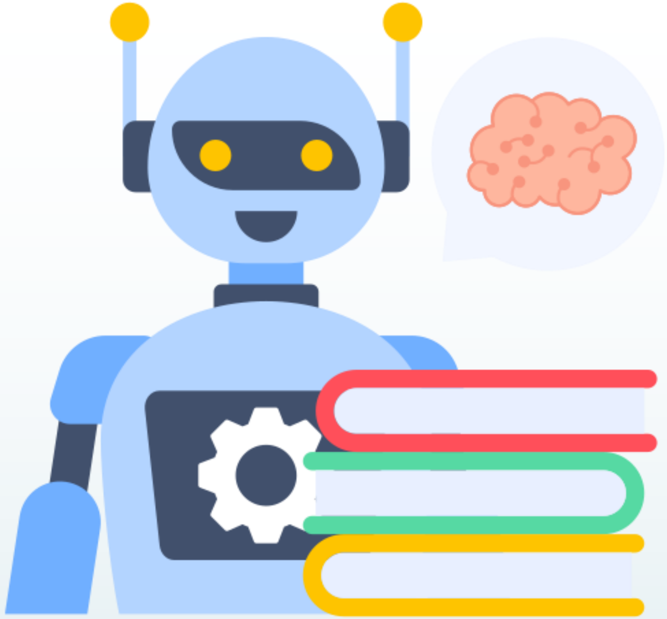
Simple Linear Regression



Presentation by:

Dr. Ashima Tyagi

Outline

- Linear Regression
- Simple linear Regression
- Working

Linear Regression

Regression: Regression is a statistical approach used to analyze the relationship between a dependent variable (target variable) and one or more independent variables (predictor variables).

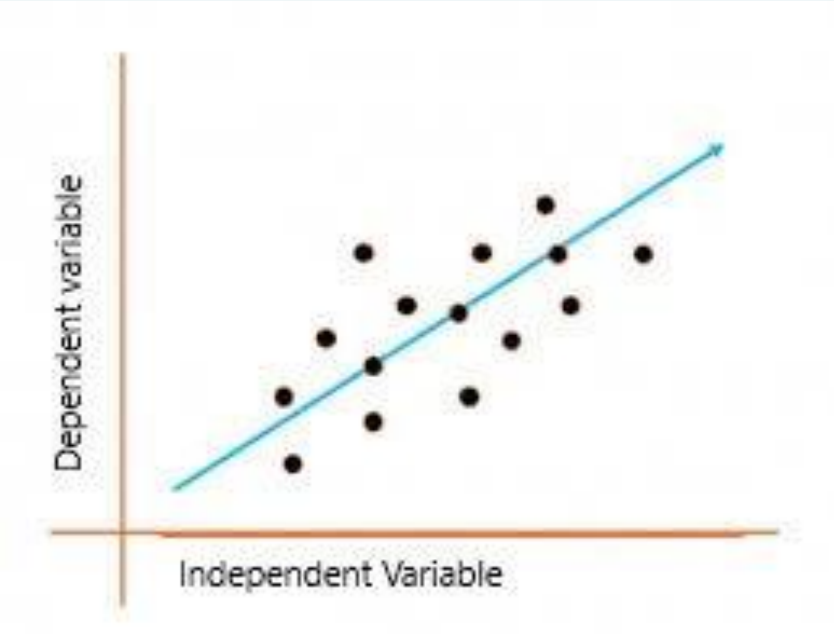
- The objective is to determine the most suitable function that characterizes the connection between these variables.
- It models the relationship between the input features and the target variable, allowing for the estimation or prediction of numerical values.

- Regression is used for **numerical values.**
- It predicts a continuous value based on input features.
- Regression analysis problem works with if output variable is a real or continuous value, such as “salary” or “weight”.
- For example, predicting house prices based on features like square footage, number of bedrooms, etc., is a regression problem because the target variable (house price) is a continuous value.
- When there is only one independent feature, it is known as **Simple Linear Regression**, and when there are more than one feature, it is known as **Multiple Linear Regression.**

Simple Linear Regression

- Simple Linear Regression (SLR) is a statistical model in which there is only one independent variable (or feature) and the *functional relationship between the outcome variable and the regression coefficient is **linear**.*

- The graph presents the linear relationship between the output(y) and predictor(X) variables. The blue line is referred to as the best-fit straight line. Based on the given data points, we attempt to plot a line that fits the points the best.



SLR

This algorithm explains the linear relationship between the dependent(output) variable y and the independent(predictor) variable X using a straight line equation

$$Y = \beta_0 + \beta_1 X + \varepsilon$$

Y = mx+c

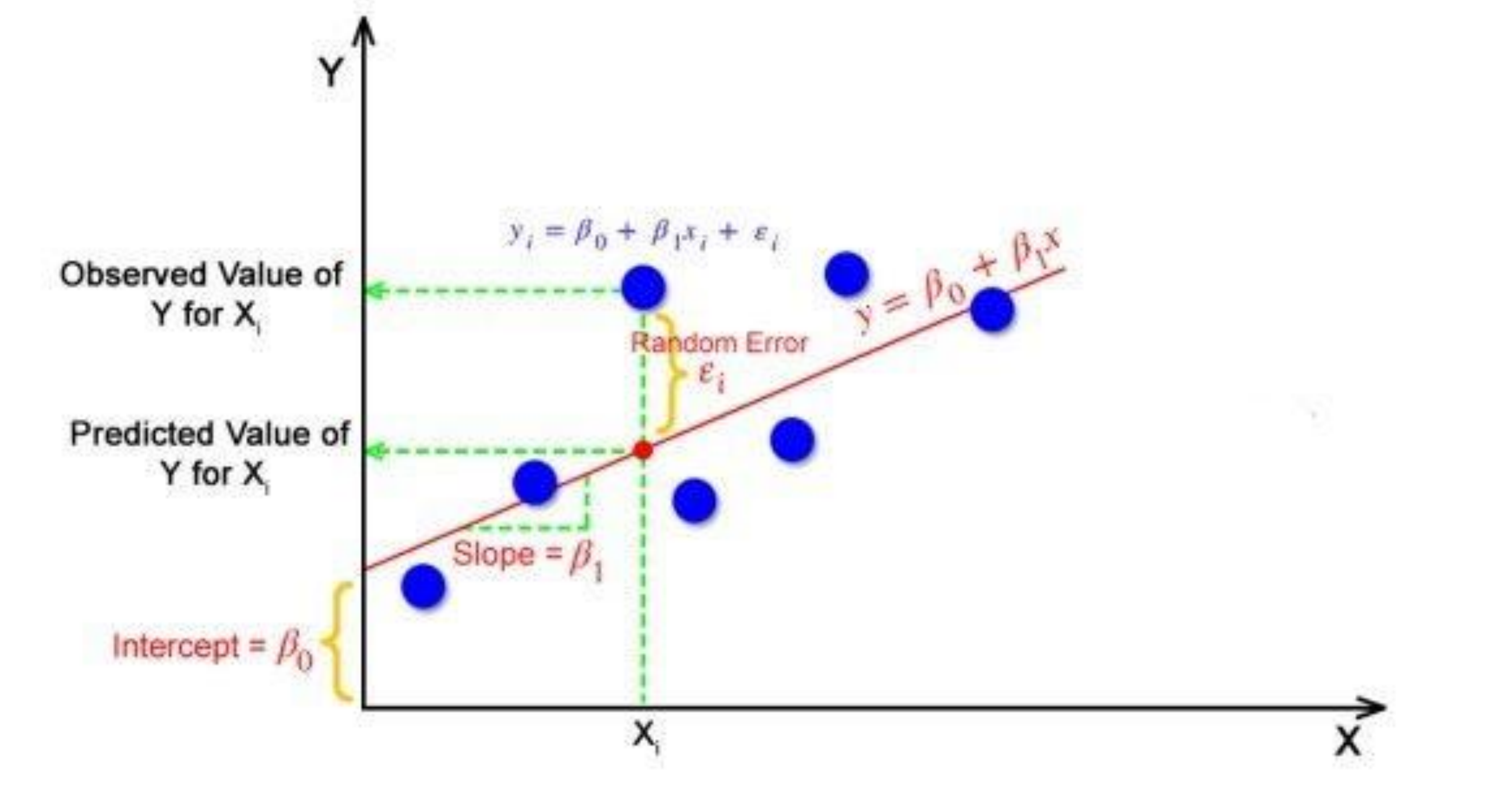
where:

- Y is the dependent variable
- X is the independent variable
- β0 is the intercept
- β1 is the slope of the regression line, which tells whether the line is increasing or decreasing.
- ε : Epsilon is the error

In a regression equation, the intercept β0 is the value of the dependent variable (y) when all independent variables (x) are zero.

The graph presents the linear relationship between the output(y) and predictor(X) variables. The pink line is referred to as the best-fit straight line. Based on the given data points, we attempt to plot a line that fits the points the best.

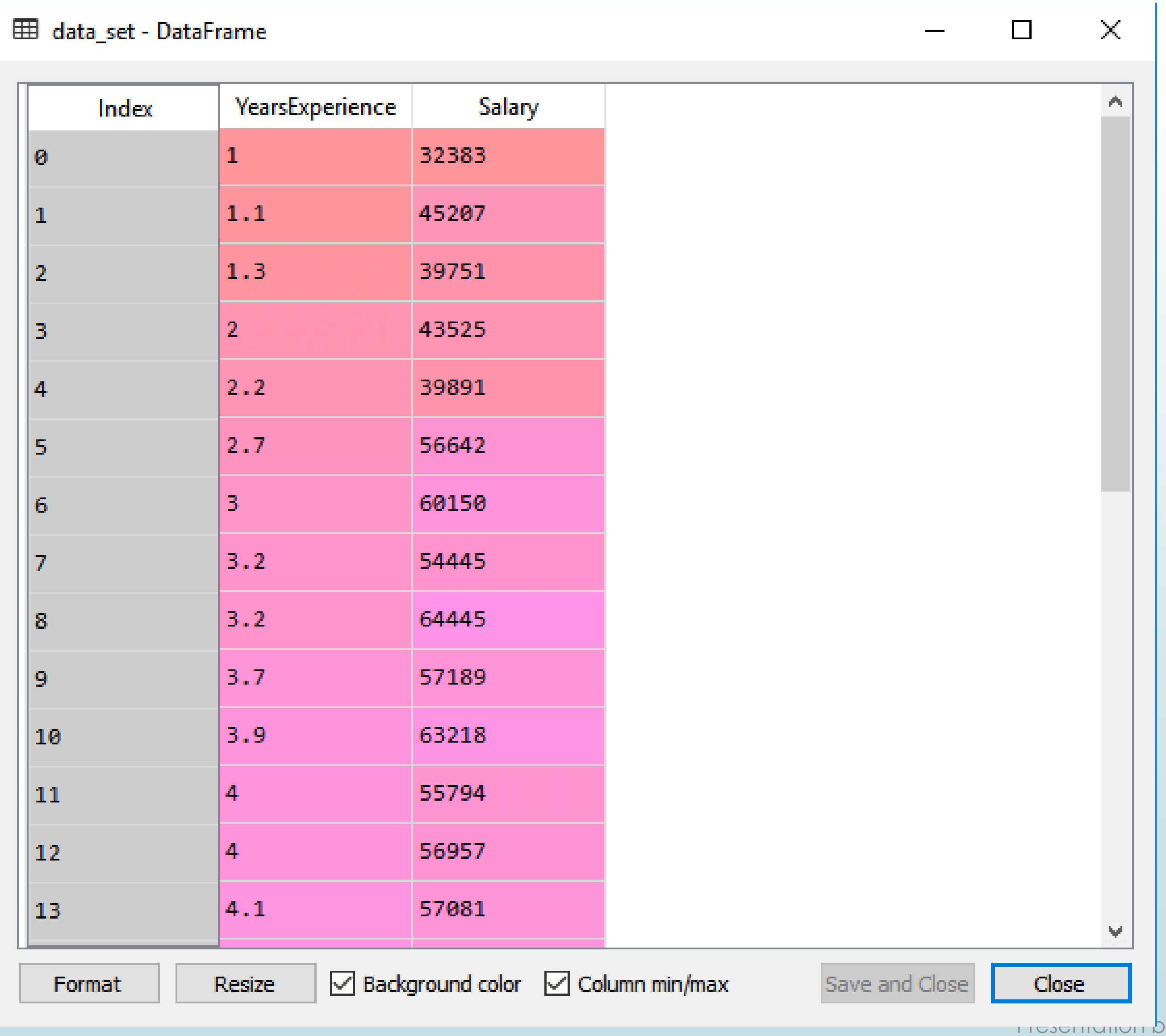
$$Y = \beta_0 + \beta_1 X + \varepsilon$$


Y
$y_i = \beta_0 + \beta_1 x_i + \varepsilon_i$
Observed Value of Y for X_i
$y = \beta_0 + \beta_1 x$
Random Error ε_i
Predicted Value of Y for X_i
Slope = β_1
Intercept = β_0
X_i
X

But how the linear regression finds out which is the best fit line?

The goal of the linear regression algorithm is to get the best values for B0 and B1 to find the best fit line. The best fit line is a line that has the least error which means the error between predicted values and actual values should be minimum.

Implementation of Simple Linear Regression Algorithm using Python



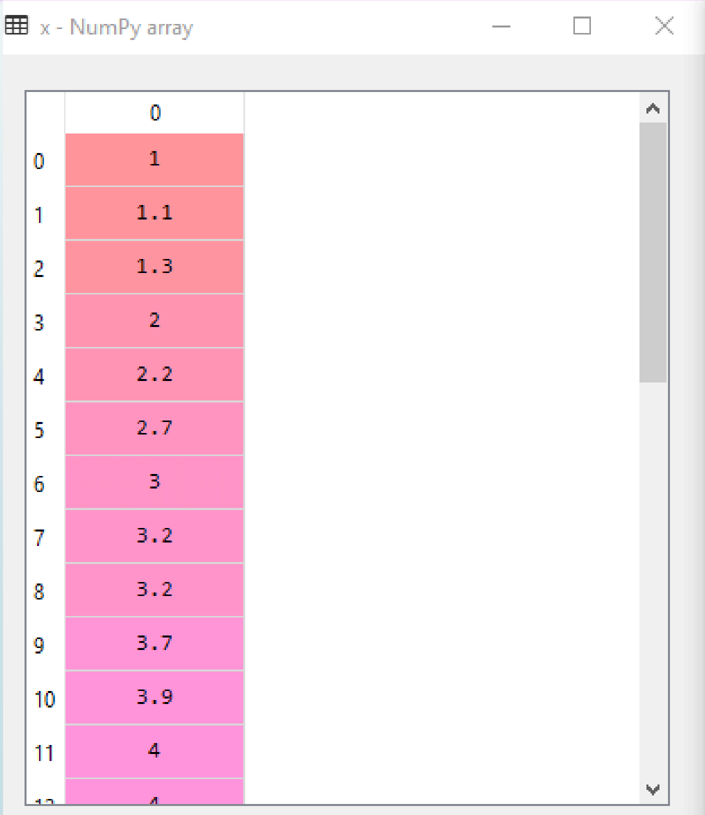
data_set - DataFrame

Index	YearsExperience	Salary
0	1	32383
1	1.1	45207
2	1.3	39751
3	2	43525
4	2.2	39891
5	2.7	56642
6	3	60150
7	3.2	54445
8	3.2	64445
9	3.7	57189
10	3.9	63218
11	4	55794
12	4	56957
13	4.1	57081

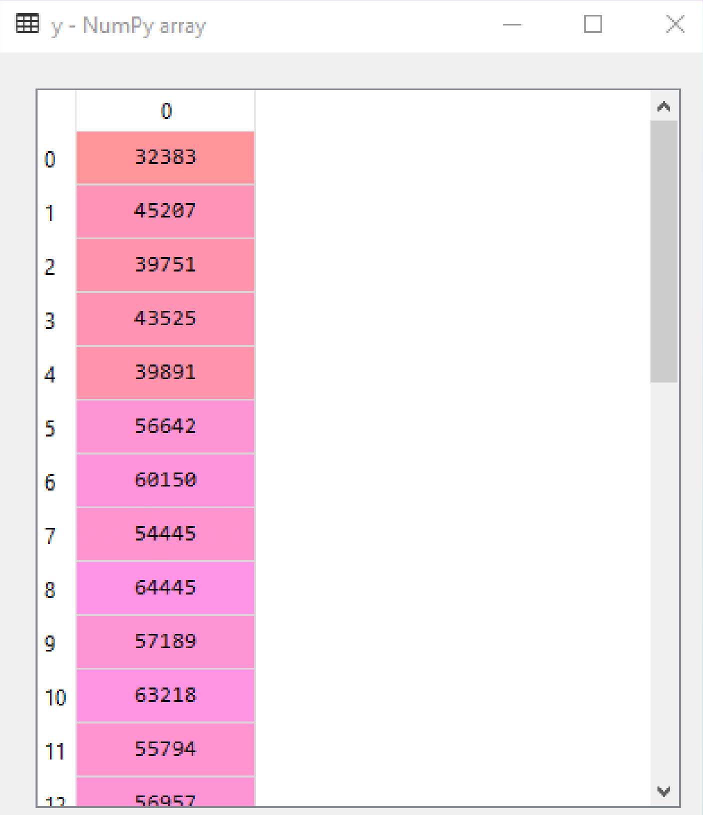
Here we are taking a dataset that has two variables: salary (dependent variable) and experience (Independent variable). The goals of this problem is:

- **We want to find out if there is any correlation between these two variables**
- **We will find the best fit line for the dataset.**
- **How the dependent variable is changing by changing the independent variable.**

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

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

x= data_set.iloc[:, :-1].values
y= data_set.iloc[:, 1].values
```



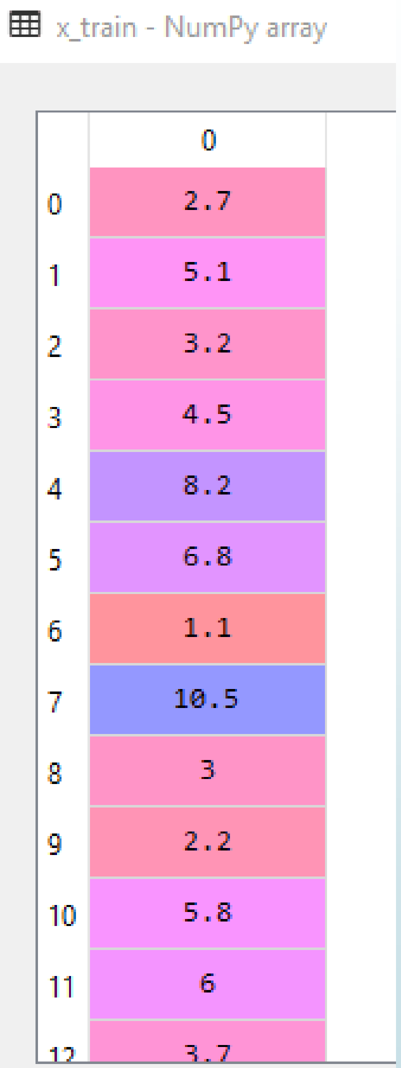
	0
0	1
1	1.1
2	1.3
3	2
4	2.2
5	2.7
6	3
7	3.2
8	3.2
9	3.7
10	3.9
11	4
12	4



	0
0	32383
1	45207
2	39751
3	43525
4	39891
5	56642
6	60150
7	54445
8	64445
9	57189
10	63218
11	55794
12	56957

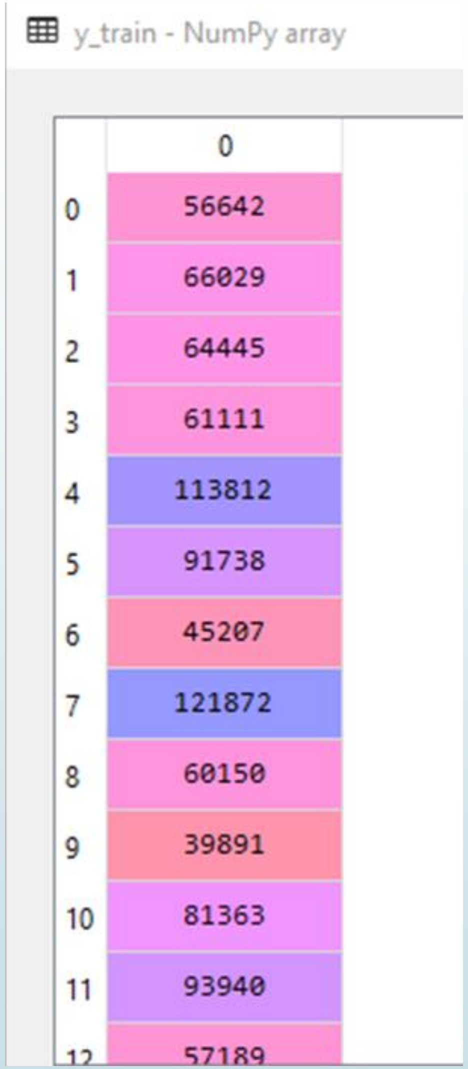
Next, we will split both variables into the test set and training set. We have 30 observations, so we will take 20 observations for the training set and 10 observations for the test set. We are splitting our dataset so that we can train our model using a training dataset and then test the model using a test dataset. The code for this is given below:

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

data_set= pd.read_csv('Salary_Data.csv’)

x= data_set.iloc[:, :-1].values
y= data_set.iloc[:, 1].values

# Splitting the dataset into training 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=0.3, random_state=0)
```



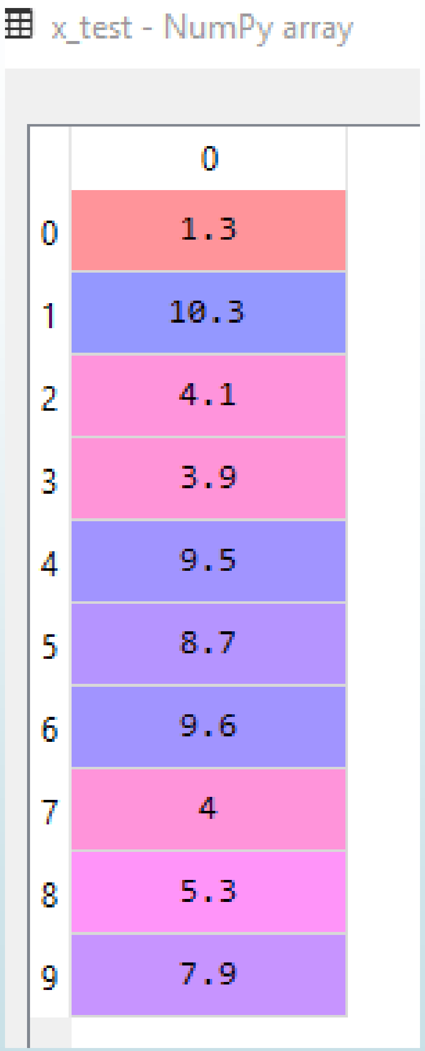
x_train - NumPy array

	0
0	2.7
1	5.1
2	3.2
3	4.5
4	8.2
5	6.8
6	1.1
7	10.5
8	3
9	2.2
10	5.8
11	6
12	3.7



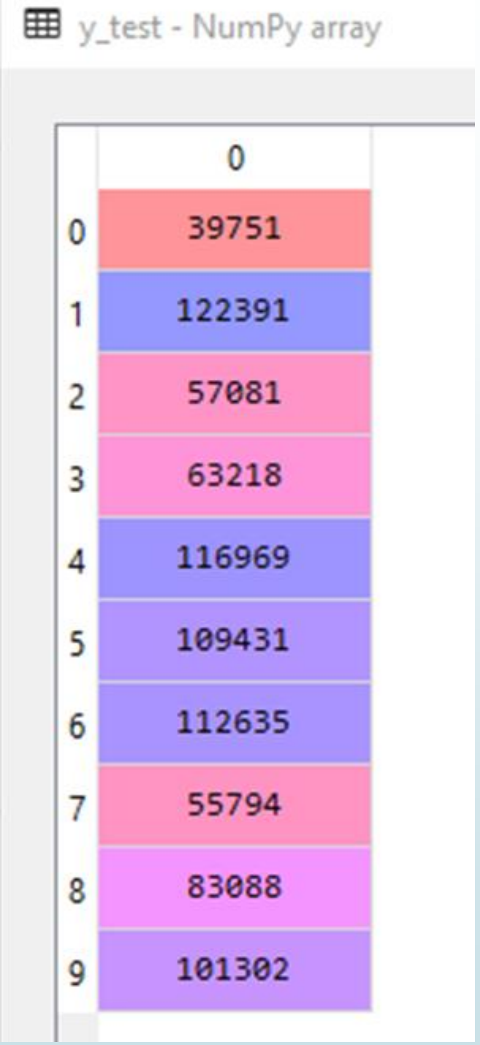
y_train - NumPy array

	0
0	56642
1	66029
2	64445
3	61111
4	113812
5	91738
6	45207
7	121872
8	60150
9	39891
10	81363
11	93940
12	57189



x_test - NumPy array

	0
0	1.3
1	10.3
2	4.1
3	3.9
4	9.5
5	8.7
6	9.6
7	4
8	5.3
9	7.9



y_test - NumPy array

	0
0	39751
1	122391
2	57081
3	63218
4	116969
5	109431
6	112635
7	55794
8	83088
9	101302

Fitting the Simple Linear Regression to the Training Set:

```
#Fitting the Simple Linear Regression model to the training dataset

from sklearn.linear_model import LinearRegression
regressor= LinearRegression()
regressor.fit(x_train, y_train)
```

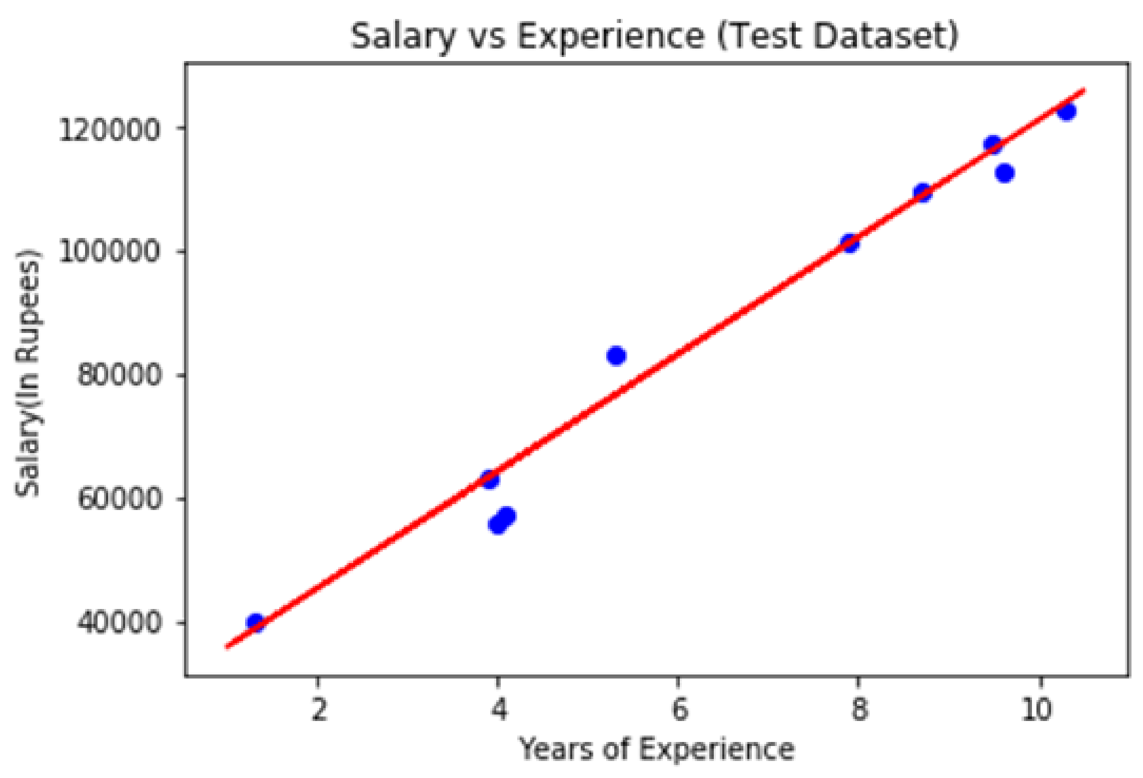
```
Out[7]: LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
```

Prediction of test set result:

```
#Prediction of Test set result

y_pred= regressor.predict(x_test)
```

Visualization Testing

```
#visualizing the Test set results
plt.scatter(x_test, y_test, color="blue")
plt.plot(x_test, y_pred, color="red")
plt.title("Salary vs Experience (Test Dataset)")
plt.xlabel("Years of Experience")
plt.ylabel("Salary(In Rupees)")
plt.show()
```

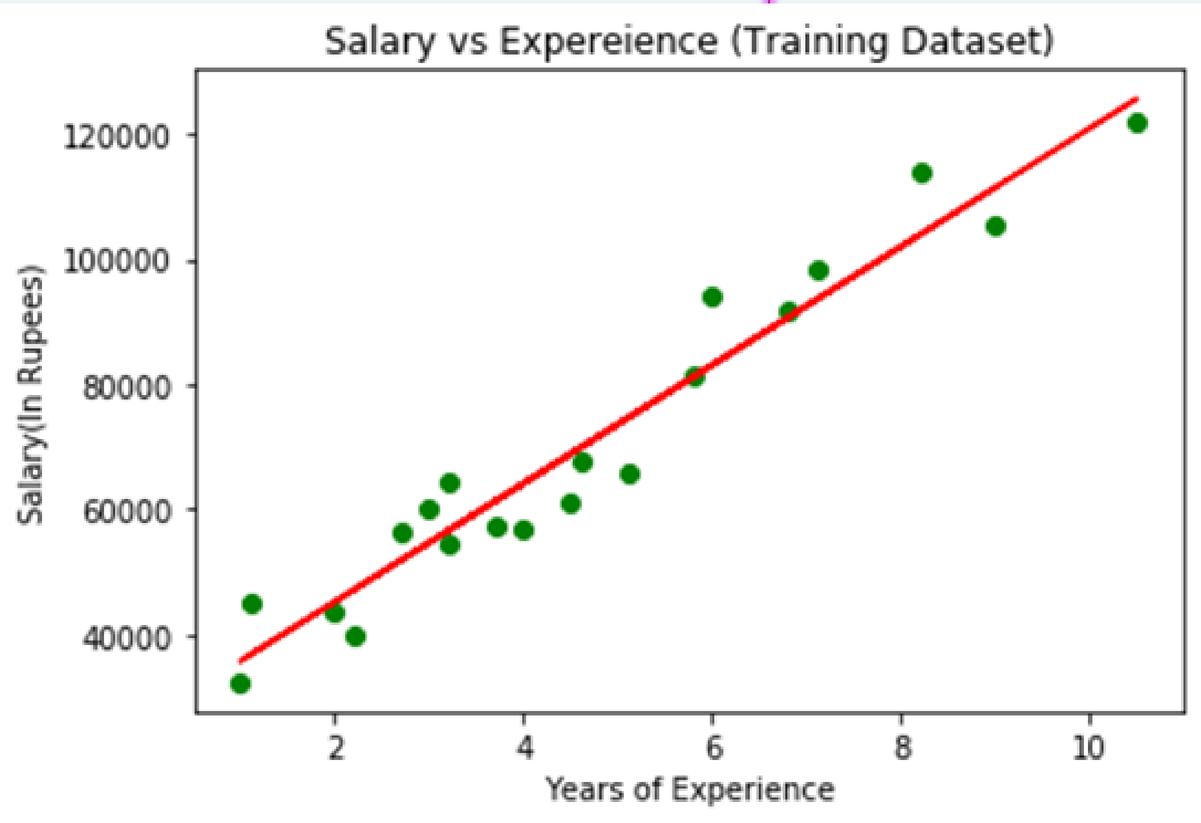


Prediction of train set result:

```
#Prediction of Training set result

x_pred= regressor.predict(x_train)
```

Visualization Training

```
plt.scatter(x_train, y_train, color="green")
plt.plot(x_train, x_pred, color="red")
plt.title("Salary vs Experience (Training Dataset)")
plt.xlabel("Years of Experience")
plt.ylabel("Salary(In Rupees)")
plt.show()
```



Advantages:

- Easy to Understand: Simple linear regression is easy to understand and interpret, making it accessible to those without a deep statistical background.
- Computationally Efficient: It requires less computation compared to more complex regression models, making it faster to train and apply.

Disadvantages:

- Underfitting: due to less features
- Limited Complexity: It can only capture linear relationships between the independent and dependent variables, limiting its ability to model more complex relationships.
- Assumes Linearity: It assumes that the relationship between the variables is linear, which may not always be the case in real-world scenarios.
- Sensitivity to Outliers: Simple linear regression is sensitive to outliers, which can significantly affect the model's performance.

Thank You