Logistic Regression on Wisconsin Diagnostic Breast Cancer (WDBC) dataset

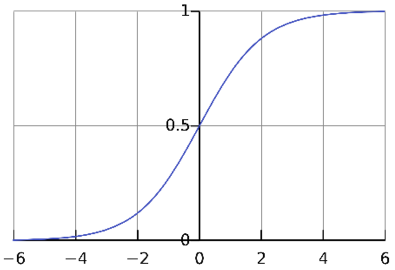
Akash Kumar Roy
akashroy@buffalo.edu
Person Number: 50316991
University at Buffalo, State University of New York

Abstract

This is a report for a Logistic Regression Model, performed on a Cancer Dataset. Here First I have preprocessed the data and then divided the data into Training, Validation and Test sets. To find a optimum minima for the weight matrix and bias I have used gradient descent and to calculate the loss I have used Cross Entropy loss function. I have trained this model on a dataset which has 569 samples along with 32 features. After training the model with different learning rates and different epoch I ended up a pretty good accuracy for the test dataset.

1. Introduction

Among many Machine Learning Classification Algorithms, Logistic Regression is one of the widely used and very popular one. Logistic regression is a classification algorithm used to assign observations to a discrete set of classes. It can be used in both Binary and Multi-Class Classification Problems. Unlike linear regression which outputs continuous number values, logistic regression transforms its output using the logistic sigmoid function to return a probability value which can then be mapped to two or more discrete classes. The sigmoid Function for Logistic Regression looks as Follows:



Here the given dataset will be trained and predicted through Binary Logistic Regression as the output has only two class either 0 (Cancer) or 1 (Not Cancer).

2. Dataset

The given Dataset is a Wisconsin Diagnostic Breast Cancer (WDBC) dataset. The dataset has 569 samples and 32 features. Among those 32 features 30 features are real value input features. The other two are column Id and the output prediction of the cancer cell. The main ten features of this dataset are radius, texture, perimeter, area, smoothness, compactness, concavity, concave points, symmetry, fractal dimensions. The other features are implemented by taking mean , standard deviations of these features.

3. Preprocessing:

Preprocessing of Data is generally necessary to remove redundant data, handling Null values and remove multicollinearity. In this Dataset I have performed three operations as a part of Preprocessing.

a. ***Adding Header to the Data Set:*** As the given Dataset doesn't contain any header so I have added header for each feature. This makes easier to access the column values.

b. ***Removal of the First Column:*** The first column in this data set represents the column ID. This is a redundant column because this feature doesn't have any impact on the output. So, we can perform our logistic regression without this column.

c. ***Changing M=1 and B=0:*** As the output result variable (cancer cell type) is given as a string I have mapped those values to binary. If the cell type is malignant then the column value will be 1 and if the cell type is benign then the column value will be 0.

d. ***Choosing x,y for Logistic Regression:*** Now I have divided the Dataset into x_data and y_data, where x_data is the dataset of the independent variables and y_data is the dataset of the dependent variable. y_data contains the value of type of the cancer cell that is either 1 or 0 and x_data contains the value of other 30 features.

e. ***Dividing the Dataset into Train, Validation and Test:*** The dataset has 569 samples. I have divided this dataset into three parts:

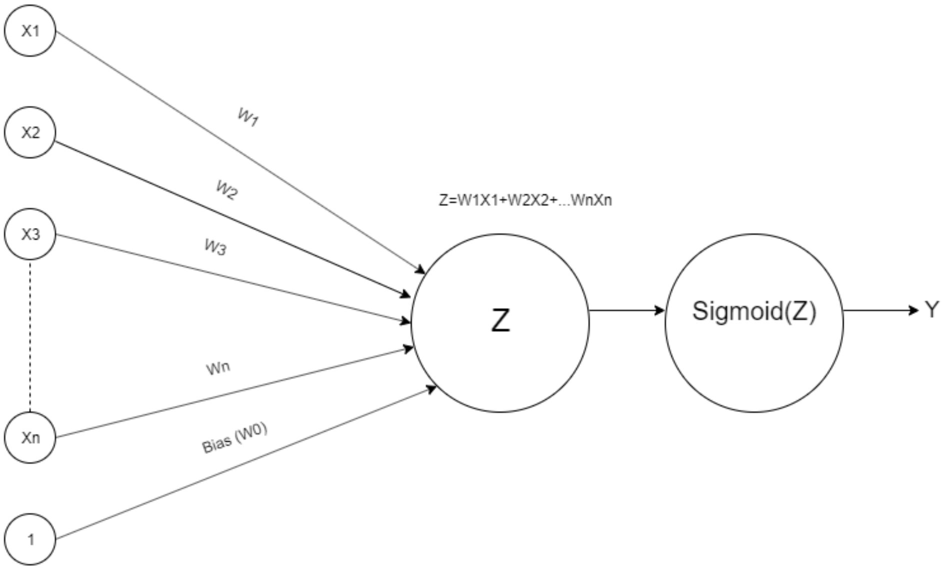
- Training Data (Contains 80% of the Dataset)
- Validation Data (Contains 10% of the Dataset)
- Test Data (Contains 10% of the Dataset)

f. ***Normalizing the Dataset:*** The goal of normalization is to change the values of numeric columns in the dataset to a common scale, without distorting differences in the ranges of values. As this is a multivariate problem I have normalized the dataset before the Logistic Regression. I have calculated the mean and standard deviation of x_training, x_validation, x_test and then used the following formula:

X_norm = (X - mean)/std

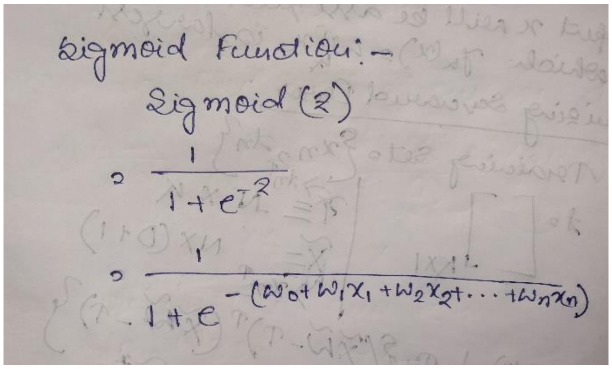
4. Architecture

a. Computational Graph for Logistic Regression



b. Equations for Logistic Regression

- **Sigmoid Function:**



Sigmoid Function:-

Sigmoid (z)

$$= \frac{1}{1+e^{-z}}$$

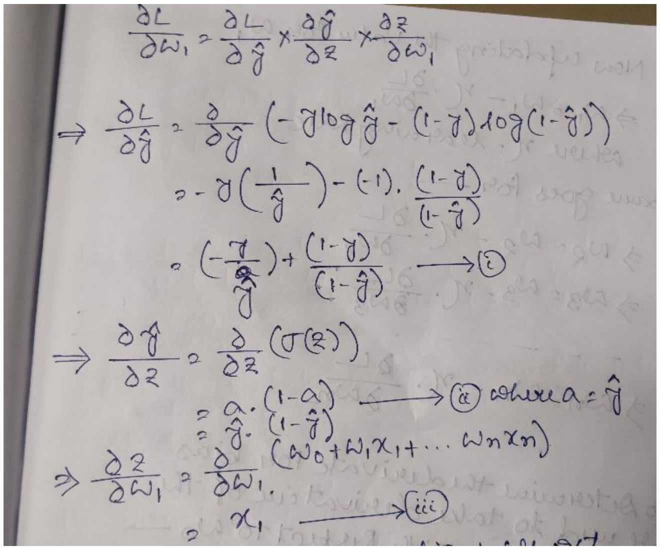
$$= \frac{1}{1+e^{-(w_0+w_1x_1+w_2x_2+\dots+w_nx_n)}}$$

- **Loss Function: (Cross Entropy Loss):**

$$L(p, y) = -(ylogp + (1-y)\,log(1-p))$$

- **Gradient Descent:** To calculate optimal minimum weights we need to derivative of the Loss function with respect to W1,W2,…Wn. And to calculate the optimum bias we need to take derivative of the loss function with respect to W0.

Lets update the weights first and take derivative with respect to w1 and it will become:



$$\frac{\partial L}{\partial w_1} = \frac{\partial L}{\partial \hat{y}} \times \frac{\partial \hat{y}}{\partial z} \times \frac{\partial z}{\partial w_1}$$

$$\Rightarrow \frac{\partial L}{\partial \hat{y}} = \frac{\partial}{\partial \hat{y}}\left(-y\log\hat{y} - (1-y)\log(1-\hat{y})\right)$$

$$= -y\left(\frac{1}{\hat{y}}\right) - (-1).\frac{(1-y)}{(1-\hat{y})}$$

$$= \left(-\frac{y}{\hat{y}}\right) + \frac{(1-y)}{(1-\hat{y})} \longrightarrow \textcircled{i}$$

$$\Rightarrow \frac{\partial \hat{y}}{\partial z} = \frac{\partial}{\partial z}(\sigma(z))$$

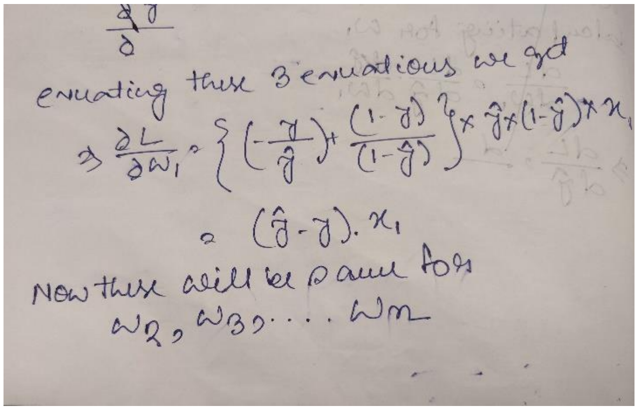
$$= a.(1-a) \longrightarrow \textcircled{ii} \text{ where } a = \hat{y}$$

$$= \hat{y}.(1-\hat{y})$$

$$\Rightarrow \frac{\partial z}{\partial w_1} = \frac{\partial}{\partial w_1}.(w_0 + w_1x_1 + \dots w_nx_n)$$

$$= x_1 \longrightarrow \textcircled{iii}$$

So the final derivative we get as follows:



equating these 3 equations we get

$$\Rightarrow \frac{\partial L}{\partial w_1} = \left\{\left(-\frac{y}{\hat{y}}\right) + \frac{(1-y)}{(1-\hat{y})}\right\} \times \hat{y} \times (1-\hat{y}) \times x_1$$

$$= (\hat{y} - y).x_1$$

Now these will be same for

$w_2, w_3, \dots\dots w_m$

Now updating weight(W) value and Bias value:

$$\boxed{\begin{aligned} b &:= b - \alpha db \\ w &:= w - \alpha dw \end{aligned}}$$

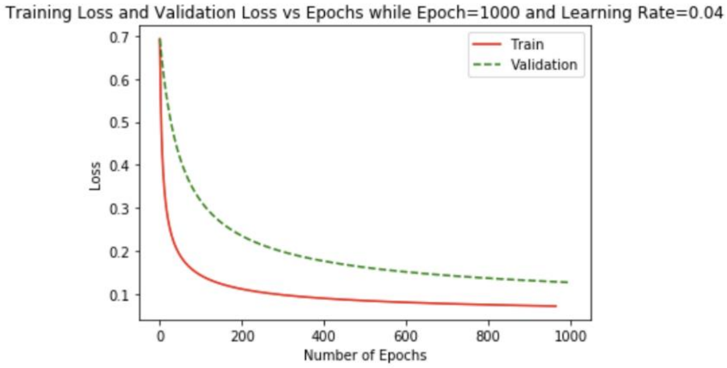
Where α= Learning Rate

5. Results:

Graph 1: Training loss and Validation loss VS Number of Epochs

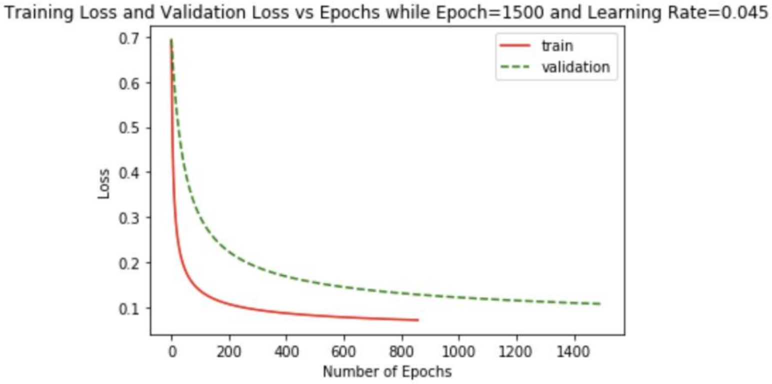
CASE 1: At first, I have taken the following Epoch and Learning Rate Intuitively:

- Epoch=1000 and Learning Rate= 0.04



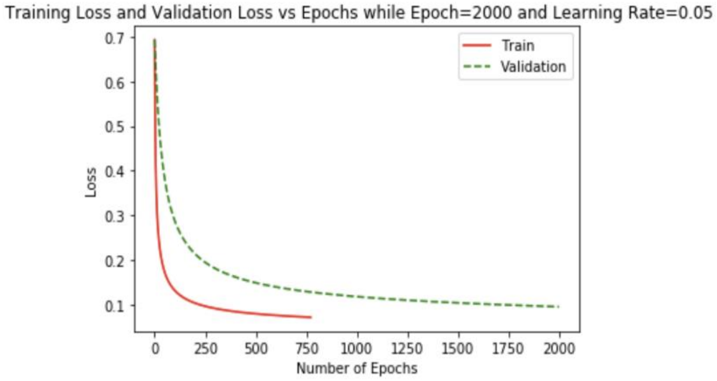
CASE 2: Updating Hyper-Parameters: To get a Better result updated the Epoch and Learning Rate

- Epoch=1500 and Learning Rate=0.045



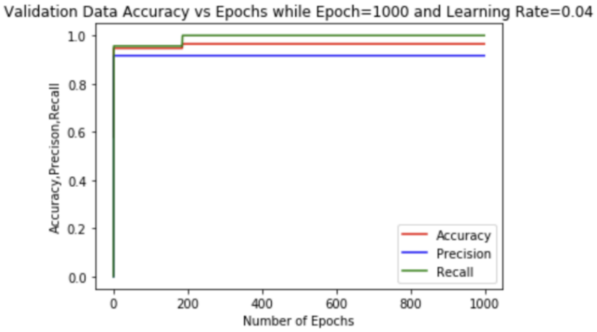
CASE3: Updating Hyper-Parameters: Updating the values once again to get a more optimal result

- Epoch=2000 and Learning Rate=0.05

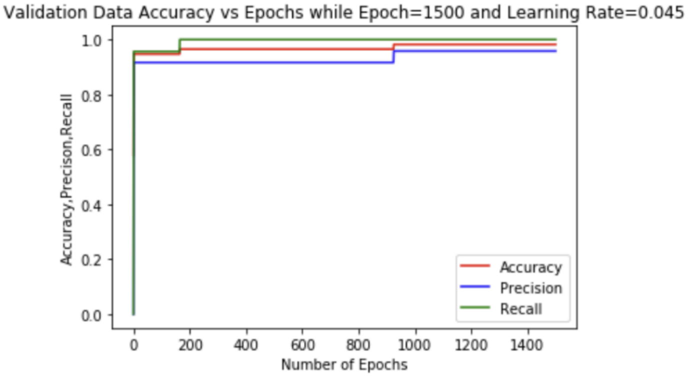


GRAPH 2: Accuracy, Precision, Recall vs Number of Epochs for Validation Sets

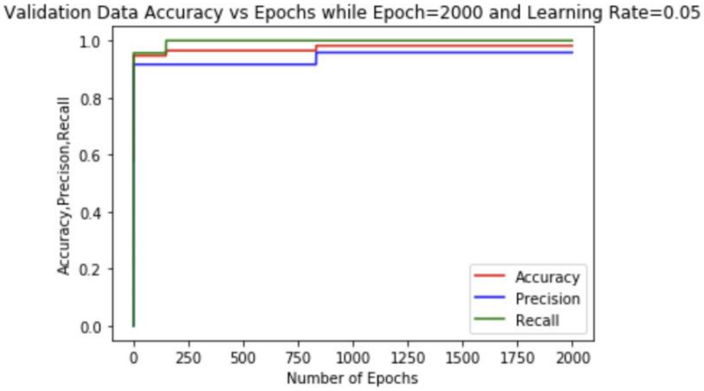
CASE 1: Epoch=1000 and Learning Rate= 0.04



CASE 2: Epoch=1500 and Learning Rate=0.045



CASE 3: Epoch=2000 and Learning Rate=0.05



Calculation of Accuracy, Precision, Recall for Test Dataset: We can see that for Epoch= 2000 and learning rate= 0.05 we are getting most optimum result. So now I have used those value to calculate my most optimum weights (W_i) and bias (W0) value. Once I got that, I have used those optimum weights (W_i) and bias (W0) to predict the output of the test dataset and after that I got the following confusion matrix:

Confusion Matrix: [[36 0] [1 20]]

Accuracy: (36+20)/(36+0+1+20) => 0.9824561403508771 * 100 => 98%

Precision: (36)/(36+1) => 0.972972972972973 => 97%

Recall: (36)/(36+0) => 1 => 100%

6. Conclusion:

To sum it up here we have used Binary logistic regression because the dataset contains a dependent variable is dichotomous and the independent variables are either continuous or categorical. After training the data for three different learning rate finally we found a proper weight and bias for which we get an accuracy of 98% in the training dataset.

7. References

- Lecture Slides
- Bishop - Pattern Recognition And Machine Learning - Springer 2006
- Wikipedia