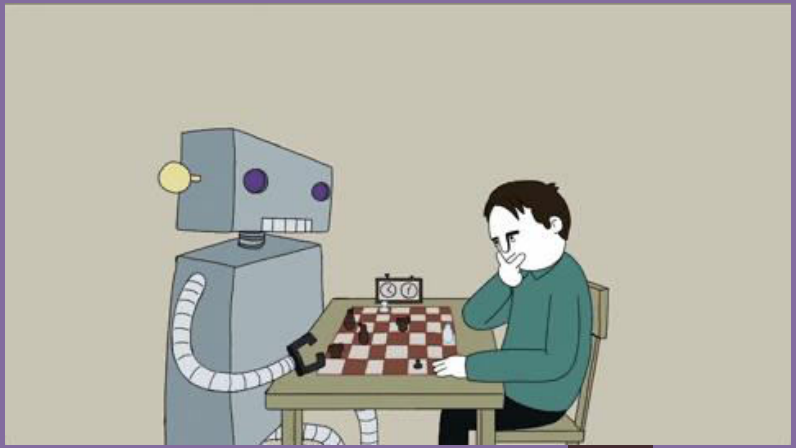
# CSE4006 DEEP LEARNING



**Dr K G Suma**
Associate Professor
School  of Computer Science and Engineering

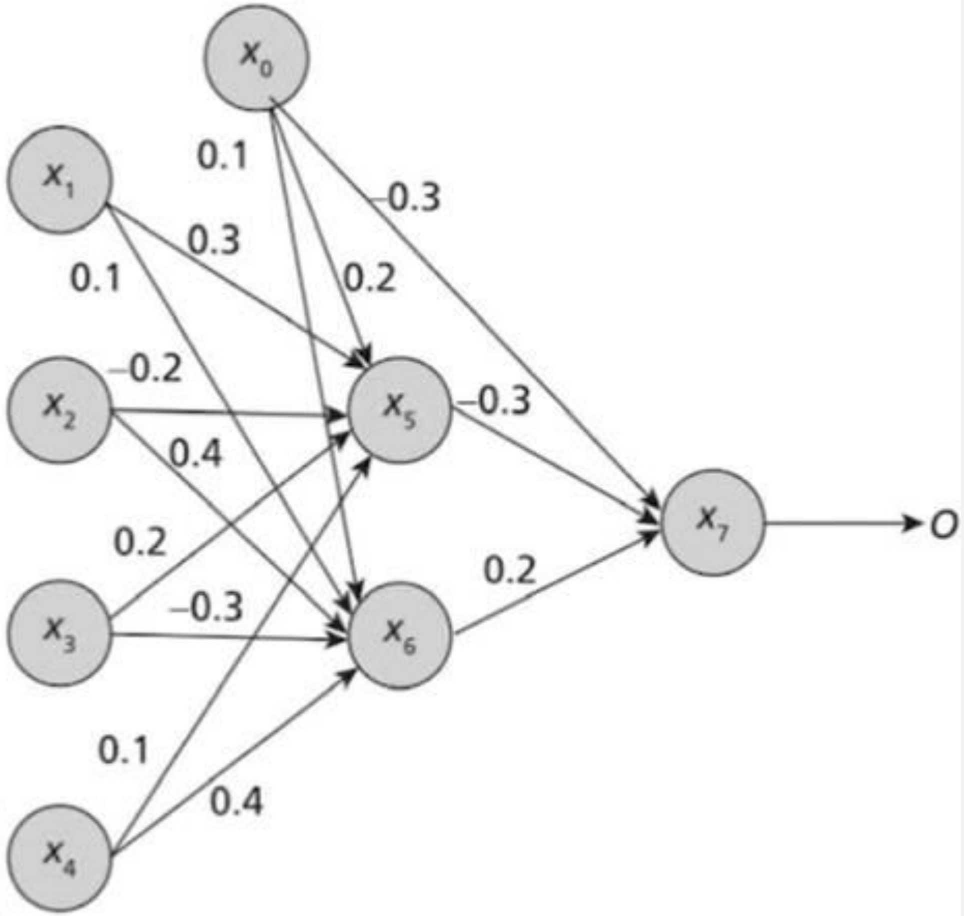
# Module No. 2
# Practical Deep Networks
# 8 Hours

- Multilayer Perceptron
- Gradient based Learning
- Backpropagation Algorithm
- Regularization for Deep Learning
- Optimization for training deep models

# Multi-Layer Perceptron Learning – Solved Example

- The given MLP consists of an Input layer, one Hidden layer and an Output layer.
- The input layer has 4 neurons, the hidden layer has 2 neurons and the output layer has a single neuron.
- Train the MLP by updating the weights and biases in the network.
- Learning Rate = 0.8

| $x_1$ | $x_2$ | $x_3$ | $x_4$ | $O_{Desired}$ |
| --- | --- | --- | --- | --- |
| 1 | 1 | 0 | 1 | 1 |



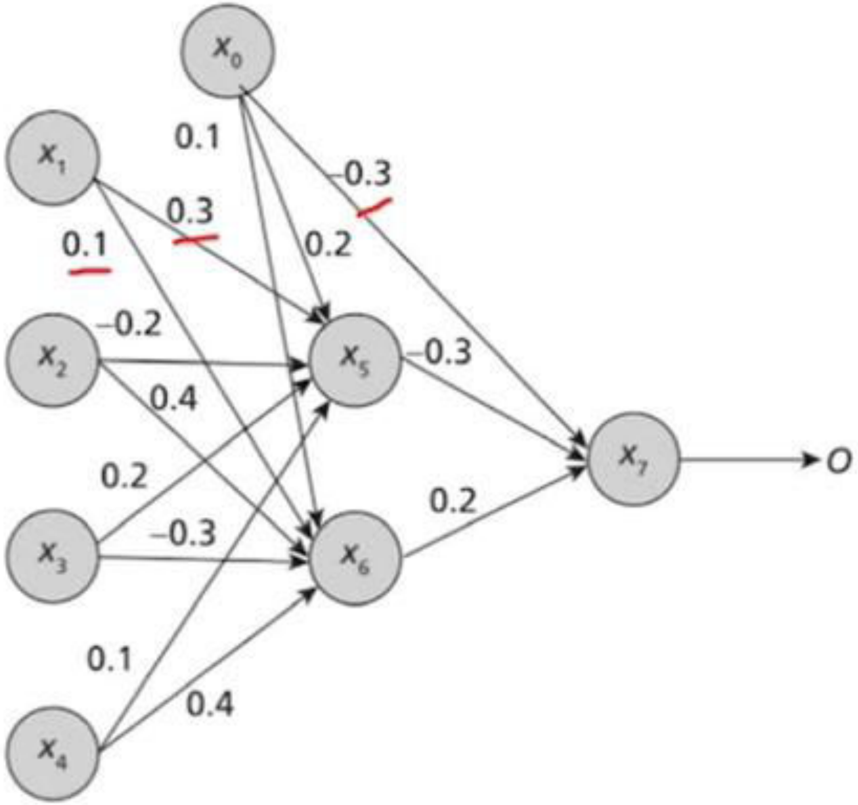
# Multi-Layer Perceptron Learning – Solved Example

| $x_1$ | $x_2$ | $x_3$ | $x_4$ | $w_{15}$ | $w_{16}$ | $w_{25}$ | $w_{26}$ | $w_{35}$ | $w_{36}$ | $w_{45}$ | $w_{46}$ | $w_{57}$ | $w_{67}$ | $\theta_5$ | $\theta_6$ | $\theta_7$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 1 | 0.3 | 0.1 | –0.2 | 0.4 | 0.2 | –0.3 | 0.1 | 0.4 | –0.3 | 0.2 | 0.2 | 0.1 | –0.3 |

1. Calculate Input and Output in the Input Layer

Net Input and Output Calculation

| Input layer | $I_j$ | $O_j$ |
|---|---|---|
| $x_1$ | 1 | 1 |
| $x_2$ | 1 | 1 |
| $x_3$ | 0 | 0 |
| $x_4$ | 1 | 1 |



# Multi-Layer Perceptron Learning – Solved Example

| $x_1$ | $x_2$ | $x_3$ | $x_4$ | $w_{15}$ | $w_{16}$ | $w_{25}$ | $w_{26}$ | $w_{35}$ | $w_{36}$ | $w_{45}$ | $w_{46}$ | $w_{57}$ | $w_{67}$ | $\theta_5$ | $\theta_6$ | $\theta_7$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 1 | 0.3 | 0.1 | –0.2 | 0.4 | 0.2 | –0.3 | 0.1 | 0.4 | –0.3 | 0.2 | 0.2 | 0.1 | –0.3 |

2. Calculate Net Input and Output in the Hidden Layer and Output Layer as

| Unit$_j$ | Net Input $I_j$ | Output $O_j$ |
|---|---|---|
| $x_5$ | $I_5 = x_1 \times w_{15} + x_2 \times w_{25} + x_3 \times w_{35} + x_4 \times w_{45} + x_0 \times \theta_5$ <br> $I_5 = 1 \times 0.3 + 1 \times -0.2 + 0 \times 0.2 + 1 \times 0.1 + 1 \times 0.2 = 0.4$ | $O_5 = \frac{1}{1+e^{-I_5}} \quad \frac{1}{1+e^{-0.4}} = 0.599$ |
| $x_6$ | $I_6 = x_1 \times w_{15} + x_2 \times w_{26} + x_3 \times w_{36} + x_4 \times w_{46} + x_0 \times \theta_6$ <br> $I_6 = 1 \times 0.3 + 1 \times 0.4 + 0 \times -0.3 + 1 \times 0.4 + 1 \times 0.1 = 1.2$ | $O_6 = \frac{1}{1+e^{-I_6}} \quad \frac{1}{1+e^{-1.2}} = 0.769$ |
| $x_7$ | $I_7 = O_5 \times w_{57} + O_6 \times w_{67} + x_0 \times \theta_7$ <br> $I_7 = 0.599 \times -0.3 + 0.769 \times 0.2 + 1 \times -0.3 = -0.326$ | $O_7 = \frac{1}{1+e^{-I_7}} \quad \frac{1}{1+e^{0.326}} = 0.419$ |

# Multi-Layer Perceptron Learning – Solved Example

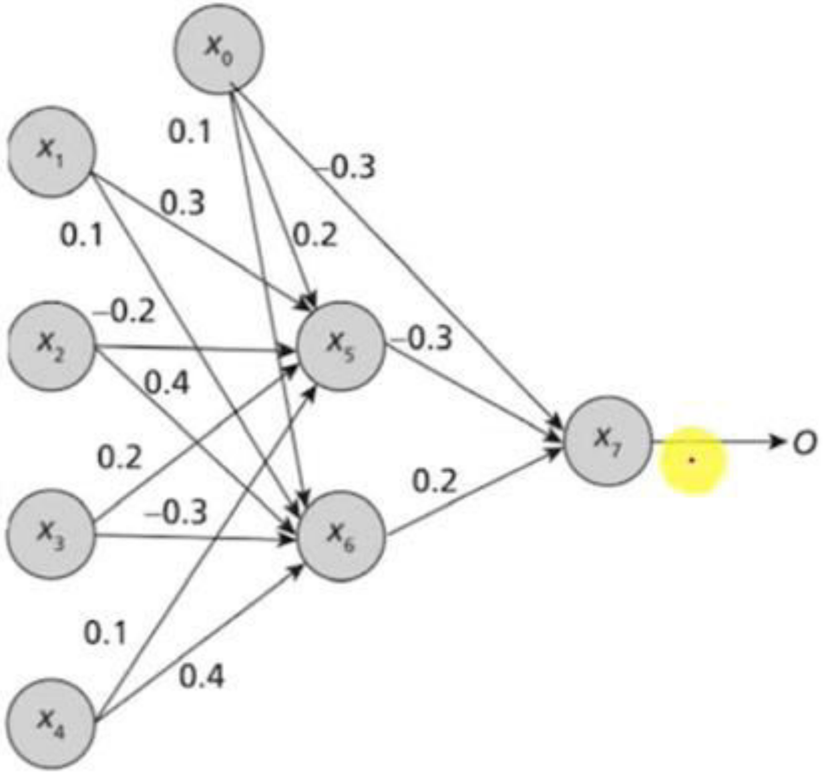
| $x_1$ | $x_2$ | $x_3$ | $x_4$ | $w_{15}$ | $w_{16}$ | $w_{25}$ | $w_{26}$ | $w_{35}$ | $w_{36}$ | $w_{45}$ | $w_{46}$ | $w_{57}$ | $w_{67}$ | $\theta_5$ | $\theta_6$ | $\theta_7$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 1 | 0.3 | 0.1 | –0.2 | 0.4 | 0.2 | –0.3 | 0.1 | 0.4 | –0.3 | 0.2 | 0.2 | 0.1 | –0.3 |

3. Calculate Error $= O_{desired} - O_{Estimated}$

   So, error for this network is:

   $\text{Error} = O_{desired} - O_7 = 1 - 0.419 = 0.581$

   So, we need to back propagate to reduce the error.



# Multi-Layer Perceptron Learning – Solved Example

| $x_1$ | $x_2$ | $x_3$ | $x_4$ | $w_{15}$ | $w_{16}$ | $w_{25}$ | $w_{26}$ | $w_{35}$ | $w_{36}$ | $w_{45}$ | $w_{46}$ | $w_{57}$ | $w_{67}$ | $\theta_5$ | $\theta_6$ | $\theta_7$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 1 | 0.3 | 0.1 | –0.2 | 0.4 | 0.2 | –0.3 | 0.1 | 0.4 | –0.3 | 0.2 | 0.2 | 0.1 | –0.3 |

**Step 2: Backward Propagation**

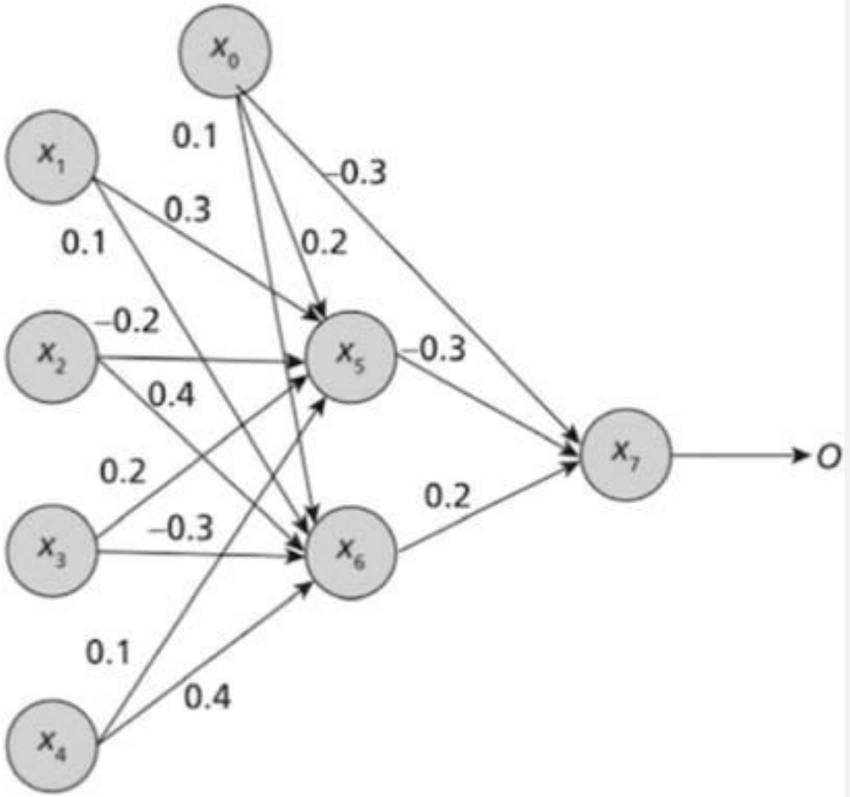
1. Calculate Error at each node

   For each unit $k$ in the output layer, calculate:

   $\text{Error}_k = O_k(1 - O_k)(O_{desired} - O_k)$

   For each unit $j$ in the hidden layer, calculate:

   $\text{Error}_j = O_j(1 - O_j) \sum_k \text{Error}_k \, w_{jk}$



# Multi-Layer Perceptron Learning – Solved Example

| $x_1$ | $x_2$ | $x_3$ | $x_4$ | $w_{15}$ | $w_{16}$ | $w_{25}$ | $w_{26}$ | $w_{35}$ | $w_{36}$ | $w_{45}$ | $w_{46}$ | $w_{57}$ | $w_{67}$ | $\theta_5$ | $\theta_6$ | $\theta_7$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 1 | 0.3 | 0.1 | –0.2 | 0.4 | 0.2 | –0.3 | 0.1 | 0.4 | –0.3 | 0.2 | 0.2 | 0.1 | –0.3 |

Error Calculation for Each Unit in the Output Layer and Hidden Layer

| For Output Layer Unit$_k$ | Error$_k$ |
|---|---|
| $x_7$ | $\text{Error}_7 = O_7(1-O_7)(Y_n - O_7)$ <br> $= 0.419 \times (1-0.419) \times (1-0.419) = 0.141$ |
| **For Hidden Layer Unit$_j$** | **Error$_j$** |
| $x_6$ | $\text{Error}_6 = O_6(1-O_6)\sum_k \text{Error}_k\, w_{jk} = O_6(1-O_6)\,\text{Error}_7\, w_{67}$ <br> $= 0.769\,(1-0.769) \times 0.2 \times 0.141 = 0.005$ |
| $x_5$ | $\text{Error}_5 = O_5(1-O_5)\sum_k \text{Error}_k\, w_{jk} = O_5(1-O_5)\,\text{Error}_7\, w_{57}$ <br> $= 0.599(1-0.599) \times 0.141 \times -0.3 = -0.0101$ |

# Multi-Layer Perceptron Learning – Solved Example

| $x_1$ | $x_2$ | $x_3$ | $x_4$ | $w_{15}$ | $w_{16}$ | $w_{25}$ | $w_{26}$ | $w_{35}$ | $w_{36}$ | $w_{45}$ | $w_{46}$ | $w_{57}$ | $w_{67}$ | $\theta_5$ | $\theta_6$ | $\theta_7$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 1 | 0.3 | 0.1 | –0.2 | 0.4 | 0.2 | –0.3 | 0.1 | 0.4 | –0.3 | 0.2 | 0.2 | 0.1 | –0.3 |

2. Update weight using the below formula:

   Learning rate $\alpha = 0.8$.

   $w_{ij} = w_{ij} + \propto \times \text{Error}_j \times O_i$

   The updated weights and bias

| $w_{ij}$ | $w_{ij} = w_{ij} + \propto \times \text{Error}_j \times O_i$ | New Weight |
|---|---|---|
| $w_{15}$ | $w_{15} = w_{15} + 0.8 \times \text{Error}_5 \times O_1$ <br> $= 0.3 + 0.8 \times -0.0101 \times 1$ | 0.292 |
| $w_{16}$ | $w_{16} = w_{16} + 0.8 \times \text{Error}_6 \times O_1$ <br> $= 0.1 + 0.8 \times 0.005 \times 1$ | 0.104 |
| $w_{25}$ | $w_{25} = w_{25} + 0.8 \times \text{Error}_5 \times O_2$ <br> $= -0.2 + 0.8 \times -0.0101 \times 1$ | –0.208 |
| $w_{26}$ | $w_{26} = w_{26} + 0.8 \times \text{Error}_6 \times O_2$ <br> $= 0.4 + 0.8 \times 0.005 \times 1$ | 0.404 |

| $w_{ij}$ | $w_{ij} = w_{ij} + \propto \times \text{Error}_j \times O_i$ | New Weight |
|---|---|---|
| $w_{35}$ | $w_{35} = w_{35} + 0.8 \times \text{Error}_5 \times O_3$ <br> $= 0.2 + 0.8 \times -0.0101 \times 0$ | 0.2 |
| $w_{36}$ | $w_{36} = w_{36} + 0.8 \times \text{Error}_6 \times O_3$ <br> $= -0.3 + 0.8 \times 0.005 \times 0$ | –0.3 |
| $w_{45}$ | $w_{45} = w_{45} + 0.8 \times \text{Error}_5 \times O_4$ <br> $= 0.1 + 0.8 \times -0.0101 \times 1$ | 0.092 |
| $w_{46}$ | $w_{46} = w_{46} + 0.8 \times \text{Error}_6 \times O_4$ <br> $= 0.4 + 0.8 \times 0.005 \times 1$ | 0.404 |
| $w_{57}$ | $w_{57} = w_{57} + 0.8 \times \text{Error}_7 \times O_5$ <br> $= -0.3 + 0.8 \times 0.141 \times 0.599$ | –0.232 |
| $w_{67}$ | $w_{67} = w_{67} + 0.8 \times \text{Error}_7 \times O_6$ <br> $= 0.2 + 0.8 \times 0.141 \times 0.769$ | 0.287 |

# Multi-Layer Perceptron Learning – Solved Example

| $x_1$ | $x_2$ | $x_3$ | $x_4$ | $w_{15}$ | $w_{16}$ | $w_{25}$ | $w_{26}$ | $w_{35}$ | $w_{36}$ | $w_{45}$ | $w_{46}$ | $w_{57}$ | $w_{67}$ | $\theta_5$ | $\theta_6$ | $\theta_7$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 1 | 0.3 | 0.1 | –0.2 | 0.4 | 0.2 | –0.3 | 0.1 | 0.4 | –0.3 | 0.2 | 0.2 | 0.1 | –0.3 |

Update bias using the below formula:

$$\theta_j = \theta_j + \propto \times \text{Error}_j$$

| $\theta_j$ | $\theta_j = \theta_j + \propto \times \text{Error}_j$ | New Bias |
|---|---|---|
| $\theta_5$ | $\theta_5 = \theta_5 + \propto \times \text{Error}_5$ $= 0.2 + 0.8 \times -0.0101$ | 0.192 |
| $\theta_6$ | $\theta_6 = \theta_6 + \propto \times \text{Error}_6$ $= 0.1 + 0.8 \times 0.005$ | 0.104 |
| $\theta_7$ | $\theta_7 = \theta_7 + \propto \times \text{Error}_7$ $= -0.3 + 0.8 \times 0.141$ | –0.187 |

# Multi-Layer Perceptron Learning – Solved Example

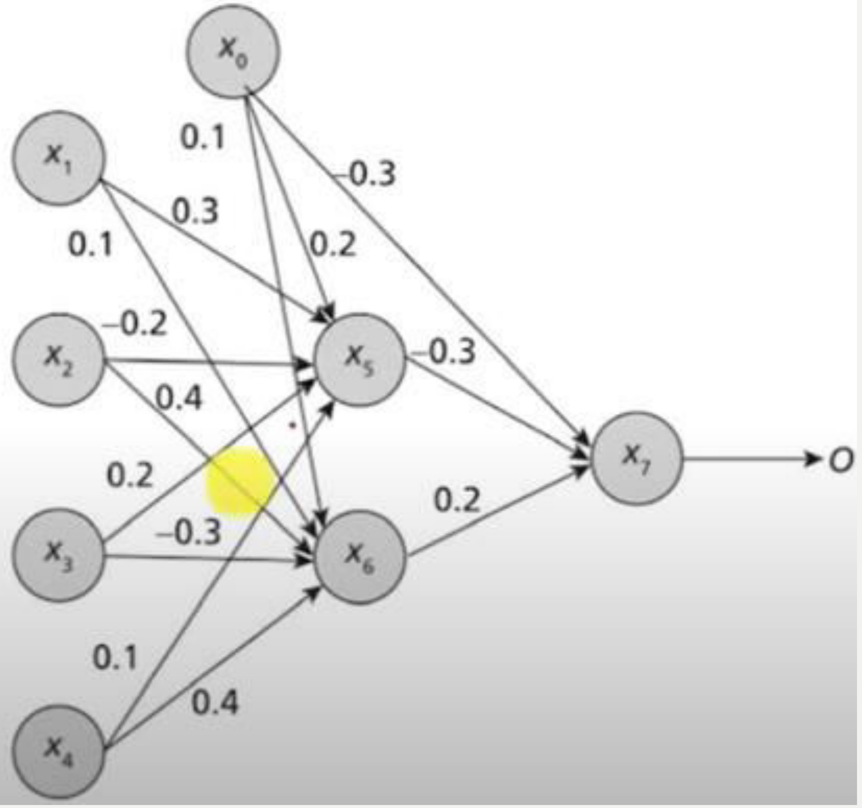
**Iteration 2**

Now, with the updated weights and biases:

1. Calculate Input and Output in the Input Layer

Net Input and Output Calculation

| Input Layer | $I_j$ | $O_j$ |
|---|---|---|
| $x_1$ | 1 | 1 |
| $x_2$ | 1 | 1 |
| $x_3$ | 0 | 0 |
| $x_4$ | 1 | 1 |



# Multi-Layer Perceptron Learning – Solved Example

2. Calculate Net Input and Output in the Hidden Layer and Output Layer

Net Input and Output Calculation in the Hidden Layer and Output Layer

| Unit $j$ | Net Input $I_j$ | Output $O_j$ |
|---|---|---|
| $x_5$ | $I_5 = x_1 \times w_{15} + x_2 \times w_{25} + x_3 \times w_{35} + x_4 \times w_{45} + x_0 \times \theta_5$<br>$I_5 = 1 \times 0.292 + 1 \times -0.208 + 0 \times 0.2 + 1 \times 0.092 + 1 \times 0.192 = 0.368$ | $O_5 = \frac{1}{1+e^{-I_5}} = \frac{1}{1+e^{-0.368}} = 0.591$ |
| $x_6$ | $I_6 = x_1 \times w_{15} + x_2 \times w_{26} + x_3 \times w_{36} + x_4 \times w_{46} + x_0 \times \theta_6$<br>$I_6 = 1 \times 0.292 + 1 \times 0.404 + 0 \times -0.3 + 1 \times 0.404 + 1 \times 0.104 = 1.204$ | $O_6 = \frac{1}{1+e^{-I_6}} = \frac{1}{1+e^{-1.204}} = 0.7692$ |
| $x_7$ | $I_7 = O_5 \times w_{57} + O_6 \times w_{67} + x_0 \times \theta_7$<br>$I_7 = 0.591 \times -0.232 + 0.7692 \times 0.287 + 1 \times -0.187 = -0.326$ | $O_7 = \frac{1}{1+e^{-I_7}} = \frac{1}{1+e^{0.1034}} = 0.474$ |

# Multi-Layer Perceptron Learning – Solved Example

- The output we receive in the network at node 7 is 0.474

**Error= 1 - 0.474 = 0.526**

- Now, when we compare the error, we get in the previous iteration and in the current iteration,

**Error reduced is 0.581 - 0.526 = 0.055**

- It is visible that the network has learnt and reduced the error by 0.055.
- Thus, the training is continued for a predefined number of epochs or until the training error is reduced below a threshold value.