**Lab Assignment 4: Data Analysis**

You are working for a large e-commerce platform, and your task is to perform customer segmentation based on their shopping behaviour. You have access to a dataset containing information about customer transactions. The dataset is in a CSV format and contains the following columns:

1. Customer ID: Unique identifier for each customer.
2. Total Amount Spent: The total amount spent by each customer on the platform.
3. Total Items Purchased: The total number of items purchased by each customer.
4. Last Purchase Date: The date of the customer's most recent purchase.
5. Average Purchase Value: The average value of each customer's purchases.

Using NumPy and Pandas, your goal is to perform the following tasks:

1. Data Loading: Load the dataset into a Pandas DataFrame for analysis.
2. Data Cleaning: Check for missing values, duplicates, or any inconsistencies in the data. If found, clean the data appropriately.
3. Descriptive Statistics: Calculate basic statistics such as mean, median, and standard deviation of TotalAmountSpent and TotalItemsPurchased.
4. Customer Segmentation: Divide the customers into segments based on their shopping behaviour. You can use techniques like K-means clustering or any other method you prefer. For example, you might create segments like "High Spenders," "Frequent Shoppers," and "Inactive Customers."
5. Visualization: Create visualizations (e.g., scatter plots, bar charts) to represent the different customer segments you've identified.
6. Customer Insights: Provide insights into each customer segment. What distinguishes one segment from another? How can the e-commerce platform tailor its marketing strategies for each segment?
7. Customer Engagement Recommendations: Based on your analysis, provide recommendations for the ecommerce platform on how to engage with each customer segment more effectively. For example, should they offer discounts, provide personalized product recommendations, or run targeted marketing campaigns?

This problem requires you to use Pandas for data manipulation, NumPy for numerical operations, and potentially machine learning libraries for customer segmentation. It showcases the power of data analysis and segmentation for making data-driven decisions in e-commerce.

Code –

import pandas as pd

import numpy as np

from sklearn.cluster import KMeans

import matplotlib.pyplot as plt

import seaborn as sns

# 1. Data Loading

try:

    df = pd.read\_csv("D:/5th Lab/Python/Lab 4/Online Retail.csv", encoding='ISO-8859-1')

except FileNotFoundError:

    print("The file 'Online Retail.csv' was not found. Please check the file path.")

    raise

# 2. Data Cleaning

df.dropna(subset=['CustomerID'], inplace=True)  # Drop rows with missing CustomerID

df.drop\_duplicates(inplace=True)  # Remove duplicates

# Add a 'TotalAmountSpent' column (Quantity \* UnitPrice)

df['TotalAmountSpent'] = df['Quantity'] \* df['UnitPrice']

# 3. Feature Engineering

df['InvoiceDate'] = pd.to\_datetime(df['InvoiceDate'], errors='coerce')  # Convert InvoiceDate to datetime format

# Calculate Recency (days since the last purchase)

df['Recency'] = (pd.to\_datetime('today') - df['InvoiceDate']).dt.days

# Calculate Total Amount Spent and Total Items Purchased by each customer

customer\_data = df.groupby('CustomerID').agg({

    'TotalAmountSpent': 'sum',

    'Quantity': 'sum',

    'Recency': 'min'  # Use 'min' to get the most recent purchase

}).reset\_index()

customer\_data.rename(columns={'Quantity': 'TotalItemsPurchased'}, inplace=True)

# Calculate Average Purchase Value

customer\_data['AveragePurchaseValue'] = customer\_data['TotalAmountSpent'] / customer\_data['TotalItemsPurchased']

# 4. Handle Missing or Infinite Values

customer\_data.replace([np.inf, -np.inf], np.nan, inplace=True)

customer\_data.dropna(inplace=True)

# 5. Descriptive Statistics

print("Descriptive Statistics for TotalAmountSpent:")

print(customer\_data['TotalAmountSpent'].describe())

print("\nDescriptive Statistics for TotalItemsPurchased:")

print(customer\_data['TotalItemsPurchased'].describe())

# 6. Customer Segmentation using K-means Clustering

X = customer\_data[['TotalAmountSpent', 'TotalItemsPurchased', 'AveragePurchaseValue', 'Recency']]

# Apply K-means clustering

kmeans = KMeans(n\_clusters=4, random\_state=42)  # Specify n\_init to avoid warnings

customer\_data['Segment'] = kmeans.fit\_predict(X)

# 7. Visualization

plt.figure(figsize=(10, 6))

sns.scatterplot(x='TotalAmountSpent', y='TotalItemsPurchased', hue='Segment', data=customer\_data, palette='viridis')

plt.title('Customer Segmentation based on Total Amount Spent and Total Items Purchased')

plt.show()

# 8. Customer Insights

insights = customer\_data.groupby('Segment').mean().reset\_index()

print("Customer Segment Insights:")

print(insights)

# 9. Customer Engagement Recommendations

for segment in insights['Segment']:

    print(f"\nSegment {segment} Recommendations:")

    if insights.loc[segment, 'TotalAmountSpent'] > customer\_data['TotalAmountSpent'].mean():

        print("- Targeted promotions and discounts to retain high spenders.")

    if insights.loc[segment, 'Recency'] > customer\_data['Recency'].mean():

        print("- Re-engagement campaigns for inactive customers.")

    if insights.loc[segment, 'TotalItemsPurchased'] > customer\_data['TotalItemsPurchased'].mean():

        print("- Loyalty programs to reward frequent shoppers.")

* Output –





