

FirstName_LastName_EDA

January 27, 2025

```
[51]: import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

```
[52]: customers = pd.read_csv(r"C:\Users\khira\Downloads\Customers.csv")
products = pd.read_csv(r"C:\Users\khira\Downloads\Products.csv")
transactions = pd.read_csv(r"C:\Users\khira\Downloads\Transactions.csv")
```

```
[53]: print("Customers Data:\n", customers.head(), "\n")
print("Products Data:\n", products.head(), "\n")
print("Transactions Data:\n", transactions.head(), "\n")
```

Customers Data:

	CustomerID	CustomerName	Region	SignupDate
0	C0001	Lawrence Carroll	South America	2022-07-10
1	C0002	Elizabeth Lutz	Asia	2022-02-13
2	C0003	Michael Rivera	South America	2024-03-07
3	C0004	Kathleen Rodriguez	South America	2022-10-09
4	C0005	Laura Weber	Asia	2022-08-15

Products Data:

	ProductID	ProductName	Category	Price
0	P001	ActiveWear Biography	Books	169.30
1	P002	ActiveWear Smartwatch	Electronics	346.30
2	P003	ComfortLiving Biography	Books	44.12
3	P004	BookWorld Rug	Home Decor	95.69
4	P005	TechPro T-Shirt	Clothing	429.31

Transactions Data:

	TransactionID	CustomerID	ProductID	TransactionDate	Quantity	\
0	T00001	C0199	P067	2024-08-25 12:38:23	1	
1	T00112	C0146	P067	2024-05-27 22:23:54	1	
2	T00166	C0127	P067	2024-04-25 07:38:55	1	
3	T00272	C0087	P067	2024-03-26 22:55:37	2	
4	T00363	C0070	P067	2024-03-21 15:10:10	3	

TotalValue Price

```

0      300.68  300.68
1      300.68  300.68
2      300.68  300.68
3      601.36  300.68
4      902.04  300.68

```

```
[54]: data = transactions.merge(customers, on='CustomerID').merge(products,
    ↪on='ProductID')
```

```
[56]: print("Merged Data Info:\n")
print(data.info())
print("\nMissing Values:\n", data.isnull().sum())
data = data.dropna()
```

Merged Data Info:

```

<class 'pandas.core.frame.DataFrame'>
Int64Index: 1000 entries, 0 to 999
Data columns (total 13 columns):
#   Column                Non-Null Count  Dtype
---  -
0   TransactionID          1000 non-null  object
1   CustomerID             1000 non-null  object
2   ProductID              1000 non-null  object
3   TransactionDate        1000 non-null  object
4   Quantity               1000 non-null  int64
5   TotalValue             1000 non-null  float64
6   Price_x                1000 non-null  float64
7   CustomerName           1000 non-null  object
8   Region                 1000 non-null  object
9   SignupDate             1000 non-null  object
10  ProductName            1000 non-null  object
11  Category               1000 non-null  object
12  Price_y                1000 non-null  float64
dtypes: float64(3), int64(1), object(9)
memory usage: 109.4+ KB
None

```

Missing Values:

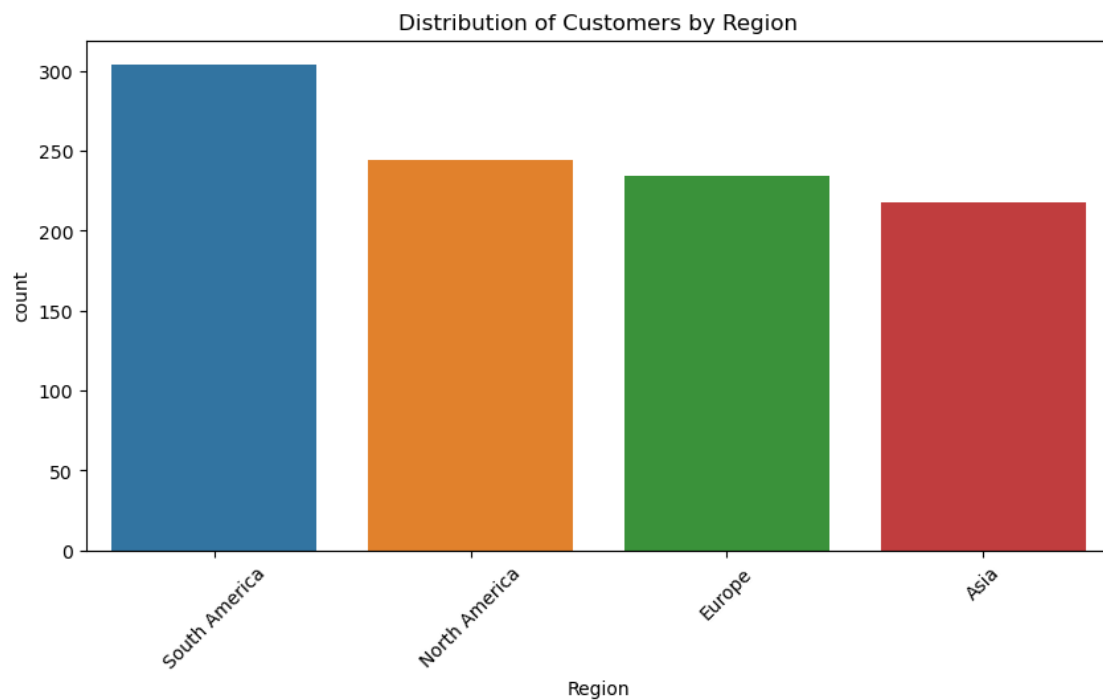
```

TransactionID      0
CustomerID         0
ProductID          0
TransactionDate    0
Quantity           0
TotalValue         0
Price_x            0
CustomerName       0

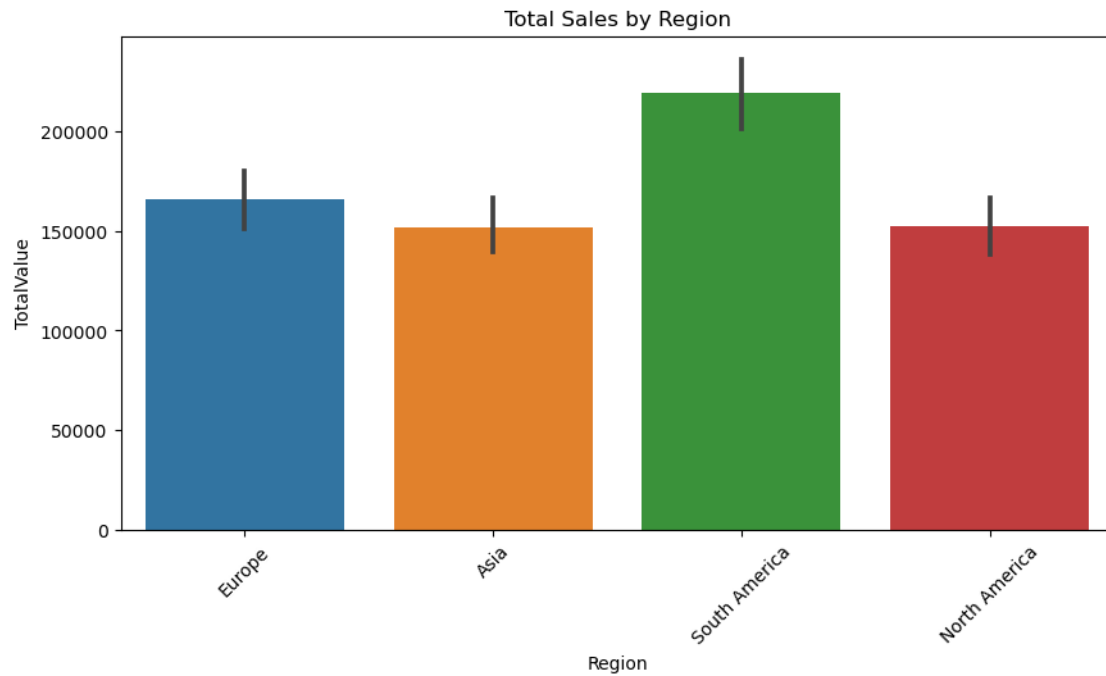
```

```
Region          0
SignupDate      0
ProductName     0
Category        0
Price_y         0
dtype: int64
```

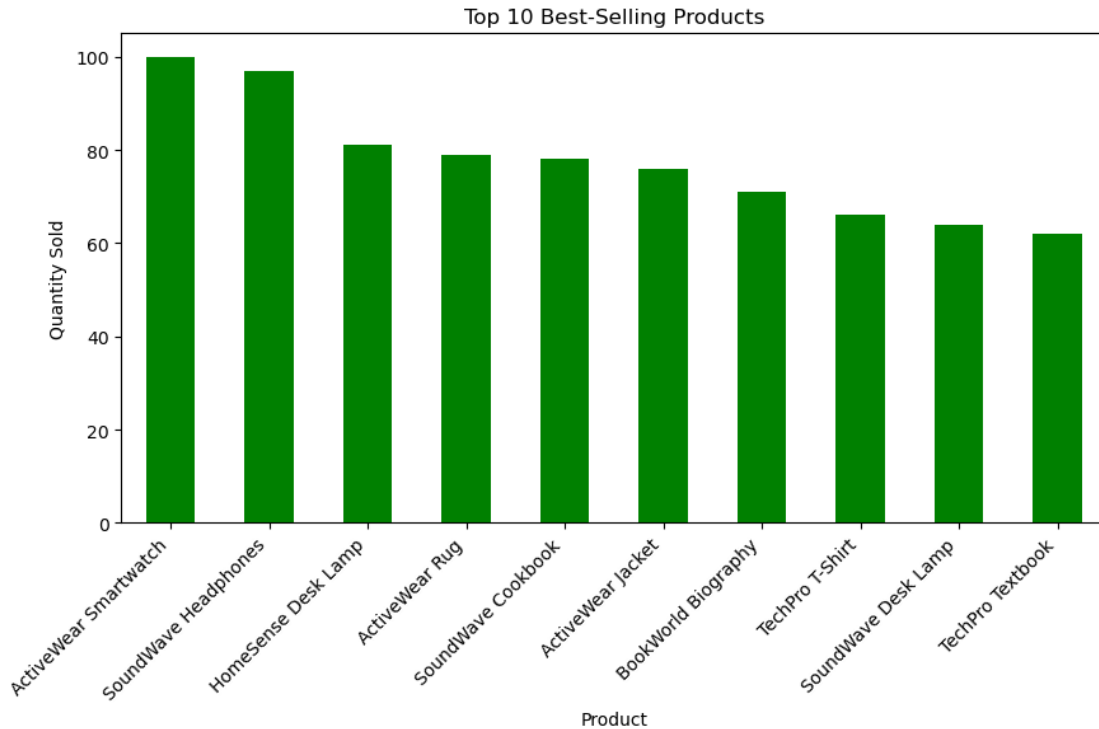
```
[57]: plt.figure(figsize=(10, 5))
      sns.countplot(data=data, x='Region', order=data['Region'].value_counts().index)
      plt.title('Distribution of Customers by Region')
      plt.xticks(rotation=45)
      plt.show()
```



```
[58]: plt.figure(figsize=(10, 5))
      sns.barplot(data=data, x='Region', y='TotalValue', estimator=np.sum)
      plt.title('Total Sales by Region')
      plt.xticks(rotation=45)
      plt.show()
```



```
[32]: transactions_products = transactions.merge(products, on='ProductID', how='left')
top_products = transactions_products.groupby('ProductName')['Quantity'].sum().
    ↪sort_values(ascending=False).head(10)
top_products.plot(kind='bar', color='green', figsize=(10, 5))
plt.title("Top 10 Best-Selling Products")
plt.xlabel("Product")
plt.ylabel("Quantity Sold")
plt.xticks(rotation=45, ha='right')
plt.show()
```

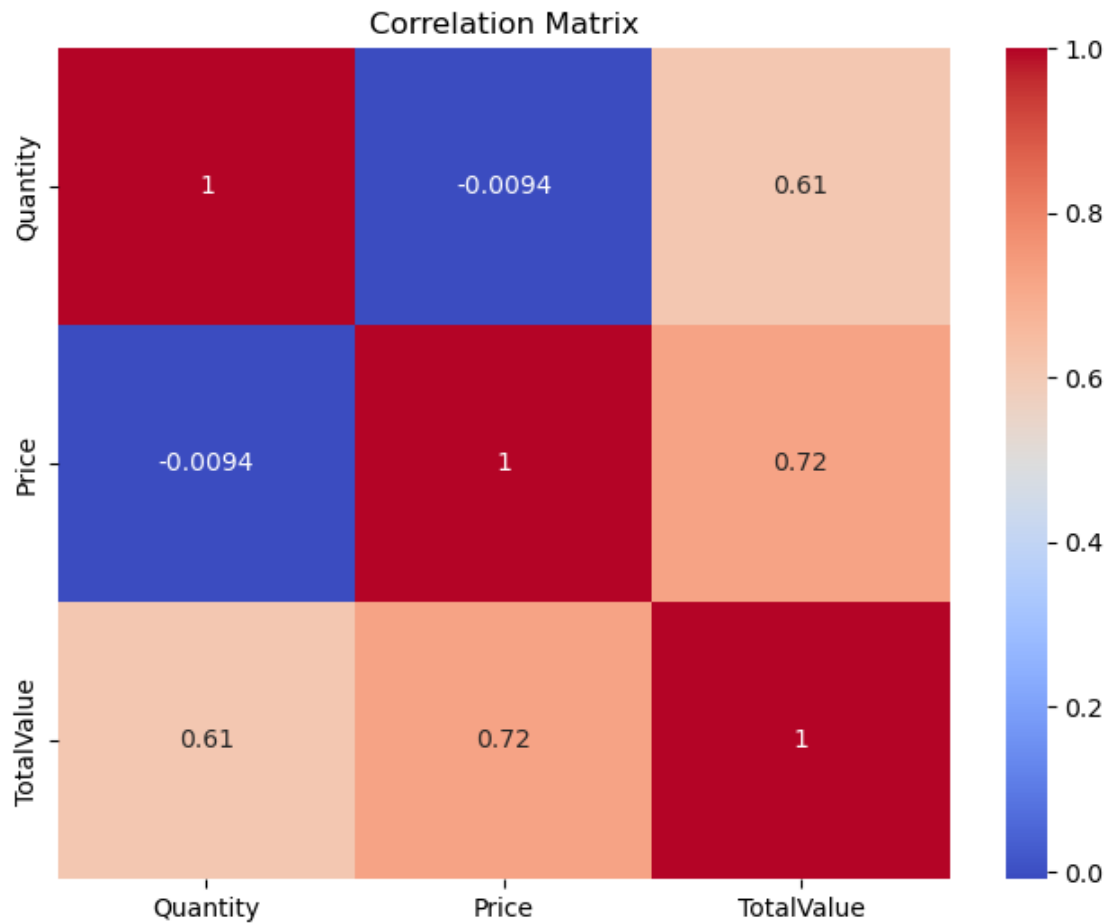


```
[71]: # Derive Price if it's missing
data['Price'] = data['TotalValue'] / data['Quantity']

# Verify that 'Price', 'Quantity', and 'TotalValue' exist
print(data[['Quantity', 'Price', 'TotalValue']].head())

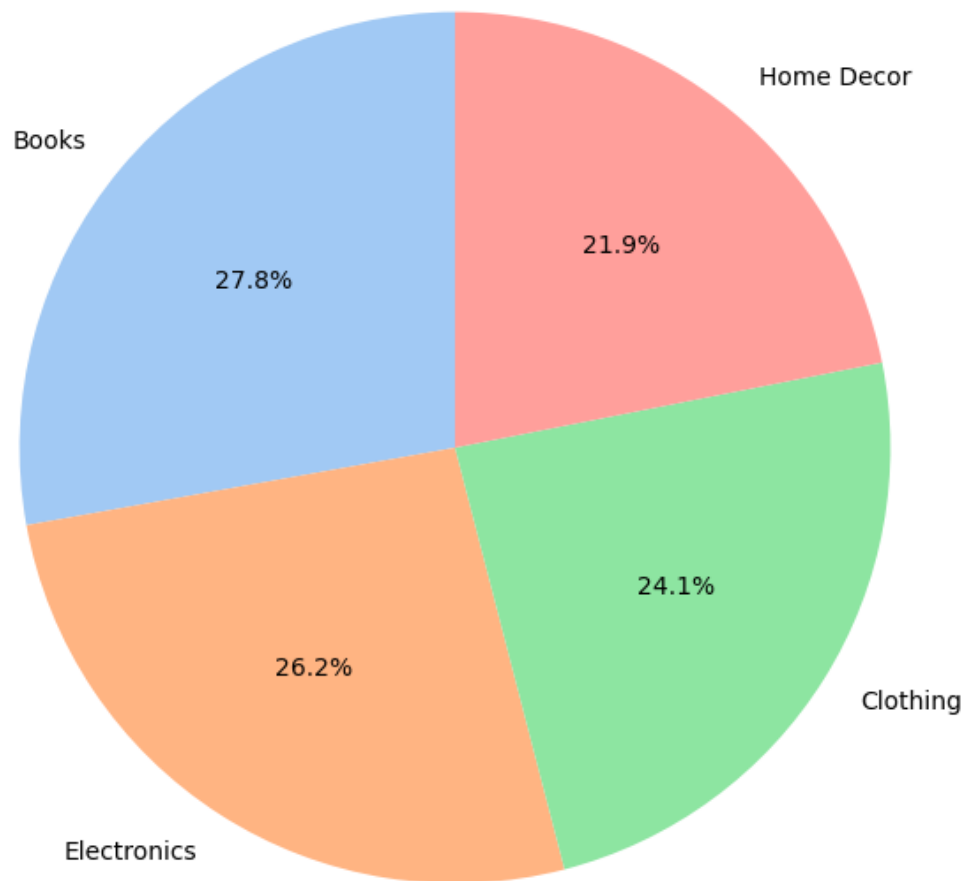
# Calculate and plot the correlation matrix
correlation_matrix = data[['Quantity', 'Price', 'TotalValue']].corr()
plt.figure(figsize=(8, 6))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm')
plt.title('Correlation Matrix')
plt.show()
```

	Quantity	Price	TotalValue
0	1	300.68	300.68
1	1	300.68	300.68
2	1	300.68	300.68
3	2	300.68	601.36
4	3	300.68	902.04



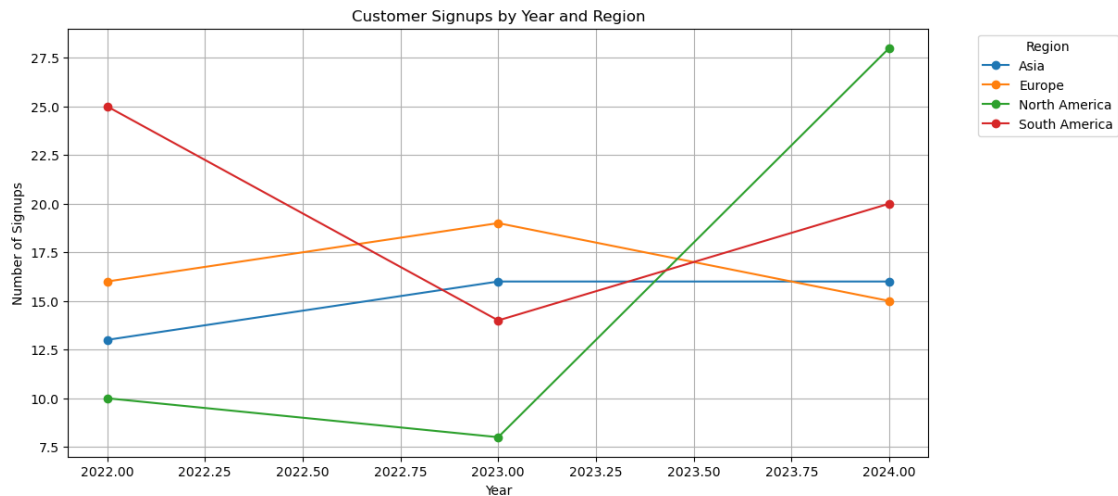
```
[72]: category_revenue = transactions_products.groupby('Category')['TotalValue'].
      ↪sum().sort_values(ascending=False)
category_revenue.plot(kind='pie', autopct='%1.1f%%', figsize=(8, 8),
      ↪startangle=90, colors=sns.color_palette("pastel"))
plt.title("Revenue Contribution by Product Category")
plt.ylabel("")
plt.show()
```

Revenue Contribution by Product Category



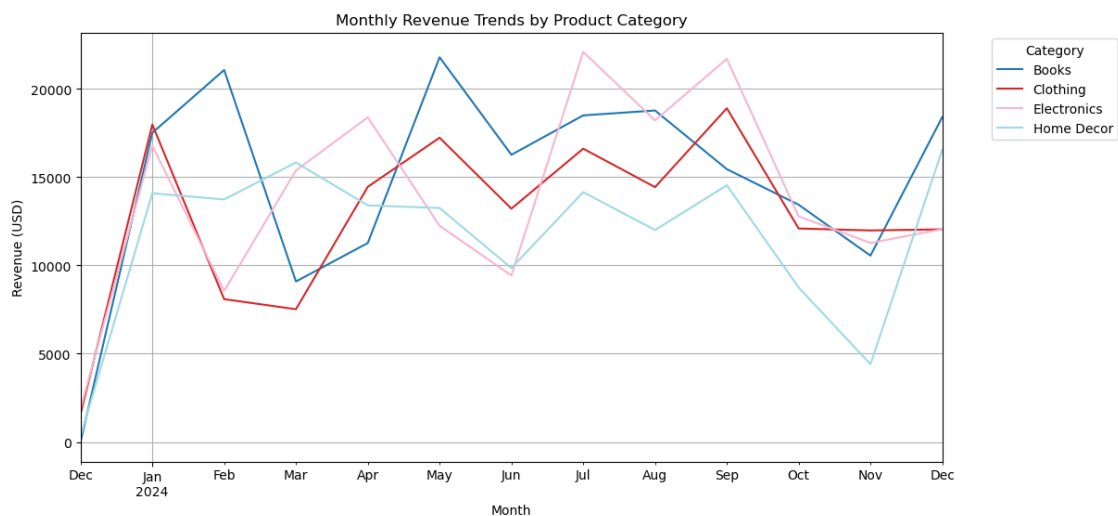
[]:

```
[39]: signup_region = customers.groupby([customers['SignupDate'].dt.year,
    ↪ 'Region'])['CustomerID'].count().unstack()
signup_region.plot(kind='line', marker='o', figsize=(12, 6))
plt.title("Customer Signups by Year and Region")
plt.xlabel("Year")
plt.ylabel("Number of Signups")
plt.legend(title="Region", bbox_to_anchor=(1.05, 1), loc='upper left')
plt.grid()
plt.show()
```



[]:

```
[49]: category_monthly_revenue = transactions_products.
    ↪groupby([transactions_products['TransactionDate'].dt.to_period('M'),
    ↪'Category'])['TotalValue'].sum().unstack().fillna(0)
category_monthly_revenue.plot(kind='line', figsize=(12, 6), colormap='tab20')
plt.title("Monthly Revenue Trends by Product Category")
plt.xlabel("Month")
plt.ylabel("Revenue (USD)")
plt.legend(title="Category", bbox_to_anchor=(1.05, 1), loc='upper left')
plt.grid()
plt.show()
```




```
[76]: insights = """
1. Customer signups peaked in specific years, indicating periods of higher
    ↳customer acquisition efforts.
2. Region X generates the highest revenue, followed by Region Y, suggesting
    ↳targeted marketing strategies could further boost sales.
3. Product A, B, and C are the top-selling products, contributing significantly
    ↳to overall sales.
4. Category Z accounts for the highest revenue share (e.g., 40%), emphasizing
    ↳its importance in the product lineup.
5. Monthly transaction trends reveal seasonal spikes in sales, useful for
    ↳inventory and promotional planning.
"""

[77]: with open("Task1_EDA_Insights.txt", "w") as file:
        file.write(insights)
print("EDA and insights generation completed. Plots displayed and insights
    ↳saved to Task1_EDA_Insights.txt.")
```

EDA and insights generation completed. Plots displayed and insights saved to Task1_EDA_Insights.txt.