Machine learning

Prepared by : Dr. Hanaa Bayomi
Updated By: Prof Abeer ElKorany

Lecture 9: Artificial Neural Network (ANN)

Topics

- **Biological Background**
- **Modelling a Neuron**
- **Processing of ANN**
- **Activation Function types**
- **Backpropagation**

Biological Inspirations

- Humans perform complex tasks like vision, motor control, or language understanding very well.

- One way to build intelligent machines is to try to imitate the (organizational principles of) human brain.

Human Brain

- The brain is a highly complex, non-linear, and parallel computer, composed of some 10^{11} neurons that are densely connected ($\sim 10^4$ connection per neuron). *We have just begun to understand how the brain works...*

- A neuron is much slower (10^{-3}sec) compared to a silicon logic gate (10^{-9}sec), however the massive interconnection between neurons make up for the comparably slow rate.
  - Complex perceptual decisions are arrived at quickly (within a few hundred milliseconds)

Biological Neuron

A biological neuron may have as many as 10,000 different inputs, and may send its output (the presence or absence of a short duration spike) to many other neurons. Neurons are wired up in a 3-dimensional pattern.

Dendrites: Nerve fibres carrying electrical signals to the cell.

Cell Body: Computes a non-linear function of its inputs .

Axon: Single long fiber that carries the electrical signal from the cell body to other neurons.

Synapse : The point of contact between the axon of one cell and the dendrite of another, regulating a chemical connection whose strength affects the input to the cell.

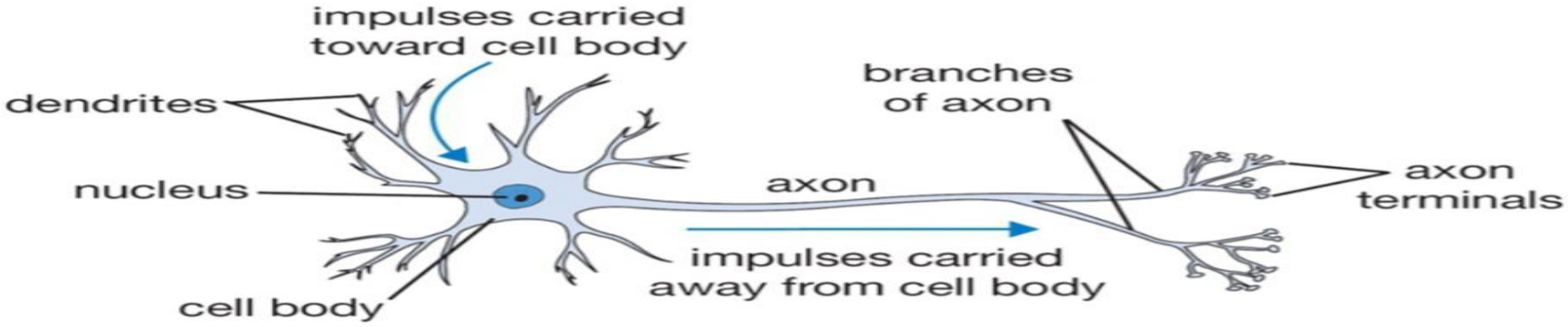


Figure: The basic computational unit of the brain: Neuron

Neural Network

- Neural networks are parallel computing devices, which is basically an attempt to make a computer model of the brain.
- The main objective is to develop a system to perform various computational tasks faster than the traditional systems.
- These tasks include pattern recognition and classification, approximation, optimization, and data clustering.

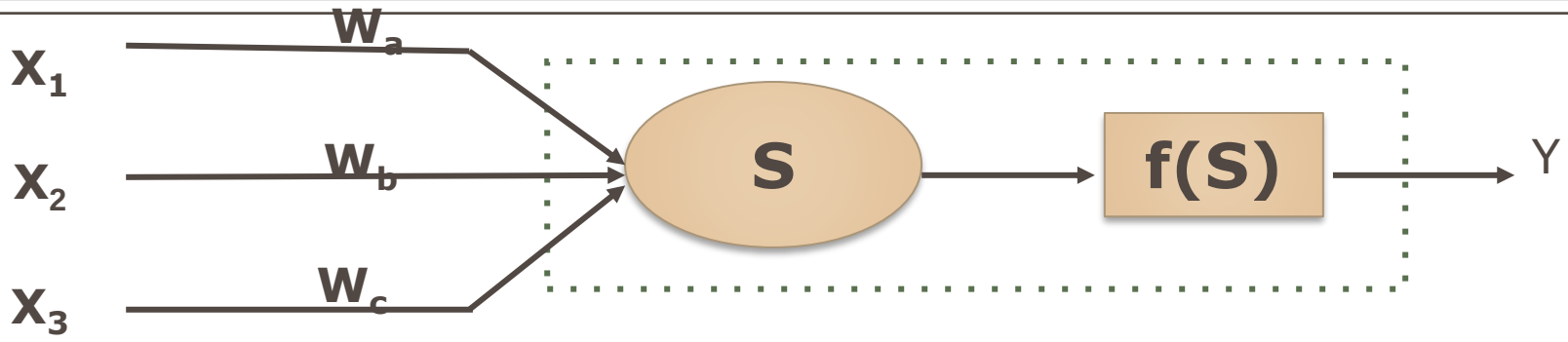
What is Artificial Neural Network?

- Computational model inspired by the human brain:
- Massive parallel, distributed system, made up of simple processing units(neurons).
- Synaptic connection strengths among neurons are used to store the acquired knowledge.
- Knowledge is acquired by the network from its environment through a learning process.

Properties of Artificial Neural Network

- **Learning from Experience:**
  - Labeled or unlabeled
- **Adaptivity:**
  - Changing the connection strengths to learn things.
- **Non-linearity:**
  - The non-linear activation functions are essential.
- **Fault tolerance:**
  - If one of the neurons or connections is damaged, the whole network still works quite well.

Model Of A Neuron



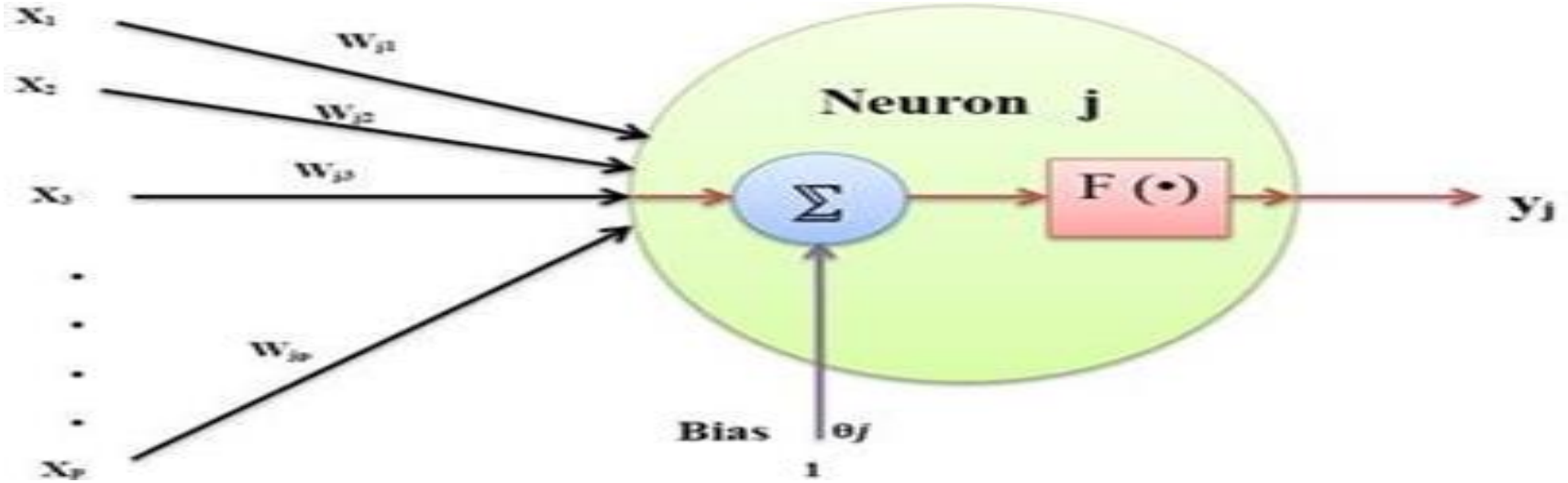
Input units (dendrite)

Connection weights (synapse)

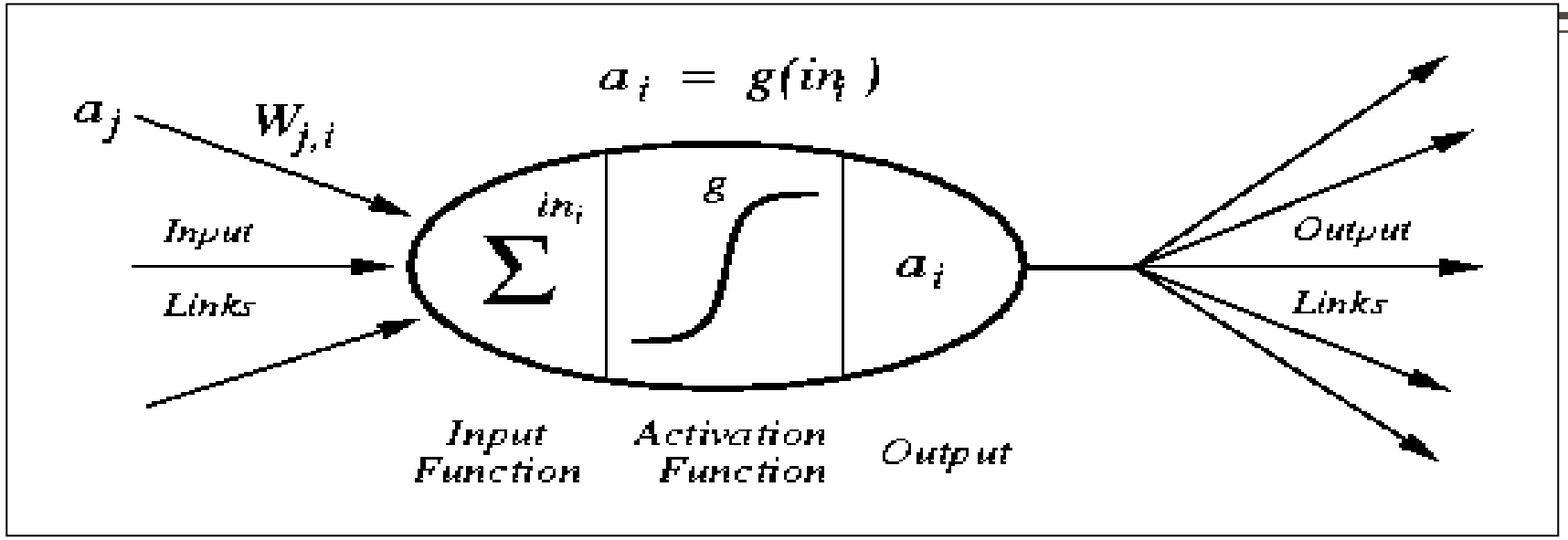
Summing function

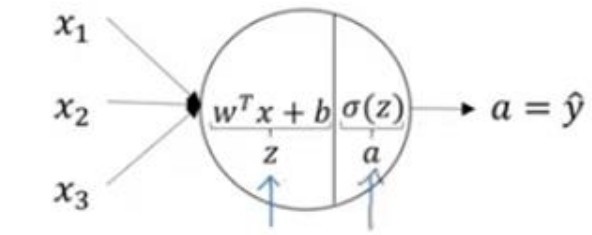
computation (soma)

(axon)



Modelling a Neuron





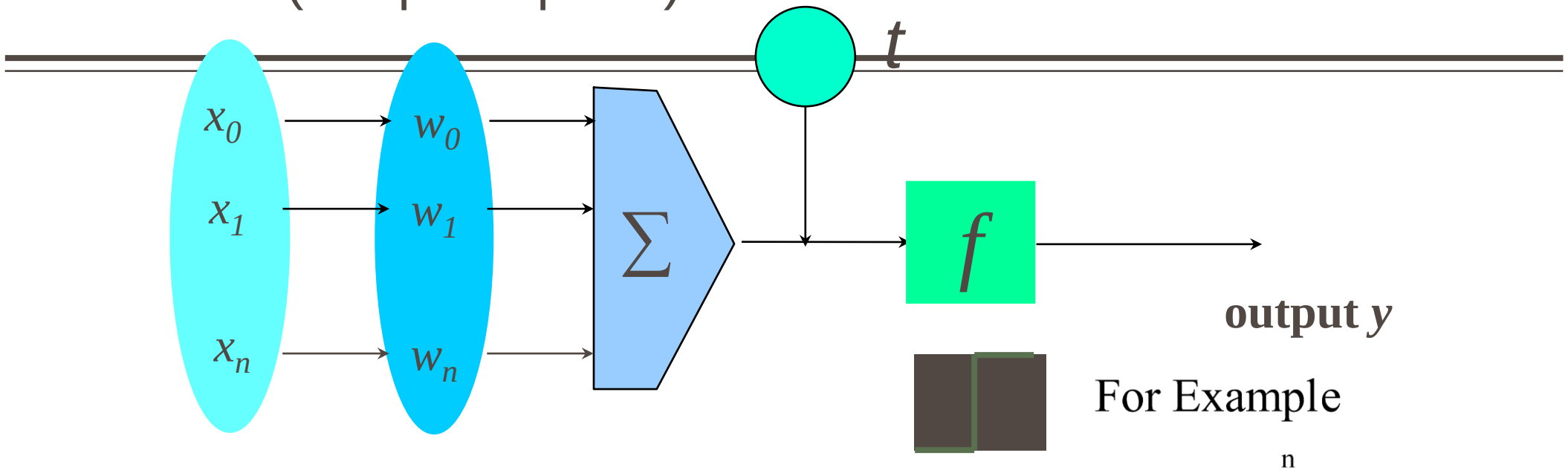
$z = w^T x + b$

$a = \sigma(z)$

$$in_i = \sum_j W_{j,i} a_j$$

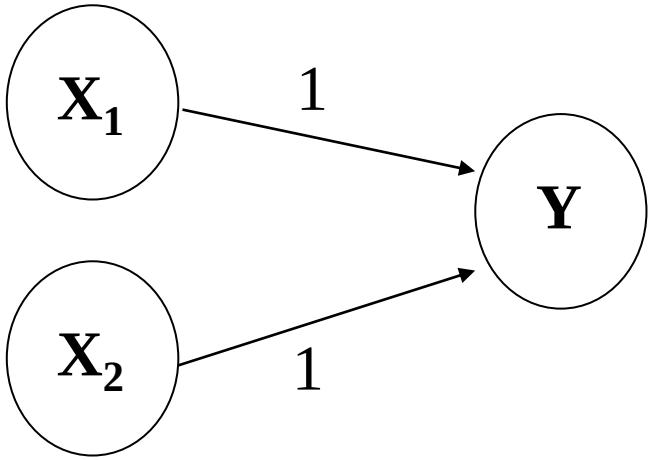
- a_j :Activation value of unit j
- $w_{j,I}$:Weight on the link from unit j to unit i
- in_I :Weighted sum of inputs to unit i
- a_I :Activation value of unit i
- g :Activation function

A Neuron (= a perceptron)



- The *n*-dimensional input vector **x** is mapped into variable y by means of the scalar product and a nonlinear function mapping

The First Neural Neural Networks

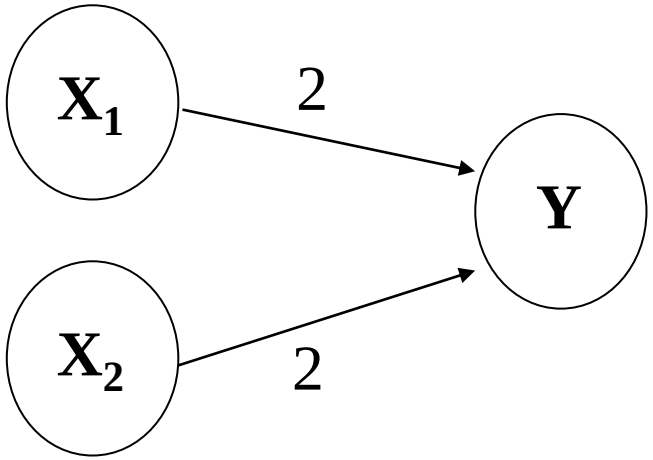


AND Function

Threshold(*Y*) = 2

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

The First Neural Neural Networks

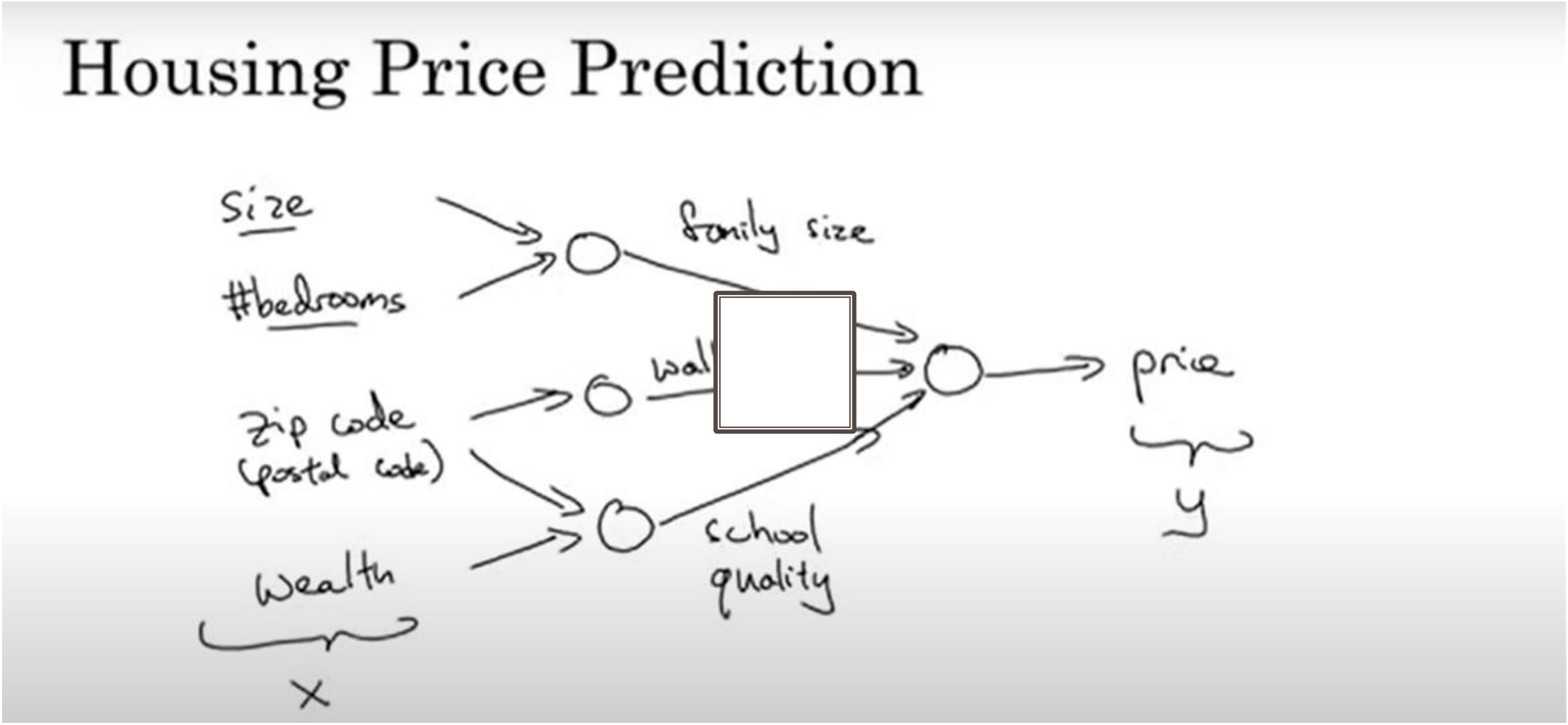


OR Function

Threshold(Y) = 2

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

Example


Housing Price Prediction
Size
#bedrooms
Zip code
(postal code)
Wealth
X
family size
walk
school
quality
price
y

Processing of ANN

- ANN depends upon the following three building blocks −
  - Network Topology
  - Adjustments of Weights or Learning
  - Activation functions

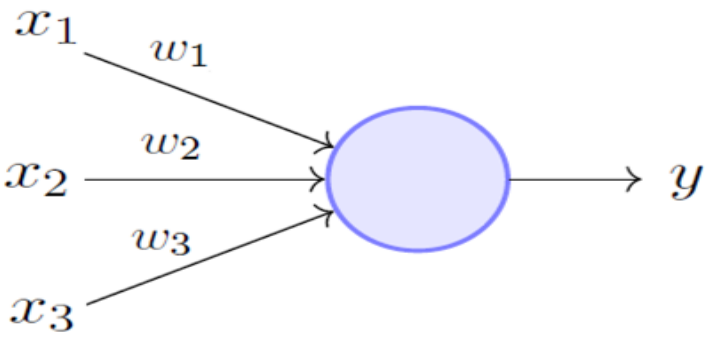
1) Network Topology

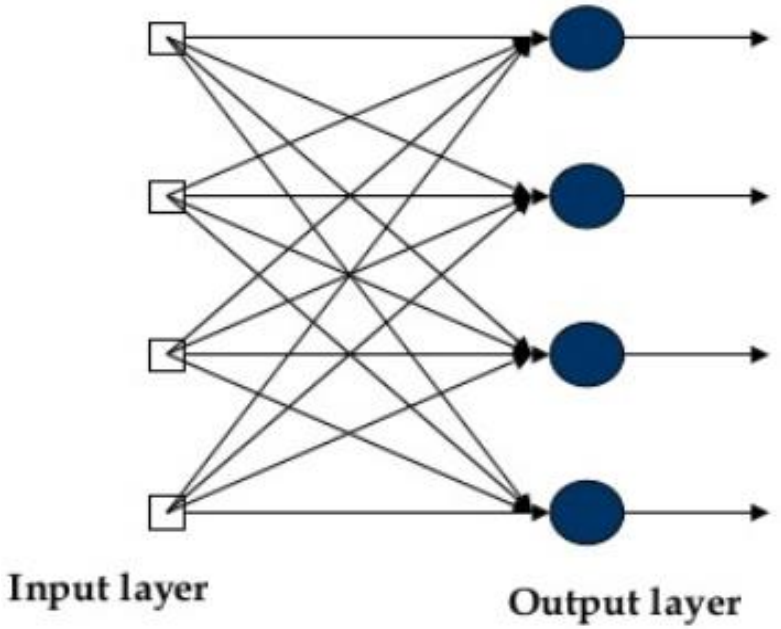
- It is the arrangement of a network along with its nodes and connecting lines. According to the topology, ANN can be classified as the following kinds –
  - Single Layer Network
  - Multi Layer Network

Single Layer Network (perceptron)

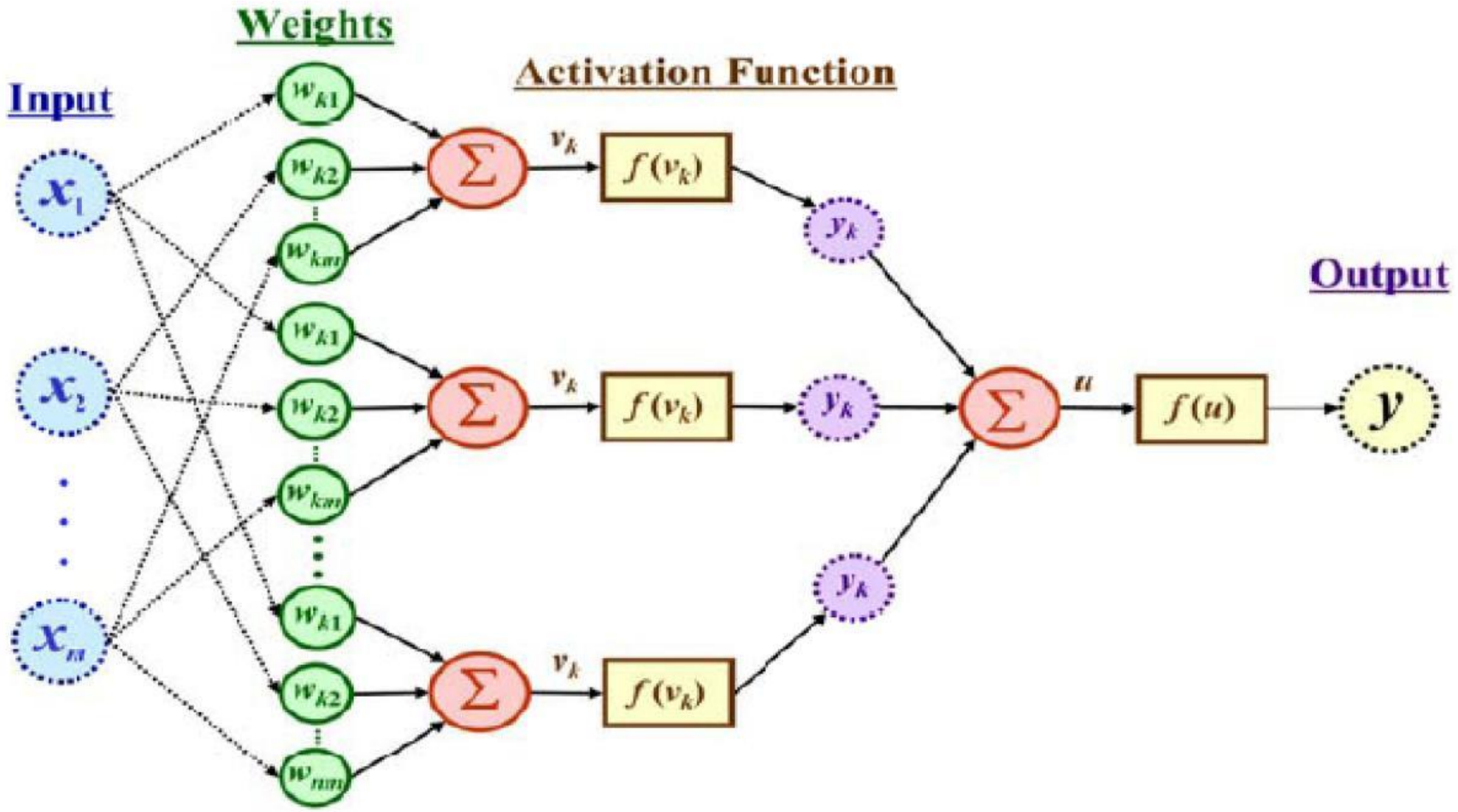
- All the nodes in a layer are connected with the nodes of the previous layers.
- The connection has different weights upon them.
- There is no feedback loop means the signal can only flow in one direction, from input to output.
- The concept of ANN having only one weighted layer. In other words, we can say the input layer is fully connected to the output layer.

=



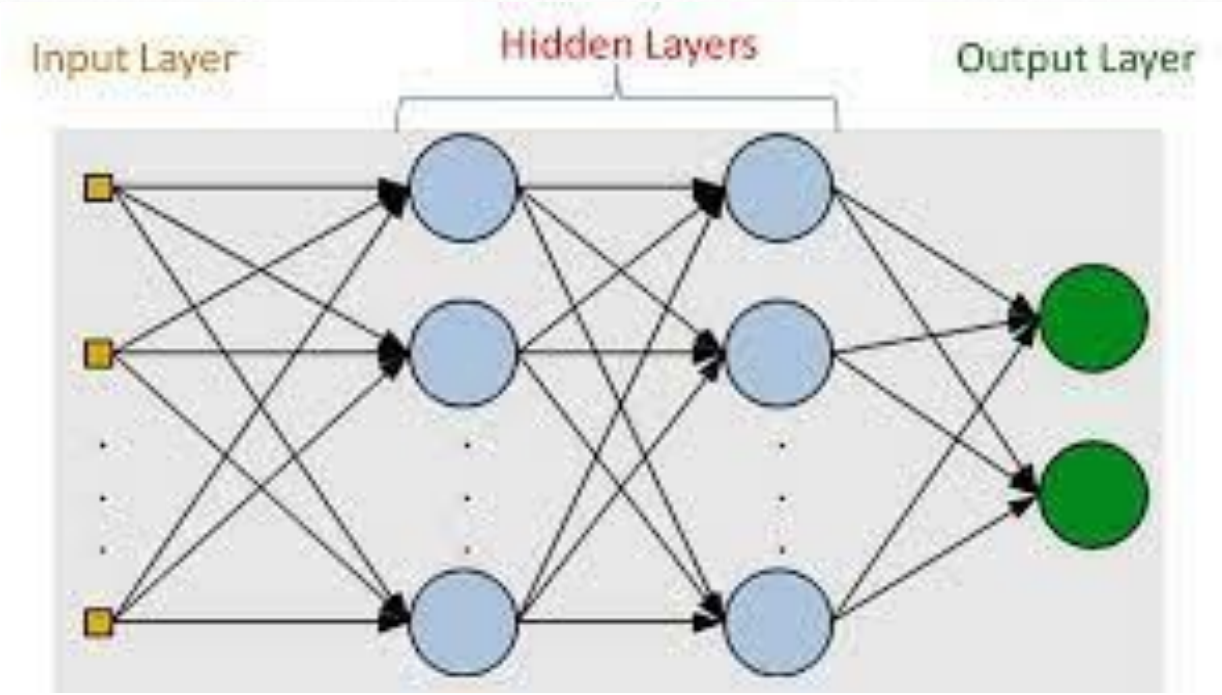


The output is a function of the input, that is affected by the weights, and the transfer functions



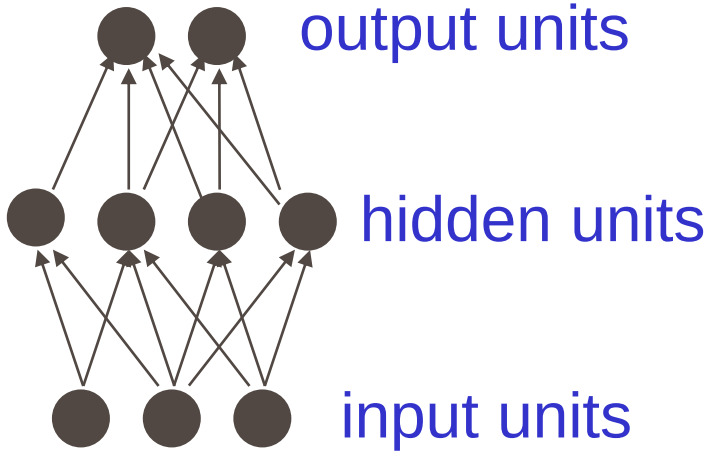
Multi Layer Network

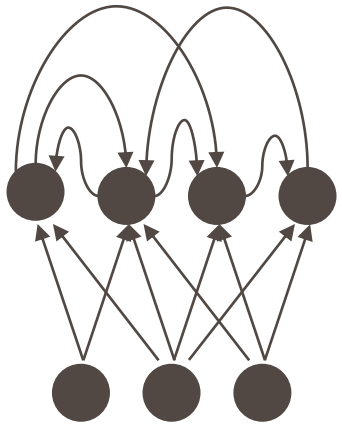
- The concept of ANN having more than one weighted layer. As this network has one or more layers between the input and the output layer, it is called hidden layers.



Types of connectivity

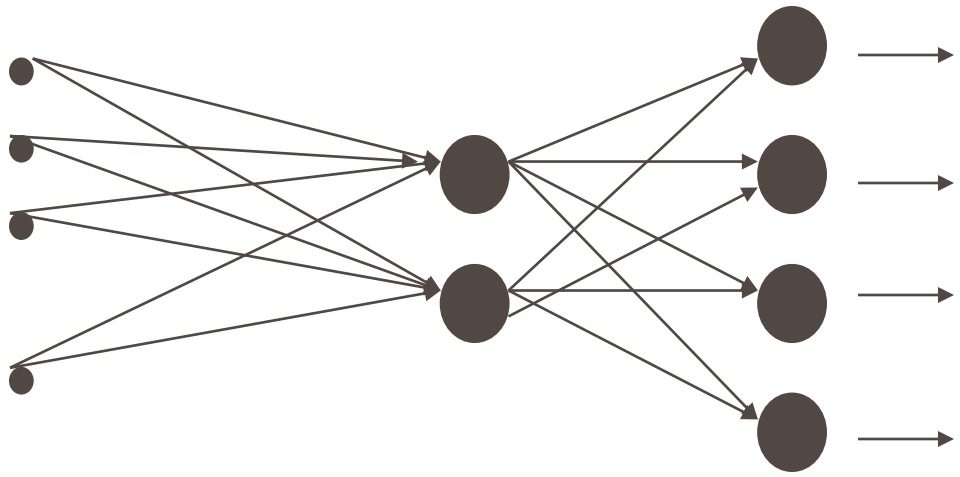
- Feedforward networks
  - These compute a series of transformations
  - Typically, the first layer is the input and the last layer is the output.
- Recurrent networks
  - These have directed cycles in their connection graph. They can have complicated dynamics.
  - More biologically realistic.





Different Network Topologies

- Multi-layer feed-forward networks
  - One or more hidden layers. Input projects only from previous layers onto a layer.



Input layer

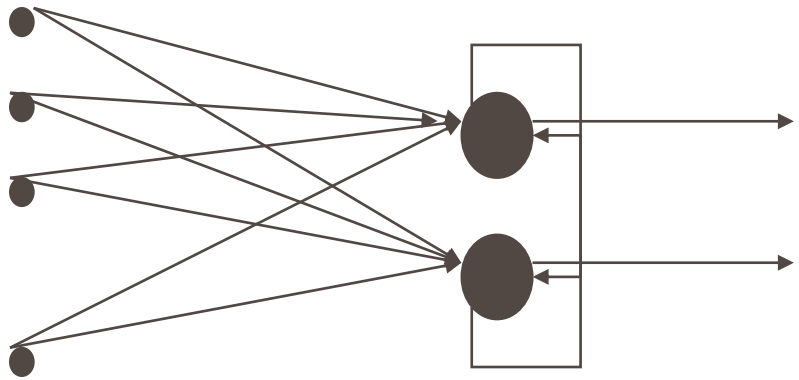
Hidden layer

Output layer

2-layer or
1-hidden layer
fully connected
network

Different Network Topologies

- Recurrent networks
  - A network with feedback, where some of its inputs are connected to some of its outputs (discrete time).



Recurrent network

Input layer

Output layer

2) Adjustments of Weights or Learning

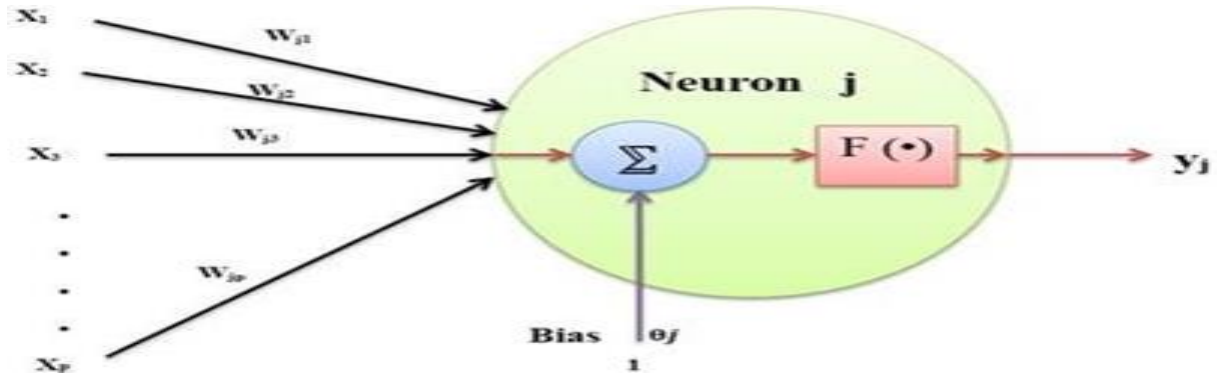
- It is the method of modifying the weights of connections between the neurons of a specified network.
- Learning in ANN can be classified into three categories:
  - Supervised learning
  - Unsupervised learning
  - Reinforcement learning

3) Activation Functions

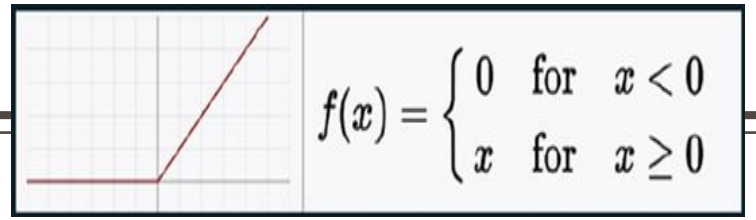
- It is the extra force or effort applied over the input to obtain an exact output. In ANN, we can also apply activation functions over the input to get the exact output.

Why do we need Activation Functions?

- Neural Network without activation function simply be a linear regression model
- a linear equation is polynomial of one degree.
- We want a neural network to not just learn and compute a linear function but something more complicated than that.
- Complicated kind of data such as images, videos, audio, speech etc.

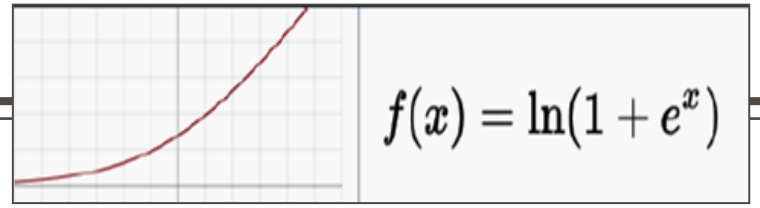


Activation Function types



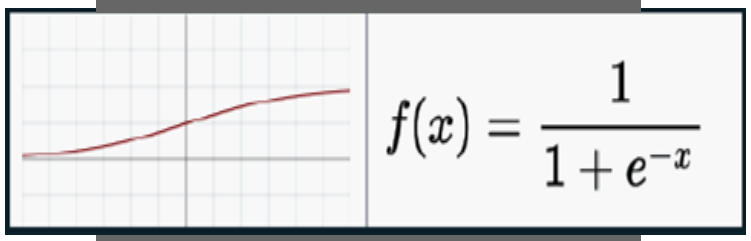
$$f(x) = \begin{cases} 0 & \text{for} \quad x < 0 \\ x & \text{for} \quad x \geq 0 \end{cases}$$

ReLU



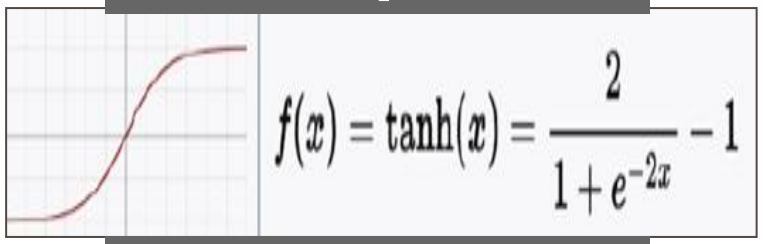
$$f(x) = \ln(1 + e^{x})$$

Softplus



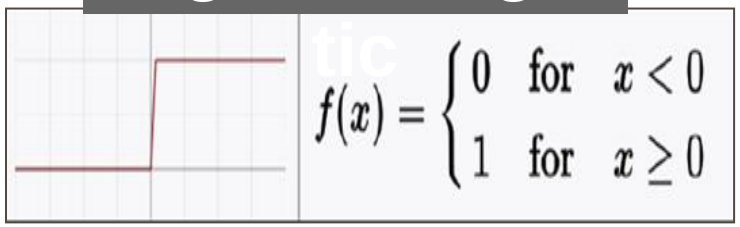
$$f(x) = \frac{1}{1 + e^{-x}}$$

Sigmoid/logis



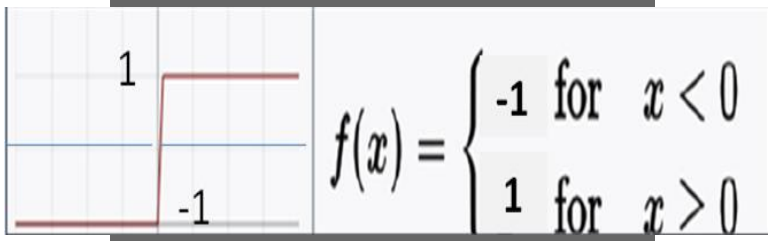
$$f(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1$$

Tanh



$$f(x) = \begin{cases} 0 & \text{for} \quad x < 0 \\ 1 & \text{for} \quad x \geq 0 \end{cases}$$

Binary



$$f(x) = \begin{cases} -1 & \text{for} \quad x < 0 \\ 1 & \text{for} \quad x > 0 \end{cases}$$

Signum

$$f_i(\vec{x}) = \frac{e^{x_i}}{\sum_{j=1}^{J} e^{x_j}} \quad \text{for } i = 1, \ldots, J$$

Softmax

How to choose the Activation Function

- You need to **match your activation function for your output layer based on the type of prediction problem** that you are solving—specifically, the type of predicted variable.

- you can begin with using the ReLU activation function and then move over to other activation functions if ReLU doesn't provide optimum results.

➢ And here are a few other guidelines to help you out.

1. **ReLU activation function** **should only be used** in **the hidden layers**.

1.Sigmoid/Logistic and Tanh functions should **not be used in hidden layers** as they make the model more sensitive to problems during training (due to vanishing gradients).

How to choose the Activation Function

Few rules for choosing the activation function for your **output layer** based on the type of prediction problem that you are solving:

- **Regression** - Linear Activation Function
- **Binary Classification** - Sigmoid/Logistic Activation Function
- **Multiclass Classification** - Softmax
- **Convolutional Neural Network (CNN)**: ReLU activation function.
- **Recurrent Neural Network**: Tanh and/or Sigmoid activation function.

Architecture of a typical artificial neural network

XOR

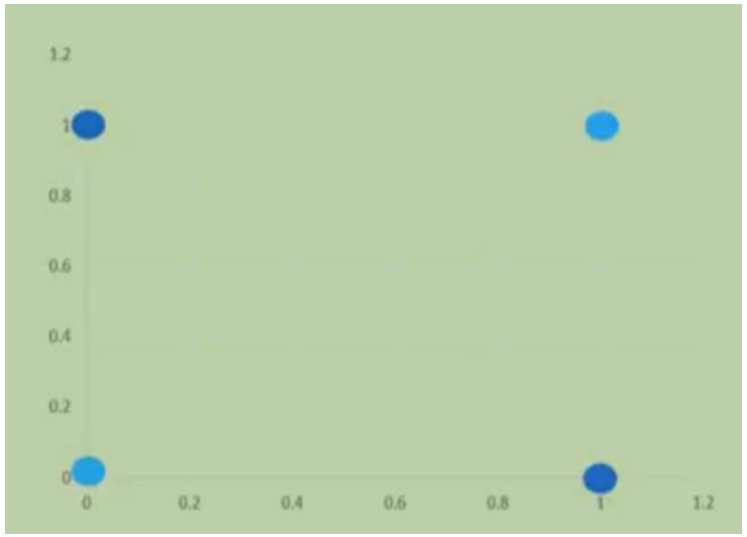
	A	B
1	1	0
	0	1
0	0	0
	1	1

w_{ij}^{l} — Layer $l-1$ to Layer l

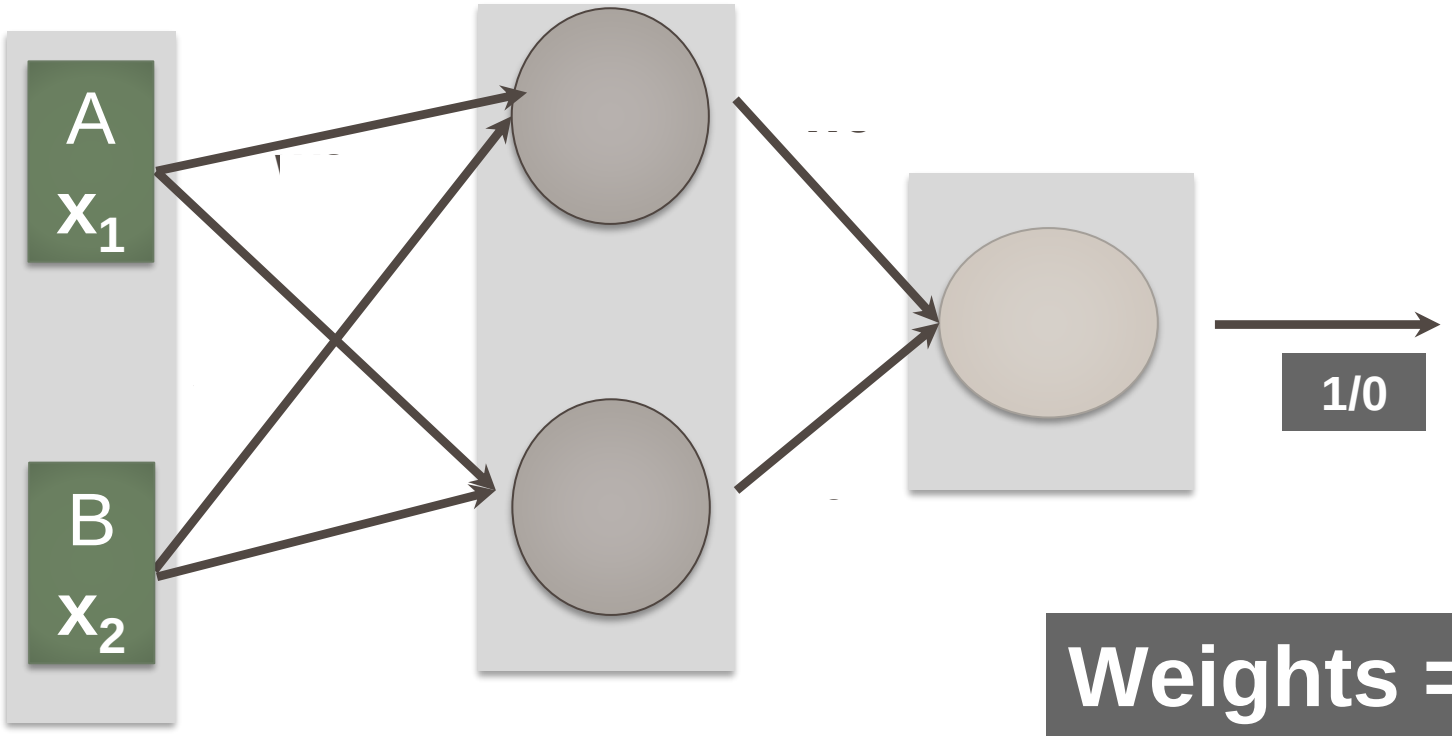
from neuron j (Layer $l-1$)
to neuron i (Layer l)

W_i	
W_2	W_{21}^1
W_3	W_{12}^1
W_4	W_{22}^1
W_5	W_{11}^2
W_6	W_{12}^2

	A	B
1	1	0
	0	1
0	0	0
	1	1



Input layer | Hidden layer | Output layer

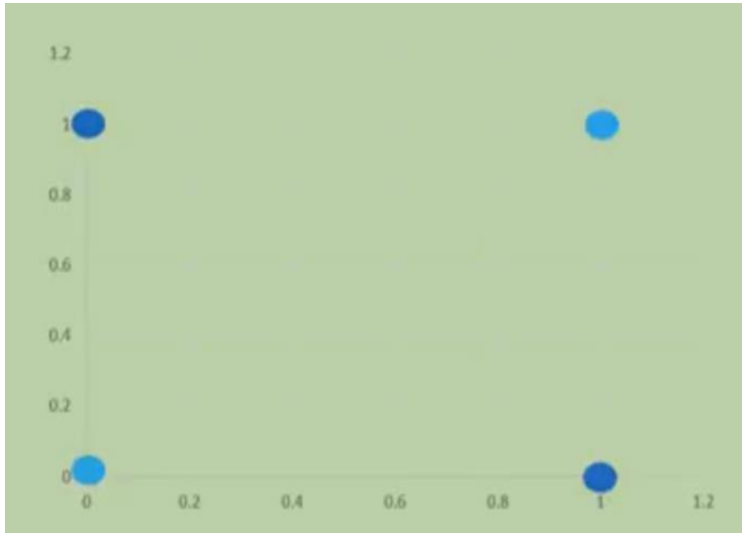


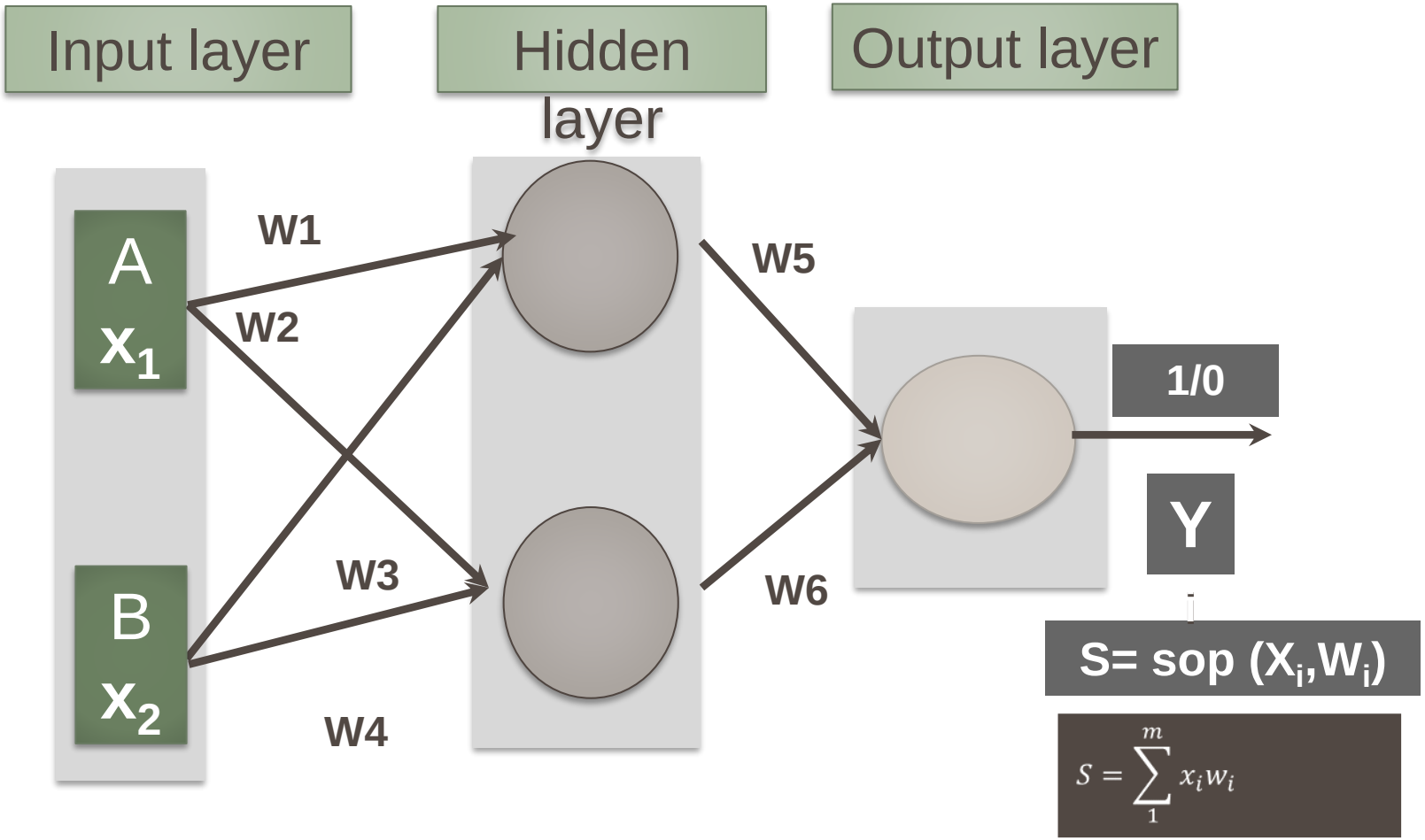
1/0

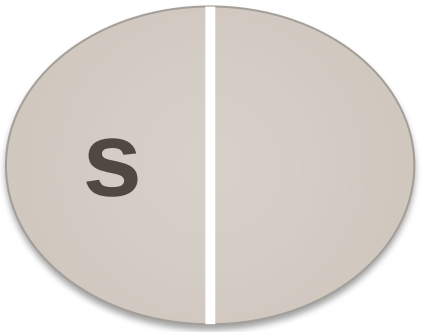
Weights =W_i

Activation function input

	A	B
1	1	0
	0	1
0	0	0
	1	1

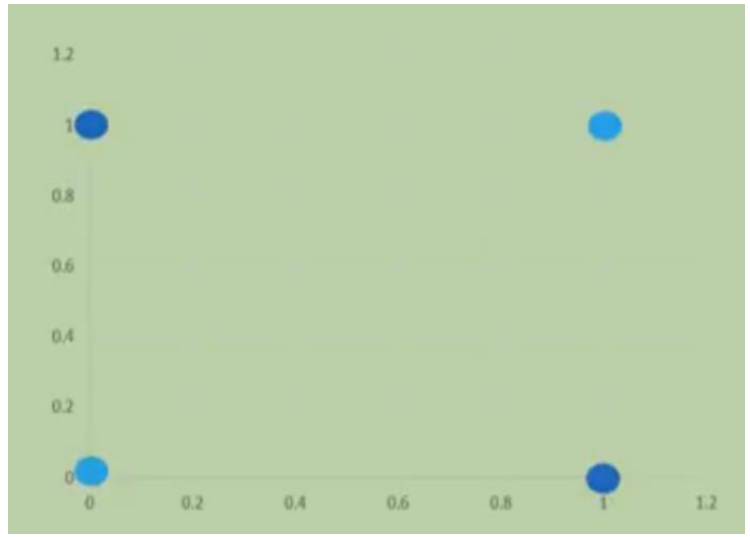


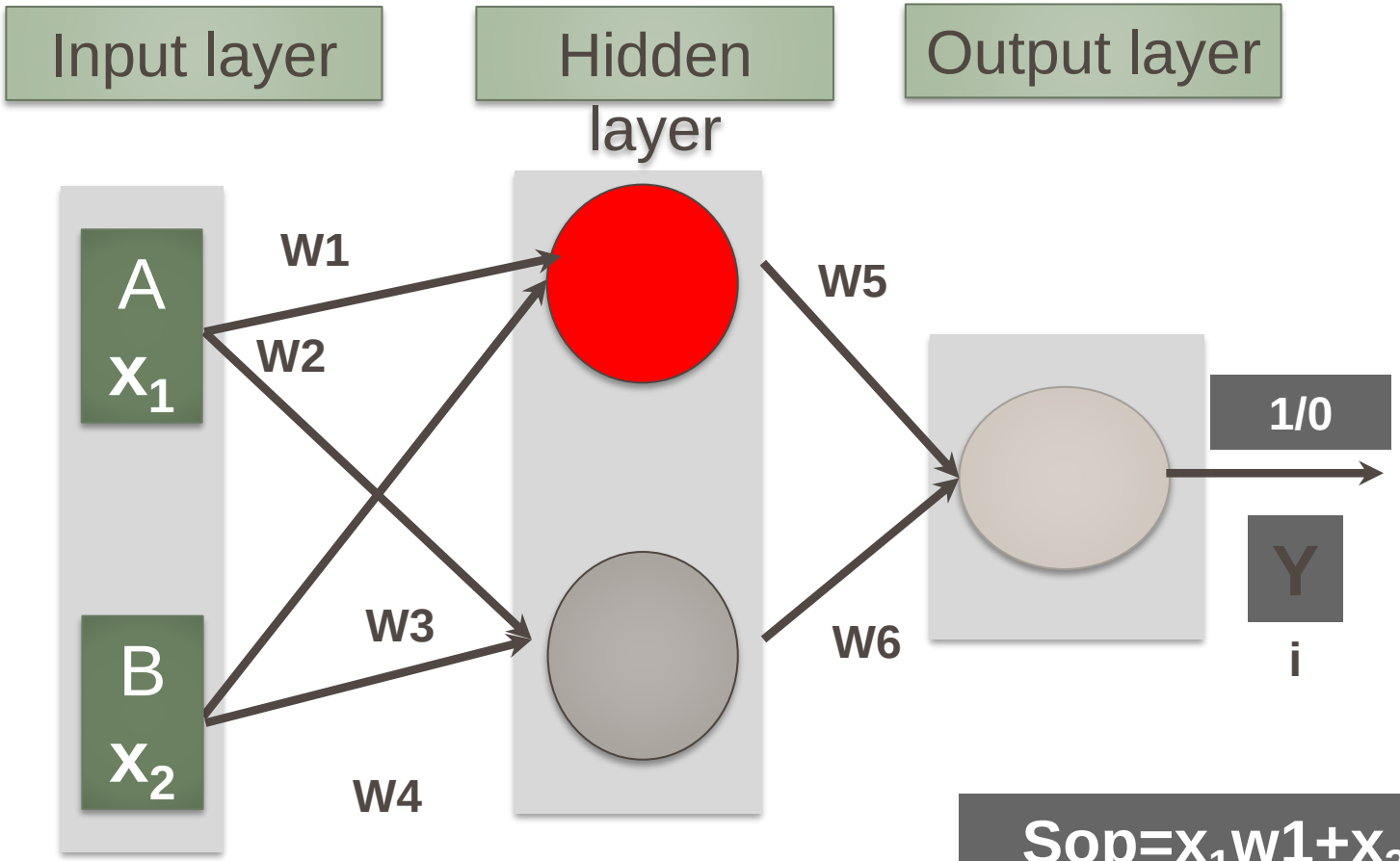




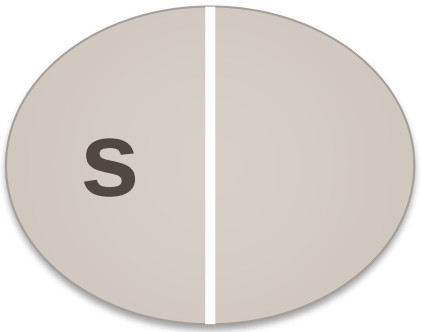
Activation function input

	A	B
1	1	0
	0	1
0	0	0
	1	1



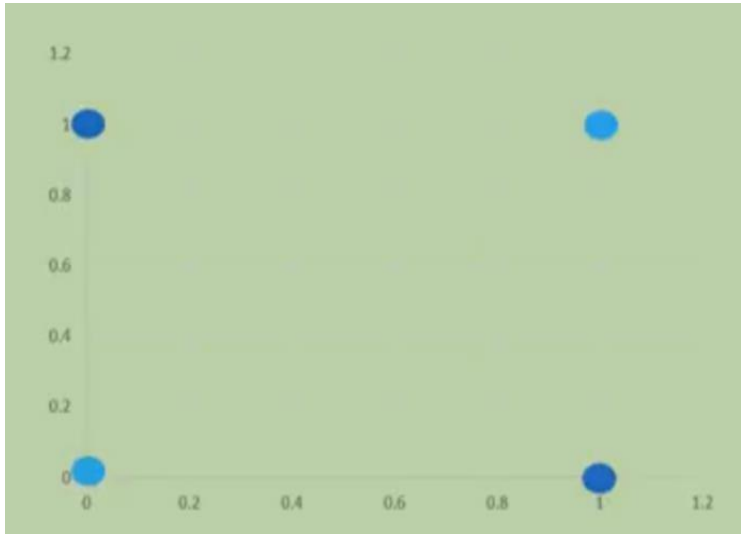


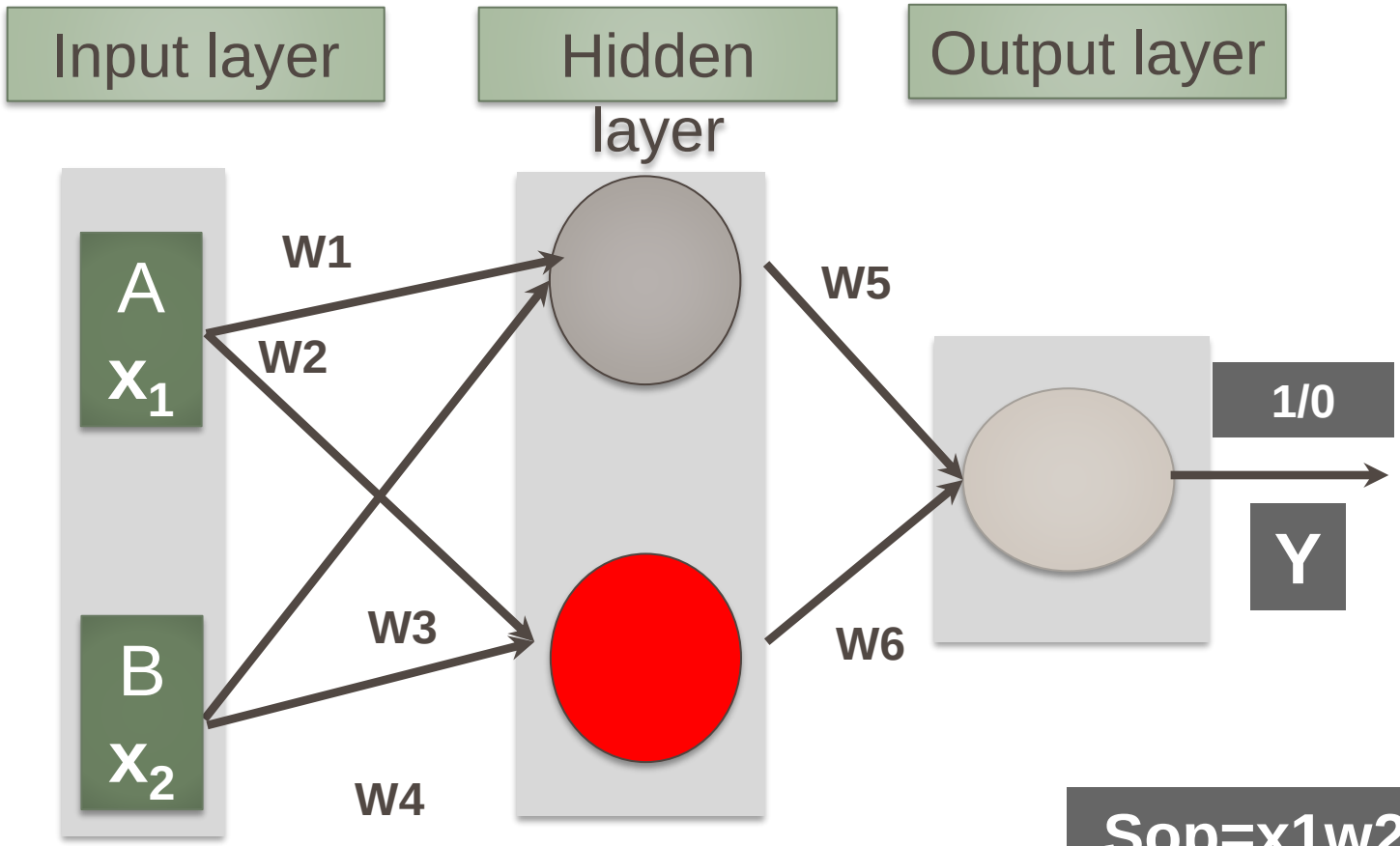
$Sop=x_1w1+x_2w3$

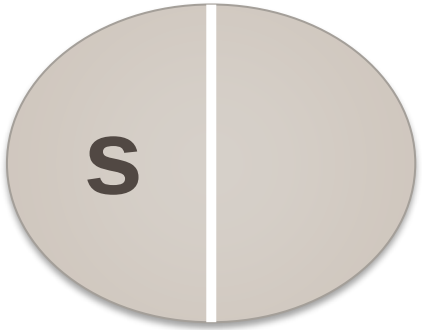


Activation function input

	A	B
1	1	0
	0	1
0	0	0
	1	1



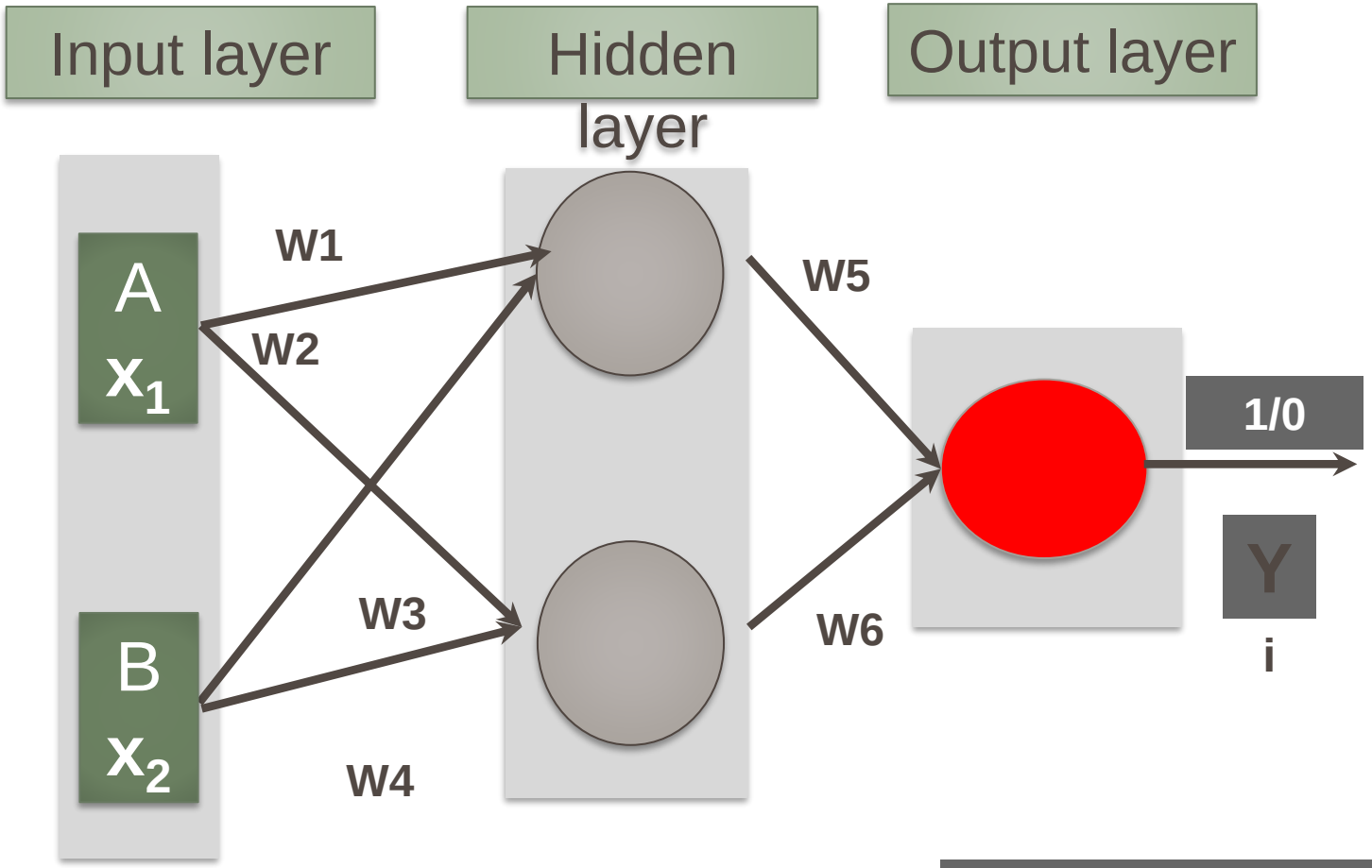


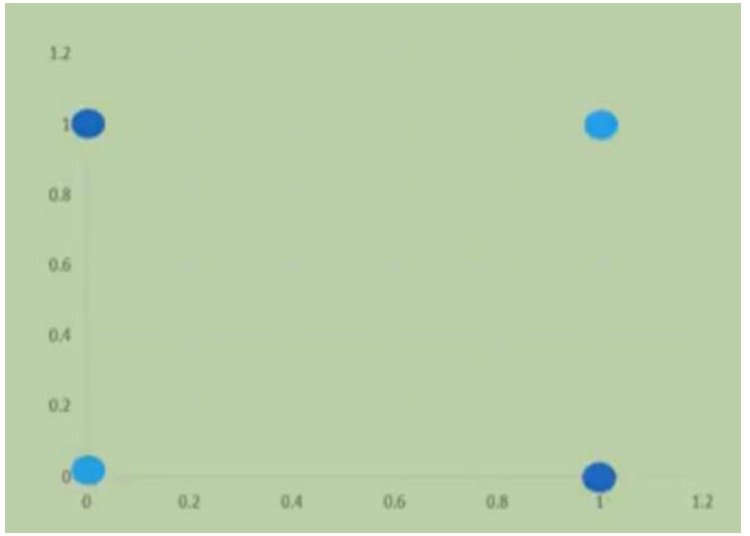


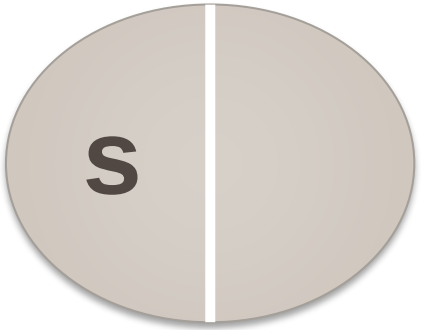
Sop=x1w2+x2

Activation function input

	A	B
1	1	0
	0	1
0	0	0
	1	1



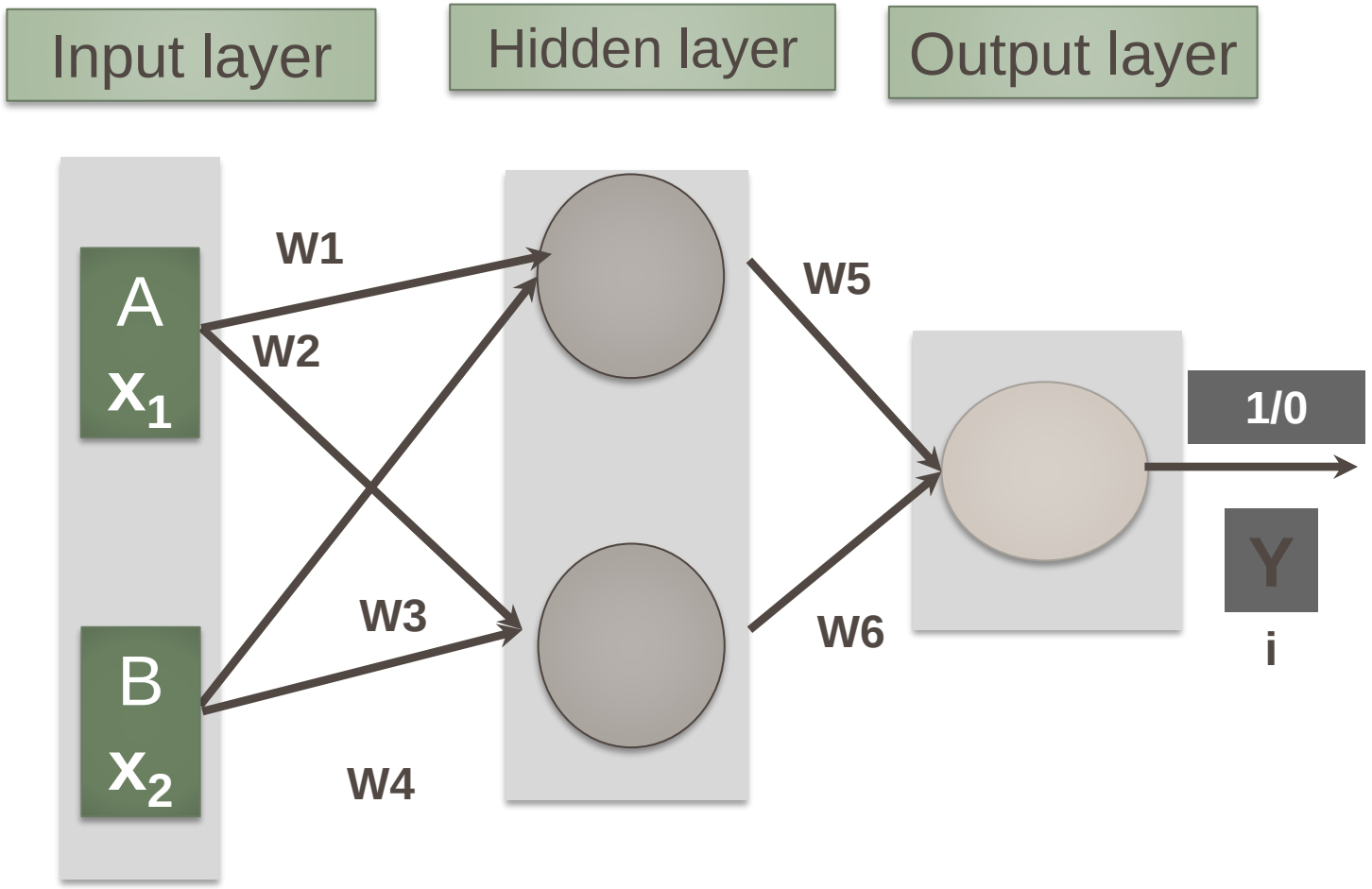


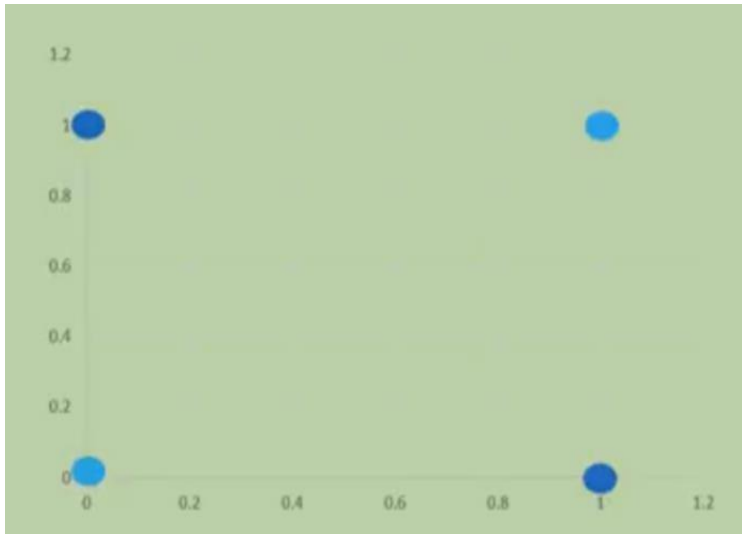


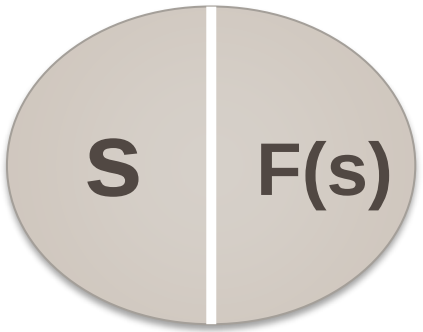
Sop=s1w5+s2w6

Activation function output

	A	B
1	1	0
	0	1
0	0	0
	1	1

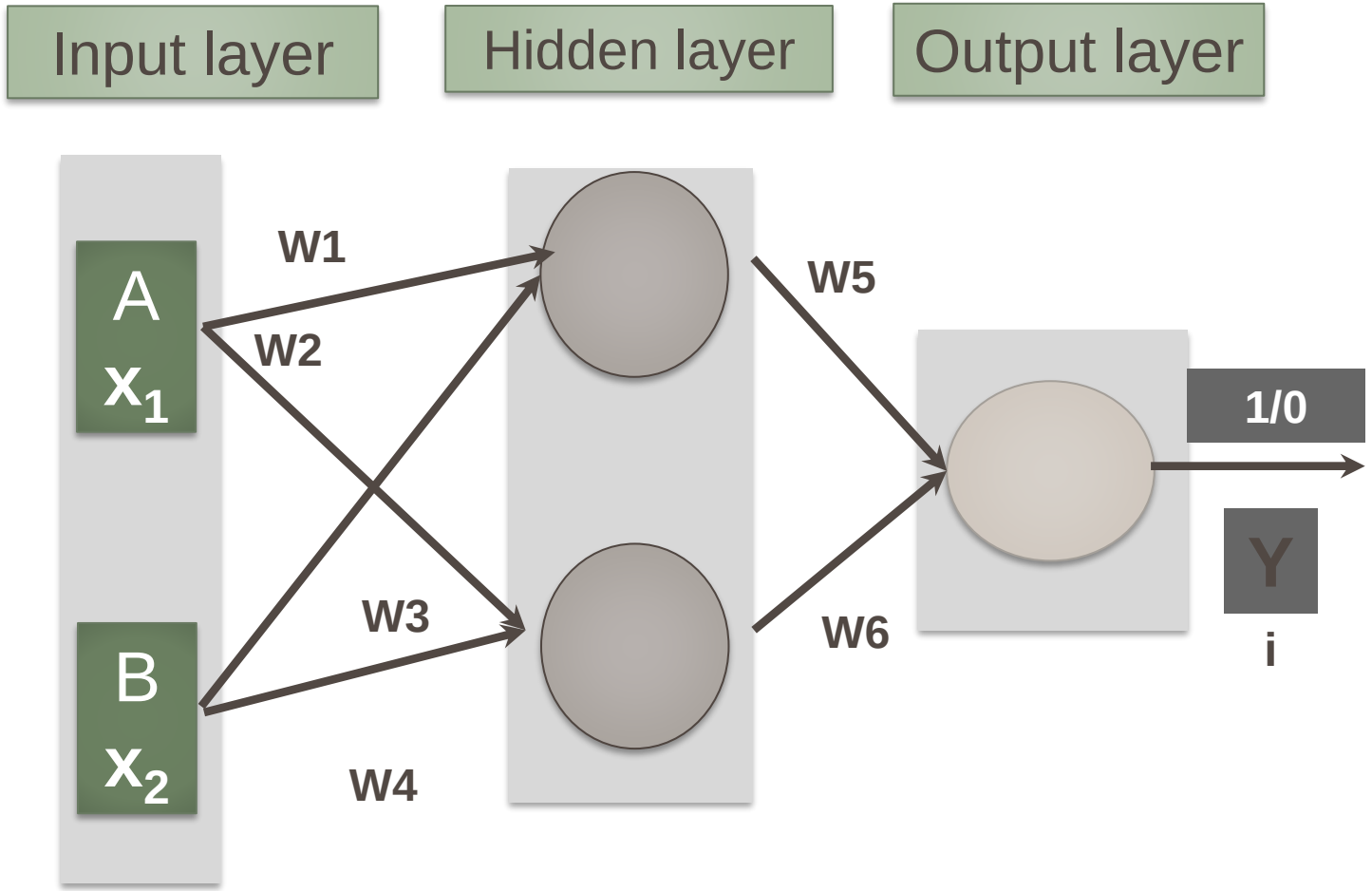


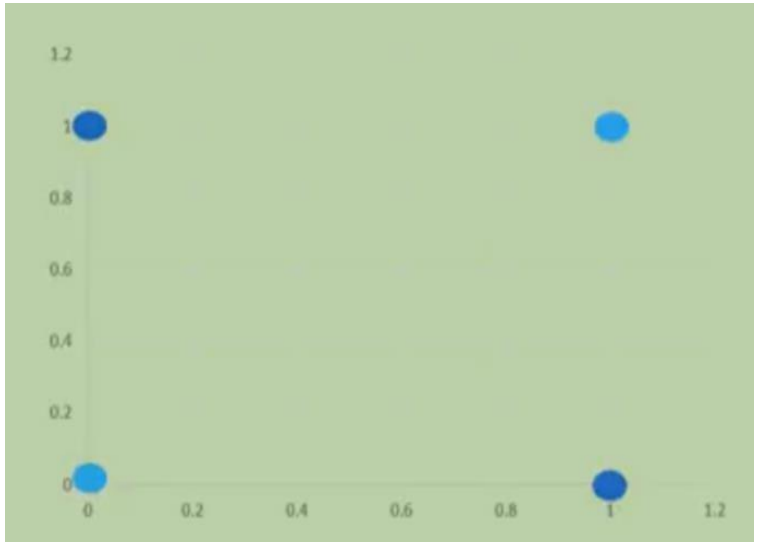


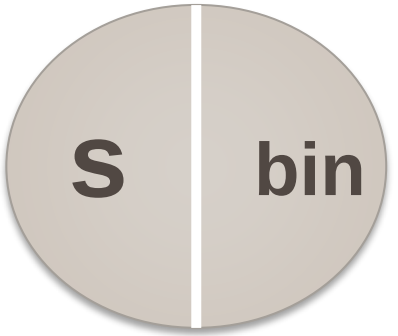


Activation function output

	A	B
1	1	0
	0	1
0	0	0
	1	1







Bias

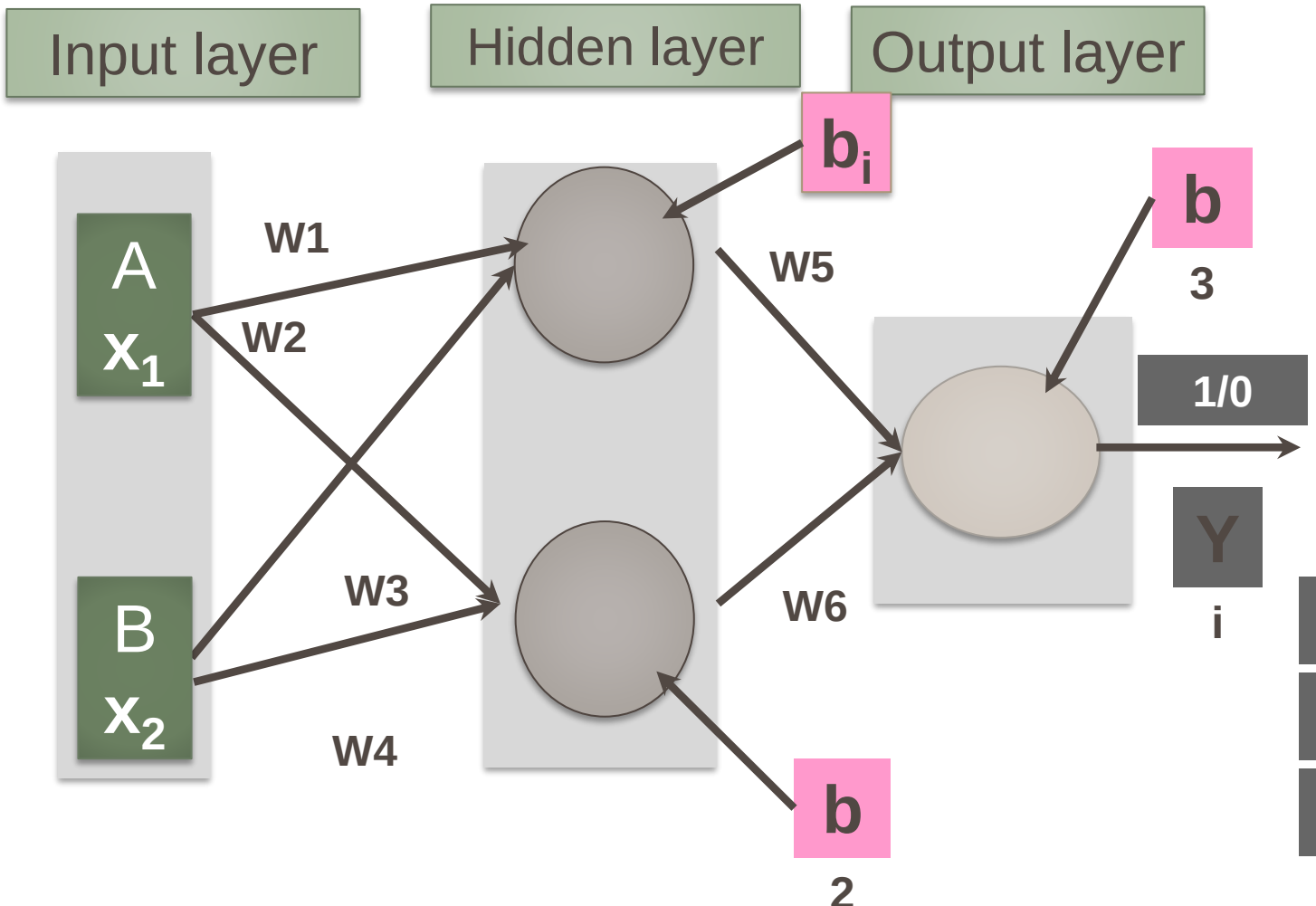
hidden & output layers neurons

	A	B
1	1	0
	0	1
0	0	0
	1	1

$$S_1=(X_1W_1+X_2W_3)$$

$$S_2=(X_1W_2+X_2W_4)$$

$$S_3=(S_1W_5+S_2W_6)$$



S1= b1 $+x_1w_1+x_2w_3$

S2= b2 $+x_1w_2+x_2w_4$

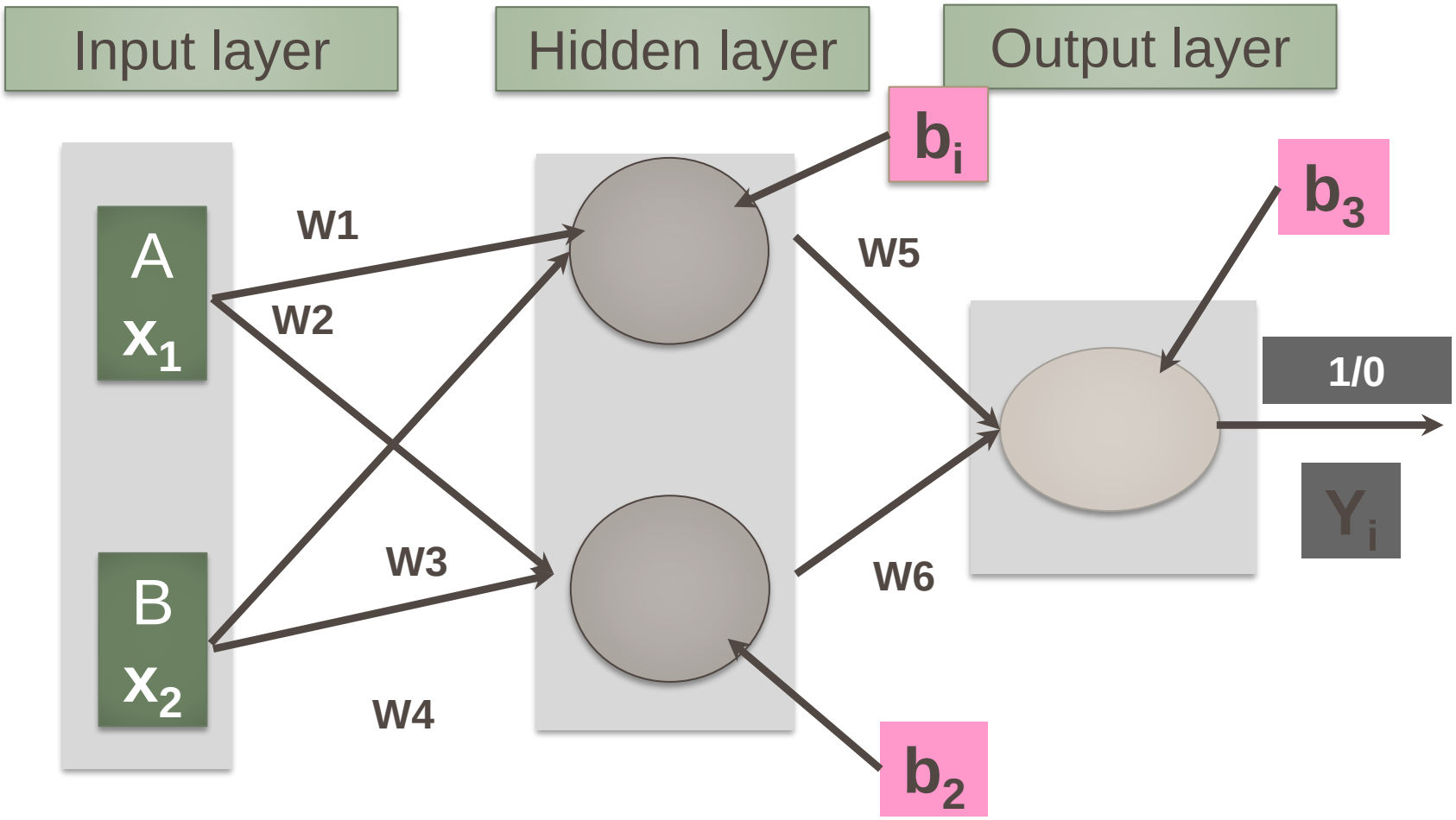
S3= b3 $+s_1w_5+s_2w_6$

Learning Rate

$0 \leq \eta \leq 1$

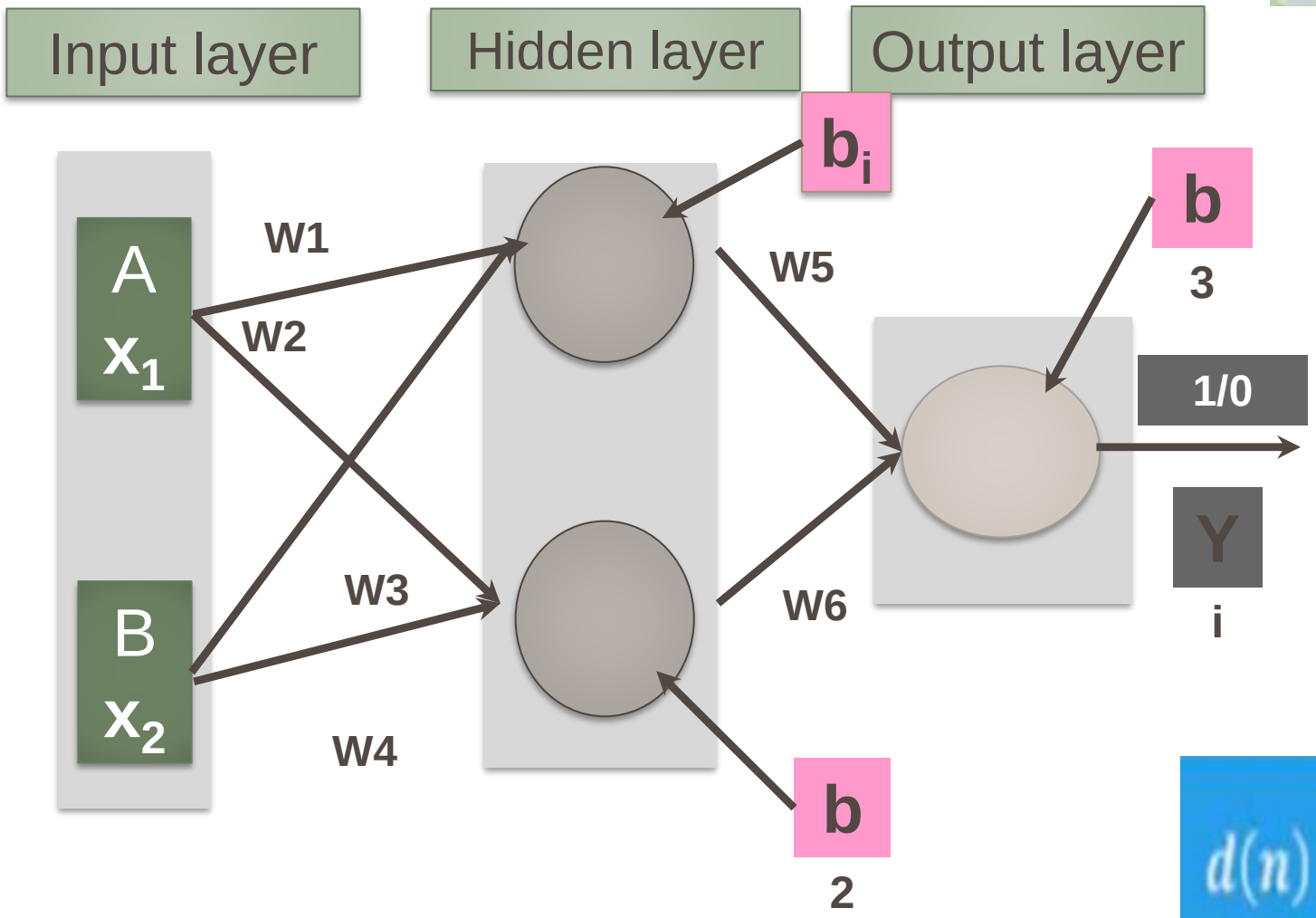
Step

$n=0,1,2,\ldots\ldots,n$



Desired Output d_j

	A	B
1	1	0
	0	1
0	0	0
	1	1



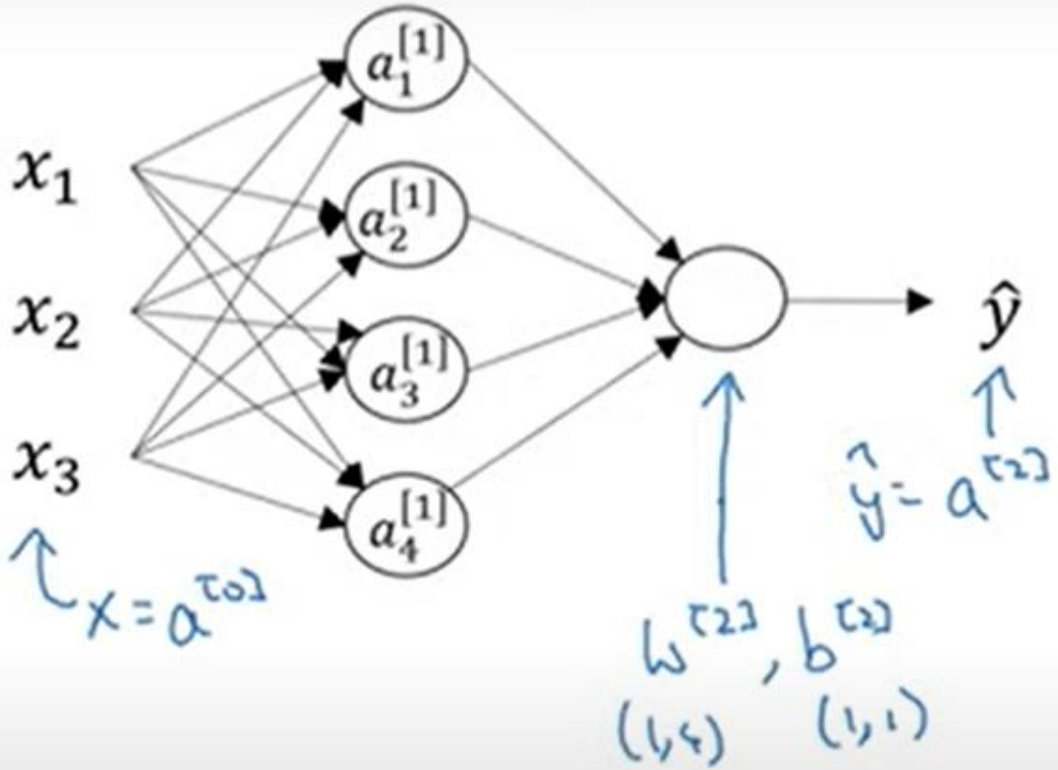
$S1 = b1 + x_1w_1 + x_2w_3$

$S2 = b2 + x_1w_2 + x_2w_4$

$S3 = b3 + s_1w_5 + s_2w_6$

$$d(n) = \begin{cases} 1, x(n) \text{ belongs to } C1\ (1) \\ 0, x(n) \text{ belongs to } C2\ (0) \end{cases}$$

Neural Network Representation learning



Given input x:

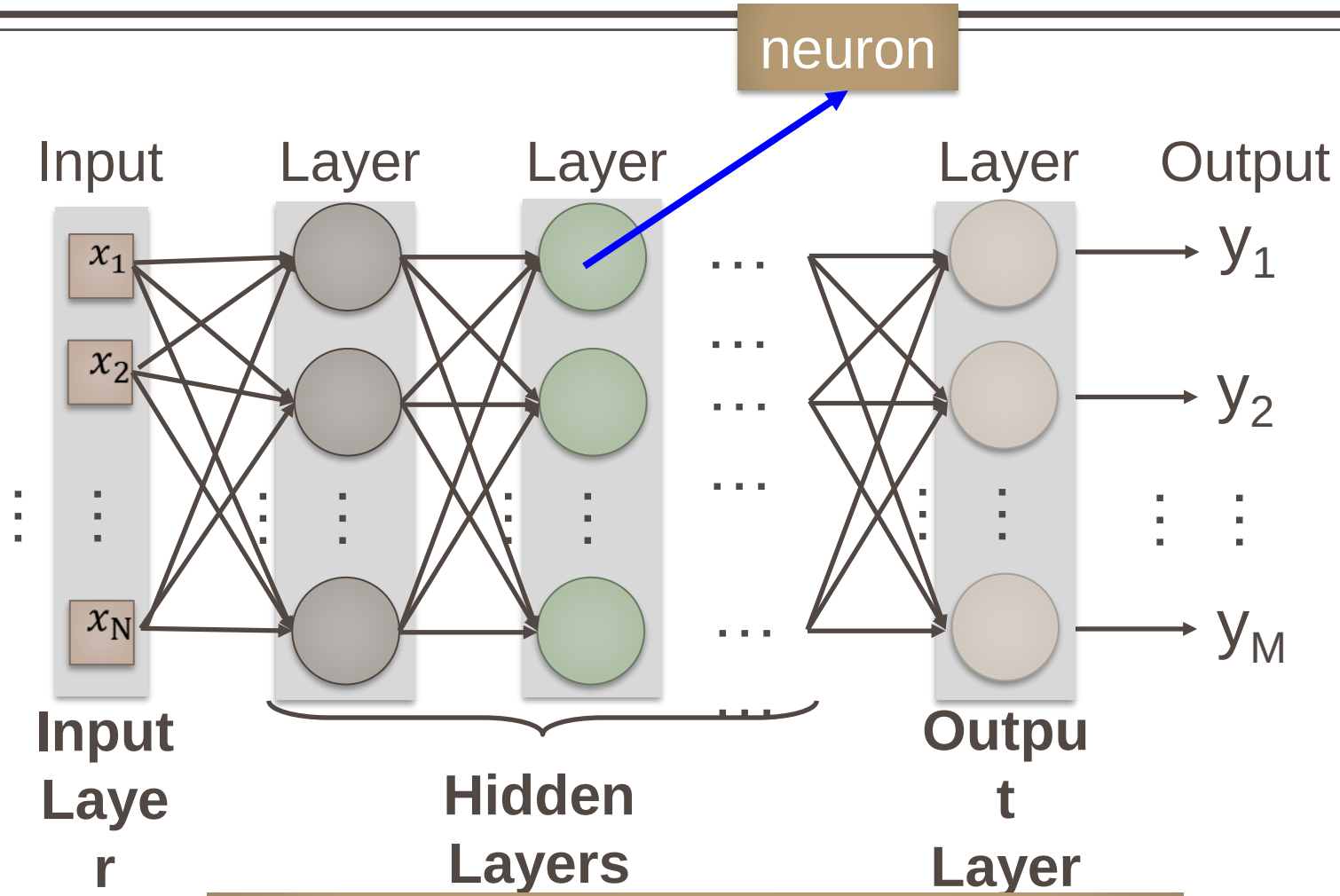
$$\rightarrow z^{[1]} = W^{[1]} a^{[0]} + b^{[1]}$$
(4,1) (4,3) (3,1) (4,1)

$$\rightarrow a^{[1]} = \sigma(z^{[1]})$$
(4,1) (4,1)

$$\rightarrow z^{[2]} = W^{[2]} a^{[1]} + b^{[2]}$$
(1,1) (1,4)

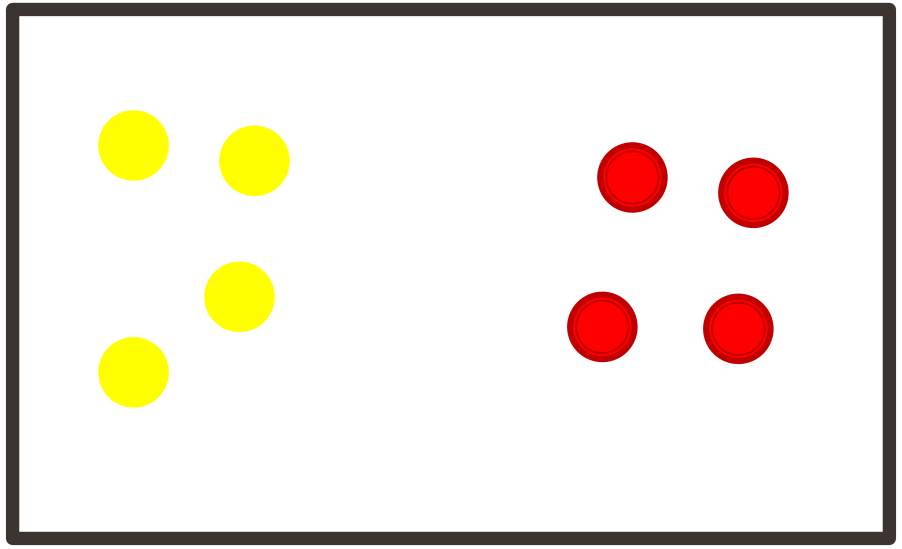
$$\rightarrow a^{[2]} = \sigma(z^{[2]})$$

Element of Neural Network



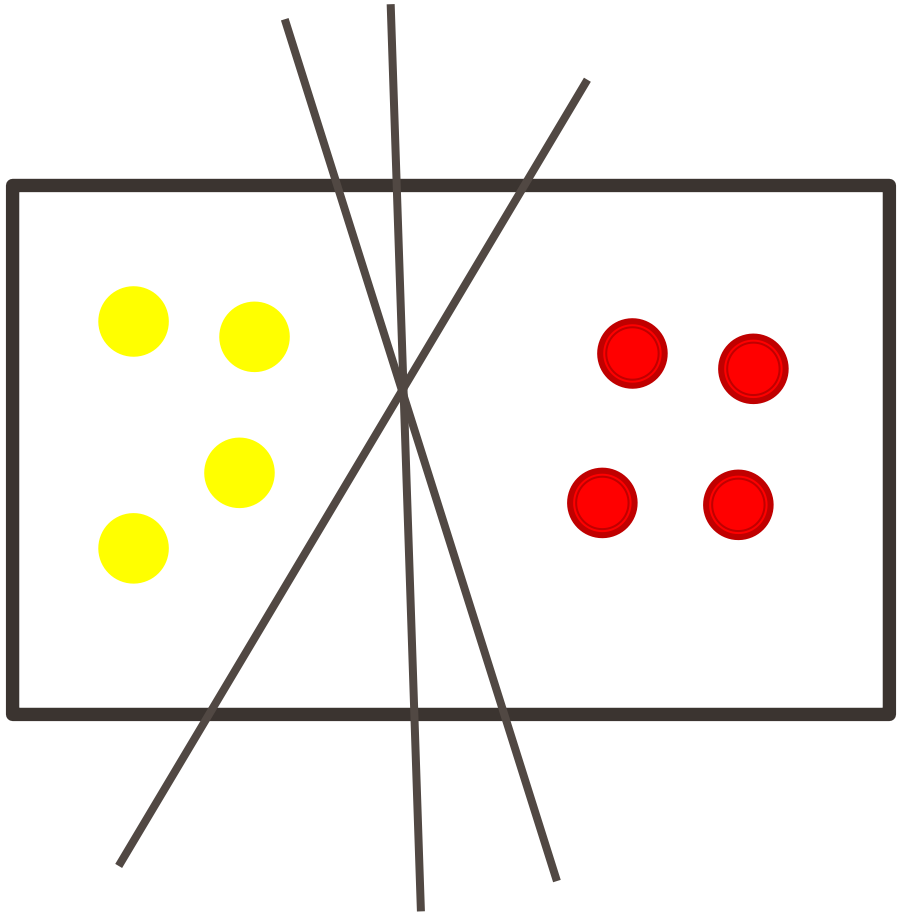
Deep means many hidden layers

Classification and Artificial neural Network



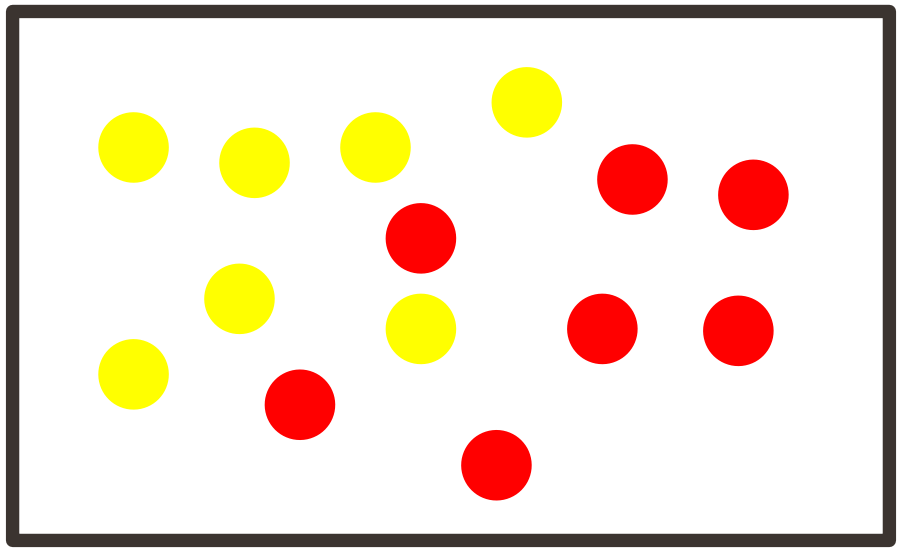
Classification and Artificial neural Network

Linear classifier



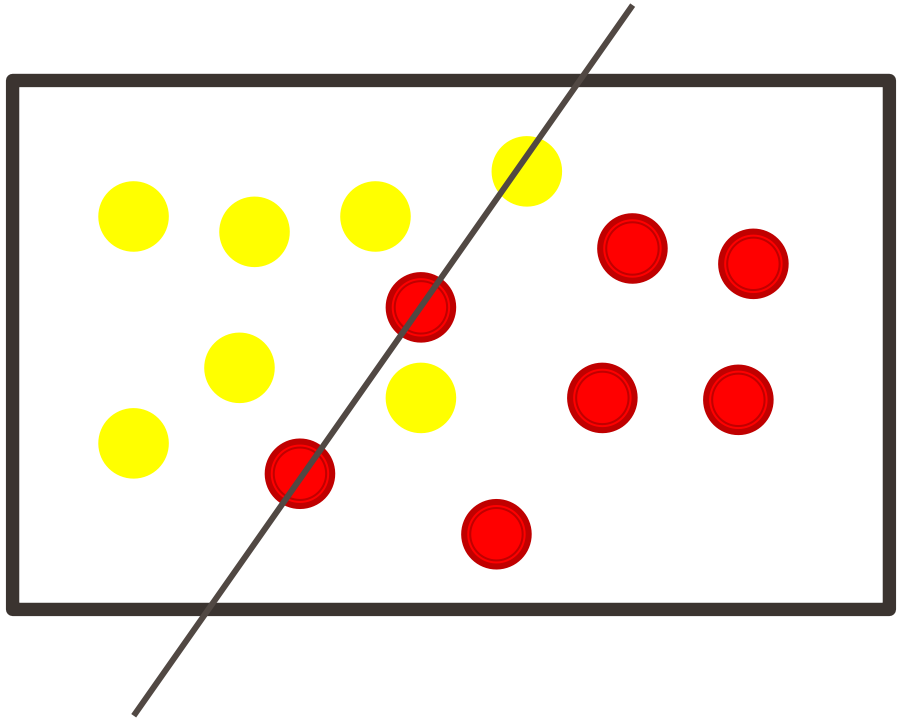
Classification and Artificial neural Network

Linear classifier and complex Data



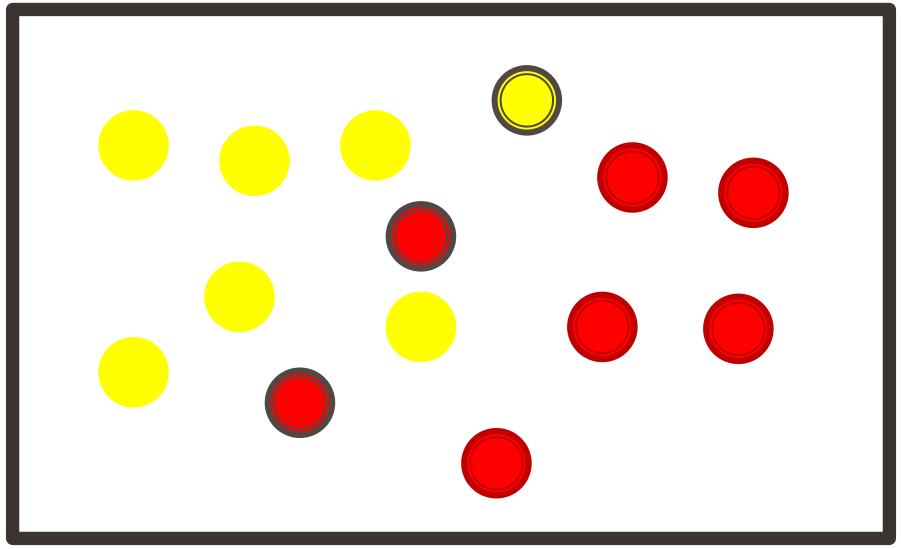
Classification and Artificial neural Network

Linear classifier and complex Data



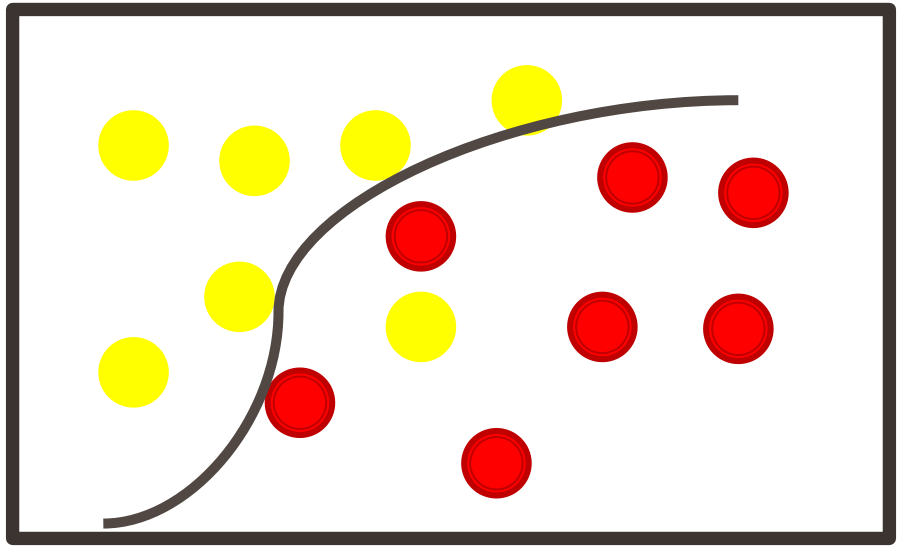
Classification and Artificial neural Network

Linear classifier and complex Data



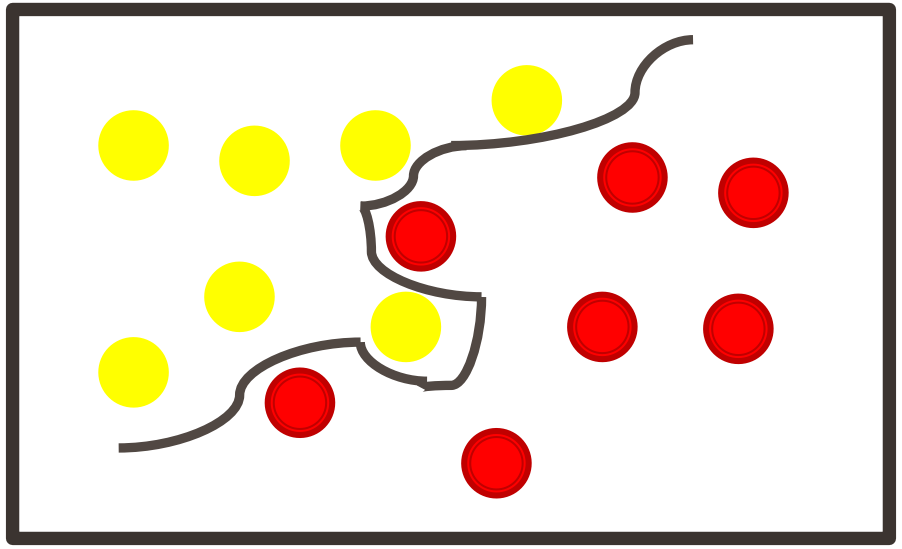
Classification and Artificial neural Network

Non Linear classifier training



Classification and Artificial neural Network

Non Linear classifier training



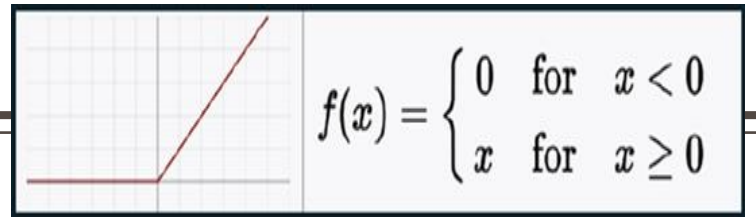
Activation Functions

In ANN, we can also apply activation functions over the input to get the exact output.
Following are some of the activation functions:

- Linear activation function (It is also called the identity function as it performs no input editing) It can be defined as: F(x) = X

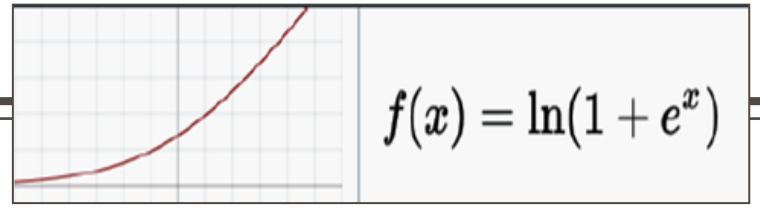
- Sigmoid activation function

Activation Function types



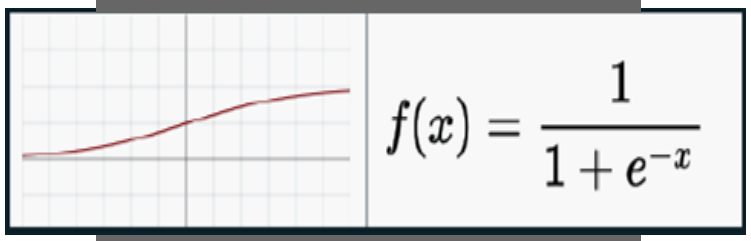
$$f(x) = \begin{cases} 0 & \text{for} \quad x < 0 \\ x & \text{for} \quad x \geq 0 \end{cases}$$

ReLU



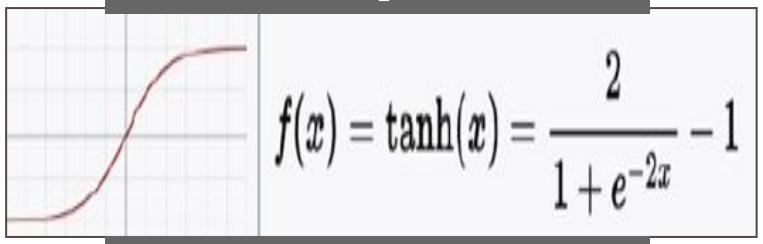
$$f(x) = \ln(1 + e^{x})$$

Softplus



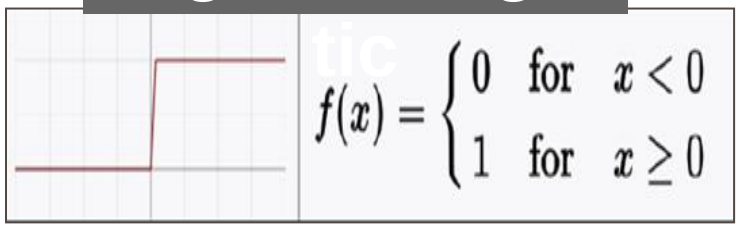
$$f(x) = \frac{1}{1 + e^{-x}}$$

Sigmoid/logis



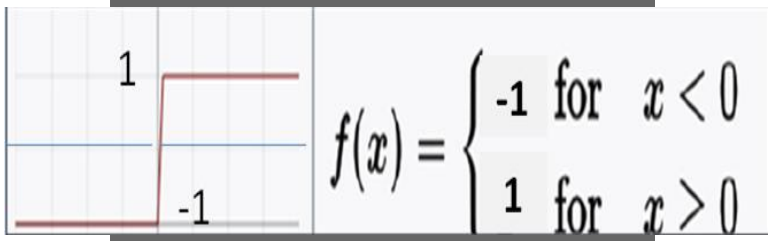
$$f(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1$$

Tanh



$$f(x) = \begin{cases} 0 & \text{for} \quad x < 0 \\ 1 & \text{for} \quad x \geq 0 \end{cases}$$

Binary



$$f(x) = \begin{cases} -1 & \text{for} \quad x < 0 \\ 1 & \text{for} \quad x > 0 \end{cases}$$

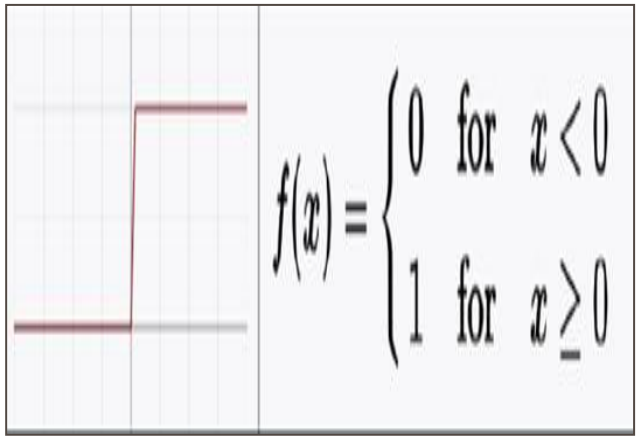
Signum

$$f_i(\vec{x}) = \frac{e^{x_i}}{\sum_{j=1}^{J} e^{x_j}} \quad \text{for } i = 1, \ldots, J$$

Softmax

Activation Function types

1- Binary Step



- Binary step function depends on a **threshold value** that decides whether a neuron should be activated or not.
- If the input is greater than it, then the neuron is activated, else it is deactivated, meaning that its output is not passed on to the next hidden layer.

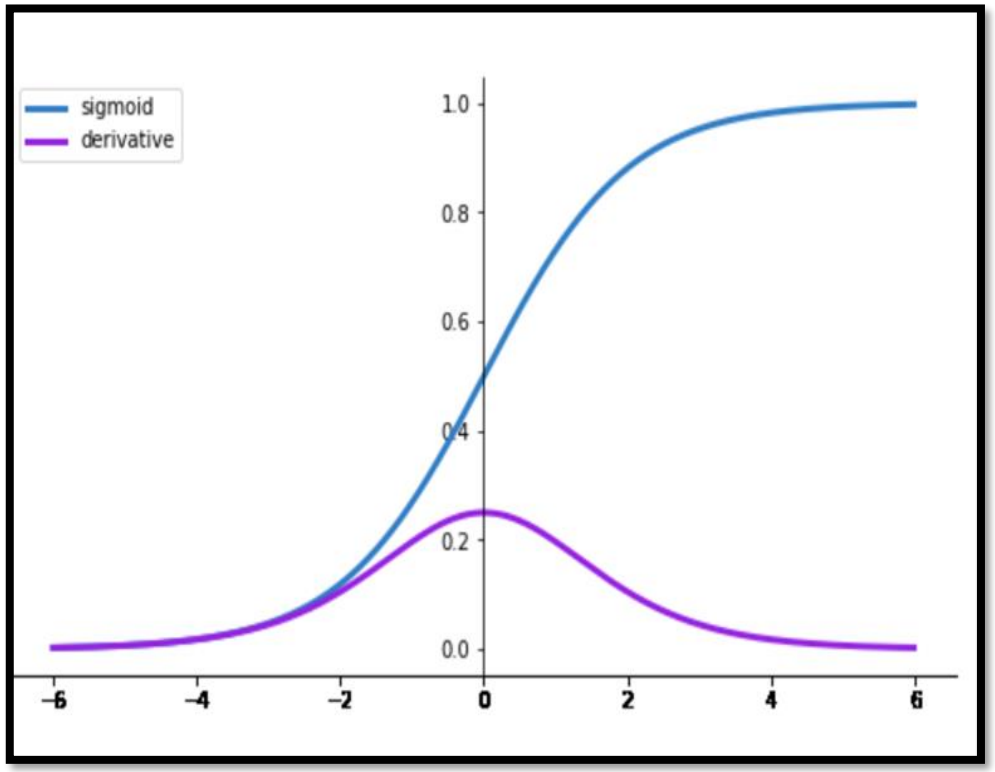
limitations

- It cannot **provide multi-value outputs**—for example, it cannot be used for multi-class classification problems.
- **The gradient of the step function is zero**, which causes a hindrance in the backpropagation process.

Activation Function types

2- Sigmoid / Logistic Activation Function

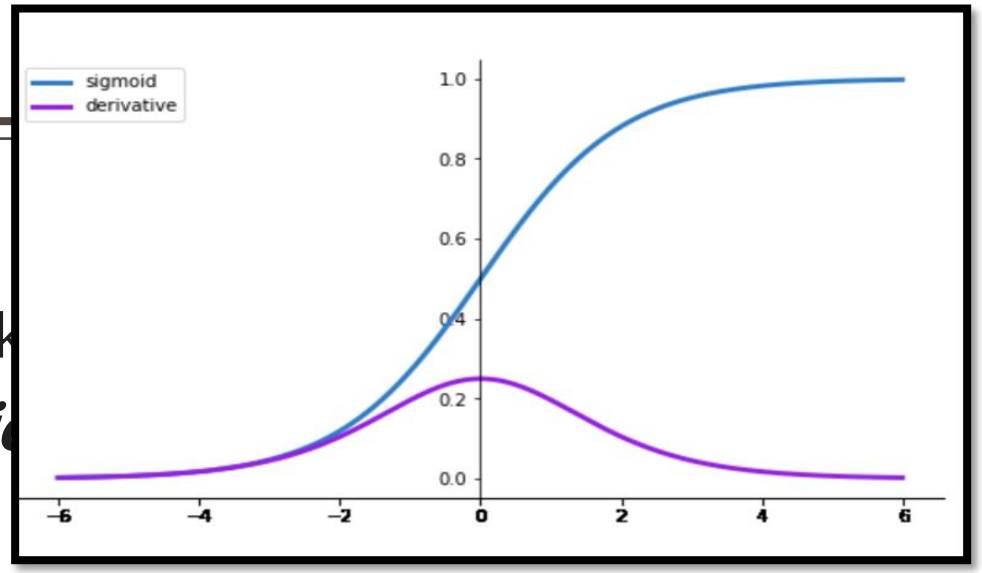
- This function takes any real value as input and outputs values in the range of 0 to 1.
- it is **commonly used** for models where we have to predict the probability as an output.
- The function is differentiable and provides a smooth gradient, i.e., preventing jumps in output values. This is represented by an S-shape. It is **derivable at every point.**



Activation Function types

2- Sigmoid / Logistic Activation Function

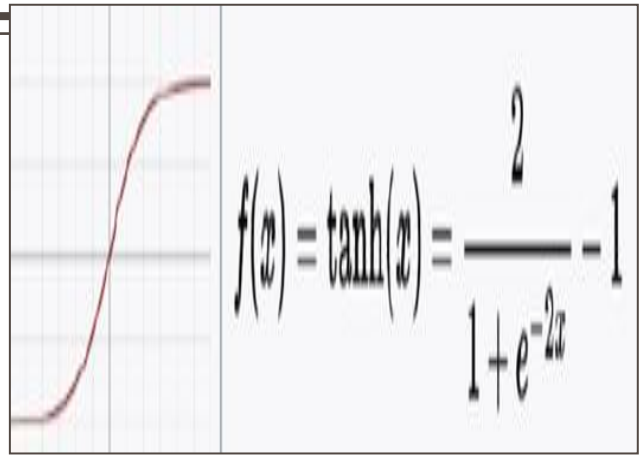
limitations



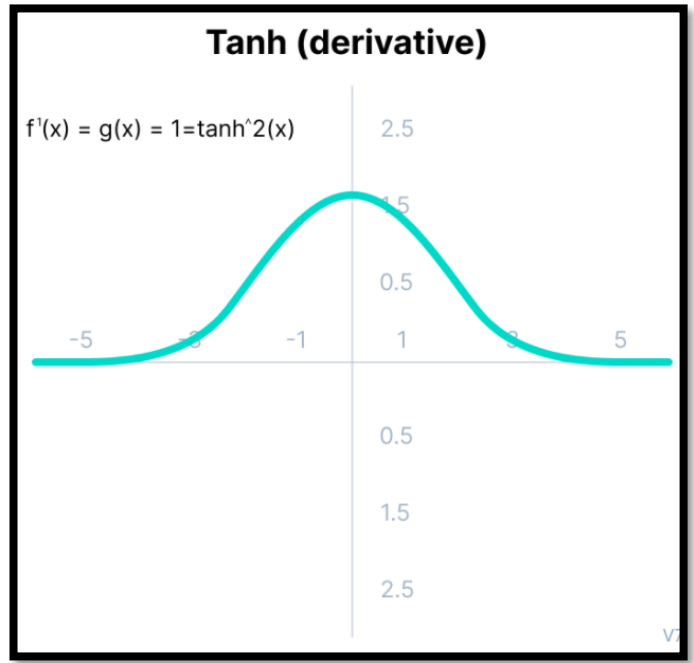
- As the gradient value approaches zero, the network stops to learn and suffers from the ***Vanishing gradi*** **problem**.
- The outputs aren't **zero centred**. The output of this activation function always lies within 0 & 1 i.e. always positive. As a result, it would take a substantially **longer** time to converge. **Whereas zero centred function helps in fast convergence.**
- It **saturates and kills** gradients. Refer to the figure of the derivative of the sigmoid. At both positive and negative ends, the value of the gradient saturates at 0. That means for those values, the gradient will be 0 or close to 0, **which simply means no learning in backpropagation.**
- It is **computationally** expensive because of the **exponential term** in it.

Activation Function types

3- Tanh (Hyperbolic Tangent)



- Tanh function is very similar to the sigmoid/logistic activation function, and even has the same S-shape with the difference in output range of -1 to 1. it is a mathematically shifted version of sigmoid .
- It has similar advantages as sigmoid but better than that because it is **zero centred**. The output of tanh lies between -1 and 1. Hence solving one of the issues with the sigmoid.

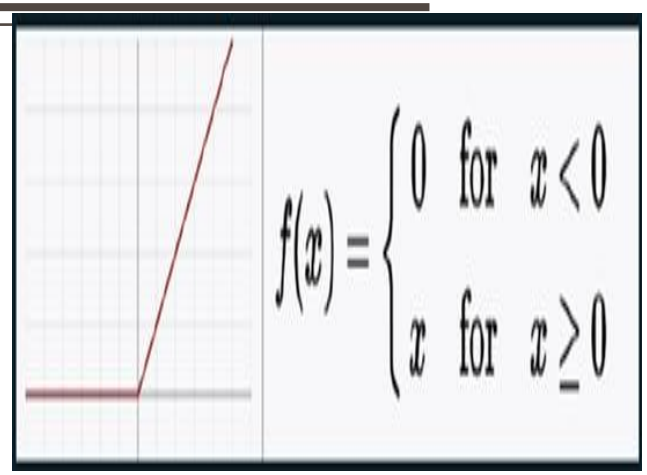


limitations

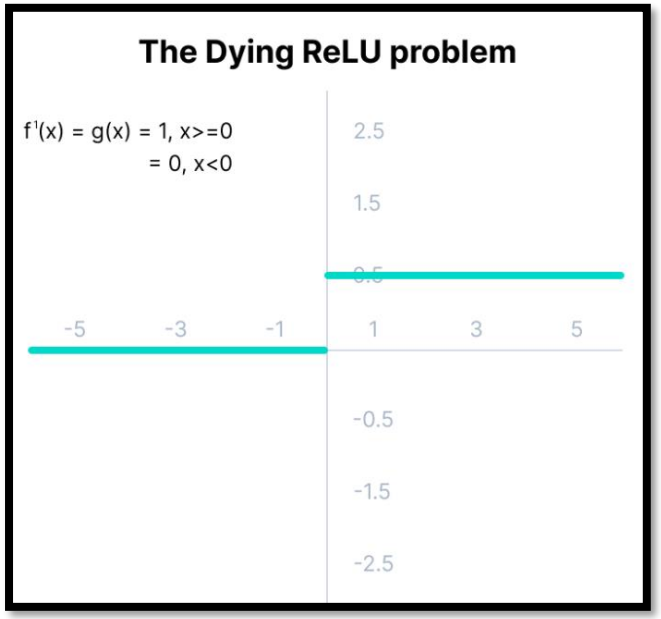
- It also **has the problem of vanishing gradient** but **the derivatives are steeper than that of the sigmoid**. Hence making the **gradients stronger for tanh than sigmoid**.
- As it is almost similar to sigmoid, tanh is also computationally expensive. because of the **exponential term** in it.
- Similar to sigmoid, here also the gradients saturate.

Activation Function types

4- ReLU Function (Rectified Linear Unit)



- It is **computationally effective** as it involves simpler mathematical operations than sigmoid and tanh.
- Although it looks like a linear function, it adds **non-linearity** to the network, making it able to **learn complex patterns**.
- It doesn't suffer from the **vanishing gradient** problem.
- It is **unbounded** at the positive side. Hence removing the problem of gradient saturation.



limitations

- It suffers from the **dying ReLU** problem. ReLU is always going to discard the negative values i.e. the deactivations by making it 0. But because of this, the gradient of these units will also become 0 .
- It is non-differentiable at 0.

Activation Function types

5- Softmax

$$f_i(\vec{x}) = \frac{e^{x_i}}{\sum_{j=1}^{J} e^{x_j}} \quad \text{for } i = 1, \ldots, J$$

- the Softmax function is described as a **combination of mu**

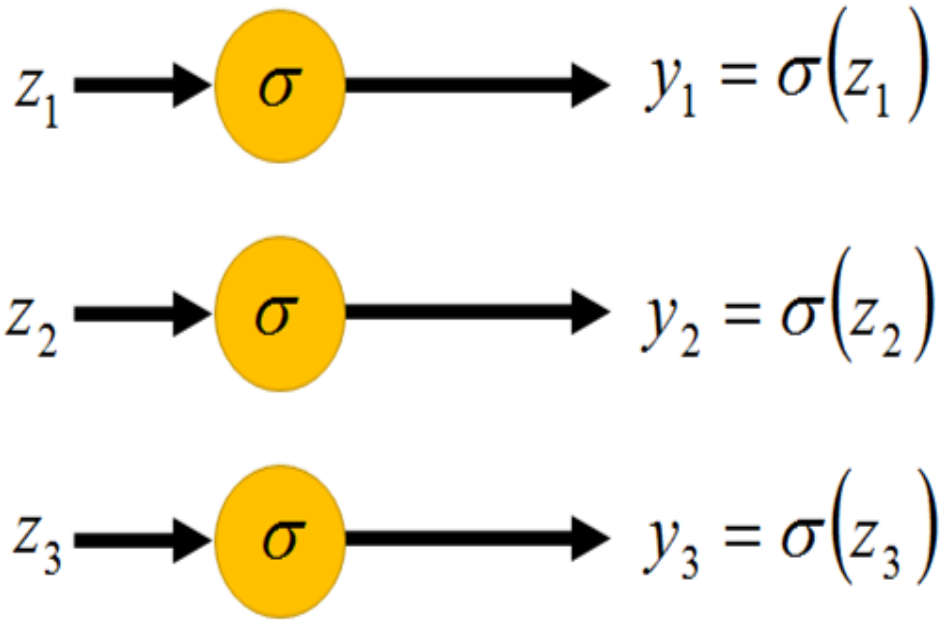
sigmoids.

- It calculates the **relative probabilities**. Similar to the sigmoid/logistic activation function, the SoftMax function returns the probability of each class.
- It is most commonly used as an activation function for **the last layer** of the neural network in the case of **multi-class classification**.
- It is able to handle **multiple classes**. It *normalizes* the outputs for each class between 0 and 1 and divides by their sum. Hence forming a **probability distribution**. Therefore giving a clear probability of input belonging to any particular class.

Softmax

- Softmax layer as the output layer

Ordinary Layer



In general, the output of network can be any value.

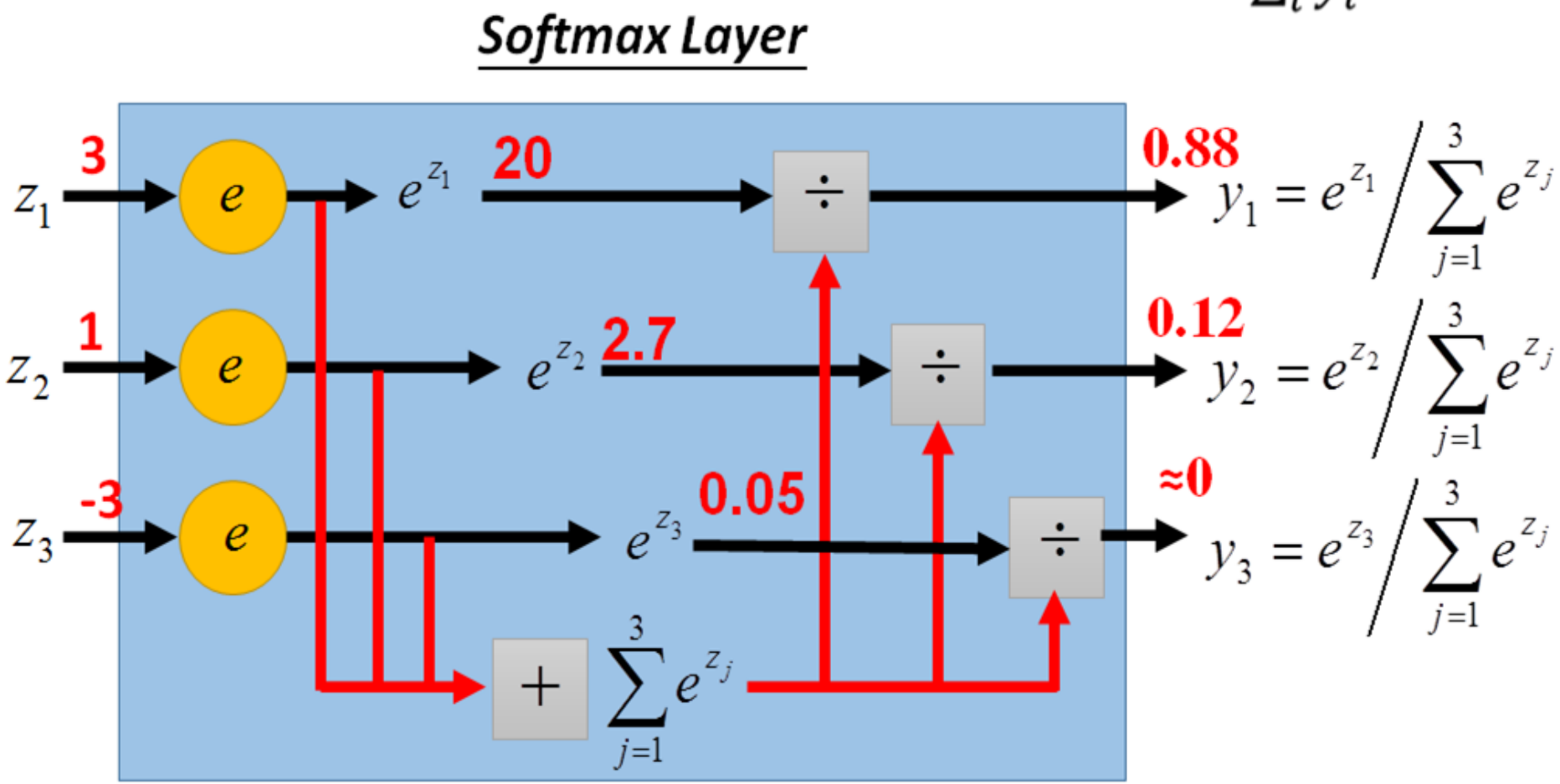
May not be easy to interpret

Softmax

- Softmax layer as the output layer

Probability:

- $1 > y_i > 0$
- $\sum_i y_i = 1$



Neural Network training steps

1. Weight Initialization
2. Inputs Application
3. Sum of inputs - Weights product
4. Activation functions
5. Weights Adaptations
6. Back to step 2

Regarding 5th step: Weights Adaptation

First

- If the predicted output Y is not the same as the desired output d, then weights are to be adapted according to the following equation:

$$W(n+1) = W(n) + \eta[d(n) - Y(n)]X(n)$$

Where

$$W(n) = [b(n), W_1(n), W_2(n), W_3(n), \dots, W_m(n)]$$

Learning Rate η $0 \leq \eta \leq 1$ $0 \leq \alpha \leq 1$

Q. Why new weights are better than old weights?

Q. What is the effect of each weight over the prediction error?

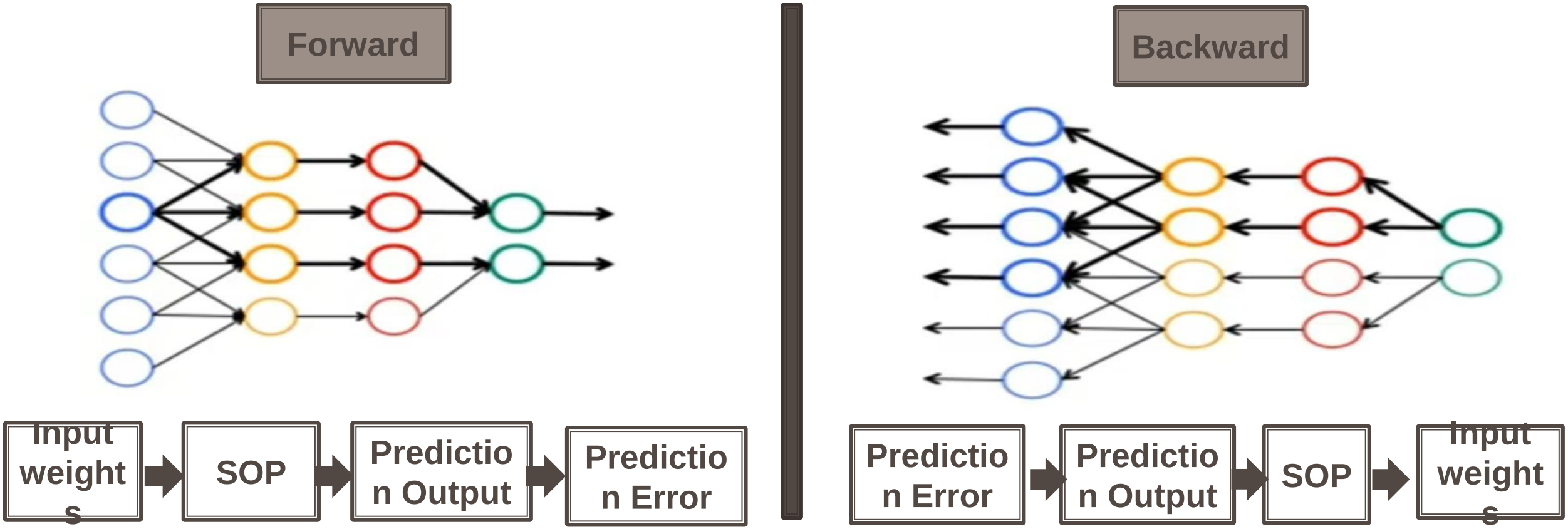
Q. How increasing or decreasing weights affects the prediction error?

Regarding 5th step: Weights Adaptation

second method: Back

- **Forward VS Backward passes**

The Backpropagation algorithm is a sensible approach for dividing the contribution of each weight.



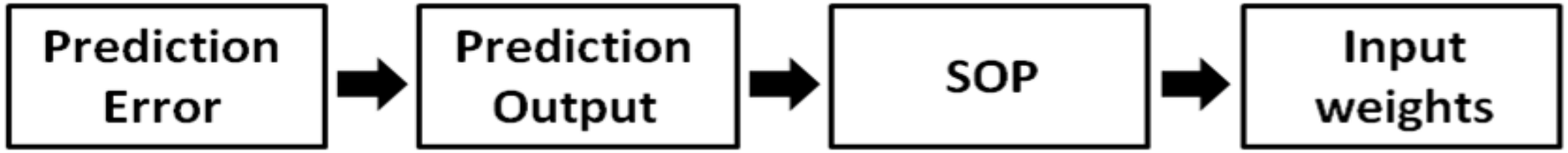
Regarding 5th step: Weights Adaptation

second method: Back

- **Backward pass**

What is the change in prediction Error (E) given the change in weight (W) ?

Get partial derivative of E W.R.T $\frac{\partial E}{\partial W}$



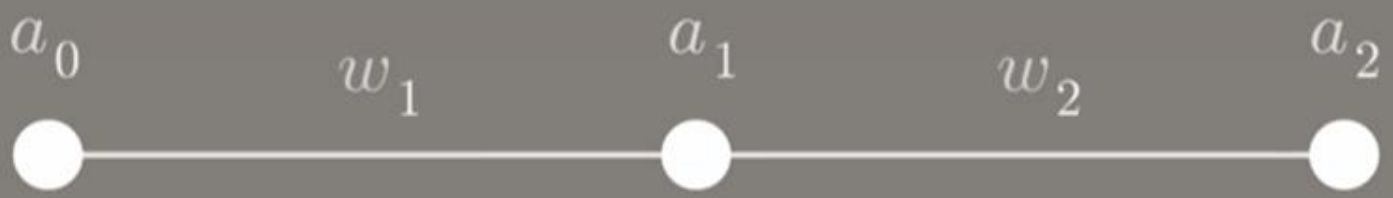
$$E = \frac{1}{2}(d - y)^2 \qquad f(s) = \frac{1}{1 + e^{-s}} \qquad s = \sum_{j}^{m} x_i \, w_{ji} + b_i \qquad w_1, w_2$$

d (desired output) Const
y (predicted output)

s (Sum Of Product SOP)

$$E = \frac{1}{2}\left(d - \frac{1}{e^{-\sum_{j}^{n} x_i w_{ij} + b_i}}\right)^2$$

Chain Rule



$$z_1 = a_0 w_1 + b_1$$

$$a_1 = A(z_1)$$

$$z_2 = a_1 w_2 + b_2$$

$$a_2 = A(z_2)$$

$$E = C(a_2, y)$$

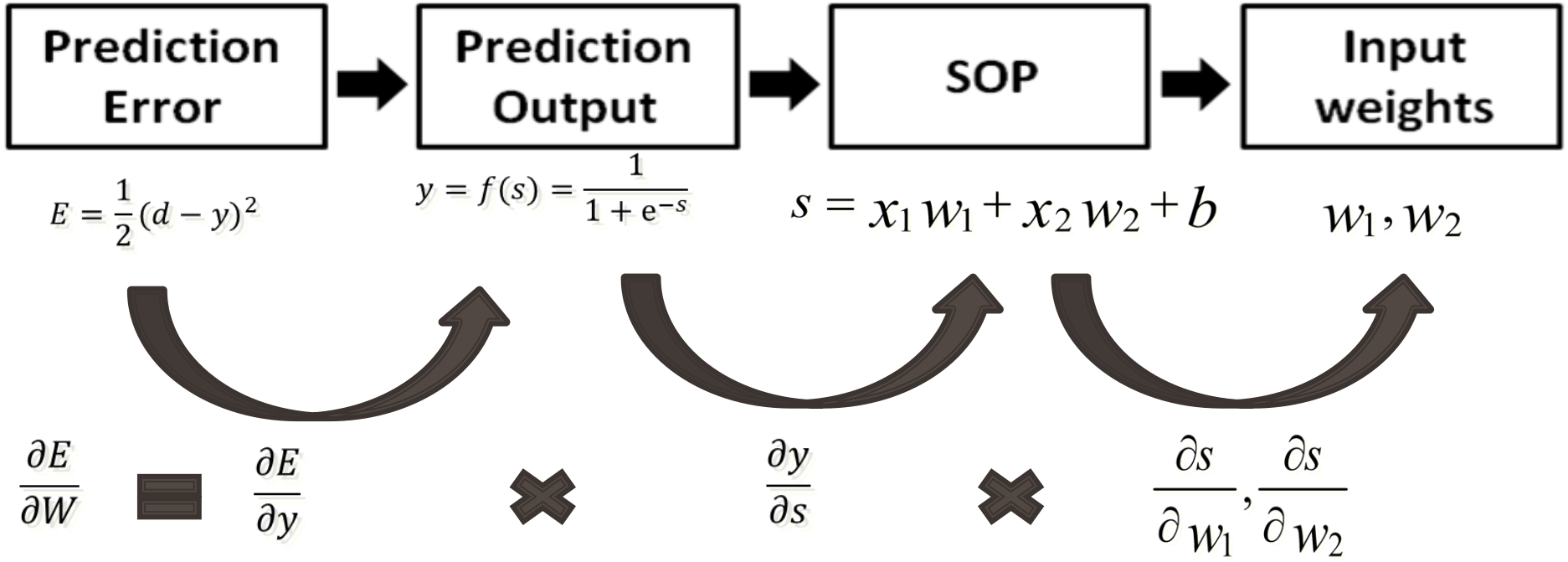
$$\frac{\partial E}{\partial w_2} = \frac{\partial z_2}{\partial w_2} \times \frac{\partial a_2}{\partial z_2} \times \frac{\partial E}{\partial a_2}$$

Regarding 5th step: Weights Adaptation

second method: Back

Chain Rule

- Weight derivative



$$\frac{\partial E}{\partial w_1} = \frac{\partial E}{\partial y} x \frac{\partial y}{\partial s} x \frac{\partial s}{\partial w_1}$$

$$\frac{\partial E}{\partial w_2} = \frac{\partial E}{\partial y} x \frac{\partial y}{\partial s} x \frac{\partial s}{\partial w_2}$$

Regarding 5th step: Weights Adaptation

second method: Back

- **Weight derivative**

$$\frac{\partial E}{\partial y} = \frac{\partial}{\partial y}\frac{1}{2}(d-y)^2 = y - d$$

$$\frac{\partial y}{\partial s} = \frac{\partial}{\partial s}\frac{1}{1+\mathrm{e}^{-s}} = \frac{1}{1+\mathrm{e}^{-s}}\left(1-\frac{1}{1+\mathrm{e}^{-s}}\right)$$

$$\frac{\partial s}{\partial w_1} = \frac{\partial}{\partial w_1} x_1 w_1 + x_2 w_2 + b = x_1$$

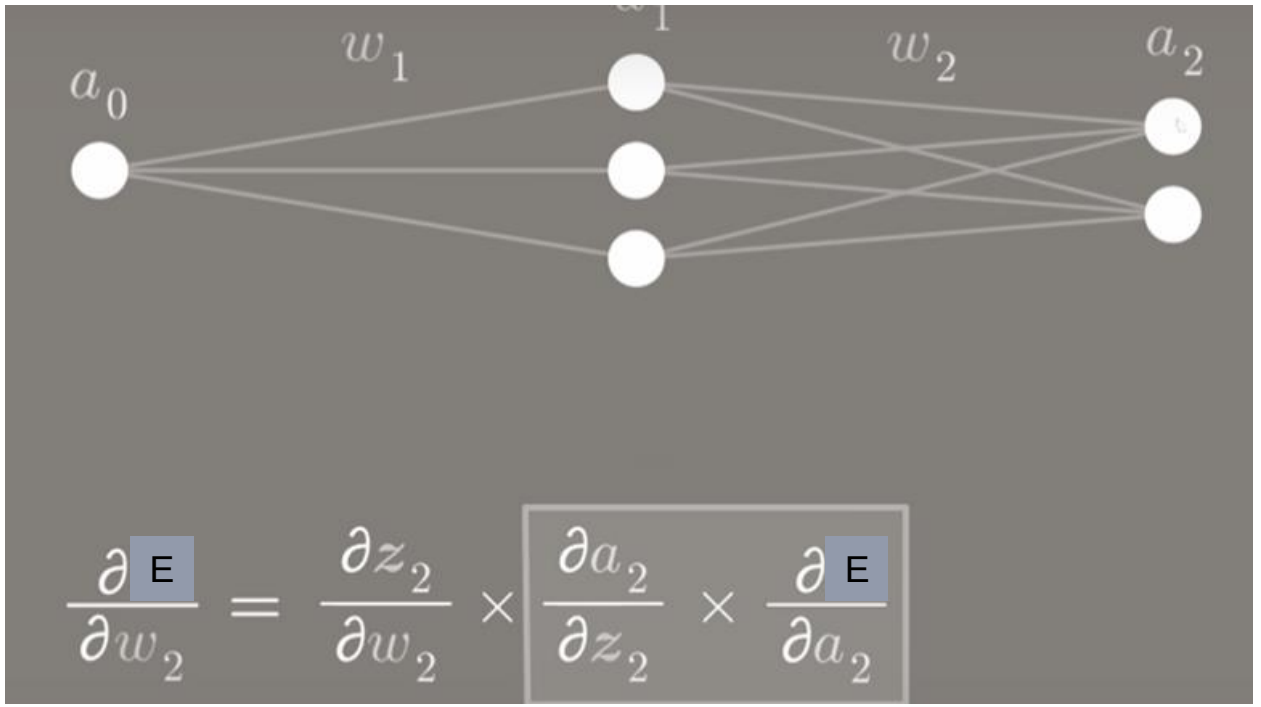
$$\frac{\partial s}{\partial w_2} = \frac{\partial}{\partial w_2} x_1 w_1 + x_2 w_2 + b = x_2$$

$$\frac{\partial E}{\partial w_i} = (y-d)\frac{1}{1+\mathrm{e}^{-s}}\left(1-\frac{1}{1+\mathrm{e}^{-s}}\right)x_i$$

Partial derivatives are used when dealing with functions with multiple variables.
For example, $E(a, y)$ has two "parts" to the derivative that we could calculate:

$\frac{\partial E}{\partial a}$ tells us how the $cost$ changes if we make a tiny change to a, but keep y the same.

$\frac{\partial E}{\partial y}$ tells us how the $cost$ changes if we make a tiny change to y, but keep a the same.



Regarding 5th step: Weights Adaptation

second method: Back

- interpreting derivatives ∇w

$$\frac{\partial E}{\partial w_i} = (y - d)\frac{\partial f(s)}{\partial s} x_i$$

Derivatives sign

Increasing/decreasing weight increases/decreases error.

Positive
directly proportional

Increasing/decreasing weight decreases/increases error.

Negative
opposite

Derivatives Magnitude

Increasing/decreasing weight by P increases/decreases error by MAG*P.

Increasing/decreasing weight by P decreases/increases error by MAG*P.

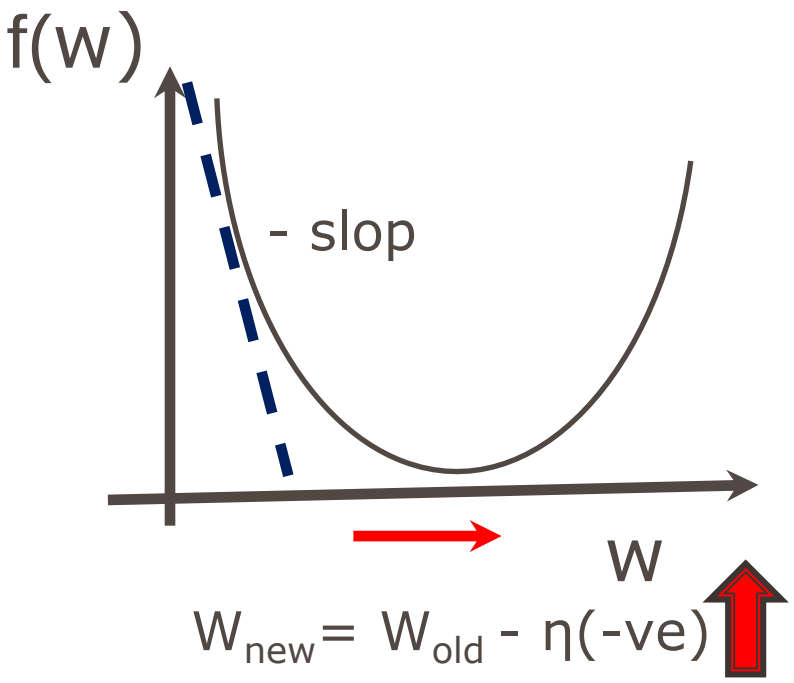
Regarding 5th step: Weights Adaptation

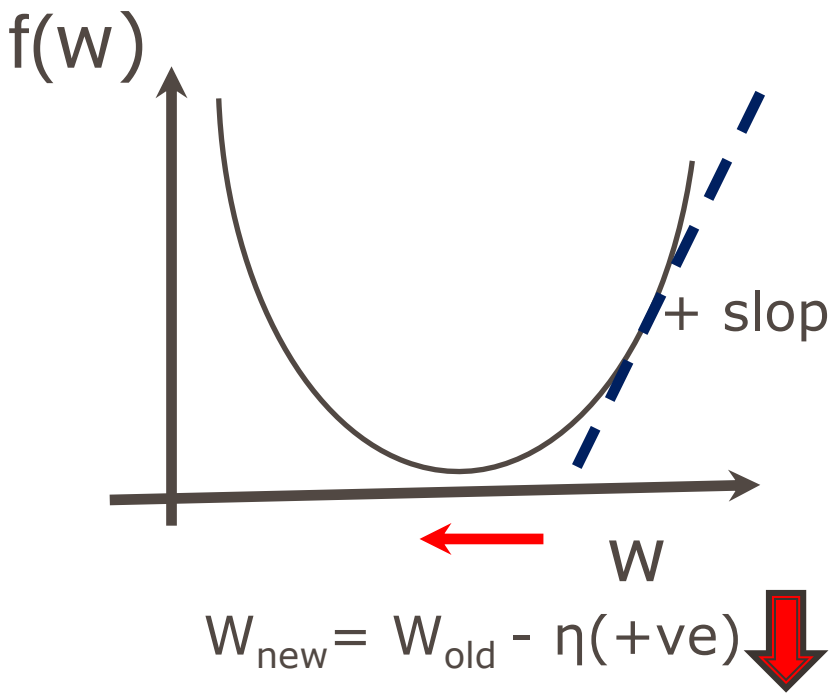
second method: Back

- **Update the Weights**

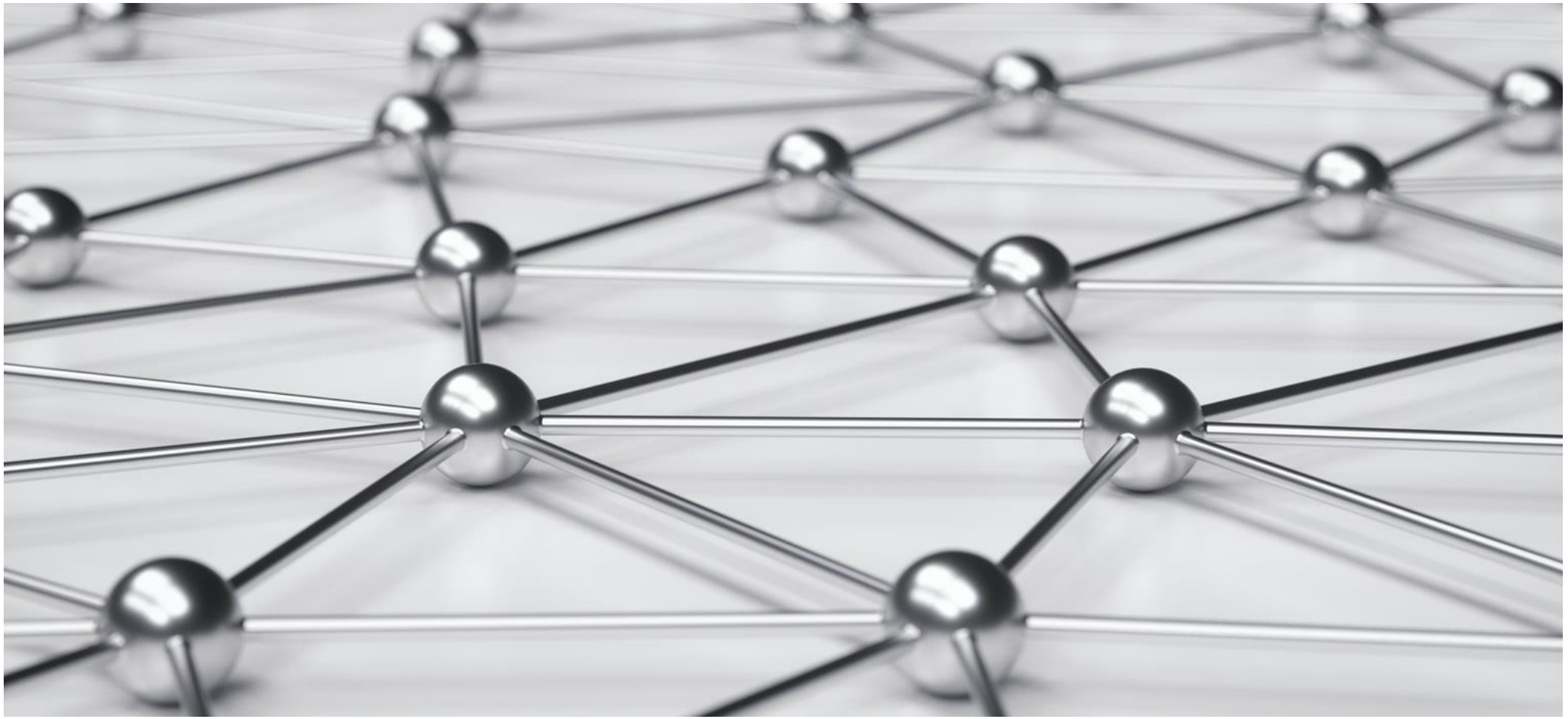
  In order to update the weights , use the Gradient Descent

$$W_{inew} = W_{iold} - \eta * \frac{\partial E}{\partial W_i}$$

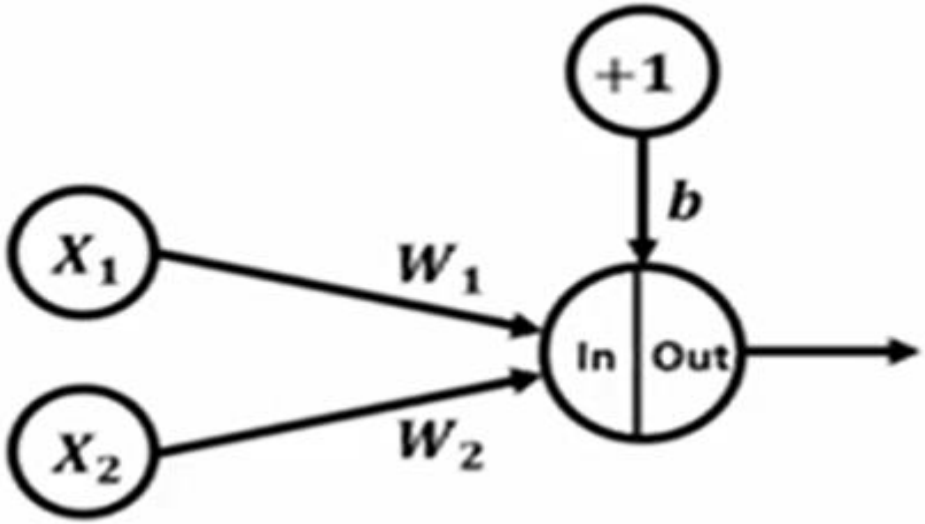


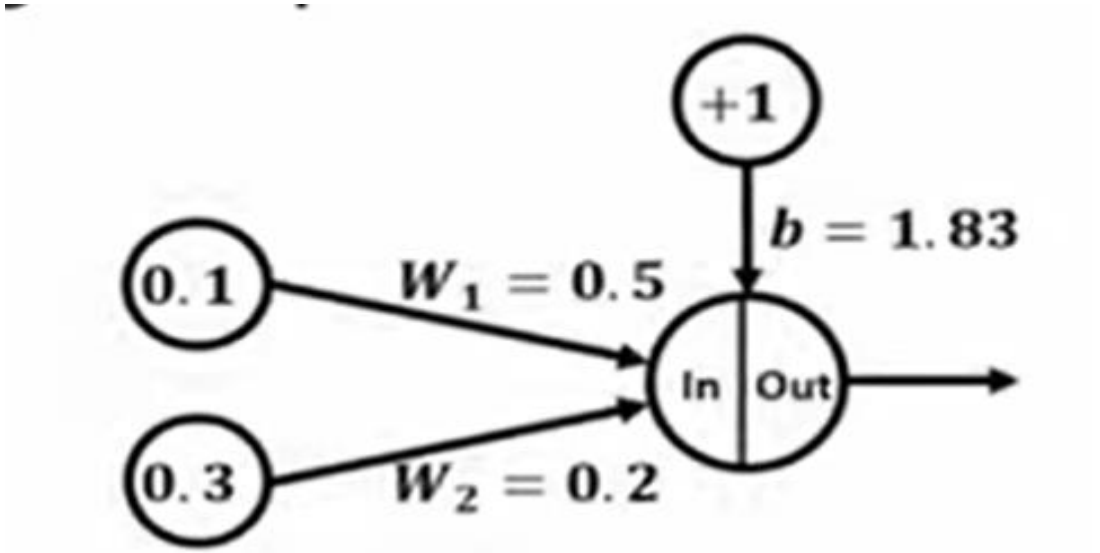


Simple Neural Network Training Example (Backpropagation)



Neural Networks training example





Training Data

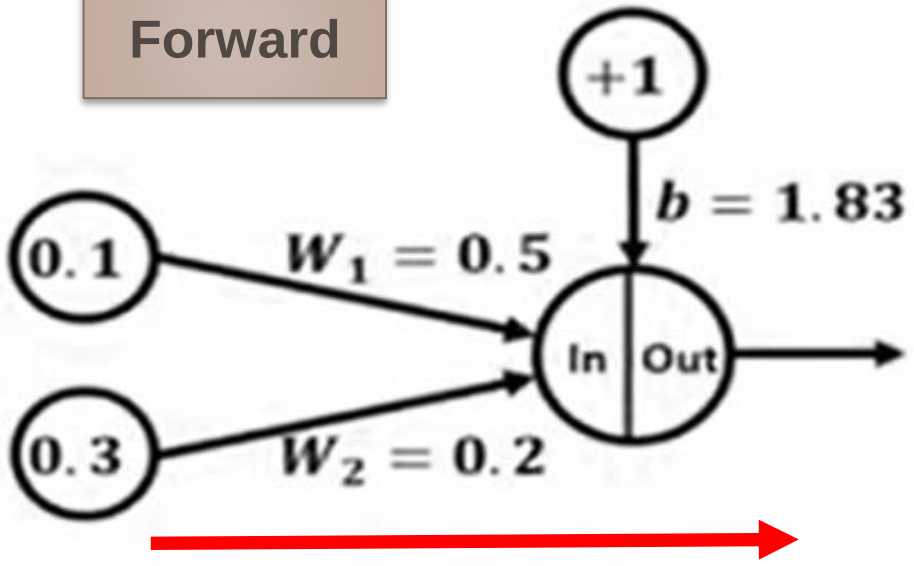
X_1	X_2	Output
0.1	0.3	0.03

Initial Weights

W_1	W_2	b
0.5	0.2	1.83

Neural Networks training : Steps

Forward



1- Activation function input (s

$$s = X_1 * W_1 + X_2 * W_2 + b$$

$$s = 0.1 * 0.5 + 0.3 * 0.2 + 1.83$$

$$s = 1.94$$

2- Activation function output

- In this example , the sigmoid activation function is used.
- Based on the **SOP** calculated previously the output is as follows

$$f(s) = \frac{1}{1 + e^{-s}}$$

$$f(s) = \frac{1}{1 + e^{-1.94}} = \frac{1}{1 + 0.144} = \frac{1}{1.144}$$

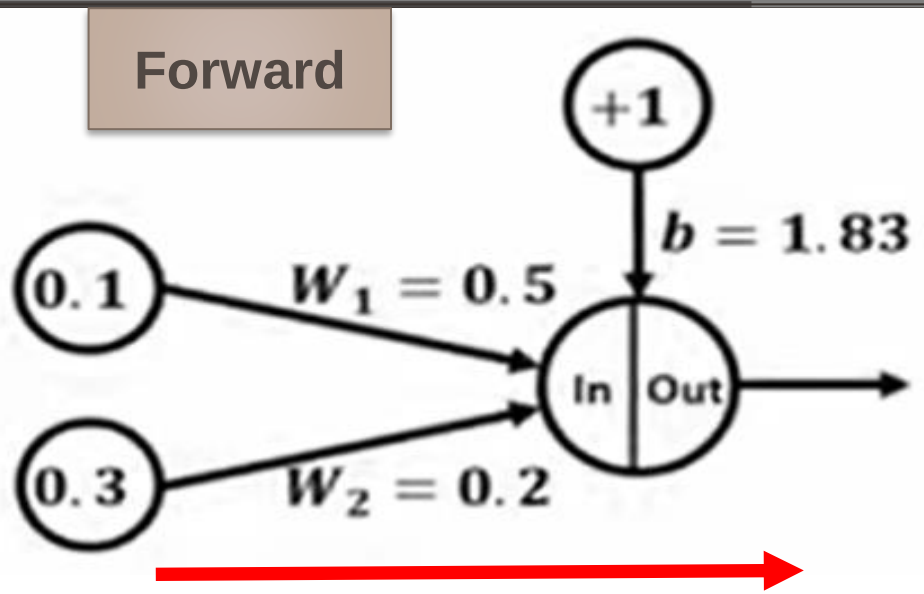
$$f(s) = 0.874$$

Neural Networks training : Steps

3- Prediction Error

- The square error function .

$$E = \frac{1}{2}(desired - predicted)^2$$



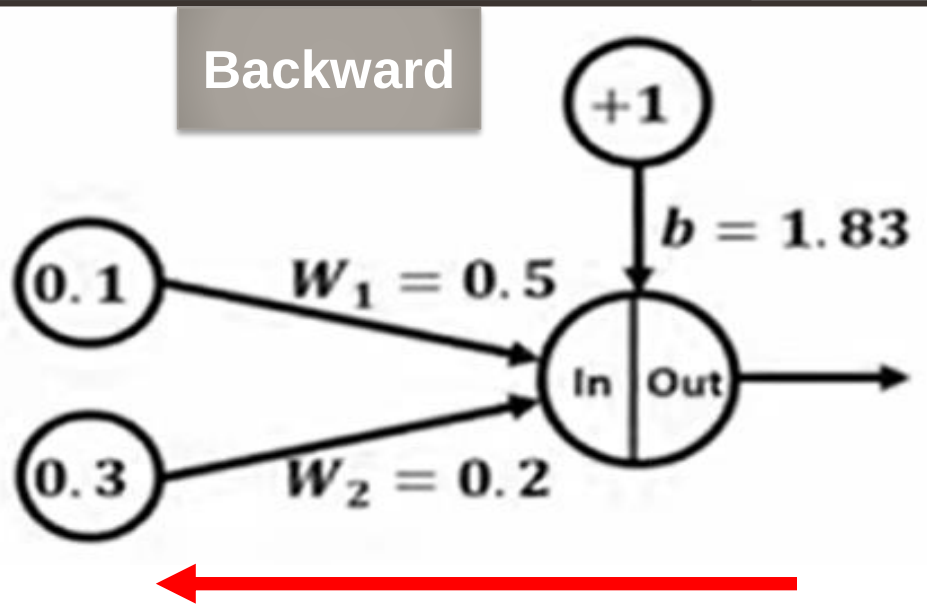
- Based on the predicted output ,The prediction error is:-

$$E = \frac{1}{2}(0.03 - 0.874)^2 = \frac{1}{2}(-0.844)^2 = \frac{1}{2}(0.713)$$

$$E = 0.357$$

Neural Networks training : Steps

$$\frac{\partial E}{\partial W} = \frac{\partial E}{\partial y} \times \frac{\partial y}{\partial s} \times \frac{\partial s}{\partial w_1}, \frac{\partial s}{\partial w_2}$$



1- partial derivative of error w.r.t. predicted output

$$\frac{\partial E}{\partial y} = \frac{\partial}{\partial y}\frac{1}{2}(d-y)^2 = y - d$$

$$\frac{\partial E}{\partial Predicted} = predicted - desired = 0.874 - 0.03$$

$$\frac{\partial E}{\partial Predicted} = 0.844$$

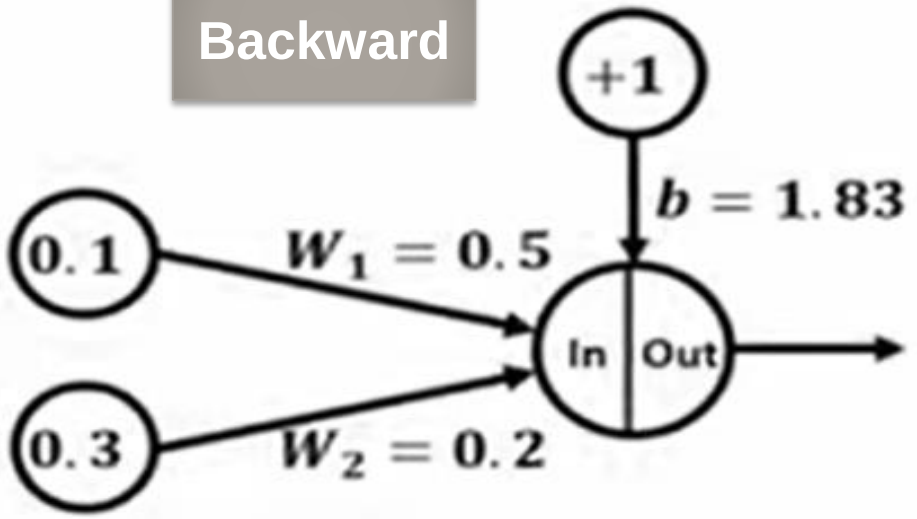
Neural Networks training : Steps

$$\frac{\partial E}{\partial W} = \frac{\partial E}{\partial y} \times \frac{\partial y}{\partial s} \times \frac{\partial s}{\partial w_1}, \frac{\partial s}{\partial w_2}$$

2- partial derivative of predicted output w.r.t. SOP

$$\frac{\partial y}{\partial s} = \frac{\partial}{\partial s}\frac{1}{1+e^{-s}} = \frac{1}{1+e^{-s}}\left(1-\frac{1}{1+e^{-s}}\right)$$

Backward



$$\frac{\partial Predicted}{\partial s} = \frac{1}{1+e^{-s}}\left(1-\frac{1}{1+e^{-s}}\right) = \frac{1}{1+e^{-1.94}}\left(1-\frac{1}{1+e^{-1.94}}\right)$$

$$= \frac{1}{1+0.144}\left(1-\frac{1}{1+0.144}\right)$$

$$= \frac{1}{1.144}\left(1-\frac{1}{1.144}\right)$$

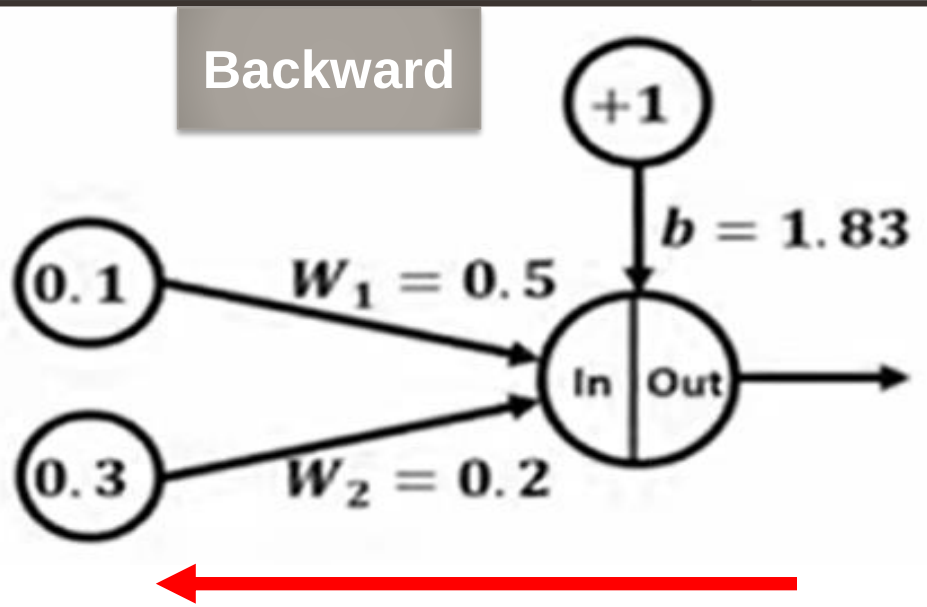
$$= 0.874(1-0.874)$$

$$= 0.874(0.126)$$

$$\frac{\partial Predicted}{\partial s} = 0.11$$

Neural Networks training : Steps

$$\frac{\partial E}{\partial W} = \frac{\partial E}{\partial y} \times \frac{\partial y}{\partial s} \times \frac{\partial s}{\partial w_1}, \frac{\partial s}{\partial w_2}$$



3- partial derivative of SOP w.r.t. W_1

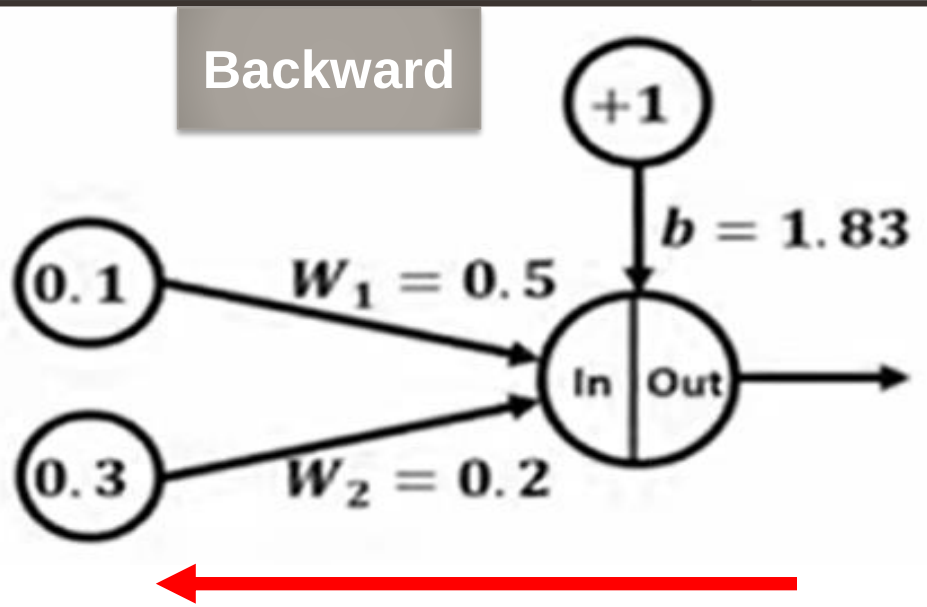
$$\frac{\partial s}{\partial w_1}, \frac{\partial s}{\partial w_2}$$

$$\frac{\partial s}{\partial W_1} = X_1$$

$$\frac{\partial s}{\partial W_1} = 0.1$$

Neural Networks training : Steps

$$\frac{\partial E}{\partial W} = \frac{\partial E}{\partial y} \times \frac{\partial y}{\partial s} \times \frac{\partial s}{\partial w_1}, \frac{\partial s}{\partial w_2}$$



3- partial derivative of SOP w.r.t. W_2

$$\frac{\partial s}{\partial w_2} = \frac{\partial}{\partial w_2} x_1 w_1 + x_2 w_2 + b = x_2$$

$$\frac{\partial s}{\partial W_2} = X_2$$

$$\frac{\partial s}{\partial W_2} = 0.3$$

Error-W_1 $(\frac{\partial E}{\partial W_1})$ Partial Derivative

- After calculating each individual derivative, we can multiply all of them to get the desired relationship between the prediction error and each weight.

Calculated Derivatives

$$\frac{\partial E}{\partial Predicted} = 0.844$$

$$\frac{\partial Predicted}{\partial s} = 0.11$$

$$\frac{\partial s}{\partial W_1} = 0.1$$

$$\frac{\partial E}{\partial W_1} = \frac{\partial E}{\partial Predicted} * \frac{\partial Predicted}{\partial s} * \frac{\partial s}{\partial W_1}$$

$$\frac{\partial E}{\partial W_1} = 0.844 * 0.11 * 0.1$$

$$\frac{\partial E}{\partial W_1} = 0.01$$

Error-W_2 $(\frac{\partial E}{\partial W_2})$ Partial Derivative

Calculated Derivatives

$$\frac{\partial E}{\partial Predicted} = 0.844$$

$$\frac{\partial Predicted}{\partial s} = 0.11$$

$$\frac{\partial s}{\partial W_2} = 0.3$$

$$\frac{\partial E}{\partial W_2} = \frac{\partial E}{\partial Predicted} * \frac{\partial Predicted}{\partial s} * \frac{\partial s}{\partial W_2}$$

$$\frac{\partial E}{\partial W_2} = 0.844 * 0.11 * 0.3$$

$$\frac{\partial E}{\partial W_2} = 0.03$$

Updating Weights

- Each weight will be updated based on its derivative according to this equation:

$$W_{inew} = W_{iold} - \eta * \frac{\partial E}{\partial W_i}$$

Updating W_1

$$W_{1new} = W_1 - \eta * \frac{\partial E}{\partial W_1}$$

$$= 0.5 - 0.01 * 0.01$$

$$W_{1new} = 0.49991$$

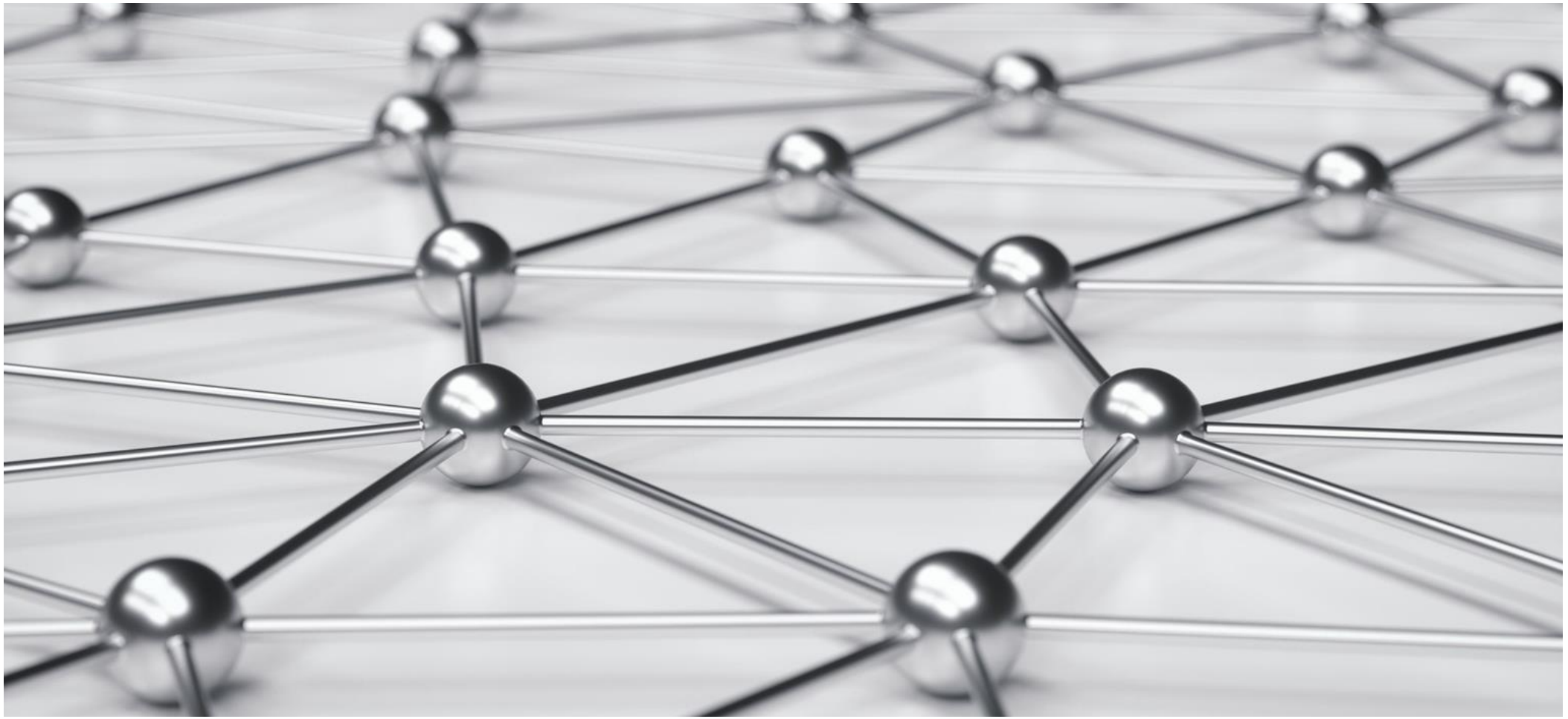
Updating W_2

$$W_{2new} = W_2 - \eta * \frac{\partial E}{\partial W_2}$$

$$= 0.2 - 0.01 * 0.028$$

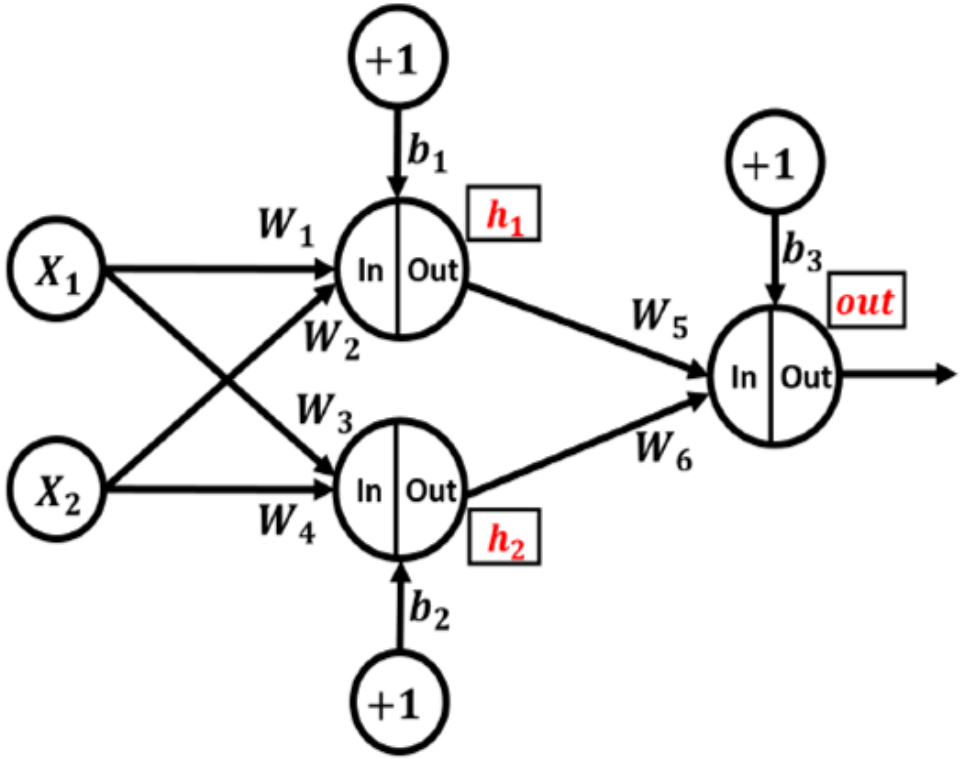
$$W_{2new} = 0.1997$$

Neural Network Training Example (Backpropagation)



Example

ANN with Hidden Layer



Training Data

X_1	X_2	Output
0.1	0.3	0.03

Initial Weights

W_1	W_2	W_3	W_4	W_5	W_6	b_1	b_2	b_3
0.5	0.1	0.62	0.2	−0.2	0.3	0.4	−0.1	1.83

Forward Pass

$$in^{h1} = \begin{pmatrix} 0.5 & 0.1 & 0.4 \\ 0.62 & 0.2 & -0.1 \end{pmatrix} \begin{bmatrix} 0.1 \\ 0.3 \\ 1 \end{bmatrix} = \begin{bmatrix} 0.48 \\ 0.022 \end{bmatrix}$$

$$\sigma(in^{h1}) = outh^{1} = \begin{bmatrix} \frac{1}{1+e^{-0.48}} \\ \frac{1}{1\_+e^{-0.022}} \end{bmatrix} = \begin{bmatrix} 0.618 \\ 0.506 \end{bmatrix}$$

$$in^{out} = \begin{bmatrix} -0.2 & 0.3 & 1.83 \end{bmatrix} \begin{bmatrix} 0.618 \\ 0.506 \\ 1 \end{bmatrix} = [1.858]$$

$$\sigma(in^{out}) = outout = \left[\frac{1}{1+e^{-1.858}}\right] = [0.865]$$

Predicted Output

Forward Pass – Hidden Layer Neurons

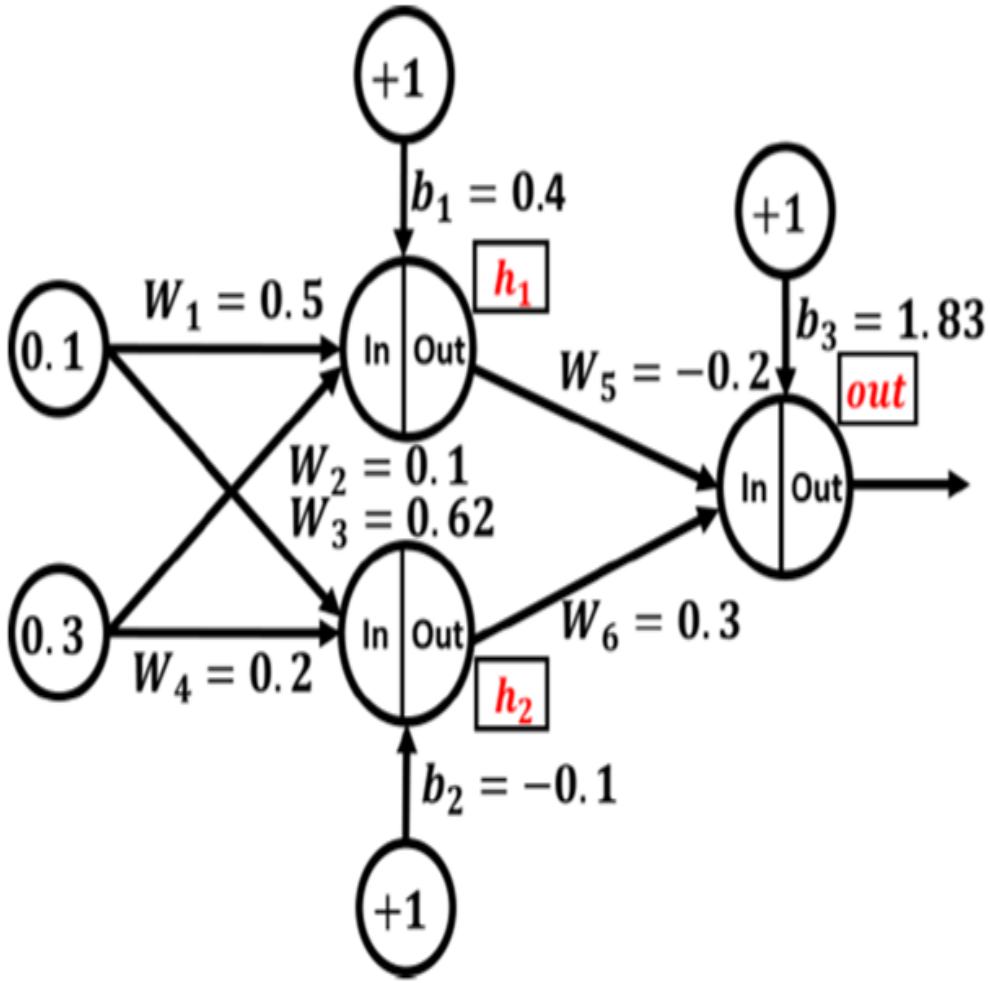
h_1

In

$$h_{1in} = X_1 * W_1 + X_2 * W_2 + b_1$$
$$= 0.1 * 0.5 + 0.3 * 0.1 + 0.4$$
$$h_{1in} = 0.48$$

Out

$$h_{1out} = \frac{1}{1 + e^{-h_{1in}}}$$
$$= \frac{1}{1 + e^{-0.48}}$$
$$h_{1out} = 0.618$$



Forward Pass – Hidden Layer Neurons

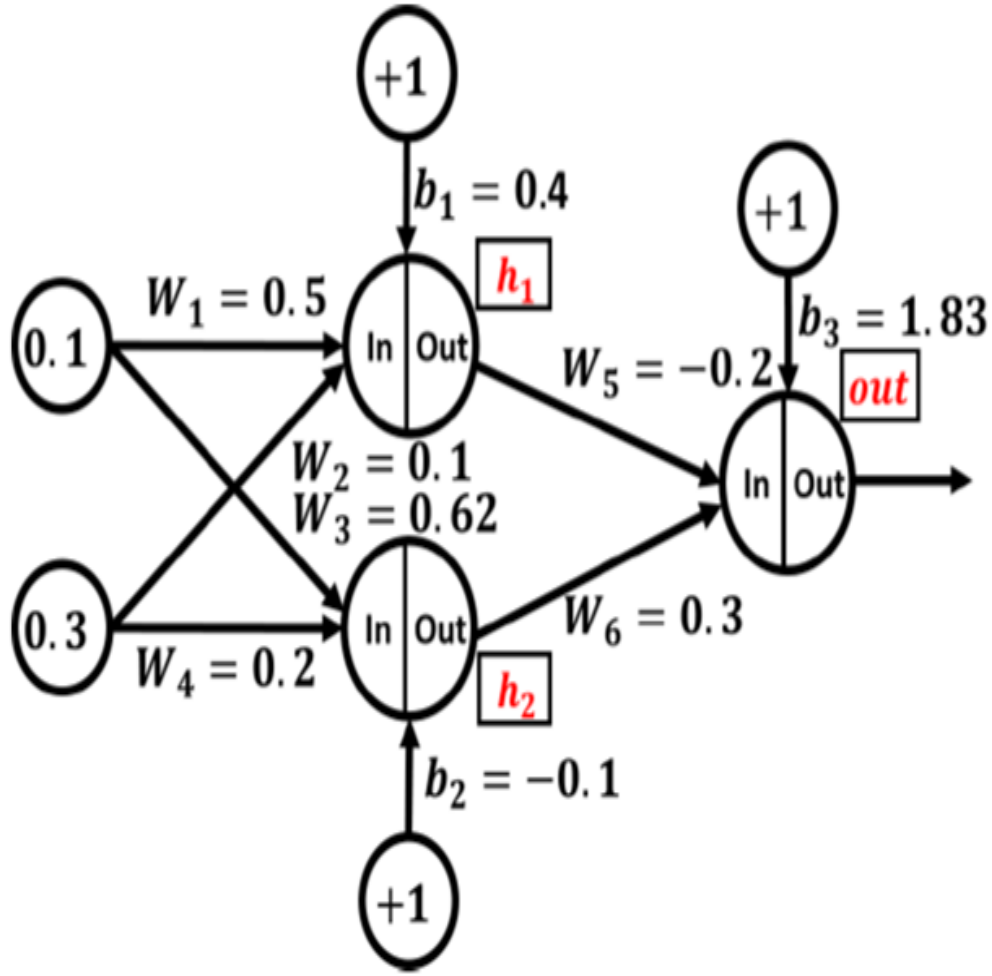
h_2

In

$$h_{2in} = X_1 * W_3 + X_2 * W_4 + b_2$$
$$= 0.1 * 0.62 + 0.3 * 0.2 - 0.1$$
$$h_{2in} = 0.022$$

Out

$$h_{2out} = \frac{1}{1 + e^{-h_{2in}}}$$
$$= \frac{1}{1 + e^{-0.022}}$$
$$h_{2out} = 0.506$$



Forward Pass – Output Layer Neuron

out

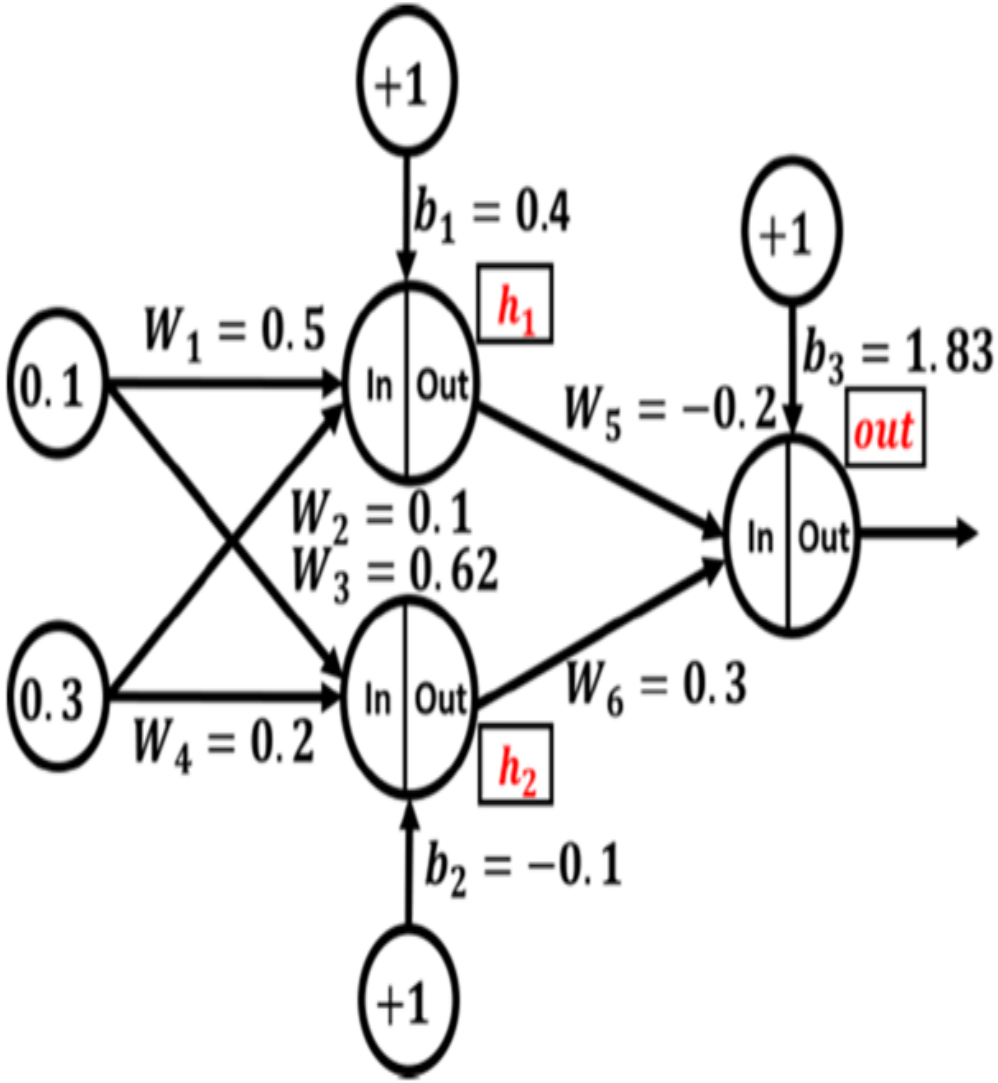
In

$$out_{in} = h_{1out} * W_5 + h_{2out} * W_6 + b_3$$
$$= 0.618 * -0.2 + 0.506 * 0.3 + 1.83$$
$$out_{in} = 1.858$$

Out

$$out_{out} = \frac{1}{1 + e^{-out_{in}}}$$
$$= \frac{1}{1 + e^{-1.858}}$$
$$out_{out} = 0.865$$

Predicted Output



Forward Pass – Prediction Error

$desired = 0.03$ $Predicted = out_{out} = 0.641$

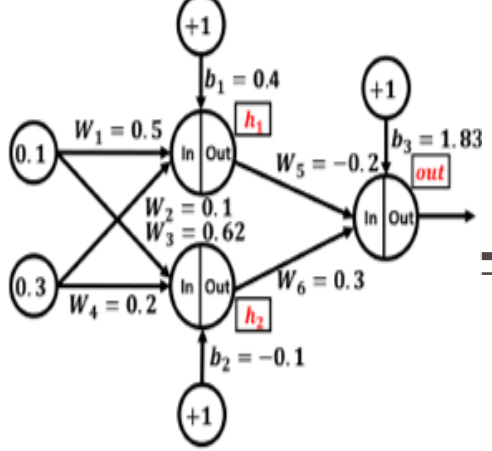
$$E = \frac{1}{2}(desired - out_{out})^2$$

$$= \frac{1}{2}(0.03 - 0.865)^2$$

$$E = 0.349$$

$$\frac{\partial E}{\partial W_1}, \frac{\partial E}{\partial W_2}, \frac{\partial E}{\partial W_3}, \frac{\partial E}{\partial W_4}, \frac{\partial E}{\partial W_5}, \frac{\partial E}{\partial W_6}$$

$E - W_5 \left(\frac{\partial E}{\partial W_5}\right)$ Parial Derivative



$$\frac{\partial E}{\partial W_5} = \frac{\partial E}{\partial out_{out}} * \frac{\partial out_{out}}{\partial out_{in}} * \frac{\partial out_{in}}{\partial W_5}$$

Partial Derivative

$$\frac{\partial E}{\partial out_{out}} = \frac{\partial}{\partial out_{out}}\left(\frac{1}{2}(desired - out_{out})^2\right)$$

$$= 2 * \frac{1}{2}(desired - out_{out})^{2-1} * (0 - 1)$$

$$= desired - out_{out} * (-1)$$

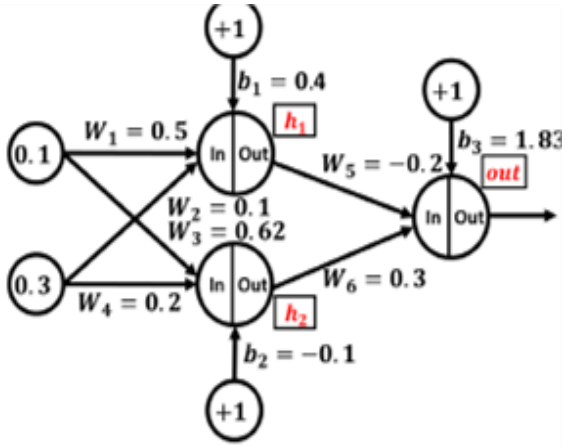
$$\frac{\partial E}{\partial out_{out}} = out_{out} - desired$$

Substitution

$$\frac{\partial E}{\partial out_{out}} = out_{out} - desired = \frac{0.86}{1} - 0.03$$

$$\frac{\partial E}{\partial out_{out}} = \mathbf{0.831}$$

E−W_5 $(\frac{\partial E}{\partial W_5})$ Parial Derivative



$$\frac{\partial E}{\partial W_5} = \frac{\partial E}{\partial out_{out}} * \frac{\partial out_{out}}{\partial out_{in}} * \frac{\partial out_{in}}{\partial W_5}$$

Partial Derivative

$$\frac{\partial out_{out}}{\partial out_{in}} = \frac{\partial}{\partial out_{in}}\left(\frac{1}{1+e^{-out_{in}}}\right)$$

$$\frac{\partial out_{out}}{\partial out_{in}} = \left(\frac{1}{1+e^{-out_{in}}}\right)\left(1-\frac{1}{1+e^{-out_{in}}}\right)$$

Substitution

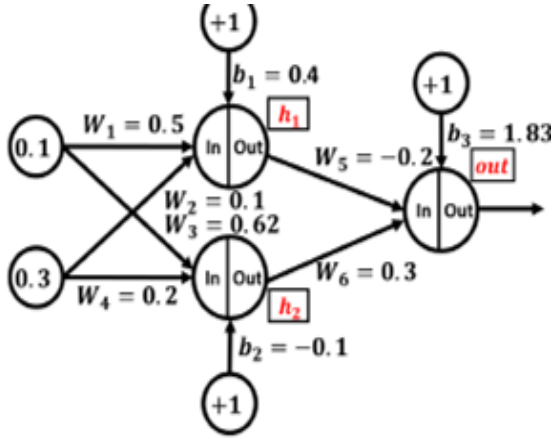
$$\frac{\partial out_{out}}{\partial out_{in}} = \left(\frac{1}{1+e^{-1.858}}\right)\left(1-\frac{1}{1+e^{-1.858}}\right)$$

$$= \left(\frac{1}{1.56}\right)\left(1-\frac{1}{1.56}\right)$$

$$= (0.641)(1-0.641) = (0.641)(0.359)$$

$$\frac{\partial out_{out}}{\partial out_{in}} = 0.23$$

E−W_5 ($\frac{\partial E}{\partial W_5}$) Parial Derivative



$$\frac{\partial E}{\partial W_5} = \frac{\partial E}{\partial out_{out}} * \frac{\partial out_{out}}{\partial out_{in}} * \frac{\partial out_{in}}{\partial W_5}$$

Partial Derivative

$$\frac{\partial out_{in}}{\partial W_5} = \frac{\partial}{\partial W_5}(h_{1out} * W_5 + h_{2out} * W_6 + b_3)$$

$$= 1 * h_{1out} * (W_5)^{1-1} + 0 + 0$$

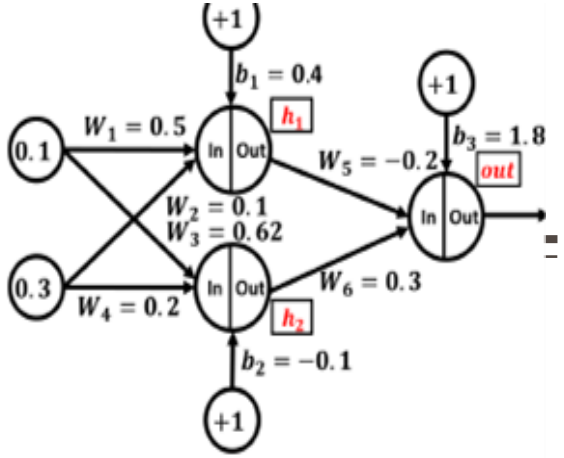
$$\frac{\partial out_{in}}{\partial W_5} = h_{1out}$$

Substitution

$$\frac{\partial out_{in}}{\partial W_5} = h_{1out}$$

$$\frac{\partial out_{in}}{\partial W_5} = 0.618$$

E−W_5 $(\frac{\partial E}{\partial W_5})$ Parial Derivative



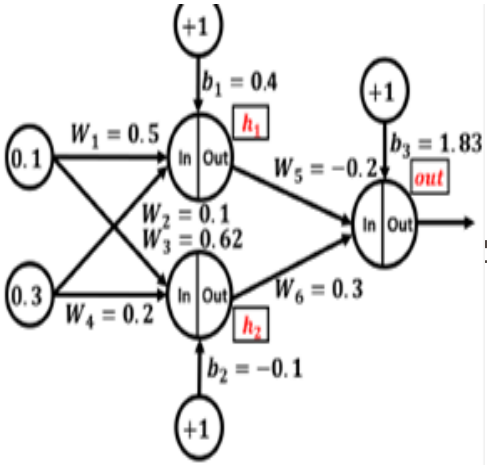
$$\frac{\partial E}{\partial W_5} = \frac{\partial E}{\partial out_{out}} * \frac{\partial out_{out}}{\partial out_{in}} * \frac{\partial out_{in}}{\partial W_5}$$

$$\frac{\partial E}{\partial out_{out}} = 0.831 \qquad \frac{\partial out_{out}}{\partial out_{in}} = 0.23 \qquad \frac{\partial out_{in}}{\partial W_5} = 0.618$$

$$\frac{\partial E}{\partial W_5} = 0.831 * 0.23 * 0.618$$

$$\frac{\partial E}{\partial W_5} = 0.119$$

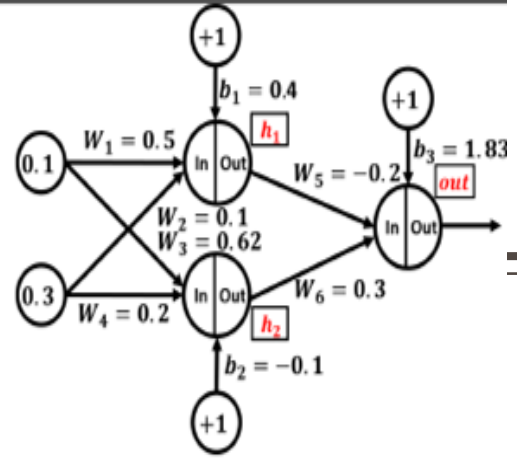
E−W_6 $\left(\frac{\partial E}{\partial W_6}\right)$ Parial Derivative



$$\frac{\partial E}{\partial W_6} = \frac{\partial E}{\partial out_{out}} * \frac{\partial out_{out}}{\partial out_{in}} * \frac{\partial out_{in}}{\partial W_6}$$

$$\frac{\partial E}{\partial out_{out}} = 0.831 \qquad \frac{\partial out_{out}}{\partial out_{in}} = 0.23$$

E−W_6 $\left(\frac{\partial E}{\partial W_6}\right)$ Parial Derivative



$$\frac{\partial E}{\partial W_5} = \frac{\partial E}{\partial out_{out}} * \frac{\partial out_{out}}{\partial out_{in}} * \frac{\partial out_{in}}{\partial W_6}$$

Partial Derivative

$$\frac{\partial out_{in}}{\partial W_6} = \frac{\partial}{\partial W_6}(h_{1out} * W_5 + h_{2out} * W_6 + b_3)$$

$$= 0 + 1 * h_{2out} * (W_6)^{1-1} + 0$$

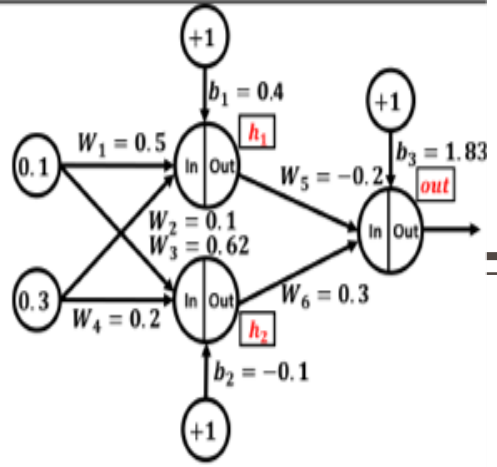
$$\frac{\partial out_{in}}{\partial W_6} = h_{2out}$$

Substitution

$$\frac{\partial out_{in}}{\partial W_6} = h_{2out}$$

$$\frac{\partial out_{in}}{\partial W_6} = 0.506$$

E−W_6 ($\frac{\partial E}{\partial W_6}$) Parial Derivative



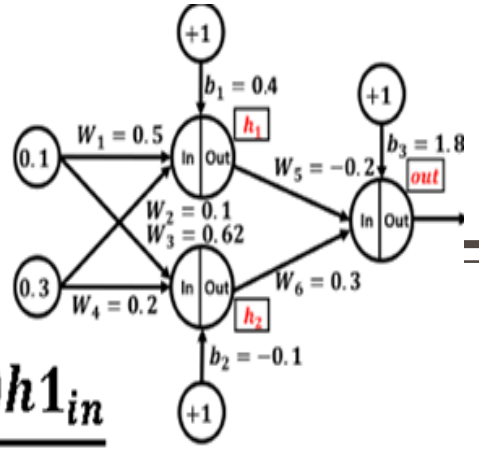
$$\frac{\partial E}{\partial W_6} = \frac{\partial E}{\partial out_{out}} * \frac{\partial out_{out}}{\partial out_{in}} * \frac{\partial out_{in}}{\partial W_6}$$

$\frac{\partial E}{\partial out_{out}} =$ 0.63 1 $\qquad \frac{\partial out_{out}}{\partial out_{in}} = 0.23 \qquad \frac{\partial out_{in}}{\partial W_6} = 0.506$

$$\frac{\partial E}{\partial W_6} = \text{0.831} * 0.23 * 0.506$$

$$\frac{\partial E}{\partial W_6} = 0.097$$

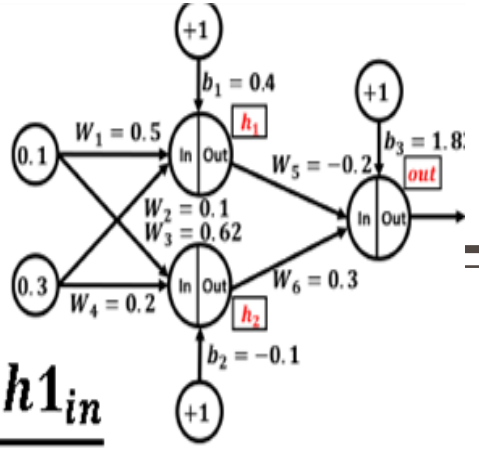
E−W_1 $\left(\frac{\partial E}{\partial W_1}\right)$ Parial Derivative



$$\frac{\partial E}{\partial W_1} = \frac{\partial E}{\partial out_{out}} * \frac{\partial out_{out}}{\partial out_{in}} * \frac{\partial out_{in}}{\partial h1_{out}} * \frac{\partial h1_{out}}{\partial h1_{in}} * \frac{\partial h1_{in}}{\partial W_1}$$

$$\frac{\partial E}{\partial out_{out}} = \text{0.831} \qquad \frac{\partial out_{out}}{\partial out_{in}} = 0.23$$

E−W_1 ($\frac{\partial E}{\partial W_1}$) Parial Derivative



$$\frac{\partial E}{\partial W_1} = \frac{\partial E}{\partial out_{out}} * \frac{\partial out_{out}}{\partial out_{in}} * \frac{\partial out_{in}}{\partial h1_{out}} * \frac{\partial h1_{out}}{\partial h1_{in}} * \frac{\partial h1_{in}}{\partial W_1}$$

Partial Derivative

$$\frac{\partial out_{in}}{\partial h1_{out}} = \frac{\partial}{\partial h1_{out}}(h_{1out} * W_5 + h_{2out} * W_6 + b_3)$$

$$= (h_{1out})^{1-1} * W_5 + 0 + 0$$

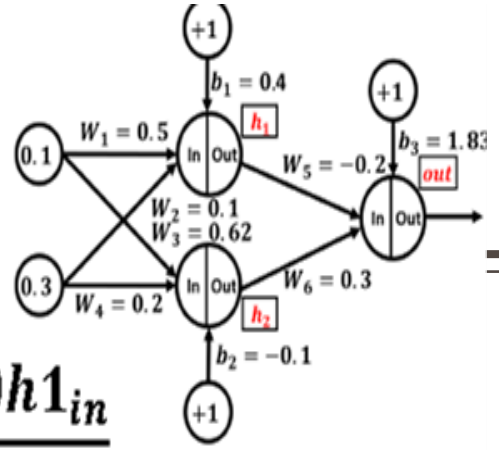
$$\frac{\partial out_{in}}{\partial h1_{out}} = W_5$$

Substitution

$$\frac{\partial out_{in}}{\partial h1_{out}} = W_5$$

$$\frac{\partial out_{in}}{\partial h1_{out}} = -0.2$$

E−W_1 $(\frac{\partial E}{\partial W_1})$ Parial Derivative



$$\frac{\partial E}{\partial W_1} = \frac{\partial E}{\partial out_{out}} * \frac{\partial out_{out}}{\partial out_{in}} * \frac{\partial out_{in}}{\partial h1_{out}} * \frac{\partial h1_{out}}{\partial h1_{in}} * \frac{\partial h1_{in}}{\partial W_1}$$

Partial Derivative

$$\frac{\partial h1_{out}}{\partial h1_{in}} = \frac{\partial}{\partial h_{1in}}\left(\frac{1}{1+e^{-h_{1in}}}\right)$$

$$\frac{\partial h1_{out}}{\partial h1_{in}} = \left(\frac{1}{1+e^{-h_{1in}}}\right)\left(1-\frac{1}{1+e^{-h_{1in}}}\right)$$

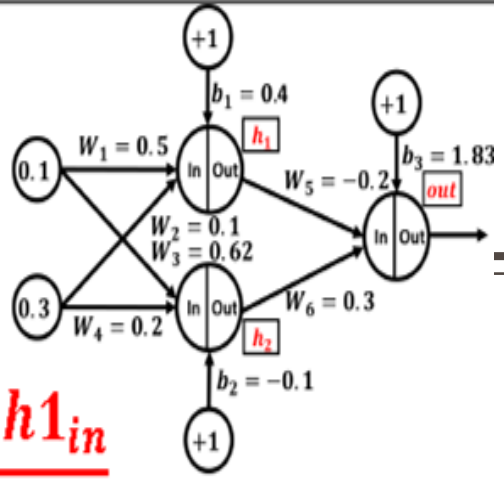
Substitution

$$\frac{\partial h1_{out}}{\partial h1_{in}} = \left(\frac{1}{1+e^{-h_{1in}}}\right)\left(1-\frac{1}{1+e^{-h_{1in}}}\right)$$

$$= \left(\frac{1}{1+e^{-0.48}}\right)\left(1-\frac{1}{1+e^{-0.48}}\right)$$

$$\frac{\partial h_{2out}}{\partial h_{2in}} = 0.236$$

E−W_1 $\left(\frac{\partial E}{\partial W_1}\right)$ Parial Derivative



$$\frac{\partial E}{\partial W_1} = \frac{\partial E}{\partial out_{out}} * \frac{\partial out_{out}}{\partial out_{in}} * \frac{\partial out_{in}}{\partial h1_{out}} * \frac{\partial h1_{out}}{\partial h1_{in}} * \frac{\partial h1_{in}}{\partial W_1}$$

Partial Derivative

$$\frac{\partial h1_{in}}{\partial W_1} = \frac{\partial}{\partial W_1}(X_1 * W_1 + X_2 * W_2 + b_1)$$

$$= X_1 * (W_1)^{1-1} + 0 + 0$$

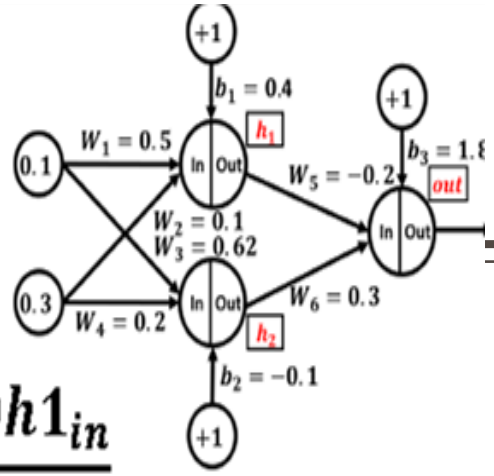
$$\frac{\partial h1_{in}}{\partial W_1} = X_1$$

Substitution

$$\frac{\partial h1_{in}}{\partial W_1} = X_1$$

$$\frac{\partial h1_{in}}{\partial W_1} = 0.1$$

$E - W_1$ $\left(\frac{\partial E}{\partial W_1}\right)$ Parial Derivative



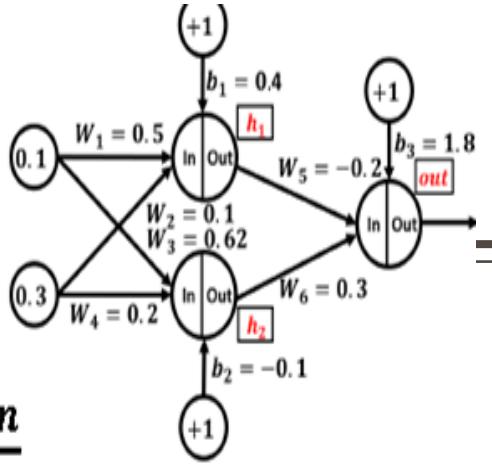
$$\frac{\partial E}{\partial W_1} = \frac{\partial E}{\partial out_{out}} * \frac{\partial out_{out}}{\partial out_{in}} * \frac{\partial out_{in}}{\partial h1_{out}} * \frac{\partial h1_{out}}{\partial h1_{in}} * \frac{\partial h1_{in}}{\partial W_1}$$

$\frac{\partial E}{\partial out_{out}}$: **0.831** $\quad \frac{\partial out_{out}}{\partial out_{in}} = 0.23 \quad \frac{\partial out_{in}}{\partial h1_{out}} = -0.2 \quad \frac{\partial h_{2out}}{\partial h_{2in}} = 0.236 \quad \frac{\partial h1_{in}}{\partial W_1} = 0.1$

$$\frac{\partial E}{\partial W_1} = 0.831 * 0.23 * -0.2 * 0.236 * 0.1$$

$$\frac{\partial E}{\partial W_1} = -0.001$$

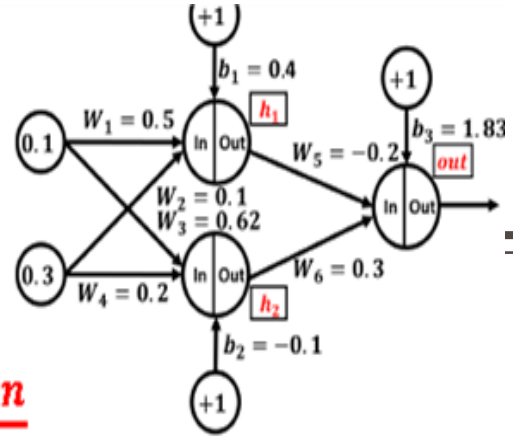
$E - W_2 \left(\frac{\partial E}{\partial W_2}\right)$ Parial Derivative:



$$\frac{\partial E}{\partial W_2} = \frac{\partial E}{\partial out_{out}} * \frac{\partial out_{out}}{\partial out_{in}} * \frac{\partial out_{in}}{\partial h1_{out}} * \frac{\partial h1_{out}}{\partial h1_{in}} * \frac{\partial h1_{in}}{\partial W_2}$$

$$\frac{\partial E}{\partial out_{out}} = \mathbf{0.831} \quad \frac{\partial out_{out}}{\partial out_{in}} = 0.23 \quad \frac{\partial out_{in}}{\partial h1_{out}} = -0.2 \quad \frac{\partial h_{2out}}{\partial h_{2in}} = 0.236$$

E−W_2 ($\frac{\partial E}{\partial W_2}$) Parial Derivative:



$$\frac{\partial E}{\partial W_2} = \frac{\partial E}{\partial out_{out}} * \frac{\partial out_{out}}{\partial out_{in}} * \frac{\partial out_{in}}{\partial h1_{out}} * \frac{\partial h1_{out}}{\partial h1_{in}} * \frac{\partial h1_{in}}{\partial W_2}$$

Partial Derivative

$$\frac{\partial h1_{in}}{\partial W_2} = \frac{\partial}{\partial W_2}(X_1 * W_1 + X_2 * W_2 + b_1)$$

$$= 0 + X_2 * (W_2)^{1-1} + 0$$

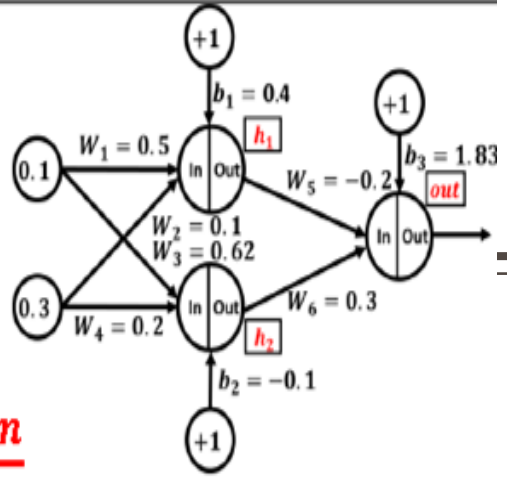
$$\frac{\partial h1_{in}}{\partial W_2} = X_2$$

Substitution

$$\frac{\partial h1_{in}}{\partial W_2} = X_2$$

$$\frac{\partial h1_{in}}{\partial W_2} = 0.3$$

E−W_2 $\left(\frac{\partial E}{\partial W_2}\right)$ Parial Derivative:



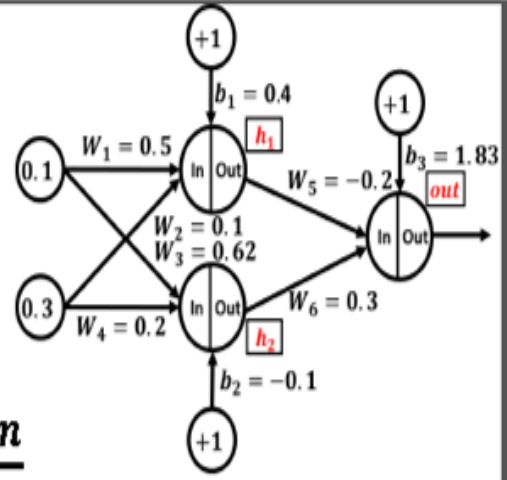
$$\frac{\partial E}{\partial W_2} = \frac{\partial E}{\partial out_{out}} * \frac{\partial out_{out}}{\partial out_{in}} * \frac{\partial out_{in}}{\partial h1_{out}} * \frac{\partial h1_{out}}{\partial h1_{in}} * \frac{\partial h1_{in}}{\partial W_2}$$

$$\frac{\partial E}{\partial out_{out}} = 0.831 \quad \frac{\partial out_{out}}{\partial out_{in}} = 0.23 \quad \frac{\partial out_{in}}{\partial h1_{out}} = -0.2 \quad \frac{\partial h_{2out}}{\partial h_{2in}} = 0.236 \quad \frac{\partial h1_{in}}{\partial W_2} = 0.3$$

$$\frac{\partial E}{\partial W_2} = 0.831 \quad 0.23 * -0.2 * 0.236 * 0.3$$

$$\frac{\partial E}{\partial W_2} = -.003$$

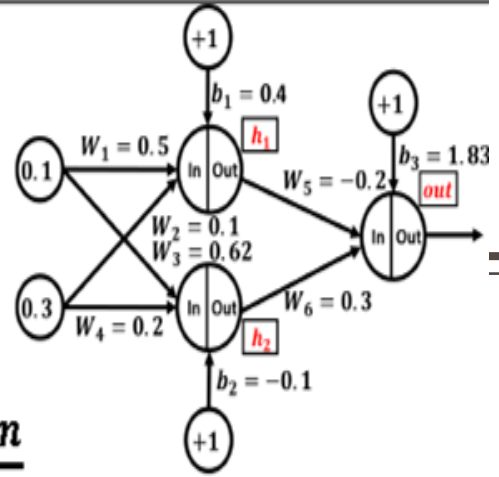
E−W_3 $\left(\frac{\partial E}{\partial W_3}\right)$ Parial Derivative:



$$\frac{\partial E}{\partial W_3} = \frac{\partial E}{\partial out_{out}} * \frac{\partial out_{out}}{\partial out_{in}} * \frac{\partial out_{in}}{\partial h2_{out}} * \frac{\partial h2_{out}}{\partial h2_{in}} * \frac{\partial h2_{in}}{\partial W_3}$$

$$\frac{\partial E}{\partial out_{out}} = 0.831 \qquad \frac{\partial out_{out}}{\partial out_{in}} = 0.23$$

$E - W_3 \left(\frac{\partial E}{\partial W_3}\right)$ Parial Derivative:



$$\frac{\partial E}{\partial W_3} = \frac{\partial E}{\partial out_{out}} * \frac{\partial out_{out}}{\partial out_{in}} * \frac{\partial out_{in}}{\partial h2_{out}} * \frac{\partial h2_{out}}{\partial h2_{in}} * \frac{\partial h2_{in}}{\partial W_3}$$

Partial Derivative

$$\frac{\partial out_{in}}{\partial h2_{out}} = \frac{\partial}{\partial h2_{out}}(h_{1out} * W_5 + h_{2out} * W_6 + b_3)$$

$$= 0 + (h_{2out})^{1-1} * W_6 + 0$$

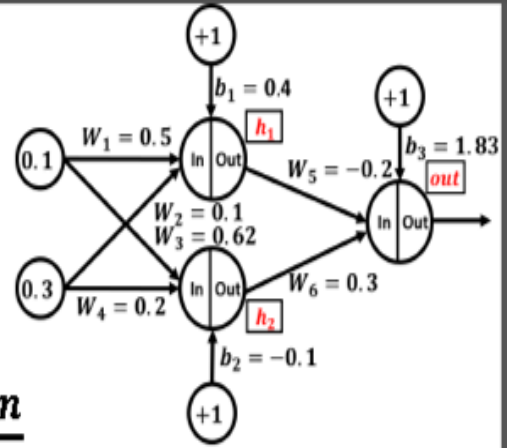
$$\frac{\partial out_{in}}{\partial h2_{out}} = W_6$$

Substitution

$$\frac{\partial out_{in}}{\partial h2_{out}} = W_6$$

$$\frac{\partial out_{in}}{\partial h2_{out}} = 0.3$$

E−W_3 $(\frac{\partial E}{\partial W_3})$ Parial Derivative:



$$\frac{\partial E}{\partial W_3} = \frac{\partial E}{\partial out_{out}} * \frac{\partial out_{out}}{\partial out_{in}} * \frac{\partial out_{in}}{\partial h2_{out}} * \frac{\partial h2_{out}}{\partial h2_{in}} * \frac{\partial h2_{in}}{\partial W_3}$$

Partial Derivative

$$\frac{\partial h2_{out}}{\partial h2_{in}} = \frac{\partial}{\partial h_{2in}}\left(\frac{1}{1+e^{-h_{2in}}}\right)$$

$$\frac{\partial h2_{out}}{\partial h2_{in}} = \left(\frac{1}{1+e^{-h_{2in}}}\right)\left(1-\frac{1}{1+e^{-h_{2in}}}\right)$$

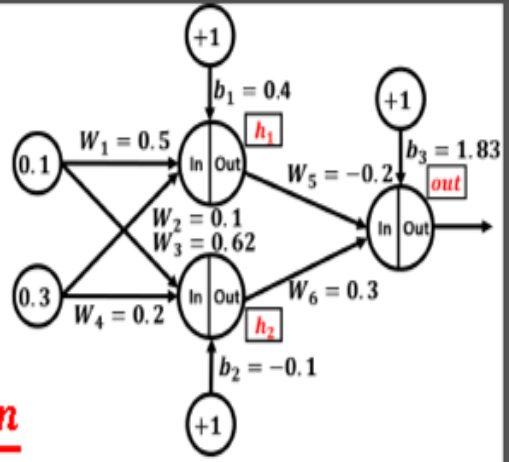
Substitution

$$\frac{\partial h2_{out}}{\partial h2_{in}} = \left(\frac{1}{1+e^{-h_{2in}}}\right)\left(1-\frac{1}{1+e^{-h_{2in}}}\right)$$

$$= \left(\frac{1}{1+e^{-0.022}}\right)\left(1-\frac{1}{1+e^{-0.022}}\right)$$

$$\frac{\partial h_{2out}}{\partial h_{2in}} = 0.25$$

E−W_3 ($\frac{\partial E}{\partial W_3}$) Parial Derivative:



$$\frac{\partial E}{\partial W_3} = \frac{\partial E}{\partial out_{out}} * \frac{\partial out_{out}}{\partial out_{in}} * \frac{\partial out_{in}}{\partial h2_{out}} * \frac{\partial h2_{out}}{\partial h2_{in}} * \frac{\partial h2_{in}}{\partial W_3}$$

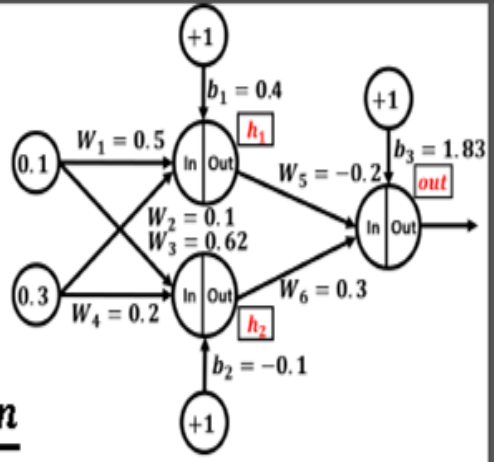
Partial Derivative

$$\frac{\partial h2_{in}}{\partial W_3} = \frac{\partial}{\partial W_3}(X_1 * W_3 + X_2 * W_4 + b_2)$$
$$= X_1 * W_3 + X_2 * W_4 + b_2$$
$$= (X_1)^{1-1} * W_3 + 0 + 0$$
$$\frac{\partial h2_{in}}{\partial W_3} = W_3$$

Substitution

$$\frac{\partial h2_{in}}{\partial W_3} = W_3$$
$$\frac{\partial h2_{in}}{\partial W_3} = 0.62$$

$E - W_3 \left(\frac{\partial E}{\partial W_3}\right)$ Parial Derivative:



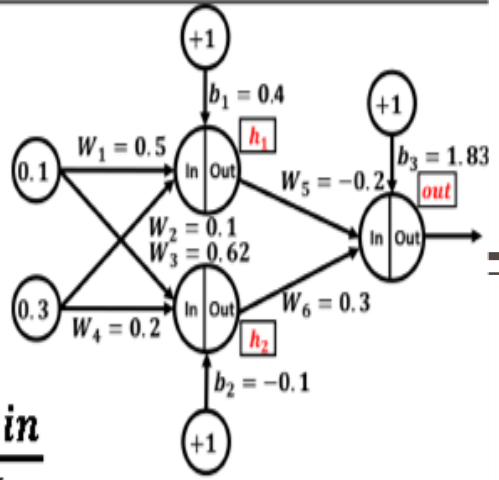
$$\frac{\partial E}{\partial W_3} = \frac{\partial E}{\partial out_{out}} * \frac{\partial out_{out}}{\partial out_{in}} * \frac{\partial out_{in}}{\partial h2_{out}} * \frac{\partial h2_{out}}{\partial h2_{in}} * \frac{\partial h2_{in}}{\partial W_3}$$

$\frac{\partial E}{\partial out_{out}} = 0.631$ $\quad \frac{\partial out_{out}}{\partial out_{in}} = 0.23$ $\quad \frac{\partial out_{in}}{\partial h2_{out}} = 0.3$ $\quad \frac{\partial h_{2out}}{\partial h_{2in}} = 0.25$ $\quad \frac{\partial h2_{in}}{\partial W_3} = 0.62$

$$\frac{\partial E}{\partial W_3} = 0.831 \quad 0.23 * 0.3 * 0.25 * 0.62$$

$$\frac{\partial E}{\partial W_3} = 0.009$$

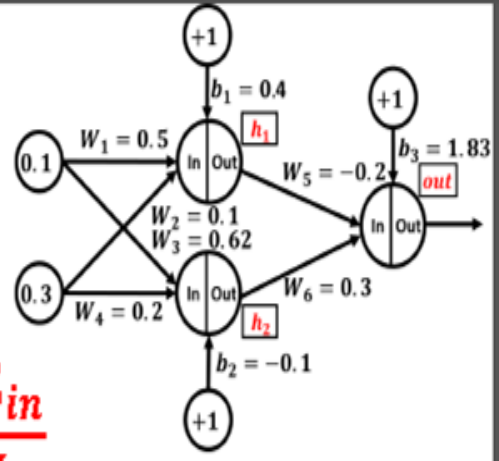
E−W_4 $\left(\frac{\partial E}{\partial W_4}\right)$ Parial Derivative:



$$\frac{\partial E}{\partial W_4} = \frac{\partial E}{\partial out_{out}} * \frac{\partial out_{out}}{\partial out_{in}} * \frac{\partial out_{in}}{\partial h2_{out}} * \frac{\partial h2_{out}}{\partial h2_{in}} * \frac{\partial h2_{in}}{\partial W_4}$$

$$\frac{\partial E}{\partial out_{out}} = \mathbf{0.831} \quad \frac{\partial out_{out}}{\partial out_{in}} = 0.23 \quad \frac{\partial out_{in}}{\partial h2_{out}} = 0.3 \quad \frac{\partial h_{2out}}{\partial h_{2in}} = 0.25$$

E−W_4 $\left(\frac{\partial E}{\partial W_4}\right)$ Parial Derivative:



$$\frac{\partial E}{\partial W_4} = \frac{\partial E}{\partial out_{out}} * \frac{\partial out_{out}}{\partial out_{in}} * \frac{\partial out_{in}}{\partial h2_{out}} * \frac{\partial h2_{out}}{\partial h2_{in}} * \frac{\partial h2_{in}}{\partial W_4}$$

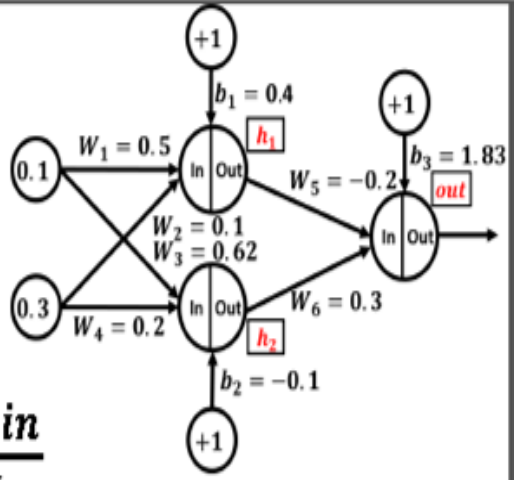
Partial Derivative

$$\frac{\partial h2_{in}}{\partial W_4} = \frac{\partial}{\partial W_4}(X_1 * W_3 + X_2 * W_4 + b_2)$$
$$= X_1 * W_3 + X_2 * W_4 + b_2$$
$$= 0 + (X_2)^{1-1} * W_4 + 0$$
$$\frac{\partial h2_{in}}{\partial W_4} = W_4$$

Substitution

$$\frac{\partial h2_{in}}{\partial W_4} = W_4$$
$$\frac{\partial h2_{in}}{\partial W_4} = 0.2$$

E$-W_4$ $\left(\frac{\partial E}{\partial W_4}\right)$ Parial Derivative:



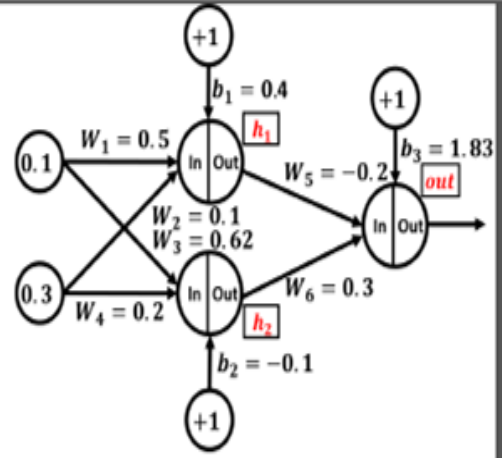
$$\frac{\partial E}{\partial W_4} = \frac{\partial E}{\partial out_{out}} * \frac{\partial out_{out}}{\partial out_{in}} * \frac{\partial out_{in}}{\partial h2_{out}} * \frac{\partial h2_{out}}{\partial h2_{in}} * \frac{\partial h2_{in}}{\partial W_4}$$

$\frac{\partial E}{\partial out_{out}} =$ **0.831** $\quad \frac{\partial out_{out}}{\partial out_{in}} = 0.23 \quad \frac{\partial out_{in}}{\partial h2_{out}} = 0.3 \quad \frac{\partial h_{2out}}{\partial h_{2in}} = 0.25 \quad \frac{\partial h2_{in}}{\partial W_4} = 0.2$

$$\frac{\partial E}{\partial W_4} = 0.631 * 0.23 * 0.3 * 0.25 * 0.2$$

$$\frac{\partial E}{\partial W_4} = 0.003$$

All Error-Weights Partial Derivatives



$$\frac{\partial E}{\partial W_1} = -0.001 \qquad \frac{\partial E}{\partial W_2} = -.003$$

$$\frac{\partial E}{\partial W_3} = 0.009 \qquad \frac{\partial E}{\partial W_4} = 0.003$$

$$\frac{\partial E}{\partial W_5} = 0.119 \qquad \frac{\partial E}{\partial W_6} = 0.097$$

Updated Weights

$$W_{1new} = W_1 - \eta * \frac{\partial E}{\partial W_1} = 0.5 - 0.01 * -0.001 = 0.50001$$

$$W_{2new} = W_2 - \eta * \frac{\partial E}{\partial W_2} = 0.1 - 0.01 * -0.003 = 0.10003$$

$$W_{3new} = W_3 - \eta * \frac{\partial E}{\partial W_3} = 0.62 - 0.01 * 0.009 = 0.61991$$

$$W_{4new} = W_4 - \eta * \frac{\partial E}{\partial W_4} = 0.2 - 0.01 * 0.003 = 0.1997$$

$$W_{5new} = W_5 - \eta * \frac{\partial E}{\partial W_5} = -0.2 - 0.01 * 0.618 = -0.20618$$

$$W_{6new} = W_6 - \eta * \frac{\partial E}{\partial W_6} = 0.3 - 0.01 * 0.097 = 0.29903$$

Continue updating weights according to derivatives and re-train the network until reaching an acceptable error.