lOMoARcPSD|17241975

Name : Khan Mohd Aaquib Sameer

Roll no : 22CO21

Batch : 2

**Experiment 8**

**Aim:** Implementation of any one Hierarchical Clustering method

# **Theory:**

**Hierarchical Clustering:**

Hierarchical clustering is another unsupervised machine learning algorithm, which is used to group the unlabeled datasets into a cluster and also known as **hierarchical cluster analysis** or HCA. In this algorithm, we develop the hierarchy of clusters in the form of a tree, and this tree- shaped structure is known as the **dendrogram**.

Hierarchical Clustering is of two types:

1. **Divisive: -**

**Divisive Hierarchical Clustering** is also termed as a top-down clustering approach. In this technique, entire data or observation is assigned to a single cluster. The cluster is further split until there is one cluster for each data or observation.

1. **Agglomerative Hierarchical Clustering: -**

**Agglomerative Hierarchical Clustering** is popularly known as a bottom-up approach, wherein each data or observation is treated as its cluster. A pair of clusters are combined until all clusters are merged into one big cluster that contains all the data.

# **Agglomerative Hierarchical clustering Algorithm:**

Let X = {x1, x2, x3, ..., xn} be the set of data points.

1. Begin with the disjoint clustering having level L(0) = 0 and sequence number m = 0.
2. Find the least distance pair of clusters in the current clustering, say pair (r), (s), according to d[(r), (s)] = min d[(i), (j)] where the minimum is over all pairs of clusters in the current clustering.
3. Increment the sequence number: m = m +1. Merge clusters (r) and (s) into a single cluster to form the next clustering m. Set the level of this clustering to L(m) = d[(r), (s)].
4. Update the distance matrix, D, by deleting the rows and columns corresponding to clusters

(r) and (s) and adding a row and column corresponding to the newly formed cluster. The distance between the new cluster, denoted (r,s) and old cluster(k) is defined in this way: d[(k), (r,s)] = min (d[(k),(r)], d[(k),(s)]).

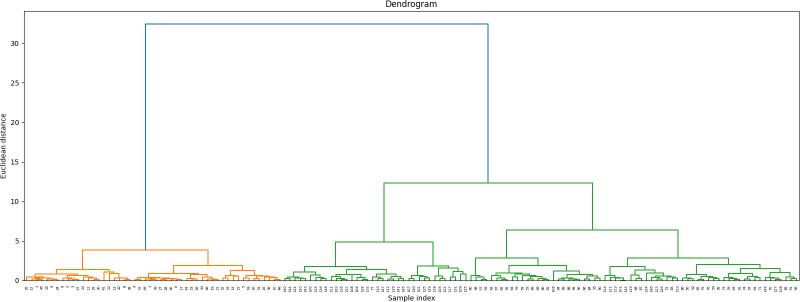
1. If all the data points are in one cluster, then stop, else repeat from step 2).

**Divisive Hierarchical clustering -** It is just the reverse of Agglomerative Hierarchical approach.

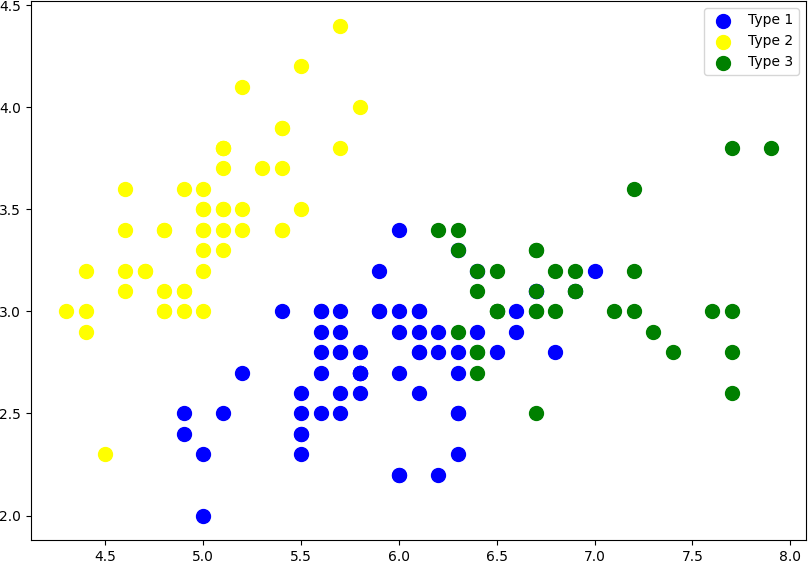
# **Program:**

| *import* matplotlib.pyplot *as* plt    *import* pandas *as* pd    *import* scipy.cluster.hierarchy *as* sc    *from* sklearn *import* datasets    *from* sklearn.cluster *import* AgglomerativeClustering        *# Import iris data*    iris = datasets.load\_iris()        iris\_data = pd.DataFrame(iris.data) iris\_data.columns = iris.feature\_names iris\_data['flower\_type'] = iris.target iris\_data.head()        *# Visualise the classes*    iris\_X = iris\_data.iloc[:, [0, 1, 2, 3]].values iris\_Y = iris\_data.iloc[:, 4].values | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| plt.figure(figsize=(10, 7)) plt.scatter(iris\_X[iris\_Y == | 0, | 0], | iris\_X[iris\_Y | == | 0, | 1], | s=100, | c='blue', |
| label=('Type 1')    plt.scatter(iris\_X[iris\_Y == | 1, | 0], | iris\_X[iris\_Y | == | 1, | 1], | s=100, |  |
| c=('yellow', label='Type 2') plt.scatter(iris\_X[iris\_Y == | 2, | 0], | iris\_X[iris\_Y | == | 2, | 1], | s=100, | c='green', |
| label=('Type 3') plt.legend() plt.show() |  |  |  |  |  |  |  |  |
| *# Plot dendrogram* plt.figure(figsize=(20, 7)) plt.title("Dendrograms") |  |  |  |  |  |  |  |  |
| *# Create dendrogram*    sc.dendrogram(sc.linkage(iris\_X, method='ward'))        plt.title('Dendrogram') plt.xlabel('Sample index') plt.ylabel('Euclidean distance')        cluster = AgglomerativeClustering(    n\_clusters=3, affinity='euclidean', linkage='ward') | | | | | | | | |
| cluster.fit(iris\_X) labels = cluster.labels\_ print(labels) |  |  |  |  |  |  |  |  |
| plt.figure(figsize=(10, 7)) plt.scatter(iris\_X[labels == | 0, | 0], | iris\_X[labels | == | 0, | 1], | s=100, | c='blue', |
| label='Type 1') plt.scatter(iris\_X[labels == | 1, | 0], | iris\_X[labels | == | 1, | 1], | s=100, |  |
| c=('yellow', label='Type 2') plt.scatter(iris\_X[labels == | 2, | 0], | iris\_X[labels | == | 2, | 1], | s=100, | c='green', |
| label=('Type 3') plt.legend()    plt.show() |  |  |  |  |  |  |  |  |

**Dendrogram:**

****

# **Scatterplot:**

****

**Conclusion:** Thus, we successfully implemented Hierarchical Clustering method.