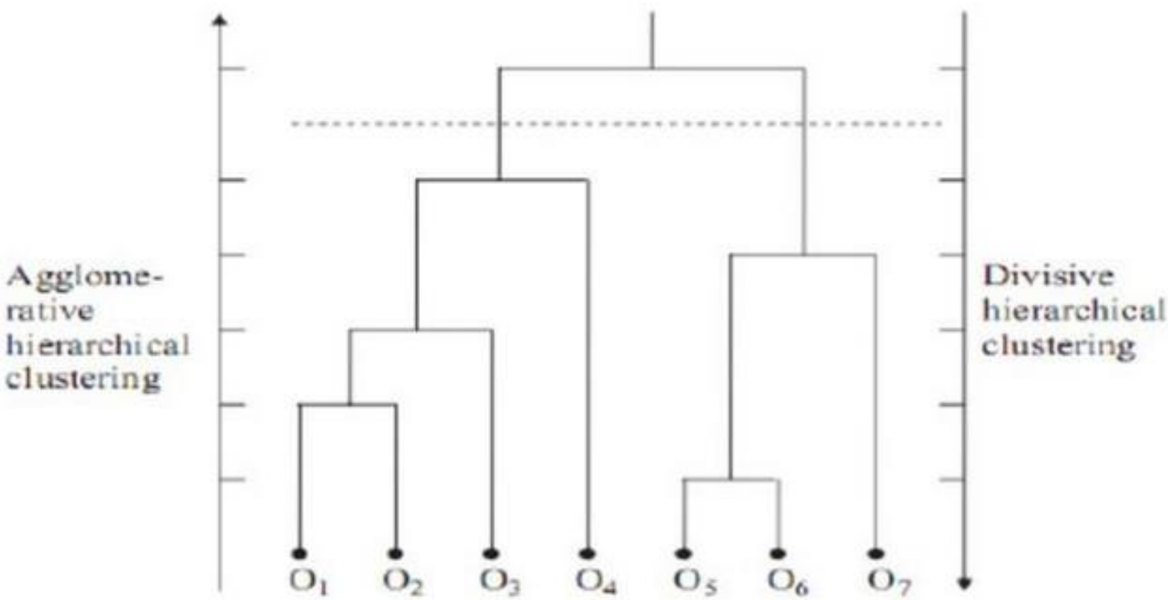
HIERARCHICAL CLUSTERING

A **Hierarchical clustering** method works via grouping data into a tree of clusters. Hierarchical clustering begins by treating every data point as a separate cluster. Then, it repeatedly executes the subsequent steps:

1. Identify the 2 clusters which can be closest together, and
2. Merge the 2 maximum comparable clusters. We need to continue these steps until all the clusters are merged together.

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 clusters are broken up (top-down view).

Hierarchical clustering is a method of cluster analysis in data mining that creates a hierarchical representation of the clusters in a dataset. The method starts by treating each data point as a separate cluster and then iteratively combines the closest clusters until a stopping criterion is reached. The result of hierarchical clustering is a tree-like structure, called a dendrogram, which illustrates the hierarchical relationships among the clusters.



Dendrogram for Clustering

Agglomerative

It uses a bottom-up approach. It starts with each object forming its own cluster and then iteratively merges the clusters according to their similarity to form large clusters.

It terminates either

- When certain clustering condition imposed by user is achieved or
- All clusters merge into a single cluster

```
given a dataset (d1, d2, d3, ....dN) of size N
# compute the distance matrix
for i=1 to N:
   # as the distance matrix is symmetric about
   # the primary diagonal so we compute only lower
   # part of the primary diagonal
   for j=1 to i:
      dis_mat[i][j] = distance[di, dj]
each data point is a singleton cluster
repeat
   merge the two cluster having minimum distance
   update the distance matrix
until only a single cluster remains
```

Why Agglomerative Hierarchical Clustering?

- It works from the dissimilarities between the objects to be grouped together. A type of dissimilarity can be suited to the subject studied and the nature of the data.
- One of the results is the dendrogram which shows the progressive grouping of the data. It is then possible to gain an idea of a suitable number of classes into which the data can be grouped.

Agglomerative Algorithm: Single Link

Single-nearest distance or single linkage is the agglomerative method that uses the distance between the closest members of the two clusters. We will now solve a problem to understand it better:

Question. Find the clusters using a **single link technique. Use Euclidean distance and draw the dendrogram.**

Sample No.	X	Y
P1	0.40	0.53
P2	0.22	0.38
P3	0.35	0.32
P4	0.26	0.19
P5	0.08	0.41
P6	0.45	0.30

Solution:

Step 1: Compute the distance matrix by:

$$d[(x,y)(a,b)] = \sqrt{(x-a)^2 + (y-b)^2}$$

So we have to find the Euclidean distance between each and every points, say we first find the euclidean distance between P1 and P2

$$d(P1,P2) = \sqrt{(0.4-0.22)^2 + (0.53-0.38)^2} =$$

$$\sqrt{(0.18)^2 + (0.15)^2} = \sqrt{0.0324 + 0.0225} = 0.23$$

So the DISTANCE MATRIX will look like this:

$$\begin{pmatrix} & P1 & P2 & P3 & P4 & P5 & P6 \\ P1 & 0 & & & & & \\ P2 & 0.23 & 0 & & & & \\ P3 & & & 0 & & & \\ P4 & & & & 0 & & \\ P5 & & & & & 0 & \\ P6 & & & & & & 0 \end{pmatrix}$$

Similarly, find the Euclidean distance for every point. But there is one point to focus on that the diagonal of the above distance matrix is a special point for us. The distance above and below the diagonal will be same. **For eg: d(P2, P5) is equivalent to d(P5, P2)**.

Therefore, the updated Distance Matrix for all points will be :

$$\begin{pmatrix} & P1 & P2 & P3 & P4 & P5 & P6 \\ P1 & 0 & & & & & \\ P2 & 0.23 & 0 & & & & \\ P3 & 0.22 & 0.14 & 0 & & & \\ P4 & 0.37 & 0.19 & 0.13 & 0 & & \\ P5 & 0.34 & 0.14 & 0.28 & 0.23 & 0 & \\ P6 & 0.24 & 0.24 & 0.10 & 0.22 & 0.39 & 0 \end{pmatrix}$$

Step 2: Merging the two closest members of the two clusters and finding the minimum element in distance matrix. Here the **minimum value is 0.10** and hence we combine P3 and P6 (as 0.10 came in the P6 row and P3 column). Now, form clusters of elements corresponding to the minimum value and update the distance matrix. To update the distance matrix:

min ((P3,P6), P1) = min ((P3,P1), (P6,P1)) = min (0.22,0.24) = 0.22
min ((P3,P6), P2) = min ((P3,P2), (P6,P2)) = min (0.14,0.24) = 0.14
min ((P3,P6), P4) = min ((P3,P4), (P6,P4)) = min (0.13,0.22) = 0.13
min ((P3,P6), P5) = min ((P3,P5), (P6,P5)) = min (0.28,0.39) = 0.28

Now we will update the Distance Matrix:

$$\begin{pmatrix} & P1 & P2 & P3,P6 & P4 & P5 \\ P1 & 0 & & & & \\ P2 & 0.23 & 0 & & & \\ P3,P6 & 0.22 & 0.14 & 0 & & \\ P4 & 0.37 & 0.19 & 0.13 & 0 & \\ P5 & 0.34 & 0.14 & 0.28 & 0.23 & 0 \end{pmatrix}$$

Now we will repeat the same process. Merge two closest members of the two clusters and find the minimum element in distance matrix. The **minimum value is 0.13** and hence we combine P3, P6 and P4. Now, form the clusters of elements corresponding to the minimum values and update the Distance matrix. In order to find, what we have to update in distance matrix,

min (((P3,P6) P4), P1) = min (((P3,P6), P1), (P4,P1)) = min (0.22,0.37) = 0.22

min (((P3,P6), P4), P2) = min (((P3,P6), P2), (P4,P2)) = min (0.14,0.19) = 0.14

min (((P3,P6), P4), P5) = min (((P3,P6), P5), (P4,P5)) = min (0.28,0.23) = 0.23

Now we will update the Distance Matrix:

$$\begin{pmatrix} & P1 & P2 & P3,P6,P4 & P5 \\ P1 & 0 & & & \\ P2 & 0.23 & 0 & & \\ P3,P6,P4 & 0.22 & 0.14 & 0 & \\ P5 & 0.34 & 0.14 & 0.23 & 0 \end{pmatrix}$$

Again repeating the same process: The **minimum value is 0.14** and hence we combine P2 and P5. Now, form cluster of elements corresponding to minimum value and update the distance matrix. To update the distance matrix:

min ((P2,P5), P1) = min ((P2,P1), (P5,P1)) = min (0.23, 0.34) = 0.23

min ((P2,P5), (P3,P6,P4)) = min ((P3,P6,P4), (P3,P6,P4)) = min (0.14. 0.23) = 0.14

Update Distance Matrix will be:

$$\begin{pmatrix} & P1 & P2,P5 & P3,P6,P4 \\ P1 & 0 & & \\ P2,P5 & 0.23 & 0 & \\ P3,P6,P4 & 0.22 & 0.14 & 0 \end{pmatrix}$$

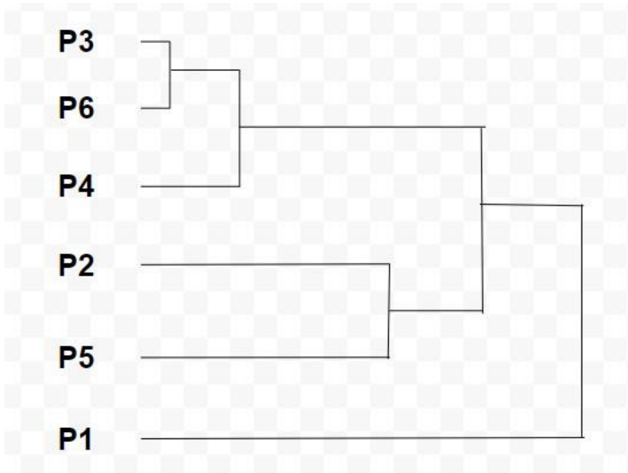
Again repeating the same process: The **minimum value is 0.14** and hence we combine P2,P5 and P3,P6,P4. Now, form cluster of elements corresponding to minimum value and update the distance matrix. To update the distance matrix:

min ((P2,P5,P3,P6,P4), P1) = min ((P2,P5), P1), ((P3,P6,P4), P1)) = min (0.23, 0.22) = 0.22

Updated Distance Matrix will be:

$$\begin{pmatrix} & P1 & P2,P5,P3,P6,P4 \\ P1 & 0 & \\ P2,P5,P3,P6,P4 & 0.22 & 0 \end{pmatrix}$$

So now we have reached to the solution finally, the dendrogram for those question will be as follows:



Dendogram of the cluster formed

2. Agglomerative Algorithm: Complete Link

In this algorithm, complete farthest distance or complete linkage is the agglomerative method that uses the distance between the members that are the farthest apart.

Question. For the given set of points, identify clusters using the complete link agglomerative clustering

Sample No	X	Y
P1	1	1
P2	1.5	1.5
P3	5	5
P4	3	4
P5	4	4
P6	3	3.5

Solution.

Step 1: Compute the distance matrix by:

$$d[(x,y)(a,b)] = \sqrt{(x-a)^2 + (y-b)^2}$$

So we have to find the euclidean distance between each and every point, say we first find the euclidean distance between P1 and P2. (FORMULA FOR CALCULATING THE DISTANCE IS SAME AS ABOVE)

Say the Distance Matrix for some points is:

$$\begin{array}{c} \\ P1 \\ P2 \\ P3 \\ P4 \\ P5 \\ P6 \end{array} \begin{pmatrix} P1 & P2 & P3 & P4 & P5 & P6 \\ 0 & & & & & \\ 0.71 & 0 & & & & \\ 5.66 & 4.95 & 0 & & & \\ 3.6 & 2.92 & 2.24 & 0 & & \\ 4.24 & 3.53 & 1.41 & 1.0 & 0 & \\ 3.20 & 2.5 & 2.5 & 0.5 & 1.12 & 0 \end{pmatrix}$$

Step 2: Merging the two closest members of the two clusters and finding the minimum element in distance matrix. So, the **minimum value is 0.5** and hence we combine P4 and P6. To update the distance matrix,

max (d(P4,P6), P1) = max (d(P4,P1), d(P6,P1)) = **max** (3.6, 3.2) = 3.6

max (d(P4,P6), P2) = max (d(P4,P2), d(P6,P2)) = max (2.92, 2.5) = 2.92

max (d(P4,P6), P3) = max (d(P4,P3), d(P6,P3)) = max (2.24, 2.5) = 2.5

max (d(P4,P6), P5) = max (d(P4,P5), d(P6,P5)) = max (1.0, 1.12) = 1.12

Updated distance matrix is:

$$\begin{pmatrix} & P1 & P2 & P3 & P4,P6 & P5 \\ P1 & 0 & & & & \\ P2 & 0.71 & 0 & & & \\ P3 & 5.66 & 4.95 & 0 & & \\ P4,P6 & 3.6 & 2.92 & 2.5 & 0 & \\ P5 & 4.24 & 3.53 & 1.41 & 1.12 & 0 \end{pmatrix}$$

Again, merging the two closest members of the two clusters and finding the minimum element in distance matrix. We get the **minimum value as 0.71** and hence we combine P1 and P2. To update the distance matrix,

max (d(P1, P2), P3) = max (d(P1, P3), d(P2, P3)) = max (5.66, 4.95) = 5.66

max (d(P1,P2), (P4,P6)) = max (d(P1, P4, P6), d(P2, P4, P6)) = max (3.6, 2.92) = 3.6

max (d(P1,P2), P5) = max (d(P1, P5), d(P2, P5)) = max (4.24, 3.53) = 4.24

Updated distance matrix is:

$$\begin{pmatrix} & P1,P2 & P3 & P4,P6 & P5 \\ P1,P2 & 0 & & & \\ P3 & 5.66 & 0 & & \\ P4,P6 & 3.6 & 2.5 & 0 & \\ P5 & 4.24 & 1.41 & 1.12 & 0 \end{pmatrix}$$

Again, merging the two closest members of the two clusters and finding the minimum element in distance matrix. We get the **minimum value as 1.12** and hence we combine P4, P6 and P5. To update the distance matrix,

max (d(P4,P6,P5), (P1,P2)) = max (d(P4,P6,P1,P2), d(P5,P1,P2)) = max (3.6, 4.24) = 4.24

max (d(P4,P6,P5), P3) = max (d(P4,P6,P3), d(P5, P3)) = max (2.5, 1.41) = 2.5

Updated distance matrix is:

$$\begin{pmatrix} & P1,P2 & P3 & P4,P6,P5 \\ P1,P2 & 0 & & \\ P3 & 5.66 & 0 & \\ P4,P6,P5 & 4.24 & 2.5 & 0 \end{pmatrix}$$

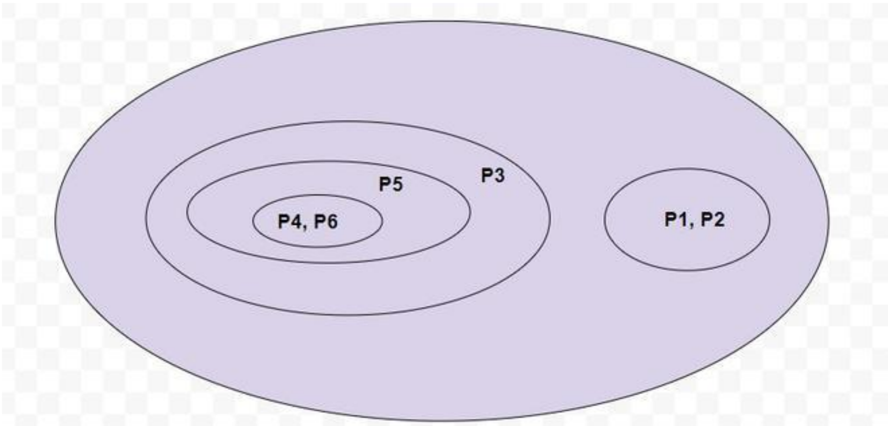
Again, merging the two closest members of the two clusters and finding the minimum element in distance matrix. We get the **minimum value as 2.5** and hence combine P4,P6,P5 and P3. to update the distance matrix,

min (d(P4,P6,P5,P3), (P1,P2)) = max (d(P4,P6,P5,P1,P2), d(P3,P1,P2)) = mac (3.6, 5.66) = 5.66

Updated distance matrix is:

$$\begin{pmatrix} & P1,P2 & P4,P6,P5,P3 \\ P1,P2 & 0 & \\ P4,P6,P5,P3 & 5.66 & 0 \end{pmatrix}$$

So now we have reached to the solution finally, the dendrogram for those question will be as follows:



Cluster formed after merging all data points

3. Agglomerative Algorithm: Average Link

Average-average distance or average linkage is the method that involves looking at the distances between all pairs and averages all of these distances. This is also called **Universal Pair Group Mean Averaging.**

Question. For the points given in the previous question, identify clusters using average link agglomerative clustering

Solution:

We have to first find the Distance Matrix, as we have picked the same question the distance matrix will be same as above:

$$\begin{pmatrix} & P1 & P2 & P3 & P4 & P5 & P6 \\ P1 & 0 & & & & & \\ P2 & 0.71 & 0 & & & & \\ P3 & 5.66 & 4.95 & 0 & & & \\ P4 & 3.6 & 2.92 & 2.24 & 0 & & \\ P5 & 4.24 & 3.53 & 1.41 & 1.0 & 0 & \\ P6 & 3.20 & 2.5 & 2.5 & 0.5 & 1.12 & 0 \end{pmatrix}$$

Merging two closest members of the two clusters and finding the minimum elements in distance matrix. We get the minimum value as 0.5 and hence we combine P4 and P6. To update the distance matrix :

average (d(P4,P6), P1) = average (d(P4,P1), d(P6,P1)) = average (3.6, 3.20) = 3.4
average (d(P4,P6), P2) = average (d(P4,P2), d(P6,P2)) = average (2.92, 2.5) = 2.71
average (d(P4,P6), P3) = average (d(P4,P3), d(P6,P3)) = average (2.24, 2.5) = 2.37
average (d(P4,P6), P5) = average (d(P4,P5), d(P6,P5)) = average (1.0, 1.12) = 1.06

Updated distance matrix is:

$$\begin{array}{cc} & \begin{array}{ccccc} P1 & P2 & P3 & P4,P6 & P5 \end{array} \\ \begin{array}{c} P1 \\ P2 \\ P3 \\ P4,P6 \\ P5 \end{array} & \begin{pmatrix} 0 & & & & \\ 0.71 & 0 & & & \\ 5.66 & 4.95 & 0 & & \\ 3.4 & 2.71 & 2.37 & 0 & \\ 4.24 & 3.53 & 1.41 & 1.06 & 0 \end{pmatrix} \end{array}$$

Merging two closest members of the two clusters and finding the minimum elements in distance matrix. We get the minimum value as 0.71 and hence we combine P1 and P2. To update the distance matrix:

average (d(P1,P2), P3) = average (d(P1,P3), d(P2,P3)) = average (5.66, 4.95) = 5.31

average (d(P1,P2), (P4,P6)) = average (d(P1,P4,P6), d(P2,P4,P6)) = average (3.2, 2.71) = 2.96

average (d(P1,P2), P5) = average (d(P1,P5), d(P2,P5)) = average (4.24, 3.53) = 3.89

Updated distance matrix is:

$$\begin{array}{cc} & \begin{array}{cccc} P1,P2 & P3 & P4,P6 & P5 \end{array} \\ \begin{array}{c} P1,P2 \\ P3 \\ P4,P6 \\ P5 \end{array} & \begin{pmatrix} 0 & & & \\ 5.31 & 0 & & \\ 2.96 & 2.5 & 0 & \\ 3.89 & 1.41 & 1.12 & 0 \end{pmatrix} \end{array}$$

Merging two closest members of the two clusters and finding the minimum elements in distance matrix. We get the minimum value as 1.12 and hence we combine P4,P6 and P5. To update the distance matrix:

average (d(P4,P6,P5), (P1,P2)) = average (d(P4,P6,P1,P2), d(P5,P1,P2)) = average (2.96, 3.89) = 3.43

average (d(P4,P6,P5), P3) = average (d(P4,P6,P3), d(P5,P3)) = average (2.5, 1.41) = 1.96

Updated distance matrix is:

$$\begin{array}{cc} & \begin{array}{ccc} P1,P2 & P3 & P4,P6,P5 \end{array} \\ \begin{array}{c} P1,P2 \\ P3 \\ P4,P6,P5 \end{array} & \begin{pmatrix} 0 & & \\ 5.66 & 0 & \\ 3.43 & 1.96 & 0 \end{pmatrix} \end{array}$$

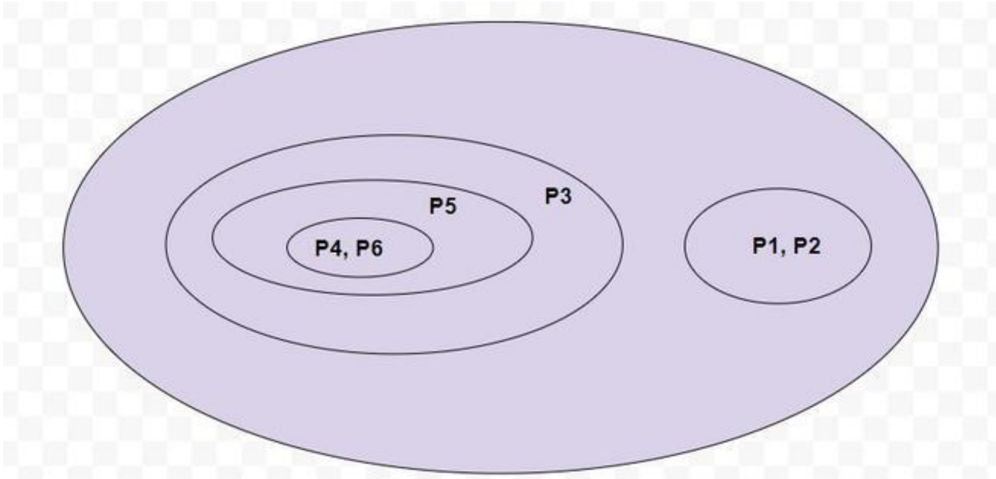
Merging two closest members of the two clusters and finding the minimum elements in distance matrix. We get the minimum value as 1.96 and hence we combine P4,P6,P5 and P3. To update the distance matrix:

average (d(P4,P6,P5,P3), (P1,P2)) = average (d(P4,P6,P5,P1,P2), d(P3,P1P2)) = average (3.43, 5.66) = 4.55

Updated distance matrix is:

$$\begin{array}{cc} & \begin{array}{cc} P1,P2 & P4,P6,P5,P3 \end{array} \\ \begin{array}{c} P1,P2 \\ P4,P6,P5,P3 \end{array} & \begin{pmatrix} 0 & \\ 4.55 & 0 \end{pmatrix} \end{array}$$

So, the final cluster can be drawn is shown as:



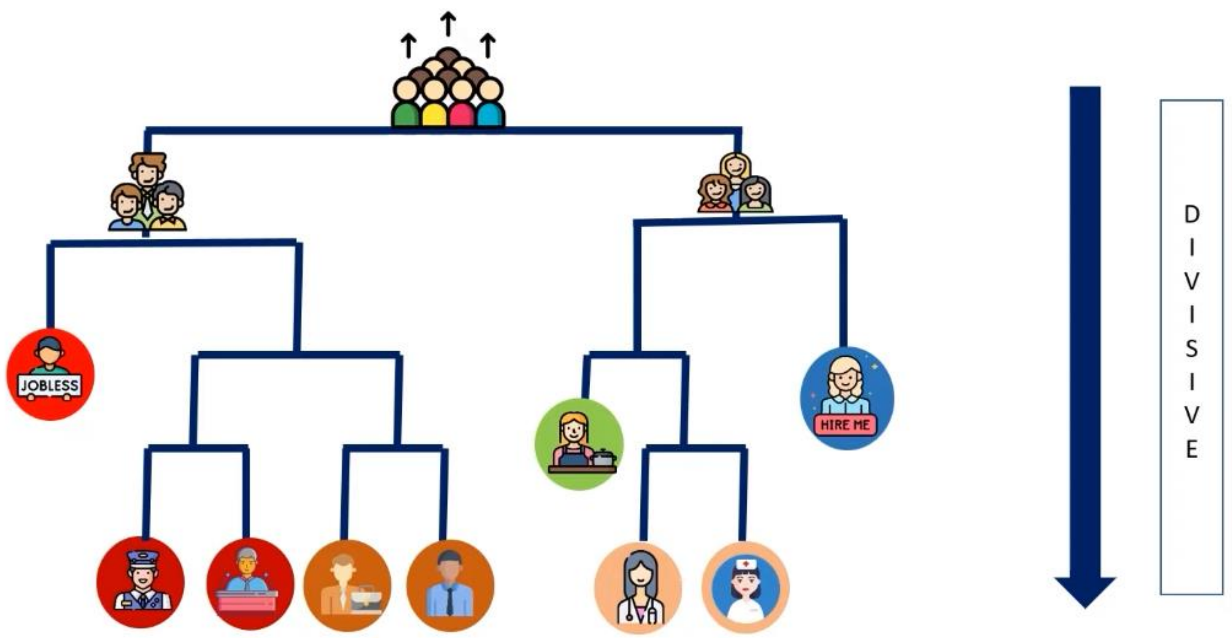
Cluster formed after merging all data points

Divisive Hierarchical Clustering

This is a "top-down" approach: all observations start in one cluster, and splits are performed recursively as we move from top to bottom.

Divisive clustering is also a type of hierarchical clustering that is used to create clusters of data points. It is an unsupervised learning algorithm that begins by placing all the data points in a single cluster and then progressively splits the clusters until each data point is in its own cluster. Divisive clustering is useful for analyzing datasets that may have complex structures or patterns, as it can help identify clusters that may not be obvious at first glance.

Divisive clustering works by first assigning all the data points to one cluster. Then, it looks for ways to split this cluster into two or more smaller clusters. This process continues until each data point is in its own cluster. For example, consider the following image.



Divisive Clustering Example (Image taken from YouTube video on Hierarchical clustering by Edureka)

- In the given example, the divisive clustering algorithm first considers the entire population as a single cluster. Then, it divides the population into two clusters containing males and females.
- Next, the algorithm divides the clusters containing males and females on the basis of whether they are employed or unemployed.
- The algorithm keeps dividing the clusters into smaller clusters until it reaches a point where the clusters consist of a single person.

We are going to apply the divisive clustering technique for the similar demo dataset described below again,

Observations	x	y
P1	0.40	0.53
P2	0.22	0.38
P3	0.35	0.32
P4	0.26	0.19
P5	0.08	0.41
P6	0.45	0.30

Step-1:

Similar to agglomerative method, we consider the Euclidean distance measure for calculating distance between two points and use complete-linkage method (opposite of single-linkage method to receive the similar result, as it is top-down approach). For example, we can calculate the distance between P1 and P2 in the following way,

$$D(x,y) = D(P1, P2) = \sqrt{(0.40 - 0.22)^2 + (0.53 - 0.38)^2} = 0.23$$

Similarly, we will calculate the distance among each pair of data points and it can be represented in the form of distance matrix below. This kind of distance matrix is also called proximity matrix in hierarchical clustering.

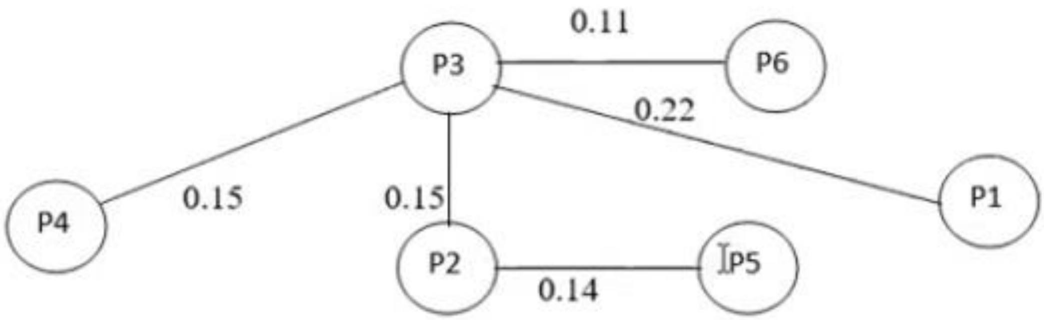
	P1	P2	P3	P4	P5	P6
P1	0					
P2	0.23	0				
P3	0.22	0.15	0			
P4	0.37	0.20	0.15	0		
P5	0.34	0.14	0.28	0.29	0	
P6	0.23	0.25	0.11	0.22	0.39	0

Step-2:

Now, we compute a minimum spanning tree (MST) from the above proximity matrix by using either Kruskal's or Prim's algorithm. We are using here Prim's algorithm for sake of simplicity and arrange all the distances in ascending order.

Edge	Cost
P3,P6	0.11
P2,P5	0.14
P2,P3	0.15
P3,P4	0.15
P2,P4	0.20
P1,P3	0.22
P4,P6	0.22
P1,P2	0.23
P1,P6	0.23
P2,P6	0.25
P3,P5	0.28
P4,P5	0.29
P1,P5	0.34
P1,P4	0.37
P5,P6	0.39

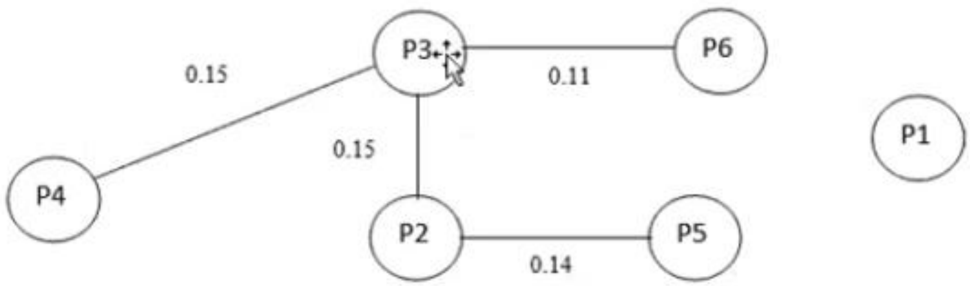
Now, the final MST can be constructed by connecting all the edges between points (starting from the minimum) until all the points will be participated and no closed loop or circuit will be formed. So, following the greedy approach, we first connect P3, P6 with the most lowest distance (0.11), then connect P2 & P5 with the second lowest distance 0.14, then connect P2 with P3 with the third lowest distance 0.15 and so on.



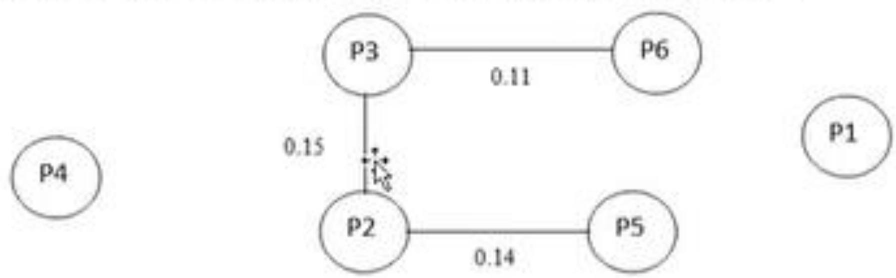
Step-3:

Therefore, we apply complete-linkage method to break the edges of the final MST according to the maximum cost or distance gradually and we are able to create new clusters by considering those largest distances.

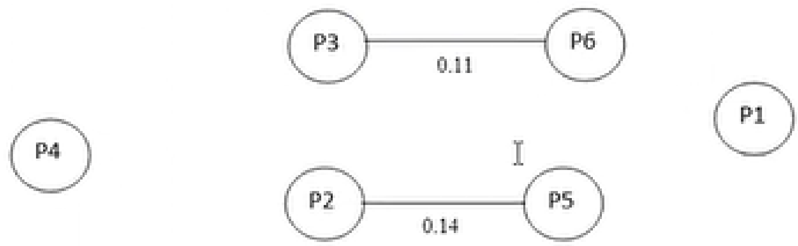
i. At first we break the edge whose cost is 0.22, i.e. (P1, P3). So, two clusters get formed cluster-1 consists of P1 and cluster-2 consists of (P2, P3, P4, P5, P6).



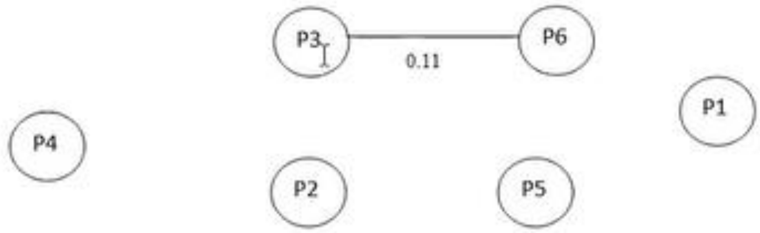
ii. Now, we will seek for the next maximum distance greedily and we will choose (P3,P4) as the next highest cost i.e. 0.15 and break that edge to create three separate clusters, cluster-1={P1}, cluster-2={P2,P3,P5,P6}, cluster-3={P4}. We will continue this splitting iteration until each new cluster contains only a single object.



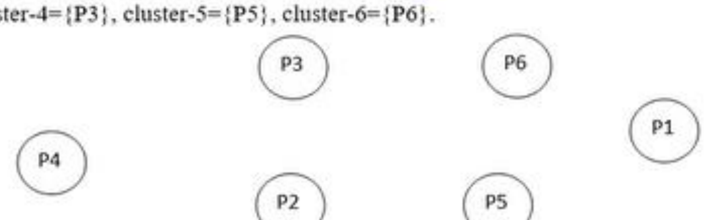
iii. Now, the next maximum cost is (P2,P3) i.e. 0.15 and we will break that edge. So, our final four clusters will be, cluster-1={P1}, cluster-2={P2,P5}, cluster-3={P4}, cluster-4={P3,P6}.

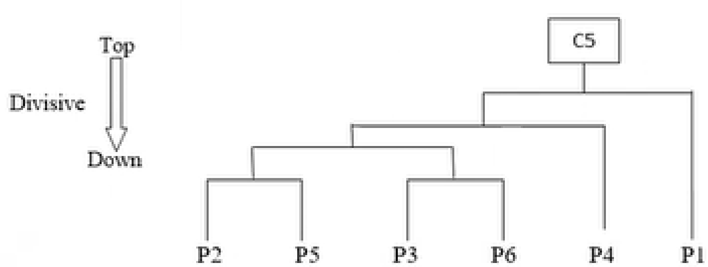


iv. Then the next maximum distance or cost is 0.14 i.e. (P2,P5) and we will break that edge. So, our final five clusters will be, cluster-1={P1}, cluster-2={P2}, cluster-3={P4}, cluster-4={P3,P6}, cluster-5={P5}.



v. At the final step, we will break the last edge (P3,P6) whose cost is 0.11 and separate all the data points into individual clusters.
So, our final six clusters will be, cluster-1={P1}, cluster-2={P2}, cluster-3={P4}, cluster-4={P3}, cluster-5={P5}, cluster-6={P6}.





- ✓ The computational complexity of hierarchical clustering is $O(n^2)$ and space complexity is $O(n^3)$ whereas the computational complexity of K-means algorithm is O(n). Therefore, hierarchical clustering is not suitable to handle large data, as it is too much computationally expensive.

Divisive Approach by Minimum Spanning Tree (MST) Concept

1	0				
2	9	0			
3	3	7	0		
4	6	5	9	0	
5	11	10	2	8	0
	1	**2**	**3**	**4**	**5**

Draw MST by either Kruskal's or Prim's Algorithm

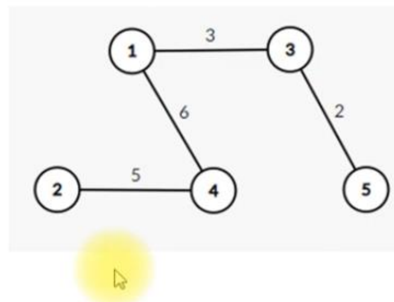
Kruslak's Method: Arrange edge in ascending order of their cost.

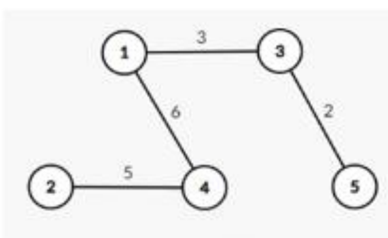
For V-1=E, we may stop constructing MST, as it covers all vertices

Edge	Cost
3-5	2
1-3	3
2-4	5
1-4	6
2-3	7
4-5	8
1-2	9
3-4	9
2-5	10
4-5	11

Minimum Spanning Tree (MST)

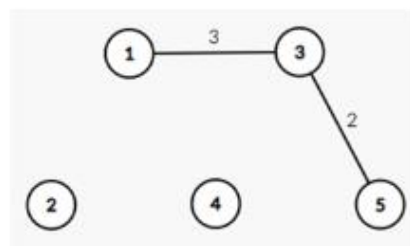
Edge	Cost
3-5	2
1-3	3
2-4	5
1-4	6



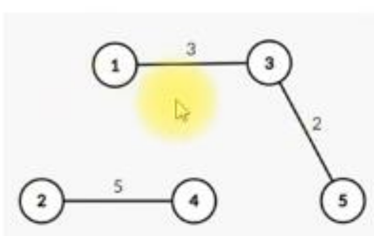


Break edge **{1,4}** having cost **6**.

Two clusters get formed **{2,4}** and **{1,3,5}**



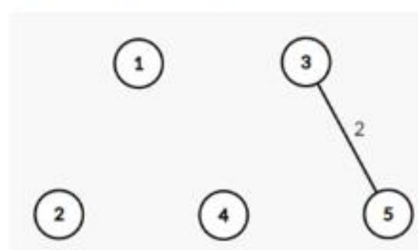
Break edge **{1,3}** having cost **3**.
Four clusters get formed
{2}, **{4}**, **{1}** and **{3,5}**



Break edge **{2,4}** having cost **5**.

Three clusters get formed

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



Break edge **{3,5}** having cost **2**.

Five clusters get formed

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

Divisive Approach by Minimum Spanning Tree (MST) Concept

1	0			
2	11	0		
3	10	9	0	
4	9	6	5	0
	1	**2**	**3**	**4**

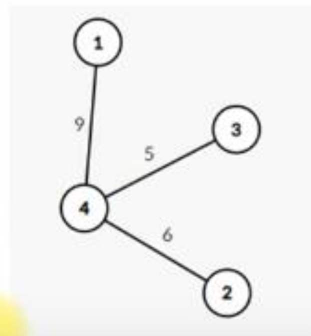
Draw MST by either Kruskal's or Prim's Algorithm

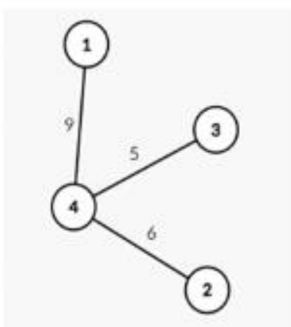
Kruslak's Method: Arrange edge in ascending order of their cost.

Edge	Cost
3-4	5
2-4	6
2-3	9
1-4	9
1-3	10
1-2	11

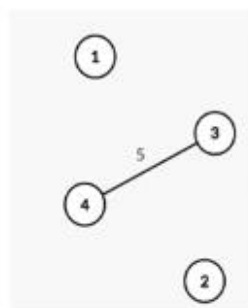
Minimum Spanning Tree (MST)

Edge	Cost
3-4	5
2-4	6
1-4	9





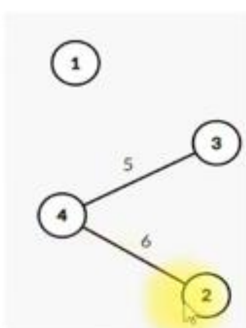
Break edge **{4,2}** having cost **6**. Three clusters get formed **{1}**, **{2}** and **{3,4}**



Break edge **{1,4}** having cost **9**.

Two clusters get formed

{2,3,4} and **{1}**



Break edge **{3,4}** having cost **5**. Four clusters get formed **{1}**, **{2}**, **{3}**, **{4}**

- Additional Exercise: Try for yourself

A	0					
B	16	0				
C	47	37	0			
D	72	57	40	0		
E	77	65	30	31	0	
F	79	66	35	23	10	0
	A	**B**	**C**	**D**	**E**	**F**

A	0				
B	4	0			
C	4	4	0		
D	5	3	5	0	
E	7	3	7	2	0
	A	**B**	**C**	**D**	**E**

Difference between agglomerative clustering and Divisive clustering:

SL	Parameters	Agglomerative Clustering	Divisive Clustering
1.	Category	Bottom-up approach	Top-down approach
2.	Approach	each data point starts in its own cluster, and the algorithm recursively merges the closest pairs of clusters until a single cluster containing all the data points is obtained.	all data points start in a single cluster, and the algorithm recursively splits the cluster into smaller sub-clusters until each data point is in its own cluster.
3.	Complexity level	Agglomerative clustering is generally more computationally expensive, especially for large datasets as this approach requires the calculation of all pairwise distances between data points, which can be computationally expensive.	Comparatively less expensive as divisive clustering only requires the calculation of distances between sub-clusters, which can reduce the computational burden.
4.	Outliers	Agglomerative clustering can handle outliers better than divisive clustering since outliers can be absorbed into larger clusters	divisive clustering may create sub-clusters around outliers, leading to suboptimal clustering results.
5.	Interpretability	Agglomerative clustering tends to produce more interpretable results since the dendrogram shows the merging process of the clusters, and the user can choose the number of clusters based on the desired level of granularity.	divisive clustering can be more difficult to interpret since the dendrogram shows the splitting process of the clusters, and the user must choose a stopping criterion to determine the number of clusters.
6.	Implementation	Scikit-learn provides multiple linkage methods for agglomerative clustering, such as “ward,” “complete,” “average,” and “single,”	divisive clustering is not currently implemented in Scikit-learn.
7.	Example	Here are some of the applications in which Agglomerative Clustering is used : Image segmentation, Customer segmentation, Social network analysis, Document clustering, Genetics, genomics, etc., and many more.	Here are some of the applications in which Divisive Clustering is used : Market segmentation, Anomaly detection, Biological classification, Natural language processing, etc.

Advantages of Hierarchical Clustering

Hierarchical clustering is a widely used technique in data analysis, which involves the grouping of objects into clusters based on their similarity. This method of clustering is advantageous in a variety of ways and can be used to solve various types of problems. Here are 10 advantages of hierarchical clustering:

1. Robustness: Hierarchical clustering is more robust than other methods since it does not require a predetermined number of clusters to be specified. Instead, it creates hierarchical clusters based on the similarity between the objects, which makes it more reliable and accurate.
2. Easy to interpret: Hierarchical clustering produces a tree-like structure that is easy to interpret and understand. This makes it ideal for data analysis as it can provide insights into the data without requiring complex algorithms or deep learning models.
3. Flexible: Hierarchical clustering is a flexible method that can be used on any type of data. It can also be used with different types of similarity functions and distance measures, allowing for customization based on the application at hand.
4. Scalable: Hierarchical clustering is a scalable method that can easily handle large datasets without becoming computationally expensive or time-consuming. This makes it suitable for applications such as customer segmentation where large datasets need to be processed quickly and accurately.
5. Visualization: Hierarchical clustering produces a visual tree structure that can be used to gain insights into the data quickly and easily. This makes it an ideal choice for exploratory data analysis as it allows researchers to gain an understanding of the data at a glance.
6. Versatile: Hierarchical clustering can be used for both supervised and unsupervised learning tasks, making it extremely versatile in its range of applications.
7. Easier to apply: Since there are no parameters to specify in hierarchical clustering, it is much easier to apply compared to other methods such as k-means clustering or k-prototypes clustering. This makes it ideal for novice users who need to quickly apply clustering techniques with minimal effort.
8. Greater accuracy: Hierarchical clustering often tends to produce superior results compared to other methods of clustering due to its ability to create more meaningful clusters based on similarities between objects rather than arbitrary boundaries set by cluster centroids or other parameters.
9. Non-linearity: Agglomerative or divisive clustering can handle non-linear datasets better than other methods, which makes it suitable for cases where linearity cannot be assumed in the dataset being analyzed.
10. Multiple-level output: By producing a hierarchical tree structure, hierarchical clustering provides multiple levels of output which allows users to view data at different levels of detail depending on their needs. This flexibility makes it an attractive choice in many situations where multiple levels of analysis are required.

Disadvantages of Hierarchical Clustering

While hierarchical clustering is a powerful tool for discovering patterns and relationships in data sets, it also has its drawbacks. Here are 10 disadvantages of hierarchical clustering:

1. It is sensitive to outliers. Outliers have a significant influence on the clusters that are formed, and can even cause incorrect results if the data set contains these types of data points.
2. Hierarchical clustering is computationally expensive. The time required to run the algorithm increases exponentially as the number of data points increases, making it difficult to use for large datasets.
3. The results of Agglomerative or divisive clustering can sometimes be difficult to interpret the results due to its complexity. The dendrogram representation of the clusters can be hard to understand and visualize, making it difficult to draw meaningful conclusions from the results.
4. It does not guarantee optimal results or the best possible clusterings. Since it is an unsupervised learning algorithm, it relies on the researcher's judgment and experience to assess the quality of the results.
5. Hierarchical clustering methods require a predetermined number of clusters before they can begin clustering, which may not be known beforehand. This makes it difficult to use in certain applications where this information is not available.
6. The results produced by hierarchical clustering may be dependent on the order in which the data points are processed, making it difficult to reproduce or generalize them for other datasets with similar characteristics.
7. The algorithm does not provide any flexibility when dealing with multi-dimensional data sets since all variables must be treated equally in order for accurate results to be obtained.
8. Agglomerative or divisive clustering is prone to producing overlapping clusters, where different groups of data points may share common characteristics and thus be grouped together even though they should not belong to the same cluster.
9. Hierarchical clustering requires manual intervention for selecting the appropriate number of clusters, which can be time-consuming and prone to errors if done incorrectly or without proper knowledge about the data set being analyzed.
10. It cannot handle categorical variables effectively since they cannot be converted into numerical values and thus will fail to produce meaningful clusters if used in hierarchical clustering algorithms.

Applications of Hierarchical Clustering

Hierarchical clustering is a type of unsupervised machine learning that can be used for many different applications. It is used to group similar data points into clusters, which can then be used for further analysis. Here are 10 applications of hierarchical clustering:

1. Customer segmentation: Agglomerative or divisive clustering can be used to group customers into different clusters based on their demographic, spending, and other characteristics. This can be used to better understand customer behavior and to target marketing campaigns.
2. Image segmentation: Hierarchical clustering can be used to segment images into different regions, which can then be used for further analysis.
3. Text analysis: Hierarchical clustering can be used to group text documents based on their content, which can then be used for text mining or text classification tasks.

4. Gene expression analysis: Hierarchical clustering can be used to group genes based on their expression levels, which can then be used to better understand gene expression patterns.
5. Anomaly detection: Hierarchical clustering can be used to detect anomalies in data, which can then be used for fraud detection or other tasks.
6. Recommendation systems: Hierarchical clustering can be used to group users based on their preferences, which can then be used to recommend items to them.
7. Risk assessment: Agglomerative or divisive clustering can be used to group different risk factors in order to better understand the overall risk of a portfolio.
8. Network analysis: Hierarchical clustering can be used to group nodes in a network based on their connections, which can then be used to better understand network structures.
9. Market segmentation: Hierarchical clustering can be used to group markets into different segments, which can then be used to target different products or services to them.
10. Outlier detection: Hierarchical clustering can be used to detect outliers in data, which can then be used for further analysis.

REFERENCE

- https://www.geeksforgeeks.org/agglomerative-methods-in-machine-learning/
- https://www.youtube.com/watch?v=YH0r47m0kFM
- https://www.youtube.com/watch?v=d1qAwe8hthM
- https://www.youtube.com/watch?v=_nv6UdYIMhc
- https://www.youtube.com/watch?v=p-1NFmePDp4
- https://www.youtube.com/watch?v=4GbhWbMJLMY

Minimum Spanning Tree

- https://www.youtube.com/watch?v=_KX8GDvRzBc
- https://www.youtube.com/watch?v=huQojf2tevI&t=332s
- https://www.youtube.com/watch?v=Gd92jSu_cZk