NAME: TANU DAHIYA
POSITION: DATA SCIENCE INTERN

TASK 3:

Task 3: 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.

1.
CODE:

```
# Import Required Libraries
import pandas as pd
import numpy as np
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt
import seaborn as sns

# Data URLs
```

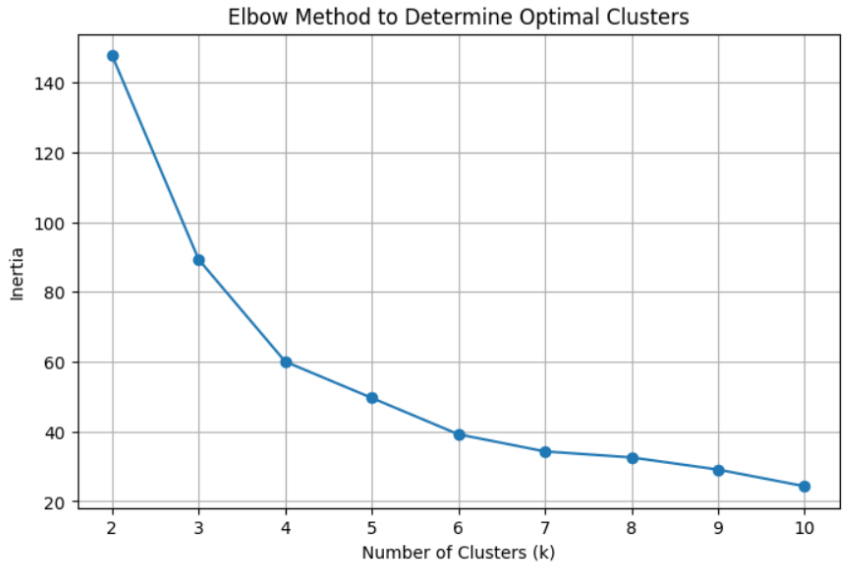
```
CUSTOMERS_URL = 'https://drive.google.com/uc?id=1bu_--mo79VdUG9oin4ybfFGRUSXAe-WE'
TRANSACTIONS_URL = 'https://drive.google.com/uc?id=1saEqdbBB-vuk2hxoAf4TzDEsykdKlzbF'

# Load Data
customers = pd.read_csv(CUSTOMERS_URL)
transactions = pd.read_csv(TRANSACTIONS_URL)

# Inspect Columns in Datasets
print("Customers Columns:", customers.columns.tolist())
print("Transactions Columns:", transactions.columns.tolist())

# Aggregate Transaction Data
transaction_summary = transactions.groupby('CustomerID').agg({
    'TotalValue': 'sum',
    'Quantity': 'sum'
}).reset_index()

# Merge with Customer Data
customer_data = customers.merge(transaction_summary, on='CustomerID',
how='inner')

# Define Features for Clustering
required_features = ['Age', 'Income', 'TotalValue', 'Quantity']
available_features = customer_data.columns.intersection(required_features)

if available_features.empty:
    raise ValueError("No valid features available for clustering!")

print(f"Using Features for Clustering: {available_features.tolist()}")

# Feature Selection
features = customer_data[available_features]

# Standardize Features
scaler = StandardScaler()
scaled_features = scaler.fit_transform(features)

# Determine Optimal Clusters Using Elbow Method
inertia = []
k_range = range(2, 11)
```

```
for k in k_range:
    kmeans = KMeans(n_clusters=k, random_state=42)
    kmeans.fit(scaled_features)
    inertia.append(kmeans.inertia_)

# Plot Elbow Curve
plt.figure(figsize=(8, 5))
plt.plot(k_range, inertia, marker='o')
plt.title('Elbow Method to Determine Optimal Clusters')
plt.xlabel('Number of Clusters (k)')
plt.ylabel('Inertia')
plt.grid()
plt.show()

# Apply KMeans Clustering
optimal_k = 4  # Replace this value based on the elbow plot
kmeans = KMeans(n_clusters=optimal_k, random_state=42)
customer_data['Cluster'] = kmeans.fit_predict(scaled_features)

# Save Clustering Results
customer_data[['CustomerID', 'Cluster']].to_csv('Customer_Segments.csv',
index=False)
print("Customer_Segments.csv created successfully!")

# Visualize Clusters Using PCA
pca = PCA(n_components=2)
pca_features = pca.fit_transform(scaled_features)
customer_data['PCA1'] = pca_features[:, 0]
customer_data['PCA2'] = pca_features[:, 1]

plt.figure(figsize=(10, 6))
sns.scatterplot(
    data=customer_data,
    x='PCA1',
    y='PCA2',
    hue='Cluster',
    palette='viridis',
    s=100
)
plt.title('Customer Segmentation Visualization')
plt.xlabel('PCA Component 1')
plt.ylabel('PCA Component 2')
plt.legend(title='Cluster')
```

```
plt.grid()
plt.tight_layout()
plt.show()

# Cluster Summary
cluster_summary = customer_data.groupby('Cluster').agg(
    {col: 'mean' for col in available_features}
).reset_index()

print("\nCluster Summary:")
print(cluster_summary)

# Save Cluster Summary
cluster_summary.to_csv("Cluster_Summary.csv", index=False)
print("Cluster_Summary.csv created successfully!")
```

OUTPUT:

```
Customers Columns: ['CustomerID', 'CustomerName', 'Region', 'SignupDate']
Transactions Columns: ['TransactionID', 'CustomerID', 'ProductID', 'TransactionDate', 'Quantity', 'TotalValue', 'Price']
Using Features for Clustering: ['TotalValue', 'Quantity']
```




```
Customer_Segments.csv created successfully!
```


```
Cluster Summary:
   Cluster   TotalValue   Quantity
0        0  6780.821905  24.190476
1        1  2967.617753  11.258427
2        2  4942.563191  17.255319
3        3  1218.567857   5.142857
Cluster_Summary.csv created successfully!
```