Linear regression

We are using this regression method in which data is linearly dependent .
Like in the form of $y = Ax +B$ *.*
In the above given equation Y is linear dependent on x. In which B is constant.

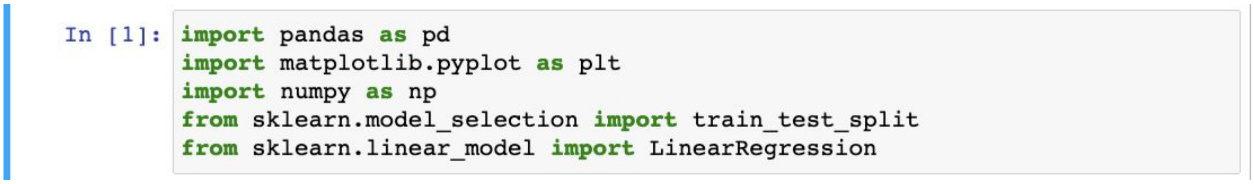
By using the above concept of linear equation we apply it for predicting the value of output by giving input .

Predict the salary on the basis of experience by using sklearn library

Step we follow to perform above regression

1. Import library
2. Download data set
3. Reshape data variable
4. Split data set for training and testing
5. Perform Linear regression
6. Predict the value
7. Plot the graph for training and testing

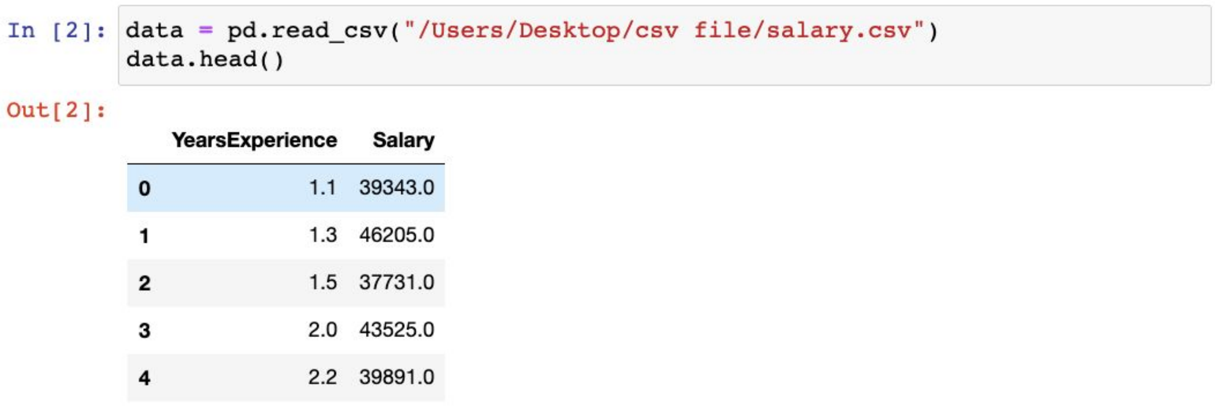
Import important library



```
In [1]: import pandas as pd
        import matplotlib.pyplot as plt
        import numpy as np
        from sklearn.model_selection import train_test_split
        from sklearn.linear_model import LinearRegression
```

Download data set
In this Given example i am using a company salary csv file in which salary is predicted by year of experience of the employee.

Data looks like in this manner

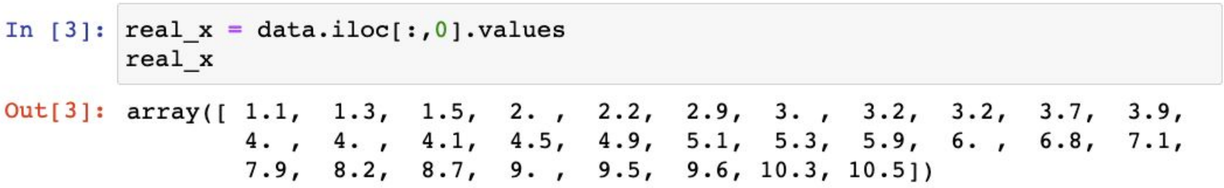


```
In [2]: data = pd.read_csv("/Users/Desktop/csv file/salary.csv")
        data.head()
```

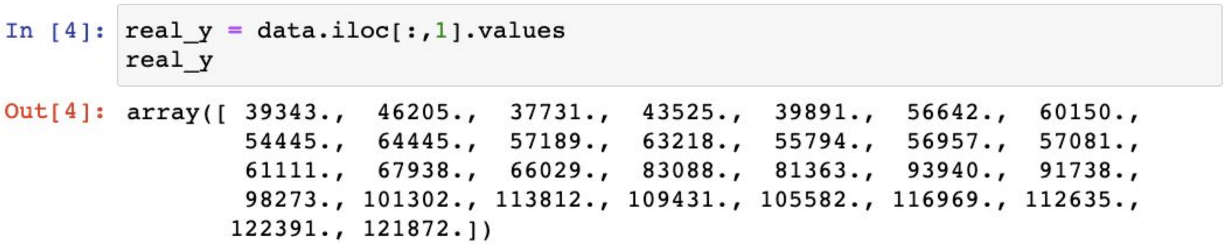
Out[2]:

	YearsExperience	Salary
0	1.1	39343.0
1	1.3	46205.0
2	1.5	37731.0
3	2.0	43525.0
4	2.2	39891.0

Now we assign data in two variable

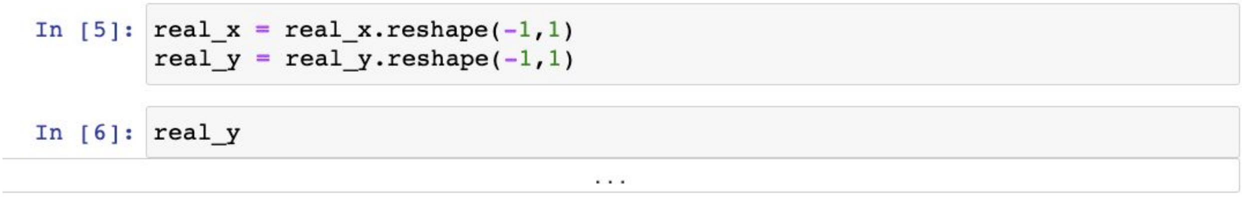


```
In [3]: real_x = data.iloc[:,0].values
        real_x

Out[3]: array([ 1.1,  1.3,  1.5,  2. ,  2.2,  2.9,  3. ,  3.2,  3.2,  3.7,  3.9,
                4. ,  4. ,  4.1,  4.5,  4.9,  5.1,  5.3,  5.9,  6. ,  6.8,  7.1,
                7.9,  8.2,  8.7,  9. ,  9.5,  9.6, 10.3, 10.5])
```



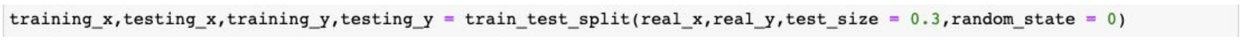
```
In [4]: real_y = data.iloc[:,1].values
        real_y

Out[4]: array([ 39343.,  46205.,  37731.,  43525.,  39891.,  56642.,  60150.,
                54445.,  64445.,  57189.,  63218.,  55794.,  56957.,  57081.,
                61111.,  67938.,  66029.,  83088.,  81363.,  93940.,  91738.,
                98273., 101302., 113812., 109431., 105582., 116969., 112635.,
               122391., 121872.])
```

Now we have to reshape the variable



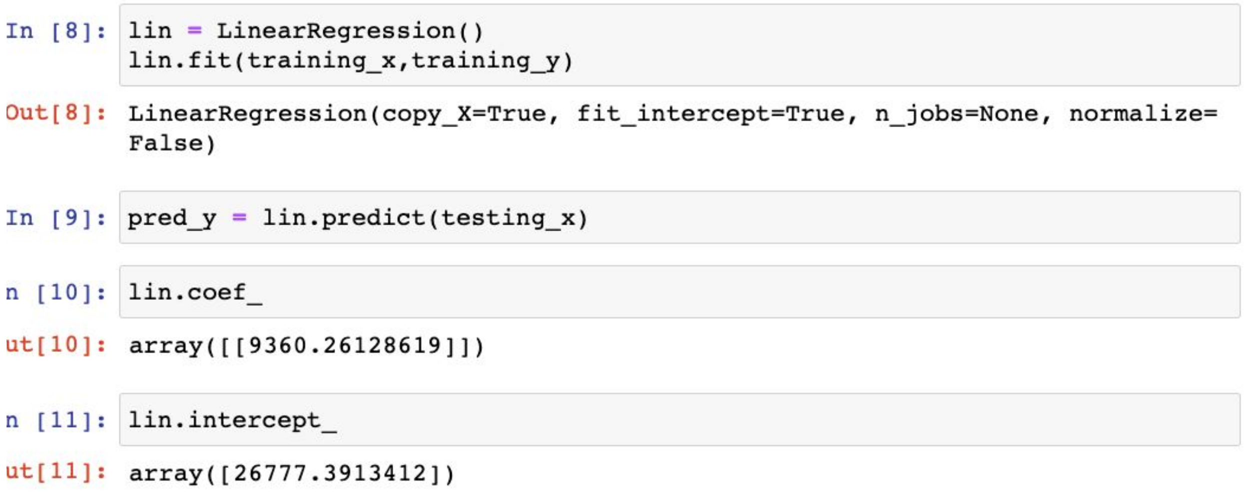
```
In [5]: real_x = real_x.reshape(-1,1)
        real_y = real_y.reshape(-1,1)

In [6]: real_y
                              ...
```

After declare the variable we have to divide the data set into training and testing
Mostly we divide 30% for tests and remaining for trains .
And put a random state (0) for matching the result from this example.



```
training_x,testing_x,training_y,testing_y = train_test_split(real_x,real_y,test_size = 0.3,random_state = 0)
```

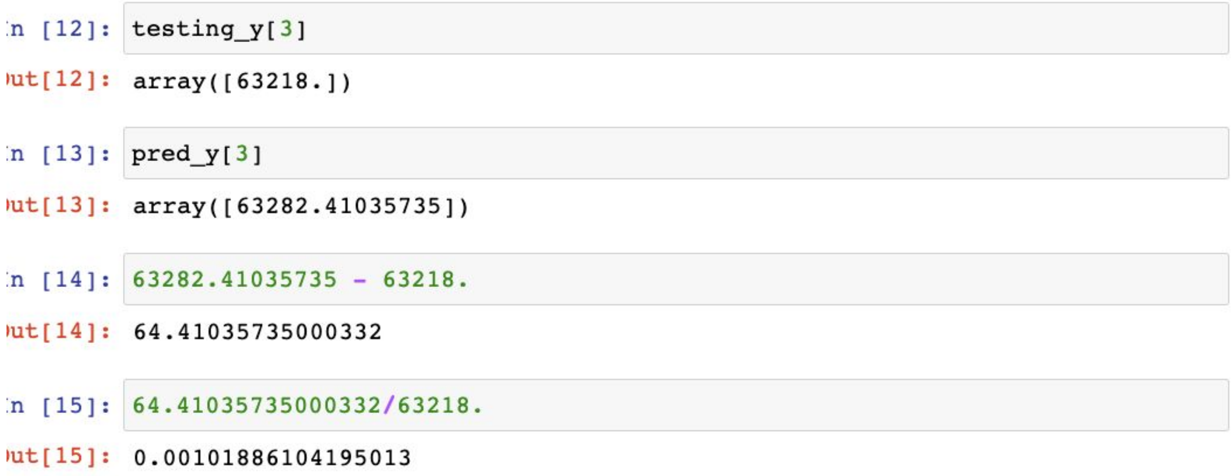
Now we perform Linear regression by using Sklearn library



```
In [8]: lin = LinearRegression()
        lin.fit(training_x,training_y)

Out[8]: LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=
        False)

In [9]: pred_y = lin.predict(testing_x)

n [10]: lin.coef_

ut[10]: array([[9360.26128619]])

n [11]: lin.intercept_

ut[11]: array([26777.3913412])
```

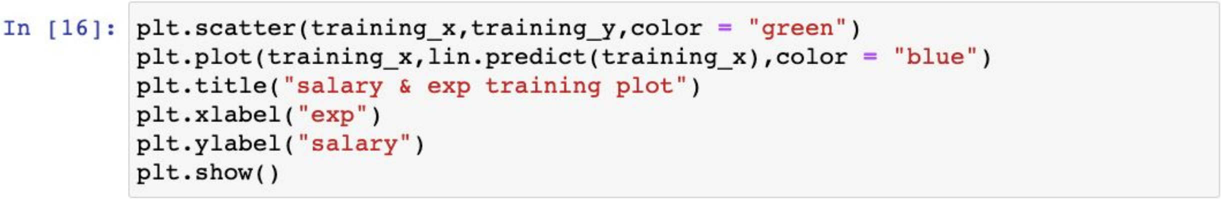
Now we get coefficient and Intercept
And by using this we can predict in this manner and also by making equation and predict it manually

Like in the form of Y = Ax+B

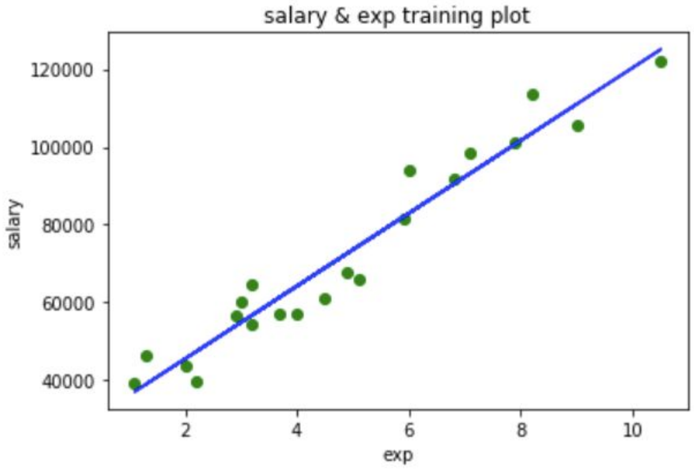


```
n [12]: testing_y[3]

ut[12]: array([63218.])

n [13]: pred_y[3]

ut[13]: array([63282.41035735])

n [14]: 63282.41035735 - 63218.

ut[14]: 64.41035735000332

n [15]: 64.41035735000332/63218.

ut[15]: 0.0010188610419501З
```

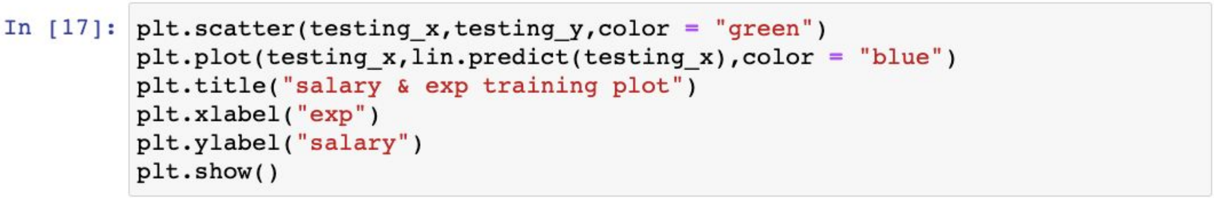
Now we plot the graph from the above regression to see how our model predict salary .



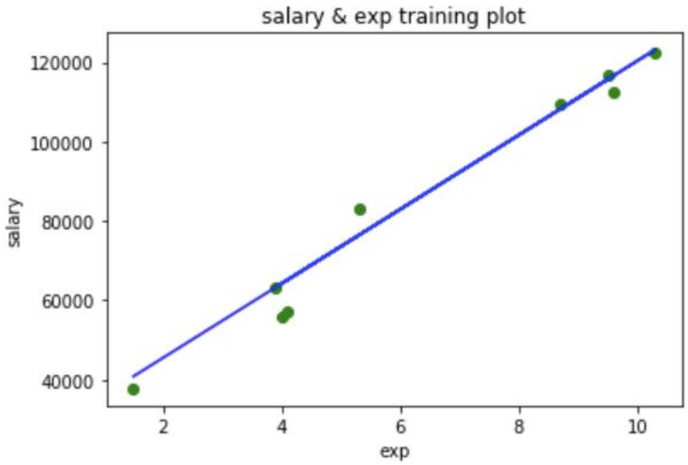
```
In [16]: plt.scatter(training_x,training_y,color = "green")
         plt.plot(training_x,lin.predict(training_x),color = "blue")
         plt.title("salary & exp training plot")
         plt.xlabel("exp")
         plt.ylabel("salary")
         plt.show()
```



Test the data and plot the graph for visualizing this given data



```
In [17]: plt.scatter(testing_x,testing_y,color = "green")
         plt.plot(testing_x,lin.predict(testing_x),color = "blue")
         plt.title("salary & exp training plot")
         plt.xlabel("exp")
         plt.ylabel("salary")
         plt.show()
```


salary & exp training plot
salary
exp