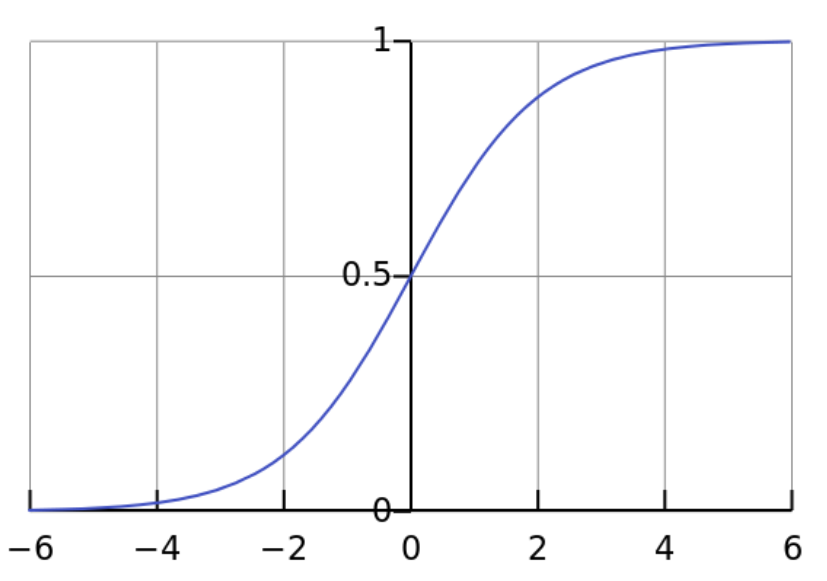
In neural networks activation functions work like an abstraction of the rate of action potential firing in a cell. The working of an activational function is that which decides whether a neuron should fire an input or not. The output is calculated as the sum of weighted input plus the bias and decided if it should be fires or not based on the outcome generated. The decision is taken by the activation function to fire or not.

Some of the activation functions used by us are

1. Sigmoid Function

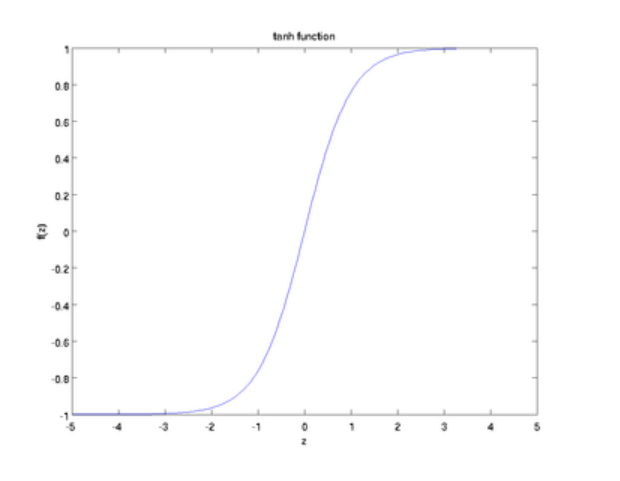
**A=**



This function is like a step function with values ranging from (0,1). This is a non-linear function. The specialty of this function is that for input range -2 to 2 the change in output is significant that is the slope is steep for the curve showing the tendency to bring the values to either side of the curve making clear distinctions on prediction. Moreover, since the range is (0,1) unlike a linear function that has a range (-∞, ∞), the activations are bound in a range. The problem with sigmoid is that it gives rise to the problem of “vanishing gradient” – the gradient is that small or vanished when the activations reach the horizontal part of the curve.

1. Tanh function

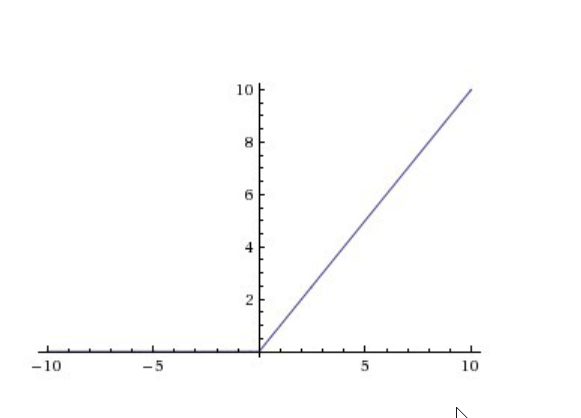
**A=**



Tanh function is similar to sigmoid function but the values range here form (-1,1), so there is no worry of activation blow ups. The major difference is that the gradient is stronger than sigmoid. Depending on the gradient strength the function is chosen between sigmoid and tanh

1. Relu function

**A=max(0,x)**



Relu function is a nonlinear function that ranges from [0, ∞). Since it is not bound there is a chance of activation blow up. Any function can be approximated using Relu. Relu function reduce the activation costs. For activations in the region where gradient is 0, the neurons will stop responding to variations in error/ input.

Adam Optimizer

Adam is an acronym that is derived from Adaptive moment estimation. Adam optimization is straightforward and computationally efficient. It also uses less memory. It is used for problems that are have large data and parameters and is appropriate for non-stationary objectives. For problems with noisy and sparse gradients, this method is ideal.

Binary Cross Entropy

The classic way of generating an error model in neural networks is called binary cross entropy. It is defined as,

If our prediction is y’ and real value is y, then binary cross entropy is defined as

b(y,y’)=-ylog(y’)-(1-y)log(1-y’)

Epoch and batch\_size

In neural network one epoch is onefoward and backward pass of all training examples and the batch size is the number of training examples in one forward/ backward pass.