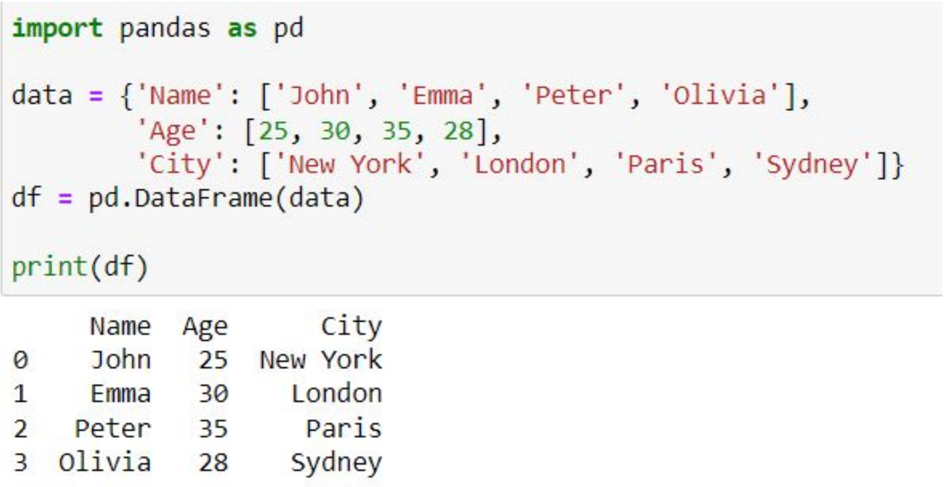
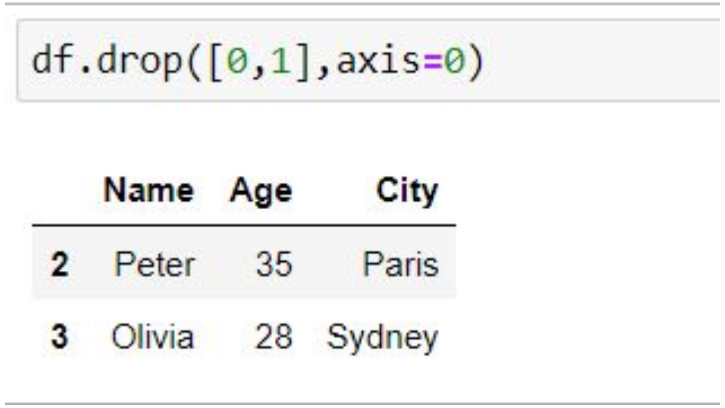
Dropping Rows



```python
import pandas as pd

data = {'Name': ['John', 'Emma', 'Peter', 'Olivia'],
        'Age': [25, 30, 35, 28],
        'City': ['New York', 'London', 'Paris', 'Sydney']}
df = pd.DataFrame(data)

print(df)
```

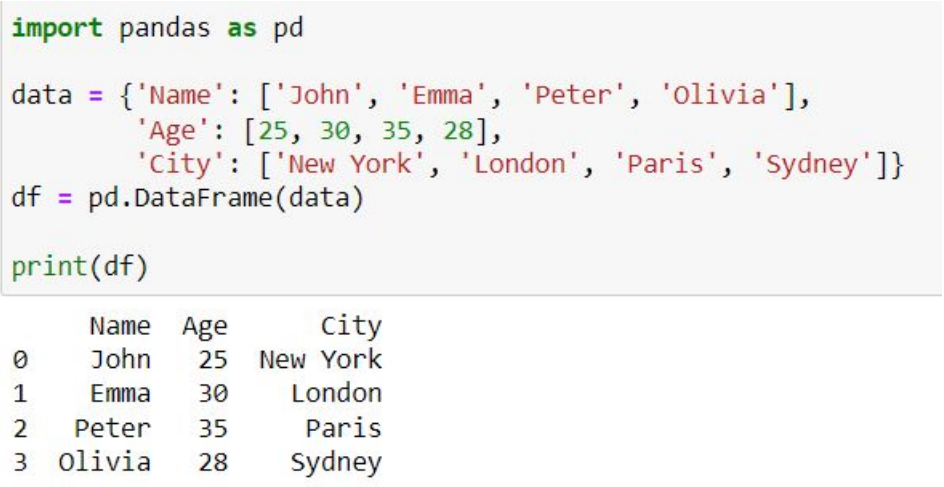
```
     Name  Age      City
0    John   25  New York
1    Emma   30    London
2   Peter   35     Paris
3  Olivia   28    Sydney
```



```python
df.drop([0,1],axis=0)
```

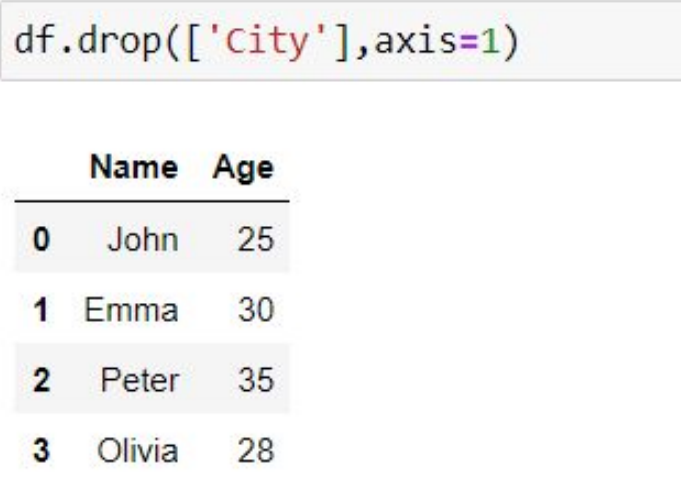
	Name	Age	City
2	Peter	35	Paris
3	Olivia	28	Sydney

Dropping Columns



```python
import pandas as pd

data = {'Name': ['John', 'Emma', 'Peter', 'Olivia'],
        'Age': [25, 30, 35, 28],
        'City': ['New York', 'London', 'Paris', 'Sydney']}
df = pd.DataFrame(data)

print(df)
```

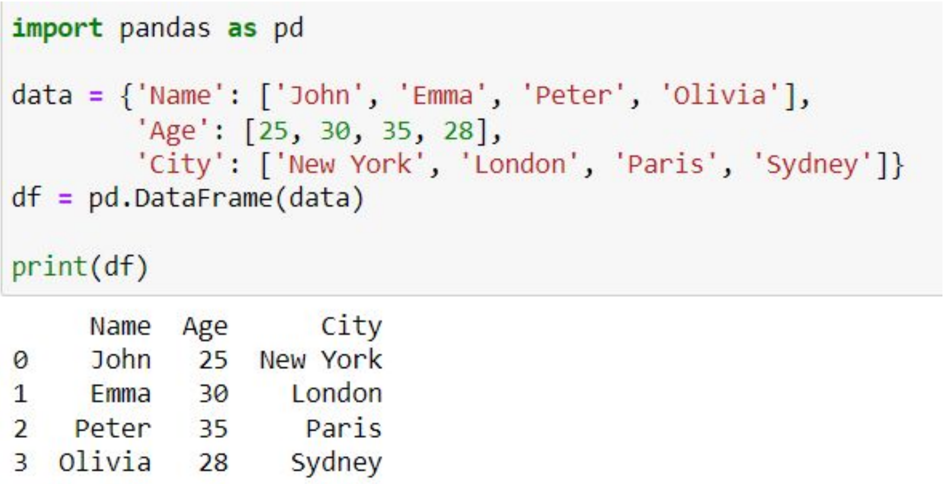
```
     Name  Age      City
0    John   25  New York
1    Emma   30    London
2   Peter   35     Paris
3  Olivia   28    Sydney
```



```python
df.drop(['City'],axis=1)
```

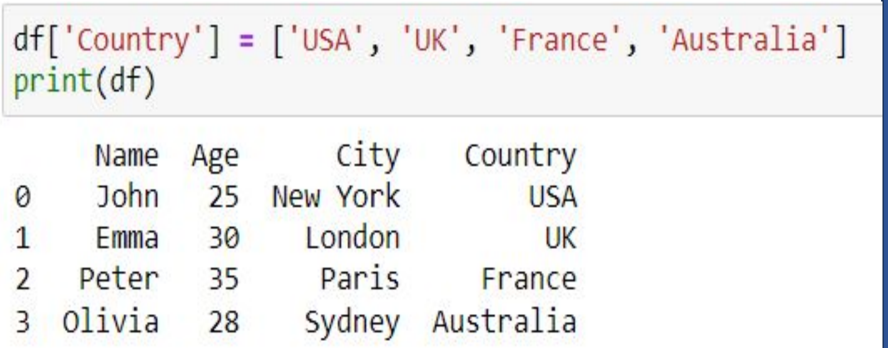
	Name	Age
0	John	25
1	Emma	30
2	Peter	35
3	Olivia	28

Adding new Columns



```
import pandas as pd

data = {'Name': ['John', 'Emma', 'Peter', 'Olivia'],
        'Age': [25, 30, 35, 28],
        'City': ['New York', 'London', 'Paris', 'Sydney']}
df = pd.DataFrame(data)

print(df)
```

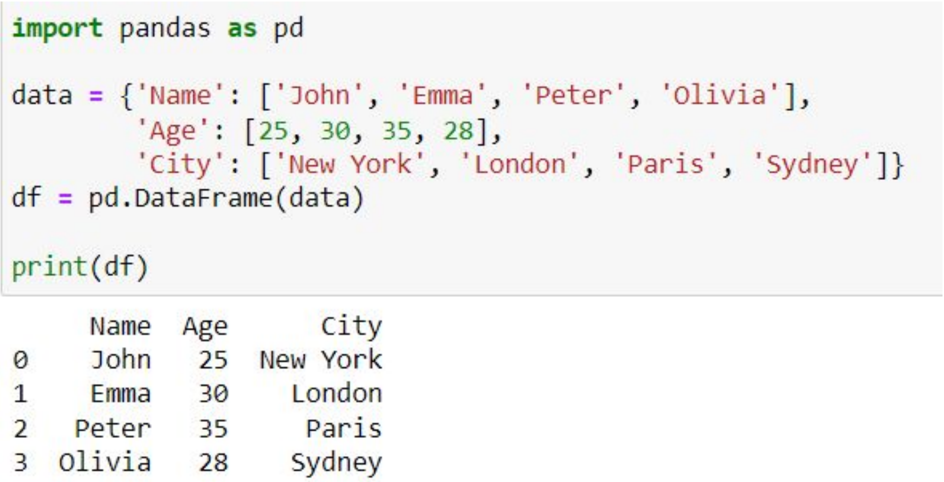
```
     Name  Age      City
0    John   25  New York
1    Emma   30    London
2   Peter   35     Paris
3  Olivia   28    Sydney
```



```
df['Country'] = ['USA', 'UK', 'France', 'Australia']
print(df)
```

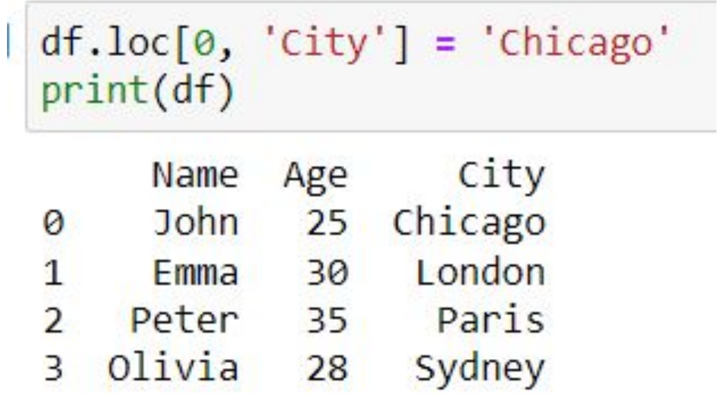
```
     Name  Age      City    Country
0    John   25  New York        USA
1    Emma   30    London         UK
2   Peter   35     Paris     France
3  Olivia   28    Sydney  Australia
```

Modifying values in a Dataframe



```
import pandas as pd

data = {'Name': ['John', 'Emma', 'Peter', 'Olivia'],
        'Age': [25, 30, 35, 28],
        'City': ['New York', 'London', 'Paris', 'Sydney']}
df = pd.DataFrame(data)

print(df)
```

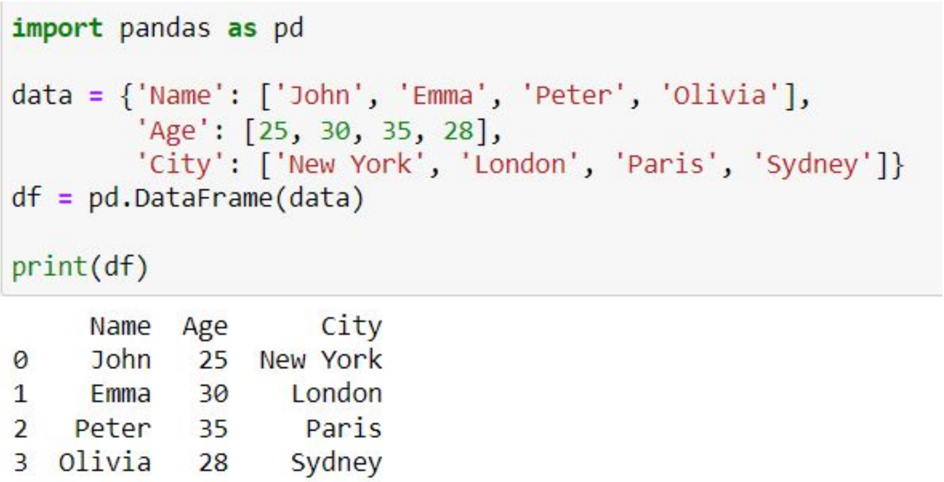
```
     Name  Age      City
0    John   25  New York
1    Emma   30    London
2   Peter   35     Paris
3  Olivia   28    Sydney
```



```
df.loc[0, 'City'] = 'Chicago'
print(df)
```

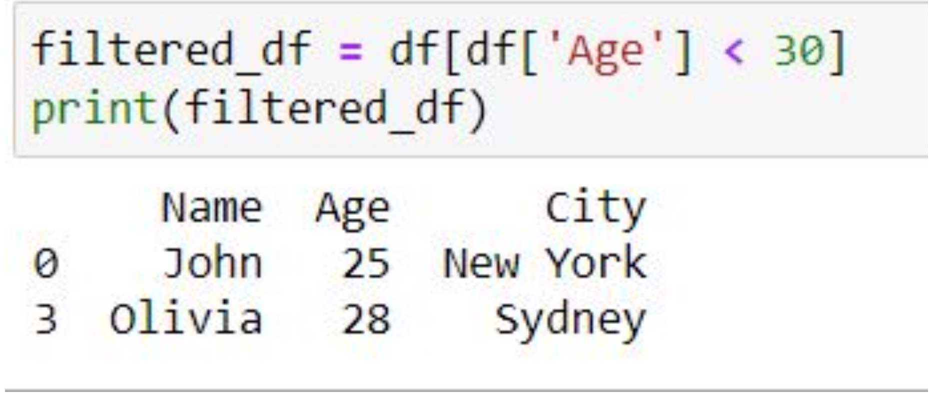
```
     Name  Age     City
0    John   25  Chicago
1    Emma   30   London
2   Peter   35    Paris
3  Olivia   28   Sydney
```

Filtering rows in a Dataframe



```
import pandas as pd

data = {'Name': ['John', 'Emma', 'Peter', 'Olivia'],
        'Age': [25, 30, 35, 28],
        'City': ['New York', 'London', 'Paris', 'Sydney']}
df = pd.DataFrame(data)

print(df)
```

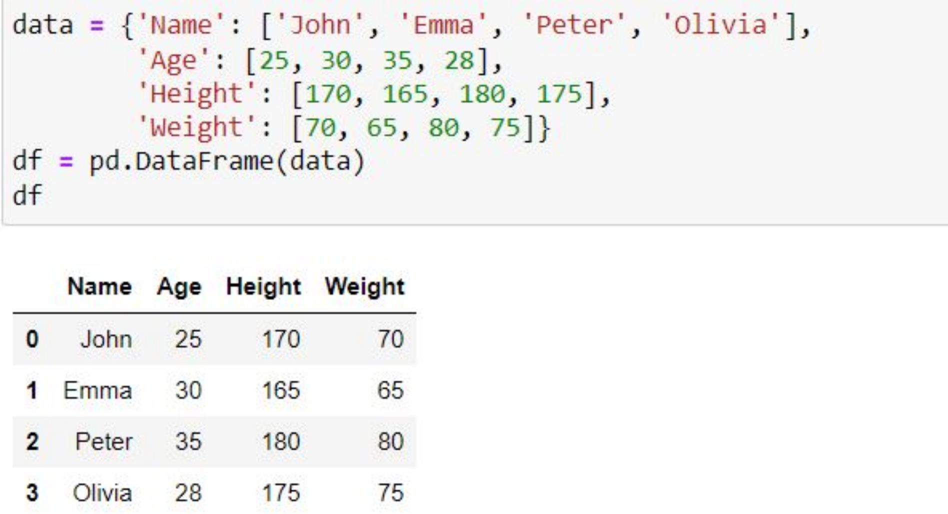
```
     Name  Age      City
0    John   25  New York
1    Emma   30    London
2   Peter   35     Paris
3  Olivia   28    Sydney
```



```
filtered_df = df[df['Age'] < 30]
print(filtered_df)
```

```
     Name  Age      City
0    John   25  New York
3  Olivia   28    Sydney
```

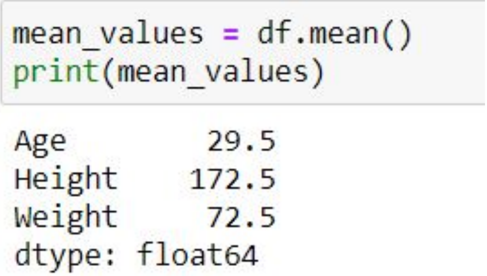
More Pandas Functions



```
data = {'Name': ['John', 'Emma', 'Peter', 'Olivia'],
        'Age': [25, 30, 35, 28],
        'Height': [170, 165, 180, 175],
        'Weight': [70, 65, 80, 75]}
df = pd.DataFrame(data)
df
```

	Name	Age	Height	Weight
0	John	25	170	70
1	Emma	30	165	65
2	Peter	35	180	80
3	Olivia	28	175	75

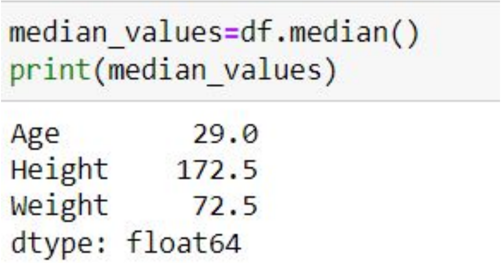
Mean



```
mean_values = df.mean()
print(mean_values)
```

```
Age        29.5
Height    172.5
Weight     72.5
dtype: float64
```

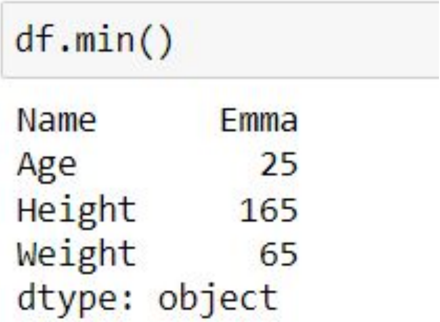
Median



```
median_values=df.median()
print(median_values)
```

```
Age        29.0
Height    172.5
Weight     72.5
dtype: float64
```

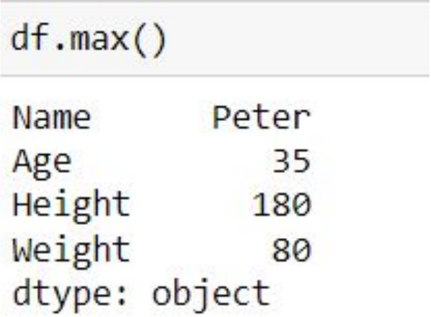
Minimum



```
df.min()
```

```
Name       Emma
Age          25
Height      165
Weight       65
dtype: object
```

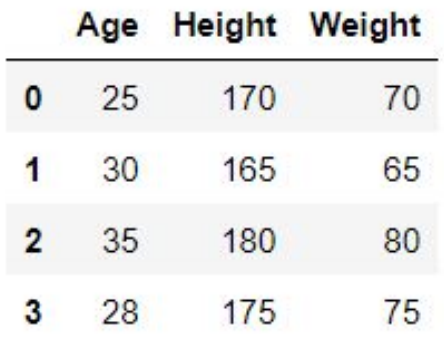
Maximum



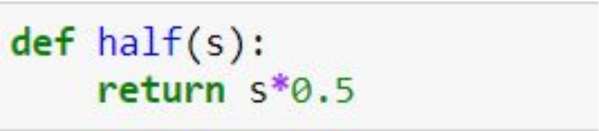
```
df.max()
```

```
Name      Peter
Age          35
Height      180
Weight       80
dtype: object
```

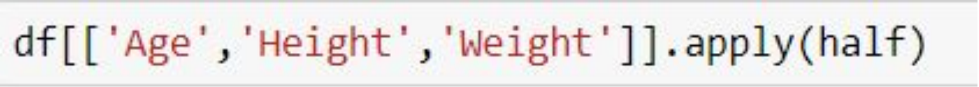
More Pandas Functions



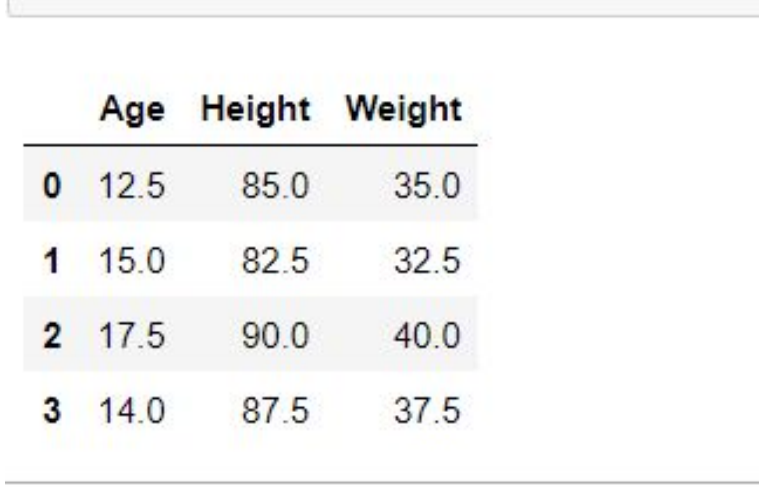
	Age	Height	Weight
0	25	170	70
1	30	165	65
2	35	180	80
3	28	175	75



```python
def half(s):
    return s*0.5
```



```python
df[['Age','Height','Weight']].apply(half)
```



	Age	Height	Weight
0	12.5	85.0	35.0
1	15.0	82.5	32.5
2	17.5	90.0	40.0
3	14.0	87.5	37.5

Python Matplotlib

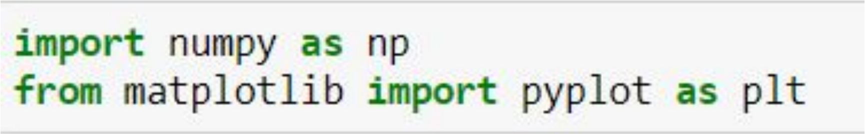
Matplotlib is a python library used for data visualization

25%
Bar Graph
Histogram
Scatter Plot
Area Plot
Pie Plot

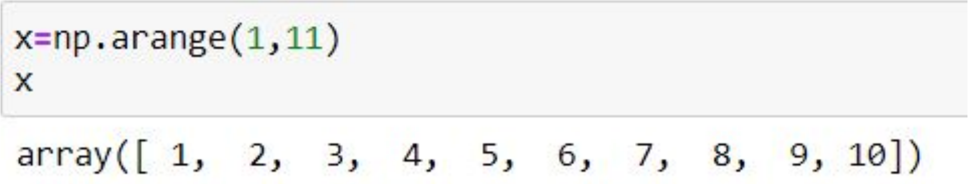
You can create bar-plots, scatter-plots, histograms and a lot more with matplotlib

Line Plot

Line charts are used to represent the relation between two data X and Y on a different axis.

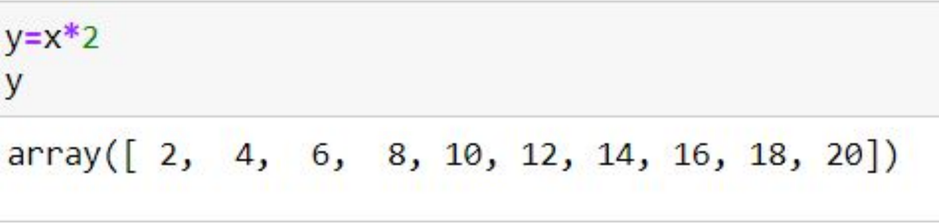


```
import numpy as np
from matplotlib import pyplot as plt
```



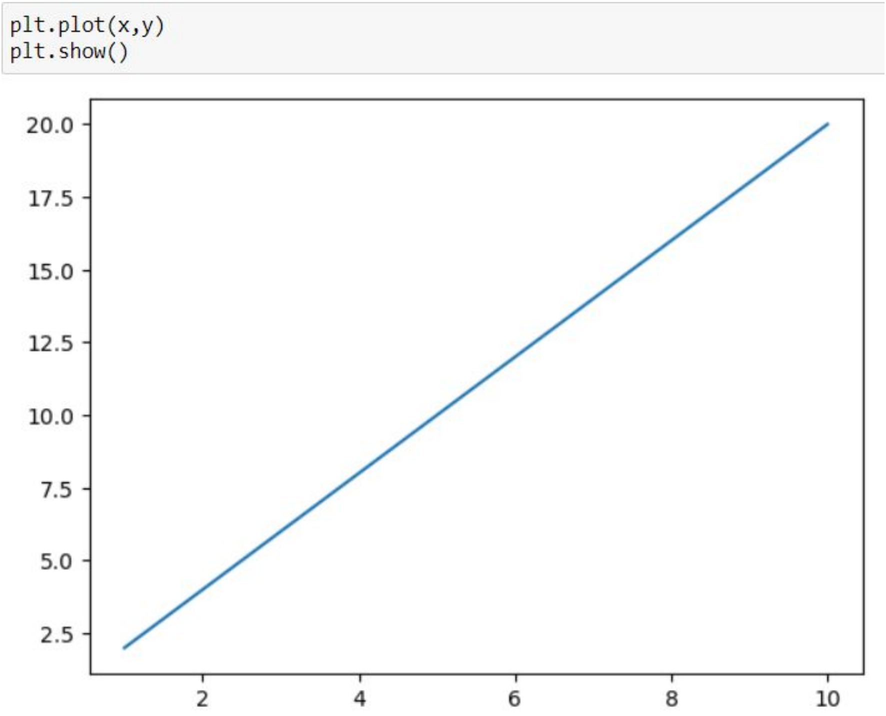
```
x=np.arange(1,11)
x
```

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



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

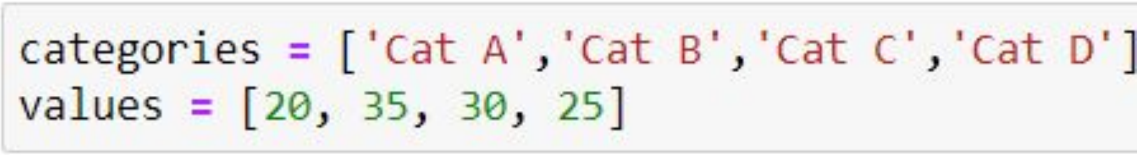
```
array([ 2,  4,  6,  8, 10, 12, 14, 16, 18, 20])
```



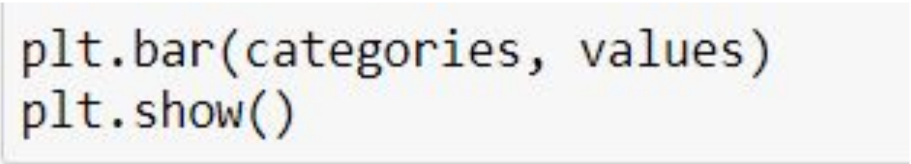
```
plt.plot(x,y)
plt.show()
```

Bar Plot

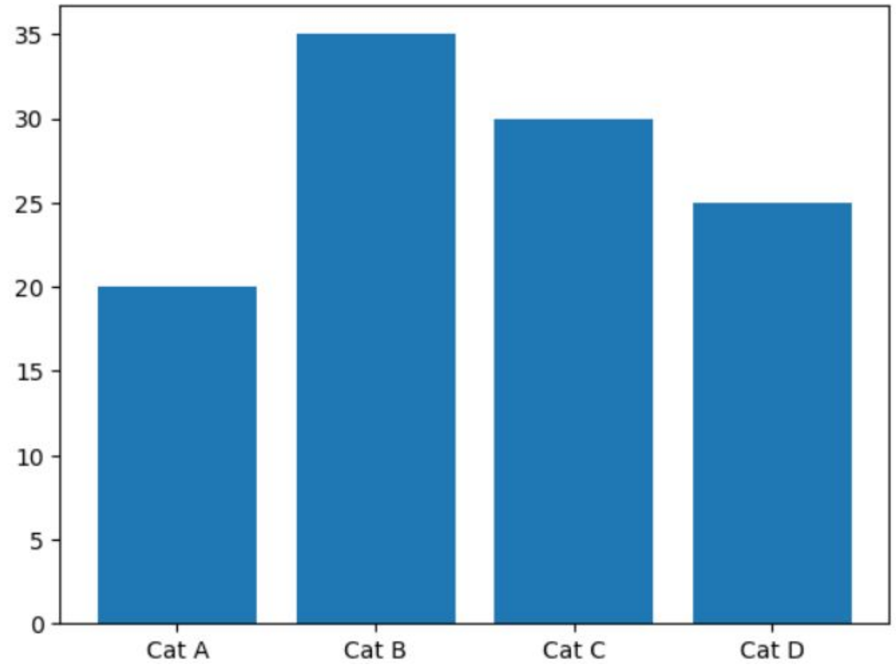
- It a graph that represents the category of data with rectangular bars with lengths and heights that is proportional to the values which they represent.
- It describes the comparisons between the discrete categories.
- One of the axis represents the specific categories being compared, other axis represents the measured values corresponding to those categories.



```
categories = ['Cat A','Cat B','Cat C','Cat D']
values = [20, 35, 30, 25]
```

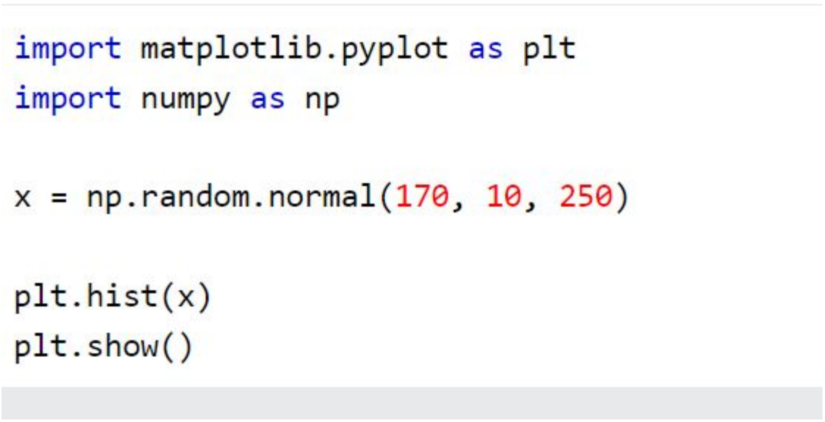


```
plt.bar(categories, values)
plt.show()
```

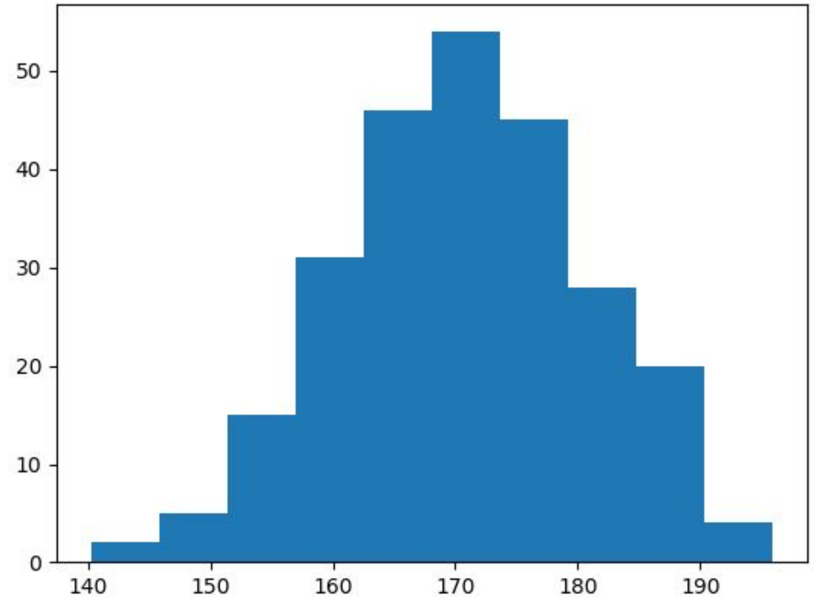


Histogram

A histogram is a graph showing *frequency* distributions.
It is a graph showing the number of observations within each given interval.

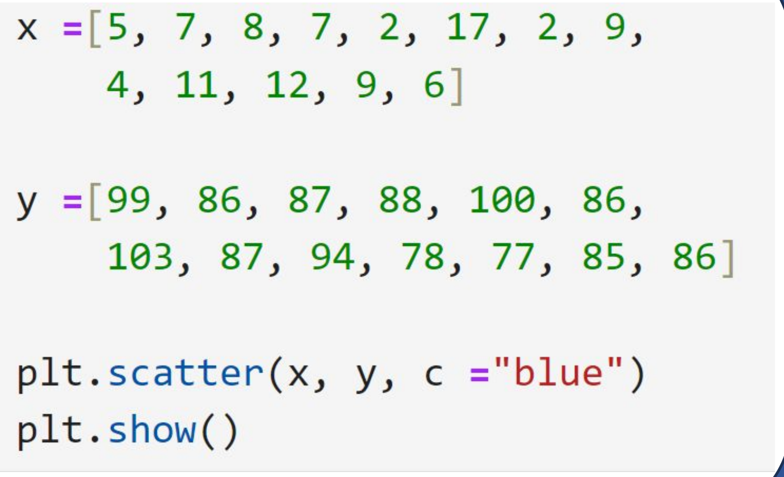


```
import matplotlib.pyplot as plt
import numpy as np

x = np.random.normal(170, 10, 250)

plt.hist(x)
plt.show()
```

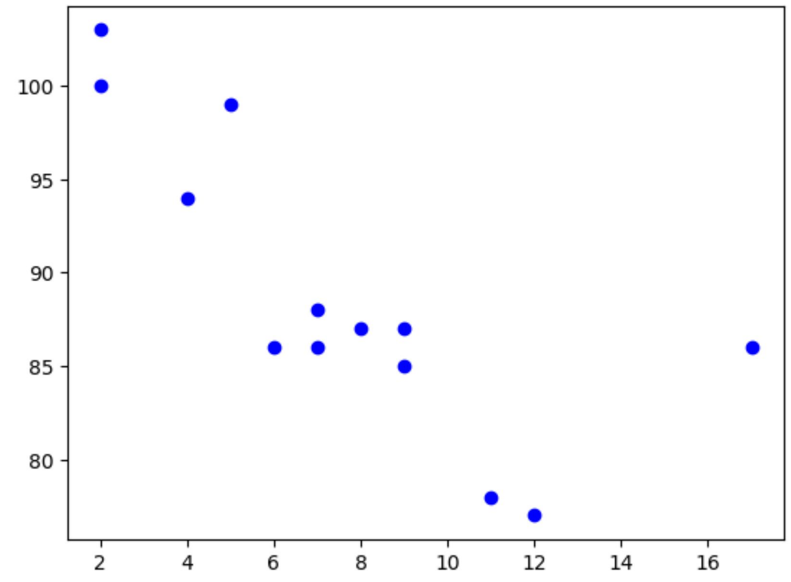


Scatter Plot

- Scatter plots are used to observe relationship between variables and uses dots to represent the relationship between them.
- The **scatter()** method in the matplotlib library is used to draw a scatter plot. Scatter plots are widely used to represent relation among variables and how change in one affects the other.

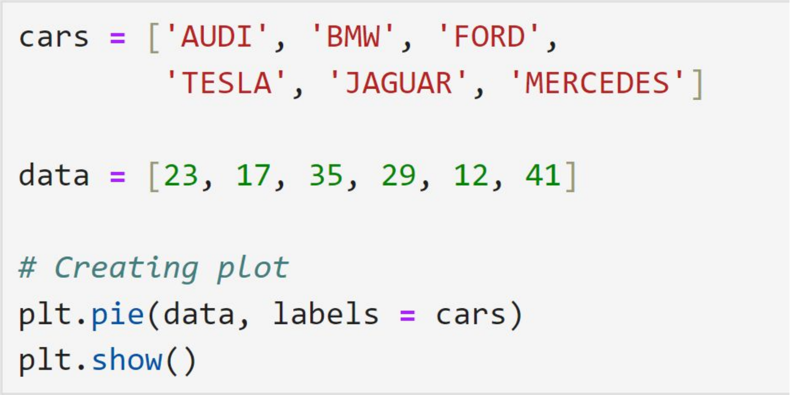


```
x =[5, 7, 8, 7, 2, 17, 2, 9,
    4, 11, 12, 9, 6]

y =[99, 86, 87, 88, 100, 86,
    103, 87, 94, 78, 77, 85, 86]

plt.scatter(x, y, c ="blue")
plt.show()
```

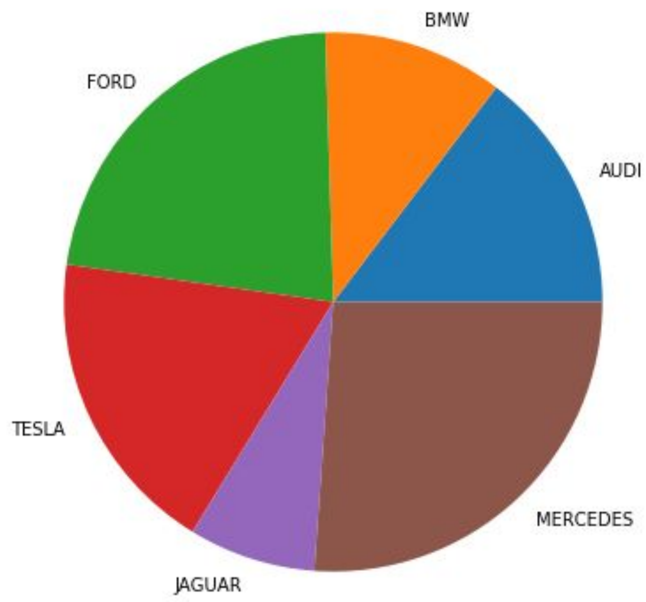


Pie Chart

- A Pie Chart is a circular statistical plot that can display only one series of data.
- It illustrates data proportions using wedges within a circular layout. Each wedge's area represents a percentage of the whole dataset.
- Commonly used in business presentations for summarizing categories like sales, survey results, and resource allocation.

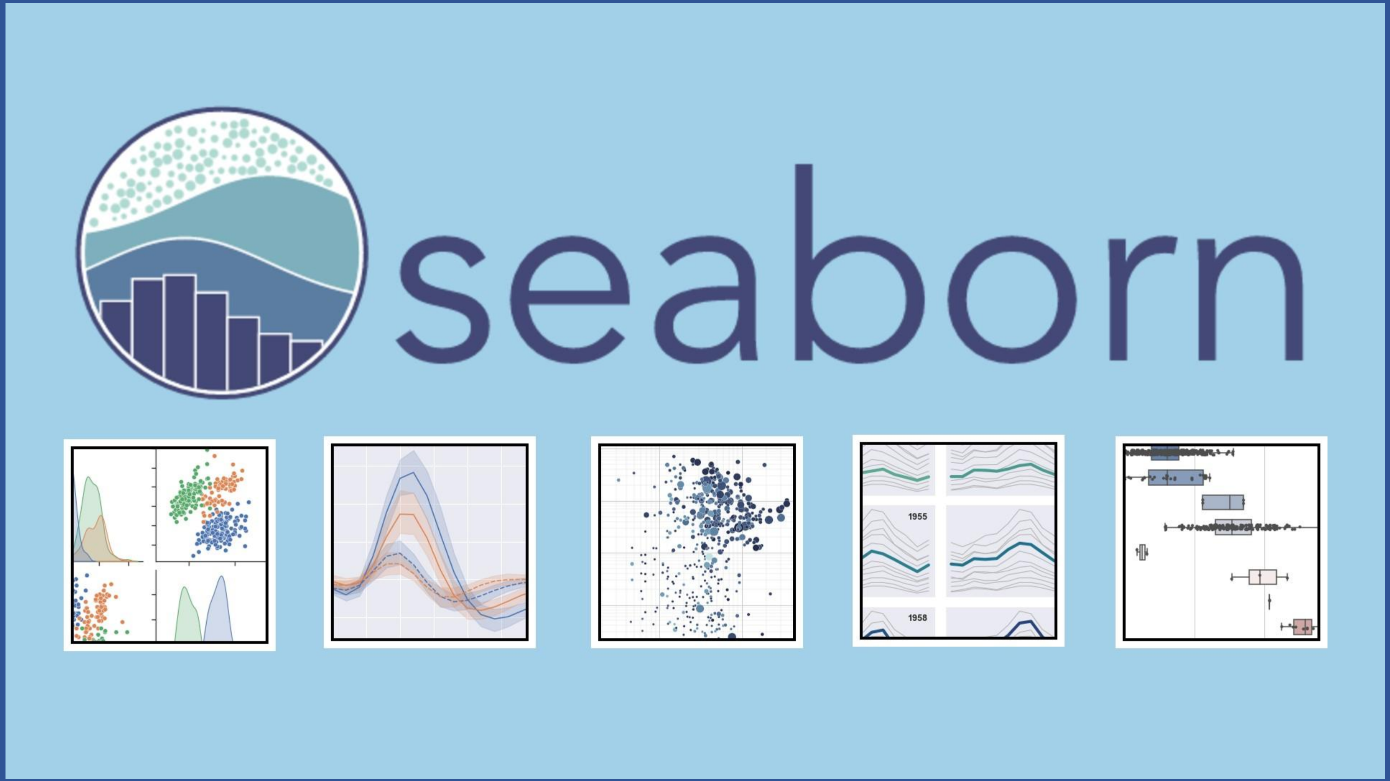


```
cars = ['AUDI', 'BMW', 'FORD',
        'TESLA', 'JAGUAR', 'MERCEDES']

data = [23, 17, 35, 29, 12, 41]

# Creating plot
plt.pie(data, labels = cars)
plt.show()
```



Python Seaborn

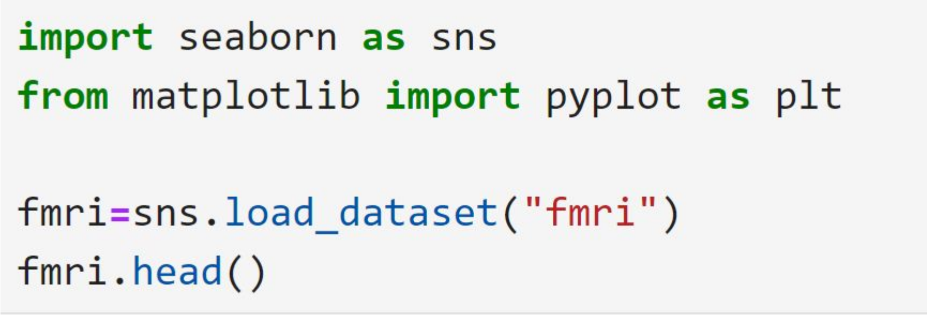
Seaborn is an amazing visualization library for statistical graphics plotting in Python. It is built on the top of matplotlib library and also closely integrated into the data structures from pandas.



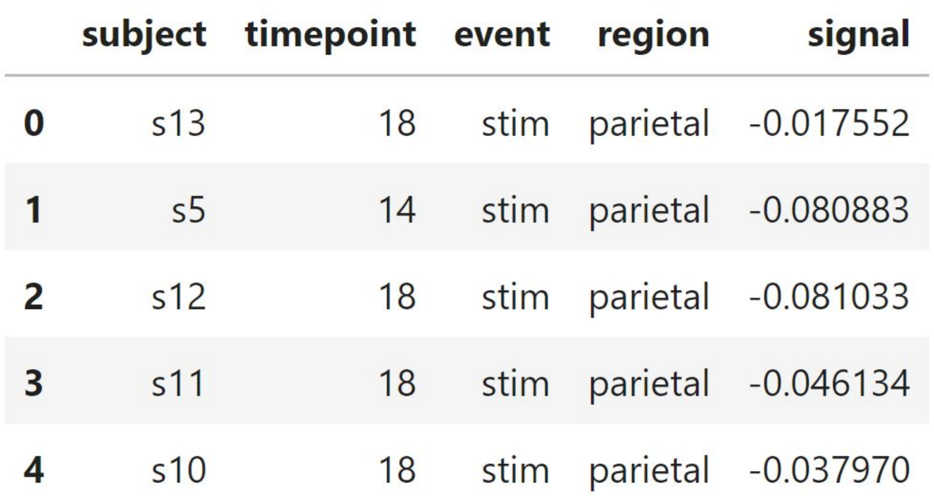
For Python Environment:
pip install seaborn

Grow Data Skills

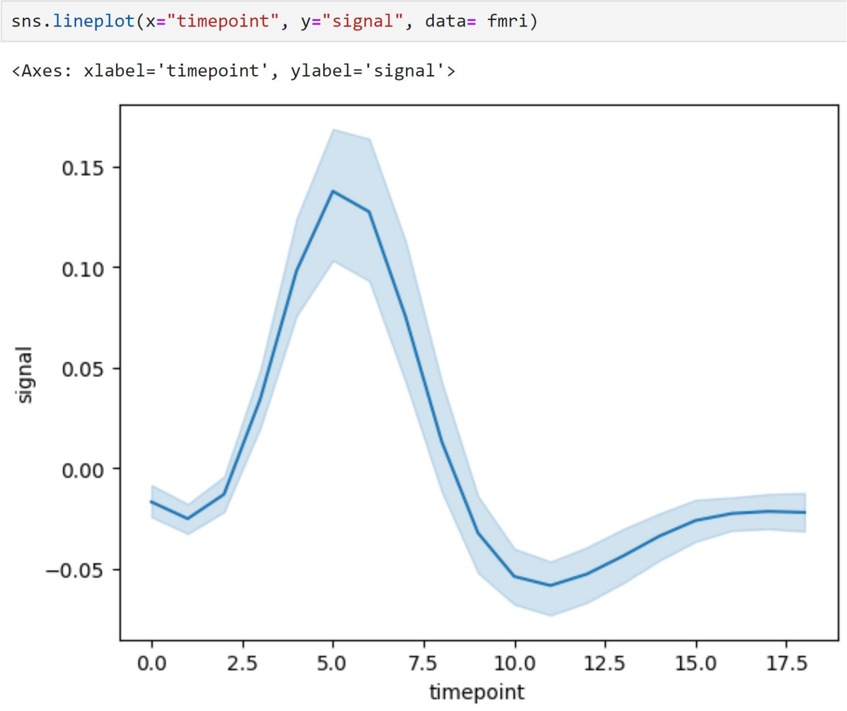
Seaborn Line Plot



```python
import seaborn as sns
from matplotlib import pyplot as plt

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

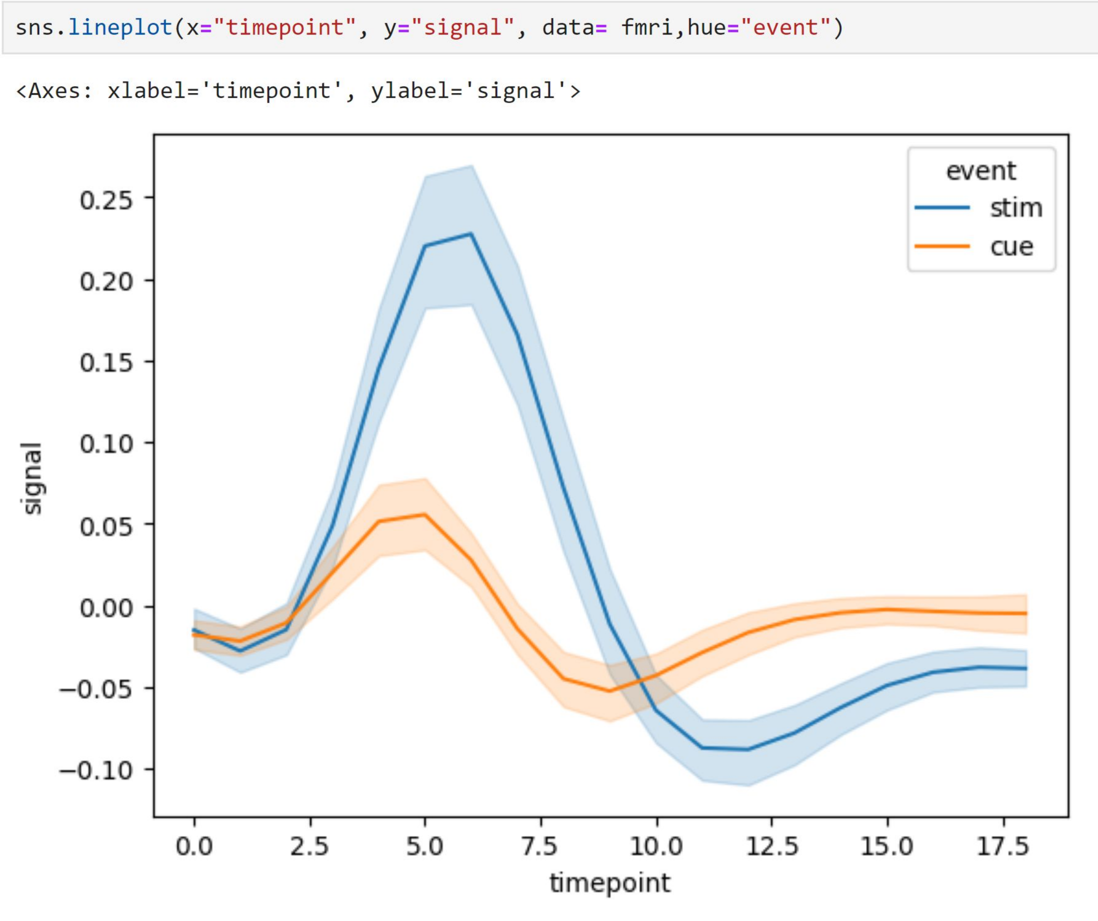


```python
sns.lineplot(x="timepoint", y="signal", data= fmri)
```

```
<Axes: xlabel='timepoint', ylabel='signal'>
```

Seaborn Line Plot

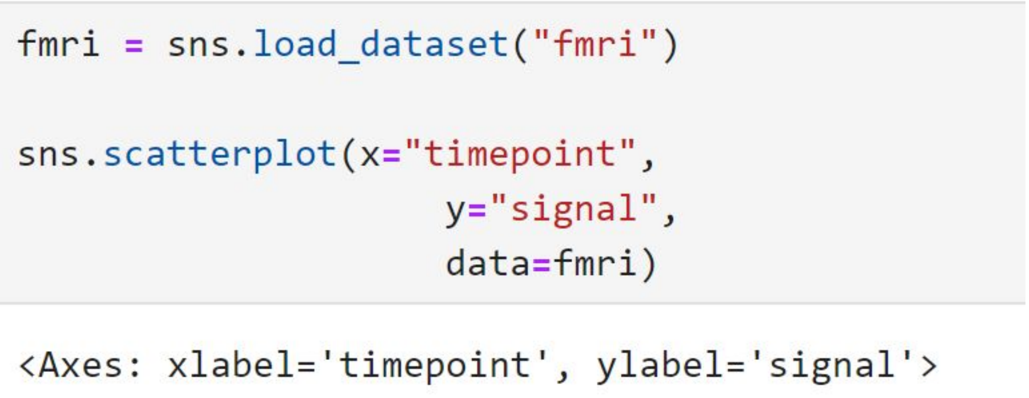
Grouping Data with “hue”



```
sns.lineplot(x="timepoint", y="signal", data= fmri,hue="event")
```

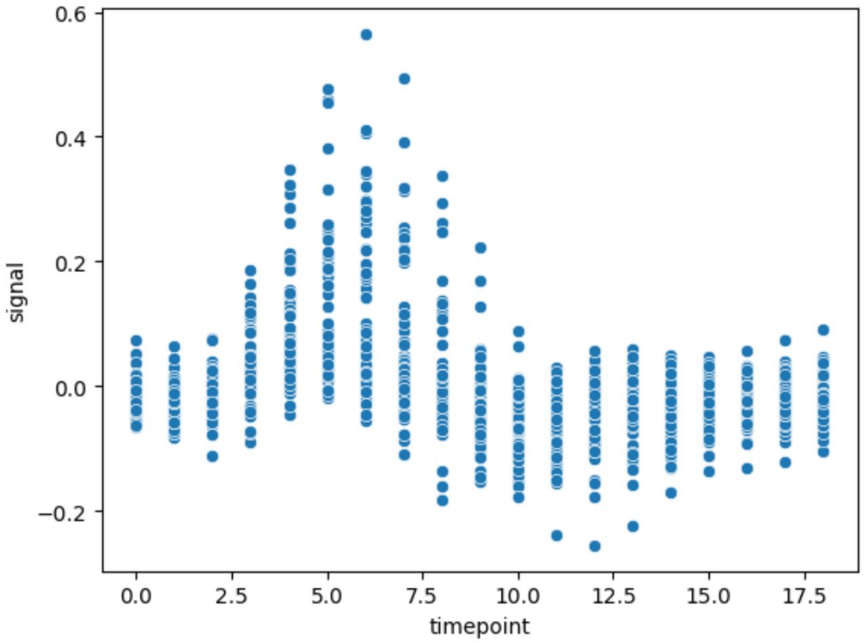
```
<Axes: xlabel='timepoint', ylabel='signal'>
```

Seaborn Scatter Plot

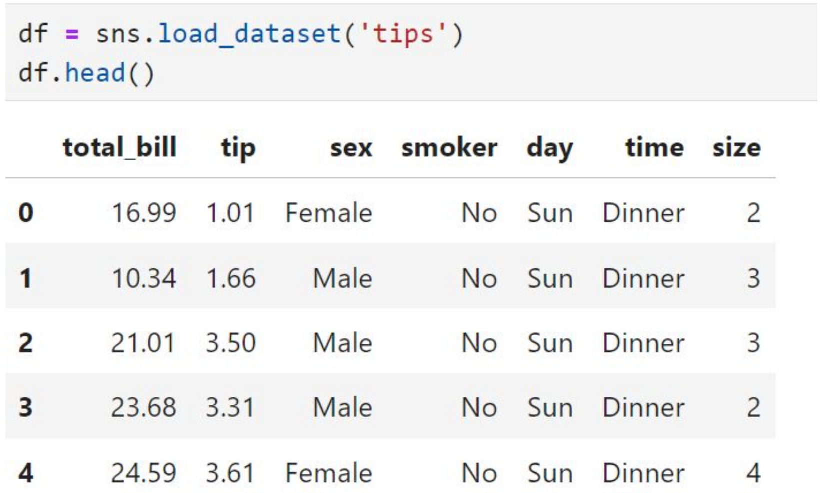


```
fmri = sns.load_dataset("fmri")

sns.scatterplot(x="timepoint",
                y="signal",
                data=fmri)
```

```
<Axes: xlabel='timepoint', ylabel='signal'>
```

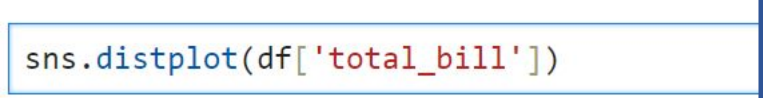


Seaborn Histogram/Distplot

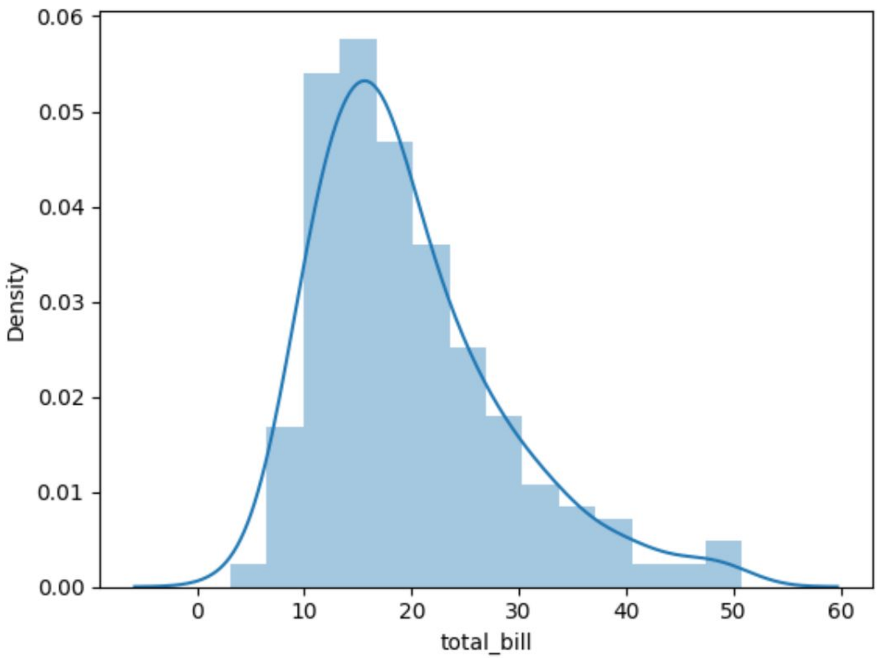


```
df = sns.load_dataset('tips')
df.head()
```

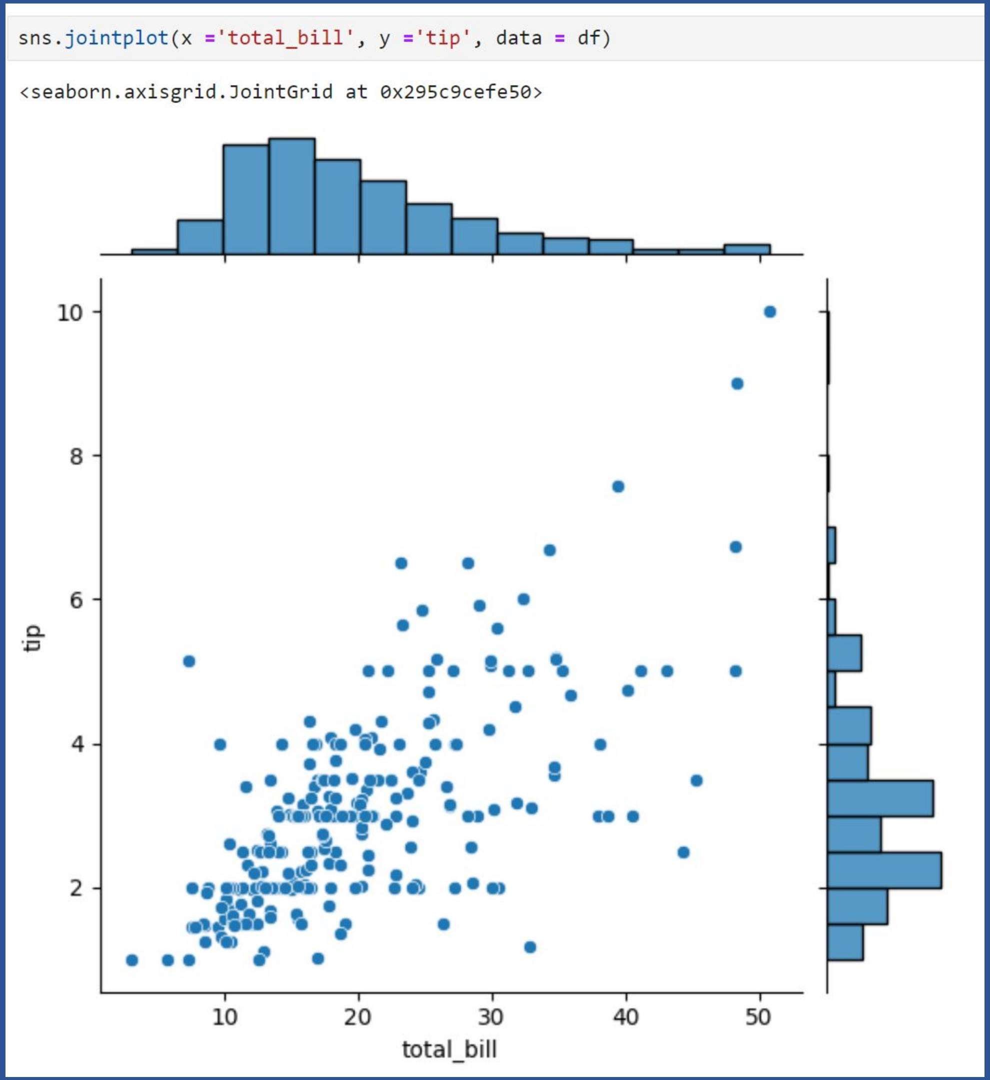
	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4



```
sns.distplot(df['total_bill'])
```



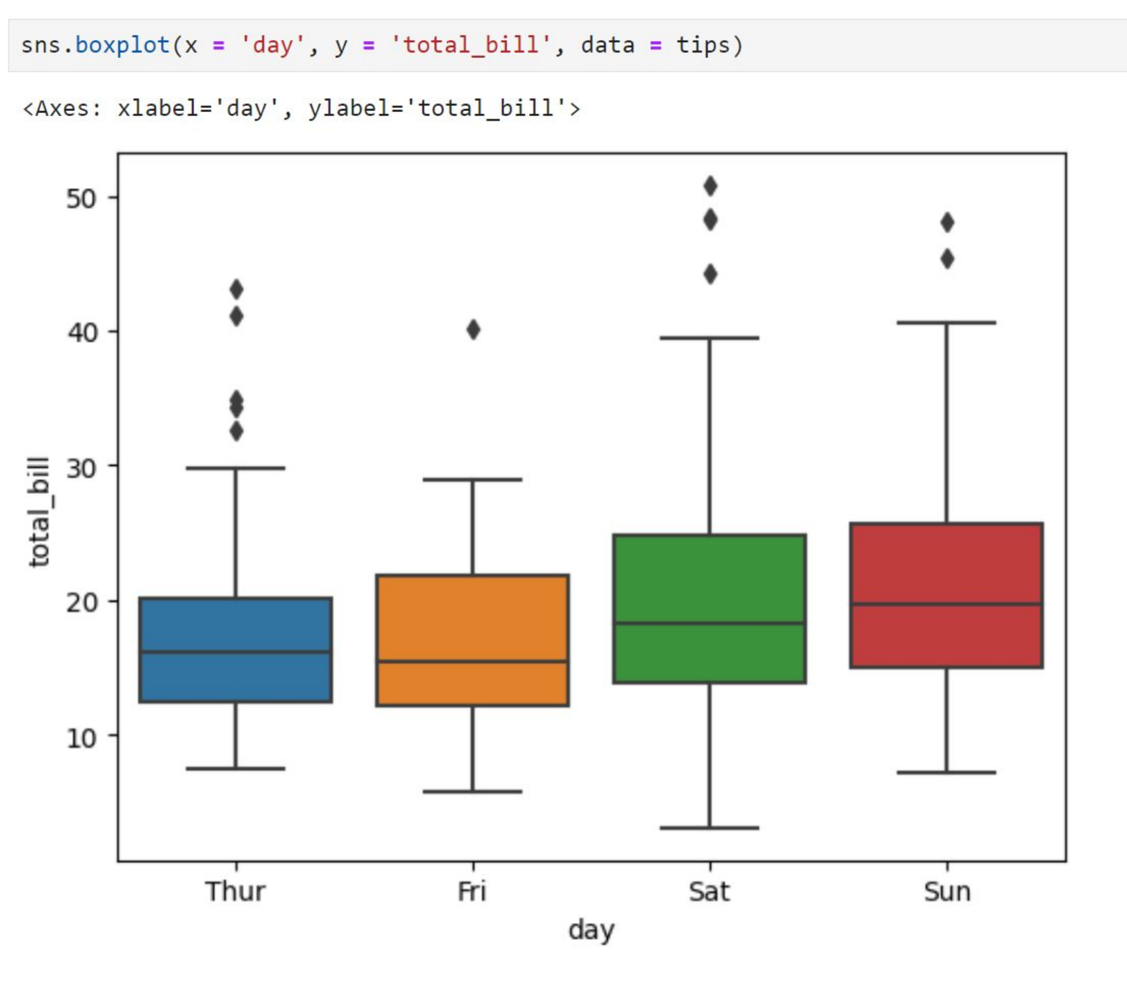
Seaborn JointPlot



```
sns.jointplot(x ='total_bill', y ='tip', data = df)
```

```
<seaborn.axisgrid.JointGrid at 0x295c9cefe50>
```

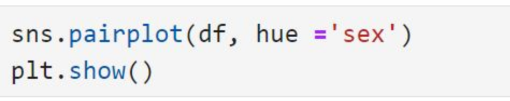
Seaborn Box Plot



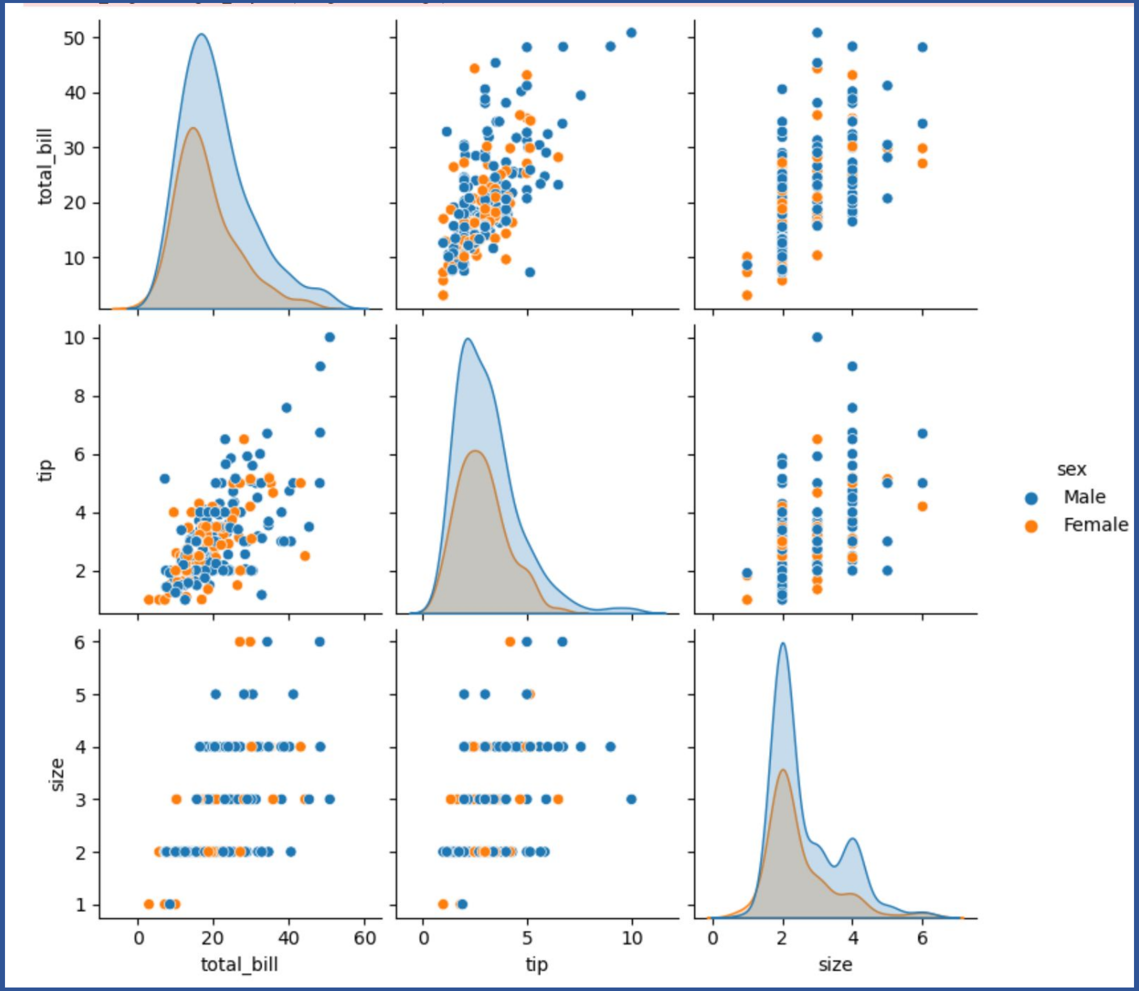
```
sns.boxplot(x = 'day', y = 'total_bill', data = tips)
```

```
<Axes: xlabel='day', ylabel='total_bill'>
```

Seaborn Pair Plot



```
sns.pairplot(df, hue ='sex')
plt.show()
```


total_bill
tip
size
sex
Male
Female