- Please submit a link to your GitHub repository for your class.

  Class Repo: https://github.com/jkalldre/cs450

  Prove08 Repo: https://github.com/jkalldre/cs450/tree/master/python/08Prove

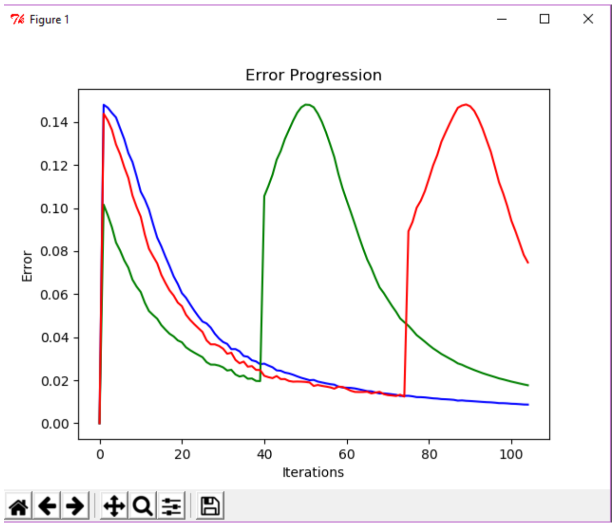
- Describe your overall approach to implementing the algorithm in code. How are your classes/data structures organized? How do you keep track of the necessary pieces for back-propagation.

I implemented a Node, Layer, and NNet classifier class to help me keep track of member variables. Node has two arrays (inputs, weights) to keep track of corresponding inputs and weights. These could be the initial inputs, outputs from the prior layer or the bias nodes they all can be treated the same. I implemented the Sigmoid function here to calculate a and output function that calculates h. I also have two member variables d and t that hold error and target respectively. Update sets up new inputs using the outputs of the previous nodes. Layer applies update to all of the nodes contained within it and keeps track of the bias/inputs. NNet holds the dataset, targets, layers, learning variable n, and the errors up to this point. It feeds in datapoints forward, back propagates and then manipulates the weights between nodes. Main then feeds these error points into pyplot to show the progression of the algorithm over each iteration.

- Describe the part of the assignment that gave you the most trouble, and how you overcame it.

The hardest part was keeping track of where I was at any given time within a nested loop. I tried to use clear variable names and local variables where ever possible to minimize the chance of user spawned error.

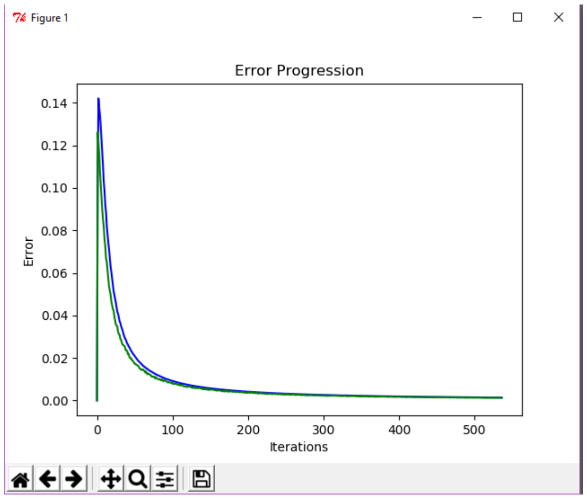
- Produce at least one graph to show the training progress for the Iris dataset.



- Compare your results on the Iris dataset to those of an existing implementation.

I am not sure why exactly I get these odd returns in error on the iris data set but I believe that it is because the training set is so small. Looking at the results I got from existing implementations they were able to limit the rise in error.

- Produce at least one graph to show the training progress for the Diabetes dataset.



- Compare your results on the Diabetes dataset to those of an existing implementation.

I was extremely happy with the results of my Pima Indian dataset. I got a steep fall off in error around 100 iterations so I would tell my algorithm to stop around 200 so that I would not waste data points with training.

- Describe any efforts you made to go above and beyond.

I enabled user input to prematurely end the training process if they found they liked the results at a certain iteration.

- Please state which category you feel best describes your assignment and give a 1-2 sentence justification for your choice:

A) Some attempt was made

B) Developing, but significantly deficient

C) Slightly deficient, but still mostly adequate

D) Meets requirements

--[E) Shows creativity and excels above and beyond requirements.]—

I met all of the requirements of the assignment. I allowed the premature end to the training process to save valuable data points to be used in testing, I kept my code readable, modular, and encapsulated with use of classes and methods.