REGRESSION

What is Regression?

A regression problem is when the output variable is a real or continuous value, such as “salary” or “weight”. Many different models can be used, the simplest is the linear regression. It tries to fit data with the best hyper-plane which goes through the points.

Thing Under the Regression:

- Linear Regression
- Backward Elimination
- Polynomial Regression

What is Linear Regression?

Linear regression algorithm shows a linear relationship between a dependent (y) and one or more independent (y) variables, hence called as linear regression. Since linear regression shows the linear relationship, which means it finds how the value of the dependent variable is changing according to the value of the independent variable.

Types of Linear Regression

Linear regression can be further divided into two types of the algorithm:

- Simple Linear Regression:
  If a single independent variable is used to predict the value of a numerical dependent variable, then such a Linear Regression algorithm is called Simple Linear Regression.
- Multiple Linear regression:
  If more than one independent variable is used to predict the value of a numerical dependent variable, then such a Linear Regression algorithm is called Multiple Linear Regression.

SIMPLE LINEAR REGRESSION

Simple Linear regression is a type of Regression algorithm, that models the relationship between a dependent variable and a single independent variable.

The relationship shown by a simple linear regression model is linear or Sloped Straight line, hence its called Simple Linear Regression.

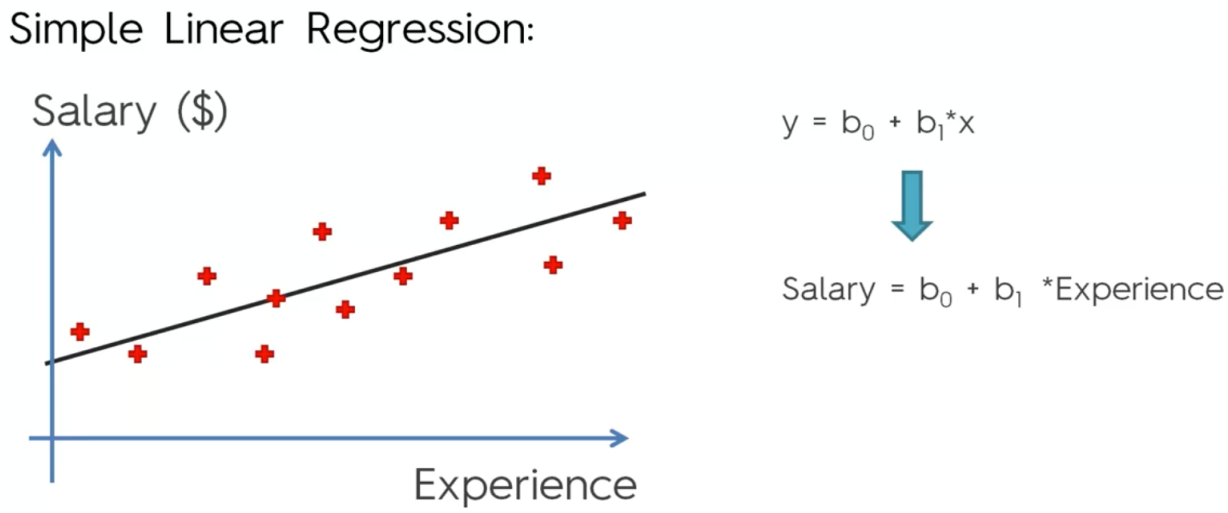
The Simple Linear Regression Model can be represented using the below equation. **Equation →** $\mathbf{Y= b_0+b_1*X}$

Y= denotes the Dependent variable

X= denoted the Independent Variable

b_0 = It is a constant term.

b_1 = It is a co-efficient for the Independent variable



Problem Statement Example for the Simple Linear Regression:

Assume we are taking a dataset that has two variables:

- Salary– (Dependent Variable)
- Experience- (Independent Variable)

- We need to Find the correlation between the two variables.
- Need to find out the best fit line for the dataset.
- How the Dependent variable is changing by changing the dependent variable?

STEP BY STEP PROCEDURE FOR SIMPLE LINEAR REGRESSION MODEL PROGRAM

Steps of Simple Linear Regression Model:

- Data Preprocessing
- Fitting the Simple Linear Regression to the Training Set
- Prediction of test set result
- Visualizing the Training set results
- Visualizing the Test set results

Data Preprocessing:

Data Preprocessing is the main proportion of the Program. We need Import the Libraries and Dataset, to train and test the data's.

//Python Program

//Importing Libraries and Dataset

```
import numpy as np

import matplotlib.pyplot as plt

import pandas as pd

dataset = pd.read_csv('Salary_Data.csv')

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

Y = dataset.iloc[:, 1].values

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, random_state = 0)
```

Fitting the Simple Linear Regression to the Training Set:

Now, we will import the **LinearRegression** class of the **linear_model** library from the **scikit learn**. After importing the class, we are going to create an object of the class named as a **regressor**.

//Python Code

// Fitting Simple Linear Regression to the Training Set

```
from sklearn.linear_model import LinearRegression

regressor=LinearRegression()

regressor.fit(X_train, Y_train)
```

fit() : This method to fit our Simple Linear Regression object to the training set

Predicting the test set results:

So, now, our model is ready to predict the output for the new observations. In this step, we will provide the test dataset (new observations) to the model to check whether it can predict the correct output or not.

We will create a prediction vector **y_pred**, and **x_pred**, which will contain predictions of test dataset, and prediction of training set respectively.

//Python Program

#Predicting the Test set results

```
Y_pred=regressor.predict(X_test)

X_pred=regressor.predict(X_train)
```

y_pred & x_pred: It will contain predictions of test dataset.

Visualizing the Training set results:

we will visualize the training set result. To do so, we will use the scatter() function of the pyplot library, which we have already imported in the pre-processing step.

//Python Program

#Visualizing the Training set results

```
plt.scatter(X_train,Y_train,color='red')

plt.plot(X_train,X_pred,color='blue')

plt.title('Salary vs Experience (Training set)')

plt.xlabel("Years of Experience")

plt.ylabel('Salary')

plt.show()
```

scatter(): This function will create a scatter plot of observations.

plot(): This function use the Pyplot library, to use this to plot the Data's.

title(): To display a title to the result.

xlabel() & ylabel(): It will assign the labels of X-axis and Y-axis.

show(): Will generate result of the graph.

Visualizing the Test Set results:

We have visualized the performance of our model on the training set. Now, we will do the same for the Test set. The complete code will remain the same as the above code, except in this, we will use **x_test, and y_test** instead of **x_train and y_train.**

//Python Program

#Visualizing the Test set results

```
plt.scatter(X_test,Y_test,color='red')

plt.plot(X_train,X_pred,color='blue')

plt.title('Salary vs Experience (Test set)')

plt.xlabel("Years of Experience")

plt.ylabel('Salary')

plt.show()
```

scatter(): This function will create a scatter plot of observations.

plot(): This function use the Pyplot library, to use this to plot the Data's.

title(): To display a title to the result.

xlabel() & ylabel(): It will assign the labels of X-axis and Y-axis.

show(): Will generate result of the graph.

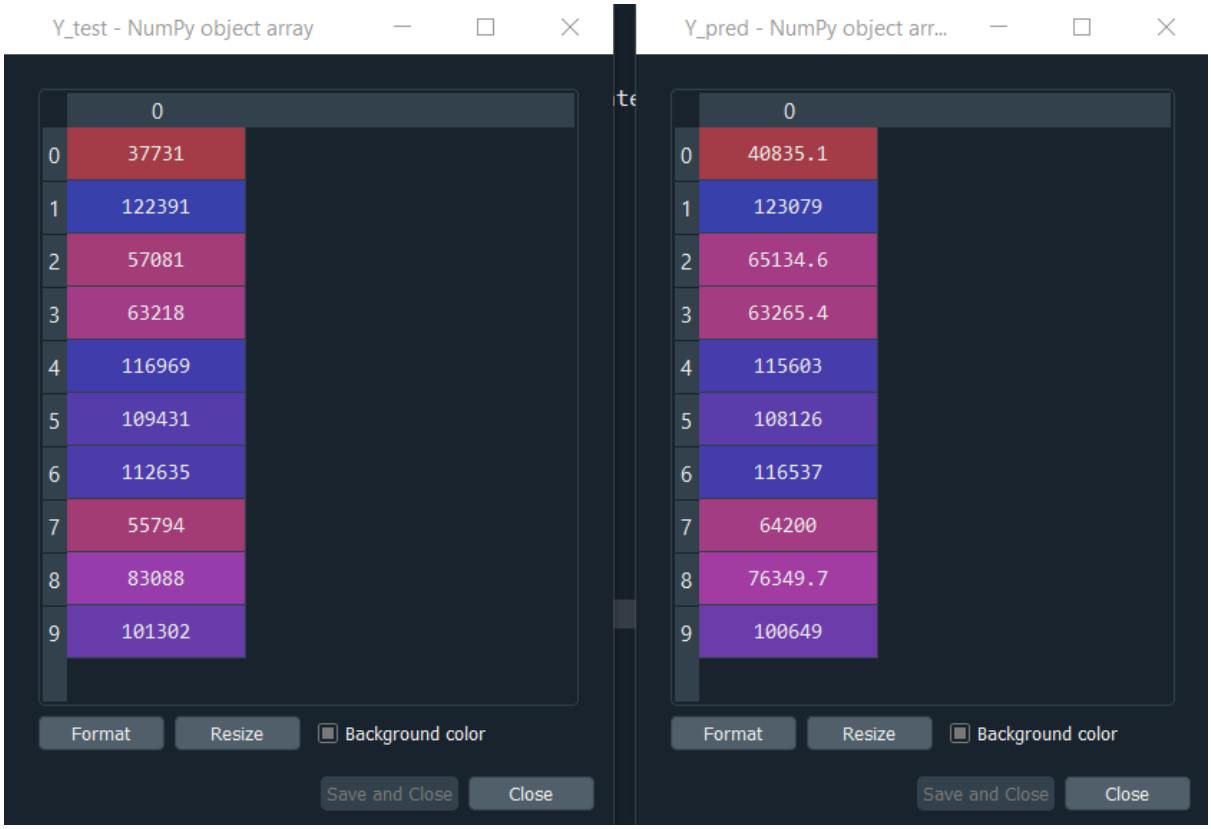
SIMPEL LINEAR REGRESSION TEMPLATE

//Complete Program of Simple Linear Regression

```
import numpy as np

import matplotlib.pyplot as plt

import pandas as pd


dataset = pd.read_csv('Salary_Data.csv')


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

Y = dataset.iloc[:, 1].values

from sklearn.model_selection import train_test_split

X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 1/2,
random_state = 0)


from sklearn.linear_model import LinearRegression

regressor=LinearRegression()

regressor.fit(X_train, Y_train)


Y_pred=regressor.predict(X_test)

X_pred=regressor.predict(X_train)
```

```
plt.scatter(X_train,Y_train,color='red')
plt.plot(X_train,X_pred,color='blue')
plt.title('Salary vs Experience (Training set)')
plt.xlabel("Years of Experience")
plt.ylabel('Salary')
plt.show()

plt.scatter(X_test,Y_test,color='red')
plt.plot(X_train,X_pred,color='blue')
plt.title('Salary vs Experience (Test set)')
plt.xlabel("Years of Experience")
plt.ylabel('Salary')
plt.show()
```

Result/Output of the Model:



Y_test= Training Data Set.

Y_Pred= Prediction Data Set.

In Above Picture, we can see the Test set and predicted set Data of Salary.

Hence the Output of the predicted data is **85%** similar to training set.