**Back Propogation lab**
**Jacob Pettingill**
**CS 478**
**Mike Brodie**

## Part 1

*See Code submitted*

## Part 2

I was able to successfully implement the backpropagation learner so that it could handle the iris dataset. When testing on the iris dataset I used a random 75/25 split of the data for the training/test set and also a learning rate of .1. I also maintained a momentum term of .5, which seemed to be a good middle point for these tests. I used the validation set (VS) for the stopping criteria. Using the validation set for my stopping criteria was helped to establish when my training sets had converged upon some value.

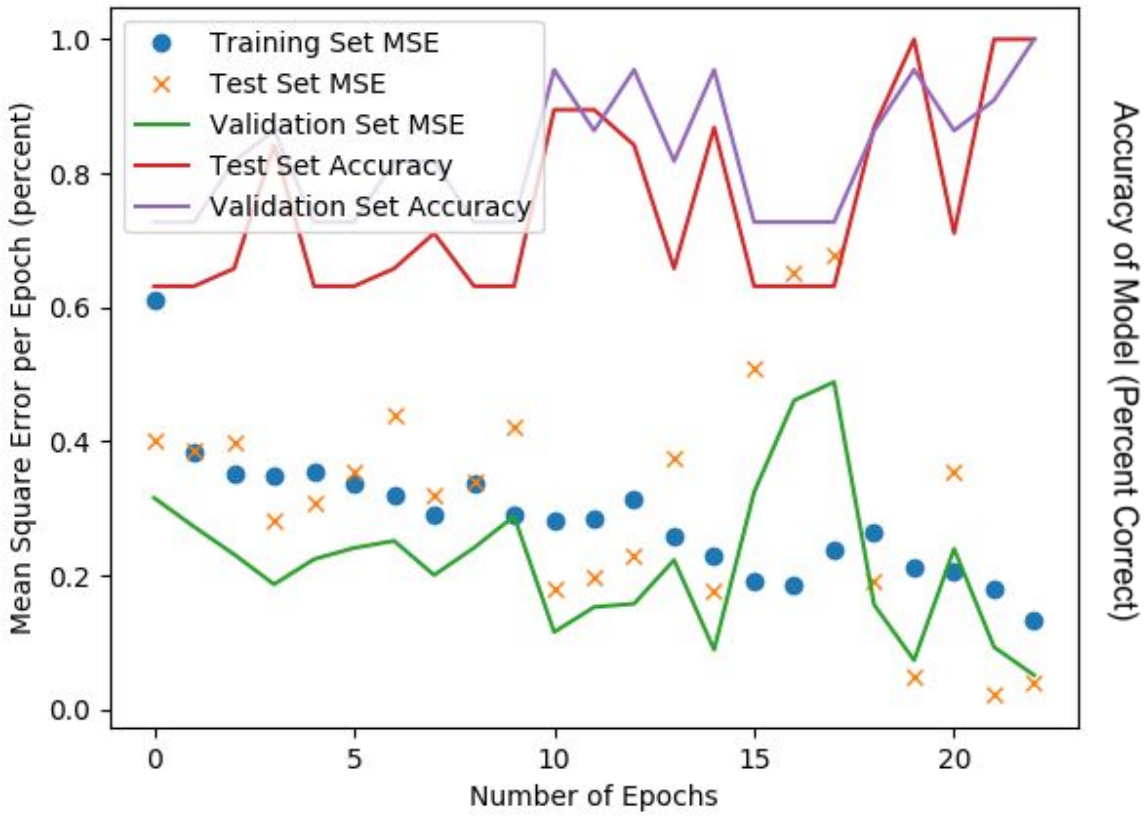



*Fig 1: Mean Square Error and Accuracy on Iris Data set (Run 1)*

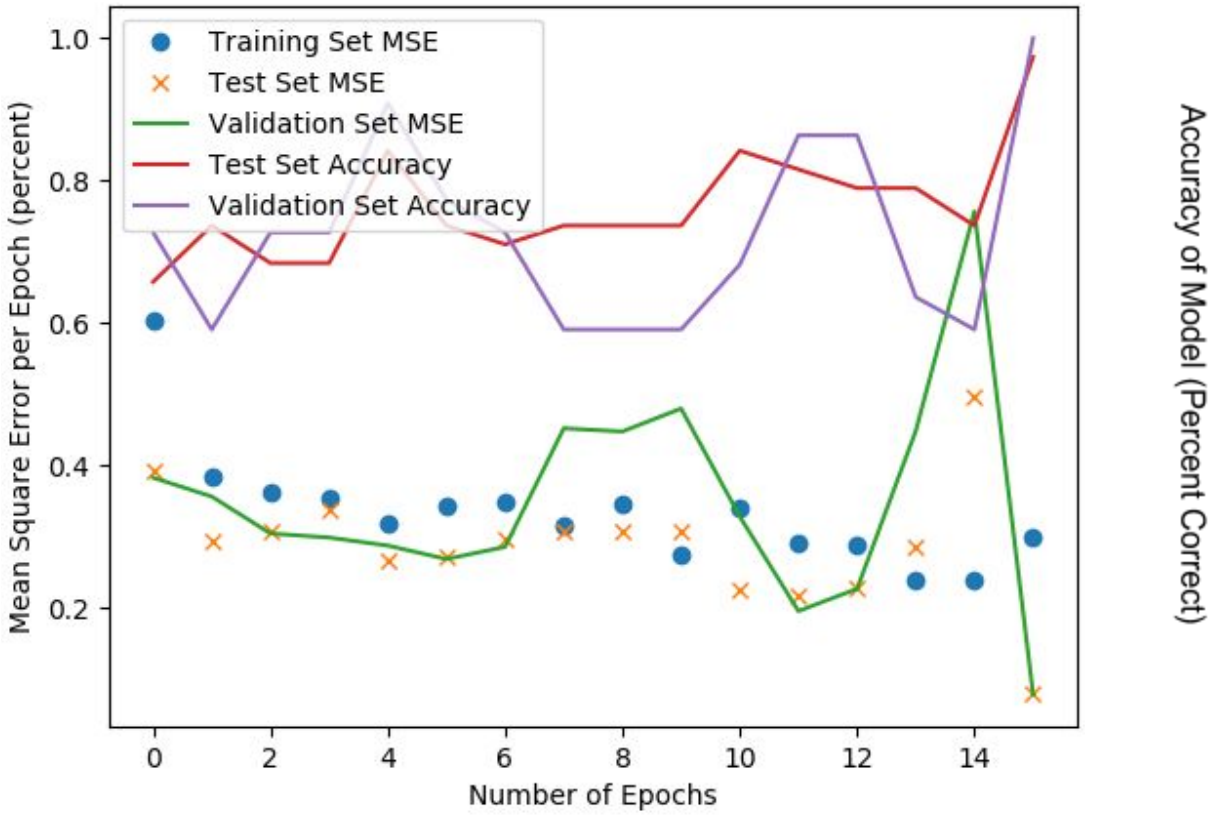



*Fig 2: Same as Fig 1 (Run 2)*

**Part 3**

In this part of the lab, I took the vowel set and did some feature selection on the dataset to help minimize the number of dimensions that were present in the dataset and allowing the model to be more accurate. I used the minimized version of the dataset for the last parts of the lab. The baseline accuracy for this lab with with all 14 features and 28 hidden nodes, was about 65 percent accuracy. This is shown in Figure 3. The model converges nicely after about 100 epochs but never is able to achieve accuracy on any of the data sets of more than about 65 percent, which happened to be on the Test Set.

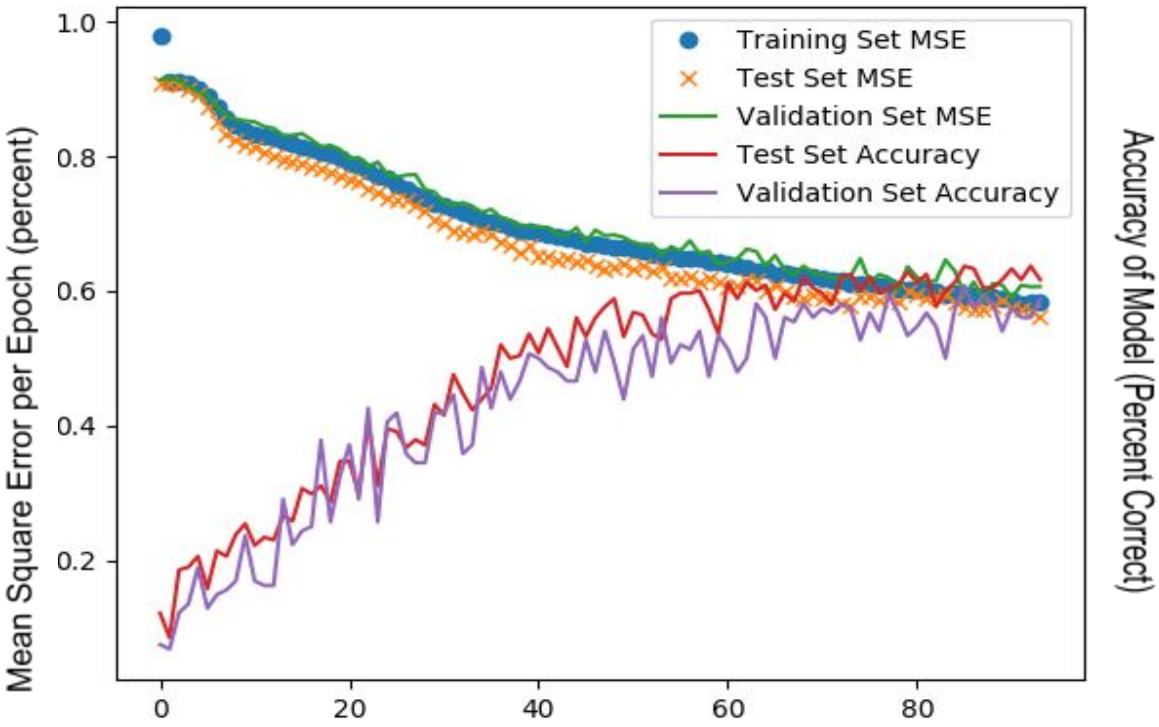



*Fig 3: Vowel Data set Accuracy*

I considered the dataset and was able to minimize it down to several different datasets. I first tried getting rid of all the extra vague features that made up a large portion of the data sets variety. I thought that this might help because I did not really know what feature 0-11 even represented and so I assumed that it might be irrelevant. This caused my accuracy to drop significantly and so I reevaluated my methods. In the end I went into WEKA and used the feature selection tools in there to help me pick off the most important features. I used the CFsubset evaluator and the bestFirst methods to determine the highest correlated data. That was surprising because it only kept features 4,5,7,8,10,11, and 13 leaving 7 features in total. I was surprised by this because gender and speaker number seemed to be not as important. I think that this is because with in the gender, speaker number, and train/test columns there is not much change in values, making it easy for the model to saturate its nodes quickly. The variety

in the other columns, as well as the spread in values made them more important in classifying.

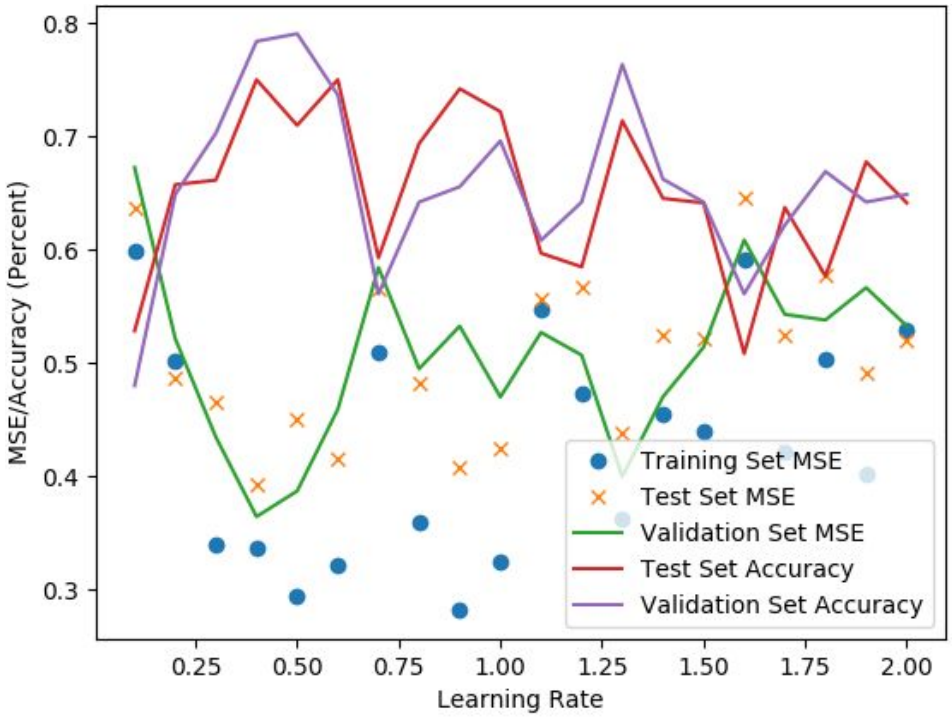
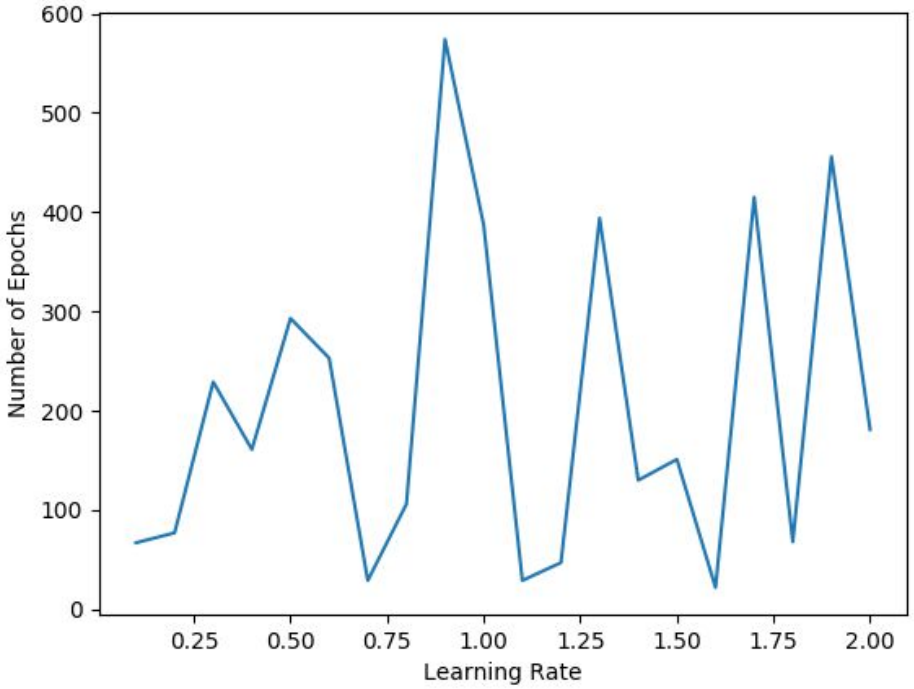



*Fig 4: MSE /Accuracy on the Vowel Data Set with varying learning rates*



Number of Epochs required to Converge

## Part 4

Once I had trimmed my dataset to the most important features, I began to play with the learning rate to determine what values would help my model converge on an MSE that was better than before. Figure 4 shows the varying Accuracies and MSE values for the vowel dataset. From this I saw that a learning rate that was between .3 and .4 was able to produce the best results. The initial conditions included a momentum value of .1 and 14 hidden nodes (which is double the number input nodes after the dataset was trimmed). I next played with the momentum values and hidden node values to see what values would help produce the best accuracy and lowest MSE values. Figures 5 and 6 show the results of varying those values.

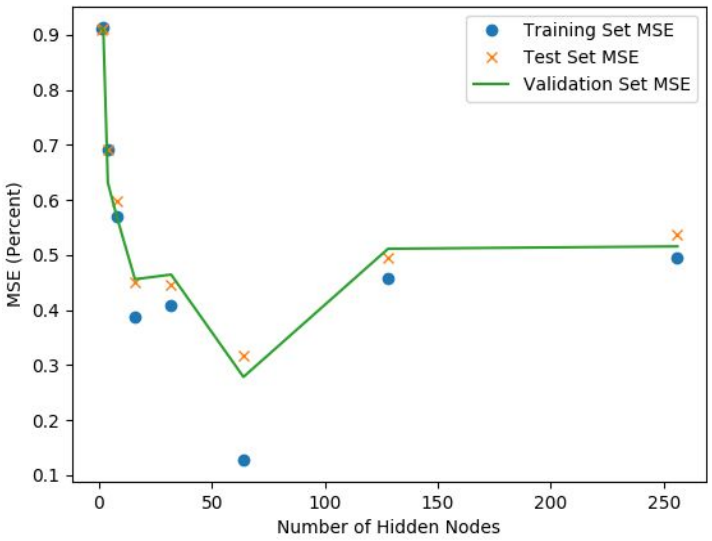



*Fig 5: MSE Value, Varying hidden nodes*

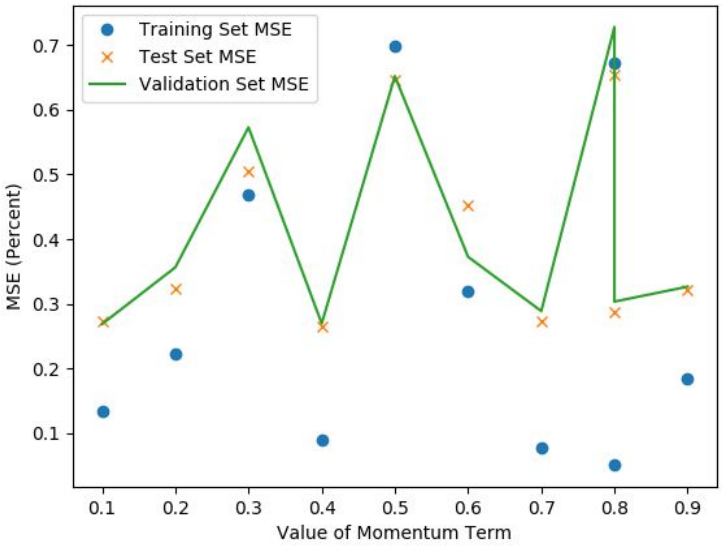



*Fig 6: MSE Vals, Varying Momentum*

The Results of varying the hidden nodes and momentum showed the best values from hidden nodes is about 64 (for a 7 input dataset) and about .4 percent to be included for momentum (again for the 7 input dataset).

## Part 5

For my creativity project I decided to test number of layers. I made my neural network totally dynamic and made the number of layer adjustable as well as the number of nodes of each layer adjustable. I ran the vowel set, adjusting layers (14 nodes per layer) and checking MSE. I don't think that my layers are fully and correctly implemented. The more layers that I added into the neural network made the MSE go up. The best accuracy that I got was with a single layer. This might have been caused by the number features that were included and maybe I needed to include more features. However I tried this and it didn't change that my accuracy decreased significantly with as I

increased the layers.