	Marwadi University **Faculty of Technology** **Department of Information and Communication Technology**	
Subject: Machine Learning (01CT0519)	**Aim: To obtain the distinct clusters for unsupervised data using Hierarchical Clustering**	
Experiment No: 09	**Date:**	**Enrolment No:92000133018**

Aim: To obtain the distinct clusters for unsupervised data using Hierarchical Clustering

IDE: Google Colab

Theory:

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.

Sometimes the results of K-means clustering and hierarchical clustering may look similar, but they both differ depending on how they work. As there is no requirement to predetermine the number of clusters as we did in the K-Means algorithm.

The hierarchical clustering technique has two approaches:

1. 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.
2. 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.

How the Agglomerative Hierarchical clustering Work?

The working of the AHC algorithm can be explained using the below steps:

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

- **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.
- **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.
2. **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.
3. **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.
4. **Centroid Linkage:** It is the linkage method in which the distance between the centroid of the clusters is calculated.

Program (Code):

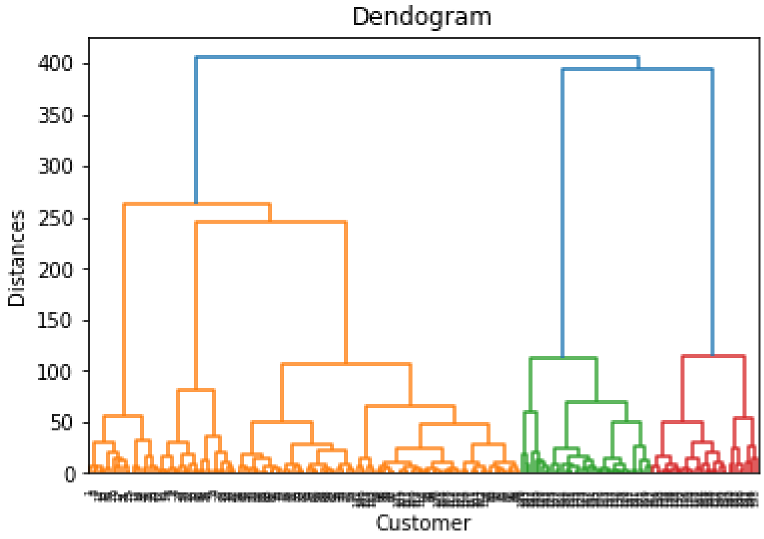
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
dataset = pd.read_csv("/content/KMeans.csv")

dataset
X = dataset.iloc[:, [1, 2]].values
X.shape
import scipy.cluster.hierarchy as sch
dendrogram = sch.dendrogram(sch.linkage(X, method='ward'))
plt.title('Dendogram')
plt.xlabel('Customer')
plt.ylabel('Distances')
plt.show();
from sklearn.cluster import AgglomerativeClustering
hc = AgglomerativeClustering(n_clusters = 5, affinity ='euclidean', linkage = 'ward' )
```

```python
y_hc=hc.fit_predict(X)
# Visualising the clusters
plt.scatter(X[y_hc == 0, 0], X[y_hc == 0, 1], s = 50, c = 'red', label = 'Cluster 1')
plt.scatter(X[y_hc == 1, 0], X[y_hc == 1, 1], s = 50, c = 'blue', label = 'Cluster 2')
plt.scatter(X[y_hc == 2, 0], X[y_hc == 2, 1], s = 50, c = 'green', label = 'Cluster 3')
plt.scatter(X[y_hc == 3, 0], X[y_hc == 3, 1], s = 50, c = 'cyan', label = 'Cluster 4')
plt.scatter(X[y_hc == 4, 0], X[y_hc == 4, 1], s = 50, c = 'magenta', label = 'Cluster 5')

plt.title('Clusters of customers')
plt.xlabel('Annual Income (k$)')
plt.ylabel('Spending Score (1-100)')
plt.legend()
# Using the elbow method to find the optimal number of clusters
from sklearn.cluster import KMeans
wcss = []
for i in range(1, 11):
    kmeans = KMeans(n_clusters = i, init = 'k-means++', random_state = 0)
    kmeans.fit(X)
    wcss.append(kmeans.inertia_)
plt.plot(range(1, 11), wcss)
plt.title('The Elbow Method')
plt.xlabel('Number of clusters')
plt.ylabel('WCSS')
plt.show()
# Fitting K-Means to the dataset
kmeans = KMeans(n_clusters = 5, init = 'k-means++', random_state = 0)
y_kmeans = kmeans.fit_predict(X)

# Visualising the clusters
plt.scatter(X[y_kmeans == 0, 0], X[y_kmeans == 0, 1], s = 50, c = 'magenta', label = 'Cluster 1')
plt.scatter(X[y_kmeans == 1, 0], X[y_kmeans == 1, 1], s = 50, c = 'red', label = 'Cluster 2')
plt.scatter(X[y_kmeans == 2, 0], X[y_kmeans == 2, 1], s = 50, c = 'blue', label = 'Cluster 3')
plt.scatter(X[y_kmeans == 3, 0], X[y_kmeans == 3, 1], s = 50, c = 'cyan', label = 'Cluster 4')
plt.scatter(X[y_kmeans == 4, 0], X[y_kmeans == 4, 1], s = 50, c = 'yellow', label = 'Cluster 5')
plt.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1], s = 200, c = 'green', label = 'Centroids')
plt.title('Clusters of customers')
plt.xlabel('Annual Income (k$)')
plt.ylabel('Spending Score (1-100)')
plt.legend()
plt.show()
```

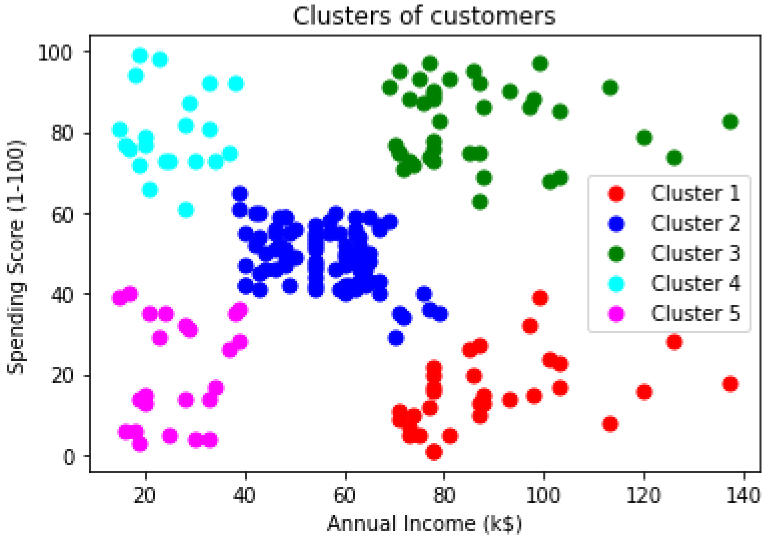
Results:

To be attached with

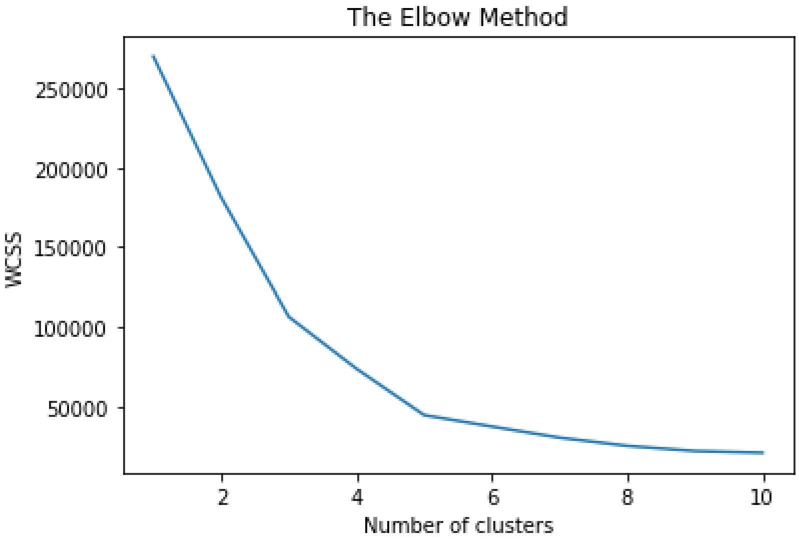
a. Dendogram

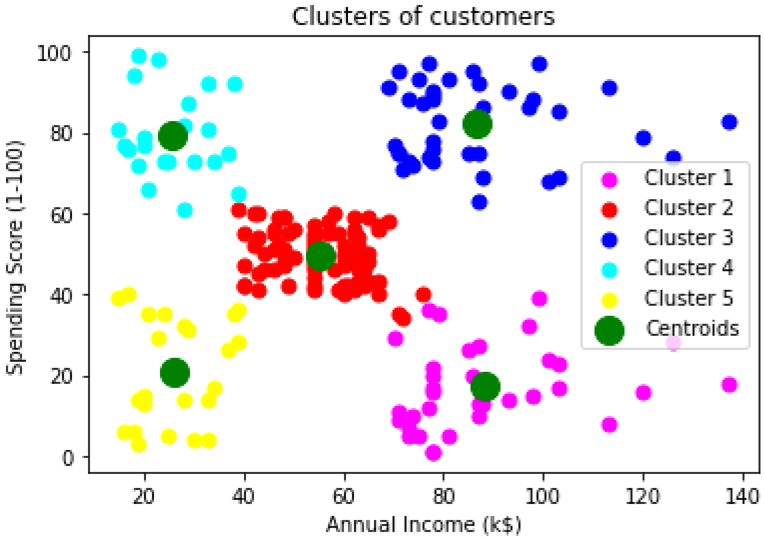


b. Cluster Distribution



c. Elbow method for Number of clusters VS WCSS





Observation and Result Analysis:

a. Nature of the dataset

b. During Training Process

c. After the training Process

d. Observation over the dendogram

Post Lab Exercise:

a. Difference between Agglomerative and Divisive Hierarchical clustering

b. List down the pros and cons of complete and single linkages methods in the Hierarchical Clustering Algorithm.

c. What is the group average method for calculating the similarity between two clusters for the Hierarchical Clustering Algorithm?

d. What is the Ward's method for calculating the similarity between two clusters in the Hierarchical Clustering Algorithm?

e. What is Space and Time Complexity of the Hierarchical Clustering Algorithm?

f. How to Find the Optimal Number of Clusters in Agglomerative Clustering Algorithm?

g. What are the advantages and disadvantages of the Hierarchical Clustering Algorithm?

Post Lab Activity:

Consider any dataset from **https://archive.ics.uci.edu/ml/datasets.php** and perform the multiple variable linear regression analysis over the dataset and obtain the best fit line. Make sure that the dataset is not matching with your classmates. You can also select the dataset from other ML repositories with prior permission from your concerned subject faculty.