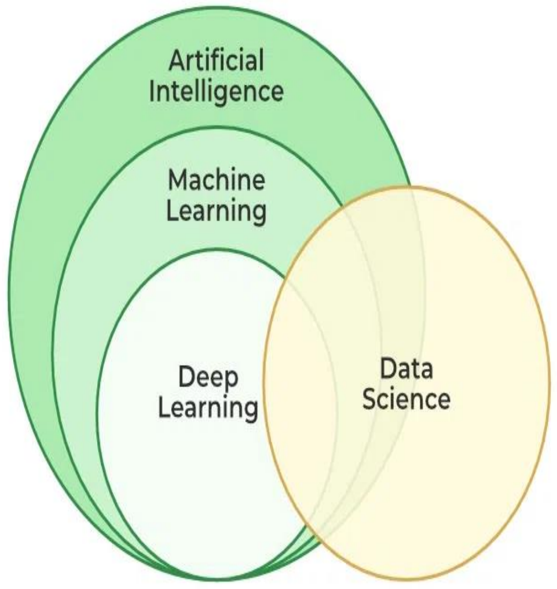
Deep Learning Techniques

II M.Sc (DS / CS)

Unit - II

Introduction

- Deep learning is a subset of machine learning, which is essentially a neural network with three or more layers to solve complex problems.
- These neural networks attempt to simulate the behavior of the human brain
- Deep learning is a key technology behind driverless cars, enabling them to recognize a stop sign, or to distinguish a pedestrian from a lamppost.



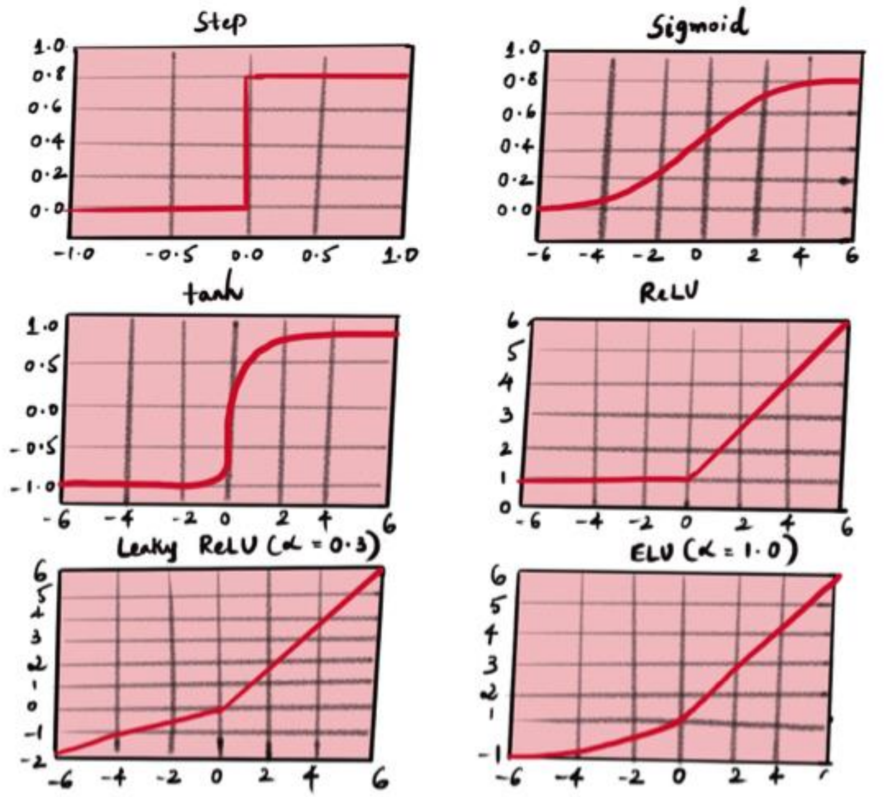
Feed Forward Neural Networks

- Feedforward neural networks were among the first and most successful learning algorithms.
- They are also called deep networks, multi-layer perceptron (MLP), or simply neural networks.
- As data travels through the network's artificial mesh, each layer processes an aspect of the data, filters outliers, spots familiar entities and produces the final output.
- Feedforward neural networks are made up of the following:
- Input layer: This layer consists of the neurons that receive inputs and pass them on to the other layers. The number of neurons in the input layer should be equal to the attributes or features in the dataset.
- Output layer: The output layer is the predicted feature and depends on the type of model you're building.
- Hidden layer: In between the input and output layer, there are hidden layers based on the type of model. Hidden layers contain a vast number of neurons which apply transformations to the inputs before passing them. As the network is trained, the weights are updated to be more predictive.

Cont..

- Neuron weights: Weights refer to the strength or amplitude of a connection between two neurons. Weights are often initialized to small random values, such as values in the range 0 to 1.
- There are several activation functions for different use cases. The most commonly used activation functions are relu, tanh and softmax
- The activation function decides whether a neuron should be activated or not by calculating the weighted sum and further adding bias to it.

The purpose of the activation function is to introduce non-linearity into the output of a neuron.



Cont..

- This equation represents how a neural network processes the input data at each layer and eventually produces a predicted output value.

$$\hat{y} = \sigma(w^T x + b)$$

- To **train** — the process by which the model maps the relationship between the training data and the outputs — the neural network updates its hyperparameters, the weights, *wT*, and biases, *b,* to satisfy the equation above.
- Each training input is loaded into the neural network in a process called **forward propagation**.
- Once the model has produced an output, this predicted output is compared against the given target output in a process called **backpropagation** — the hyperparameters of the model are then adjusted so that it now outputs a result closer to the target output.
- This is where loss functions come in.

Calculating Loss

- A loss function quantifies how “good” or “bad” a given model is in classifying the input data.
- In most learning networks, the loss is calculated as the difference between the actual output and the predicted output.
- Measures how well the neural network models the training data.
- During training, we aim to minimize this loss between the predicted and target outputs.
- The function that is used to compute this error is known as loss function J(.).
- The **hyperparameters** are adjusted to minimize the average loss — we find the weights, *wT*, and biases, *b*, that minimize the value of *J* (average loss).

Loss Function cont..

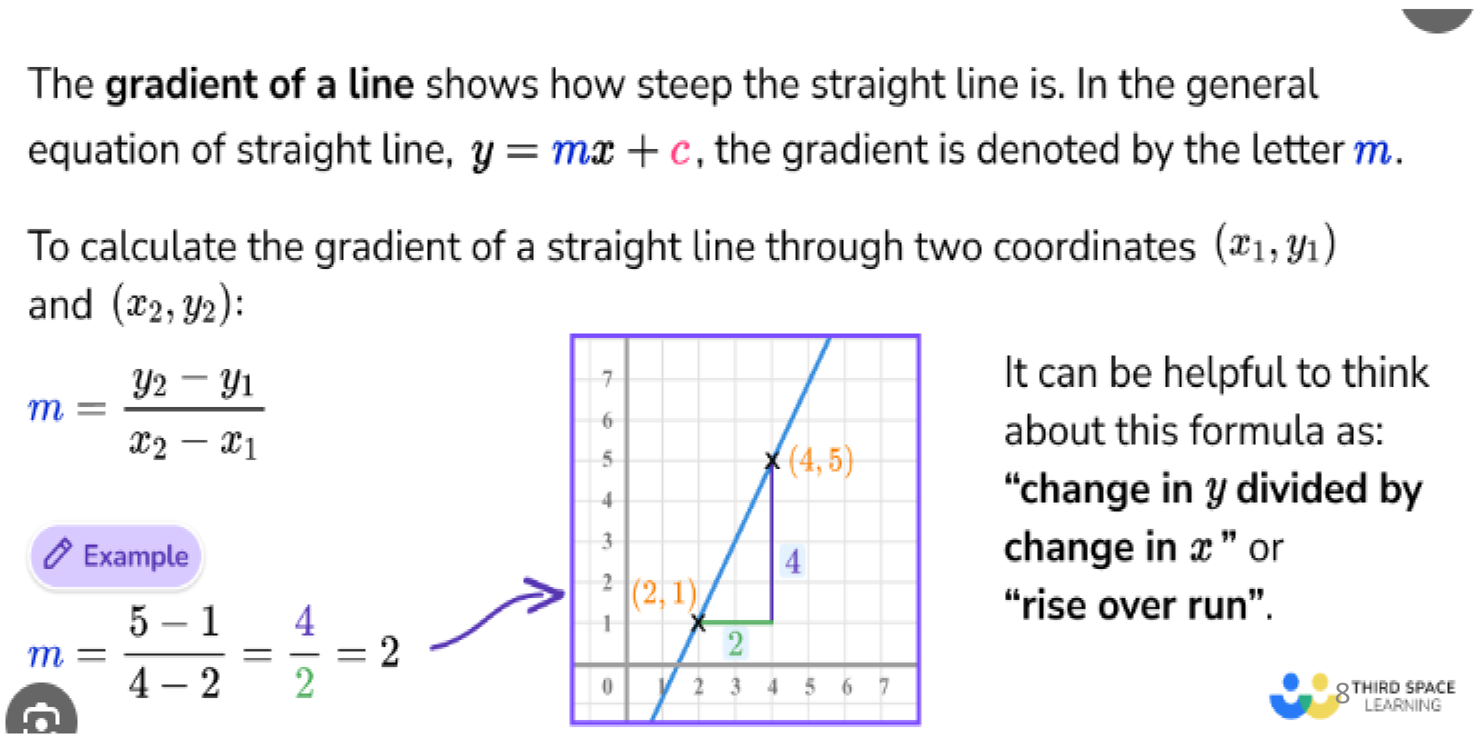
$$J(w^T, b) = \frac{1}{m}\sum_{i=1}^{m} L(\hat{y}^{(i)}, y^{(i)})$$

- Where w –weight
- b – bias
- m – dataset size
- y – the actual value of the data point. Also known as true value.
- ŷ – the predicted value of the data point. This value is returned by the model.

Gradient Descent

Gradient - A Slope. Descent:- That means descending.

- Gradient descent is an optimization algorithm which is commonly-used to train machine learning models and neural networks.
- A gradient is a measurement that quantifies the steepness of a line or curve.
- Mathematically, it details the direction of the ascent or descent of a line.
- Descent is the action of going downwards. Therefore, the gradient descent algorithm quantifies downward motion.



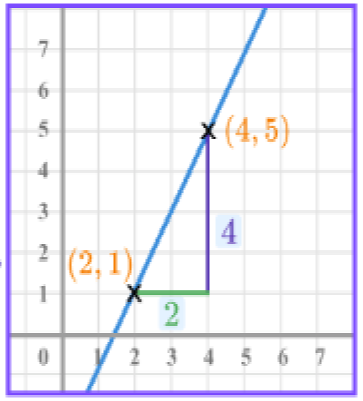
The **gradient of a line** shows how steep the straight line is. In the general equation of straight line, $y = mx + c$, the gradient is denoted by the letter m.

To calculate the gradient of a straight line through two coordinates (x_1, y_1) and (x_2, y_2):

$$m = \frac{y_2 - y_1}{x_2 - x_1}$$

Example

$$m = \frac{5-1}{4-2} = \frac{4}{2} = 2$$



It can be helpful to think about this formula as: **"change in y divided by change in x"** or **"rise over run"**.

THIRD SPACE LEARNING

Cont..

- Gradient descent is an optimization algorithm used in machine learning to minimize the cost function of a model by iteratively adjusting its parameters in the opposite direction of the gradient.
- *It is a function that measures the performance of a model for any given data. Cost Function quantifies the error between predicted values and expected values and presents it in the form of a single real number.*
- A gradient simply measures the change in all weights with regard to the change in error.
- The gradient is the slope of the cost function, and by moving in the direction of the negative gradient, the algorithm can converge to the optimal set of parameters that best fits the training data.
- Gradient descent can be applied to a wide range of machine learning algorithms, including linear regression, logistic regression, neural networks, and support vector machines.
- There are different variations of gradient descent, including batch gradient descent, stochastic gradient descent, and mini-batch gradient descent, each with its own advantages and limitations.

How Gradient Descent works?

- Assume a climber downhill to the bottom of the valley
- He will start by stepping in the direction that feels lower to him and he will continue doing so till he reaches the lowest point of the valley.
- So, it is a minimization function that minimizes a given function
- Equation says about what gradient descent works

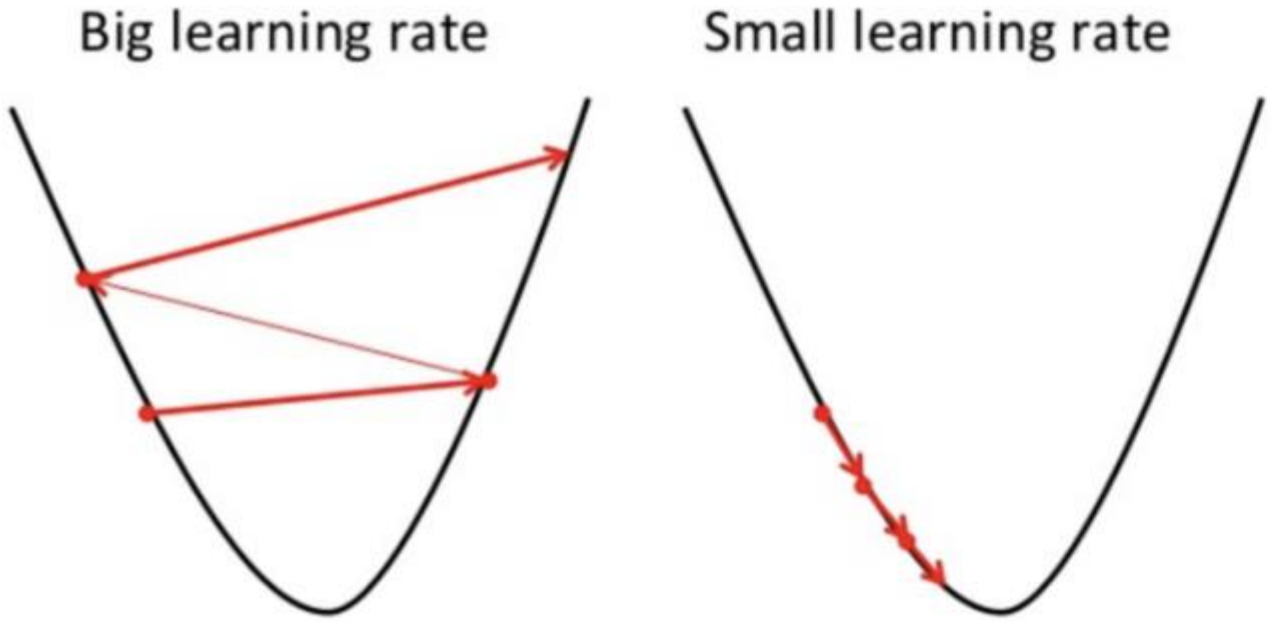
$$b = a - \gamma \nabla f(a)$$

- b is the next position of the climber
- a is the current position
- - refers minimization part of gradient descent algorithm
- Gamma is the middle of waiting factor(i.e., learning rate)
- Delta f(a) / (Δf(a)) is simply the direction of the steepest descent (direction of the minimum point)
- Equation tells which position next to go

Cont..

- the term γΔ f(a) is subtracted from a because the goal is to move against the gradient, toward the local minimum.
- How do we minimize the cost function?
- It is done by making required changes in the learning rate the algorithm takes to reach the local or global minima.
- ***Learning Rate -*** Learning Rate is the size of the steps that are taken to reach the local minimum of the function(Basically a Hyperparameter).
- Need to take correct steps to reach global minimum.
- Number of steps would decide, how soon we will reach global minima, which is the objective of learning rate.
- If learning rate is too high, it will skip global minima, it will be bouncing between the convex function of gradient.
- If learning rate is too small, it will take long time to reach the minima, which will require many iterations.

Cont..



Types

- **BATCH GRADIENT DESCENT**
- Batch Gradient Descent, also called vanilla gradient descent, is the simplest form of Gradient Descent.
- Its main feature is that we take small steps in the direction of the gradient.
- It basically calculates the error for each sample in the data and updates the model only after the training example has been calculated.
- Advantage of Vanilla Gradient Descent is that it is computationally efficient.
- ***Stochastic Gradient Descent***
- Stochastic Gradient Descent (***SGD***) on the other hand, does this for each training example on the dataset, which means it updates the parameter for each training example one by one.
- This makes this one much faster than Vanilla Gradient Descent but quite computationally expensive.
- Expensive because the frequent updates takes into account the noisy gradients as well which may cause the number of iterations to reach the minima longer.

Cont..

- ***Mini Batch Gradient Descent***
- It is amalgamation of both the Vanilla Descent and SGD.
- Takes mini batches of a fixed size and then go on to update for each of the batches.
- This is the go-to method to reach the minima efficiently.
- Therefore, it creates a balance between the robustness of stochastic gradient descent and the efficiency of batch gradient descent.
- The usual batch sizes ranges from 50 to 250 but there is no clear rule as to how much should be the usual batch size
- so, based on requirements keep the batch size changing.

Back Propagation Algorithm

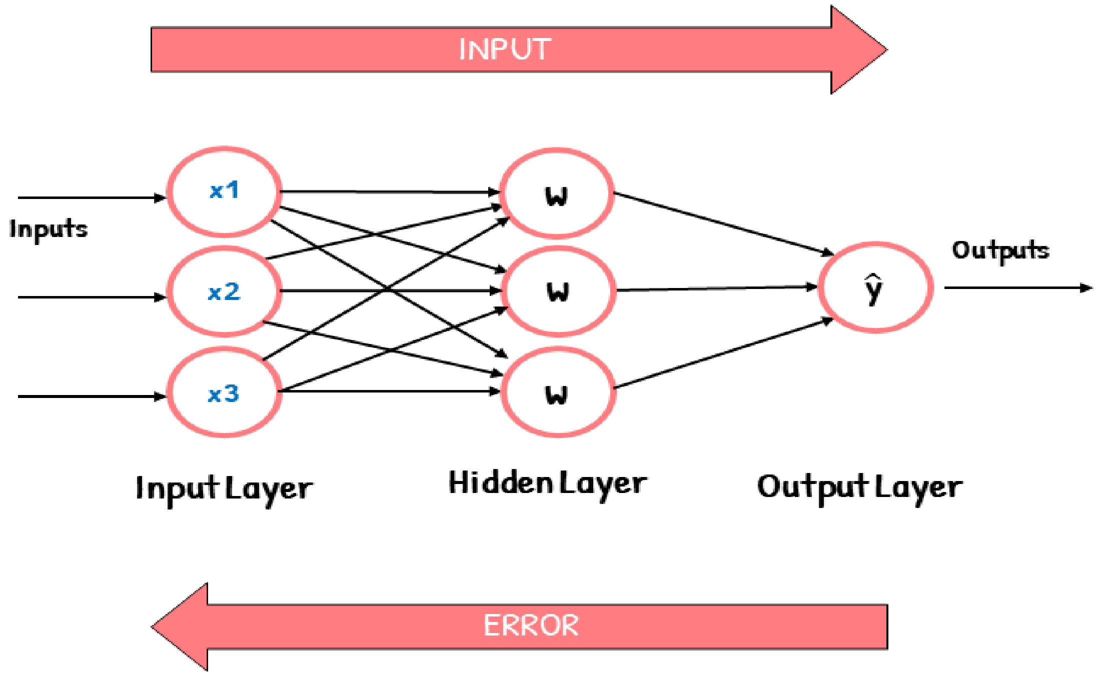
- BP is a training algorithm used for training feedforward neural networks.
- It plays an important part in improving the predictions made by neural networks.
- This is because backpropagation is able to improve the output of the neural network iteratively.
- Backpropagation is a process involved in training a neural network.
- It involves taking the error rate of a forward propagation and feeding this loss backward through the neural network layers to fine-tune the weights.
- Backpropagation aims to minimize the cost function by adjusting network's weights and biases.
- The level of adjustment is determined by the gradients of the cost function with respect to those parameters.
- Back propagation is executed to compute the gradient.

How backpropagation and gradient descent interconnected?

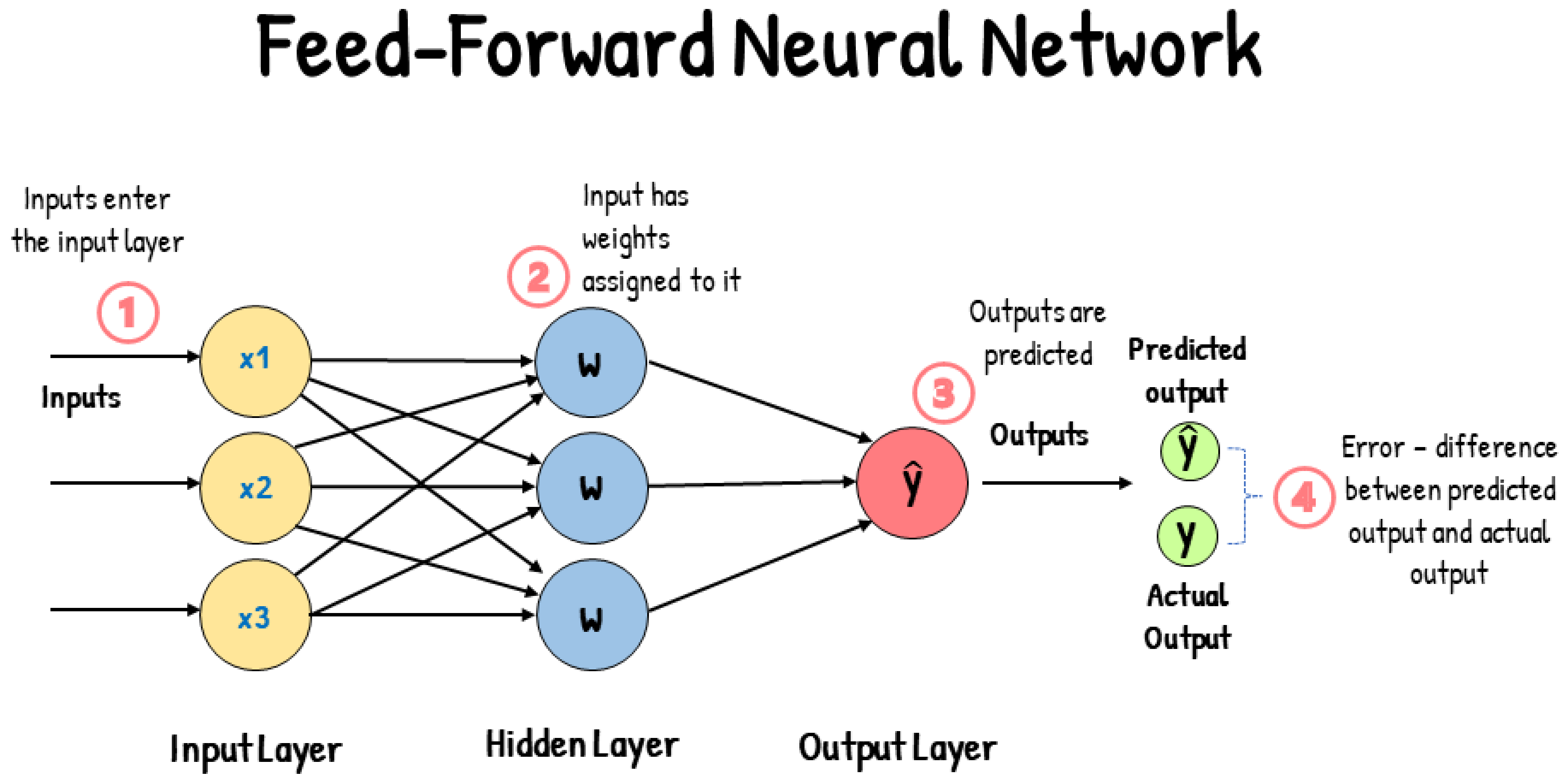
- Together, backpropagation and gradient descent is used for the purpose of improving the prediction accuracy of neural networks.
- They help improve prediction accuracy by reducing the output error in neural networks.
- Backpropagation plays the role of calculating the gradient, while gradient descent plays the role of descending through the gradient.

Recap

- In a feedforward neural network, the input moves forward from the input layer to the output layer.
- Backpropagation helps improve the neural network's output.
- It does this by propagating the error backward from the output layer to the input layer.



- When NN gives incorrect output, this leads to output error.
- This error is the difference between actual and predicted output
- A cost function measures this error.

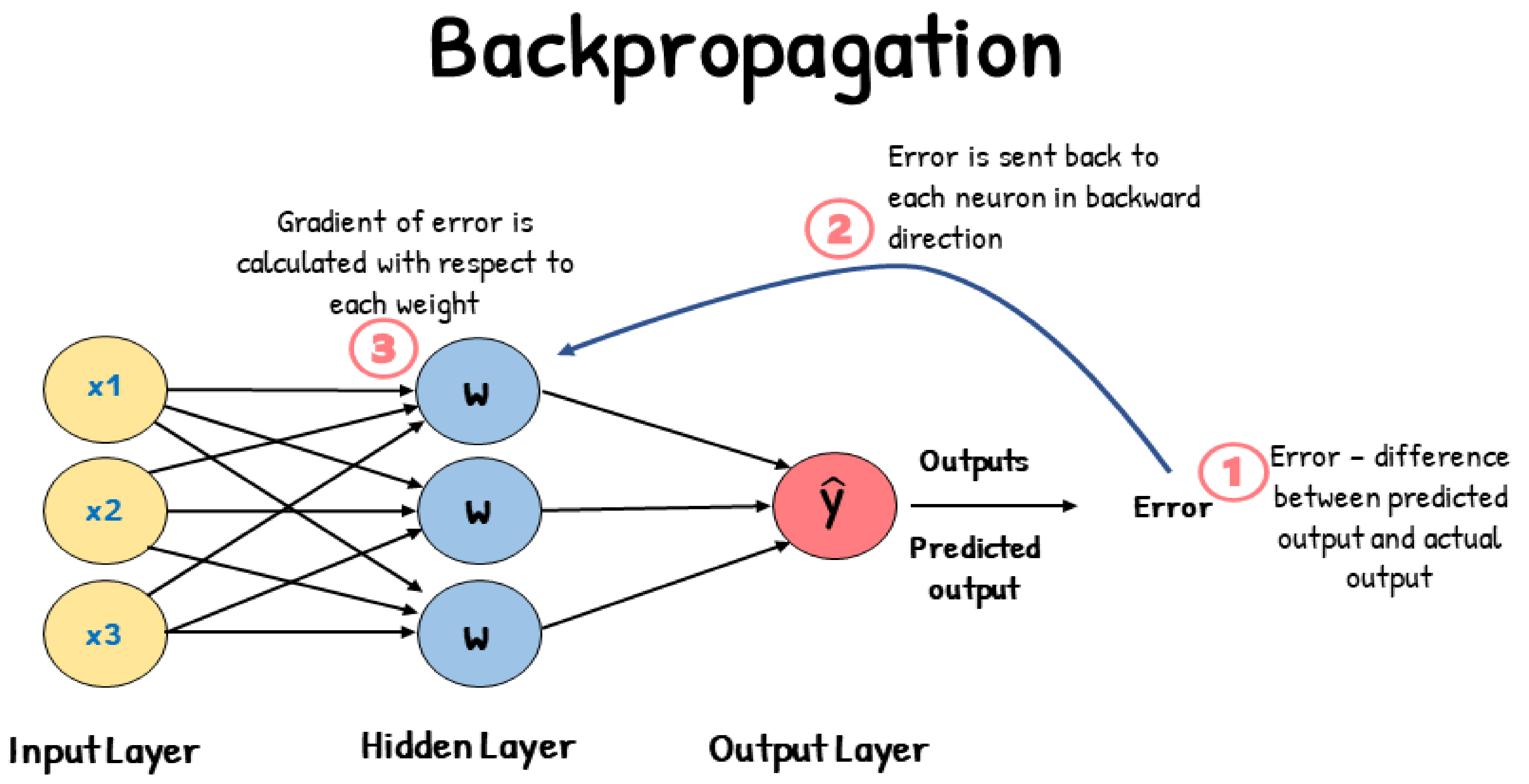

Feed-Forward Neural Network
Inputs enter the input layer
1
Inputs
x1
x2
x3
2
Input has weights assigned to it
w
w
w
Outputs are predicted
3
ŷ
Outputs
Predicted output
ŷ
y
Actual Output
4
Error – difference between predicted output and actual output
Input Layer
Hidden Layer
Output Layer

Cont..

- The **cost function (J) indicates how accurately the model performs**.
- It tells us how far-off our predicted output values are from our actual values.
- It is also known as the error.
- Because the cost function quantifies the error, we aim to minimize the cost function.
- The goal is to reduce the output error
- Since the weights affect the error, the weights need to be readjusted.
- To adjust the weights such that we have a combination of weights that minimizes the cost function.
- Here the task of backpropagation comes in:

Cont..

- BP allows us to readjust our weights to reduce output error.
- The error is propagated backward during backpropagation from the output to the input layer.
- This error is then used to calculate the gradient of the cost function with respect to each weight.

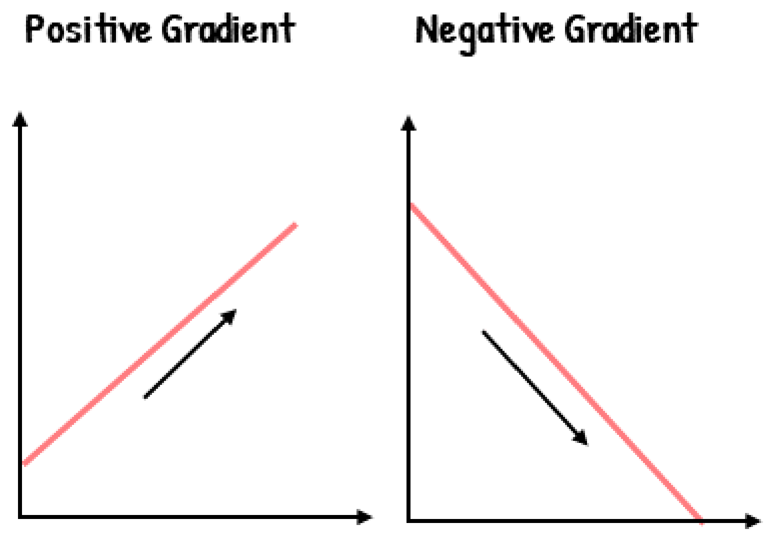

Backpropagation
Error is sent back to each neuron in backward direction
2
Gradient of error is calculated with respect to each weight
3
x1
x2
x3
w
w
w
ŷ
Outputs
Predicted output
Error
1
Error – difference between predicted output and actual output
Input Layer
Hidden Layer
Output Layer

Cont..

- Backpropagation aims to calculate the negative gradient of the cost function.
- This negative gradient is what helps in adjusting of the weights. It gives us an idea of how we need to change the weights so that we can reduce the cost function.
- Backpropagation uses the chain rule to calculate the gradient of the cost function.
- The chain rule involves taking the derivative.
- This involves calculating the partial derivative of each parameter.
- These derivatives are calculated by differentiating one weight and treating the other(s) as a constant. As a result of doing this, gradient is derived.
- Since gradient is calculated, weight can be adjusted.

Gradient Descent

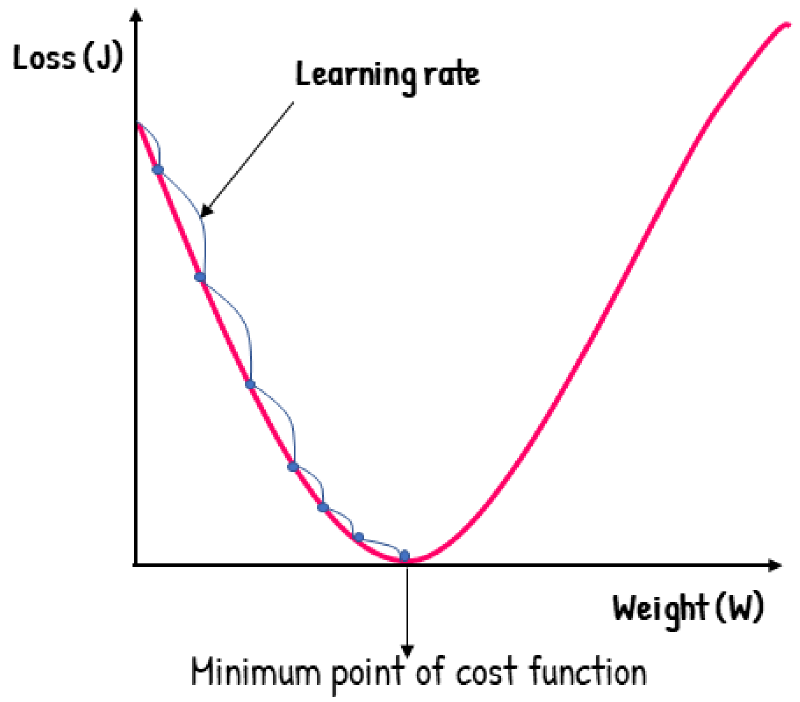
- The weights are adjusted using a process called gradient descent
- Gradient descent is an optimization algorithm that is used to find the weights that minimize the cost function
- Minimizing the cost function means getting to the minimum point of the cost function.
- So, gradient descent aims to find a weight corresponding to the cost function's minimum point
- To find this weight, we must navigate down the cost function until we find its minimum point.
- To navigate the cost function, two things has to be done: the direction in which to navigate and the size of the steps for navigating.
- The direction for navigating the cost function is found using the gradient.
- To know in which direction to navigate, gradient descent uses backpropagation. More specifically, it uses the gradients calculated through backpropagation.
- The aim is to find the negative gradient. This is because a negative gradient indicates a decreasing slope. A decreasing slope means that moving downward will lead us to the minimum point.


Positive Gradient
Negative Gradient

Cont..

- *Step Size*
- The step size for navigating the cost function is determined using the learning rate.
- *Learning Rate*
- The learning rate is a tuning parameter that determines the step size at each iteration of gradient descent. It determines the speed at which to move down the slope.
- The step size plays an important part in ensuring a balance between optimization time and accuracy.
- The step size is measured by a parameter alpha (α).
- A small α means a small step size, and a large α means a large step size.
- If the step sizes are too large, minimum point would be missed completely. This leads to inaccurate results
- If the step size is too small, the optimization process could take too much time. This will lead to a waste of computational power.

Learning Rate

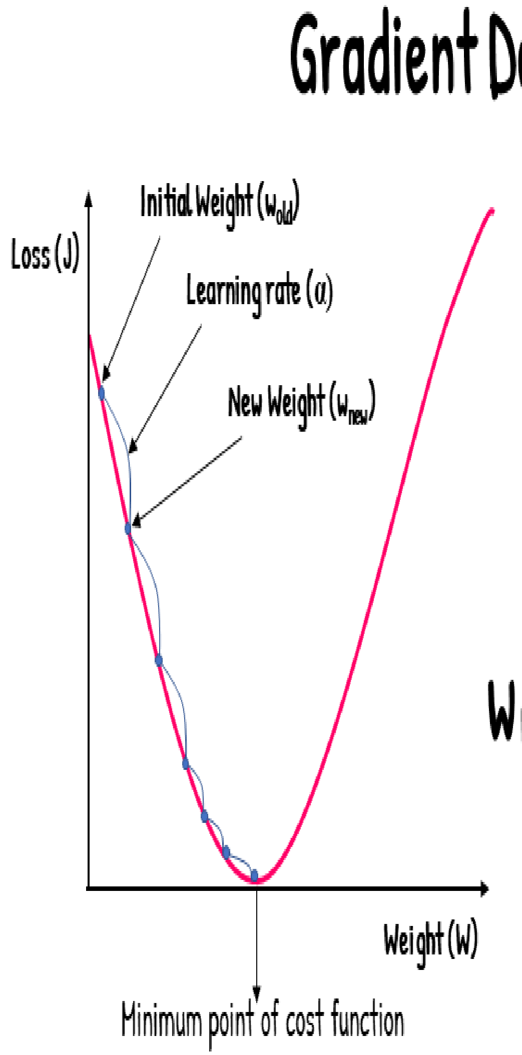


Cont..

- The step size is evaluated and updated according to the behavior of the cost function.
- The higher the gradient of the cost function, the steeper the slope and the faster a model can learn (high learning rate).
- A high learning rate results in a higher step value,

and a lower learning rate results in a lower step value.

- If the gradient of the cost function is zero, the model stops learning.
- *Descending the cost function:*
- Navigating the cost function consists of adjusting the weights.
- The weights are adjusted using the following formula



$$w_{new} = w_{old} - \alpha \frac{\delta J}{\delta w}$$

Steps from the graph of the cost function

- To start descending the cost function, we first initialize a random weight.
- Then, we take a step down and obtain a new weight using the gradient and learning rate. With the gradient, we can know which direction to navigate. We can know the step size for navigating the cost function using the learning rate.
- We are then able to obtain a new weight using the gradient descent formula.
- We repeat this process until we reach the minimum point of the cost function.
- Once we’ve reached the minimum point, we find the weights that correspond to the minimum of the cost function.

References

- https://builtin.com/data-science/feedforward-neural-network-intro
- https://builtin.com/data-science/gradient-descent
- https://www.ibm.com/topics/gradient-descent
- https://neptune.ai/blog/backpropagation-algorithm-in-neural-networks-guide
- https://towardsdatascience.com/understanding-backpropagation-algorithm-7bb3aa2f95fd
- https://www.youtube.com/watch?v=tIeHLnjs5U8 – to understand chain rule in backpropagation
- https://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example/
- https://www.analyticsvidhya.com/blog/2023/01/gradient-descent-vs-backpropagation-whats-the-difference/