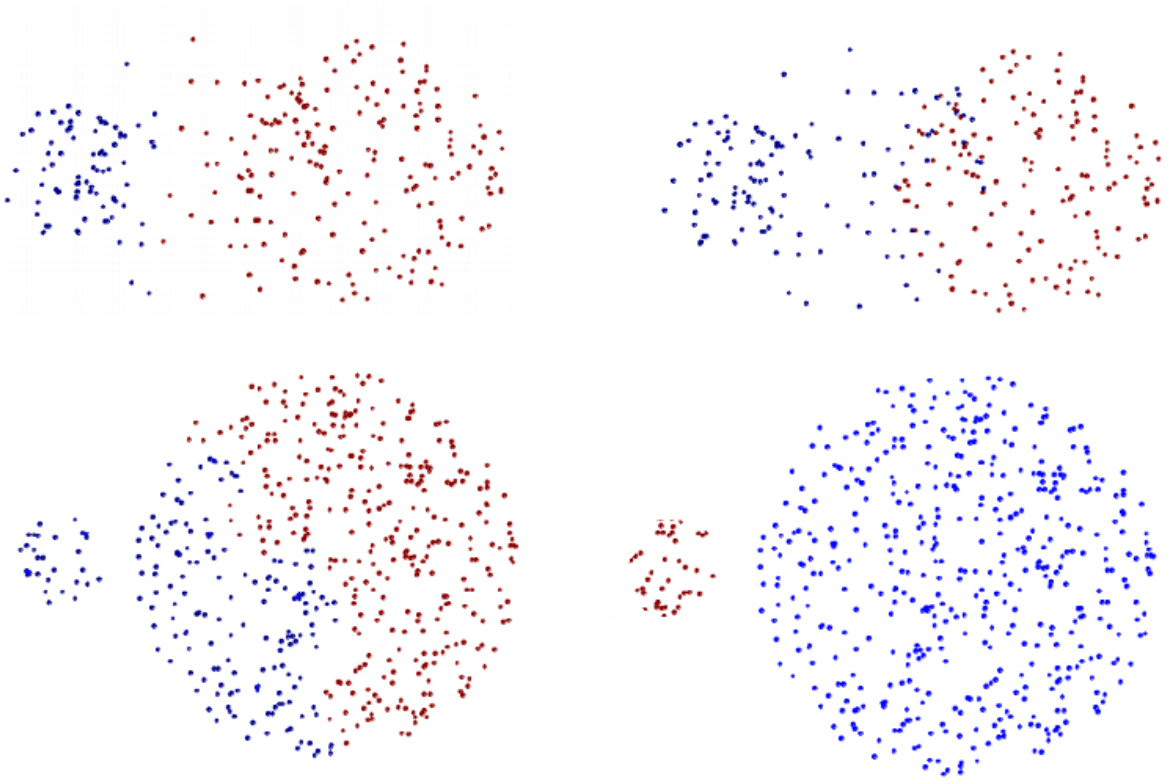
# Problem

Consider the two-dimensional data sets below. How would the following clustering algorithms split the data into two clusters: k-means, single-linkage and complete-linkage hierarchical clustering?



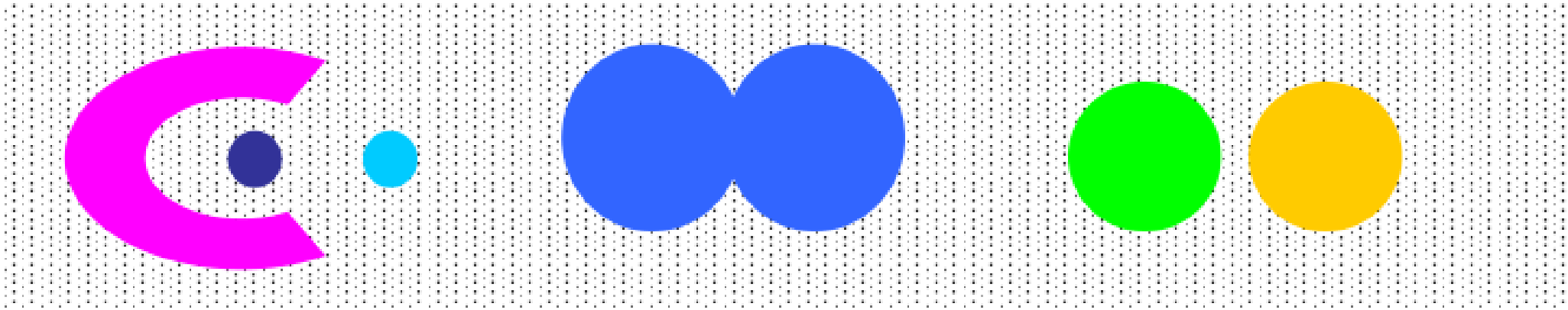


# Solution



# Density based clustering

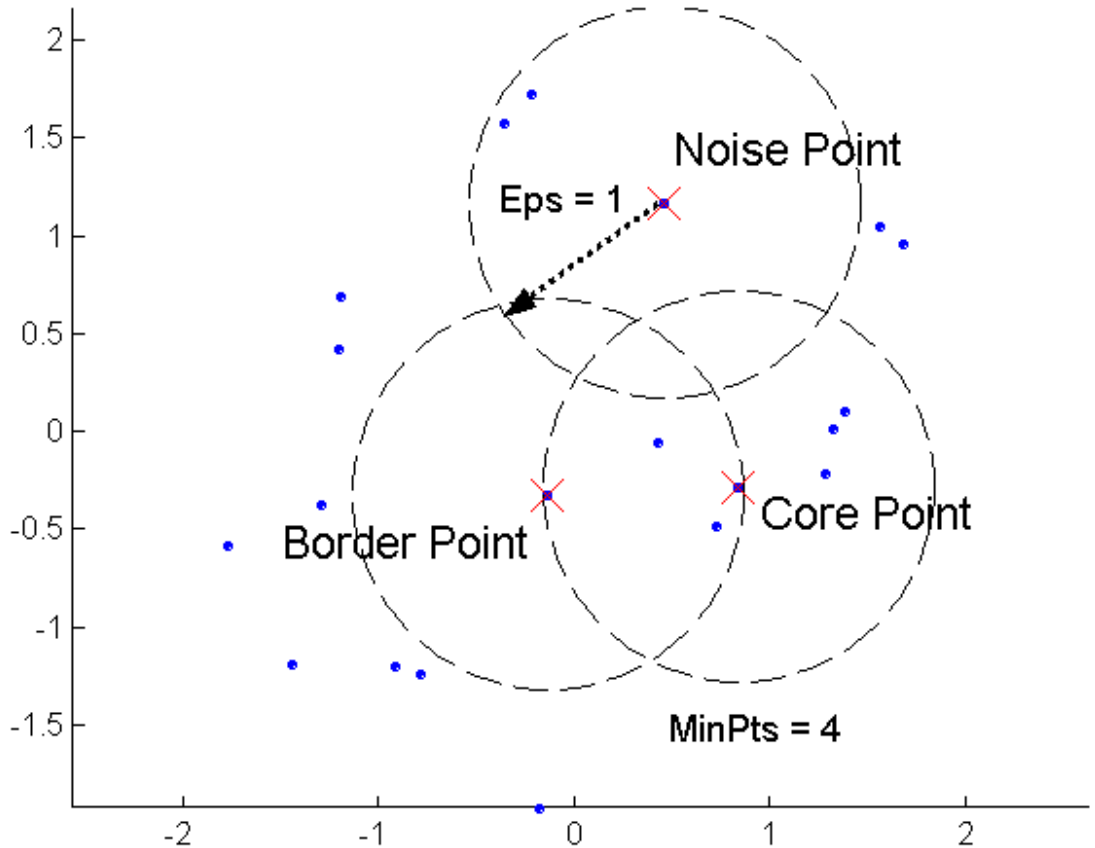
- Clusters are the areas with high density, between clusters the data is less dense
  - 6 density based clusters:



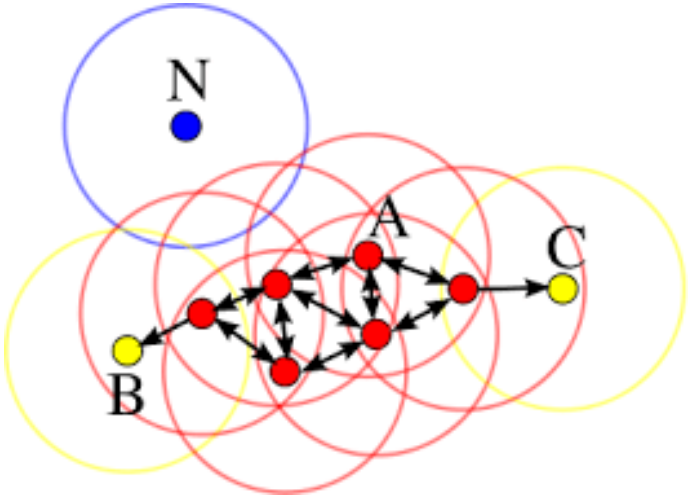
# DBSCAN algorithm

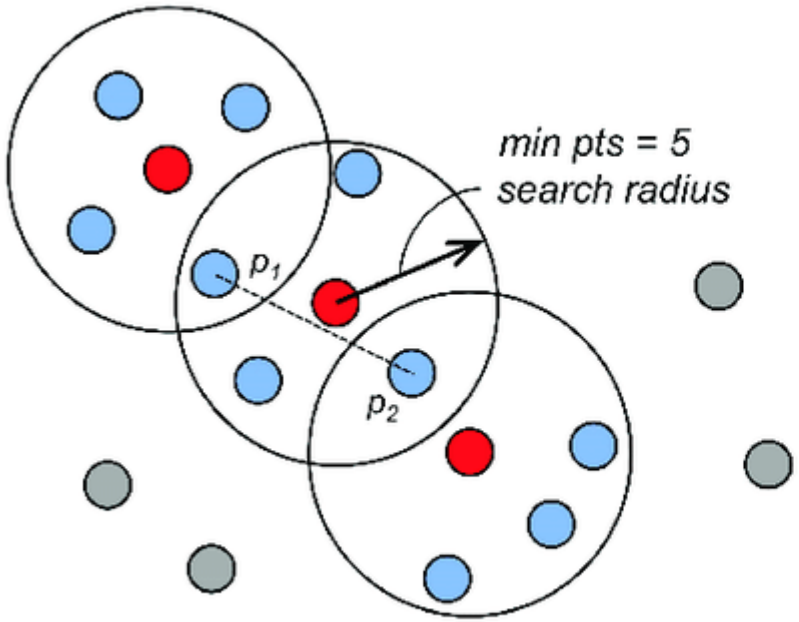
- DBSCAN: **D**ensity-**b**ased **s**patial **c**lustering of **a**pplications with **n**oise
  - Density: number of data points within a certain radius (*Eps*)
  - **Core points**: data points that have at least *MinPts* points within distance *Eps* of them (in their *Eps*-neighborhood)
    - The core points form the interior of the clusters
  - **Border points**: data points that have fewer than *MinPts* points in their *Eps*-neighborhood, but they themselves are in the *Eps*-neighborhood of a core point
    - The border points form the borders of the clusters
  - **Noise points (outliers)**: data points that are neither core points nor border points
    - Noise points are not clustered

# Core points, border points, noise points

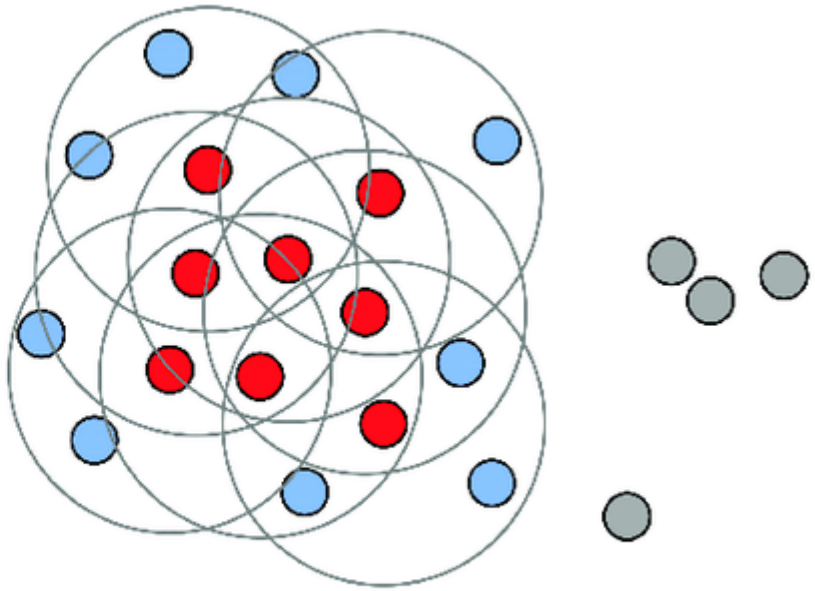


# DBSCAN algorithm



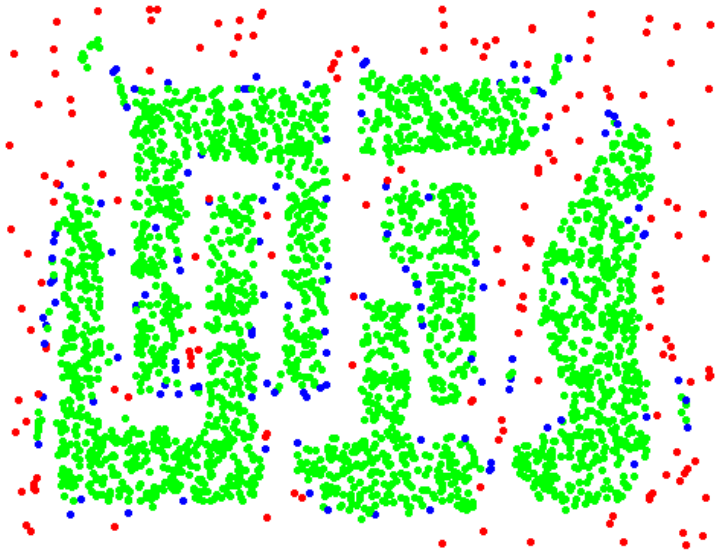

Cluster 1



Cluster 2

# DBSCAN – types of data points

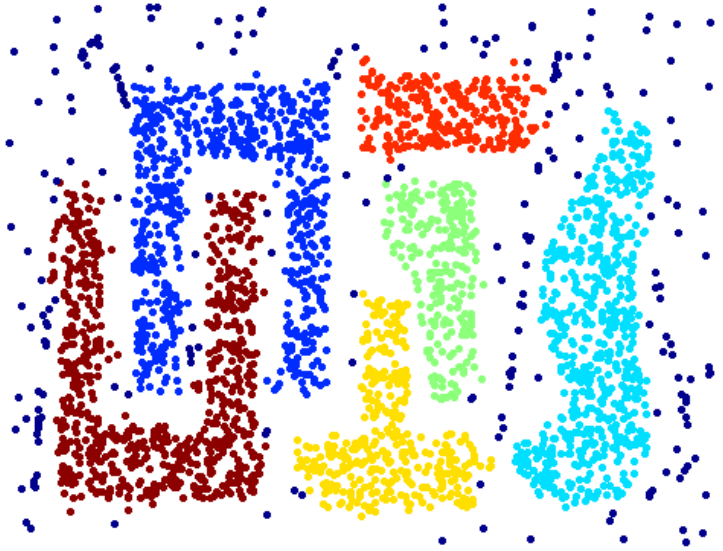
The data



Types of points: core, border and noise

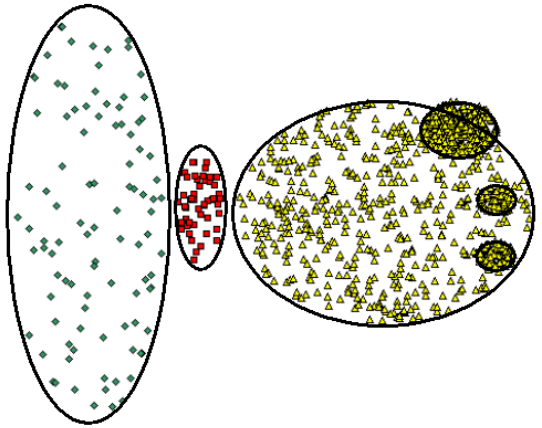
Eps = 10, MinPts = 4

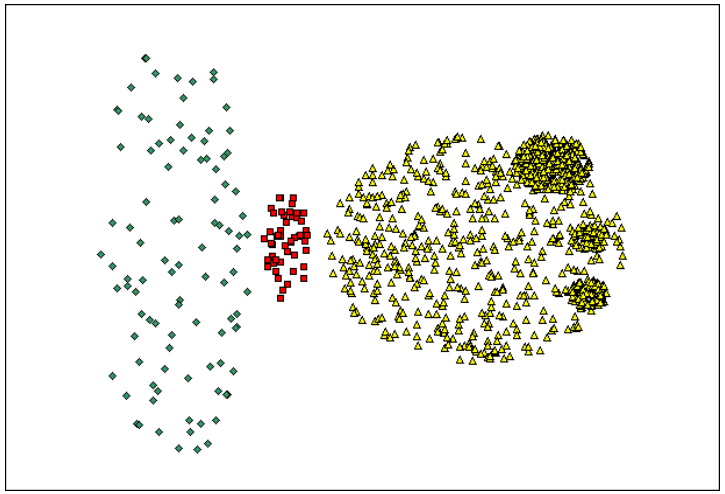
# DBSCAN - example



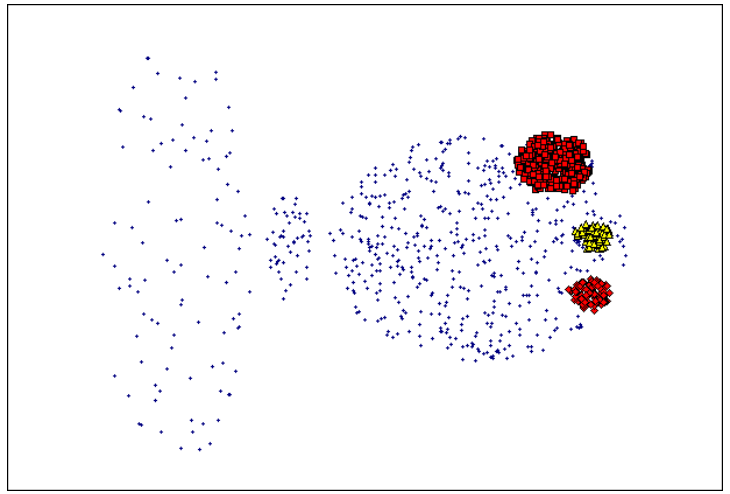
Clusters

# DBSCAN - example





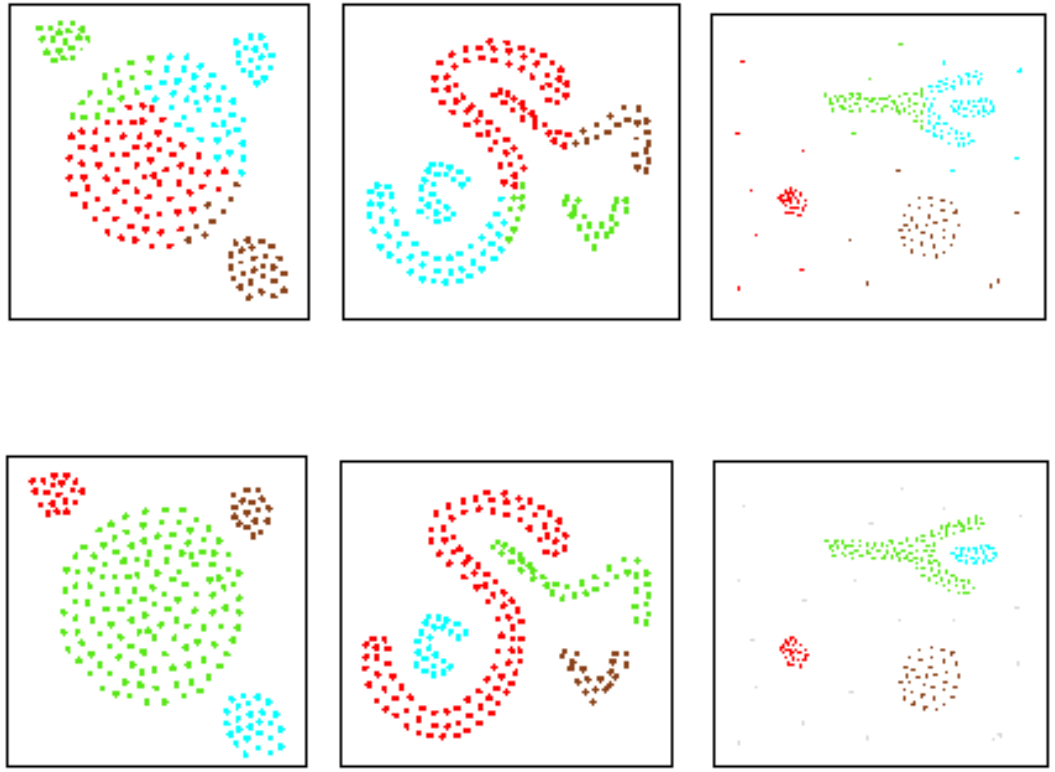
(MinPts=4, Eps=9.75)



(MinPts=4, Eps=9.62)

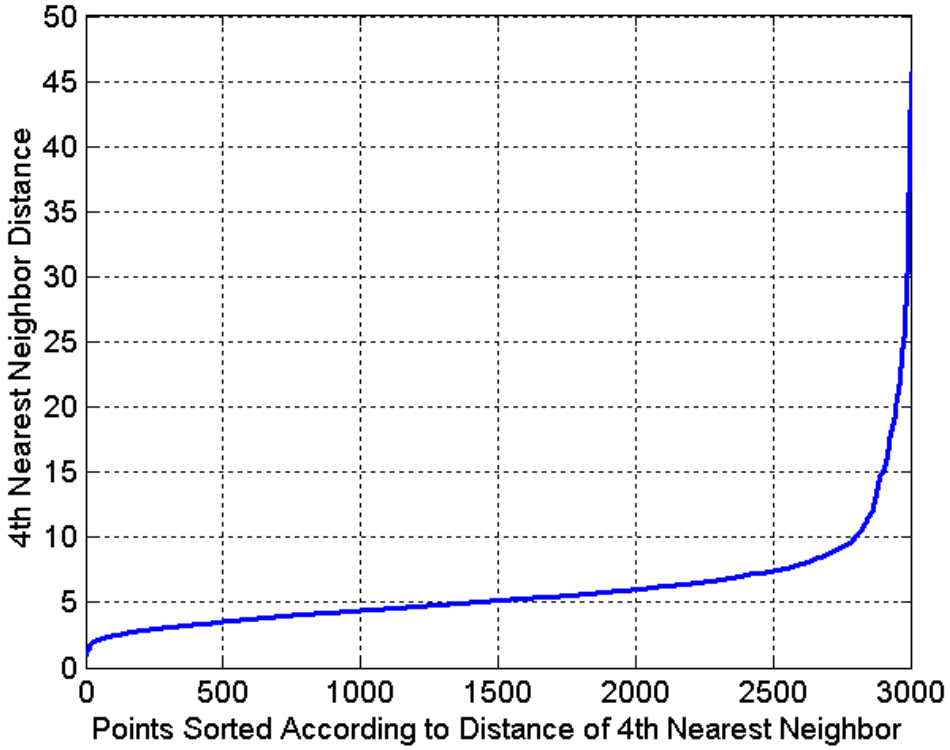
# Evaluation of DBSCAN

- Insensitive to noise
- Not all the data points are clustered
- It can automatically handle outliers (by ignoring them)
- Treats clusters with differing size and shapes well
- Can’t handle clusters with differing density
- Sensitive for the choice of hyperparameters (*MinPts*, *Eps*)



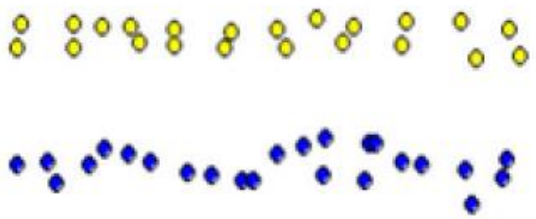
# Determining the hyperparameters

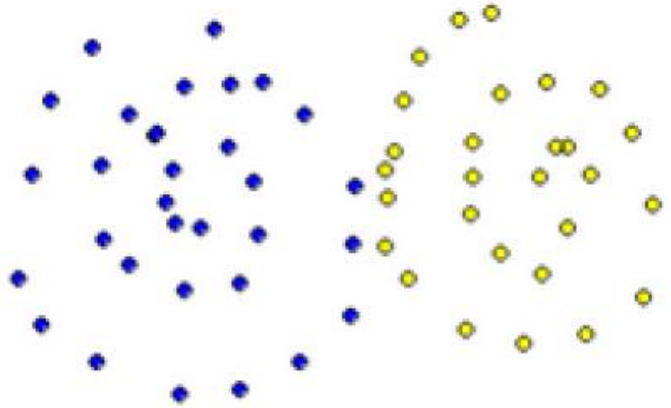
- Plot the distance of the *k*th nearest neighbor for the data points
  - Idea: in dense areas the *k*th nearest neighbors are almost constants but the *k*th nearest neighbor of a noise point is much further away
  - Can we observe an angle in the graph?
    - The corresponding distance value is a reasonable choice for *Eps*



# Problem

Which clustering algorithm would you use if the goal was to find the two natural clusters (marked by blue and yellow colors)? Consider the following algorithms: k-means, hierarchical clustering (both single and complete linkage) and DBSCAN.





# Acknowledgement


- András Benczúr, Róbert Pálovics, SZTAKI-AIT, DM1-2
- Krisztián Buza, MTA-BME, VISZJV68
- Bálint Daróczy, SZTAKI-BME, VISZAMA01
- Judit Csima, BME, VISZM185
- Gábor Horváth, Péter Antal, BME, VIMMD294, VIMIA313
- Lukács András, ELTE, MM1C1AB6E
- Tim Kraska, Brown University, CS195
- Dan Potter, Carsten Binnig, Eli Upfal, Brown University, CS1951A
- Erik Sudderth, Brown University, CS142
- Joe Blitzstein, Hanspeter Pfister, Verena Kaynig-Fittkau, Harvard University, CS109
- Rajan Patel, Stanford University, STAT202
- Andrew Ng, John Duchi, Stanford University, CS229