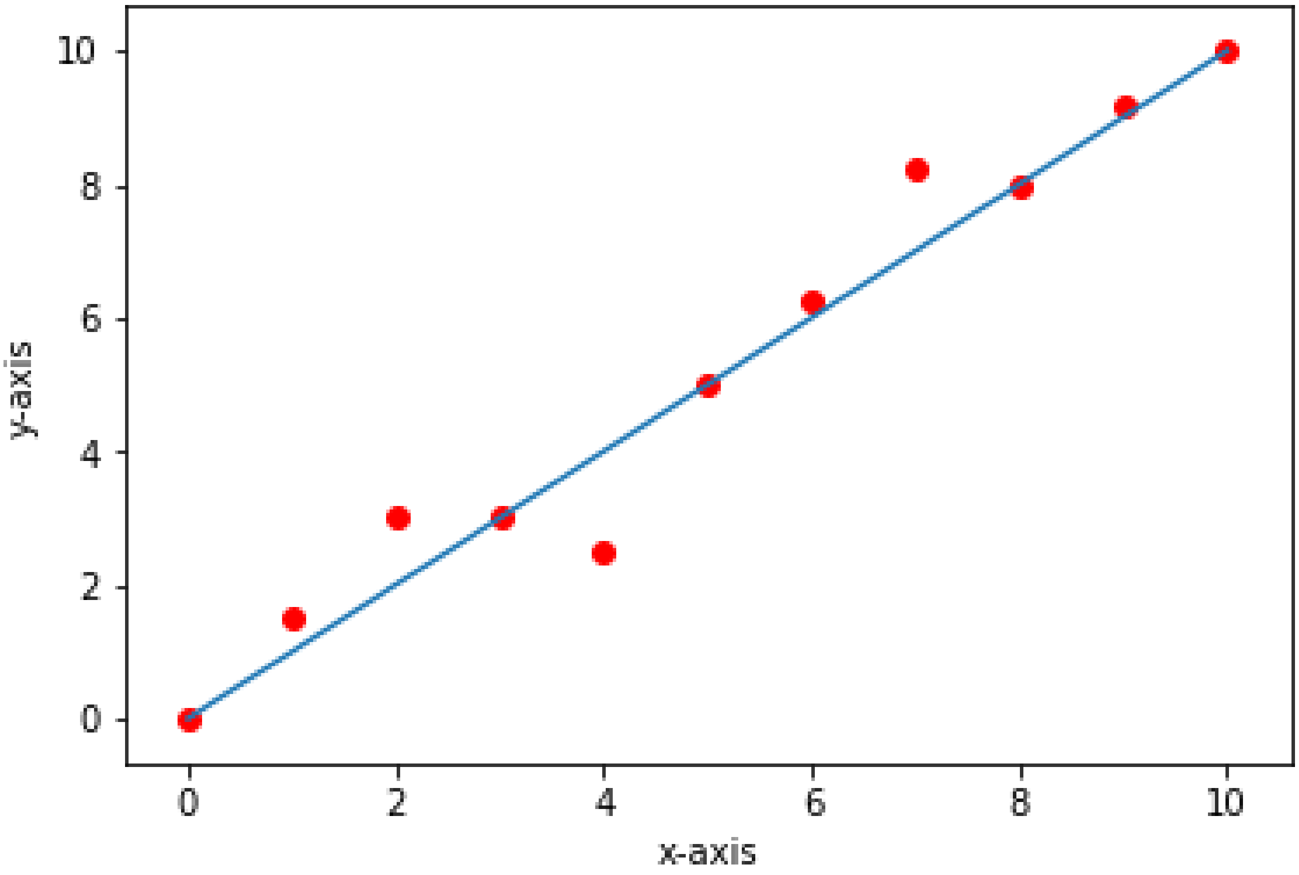
Linear Regression from Scratch Statistical Approach



The Theory

Linear Regression is the process of fitting a line to the dataset.

Single Variable Linear Regression

The Mathematics

The equation of Line is

$y = m \cdot x + c$

Where,

y = dependent variable

X = independent variable

C = intercept

The algorithm is trying to fit a line to the data by adjusting the values of m and c. Its Objective is to attain to a value of m such that for any given value of x it would be properly predicting the value of y.

There are various ways in which we can attain the values of m and c

1. Statistical approach
2. Iterative approach

In this post we are discussing Statistical approach. If you find the derivation too long you can take look ar final m and c formulas.

We were given data, set of x and y values and we were asked to find a line which best fits that which means mathematically we should be able to fine the slope and c values of the line to describe the line.

The derivation to the slope and intercept values of the line

The equation of the Line is

$$y = mx * c$$

Which means for a particular value of x, let us say x_i the value of y would be $y_i = m*x_i + c$

For x_1 value of x y_1 value of y is obtained,

For x_2 value of x y_2 value of y is obtained,

And goes on

On summating all these values into one equation, it can be written as,

$$\sum_{i=0}^{n} c + \sum_{i=0}^{n} x_i * m = \sum_{i=0}^{n} y_i$$

$$c * \sum_{i=0}^{n} x_i + \sum_{i=0}^{n} x_i^2 * m = \sum_{i=0}^{n} y_i x_i$$

The objective is to solve equation 1 and 2 to attain the values of c and m

On solving the c term in Equation 1, it can also be written as,

$$n * c + \sum_{i=0}^{n} x_i * m = \sum_{i=0}^{n} y_i$$

Rewriting it as,

$$m = \frac{\sum_{i=0}^{n} y_i - n * c}{\sum_{i=0}^{n} x_i}$$

Substituting the value of m in Equation 2

$$c * \sum_{i=0}^{n} x_i + \sum_{i=0}^{n} x_i^2 * \left(\frac{\sum_{i=0}^{n} y_i - n * c}{\sum_{i=0}^{n} x_i}\right) = \sum_{i=0}^{n} y_i x_i$$

Now only c is there in Equation 3, solving for c

Step1,

$$c * \left(\sum_{i=0}^{n} x_i - \frac{n * \sum_{i=0}^{n} x_i^2}{\sum_{i=0}^{n} x_i}\right) + \frac{\sum_{i=0}^{n} y_i * \sum_{i=0}^{n} x_i^2}{\sum_{i=0}^{n} x_i} = \sum_{i=0}^{n} y_i x_i$$

Step2,

$$c * \left(\left(\sum_{i=0}^{n} x_i\right)^2 - n * \sum_{i=0}^{n} x_i^2\right) + \sum_{i=0}^{n} y_i * \sum_{i=0}^{n} x_i^2 = \sum_{i=0}^{n} y_i x_i * \sum_{i=0}^{n} x_i$$

Step 3,

$$c = \frac{\sum_{i=0}^{n} y_i x_i * \sum_{i=0}^{n} x_i - \sum_{i=0}^{n} y_i * \sum_{i=0}^{n} x_i^2}{\left(\sum_{i=0}^{n} x_i\right)^2 - n * \sum_{i=0}^{n} x_i^2}$$

On multiplying both numerator and denominator by -1

$$c = \frac{\sum_{i=0}^{n} y_i * \sum_{i=0}^{n} x_i^2 - \sum_{i=0}^{n} y_i x_i * \sum_{i=0}^{n} x_i}{n * \sum_{i=0}^{n} x_i^2 - \left(\sum_{i=0}^{n} x_i\right)^2}$$

To calculate the value of m taking the value of c and substituting in Equation -1,

$$n * \left(\frac{\sum_{i=0}^{n} y_i * \sum_{i=0}^{n} x_i^2 - \sum_{i=0}^{n} y_i x_i * \sum_{i=0}^{n} x_i}{n * \sum_{i=0}^{n} x_i^2 - \left(\sum_{i=0}^{n} x_i\right)^2}\right) + m * \sum_{i=0}^{n} x_i = \sum_{i=0}^{n} y_i$$

Equation 4 have only one parameter m, so solving for m,

$$m * \sum_{i=0}^{n} x_i * \left(n * \sum_{i=0}^{n} x_i^2 - \left(\sum_{i=0}^{n} x_i\right)^2\right) = \sum_{i=0}^{n} y_i * \left(n * \sum_{i=0}^{n} x_i^2 - \left(\sum_{i=0}^{n} x_i\right)^2\right) - n * \left(\sum_{i=0}^{n} y_i * \sum_{i=0}^{n} x_i^2 - \sum_{i=0}^{n} y_i x_i * \sum_{i=0}^{n} x_i\right)$$

On expanding RHS and cancelling terms,

$$m * \sum_{i=0}^{n} x_i * \left(n * \sum_{i=0}^{n} x_i^2 - \left(\sum_{i=0}^{n} x_i\right)^2\right) = n * \sum_{i=0}^{n} x_i * \sum_{i=0}^{n} y_i x_i - \left(\sum_{i=0}^{n} x_i\right)^2 * \sum_{i=0}^{n} y_i$$

The value of m,

$$m = \frac{n * \sum_{i=0}^{n} y_i x_i - \sum_{i=0}^{n} x_i * \sum_{i=0}^{n} y_i}{n * \sum_{i=0}^{n} x_i^2 - \left(\sum_{i=0}^{n} x_i\right)^2}$$

The values of c and m are,

$$m = \frac{n * \sum_{i=0}^{n} y_i x_i - \sum_{i=0}^{n} x_i * \sum_{i=0}^{n} y_i}{n * \sum_{i=0}^{n} x_i^2 - \left(\sum_{i=0}^{n} x_i\right)^2}$$

$$c = \frac{\sum_{i=0}^{n} y_i * \sum_{i=0}^{n} x_i^2 - \sum_{i=0}^{n} y_i x_i * \sum_{i=0}^{n} x_i}{n * \sum_{i=0}^{n} x_i^2 - \left(\sum_{i=0}^{n} x_i\right)^2}$$

We use the m and c formulas obtained in derivation in the code.

The Dataset

Dataset consists of two columns namely X and y

Where

For List Price Vs. Best Price for a New GMC Pickup dataset

X = List price (in $1000) for a GMC pickup truck

Y = Best price (in $1000) for a GMC pickup truck

The data is taken from *Consumer's Digest.*

For Fire and Theft in Chicago

X = fires per 100 housing units

Y = thefts per 1000 population within the same Zip code in the Chicago metro area

The data is taken from U.S Commission of Civil Rights.

For Auto Insurance in Sweden dataset

X = number of claims

Y = total payment for all the claims in thousands of Swedish Kronor

The data is taken from Swedish Committee on Analysis of Risk Premium in Motor Insurance.

For Gray Kangaroos dataset

X = nasal length (mm ¥10)

Y = nasal width (mm ¥ 10)
for a male gray kangaroo from a random sample of such animals

The data is taken from Australian *Journal of Zoology*, Vol. 28, p607-613.

Link to All Datasets

The Code

The Code was written in three phases

1. Data preprocessing phase
2. Training
3. Prediction and plotting

The data preprocessing phase

Imports

Numpy import for array processing, python doesn't have built in array support. The feature of working with native arrays can be used in python with the help of numpy library.

Pandas is a library of python used for working with tables, on importing the data, mostly data will be of table format, for ease manipulation of tables pandas library is imported

Matplotlib is a library of python used to plot graphs, for the purpose of visualizing the results we would be plotting the results with the help of matplotlib library.

Math library is import for calculating powers of numbers.

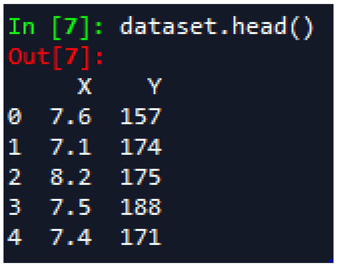
```
# Imports
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import math
```

Reading the dataset from data

In this line of code using the read_excel method of pandas library, the dataset has been imported from data folder and stored in dataset variable.

```
# Reading the dataset from data
dataset = pd.read_csv(r'..\\data\\auto_insurance.csv')
```

On viewing the dataset, it contains of two columns X and Y where X is dependent variable and Y is Independent Variable



```
In [7]: dataset.head()
Out[7]:
     X    Y
0  7.6  157
1  7.1  174
2  8.2  175
3  7.5  188
4  7.4  171
```

X is an independent variable

Y is dependent variable Inference

For x-value of 7.6 ,157 y-value

for   x-value of 7.1 ,174 y-value

And goes on

Creating Dependent and Independent variables

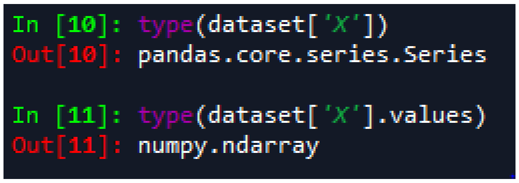
The X Column from the dataset is extracted into an X variable of type numpy, similarly the y variable

X is an independent variable

Y is dependent variable Inference

```
# Creating Dependent and Independent variables
```

```
X = dataset['X'].values
y = dataset['Y'].values
```



```
In [10]: type(dataset['X'])
Out[10]: pandas.core.series.Series

In [11]: type(dataset['X'].values)
Out[11]: numpy.ndarray
```

On input 10 it would result in a pandas Series object

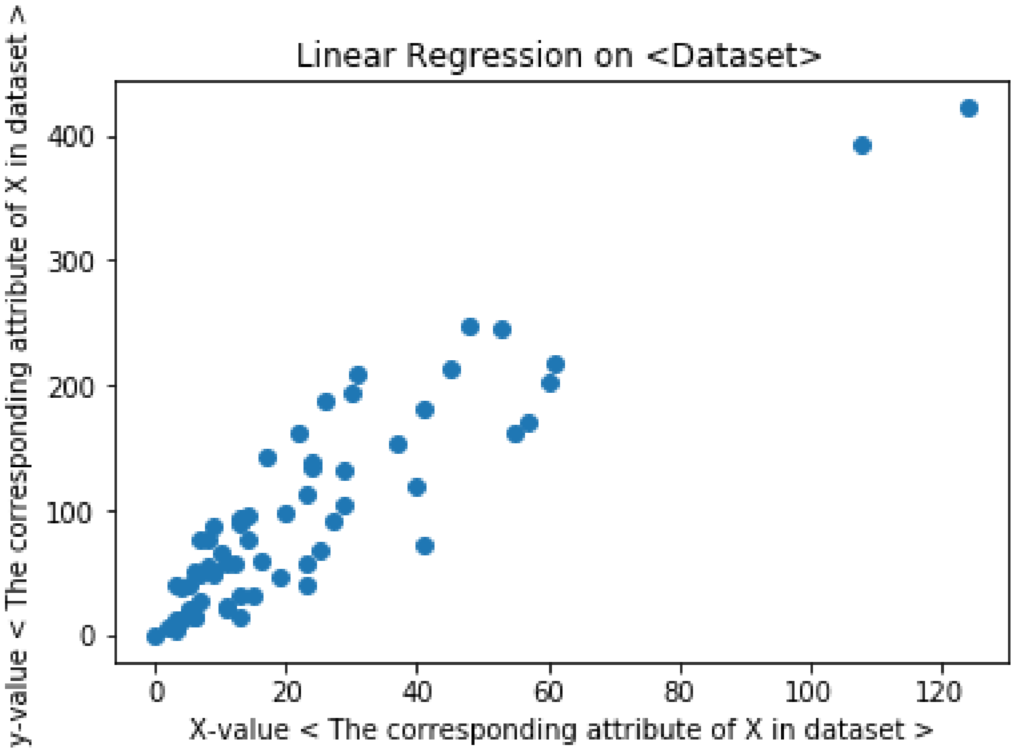
So, values attribute is used to attain an numpy array

Visualizing the data

The step is to just see how the dataset is

On visualization the data would appear something like this

```
# Visualizing the data
title='Linear Regression on <Dataset>'
x_axis_label = 'X-value < The corresponding attribute of X in dataset >'
y_axis_label = 'y-value < The corresponding attribute of X in dataset >'
plt.scatter(X,y)
plt.title(title)
plt.xlabel(x_axis_label)
plt.ylabel(y_axis_label)
plt.show()
```



Each point on the plot is a data point showing the respective list price on x-axis and Best Price on y-axis

The X and Y attributes would vary based on dataset.

Splitting the data into training set and test set

We are splitting the whole dataset into training and test set where training set is used for fitting the line to data and test set is used to check how good the line if for the data.

```
# Splitting the data into training set and test set
X_train,X_test = np.split(X,indices_or_sections = [int(len(X)*0.2)])
y_train,y_test = np.split(y,indices_or_sections = [int(len(X)*0.2)])
```

The Training phase

Computing the values of sigma

As per the derivation formula we are computing the values of sigma x sigma x^2 sigma y sigma x*y

n is the number of terms in the dataset.

```
# Computing the values of sigma
sigma_X = sum(X_train)
sigma_y = sum(y_train)
sigma_xy = sum(np.multiply(X_train,y_train))
sigma_X_square = sum(np.square(X_train))
n = len(X_train)
```

Computing the values of Slope and Intercept

As our linear regression model requires a slope and intercept, we are computing their values using statistical formulas.

```
# Computing the values of slope and intercept
m_numerator = (n*sigma_xy)-(sigma_X*sigma_y)
m_denominator =  n*sigma_X_square - math.pow(sigma_X,2)
m = m_numerator/m_denominator

c_numerator = (sigma_y*sigma_X_square)-(sigma_xy*sigma_X)
c_denominator = (n*sigma_X_square) - math.pow(sigma_X,2)
c = c_numerator/c_denominator
```

The Prediction phase

Predicting the Results

By the knowing the slope and intercept values of linear regression model we are trying to predict the values of test data. Y_pred variable contains all the predicted y-values of the test x-values.

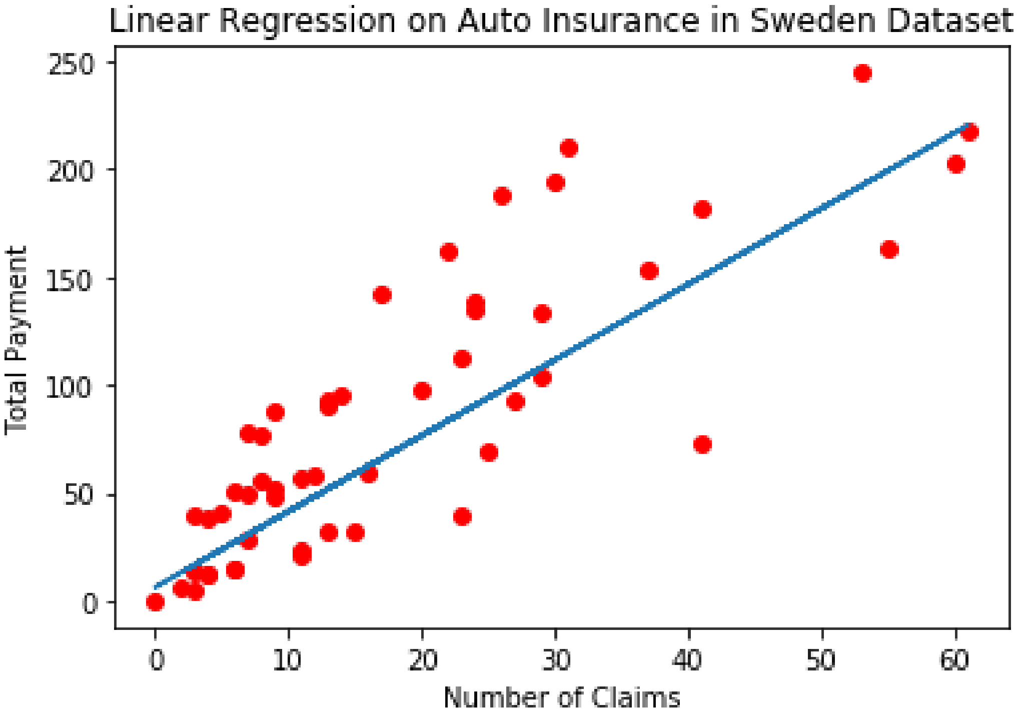
```
# Predicting the Results
y_pred = X_test*m + c
```

Visualizing the Results

As we have predicted the y-values for a set of x-values we are visualizing the results to check how good did our line fit for our predictions.

The plot shows the red points are the data points are actual values where the blue line is the predictions.

```
# Visualizing the Results
plt.scatter(X_test,y_test,c='red')
plt.plot(X_test,y_pred)
plt.title(title)
plt.xlabel(x_axis_label)
plt.ylabel(y_axis_label)
plt.show()
```


Linear Regression on Auto Insurance in Sweden Dataset
Total Payment
0
50
100
150
200
250
0
10
20
30
40
50
60
Number of Claims