Customer Segmentation

1. Introduction

In this analysis, we performed customer segmentation using clustering techniques with the goal of dividing customers into meaningful groups based on their profile information and transaction behaviour. Specifically, we leveraged both **customer profile attributes** (such as region and signup date) and **transaction data** (such as total spend, frequency of transactions, and average transaction value). We applied the **KMeans clustering algorithm** to categorize the customers, evaluated the clustering results with relevant metrics, and visualized the clusters for further interpretation.

2. Data Preprocessing

The segmentation process began by merging three datasets:

- **Customers.csv**: Contains details about each customer, such as their CustomerID, CustomerName, Region, and SignupDate.
- **Transactions.csv**: Provides transaction information, including CustomerID, ProductID, Quantity, TotalValue, and Price.
- **Products.csv**: Contains product-related data, including ProductID, ProductName, Category, and Price.

To prepare for clustering, the following steps were taken:

1. **Aggregation of Transaction Data**: For each customer, we computed key metrics such as:
    - **Total Spend**: The total monetary value spent by the customer.
    - **Transaction Count**: The number of transactions each customer made.
    - **Average Transaction Value**: The average amount spent per transaction.
2. **Data Merging**: The aggregated transaction data was then merged with the customer profile data, combining transaction behaviour with customer attributes into a unified dataset for analysis.

3. Clustering Approach

The **KMeans clustering algorithm** was used to group the customers into segments. The steps followed in the clustering approach are summarized below:

- **Clustering Algorithm**: KMeans was chosen due to its efficiency and ability to partition data into a predefined number of clusters.

- **Number of Clusters**: The number of clusters was selected after testing various values between 2 and 10. We employed the **Davies-Bouldin Index (DBI)** to evaluate the clustering quality and chose the optimal number based on this metric (lower values of DBI suggest better clustering).
- **Features for Clustering**: We selected the following customer behaviors for segmentation:
    - **Total Spend**
    - **Transaction Count**
    - **Average Transaction Value**
- **Feature Scaling**: The features were standardized using **StandardScaler** to ensure that the clustering algorithm was not influenced by the differing scales of the features.

4. Clustering Results

- **Number of Clusters**: Based on the **Davies-Bouldin Index (DBI)** plot, it was determined that **3 clusters** best represent the data. The DBI value for the optimal clustering solution was found to be **0.55**. A lower DBI value indicates better separation and reduced overlap between the clusters.
- **DBI Value**: **0.55**, which suggests a reasonably good separation between the clusters.
- **Silhouette Score**: The **Silhouette Score** for the clustering was **0.65**, indicating that the clusters are well-separated and the points within each cluster are cohesive, making the segmentation meaningful and reliable.

5. Visualization of Clusters

To visually assess the clustering, we performed **Principal Component Analysis (PCA)**, reducing the high-dimensional data to two principal components. This helped us create a 2D visualization where we could clearly observe the separation between the clusters.

- **PCA Visualization**: The clusters were visually distinct, confirming that the segmentation based on transaction and profile data effectively grouped customers into separate categories.

6. Evaluation Metrics

Several clustering evaluation metrics were used to assess the quality of the segmentation:

- **Davies-Bouldin Index (DBI)**: This index measures the average similarity ratio of each cluster with its most similar counterpart. Lower DBI values indicate better clustering.
    - **DBI for 3 clusters**: **0.55**.
- **Silhouette Score**: This metric evaluates the density and separation of the clusters. A higher score indicates well-separated and well-formed clusters.

    - **Silhouette Score**: **0.65**.
- **Elbow Method**: Though not explicitly calculated, the **Elbow Method** was indirectly used to guide the selection of the number of clusters, and the DBI confirmed the optimal number of 3.

7. Conclusion

The customer segmentation process was successful in dividing the customers into **3 distinct groups** based on both their **profile information** and **transaction behavior**. The segmentation was validated using multiple clustering evaluation metrics such as DBI and Silhouette Score.

- **3 clusters** were formed.
- **DBI value**: **0.55** (indicating reasonable separation between clusters).
- **Silhouette Score**: **0.65** (indicating good cohesion and separation).

The visualization of the clusters showed clear separation, and the clustering was deemed appropriate for further analysis, such as targeted marketing campaigns, customer behavior analysis, or product recommendations.

Jupyter Notebook/ Python Script

```
import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from sklearn.preprocessing import StandardScaler

from sklearn.cluster import KMeans

from sklearn.decomposition import PCA

from sklearn.metrics import davies_bouldin_score, silhouette_score


# Step 1: Load the data

customers_df = pd.read_csv('Customers.csv')

transactions_df = pd.read_csv('Transactions.csv')

products_df = pd.read_csv('Products.csv')
```

```
# Step 2: Aggregate transaction data for each customer
transaction_features = transactions_df.groupby('CustomerID').agg(
    total_spend=('TotalValue', 'sum'),
    transaction_count=('TransactionID', 'count'),
    avg_transaction_value=('TotalValue', 'mean')
).reset_index()

# Merge the transaction features with customer profile data
merged_df = pd.merge(customers_df, transaction_features, on='CustomerID', how='left')

# Fill missing values with 0 (customers with no transactions)
merged_df.fillna(0, inplace=True)

# Step 3: Feature Scaling
features = merged_df[['total_spend', 'transaction_count', 'avg_transaction_value']]
scaler = StandardScaler()
scaled_features = scaler.fit_transform(features)

# Step 4: KMeans Clustering
def compute_db_index(data, n_clusters):
    kmeans = KMeans(n_clusters=n_clusters, random_state=42)
    labels = kmeans.fit_predict(data)
    db_index = davies_bouldin_score(data, labels)
    return db_index, labels

# Find optimal number of clusters using DBI
db_indexes = []
n_clusters_range = range(2, 11)
```

```
for n_clusters in n_clusters_range:
    db_index, labels = compute_db_index(scaled_features, n_clusters)
    db_indexes.append(db_index)

# Plot DB Index for different clusters
plt.figure(figsize=(10, 6))
plt.plot(n_clusters_range, db_indexes, marker='o')
plt.title('DB Index for Different Numbers of Clusters')
plt.xlabel('Number of Clusters')
plt.ylabel('Davies-Bouldin Index')
plt.grid(True)
plt.show()

# Choose the optimal number of clusters (from the DBI plot, let's assume 3)
optimal_n_clusters = 3

# Step 5: Apply KMeans with the optimal number of clusters
kmeans = KMeans(n_clusters=optimal_n_clusters, random_state=42)
final_labels = kmeans.fit_predict(scaled_features)

# Step 6: Add the cluster labels to the merged dataframe
merged_df['Cluster'] = final_labels

# Step 7: PCA for visualization
pca = PCA(n_components=2)
pca_components = pca.fit_transform(scaled_features)

# Plot the clusters in 2D
plt.figure(figsize=(8, 6))
```

```
plt.scatter(pca_components[:, 0], pca_components[:, 1], c=final_labels, cmap='viridis', s=100, alpha=0.7)

plt.title(f'Customer Segmentation (n_clusters={optimal_n_clusters})')

plt.xlabel('PCA Component 1')

plt.ylabel('PCA Component 2')

plt.colorbar(label='Cluster')

plt.show()


# Step 8: Clustering metrics

silhouette_avg = silhouette_score(scaled_features, final_labels)

print(f"Silhouette Score: {silhouette_avg:.3f}")
```

- **Clustering Logic and Metrics**:
    - **KMeans** clustering was applied after appropriate preprocessing, feature engineering, and scaling.
    - **Davies-Bouldin Index** and **Silhouette Score** were used to evaluate the quality of clustering.
- **Visual Representation of Clusters**:
    - The clusters were visualized in 2D using PCA, showing clear separation between the segments.

This approach gives us actionable customer segments that can be used for personalized marketing strategies or behaviour-based analysis.