Bangabandhu Sheikh Mujibur Rahman Digital University

Bangabandhu Sheikh Mujibur Rahman Digital University, Bangladesh.

Faculty of Cyber Physical Systems
Department of Internet of Things and Robotics Engineering
B.Sc. in Internet of Things and Robotics Engineering

Course Title: Data Science
Course Code: IOT 4313
Assignment On: Clustering

Submitted To:
Nurjahan Nipa
Lecturer,
Dept. of IRE, BDU.

Submitted By:

MD: AL MORSALINE
ID:1901020
3rd Year 2nd Semester
Session: 2019-20

Submission Date: 14th September 2023.

PART (A)

A detail explanation of your approaches and the results obtained for all the clustering algorithms required in Parts A

K-Means Clustering is a popular unsupervised machine learning algorithm used for clustering or grouping data points based on their features into K number of distinct groups. The primary goal of K-Means is to partition the data into clusters such that each data point belongs to the cluster with the nearest mean. The "mean" in this context refers to the center of a cluster.

Initialization:

Choose the number of clusters, K, that you want to identify. Initialize K centroids randomly in the feature space, which represents the center of each cluster.

Assignment Step:

Assign each data point to the nearest centroid, effectively grouping them based on proximity.

Update Step:

Recalculate the centroids by computing the mean of all data points assigned to each centroid.

Repeat:

Repeat the assignment and update steps until the centroids stabilize (i.e., they no longer change significantly) or a predefined number of iterations is reached.

Convergence:

The algorithm converges when the centroids no longer change or change very minimally between iterations.

Output:

The final clusters are formed, and each data point is assigned to one of the K clusters.

Objective Function (Sum of Squared Errors, SSE):

The algorithm aims to minimize the sum of squared distances (Euclidean distance) between data points and their respective centroids within a cluster. This sum is known as the SSE, and it quantifies the compactness of the clusters.

$$SSE=\sum_{i=1}^{K}\sum_{x\in C_i}||x-\mu_i||^2$$

Elbow Method:

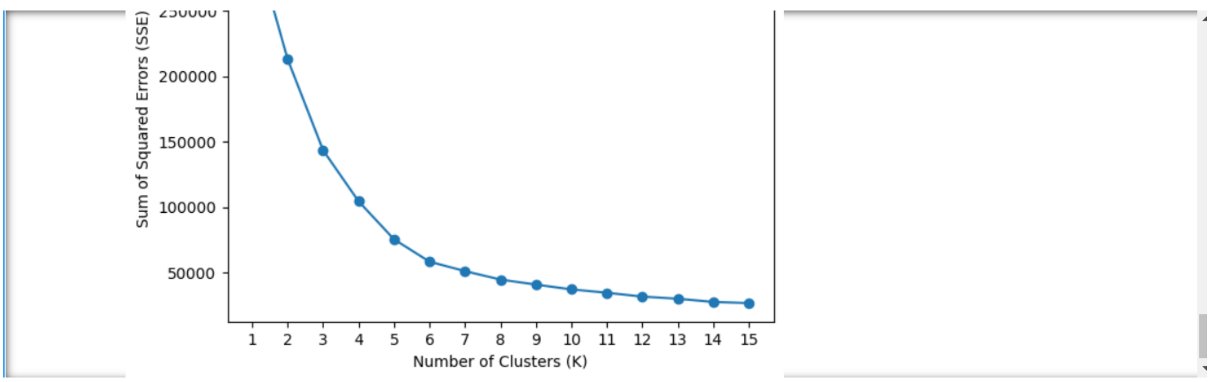
The "Elbow Method" is a technique to determine the optimal number of clusters (K). It involves plotting the SSE for a range of K values and identifying the "elbow" point, which signifies the point at which the SSE starts decreasing at a slower rate. The number of clusters corresponding to this "elbow" is often chosen as the optimal K because it balances model complexity and data fit.

K-Means clustering is a popular unsupervised machine learning technique used for clustering data points into distinct groups or clusters based on their feature similarities. The main objective of K-Means is to minimize the variance within each cluster and maximize the variance between different clusters.

Explanation of the result:

Visualization of the final product: I chose two features To see these five groupings visually, a scatter plot was created using the annual income and spending scores.

K-means is efficient and works well for well-defined, spherical clusters, but it requires specifying the number of clusters.



```
KMeans
KMeans(n_clusters=5, random_state=42)
```

- Import Necessary Libraries:
- Load the Dataset:
- Data Preprocessing:

  Convert the 'Gender' column to binary values (1 for Male, 0 for Female) to make it suitable for clustering:

- Extract relevant features for clustering (Age, Annual Income, Spending Score, and Gender) and store them in a feature matrix (X).

- Determine the Optimal Number of Clusters (K):

- Iterate through a range of K values (1 to 15) and calculate the Sum of Squared Errors (SSE) for each K using the Elbow Method. SSE is the sum of squared distances between each data point and its corresponding cluster center.
- Plot the SSE for each K to identify the "elbow" point, which indicates the optimal K where further clustering gains become marginal.

- Perform K-Means Clustering with the Optimal K:
  - Choose the optimal K based on the elbow method.
  - Fit the K-Means model with the optimal K and obtain cluster assignments for each data point.
  - Update the dataset with the cluster assignments for visualization.

- Visualize the Clusters:

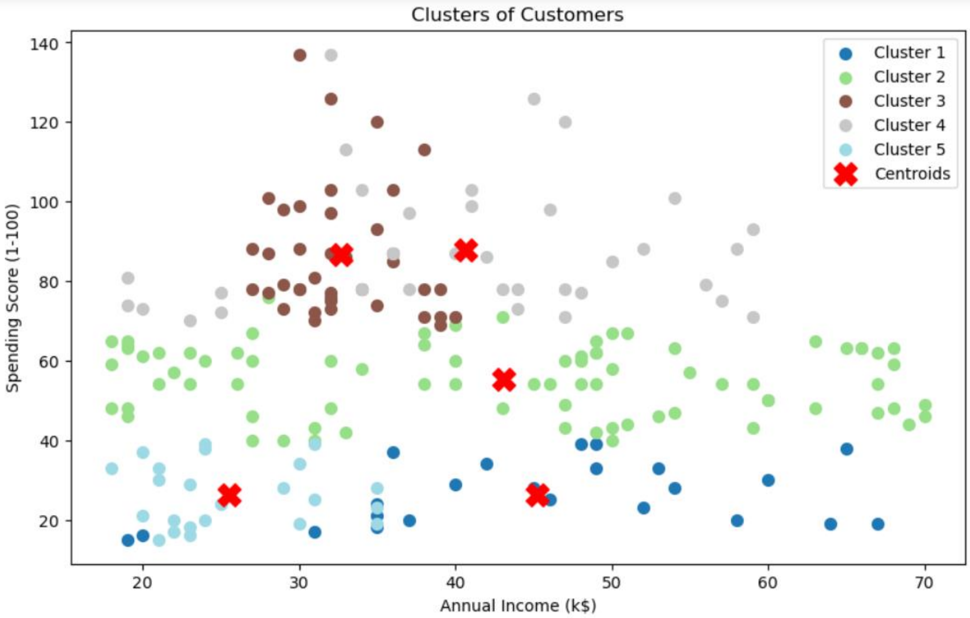
- Create a scatter plot to visualize the clusters based on 'Annual Income' and 'Spending Score', with different colors representing each cluster.
- Plot the cluster centroids (representing the center of each cluster) in red.

K-Means clustering is widely used in various domains, including customer segmentation, image compression, and anomaly detection, among others. It's important to note that the choice of the optimal number of clusters (K) is a crucial step, and the Elbow Method is a common approach to make an informed decision.

```
     CustomerID  Genre  Age  Annual Income (k$)  Spending Score (1-100)  \
0             1      1   19                  15                      39
1             2      1   21                  15                      81
2             3      0   20                  16                       6
3             4      0   23                  16                      77
4             5      0   31                  17                      40
..          ...    ...  ...                 ...                     ...
195         196      0   35                 120                      79
196         197      0   45                 126                      28
197         198      1   32                 126                      74
198         199      1   32                 137                      18
199         200      1   30                 137                      83

     Cluster
0          0
1          4
2          0
3          4
4          0
..       ...
195        2
196        3
197        2
198        3
199        2

[200 rows x 6 columns]
```

Explanation of the result:

We discovered that five clusters is the ideal amount for K-means clustering. Based on how similar the data points are, it divides them into K clusters.



PART (B)

Hierarchical Clustering is a clustering algorithm used in unsupervised machine learning. It aims to group similar data points into clusters based on their features. The result is a tree-like structure called a dendrogram, which shows the arrangement of the clusters.

Importing Libraries:

Import necessary libraries, including numpy, pandas, matplotlib.pyplot, AgglomerativeClustering from sklearn.cluster, and dendrogram, linkage from scipy.cluster.hierarchy.

Loading the Dataset:

Load the dataset from a CSV file (assuming it's in the same directory) using pd.read_csv. In this example, the dataset contains information about mall customers.

Data Preprocessing:

Convert categorical data (e.g., 'Genre') to a numerical format. In this case, 'Genre' is converted to binary values (1 for Male, 0 for Female).

Feature Preparation:

Select relevant features for clustering, such as 'Age', 'Annual Income', 'Spending Score', and the binary 'Genre' values.

Linkage Matrix Generation:

Use the linkage function from scipy.cluster.hierarchy to generate a linkage matrix. Linkage matrix calculates the distances between clusters.

Dendrogram Plotting:

Plot a dendrogram using the linkage matrix. The dendrogram visually represents the clustering process and helps determine the appropriate number of clusters.

Determining the Number of Clusters (optional):

Analyze the dendrogram to determine an appropriate number of clusters based on the structure of the dendrogram.

Hierarchical Clustering Model Fitting:

Create an instance of AgglomerativeClustering from sklearn.cluster and fit the model to the data, specifying the number of clusters, affinity (distance metric), and linkage method.

Cluster Assignment and Visualization:

- Assign each data point to a cluster using the fit_predict method.
- Visualize the clusters in a scatter plot using 'Annual Income' and 'Spending Score' as features. Each cluster is represented by a different color.

Displaying Cluster Labels in the Dataset:

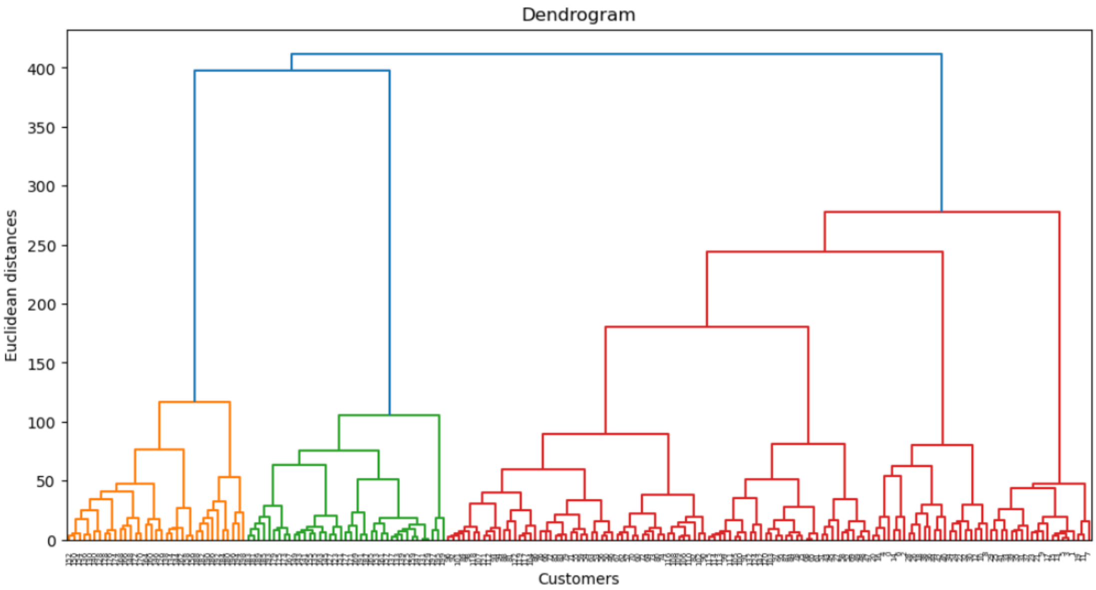
- Add the cluster labels as a new column in the dataset.
- Display the updated dataset with cluster labels using print(mall_data.head()).

Explanation of the result:

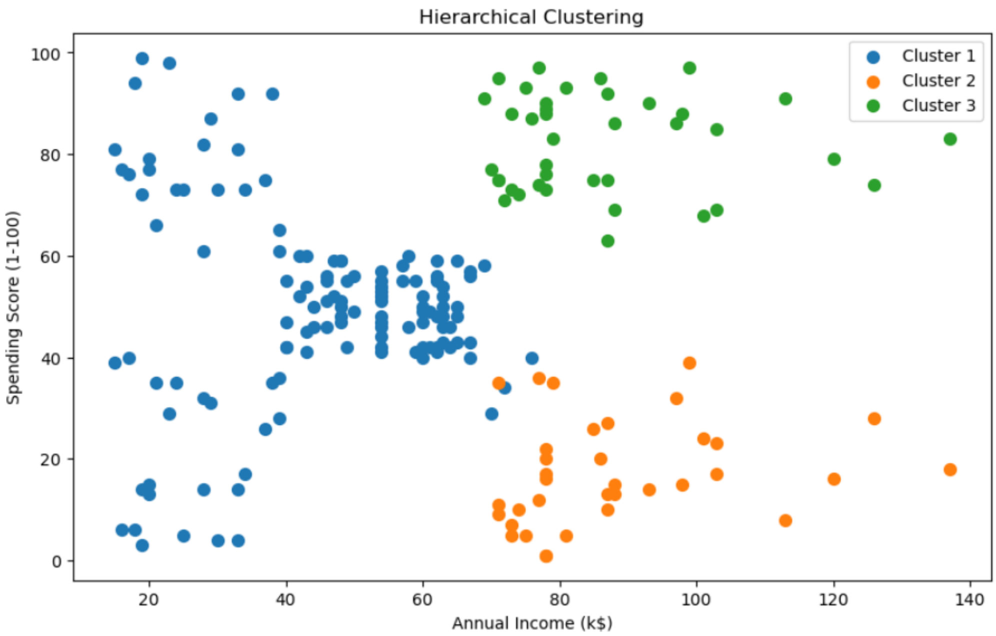
We discovered that five clusters is the ideal amount for hierarchical clustering. A tree-like structure of clusters is created via hierarchical clustering, and different levels of cutting can produce varied numbers of clusters.

Hierarchical Clustering is versatile and does not require pre-specifying the number of clusters, making it suitable for various datasets. We discovered

that there are 3 clusters in this dendrogram.




```
   CustomerID  Genre  Age  Annual Income (k$)  Spending Score (1-100)  Cluster
0           1      1   19                  15                      39        0
1           2      1   21                  15                      81        0
2           3      0   20                  16                       6        0
3           4      0   23                  16                      77        0
4           5      0   31                  17                      40        0
```



PART (C)

Hierarchical Clustering is a popular unsupervised machine learning technique used for grouping or clustering similar data points into clusters based on their similarities. Unlike K-means clustering, which is a centroid-based method, hierarchical clustering builds a tree-like hierarchy of clusters, which can be visually represented as a dendrogram. There are two main types of hierarchical clustering: Agglomerative (bottom-up) and Divisive (top-down).

- **Agglomerative Hierarchical Clustering:**

Agglomerative Hierarchical Clustering starts by treating each data point as a single cluster and then iteratively merges the closest clusters until only one cluster remains.

- **Initialization:**

Start by considering each data point as an individual cluster.

- **Calculate Distance Between Clusters:**

Measure the distance (similarity) between each pair of clusters. Common distance metrics include Euclidean distance, Manhattan distance, and cosine similarity.

- **Merge Closest Clusters:**

Merge the two closest clusters based on the chosen distance metric, creating a new larger cluster.

- **Update Distance Matrix:**

Recalculate the distance between the new cluster and all other clusters.

- **Repeat:**

Repeat steps 2-4 until only a single cluster remains.

- **Dendrogram:**

A dendrogram is a tree-like diagram that displays the steps of cluster merging. It helps to visualize the hierarchy and decide the optimal number of clusters.

Dendrogram Interpretation:

In a dendrogram:

- The vertical lines represent data points or clusters.
- The height of the vertical lines indicates the distance at which clusters were merged.

Choosing the Number of Clusters:

Identify the longest vertical line in the dendrogram that does not intersect any horizontal line. This is a good indicator of the optimal number of clusters.

Advantages of Hierarchical Clustering:

- Doesn't require specifying the number of clusters beforehand.
- Provides a hierarchy of clusters, allowing for more detailed analysis.
- Dendrograms offer a clear visualization of cluster relationships.

Disadvantages of Hierarchical Clustering:

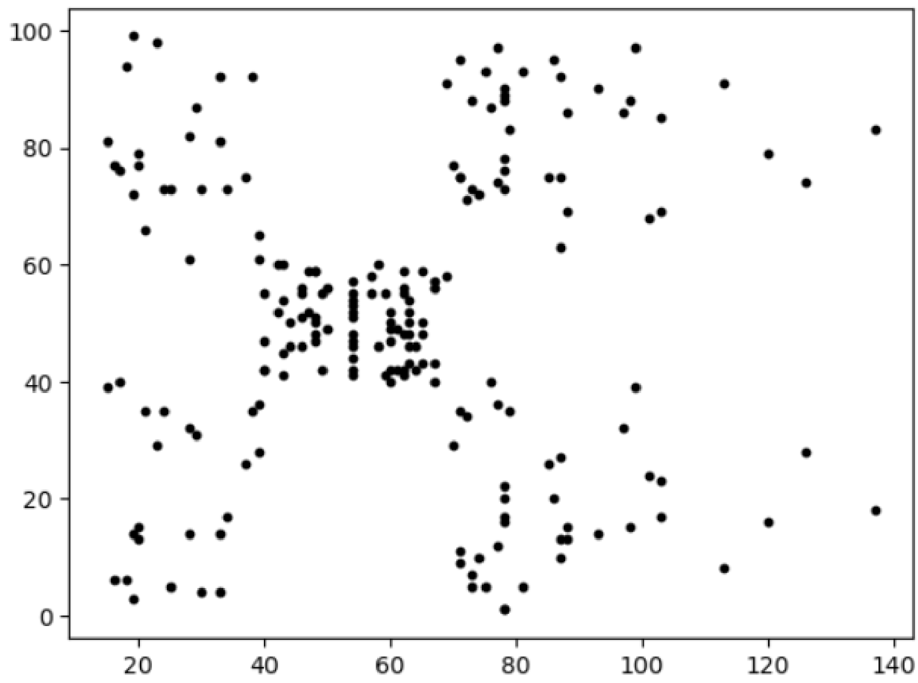
- Computationally intensive for large datasets.
- Difficult to use with very large datasets due to its time and memory complexity.

In summary, hierarchical clustering is a versatile technique that can be useful in various applications, such as customer segmentation, biological taxonomy, and more, to discover patterns and relationships within data in a hierarchical manner.

	CustomerID	Genre	Age	Annual Income (k$)	Spending Score (1-100)
0	1	Male	19	15	39
1	2	Male	21	15	81
2	3	Female	20	16	6
3	4	Female	23	16	77
4	5	Female	31	17	40

	CustomerID	Genre	Age	Annual Income (k$)	Spending Score (1-100)
195	196	Female	35	120	79
196	197	Female	45	126	28
197	198	Male	32	126	74
198	199	Male	32	137	18
199	200	Male	30	137	83

```
Out[14]: array([[ 15,  39],
               [ 15,  81],
               [ 16,   6],
               [ 16,  77],
               [ 17,  40],
               [ 17,  76],
               [ 18,   6],
               [ 18,  94],
               [ 19,   3],
               [ 19,  72],
               [ 19,  14],
               [ 19,  99],
               [ 20,  15],
               [ 20,  77],
               [ 20,  13],
               [ 20,  79],
               [ 21,  35],
               [ 21,  66],
               [ 23,  29],
```

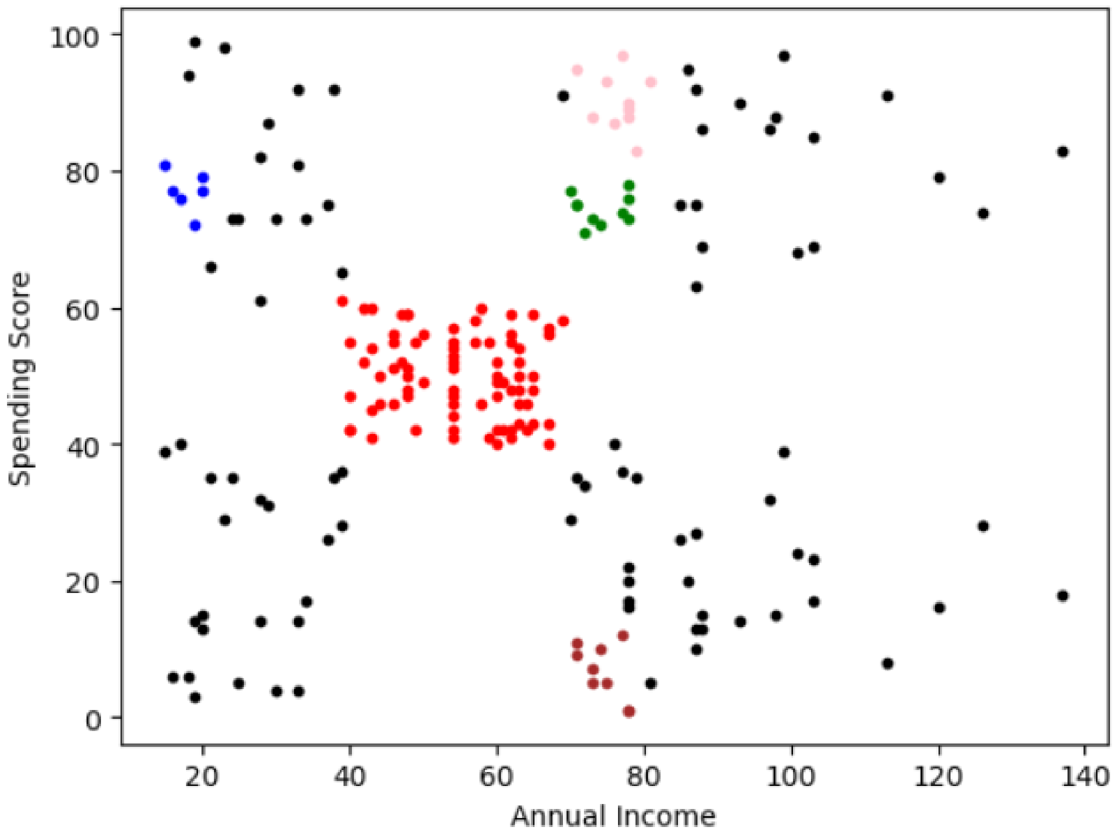


```
     CustomerID   Genre  Age  Annual Income (k$)  Spending Score (1-100)  \
0             1    Male   19                  15                      39
1             2    Male   21                  15                      81
2             3  Female   20                  16                       6
3             4  Female   23                  16                      77
4             5  Female   31                  17                      40
..          ...     ...  ...                 ...                     ...
195         196  Female   35                 120                      79
196         197  Female   45                 126                      28
197         198    Male   32                 126                      74
198         199    Male   32                 137                      18
199         200    Male   30                 137                      83

     Cluster  Cluster
0         -1       -1
1          0        0
2         -1       -1
3          0        0
4         -1       -1
..       ...      ...
195       -1       -1
196       -1       -1
197       -1       -1
198       -1       -1
199       -1       -1

[200 rows x 7 columns]
```

Explanation of the result:

Based on the number of data points, DBSCAN locates clusters. The scatter plot below shows the number of clusters and noise points for density-based clustering; the black point denotes the noise point, while the clusters are represented by other colors.

DBSCAN is effective in finding clusters of arbitrary shapes and handling noise, but it is sensitive to parameter settings and struggles with varying densities.



GitHub link: https://github.com/almorsaline/Clustering.git