MACHINE LEARNING

Unsupervised learning

Olivier LÉZORAY

olivier.lezoray@unicaen.fr

https://lezoray.users.greyc.fr/

LEARNING OBJECTIVES

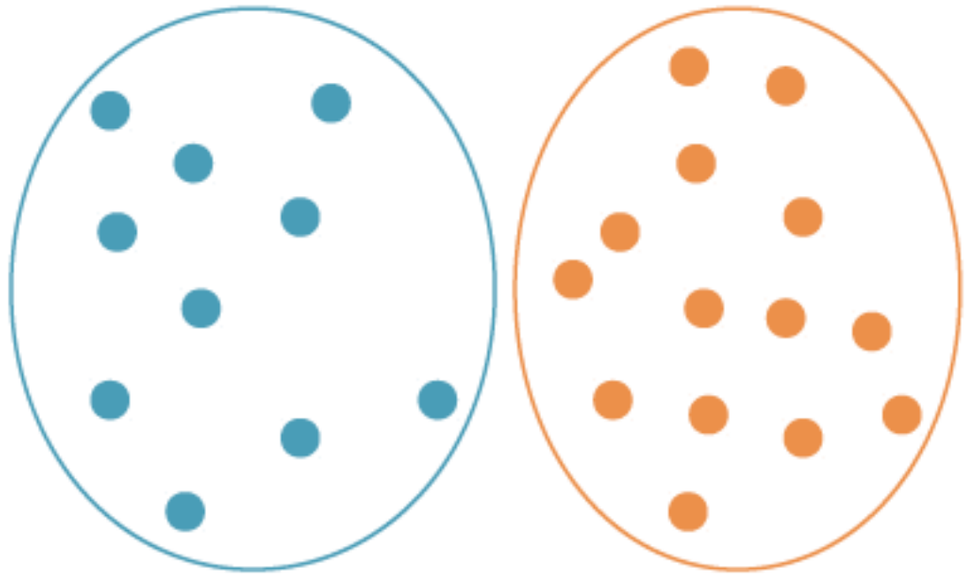
- For what clustering algorithms can be used for ?
- Recall on Distances
- How to evaluate clusterings qualities
- Present different clustering methods
  - Hierarchical clustering
  - Partitioning methods (probabilistic or not):
    - K-Means
    - Gaussian Mixture Models
  - Density based :
    - DBSCAN
  - Graph based :
    - Spectral Clustering

LEARNING OBJECTIVES

- **For what clustering algorithms can be used for ?**
- Recall on Distances
- How to evaluate clusterings qualities
- Present different clustering methods
  - Hierarchical clustering
  - Partitioning methods (probabilistic or not):
    - K-Means
    - Gaussian Mixture Models
  - Density based :
    - DBSCAN
  - Graph based :
    - Spectral Clustering

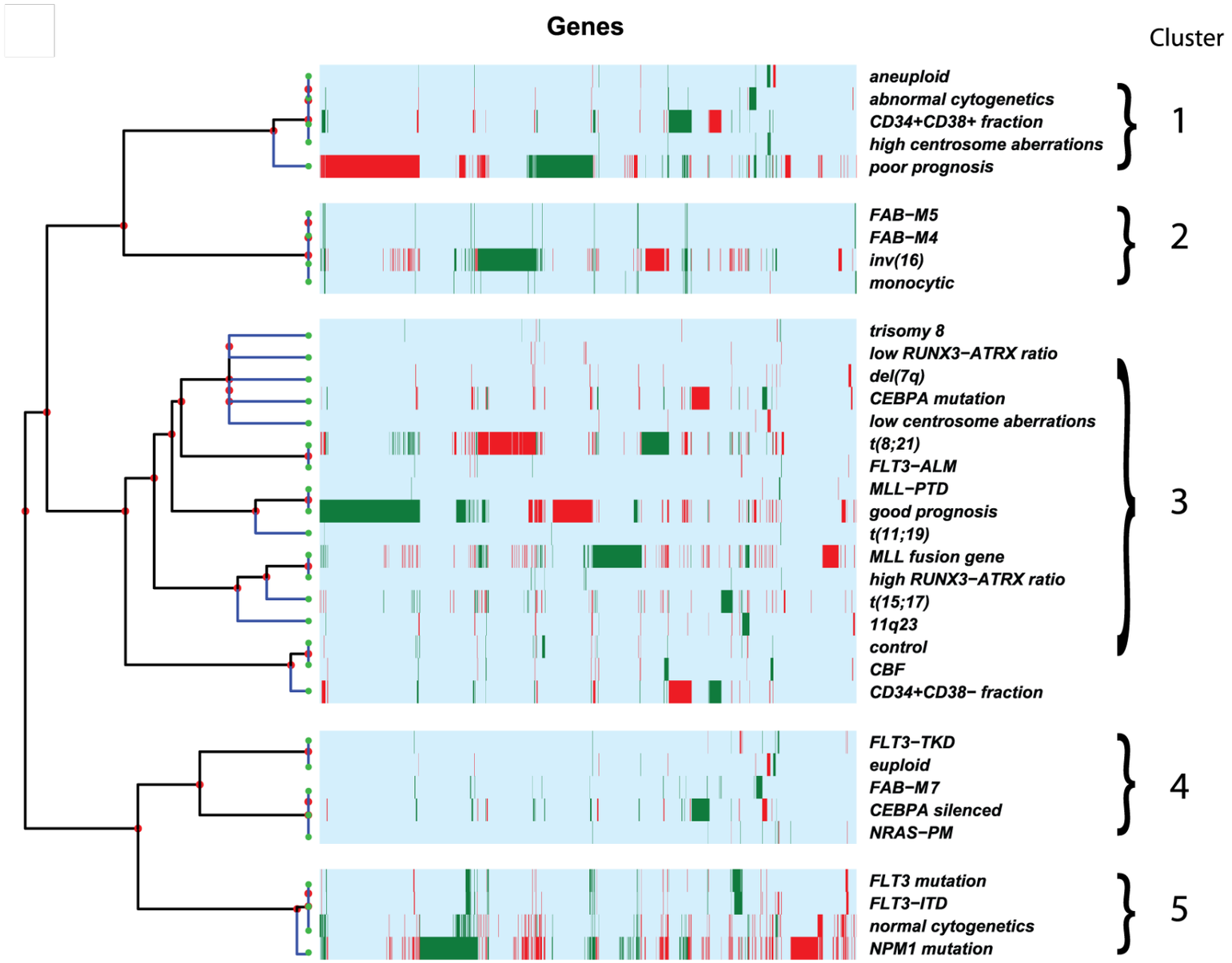
FOR WHAT CLUSTERING ALGORITHMS CAN BE USED FOR ?

- Goal : group objects that are similar into clusters : classes are unknown beforehand



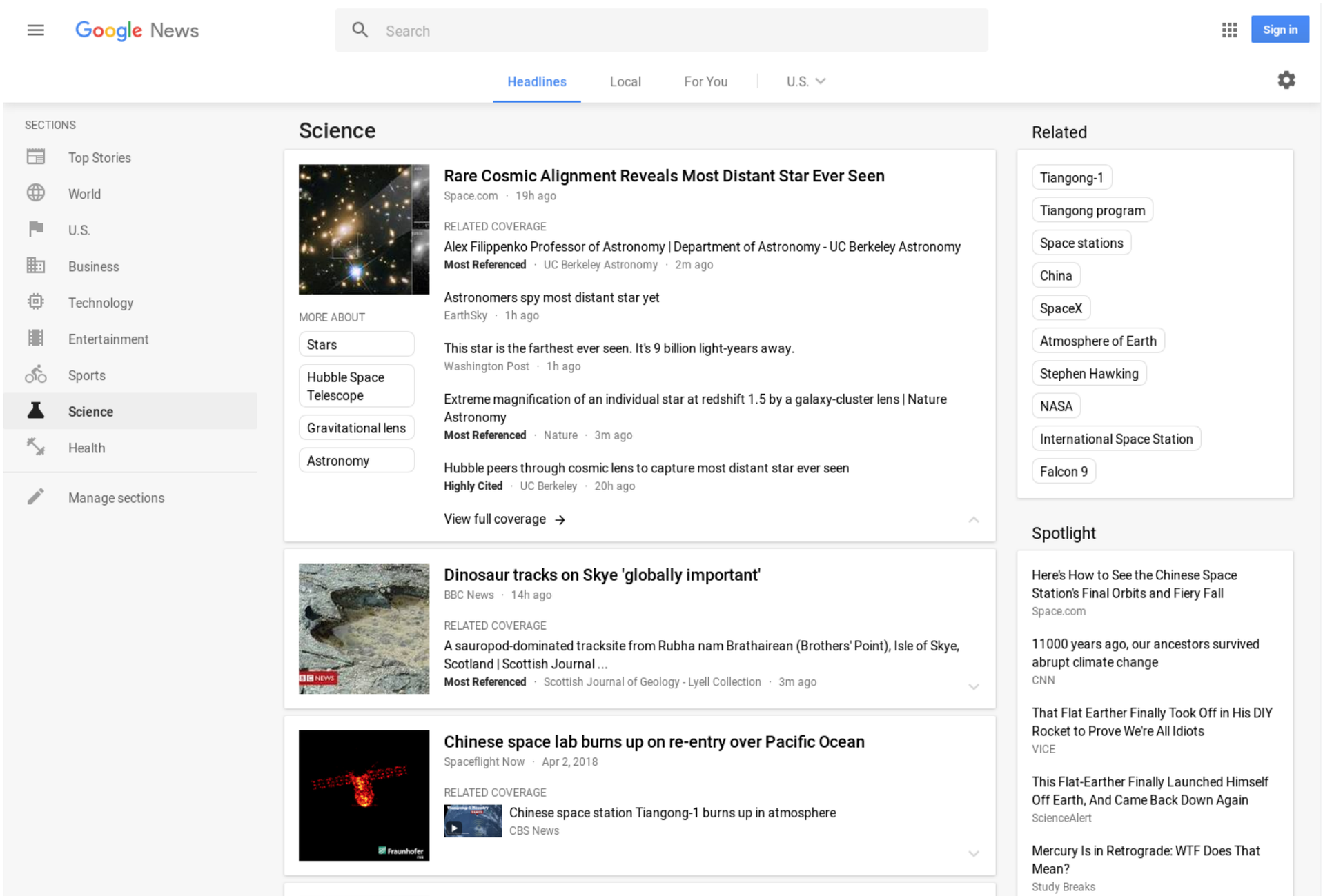
FOR WHAT CLUSTERING ALGORITHMS CAN BE USED FOR ?

- Group genes that are similarly affected by a disease



FOR WHAT CLUSTERING ALGORITHMS CAN BE USED FOR ?

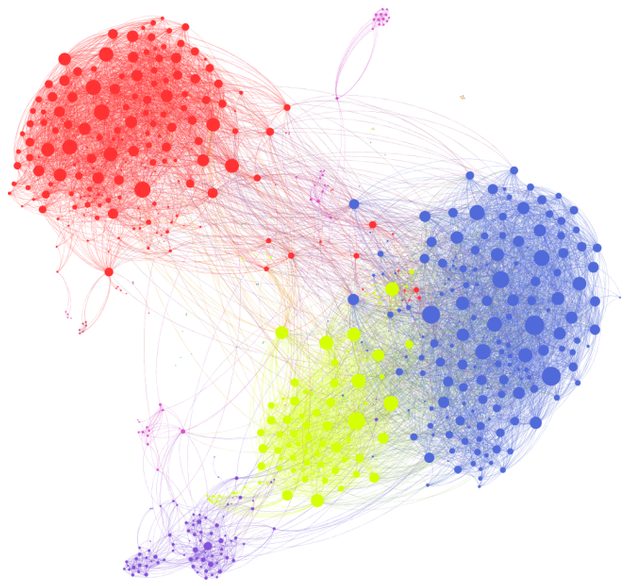
- Group similar web pages into categories



APPLICATIONS OF CLUSTERING

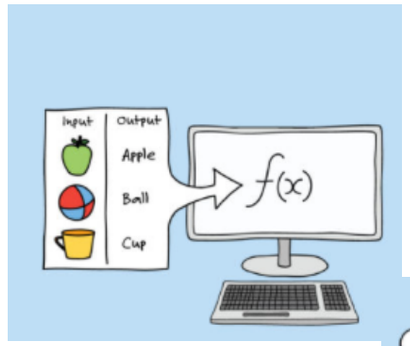
- **Understand** the general characteristics of the data
- **Visualize** the data
- **Infer** some properties of a data point based on how it relates to other points

- Example :
  - Detect communities in social networks



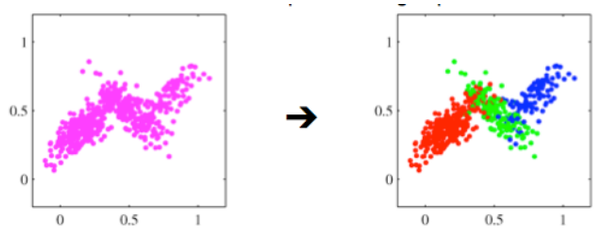
CLUSTERING IS UNSUPERVISED LEARNING

- Machine learning methods can be supervised or unsupervised
- **Supervised learning**
  - Each example of the dataset is made of **labeled observations**
    - The observations are vectorial data $\mathbf{x}_i \in \mathbb{R}^d$
    - Each observation has a label $\{(\mathbf{x}_1, y_i), \cdots, (\mathbf{x}_N, y_N)\}$
  - The learning is supervised since it is guided by a supervisor (the labeling of the data points)

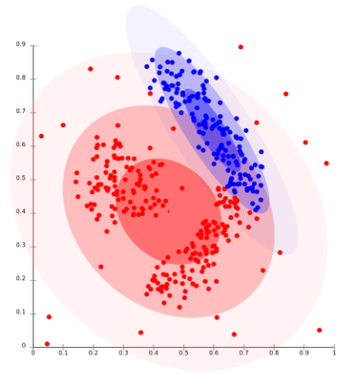


- **Unsupervised learning**
  - These methods operate on **unlabeled observations** $\{\mathbf{x}_1, \cdots, \mathbf{x}_N\}$
  - They are unsupervised since no label of reference is given for each example
  - Given a set of observations without labels, we want to capture the structure of the data

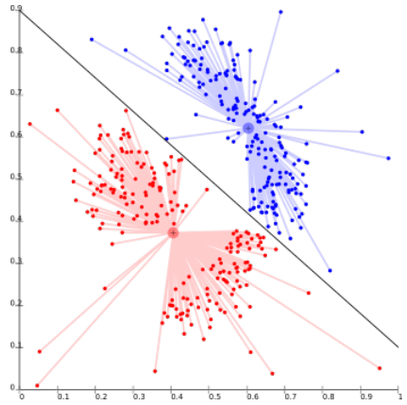
CLUSTERING METHODS



- **Parametric unsupervised learning**
  - Based on an estimate of the underlying density (Mixture of Gaussians, etc.)



- **Non-parametric unsupervised learning**
  - No density estimation function is used.
  - The aim is to extract significant groupings from the data.
  - This is based on several steps:
    - Defining a measure of (di)similarity between the examples
    - Defining a criterion to be optimized for clustering
    - Defining an algorithm to minimize (or maximize) this criterion

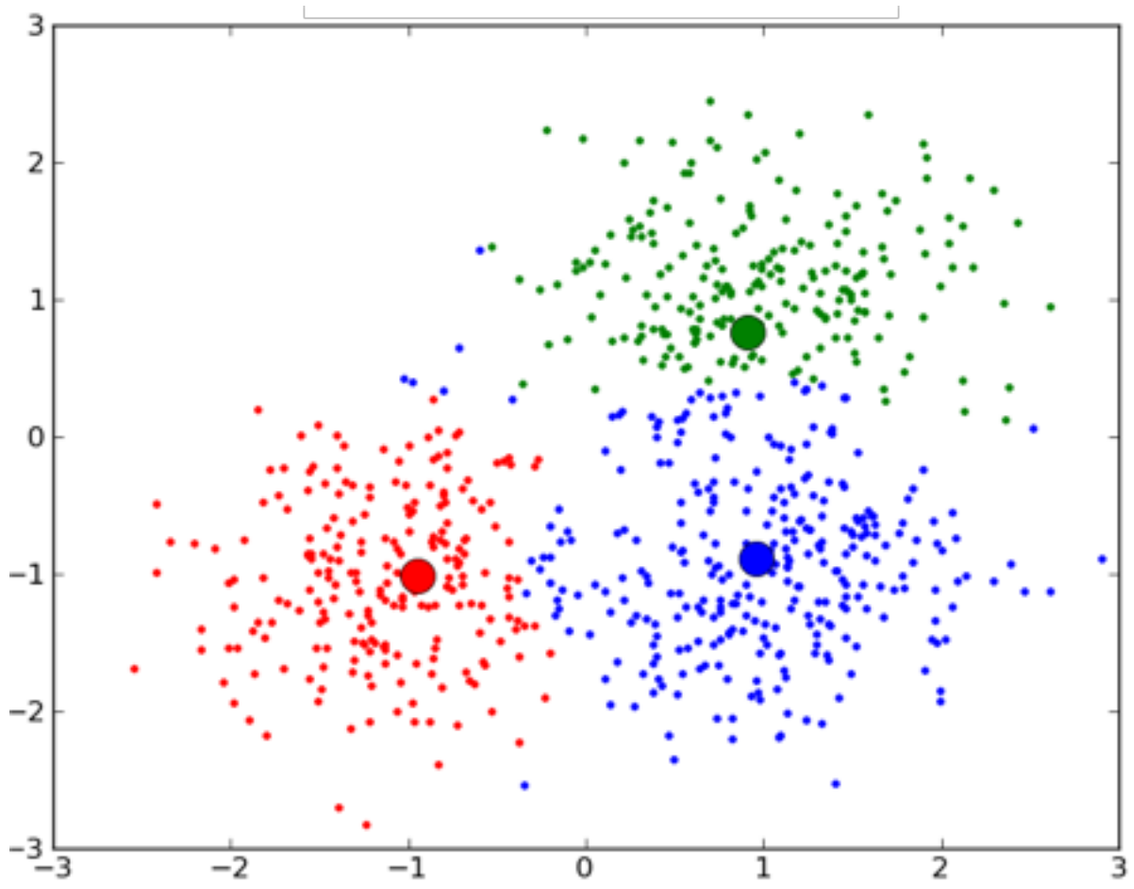


LEARNING OBJECTIVES

- For what clustering algorithms can be used for ?
- **Recall on Distances**
- How to evaluate clusterings qualities
- Present different clustering methods
  - Hierarchical clustering
  - Partitioning methods (probabilistic or not):
    - K-Means
    - Gaussian Mixture Models
  - Density based :
    - DBSCAN
  - Graph based :
    - Spectral Clustering

DISTANCES

- Enable to assess how **close / far**
  - data points are from each other
  - a data point is from a cluster
  - two clusters are from each other



DISTANCES : DEFINITIONS

- **Definition of a metric distance**
  - a proximity measure $d(x, y)$ between two vectors x and y is a metric iff it satisfies the following properties :

$$d(x,y) = 0 \Leftrightarrow x = y \qquad \textit{separation} \qquad (1)$$

$$d(x,y) = d(y,x) \qquad \textit{symmetry} \qquad (2)$$

$$d(x,y) \leq d(x,z) + d(z,y) \qquad \textit{triangle inequality} \qquad (3)$$

$$d(x,y) \geq 0 \qquad \textit{positiveness} \qquad (4)$$

  - It is a norm if $d(a \cdot x, a \cdot y) = |a| d(x,y)$ and is denoted by $\|x - y\|$
  - The more general form of a metric distance is the $L_{p/r}$ norm

$$\|x - y\|_{p/r} = \left(\sum_{i=1}^{d} |x_i - y_i|^p \right)^{1/r}$$

USUAL METRIC DISTANCES

- They are derived from the standard $L_{p/r}$ norm
- Minkowski's metric (L_k norm)

$$\|x - y\|_k = \left(\sum_{i=1}^{d} |x_i - y_i|^k \right)^{1/k}$$

- Manhattan (L_1 norm)

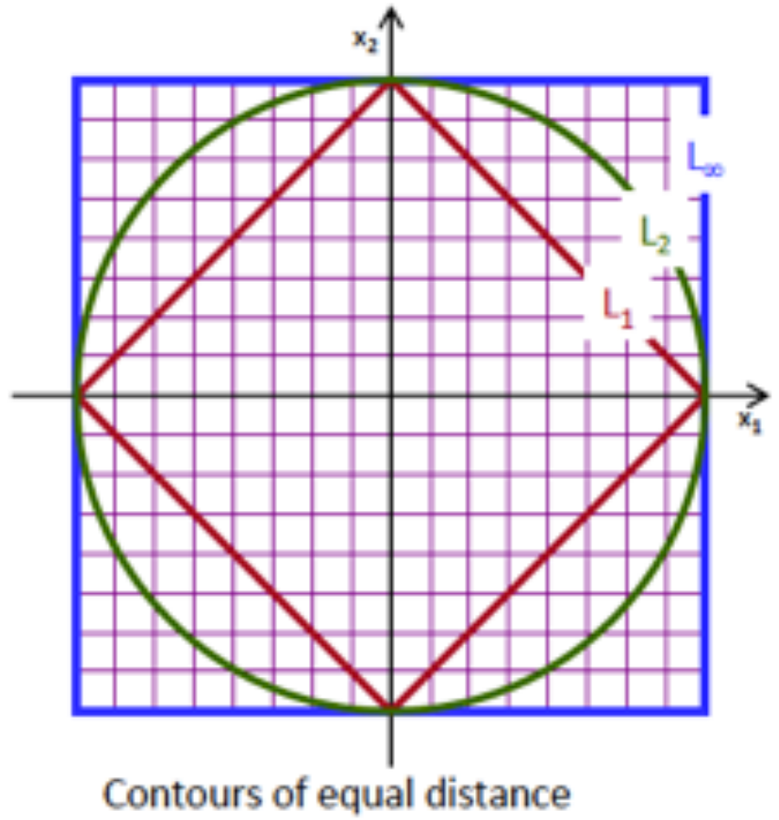
$$\|x - y\|_1 = \left(\sum_{i=1}^{d} |x_i - y_i| \right)$$

- Euclidean (L_2 norm)

$$\|x - y\|_2 = \left(\sum_{i=1}^{d} |x_i - y_i|^2 \right)^{1/2}$$

- Chebyshev (L_{inf} norm)

$$\|x - y\|_\infty = \left(\max_{i=1}^{d} |x_i - y_i| \right)$$



Contours of equal distance

NORMALIZATION

- The problem : features with large values dominate the others in the distance computation
  - maximum distance from salary: 100000-19000 = 81000
  - maximum distance from age: 52-27 = 25

ID	Sexe	Age	Salaire
1	F	27	19 000
2	M	51	64 000
3	M	52	100 000
4	F	33	55 000
5	M	45	45 000

ID	Sexe	Age	Salaire
1	1	0,00	0,00
2	0	0,96	0,56
3	0	1,00	1,00
4	1	0,24	0,44
5	0	0,72	0,32

- dist(ID2, ID3) = SQRT(0 + $(0.04)^2$ + $(0.44)^2$) = 0.44
- dist(ID2, ID4) = SQRT(1 + $(0.72)^2$ + $(0.12)^2$) = 1.24

- Solution : **normalize** the attributes
- Aim : get the output values between 0 and 1
- Two popular choices :

$$x_i^n = \frac{x_i - \min x_i}{\max x_i - \min x_i} \qquad x_i^n = \frac{x_i - \mu_i}{\sigma_i}$$

OTHER POPULAR METRICS

- If one wishes to give different weights to the attributes
  - **Sebestyen's distance** : W is a diagonal matrix

$$d(x,y) = \left((x-y)W(x-y)^T\right)^{1/2}$$

- If the correlated variables become too important, the Euclidean distance can be normalized by the covariance matrix C
  - **Mahalanobis distance**

$$d(x,y) = \left((x-y)C^{-1}(x-y)^T\right)^{1/2}$$

- **Note:** All these metrics provide dissimilarities (0: similar, infinity: dissimilar) that can be transformed into similarities (0: dissimilar, 1 similar).

FROM DISTANCES TO SIMILARITIES

- How to transform **distances into similarities** ?
- Many different ways

$$\text{sim}(x, y) = \frac{1}{1 + d(x, y)}$$

$$\text{sim}(x, y) = 1 - \frac{d(x, y)}{\max d(x, y)}$$

$$\text{sim}(x, y) = exp\left(-\frac{d(x, y)^2}{\sigma^2}\right)$$

LEARNING OBJECTIVES

- For what clustering algorithms can be used for ?
- Recall on Distances
- **How to evaluate clusterings qualities**
- Present different clustering methods
  - Hierarchical clustering
  - Partitioning methods (probabilistic or not):
    - K-Means
    - Gaussian Mixture Models
  - Density based :
    - DBSCAN
  - Graph based :
    - Spectral Clustering

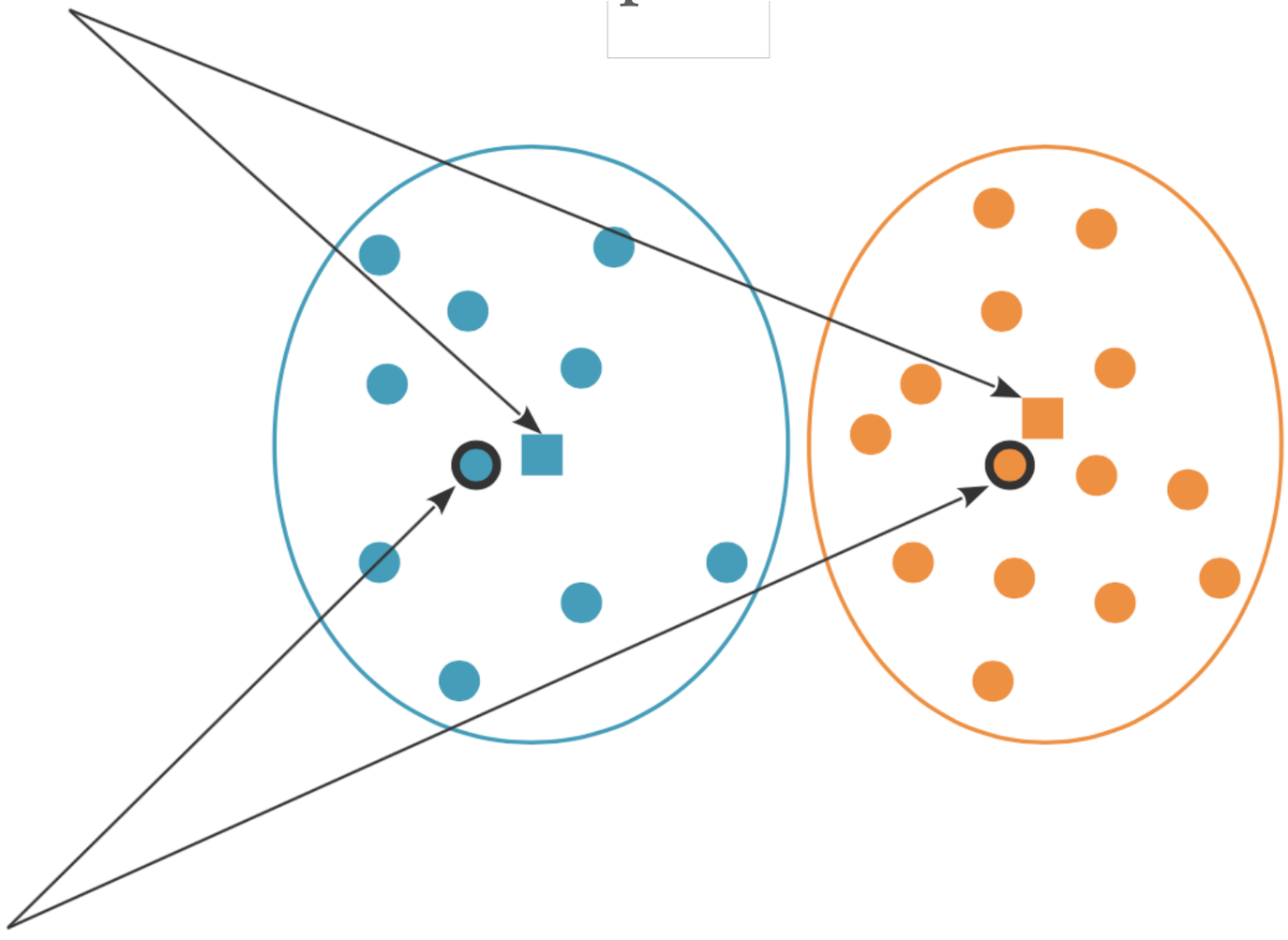
HOW TO EVALUATE CLUSTERS ?

- Clustering is **unsupervised**.
- There is no ground truth.
- How do we evaluate the quality of a clustering algorithm?

1. **Based on the shape of the clusters:**
   Points within the same cluster should be nearby/similar and points far from each other should belong to different clusters.
2. **Based on the stability of the clusters:**
   We should get the same results if we remove some data points, add noise, etc.

CENTROIDS AND MEDOIDS

- **Centroid**: mean of the points in the cluster

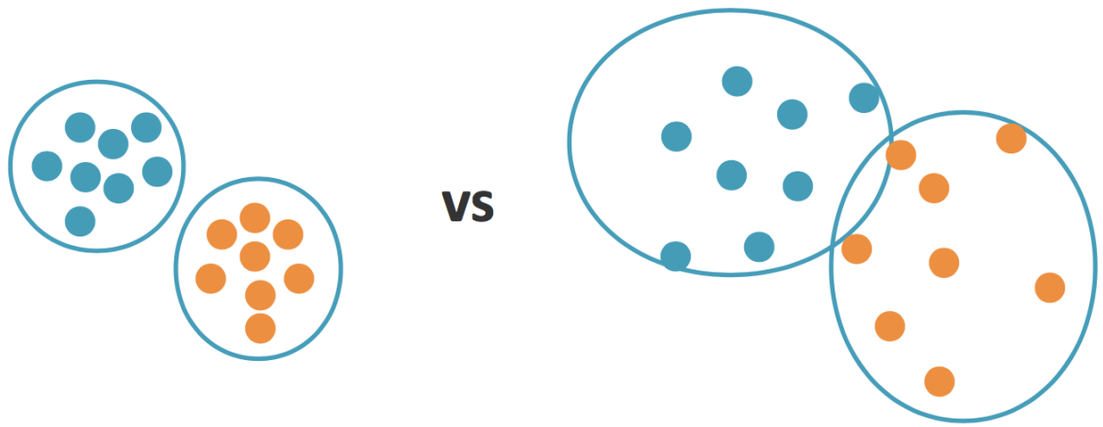


$$\mu_i = \frac{1}{|C_k|} \sum_{x_i \in C_k} x_i$$

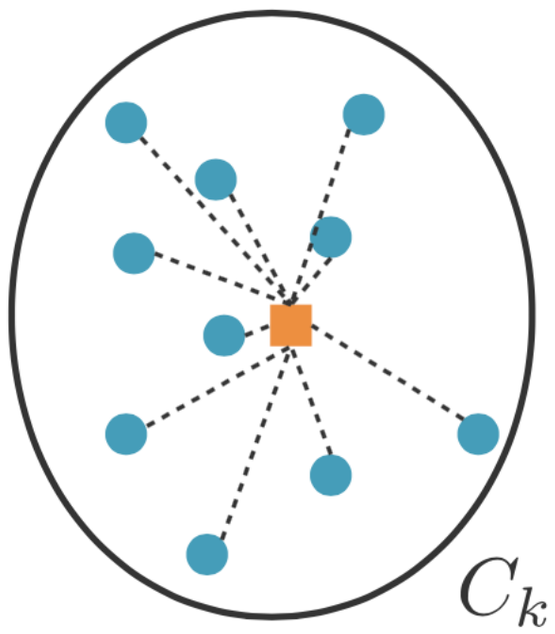
- **Medoid**: point in the cluster that is closest to the centroid

$$m_i = \arg \min_{x_i \in C_k} d(x_i, \mu_i)$$

CLUSTER SHAPE : TIGHTNESS

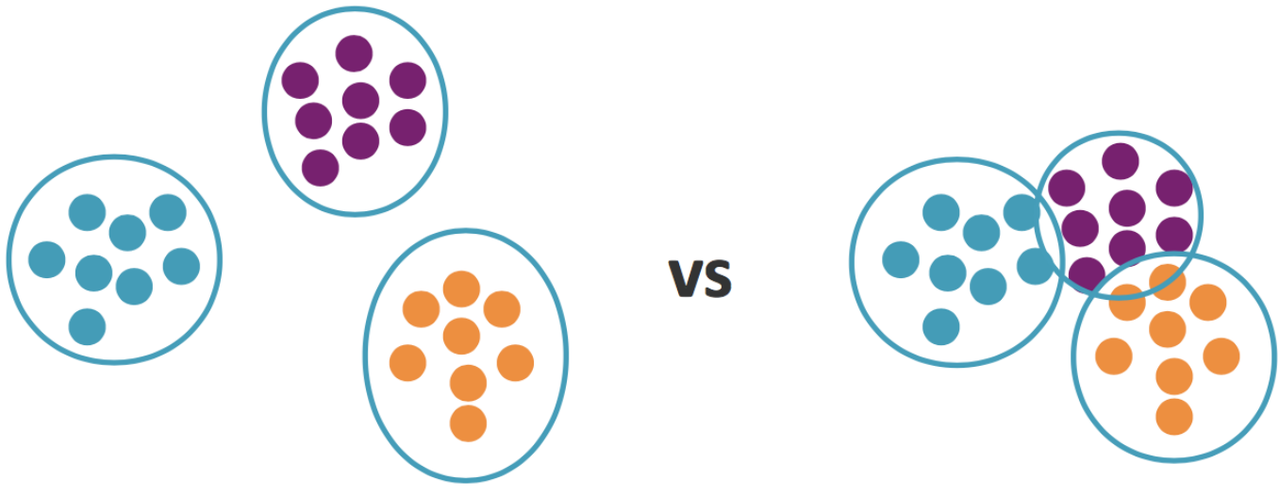


- Within-cluster inertia :
- Variance of the points within the same cluster

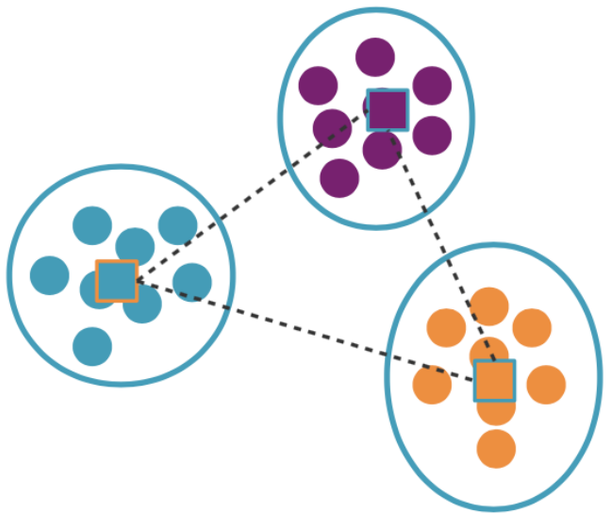


$$J_w = \sum_{k=1}^{K} \sum_{x \in C_k} d^2(x, \mu_k)$$

CLUSTER SHAPE : SEPARABILITY

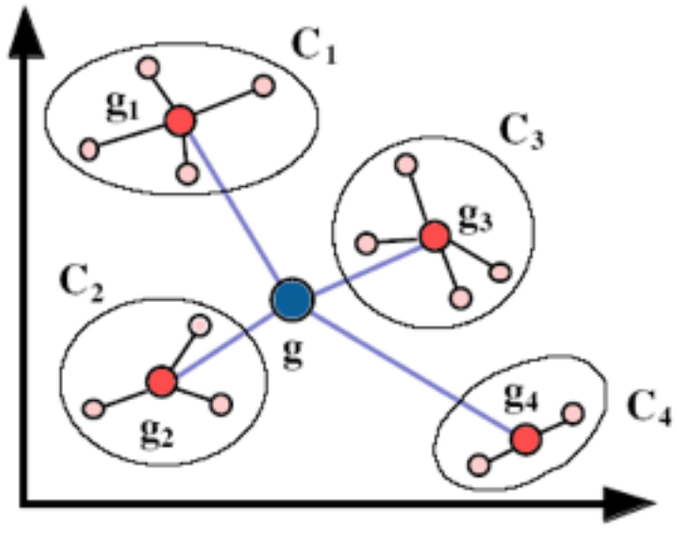


- Between clusters inertia:
  - Variance of cluster centers



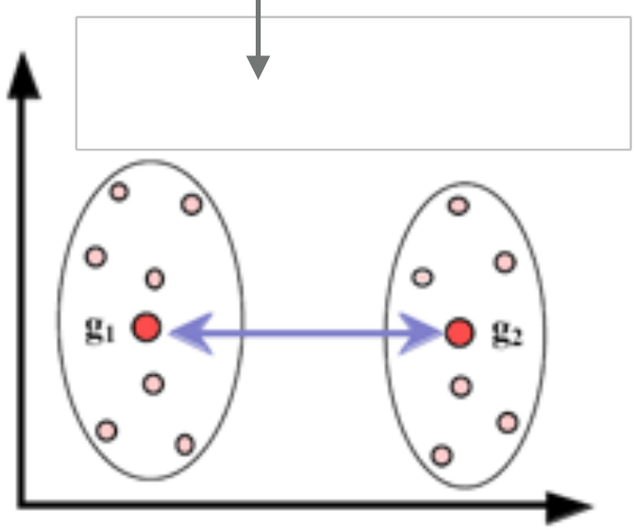
$$J_b = \sum_{k=1}^{K} \sum_{l=k+1}^{K} d^2(\mu_k, \mu_l)$$

THE IDEAL CLUSTERING GIVEN SHAPE QUALITIES



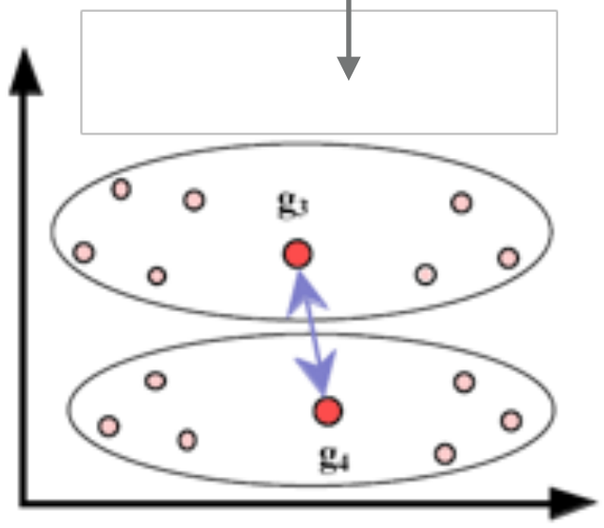
Large *inertia between clusters*

Low *inertia within clusters*



Low *inertia between clusters*

Large *inertia within clusters*



Aim: minimize within-cluster inertia and maximize between-cluster inertia

LEARNING OBJECTIVES

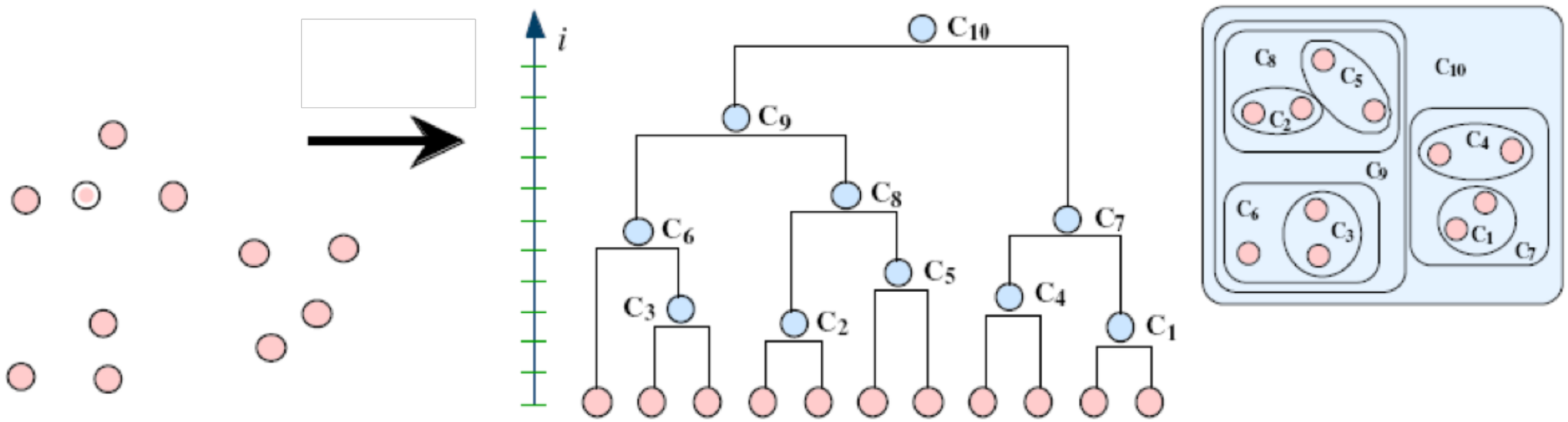
- For what clustering algorithms can be used for ?
- Recall on Distances
- How to evaluate clusterings qualities
- Present different clustering methods
  - **Hierarchical clustering**
  - Partitioning methods (probabilistic or not):
    - K-Means
    - Gaussian Mixture Models
  - Density based :
    - DBSCAN
  - Graph based :
    - Spectral Clustering

HIERARCHICAL CLUSTERING

- Group data over a variety of possible scales in a multi-level hierarchy
- Two construction approaches
  - **Agglomerative** approach (**bottom-up**)
    - Start with each element in its own cluster
    - Iteratively **join** neighboring clusters
  - **Divisive** approach (**top-down**)
    - Start with all the elements in the same cluster
    - iteratively **separate** into smaller clusters

DENDOGRAM

- The result of a hierarchical clustering algorithm are presented in a **dendogram**

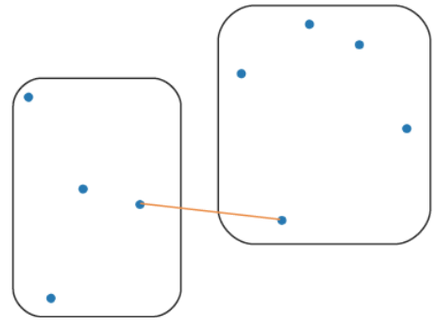


- The height of a cluster in the dendogram depends on the similarity between two clusters : **how to measure it ?**

WHICH METRIC FOR HIERARCHICAL CLUSTERING ?

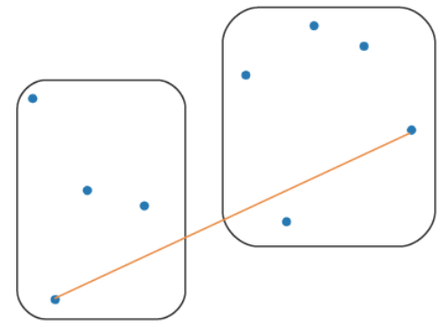
- The distance between two clusters is called an **ultra-metric**
- An ultra-metric is a metric that verifies $d(x,z) \leq \max(d(x,y), d(y,z))$
- **Single** linkage :

$$d(C_1, C_2) = \min_{x \in C_1, y \in C_2} d(x,y)$$



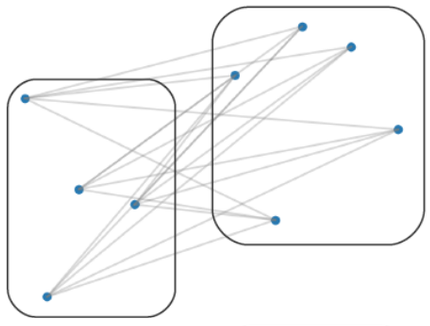
- **Complete** linkage:

$$d(C_1, C_2) = \max_{x \in C_1, y \in C_2} d(x,y)$$



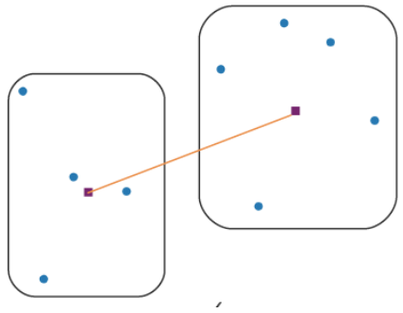
- **Average** linkage:

$$d(C_1, C_2) = \frac{1}{|C_1||C_2|} \sum_{x \in C_1} \sum_{y \in C_2} d(x,y)$$

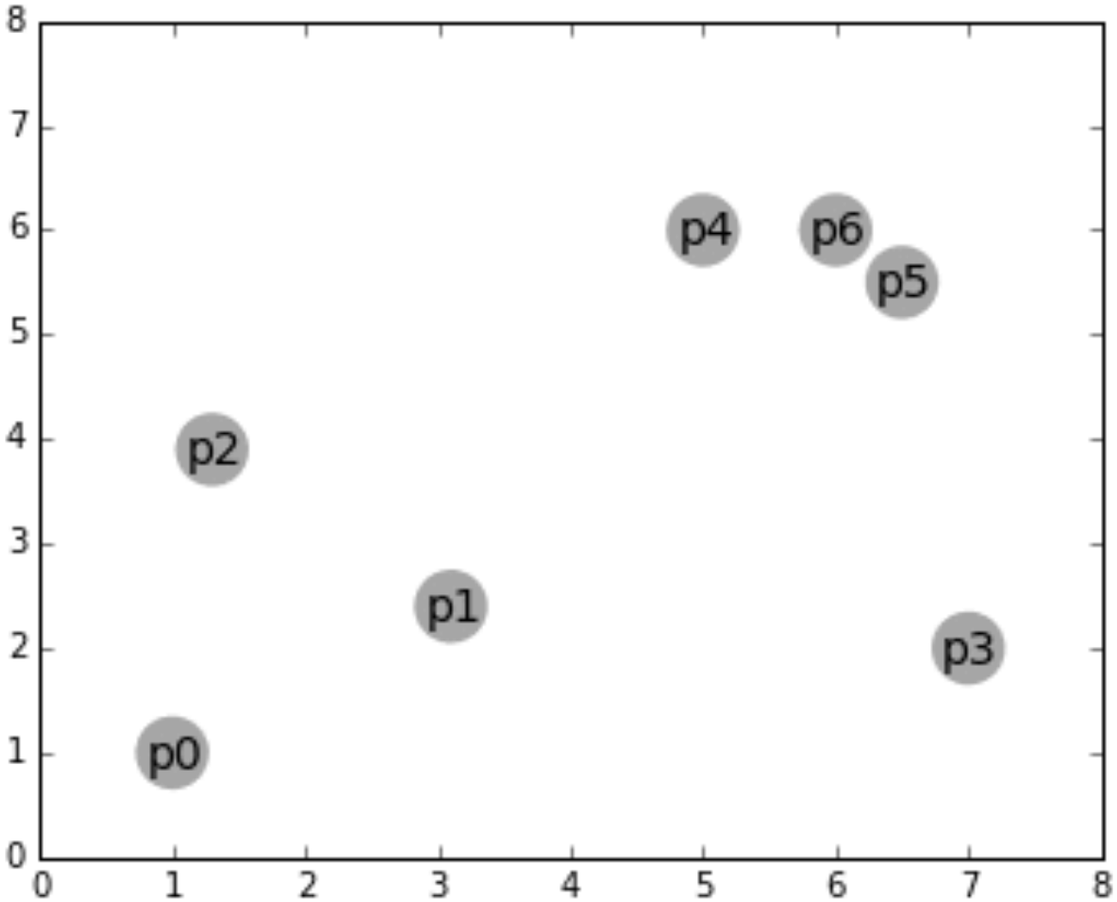


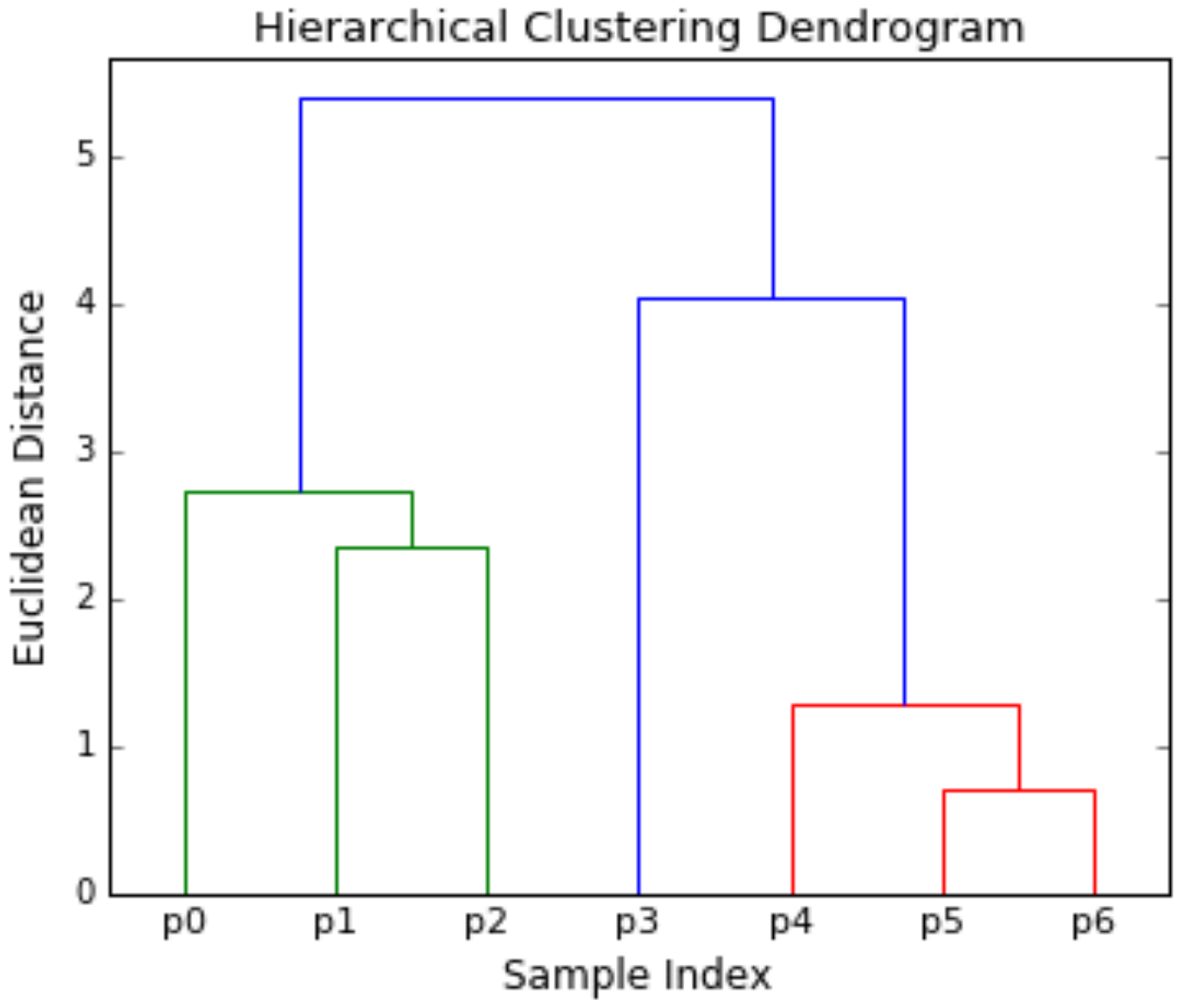
- **Centroid** linkage:

$$d(C_1, C_2) = d(\mu_1, \mu_2)$$



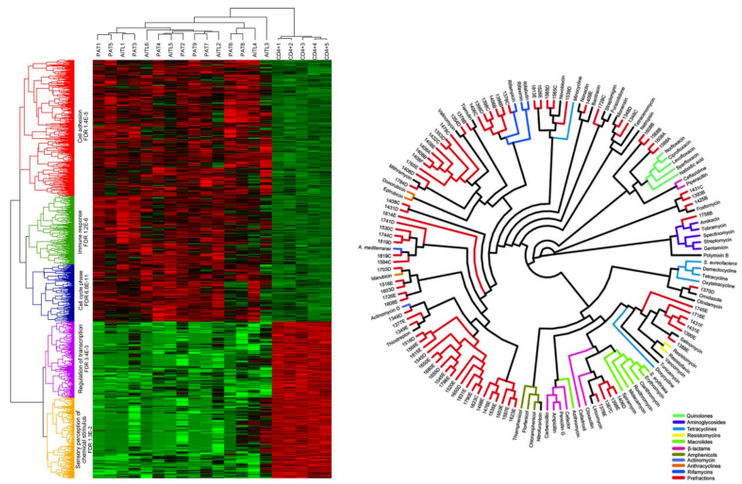
DEMO

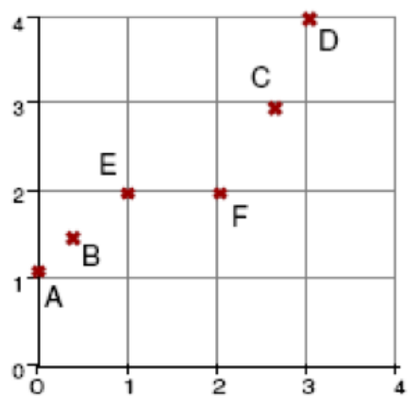

p4
p6
p5
p2
p1
p3
p0
0
1
2
3
4
5
6
7
8


Hierarchical Clustering Dendrogram
Euclidean Distance
Sample Index
0
1
2
3
4
5
p0
p1
p2
p3
p4
p5
p6

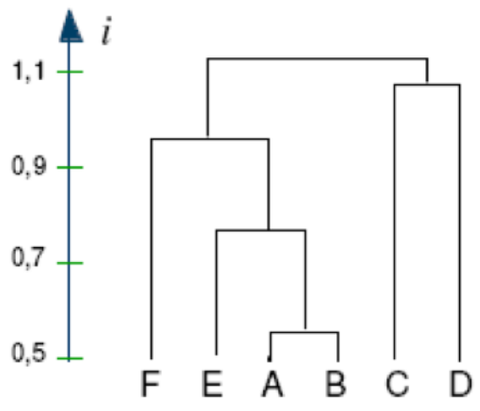
HIERARCHICAL CLUSTERING

- **Advantages**
  - No need to pre-define the number of clusters
  - Interpretability
- **Drawbacks**
  - Computational complexity needed to compute all pairwise distances.
  - Must decide at which level of the hierarchy to split
  - Lack of robustness: results depend on the ultra-metric

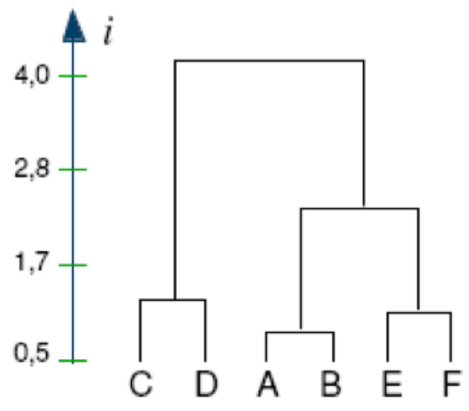




Single linkage



Complete linkage



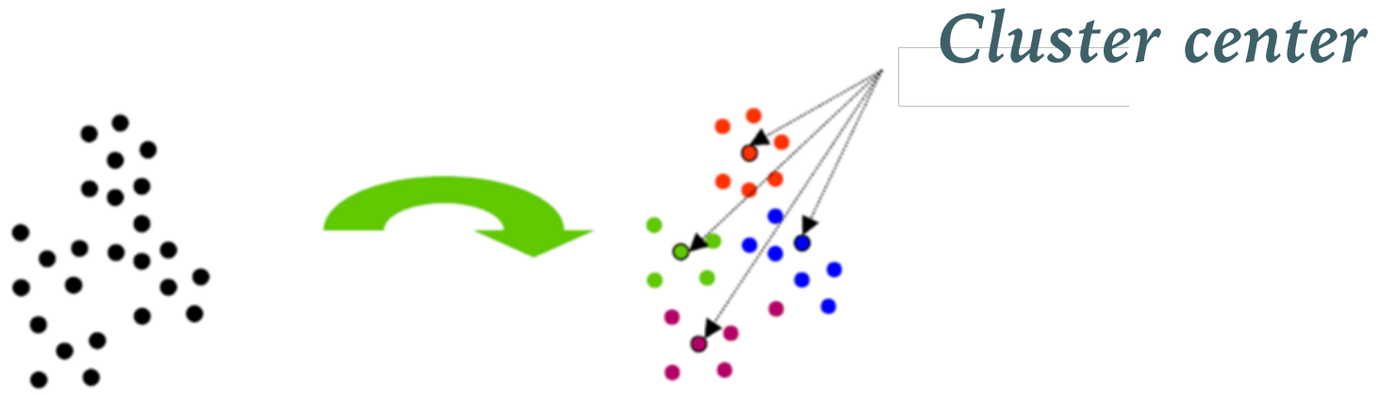
LEARNING OBJECTIVES

- For what clustering algorithms can be used for ?
- Recall on Distances
- How to evaluate clusterings qualities
- Present different clustering methods
  - Hierarchical clustering
  - **Partitioning methods (probabilistic or not):**
    - K-Means
    - Gaussian Mixture Models
  - Density based :
    - DBSCAN
  - Graph based :
    - Spectral Clustering

PARTITIONING METHODS

- **Partitioning**: the objects in the dataset are grouped into K clusters
- Given a value of K, find a partition in K clusters that optimizes a partitioning criterion (based on the obtained clustering quality)

- Typical approach :
  - K-means: each cluster is represented by its center of gravity



LEARNING OBJECTIVES

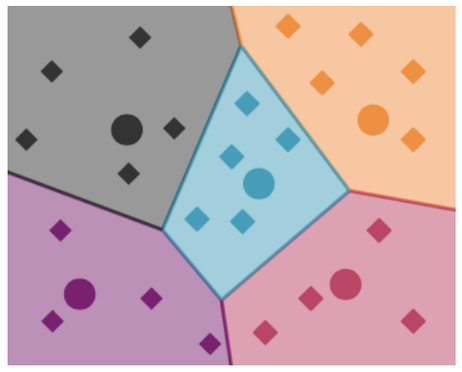
- For what clustering algorithms can be used for ?
- Recall on Distances
- How to evaluate clusterings qualities
- Present different clustering methods
  - Hierarchical clustering
  - Partitioning methods (probabilistic or not):
    - **K-Means**
    - Gaussian Mixture Models
  - Density based :
    - DBSCAN
  - Graph based :
    - Spectral Clustering

K-MEANS CLUSTERING

- Given a set of data $\{\mathbf{x}_1, \cdots, \mathbf{x}_N\}$ where $\mathbf{x}_i \in \mathbb{R}^d$
- **Goal**: Partition the data into K clusters
  - Each cluster contains points close to each other
  - Each cluster is represented by a **prototype** : its center
    - For each cluster, the points in that cluster are those that are closest to its centroid than to any other centroid
- Define an **objective** given this goal
  - Minimize the **within-cluster inertia**

$$\sum_{k=1}^{K} \sum_{x \in C_k} \|\mathbf{x} - \boldsymbol{\mu}_k\|_2^2$$

Voronoi Tesselation

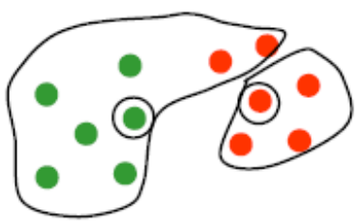


LLOYD'S ALGORITHM

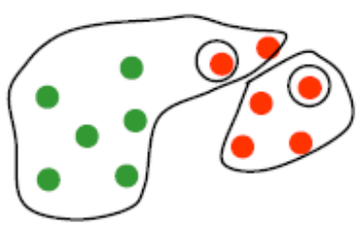
- The objective of K-means is **NP-hard** and difficult to optimize
- A **greedy** strategy is adopted instead

1. Start with two points taken at random

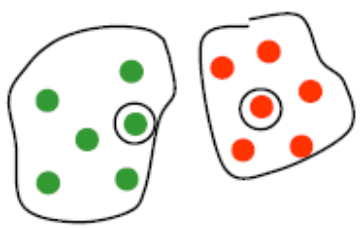
2. Assign each point to the cluster whose centroid it is closest to



3. Compute the centroid of each cluster



4. Repeat until cluster membership converges



K-MEANS ALGORITHM

1. **Initialize** K prototypes to random locations $\boldsymbol{\mu}_1, \cdots, \boldsymbol{\mu}_K$
2. **Repeat** until no change in assignment
   a. **Assign** each example to the closest mean

$$y_i = \arg\min_{k=1}^{K} \|\mathbf{x}_i - \boldsymbol{\mu}_k\|_2$$

   b. **Re-estimate** each mean based only on examples assigned to it

$$\boldsymbol{\mu}_K = \frac{1}{|C_k|} \sum_{y_i = C_k} \mathbf{x}_i$$

K-MEANS CLUSTERING – INTERPRETATION

- Define **binary indicator variables**

$$r_{nk} = \begin{cases} 1 & \text{if data point } \mathbf{x}_n \text{belongs to cluster } k \\ 0 & \text{otherwise} \end{cases}$$

- Define a **distortion measure**

$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \|\mathbf{x}_n - \boldsymbol{\mu}_k\|_2^2$$

- Find the values for $\{r_{nk}\}$ and $\{\boldsymbol{\mu}_k\}$ so as to minimize J

K-MEANS CLUSTERING – INTERPRETATION

- Problem : $\{r_{nk}\}$ depends on $\{\boldsymbol{\mu}_k\}$, $\{\boldsymbol{\mu}_k\}$ depends on $\{r_{nk}\}$
- **Iterate alternate minimization until no further change**

1. Minimize J w.r.t $\{r_{nk}\}$ while keeping $\{\boldsymbol{\mu}_k\}$ fixed

$$r_{nk} = \begin{cases} 1 & \text{if } k = \arg\min_{j=1}^{K} \|\mathbf{x}_n - \boldsymbol{\mu}_j\|_2 \\ 0 & \text{otherwise} \end{cases}$$

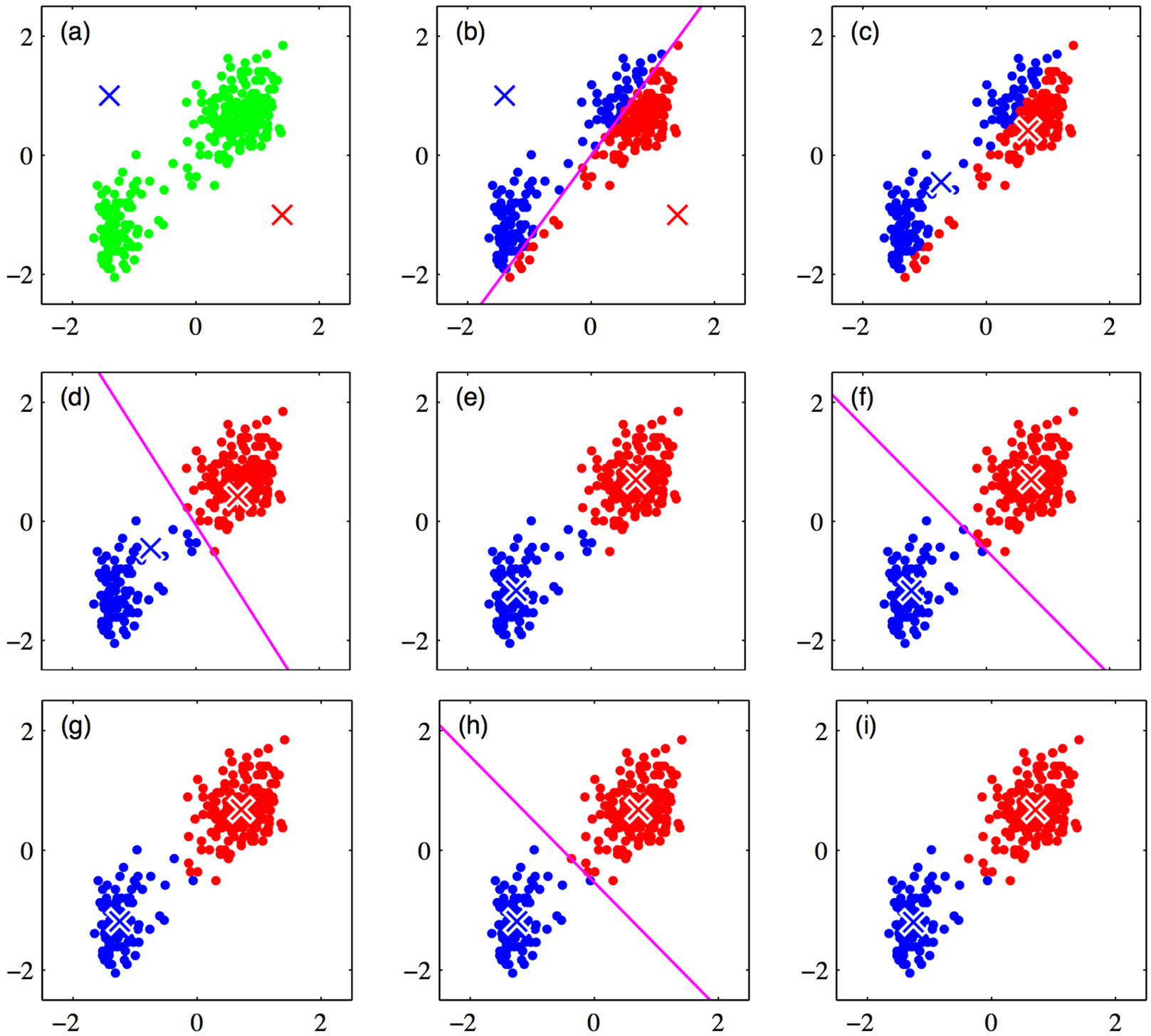
Expectation step

2. Minimize J w.r.t $\{\boldsymbol{\mu}_k\}$ while keeping $\{r_{nk}\}$ fixed

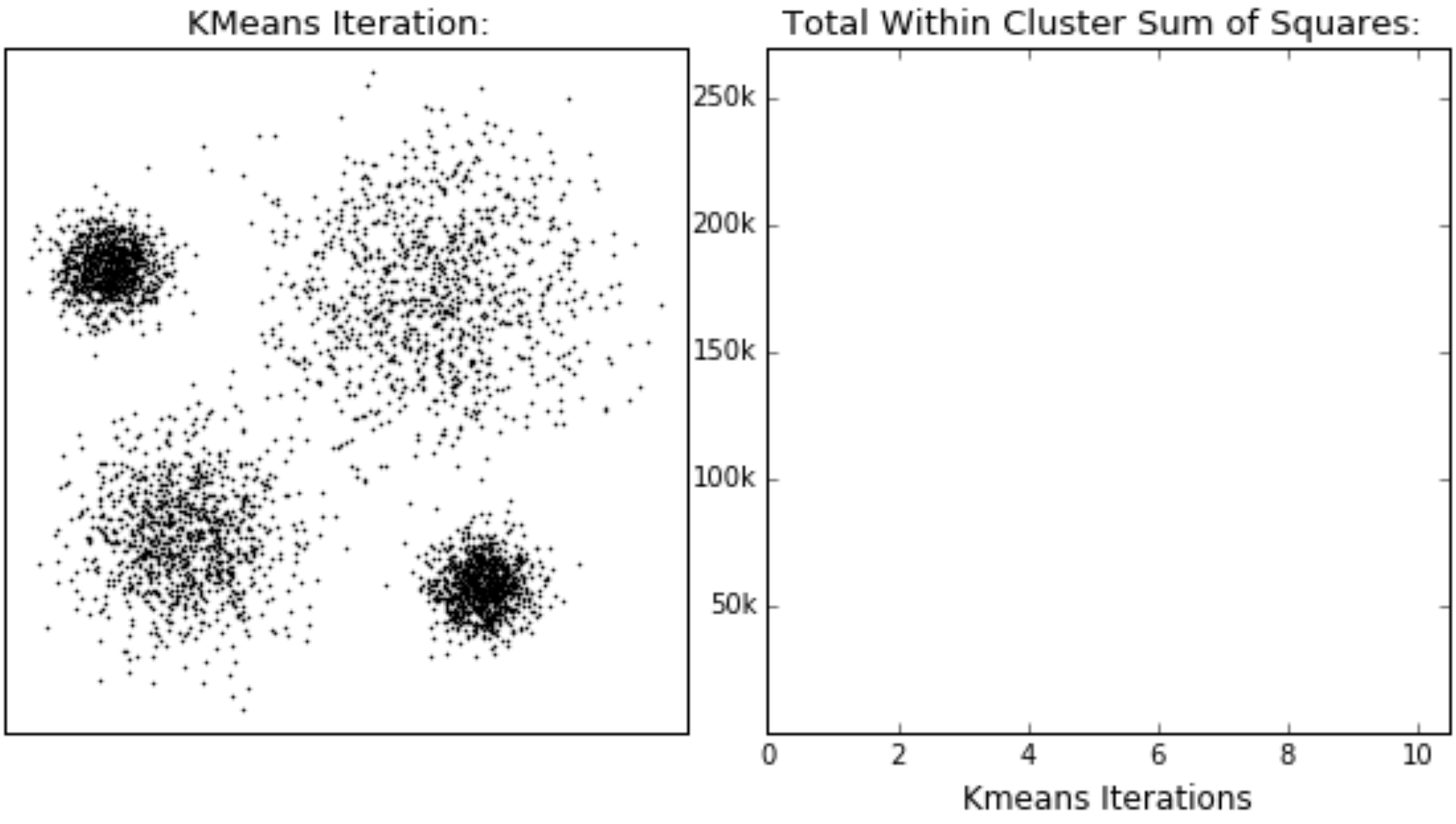
Maximization step

$$\frac{\partial J}{\partial \mu_k} = 2\sum_{n=1}^{N} r_{nk}(\mathbf{x}_n - \boldsymbol{\mu}_k) = 0 \qquad \boldsymbol{\mu}_k = \frac{\sum_{n=1}^{N} r_{nk}\mathbf{x}_n}{\sum_{n=1}^{N} r_{nk}}$$

K-MEANS – EXAMPLE

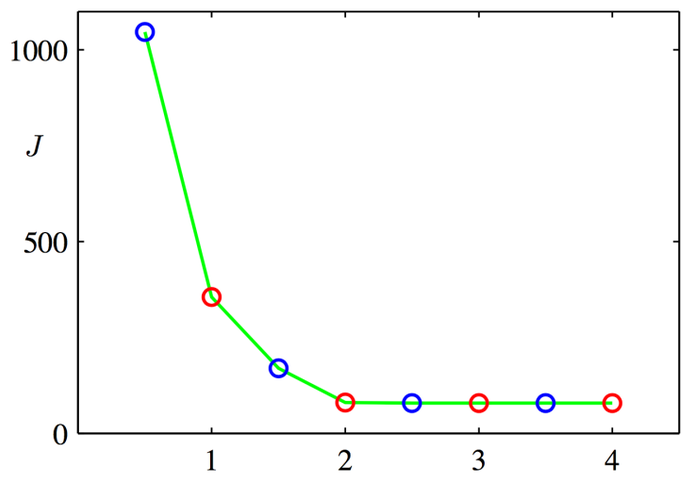


DEMO


KMeans Iteration:
Total Within Cluster Sum of Squares:
250k
200k
150k
100k
50k
0
2
4
6
8
10
Kmeans Iterations

WHICH VALUE OF K ?

- The number of classes K is set by the user
  - Either by using a priori knowledge (K=10 for the clustering of numbers)
  - Either in an empirical way: try different values of K and choose the one that optimizes a quality/validity criterion of the clustering obtained
- **Beware : Error J is not a good indication of quality/validity**
  - Decreases monotonically with K, one can look for the inflection of the curve.



- **Consider J_w / J_b instead**

K-MEANS : PROS AND CONS

- Initialization of the $\{\boldsymbol{\mu}_k\}$
  - Randomly on the range of the attributes of $\mathbf{x}_i \in \mathbb{R}^d$
  - Randomly on the set $\{\mathbf{x}_1, \cdots, \mathbf{x}_N\}$
- Different initializations can lead to different clusters due to **local minima** of J
  - K-means++ : Seeding algorithm to initialize clusters with centroids spread-out through the data
  - K-medoids : take the medoid instead of the mean to represent the cluster
- **K-means is very popular**
  - Efficient with linear computational time
  - Easily implementable

$$O(N \cdot K \cdot d \cdot t)$$

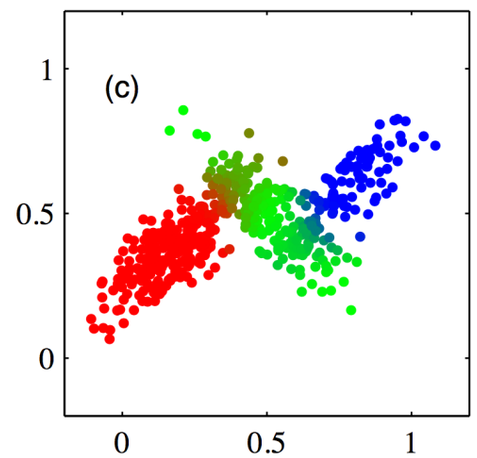
Number of examples — *Number of clusters* — *vectors' dimension* — *Number of iterations*

LEARNING OBJECTIVES

- For what clustering algorithms can be used for ?
- Recall on Distances
- How to evaluate clusterings qualities
- Present different clustering methods
  - Hierarchical clustering
  - Partitioning methods (probabilistic or not):
    - K-Means
    - **Gaussian Mixture Models**
  - Density based :
    - DBSCAN
  - Graph based :
    - Spectral Clustering

FROM K-MEANS TO GAUSSIAN MIXTURE MODELS

- K-means has some limitations
  - it is not robust to outliers
  - the extracted clusters are supposed to be spheric
- We take instead a probabilistic view of the clustering
- The data to cluster will be modeled as a mixture of probability distributions
- The aim is to determine the membership probability of each example to all the clusters
- Clusters are necessarily equally likely



FROM K-MEANS TO EM

- K-means guesses initial clusters then iteratively
  - Assigns each example to a closest cluster
  - Recomputes the cluster centroids
- This actually optimizes the parameters of K spherical Gaussians
  - The centroids represent the means of the K Gaussians
  - The covariance matrix is assumed to have the form $\sigma^2\mathbf{I}$
- The minimized criterion (sum of Euclidean distances to the centroids) is similar to estimate the maximum likelihood
- This is a special case of Expectation Maximisation (EM)

THE EM ALGORITHM

- Method for estimating model parameters given incomplete data (we do not have the class labels)
- Two steps
  - **E-step** : Guess the missing data assuming current expectation (the parameters)
    - This corresponds to **estimate the membership probabilities** of each example to a cluster
  - **M-step** : Compute new parameters that **maximize the likelihood** of the completed data
    - This corresponds to use the estimated membership probabilities to maximize the probability that the distribution parameters fit the data
- EM can be used to estimate the membership probabilities of the examples to the clusters and the parameters of the clusters distributions

MIXTURE OF DISTRIBUTIONS

- We assume that the probability density is given by the following mixture:

$$f(\mathbf{x}) = p(\mathbf{x}|\theta) = \sum_{i=1}^{K} P(\mathbf{x}|y_i, \theta_i) P(y_i)$$

- K: the number of clusters
- θ_i : the parameters of the distributions
- $P(\mathbf{x}|y_i, \theta_i)$: the conditional probability that *x* was generated from cluster y_i
- $P(y_i)$: the a-priori probability that *x* was generated from cluster y_i , its is denoted by π_i with $\sum_{i=1}^{K} \pi_i = 1$

MIXTURE OF GAUSSIANS

- We focus on the mixture of **Gaussians**:

$$f(\mathbf{x}) = \sum_{i=1}^{K} \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)\pi_i$$

$$\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) = \frac{1}{(2\pi)^{d/2}|\boldsymbol{\Sigma}_i|^{1/2}} e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i^{-1}(\mathbf{x}-\boldsymbol{\mu}_i)}$$

$$P(\mathbf{x}|y_i, \theta_i) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$$

- d is the dimensionality of the data
- **Why this distribution** ? has an easy form to work with the log-likelihood

MAXIMUM LIKELIHOOD

- We assume that $\mathcal{X} = \{\mathbf{x}_1, \cdots, \mathbf{x}_N\}$ are i.i.d non-labelled samples generated from $f(\mathbf{x})$
- The goal is to find the parameters $\theta = \{\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i, \pi_i\}_{i=1}^{K}$ that best describe the data
- We will maximize the likelihood of the data, i.e., the probability of the observed data given the parameters:

$$p(\mathcal{X}|\theta) = \prod_{j=1}^{N} p(\mathbf{x}_j|\theta)$$

- We will focus on maximizing the log-likelihood:

$$\log(p(\mathcal{X}|\theta)) = \sum_{j=1}^{N} \log(p(\mathbf{x}_j|\theta)) = \sum_{j=1}^{N} \log \sum_{i=1}^{K} \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)\pi_i$$

EM TO MAXIMIZE THE LOG-LIKELIHOOD

- We need an algorithm to maximize the log-likelihood on the data with $\theta = \{\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i, \pi_i\}_{i=1}^{K}$ the parameters of the Gaussian

$$\hat{\theta} = \arg\max \log(p(\mathcal{X}|\theta)) = \arg\max \sum_{j=1}^{N} \log \sum_{i=1}^{K} \mathcal{N}(\mathbf{x}|\theta_i)\pi_i$$

- **We can use the EM algorithm to do this !**
- We denote the membership of an example $\mathbf{x}_i$ to a cluster k by δ_{ik}
- These variables are hidden (i.e., not observed) but we can estimate them from the data : **E-step**
- Once the expected values γ_{ik} of the δ_{ik} are known, we can find θ that maximize the likelihood of the completed data : **M-step**

EXPECTED MEMBERSHIPS TO CLUSTERS

- With the Bayes rule, we can obtain the expected membership of an example x_i to a cluster k

$$\gamma_{ik} = p(\delta_{ik} = 1|\mathbf{x}_i) = \frac{p(\delta_{ik} = 1)p(\mathbf{x_i}|\delta_{ik} = 1)}{\sum_{j=1}^{K} p(\delta_{jk} = 1)p(\mathbf{x_j}|\delta_{jk} = 1)}$$

$$= \frac{\pi_k \mathcal{N}(\mathbf{x_i}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x_i}|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}$$

MAXIMIZE THE LIKELIHOOD

- We seek the maximum of the log-likelihood

$$\log(p(\mathcal{X}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})) = \sum_{j=1}^{N} \log \sum_{i=1}^{K} \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)\pi_i$$

- We set the derivative to 0 with respect to $\boldsymbol{\mu}_k$ and obtain

$$0 = \sum_{n=1}^{N} \frac{\pi_k \mathcal{N}(\mathbf{x_n}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x_n}|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} \boldsymbol{\Sigma}_k^{-1}(\mathbf{x}_n - \boldsymbol{\mu}_k)$$

MAXIMIZE THE LIKELIHOOD

- The solutions are then given by

$$\boldsymbol{\mu}_k = \frac{\sum_{n=1}^{N} \gamma_{nk} \mathbf{x}_n}{\sum_{n=1}^{N} \gamma_{nk}} \qquad \pi_k = \frac{\sum_{n=1}^{N} \gamma_{nk}}{N}$$

$$\boldsymbol{\Sigma}_k = \frac{\sum_{n=1}^{N} \gamma_{nk} (\mathbf{x}_n - \boldsymbol{\mu}_k)(\mathbf{x}_n - \boldsymbol{\mu}_k)^T}{\sum_{n=1}^{N} \gamma_{nk}}$$

THE FINAL EM ALGORITHM FOR GMM

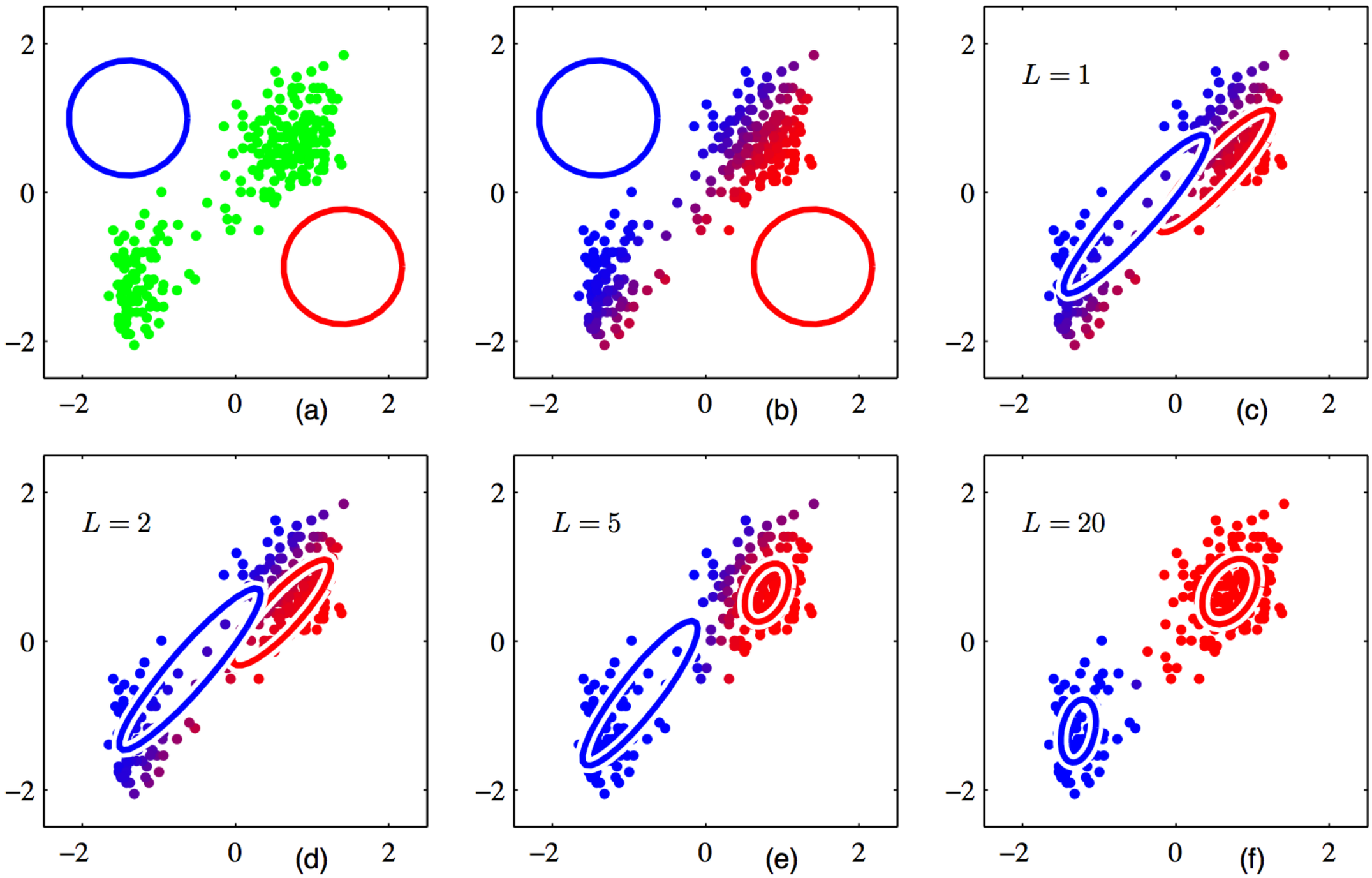
1. Initialize the means $\boldsymbol{\mu}_k$, covariances $\boldsymbol{\Sigma}_k$, and mixing coefficients $\boldsymbol{\pi}_k$
2. **E-Step** : evaluate the memberships using the current parameters

$$\gamma_{ik} = \frac{\pi_k \mathcal{N}(\mathbf{x_i}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x_i}|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}$$

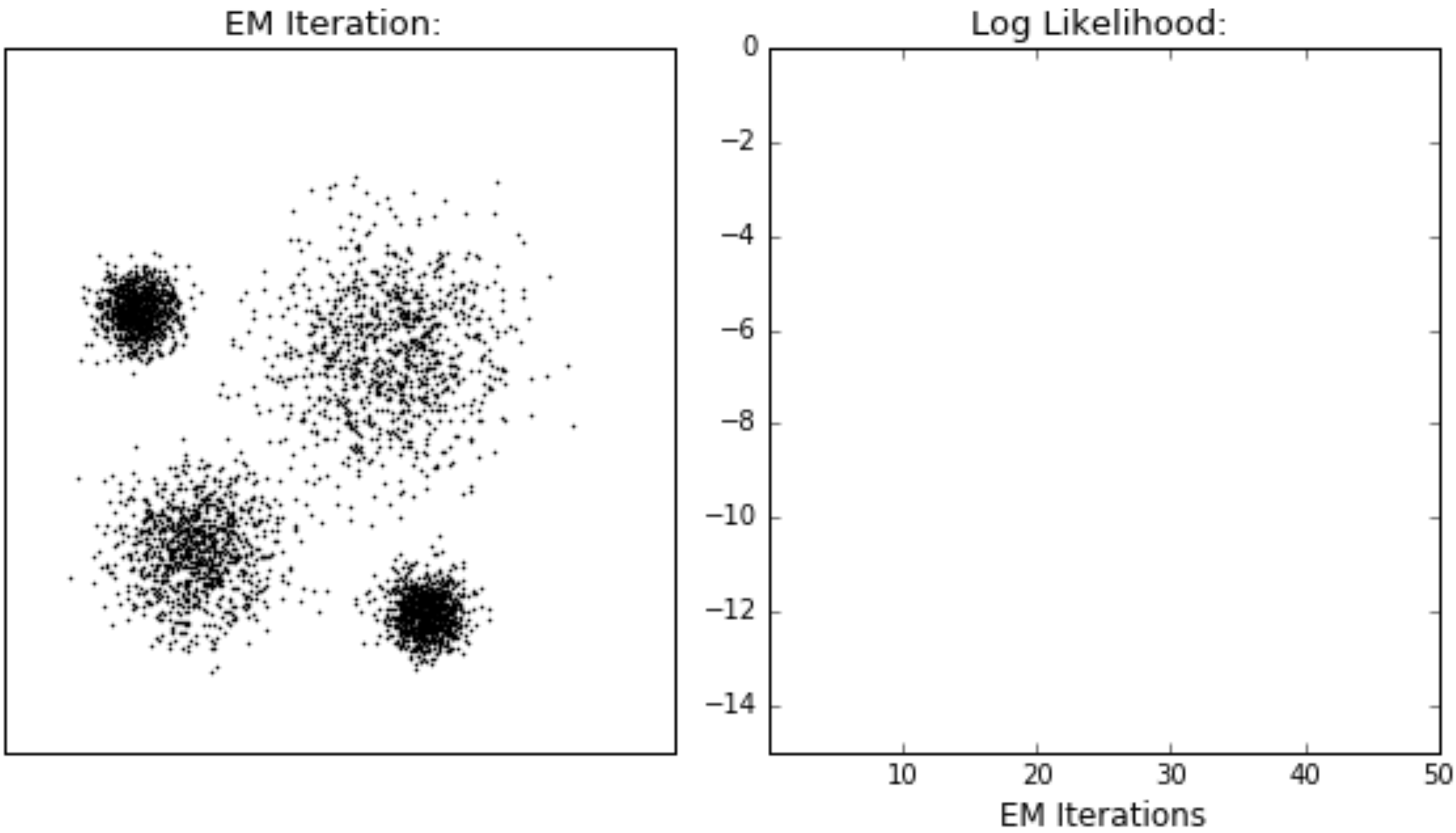
3. **M-step :** Re-estimate the parameters using the current memberships

$$\boldsymbol{\mu}_k = \frac{\sum_{n=1}^{N} \gamma_{nk} \mathbf{x}_n}{\sum_{n=1}^{N} \gamma_{nk}} \qquad \boldsymbol{\Sigma}_k = \frac{\sum_{n=1}^{N} \gamma_{nk} (\mathbf{x}_n - \boldsymbol{\mu}_k)(\mathbf{x}_n - \boldsymbol{\mu}_k)^T}{\sum_{n=1}^{N} \gamma_{nk}} \qquad \pi_k = \frac{\sum_{n=1}^{N} \gamma_{nk}}{N}$$

EXAMPLE OF GMM



DEMO


EM Iteration:
Log Likelihood:
0
−2
−4
−6
−8
−10
−12
−14
10
20
30
40
50
EM Iterations

RELATION TO K-MEANS

- If we set covariance matrices to $\epsilon \mathbf{I}$ where ϵ is shared by all the components, we have $p(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) = \frac{1}{(2\pi)^{d/2}} e^{-\frac{1}{2\epsilon}\|\mathbf{x}-\boldsymbol{\mu}_k\|^2}$
- Then the memberships are $\gamma_{ik} = \frac{\pi_k e^{-\frac{1}{2\epsilon}\|\mathbf{x}_n-\boldsymbol{\mu}_k\|^2}}{\sum_{j=1}^{K} \pi_j e^{-\frac{1}{2\epsilon}\|\mathbf{x}_n-\boldsymbol{\mu}_j\|^2}}$
- Taking the limit $\epsilon \to 0$ we can show that $\lim_{\epsilon \to 0} \gamma_{nk} = r_{nk}$
- A hard assignment to only one cluster is obtained
- One can show, that as $\epsilon \to 0$

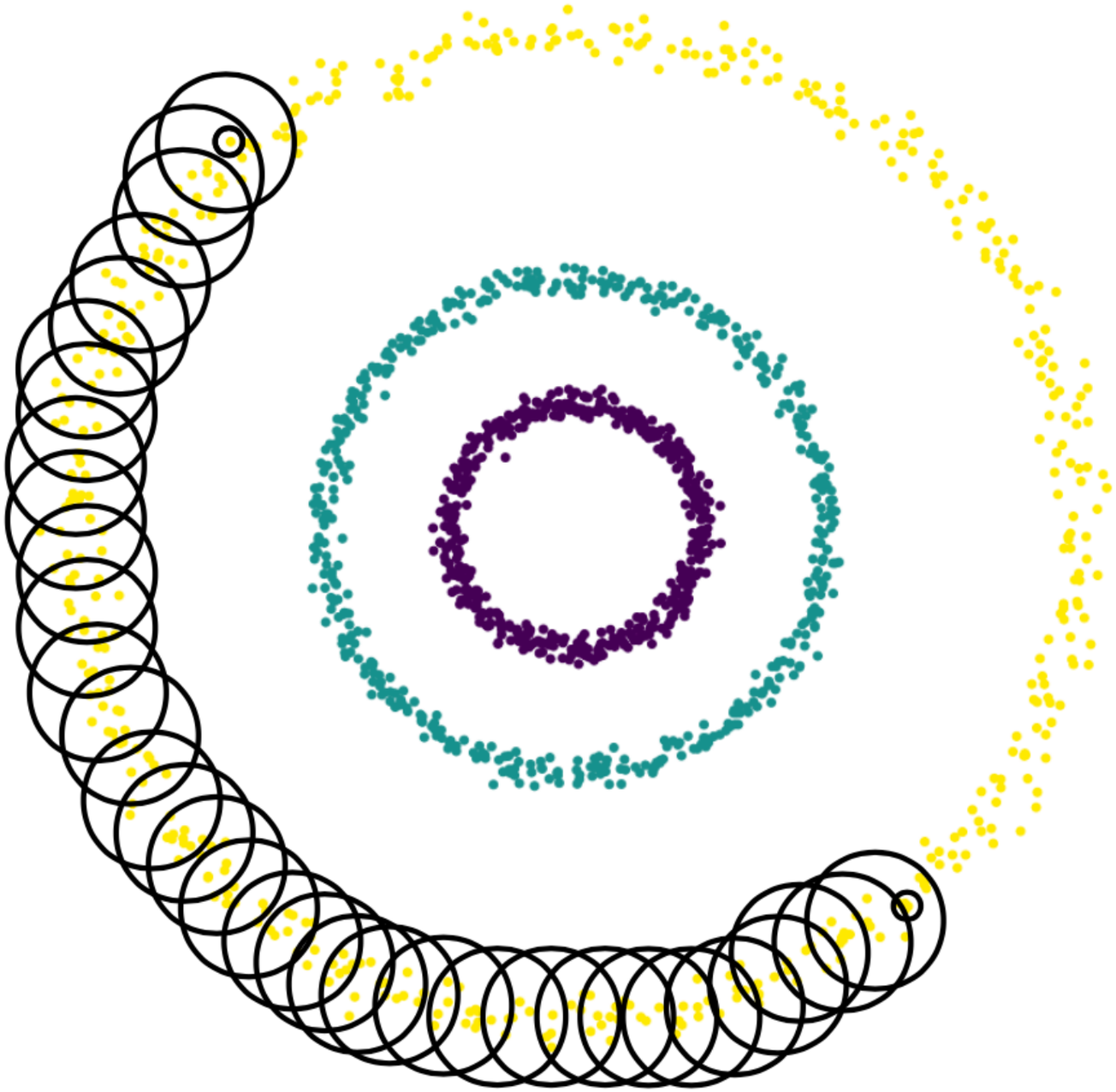
$$E_\gamma[\log\, p(\boldsymbol{\mathcal{X}}, \boldsymbol{\gamma}|\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\pi})] \to -\frac{1}{2}\sum_{n=1}^{N}\sum_{k=1}^{K} r_{nk}\|\mathbf{x}_n - \boldsymbol{\mu}_k\|_2^2 + const$$

LEARNING OBJECTIVES

- For what clustering algorithms can be used for ?
- Recall on Distances
- How to evaluate clusterings qualities
- Present different clustering methods
  - Hierarchical clustering
  - Partitioning methods (probabilistic or not):
    - K-Means
    - Gaussian Mixture Models
  - **Density based :**
    - DBSCAN
  - Graph based :
    - Spectral Clustering

DENSITY BASED METHODS

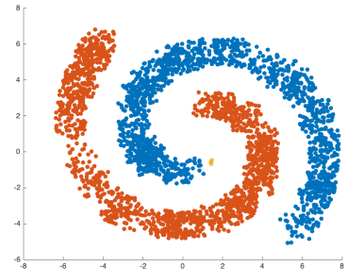
- clusters are made of dense neighborhoods of points



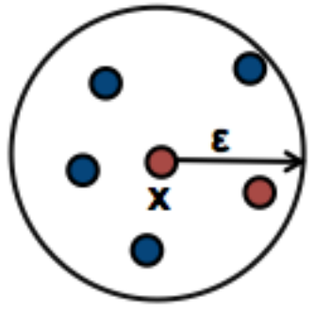
LEARNING OBJECTIVES

- For what clustering algorithms can be used for ?
- Recall on Distances
- How to evaluate clusterings qualities
- Present different clustering methods
  - Hierarchical clustering
  - Partitioning methods (probabilistic or not):
    - K-Means
    - Gaussian Mixture Models
  - Density based :
    - **DBSCAN**
  - Graph based :
    - Spectral Clustering

DBSCAN : DENSITY BASED SPATIAL CLUSTERING ALGORITHM WITH NOISE



- DBSCAN discovers clusters of arbitrary shapes
- Introduces a density-based notion of a cluster
  - **A cluster is defined as a maximal set of density-connected points**
- Two parameters:
  - **Epsilon** : Maximum radius of the neighborhood
  - **MinPts** : minimum number of points in the Epsilon-neighborhood



- The **Epsilon-neighborhood** of a point p:

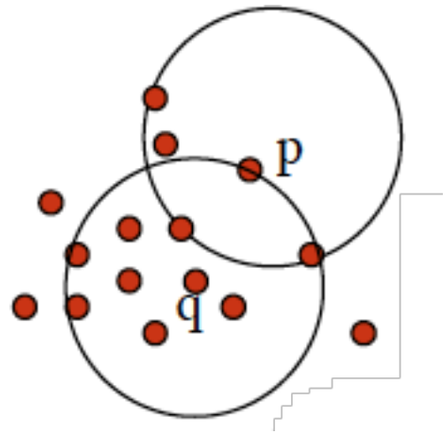
$$N_\epsilon(p) = \{q | d(p,q) \leq \epsilon\}$$

- **Core points** :

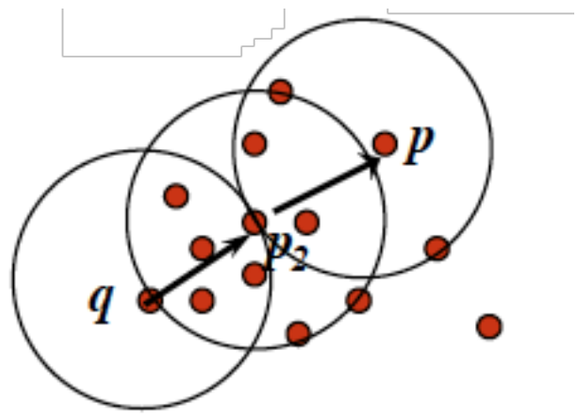
$$p : |N_\epsilon(p)| \geq \text{MinPts}$$

DENSITY DEFINITIONS

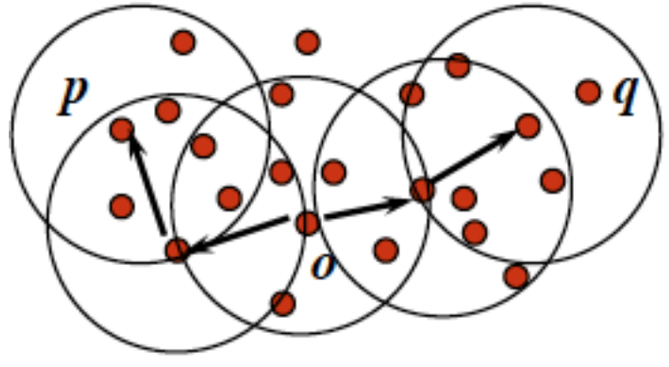
- **Directly density-reachable**
  - A point p is directly density-reachable from a point q if $p \in N_\epsilon(q)$ and $N_\epsilon(q)| \geq \text{MinPts}$



- **Density-reachable**
  - A point is density reachable from a point q if there is a chain of points such p_{i+1} is directly density reachable from p_i

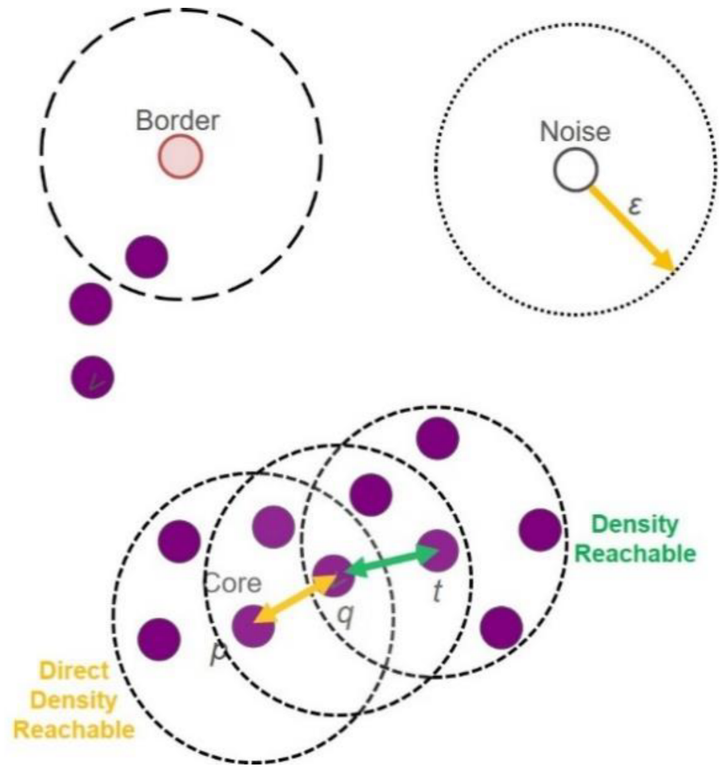


- **Density-connected**
  - A point p is density-connected to a point q if there is a point o such that both p and q are density reachable from o



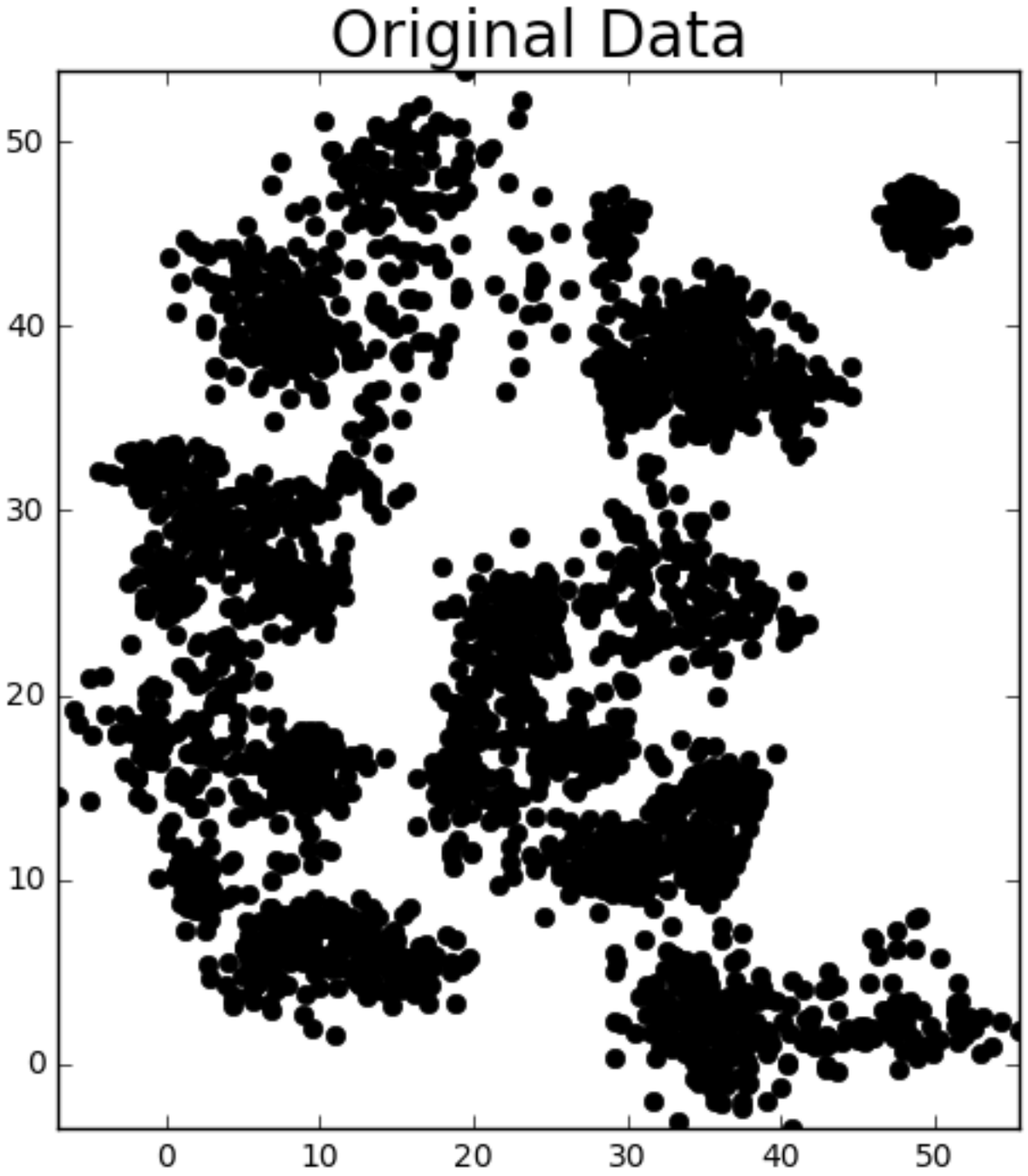
DBSCAN ALGORITHM

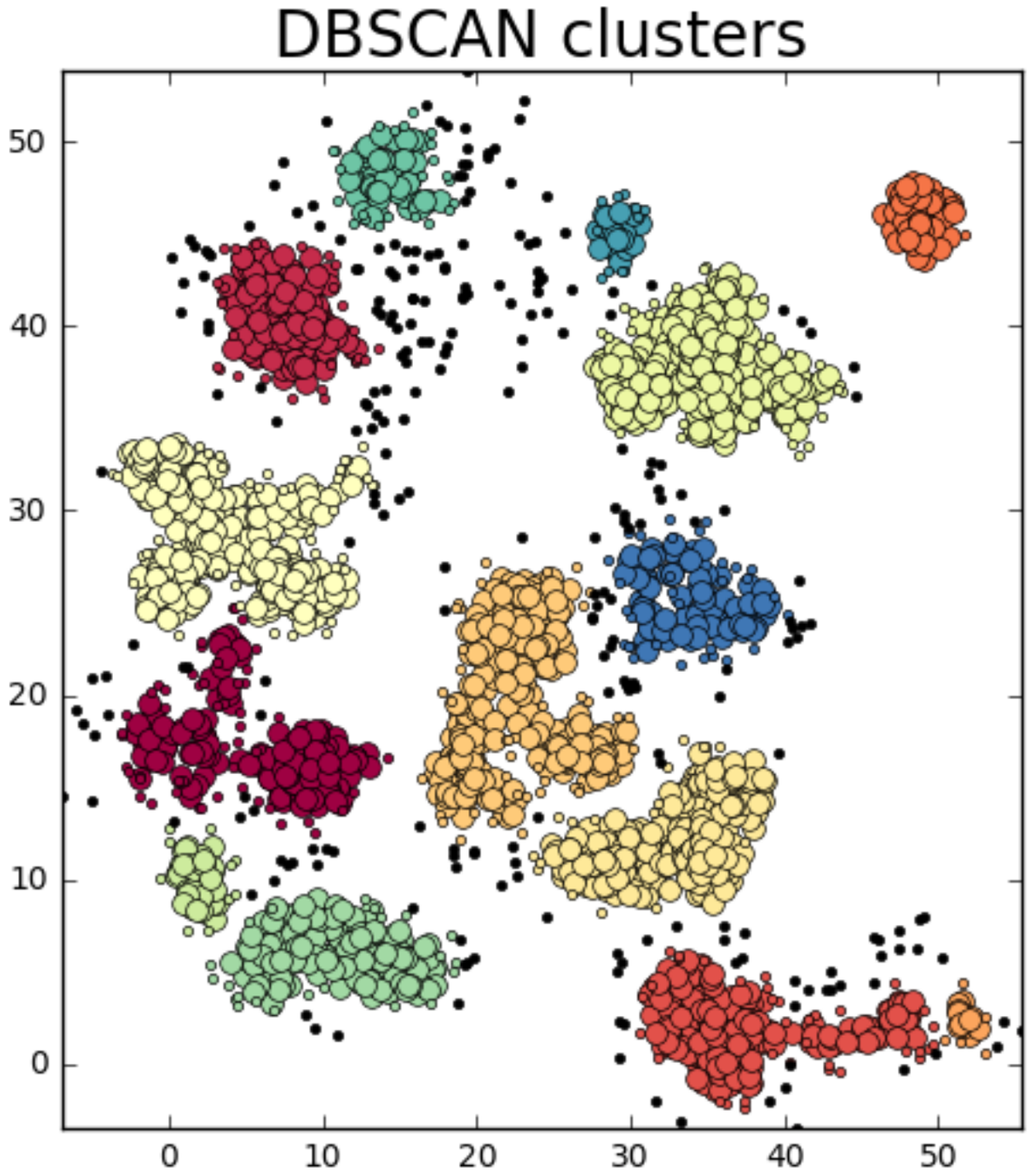
- Randomly select a point p
- Retrieve all points density-reachable from p
  - if p is a core point, a cluster is created
  - if p is a border point or an outlier, no points are directly reachable from p and another point is selected
- Continue until all the points have been visited



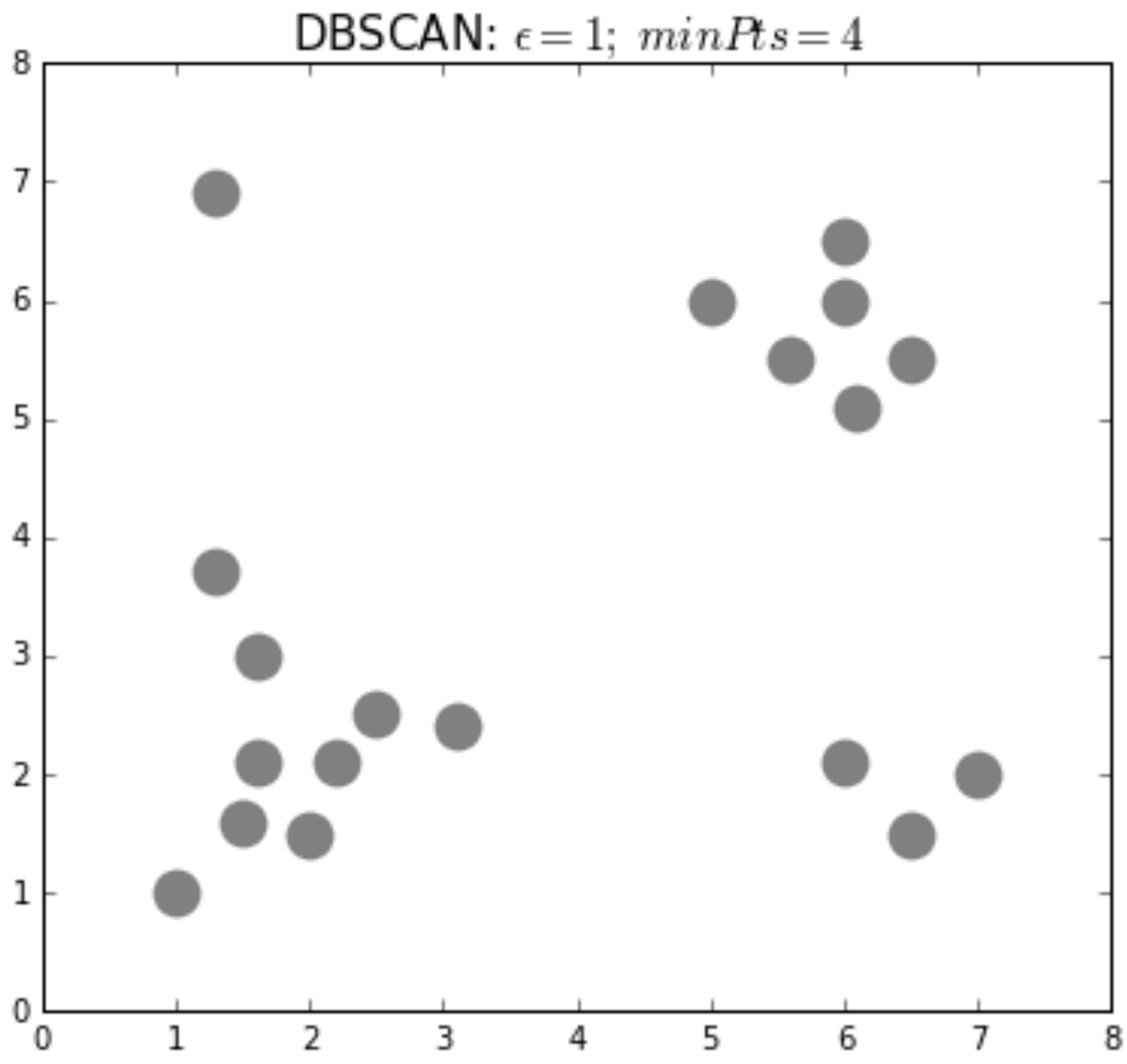
Complexity : O(N log N) with efficient data structures

EXAMPLE


Original Data
50
40
30
20
10
0
0
10
20
30
40
50


DBSCAN clusters
50
40
30
20
10
0
0
10
20
30
40
50

DEMO

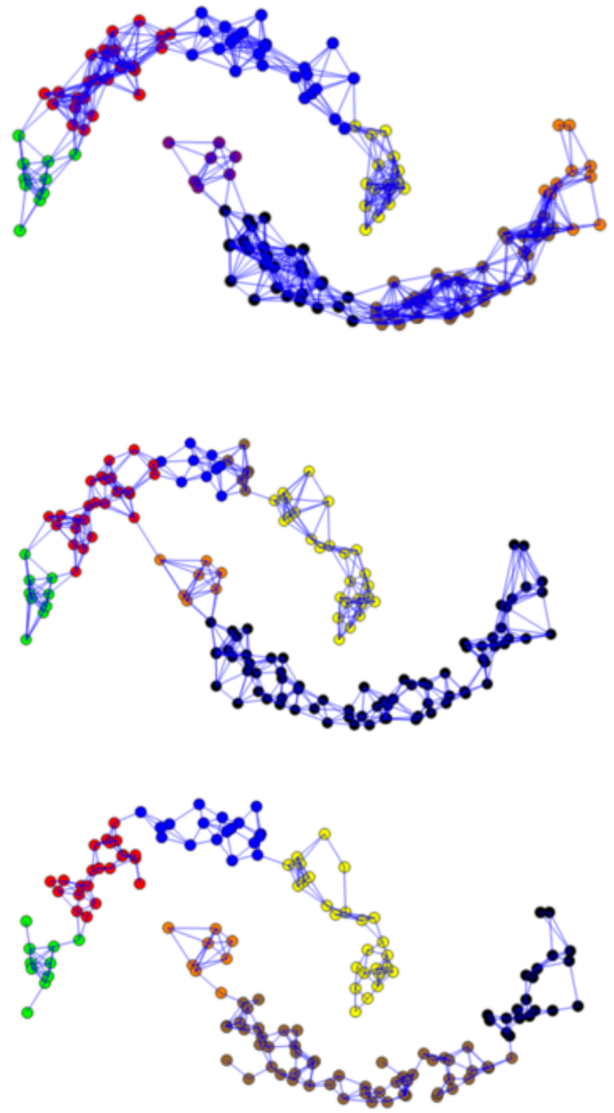


LEARNING OBJECTIVES

- For what clustering algorithms can be used for ?
- Recall on Distances
- How to evaluate clusterings qualities
- Present different clustering methods
  - Hierarchical clustering
  - Partitioning methods (probabilistic or not):
    - K-Means
    - Gaussian Mixture Models
  - Density based :
    - DBSCAN
  - **Graph based :**
    - Spectral Clustering

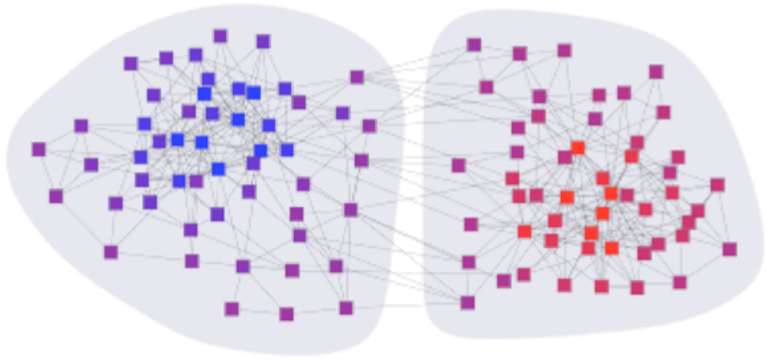
CONSTRUCTION OF GRAPH FROM A DATASET

- Given a data set $\mathcal{X} = \{\mathbf{x}_1, \cdots, \mathbf{x}_N\}$ and a distance measure, construct a graph and weight the edges with a similarity function
- Two popular choices :
  - Epsilon-neighborhood : Given ε all points at a distance less than ε are connected
  - K Nearest Neighbor : connect each vertices to its k nearest neighbors
  - Mutual k-nn : connect vertices v_i and v_j if v_j is among the k-nn of v_i and v_i among the k-nn of v_j



GRAPH-BASED CLUSTERING

- Clustering partitions the vertices of the graph
- A good clustering places dissimilar vertices in different partitions
- The aim of the clustering is to find a cut in the graph



- Graph-based clustering :
  - Define the cut criterion
  - Optimize that criterion

LEARNING OBJECTIVES

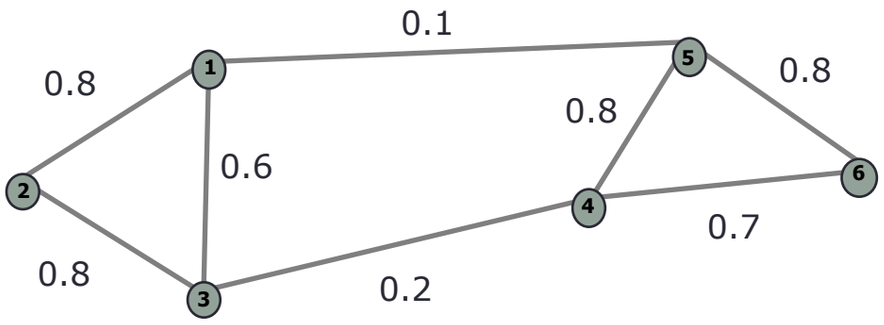
- For what clustering algorithms can be used for ?
- Recall on Distances
- How to evaluate clusterings qualities
- Present different clustering methods
  - Hierarchical clustering
  - Partitioning methods (probabilistic or not):
    - K-Means
    - Gaussian Mixture Models
  - Density based :
    - DBSCAN
  - Graph based :
    - **Spectral Clustering**

SPECTRAL CLUSTERING

- Spectral clustering is a clustering algorithm that
  - Treats clustering as a graph partitioning problem without any assumptions about cluster shapes (can be non-convex)
  - Cluster points using **matrix** eigenvectors **extracted from the data**
  - Projects data into a small space where they can be more easily separated
- **Advantages**:
  - No assumptions about cluster shapes
  - Easy to implement
  - Fast for sparse graphs or small datasets
- **Disadvantages**:
  - Sensitive to the choice of parameters
  - Time-consuming calculation for large amounts of data

ADJACENCY MATRIX A

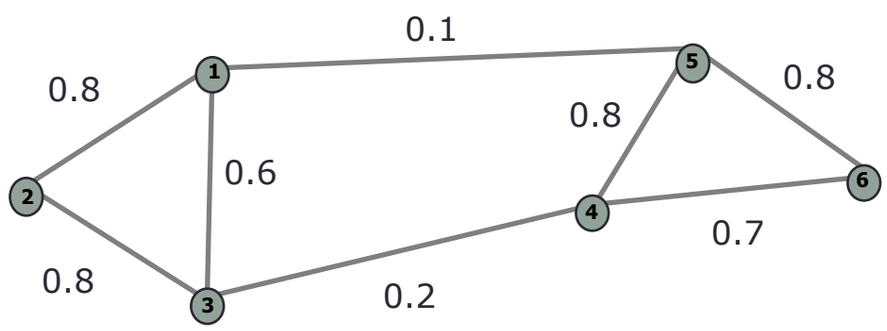
- The adjacency matrix $\boldsymbol{A}$ of an undirected graph
  - Is a matrix of size $N \times N$
  - The rows and columns represent the nodes of the graph and the entries represent the edges.
- $A(i,j) = 0$: nodes v_i and v_j are not connected
- $A(i,j) = 1$: nodes v_i and v_j are connected



	x_1	x_2	x_3	x_4	x_5	x_6
x_1	0	1	1	0	1	0
x_2	1	0	1	0	0	0
x_3	1	1	0	1	0	0
x_4	0	0	1	0	1	1
x_5	1	0	0	1	0	1
x_6	0	0	0	1	1	0

SIMILARITY MATRIX W

- The similarity matrix $\boldsymbol{W}$ of an undirected graph
  - Is a symmetrical matrix of size $N \times N$
  - Is a weighted adjacency matrix
  - Each edge is weighted by a measure of similarity between the vectors.
- $W(i,j) = s(v_i, v_j)$ if nodes v_i and v_j are connected
- $W(i,j) = 0$ if nodes v_i and v_j are not connected

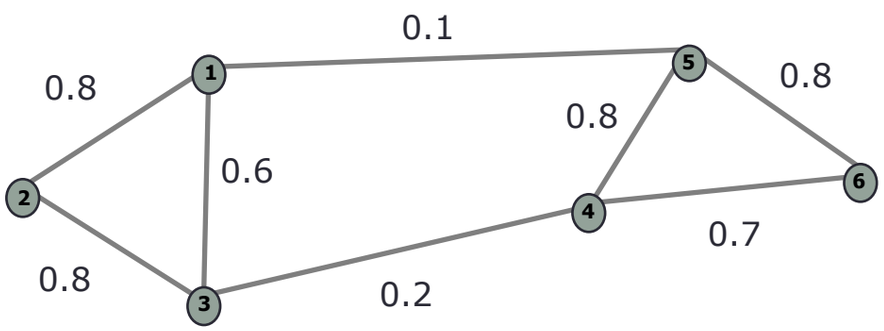


	x_1	x_2	x_3	x_4	x_5	x_6
x_1	0	0.8	0.6	0	0.1	0
x_2	0.8	0	0.8	0	0	0
x_3	0.6	0.8	0	0.2	0	0
x_4	0	0	0.2	0	0.8	0.7
x_5	0.1	0	0	0.8	0	0.8
x_6	0	0	0	0.7	0.8	0

DEGREE MATRIX D

- The matrix of degrees ***D*** of an undirected graph
  - Is a diagonal matrix of size *NxN*
  - The degree of a node represents the total weight of the edges incident to the node:

$$D(i,i) = \sum_j w(i,j)$$



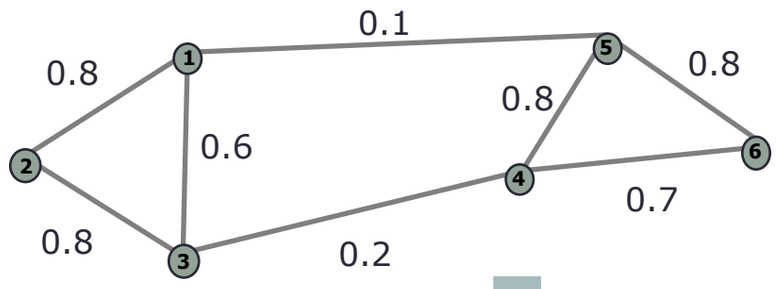
	x_1	x_2	x_3	x_4	x_5	x_6
x_1	1.5	0	0	0	0	0
x_2	0	1.6	0	0	0	0
x_3	0	0	1.6	0	0	0
x_4	0	0	0	1.7	0	0
x_5	0	0	0	0	1.7	0
x_6	0	0	0	0	0	1.5

UNNORMALIZED LAPLACIAN

- The unnormalized Laplacian matrix
  - Is a symmetrical matrix of size $N \times N$

$$L(i,j) = \begin{cases} D(i,j) & \text{if } i = j \\ -W(i,j) & \text{if } i \sim j \\ 0 & \text{otherwise} \end{cases}$$

$$\mathbf{L} = \mathbf{D} - \mathbf{W}$$



	x_1	x_2	x_3	x_4	x_5	x_6
x_1	1.5	-0.8	-0.6	0	-0.1	0
x_2	-0.8	1.6	-0.8	0	0	0
x_3	-0.6	-0.8	1.6	-0.2	0	0
x_4	0	0	-0.2	1.7	-0.8	-0.7
x_5	-0.1	0	0	-0.8	1.7	-0.8
x_6	0	0	0	-0.7	-0.8	1.5

- *Properties:*
  - L is positive semi-definite
  - Recall: if $\mathbf{Lf} = \lambda\mathbf{f}$ then λ is an eigenvalue of the Laplacian
  - All eigenvalues are non-negative reals $0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_N$
  - The first eigenvector is $\mathbf{1}$
  - The multiplicity k of the first eigenvector gives the number of connected components of the graph

L IS SPD

- Consider a graph function $\mathbf{f} \in \mathbb{R}^N$ that assigns values to vertices : $\mathbf{f} : \mathcal{V} \to \mathbb{R}$
- $$\mathbf{f}^T\mathbf{L}\mathbf{f} = \mathbf{f}^T\mathbf{D}\mathbf{f} - \mathbf{f}^T\mathbf{W}\mathbf{f} = \sum_{i=1}^{N} d_i \mathbf{f}_i^2 - \sum_{i,j} \mathbf{f}_i \mathbf{f}_j w_{ij}$$

$$= \frac{1}{2}\left(\sum_{i=1}^{N} d_i \mathbf{f}_i^2 - 2\sum_{i,j} \mathbf{f}_i \mathbf{f}_j w_{ij} + \sum_{j=1}^{N} d_j \mathbf{f}_j^2\right)$$

$$= \frac{1}{2}\left(\sum_{i=1}^{N}\sum_{j=1}^{N} w_{ij} \mathbf{f}_i^2 - 2\sum_{i,j} \mathbf{f}_i \mathbf{f}_j w_{ij} + \sum_{j=1}^{N}\sum_{i=1}^{N} w_{ij} \mathbf{f}_j^2\right)$$

$$= \frac{1}{2}\sum_{i,j} w_{ij}(\mathbf{f}_i - \mathbf{f}_j)^2$$

NORMALIZED LAPLACIAN

- The normalized Laplacian matrix
  - Is a symmetrical matrix of size *NxN*

$$\mathcal{L} = \mathbf{D}^{-1/2} \cdot (\mathbf{D} - \mathbf{W}) \cdot \mathbf{D}^{-1/2} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$$

$$\mathbf{f}^T \mathcal{L} \mathbf{f} = \frac{1}{2} \sum_{i,j} w_{ij} \left(\frac{\mathbf{f}_i}{\sqrt{d_i}} - \frac{\mathbf{f}_j}{\sqrt{d_j}} \right)^2$$

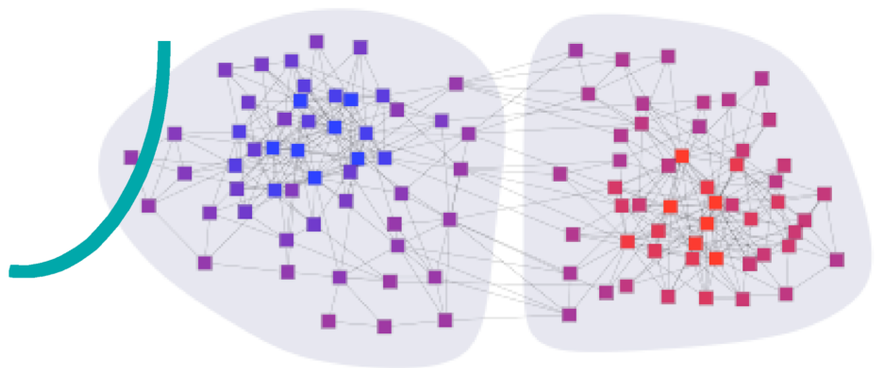
- ***Properties:***
  - $\mathcal{L}$ is positive semi-definite
  - All eigenvalues are non-negative reals $0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_N$
  - The first eigenvector is $\mathbf{D}^{1/2}\mathbf{1}$
  - The multiplicity *k* of the first eigenvector gives the number of connected components of the graph

CLUSTERING FORMALIZED AS A PARTITIONING PROBLEM ON GRAPHS

- Given a graph and a weight matrix, we search a partition in two clusters *(A,B)*
- The loss function for this partition is given by the cut

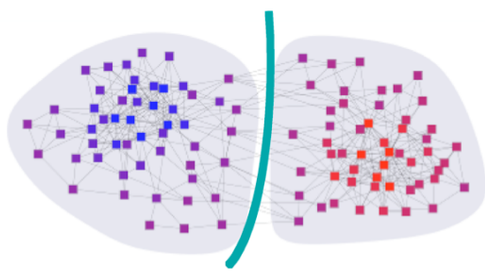
$$\text{cut}(A, B) = \sum_{i \in A, j \in B} w_{ij}$$

- Find the cut that minimizes this cut : the MinCut
- **Problem : all the cuts are not interesting !**



GET BALANCED CUTS

- A good partition should separate dissimilar vertices and should produce balanced clusters.



- Loss functions that favor such clusters are
  - The RatioCut

$$\text{RatioCut}(A, B) = \sum_{i \in A, j \in B} w_{ij} \left(\frac{1}{|A|} + \frac{1}{|B|} \right) = \text{cut}(A, B) \left(\frac{1}{|A|} + \frac{1}{|B|} \right)$$

  - The NormalizedCut :

$$\text{NormalizedCut}(A, B) = \sum_{i \in A, j \in B} w_{ij} \left(\frac{1}{\text{vol}(A)} + \frac{1}{\text{vol}(B)} \right) = \text{cut}(A, B) \left(\frac{1}{\text{vol}(A)} + \frac{1}{\text{vol}(B)} \right)$$

$$\text{with} \quad \text{vol}(A) = \sum_{i \in A} d_i$$

- **Problem : Both cuts are NP hard, so we need to relax them**

RELAXING BALANCED (SIMPLE) CUTS

$$\min_{A,B} \text{cut}(A,B) \qquad \text{s.t.} \qquad |A| = |B|$$

- We take a cluster membership function $f_i = \begin{cases} 1 & \text{if } v_i \in A \\ -1 & \text{if } v_i \in B \end{cases}$

$$\text{cut}(A,B) = \sum_{i \in A, j \in B} w_{ij} = \frac{1}{4} \sum_{i,j} w_{ij} (\mathbf{f}_i - \mathbf{f}_j)^2 = \frac{1}{2} \mathbf{f}^T \mathbf{L} \mathbf{f}$$

$$|A| = |B| \Rightarrow \sum_i f_i = 0 \Rightarrow \mathbf{f}^T \mathbf{1} = 0 \Rightarrow \mathbf{f} \perp \mathbf{1}$$

$$\|\mathbf{f}\| = \sqrt{N}$$

- **Relaxation : allow** $f_i \in \mathbb{R}$
- **Final objective :**

$$\min_{\mathbf{f}} \mathbf{f}^T \mathbf{L} \mathbf{f} \qquad \text{s.t.} \qquad f_i \in \mathbb{R}, \quad \mathbf{f} \perp \mathbf{1}, \quad \|\mathbf{f}\| = \sqrt{N}$$

RALEIGHT-RITZ THEOREM

- If $0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_N$ are the eigenvalues of a real symmetric matrix $\mathbf{L}$ then

$$\lambda_1 = \min_{\mathbf{f} \neq 0} \frac{\mathbf{f}^T \mathbf{L} \mathbf{f}}{\mathbf{f}^T \mathbf{f}} = \min_{\mathbf{f}^T \mathbf{f} = 1} \mathbf{f}^T \mathbf{L} \mathbf{f}$$

- which is equal to 0 and first eigenvector is $\mathbf{1}$

$$\lambda_{k+1} = \min_{\mathbf{f} \neq 0, \mathbf{f} \perp \mathbf{v}_1, \ldots \mathbf{v}_{k+1}} \frac{\mathbf{f}^T \mathbf{L} \mathbf{f}}{\mathbf{f}^T \mathbf{f}} = \min_{\mathbf{f}^T \mathbf{f} = 1, \mathbf{f} \perp \mathbf{v}_1, \ldots \mathbf{v}_{k+1}} \mathbf{f}^T \mathbf{L} \mathbf{f}$$

- The solution is given by the second eigenvector of the unnormalized Laplacian

$$\text{cluster}_i = \begin{cases} 1 & \text{if } f_i \geq 0 \\ -1 & \text{if } f_i < 0 \end{cases}$$

APPROXIMATING RATIO CUT

$$\text{RatioCut}(A, B) = \text{cut}(A, B) \left(\frac{1}{|A|} + \frac{1}{|B|} \right)$$

- Define a cluster membership function

$$f_i = \begin{cases} \sqrt{\frac{|B|}{|A|}} & \text{if } v_i \in A \\ -\sqrt{\frac{|A|}{|B|}} & \text{if } v_i \in B \end{cases}$$

- One can show that

$$\mathbf{f}^T \mathbf{L} \mathbf{f} = N \cdot \text{RatioCut}(A, B)$$

- **The solution is the same than for the simple cut**

NORMALIZED CUT

$$\text{NormalizedCut}(A, B) = \text{cut}(A, B)\left(\frac{1}{\text{vol}(A)} + \frac{1}{\text{vol}(B)}\right)$$

- Define a cluster membership function

$$f_i = \begin{cases} \frac{1}{\text{vol}(A)} & \text{if } v_i \in A \\ -\frac{1}{\text{vol}(B)} & \text{if } v_i \in B \end{cases}$$

$$\mathbf{f}^T\mathbf{L}\mathbf{f} = \frac{1}{2}\sum_{i,j} w_{ij}(\mathbf{f}_i - \mathbf{f}_j)^2 = \frac{1}{2}\left(\frac{1}{\text{vol}(A)} + \frac{1}{\text{vol}(B)}\right)^2 \text{cut}(A, B)$$

$$\mathbf{f}^T\mathbf{D}\mathbf{f} = \sum_i f_i^2 d_i = \sum_{i\in A} \frac{1}{\text{vol}(A)^2} d_i + \sum_{i\in B} \frac{1}{\text{vol}(B)^2} d_i$$

$$= \frac{1}{\text{vol}(A)^2}\text{vol}(A) + \frac{1}{\text{vol}(B)^2}\text{vol}(B) = \frac{1}{\text{vol}(A)} + \frac{1}{\text{vol}(B)}$$

therefore $$\frac{\mathbf{f}^T\mathbf{L}\mathbf{f}}{\mathbf{f}^T\mathbf{D}\mathbf{f}} = \text{NormalizedCut}(A, B)$$

NORMALIZED CUT

- On can show that $\frac{\mathbf{f}^T\mathbf{L}\mathbf{f}}{\mathbf{f}^T\mathbf{D}\mathbf{f}} = \frac{\mathbf{g}^T\boldsymbol{\mathcal{L}}\mathbf{g}}{\mathbf{g}^T\mathbf{g}}$ with $\mathbf{g} = \mathbf{D}^{1/2}\mathbf{f}$

$$\frac{\mathbf{f}^T\mathbf{L}\mathbf{f}}{\mathbf{f}^T\mathbf{D}\mathbf{f}} = \frac{\mathbf{g}^T\mathbf{D}^{-1/2}\mathbf{L}\mathbf{D}^{-1/2}\mathbf{g}}{\mathbf{g}^T\mathbf{D}^{-1/2}\mathbf{D}\mathbf{D}^{-1/2}\mathbf{g}} = \frac{\mathbf{g}^T\boldsymbol{\mathcal{L}}\mathbf{g}}{\mathbf{g}^T\mathbf{g}}$$

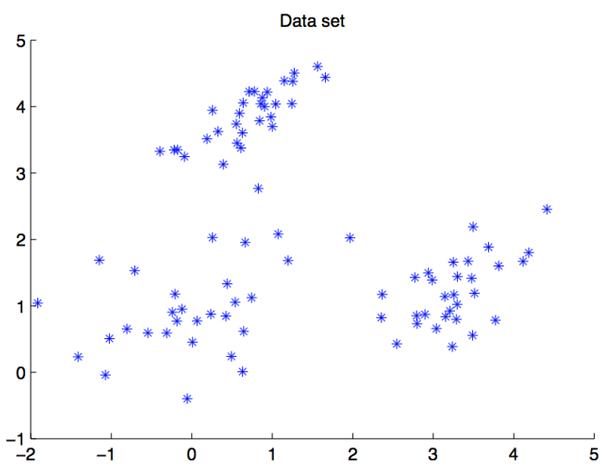
- And one has

$$\lambda_{k+1} = \min_{\mathbf{g}\neq 0, \mathbf{g}\perp \mathbf{D}^{1/2}\mathbf{v}_1, \ldots \mathbf{D}^{1/2}\mathbf{v}_{k+1}} \frac{\mathbf{g}^T\boldsymbol{\mathcal{L}}\mathbf{g}}{\mathbf{g}^T\mathbf{g}} = \min_{\mathbf{f}\neq 0, \mathbf{f}\perp \mathbf{v}_1, \ldots \mathbf{v}_{k+1}} \frac{\mathbf{f}^T\mathbf{L}\mathbf{f}}{\mathbf{f}^T\mathbf{D}\mathbf{f}}$$

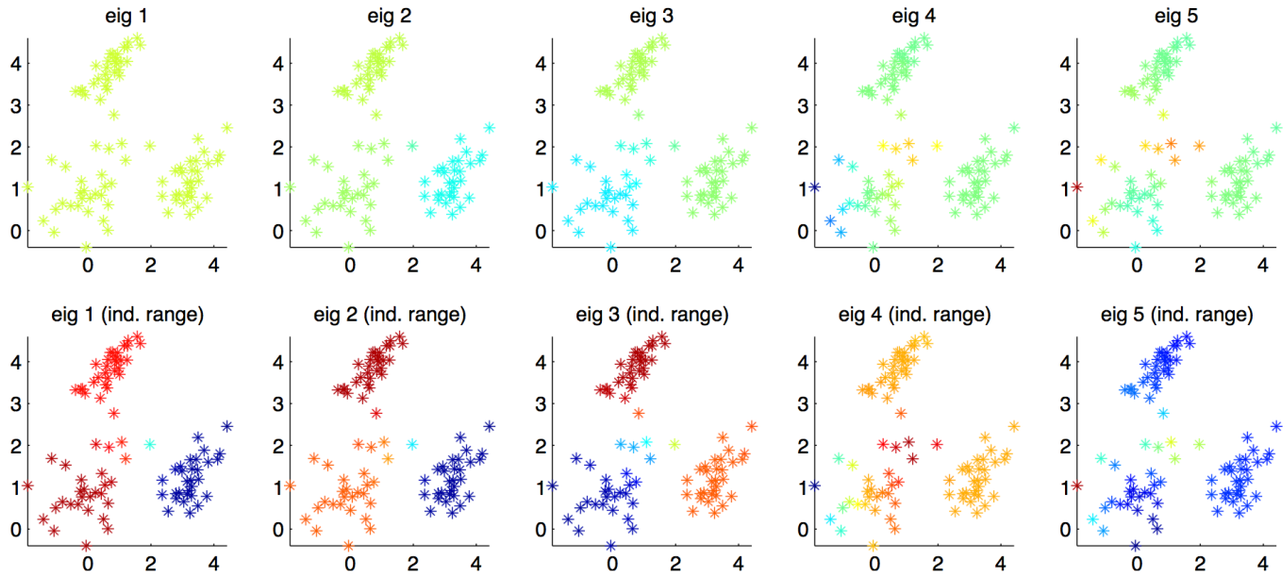
- **The solution of the normalized cut is given by the second eigenvector of the normalized Laplacian**

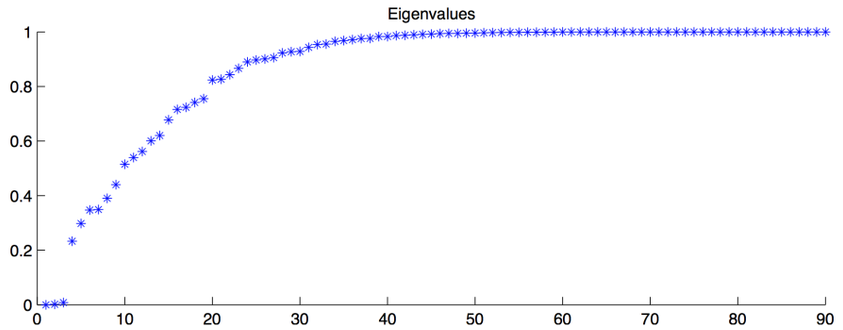
EXTENSION TO SEVERAL CLUSTERS

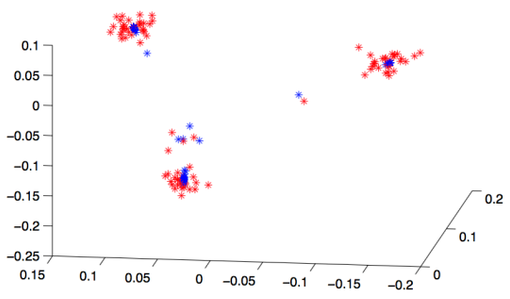
- Input: $\{(\mathbf{x}_1, y_i), \cdots, (\mathbf{x}_N, y_N)\}$ and a graph $G = (V, E)$
- Compute matrices $\mathbf{W}$, $\mathbf{D}$ and $\mathbf{L}$ (or $\boldsymbol{\mathcal{L}}$).
- Compute the first k eigenvectors $\mathbf{v}_1, \cdots, \mathbf{v}_k$ of $\mathbf{L}$
- Let $\mathbf{V} \in \mathbb{R}^{N \times k}$ the matrix that contains the eigenvectors as columns
- Ley $\mathbf{y}_i \in \mathbb{R}^k$ be the i-th row of matrix $\mathbf{V}$
- Cluster the points $\mathbf{y}_1, \cdots, \mathbf{y}_N$ into k clusters with k-means

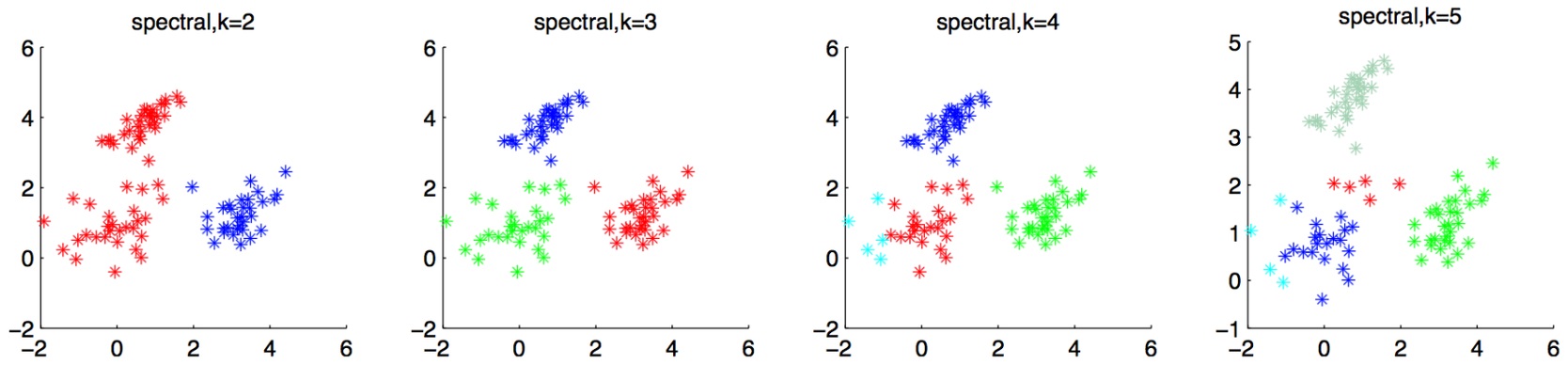


TOY EXAMPLE


eig 1
eig 2
eig 3
eig 4
eig 5
eig 1 (ind. range)
eig 2 (ind. range)
eig 3 (ind. range)
eig 4 (ind. range)
eig 5 (ind. range)

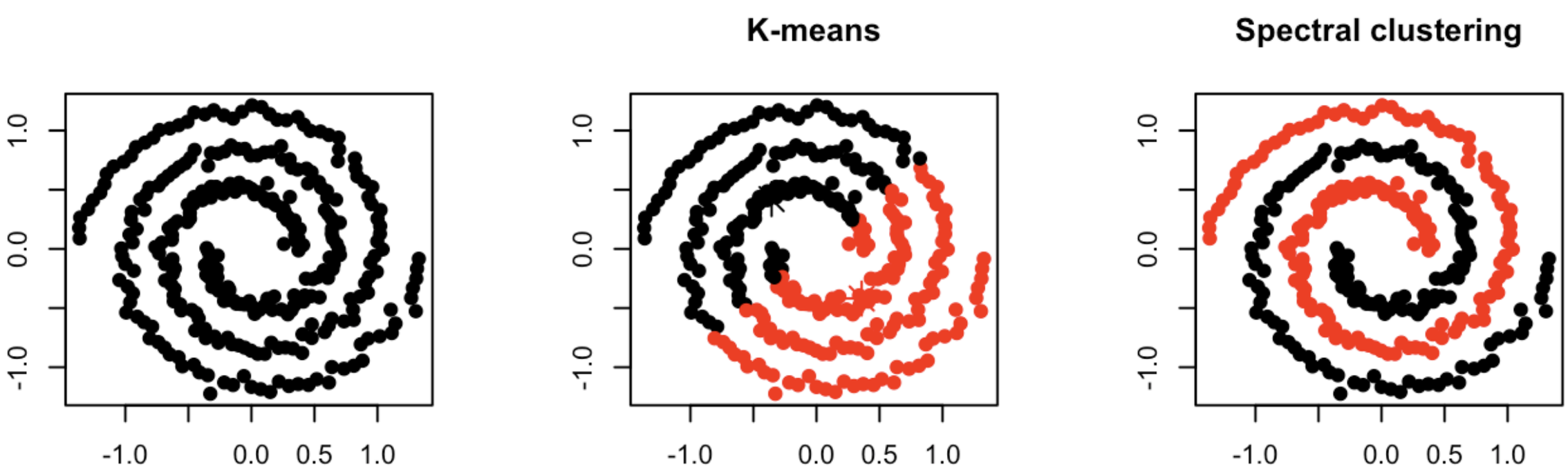

Eigenvalues



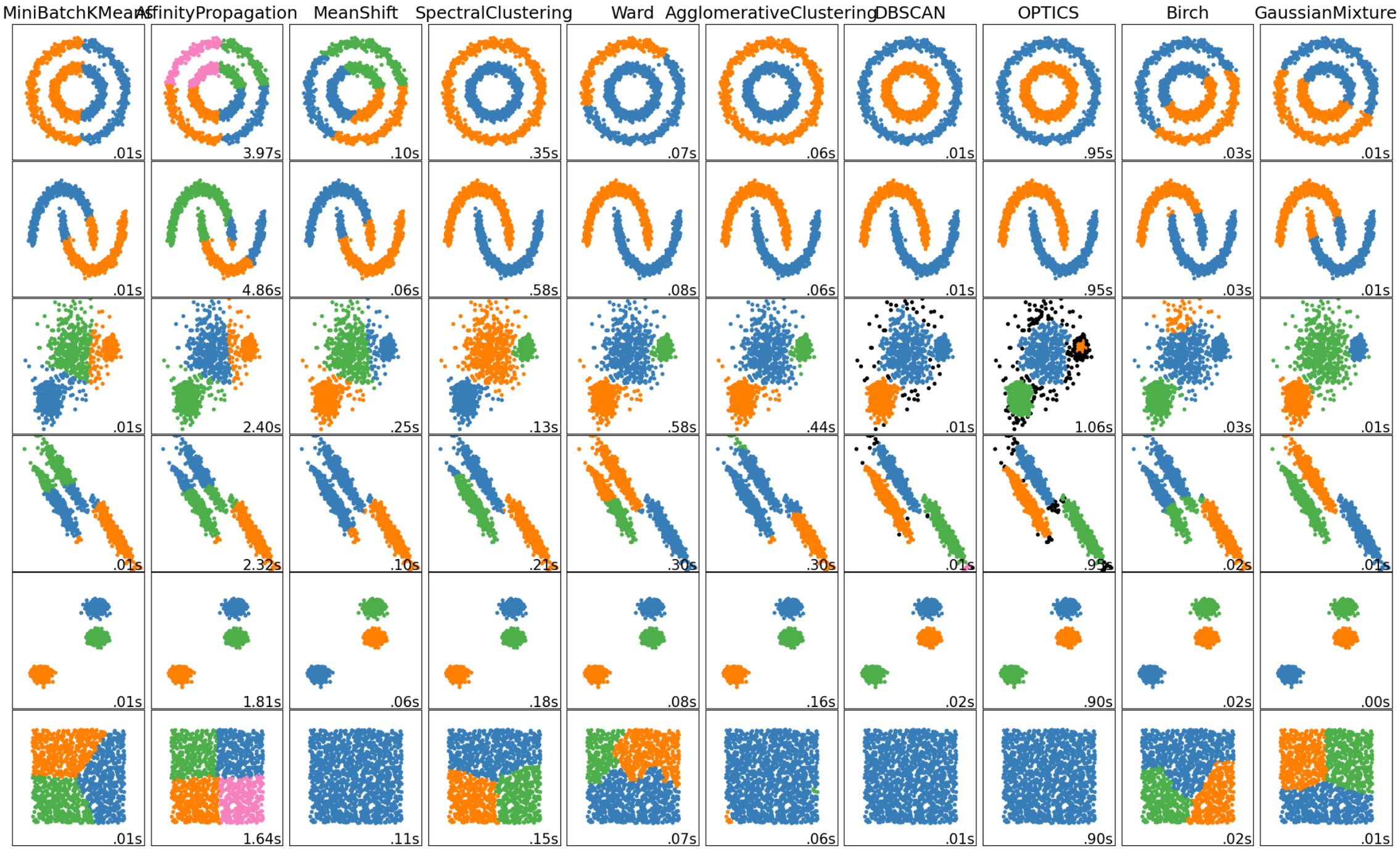

spectral,k=2
spectral,k=3
spectral,k=4
spectral,k=5

EXAMPLES

We can get non convex clusters with spectral clustering



THE END


MiniBatchKMeans
AffinityPropagation
MeanShift
SpectralClustering
Ward
AgglomerativeClustering
DBSCAN
OPTICS
Birch
GaussianMixture
.01s
3.97s
.10s
.35s
.07s
.06s
.01s
.95s
.03s
.01s
.01s
4.86s
.06s
.58s
.08s
.06s
.01s
.95s
.03s
.01s
.01s
2.40s
.25s
.13s
.58s
.44s
.01s
1.06s
.03s
.01s
.01s
2.32s
.10s
.21s
.30s
.30s
.01s
.95s
.02s
.01s
.01s
1.81s
.06s
.18s
.08s
.16s
.02s
.90s
.02s
.00s
.01s
1.64s
.11s
.15s
.07s
.06s
.01s
.90s
.02s
.01s