Assignment 1 - CSE 574 - Introduction to Machine Learning

Adeline Grace George

October 10, 2021

1 Assignment Overview

The aim of the assignment is to perform classification on a two-class problem using Machine Learning and classify if a person has Diabetes(class 1) or not(class 0). In the first part of the project, Logistic Regression is performed using Gradient Descent and Cost function equations derived in class on the given dataset.

2 Dataset

To implement Logisitic Regression with Pima Indians Diabetes Database dataset with 768 instances, the data was randomly partitioned into training, validation and test datasets each consitituting 60%, 20% and 20% of the overall data. The training dataset has 460 values, the validation and testing datasets have 154 values each.

1	Glucose (Blood Glucose Level
2	Pregnancies (The number of pregnancies the patient has had)
3	Blood Pressure (mm Hg)
4	Skin Thickness (Triceps skin fold thickness (mm))
5	Insulin level
6	BMI (Body Mass Index : weight in kg/(height in m)
7	Diabetes Pedigree Function
8	Age (In years)

3 Python

Spyder IDE has been used for implementation.

3.1 Logistic Regression Implementation - Part 1

In the given dataset, there are 8 features or independent variables and 1 dependent variable. The goal is to perform Logistic Regression to calculate weights

to be applied on each of the feature vectors by minimizing the loss.

3.1.1 Parameters and Equations used

Let 'm' be the number of samples in the training set and let 'n' be the number of features.

Feature Vector (**X**) be given by: $[. \quad . \quad .]_{nxm}$

n $\rightarrow$ Number of features (8 in our case)
m $\rightarrow$ Number of Samples in the training dataset (460 in our case)

Weight Vector (**W**) given by: $[. \quad . \quad .]_{nx1}$ is initialized with zeroes

Single Weight (**B**) is also initialized with zero

Outcome Vector (**Y**) be given by: $[. \quad . \quad .]_{1xm}$

Sigmoid Function (σ) be given by: $\frac{1}{1+e^{-}x}$

Probabilistic Predicitons (**A**) be calculated using: $\sigma(W^T * X + B) = \frac{1}{1+e^{-}(W^T * X + B)}$

3.1.2 Cost Function

Using the actual and predicted values we have at hand, loss is computed for each iteration using the Cost Function,

$$Cost = -\frac{1}{m}\sum_{i=1}^{m} y^{(i)} \log(A^{(i)}) + (1 - y^{(i)}) \log(1 - A^{(i)})$$

3.1.3 Gradient Descent

Calculated the Gradient Descent for each iteration to move closer to minimum loss using the functions,

$$dW = \frac{dCost}{dW} = (A - Y) * X^T$$
$$dB = \frac{dCost}{dB} = (A - Y)$$

The Weight vector and Single weight are updated and the process is repeated for the specified number of iterations (which is taken as 1,000,000 in our case)

$$W = W - \alpha * dW^T$$

$$B = B - \alpha * dB^T$$

where $\alpha \rightarrow$ Learning Rate (fixed at 0.00023 to minimize loss)

3.1.4 Visualization

The below graph shows how the cost decreases with increasing iterations:

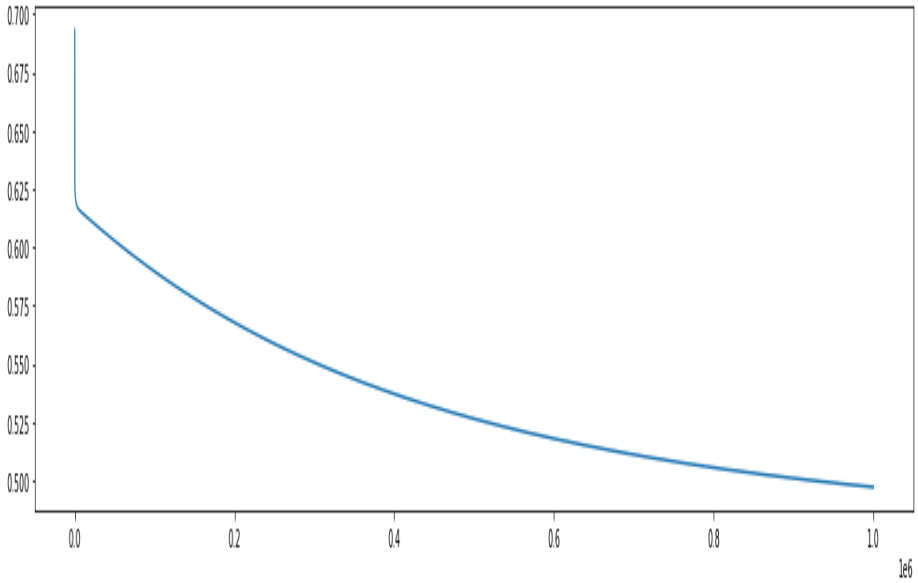

Figure 1: Cost Function plot

3.1.5 Results

The Weight vector (W) is given by:

$$W = \begin{bmatrix} 7.44378749e-02 \\ 2.20031710e-02 \\ -1.90178653e-02 \\ -5.68178163e-03 \\ 2.63190367e-04 \\ 6.36173723e-02 \\ 2.65722839e-01 \\ 2.08435967e-02 \end{bmatrix}$$

The Single Weight (B) is given by:

B= -4.996789617956224

3.1.6 Accuracy

The accuracy of validation dataset: 79.22 %
The accuracy of test dataset: 77.27 %

3.2 Neural Networks Implementation - Part 2

In the given dataset, there are 8 features or independent variables and 1 dependent variable. The goal is to implement Neural Networks algorithm with 1,2 or 3 hidden layers with different regularization methods(l1, l2 or dropout). The aim is to effectively predict if the person has diabetes or not.

The advantage of using Neural Networks is that it is designed to mimic the brain in it's capabilities to learn from the given set of features and make calculated predictions.

3.2.1 Implementation of the Model

The model is implemented by making use of **Keras** and **Tensorflow** libraries and the inbuilt functions extensively. This model has 8 neurons in the input layer, followed by 12 neurons in the hidden layer and 1 neuron in the output layer(since this is a binary classifier problem).

The **Sequential Model** with a single hidden layer has been used to implement the Neural Network. Since there are 8 features, the overfitting problem is bound to arise and regularization is used to minimize this and result in a smooth curve. L1 regularization of value 0.0001 is applied to both kernel and bias to give the highest accuracy for this particular dataset with epoch fixed at 100 and batch size fixed at 1.

3.2.2 Visualization

The below graph shows the train accuracy vs validation accuracy plotted against the number of iterations(epochs):

The below graph shows the train accuracy vs validation loss plotted against the number of iterations(epochs):

3.2.3 Accuracy

The accuracy of validation dataset: 75.98753944397 %
The accuracy of test dataset: 76.72743556456 %
The accuracy of training dataset: 75.2534556472 %

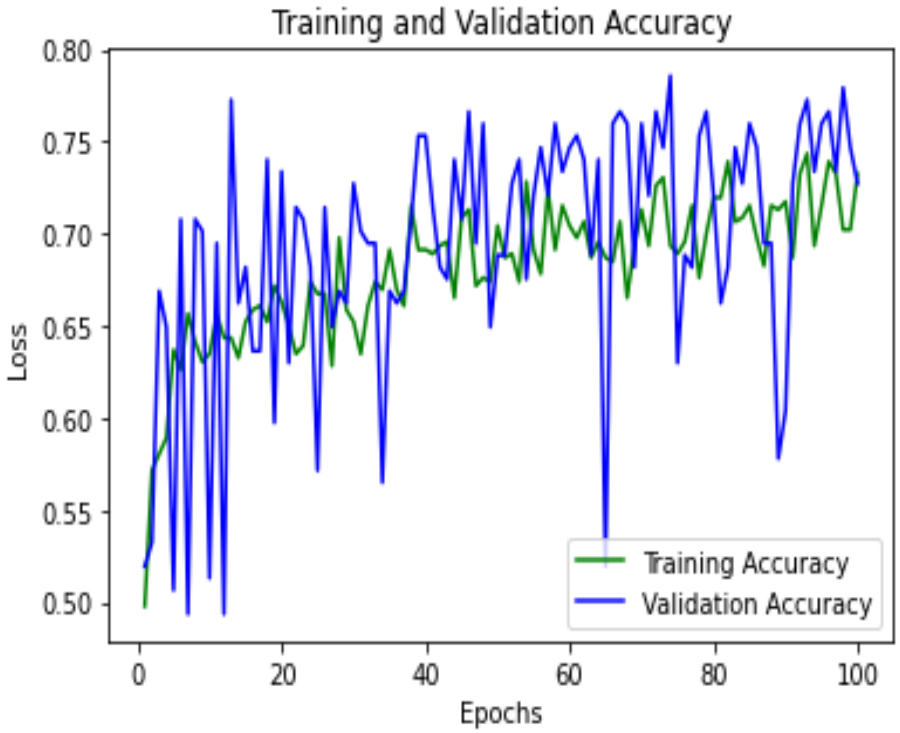

Figure 2: Training vs Validation Accuracy plot

3.3 Adding Dropout Regularization - Part 3

Performed a comparison between the different regularization methods - L1, L2 and dropout and the results from dropout are indicted below. It resulted in a slightly lower accuracy percentage for training set, validation as well as the test data set.

3.3.1 Visualizaton

The below graph shows the train accuracy vs validation accuracy plotted against the number of iterations(epochs):

The below graph shows the train accuracy vs validation loss plotted against the number of iterations(epochs):

3.3.2 Accuracy

The accuracy of validation dataset: 74.67532753944397 %
The accuracy of test dataset: 72.72727489471436 %
The accuracy of training dataset: 72.54767692878965 %

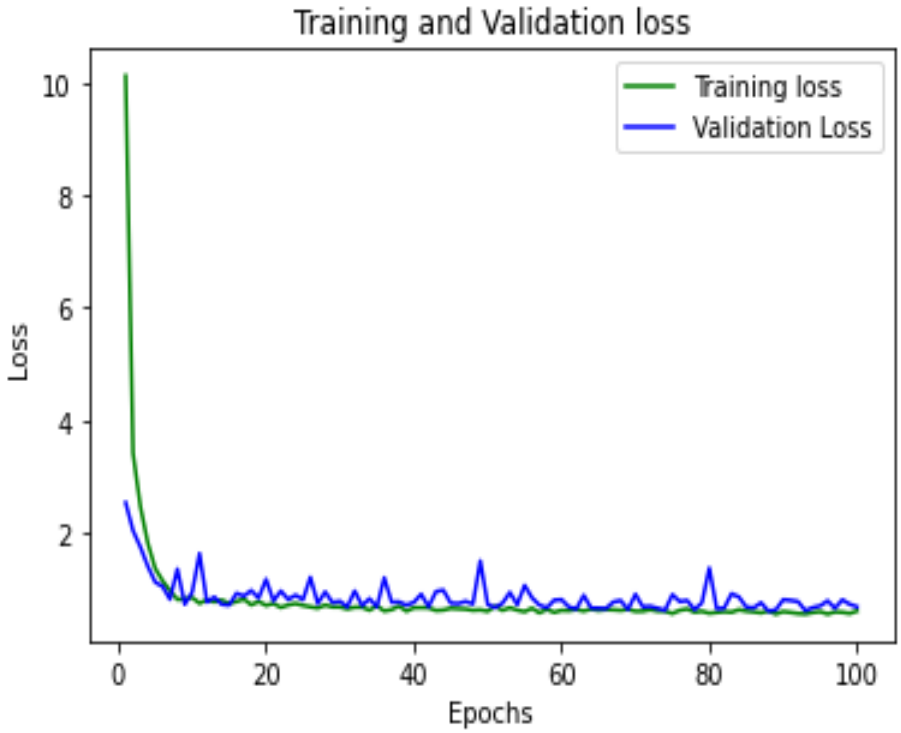

Figure 3: Training vs Validation Loss plot

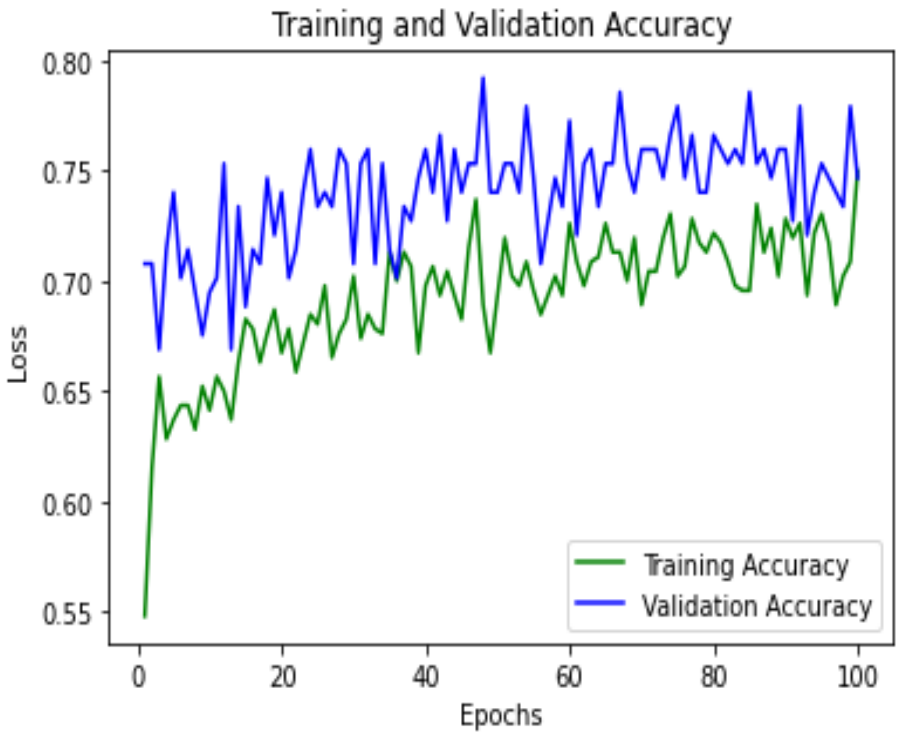

Figure 4: Training vs Validation Accuracy plot

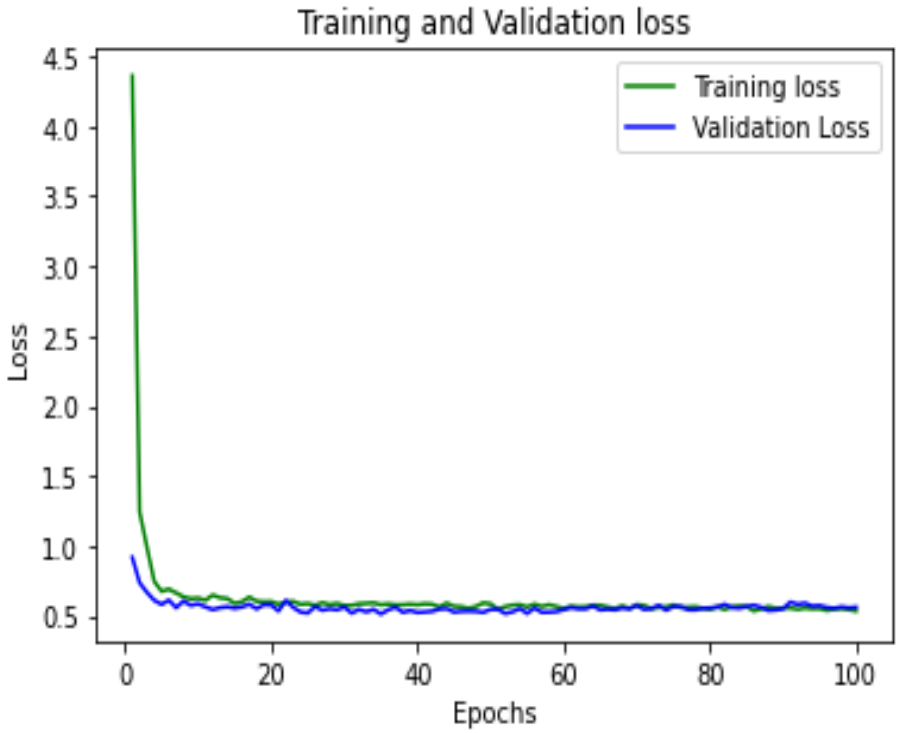

Figure 5: Training vs Validation Loss plot