Assignment 1 – Logistic Regression

Vishal Patil

10rd Oct 2021

1.Assignment overview

Goal of the assignment is to apply Logistic regression to classify and predict given data based on certain features. For this purpose, I have used the sigmoid function to predict the output into 2 outcomes. Gradient descent algorithm is used. Bias is considered.

2. Data set

PIMA Indian diabetes data base is used with 8 different features 768 records. I have split the given dataset into train, validation, and test set based on the 60-20-20 percent ratio respectively. Training data has 460 samples and both validation and test datasets have 154 rows each.

3. Python Editor

Jupyter Notebook IDE was used for all the coding implementation.

4. Logistic Regression:

4.1 Sigmoid Function

Sigmoid function or logistic function is a special type of function at gives binary outputs and has an "S" shape output or sigmoid curve as it is called. The equation of the sigmoid function is as given below:

$\sigma(Xw) = 1/(1+e\hat{\ }-(Xw))$

4.2 Gradient Descent

Gradient descent is a technique using to find the minima of the function, that being the error function here. Minimizing this should give us the best predicted outputs. It is a way with which we can calculate the weights related to each feature by an iterative method that involves adjusting the weights and bias with each iteration with an approximate learning rate

Gradient is:

$J(w) = \sum_{i=1}^{N} -C_i \, log \; \sigma(w^T x_i) - (1-C_i) log \; (1-\sigma(w^T x_i))$

Which can be rewritten as :
$\nabla J(w) = X^T(Y_{predicted} - Y)$

Here $Y_{predicted}$ Ypredicted and Y Y are column vector with the predicted output and the Actual class(target) respectively.

5.1 Implementation: Part 1

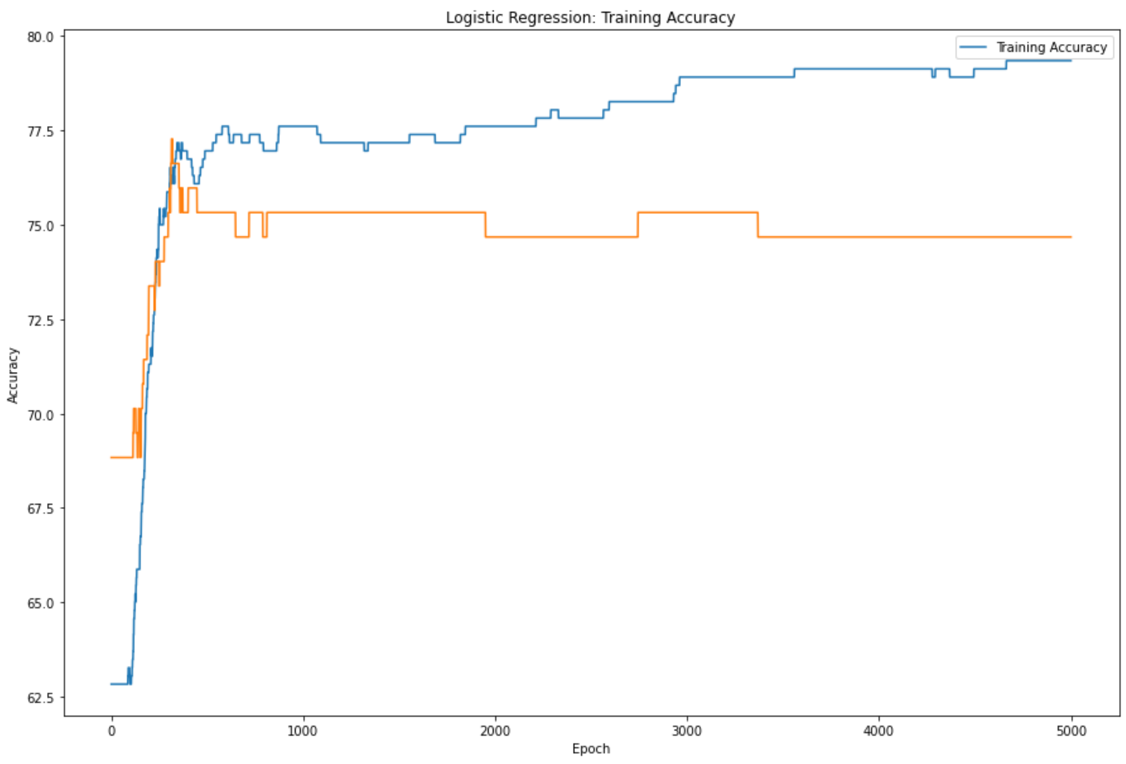
The data was pre-processed by replacing unacceptable values, zeros, is certain fields where data is required with the mean of value of the feature(column). Normalisation was done on the data since Gradient descent requires data from a range of 0 to 1 for accurate prediction and calculation of the weights.

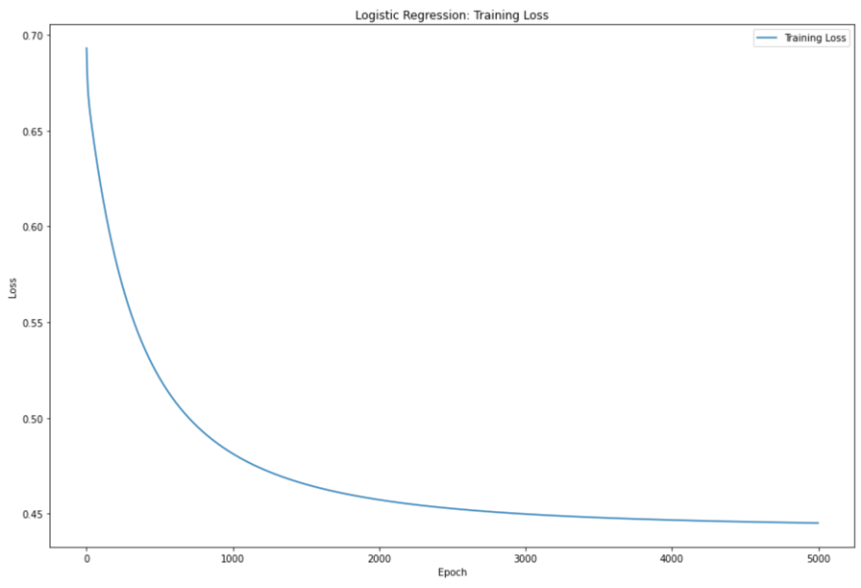
Sigmoid function and Gradient descent function operations were performed in accordance with the respective formulas stated above. Calculations were made on the training set with the iterative loop count as 5000 and the learning rate as 0.2

The following results were achieved:

Weight vector:

```
[[ 2.41779035]
 [ 5.95940742]
 [-1.30972846]
 [ 0.43415762]
 [ 0.0382174 ]
 [ 4.92321465]
 [ 1.53830613]
 [ 0.86218686]]
Accuracy of the model is:  79.35 %
validation data set accuracy:
Accuracy of the model is:  74.68 %
testing data set accuracy:
Accuracy of the model is:  75.97 %
```





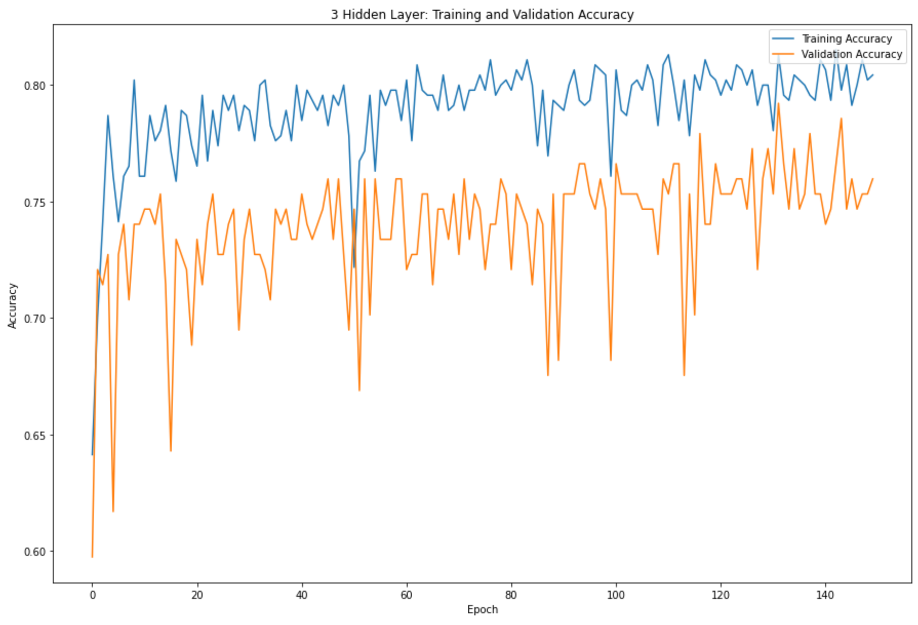
5.1 Implementation: Part 2

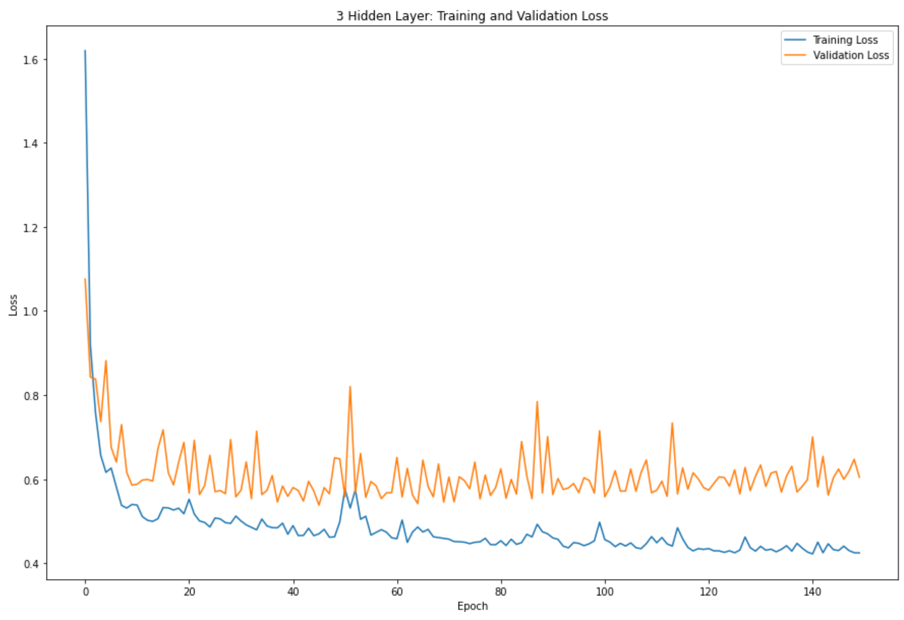
The data was pre-processed by replacing unacceptable values, zeros, is certain fields where data is required with the mean of value of the feature(column). Normalisation was done on the data.

3 hidden layers were used with ‘relu’ as the activation function. For hidden layer 1, L1 regularization was used with 64 nodes. For hidden layer 2, L2 regularization was used with 256 nodes. For Hidden Layer 3, L1-L2 regularization was used with 256 nodes. Adam function was used as the optimizer with learning rate as 0.01.

For the output layer, sigmoid is used as the activation function. Batch size = 64 and Epoch = 150

Output: `Loss 0.6044805645942688 Accuracy: 0.7597402334213257`





5.1 Implementation: Part 3

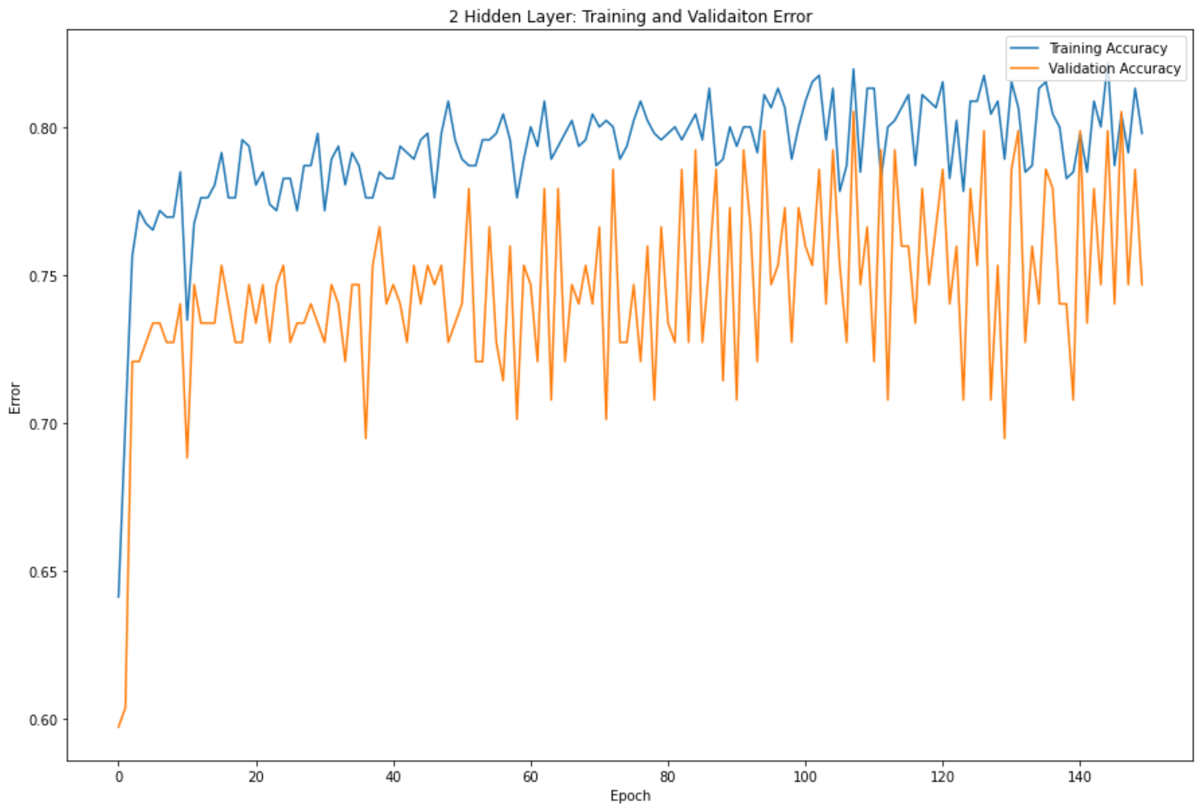
The data was pre-processed by replacing unacceptable values, zeros, is certain fields where data is required with the mean of value of the feature(column). Normalisation was done on the data.

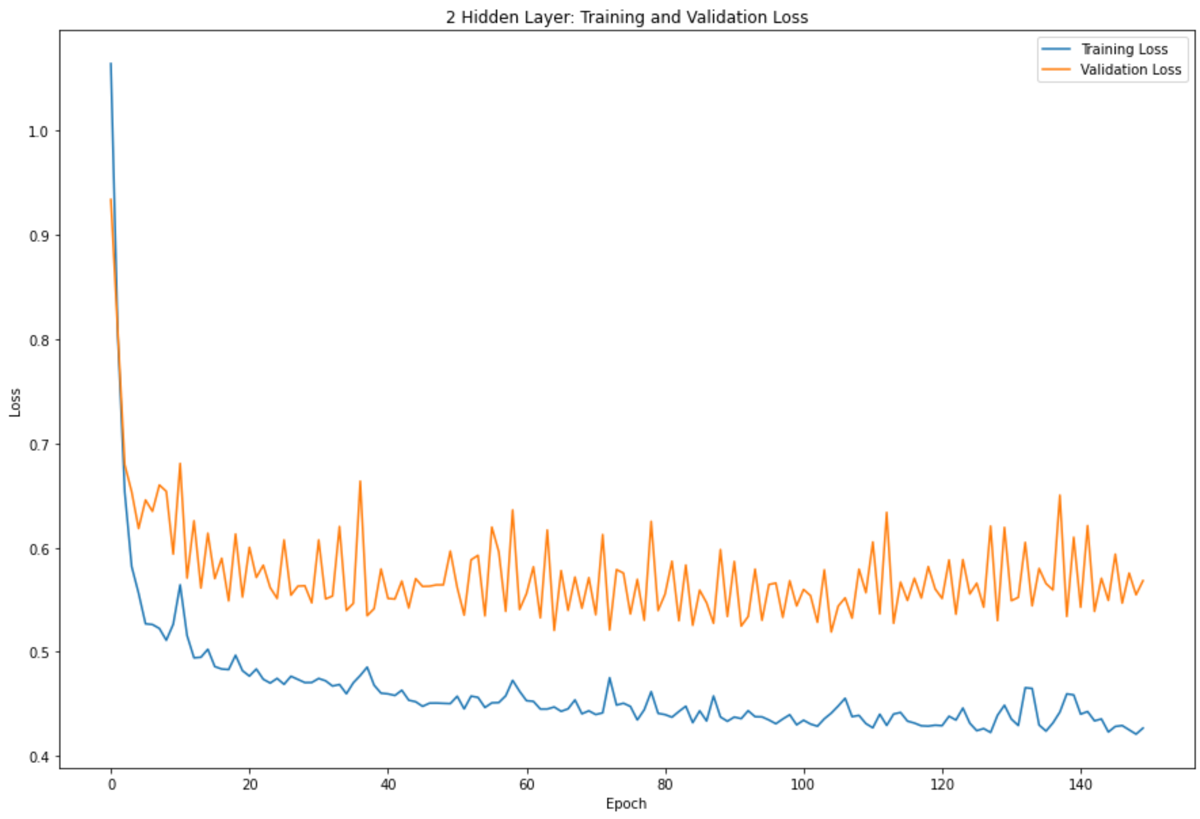
2 different neural networks, one with only 2 hidden layers with L1 regularization, and one with only 1 Dropout layer were created.

Details of NN with L1 regularization:

2 hidden layers with 64 nodes each, L1 regularization with parameter value as 0.001 epoch= 150, batch size = 64 optimizer = Adam, learning rate = 0.01.

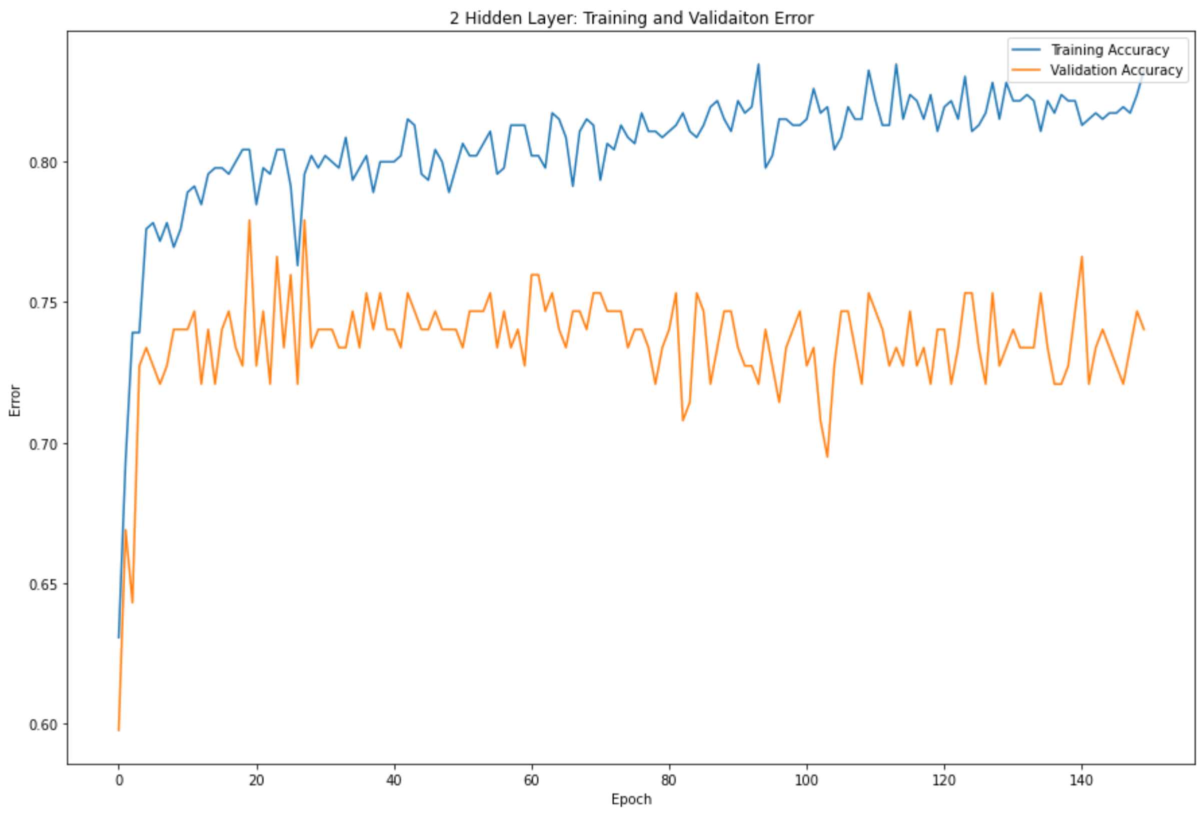
Output: Loss 0.5563203692436218 Accuracy: 0.7532467246055603

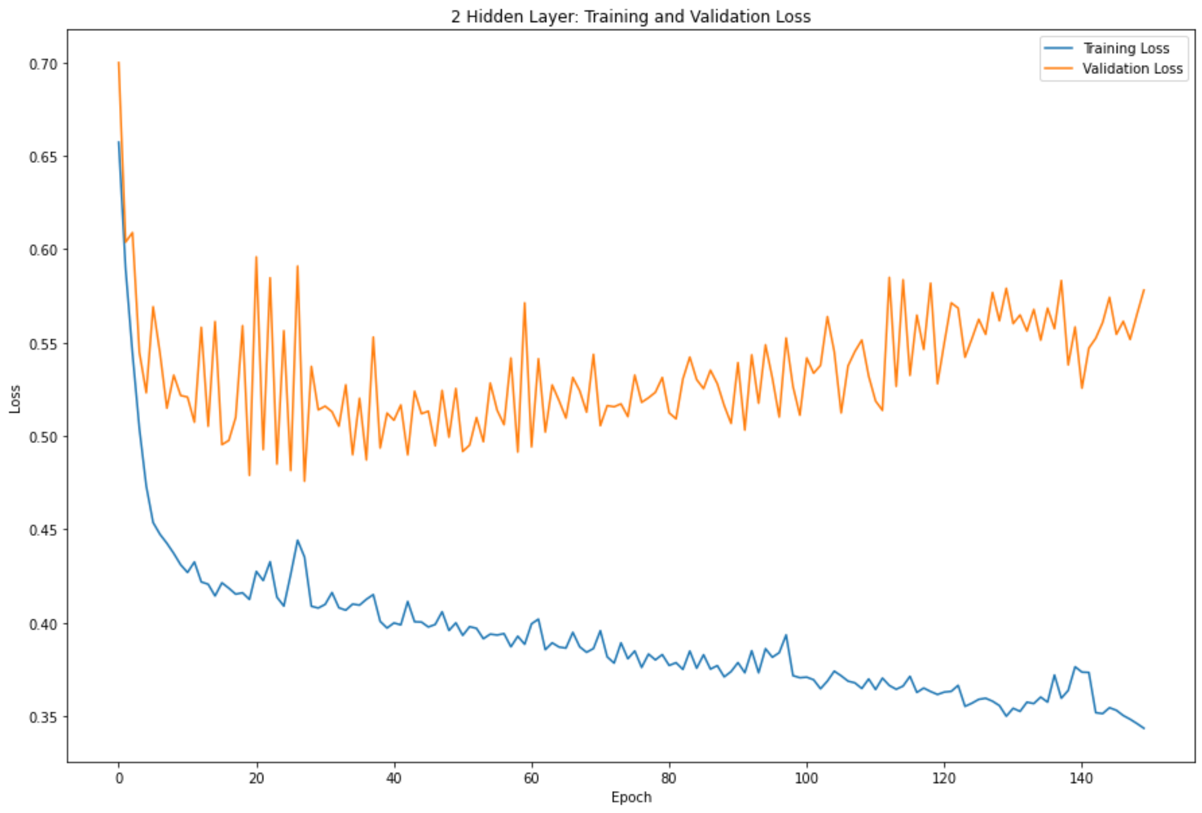




Details of NN with only 1 Dropout layer:

1 hidden layer with 64 nodes, one dropout layer with parameter value as 0.002 epoch= 150, batch size = 64 optimizer= Adam, learning rate = 0.01.





```
Output: Loss 0.5949813723564148 Accuracy: 0.7402597665786743
```

Observation: For Part3, it is observed that 2 hidden layers with L1 regularization performed better than NN with only a dropout layer.