Introduction to neural network

‘Real life is not linear ‘
‘Because we want to be better forecaster we need better model ‘

Neural networks are trying to accomplish the same thing as any other model, to make good predictions. We have a set of inputs and a set of target values, and we are trying to get predictions that match those target values as closely as possible.

We can regard the linear model as a simple neural network since it has 2 layers the input layer and output layer

Simple neural network

In the minimal example, we trained a neural network that had no depth. There were solely an input layer and an output layer. Moreover, the output was simply a linear combination of the input

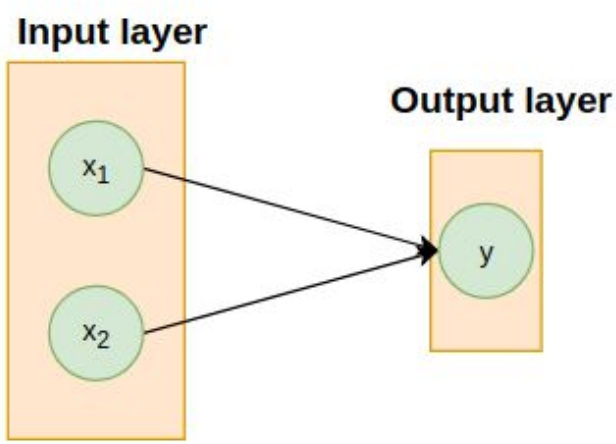


Fig 1.

since we are talking about the simple neural network then there's a complex one

Neural network

Neural network steps on a linear combination but adds a non-linearity to each one of them. Mixing linear combinations and non-linearity allow us to model arbitrary functions.

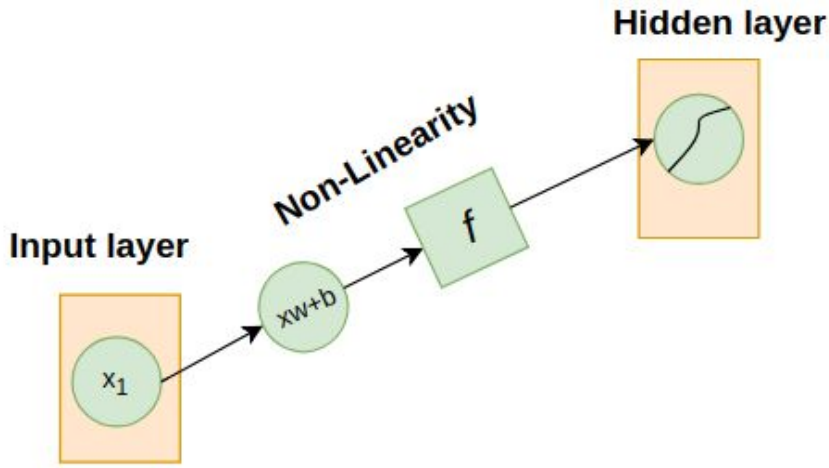


Fig 2.

I bet you are wondering why we need that non-linearity after linear combination to create the next layer.

" Non-linearity is needed so we can break the linearity and represent more complicated relationships "
An important consequence of including non-linearity is the ability to stack layers. **Stacking layers** is the process of placing one layer after another in a meaningful way.

You can see the neural network without the non-linearity only linear combination. We are going to explain this more in the section of **non-linearity**.

We cannot stack the layers when we only have linear relationships.

Let's take the temperature as an example to understand more the non-linearity.
If the temperature starts decreasing, our brain will tell us whether it is cold enough for us to put on a jacket.
We can say that putting the jacket is like a binary action,
putting jacket: 1,
no jacket : 0,
in this case, our brain acts like non-linearity.

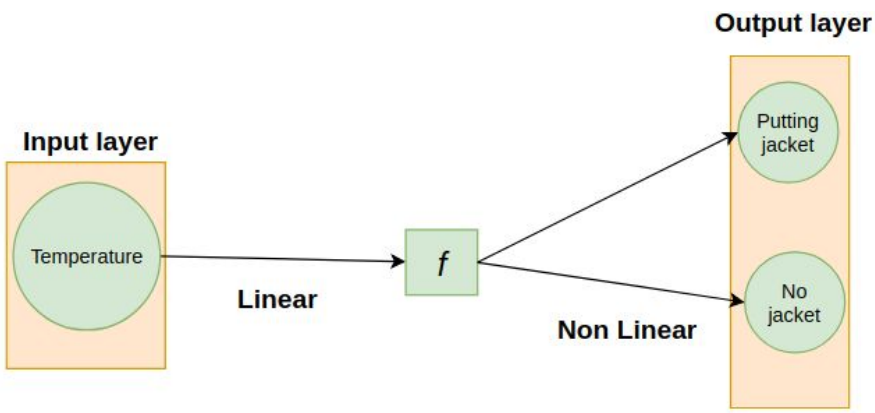


Fig 3.

The elements of the neural network

Now let's see a deep neural network and explain each element.
This is a deep neural network with 5 layers

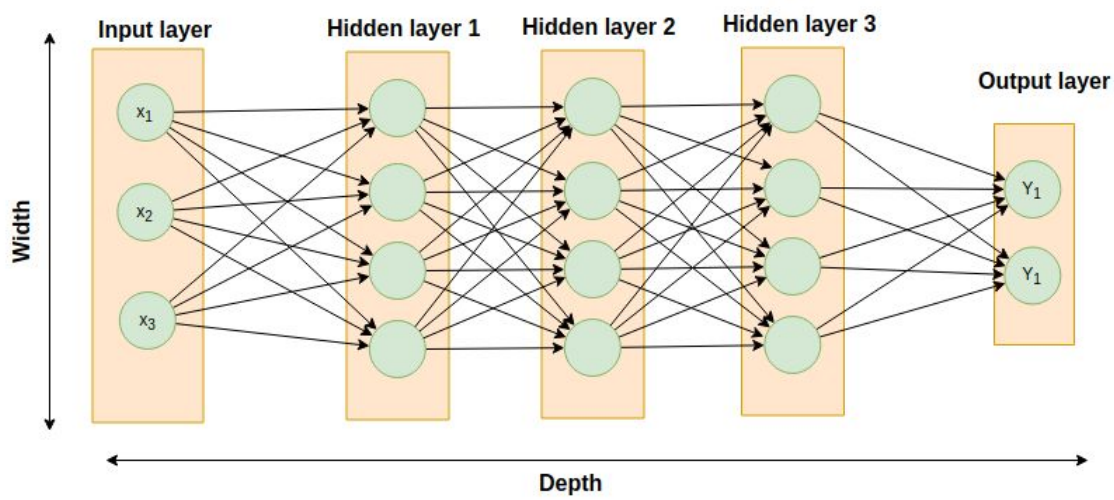


Fig 4.

First things we must explain each element of the neural network

a - Layer is the building block of the neural network,
b - Hidden layer, we call the hidden as we know the inputs and outputs the rest remains hidden.

c - Width: The number of nodes in the layer is often referred to as the width of the layer.
how can we define the width of the neural network?

The width of the neural network is the number of units or neurons of the largest layer . in Fig 3. the largest are the layers 3, 2, and 1 since they have the same units so the width of this neural network is 4.

d - Depth is equal to the number of layers or the number of hidden layers, but we are interested in the number of hidden layers.

The last element in this neural network is the arrow.

e - Arrow represents the mathematical transformation of certain values.

Non-linearity

Another example to explore more about non-linearity, and to understand the importance of the non-linearity, and why we need it.

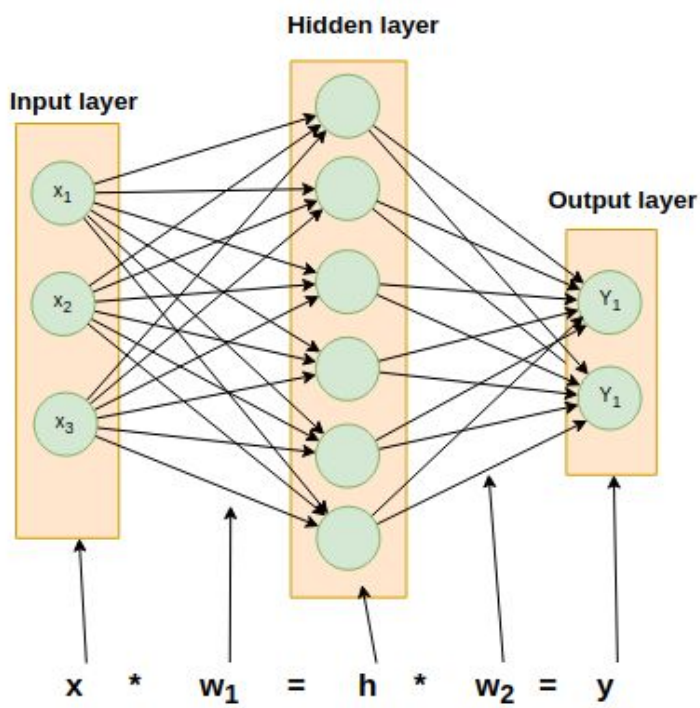


Fig 5.

With :

$x : (1 * 3)$, $w_1 : (3 * 6)$, $h : (1 * 6)$, $w_2 : (6 * 2)$ and $y : (1 * 2)$

Now we should develop this equation

$$h = x * w_1$$

$$y = h * w_2$$

$$y = x * w_1 * w_2$$

put

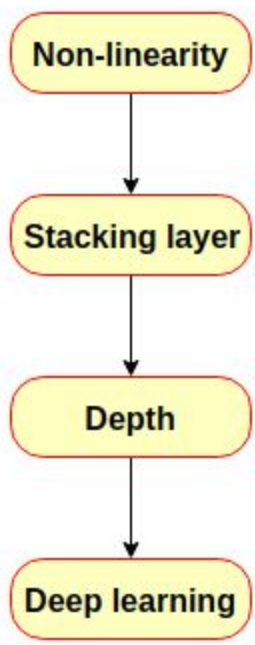
$$w^* = w_1 * w_2$$

Finally, the equation becomes

$$y = x * w^*$$

In this case, The hidden layer is completely useless
We can just train this simple linear model and we would get the same result . in mathematics this is an obvious fact, however in machine learning is not clear from the beginning.
Even if we add 80, 90 or 100 layers the problem would be simplified to a simple transformation, that is the reason we need the non-linearity.

To summarize, In order to have a deep neural network and to find the complex relationships through arbitrary function we need the non-linearity.



After finishing this section we have to know that in Machine learning context the nonlinearity is called activation function and one of the common activation functions is the sigmoid or the logistic function
Mathematical formula :

$$\sigma(a) = \frac{1}{1+e^{-a}} = \frac{e^a}{1+e^a}$$

Forward propagation

In this part, we will discuss how it goes from the input to the output
Let's take this simple example and explore it

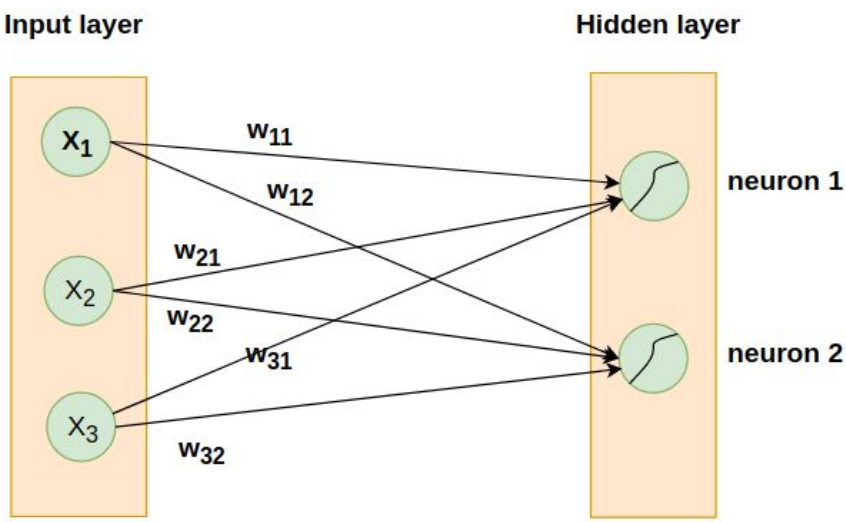


Fig 6.

Now let's calculate the output Y_1 and Y_2 using the following formula

$$Y_1 = w_1 X_1 + w_2 X_2 + w_3 X_3 + BiaseNeuron1$$

In neural networks, each neuron has its own bias, but in terms of mathematics we call it the intercept
After this step, we must apply the activation function as we see we have a lot of activation functions but we go with the popular one which is sigmoid

$$Sigmoid(Y_1) = Predected\ Probability$$

We can use the linear algebra and we take advantage of using matrix, so instead of calculating each input separately, we calculate all the outputs together

$$\begin{bmatrix} w_{11} & w_{21} & w_{31} \\ w_{12} & w_{22} & w_{32} \end{bmatrix} * \begin{bmatrix} X_1 \\ X_2 \\ X_3 \end{bmatrix} + \begin{bmatrix} bias_1 \\ bias_2 \end{bmatrix} = \begin{bmatrix} Y_1 \\ Y_2 \end{bmatrix}$$

By repeatedly calculating Y and applying the activation function to it for each successive layer, we can move on from the **input** to the **output**, this process called **FORWARD PROPAGATION**

The objective of **FORWARD PROPAGATION** is to calculate the activation at each neuron for each successive hidden layer until we reach the output

Backpropagation

Build an intuitive understanding of how and why backpropagation works
We observe that the forward propagation is moving forward through the neural network from the input to the output so the backpropagation is the reverse except we are moving the error backward through our model

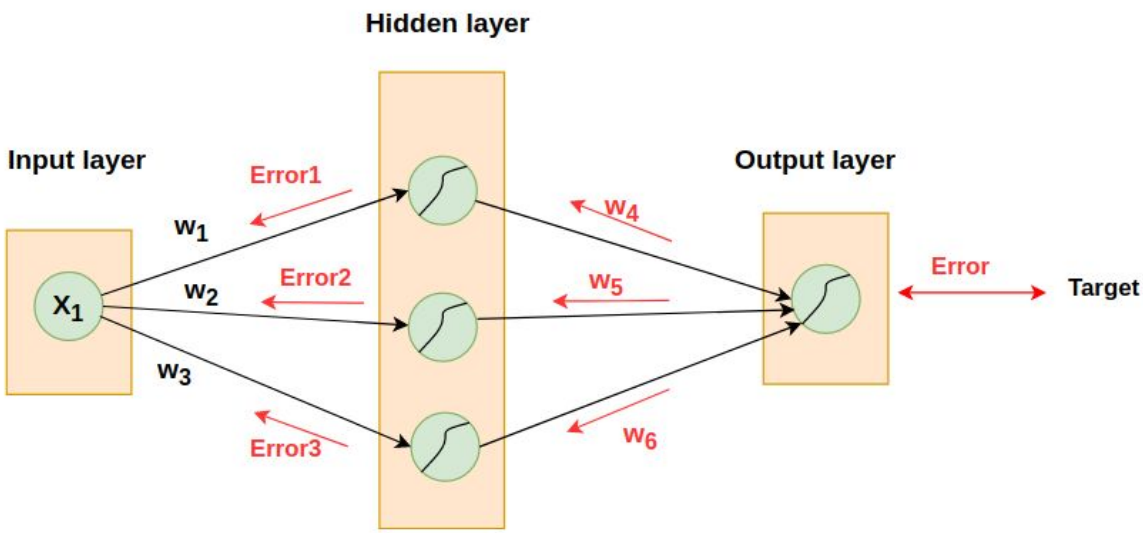


Fig 7.

you might be wondering, why do we care about the error of each neuron?
these weights and biases across the entire neural network are also the dials that we tweak to change the predictions made by the model.

the magnitude of the error of a specific neuron is directly proportional to the impact of that neuron's output, so if a particular neuron has a much larger error then all their neurons at the end will have a greater impact on our model.

Eventually, the obtained output is compared to the target. So we must update all the weights related to the input and the hidden layer, and keep remembering that there is no target for the hidden layer.
Based on the error we can adjust the weight to get better prediction.

Finally, In this article, We have discussed the regular neural network, and how it works, however, there are different types of neural networks, such as CNN (Convolutional Neural Networks), RNN (Recurrent Neural Networks).

Furthermore, we provided the sigmoid function of activation, but keep in mind there are others like Relu, Softmax, SoftPlus, Softsign, Hardtanh …