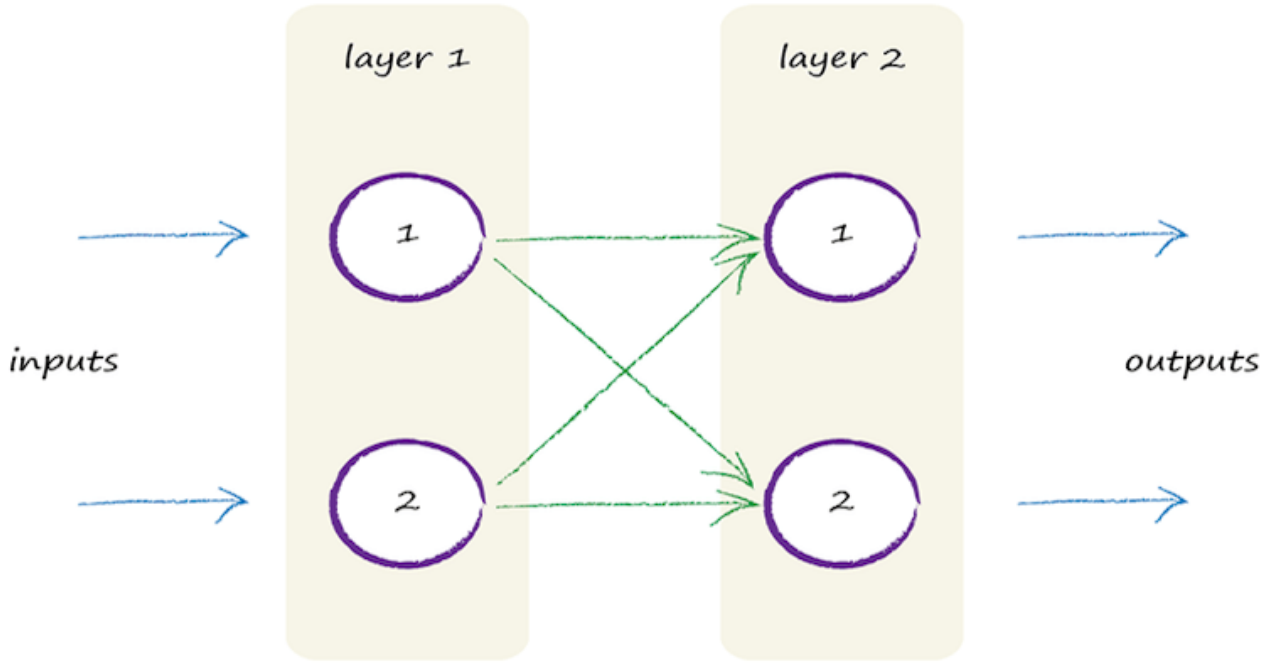
Following Signals Through A Simpler Network

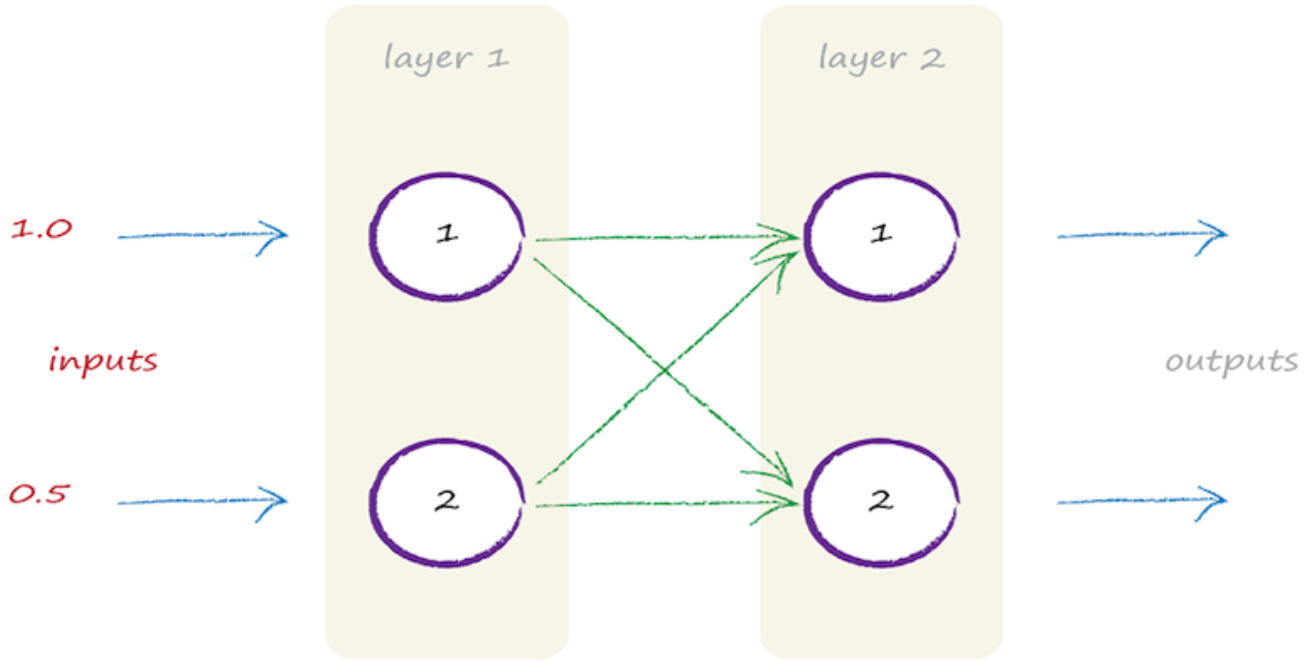
To get a better understanding of things, we will build a simpler neural network with only two inputs and see what really happens inside the layers before we get the final output.

That picture of 3 layers of neurons in the previous lesson, with each neuron connected to every other in the next and previous layers, looked pretty amazing. But the idea of calculating how signals progress from the inputs through the layers to become the outputs seems a little daunting and, well, too much like hard work!

I will agree that it is hard work, but it is also important to illustrate its working, so we always know what is really happening inside a neural network, even if we later use a computer to do all the work for us. So we'll try doing the workings out with a smaller neural network with only two layers, each with two neurons, as shown below:



Let's imagine the two inputs are 1.0 and 0.5. The following shows these inputs entering this smaller neural network.



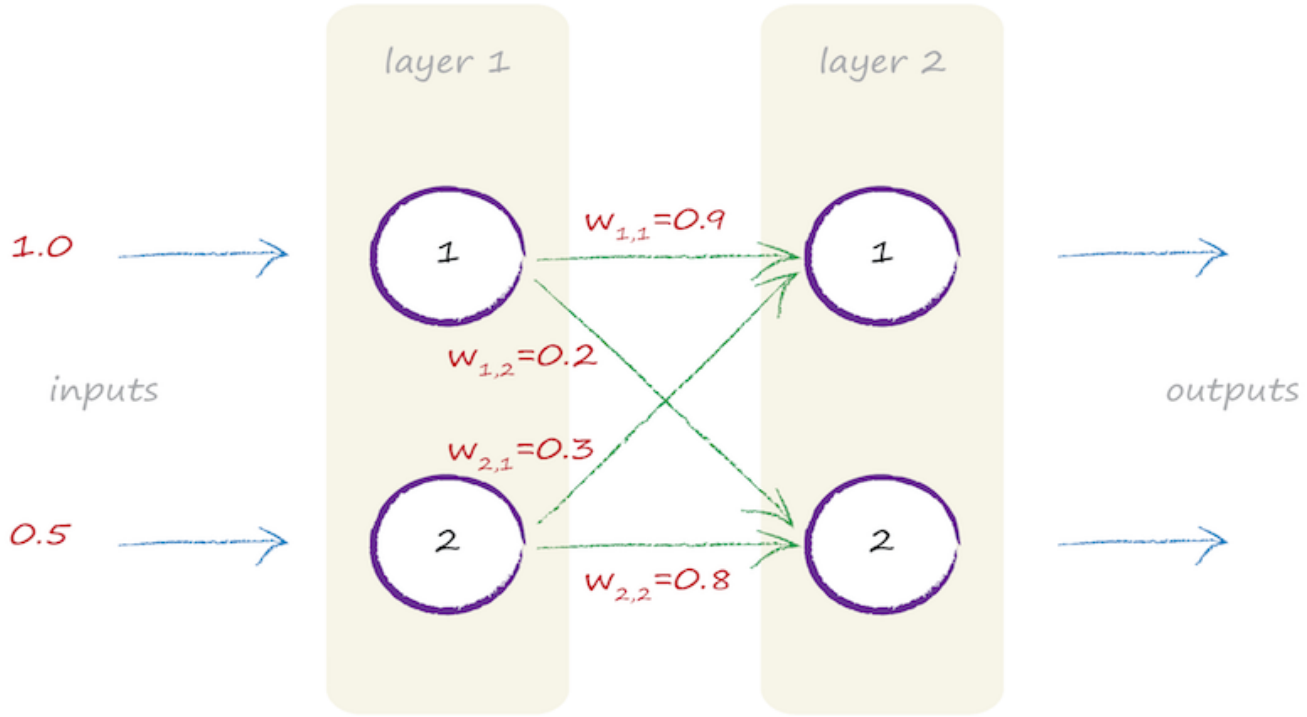
Just as before, each node turns the sum of the inputs into an output using an activation function. We will also use the sigmoid function

$$y = \frac{1}{1 + e^{-x}}$$

that we saw before, where x is the sum of incoming signals to a neuron and y is the output of that neuron. What about the weights? That's a very good question — what value should they start with? Let's go with some random weights:

- $W_{1,1}$ = 0.9
- $W_{1,2}$ = 0.2
- $W_{2,1}$ = 0.3
- $W_{2,2}$ = 0.8

Random starting values aren't such a bad idea, and it is what we did when we chose an initial slope value for the simple linear classifiers earlier on. The random value got improved with each example that the classifier learned from. The same should be true for neural networks link weights.



There are only four weights in this small neural network, as that's all the combinations for connecting the 2 nodes in each layer. The following diagram shows all these numbers now marked. The next lessson will cover all the calculations of this neural net that we just built. See you!