Artificial Neural Network

Our human brain consists of millions of neurons interconnected with each other, activating and deactivating with electric pulses for different functions of the brain.ANN's are designed to replicate biological neural networks in our brain and how they work.
ANN's consist of three layers:

- Input : receives the input as data from the dataset.
- Hidden : This layer does all calculations and features extraction. Generally, there will be more than one hidden layer.
- Output : gives the output as a prediction from the model.

<u>Computation performed in hidden layers is done in two steps which are as follows</u> :

- First of all, all the inputs are multiplied by their weights. Weight is the coefficient of each variable. It shows the strength of the particular input. After assigning the weights, a bias variable is added. Bias is a constant that helps the model to fit in the best way possible.
- Then in the second step, the activation function is applied to the linear equation. The activation function is a nonlinear transformation that is applied to the input before sending it to the next layer.

Activation functions are attached to each neuron and are mathematical equations that determine whether a neuron should be activated or not based on whether the neuron's input is relevant for the model's prediction or not. The purpose of the activation function is to introduce the nonlinearity in the data.Various Types of Activation Functions are : Sigmoid,TanH/Hyperbolic Tangent, Rectified Linear Unit (ReLU), Leaky ReLU, Softmax

The result from the hidden layer is passed to the output layer to predict the output. After that the error is calculated using different error functions like MSE, Binary Cross Entropy, Cross Entropy Loss

These calculations are part of the process called Forward Propagation

Back Propagation
Back Propagation is the process of updating and finding the optimal values of weights or coefficients which helps the model to minimize the error.The weights

are updated with the help of optimizers. Optimizers are the methods/ mathematical formulations to change the attributes of neural networks.There are different optimizers like gradient descent, RMSprop, Adam which basically calculates the derivative of the loss function to find the minimum.

<u>General process of finding minimum:</u>

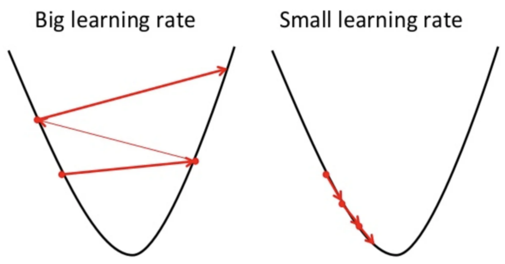
1. The gradient is calculated i.e derivative of error w.r.t current weights

2. Then new weights are calculated using the below formula, where a is the learning rate which is the parameter also known as step size to control the speed or steps of the backpropagation. It gives additional control on how fast we want to move on the curve to reach minima.

$$\underset{\uparrow \atop \text{New weight}}{{}^{*}W_x} = \overset{\text{Old weight} \atop \downarrow}{W_x} - \underset{\uparrow \atop \text{Learning rate}}{a}\left(\overset{\text{Derivative of Error with respect to weight} \atop \downarrow}{\frac{\partial Error}{\partial W_x}}\right)$$

3. This process of calculating the new weights, then errors from the new weights, and then updating the weights continues till we reach global minima and loss is minimized.

A point to note here is that the learning rate in our weight updation equation should be chosen wisely. Learning rate is the amount of change or step size taken towards reaching global minima. It should not be very small as it will take time to converge as well as it should not be very large that it doesn't reach global minimums at all. Therefore, the learning rate is the hyperparameter that we have to choose based on the model.



Some points to remember:

- We can tune different hyperparameters like learning rate, choice of optimizer and activation to better optimize our model.
- Also instead of a training model in the whole training set we can divide that into a mini batch while making our training process much more faster and accurate.
- Neural networks work better with huge amounts of datas so our training examples should be very large. We can consider splitting validation and test set to very low like 0.1% in total in order to give our model much more datas to train on.