How Neural Network Works?

# Neural Networks Forward Propagation

Neural networks can be thought of as a function that can map between inputs and outputs. In theory, no matter how complex that function is neural networks should be able to approximate that function.

However, most of supervised learning, if not all, is about learning a function that maps given X and Y, and later using that function to find the appropriate Y for a new X. If so, what is the difference between traditional machine learning algorithms and neural networks?

The answer is something known as Inductive Bias. The term might look new. But it is nothing but the assumption that we place upon the relationship between X and Y before fitting a machine learning model into it.

For example, if we think the relationship between X and Y is linear, we could use linear regression. The Inductive Bias of linear regression is that the relationship between X and Y is linear. Hence, it fits a line or hyperplane to the data.

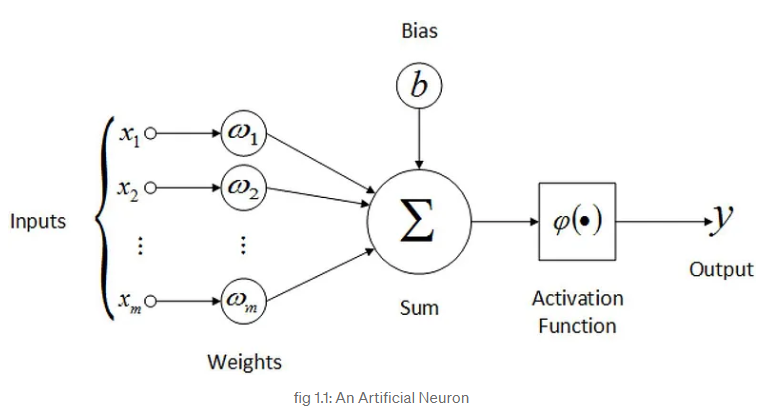
But when a non-linear and complex relationship exists between the X and Y, a linear regression algorithm might not do a great job in predicting Y. In this case, we might require a curve or a multi-dimensional curve to approximate that relationship. The main advantage of neural networks is it’s Inductive Bias is very weak and hence, no mater how complex this relationship or function is, the network is somehow able to approximate it.

But also, based on the complexity of the function, we might have to manually set the number of neurons in each layer and the number of layers in the network. This is usually done by trial and error and experience. Hence, these parameters are called hyperparameters.

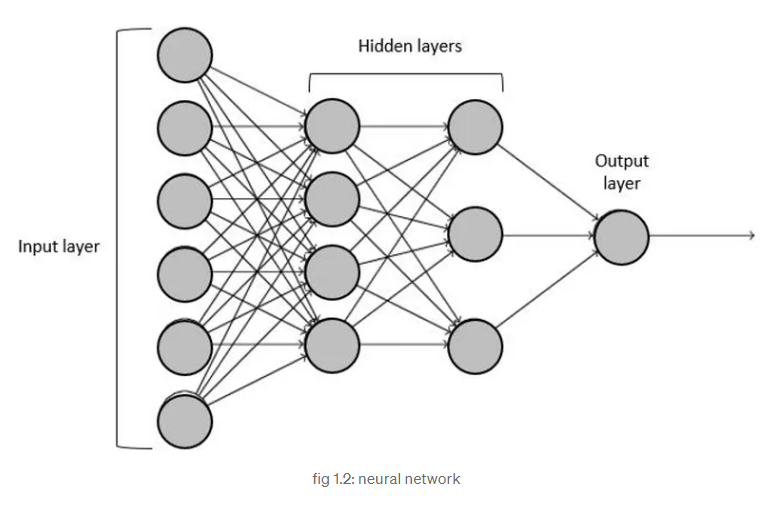
*Neural networks are nothing but complex curve-fitting machines.*

***Josh Starmer***

# Architecture and working of neural networks

Before we see why whatever neural networks, it would be appropriate to show what neural networks do. Before understanding the architecture of a neural network, we need to look into what a neuron does first.

Each input to an artificial neuron has a weight associated with it. The inputs are first multiplied with their respective weights and a bias is added to the result. We can call this the weighted sum. The weighted sum then goes through an activation function, which is basically a non-linear function.

So, an artificial neuron can be thought of as a simple or multiple linear regression model with an activation function at the end. Having said that, let’s move on to neural network architecture.

A neural network typically has multiple layers with each layer having multiple neurons, where all the neurons from one layer are connected to all the neurons in the next layer and so on.

In the picture above, we have 4 layers. The first layer is the input layer which looks like it contains 6 neurons, but in reality, it is nothing, but the data is served as input to the neural network (There are 6 neurons because the input data probably has 6 columns). The final layer is the output layer. The number of neurons in the final years and the first layer is predetermined by the dataset and the type of the problem (number of output classes and such). The number of neurons in the hidden layers and the number of hidden layers are to be chosen by trail and error.

A neuron from layer i will take the output of all the neurons from layer i-1 as inputs and calculates the weighted sum addas a bias to it and then finally, send it through an activation function, as we saw above in the case of an artificial neuron. The first neuron from the first hidden layer will be connected to all the inputs from the previous layer (input layer).

Similarly, the second neuron from the first hidden layer will also be connected to all the inputs from the previous layer, and so on for all the neurons in the first hidden layer. For neurons in the second hidden layer, outputs of the previous hidden layer are considered as the inputs and each of these neurons are connected to all the previous neurons, likewise.