Simple Linear Regression

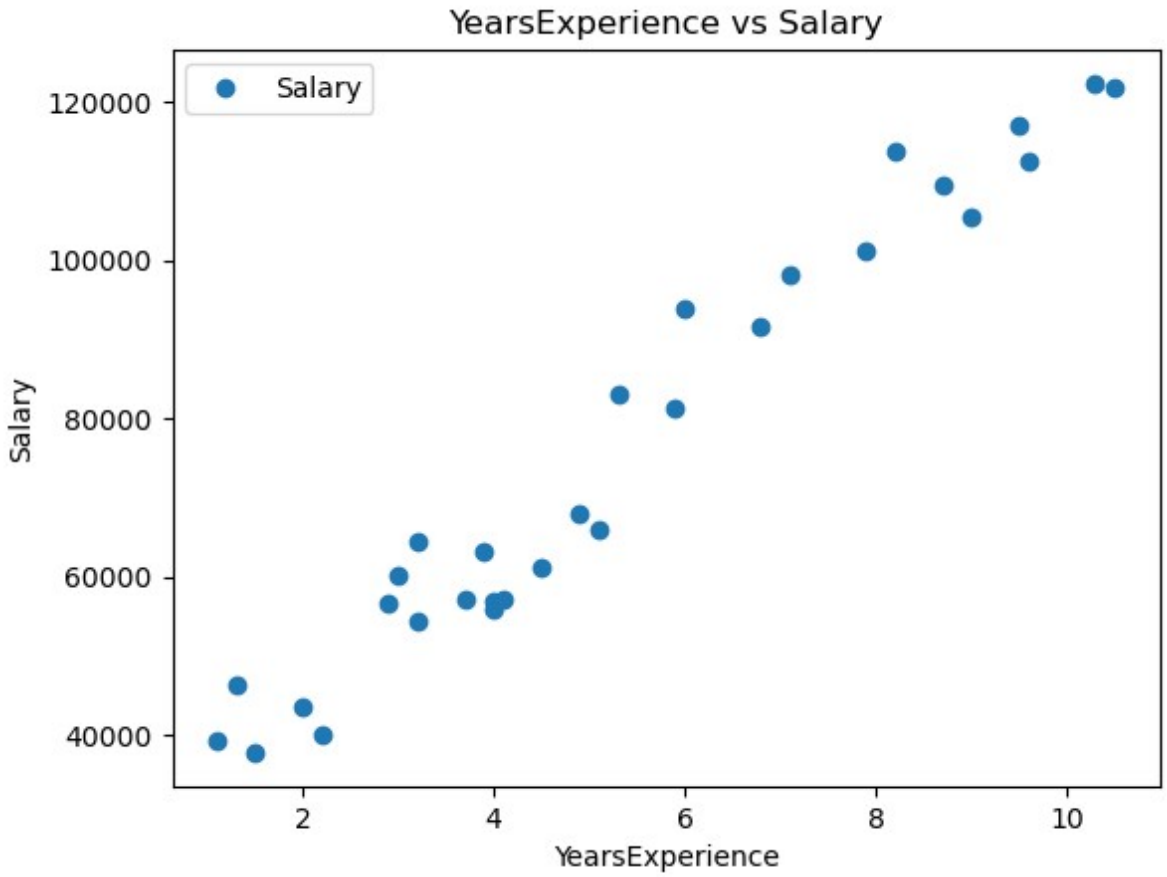
In [6]:
```
import numpy as np
import matplotlib.pyplot as plot
import pandas
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn import metrics
```

After importing the required libraries, we go ahead and import the dataset. Make sure that you have the dataset in the same folder as your notebook. Otherwise provide the complete location of your destination data file.

In [7]:
```
# Import the dataset
dataset = pandas.read_excel('SalaryAge.xlsx')
x = dataset['YearsExperience'].values.reshape(-1,1)
y = dataset['Salary'].values.reshape(-1,1)
```

let's plot our data points on a 2-D graph to look at our dataset and see if we can manually find any relationship between the data using the below script:

In [8]:
```
dataset.plot(x = 'YearsExperience', y ='Salary', style= 'o')
plot.title('YearsExperience vs Salary')
plot.xlabel('YearsExperience')
plot.ylabel('Salary')
```



Training and Testing Split

In [9]:
```
```

Initiating the Regression model and fitting it on our training data

In [10]:
```
linearRegressor = LinearRegression()
```

Out[10]:
```
▾ LinearRegression
LinearRegression()
```

After the regression model has been learnt, we go ahead and make predictions

In [11]:
```
```

Now compare the actual output values for X_test with the predicted values, execute the following script:

In [12]:
```
df = pandas.DataFrame({'Actual': yTest.flatten(), 'Predicted': yPrediction.fla
```

Out[12]:

	Actual	Predicted
0	37731	40748.961841
1	122391	122699.622956
2	57081	64961.657170
3	63218	63099.142145
4	116969	115249.562855
5	109431	107799.502753

Observing the Regression line on the training dataset

In [13]:
```
plot.scatter(xTrain, yTrain, color = 'red')
plot.plot(xTrain, linearRegressor.predict(xTrain), color = 'blue')
plot.title('Salary vs Experience (Training set)')
plot.xlabel('Years of Experience')
plot.ylabel('Salary')
```

Observing the same regression line against the testing instances.

In [14]:
```
plot.scatter(xTest, yTest, color = 'red')
plot.plot(xTest, linearRegressor.predict(xTest), color = 'blue')
plot.title('Salary vs Experience (Test set)')
plot.xlabel('Years of Experience')
plot.ylabel('Salary')
```

Error Evaluations

The final step is to evaluate the performance of the algorithm. This step is particularly important to compare how well different algorithms perform on a particular dataset. For regression algorithms, three evaluation metrics are commonly used:

In [15]:
```
print('Mean Absolute Error:', metrics.mean_absolute_error(yTest, yPrediction))
print('Mean Squared Error:', metrics.mean_squared_error(yTest, yPrediction))
```

```
Mean Absolute Error: 2446.1723690465055
Mean Squared Error: 12823412.298126549
Root Mean Squared Error: 3580.979237321343
```