Linear Regression

"Linear Regression" is a method to predict dependent variable (Y) based on values of independent variables (X). It can be used for the cases where we want to predict some continuous quantity

To start with Linear Regression, you must be aware of a few basic concepts of statistics. i.e.,

Correlation (r) – Explains the relationship between two variables, possible values -1 to +1

Variance (σ2)– Measure of spread in your data

Standard Deviation (σ) – Measure of spread in your data (Square root of Variance)

Normal distribution

Residual (error term) – {Actual value – Predicted value}

Linear Regression Line

While doing linear regression our objective is to fit a line through the distribution which is nearest to most of the points.

if we have one dependent variable 'Y' and one independent variable 'X' – relationship between 'X' & 'Y' can be represented in a form of following equation:

Y = B0 + B1X

Where,

Y = Dependent Variable

X = Independent Variable

B0 = Constant term (y axis intercept)

B1 = Coefficient of relationship between 'X' & 'Y' (i.e. slope)

Model Performance

R – Square (R2)

- R2 ranges from 0 to 1.
- R2 of 0 means that the dependent variable cannot be predicted from the independent variable
- R2 of 1 means the dependent variable can be predicted without error from the independent variable
- An R2 between 0 and 1 indicates the extent to which the dependent variable is predictable.

An R2 of 0.20 means that 20 percent of the variance in Y is predictable from X

Root Mean Squared Error (RMSE) :

RMSE tells the measure of dispersion of predicted values from actual values.

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{j=1}^{n}(y_j - \hat{y}_j)^2}$$

Mean Absolute Error (MAE)

The Mean Absolute Error measures the average of the absolute difference between each ground truth and the predictions.

$$\mathrm{MAE} = \frac{1}{n} \sum_{j=1}^{n} |y_j - \hat{y}_j|$$

Where:

n = the number of errors,

Σ = summation symbol (which means “add them all up”),

|xi – x| = the absolute errors.

Mean Squared Error (MSE)

The mean squared error tells you how close a regression line is to a set of points.

It does this by taking the distances from the points to the regression line (these distances are the “errors”) and squaring them.

The smaller the means squared error, the closer you are to finding the line of best fit.

Depending on your data, it may be impossible to get a very small value for the mean squared error.

Root Mean Square Error (RMSE)

RMSE is a measure of how spread out these residuals are.

In other words, it tells you how concentrated the data is around the line of best fit.

Root mean square error is commonly used in forecasting, and regression analysis to verify experimental results.

from sklearn import metrics print('Mean Absolute Error:', metrics.mean_absolute_error(y_test, y_pred))
print('Mean Squared Error:', metrics.mean_squared_error(y_test, y_pred))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_test, y_pred)))

In [ ]:

```
```

In [ ]:

```
```

In [15]:

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

In [24]:
```
data = pd.DataFrame({'x':[1,2,3,4,5,6,7,8,9,10],'y':[2,1,3,6,9,11,13,15,17,20]})
data
```

Out[24]:

	x	y
0	1	2
1	2	1
2	3	3
3	4	6
4	5	9
5	6	11
6	7	13
7	8	15
8	9	17
9	10	20

In [25]:
```
std_x = data.x.std()
std_x
```

Out[25]:
```
3.0276503540974917
```

In [26]:
```
std_y = data.y.std()
std_y
```

Out[26]:
```
6.61731734835869
```

In [28]:
```
mean_x = data.x.mean()
mean_x
```

Out[28]:
```
5.5
```

In [29]:
```
mean_y = data.y.mean()
mean_y
```

Out[29]:
```
9.7
```

In [30]:
```
corr = np.corrcoef(data.x, data.y)
corr
```

Out[30]:
```
array([[1.        , 0.98993808],
       [0.98993808, 1.        ]])
```

In [ ]:
```
# B1 = Correlation * (Std. Dev. of y/ Std. Dev. of x)
```

In [31]:
```
b1 = 0.989938 * (std_y/std_x)
b1
```

Out[31]:
```
2.1636361980616496
```

In [32]:
```
# B0 = Mean(Y) – B1 * Mean(X)
b0 = mean_y - (b1*mean_x)
b0
```

Out[32]:
```
-2.199999089339073
```

In [23]:
```
# regression equation

#y = -2.2 + 2.16*x
```

In [ ]:
```

```

```
In [35]: # Importing the libraries
         import numpy as np
         import matplotlib.pyplot as plt
         import pandas as pd
```

```
In [36]: # Importing the dataset
         dataset = pd.read_csv('Salary_Data.csv')
```

```
In [37]: # Preprocessing ( data clean , data formation , co-relation)
```

```
In [38]: X = dataset.iloc[:, :-1].values
         y = dataset.iloc[:, 1].values
```

```
In [52]: y
```

```
Out[52]: 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.])
```

```
In [42]: !conda list scikit-learn
```

```
# packages in environment at C:\Anaconda3\envs\daailab2020:
#
# Name                    Version                   Build  Channel
scikit-learn              0.23.1           py37h25d0782_0
```

```
In [43]: # Splitting the dataset into the Training set and Test set
         from sklearn.model_selection import train_test_split
         X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 0)
```

```
In [46]: y_train
```

```
Out[46]: array([112635.,  55794.,  83088., 101302.,  56642.,  66029.,  64445.,
                 61111., 113812.,  91738.,  46205., 121872.,  60150.,  39891.,
                 81363.,  93940.,  57189.,  54445., 105582.,  43525.,  39343.,
                 98273.,  67938.,  56957.])
```

```
In [47]: # Fitting Simple Linear Regression to the Training set
         from sklearn.linear_model import LinearRegression
         regressor = LinearRegression()
         regressor.fit(X_train, y_train)
```

```
Out[47]: LinearRegression()
```

```
In [48]: # Predicting the Test set results
         y_pred = regressor.predict(X_test)
```

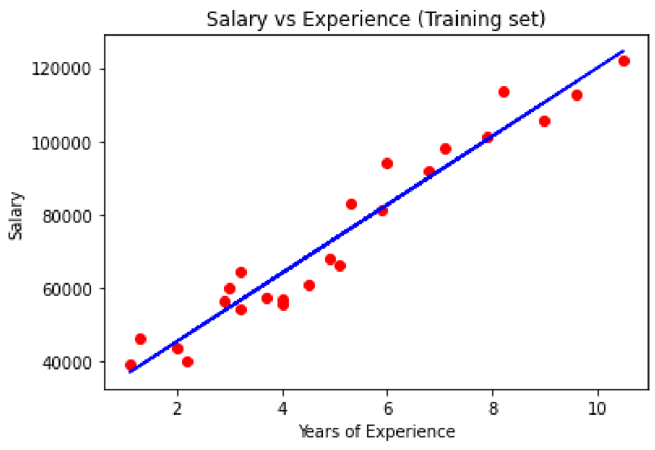
```
In [49]: y_pred
```

```
Out[49]: array([ 40748.96184072, 122699.62295594,  64961.65717022,  63099.14214487,
                115249.56285456, 107799.50275317])
```

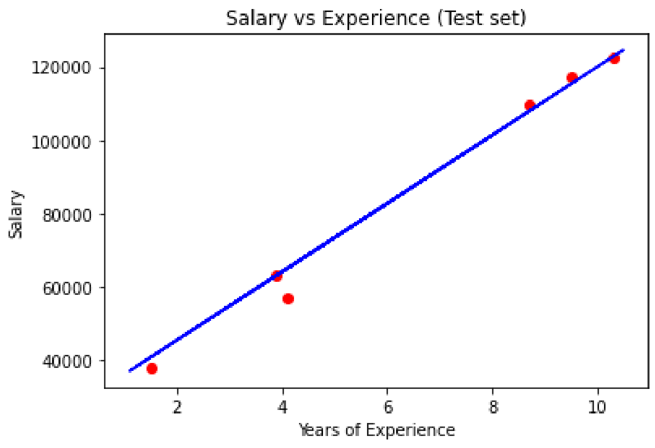
```
In [50]: y_test-y_pred
```

```
Out[50]: array([-3017.96184072,  -308.62295594, -7880.65717022,   118.85785513,
                 1719.43714544,  1631.49724683])
```

In [11]:
```python
# 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 [15]:
```python
# Visualising the Test set results
plt.scatter(X_test, y_test, color = 'red')
plt.plot(X_train, regressor.predict(X_train), color = 'blue')
plt.title('Salary vs Experience (Test set)')
plt.xlabel('Years of Experience')
plt.ylabel('Salary')
plt.show()
```



In [ ]:
```python

```