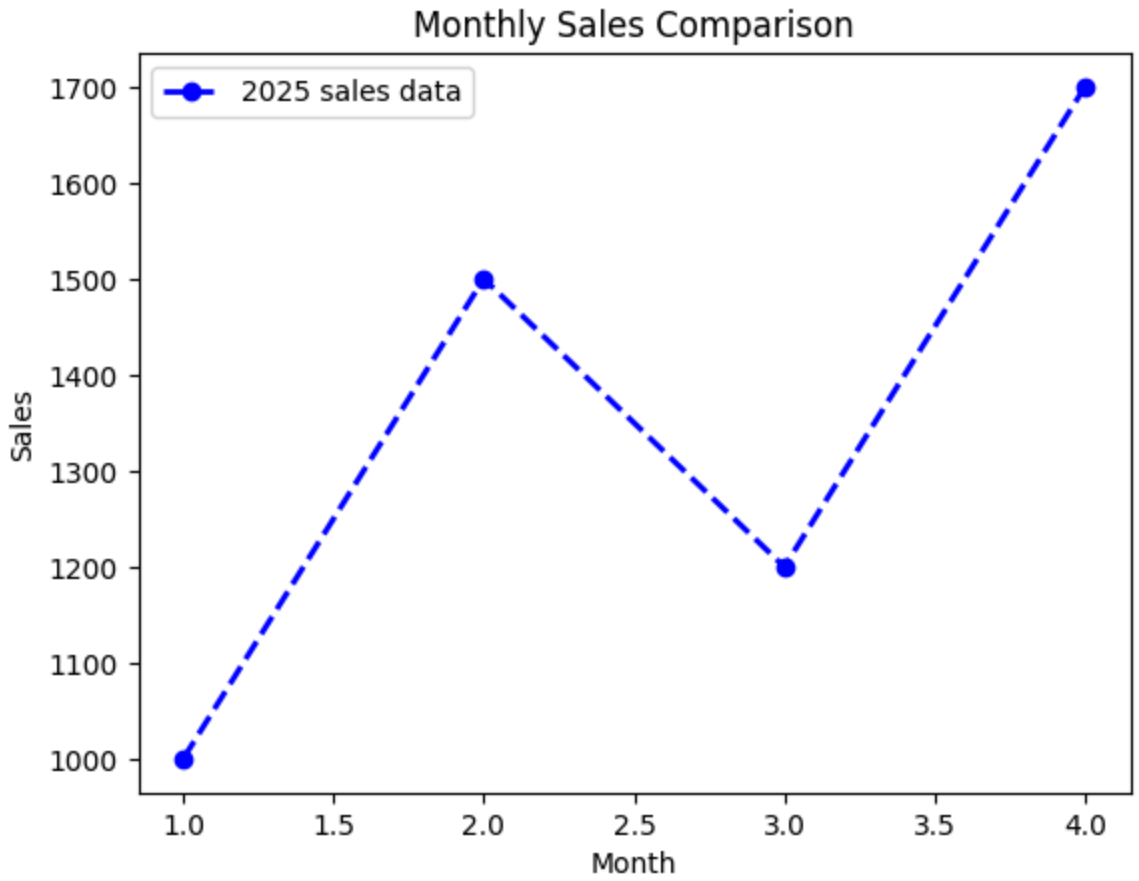
Import matplotlib library

```
import matplotlib.pyplot as plt
```

use of plot() function

plt.plot(x,y, color = 'color_name', linestyle = 'linestyle', marker = 'marker symble, label = 'label name')

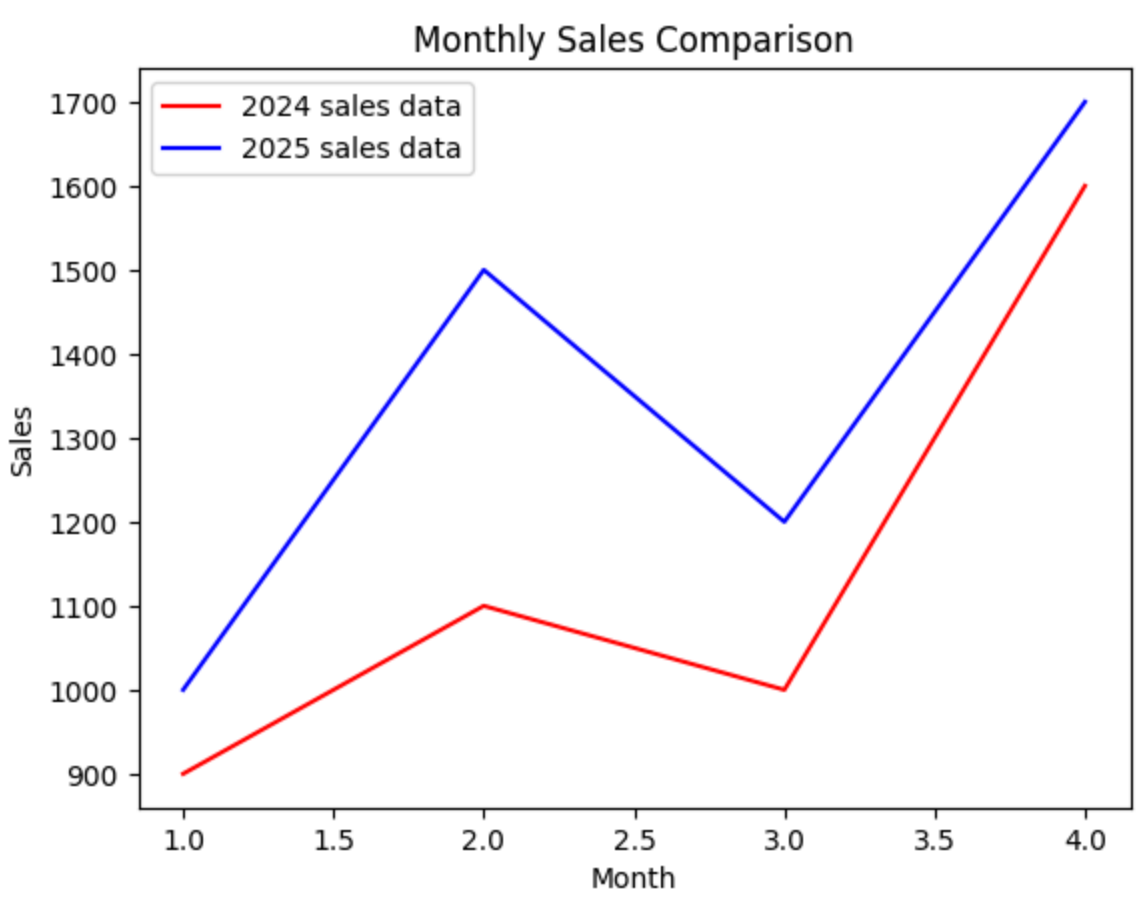
```
months = [1,2,3,4]
sales = [1000,1500,1200,1700]
plt.plot(months,sales, color= 'blue', linestyle='--', linewidth = 2, marker = 'o', label = '2025 sales data')
plt.xlabel('Month')
plt.ylabel('Sales')
plt.title('Monthly Sales Comparison')
plt.legend()
plt.show()
```



uese of label parametres

```
months = [1, 2, 3, 4]
sales_2024 = [900, 1100, 1000, 1600]
sales_2025 = [1000, 1500, 1200, 1700]

plt.plot(months, sales_2024, label='2024 sales data', color='red')
plt.plot(months, sales_2025, label='2025 sales data', color='blue')

plt.xlabel('Month')
```

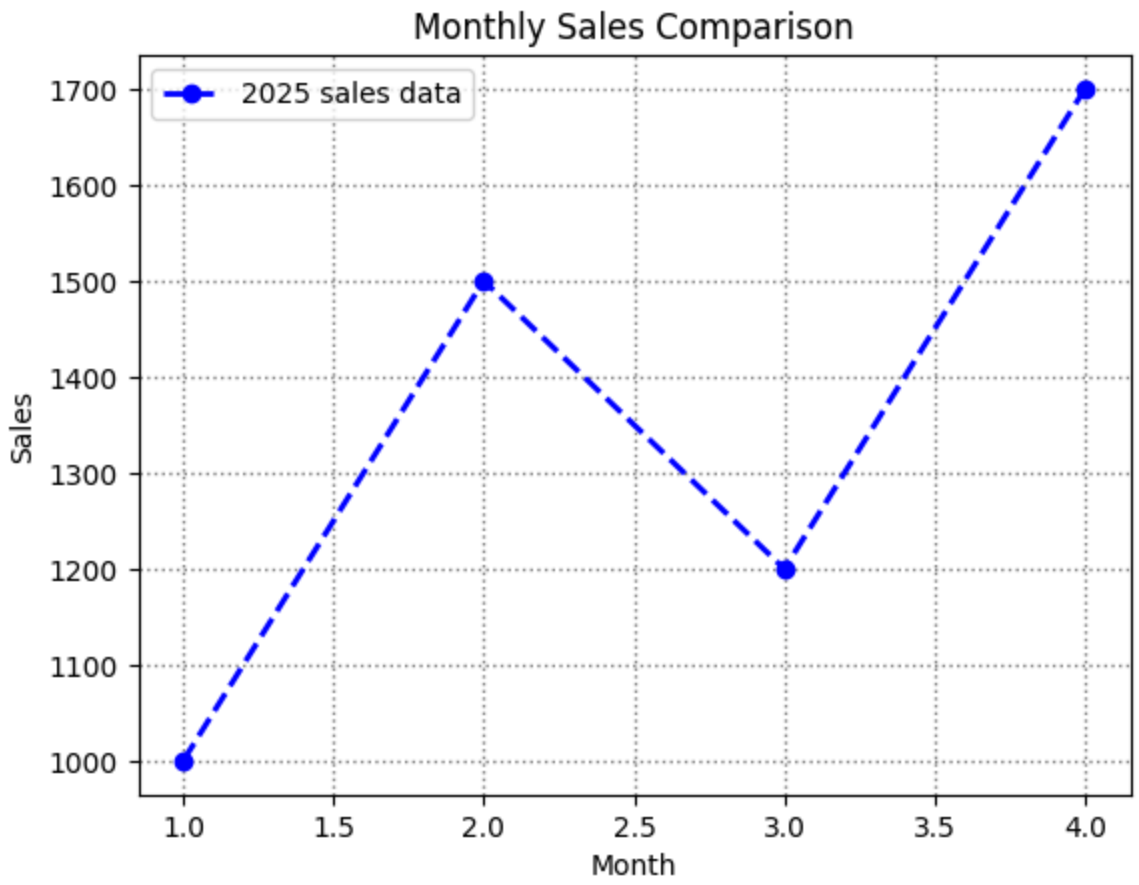
```
plt.ylabel('Sales')
plt.title('Monthly Sales Comparison')
plt.legend()  # Show the labels
plt.show()
```



use grid() function. to show background in figure.

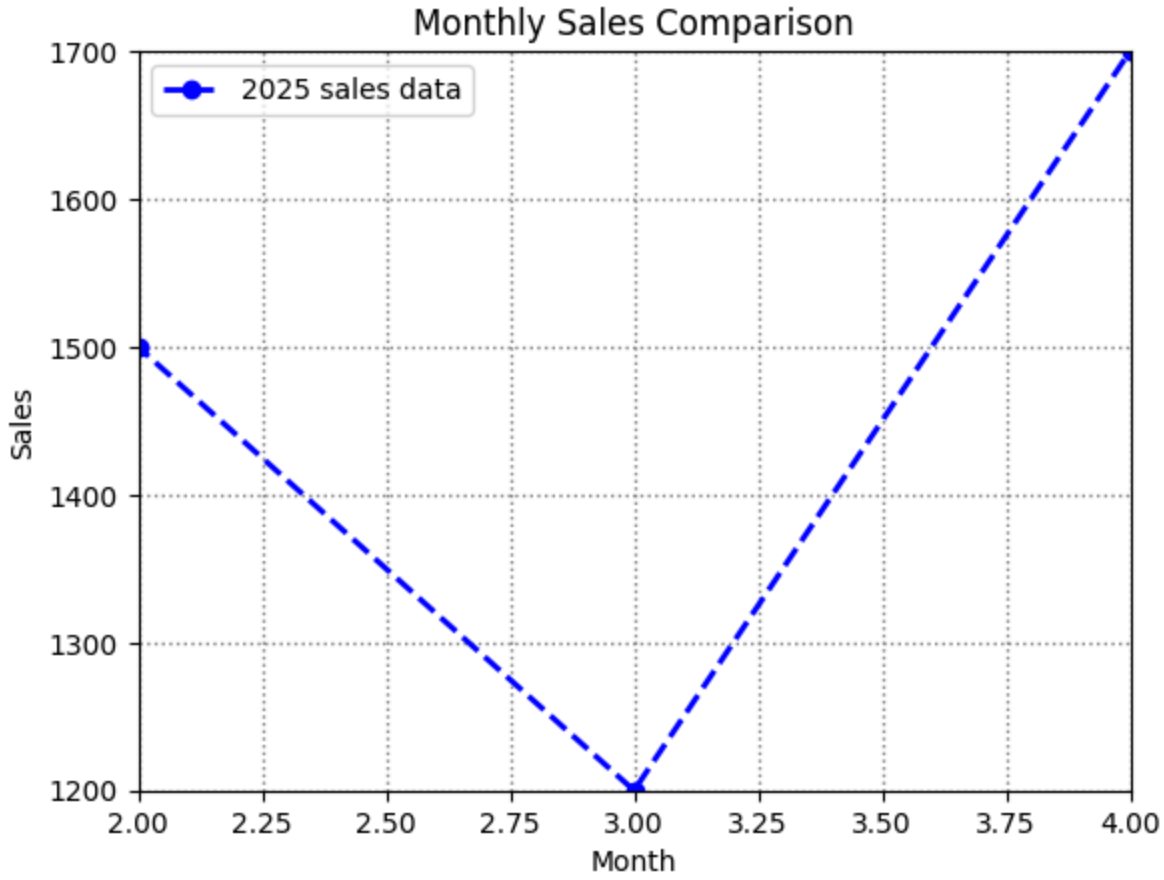
```
months = [1,2,3,4]
sales = [1000,1500,1200,1700]
plt.plot(months,sales, color= 'blue', linestyle='--', linewidth = 2, marker = 'o', label = '2025 sales data')
```

```
plt.xlabel('Month')
plt.ylabel('Sales')
plt.title('Monthly Sales Comparison')
plt.legend()
plt.grid(color='gray', linestyle = ':', linewidth =1)
plt.show()
```



use of xlim() and ylim() function. It use to set range of x and y value. here x is my input value and y is my out value.

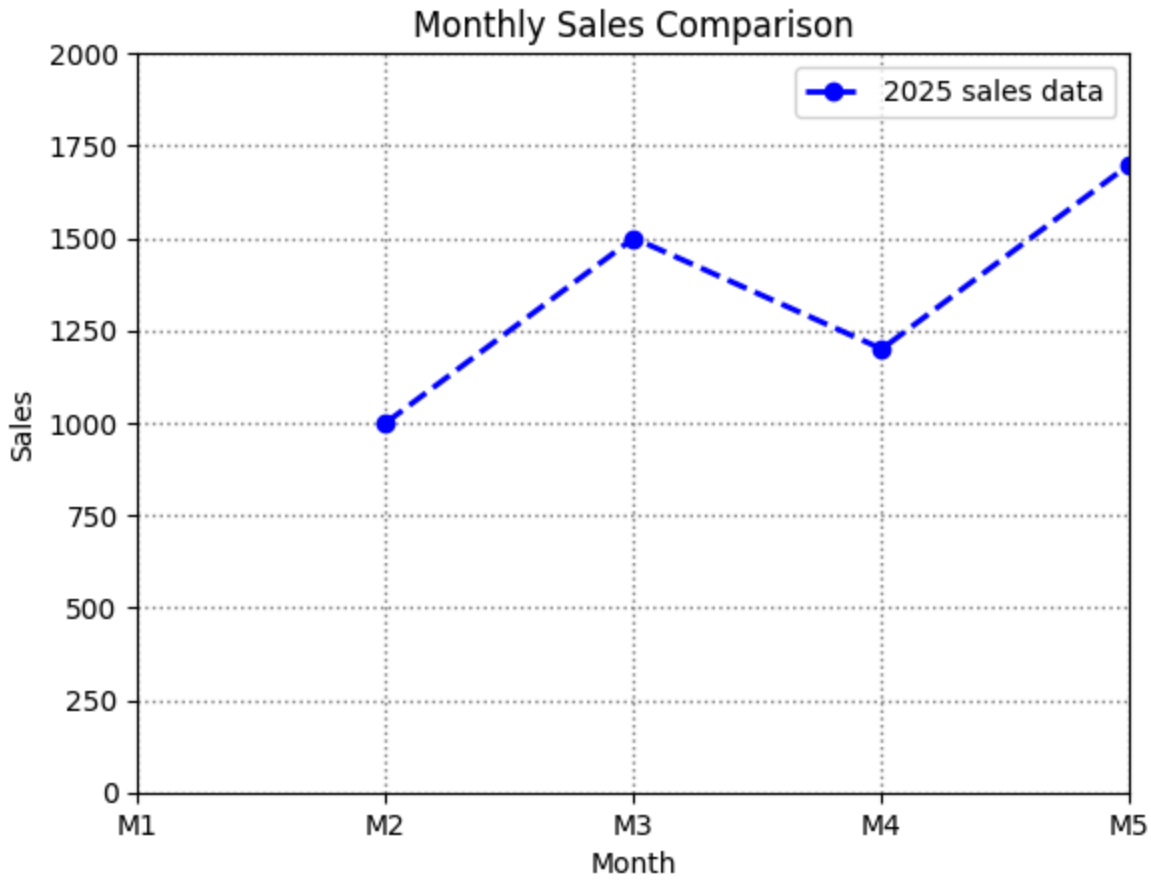
```
months = [1,2,3,4]
sales = [1000,1500,1200,1700]
plt.plot(months,sales, color= 'blue', linestyle='--', linewidth = 2, marker = 'o', label = '2025 sales data')
plt.xlabel('Month')
plt.ylabel('Sales')
plt.title('Monthly Sales Comparison')
plt.legend()
plt.grid(color='gray', linestyle = ':', linewidth =1)
plt.xlim(2,4)
plt.ylim(1200,1700)
plt.show()
```



use of xticks() and yticks() function.

it used to customize the tick marks on the x-axis and y-axis of a plot.

```
months = [1,2,3,4]
sales = [1000,1500,1200,1700]
plt.plot(months,sales, color= 'blue', linestyle='--', linewidth = 2, marker = 'o', label = '2025 sales data')
plt.xlabel('Month')
plt.ylabel('Sales')
plt.title('Monthly Sales Comparison')
plt.legend()
plt.grid(color='gray', linestyle = ':', linewidth =1)
plt.xlim(0,4)
plt.ylim(0,2000)
plt.xticks([0,1,2,3,4],['M1','M2','M3','M4','M5'])

plt.show()
```

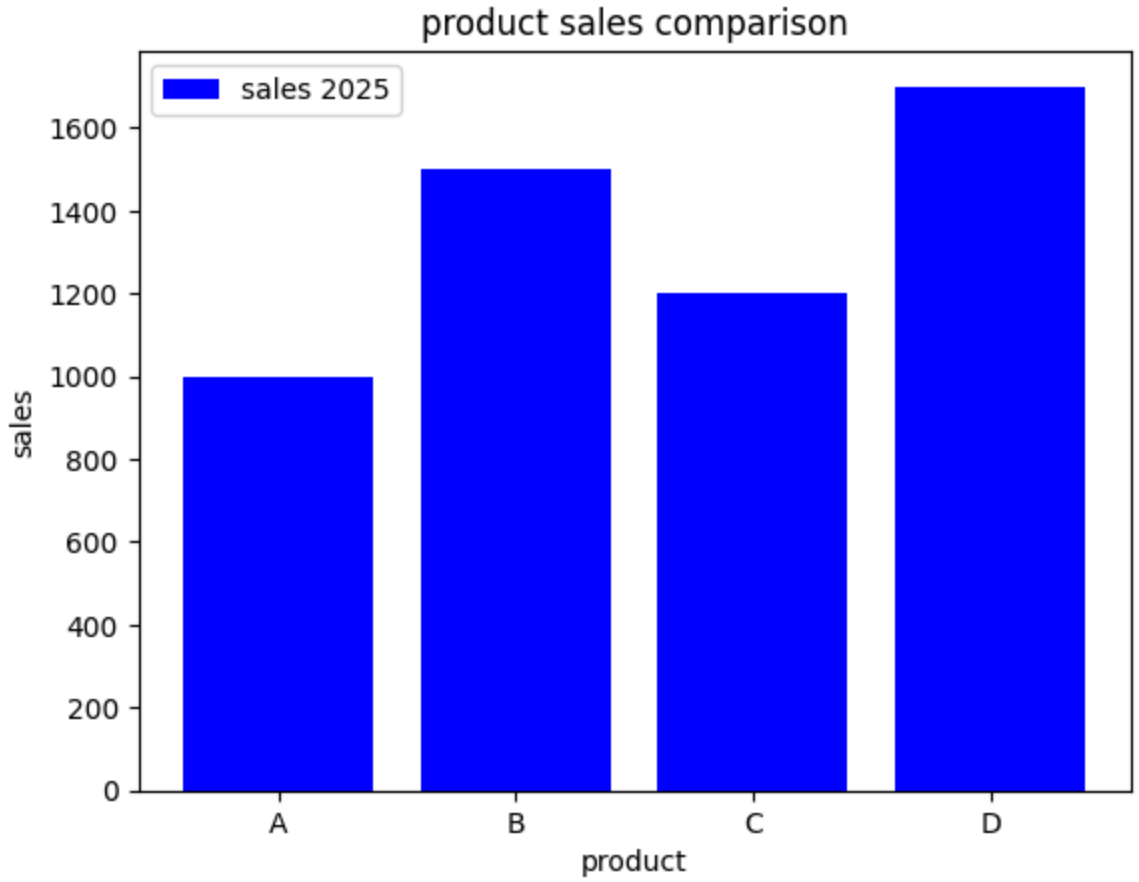


Data visualization tools and their purpose.

1. Bar chart(category comparison data analysis).

```
product = ['A','B','C','D']
sales = [1000,1500,1200,1700]
plt.bar(product,sales, color = 'blue', label = 'sales 2025')
plt.xlabel('product')
```

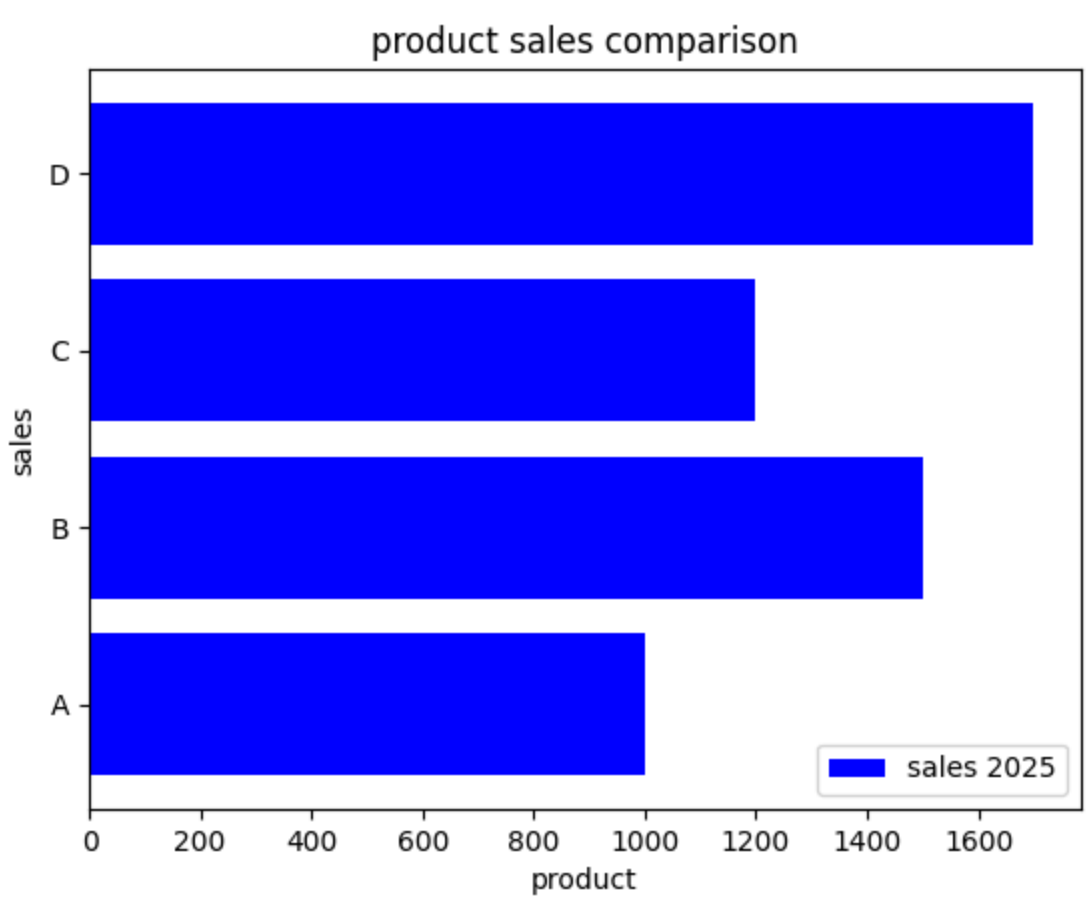
```
plt.ylabel('sales')
plt.title('product sales comparison')
plt.legend()
plt.show()
```



use barh() funtion to create a horizontal bar chart.

```
product = ['A','B','C','D']
sales = [1000,1500,1200,1700]
plt.barh(product,sales, color = 'blue', label = 'sales 2025')
plt.xlabel('product')
```

```
plt.ylabel('sales')
plt.title('product sales comparison')
plt.legend()
plt.show()
```

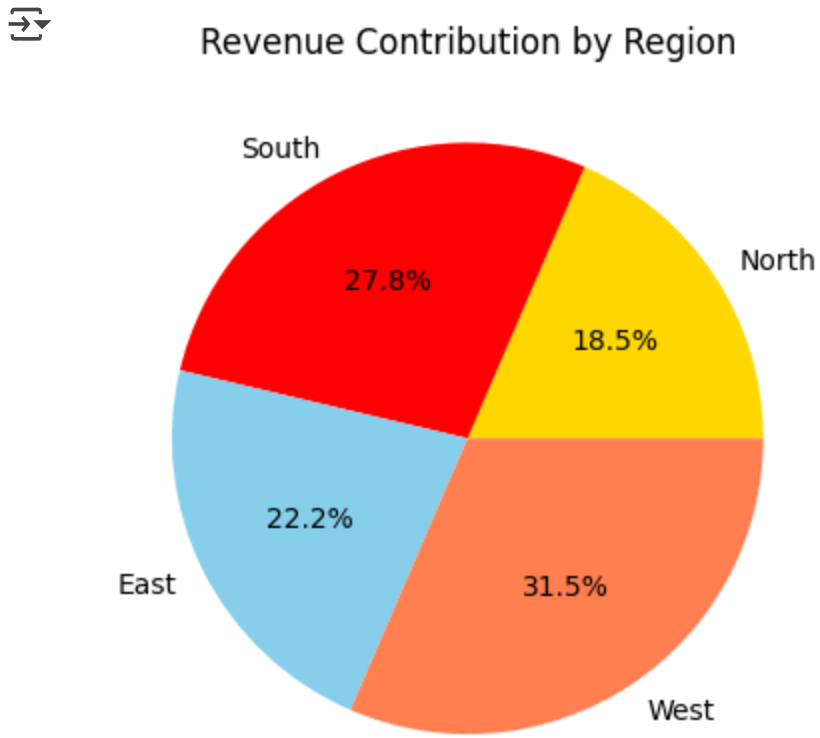


2.Pie chart(Proportion illustration whole reprensentation).

used to create a pie chart -- a circular chart divided into slice to show proportions or persentages.

```
regions = ['North','South','East','West']
revenue = [1000,1500,1200,1700]
```

```
plt.pie(revenue,labels=regions, autopct='%1.1f%%', colors = ['gold','red','skyblue','coral'] )
plt.title('Revenue Contribution by Region')
plt.show()
```

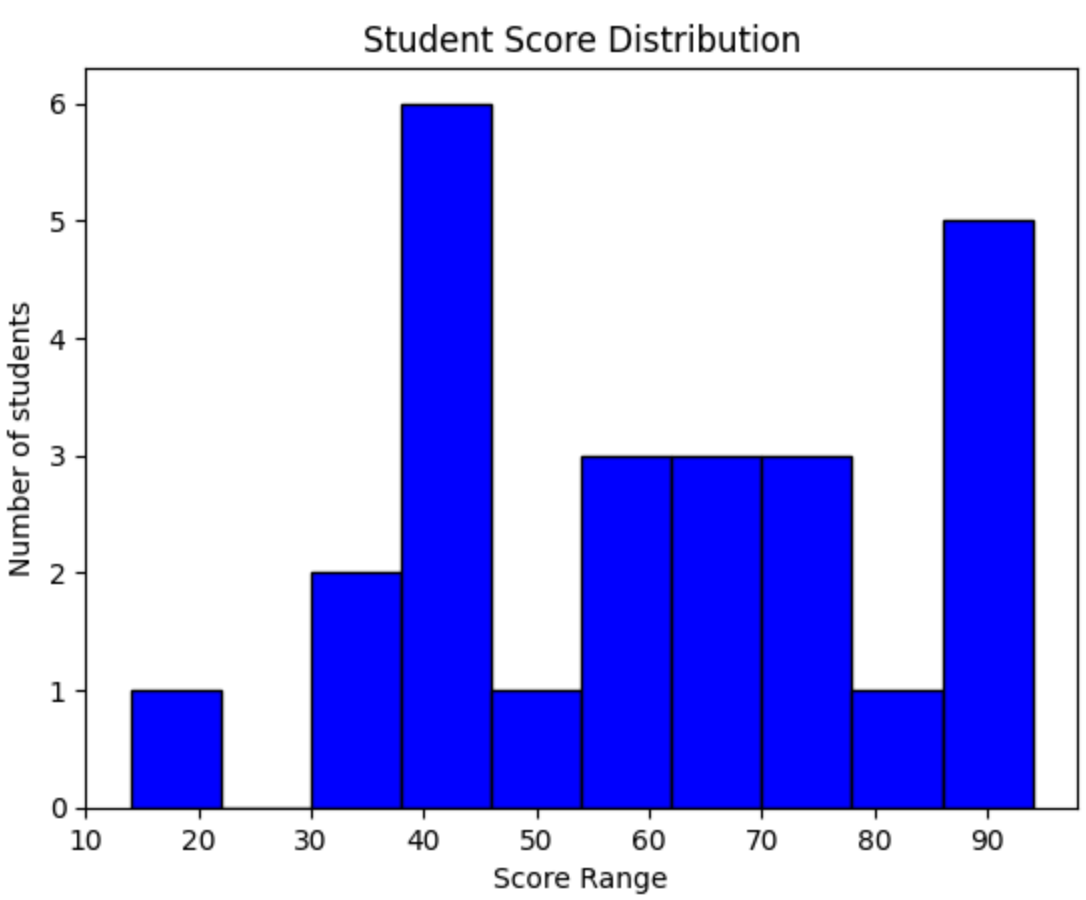


3.Histogram chart(Numerical Distribution Data Insight).

it's a plot the show distribution of numerical data by grouping value into bins(intervels) and counting how many data point fall into the bin.

```
scores = [45,60,90,77,44,67,87,65,45,77,88,87,94,45,56,44,78,34,66,33,44,55,76,14,50]
plt.hist(scores, bins=10, color='blue', edgecolor = 'black')
plt.xlabel('Score Range')
plt.ylabel('Number of students')
```

```
plt.title('Student Score Distribution')
plt.show()
```

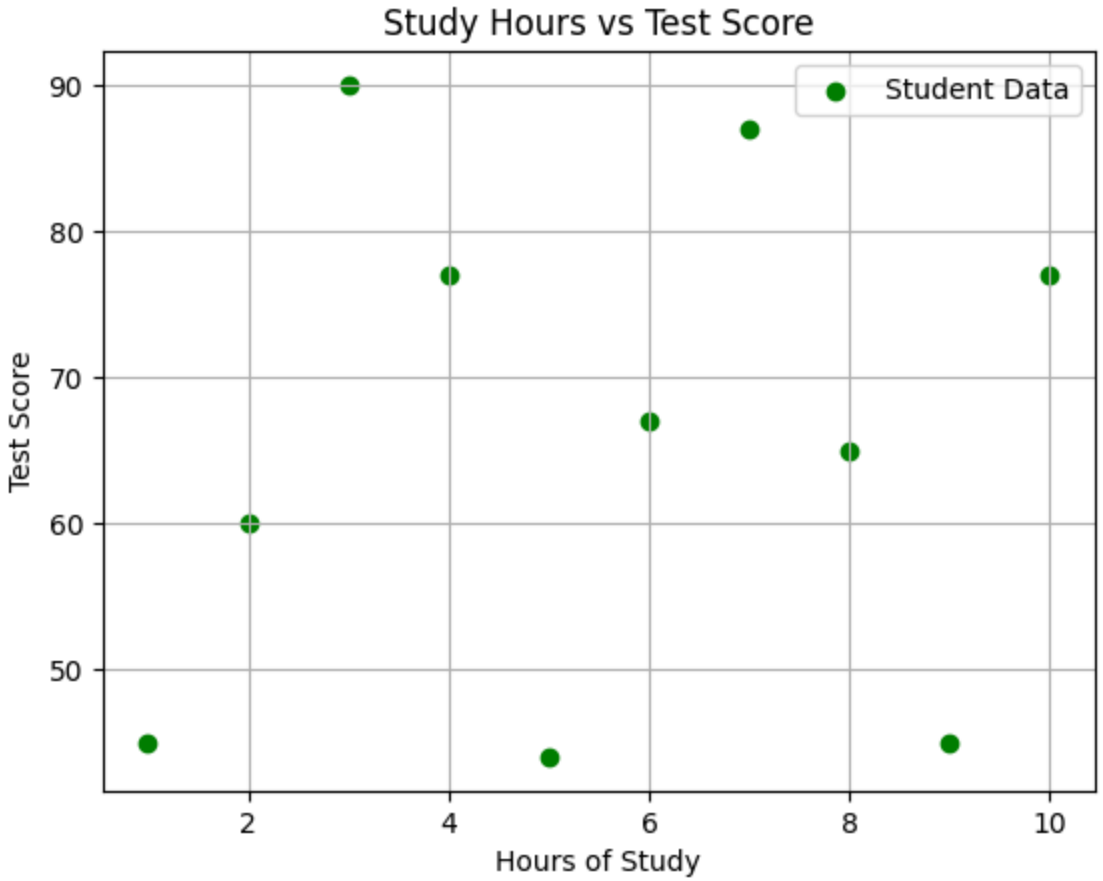


4.Scatter plot

it's used to display indivisula data point on a 2D plane ---- Specially useful for showing relationship or pattern two continues values.

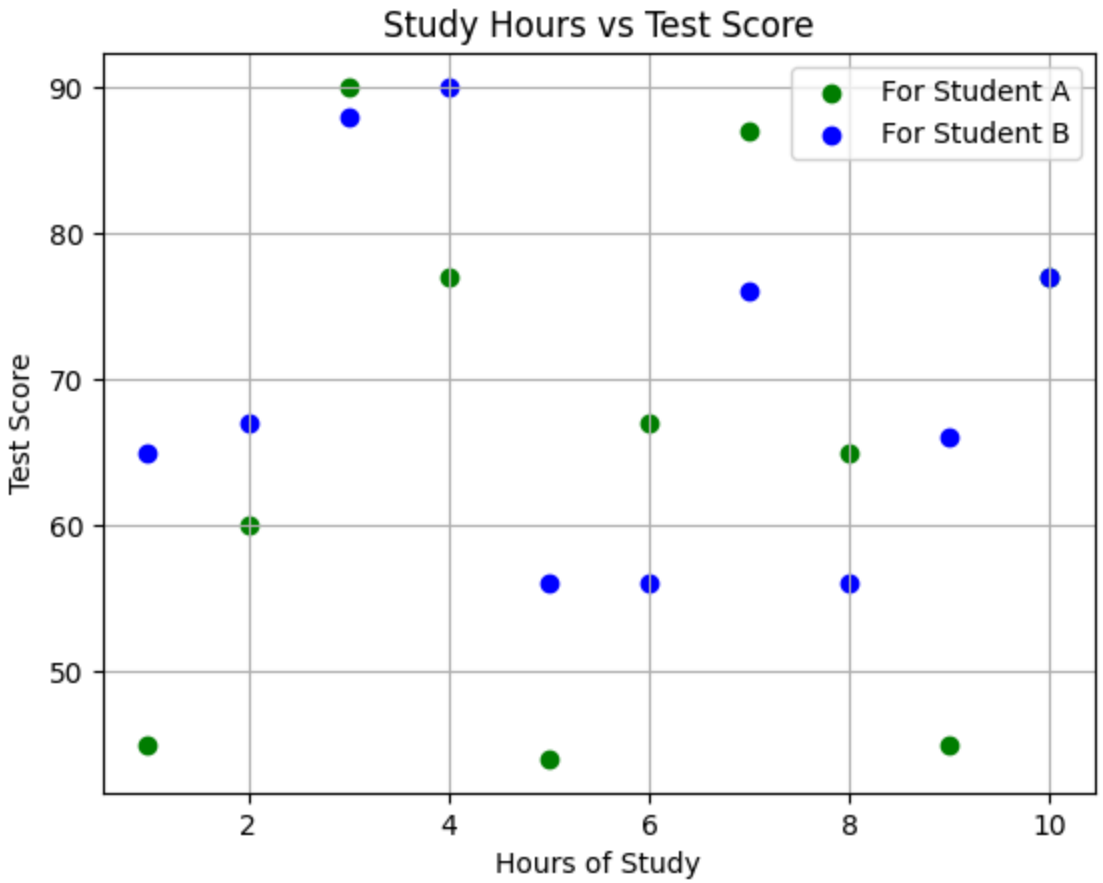
```
hours_studies = [1,2,3,4,5,6,7,8,9,10]
scores = [45,60,90,77,44,67,87,65,45,77]

# Scatter plot
```

```
plt.scatter(hours_studies,scores, color='green',marker = 'o', label = 'Student Data')
plt.xlabel('Hours of Study')
plt.ylabel('Test Score')
plt.title('Study Hours vs Test Score')
plt.legend()
plt.grid(True)
plt.show()
```



For plotting a comparison scatter plot between two students' study hours and test scores.

```
hours_studiesFor_Student_A = [1,2,3,4,5,6,7,8,9,10]
scores_For_student_A = [45,60,90,77,44,67,87,65,45,77]

hours_studiesFor_Student_B = [1,2,3,4,5,6,7,8,9,10]
scores_For_student_B = [65,67,88,90,56,56,76,56,66,77]

# Scatter plot
plt.scatter(hours_studiesFor_Student_A,scores_For_student_A, color='green',marker = 'o', label = 'For Student A')
plt.scatter(hours_studiesFor_Student_B,scores_For_student_B, color='Blue',marker = 'o', label = 'For Student B')

plt.xlabel('Hours of Study')
plt.ylabel('Test Score')
plt.title('Study Hours vs Test Score')
plt.legend()
plt.grid(True)
plt.show()
```

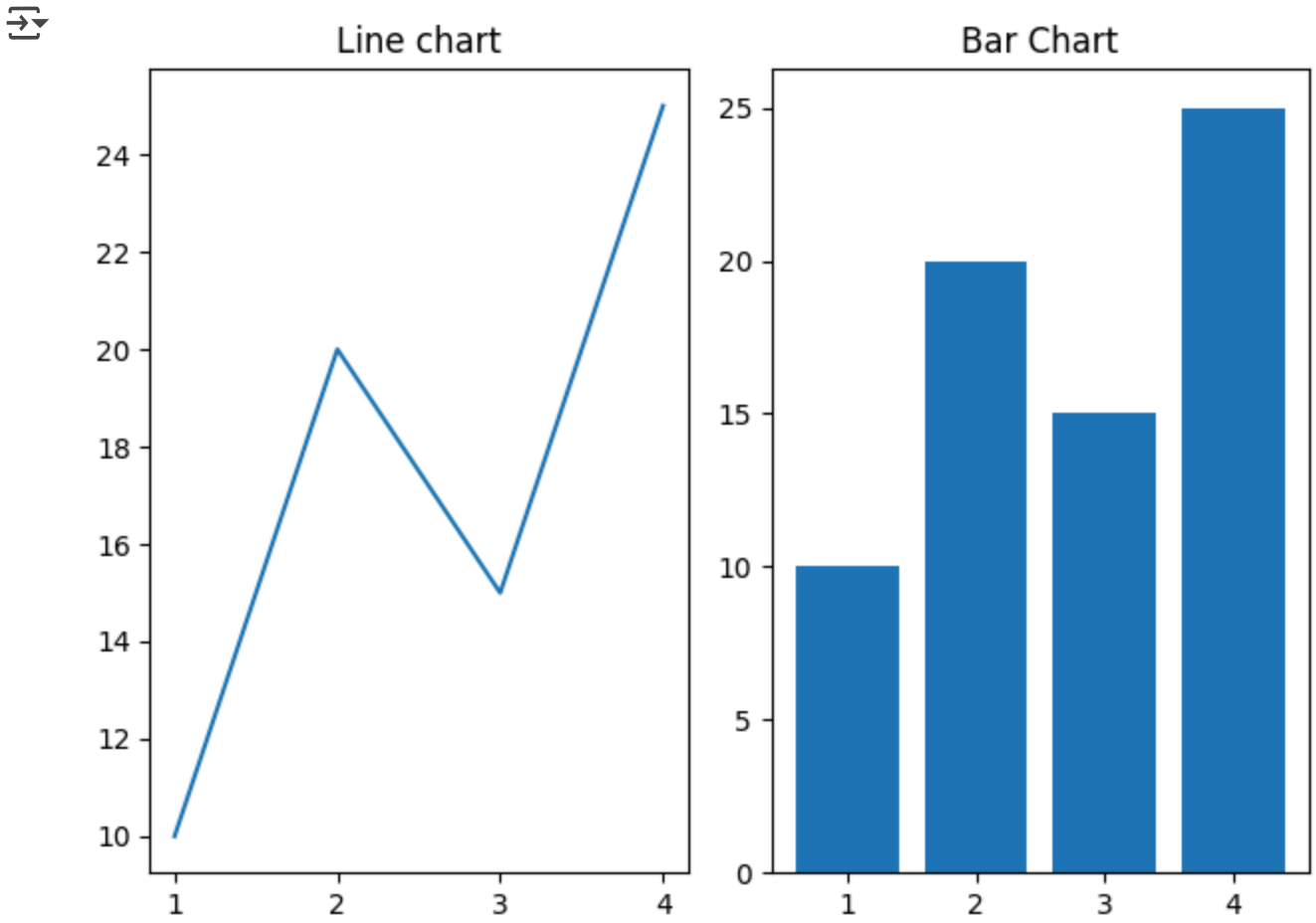


subplot() function.

used to create multiple plots (subplots) in a single figure.

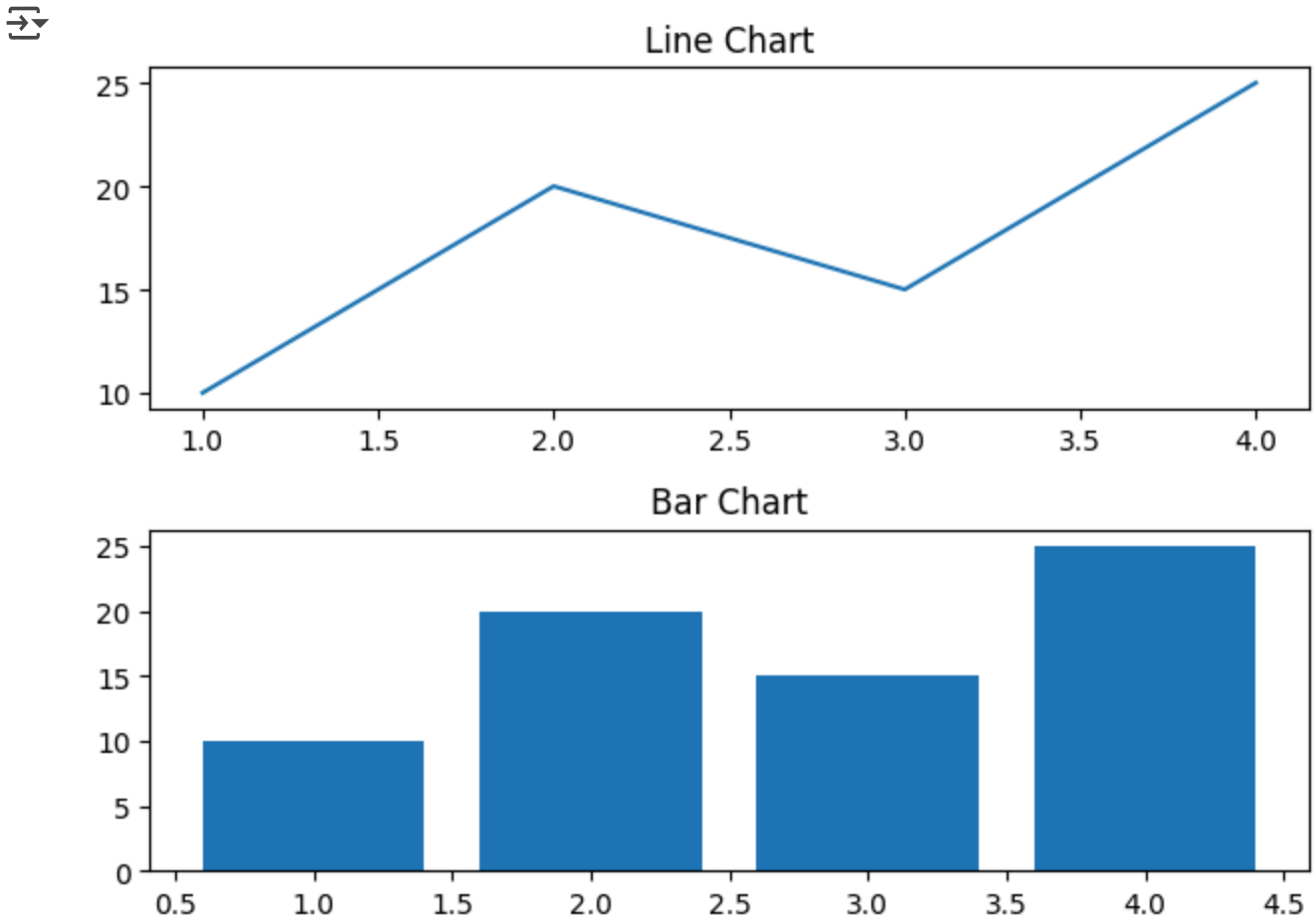
```
x = [1,2,3,4]
y = [10,20,15,25]

plt.subplot(1,2,1)
plt.plot(x,y)
plt.title('Line chart')
```

```
plt.subplot(1,2,2)
plt.bar(x,y)
plt.title('Bar Chart')

plt.tight_layout()
plt.show()
```



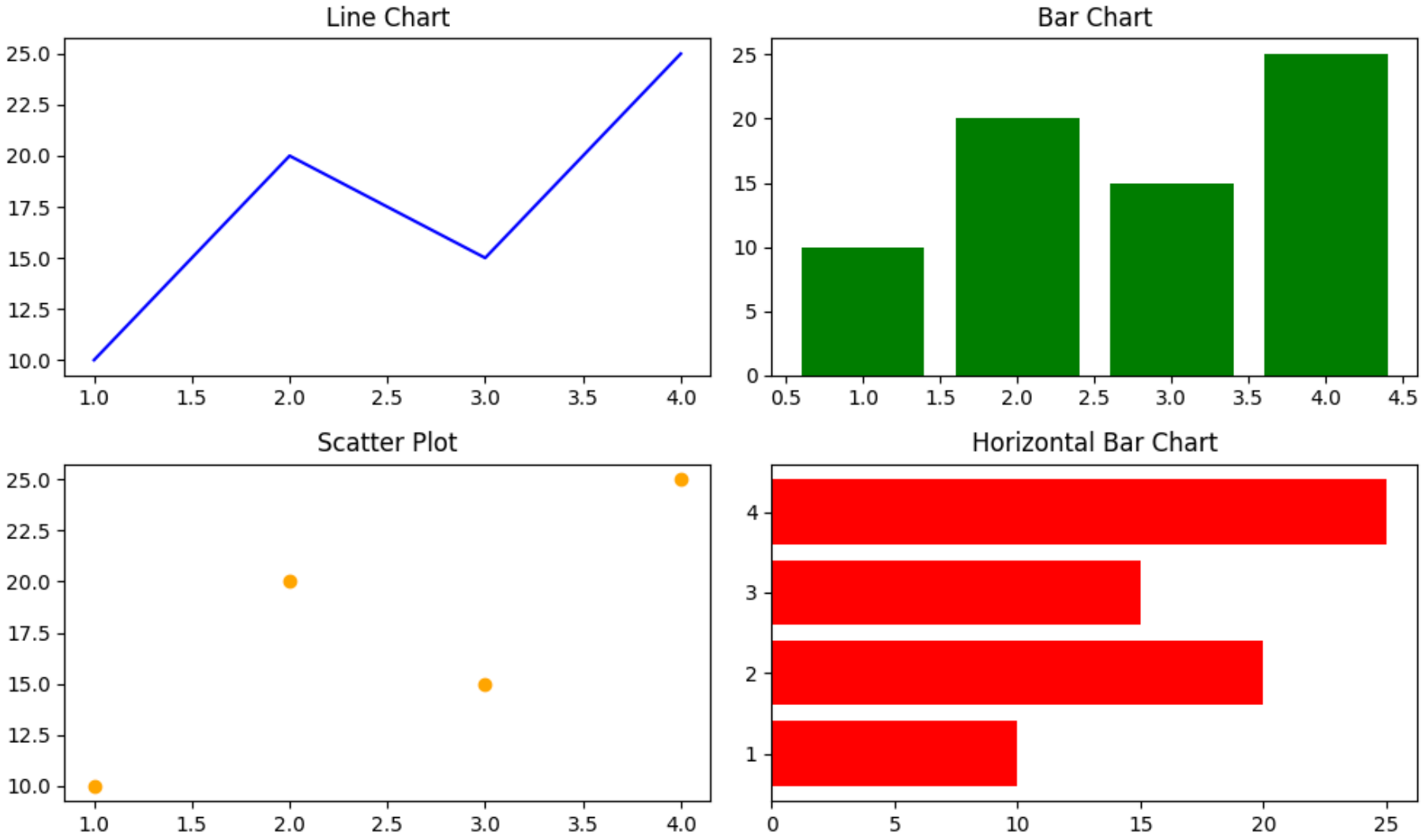

```
plt.subplot(2, 1, 1)  # 2 rows, 1 column, first plot
plt.plot(x, y)
plt.title('Line Chart')
```

```
plt.subplot(2, 1, 2)  # 2 rows, 1 column, second plot
plt.bar(x, y)
plt.title('Bar Chart')

plt.tight_layout()
plt.show()
```



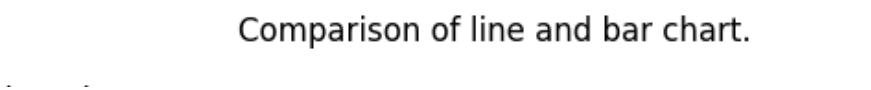

```
import matplotlib.pyplot as plt

x = [1, 2, 3, 4]
y = [10, 20, 15, 25]
```

```
plt.figure(figsize=(10, 6))  # Optional: Set figure size

# Plot 1: Line Chart
plt.subplot(2, 2, 1)
plt.plot(x, y, color='blue')
plt.title('Line Chart')

# Plot 2: Bar Chart
plt.subplot(2, 2, 2)
plt.bar(x, y, color='green')
plt.title('Bar Chart')

# Plot 3: Scatter Plot
plt.subplot(2, 2, 3)
plt.scatter(x, y, color='orange')
plt.title('Scatter Plot')

# Plot 4: Horizontal Bar Chart
plt.subplot(2, 2, 4)
plt.barh(x, y, color='red')
plt.title('Horizontal Bar Chart')

plt.tight_layout()
plt.show()
```



Uses Of Object oriented API And Creat multiple plot within a Window or Canvas.

```
x = [1, 2, 3, 4]
y = [10, 20, 15, 25]

# fig , ax = plt.subplots(nrows,ncols, figure =(width,height))
fig, ax = plt.subplots(1,2, figsize =(10,5))
ax[0].plot(x,y, color = 'blue')
ax[0].set_title('Line Chart')
ax[1].bar(x,y, color = 'green')
ax[1].set_title('Bar Chart')
fig.suptitle('Comparison of line and bar chart.')
plt.tight_layout()
plt.show()
```


Comparison of line and bar chart.