Task 3: Customer Segmentation / Clustering

Objective:

Perform customer segmentation using clustering techniques by leveraging customer profile and transaction data. The goal is to segment customers into meaningful groups and evaluate the clustering quality using the Davies-Bouldin (DB) Index.

Steps to Solution:

Step 1: Data Preparation

1. **Load the Datasets:**
   - Three datasets are provided: Customers.csv, Products.csv, and Transactions.csv.
   - Read these CSV files into pandas DataFrames.
2. **Merge the Datasets:**
   - To create a comprehensive dataset, we performed the following merges:
     - Merge Transactions.csv with Products.csv on ProductID.
     - Merge the result with Customers.csv on CustomerID.
   - Renamed columns for clarity:
     - Price in Transactions.csv was renamed to ProductPrice.
     - Price in Products.csv was renamed to Price_product.
3. **Create Customer Profiles:**
   - Aggregate transaction data by CustomerID to calculate key metrics:
     - **TotalSpending:** Sum of TotalValue.
     - **TotalTransactions:** Count of transactions.
     - **TotalQuantity:** Total quantity purchased.
     - **AvgProductPrice:** Mean of product prices.
     - **FavoriteProduct:** Most frequently purchased product (using mode).

- Used one-hot encoding for the FavoriteProduct column to convert it into numerical features.
- Merged demographic information from Customers.csv into the customer profile.

Step 2: Normalize Features

- Dropped non-numeric and irrelevant columns such as CustomerID, CustomerName, Region, and SignupDate.
- Applied StandardScaler to normalize numerical features, ensuring all features had zero mean and unit variance. This step helps improve the performance of clustering algorithms.

Step 3: Clustering and Evaluation

1. **Range of Clusters:**
    - Defined a range of clusters (k) to evaluate: 2 to 10 clusters.
2. **Clustering Algorithms:**
    - Used the KMeans algorithm to perform clustering.
3. **Evaluation Metrics:**
    - **Davies-Bouldin Index (DB Index):** Lower values indicate better cluster separation and compactness.
    - **Silhouette Score:** Measures how similar a data point is to its own cluster compared to other clusters. Higher scores indicate better-defined clusters.
4. **Iterative Clustering:**
    - For each k in the range (2 to 10):
        - Fit the KMeans model and predict cluster labels.
        - Compute DB Index and Silhouette Score.
    - Recorded scores for each value of k and identified the optimal number of clusters (optimal_k) with the lowest DB Index.

Step 4: Visualize Metrics

- Plotted the DB Index and Silhouette Scores against the number of clusters (k).
- Visual inspection helped verify the optimal number of clusters.

Step 5: Final Clustering

1. **Optimal KMeans:**
   - Re-ran the KMeans algorithm using optimal_k to generate final cluster labels.
2. **PCA for Visualization:**
   - Used Principal Component Analysis (PCA) to reduce high-dimensional data to 2D for visualization.
   - Plotted the clusters with distinct colors to observe separations visually.
3. **Cluster Assignment:**
   - Added the cluster labels to the customer_profile DataFrame for further analysis.

Step 6: Reporting and Results

1. **Clustering Report:**
   - **Optimal Clusters (k):** Number of clusters with the lowest DB Index.
   - **Best DB Index:** The minimum DB Index value achieved.
   - **Silhouette Score (for optimal k):** Quality of clustering for the optimal number of clusters.
2. **Visualization:**
   - Metrics plot showing DB Index and Silhouette Scores for different cluster counts.
   - Scatter plot of clusters (reduced to 2D using PCA).

Code Summary:

1. Preprocessing:
   - Merging datasets, creating customer_profile, and normalizing features.

2. Clustering:
    - Iterative clustering using KMeans for a range of clusters (2 to 10).
    - Evaluated DB Index and Silhouette Score.
3. Final Clustering:
    - Identified optimal clusters and visualized results.

Evaluation Metrics:

1. **Davies-Bouldin Index:** Measures intra-cluster similarity and inter-cluster differences. Lower is better.
2. **Silhouette Score:** Measures how well clusters are defined. Higher is better.

Deliverables:

1. Clustering report containing:
    - Optimal clusters, DB Index, and Silhouette Score.
2. Visualizations:
    - Metrics plot (DB Index and Silhouette Scores).
    - 2D PCA scatter plot of clusters.
3. Python script or Jupyter Notebook with the complete solution.

Final Notes:

This process combines transactional and demographic data for customer segmentation. The use of both DB Index and Silhouette Score ensures a robust evaluation of clustering quality, while PCA visualization aids in interpreting the cluster structure.