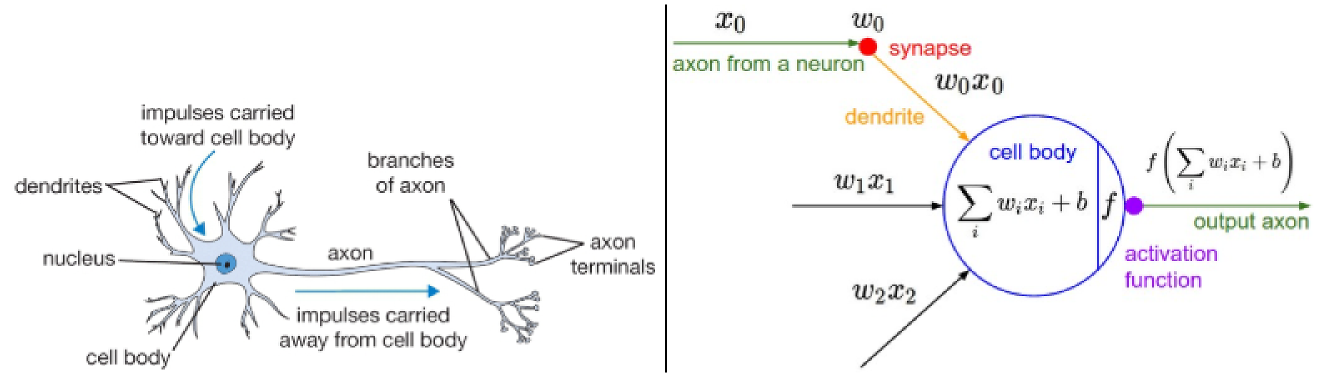
Activation Function

> What is an activation function? an activation function is a function that is added into an artificial neural network in order to help the network learn complex patterns in the data. When comparing with a neuron-based model that is in our brains, the activation function is at the end deciding what is to be fired to the next neuron. That is exactly what an activation function does in an ANN as well. It takes in the output signal from the previous cell and converts it into some form that can be taken as input to the next cell. The comparison can be summarized in the figure below.



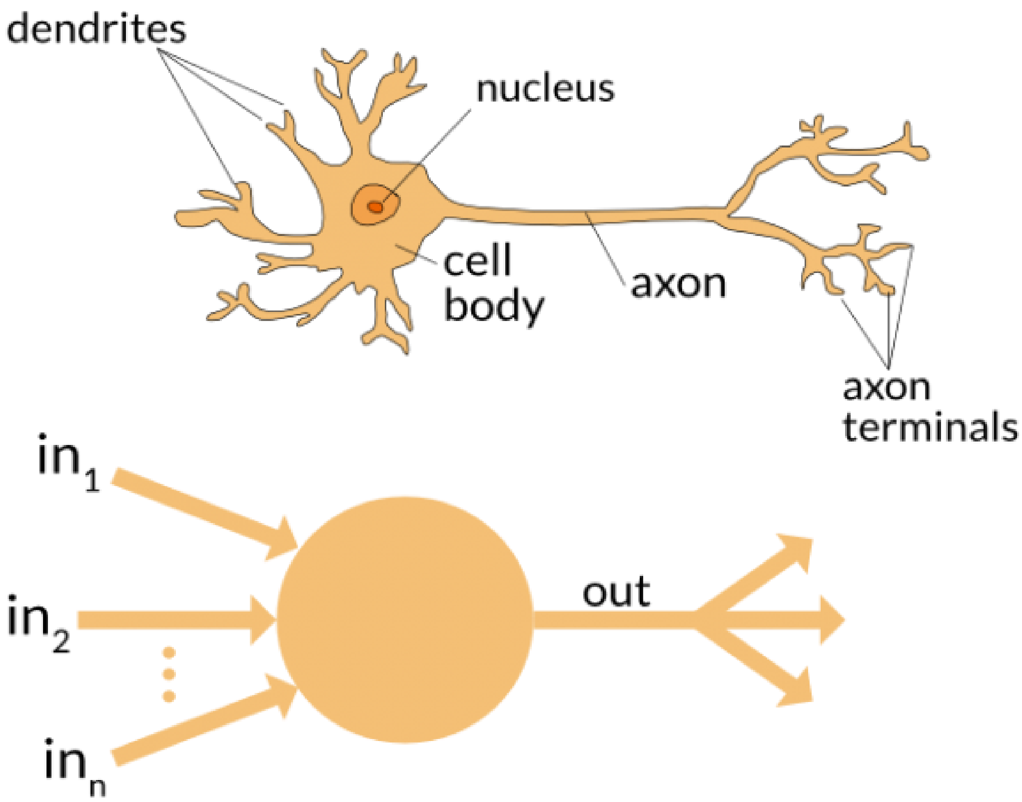
A cartoon drawing of a biological neuron (left) and its mathematical model (right).

Types of activation functions :

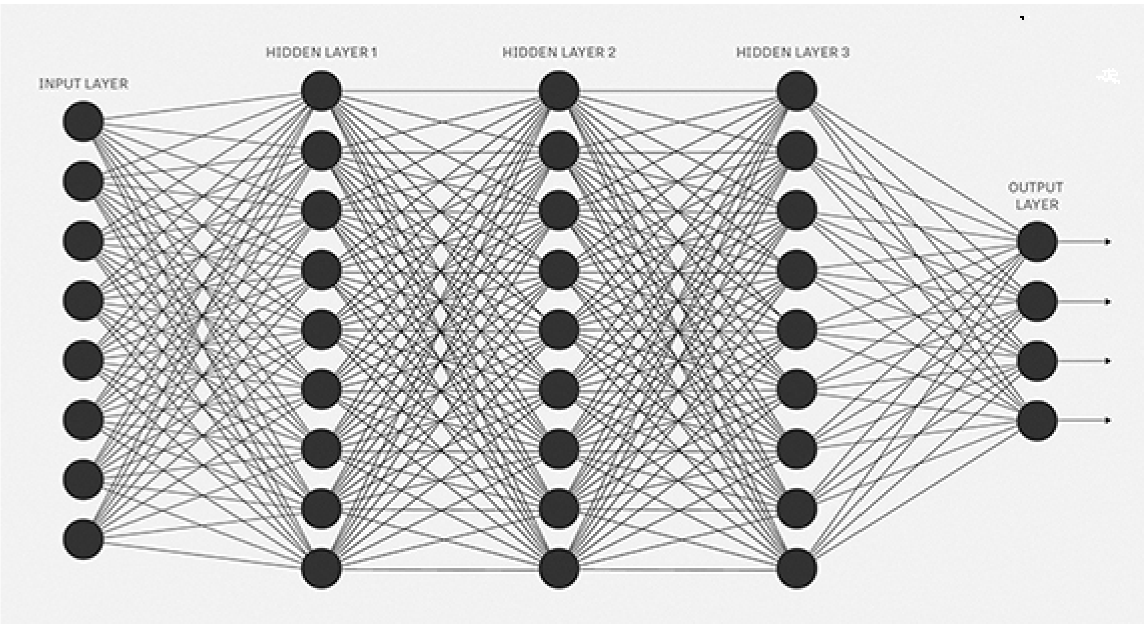
```
1. Binary Step\
2. Linear\
3. Sigmoid\
4. Tanh\
5. ReLU\
6. Leaky ReLU\
7. Parameterised ReLU\
8. Exponential Linear Unit\
9. Swish\
10. Softmax
```

Brief overview of neural networks

> The brain receives the stimulus from the outside world, does the processing on the input, and then generates the output. As the task gets complicated, multiple neurons form a complex network, passing information among themselves.



An Artificial Neural Network tries to mimic a similar behavior. The network you see below is a neural network made of interconnected neurons. Each neuron is characterized by its weight, bias and activation function.



Deep Learning Intermediate Python Technique Overview

```
Activation function is one of the building blocks on Neural Network
Learn about the different activation functions in deep learning
Code activation functions in python and visualize results in live coding window
```

This article was originally published in October 2017 and updated in January 2020 with three new activation functions and python codes. Introduction

The Internet provides access to plethora of information today. Whatever we need is just a Google (search) away. However, when we have so much information, the challenge is to segregate between relevant and irrelevant information.

When our brain is fed with a lot of information simultaneously, it tries hard to understand and classify the information into “useful” and “not-so-useful” information. We need a similar mechanism for classifying incoming information as “useful” or “less-useful” in case of Neural Networks.

This is important in the way a network learns because not all the information is equally useful. Some of it is just noise. This is where activation functions come into picture. The activation functions help the network use the important information and suppress the irrelevant data points.

Let us go through these activation functions, learn how they work and figure out which activation functions fits well into what kind of problem statement.

Table of Contents

Brief overview of neural networks

Before I delve into the details of activation functions, let us quickly go through the concept of neural networks and how they work. A neural network is a very powerful machine learning mechanism which basically mimics how a human brain learns.

The brain receives the stimulus from the outside world, does the processing on the input, and then generates the output. As the task gets complicated, multiple neurons form a complex network, passing information among themselves.

An Artificial Neural Network tries to mimic a similar behavior. The network you see below is a neural network made of interconnected neurons. Each neuron is characterized by its weight, bias and activation function.

---The input is fed to the input layer, the neurons perform a linear transformation on this input using the weights and biases.

x = (weight * input) + bias

Post that, an activation function is applied on the above result.

Finally, the output from the activation function moves to the next hidden layer and the same process is repeated. This forward movement of information is known as the forward propagation.

What if the output generated is far away from the actual value? Using the output from the forward propagation, error is calculated. Based on this error value, the weights and biases of the neurons are updated. This process is known as back-propagation.---

1. Binary Step Function

```
f(x) = 1, x>=0
     = 0, x<0
```

2. Linear Function

```
f(x)=ax
```

3. Sigmoid

```
f(x) = 1/(1+e^-x)
```

4. Tanh

```
tanh(x)=2sigmoid(2x)-1
```

5. ReLU

```
f(x)=max(0,x)
```

6. Leaky ReLU

```
f(x)= 0.01x, x<0
    =   x, x>=0
```

7. Parameterised ReLU

```
f(x) = x, x>=0
    = ax, x<0
```

8. Exponential Linear Unit

```
f(x) = x,   x>=0
    = a(e^x-1), x<0
```

9.Swish

```
f(x) = x*sigmoid(x)
f(x) = x/(1-e^-x)
```

10.Softmax

```
def softmax_function(x):
    z = np.exp(x)
    z_ = z/z.sum()
    return z_
```