KESHAV MEMORIAL INSTITUTE OF TECHNOLOGY
AN AUTONOMOUS INSTITUTION - ACCREDITED BY NAAC WITH 'A' GRADE
Narayanaguda, Hyderabad.

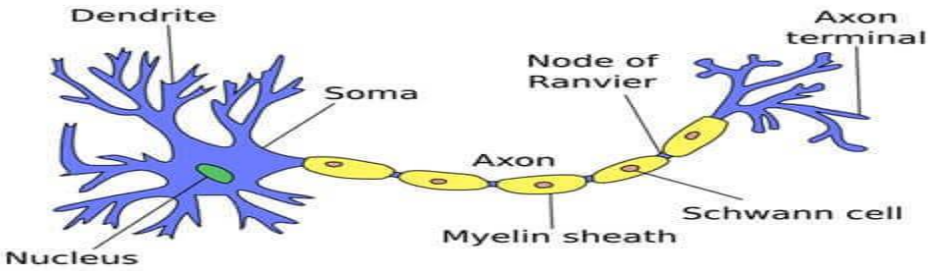
Deep Learning

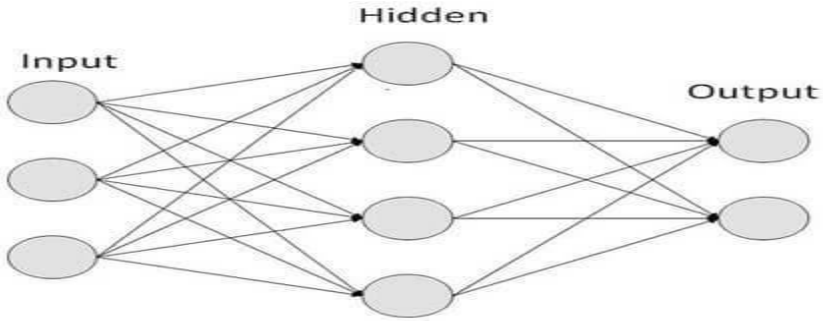
SESSION-3

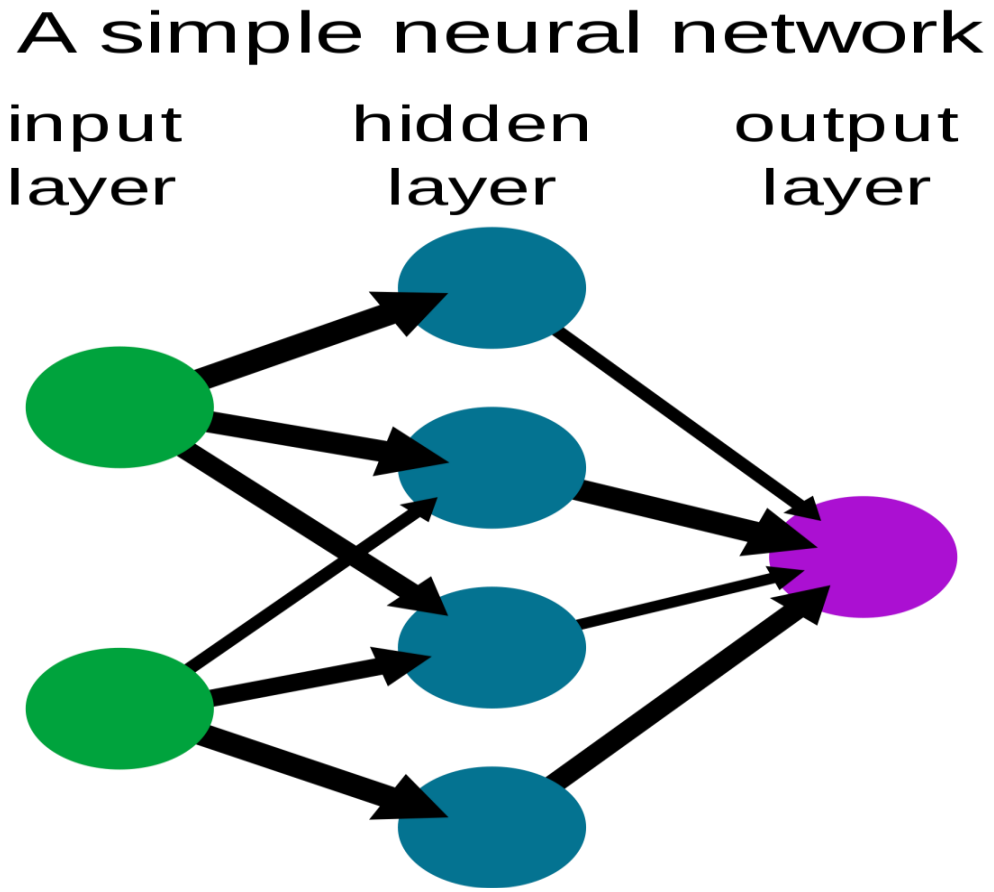
BY
ASHA

Introduction to Neural Networks

- Neural networks are computational models inspired by the human brain. They consist of interconnected units or nodes called neurons, organized into layers.
- Neural networks are capable of learning from data and making predictions or decisions without being explicitly programmed.

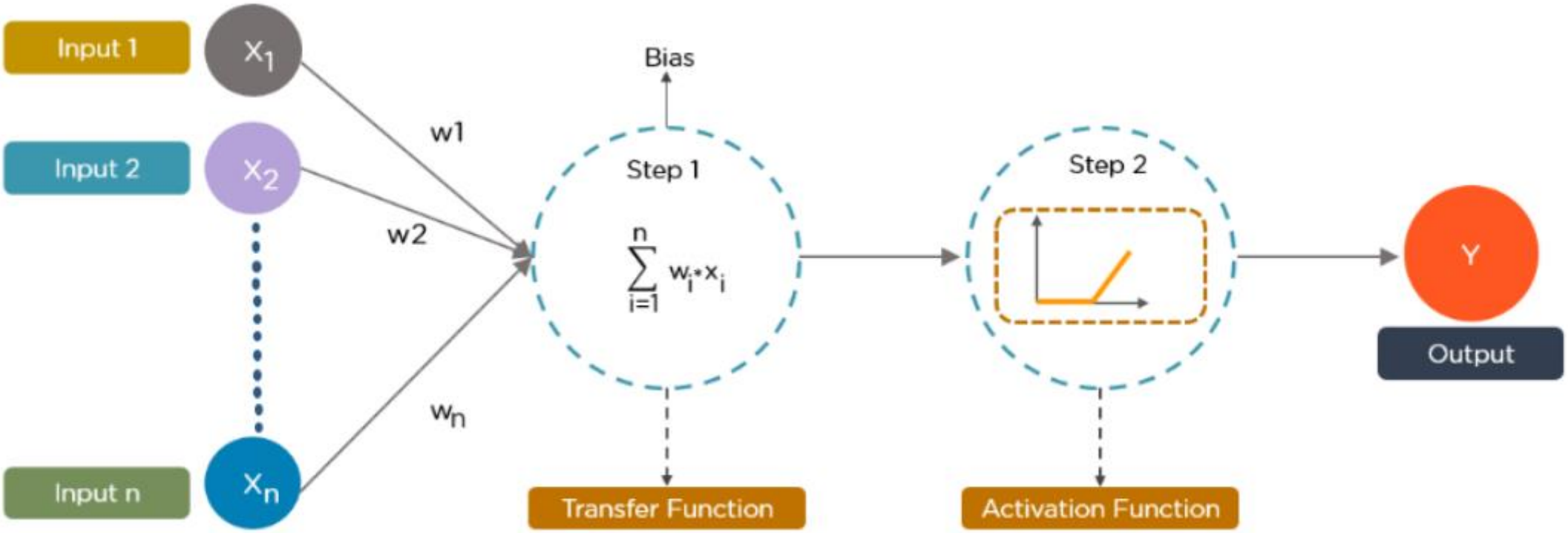






A simple neural network consists of three components :

- Input layer
- Hidden layer
- Output layer


Input 1
Input 2
Input n
X_1
X_2
X_n
w1
w2
w_n
Bias
Step 1
$\sum_{i=1}^{n} w_i * x_i$
Step 2
Y
Output
Transfer Function
Activation Function

1. In the first step, **Input units are passed i.e data is passed with some weights attached to it to the hidden layer**. We can have any number of hidden layers. In the above image inputs x_1,x_2,x_3,….x_n is passed.
2. Each hidden layer consists of neurons. All the inputs are connected to each neuron.
3. After passing on the inputs, **all the computation is performed in the hidden layer**.

- Computation performed in hidden layers are done in two steps which are as follows :
- First of all, **all the inputs are multiplied by their weights**. Weight is the gradient or 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.

$$Z_1 = W_1*In_1 + W_2*In_2 + W_3*In_3 + W_4*In_4 + W_5*In_5 + b$$

- W_1, W_2, W_3, W_4, W5 are the weights assigned to the inputs In_1, In_2, In_3, In_4, In_5, and b is the bias.
- Then in the second step, the **activation function is applied to the linear equation Z1.** The activation function is a nonlinear transformation that is applied to the input before sending it to the next layer of neurons. The importance of the activation function is to inculcate nonlinearity in the model.

4. The whole process described in point 3 is performed in each hidden layer. After passing through every hidden layer, **we move to the last layer i.e our output layer which gives us the final output.**

   The process explained above is known as forwarding Propagation.

5. After getting the predictions from the output layer, the **error is calculated i.e the difference between the actual and the predicted output.**

If the error is large, then the steps are taken to minimize the error and for the same purpose, **Back Propagation is performed.**

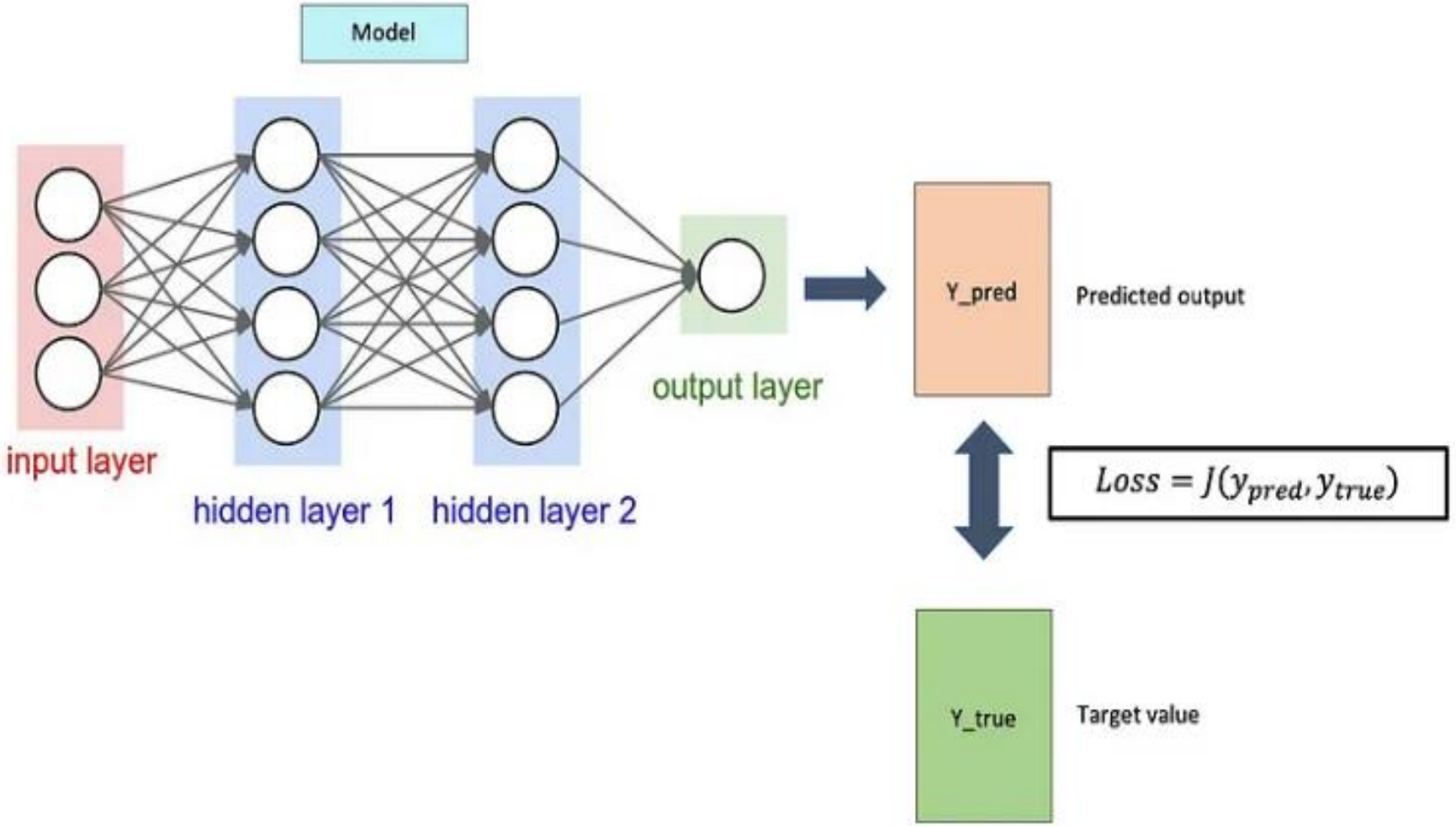
Forward Propagation

- The process of computing the output of a neural network. It involves:

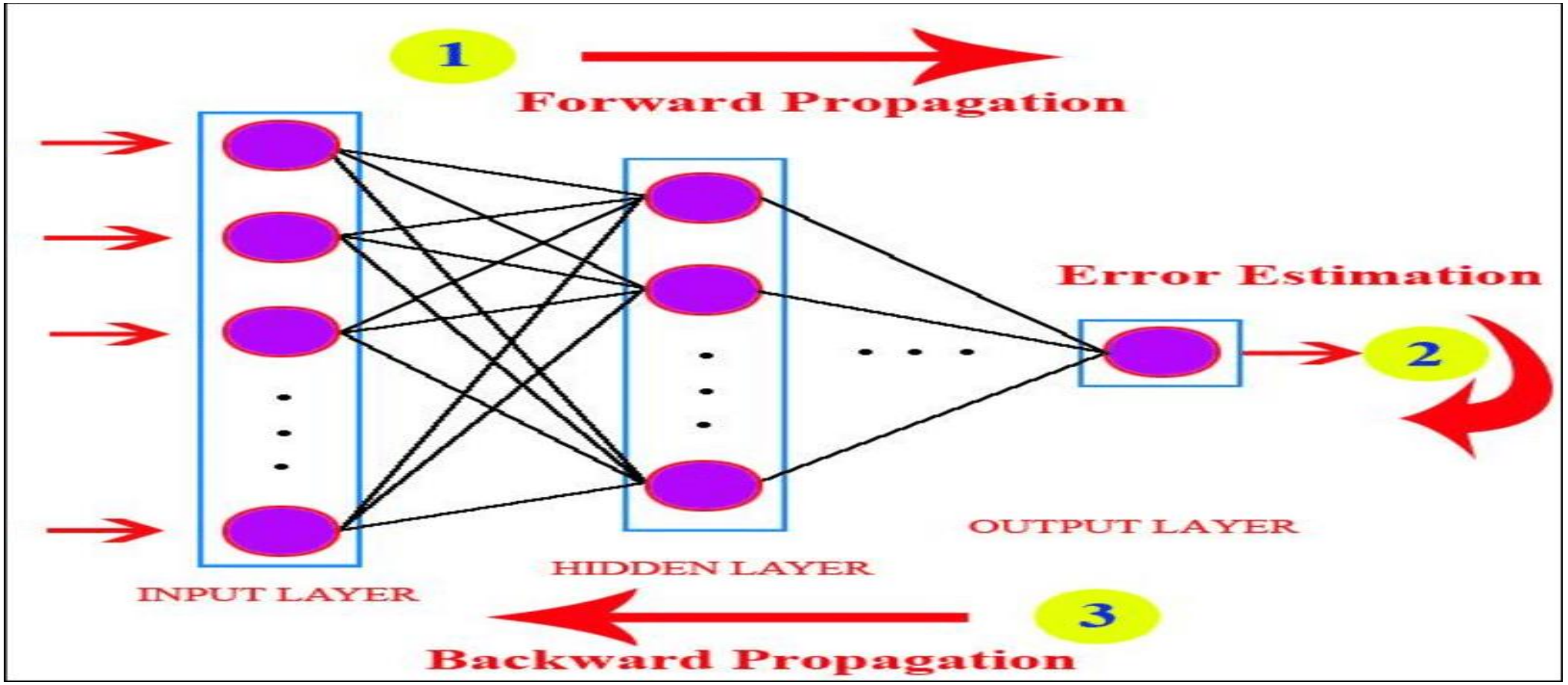
1. Multiplying inputs by weights.
2. Adding biases.
3. Applying the activation function to produce the output.

Loss Function

- A measure of how well the neural network's predictions match the actual results. Common loss functions include:
- **Mean Squared Error (MSE):** For regression tasks.
- **Cross-Entropy Loss:** For classification tasks.


Model
input layer
hidden layer 1
hidden layer 2
output layer
Y_pred
Predicted output
$Loss = J(y_{pred}, y_{true})$
Y_true
Target value

Task	Error type	Loss function	Note
Regression	Mean-squared error	$\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$	Easy to learn but sensitive to outliers (MSE, L2 loss)
	Mean absolute error	$\frac{1}{n}\sum_{i=1}^{n}\lvert y_i - \hat{y}_i\rvert$	Robust to outliers but not differentiable (MAE, L1 loss)
Classification	Cross entropy = Log loss	$-\frac{1}{n}\sum_{i=1}^{n}[y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)] =$	Quantify the difference between two probability


1
Forward Propagation
Error Estimation
2
OUTPUT LAYER
HIDDEN LAYER
INPUT LAYER
3
Backward Propagation

Backpropagation

- The process of updating weights and biases based on the error of the network's predictions. It involves:

1. Computing the gradient of the loss function with respect to each weight using the chain rule.
2. Adjusting weights and biases to minimize the loss function.

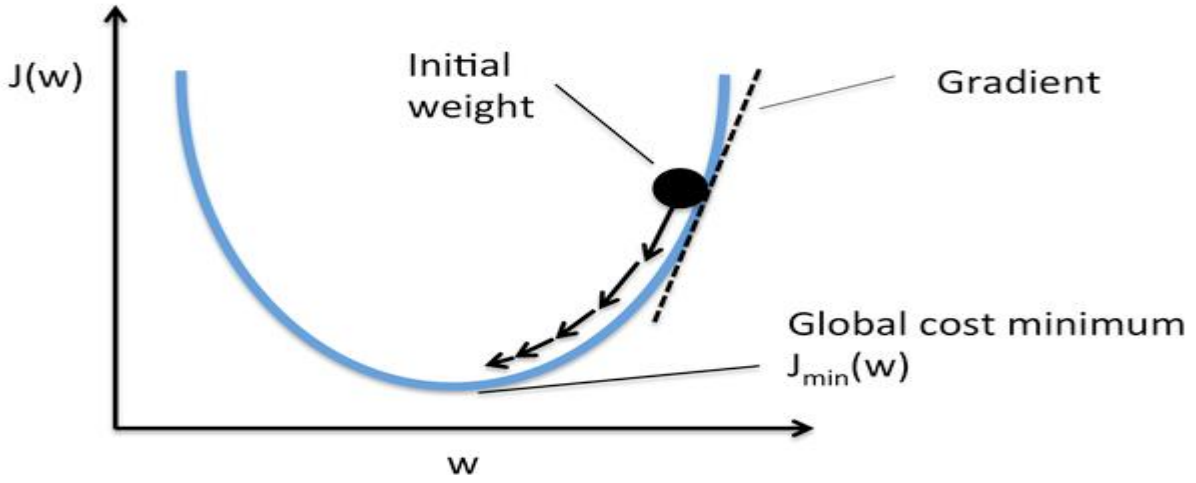
What is Back Propagation and How it works?

- Back Propagation is the process of updating and finding the optimal values of weights or coefficients which helps the model to minimize the error i.e difference between the actual and predicted values.

- How the weights are updated and new weights are calculated?
- **The weights are updated with the help of optimizers**. Optimizers are the methods/ mathematical formulations to change the attributes of neural networks i.e weights to minimizer the error

Back Propagation with Gradient Descent

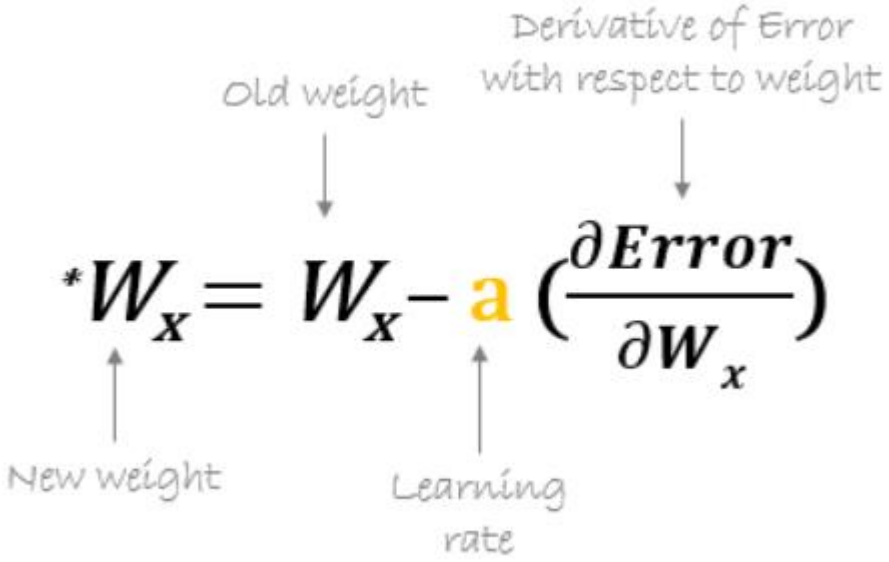
- Gradient Descent is one of the optimizers which helps in calculating the new weights. Let's understand step by step how Gradient Descent optimizes the cost function.
- In the image below, the curve is our cost function curve and our aim is the minimize the error such that J_{min} i.e global minima is achieved.



Steps to achieve the global minima:

1. First, **the weights are initialized randomly** i.e random value of the weight, and intercepts are assigned to the model while forward propagation and **the errors are calculated** after all the computation. (As discussed above)
2. Then the **gradient is calculated i.e derivative of error w.r.t current weights**

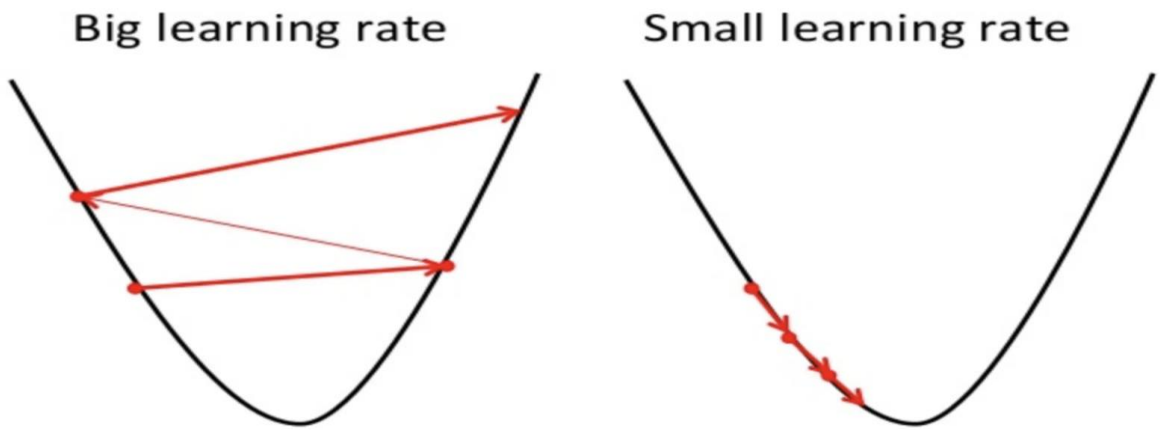
- Then new weights are calculated using the below formula, where **a is the learning rate** which is the parameter also known as step size to control the speed or steps of the backpropagation. It gives additional control on how fast we want to move on the curve to reach global minima.



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

A point to note here is that the learning rate i.e **a 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 minima at all.



BUILDING ANN FROM SCRATCH

Step 1: Load the Dataset

Load the breast cancer dataset from scikit-learn. The Churn Modeling dataset is commonly used in machine learning to predict customer churn in the banking sector. Churn refers to the loss of customers, which can significantly impact a company's profitability. The dataset typically contains various features about customers that can help predict whether they will leave the bank.

Step 2: Split the Dataset

Split the data into training and testing sets. This helps in evaluating the performance of the ANN on unseen data.

Step 3: Normalize the Features

Normalize the input features to ensure all features contribute equally to the training process. This can be done using methods like standard scaling.

Step 4: Initialize the ANN Parameters

Decide on the architecture of the ANN, including the number of layers and the number of neurons in each layer. Initialize the weights and biases for each layer randomly.

Step 5: Define the Activation Functions
Choose activation functions for the neurons in each layer. Common choices are the sigmoid function for the output layer (binary classification).

Step 6: Forward Propagation
Implement the forward propagation process, where the input data passes through the network layer by layer, applying the weights, biases, and activation functions to produce the final output.

Step 7: Compute the Loss
Calculate the loss using an appropriate loss function. For binary classification, the binary cross-entropy loss is typically used.

Step 8: Backward Propagation
Implement backward propagation to compute the gradients of the loss function with respect to the weights and biases. This involves calculating the gradient of the loss function from the output layer back to the input layer.

Step 9: Update the Weights and Biases
Use an optimization algorithm, such as gradient descent, to update the weights and biases using the gradients computed during backward propagation. This step aims to minimize the loss function.

Step 10: Train the ANN
Iterate through the training dataset for a specified number of epochs. In each epoch, perform forward propagation, compute the loss, perform backward propagation, and update the weights and biases.

Step 11: Evaluate the ANN
After training, evaluate the performance of the ANN on the testing set. This involves using the trained model to make predictions on the test data and comparing these predictions to the true labels to calculate metrics like accuracy.

Step 12: Make Predictions
Use the trained ANN to make predictions on new or unseen data by performing forward propagation with the learned weights and biases.


ANY
QUESTIONS

Thank You