Linear_Regression

December 7, 2025

Simple Linear Regression

- In statistics, simple linear regression is a linear regression model with a single explanatory variable.
- In simple linear regression, we predict scores on one variable based on results on another.
- The criteria variable Y is the variable we are predicting. Predictor variable X is the variable using which we are making our predictions.
- The prediction approach is known as simple regression as there is only one predictor variable.
- As a result, a linear function that predicts the values of the dependent variable as a function of the independent variable is discovered for two-dimensional sample points with one independent variable and one dependent variable.

Equation : $y = \beta_0 + \beta_1 x$

- This is the simple linear regression equation where β_0 is the constant and β_1 is the slope and describes the relationship between x (independent variable) and y(dependent variable)
- $y_i = \beta_0 + \beta_1 x_i$, β_0 ($y-$intercept) and β_1 (slope) are the coefficients whose values represent the accuracy of predicted values with the actual values.

Implementation In this example, we will use the `salary_Data` concerning the experience ofemployees. In this dataset, we have two columns `YearsExperience` and `Salary`

read.csv('Salary_Data.csv')

[1]:
```
# Import libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
```

[2]:
```
# Get dataset
df_sal = pd.read_csv('Salary_Data.csv')
df_sal.head()
```

[2]:

	YearsExperience	Salary
0	1.1	39343.0
1	1.3	46205.0
2	1.5	37731.0

```
3              2.0  43525.0
4              2.2  39891.0
```

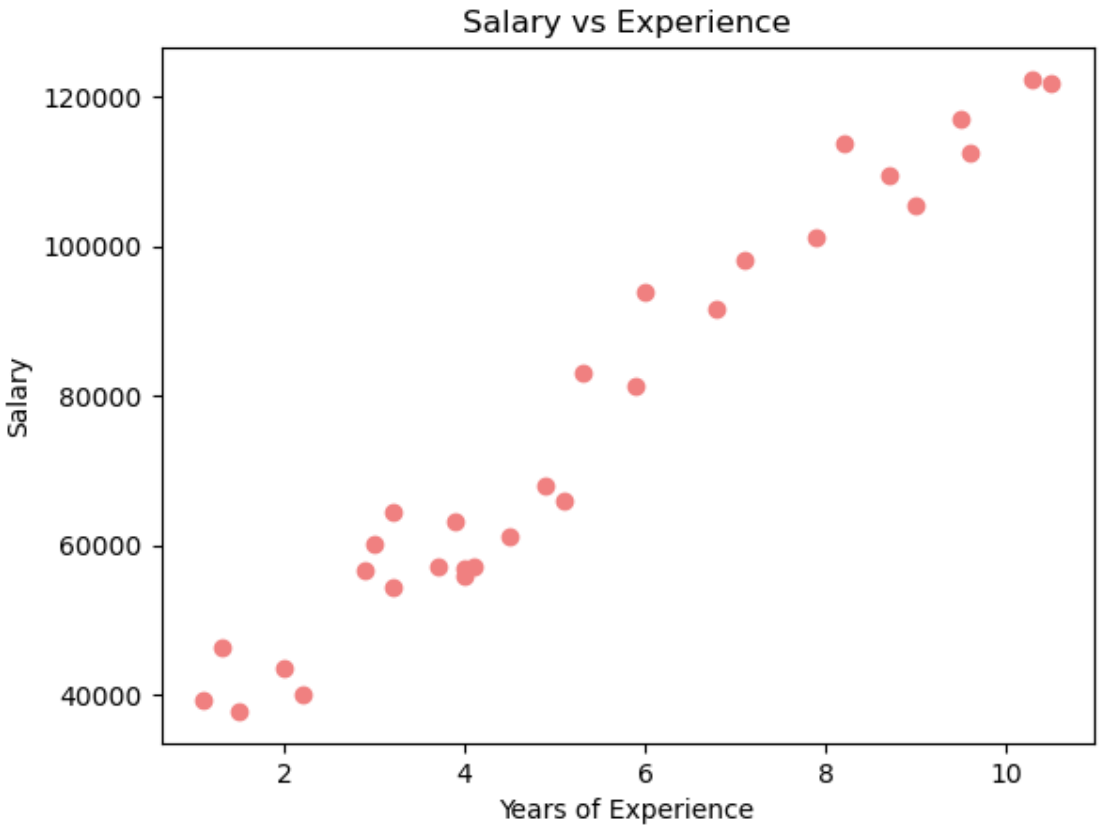
[3]:
```
# Describe data
df_sal.describe()
```

[3]:
```
       YearsExperience         Salary
count        30.000000      30.000000
mean          5.313333   76003.000000
std           2.837888   27414.429785
min           1.100000   37731.000000
25%           3.200000   56720.750000
50%           4.700000   65237.000000
75%           7.700000  100544.750000
max          10.500000  122391.000000
```

- Here, we can see Salary ranges from 37731 to 122391 and a median of 65237

Relationship between Salary and Experience

[4]:
```
plt.scatter(df_sal['YearsExperience'], df_sal['Salary'], color ='lightcoral')
plt.title('Salary vs Experience')
plt.xlabel('Years of Experience')
plt.ylabel('Salary')
plt.box(True)
plt.show()
```



- It is clearly visible now, our data varies linearly.
- That means, that an individual receives more `Salary` as they gain `Experience`.

Split the dataset into dependent/independent variables

[5]:
```
# Splitting variables
x = df_sal.iloc[:,:1]
# independent
y = df_sal.iloc[:,1:]
# dependent
```

Split data into Train/Test sets Further, split your data into training (80%) and test (20%) sets using `train_test_split`

[6]:
```
# Splitting dataset into test/train
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.2,␣
 ↪random_state = 0)
```

Train the regression model

[7]:
```
# Regressor model
regressor = LinearRegression()
```

```
regressor.fit(x_train, y_train)
```

[7]:
```
LinearRegression()
```

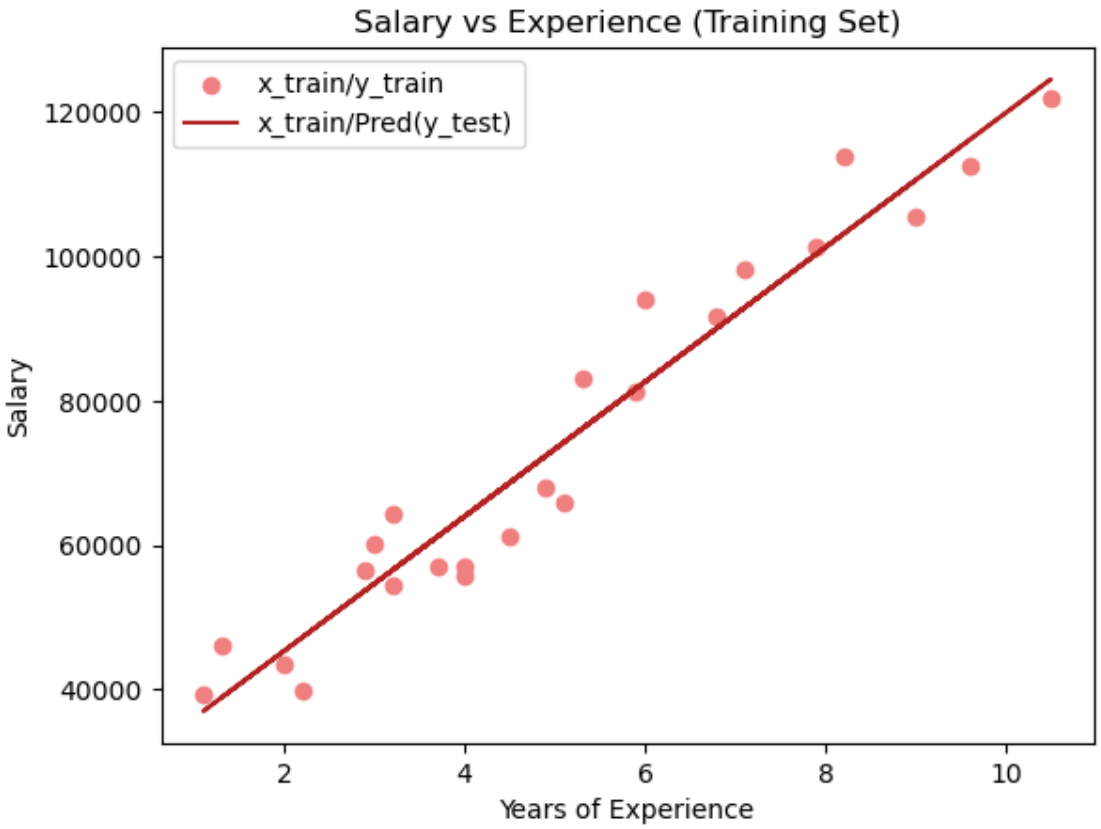
Predict the result Here comes the interesting part, when we are all set and ready to predict any value of y (`Salary`) dependent on x (`Experience`) with the trained model using `regressor.predict`

[8]:
```
# Prediction result
y_pred_test = regressor.predict(x_test)      # predicted value of y_test
y_pred_train = regressor.predict(x_train)    # predicted value of y_train
```

Plot training set data vs predictions

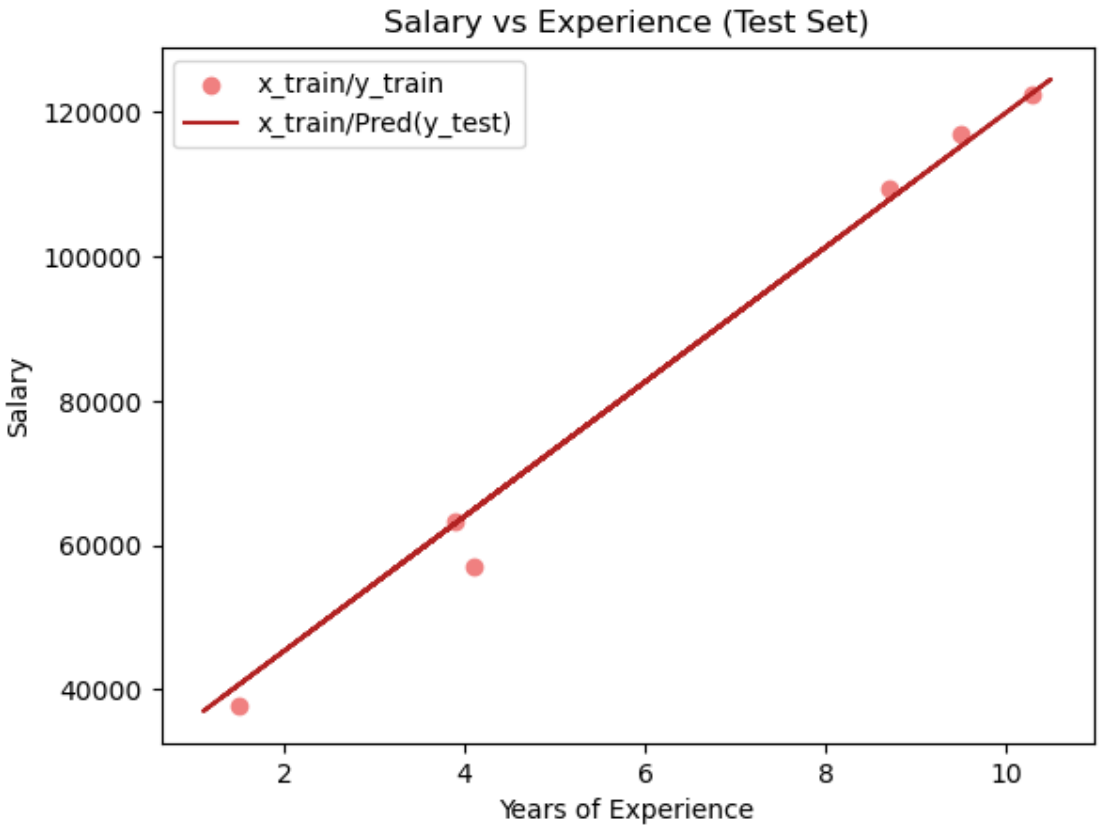
- First we plot the result of training sets (x_train, y_train) with x_train and predicted value of y_train (regressor.predict(x_train))

[9]:
```
# Prediction on training set
plt.scatter(x_train, y_train, color = 'lightcoral')
plt.plot(x_train, y_pred_train, color = 'firebrick')
plt.title('Salary vs Experience (Training Set)')
plt.xlabel('Years of Experience')
plt.ylabel('Salary')
plt.legend(['x_train/y_train', 'x_train/Pred(y_test)'],  loc='best',␣
 ↪facecolor='white')
plt.box(True)
plt.show()
```



Plot test set data vs predictions Secondly, we plot the result of test sets (`X_test, y_test`) with `X_train` and predicted value of `y_train` (`regressor.predict(X_train)`)

[10]:
```
# Prediction on test set
plt.scatter(x_test, y_test, color = 'lightcoral')
plt.plot(x_train, y_pred_train, color = 'firebrick')
plt.title('Salary vs Experience (Test Set)')
plt.xlabel('Years of Experience')
plt.ylabel('Salary')
plt.legend(['x_train/y_train', 'x_train/Pred(y_test)'], loc='best',␣
 ↪facecolor='white')
plt.box(True)
plt.show()
```



[11]:
```
# Regressor coefficients and intercept
print(f'Coefficient beta_1: {regressor.coef_}')
print(f'Intercept beta_0: {regressor.intercept_}')
```

```
Coefficient beta_1: [[9312.57512673]]
Intercept beta_0: [26780.09915063]
```

[ ]: