CS 4375
ASSIGNMENT 2

Names of students in your group:
James Hooper
Hritik Panchasara

Number of free late days used: 0

Note: You are allowed a **total** of 4 free late days for the **entire semester**. You can use at most 2 for each assignment. After that, there will be a penalty of 10% for each late day.

Please list clearly all the sources/references that you have used in this assignment.

For preprocessing data split
https://scikit-learn.org/stable/
For data manipulation
https://pandas.pydata.org/
https://numpy.org/

REPORT

Summarizing the Results

Before going through the log file results, we must clarify what each part of the log means. Initially we list out the three-activation function alongside the number of iterations used & their final error values. This is just a quick summary that occurs as the models are being run. Next we print out this current logs Iterations, Test Size (so .1 equals 90/10 split for Train/Test), Seed 1 which will give us a seed value to determine the state of the train-test-split tool, Seed 2 which will give us a seed value to determine the random seed that will be fed to the creation of the initial weight values, the amount of nodes/neurons for the first hidden layer, and finally the amount of nodes for the second hidden layer. The main reason to have all these values is to have verifiable and repeatable results. After this there are three print outs that describe the train & test accuracy for all three activation functions: Sigmoid, ReLu, and Tanh. The accuracy is represented by two different values. The first value is the percent correct which is just a percentage of how many samples did the network correctly calculate the answer for. For example, the boundaries for rounding in this case would be 0 = 0 <-> .499999 & 1 = .5 <-> 1. The second value is the mean squared error found for the given pass through the trained neural network model This value should be minimized from the training part of each individual model.

A key thing to keep in mind about this Neural Network is that it utilizes the Adam Optimizer, so the learning rate is a constant value called alpha. The Adam parameters are consistent between all logs and goes as follows: alpha = .001, beta1 = .9, beta2 = .999, epsilon = 10**-8, m = 0, v = 0, m_hat = 0, and v_hat = 0. This enhancement is found in the train function for the NeuralNet class. The generic equations for each of the gradients that occurs each iteration goes as follows:

m[i] = (beta1 * m[i]) + ((1 - beta1) * Gradients[i])
v[i] = (beta2 * v[i]) + ((1 - beta2) * (Gradients[i] ** 2))
m_hat[i] = m[i]/(1 - beta1)
v_hat[i] = v[i]/(1 - beta2)
updateGrads[i] = (((alpha)/(np.sqrt(v_hat[i]) + epsilon)) * m_hat[i])

From the logs we gave/found the consensus is that the best parameters are: Iterations = 9000, Test Size = .1, Hidden Layer 1 nodes = 4, and Hidden Layer 2 nodes = 8. Each time we tested different states with these parameters we got 100% test accuracy for each activation function. In fact, this entire process of tuning we found it easy to find parameters that enable such high accuracy. The only time we had poor results is when it felt like we were purposefully making the model bad to see the differences between activation functions. This occurs with parameters: Iterations = 1000, Test Size = .3, Hidden Layer 1 nodes = 1, and Hidden Layer 2 nodes = 1. Now with a more normal set of parameters the worse we would get for this dataset is about 98-99% accuracy for both training and testing predictions. The last log in the file is ran at an extreme amount of iterations to just showcase how low the error could go for each activation function. From these results there are two main things to takeaway from the use of this dataset & the Adam Optimizer. First, in the very worst log case, the best performing activation function without a doubt was Tanh with the worst being ReLu. Even in very poor conditions Tanh performed exceedingly well. Second, the best log found is probably overkill for this problem. It is very likely there is a set of parameters that optimize computation time. An example of this is from the log with parameters: Iterations = 1000, Test Size = .1, Hidden Layer 1 nodes = 3, and

Hidden Layer 2 nodes = 3. In this case all three activation functions performed with a train/test accuracy of above 95%, with Tanh having the best overall. In this example it is clear that the use of extremities such as 9000 iterations or 8 hidden layer nodes may not be optimal and in fact overkill even if they give very accurate results. Again, this would be a decision that would have to be analyzed further to fully understand if differing states affects the accuracy for the lesser complex models. It could also be inferred that the more data given for this dataset may end up showing that the more complex models perform more accurately more consistently.