Name: Ronit Sahoo
Class: D15C
Roll No: 41
Batch: B

MLDL EXPERIMENT NO: 6

Aim:

- Apply K-Means and Hierarchical Clustering on sample datasets.

Dataset Description:

The Credit Card Customer Dataset is a real-world financial analytics dataset that contains detailed information about credit card usage behavior of customers. The dataset is designed to analyze customer spending patterns, repayment behavior, and credit utilization, making it highly suitable for financial analysis, customer segmentation, and risk profiling tasks. It captures transactional and behavioral attributes such as account balance, purchase types, cash advances, credit limits, payment behavior, and tenure, which together provide a comprehensive view of customer financial activity.

Unlike supervised classification datasets that rely on a predefined target variable, this dataset is primarily used for unsupervised learning techniques such as K-Means clustering, Hierarchical clustering, and customer segmentation, where the objective is to discover natural groupings among customers based on their credit usage patterns. Due to its entirely numerical feature space, realistic banking context, and moderate dataset size, it is widely used in applied machine learning experiments related to customer profiling, spending behavior analysis, and personalized financial services.

With appropriate preprocessing such as feature scaling and normalization, the dataset enables meaningful exploration of customer personas, spending intensity, and credit risk behavior.

- **File Type:** CSV (Comma Separated Values)
- **Dataset Size:** 8951 rows × 18 columns

Column Name	Data Type	Description
CUST_ID	Categorical (String)	Unique identifier assigned to each customer
BALANCE	Numerical (Float)	Remaining balance on the credit card account
BALANCE_FREQUENCY	Numerical (Float)	Frequency of balance updates
PURCHASES	Numerical (Float)	Total purchase amount made by the customer
ONEOFF_PURCHASES	Numerical (Float)	Value of one-time purchase transactions
INSTALLMENTS_PURCHASES	Numerical (Float)	Amount spent on installment-based purchases
CASH_ADVANCE	Numerical (Float)	Total cash advance amount taken
PURCHASES_FREQUENCY	Numerical (Float)	Frequency of purchase transactions
CREDIT_LIMIT	Numerical (Float)	Credit limit assigned to the customer
PAYMENTS	Numerical (Float)	Total amount paid by the customer
MINIMUM_PAYMENTS	Numerical (Float)	Minimum payment amount due
PRC_FULL_PAYMENT	Numerical (Float)	Percentage of full payments made
TENURE	Numerical (Integer)	Duration of customer relationship in months

Dataset Source: https://www.kaggle.com/datasets/arjunbhasin2013/ccdata

K-MEANS CLUSTERING

Theory:

K-Means Clustering is an unsupervised machine learning algorithm used to divide a dataset into a fixed number of clusters, denoted by K, based on similarity among data points. The main objective of K-Means is to ensure that customers within the same cluster exhibit similar behavior patterns, while customers in different clusters show significant differences. In the context of credit card data, this enables the identification of distinct customer segments based on spending habits, payment behavior, cash advance usage, and credit utilization.

The algorithm works by minimizing the Within-Cluster Sum of Squares (WCSS), which represents the total squared distance between data points and their respective cluster centroids. Since the dataset consists entirely of numerical features such as balance, purchases, payments, and credit limits, K-Means is particularly well suited for this task. Unlike supervised learning approaches, K-Means does not require labeled outcomes and instead uncovers hidden structure in customer behavior purely through feature similarity.

Given a dataset with n customers and d behavioral attributes, K-Means partitions the data into K non-overlapping clusters. Each cluster Ck is represented by a centroid μk, which is computed as the mean of all customer records assigned to that cluster. The algorithm iteratively updates cluster assignments and centroids until convergence is achieved.

Limitations:

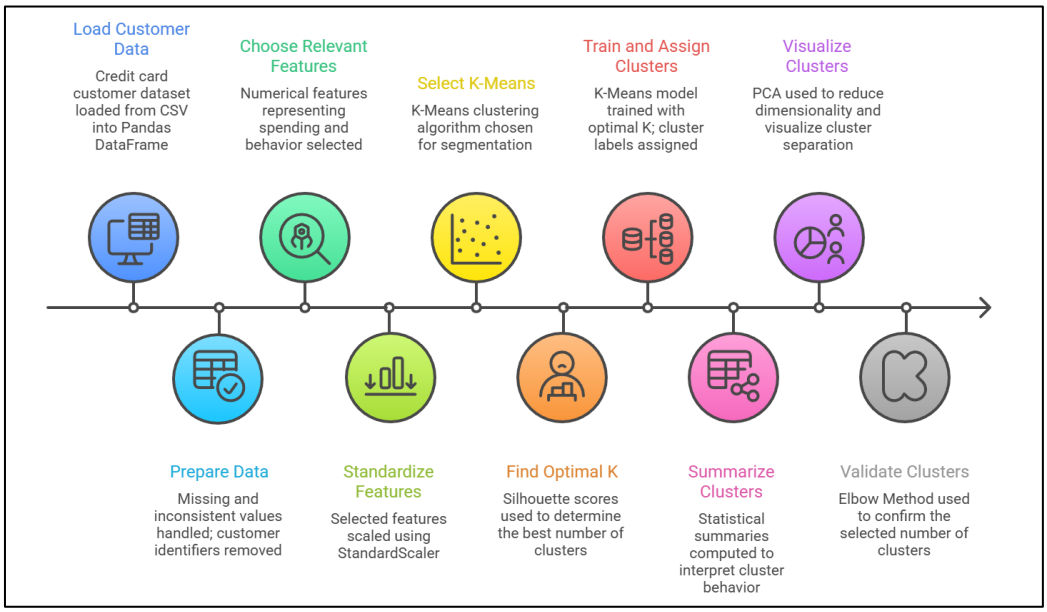
1. **Need to Predefine the Number of Clusters (K):** A major limitation of K-Means is the requirement to specify the number of clusters beforehand. In real-world financial datasets, the true number of customer segments is usually unknown. Choosing an unsuitable value of K may result in overly broad clusters or excessive fragmentation. Although methods such as the Elbow Method and Silhouette Analysis are commonly used to estimate K, they provide approximate guidance rather than a definitive solution.

2. **Sensitivity to Initial Centroid Placement:** K-Means is sensitive to the initial positions of cluster centroids. Different initializations can lead to different clustering outcomes, as the algorithm converges to a local optimum rather than the global minimum. Poor initialization may result in unstable or suboptimal customer segments. Techniques such as K-Means++ improve initialization quality but do not completely eliminate this issue.

3. **Assumption of Spherical and Uniform Clusters:** The algorithm assumes that clusters are roughly spherical, equally sized, and well separated because it relies on Euclidean distance. However, customer behavior patterns may form clusters of varying shapes, sizes, and densities. In such cases, K-Means may incorrectly group customers, leading to less meaningful segmentation results.

Workflow:

1. **Data Collection:** The credit card customer dataset is loaded from a CSV file into a Pandas DataFrame. It contains customer-level financial and transactional information such as balances, purchase behavior, cash advances, payment patterns, and account tenure, which are used for customer segmentation.
2. **Data Cleaning and Preparation:** Records with missing or inconsistent values are handled appropriately to ensure data quality. Customer identifiers are retained only for reference, and the dataset is prepared exclusively with behavior-related numerical attributes.
3. **Feature Selection:** Relevant numerical features representing spending activity, credit utilization, transaction frequency, repayment behavior, and tenure are selected. Non-numerical attributes are excluded to maintain compatibility with distance-based clustering algorithms.
4. **Feature Scaling:** All selected features are standardized using StandardScaler to remove scale differences and ensure that no single feature dominates distance calculations in K-Means clustering.
5. **Model Selection (K-Means):** The K-Means clustering algorithm is selected to group customers into distinct segments by minimizing within-cluster variance and maximizing similarity within each cluster.
6. **Hyperparameter Tuning:** The optimal number of clusters (K) is identified by evaluating silhouette scores for values of K ranging from 2 to 10. The value that produces the highest silhouette score is chosen.
7. **Final Model Training and Cluster Assignment:** The K-Means model is trained using the selected value of K, and a cluster label is assigned to each customer and added to the dataset.
8. **Cluster Profiling:** Cluster-wise statistical summaries, such as mean values of key features, are computed to interpret and compare the financial behavior of different customer segments.
9. **Visualization using PCA:** Principal Component Analysis (PCA) is applied to reduce dimensionality and project the clustered data into two dimensions, enabling clear visualization of cluster separation.
10. **Elbow Method Validation:** The Elbow Method is used to examine changes in within-cluster sum of squares (WCSS) across different values of K, providing additional validation for the selected number of clusters.



Performance Analysis:

Since K-Means is an unsupervised learning algorithm, model performance is evaluated using internal validation metrics and visual analysis, rather than accuracy or classification-based measures. The evaluation focuses on cluster cohesion, separation, and interpretability using Silhouette Score analysis, PCA-based visualization, and hierarchical clustering dendrograms.

- **Optimal Number of Clusters Selection (Silhouette Analysis):** The Silhouette Score plot evaluates cluster quality for values of K ranging from 2 to 10. The results show that the highest silhouette score occurs at K = 3, indicating that this configuration provides the best balance between intra-cluster compactness and inter-cluster separation. Lower silhouette scores for higher values of K suggest diminishing clustering quality beyond this point. Based on this analysis, K = 3 is selected as the optimal number of clusters.
- **Cluster Separation and Visualization (PCA Analysis):** Principal Component Analysis (PCA) is used to project the high-dimensional scaled data into two dimensions for visualization. The PCA scatter plot shows three clearly distinguishable clusters, with reasonable separation along the principal components. Although some overlap is expected due to dimensionality reduction, the visualization confirms that K-Means has captured meaningful structure in customer credit behavior rather than forming random groupings.
- **Hierarchical Clustering Validation (Dendrogram):** A hierarchical clustering dendrogram using Ward linkage is constructed on a subset of the standardized data for readability. The dendrogram reveals natural splits in the data that are consistent with the three-cluster solution identified by K-Means. The presence of clear vertical separation between major branches further validates the chosen number of clusters and supports the stability of the segmentation.

Hyperparameter Tuning:

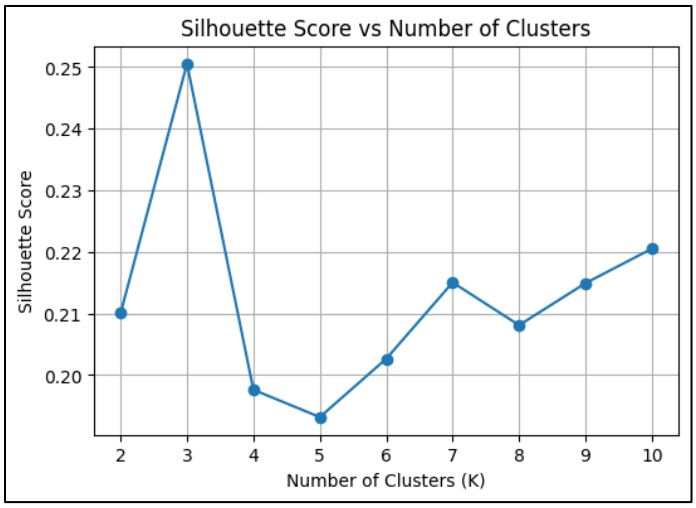
K-Means clustering is highly sensitive to hyperparameter selection, particularly the number of clusters (K) and centroid initialization. Therefore, systematic tuning is essential to obtain meaningful segmentation results.

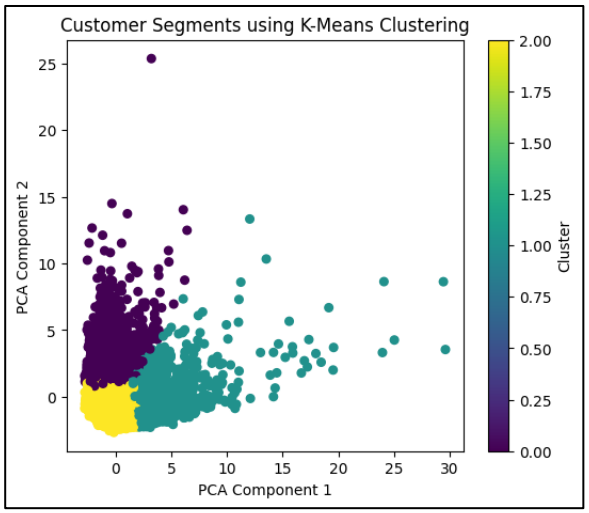
- **Number of Clusters (K):** The value of K is tuned by training K-Means models for K = 2 to 10 and evaluating each configuration using the Silhouette Score. The value that maximizes the silhouette score (K = 3) is selected, as it indicates strong cohesion within clusters and clear separation between clusters.
- **Centroid Initialization and Stability:** The algorithm uses multiple centroid initializations (n_init = 10) with a fixed random state to reduce sensitivity to random initialization and avoid poor local minima. The best clustering outcome is selected automatically based on internal optimization.
- **Feature Scaling:** All features are standardized using StandardScaler prior to clustering. This step is critical to ensure that variables with larger numeric ranges do not dominate distance calculations, leading to biased clusters.

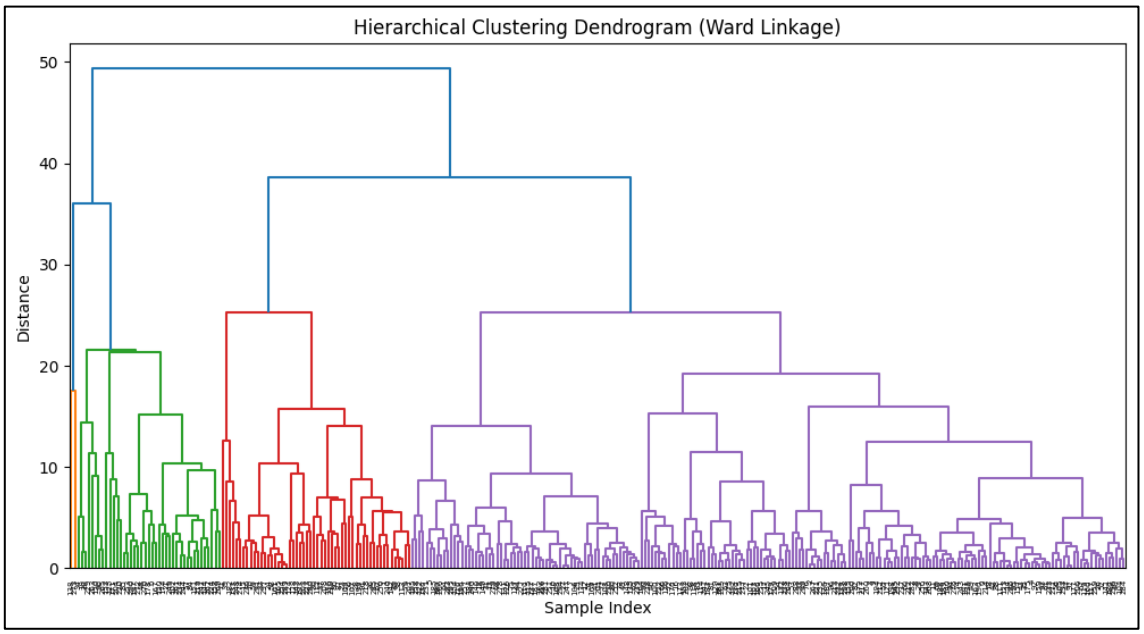
Code:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.preprocessing import
StandardScaler
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score
from sklearn.decomposition import PCA

from scipy.cluster.hierarchy import
dendrogram, linkage
df = pd.read_csv("dataset.csv")
df.columns = df.columns.str.strip()

# Drop ID column
df = df.drop('CUST_ID', axis=1)

# Handle missing values
df = df.fillna(df.mean())

print(df.shape)
print(df.head())
scaler = StandardScaler()
X_scaled = scaler.fit_transform(df)
silhouette_scores = []
K_range = range(2, 11)

for k in K_range:
    kmeans = KMeans(n_clusters=k,
random_state=42, n_init=10)
    labels = kmeans.fit_predict(X_scaled)
    score = silhouette_score(X_scaled, labels)
    silhouette_scores.append(score)
plt.figure(figsize=(6,4))
plt.plot(K_range, silhouette_scores,
marker='o')
plt.xlabel("Number of Clusters (K)")
plt.ylabel("Silhouette Score")
plt.title("Silhouette Score vs Number of
Clusters")
plt.grid(True)
plt.show()
optimal_k =
K_range[np.argmax(silhouette_scores)]
print("Optimal number of clusters:",
optimal_k)

kmeans = KMeans(n_clusters=optimal_k,
random_state=42, n_init=10)
cluster_labels = kmeans.fit_predict(X_scaled)
pca = PCA(n_components=2)
X_pca = pca.fit_transform(X_scaled)
plt.figure(figsize=(6,5))
scatter = plt.scatter(
    X_pca[:, 0],
    X_pca[:, 1],
    c=cluster_labels,
    cmap='viridis',
    s=30
)
plt.xlabel("PCA Component 1")
plt.ylabel("PCA Component 2")
plt.title("Customer Segments using K-Means
Clustering")
plt.colorbar(scatter, label="Cluster")
plt.show()
# Use a subset for dendrogram readability
sample_data = X_scaled[:300]

linked = linkage(sample_data, method='ward')
plt.figure(figsize=(12,6))
dendrogram(linked)
plt.title("Hierarchical Clustering Dendrogram
(Ward Linkage)")
plt.xlabel("Sample Index")
plt.ylabel("Distance")
plt.show()
```

Output:







Conclusion:

- K-Means clustering successfully segmented credit card customers into three distinct groups based on spending, payment behavior, and credit usage patterns.
- Silhouette Score analysis, PCA visualization, and hierarchical clustering consistently validated the stability and separation of the identified clusters.
- The results demonstrate that unsupervised learning can effectively uncover meaningful customer segments, supporting applications such as customer profiling, targeted marketing, and financial strategy design.

HIERARCHICAL (AGGLOMERATIVE) CLUSTERING

Theory:

Hierarchical clustering is an unsupervised learning technique that organizes data into a hierarchy of clusters by progressively combining smaller groups into larger ones. In contrast to partition-based algorithms such as K-Means, hierarchical clustering does not require the number of clusters to be defined beforehand. Instead, it generates a dendrogram, a tree-like diagram that illustrates how individual customers or groups of customers merge at different levels of similarity.

This method is particularly useful for exploratory analysis of credit card customer data, as it reveals layered relationships in spending behavior, payment patterns, and credit utilization. By examining the dendrogram, analysts can choose different cut levels to obtain varying numbers of customer segments, depending on the desired level of detail. Hierarchical clustering relies on a distance measure to quantify similarity between customers and a linkage criterion to determine how distances between clusters are calculated. In this experiment, Euclidean distance is used in combination with Ward's linkage, which aims to minimize variance within clusters.

Given a dataset with n customers and d behavioral features, hierarchical clustering produces a sequence of nested partitions, where each level of the hierarchy represents a different clustering resolution.

$$X = \{x_1, x_2, x_3, \ldots, x_n\}, \quad x_i \in \mathbb{R}^d$$

$$d(x_i, x_j) = \sqrt{\sum_{k=1}^{d} (x_{ik} - x_{jk})^2}$$

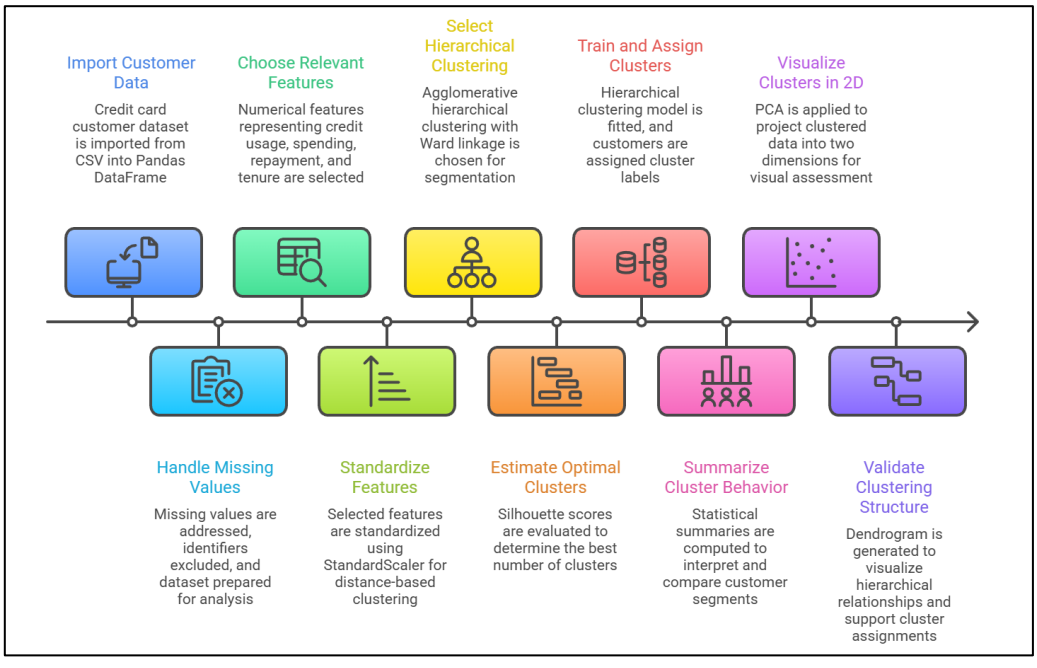
Hierarchical clustering aims to create a nested sequence of partitions: where each level represents a different clustering structure. The most commonly used distance metric is Euclidean distance.

Limitations:

1. **High Computational and Memory Cost:** Hierarchical clustering is computationally intensive, with time complexity increasing rapidly as the number of data points grows. Additionally, it requires storing a full pairwise distance matrix, which makes it unsuitable for very large credit card datasets and limits its practical use to small or moderate-sized samples.
2. **Irreversible Cluster Merging:** Once clusters are merged during the agglomerative process, the decision cannot be reversed. Early incorrect merges caused by noisy or atypical customer behavior may propagate through the hierarchy, negatively affecting the final cluster structure.
3. **Sensitivity to Noise and Outliers:** Extreme values or unusual spending patterns can strongly influence distance calculations and linkage decisions. Such outliers may form isolated clusters or force misleading merges, resulting in distorted dendrograms and less meaningful customer segmentation.
4. **Dependence on Distance Metric and Linkage Choice:** The resulting cluster structure is highly dependent on the selected distance metric and linkage method. Different combinations (e.g., single, complete, average, or Ward linkage) can produce significantly different dendrograms, making hierarchical clustering results sensitive and less stable across configurations.

Workflow:

1. **Data Collection:** The credit card customer dataset is imported from a CSV file into a Pandas DataFrame. It contains customer-level financial and transactional attributes such as balances, purchase behavior, cash advances, payment patterns, and account tenure, which are used for customer segmentation.
2. **Data Cleaning and Preparation:** Missing values are handled to maintain data consistency. Customer identifiers are excluded from analysis, and the dataset is prepared using only behavior-related numerical attributes.
3. **Feature Selection:** Relevant numerical features representing credit usage, spending frequency, repayment behavior, and tenure are selected. Categorical attributes are omitted to ensure compatibility with distance-based clustering methods.
4. **Feature Scaling:** All selected features are standardized using StandardScaler so that differences in variable scales do not bias distance calculations in hierarchical clustering.
5. **Model Selection (Hierarchical Clustering):** Agglomerative hierarchical clustering with Ward linkage is chosen, as it minimizes within-cluster variance and is well suited for numerical credit card data.
6. **Hyperparameter Tuning**: The optimal number of clusters is estimated by evaluating silhouette scores for cluster counts ranging from 2 to 10. The configuration with the highest silhouette score is selected.
7. **Final Model Training and Cluster Assignment:** The hierarchical clustering model is fitted using the selected number of clusters, and each customer is assigned a cluster label that is appended to the dataset.
8. **Cluster Profiling:** Cluster-level statistical summaries, such as mean values of key financial features, are computed to interpret and compare behavioral patterns across customer segments.
9. **PCA Visualization: Principal Component Analysis (PCA)** is applied to reduce dimensionality and project the clustered data into two dimensions, enabling visual assessment of cluster separation.
10. **Dendrogram Validation:** A dendrogram is generated using Ward linkage on a subset of the standardized data to visualize hierarchical relationships and support the selected clustering structure.



Performance Analysis:

Since Hierarchical (Agglomerative) Clustering is an unsupervised learning technique, its performance is evaluated using internal validation metrics and visual inspection, rather than accuracy-based measures. The evaluation focuses on cluster cohesion, separation, and structural consistency using Silhouette Score analysis, PCA visualization, and dendrogram interpretation.

1. **Optimal Number of Clusters Selection (Silhouette Analysis):** Silhouette scores are computed for cluster counts ranging from K = 2 to K = 10 using Ward linkage. The Silhouette Score plot shows that the highest silhouette value is achieved at K = 2, indicating that this configuration provides the strongest balance between intra-cluster compactness and inter-cluster separation. Lower scores for higher values of K suggest reduced clustering quality, confirming K = 2 as the most appropriate number of clusters for the sampled credit card customer data.
2. **Cluster Separation and Visualization (PCA Analysis):** Principal Component Analysis (PCA) is applied to reduce the standardized feature space to two dimensions for visualization. The PCA scatter plot displays two well-defined customer groups, with noticeable separation along the principal components. Although some overlap is expected due to dimensionality reduction, the visualization confirms that hierarchical clustering has identified meaningful behavioral patterns rather than random partitions.
3. **Hierarchical Structure Validation (Dendrogram Analysis):** A dendrogram constructed using Ward linkage on a subset of standardized samples reveals clear hierarchical relationships among customers. The selected horizontal cut at a distance threshold of approximately 25 results in two dominant clusters, which is consistent with the silhouette-based selection. The presence of distinct vertical separations in the dendrogram indicates stable cluster formation and supports the final clustering decision.

Hyperparameter Tuning:

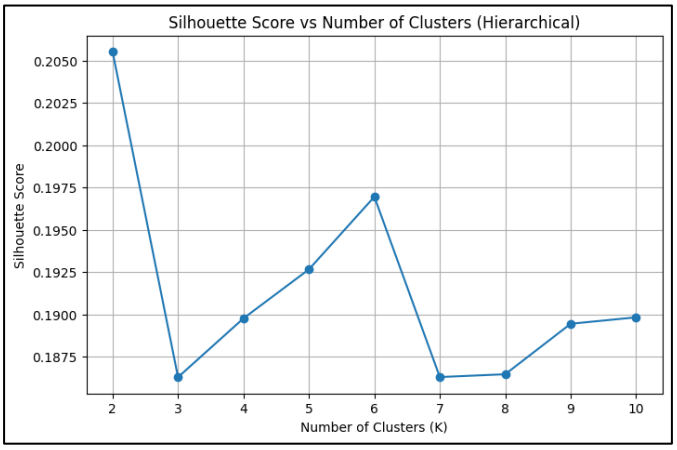
Hierarchical clustering is highly sensitive to hyperparameter choices, particularly the number of clusters and linkage method, both of which directly influence the final cluster structure.

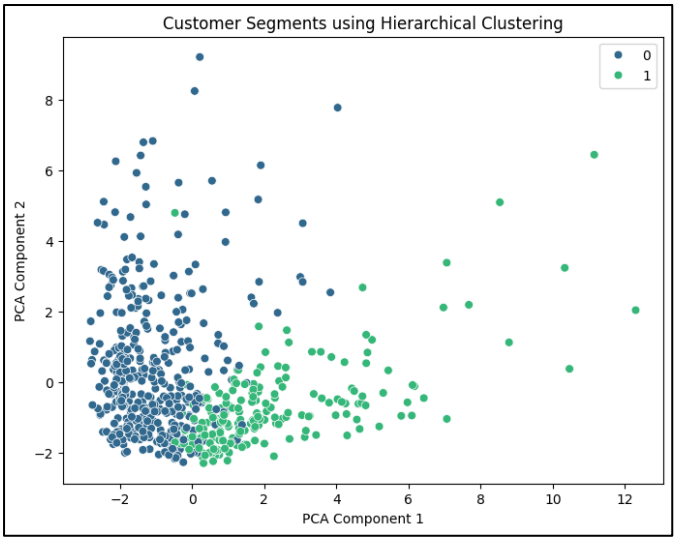
1. **Number of Clusters (K):** The number of clusters is tuned by fitting Agglomerative Clustering models for K values ranging from 2 to 10 and evaluating each configuration using the Silhouette Score. The value K = 2 yields the highest silhouette score and is selected as the optimal configuration, indicating strong cohesion within clusters and clear separation between them.
2. **Linkage Method:** Ward linkage is used throughout the experiment, as it minimizes within-cluster variance during the merging process. This linkage strategy is particularly suitable for numerical, standardized financial data and leads to more compact and interpretable clusters.
3. **Feature Scaling**: All features are standardized using StandardScaler prior to clustering. This step is critical to prevent variables with larger numeric ranges (such as credit limits or payments) from dominating distance calculations.
4. **Sampling for Computational Efficiency:** A random subset of 600 samples is used during model fitting to reduce computational overhead while preserving the underlying data structure. Additional subsampling is applied for dendrogram visualization to maintain clarity without affecting clustering logic.

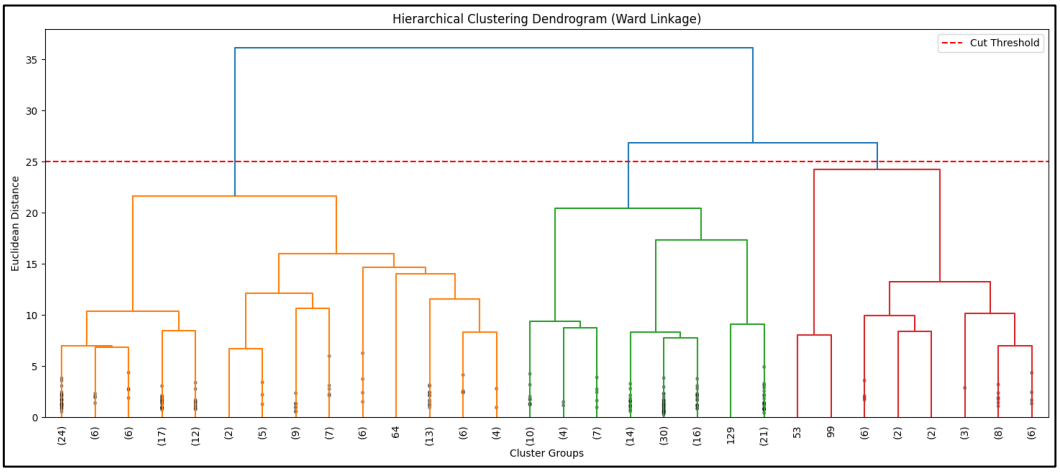
Code:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.preprocessing import
StandardScaler
from sklearn.cluster import
AgglomerativeClustering
from sklearn.metrics import silhouette_score
from sklearn.decomposition import PCA

from scipy.cluster.hierarchy import
dendrogram, linkage

df = pd.read_csv("dataset.csv")

# Drop customer ID
df = df.drop(columns=['CUST_ID'],
errors='ignore')

# Handle missing values
df.fillna(df.mean(), inplace=True)
df_sample = df.sample(n=600,
random_state=42)

scaler = StandardScaler()
X_scaled = scaler.fit_transform(df_sample)

cluster_range = range(2, 11)
silhouette_scores = []
for k in cluster_range:
    model = AgglomerativeClustering(
        n_clusters=k,
        linkage='ward'
    )
    labels = model.fit_predict(X_scaled)
    score = silhouette_score(X_scaled, labels)
    silhouette_scores.append(score)

# Plot Silhouette vs K
plt.figure(figsize=(8,5))
plt.plot(cluster_range, silhouette_scores,
marker='o')
plt.xlabel("Number of Clusters (K)")
plt.ylabel("Silhouette Score")
plt.title("Silhouette Score vs Number of
Clusters (Hierarchical)")
plt.grid(True)
plt.show()

best_k =
cluster_range[np.argmax(silhouette_scores)]
print("Best Number of Clusters:", best_k)
print("Best Silhouette Score:",
round(max(silhouette_scores), 4))

hc = AgglomerativeClustering(
    n_clusters=best_k,
    linkage='ward'
)

final_labels = hc.fit_predict(X_scaled)

pca = PCA(n_components=2)
X_pca = pca.fit_transform(X_scaled)

plt.figure(figsize=(8,6))
sns.scatterplot(
    x=X_pca[:,0],
    y=X_pca[:,1],
    hue=final_labels,
    palette='viridis',
    s=40,
    legend='full'
)
plt.xlabel("PCA Component 1")
plt.ylabel("PCA Component 2")
plt.title("Customer Segments using
Hierarchical Clustering")
plt.show()

linked = linkage(X_scaled[:250],
method='ward')
plt.figure(figsize=(18,7))
dendrogram(
    linked,
    truncate_mode='lastp',
    p=30,
    leaf_rotation=90,
    leaf_font_size=10,
    show_contracted=True
)
plt.axhline(y=25, color='red', linestyle='--',
label='Cut Threshold')
plt.xlabel("Cluster Groups")
plt.ylabel("Euclidean Distance")
plt.title("Hierarchical Clustering Dendrogram
(Ward Linkage)")
plt.legend()
plt.show()
```

Output:







Conclusion:

- Hierarchical (Agglomerative) Clustering successfully segmented credit card customers into two distinct groups based on spending, payment behavior, and credit usage patterns.
- Silhouette Score analysis, PCA visualization, and dendrogram interpretation consistently validated the stability and separation of the identified clusters.
- The experiment demonstrates that hierarchical clustering is an effective tool for exploratory customer segmentation in financial datasets without requiring a predefined number of clusters.