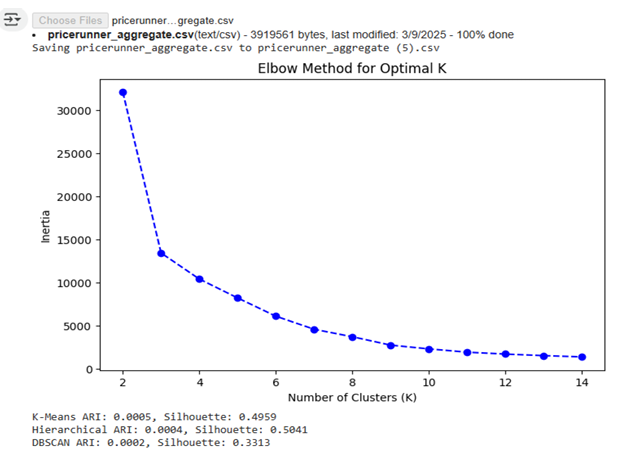
|  |
| --- |
| PRACTICAL - 6 |

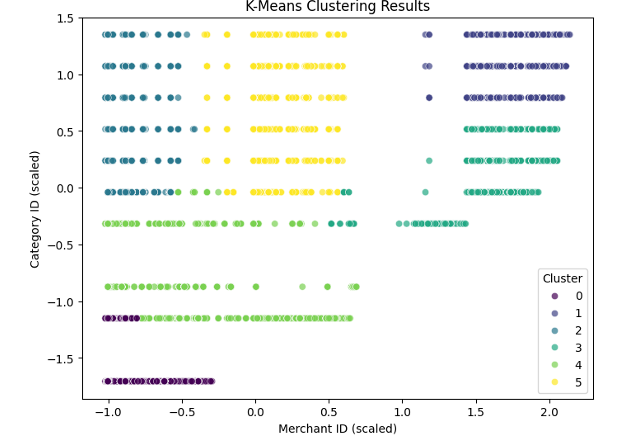
**AIM : Write a program to perform clustering on given product data base using (a) k-mean clustering (b) Hierarchal clustering (c) DBSCAN compare the accuracy of all these three approaches.(**[**https://archive.ics.uci.edu/dataset/837/product+classification+and+clustering**](https://archive.ics.uci.edu/dataset/837/product+classification+and+clustering)**)**

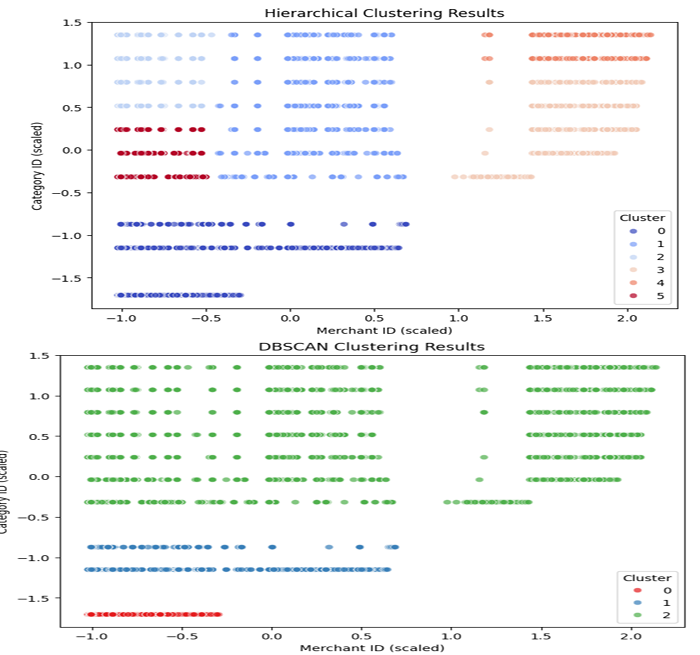
****

|  |
| --- |
| 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, AgglomerativeClustering, DBSCAN  from sklearn.metrics import adjusted\_rand\_score, silhouette\_score    from google.colab import files  uploaded = files.upload()    file\_path = "pricerunner\_aggregate.csv"  df = pd.read\_csv(file\_path)    df.columns = df.columns.str.strip()    X = df[['Merchant ID', 'Category ID']]    scaler = StandardScaler()  X\_scaled = scaler.fit\_transform(X)    true\_labels = df['Cluster ID'].values    inertia = []  k\_values = range(2, 15)    for k in k\_values:      kmeans = KMeans(n\_clusters=k, random\_state=42, n\_init=10)      kmeans.fit(X\_scaled)      inertia.append(kmeans.inertia\_)    plt.figure(figsize=(8, 5))  plt.plot(k\_values, inertia, marker='o', linestyle='--', color='b')  plt.xlabel('Number of Clusters (K)')  plt.ylabel('Inertia')  plt.title('Elbow Method for Optimal K')  plt.show()    optimal\_k = 6    kmeans = KMeans(n\_clusters=optimal\_k, random\_state=42, n\_init=10)  kmeans\_labels = kmeans.fit\_predict(X\_scaled)    hierarchical = AgglomerativeClustering(n\_clusters=optimal\_k)  hierarchical\_labels = hierarchical.fit\_predict(X\_scaled)    dbscan = DBSCAN(eps=0.5, min\_samples=5)  dbscan\_labels = dbscan.fit\_predict(X\_scaled)    ari\_kmeans = adjusted\_rand\_score(true\_labels, kmeans\_labels)  ari\_hierarchical = adjusted\_rand\_score(true\_labels, hierarchical\_labels)  ari\_dbscan = adjusted\_rand\_score(true\_labels, dbscan\_labels)    silhouette\_kmeans = silhouette\_score(X\_scaled, kmeans\_labels)  silhouette\_hierarchical = silhouette\_score(X\_scaled, hierarchical\_labels)    if len(set(dbscan\_labels)) > 1:        silhouette\_dbscan = silhouette\_score(X\_scaled, dbscan\_labels)  else:      silhouette\_dbscan = -1    print(f"K-Means ARI: {ari\_kmeans:.4f}, Silhouette: {silhouette\_kmeans:.4f}")  print(f"Hierarchical ARI: {ari\_hierarchical:.4f}, Silhouette: {silhouette\_hierarchical:.4f}")  print(f"DBSCAN ARI: {ari\_dbscan:.4f}, Silhouette: {silhouette\_dbscan:.4f}")    plt.figure(figsize=(8, 6))  sns.scatterplot(x=X\_scaled[:, 0], y=X\_scaled[:, 1], hue=kmeans\_labels, palette="viridis", alpha=0.7)  plt.title("K-Means Clustering Results")  plt.xlabel("Merchant ID (scaled)")  plt.ylabel("Category ID (scaled)")  plt.legend(title="Cluster", loc="best")  plt.show()    plt.figure(figsize=(8, 6))  sns.scatterplot(x=X\_scaled[:, 0], y=X\_scaled[:, 1], hue=hierarchical\_labels, palette="coolwarm", alpha=0.7)  plt.title("Hierarchical Clustering Results")  plt.xlabel("Merchant ID (scaled)")  plt.ylabel("Category ID (scaled)")  plt.legend(title="Cluster", loc="best")  plt.show()    plt.figure(figsize=(8, 6))  sns.scatterplot(x=X\_scaled[:, 0], y=X\_scaled[:, 1], hue=dbscan\_labels, palette="Set1", alpha=0.7)  plt.title("DBSCAN Clustering Results")  plt.xlabel("Merchant ID (scaled)")  plt.ylabel("Category ID (scaled)")  plt.legend(title="Cluster", loc="best")  plt.show()  accuracy\_df = pd.DataFrame({      "Clustering Algorithm": ["K-Means", "Hierarchical", "DBSCAN"],      "Adjusted Rand Index (ARI)": [ari\_kmeans, ari\_hierarchical, ari\_dbscan],      "Silhouette Score": [silhouette\_kmeans, silhouette\_hierarchical, silhouette\_dbscan]  })    print("\n### Clustering Performance Comparison ###\n")  print(accuracy\_df)    plt.figure(figsize=(10, 5))  accuracy\_df.set\_index("Clustering Algorithm").plot(kind="bar", colormap="viridis", figsize=(10, 5))  plt.title("Comparison of Clustering Methods")  plt.ylabel("Score")  plt.xticks(rotation=0)  plt.legend(title="Metrics")  plt.show()    best\_ari = accuracy\_df.iloc[accuracy\_df['Adjusted Rand Index (ARI)'].idxmax()]  best\_silhouette = accuracy\_df.iloc[accuracy\_df['Silhouette Score'].idxmax()]    print("\n### Best Clustering Approach Based on ARI ###")  print(f"{best\_ari['Clustering Algorithm']} with ARI: {best\_ari['Adjusted Rand Index (ARI)']:.4f}")    print("\n### Best Clustering Approach Based on Silhouette Score ###")  print(f"{best\_silhouette['Clustering Algorithm']} with Silhouette Score: {best\_silhouette['Silhouette Score']:.4f}") |

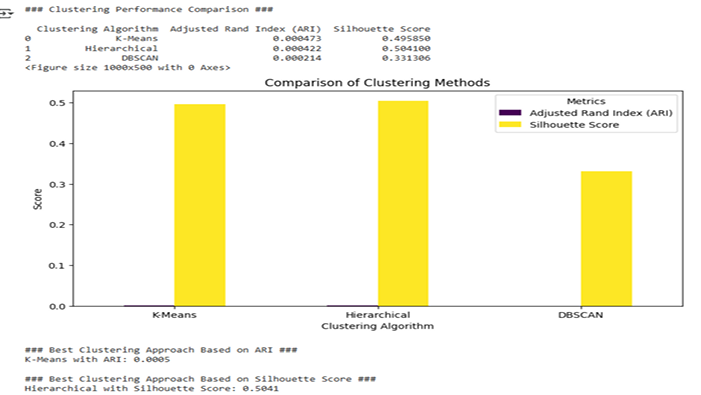
OUTPUT :







6

****