CSC 578 Section 901
Homework #3
Name: Amy Aumpansub

Documentation

Part 1: Preliminary Steps

My codes passed these steps successfully as all outputs exactly match outputs shown in html file.

Part 2: Application Outputs

I modified my 578hw3.ipynb file to check my implementation of the codes I modified in NN578_network.py for evaluate function, SGD function for calling evaluate function, printing the evaluation results, return them, and add early stopping when reaching the accuracy threshold.

My application codes provide all outputs exactly match with outputs with preliminary step 2 and with both Results-1 and Results-2.txt.

(i) The output from preliminary step 2 with test data also matches with output in 578hw3-checktestset.html in which the output shows MSE, Cross-entropy and log- likelihood, correctcount, and accuracy rate for both training and testing samples from Epoch.

(ii) Results-1.txt matches with my outputs generated from training the net1 network created from iris-423.dat with the SGD function of 100 epochs, mini batch_size=5, eta=0.5 as input parameters. The output shows MSE, Cross-entropy and log- likelihood, correctcount, and accuracy rate for each epoch step from Epoch 0-99.

(iii) Results-2.txt matches with my outputs generated from training the net3 network of sizes [4-20-7-3] created from iris-4-20-7-3.dat with the SGD function of 100 epochs, mini batch_size=5, eta=0.5, stopping accuracy of 0.75 as input parameters. The output shows the MSE, Cross-entropy and log- likelihood, correctcount, and accuracy rate for each epoch step from Epoch 0-62 in which in Epoch#62, we first reached the accuracy of 0.7533 which is above 0.75 so the program terminates the epoch loop.

Part 3: Visualization Results

The matplotlib library was utilized to visualize the error results from each cost function from training the "net4" network of 3 layers [4,20,3] with 50 epochs, eta of 0.1, mini batch size of 5.

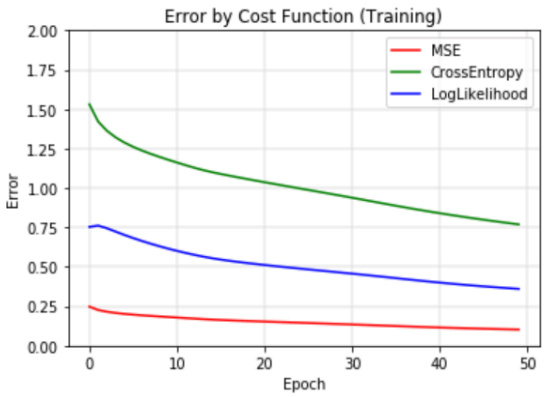
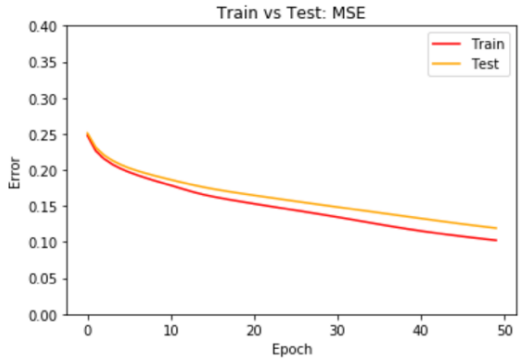


Figure 1

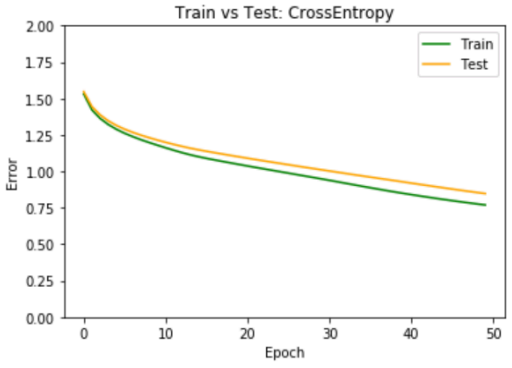
The figure 1 shows the model errors from training data (70% of original dataset) from epoch 0 to 49 (50 steps). Our goal is to smooth and minimize the curve, so MSE performs well as it has the lowest errors and its curve decreases gradually over the steps from 0.3 in epoch 0 to 0.1 in epoch 49. The loglikelihood curve decline slower, so the learning seems slower than cross-entropy. After epoch 40, MSE starts to converge so the learning is almost constant till the last epoch. The loglikelihood curve is always below the cross-entropy curve because it is computed by taking only the output of the target node not the sum of outputs from all nodes like cross-entropy loss.

Figure 2



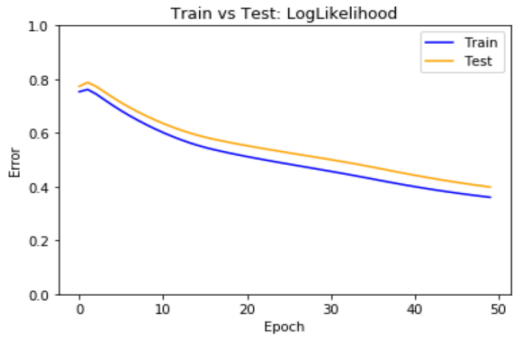
MSE is a mean error of the network across samples (n) calculated as average of the squared differences between target values and activation outputs. MSE ranges from 0.1 to 0.25 and both curves are minimized and smooth. Both curves decrease gradually over 50 epochs, but MSE curve of test set lies above the curve of training set which implies overfitting problem as MSE errors for training data are more minimized than test set beginning at epoch 10. After the training process goes, gap gets bigger, so an effect of overfitting network has more impact on later epochs.

Figure 3



Cross Entropy (CE) has the highest error at a starting epoch, compared to MSE and Loglikelihood. However, CE curves show that errors decrease faster in the beginning of training process between epoch 0 to 10. Cross entropy with sigmoid activation is found to speed up the learning process at the beginning and its curve decreases faster in the beginning epochs. The CE values ranges from 0.75 to 1.6. The errors of test set are slightly higher as the model minimized more errors on the training data which signal some overfitting issue after epoch 10.

Figure 4



Log-Likelihood (LL) curves show that the errors decrease more quickly in the beginning of training process as the slope is steeper between epoch 0 and 10. The curves have a similar pattern with Cross Entropy curves, but the Log-Likelihood curves always lie below the Cross-Entropy curves because LL computes only the output of the target node not the sum of outputs from all nodes. Errors range from 0.4 to 0.8. As process goes beyond epoch 10, the overfitting appears as the gap between curves are wider.

Part 4: Personal Reflection

I enjoyed doing this assignment as I learned how to apply the theories to use in real data with this assignment. I think the difficult level is on average as we were provided with the original codes to modify and professor also walked through this assignment in details, so it helps me understand and better approach the problems. I found the loglikelihood part to be challenging as I need to correctly find the index of the target label from y with no mistakes to derives the correct error values, so I tried with small number of epochs and prints the results out to check if my codes process correctly as I did in preliminary steps. After I passed those steps, I started to write the application codes to call my modified functions in python files and check my outputs with the provided outputs in .txt files (they are so helpful to confirm my formula in codes). I found that I need to review the class slides and videos a few times to correctly solve all problems. Overall, I learned more about the cost functions and neural network process from this assignment.