In [1]:

```
# Working upon Salary Data set
```

In [5]:

```
import numpy as np
import pandas as pd
```

In [9]:

```
# importing the salary data set
salary_df = pd.read_csv(r'F:\Prthon Programming\Udemy\Part 2 - Regression\Section 4 - Simple Linear Regression\Salary_Data.csv')
```

In [10]:

```
salary_df.head() # by default first 5 obv it shows
```

Out[10]:

	YearsExperience	Salary
0	1	39343
1	1	46205
2	2	37731
3	2	43525
4	2	39891

In [11]:

```
salary_df.tail() # by default last 5 obv it shows
```

Out[11]:

	YearsExperience	Salary
25	9	105582
26	10	116969
27	10	112635
28	10	122391
29	11	121872

In [12]:

```
salary_df.shape # try to see number of obv + number of columns
```

Out[12]:

```
(30, 2)
```

In [13]:

```
salary_df.isnull().sum() # checking if there are any missing values in data set
```

Out[13]:

```
YearsExperience    0
Salary             0
dtype: int64
```

In [19]:

```
X = salary_df.values[:,:-1] # -1 means that i m not taking the salary column in (X variable) is IV
Y = salary_df.values[:,-1] # (Y variable) salary is my dependent variable
```

In [20]:

```
print(X) # now my indepentdent variable is ready
```

```
[[ 1]
 [ 1]
 [ 2]
 [ 2]
 [ 2]
 [ 3]
 [ 3]
 [ 3]
 [ 3]
 [ 4]
 [ 4]
 [ 4]
 [ 4]
 [ 4]
 [ 5]
 [ 5]
 [ 5]
 [ 5]
 [ 6]
 [ 6]
 [ 7]
 [ 7]
 [ 8]
 [ 8]
 [ 9]
 [ 9]
 [10]
 [10]
 [10]
 [11]]
```

In [29]:

```
print(Y) # now my dependent variable is ready
```

```
[ 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]
```

In [41]:

```
#from sklearn.cross_validation import train_test_split
from sklearn.model_selection import train_test_split

X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 1/3, ran
dom_state = 0)
# I have split the data set in 80:20 ratio with the help of model_selection
```

In [42]:

```
print(X_train)
```

```
[[ 3]
 [ 5]
 [ 3]
 [ 5]
 [ 8]
 [ 7]
 [ 1]
 [11]
 [ 3]
 [ 2]
 [ 6]
 [ 6]
 [ 4]
 [ 3]
 [ 9]
 [ 2]
 [ 1]
 [ 7]
 [ 5]
 [ 4]]
```

In [43]:

```
print(Y_test)
```

```
[ 37731 122391  57081  63218 116969 109431 112635  55794  83088 101302]
```

In [ ]:

```
#=============================================END OF THE DATA PREPROCESSING=================================================================
```

In [44]:

```
from sklearn.linear_model import LinearRegression # taken class as LinearRegression

regressor = LinearRegression()
regressor.fit(X_train, Y_train) # fitting X_train and Y_train by using fit function
```

Out[44]:

```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None,
         normalize=False)
```

In [ ]:

```
#=============================================WE HAVE TRAIN THE MACHINE=============================================================
```

In [45]:

```
Y_pred = regressor.predict(X_test) # predicting test set results
```

In [48]:

```
print(Y_pred)
```

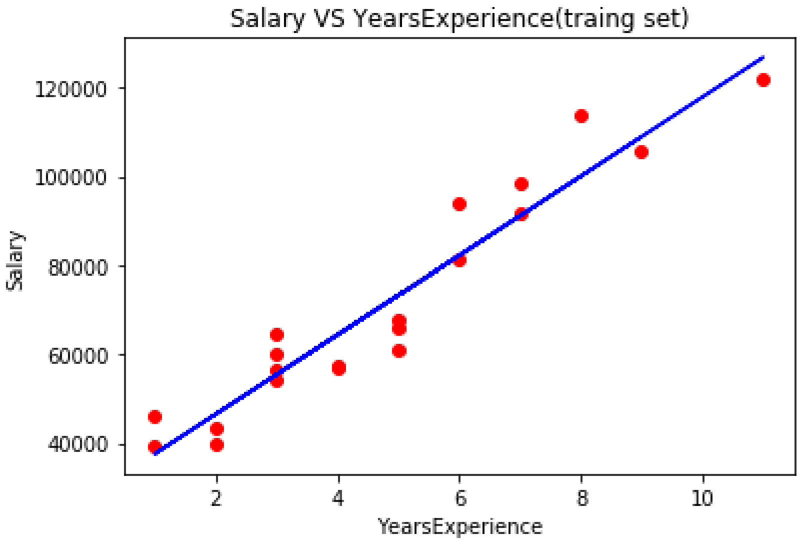
```
[ 46517.38475499 117804.99274047  64339.28675136  64339.28675136
 117804.99274047 108894.04174229 117804.99274047  64339.28675136
  73250.23774955  99983.0907441 ]
```

In [47]:

```
import matplotlib.pyplot as plt # data visualisation
```

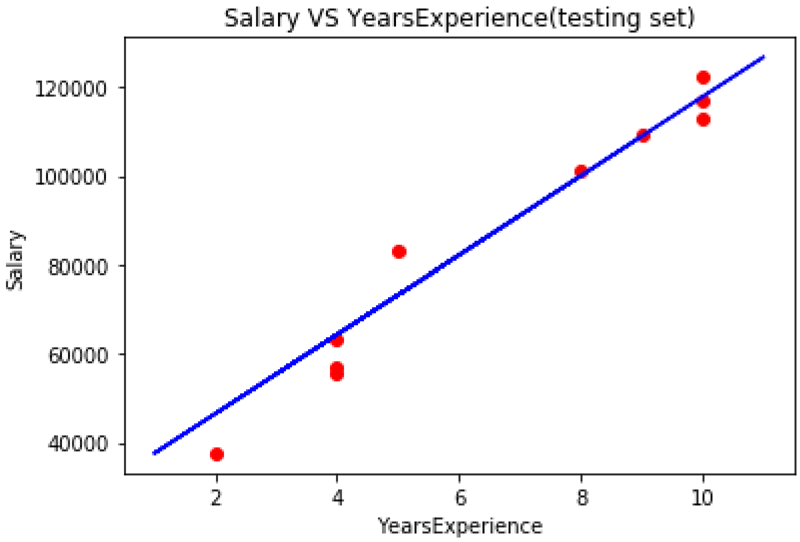
```
In [50]: # visualisation training set results
         plt.scatter(X_train, Y_train, Color = 'red') # data points
         plt.plot(X_train, regressor.predict(X_train), color = 'blue') # slope line
         plt.title('Salary VS YearsExperience(traing set)')
         plt.xlabel('YearsExperience')
         plt.ylabel('Salary')
         plt.show
         # real salary with predicted salary
```

```
Out[50]: <function matplotlib.pyplot.show(*args, **kw)>
```



```
In [51]: # visualisation testing set results
         plt.scatter(X_test, Y_test, Color = 'red')
         plt.plot(X_train, regressor.predict(X_train), color = 'blue')
         plt.title('Salary VS YearsExperience(testing set)')
         plt.xlabel('YearsExperience')
         plt.ylabel('Salary')
         plt.show
```

```
Out[51]: <function matplotlib.pyplot.show(*args, **kw)>
```



In [ ]:

```
#============================================END WITH A GOOD RESULT===========
==================================================
```