Unit 4

Cluster analysis

Classification:

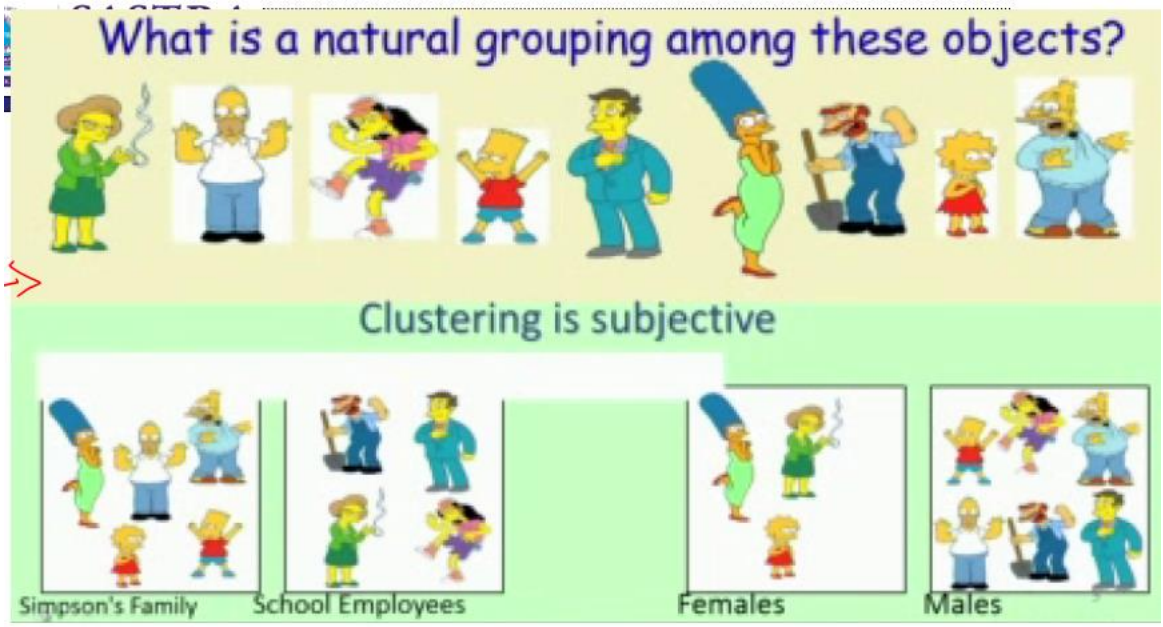
- Supervised- the number of classes are fixed beforehand and you have a training data which for a given input tells you that this would be the class and you train on that.
- Unsupervised – The class labels are not fixed and classification is done based on similarity between data- clustering
- in some of the application it may be the case that we need unsupervised learning or exploratory data analysis.
- *For example, -customers in a supermarket say big bazaar or something and -find group of customers I do not know beforehand what are the groups.*
- *But looking at the behaviour of the customers - find out the groups. And there are many other applications for example, I want to group the images in internet into similar things.*
- learning by observations (unsupervised) vs. learning by examples (supervised)

Clustering: Application Examples

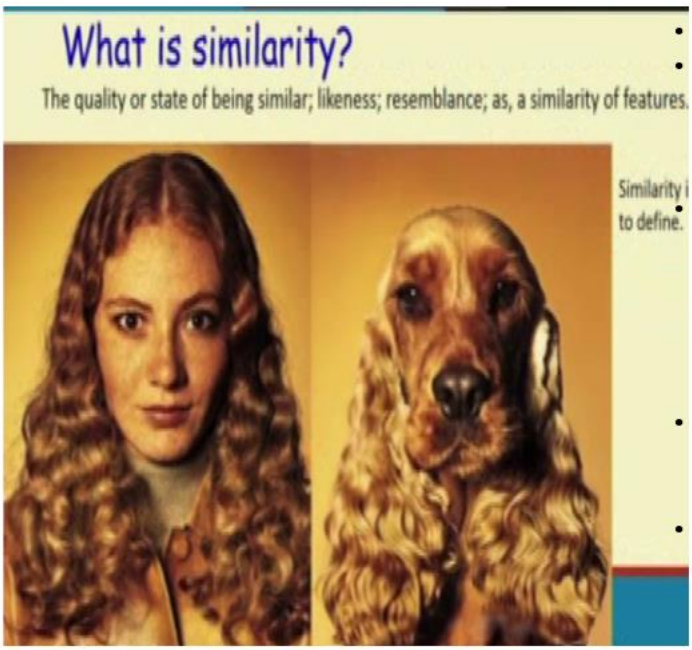
- **Biology**: taxonomy of living things: kingdom, phylum, class, order, family, genus and species
- **Information retrieval**: document clustering
- **Land use**: Identification of areas of similar land use in an earth observation database
- **Marketing**: Help marketers discover distinct groups in their customer bases, and then use this knowledge to develop targeted marketing programs
- **City-planning**: Identifying groups of houses according to their house type, value, and geographical location
- **Earth-quake studies**: Observed earth quake epicenters should be clustered along continent faults
- **Climate**: understanding earth climate, find patterns of atmospheric and ocean
- **Economic Science**: market research

- So, the definition of clustering is you have to organize the data the points, the examples into groups which are homogeneous.
- Homogeneous means if you take points from the same group they are close whereas, you want to find if you take the points from different groups they are apart.

- So, you want to maximize intra class similarity and minimize inter class similarity among the points.
- So, this actually amounts to finding the natural groups in the data

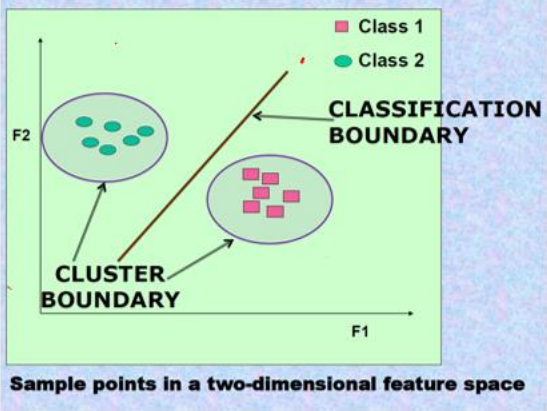


group your data into similar objects.
So, the terms similar you have to first define before you define what is a group of similar object.
So, that this similarity measure defines what is a natural group.



- naturally it will depend on the
- attributes of the or the features of the objects between which you find in similarity.
- For example, here if you are looking at the texture of the images or the colour of the images or the position of the eyes in the images they are similar.
- But actually in there is some other aspect in which they are not similar.
- So, it is a difficult task to define similarity

- Finding groups of similar objects



Similarity ≡ inverse of distance between two points

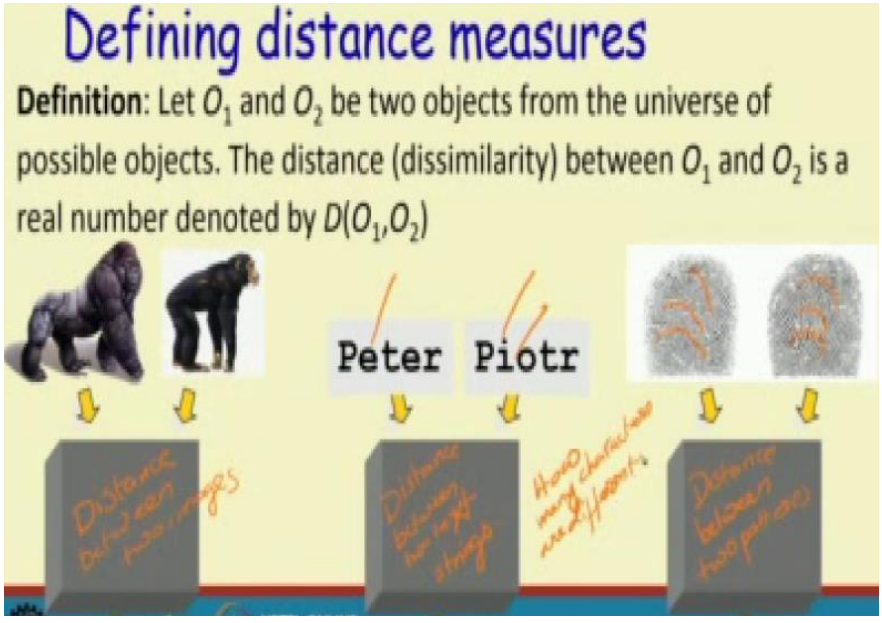
Distance ≡ disimilarity

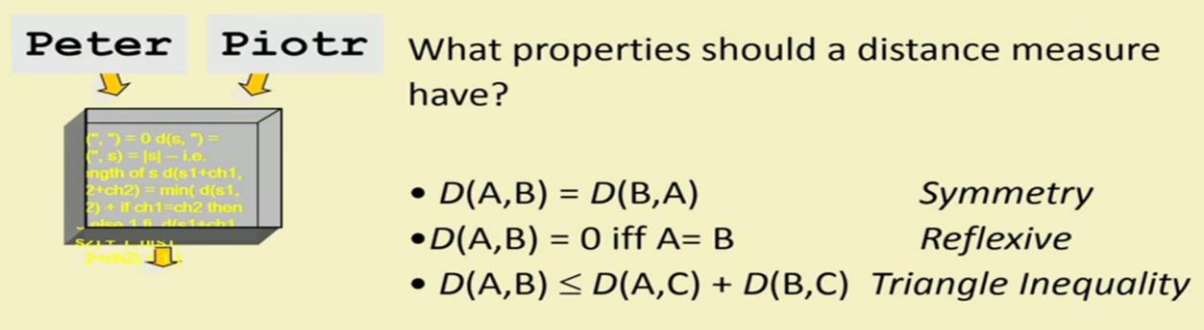
Minimizes intracluster distances
Maximizes intercluster distances

Measurement – how good the clustering is : Scatter coefficient
Scatter coefficient=Avg. intra cluster distance / avg. inter cluster distance

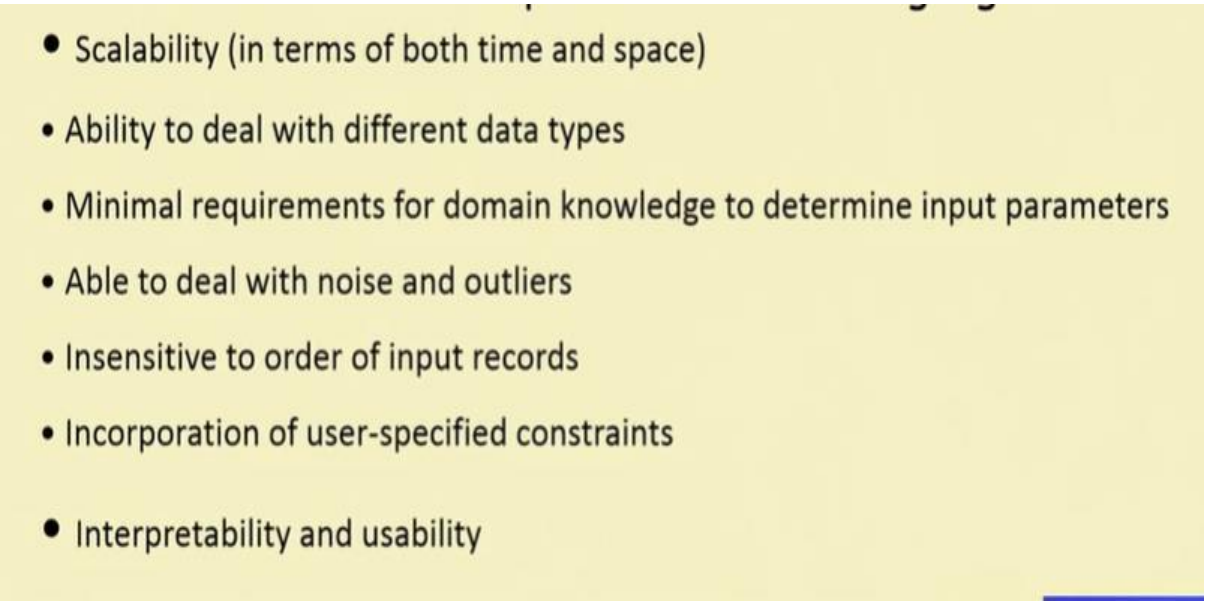
Lower the scatter- better is the clustering

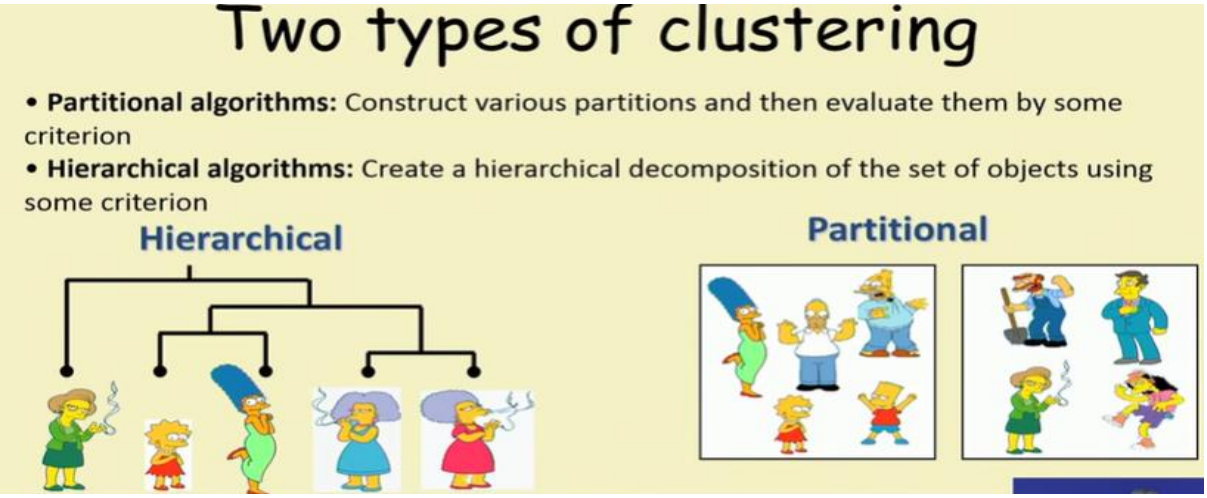
within cluster = 5 ↓2
between cluster = 2 ↑5
S.C = 5/2 = 2.5
S.C = 2/5 = 0.4





Properties of algorithm:





Partitioned clustering:

- Partition: is a division or grouping points into groups such that
  - Every point belongs to a group
  - Every point belongs to only one group
- Many different partitions are made

- Evaluates each partition and compares them based on how ‘good’ the clustering is

What is good clustering?

- A good clustering method will produce high quality clusters with
  - high intra-class similarity
  - low inter-class similarity
- The quality of a clustering result depends on both the similarity measure used by the method and its implementation.
- The quality of a clustering method is also measured by its ability to discover some or all of the hidden patterns.

Similarity and Dissimilarity Between Objects

- Distances are normally used to measure the similarity or dissimilarity between two data objects

Manhattan distance

$$d(i,j)=|x_{i_1}-x_{j_1}|+|x_{i_2}-x_{j_2}|+...+|x_{i_p}-x_{j_p}|$$

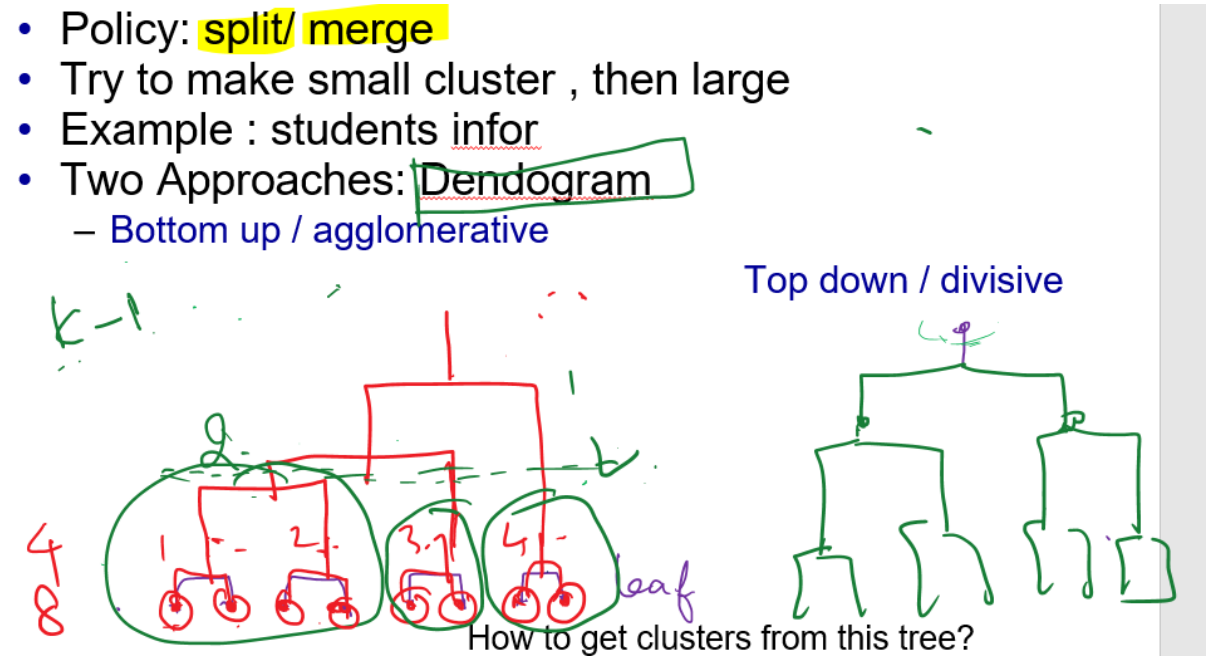
Euclidean distance:

$$d(i,j)=\sqrt{(|x_{i_1}-x_{j_1}|^2+|x_{i_2}-x_{j_2}|^2+...+|x_{i_p}-x_{j_p}|^2)}$$

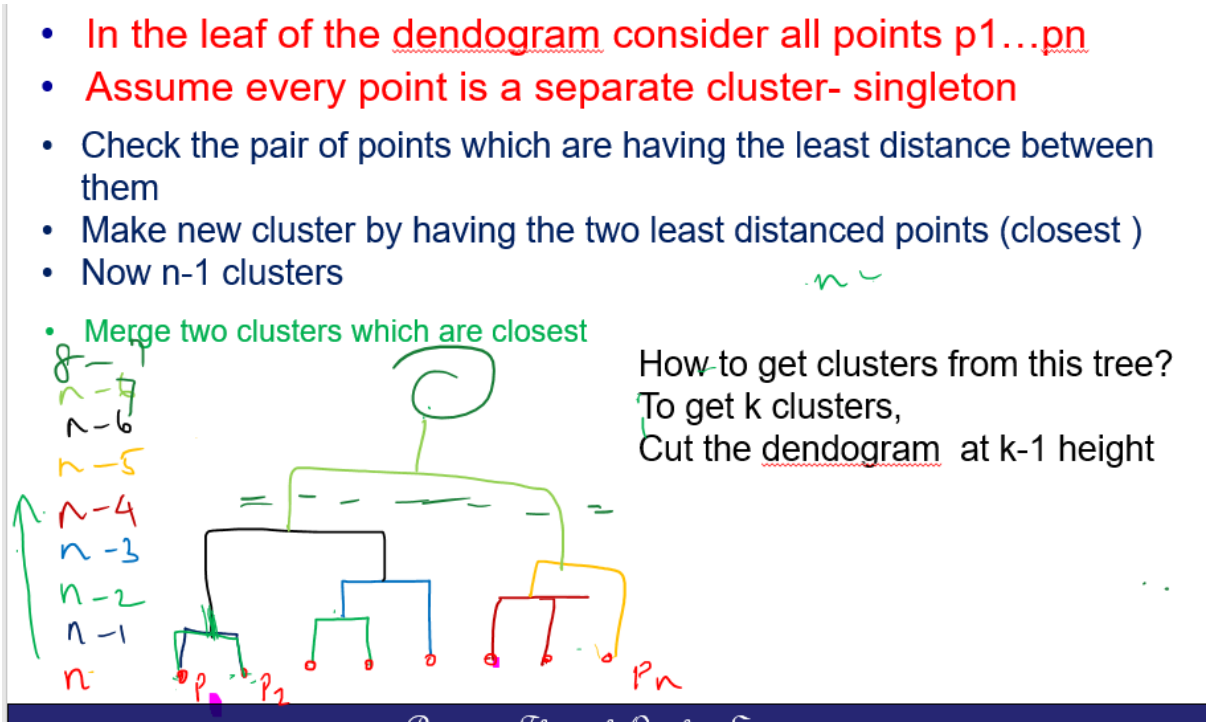
Major Clustering Approaches

- Partitioning algorithms: Construct various partitions and then evaluate them by some criterion
- Hierarchy algorithms: Create a hierarchical decomposition of the set of data (or objects) using some criterion
- Density-based: based on connectivity and density functions
- Grid-based: based on a multiple-level granularity structure
- Model-based: A model is hypothesized for each of the clusters and the idea is to find the best fit of that model to each other

Hierarchical Clustering:



Hierarchical Agglomerative Clustering:



- A **Hierarchical clustering** method works via grouping data into a tree of clusters. Hierarchical clustering begins by treating every data points as a separate cluster. Then, it repeatedly executes the subsequent steps:
- Identify the 2 clusters which can be closest together, and
- Merge the 2 maximum comparable clusters. We need to continue these steps until all the clusters are merged together.
- In Hierarchical Clustering, the aim is to produce a hierarchical series of nested clusters. A diagram called **Dendrogram** (A Dendrogram is a tree-like diagram that statistics the

sequences of merges or splits) graphically represents this hierarchy and is an inverted tree that describes the order in which factors are merged (bottom-up view) or cluster are break up (top-down view).

1. Agglomerative: (agglomerative nesting) AGNES

Initially consider every data point as an **individual** Cluster and at every step, **merge** the nearest pairs of the cluster. (It is a bottom-up method). At first everydata set is considered as individual entity or cluster. At every iteration, the clusters merge with different clusters until one cluster is formed.

- Algorithm for Agglomerative Hierarchical Clustering is:

1. Calculate the similarity of one cluster with all the other clusters (calculate proximity matrix)
2. Consider every data point as a individual cluster
3. Merge the clusters which are highly similar or close to each other.
4. Recalculate the proximity matrix for each cluster
5. Repeat Step 3 and 4 until only a single cluster remains.
6. Let's see the graphical representation of this algorithm using a dendrogram.

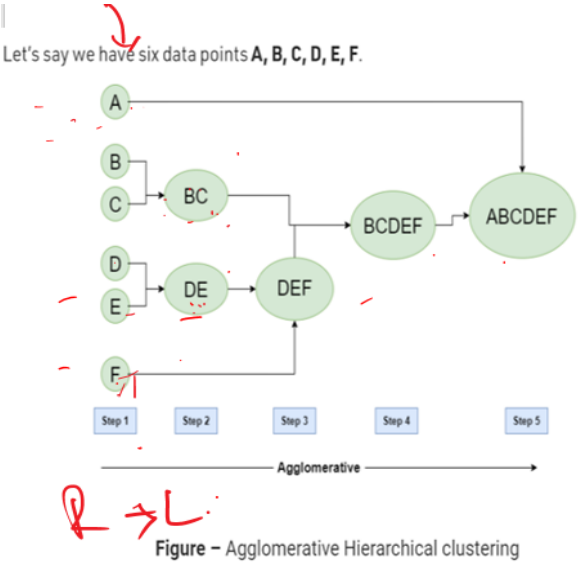
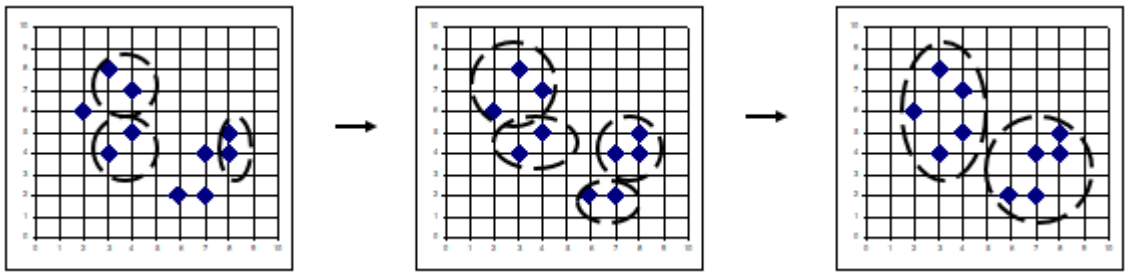


Figure – Agglomerative Hierarchical clustering

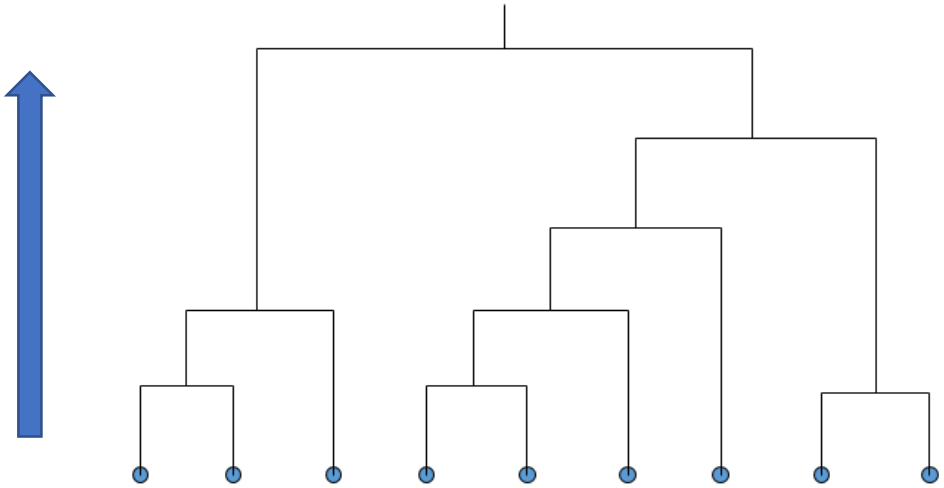
•Step-1:
Consider each alphabet as a single cluster and calculate the distance of one cluster from all the other clusters.
•Step-2:
In the second step comparable clusters are merged together to form a single cluster. Let's say cluster (B) and cluster (C) are very similar to each other therefore we merge them in the second step similarly with cluster (D) and (E) and at last, we get the clusters
[(A), (BC), (DE), (F)]
•Step-3:
We recalculate the proximity according to the algorithm and merge the two nearest clusters([(DE), (F)]) together to form new clusters as [(A), (BC), (DEF)]
•Step-4:
Repeating the same process; The clusters DEF and BC are comparable and merged together to form a new cluster. We're now left with clusters [(A), (BCDEF)].
•Step-5:
At last the two remaining clusters are merged together to form a single cluster [(ABCDEF)].



A *Dendrogram* Shows How the Clusters are Merged Hierarchically

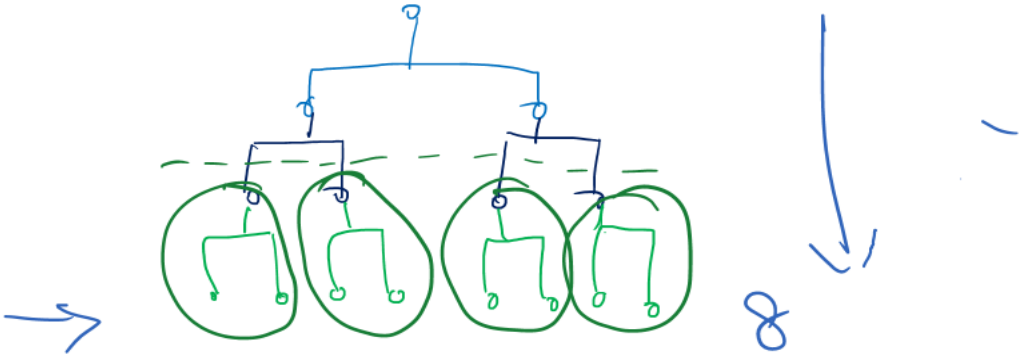
Decompose data objects into a several levels of nested partitioning (tree of clusters), called a dendrogram.

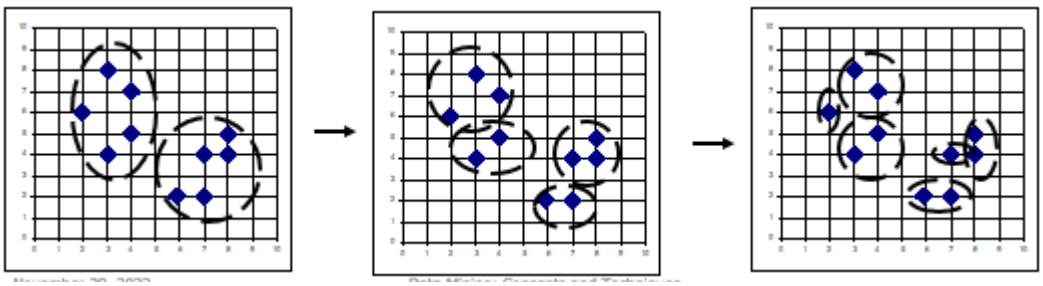
A clustering of the data objects is obtained by cutting the dendrogram at the desired level, then each connected component forms a cluster.



Hierarchical Divisive Analysis: (DIANA)

- All the n points are in one cluster
- Then divided into two and so on till singleton cluster





Divisive:

We can say that the Divisive Hierarchical clustering is precisely the **opposite** of the Agglomerative Hierarchical clustering. In Divisive Hierarchical clustering, we take into account all of the data points as a single cluster and in every iteration, we separate the data points from the clusters which aren't comparable. In the end, we are left with N clusters.

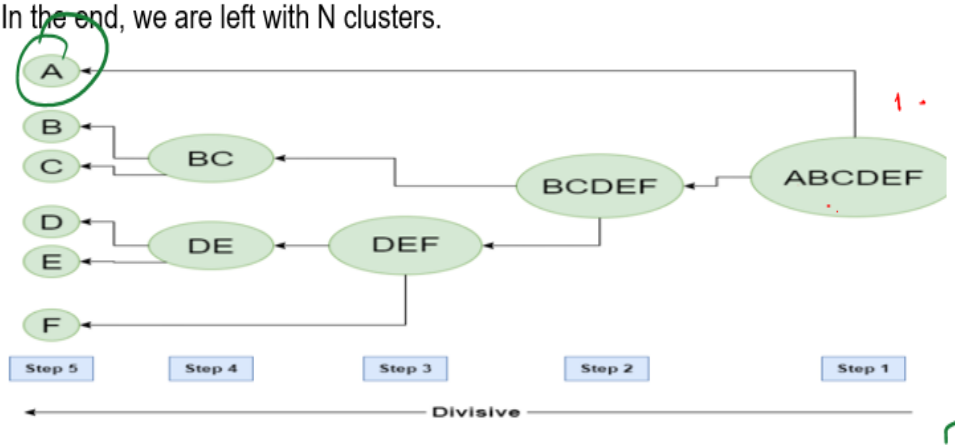
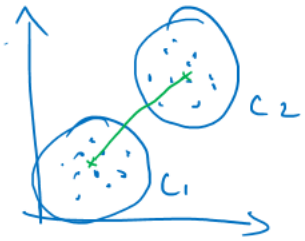


Figure – Divisive Hierarchical clustering

Measurement between Clusters:

- How do you measure closeness between two clusters?
  - Average distance: centroid of both clusters – distance between them is the distance between two clusters (average linkage clustering)
  - Minimum distance : distance between the closest two points in the clusters (single linkage clustering)
  - Maximum distance: distance between the farthest two points in the clusters (Complete linkage clustering)



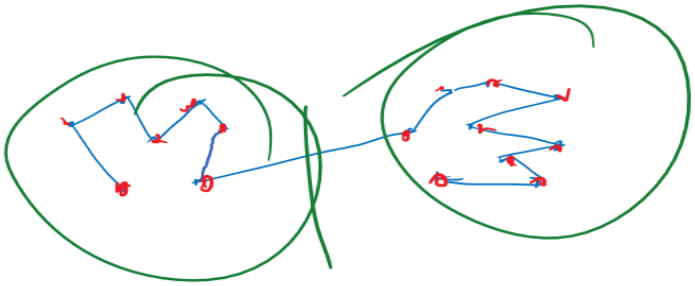
Three types of HC based on closeness measurement:

- Depending on which of these 3 you use, you have 3 names for this hierarchical algorithms.
- if you use the average distance you call it a average linkage clustering, -average linkage clustering.

- If you have the minimum distance as the closeness -single linkage clustering (most popular)
- If you have the maximum distance - a complete linkage clustering.
- Method is same, same dendogram, but closeness of two clusters is differently defined.

a) **single linkage clustering**

- Dendogram construction- mininmal spanning tree
- Only one path between nodes
- Total cost of the tree is sum of all the edges.
- Among all possible tree this is mimimum
- To get k cluster cut the heaviest k-1 edge



- Advantages of single and complete linkage clustering
  - Can find arbitrary – non convex cluster

- Disadvantages
  - Sensitive to noise
  - Time consuming

K-means clustering (Centroid Clustering)

Basic idea

- Start with an initial partition
- Refine the partitions over iteration
- Stop when good clusters are obtained or no change over iterations
- Cluster points into K-clusters- accordingly choose k
- How to decide the k? – no guide line

- Sol: Depending on the domain or requirement decide it.

The Basic Idea

The basic idea behind k-means clustering consists of defining clusters so that the total intra-cluster variation (known as total within-cluster variation) is minimized. There are several k-means algorithms available. The standard algorithm is the Hartigan-Wong algorithm (1979), which defines the total within-cluster variation as the sum of squared distances Euclidean distances between items and the corresponding centroid:

$$W(C_k) = \sum_{x_i \in C_k} (x_i - \mu_k)^2 \qquad (6)$$

where:

- x_i is a data point belonging to the cluster C_k
- μ_k is the mean value of the points assigned to the cluster C_k

Each observation (x_i) is assigned to a given cluster such that the sum of squares (SS) distance of the observation to their assigned cluster centers (μ_k) is minimized.

We define the total within-cluster variation as follows:

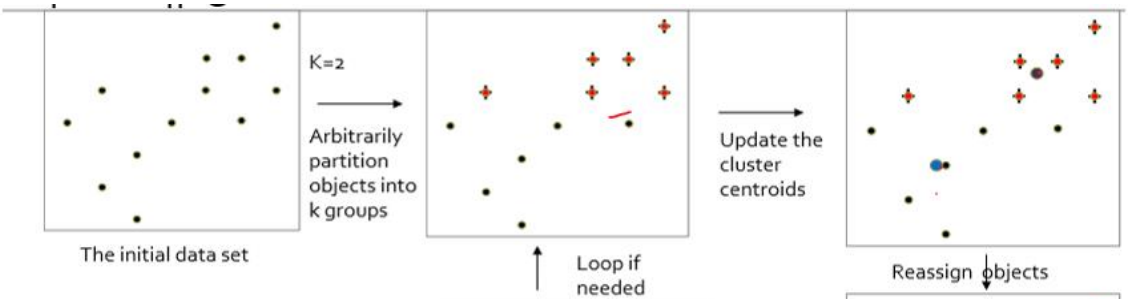
$$tot.withiness = \sum_{k=1}^{k} W(C_k) = \sum_{k=1}^{k} \sum_{x_i \in C_k} (x_i - \mu_k)^2 \qquad (7)$$

The *total within-cluster sum of square* measures the compactness (i.e goodness) of the clustering and we want it to be as small as possible.

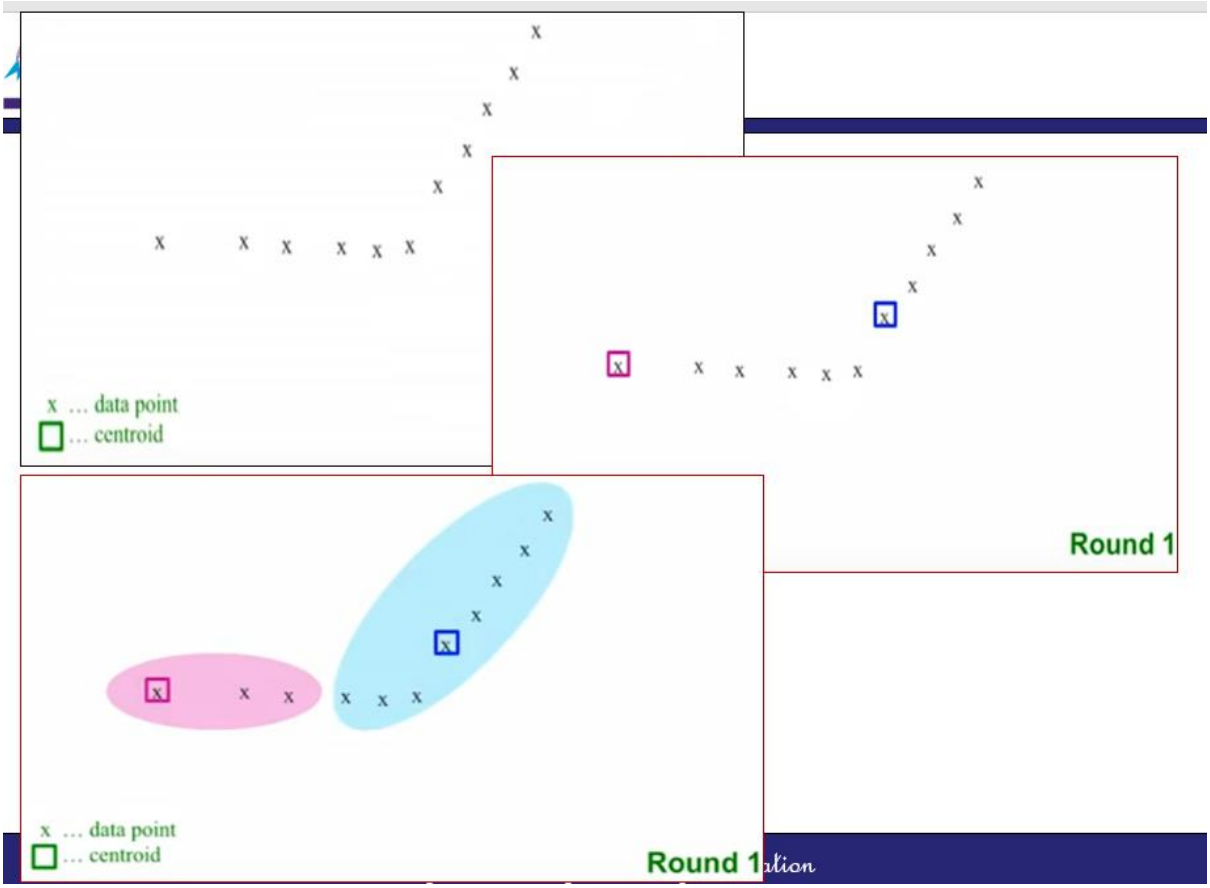
Method:

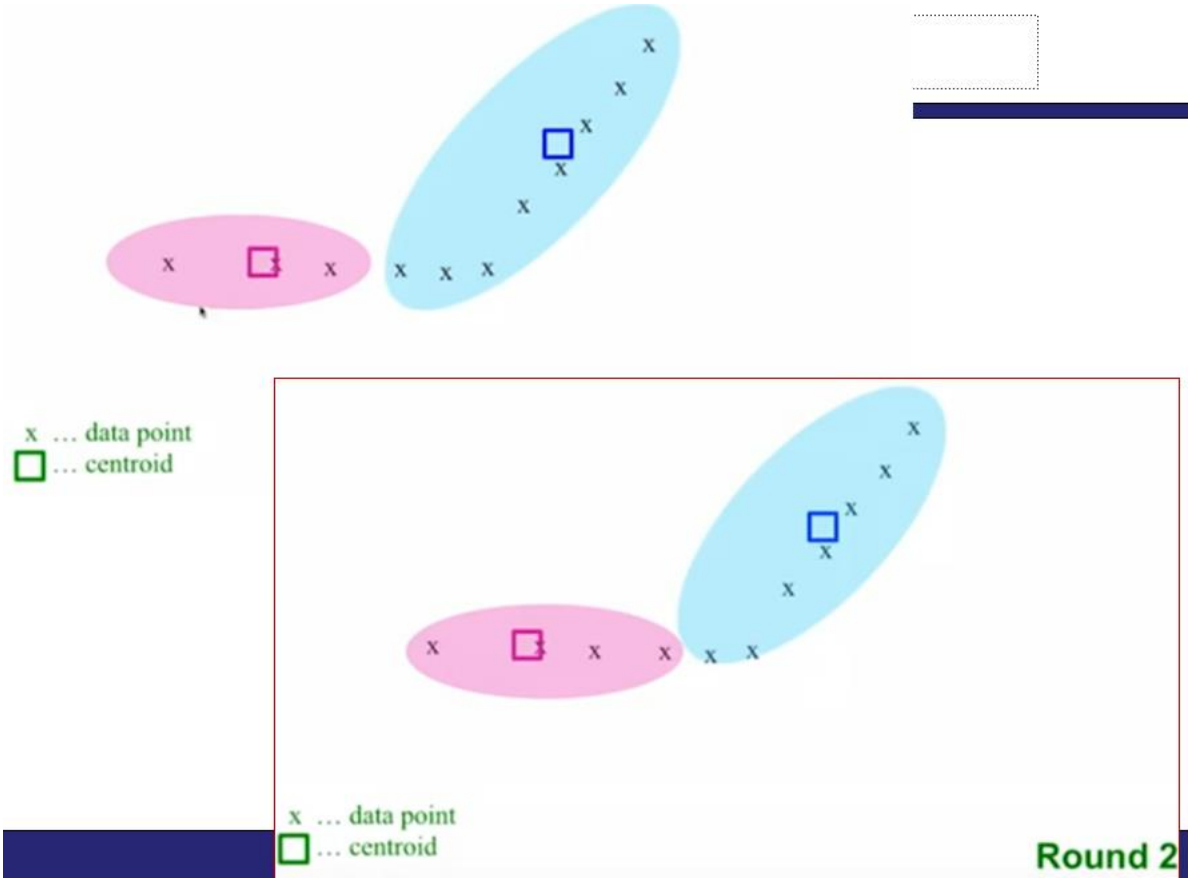
- Given *k*, the *k-means* algorithm is implemented in four steps:
  1. Partition objects into *k* nonempty subsets
  2. Compute seed points as the centroids of the clusters of the current partitioning (the centroid is the center, i.e., *mean point*, of the cluster) (From p points, select randomly)
  3. Assign each object to the cluster with the nearest seed point
  4. Go back to Step 2, stop when the assignment does not change

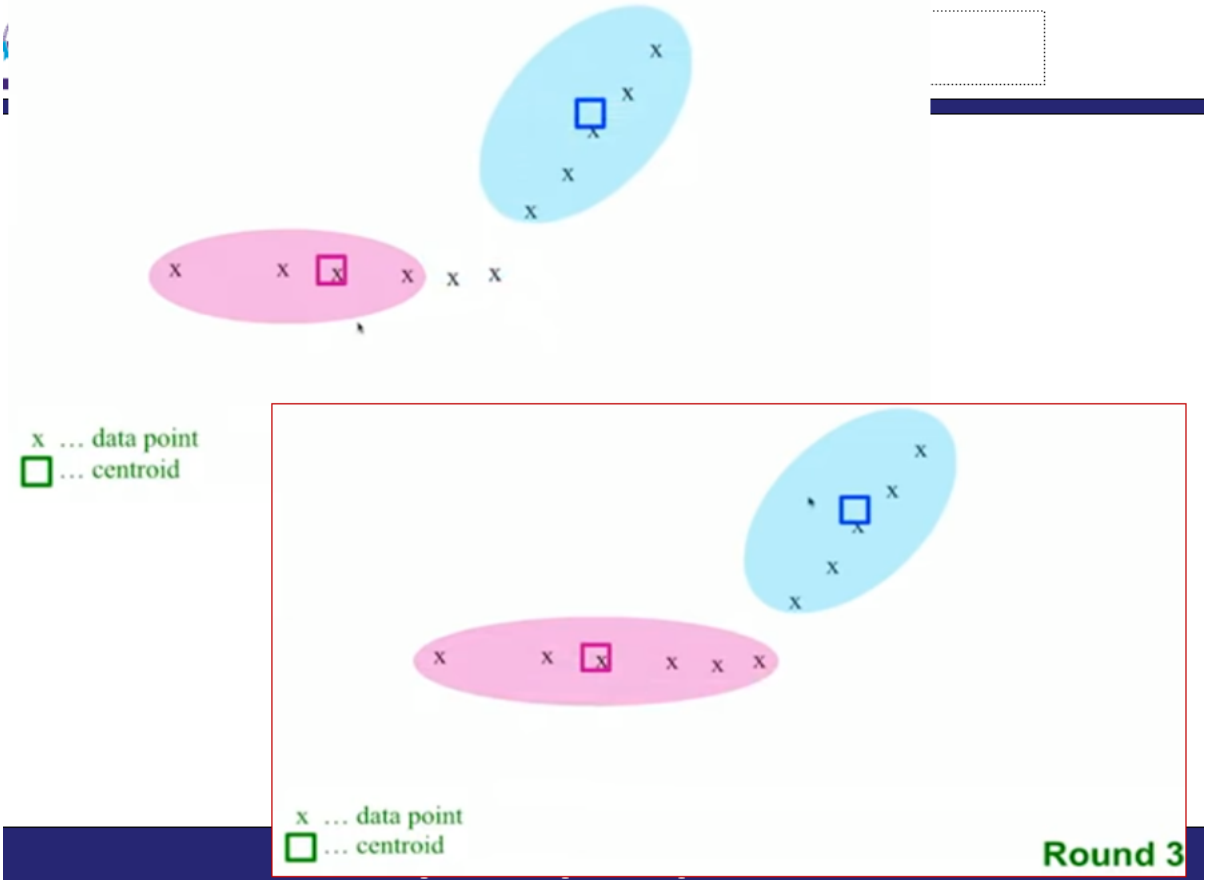
Explanation using Diagram:



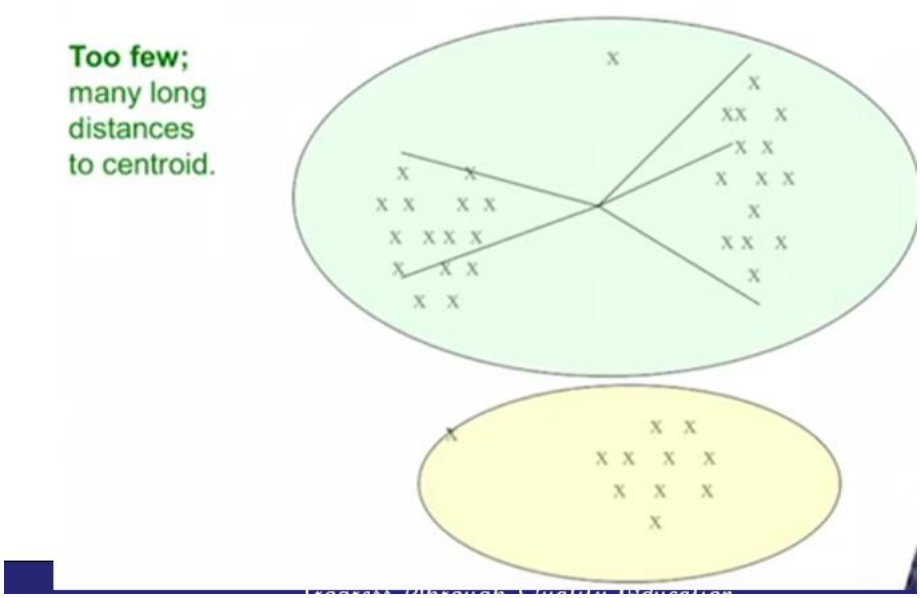
- Partition objects into *k* nonempty subsets
- Repeat
  - Compute centroid (i.e., mean point) for each partition
  - Assign each object to the cluster of its nearest centroid
- Until no change

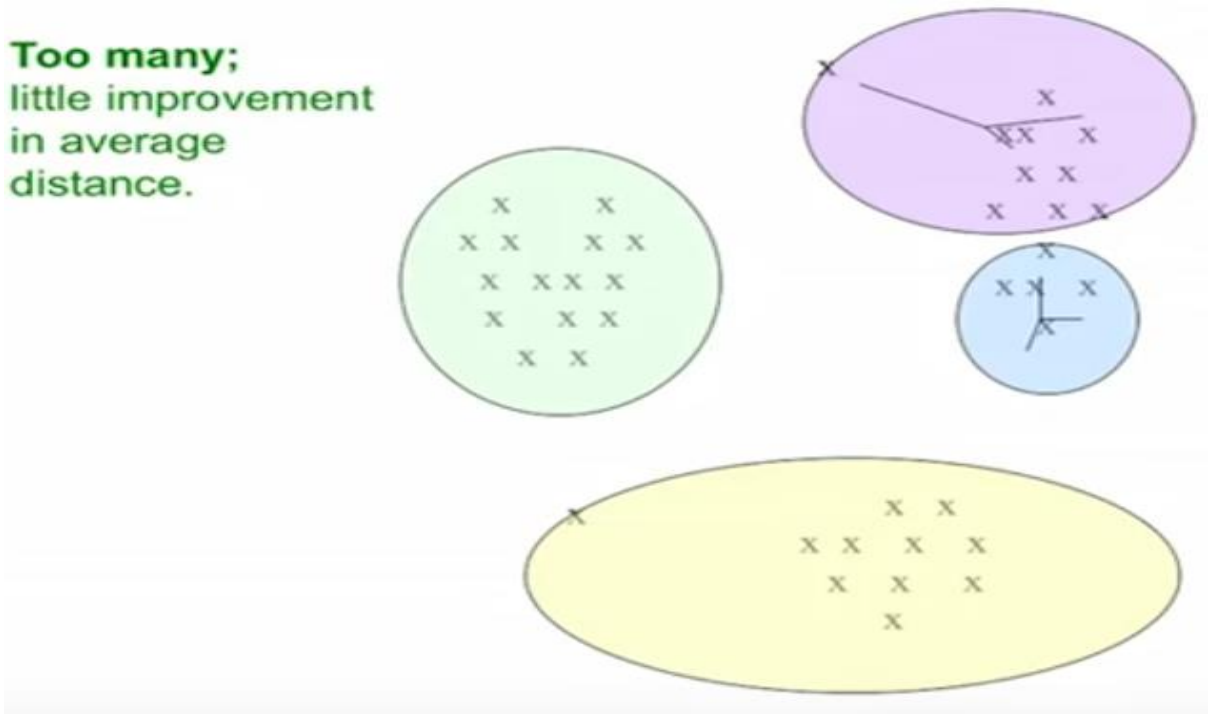


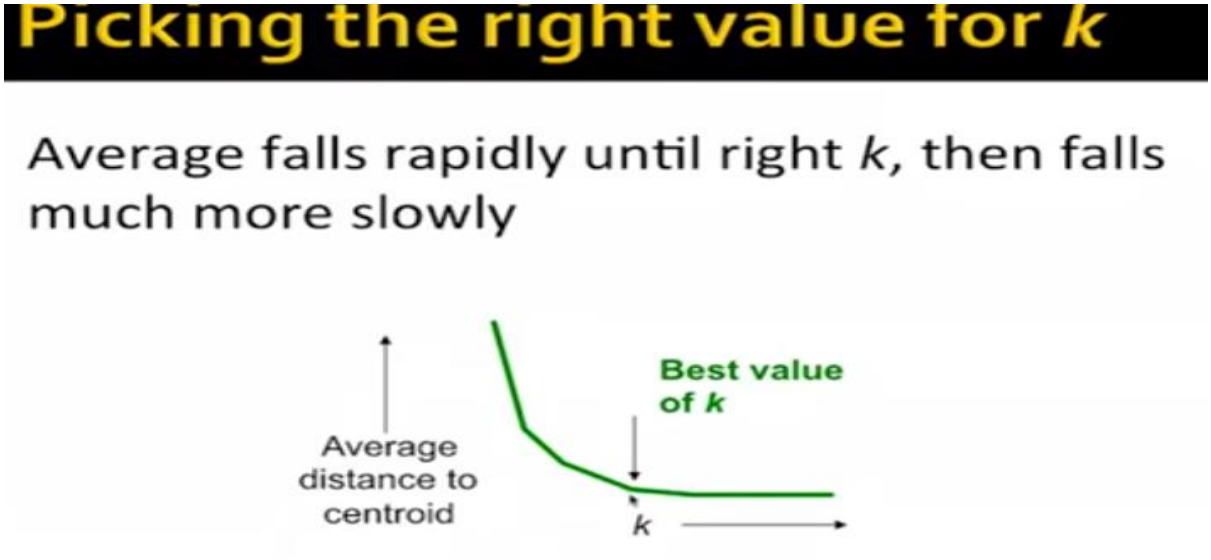

x … data point
… centroid
x … data point
… centroid
Round 2


x … data point
… centroid
x … data point
… centroid
Round 3

- How to select K







Comments on the *K-Means* Method

- Strength: *Efficient*: $O(tkn)$, where n is # objects, k is # clusters, and t is # iterations. Normally, k, $t << n$.
- Robust to noise, spherical, fast
- Weakness
  - Applicable only to objects in a continuous n-dimensional space
  - Need to specify k, the *number* of clusters, in advance (there are ways to automatically determine the best k (see Hastie et al., 2009)
  - Sensitive to noisy data and *outliers*
  - Not suitable to discover clusters with *non-convex shapes*

Example Problem:

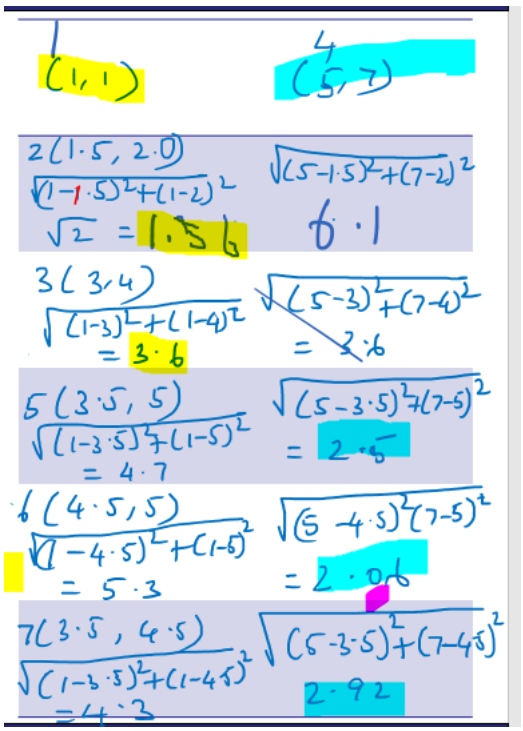
As a simple illustration of a k-means algorithm, consider the following data set consisting of the scores of two variables on each of seven individuals:

Subject	A	B
1	1.0	1.0
2	1.5	2.0
3	3.0	4.0
4	5.0	7.0
5	3.5	5.0
6	4.5	5.0
7	3.5	4.5

This data set is to be grouped into two clusters. As a first step in finding a sensible initial partition, let the A & B values of the two individuals furthest apart (using the Euclidean distance measure), define the initial cluster means, giving:

	Individual	Mean Vector (centroid)
Group 1	1	(1.0, 1.0)
Group 2	4	(5.0, 7.0)

The remaining individuals are now examined in sequence and allocated to the cluster to which they are closest, in terms of Euclidean distance to the cluster mean. The mean vector is recalculated each time a new member is added. This leads to the following series of steps:



	Cluster 1		Cluster 2	
Step	Individual		Individual	
1	1		4	
2	1, 2		4	
3	1, 2, 3		4	
4	1, 2, 3		4, 5	
5	1, 2, 3		4, 5, 6	
6	1, 2, 3		4, 5, 6, 7	

	x	y
1	1	1
2	1·5	2·0
3	3	4
Σ	5·5	7
	5·5/3 = 1·8	7/3 = 2·3

4 Σ 5
5 3·5
6 4
7 3·5

Σ
5
5
4·5

= 16.5/4 21.5/4

= 4.1 5.4

	Individual	Mean Vector (centroid)
Cluster 1	1, 2, 3	(1.8, 2.3)
Cluster 2	4, 5, 6, 7	(4.1, 5.4)

But we cannot yet be sure that each individual has been assigned to the right cluster. So, we compare each individual's distance to its own cluster mean and to that of the opposite cluster. And we find:

Individual	Distance to mean (centroid) of Cluster 1	Distance to mean (centroid) of Cluster 2
1	1.5	5.4
2	0.4	4.3
3	2.1	1.8
4	5.7	1.8
5	3.2	0.7
6	3.8	0.6
7	2.8	1.1

Only individual 3 is nearer to the mean of the opposite cluster (Cluster 2) than its own (Cluster 1). In other words, each individual's distance to its own cluster mean should be smaller that the distance to the other cluster's mean (which is not the case with individual 3). Thus, individual 3 is relocated to Cluster 2 resulting in the new partition:

	Individual	Mean Vector (centroid)
Cluster 1	1, 2	(1.3, 1.5)
Cluster 2	3, 4, 5, 6, 7	(3.9, 5.1)

The iterative relocation would now continue from this new partition until no more relocations occur. However, in this example each individual is now nearer its own cluster mean than that of the other cluster and the iteration stops, choosing the latest partitioning as the final cluster solution.

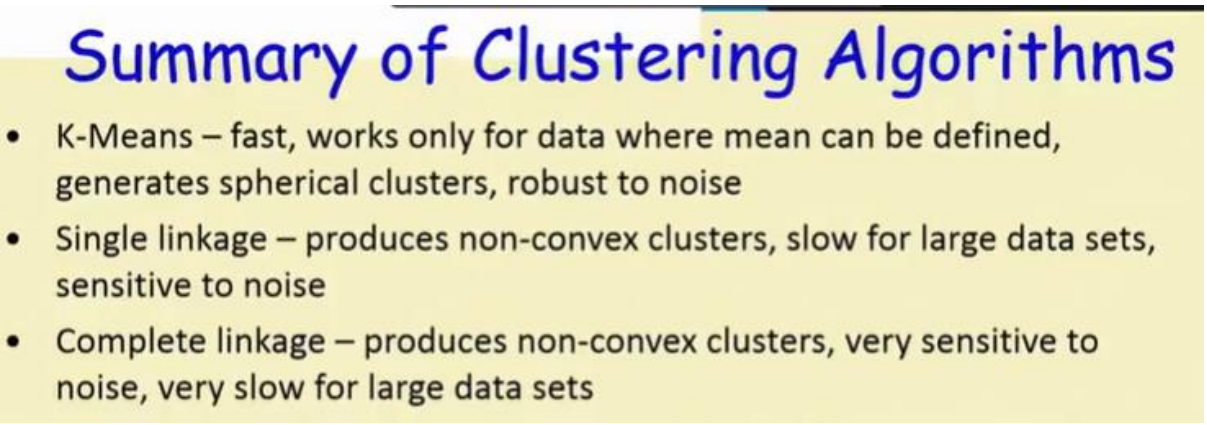
Also, it is possible that the k-means algorithm won't find a final solution. In this case it would be a good idea to consider stopping the algorithm after a pre-chosen maximum of iterations.

Stopping Criteria:

When to stop the clustering iteration. When the movement of points from one cluster to another stopped, we can stop the process. But when data points are more we cannot verify the movement, but still we can check the centroid value. If the centroid calculation of consecutive iterations are equal, then we can stop the clustering process.

Applications:

- **Behavioral segmentation:**
  - **Segment by purchase history**
  - **Segment by activities on application, website, or platform**
  - **Define personas based on interests**
  - **Create profiles based on activity monitoring**
- **Inventory categorization:**
  - **Group inventory by sales activity**
  - **Group inventory by manufacturing metrics**
- **Sorting sensor measurements:**
  - **Detect activity types in motion sensors**
  - **Group images**
  - **Separate audio**
  - **Identify groups in health monitoring**
- **Detecting bots or anomalies:**
  - **Separate valid activity groups from bots**
  - **Group valid activity to clean up outlier detection**

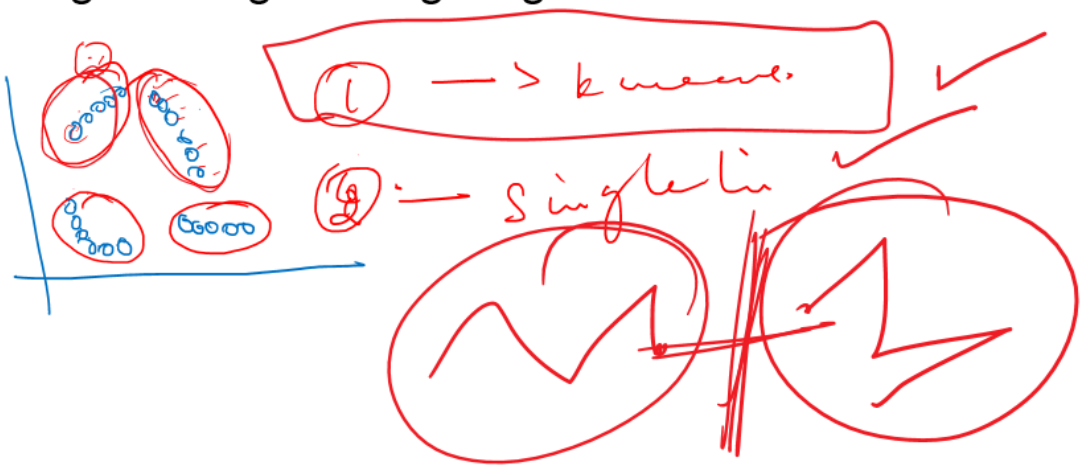


Hybrid clustering algorithm:

- **Combine- hierarchial with partitioned clustering alg**
  - **Single Linkage k-means**
  - **Takes the advantage of computational efficiency of k-means, robustness to noise and arbitrary shaped clusters from single linkage clustering- CLARA algorithm**

CLARA Algorithm → HC.

1. Cluster all the points using the k-means algorithm with a high value of k- lot of small spherical clusters
2. Merge the large number of small spherical clusters using the single linkage algorithm



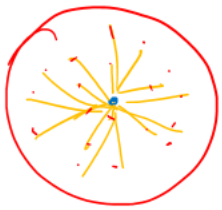
- For supervised classification we have a variety of measures to evaluate how good our model is
  - Accuracy, precision, recall
- For cluster analysis, the analogous question is how to evaluate the "goodness" of the resulting clusters?
- But "clusters are in the eye of the beholder"!
- Then why do we want to evaluate them?
  - To avoid finding patterns in noise
  - To compare clustering algorithms
  - To compare two sets of clusters
  - To compare two clusters

Different Aspects of Cluster Validation

1. Determining the clustering tendency of a set of data, i.e., distinguishing whether non-random structure actually exists in the data.
2. Comparing the results of a cluster analysis to externally known results, e.g., to externally given class labels.
3. Evaluating how well the results of a cluster analysis fit the data *without* reference to external information.
   - Use only the data
4. Comparing the results of two different sets of cluster analyses to determine which is better.

How Good?

- Sum square error
  - Sum of distance of all cluster points to the cluster center
  - If it is high –not good
  - If it is low – good
- Scatter coefficient =Avg. intra cluster distance / avg. inter cluster distance



Outliers

- **Points which are not part of any cluster**
- **The points that are away from clusters**
- **In many applications outliers are important**
  - **Credit card fraud**
  - **Disease outbreak**
  - **Extreme whether**

Many outlier detection algorithms are available

Single linkage Calculation:

Example 12.3 (Clustering using single linkage) To illustrate the single linkage algorithm, we consider the hypothetical distances between pairs of five objects as follows:

$$\mathbf{D} = \{d_{ik}\} = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{bmatrix} 0 & & & & \\ 9 & 0 & & & \\ 3 & 7 & 0 & & \\ 6 & 5 & 9 & 0 & \\ 11 & 10 & \textcircled{2} & 8 & 0 \end{bmatrix} \end{array}$$

Treating each object as a cluster, we commence clustering by merging the two closest items. Since

$$\min_{i,k} \{d_{ik}\} = d_{53} = 2$$

objects 5 and 3 are merged to form the cluster (35). To implement the next level of clustering, we need the distances between the cluster (35) and the remaining objects, 1, 2, and 4. The nearest neighbor distances are

$$d_{(35)1} = \min\{d_{31}, d_{51}\} = \min\{3, 11\} = 3$$
$$d_{(35)2} = \min\{d_{32}, d_{52}\} = \min\{7, 10\} = 7$$
$$d_{(35)4} = \min\{d_{34}, d_{54}\} = \min\{9, 8\} = 8$$

Deleting the rows and columns of **D** corresponding to objects 3 and 5, and adding a row and column for the cluster (35), we obtain the new distance matrix

$$\begin{array}{c} \\ (35) \\ 1 \\ 2 \\ 4 \end{array} \begin{array}{c} \begin{array}{cccc} (35) & 1 & 2 & 4 \end{array} \\ \begin{bmatrix} 0 & & & \\ \textcircled{3} & 0 & & \\ 7 & 9 & 0 & \\ 8 & 6 & 5 & 0 \end{bmatrix} \end{array}$$

The smallest distance between pairs of clusters is now $d_{(35)1} = 3$, and we merge cluster (1) with cluster (35) to get the next cluster, (135). Calculating

$$d_{(135)2} = \min\{d_{(35)2}, d_{12}\} = \min\{7, 9\} = 7$$
$$d_{(135)4} = \min\{d_{(35)4}, d_{14}\} = \min\{8, 6\} = 6$$

we find that the distance matrix for the next level of clustering is

$$\begin{array}{c} \\ (135) \\ 2 \\ 4 \end{array} \begin{array}{c} \begin{array}{ccc} (135) & 2 & 4 \end{array} \\ \begin{bmatrix} 0 & & \\ 7 & 0 & \\ 6 & \textcircled{5} & 0 \end{bmatrix} \end{array}$$

The minimum nearest neighbor distance between pairs of clusters is $d_{42} = 5$, and we merge objects 4 and 2 to get the cluster (24).

At this point we have two distinct clusters, (135) and (24). Their nearest neighbor distance is

$$d_{(135)(24)} = \min\{d_{(135)2}, d_{(135)4}\} = \min\{7, 6\} = 6$$

The final distance matrix becomes

$$\begin{array}{c} \\ (135) \\ (24) \end{array} \begin{array}{c} \begin{array}{cc} (135) & (24) \end{array} \\ \begin{bmatrix} 0 & \\ \textcircled{6} & 0 \end{bmatrix} \end{array}$$

Consequently, clusters (135) and (24) are merged to form a single cluster of all five objects, (12345), when the nearest neighbor distance reaches 6.

The dendrogram picturing the hierarchical clustering just concluded is shown in Figure 12.3. The groupings and the distance levels at which they occur are clearly illustrated by the dendrogram. ■

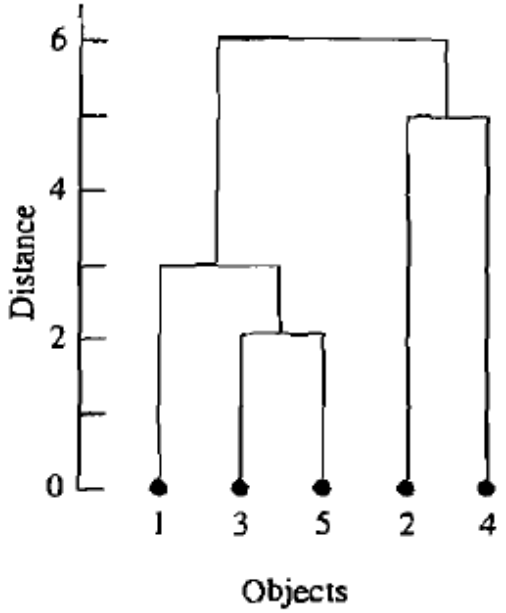


Figure 12.3 Single linkage dendrogram for distances between five objects.

Complete Linkage calculation

Example 12.5 (Clustering using complete linkage) Let us return to the distance matrix introduced in Example 12.3:

$$\mathbf{D} = \{d_{ik}\} = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{bmatrix} 0 & & & & \\ 9 & 0 & & & \\ 3 & 7 & 0 & & \\ 6 & 5 & 9 & 0 & \\ 11 & 10 & \textcircled{2} & 8 & 0 \end{bmatrix} \end{array}$$

At the first stage, objects 3 and 5 are merged, since they are most similar. This gives the cluster (35). At stage 2, we compute

$$d_{(35)1} = \max\{d_{31}, d_{51}\} = \max\{3, 11\} = 11$$
$$d_{(35)2} = \max\{d_{32}, d_{52}\} = 10$$
$$d_{(35)4} = \max\{d_{34}, d_{54}\} = 9$$

and the modified distance matrix becomes

$$\begin{array}{c} \\ (35) \\ 1 \\ 2 \\ 4 \end{array} \begin{array}{c} \begin{array}{cccc} (35) & 1 & 2 & 4 \end{array} \\ \begin{bmatrix} 0 & & & \\ 11 & 0 & & \\ 10 & 9 & 0 & \\ 9 & 6 & \textcircled{5} & 0 \end{bmatrix} \end{array}$$

The next merger occurs between the most similar groups, 2 and 4, to give the cluster (24). At stage 3, we have

$$d_{(24)(35)} = \max\{d_{2(35)}, d_{4(35)}\} = \max\{10, 9\} = 10$$
$$d_{(24)1} = \max\{d_{21}, d_{41}\} = 9$$

and the distance matrix

$$\begin{array}{c} \\ (35) \\ (24) \\ 1 \end{array} \begin{array}{c} \begin{array}{ccc} (35) & (24) & 1 \end{array} \\ \begin{bmatrix} 0 & & \\ 10 & 0 & \\ 11 & \textcircled{9} & 0 \end{bmatrix} \end{array}$$

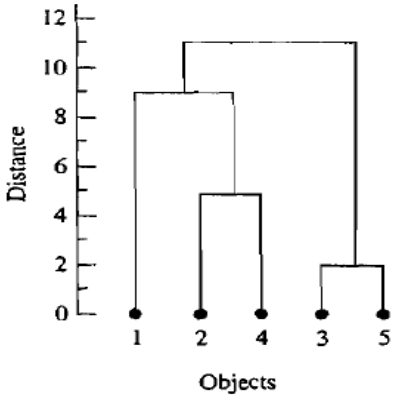


Figure 12.6 Complete linkage dendrogram for distances between five objects.

The next merger produces the cluster (124). At the final stage, the groups (35) and (124) are merged as the single cluster (12345) at level

$$d_{(124)(35)} = \max\{d_{1(35)}, d_{(24)(35)}\} = \max\{11, 10\} = 11$$

The dendrogram is given in Figure 12.6. ■

Comparing Figures 12.3 and 12.6, we see that the dendrograms for single linkage and complete linkage differ in the allocation of object 1 to previous groups.