```
#mathematical operation for array
import numpy as np
#to read the data file
import pandas as pd
#for making statistical graphs
import seaborn as sns
#2D plotting. static,animated, interactive 2D plots or figures and visualazing the data we use
import matplotlib.pyplot as plt
#is used to import the os module, which provides functions for interacting with the operating system.
import os
```

```
iris =  pd.read_csv('IRIS.csv')
```

```
iris
```

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa
...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	virginica
146	6.3	2.5	5.0	1.9	virginica
147	6.5	3.0	5.2	2.0	virginica
148	6.2	3.4	5.4	2.3	virginica
149	5.9	3.0	5.1	1.8	virginica

150 rows × 5 columns

```
iris.columns
```

```
Index(['sepal_length', 'sepal_width', 'petal_length', 'petal_width',
       'species'],
      dtype='object')
```

```
iris.shape
```

```
(150, 5)
```

```
#now extract the petal length and petal width from the data for performing LR
iris = iris[['petal_length' , 'petal_width']]
iris
```

	petal_length	petal_width
0	1.4	0.2
1	1.4	0.2
2	1.3	0.2
3	1.5	0.2
4	1.4	0.2
...	...	...
145	5.2	2.3
146	5.0	1.9
147	5.2	2.0
148	5.4	2.3
149	5.1	1.8

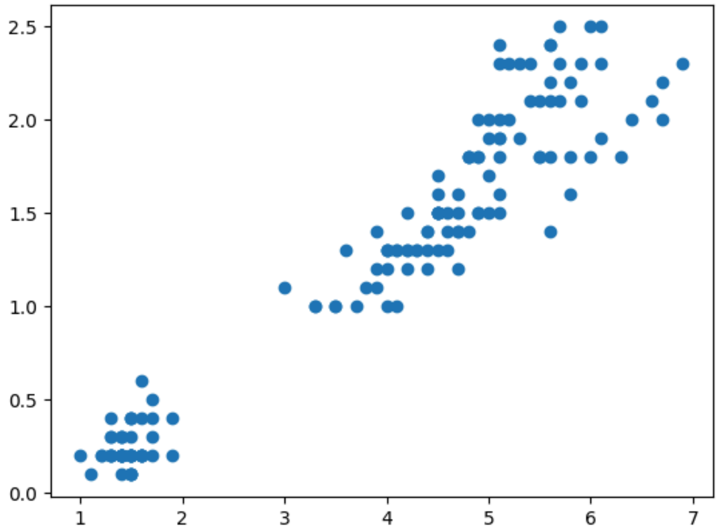
150 rows × 2 columns

```
#now define X and Y to plot the graph
X = iris['petal_length']
Y = iris['petal_width']
```

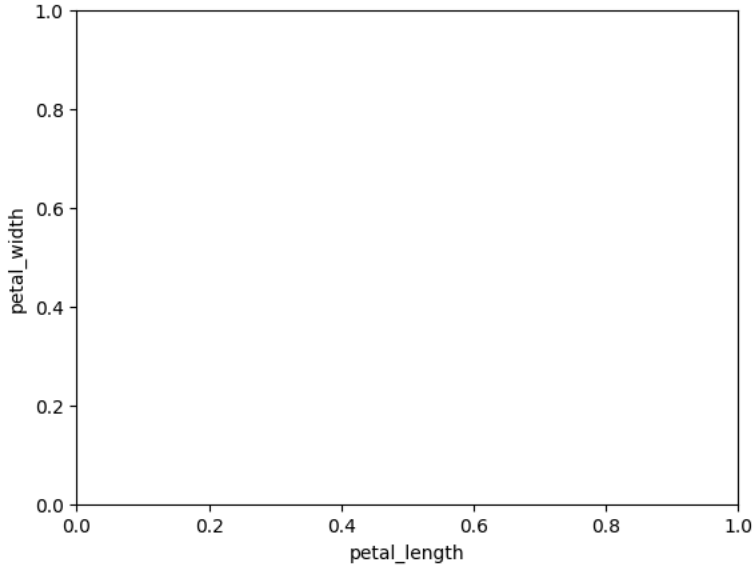
```
#import matplotlib to plot and to check whether X and Y are linerally correlated or not
import matplotlib.pyplot as plt
```

```
plt.scatter(X,Y)
```

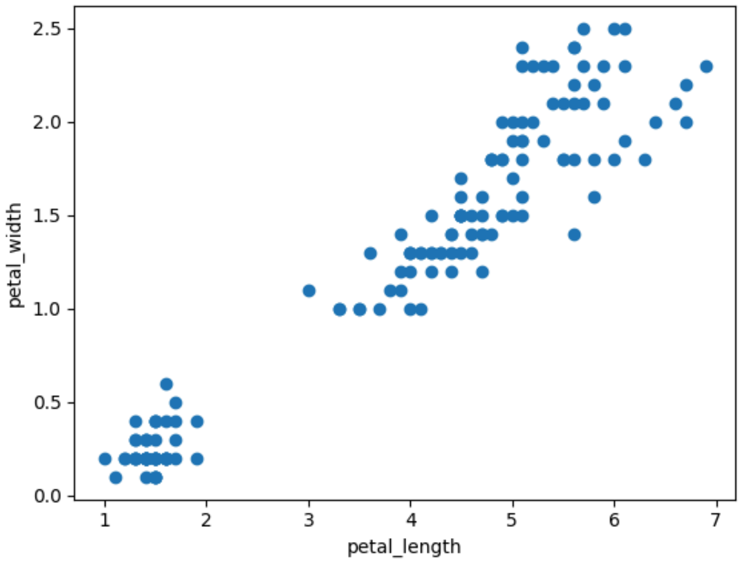
```
<matplotlib.collections.PathCollection at 0x27c3a975820>
```



```
#now name the x axis and y axis
plt.xlabel("petal_length")
plt.ylabel("petal_width")
plt.show()
```




```
#combining all the commands
plt.xlabel("petal_length")
plt.ylabel("petal_width")
plt.scatter(X,Y)
plt.show()
```




```
# so to start the LR intially we divide the data into train and test
import numpy as np
from sklearn.model_selection import train_test_split
X_train,X_test, Y_train, Y_test=train_test_split(X,Y, test_size=0.4, random_state=25)
```

```
X_train
```

```
126    4.8
108    5.8
80     3.8
19     1.5
119    5.0
      ...
118    6.9
61     4.2
143    5.9
62     4.0
132    5.6
Name: petal_length, Length: 90, dtype: float64
```

```
X_train = np.array(X_train).reshape(-1,1)
X_train
```

```
array([[4.8],
       [5.8],
       [3.8],
       [1.5],
       [5. ],
       [4.7],
       [5.1],
       [1.5],
       [3.9],
       [5. ],
       [1.5],
       [1.9],
       [5.7],
       [4.7],
       [4.4],
       [1.9],
       [6.6],
       [4.1],
       [5.1],
       [4.2],
       [5.5],
       [1.1],
       [1.6],
       [4. ],
       [1.6],
       [1.5],
       [4.8],
       [4.9],
       [1.5],
       [5.4],
       [4. ],
       [4.3],
       [4.5],
       [6.1],
       [4.4],
       [1.5],
       [6. ],
```

```
       [1.2],
       [1.4],
       [4.3],
       [5.1],
       [5.5],
       [5. ],
       [4.4],
       [1.6],
       [5.3],
       [1.6],
       [1. ],
       [4.7],
       [3.3],
       [1.7],
       [1.3],
       [1.3],
       [5.5],
       [6.3],
       [5.8],
       [5.1],
```

```
X_test = np.array(X_test).reshape(-1,1)
X_test
```

```
array([[1.4],
       [4.9],
       [4.8],
       [4.5],
       [5.6],
       [4.4],
       [4.9],
       [1.3],
       [3.7],
       [3.3],
       [1.5],
       [1.5],
       [1.6],
       [5.1],
       [1.4],
       [4.6],
       [6. ],
       [5.4],
       [3.6],
       [4.2],
       [3.5],
       [4.1],
       [4.5],
       [1.7],
       [1.5],
       [6.7],
       [3.9],
       [6.7],
       [5.7],
       [1.3],
       [4.6],
       [5. ],
       [6.1],
       [1.4],
       [5.3],
       [4.9],
       [4.5],
       [1.3],
       [1.7],
       [1.2],
       [1.3],
       [1.4],
       [1.4],
       [1.6],
       [4.8],
       [1.5],
       [1.4],
       [4.5],
       [1.5],
       [4.9],
       [5.9],
       [1.5],
       [1.4],
       [5.6],
       [3.9],
       [5.1],
       [5.2],
       [4.5],
```

```
from sklearn.linear_model import LinearRegression
```

```
Ir = LinearRegression()
Ir.fit(X_train, Y_train)
```

```
▾ LinearRegression
LinearRegression()
```

```
c = Ir.intercept_
c
```

```
-0.3818060650247206
```

```
m= Ir.coef_
m
```

```
array([0.41723083])
```

```
Y_pred_train = m*X_train+c
Y_pred_train.flatten()
```

```
array([1.62090194, 2.03813278, 1.20367111, 0.24404019, 1.70434811,
       1.57917886, 1.74607119, 0.24404019, 1.24539419, 1.70434811,
       0.24404019, 0.41093252, 1.99640969, 1.57917886, 1.45400961,
       0.41093252, 2.37191744, 1.32884036, 1.74607119, 1.37056344,
       1.91296353, 0.07714785, 0.28576327, 1.28711727, 0.28576327,
       0.24404019, 1.62090194, 1.66262503, 0.24404019, 1.87124044,
       1.28711727, 1.41228652, 1.49573269, 2.16330203, 1.45400961,
       0.24404019, 2.12157894, 0.11887094, 0.2023171 , 1.41228652,
       1.74607119, 1.91296353, 1.70434811, 1.45400961, 0.28576327,
       1.82951736, 0.28576327, 0.03542477, 1.57917886, 0.99505569,
       0.32748635, 0.16059402, 0.16059402, 1.91296353, 2.24674819,
       2.03813278, 1.74607119, 0.28576327, 0.86988644, 1.74607119,
       1.28711727, 0.32748635, 0.2023171 , 1.53745578, 1.57917886,
       1.95468661, 1.49573269, 1.78779428, 1.07850186, 0.16059402,
       1.57917886, 2.16330203, 0.2023171 , 0.24404019, 0.2023171 ,
       2.28847128, 0.2023171 , 1.49573269, 1.95468661, 1.32884036,
       0.24404019, 2.03813278, 1.28711727, 1.74607119, 1.95468661,
       2.4970867 , 1.37056344, 2.07985586, 1.28711727, 1.95468661])
```

```
Y_pred_train1 = Ir.predict(X_train)
Y_pred_train1
```


```
array([1.62090194, 2.03813278, 1.20367111, 0.24404019, 1.70434811,
       1.57917886, 1.74607119, 0.24404019, 1.24539419, 1.70434811,
       0.24404019, 0.41093252, 1.99640969, 1.57917886, 1.45400961,
       0.41093252, 2.37191744, 1.32884036, 1.74607119, 1.37056344,
       1.91296353, 0.07714785, 0.28576327, 1.28711727, 0.28576327,
       0.24404019, 1.62090194, 1.66262503, 0.24404019, 1.87124044,
       1.28711727, 1.41228652, 1.49573269, 2.16330203, 1.45400961,
       0.24404019, 2.12157894, 0.11887094, 0.2023171 , 1.41228652,
       1.74607119, 1.91296353, 1.70434811, 1.45400961, 0.28576327,
       1.82951736, 0.28576327, 0.03542477, 1.57917886, 0.99505569,
       0.32748635, 0.16059402, 0.16059402, 1.91296353, 2.24674819,
       2.03813278, 1.74607119, 0.28576327, 0.86988644, 1.74607119,
       1.28711727, 0.32748635, 0.2023171 , 1.53745578, 1.57917886,
       1.95468661, 1.49573269, 1.78779428, 1.07850186, 0.16059402,
       1.57917886, 2.16330203, 0.2023171 , 0.24404019, 0.2023171 ,
       2.28847128, 0.2023171 , 1.49573269, 1.95468661, 1.32884036,
       0.24404019, 2.03813278, 1.28711727, 1.74607119, 1.95468661,
       2.4970867 , 1.37056344, 2.07985586, 1.28711727, 1.95468661])
```

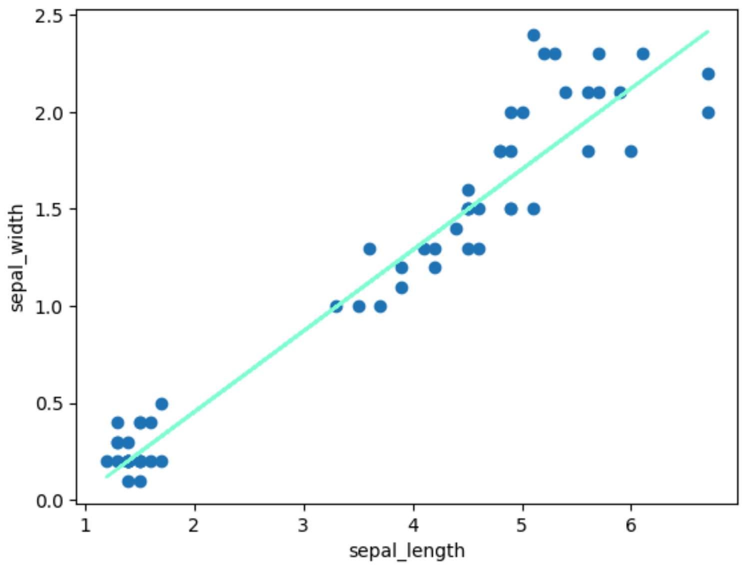

```
plt.scatter(X_train,Y_train)
plt.plot(X_train,Y_pred_train1,color= 'aquamarine')
plt.xlabel("sepal_length")
plt.ylabel("sepal_width")
plt.show()
```


```
Y_pred_test1 = Ir.predict(X_test)
Y_pred_test1
```

```
array([0.2023171 , 1.66262503, 1.62090194, 1.49573269, 1.95468661,
       1.45400961, 1.66262503, 0.16059402, 1.16194802, 0.99505569,
       0.24404019, 0.24404019, 0.28576327, 1.74607119, 0.2023171 ,
       1.53745578, 2.12157894, 1.87124044, 1.12022494, 1.37056344,
       1.07850186, 1.32884036, 1.49573269, 0.32748635, 0.24404019,
       2.41364053, 1.24539419, 2.41364053, 1.99640969, 0.16059402,
       1.53745578, 1.70434811, 2.16330203, 0.2023171 , 1.82951736,
       1.66262503, 1.49573269, 0.16059402, 0.32748635, 0.11887094,
       0.16059402, 0.2023171 , 0.2023171 , 0.28576327, 1.62090194,
       0.24404019, 0.2023171 , 1.49573269, 0.24404019, 1.66262503,
       2.07985586, 0.24404019, 0.2023171 , 1.95468661, 1.24539419,
       1.74607119, 1.78779428, 1.49573269, 1.99640969, 1.37056344])
```

sepal length

```
plt.scatter(X_test,Y_test)
plt.plot(X_test,Y_pred_test1,color= 'aquamarine')
plt.xlabel("sepal_length")
plt.ylabel("sepal_width")
plt.show()
```



Start coding or generate with AI.

Start coding or generate with AI.

Start coding or generate with AI.