Implimentation of Linear Regression

Lab Assignment 2

Vedant Mohan Phuse
Dept. of Electronics and Communication Engineering
Visvesvaraya National Institute of Technology
Nagpur, India
vedant.phuse@hotmail.com

***Abstract*—Linear Regression analysis is used to predict the value of a variable based on the value of another variable. This curve fitting ability of linear regression is useful in data interpolation, prediction and forecasting.**

In this work we tried to use analytical solution to fit a curve to the Matlab_accident Dataset and compared its performance with learning based linear regression with different capacities or degrees of freedom. We also tried to compare the performance against the some benchmark parameters like time complexity and spacial complexity and computation efficiency and the detail results are presented at the end of this report.

***Index Terms*—mathworks's us accidents dataset, linear regression, python, pseudo inverse, gradient descent**

I. Introduction

Artificial Neural Networks [1] are the Computational Networks, that are created on the basis of human neurons in the brain to perform computations and mimic the brain activity.

A Neural Network [2] consists of Neurons as the fundamental computational unit, and the data sent between these connections resembles the synapses in the biological brain. These neural networks, also just called as Neural Nets are capable of taking inputs from the users and predicting the output as expected for the given data points. These neural networks can go from the most basic Artificial Neural Networks, and keep getting complicated and complicated by following the likes of Convolutional Neural Networks. [3], Recurrent Neural Networks [4], Transformer Networks [5] and so on.

This experiment serves as the basis for all the machine learning algorithms by employing the most basic Linear Regression [6] models and respective gradient descent algorithms by creating a problem statement based on the US Traffic Accidents Dataset and predicting one of the fields, by using the train-test split function from previous lab assignment.

II. Method

In this section, we will explain the individual steps involved in fitting a curve using both least square approach as well as learning based approach:

A. Data Preprocessing

Data processing is one of the most essential step in any machine learning algorithm.

The data provided to us was in the form of a .mat file, which can be read in python or jupyter notebooks by the use of loadmat functionality provided in the scipy.io module. This returns us a dictionary of all the different of individual example present in the mat file.

So to check the generalisability of any ml model we split the data into train and test data which acts like a guide to evaluate the training process. So this huge dictionary has was parsed, and we can observe the datasets in the form of multiple numpy arrays. These numpy arrays are processed, the headers are applied, and the combined array is converted into a pandas dataframe, which can be splited on to train-test data by us with the help of the train test split function that was defined by us in the last assignment.

B. Pseudo Inverse Method

The pseudo inverse method is a least square approach which is used to calculate the optimal curve parameters, which minimizes the square error between all the points available in dataset. it is given by,

$$w = \underbrace{(X^T}_{(dxN)}\underbrace{X)^{-1}}_{(Nxd)}\underbrace{X^T}_{(dxN)}\underbrace{y}_{(Nx1)}$$

we implemented pseudo inverse method and the results are shown in the above figure.

C. Linear Regression using Gradient Decent Algorithms

Pseudo inverse gives the best solution which perfectly fits the data but the spacial computation cost of for calculating the parameter is the large and it fails if the $X^T X$ is singular. so to fine optimal solution of such degenerate cases gradient decent based methods or learning based mehtods are used.

So to find out the optimal parameters of the curve, we use the basic machine learning algorithm framework of data processing, method defining, loss function selection, training and finally evalution. the results for different hyper-parameter setting are discussed in the upcoming section.

III. Experiment

Any procedure involving the training of an Artificial Neural Network Model has the standard procedure.

First, the data is gathered and then it is cleaned such that we are able to process it using the standard python and machine learning libraries. The data here is in the form of .mat file,

which has multiple datasets of different shapes, that can be read with the help of the scipy.io library as numpy arrays.

As mension in the methods section, we preprocessed data and using the pseudo inverse formula we calculated analylitical solution. we experiemented with different hyperparameter like learning rate of 0.001,0.01 and different number of iterations like 1000, 10000.

we also experimented with different order of input features like tried to fit a quadratic curve for the same data.

IV. Results

The plots of the loss function vs number of iterations, and the best fit of the model are obtained from the code as per the question, and are displayed as follows:

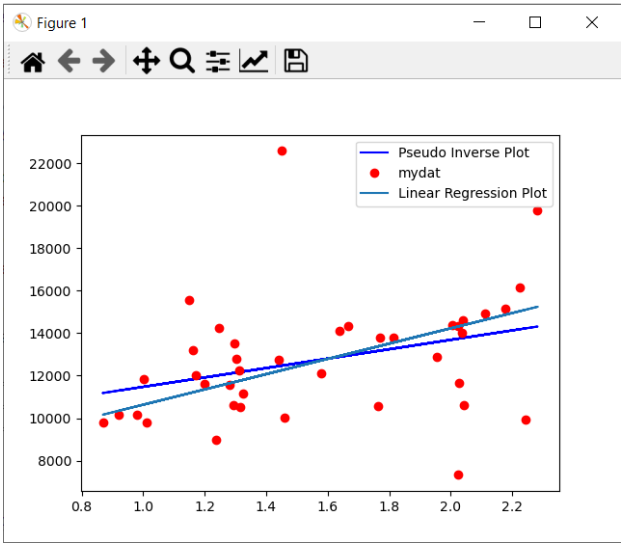

Fig. 1. a1

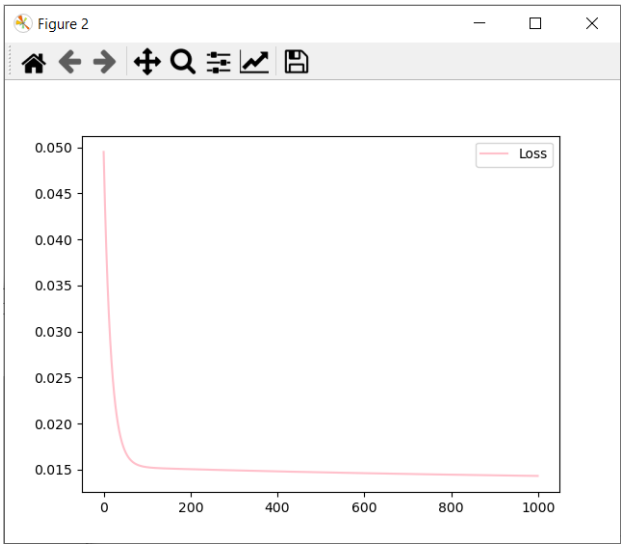

Fig. 2. a2

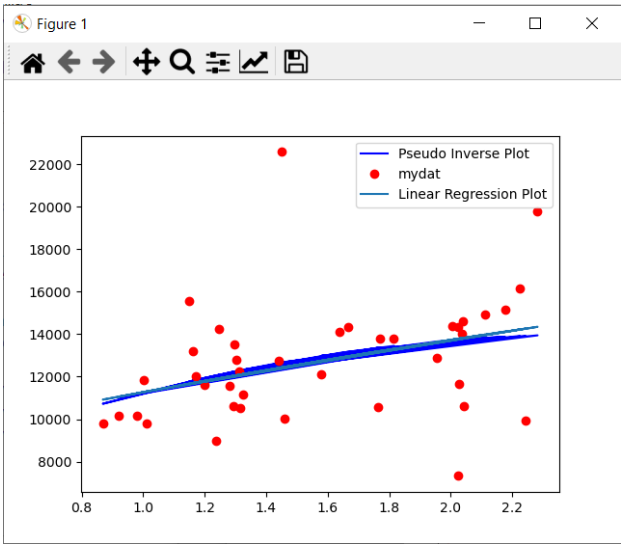

Fig. 3. b1

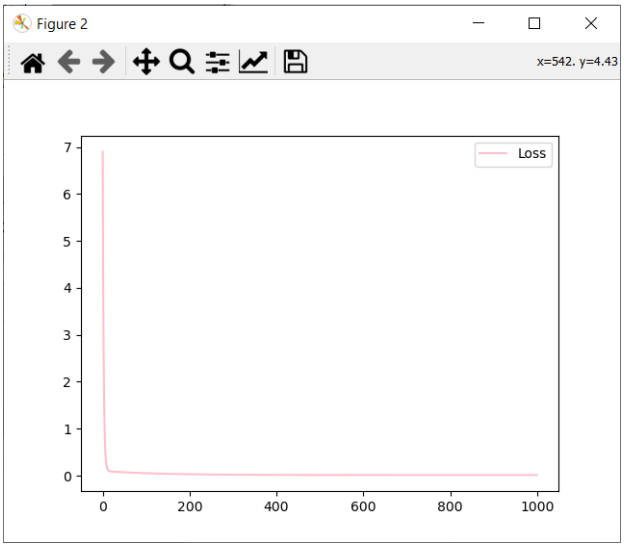

Fig. 4. b2

V. Discussion

It was observed that to train a network using gradient decent we have to normalize the input data between 0 to 1 so that optimization of parameter becomes easy.

It wasa also observed that with increse in the number of the iterations we the gradient decent solution reached close to the analytical solution as observed in the result sectio

VI. Conclusion

Linear Regression has been implemented on the given US Accidents dataset and has been studied by applying for different parameters and different methods.

References

[1] B. Yegnanarayana, *Artificial neural networks*. PHI Learning Pvt. Ltd., 2009.

[2] J. A. Anderson, *An introduction to neural networks*. MIT press, 1995.
[3] J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, T. Liu, X. Wang, G. Wang, J. Cai *et al.*, "Recent advances in convolutional neural networks," *Pattern recognition*, vol. 77, pp. 354–377, 2018.
[4] L. R. Medsker and L. Jain, "Recurrent neural networks," *Design and Applications*, vol. 5, pp. 64–67, 2001.
[5] M. Jaderberg, K. Simonyan, A. Zisserman *et al.*, "Spatial transformer networks," *Advances in neural information processing systems*, vol. 28, 2015.
[6] G. A. Seber and A. J. Lee, *Linear regression analysis*. John Wiley & Sons, 2012.

APPENDIX (CODE)

The code used for the following implementation is hosted on this link:

https://hackmd.io/@vedantphuse/MLexp2

Link to Code