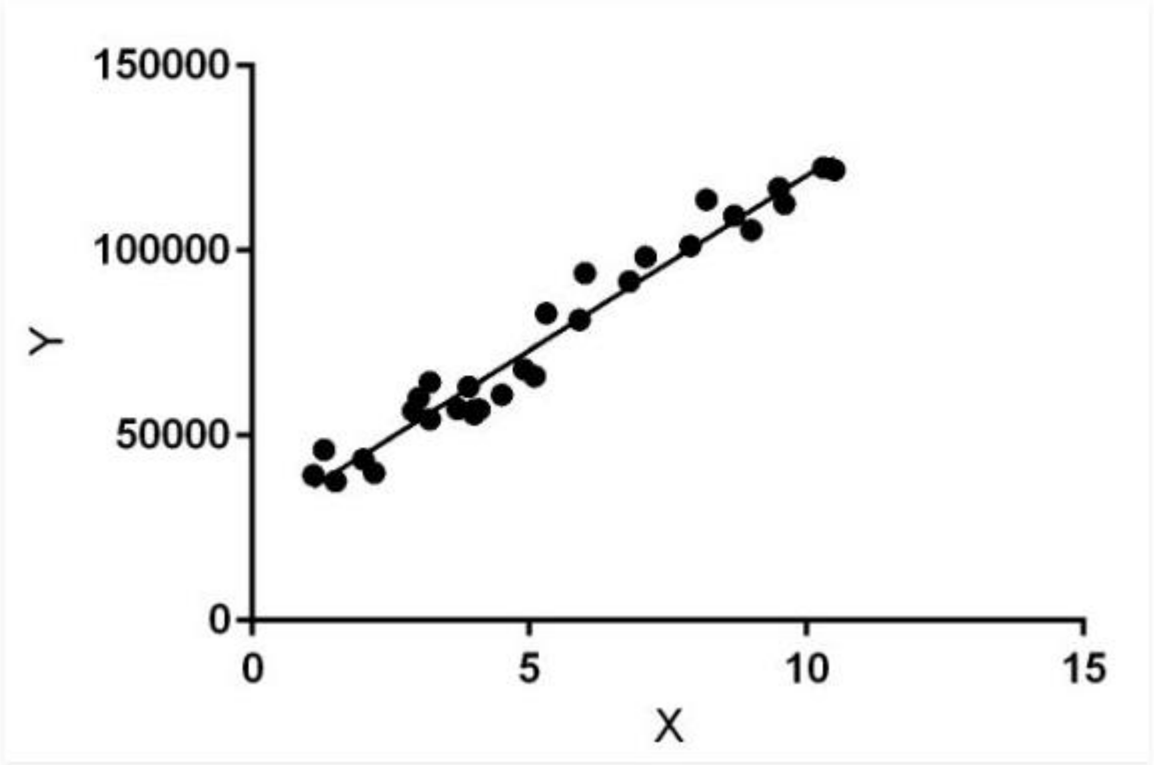
Experiment 4 – Implementation of Simple Linear Regression

Name:	**Shaikh Shadab**	**Rollno**	**:**	**17DCO74**
Class :	**TE.CO**	**Batch**	**:**	**B3**

#Theory

Linear Regression is a machine learning algorithm based on supervised learning. It performs a regression task. Regression models a target prediction value based on independent variables. It is mostly used for finding out the relationship between variables and forecasting. Different regression models differ based on – the kind of relationship between dependent and independent variables, they are considering and the number of independent variables being used.



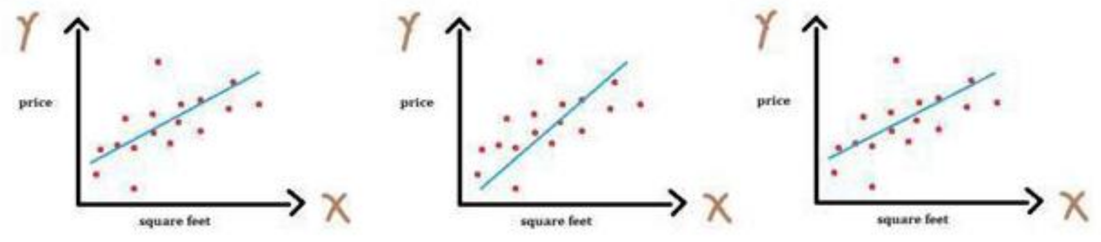
Linear regression performs the task to predict a dependent variable value (y) based on a given independent variable (x). So, this regression technique finds out a linear relationship between x (input) and y(output). Hence, the name is Linear Regression.
In the figure above, X (input) is the work experience and Y (output) is the salary of a person. The regression line is the best fit line for our model.

Hypothesis function for Linear Regression :

$$y = \theta_1 + \theta_2 . x$$

while training the model we are given :
x: input training data (univariate – one input variable(parameter)) - Features
y: labels to data (supervised learning) - Target
When training the model – it fits the best line to predict the value of y for a given value of x. The model gets the best regression fit line by finding the best θ1 and θ2 values.
θ1: intercept
θ2: coefficient of x
Once we find the best θ1 and θ2 values, we get the best fit line. So when we are finally using our model for prediction, it will predict the value of y for the input value of x.

- **Simple linear regression**
  In simple linear regression, we establish a relationship between target variable and input variables by fitting a line, known as the regression line.
  In general, a line can be represented by linear equation y = m * X + b. Where, y is the dependent variable, X is the independent variable, m is the slope, b is the intercept.
  In machine learning, we rewrite our equation as y(x) = w0+ w1 * x where w's are the parameters of the model, x is the input, and y is the target variable. Different values of w0 and w1 will give us different lines, as shown below



- **Cost Function (J) to update θ1 and θ2 values to get the best fit line:**
  By achieving the best-fit regression line, the model aims to predict y value such that the error difference between predicted value and true value is minimum. So, it is very important to update the θ1 and θ2 values, to reach the best value that minimize the error between predicted y value (pred) and true y value (y).

$$J(w) = \frac{1}{2n} \sum_{i=1}^{n} (y(x^i) - y^i_{true})^2$$

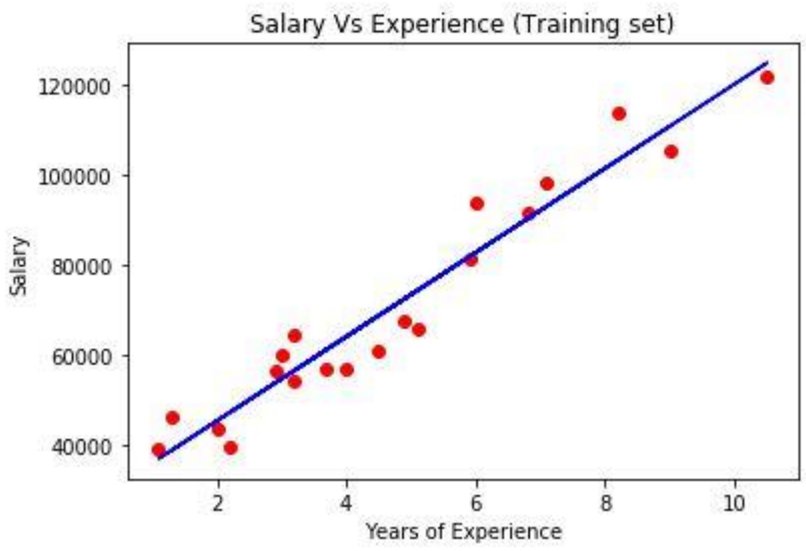
  Cost function(J) of Linear Regression is the Root Mean Squared Error (RMSE) between predicted y value (pred) and true y value (y). y^ denotes predicted output value and y denotes actual value. n is the number of training sample and 2 is to normalize the form.

- **Gradient Descent:**
  To update θ1 and θ2 values in order to reduce Cost function (minimizing RMSE value) and achieving the best fit line the model uses Gradient Descent. The idea is to start with random θ1 and θ2 values and then iteratively updating the values, reaching minimum cost.

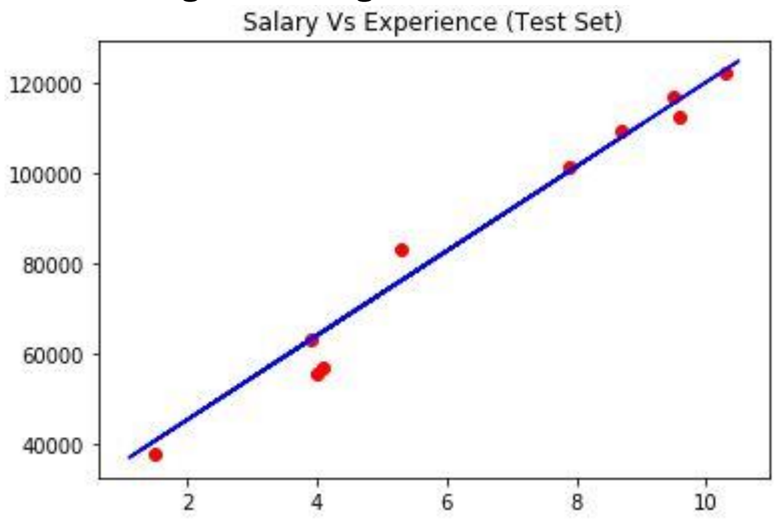
#Source Code

```
import numpy as np
import pandas as pd
import matplotlib as mpl
import matplotlib.pyplot as plt                    #importing all the necessary libraries

#importing dataset
dataset=pd.read_csv('Salary_Data.csv')
X=dataset.iloc[:,:-1].values                       #Getting all rows and first column from dataset
y=dataset.iloc[:,1:].values                        #Getting all rows & second column from dataset

#Splitting the dataset into the training and test sets
from sklearn.model_selection import train_test_split

#used model selection in placed of cross_validation since the latter is deprecated
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=1/3, random_state=0)

#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()

#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)')
```

#Visualising the training set results



#Visualising the testing set results



#Conclusion

In this experiment we have seen the concept of linear regression, and also have seen various aspect associated with it such as hypothesis function, Cost function, and gradient descent
We took a basic example of predicting the salary based on the experience and implemented it with the help of simple linear regression and also have visualised it.