**Neural Networks**

**Project 3 Report**

***Braeden Brettin and Pierre Balinda***

**3/13/2017**

**Table of Contents**

Table of Figures

Introduction to Neural Networks

Brainstorming

Defining the Parameters

Creating the Network

Training the Network

The Final Product

References

**Table of Figures**

Figure 1. Neuron Anatomy

Figure 2. Neural Network Layers

Figure 3. Neural Network Classes

**Introduction to Neural Networks**

Neural networks, along with genetic algorithms and fuzzy logic, are a critical area of advanced computational techniques. Using knowledge gained from a study of neurology, programmers can mimic the way neurons communicate with one another to perform advanced computational tasks otherwise unachievable through normal object-oriented programming methods. In the past, neural networks have primarily been used in the field of character recognition, although advances in the field of software engineering and neurology have enabled programmers to attempt far more difficult tasks.

In our brains, neurons relate to thousands of other neurons, sending electrochemical signals to communicate with each other. Inputs are received in the neurons through synapses, and the neuron itself sums up the various inputs and fires if the output is greater than a threshold value. If the neuron fires, it will output a signal along an axon, which will then input to other neurons (*Neural Networks in Plain English*, p. 2). The outline of this neuron is as shown in Figure 1 (, below. Artificial neural networks model this behavior on a much smaller scale.

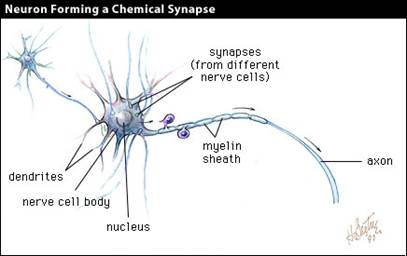


Figure 1. Neuron Anatomy

To model the hierarchy of a neural network, a layer of neurons is created which will take in the inputs to the program. These inputs are the various values of the defined parameters that will determine the final output result. The outputs of these neurons are then combined as inputs to another layer of neurons called the hidden layer. By defining weights for each of these inputs/outputs, an output can be derived for each of the neurons in the hidden layer using one of several types of functions, the most common of which being the sigmoid function, a more realistic deviation of a step function. The outputs from the hidden layer are then combined into the final neuron, which calculates the final output of the neural networks.

As it exists right now, the network is far from ideal. With weights having randomly been assigned to each input, the programmer has little to no idea of the importance of each parameter in regards to the final output. As such, the programmer must ‘train’ the network. This training occurs in the form of both unsupervised and supervised training. For the purposes of this project, the team will address only supervised training, as this process will be carried out to train the team’s network. This training is achieved by inputting values for an example for which the team knows the output. By finding the difference between the network’s calculated output and the expected output, the team can adjust the weights of the various inputs. The full calculations will be derived from Miller’s work on neural networks (p. 1). This process is repeated until the network gives reliable results.

With the theory behind neural networks discussed, the team can proceed into the creation of the actual network. The creation of an artificial neural network involves three major steps:

1. Define the parameters for the system. These parameters are a set of variables which will be used to calculate the final output.
2. Create the hierarchy of the network. This step involves creating classes for the neuron, layer, and network.
3. Train the network using unsupervised or supervised learning. The team’s project will use supervised learning, as it is far easier to explain and is more useful in displaying the modeling of an artificial network.

With these steps completed, the team can now input any set of values into the network and achieve a desirable, reliable, and realistic outcome.

**Brainstorming**

As previously mentioned, neural networks are most well-known for their use in character recognition. However, the team wanted to approach this technique from a different perspective. As such, the team desired to model a program with significant real-world applications. Having recently watched ‘The Big Short,’ a film dealing with the housing market collapse of 2008, the team had an interest in housing bonds. After researching these bonds further, the team found the topic to be an extremely pertinent one. No set formula exists to determine the credit grade of a bond; however, factors such as liquidity, credit score, and earnings play an important role in determining the grade of the bond (*Standard & Poor’s Definitions*, p. 1). The team ultimately decided to create a neural network that would specialize in determining the grade of a bond given several parameters.

To begin, the team created an input layer of neurons, each neuron taking in one input that represented the value of a parameter. Each input would then be passed through a positive or negative linear function that would return a value between 0 and 1. This value would then be passed to the hidden layer. The hidden layer consists of three neurons, each taking in every output from the input layer. However, those parameters which are more similar, such as earnings and credit score, would be given a greater weight at one of the hidden layer neurons. The sum of the inputs multiplied by the weights would then be passed through a sigmoid function to return a value between 0 and 1. The average of these outputs is the result of the output layer, with each score corresponding to a credit grade, as shown in Figure 2, below.

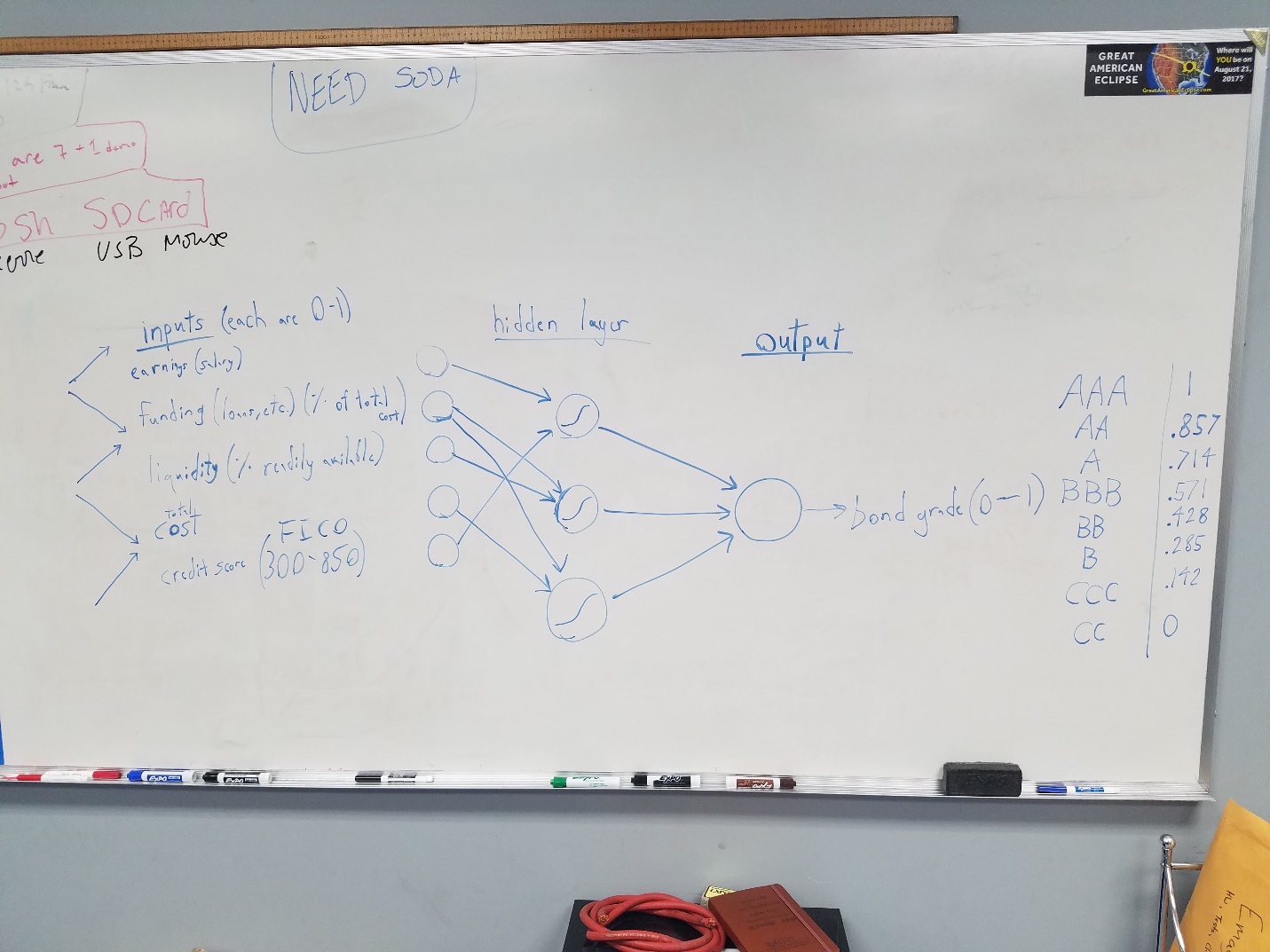


Figure 2. Neural Network Layers

After determining the hierarchy of the neural network, the team set about brainstorming the programming side of the network. To code this network, three classes must be created: the neuron class, the layer class, which holds a group of neurons, and the network class, which holds a group of layers. The variables and methods found in each class are shown in Figure 3, below.

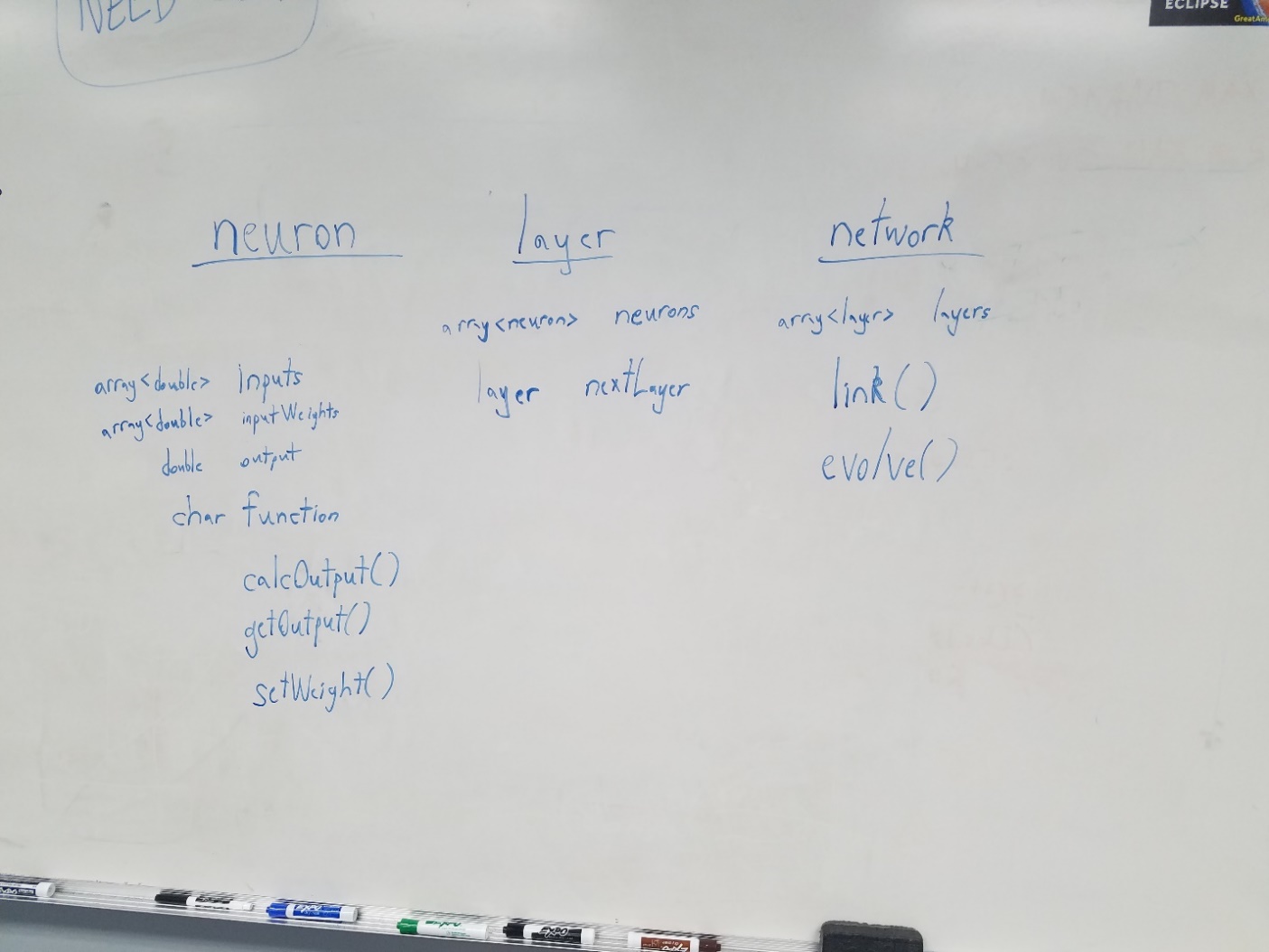


Figure 3. Neural Network Classes

With the hierarchy and programming structure of the neural network defined, the team can now proceed to creating the various classes needed for this network. The team will walk through each of the three major steps required to create a neural network in greater detail as this report progresses: defining the parameters, creating the network, and training the network. The team will begin by defining the parameters required for this network.

**Defining the Parameters**

**Creating the Network**

**Training the Network**

**The Final Product**

**References**

Detweiler, G. (Dec. 8, 2016). What is a Good Credit Score? Retrieved Mar. 1, 2017, from <https://www.credit.com/credit-scores/what-is-a-good-credit-score/>

# iamtrask. (July 12, 2015). A Neural Network in 11 Lines of Python (Part 1). Retrieved Feb. 28, 2017, from http://iamtrask.github.io/2015/07/12/basic-python-network/

Miller, S. (Aug. 10, 2015). Mind: How to Build a Neural Network (Part One). Retrieved March 1, 2017, from https://stevenmiller888.github.io/mind-how-to-build-a-neural-network/

Nielsen, M. (Jan. 2017). Using Neural Nets to Recognize Handwritten Digits. Retrieved March 1, 2017, from http://neuralnetworksanddeeplearning.com/chap1.html

Stergiou, C. & Siganos, D. Neural Networks. Retrieved March 2, 2017, from <https://www.doc.ic.ac.uk/~nd/surprise_96/journal/vol4/cs11/report.html>

Example of a Neural Network. Retrieved Feb. 28, 2017, from http://www.jmp.com/support/help/Example\_of\_a\_Neural\_Network.shtml

Neural Networks in Plain English. Retrieved Feb. 26, 2017, from <http://www.ai-junkie.com/ann/evolved/nnt1.html>

(Sep. 20, 2010). Standard & Poor’s Definitions. Retrieved Mar. 3, 2017, from <http://www.bankersalmanac.com/addcon/infobank/credit_ratings/standardandpoors.aspx>