Winter of Code

ML Bootcamp

Project Report

The main objective was to implement Linear regression, Logistic regression, KNN and Neural Network from scratch.

All the codes of models implemented has been pushed in the GitHub repository.

All the work has been done in the Google Colaboratory notebook, links present in the respective sections.

GitHub repo link:
https://github.com/Aman-krishna?tab=repositories

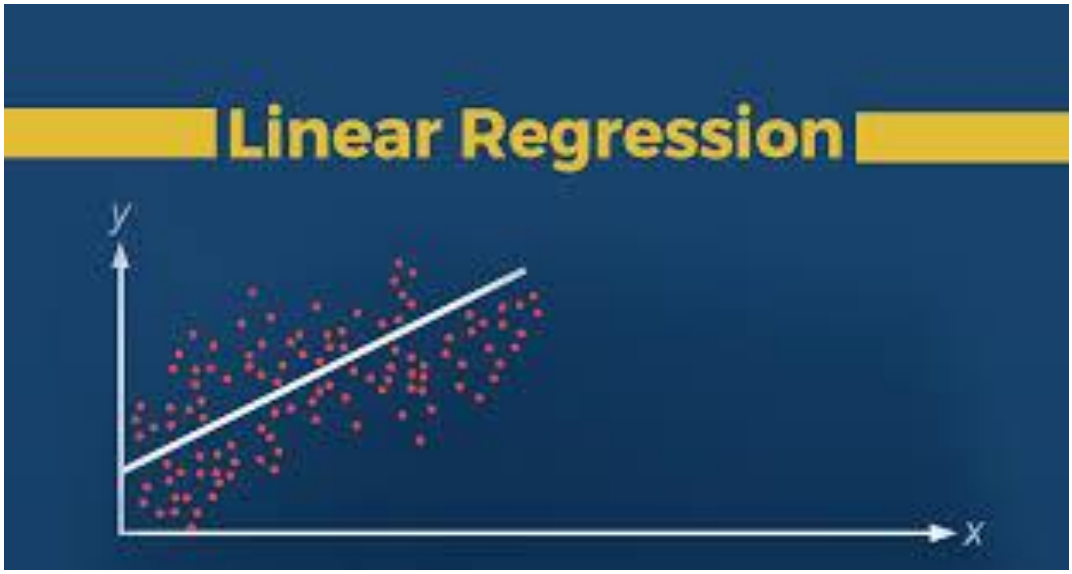
Linear Regression:

Google Colab notebook link:
https://colab.research.google.com/drive/10z7i1s5AJtGUGpUX3LVlA-vp_w2loMUd

The basic idea was to discover a linear equation which when trained by the provided data gives the minimum possible error when executed and used to predict values.

The linear equation is formed by finding appropriate values of the slope and the intercept, which are sufficient for writing any equation of line.

Visual representation of the concept:



The appropriate value of slope and intercept is found by minimizing the error value obtained by the error function, mean squared error function to be specific, with the concepts of gradient and gradient descent. We can adjust the learning rate and number of iterations as per our need keeping in mind that the model's error ebbs with number of iterations but with the cost of execution time.

Once the required values are found, the equation is completed and we can implement the model. We can also plot the line and dataset for a better understanding.

In implementing I have used pandas, NumPy and matplotlib for accessing and reading the file, mathematical calculations and plotting graphs (wherever required), respectively.

Logistic Regression:

Google Colab notebook link:
https://colab.research.google.com/drive/1v3wSqTVvqDsrGSg12kV82FEariyStoqK

In implementing the logistic regression model, firstly I have defined the basic sigmoid function. The predicted value i.e., the value of Y should lie between 0 and 1, so for that sigmoid function is used.

Then I have defined the error function for the model. After that just like I did in linear regression, similarly, here also I have defined and used the gradient and gradient descent function and updated the variables assumed in the model after each iteration until the error obtained is minimum.

After discovering the required variables our model is complete and ready to be implemented.

KNN:

Google Colab notebook link:
https://colab.research.google.com/drive/1inJ5metdYeLgtF4eHr32aEEXTV5e7qiD

KNN model involves finding distance between the target point and the elements provided in the dataset. Arranging the distance so obtained in a structured and organized way and finding the elements of the dataset from which the target point is nearer and thus classifying that point according to the class or characteristic obtained.

In my KNN model firstly I have defined the distance function, it finds the Euclidian distance between two concerned points.

After that I created an empty list and appended the values of distance obtained from the distance function in it. I have arranged the values in an order and then converted the list into array and added another row to append the value of Ys from

the dataset. I have tried to add unique values in the list and mention the number of times it has occurred.

By indexing, I tried to identify and bring out the elements representing characteristic or class of the dataset as output.

By doing this we can predict of which class(characteristic) the target element belongs and the accuracy can by reported by the functions we use.

Neural Network:

Google Colab notebook link:
https://colab.research.google.com/drive/1W7TNMTIJMJegCy0EEZQKNitHET58NebX

In the neural network model implementation, I tried a simple neural network including the backpropagation process.

I first defined the basic equation by which the predicted value would come. It was a sigmoid function with the variables comprising of a bias, which is an unknown value that gets updated with every backpropagating step.

The equation is like: $Z = \text{sum}(k_i X_i) + \text{bias}$

After this I have defined the error function and used the gradient function, as I did in implementation of previous models. The bias term was getting updated in a similar way as the variables used to get updated, by using appropriate number of iterations and learning rate.

After all the process of minimizing error and finding of the bias, the model is ready to be implemented.

By:

Aman Krishna
20JE0105