**Part X Example**

**Initial Remarks**

We will use the following notation for our example:

Recall the Back-Propagation Algorithm:

1. Forward Pass:
   1. Compute
2. Backward Pass:
   1. Compute
3. Update:

During class, we have learned that the sigmoid function used for our Neural network is

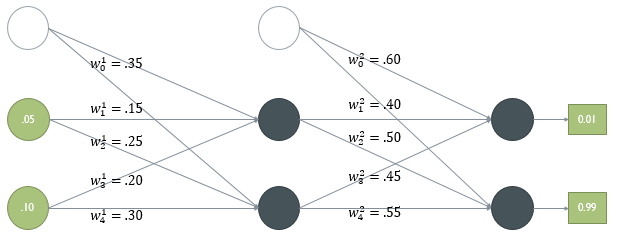
However, for our example, we will be using , the same sigmoid used for logistic regression. There are many texts on Neural Network, and some use this different function. It is good to be able to understand and spot the differences. This changes our back-propagation formulae slightly during the back-pass step. Particularly,

Is replaced by:

This is simply due to the slightly different way the sigmoid function is differentiated.

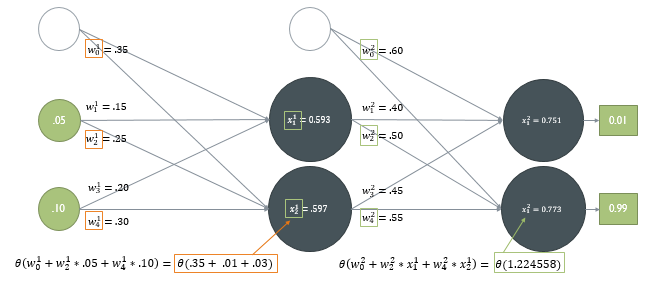
**Step 0: Initial Case**

We begin our example with the following weights, biases, and inputs, and expected outputs.



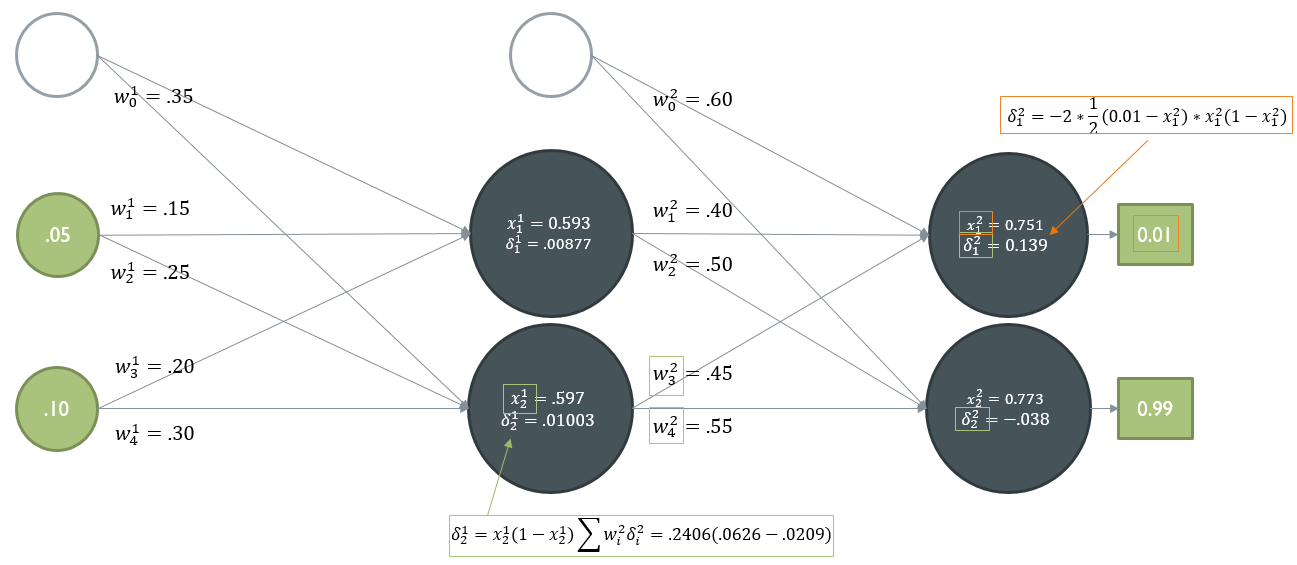
**Step 1: Forward Pass**

We first compute at every node. This is essentially the sum of the products of all weights with the node-values from the previous nodes.



and are calculated in orange and green. The components needed to calculate each value is also shown in orange and green boxes.

**Step 2: Backward Pass**

Next, we must calculate how much each node contributed to the total error of our output. We begin with base cases at the last layer of nodes (). Then we can use the calculated and traverse backwards in the graph.

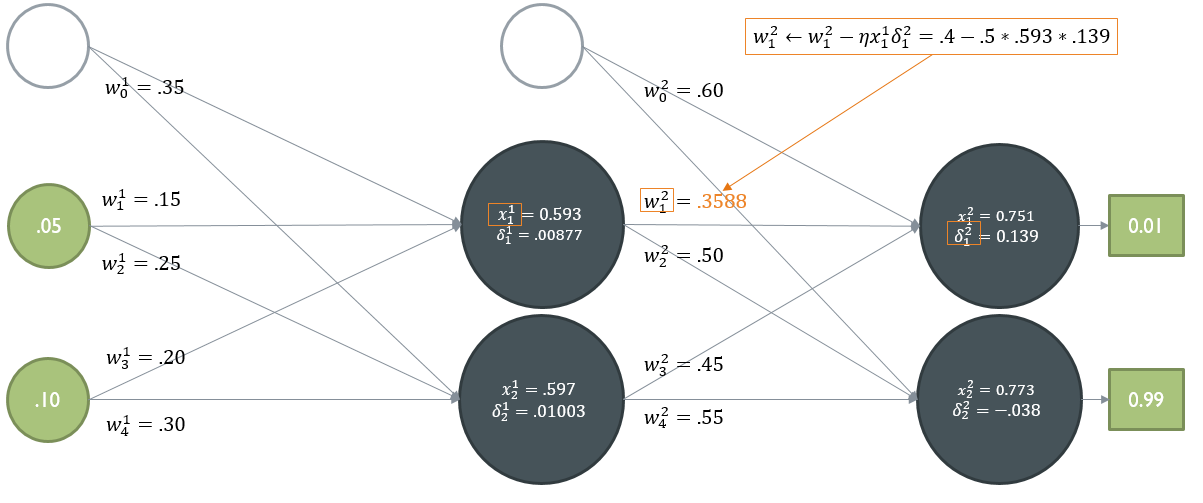
In this example, the base-case of (shown in orange) is calculated using the formula:

Again, we must note, that since we are using a different sigmoid function, is replaced with . We must also note that our errors are normalized. Thus, since there are 2 output nodes, the error is multiplied by .

is computed in a similar fashion, and is shown in green.

**Step 3: Update Weights**

Finally, using and , we can update .



The calculation of is shown in orange. We also note that in this example, we used .

**Part X: Applications of Neural Networks**

Neural Networks are a powerful tool to learn and predict from a vast amount of data. It can arrive at an ‘intuition’ of any given situation even if one doesn’t quite understand what is going on.

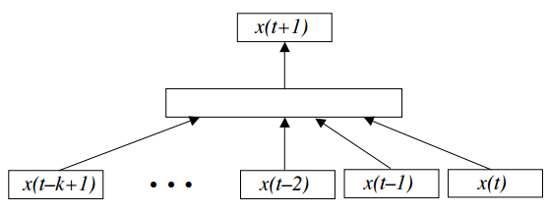
However, it falls short in a couple of regards. NN’s are not very good when exact values are required – as it simply guesses (very well) from previous experiences. Additionally, as mentioned, it is able to come up with good answers when a situation is not fully understood – however that advantage is its own shortcoming, as it will never be able to **explain** the reason behind the answers. Finally, NN’s require a large amount of data and epochs of iteration to learn properly. When there is not sufficient data, NN will perform poorly.

Here are some basic where Neural Networks are particularly good at and are used.

**Feed-Forward NN**

Time Series Prediction

* Neural networks are rather good at predicting what is going to happen at a given time, based on its prior knowledge and data.
* Examples of this situation include Stocks, Weather, etc.
* The basic organization of a Neural Network for this problem can look like:



Pattern Recognition

* Neural Networks can recognize patterns in images, sound, etc. and be used to identify features.
* Examples of this case include text-recognition, speech-recognition, etc.
* The basic organization of a NN that recognizes what digit is written on an image can look like:

