Artificial Neural Networks

KOE073 MACHINE LEARNING

I INTRODUCTION – Well defined learning problems, Designing a Learning System, Issues in Machine Learning; THE CONCEPT LEARNING TASK - General-to-specific ordering of hypotheses, Find-S, List then eliminate algorithm, Candidate elimination algorithm, Inductive bias.

II DECISION TREE LEARNING - Decision tree learning algorithm-Inductive bias- Issues in Decision tree learning; ARTIFICIAL NEURALNETWORKS – Perceptrons, Gradient descent and the Delta rule, Adaline ,Multilayer networks, Derivation of backpropagation rule Backpropagation, Algorithm Convergence, Generalization.

III Evaluating Hypotheses: Estimating Hypotheses Accuracy, Basics of sampling Theory, Comparing Learning Algorithms; Bayesian Learning: Bayes theorem, Concept learning, Bayes Optimal Classifier, Naïve Bayes classifier, Bayesian belief networks, EM algorithm.

IV Computational Learning Theory: Sample Complexity for Finite Hypothesis spaces, Sample Complexity for Infinite Hypothesis spaces, The Mistake Bound Model of Learning; INSTANCE-BASED LEARNING – k-Nearest Neighbour Learning, Locally Weighted Regression, Radial basis function networks, Case-based learning.

V Genetic Algorithms: an illustrative example, Hypothesis space search, Genetic Programming, Models of Evolution and Learning; Learning first order rules- sequential covering algorithms-General to specific beam search-FOIL; REINFORCEMENT LEARNING - The Learning Task, Q learning.

Artificial Neural Networks

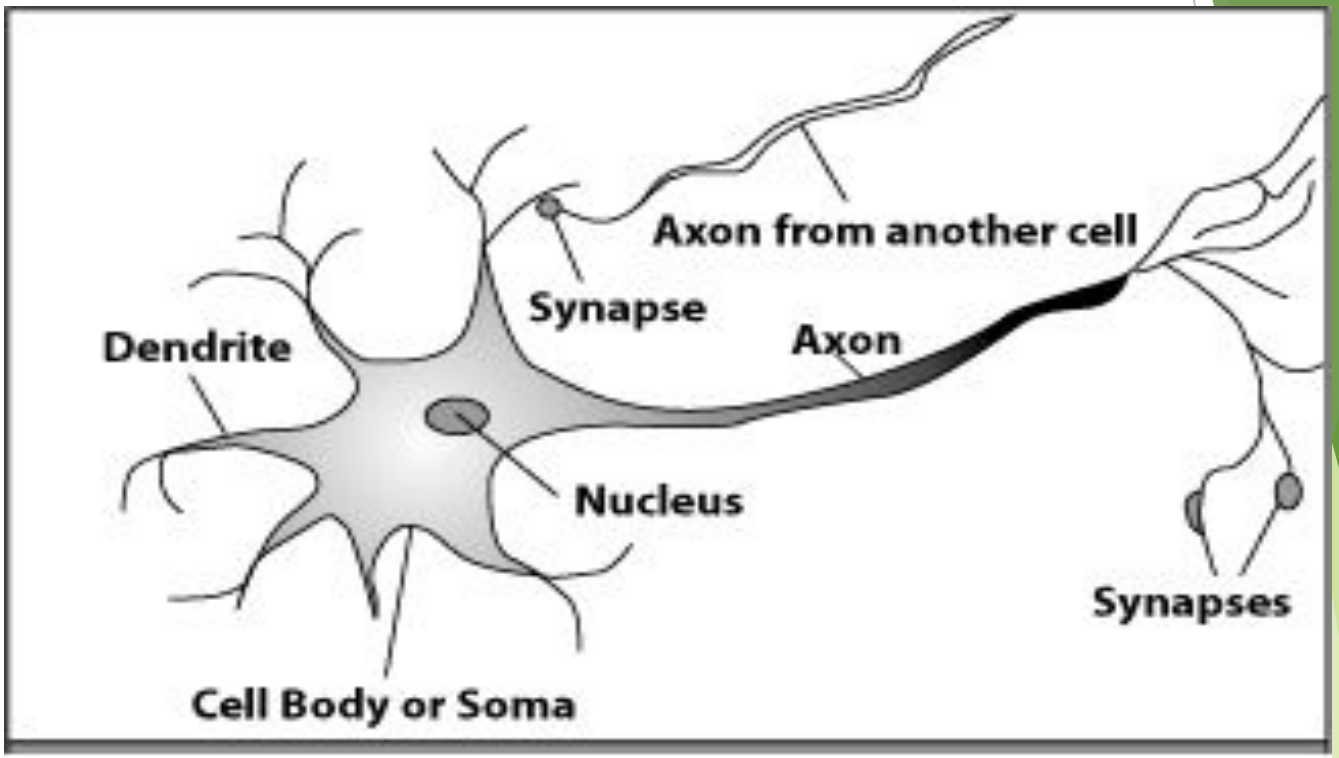
- Artificial Neural networks is an interconnected group of units which performs computation.
- Inspired by biological neural networks
- The "building blocks" of neural networks are the **neurons**.
- In technical systems, we also refer to them as **units** or **nodes**.
- Artificial Neural Networks are an imitation of the biological neural networks, but much simpler ones.
- The computing would have a lot to gain from neural networks. Their ability to learn by example makes them very flexible and powerful.

How do our brains work?

- The Brain is a massively parallel information processing system.
- Our brains is a huge network of processing elements. A typical brain contains a network of 10 billion neurons.

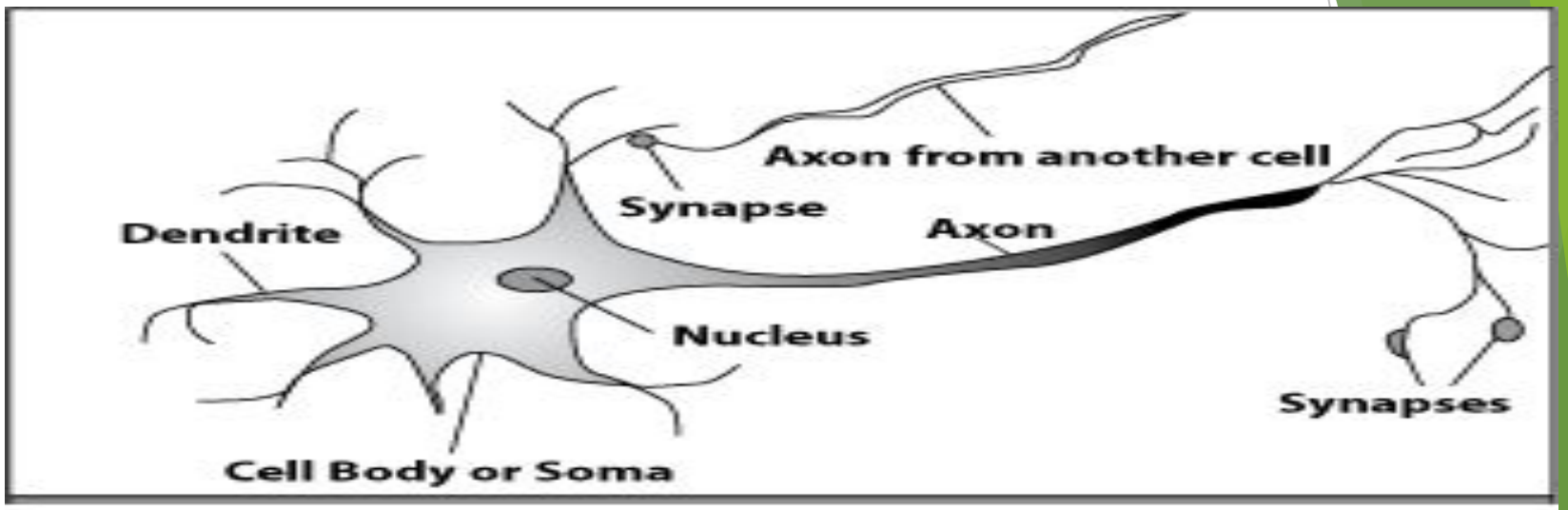
How do our brains work?

- A processing element



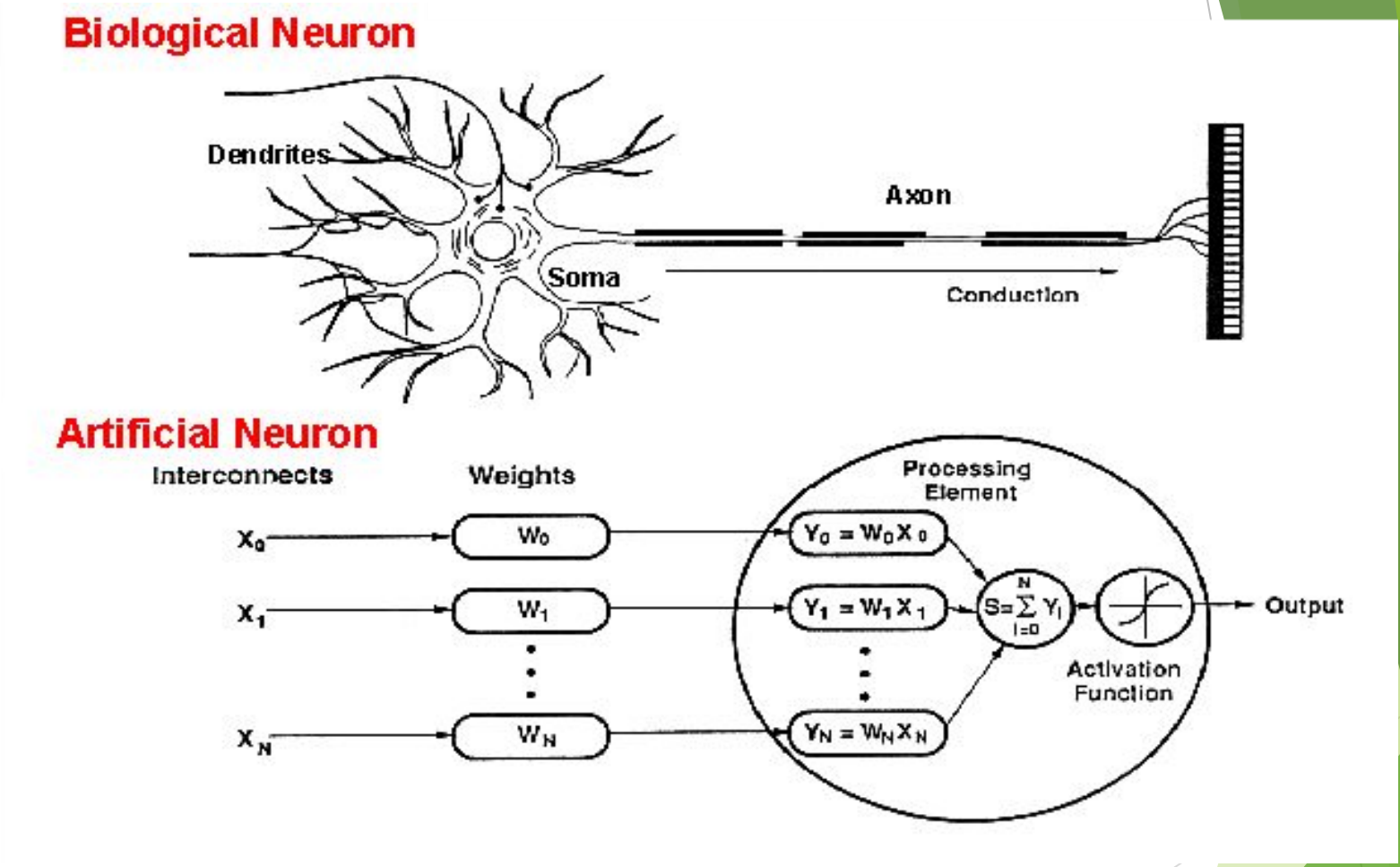
Dendrites: Input
Cell body: Processor
Synaptic: Link
Axon: Output

How do our brains work?



- A neuron is connected to other neurons through about *10,000 synapses*
- A neuron receives input from other neurons. Inputs are combined.
- Once input exceeds a critical level, the neuron discharges a spike – an electrical pulse that travels from the body, down the axon, to the next neuron(s)
- The axon endings almost touch the dendrites or cell body of the next neuron.
- Transmission of an electrical signal from one neuron to the next is effected by neurotransmitters.
- Neurotransmitters are chemicals which are released from the first neuron and which bind to the Second.
- This link is called a synapse. The strength of the signal that reaches the next neuron depends on factors such as the amount of neurotransmitter available.

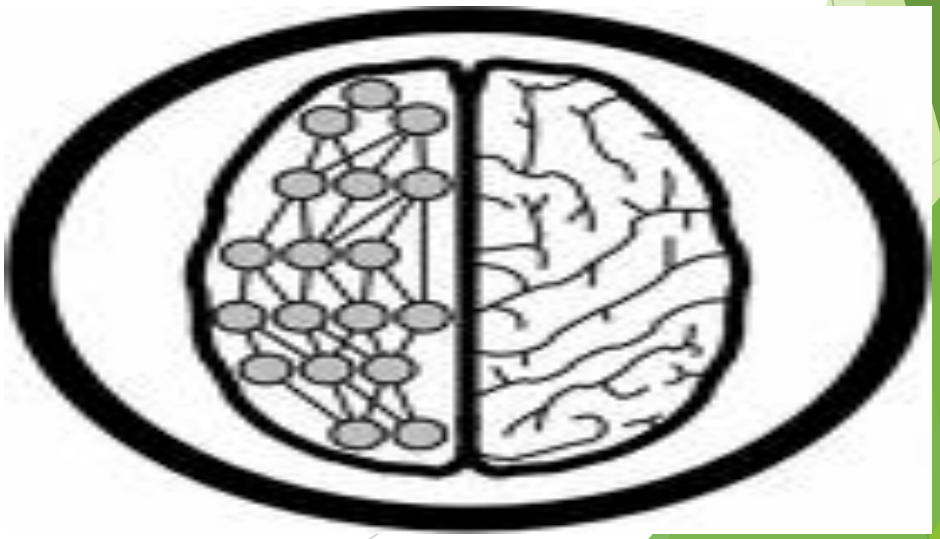
How do ANNs work?



An artificial neuron is an imitation of a human neuron

ANNs

- An artificial neural network (ANN) is either a hardware implementation or a computer program which strives to simulate the information processing capabilities of its biological exemplar. ANNs are typically composed of a great number of interconnected artificial neurons. The artificial neurons are simplified models of their biological counterparts.
- ANN is a technique for solving problems by constructing software that works like our brains.



How do ANNs work?

- Now, let us have a look at the model of an artificial neuron. I

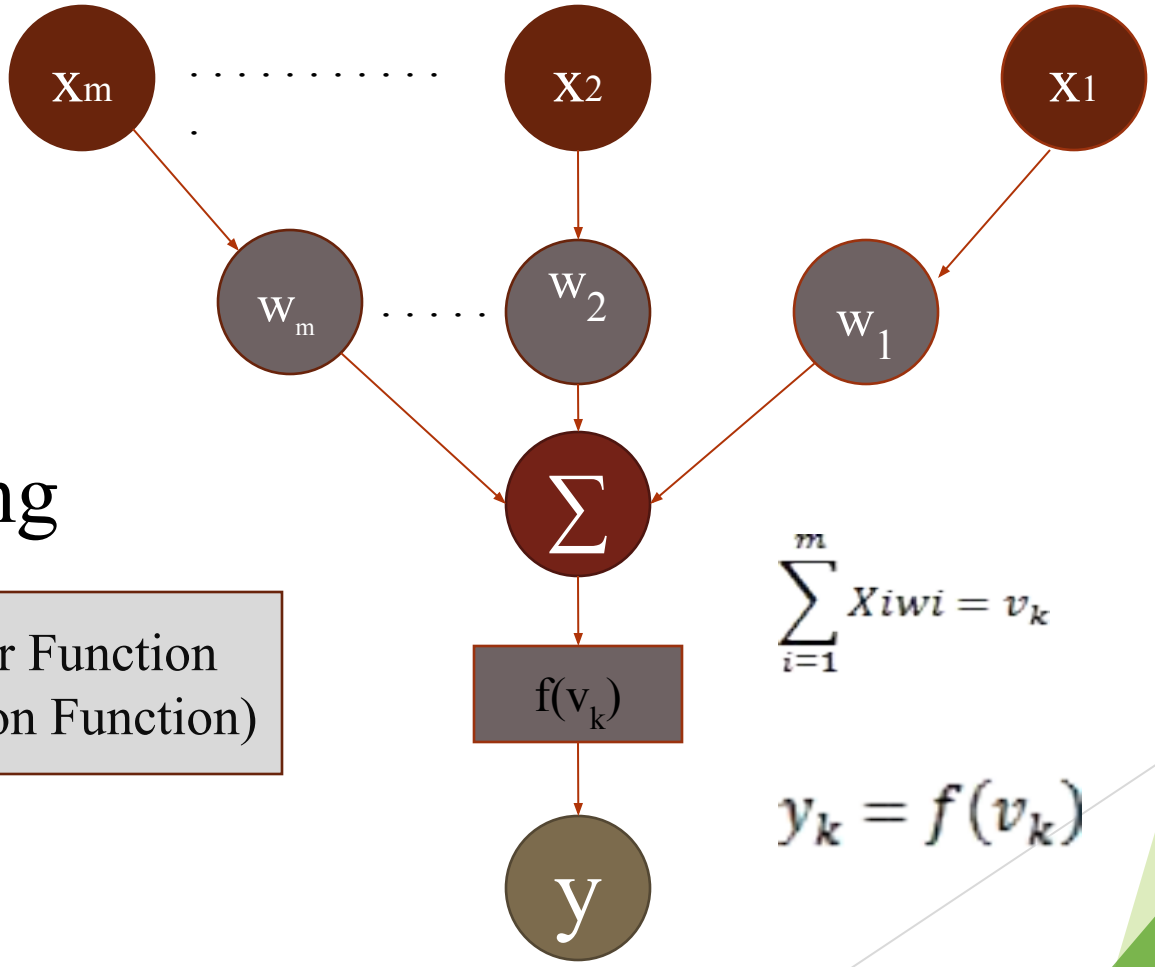
Input

weights

Processing

Transfer Function
(Activation Function)

Output

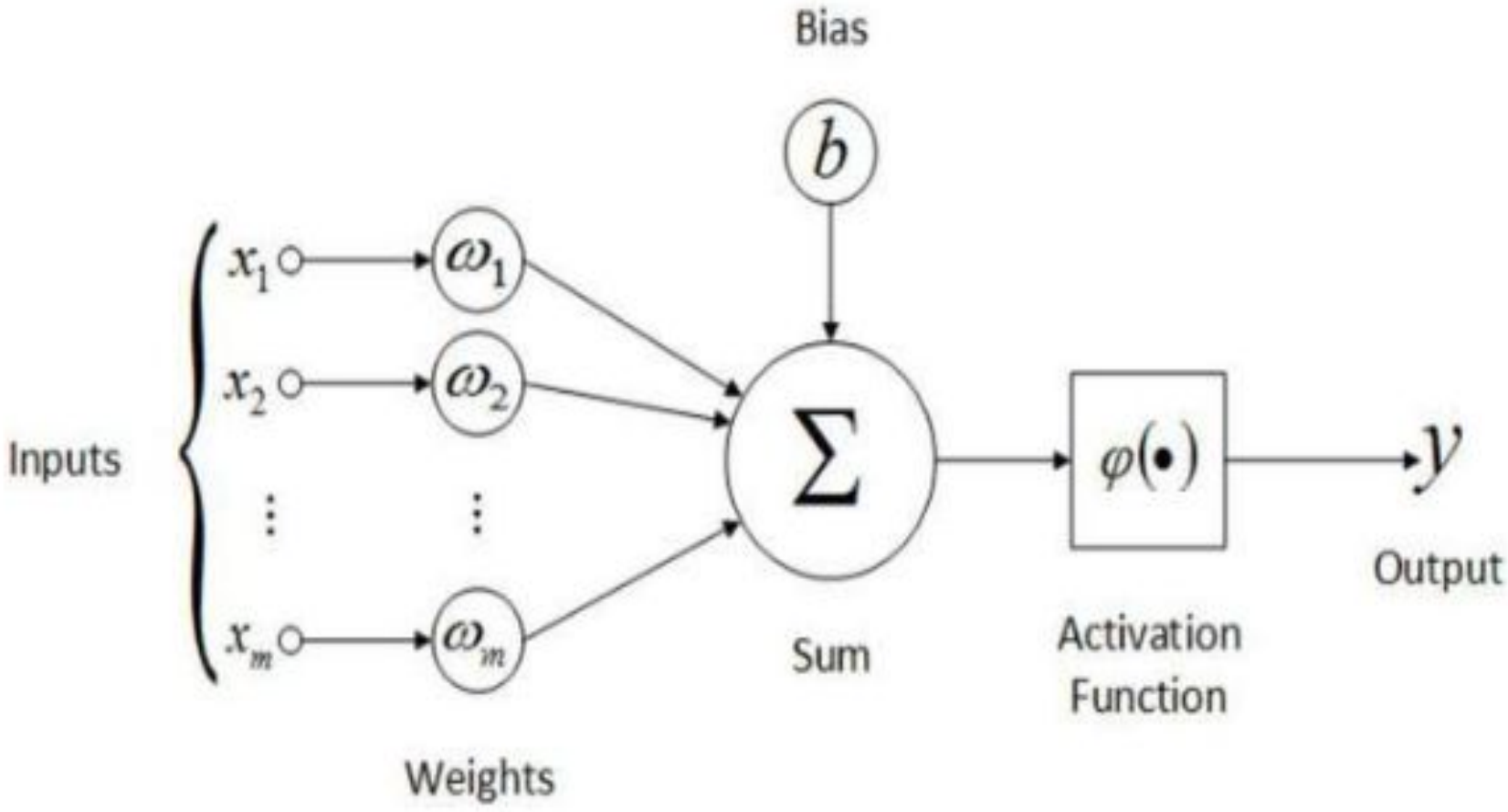


$$\sum_{i=1}^{m} Xiwi = v_k$$

$$y_k = f(v_k)$$

Perceptron

Perceptron is a simplest possible neural network.
It is a linear machine learning algorithm used for supervised learning for binary classifiers. It learns the weights in order to draw a linear decision boundary



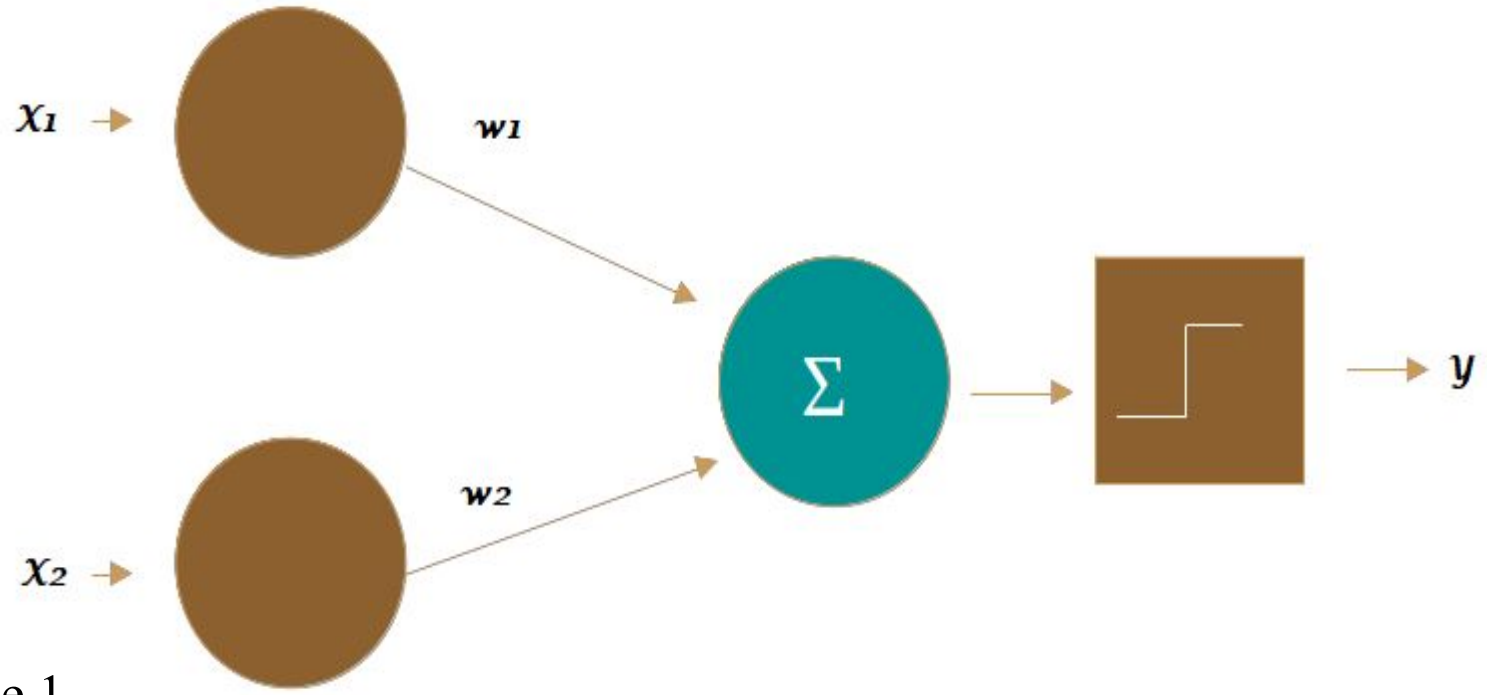
Characteristics of Perceptron Model

1. Perceptron is a machine learning algorithm for supervised learning for binary classifiers.
2. In Perceptron, the weight coefficient is automatically learned.
3. Weights multiplied with input features, decides whether the neuron is fired or not.
4. The activation function applies rule to check the weighted sum is compared with the preset threshold.
5. If the added sum of all input values is more than the threshold value, it must have an output signal; otherwise, no output will be shown.
6. The linear decision boundary is drawn, enabling the distinction between the two linearly separable classes +1 and -1.

Implementation of AND gate with Perceptron

X1	X2	Y
0	0	0
0	1	0
1	0	0
1	1	1

Assume the weight values are w1=1, w2=1& threshold value=1.5



For case 1
y= w1*x1+w2*x2
=0*1+0*0
=0<1.5 so final output is 0

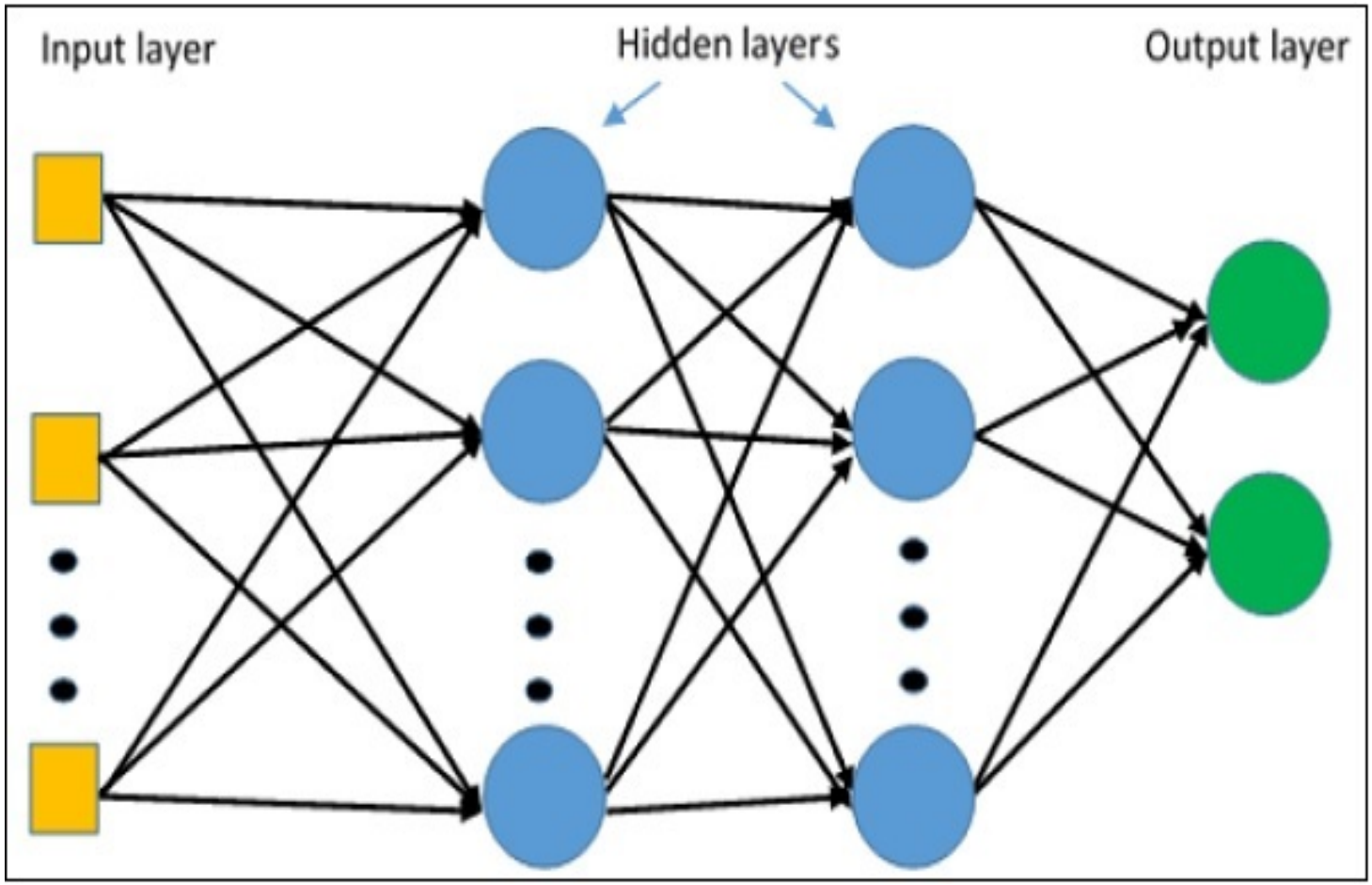
Case 4
y= w1*x1+w2*x2
=1*1+1*1
=2>1.50 so final output is 1

Multilayer Layer Perceptron (MLP)

MLP networks are usually used for supervised learning format.

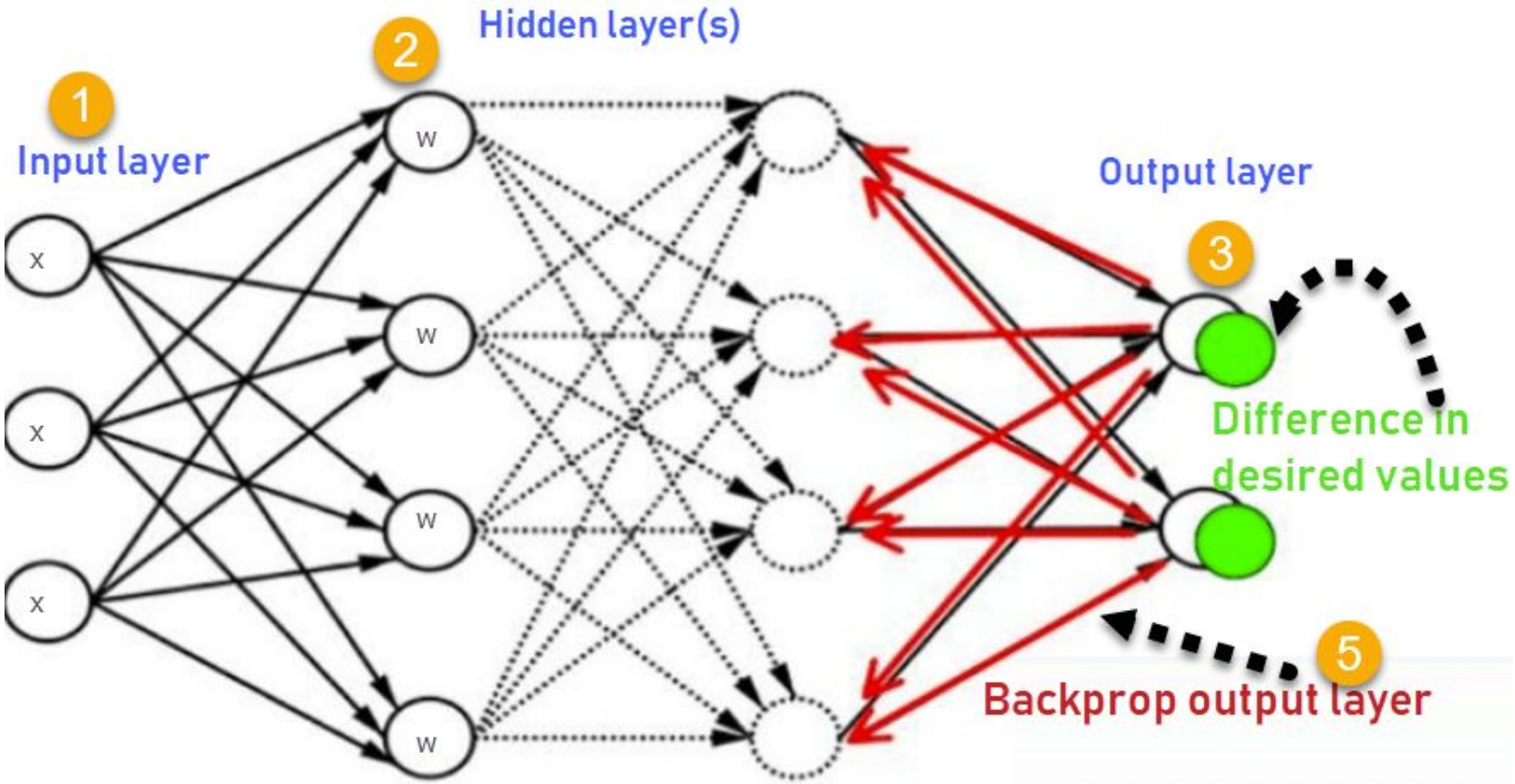
A typical learning algorithm for MLP networks is also called back propagation's algorithm

It train the model using Gradient Descent Rule..



Back Propagation Algorithm

- **Backpropagation** is the essence of neural network training. It is the method of fine-tuning the weights of a neural network based on the error rate obtained in the previous epoch (i.e., iteration). Proper tuning of the weights allows you to reduce error rates and make the model reliable by increasing its generalization.
- Backpropagation in neural network is a short form for “backward propagation of errors.”
- It is a standard method of training artificial neural networks.
- This method helps calculate the gradient of a loss


1
Input layer
2
Hidden layer(s)
Output layer
3
x
x
x
w
w
w
w
Difference in
desired values
5
Backprop output layer

*

1. Inputs X, arrive through the preconnected path
2. Input is modeled using real weights W. The weights are usually randomly selected.
3. Calculate the output for every neuron from the input layer, to the hidden layers, to the output layer.
4. Calculate the error in the outputs.

- Travel back from the output layer to the hidden layer to adjust the weights such that the error is decreased.
- Keep repeating the process until the desired output is achieved
- **Why We Need Backpropagation?**
- Most prominent advantages of Backpropagation are:
- Backpropagation is fast, simple and easy to program
- It has no parameters to tune apart from the numbers of input
- It is a flexible method as it does not require prior knowledge about the network
- It is a standard method that generally works well
- It does not need any special mention of the features of the function to be learned.

Back Propagation Algorithm

- Calculate output of each node at every layer
- Compare the final output with target value
- Calculate the total error.
- Update the weights by propagating the error in reverse till you reaches the input stage
  - Each weight changed by:

$$\Delta w_{ji} = \eta \delta_j o_i$$

$$\delta_j = o_j(1-o_j)(t_j - o_j) \qquad \text{if } j \text{ is an output unit}$$

$$\delta_j = o_j(1-o_j)\sum_k \delta_k w_{kj} \qquad \text{if } j \text{ is a hidden unit}$$

- Repeat from Step1 till the error is minimized

*

Thank You