Multi Layer Perceptron

24 August 2024 20:15

A multi-layer perceptron (MLP) is a type of artificial neural network (ANN) that consists of multiple layer of nodes (neurons)

- It is also known as feed-forward neural network

Structure of MLP

Input Layer: it takes the input features of the data basically passes the input features to the next layer without any computation

each node/ neuron in input layer represents a feature of the input data.

Hidden Layer: These are intermediate layers where computations are performed.

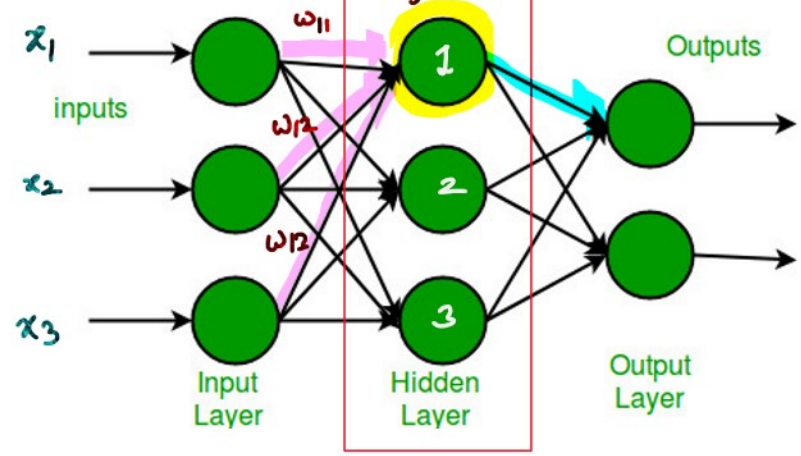
→ weighted sum of the inputs and add a bias and then applies the activation function.

hidden layers are one or more layers that perform intermediate computations. Each hidden layer consists of neurons (nodes) that apply activation functions to weighted sum of inputs along with bias

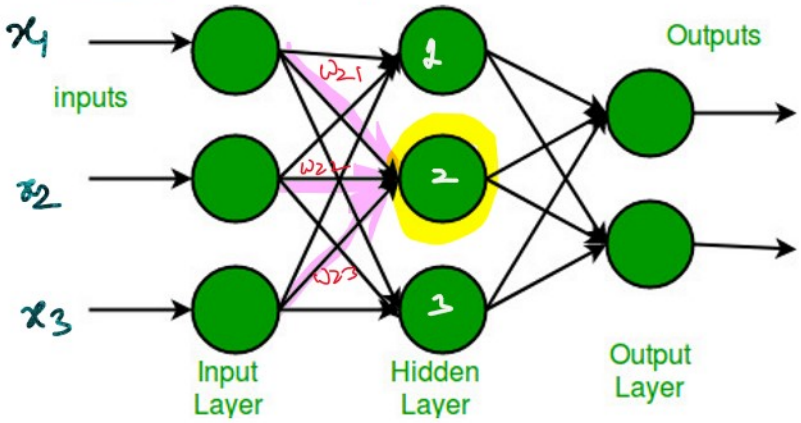
output layer: it produces the final output, with the no. of neurons corresponding to the number of classes in classification tasks

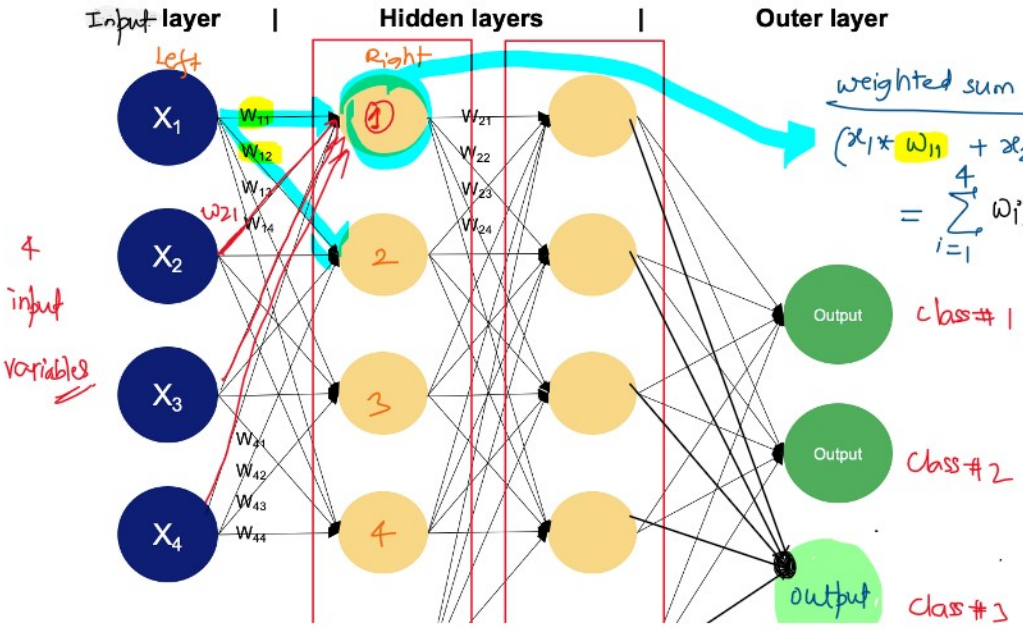
OR one neuron in regression tasks

1st neuron in the hidden layer



2nd neuron in hidden layer



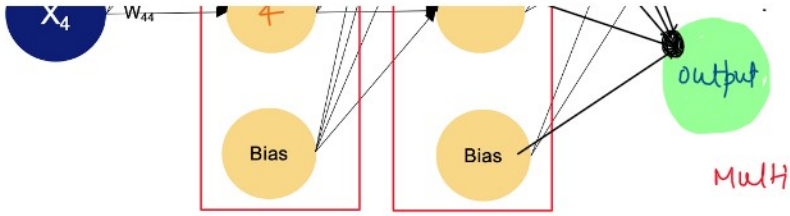


weighted sum of inputs

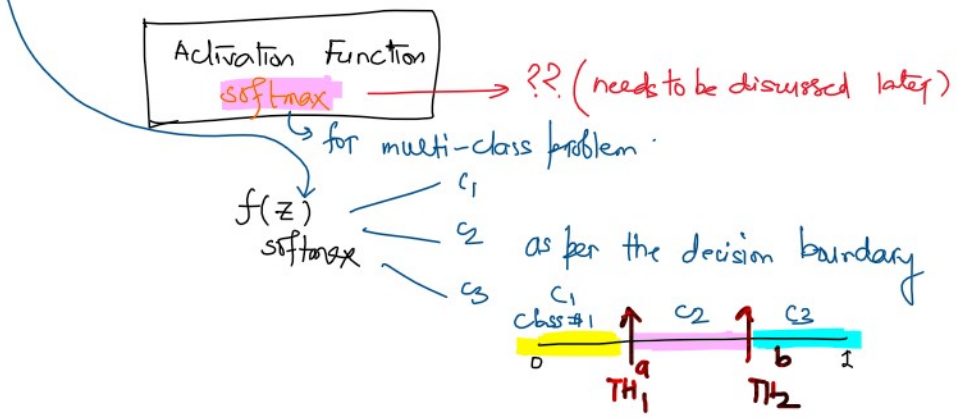
$(x_1 * w_{11} + x_2 * w_{21} + x_3 * w_{31} + x_4 * w_{41})$

$= \sum_{i=1}^{4} w_{i1} * x_i$

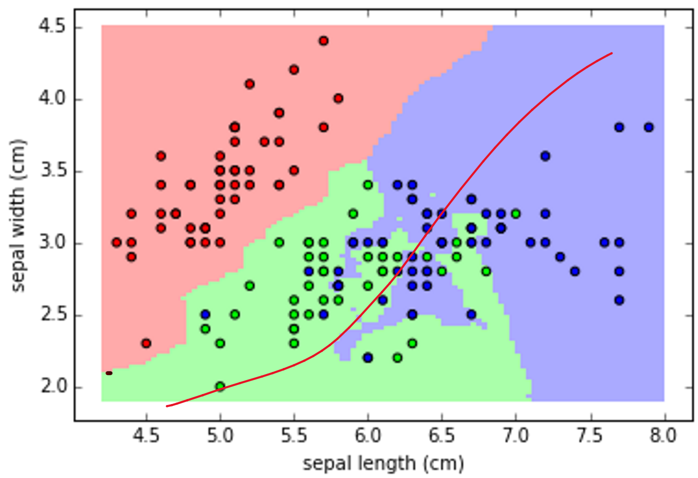
↳ first neuron weighted sum in HL

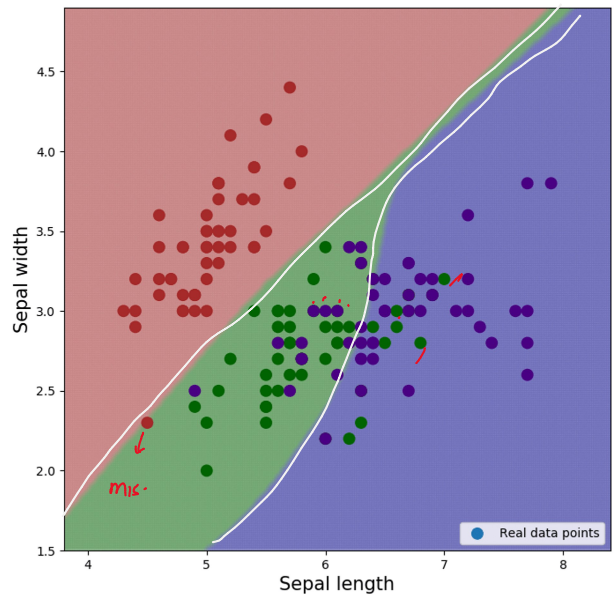


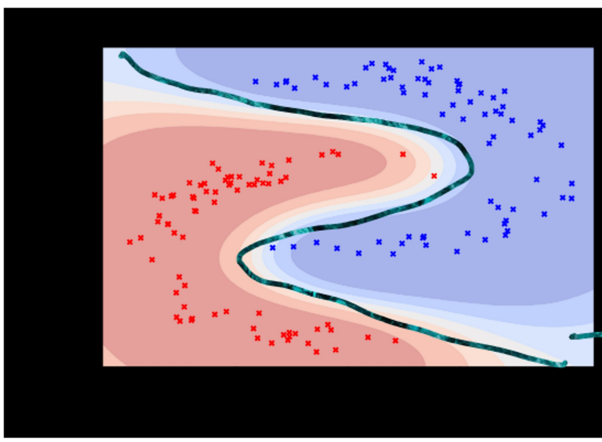
$$z = \sum w_{i1} * x_i + b$$

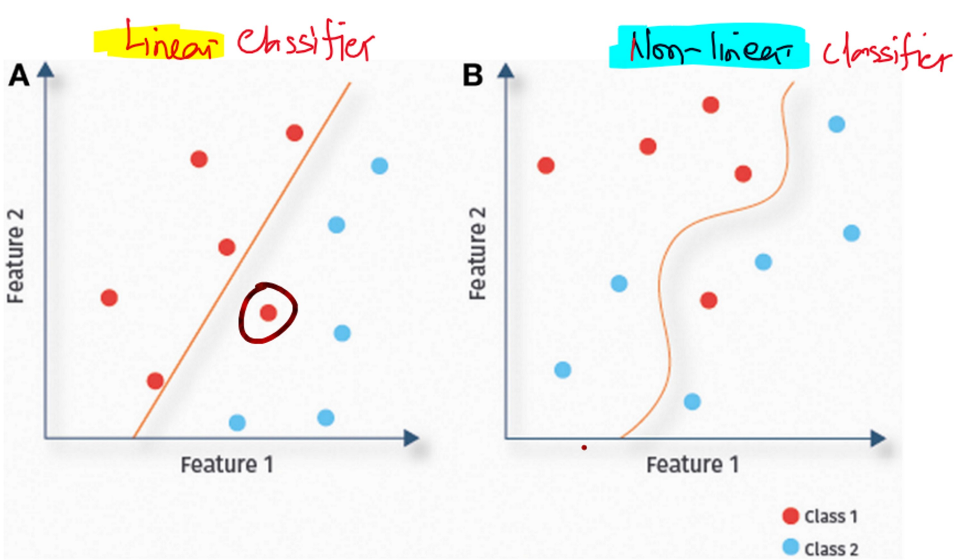


Weighted sum is then passes through an activation function denoted as $f(z)$. Activation function is crucial as it introduces 'non-linearity' into the model, allowing it to learn more complex patterns.









Forward Propagation

forward Propagation

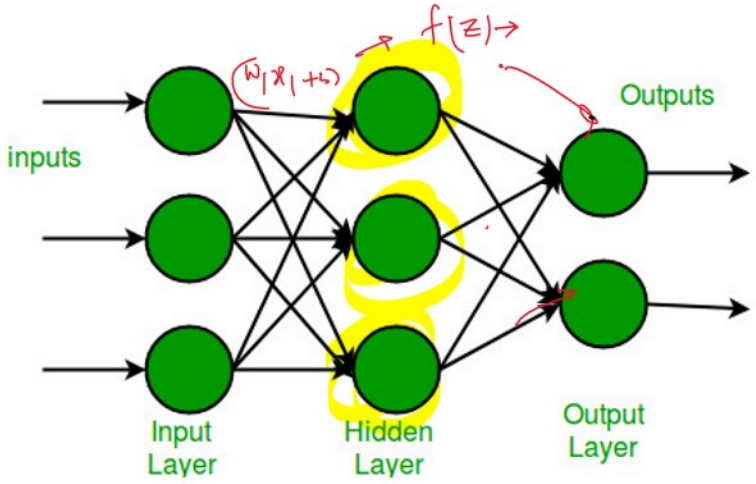
The process from the input layer through the hidden layers is called forward propagation. In each layer, steps such as weighted sum, bias addition, applying activation function are performed to compute the layer's output.

output layer:

Final layer is the output layer.

In a classification task, this layer often uses a sigmoid function for binary classification and softmax function for multiclass classification.

output layer of an MLP produces the final predictions or outputs of the network.



Forward Propagation and Backward Propagation

calculation of predictions ✓ calculation of gradients

~~calculation of prediction~~ calculation of gradients

Forward Propagation (FP)

In forward propagation, input data is passed through the neural network to compute the output (predictions)

Mathematical representation

For a given layer l, the forward pass computes:

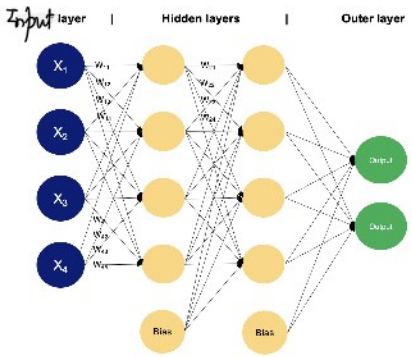
activation function → Z

$$a^{(l)} = \sigma(W^{(l)} \cdot a^{(l-1)} + b^{(l)})$$

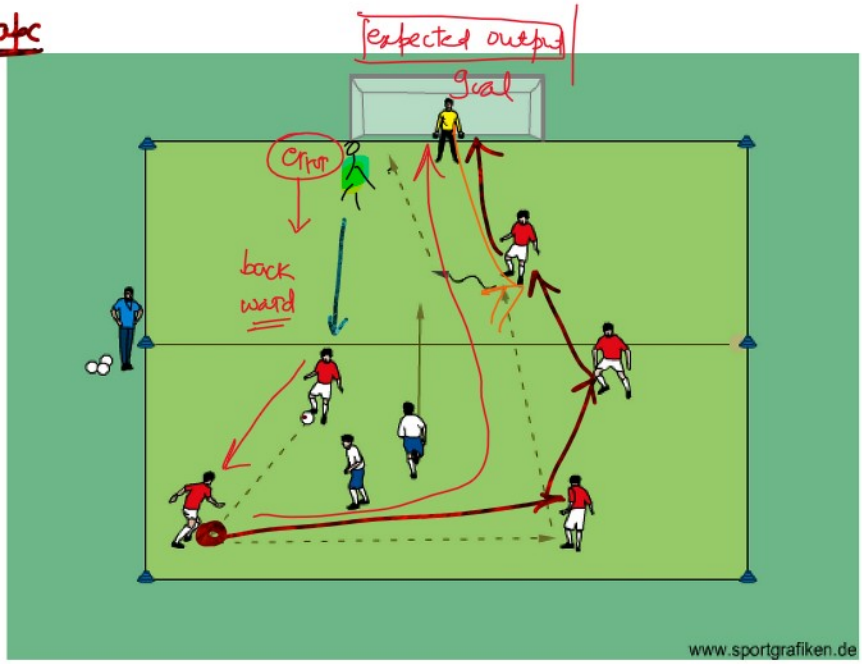
(weights × activations) + bias =

Where:

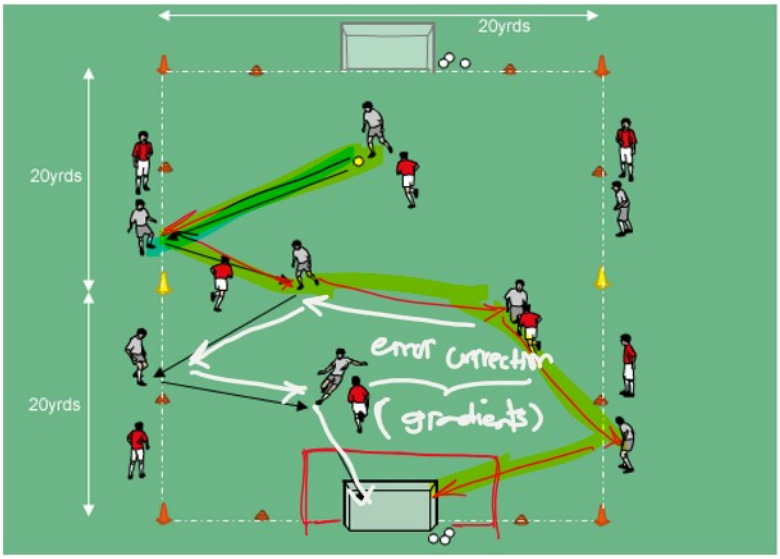
- $a^{(l-1)}$: Activations from the previous layer (or input features for the first hidden layer).
- $W^{(l)}$: Weight matrix of the current layer.
- $b^{(l)}$: Bias vector of the current layer.
- σ: Activation function (e.g., ReLU, sigmoid).
- $a^{(l)}$: Output activations of the current layer.



©ofc

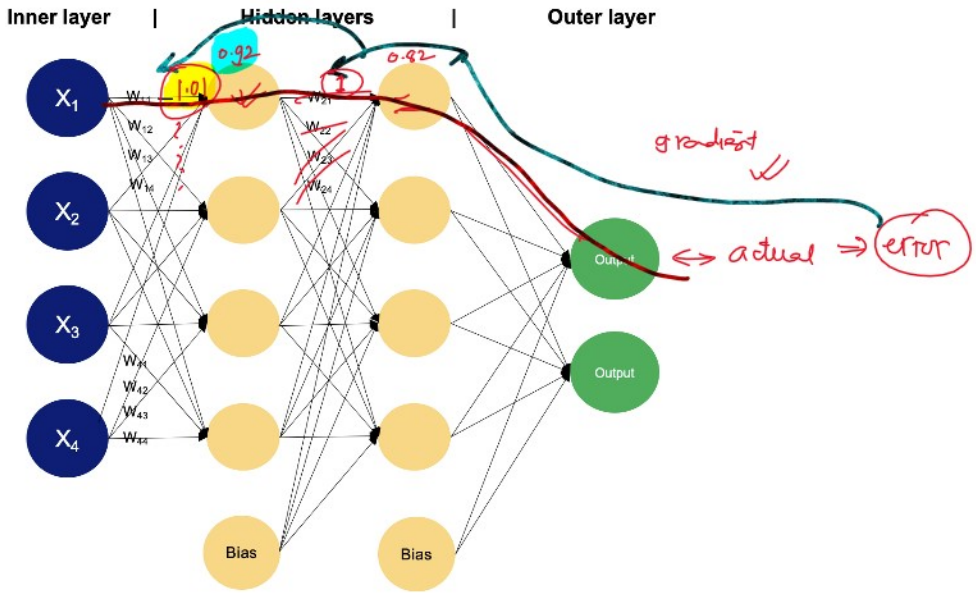


Note: Forward propagation doesn't involve the optimizer. It only involves passing data through the neural network to compute predictions.



2. Backward Propagation: Calculation of gradients

Backward propagation is where the model learns.
It computes the gradient of the loss function w.r.t each weight and bias by applying chain rule (differentation)
These gradients are then used to update the parameters of the network.



Mathematical Representation

Gradient of loss function L w.r.t weights W and biases b is computed:

$$\frac{\partial L}{\partial W^{(l)}} = \frac{\partial L}{\partial a^{(l)}} \cdot \frac{\partial a^{(l)}}{\partial W^{(l)}}$$

$$\rightarrow \frac{\partial L}{\partial b^{(l)}} = \frac{\partial L}{\partial a^{(l)}} \cdot \frac{\partial a^{(l)}}{\partial b^{(l)}}$$