Linear Regression

Short note about Linear Regression:

Linear regression refers to a model that can show relationship between two variables and how one can impact the other. In essence, it involves showing how the variation in the “dependent variable” can be captured by change in the “independent variables”

Decode Complex Algorithm

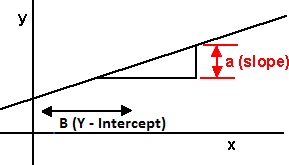
Y = ax + b

x,y are the coordinates of any point on the line

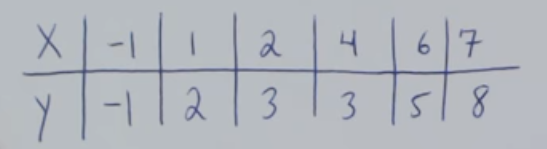
a is the slope of the line

b is the y-intercept (where the line crosses the y-axis)

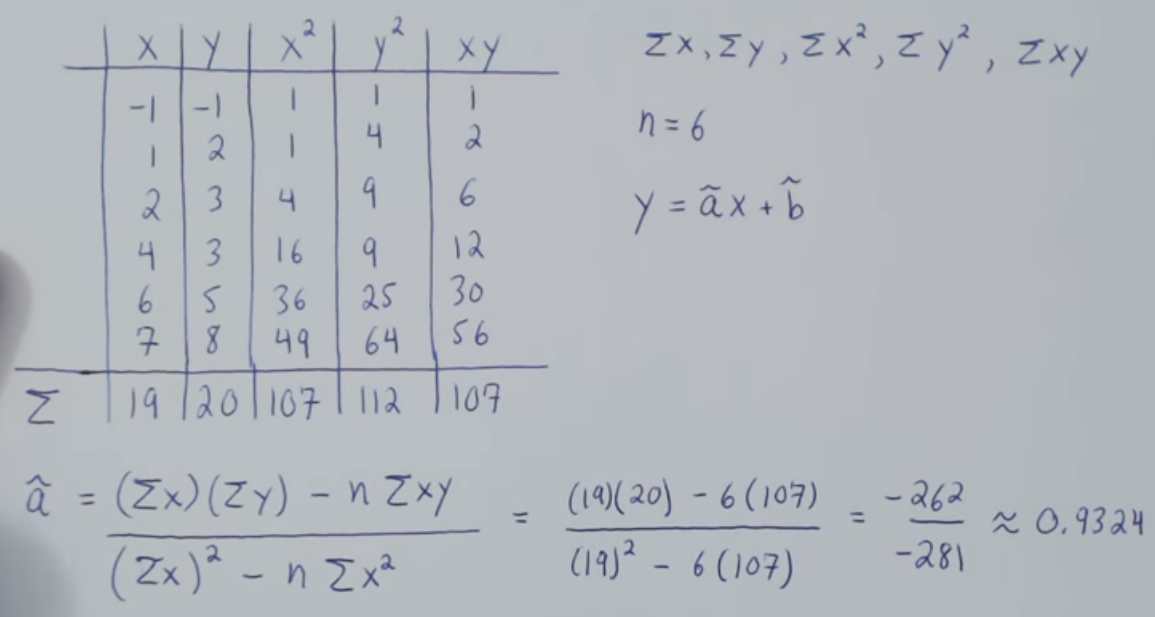
(i) IN GRAPH



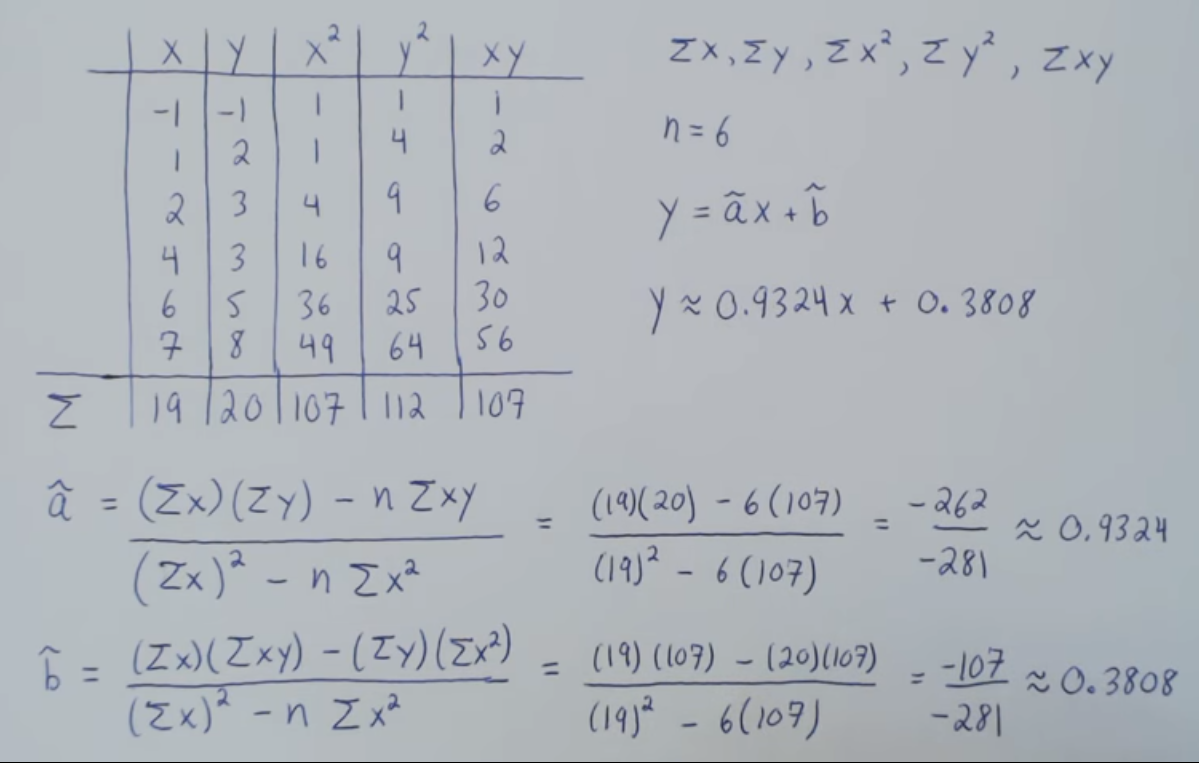
(ii) IN Calculation



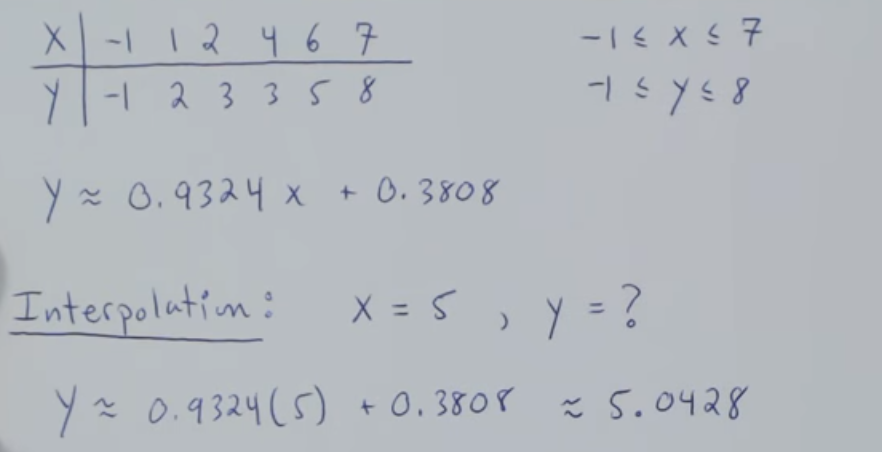
Step -1 [Finding Slope “a”]



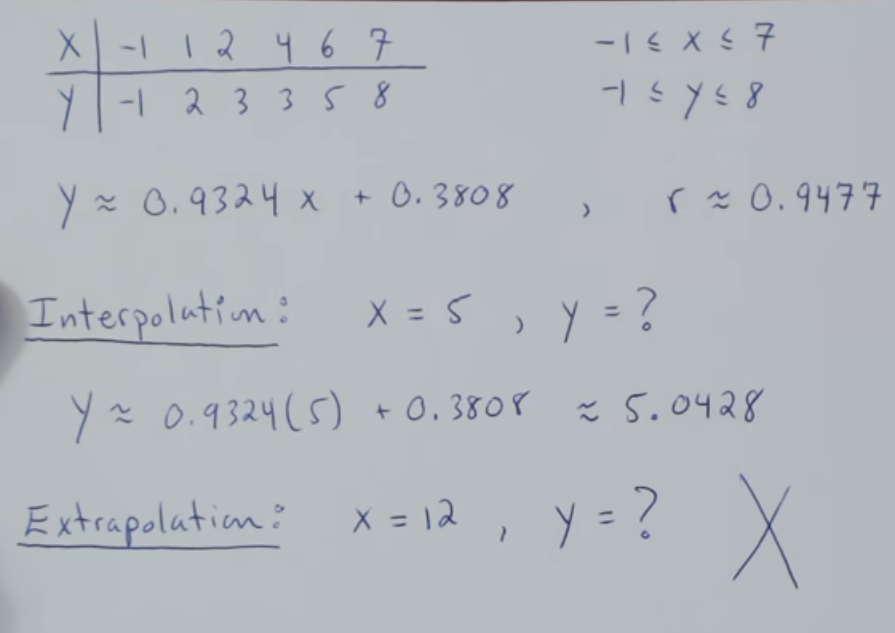
Step – 2 [Finding Y – Intercept “B”]



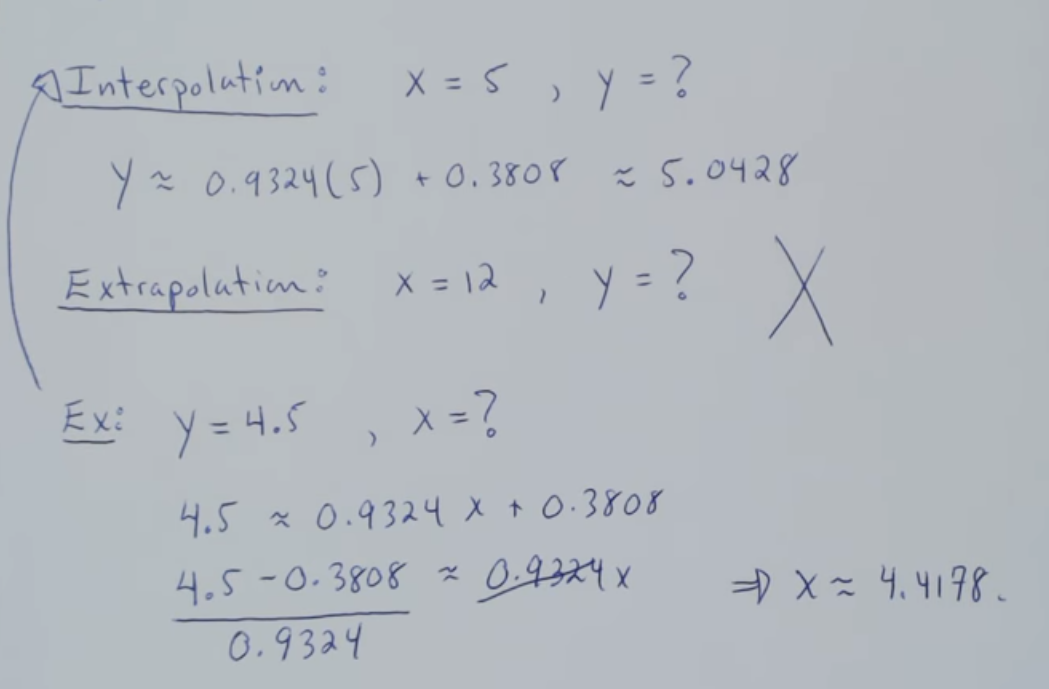
Step 3 – Finding X Interpolation



Step 4 – Finding Y Extrapolation is not possible



Step 5 Finding Interpolation but using backward



USE CASES OF Linear Regression

* FINDING MISSING VALUES WITHIN THE DATASET
* FINDING THE LOSS VALUE BY SCATTER POINT & SLOPE LINE

For Example

Finding Salary with years of experience

Finding Advertising cost with Sales

Finding Sales with financial years

Finding height with body weight

Finding units demand with price

Finding quantity with yield

Determining the poverty reasons with census data with respect to a various parameters which might be a reason.

Determining the land / house price with respect to a certain parameter.

Speed and distance relationship

Estimation of revenue to its expenditure

Finding rate of growth of the economy of a institution

PYTHON CODE USING LIBRARY

LINEAR REGRESSION

# Importing Library

import numpy as np

import matplotlib.pyplot as plt

import pandas as pd

# Importing the dataset

dataset = pd.read\_csv('Salary.csv')

X = dataset.iloc[:, :-1].values

y = dataset.iloc[:, 1].values

# Splitting the dataset into the Training set and Test set

from sklearn.cross\_validation import train\_test\_split X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size = 1/3, random\_state = 0)

# Fitting Single Linear Regression to the Training set

from sklearn.linear\_model import LinearRegression

regressor = LinearRegression()

regressor.fit(X\_train, y\_train)

# Predicting Values

y\_pred = regressor.predict(X\_test)

# Visualization

plt.scatter(X\_test, y\_test, color = 'red')

plt.plot(X\_train, regressor.predict(X\_train), color = 'blue')

plt.title('Salary vs Experience (Test set)')

plt.xlabel('Years of Experience')

plt.ylabel('Salary')

plt.show()

PYTHON CODE WITHOUT LIBRARY

import pandas as pd

import os

import numpy as np

import matplotlib.pyplot as plt

df = pd.read\_csv("data.csv",sep='\t')

print(df.head(6))

X=df['PovPct'].values

Y=df['Brth15to17'].values

plt.scatter(X,Y,c='red',label='Scatter Plot')

plt.xlabel('Poverty Range')

plt.ylabel('Birth 15 - 17')

plt.legend()

plt.show()

# mean not required

n=len(X)

# X.reshape(n,-1)

# Y.reshape(n,-1)

sumX=np.sum(X)

sumY=np.sum(Y)

mulXY=np.multiply(X,Y)

sum\_mul\_XY=np.sum(mulXY)

Xsquare = np.multiply(X,X)

Ysquare = np.multiply(Y,Y)

sum\_X\_square=np.sum(Xsquare)

a\_hat= ((sumX\*sumY)-(n\*sum\_mul\_XY))/((sumX\*sumX)-(n\*sum\_X\_square))

b\_hat= ((sumX\*sum\_mul\_XY)-(sumY\*sum\_X\_square))/((sumX\*sumX)-(n\*sum\_X\_square))

print(a\_hat)

print(b\_hat)

updated\_Y= a\_hat\*X + b\_hat

plt.plot(X, updated\_Y, color='red', label='Regression Line')

plt.scatter(X, Y, c='blue', label='Actual data')

plt.xlabel('Poverty Rate')

plt.ylabel('15-17 yo Birth rate/100')

plt.legend()

plot1 = plt.show()

VISUALIZATIONS

