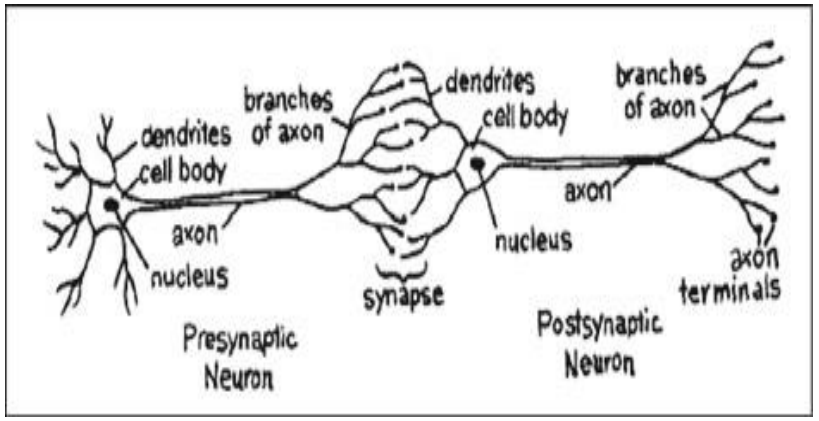
Chapter .2

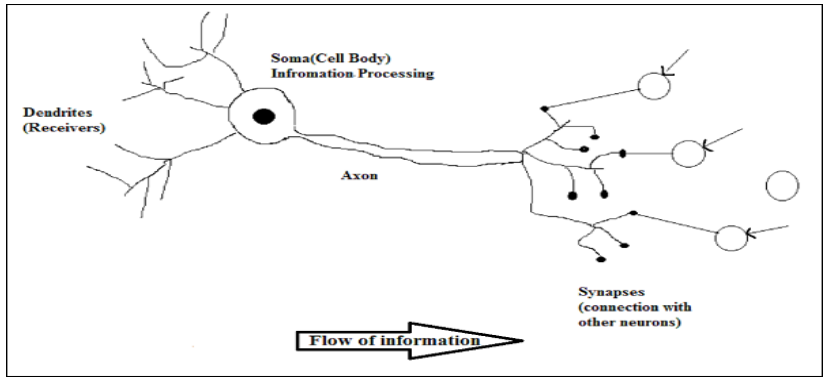
Introduction to A.N.N

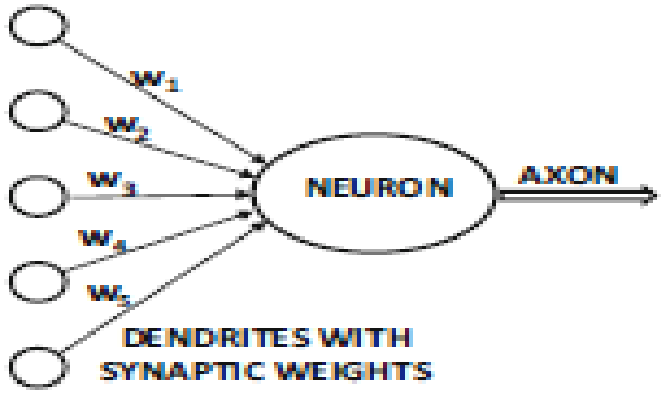
Introduction

A.N.N is popular machine learning techniques (parallel computing devices) that simulate the mechanism of learning in biological organisms.

- The brain has $10^{10} - 10^{11}$ cells that are highly connected
- Every cell (called neuron) is connected to about





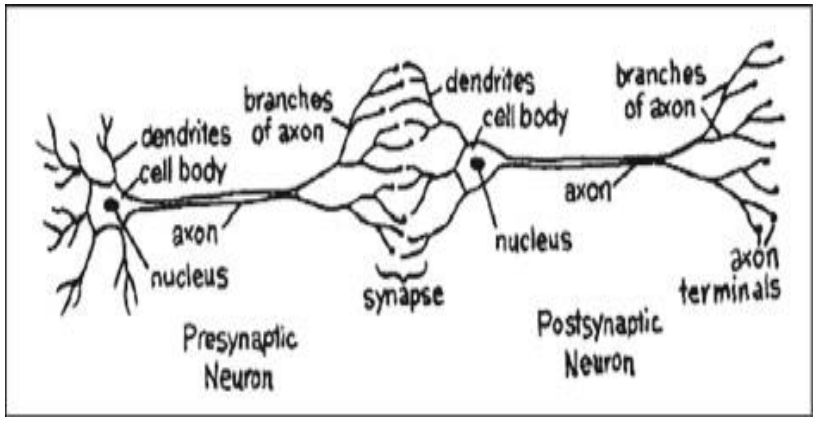


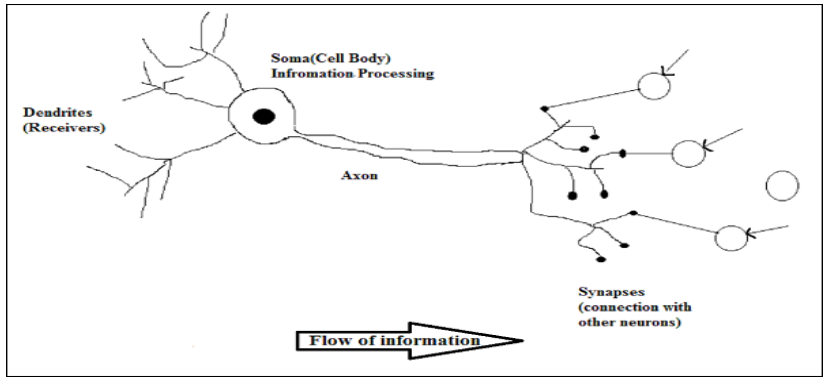
Biological Neural Network BNN	Artificial Neural Network ANN
Soma	Node
Dendrites	Input
Synapse	Weights or Interconnections
Axon	Output

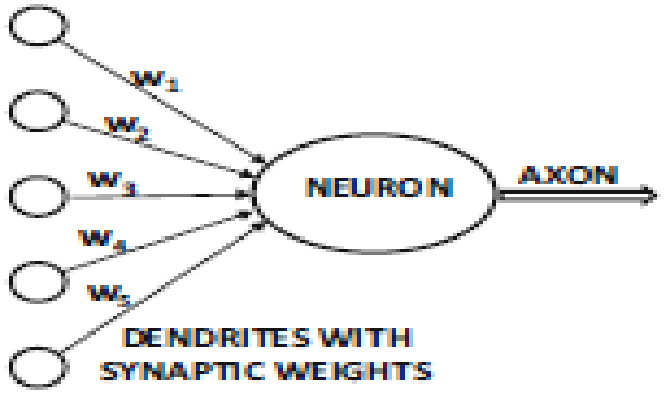
Introduction

A.N.N is popular machine learning techniques (parallel computing devices) that simulate the mechanism of learning in biological organisms.

- The brain is a highly complex, non-linear, has 10^{10} – 10^{11} cells that are highly connected
- Every cell (called neuron) is connected to about 10^4 neuron



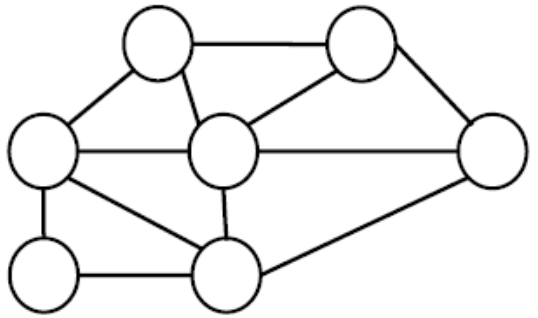


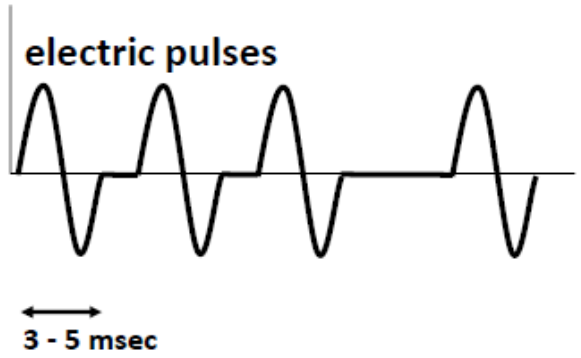


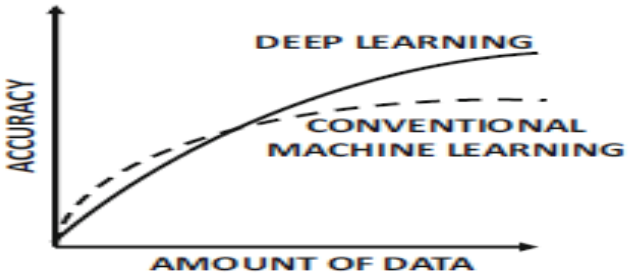
Biological Neural Network BNN	Artificial Neural Network ANN
Soma	Node
Dendrites	Input
Synapse	Weights or Interconnections
Axon	Output

Humans Versus Computers

- Success of the brain is a result of the parallelism of this huge network of cells
- Time constant ≈ 3 – 5 msec
- Traditional computer's time constant ~ 10-9 or nano sec
- The "time constant" of the brain is significantly larger than that of the traditional computers
- Brain is superior because of the massive parallelism

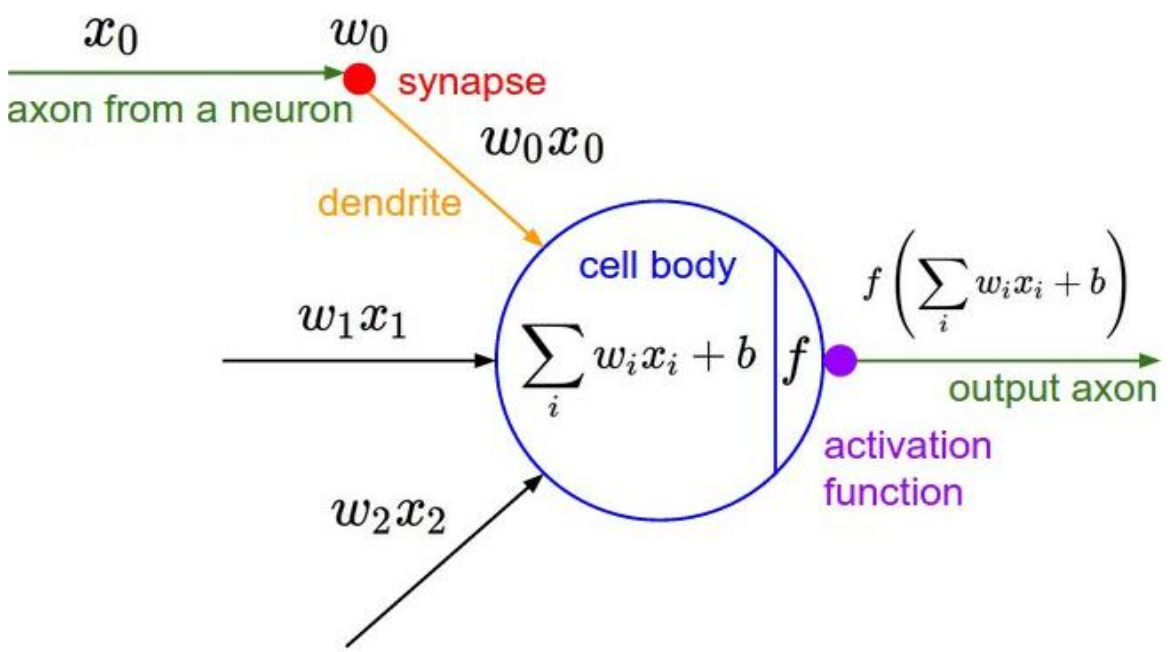






Model of Artificial Neural Network

- Mimic how the brain works
- $W_j \equiv$ weight à gives the degree of how input cell j affects the receiving cell I
- b ≡ some constant called bias or threshold
- Every neuron produces output X_j
- If neuron j is connected to neuron i then X_j will have an effect on X_i
- Each neuron can have a different effect, hence, we multiply by weight w_j
- Weights are the parameters that encode the information in the brain



Artificial Neural Network

- Every neuron produces output X_j
- If neuron j is connected to neuron i then X_j will have an effect on X_i
- Each neuron can have a different effect, hence, we multiply by weight w_j
- Weights act like the "storage" in computers
- When a human encounters a new experience, the weights of his brain gets adjusted
- This adjustment results in his learning of the new experience
- Memories are encoded in the weights.
- We try in the field of (NNs) to exploit the learning feature observed in the brain.
- The weights determine the functionality of the model
- We apply some learning algorithm that adjusts the weights in small steps
- The goal is to lead the NN to learn to implement the given problem

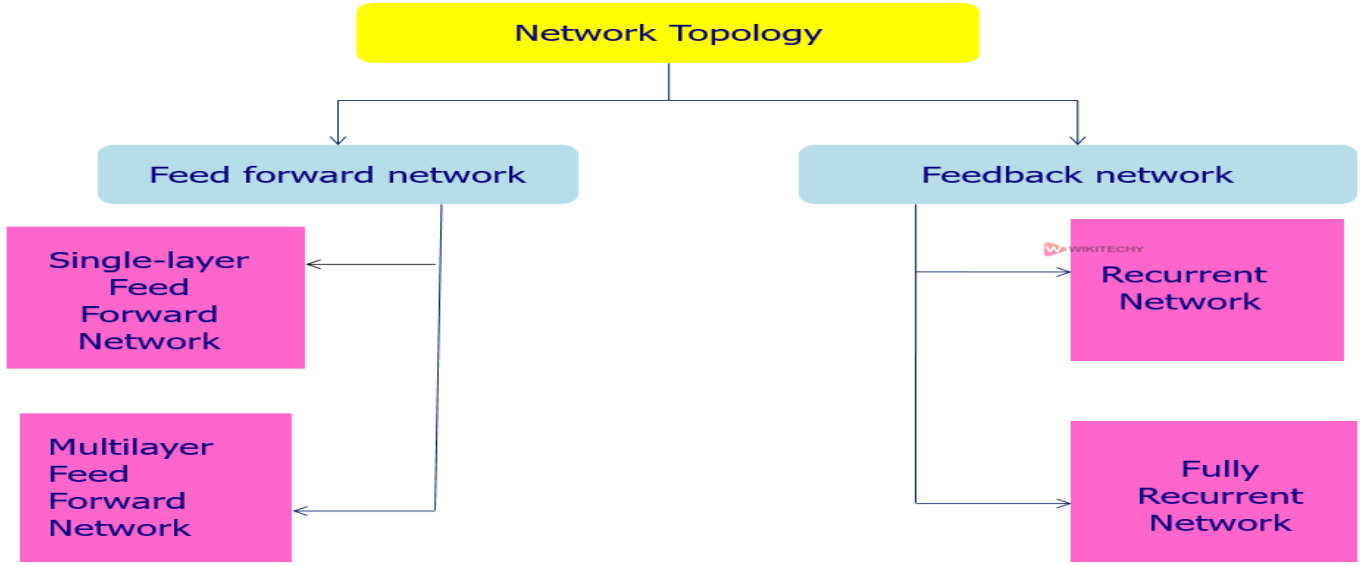
Artificial Neural Network

Processing of ANN depends upon the following three building blocks

- Network Topology
- Adjustments of Weights or Learning
- Activation Functions

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:

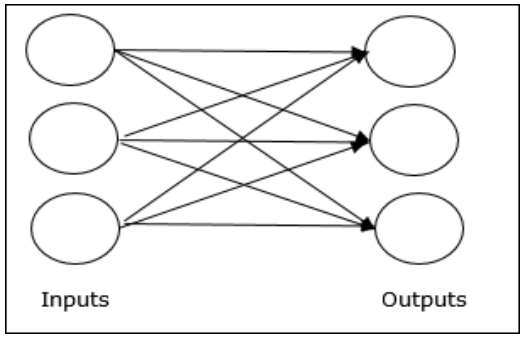


Artificial Neural Network (**Network Topology**)

1) Feedforward Network

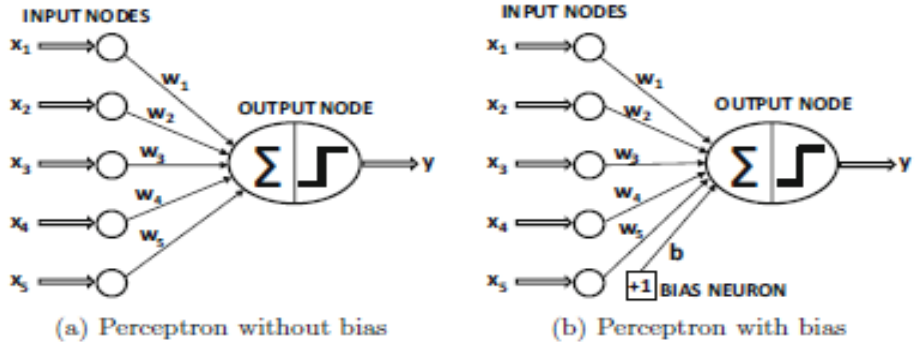
It is a non-recurrent network having processing units/nodes in layers and all the nodes in a layer are connected with the nodes of the previous layers. The connection has different weights upon them. It may be divided into two types of architecture:

(a) Single-layer feedforward network (perceptron)



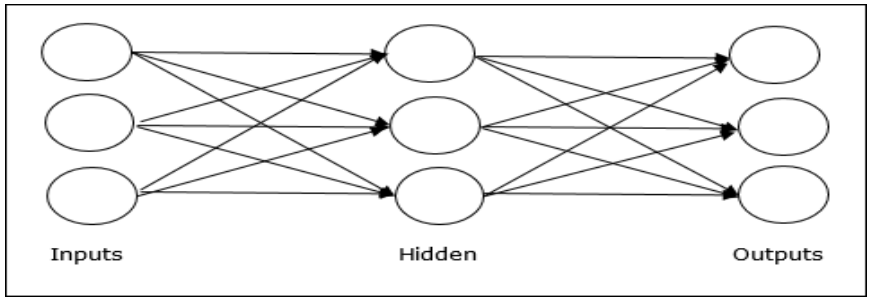
$$\bar{W} = [w_1 \ldots\ldots w_d]$$

$$\bar{W}.\bar{X} = \sum_{i-1}^{d} w_i x_i$$



(a) Perceptron without bias (b) Perceptron with bias

(B) Multilayer feedforward network

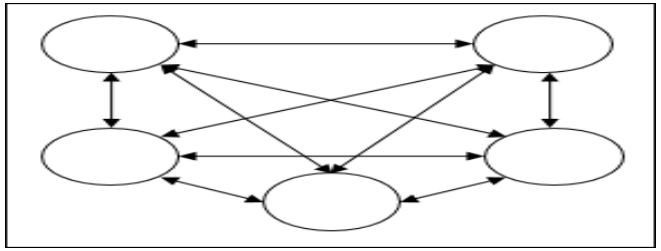


*Artificial Neural Network (*Network Topology*)*

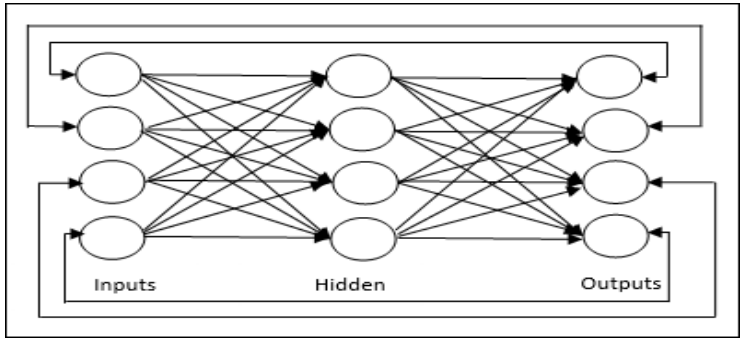
2) Feedback Network

It is 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:

(a) 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.

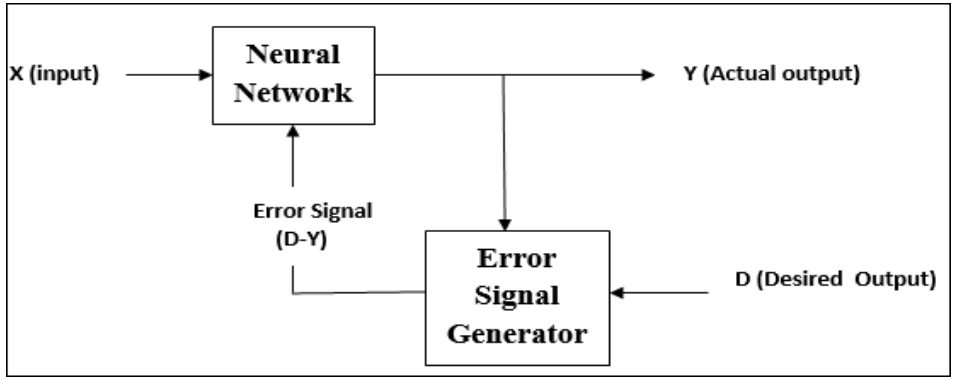


(b) Jordan Network: It is a closed loop network in which the output will go to the input again as feedback as shown in the following diagram.

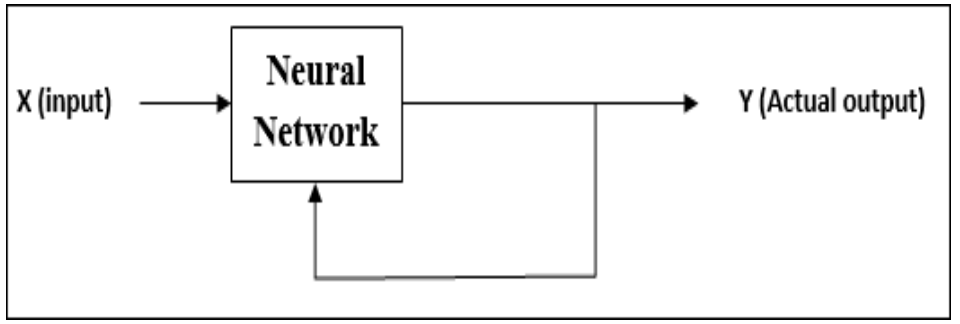


Artificial Neural Network (Adjustments Of Weights Or Learning)

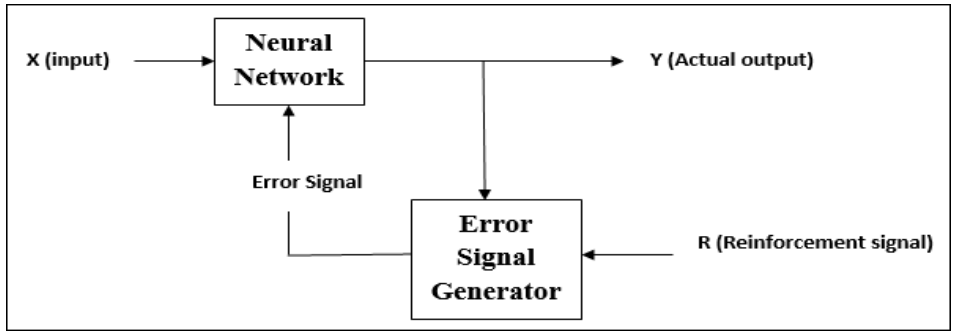
1) Supervised learning:



2) Unsupervised learning:

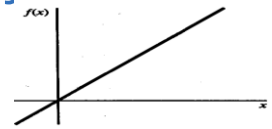


3) Reinforcement learning:



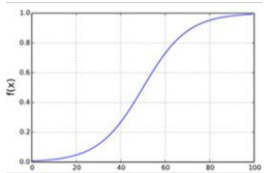
Artificial Neural Network (Activation Functions)

- **Linear Activation Function: Or** identity function as it performs no input editing. $F(x) = x$



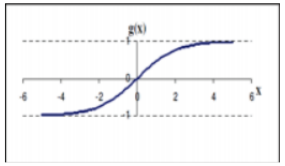
- **Sigmoid Activation Function:** It is of two types as follows
  - **Binary sigmoidal function:** Performs input editing between 0 and 1. It is positive in nature. It is always bounded, which means its output cannot be less than 0 and more than 1. Defined a**s**

$$F(x) = \text{sigm}(x) = \frac{1}{1+\exp(-x)}$$



  - **Bipolar sigmoidal function:** Performs input editing between -1 and 1. It can be positive or negative in nature. It is always bounded, which means its output cannot be less than -1 and more than 1. Defined as

$$F(x) = \text{sigm}(x) = \frac{2}{1+\exp(-x)} - 1 = \frac{1-\exp(x)}{1+\exp(x)}$$



Name	Input/Output Relation	Icon	MATLAB Function
Hard Limit	$a = 0 \quad n<0$ $a = 1 \quad n\geq 0$		hardlim
Symmetrical Hard Limit	$a = -1 \quad n<0$ $a = +1 \quad n\geq 0$		hardlims
Linear	$a = n$		purelin
Saturating Linear	$a = 0 \quad n<0$ $a = n \quad 0\leq n\leq 1$ $a = 1 \quad n>1$		satlin
Symmetric Saturating Linear	$a = -1 \quad n<-1$ $a = n \quad -1\leq n\leq 1$ $a = 1 \quad n>1$		satlins
Log-Sigmoid	$a = \frac{1}{1+e^{-n}}$		logsig
Hyperbolic Tangent Sigmoid	$a = \frac{e^{n}-e^{-n}}{e^{n}+e^{-n}}$		tansig
Positive Linear	$a = 0 \quad n<0$ $a = n \quad 0\leq n$		poslin

Loss Function

- It is a measurable way to gauge the performance and accuracy of a machine learning model.
- Acts as a guide for the learning process within a model or machine learning algorithm.
- Also referred to as the error function or cost function, quantifies the difference between the predicted outputs of a machine learning algorithm and the actual target values.
- The learning algorithm in a M. L model are optimized to minimize the prediction error.
- Most loss functions apply to **regression and classification** machine learning problems.
- When exploring the topic of loss function, the topic of Empirical Risk Minimization (ERM) comes up. ERM is an approach to selecting the optimal parameters of a machine learning algorithm that minimizes the empirical risk.

Loss Function

- It includes the following:
  - **Performance measurement**: Offer a clear metric to evaluate a model's performance by quantifying the difference between predictions and actual results.
  - **Direction for improvement**: Guide model improvement by directing the algorithm to adjust parameters(weights) iteratively to reduce loss and improve predictions.
  - **Balancing bias and variance**: Helps balance model bias (oversimplification) and variance (overfitting), essential for the model's generalization to new data.
  - **Influencing model behavior**: Can affect the model's behavior, such as being more robust against data outliers or prioritizing specific types of errors.

Types Of Loss Functions (For Regression)

1) Mean Square Error (MSE) / L2 Loss:

Quantifies the magnitude of the error between a machine learning algorithm prediction and an actual output by taking the average of the squared difference between the predictions and the target values.

$$MSE = (1/n) * \Sigma(y_i - \bar{y})^2$$

Where:

n is the number of samples in the dataset

y_i is the predicted value for the i-th sample

$\bar{y}$ is the target value for the i-th sample

When to use MSE: MSE is recommended for ML scenarios where it is conducive to the learning process to penalize significantly the presence of outliers.

Types Of Loss Functions (For Regression)

2) Mean Absolute Error (MAE) / L1 Loss:

Calculates the average absolute differences between predicted values from a machine learning model and the actual target values.

$$MAE = (1/n) * \Sigma|y_i - \bar{y}|$$

Where:

n is the number of samples in the dataset

y_i is the predicted value for the i-th sample

$\bar{y}$ is the target value for the i-th sample

When to use MAE: MAE measures the average absolute difference between the predicted and actual values. Unlike MSE, MAE does not square the differences, which makes it less sensitive to outliers.

Types Of Loss Functions (For Regression)

3) Huber Loss / Smooth Mean Absolute Error:

Combines MSE and MAE, applying MSE for small errors and MAE for large errors, making it robust to outliers. The hybrid nature of Huber Loss makes it less sensitive to outliers, just like MAE, but also penalizes minor errors within the data sample, similar to MSE. The Huber Loss function is also utilized in regression machine learning tasks. The mathematical equation for Huber Loss is as follows:

$$L(y,\hat{y}) = \begin{cases} \frac{1}{2}(y-\hat{y})^2 & \text{for } |y-\hat{y}| \le \delta \\ \delta(|y-\hat{y}| - \frac{1}{2}\delta) & \text{otherwise,} \end{cases}$$

Where:

L represents the Huber Loss function
δ is the delta parameter, which determines the threshold for switching between the quadratic and linear components of the loss function
y is the true value or target value

When to use MSE: MSE is recommended for ML scenarios where it is conducive to the learning process to penalize significantly the presence of outliers.

Types Of Loss Functions (For Classification)

Binary Cross Entropy Loss (BCE):

- BCE is a loss function measures the difference between probability distributions of binary variables. Binary classification refers to a task where the goal is to classify data into one of two possible classes or categories, often represented as 0 and 1, or "negative" and "positive".
- The binary cross-entropy loss function quantifies the difference between the predicted probability distribution and the actual binary labels of the data. It calculates the discrepancy between the predicted probabilities and the true labels, penalizing the model more for incorrect predictions that are further from the true labels.

$$L(y, f(x)) = -[y * \log(f(x)) + (1 - y) * \log(1 - f(x))]$$

Where:

L represents the Binary Cross-Entropy Loss function
y is the true binary label (0 or 1)
f(x) is the predicted probability of the positive class (between 0 and 1)

When to use MSE:

Binary cross-entropy (BCE), is a loss function commonly used in binary classification tasks, particularly in machine learning algorithms such as logistic regression and neural networks.

Types Of Loss Functions

Loss Function	Applicability to Classification	Applicability to Regression
Mean Square Error (MSE) / L2 Loss	✗	✓
Mean Absolute Error (MAE) / L1 Loss	✗	✓
Binary Cross-Entropy Loss / Log Loss	✓	✗
Categorical Cross-Entropy Loss	✓	✗
Hinge Loss	✓	✗
Huber Loss / Smooth Mean Absolute Error	✗	✓
Log Loss	✓	✗

Choosing The Right Loss Function

- An appropriately selected loss function acts as a guide, steering the learning algorithm towards accuracy and reliability, ensuring that it captures the underlying patterns in the data while avoiding **overfitting** or **underfitting**.
- The choice of loss functions depends on the task:
  - Use MSE or MAE for regression tasks.
  - Use Binary Cross-Entropy for binary classification and Categorical Cross-Entropy for multi-class classification.
  - For imbalanced datasets, consider Focal Loss to focus on hard-to-predict examples.
  - For tasks with outliers, Huber Loss is a robust choice.

Choosing The Right Loss Function

Factor	Description
Type of Learning Problem	Classification vs Regression; Binary vs Multiclass Classification.
Model Sensitivity to Outliers	Some loss functions are more sensitive to outliers (e.g., MSE), while others are more robust (e.g., MAE).
Desired Model Behavior	Influences how the model behaves, e.g., hinge loss in SVMs focuses on maximizing the margin.
Computational Efficiency	Some loss functions are more computationally intensive, impacting the choice based on available resources.
Convergence Properties	The smoothness and convexity of a loss function can affect the ease and speed of training.
The scale of the Task	For large-scale tasks, a loss function that scales well and can be efficiently optimized is crucial.

Q & A