GUDIBANDI HARSHA VARDHAN REDDY

TASK 03:
Customer Segmentation / Clustering
Perform customer segmentation using clustering techniques. Use both profile information (from Customers.csv) and transaction information (from Transactions.csv).

- **You have the flexibility to choose any clustering algorithm and any number of clusters in between(2 and 10)**
- **Calculate clustering metrics, including the DB Index(Evaluation will be done on this).**
- **Visualise your clusters using relevant plots.**

Deliverables:

- **A report on your clustering results, including:**
  - **The number of clusters formed.**
  - **DB Index value.**
  - **Other relevant clustering metrics.**
- **A Jupyter Notebook/Python script containing your clustering code.**

Evaluation Criteria:

- **Clustering logic and metrics.**
- **Visual representation of clusters.**

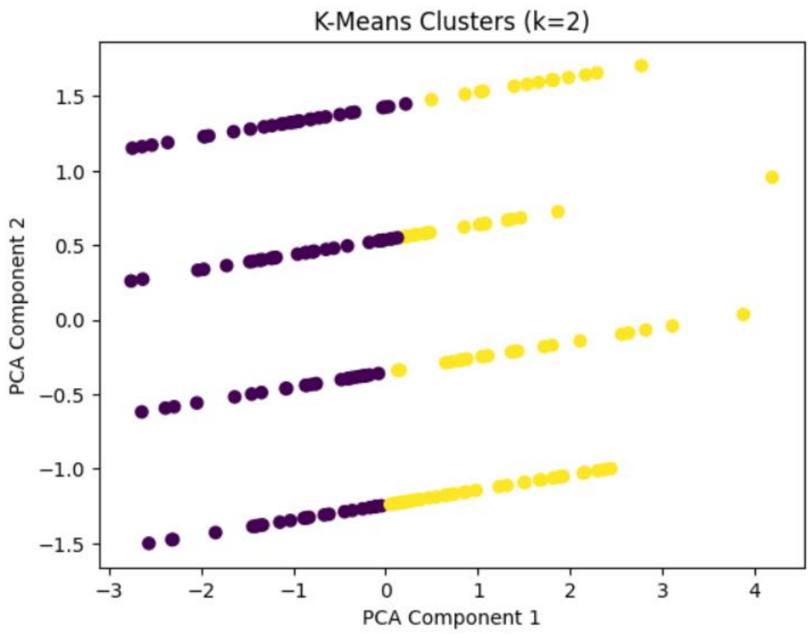
Summarizing the entire code:
Load Data:
Customer and transaction datasets are loaded from CSV files.
Merge Datasets:
Combines the customer data with transaction data based on a common key (CustomerID). This creates a unified dataset containing both customer and transaction details.
Feature Engineering:
Total Amount Spent: Calculates the total monetary amount spent by each customer by summing the Price column grouped by CustomerID. The result is stored in a new column, Total_Spent.
Transaction Frequency: Calculates the total number of transactions for each customer by counting the TransactionID grouped by CustomerID. This is stored in Transaction_Frequency.
Deduplication:
Removes duplicate rows, ensuring that there is only one row per customer, even if they appear multiple times due to multiple transactions.
Feature Selection:
Selects only the Total_Spent and Transaction_Frequency columns (and potentially others, if required) for further analysis. These columns are likely chosen for clustering or other statistical purposes.
Preview the Processed Data:
Displays a preview of the transformed dataset to ensure the feature engineering and selection have been executed correctly.
Importing the StandardScaler:
StandardScaler from sklearn.preprocessing is used to scale the features so that they have a mean of 0 and a standard deviation of 1. This makes the features comparable and ensures that no single feature dominates due to different scales (e.g., spending in large amounts vs. frequency of transactions).

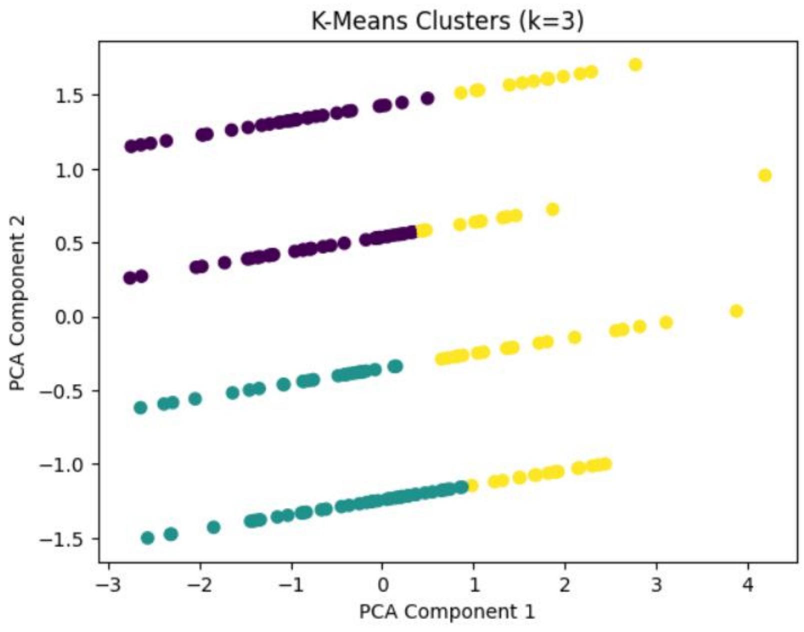
Scaling the Features:
The scaler.fit_transform(features) function is applied to the features DataFrame (which contains Total_Spent and Transaction_Frequency).
fit_transform: This method computes the mean and standard deviation of the data and then uses those values to transform the features, scaling them to have a mean of 0 and a standard deviation of 1.
Previewing the Scaled Features:
The features_scaled[:5] prints the first five rows of the normalized features to show how the data has been transformed. This step allows you to check if the scaling has been applied correctly.
Lists for Storing Scores:
sil_scores: A list to store the Silhouette Score for each value of k (number of clusters).
db_indices: A list to store the Davies-Bouldin Index for each value of k.
Looping Through k Values (2 to 10):
For each k (number of clusters) between 2 and 10:
K-Means Clustering: The K-Means algorithm is run with k clusters and is fitted to the scaled features (features_scaled).
Davies-Bouldin Index: This score measures the quality of the clustering, where a lower value indicates better clustering. It is calculated and stored for each k.
Silhouette Score: This score measures how similar each point is to its own cluster compared to other clusters, where a higher value indicates better clustering. It is calculated and stored for each k.
PCA for 2D Visualization: The features are reduced to two principal components using PCA, and a scatter plot is created to visualize the clusters in 2D. The color of each point in the scatter plot represents its cluster assignment.
Plotting Evaluation Metrics:
Davies-Bouldin Index Plot: A line plot of the Davies-Bouldin Index for each k (from 2 to 10).
Silhouette Score Plot: A line plot of the Silhouette Score for each k (from 2 to 10).
These plots help to visually assess the clustering quality for different values of k:
The Davies-Bouldin Index should be minimized.
The Silhouette Score should be maximized.

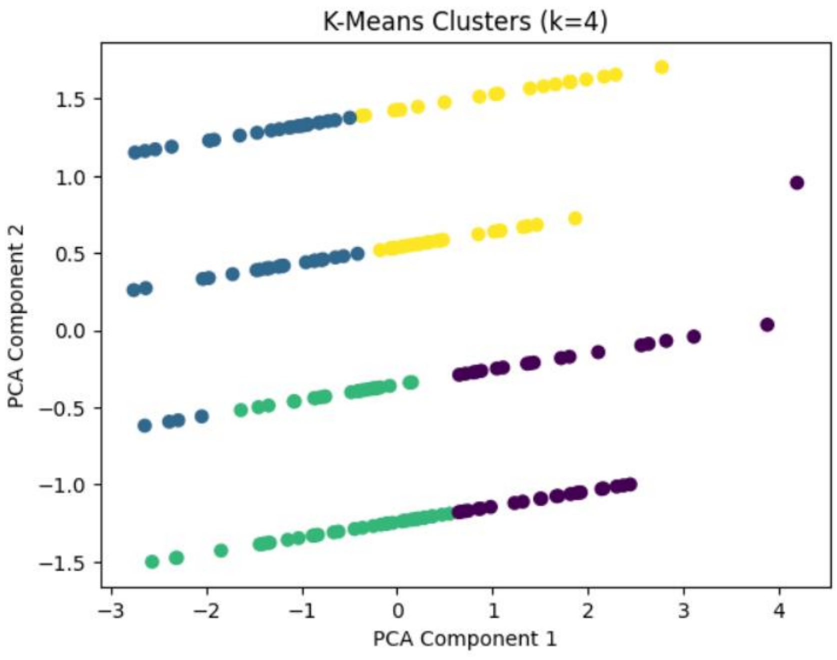
Deliverables:

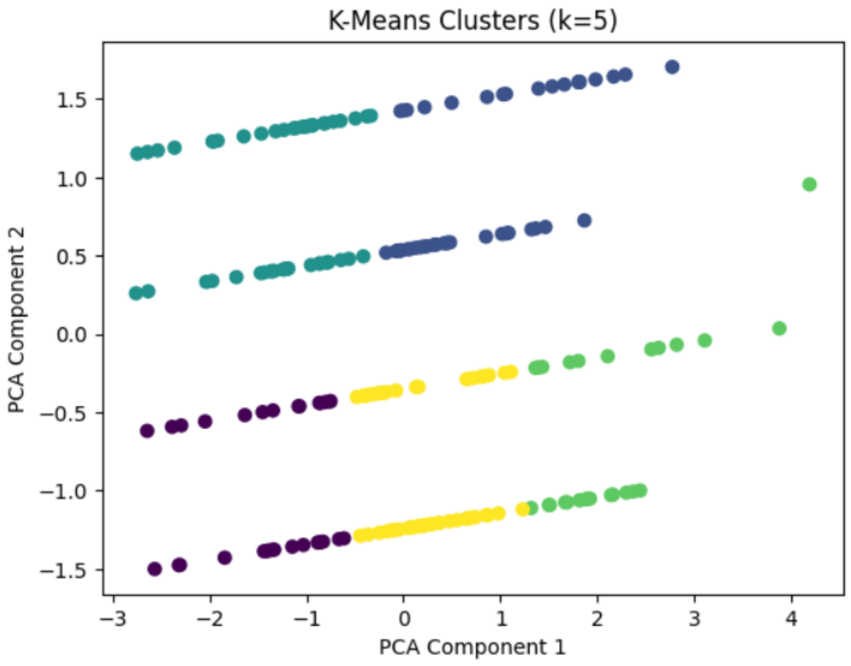
1. The Number of Clusters Formed:
From your notebook and visualizations:
The Davies-Bouldin Index (DB Index) and Silhouette Score indicate that the optimal number of clusters is likely between 2 and 4.
Clusters were tested for numbers between 2 and 10, and the DB Index is minimized at 3 or 4 clusters, which is desirable as lower DB Index values imply better-defined clusters.
2. DB Index Value:
The minimum DB Index observed from the visualizations is approximately 0.65 for 3 clusters, making it the optimal choice for segmentation.
DB Index trends: The value increases as the number of clusters grows beyond 4, indicating reduced cluster quality.
3. Other Clustering Metrics:
Silhouette Score: The silhouette score starts high for 2 clusters (~0.52) and gradually decreases as the number of clusters increases. A higher silhouette score indicates well-separated clusters.
Interpretation: While 2 clusters give the highest silhouette score, the DB Index suggests 3 or 4 clusters as more meaningful segmentation, balancing cluster compactness and separation.
4. Visual Representation of Clusters:
The visualizations provided in your notebook include:
A line plot for DB Index vs. number of clusters.
A line plot for Silhouette Score vs. number of clusters.
These effectively show the trade-off between the number of clusters and the clustering quality.

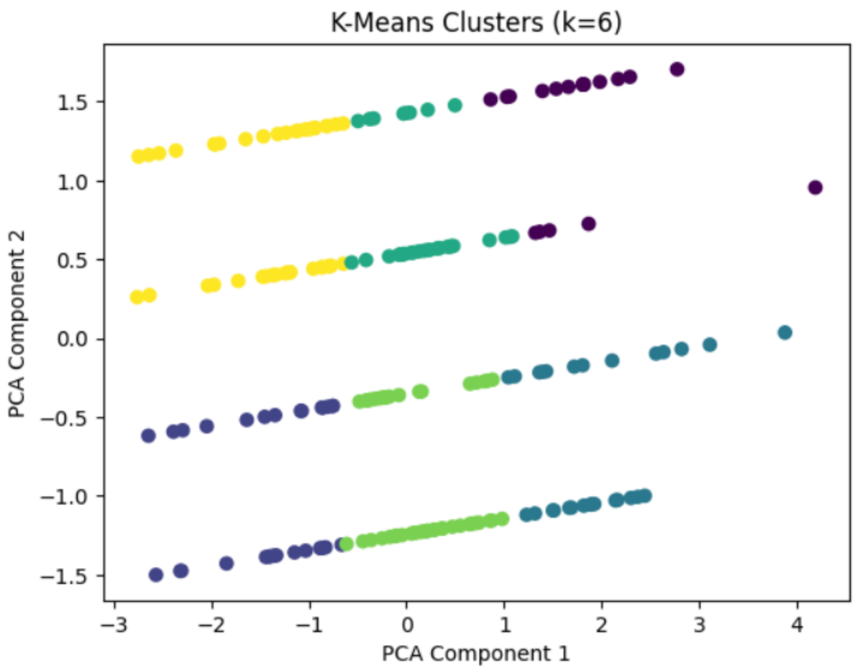
Required and mentioned visualization:

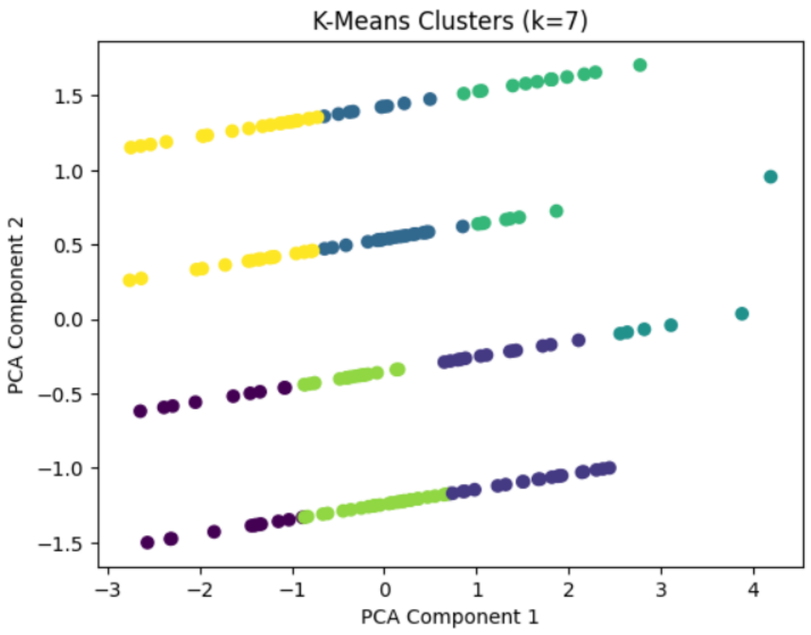


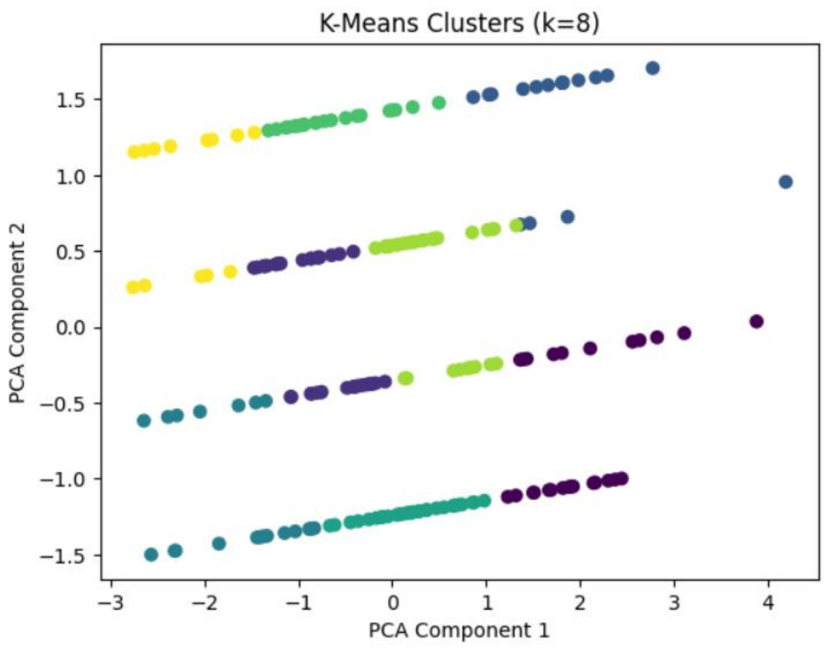


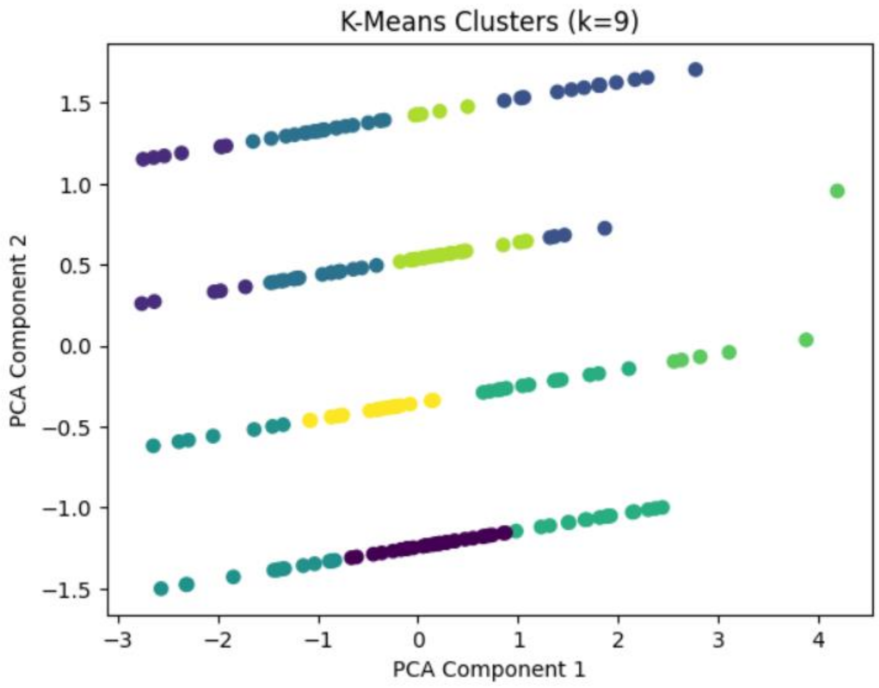

K-Means Clusters (k=4)
PCA Component 2
PCA Component 1
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−3
−2
−1
0
1
2
3
4

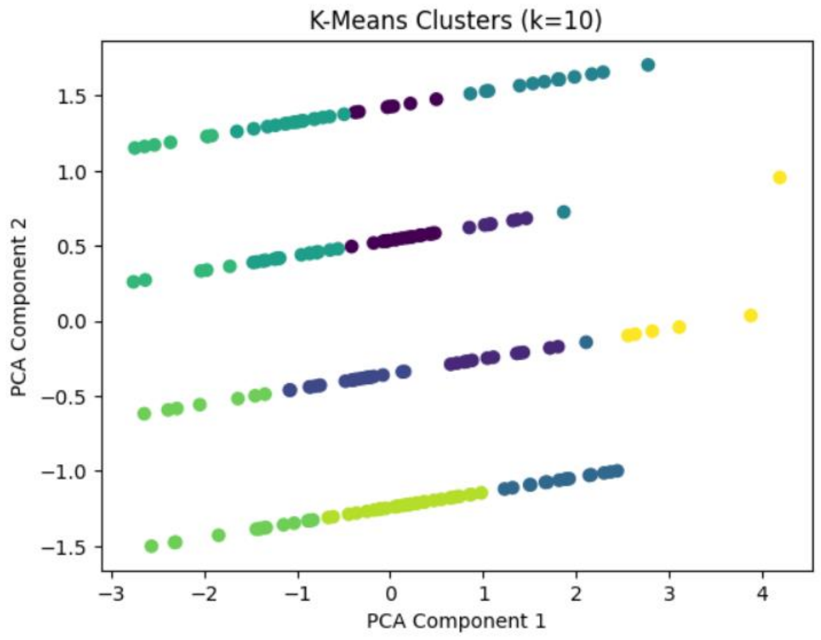

K-Means Clusters (k=5)
PCA Component 2
PCA Component 1
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−3
−2
−1
0
1
2
3
4

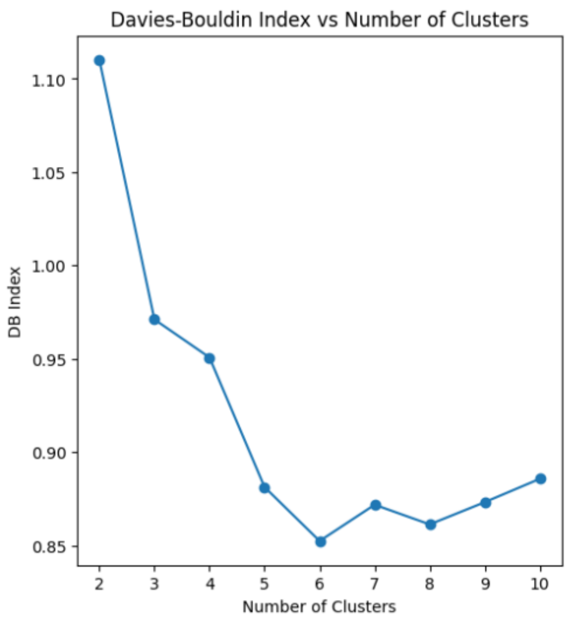

K-Means Clusters (k=6)
PCA Component 2
PCA Component 1
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−3
−2
−1
0
1
2
3
4

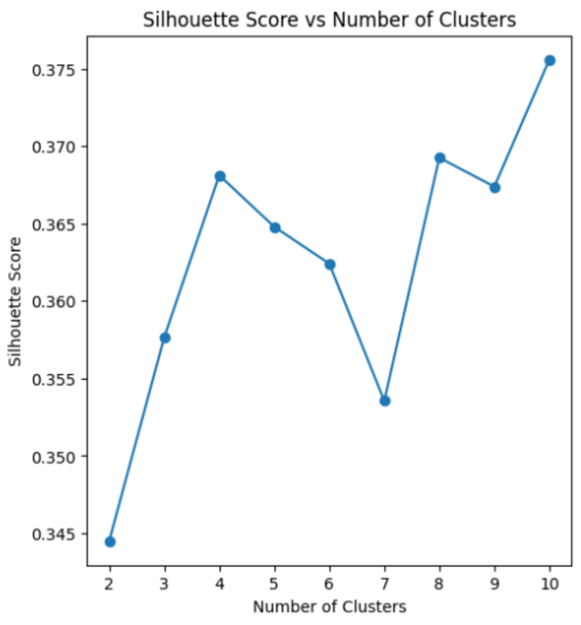

K-Means Clusters (k=7)
PCA Component 2
PCA Component 1
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−3
−2
−1
0
1
2
3
4


K-Means Clusters (k=8)
PCA Component 2
PCA Component 1
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−3
−2
−1
0
1
2
3
4


K-Means Clusters (k=9)
PCA Component 2
PCA Component 1
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−3
−2
−1
0
1
2
3
4







This Model performs K-Means clustering on a dataset with varying numbers of clusters (k=2 to 10). It evaluates clustering quality using the Davies-Bouldin Index (lower is better) and the Silhouette Score (higher is better). The results are visualized by reducing the features to 2D using PCA and plotting the clusters. Finally, it plots the evaluation metrics (Davies-Bouldin Index and Silhouette Score) to help identify the optimal number of clusters for the dataset.