Armand Nasseri
912679383
ECS 171

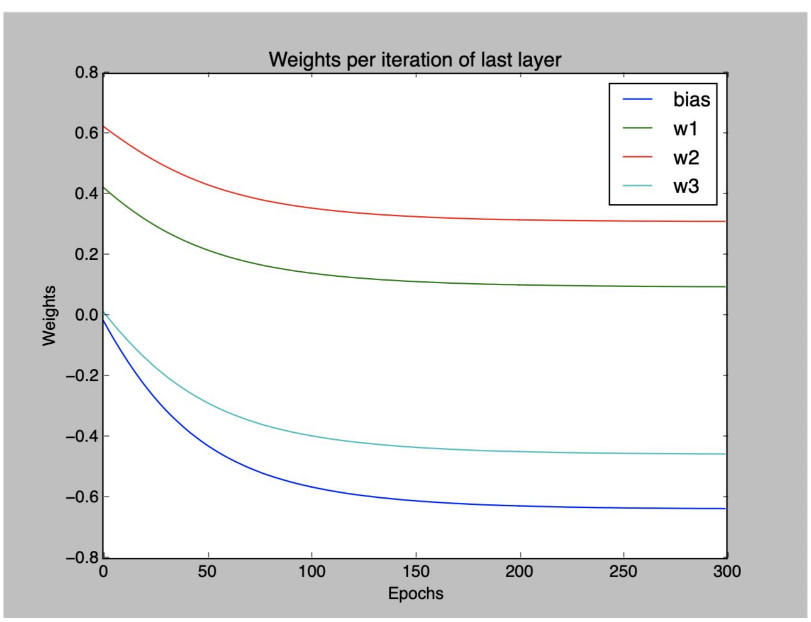
Homework 2

For this homework assignment, I have created 7 different files that correspond to each of the problems in the assignment description. Each file will have to be run separately to view the output.

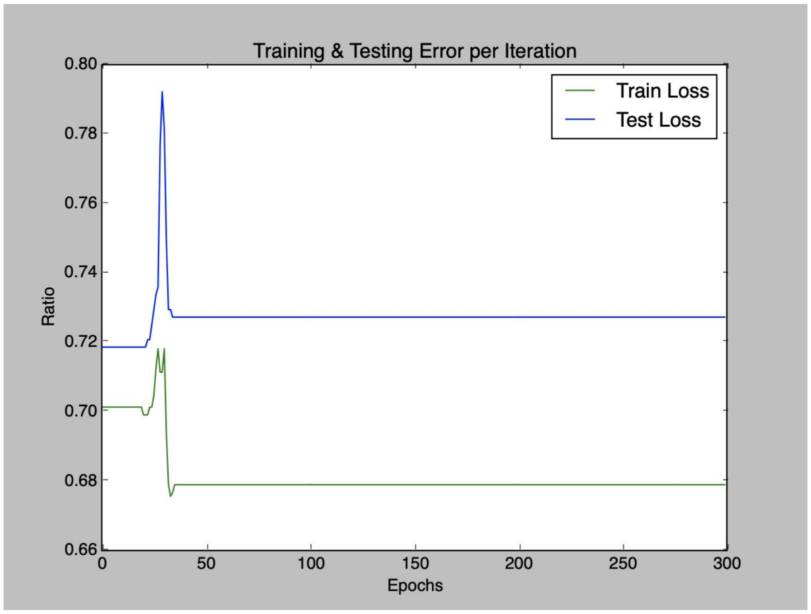
1)

a. The two outlier detection algorithms that I applied to my dataset were **one-class SVM** and **Isolation Forest**. Both algorithms used in my dataset detected outliers.
b. One-class SVM showed that **33.08 %** of my dataset contained outliers while Isolation Forest showed that only **8.56 %** of my dataset contained outliers. This difference between the algorithms clearly shows that the methods do not agree.
c. One-class SVM explains how it is very sensitive to outlier detection. The algorithm appears to work best for "novelty detection", which assumes that the training set does not include any outliers. Therefore, outlier detection without any assumptions on the distribution of data could yield not very accurate results. Isolation forest works better for the dataset because it only compares the score of anomalies of one sample with the scores of its other neighbors. The biggest advantage with Isolation forest is that it works very well with large datasets.

2) I. **Plot for weight values per iteration for the last layer (3 weights and bias):**



II. **Plot for training and test error per iteration:**



3)

Final Hidden Layer Weights/Training Error

Bias	Weight 1	Weight 2	Weight 3	Training Error
-0.6335665	**0.100725956**	**0.31474856**	**-0.45109203**	**0.68800538**

Final activation function formula for class "CYT"

Activation Function | Problem 3

Sigmoid $= \dfrac{1}{1+e^{-z}}$

$$z_1^{(4)} = \sum_{i=0}^{3} w_{1i}^{(3)} a_i^{(3)}$$

$$= W_{10}^{(3)} + W_{11}^{(3)} a_1^{(3)} + W_{12}^{(3)} a_2^{(3)} + W_{13}^{(3)} a_3^{(3)}$$

$$= -0.6335 + 0.1007\, a_1^{(3)} + 0.3147\, a_2^{(3)} - 0.4511\, a_3^{(3)}$$

$$a_1^{(4)} = g(z_1^{(4)})$$

where $g(z) = \dfrac{1}{1+e^{-z}}$

4)

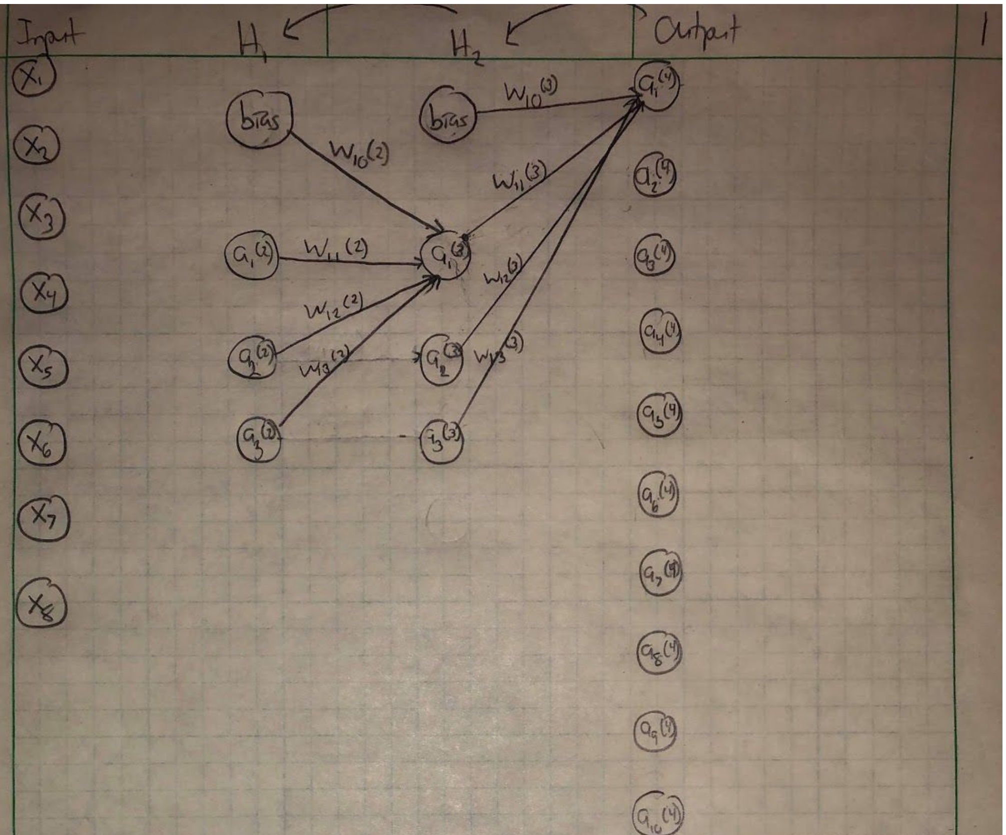
Calculations done by code

Bias Hidden Layer 1: 0.9999982
Bias Output Layer: 0.99991614

Weight 1 Hidden Layer 2: 0.9999982
Weight 2 Hidden Layer 2: 0.9999987
Weight 3 Hidden Layer 2: 0.9999991

Weight 1 Output Layer: 0.99991935
Weight 2 Output Layer: 0.99997395
Weight 3 Output Layer: 0.9999364

Calculations done by hand



Output Layer, [redacted]

- $\alpha = 0.01$
- Initializing all weights to 1.

① $$\frac{\partial RSS}{\partial w_{10}^{(3)}} = \frac{\partial RSS}{\partial a_1^{(4)}} \cdot \frac{\partial a_1^{(4)}}{\partial z^{(4)}} \cdot \frac{\partial z^{(4)}}{w_{10}^{(2)}}$$

$$\frac{\partial RSS}{\partial w_{10}^{(3)}} = -(y_k - a_1^{(4)})(1 - a_1^{(4)}) \cdot a_1^{(4)} \cdot a_1^{(4)}$$

Forward Pass

- All weights initialized to zero, except calculated weights which will be initialized to one.

We have:

$W_{10}^{(3)} = 1$ $\quad$ $W_{10}^{(2)} = 1$

$W_{11}^{(3)} = 1$ $\quad$ $W_{11}^{(2)} = 1$

$W_{12}^{(3)} = 1$ $\quad$ $W_{12}^{(2)} = 1$

$W_{13}^{(3)} = 1$ $\quad$ $W_{13}^{(2)} = 1$

First Sample Input

$X = [0.58, 0.61, 0.47, 0.13, 0.5, 0, 0.48, 0.22]$

First Sample Output (MIT)

$Y = [0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0]$

Feed Forward

$a_1^{(2)} = \text{Sigmoid}\left(\sum_{i=1}^{8} W_i^{(1)} x_j^{(1)} + W_{10}^{(1)}\right)$

$\text{Sigmoid}(0^T X + W_{10}^{(1)}) = \text{Sigmoid}(0+0) = \frac{1}{1+e^{-0}} = \frac{1}{2}$

$a_2^{(2)} = \text{Sigmoid}\left(\sum_{i=1}^{8} W_i x_j + W_{20}^{(1)}\right)$

$\text{Sigmoid}(0^T X + W_{20}^{(1)}) = \text{Sigmoid}(0+0) = \frac{1}{1+e^{-0}} = \frac{1}{2}$

We will have similar values for the rest of the activation functions.

$$\boxed{a_1^{(2)} = \frac{1}{2},\ a_2^{(2)} = \frac{1}{2},\ a_3^{(2)} = \frac{1}{2}}$$

$a_1^{(3)} = \text{Sigmoid}\left(\sum_{i=1}^{3} a_i^{(2)} \cdot W_i^{(2)}\right) + W_{10}^{(2)}$

$= \text{Sigmoid}\left((1 \cdot \tfrac{1}{2})(1 \cdot \tfrac{1}{2})(1 \cdot \tfrac{1}{2})\right) + 1 = \frac{1}{1+e^{-2.5}} = 0.924$

$a_2^{(3)} = \text{Sigmoid}\left(\sum_{i=1}^{3} a_i^{(2)} \cdot W_{2i}^{(2)}\right) + W_{20}^{(2)}$

$= \text{Sigmoid}\left((\tfrac{1}{2} \cdot 0) \cdot (\tfrac{1}{2} \cdot 0) \cdot (\tfrac{1}{2} \cdot 0) + 0\right) = \frac{1}{1+e^{-0}} = \frac{1}{2}$

$a_3^{(3)} = \text{Sigmoid}\left((\tfrac{1}{2} \cdot 0)(\tfrac{1}{2} \cdot 0)(\tfrac{1}{2} \cdot 0)) + 0\right) = \frac{1}{1+e^{-0}} = \frac{1}{2}$

$$\boxed{a_1^{(3)} = 0.924,\ a_2^{(3)} = \frac{1}{2},\ a_3^{(3)} = \frac{1}{2}}$$

Final Feed Forward

$$a_1^{(4)} = \text{Sigmoid}\left(\sum_{i=1}^{3} w_{1i}^{(3)} a_i^{(3)}\right) + w_{10}^{(3)}$$

$$= \frac{1}{1 + e^{-w_{10}^{(3)} + w_{11}^{(3)} \cdot a_1^{(3)} + w_{12}^{(3)} \cdot a_2^{(3)} + w_{13}^{(3)} \cdot a_3^{(3)}}}$$

$$= \frac{1}{1 + e^{-(1 + 0.924 + 1/2 + 1/2)}} = 0.95$$

Backpropagation (Output Layer)

Output Layer to Hidden Layer 2

$$\frac{\partial RSS}{\partial w_{10}^{(3)}} = \frac{\partial RSS}{\partial a_1^{(4)}} \cdot \frac{\partial a_1^{(4)}}{\partial z^{(4)}} \cdot \frac{\partial z^{(4)}}{\partial_{10}^{(3)}}$$

$$w_{10}^{(3)} := w_{10}^{(3)} + \alpha(y_k - a_1^{(4)})(1 - a_1^{(4)}) \cdot a_0^{(3)}$$
$$= 1 + (0.01)(0 - 0.949)(1 - 0.949) \cdot 1 \ (0.949)$$
$$= \boxed{0.9995}$$

$$w_{11}^{(3)} := w_{11}^{(3)} + \alpha(y_k - a_1^{(4)})(1 - a_1^{(4)}) \cdot a_1^{(3)}$$
$$= 1 + (0.01)(0 - 0.949)(1 - 0.949) \cdot 0.949$$
$$= \boxed{0.9996}$$

$$w_{12}^{(3)} := w_{12}^{(3)} + \alpha(y_k - a_1^{(4)})(1 - a_1^{(4)}) \cdot a_2^{(3)}$$
$$= 1 + (0.01)(0 - 0.949)(1 - 0.949) \cdot \tfrac{1}{2} \ (0.949)$$
$$= \boxed{0.9998}$$

$$w_{13}^{(3)} : w_{13}^{(3)} + \alpha(y_k - a_1^{(4)})(1 - a_1^{(4)}) \cdot a_3^{(3)}$$
$$= 1 + (0.01)(0 - 0.949)(1 - 0.949) \cdot \tfrac{1}{2} \ (0.949)$$
$$= \boxed{0.9998}$$

Backpropagation (Hidden Layer 2)

$$\frac{\partial RSS}{\partial w_{ji}^{(l-1)}} = -\delta_j^{(l-1)} \cdot a_i^{(l-2)}$$

$$= \left(\sum_{i=1}^{14} \delta_k^{(4)} \cdot w_{k1}^{(3)} \cdot (1-a_1^{(3)}) \cdot a_1^{(3)}\right)$$

Since all outputs are zero

$$= \delta_1^{(4)} \cdot w_{11}^{(3)} \cdot (1-a_1^{(3)}) \cdot a_1^{(3)}$$

$$= \delta_1^{(4)} \cdot 1 \cdot (1-0.924) \cdot (0.949)$$

$$= (0-0.949)(1-0.949) \cdot (0.949) \cdot 1$$

$$\boxed{\delta_1^{(4)} = -0.045}$$

$$\delta_1^{(3)} = -0.045 \cdot 1\,(1-0.924)(0.929)$$

$$\boxed{\delta_1^{(3)} = -0.0032}$$

$$W_{10}^{(2)} := w_{10}^{(2)} + (0.01)(-0.032) \cdot 1$$

$$= 1 + (0.01)(-0.032) \cdot 1$$

$$\boxed{W_{10}^{(2)} = 0.999968}$$

$$W_{11}^{(2)} := w_{11}^{(2)} + (0.01)(-0.0032)\,\tfrac{1}{2}$$

$$\boxed{W_{11}^{(2)} = 0.999984}$$

$$W_{12}^{(2)} := W_{12}^{(2)} + (0.01)(-0.0032)\,\tfrac{1}{2}$$

$$\boxed{W_{12}^{(2)} = 0.999984}$$

My hand calculations were relatively close to the calculations provided by my code output. One possible reason for slight differences between the calculations was most likely due to the way that the keras optimizer handles the weights.

5)

Grid Search Matrix with Testing Error per cell

	3 Nodes	6 Nodes	9 Nodes	12 Nodes
Layer 1	**0.700223713546**	**0.661073825470**	**0.680089485058**	**0.681208053291**
Layer 2	**0.701342281779**	**0.678970916825**	**0.678970916825**	**0.678970916825**
Layer 3	**0.678970916825**	**0.678970916825**	**0.678970916825**	**0.678970916825**

The optimal configuration is 1 Layer and 6 Nodes with a testing error of 0.6610738254700197. It appears that the relationship of having multiple nodes allows for a more accurate calculation and a smaller testing error. In this case for this model, having more nodes compared to more layers was the most optimal configuration.

6) The predicted class that the sample belongs to is: **CYT.**

7)

Grid Search Matrix with Testing Error per cell

	3 Nodes	6 Nodes	9 Nodes	12 Nodes
Layer 1	**0.096516874800**	**0.095393285724**	**0.091573057147**	**0.091460684712**
Layer 2	**0.100000025583**	**0.094157336669**	**0.090898891245**	**0.086629224493**
Layer 3	**0.096179799149**	**0.092022482464**	**0.090000011813**	**0.080674164884**

Changing the activation functions to relu and softmax is clearly a better choice as it improved my overall performance from a 32% accuracy to a ~90% accuracy. The optimal configuration is 3 Layers and 12 Nodes with a testing error of 0.080674164884. It appears that the relationship of having multiple layers as well as having many nodes allows for a more accurate calculation and a smaller testing error. In this case for this model, having both more layers and more nodes was the most optimal configuration.

Plot for training and test error per iteration: