**CUSTOMER SEGMENTATION USING K-MEANS CLUSTERING**

**Objective :**

This Project aims to perform the customer segmentation for the customers of an E-Commerce website name Olist

**Required Libraries:**

* **Pandas -** Python library used for working with data sets
* **Numpy -** Used to perform a wide variety of mathematical operations on arrays
* **Matplotlib –** To plot and visualize interactive graphs
* **Seaborn -**  Provides a higher-level interface for creating statistical graphs
* **Sklearn-**
* **Kneed-**

**Steps Followed:**

The following data explorations were performed to understand the data more.

1.Imports the required datasets

2.Merge the datasets for comprehensive analysis.

i) Merge customer\_df and order\_df based on customer\_id and store as olist\_df1

ii) Merge olist\_df1 with customer\_review\_df and store as olist\_df2

iii) Merge olist\_df2 with order\_items\_df and store as olist\_df3

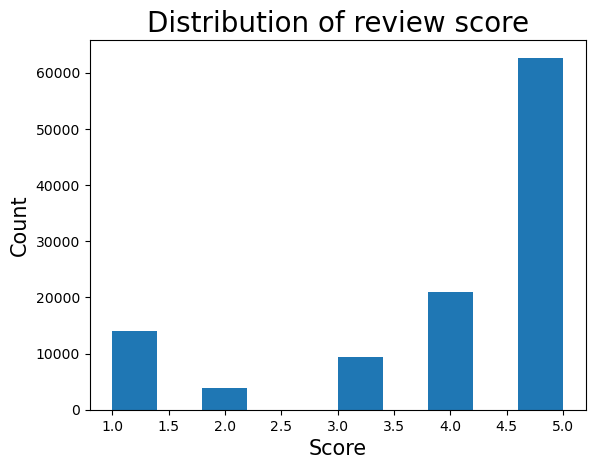
iv) Merge olist\_df3 with product\_df and store as olist\_df4

3. Analyse the olist\_df4 dataframe and create olist\_df with important attributes that are required for our analysis

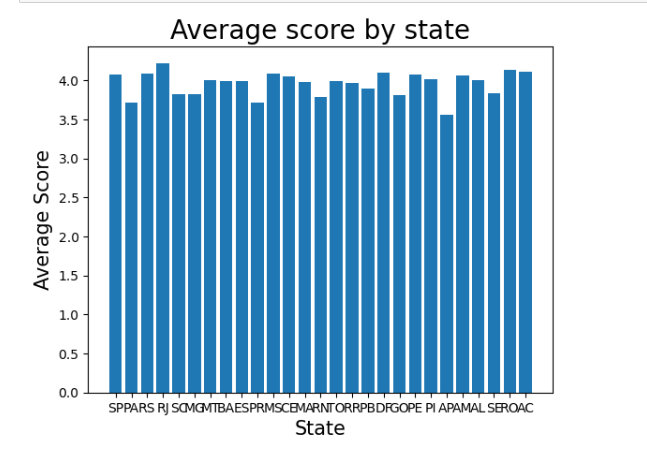
4.Explore the number of records present in the olist\_df using shape function



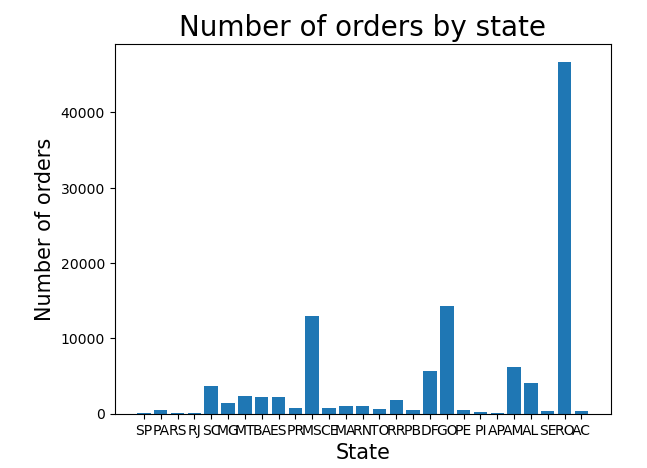
5. Obtain the distribution of review score – The below given distribution shows that number of records with higher ratings are quite high. This implies good quality products and service from Olist.



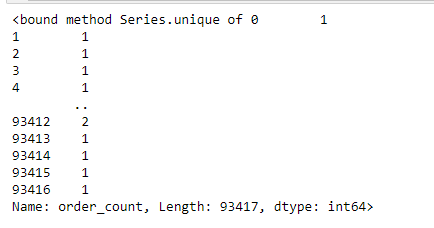
6.Obtain the distribution of review score by state – The given below distribution shows the average review score based on state



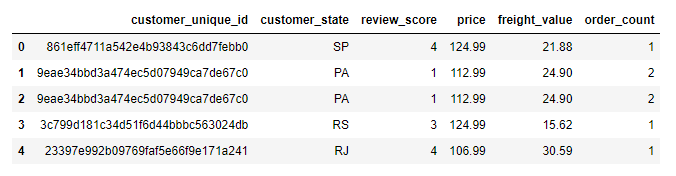
7.Obtain the count of orders by state - The below given distribution shows that the customers in Rio de jeniro (RO) shown more interest to buy olist products.



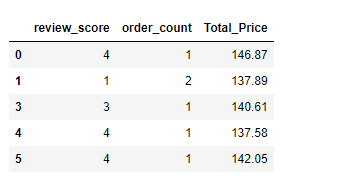
8.Obtain the number of orders placed by every customer –To count the number of orders placed by every customer by using aggregate function by grouping them using customer\_unique\_id



9. Create features data frame with the required features



10.Prepare the dataset for clustering by selecting the required features , summarizing the total\_price and by cleaning the data



11.Standardize feature\_2 dataframe

**STANDARDIZATION:**

Standardization is a common preprocessing step before applying clustering algorithms. Standardization involves transforming the data to have a mean of 0 and a standard deviation of 1. It helps algorithms treat each feature equally during the clustering process, improving the stability and performance of the clustering results.

**K-MEANS CLUSTERING**

K-means clustering is a popular unsupervised machine learning algorithm used for partitioning a dataset into a specific number of clusters (k). The goal of K-means is to group similar data points into clusters, with each cluster having a centroid (center point) that represents the mean of the data points in that cluster. The algorithm iteratively assigns data points to clusters and updates the centroids until convergence.

Here are the main steps of the K-means algorithm:

1. **Initialization:**
   * Choose the number of clusters (k).
   * Randomly initialize k centroids, one for each cluster.
2. **Assignment Step:**
   * Assign each data point to the cluster whose centroid is the closest (usually based on Euclidean distance).
   * Formally, each data point is assigned to the cluster with the nearest centroid.
3. **Update Step:**
   * Recalculate the centroids of the clusters based on the mean of the data points assigned to each cluster.
4. **Repeat:**
   * Repeat steps 2 and 3 until convergence. Convergence occurs when the assignment of data points to clusters and the positions of centroids no longer change significantly.

The main objective of the K-means algorithm is to minimize the within-cluster sum of squared distances (SSE). The SSE is calculated as the sum of the squared distances between each data point and its assigned cluster centroid.

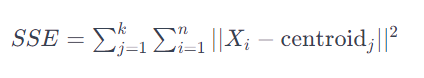
The optimal number of clusters can be chosen based on

1. Elbow Method
2. Silhouette Score.

**ELBOW METHOD**

12.Determine sum of squared error(SSE)

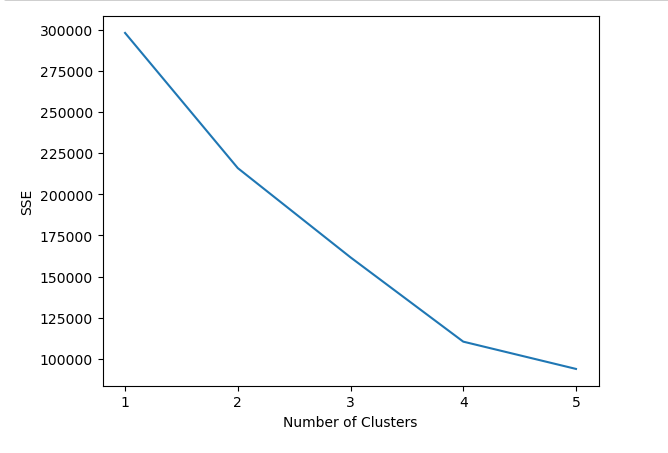
**SUM OF SQUARED ERROR:**

****

The Sum of Squared Errors (SSE) is a measure used to evaluate the performance of a clustering algorithm, particularly in the context of K-means clustering. SSE calculates the sum of the squared differences between each data point in a cluster and the centroid of that cluster. The goal in K-means clustering is to minimize the SSE. A lower SSE indicates that the data points are closer to the centroids of their respective clusters, suggesting a more compact and well-defined clustering solution.

It's important to note that SSE alone may not be sufficient for evaluating the quality of a clustering solution, and other metrics or domain-specific considerations should be taken into account. Additionally, SSE tends to decrease as the number of clusters increases, so it is often used in conjunction with methods like the Elbow Method to determine the optimal number of clusters.

13. Plot the Sum of squared error using elbow method to find the elbow point. The below given graph shows the number of clusters

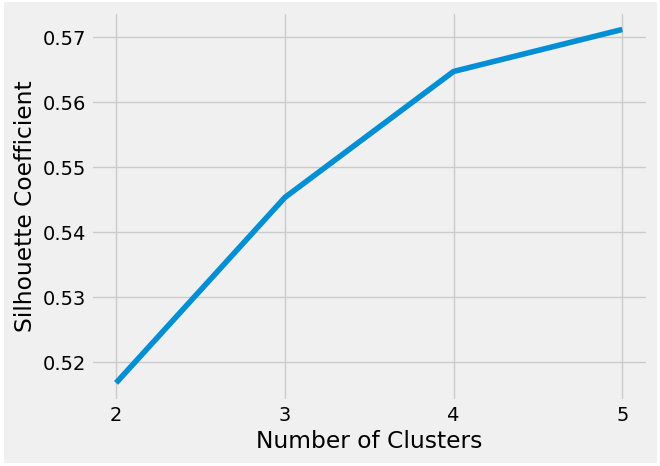


**SILHOUETTE COEFFICIENT**

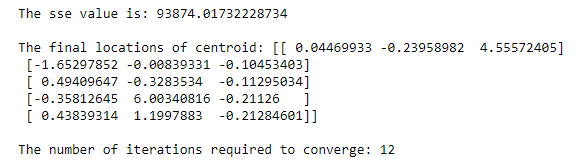
The average silhouette coefficient across all dataset occurrences is used to calculate the silhouette score. The silhouette coefficient, which ranges from -1 to 1, measures how close points in one cluster are to points in nearby clusters. [16]. The silhouette coefficient is calculated as follows:

**Silhouette coefficient = b-a / max(a,b)**

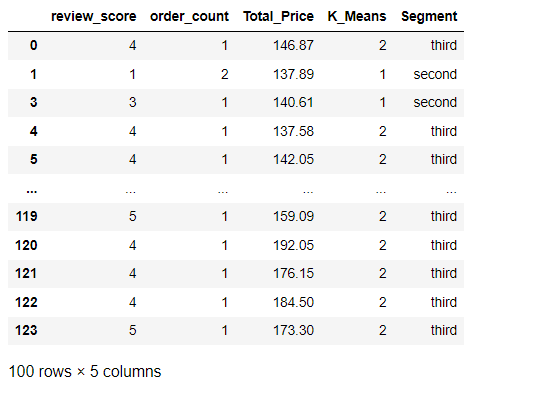
10.Calculate the silhouette coefficient and plot the silhouette score. From the below graph we observe 5 is the optimal number of clusters to perform clustering

****

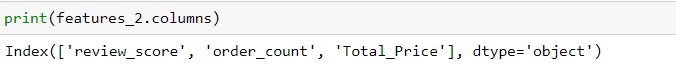
11. Perform K-means clustering with optimal number of clusters as 5 and calculate the sse value and locations of centroids in the cluster.



12. Adding a K-Means cluster label to your **features\_2** dataframe based on the predictions made by our model and map the cluster labels to understandable names (first, second, third, fourth, fifth)



13. Finally export the dataset as customer\_segment.csv



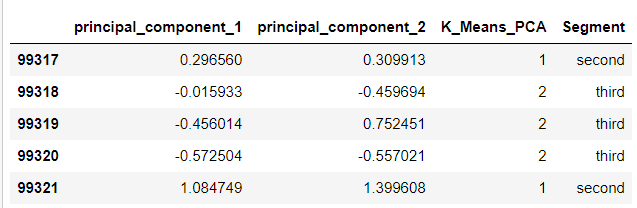
**PCA – Principal Component Analysis**

PCA is a class from scikit-lean (sklearn) to perform dimensionality Reduction on a dataset.

It reduces the component based on the given parameter(component=2). The dataset will be transformed

to 2-dimensional Space. This fits the line and stored in a variable als\_pca.

we create a DataFrame for the two columns to get the result. Adding the new column ‘K\_Means\_PCA’ to the DataFrame which contains cluster labels assigned by K-Means clustering model. Then mapping the segments and then we printing the last 5 rows of the DataFrame.



14 .Create the Scatterplot by using sns with x axis as principal component 2 and y axis as principal component 1 and hue represents the color based on the segment.

The final Clusters formed by PCA components for the Customer\_segmentation is,

