**DS USING PYTHON LAB**

**EXPERIMENT: 07**

**AIM:** Implementation of different clustering algorithms.

**PROBLEM STATEMENT:**

1. Clustering algorithm for unsupervised classification (K-means, density based

(DBSCAN), Hierarchical clustering)

1. Plot the cluster data and show mathematical steps.

**THEORY:**

Clustering is a method that can help machine learning engineers understand unlabeled data by creating meaningful groups or clusters. This often reveals patterns in data, which can be a useful first step in machine learning. Since the data you are working with is unlabeled, clustering is an unsupervised machine learning task.

Data is categorized into groups based on their similarity to each other through a metric known as the similarity measure, which is used to find out how similar the objects in the dataset are. To calculate this similarity measure, the feature data of the object in the dataset is used. A cluster ID is provided for each cluster, which is a powerful application of clustering. This allows large datasets to be simplified and also allows you to condense the entire feature set for an object into its cluster ID.

In organizations like Google, clustering is used for:

* Generalization: when objects in clusters have missing feature data, they can be inferred from other objects in their cluster.
* Data compression: feature data can be entirely replaced by the cluster ID. This saves storage and simplifies the feature space. This can help make ML model training simpler and faster.
* Preserve privacy: clustering groups of users and associating their data with cluster IDs prevent associating user data with specific users, ensuring user privacy.

**K-means**

K-Means Clustering is an Unsupervised Learning algorithm, which groups the unlabeled dataset into different clusters. Here K defines the number of predefined clusters that need to be created in the process, as if K=2, there will be two clusters, and for K=3, there will be three clusters, and so on.

It is an iterative algorithm that divides the unlabeled dataset into k different clusters in such a way that each dataset belongs to only one group that has similar properties.

It allows us to cluster the data into different groups and a convenient way to discover the categories of groups in the unlabeled dataset on its own without the need for any training.

It is a centroid-based algorithm, where each cluster is associated with a centroid. The main aim of this algorithm is to minimize the sum of distances between the data point and their corresponding clusters.

The k-means clustering algorithm mainly performs two tasks:

* Determines the best value for K center points or centroids by an iterative process.
* Assigns each data point to its closest k-center. Those data points which are near to the particular k-center, create a cluster.

The working of the K-Means algorithm is explained in the below steps:

Step-1: Select the number K to decide the number of clusters.

Step-2: Select random K points or centroids. (It can be different from the input dataset).

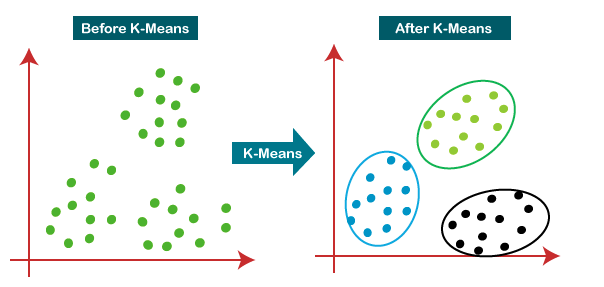
Step-3: Assign each data point to their closest centroid, which will form the predefined K clusters.

Step-4: Calculate the variance and place a new centroid of each cluster.

Step-5: Repeat the third steps, which means assign each datapoint to the new closest centroid of each cluster.

Step-6: If any reassignment occurs, then go to step-4 else go to FINISH.

Step-7: The model is ready.



**Density based (DBSCAN)**

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is a clustering algorithm that defines clusters as continuous regions of high density and works well if all the clusters are dense enough and well separated by low-density regions.

In the case of DBSCAN, instead of guessing the number of clusters, it will define two hyperparameters: epsilon and minPoints to arrive at clusters.

* Epsilon (ε): A distance measure that will be used to locate the points/to check the density in the neighborhood of any point.
* minPoints(n): The minimum number of points (a threshold) clustered together for a region to be considered dense.

Note:

In the case of higher dimensions, epsilon can be viewed as the radius of that hypersphere and minPoints as the minimum number of data points required inside that hypersphere.

After the DBSCAN clustering is complete, we end up with three types of data points as follows:

* Core: This is a point from which the two parameters above are fully defined, i.e., a point with at least Min Points within the Eps distance from itself.
* Border: This is any data point that is not a core point, but it has at least one Core point within Eps distance from itself.
* Noise: This is a point with less than Min Points within distance Eps from itself. Thus, it’s not a Core or a Border

The following are the DBSCAN clustering algorithm steps:

Step 1: Initially, the algorithms start by selecting a point (x) randomly from the data set and finding all the neighbor points within Eps from it. If the number of Eps-neighbors is greater than or equal to MinPoints, we consider x a core point. Then, with its Eps-neighbors, x forms the first cluster.

After creating the first cluster, we examine all its member points and find their respective Eps -neighbors. If a member has at least MinPoints Eps-neighbors, we expand the initial cluster by adding those Eps-neighbors to the cluster. This continues until there are no more points to add to this cluster.

Step 2: For any other core point not assigned to a cluster, create a new cluster.

Step 3: To the core point cluster, find and assign all points that are recursively connected to it.

Step 4: Iterate through all unattended points in the dataset and assign them to the nearest cluster at Eps distance from themselves. If a point does not fit any available clusters, locate it as a noise point.

**Hierarchical clustering**

Hierarchical clustering refers to an unsupervised learning procedure that determines successive clusters based on previously defined clusters. It works via grouping data into a tree of clusters. Hierarchical clustering starts by treating each data point as an individual cluster. The endpoint refers to a different set of clusters, where each cluster is different from the other cluster, and the objects within each cluster are the same as one another.

There are two types of hierarchical clustering

* Agglomerative Hierarchical Clustering
* Divisive Clustering

Agglomerative hierarchical clustering

Agglomerative clustering is one of the most common types of hierarchical clustering used to group similar objects in clusters. Agglomerative clustering is also known as AGNES (Agglomerative Nesting). In agglomerative clustering, each data point act as an individual cluster and at each step, data objects are grouped in a bottom-up method. Initially, each data object is in its cluster. At each iteration, the clusters are combined with different clusters until one cluster is formed.

The following are the Agglomerative hierarchical clustering steps:

* Determine the similarity between individuals and all other clusters. (Find proximity matrix).
* Consider each data point as an individual cluster.
* Combine similar clusters.
* Recalculate the proximity matrix for each cluster.
* Repeat step 3 and step 4 until you get a single cluster.

Divisive Hierarchical Clustering

Divisive hierarchical clustering is exactly the opposite of Agglomerative Hierarchical clustering. In Divisive Hierarchical clustering, all the data points are considered an individual cluster, and in every iteration, the data points that are not similar are separated from the cluster. The separated data points are treated as an individual cluster. Finally, we are left with N clusters.

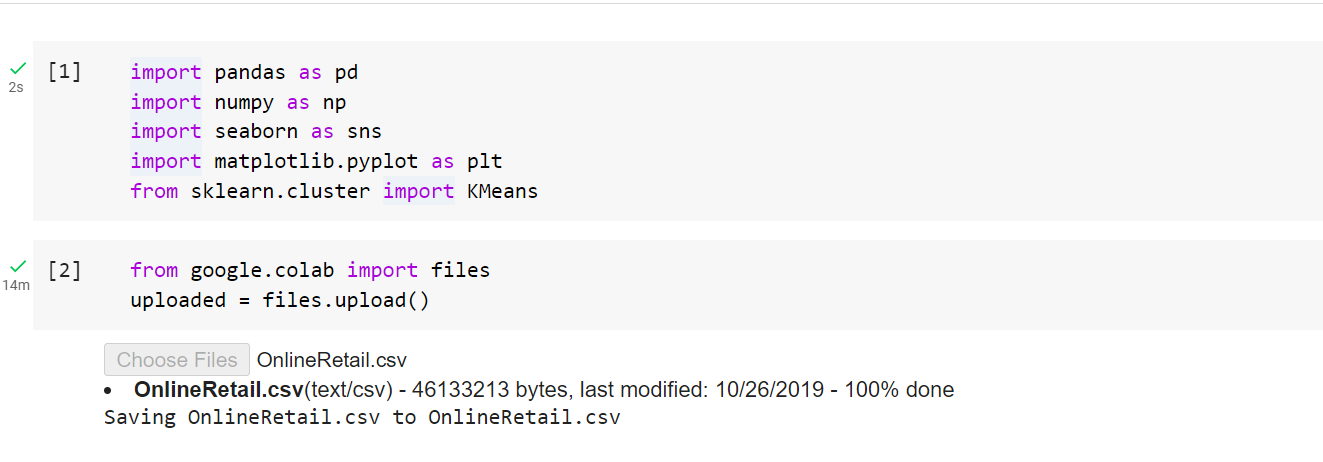
Algorithm :

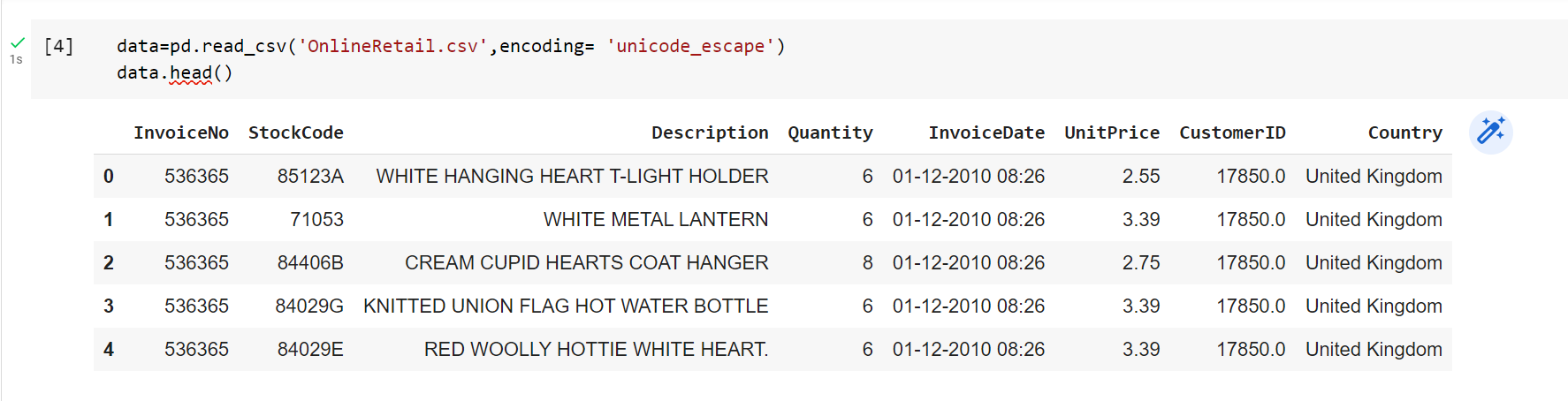
* given a dataset (d1, d2, d3, ....dN) of size N
* at the top we have all data in one cluster
* the cluster is split using a flat clustering method eg. K-Means etc
* repeat
* choose the best cluster among all the clusters to split
* split that cluster by the flat clustering algorithm
* until each data is in its own singleton cluster

**IMPLEMENTATION:**

**K MEANS**

1.Importing required libraries and dataset.

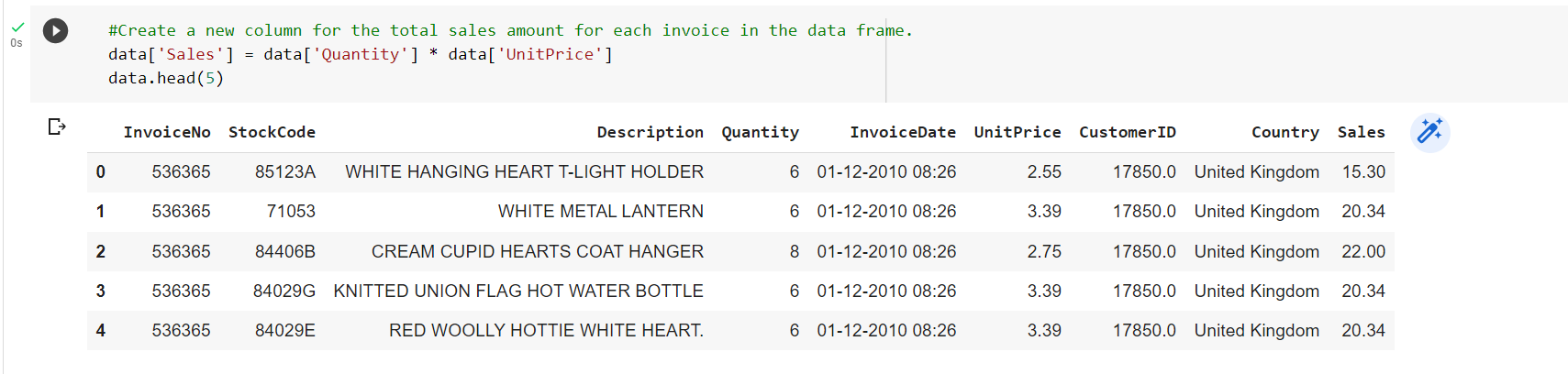




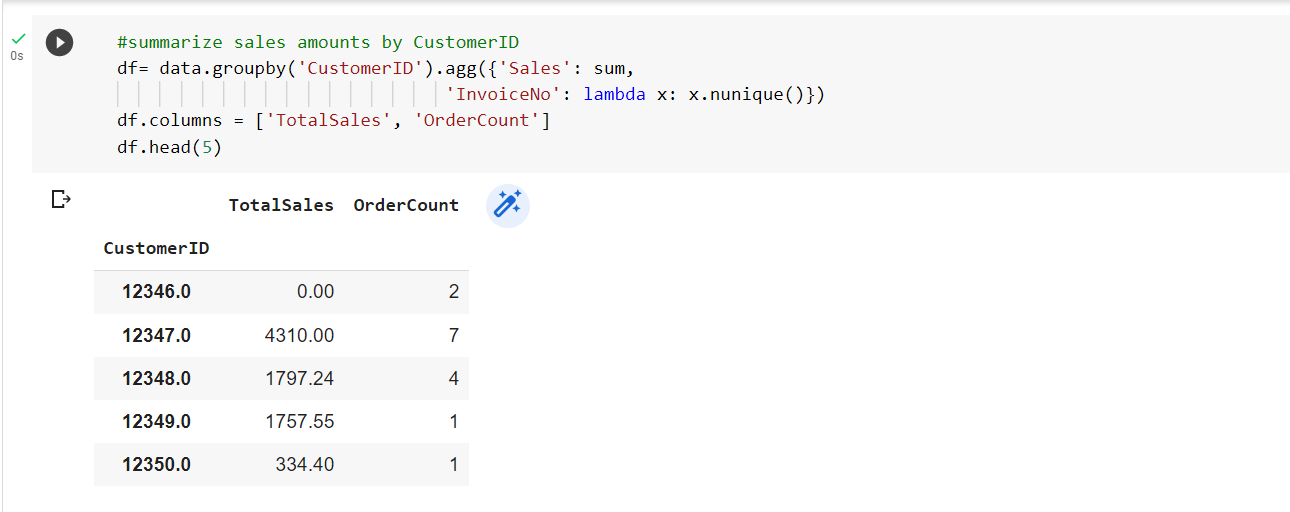
2.Data Preprocessing



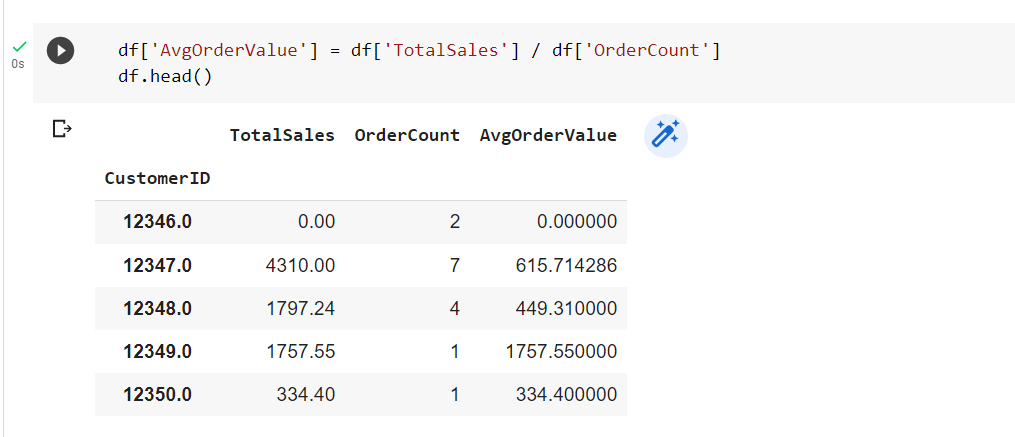
3.Create a new column for the total sales amount for each invoice in the data frame. This is calculated by multiplying quantity times the unit price for each row.



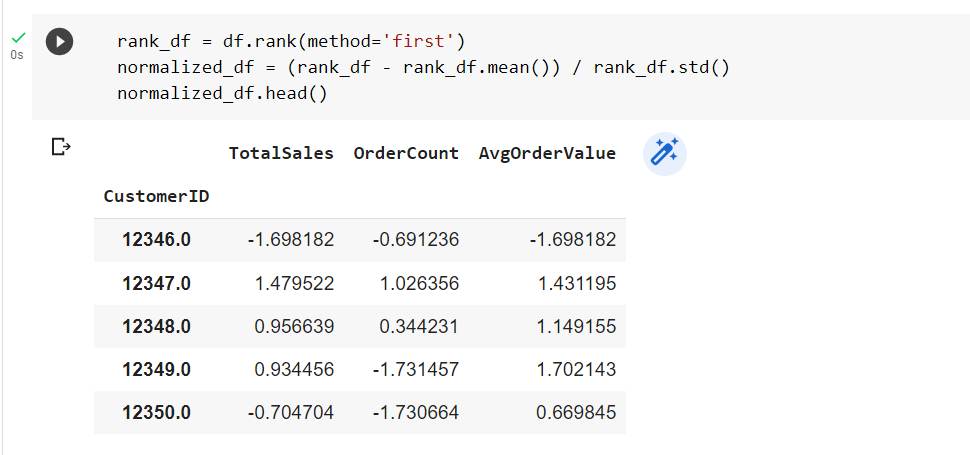
4. Transforming the data so that each record represents a single customer’s purchase history. We will use group by feature to summarize sales amounts by CustomerID. The lambda function will be used to sum the number of invoices by Customer ID.



5.Create a new column for the average order value for each customer in the data frame. This is calculated by dividing the total sales amount by the order count for each row.

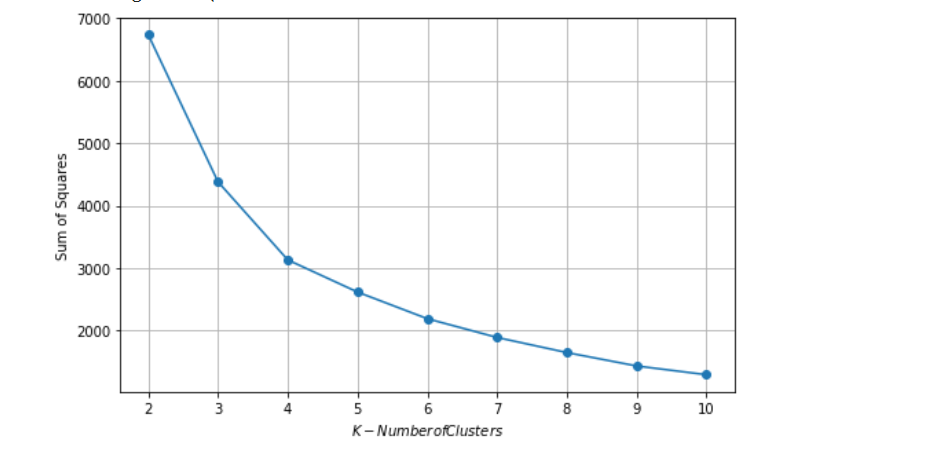


6.Normalization makes the features more consistent with each other, which allows the model to predict outputs more accurately. This is done for each feature or column in the data frame. The rank (method=first) function ranks the data in the order in which they appear in the data frame.

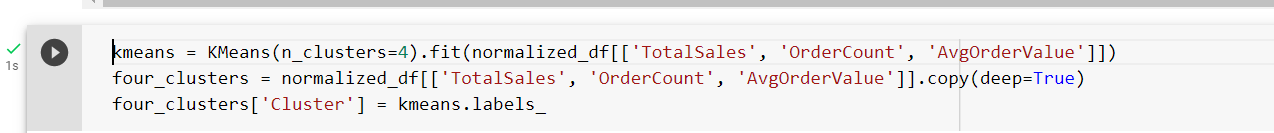


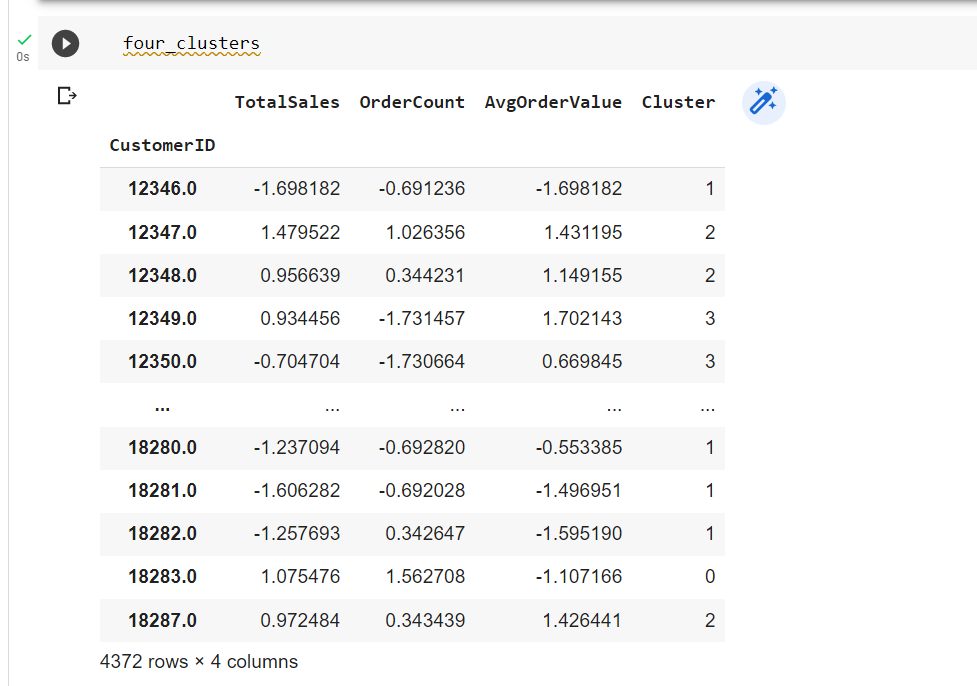
7.The elbow method of the K-means clustering algorithm is used to choose the optimum number of clusters by fitting the model with a range of values for K in the K-means algorithm. The Elbow method requires drawing a line plot between SSE (Sum of Squared errors) vs a number of clusters and finding the point representing the “elbow point”.



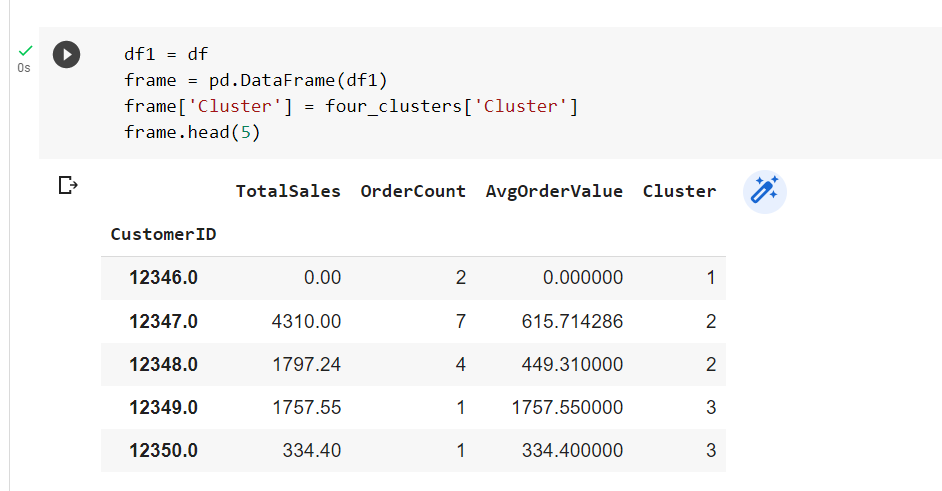


8.Training the k-means clustering model on the normalized data frame using the optimal number of clusters found which is 4.

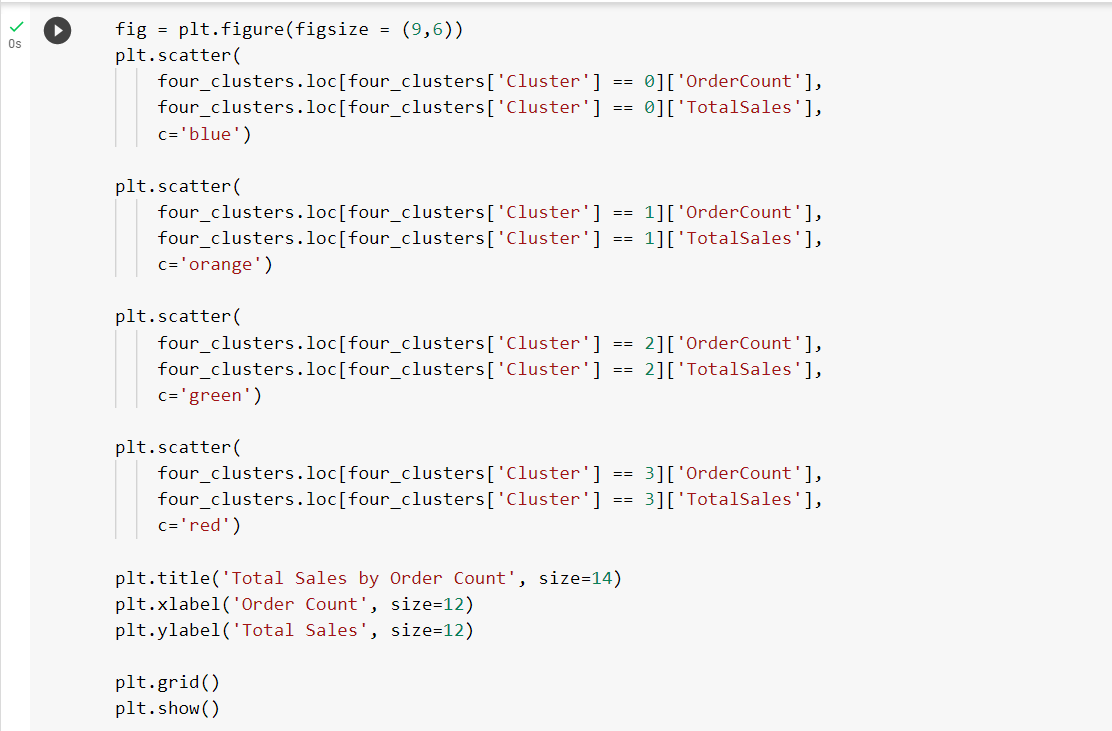


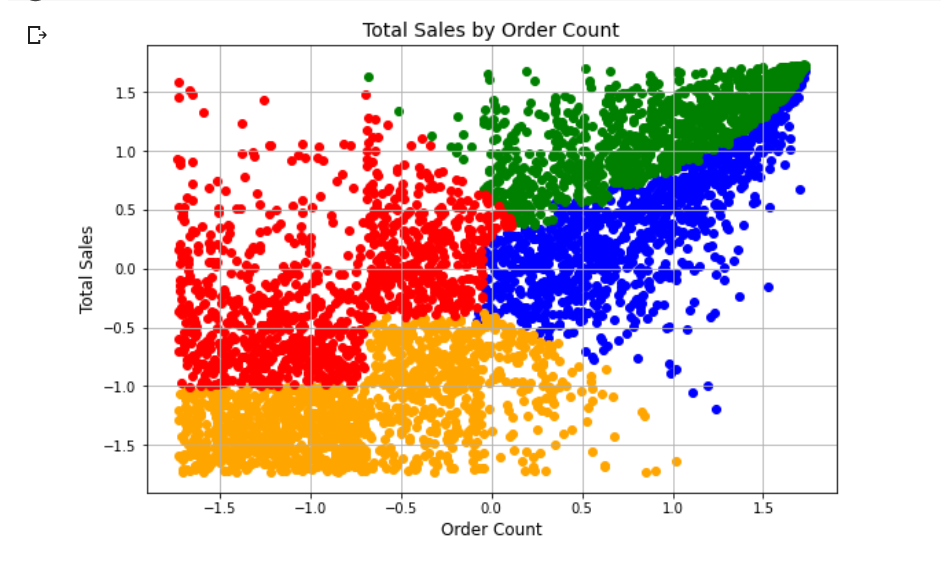


9.creating a new data frame from the previous data frame where each record represented a single customer’s purchase history prior to normalization.



10.Visualizing the clusters for total sales by order count.



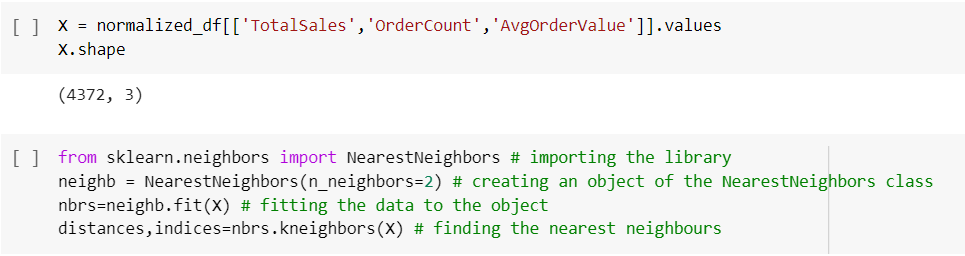


The customers in yellow have low total sales and low order count, meaning they are low-value customers. The customers in green have high total sales and high order counts, indicating they are high-value customers.

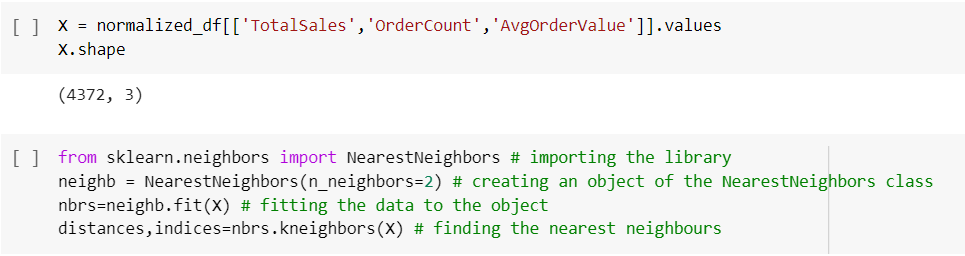
**Density based (DBSCAN):**

Followed the steps 1-6 same

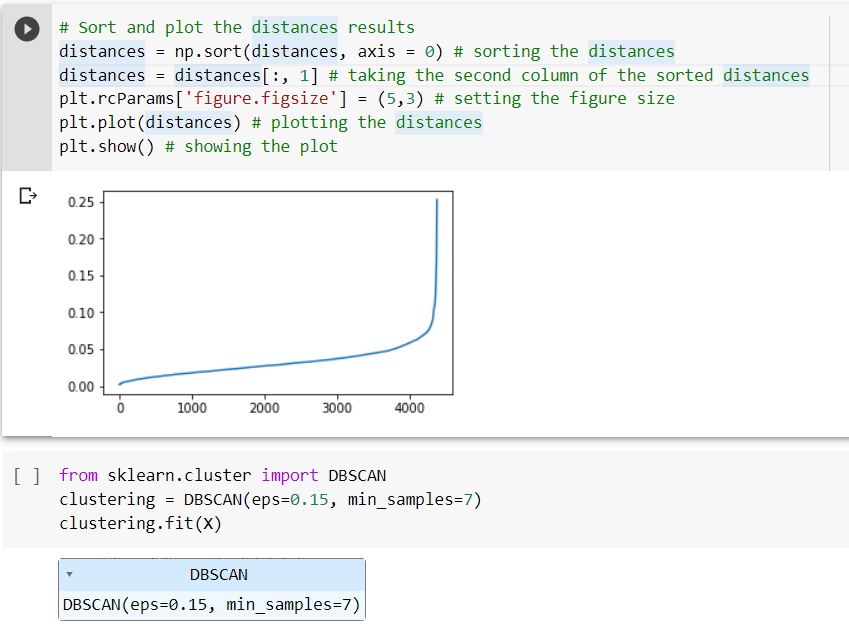
1. Declare the features for clustering.

****

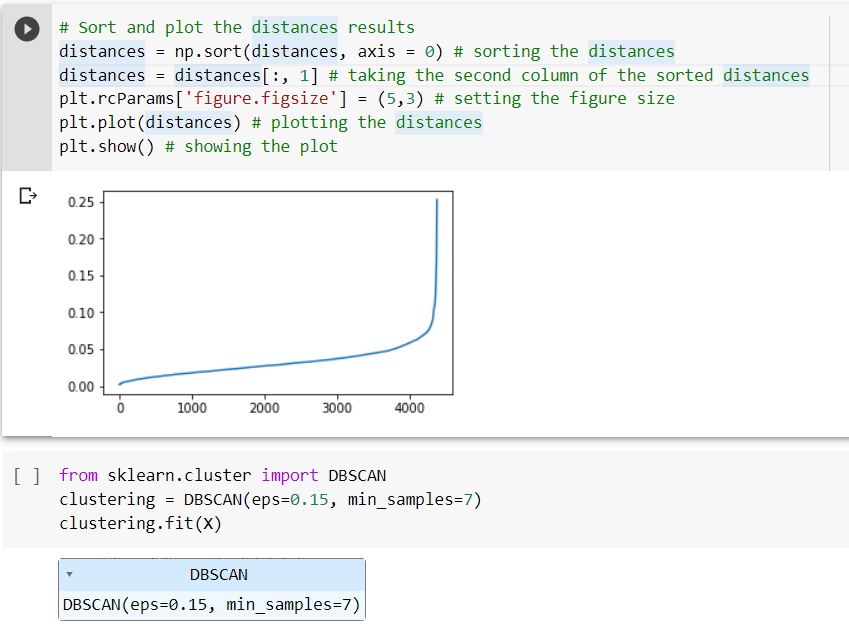
1. Importing the Nearest Neighbor library and fitting our features in it.

****

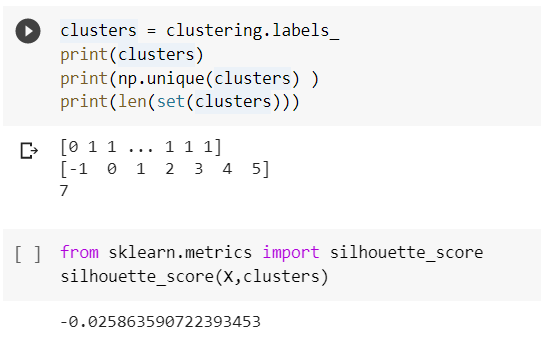
1. Plotting the k-distance graph which will be useful for determining the epsilon value for DBScan. The point on y axis from where the graph takes a steep and sudden increase will be our point of epsilon. Here it will be 0.10 or 0.15.

****

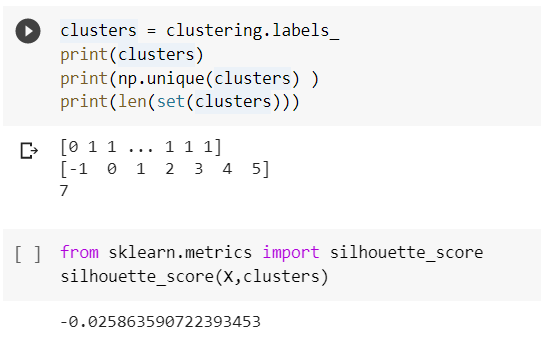
1. Importing the DBSCAN library and fitting our features in it and creating the model.

****

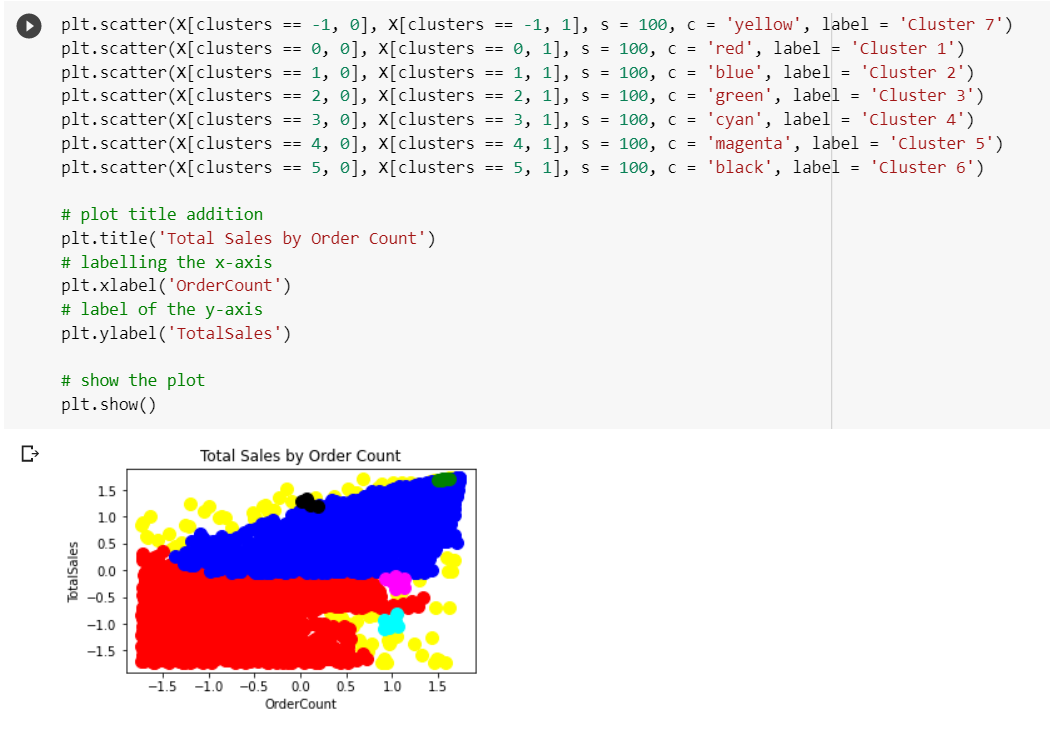
1. Checking for number of clusters and cluster labels. Here -1 represents the point which are outliers.

****

1. Calculating the silhouette score which represents the strength of our model.

****

1. Plotting the clusters.

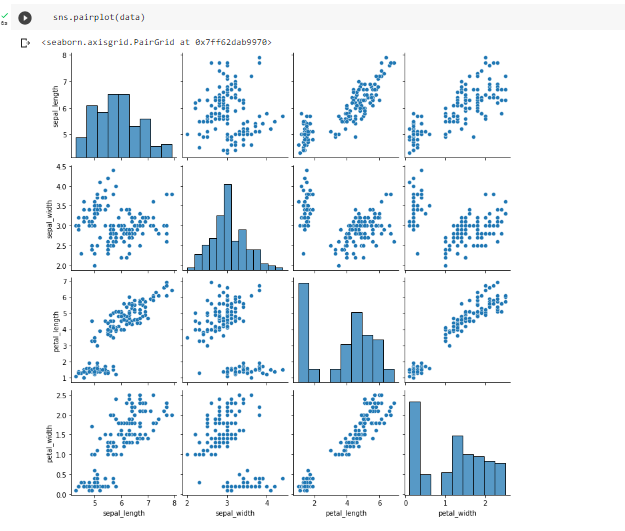
****

**HIERARCHICAL CLUSTERING**

1.Load the Iris dataset.Drop NaN values if any and also species column .

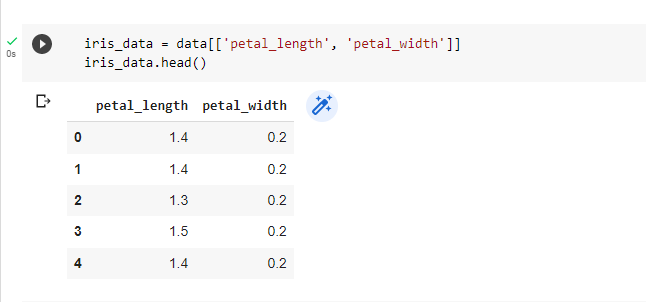
****

2. Plotting a pair plot to see if we can find any relation between the attributes.

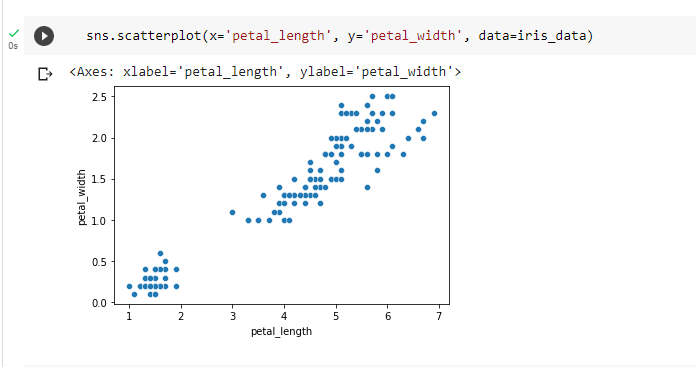
****

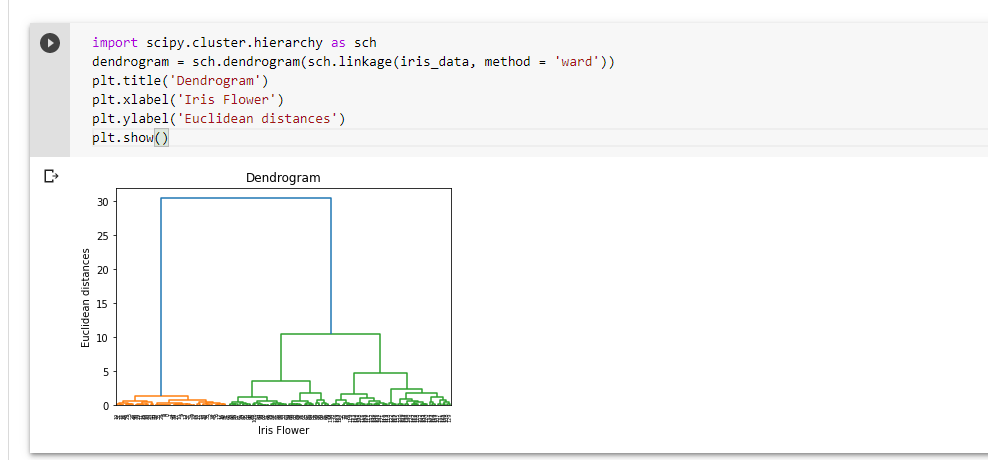
From the output, you can clearly see that there is a positive correlation between the petal length and petal-width column which is a good indicator for clustering.

1. Creating a new data frame with petal-length and petal-width columns.

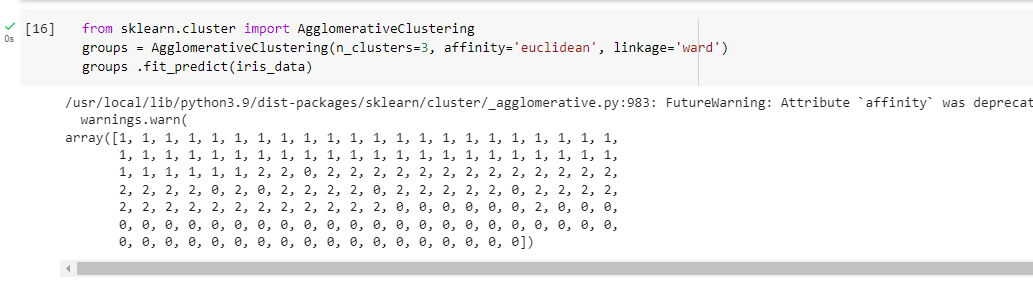
****

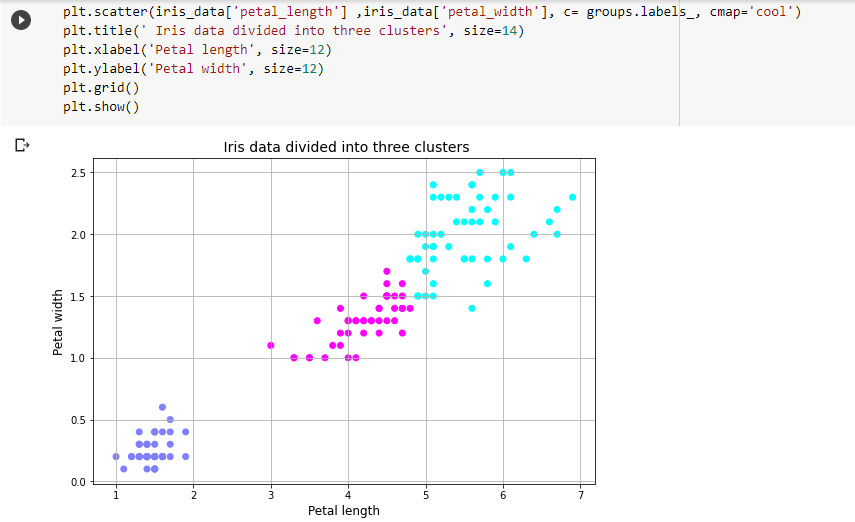
4.Plotting scatterplot of above dataframe

****

5. Plotting the dendrogram.****

6.Dividing data into three clusters

****

****

**CONCLUSION:**

Clustering Algorithms such as K-means, DBSCAN, and Hierarchical Clustering were studied thoroughly in this experiment and accordingly implemented on our datasets. The Clusters formed were also plotted for a better understanding of how these algorithms actually work.