Business Data Analytics

Project

BY:

Abdulrahman Abuhani (2132462)
Belal Khaled (2136873)

Table Of Contents

Topics: **Page:**

Introduction:

In this report, we delve into the comprehensive process of analyzing a dataset, starting from: **Data Acquisition** to **Data Cleaning, Exploratory Analysis, Analysis Model** then **Visualization** and Finally **Prescriptive.**

We'll follow the five main steps of the Data Science process:

Step 1 **(Data Acquisition):** **We'll begin by loading the dataset into a Pandas DataFrame.**

Step 2 **(Data Cleaning):** **This step involves handling missing values, correcting data types and removing duplicates.**

Step 3 **(Exploratory Analysis):** **Conducting exploratory data analysis to understand the data distribution, relationships, key statistics and Examine the data to find trends and patterns.**

Step 4 **(Analysis Model):** **Building a simple analysis model, such as a linear regression to predict sales based on other variables.**

Step 5 **(Visualization):** **Visualizing the results of the analysis model.**

Prescriptive: **providing recommendations or guidelines on what actions to take to achieve a desired outcome.**

➢ *Importing all libraries we need:*

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.preprocessing import OneHotEncoder
```

STEP 1 (Data Acquisition):

1.1 Load the dataset:

** We'll begin by loading the dataset into a Pandas DataFrame.*

- ***Code:***

```
dataset = pd.read_csv('shopping_trends_updated.csv')
dataset
#The data contains 3900 `record` and 18 `attribute`, let's explore their types.

dataset.info()
```

➢ *Output for 1.1 (Data Types):*

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3900 entries, 0 to 3899
Data columns (total 18 columns):
```

#	Column	Non-Null Count	Dtype
0	Customer ID	3900 non-null	int64
1	Age	3900 non-null	int64
2	Gender	3900 non-null	object
3	Item Purchased	3900 non-null	object
4	Category	3900 non-null	object
5	Purchase Amount (USD)	3900 non-null	int64
6	Location	3900 non-null	object
7	Size	3900 non-null	object
8	Color	3900 non-null	object
9	Season	3900 non-null	object
10	Review Rating	3900 non-null	float64
11	Subscription Status	3900 non-null	object
12	Shipping Type	3900 non-null	object
13	Discount Applied	3900 non-null	object
14	Promo Code Used	3900 non-null	object
15	Previous Purchases	3900 non-null	int64
16	Payment Method	3900 non-null	object
17	Frequency of Purchases	3900 non-null	object

```
dtypes: float64(1), int64(4), object(13)
memory usage: 548.6+ KB
```

The object data type usually indicates string data or mixed types (numeric and non-numeric data), while int64 and float64 represent integer and floating-point numbers, respectively.

STEP 2(Data Cleaning):

2.1 Check For Missing Values:

- ***Code:***

Output:

```
dataset.describe(include=object)
```

	Gender	Item Purchased	Category	Location	Size	Color	Season	Subscription Status	Shipping Type	Discount Applied	Promo Code Used	Payment Method	Frequency of Purchases
count	3900	3900	3900	3900	3900	3900	3900	3900	3900	3900	3900	3900	3900
unique	2	25	4	50	4	25	4	2	6	2	2	6	7
top	Male	Blouse	Clothing	Montana	M	Olive	Spring	No	Free Shipping	No	No	PayPal	Every 3 Months
freq	2652	171	1737	96	1755	177	999	2847	675	2223	2223	677	584

```
null_counts = dataset.isnull().sum()
null_counts
```

➢ *Output for 2.1(Check For Missing Values):*

Attribute:	Null counts:
Customer ID	0
Age	0
Gender	0
Item Purchased	0
Category	0
Purchase Amount (USD)	0
Location	0
Size	0
Color	0
Season	0
Review Rating	0
Subscription Status	0
Shipping Type	0
Discount Applied	0
Promo Code Used	0
Previous Purchases	0
Payment Method	0
Frequency of Purchases	0

dtype: int64

STEP 2(Data Cleaning):

2.2 Check For Duplicates:

- ***Code:***

```
duplicates = dataset.duplicated().sum()
duplicates
```

➢ *Output for 2.2(Check For Duplicates):*

➔ **0**

- ***We have (0) Missing values and (0) Duplicates.***

STEP 3(Exploratory Analysis):

3.1 Basic statistics:

➢ ***Conducting exploratory data analysis to understand the data distribution, relationships, and key statistics (Descriptive).***

- ***Code:***

```
summary_stats = dataset.describe()
summary_stats
```

Output:

	Customer ID	Age	Purchase Amount (USD)	Review Rating	Previous Purchases
count	3900.000000	3900.000000	3900.000000	3900.000000	3900.000000
mean	1950.500000	44.068462	59.764359	3.749949	25.351538
std	1125.977353	15.207589	23.685392	0.716223	14.447125
min	1.000000	18.000000	20.000000	2.500000	1.000000
25%	975.750000	31.000000	39.000000	3.100000	13.000000
50%	1950.500000	44.000000	60.000000	3.700000	25.000000
75%	2925.250000	57.000000	81.000000	4.400000	38.000000
max	3900.000000	70.000000	100.000000	5.000000	50.000000

STEP 3(Exploratory Analysis):

3.2 Provided columns:

➢ ***The dataset contains information about various purchases made by customers. Here are the columns provided:***

1. **Customer ID: Unique identifier for each customer.**
2. **Age: Age of the customer.**
3. **Gender: Gender of the customer.**
4. **Item Purchased: Description of the item.**
5. **Category: Category of the item (e.g., Clothing, Footwear).**
6. **Purchase Amount (USD): Amount spent on the purchase.**
7. **Location: Customer's location.**
8. **Size: Size of the item purchased.**
9. **Color: Color of the item.**
10. **Season: Season when the item was purchased.**
11. **Review Rating: Rating given by the customer.**
12. **Subscription Status: Whether the customer has a subscription.**
13. **Shipping Type: Type of shipping.**
14. **Discount Applied: Whether a discount was applied.**
15. **Promo Code Used: Whether a promo code was used.**
16. **Previous Purchases: Number of previous purchases.**
17. **Payment Method: Payment method used.**
18. **Frequency of Purchases: Frequency of purchases.**

STEP 3(Exploratory Analysis):

3.3 Plot distribution:

➢ ***We will start with these analyses to uncover trends and patterns in the data. Let's create some visualizations to help with this exploration.***

- ## *Code 1.1:*

```
#Plot distribution of purchase amounts:
plt.figure(figsize=(8, 6))
sns.histplot(dataset['Purchase Amount (USD)'], bins=30, color='blue')
plt.title('Distribution of Purchase Amounts')
plt.xlabel('Purchase Amount (USD)')
plt.ylabel('Frequency')
plt.grid(True)
plt.show()
```

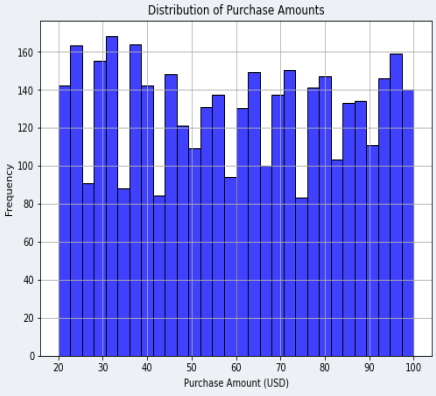
- ## *Code 1.2:*

```
#Plot distribution of Purchase Amount Distribution by Category:
plt.figure(figsize=(10, 5))
sns.boxplot(data=dataset, x='Category', y='Purchase Amount (USD)', palette='Set3')
plt.title('Purchase Amount Distribution by Category')
plt.xlabel('Category')
plt.ylabel('Purchase Amount (USD)')
plt.xticks(rotation=45)
plt.show()
```

- ## *Code 1.3:*

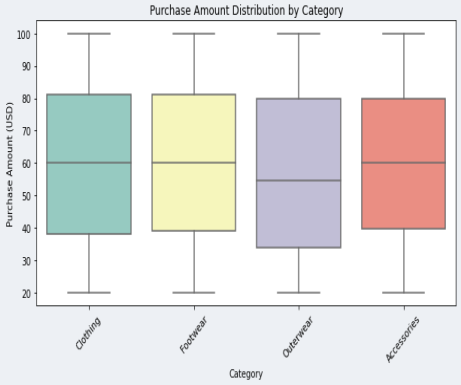
```
#Plot distribution of previous purchases:
plt.figure(figsize=(8, 6))
sns.histplot(dataset['Previous Purchases'], bins=30, color='green')
plt.title('Distribution of Previous Purchases')
plt.xlabel('Number of Previous Purchases')
plt.ylabel('Frequency')
plt.grid(True)
plt.show()
```

➢ ***Output for 1.1:***



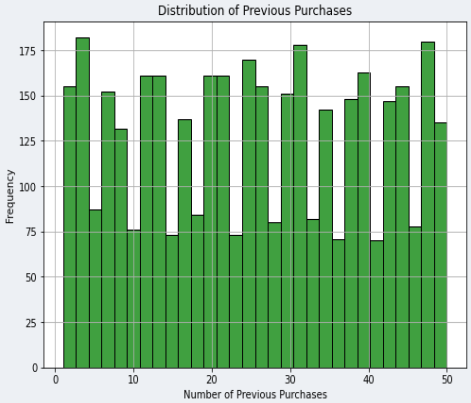
Plot distribution of purchase amounts.

Output for 1.2:



Plot distribution of Purchase Amount Distribution by Category.

Output for 1.3:



Plot distribution of previous purchases.

STEP 3(Exploratory Analysis):

3.4 Count plot:

- ## *Code 1.1:*

```
#Count plot for gender distribution:
plt.figure(figsize=(8, 6))
sns.countplot(x='Gender', data=dataset, palette='coolwarm')
plt.title('Gender Distribution')
plt.xlabel('Gender')
plt.ylabel('Count')
plt.grid(True)
plt.show()
```

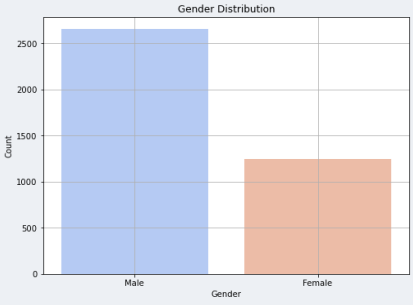
- ## *Code 1.2:*

```
#Count plot for Frequency of Purchases by Subscription Status distribution:
plt.figure(figsize=(10, 5))
sns.countplot(data=dataset, x='Frequency of Purchases', hue='Subscription Status', palette='Set2')
plt.title('Frequency of Purchases by Subscription Status')
plt.xlabel('Frequency of Purchases')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.show()
```

- ## *Code 1.3:*

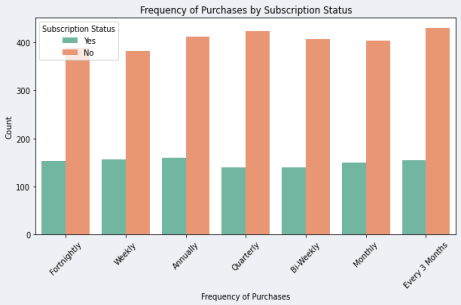
```
#Count plot for top categories:
plt.figure(figsize=(8, 6))
sns.countplot(y='Category', data=dataset, palette='autumn')
plt.title('Top Categories of Purchases')
plt.xlabel('Count')
plt.ylabel('Category')
plt.grid(True)
plt.show()
```

➢ ***Output for 1.1:***



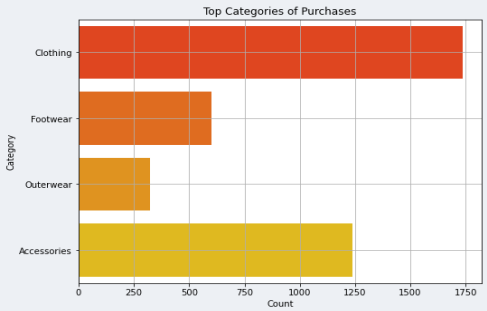
Count plot for gender distribution.

Output for 1.2:



Count plot for Frequency of Purchases by Subscription Status distribution.

Output for 1.3:



Count plot for top categories.

STEP 3(Exploratory Analysis):

3.5 Correlation matrix:

- ***Code 1.1:***

```
#Correlation matrix:
correlation_matrix = dataset.Corr()
correlation_matrix
```

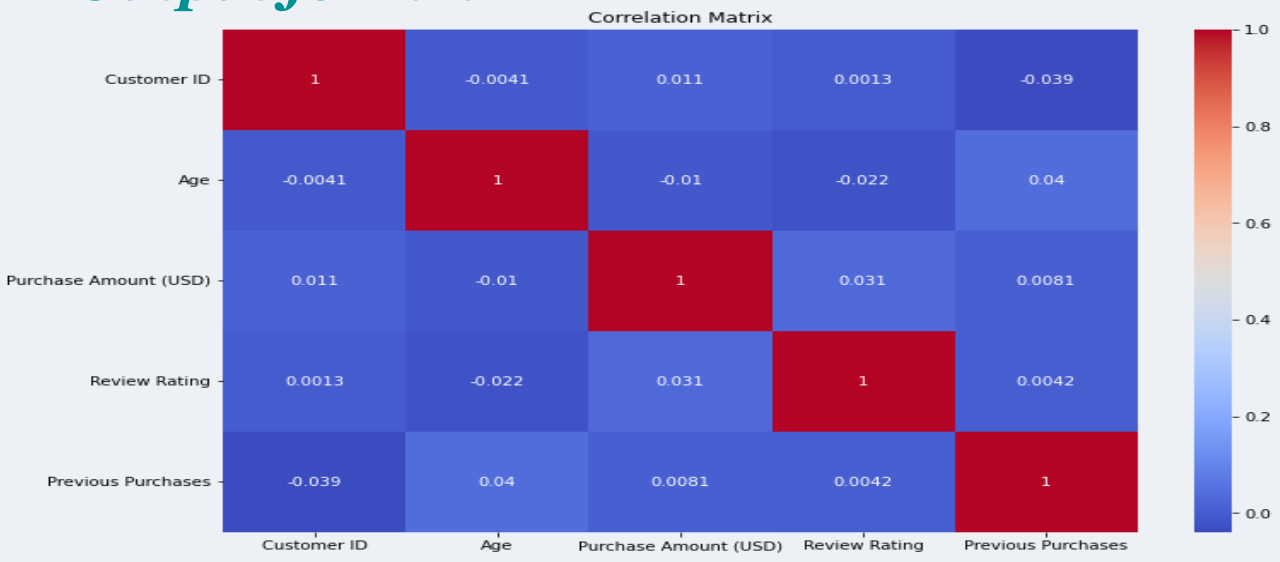
➢ *Output for 1.1:*

	Customer ID	Age	Purchase Amount (USD)	Review Rating	Previous Purchases
Customer ID	1.000000	-0.004079	0.011048	0.001343	-0.039159
Age	-0.004079	1.000000	-0.010424	-0.021949	0.040445
Purchase Amount (USD)	0.011048	-0.010424	1.000000	0.030776	0.008063
Review Rating	0.001343	-0.021949	0.030776	1.000000	0.004229
Previous Purchases	-0.039159	0.040445	0.008063	0.004229	1.000000

- ***Code 1.2:***

```
#Heatmap of the correlation matrix:
plt.figure(figsize=(12, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm')
plt.title('Correlation Matrix')
plt.show()
```

➢ *Output for 1.2:*



STEP 4(Analysis Model):

4.1 Preparing data for modeling:

- ***Here we'll apply a simple predictive model to estimate future purchase amounts based on available features.***
- ***Let's prepare the data for modeling by encoding categorical variables.***

▪ *One-hot encode categorical variables example:*

Label Encoding

Food Name	Categorical #	Calories
Apple	1	95
Chicken	2	231
Broccoli	3	50

→

One Hot Encoding

Apple	Chicken	Broccoli	Calories
1	0	0	95
0	1	0	231
0	0	1	50

• *Code :*

```
#One-hot encode categorical variables:
encoder = OneHotEncoder(sparse_output=False)
categorical columns = ['Gender', 'Category', 'Location', 'Season', 'Subscription Status',
                       'Shipping Type', 'Discount Applied', 'Promo Code Used',
                       'Payment Method', 'Frequency of Purchases']
encoded data = encoder.fit_transform(dataset[categorical_columns])

#Create a DataFrame from the encoded data:
encoded_df = pd.DataFrame(encoded_data, columns=encoder.get_feature_names_out
                                                (categorical_columns))

#Combine the encoded DF with the continuous variables:
model_data = pd.concat([dataset[['Age', 'Previous Purchases',
                                 'Purchase Amount (USD)']], encoded_df], axis=1)
model_data.head()
```

➢ *Output : ➔ 88 COLUMNS !!*

	Age	Previous Purchases	Purchase Amount (USD)	Gender_Female	Gender_Male	Category_Accessories	Category_Clothing	Category_Footwear	Category_Outerv
0	55	14	53	0.0	1.0	0.0	1.0	0.0	
1	19	2	64	0.0	1.0	0.0	1.0	0.0	
2	50	23	73	0.0	1.0	0.0	1.0	0.0	
3	21	49	90	0.0	1.0	0.0	0.0	1.0	
4	45	31	49	0.0	1.0	0.0	1.0	0.0	

5 rows × 88 columns

STEP 4(Analysis Model):

4.2 Training a regression model :

- ***Code :***

```
#Define features and target:
X = model_data.drop('Purchase Amount (USD)', axis=1)
y = model_data['Purchase Amount (USD)']

#Split the data into training and testing sets:
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

#Create a linear regression model:
model = LinearRegression()
model.fit(X_train, y_train)

#Predict on the testing set:
y_pred = model.predict(X_test)
```

STEP 4(Analysis Model):

4.3 Display model performance :

- ***Code :***

```
#Calculate the mean squared error of the model:
mse = mean_squared_error(y_test, y_pred)
print(f'Mean Squared Error: {mse}')
```

➔ Mean Squared Error: 570.4286629823538

```
# Calculate the R-squared of the model:
r2 = r2_score(y_test, y_pred)
print(f'R-squared: {r2}')
```

➔ R-squared: -0.0193828321515781

- ***Notes :***

- The Linear Regression model has been trained, and the mean squared error (MSE) on the test set is approximately 570.43. This value indicates the average squared difference between the predicted and actual purchase amounts. Lower MSE values would suggest a better fit of the model to the data.

- The Linear Regression model has been trained, and the R-squared value on the test set is approximately -0.019. This value indicates the proportion of the variance in the actual purchase amounts that is predictable from the independent variables used in the model.

STEP 5(Visualization):

5.1 Visualization of Actual vs Predicted Purchase Amounts :

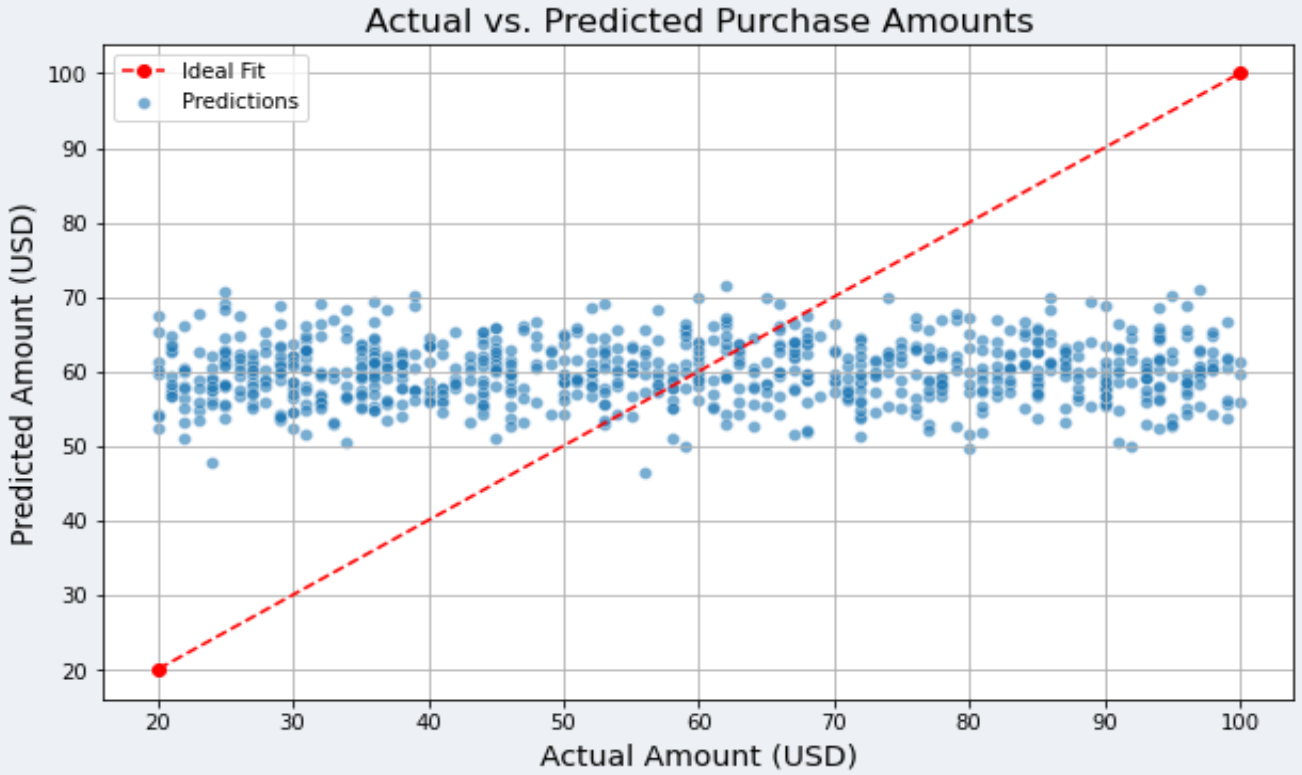
- **Let's visualize the actual vs. predicted purchase amounts to understand the model's performance visually.**

- ***This will help us see how closely the predicted values align with the actual values.***

• *Code :*

```
plt.figure(figsize=(10, 6))
scatter = sns.scatterplot(x=y_test, y=y_pred, alpha=0.6, label='Predictions')
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], '--r', marker='o',
                                                                    label='Ideal Fit')
plt.title('Actual vs. Predicted Purchase Amounts', fontsize=16)
plt.xlabel('Actual Amount (USD)', fontsize=14)
plt.ylabel('Predicted Amount (USD)', fontsize=14)
plt.legend(loc='best')
plt.grid(True)
plt.show()
```

➢ *Output for 1.1:*



Prescriptive:

1. **The plot of Actual vs. Predicted Purchase Amounts: Shows a reasonable alignment between the predicted values and the actual values, indicated by the proximity of the points to the dashed red line (ideal line where actual equals predicted). While there is some variance, which is expected in real-world scenarios, the model appears to provide a fair estimation of purchase amounts.**

2. **Distribution of Purchase Amounts: The purchase amounts are somewhat uniformly distributed, with a few higher purchase amounts indicating some premium items or bulk purchases.**

3. **Distribution of Previous Purchases: Most customers have made a moderate number of previous purchases, with a skew towards lower numbers, suggesting many new or infrequent customers.**

4. **Gender Distribution: The dataset seems fairly balanced in terms of gender distribution, which allows for gender-specific analysis if needed.**

5. **Top Categories of Purchases: Clothing appears to be the most popular category, followed by others such as footwear. This suggests focusing on these categories for promotions or targeted marketing.**

✓*References:*

1. DataSet link:

https://www.kaggle.com/datasets/iamsouravbanerjee/customer-shopping-trends-dataset/data

2. Data Acquisition:

https://www.geeksforgeeks.org/data-acquisition-system/

3. Data Cleaning:

https://www.sisense.com/glossary/data-cleaning/

4.Exploratory Analysis:

https://www.ibm.com/topics/exploratory-data-analysis

5.Analysis Model:

https://realpython.com/linear-regression-in-python/

6.Visualization:

https://www.simplilearn.com/tutorials/python-tutorial/data-visualization-in-python

7.Prescriptive:

https://www.qlik.com/us/augmented-analytics/prescriptive-analytics#:~:text=Prescriptive%20analytics%20is%20the%20use,Two%20factors%20driving%20growth.