Simple Linear Regression

May 20, 2021

1 Simple Linear Regression Intution

Simple linear regression can be also known as the best fitting line.

1.1 Importing the libraries

[1]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from matplotlib.pyplot import figure
```

1.2 Importing the dataset

[2]:
```
df = pd.read_csv('Salary_Data.csv')
df
```

[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
5	2.9	56642.0
6	3.0	60150.0
7	3.2	54445.0
8	3.2	64445.0
9	3.7	57189.0
10	3.9	63218.0
11	4.0	55794.0
12	4.0	56957.0
13	4.1	57081.0
14	4.5	61111.0
15	4.9	67938.0
16	5.1	66029.0
17	5.3	83088.0
18	5.9	81363.0
19	6.0	93940.0
20	6.8	91738.0

```
21              7.1   98273.0
22              7.9  101302.0
23              8.2  113812.0
24              8.7  109431.0
25              9.0  105582.0
26              9.5  116969.0
27              9.6  112635.0
28             10.3  122391.0
29             10.5  121872.0
```

[3]:
```
#Check for total number of null values present
df.isnull().sum()
```

[3]:
```
YearsExperience    0
Salary             0
dtype: int64
```

2 Seperating into dependant and independant variables

[4]:
```
X = df.iloc[:,:-1].values #Independant variables
y= df.iloc[:,-1].values
```

[5]:
```
X #Year of experience
```

[5]:
```
array([[ 1.1],
       [ 1.3],
       [ 1.5],
       [ 2. ],
       [ 2.2],
       [ 2.9],
       [ 3. ],
       [ 3.2],
       [ 3.2],
       [ 3.7],
       [ 3.9],
       [ 4. ],
       [ 4. ],
       [ 4.1],
       [ 4.5],
       [ 4.9],
       [ 5.1],
       [ 5.3],
       [ 5.9],
       [ 6. ],
       [ 6.8],
       [ 7.1],
```

```
       [ 7.9],
       [ 8.2],
       [ 8.7],
       [ 9. ],
       [ 9.5],
       [ 9.6],
       [10.3],
       [10.5]])
```

[6]:
```
y # Salary which is dependant on the years of experience
```

[6]:
```
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.])
```

2.1 Splitting the dataset into the Training set and Test set

[7]:
```
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)
```

[8]:
```
X_train
```

[8]:
```
array([[1.1],
       [9.5],
       [8.2],
       [4. ],
       [3.7],
       [5.9],
       [3. ],
       [1.3],
       [3.9],
       [5.1],
       [1.5],
       [7.1],
       [7.9],
       [9.6],
       [8.7],
       [2.2],
       [5.3],
       [2.9],
       [3.2],
       [4.1],
       [6. ],
       [4.5],
       [3.2],
```

```
        [6.8]])
```

[9]:
```
X_test
```

[9]:
```
array([[ 9. ],
       [ 2. ],
       [10.5],
       [ 4.9],
       [ 4. ],
       [10.3]])
```

[10]:
```
y_train
```

[10]:
```
array([ 39343., 116969., 113812.,  56957.,  57189.,  81363.,  60150.,
        46205.,  63218.,  66029.,  37731.,  98273., 101302., 112635.,
       109431.,  39891.,  83088.,  56642.,  64445.,  57081.,  93940.,
        61111.,  54445.,  91738.])
```

[11]:
```
y_test
```

[11]:
```
array([105582.,  43525., 121872.,  67938.,  55794., 122391.])
```

3 Training the Simple Linear Regression Model

[12]:
```
from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(X_train,y_train)
```

[12]:
```
LinearRegression()
```

4 Prediction using the Test set

[13]:
```
y_pred = regressor.predict(X_test)
```

[14]:
```
y_pred
```

[14]:
```
array([112615.32272029,  45005.26910253, 127103.19135266,  73015.14845846,
        64322.42727903, 125171.47553501])
```

5 Visualising the Training set results

[15]:
```
# Xaxis - > Number of years of experrience
#Y axis - > The salary
figure(figsize=(8, 6), dpi=100) #Change size of fig
plt.scatter(X_train,y_train,color ='red')
#Now we plot the regression line
```

```
plt.plot(X_train, regressor.predict(X_train))
plt.title('Salary vs Experience (Training set)')
plt.xlabel('Year of experience')
plt.ylabel('Salary($)')
plt.show()
```

6 Visualising the Test set results

[16]:
```
# Xaxis - > Number of years of experrience
#Y axis - > The salary
figure(figsize=(8, 6), dpi=100) #Change size of fig
plt.scatter(X_test,y_test,color ='red')
#Now we plot the regression line
plt.plot(X_train, regressor.predict(X_train))
plt.title('Salary vs Experience (Training set)')
plt.xlabel('Year of experience')
plt.ylabel('Salary($)')
plt.show()
```

7 Visualising the Training set results

[17]:
```
#Getting equation of the regression line
print(regressor.coef_)
print(regressor.intercept_)
```

```
[9658.57908825]
25688.110926022826
```

Thus, the equation is y = 26718.983361124345+ 9297.61457837 * years of experience

[18]:
```
#Verifying the above equation
print(f"Salary for employee with 10.5 years experience using regressor.predict :
 ↪ {regressor.predict([[10.5]])[0]}")
print(f"Salary for employee with 10.5 years experience using regressor.predict :
 ↪ {26718.983361124345+ 9297.61457837 * 10.5}")
```

```
Salary for employee with 10.5 years experience using regressor.predict :
127103.19135266349
Salary for employee with 10.5 years experience using regressor.predict :
124343.93643400935
```