IMLA Lab: Linear Regression

Prof. P.S. Waghamode

Objective

- To Understand Linear Regression
- Apply LR on standard dataset using SKLearn and
- Predict the unknown.

What is Linear?

Let us take example,

- First, let's say that you are shopping at Bazzar. Whether you buy goods or not, you have to pay $2.00 for parking ticket.
- Each apple price $1.5, and you have to buy an (*x)* item of apple.

Then we can populate a price list as shown in table.

Quantity	Price
1	$ 3.50
2	$ 5.00
3	$ 6.50
4	$ 8.00
5	$ 9.50
…	…
10	$ 17.00
11	$ 18.50
…	…
x	y

It's easy to predict (or calculate) the Price based on Value and vice versa using the equation of ***y=2+1.5x*** for this example or:

$$y = a + bx$$

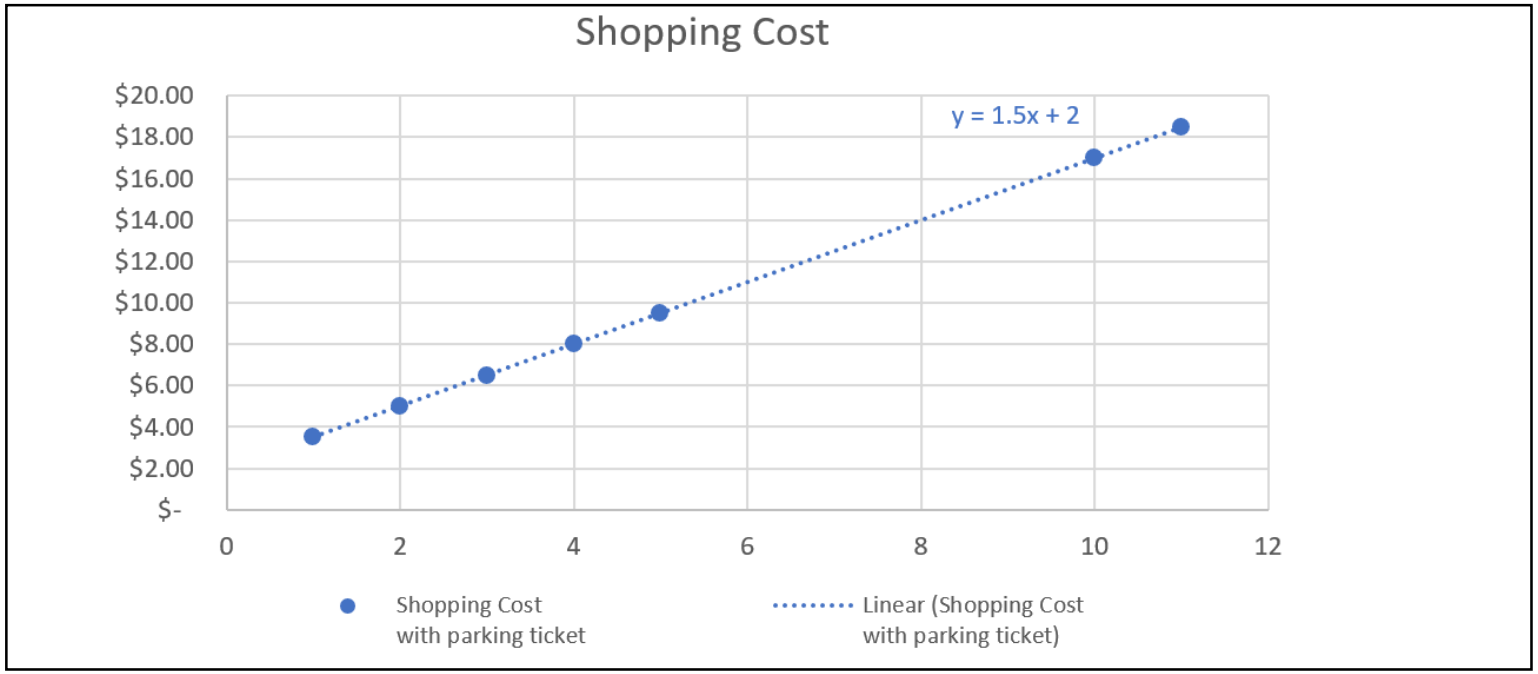
with:

•a = 2 and b = 1.5

- A **linear function** has one independent variable and one dependent variable. The independent variable is ***x*** and the dependent variable is ***y***.
- ***a*** is the constant term or the y intercept. It is the value of the dependent variable when ***x*** = 0.
- ***b*** is the coefficient of the independent variable. It is also known as the slope and gives the rate of change of the dependent variable.
- Why we call it linear? Alright, let's visualize the data set we got above!

After plotting all value of the shopping cost (in blue line), you can see, they all are in **one line**, that's why we call it ***linear***.

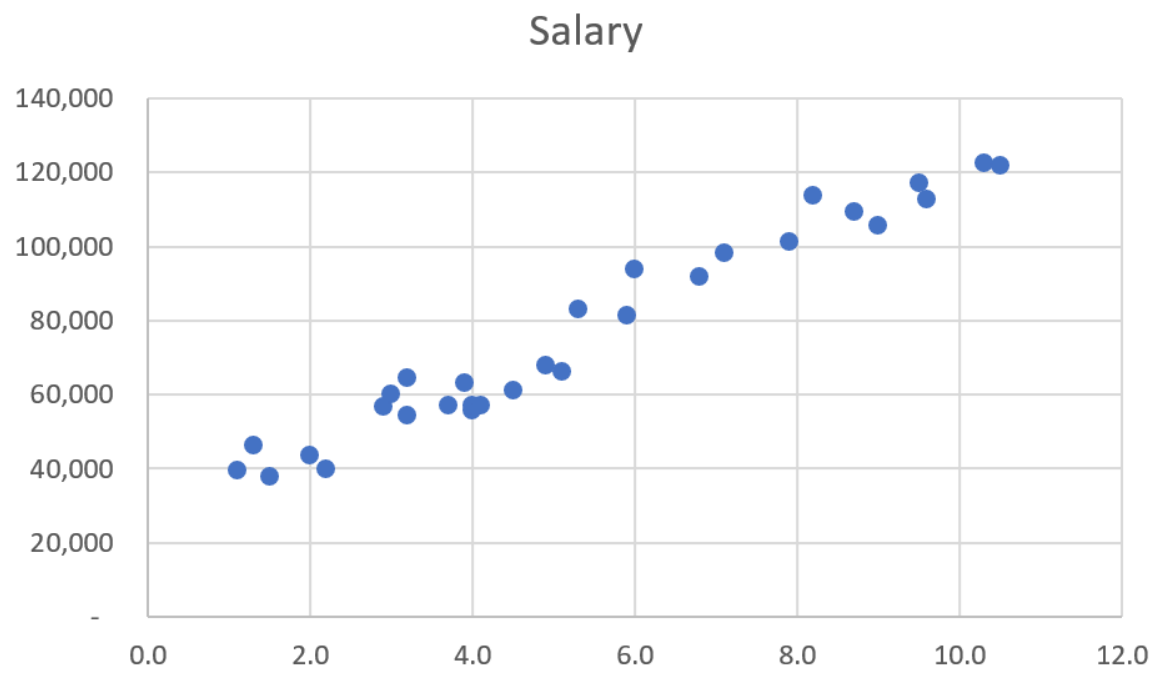
With the equation of linear (y=a+bx), the a is an independent variable. Even if a=0 (you have no need to pay for the parking ticket), the Shopping Cost line will shift down and they are still in a line (orange line).



Example-2

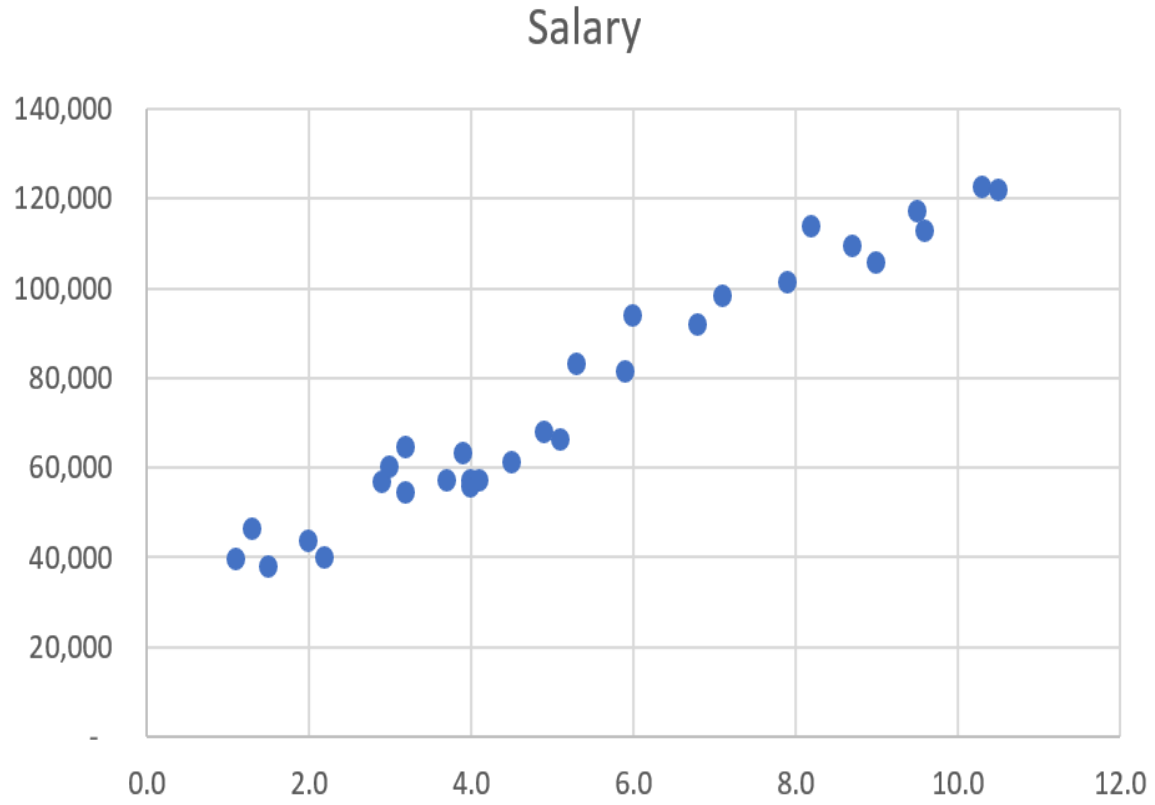
- Let's take another example, in AB Company, there is a salary distribution table based on Year of Experience as per table:

"The scenario is you are a HR officer, you got a candidate with 5 years of experience. Then what is the best salary you should offer to him?"



YearsExperience	Salary
1.1	39,343
1.3	46,205
1.5	37,731
2.0	43,525
2.2	39,891
2.9	56,642
3.0	60,150
3.2	54,445
3.2	64,445
3.7	57,189
3.9	63,218
4.0	55,794
4.0	56,957
4.1	57,081
4.5	61,111
4.9	67,938
5.1	66,029
5.3	83,088
5.9	81,363
6.0	93,940
6.8	91,738
7.1	98,273
7.9	101,302
8.2	113,812
8.7	109,431
9.0	105,582
9.5	116,969
9.6	112,635
10.3	122,391
10.5	121,872

- Please look at this chart carefully. Now we have a bad news: all the observations are not in a line. It means we cannot find out the equation to calculate the (y) value.
- So what now? Don’t worry, we have a good news for you!
- Look at the Scatter Plot again before scrolling down. Do you see it?
- All the points is not in a line BUT they are in a line-shape! **It’s linear!**



- Based on our observation, we can guess that the salary range of 5 Years Experience should be in the red range.

- Of course, we can offer to our candidate any number in that red range. But how to pick the best number for him? It’s time to use Machine Learning to predict the best salary for our candidate.

Linear Regression with Python

Before moving on, we summarize 2 basic steps of Machine Learning as per below:

- Training
- Predict

we will use 4 libraries such as numpy and pandas to work with data set, sklearn to implement machine learning functions, and matplotlib to visualize our plots for viewing:

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


# Importing the dataset
dataset = pd.read_csv('salary_data.csv')
X = dataset.iloc[:, :-1].values #get a copy of dataset exclude last column
y = dataset.iloc[:, 1].values #get array of dataset in column 1st
```

Code explanation:

- dataset: the table contains all values in our csv file
- x: the first column which contains Years Experience array
- y: the last column which contains Salary array

- Next, we have to split our dataset (total 30 observations) into 2 sets: training set which used for training and test set which used for testing:

```
1  # Splitting the dataset into the Training set and Test set
2  from sklearn.model_selection import train_test_split
3  X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=1/3, random_state=0)
```

Code explanation:

•test_size=1/3: we will split our dataset (30 observations) into 2 parts (training set, test set) and the ratio of **test set** compare to dataset is 1/3

• (10 observations will be put into the **test set**. You can put it 1/2 to get 50% or 0.5, they are the same.

• We should not let the test set too big; if it's too big, we will lack of data to train. Normally, we should pick around 5% to 30%.

•train_size: if we use the test_size already, the rest of data will automatically be assigned to train_size.

•random_state: this is the seed for the random number generator.

• We can put an instance of the RandomState class as well. If we leave it blank or 0,

• the RandomState instance used by np.random will be used instead.

- We already have the train set and test set, now we have to build the Regression Model:

```
1  # Fitting Simple Linear Regression to the Training set
2  from sklearn.linear_model import LinearRegression
3  regressor = LinearRegression()
4  regressor.fit(X_train, y_train)
```

Code explanation:

- **regressor = LinearRegression():** our training model which will implement the Linear Regression.
- **regressor.fit:** in this line, we pass the X_train which contains value of **Year Experience** and y_train which contains values of **particular Salary** to form up the model. This is the training process.

visualize our training model and testing model:

```
# Visualizing the Training set results
viz_train = plt
viz_train.scatter(X_train, y_train, color='red')
viz_train.plot(X_train, regressor.predict(X_train), color='blue')
viz_train.title('Salary VS Experience (Training set)')
viz_train.xlabel('Year of Experience')
viz_train.ylabel('Salary')
viz_train.show()

# Visualizing the Test set results
viz_test = plt
viz_test.scatter(X_test, y_test, color='red')
viz_test.plot(X_train, regressor.predict(X_train), color='blue')
viz_test.title('Salary VS Experience (Test set)')
viz_test.xlabel('Year of Experience')
viz_test.ylabel('Salary')
viz_test.show()
```

- We already have the model, now we can use it to calculate (predict) *any values of X depends on y* or *any values of y depends on X*. This is how we do it:
-

```
1  # Predicting the result of 5 Years Experience
2  y_pred = regressor.predict(5)
```