**Project 2 Writeup**

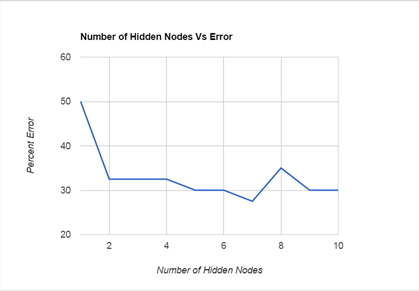
**Odell Dotson and Ethan Prihar**

1.

|  |  |  |
| --- | --- | --- |
| **Holdout Percentage** | **First or Last Data Points** | **Percent Error** |
| 20 | First | 30 |
| 40 | First | 26.25 |
| 60 | First | 27.5 |
| 20 | Last | 30 |
| 40 | Last | 21.25 |
| 60 | Last | 21.67 |

Based on the data, when different data is used to train the neural network the range of error can change slightly. When using the first 80% or last 80% of data there was enough overlap so that when testing on the remaining data the resulting error was the same. When testing on the first 60% or last 60% of the data there was still some overlap but enough unique data points to show a 5% difference in error, which is significant. This difference in error is maintained when testing on 40% of the data. Because the neural net is trained using different data points it will have slightly different weights. The significant change in error leads to the conclusion that we are over constraining the neural network.

2.



Based on the data, it appears that there is an ideal number of hidden nodes to use for the amount of inputs and outputs. From the graph you can see that the error decreases slowly from 1 to 7 nodes that then increases from 7 to 10 nodes. This shows that using 7 hidden nodes is the ideal number to reduce the error of the neural network when predicting values. It makes sense for there to be an ideal number of nodes because with too little nodes the neural net wouldn’t be able to properly associate different input with the output they create and with too many nodes the neural network would become over constrained and it would not predict the output of new data accurately.

3.

When implementing this neural network, we did make the assumption that we would only ever have to read in (x,y) touples corresponding to a single output. We also assumed that the number of hidden layers would always be just one. Changing the number of input nodes would require only a few small changes, but making the number of input nodes to be fully general would probably take a little longer. Changing the number of hidden nodes would take a decent amount of time, but wouldn't be really hard because all the code is already there for creating the first layer. Again, making it general so the user selects the number of hidden layers would take a decent amount of time to implement.

Having it map to multiple output nodes would probably not be too tricky of a thing to add in, but making it general for any number of output nodes could get a little tricky. Any major changes to the overall network topology of the neural network would likely require a lot of work to implement, such as multiple hidden layers that all spoke to each other or something like that.