Independent Study IMT 600

Introduction to Deep Learning

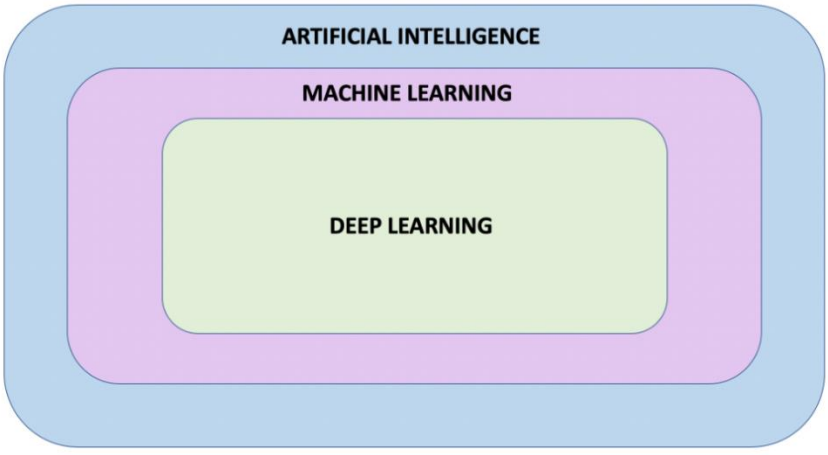
What is Deep Learning?

When studying Machine Learning you will come across many different terms such as artificial intelligence, machine learning, neural network, and deep learning. But what do these terms actually mean and how do they relate to each other?

Below I give a brief description of these terms:

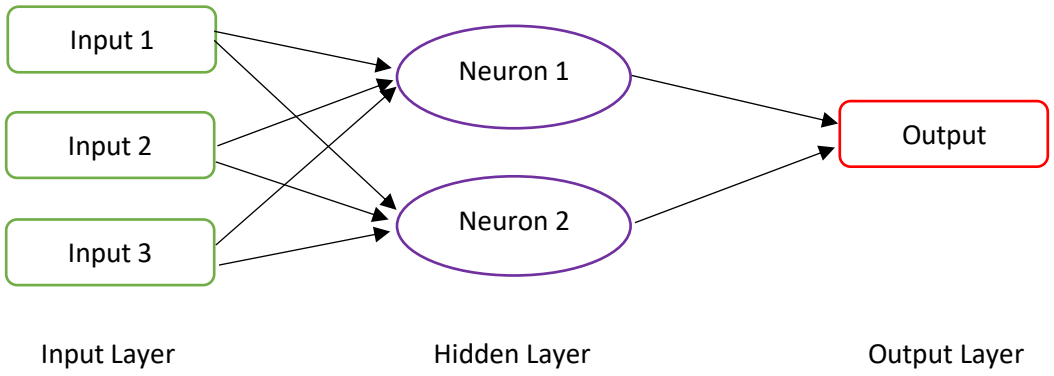
Artificial Intelligence: A field of computer science that aims to make computers achieve human-style intelligence. There are many approaches to reaching this goal, including machine learning and deep learning.

- **Machine Learning:** A set of related techniques in which computers are trained to perform a particular task rather than by explicitly programming them.
- **Deep Learning:** A subfield of machine learning that uses multi-layered neural networks. Often, "machine learning" and "deep learning" are used interchangeably.



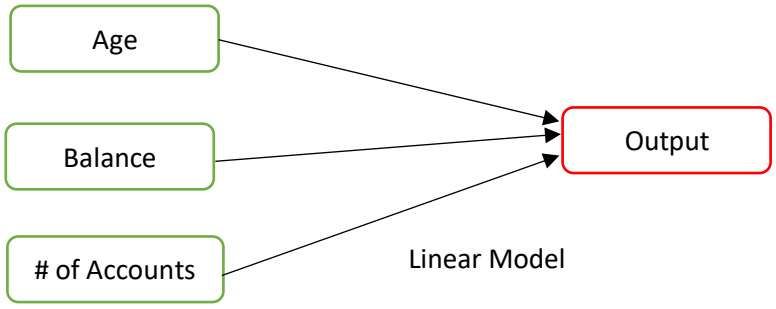
What are neural networks?

Neural networks are a set of algorithms, modeled loosely after the human brain, that are designed to recognize patterns. Neural networks are powered by neurons which are tiny units arranged in a series of layers connected to one another. One of these layers is called the input unit which is designed to receive different forms of information from the outside world and then recognize, interpret, and classify. Another unit is output and sits on the opposite end of the network awaiting the result of the process. In between the input and output are hidden units which perform most of the work determining how to process the information coming into the inputs.



Why neural network models over classic regression models?

Imagine you are working in a bank and you want to predict how many transactions each customer will make next year depending on the variables like age, number of account and bank balance. The linear model assumes that the input variables has an effect on the predicted outcome separately. In other words, it does not capture the interaction between the input variables themselves.



On the other hand, Neural Networks captures the complex interactions between variables which helps in building very accurate models. Their ability to capture extremely complex interactions allows them to work with data like text, images, audio, video etc.

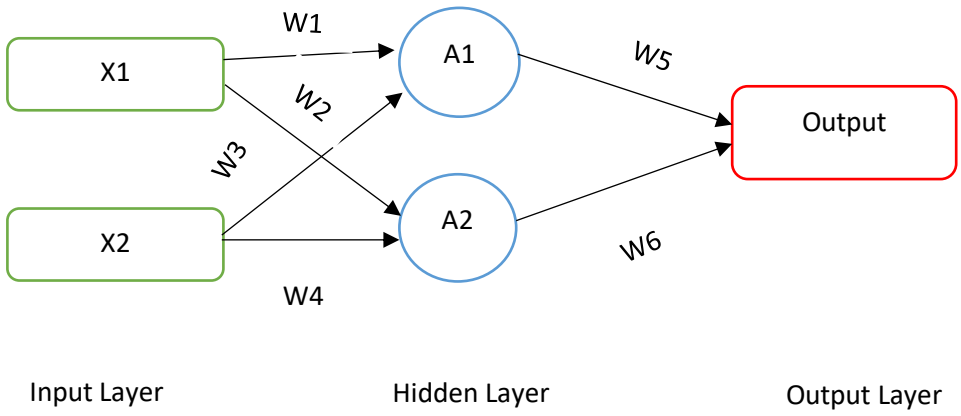
Neurons: Building Blocks

A neuron is the basic unit of a neural network. A neuron takes inputs, does mathematical operations with them, and produces one output. It contains the following components:

Input nodes: Each neuron receives input from the units to its left.

Weights (w): The connections between one unit and another are called weights and can be either positive or negative that can amplify or dampen that input, thereby assigning significance to inputs with regard to the task the algorithm is trying to learn.

Neuron: Includes a **bias term (b)** and an **activation function** that 'activates' the neuron based on the input. Activation function decides whether a neuron should be activated or not by calculating weighted sum and further adding bias with it. The purpose of the activation function is to **introduce non-linearity** into the output of a neuron.



Output calculation for each neuron:

There are many activation functions available for us to use. Some of the popular ones include Sigmoid and ReLU activation functions. We will be using ReLU activation function since it is the most commonly used in deep neural networks. The output of ReLU function is x if the input value is greater than 0. If input value is negative ReLU outputs 0. ReLU = max(0,X).

Z1: X1*W1 + X2*W3 + b1 (bias)

A1 = ReLU(0, Z1)

Z2: X1*W2 + X2*W4 + b2 (bias)

A2 = ReLU(0, Z2)

Z3 = A1*W5 + A2*W6

Output = ReLU(0, Z3)

By repeatedly calculating Z and applying the activation function to it for each successive layer, we can move from input to output. This process is known as **forward propagation**.

Training the Neural Network

Now that we know how the outputs are calculated, it is time to start evaluating the quality of the outputs and training our neural network. Once a value is predicted, the difference between that predicted value and the correct value is calculated. This difference is called the loss, and it is a measure of how well the model performed the mapping task. The value of the loss is calculated using a loss function like Mean Squared Error:

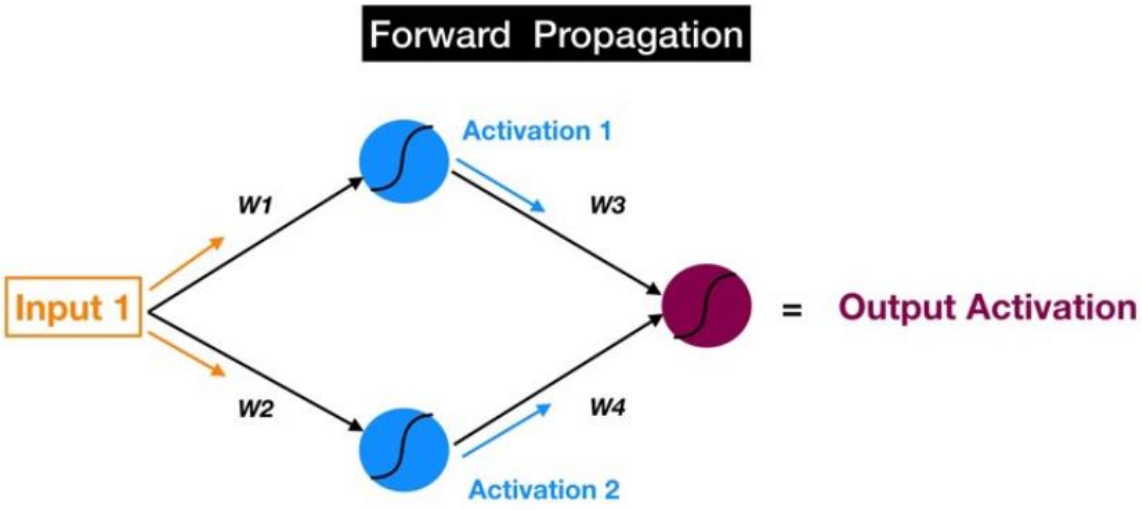
$$MSE = Sum\ [\ (\ Prediction - Actual\ )^2\ ]\ *\ (1\ /\ num\_observations)$$

Training the Neural Network is basically **adjusting the internal variables: weights and biases** to enable the model to best match the inputs to the outputs. This is achieved through an optimization process called Gradient Descent, which uses Numeric Analysis to find the best possible values to the internal variables of the model. Gradient descent iteratively adjusts parameters, nudging them in the correct direction a bit at a time until they reach the best values. In this case "best values" means that nudging them any more would make the model perform worse. The function that measures how good or bad the model is during each iteration is called the "loss function", and the goal of each nudge is to "minimize the loss function."

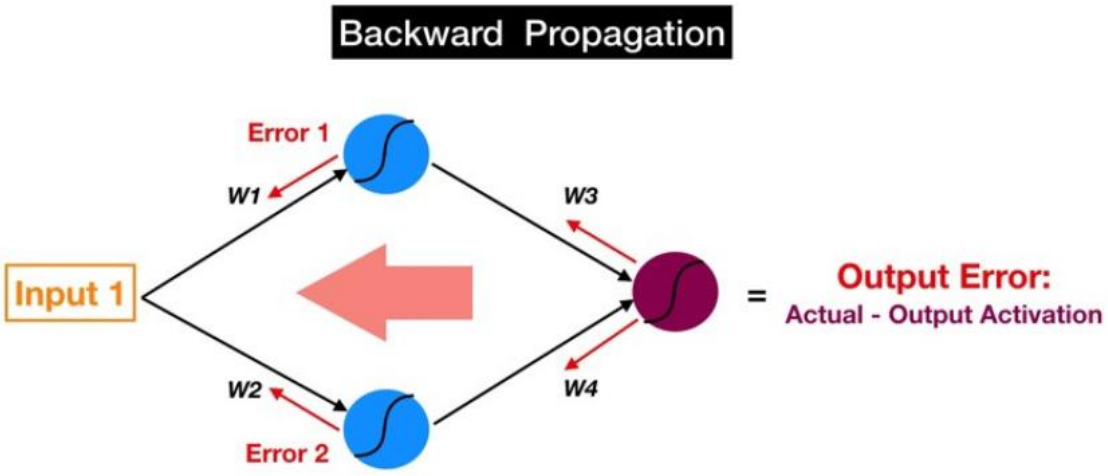
After the loss is calculated, the internal variables (weights and biases) of all the layers of the neural network are adjusted, so as to minimize this loss — that is, to make the output value closer to the correct value.

The above steps are performed during a process called **Backpropagation**. The objective of it is to calculate the error attributable to each neuron starting from the layer closest to the output all the way back to the starting layer of our model.

The objective of forward propagation is to calculate the activations at each neuron for each successive hidden layer until we arrive at the output.



Now let us just reverse it. If you follow the red arrows (in the picture below), you will notice that we are now starting at the output of the magenta neuron. That is our output activation, which we use to make our prediction, and the ultimate source of error in our model. We then move this error backwards through our model via the same weights and connections that we use for forward propagating our signal (so instead of Activation 1, now we have Error1 — the error attributable to the top blue neuron).



The magnitude of the error of a specific neuron (relative to the errors of all the other neurons) is directly proportional to the impact of that neuron's output (a.k.a. activation) on our cost function.

A simple Neural Network

To demonstrate how a neural network 'learns' the best values for the internal variables i.e weights and biases, I decided to build a single layer neural network to predict the temperature in Fahrenheit given its temperature in degree Celsius.

We already know that the formula to convert Celsius to Fahrenheit is:

F = C * 1.8 + 32

The equation for a neuron would be:

$A1 = X*w + b$

Where X is the input, w is the weight and b is the bias.

If the neural network learns well, the internal variable weight should be close to the value 1.8 and internal variable b should be close to the value 32.

I've implemented this neural network using TensorFlow platform.