Brandon TerLouw & Warren Cho

Milestone B Progress Report

1. A revolutionary classifier
2. Brandon TerLouw – Began development of neural network

Warren Cho – Began development of decision tree

1. The classifier is supposed to use data to predict the class of a specific instance
2. Neural Network – A neural network is one way to implement a classifier. It can be made up of multiple layers of perceptrons. A perceptron is a device that takes an array of weights, and an array of input values, multiplies them and sums all the values. It then either outputs a 1 if the summation is above a threshold, and 0 if it is below. Another version of a perceptron uses a sigmoid function and returns a value between 0 and 1. A neural network can be trained by changing the weights until the error is close to a minimum. Once a neural network has been trained, ideally it will be able to take data and assign it to a class.

Decision tree –

1. Neural Network:

After training a single perceptron to work with 2 classes, we were able to get the neural network to identify if a plant is a Setosa or a Versicolor. The data was from the Iris Data Set, with the third plant omitted. Below is a log, our perceptron worked 100% of the time due to the data being linearly separable.

Testing training data:

We compute 0, actually: 0

We compute 0, actually: 0

We compute 1, actually: 1

We compute 0, actually: 0

We compute 0, actually: 0

We compute 0, actually: 0

We compute 1, actually: 1

We compute 0, actually: 0

We compute 1, actually: 1

We compute 1, actually: 1

We compute 0, actually: 0

We compute 0, actually: 0

We compute 1, actually: 1

We compute 0, actually: 0

We compute 0, actually: 0

We compute 1, actually: 1

We compute 1, actually: 1

We compute 0, actually: 0

We compute 1, actually: 1

We compute 0, actually: 0

We compute 1, actually: 1

We compute 0, actually: 0

We compute 1, actually: 1

We compute 0, actually: 0

We compute 0, actually: 0

We compute 0, actually: 0

We compute 1, actually: 1

We compute 1, actually: 1

We compute 1, actually: 1

We compute 0, actually: 0

We compute 1, actually: 1

We compute 1, actually: 1

We compute 1, actually: 1

We compute 1, actually: 1

We compute 0, actually: 0

We compute 1, actually: 1

We compute 0, actually: 0

We compute 0, actually: 0

We compute 1, actually: 1

We compute 1, actually: 1

We compute 0, actually: 0

We compute 1, actually: 1

We compute 1, actually: 1

We compute 0, actually: 0

We compute 1, actually: 1

We compute 0, actually: 0

We compute 1, actually: 1

We compute 0, actually: 0

We compute 1, actually: 1

We compute 0, actually: 0

We compute 0, actually: 0

We compute 1, actually: 1

We compute 1, actually: 1

We compute 0, actually: 0

We compute 0, actually: 0

We compute 0, actually: 0

We compute 1, actually: 1

We compute 0, actually: 0

We compute 0, actually: 0

We compute 0, actually: 0

We compute 1, actually: 1

We compute 0, actually: 0

We compute 1, actually: 1

We compute 1, actually: 1

We compute 1, actually: 1

We compute 1, actually: 1

We compute 0, actually: 0

We compute 1, actually: 1

We compute 0, actually: 0

We compute 0, actually: 0

We compute 1, actually: 1

We compute 1, actually: 1

We compute 1, actually: 1

We compute 0, actually: 0

We compute 0, actually: 0

We compute 0, actually: 0

We compute 1, actually: 1

We compute 1, actually: 1

We compute 1, actually: 1

We compute 1, actually: 1

We compute 0, actually: 0

We compute 1, actually: 1

We compute 0, actually: 0

We compute 1, actually: 1

We compute 0, actually: 0

We compute 1, actually: 1

We compute 1, actually: 1

We compute 0, actually: 0

We compute 1, actually: 1

We compute 1, actually: 1

We compute 1, actually: 1

We compute 0, actually: 0

We compute 0, actually: 0

We compute 1, actually: 1

We compute 0, actually: 0

We compute 0, actually: 0

We compute 0, actually: 0

We compute 0, actually: 0

We compute 1, actually: 1

Testing never seen data:

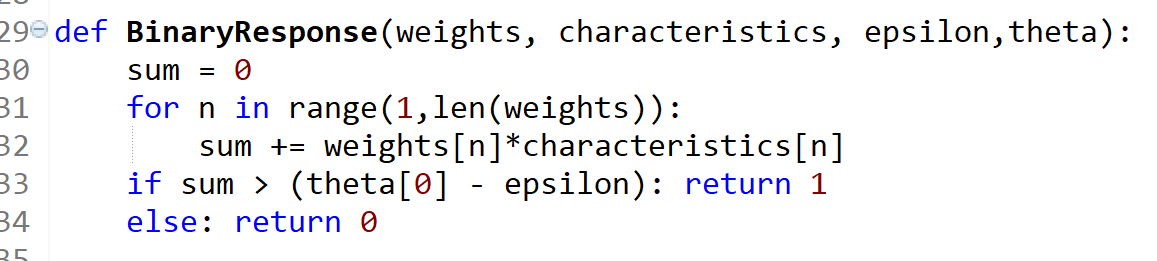
We compute 1, actually: 1

Decision Tree:

1. Neural Network – To run demo you must run ‘NeuralNetCall.py”. It must be in the same folder as NeuralNet.py and irisData.csv

Decision Tree –

1. Neural Network – Below is a screen shot of the code used for a perceptron. It takes weights, a threshold, epsilon and data. It then returns either a 1 or 0.



Decision Tree –

1. Brandon TerLouw – So far I have learned much about how a neural network operates. I learned what it takes to program and train a simple perceptron in order for it to classify linear separable data. I have read much on backpropagation and believe I am able to combine it with my simple perceptron to create a two layer neural network. I also learned how to import data from a csv file.

Warren Cho -

1. Neural Network – Next we need to turn our simple perceptron into a multilayer neural network so it can classify data into more than 2 categories.

Decision Tree -

1. Neural Network – For the most part I used information on slide to program and train perceptron.

This is the article I read to better understand math behind back propogation:

<https://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example/>

Decision Tree -