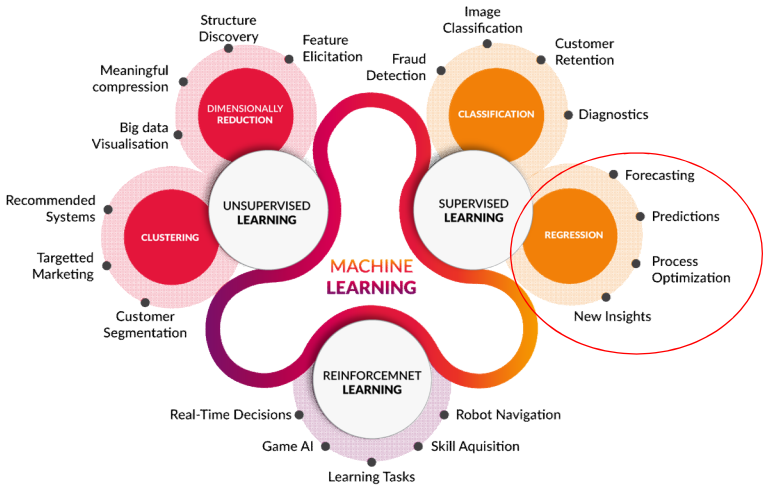
# Simple Linear Regression

Machine Learning

Dr. Adnan Abid

Courtesy Super Data Science




https://www.linkedin.com/pulse/machine-learning-types-problems-tasks-hamdi-tarek/

# Regressions

**Simple Linear Regression**

Coefficient

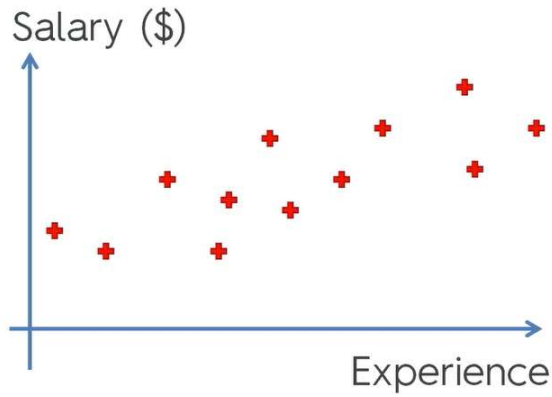
$$y = b_0 + b_1 * x_1$$

Dependent variable (DV) — Independent variable (IV)



# Regressions

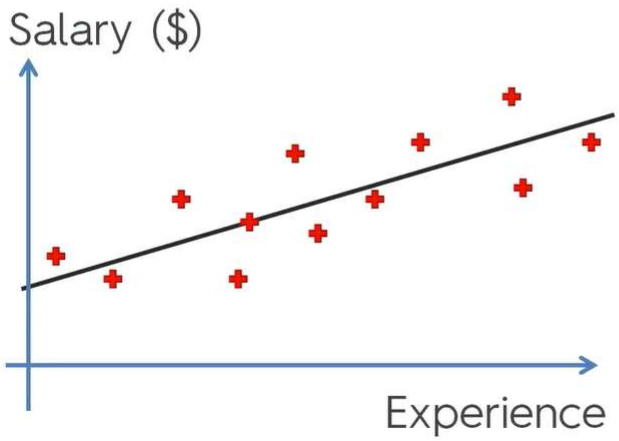
Simple Linear Regression:




| YearsExperience | Salary |
|---|---|
| 1.1 | 39343 |
| 1.3 | 46205 |
| 1.5 | 37731 |
| 2 | 43525 |
| 2.2 | 39891 |
| 2.9 | 56642 |
| 3 | 60150 |
| 3.2 | 54445 |
| 3.2 | 64445 |
| 3.7 | 57189 |

# Regressions

Simple Linear Regression:




$y = b_0 + b_1 * x$

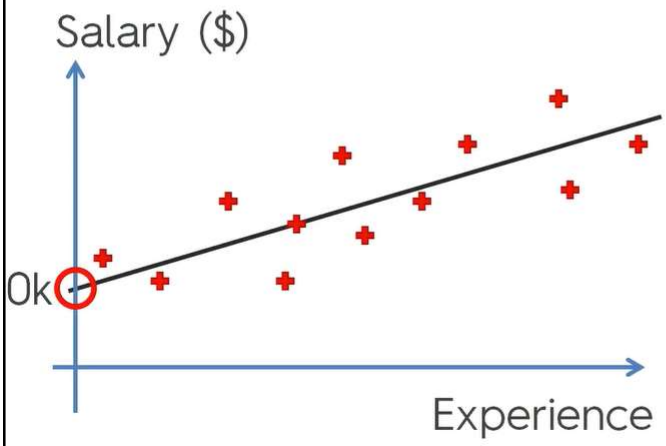
$\downarrow$

Salary = $b_0$ + $b_1$ *Experience



# Regressions

Simple Linear Regression:



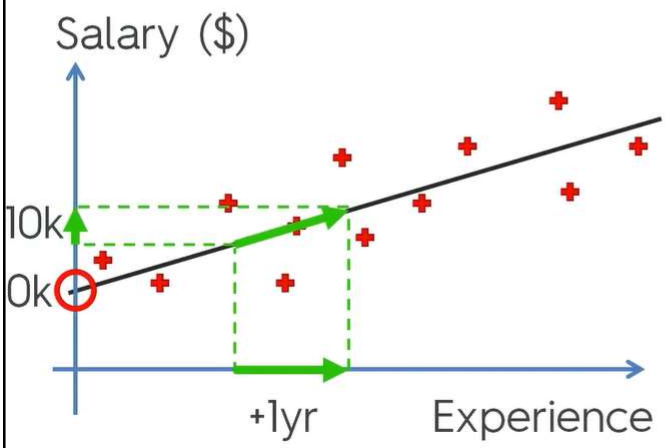

$y = b_0 + b_1 * x$

$\downarrow$

Salary = $b_0$ + $b_1$ *Experience

# Regressions

Simple Linear Regression:




$y = b_0 + b_1 * x$

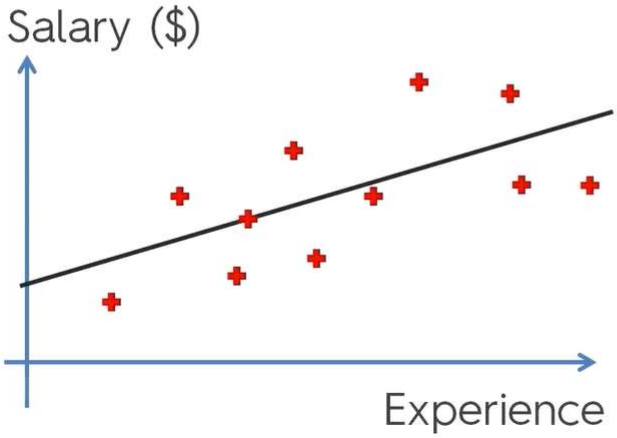
Salary = $b_0$ + $b_1$ *Experience



## How Good is the Regression Line?

# Ordinary Least Squares

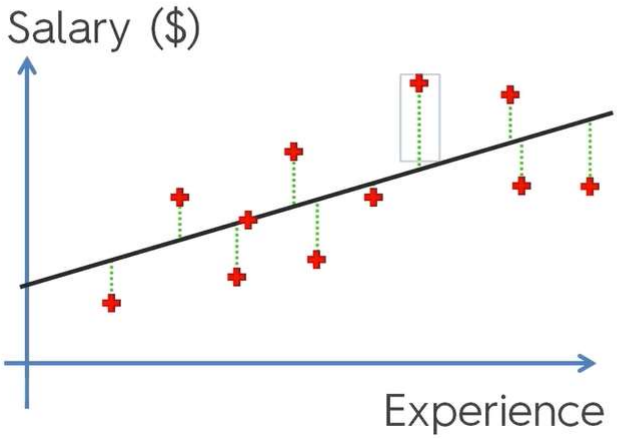
Simple Linear Regression:



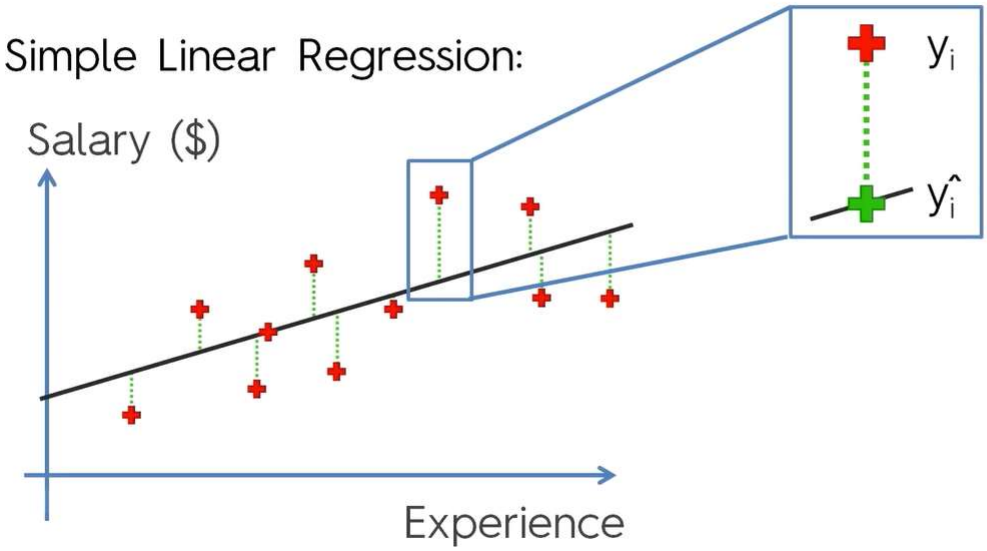



# Ordinary Least Squares

Simple Linear Regression:

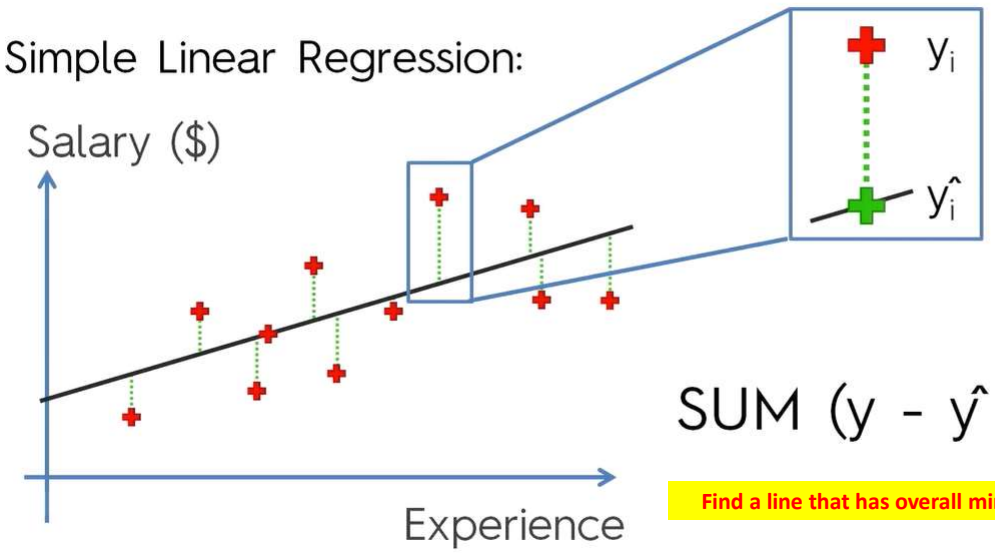


# Ordinary Least Squares






# Ordinary Least Squares




$$SUM\ (y - \hat{y})^2 \rightarrow min$$

**Find a line that has overall min least squared error**

# Computation of **a** and **b** for Linear Regression

**y = a x + b**

- **a** – slope

**b** – y-intercept

$$a = \frac{n\sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} x_i \sum_{i=1}^{n} y_i}{n\sum_{i=1}^{n} x_i^2 - (\sum_{i=1}^{n} x_i)^2}$$

$$b = \frac{1}{n}(\sum_{i=1}^{n} y_i - a\sum_{i=1}^{n} x_i)$$

**Problem 1**
Consider the following set of points:

**{(-2 , -1) , (1 , 1) , (3 , 2)}**

**n = 3**

| x | y | x y | $x^2$ |
|---|---|---|---|
| -2 | -1 | 2 | 4 |
| 1 | 1 | 1 | 1 |
| 3 | 2 | 6 | 9 |
| Σx = 2 | Σy = 2 | Σxy = 9 | $\Sigma x^2 = 14$ |

$$a = \frac{n\sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} x_i \sum_{i=1}^{n} y_i}{n\sum_{i=1}^{n} x_i^2 - (\sum_{i=1}^{n} x_i)^2}$$

$$b = \frac{1}{n}(\sum_{i=1}^{n} y_i - a\sum_{i=1}^{n} x_i)$$

| x | y | x y | $x^2$ |
|---|---|---|---|
| -2 | -1 | 2 | 4 |
| 1 | 1 | 1 | 1 |
| 3 | 2 | 6 | 9 |
| Σx = 2 | Σy = 2 | Σxy = 9 | $\Sigma x^2 = 14$ |

**a** = $\frac{(n\,\Sigma x\,y - \Sigma x \Sigma y)}{(n\Sigma x^2 - (\Sigma x)^2)}$

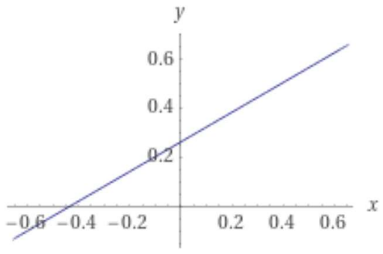
**a** = $\frac{(3*9 - 2*2)}{(3*14 - 2^2)}$

**a** = 23/38

**b** = $\frac{(1)}{n}(\Sigma y - a\,\Sigma x)$

**b** = (1/3)(2 - (23/38)*2)

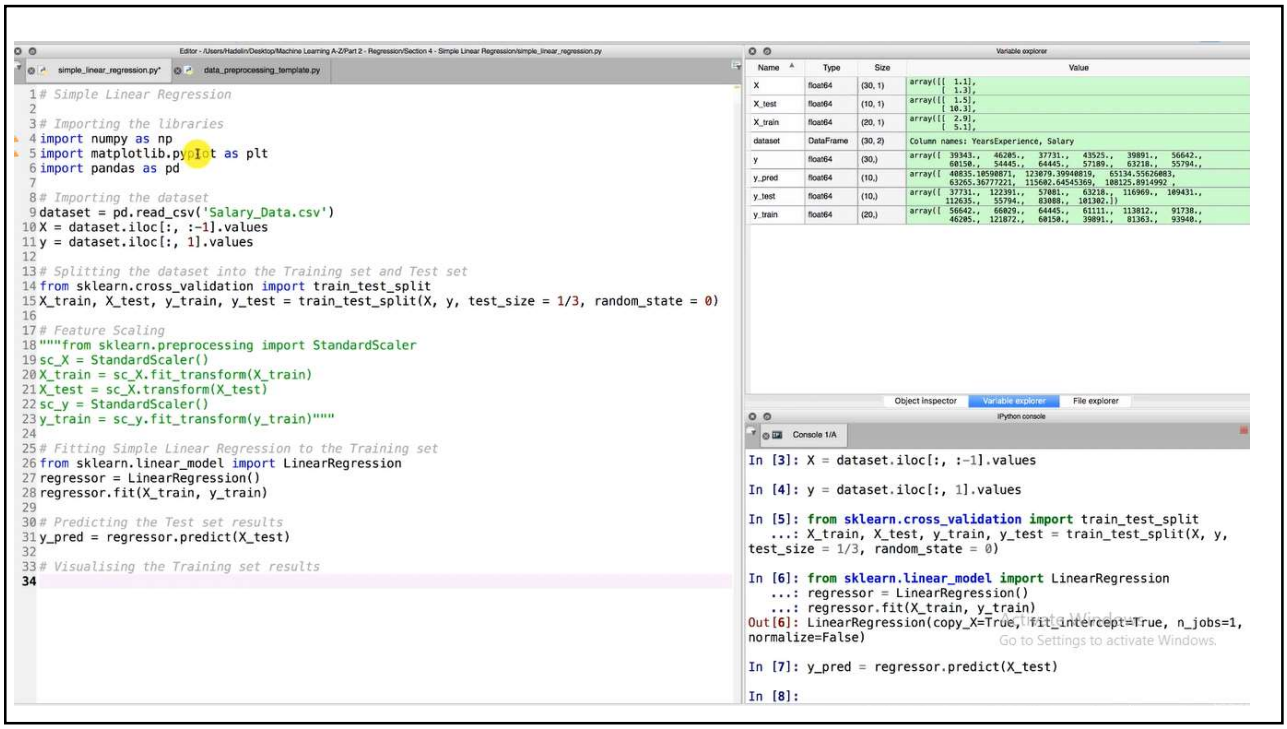
**b** = 5/19




**y = a x + b**

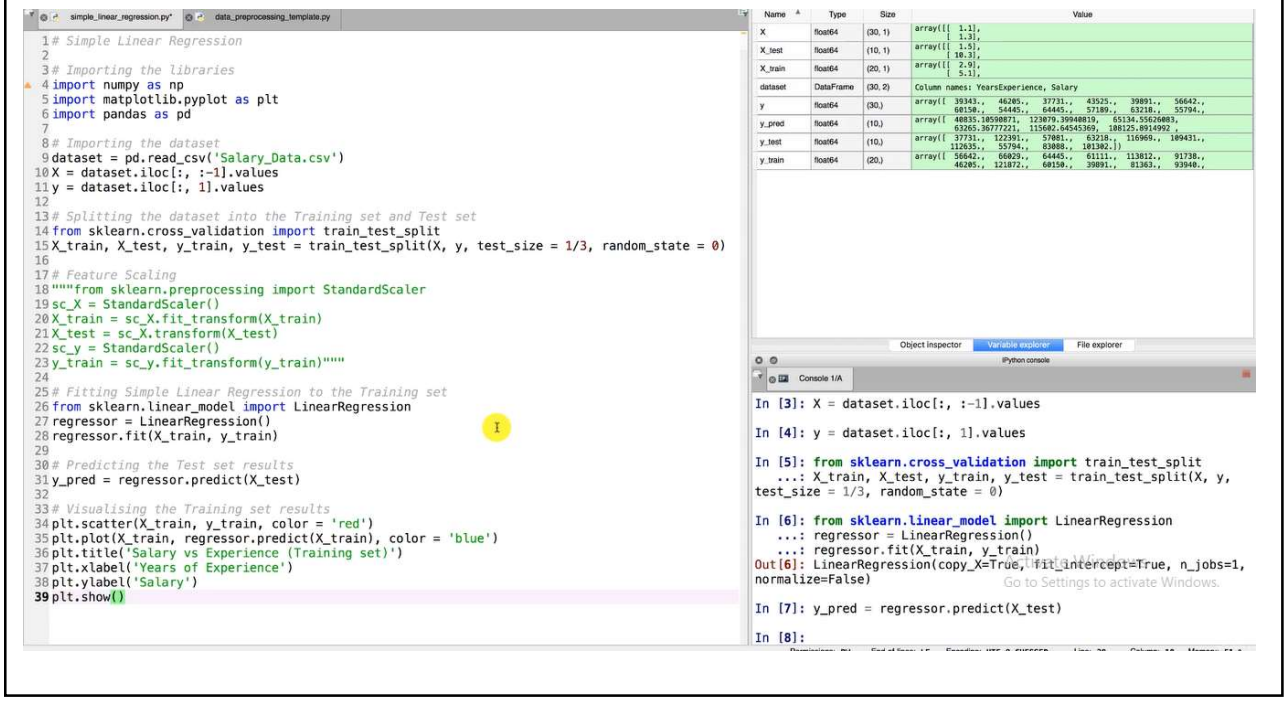
$$y = \frac{23\,x}{38} + \frac{5}{19}$$

https://www.khanacademy.org/math/statistics-probability/describing-relationships-quantitative-data/more-on-regression/v/regression-line-example



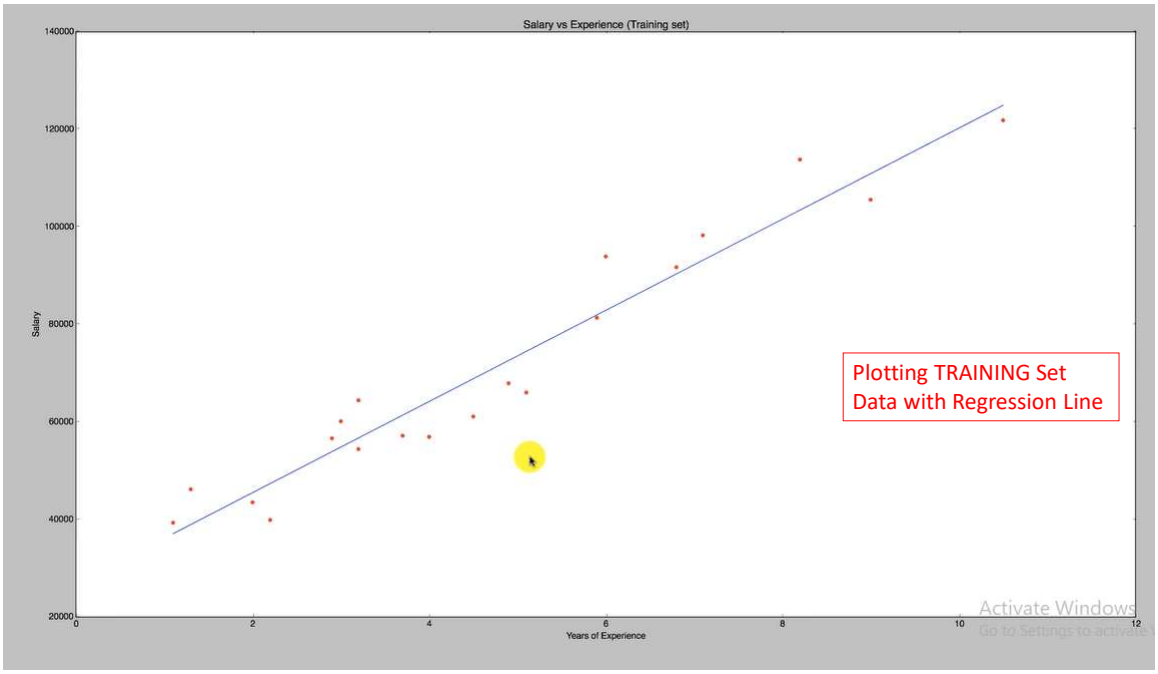
```
# Simple Linear Regression

# Importing the libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

# Importing the dataset
dataset = pd.read_csv('Salary_Data.csv')
X = dataset.iloc[:, :-1].values
y = dataset.iloc[:, 1].values

# Splitting the dataset into the Training set and Test set
from sklearn.cross_validation import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 1/3, random_state = 0)

# Feature Scaling
"""from sklearn.preprocessing import StandardScaler
sc_X = StandardScaler()
X_train = sc_X.fit_transform(X_train)
X_test = sc_X.transform(X_test)
sc_y = StandardScaler()
y_train = sc_y.fit_transform(y_train)"""

# Fitting Simple Linear Regression to the Training set
from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(X_train, y_train)

# Predicting the Test set results
y_pred = regressor.predict(X_test)

# Visualising the Training set results
```

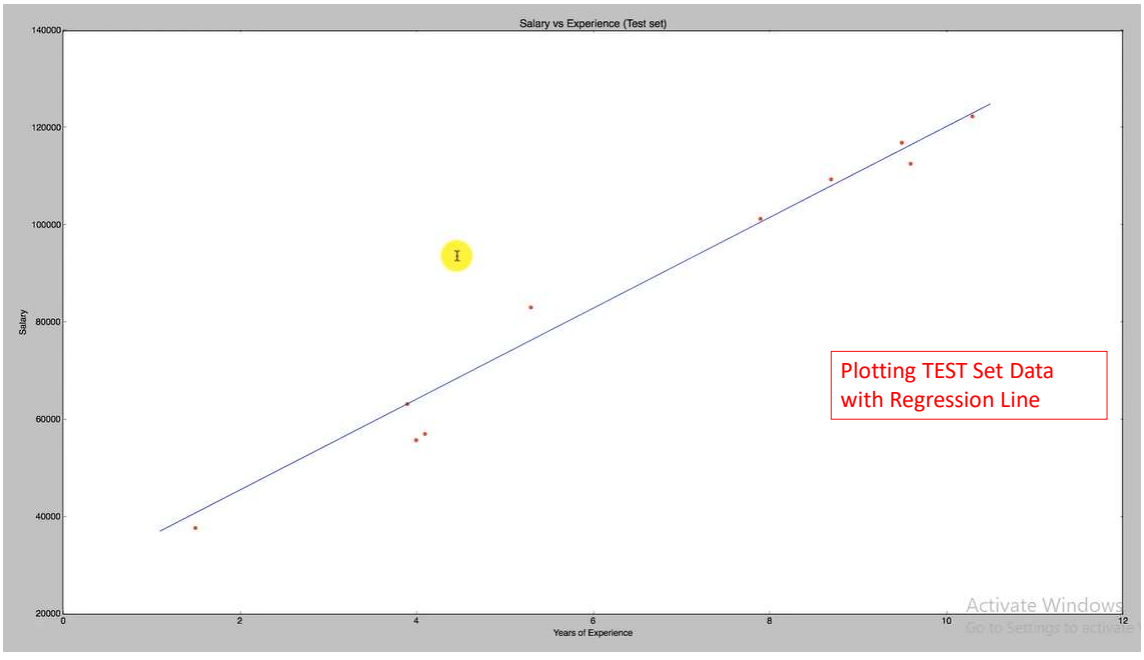
```
In [3]: X = dataset.iloc[:, :-1].values

In [4]: y = dataset.iloc[:, 1].values

In [5]: from sklearn.cross_validation import train_test_split
   ...: X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size = 1/3, random_state = 0)

In [6]: from sklearn.linear_model import LinearRegression
   ...: regressor = LinearRegression()
   ...: regressor.fit(X_train, y_train)
Out[6]: LinearRegression(copy_X=True, fit_intercept=True, n_jobs=1,
normalize=False)

In [7]: y_pred = regressor.predict(X_test)

In [8]:
```



```
# Simple Linear Regression

# Importing the libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

# Importing the dataset
dataset = pd.read_csv('Salary_Data.csv')
X = dataset.iloc[:, :-1].values
y = dataset.iloc[:, 1].values

# Splitting the dataset into the Training set and Test set
from sklearn.cross_validation import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 1/3, random_state = 0)

# Feature Scaling
"""from sklearn.preprocessing import StandardScaler
sc_X = StandardScaler()
X_train = sc_X.fit_transform(X_train)
X_test = sc_X.transform(X_test)
sc_y = StandardScaler()
y_train = sc_y.fit_transform(y_train)"""

# Fitting Simple Linear Regression to the Training set
from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(X_train, y_train)

# Predicting the Test set results
y_pred = regressor.predict(X_test)

# Visualising the Training set results
plt.scatter(X_train, y_train, color = 'red')
plt.plot(X_train, regressor.predict(X_train), color = 'blue')
plt.title('Salary vs Experience (Training set)')
plt.xlabel('Years of Experience')
plt.ylabel('Salary')
plt.show()
```

```
In [3]: X = dataset.iloc[:, :-1].values

In [4]: y = dataset.iloc[:, 1].values

In [5]: from sklearn.cross_validation import train_test_split
   ...: X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size = 1/3, random_state = 0)

In [6]: from sklearn.linear_model import LinearRegression
   ...: regressor = LinearRegression()
   ...: regressor.fit(X_train, y_train)
Out[6]: LinearRegression(copy_X=True, fit_intercept=True, n_jobs=1,
normalize=False)

In [7]: y_pred = regressor.predict(X_test)

In [8]:
```


Salary vs Experience (Training set)
Salary
Years of Experience
140000
120000
100000
80000
60000
40000
20000
0
2
4
6
8
10
12
Plotting TRAINING Set Data with Regression Line
Activate Windows
Go to Settings to activate


Salary vs Experience (Test set)
Salary
Years of Experience
140000
120000
100000
80000
60000
40000
20000
0
2
4
6
8
10
12
Plotting TEST Set Data with Regression Line
Activate Windows
Go to Settings to activate