CSE 326 **Due on April 11**

Problem Set IV Gustavo Grinsteins CSE 326

Two Layer Neural Network Design (CSE 326 (60%)**)**

II. Analysis (CSE 326 (40%)**)**

Run the tests on both 20ng and MNIST dataset and report the results. For each of the output, make sure to include the graph and describe in a short paragraph the results about what you see and whether they make sense from the perspective of what you expect things like regularization and loss functions to do.

20ng Dataset

- (10%) Try your neural network with no regularization parameter λ and (2, 5, 10) hidden nodes. Plot performance (both train and test) as a function of iteration.

Please refer to Figures 1-3.

*The model over fits as you increase the hidden nodes. This results in higher train values and lower test values. In other words, the more hidden nodes you have the more complexity therefore the more likely you are to over fit.

- (10%) Try different learning rates and plot the performance (both train and test) as a function of iteration.

Please refer to Figures 4-7.

*Too large values for the learning rate might overshoot the minimum and go to really large values.

*Too small values might get us stuck in undesirable minimum that will give bad results.

- (10%) Vary the regularization parameter with the number of iterations fixed and plot the performance.

Please refer to Figures 8-12.

*Note that the learning and training curves come closer together due to the regularization. The regularization puts a penalty on large weights that forces the model to not over fit the data.

MNIST Dataset

- (10%) Try your neural network with no regularization parameter λ and (2, 5, 10) hidden nodes. Plot performance (both train and test) as a function of iteration.

Please refer to Figures 13-15.

*Same pattern as the 20ng data happens.

- (10%) Try different learning rates and plot the performance (both train and test) as a function of iteration.

Please refer to Figures 16-18.

*Same pattern as the 20ng data happens.

- (10%) Vary the regularization parameter with the number of iterations fixed and plot the performance.

Please refer to Figures 19-21.

*Similar pattern as the 20ng data happens.

Summary of the results for MNIST and 20ng

The train and test accuracy curves for the MNIST data set where closer together in general than the 20ng data set. This might be due to the variability in the data. The grey scale values in MNIST range from 0 to 250 while the values in 20ng range from 0 to 1000s.

(10%) Answer questions according to your experiment results: Would you prefer to regularize or just stop optimizing early? Which seems more stable? Is the two-layer network doing better than the single-layer network? Include answers to all these questions in your writeup.

Stopping the algorithm at a given iteration would be better than keep the algorithm going until it converges. Waiting for the algorithm to converge can be time consuming and also take a lot of computing power. Therefore, this is why I think stopping early is more stable.

The two-layer network performs better than the single-layer network in real world applications since most data is not linearly separable. It is important to notice that two layer network adds more complexity to our system therefore possibly over fitting. In the case of linearly separable data.

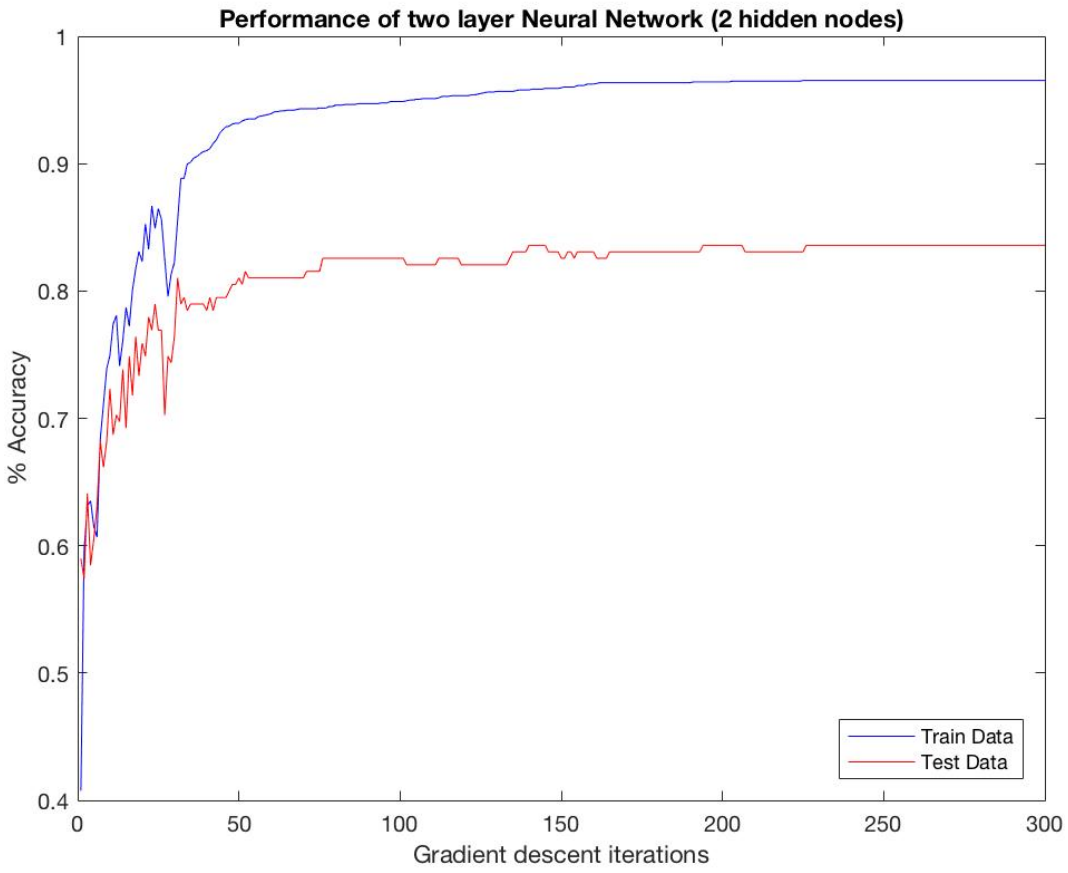

Figure 1: 20NG: Plot of accuracy (train and test) as a function of gradient descent iteration. 2 layer Neural Network with no regularization parameter λ. Final accuracy in test and train: 83% and 96% respectively.

Results from the 2 layer Neural Network 20ng Data plots figures 1-12 and MNIST Data plots figures 13-21

The figure number that corresponds to each task are specified for each problem for each data set. Please refer to these above.

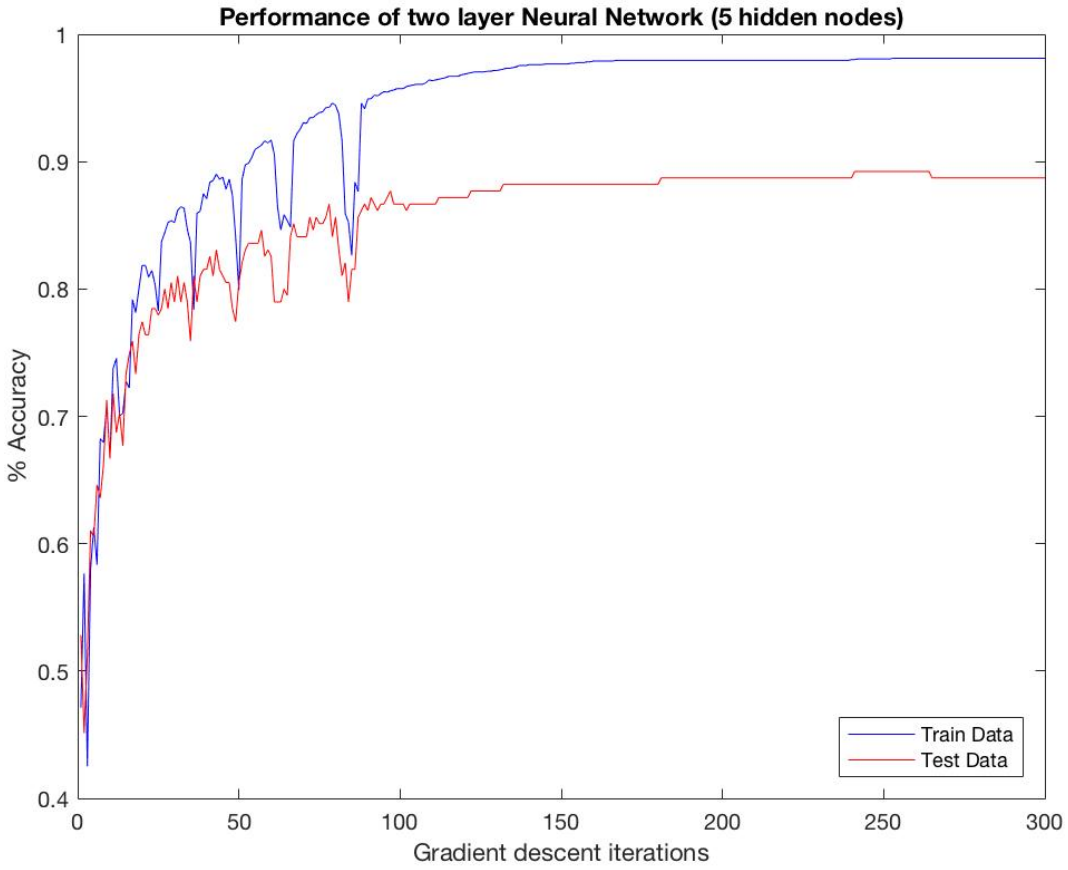

Figure 2: 20NG: Plot of accuracy (train and test) as a function of gradient descent iteration. 5 layer Neural Network with no regularization parameter λ. Final accuracy in test and train: 89% and 98% respectively.

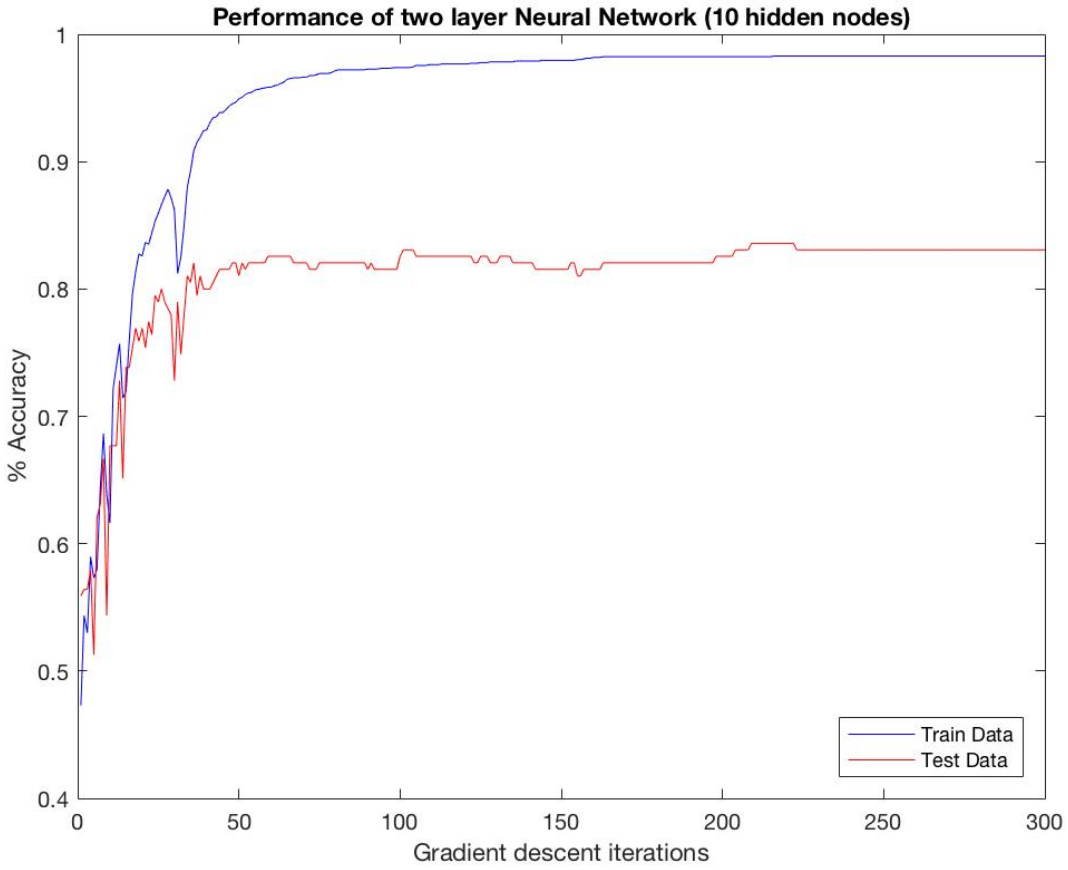

Figure 3: 20NG: Plot of accuracy (train and test) as a function of gradient descent iteration. 10 layer Neural Network with no regularization parameter λ. Final accuracy in test and train: 83% and 98% respectively.

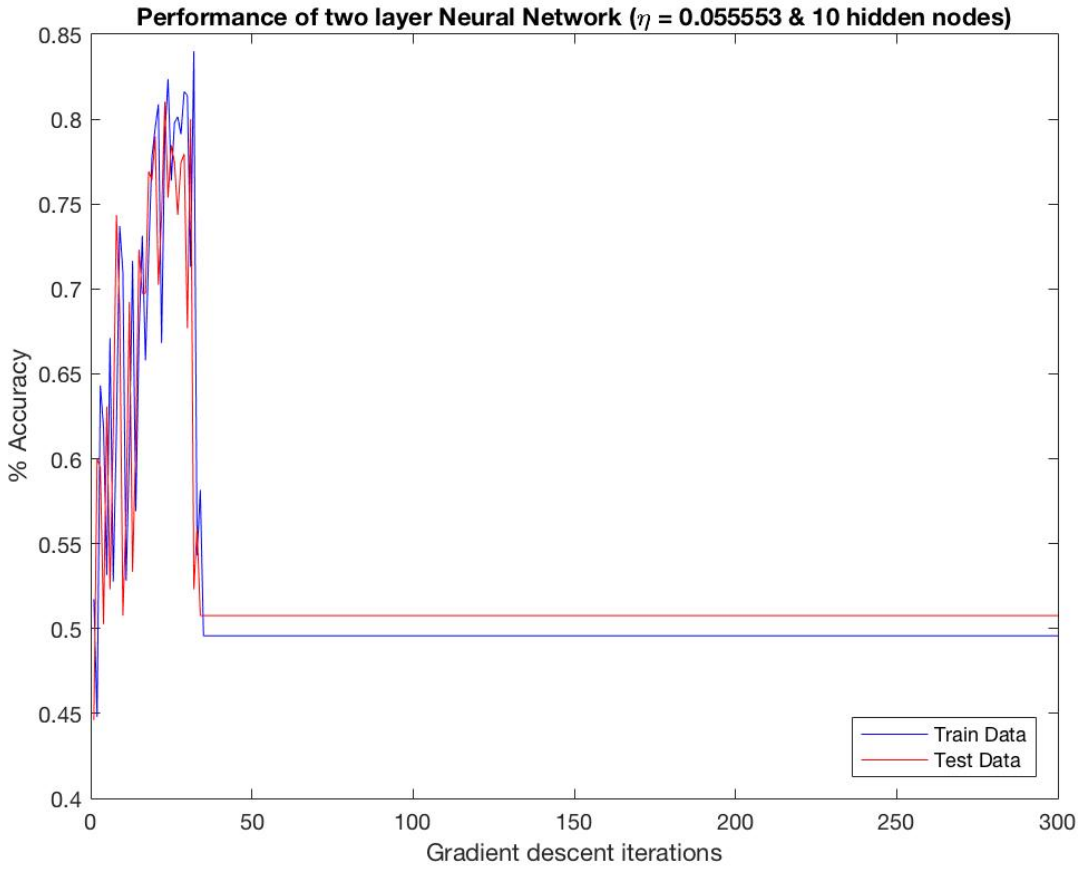

Figure 4: 20NG: Plot of percent accuracy with learning rate of 0.055553, 10 hidden nodes. all against gradient descend iterations (both train and test) as a function of iteration. This relatively large learning rate results in non-reliable values.

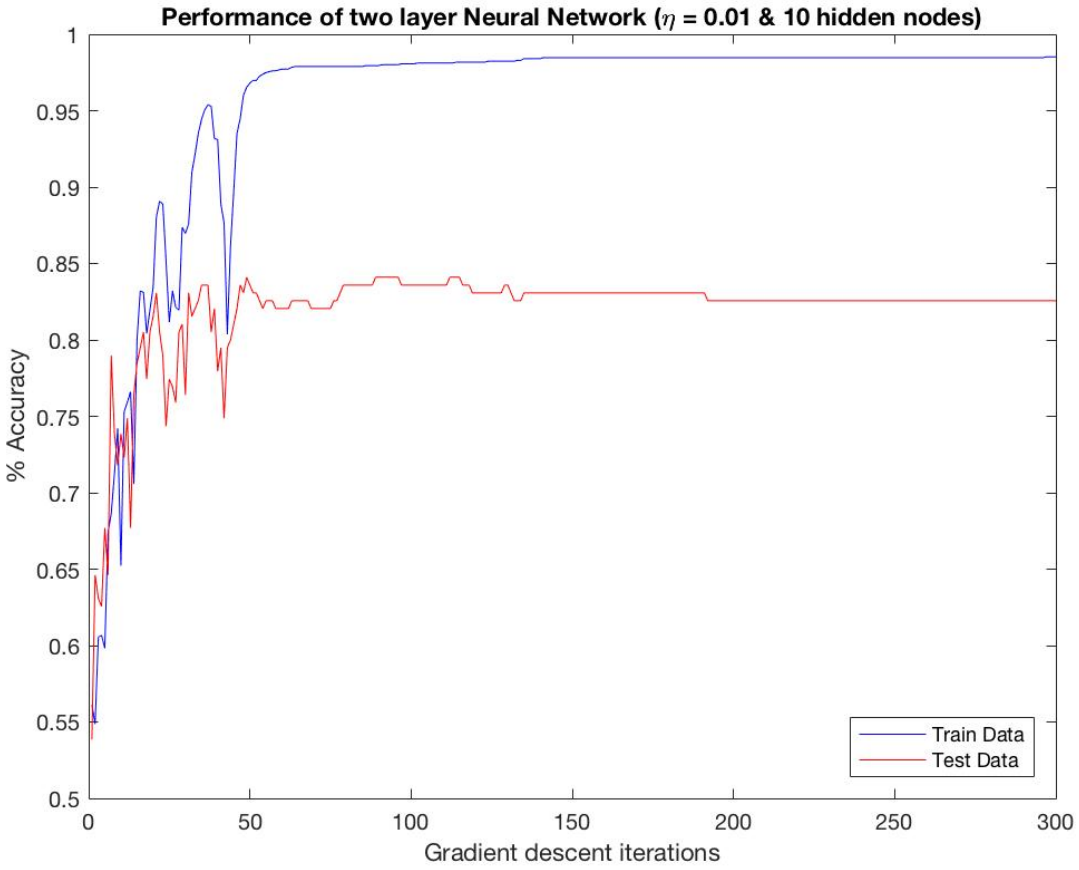

Figure 5: 20NG: Plot of percent accuracy with learning rate of 0.01, 10 hidden nodes. all against gradient descend iterations (both train and test) as a function of iteration. Final accuracy in test and train: 82% and 98.5% respectively.

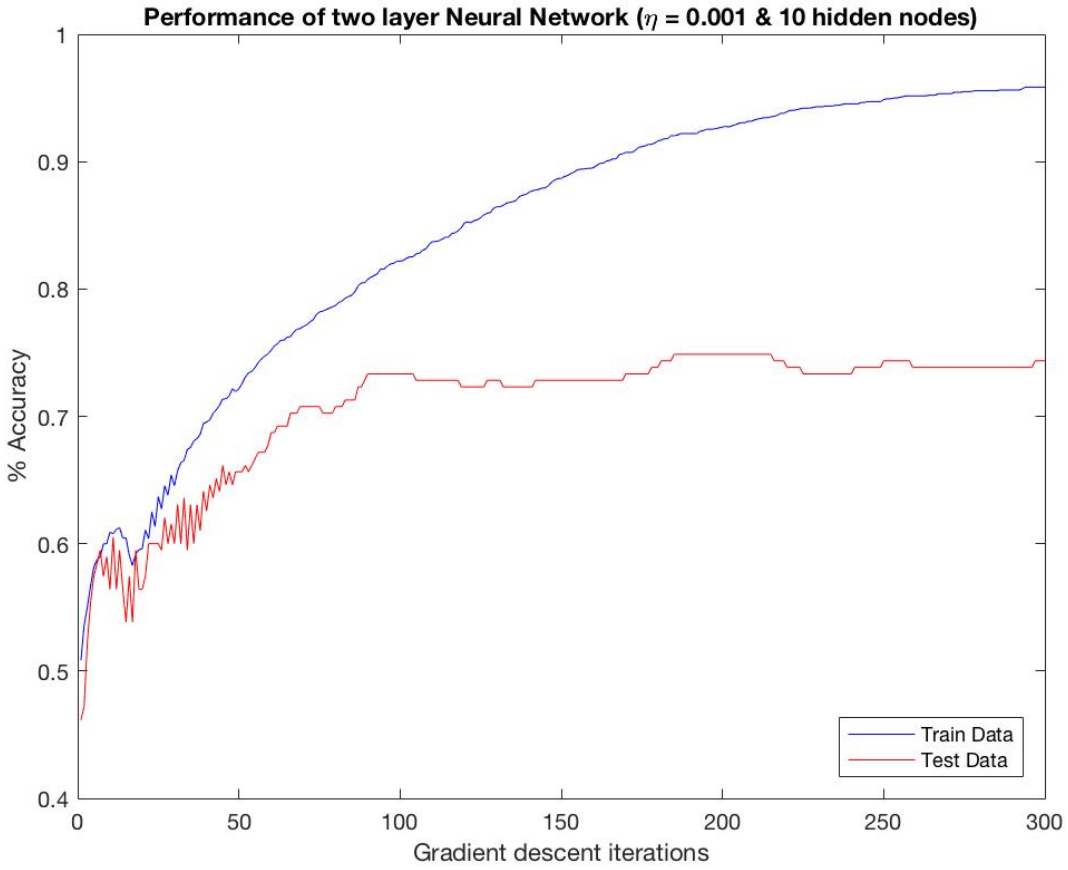

Figure 6: 20NG: Plot of percent accuracy with learning rate of 0.001, 10 hidden nodes. all against gradient descend iterations (both train and test) as a function of iteration. Final accuracy in test and train: 74% and 95.8% respectively.

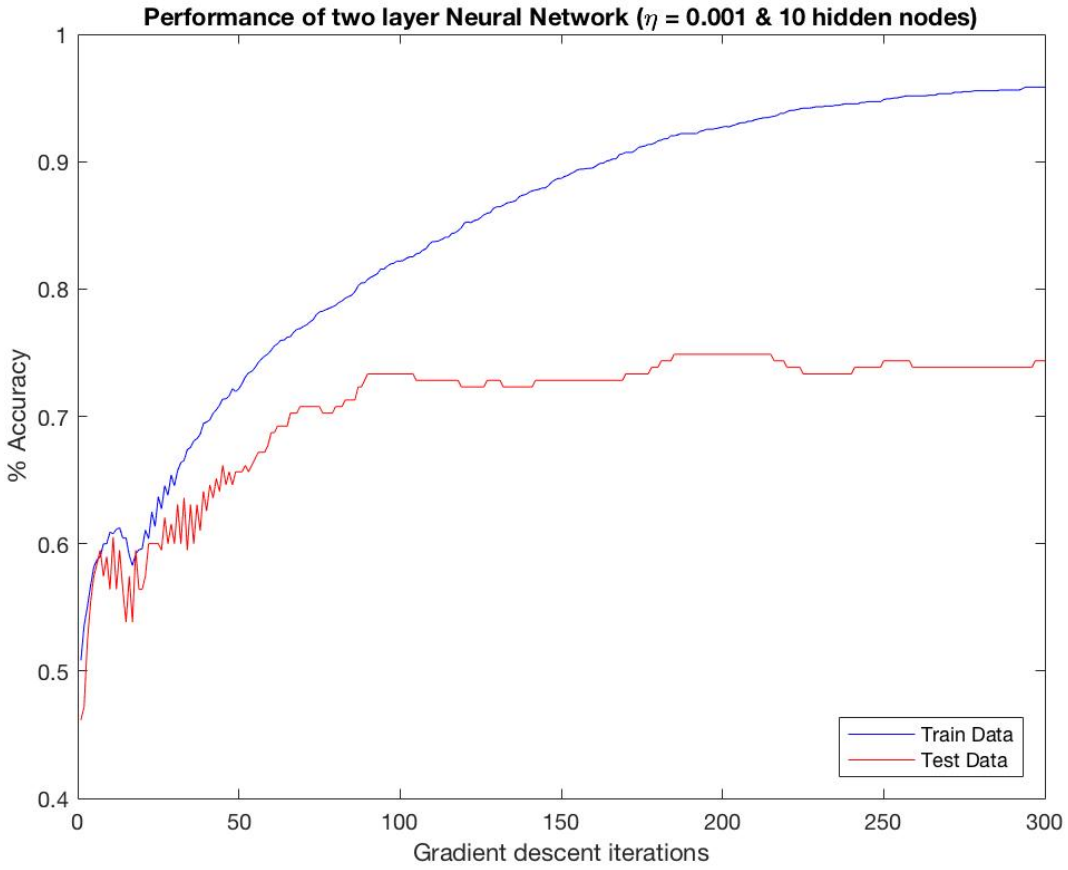

Figure 7: 20NG: Plot of percent accuracy with learning rate of 0.001, 10 hidden nodes. all against gradient descend iterations (both train and test) as a function of iteration. Final accuracy in test and train: 74% and 95.8% respectively.

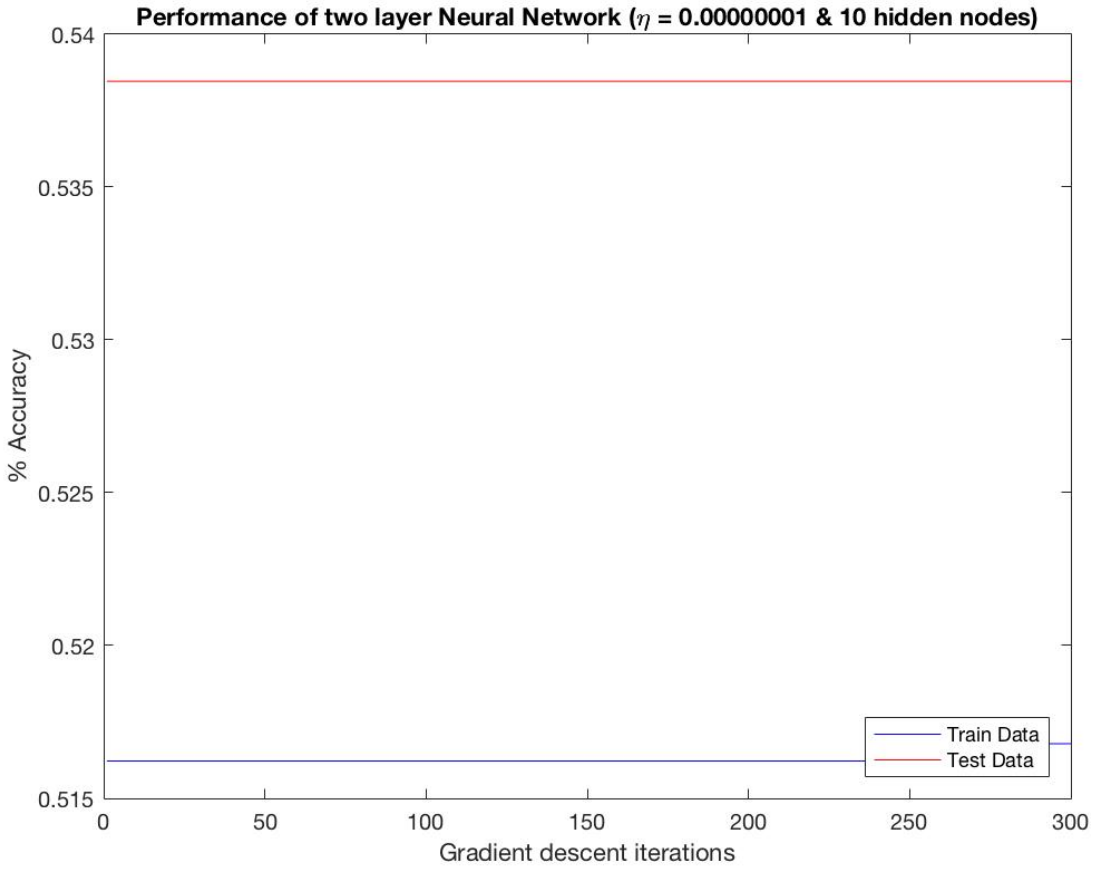

Figure 8: 20NG: Plot of percent accuracy with learning rate of 0.00000001, 10 hidden nodes. all against gradient descend iterations (both train and test) as a function of iteration. At very low learning rates the optimization might get stuck on non-optimal local minima producing non-reliable classifications

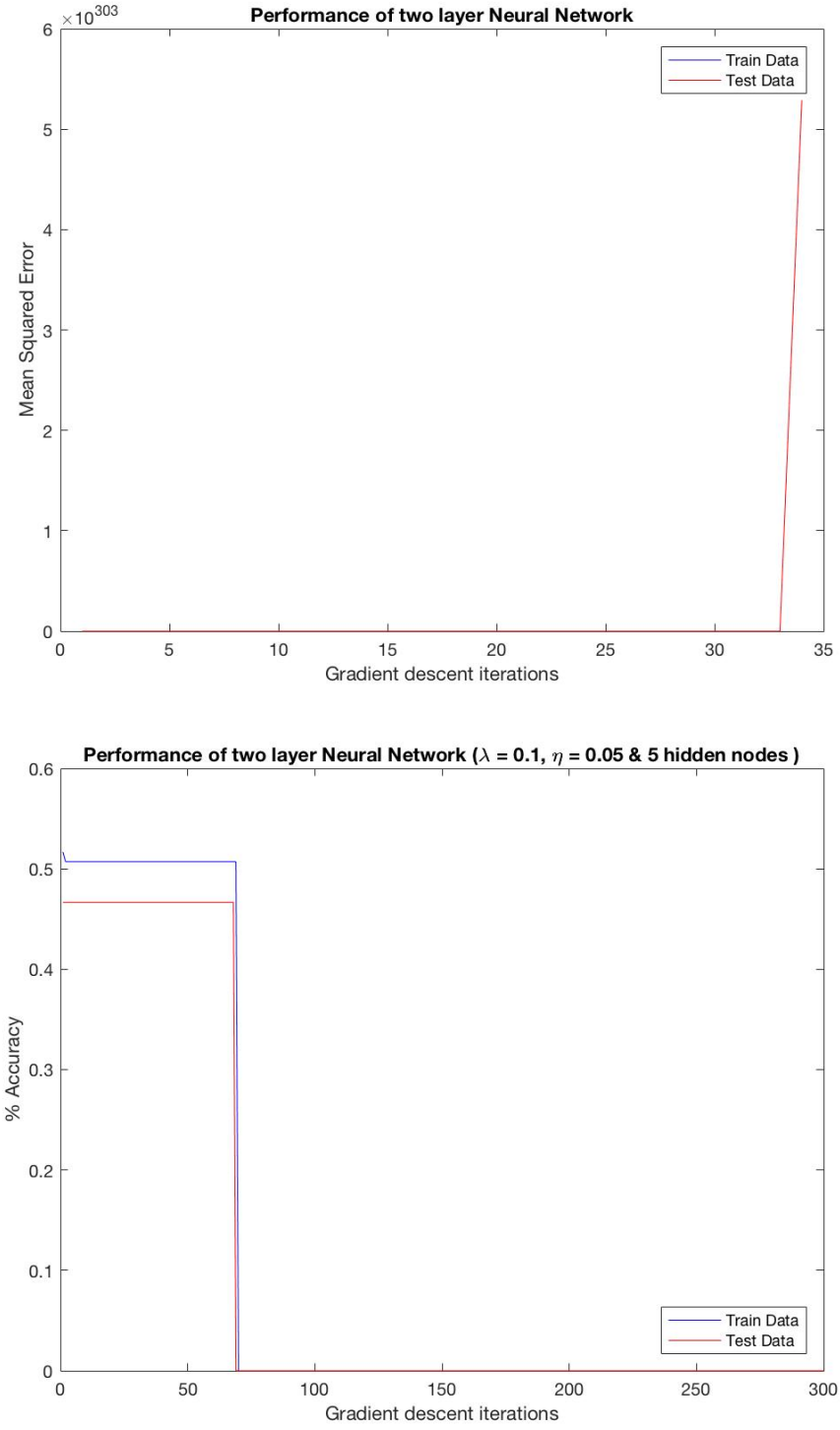

Figure 9: 20NG: Plot of mean square error and plot with the regularization parameter $\lambda = 0.1$ with the number of iterations fixed. Note that with this lambda value the MSE function is not minimized and rises in value really quickly. This produces poor classifications

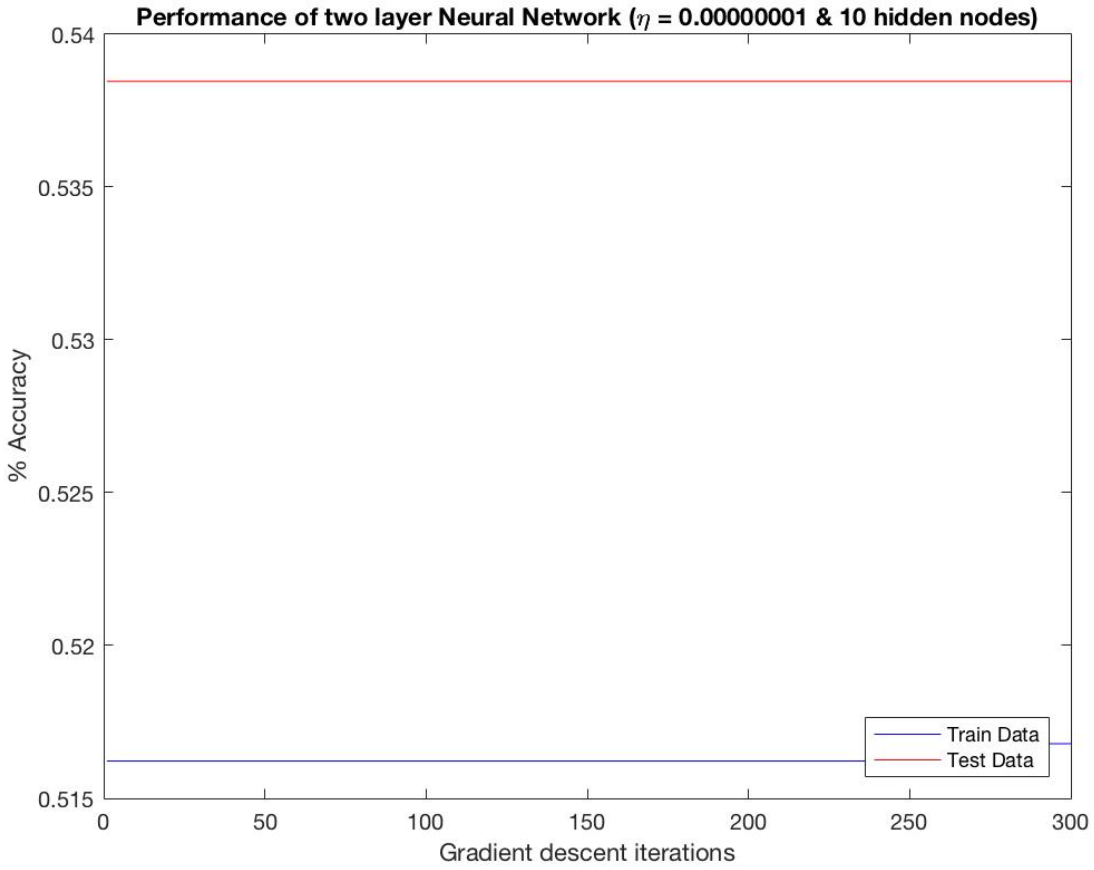

Figure 10: 20NG: Plot with the regularization parameter $\lambda = 0.01$ with the number of iterations fixed. Note that with this lambda value the MSE function is not minimized and rises in value really quickly. This produces really poor classifications for the neural net

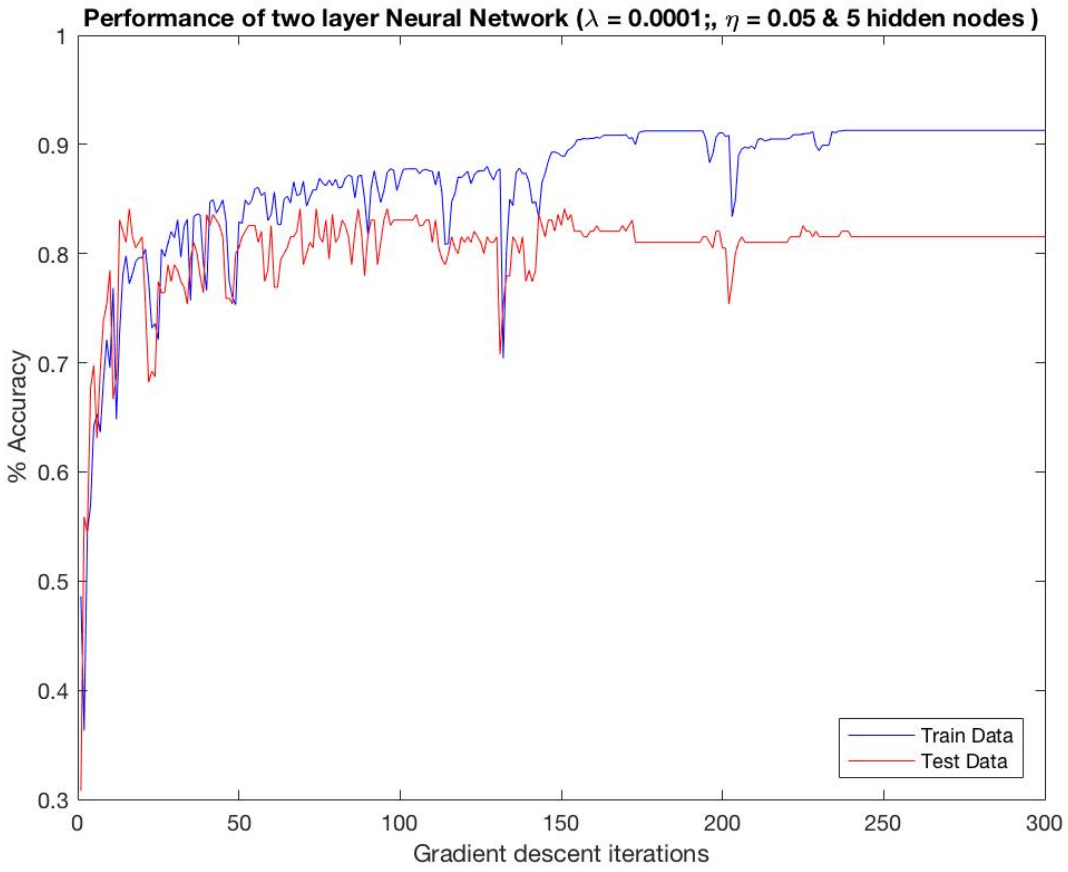

Figure 11: 20NG: Plot with the regularization parameter $\lambda = 0.0001$ with the number of iterations fixed. Final accuracy in test and train: 81.5% and 91.3% respectively.

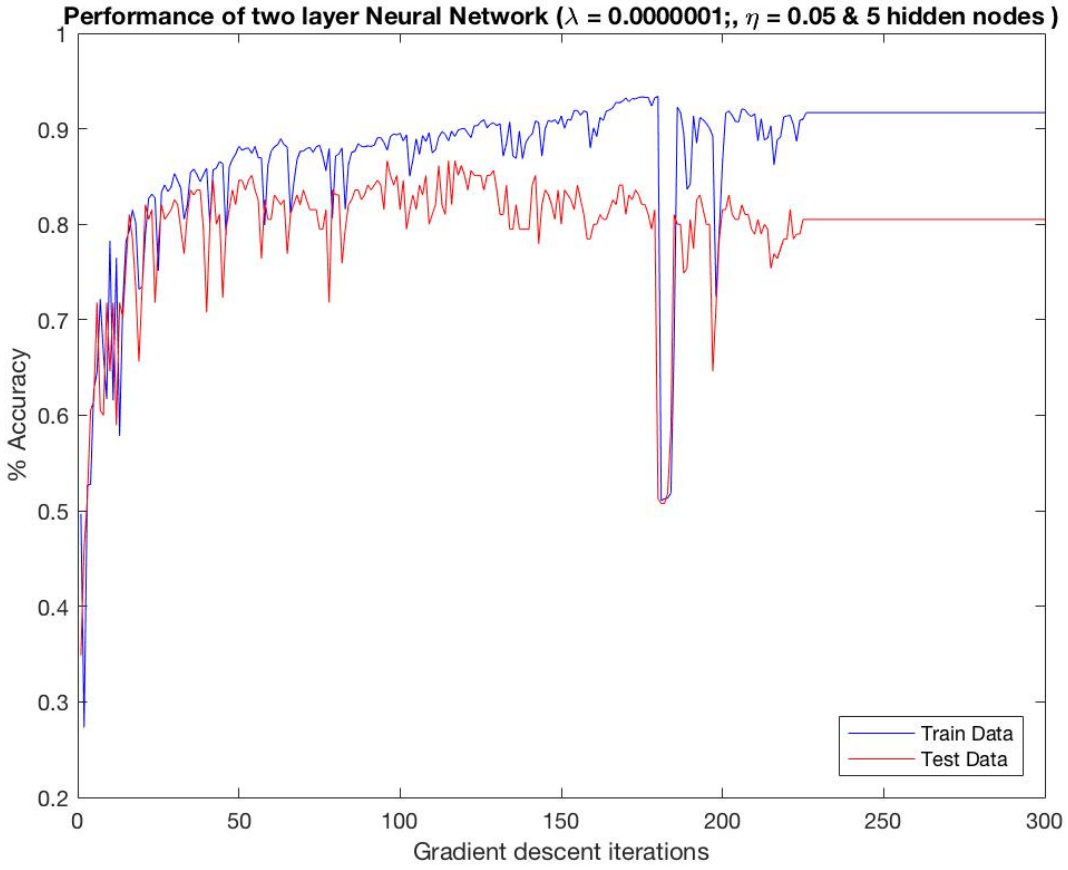

Figure 12: 20NG: Plot with the regularization parameter $\lambda = 0.0001$ with the number of iterations fixed. Final accuracy in test and train: 80.5% and 91.7% respectively.

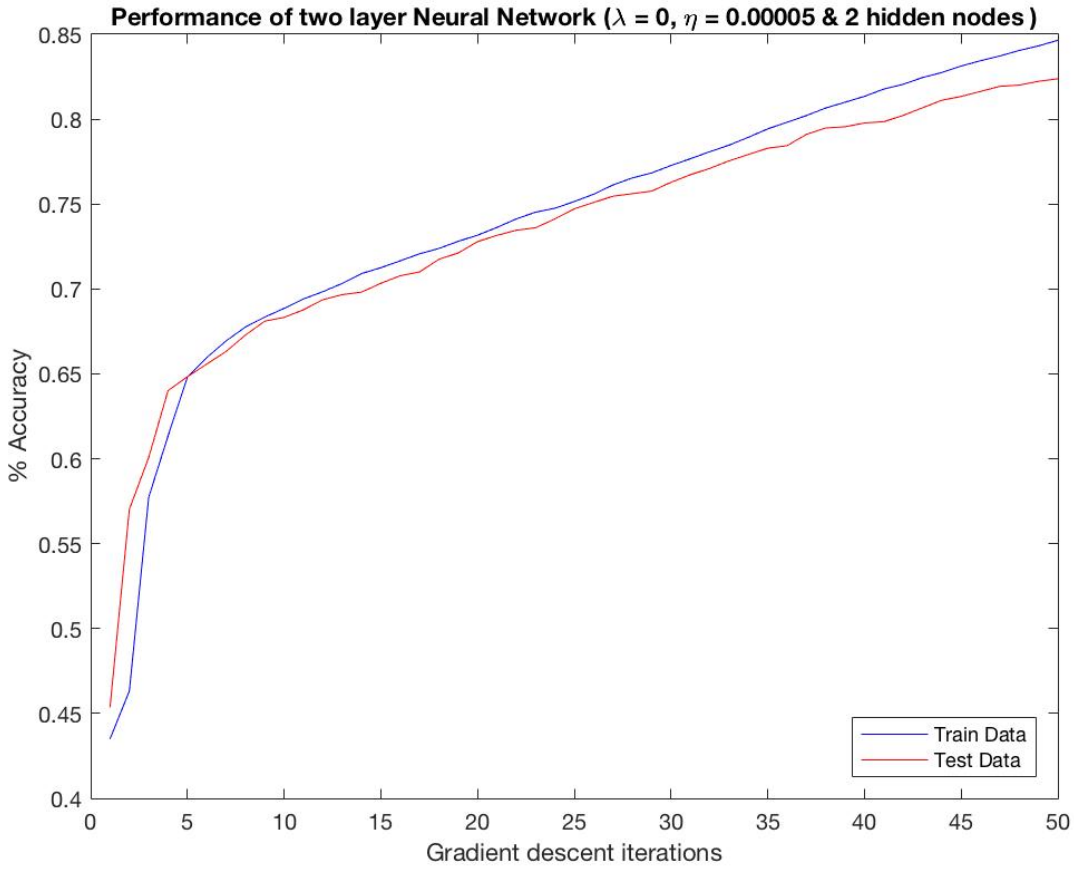

Figure 13: MNIST: Plot of accuracy (train and test) as a function of gradient descent iteration. 2 layer Neural Network with no regularization parameter λ. Final accuracy in test and train: 82.3% and 85% respectively.

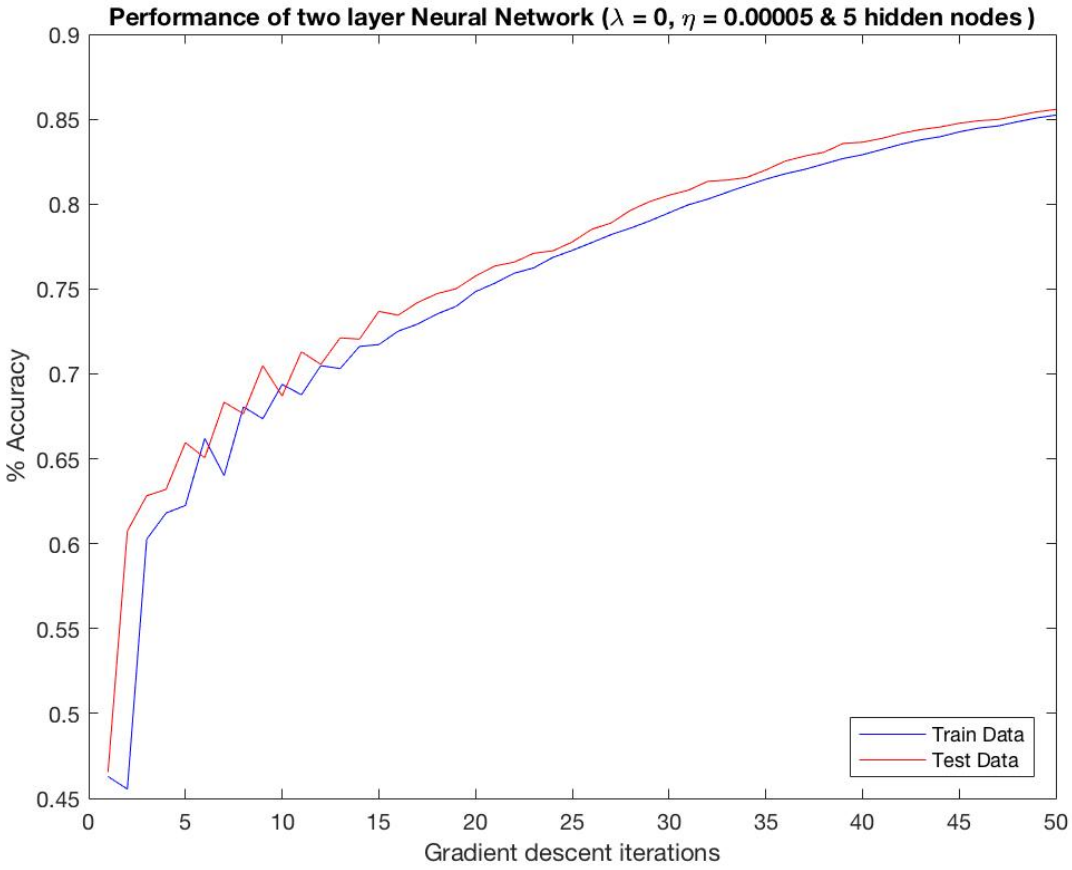

Figure 14: MNIST: Plot of accuracy (train and test) as a function of gradient descent iteration. 5 layer Neural Network with no regularization parameter λ. Final accuracy in test and train: 85% and 85.2% respectively.

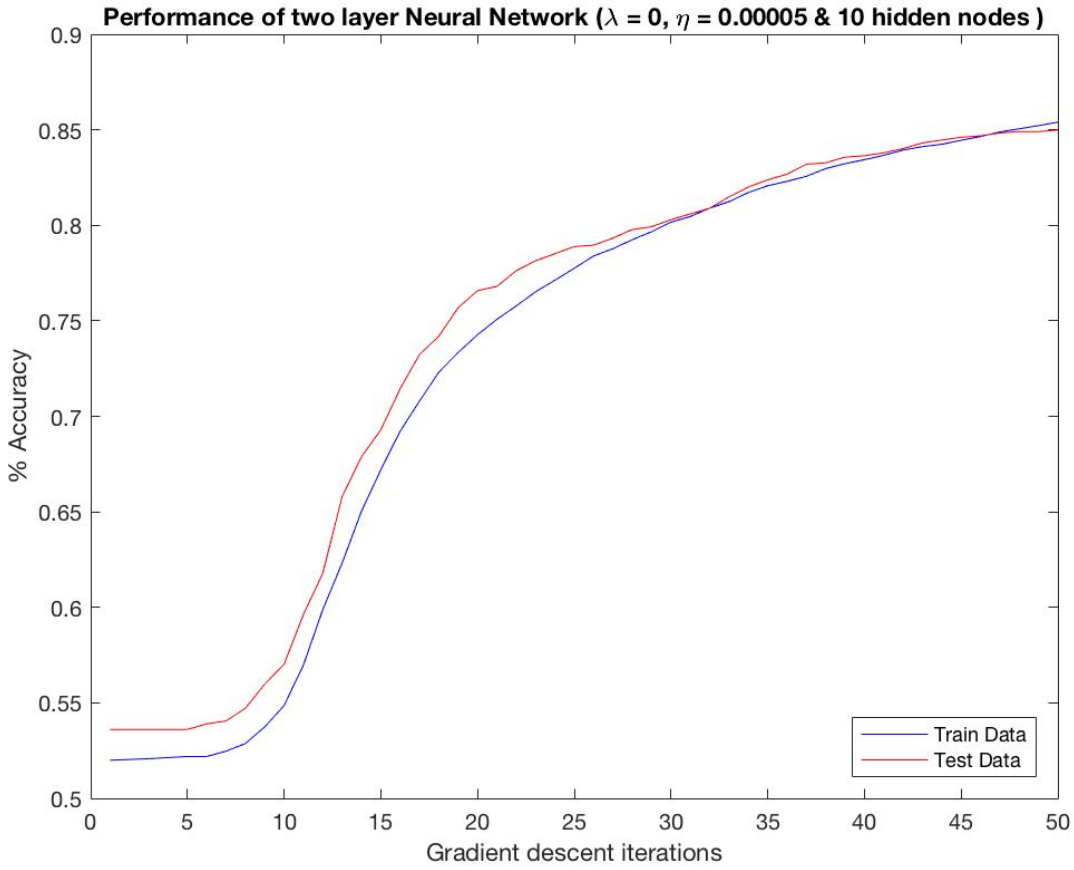

Figure 15: MNIST: Plot of accuracy (train and test) as a function of gradient descent iteration. 10 layer Neural Network with no regularization parameter λ. Final accuracy in test and train: 85% and 85.4% respectively.

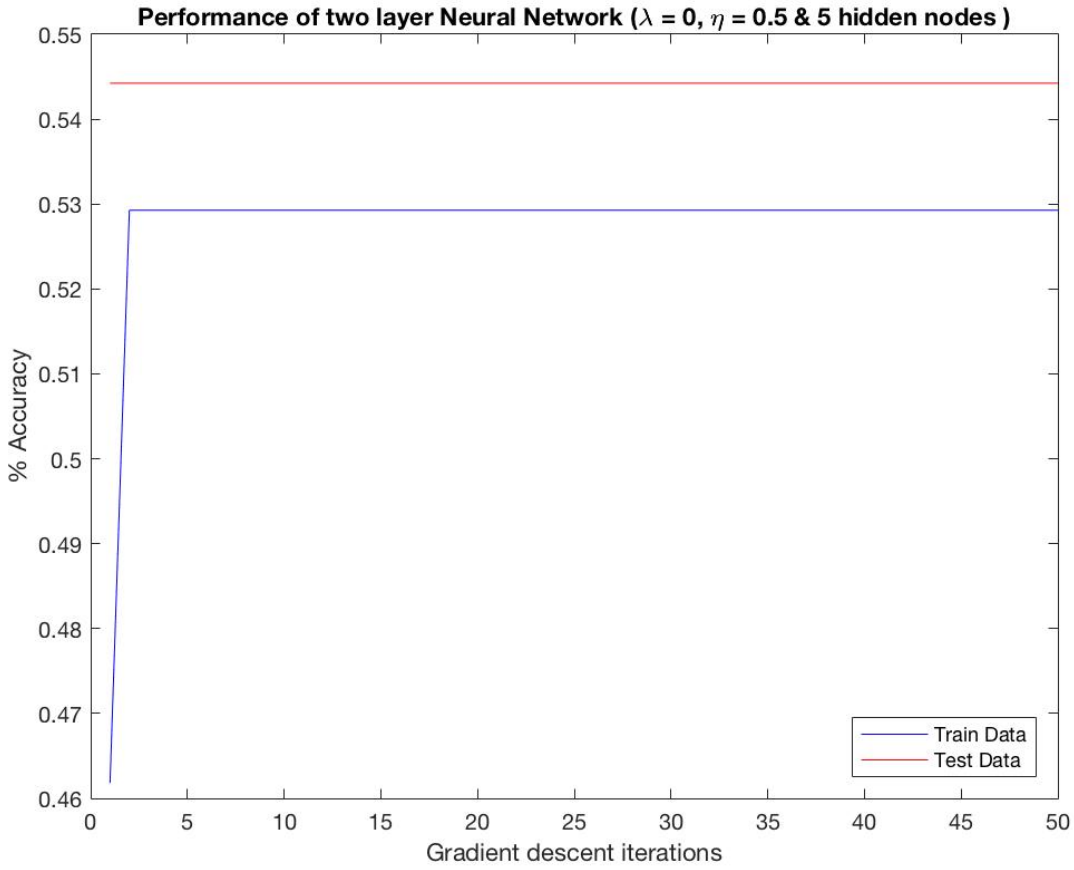

Figure 16: MNIST: Plot of percent accuracy with learning rate of 0.5, 5 hidden nodes. all against gradient descend iterations (both train and test) as a function of iteration. This relatively large learning rate results in non-reliable values.

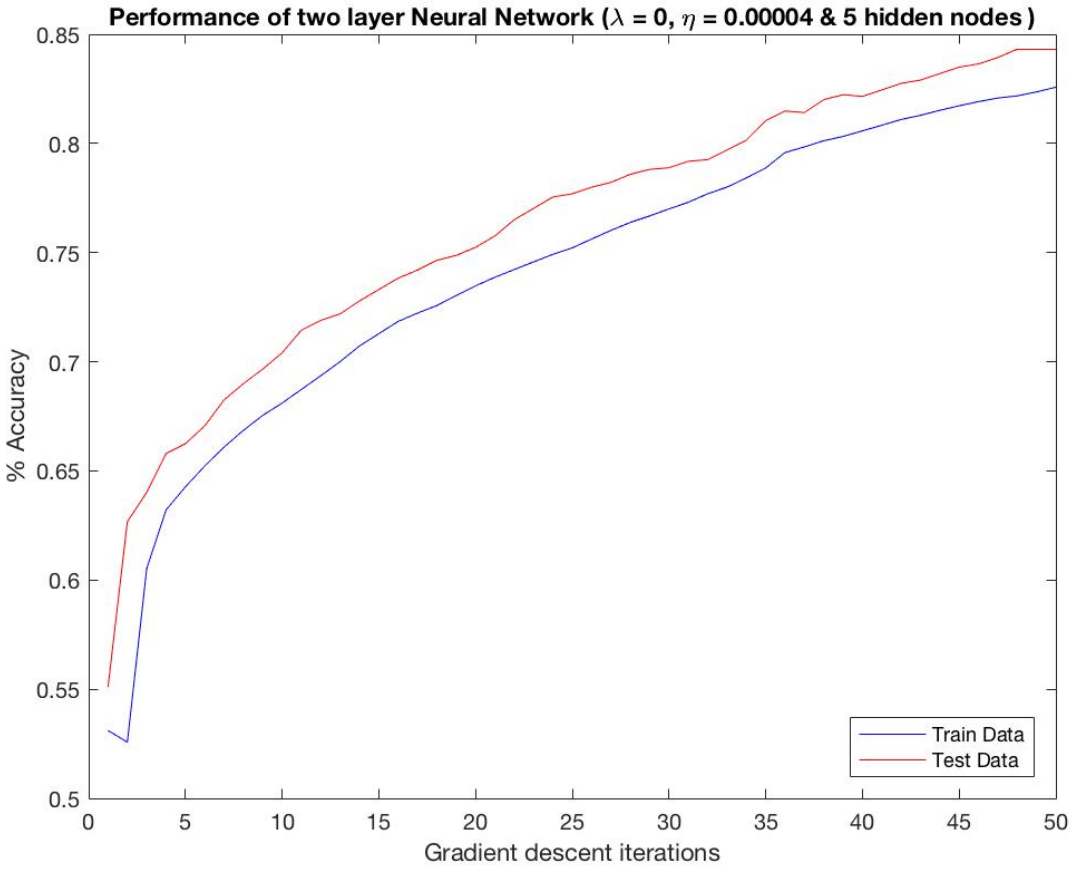

Figure 17: MNIST: Plot of percent accuracy with learning rate of 0.00004, 5 hidden nodes. all against gradient descend iterations (both train and test) as a function of iteration. Final accuracy in test and train: 84.3% and 82.5% respectively. Note that the test accuracy is higher than the train and this is unusual

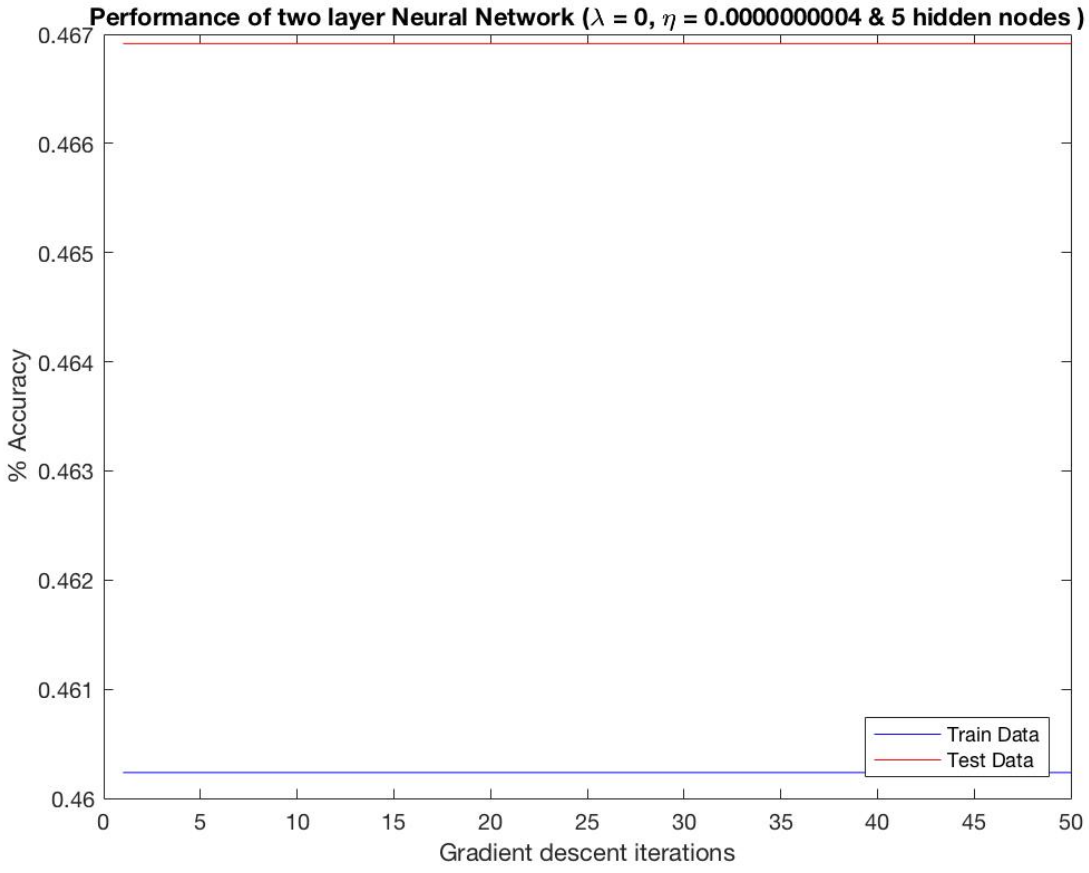

Figure 18: MNIST: lot of percent accuracy with learning rate of 0.0000000004, 5 hidden nodes. all against gradient descend iterations (both train and test) as a function of iteration. At very low learning rates the optimization might get stuck on non-optimal local minima producing non-reliable classifications

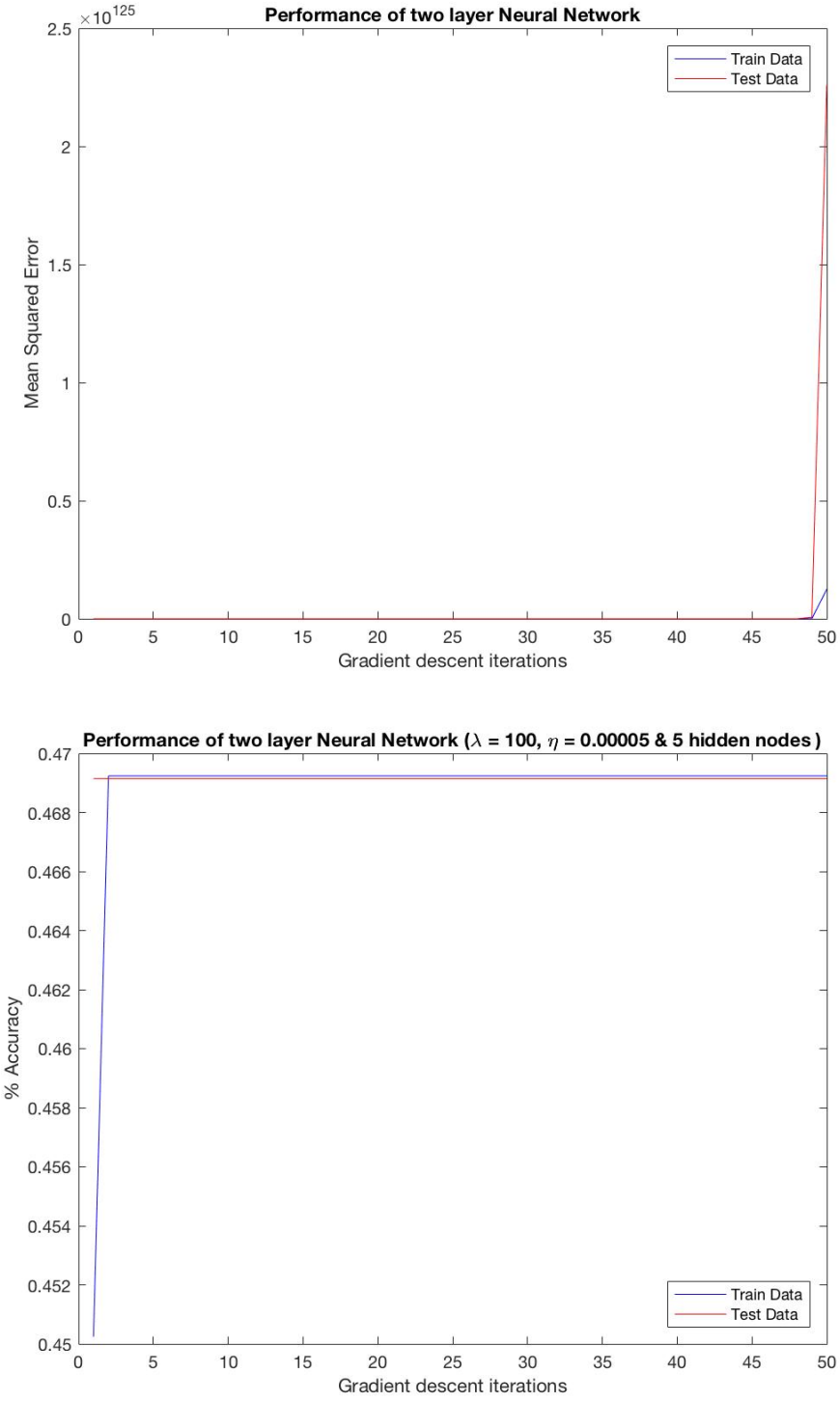

Figure 19: MNIST: Plot of mean square error and plot with the regularization parameter $\lambda = 100$ with the number of iterations fixed. Note that with this lambda value the MSE function is not minimized and rises in value really quickly. This produces poor classifications

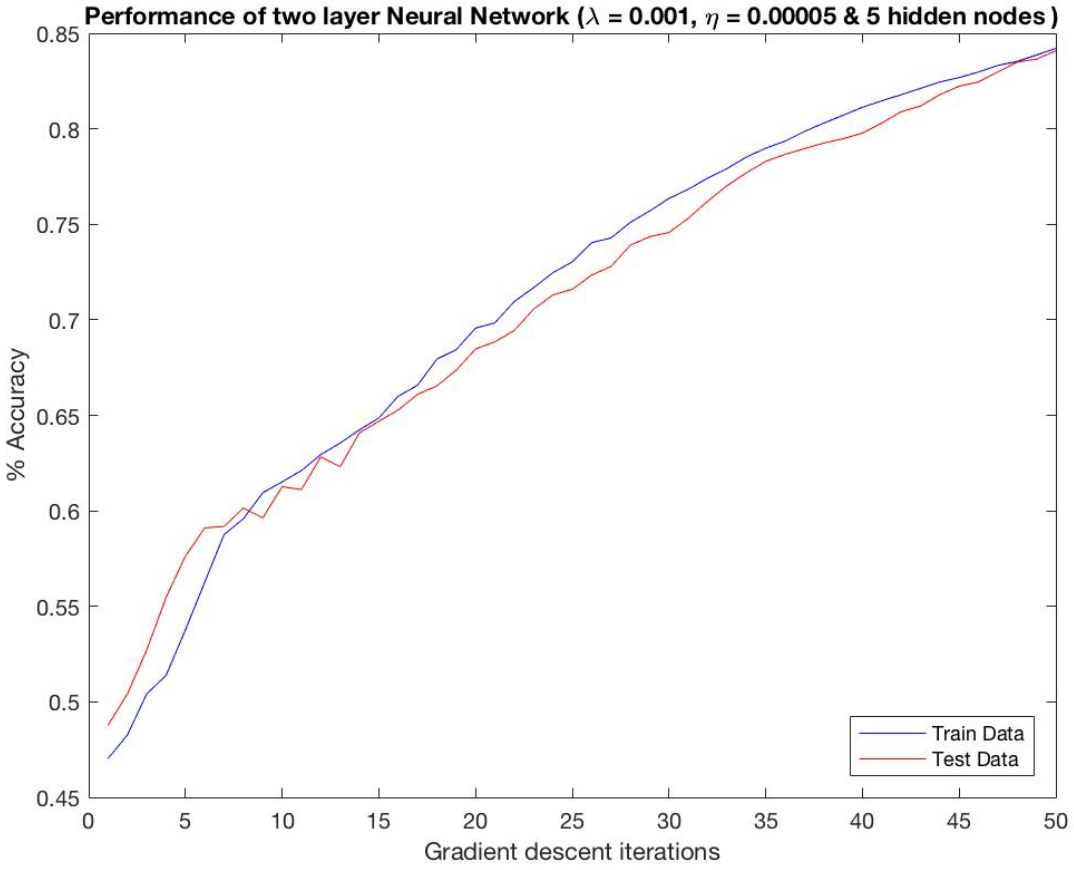

Figure 20: MNIST: Plot with the regularization parameter $\lambda = 0.001$ with the number of iterations fixed. Final accuracy in test and train: 84.1% and 84.2% respectively.

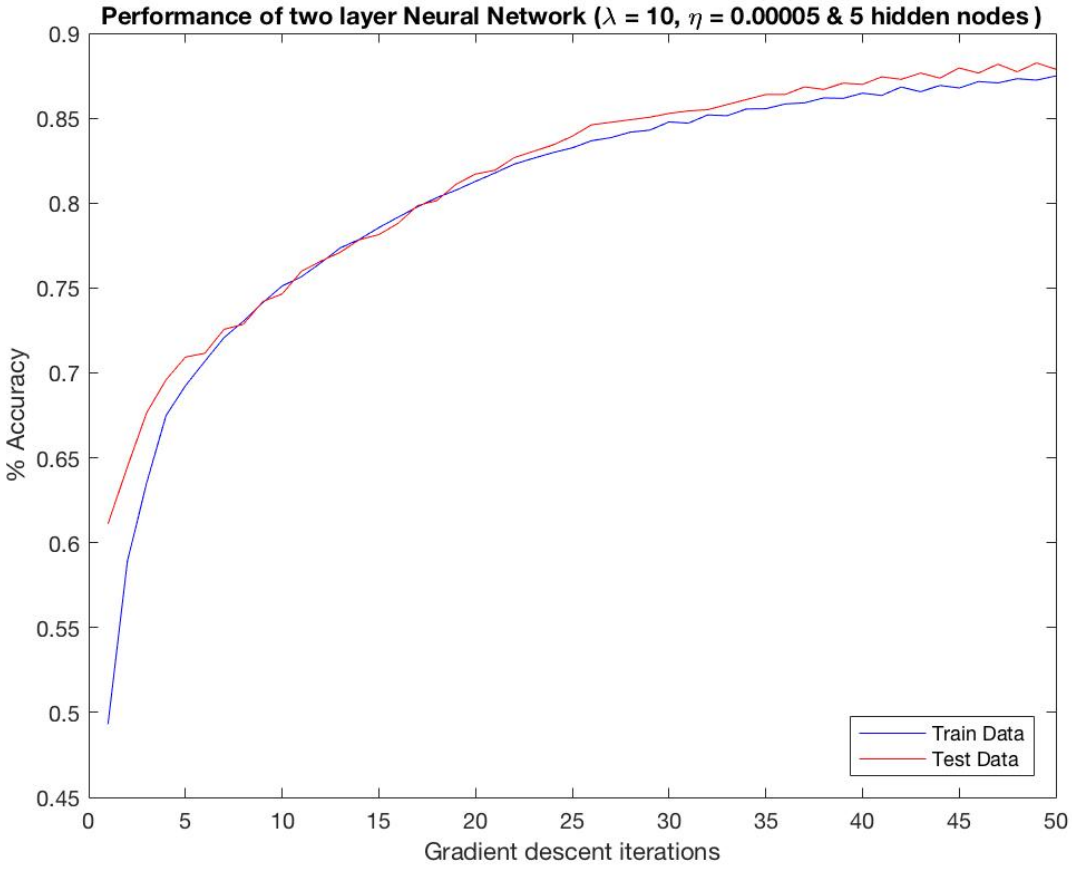

Figure 21: MNIST: Plot with the regularization parameter $\lambda = 10$ with the number of iterations fixed. Final accuracy in test and train: 87.9% and 87.5% respectively.