Module 5: Data Visualisation

Case Study II

1. Domain – Retail (focus – Visualize the sales data)

Business challenge/requirement
BigMart is one of the biggest retailer of Europe and has operations across multiple countries. You are a data analyst in IT team of BigMart. Invoice and SKU wise Sales Data for Year 2011 is shared with you. You need to prepare meaningful charts to show case the various sales trends for 2011 to top management.

Key issues
Data should be displayed pictorially to capture the attention of top management

Data volume
- Approx 500K records – file BigMartSalesData.csv

Business benefits
This exercise is an annual exercise and BigMart makes important investment decision based on trends

Approach to Solve
You have to use fundamentals of Matplotlib covered in module 5 and plot following 4 chart/graph

1. Plot Total Sales Per Month for Year 2011. How the total sales have increased over months in Year 2011. Which month has lowest Sales?
2. Plot Total Sales Per Month for Year 2011 as Bar Chart. Is Bar Chart Better to visualize than Simple Plot?
3. Plot Pie Chart for Year 2011 Country Wise. Which Country contributes highest towards sales?
4. Plot Scatter Plot for the invoice amounts and see the concentration of amount. In which range most of the invoice amounts are concentrated

Enhancements for code
You can try these enhancements in code

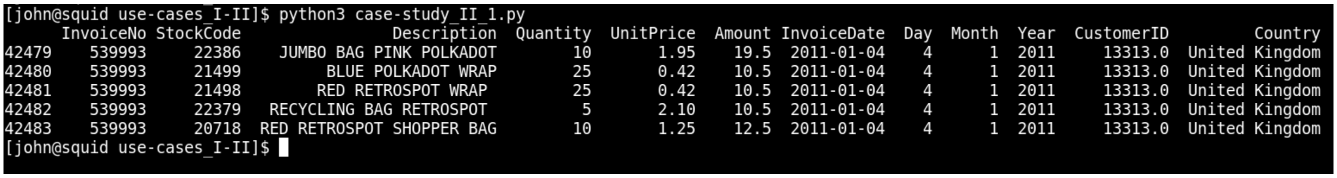
1. Change the bar chart to show the value of bar
2. In Pie Chart Play With Parameters shadow=True, startangle=90 and see how different the chart looks
3. In scatter plot change the color of Scatter Points

Let's go through the steps to analyze and visualize the sales data from BigMart for the year 2011 using Python libraries NumPy, Pandas, and Matplotlib.

First, we will load the data and perform some preliminary processing. Then, we will create the required visualizations.

Step 1: Load and preprocess the data

```
import pandas as pd

# Load the data
data = pd.read_csv('BigMartSalesData.csv')

# Convert InvoiceDate to datetime format
data['InvoiceDate'] = pd.to_datetime(data['InvoiceDate'], format='%d-%m-%y')

# Filter data for the year 2011
data_2011 = data[data['InvoiceDate'].dt.year == 2011].copy()

# Ensure all columns are in correct types
data_2011['Month'] = data_2011['InvoiceDate'].dt.month
data_2011['Country'] = data_2011['Country'].astype('category')

# Preview the data
print(data_2011.head())
```

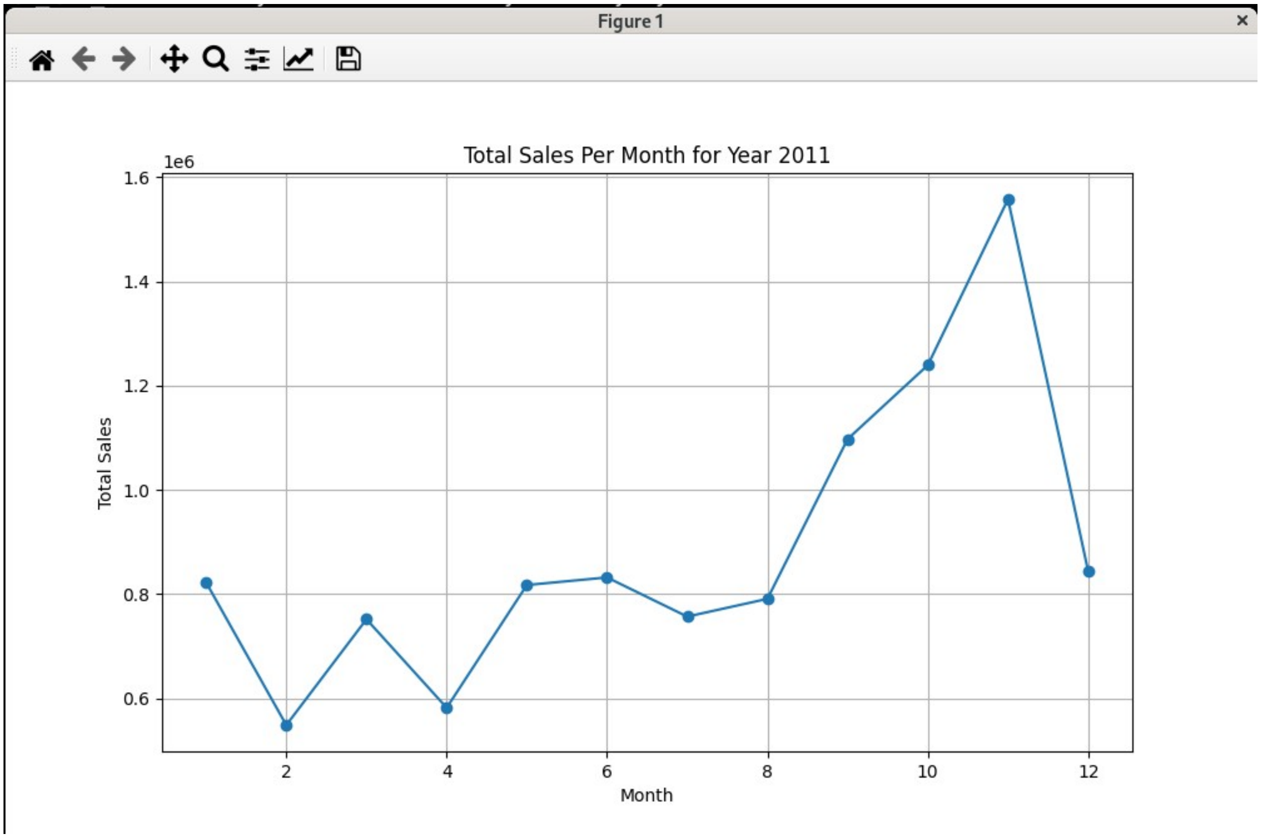


```
[john@squid use-cases_I-II]$ python3 case-study_II_1.py
      InvoiceNo StockCode                 Description  Quantity  UnitPrice  Amount InvoiceDate  Day  Month  Year  CustomerID         Country
42479    539993     22386     JUMBO BAG PINK POLKADOT        10       1.95    19.5  2011-01-04    4      1  2011     13313.0  United Kingdom
42480    539993     21499          BLUE POLKADOT WRAP        25       0.42    10.5  2011-01-04    4      1  2011     13313.0  United Kingdom
42481    539993     21498         RED RETROSPOT WRAP        25       0.42    10.5  2011-01-04    4      1  2011     13313.0  United Kingdom
42482    539993     22379    RECYCLING BAG RETROSPOT         5       2.10    10.5  2011-01-04    4      1  2011     13313.0  United Kingdom
42483    539993     20718  RED RETROSPOT SHOPPER BAG        10       1.25    12.5  2011-01-04    4      1  2011     13313.0  United Kingdom
[john@squid use-cases_I-II]$ 
```

Step 2: Plot Total Sales Per Month for Year 2011

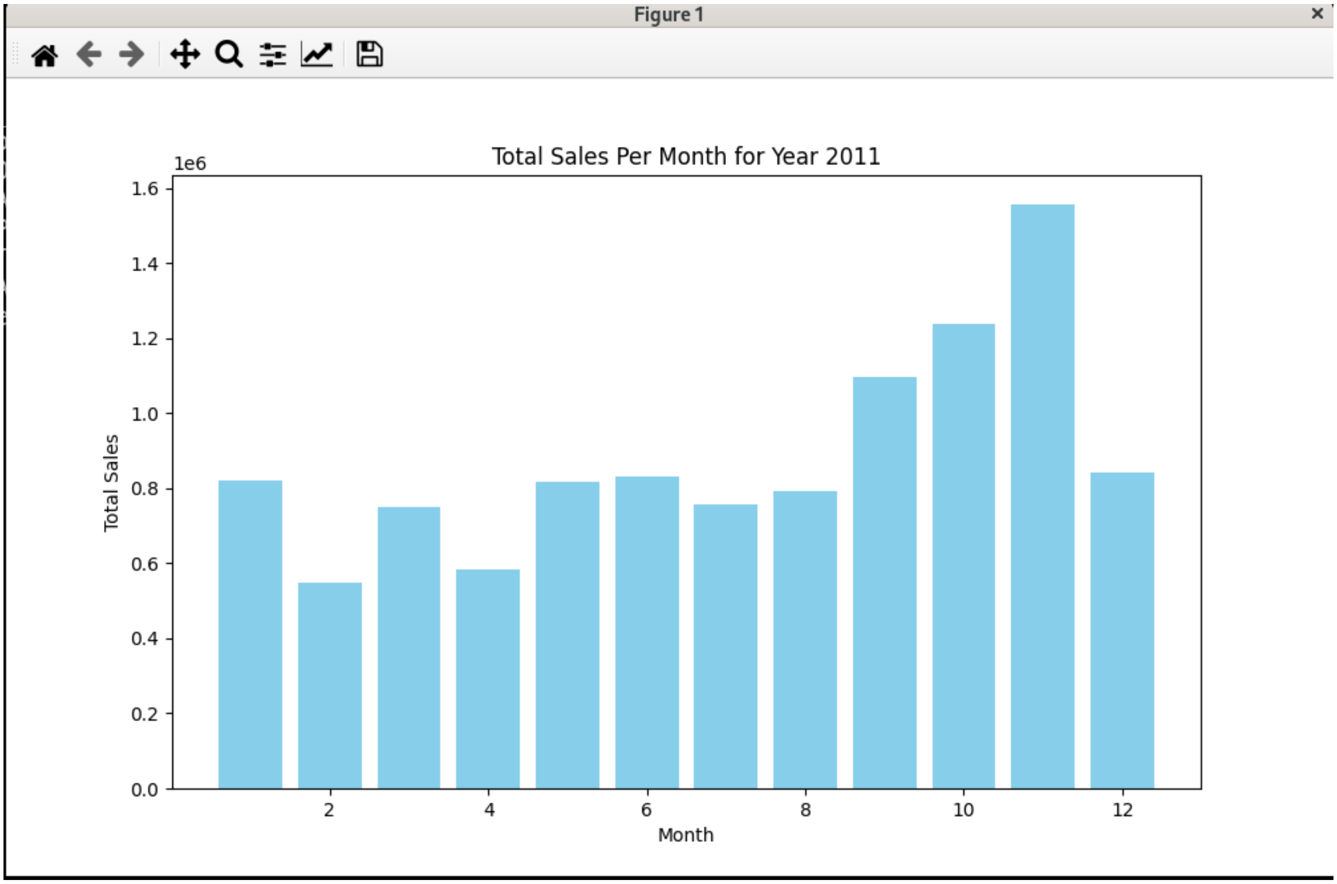
```
import pandas as pd
import matplotlib.pyplot as plt

# Load the data
data = pd.read_csv('BigMartSalesData.csv')

# Convert InvoiceDate to datetime format
data['InvoiceDate'] = pd.to_datetime(data['InvoiceDate'], format='%d-%m-%y')

# Filter data for the year 2011
data_2011 = data[data['InvoiceDate'].dt.year == 2011].copy()
```

```
# Ensure all columns are in correct types
data_2011['Month'] = data_2011['InvoiceDate'].dt.month
data_2011['Country'] = data_2011['Country'].astype('category')

# Preview the data
print(data_2011.head())

# Group by month and calculate total sales
monthly_sales = data_2011.groupby('Month')['Amount'].sum()

# Plot Total Sales Per Month
plt.figure(figsize=(10, 6))
plt.plot(monthly_sales.index, monthly_sales.values, marker='o')
plt.title('Total Sales Per Month for Year 2011')
plt.xlabel('Month')
plt.ylabel('Total Sales')
plt.grid(True)
plt.show()

# Identify the month with the lowest sales
lowest_sales_month = monthly_sales.idxmin()
print(lowest_sales_month)
```



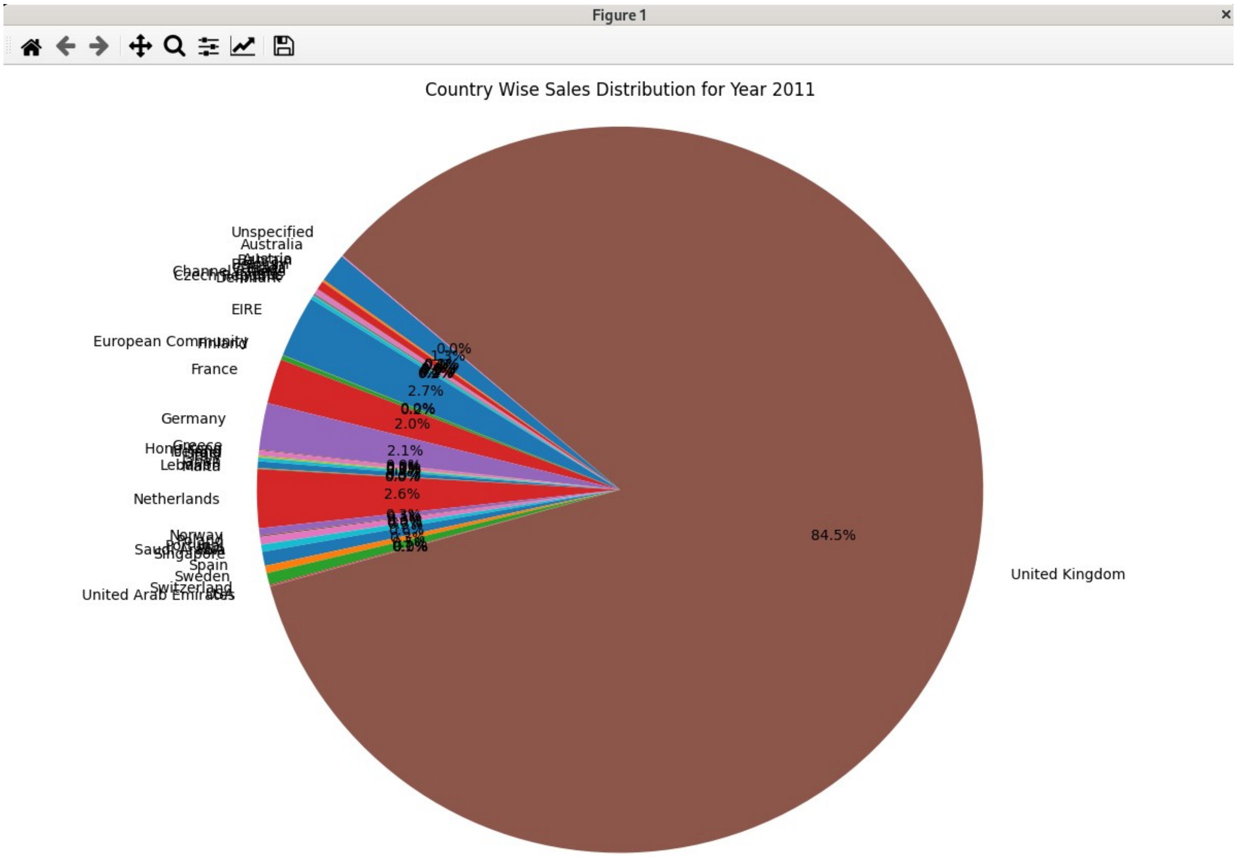
Step 3: Plot Total Sales Per Month for Year 2011 as Bar Chart

```
# Bar Chart of Total Sales Per Month
plt.figure(figsize=(10, 6))
plt.bar(monthly_sales.index, monthly_sales.values, color='skyblue')
plt.title('Total Sales Per Month for Year 2011')
plt.xlabel('Month')
plt.ylabel('Total Sales')
plt.show()
```



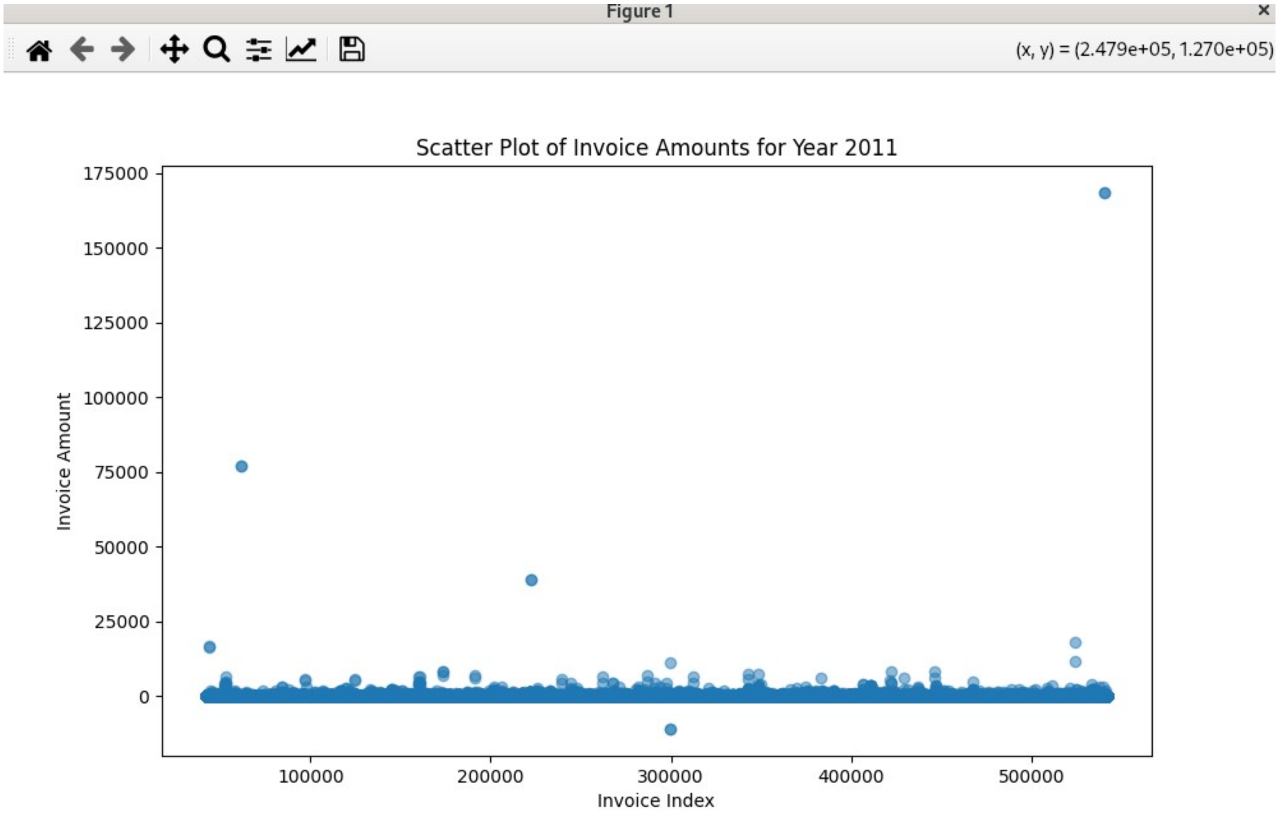
Step 4: Plot Pie Chart for Year 2011 Country Wise

```
# Group by country and calculate total sales
country_sales = data_2011.groupby('Country')['Amount'].sum()

# Plot Pie Chart for Country Wise Sales
plt.figure(figsize=(12, 8))
plt.pie(country_sales, labels=country_sales.index, autopct='%1.1f%%', startangle=140)
plt.title('Country Wise Sales Distribution for Year 2011')
plt.axis('equal')
plt.tight_layout()  # Adjust layout to not cut off labels
plt.show()

# Identify the country with the highest sales
highest_sales_country = country_sales.idxmax()
print(highest_sales_country)
```



Step 5: Plot Scatter Plot for the Invoice Amounts

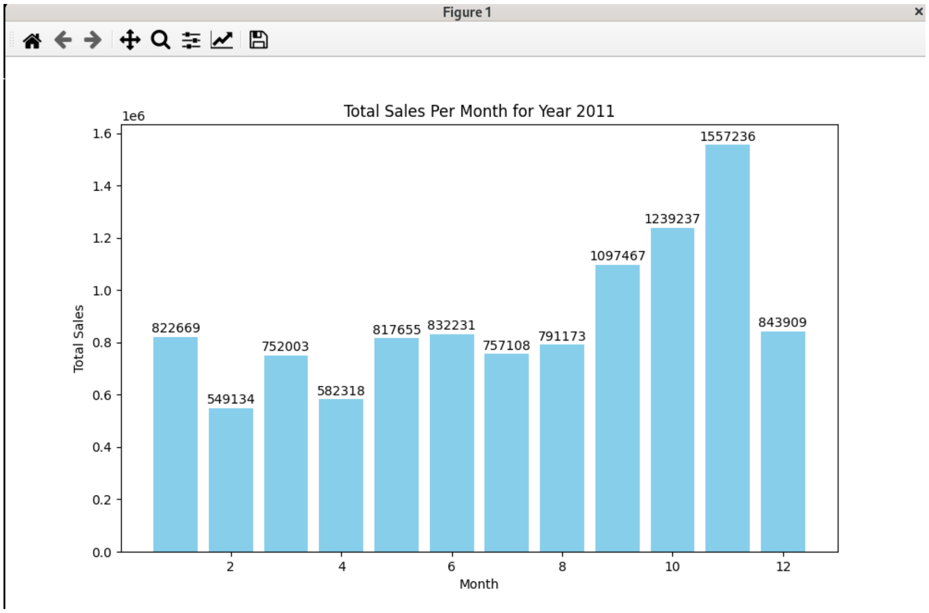
```
# Step 5: Plot Scatter Plot for the Invoice Amounts
# Scatter Plot for Invoice Amounts
plt.figure(figsize=(10, 6))
plt.scatter(data_2011.index, data_2011['Amount'], alpha=0.5)
plt.title('Scatter Plot of Invoice Amounts for Year 2011')
plt.xlabel('Invoice Index')
plt.ylabel('Invoice Amount')
plt.show()

# Analyze the concentration of invoice amounts
invoice_amount_range = data_2011['Amount'].value_counts(bins=10).sort_index()
print(invoice_amount_range)
```

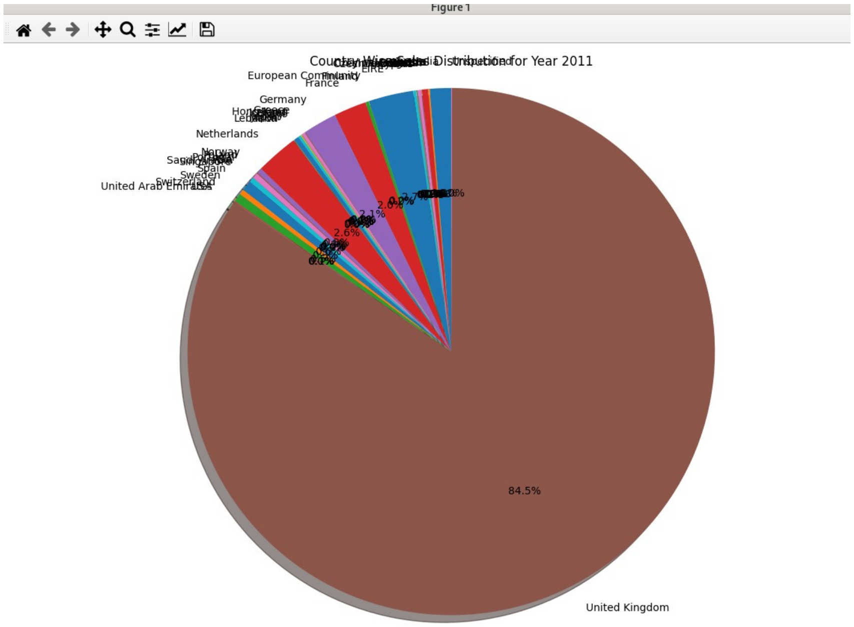


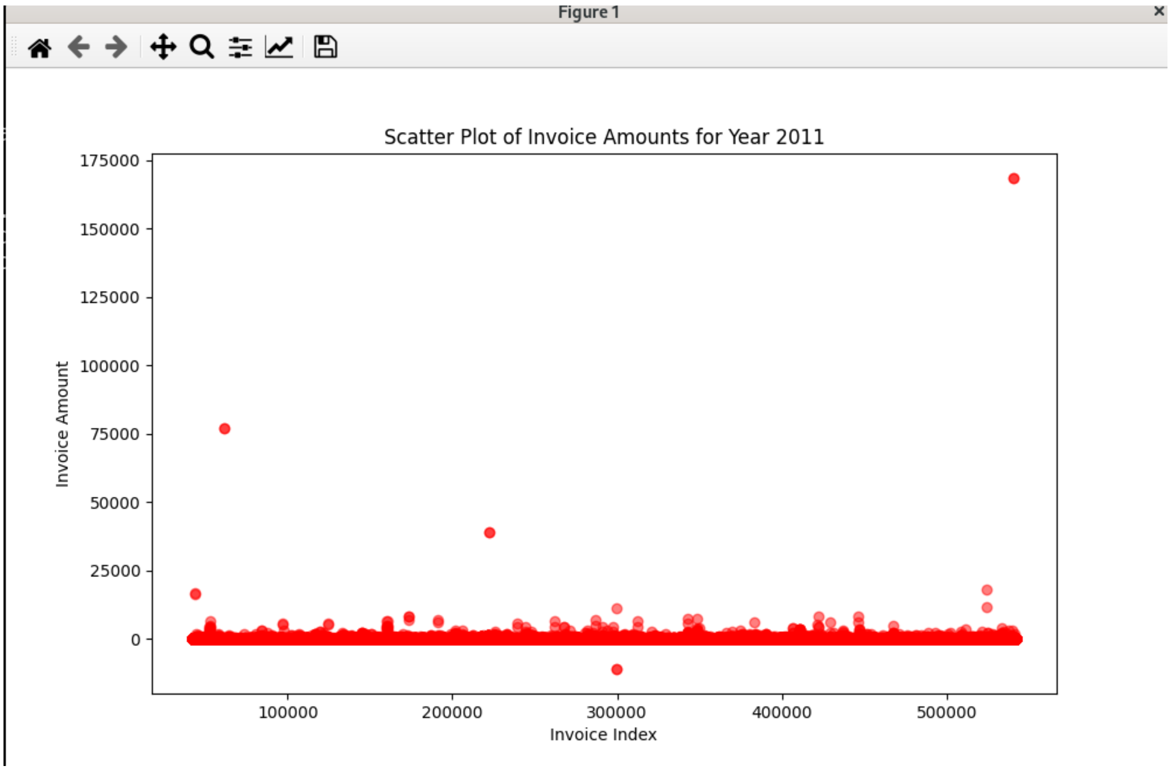
Enhancements for code

You can try these enhancements in code

1. Change the bar chart to show the value of bar
2. In Pie Chart Play With Parameters shadow=True, startangle=90 and see how different the chart looks
3. In scatter plot change the color of Scatter Points







```
import pandas as pd
import matplotlib.pyplot as plt

# Step 1: Load and preprocess the data
# Load the data
data = pd.read_csv('BigMartSalesData.csv')

# Convert InvoiceDate to datetime format
data['InvoiceDate'] = pd.to_datetime(data['InvoiceDate'], format='%d-%m-%y')

# Filter data for the year 2011
data_2011 = data[data['InvoiceDate'].dt.year == 2011].copy()

# Ensure all columns are in correct types
data_2011['Month'] = data_2011['InvoiceDate'].dt.month
data_2011['Country'] = data_2011['Country'].astype('category')

# Preview the data
print(data_2011.head())

# Group by month and calculate total sales
monthly_sales = data_2011.groupby('Month')['Amount'].sum()
```

```
# Step 2: Plot Total Sales Per Month for Year 2011
# Plot Total Sales Per Month
plt.figure(figsize=(10, 6))
plt.plot(monthly_sales.index, monthly_sales.values, marker='o')
plt.title('Total Sales Per Month for Year 2011')
plt.xlabel('Month')
plt.ylabel('Total Sales')
plt.grid(True)
plt.show()

# Identify the month with the lowest sales
lowest_sales_month = monthly_sales.idxmin()
print(lowest_sales_month)

# Step 3: Plot Total Sales Per Month for Year 2011 as Bar Chart
# Bar Chart of Total Sales Per Month
plt.figure(figsize=(10, 6))
bars = plt.bar(monthly_sales.index, monthly_sales.values, color='skyblue')
plt.title('Total Sales Per Month for Year 2011')
plt.xlabel('Month')
plt.ylabel('Total Sales')

# Adding value labels on top of the bars
for bar in bars:
    yval = bar.get_height()
    plt.text(bar.get_x() + bar.get_width()/2, yval + 5000, int(yval), ha='center', va='bottom')

plt.show()

# Step 4: Plot Pie Chart for Year 2011 Country Wise
# Group by country and calculate total sales
country_sales = data_2011.groupby('Country')['Amount'].sum()

# Plot Pie Chart for Country Wise Sales
plt.figure(figsize=(12, 8))
plt.pie(country_sales, labels=country_sales.index, autopct='%1.1f%%', shadow=True, startangle=90)
plt.title('Country Wise Sales Distribution for Year 2011')
plt.axis('equal')
plt.tight_layout()  # Adjust layout to not cut off labels
plt.show()

# Identify the country with the highest sales
highest_sales_country = country_sales.idxmax()
print(highest_sales_country)

# Step 5: Plot Scatter Plot for the Invoice Amounts
# Scatter Plot for Invoice Amounts
```

```
plt.figure(figsize=(10, 6))
plt.scatter(data_2011.index, data_2011['Amount'], alpha=0.5, color='red')
plt.title('Scatter Plot of Invoice Amounts for Year 2011')
plt.xlabel('Invoice Index')
plt.ylabel('Invoice Amount')
plt.show()

# Analyze the concentration of invoice amounts
invoice_amount_range = data_2011['Amount'].value_counts(bins=10).sort_index()
print(invoice_amount_range)
```