2) Neural Networks:

Approach:

* Split the data into 80 : 20 train : test
* The input features have been normalized to get mean as zero and standard deviation as 1.
* Initisalize the weights at each hidden layer(normal or uniform)
* When each batch is forward propagated, the errors are calculated at the final output layer and they are backpropagated to the previous hidden layers, then weights at each layer are updated using the gradients calculated during backpropagation. This will run for number of epochs given as input.
* Finally compute the accuracy and f\_score over test data

Dataset:

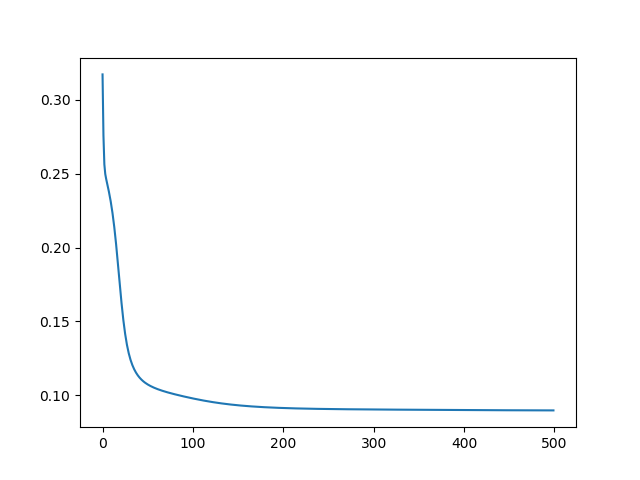
* In this assignment Neural Network has been implemented with multiple hidden layers to use on house prices dataset to predict if house prices are above median(1) or below median(0) . The dataset size is 1460 rows x 11 columns. The first 10 columns are the 10 input features and the last column contains target feature. All input features are continuous variables except one feature (Half bath) which is a binary feature. The target feature is a binary variable (0 or 1).
* The training and testing was done on: <housepricedata.csv>

Dependencies Used:

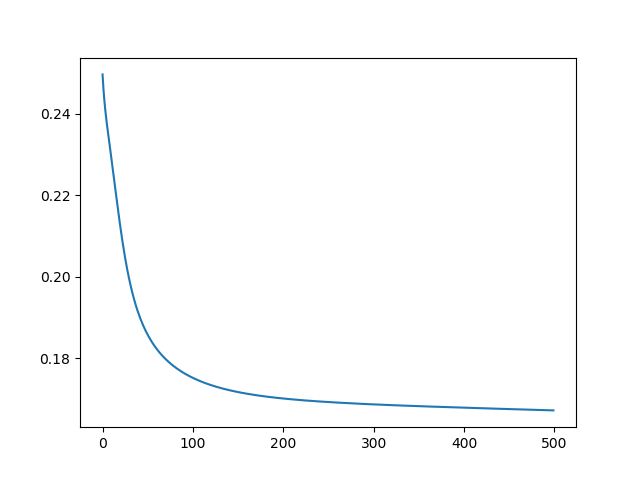
* Numpy to perform operations such as finding mean,std,weight initialization,etc on the inputs
* Matplotlib to perform graph plotting operations

Results:

Two Hidden Layer Model:

* Plot of the loss function looks like this:
* 
* Hidden layers : 2 hiden layers each of 5 neurons and 500 steps. And learning rate of 0.005
* (irrespective of which distribution was followed while initializing, final outcome was pretty much similar)
* precision: 0.8846153846153846 recall: 0.92
* accuracy: 0.8972602739726028
* f1\_score: 0.9019607843137256
* [Finished in 2.4s]

One Hidden Layer Model:

* Plot of the loss function looks like this:
* 
* Hidden layers : 1 hiden layer of 5 neurons and 500 steps. And learning rate of 0.005
* (irrespective of which distribution was followed while initializing, final outcome was pretty much similar)
* precision: 0.8513513513513513 recall: 0.9064748201438849
* accuracy: 0.8801369863013698
* f1\_score: 0.8780487804878048
* [Finished in 1.7s]