MACHINE LEARNING

MEI/1

University of Beira Interior,
Department of Informatics

Hugo Pedro Proença,
hugomcp@di.ubi.pt, 2024/2025

Machine Learning

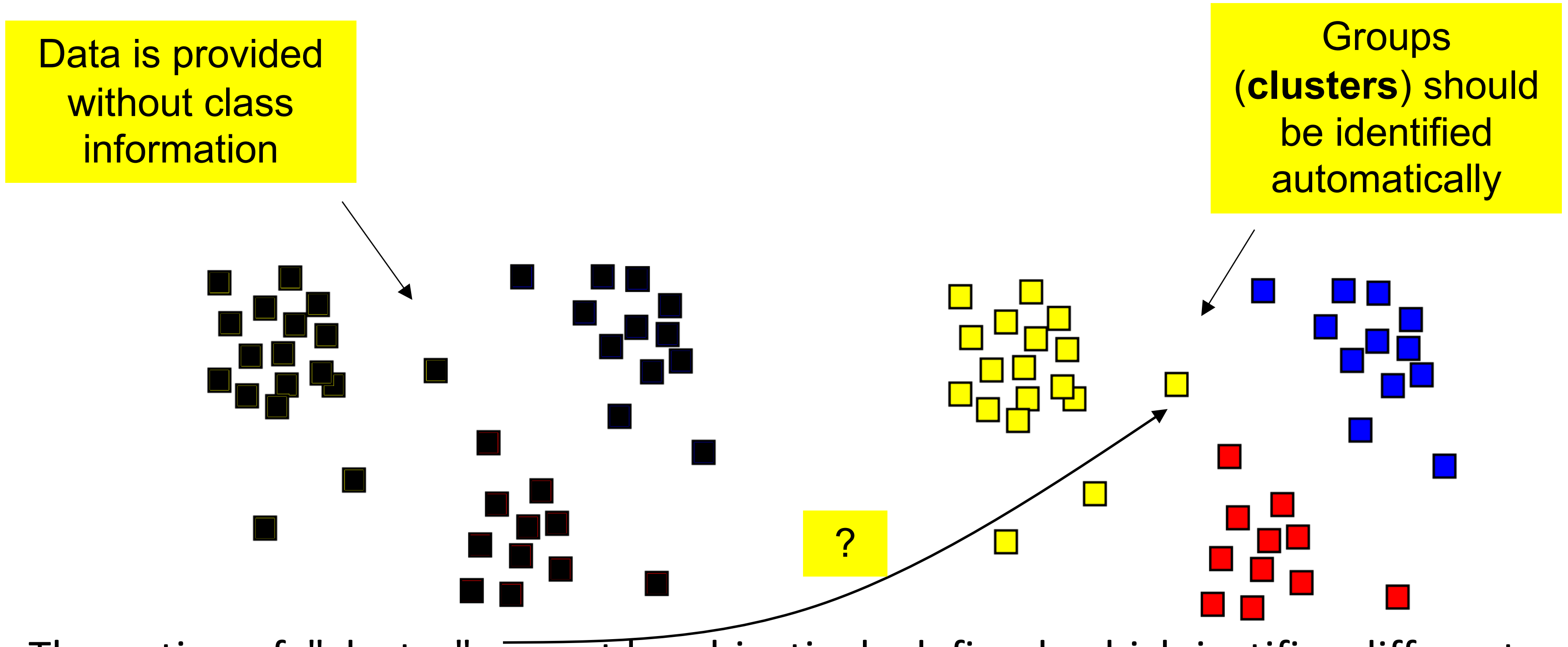
[06]

Syllabus

- Unsupervised Learning
  - K-Means
  - DBScan
  - Self Organized Maps
  - Deep Clustering

Unsupervised Learning

- This concept is associated to learning without a “**supervisor**”
  - It also known as self-organization, or cluster analysis
- The basic idea is that, instead of attempting to mimic the behavior of the supervisor, to identify commonalities in the data


Data is provided without class information
Groups (**clusters**) should be identified automatically
?

- The notion of "cluster" cannot be objectively defined, which justifies different clustering algorithms.

Unsupervised Learning

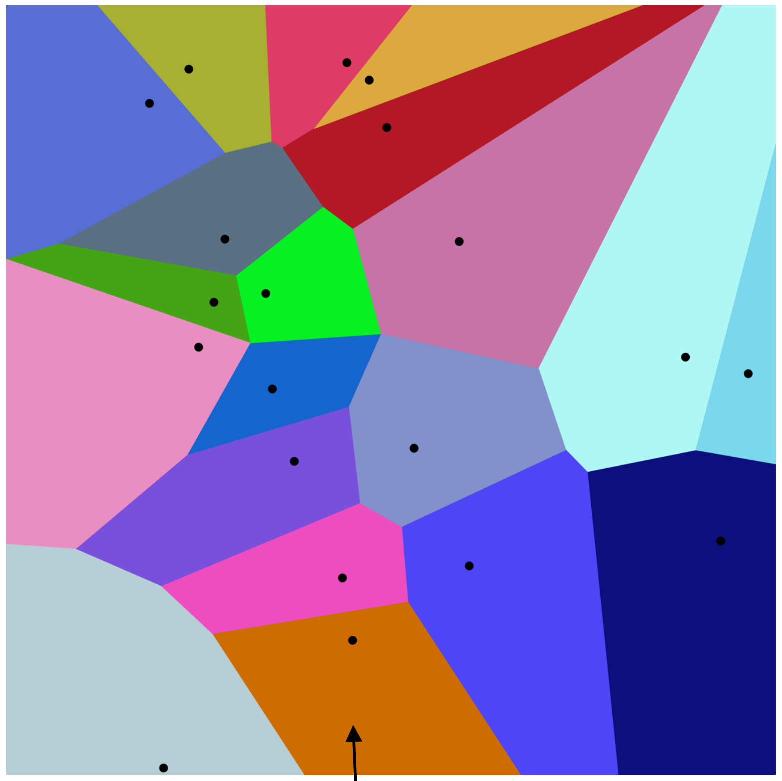
- There are different families of methods to perform clustering:
  - **Connectivity** models, in which models are built based on distance connectivity
    - Hierarchical clustering
  - **Centroid** models, that represent clusters by mean vectors (i.e., centroids)
    - K-means
  - **Distribution** models, where clusters are modelled according to statistical distributions
    - DBSCAN
  - **Neural** models, where networks implement a form of PCA that finds appropriate feature subspaces
    - Self-Organizing Map (SOM)
- **Clusters Evaluation**
  - **Internal** Evaluation, when the model is evaluated based on the data that was clustered itself
    - Davies-Bouldin index: $DB = \frac{1}{N} \sum_{i=1}^{N} \max_{j \neq i} \frac{\sigma_i + \sigma_j}{d(ci,cj)}$

    where “c” represents one centroid, “σ” is the average distance of the elements in one cluster to its centroid and “N” is the number of clusters
  - **External** Evaluation, when the model is evaluated based on new data, typically with class labels
    - Purity: $P = \frac{1}{N} \sum_{i=1}^{M} \max_{d \in D} |m \cap d|$

    where “M” represents the set of clusters, and “D” is the labeled data

K-Means

- It is the most used clustering algorithm, due to its effectiveness and easiness of implementation.
  - Aims to partition **"n" observations** into **"k" clusters**
  - Each observation belongs to the nearest cluster centroid, which is the prototype of the cluster.
  - This results in a partitioning of the data space into Voronoi cells.
    - A Voronoi diagram is a partitioning of a plane into regions based on distance to points in a subset of the plane.
    - These points (a.k.a. prototypes) determine the shape of the corresponding Voronoi cell.
    - For each prototype there is a corresponding region consisting of all points closer to that seed than to any other. These regions are called Voronoi cells.



Positions in each cell (color) are closest to the corresponding centroid than to any other

K-Means

- For K-Means, the value of “K” must be given beforehand
  - There are diferente heuristics to automatically find the optimal value of “K”, but depend of the specific problema considered
- Having a data set **X**: $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_n\}$

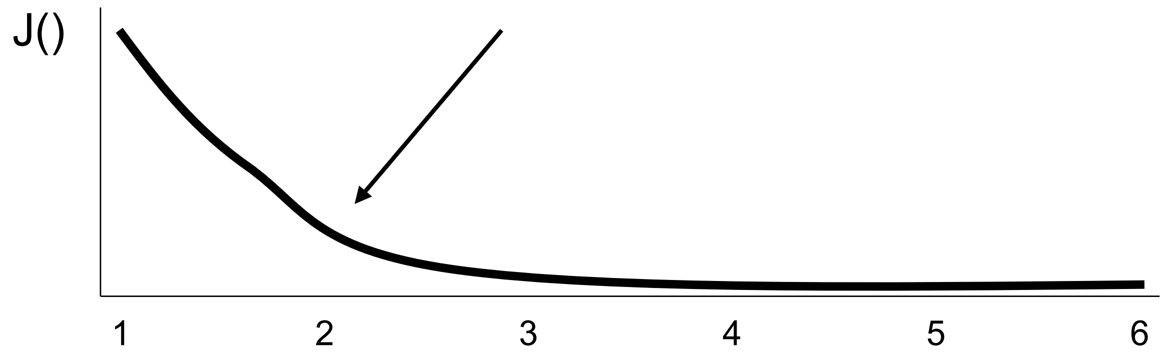
```
1.  Initialize (randomly)   “K” centroids $\boldsymbol{\mu}$: $\{\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \dots, \boldsymbol{\mu}_k\}$
2.  While (¬ stopping_criterium($\boldsymbol{\mu}$, X))

    1.  For every $\boldsymbol{x}_i$:
            $\mathbf{c}_i = \arg\min_i d(\boldsymbol{x}_i, \boldsymbol{\mu}_i)$            //cluster assignment

    2.  For every $\boldsymbol{\mu}_i$:
            $\boldsymbol{\mu}_i = \sum_j^n \boldsymbol{x}_j \mid \boldsymbol{x}_j$ assigned to $\mathbf{c}_i$      // centroid update
```

K-Means

- **Stopping criteria**. There are a number of diferent possibilities
  - Simplistic: Predefine a **number of iterations**
    - Might be "*too many*", or "*too few*", depending of the complexity of the feature space
  - Elaborate 1: Evaluate clusters **stationarity** and stop when the changes in clusters positions between consecutive iterations is less than a small threshold.
  - Elaborate 2: Evaluate samples **assignments** and stop when no samples (or a very small number) of samples changes its centroid between consecutive iterations.
- **Choose the value of "K"**
  - Elbow method.
  - Define a cost function J() and repeat the clustering procedure for a growing number of clusters. Define "K" as the value where the curvature of J() is maximal

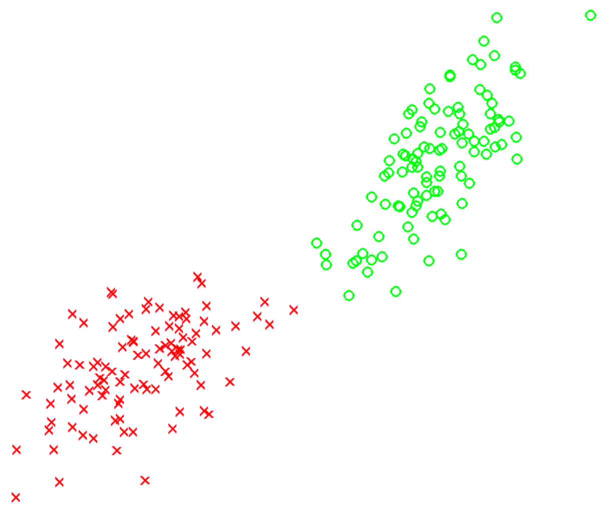


K-Means

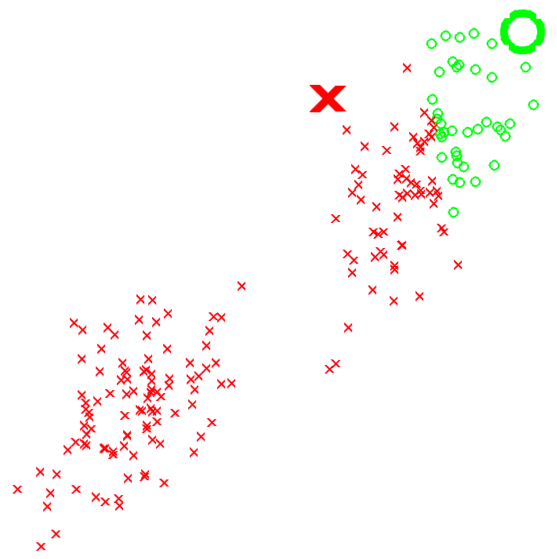
- **Distance Functions**
  - Different functions can be used, as long as they met the properties of being a "**metric**"
  - A metric on a set X is a function d : X × X → [0 , ∞), where for all x , y , z ∈ X, the following conditions are satisfied:
    - d(**x**, **y**) ≥ 0 // non-negativity or separation axiom
    - d(**x**, **y**) = 0 ⇔ **x** = **y** //identity of indiscernibles
    - d(**x**, **y**) = d(**y**, **x**) //symmetry
    - d(**x**, **z**) ≤ d(**x**, **y**) + d(**y**, **z**) //triangle inequality
  - Examples:
    - Euclidean distance: d(**x**,**y**) = $\sqrt{\sum(x_i - yi)^2}$
    - Manhatan distance: d(**x**,**y**) = $\sum(|xi - yi|)$
    - Chebyshev distance: d(**x**,**y**) = $\max|xi - yi|$

K-Means: Example

- Consider the following synthetic dataset:

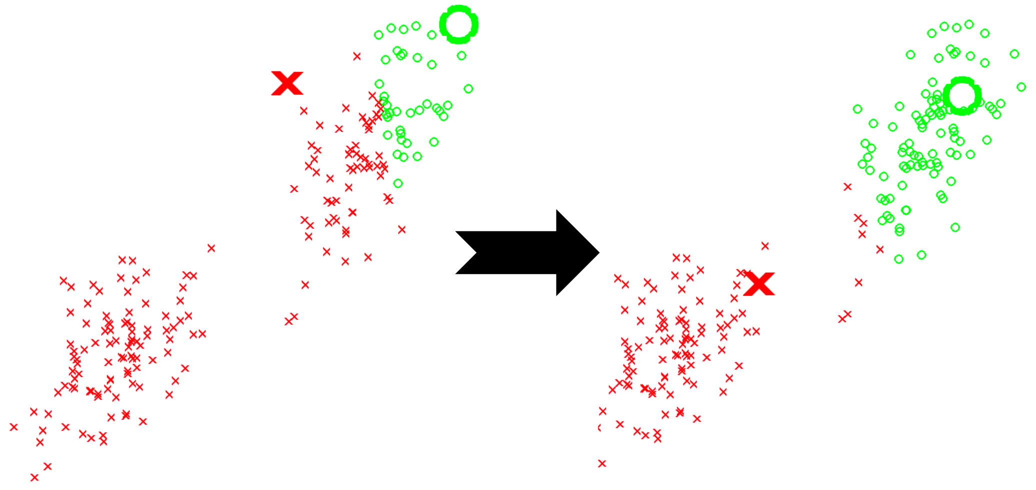


- Random initialization of 2 clusters:

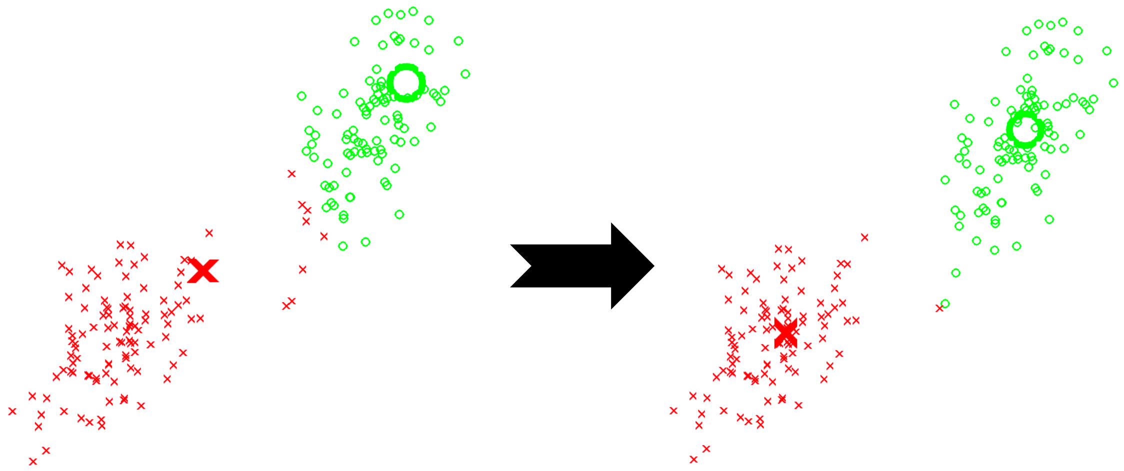


K-Means: Example

- K-Means: Iteration 1

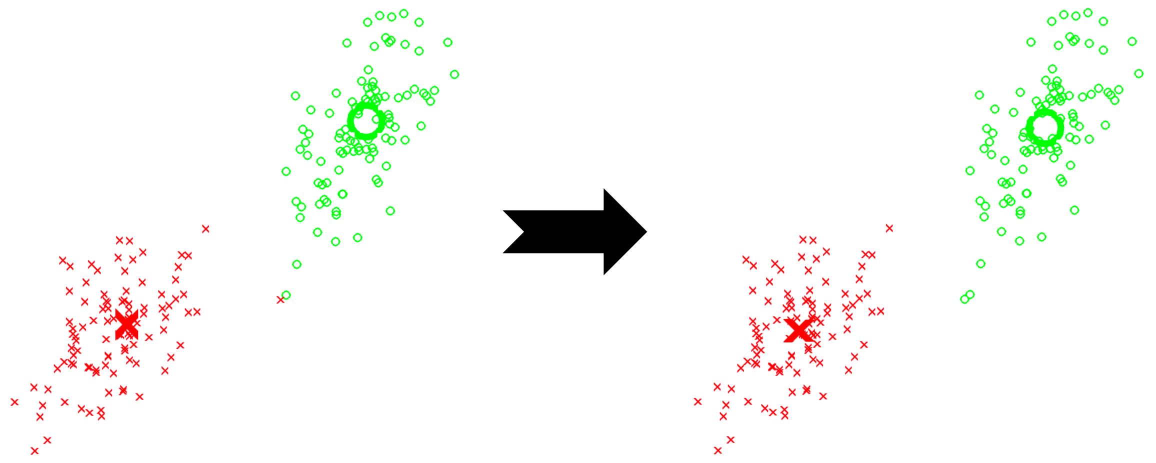


- K-Means: Iteration 2

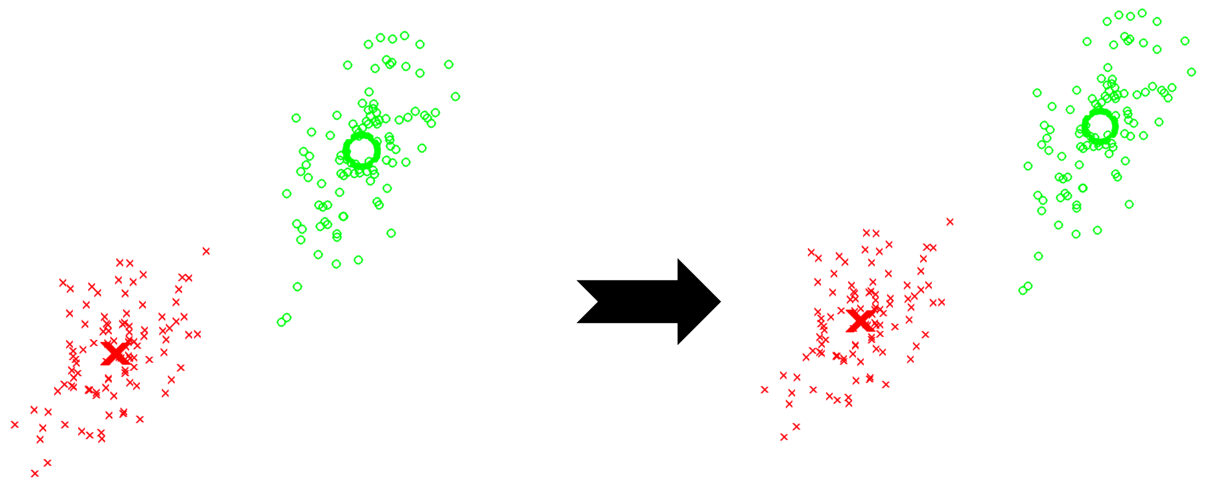


K-Means: Example

- K-Means: Iteration 3

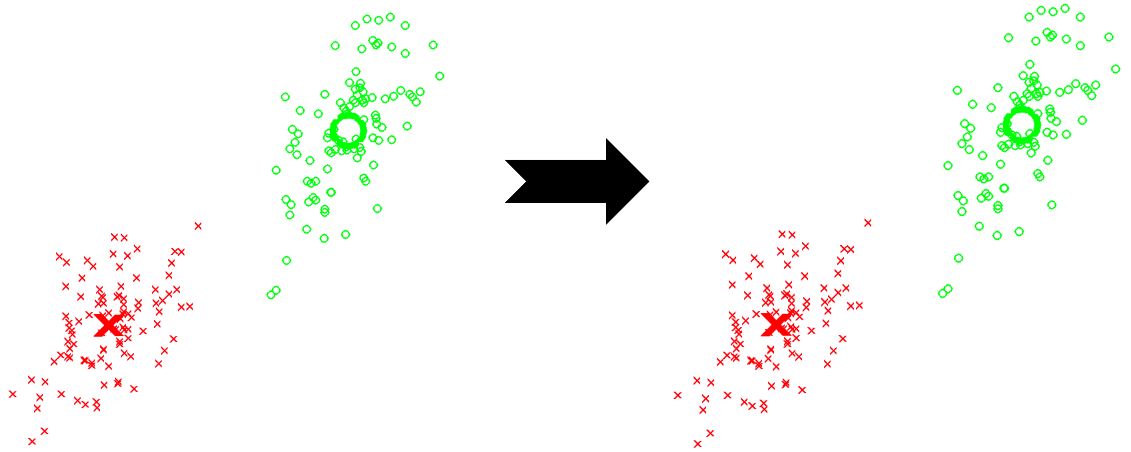


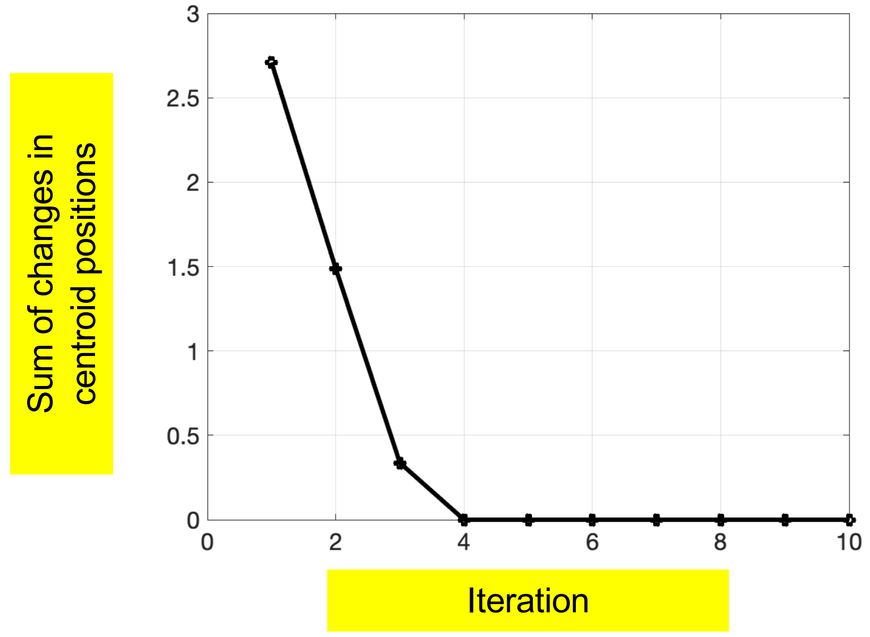
- K-Means: Iteration 4

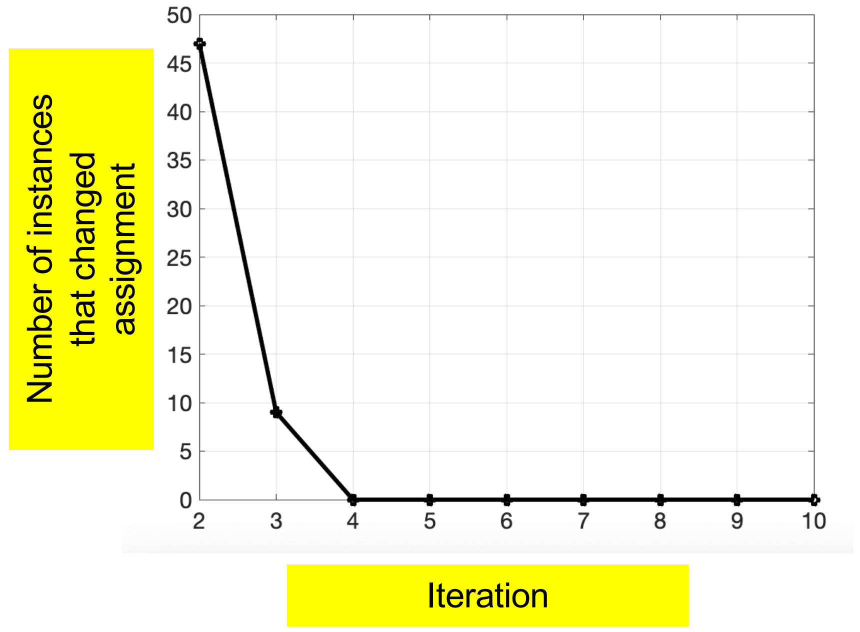


K-Means: Example

- K-Means: Iteration 5

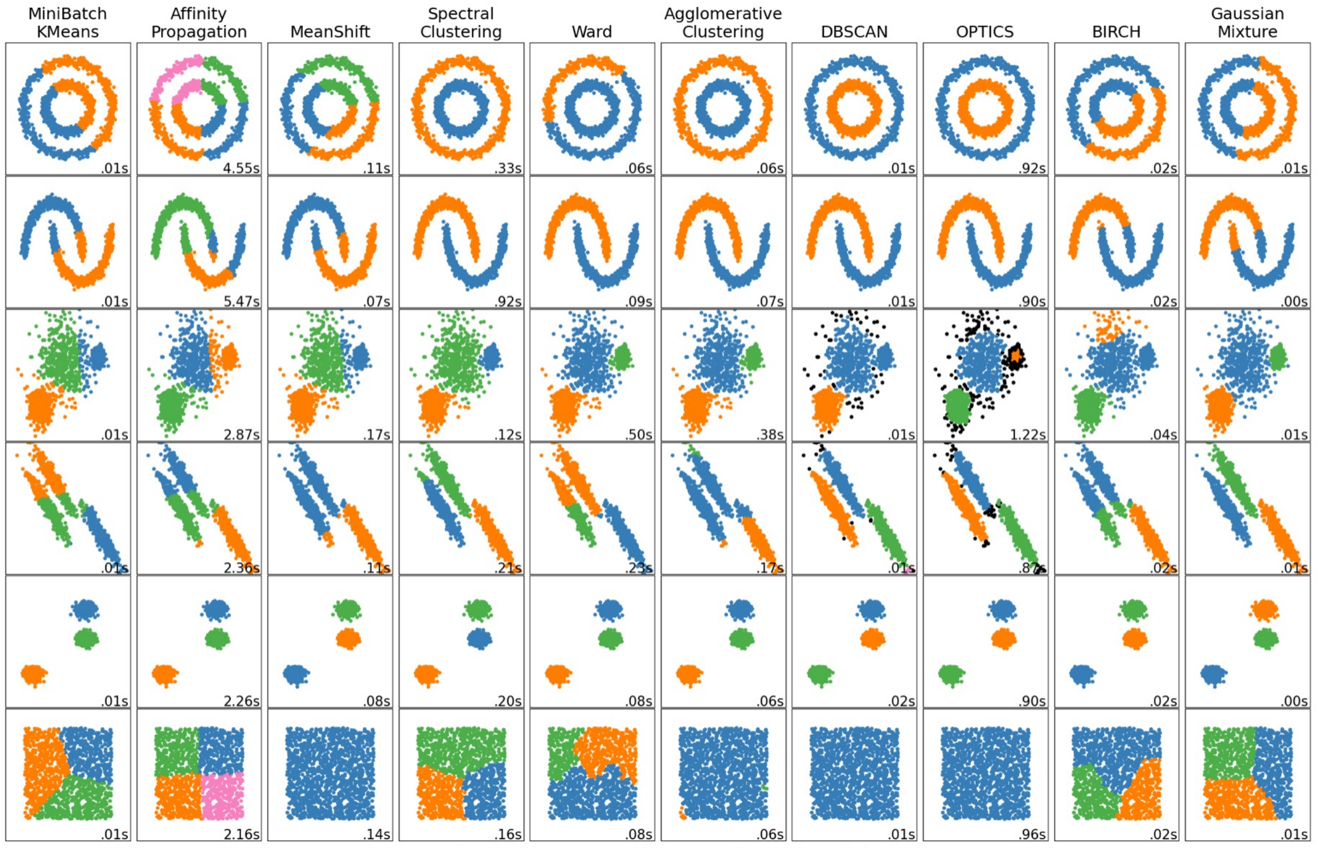






Unsupervised Learning

- There is a variety of clustering algorithms available for multiple languages.
- As an example, the “*sklearn.cluster*” library offers:



Unsupervised Learning

- The existing clustering algorithms can be broadly divided into two families:
  - **Hard Clustering**: Here, each data point either belongs to a cluster completely or not.
  - **Soft Clustering**: Instead of putting each data point into a separate cluster, a probability or likelihood of that data point to be in those clusters is assigned.

- In terms of the techniques used in clustering algorithms, four main families can be identified:
  - **Connectivity models**: Models are based on the notion that the data points closer in data space exhibit more similarity to each other than the data points lying farther away. These models are easy to interpret but lack scalability for handling big datasets. Examples are hierarchical clustering algorithm and its variants.
  - **Centroid models**: These are iterative algorithms, in which the notion of similarity is derived by the closeness of a data point to the centroid of the clusters. K-Means clustering algorithm is a popular algorithm that falls into this category. In these models, the no. of clusters is required at the end have to be mentioned beforehand, which makes it important to have prior knowledge of the dataset.
  - **Distribution models**: Based on the notion of how probable is it that all data points in the cluster belong to the same distribution (e.g., Gaussian). A popular example of these models is Expectation-maximization algorithm which uses multivariate normal distributions.
  - **Density Models**: These models search the data space for areas of varied density of data points in the data space. They isolate different density regions and assign the data points within these regions in the same cluster. Popular examples of density models are DBSCAN and OPTICS.

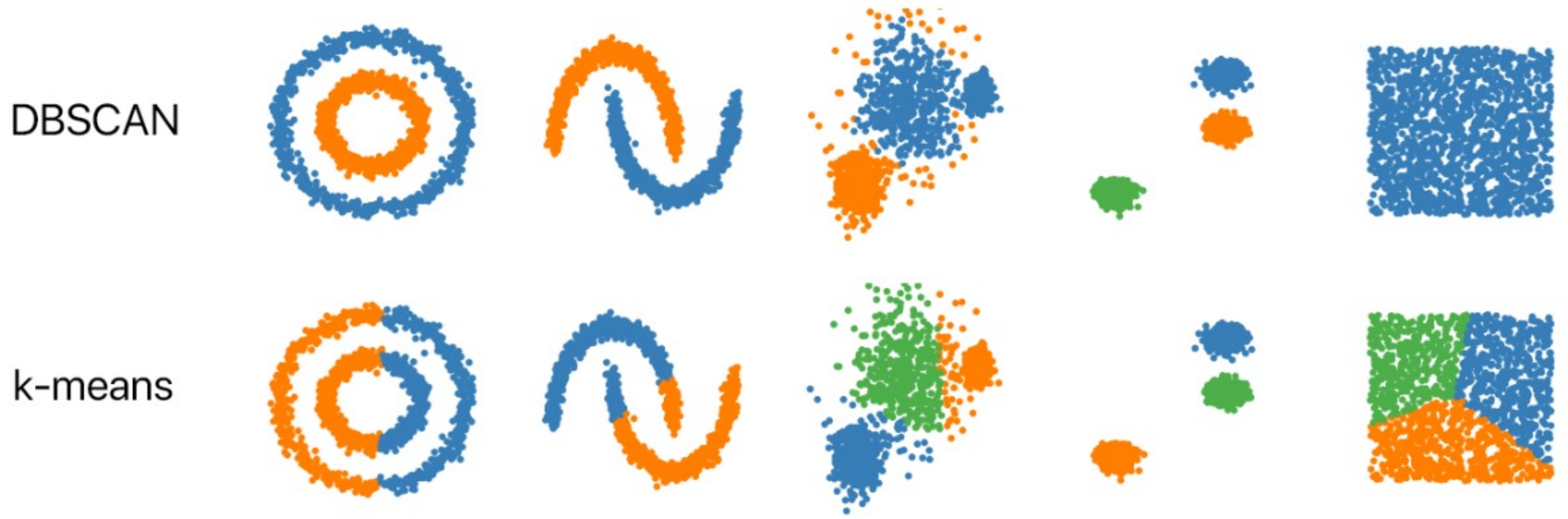
DBSCAN

- ❑The DBSCAN algorithm is a particularly interesting example.
- ❑It was proposed by Martin Ester et al. in 1996. DBSCAN is a density-based clustering algorithm that works on the assumption that clusters are **dense regions** in space **separated** by regions of **lower density**.
- ❑It groups ‘densely grouped’ data points into a single cluster.
- ❑It can identify clusters in large spatial datasets by looking at the local density of the data points.
- ❑The most exciting feature of DBSCAN clustering is that it is robust to outliers. It also does not require the number of clusters to be told beforehand, unlike K-Means, where we must specify the number of centroids..

Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD'96). AAAI Press, 226–231.

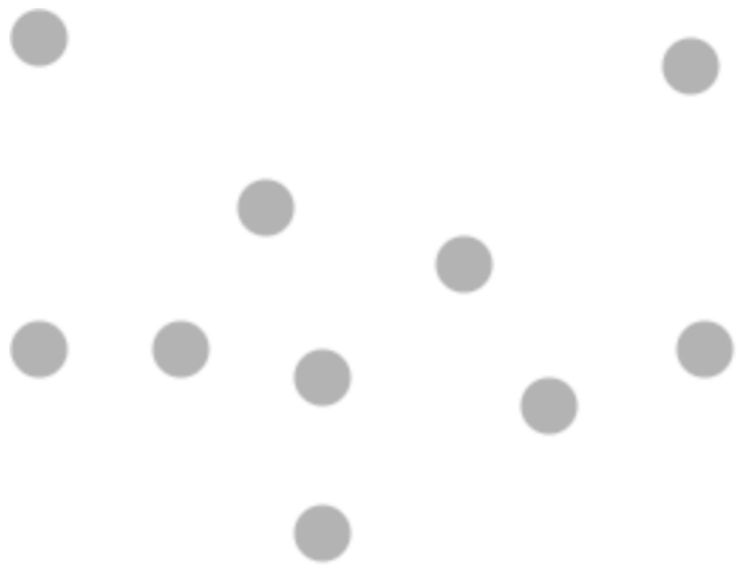
DBSCAN

- ❑ DBSCAN requires only two parameters: “epsilon” and “minPoints”.
- ❑“Epsilon” is the radius of the circle to be created around each data point to check the density
- ❑“minPoints” is the minimum number of data points required inside that circle for that data point to be classified as a **Core point**.
- ❑In high dimensions the circle most be understood as an hypersphere, where “epsilon” is the radius of that hypersphere, and minPoints is the minimum number of data points required inside that hypersphere.



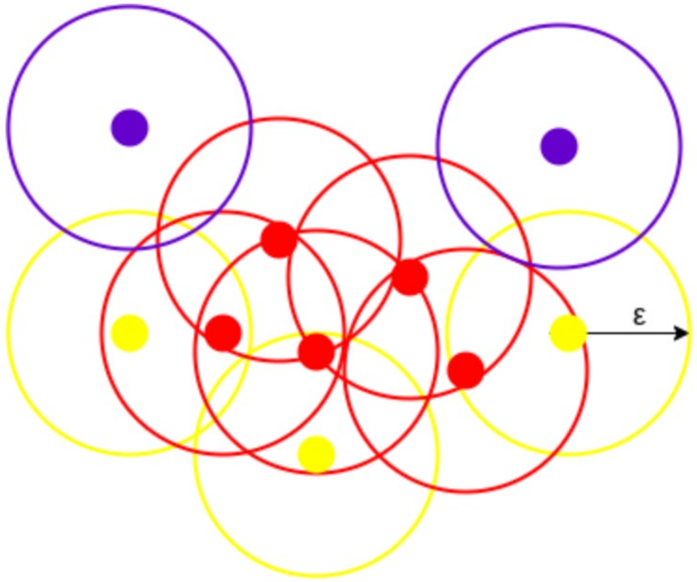
DBSCAN

- Let us consider a 2D example with 10 points. In this case, it will correspond to 10 instances, each one represented with two feature values.



DBSCAN

- ❑The algorithm starts by defining a neighborhood of radius ε around each point, and classifies them into one of three families:
  - ❑**Core**: A point is considered a "Core" if it has at least "minPoints" in its neighborhood.
  - ❑**Border**: A point is considered a "Border" if it has some neighbors, but less than "minPoints"
  - ❑**Noise**: A point is considered an outlier (noise) when there are no neighbors in its neighborhood.
- ❑In the example below, "minPoints=3"



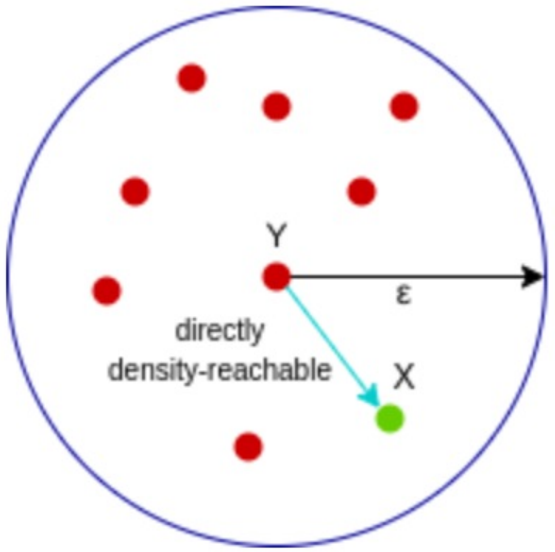
DBSCAN

- The key concepts in DBSCAN are “Reachability” and “Connectivity”.
  - Reachability states if a data point can be accessed from another data point directly or indirectly;
  - Connectivity states whether two data points belong to the same cluster or not.

- Generally, two points can be referred in DBSCAN as:
  - Directly Density-Reachable
  - Density-Reachable
  - Density-Connected

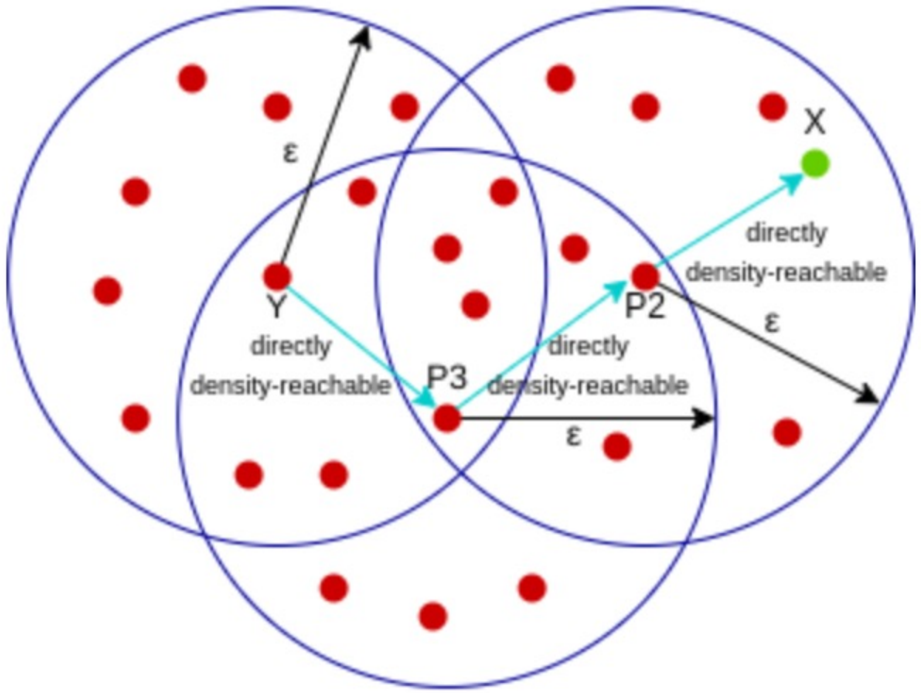
DBSCAN: Directly Density-Reachable

- ❑A point **X is density reachable from Y** when Y is a "**Core**" point and dist(X,Y) < ε.
  - ❑Here dist(,) is typically the Euclidean distance, but other distance metrics can be used.



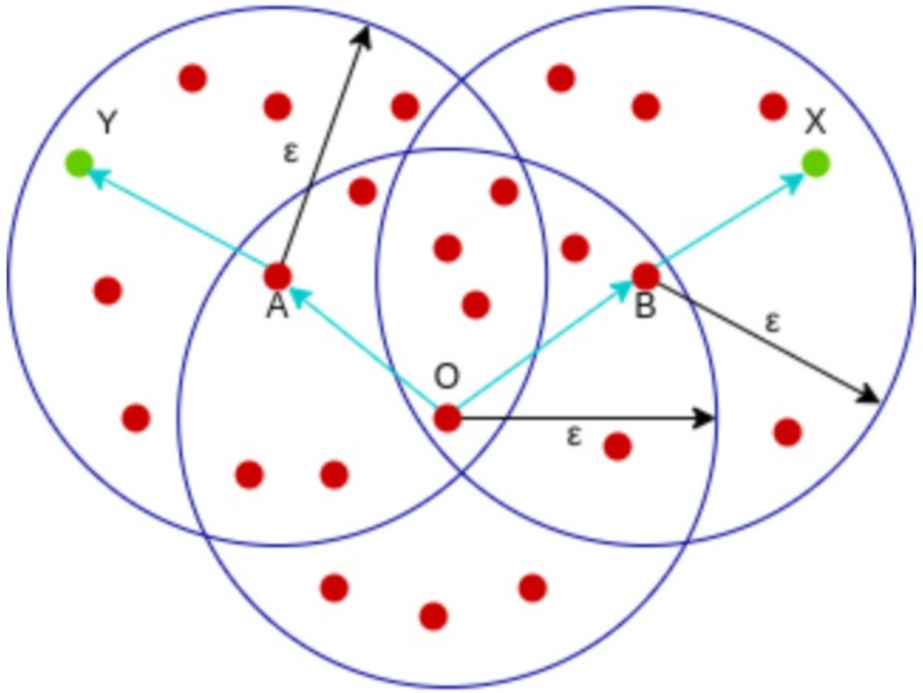
DBSCAN: Density Reachable

- A point X is density-reachable from point Y w.r.t epsilon, minPoints if there is a chain of points p_1, p_2, p_3, …, p_n and p_1=X and p_n=Y such that p_{i+1} is directly density-reachable from p_i.
  - In practice terms, there is the concept of “transitivity” here. If we can move “step-by-step” between points p_1 and p_n , then the latter çpoint is said to be density reachable from the former.



DBSCAN: Density-Connected

- Finally, a point X is density-connected from point Y w.r.t epsilon and minPoints if there exists a point O such that both X and Y are density-reachable from O w.r.t to epsilon and minPoints.
  - In the example below, X and Y are both density reachable from O. Hence, we say that X is density-connected to Y



DBSCAN

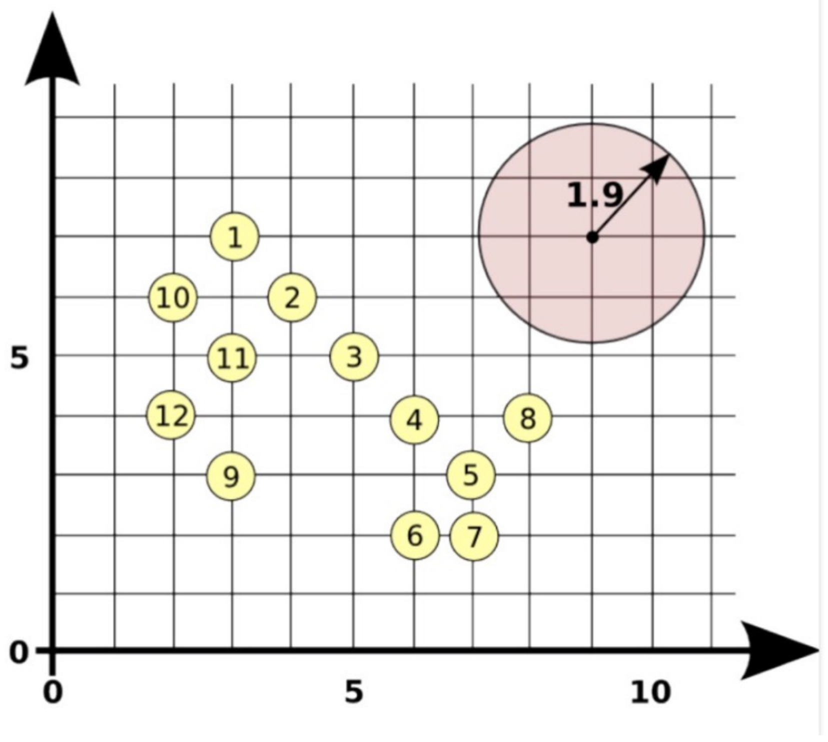
- DBSCAN is very sensitive to the values of the parameters.
- It is very important to understand how to select the values of ε and minPoints.
- A slight variation in these values significantly changes the results produced by the DBSCAN algorithm.
- In practice, the value of minPoints should be at least one greater than the number of dimensions of the dataset, i.e.,
  - minPoints>=Dimensions+1.
  - It does not make sense to take minPoints as 1 because it will result in each point being a separate cluster.
  - Therefore, it must be at least 3. Generally, it is twice the dimensions. But domain knowledge also decides its value.
- The value of epsilon can be decided from the K-distance graph.
  - The point of maximum curvature (elbow) in this graph tells us about the value of epsilon. If the value of epsilon chosen is too small then a higher number of clusters will be created, and more data points will be taken as noise. Whereas, if chosen too big then various small clusters will merge into a big cluster, and we will lose details.

DBSCAN

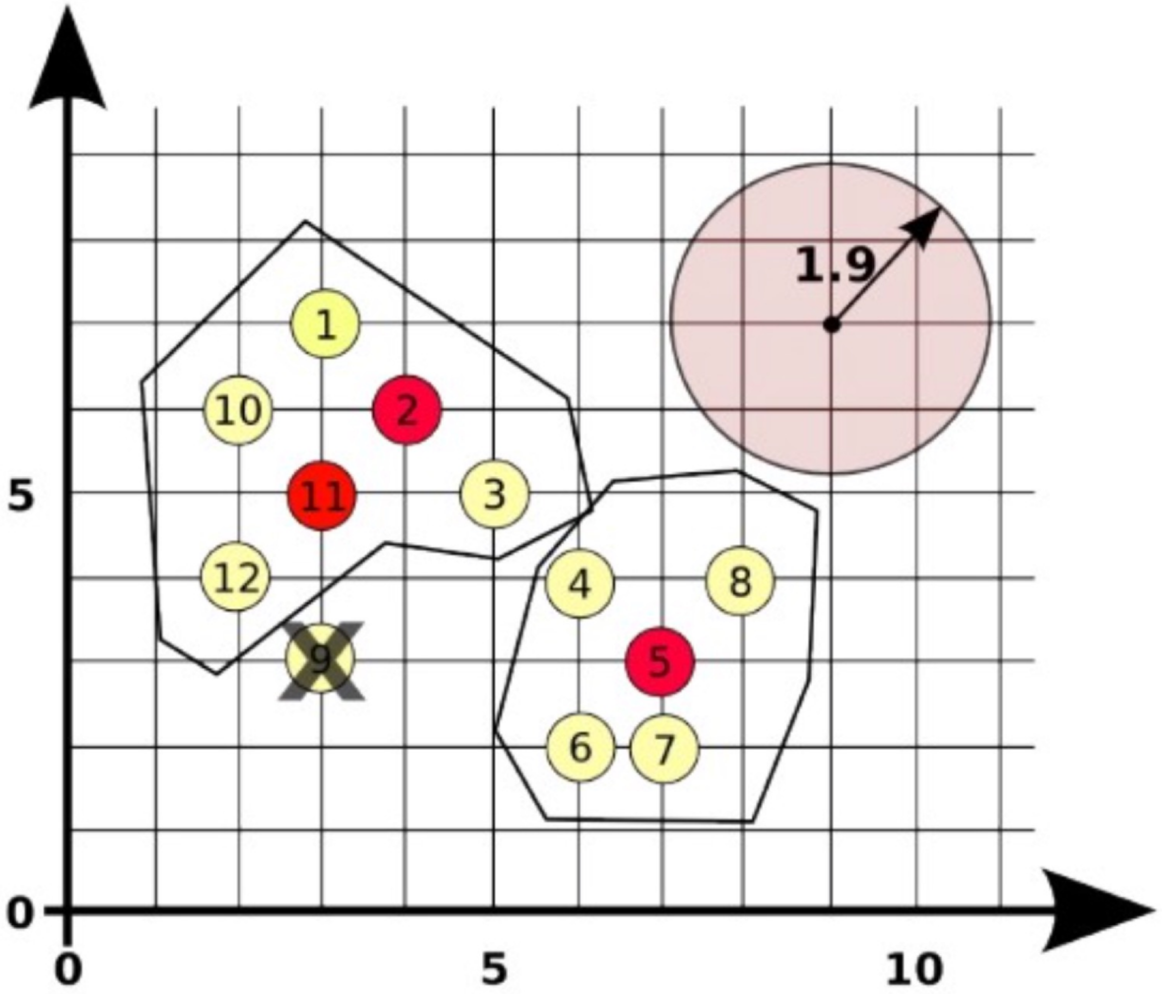
- ❑Finally, if p is a core point, then it forms a cluster together with all points (core or non-core) that are reachable from it.
  - ❑Each cluster contains at least one core point; non-core points can be part of a cluster, but they form its "edge", since they cannot be used to reach more points.
- ❑According to this definition, clusters satisfy two properties:
  - ❑All points within the cluster are mutually density-connected.
  - ❑If a point is density-reachable from some point of the cluster, it is part of the cluster as well.

DBSCAN Exercise

- ❑Apply the DBSCAN algorithm with $\varepsilon = 1.9$ and minPoints=3
  - ❑Start by discriminate between “Core”, “Border” and “Noise” points.
  - ❑Finally, indicate the clusters obtained.

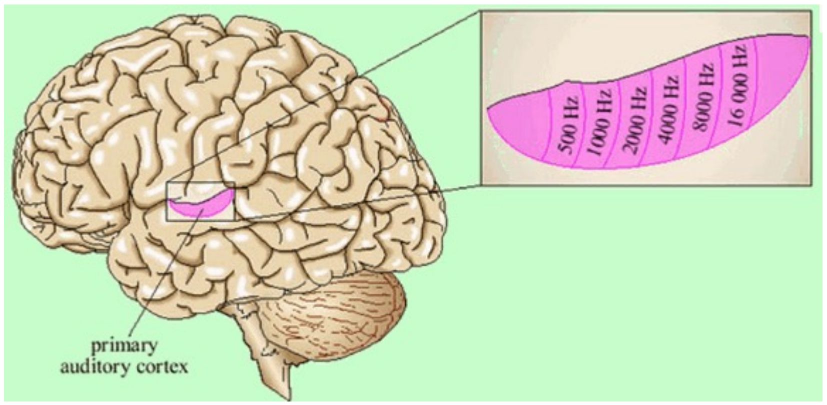


DBSCAN Exercise: Solution



Unsupervised Learning: SOMs

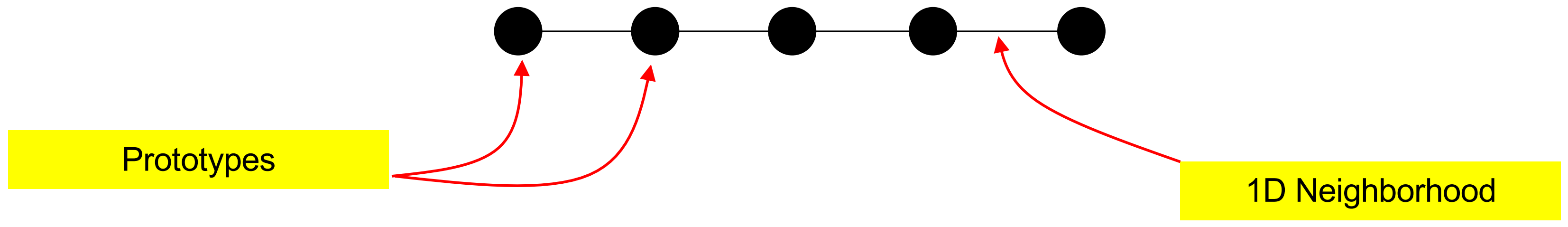
- The **Self-Organizing Map** (**SOM**) algorithm was one of the earliest neural network models (proposed by Kohonen in 1984), as a way to explain the spatial organization of the brain's functions, as observed especially in the cerebral cortex.
  - As an example, sound signals of different frequencies are mapped to the primary auditory cortex in which neighboring neurons respond to similar frequencies.



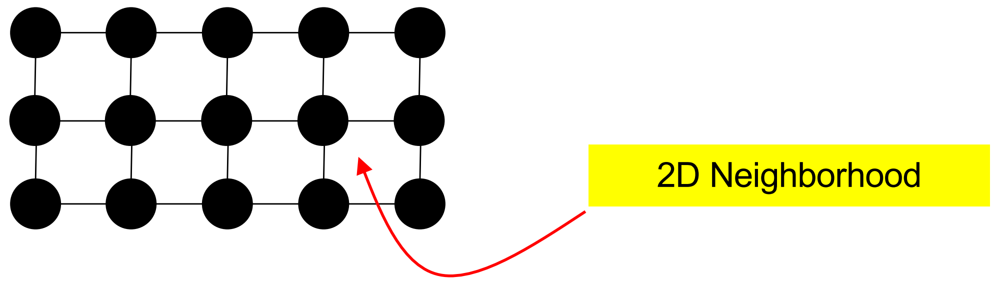
- Following the ideas of spatially ordered line detectors (Von der Malsburg, 1973) and the neural fields (Amari, 1980), the key points in SOMs were:
  - introduce a model composed of two interacting subsystems of different natures, where the key is a competitive neural network that implements the winner-take-all function,
  - Design a synaptic plasticity model for the neurons in learning. In practice terms, the learning is restricted spatially to the local neighborhood of the most active neurons.

Unsupervised Learning: SOMs

- Computationally, a **Self-Organizing Map** (**SOM**) is an artificial neural network, trained according to the unsupervised learning paradigm
  - The topology of the network lies typically in a low dimensional hyperspace:
    - 1D Lines: e.g., 5 cells



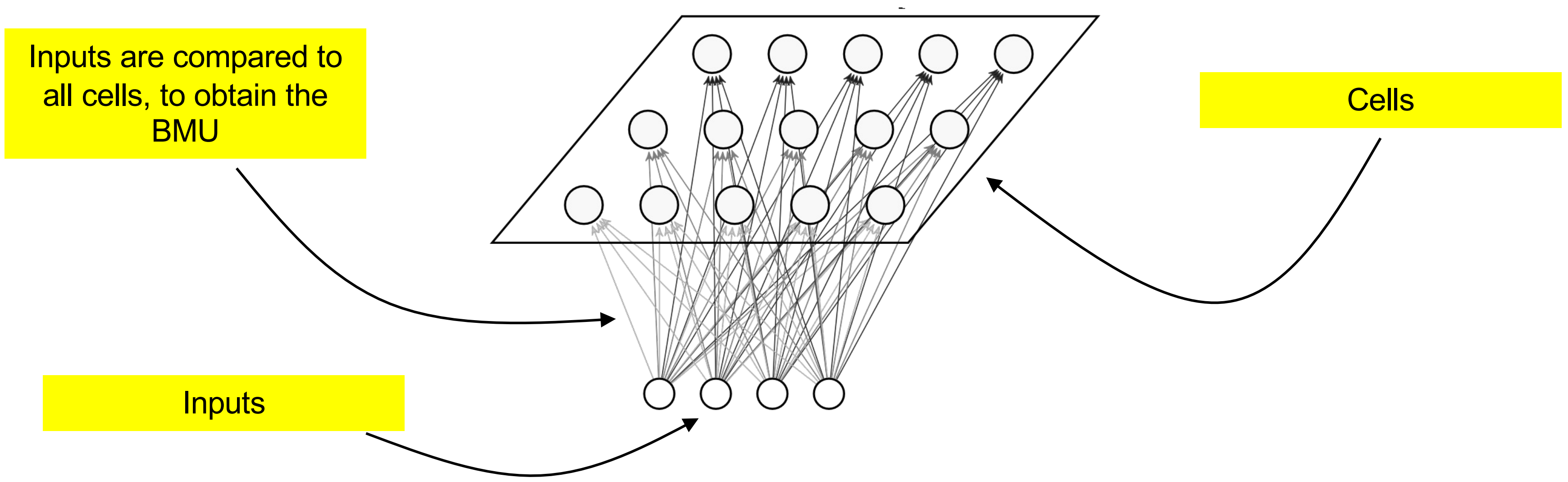
    - 2D Grids: e.g., 3 x 5 cells



    - 3D Spaces: ...

Unsupervised Learning: SOMs

- The cells of SOMs form the set of prototypes (clusters), and are autonomously inferred, according to a competitive learning paradigm
  - Having a set of input instances (learning set), each one lying in a (typically) high hyperspace, the idea is:
    - To find the SOM prototype that best represents each instance and "*assign the prototype to it*".
      - Called the **BMU** (Best Matching Unit)
    - Move (organize) the cells in the SOM, such that the BMU and its neighbors adjust their configurations (weights) towards the input instance
- Conceptually, there are only two entities in SOMS:



Unsupervised Learning: SOMs

- **SOM Learning Algorithm**
    1. Initialize the weights of the cells (c_i) randomly
    2. While ! Stop_criterium()
        1. For each input instance $\boldsymbol{x}^j \in \mathcal{R}^n$
            1. For each cell c_i, get the distance between x^j and c_i

$$d(\boldsymbol{x}^j,\mathbf{c}_i)$$

e.g., Euclidean Distance

2. Find the BMU

$$\mathscr{b} = \arg\min_i d(\boldsymbol{x}^j, \mathbf{c}_i)$$

3. Update the weights of the BMU cell:

Learning Rate

$$\boldsymbol{\theta} = \boldsymbol{\theta} + \gamma\,(\boldsymbol{\theta} - \boldsymbol{x})$$

5. Update the weights of the neighbour cells (c^k):

$$\boldsymbol{\theta} = \boldsymbol{\theta} + \gamma \exp\left(-\frac{d(\boldsymbol{c}_k,\boldsymbol{c}_{\mathscr{b}})}{\sigma}\right)(\boldsymbol{\theta} - \boldsymbol{x})$$

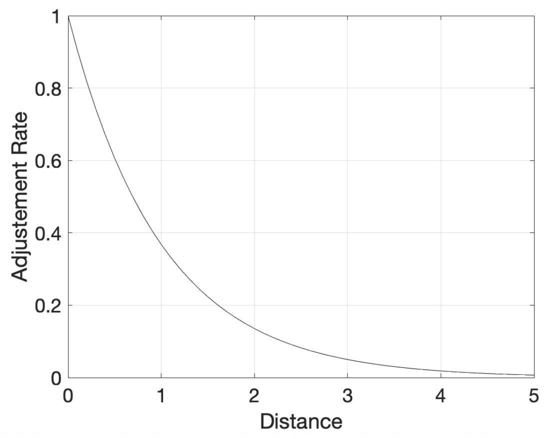
Neighborhood Constant

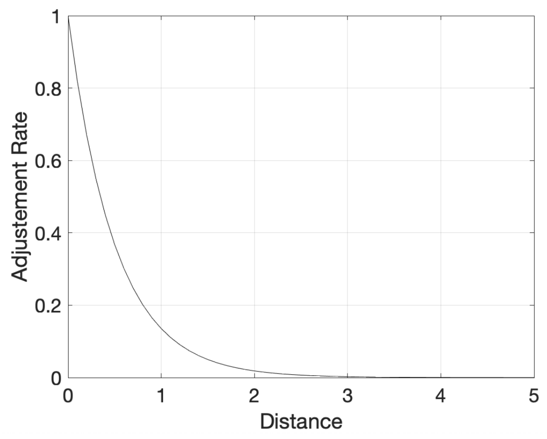
Unsupervised Learning: SOMs

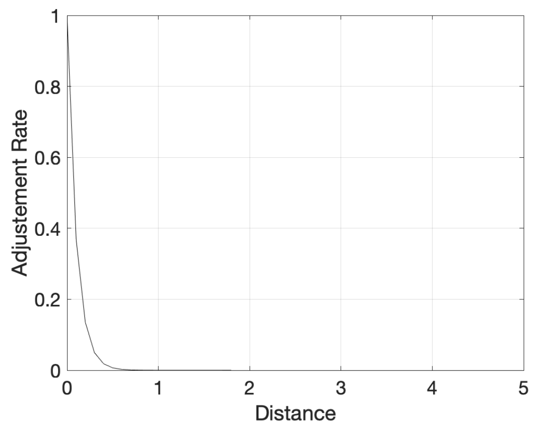
- The shape of the neighbood function determines how much (and how many) cells adjacent to the BMU are updated.
  - If the value is high, each BMU will imply movements in a large number of neighbors. The manifold will be “smooth”.
  - If the value is low, there is a short number of movements for each BMU, and the resulting manifold will be “peaked”.

Many cells updated

Few cells updated







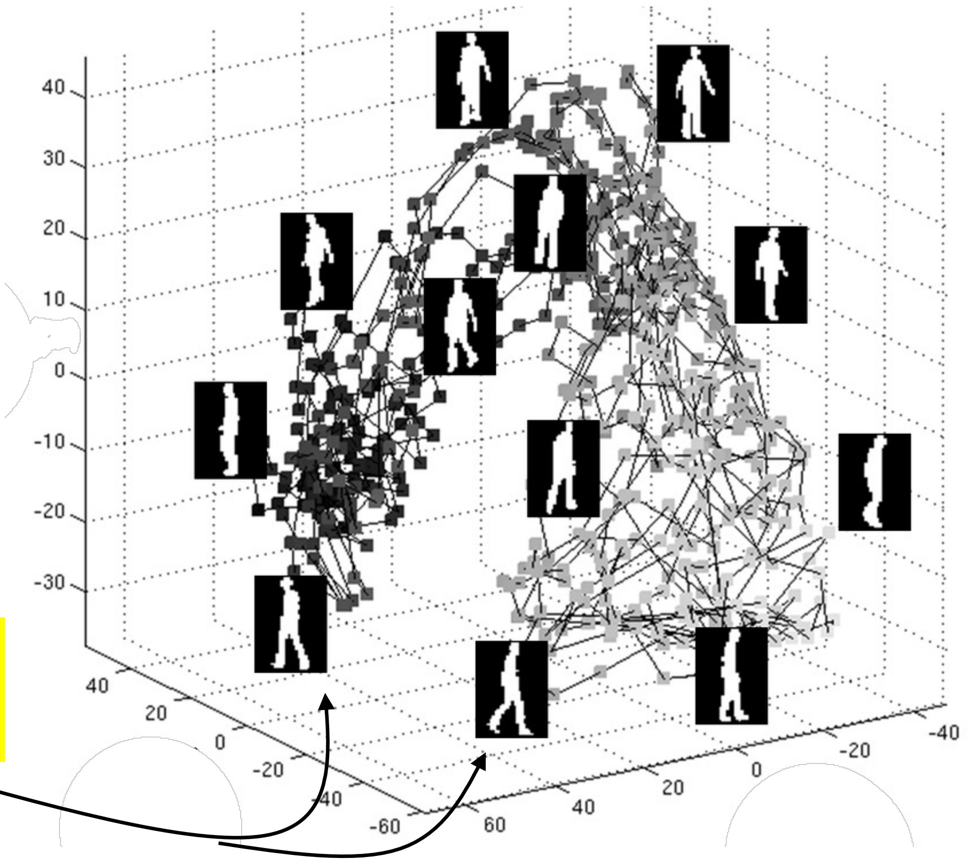
$\sigma = 1$

$\sigma = 0.5$

$\sigma = 0.1$

Unsupervised Learning: SOMs

- **More than just clustering, SOMs** build a low dimensional manifold, where high dimensional input samples are projected.
    - The relevant advantage of manifolds is the topological space property, i.e., neighbor elements "look-alike", in opposition to elements that are farther in the destiny-space.



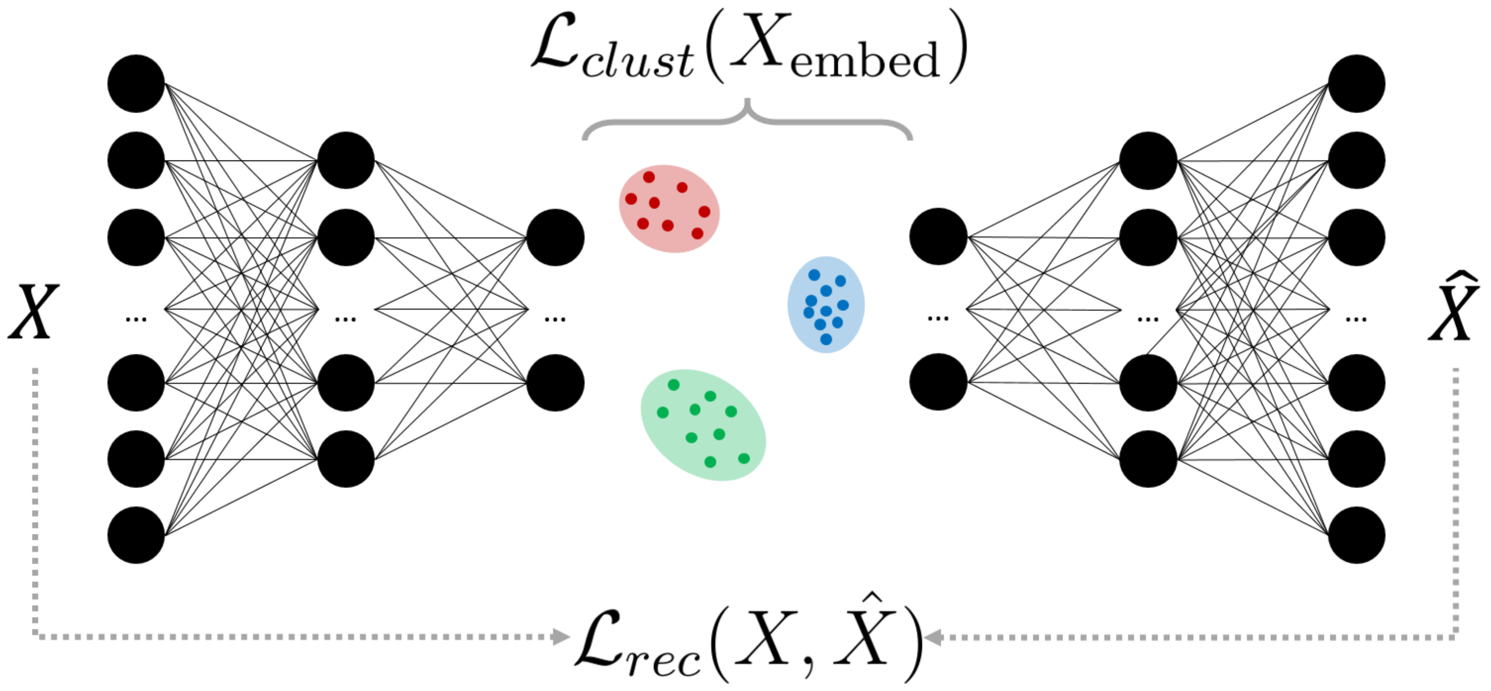
Close in terms of distance, but **very far** in the manifold

Machine Learning: SOM Exercise

- Consider the "AR.tar" dataset, available at the course web page.
  - It contains 3.315 [48 x 64] face images
  - We will use it to distinguish between "Male" and "Female" genders
- Implement a "Python" that:
  - Loads the set of images
  - Divide the set into two disjoint parts: "learning" and "test"
    - 90% for learning, 10% for test, randmly chosen
  - Builds a **SOM** manifold
- Check which SOM topology is more appropriate for distinguishing between...
  - Identities
  - Gender

Deep Clustering

- The basic idea of Deep Clustering is to obtain a compact representation that simoultaneously:
  - Is able to reconstruct the original data $\mathcal{L}_{rec}(X, \hat{X})$
  - Separates the dataset into homogenous groups $\mathcal{L}_{clust}(X_{\text{embed}})$


$\mathcal{L}_{clust}(X_{\text{embed}})$
X
...
$\hat{X}$
$\mathcal{L}_{rec}(X, \hat{X})$

Deep Clustering

Alternating iteratively optimization between clustering and autoencoder:

1) K-Means
2) Autoencoder Step (Reconstruction + Compression losses)

$$\mathcal{L}_{rec}(X, \hat{X}) = |\hat{x}_i - x_i|_2$$

Distance between input element (x_i) and its reconstruction ($\hat{x}_i$)

$$\mathcal{L}_{clust}(X_{\text{embed}}) = |z_i - \mu_i|_2$$

Distance between each latent code (z_i) and the centroid assigned by K-Means (μ_i)