INTRODUCTION-
What is clustering?

- **Clustering** is the classification of objects into different groups, or more precisely, the partitioning of a data set into subsets (clusters), so that the data in each subset (ideally) share some common trait - often according to some defined distance measure.

Types of clustering:

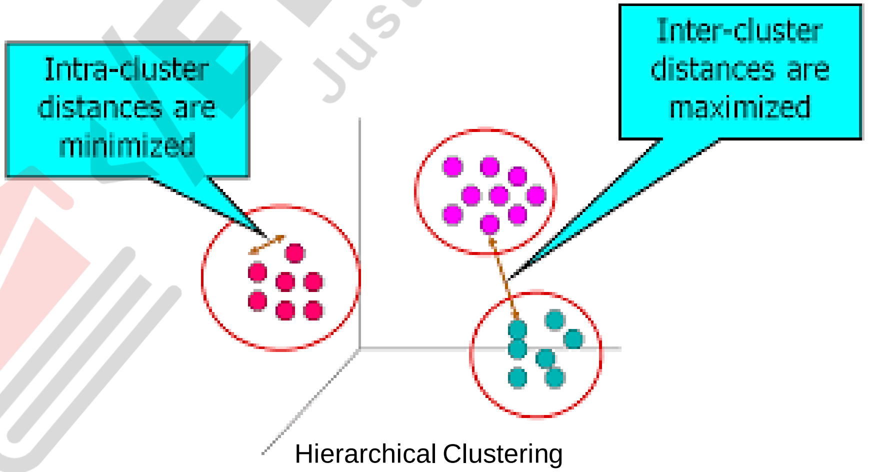
1. **Hierarchical algorithms**: these find successive clusters using previously established clusters.
   1. Agglomerative ("bottom-up"): Agglomerative algorithms begin with each element as a separate cluster and merge them into successively larger clusters.
   2. Divisive ("top-down"): Divisive algorithms begin with the whole set and proceed to divide it into successively smaller clusters.

2. **Partitional clustering:** Partitional algorithms determine all clusters at once. They include:

- ***K*-means and derivatives**
- Fuzzy *c*-means clustering
- QT clustering algorithm

What Is Good Clustering ?

- A good clustering method will produce high quality cluster with
  - High intra-class similarity.
  - Low inter-class similarity.
- The quality of clustering result depends on both the similarity measure used by the method and its implement


Intra-cluster distances are minimized
Inter-cluster distances are maximized

Common Distance measures:

- *Distance measure* will determine how the *similarity* of two elements is calculated and it will influence the shape of the clusters.

  They include:

1. The Euclidean distance (also called 2-norm distance) is given by:

$$d(x,y)=\sum_{i=1}^{p}|x_i-y_i|$$

2. The Manhattan distance (also called taxicab norm or 1-norm) is given by:

$$d(x,y)=\sqrt[2]{\sum_{i=1}^{p}|x_i-y_i|^2}$$

3. The maximum norm is given by:

$$d(x,y) = \max_{1 \le i \le p} |x_i - y_i|$$

4. The Mahalanobis distance corrects data for different scales and correlations in the variables.
5. Inner product space: The angle between two vectors can be used as a distance measure when clustering high dimensional data
6. Hamming distance (sometimes edit distance) measures the minimum number of substitutions required to change one member into another.

K-MEANS CLUSTERING

- The **k-means algorithm** is an algorithm to cluster *n* objects based on attributes into *k* partitions, where $k < n$.

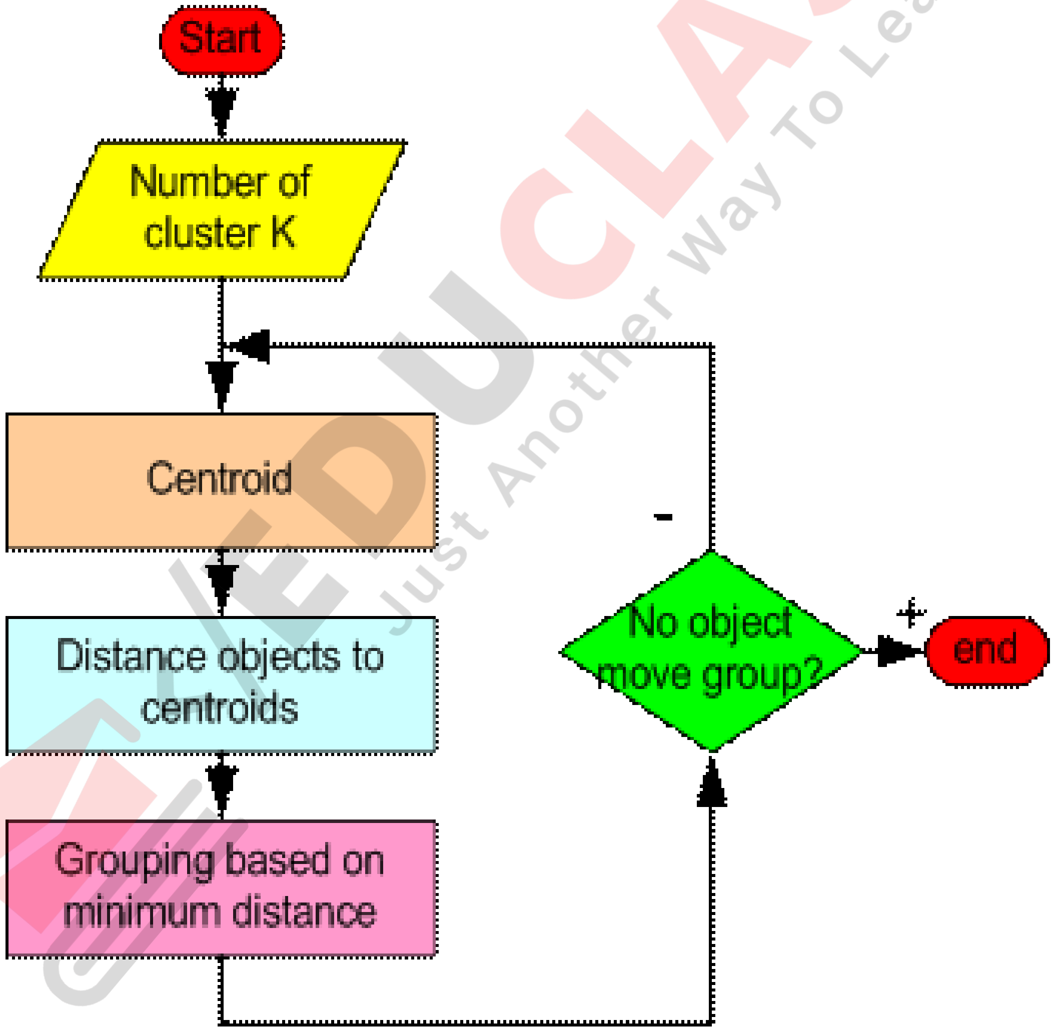
- An algorithm for partitioning (or clustering) N data points into K disjoint subsets S_j containing data points so as to minimize the sum-of-squares criterion

$$J = \sum_{j=1}^{K} \sum_{n \in S_j} |x_n - \mu_j|^2 ,$$

where x_n is a vector representing the the n^th data point and u_j is the geometric centroid of the data points in S_j.

- Simply speaking k-means clustering is an algorithm to classify or to group the objects based on attributes/features into K number of group.
- K is positive integer number.
- The grouping is done by minimizing the sum of squares of distances between data and the corresponding cluster centroid.

How the K-Mean Clustering algorithm works?

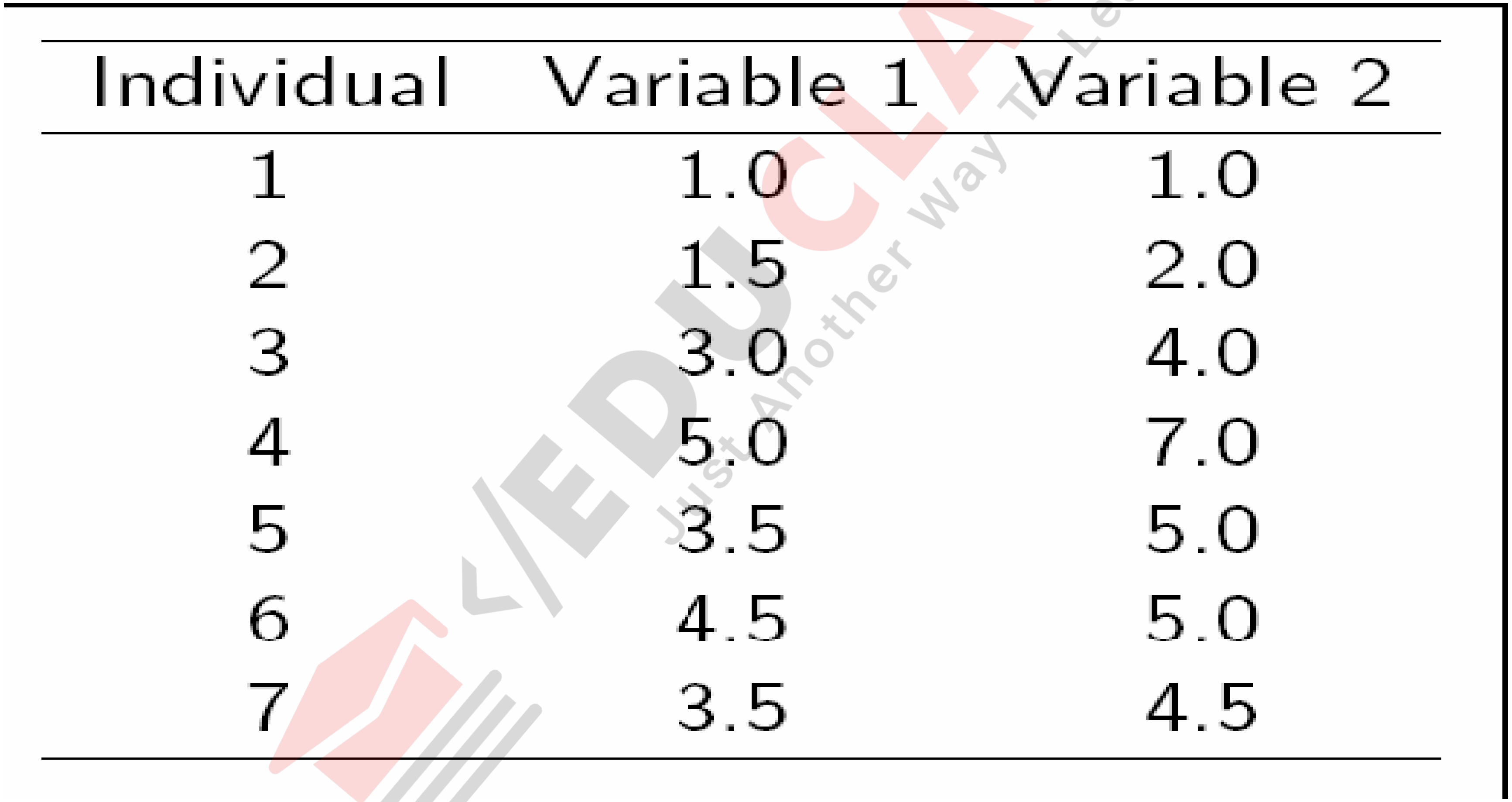

Start
Number of cluster K
Centroid
Distance objects to centroids
Grouping based on minimum distance
No object move group?
-
+
end

- **Step 1:** Begin with a decision on the value of k = number of clusters .
- **Step 2**: Put any initial partition that classifies the data into k clusters. You may assign the training samples randomly,or systematically as the following:

  1. Take the first k training sample as single-element clusters
  2. Assign each of the remaining (N-k) training sample to the cluster with the nearest centroid. After each assignment, recompute the centroid of the gaining cluster.

- **Step 3:** Take each sample in sequence and compute its distance from the centroid of each of the clusters. If a sample is not currently in the cluster with the closest centroid, switch this sample to that cluster and update the centroid of the cluster gaining the new sample and the cluster losing the sample.
- **Step 4 .** Repeat step 3 until convergence is achieved, that is until a pass through the training sample causes no new assignments.

A Simple example showing the implementation of k-means algorithm (using K=2)



Individual	Variable 1	Variable 2
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

Step 1:

Initialization: Randomly we choose following two centroids (k=2) for two clusters.

In this case the 2 centroid are: m1=(1.0,1.0) and m2=(5.0,7.0).

Individual	Variable 1	Variable 2
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

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

Step 2:

- Thus, we obtain two clusters containing:

  {1,2,3} and {4,5,6,7}.
- Their new centroids are:

$$m_1 = \left(\frac{1}{3}(1.0+1.5+3.0), \frac{1}{3}(1.0+2.0+4.0)\right) = (1.83, 2.33)$$

$$m_2 = \left(\frac{1}{4}(5.0+3.5+4.5+3.5), \frac{1}{4}(7.0+5.0+5.0+4.5)\right)$$

$$= (4.12, 5.38)$$

Individual	Centroid 1	Centroid 2
1	0	7.21
2 (1.5, 2.0)	1.12	6.10
3	3.61	3.61
4	7.21	0
5	4.72	2.5
6	5.31	2.06
7	4.30	2.92

$$d(m_1, 2) = \sqrt{|1.0-1.5|^2 + |1.0-2.0|^2} = 1.12$$

$$d(m_2, 2) = \sqrt{|5.0-1.5|^2 + |7.0-2.0|^2} = 6.10$$

<u>Step 3:</u>

- Now using these centroids we compute the Euclidean distance of each object, as shown in table.

- Therefore, the new clusters are:

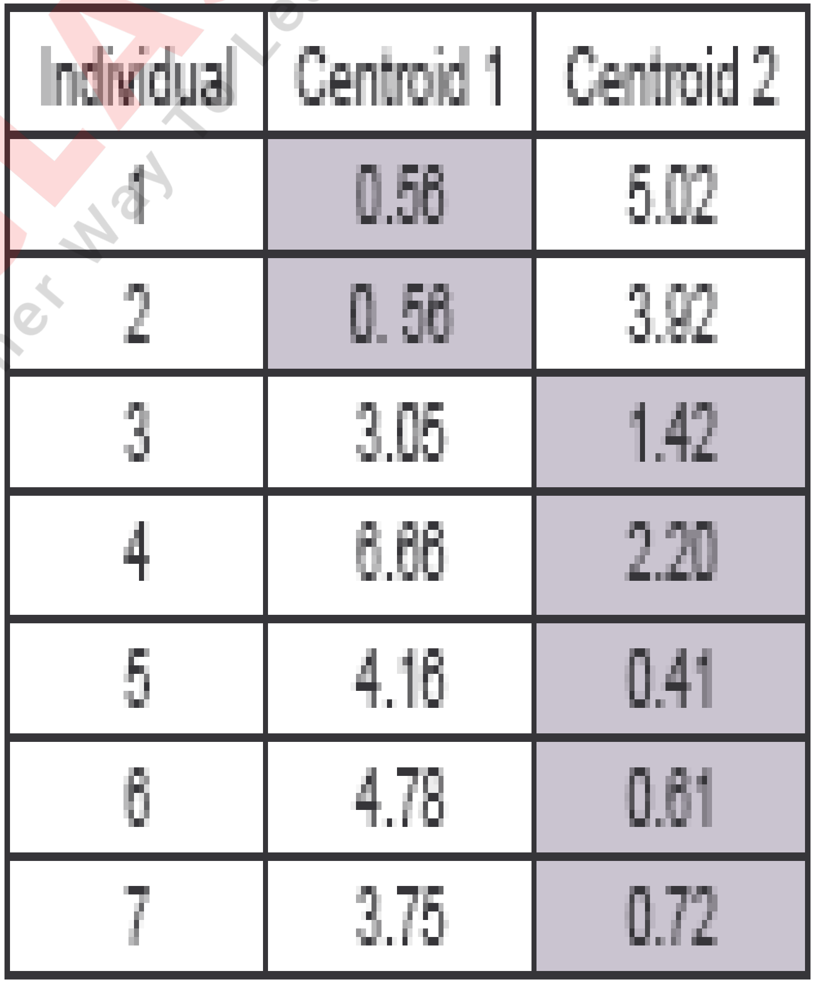
  {1,2} and {**3**,4,5,6,7}

- Next centroids are: m1=(1.25,1.5) and m2 = (3.9,5.1)

Individual	Centroid 1	Centroid 2
1	1.57	5.38
2	0.47	4.28
3	2.04	1.78
4	5.64	1.84
5	3.15	0.73
6	3.78	0.54
7	2.74	1.08

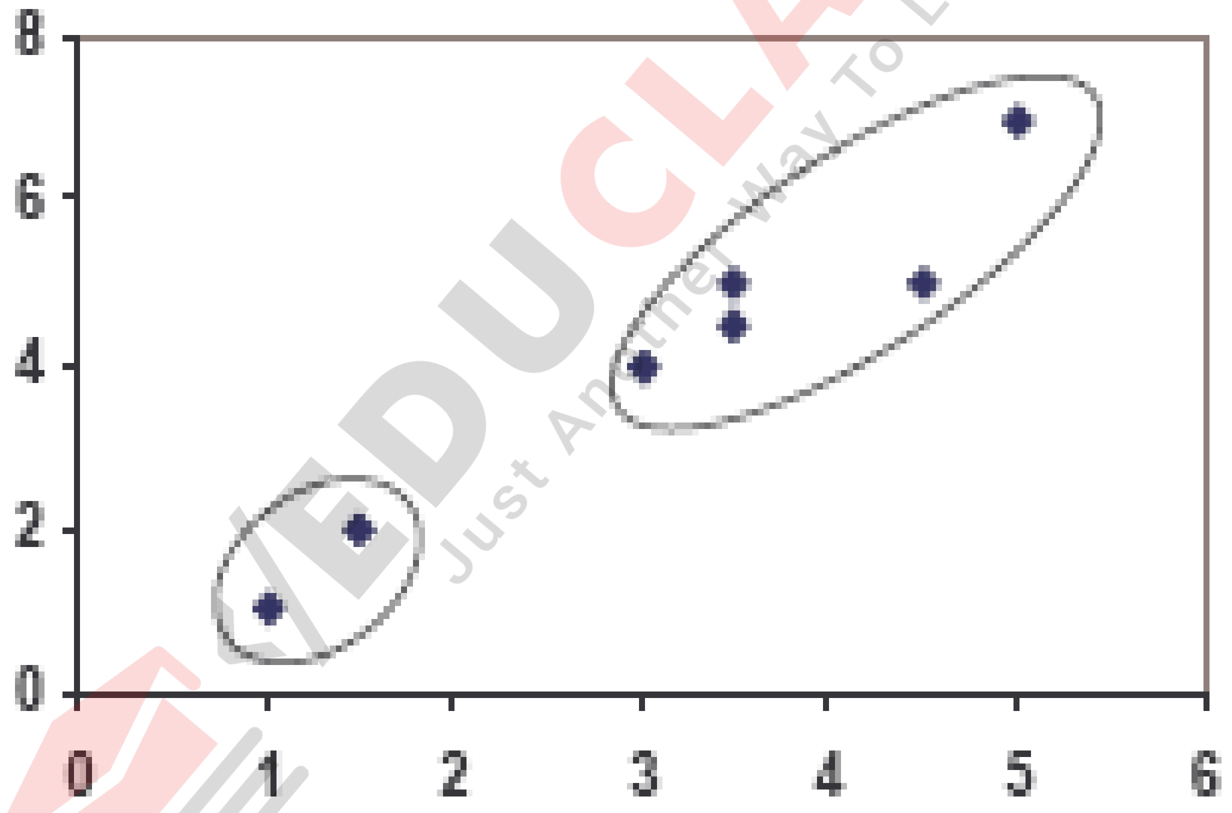
- Step 4 :
  The clusters obtained are:
  {1,2} and {3,4,5,6,7}

- Therefore, there is no change in the cluster.
- Thus, the algorithm comes to a halt here and final result consist of 2 clusters {1,2} and {3,4,5,6,7}.

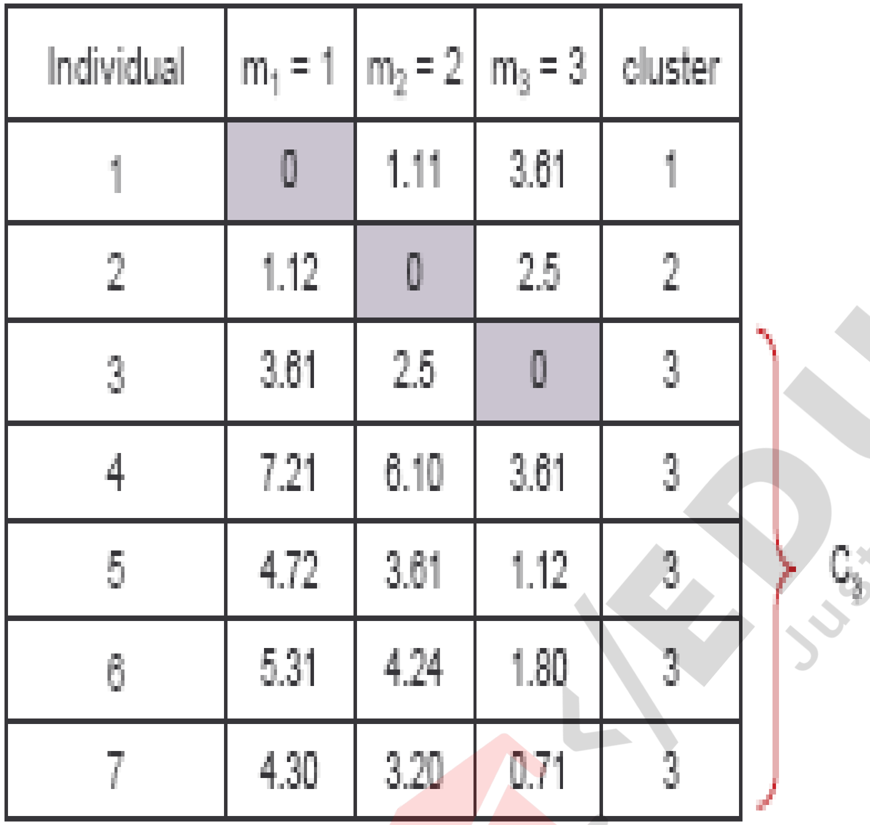


Individual	Centroid 1	Centroid 2
1	0.56	5.02
2	0.56	3.92
3	3.05	1.42
4	6.66	2.20
5	4.16	0.41
6	4.78	0.61
7	3.75	0.72

PLOT



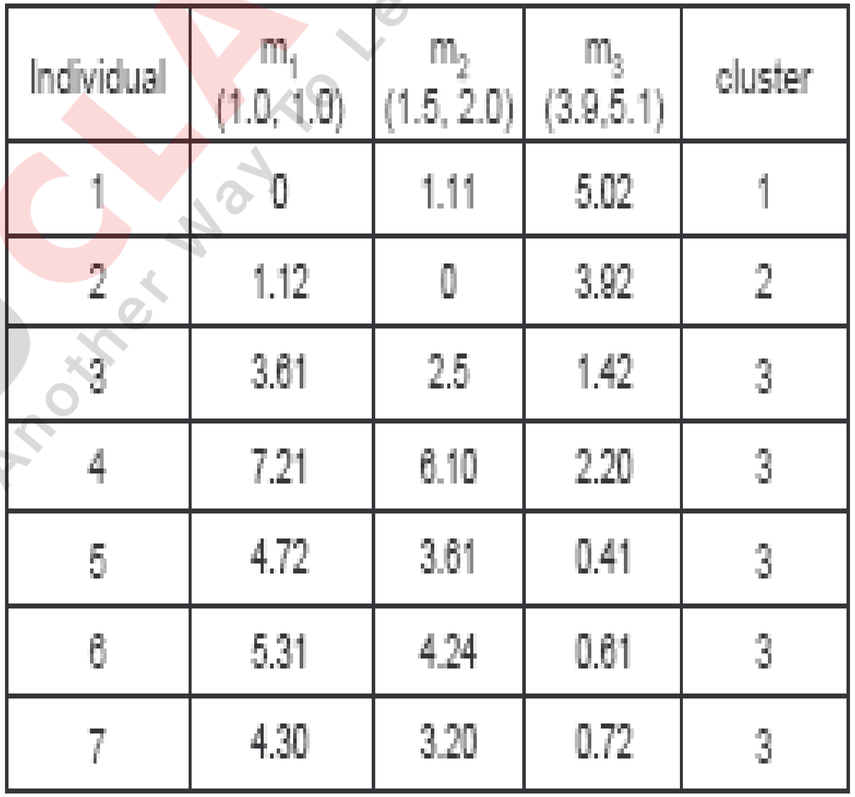
(with K=3)



Individual	$m_1 = 1$	$m_2 = 2$	$m_3 = 3$	cluster
1	0	1.11	3.61	1
2	1.12	0	2.5	2
3	3.61	2.5	0	3
4	7.21	6.10	3.61	3
5	4.72	3.61	1.12	3
6	5.31	4.24	1.80	3
7	4.30	3.20	0.71	3

C_3

clustering with initial centroids (1, 2, 3)

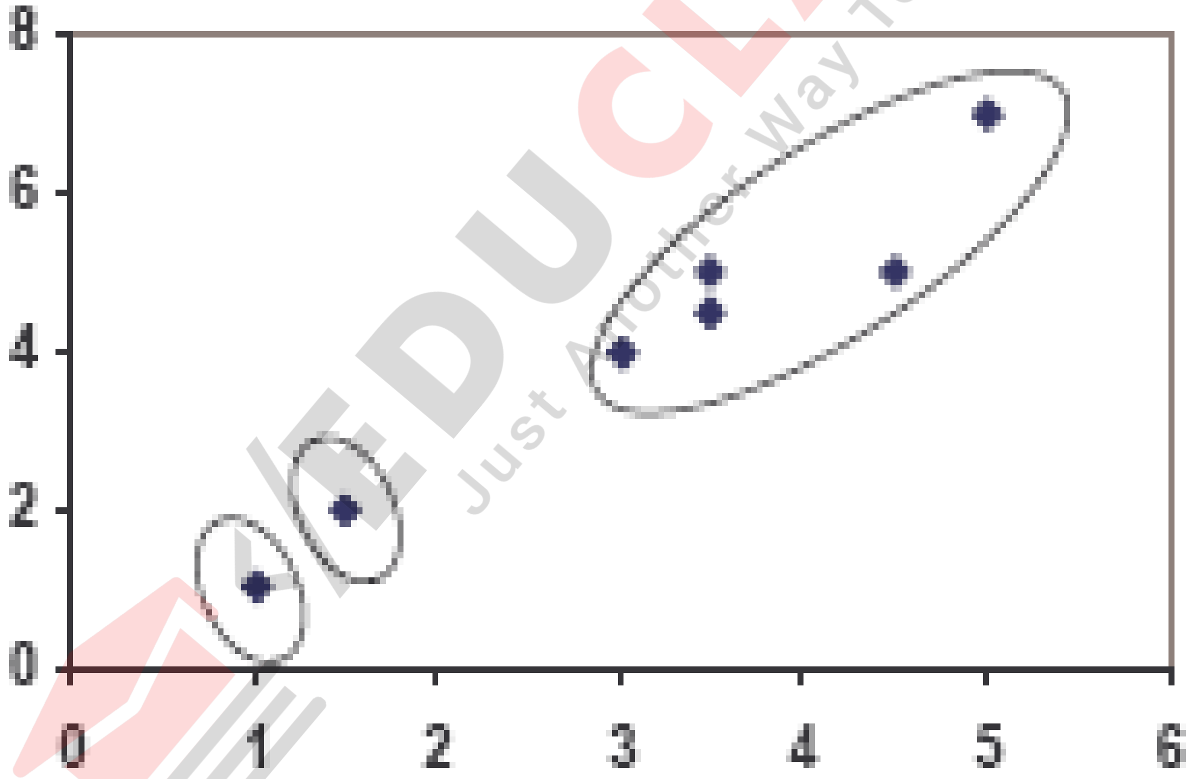


Individual	m_1 (1.0, 1.0)	m_2 (1.5, 2.0)	m_3 (3.9,5.1)	cluster
1	0	1.11	5.02	1
2	1.12	0	3.92	2
3	3.61	2.5	1.42	3
4	7.21	6.10	2.20	3
5	4.72	3.61	0.41	3
6	5.31	4.24	0.61	3
7	4.30	3.20	0.72	3

<u>Step 1</u>

<u>Step 2</u>

PLOT



Weaknesses of K-Mean Clustering

1. When the numbers of data are not so many, initial grouping will determine the cluster significantly.
2. The number of cluster, 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*.
5. Centroids can be dragged by outliers, or outliers might their own cluster instead of being ignored. Consider removing or clipping outliers before clustering.

Advantages of k-means

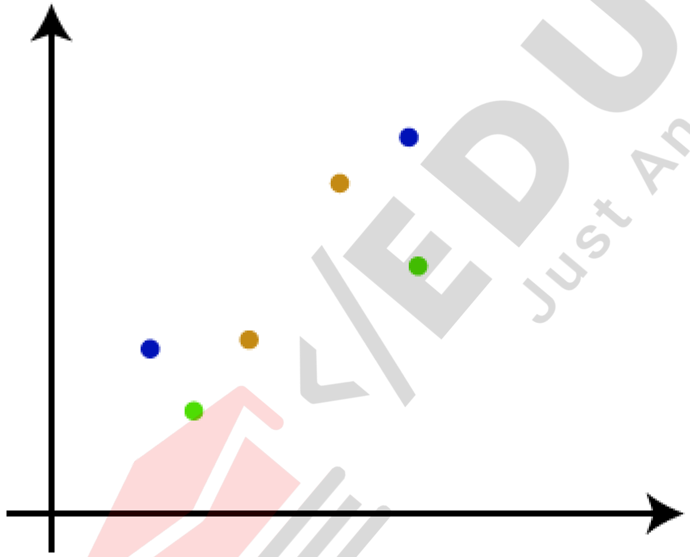
Relatively simple to implement.
Scales to large data sets.
Guarantees convergence.
Can warm-start the positions of centroids.
Easily adapts to new examples.
Generalizes to clusters of different shapes and sizes, such as elliptical clusters.

- The hierarchical clustering technique has two approaches:
- **Agglomerative:** Agglomerative is a **bottom-up** approach, in which the algorithm starts with taking all data points as single clusters and merging them until one cluster is left.
- **Divisive:** Divisive algorithm is the reverse of the agglomerative algorithm as it is a **top-down approach.**

Agglomerative Hierarchical clustering

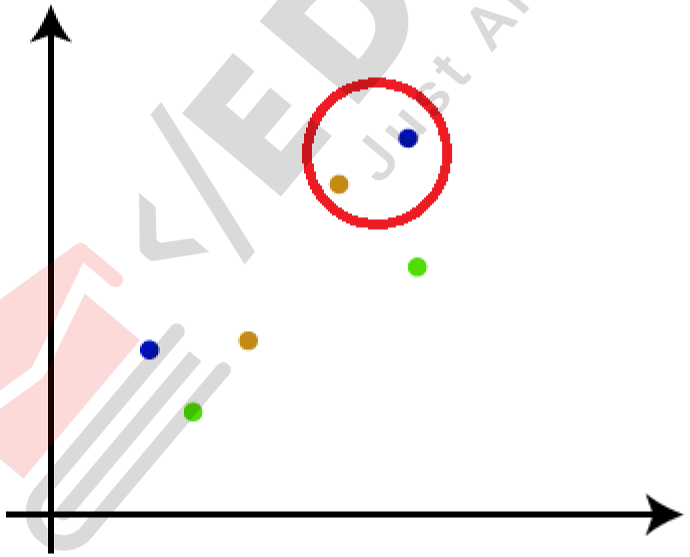
- The agglomerative hierarchical clustering algorithm is a popular example of HCA. To group the datasets into clusters, it follows the **bottom-up approach**. It means, this algorithm considers each dataset as a single cluster at the beginning, and then start combining the closest pair of clusters together. It does this until all the clusters are merged into a single cluster that contains all the datasets.
- This hierarchy of clusters is represented in the form of the dendrogram.

Working of the AHC algorithm

- **Step-1:** Create each data point as a single cluster. Let's say there are N data points, so the number of clusters will also be N.
- 

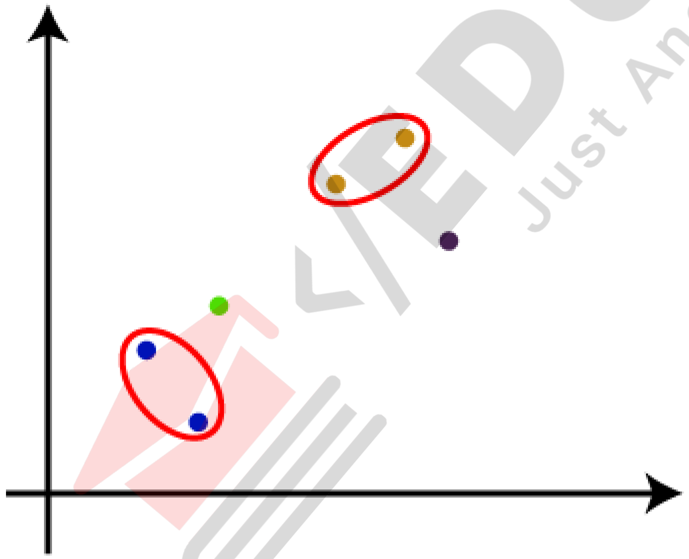
Cont…

- **Step-2:** Take two closest data points or clusters and merge them to form one cluster. So, there will now be N-1 clusters.

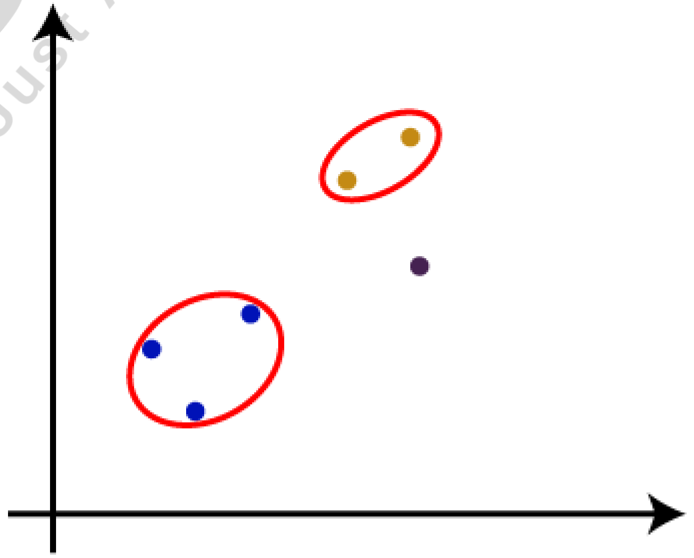


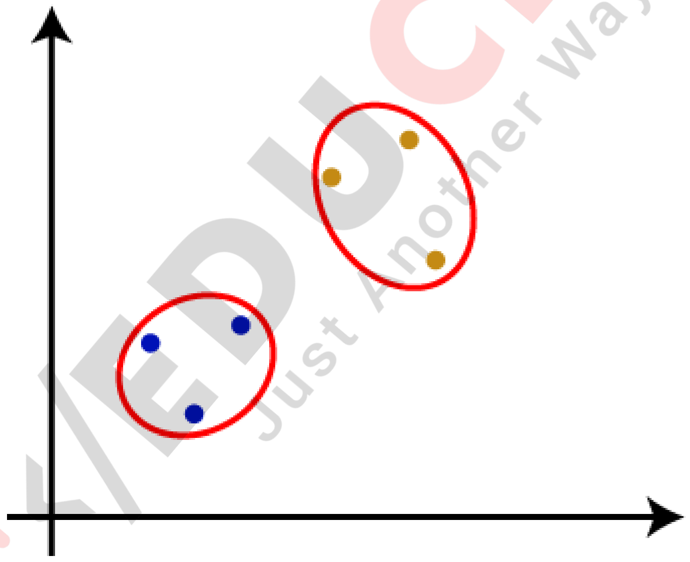
Cont…

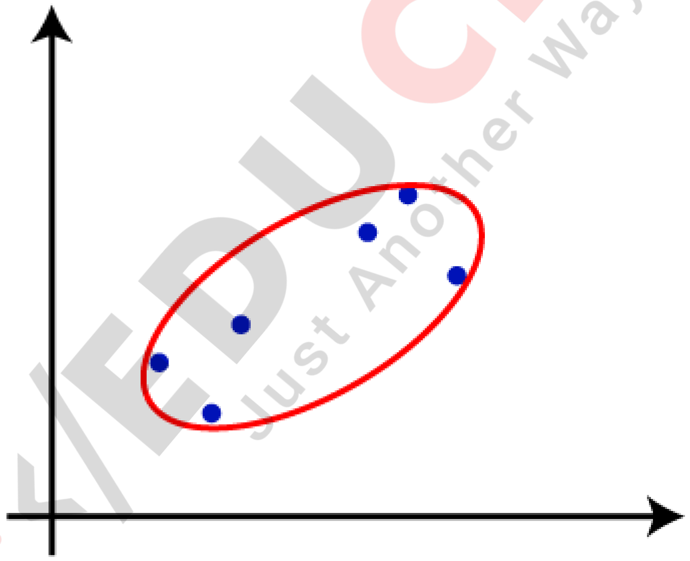
- **Step-3**: Again, take the two closest clusters and merge them together to form one cluster. There will be N-2 clusters.



- **Step-4:** Repeat Step 3 until only one cluster left. So, we will get the following clusters. Consider the below images:



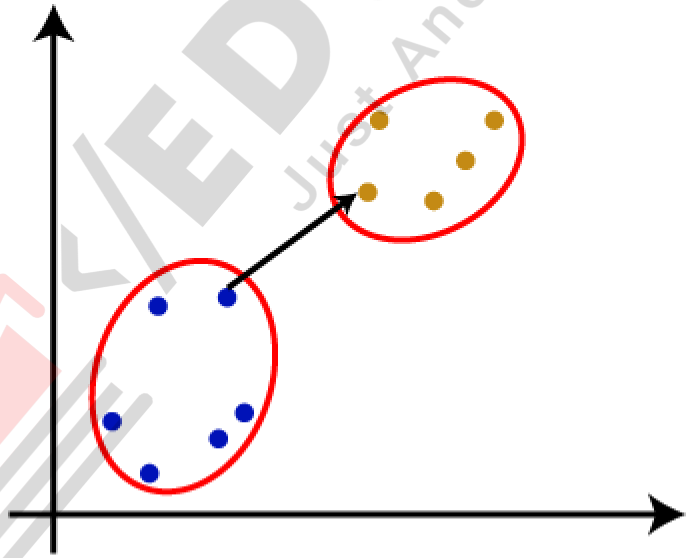




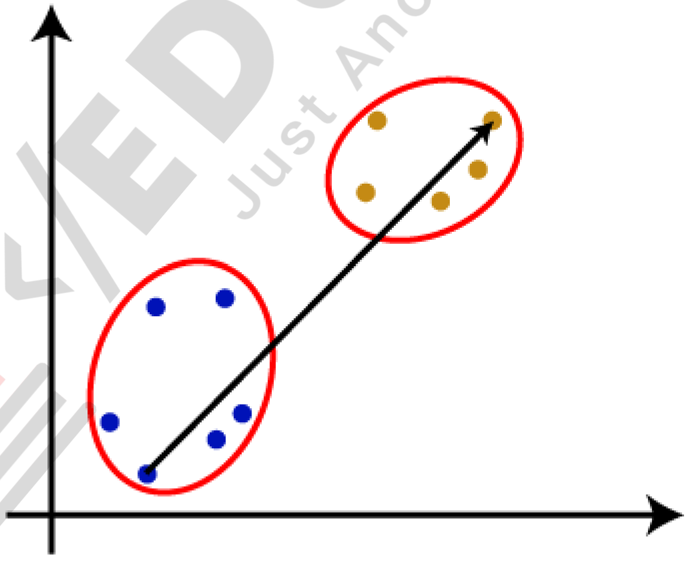
- **Step-5:** Once all the clusters are combined into one big cluster, develop the dendrogram to divide the clusters as per the problem.

Measure for the distance between two clusters

- The **closest distance** between the two clusters is crucial for the hierarchical clustering.
- There are various ways to calculate the distance between two clusters, and these ways decide the rule for clustering.
- These measures are called **Linkage methods**. Some of the popular linkage methods are given below:

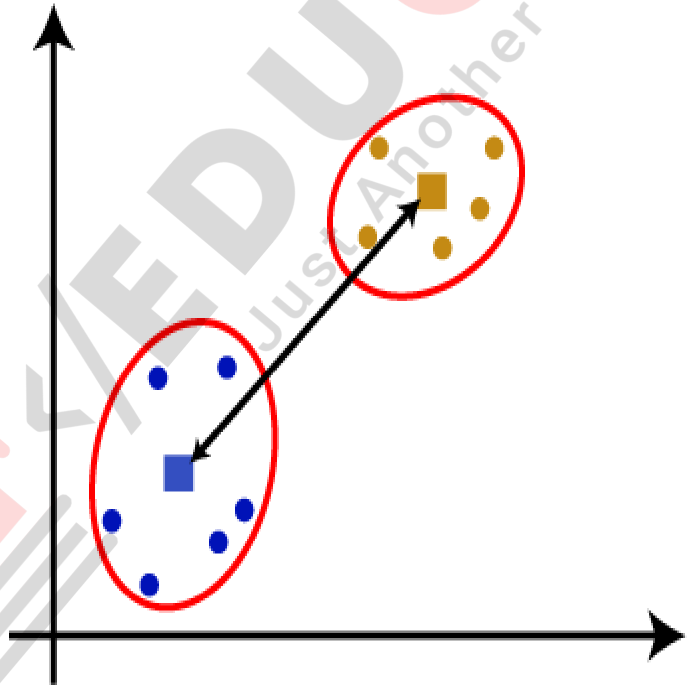
- **1. Single Linkage:** It is the Shortest Distance between the closest points of the clusters. Consider the below image:
- 

- **Complete Linkage:** It is the farthest distance between the two points of two different clusters. It is one of the popular linkage methods as it forms tighter clusters than single-linkage.



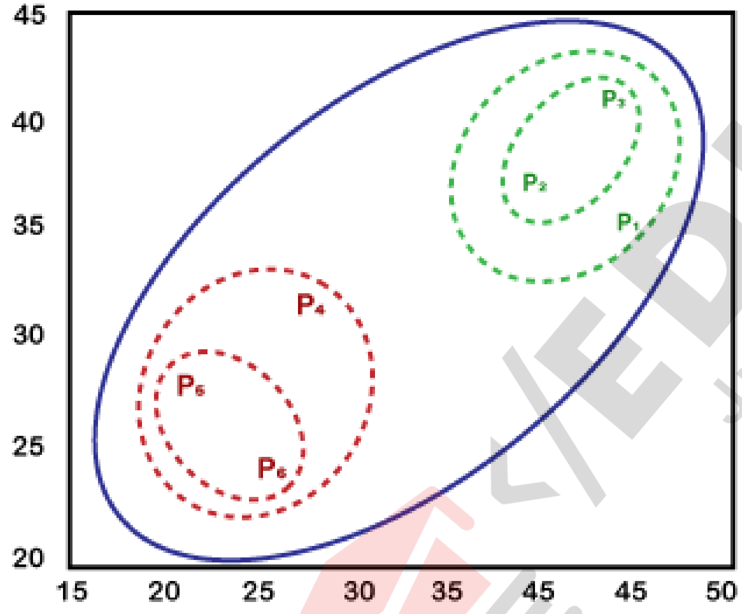
- **Average Linkage:** It is the linkage method in which the distance between each pair of datasets is added up and then divided by the total number of datasets to calculate the average distance between two clusters. It is also one of the most popular linkage methods.

- **Centroid Linkage:** It is the linkage method in which the distance between the centroid of the clusters is calculated. Consider the below image:

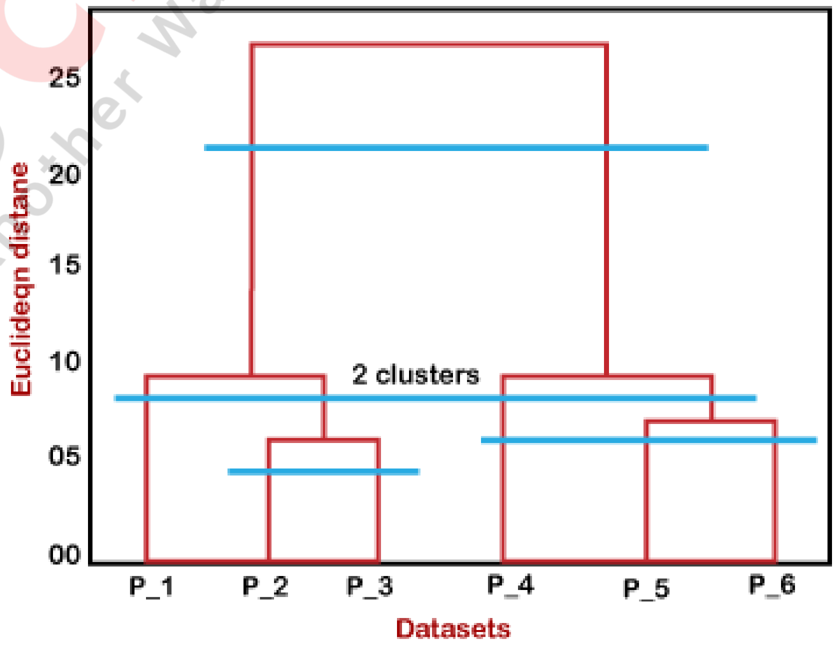


Dendrogram in Hierarchical clustering

- The dendrogram is a tree-like structure that is mainly used to store each step as a memory that the HC algorithm performs. In the dendrogram plot, the Y-axis shows the Euclidean distances between the data points, and the x-axis shows all the data points of the given dataset.
- The working of the dendrogram can be explained using the below diagram:



45
40
35
30
25
20
15
20
25
30
35
45
45
50
P₁
P₂
P₃
P₄
P₅
P₆



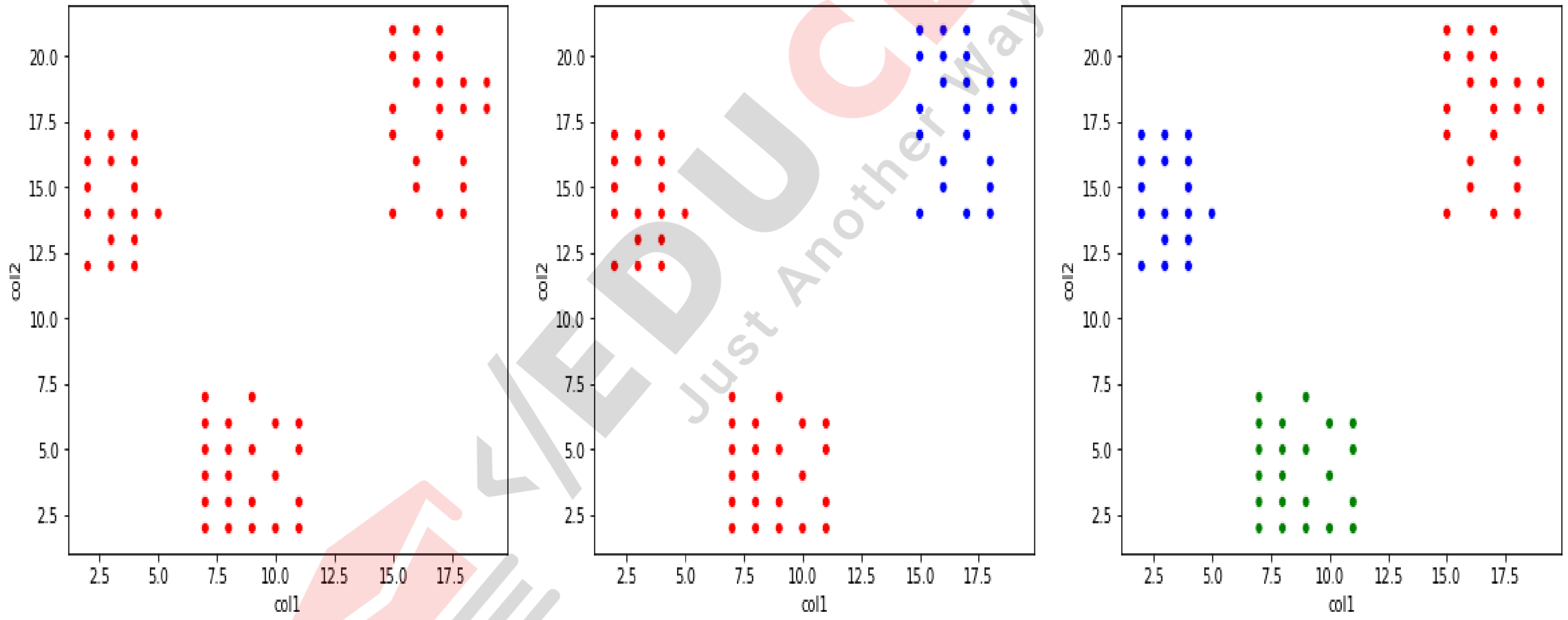
Euclideqn distane
25
20
15
10
05
00
2 clusters
P_1
P_2
P_3
P_4
P_5
P_6
Datasets

- In the above diagram, the left part is showing how clusters are created in agglomerative clustering, and the right part is showing the corresponding dendrogram.
- As we have discussed above, firstly, the datapoints P2 and P3 combine together and form a cluster, correspondingly a dendrogram is created, which connects P2 and P3 with a rectangular shape. The hight is decided according to the Euclidean distance between the data points.
- In the next step, P5 and P6 form a cluster, and the corresponding dendrogram is created. It is higher than of previous, as the Euclidean distance between P5 and P6 is a little bit greater than the P2 and P3.
- Again, two new dendrograms are created that combine P1, P2, and P3 in one dendrogram, and P4, P5, and P6, in another dendrogram.
- At last, the final dendrogram is created that combines all the data points together.

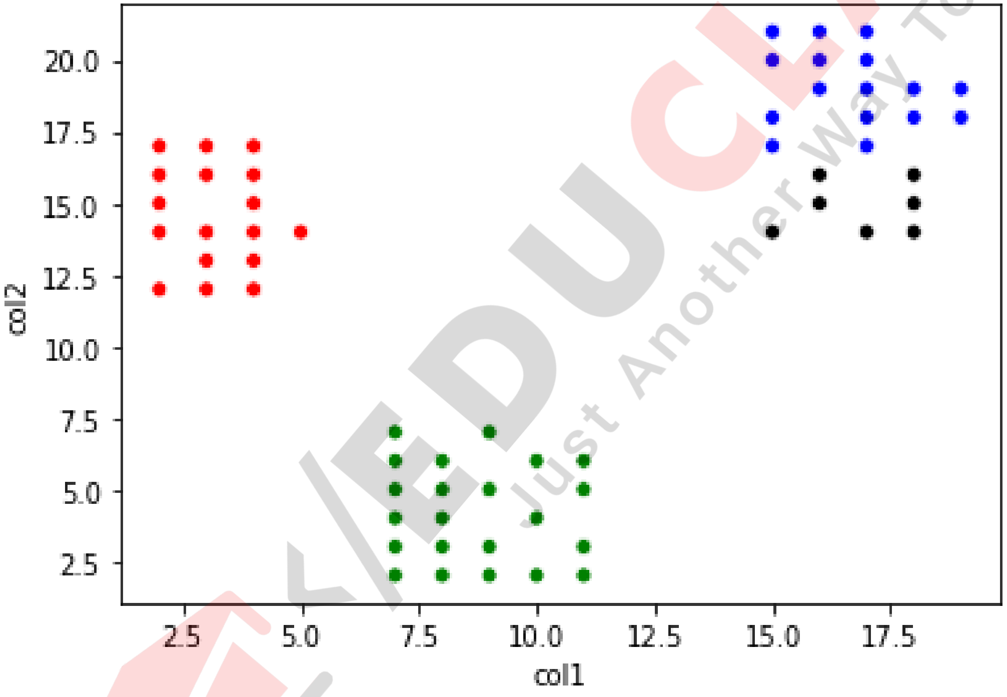
Divisive Clustering

- The divisive clustering algorithm is a top-down clustering approach, initially, all the points in the dataset belong to one cluster and split is performed recursively as one moves down the hierarchy.

- Steps of Divisive Clustering:
- Initially, all points in the dataset belong to one single cluster.
- Partition the cluster into two least similar cluster
- Proceed recursively to form new clusters until the desired number of clusters is obtained.


col2
col1
20.0
17.5
15.0
12.5
10.0
7.5
5.0
2.5

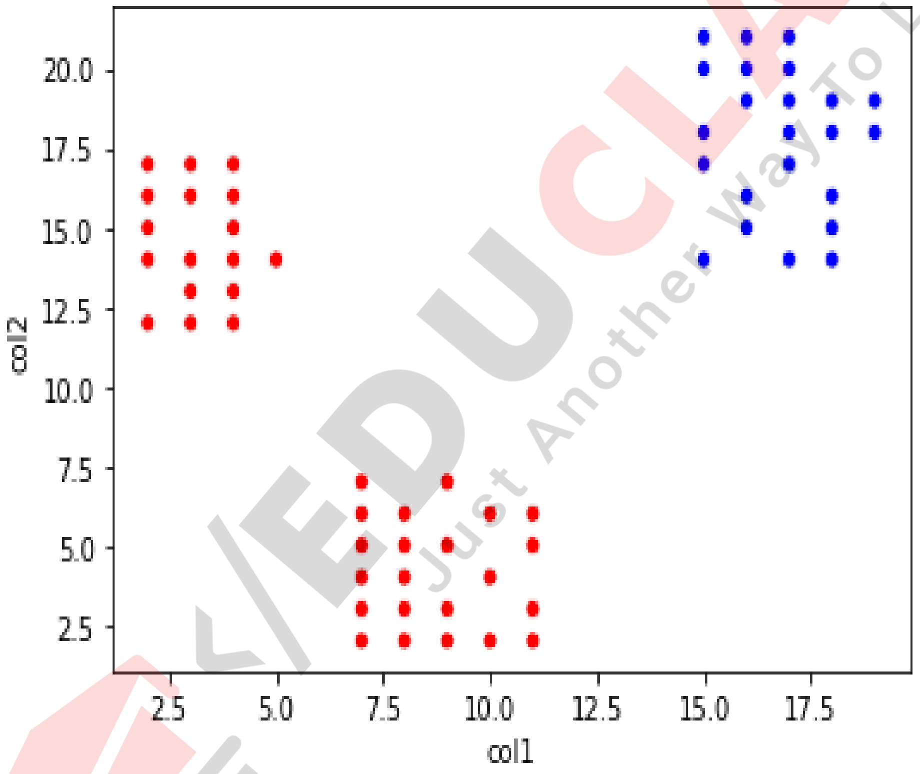
- In the above sample dataset, it is observed that there is 3 cluster that is far separated from each other. So we stopped after getting 3 clusters.
- Even if start separating further more clusters, below is the obtained result.



20.0
17.5
15.0
12.5
10.0
7.5
5.0
2.5
col2
2.5
5.0
7.5
10.0
12.5
15.0
17.5
col1

How to choose which cluster to split?

- Check the sum of squared errors of each cluster and choose the one with the largest value.
- In the below 2-dimension dataset, currently, the data points are separated into 2 clusters, for further separating it to form the 3rd cluster find the sum of squared errors (SSE) for each of the points in a red cluster and blue cluster.


col1
col2
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0

- The cluster with the largest SSE value is separated into 2 clusters, hence forming a new cluster. In the above image, it is observed red cluster has larger SSE so it is separated into 2 clusters forming 3 total clusters.

How to split the above-chosen cluster?

- Once we have decided to split which cluster, then the question arises on how to split the chosen cluster into 2 clusters. One way is to use Ward's criterion to chase for the largest reduction in the difference in the SSE criterion as a result of the split.

How to handle the noise or outlier?

- Due to the presence of outlier or noise, can result to form a new cluster of its own. To handle the noise in the dataset using a threshold to determine the termination criterion that means do not generate clusters that are too small.