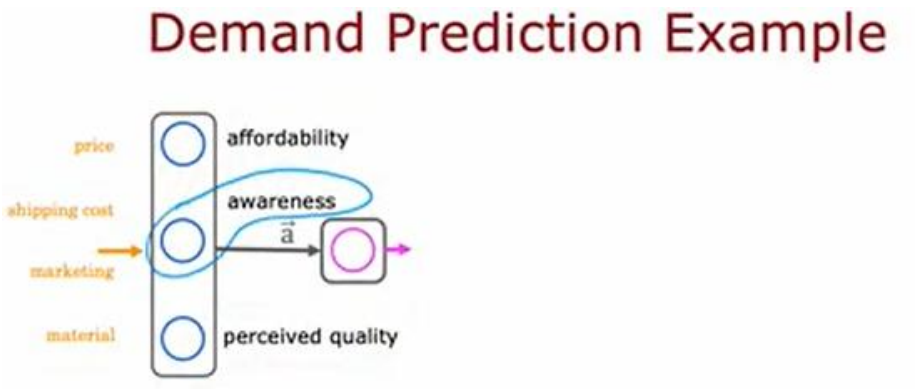
Alternatives to sigmoid activation

So far, we've been using the sigmoid activation function in all the nodes in the hidden layers and in the output layer. And we started that way because we were building up neural networks by taking logistic regression and creating a lot of logistic regression units and string them together. But if you use other activation functions, your neural network can become much more powerful.

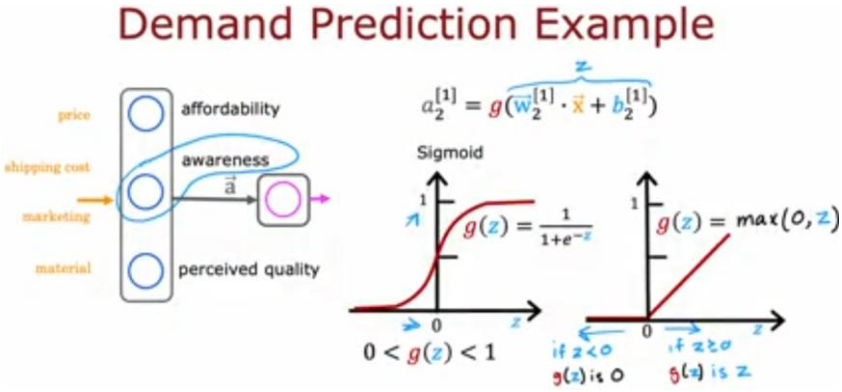
Let's take a look at how to do that. Recall the demand prediction example from last week where given price, shipping cost, marketing, material, you would try to predict if something is highly affordable. If there's good awareness and high perceived quality and based on that try to predict it was a top seller. But this assumes that awareness is maybe binary is either people are aware or they are not.



But it seems like the degree to which possible buyers are aware of the t shirt you're selling may not be binary, they can be a little bit aware, somewhat aware, extremely aware or it could have gone completely viral. So rather than modeling awareness as a binary number 0, 1, that you try to estimate the probability of awareness or rather than modeling awareness is just a number between 0 and 1.

Maybe awareness should be any non negative number because there can be any non negative value of awareness going from 0 up to very very large numbers. So whereas previously we had used this equation to calculate the activation of that second hidden unit estimating awareness where g was the sigmoid function and just goes between 0 and 1. If you want to allow a,1, 2 to potentially take on much larger positive values, we can instead swap in a different activation function.

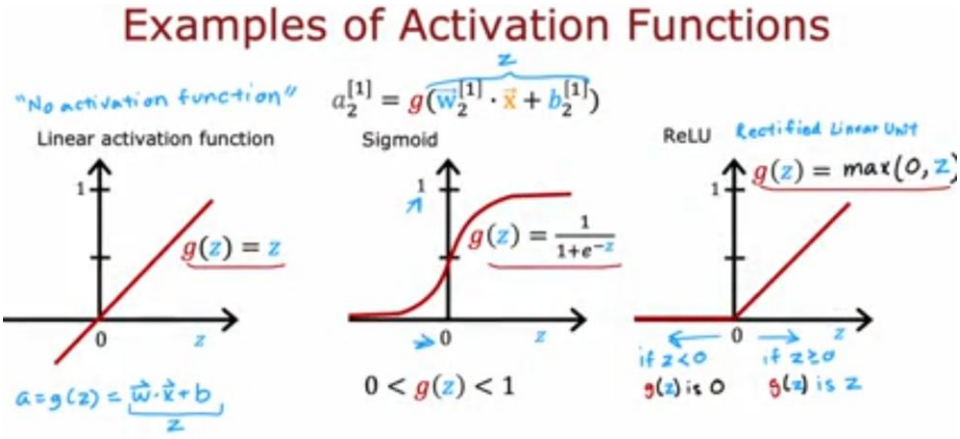
It turns out that a very common choice of activation function in neural networks is this function. It looks like this.



It goes if z is this, then g(z) is 0 to the left and then there's this straight line 45° to the right of 0. And so when z is greater than or equal to 0, g(z) is just equal to z. That is to the right half of this diagram. And the mathematical equation for this is g(z) equals max(0, z). Feel free to verify for yourself that max(0, z) results in this curve that I've drawn over here. And if a 1, 2 is g(z) for this value of z, then a, the deactivation value cannot take on 0 or any non negative value. This activation function has a name.

It goes by the name ReLU with this funny capitalization and ReLU stands for again, somewhat arcane term, but it stands for rectified linear unit. Don't worry too much about what rectified means and what linear unit means. This was just the name that the authors had given to this particular activation function when they came up with it.

But most people in deep learning just say ReLU to refer to this g(z). More generally you have a choice of what to use for g(z) and sometimes we'll use a different choice than the sigmoid activation function. Here are the most commonly used activation functions. You saw the sigmoid activation function, g(z) equals this sigmoid function.



On the last slide we just looked at the ReLU or rectified linear unit g(z) equals max(0, z). There's one other activation function which is worth mentioning, which is called the linear activation function, which is just g(z) equals to z. Sometimes if you use the linear activation function, people will say we're not using any activation function because if a is g(z) where g(z) equals z, then a is just equal to this wx plus b z.

And so it's as if there was no g in there at all. So when you are using this linear activation function g(z) sometimes people say, well, we're not using any activation function. Although in this class, I will refer to using the linear activation function rather than no activation function. But if you hear someone else use that terminology, that's what they mean.

It just refers to the linear activation function. And these three are probably by far the most commonly used activation functions in neural networks. Later this week, we'll touch on the fourth one called the softmax activation function. But with these activation functions you'll be able to build a rich variety of powerful neural networks. So when building a neural network for each neuron, do you want to use the sigmoid activation function or the ReLU activation function? Or a linear activation function? How do you choose between these different activation functions?