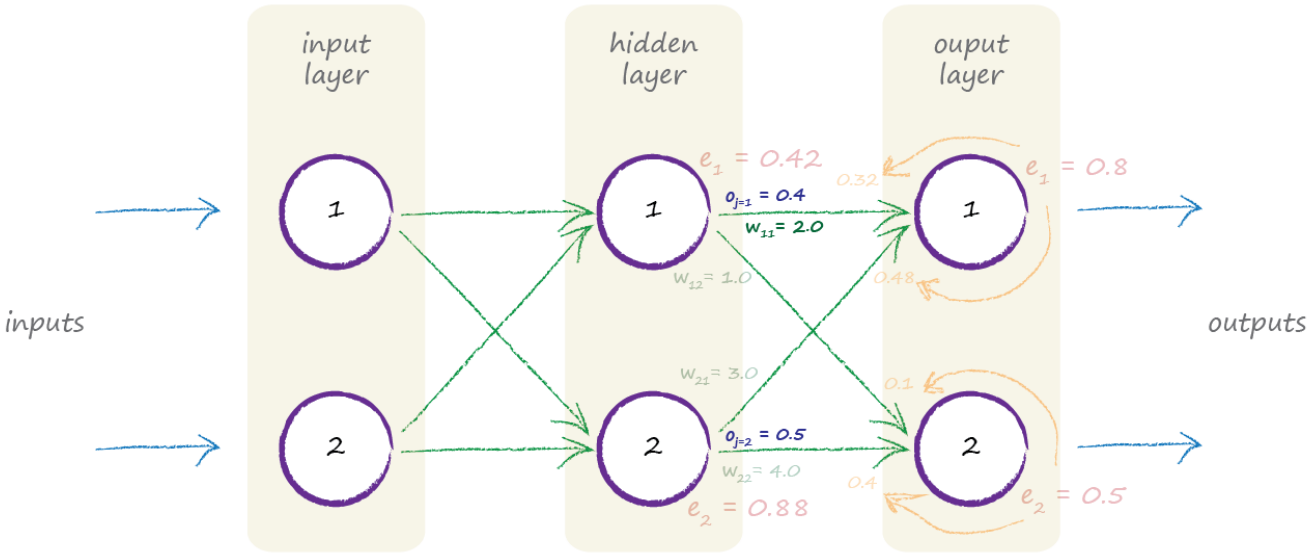
Weight Update Worked Example

Let's work through a couple of examples with numbers, just to see this weight update method working in Neural Network.

The following network is the one we worked with before, but this time we've added example output values from the first hidden node o_j=1 and the second hidden node o_j=2. These are just made up numbers to illustrate the method and aren't worked out properly by feeding forward signals from the input layer.



We want to update the weight w_{11} between the hidden and output layers, which currently has the value 2.0.

Let's write out the error slope again.

$$\frac{\partial E}{\partial w_{jk}} = -(t_k - o_k) \cdot sigmoid(\Sigma_j w_{jk} \cdot o_j)(1 - sigmoid(\Sigma_j w_{jk} \cdot o_j)) \cdot o_j$$

Let's do this bit by bit:

- The first bit (t_k - o_k) is the error e_1 = 0.8, just as we saw before.
- The sum inside the sigmoid functions $\Sigma_j\ w_{jk}\ o_j$ is $(2.0 * 0.4) + (3.0 * 0.5) = 2.3$.
- The sigmoid $1/(1 +$e^{-2.3}$)$ is then 0.909. That middle expression is then $0.909 * (1 - 0.909) = 0.083$.
- The last part is simply o_j which is o_j=1 because we are interested in the weight w_{11} where $j = 1$. Here it is simply 0.4.

Multiplying all these three bits together and not forgetting the minus sign at the start gives us -0.0265.

If we have a learning rate of 0.1 that give us a change of $-(0.1 * -0.04969) = +0.002650$. So the new w_{11} is the original 2.0 plus $0.00265 = 2.00265$.

This is quite a small change, but over many hundreds or thousands of iterations the weights will eventually settle down to a configuration so that the well trained neural network produces outputs that reflect the training examples.