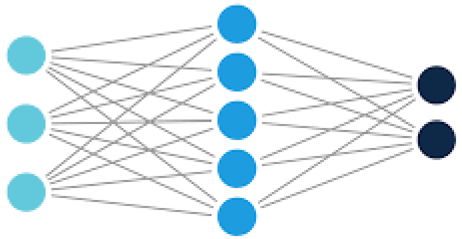
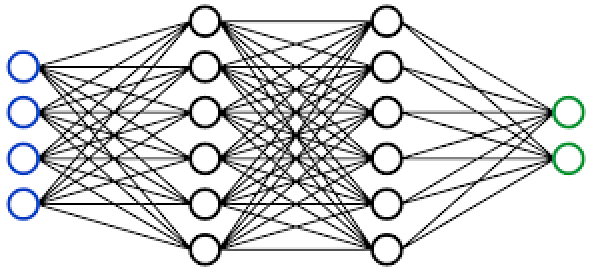
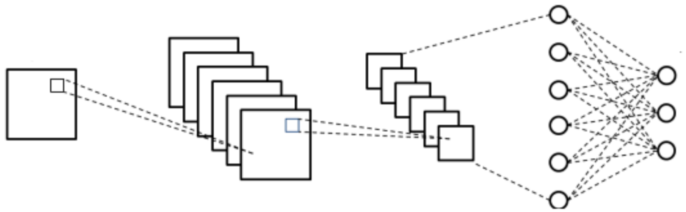
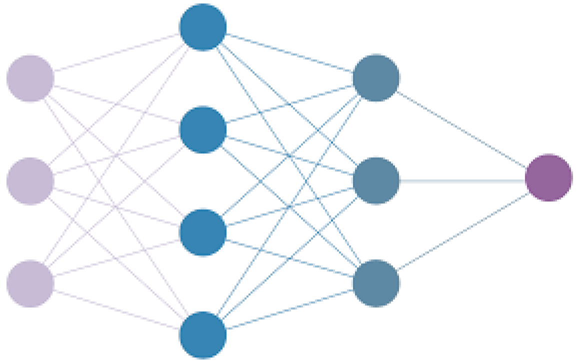
Lesson – 8

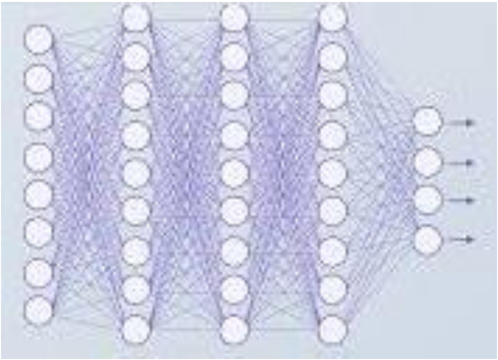
Neural Network: An inspiration from Human Brain

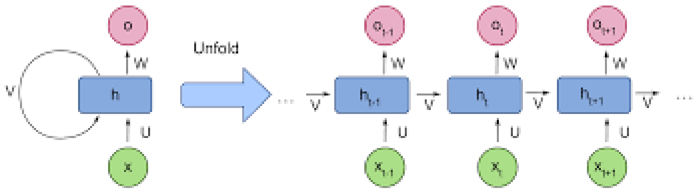






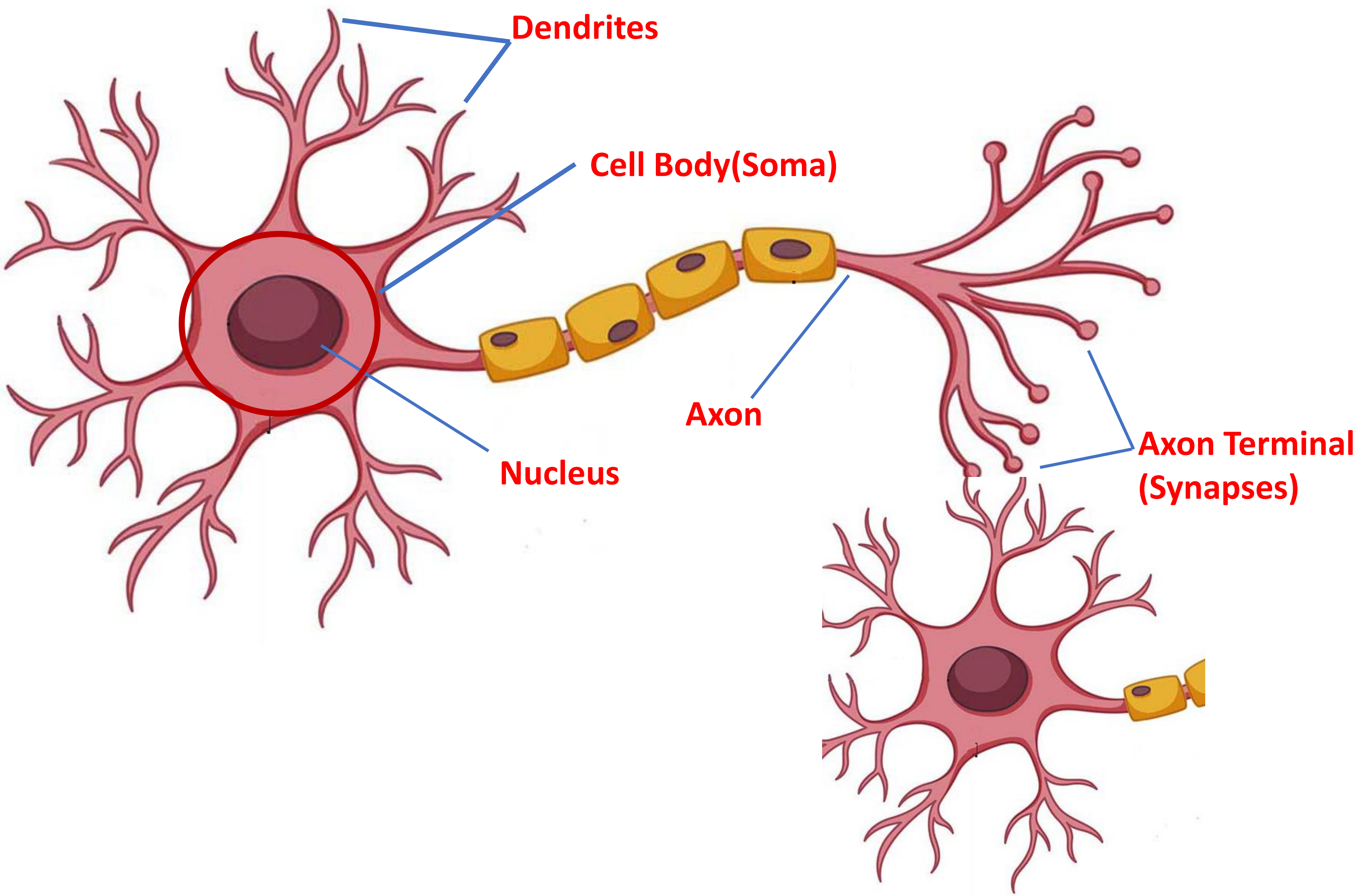


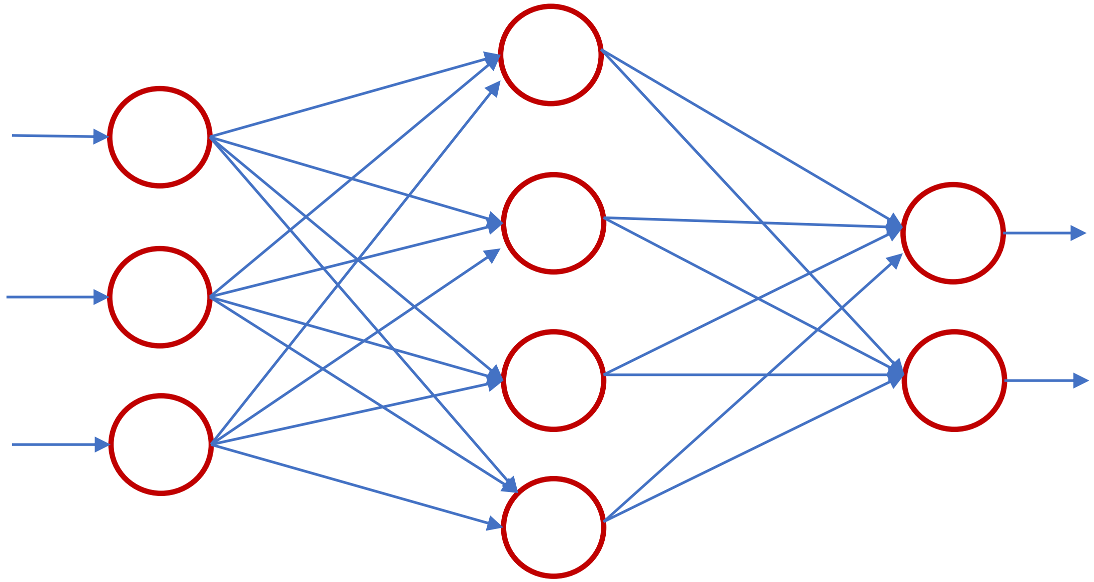


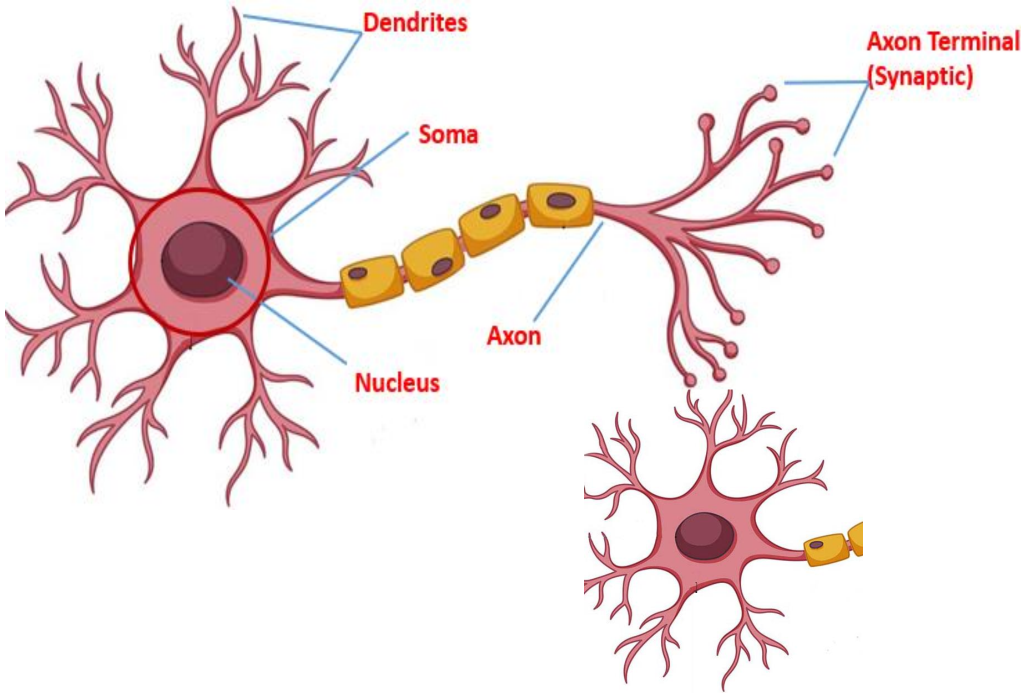


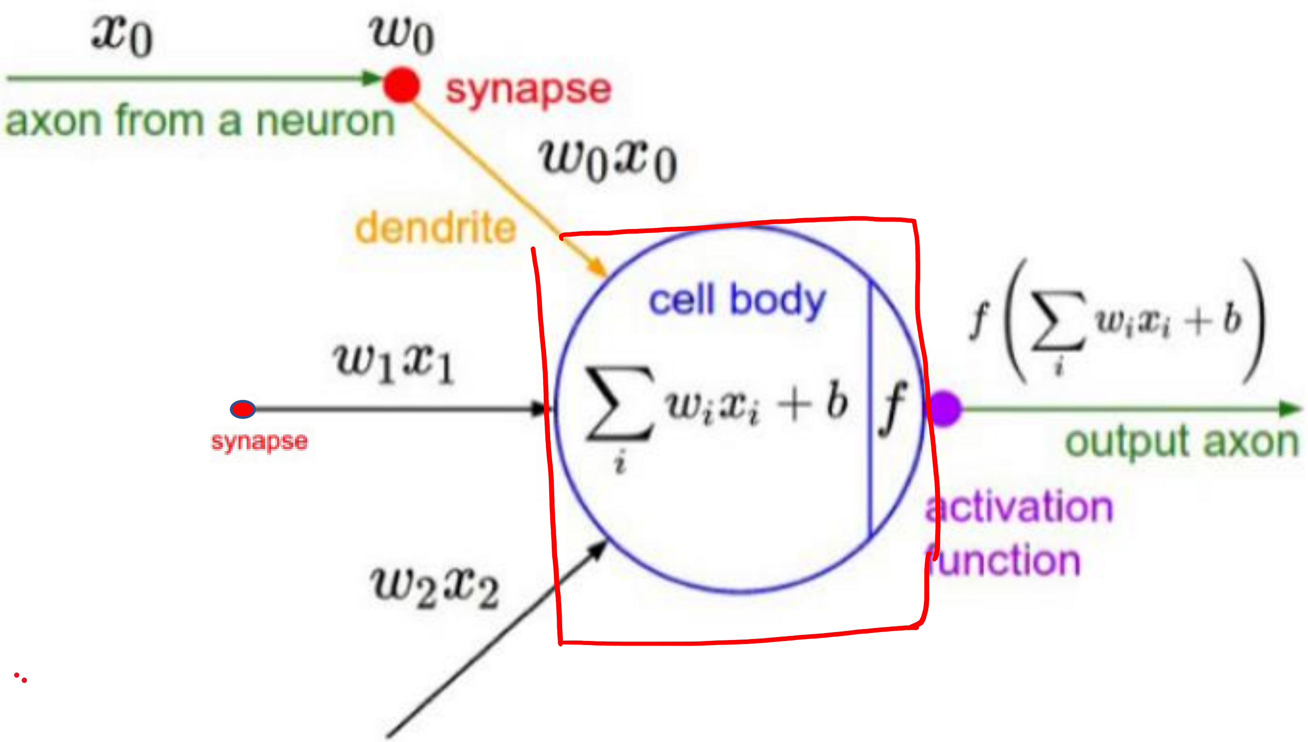
o
W
V
h
U
x
Unfold
o_{t-1}
h_{t-1}
x_{t-1}
o_t
h_t
x_t
o_{t+1}
h_{t+1}
x_{t+1}
...

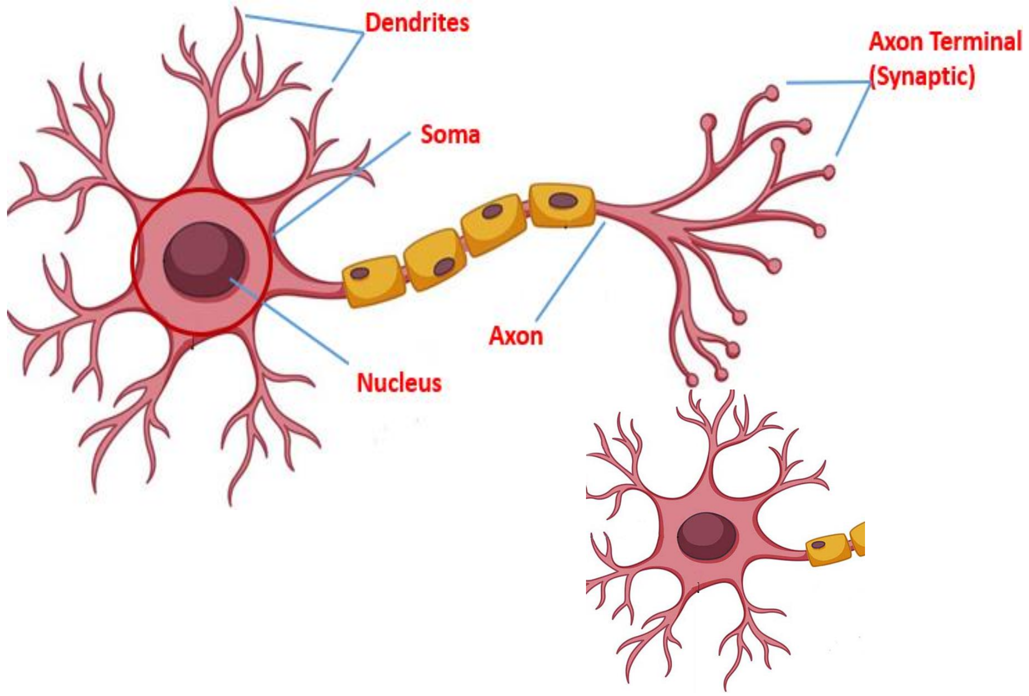
Network of neurons in Human Brain

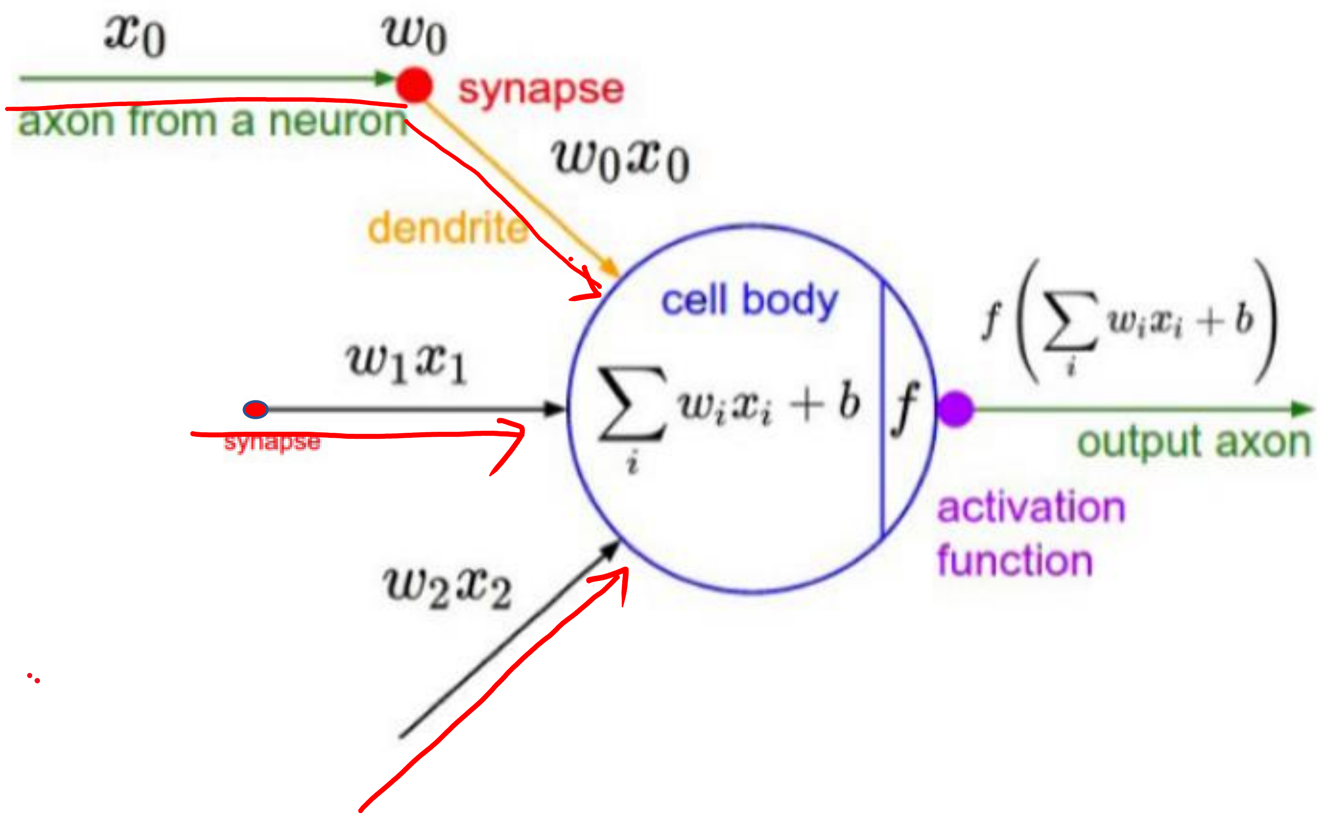

Dendrites
Cell Body(Soma)
Axon
Nucleus
Axon Terminal
(Synapses)

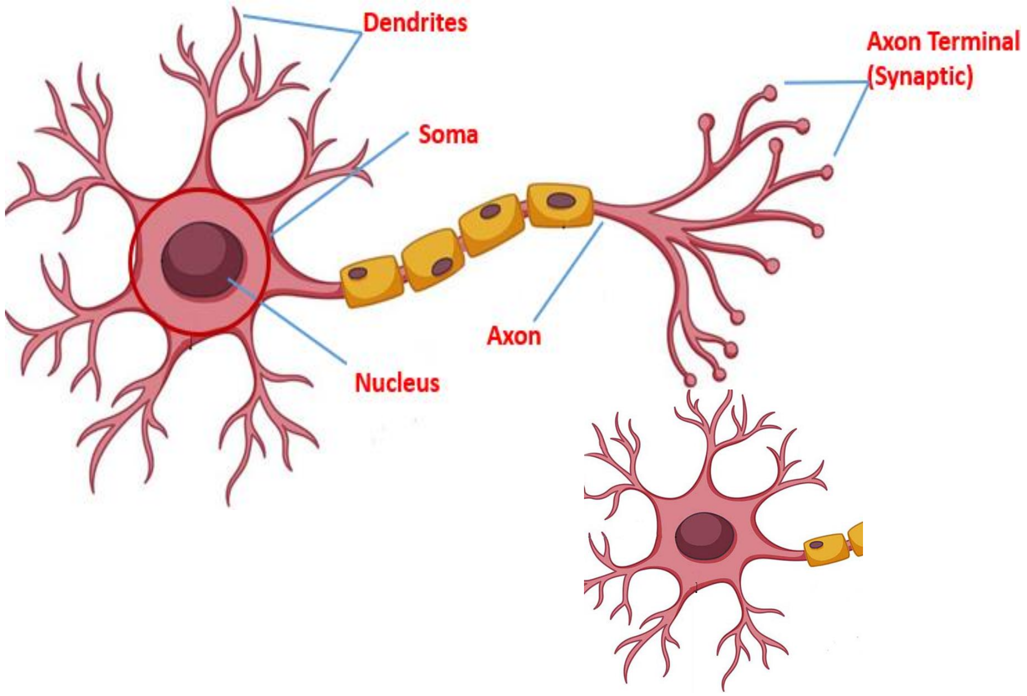


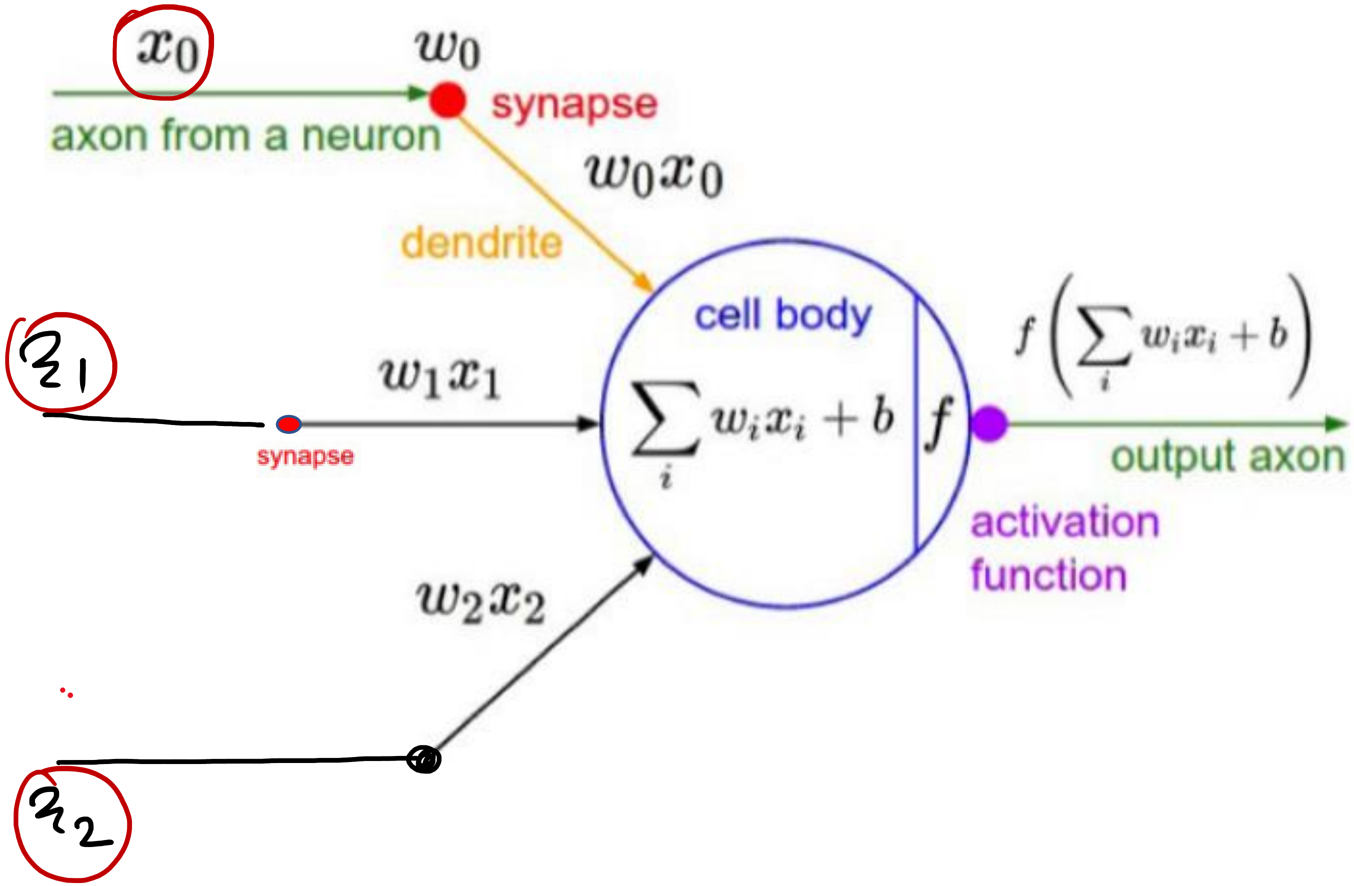

Dendrites
Axon Terminal
(Synaptic)
Soma
Axon
Nucleus

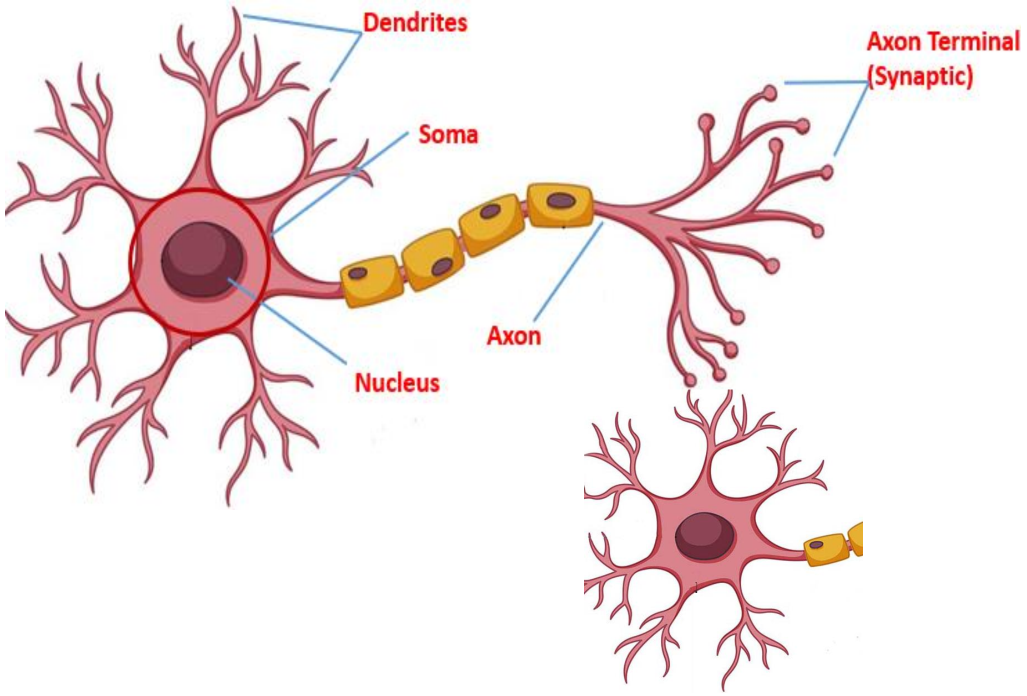

x_0
w_0
synapse
axon from a neuron
$w_0 x_0$
dendrite
cell body
$f\left(\sum_i w_i x_i + b\right)$
$w_1 x_1$
$\sum_i w_i x_i + b$
f
synapse
output axon
activation
function
$w_2 x_2$

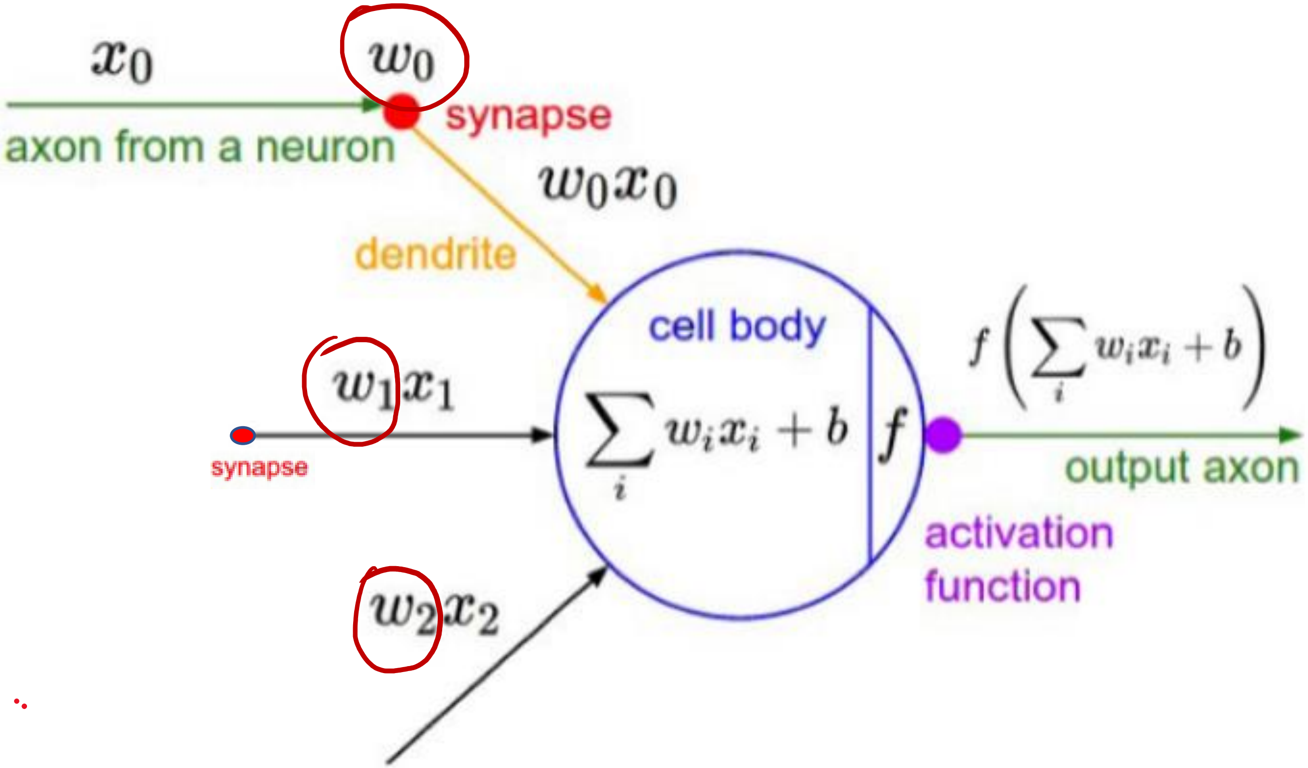

Dendrites
Axon Terminal
(Synaptic)
Soma
Axon
Nucleus

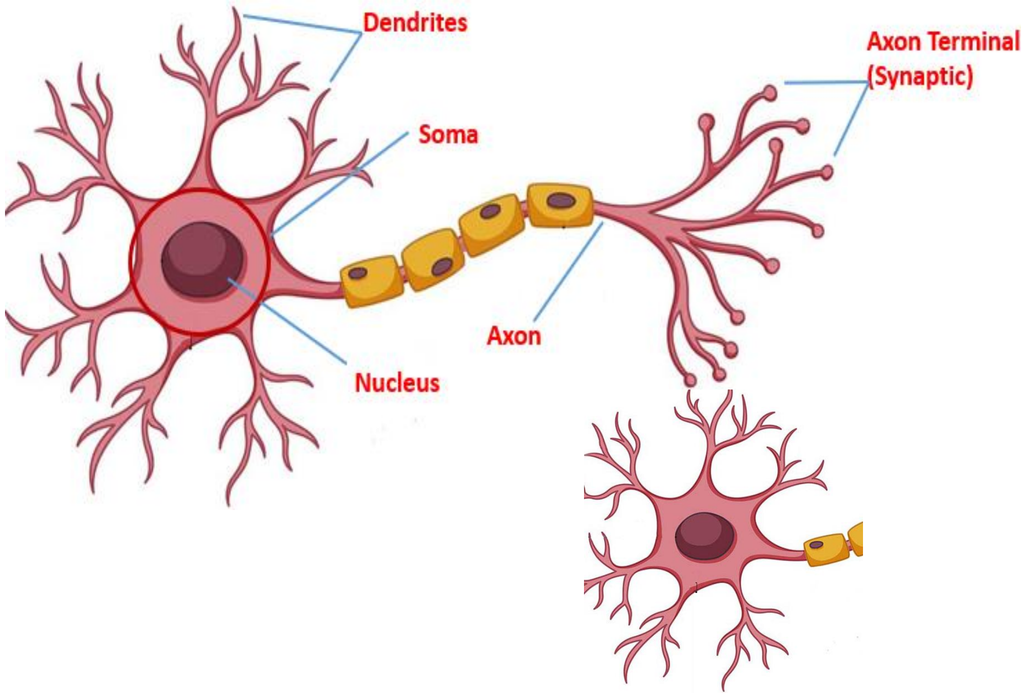

x_0
w_0
synapse
axon from a neuron
$w_0 x_0$
dendrite
cell body
$f\left(\sum_i w_i x_i + b\right)$
$w_1 x_1$
$\sum_i w_i x_i + b$
f
output axon
synapse
activation
function
$w_2 x_2$

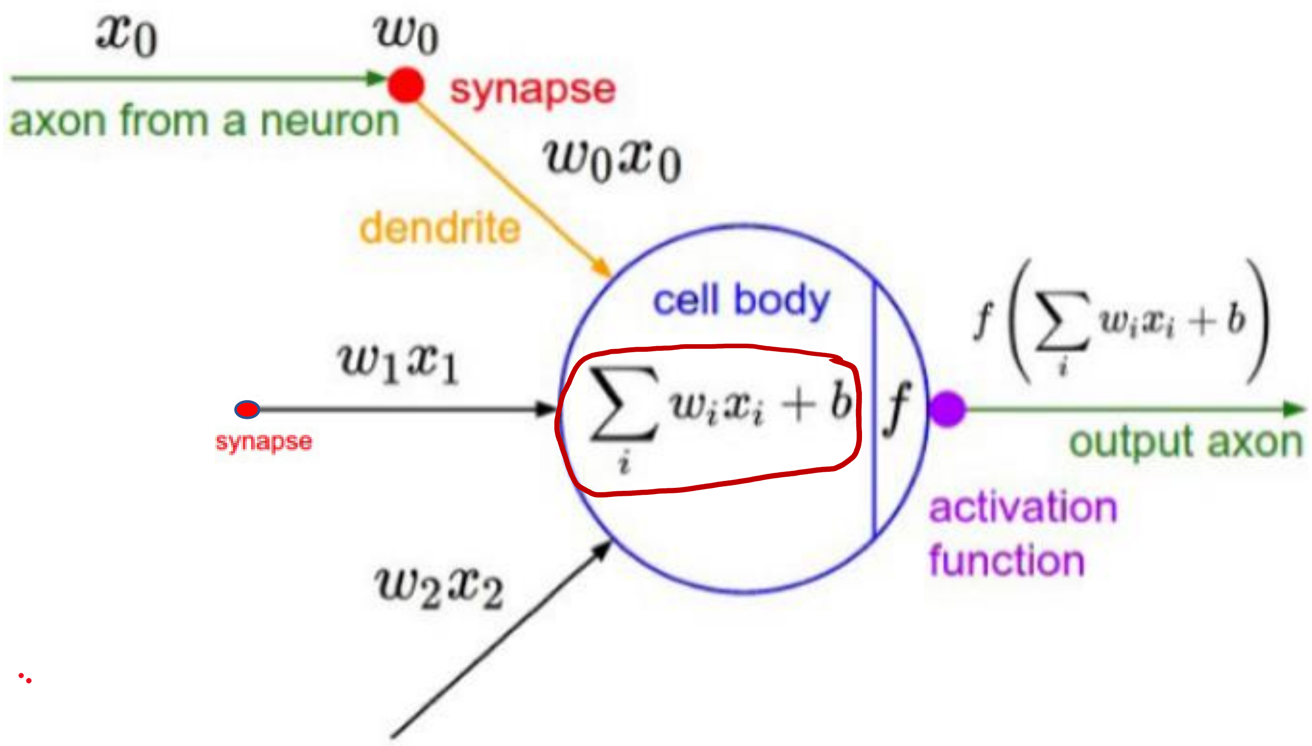

Dendrites
Axon Terminal
(Synaptic)
Soma
Axon
Nucleus

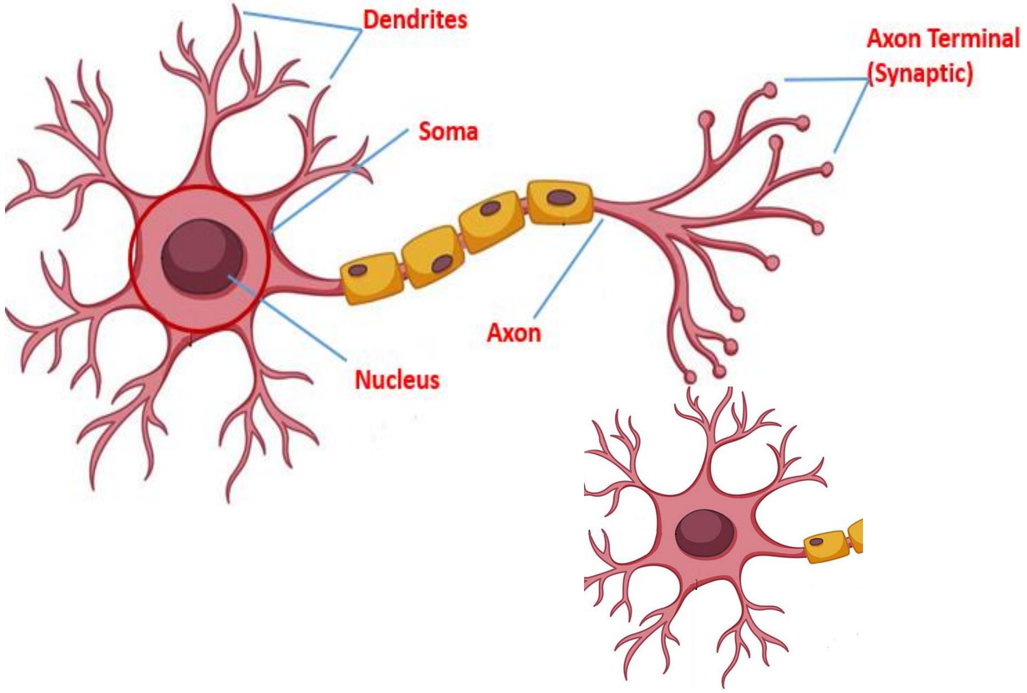

x_0
w_0
synapse
axon from a neuron
$w_0 x_0$
dendrite
cell body
$f\left(\sum_i w_i x_i + b\right)$
z_1
$w_1 x_1$
$\sum_i w_i x_i + b$
f
synapse
output axon
activation
function
$w_2 x_2$
z_2

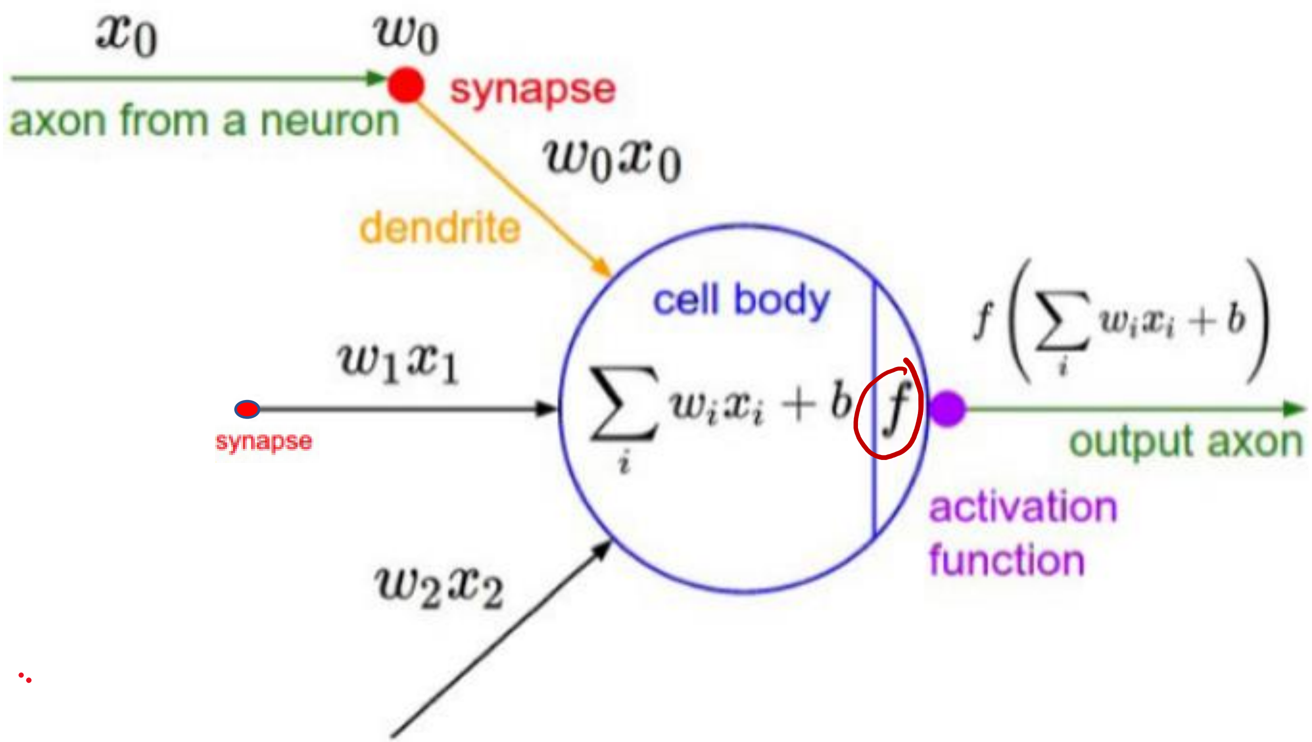

Dendrites
Axon Terminal
(Synaptic)
Soma
Axon
Nucleus

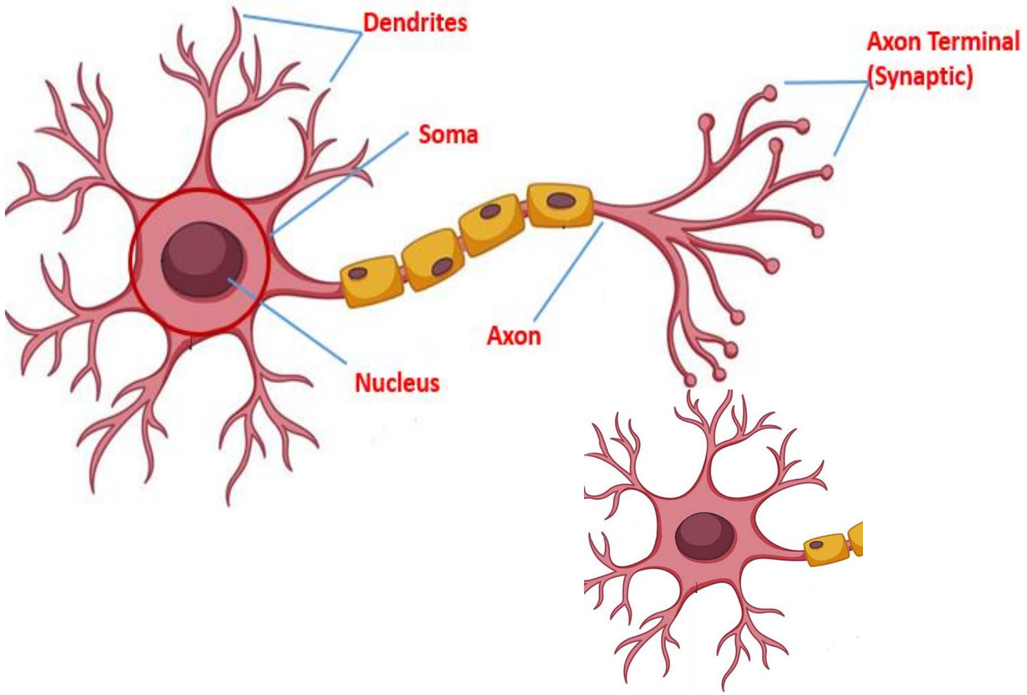

x_0
w_0
synapse
axon from a neuron
$w_0 x_0$
dendrite
cell body
$w_1 x_1$
synapse
$\sum_i w_i x_i + b$
f
$f\left(\sum_i w_i x_i + b\right)$
output axon
activation
function
$w_2 x_2$

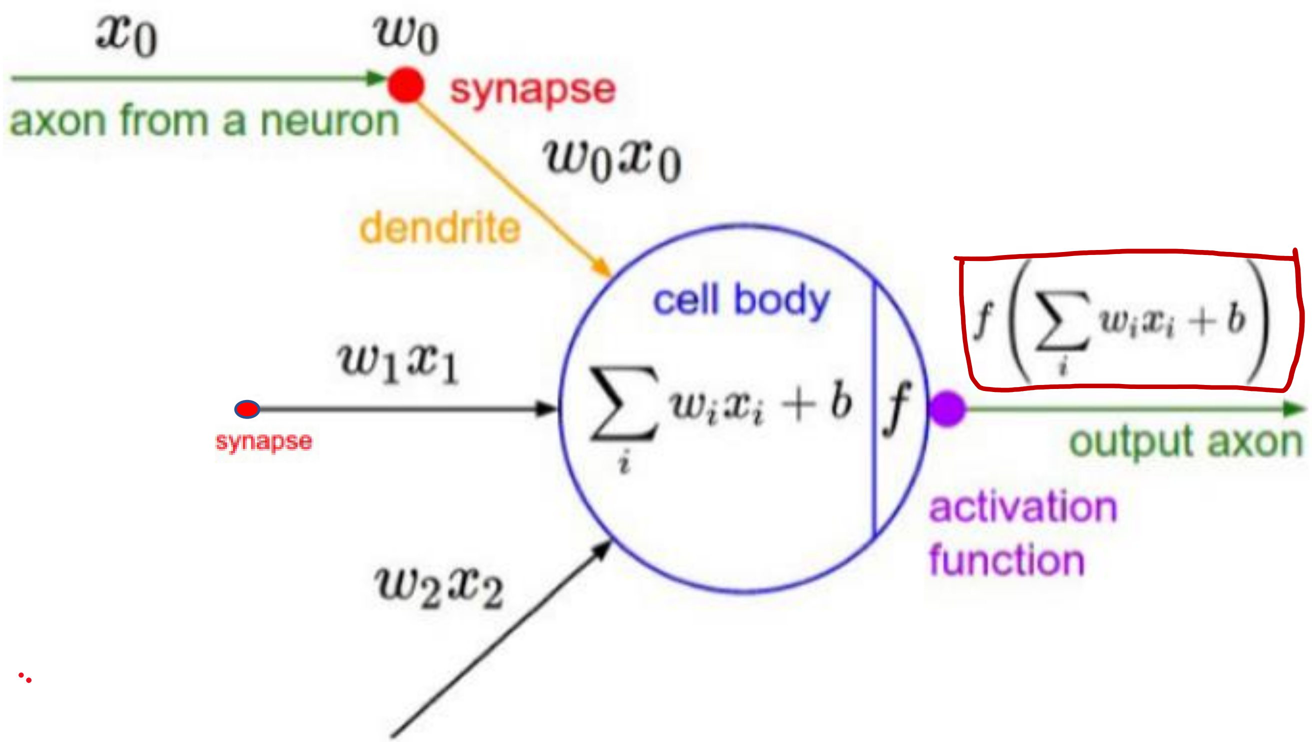

Dendrites
Axon Terminal
(Synaptic)
Soma
Axon
Nucleus


x_0
w_0
synapse
axon from a neuron
$w_0 x_0$
dendrite
cell body
$f\left(\sum_i w_i x_i + b\right)$
$w_1 x_1$
$\sum_i w_i x_i + b$
f
synapse
output axon
activation
function
$w_2 x_2$

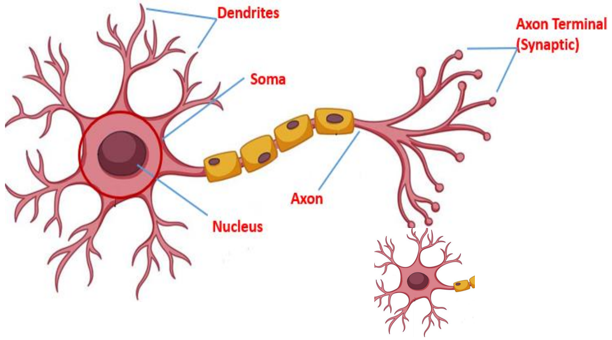

Dendrites
Axon Terminal
(Synaptic)
Soma
Axon
Nucleus

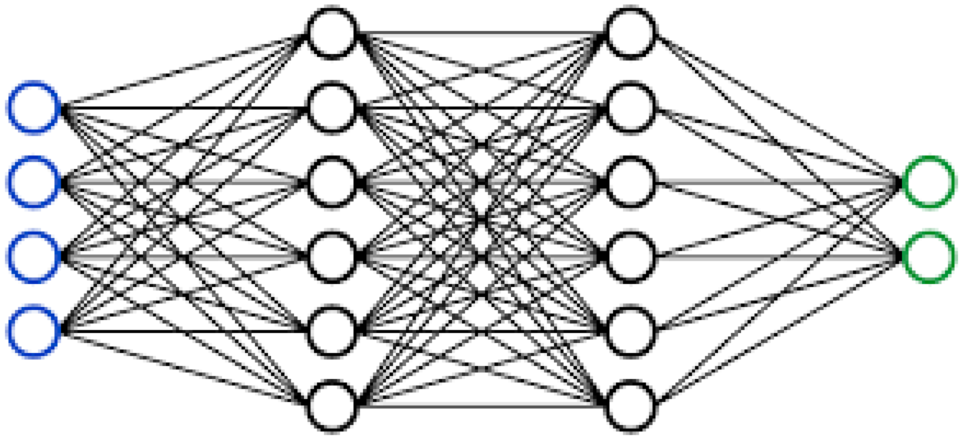

x_0
w_0
synapse
axon from a neuron
$w_0 x_0$
dendrite
cell body
$f\left(\sum_i w_i x_i + b\right)$
$w_1 x_1$
$\sum_i w_i x_i + b$
f
synapse
output axon
activation
function
$w_2 x_2$


Dendrites
Axon Terminal
(Synaptic)
Soma
Axon
Nucleus

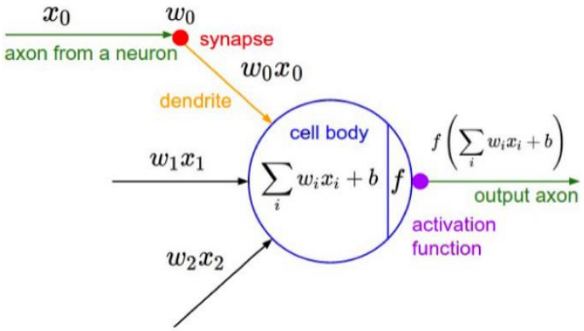

x_0
w_0
synapse
axon from a neuron
$w_0 x_0$
dendrite
cell body
$f\left(\sum_i w_i x_i + b\right)$
$w_1 x_1$
synapse
$\sum_i w_i x_i + b$
f
output axon
activation
function
$w_2 x_2$

Network of neurons



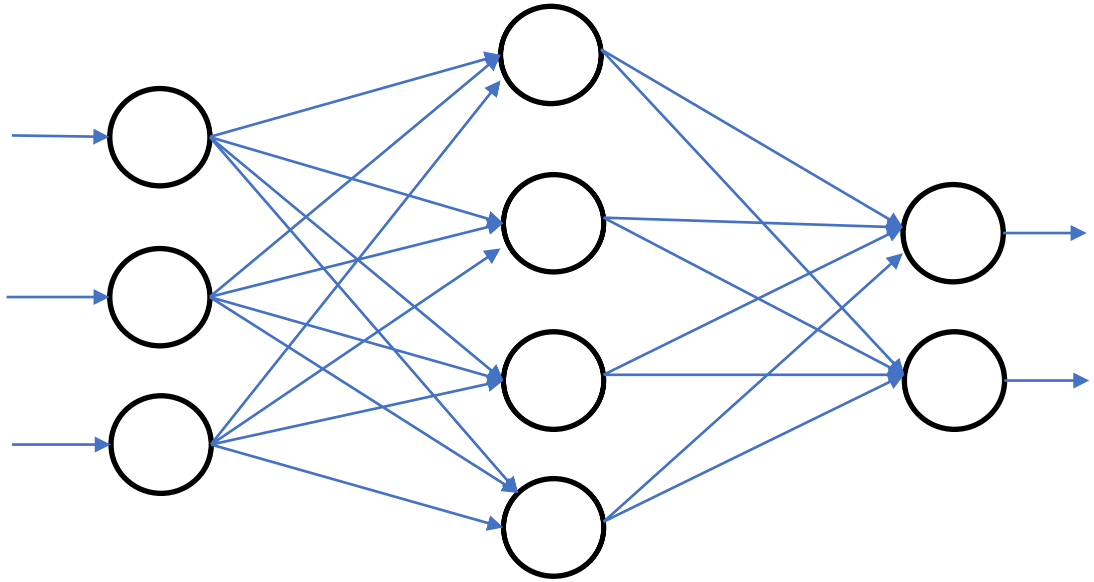


Network of neurons

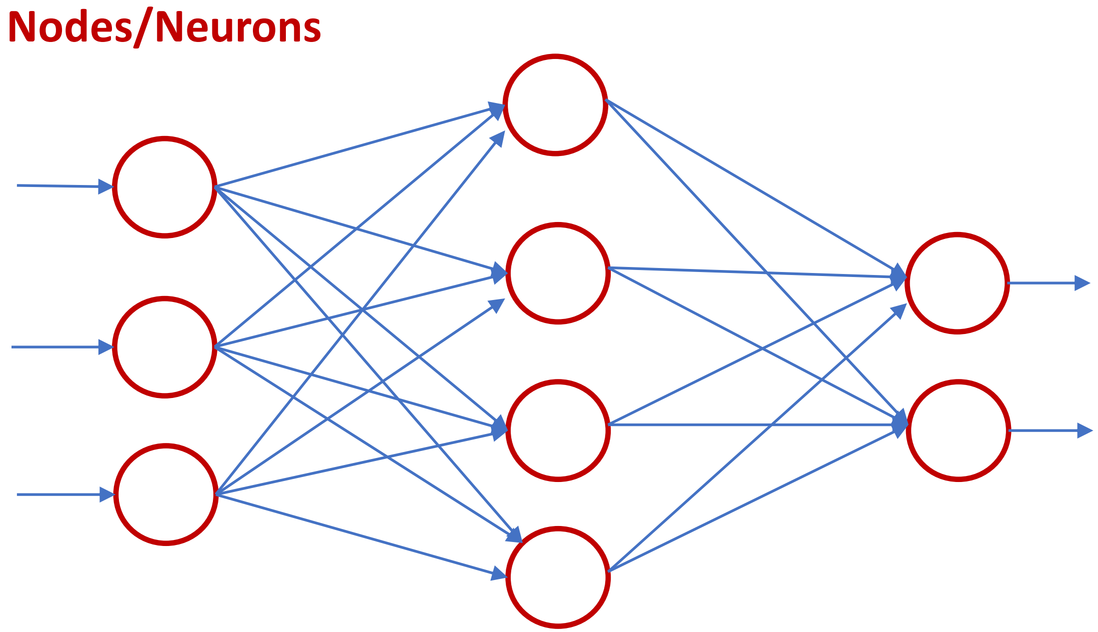


Lesson 8:
Neural Network: Terminologies and a Toy Example

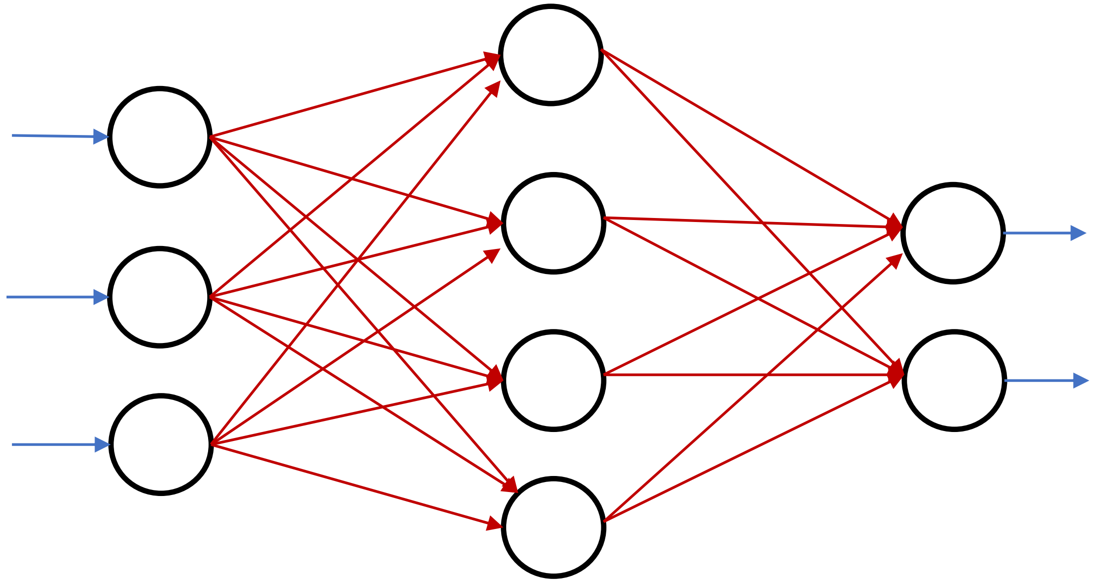
Terminologies in Artificial Neural Network



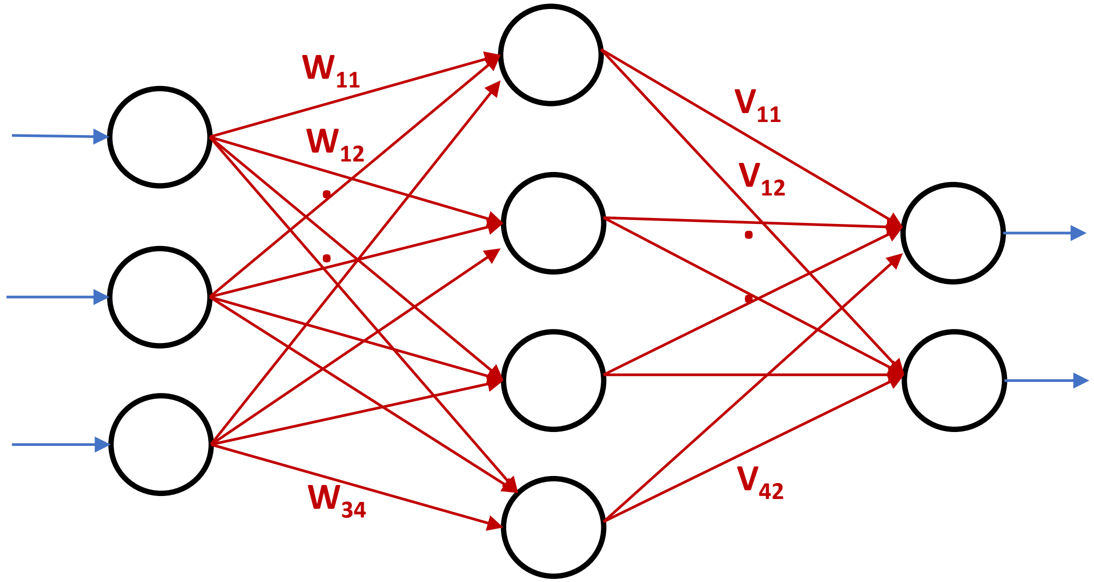
Terminologies in Artificial Neural Network

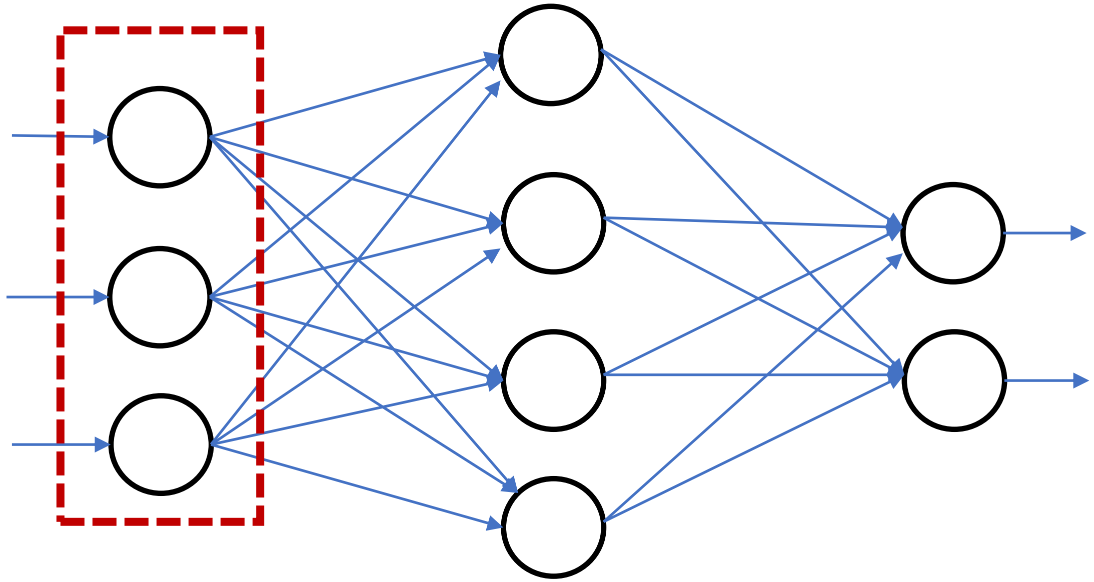


Connections/Dendrites

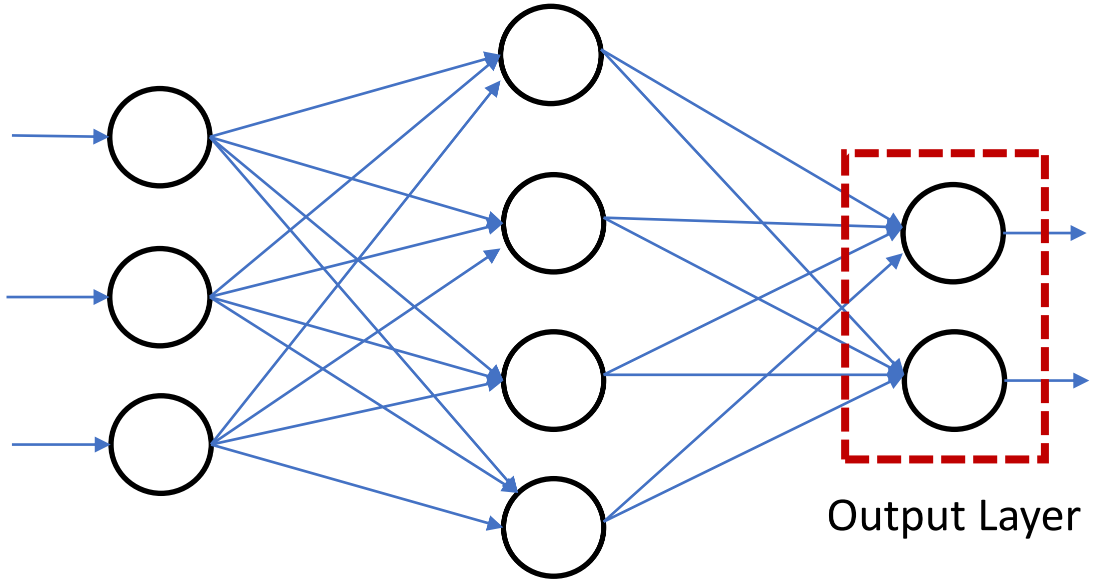


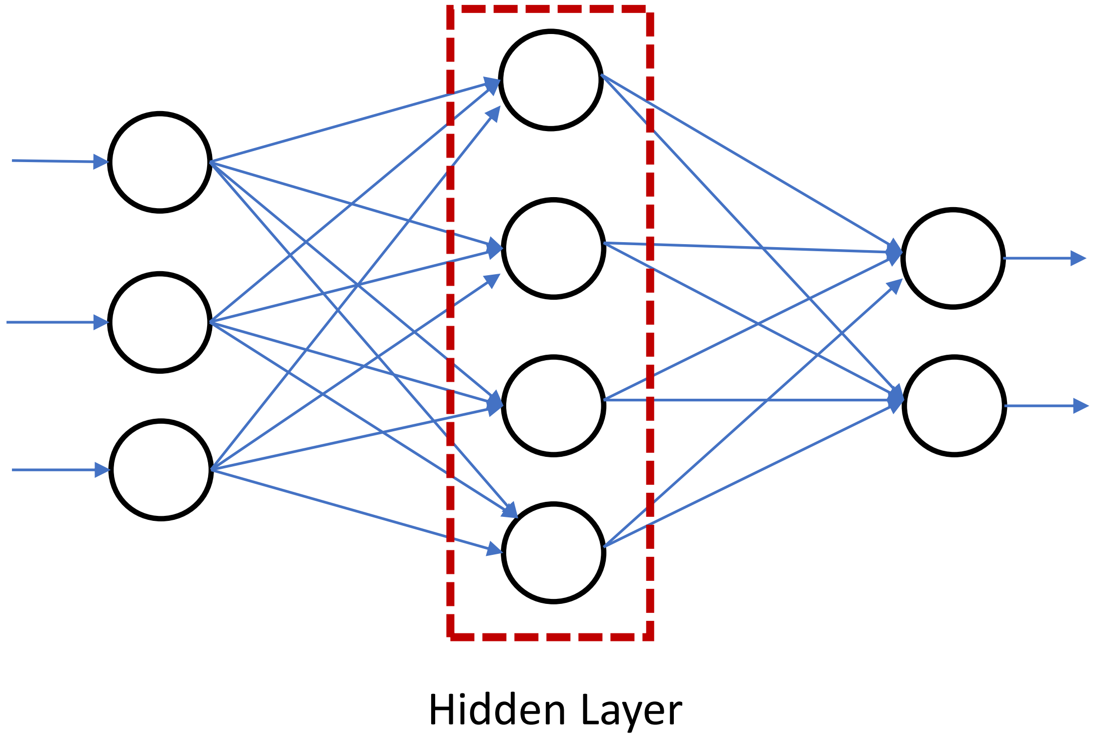
Weights/Synapses

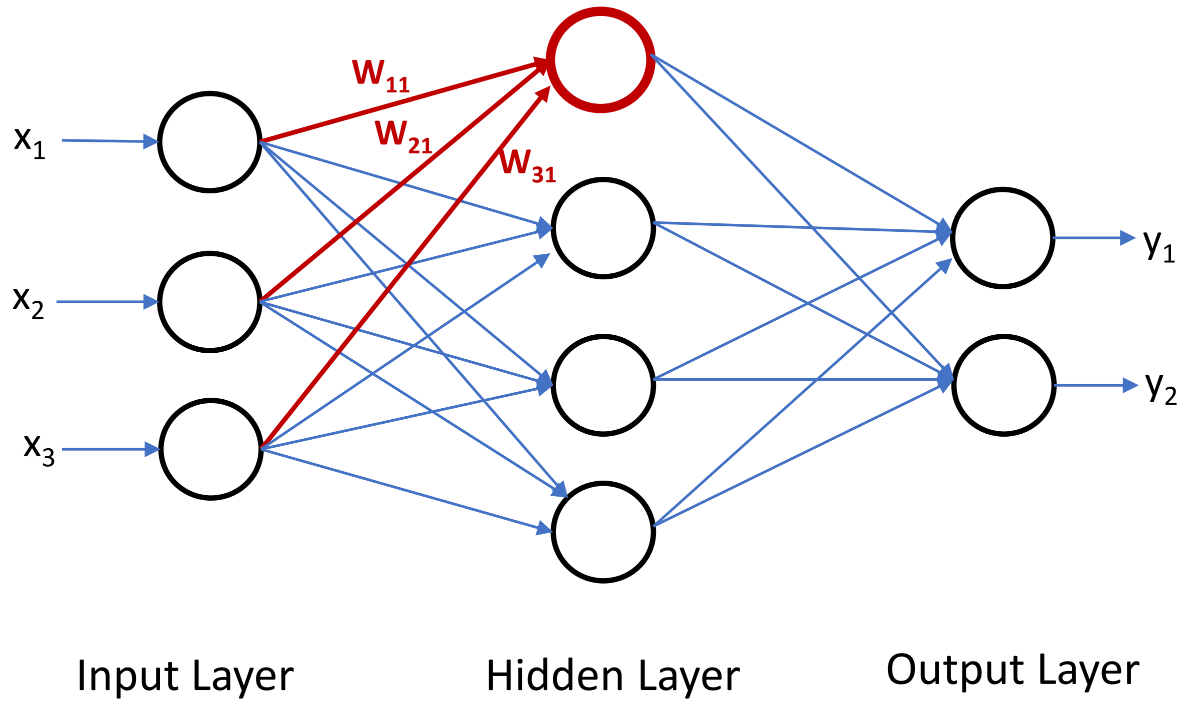




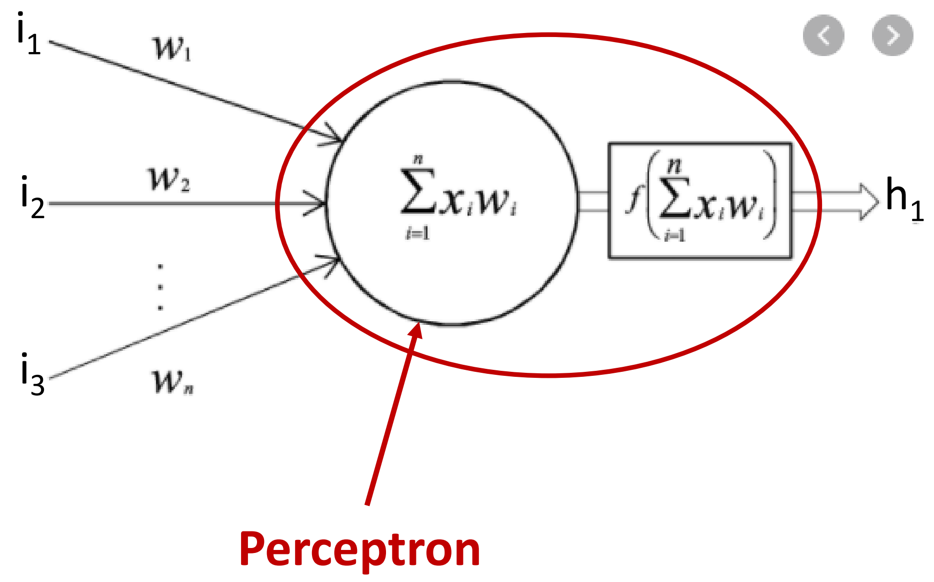
Input Layer


Output Layer

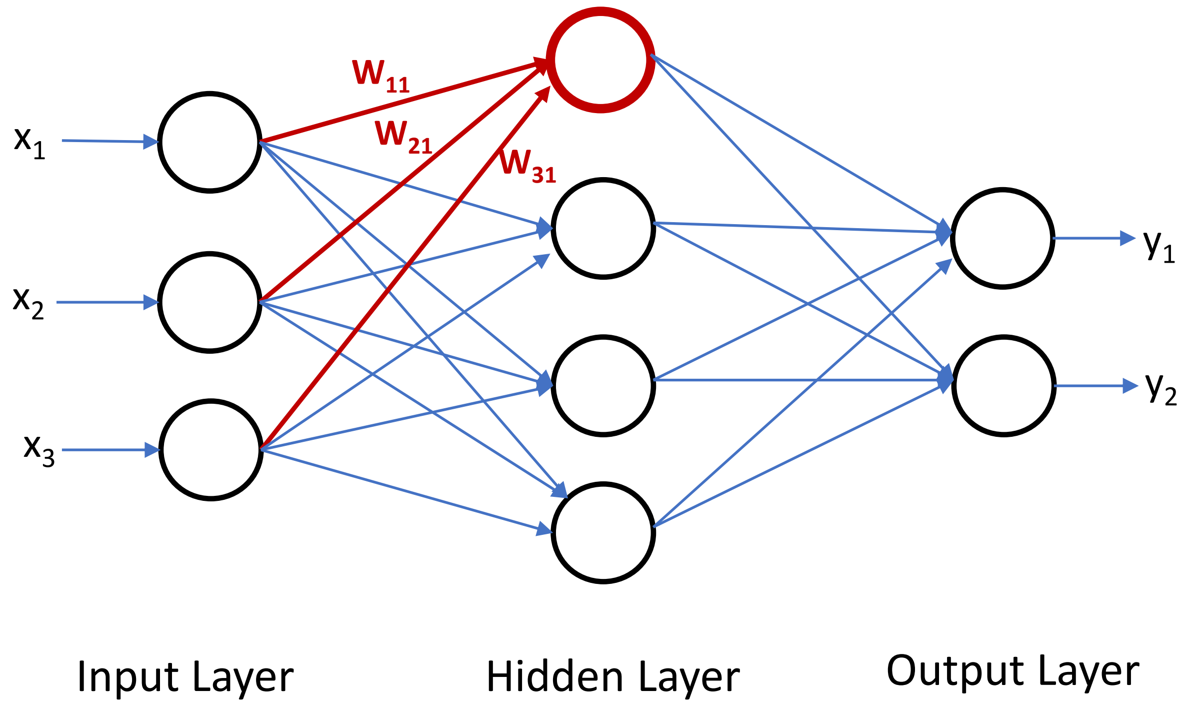

Hidden Layer



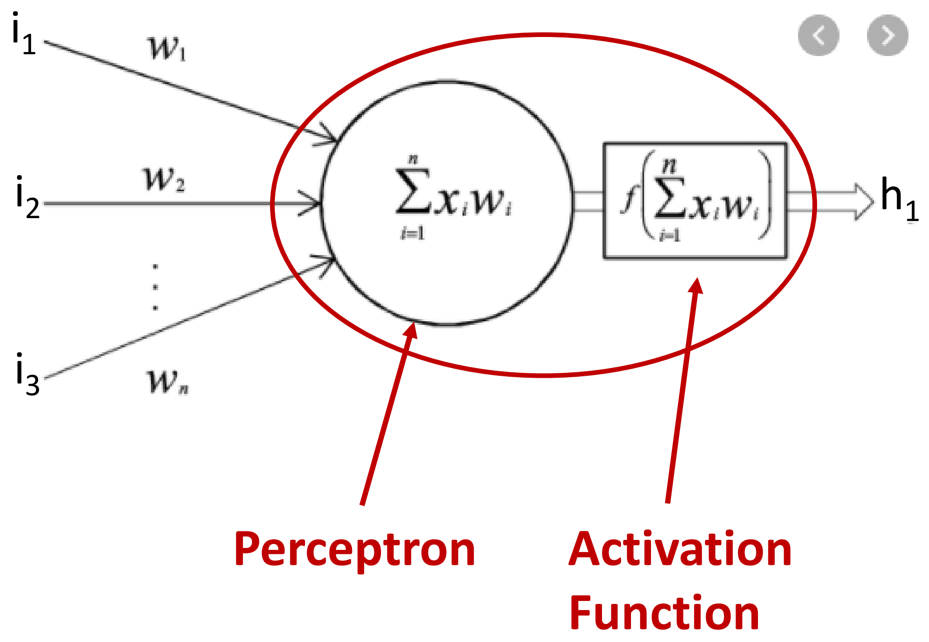
x1
x2
x3
W11
W21
W31
y1
y2
Input Layer
Hidden Layer
Output Layer



i1
i2
i3
w1
w2
wn
h1
Perceptron

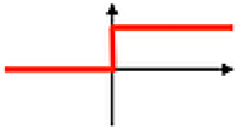
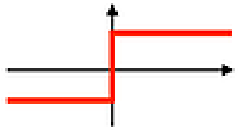
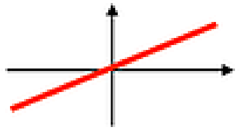
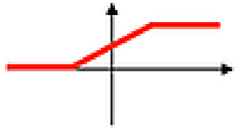
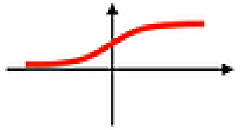
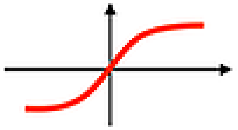
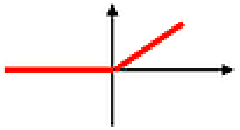
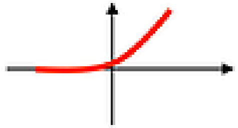


x1
x2
x3
W11
W21
W31
y1
y2
Input Layer
Hidden Layer
Output Layer

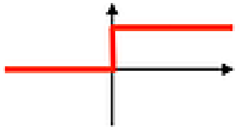
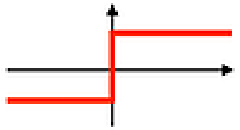
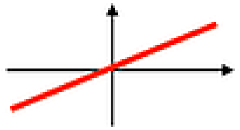
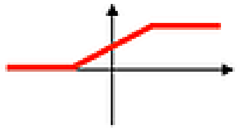
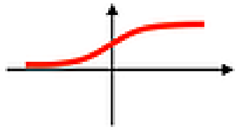
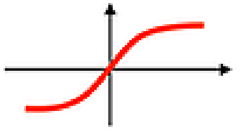
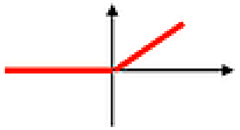
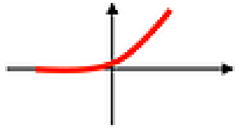


i1
i2
i3
w1
w2
wn
h1
Perceptron
Activation Function

Activation Functions

Activation function	Equation	Example	1D Graph
Unit step (Heaviside)	$\phi(z) = \begin{cases} 0, & z < 0, \\ 0.5, & z = 0, \\ 1, & z > 0, \end{cases}$	Perceptron variant	
Sign (Signum)	$\phi(z) = \begin{cases} -1, & z < 0, \\ 0, & z = 0, \\ 1, & z > 0, \end{cases}$	Perceptron variant	
Linear	$\phi(z) = z$	Adaline, linear regression	
Piece-wise linear	$\phi(z) = \begin{cases} 1, & z \geq \frac{1}{2}, \\ z + \frac{1}{2}, & -\frac{1}{2} < z < \frac{1}{2}, \\ 0, & z \leq -\frac{1}{2}, \end{cases}$	Support vector machine	
Logistic (sigmoid)	$\phi(z) = \frac{1}{1 + e^{-z}}$	Logistic regression, Multi-layer NN	
Hyperbolic tangent	$\phi(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$	Multi-layer Neural Networks	
Rectifier, ReLU (Rectified Linear Unit)	$\phi(z) = max(0, z)$	Multi-layer Neural Networks	
Rectifier, softplus	$\phi(z) = \ln(1 + e^z)$	Multi-layer Neural Networks	

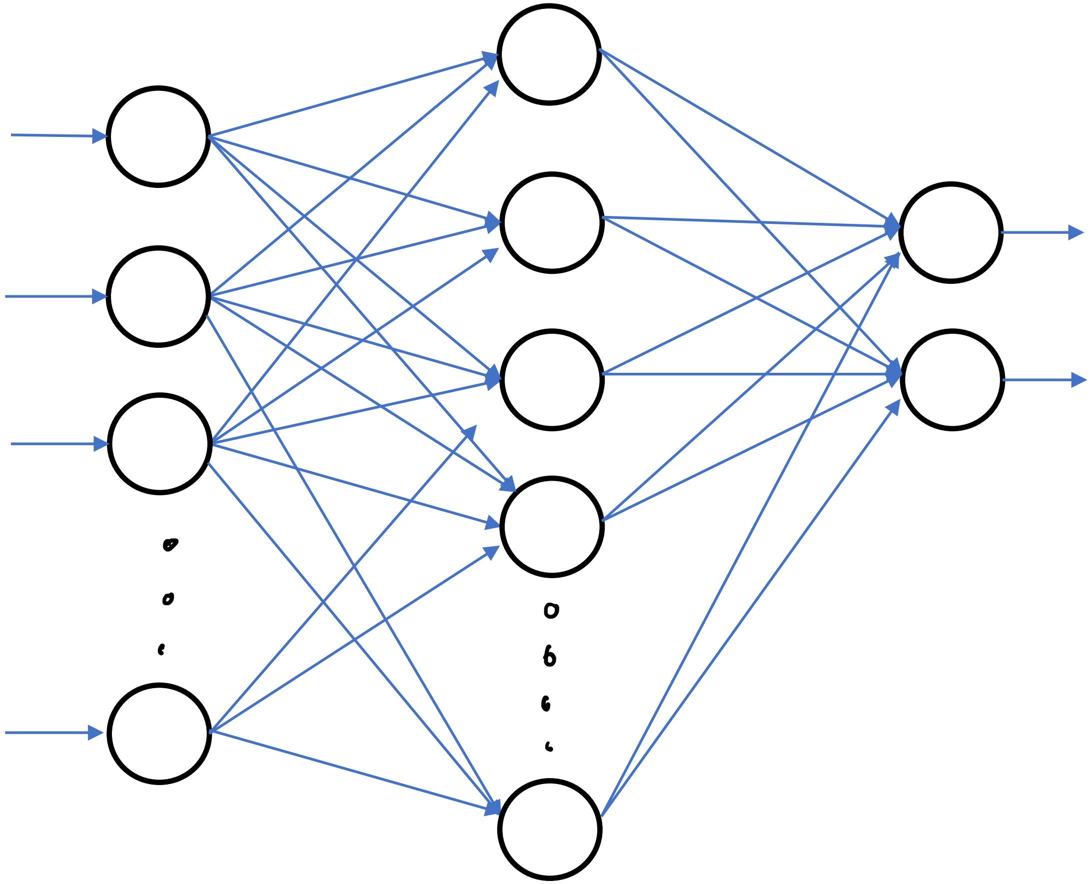
Activation Functions

Activation function	Equation	Example	1D Graph
Unit step (Heaviside)	$\phi(z) = \begin{cases} 0, & z < 0, \\ 0.5, & z = 0, \\ 1, & z > 0, \end{cases}$	Perceptron variant	
Sign (Signum)	$\phi(z) = \begin{cases} -1, & z < 0, \\ 0, & z = 0, \\ 1, & z > 0, \end{cases}$	Perceptron variant	
Linear	$\phi(z) = z$	Adaline, linear regression	
Piece-wise linear	$\phi(z) = \begin{cases} 1, & z \geq \frac{1}{2}, \\ z + \frac{1}{2}, & -\frac{1}{2} < z < \frac{1}{2}, \\ 0, & z \leq -\frac{1}{2}, \end{cases}$	Support vector machine	
Logistic (sigmoid)	$\phi(z) = \frac{1}{1 + e^{-z}}$	Logistic regression, Multi-layer NN	
Hyperbolic tangent	$\phi(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$	Multi-layer Neural Networks	
Rectifier, ReLU (Rectified Linear Unit)	$\phi(z) = max(0, z)$	Multi-layer Neural Networks	
Rectifier, softplus Copyright © Sebastian Raschka 2016 (http://sebastianraschka.com)	$\phi(z) = \ln(1 + e^z)$	Multi-layer Neural Networks	

A Toy Example

60x60=3600

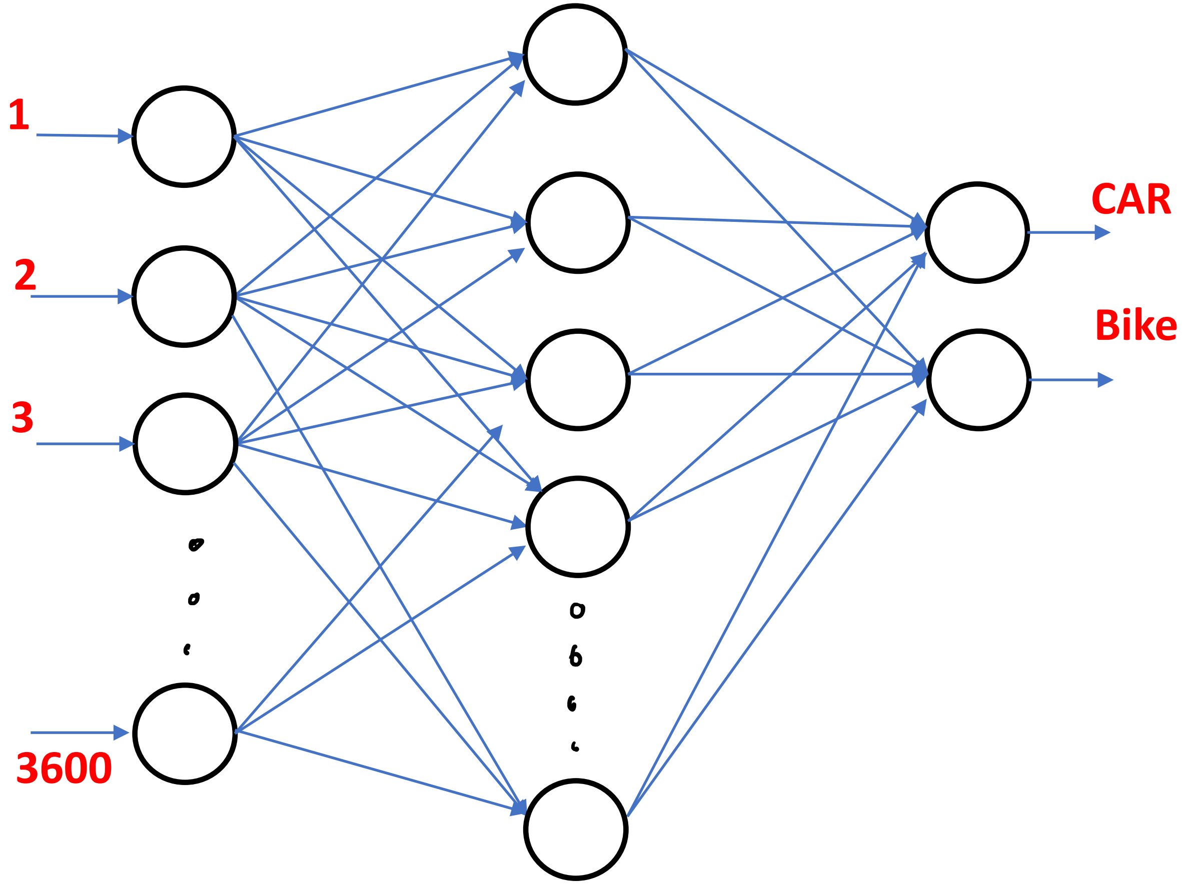
60x60=3600



A Toy Example

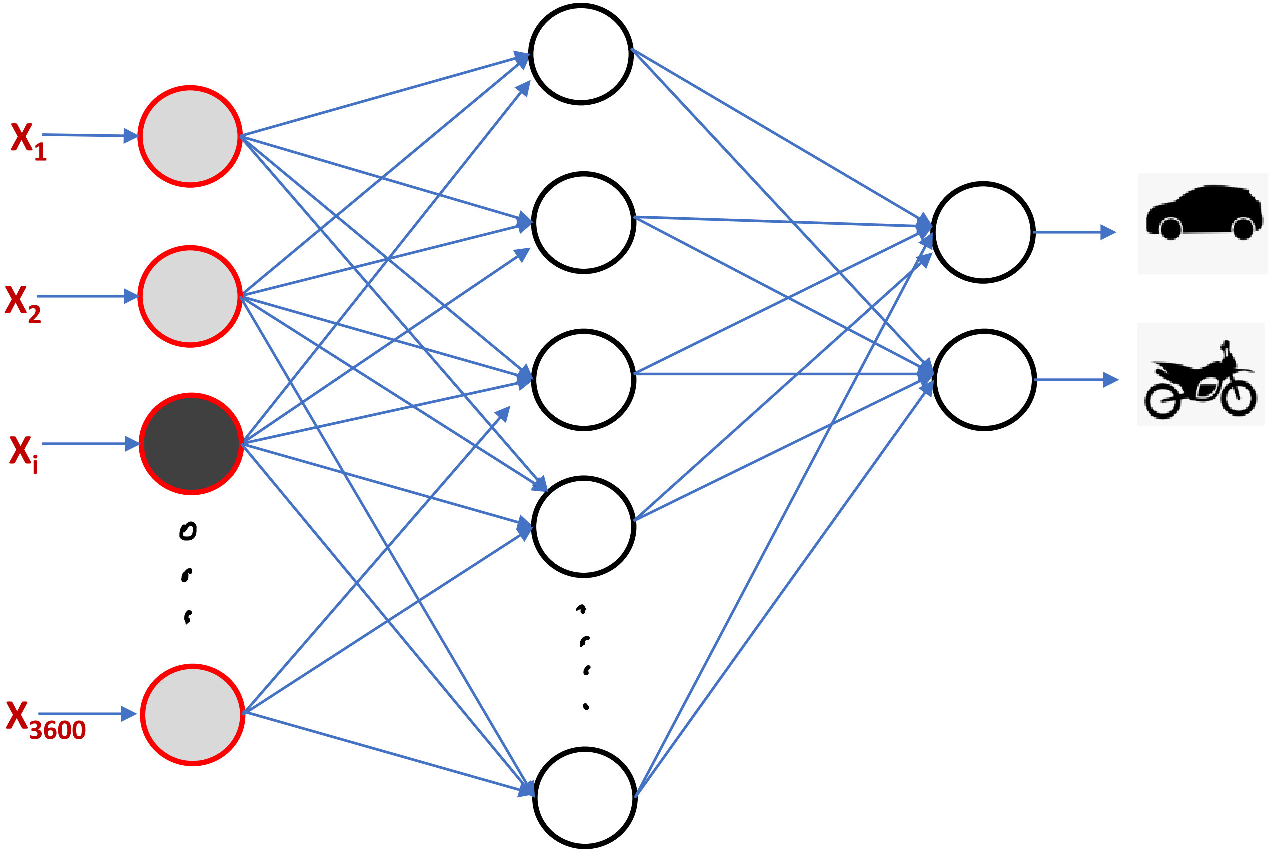
60x60=3600

60x60=3600



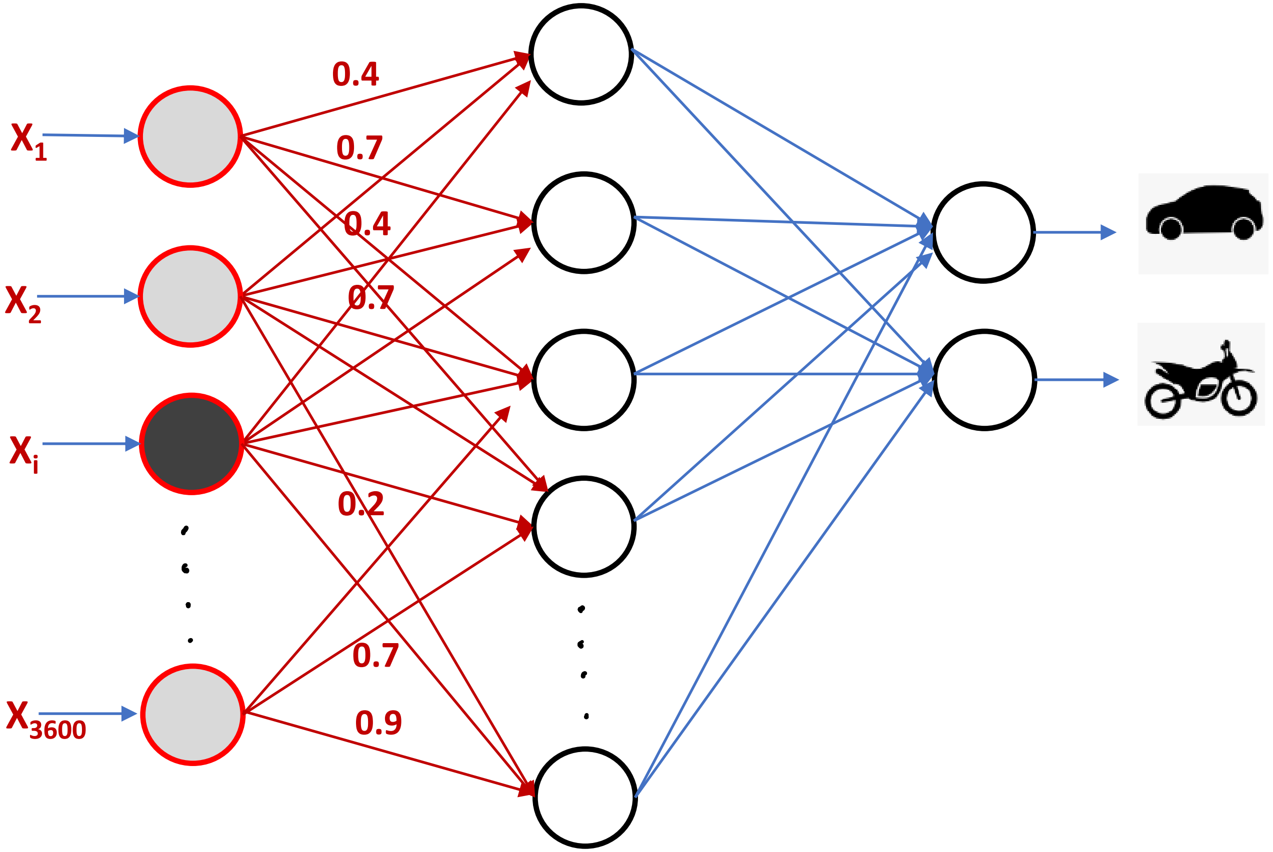
A Toy Example

60x60=3600



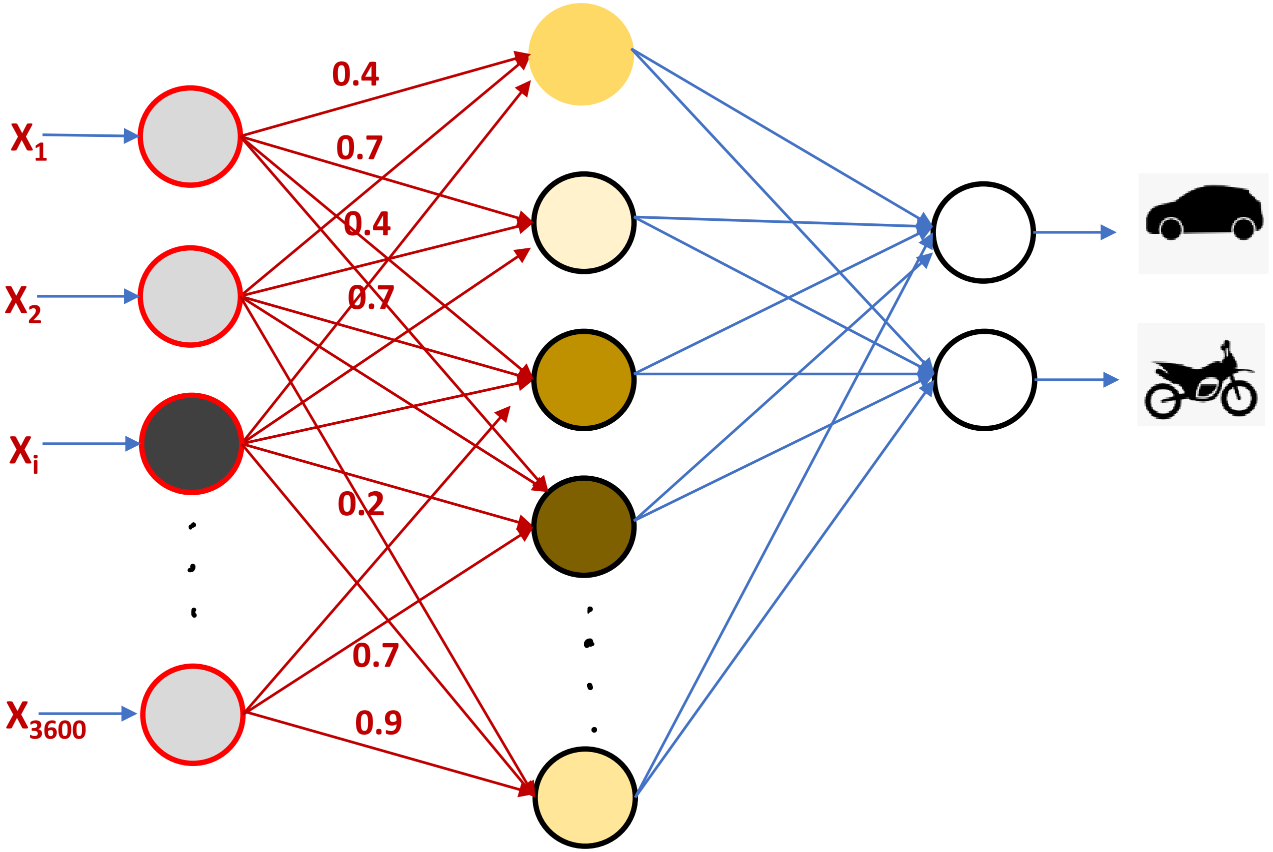
A Toy Example

60x60=3600

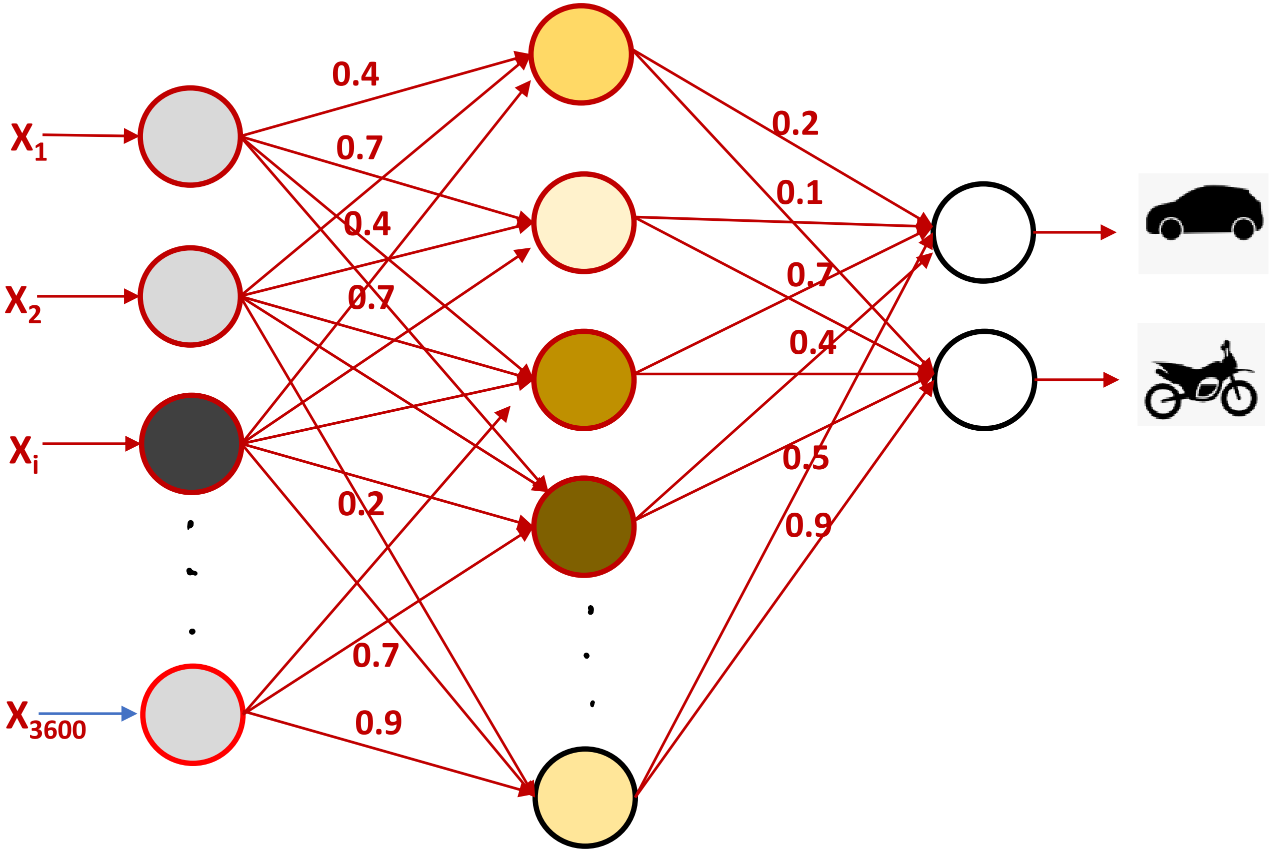


A Toy Example

60x60=3600

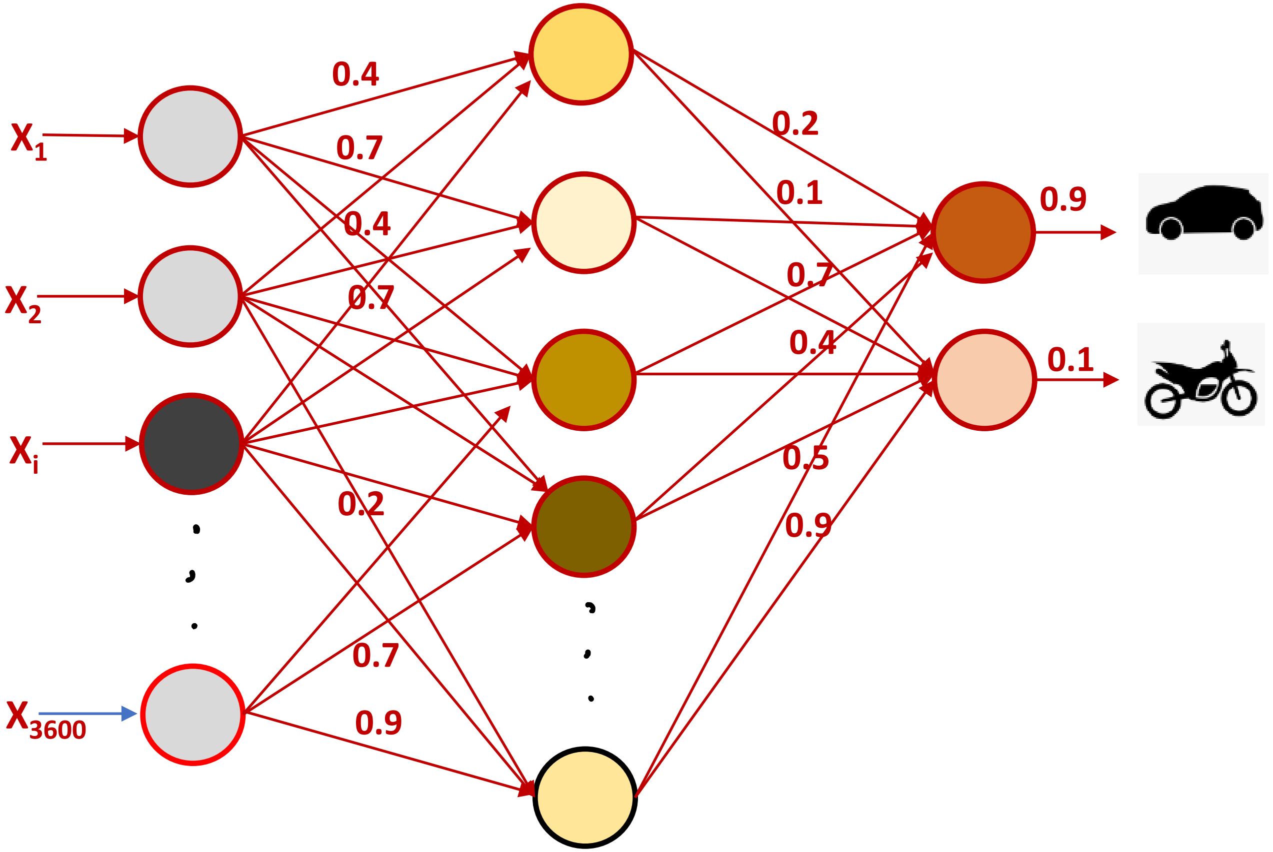


A Toy Example



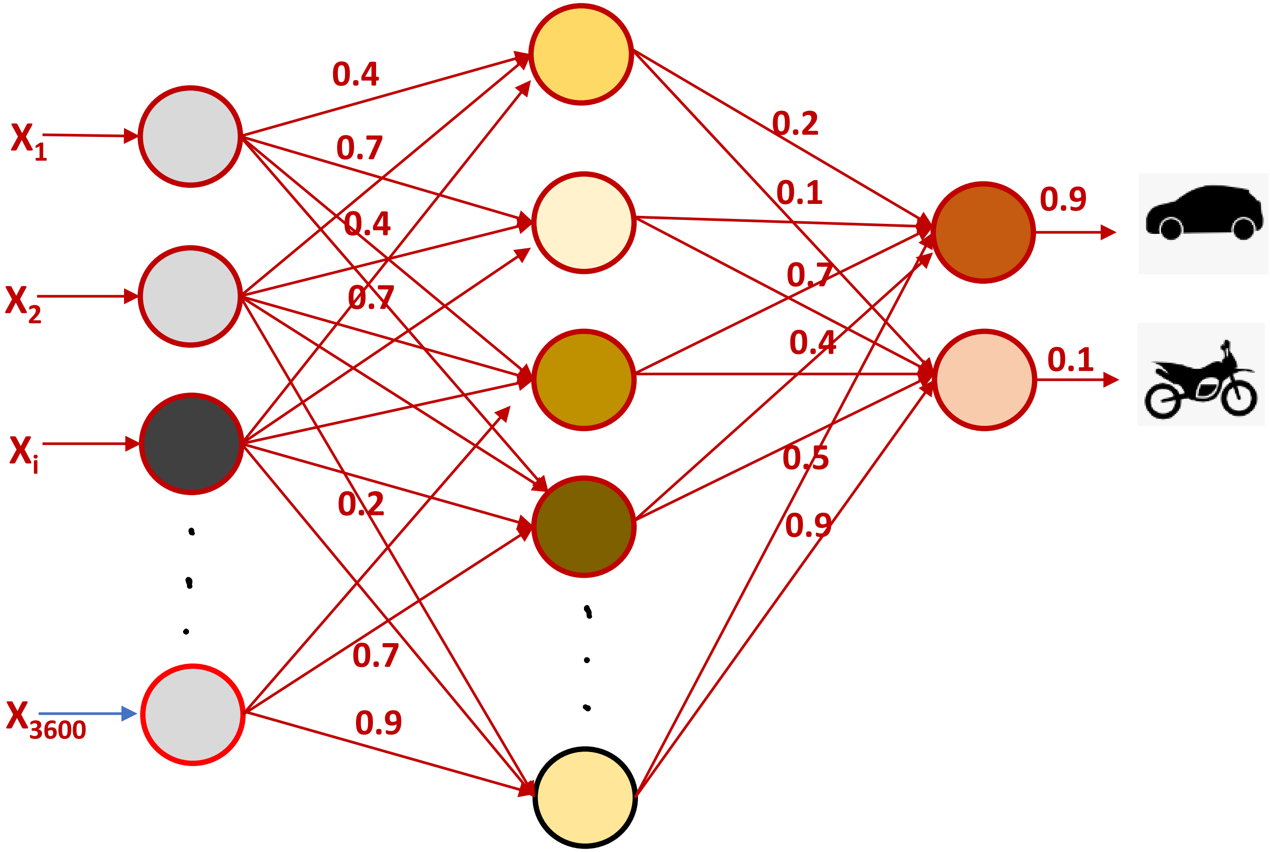
A Toy Example

60x60=3600



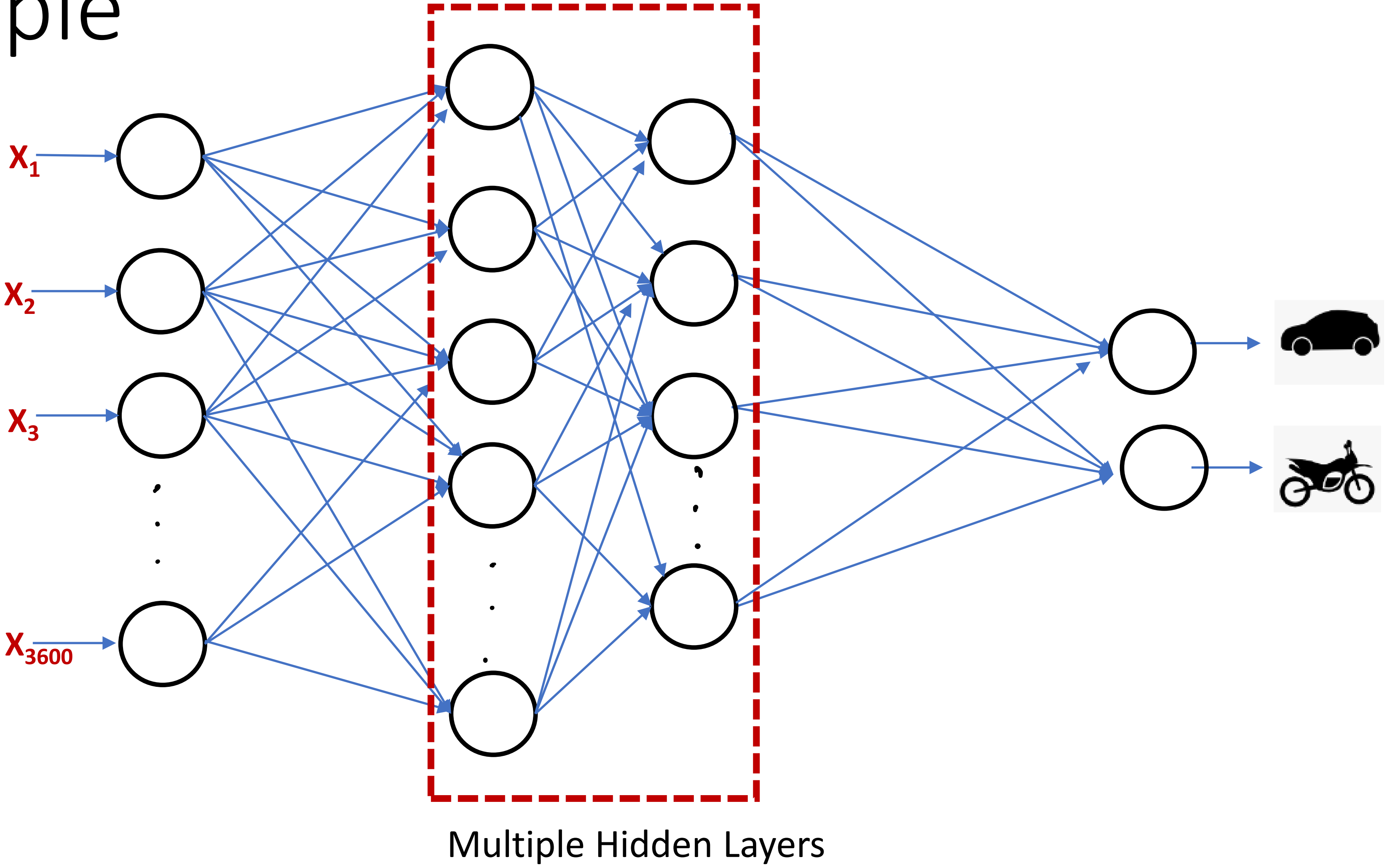
A Toy Example

60x60=3600

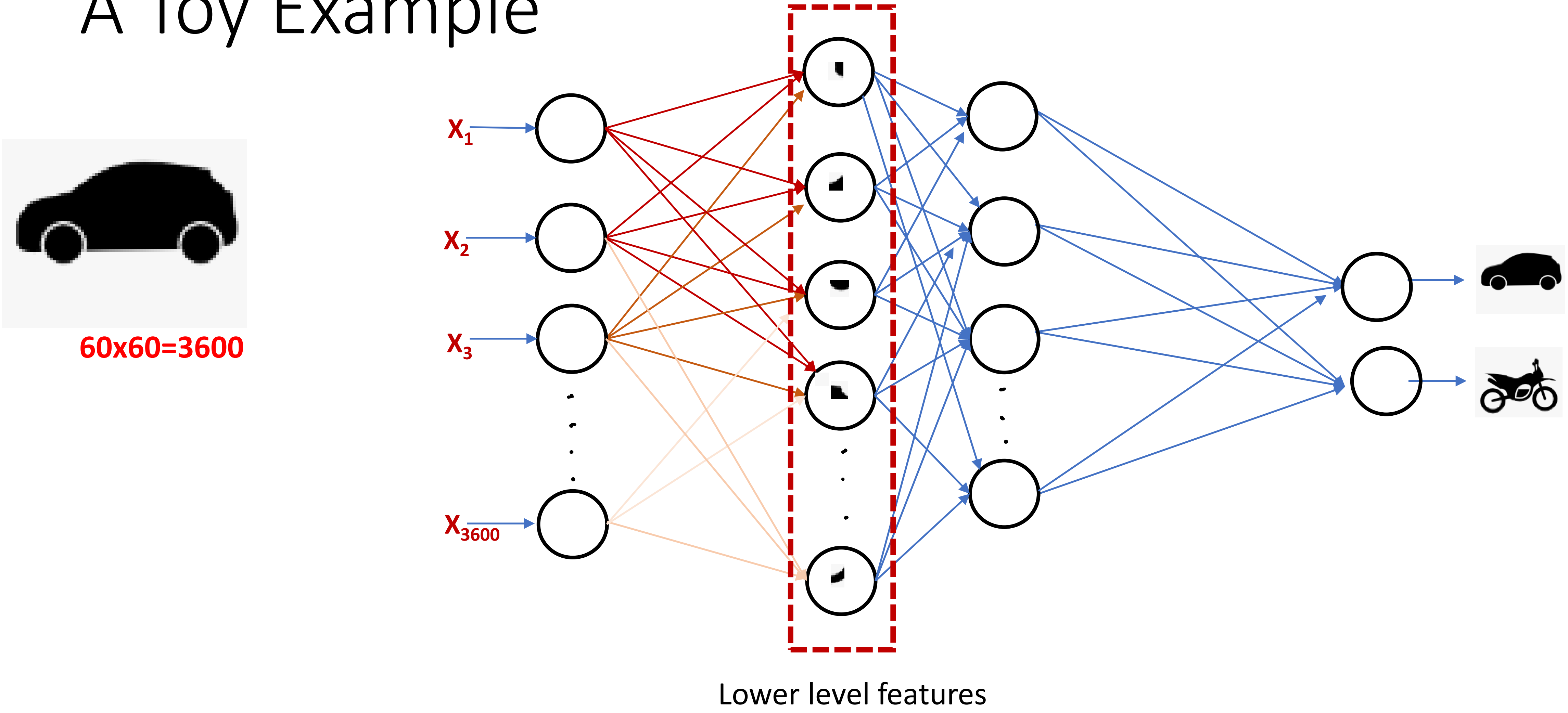


Forward pass

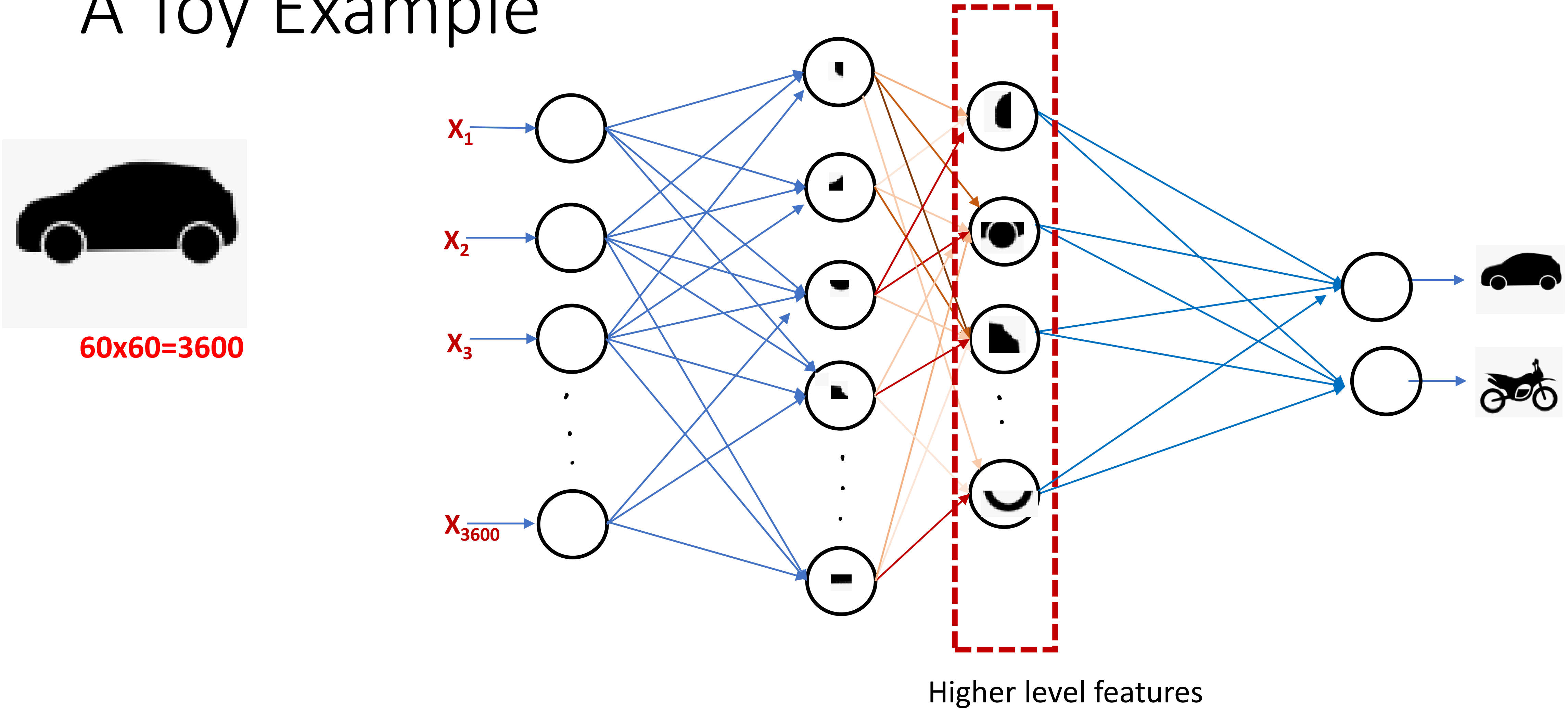
A Toy Example



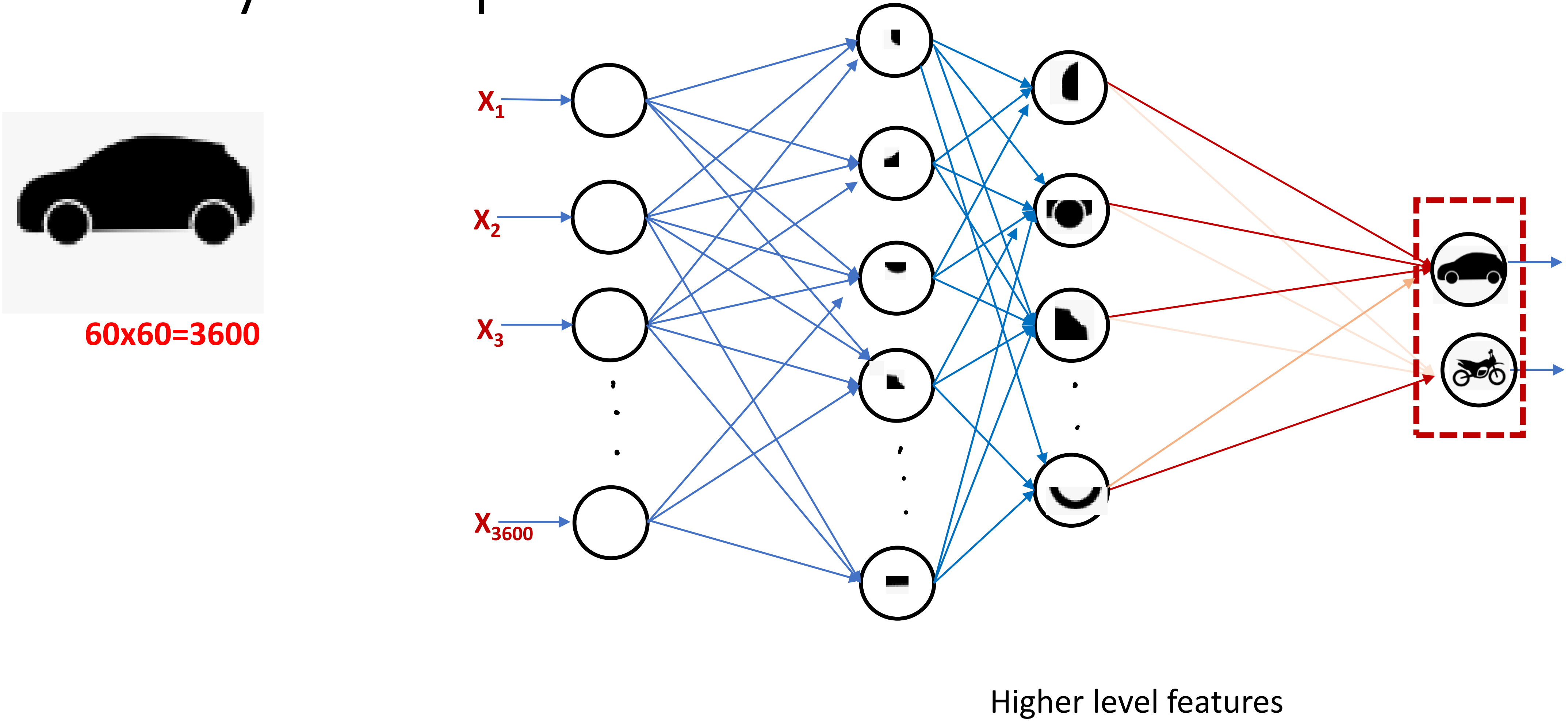
A Toy Example



A Toy Example

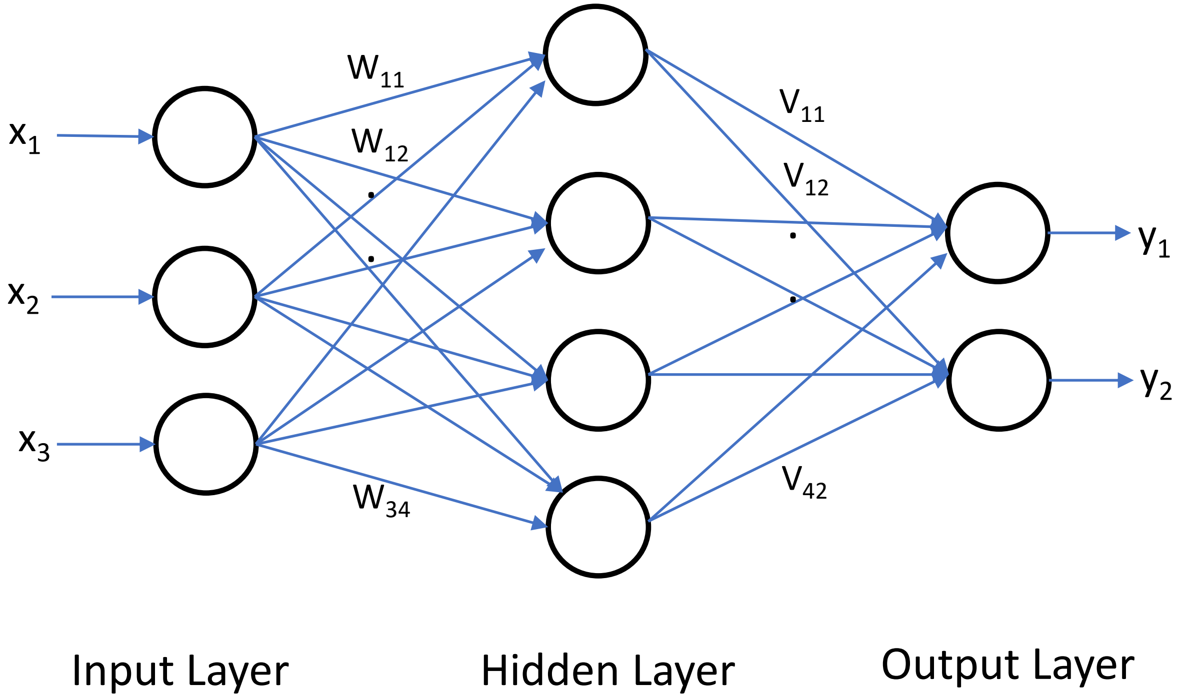


A Toy Example

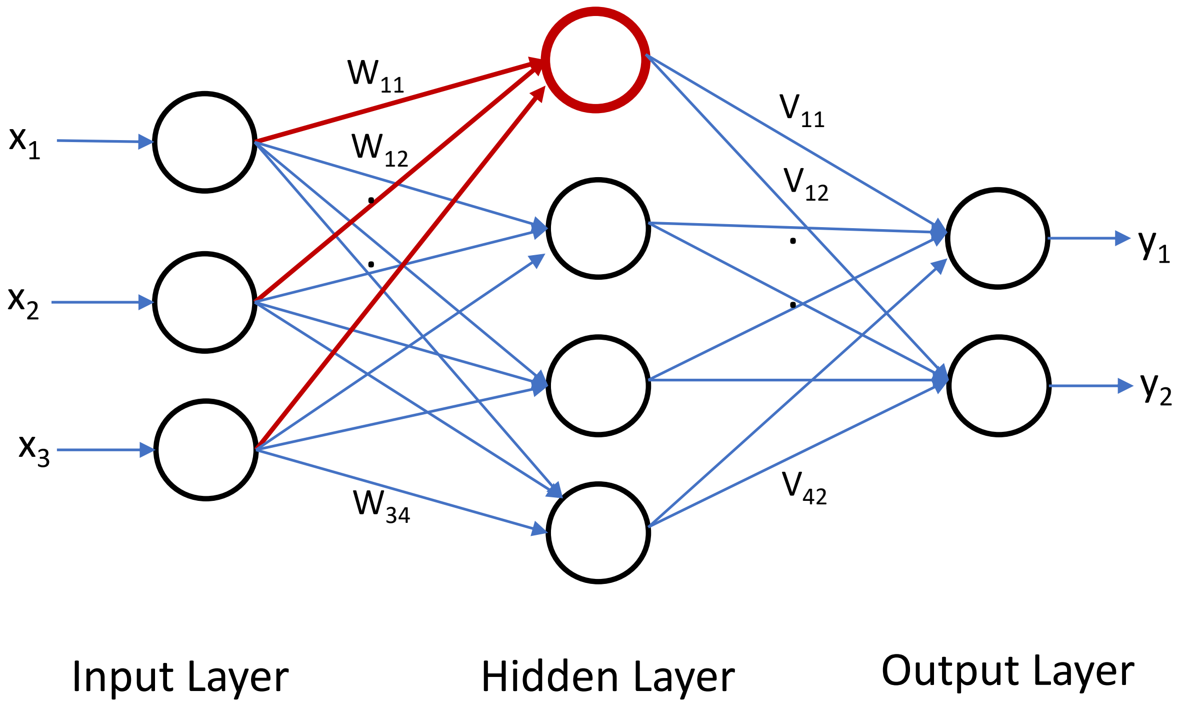


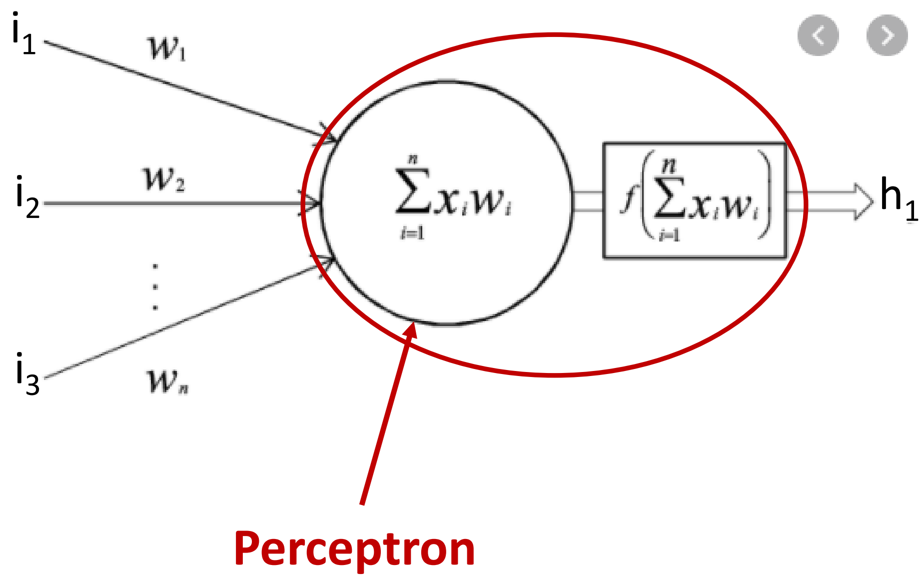
Lesson 9
Multilayer Perceptron

What is Multilayer Perceptron Neural Network?



Multilayer Perceptron

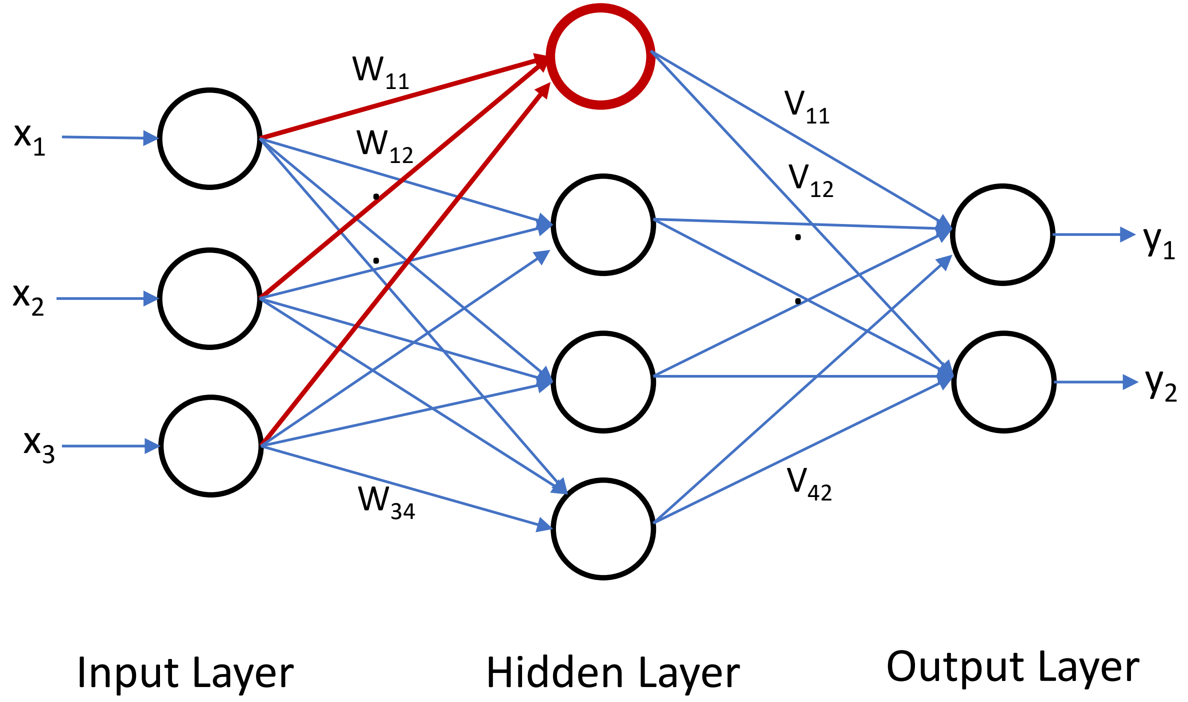


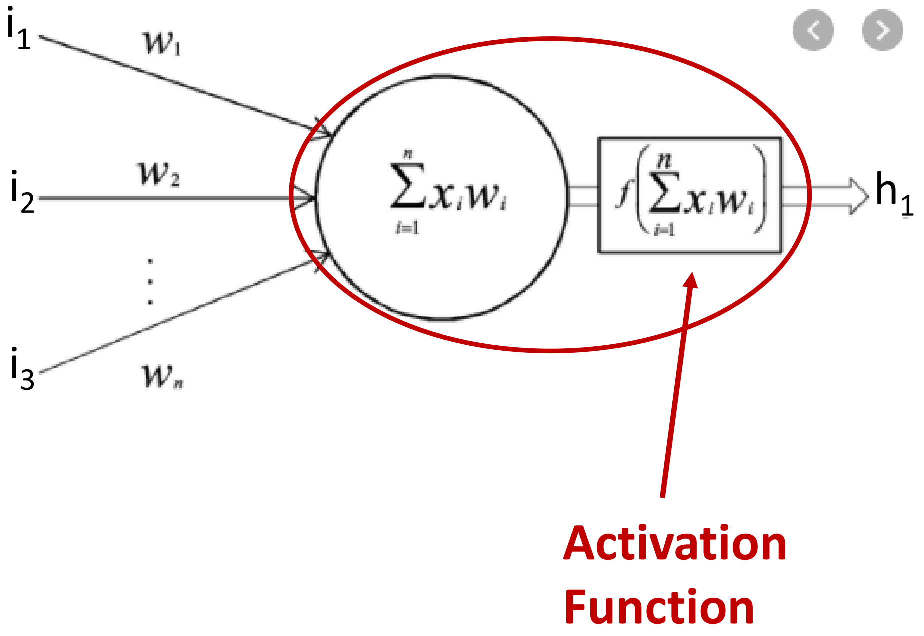


For a given node I, the perceptron is defined as weighted summation of the incoming data from the nodes of the previous layer.

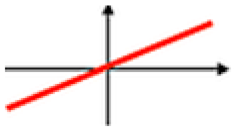
$$p_i = x_0 w_{0i} + x_2 w_{2i} + \ldots + x_n w_{ni} = \sum_{j=1}^{n} x_j w_{ji}$$

Multilayer Perceptron

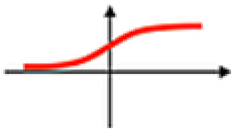




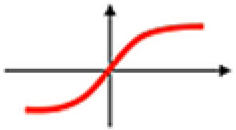
Linear $\phi(z) = z$



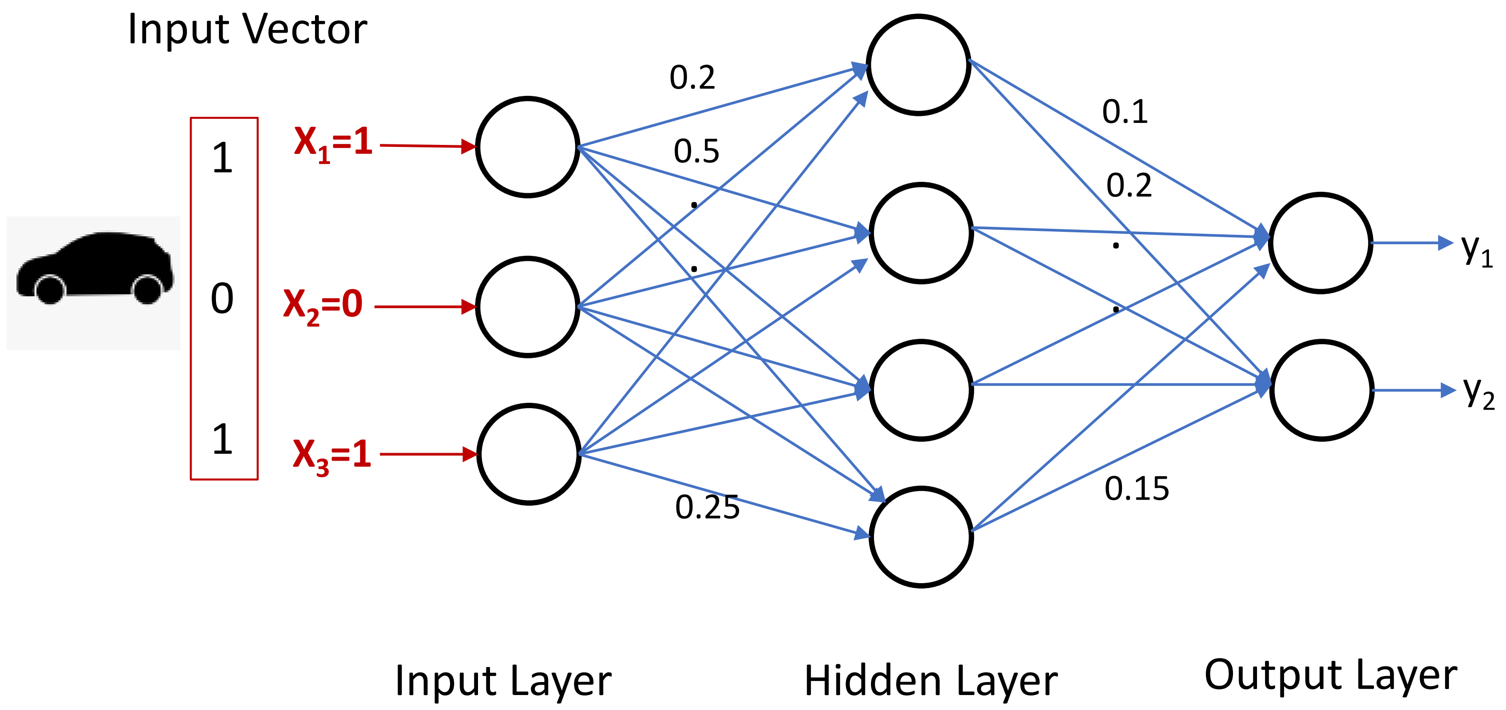
Logistic (sigmoid) $\phi(z) = \dfrac{1}{1 + e^{-z}}$



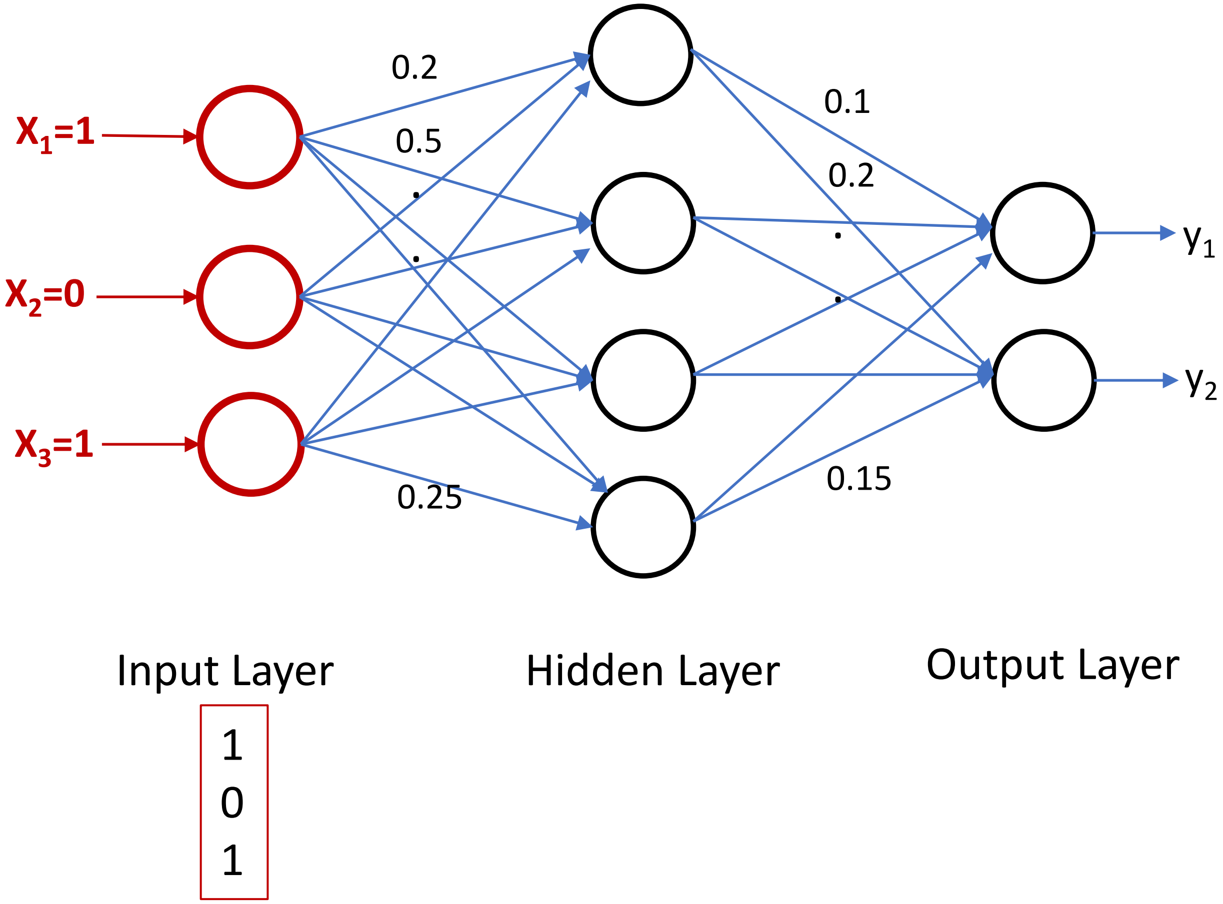
Hyperbolic tangent $\phi(z) = \dfrac{e^{z} - e^{-z}}{e^{z} + e^{-z}}$



Forward Pass



Forward Pass

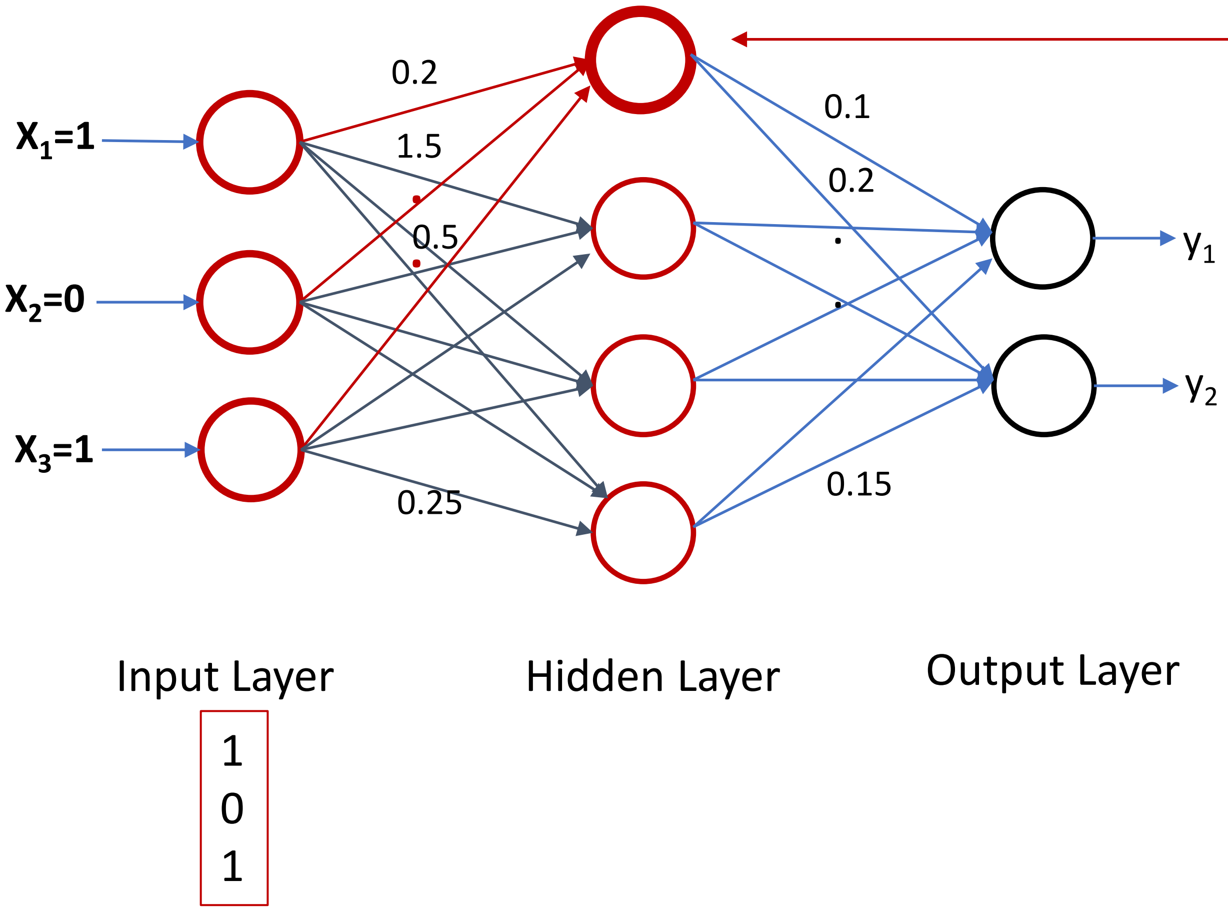


x=Perceptron(x)

Linear Activation function

x=f(x)

Forward Pass

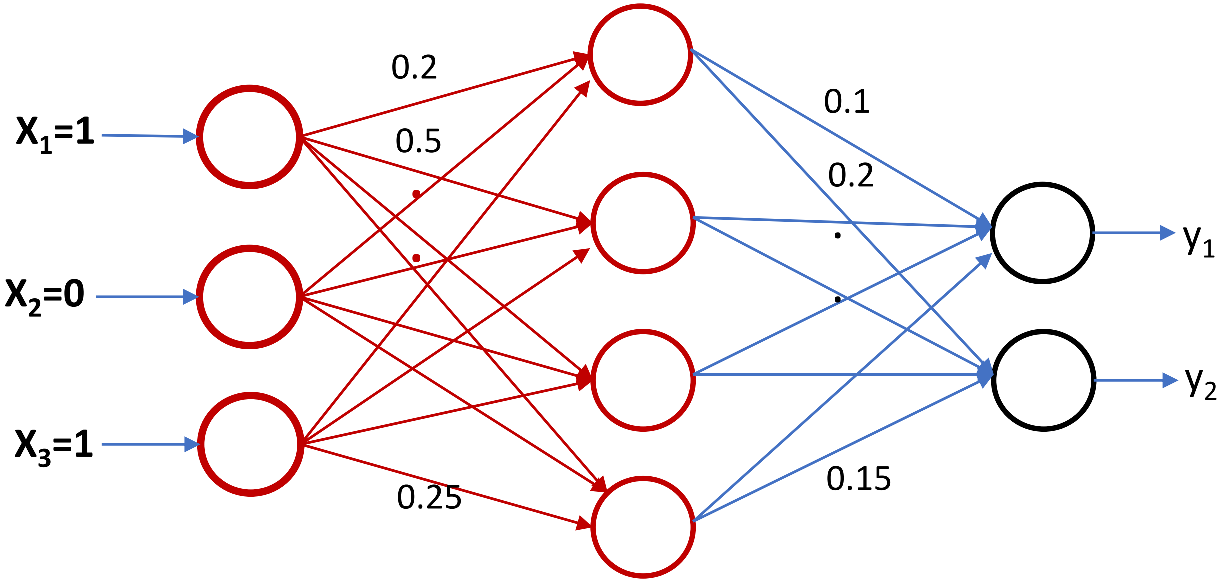


1x0.2 + 0x1.5 + 1x0.5= 0.7

Sigmoid Activation function

f(0.7)=0.668

Forward Pass



Input Layer

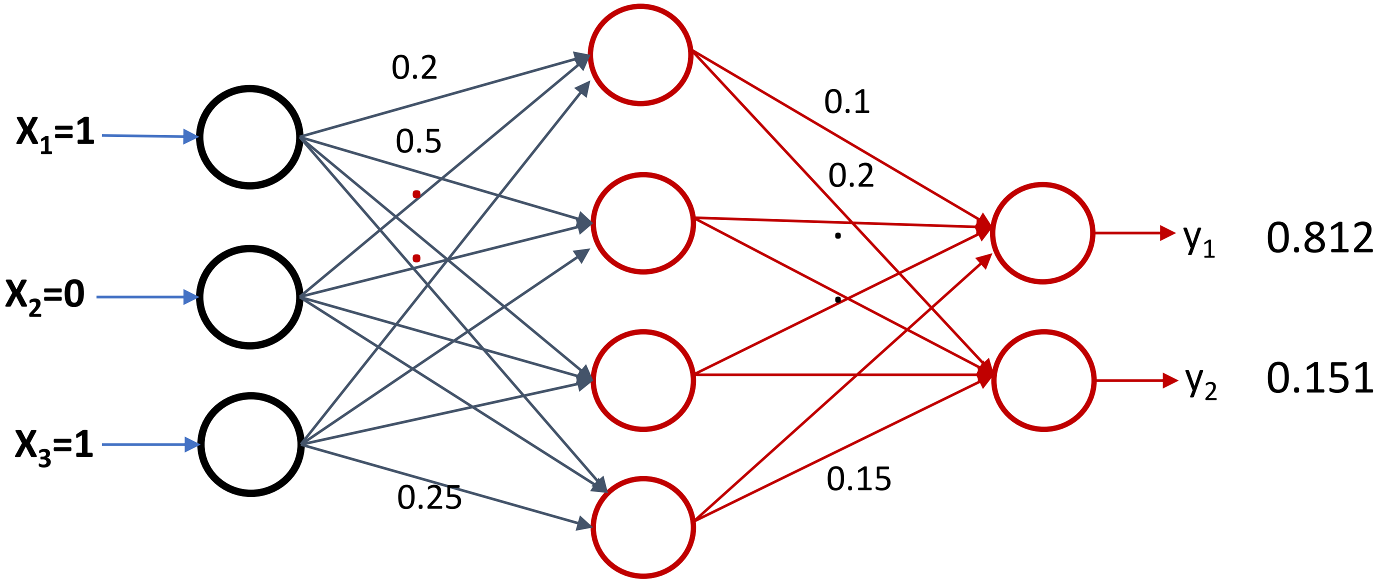
1
0
1

Hidden Layer

0.668
0.912
0.102
0.471

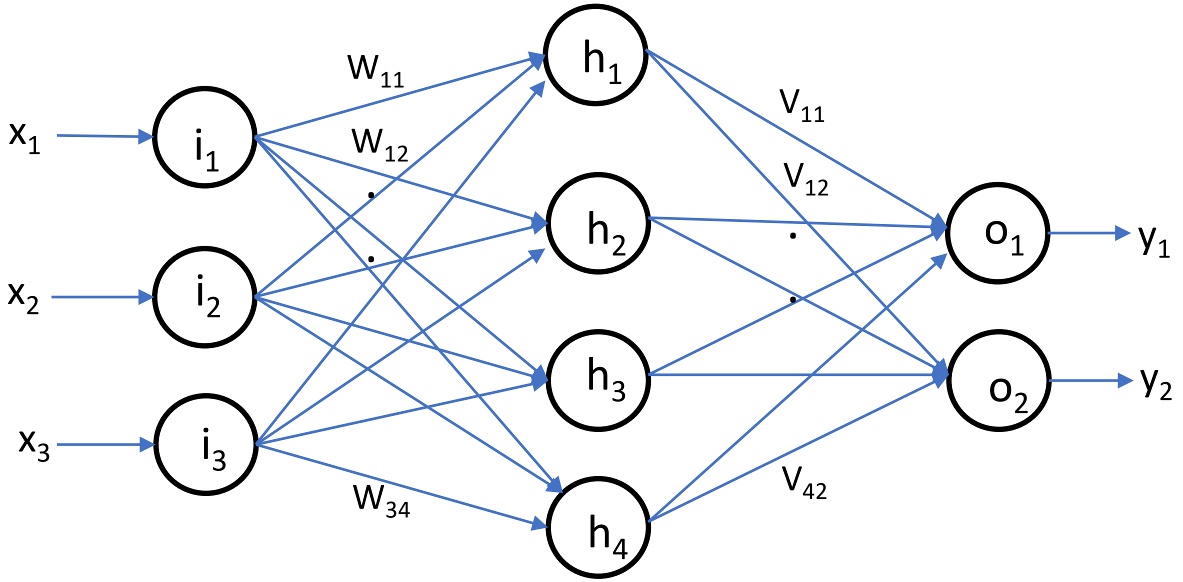
Output Layer

Forward Pass



Input Layer	Hidden Layer	Output Layer
1	0.668	0.812
0	0.912	0.151
1	0.102	
	0.471	

Weight Matrix



$$\begin{array}{c|cccc|}
 & h_1 & h_2 & h_3 & h_4 \\
\hline
i_1 & W_{11} & W_{12} & W_{13} & W_{14} \\
i_2 & W_{21} & W_{22} & W_{23} & W_{24} \\
i_3 & W_{31} & W_{32} & W_{33} & W_{34} \\
\hline
\end{array}$$

W

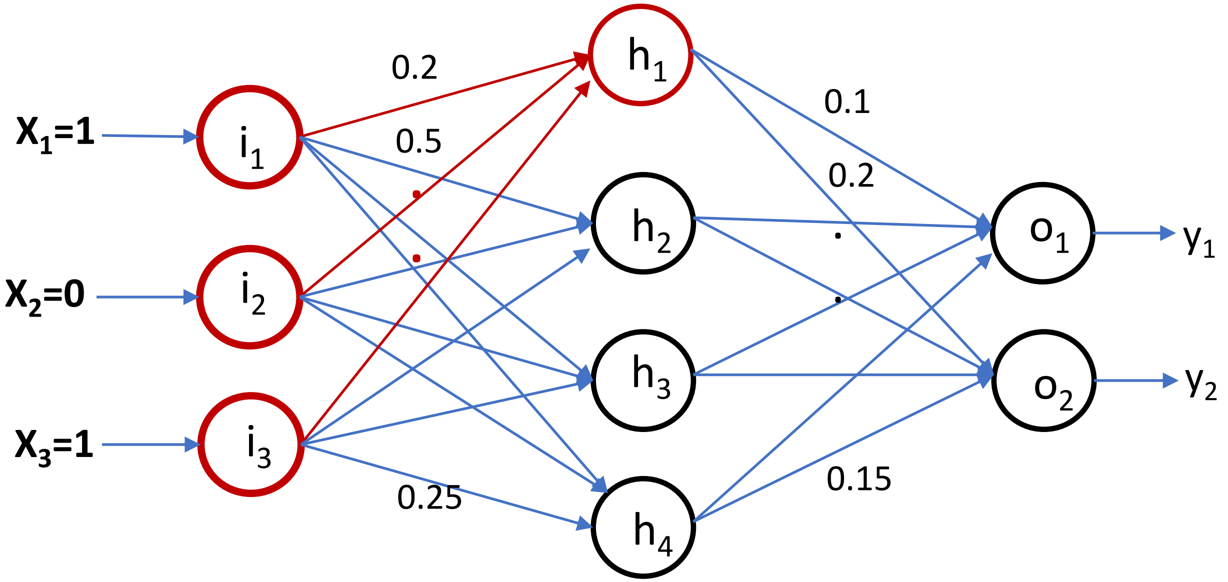
$$\begin{array}{c|cc|}
 & o_1 & o_2 \\
\hline
h_1 & & \\
h_2 & & \\
h_3 & & \\
h_4 & & \\
\hline
\end{array}$$

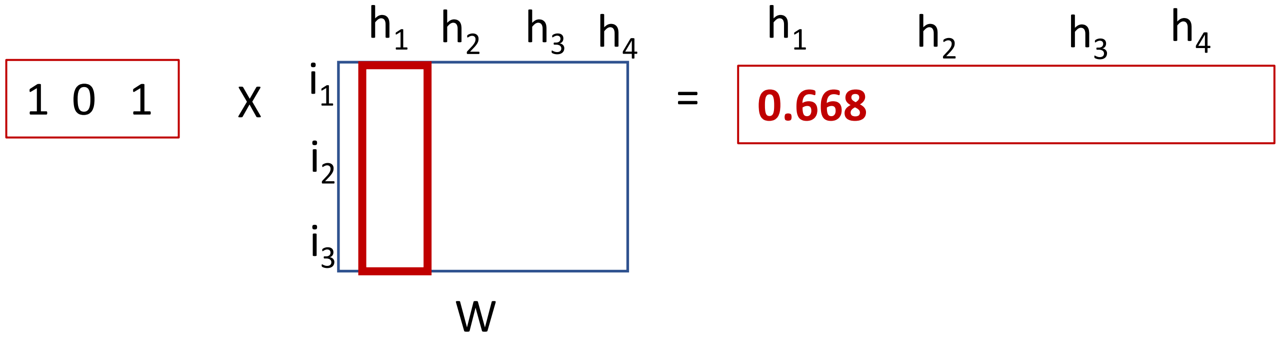
V

Forward Pass

$$p_1 = x_1 W_{11} + x_2 W_{21} + x_3 W_{31}$$

$$h_1 = Sigmoid(p_1)$$

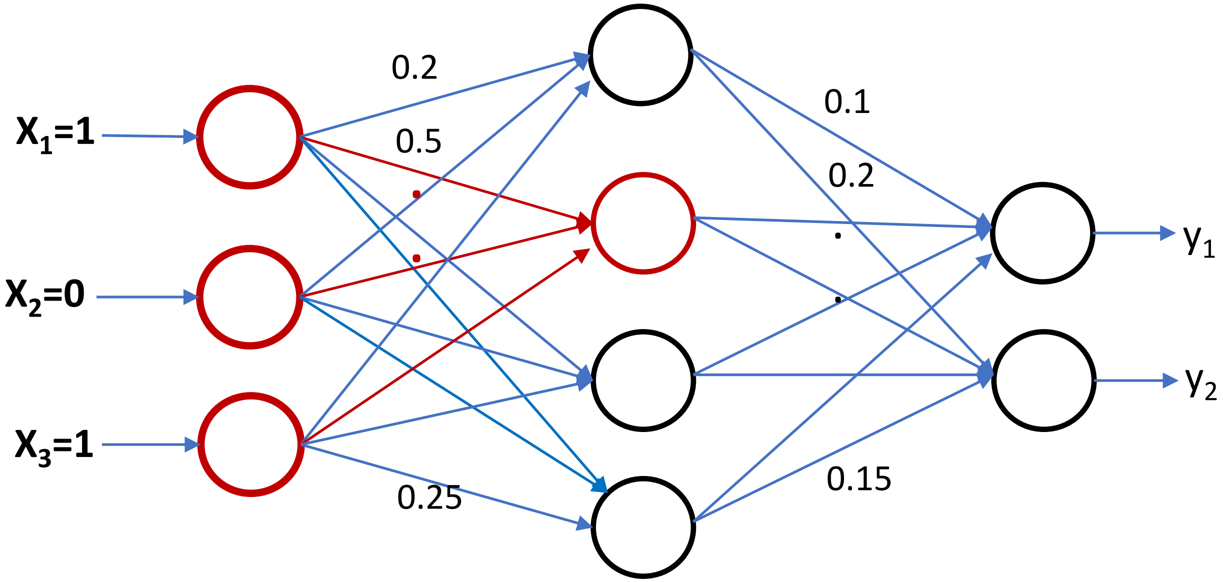


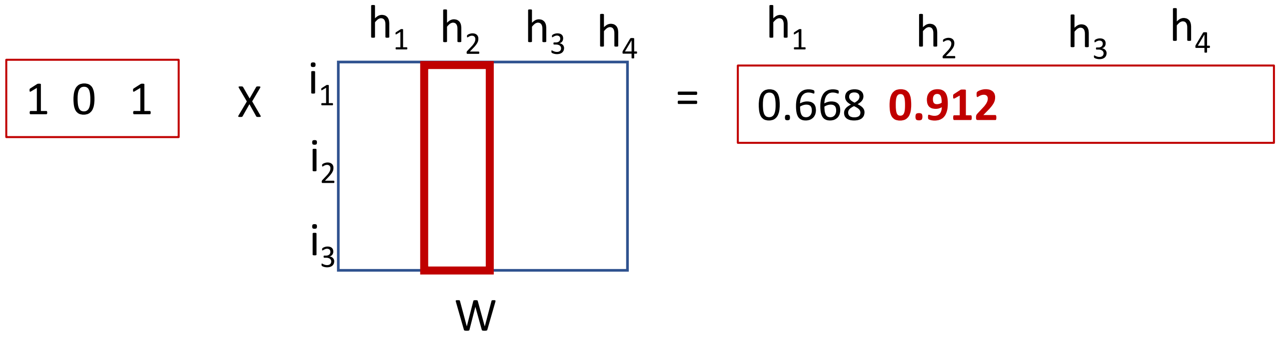


Forward Pass

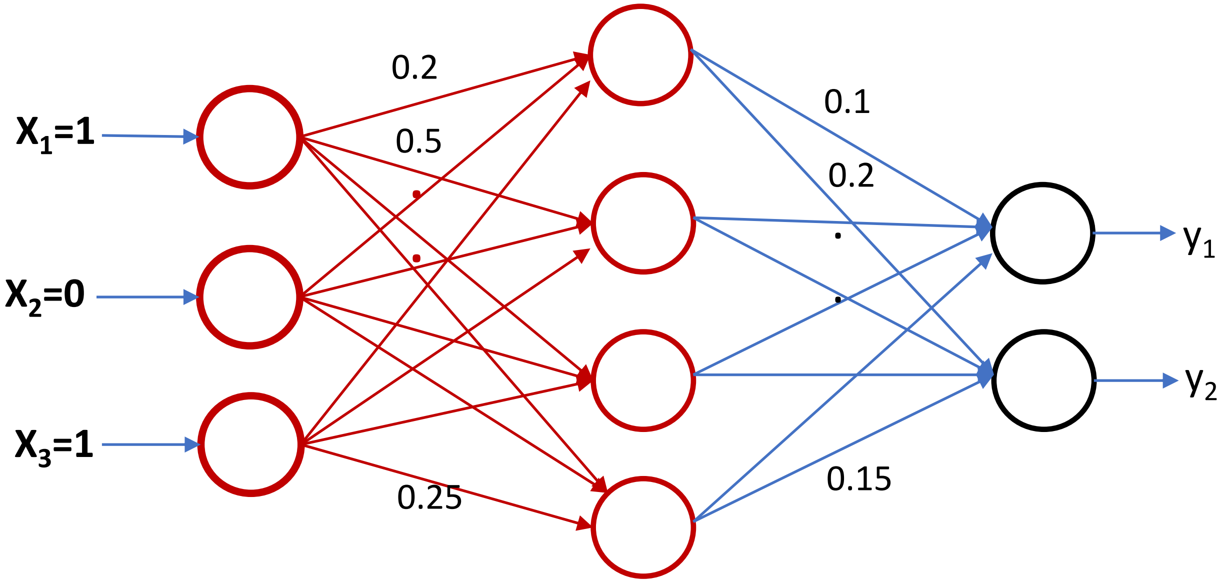
$$p_2 = x_1 W_{12} + x_2 W_{22} + x_3 W_{32}$$

$$h_2 = Sigmoid(p_2)$$





Forward Pass

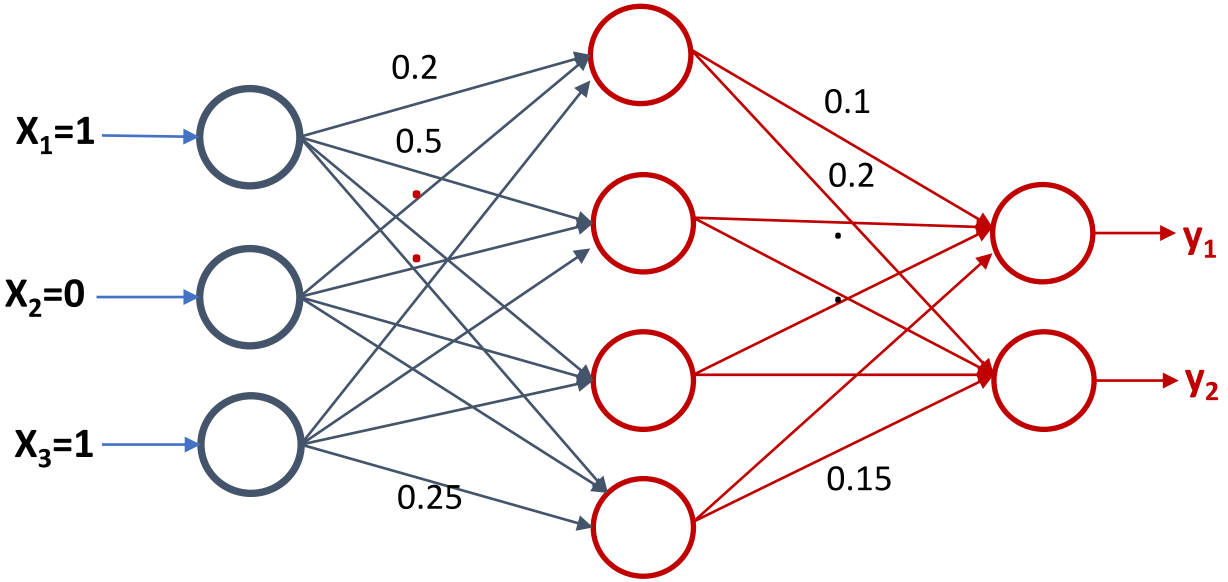


$$\bar{p}^T = \bar{x}^T . W$$

$$\bar{h}^T = Sigmoid(\bar{p}^T)$$

$$\underbrace{\boxed{1 \quad 0 \quad 1}}_{\bar{x}^T} \times \underbrace{\begin{array}{c|cccc} & h_1 & h_2 & h_3 & h_4 \\ \hline i_1 & & & & \\ i_2 & & & & \\ i_3 & & & & \end{array}}_{W} = \underbrace{\begin{array}{cccc} h_1 & h_2 & h_3 & h_4 \\ \hline 0.668 & 0.912 & 0.102 & 0.471 \end{array}}_{\bar{h}^T}$$

Forward Pass

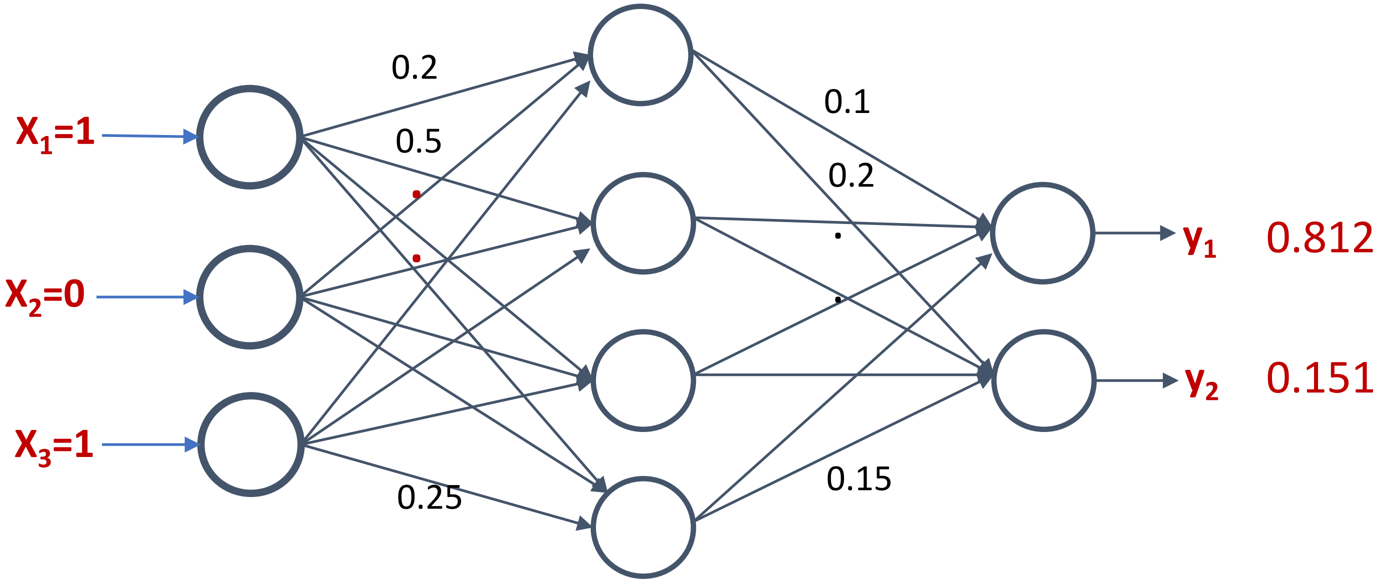


$$\bar{p}^T = \bar{h}^T . V$$

$$\bar{y}^T = Sigmoid(\bar{p}^T)$$

$$\underbrace{\boxed{0.668 \;\; 0.912 \;\; 0.102 \;\; 0.471}}_{\bar{h}^T} \;\; \text{X} \;\; \underbrace{\begin{array}{c|cc|} & o_1 & o_2 \\ h_1 & & \\ h_2 & & \\ h_3 & & \\ h_4 & & \end{array}}_{V} \;\; = \;\; \underbrace{\boxed{0.812 \;\; 0.151}}_{\bar{y}^T}$$

Forward Pass

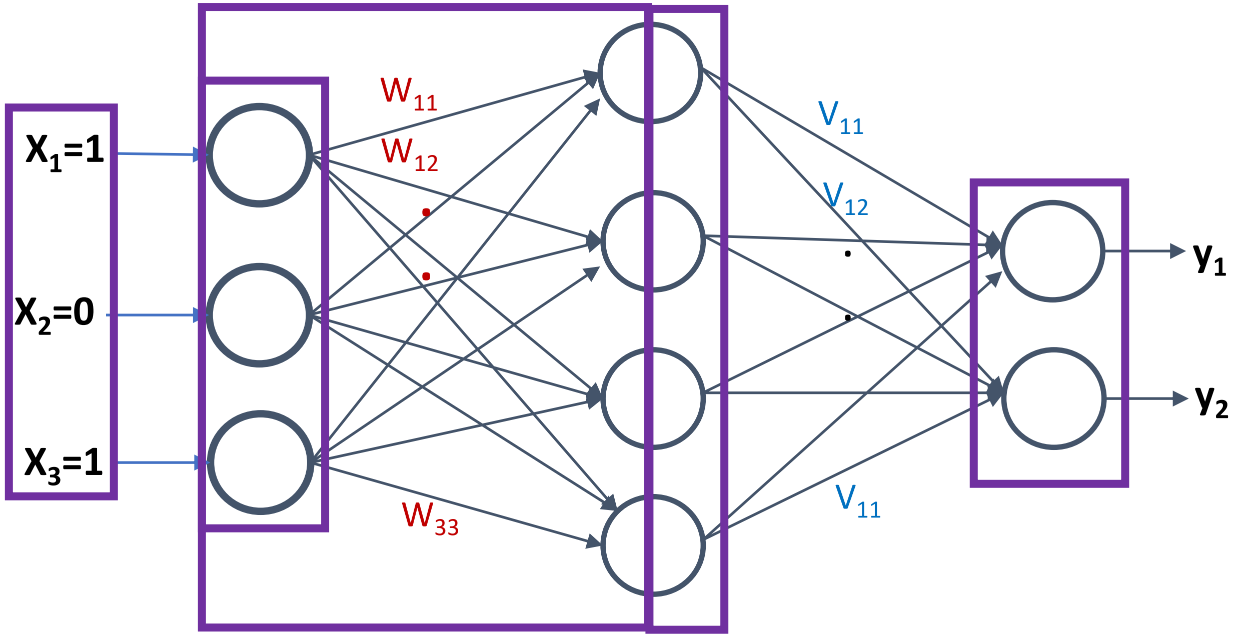


$$\bar{y}^T = Sigmoid(\, Sigmoid(\bar{x}^T W)^T . V \,)$$

Lesson 11

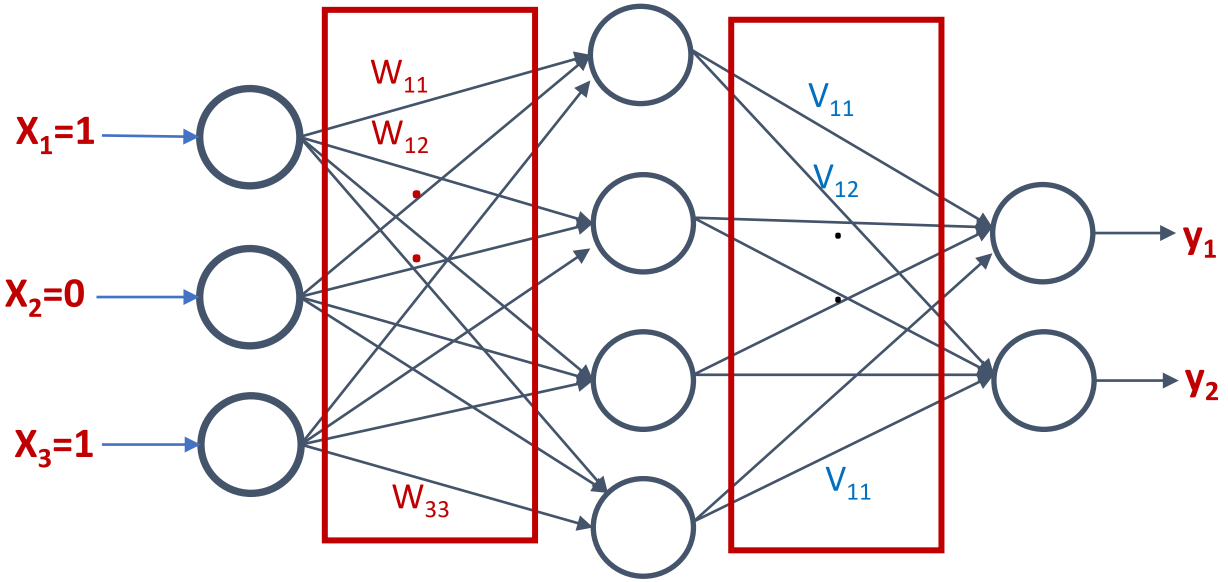
Learning the parameters
– Backpropagation through time

What are the parameters?



$$\boxed{\bar{y}^T} = \boxed{f_{hidden}\left(\boxed{f_{output}\left(\boxed{\bar{x}^T \boxed{\boldsymbol{W}}}\right)^T}.\boxed{\boldsymbol{V}}\right)}$$

What are the parameters?



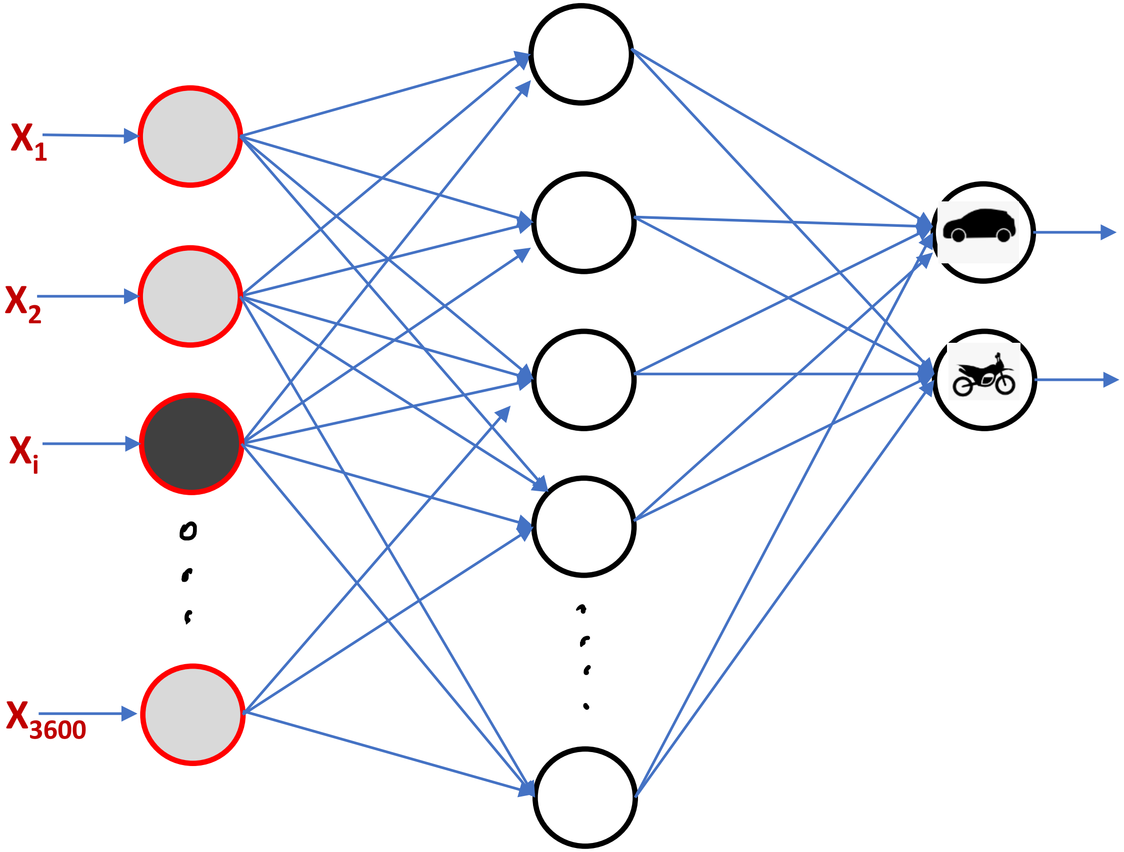
$$
\begin{array}{c|cccc|}
 & h_1 & h_2 & h_3 & h_4 \\
\hline
i_1 & W_{11} & W_{12} & W_{13} & W_{14} \\
i_2 & W_{21} & W_{22} & W_{23} & W_{24} \\
i_3 & W_{31} & W_{32} & W_{33} & W_{34} \\
\hline
\end{array}
$$

W

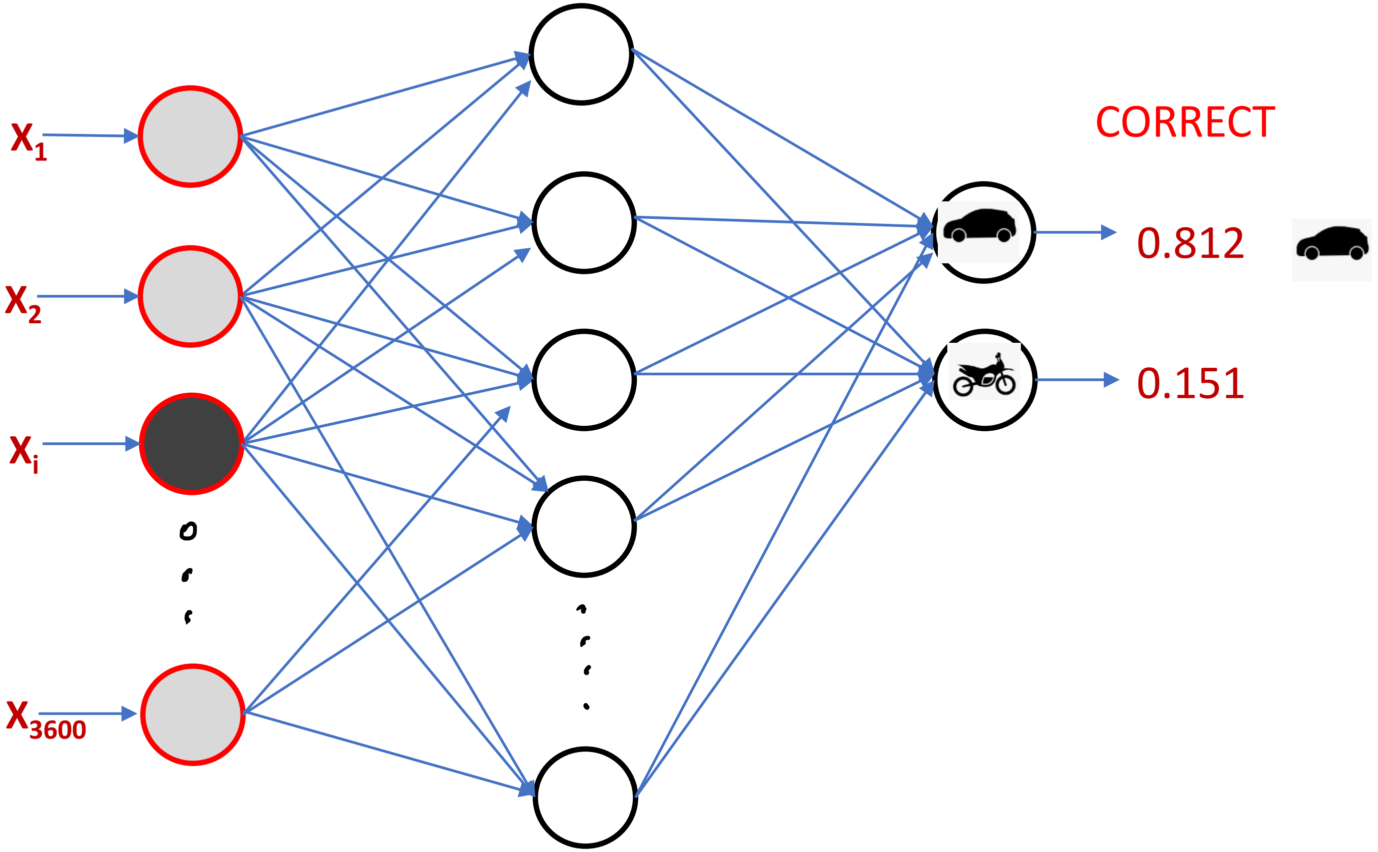
$$
\begin{array}{c|cc|}
 & o_1 & o_2 \\
\hline
h_1 & V_{11} & V_{12} \\
h_2 & V_{21} & V_{22} \\
h_3 & V_{31} & V_{32} \\
h_4 & V_{41} & V_{42} \\
\hline
\end{array}
$$

V

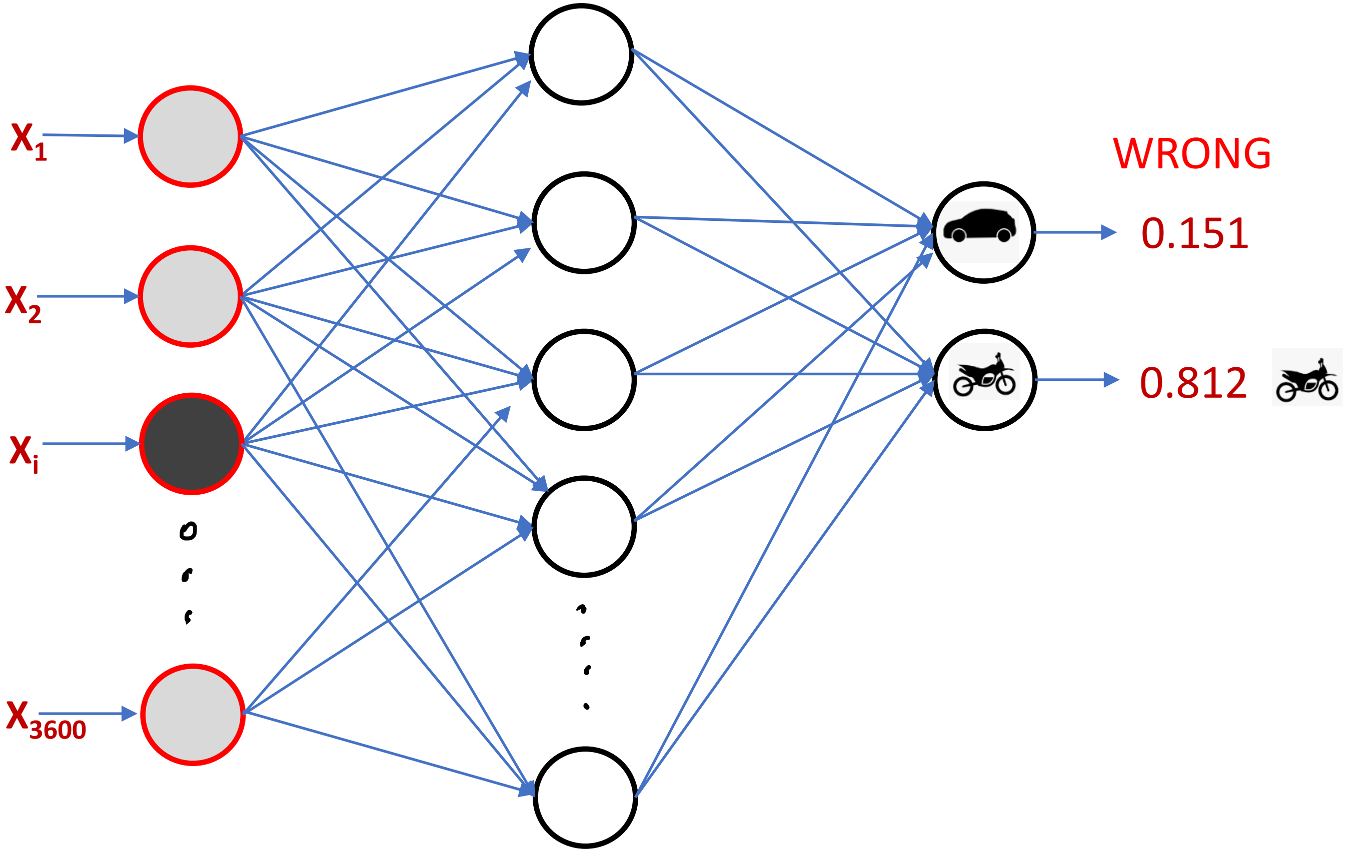
60x60=3600



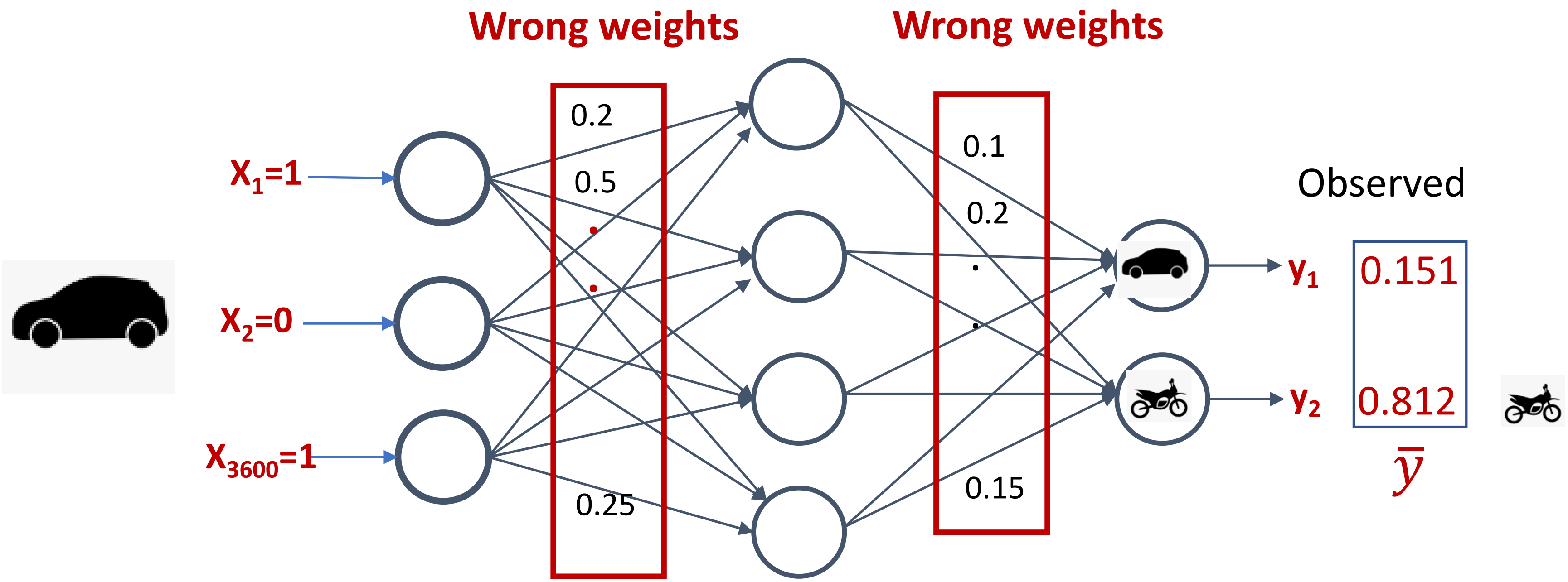
60x60=3600

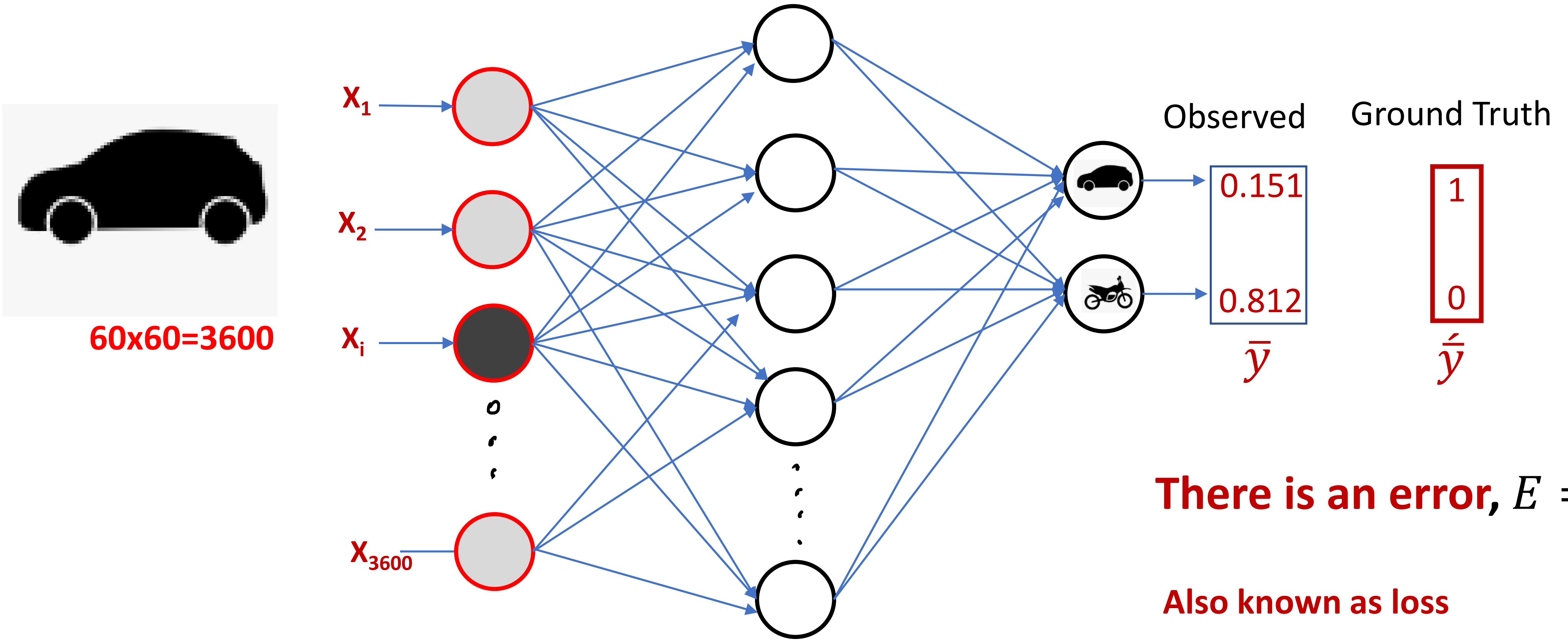


60x60=3600



Why there is an error in prediction?

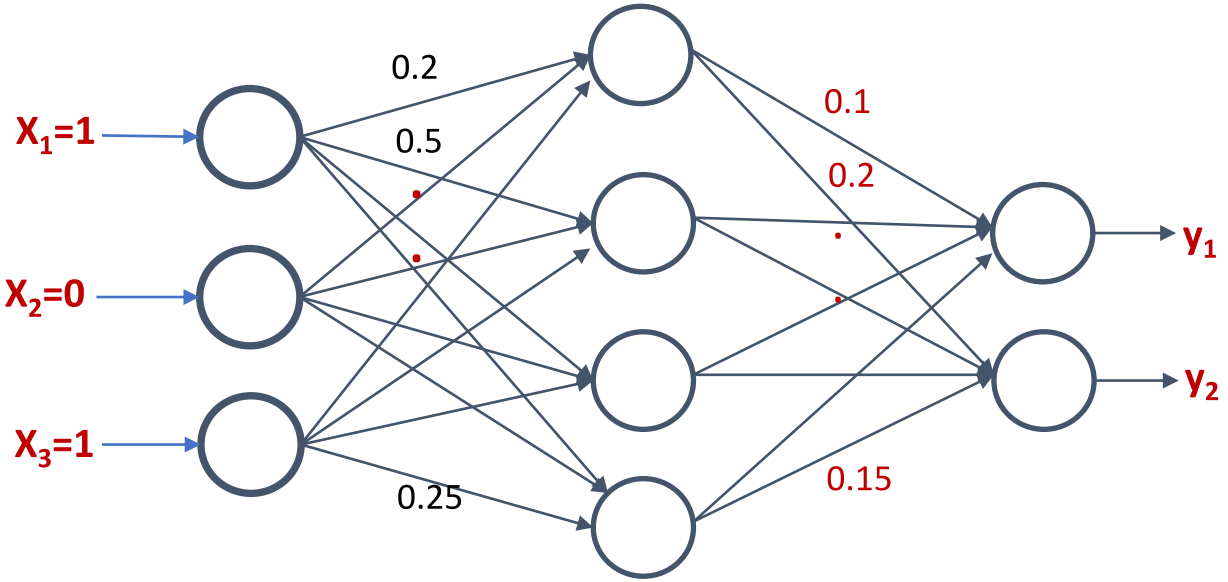




There is an error, $E = \|\bar{y} - \acute{y}\|$

Also known as loss

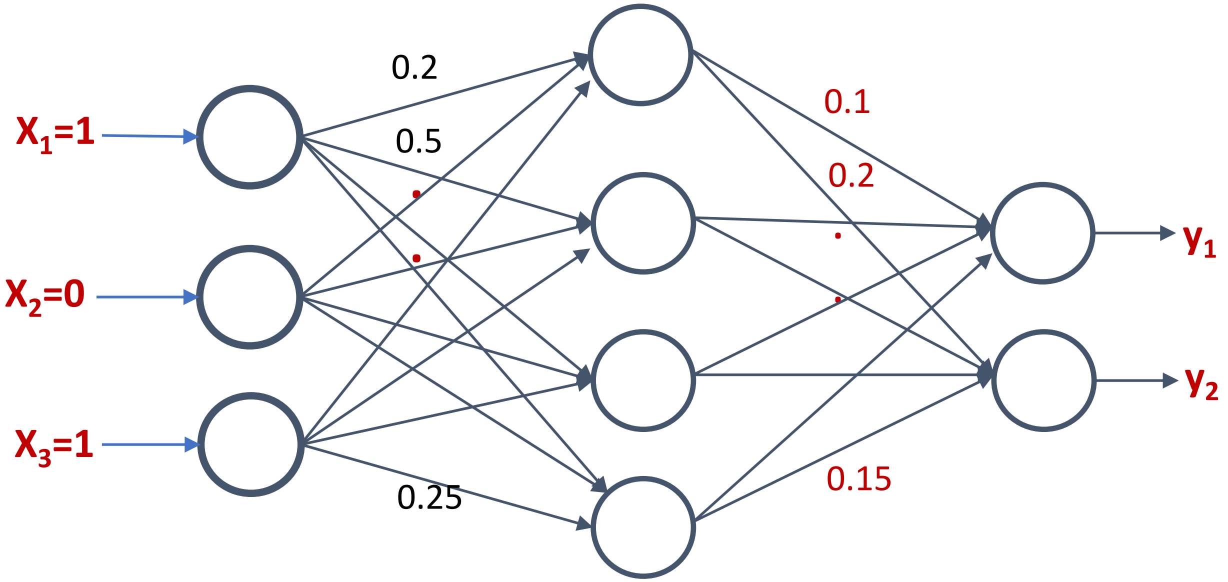
Backpropagation Through Time



Minimize the Loss function $E = \|\bar{y} - \acute{\bar{y}}\|$

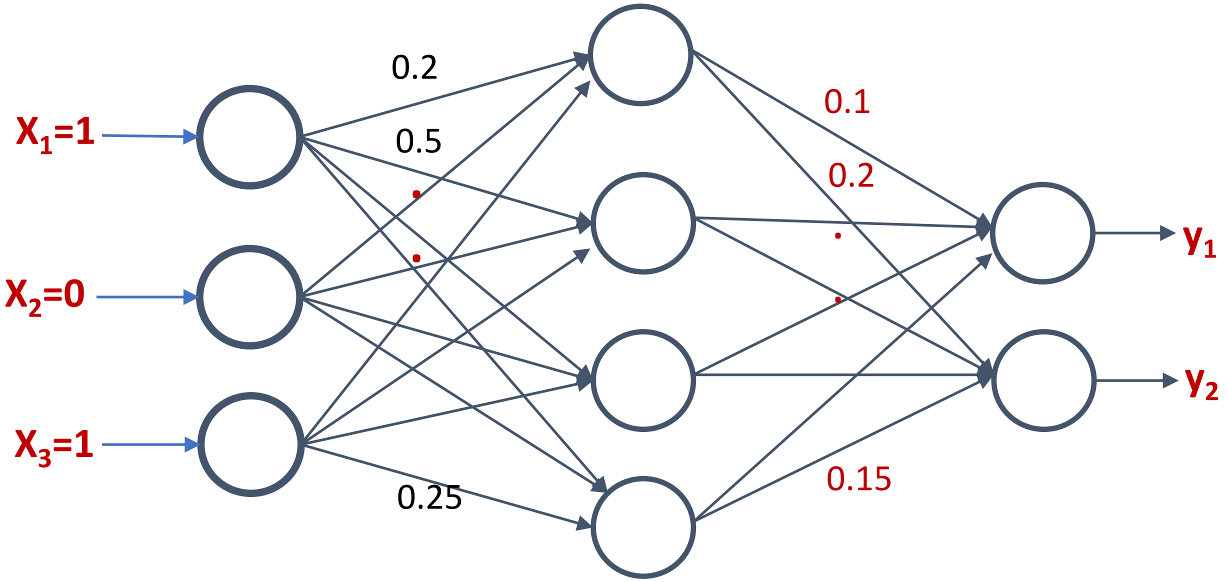
Backpropagation Through Time

Minimize the Loss function $E = \|\bar{y} - \acute{y}\|$



Backpropagation Through Time

Minimize the Loss function $E = \|\bar{y} - \acute{y}\|$

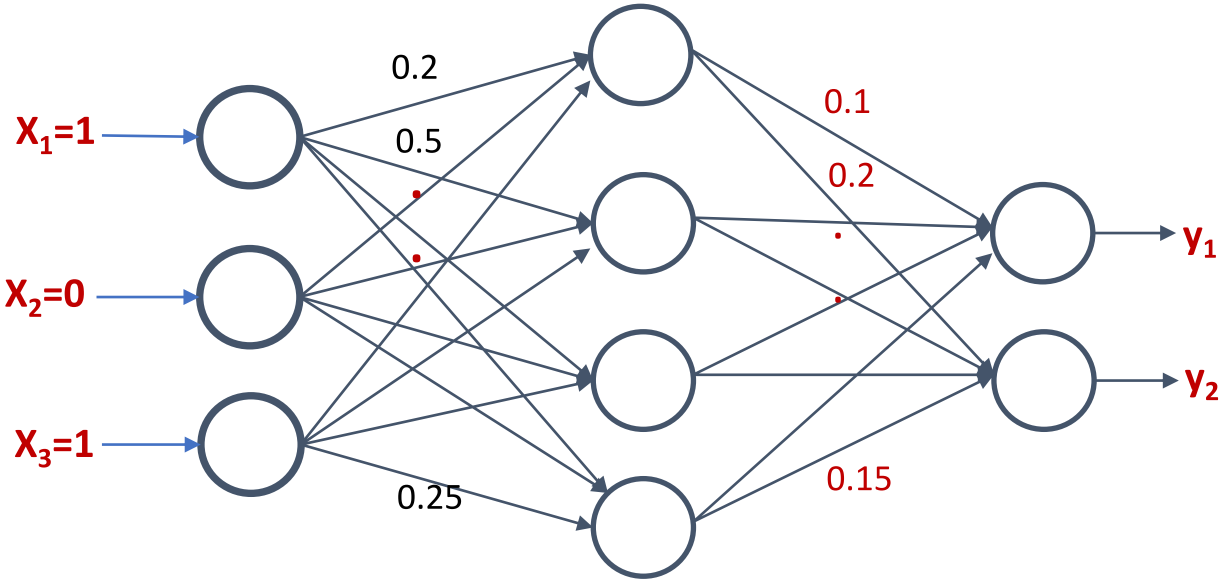


$$\nabla = \frac{\delta E}{\delta V} = 0$$

$$\nabla_{ij} = \frac{\delta E}{\delta V_{ij}} = \frac{\delta \|\bar{y} - \acute{y}\|}{\delta V_{ij}} = 0$$

Backpropagation Through Time

Minimize the Loss function $E = \|\bar{y} - \acute{y}\|$



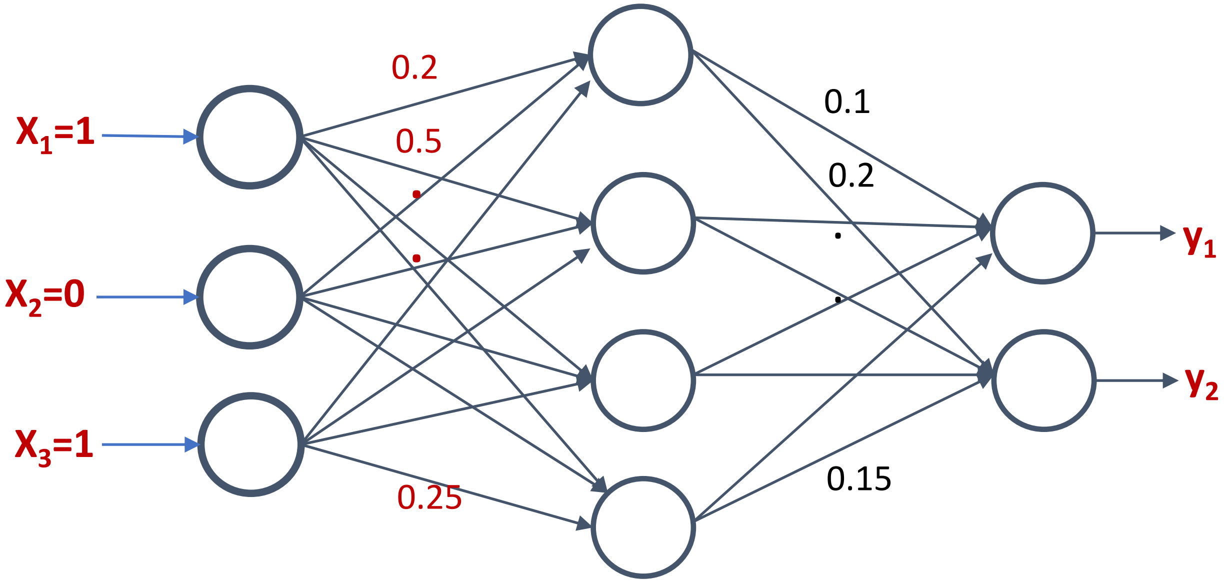
$$\nabla = \frac{\delta E}{\delta V} = 0$$

$$\nabla_{ij} = \frac{\delta E}{\delta V_{ij}} = \frac{\delta \|\bar{y} - \acute{y}\|}{\delta V_{ij}} = 0$$

$$V_{ij}^{t} = V_{ij}^{t-1} + \eta \, \nabla_{ij}^{t}$$

$$\boldsymbol{\eta = [0, 1]}$$

Backpropagation Through Time

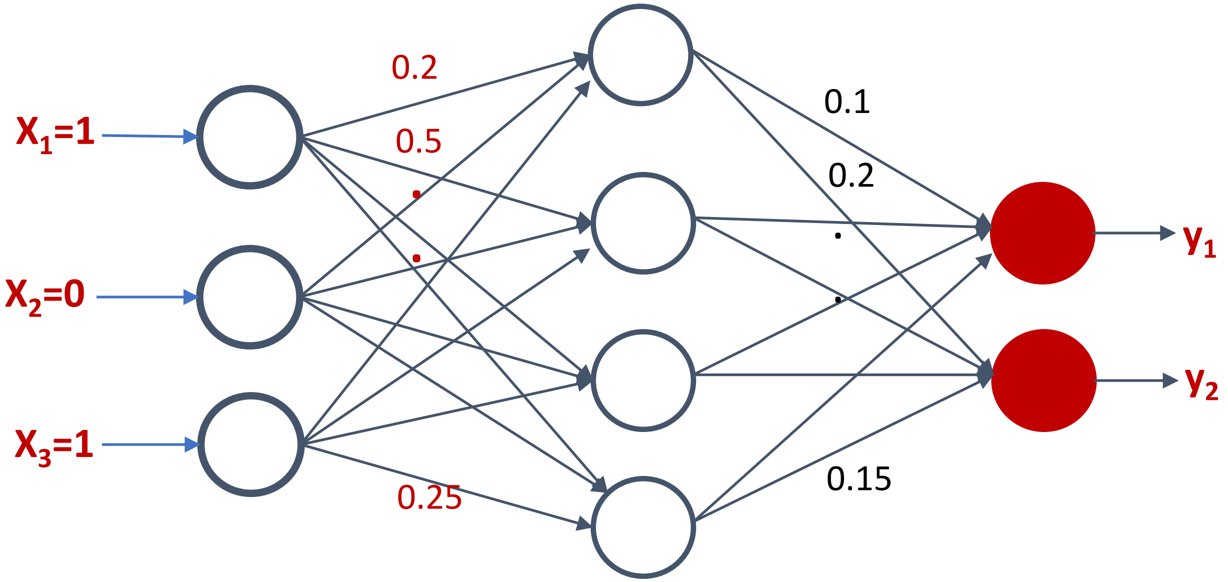


$E = \|\bar{y} - \acute{\bar{y}}\| = 0$

or

$E = \|\bar{y} - \acute{\bar{y}}\| \leq \epsilon$

Backpropagation Through Time

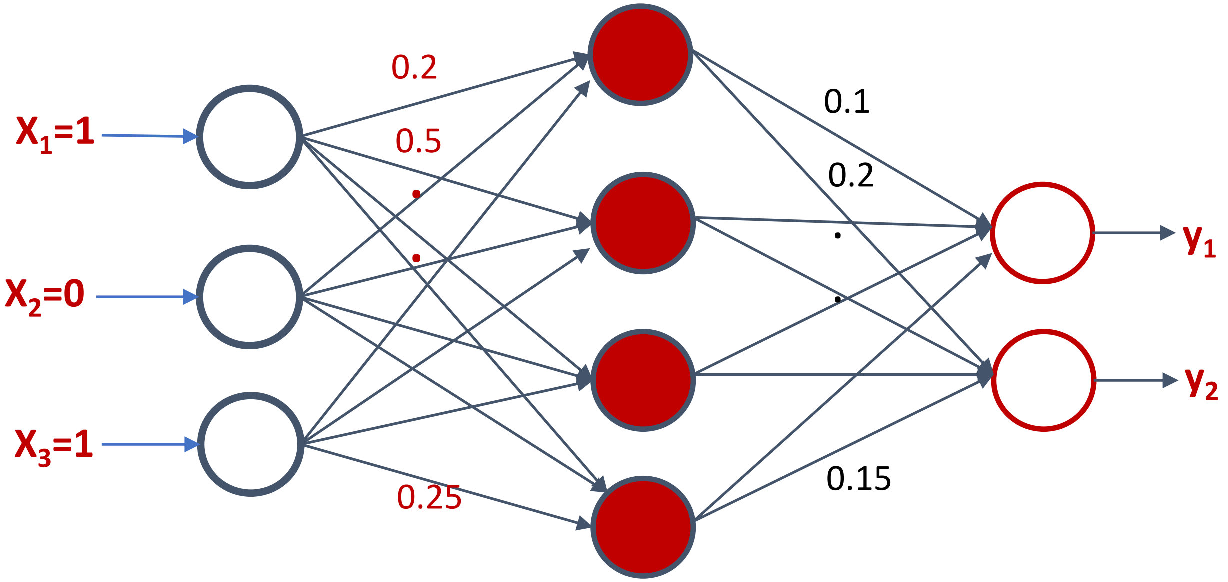


$$E = \|\bar{y} - \acute{\bar{y}}\| = 0$$

or

$$E = \|\bar{y} - \acute{\bar{y}}\| \leq \epsilon$$

Backpropagation Through Time

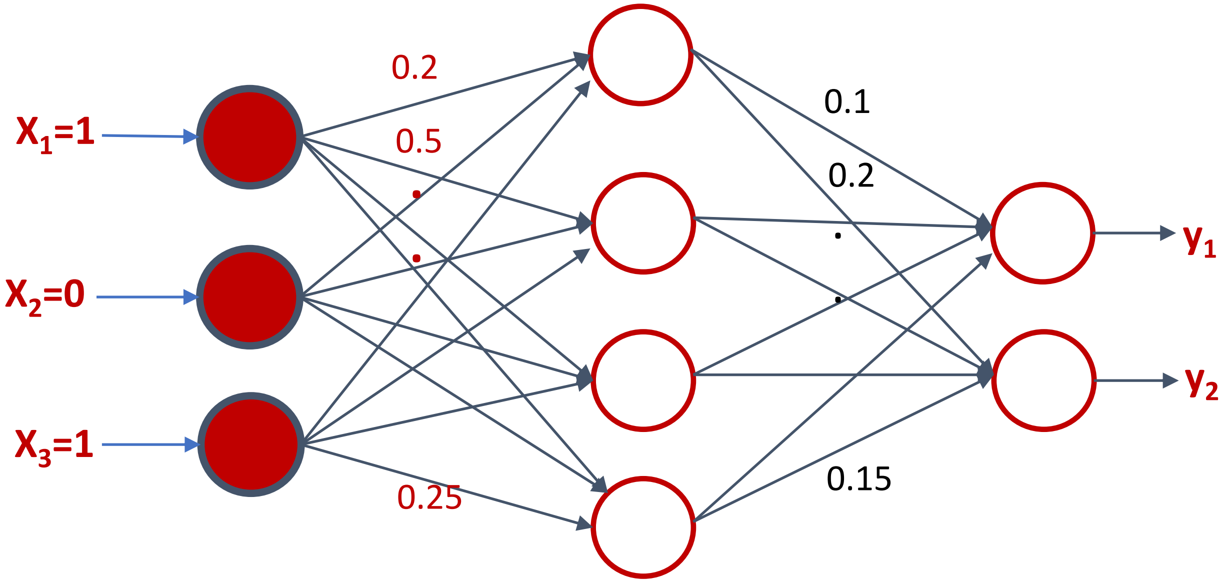


$E = \|\bar{y} - \acute{\bar{y}}\| = 0$

or

$E = \|\bar{y} - \acute{\bar{y}}\| \leq \epsilon$

Backpropagation Through Time



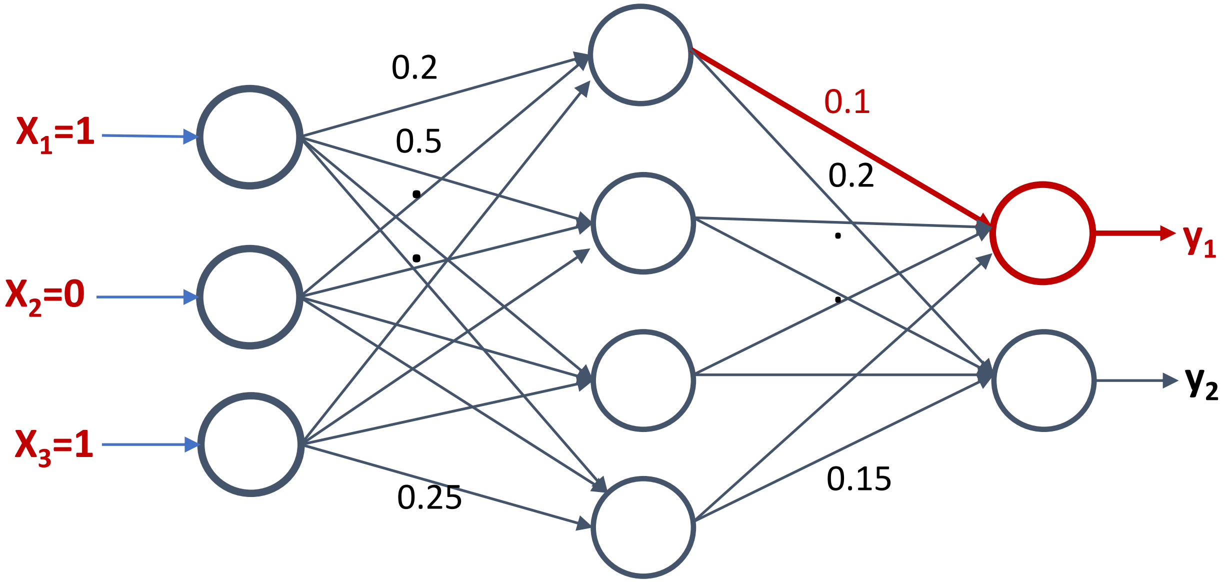
$$E = \|\bar{y} - \acute{\bar{y}}\| = 0$$

or

$$E = \|\bar{y} - \acute{\bar{y}}\| \leq \epsilon$$

How are the Derivatives performed

Loss function $E = \|\bar{y} - \acute{y}\|$

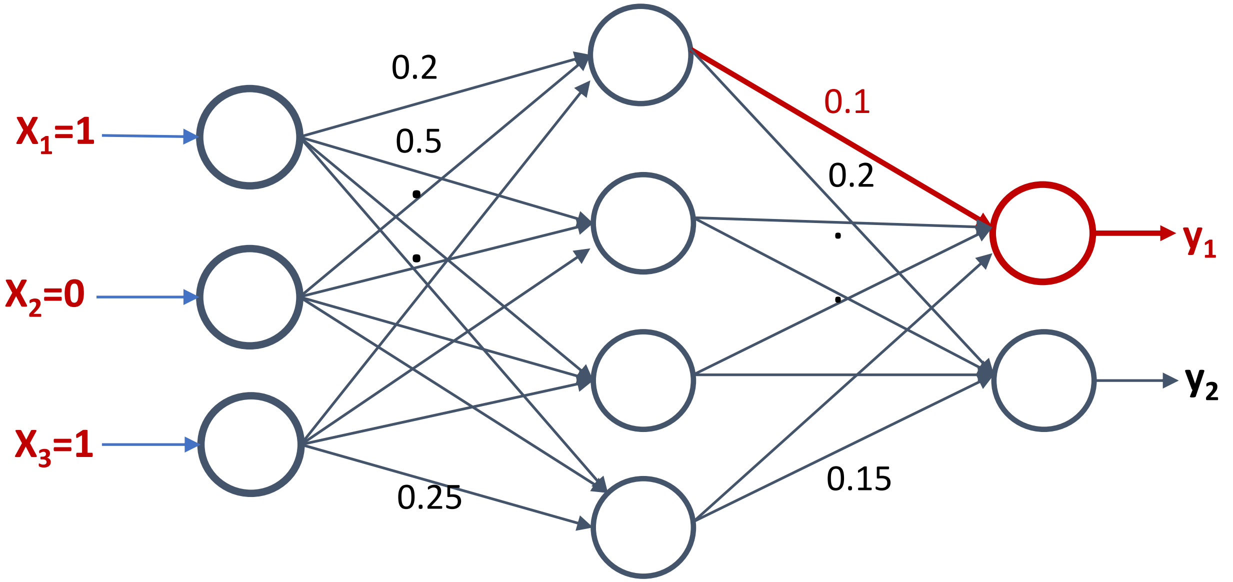


$$\nabla = \frac{\delta E}{\delta V} = 0$$

$$\nabla_{11} = \frac{\delta E}{\delta V_{11}} = 0$$

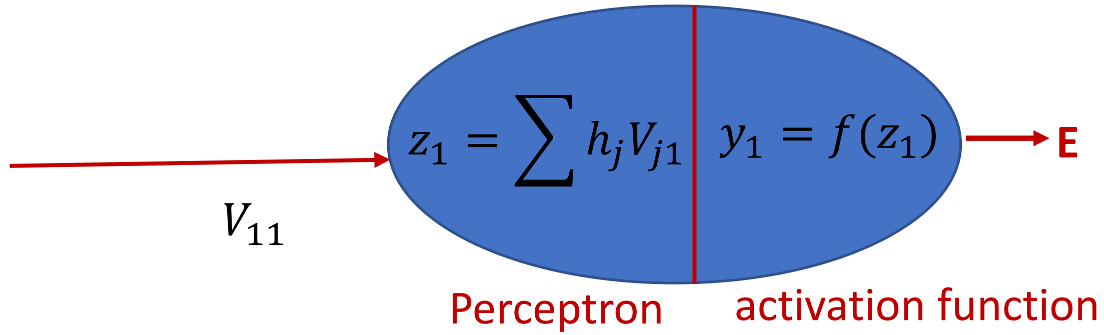
How are the Derivatives performed

Loss function $E = \|\bar{y} - \acute{y}\|$



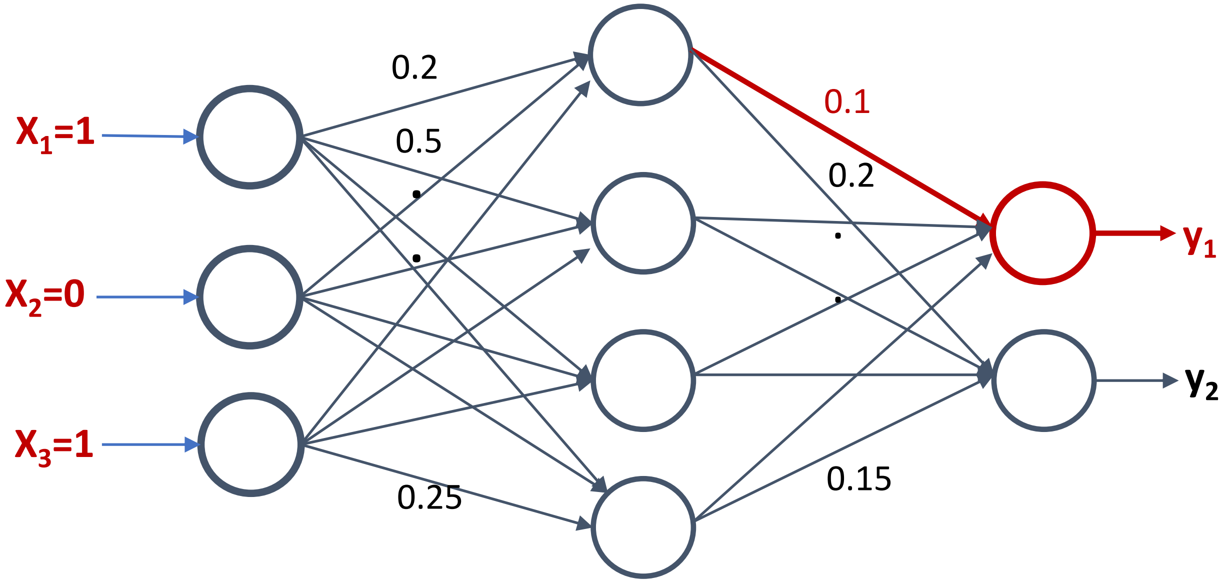
$$\nabla = \frac{\delta E}{\delta V} = 0$$

$$\nabla_{11} = \frac{\delta E}{\delta V_{11}} = 0$$



How are the Derivatives performed

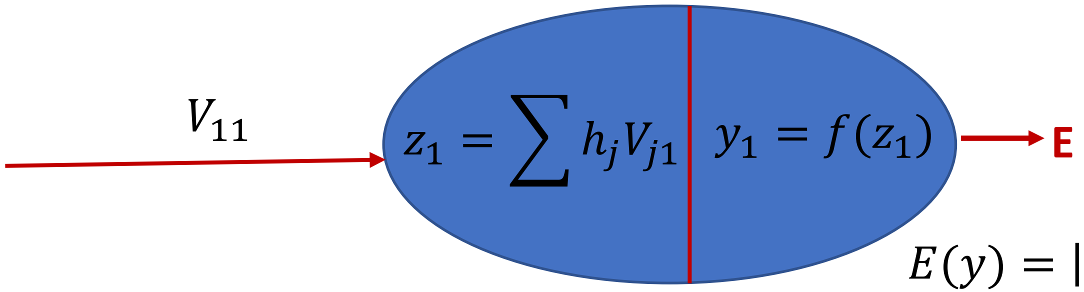
Loss function $E = \|\bar{y} - \acute{y}\|$



$$\nabla = \frac{\delta E}{\delta V} = 0$$

$$\nabla_{11} = \frac{\delta E}{\delta V_{11}} = 0$$

$$z_1(V_{11}) = h_1V_{11} + h_2V_{21} + h_3V_{31} + h_4V_{41}$$

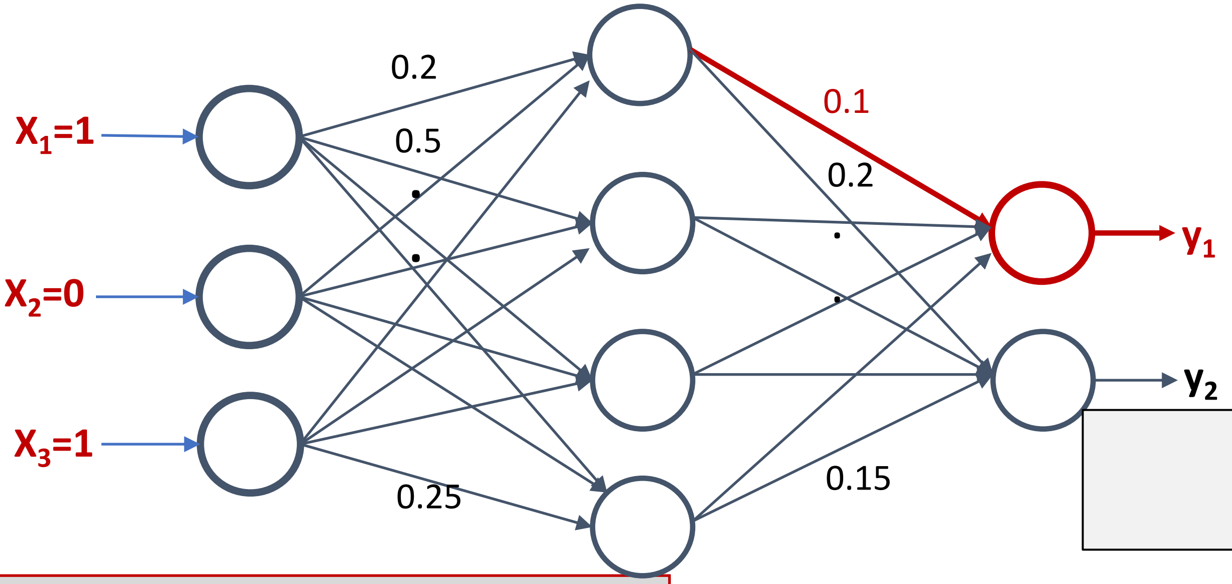


$$E(y) = \|\bar{y} - \acute{y}\|$$

$$y_1(z_1) = Sigmoid(z_1)$$

How are the Derivatives performed

Loss function $E = \|\bar{y} - \acute{y}\|$



$$\nabla = \frac{\delta E}{\delta V} = 0$$

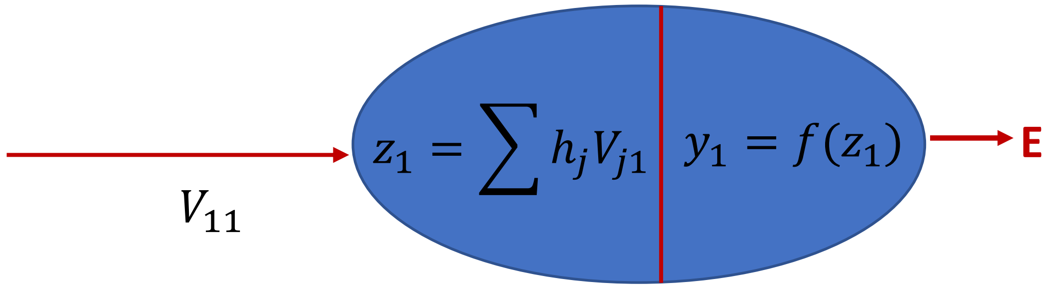
$$\nabla_{ij} = \frac{\delta E}{\delta V_{ij}} = 0$$

$$\frac{\delta E}{\delta V_{11}} = \frac{\delta z_1}{\delta V_{11}} \times \frac{\delta y_1}{\delta z_1} \times \frac{\delta E}{\delta y_1}$$

$$\nabla_{11} = \frac{\delta E}{\delta V_{11}} = \frac{\delta z_1}{\delta V_{11}} \cdot \frac{\delta y_1}{\delta z_1} \cdot \frac{\delta E}{\delta y_1}$$

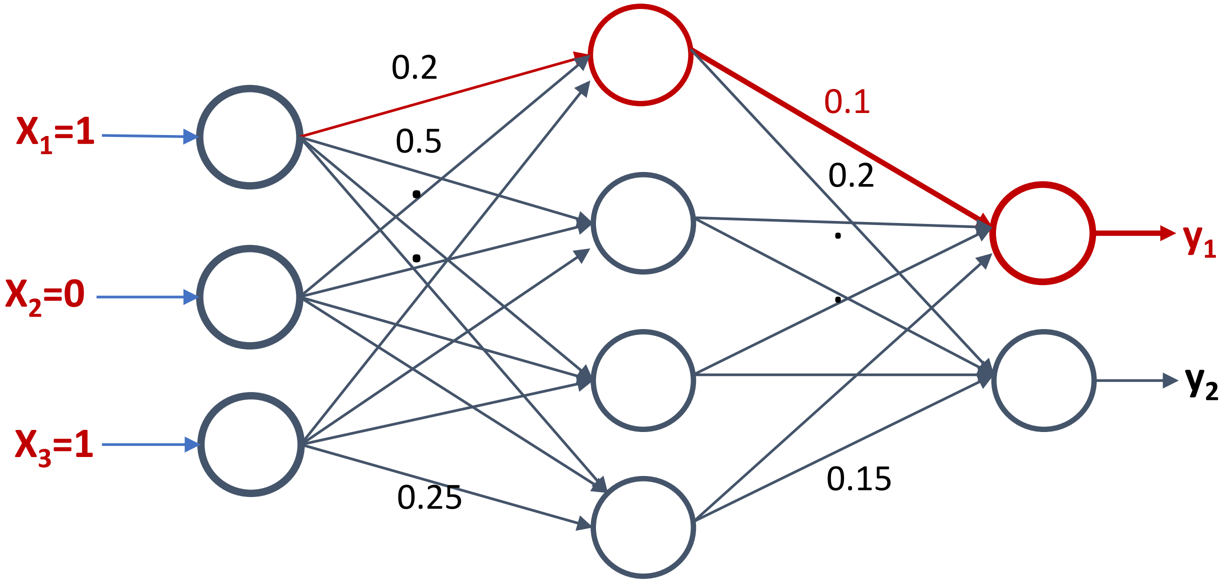
$$z_1 = h_1 V_{11} + h_2 V_{21} + h_3 V_{31} + h_4 V_{41}$$

$$y_1 = Sigmoid(\boldsymbol{z_1})$$



Backpropagation

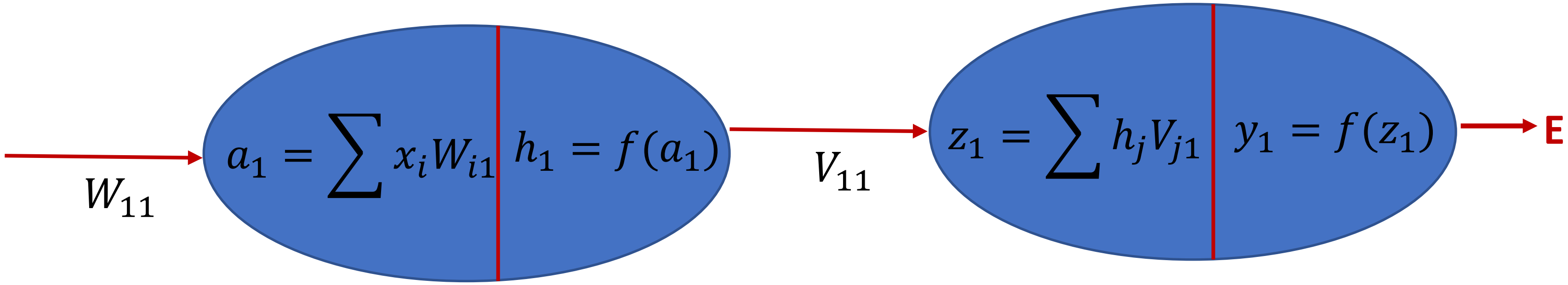
Loss function $E = \|\bar{y} - \acute{y}\|$



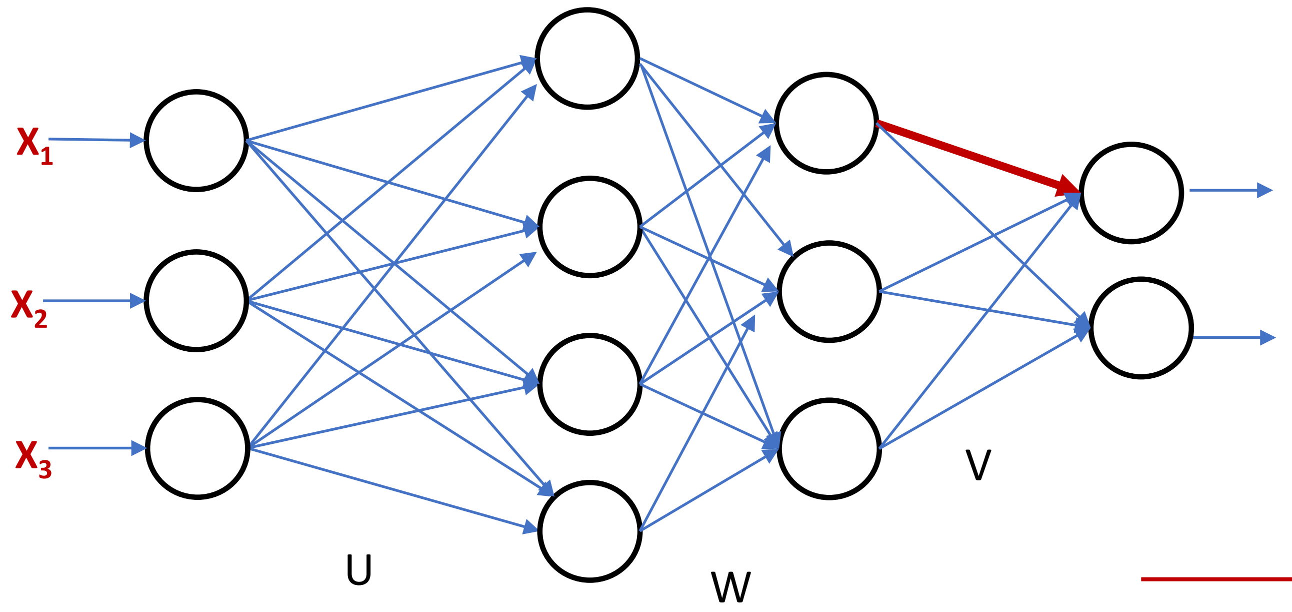
$$\nabla = \frac{\delta E}{\delta W} = 0$$

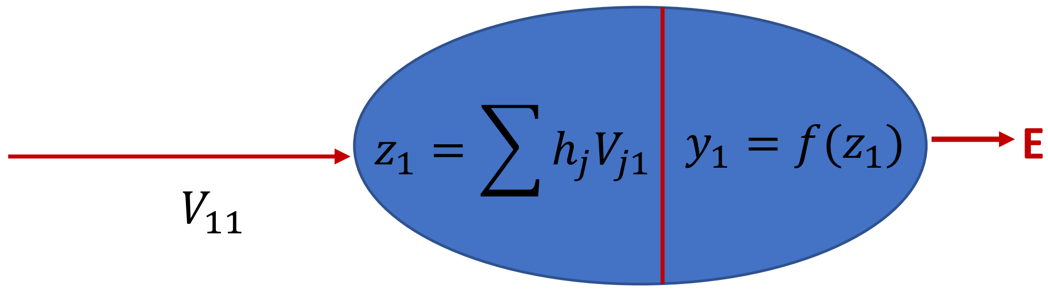
$$\nabla_{ij} = \frac{\delta E}{\delta W_{ij}} = 0$$

$$\frac{\delta E}{\delta W_{11}} = \frac{\delta a_1}{\delta W_{11}} \times \frac{\delta h_1}{\delta a_1} \times \qquad \frac{\delta z_1}{\delta h_1} \times \frac{\delta y_1}{\delta z_1} \times \frac{\delta E}{\delta y_1}$$



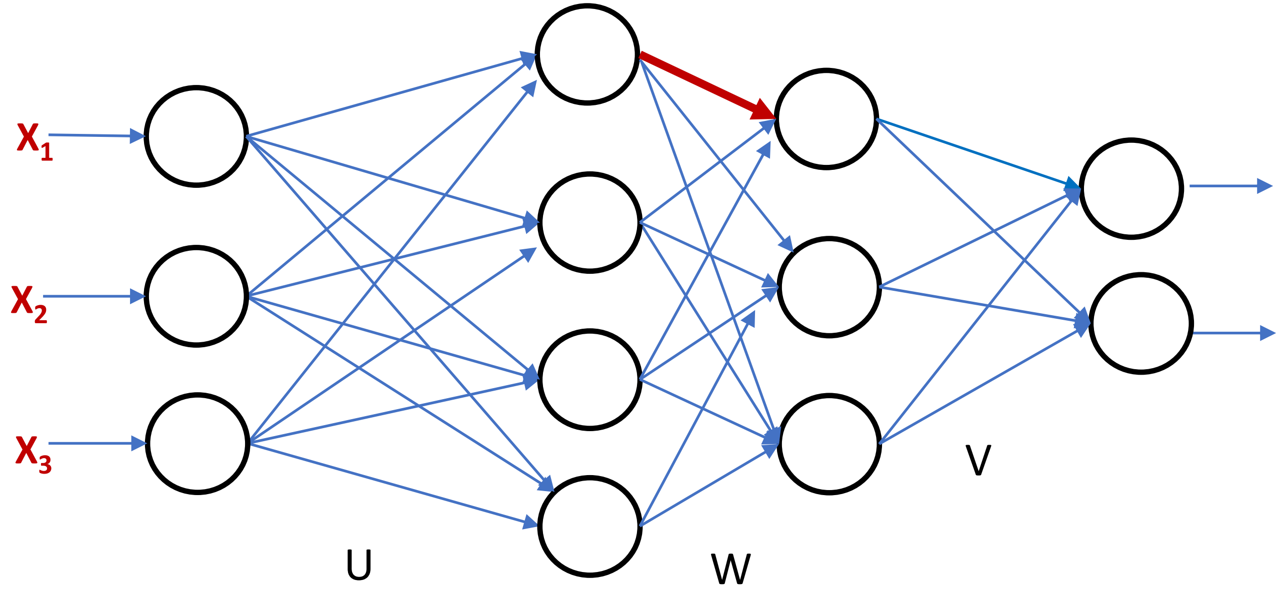
We may have multiple layers.



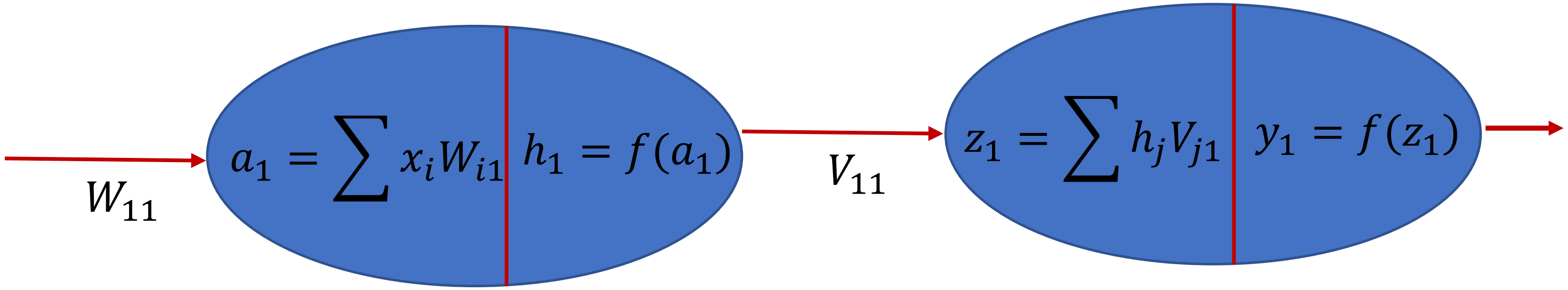


$$\frac{\delta E}{\delta V_{11}} = \frac{\delta z_1}{\delta V_{11}} \times \frac{\delta y_1}{\delta z_1} \times \frac{\delta E}{\delta y_1}$$

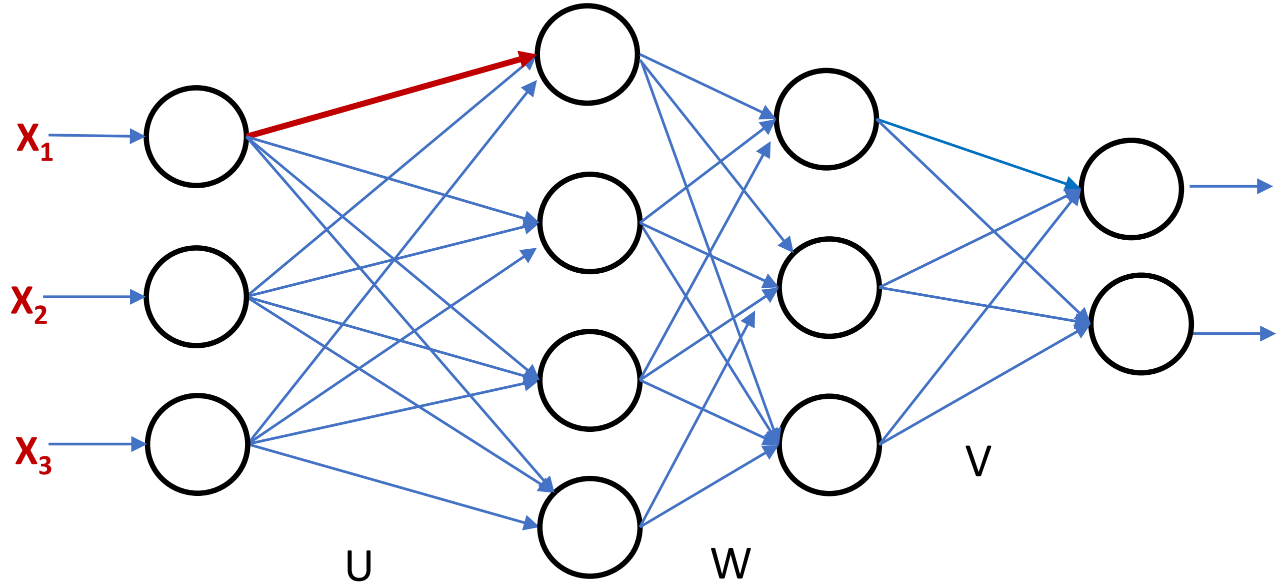
We may have multiple layers



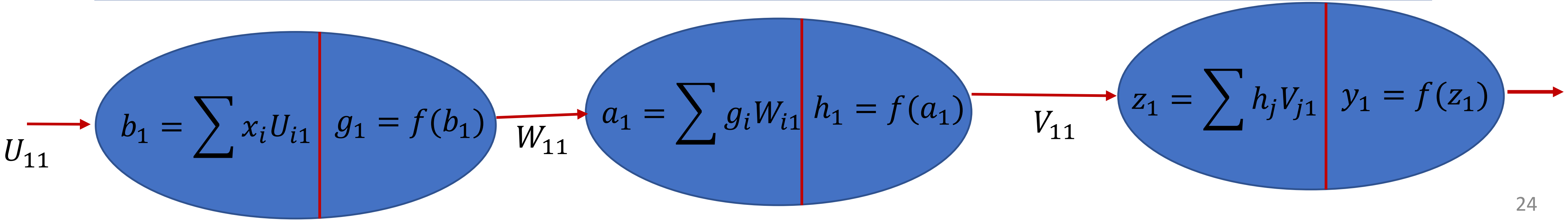
$$\frac{\delta E}{\delta W_{11}} = \frac{\delta a_1}{\delta W_{11}} \times \frac{\delta h_1}{\delta a_1} \times \qquad \frac{\delta z_1}{\delta h_1} \times \frac{\delta y_1}{\delta z_1} \times \frac{\delta E}{\delta y_1}$$



We may have multiple layers



$$\frac{\delta E}{\delta U_{11}} = \frac{\delta b_1}{\delta U_{11}} \times \frac{\delta g_1}{\delta b_1} \times \frac{\delta a_1}{\delta g_1} \times \frac{\delta h_1}{\delta a_1} \times \frac{\delta z_1}{\delta h_1} \times \frac{\delta y_1}{\delta z_1} \times \frac{\delta E}{\delta y_1}$$



Summary

- What is loss function?
- What are the parameters of a multilayer perceptron neural network?
- How to estimate parameters using backpropagation through time?

Lesson 11

Learning with different Loss Functions and Their Derivatives

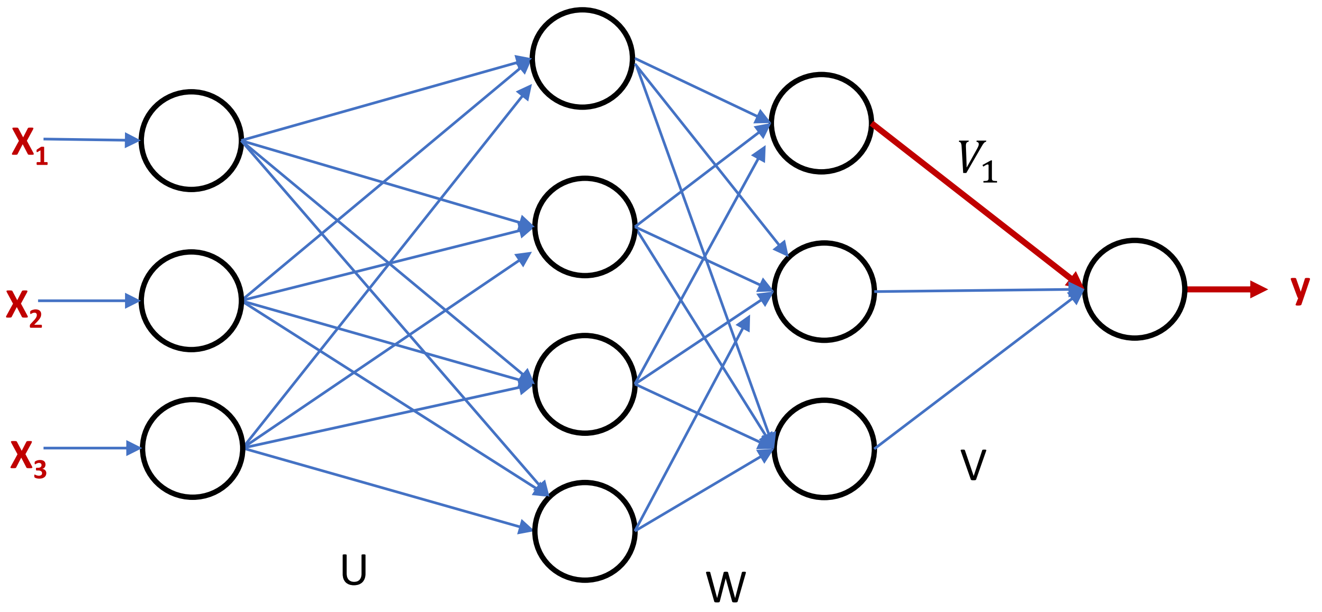
Two Commonly used Loss Functions are

- Mean Square Error – Standard Loss Function for Regression
- Cross Entropy Loss - Standard Loss Function for Classification

Mean Square Error (MSE)

For the Single Sample

MSE $E = (y - \acute{y})^2$

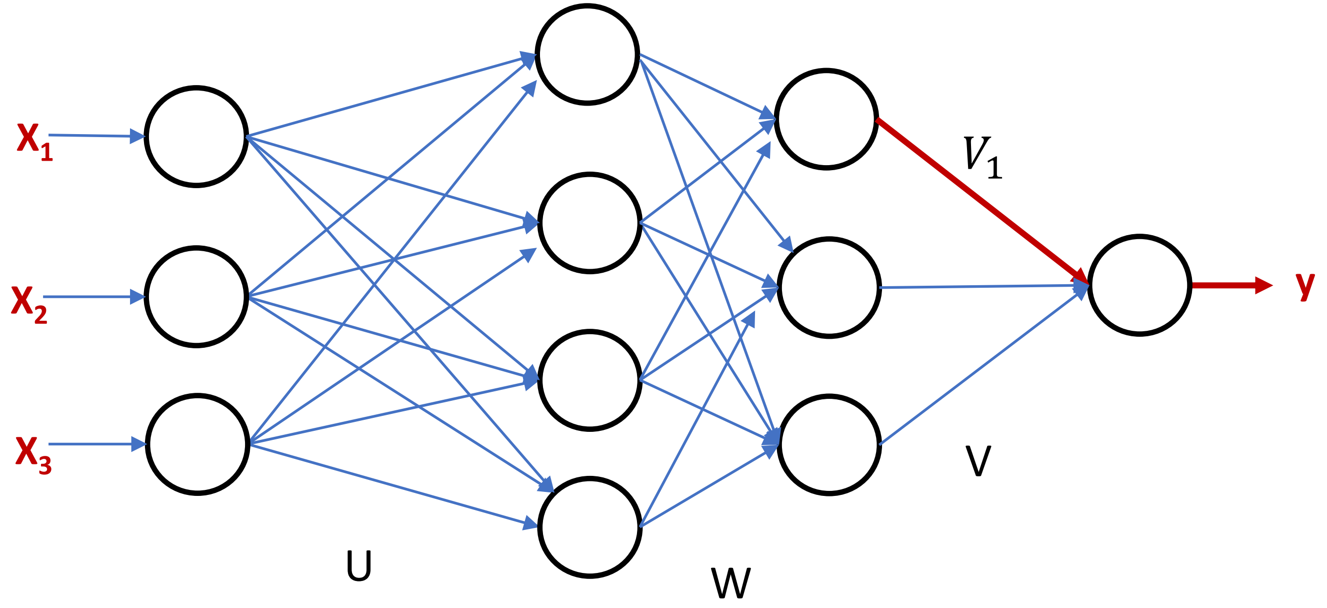


Ground truth is $\acute{y}$

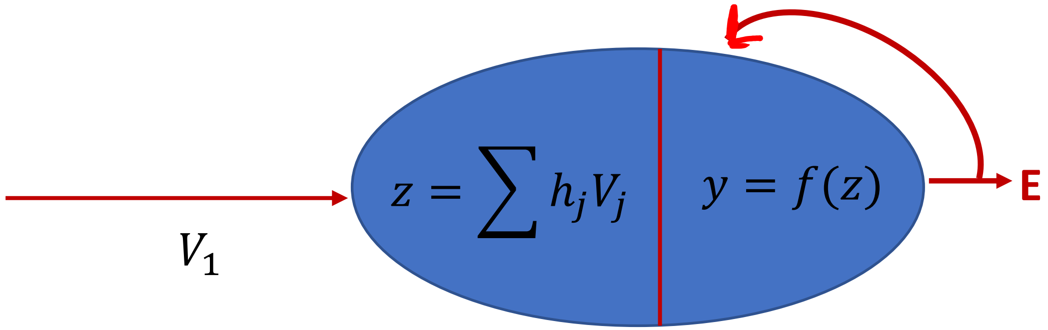
Mean Square Error (MSE)

For the Single Sample

MSE $E = (y - \acute{y})^2$

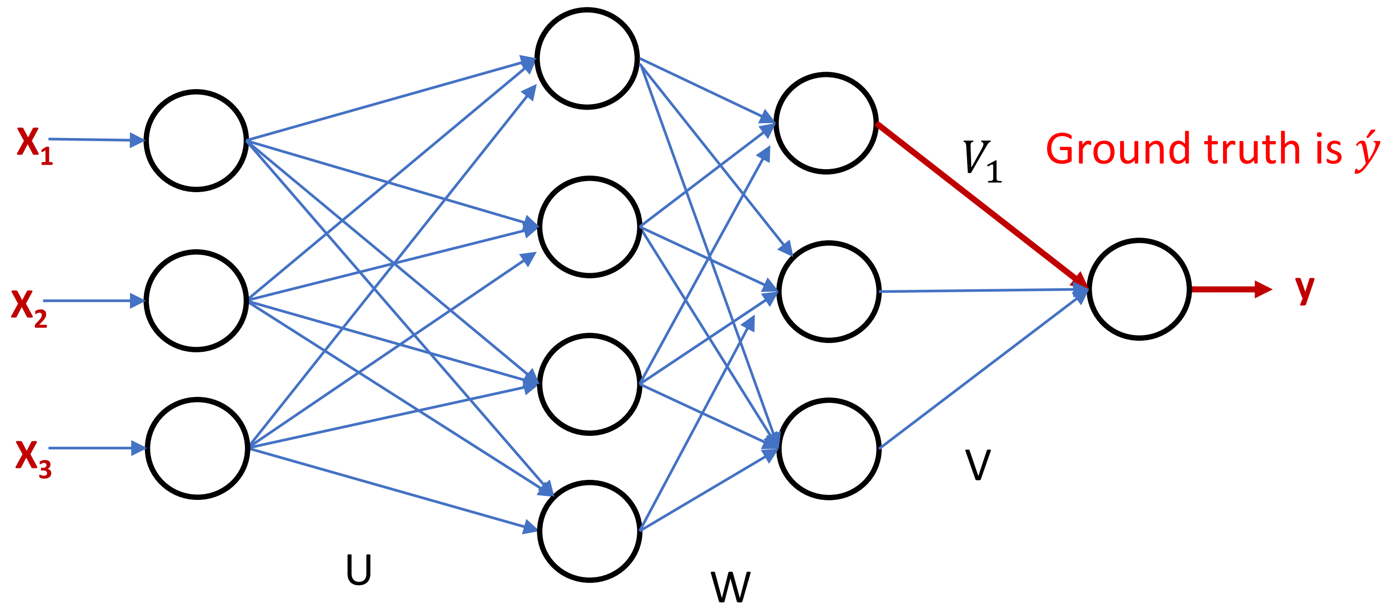


Ground truth is
$\acute{y}$



$$\frac{\delta E}{\delta V_1} = \frac{\delta z}{\delta V_1} \times \frac{\delta y}{\delta z} \times \boxed{\frac{\delta E}{\delta y}}$$

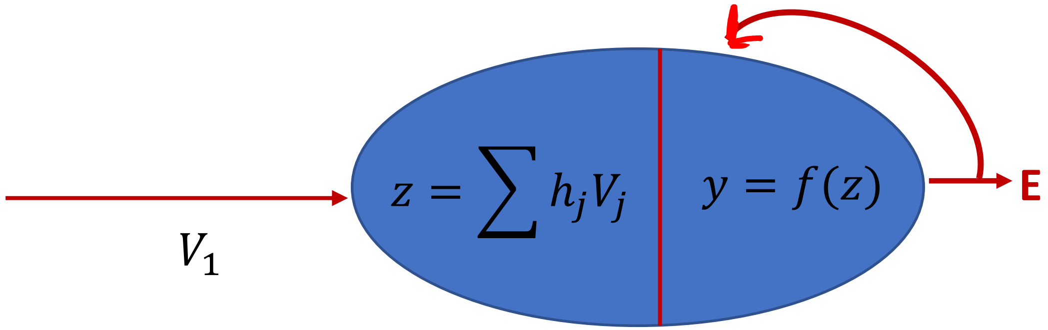
Mean Square Errors



For the Single Sample

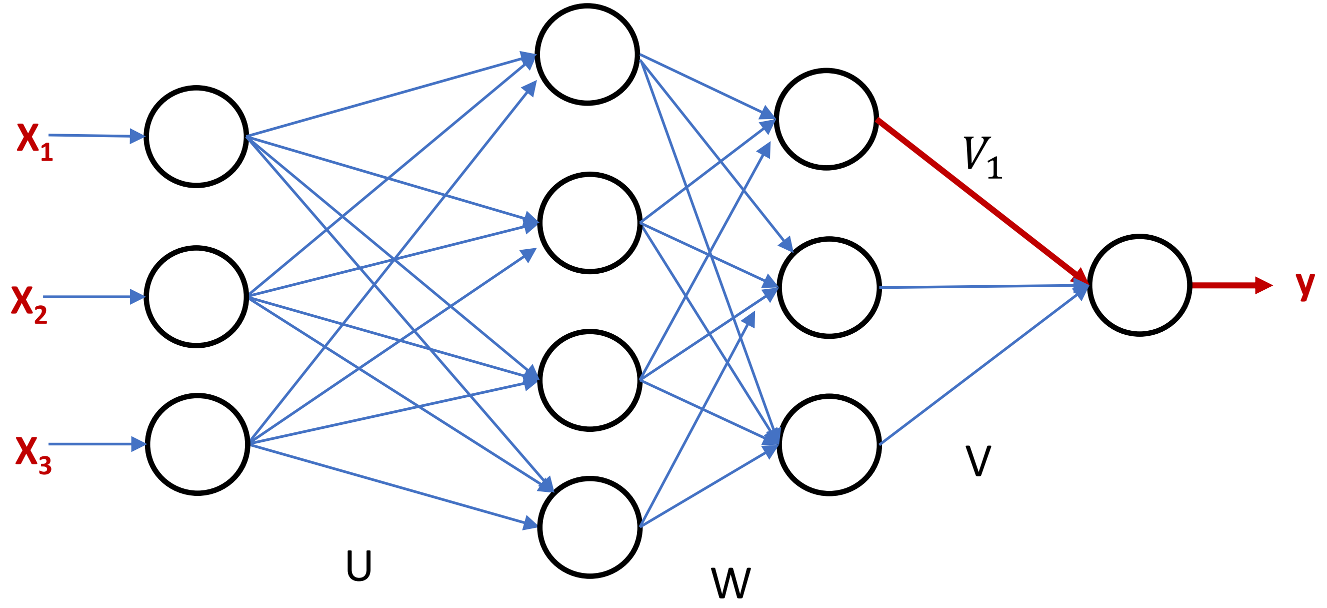
$$E = (y - \acute{y})^2$$

$$\frac{\delta E}{\delta y} = \frac{\delta (y - \acute{y})^2}{\delta y} = 2(y - \acute{y})$$



$$\frac{\delta E}{\delta V_1} = \frac{\delta z}{\delta V_1} \times \frac{\delta y}{\delta z} \times \boxed{\frac{\delta E}{\delta y}}$$

Mean Square Errors

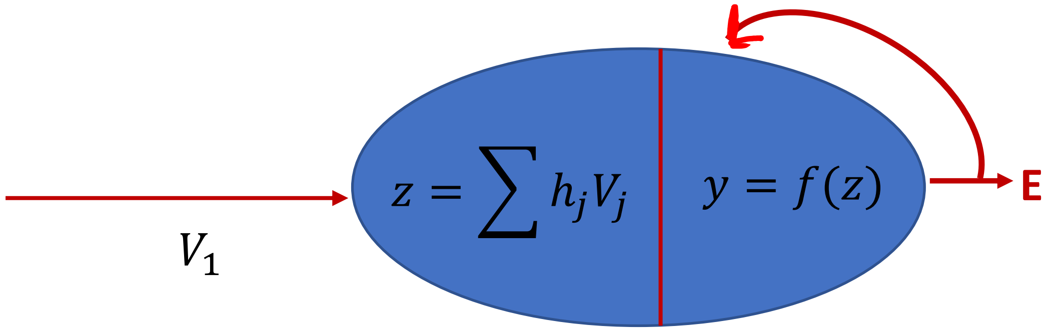


If the ground truth is $\acute{y}$

For n Samples

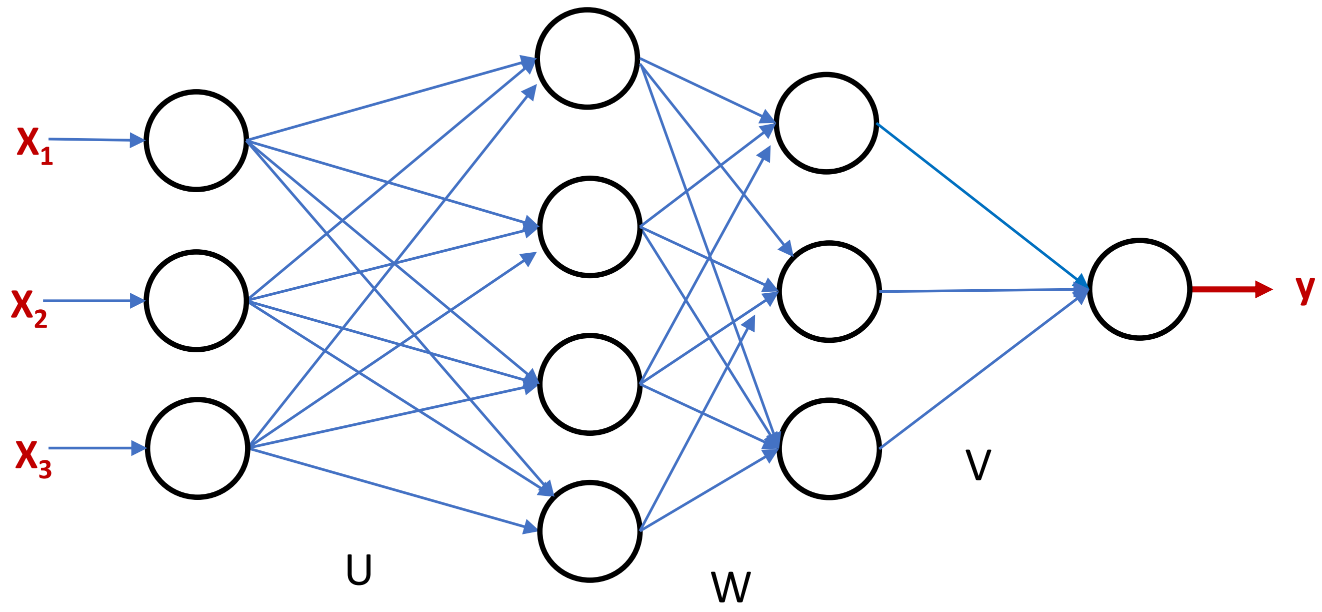
$$E = \frac{1}{n}\sum_{i=1}^{n}(y^i - \acute{y}^i)^2$$

$$\frac{\delta E}{\delta y} = \frac{\delta \frac{1}{n}\sum_{i=1}^{n}(y^i - \acute{y}^i)^2}{\delta y} = \frac{2}{n}\sum_{i=1}^{n}(y^i - \acute{y}^i)$$



$$\frac{\delta E}{\delta V_1} = \frac{\delta z}{\delta V_1} \times \frac{\delta y}{\delta z} \times \boxed{\frac{\delta E}{\delta y}}$$

Mean Square Errors



If the ground truth is $\acute{y}$

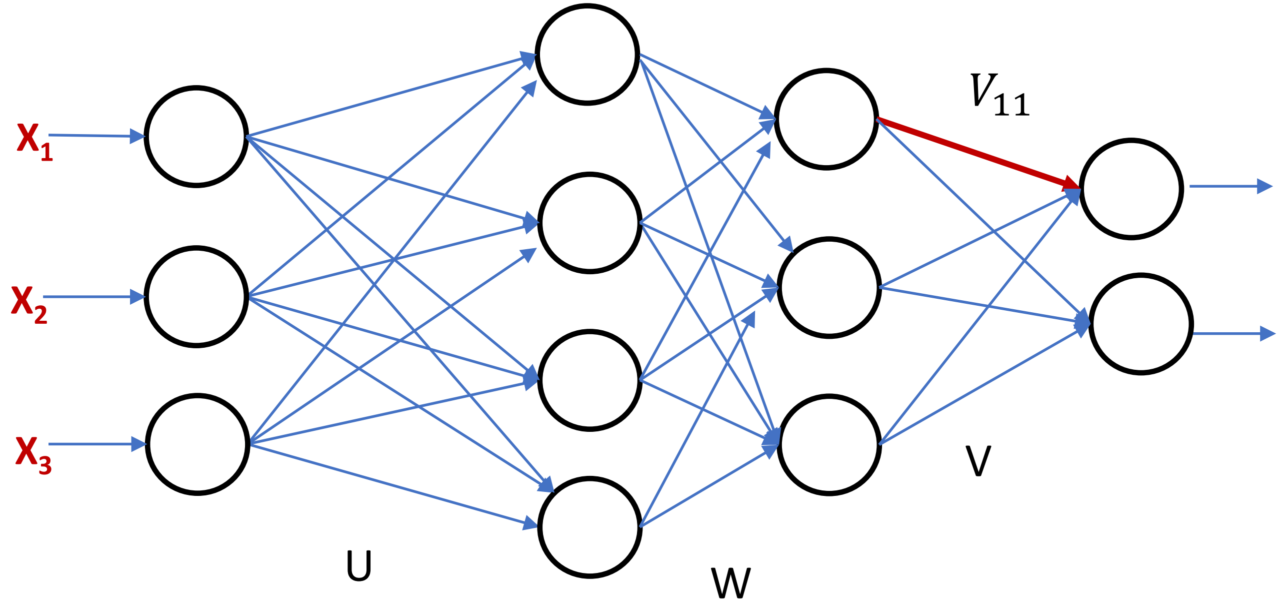
For *n* Samples

$$E = \frac{1}{n}\sum_{i=1}^{n}(y^i - \acute{y}^i)^2$$

$$\frac{\delta E}{\delta y} = \frac{\delta \frac{1}{n}\sum_{i=1}^{n}(y^i - \acute{y}^i)^2}{\delta y} = \frac{2}{n}\sum_{i=1}^{n}(y^i - \acute{y}^i)$$

Backpropagation will be done after a batch of *n* Samples

Mean Square Errors

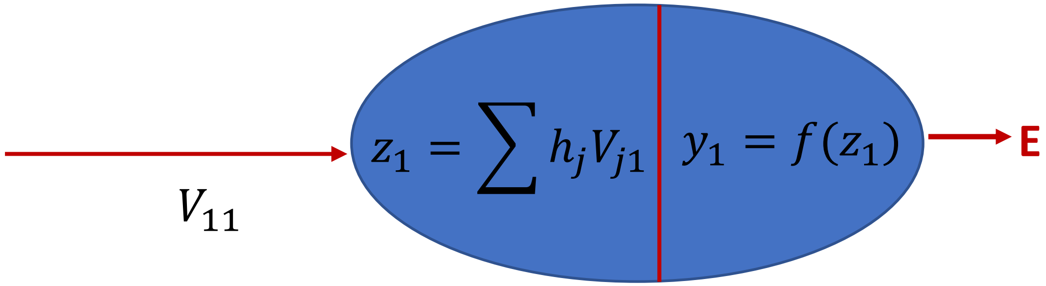


If the ground truth is $\acute{y}$

For the n Sample

$$\boldsymbol{E} = \frac{1}{n}\sum_{i=1}^{n}(\boldsymbol{y}^i - \boldsymbol{\acute{y}}^i)^2$$

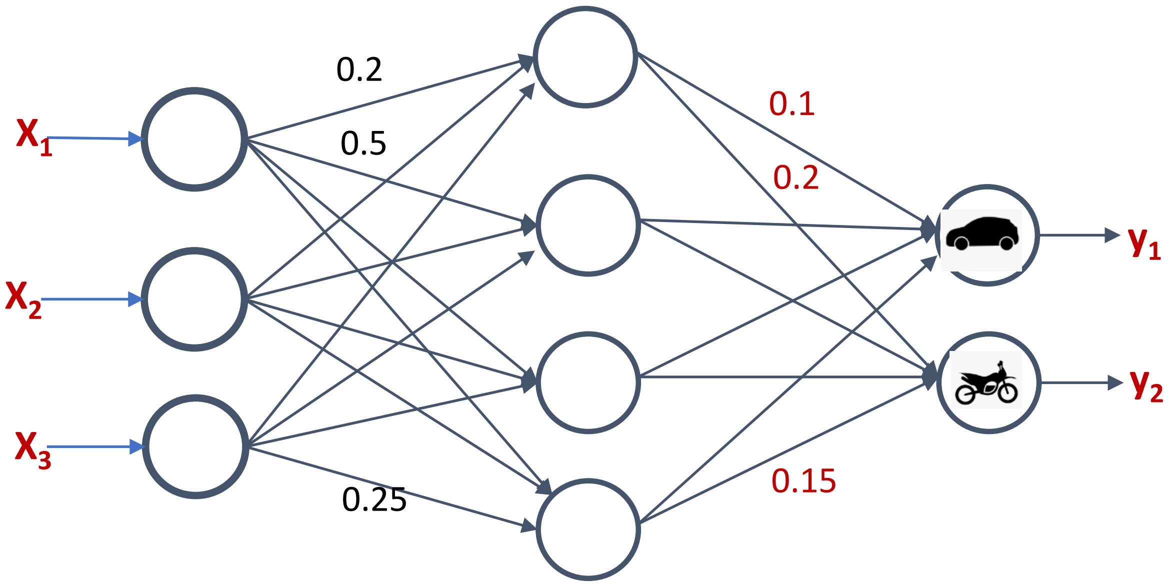
$$\frac{\delta \boldsymbol{E}}{\delta y_1} = \frac{\delta \frac{1}{n}\sum_{i=1}^{n}(\boldsymbol{y}^i - \boldsymbol{\acute{y}}^i)^2}{\delta y_1} = \frac{2}{n}\sum_{i=1}^{n}(y_1{}^i - \acute{y}_1{}^i)$$



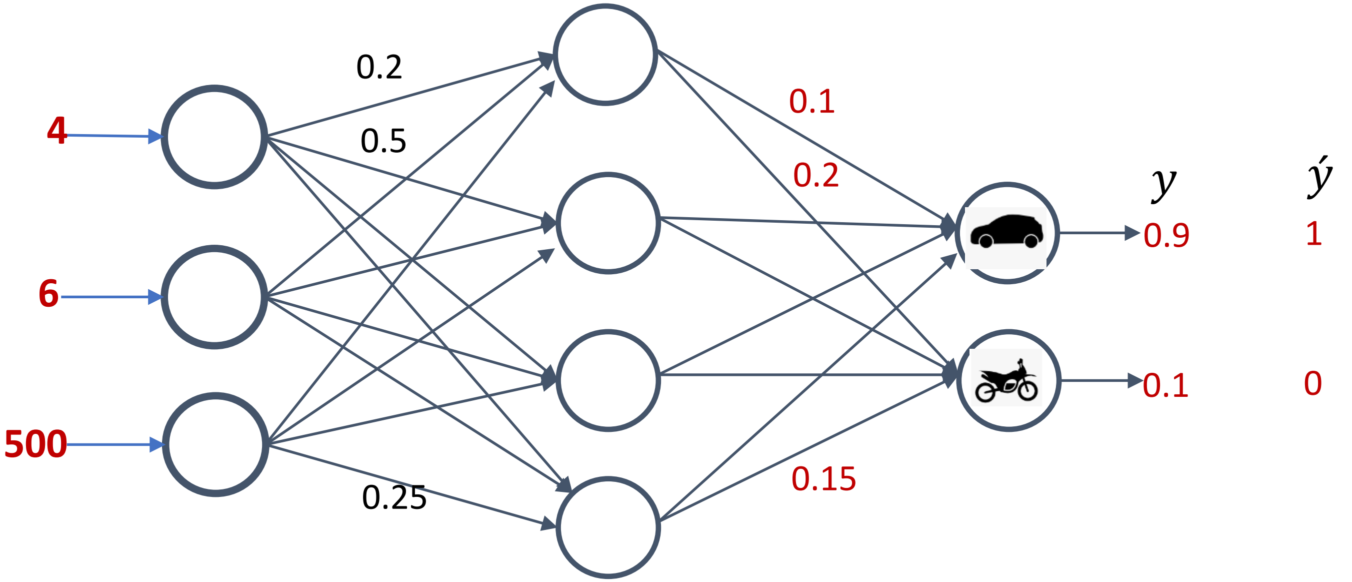
$$\frac{\delta E}{\delta V_{11}} = \frac{\delta z_1}{\delta V_{11}} \times \frac{\delta y_1}{\delta z_1} \times \frac{\delta \boldsymbol{E}}{\delta y_1}$$

Let us illustrate with a toy example

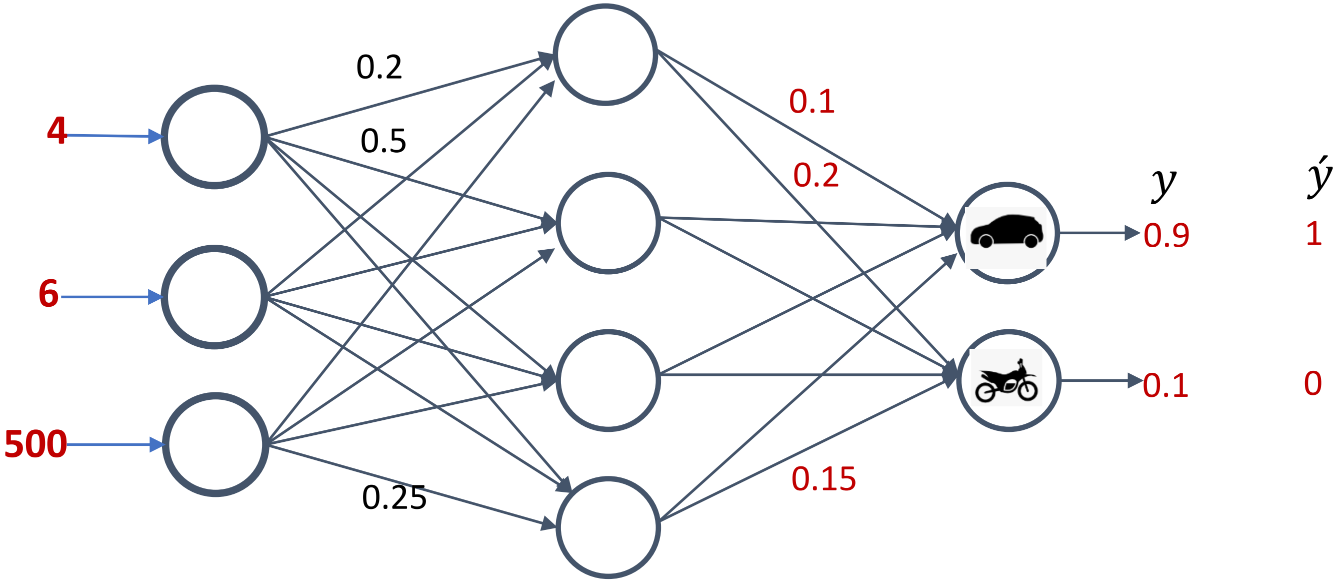
#Wheel	Height	Weight	
4	6	500	
4	5.5	600	
4	5	550	
2	3	200	
2	3.5	150	
2	4	250	



#Wheel	Height	Weight	
4	6	500	
4	5.5	600	
4	5	550	
2	3	200	
2	3.5	150	
2	4	250	



#Wheel	Height	Weight	
4	6	500	
4	5.5	600	
4	5	550	
2	3	200	
2	3.5	150	
2	4	250	



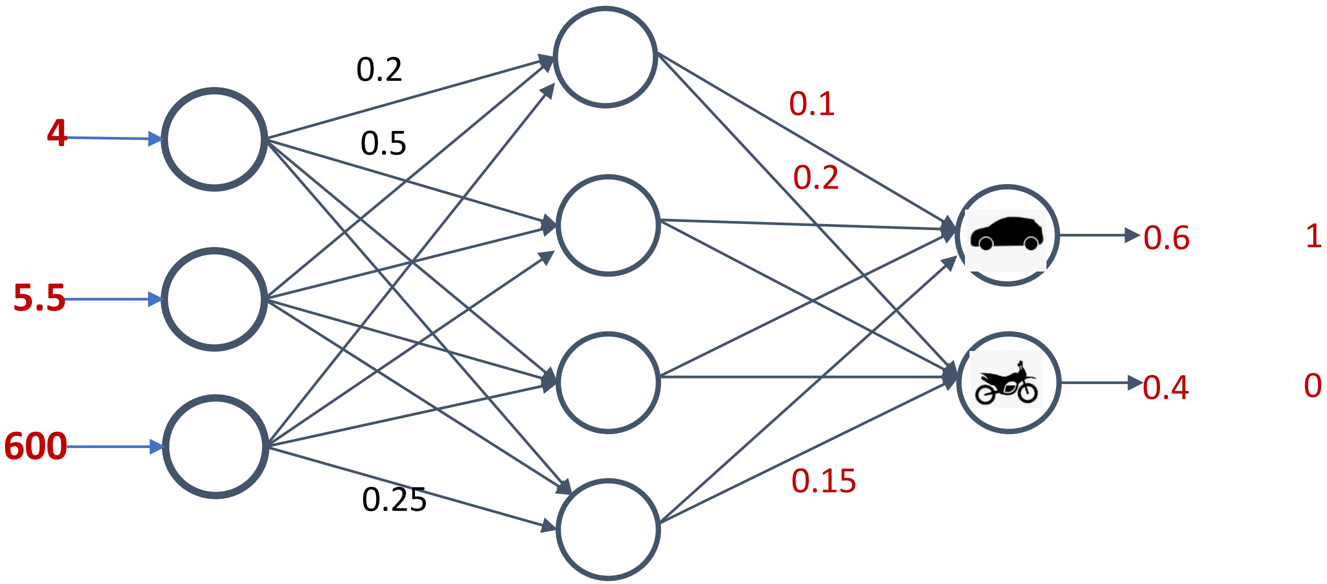
Error $E = (y - \acute{y})^2$

Error $E_{y_1} = (0.9 - 1)^2 = 0.01$

Error $E_{y_2} = (0.1 - 0)^2 = 0.01$

If these errors are not acceptable, then Backpropagate.

#Wheel	Height	Weight	
4	6	500	
4	5.5	600	
4	5	550	
2	3	200	
2	3.5	150	
2	4	250	



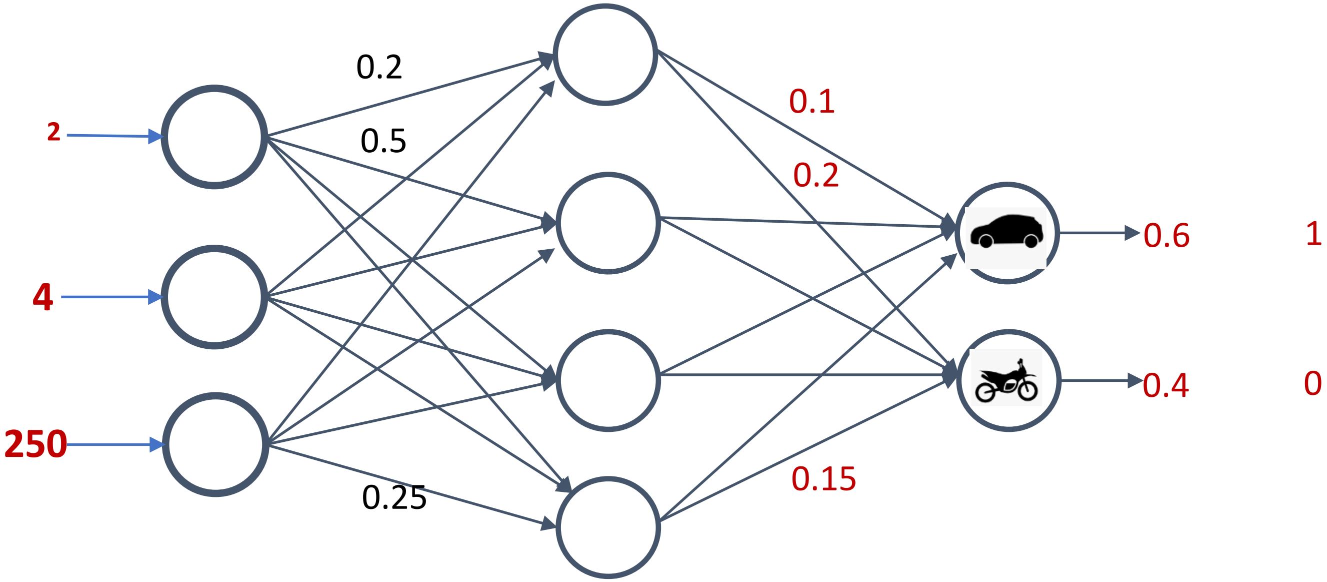
Error $E = (y - \acute{y})^2$

Error $E_{y_1} = (0.6 - 1)^2 = 0.16$

Error $E_{y_2} = (0.4 - 0)^2 = 0.36$

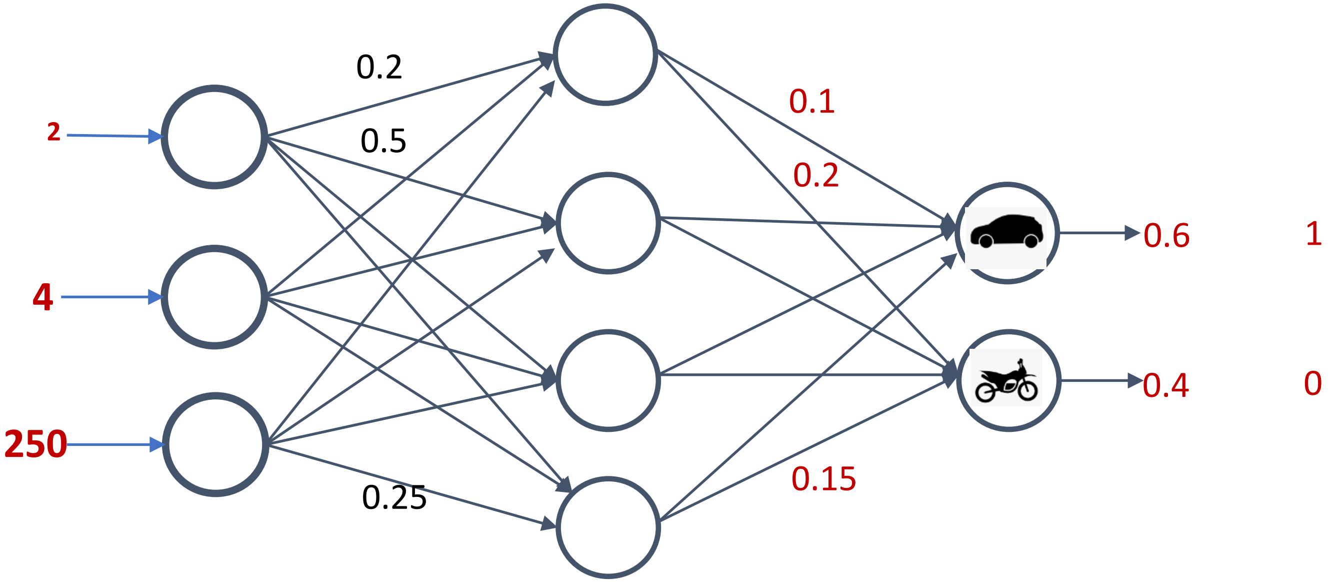
If these errors are not acceptable, then Backpropagate.

#Wheel	Height	Weight	
4	6	500	
4	5.5	600	
4	5	550	
2	3	200	
2	3.5	150	
$\boxed{2 \quad 4 \quad 250}$			



One complete cycle of training is called Epoch

#Wheel	Height	Weight	
4	6	500	
4	5.5	600	
4	5	550	
2	3	200	
2	3.5	150	
2	4	250	



Backpropagation after every sample is expensive.

Do it in batches.

Summary

- Mean Square Loss Function and how to estimate its gradient

Lesson 12

Learning with different Loss Functions and Their Derivatives

Two Commonly used Loss Functions are

- Mean Square Error – Standard Loss Function for Regression
- Cross Entropy Loss - Standard Loss Function for Classification

Cross Entropy Loss

Cross-entropy is a measure of the difference between two probability distributions for a given random variable or set of events. If p and q are two probability distributions drown from a random variable X, cross entropy is defined as

$$CE = -\sum_{x \epsilon X}^{n} p(x) \log q(x)$$

Cross Entropy Loss

Cross-entropy is a measure of the difference between two probability distributions for a given random variable or set of events. If p and q are two probability distributions drown from a random variable X, the distance of p from q i.e., cross entropy is defined as

$$CE = -\sum_{x \epsilon X}^{n} q(x) \log p(x)$$

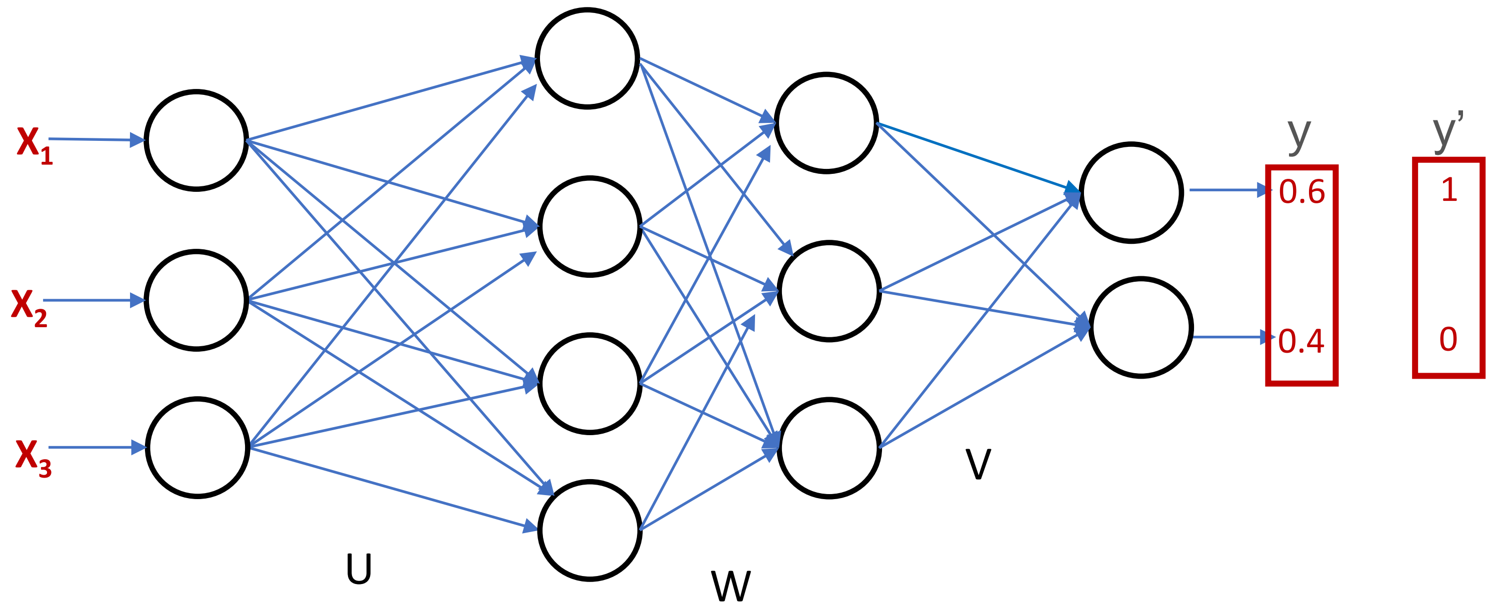
X = {H, T}, p = {0.9, 0.1} and q = {0.6,0.4}

How different p from q?

CE = - 0.6 log(0.9) - 0.4 log(0.1) = 1.42

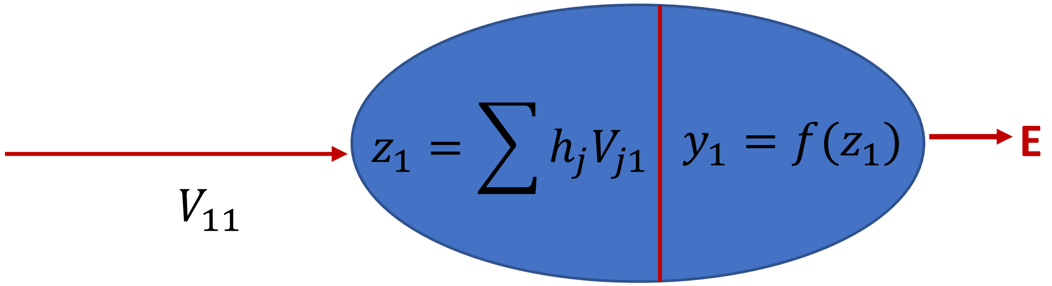
Cross Entropy Loss

$$CE = -\sum_{x \epsilon X}^{n} q(x) \log p(x)$$



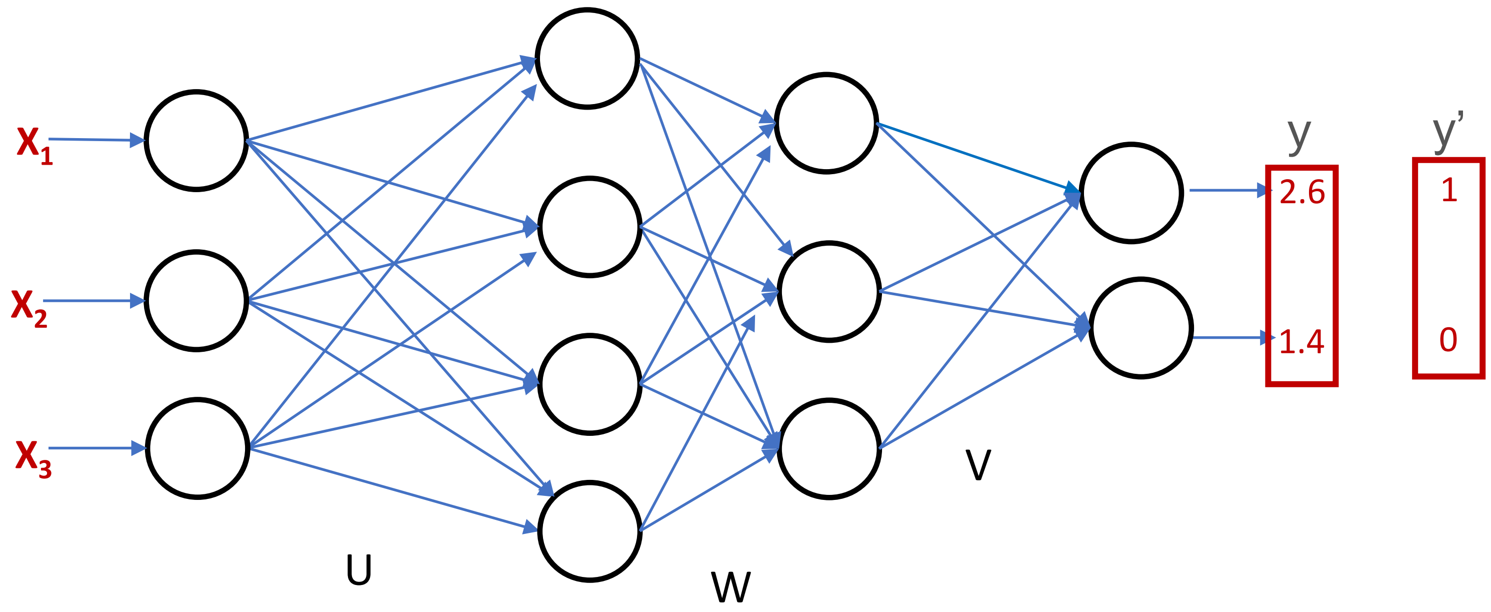
$$E = -\sum_{i}^{C} \acute{y}_i \log(y_i)$$

$$\frac{\delta E}{\delta y_i} = -\sum_{i}^{C} \frac{\acute{y}_i}{y_i}$$

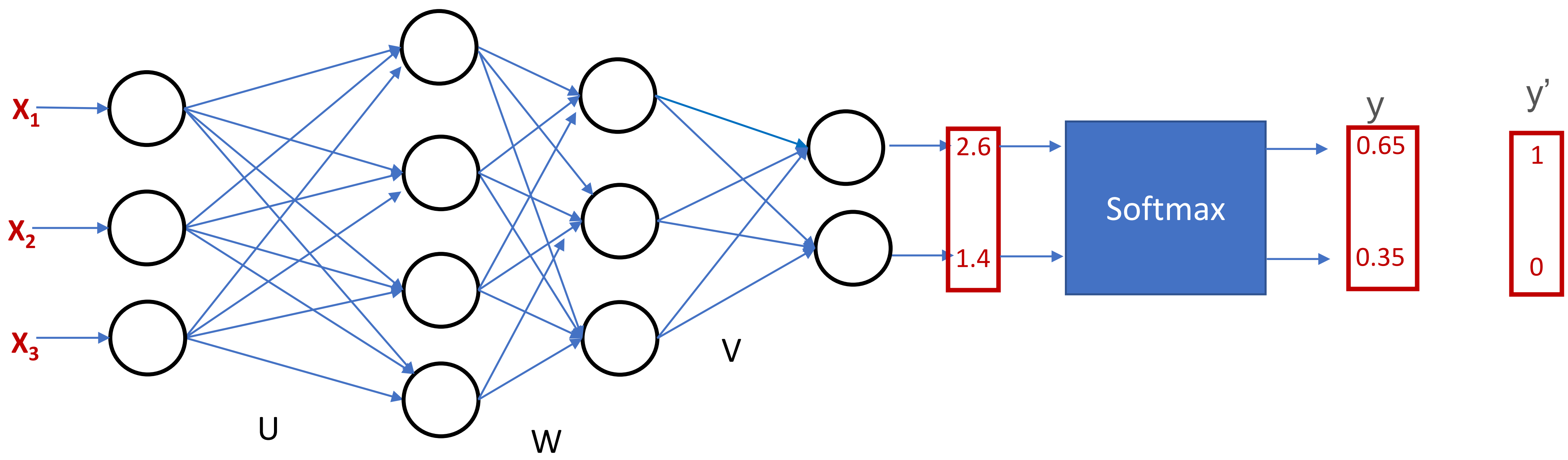


$$\boxed{\frac{\delta E}{\delta V_{11}} = \frac{\delta z_1}{\delta V_{11}} \times \frac{\delta y_1}{\delta z_1} \times \frac{\delta E}{\delta y_1}}$$

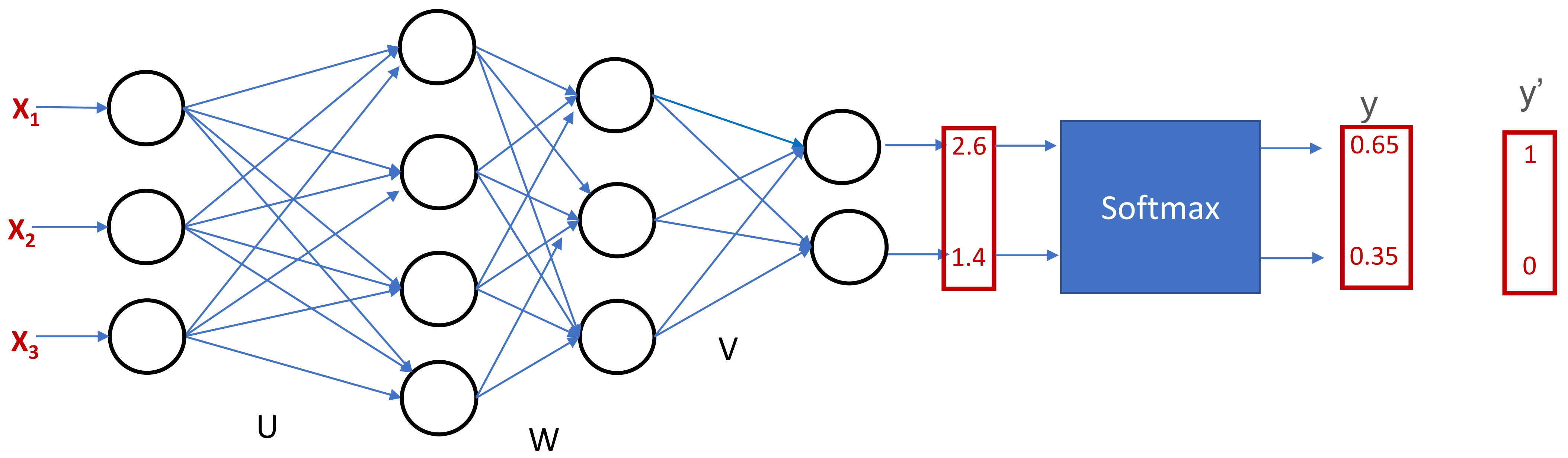
Cross Entropy Loss



Cross Entropy Loss



Backpropagation



Softmax

$\text{softmax}: \mathbb{R}^n \to \mathbb{R}^n$

z$= \{z_1, z_2, z_3, \dots., z_n\}$

Z = {1.1, 2.2, 0.2, -1.7}

$$\text{Softmax}(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{n} e^{z_j}}$$

Z = {0.224, 0.672, 0.091, 0.013}

$$\text{Softmax}(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{n} e^{z_j}}$$

Z = {1.1, 2.2, 0.2, -1.7}

$$\left\{ \frac{e^{z_1}}{\sum_{j=1}^{n} e^{z_j}}, \quad \frac{e^{z_2}}{\sum_{j=1}^{n} e^{z_j}}, \quad \frac{e^{z_3}}{\sum_{j=1}^{n} e^{z_j}} \cdots, \quad \frac{e^{z_n}}{\sum_{j=1}^{n} e^{z_j}} \right\}$$

$$\text{Softmax}(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{n} e^{z_j}}$$

Z = {1.1, 2.2, 0.2, -1.7}

$$\left\{ \frac{e^{z_1}}{\sum_{j=1}^{n} e^{z_j}}, \quad \frac{e^{z_2}}{\sum_{j=1}^{n} e^{z_j}}, \quad \frac{e^{z_3}}{\sum_{j=1}^{n} e^{z_j}} \dots, \quad \frac{e^{z_n}}{\sum_{j=1}^{n} e^{z_j}} \right\}$$

If i = k

$$\frac{\delta \frac{e^{z_i}}{\sum_{j=1}^{n} e^{z_j}}}{\delta z_k} = \frac{e^{z_i} \sum_{j=1}^{n} e^{z_j} - e^{z_k} e^{z_i}}{(\sum_{j=1}^{n} e^{z_j})^2} = \frac{e^{z_i} (\sum_{j=1}^{n} e^{z_j} - e^{z_k})}{(\sum_{j=1}^{n} e^{z_j})^2} = \frac{e^{z_i}}{\sum_{j=1}^{n} e^{z_j}} \times \frac{\sum_{j=1}^{n} e^{z_j} - e^{z_k}}{\sum_{j=1}^{n} e^{z_j}} = p_i(1 - p_i)$$

$$\text{Softmax}(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{n} e^{z_j}}$$

Z = {1.1, 2.2, 0.2, -1.7}

$$\{ \quad \frac{e^{z_1}}{\sum_{j=1}^{n} e^{z_j}}, \quad \frac{e^{z_2}}{\sum_{j=1}^{n} e^{z_j}}, \quad \frac{e^{z_3}}{\sum_{j=1}^{n} e^{z_j}} \ldots, \quad \frac{e^{z_n}}{\sum_{j=1}^{n} e^{z_j}} \}$$

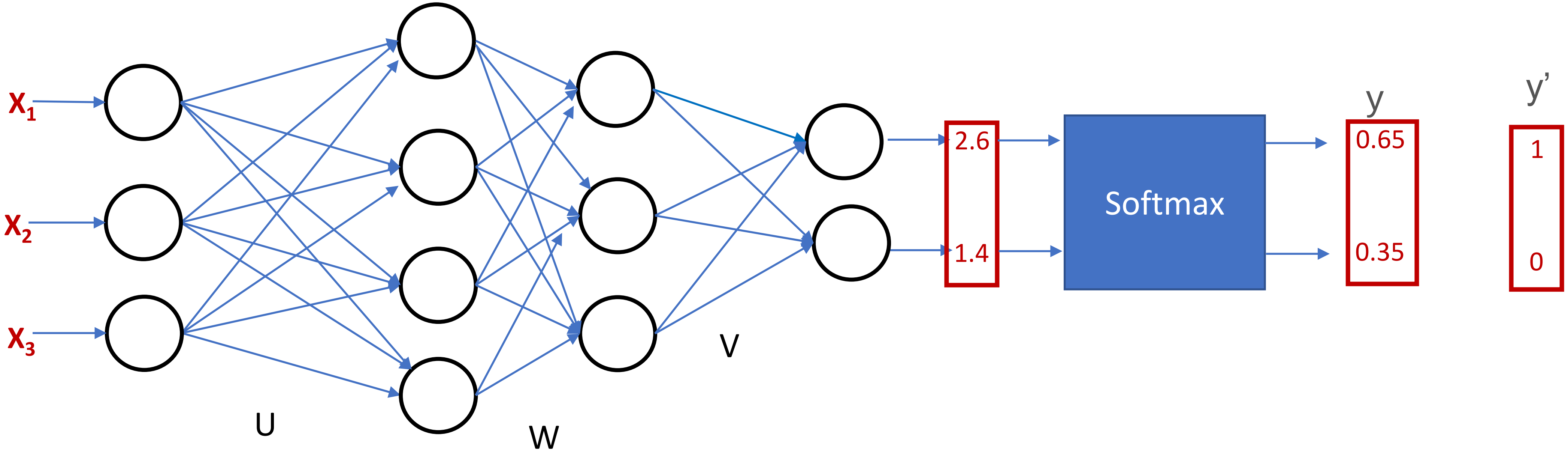
If i = k

$$\frac{\delta \frac{e^{z_i}}{\sum_{j=1}^{n} e^{z_j}}}{\delta z_k} = \frac{e^{z_i} \sum_{j=1}^{n} e^{z_j} - e^{z_k} e^{z_i}}{(\sum_{j=1}^{n} e^{z_j})^2} = \frac{e^{z_i} (\sum_{j=1}^{n} e^{z_j} - e^{z_k})}{(\sum_{j=1}^{n} e^{z_j})^2} = \frac{e^{z_i}}{\sum_{j=1}^{n} e^{z_j}} \times \frac{\sum_{j=1}^{n} e^{z_j} - e^{z_k}}{\sum_{j=1}^{n} e^{z_j}} = p_i(1 - p_i)$$

If i != k

$$\frac{\delta \frac{e^{z_i}}{\sum_{j=1}^{n} e^{z_j}}}{\delta z_k} = \frac{0 - e^{z_k} e^{z_i}}{(\sum_{j=1}^{n} e^{z_j})^2} = \frac{-e^{z_k}}{\sum_{j=1}^{n} e^{z_j}} \times \frac{e^{z_i}}{\sum_{j=1}^{n} e^{z_j}} = -p_i p_k$$

Backpropagation



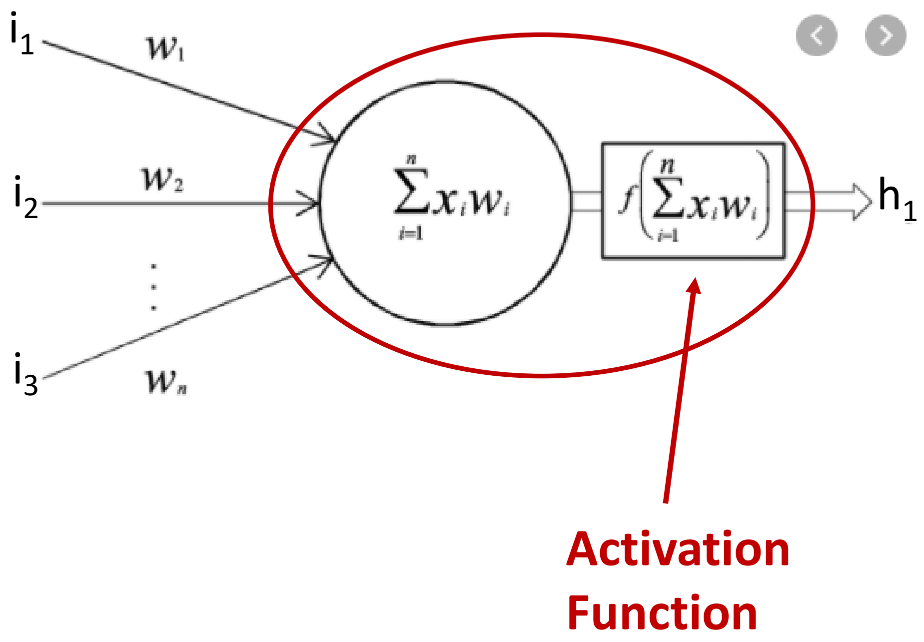
Summary

- Cross Entropy Loss Function and Softmax and their gradients.

Lesson 13

Activation Functions and Their Derivatives

Activation Functions



Activation Function is applied over the linear weighted summation of the incoming information to a node.

Convert linear input signals from perceptron to a linear/non-linear output signal.

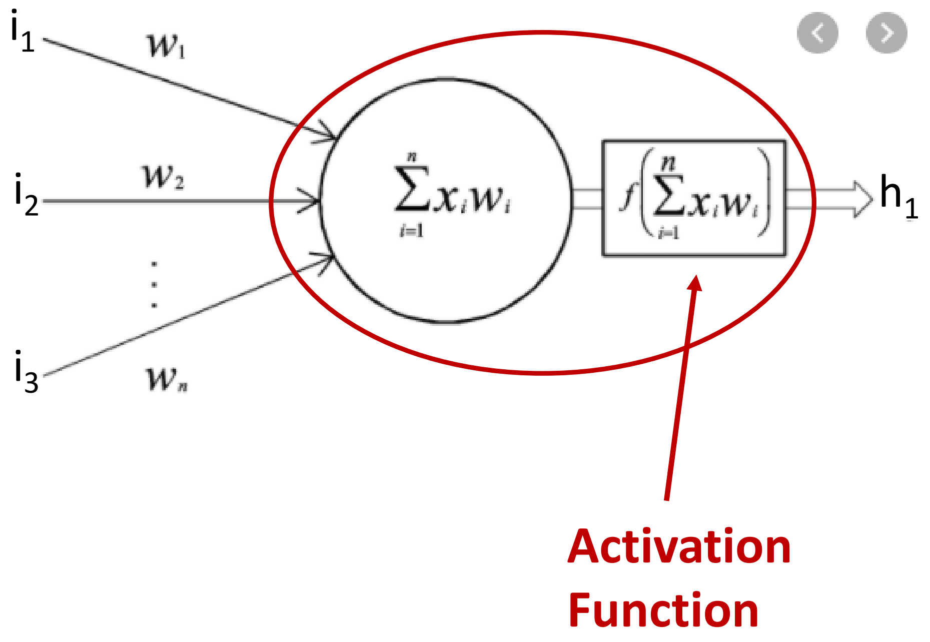
It decides whether to activate a node or not.

Activation Functions

Activation functions must be monotonic, differentiable, and quickly converging.

Types of Activation Functions:

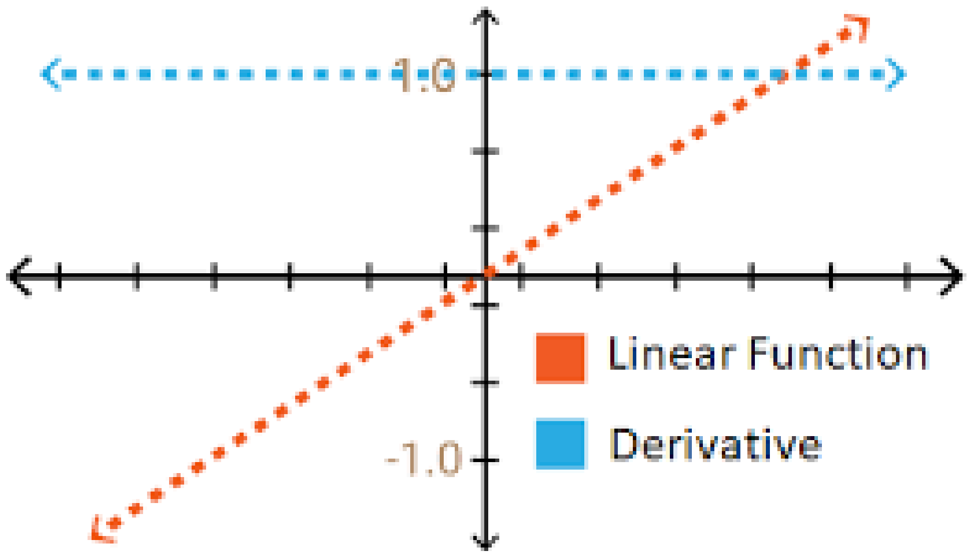
- Linear
- Non-Linear



Linear

$$f(x) = ax + b$$

$$\frac{df(x)}{dx} = a$$



Observations:

- Constant gradient
- Gradient does not depend on the change in the input

Linear

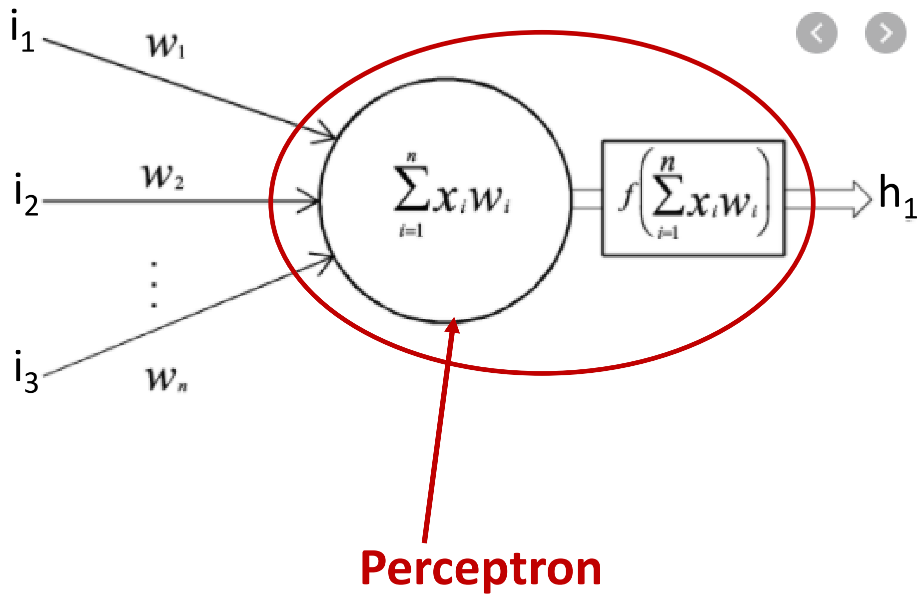
$$f(x) = ax + b$$

$$f(x) = a_1x_1 + a_2x_2 + a_3x_3 + \cdots + b$$

Linear

$$f(x) = ax + b$$

$$f(x) = a_1x_1 + a_2x_2 + a_3x_3 + \cdots + b$$



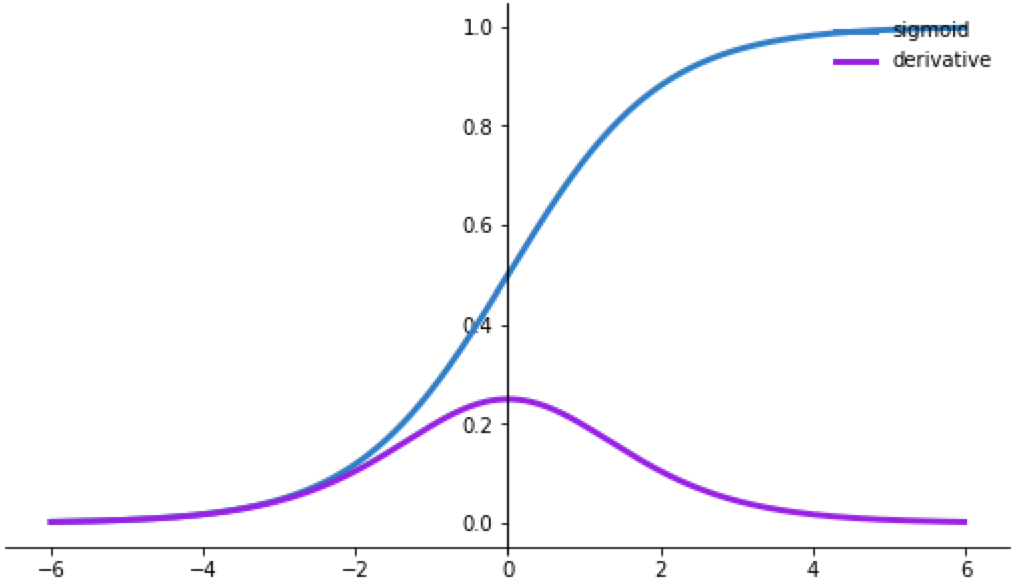
Non-Linear

- Sigmoid (Logistic)
- Hyperbolic Tangent (Tanh)
- Rectified Linear Unit (ReLU)
  - *Leaky Relu*
  - *Parametric Relu*
- Exponential Linear Unit (ELU)

Sigmoid Activation Functions (Logistics)

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

$$\frac{df(x)}{dx} = f(x)(1 - f(x))$$



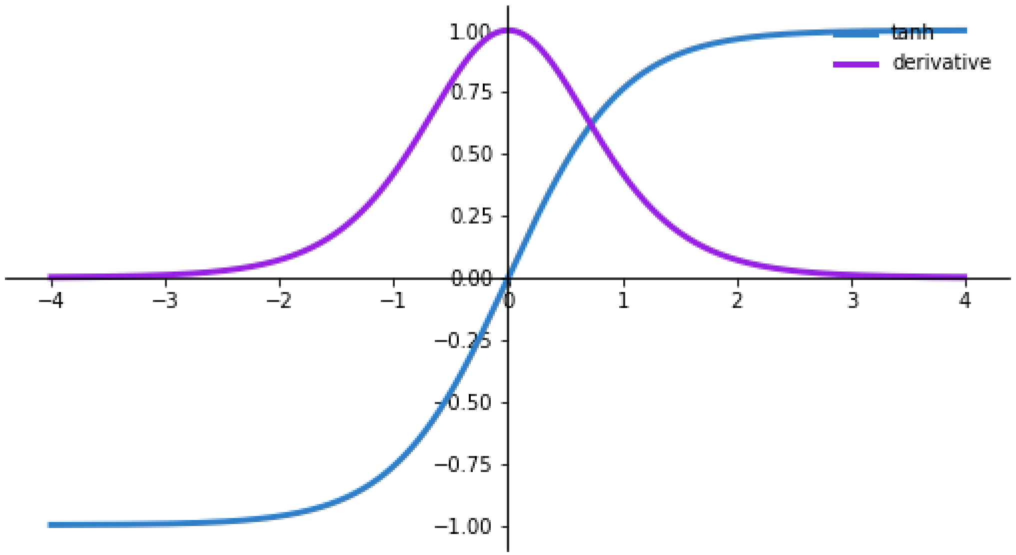
Observations:

- Output: 0 to 1
- Outputs are not zero-centered
- Can saturate and kill (vanish) gradients

Tanh Activation Function

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

$$\frac{df(x)}{dx} = 1 - f(x)^2$$

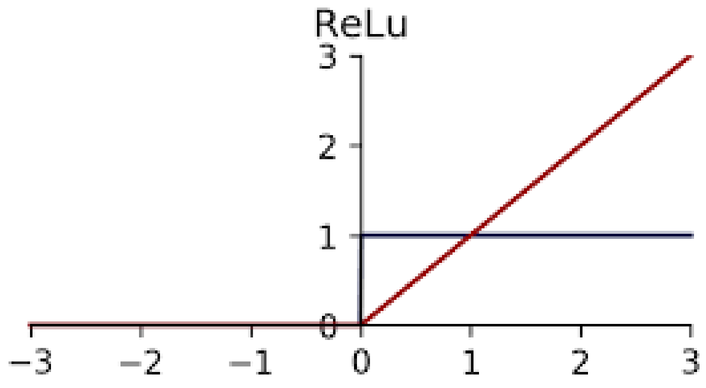


Observations:

- Output: -1 to +1
- Outputs are zero-centered
- Can Saturate and kill (vanish) gradients
- Gradient is more steeped than Sigmoid, resulting in faster convergence

Rectified Linear Unit(ReLU)

$$f(x) = \max(0, x)$$



$$\frac{df(x)}{dx} = 1$$

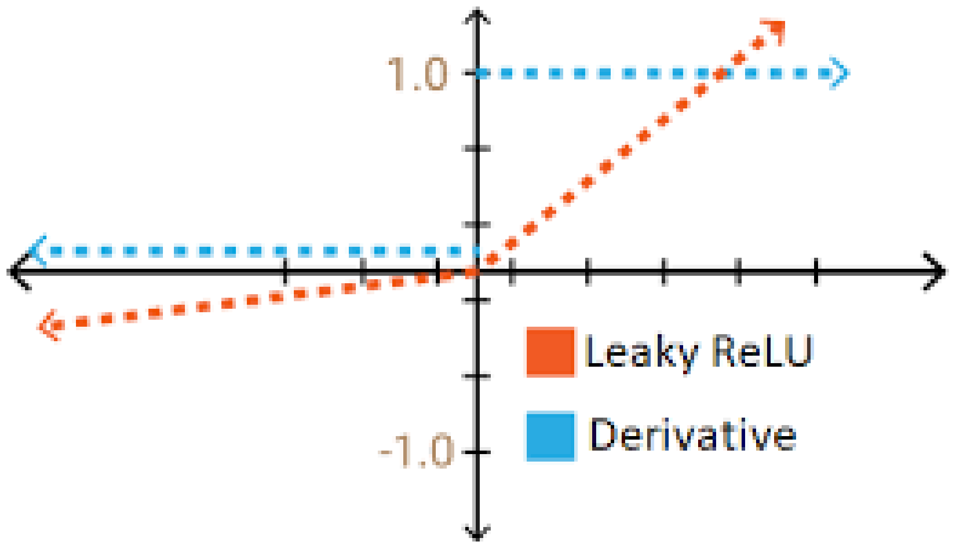
Observations:

- Greatly increase training speed compared to tanh and sigmoid
- Reduces likelihood of killing(vanishing) gradient
- It can blow up activation
- Dead nodes

Leaky-ReLU

$$f(x) = \max(0.01x, x)$$

$$\frac{df(x)}{dx} = \begin{cases} 0.01, & x < 0 \\ 1, & x \geq 0 \end{cases}$$



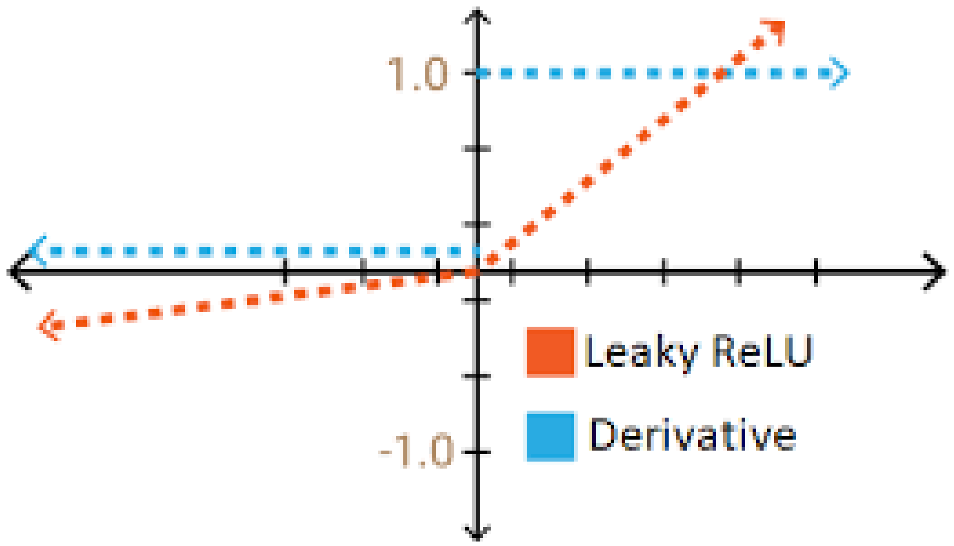
Observations:

- Fixed dying ReLU

Parameterized-ReLU

$$f(x) = \max(\alpha x, x)$$

$$\frac{df(x)}{dx} = \begin{cases} \alpha, & x < 0 \\ 1, & x \geq 0 \end{cases}$$

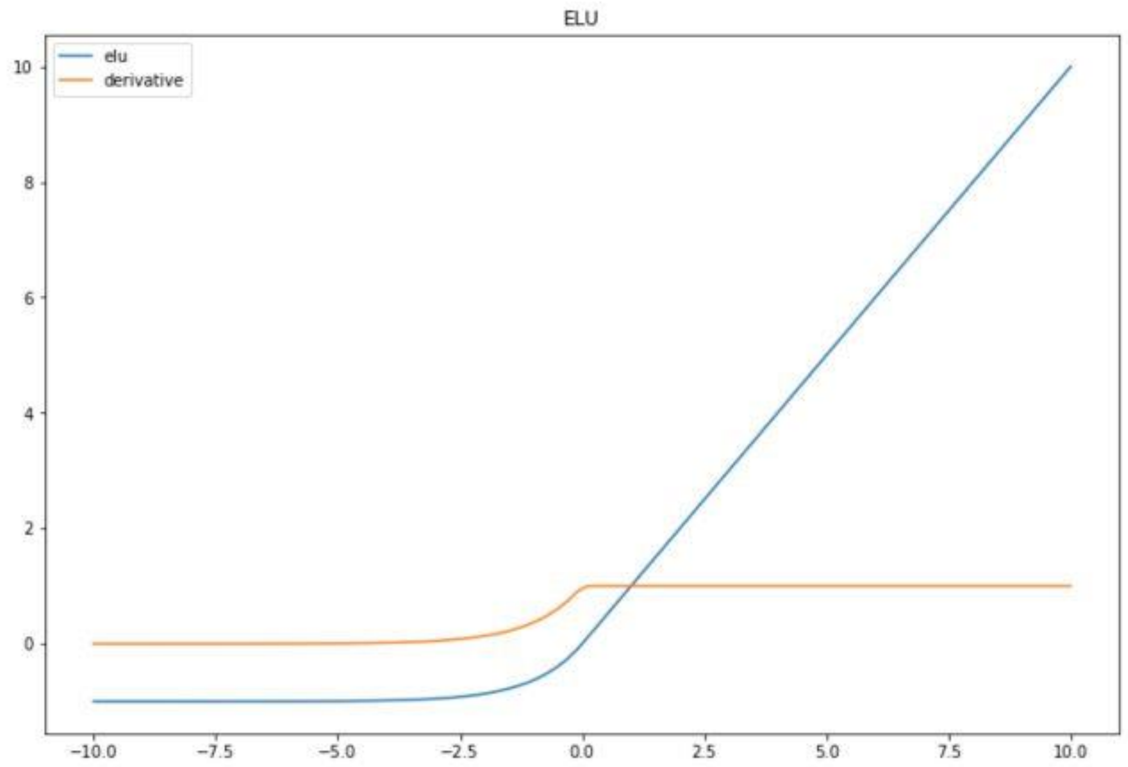


Observations:

Exponential Linear Unit (ELU)

$$\mathrm{f}(x) = \begin{cases} \alpha(e^{x} - 1), & x < 0 \\ 1x \;\; x \geq 0 \end{cases}$$

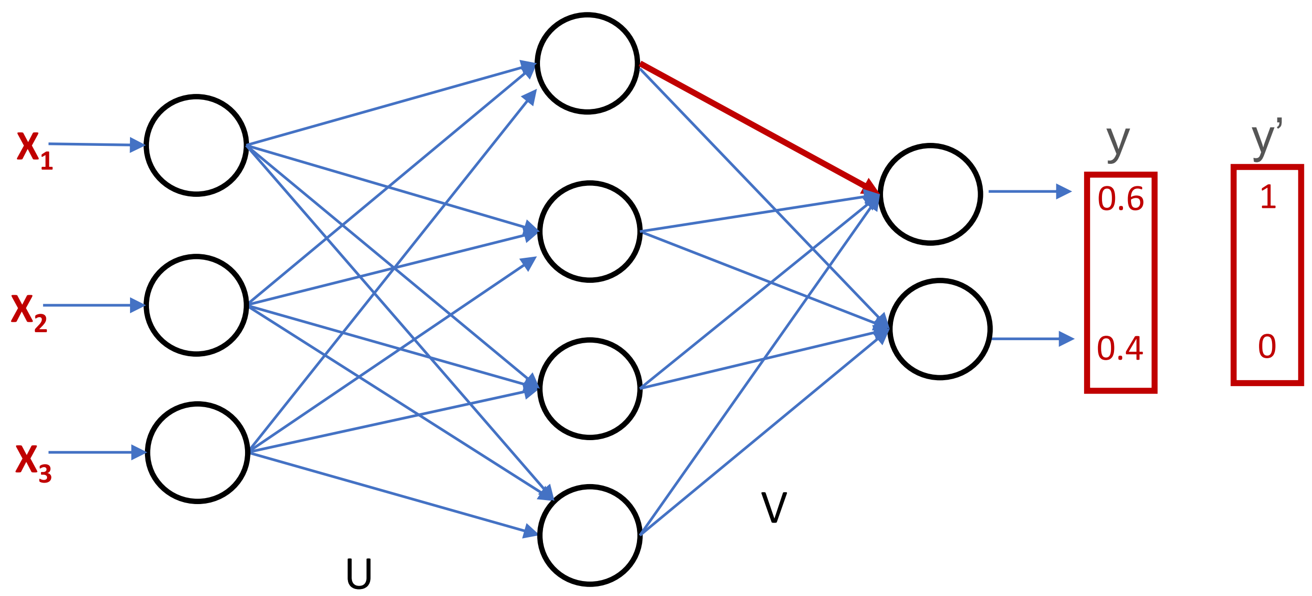
$$\frac{df(x)}{dx} = \begin{cases} f(x) + \alpha, & x < 0 \\ 1, & x \geq 0 \end{cases}$$

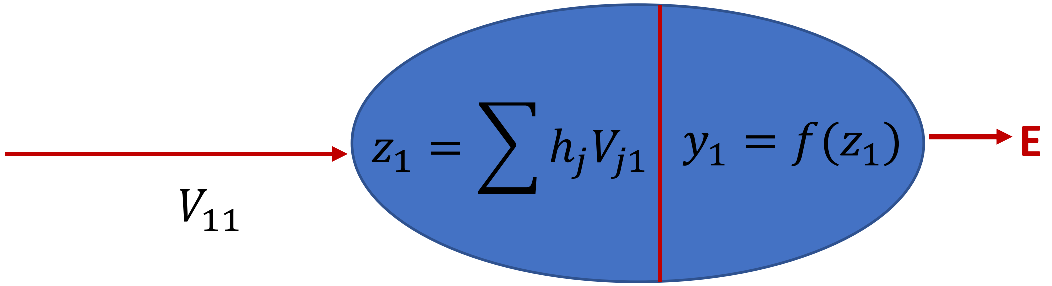


Observations:

- **It can produce –ve output**
- **It can blow up activation function**

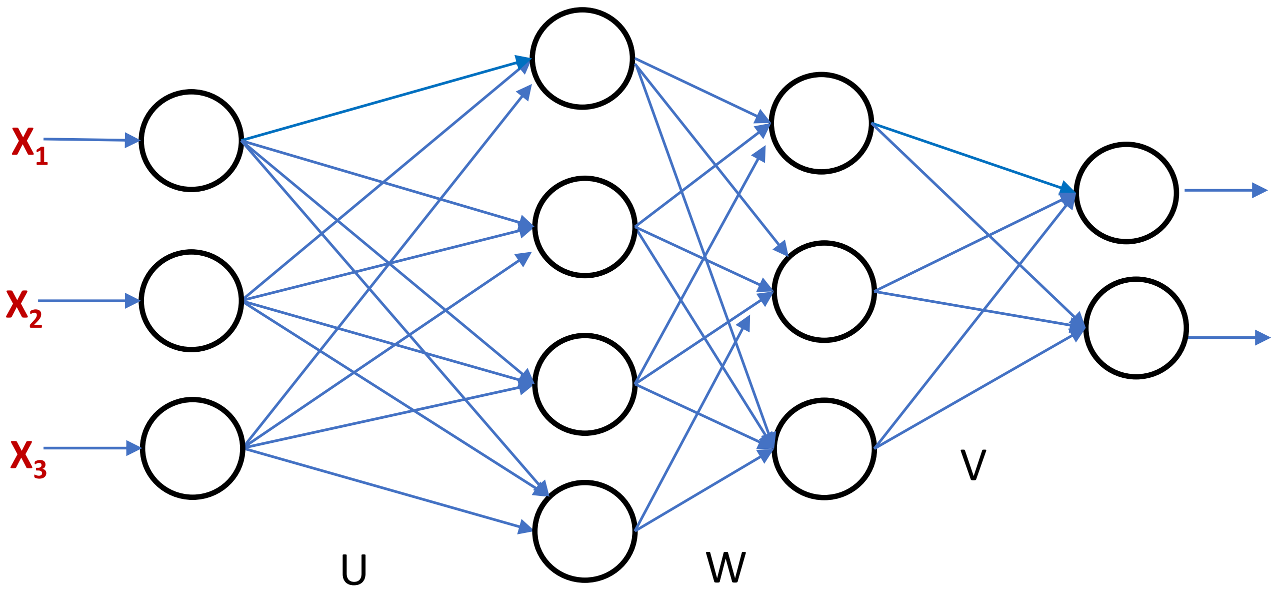
Complete Chain



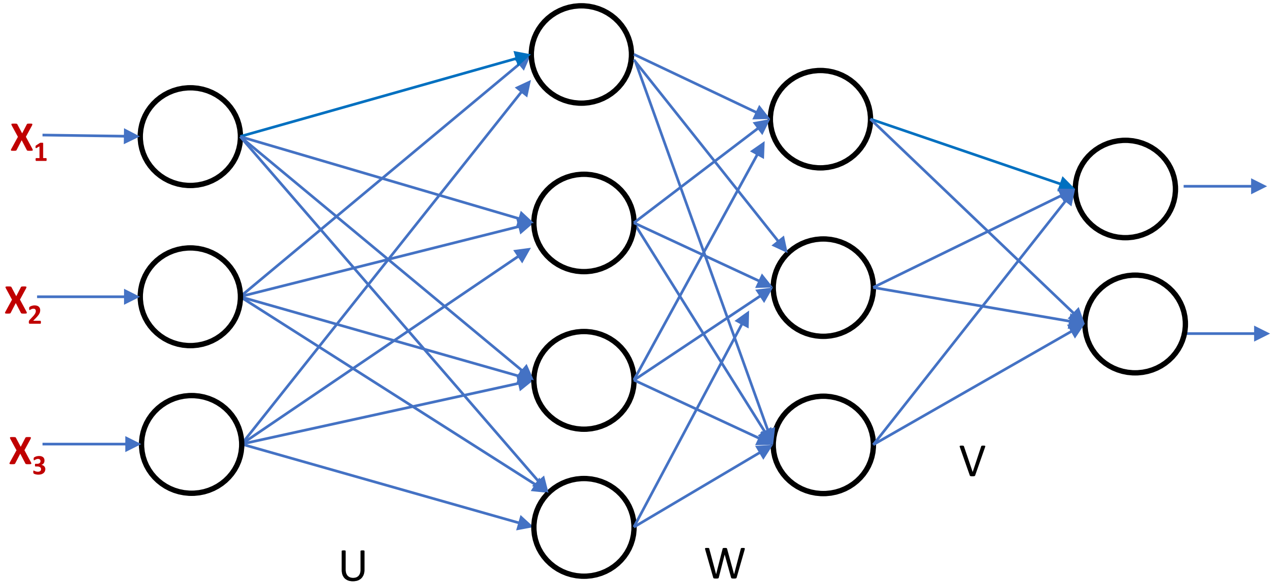


$$\frac{\delta E}{\delta V_{11}} = \frac{\delta z_1}{\delta V_{11}} \times \frac{\delta y_1}{\delta z_1} \times \frac{\delta E}{\delta y_1}$$

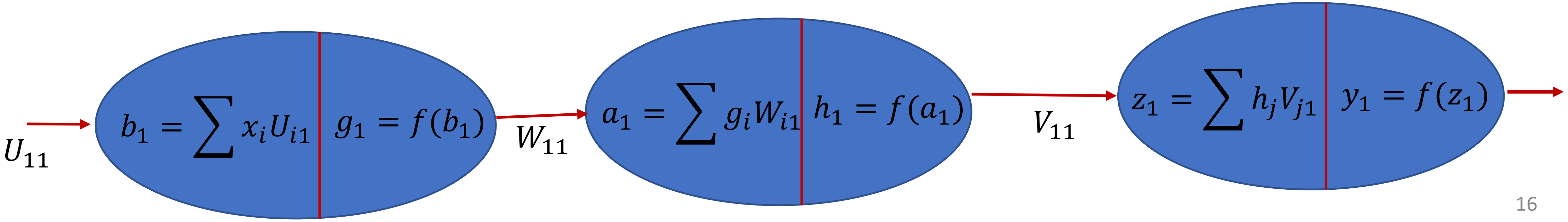
Deep Network



Deep Network - Vanishing/Exploding Gradient



$$\frac{\delta E}{\delta U_{11}} = \frac{\delta b_1}{\delta U_{11}} \times \frac{\delta g_1}{\delta b_1} \times \frac{\delta a_1}{\delta g_1} \times \frac{\delta h_1}{\delta a_1} \times \frac{\delta z_1}{\delta h_1} \times \frac{\delta y_1}{\delta z_1} \times \frac{\delta E}{\delta y_1}$$



Summary

- We learn characteristics of different Activation Functions and their gradient
- The choice of activation function depend on the nature of the problem, nature of the target output and the deepness of the network.