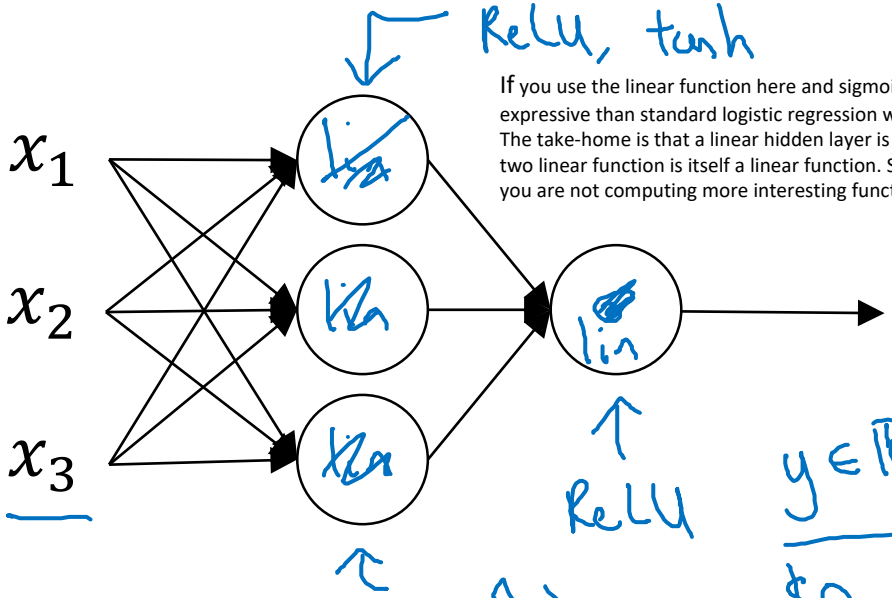
deeplearning.ai

One hidden layer Neural Network

Why do you need non-linear activation functions?

Activation function

Deep networks with many many layers, many many hidden layers and turns out that if you use a linear activation function or alternatively if you don't have an activation function then no matter how many layers your network has, always doing is computing a linear activation function, so you might as well not have any hidden layers.



If you use the linear function here and sigmoid function here, then this model is no more expressive than standard logistic regression without any hidden layer.
The take-home is that a linear hidden layer is more or less useless,because the composition of two linear function is itself a linear function. So unless you throw a non-linearty in there, then you are not computing more interesting functions even as you good deeper in the network

$y \in \mathbb{R}$

$\$0 \ldots \$1{,}000{,}000$s

$\hat{y} \geq 0$.

$$a^{[1]} = z^{[1]} = W^{[1]}x + b^{[1]}$$

$$a^{[2]} = z^{[2]} = W^{[2]}a^{[1]} + b^{[2]}$$

$$a^{[2]} = W^{[2]}\left(\underbrace{W^{[1]}x + b^{[1]}}_{a^{[1]}}\right) + b^{[2]}$$

$$= \underbrace{\left(W^{[2]}W^{[1]}\right)}_{W'}x + \underbrace{\left(W^{[2]}b^{[1]} + b^{[2]}\right)}_{b'}$$

$$= W'x + b'$$

$g(z) = z$

the neural network is just outputting a linear function of the input.

Given x:

$$\begin{cases} z^{[1]} = W^{[1]}x + b^{[1]} \\ a^{[1]} = g^{[1]}(z^{[1]}) \end{cases}$$ forward propagation equations

$z^{[1]}$

$$z^{[2]} = W^{[2]}a^{[1]} + b^{[2]}$$

$$a^{[2]} = g^{[2]}(z^{[2]})$$

$z^{[2]}$

"linear activation function"

Identity activation function

$g(z) = z$

Andrew Ng