MACHINE LEARNING

NOVEMBER 6, 2017

LAB REPORT ON ARTIFICIAL NEURAL NETWORKS

PRESENTED BY :- UNIT PATEL 2015CSB1038

JATIN GOYAL 2015CSB1014

LAB REPORT ON ARTIFICIAL NEURAL NETWORKS

MULTILAYER PERCEPTRON

INTRODUCTION

The main aim of this part is to study the changes to the decision boundary and the training error with respect to parameters such as number of training iterations, number of hidden layer neurons and finally the learning rate.

OBSERVATIONS

The model is trained using 1000 epochs and the fixed learning rate of 0.01 having only one hidden layer but varying the number of nodes in the hidden layer.

Figure 1.1 shows training error against number of epochs for the different choices of hidden layer units. As the hidden layer units increase by a power of two the training error the curve gets steeper i.e. we see a steep decrease in the training error. We can derive this observation using the fact that as the number of hidden layer units increases convolutional model becomes more complex and hence fits the data faster and completely compared to models having less numbers of hidden layer units.

{ formula for the error is E(w,v) = - ∑y(k)log(o(k)) }*

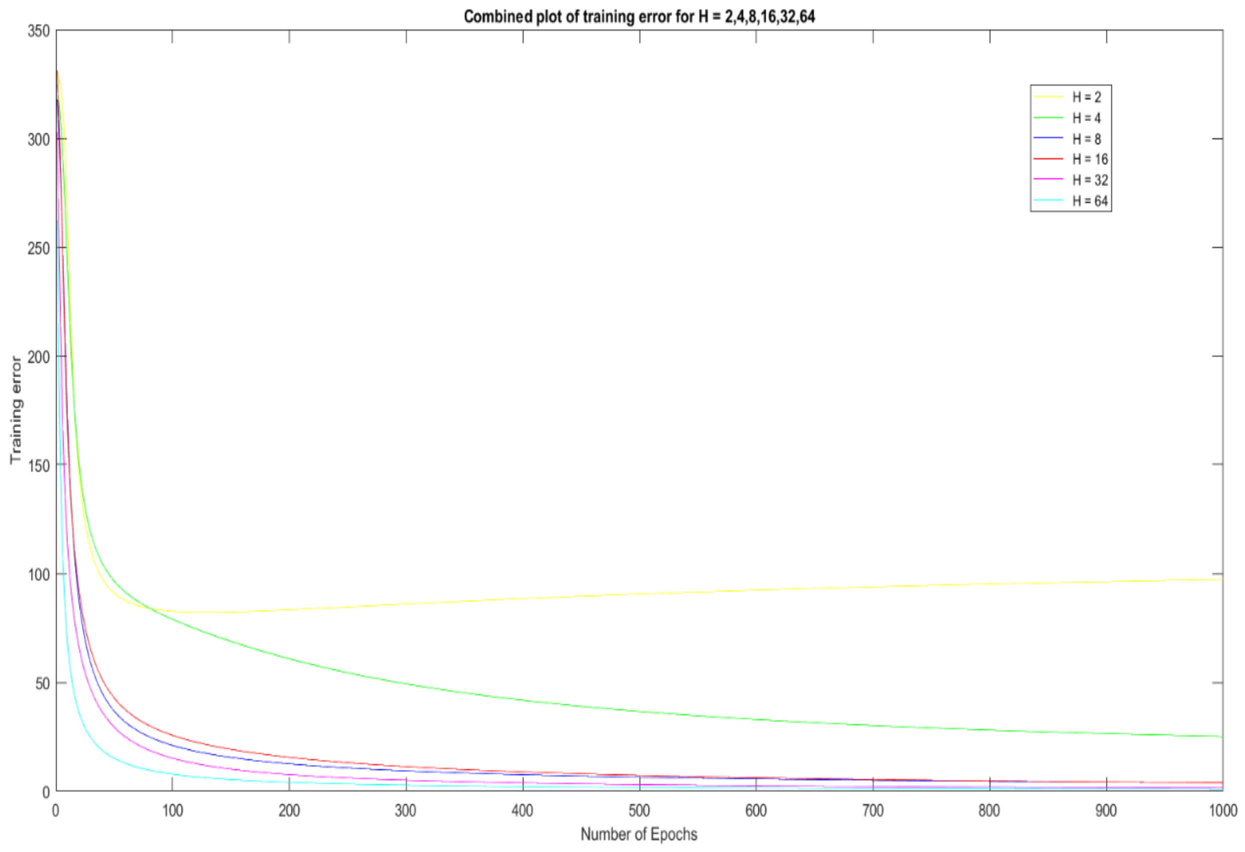


Figure 1.1

PLOTS FOR THE DECISION BOUNDARIES FOR DIFFERENT CHOICE OF LAYER UNITS

Figures in section 1.2 shows the decision boundaries for the hidden layer units 2 to 64 incremented by a power of 2. We observe that as the number of hidden layer unit increases, complexity increases and therefore we get better fitted and accurate decision boundaries. For H=2 we see that the model is too biased to classify the data. From H=4 onwards we can clearly observe that we are to classify the data correctly hence the model with H=4 is complex enough to classify given data.

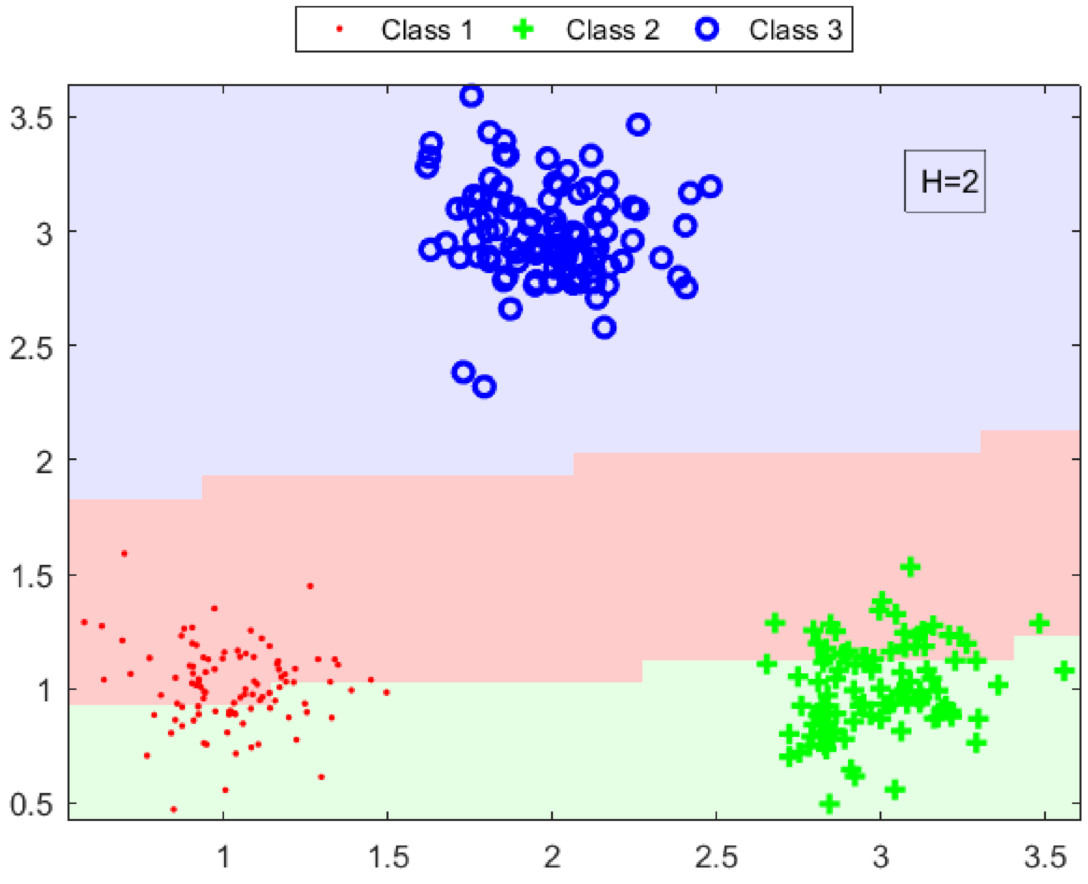


Figure 1.2(1) Decision Boundary for H=2

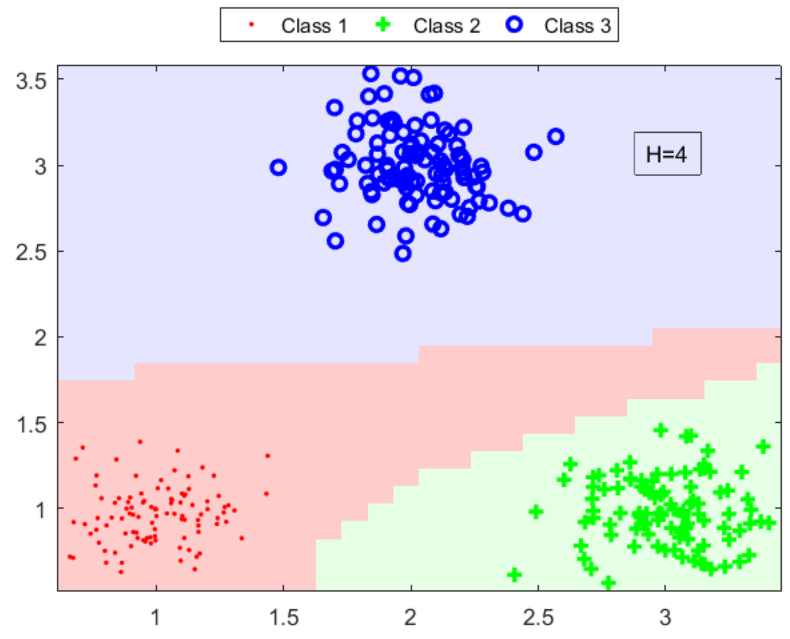


Figure 1.2(2) Decision Boundary for H=4

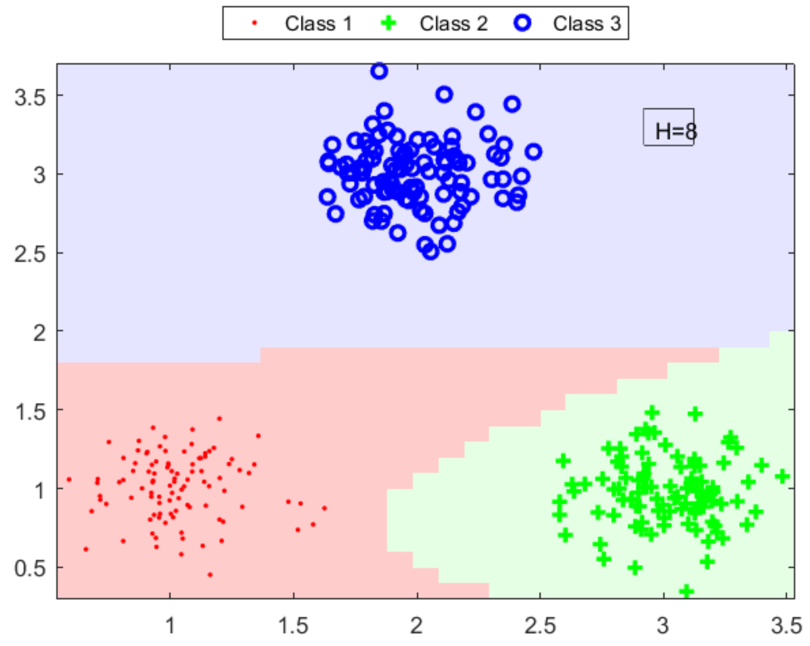


Figure 1.2(3) Decision Boundary for H=8

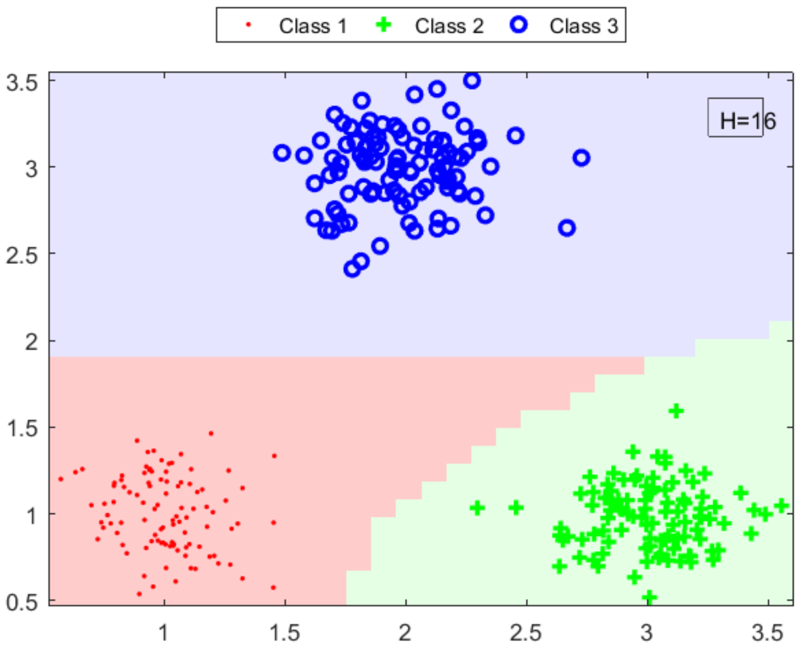


Figure 1.2(4) Decision Boundary for H=16

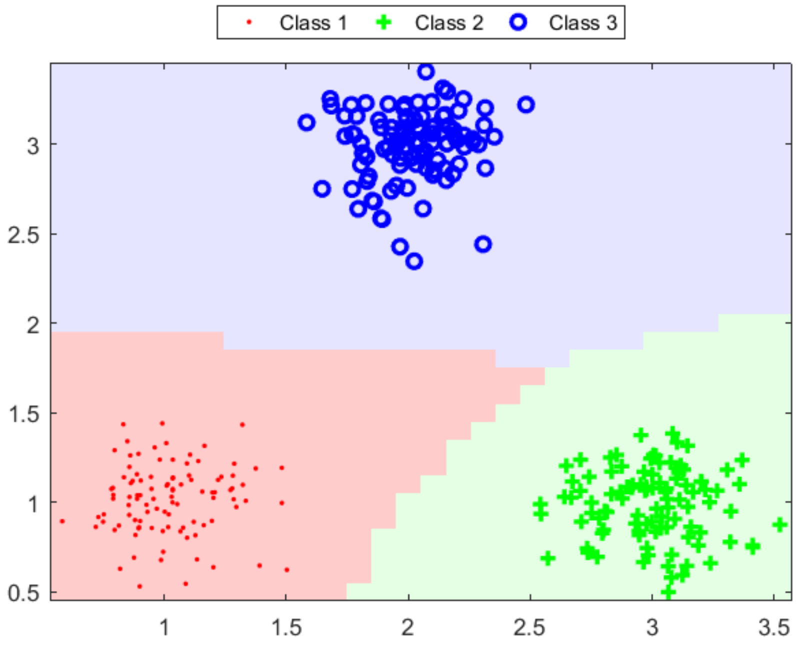


Figure 1.2(5) Decision Boundary for H=32

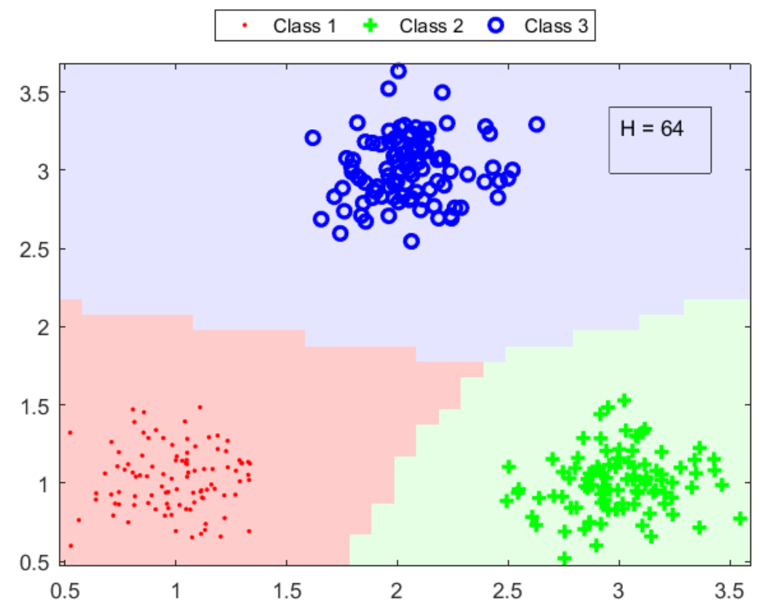


Figure 1.2(6) Decision Boundary for H=64

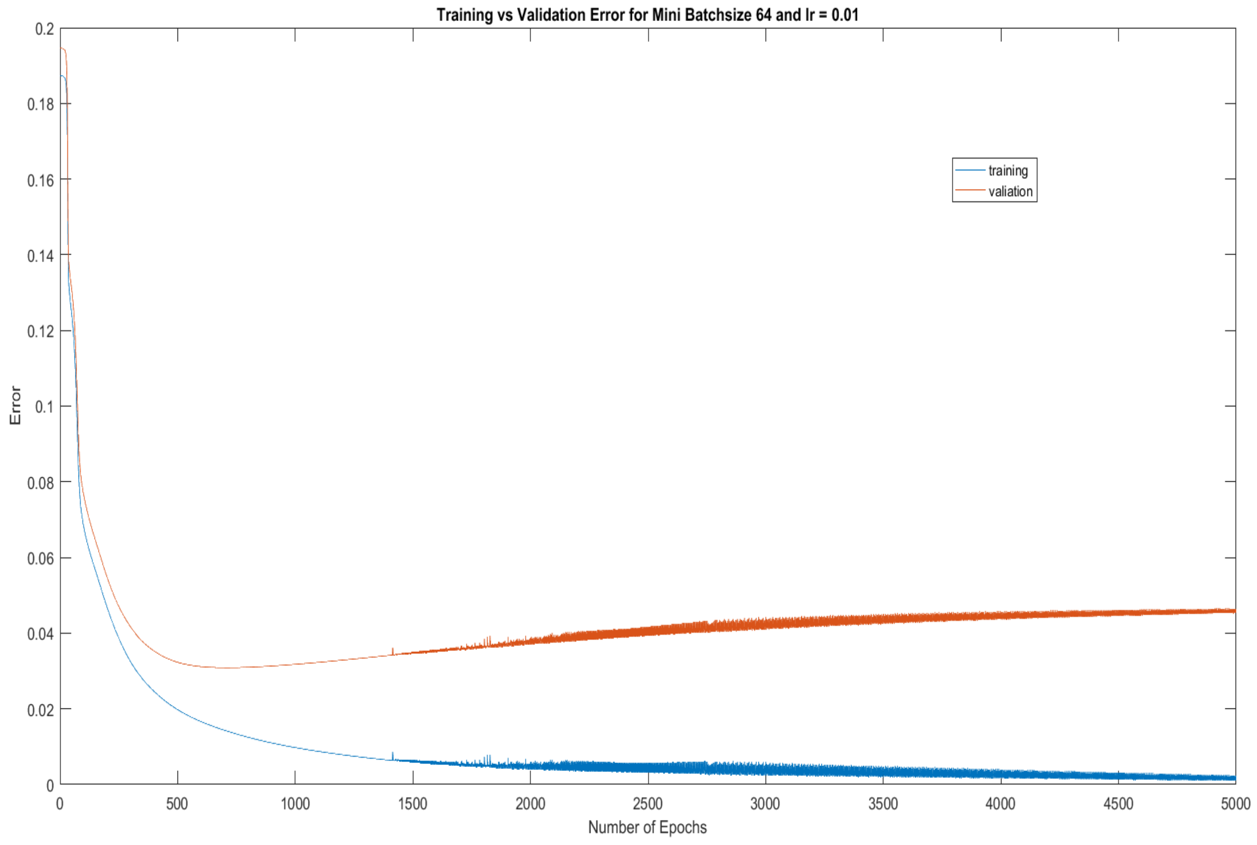
NEURAL NETWORK

INTRODUCTION

This part discusses the basic neural network and its implementation to predict the steering angle for a self-driving car application that is inspired by Udacity's Behavior Cloning Project given the image of the road ahead. The train data includes 32x32 grayscale images of the road ahead and are labelled with the steering angle. The preprocessing part is done by converting a 32x32image into a vector of 1024 size and each pixel is mapped to a input node of a neural network architecture 1024-512-64-1. Implemented a model with two hidden layers having number of units of size 512 and 64 and the output as a steering angle. The dataset is divided to training and validation in ratio 80:20. Weights of each layer have been initialized by uniformly spacing over the range [-0.01, 0.01]. The number of epochs, minibatch size, earning rate and the dropout percentages can be tuned

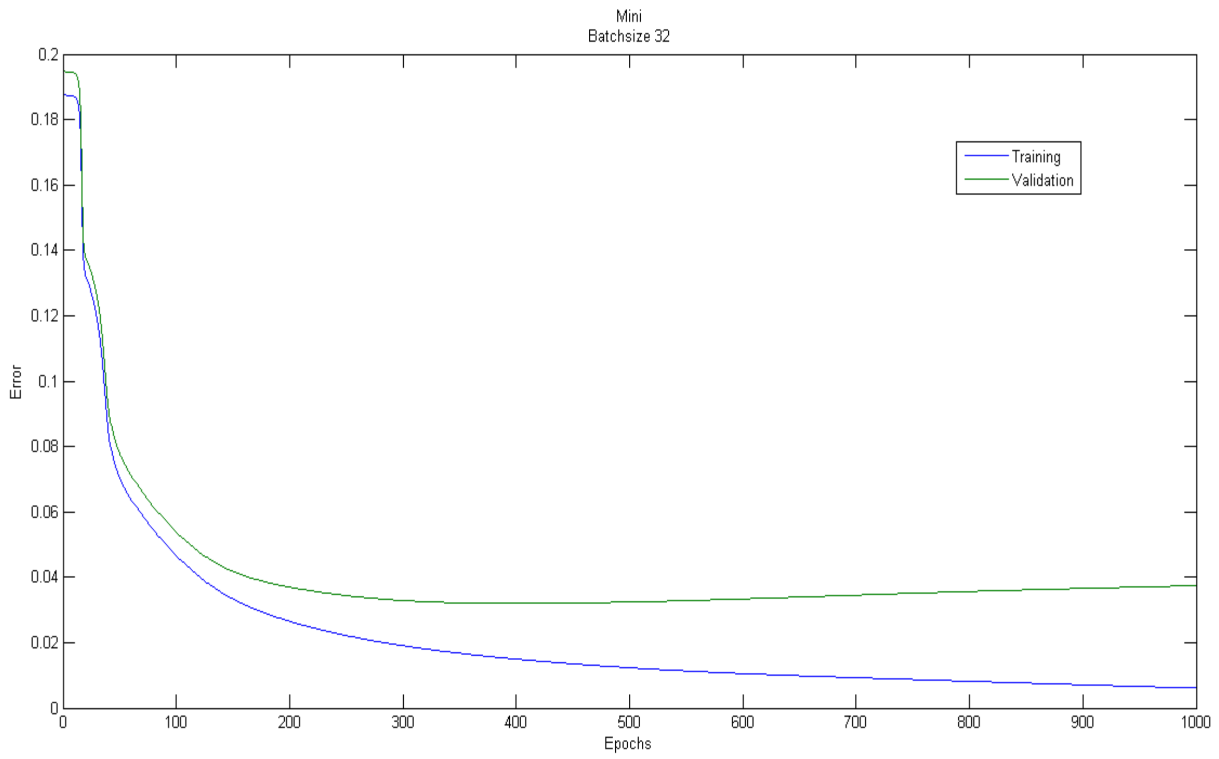
OBSERVATION

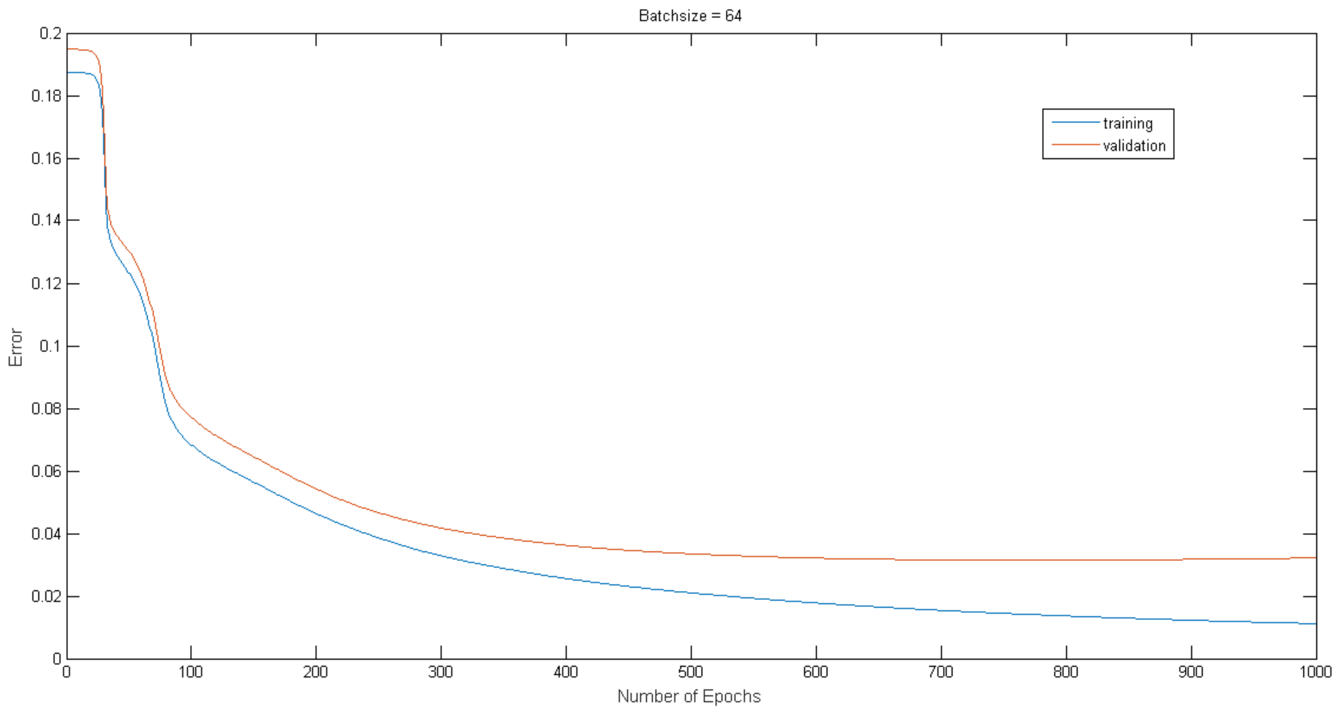
1. Shown below is the plot of sum of squares error on the training and validation set as a function of training iterations for 5000 epochs, with learning rate 0.01 and minibatch size of 64. As the training epochs increases, initially the training and the validation error

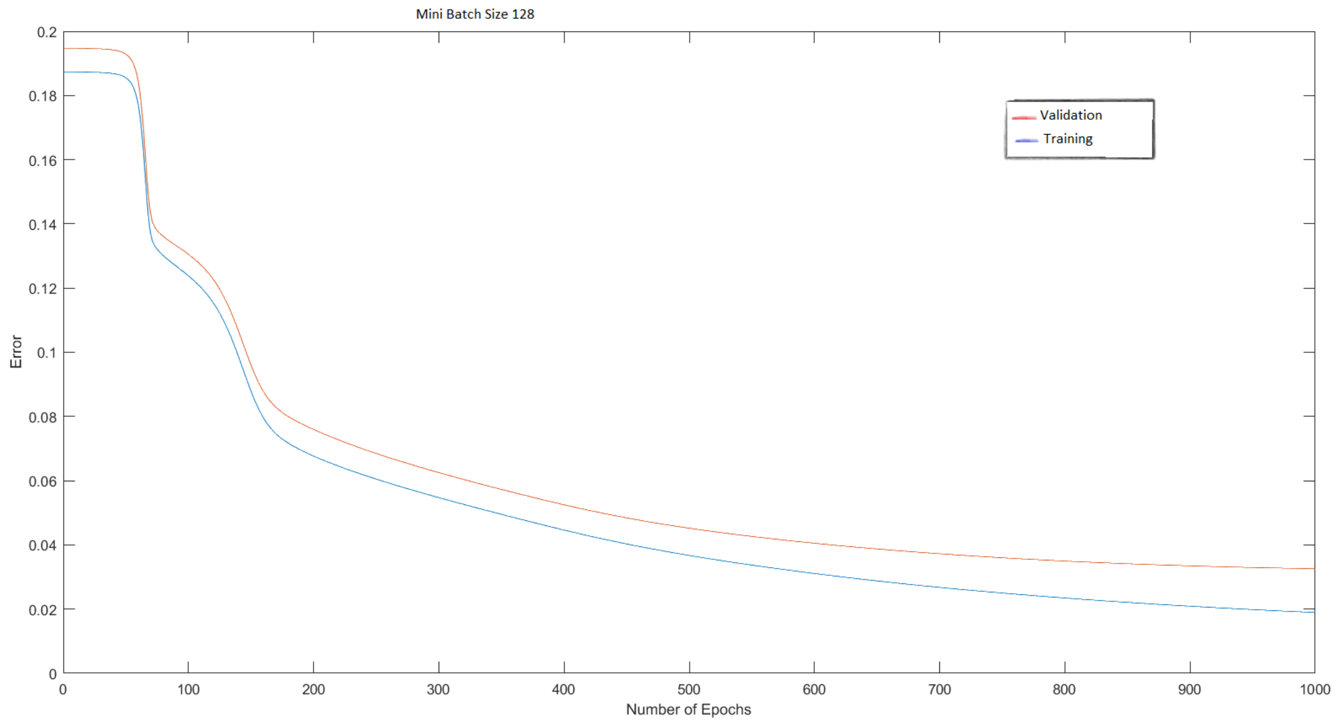


decreases at a faster pace but later on it decreases at slower pace. The training error continuously decreases but the validation error increases after taking a dip. The observation of the fact can be derived from the reasoning that as the number of training epochs increases the model gets overfitted.

2. Shown below is the plot of sum of squares error on the training and validation set as a function of training iterations for 1000 epochs, with learning rate 0.01 and varying mini batch sizes as shown in the figure.
   We can observe that the difference between the training and the validation error decreases after all the training epochs are completed. The reasoning for these observations can be derived from the fact that smaller the batch size, the model fits more to the training set rather than validation set because there are more update equations to minimize the mean squared error. Contradictary we observe that smaller batchsize leads to faster convergence. Hence we have to optimally choose the batchsize consodering both the factors in mind for the good model.




Batchsize = 64
Error
Number of Epochs
training
validation
0
0.02
0.04
0.06
0.08
0.1
0.12
0.14
0.16
0.18
0.2
0
100
200
300
400
500
600
700
800
900
1000


Mini Batch Size 128
Error
Number of Epochs
Validation
Training
0
0.02
0.04
0.06
0.08
0.1
0.12
0.14
0.16
0.18
0.2
0
100
200
300
400
500
600
700
800
900
1000

3. Shown below is the plot of sum of squares error on the training and validation set as a function of training iterations for 1000 epochs, with learning rate 0.01 and mini batch size of 64 and dropout probability of all the three layers namely input, hidden layer 1 and hidden layer 2 as 0.5.
   The training error and validation errors are significantly higher than the above models as the complexity of the model is decreased and model is unable to completely utilize the given information. The training and the validation error keeps fluctuating and do not converge to the minimum.

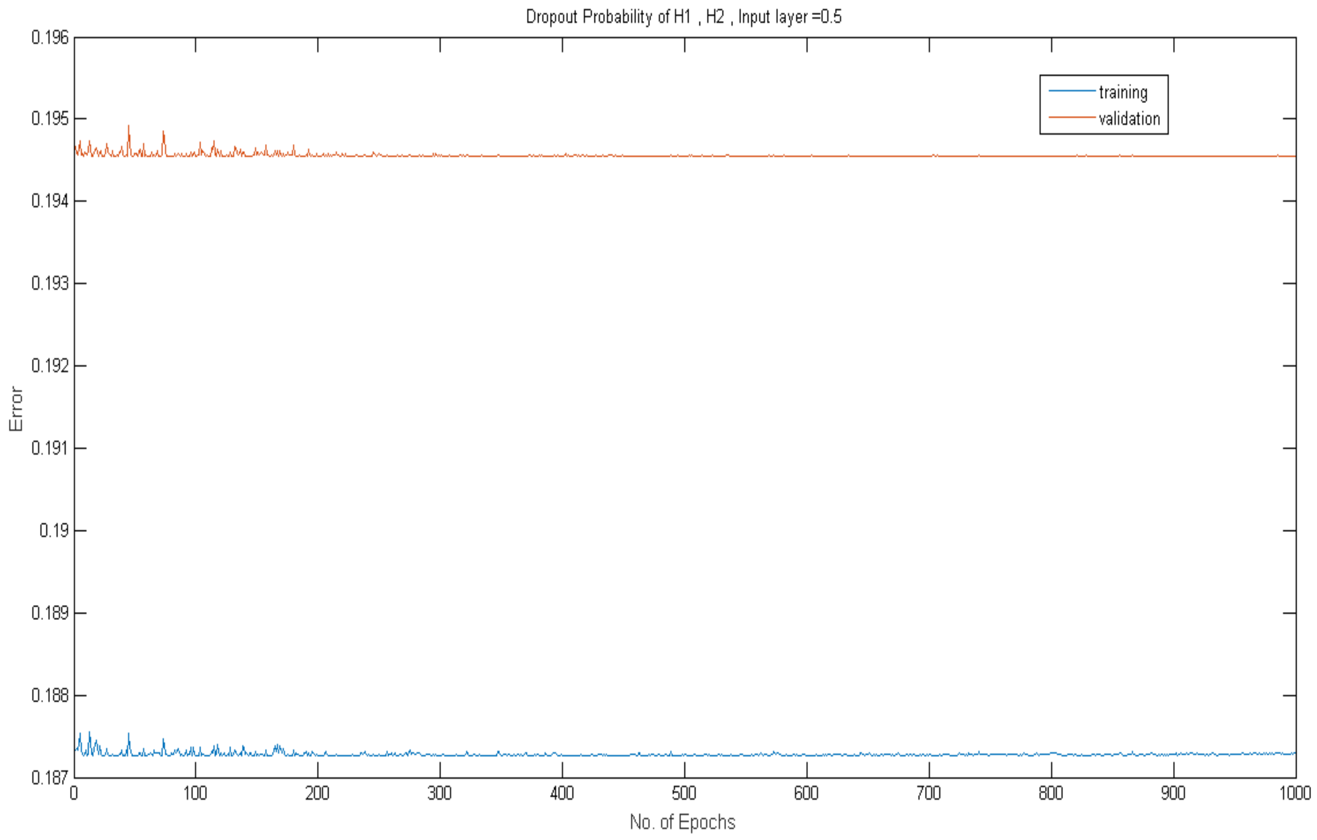
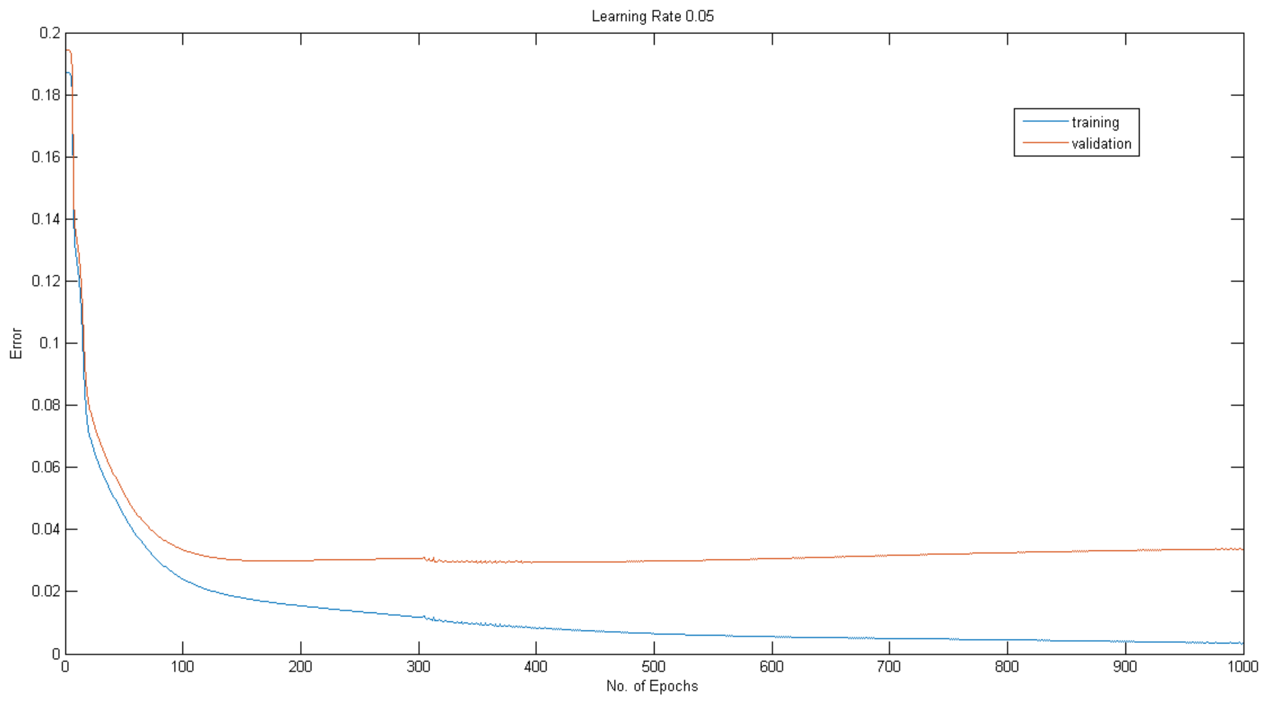
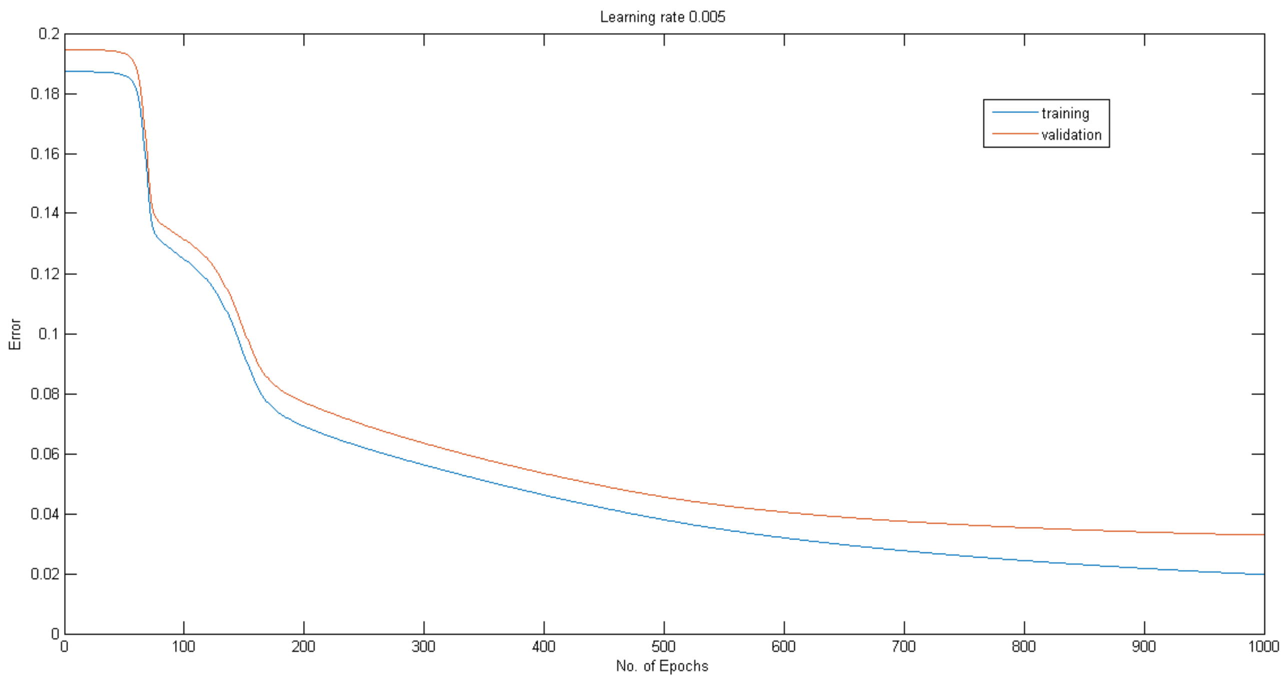
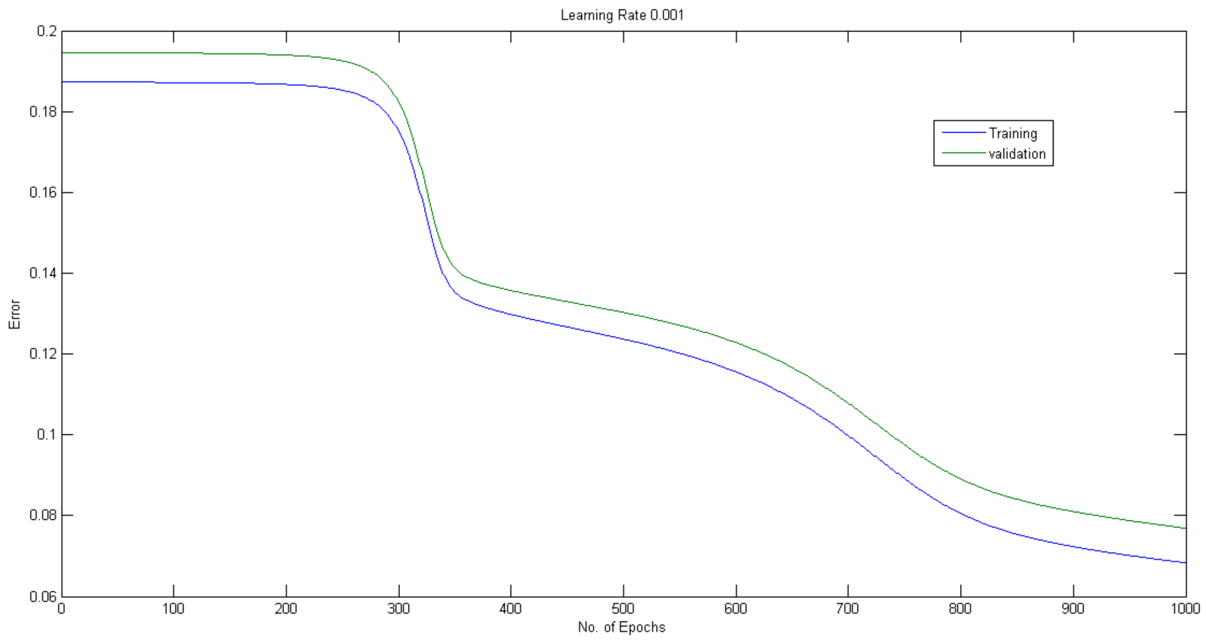


Figure : No. of epochs vs Training and validation error

4. Shown below is the plot of sum of squares error on the training and validation set as a function of training iterations for 1000 epochs, with mini batch size of 64 and varying learning rate as shown in the figure.







From the above graphs we observe that greater the learning rate, the faster is the convergence of the model. But we can also see that as the number of training epochs increases the model gets over fitted in higher learning rate. Hence we should optimally choose the learning rate to get the best model.

{ Error function used for all the error calculations: -1/2N**$\sum(y(k) - o(k))$^2 *}*

COMPETITION PART

INTRODUCTION

In this part, the basic neural network was improved upon to provide better predictions for the test data. The techniques together provided faster convergence rates, better resistance to overfitting and more emphasis on the relevant features. The techniques were implemented such that the validation error decreases as much as possible. Some of the parameters were also guessed via hit and trial method to get the best model. As a part of pre-processing some filters were used to enhance the features of the images so that more information can be retrieved from the given data set.

On experimentation we observed that on increasing the number of hidden layers the model got over fitted hence the validation error increased. On decreasing the mini batch size an optimal model was obtained. As concluded from the above arguments in the question 2 we chose the optimal learning rate by experimenting with different learning rates.

Average filter { $h = 1/32[1]_{32x32}$ } was convolved with the input images by which the validation error decreased significantly.