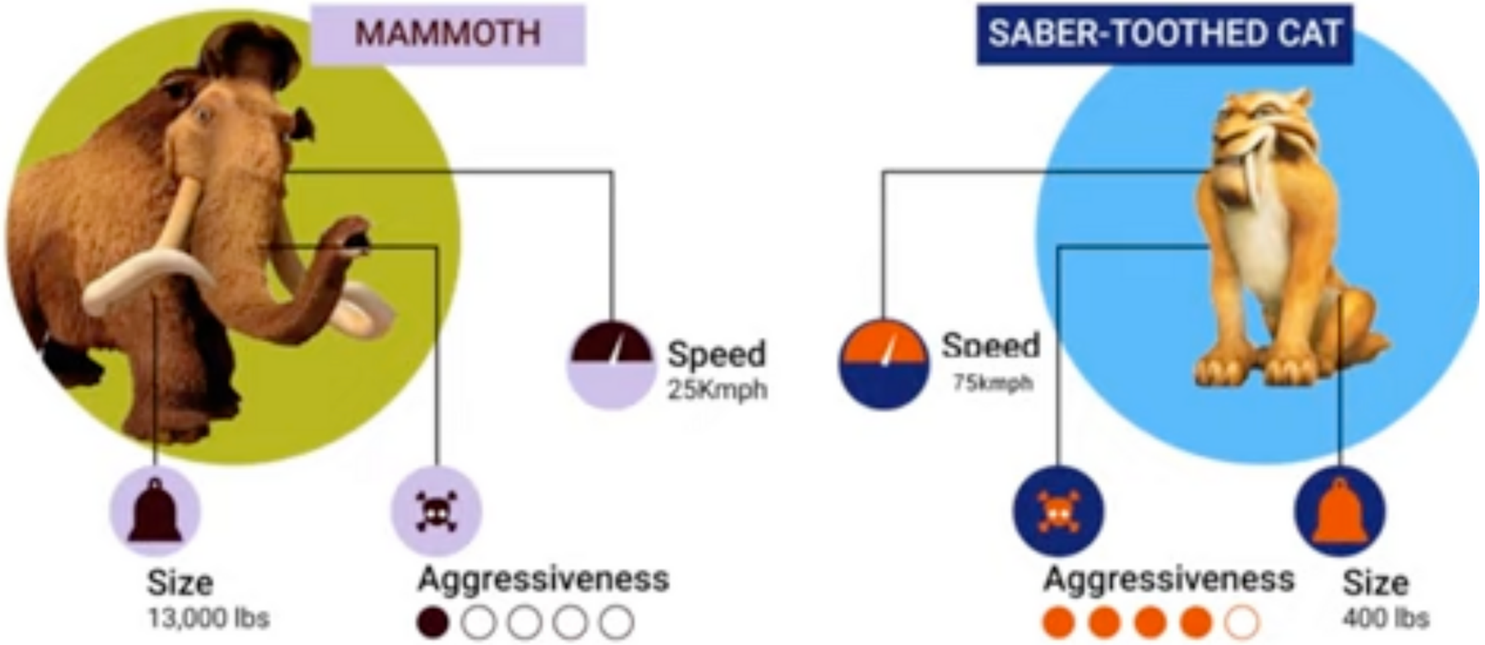
Data Visualization

Name of Animal	Speed	Aggressiveness	Size
Mammoth	25	low	13000lbs
Saber-Tooth Cat	75	High	400lbs

Data



Data Visualization

Why do we need Data Visualization?

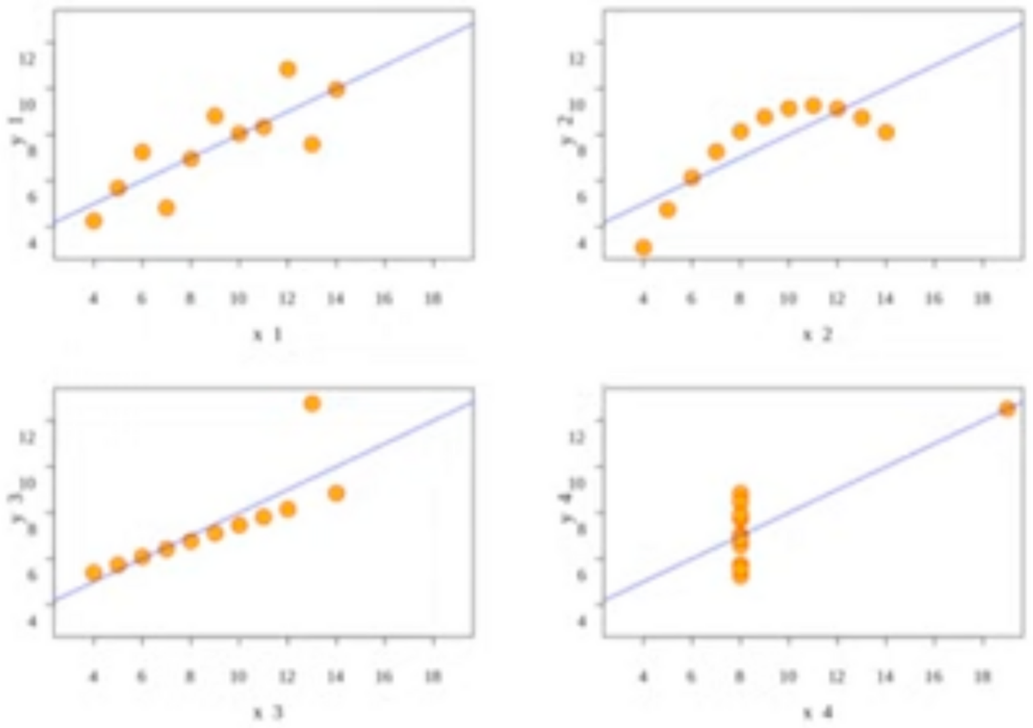
Anscombe's Quartet

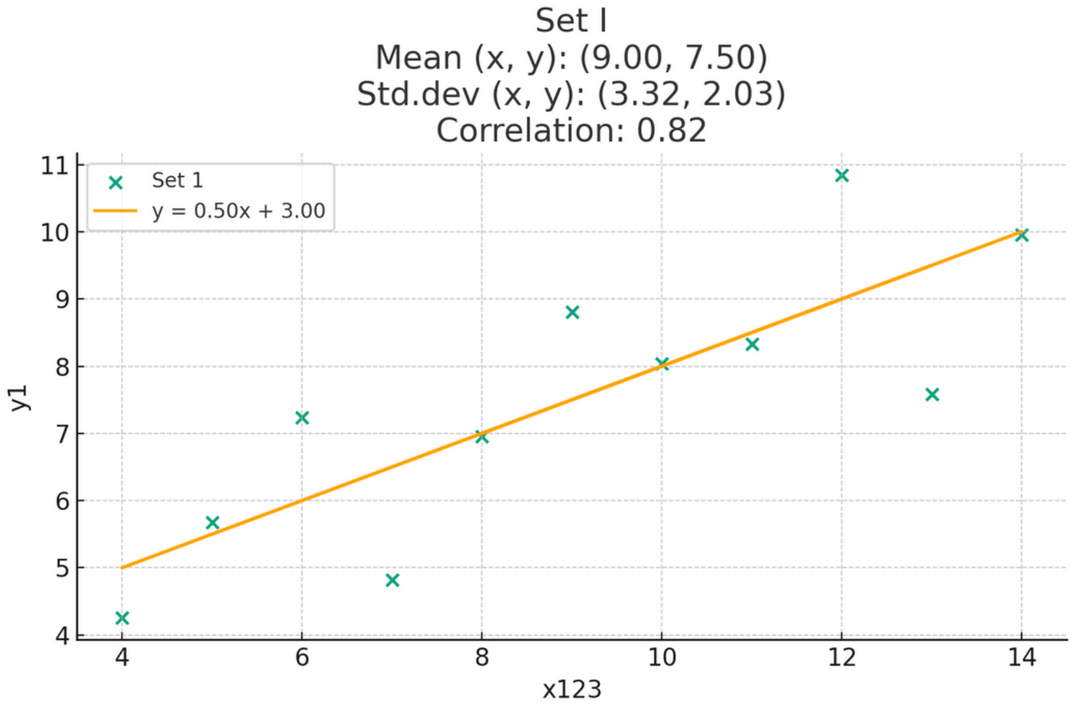
	I		II		III		IV	
	x	y	x	y	x	y	x	y
	10	8,04	10	9,14	10	7,46	8	6,58
	8	6,95	8	8,14	8	6,77	8	5,76
	13	7,58	13	8,74	13	12,74	8	7,71
	9	8,81	9	8,77	9	7,11	8	8,84
	11	8,33	11	9,26	11	7,81	8	8,47
	14	9,96	14	8,1	14	8,84	8	7,04
	6	7,24	6	6,13	6	6,08	8	5,25
	4	4,26	4	3,1	4	5,39	19	12,5
	12	10,84	12	9,13	12	8,15	8	5,56
	7	4,82	7	7,26	7	6,42	8	7,91
	5	5,68	5	4,74	5	5,73	8	6,89
SUM	99,00	82,51	99,00	82,51	99,00	82,50	99,00	82,51
AVG	9,00	7,50	9,00	7,50	9,00	7,50	9,00	7,50
STDEV	3,32	2,03	3,32	2,03	3,32	2,03	3,32	2,03

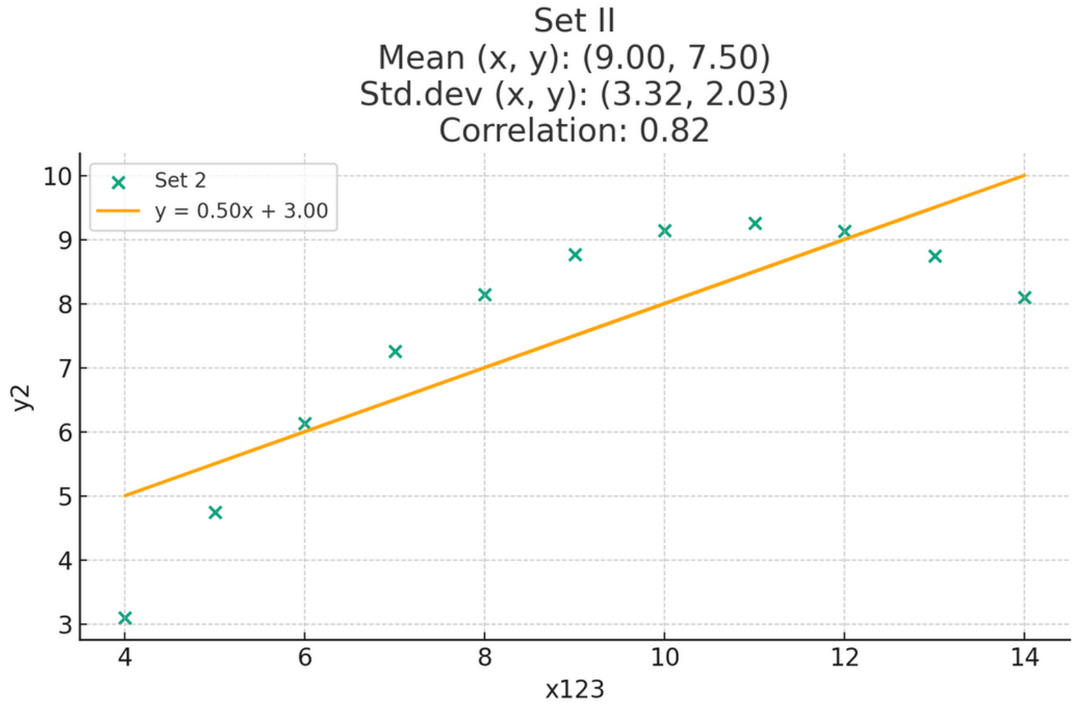
Why do we need Data Visualization?

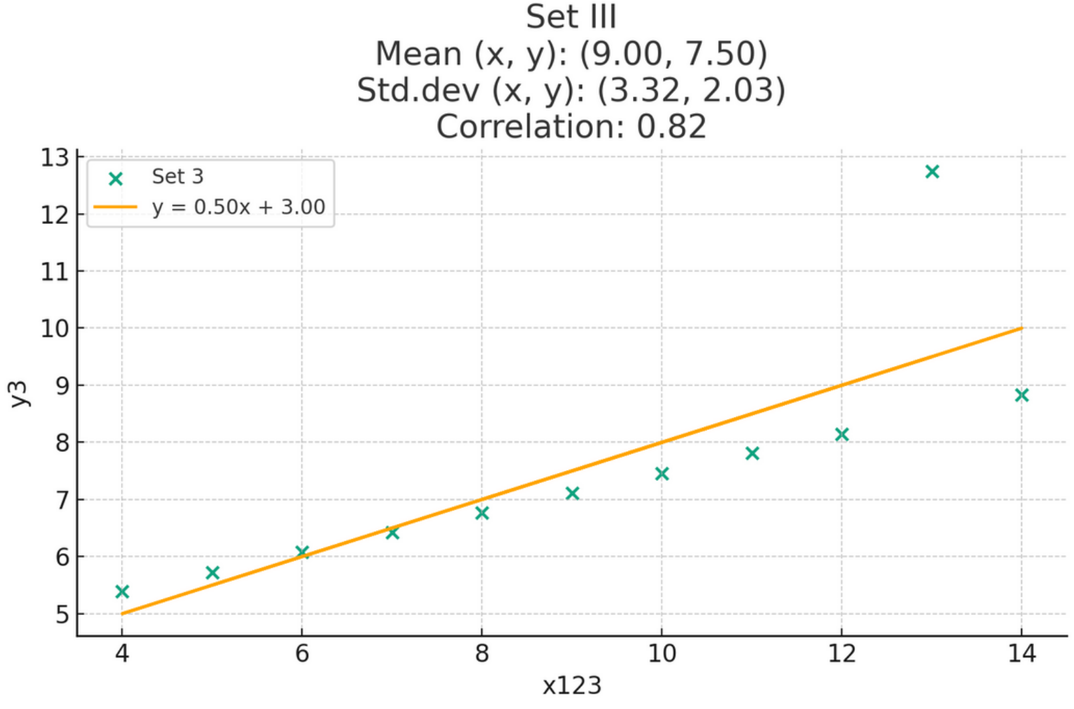
Anscombe's Quartet

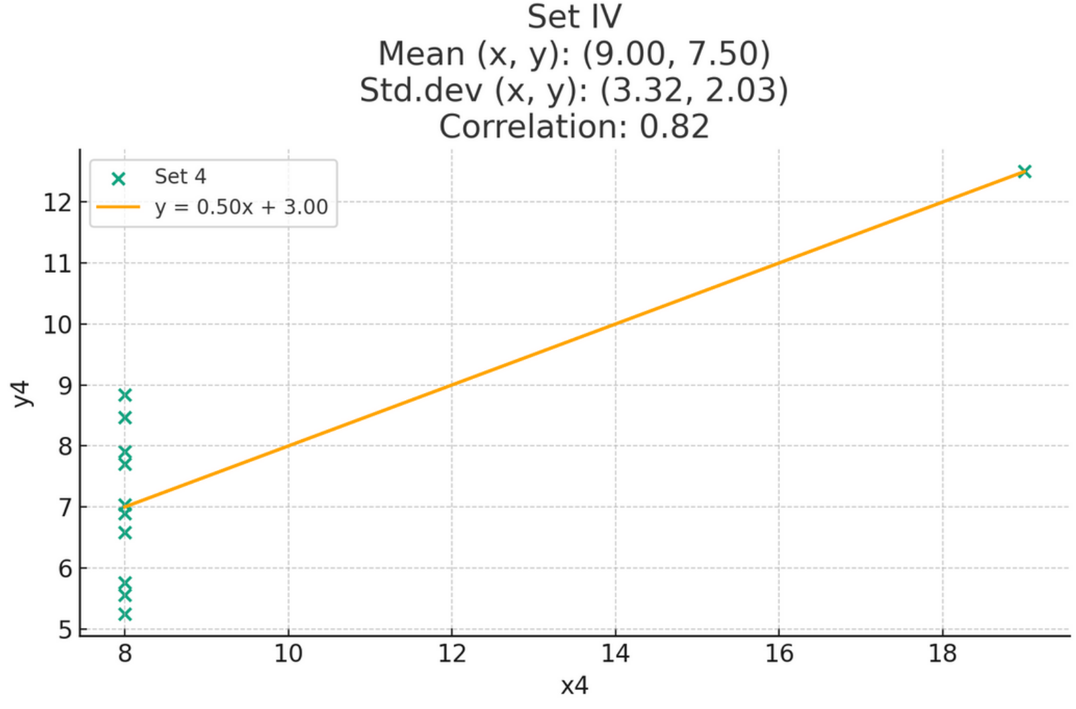
I		II		III		IV	
x	y	x	y	x	y	x	y
10	8,04	10	9,14	10	7,46	8	6,58
8	6,95	8	8,14	8	6,77	8	5,76
13	7,58	13	8,74	13	12,74	8	7,71
9	8,81	9	8,77	9	7,11	8	8,84
11	8,33	11	9,26	11	7,81	8	8,47
14	9,96	14	8,1	14	8,84	8	7,04
6	7,24	6	6,13	6	6,08	8	5,25
4	4,26	4	3,1	4	5,39	19	12,5
12	10,84	12	9,13	12	8,15	8	5,56
7	4,82	7	7,26	7	6,42	8	7,91
5	5,68	5	4,74	5	5,73	8	6,89
SUM	99,00	82,51	99,00	82,51	99,00	82,50	99,00	82,51
AVG	9,00	7,50	9,00	7,50	9,00	7,50	9,00	7,50
STDEV	3,32	2,03	3,32	2,03	3,32	2,03	3,32	2,03




Set I
Mean (x, y): (9.00, 7.50)
Std.dev (x, y): (3.32, 2.03)
Correlation: 0.82
Set 1
y = 0.50x + 3.00
y1
x123


Set II
Mean (x, y): (9.00, 7.50)
Std.dev (x, y): (3.32, 2.03)
Correlation: 0.82
Set 2
y = 0.50x + 3.00
y2
x123


Set III
Mean (x, y): (9.00, 7.50)
Std.dev (x, y): (3.32, 2.03)
Correlation: 0.82
Set 3
y = 0.50x + 3.00
y3
x123


Set IV
Mean (x, y): (9.00, 7.50)
Std.dev (x, y): (3.32, 2.03)
Correlation: 0.82
Set 4
y = 0.50x + 3.00
y4
x4

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

# Load the dataset
file_path = 'path_to_your_file.csv'  # Replace with your file path
anscombe_data = pd.read_csv(file_path)

# Initialize the subplots for the Anscombe's quartet visualization
fig, axs = plt.subplots(2, 2, figsize=(15, 10), sharex=False, sharey=False)
axs = axs.flatten()  # Flatten to 1D for easy indexing

# Titles for the subplots
titles = ['Set I', 'Set II', 'Set III', 'Set IV']

# Statistical indicators
means = anscombe_data.mean()
stds = anscombe_data.std()
corrs = [anscombe_data[['x123', 'y1']].corr().iloc[0, 1], anscombe_data[['x123', 'y2']].corr().iloc[0, 1],
        anscombe_data[['x123', 'y3']].corr().iloc[0, 1], anscombe_data[['x4', 'y4']].corr().iloc[0, 1]]

# Iterate over the 4 datasets
for index, ax in enumerate(axs):
    # For the x values, use 'x123' for sets I-III, and 'x4' for set IV
    x = 'x123' if index < 3 else 'x4'
    y = f'y{index+1}'

    # Scatter plot
    ax.scatter(anscombe_data[x], anscombe_data[y], label=f'Set {index+1}')

    # Calculating the linear regression line
    m, b = np.polyfit(anscombe_data[x], anscombe_data[y], 1)
    ax.plot(anscombe_data[x], m*anscombe_data[x] + b, color='orange', label=f'y = {m:.2f}x + {b:.2f}')

    # Set titles and labels with statistical info
    ax.set_title(f'{titles[index]}\nMean (x, y): ({means[x]:.2f}, {means[y]:.2f})\n'
                 f'Std.dev (x, y): ({stds[x]:.2f}, {stds[y]:.2f})\n'
                 f'Correlation: {corrs[index]:.2f}')
    ax.set_xlabel('x')
    ax.set_ylabel('y')
    ax.legend()

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

- Now we have the visualizations for the Anscombe's quartet with linear regression lines included. Each subplot corresponds to a different dataset within the quartet:

- Set I (Top Left)
- Set II (Top Right)
- Set III (Bottom Left)
- Set IV (Bottom Right)
- In each plot, the orange line represents the best fit line obtained through linear regression, highlighting the linear relationship between the x and y variables within each set.

- The visualizations underscore the importance of graphing data to uncover underlying patterns that are not evident from statistics alone. While all four sets have nearly identical descriptive statistics, their graphical representations reveal distinctly different distributions. For instance:

- Set I shows a typical linear relationship with some random scatter.
- Set II shows a curve, indicating that a linear model might not be appropriate.
- Set III shows a linear relationship except for an outlier that exerts a strong influence on the regression line.
- Set IV demonstrates how a single outlier can dramatically affect the slope of the regression line, even if the rest of the points have no relationship at all.
- This is a classic illustration of why data visualization is crucial: it helps to detect anomalies, patterns, or relationships that a simple statistical summary might miss

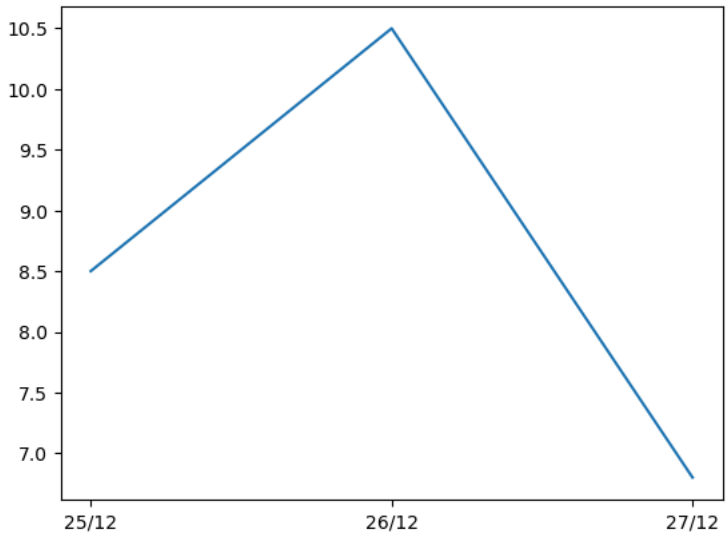
Data Visualization Libraries

ggplot

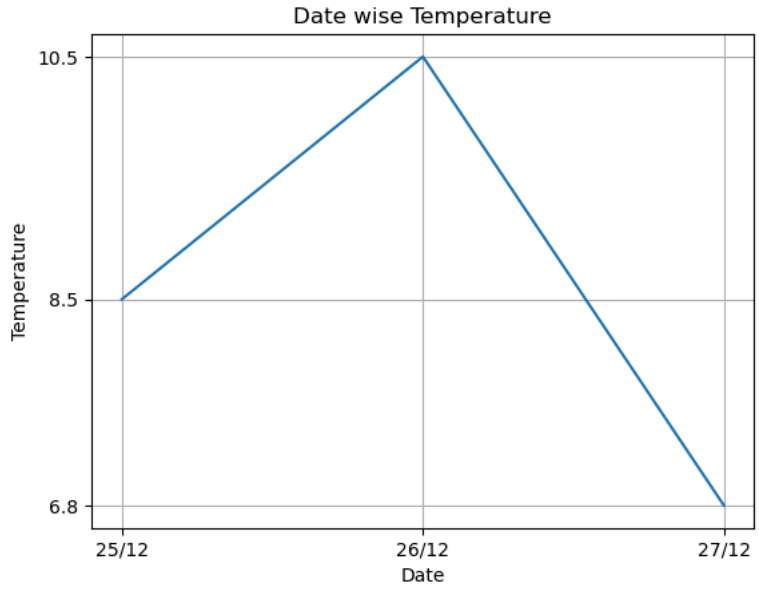
seaborn

geoplotlib

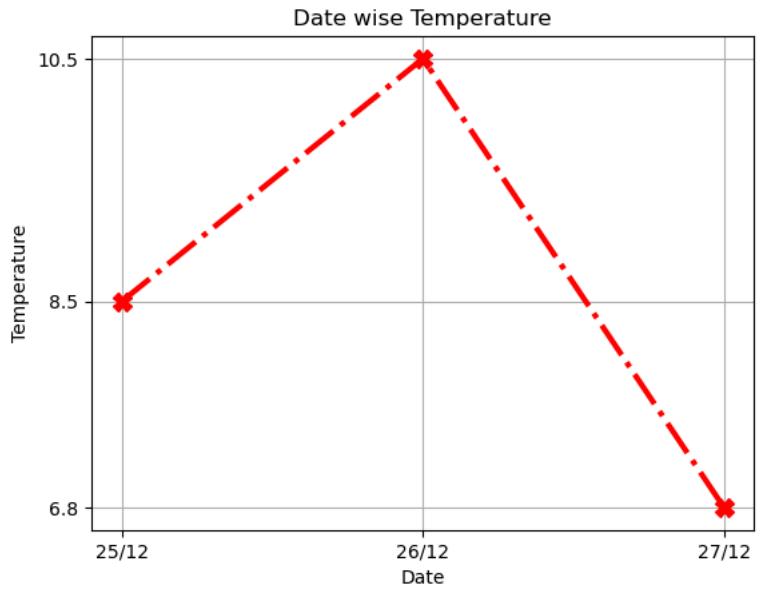
```
import matplotlib.pyplot as plt
#from matplotlib import pyplot as plt


#list storing date in string format
date=["25/12","26/12","27/12"]

#list storing temperature values
temp=[8.5,10.5,6.8]


#create a figure plotting temp versus date
plt.plot(date, temp)

#save the figure to the folder
plt.savefig('x.png')

#show the figure
plt.show()
```



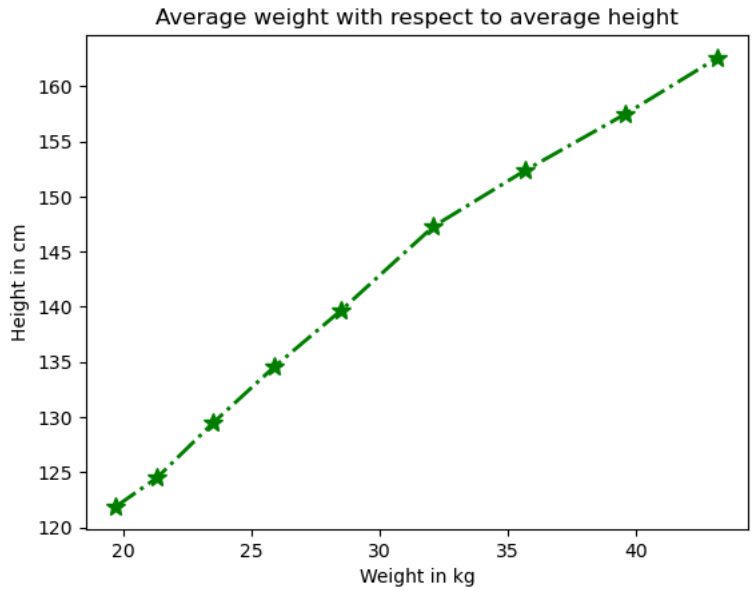

```
import matplotlib.pyplot as plt

date=["25/12","26/12","27/12"]
temp=[8.5,10.5,6.8]


plt.plot(date, temp)

plt.xlabel("Date") #add the Label on x-axis
plt.ylabel("Temperature") #add the Label on y-axis
plt.title("Date wise Temperature") #add the title to the chart
plt.grid(True) #add gridlines to the background

plt.yticks(temp)
plt.show()
```



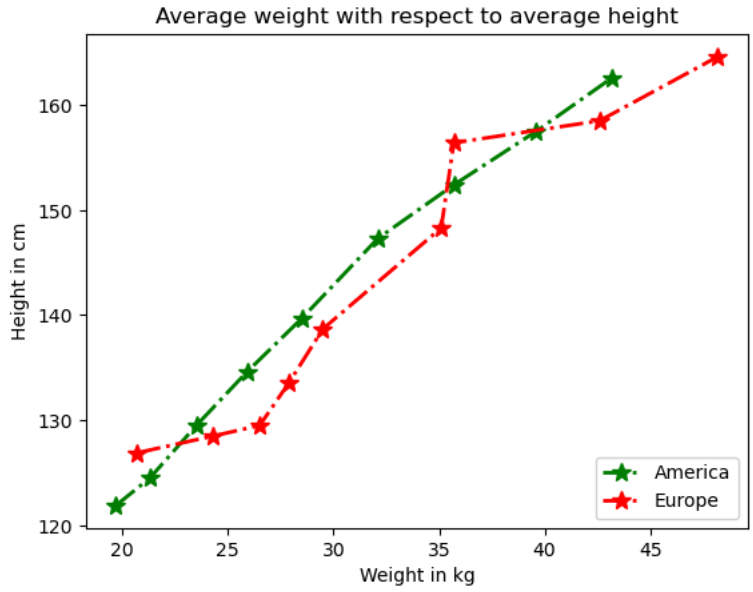

```
import matplotlib.pyplot as plt

date=["25/12","26/12","27/12"]
temp=[8.5,10.5,6.8]


plt.plot(date, temp,marker='X',markersize=10,color='red',linewidth=3, linestyle='dashdot')

plt.xlabel("Date") #add the Label on x-axis
plt.ylabel("Temperature") #add the Label on y-axis
plt.title("Date wise Temperature") #add the title to the chart
plt.grid(True) #add gridlines to the background

plt.yticks(temp)
plt.show()
```



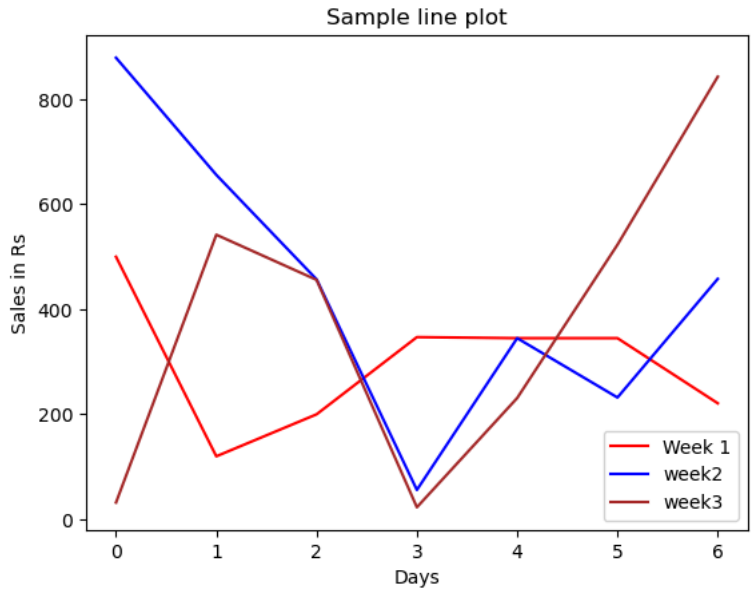
```
import matplotlib.pyplot as plt
import pandas as pd

height=[121.9,124.5,129.5,134.6,139.7,147.3,152.4,157.5,162.6]
weight=[19.7,21.3,23.5,25.9,28.5,32.1,35.7,39.6,43.2]

#df=pd.DataFrame({"height":height,"weight":weight})

#Set xlabel for the plot
plt.xlabel('Weight in kg')

#Set ylabel for the plot
plt.ylabel('Height in cm')

#Set chart title:
plt.title('Average weight with respect to average height')

#plot using marker'-*' and line colour as green
plt.plot(weight,height,marker='*',markersize=10,color='green',linewidth=2, linestyle='dashdot')

plt.show()
```




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

#height and weight in america
height=[121.9,124.5,129.5,134.6,139.7,147.3,152.4,157.5,162.6]
weight=[19.7,21.3,23.5,25.9,28.5,32.1,35.7,39.6,43.2]
plt.plot(weight,height,marker='*',markersize=10,color='green',linewidth=2, linestyle='dashdot')

#height and weight in europe
height=[126.9,128.5,129.5,133.6,138.7,148.3,156.4,158.5,164.6]
weight=[20.7,24.3,26.5,27.9,29.5,35.1,35.7,42.6,48.2]
plt.plot(weight,height,marker='*',markersize=10,color='red',linewidth=2, linestyle='dashdot')
#df=pd.DataFrame({"height":height,"weight":weight})

#Set xlabel for the plot
plt.xlabel('Weight in kg')

#Set ylabel for the plot
plt.ylabel('Height in cm')

#Set chart title:
plt.title('Average weight with respect to average height')



plt.legend(["America", "Europe"], loc ="lower right")
plt.show()
```




```
#How to plot data using Pandas
import pandas as pd
import matplotlib.pyplot as plt

# reads "Example_1.csv" to df by giving path to the file
df=pd.read_csv("Example_1.csv")

#create a line plot of different color for each week
df.plot(kind='line', color=['red','blue','brown'])

# Set title to "Mela Sales Report"
plt.title('Sample line plot')

# Label x axis as "Days"
plt.xlabel('Days')

# Label y axis as "Sales in Rs"
plt.ylabel('Sales in Rs')

#Display the figure
plt.show()
```

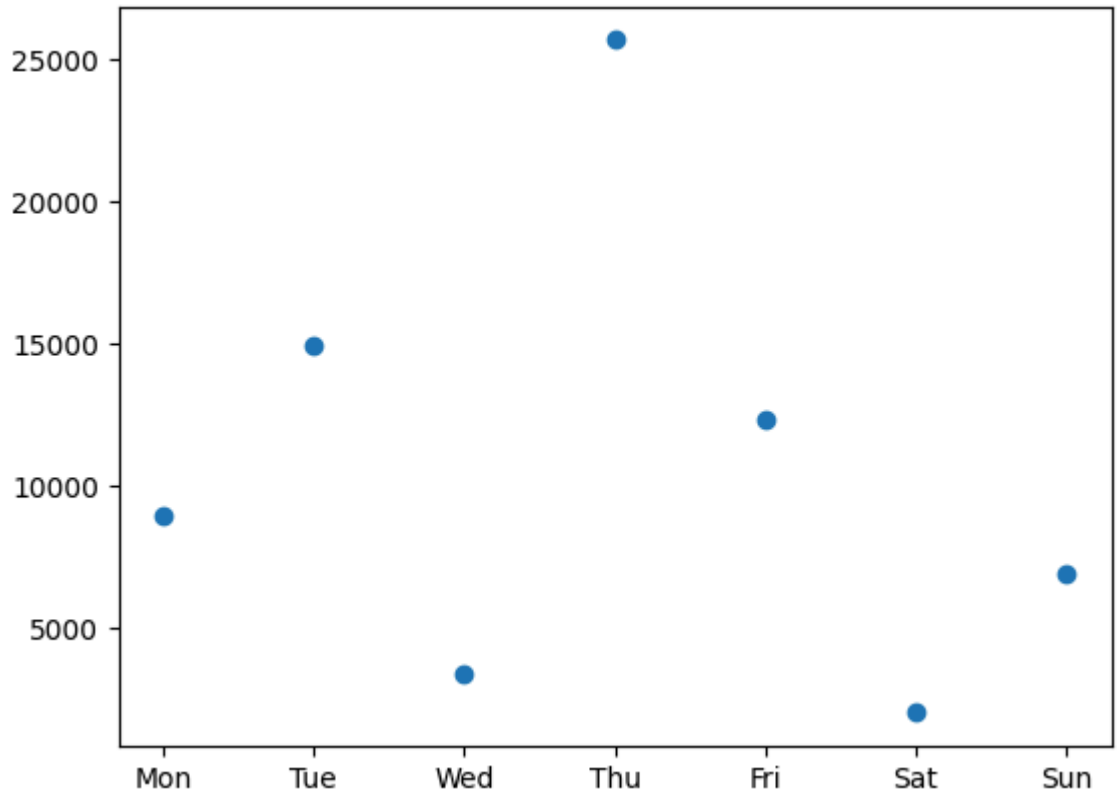


```
import pandas as pd
import matplotlib.pyplot as plt

# reads "Example_1.csv" to df by giving path to the file
df=pd.read_csv("/content/Example_1.csv")

#create a line plot of different color for each week
df.plot(kind='line', color=['red','blue','brown'],marker="*",markersize=10,linewidth=3,linestyle="--")

# Set title to "Mela Sales Report"
plt.title('Sample line plot')

# Label x axis as "Days"
plt.xlabel('Days')

# Label y axis as "Sales in Rs"
plt.ylabel('Sales in Rs')

#Display the figure
plt.show()
```

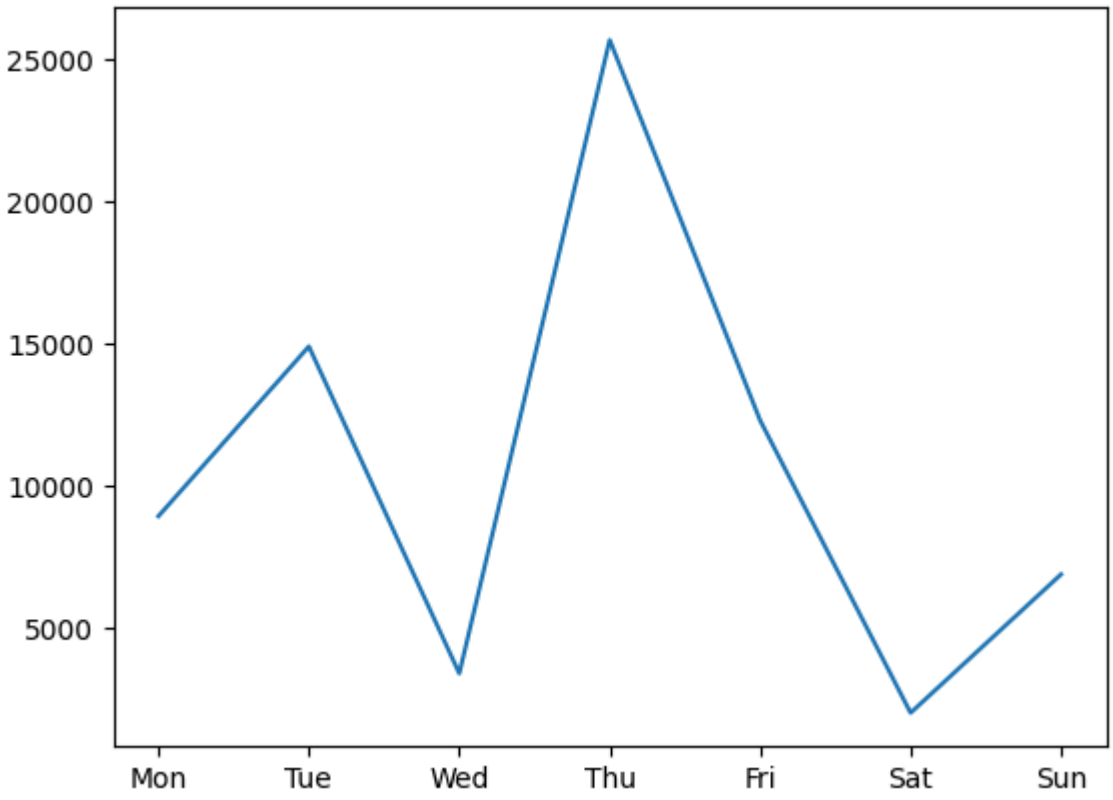
Import the required libraries

```
import matplotlib.pyplot as plt
import numpy as np
```

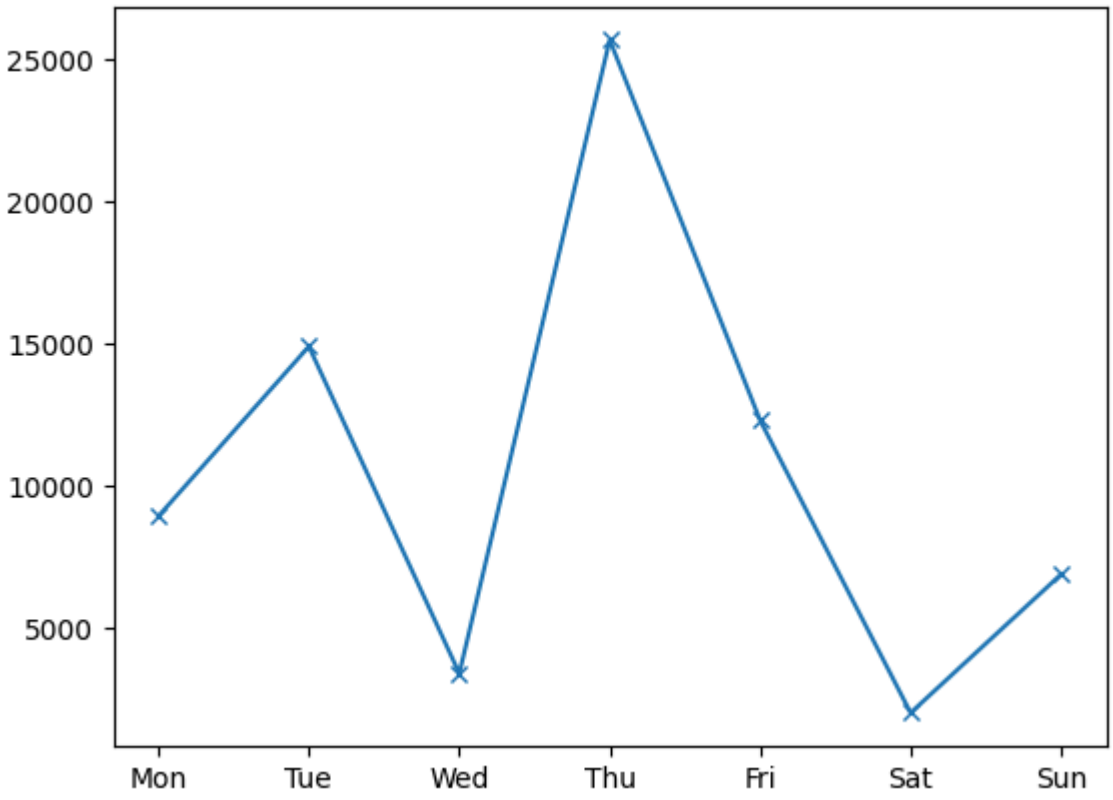
```
days = ["Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun"]
steps_walked = [8934, 14902, 3409, 25672, 12300, 2023, 6890]
plt.plot(days, steps_walked, "o")
plt.show()
```



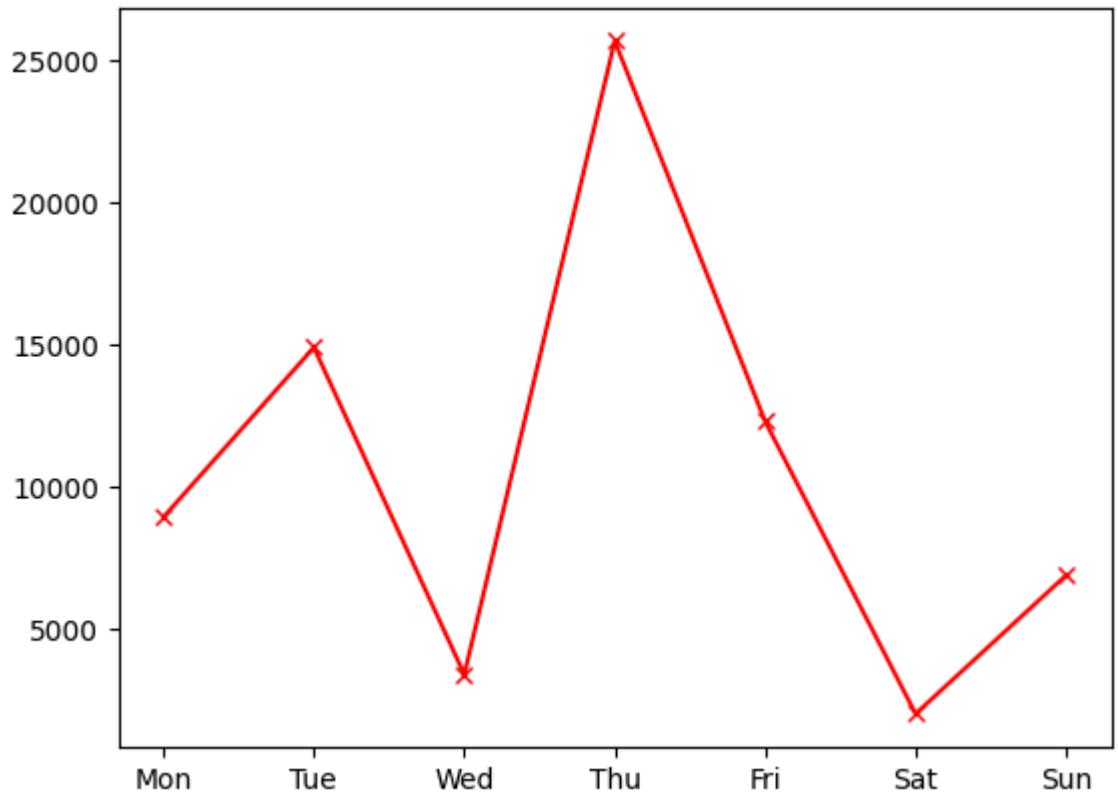

```
days = ["Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun"]
steps_walked = [8934, 14902, 3409, 25672, 12300, 2023, 6890]
plt.plot(days, steps_walked)
plt.show()
```



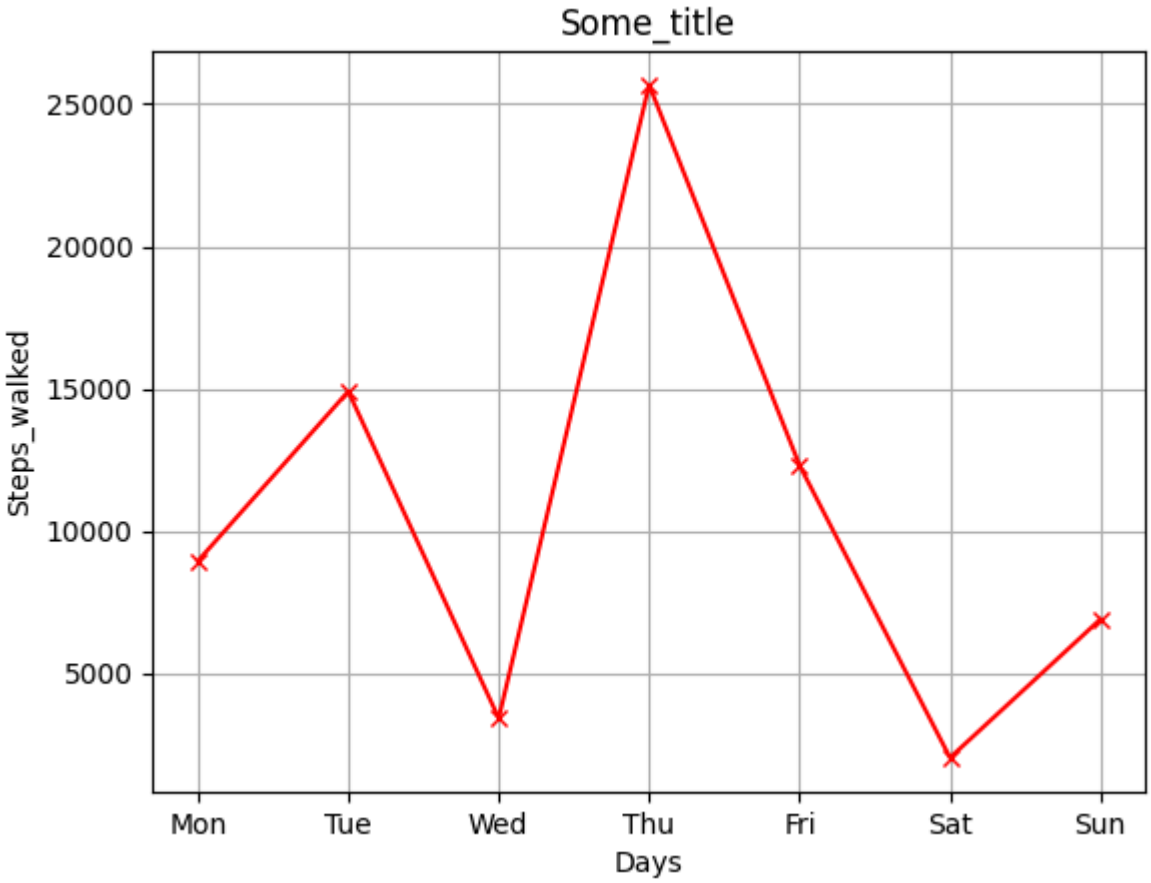

```
days = ["Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun"]
steps_walked = [8934, 14902, 3409, 25672, 12300, 2023, 6890]
plt.plot(days, steps_walked,marker='x')
plt.show()
```



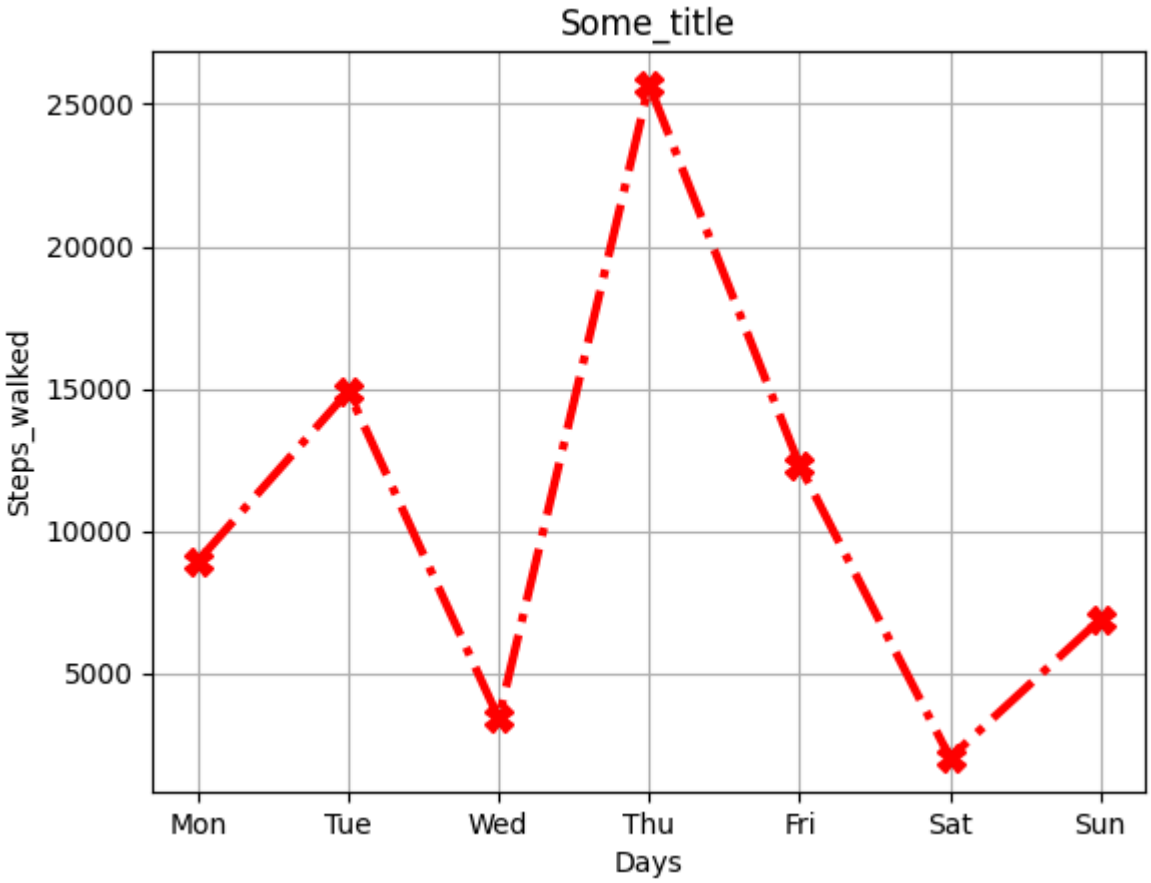
```
days = ["Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun"]
steps_walked = [8934, 14902, 3409, 25672, 12300, 2023, 6890]
plt.plot(days, steps_walked,marker='x',color='r')
plt.show()
```



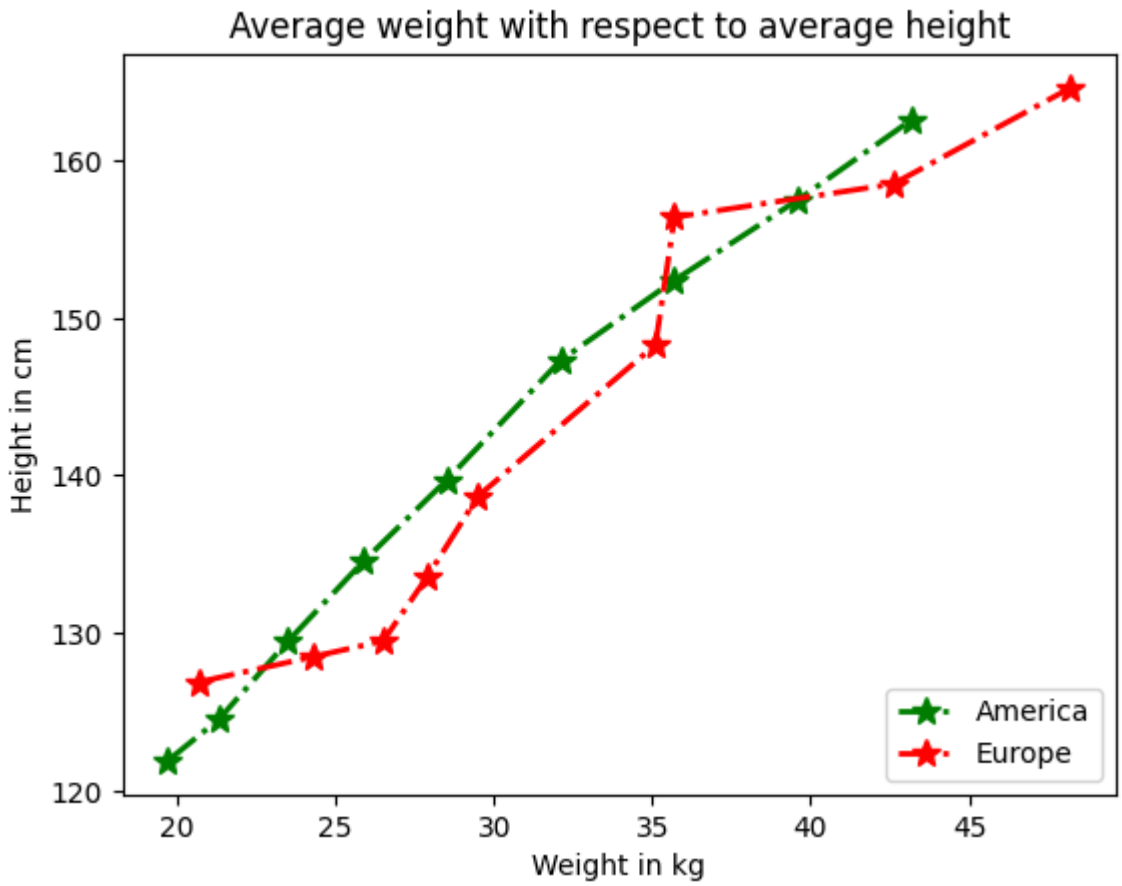

```
days = ["Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun"]
steps_walked = [8934, 14902, 3409, 25672, 12300, 2023, 6890]
plt.plot(days, steps_walked,marker='x',color='r')

plt.xlabel("Days") #add the Label on x-axis
plt.ylabel("Steps_walked") #add the Label on y-axis
plt.title("Some_title") #add the title to the chart
plt.grid(True) #add gridlines to the background
plt.show()
```



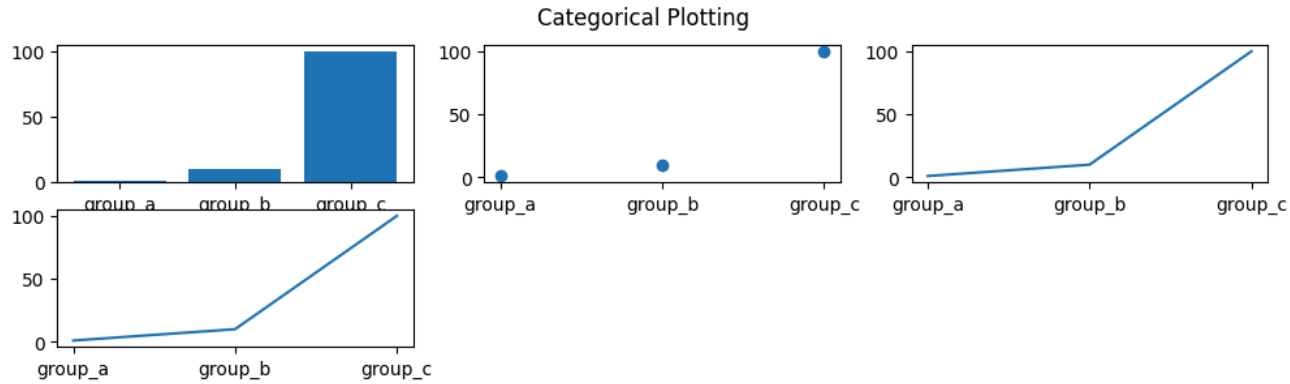

```
days = ["Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun"]
steps_walked = [8934, 14902, 3409, 25672, 12300, 2023, 6890]
plt.plot(days, steps_walked,marker='X',markersize=10,color='red',linewidth=3, linestyle='

plt.xlabel("Days") #add the Label on x-axis
plt.ylabel("Steps_walked") #add the Label on y-axis
plt.title("Some_title") #add the title to the chart
plt.grid(True) #add gridlines to the background
plt.show()
```




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

#height and weight in america
height=[121.9,124.5,129.5,134.6,139.7,147.3,152.4,157.5,162.6]
weight=[19.7,21.3,23.5,25.9,28.5,32.1,35.7,39.6,43.2]
plt.plot(weight,height,marker='*',markersize=10,color='green',linewidth=2, linestyle='das

#height and weight in europe
height=[126.9,128.5,129.5,133.6,138.7,148.3,156.4,158.5,164.6]
weight=[20.7,24.3,26.5,27.9,29.5,35.1,35.7,42.6,48.2]
plt.plot(weight,height,marker='*',markersize=10,color='red',linewidth=2, linestyle='dashd
#df=pd.DataFrame({"height":height,"weight":weight})

#Set xlabel for the plot
plt.xlabel('Weight in kg')

#Set ylabel for the plot
plt.ylabel('Height in cm')

#Set chart title:
plt.title('Average weight with respect to average height')



plt.legend(["America", "Europe"], loc ="lower right")
plt.show()
```

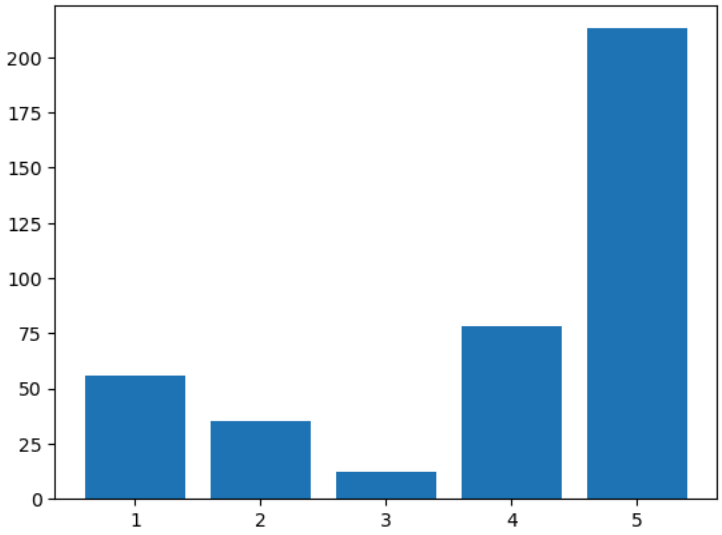



```
names = ['group_a', 'group_b', 'group_c']
values = [1, 10, 100]

plt.figure(figsize=(12, 3))

plt.subplot(231)
plt.bar(names, values)

plt.subplot(232)
plt.scatter(names, values)

plt.subplot(233)
plt.plot(names, values)

plt.subplot(234)
plt.plot(names, values)

plt.suptitle('Categorical Plotting')
plt.show()
```

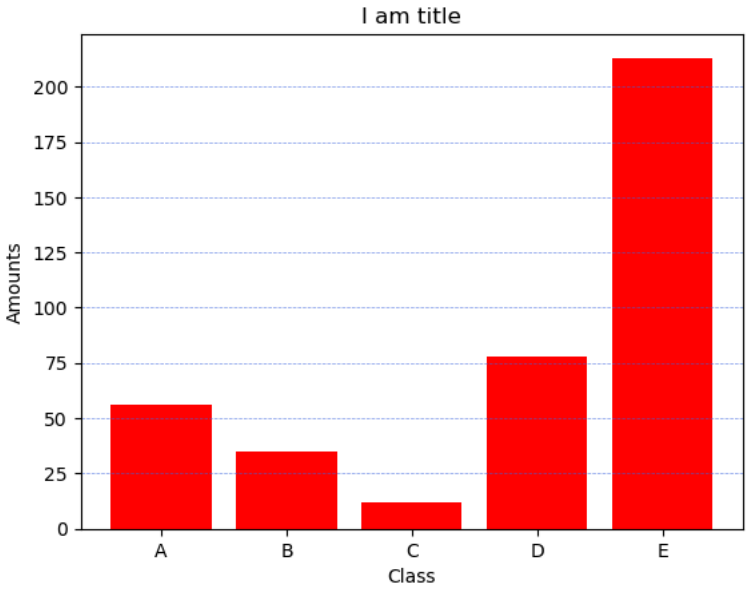

Categorical Plotting
100
50
0
group_a
group_b
group_c

```
# Fixing random state for reproducibility
np.random.seed(19680801)

# make up some data in the open interval (0, 1)
y = np.random.normal(loc=0.5, scale=0.4, size=1000)
y = y[(y > 0) & (y < 1)]
y.sort()
x = np.arange(len(y))

# plot with various axes scales
plt.figure()

# linear
plt.subplot(221)
plt.plot(x, y)
plt.yscale('linear')
plt.title('linear')

plt.grid(True)

# log
plt.subplot(222)
plt.plot(x, y)
plt.yscale('linear')
plt.title('linear')
plt.grid(True)

# symmetric log
plt.subplot(223)
plt.plot(x, y)
plt.yscale('linear')
plt.title('linear')
plt.grid(True)
```

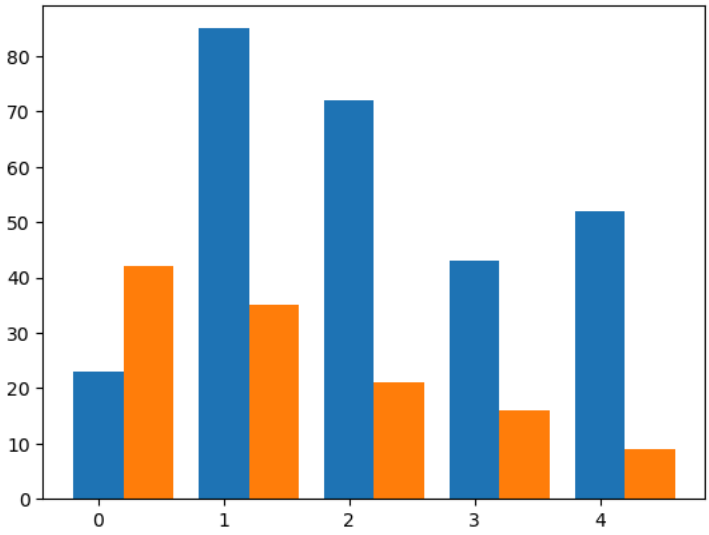
```
import matplotlib.pyplot as plt

data = [56, 35, 12, 78, 213]
x=[1,2,3,4,5]
plt.bar(x, data)

#plt.bar(x,data,color='red')
#plt.grid(color='royalblue', linestyle='--', linewidth=0.5, axis='y', alpha=0.7)
plt.show()
```




```
import matplotlib.pyplot as plt

data = [56, 35, 12, 78, 213]
labels = ['A', 'B', 'C', 'D', 'E']


plt.xticks(range(len(data)), labels)
plt.xlabel('Class')
plt.ylabel('Amounts')
plt.title('I am title')

plt.bar(range(len(data)),data,color='red')
plt.grid(color='royalblue', linestyle='--', linewidth=0.5, axis='y', alpha=0.7)

plt.show()
```

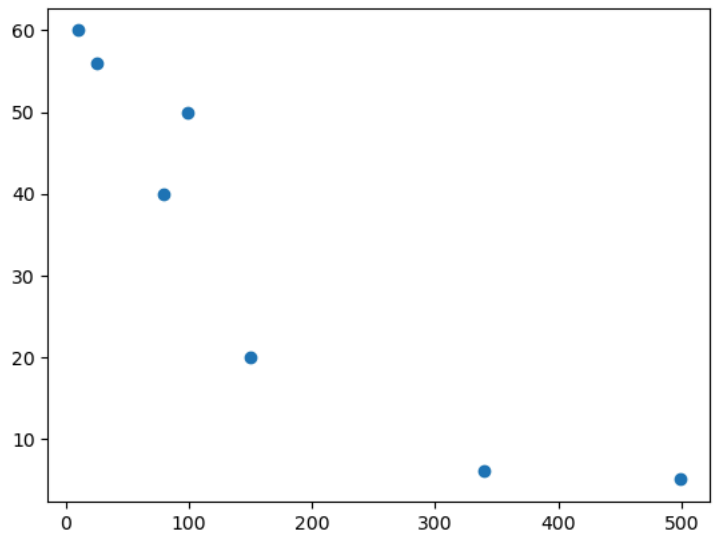


```
import matplotlib.pyplot as plt
import numpy as np

data1 = [23,85, 72, 43, 52]
data2 = [42, 35, 21, 16, 9]
width =0.4
plt.bar(np.arange(len(data1)), data1, width=width)
plt.bar(np.arange(len(data2))+ width, data2, width=width)
plt.show()
```

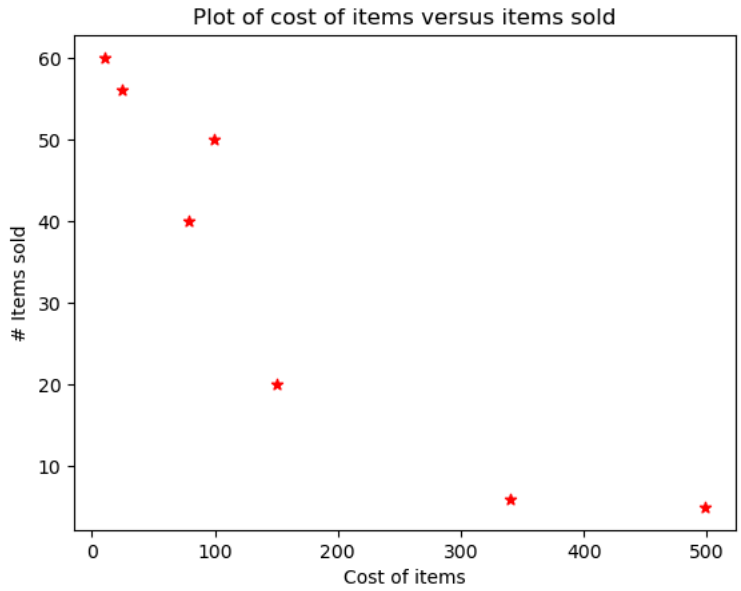



```
Start coding or generate with AI.
```

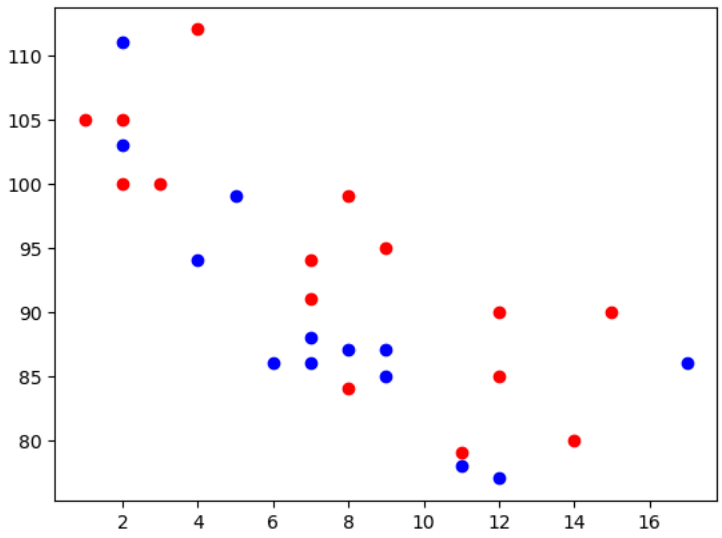
```
#Scatter plots
import pandas as pd
import matplotlib.pyplot as plt


Cost_of_items=[10, 25, 79, 99, 150, 499, 340]
Items_sold_each_day=[60, 56, 40, 50, 20, 5, 6]

plt.scatter(Cost_of_items, Items_sold_each_day)
plt.show()
```




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


Cost_of_items=[10, 25, 79, 99, 150, 499, 340]
Items_sold_each_day=[60, 56, 40, 50, 20, 5, 6]

plt.scatter(Cost_of_items, Items_sold_each_day,marker="*",color='red' )

# Set title to "Mela Sales Report"
plt.title('Plot of cost of items versus items sold')

# Label x axis as "Days"
plt.xlabel('Cost of items')

# Label y axis as "Sales in Rs"
plt.ylabel('# Items sold')

plt.show()
```



```
### How to draw two plots on same figure:
import matplotlib.pyplot as plt
import numpy as np


x = np.array([5,7,8,7,2,17,2,9,4,11,12,9,6])
y = np.array([99,86,87,88,111,86,103,87,94,78,77,85,86])
plt.scatter(x, y,color='blue')


x = np.array([2,2,8,1,15,8,12,9,7,3,11,4,7,14,12])
y = np.array([100,105,84,105,90,99,90,95,94,100,79,112,91,80,85])
plt.scatter(x, y,color='red')


plt.show()
```




```
Start coding or generate with AI.
```