Bangabandhu Sheikh Mujibur Rahman Digital University

Faculty of Cyber Physical Systems

Department of Internet of Things and Robotics Engineering

B.Sc. (Honors) in Internet of Things and Robotics Engineering

Course Code: IOT 4313

Course Title: Data Science

Assignment 02: Clustering

Submitted By

Mst.Arifa Azmary(1901006)

Session :2019-20

Date Of Submission:14/10/2023

Submitted To

NURJAHAN NIPA
Lecturer,
Department of IRE, BDU.

Department Of ICT,
BDU

K-means Clustering: In this part, you will be utilizing K-means clustering algorithm to identify the appropriate number of clusters. You may use any language and libraries to implement K-mean clustering algorithm. Your K-mean clustering algorithm should look for appropriate values of K at least in the range of 0 to 15 and show their corresponding sum-of-squared errors (SSE).

Customer Segmentation is a powerful method for understanding the diversity of customer groups and tailoring marketing strategies to meet their unique needs. In this section, we apply the K-means clustering algorithm to the dataset of a supermarket mall's customers. The objective is to identify an appropriate number of customer segments (clusters) based on age, gender, annual income, and spending score.

Dataset

The dataset provides information on customers, including:

- Customer ID: A unique identifier for each customer.
- Gender: Gender of the customer.
- Age: Age of the customer.
- Annual Income (k$): The customer's annual income.
- Spending Score (1-100): A score assigned by the mall based on customer behavior and spending nature.

K-means Clustering Algorithm

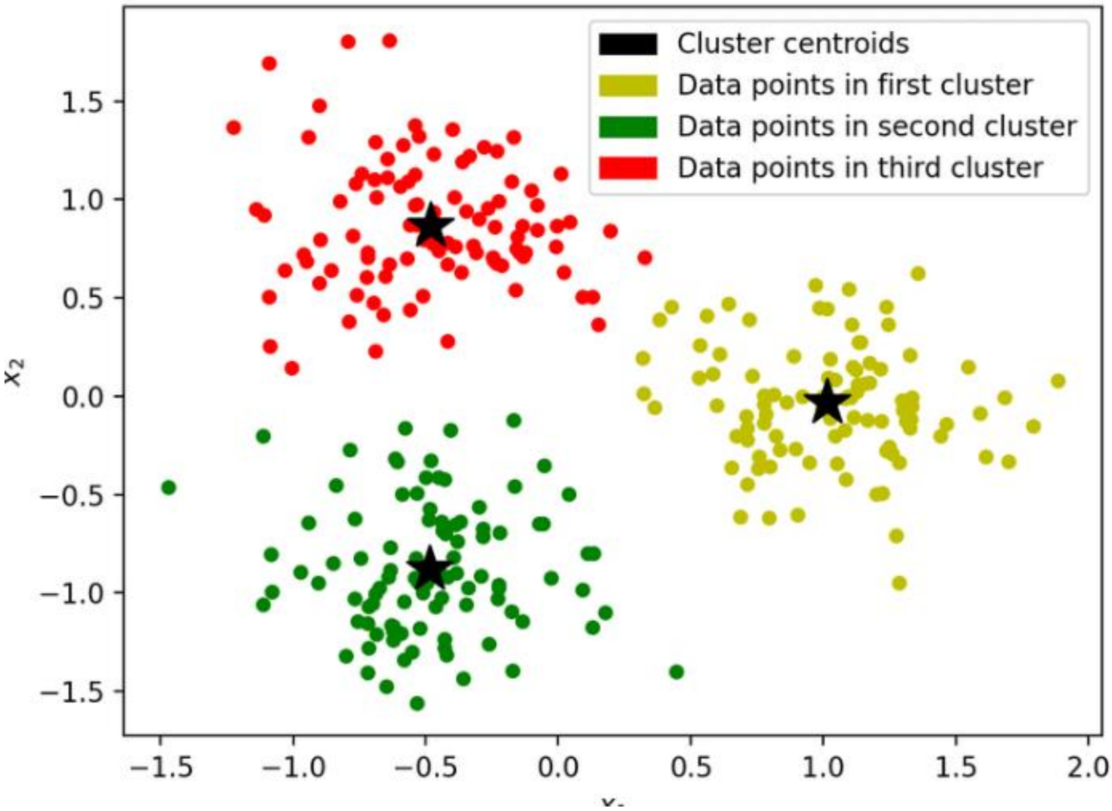
The K-means clustering algorithm is used for customer segmentation. The steps involved are as follows:

1. Specify the number of clusters, K, to be explored. In our analysis, we consider K values in the range from 0 to 15.
2. Initialize the cluster centroids. We shuffle the dataset and randomly select K data points as initial centroids without replacement.
3. Iterate until convergence. The assignment of data points to clusters is continuously updated, and the algorithm stops when there is no further change in the cluster assignments.

Advantages of Customer Segmentation

Customer segmentation provides various advantages:

- Appropriate product pricing can be determined for different customer groups.
- Customized marketing campaigns can be designed, catering to the unique needs and preferences of each segment.
- An optimal distribution strategy can be developed, considering the location and characteristics of each customer segment.
- Specific product features can be prioritized and tailored for deployment.
- New product development efforts can be focused on areas with the highest potential for success.

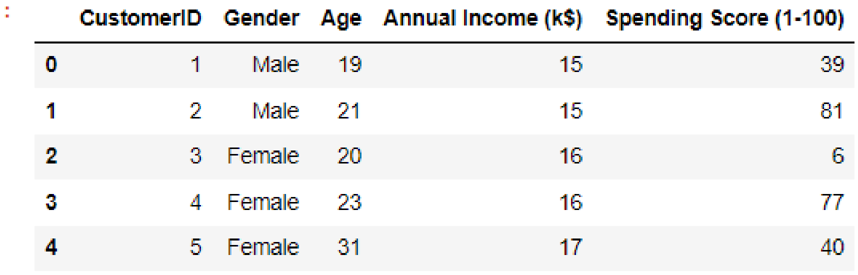


K Means Clustering where K=3

Environment and Tools Used

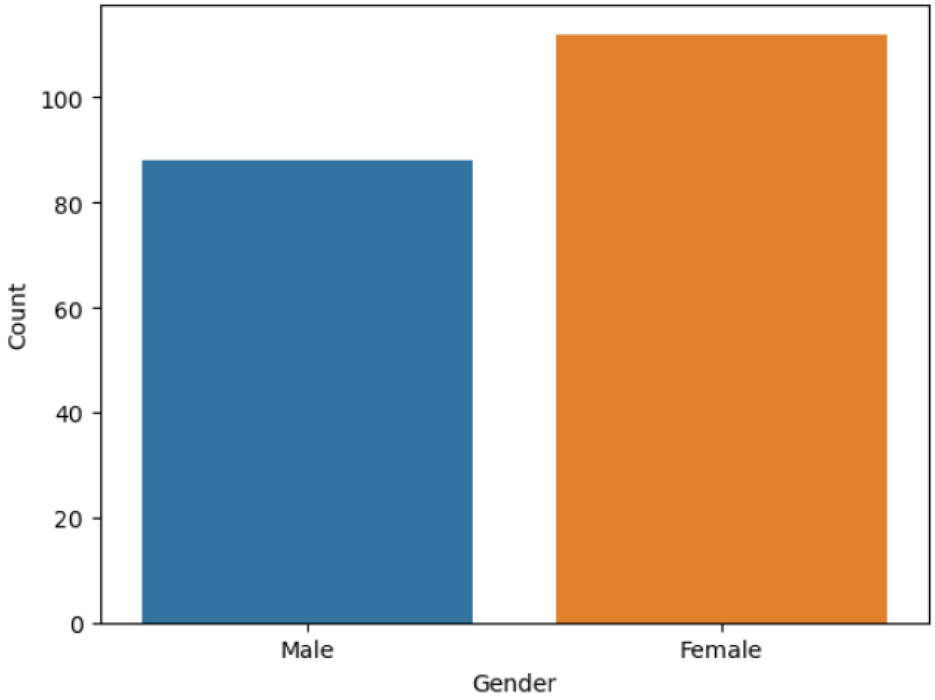
1. scikit-learn
2. seaborn
3. numpy
4. pandas
5. matplotlib

I began by importing the necessary libraries and dependencies. The dataset consists of columns for customer ID, gender, age, annual income, and spending score.

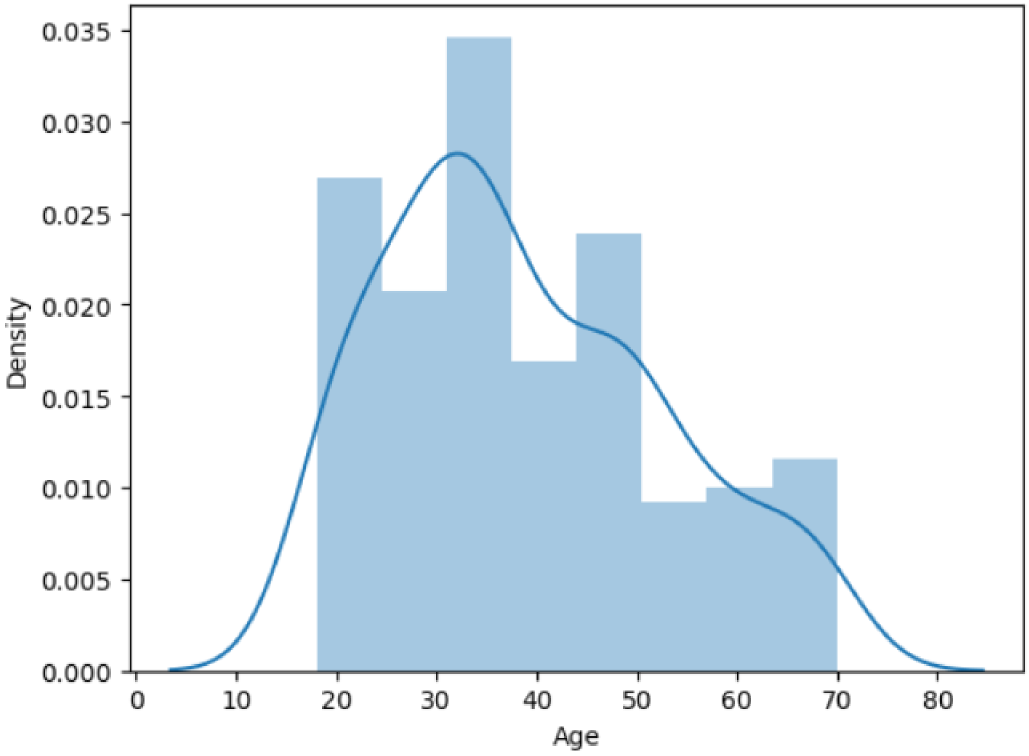


	CustomerID	Gender	Age	Annual Income (k$)	Spending Score (1-100)
0	1	Male	19	15	39
1	2	Male	21	15	81
2	3	Female	20	16	6
3	4	Female	23	16	77
4	5	Female	31	17	40

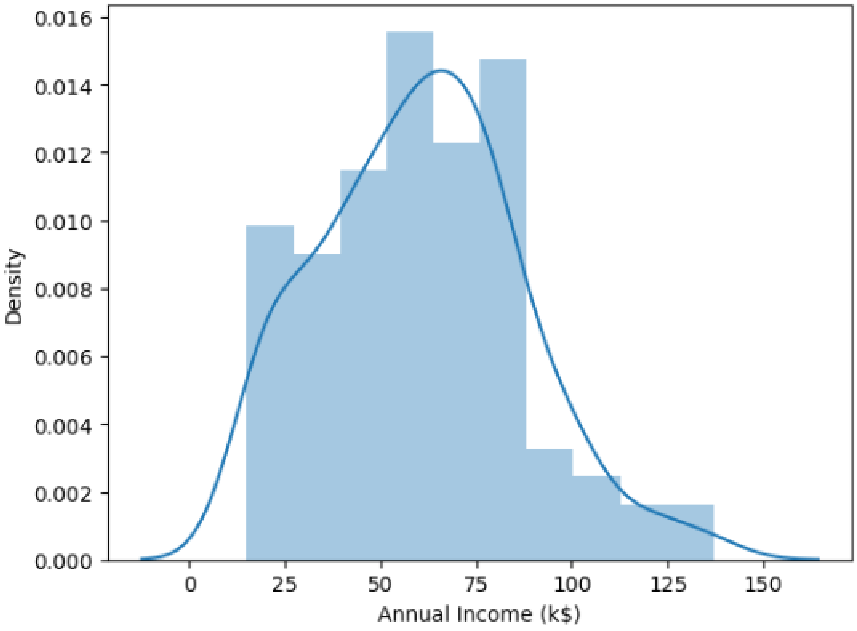
I created a bar plot to visualize the distribution of gender within the dataset. It's evident that the number of female customers significantly surpasses the number of male customers.



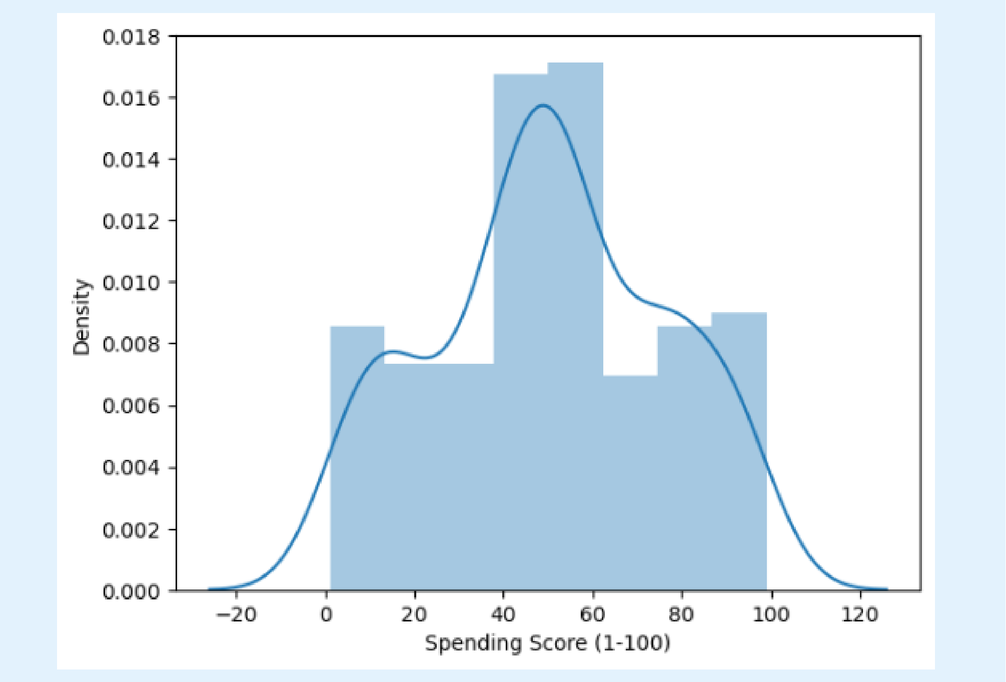
Next, I created a histogram to visualize the distribution of ages in the dataset using the 'Age' column.



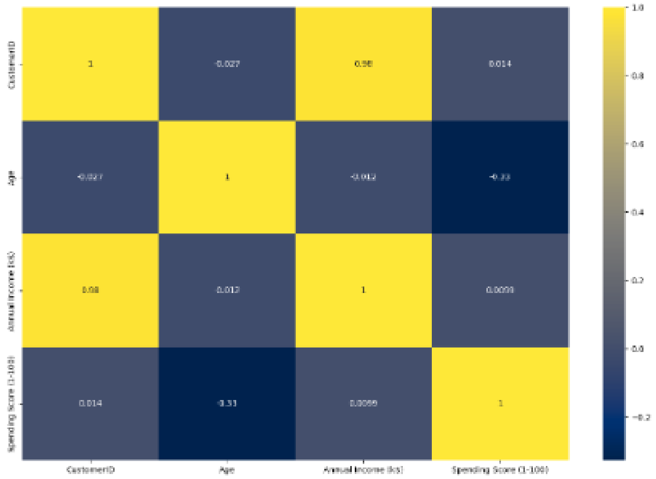
Next, I created a histogram to display the distribution of annual income using the 'Annual Income' column.

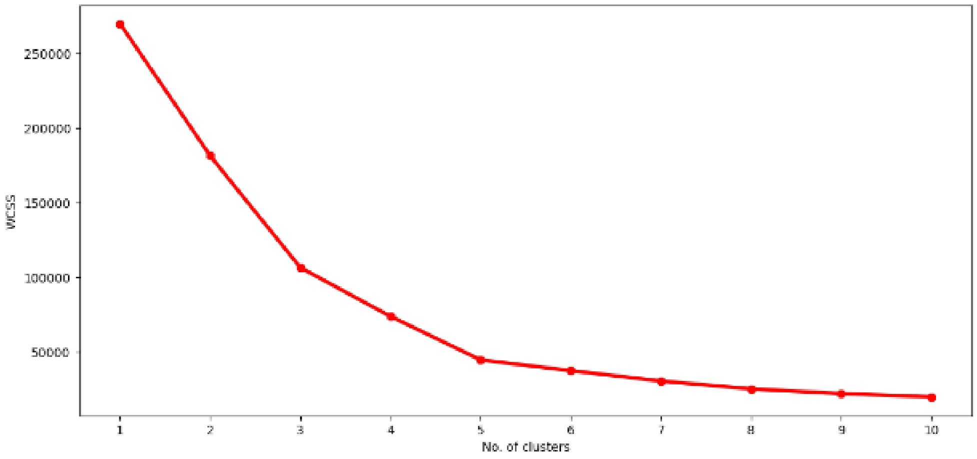


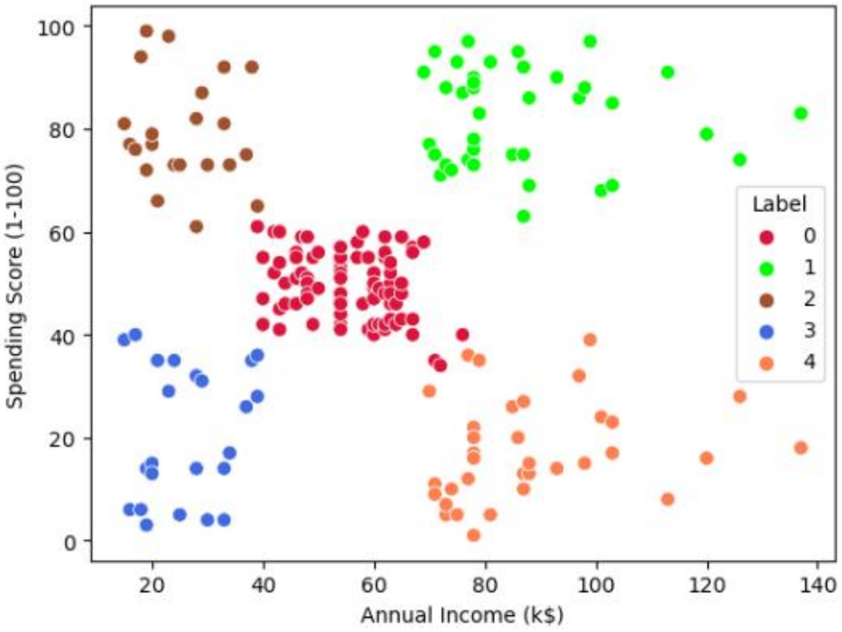
Next, I made create a distribution plot (histogram) for the 'Spending score' column.



After that,I generated a correlation matrix to visualize the relationships between numerical variables, and I annotated the matrix cells with correlation coefficients. Additionally, I created a scatterplot to further explore these correlations.







Next, I plotted the Within Cluster Sum Of Squares (WCSS) against the number of clusters (K Value) to determine the optimal number of clusters. WCSS measures the sum of distances of observations from their respective cluster centroids, as defined by the formula below.

$$WCSS = \sum_{i \in n} (X_i - Y_i)^2$$

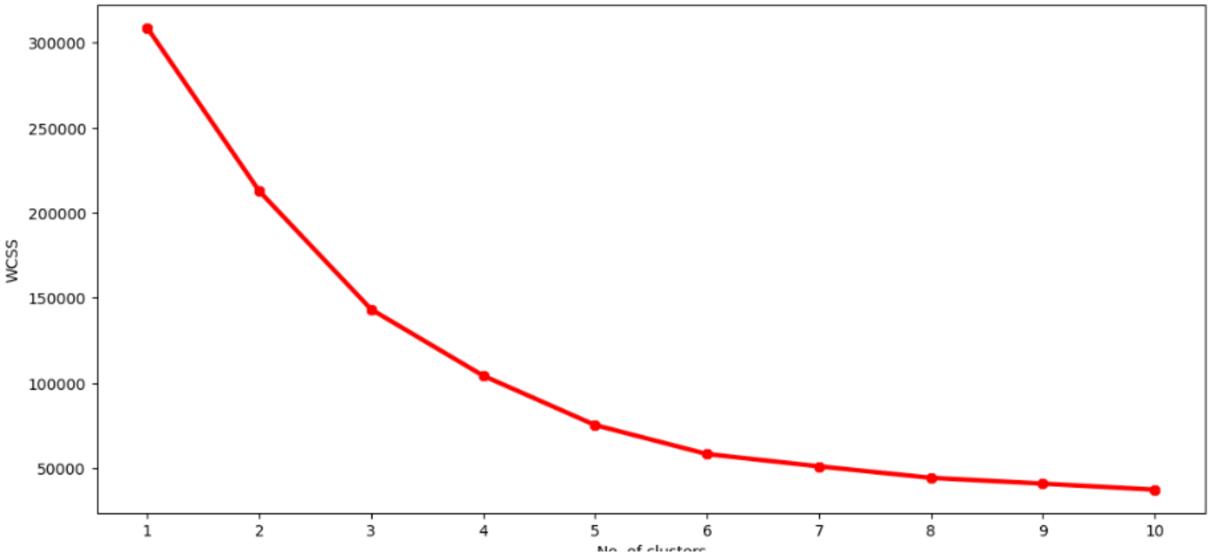
where Yi is centroid for observation Xi. The main goal is to maximize number of clusters and in limiting case each data point becomes its own cluster centroid.

The Elbow Method

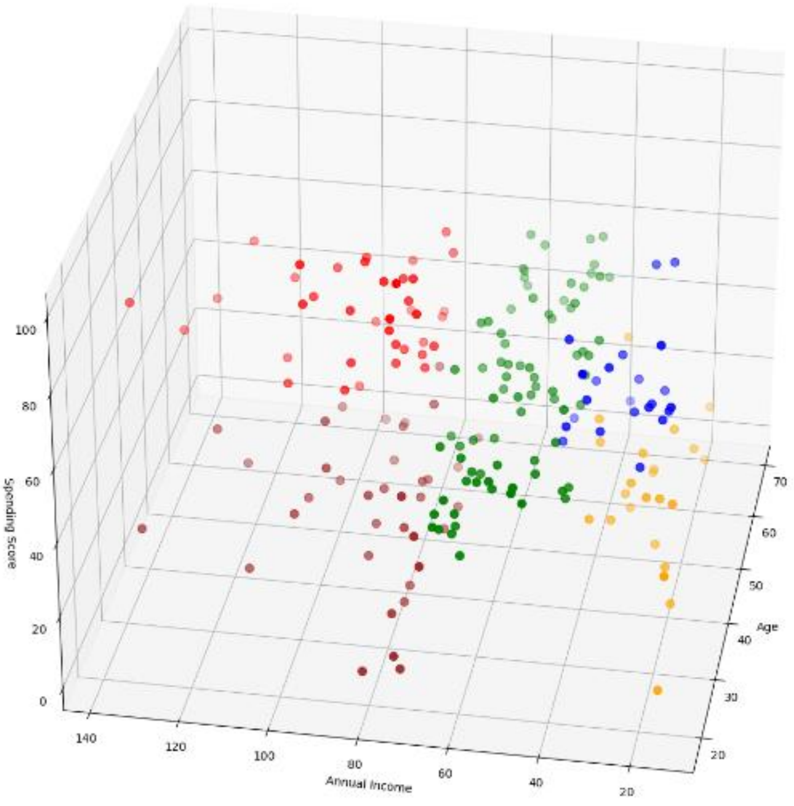
The Elbow Method involves calculating the Within Cluster Sum of Squared Errors (WSS) for various values of k and selecting the k value at which WSS begins to decrease noticeably. This "elbow" point is identified in the WSS-versus-k plot.

The steps can be summarized as follows:

1. Perform K-Means clustering with K values ranging from 1 to 10 clusters.
2. For each K value, compute the total within-cluster sum of squares (WCSS).
3. Create a plot that visualizes the relationship between WCSS and the number of clusters (K).
4. The location on the plot where a distinct bend or "knee" appears is typically indicative of the optimal number of clusters.



The optimal K value, determined through the elbow method, is found to be 5. In the final step, I created a 3D plot to visualize the relationship between customers' annual income and spending score. The data points are categorized into 5 distinct classes, each represented by a different color, as depicted in the 3D plot.



Conclusions

K-means clustering is a widely used and effective algorithm for clustering tasks. It is often the initial approach adopted by practitioners to gain insights into the structure of a dataset. The primary objective of K-means is to group data points into distinct, non-overlapping subgroups.

One of the significant applications of K-means clustering is customer segmentation. It provides a valuable way to gain a deeper understanding of customers, which, in turn, can be leveraged to enhance company revenue and tailor marketing strategies effectively.

PART (B)

Hierarchical Clustering: In this part, you will apply hierarchical clustering algorithm (agglomerative or divisive) to the provided mall dataset.

Hierarchical Clustering

Hierarchical clustering is an unsupervised machine learning algorithm employed for grouping unlabeled datasets into clusters. It is also referred to as hierarchical cluster analysis or HCA. In this algorithm, clusters are organized hierarchically, creating a tree-like structure known as a dendrogram.

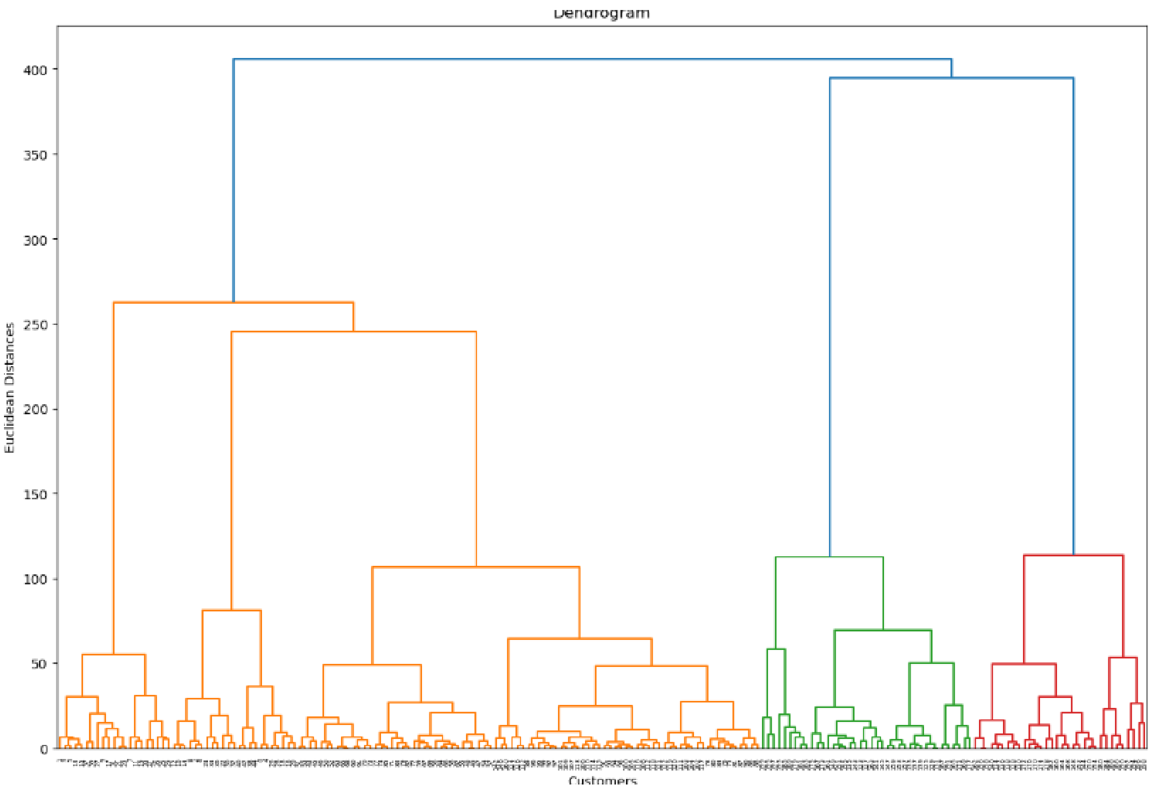
There are primarily two types of hierarchical clustering:

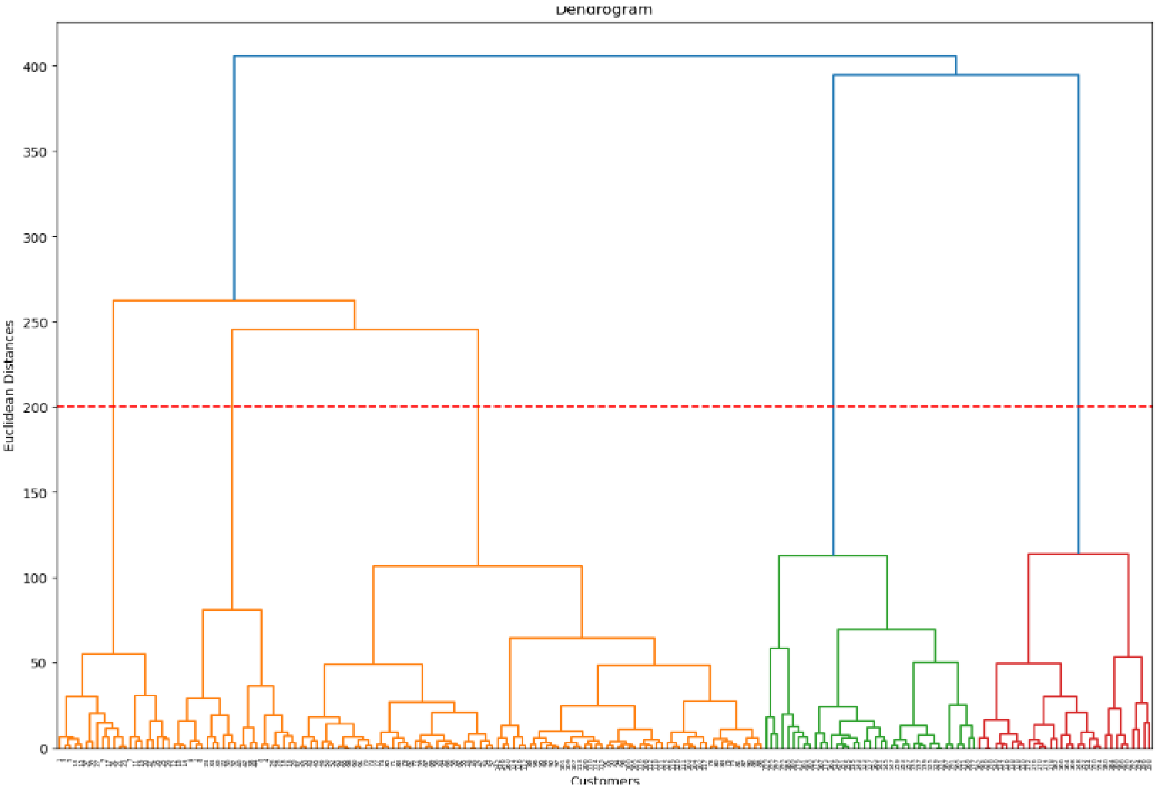
1. **Agglomerative Hierarchical Clustering:** In this method, data points start as individual clusters and are merged together step by step to form larger clusters, creating a hierarchy.
2. **Divisive Hierarchical Clustering:** This approach is the opposite of agglomerative clustering. It begins with all data points in a single cluster and then splits them into smaller clusters iteratively.

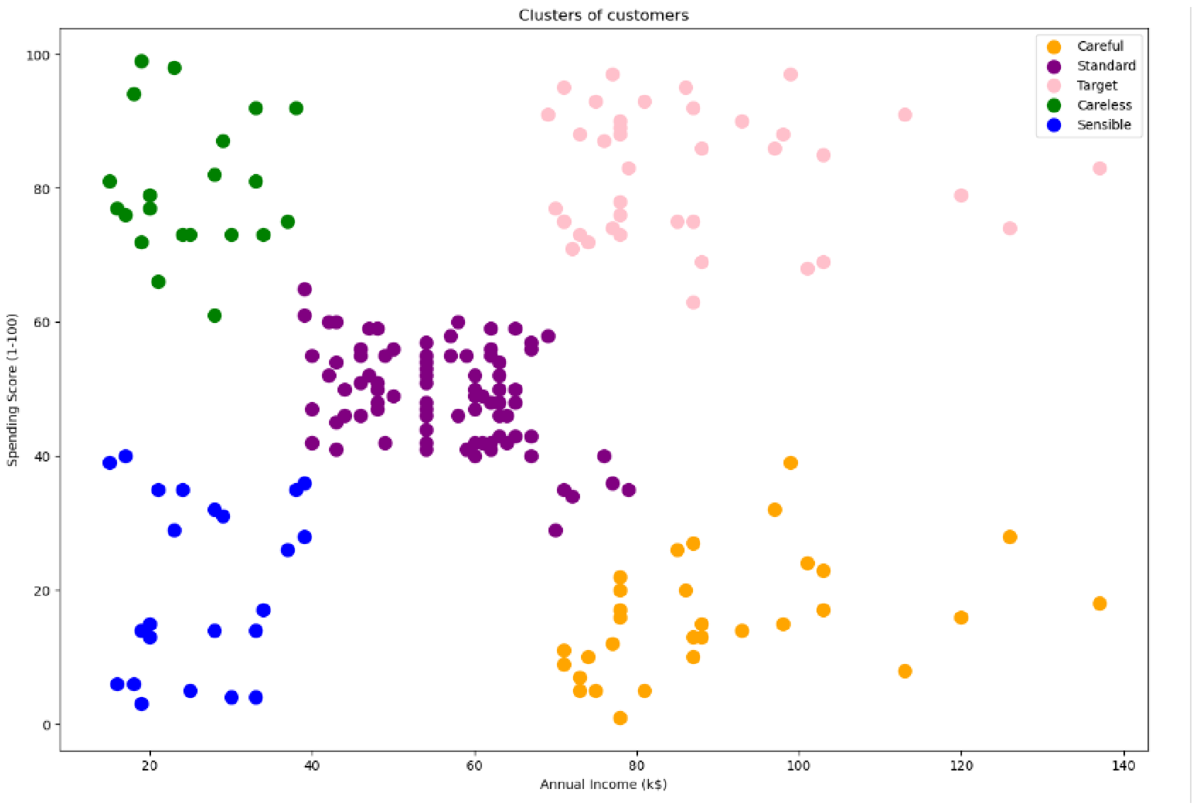
Hierarchical clustering is valuable for revealing the hierarchical relationships among data points and providing a visual representation of the clustering process through dendrograms. It is widely used in various fields for exploratory data analysis and pattern recognition.

Proximity Matrix

A proximity matrix is a square matrix with dimensions n x n, where n represents the number of observations in the dataset. Each element in the matrix signifies the distance between a pair of points. Euclidean distance formula is commonly utilized to calculate these distances. The proximity matrix provides a comprehensive view of the relationships and distances between all data points, serving as a foundational element in various clustering algorithms, including hierarchical clustering.


Dendrogram
Euclidean Distances
0
50
100
150
200
250
300
350
400
Customers


Dendrogram
Euclidean Distances
0
50
100
150
200
250
300
350
400
Customers



PART (C)

Density-based Clustering: In this part, you will apply density-based clustering algorithm to the provided dataset.

DBSCAN - Density-Based Spatial Clustering Application with Noise

DBSCAN stands for Density-Based Spatial Clustering Application with Noise. It's an unsupervised machine learning algorithm that creates clusters based on the density of data points or their proximity to each other. In DBSCAN, data points in denser regions are grouped together into clusters, while data points that are outside these dense regions are identified as noise or outliers. This algorithm is particularly useful for identifying clusters with irregular shapes and handling datasets with varying cluster densities effectively.

Algorithm - DBSCAN (Density-Based Spatial Clustering Application with Noise)

1. **Step 1 - Core Point Identification**:
    - For each data point in the dataset:
        - Calculate the distance to all other data points.
        - If the distance is less than or equal to a specified radius (epsilon), mark the data point as a "neighbor."
2. **Step 2 - Density-Based Clustering**:
    - For each core point:
        - Create a cluster and add the core point to it.
        - For each of the core point's neighbors:
            - If the neighbor is also a core point, add it to the same cluster.
            - If the neighbor is not a core point but has not been assigned to any cluster yet, include it in the cluster.
            - Continue this process recursively until no more data points can be added to the cluster.
3. **Step 3 - Handling Border Points**:
    - For each border point (a point that is within the epsilon radius of a core point but is not a core point itself):
        - Assign the border point to the cluster of its associated core point.
4. **Step 4 - Noise Detection**:
    - Any data point that is neither a core point nor a border point is considered noise and is not assigned to any cluster.

Key Parameters:

- Epsilon (ε): The maximum radius around a data point to define its neighborhood.
- Minimum Points (MinPts): The minimum number of data points required within the epsilon radius to consider a data point as a core point.

Advantages:

- Capable of identifying clusters with irregular shapes.
- Effective in handling varying cluster densities.
- Robust to noise and outliers.

Limitations:

- Sensitivity to the choice of epsilon and MinPts parameters.
- Not suitable for high-dimensional data due to the "curse of dimensionality."

DBSCAN is a versatile clustering algorithm widely used in various applications, especially when dealing with spatial and geographical data analysis. It offers a unique approach to density-based clustering by identifying clusters based on local density rather than a global criterion.

Result:

GitHub Link: https://github.com/AzmarySaRa/Data-Science-Clustering-Assignment-02