Introduction

Artificial Neural Networks are computing systems that are inspired by the working of the Human Neuron. It is the backbone of **Deep Learning** that led to the achievement of bigger milestones in almost all the fields thereby bringing an evolution in which we approach a problem.

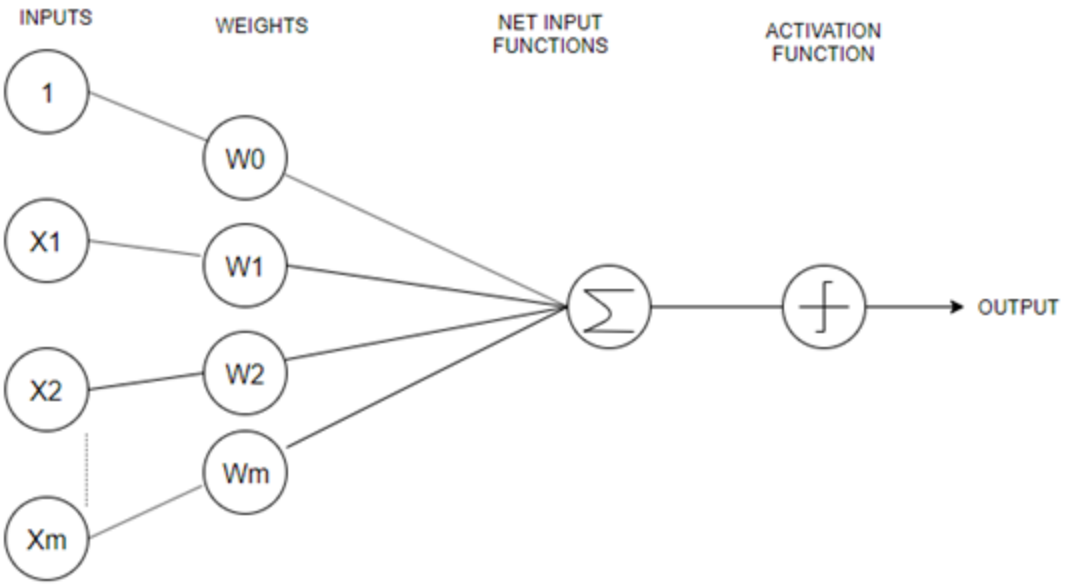
Therefore it becomes necessary for every aspiring **Data Scientist** and **Machine Learning Engineer** to have a good knowledge of these Neural Networks.

In this article, we will discuss the most important questions on the **Artificial Neural Networks (ANNs)** which is helpful to get you a clear understanding of the techniques, and also for **Data Science Interviews,** which covers its very fundamental level to complex concepts.

1. What do you mean by Perceptron?

A perceptron also called an **artificial neuron** is a neural network unit that does certain computations to detect features.

It is a single-layer neural network used as a linear classifier while working with a set of input data. Since perceptron uses classified data points which are already labeled, it is a **supervised learning algorithm**. This algorithm is used to enable neurons to learn and process elements in the training set one at a time.



2. What are the different types of Perceptrons?

There are two types of perceptrons:

1. Single-Layer Perceptrons

Single-layer perceptrons can learn only linearly separable patterns.

2. Multilayer Perceptrons

Multilayer perceptrons, also known as **feedforward neural networks** having two or more layers have a higher processing power.

3. What is the use of the Loss functions?

The loss function is used as a measure of accuracy to identify whether our neural network has learned the patterns accurately or not with the help of the training data.

This is completed by comparing the training data with the testing data.

Therefore, the loss function is considered as a primary measure for the performance of the neural network. In Deep Learning, a good-performing neural network will have a low value of the loss function at all times when training happens.

4. What is the role of the Activation functions in Neural Networks?

The reason for using activation functions in Neural Networks are as follows:

1. The idea behind the activation function is to introduce nonlinearity into the neural network so that it can learn more complex functions.

2. Without the Activation function, the neural network behaves as a linear classifier, learning the function which is a linear combination of its input data.

3. The activation function converts the inputs into outputs.

4. The activation function is responsible for deciding whether a neuron should be activated i.e, fired or not.

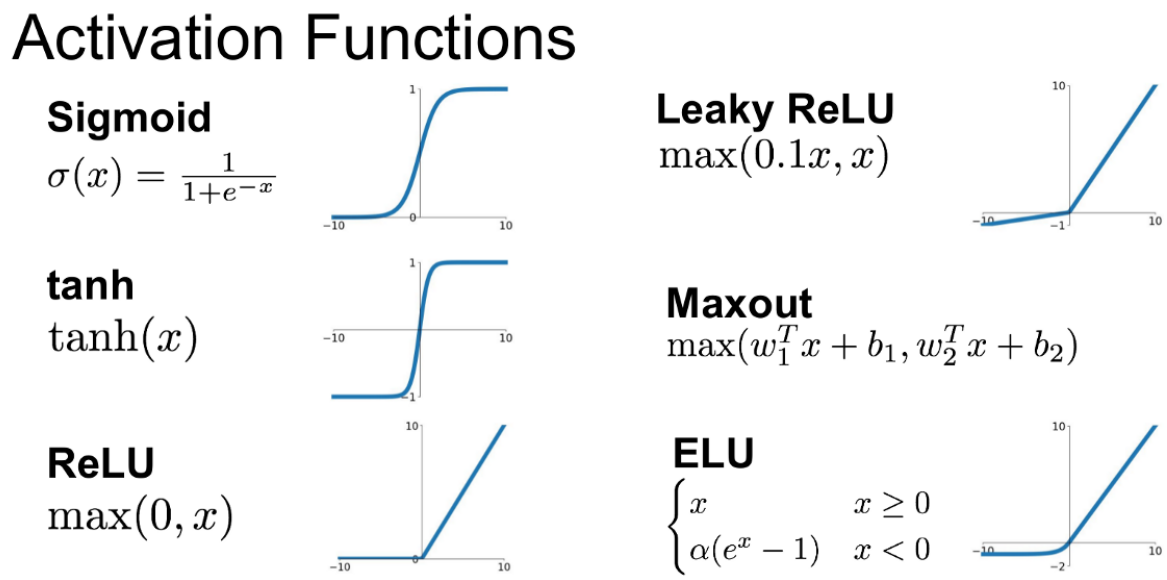
5. To make the decision, firstly it calculates the weighted sum and further adds bias with it.

6. So, the basic purpose of the activation function is to introduce non-linearity into the output of a neuron.

5. List down the names of some popular Activation Functions used in Neural Networks.

Some of the popular activation functions that are used while building the deep learning models are as follows:

- Sigmoid function
- Hyperbolic tangent function
- Rectified linear unit (RELU) function
- Leaky RELU function
- Maxout function
- Exponential Linear unit (ELU) function



6. What do you mean by Cost Function?

While building deep learning models, our whole objective is to minimize the cost function.

A cost function explains how well the neural network is performing for its given training data and the expected output.

It may depend on the neural network parameters such as weights and biases. As a whole, it provides the performance of a neural network.

7. What do you mean by Backpropagation?

The backpropagation algorithm is used to train multilayer perceptrons. It propagates the error information from the end of the network to all the weights inside the network. It allows the efficient computation of the gradient or derivatives.

Backpropagation can be divided into the following steps:

- It can forward the propagation of training data through the network to generate output.
- It uses target value and output value to compute error derivatives by concerning the output activations.
- It can backpropagate to calculate the derivatives of the error concerning output activations in the previous layer and continue for all the hidden layers.
- It uses the previously computed derivatives for output and all hidden layers to calculate the error derivative concerning weights.
- It updates the weights and repeats until the cost function is minimized.

8. How to initialize Weights and Biases in Neural Networks?

Neural network initialization means initialized the values of the parameters i.e, weights and biases. Biases can be initialized to zero but we can’t initialize weights with zero.

Weight initialization is one of the crucial factors in neural networks since bad weight initialization can prevent a neural network from learning the patterns.

On the contrary, a good weight initialization helps in giving a quicker convergence to the global minimum. As a rule of thumb, the rule for initializing the weights is to be close to zero without being too small.

9. Why is zero initialization of weight, not a good initialization technique?

If we initialize the set of weights in the neural network as zero, then all the neurons at each layer will start producing the same output and the same gradients during backpropagation.

As a result, the neural network cannot learn anything at all because there is no source of asymmetry between different neurons. Therefore, we add randomness while initializing the weight in neural networks.

10. Explain Gradient Descent and its types.

Gradient Descent is an optimization algorithm that aims to minimize the cost function or to minimize an error. Its main objective is to find the local or global minima of a function based on its convexity. This determines in which direction the model should go to reduce the error.

There are three types of gradient descent:

- Mini-Batch Gradient Descent
- Stochastic Gradient Descent
- Batch Gradient Descent

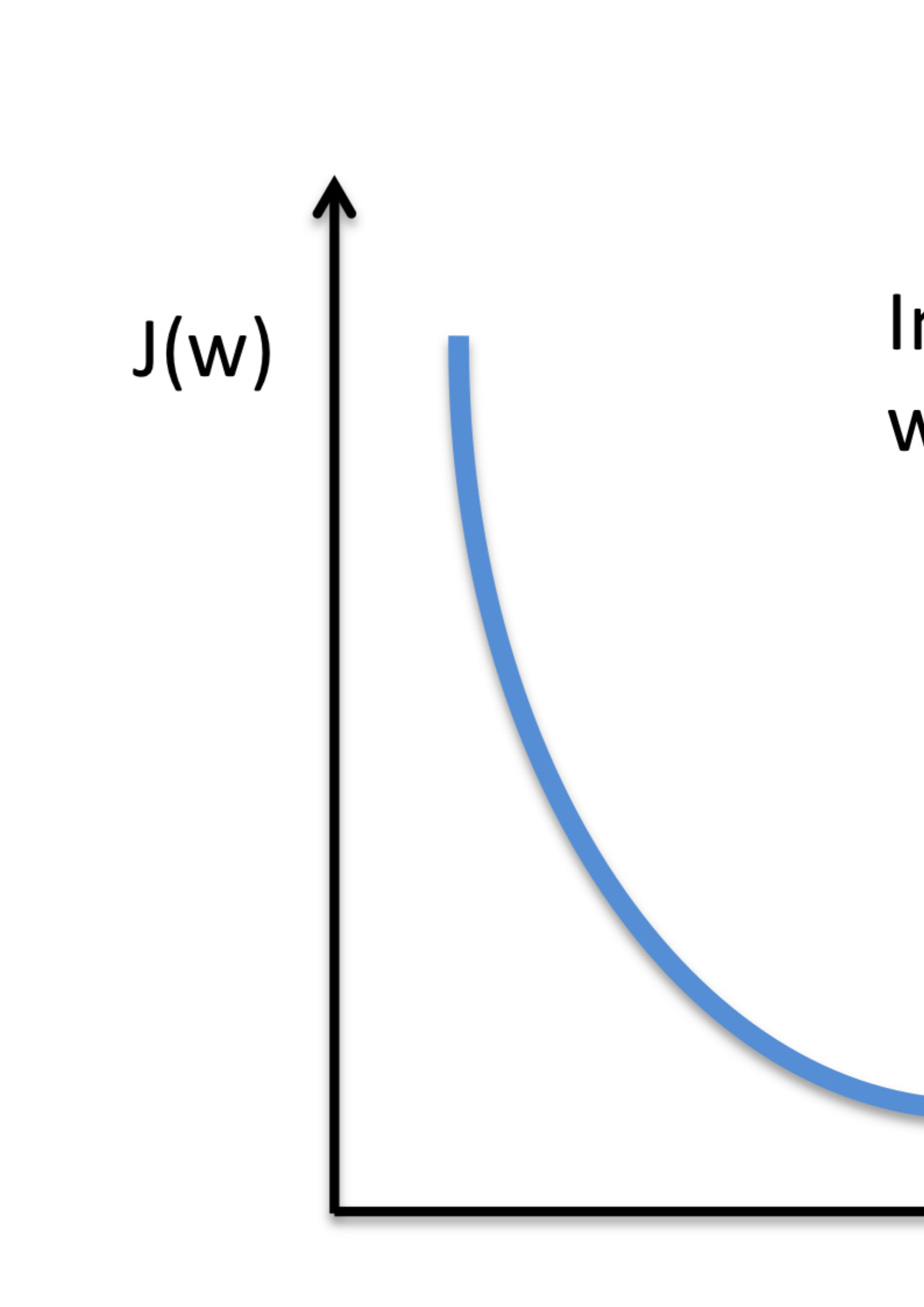

J(w)
In
w

Image Source: Google Images

11. Explain the different steps used in Gradient Descent Algorithm.

The five main steps that are used to initialize and use the gradient descent algorithm are as follows:

- Initialize biases and weights for the neural network.
- Pass the input data through the network i.e, the input layer.
- Compute the difference or the error between the expected and the predicted values.
- Adjust the values i.e, weight updation in neurons to minimize the loss function.
- We repeat the same steps i.e, multiple iterations to determine the best weights for efficient working.

12. Explain the term “Data Normalization”.

Data normalization is an essential preprocessing step, which is used to rescale the initial values to a specific range. It ensures better convergence during backpropagation.

In general, data normalization boils down each of the data points to subtracting the mean and dividing by its standard deviation. This technique improves the performance and stability of neural networks since we normalized the inputs in every layer.

13. What is the difference between Forward propagation and Backward Propagation in Neural Networks?

Forward propagation: The input is fed into the network. In each layer, there is a specific activation function and between layers, there are weights that represent the connection strength of the neurons. The input runs through the individual layers of the network, which ultimately generates an output.

Backward propagation: an error function measures how accurate the output of the network is. To improve the output, the weights have to be optimized. The backpropagation algorithm is used to determine how the individual weights have to be adjusted. The weights are adjusted during the gradient descent method.

14. Explain the different types of Gradient Descent in detail.

Stochastic Gradient Descent: In Stochastic gradient descent, a batch size of 1 is used. As a result, we get n batches. Therefore, the weights of the neural networks are updated after each training sample.

Mini-batch Gradient Descent: In Mini-batch Gradient Descent, the batch size must be between 1 and the size of the training dataset. As a result, we get k batches. Therefore, the weights of the neural networks are updated after each mini-batch iteration.

Batch Gradient Descent: In Batch Gradient Descent, the batch size is equal to the size of the training dataset. Therefore, the weights of the neural network are updated after each epoch.

15. What do you mean by Boltzmann Machine?

One of the most basic Deep Learning models is a Boltzmann Machine, which resembles a simplified version of the Multi-Layer Perceptron.

This model features a visible input layer and a hidden layer — just a two-layer neural network that makes stochastic decisions as to whether a neuron should be activated or not.

In the Boltzmann Machine, nodes are connected across the layers, but no two nodes of the same layer are connected.

16. How does the learning rate affect the training of the Neural Network?

While selecting the learning rate to train the neural network, we have to choose the value very carefully due to the following reasons:

If the learning rate is set too low, training of the model will continue very slowly as we are making very small changes to the weights since our step size that is governed by the equation of gradient descent is small. It will take many iterations before reaching the point of minimum loss.

If the learning rate is set too high, this causes undesirable divergent behavior to the loss function due to large changes in weights due to a larger value of step size. It may fail to converge (the model can give a good output) or even diverge (data is too chaotic for the network to train).

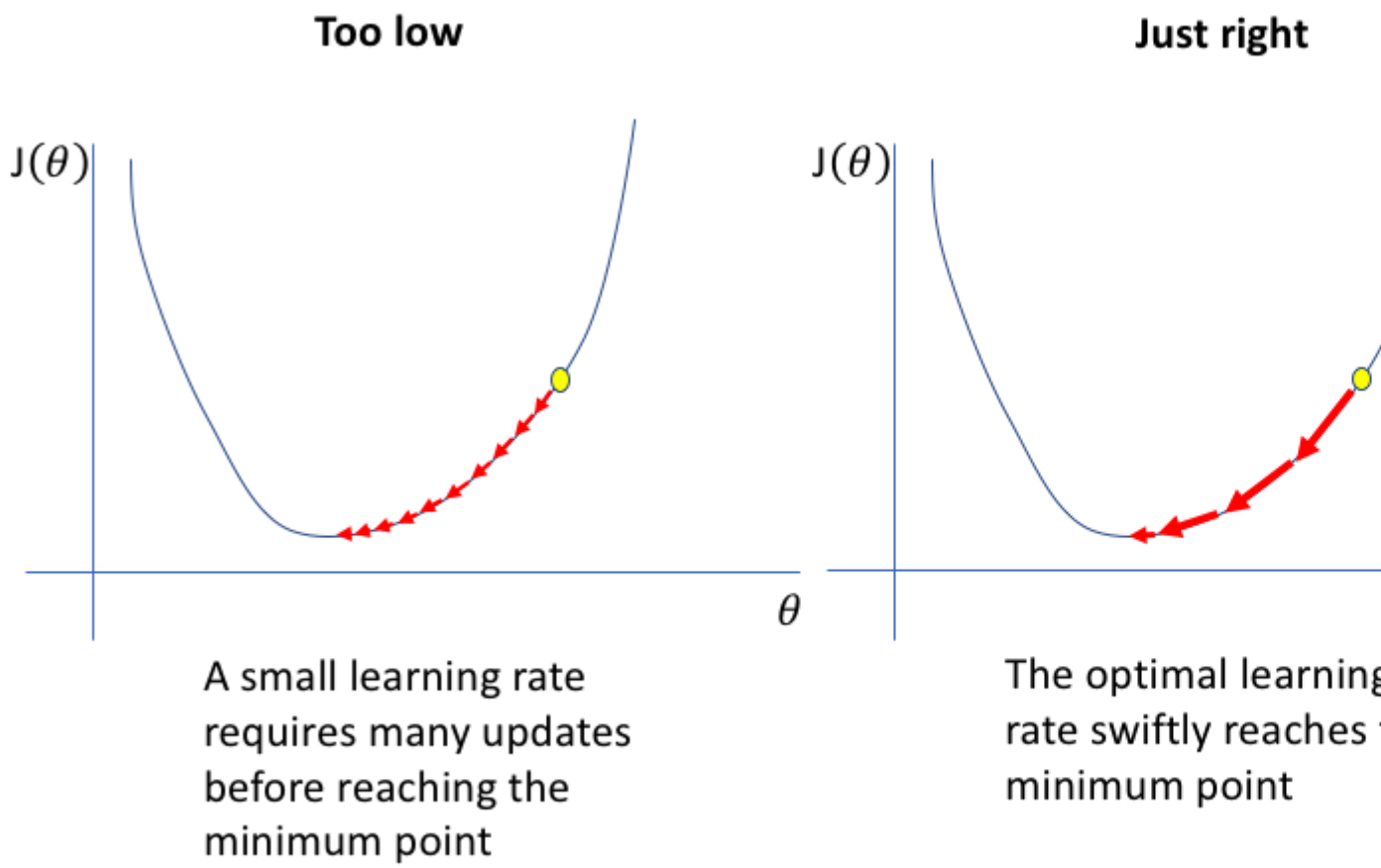


Image Source: Google Images

17. What do you mean by Hyperparameters?

Once the data is formatted correctly, we are usually working with hyperparameters in neural networks. A hyperparameter is a kind of parameter whose values are fixed before the learning process begins.

It decides how a neural network is trained and also the structure of the network which includes:

- The number of hidden units
- The learning rate
- The number of epochs, etc.

18. Why is ReLU the most commonly used Activation Function?

ReLU (Rectified Linear Unit) is the most commonly used activation function in neural networks due to the following reasons:

1. No vanishing gradient: The derivative of the RELU activation function is either 0 or 1, so it could be not in the range of [0,1]. As a result, the product of several derivatives would also be either 0 or 1, because of this property, the vanishing gradient problem doesn't occur during backpropagation.

2. Faster training: Networks with RELU tend to show better convergence performance. Therefore, we have a much lower run time.

3. Sparsity: For all negative inputs, a RELU generates an output of 0. This means that fewer neurons of the network are firing. So we have sparse and efficient activations in the neural network.

19. Explain the vanishing and exploding gradient problems.

These are the major problems in training deep neural networks.

While Backpropagation, in a network of n hidden layers, n derivatives will be multiplied together. If the derivatives are large **e.g, If use ReLU like activation function** then the value of the gradient will increase exponentially as we propagate down the model until they eventually explode, and this is what we call the problem of Exploding gradient.

On the contrary, if the derivatives are small **e.g, If use a Sigmoid activation function** then the gradient will decrease exponentially as we propagate through the model until it eventually vanishes, and this is the Vanishing gradient problem.