DATA MINING AND WAREHOUSE

Experiment 5

Name: Ayush Jain
SAP ID:60004200132
DIV:B2

Aim : Implementation of Clustering Algorithm Using

1. **k-means**
2. **Hierarchical (single/complete/average)**

Theory:

- **What is Clustering?**

When you're trying to learn about something, say music, one approach might be to look for meaningful groups or collections. You might organize music by genre, while your friend might organize music by decade. How you choose to group items helps you to understand more about them as individual pieces of music. You might find that you have a deep affinity for punk rock and further break down the genre into different approaches or music from different locations. On the other hand, your friend might look at music from the 1980's and be able to understand how the music across genres at that time was influenced by the sociopolitical climate. In both cases, you and your friend have learned something interesting about music, even though you took different approaches.

.

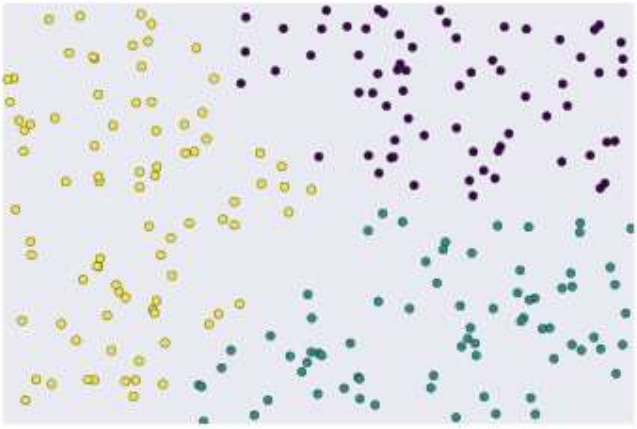


Figure 1: Unlabeled examples grouped into three clusters.

What are the Uses of Clustering?

Clustering has a myriad of uses in a variety of industries. Some common applications for clustering include the following:

- market segmentation
- social network analysis
- search result grouping
- medical imaging
- image segmentation
- anomaly detection

After clustering, each cluster is assigned a number called a **cluster ID**. Now, you can condense the entire feature set for an example into its cluster ID. Representing a complex example by a simple cluster ID makes clustering powerful. Extending the idea, clustering data can simplify large datasets.

What are clustering algorithms?

Clustering algorithms are used to group data points based on certain similarities. There's no criterion for good clustering. Clustering determines the grouping with unlabelled data. It mainly depends on the specific user and the scenario.

Typical cluster models include:

- Connectivity models – like hierarchical clustering, which builds models based on distance connectivity.
- Centroid models – like K-Means clustering, which represents each cluster with a single mean vector.
- Distribution models – here, clusters are modeled using statistical distributions.
- Density models – like DBSCAN and OPTICS, which define clustering as a connected dense region in data space.
- Group models – these models don't provide refined results. They only offer grouping information.
- Graph-based models – a subset of nodes in the graph such that an edge connects every two nodes in the subset can be considered as a prototypical form of cluster.
- Neural models – self-organizing maps are one of the most commonly known Unsupervised Neural networks (NN), and they're characterized as similar to one or more models above.

Note, there are different types of clustering:

- Hard clustering – the data point either entirely belongs to the cluster, or doesn't. For example, consider customer segmentation with four groups. Each customer can belong to either one of four groups.
- Soft clustering – a probability score is assigned to data points to be in those clusters.

What is K-mean?

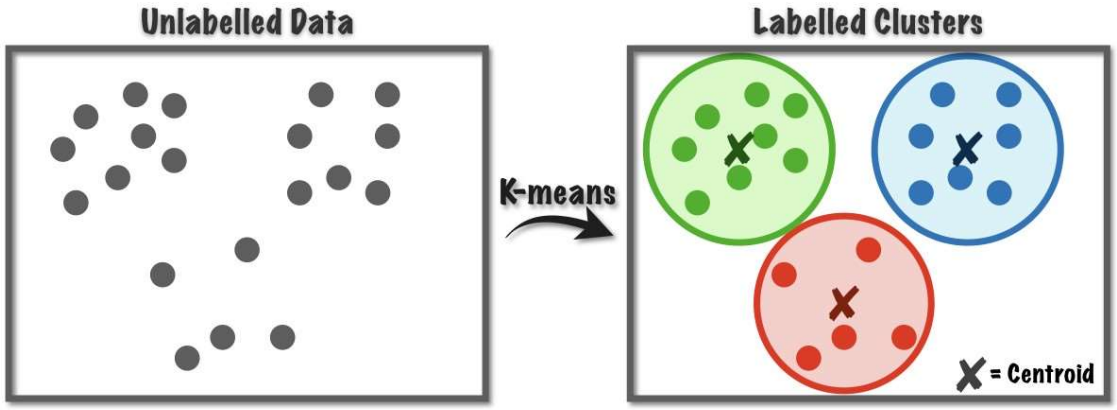
K-means clustering is one of the simplest and popular unsupervised machine learning algorithms.

Typically, unsupervised algorithms make inferences from datasets using only input vectors without referring to known, or labelled, outcomes.

AndreyBu, who has more than 5 years of machine learning experience and currently teaches people his skills, says that “the objective of K-means is simple: group similar data points together and discover underlying patterns. To achieve this objective, K-means looks for a fixed number (k) of clusters in a dataset.”A cluster refers to a collection of data points aggregated together because of certain similarities.

You’ll define a target number k, which refers to the number of centroids you need in the dataset. A centroid is the imaginary or real location representing the center of the cluster. Every data point is allocated to each of the clusters through reducing the in-cluster sum of squares.

The ‘means’ in the K-means refers to averaging of the data; that is, finding the centroid.



- **How the K-means algorithm works?**

To process the learning data, the K-means algorithm in data mining starts with a first group of randomly selected centroids, which are used as the beginning points for every cluster, and then performs iterative (repetitive) calculations to optimize the positions of the centroids

It halts creating and optimizing clusters when either:

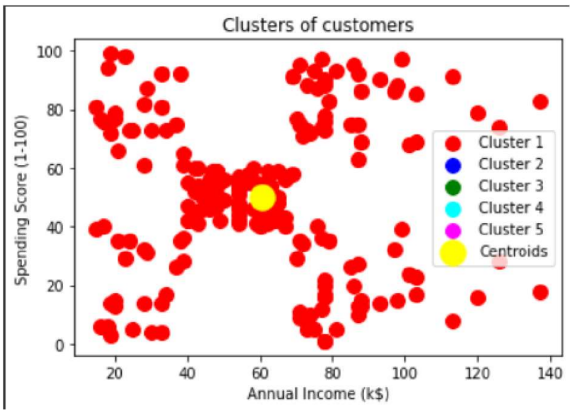
- The centroids have stabilized — there is no change in their values because the clustering has been successful.
- The defined number of iterations has been achieved.

Code:

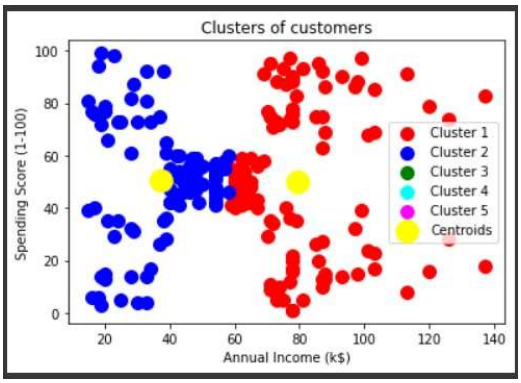
```
import numpy as np import
matplotlib.pyplot as plt import
pandas as pd from sklearn.cluster
import KMeans dataset =
pd.read_csv('mall.csv')
dataset.head()
X = dataset.iloc[:, [3, 4]].values kmeans = KMeans(n_clusters = 1,
init = 'k-means++', random_state = 42) y_kmeans =
kmeans.fit_predict(X)
```

```
plt.scatter(X[y_kmeans == 0, 0], X[y_kmeans == 0, 1], s = 100, c = 'red
', label = 'Cluster 1') plt.scatter(X[y_kmeans == 1, 0], X[y_kmeans ==
1, 1], s = 100, c = 'blu e', label = 'Cluster 2')
plt.scatter(X[y_kmeans == 2, 0], X[y_kmeans == 2, 1], s = 100, c = 'gre
en', label = 'Cluster 3') plt.scatter(X[y_kmeans == 3, 0], X[y_kmeans
== 3, 1], s = 100, c = 'cya n', label = 'Cluster 4')
plt.scatter(X[y_kmeans == 4, 0], X[y_kmeans == 4, 1], s = 100, c = 'mag
enta', label = 'Cluster 5') plt.scatter(kmeans.cluster_centers_[:, 0],
kmeans.cluster_centers_[:, 1
], s = 300, c = 'yellow', label = 'Centroids')
plt.title('Clusters of customers')
plt.xlabel('Annual Income (k$)')
plt.ylabel('Spending Score (1-100)')
plt.legend() plt.show()
```

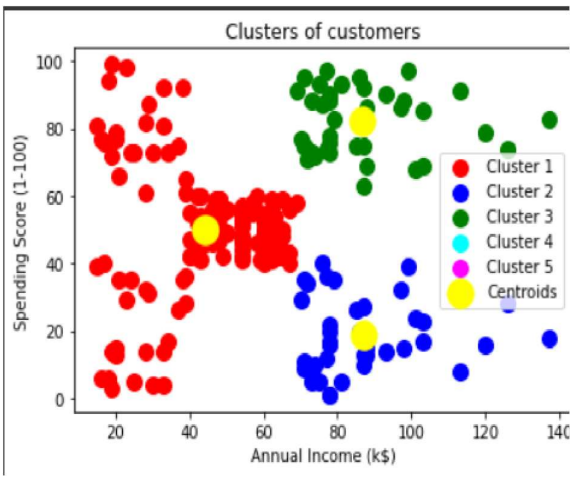
Output:



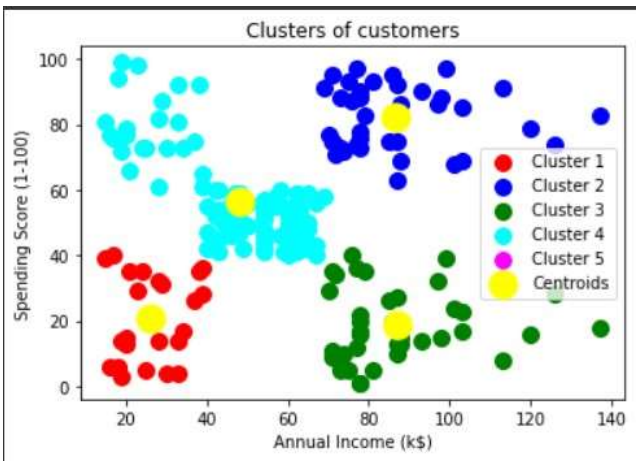
Cluster 1



Cluster 2



Cluster 3



Cluster 4

Part B:

- **Hierarchical Clustering (Single , complete , average)**
- **Dendogram (for all 3)**

- **Hierarchical clustering Technique:**

  Hierarchical clustering is one of the popular and easy to understand clustering technique. This clustering technique is divided into two types:

  1. Agglomerative
  2. Divisive

- **Agglomerative Hierarchical clustering Technique:** In this technique, initially each data point is considered as an individual cluster. At each iteration, the similar clusters merge with other clusters until one cluster or K clusters are formed.

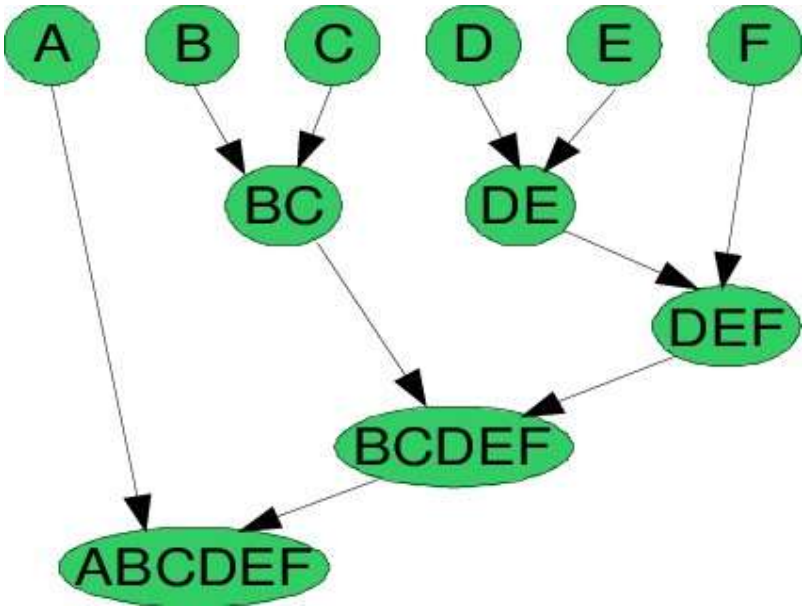
  The basic algorithm of Agglomerative is straight forward.

  - Compute the proximity matrix
  - Let each data point be a cluster
  - Repeat: Merge the two closest clusters and update the proximity matrix
  - Until only a single cluster remains

  Key operation is the computation of the proximity of two clusters

  To understand better let's see a pictorial representation of the Agglomerative Hierarchical clustering Technique. Lets say we have six data points {A,B,C,D,E,F}.

  - Step- 1: In the initial step, we calculate the proximity of individual points and consider all the six data points as individual clusters as shown in the image below.

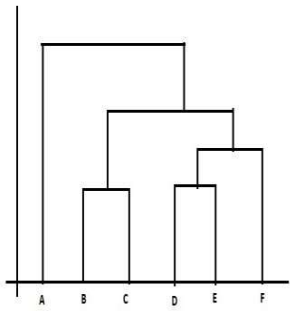


Agglomerative Hierarchical Clustering Technique

- Step- 2: In step two, similar clusters are merged together and formed as a single cluster. Let's consider B,C, and D,E are similar clusters that are merged in step two. Now, we're left with four clusters which are A, BC, DE, F.
- Step- 3: We again calculate the proximity of new clusters and merge the similar clusters to form new clusters A, BC, DEF.
- Step- 4: Calculate the proximity of the new clusters. The clusters DEF and BC are similar and merged together to form a new cluster. We're now left with two clusters A, BCDEF.
- Step- 5: Finally, all the clusters are merged together and form a single cluster.

The Hierarchical clustering Technique can be visualized using a **Dendrogram.**

A **Dendrogram** is a tree-like diagram that records the sequences of merges or splits.



Dendrogram representation

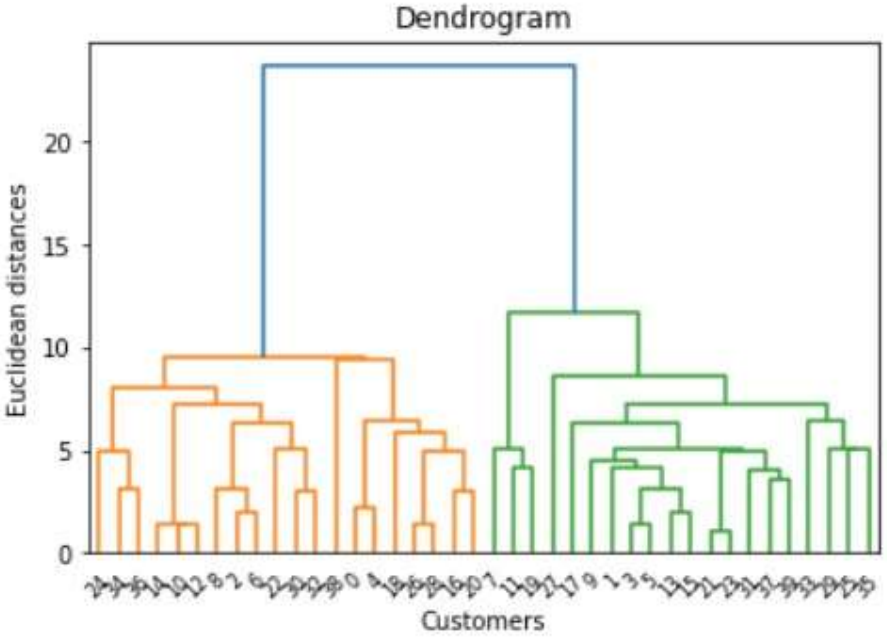
2. Divisive Hierarchical clustering Technique: Since the Divisive Hierarchical clustering Technique is not much used in the real world, I'll give a brief of the Divisive Hierarchical clustering Technique.

In simple words, we can say that the Divisive Hierarchical clustering is exactly the opposite of the **Agglomerative Hierarchical clustering.** In Divisive Hierarchical clustering, we consider all the data points as a single cluster and in each iteration, we separate the data points from the cluster which are not similar. Each data point which is separated is considered as an individual cluster. In the end, we'll be left with n clusters.

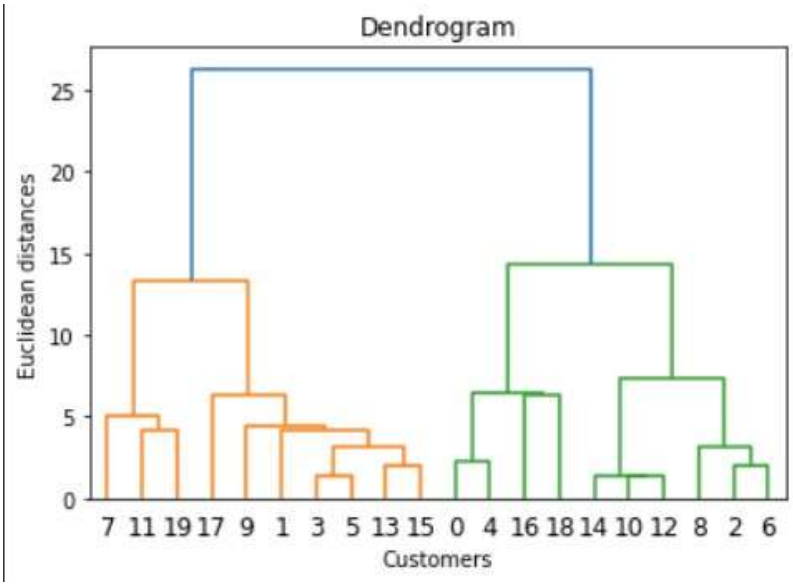
Code:

```
import numpy as np import
matplotlib.pyplot as plt import
pandas as pd

# Importing the dataset
dataset = pd.read_csv('mall - Copy.csv')
X = dataset.iloc[:, [3, 4]].values len(X)

# Using the dendrogram to find the optimal number of clusters
import scipy.cluster.hierarchy as sch dendrogram =
sch.dendrogram(sch.linkage(X, method = 'single'))
plt.title('Dendrogram') plt.xlabel('Customers')
plt.ylabel('Euclidean distances') plt.show()
plt.scatter(X[:,1], X[:,1], cmap="rainbow")
```

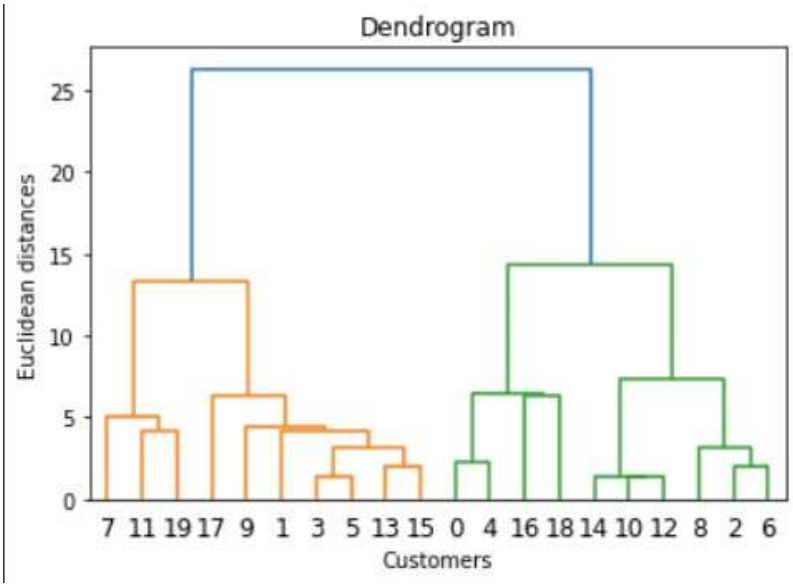
Output:



Single Hierarchical Clustering



Complete Hierarchical Clustering



Average Hierarchical Clustering

Part C:

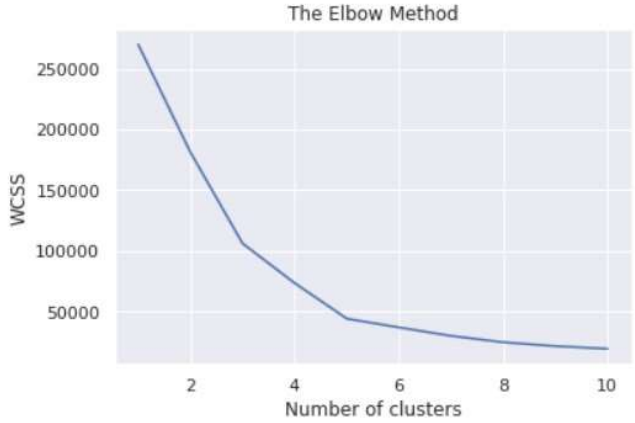
- **Plot Elbow Method**
- **Suggest optimal number of clusters**

A fundamental step for any unsupervised algorithm is to determine the optimal number of clusters into which the data may be clustered. The Elbow Method is one of the most popular methods to determine this optimal value of k.

Program:

```
# Using the elbow method to find the optimal number of clusters
X = dataset.iloc[:, [3, 4]].values
 from sklearn.cluster import
KMeans wcss = [] for i in range(1,
11):
    kmeans = KMeans(n_clusters = i, init =
'kmeans++', random_state = 42)
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()
```

Output :



Conclusion: We successfully implemented Clustering Algorithm Using 1. k-means 2. Hierarchical (single/complete/average)