**Steps to Follow:**

**1. Data Preparation**

- Download the datasets from the provided links and load them into your working environment.
- Check for missing values, duplicates, or anomalies in the data.

**2. Exploratory Data Analysis (EDA)**

Perform the following analysis:

- **Customer Analysis**:
    - Count of customers by region.
    - Customer signup trends (e.g., by year or month).
- **Product Analysis**:
    - Distribution of products by category.
    - Pricing trends (average, min, and max prices).
- **Transaction Analysis**:
    - Distribution of transactions over time.
    - Most frequently sold products.
    - Average transaction value.
- **Combined Analysis**:
    - Region-wise product preferences.
    - High-value customers (top spenders).

**3. Insights**

Extract actionable business insights based on the analysis:

1. **Top-Selling Products and Categories**: Identify the best-performing product categories to focus marketing efforts.
2. **Customer Segmentation**: Classify customers based on spending habits and region for targeted campaigns.
3. **Signup Trends**: Analyze when customers are more likely to sign up to plan promotional events.
4. **Seasonal Sales Trends**: Spot seasonal spikes in transactions for inventory planning.
5. **High-Value Customers**: Highlight loyal customers contributing significantly to revenue.

**4. Predictive Modeling**

- Build models to predict outcomes, e.g.:

    - Customer churn.
    - Likelihood of purchase for a product.
    - Forecasting sales for the next month.
- Use machine learning models like:
    - Logistic Regression, Random Forest, or XGBoost for classification.
    - Time series models (ARIMA) for forecasting.

**5. Deliverables Preparation**

- **Python Script/Jupyter Notebook**: Include:
    - Code for loading and cleaning data.
    - Visualizations and analysis.
    - Comments explaining the process.
- **Business Insights PDF**: Format:
    - Title: **EDA and Business Insights for eCommerce Transactions**
    - Introduction (brief overview of the dataset and tasks).
    - Insights with graphs or charts where applicable (screenshots or images).
    - Recommendations based on insights.

```
# Import necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from datetime import datetime

# Load the datasets
customers = pd.read_csv("Customers.csv")
products = pd.read_csv("Products.csv")
transactions = pd.read_csv("Transactions.csv")
```

```
# Display basic information about datasets
print("Customers Dataset:")
print(customers.info())
print(customers.head())

print("\nProducts Dataset:")
print(products.info())
print(products.head())

print("\nTransactions Dataset:")
print(transactions.info())
print(transactions.head())

# Task 1: Data Cleaning and Preparation
# Check for missing values
print("\nMissing values in datasets:")
print("Customers:", customers.isnull().sum())
print("Products:", products.isnull().sum())
print("Transactions:", transactions.isnull().sum())

# Convert date columns to datetime
customers['SignupDate'] = pd.to_datetime(customers['SignupDate'])
transactions['TransactionDate'] = pd.to_datetime(transactions['TransactionDate'])

# Check for duplicates
print("\nDuplicates in datasets:")
print("Customers:", customers.duplicated().sum())
print("Products:", products.duplicated().sum())
print("Transactions:", transactions.duplicated().sum())

# Task 2: Exploratory Data Analysis (EDA)
# Customer Analysis
print("\nCustomer Count by Region:")
region_counts = customers['Region'].value_counts()
print(region_counts)

plt.figure(figsize=(8, 5))
sns.barplot(x=region_counts.index, y=region_counts.values, palette="viridis")
plt.title("Customer Count by Region")
plt.xlabel("Region")
plt.ylabel("Count")
plt.show()

# Signup trends over time
customers['SignupYear'] = customers['SignupDate'].dt.year
signup_trends = customers['SignupYear'].value_counts().sort_index()

plt.figure(figsize=(8, 5))
sns.lineplot(x=signup_trends.index, y=signup_trends.values, marker="o")
plt.title("Customer Signup Trends Over Years")
plt.xlabel("Year")
plt.ylabel("Number of Signups")
plt.show()

# Product Analysis
print("\nProducts by Category:")
category_counts = products['Category'].value_counts()
print(category_counts)

plt.figure(figsize=(8, 5))
sns.barplot(x=category_counts.index, y=category_counts.values, palette="coolwarm")
plt.title("Product Count by Category")
plt.xlabel("Category")
plt.ylabel("Count")
plt.show()

# Pricing analysis
print("\nProduct Price Statistics:")
print(products['Price'].describe())

plt.figure(figsize=(8, 5))
sns.histplot(products['Price'], bins=20, kde=True, color="blue")
plt.title("Distribution of Product Prices")
plt.xlabel("Price (USD)")
plt.ylabel("Frequency")
plt.show()

# Transaction Analysis
print("\nTransaction Value Statistics:")
print(transactions['TotalValue'].describe())

plt.figure(figsize=(8, 5))
sns.histplot(transactions['TotalValue'], bins=20, kde=True, color="green")
plt.title("Distribution of Transaction Values")
plt.xlabel("Total Value (USD)")
plt.ylabel("Frequency")
plt.show()

# Transactions over time
transactions['TransactionMonth'] = transactions['TransactionDate'].dt.to_period('M')
monthly_sales = transactions.groupby('TransactionMonth')['TotalValue'].sum()

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

```
monthly_sales.plot(kind='line', marker='o', color='purple')

plt.title("Monthly Sales Trends")

plt.xlabel("Month")

plt.ylabel("Total Sales (USD)")

plt.grid(True)

plt.show()


# Top-selling products

top_products =
transactions.groupby('ProductID')['Quantity'].sum().sort_values(ascending=False).head(10)

top_product_names = products[products['ProductID'].isin(top_products.index)]


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

sns.barplot(x=top_products.values, y=top_product_names['ProductName'], palette="magma")

plt.title("Top-Selling Products")

plt.xlabel("Total Quantity Sold")

plt.ylabel("Product Name")

plt.show()


# Task 3: Insights

# Combine datasets for region-product analysis

combined_data = transactions.merge(customers, on='CustomerID').merge(products, on='ProductID')


# Region-wise preferences

region_preferences = combined_data.groupby('Region')['TotalValue'].sum()


plt.figure(figsize=(8, 5))

region_preferences.plot(kind='bar', color='teal')

plt.title("Total Sales by Region")

plt.xlabel("Region")

plt.ylabel("Total Sales (USD)")
plt.show()


# Customer segmentation (top spenders)
top_spenders =
combined_data.groupby('CustomerID')['TotalValue'].sum().sort_values(ascending=False).head(10)
top_spender_names = customers[customers['CustomerID'].isin(top_spenders.index)]


plt.figure(figsize=(10, 6))
sns.barplot(x=top_spenders.values, y=top_spender_names['CustomerName'], palette="plasma")
plt.title("Top Spending Customers")
plt.xlabel("Total Spend (USD)")
plt.ylabel("Customer Name")
plt.show()


# Save processed datasets if needed
customers.to_csv("Processed_Customers.csv", index=False)
products.to_csv("Processed_Products.csv", index=False)
transactions.to_csv("Processed_Transactions.csv", index=False)
```