Data Science | 30 Days of Machine Learning | Day - 24

Educator Name: Nishant Dhote
Support Team: **+91-7880-113-112**

----Today Topics | Day 24----

Linear Regression

- Concept of Simple Linear Regression
- For "m & b" closed form solution
- OLS: Ordinary Least Squares regression method
- Find the Total Error and Average Error
- Calculation concept for "b"
- Calculation concept for "m"
- Use SK learn library and find "m & b" value

Dataset Link GitHub: https://github.com/TheiScale/30_Days_Machine_Learning/

- Concept of Simple Linear Regression

In the context of simple linear regression, m and b typically refer to the slope and intercept of the best-fit line, respectively.

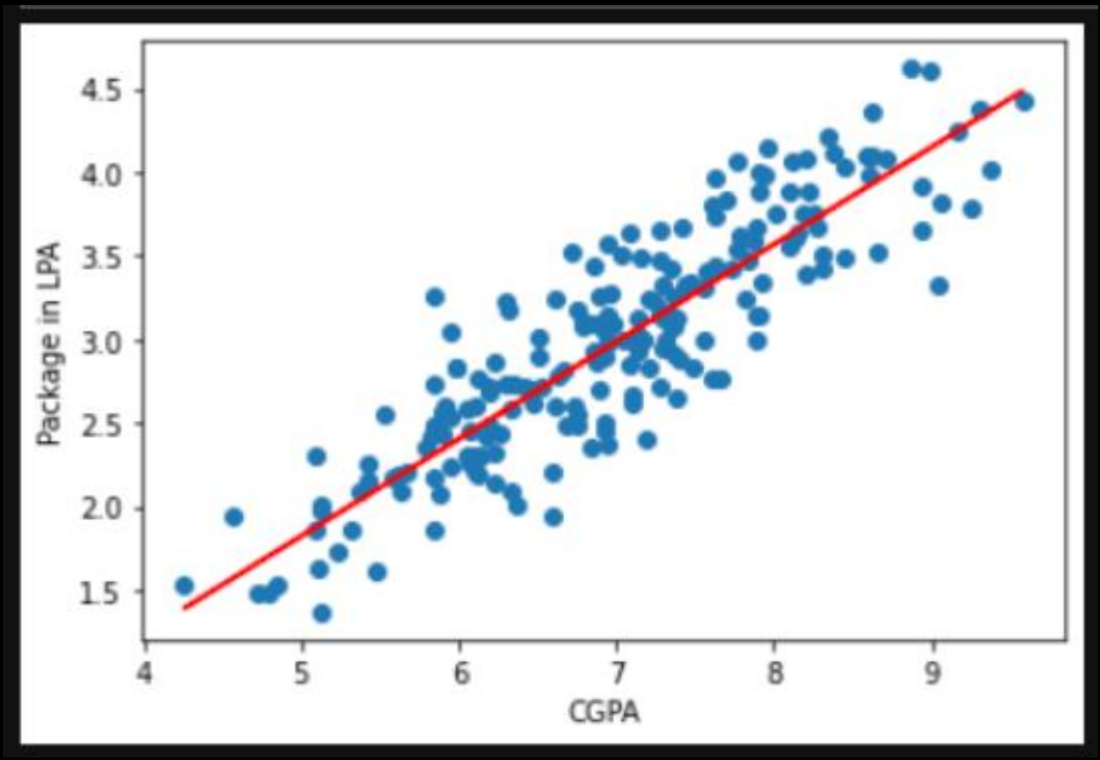
1. **Slope (m)**: The slope of the best-fit line, often denoted as $\beta 1$, represents the rate of change in the dependent variable (Y) for a one-unit change in the independent variable (X). In other words, it indicates how much Y changes on average for each unit change in X. Mathematically, the slope is calculated as the change in Y divided by the change in X for any two points on the line.

2. **Intercept (b)**: The intercept of the best-fit line, often denoted as $\beta 0$, represents the value of Y when X is zero. It is the point where the line

intersects the y-axis. The intercept accounts for the initial value of Y when X is absent or zero.
So, in the equation of the best-fit line:
$Y = mx + b$

- m represents the slope, indicating the rate of change of Y with respect to X.
- b represents the intercept, indicating the value of Y when X is zero.

These parameters m and b are estimated from the data using statistical methods like least squares regression, where the goal is to minimize the sum of the squared differences between the observed values of Y and the values predicted by the line. Once estimated, the best-fit line can be used to predict Y for any given value of X within the range of the data.



- **For “m & b” closed form solution:**

A closed-form expression is a mathematical process that can be completed in a finite number of operations. Closed-form expressions are of interest when trying to develop general solutions to problems. A closed-form solution is a general solution to a problem in the form of a closed-form expression.

https://en.wikipedia.org/wiki/Closed-form_expression

- **For “m & b” Non closed form solution:**

A closed form solution is one that has a simple algebraic expression, so you can simply plug in numbers and get the result. Non-closed form solutions generally need to be computed, often through successive approximation. For example, Newton's method or gradient descent

1D

MF

Find m and b Values

Closed form solution

Non Closed form solution

Direct Formula

OLS: Ordinary Least Squares

Gradient Descent

Stochastic Gradient Descent (SGD) Regression

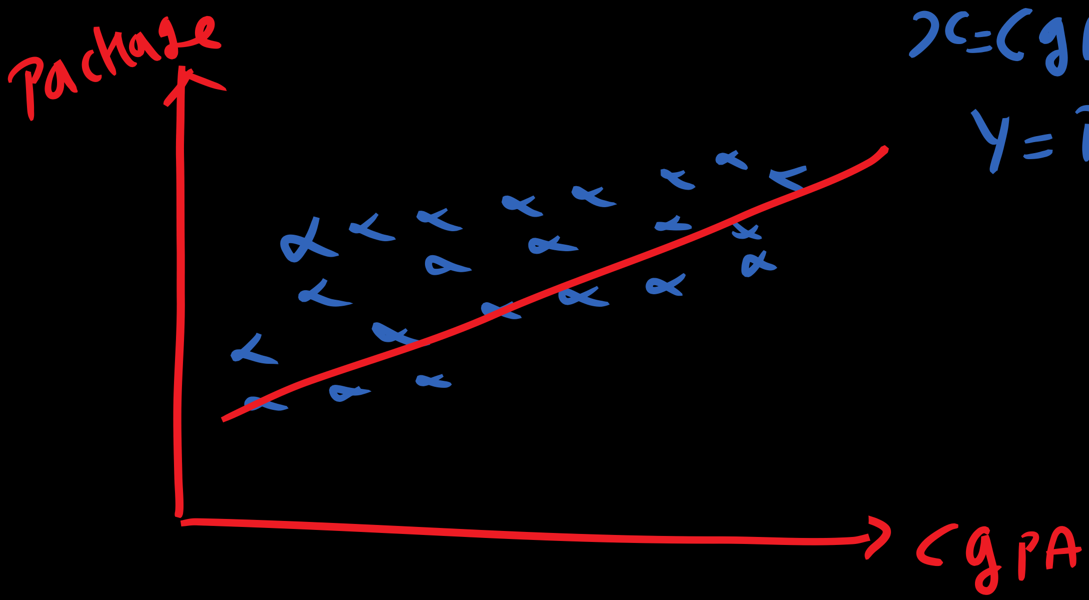
```
X = CGPA
Y = Package
```

$$b = \bar{y} - m\bar{x}$$

$$m = \frac{\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n} (x_i - \bar{x})^2}$$

- **Use OLS and find the value of “m & b”:**

Ordinary Least Squares (OLS) can safely be used for most predictions. It calculates a linear regression by minimizing standard error with respect to the target signal.



x = CGPA

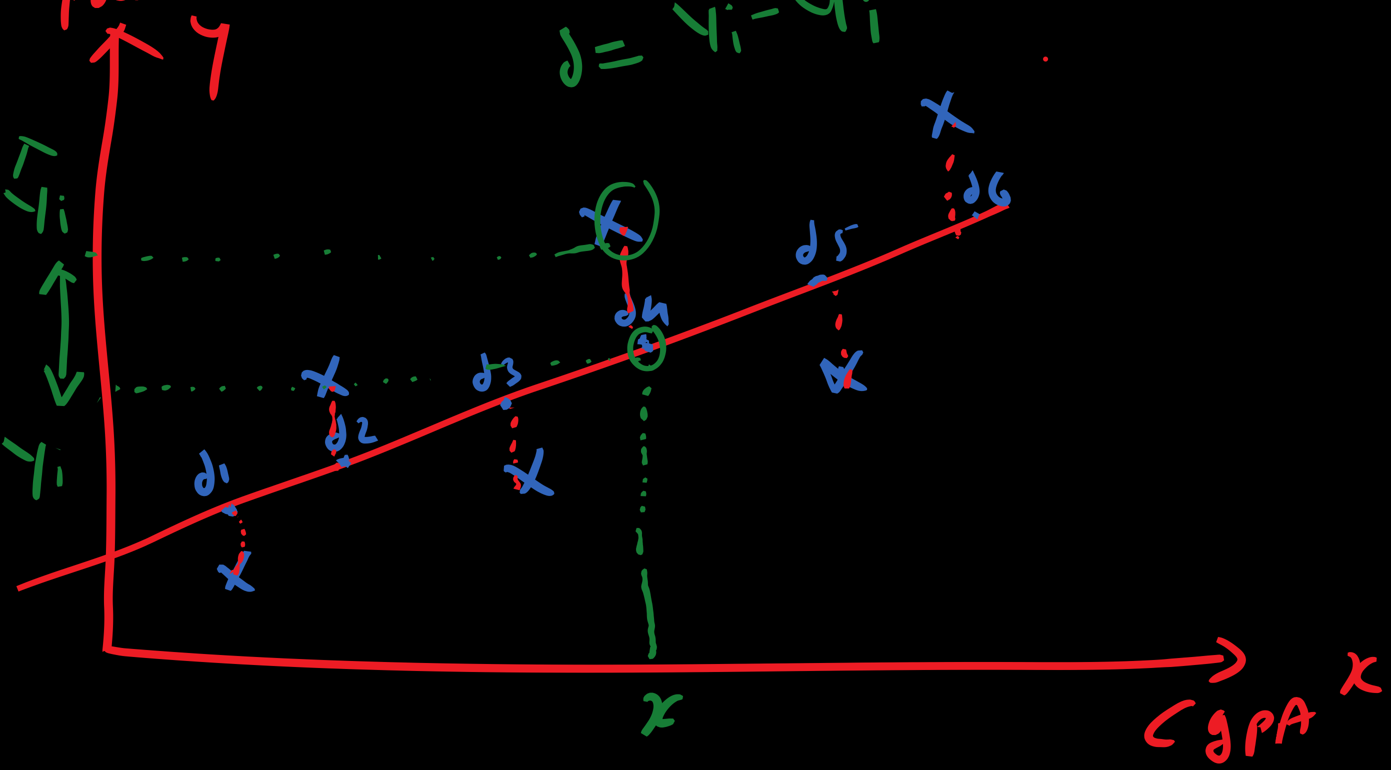
y = Package

$$b = \bar{y} - m\bar{x}$$

$$m = \frac{\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n} (x_i - \bar{x})^2}$$

$\bar{x}$ = mean of CGPA | $\bar{y}$ = mean of Package

$d = y_i - \hat{y}_i$



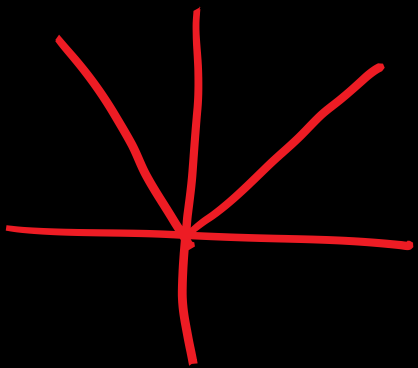
Total distance & Total Error

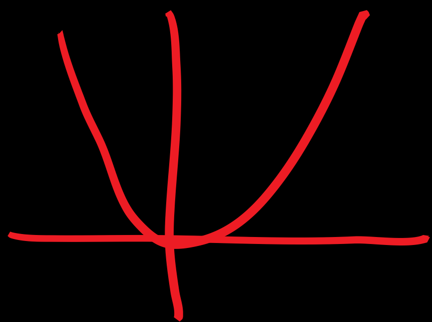
$$E = d_1 + d_2 + d_3 + \dots\dots + d_n$$

$$E = d_1^2 + d_2^2 + d_3^2 + \dots\dots + d_n^2$$

$$E = |d_1|^2 + |d_2|^2 + |d_3|^2 + \dots\dots$$

Mod not use





$$E = d_1^2 + d_2^2 + d_3^2 + \dots\dots d_n^2$$

$$\boxed{E = \sum_{i=1}^{n} d_i^2}$$

$$d_i = (Y_i - \hat{Y}_i)$$

FORMERLY Industries Helping Hands

THE I iSCALE

Total Error

$$\boxed{E = \sum_{i=1}^{n} \left(y_i - \hat{y}_i \right)^2}$$

$\to (m, b)$

$$E_{Avg} = \frac{1}{n} \times \sum_{i=1}^{n} \left(y_i - \hat{y}_i \right)^2$$

$$\boxed{\hat{y}_i = m x_i + b}$$

$$E(m, b) = \sum_{i=1}^{n} \left(y_i - m x_i - b \right)^2$$

$b = 0$

$$E(m) = \sum_{i=1}^{n} \left(y_i - m x_i \right)^2$$

$$\frac{dE}{db} = \frac{d}{db}\sum_{i=1}^{n}\left(y_i - mx_i - b\right)^2 = 0$$

$$= \sum_{i=1}^{n}\left(y_i - mx_i - b\right)^2 = 0$$

$$= \sum_{i=1}^{n} -2\left(y_i - mx_i - b\right) = 0$$

$$= \boxed{\sum_{i=1}^{n}\left(y_i - mx_i - b\right) = 0}$$

$$\overbrace{\frac{\sum_{i=1}^{n} y_i}{n}}^{\bar{y}} - m\overbrace{\frac{\sum_{i=1}^{n} x_i}{n}}^{\bar{x}} - \overbrace{\frac{\sum_{i=1}^{n} b}{n}}^{b} = 0$$

FORMERLY Industries Helping Hands

THE I iSCALE

$$y - mx - \frac{\sum b}{\sum x} = 0$$

$$\sum b = \frac{b + b + b \ldots b}{n^{th}\ \text{time}}$$

$$\boxed{\bar{y} = m\bar{x} + b}$$

$$\boxed{b = \bar{y} - m\bar{x}}$$

Start Calculation for "m"

$$E = \sum_{i=1}^{n} (y_i - m x_i - b)^2$$

$$E = \sum_{i=1}^{n} (y_i - m x_i - \underbrace{\bar{y} + m\bar{x}}_{b \text{ value.}})^2$$

$$\frac{dE}{dm} = \sum_{i=1}^{n} \frac{d}{dm} (y_i - m x_i - \bar{y} + m\bar{x})^2 = 0$$

$$= \sum_{i=1}^{n} 2 (y_i - m x_i - \bar{y} + m\bar{x}) (-x_i + \bar{x}) = 0$$

$$= \sum_{i=1}^{n} -2 (y_i - m x_i - \bar{y} + m\bar{x}) (x_i - \bar{x}) = 0$$

$\div 2$

$$= \sum_{i=1}^{n} (y_i - m x_i - \bar{y} + m\bar{x}) (x_i - \bar{x}) = 0$$

$$= \sum_{i=1}^{m} \left[(y_i - \bar{y}) - m(x_i - \bar{x}) \right] \times (x_i - \bar{x}) = 0$$

$$= \sum_{i=1}^{m} \left[(y_i - \bar{y})(x_i - \bar{x}) - m(x_i - \bar{x})^2 \right] = 0$$

$$= \sum_{i=1}^{m} (y_i - \bar{y})(x_i - \bar{x}) = m(x_i - \bar{x})^2$$

$$\boxed{\frac{\sum_{i=1}^{m} (y_i - \bar{y})(x_i - \bar{x})}{(x_i - \bar{x})^2} = m}$$

FORMERLY Industries Helping Hands

THE I iSCALE

<Start Coding>

#Define Class

```
class MyLR:

  def __init__(self):
      self.m = None
      self.b = None

  def fit(self,X_train,y_train):
      pass
  def predict(self,X_test):
      pass
```

#Import Library

```
import numpy as np
import pandas as pd
```

#Import Dataset

```
df = pd.read_csv('placement.csv')
----
df.head()
```

#Extract X & Y

```
X = df.iloc[:,0].values
y = df.iloc[:,1].values
----
X
----
y
```

```
# Use SK_Learn X Train and X Test

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,random_state=2)

----

# Re-Edit Define Class

class MyLR:

    def __init__(self):
        self.m = None
        self.b = None

    def fit(self,X_train,y_train):
        print(X_train.shape)

    def predict(self,X_test):
        pass

----
X_train.shape

# Create Object "LR"

lr = MyLR()

----
lr.fit(X_train,y_train)

----
```

Calculate Numerator & Dominator for ″m″ & “b″

```
class MyLR:

    def __init__(self):
        self.m = None
        self.b = None

    def fit(self,X_train,y_train):

        num = 0
        den = 0

        for i in range(X_train.shape[0]):

            num = num + ((X_train[i] -
X_train.mean())*(y_train[i] - y_train.mean()))
            den = den + ((X_train[i] -
X_train.mean())*(X_train[i] - X_train.mean()))

        self.m = num/den
        self.b = y_train.mean() - (self.m *
X_train.mean())
        print(self.m)
        print(self.b)

    def predict(self,X_test):

        print(X_test)

        return self.m * X_test + self.b


----
 lr.fit(X_train,y_train)
```

```
# Predict the Value
print(lr.predict(X_test[0]))


print(lr.predict(X_test[1]))
```

Day 24 | Data Curious Minds

Suggest topic – Next class