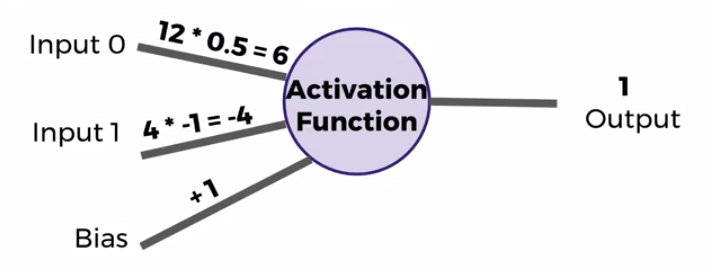
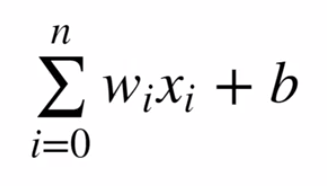
Introduction to Perceptron, has similarity to biological Neuron

* We have 2 Inputs and a single output
* Inputs will be values of features
* Inputs are multiplied by a weight
* Weights initially start off as random
* Inputs are multiplied by the weights
* Then these results are passed on to the activation function, there are many activation functions to choose from.
* If sum of the inputs is positive then return 1, if negative return 0
* In case Zero is selected as the initial weights, then the output will be 0 as well. So, we include another input bias as 1.

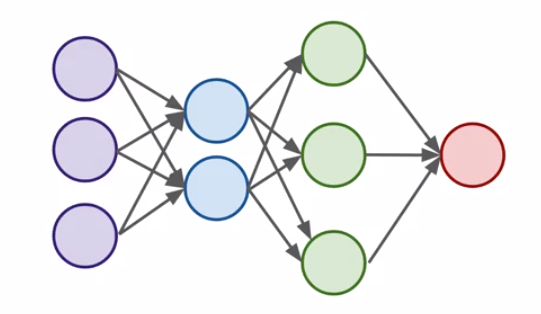


Representing perceptron model mathematically



w – Weight, x – Input values, b – bias

Multiple Perceptron Network



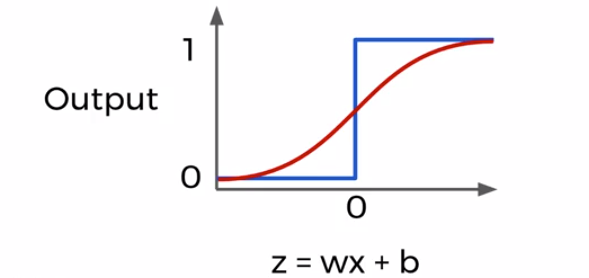
1 input layer, 2 hidden layers and 1 output layer

* **Input** Layer – Real values from the input data
* **Hidden** Layers – Layers in between input and output, 3 or more layers is Deep Network.
* **Output** Layer – Final Estimate of the output

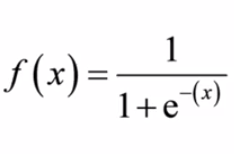
More the layers the level of abstraction increases.

When you have an activation function just outputting 0 or 1, small changes aren’t reflected

It would be better if a have an activation function, which outputs more detailed value, something like a red line.



Below function can produce values similar to the red line, this is a sigmoid function



Few other activation functions

