Data Science Assignment: eCommerce Transactions Dataset

Assignment Tasks:

Task 1: Exploratory Data Analysis (EDA) and Business Insights

Step 1: Perform EDA

1. ****Data Loading**: Import the datasets into a pandas DataFrame.**

```
# Step 1: Upload the dataset in Google Colab from google.colab
import files # Import the 'files' module from google.colab
 print("Please upload the CSV file containing the
dataset.") uploaded = files.upload() OR
df = pd.read_csv("……..")
```

2Data Inspection**: Check for missing values, data types, and basic statistics**

```
# Check for missing values
customers.isnull().sum()
products.isnull().sum()
transactions.isnull().sum()

# Summary statistics for numerical columns
transactions.describe()
```

2. ****Data Cleaning**:**

- Handle missing data.
- Convert date columns into datetime format (e.g., `SignupDate` and `TransactionDate`).
- Ensure that the `Price` column in the Products dataset and the `TotalValue` column in Transactions dataset are consistent

 # Convert date columns

 # Fill missing data (if any)

```
# Fill missing data (if any)
Customers.fillna('Unknown', inplace=True)

# Read the 'Products.csv' file into a DataFrame called 'Products'
```

```
Products = pd.read_csv("Products.csv", sep=",", on_bad_lines='skip')

# Select only numeric columns for filling missing values
import numpy as np
numeric_columns = Products.select_dtypes(include=np.number).columns
Products[numeric_columns] =
Products[numeric_columns].fillna(Products[numeric_columns].mean())
```

3. ****Visualizations**:**

- Create visualizations like histograms, box plots, and bar charts to understand the data distribution.

- Visualize the distribution of prices, customer regions, and the number of transactions per customer.

 # Distribution of product prices

 # Region-wise distribution customers

 # Number of transaction percustomer

```
import matplotlib.pyplot as plt
import seaborn as sns

# Distribution of product prices
plt.figure(figsize=(10, 6))
sns.histplot(products['Price'], bins=30, kde=True)
plt.title('Product Price Distribution')
plt.show()

# Region-wise distribution of customers
plt.figure(figsize=(10, 6))
sns.countplot(data=customers, x='Region')
plt.title('Customer Distribution by Region')
plt.show()

# Number of transactions per customer
transaction_counts = transactions['CustomerID'].value_counts()
plt.figure(figsize=(10, 6))
sns.histplot(transaction_counts, bins=30)
plt.title('Transactions per Customer')
plt.show()
```

```

## Step 2: Business Insights

**From the EDA, derive business insights:**

**1.Customer Distribution by Region: If one region has significantly more customers, marketing campaigns could be focused more on other underrepresented regions.**
**2.Top-selling Products : Identify which products are the most popular and profitable, helping inventory and sales planning.**
**3.Transaction Frequency : Customers with high transaction frequency might be targeted for loyalty programs or special offers.**
**4.Price Sensitivity : If a segment of customers mostly buys low-cost items, this could suggest a price-sensitive group.**
**5.Signup Trends: If signup dates show seasonal patterns, promotions can be tailored to coincide with these periods.**

## ### Task 2: Lookalike Model

**Step 1: Feature Engineering 1.**

**Customer Profile Features**:

- Merge the `Customers` and `Transactions` datasets on `CustomerID` to get transaction history for each customer.
- Extract features like total spending, number of transactions, and average transaction value.

**2. \*\*Product Profile Features\*\*:**

- Group the `Transactions` dataset by `ProductID` to capture product-specific features total sales and average price.
- Merge these features into the `Transactions` data.

```
# Merge with transactions
transactions = pd.merge(transactions, product_data, on='ProductID')
```

Use a similarity measure such as cosine similarity or Euclidean distance to calculate

**1. \*\*Calculate similarity score\*\*:**

- For each pair of customers, compute similarity using relevant features
```

```
# Select relevant features for similarity calculation

# Calculate similarity matrix

# For each customer, find top 3 similar customers
```