Assignment 1 - Logistic Regression

Bhavyasai Survepalli – 50418493

bhavyasa@buffalo.edu

3rd October 2021

1 Assignment Overview

The goal of the assignment is to work with logistic model for regression. In the first part of the assignment, I have implemented logistic regression using the logarithmic cost equation and used gradient descent algorithm to minimize the linear regression cost function.

2 Algorithm

Logistic regression models the probability that each input belongs to a particular category. It is one of the most common machine learning algorithms used for binary classification. It predicts the probability of occurrence of a binary outcome using a logit function. It is a special case of linear regression as it predicts the probabilities of outcome using log function.

3 Dataset

In order to implement logistic regression models with dataset consisting of 768 samples, I have split the data samples using scikit-learn library as training, validation and testing data, each constituting 60%, 20% and 20% of overall data. Training dataset has 460 samples whereas validation dataset and testing dataset has 154 samples each respectively.

4 Python Editor

The IDE used is used Jupiter Notebook IDE for implementing the Logistic Regression.

5 Part 1

5.1 Reading Data

I have extracted the feature values of data by reading comma separated values from csv file using pandas data frame.

Dataset Features:

- Pregnancies
- Glucose
- Blood Pressure
- Skin Thickness

- Insulin
- BMI
- Diabetes Pedigree Function
- Age
- Outcome

5.2 Defining the sigmoid function

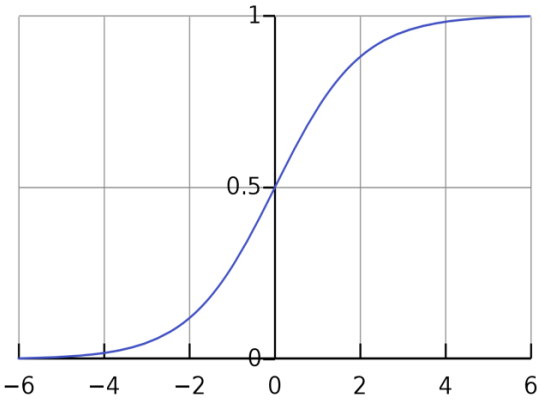
If a set of inputs X are given, the output is discrete. It belongs to one of two possible categories (0 or 1). In order to generate probabilities, logistic regression uses a function that gives outputs between 0 and 1 for all values of X. In this case the sigmoid function maps any real value into another value between 0 and 1.

Let us suppose an arbitrary instance has exactly one feature x and its output label is y. The first forward propagation step of logistic regression is defined by,

$$z = wx + b$$

Here, w is the weight associated with the feature x and b is the bias term. Later, taking the logistic sigmoid of z gives the activation value a.

$$a = \sigma(z) = \frac{1}{1 + e^{-z}}$$



5.3 Defining the cost function

$$L = -\Big(y \log a + (1 - y) \log (1 - a)\Big)$$

Where L is the Cost function, y is output label and a is the activation value.

If y = 1, when prediction = 1, the cost = 0, when prediction = 0, the learning algorithm is punished by a very large cost. Similarly, if y = 0, predicting 0 has no punishment but predicting 1 has a large value of cost.

5.4 Training the model using Gradient Descent for Logistic Regression

In order to minimize the loss function, we have to increase/decrease the weights and bias. This can be achieved by the derivative of the loss function with respect to each weight and bias. It tells us how loss would change if we updated the parameters.

$$\frac{dL}{dw} = \frac{a-y}{a(1-a)} \times a(1-a) \times x = (a-y)x$$

$$\frac{dL}{db} = \frac{a-y}{a(1-a)} \times a(1-a) \times 1 = a-y$$

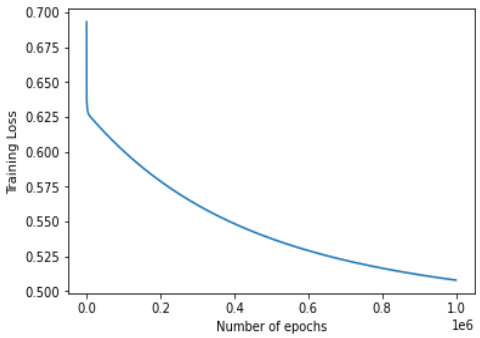
Hence, the loss function for the regression is defined by the average binary cross entropy loss averaged over the number of training instances (say m).

$$L = -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}\log a^{(i)} + \left(1-y^{(i)}\right)\log\left(1-a^{(i)}\right)\right]$$

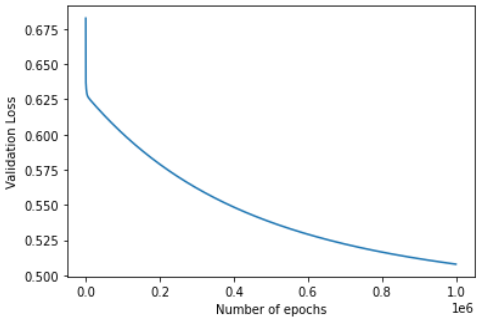
5.5 Plotting the graph of epochs vs cost values

The performance of the logistic regression on training and validation sets were analyzed for a value of the learning rate of the gradient descent algorithm. The learning rate value of 0.00019 is considered and to compare it's performance, the number of epochs is chosen to be 1000000.

a. Training cost vs Epochs



b. Validation cost vs Epochs



5.6 Defining Accuracy

Following are the epochs and learning rate values considered

epochs = 1000000

learning_rate = 0.00019

The accuracy is calculated for the validation and testing datasets and the result is obtained as follows.

Validation Accuracy: 78.571 %

Test Accuracy: 77.922 %

5.7 Summary

In this project phase, using gradient descent for logistic regression, a machine learning model to classify diabetes as class 0 or class 1 (discrete values) was built from scratch. While training the model, the performance on the validation set was analyzed for different settings of the algorithm learning rate. The best model was then chosen based on the validation accuracy and loss incurred over the epochs. The found model was evaluated on the test set and performance metrics like accuracy were calculated. Hence, we can conclude that greater accuracy can be found at Learning rate = 0.00019 and number of epochs is 1000000.

6 Part 2 – Neural Networks

6.1 Part 2 Overview

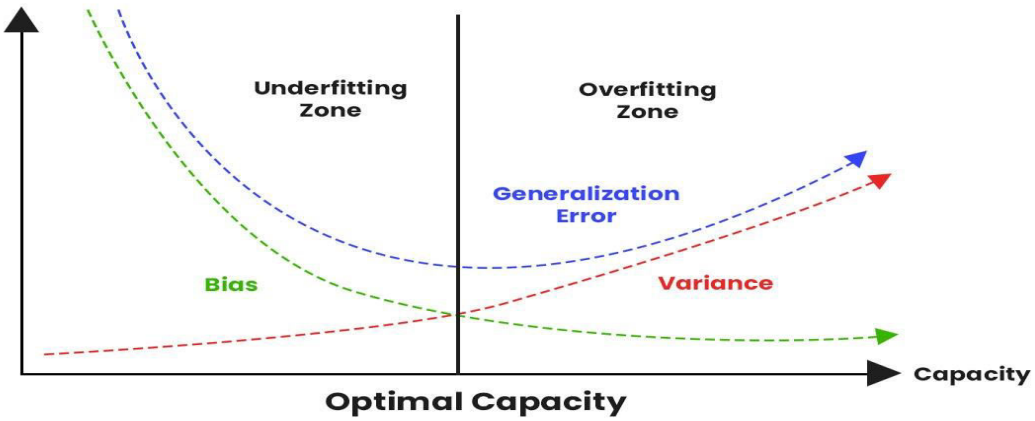
The goal of this part is to work with neural network. In the first part of the assignment, I have built neural network model first with one hidden layer and no regularization methods.

The second neural network model is built with two hidden layers and the l1, l2 regularization methods. The third neural network model is built with two hidden layers and the dropout is added to the second model.

6.2 Regularization Techniques

6.2.1 Comparing the regularization methods

Regularization is a technique which makes slight changes to the learning algorithm such that the model gives a better accuracy. This in turn improves the model's performance on the unseen data as well.



6.2.2 L1 Regularization

L1 regularization works by adding a term to the error function used by the training algorithm. The additional term castigates large weight values. For L1 regularization, the weight penalty term that's added to the error term is a small fraction of the sum of the absolute values of the weights.

6.2.3 L2 Regularization

L2 regularization works by adding a term to the error function used by the training algorithm. The additional term castigates large weight values. The two most common error functions used in neural network training are squared error and cross entropy error. In L2 regularization you add a of the sum of the squared weight values to the base error.

The most popular methods of regularization are L1 and L2. The generic cost function is updated by adding a term known as the regularization term.

Cost function = Loss (say, binary cross entropy) + Regularization term

L1 Regularization

$$\text{Cost} = \sum_{i=0}^{N} (y_i - \sum_{j=0}^{M} x_{ij} W_j)^2 + \lambda \sum_{j=0}^{M} |W_j|$$

L2 Regularization

$$\text{Cost} = \underbrace{\sum_{i=0}^{N} (y_i - \sum_{j=0}^{M} x_{ij} W_j)^2}_{\text{Loss function}} + \underbrace{\lambda \sum_{j=0}^{M} W_j^2}_{\text{Regularization Term}}$$

6.3 Dropout

Dropout means to dropping out units (both hidden and visible) in a neural network. A fully connected layer occupies most of the parameters, and hence, neurons develop co-dependency amongst each other during training which contains the individual power of each neuron leading to over-fitting of training data.

6.3.1 Phases

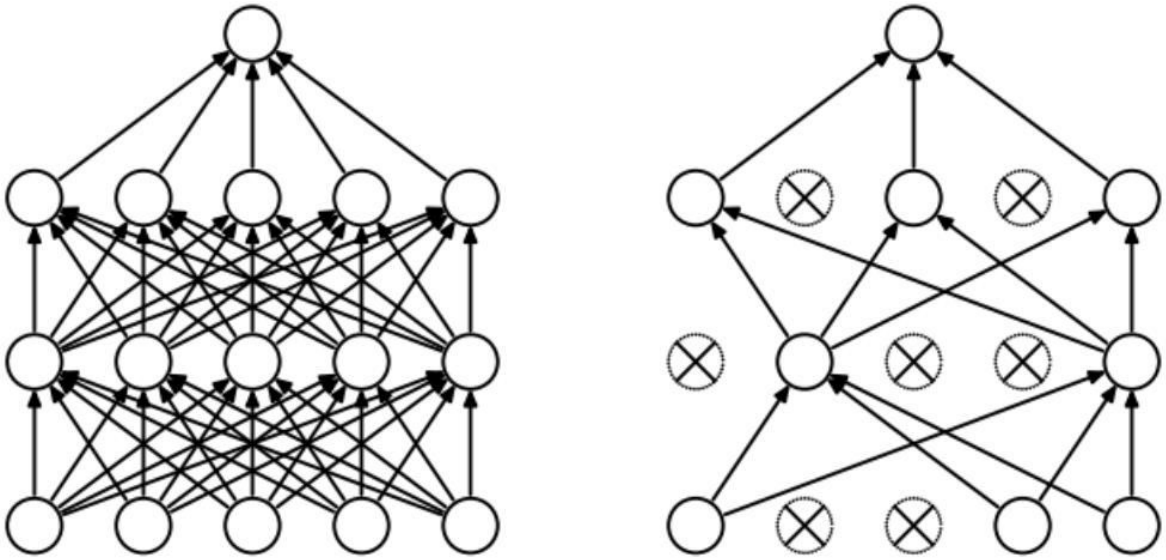
Dropout is an approach to regularization in neural networks which helps reducing interdependent learning amongst the neurons.

Training Phase:

Training Phase: For each hidden layer, for each training sample, for each iteration, ignore (zero out) a random fraction, p, of nodes (and corresponding activations).

Testing Phase:

Use all activations but reduce them by a factor p (to account for the missing activations during training).



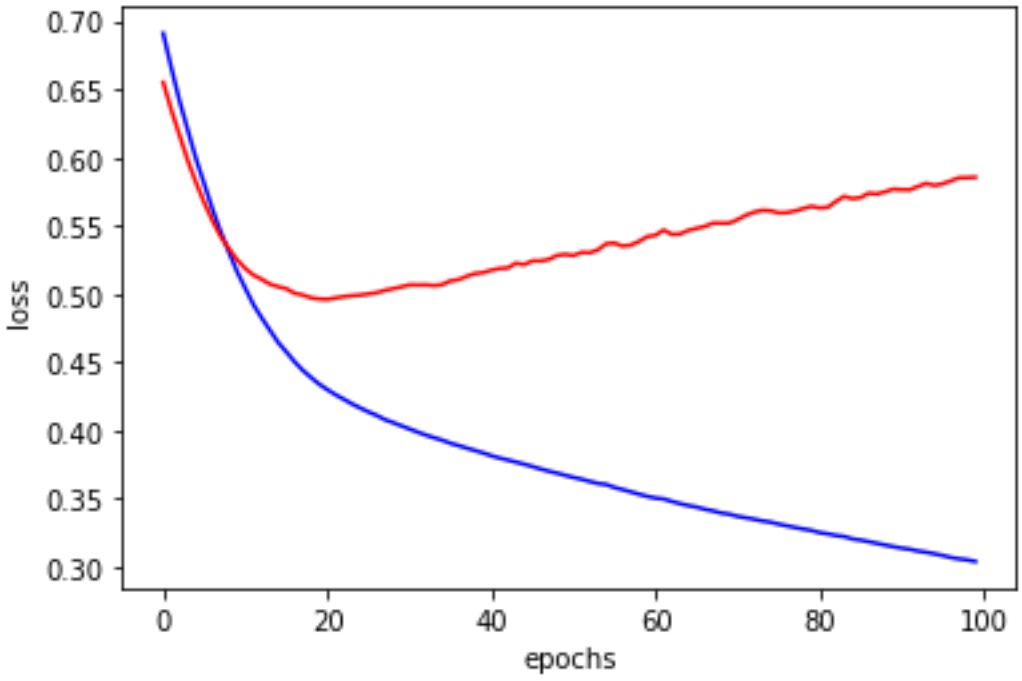
(a) Standard Neural Net (b) After applying dropout.

6.4 Model – Accuracy and Graphs

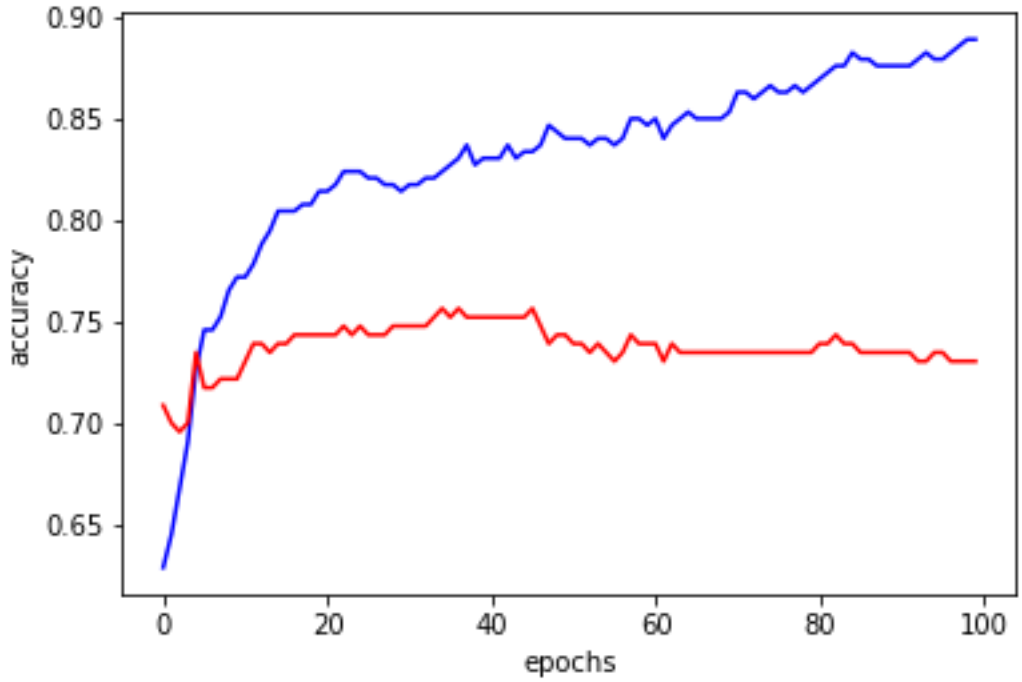
This model is based on neural network concept where it does not include any regularizations or dropout

Accuracy: 0.7402597665786743

a. Training Loss vs Epochs



b. Training Accuracy vs Epochs

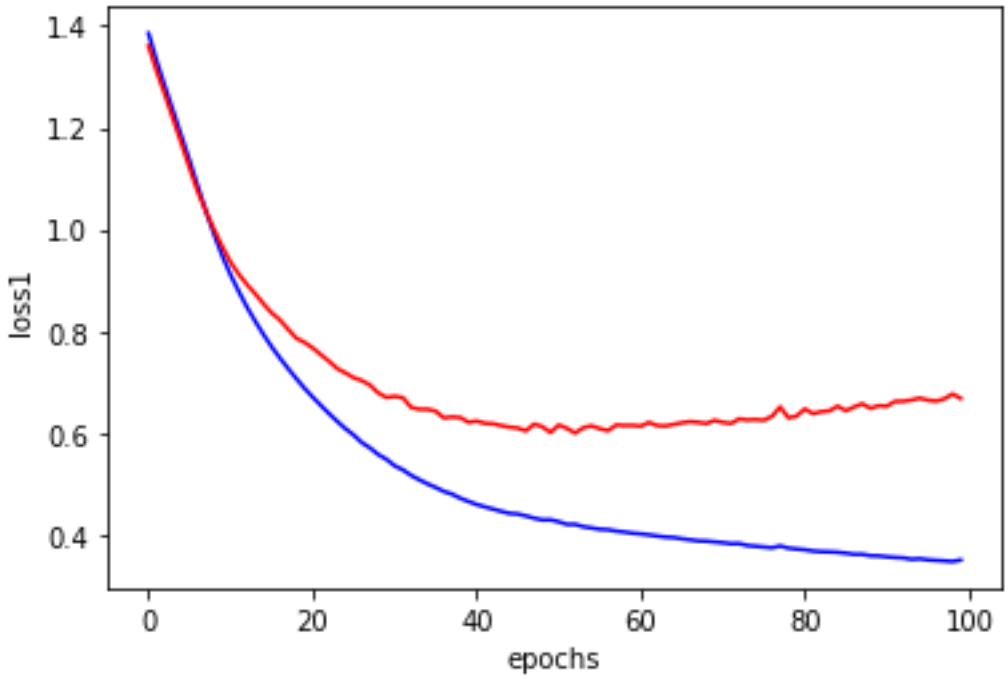


6.5 Model1 – Accuracy and Graphs

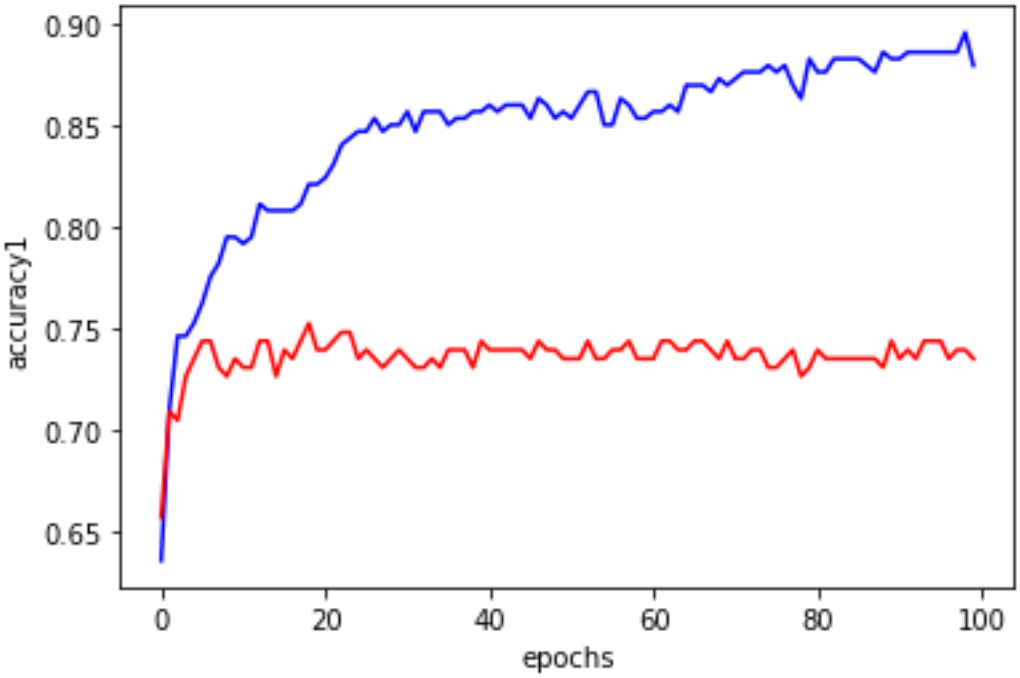
This model is based on neural network concept where it includes regularizations

Accuracy: 0.7662337422370911

a. Training Loss vs Epochs



b. Training Accuracy vs Epochs

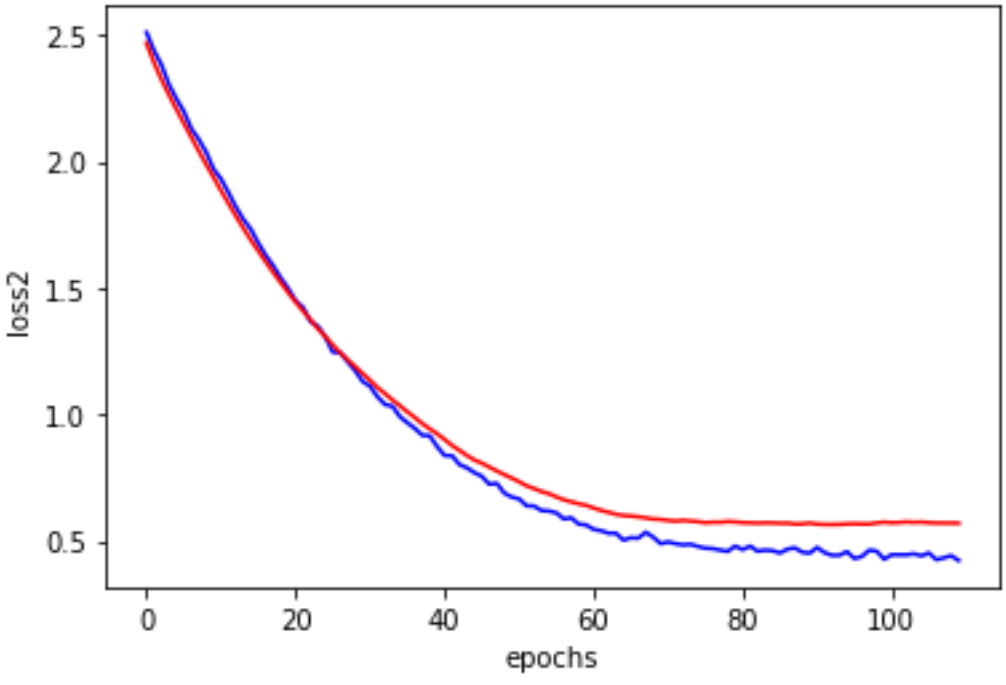


6.5 Model2 – Accuracy and Graphs

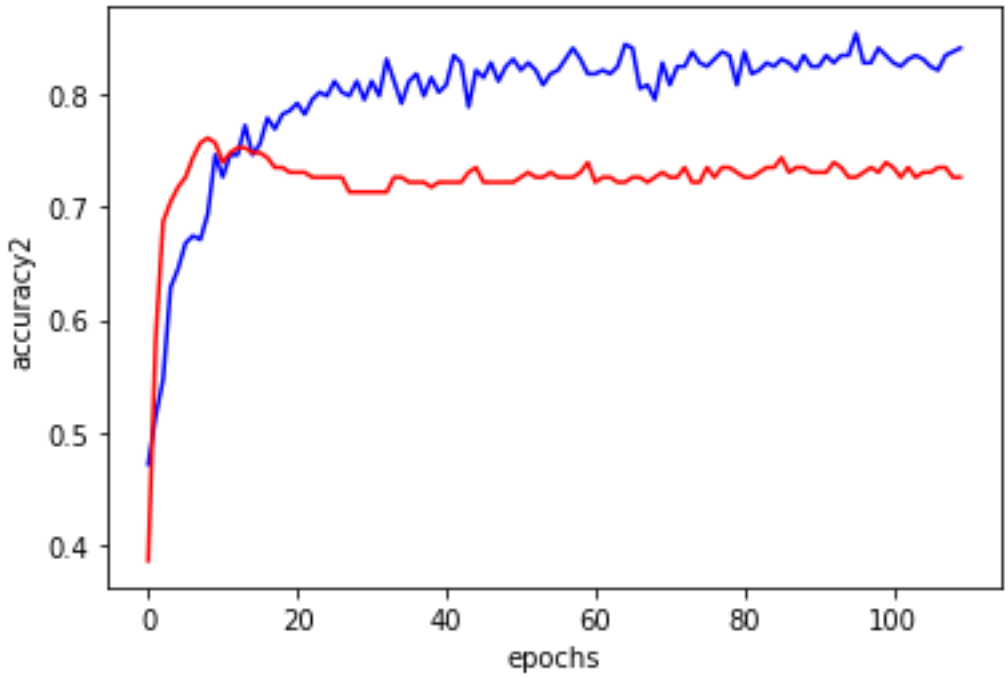
This model is based on neural network concept where it includes regularizations and dropout

Accuracy: 0.7705627679824829

a. Training Loss vs Epochs



b. Training Accuracy vs Epochs



6.6 Summary

In this project phase, neural network models were used to determine the accuracy obtained on testing data. In first model, one hidden layer and no regularizations were used. The accuracy obtained is 74%. In second model, l1_l1 regularization model was used on one of the hidden layers. The accuracy obtained is 76% as the overfitting of data was restricted with the use of regularizations. In third model, the l1,l2 regularizations were used along with the dropout to increase the accuracy obtained on the test set. The accuracy obtained is 77%. Hence, we can conclude that greater accuracy can be found when the regularization techniques and dropout is included while adding the hidden layers to the neural network model.