Artificial Learning Models

Lecture 07 : Clustering (K-Means)

By : Dr. Lamri SAYAD

2023

Agenda

- Unsupervised Learning
- Clustering
- Clustering techniques
  - K-Means

ML tasks:
Reminder

- Supervised Learning
  - Classification
  - Regression
- Unsupervised Learning
  - Clustering
  - Association
  - Dimensionality reduction
- Reinforcement Learning

Unsupervised Learning vs Supervised Learning

- **Supervised learning** is where you have input variables (x) and an output variable (Y) and you use an algorithm to learn the mapping function from the input to the output.

$$Y = f(X)$$

- *The goal* is to approximate the mapping function so well that when you have new input data (x) that you can predict the output variables (Y) for that data
- **Unsupervised learning** is where you only have input data (X) and no corresponding output variables
- *The goal* for unsupervised learning is to model the underlying structure or distribution in the data in order to learn more about the data.
- Unsupervised learning problems can be further grouped into clustering and association problems.
  - **Clustering**
  - **Association**

Unsupervised Learning

- is a paradigm in machine learning
- Learn patterns exclusively from unlabeled data.
- looks for patterns in a dataset without pre-existing labels
- users do not need to supervise the model.
- requires little human supervision

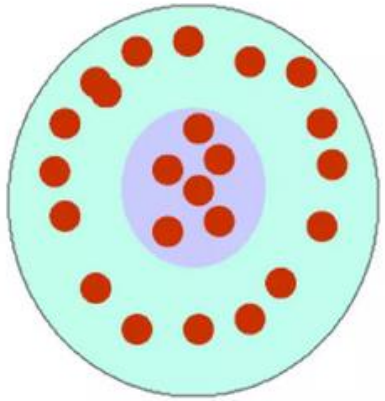
Why Unsupervised Learning?

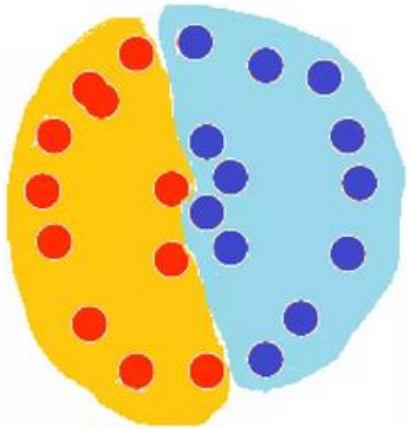
- Unsupervised machine learning finds all kind of unknown patterns in data.
- Unsupervised methods help you to find features which can be useful for categorization.
- It is taken place in real time, so all the input data to be analyzed and labeled in the presence of learners.
- It is easier to get unlabeled data from a computer than labeled data, which needs manual intervention.

Clustering

What is clustering?

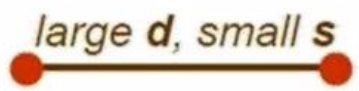
- The organization of unlabeled data into similarity groups called clusters.
- A cluster is a collection of data items which are “similar” between them, and “dissimilar” to data items in other clusters.

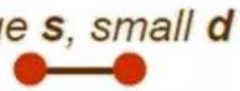




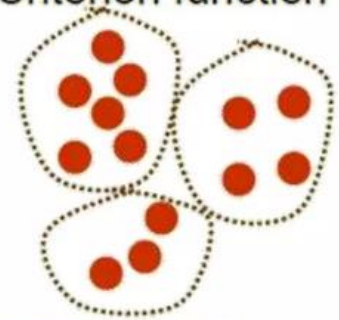
What do we need for clustering?

1. Proximity measure, *either*
   - similarity measure $s(x_i,x_k)$: large if x_i,x_k are similar
   - dissimilarity(or distance) measure $d(x_i,x_k)$: small if x_i,x_k are similar

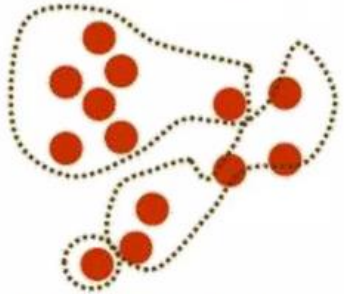




2. Criterion function to evaluate a clustering



good clustering



bad clustering

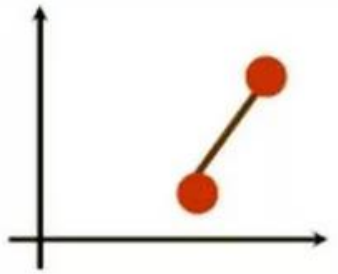
3. Algorithm to compute clustering
   - For example, by optimizing the criterion function

Distance (dissimilarity) measures

- Euclidean distance between points ***i*** and ***j*** is the length of the line segment connecting them
- In Cartesian coordinates, if $\mathbf{i} = (i_1, i_2, \ldots i_n)$ and $\mathbf{q} = (q_1, q_2, \ldots q_n)$ then the distance (**d**) from ***i*** to ***j***, or from ***j*** to ***i*** is given by:

- Euclidean distance

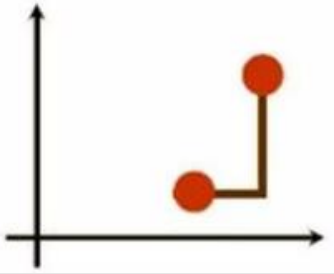
$$d(x_i, x_j) = \sqrt{\sum_{k=1}^{d} \left(x_i^{(k)} - x_j^{(k)}\right)^2}$$



- Manhattan (city block) distance

$$d(x_i, x_j) = \sum_{k=1}^{d} \left|x_i^{(k)} - x_j^{(k)}\right|$$

  - approximation to Euclidean distance, cheaper to compute



Cluster evaluation

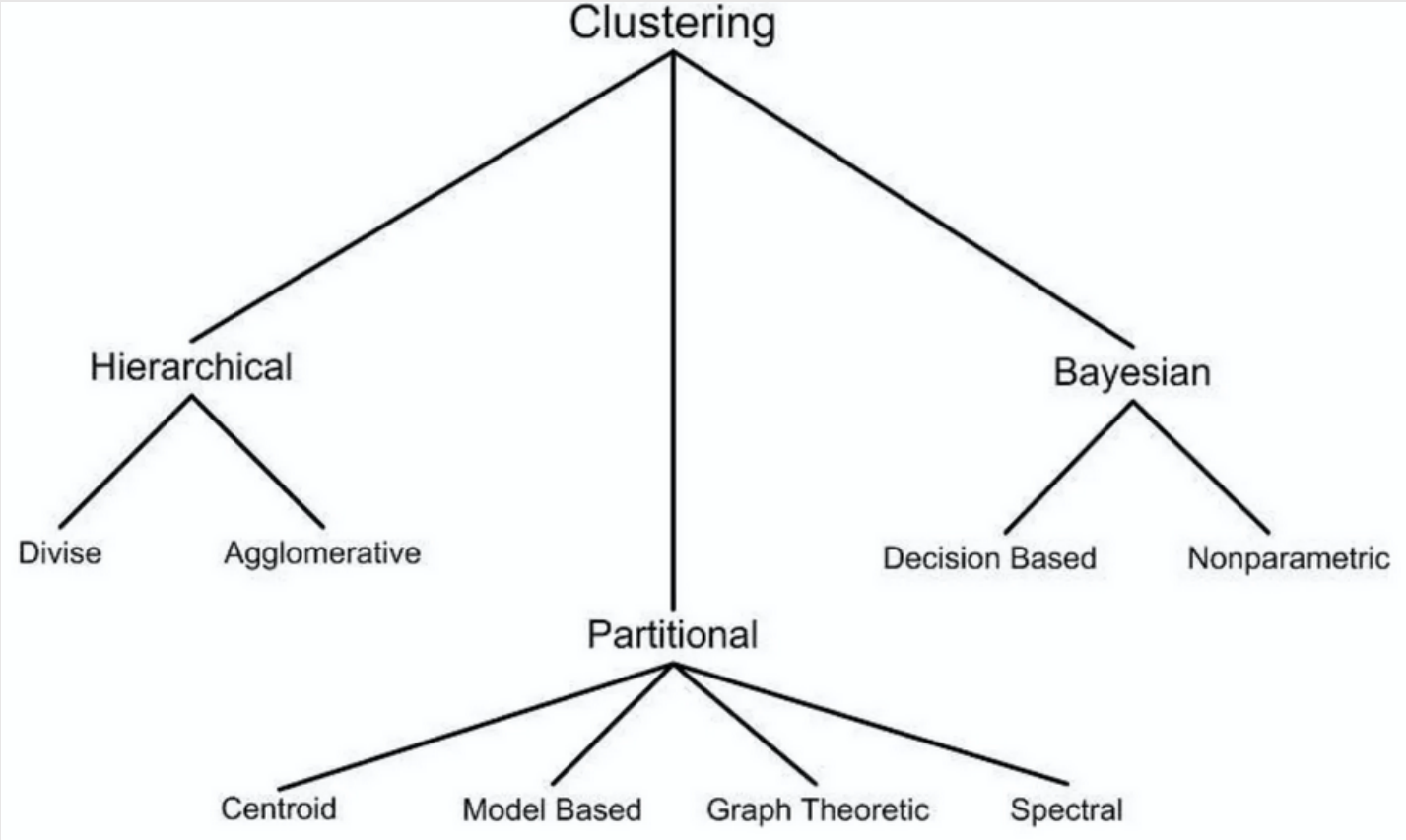
- **Intra-cluster cohesion** (compactness):
  - Cohesion measures how near the data points in a cluster are to the cluster centroid.
  - Sum of squared error (SSE) is a commonly used measure.
- **Inter-cluster separation** (isolation):
  - Separation means that different cluster centroids should be far away from one another.

How many clusters

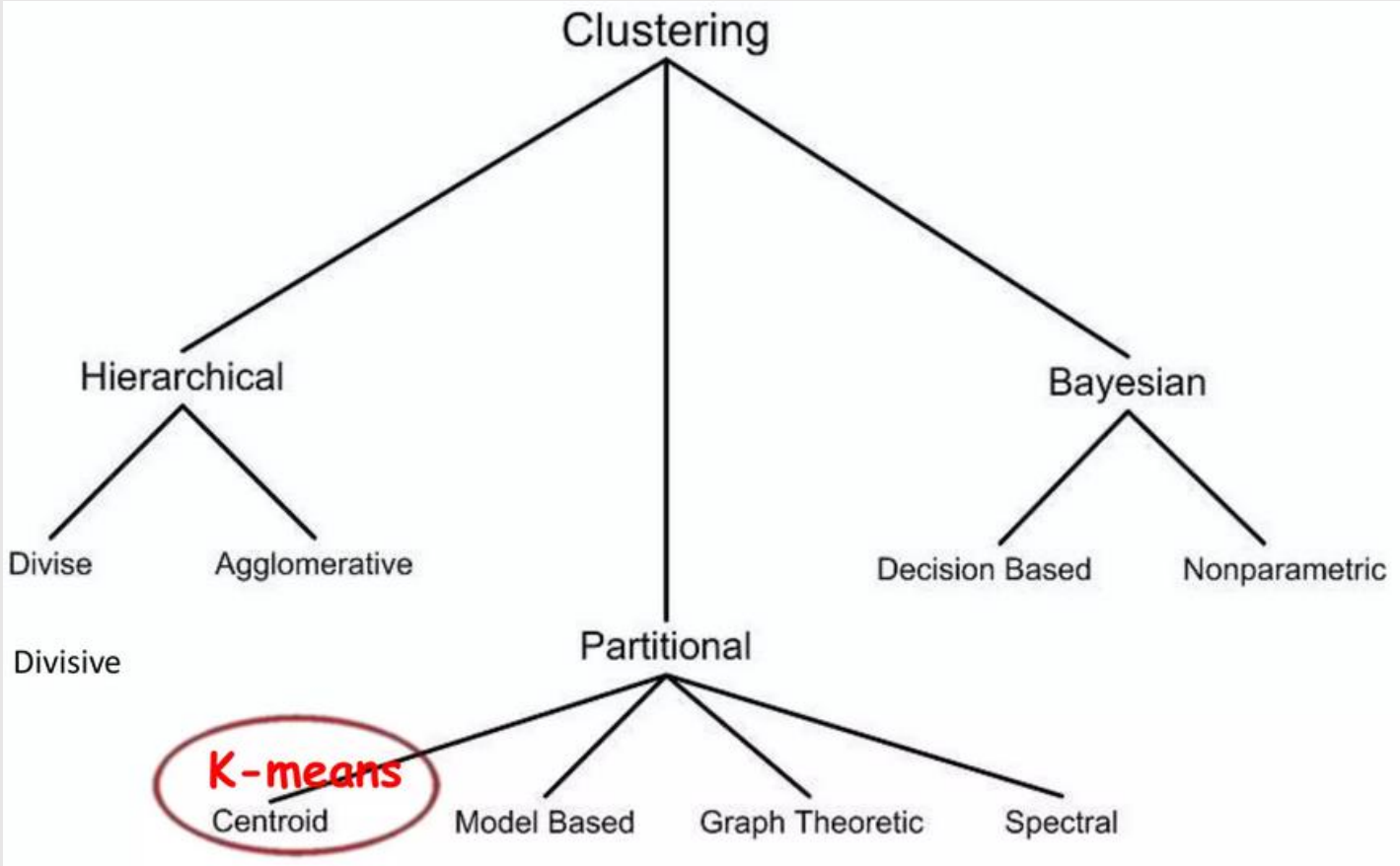
3 clusters or 2 clusters?

- Possible approaches
  1. fix the number of clusters to ***k***
  2. find the best clustering according to the criterion function (number of clusters may vary)

Clustering techniques



Clustering techniques



K-Means Clustering

- K-means (MacQueen, 1967) is a partitional clustering algorithm
- The *k*-means algorithm partitions the given data into *k* clusters:
  - Each cluster has a cluster **center**, called **centroid**.
  - *k* is specified by the user

K-Means questions

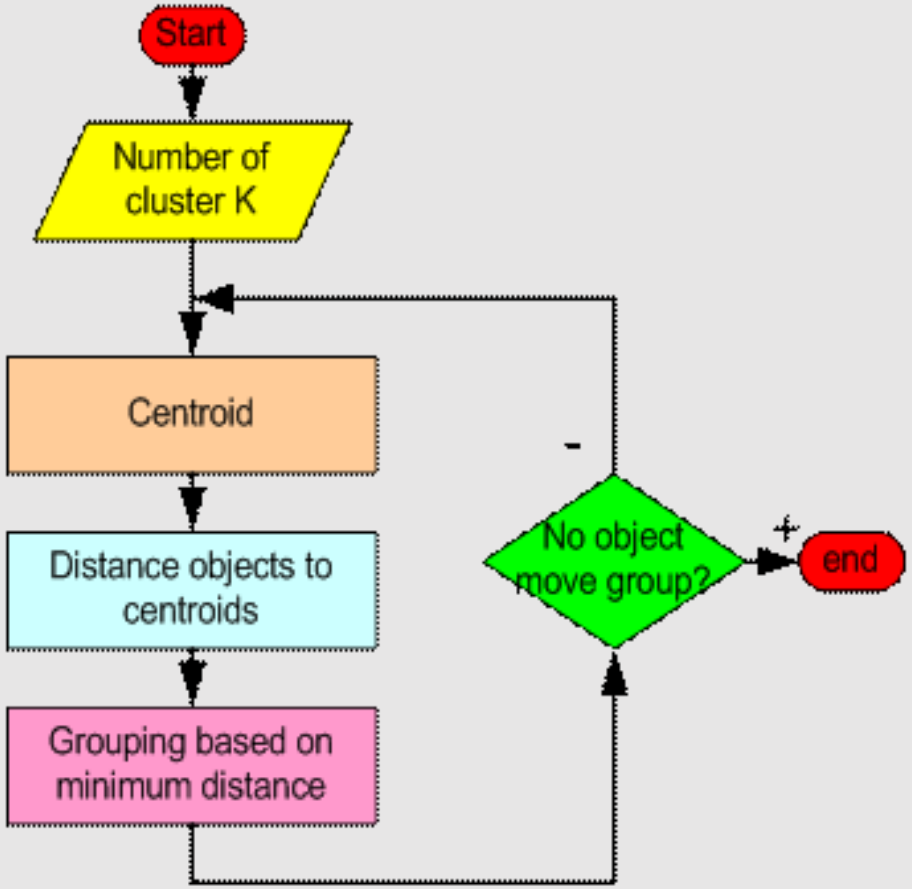
- What is it trying to optimize?
- Are we sure it will terminate?
- Are we sure it will find an optimal clustering?
- How should we start it?
- How could we automatically choose the number of centers?

….we'll deal with these questions over the next few slides

K-Means : Algorithm

- Actually how it works is really simple.
  - Randomly pick K centroids (k means)
  - Assign each data point to the centroid it's closest to
  - Recompute the centroids based on the average position of each centroid's points
  - Iterate until points stop changing assignment to centroids
- If you want to predict the cluster for new points, just find the centroid they're closest to.

K-Means : Algorithm



K-Means : Example

- Example 1
- Example 2
- Example 3

Example: K-Means for Segmentation

K=2

Goal of Segmentation is to partition an image into regions each of which has reasonably homogenous visual appearance.

Original

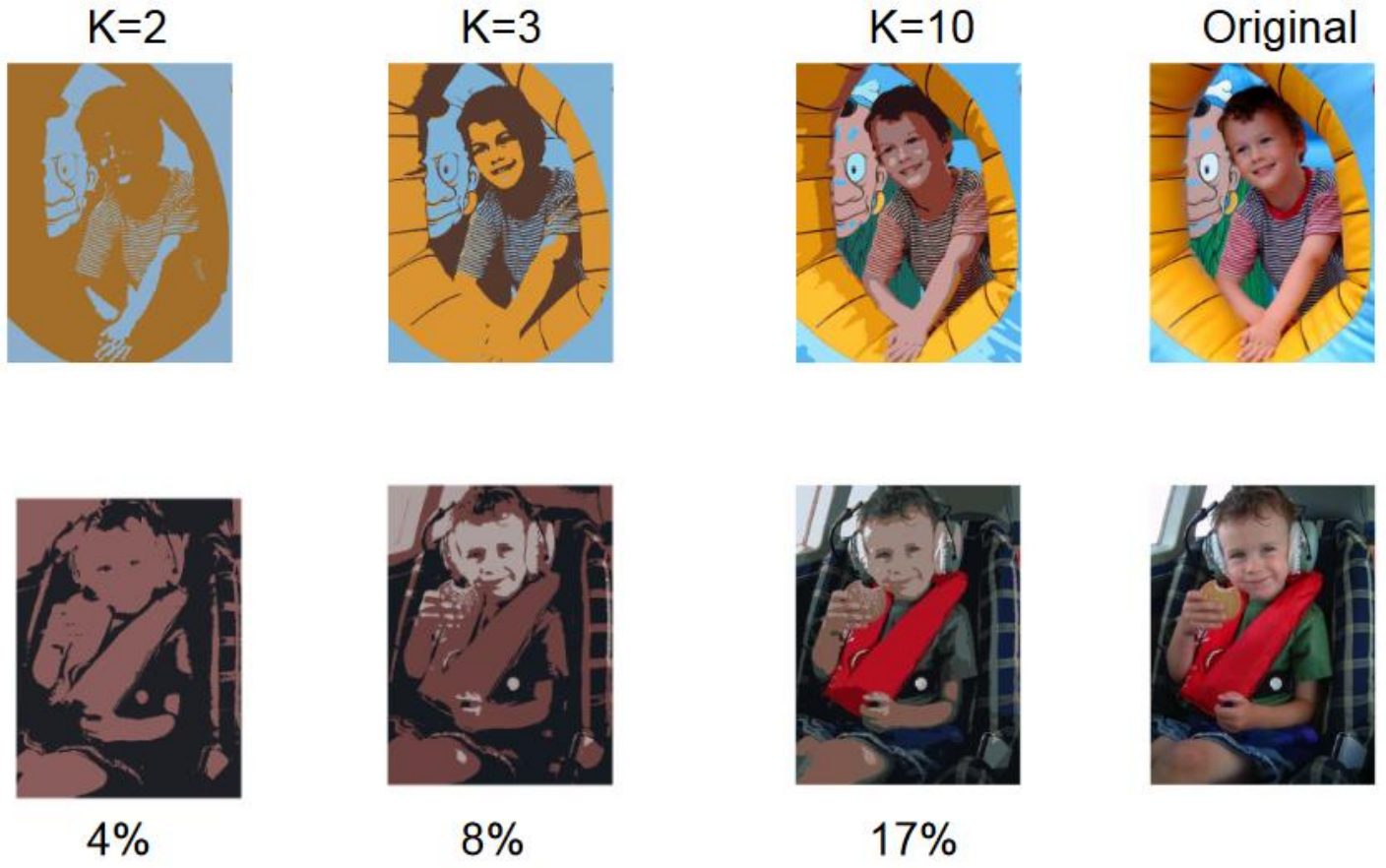
Example: K-Means for Segmentation

K=2

K=3

Original

Example: K-Means for Segmentation

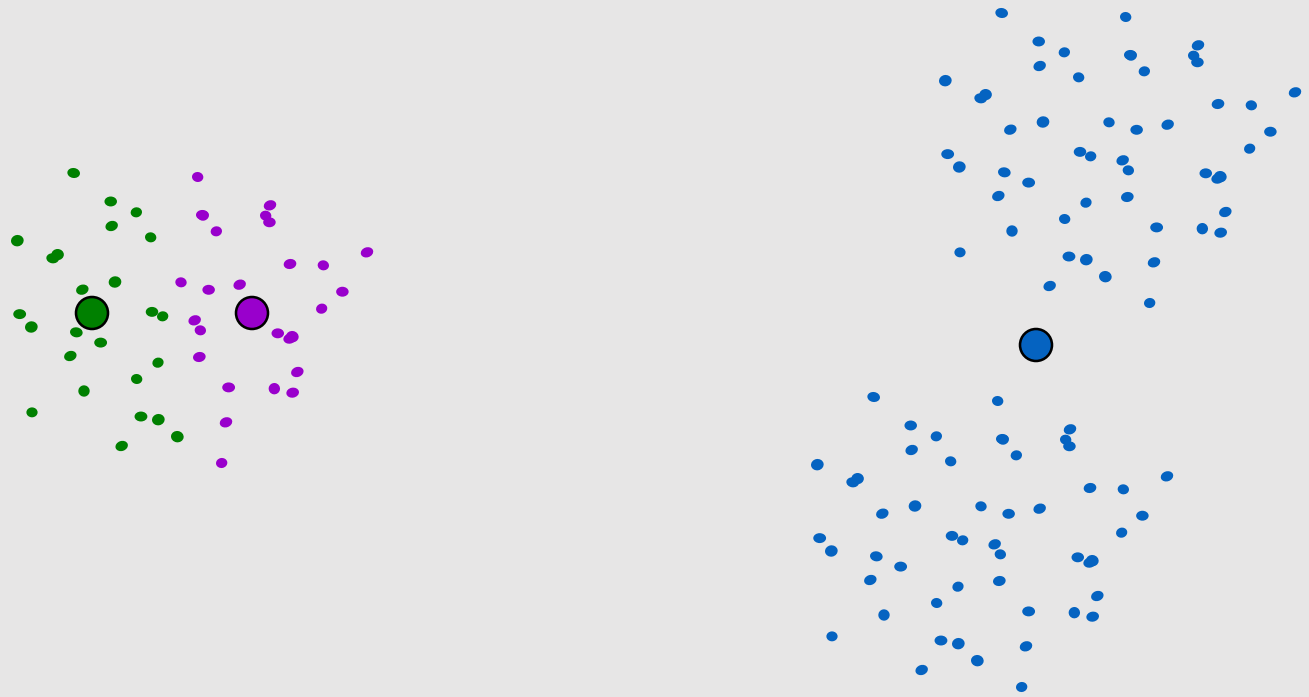


Will we find the optimal configuration?

- Not necessarily.
- Can you invent a configuration that has converged, but does not have the minimum distortion? (Hint: try a fiendish *k=3* configuration here…)
- Problem 1 : Local optimum

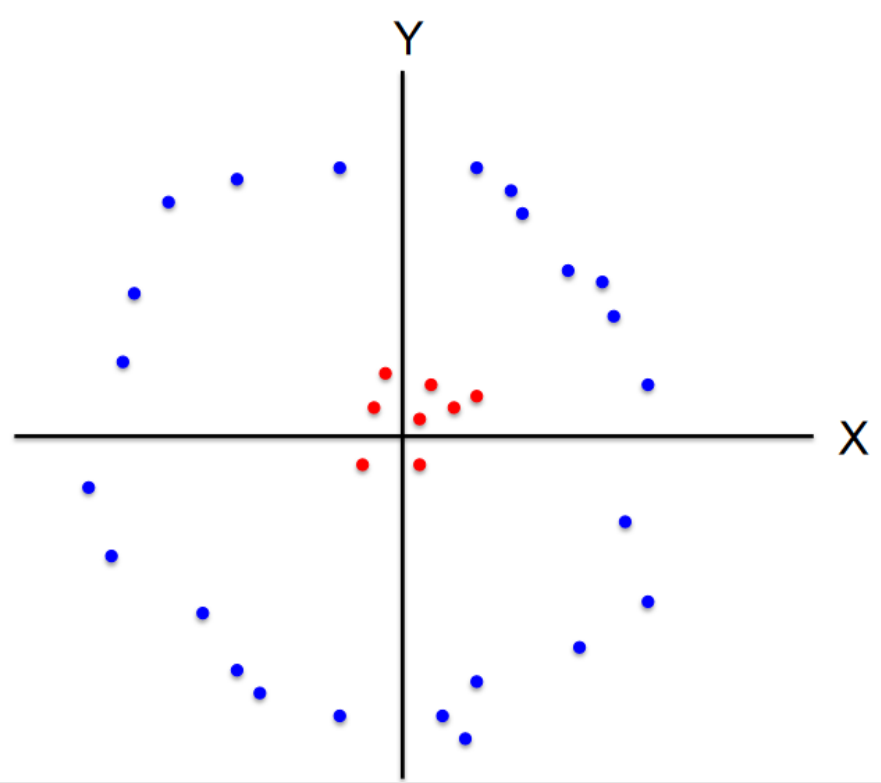
Will we find the optimal configuration?

- Not necessarily.
- Can you invent a configuration that has converged, but does not have the minimum distortion? (Hint: try a fiendish *k=3* configuration here…)



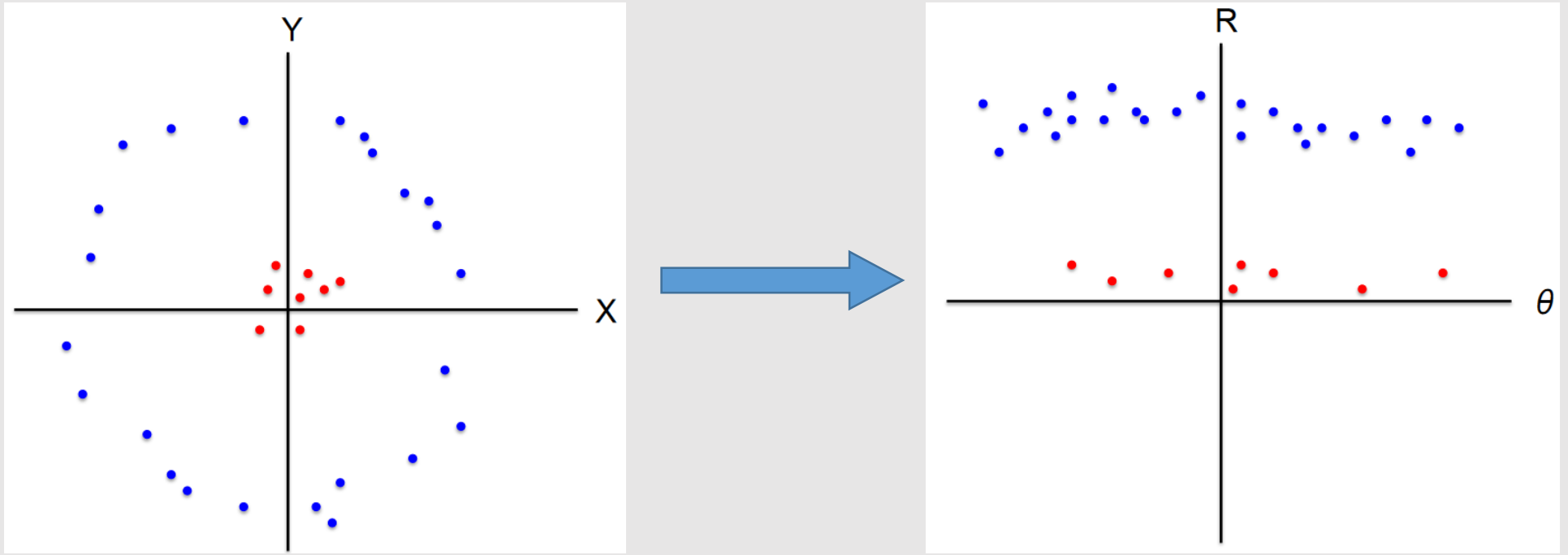
Will we find the optimal configuration?

- Problem 2 : K-means not able to properly cluster



Will we find the optimal configuration? Solutions

- Solution to Problem 1 : Iterate many times with different initial centroids positions
- Solution to Problem 1 : Changing the features (distance function)can help



Weaknesses of K-Mean Clustering

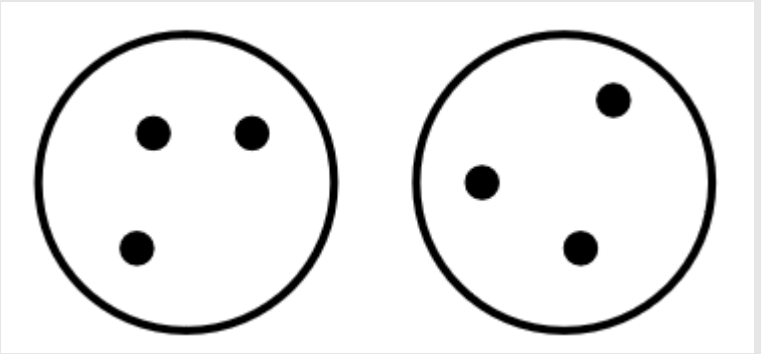
1. When the numbers of data are not so many, initial grouping will determine the cluster significantly (sensitive to initial clusers).
2. The number of clusters, K, must be determined before hand. Its disadvantage is that it does not yield the same result with each run, since the resulting clusters depend on the initial random assignments.
3. We never know the real cluster, using the same data, because if it is inputted in a different order it may produce different cluster if the number of data is few.
4. It is sensitive to initial condition. Different initial condition may produce different result of cluster. The algorithm may be trapped in the *local optimum*.

Agglomerative Hierarchical Clustering

- Idea:
  - First merge very similar instances
  - Incrementally build larger clusters out of smaller clusters
- Algorithm:
  - Maintain a set of clusters
  - Initially, each instance in its own cluster
  - Repeat:
    - Pick the two closest clusters
    - Merge them into a new cluster
    - Stop when there's only one cluster left
- Produces not one clustering, but a family of clusterings represented by a dendrogram

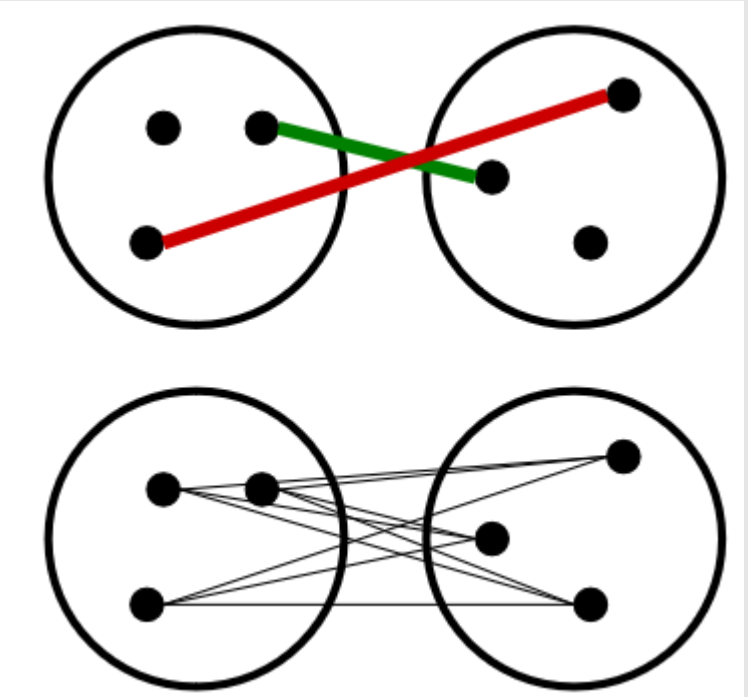
Agglomerative Hierarchical Clustering

- How should we define “closest” for clusters with multiple elements?



Agglomerative Hierarchical Clustering

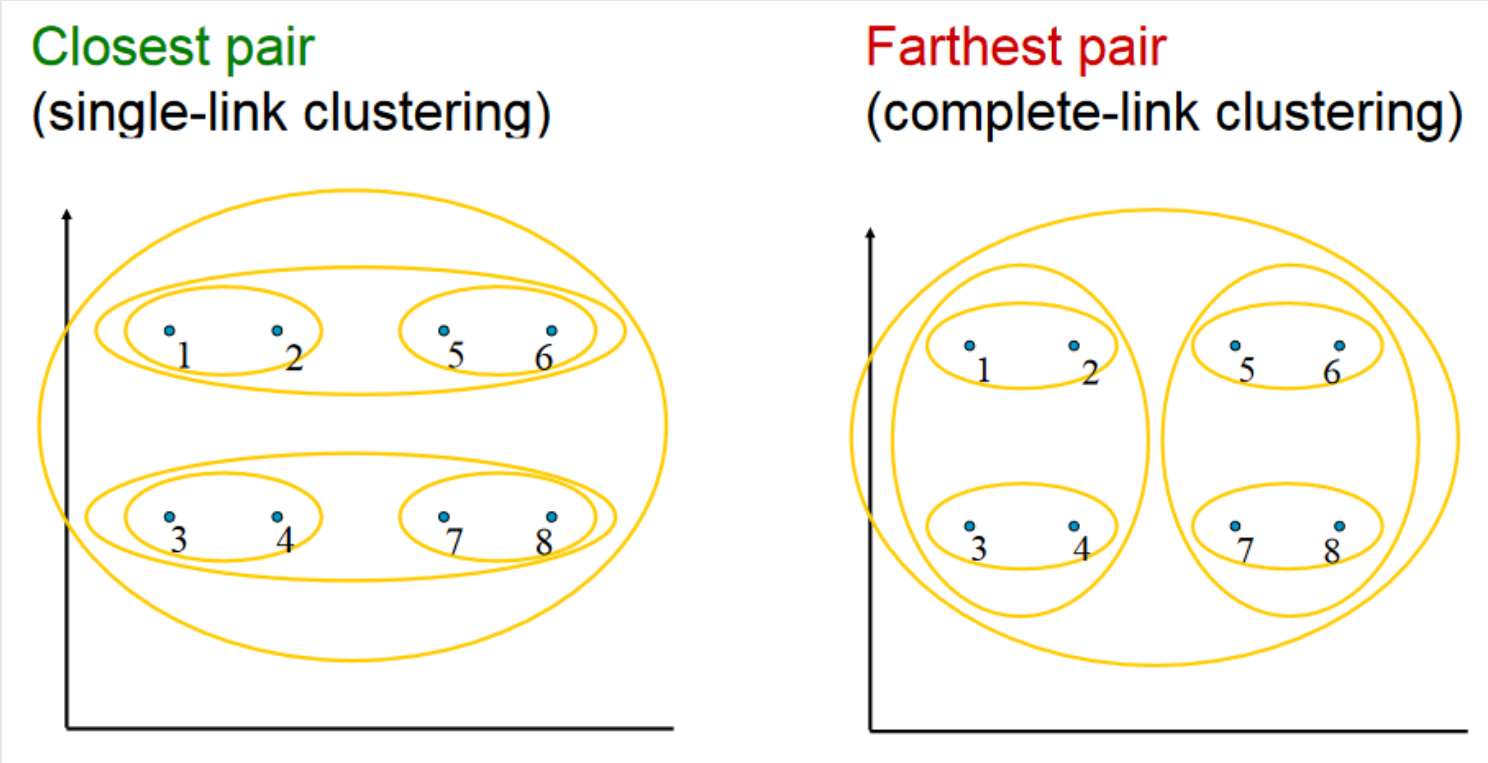
- How should we define “closest” for clusters with multiple elements?
- Many options:
  - Closest pair (single-link clustering)
  - Farthest pair (complete-link clustering)
  - Average of all pairs



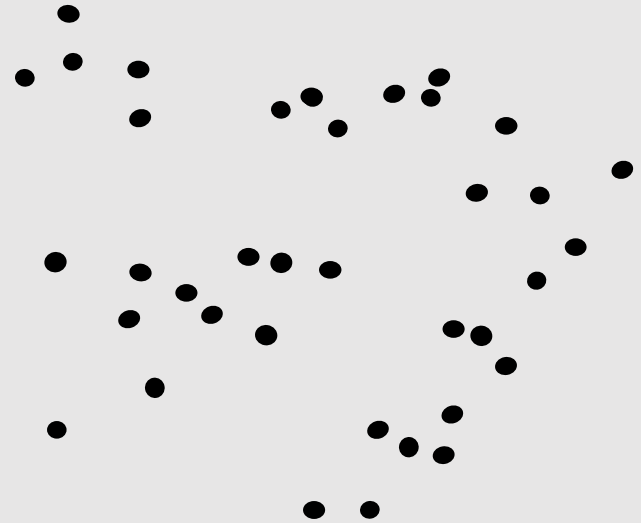
- Different choices create different clustering behaviors

Agglomerative Hierarchical Clustering

- How should we define “closest” for clusters with multiple elements?

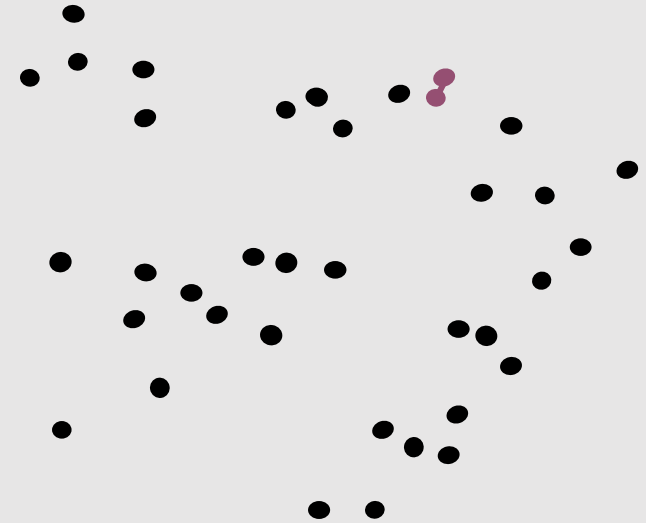


Agglomerative Hierarchical Clustering



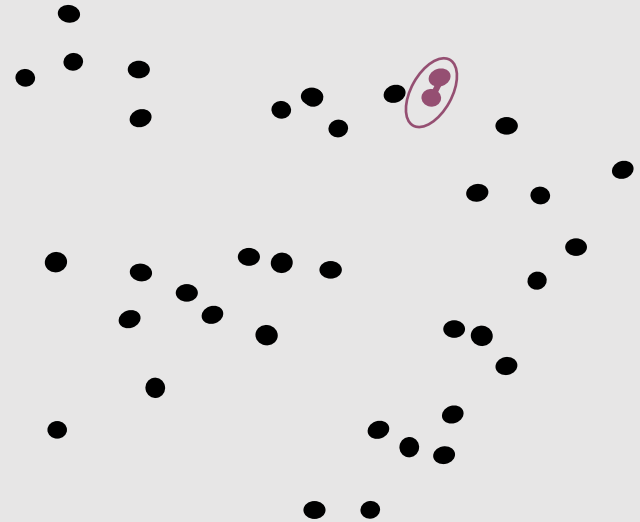
1. Say “Every point is its own cluster”

Agglomerative Hierarchical Clustering



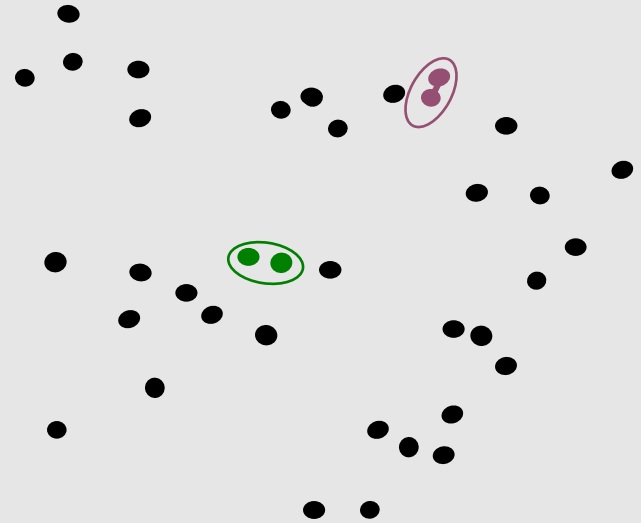
1. Say “Every point is its own cluster”
2. Find “most similar” pair of clusters

Agglomerative Hierarchical Clustering



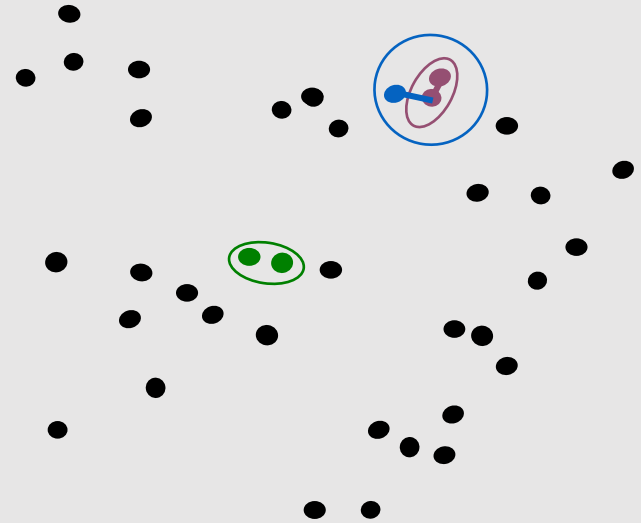
1. Say “Every point is its own cluster”
2. Find “most similar” pair of clusters
3. Merge it into a parent cluster

Agglomerative Hierarchical Clustering

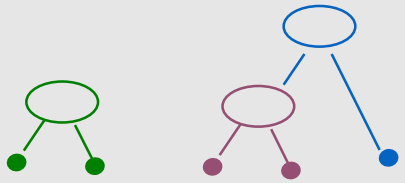


1. Say "Every point is its own cluster"
2. Find "most similar" pair of clusters
3. Merge it into a parent cluster
4. Repeat

Agglomerative Hierarchical Clustering



1. Say "Every point is its own cluster"
2. Find "most similar" pair of clusters
3. Merge it into a parent cluster
4. Repeat



Density-Based Clustering Methods

- Clustering based on density (local cluster criterion), such as density-connected points
  - Each cluster has a considerable higher density of points than outside of the cluster
- Several interesting studies:
  - DBSCAN: Ester, et al. (KDD'96)
  - OPTICS: Ankerst, et al (SIGMOD'99).
  - DENCLUE: Hinneburg & D. Keim (KDD'98)
  - CLIQUE: Agrawal, et al. (SIGMOD'98)
- Major features:
  - Discover clusters of arbitrary shape
  - Handle noise
  - One scan
  - Need density parameters as termination condition

DBSCAN

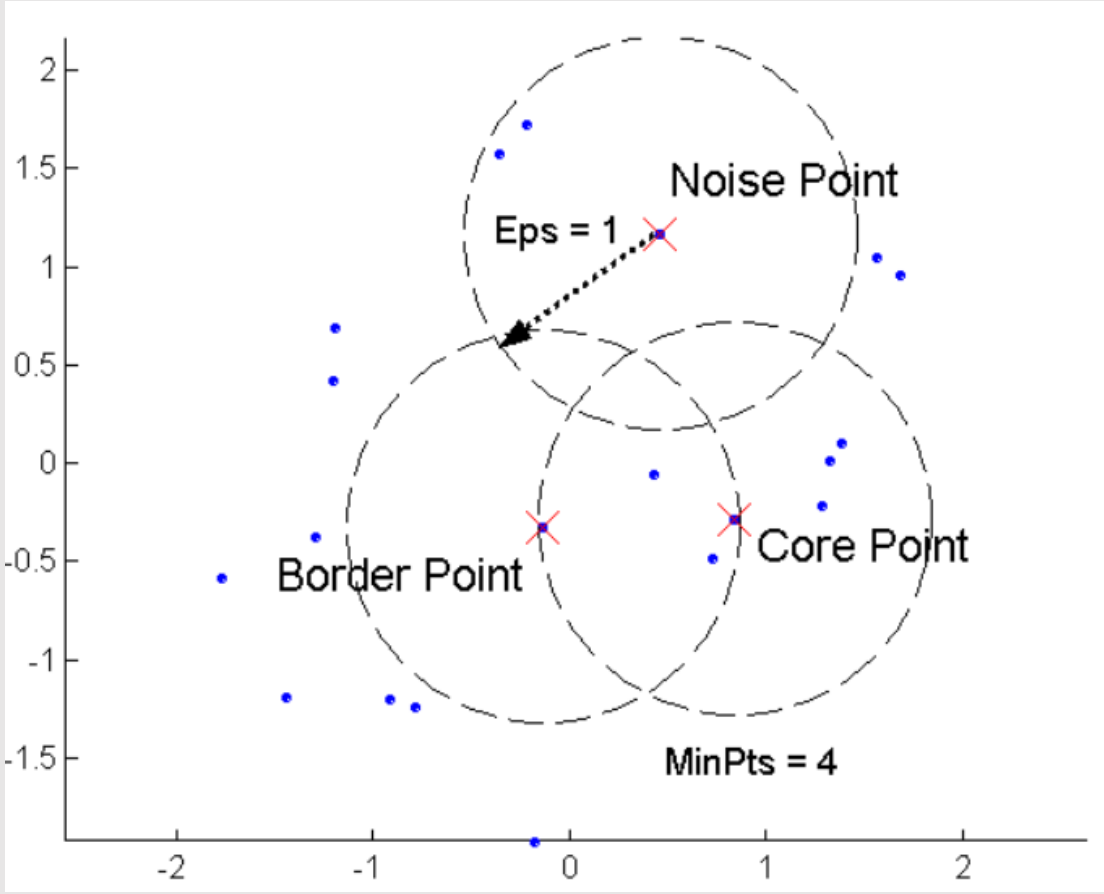
DBSCAN is a density-based algorithm.

- Density = number of points within a specified radius r (Eps)
- A point is a core point if it has more than a specified number of points (MinPts) within Eps

These are points that are at the interior of a cluster

- A border point has fewer than MinPts within Eps, but is in the neighborhood of a core point
- A noise point is any point that is not a core point or a border point.

DBSCAN:Core, Border, and Noise points

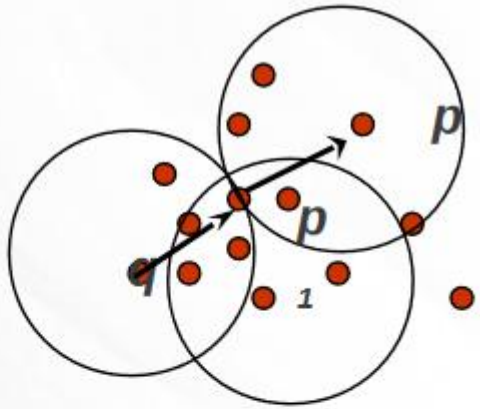


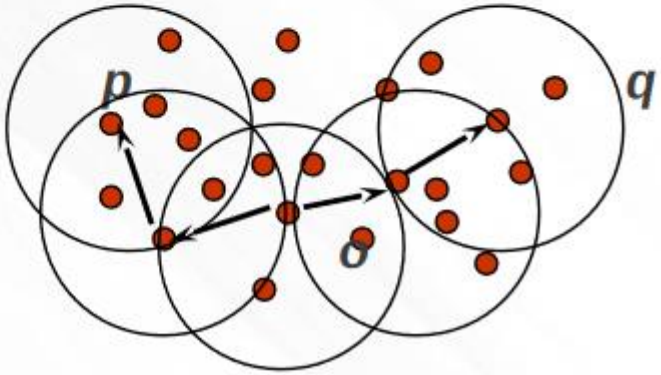
DBSCAN: Concepts

- Any two core points are close enough (within a distance *Eps* of one another) are put in the same cluster
- Any border point that is close enough to a core point is put in the same cluster as the core point
- Noise points are discarded

Density-Reachable and Density-Connected (w.r.t. Eps, MinPts)

- Let p be a core point, then every point in its *Eps* neighborhood is said to be directly density-reachable from p.
- A point *p* is density-reachable from a point core point *q* if there is a chain of points $p_1, \ldots, p_n, p_1 = q, p_n = p$
- 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

Two parameters (eps and MinPts)***:***

- ***ε***: Maximum radius of the neighbourhood
- ***MinPts***: Minimum number of points in an Eps-neighbourhood of that point
- $N_{\varepsilon}(p)$: ***{q belongs to D | dist(p,q) <= ε}***

Directly density-reachable**:** A point ***p*** is directly density-reachable from a point ***q*** wrt. ***ε***, ***MinPts*** if

1) ***p*** belongs to $N_{\varepsilon}(q)$

2) core point condition:

$|N_{\varepsilon}(q)| >= MinPts$

DBSCAN: Algorithm

Let ClusterCount=0. For every point *p*:

1. If *p* it is not a core point, assign a null label to it [e.g., zero]
2. If *p* is a core point, a new cluster is formed [with label ClusterCount:= ClusterCount+1]

   Then find all points density-reachable from p and classify them in the cluster.
   [Reassign the zero labels but not the others]

Repeat this process until all of the points have been visited.

Since all the zero labels of border points have been reassigned in 2, the remaining points with zero label are noise.