Implementation of Linear Regression

Lab Assignment 2

Sumedha Janani Siriyapuraju
Dept. of Electronics and Communication Engineering
Visvesvaraya National Institute of Technology
Nagpur, India
sumedhasjs@gmail.com

***Abstract*—Linear regression is the most commonly used method of predictive analysis. It uses linear relationships between a dependent variable (target) and one or more independent variables (predictors) to predict the future of the target. The prediction is based on the assumption that the relationship between the target and the predictors is dependent or causal.**

Here, we worked on finding an analytical solution to fit a curve for Matlab_accident Dataset.

***Index Terms*—Liner Regression, Pseudo Inverse, Gradient Descent, Learning Rate, Optimum Weights**

I. Introduction

Linear regression analysis is used to predict the value of a variable based on the value of another variable. The variable you want to predict is called the dependent variable. The variable you are using to predict the other variable's value is called the independent variable.

This form of analysis estimates the coefficients of the linear equation, involving one or more independent variables that best predict the value of the dependent variable. Linear regression fits a straight line or surface that minimizes the discrepancies between predicted and actual output values. There are simple linear regression calculators that use a "least squares" method to discover the best-fit line for a set of paired data. We then estimate the value of X (dependent variable) from Y (independent variable).

II. Method

A. Data Pre-processing

The given file was loaded and data and headers were extracted. The raw data was converted into a dataframe using initial preprocessing. The data was observed and a problem statement was created. The dependent variable was traffic fatalities and independent variable was number of licensed drivers. The entire dataframe was then divided into 2 groups: training and testing dataset in the ratio of 60:40.

B. Pseudo Inverse Method

In this, the weights are calculated using the formula W= $(X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}Y$. Where X is the array of independent variables and Y is the independent variable. The size of weight array = number of independent variables, hence 1. This obtained weight was used to find predicted value of dependent variable according to the formula $Y = W^{\mathrm{T}}X$ The output values from this equation was stored as Y-predicted. These values (Y-predicted) were then compared with the actual values of the target variable stored in testing dataset. Accuracy is checked by calculating the mean square error between predicted and actual value of dependent variable

C. Linear Regression using Gradient Decent Algorithms

The initial weight values: m and c were calculated using pseudo-inverse method. The learning rate and number of epochs were taken as 0.01 and 100 respectively. For each epoch iteration, an initial prediction of target variable is done using the formula $Y = W^{\mathrm{T}}X$. The derivative with respect to both m and c were calculated for each iteration (Dm and Dc respectively). Next, the values of these weights were updated using the learning rate and the derivatives in accordance with the formula $m = m - L * Dm$ and $c = c - L * Dc$ where L is the learning rate. Once the iterations are over, we get the optimised weights or m and c. These values are then plugged into the test dataset and the predicted values of target variable are found out. The graph of the test data and the predicted data is plotted to see the linear regression.

III. Experiment

The standard technique applies to any procedure requiring the training of an artificial neural network model.

The data is first acquired, and then it is cleaned such that we can process it using the standard machine learning and Python tools. This data is in the form of a.mat file that contains many datasets of various shapes and can be read as NumPy arrays with the help of the scipy.io tool.

As mentioned in the methods section, we preprocesessed data and derived the analytical solution using the pseudo-inverse formula. We tested with several hyper-parameters, such as a learning rate of 0.001,0.01, and various iteration counts, such as 1000, 10000.

IV. Results

	States	Census ID	Centroid Longitude	Centroid Latitude	Traffic Fatalities	Licensed drivers (thousands)	Registered vehicles (thousands)	Vehicle-miles traveled (millions)	Miles traveled per vehicle	Fatalities per 100K licensed drivers	Fatalities per 100K registered vehicles	Fatalities per 100M vehicle-miles traveled	Fatalities involving high blood alcohol	Percent Alcohol-Related	Total Population	Urb Populati
0	Wyoming	56.0	-107.555925	43.032664	164.0	380.180	671.529	9261.0	13700.915954	43.137461	24.421879	1.770867	54.0	32.926829	493782.0	32207
1	District of Columbia	11.0	-77.026928	38.892064	43.0	349.122	240.403	3742.0	15565.529548	12.316611	17.886632	1.149118	12.0	27.906977	572059.0	57205
2	Vermont	50.0	-72.556496	44.043498	98.0	550.462	551.516	7605.0	14242.560506	17.803227	17.769203	1.247613	20.0	20.408163	608827.0	23255
3	Alaska	2.0	-152.459342	64.609656	101.0	482.532	681.115	4960.0	7326.222444	20.931254	14.828627	2.024048	30.0	29.702970	626932.0	41195
4	North Dakota	38.0	-99.499793	47.469117	100.0	461.780	721.835	7594.0	10520.409789	21.655334	13.853581	1.316829	35.0	35.000000	642200.0	35839

Fig. 1. Data Preprocessing - Initial Dataframe

	Traffic fatalities	Licensed drivers (thousands)	Registered vehicles (thousands)	Vehicle-miles traveled (millions)
0	164.0	380.180	671.529	9261.0
1	43.0	349.122	240.403	3742.0
2	98.0	550.462	551.516	7855.0
3	101.0	482.532	681.115	4990.0
4	100.0	461.780	721.835	7594.0

Fig. 2. Data Preprocessing - Dataframe considered for problem statement

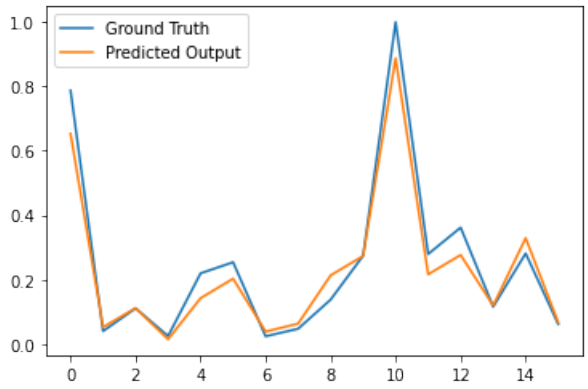

Fig. 3. Predicted Output and Ground Truth using Pseudo Inverse

```
Mean Squared Error:  0.003703069670117371
Root Mean Squared Error:  0.060852852604601625
Mean Absolute Error:  0.04440426308594414
```

Fig. 4. Pseudo Inverse MSE, RMSE and MAE

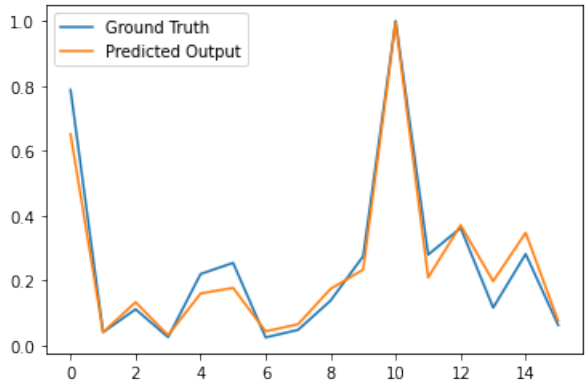

Fig. 5. Predicted Output and Ground Truth using Gradient Descent

```
Mean Squared Error:  0.003035157252952849
Root Mean Squared Error:  0.05509226128008224
Mean Absolute Error:  0.041220707075966555
```

Fig. 6. Gradient Descent MSE, RMSE and MAE

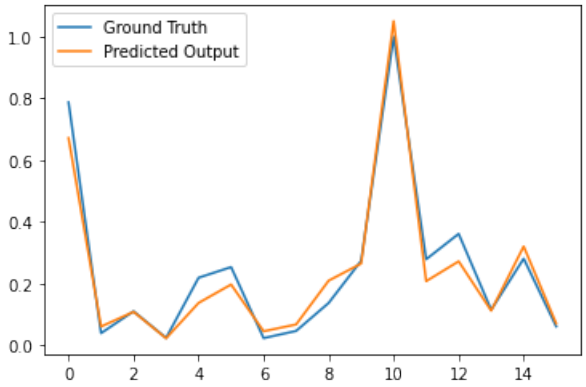

Fig. 7. Predicted Output and Ground Truth After changing the relationship between input and output

```
Mean Squared Error:  0.0029453304286035297
Root Mean Squared Error:  0.054270898542437364
Mean Absolute Error:  0.04156405753124275
```

Fig. 8. After changing the relationship between input and output- MSE, RMSE, MAE

V. Discussion

- The dataset was divided into training and test in 70:30 ratio.
- The size of X and Y variables for performing linear regression were 1 each.
- The RMS Accuracy obtained using the Pseudo Inverse Method is 94.8%.
- The RMS obtained is 95.7%. As we can see, this is a significant improvement over the Pseudo Inverse method.
- For change in relationship between the input and output variables, we consider that the fatalities (y) varies as square of Registered vehicles (thousands) (x1). This term is added after squaring x1. The RMS Accuracy obtained is 95.9%.
- The line fitting the data points most optimally was fitting better for smaller data points. But the larger data points were having more variations.
- It was observed that with increase in the number of the iterations we the gradient decent solution reached close to the analytical solution as observed in the result section.

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.

Appendix (Code)

the code is available here