Project name: Infosys_BudgetWise AI Based Expense Forecasting Tool

Numpy: NumPy (Numerical Python) is a powerful Python library used for numerical and scientific computing.

Single Dimensional: Creates a 1-D NumPy array with values 10, 20, 30.

```
import numpy as np
n1=np.array([10,20,30])
n1
```

```
array([10, 20, 30])
```

Multidimensional: Creates a 2-D array (matrix) with 2 rows and 4 columns.

```
import numpy as np
n2=np.array([[10,2,3,4],[3,4,55,66]])
n2
```

```
array([[10,  2,  3,  4],
       [ 3,  4, 55, 66]])
```

Array of Zeros: Creates a 1x2 matrix filled with zeros.

```
import numpy as np
n1=np.zeros((1,2))
n1
```

```
array([[0., 0.]])
```

```
n2=np.zeros((5,5))
n2
```

```
array([[0., 0., 0., 0., 0.],
       [0., 0., 0., 0., 0.],
       [0., 0., 0., 0., 0.],
       [0., 0., 0., 0., 0.],
       [0., 0., 0., 0., 0.]])
```

Using arange():

```
n1=np.full((2,2),10)
n1
```

```
array([[10, 10],
       [10, 10]])
```

```
n1=np.arange(10,20)
n1
```

```
array([10, 11, 12, 13, 14, 15, 16, 17, 18, 19])
```

```
n1=np.arange(10,50,5)
n1
```

```
array([10, 15, 20, 25, 30, 35, 40, 45])
```

Random Number:

```
n1=np.random.randint(1,100,5)
n1
```

```
array([61, 56, 78, 12, 34])
```

```
n1=np.array([[1,2,3],[4,5,6]])
n1.shape
```

```
(2, 3)
```

```
n1.shape=(3,2)
n1.shape
```

```
(3, 2)
```

Column Stack, VStack & hStack:

```
n1=np.array([10,20,30])
n2=np.array([40,50,60])
np.column_stack((n1,n2))
```

```
array([[10, 40],
       [20, 50],
       [30, 60]])
```

```
n1=np.array([10,20,30])
n2=np.array([40,50,60])
np.vstack((n1,n2))
```

```
array([[10, 20, 30],
       [40, 50, 60]])
```

```
n1=np.array([10,20,30])
n2=np.array([40,50,60])
np.hstack((n1,n2))
```

```
array([10, 20, 30, 40, 50, 60])
```

Addition of numpy arrays:

```
import numpy as np
n1=np.array([10,20])
n2=np.array([30,40])
np.sum([n1,n2])
```

```
np.int64(100)
```

```
np.sum([n1,n2],axis=0)
```

```
array([40, 60])
```

```
np.sum([n1,n2],axis=1)
```

```
array([30, 70])
```

Basic Addition:

```
n1=np.array([10,20,30])
n1=n1+1
n1
```

```
array([11, 21, 31])
```

Basic Multiplication:

```
n1=np.array([10,20,30])
n1=n1*2
n1
```

```
array([20, 40, 60])
```

Basic Subtraction:

```
n1=np.array([10,20,30])
n1=n1-1
n1
```

```
array([ 9, 19, 29])
```

Basic division:

```
n1=np.array([10,20,30])
n1=n1/2
n1
```

```
array([ 5., 10., 15.])
```

Mean,Standard Deviation,Median:

```
n1=np.array([10,20,30,40,50,60])
np.mean(n1)
```

```
np.float64(35.0)
```

```
n1=np.array([1,5,3,100,4,48])
np.std(n1)
```

```
np.float64(36.59424666377065)
```

```
n1=np.array([11,44,5,96,67,85])
np.median(n1)
```

```
np.float64(55.5)
```

Numpy Matrix:

```
n1=np.array([[1,2,3],[4,5,6],[7,8,9]])
n1
```

```
array([[1, 2, 3],
       [4, 5, 6],
       [7, 8, 9]])
```

+ Code + Text

```
n1[0]
```

```
array([1, 2, 3])
```

```
n1[1]
```

```
array([4, 5, 6])
```

```
n1[:,1]
```

```
array([2, 5, 8])
```

```
n1[:,2]
```

```
array([3, 6, 9])
```

Transpose:

```
import numpy as np
n1=np.array([[1,2,3],[4,5,6],[7,8,9]])
n1
```

```
array([[1, 2, 3],
       [4, 5, 6],
       [7, 8, 9]])
```

```
n1.transpose()
```

```
array([[1, 4, 7],
       [2, 5, 8],
       [3, 6, 9]])
```

Matrix Multiplication:

```
n1=np.array([[1,2,3],[4,5,6],[7,8,9]])
n1
```

```
array([[1, 2, 3],
       [4, 5, 6],
       [7, 8, 9]])
```

```
n2=np.array([[9,8,7],[6,5,4],[3,2,1]])
n2
```

```
array([[9, 8, 7],
       [6, 5, 4],
       [3, 2, 1]])
```

```
n1.dot(n2)
```

```
array([[ 30,  24,  18],
       [ 84,  69,  54],
       [138, 114,  90]])
```

```
n2.dot(n1)
```

```
array([[ 90, 114, 138],
       [ 54,  69,  84],
       [ 18,  24,  30]])
```

Pandas: Pandas is a powerful Python library for data manipulation and analysis.

Series object is one-dimensional labeled array

Series basics:

```
import pandas as pd
s1=pd.Series([1,2,3,4,5])
s1
```

	0
0	1
1	2
2	3
3	4
4	5

dtype: int64

Create a 1-D labeled Array:

```
s1=pd.Series([1,2,3,4,5],index=['a','b','c','d','e'])
s1
```

	0
a	1
b	2
c	3
d	4
e	5

dtype: int64

```
pd.Series({'a':10,'b':20,'c':30})
```

	0
a	10
b	20
c	30

dtype: int64

Create Series with custom labels: Creates Series from a dictionary.

```
pd.Series({'a':10,'b':20,'c':30},index=['b','c','d','a'])
```

	0
b	20.0
c	30.0
d	NaN
a	10.0

dtype: float64

```
import pandas as pd
s1=pd.Series([1,2,3,4,5,6,7,8,9])
s1[3]
```

```
np.int64(4)
```

```
s1=pd.Series([1,2,3,4,5,6,7,8,9])
s1[-3:]
```

	0
6	7
7	8
8	9

dtype: int64

```
s1=pd.Series([1,2,3,4,5,6,7,8,9])
s1[:4]
```

	0
0	1
1	2
2	3
3	4

dtype: int64

Basic Math Operations on series

```
import pandas as pd
s1=pd.Series([1,2,3,4,5,6,7,8,9])
s1+5
```

	0
0	6
1	7
2	8
3	9
4	10
5	11
6	12
7	13
8	14

dtype: int64

Adding two series object:

```
s1=pd.Series([1,2,3,4,5,6,7,8,9])
s2=pd.Series([10,20,30,40,50,60,70,80,90])
s1+s2
```

	0
0	11
1	22
2	33
3	44
4	55
5	66
6	77
7	88
8	99

dtype: int64

Pandas dataframe

```
pd.DataFrame({"Name":['bob','sam','Anne'],"Marks":[76,25,92]})
```

	Name	Marks
0	bob	76
1	sam	25
2	Anne	92

dataFrame In-built Functions

```
import pandas as pd
ds=pd.read_csv('Iris.csv')
```

```
ds.head()
```

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
0	1	5.1	3.5	1.4	0.2
1	2	4.9	3.0	1.4	0.2
2	3	4.7	3.2	1.3	0.2
3	4	4.6	3.1	1.5	0.2
4	5	5.0	3.6	1.4	0.2

CSV Operations:

```
ds.tail(10)
```

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidt
140	141	6.7	3.1	5.6	
141	142	6.9	3.1	5.1	
142	143	5.8	2.7	5.1	
143	144	6.8	3.2	5.9	
144	145	6.7	3.3	5.7	
145	146	6.7	3.0	5.2	
146	147	6.3	2.5	5.0	
147	148	6.5	3.0	5.2	
148	149	6.2	3.4	5.4	

```
ds.head(20)
```

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidth
0	1	5.1	3.5	1.4	
1	2	4.9	3.0	1.4	
2	3	4.7	3.2	1.3	
3	4	4.6	3.1	1.5	
4	5	5.0	3.6	1.4	
5	6	5.4	3.9	1.7	
6	7	4.6	3.4	1.4	
7	8	5.0	3.4	1.5	
8	9	4.4	2.9	1.4	
9	10	4.9	3.1	1.5	

```
ds.describe()
```

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm
count	150.000000	150.000000	150.000000	150.000000
mean	75.500000	5.843333	3.054000	3.758667
std	43.445368	0.828066	0.433594	1.764420
min	1.000000	4.300000	2.000000	1.000000
25%	38.250000	5.100000	2.800000	1.600000
50%	75.500000	5.800000	3.000000	4.350000
75%	112.750000	6.400000	3.300000	5.100000
max	150.000000	7.900000	4.400000	6.900000

```
import seaborn as sns

iris = sns.load_dataset("iris")
iris.loc[0:3, ["sepal_length", "petal_length"]]
```

	sepal_length	petal_length
0	5.1	1.4
1	4.9	1.4
2	4.7	1.3
3	4.6	1.5

```
iris.drop([1,2,3],axis=0)
```

	sepal_length	sepal_width	petal_length	petal_width	spe
0	5.1	3.5	1.4	0.2	se
4	5.0	3.6	1.4	0.2	se
5	5.4	3.9	1.7	0.4	se
6	4.6	3.4	1.4	0.3	se
7	5.0	3.4	1.5	0.2	se
...	...	...	...	...	
145	6.7	3.0	5.2	2.3	virg
146	6.3	2.5	5.0	1.9	virg
147	6.5	3.0	5.2	2.0	virg
148	6.2	3.4	5.4	2.3	virg
149	5.9	3.0	5.1	1.8	virg

147 rows × 5 columns

```
iris.mean(numeric_only=True)
```

	0
sepal_length	5.843333
sepal_width	3.057333
petal_length	3.758000
petal_width	1.199333

dtype: float64

```
iris.median(numeric_only=True)
```

	0
sepal_length	5.80
sepal_width	3.00
petal_length	4.35
petal_width	1.30

dtype: float64

```
iris.min()
```

	0
sepal_length	4.3
sepal_width	2.0
petal_length	1.0
petal_width	0.1
species	setosa

dtype: object

```
iris.max()
```

	0
sepal_length	7.9
sepal_width	4.4
petal_length	6.9
petal_width	2.5
species	virginica

dtype: object

Matplotlib: Matplotlib is a Python library for creating 2D and basic 3D visualizations.

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

```
x=np.arange(1,11)
x
```

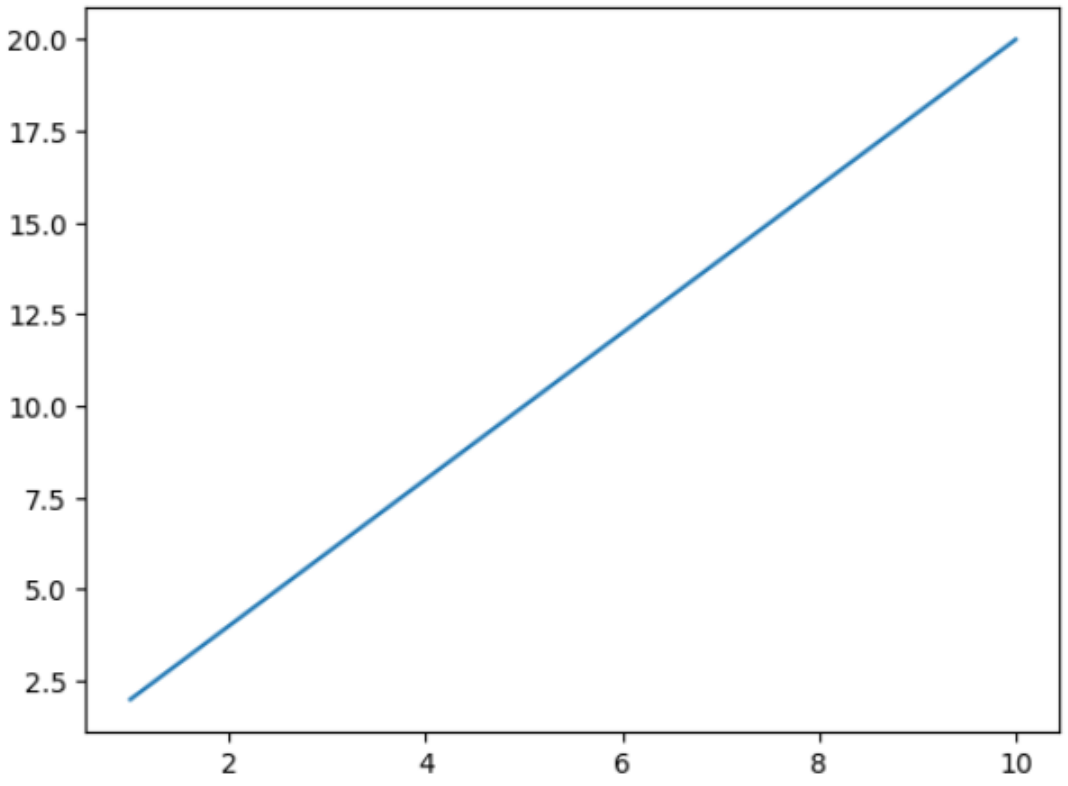
```
array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10])
```

```
y=2*x
y
```

```
array([ 2,  4,  6,  8, 10, 12, 14, 16, 18, 20])
```

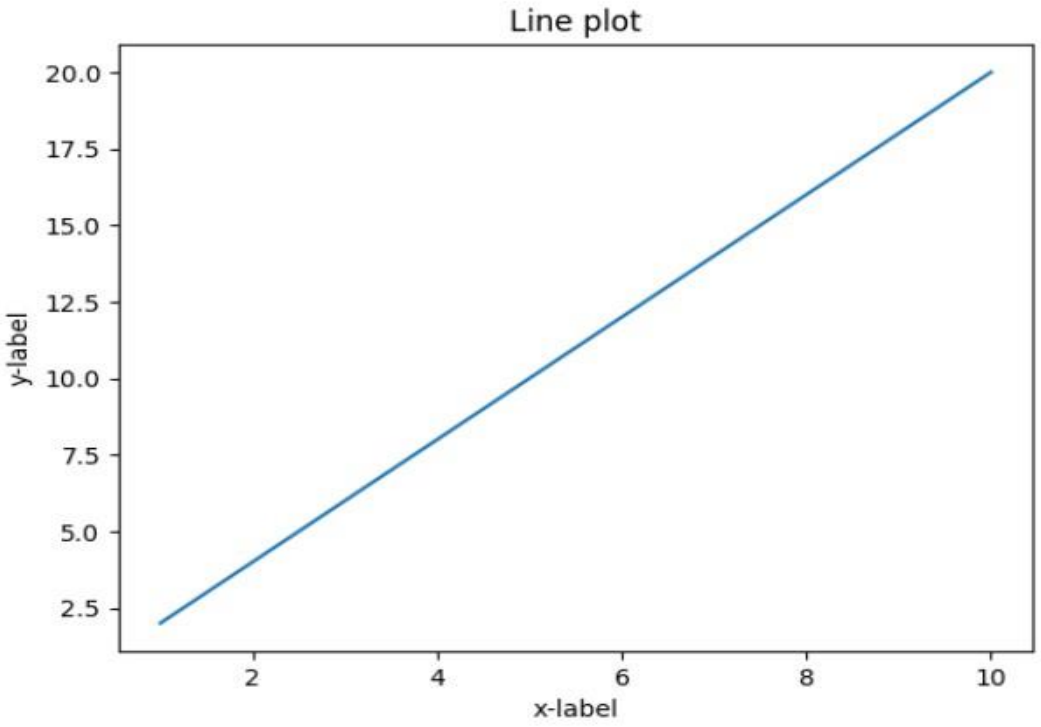
Line Plot:

```
plt.plot(x,y)
plt.show()
```

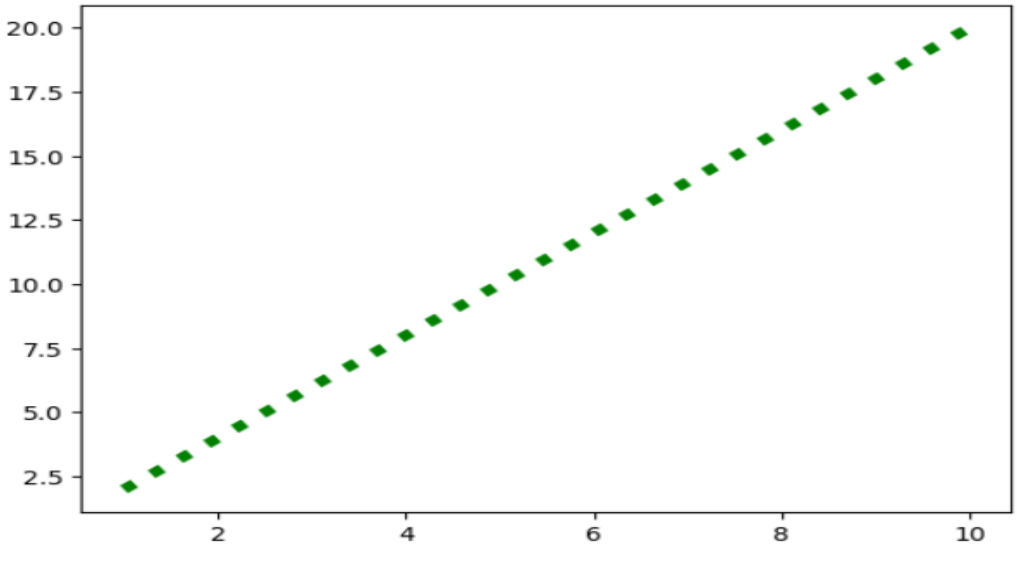


Line Plot with Title and Labels:

```
plt.plot(x,y)
plt.title("Line plot")
plt.xlabel("x-label")
plt.ylabel("y-label")
plt.show()
```

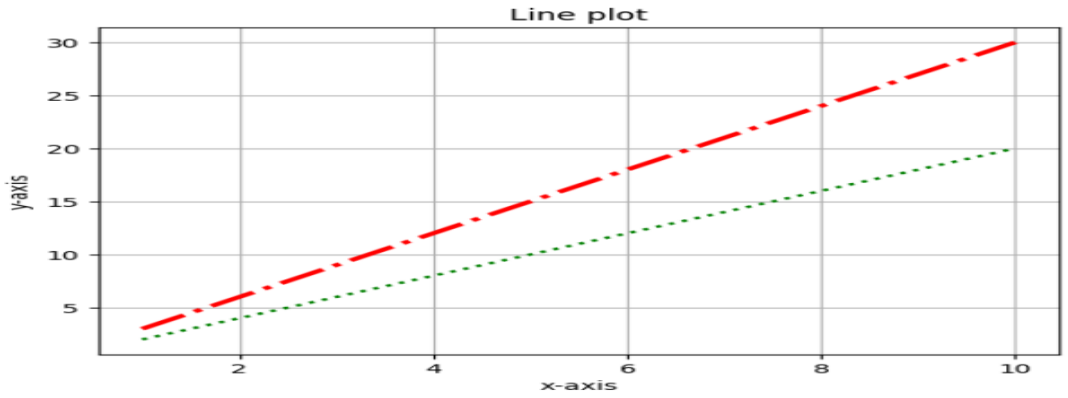



```
plt.plot(x,y,color='g',linestyle=':',linewidth=5)
plt.show()
```



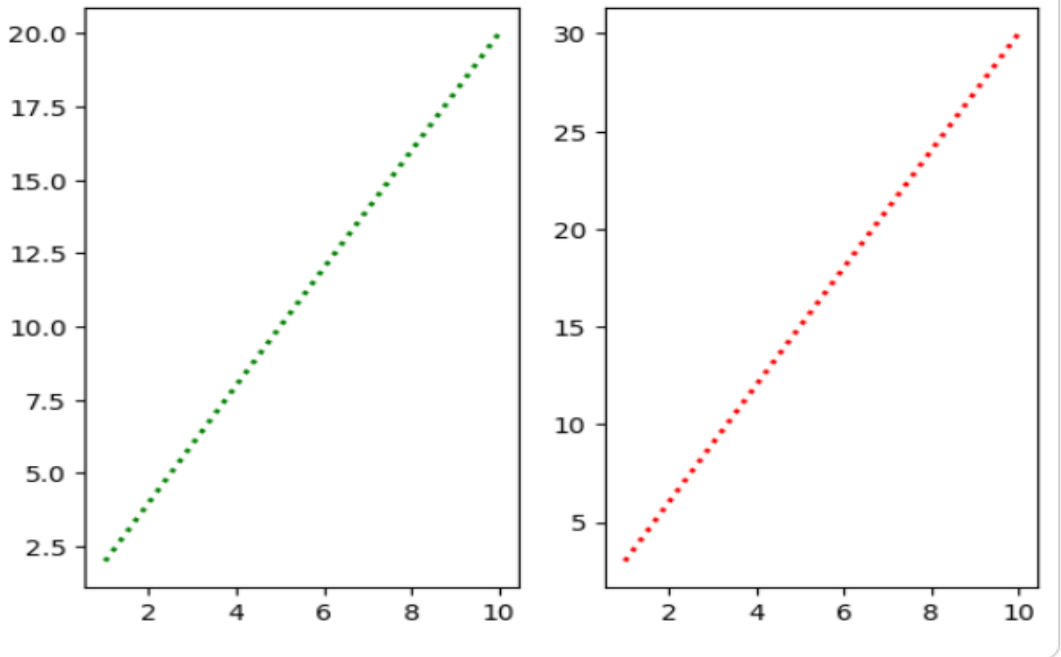
Multiple Line Styles:

```
plt.plot(x,y1,color='g',linestyle=':',linewidth=2)
plt.plot(x,y2,color='r',linestyle='-.',linewidth=3)
plt.title("Line plot")
plt.xlabel("x-axis")
plt.ylabel("y-axis")
plt.grid(True)
plt.show()
```



Subplots:

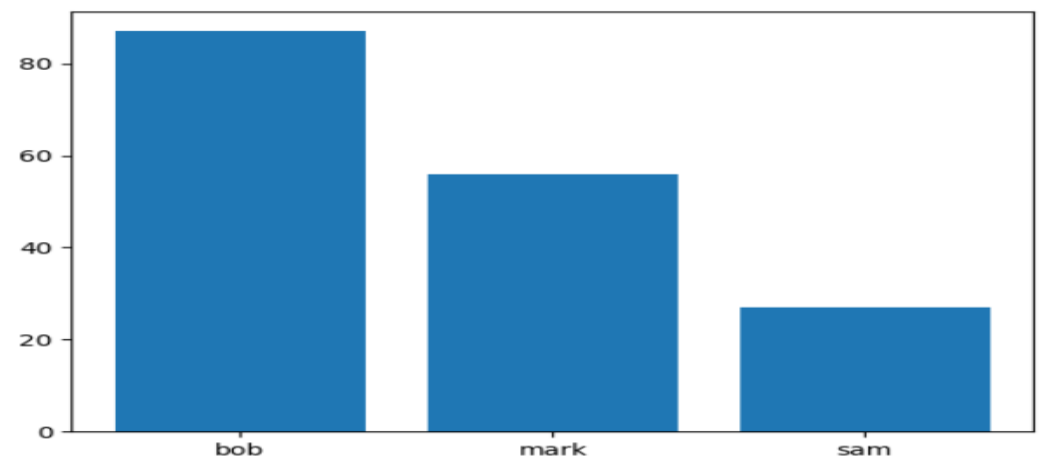
```
plt.subplot(1,2,1)
plt.plot(x,y1,color='g',linestyle=':',linewidth=2)
plt.subplot(1,2,2)
plt.plot(x,y2,color='r',linestyle=':',linewidth=2)
plt.show()
```



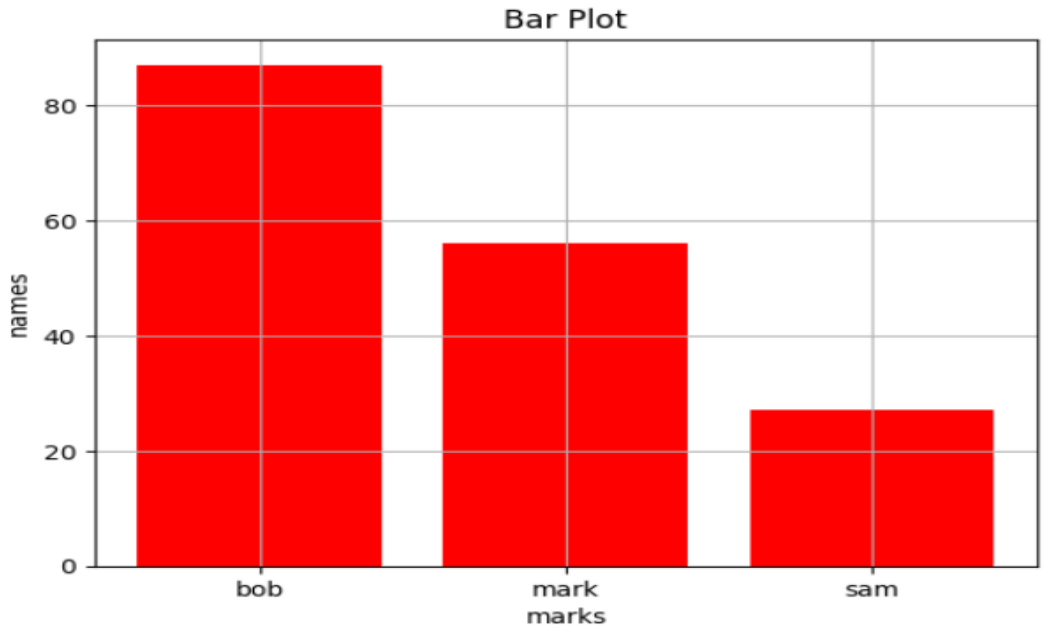
Bar Plot:

```
student={"bob":87,"mark":56,"sam":27}
```

```
names=list(student.keys())
values=list(student.values())
plt.bar(names,values)
plt.show()
```

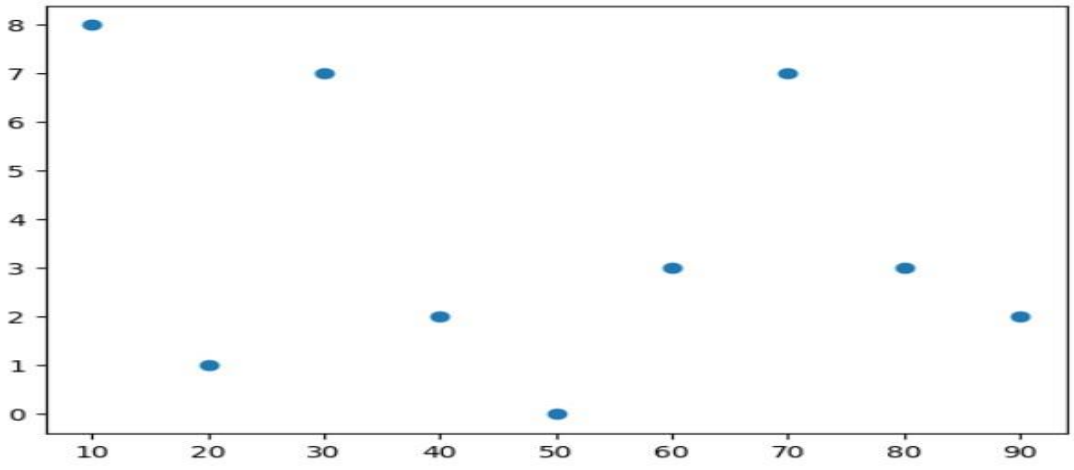



```
plt.bar(names,values,color='r')
plt.title("Bar Plot")
plt.xlabel("marks")
plt.ylabel("names")
plt.grid(True)
plt.show()
```

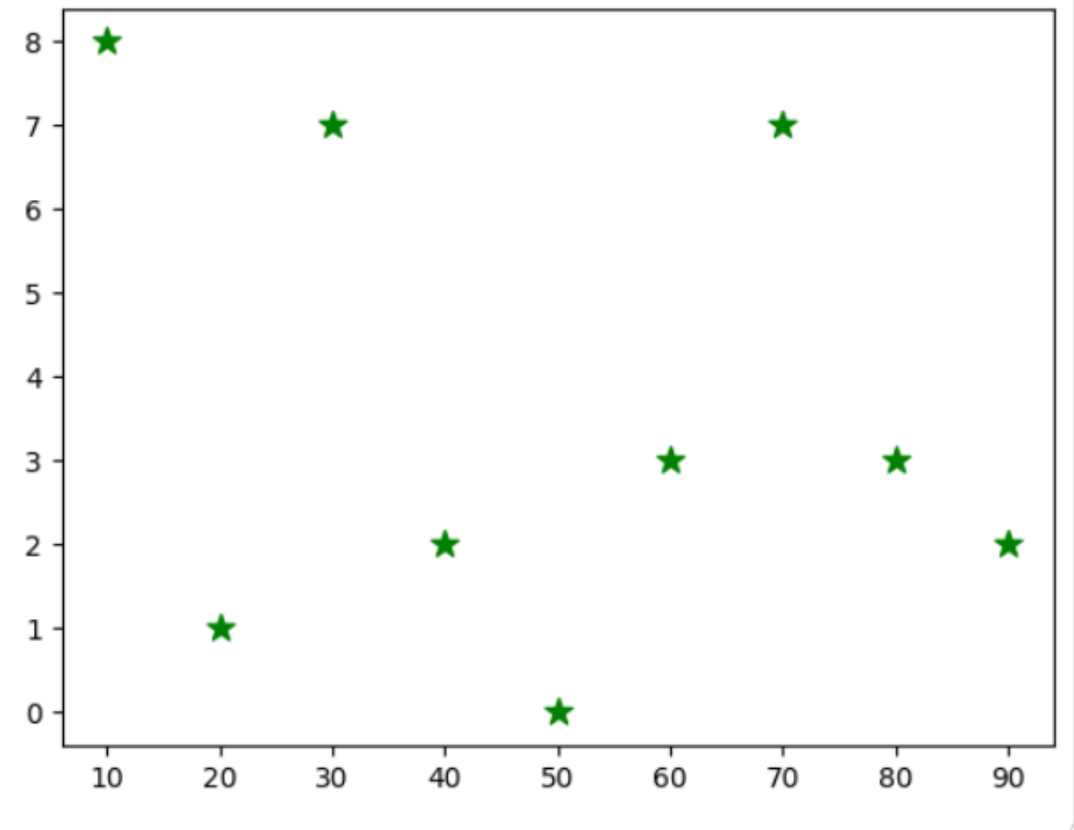


Scatter Plot:

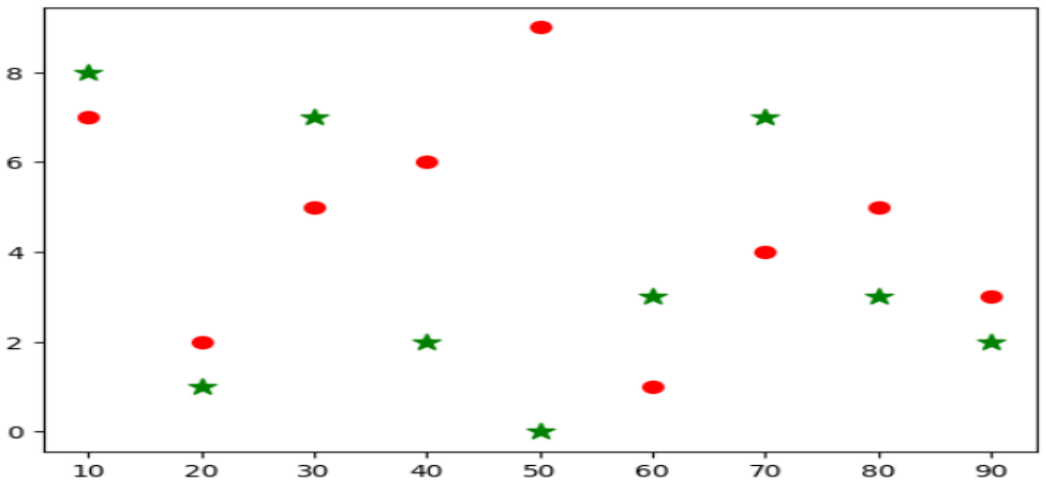
```
from matplotlib import pyplot as plt
x=[10,20,30,40,50,60,70,80,90]
y=[8,1,7,2,0,3,7,3,2]
plt.scatter(x,y)
plt.show()
```




```
plt.scatter(x,y,marker="*",c="g",s=100)
plt.show()
```

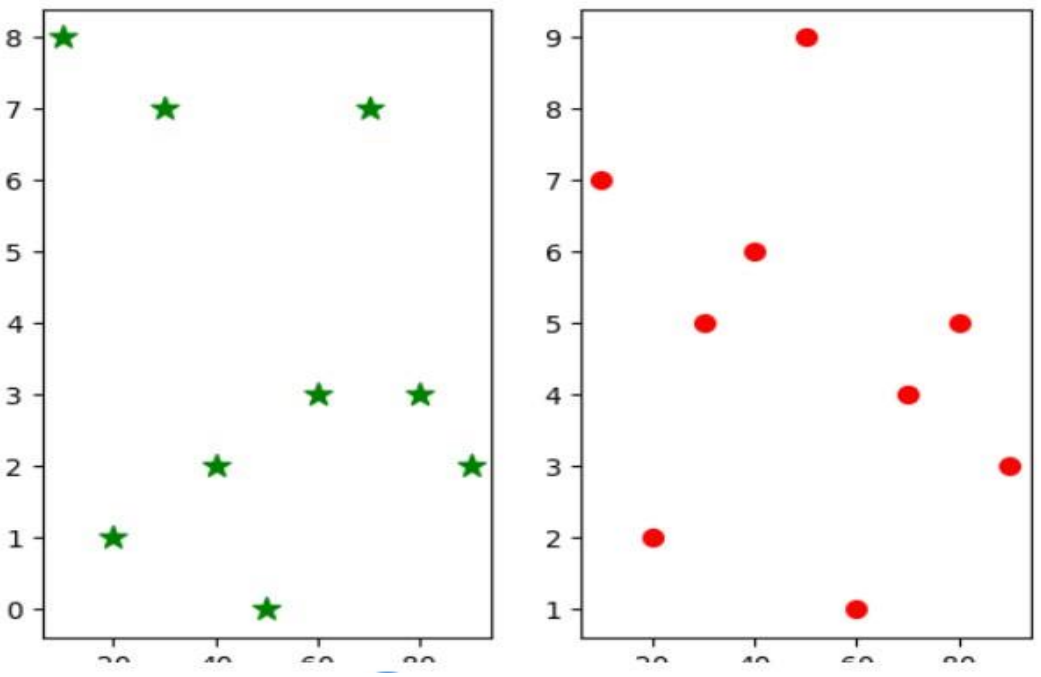


```
x=[10,20,30,40,50,60,70,80,90]
a=[8,1,7,2,0,3,7,3,2]
b=[7,2,5,6,9,1,4,5,3]
plt.scatter(x,a,marker="*",c="g",s=100)
plt.scatter(x,b,marker=".",c="r",s=200)
plt.show()
```



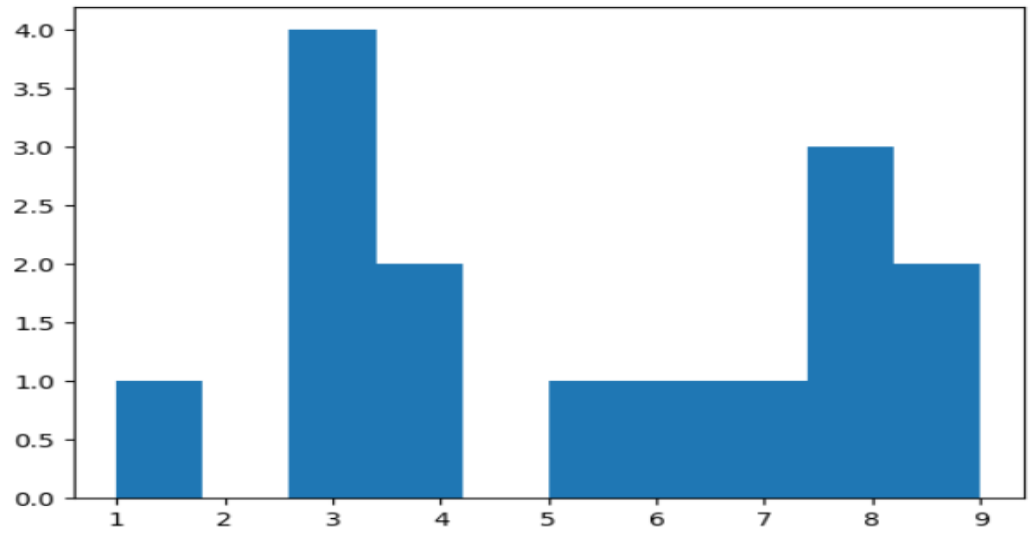

```
x=[10,20,30,40,50,60,70,80,90]
a=[8,1,7,2,0,3,7,3,2]
b=[7,2,5,6,9,1,4,5,3]
plt.subplot(1,2,1)
plt.scatter(x,a,marker="*",c="g",s=100)

plt.subplot(1,2,2)
plt.scatter(x,b,marker=".",c="r",s=200)
plt.show()
```

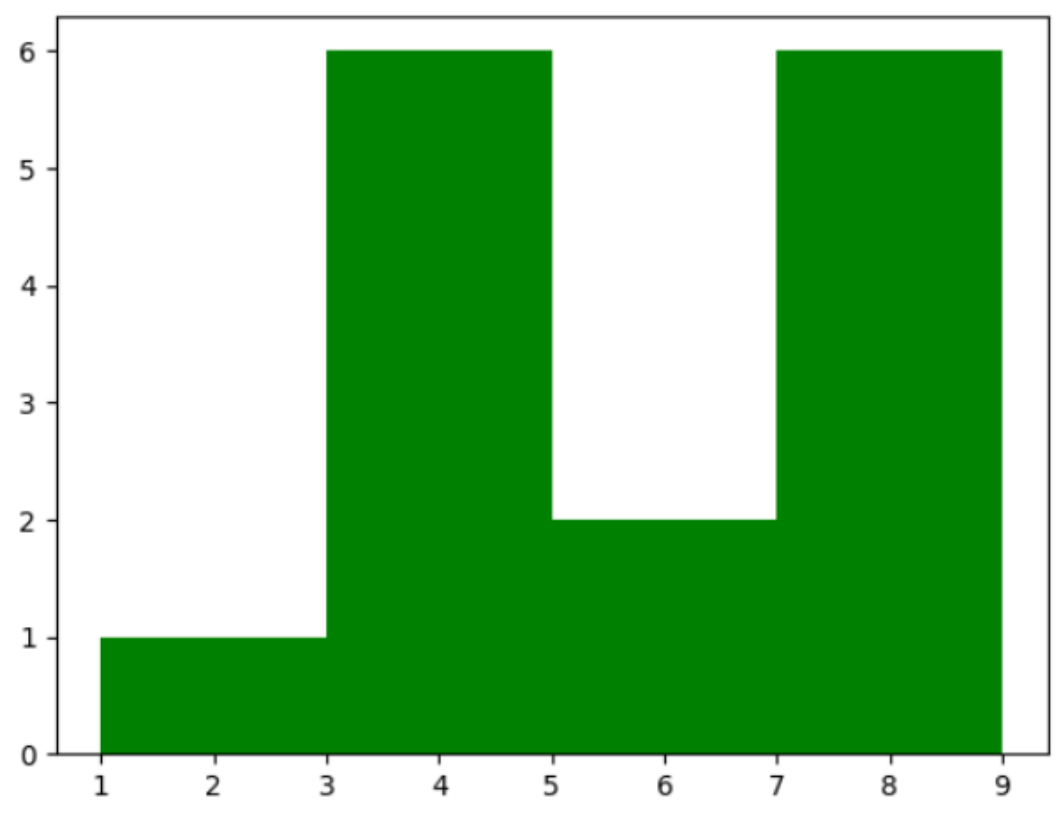


Histogram:

```
data=[1,3,3,3,3,9,9,5,4,4,8,8,8,6,7]
plt.hist(data)
plt.show()
```




```
plt.hist(data,color="g",bins=4)
plt.show()
```

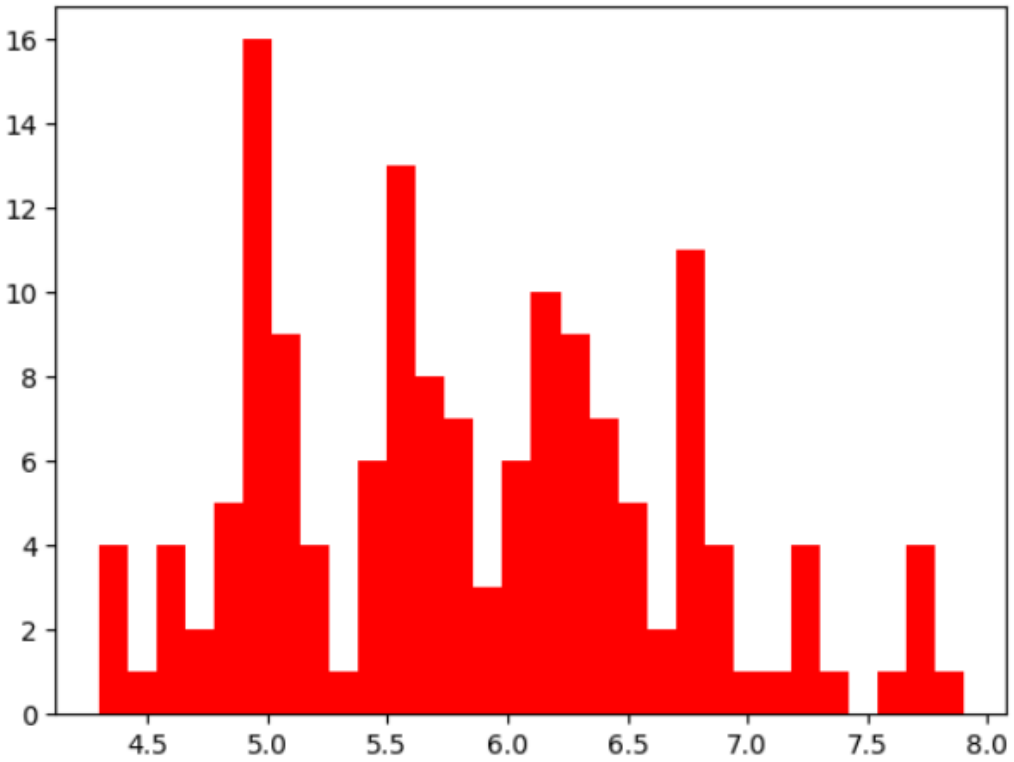


```
import pandas as pd
iris=pd.read_csv('Iris.csv')
iris.head()
```

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
0	1	5.1	3.5	1.4	0.2
1	2	4.9	3.0	1.4	0.2
2	3	4.7	3.2	1.3	0.2
3	4	4.6	3.1	1.5	0.2
4	5	5.0	3.6	1.4	0.2

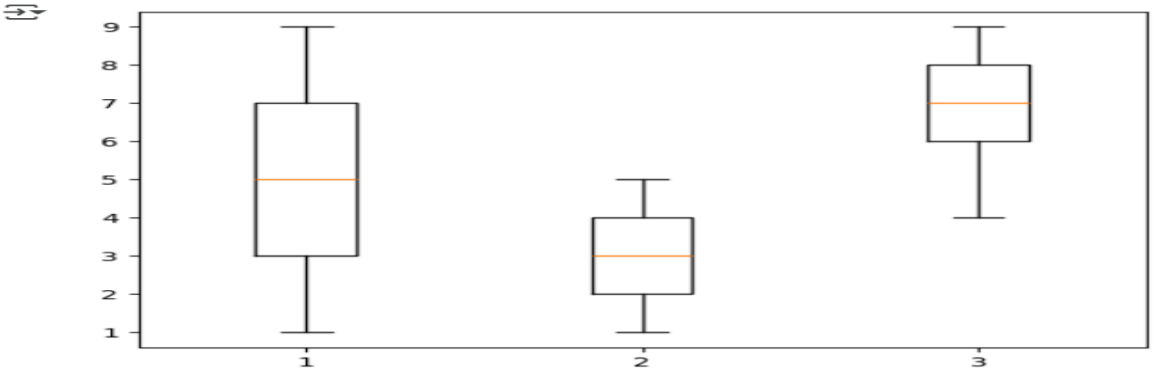
```
plt.hist(iris['sepal_length'], bins=30, color='r')

plt.show()
```



Box Plot:

```
one=[1,2,3,4,5,6,7,8,9]
two=[1,2,3,4,5,4,3,2,1]
three=[6,7,8,9,8,7,6,5,4]
data=list([one,two,three])
plt.boxplot(data)
plt.show()
```



Seaborn: Seaborn is a Python data visualization library built on top of Matplotlib.
It makes it easy to create beautiful, high-quality statistical graphs with fewer lines of code.

Seaborn Explanations:

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

```
ds=sns.load_dataset('tips')
```

```
ds.head()
```

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4

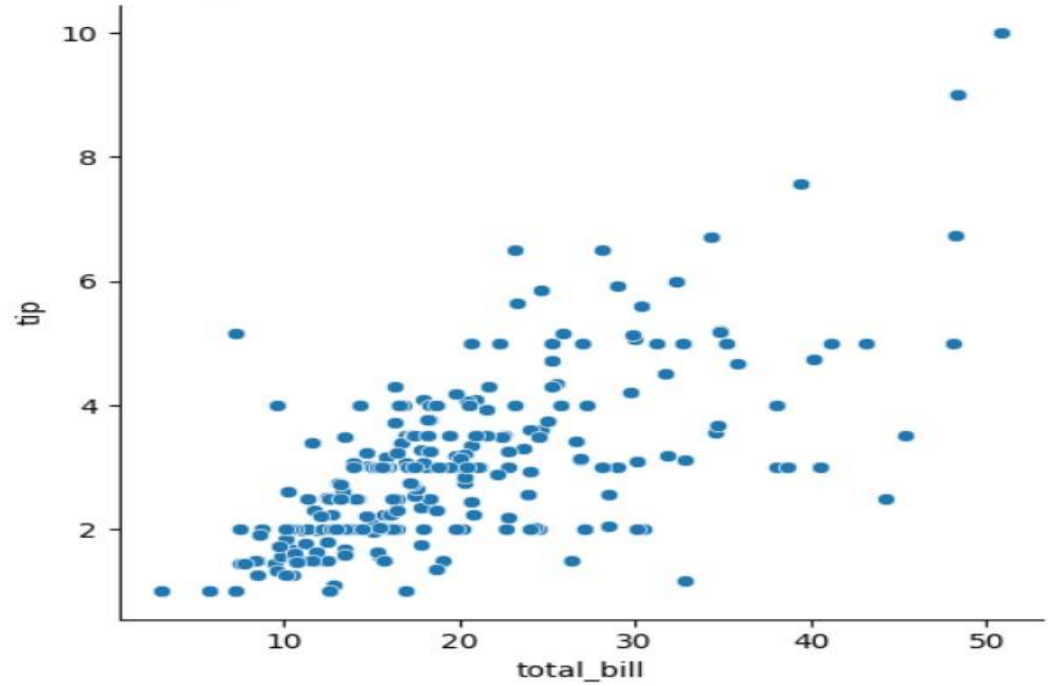
```
ds.shape
```

```
(244, 7)
```

Relplot(scatter Plot): Draws scatter plot of two variables.

```
sns.relplot(data=ds,x='total_bill',y='tip')
```

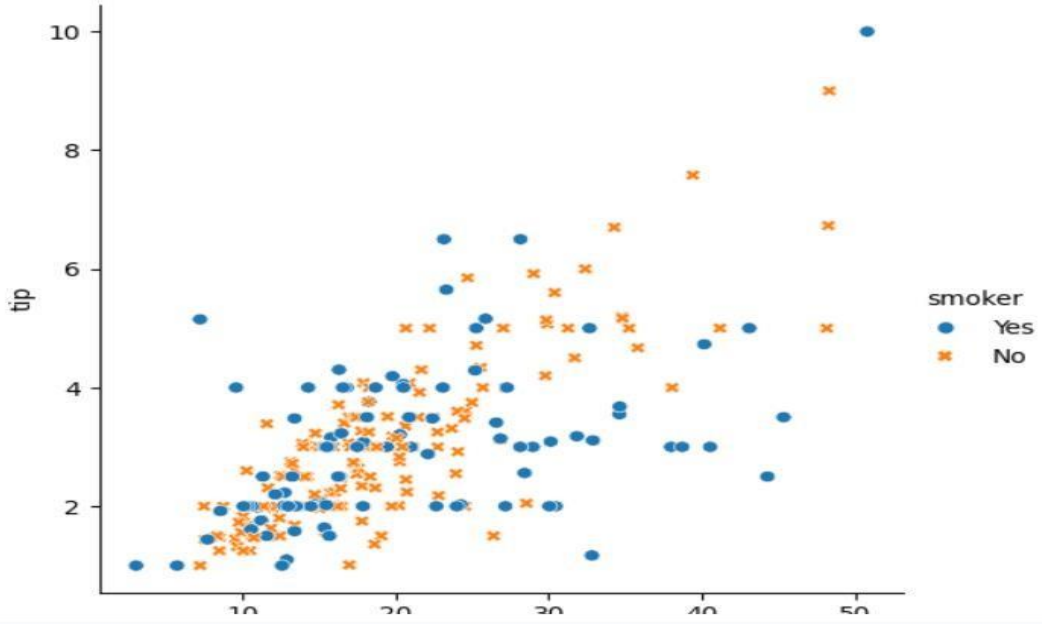
```
<seaborn.axisgrid.FacetGrid at 0x7cba6f76f9b0>
```



Using Hue/Style:Adds color and marker variation:

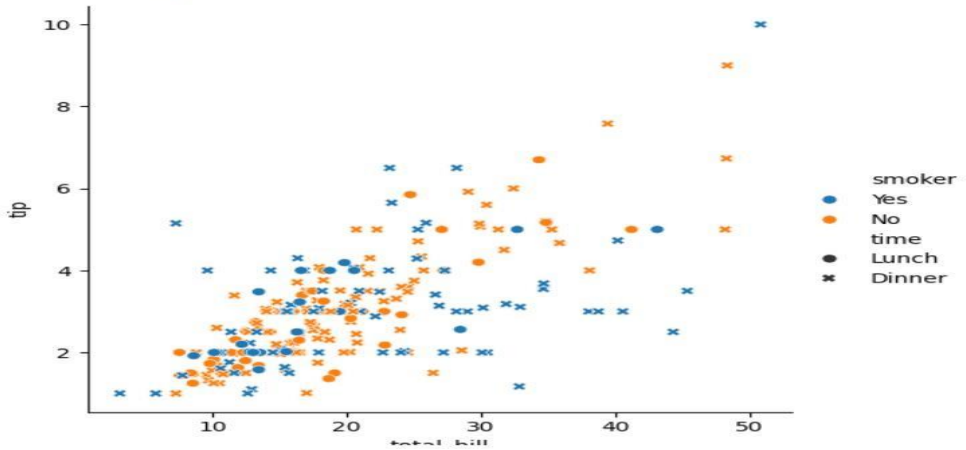
```
sns.relplot(data=ds,x='total_bill',y='tip',hue='smoker',style='smoker')
```

```
<seaborn.axisgrid.FacetGrid at 0x7cba6a781f40>
```



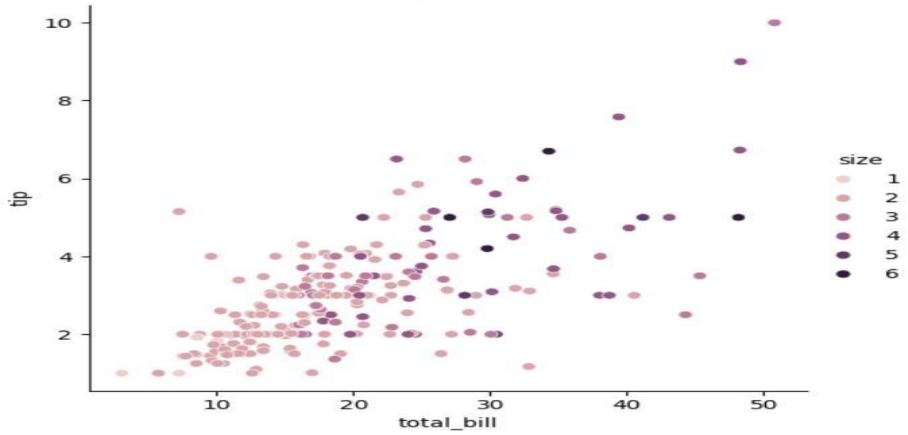
```
sns.relplot(data=ds,x='total_bill',y='tip',hue='smoker',style='time')
```

```
<seaborn.axisgrid.FacetGrid at 0x7cba6a362240>
```



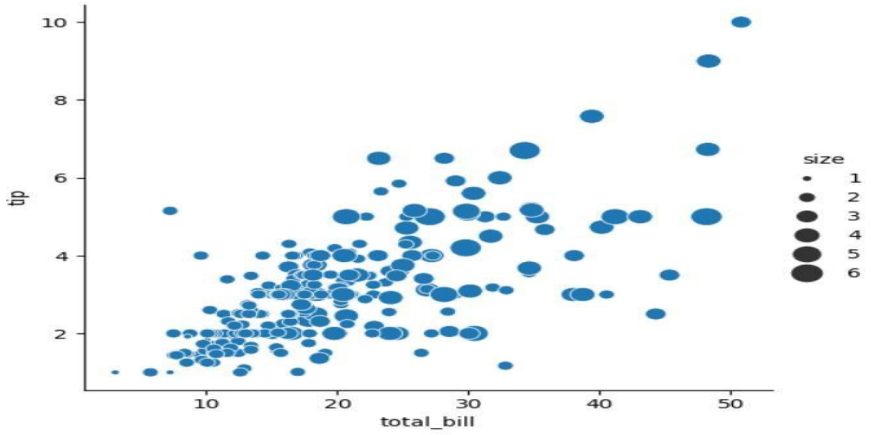

```
sns.relplot(data=ds,x='total_bill',y='tip',hue="size")
```

```
<seaborn.axisgrid.FacetGrid at 0x7cba69bb16d0>
```



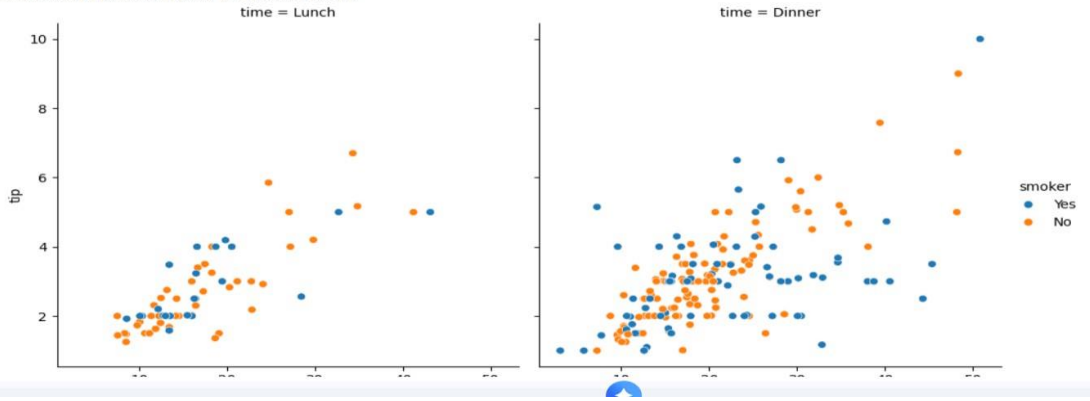

```
sns.relplot(data=ds,x='total_bill',y='tip',size="size",sizes=(15,200))
```

```
<seaborn.axisgrid.FacetGrid at 0x7cba69a58e30>
```



```
sns.relplot(data=ds,x='total_bill',y='tip',hue="smoker",col="time")
```

```
<seaborn.axisgrid.FacetGrid at 0x7cba69a025a0>
```




```
fmri=sns.load_dataset('fmri')
```

```
fmri.head()
```

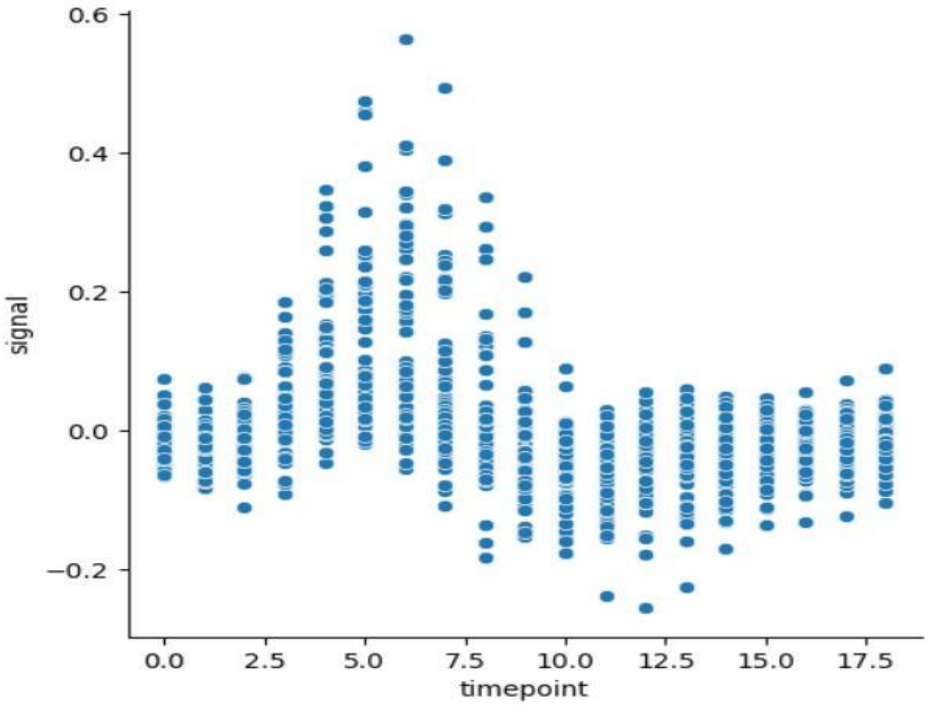
	subject	timepoint	event	region	signal
0	s13	18	stim	parietal	-0.017552
1	s5	14	stim	parietal	-0.080883
2	s12	18	stim	parietal	-0.081033
3	s11	18	stim	parietal	-0.046134
4	s10	18	stim	parietal	-0.037970

```
fmri.shape
```

```
(1064, 5)
```

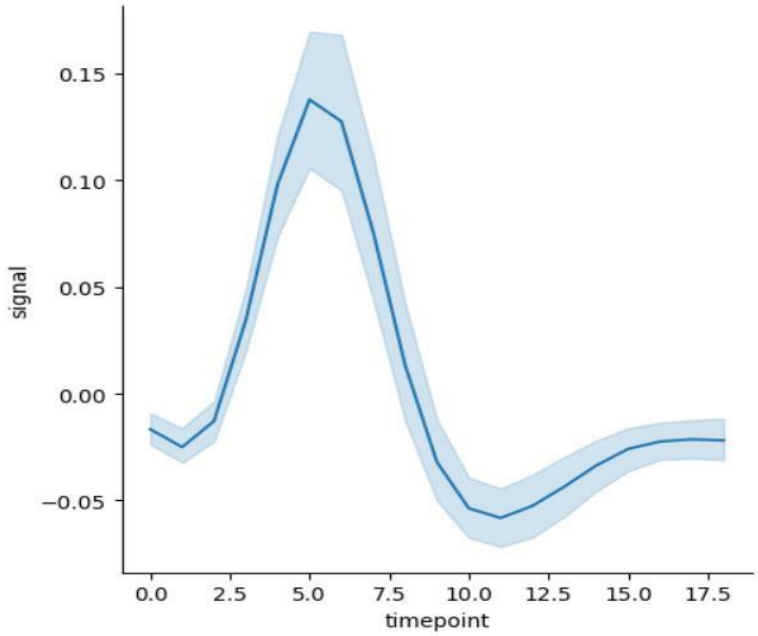
```
sns.relplot(data=fmri,x='timepoint',y='signal')
```

```
<seaborn.axisgrid.FacetGrid at 0x7d5273d8b2f0>
```



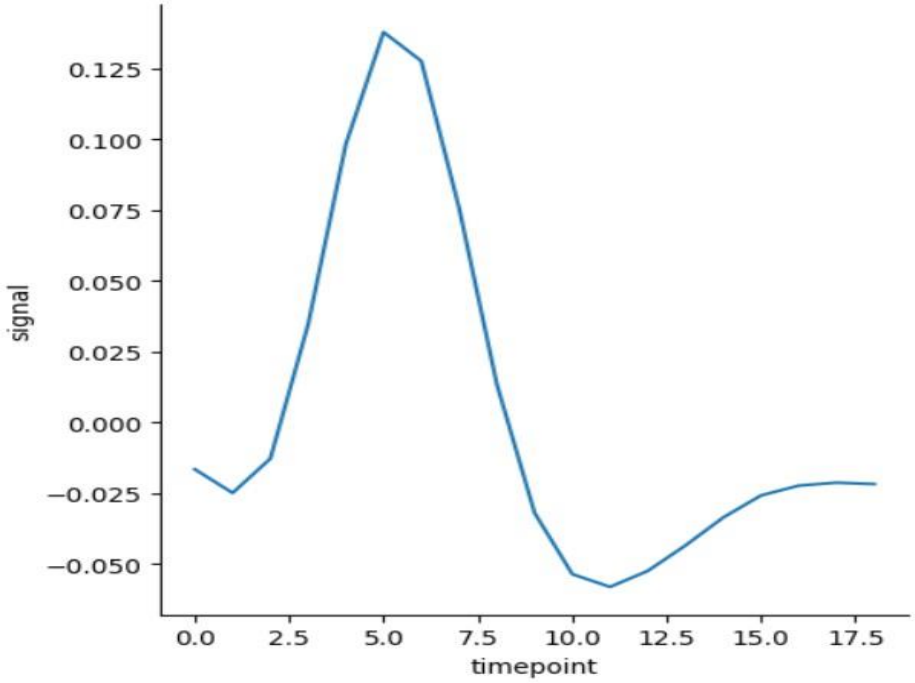
```
sns.relplot(data=fmri,x='timepoint',y='signal',kind='line')
```

```
<seaborn.axisgrid.FacetGrid at 0x7d52739552e0>
```



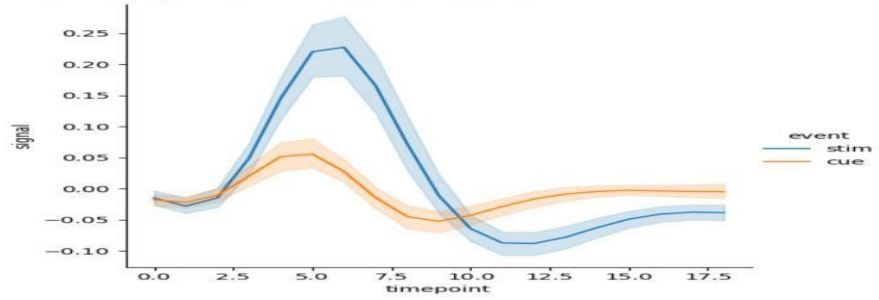

```
sns.relplot(data=fmri,kind='line',x='timepoint',y='signal',errorbar=None)
```

```
<seaborn.axisgrid.FacetGrid at 0x7d5273a23d10>
```



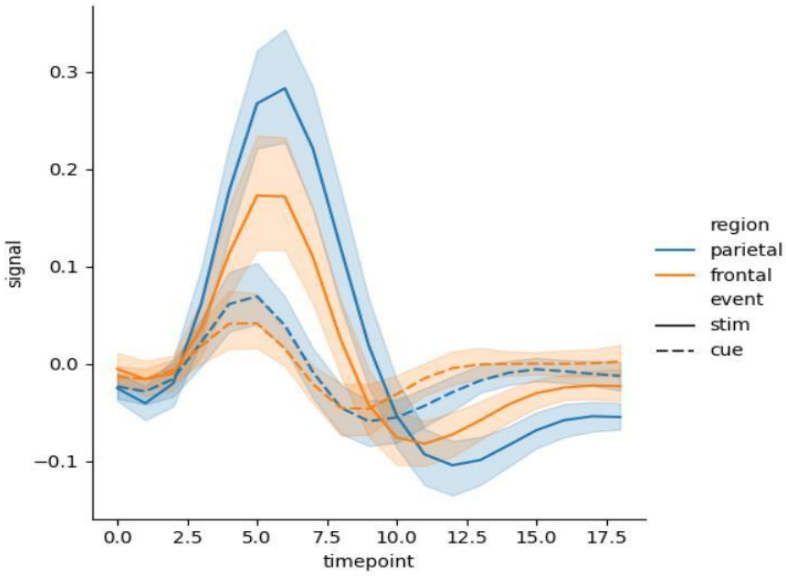
```
sns.relplot(data=fmri,kind='line',x='timepoint',y='signal',hue='event')
```

```
<seaborn.axisgrid.FacetGrid at 0x7d52739b3590>
```



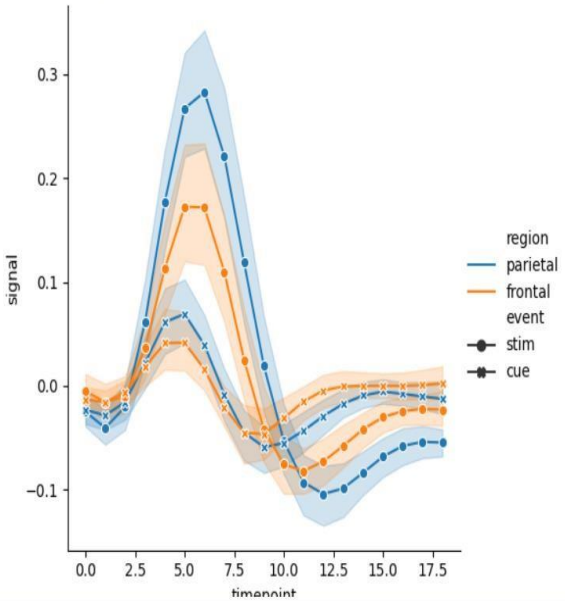

```
sns.relplot(data=fmri,kind='line',x='timepoint',y='signal',hue='region',style='event')
```

```
<seaborn.axisgrid.FacetGrid at 0x7d52714e7380>
```




```
sns.relplot(data=fmri,kind='line',x='timepoint',y='signal',hue='region',style='event',dashes=False,markers=True)
```

```
<seaborn.axisgrid.FacetGrid at 0x7d527140fad0>
```



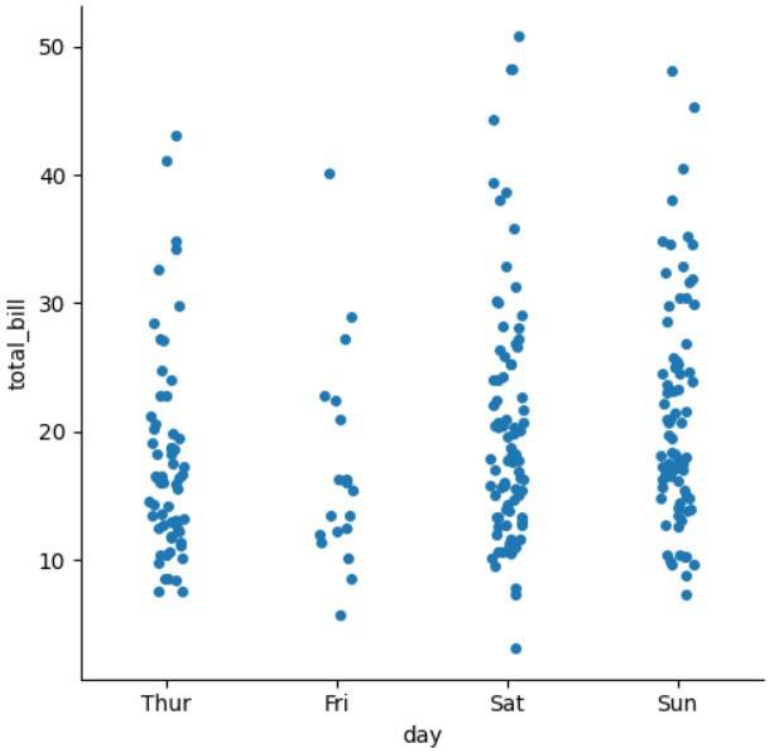
```
import seaborn as sns
tips=sns.load_dataset('tips')
tips.head()
```

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4

Catplot:creates categorical plots(swarm,box,violin):

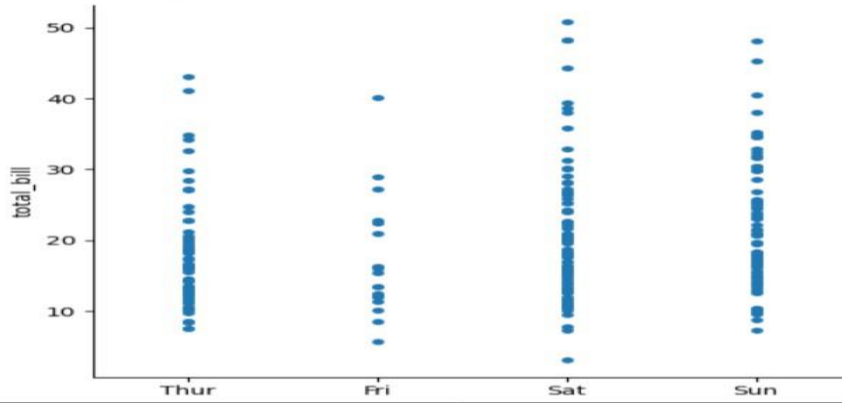
```
sns.catplot(data=tips,x='day',y='total_bill')
```

```
<seaborn.axisgrid.FacetGrid at 0x7fee23521670>
```



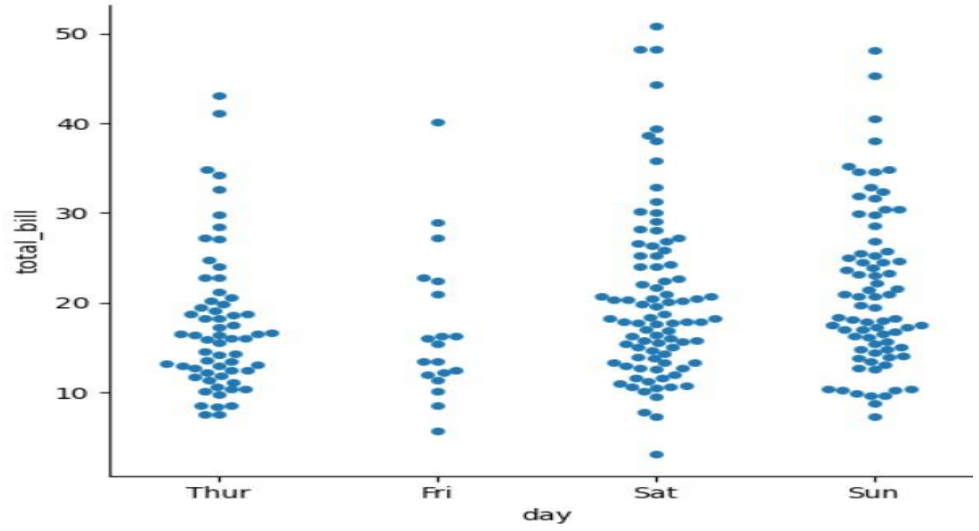
```
sns.catplot(data=tips,x='day',y='total_bill',jitter=False)
```

```
<seaborn.axisgrid.FacetGrid at 0x7fee229975c0>
```



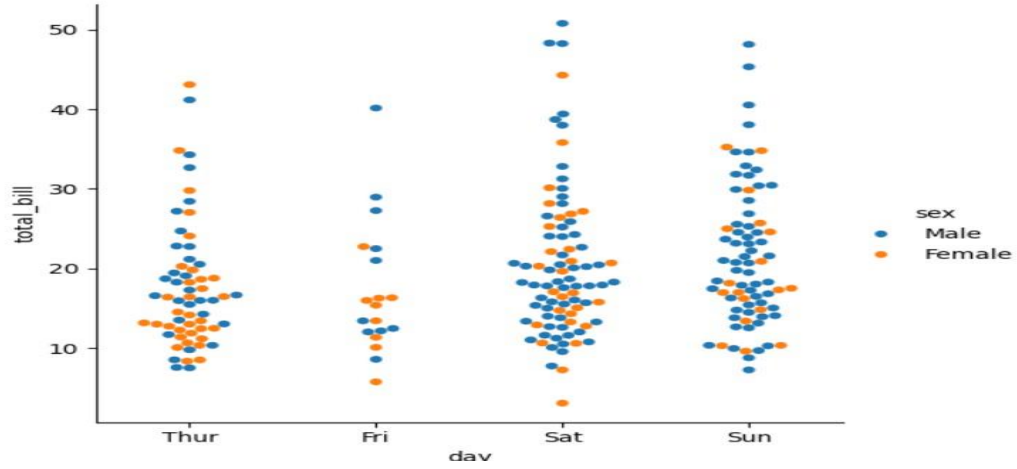

```
sns.catplot(data=tips,x='day',y='total_bill',kind='swarm')
```

```
<seaborn.axisgrid.FacetGrid at 0x7fee22f09970>
```



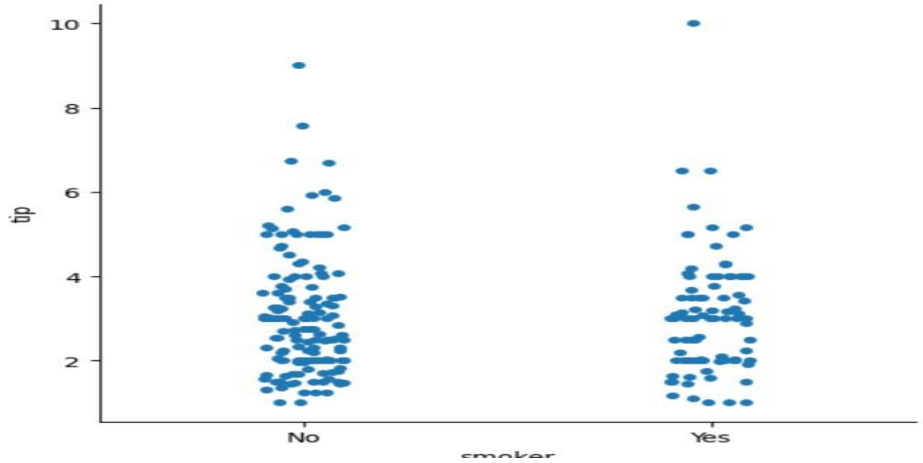

```
sns.catplot(data=tips,x='day',y='total_bill',hue='sex',kind='swarm')
```

```
<seaborn.axisgrid.FacetGrid at 0x7fee20744200>
```



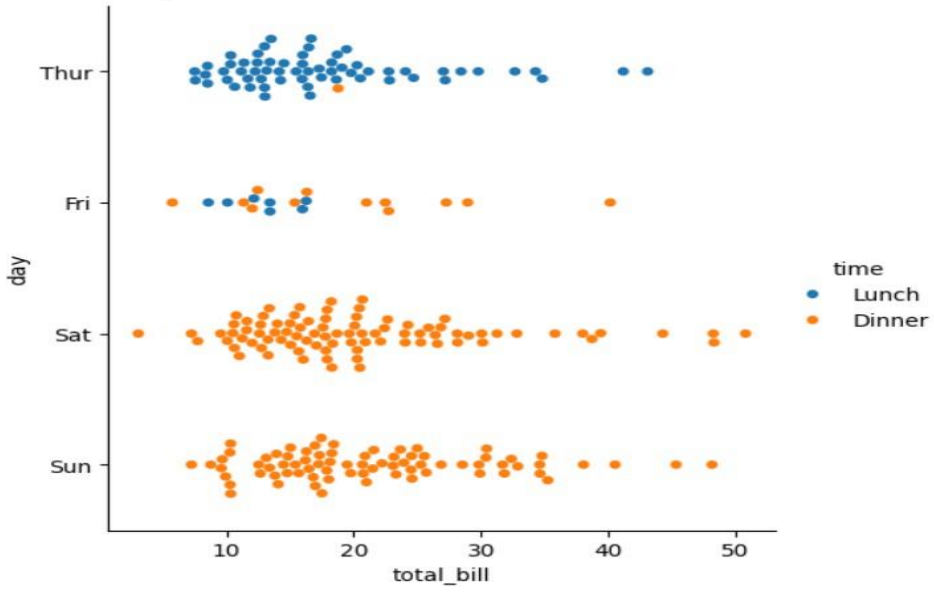
```
sns.catplot(data=tips,x='smoker',y='tip',order=['No','Yes'])
```

```
<seaborn.axisgrid.FacetGrid at 0x7fee22886600>
```



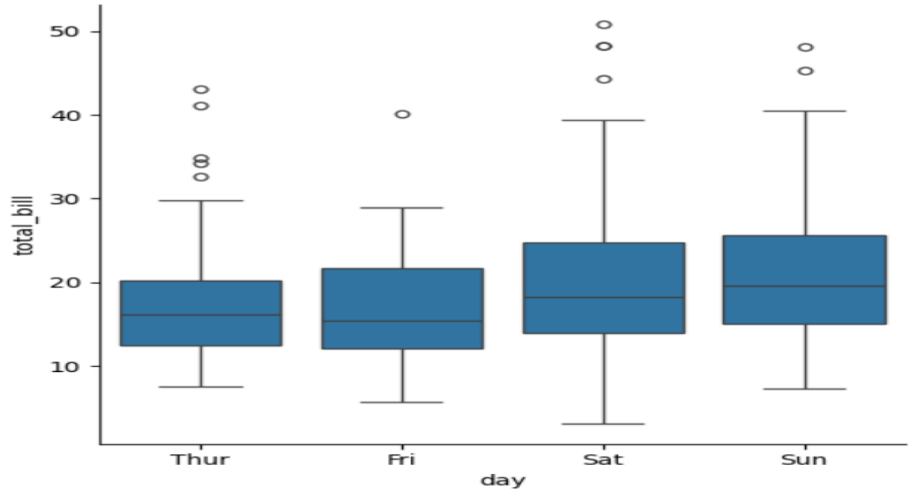

```
sns.catplot(data=tips,x='total_bill',y='day',hue='time',kind='swarm')
```

```
<seaborn.axisgrid.FacetGrid at 0x7fee2286e7b0>
```



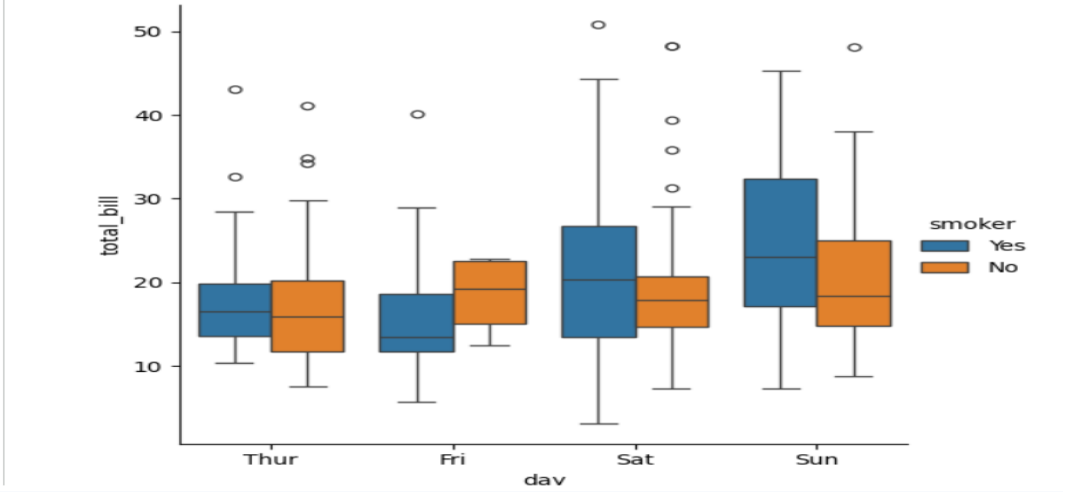

```
sns.catplot(data=tips,x='day',y='total_bill',kind='box')
```

```
<seaborn.axisgrid.FacetGrid at 0x7fee228c4050>
```



```
sns.catplot(data=tips,x='day',y='total_bill',hue='smoker',kind='box')
```

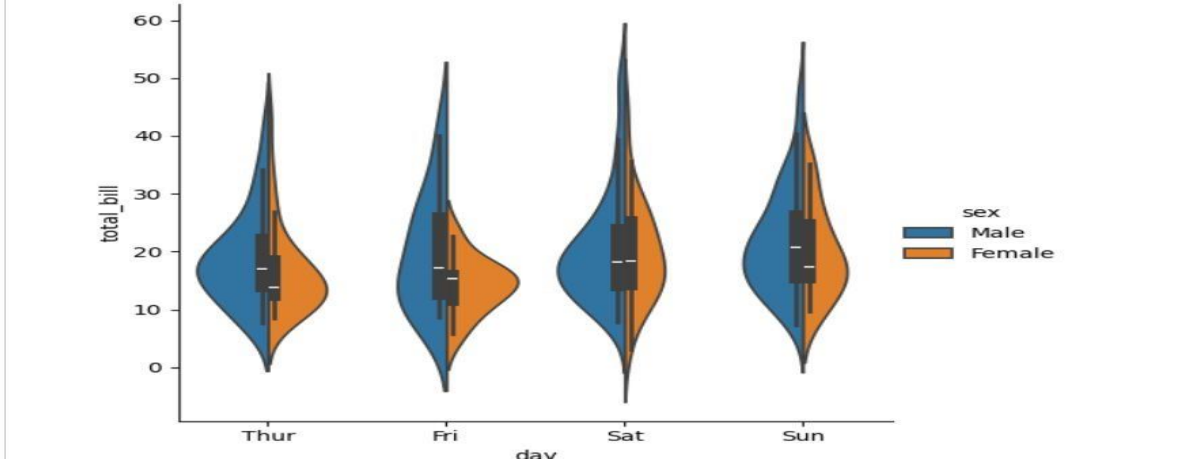
```
<seaborn.axisgrid.FacetGrid at 0x7fee206b1b20>
```



Violin plot:

```
sns.catplot(data=tips,x='day',y='total_bill',hue='sex',kind='violin',split='True')
```

```
<seaborn.axisgrid.FacetGrid at 0x7fee204f9c10>
```



Prepared By
Gummadi Chandana