

# Clustering

---

Machine Learning I  
Felipe Alonso-Atienza



Universidad  
Rey Juan Carlos

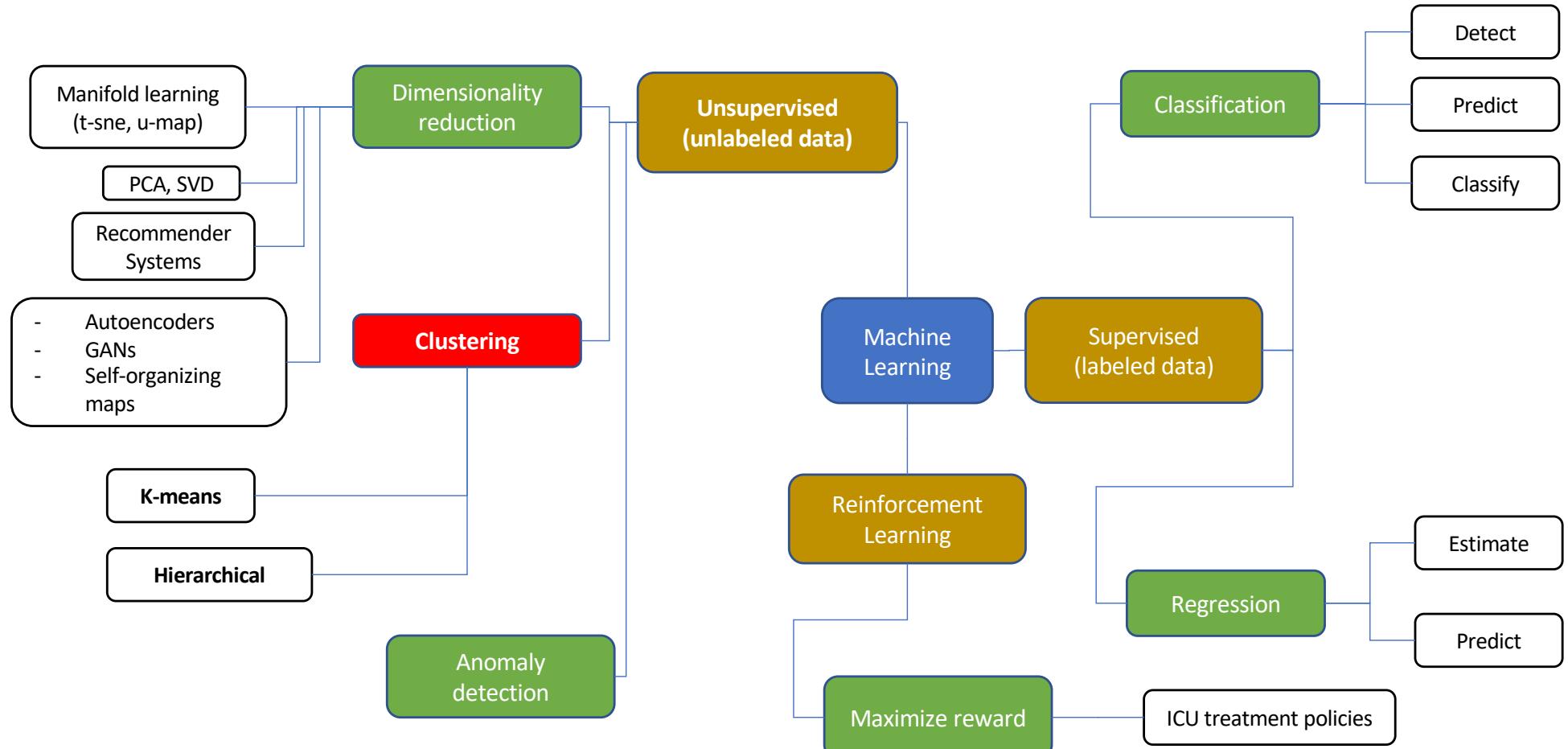
# About me

---

- Senior Expert Data Scientist @BBVA
  - AM-GW: financial engineering
  - Dynamic Pricing
  - Document Automation
    - PDF parsing and NLP
  - Churn and sales prediction
  - (Robust) customer segmentation
  - Recommender Systems
- (Part-time) Associate professor at @ETSIT-URJC
  - Researcher: statistical signal processing and machine learning with application to cardiac electrophysiology
  - Lecturer (Biomedical Engineering Degree):
    - Inference
    - Artificial Intelligence and Learning



# Machine learning taxonomy



# Unsupervised learning

---

- Data does not contain (or we do not use) any label information
- The learning algorithms just shown the **input data** and asked to **extract knowledge/patterns** from this data.
- A major challenge is **evaluating** whether the algorithm learned something useful
- Thus, unsupervised algorithms are used often in an **exploratory setting**, when a data scientist wants to understand the data better
- Another common application is as a preprocessing step for supervised algorithms
  - **Learning a new representation of the data** can sometimes improve the accuracy of supervised algorithms, or can lead to reduced memory and time consumption



# Other taxonomies

---

- Parametric vs non-parametric
  - Logistic regression vs KNN classifier
  - Scalability vs flexibility
- Discriminative vs generative
  - Logistic regression vs Naïve Bayes
  - Discriminant border vs pdf fitting
  - $p(y|x)$  vs  $p(x,y)$

# Outline

---

1. What is clustering?
2. K-means
3. Hierarchical
4. DBSCAN
5. Mixture of gaussians
6. Spectral Clustering
7. Time series clustering

# What's clustering

---

- Grouping unlabeled examples
  - It is the task of partitioning the dataset into groups, called **clusters**
  - Points within a single cluster are very similar and points in different clusters are different
  - Clustering algorithms assign (or predict) a number to each data point, indicating which cluster a particular point belongs to

# Uses of Clustering

---

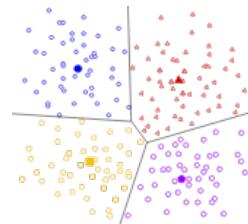
- Market segmentation
- Group items by some features: docs, pictures, music, search results
- Anomaly detection
- Infer missing data
- Data compression
- Privacy: userID vs clusterID

# Types of Clustering

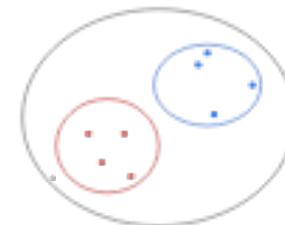
---

- For an exhaustive list, see this [paper](#).

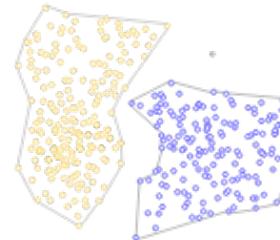
1. Centroid-based: k-means



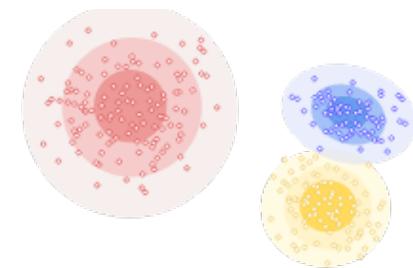
2. Connectivity-based: hierarchical



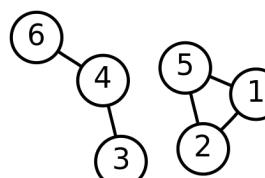
3. Density-based: DBSCAN



4. Distribution-based: mixture of gaussians



5. Graph-based: spectral clustering



# Outline

---

1. What is clustering?
2. K-means
3. Hierarchical
4. DBSCAN
5. Mixture of gaussians
6. Spectral Clustering
7. Time series clustering

# K-means

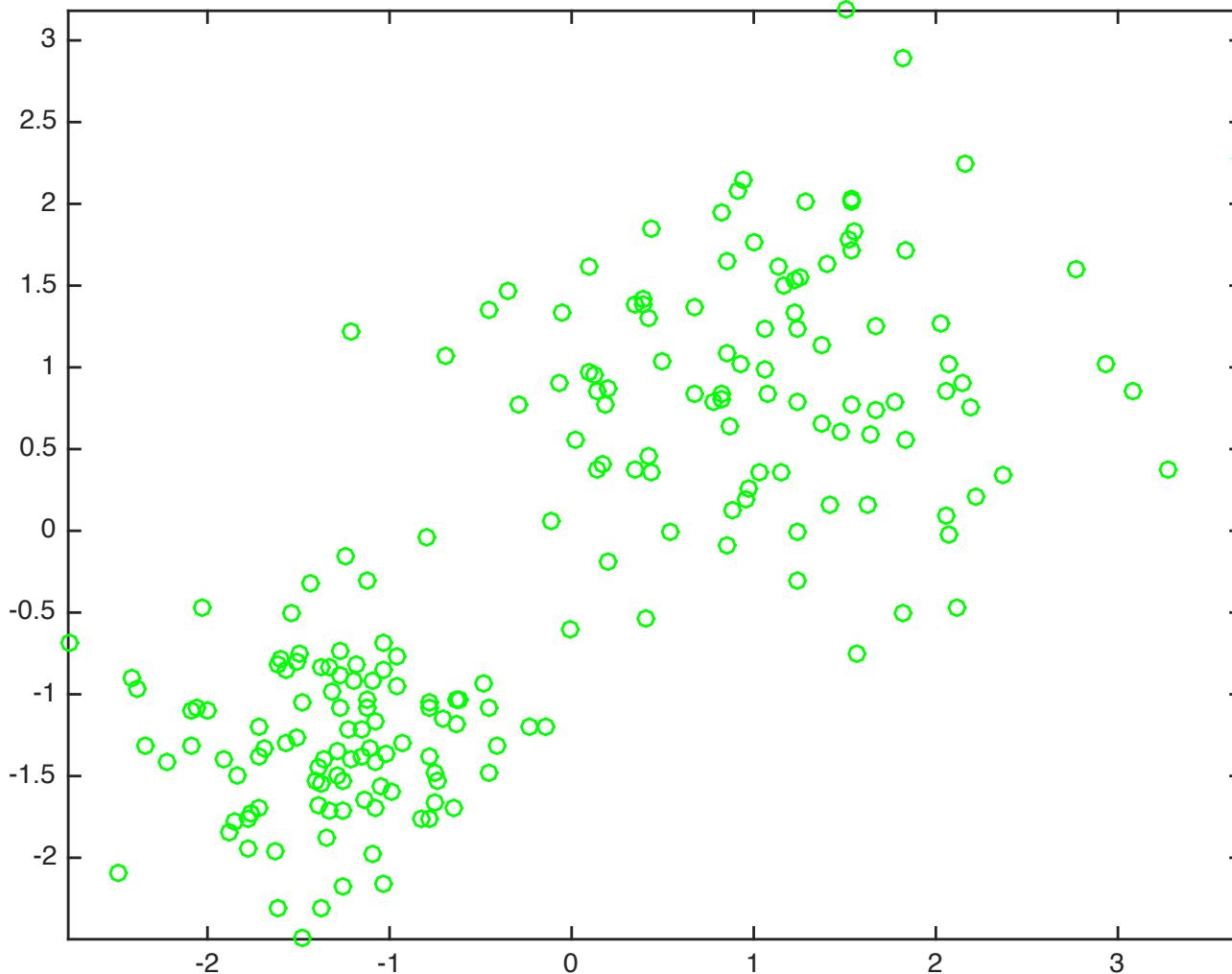
---

- One of the simplest and most used clustering algorithms
- The algorithm alternates between two steps:
  1. Assigning each data point to the closest cluster center, and then
  2. Setting each cluster center as the mean of the data points that are assigned to it.
- The algorithm is finished when the assignment of instances to clusters no longer changes



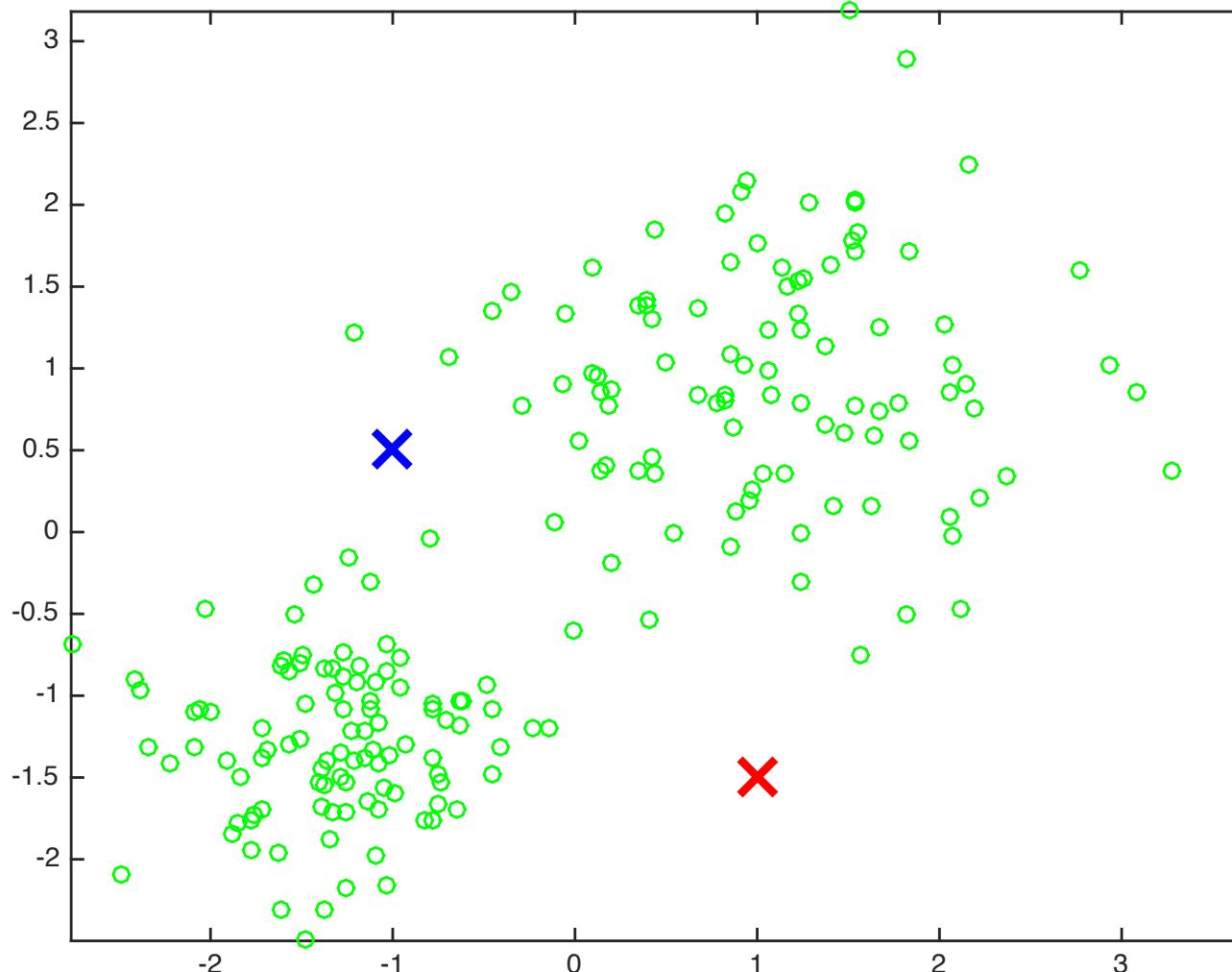
# K-means

---



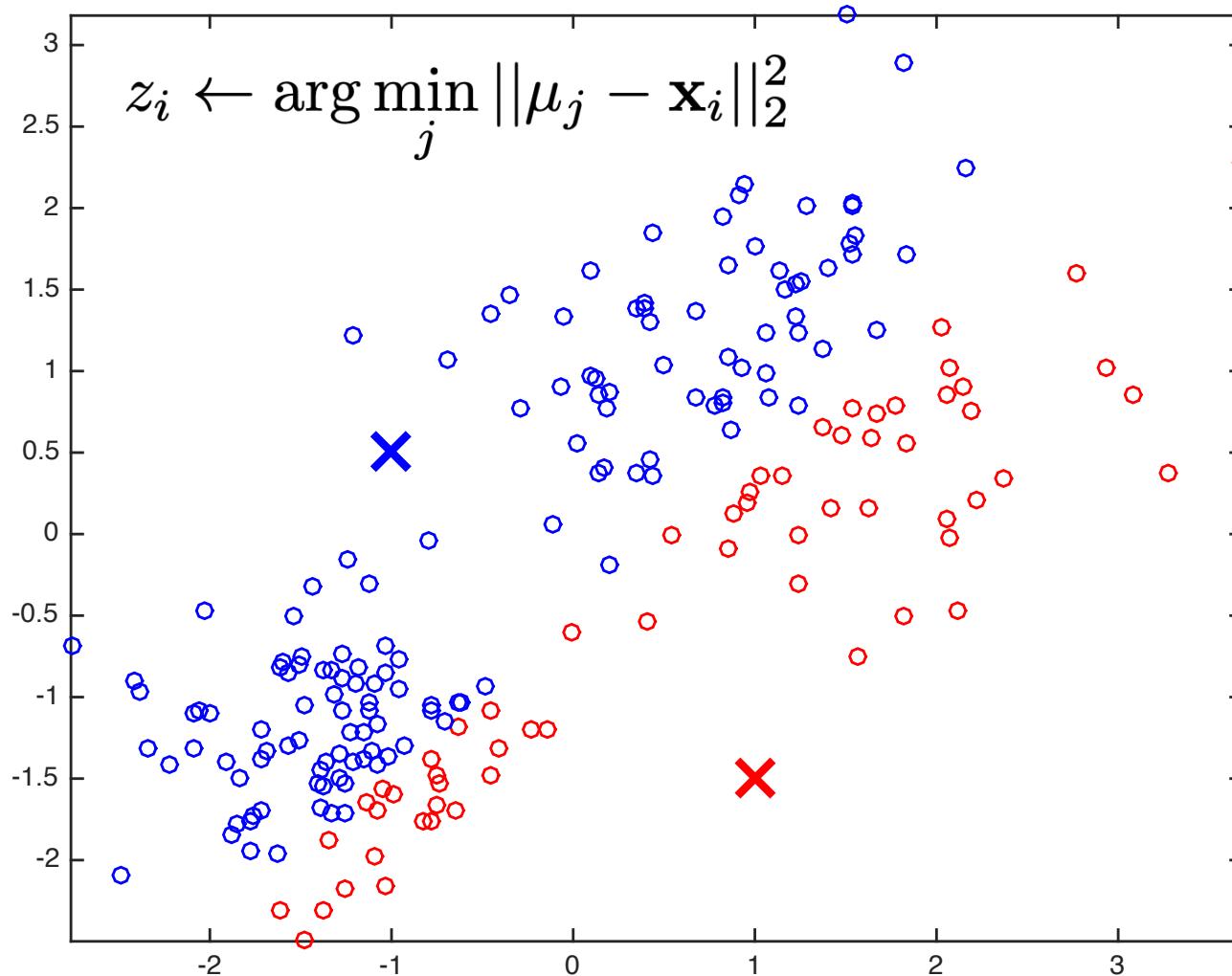
# K-means: initialize cluster centers $\mu_k$

---



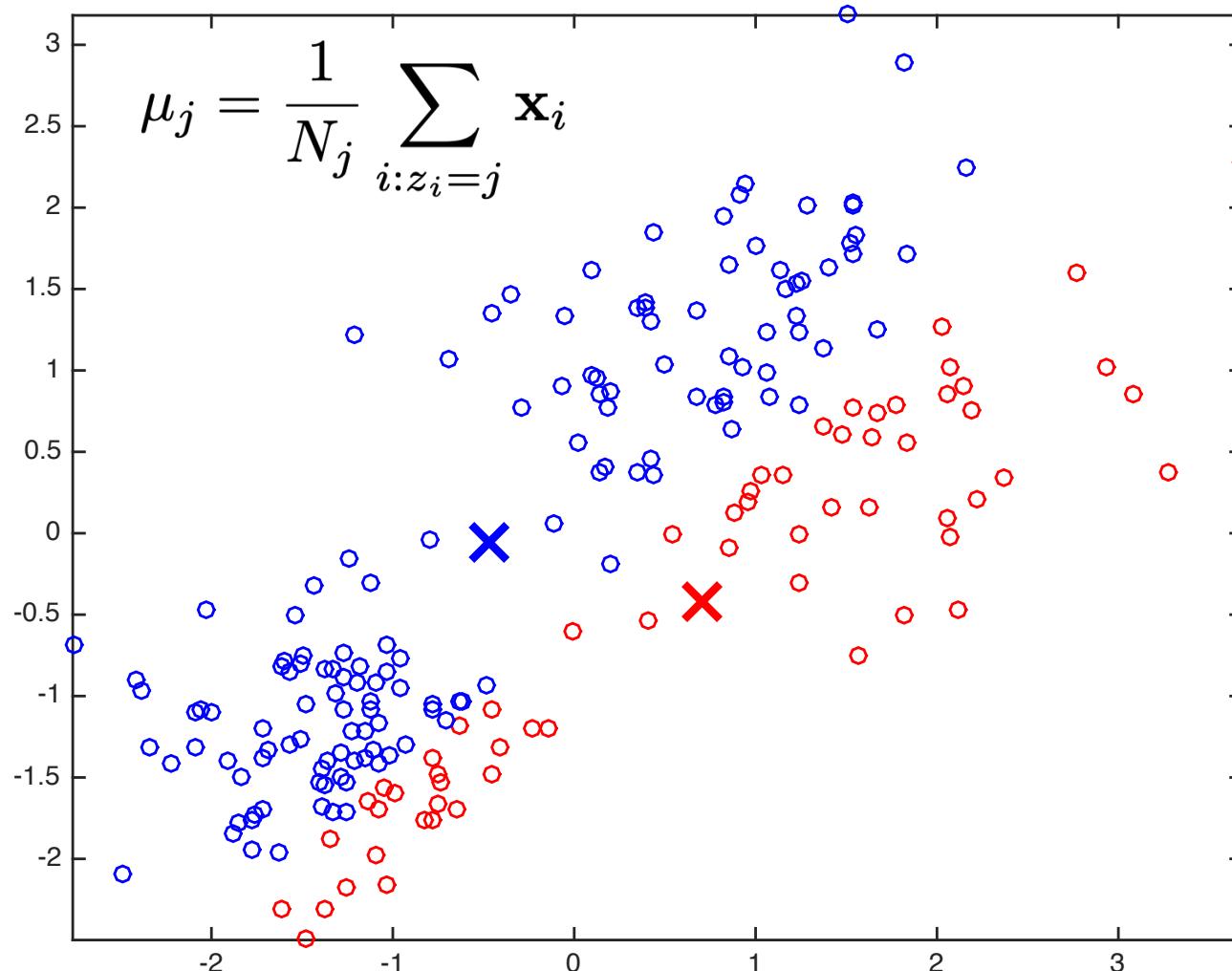
# K-means: assign observations to cluster center

---



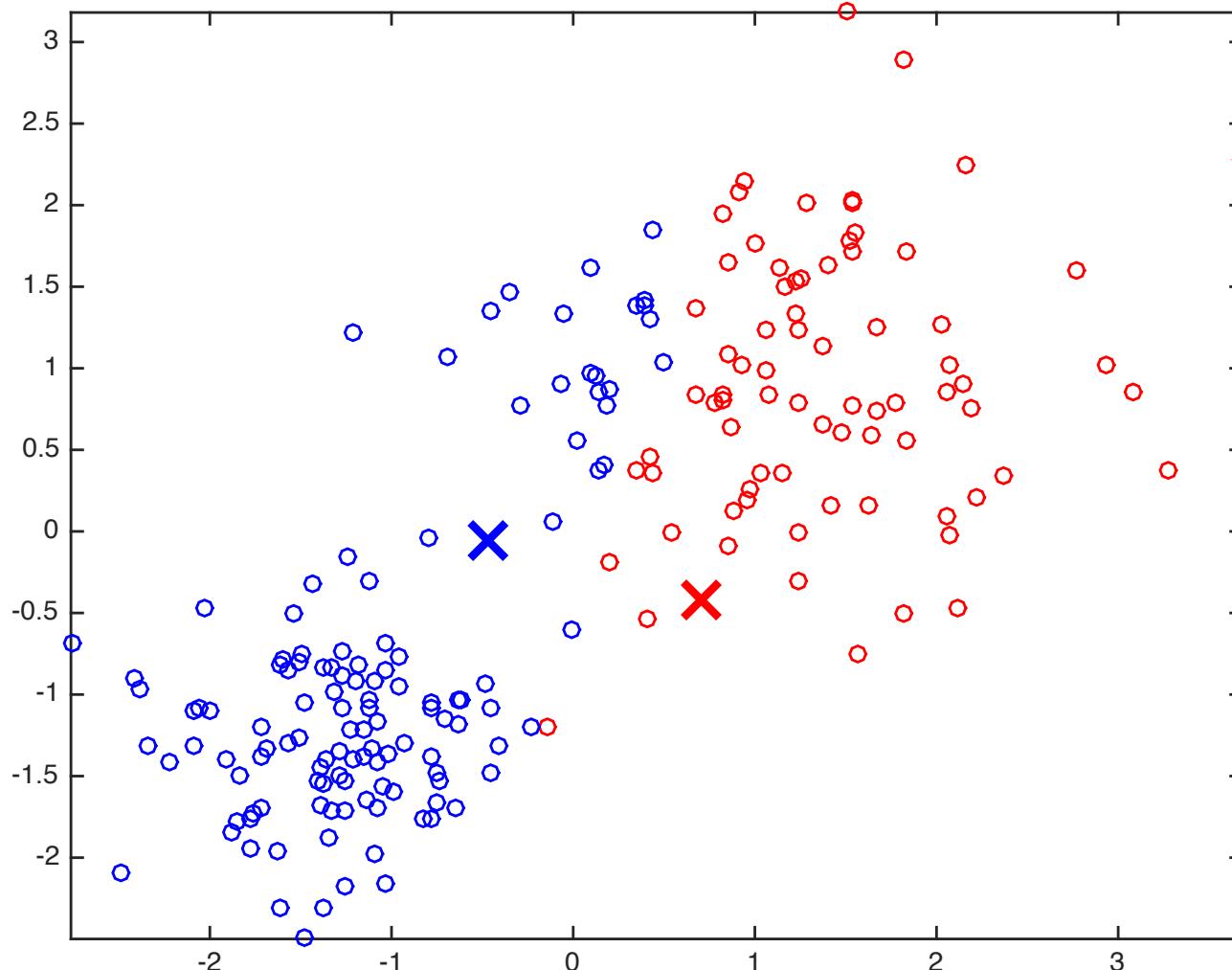
# K-means: recompute centroids

---



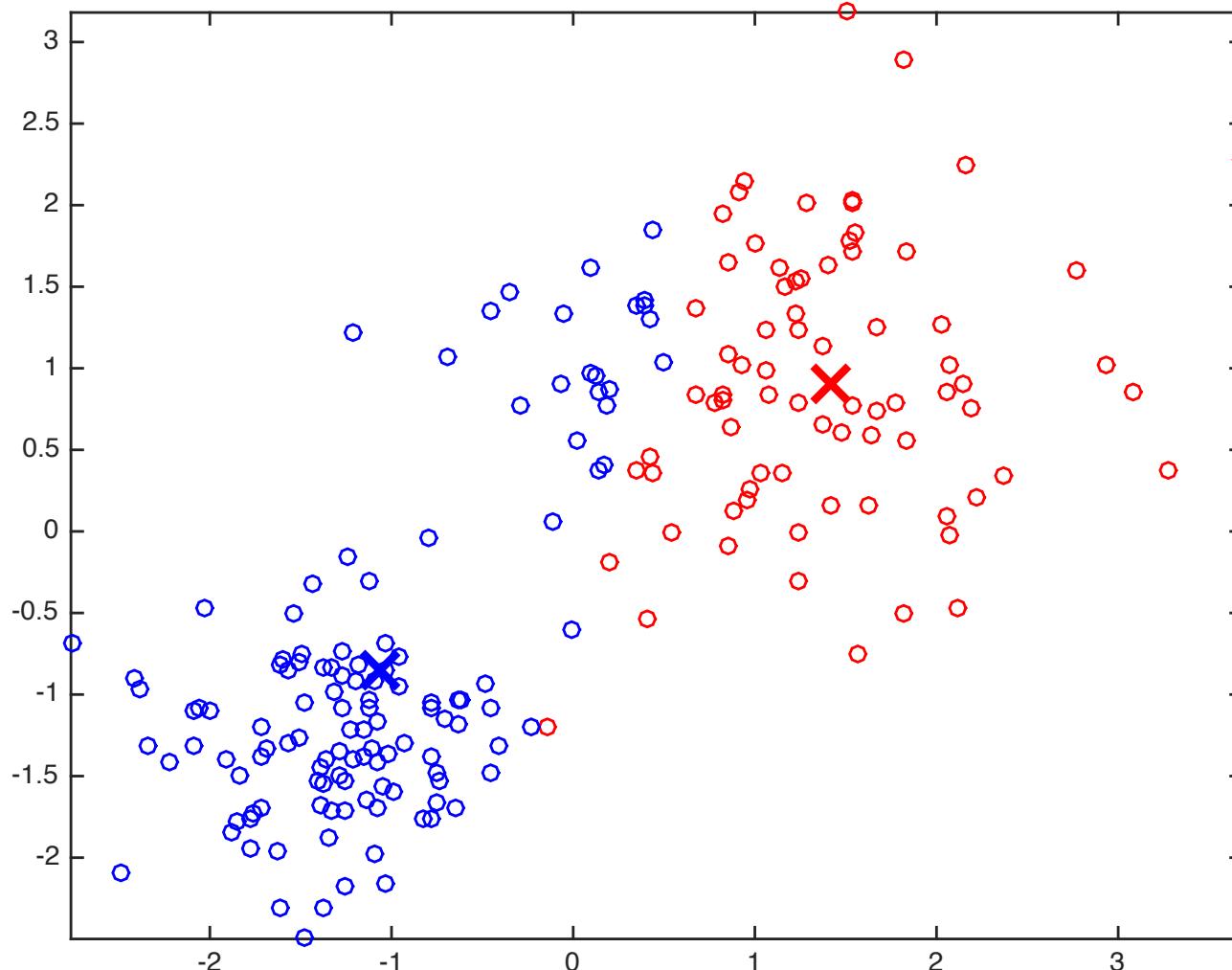
# K-means: assign observations to cluster center

---



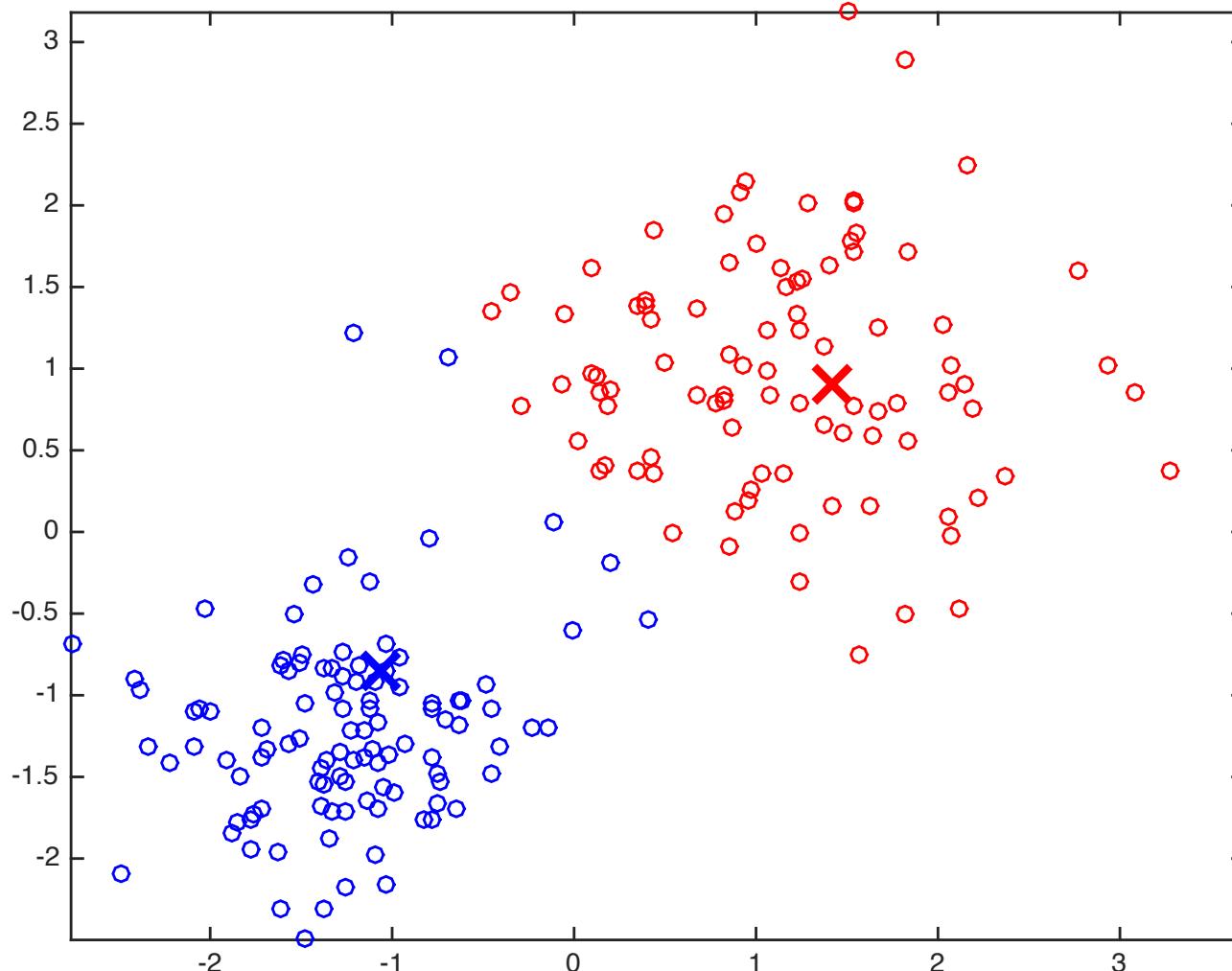
# K-means: recompute centroids

---



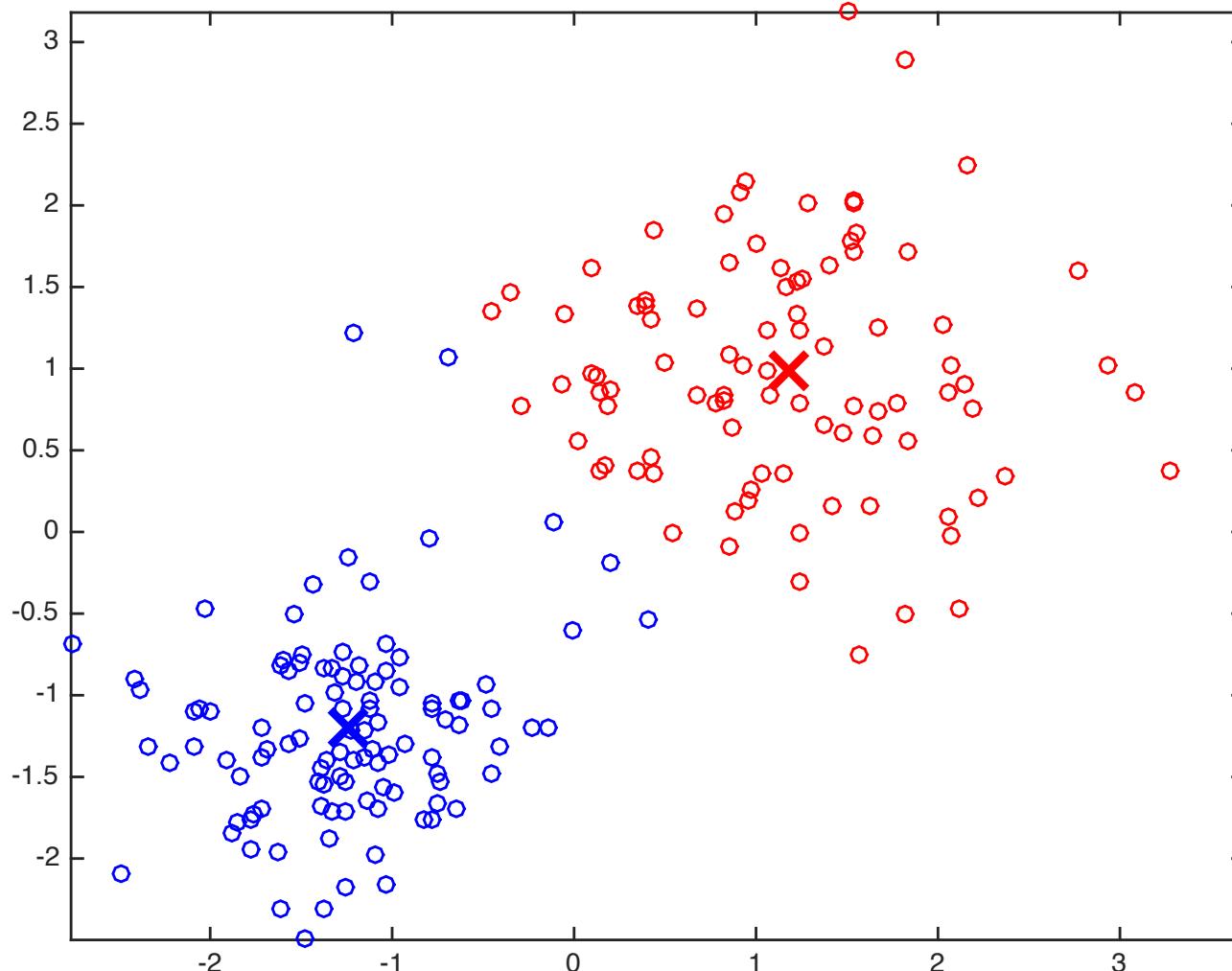
# K-means: assign observations to cluster center

---



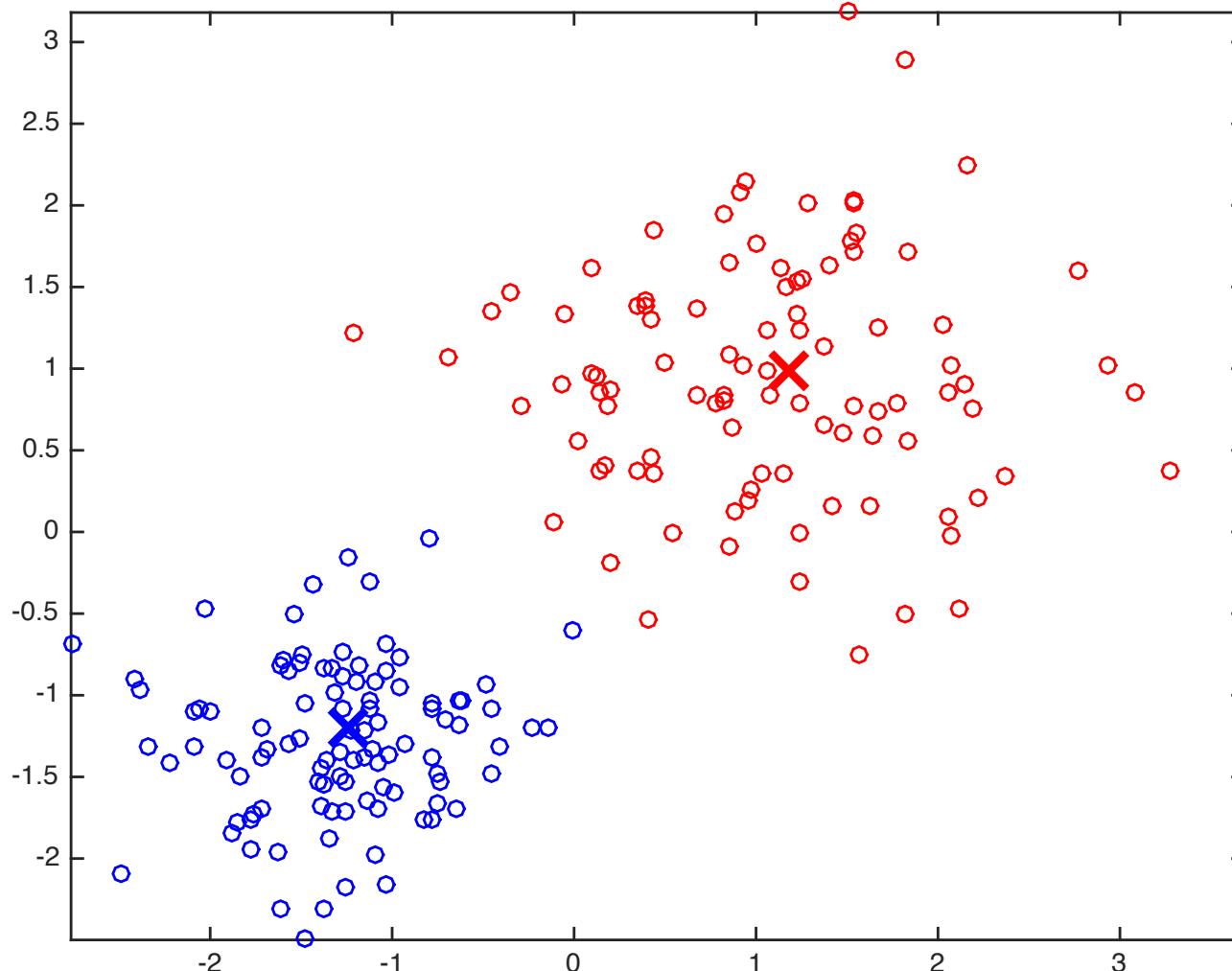
# K-means: recompute centroids

---



# K-means: assign observations to cluster center

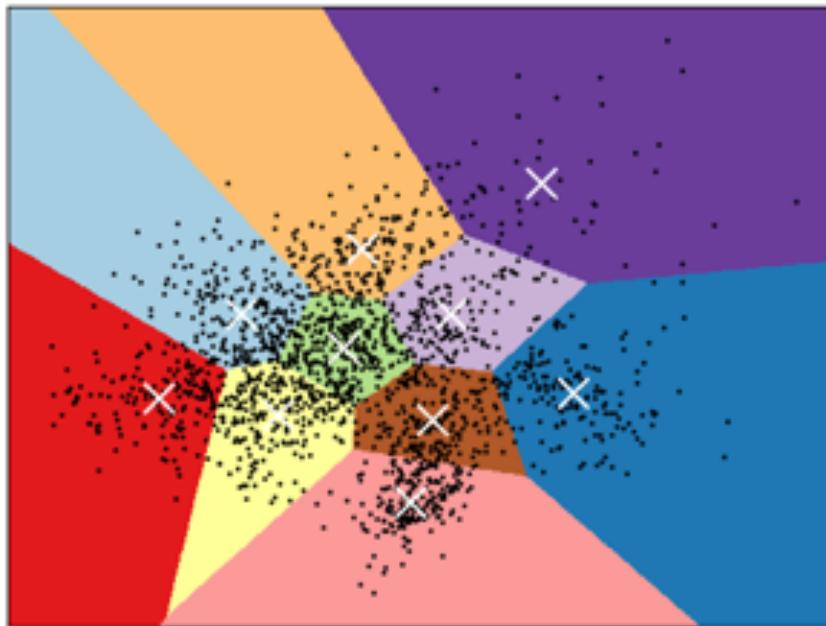
---



# K-means in action

---

K-means clustering on the digits dataset (PCA-reduced data)  
Centroids are marked with white cross



<https://github.com/FelipeURJC/clustering>

# K-means recap

---

- OKs
  - Simple and scalable (large data sets)
  - Adapts to new examples (predict)
- KOs:
  - Choosing k manually.
  - Highly dependent on the initialization of the centroids
    - Computation is often done several times, with different initializations
    - K-means++
  - Scaling with number of dimensions: as the number of dimensions increases, a distance-based similarity measure converges to a constant value between any given examples.
  - Centroids can be dragged by outliers



# Outline

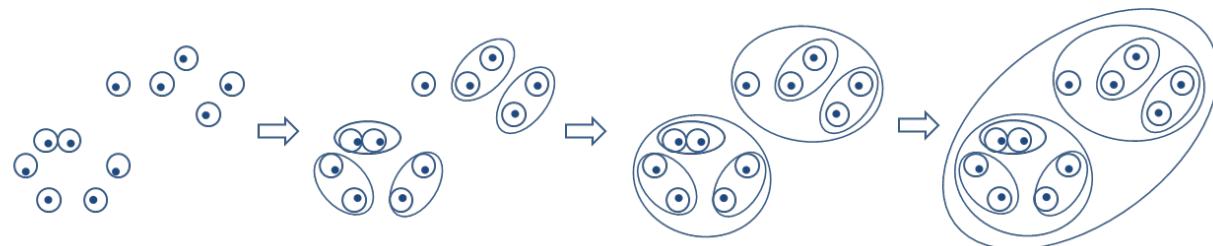
---

1. What is clustering?
2. K-means
- 3. Hierarchical**
4. DBSCAN
5. Mixture of gaussians
6. Spectral Clustering
7. Time series clustering

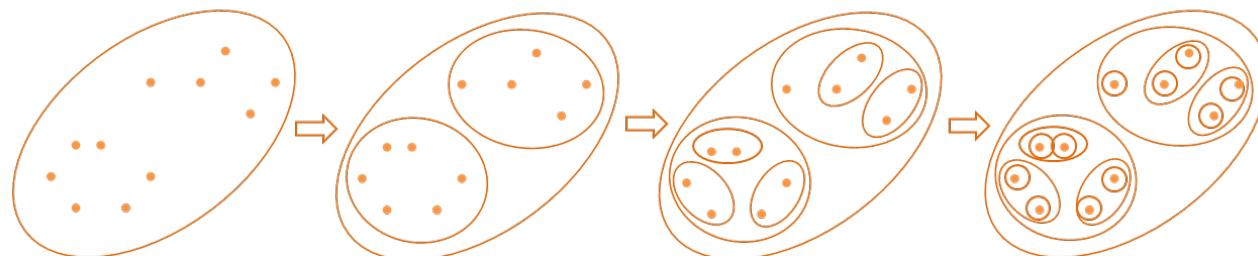
# Hierarchical clustering

- General family of clustering algorithms that build nested clusters by merging or splitting them successively:

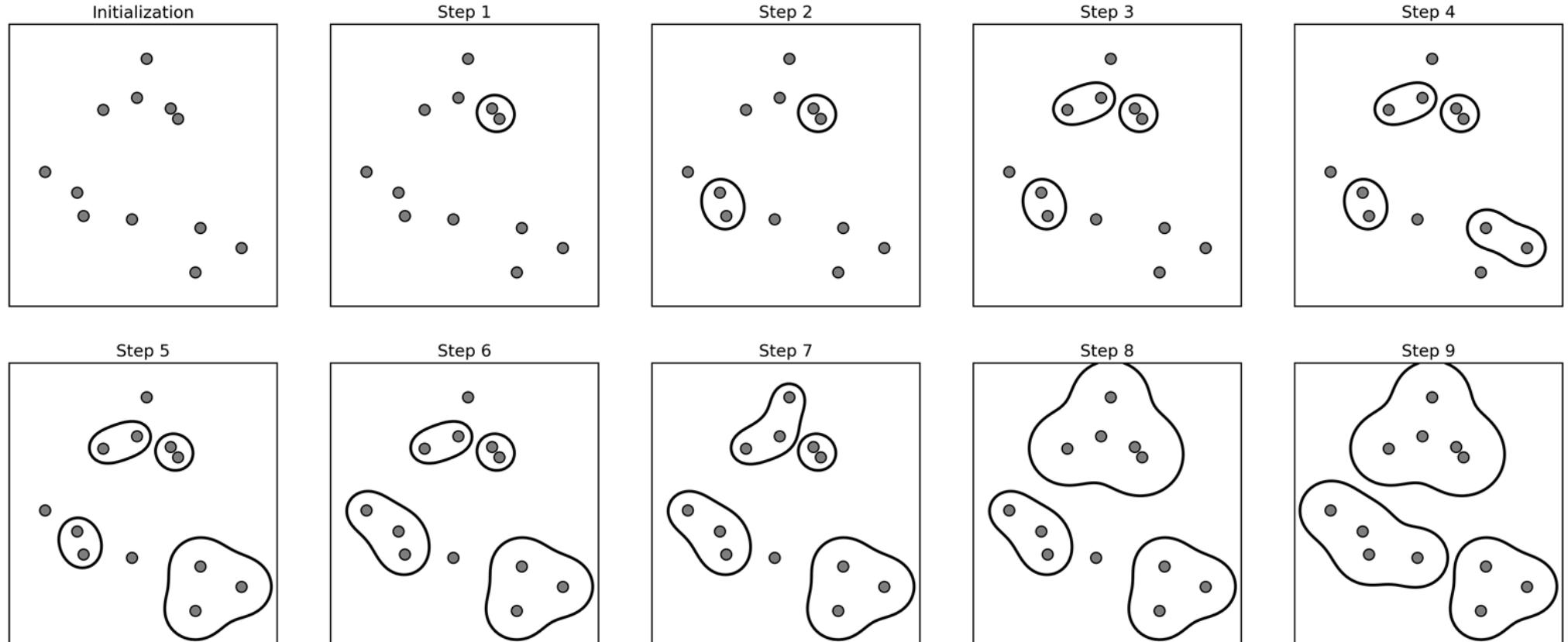
- **Agglomerative** (bottom-up): each observation starts in its own cluster, and pairs of most similar clusters are merged as one moves up the hierarchy.



- **Divisive** (top-down) all observations start in one cluster, and splits are performed recursively as one moves down the hierarchy (not in sklearn)



# Agglomerative clustering



[https://github.com/amueller/introduction\\_to\\_ml\\_with\\_python/blob/master/03-unsupervised-learning.ipynb](https://github.com/amueller/introduction_to_ml_with_python/blob/master/03-unsupervised-learning.ipynb)

# Agglomerative clustering

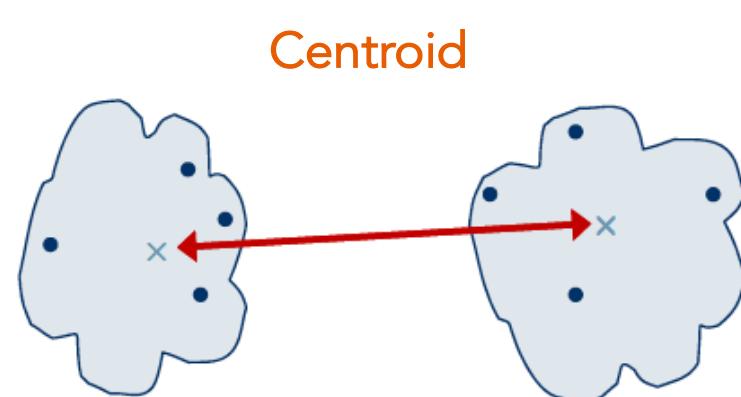
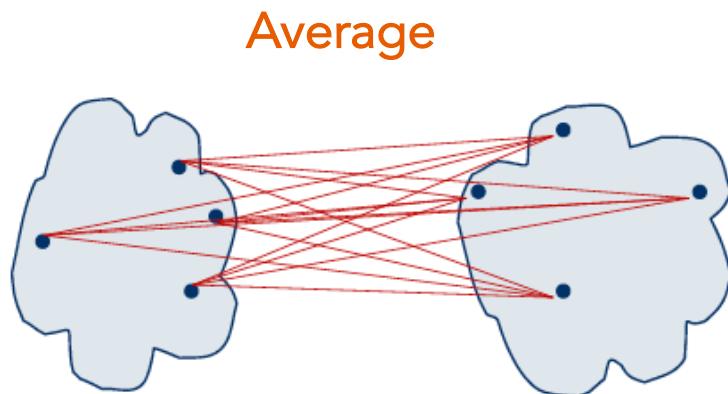
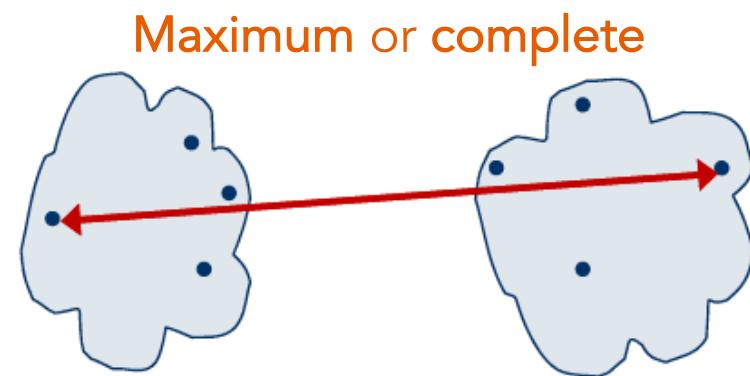
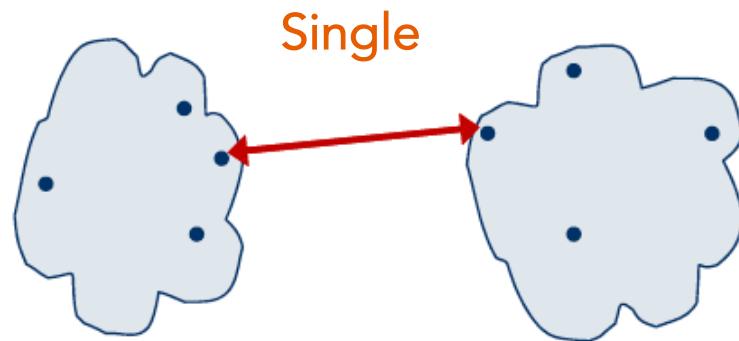
---

- There are several **linkage criteria** that specify how exactly the “most similar cluster” is measured.
- This measure is always defined between two existing clusters. :
  - **Ward** linkage minimizes the sum of squared differences within all clusters.
  - **Maximum** or **complete** linkage minimizes the maximum distance between observations of pairs of clusters.
  - **Average** linkage minimizes the average of the distances between all observations of pairs of clusters.
  - **Single** linkage minimizes the distance between the closest observations of pairs of clusters.



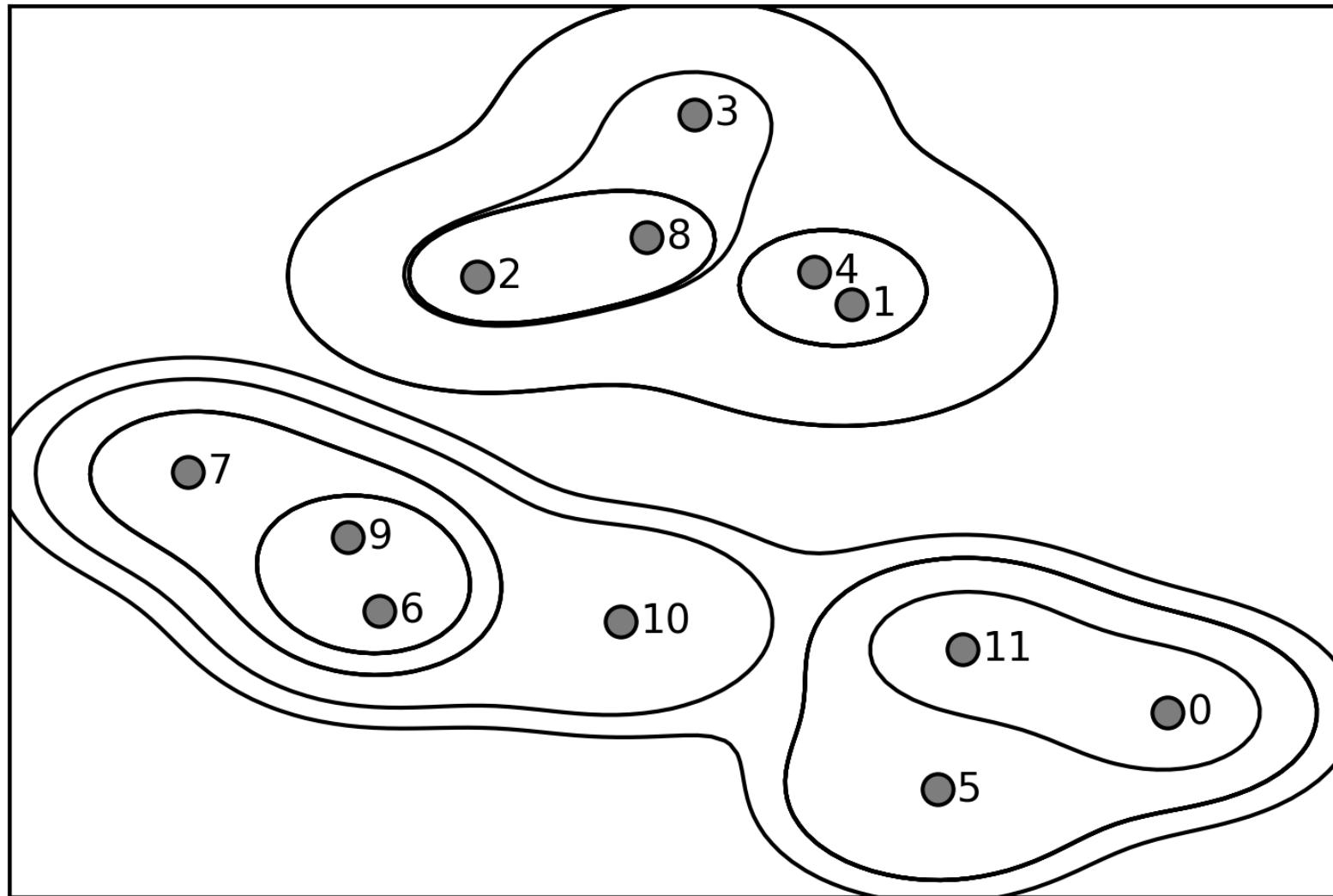
# Linkage criteria

- There are several ***linkage criteria*** that specify how exactly the “most similar cluster” is measured.



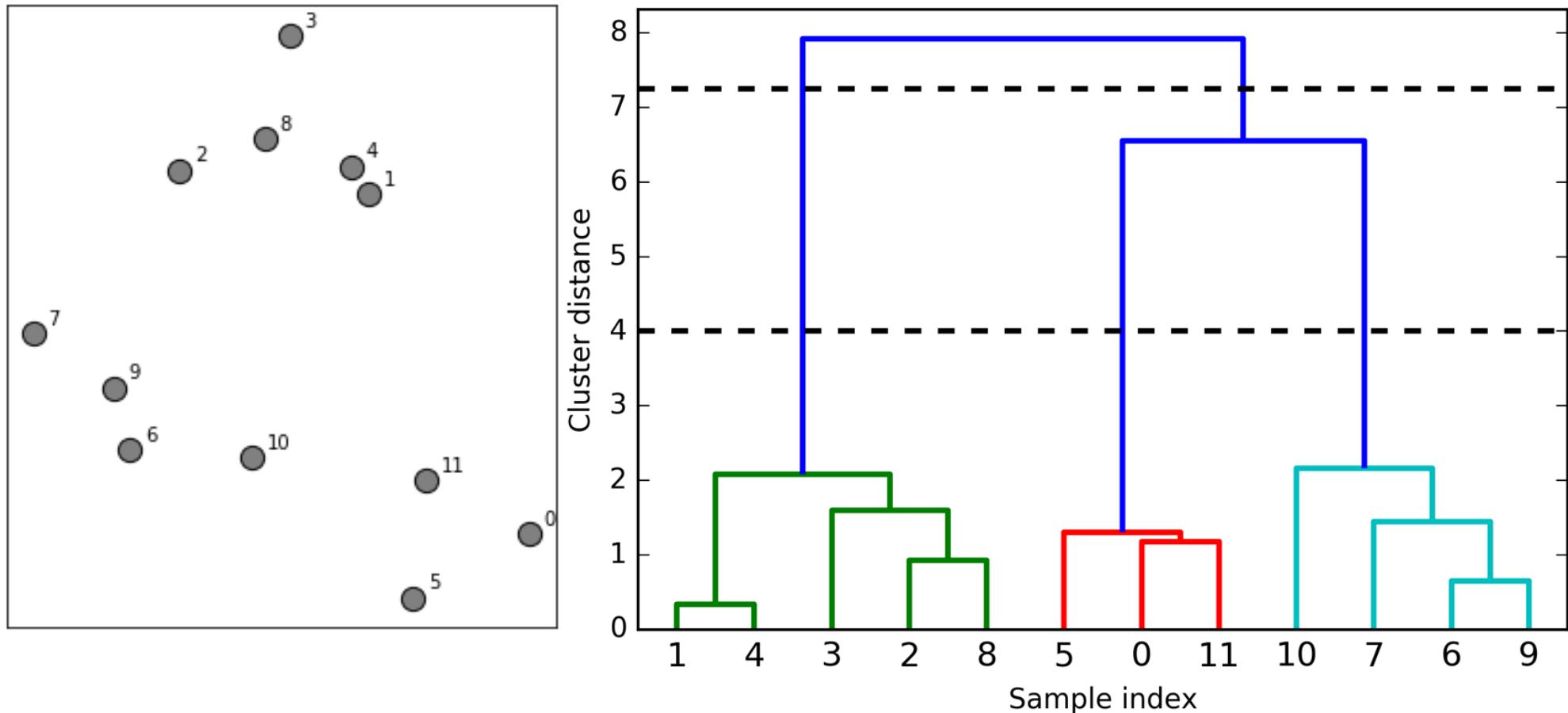
# Visualizing agglomerative clustering

---

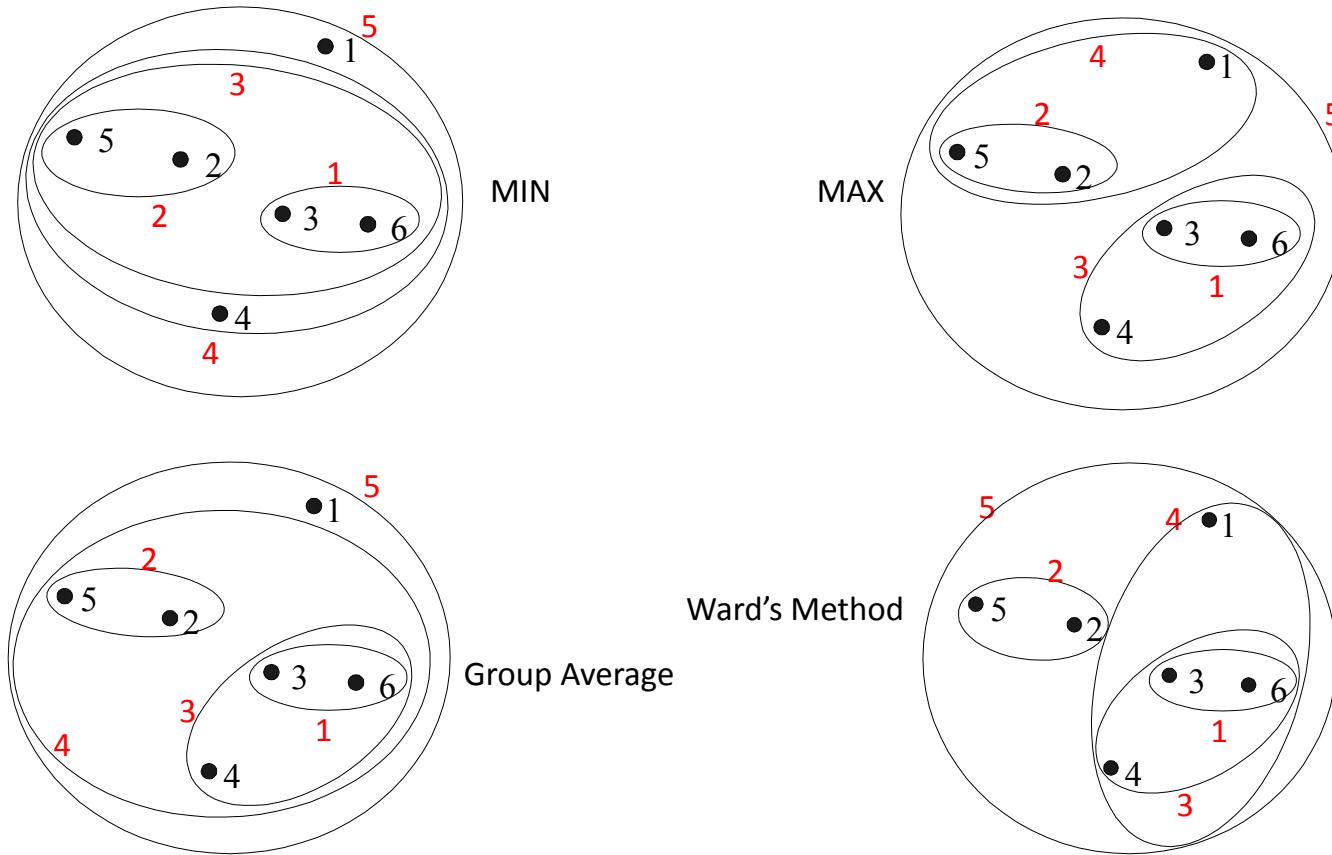


# Dendrogram

---



# Agglomerative clustering in action



<https://github.com/FelipeURJC/clustering>

# Agglomerative clustering recap

---

- OKs
  - Do not have to assume any particular number of clusters
  - They may correspond to meaningful taxonomies (e.g. shopping websites)
- KOs:
  - Does not adapt to new examples (predict)
  - Partitioning rather than clustering the data, and depending on the linkage the following problems might occur:
    - Sensitivity to noise and outliers
    - Difficulty handling different sized clusters and irregular/complex shapes
    - Breaking large clusters

# Outline

---

1. What is clustering?
2. K-means
3. Hierarchical
4. DBSCAN
5. Mixture of gaussians
6. Spectral Clustering
7. Time series clustering

# DBSCAN

---

- Stands for “density-based spatial clustering of applications with noise”
- Main features:
  - It does not require the user to set the number of clusters a priori
  - It can capture clusters of complex shapes
  - It can identify points that are not part of any cluster
  - It is somewhat slower than agglomerative clustering and k-means, but still scales to relatively large datasets
- The idea behind DBSCAN is that clusters form dense regions of data, separated by regions that are relatively empty



# DBSCAN algorithm

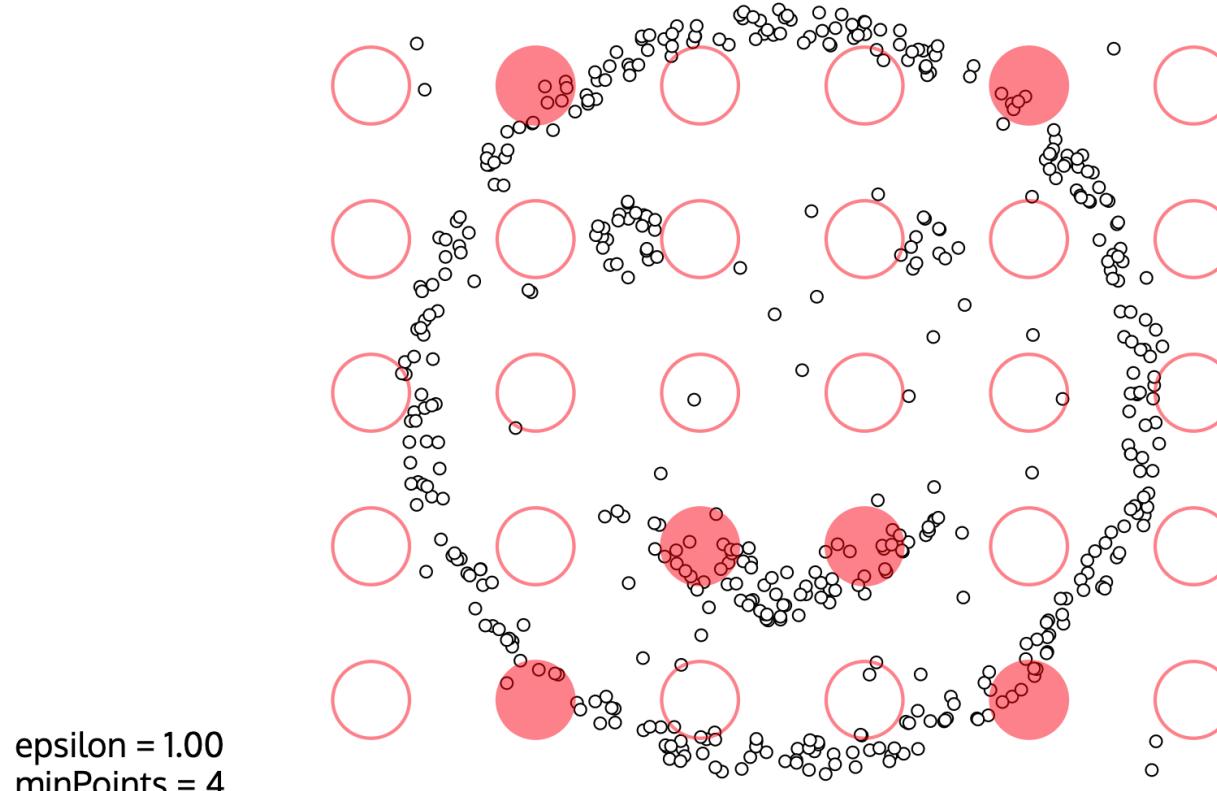
---

1. The algorithm works by picking an arbitrary point to start with
2. It then finds all points with distance **eps** or less from that point
3. If there are less than **min\_samples** points within distance **eps** of the starting point
  - this point is labeled as **noise**, meaning that it doesn't belong to any cluster
4. If there are more than **min\_samples** points within a distance of **eps**
  - the point is labeled as **core sample** and assigned a new cluster label
5. Then, all neighbors (within **eps**) of the point are visited
6. If they have not been assigned a cluster yet, they are assigned the new cluster label that was just created.
7. If they are **core samples**, their neighbors are visited in turn, and so on. The cluster grows until there are no more core samples within distance **eps** of the cluster.
8. Then another point that hasn't yet been visited is picked, and the same procedure is repeated.



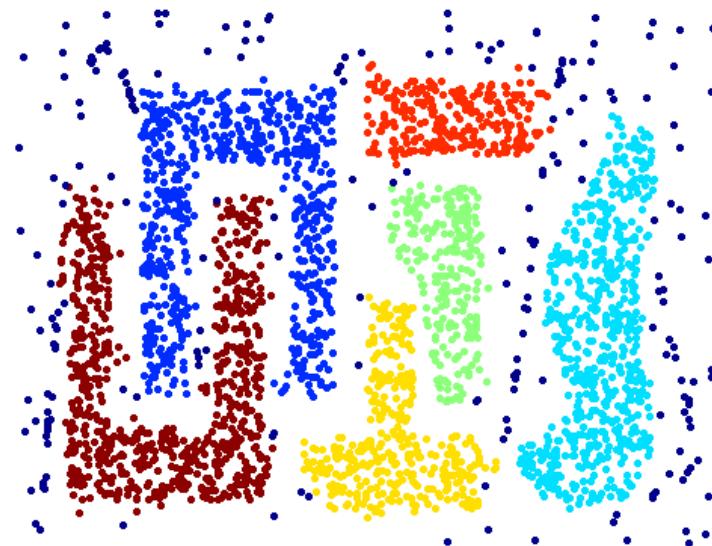
# DBSCAN algorithm

- Better if we visualize how it works



# DBSCAN in action

---



<https://github.com/FelipeURJC/clustering>

# DBSCAN recap

---

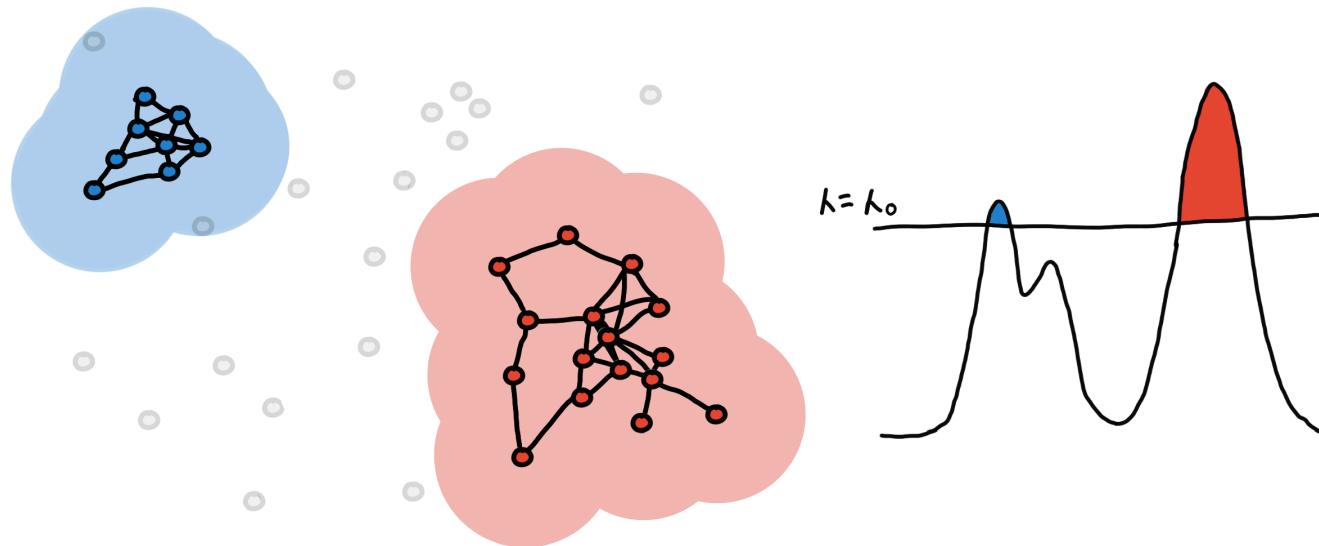
- OKs (already mentioned)
  - Resistant to Noise
  - Can handle clusters of different shapes and sizes
- KOs:
  - Does not adapt to new examples (predict)
  - Cannot handle varying densities
  - Sensitive to parameters: hard to determine the correct set of parameters



# Extra: HDBSCAN

---

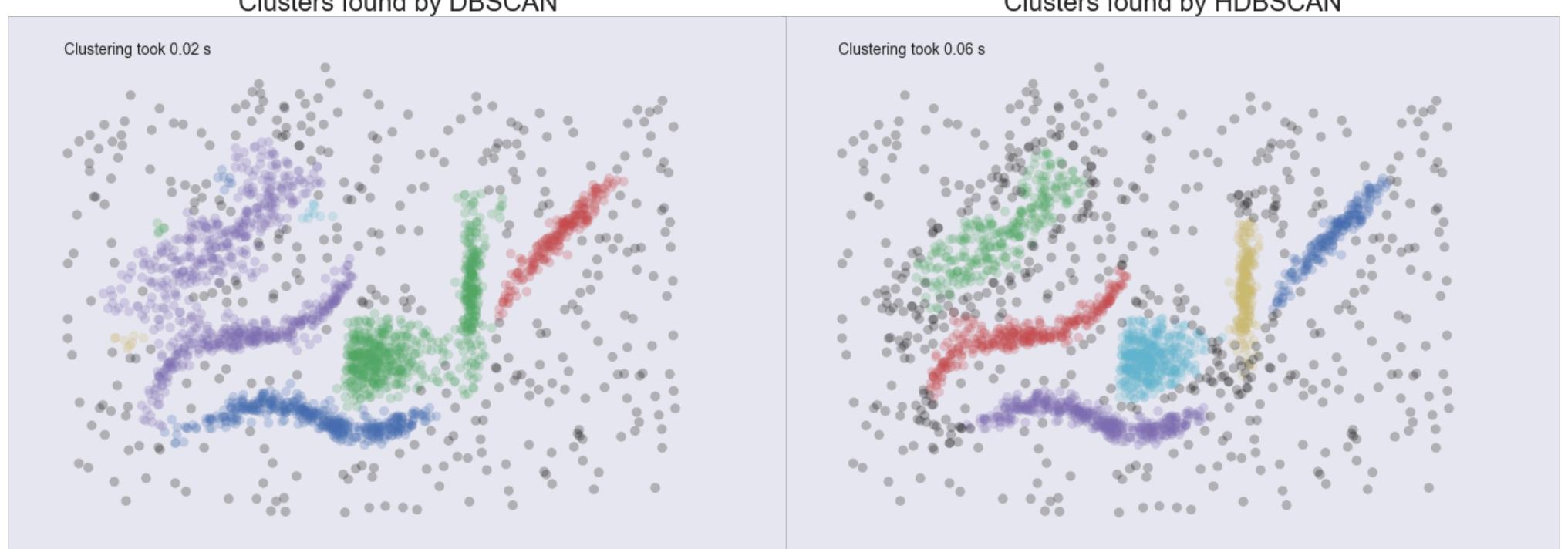
- It stands for Hierarchical DBSCAN
- It looks for regions of the data that are denser than the surrounding space



<https://github.com/scikit-learn-contrib/hdbscan>

# HDBSCAN

- Comparison to DBSCAN



<https://nbviewer.jupyter.org/github/scikit-learn-contrib/hdbscan/blob/master/notebooks/Comparing%20Clustering%20Algorithms.ipynb>

# Outline

---

1. What is clustering?
2. K-means
3. Hierarchical
4. DBSCAN
5. **Mixture of gaussians**
6. Spectral Clustering
7. Time series clustering

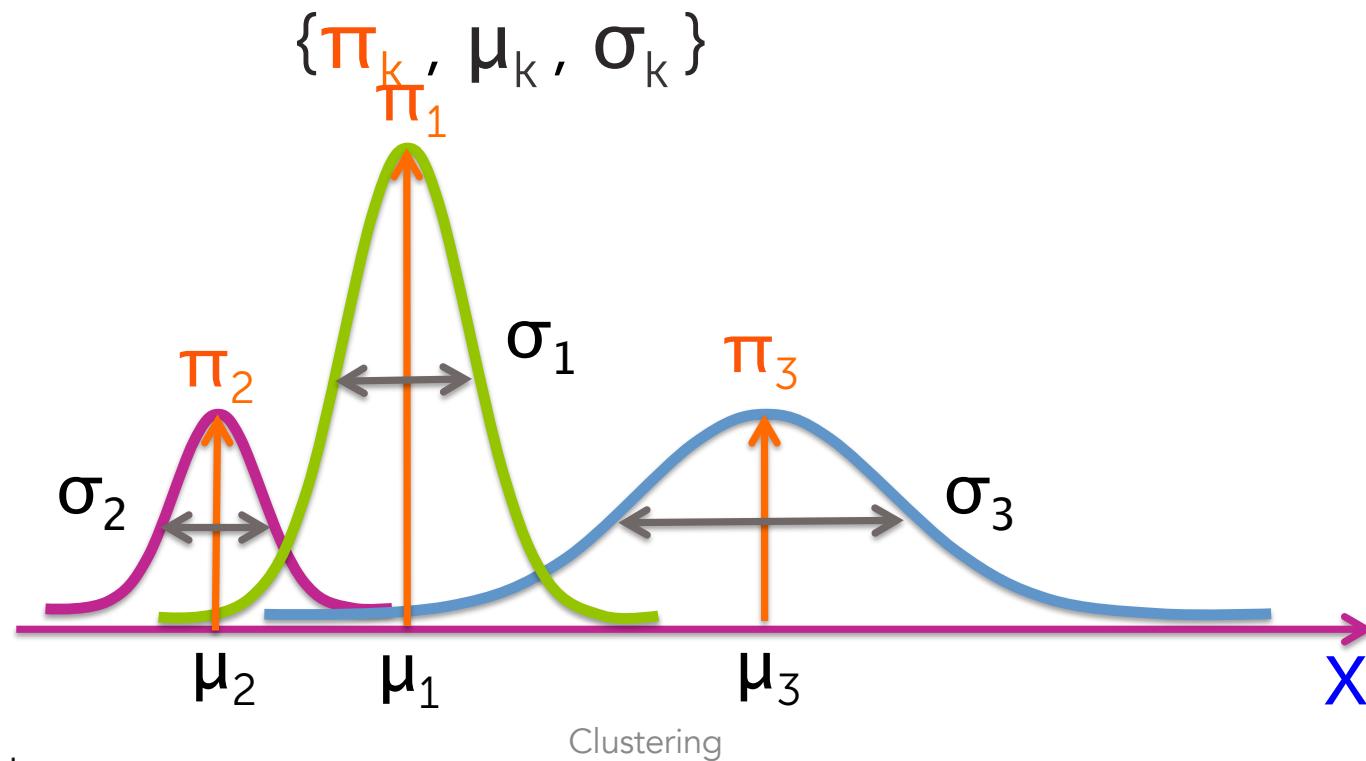
# Mixture of gaussians

---

- It is a **probabilistic model**
  - Hard vs. soft clustering
    - Hard clustering: every point belongs to exactly one cluster
    - Soft clustering: **every point belongs to several clusters with certain degrees**
      - e.g. 54% chance document is world news, 45% science, 1% sports, and 0% entertainment
  - Probabilistic clustering
    - Each cluster is mathematically represented by a parametric distribution
    - The entire data set is modeled by a mixture of these distributions

# Mixture of gaussians

- It is a **probabilistic model** that assumes all the data points are generated from a mixture of a finite number of Gaussian distributions with unknown parameters (**need to be estimated**)

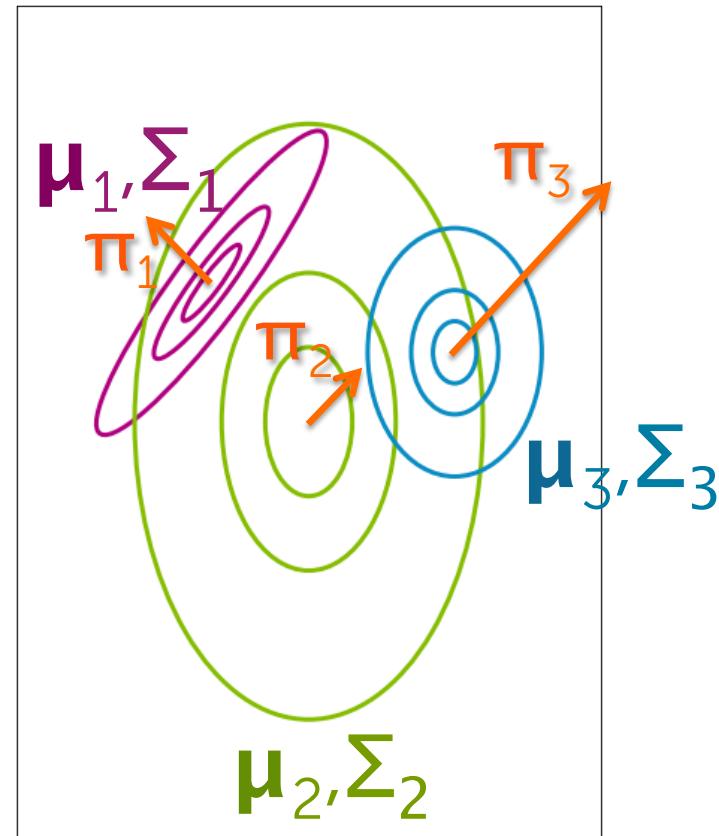


# Mixture of gaussians

---

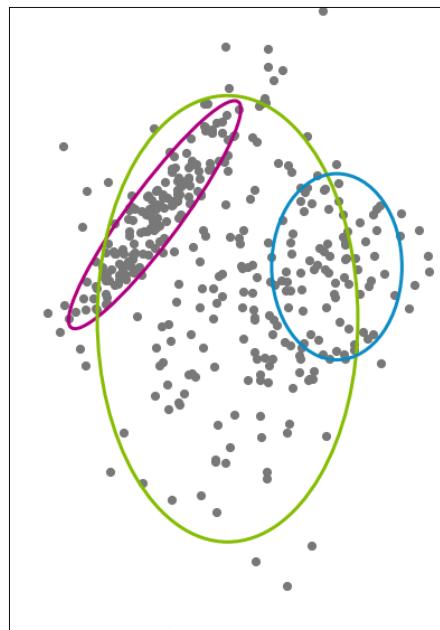
- Each mixture component represents a unique cluster specified by

$$\{\pi_k, \mu_k, \Sigma_k\}$$



# Expectation maximization (EM)

- **E-step:** for given parameter values we can compute the expected values of the responsibilities



$$r_{ik} = \pi_k N(x_i | \mu_k, \Sigma_k)$$

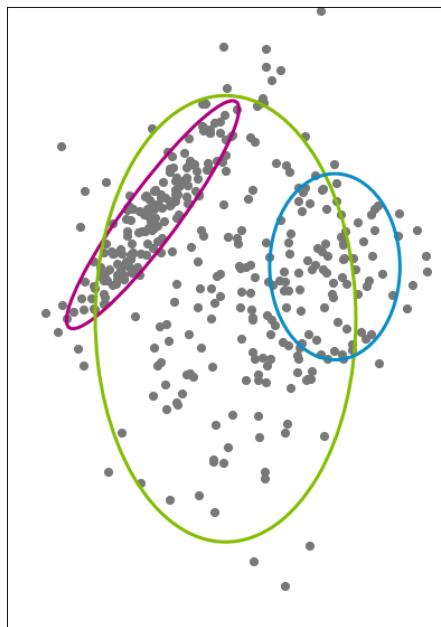
Initial probability of being from cluster k

Responsibility cluster k takes for observation i

How likely is the observed value  $x_i$  under this cluster assignment?

# Expectation maximization (EM)

- **E-step:** for given parameter values we can compute the expected values of the responsibilities



Responsibility cluster k takes for observation i

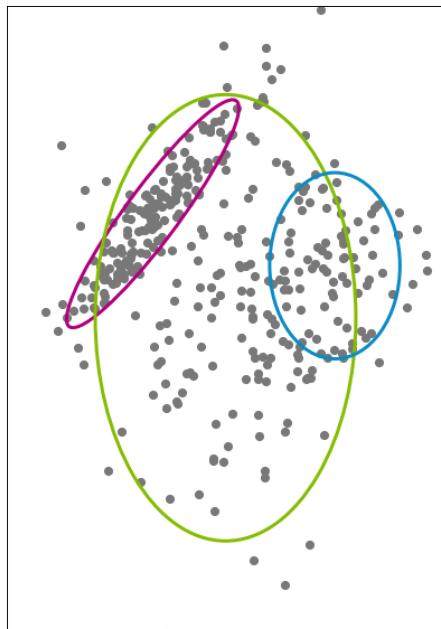
$$r_{ik} = \frac{\pi_k N(x_i | \mu_k, \Sigma_k)}{\sum_{j=1}^K \pi_j N(x_i | \mu_j, \Sigma_j)}$$

Normalized over all possible cluster assignments

# Expectation maximization (EM)

---

- **M-step**: maximize the expected complete log likelihood.
  - Update parameters



$$\pi_k = \frac{\sum_i r_{ik}}{n} \quad \mu_k = \frac{\sum_i r_{ik} x_i}{\sum_i r_{ik}}$$
$$\Sigma_k = \frac{\sum_i r_{ik} (x_i - \mu_k)(x_i - \mu_k)^T}{\sum_i r_{ik}}$$

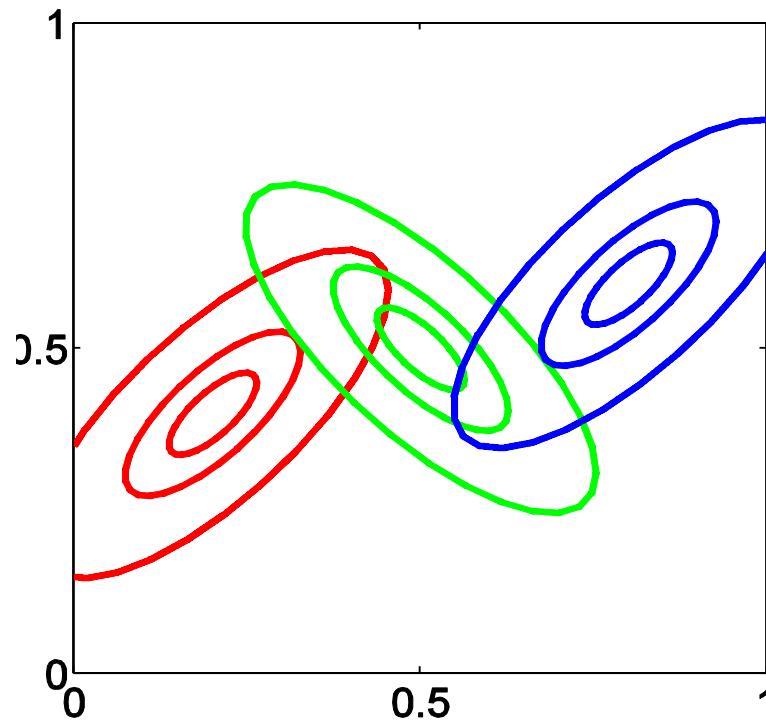
# Expectation maximization (EM)

---

- Iterate E-step and M-step until the log likelihood of data does not increase any more.
  - Converge to local optimal
  - Need to restart algorithm with different initial guess of parameters (as in K-means)

# Mixture of gaussians in action

---



<https://github.com/FelipeURJC/clustering>

# Mixture of gaussians recap

---

- OKs:
  - Give probabilistic cluster assignments
  - Have probabilistic interpretation
  - Can handle clusters with varying sizes, variance etc.
- KOs:
  - Gaussian distribution assumption
  - log likelihood can have many local minima and multiple restarts are required (like k-means)
  - Overfitting issues

# Outline

---

1. What is clustering?
2. K-means
3. Hierarchical
4. DBSCAN
5. Mixture of gaussians
6. **Spectral Clustering**
7. Time series clustering

# Spectral clustering

---

- Graph-based method that clusters data using eigenvectors (**spectral decomposition**) of matrices derived from the distance between points
- Similarity graphs: given a set of points  $X = \{x_1, x_2, \dots, x_n\}$  with pairwise similarities  $s_{ij}$  or pairwise distances  $d_{ij}$ , transform them into a graph:
  - **$\epsilon$ -neighborhood graph**: we connect all points whose pairwise distances are smaller than  $\epsilon$
  - **K-nn graph**: connect  $x_i$  with  $x_j$  if  $x_j$  is among k-nn of  $x_i$
  - **Fully connected** (or affinity matrix A)

$$s_{ij} = s(x_i, x_j) = A_{ij} = \begin{cases} \exp(-||x_i - x_j||^2 / 2\sigma^2), & \text{if } i \neq j \\ 0, & \text{if } i = j \end{cases}$$

# Spectral clustering: algorithm

- Graph Laplacian:  $L = D - A$ 
  - $D$  diagonal matrix whose  $(i,i)$ -element is the sum of  $A$ 's  $i$ -th row

## Normalized spectral clustering according to Ng, Jordan, and Weiss (2002)

Input: Similarity matrix  $S \in \mathbb{R}^{n \times n}$ , number  $k$  of clusters to construct.

- Construct a similarity graph by one of the ways described in Section 2.  
Let  $W$  be its weighted adjacency matrix.
- Compute the normalized Laplacian  $L_{\text{sym}}$ .
- **Compute the first  $k$  eigenvectors  $u_1, \dots, u_k$  of  $L_{\text{sym}}$ .**
- Let  $U \in \mathbb{R}^{n \times k}$  be the matrix containing the vectors  $u_1, \dots, u_k$  as columns.
- **Form the matrix  $T \in \mathbb{R}^{n \times k}$  from  $U$  by normalizing the rows to norm 1,**  
that is set  $t_{ij} = u_{ij}/(\sum_k u_{ik}^2)^{1/2}$ .
- For  $i = 1, \dots, n$ , let  $y_i \in \mathbb{R}^k$  be the vector corresponding to the  $i$ -th row of  $T$ .
- Cluster the points  $(y_i)_{i=1, \dots, n}$  with the  $k$ -means algorithm into clusters  $C_1, \dots, C_k$ .

Output: Clusters  $A_1, \dots, A_k$  with  $A_i = \{j \mid y_j \in C_i\}$ .

<https://papers.nips.cc/paper/2092-on-spectral-clustering-analysis-and-an-algorithm.pdf>

<https://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=D95EF48A1E4714D5C6865E945D6E3B57?doi=10.1.1.165.9323&rep=rep1&type=pdf>

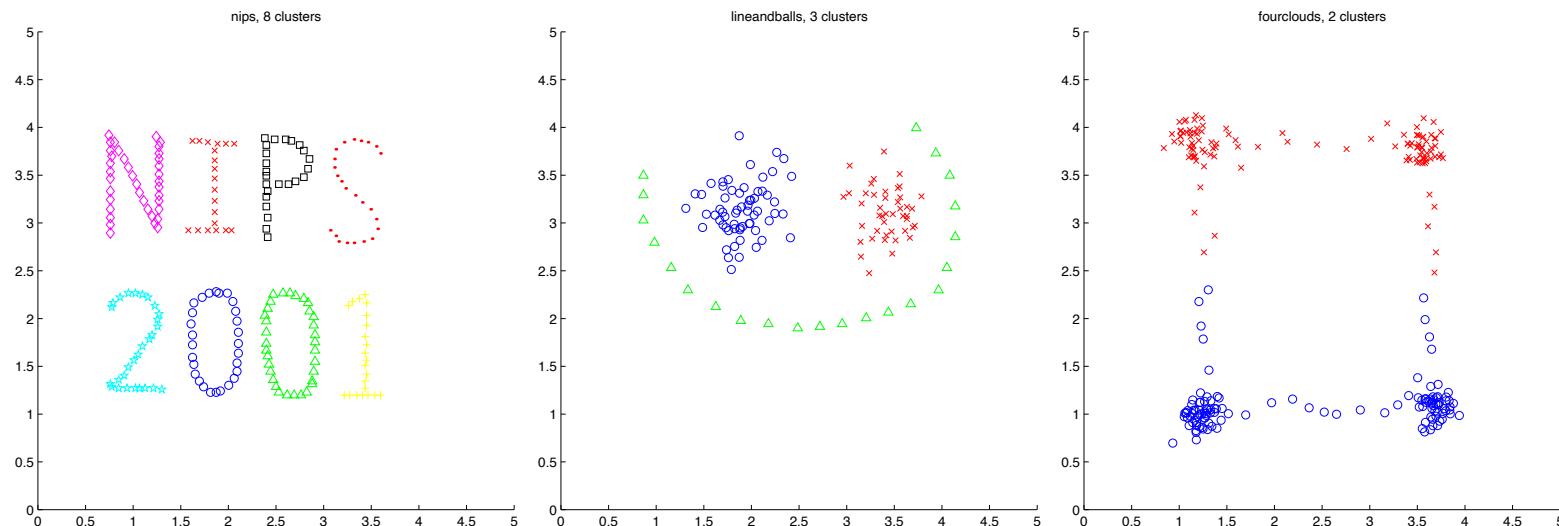
# Spectral clustering: intuition

---

- **Spectral clustering** avoids the curse of dimensionality by adding a pre-clustering step to your algorithm
  - Curse of dimensionality: as the number of dimensions increases, a distance-based similarity measure converges to a constant value between any given examples
- Other possibilities
  - Reduce the dimensionality of feature data by using PCA
  - Reduce the dimensionality of feature data by using embeddings

# Spectral clustering in action

---



<https://github.com/FelipeURJC/clustering>

# Spectral clustering recap

---

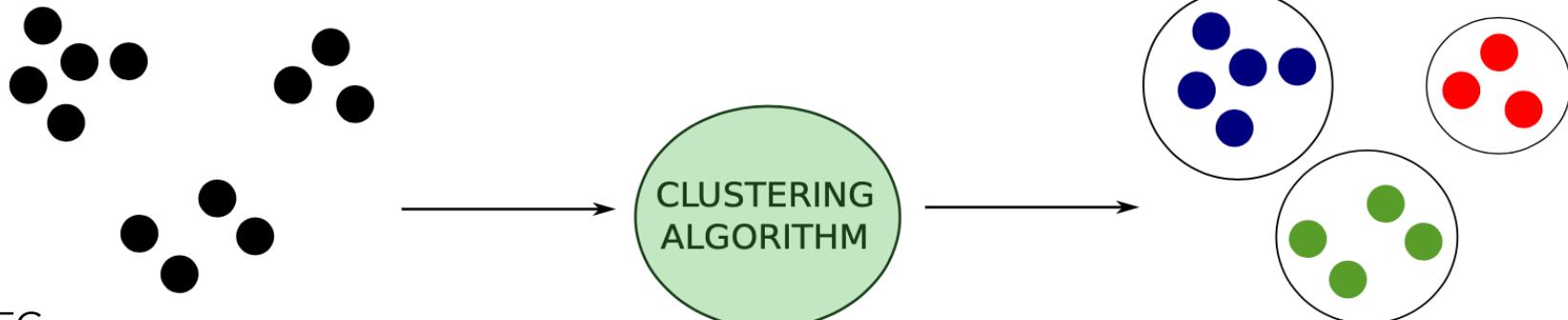
- OKs:
  - Clustering formulated as a graph cut problem
  - More stable and scalable than K-means (manifold learning)
- KOs:
  - Those inherited from K-means:
    - Number of clusters
    - Noise and outliers

# Outline

---

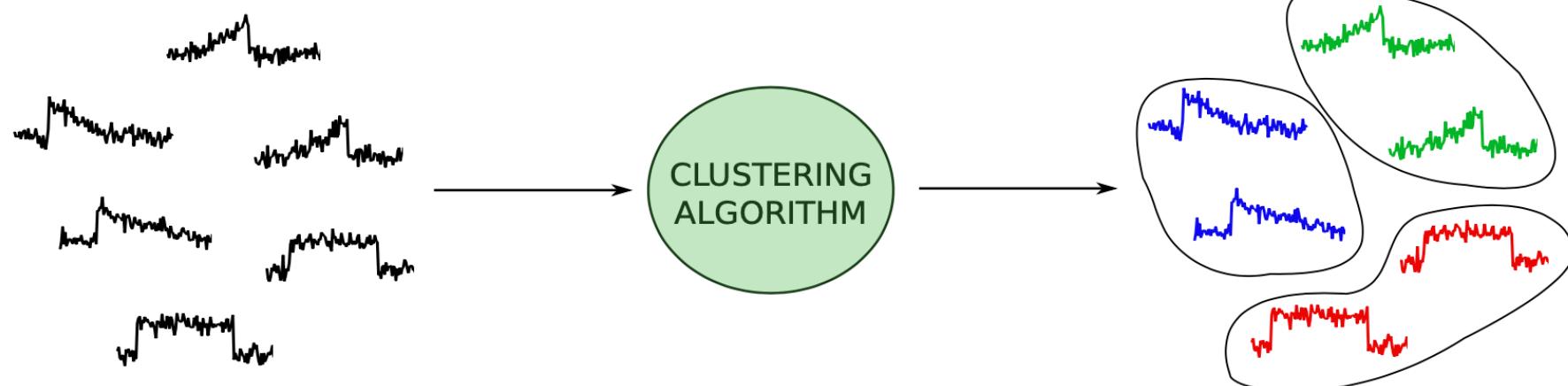
1. What is clustering?
2. K-means
3. Hierarchical
4. DBSCAN
5. Mixture of gaussians
6. Spectral Clustering
7. **Time series clustering**

# Time series clustering



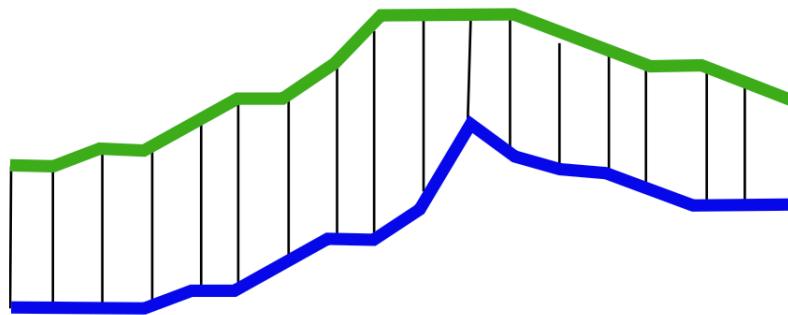
- ECG, EEG
- Gait analysis
- Financial stocks
- Electric consumption
- Account balance

A distance metric is needed

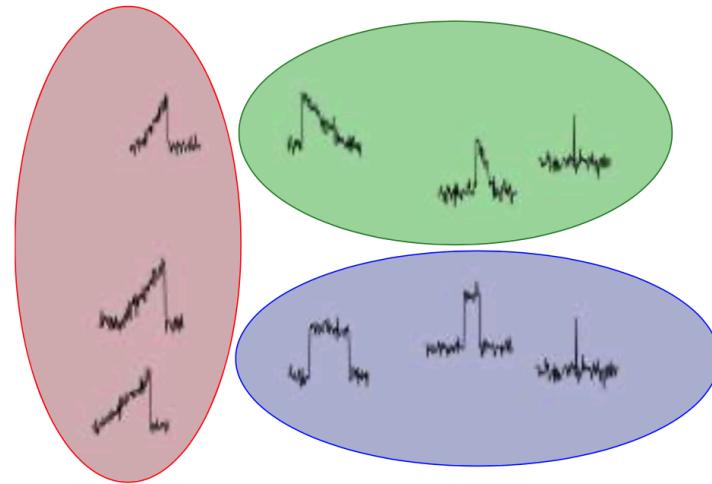
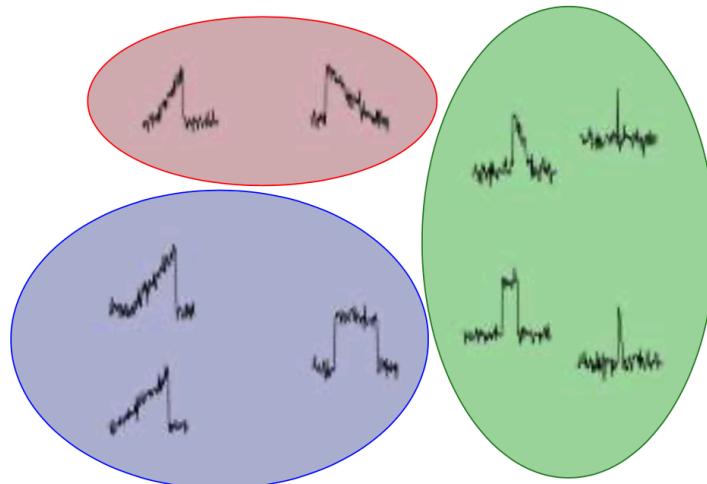
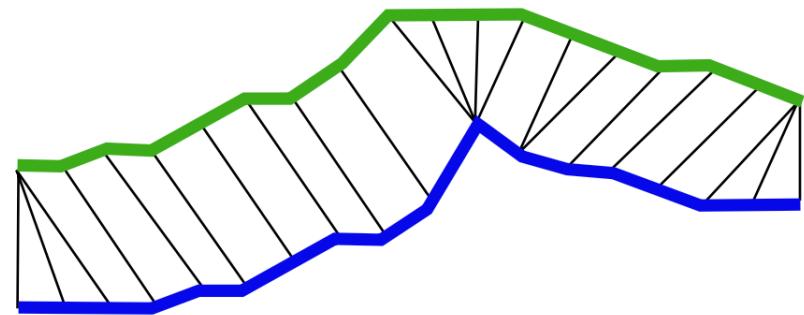


# Time series clustering: distance metrics

Rigid distance: Euclidean



Flexible distance: Dynamic Time Warping



Clustering

# Time series clustering: alternatives

---

- Represent each series by means of a set of **features** and calculate the distance between the features
- Learn a **parametric model** for each series and calculate the distance between the parameters

A nice review on time series clustering can be found [here](#)

# References

---

- D. Xu, Y. Tian, "[A comprehensive survey of clustering algorithms](#)", Ann. Data. Sci. 2015.
- A. Ng, M. Jordan, and Y. Weiss, "[On spectral clustering: Analysis and an algorithm](#)," NIPS-2001
- U. Von Luxburg, "[A tutorial on spectral clustering](#)," MPI for Biological Cybernetics, TR-149, 2006.
- A Abanda, et al. "[A review on distance based time series classification](#)", Data Mining and Knowledge Discovery 2019.
- A. Müller, S. Guido, "[Introduction to machine learning with Python](#)", O'Reilly Media 2016
- AA Patel, "[Hands-on unsupervised learning with Python](#)", O'Reilly Media 2019

