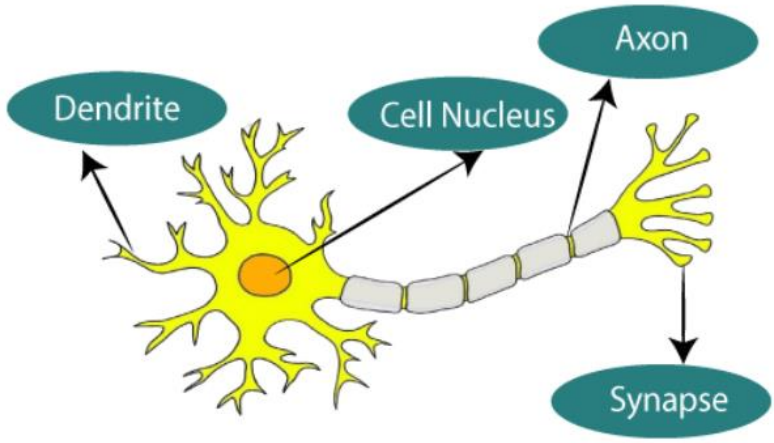
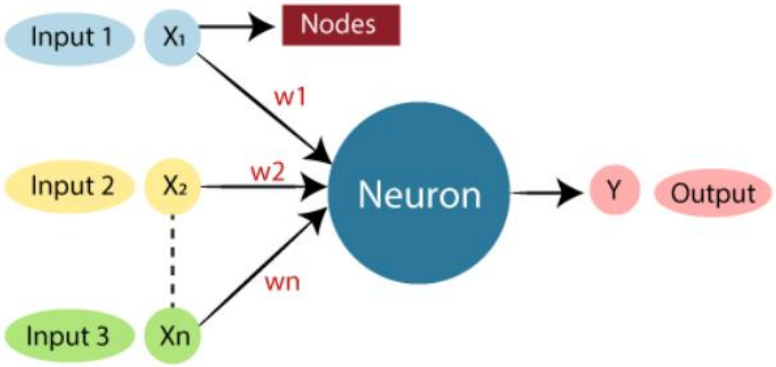
The term "Artificial neural network" refers to a biologically inspired sub-field of artificial intelligence modeled after the brain. An Artificial neural network is usually a computational network based on biological neural networks that construct the structure of the human brain. Similar to a human brain has neurons interconnected to each other, artificial neural networks also have neurons that are linked to each other in various layers of the networks. These neurons are known as nodes.

Artificial neural network tutorial covers all the aspects related to the artificial neural network. I



The given figure illustrates the typical diagram of Biological Neural Network.

The typical Artificial Neural Network looks something like the given figure.



Advantages of Artificial Neural Network (ANN)

Parallel processing capability:

Artificial neural networks have a numerical value that can perform more than one task simultaneously.

Storing data on the entire network:

Data that is used in traditional programming is stored on the whole network, not on a database. The disappearance of a couple of pieces of data in one place doesn't prevent the network from working.

Capability to work with incomplete knowledge:

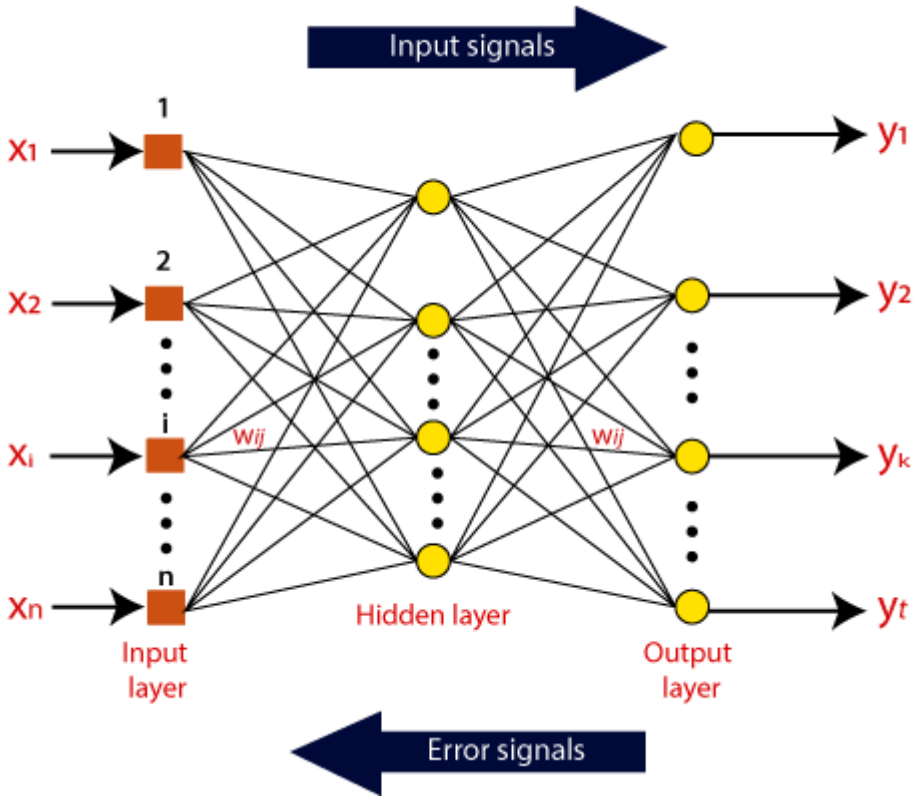
After ANN training, the information may produce output even with inadequate data. The loss of performance here relies upon the significance of missing data.

Having a memory distribution:

For ANN is to be able to adapt, it is important to determine the examples and to encourage the network according to the desired output by demonstrating these examples to the network. The succession of the network is directly proportional to the chosen instances, and if the event can't appear to the network in all its aspects, it can produce false output.

How do artificial neural networks work?

Artificial Neural Network can be best represented as a weighted directed graph, where the artificial neurons form the nodes. The association between the neurons outputs and neuron inputs can be viewed as the directed edges with weights. The Artificial Neural Network receives the input signal from the external source in the form of a pattern and image in the form of a vector. These inputs are then mathematically assigned by the notations x(n) for every n number of inputs.



Afterward, each of the input is multiplied by its corresponding weights (these weights are the details utilized by the artificial neural networks to solve a specific problem). In general terms, these weights normally represent the strength of the interconnection between neurons inside the artificial neural network. All the weighted inputs are summarized inside the computing unit.

If the weighted sum is equal to zero, then bias is added to make the output non-zero or something else to scale up to the system's response. Bias has the same input, and weight equals to 1. Here the total of weighted inputs can be in the range of 0 to positive infinity. Here, to keep the response in the limits of the desired value, a certain maximum value is benchmarked, and the total of weighted inputs is passed through the activation function.

The activation function refers to the set of transfer functions used to achieve the desired output. There is a different kind of the activation function, but primarily either linear or non-linear sets of functions. Some of the commonly used sets of activation functions are the Binary, linear, and Tan hyperbolic sigmoidal activation functions.

Multilayer Feed-Forward Neural Network in Data Mining

- Last Updated : 08 Sep, 2022

- Read

- Discuss

Multilayer Feed-Forward Neural Network(MFFNN) is an interconnected Artificial Neural Network with multiple layers that has neurons with weights associated with them and they compute the result using activation functions. It is one of the types of Neural Networks in which the flow of the network is from input to output units and it does have any loops, no feedback, and no signal moves in backward directions that is from output to hidden and input layer.

The ANN is a self-learning network that learns from sample data sets and signals, it is based on the function of the biological nervous system. The type of activation function depends on the desired output. It is a part of machine learning and AI, which are the fastest growing fields, and lots of research is going on to make it more effective.

The Architecture of the Multilayer Feed-Forward Neural Network:

This Neural Network or Artificial Neural Network has multiple hidden layers that make it a multilayer neural Network and it is feed-forward because it is a network that follows a top-down approach to train the network. In this network there are the following layers:

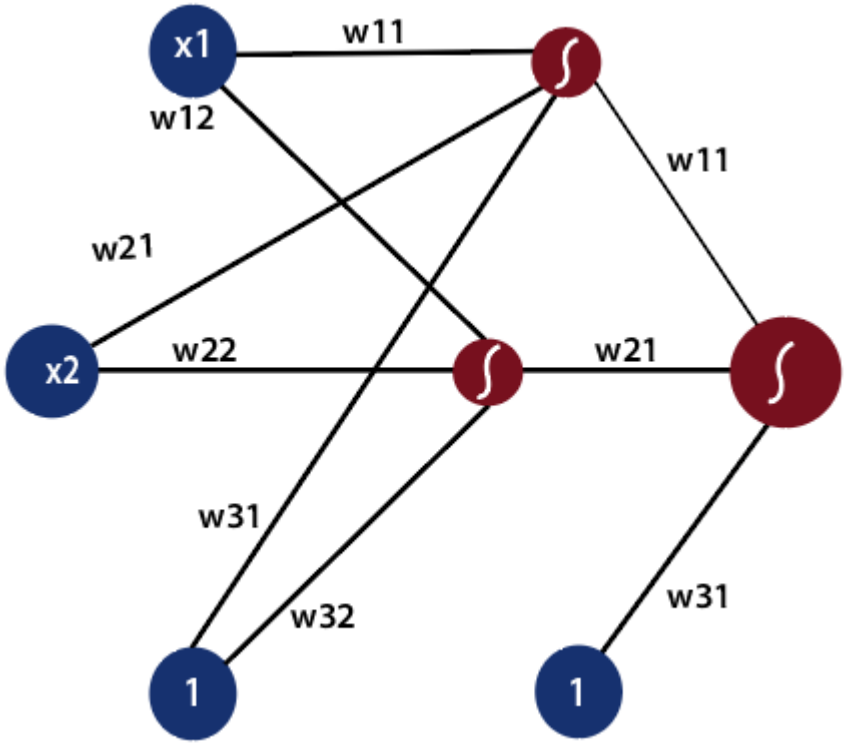
1. **Input Layer:** It is starting layer of the network that has a weight associated with the signals.

2. **Hidden Layer:** This layer lies after the input layer and contains multiple neurons that perform all computations and pass the result to the output unit.
3. **Output Layer:** It is a layer that contains output units or neurons and receives processed data from the hidden layer, if there are further hidden layers connected to it then it passes the weighted unit to the connected hidden layer for further processing to get the desired result.

The input and hidden layers use sigmoid and linear activation functions whereas the output layer uses a Heaviside step activation function at nodes because it is a two-step activation function that helps in predicting results as per requirements. All units also known as neurons have weights and calculation at the hidden layer is the summation of the dot product of all weights and their signals and finally the sigmoid function of the calculated sum. Multiple hidden and output layer increases the accuracy of the output.

"The process of receiving an input to produce some kind of output to make some kind of prediction is known as Feed Forward." Feed Forward neural network is the core of many other important neural networks such as convolution neural network.

In the feed-forward neural network, there are not any feedback loops or connections in the network. Here is simply an input layer, a hidden layer, and an output layer.



There can be multiple hidden layers which depend on what kind of data you are dealing with. The number of hidden layers is known as the depth of the neural network. The deep neural network can learn from more functions. Input layer first provides the neural network with data and the output layer then make predictions on that data which is based on a series of functions. ReLU Function is the most commonly used activation function in the deep neural network.

To gain a solid understanding of the feed-forward process, let's see this mathematically.

1) The first input is fed to the network, which is represented as matrix x1, x2, and one where one is the bias value.

$$[x_1 \quad x_2 \quad 1]$$

2) Each input is multiplied by weight with respect to the first and second model to obtain their probability of being in the positive region in each model.

So, we will multiply our inputs by a matrix of weight using matrix multiplication.

$$[x_1 \quad x_2 \quad 1] = \begin{bmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \end{bmatrix} = [\text{score} \quad \text{score}]$$

3) After that, we will take the sigmoid of our scores and gives us the probability of the point being in the positive region in both models.

$$\frac{1}{1+e^{-x}}[\text{score score}] = \text{probability}$$

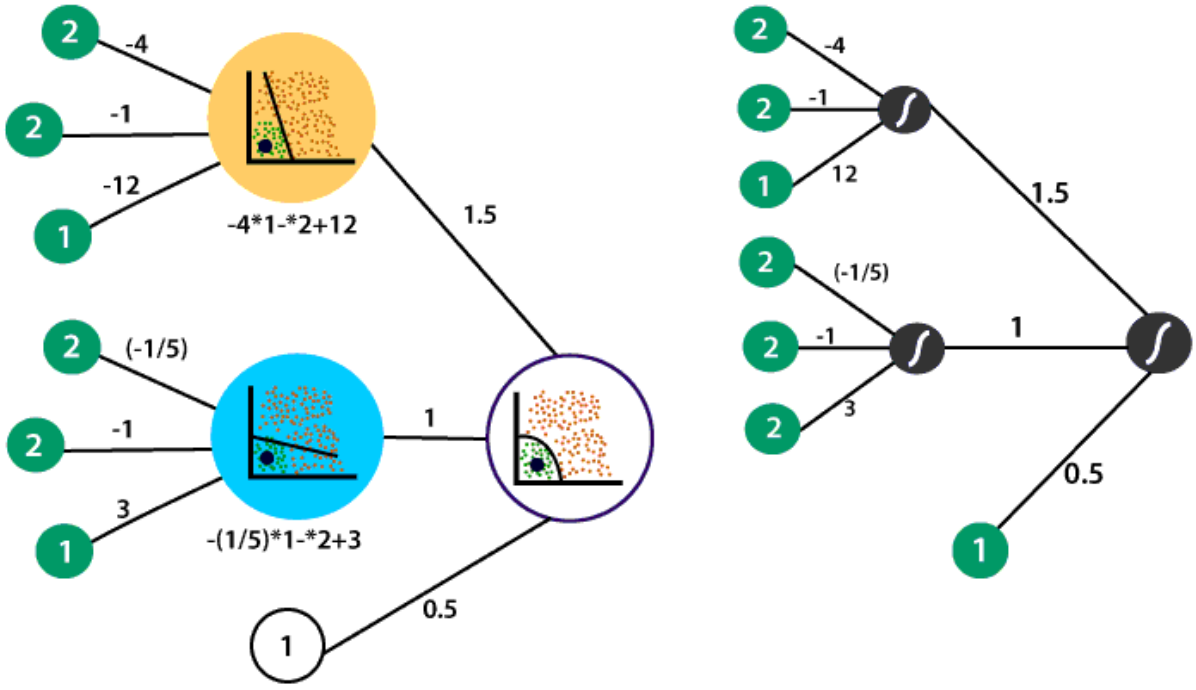
4) We multiply the probability which we have obtained from the previous step with the second set of weights. We always include a bias of one whenever taking a combination of inputs.

$$[\text{probability} \quad \text{probability} \quad 1] \times \begin{bmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \\ w_{31} & w_{32} \end{bmatrix} = [\text{score}]$$

And as we know to obtain the probability of the point being in the positive region of this model, we take the sigmoid and thus producing our final output in a feed-forward process.

$$\frac{1}{1+e^{-x}}[\text{score}] = [\text{probability}]$$

Let takes the neural network which we had previously with the following linear models and the hidden layer which combined to form the non-linear model in the output layer.



So, what we will do we use our non-linear model to produce an output that describes the probability of the point being in the positive region. The point was represented by 2 and 2. Along with bias, we will represent the input as

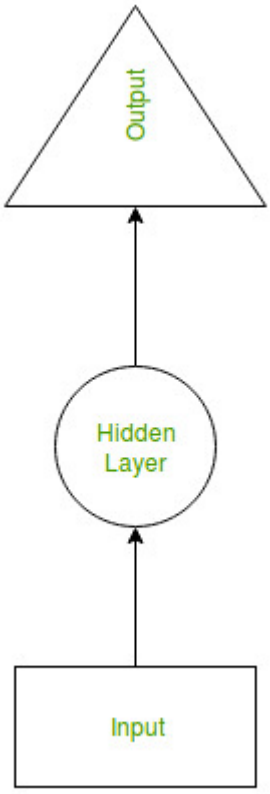
$$[2 \quad 2 \quad 1]$$

The first linear model in the hidden layer recall and the equation defined it

$$-4x_1 - x_2 + 12$$

Which means in the first layer to obtain the linear combination the inputs are multiplied by -4, -1 and the bias value is multiplied by twelve.

Recurrent Neural Network(RNN) are a type of Neural Network where the **output from previous step are fed as input to the current step**. In traditional neural networks, all the inputs and outputs are independent of each other, but in cases like when it is required to predict the next word of a sentence, the previous words are required and hence there is a need to remember the previous words. Thus RNN came into existence, which solved this issue with the help of a Hidden Layer. The main and most important feature of RNN is **Hidden state**, which remembers some information about a sequence.

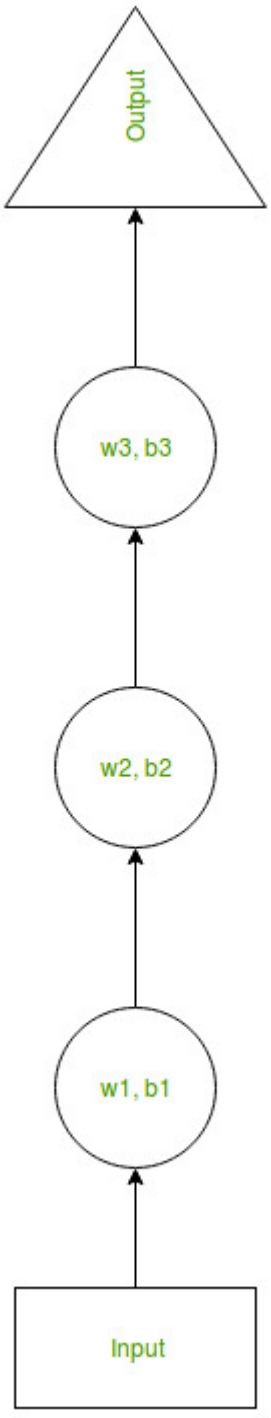


RNN have a **“memory”** which remembers all information about what has been calculated. It uses the same parameters for each input as it performs the same task on all the inputs or hidden layers to produce the output. This reduces the complexity of parameters, unlike other neural networks.

How RNN works

The working of a RNN can be understood with the help of below example:

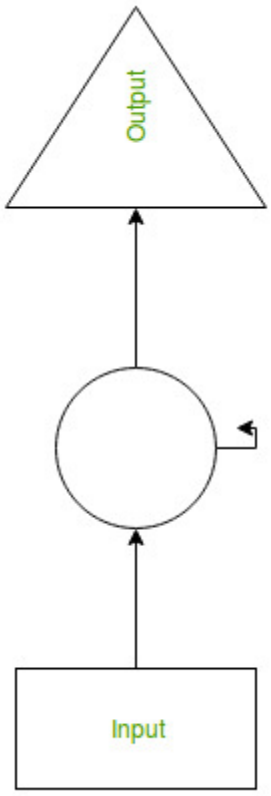
Example:
Suppose there is a deeper network with one input layer, three hidden layers and one output layer. Then like other neural networks, each hidden layer will have its own set of weights and biases, let’s say, for hidden layer 1 the weights and biases are (w1, b1), (w2, b2) for second hidden layer and (w3, b3) for third hidden layer. This means that each of these layers are independent of each other, i.e. they do not memorize the previous outputs.



Now the RNN will do the following:

- RNN converts the independent activations into dependent activations by providing the same weights and biases to all the layers, thus reducing the complexity of increasing parameters and memorizing each previous outputs by giving each output as input to the next hidden layer.

- Hence these three layers can be joined together such that the weights and bias of all the hidden layers is the same, into a single recurrent layer.



Machine Learning Course

Beginner to Advance Level

Machines are evolving, so why do you wish to get left behind? Strengthen your ML and AI foundations today and become future-ready with Machine Learning Basic and Advanced - Self-Paced Course.

Explore Now

- **Formula for calculating current state:**

$$h_t = f(h_{t-1}, x_t)$$

where:

h_t -> current state
h_{t-1} -> previous state
x_t -> input state

- **Formula for applying Activation function(tanh):**

$$h_t = \tanh (W_{hh}h_{t-1} + W_{xh}x_t)$$

where:

W_{hh} -> weight at recurrent neuron
W_{xh} -> weight at input neuron

- **Formula for calculating output:**

$$y_t = W_{hy}h_t$$

Y_t -> output
W_{hy} -> weight at output layer

Training through RNN

1. A single time step of the input is provided to the network.
2. Then calculate its current state using set of current input and the previous state.
3. The current ht becomes ht-1 for the next time step.
4. One can go as many time steps according to the problem and join the information from all the previous states.
5. Once all the time steps are completed the final current state is used to calculate the output.
6. The output is then compared to the actual output i.e the target output and the error is generated.
7. The error is then back-propagated to the network to update the weights and hence the network (RNN) is trained.

Advantages of Recurrent Neural Network

There are two Artificial Neural Network topologies − **FeedForward and Feedback**.

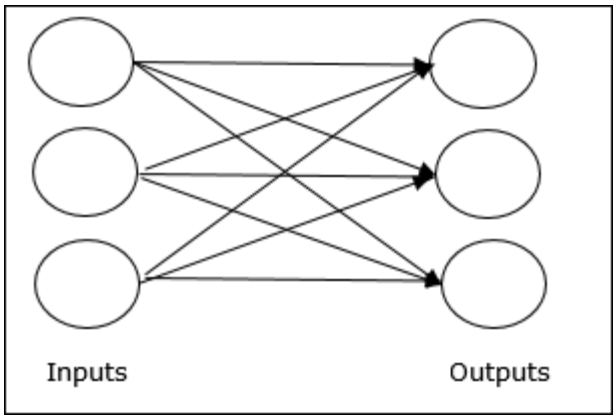
Network Topology

A network topology is the arrangement of a network along with its nodes and connecting lines. According to the topology, ANN can be classified as the following kinds −

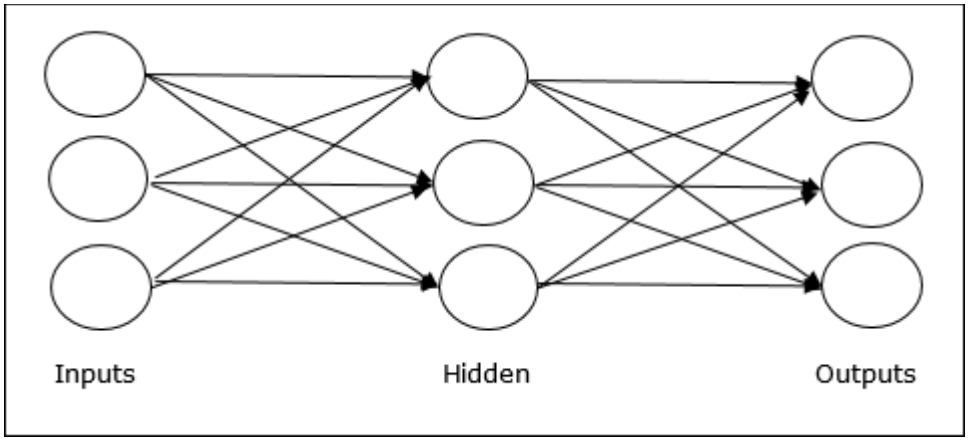
Feedforward Network

It is a non-recurrent network having processing units/nodes in layers and all the nodes in a layer are connected with the nodes of the previous layers. The connection has different weights upon them. There is no feedback loop means the signal can only flow in one direction, from input to output. It may be divided into the following two types −

- **Single layer feedforward network** − The concept is of feedforward ANN having only one weighted layer. In other words, we can say the input layer is fully connected to the output layer.



- **Multilayer feedforward network** − The concept is of feedforward ANN having more than one weighted layer. As this network has one or more layers between the input and the output layer, it is called hidden layers.



Feedback Network

As the name suggests, a feedback network has feedback paths, which means the signal can flow in both directions using loops. This makes it a non-linear dynamic system, which changes continuously until it reaches a state of equilibrium. It may be divided into the following types −

- **Recurrent networks** − They are feedback networks with closed loops. Following are the two types of recurrent networks.
- **Fully recurrent network** − It is the simplest neural network architecture because all nodes are connected to all other nodes and each node works as both input and output.
-

A Perceptron is **a neural network unit that does certain computations to detect features or business intelligence in the input data**. It is a function that maps its input “x,” which is multiplied by the learned weight coefficient, and generates an output value ”f(x).

In Machine Learning, Perceptron is considered as a single-layer neural network that consists of four main parameters named input values (Input nodes), weights and Bias, net sum, and an activation function. The perceptron model begins with the multiplication of all input values and their weights, then adds these values together to create the weighted sum. Then this weighted sum is applied to the activation function 'f' to obtain the desired output. This activation function is also known as the **step function** and is represented by **'f'**.

In linear regression, the model targets to get the best-fit regression line to predict the value of y based on the given input value (x). While training the model, the model calculates the cost function which measures the Root Mean Squared error between the predicted value (pred) and true value (y). The model targets to minimize the cost function.
To minimize the cost function, the model needs to have the best value of θ_1 and θ_2. Initially model selects θ_1 and θ_2 values randomly and then iteratively update these value in order to minimize the cost function until it reaches the minimum. By the time model achieves the minimum cost function, it will have the best θ_1 and θ_2 values. Using these finally updated values of θ_1 and θ_2 in the hypothesis equation of linear equation, the model predicts the value of x in the best manner it can.