Pemrograman Artificial Neural Network dengan Python

Rabu, 15 Januari 2020

PROBLEM SET

	INPUTS			OUTPUTS
Example 1	0	0	1	0
Example 2	1	1	1	1
Example 3	1	0	1	1
Example 4	0	1	1	0

New situation	1	0	0	?

What should the new output be?

PROBLEM SET

	INPUTS			OUTPUTS
Example 1	0	0	1	0
Example 2	1	1	1	1
Example 3	1	0	1	1
Example 4	0	1	1	0
New situation	1	0	0	?

What should the new output be?

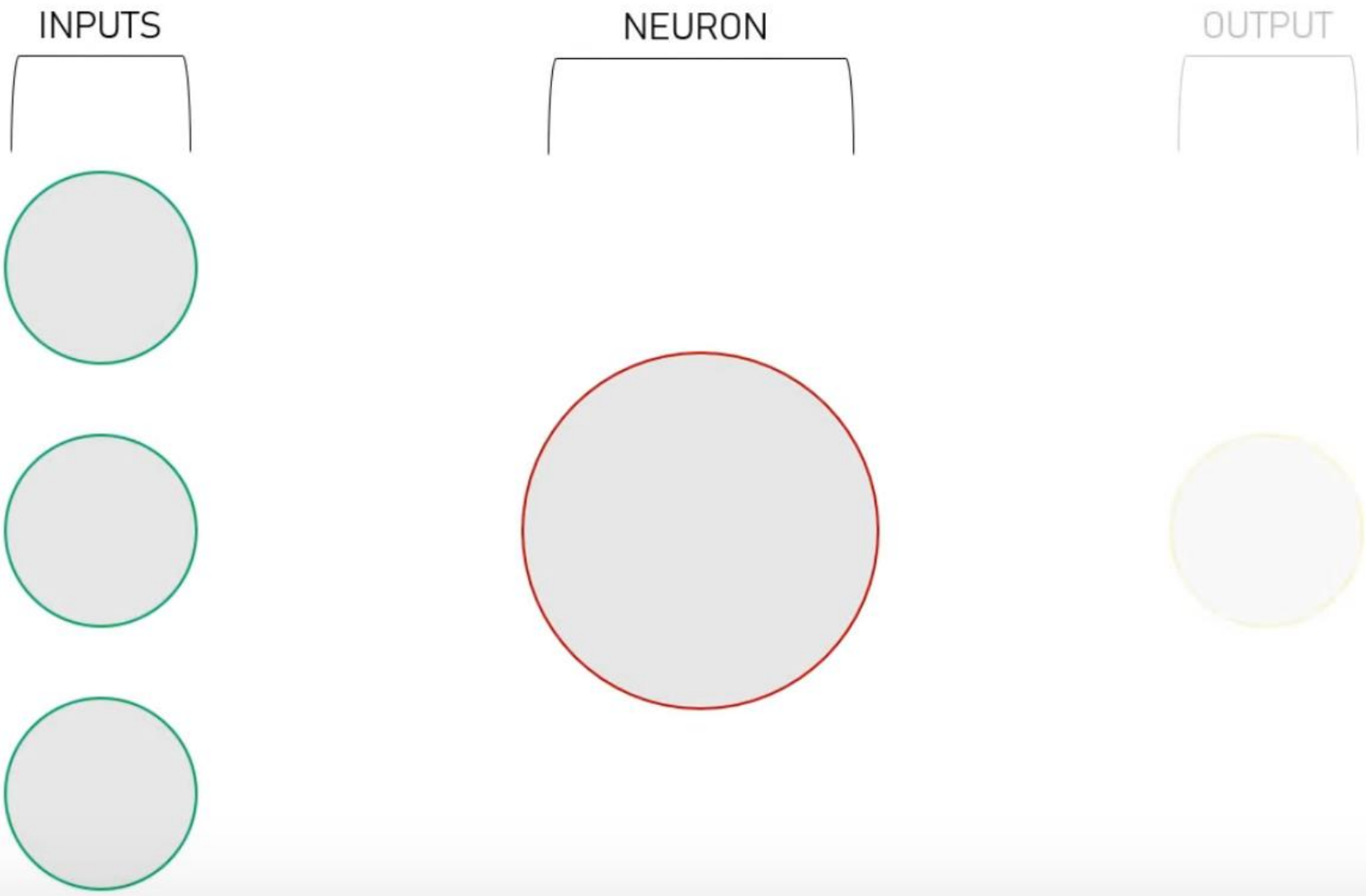
PROBLEM SET

	INPUTS			OUTPUTS
Example 1	0	0	1	0
Example 2	1	1	1	1
Example 3	1	0	1	1
Example 4	0	1	1	0

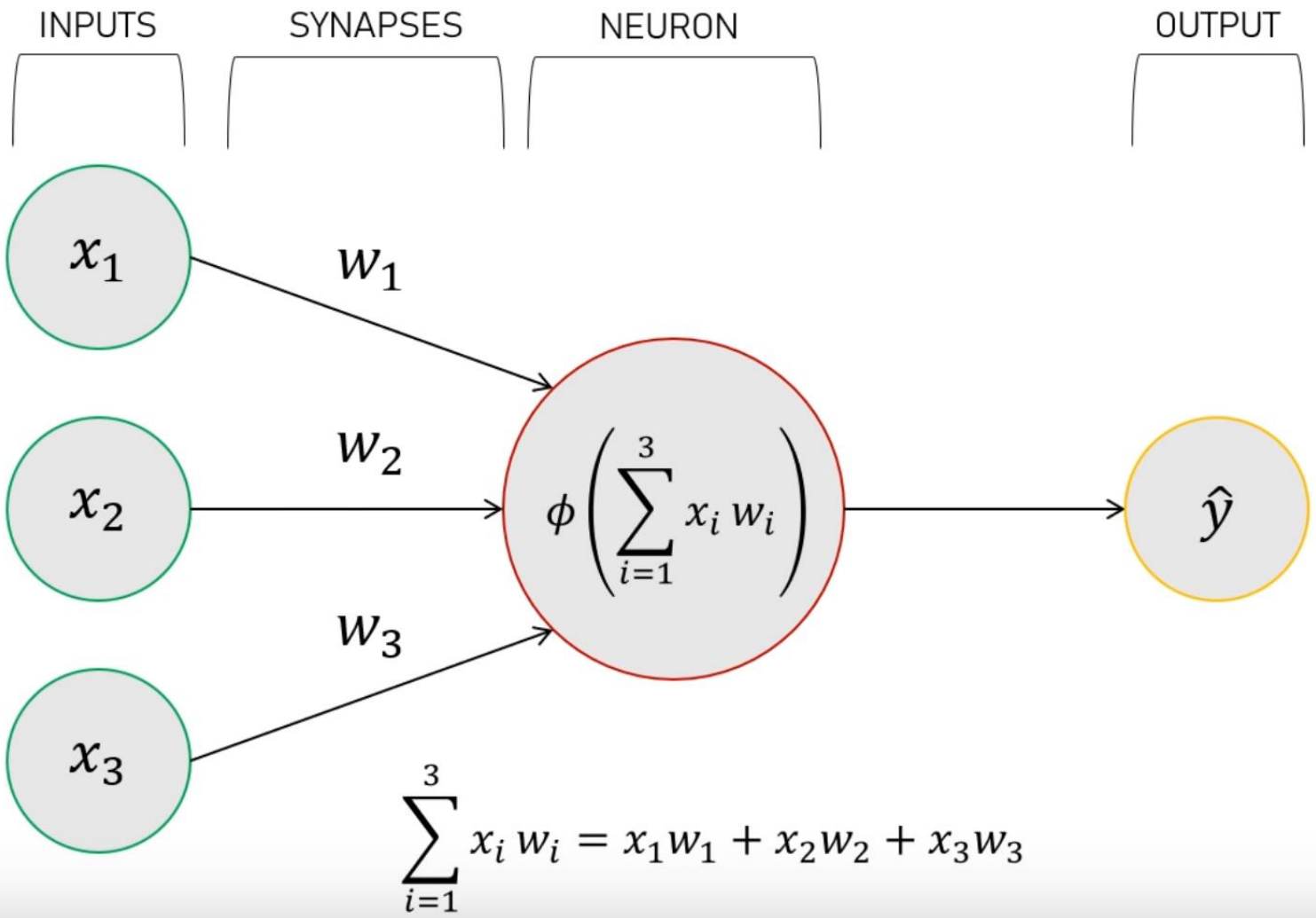
New situation	1	0	0	1

What should the new output be?

PERCEPTRON

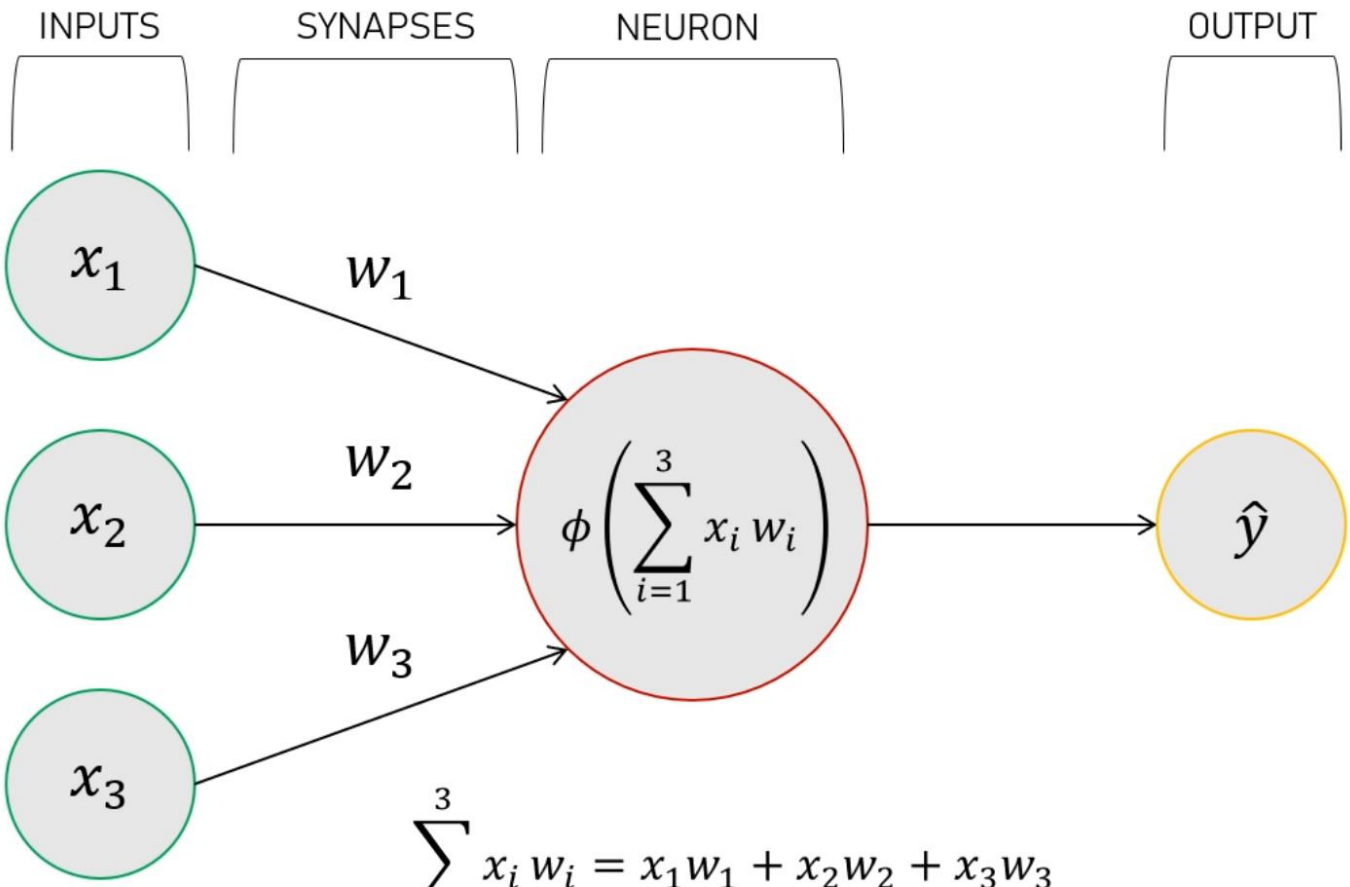


PERCEPTRON



$$\sum_{i=1}^{3} x_i\, w_i = x_1 w_1 + x_2 w_2 + x_3 w_3$$

PERCEPTRON

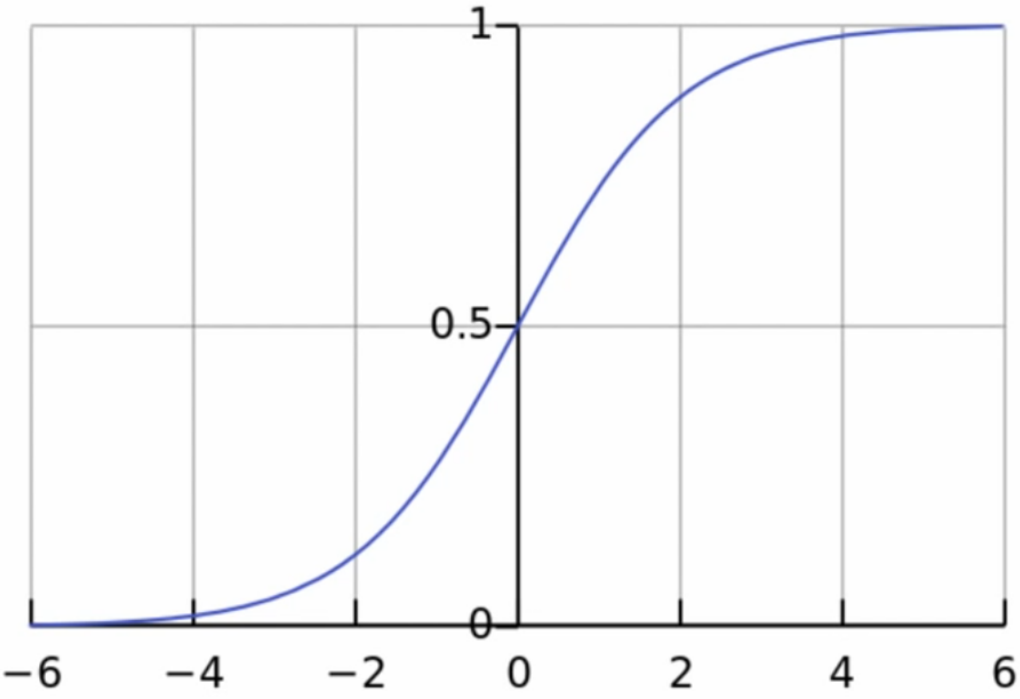


$$\sum_{i=1}^{3} x_i\, w_i = x_1 w_1 + x_2 w_2 + x_3 w_3$$

$$\phi = normalizing\ function$$

SIGMOID NORMALIZING FUNCTION

$$\phi(x) = \frac{1}{1 + e^{-x}} \longrightarrow \phi(x) = \frac{1}{1 + e^{-\sum_{i=1}^{3} x_i w_i}}$$



CODE ALONG PART 1

SOME CALCULATIONS

	INPUTS			OUTPUTS
Example 1	0	0	1	0
Example 2	1	1	1	1
Example 3	1	0	1	1
Example 4	0	1	1	0

SOME CALCULATIONS

	INPUTS			OUTPUTS
Example 1	0	0	1	0
Example 2	1	1	1	1
Example 3	1	0	1	1
Example 4	0	1	1	0

$x_1 = 0$
$x_2 = 0$
$x_3 = 1$

```
Random starting synaptic weights:
[[-0.16595599]
 [ 0.44064899]
 [-0.99977125]]
Outputs after training:
[[0.2689864 ]
 [0.3262757 ]
 [0.23762817]
 [0.36375058]]
```

SOME CALCULATIONS

	INPUTS			OUTPUTS
Example 1	0	0	1	0
Example 2	1	1	1	1
Example 3	1	0	1	1
Example 4	0	1	1	0

$x_1 = 0$
$x_2 = 0$
$x_3 = 1$

$w_1 = -0{,}165$
$w_2 = 0{,}440$
$w_3 = -0{,}999$

SOME CALCULATIONS

	INPUTS			OUTPUTS
Example 1	0	0	1	0
Example 2	1	1	1	1
Example 3	1	0	1	1
Example 4	0	1	1	0

$x_1 = 0$
$x_2 = 0$
$x_3 = 1$

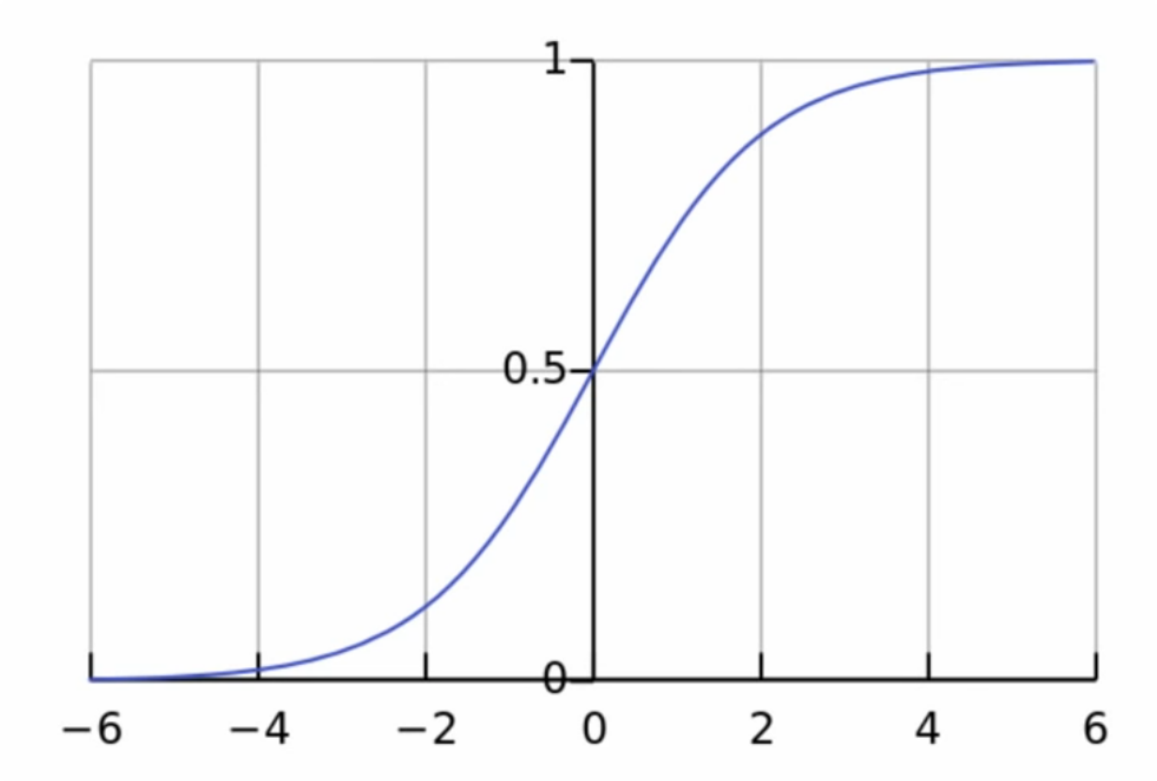
$w_1 = -0{,}165$
$w_2 = 0{,}440$
$w_3 = -0{,}999$

$$\sum_{i=1}^{3} x_i\, w_i = x_1 w_1 + x_2 w_2 + x_3 w_3$$

$$\sum_{i=1}^{3} x_i\, w_i = 0\,.\,(-0{,}165) + 0\,.\,0{,}440 + 1\,.\,(-0{,}999) = \mathbf{-0{,}999}$$

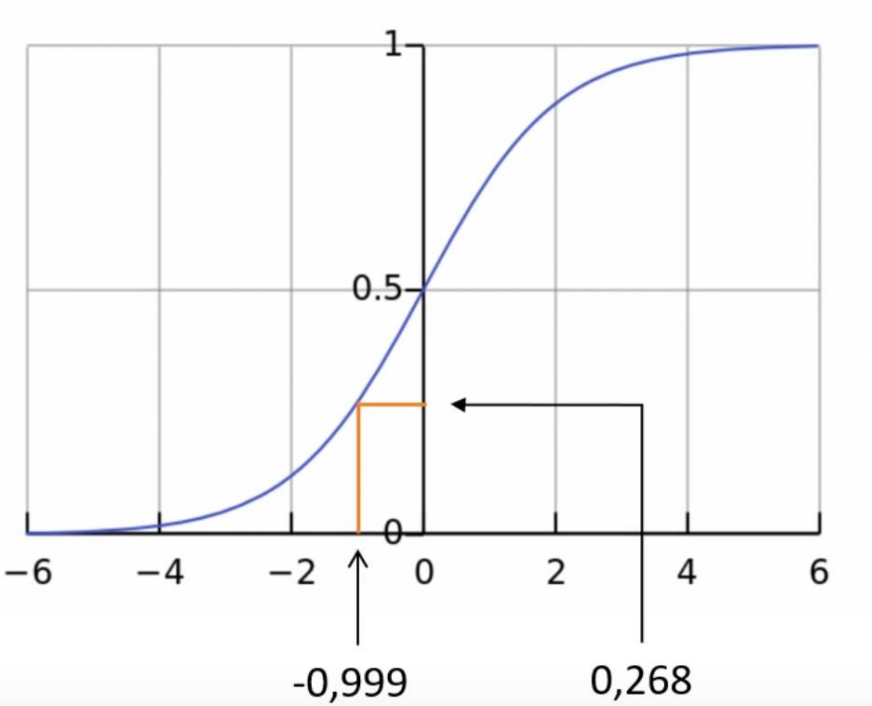
SIGMOID NORMALIZING FUNCTION

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



SIGMOID NORMALIZING FUNCTION

$$\phi(x) = \frac{1}{1 + e^{-x}} \longrightarrow \phi(x) = \frac{1}{1 + e^{(-\sum_{i=1}^{3} x_i w_i)}}$$




```
Random starting synaptic weights:
[[-0.16595599]
 [ 0.44064899]
 [-0.99977125]]
Outputs after training:
[[0.2689864 ]
 [0.3262757 ]
 [0.23762817]
 [0.36375058]]
```

TRAINING PROCESS

1. Take the inputs from the training example, and put them through our formula to get the neuron's output

2. Calculate the error, which is the difference between the output we got, and the actual output

3. Depending on the severeness of the error, adjust the weights accordingly

4. Repeat this 20,000 times

ERROR WEIGHTED DERIVATIVE

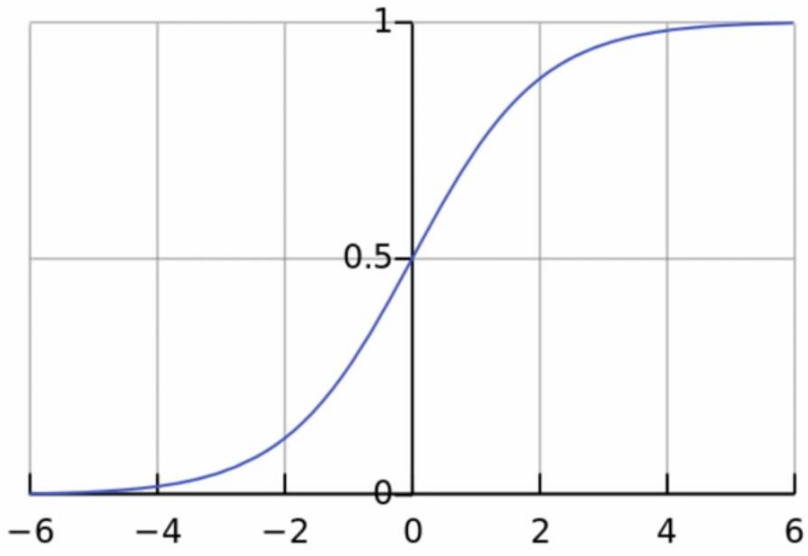
Adjust weights by $= error\ .\ input\ .\ \phi'(output)$

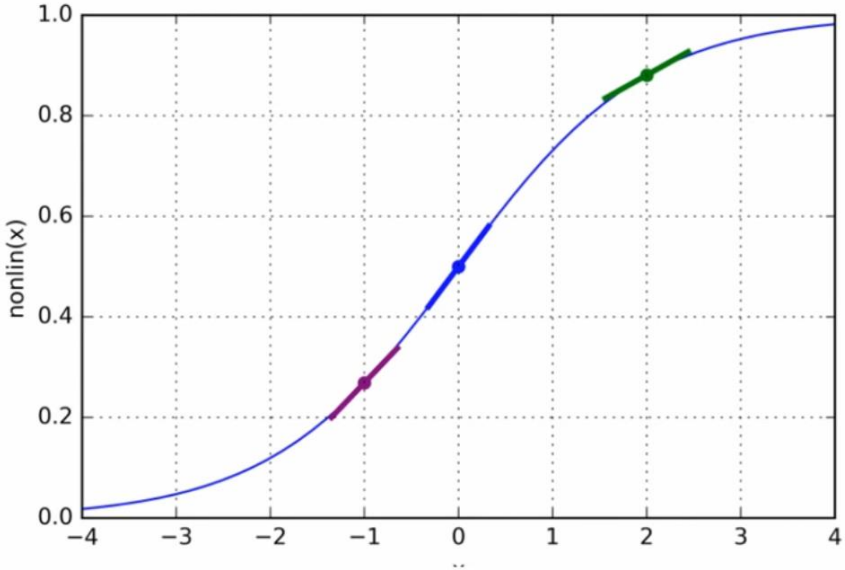
$$Error = output\ - actual\ output$$

$$Input = either\ 1\ or\ 0$$

ADJUSTMENTS

$$\phi(x) = \frac{1}{1 + e^{-x}} \longrightarrow \phi'(x) = x \,.\, (1 - x)$$

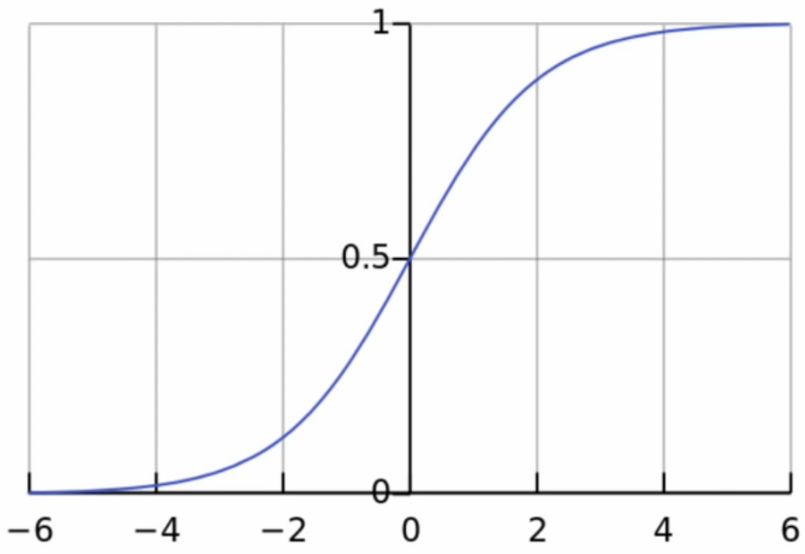


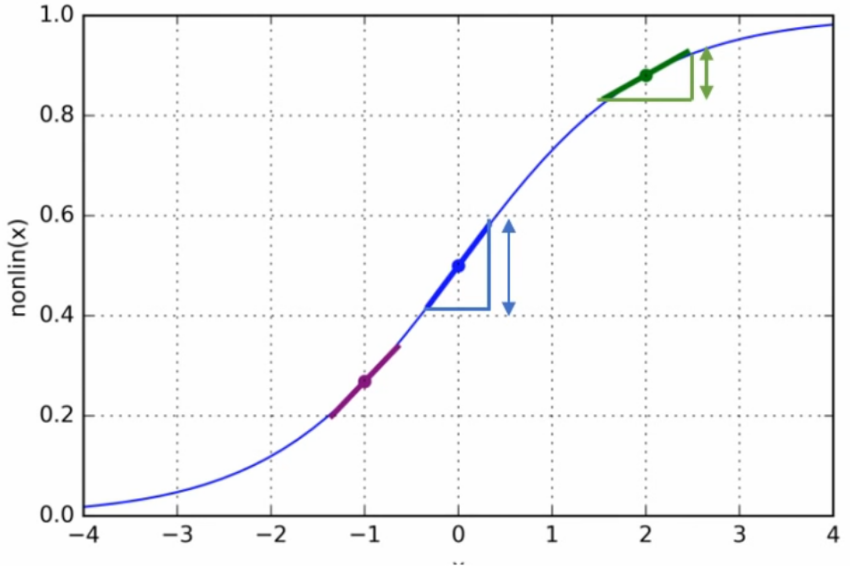


Adjust weights by $= error \,.\, input \,.\, \phi'(output)$

ADJUSTMENTS

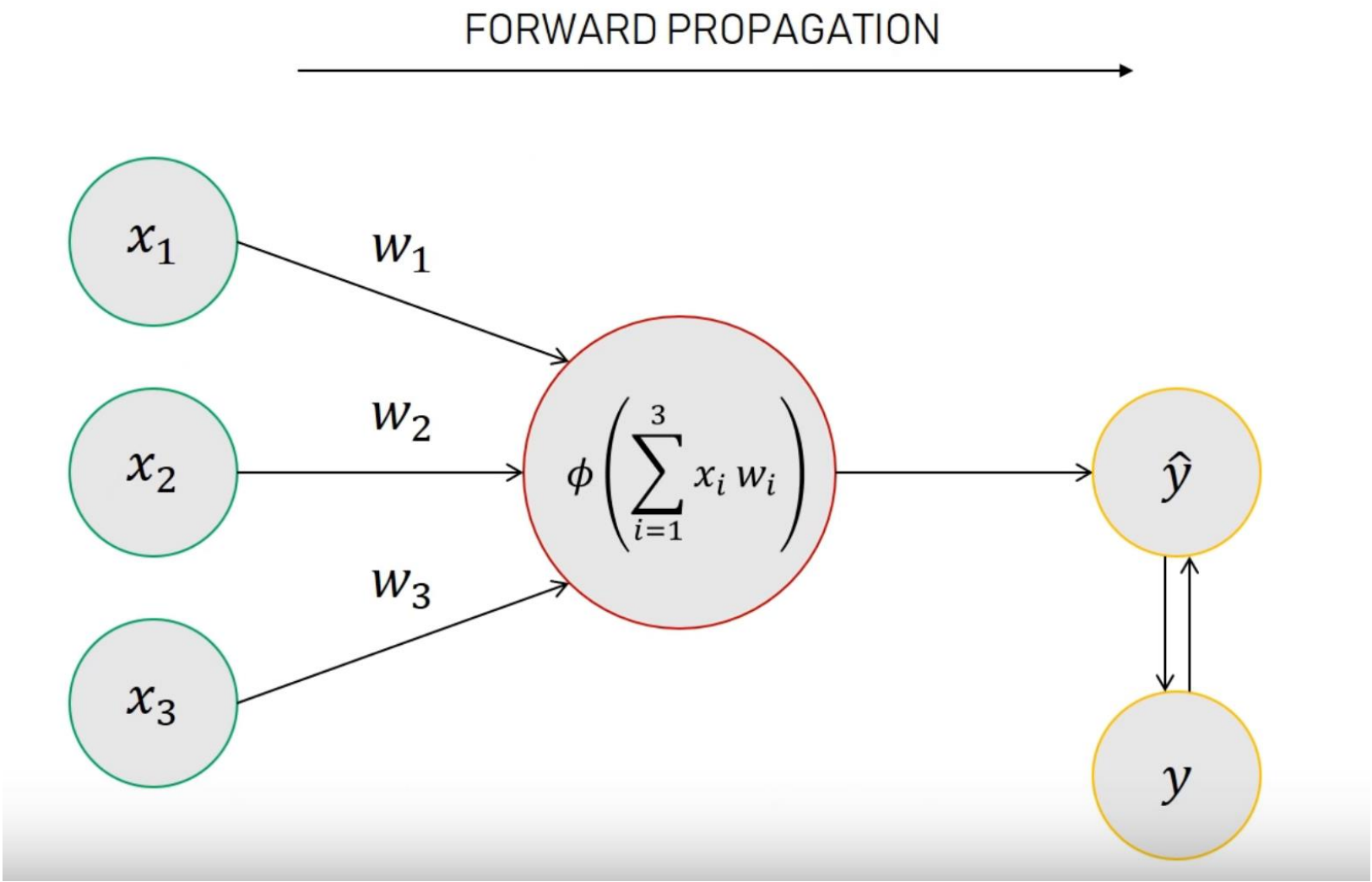
$$\phi(x) = \frac{1}{1 + e^{-x}} \longrightarrow \phi'(x) = x \,.\, (1 - x)$$



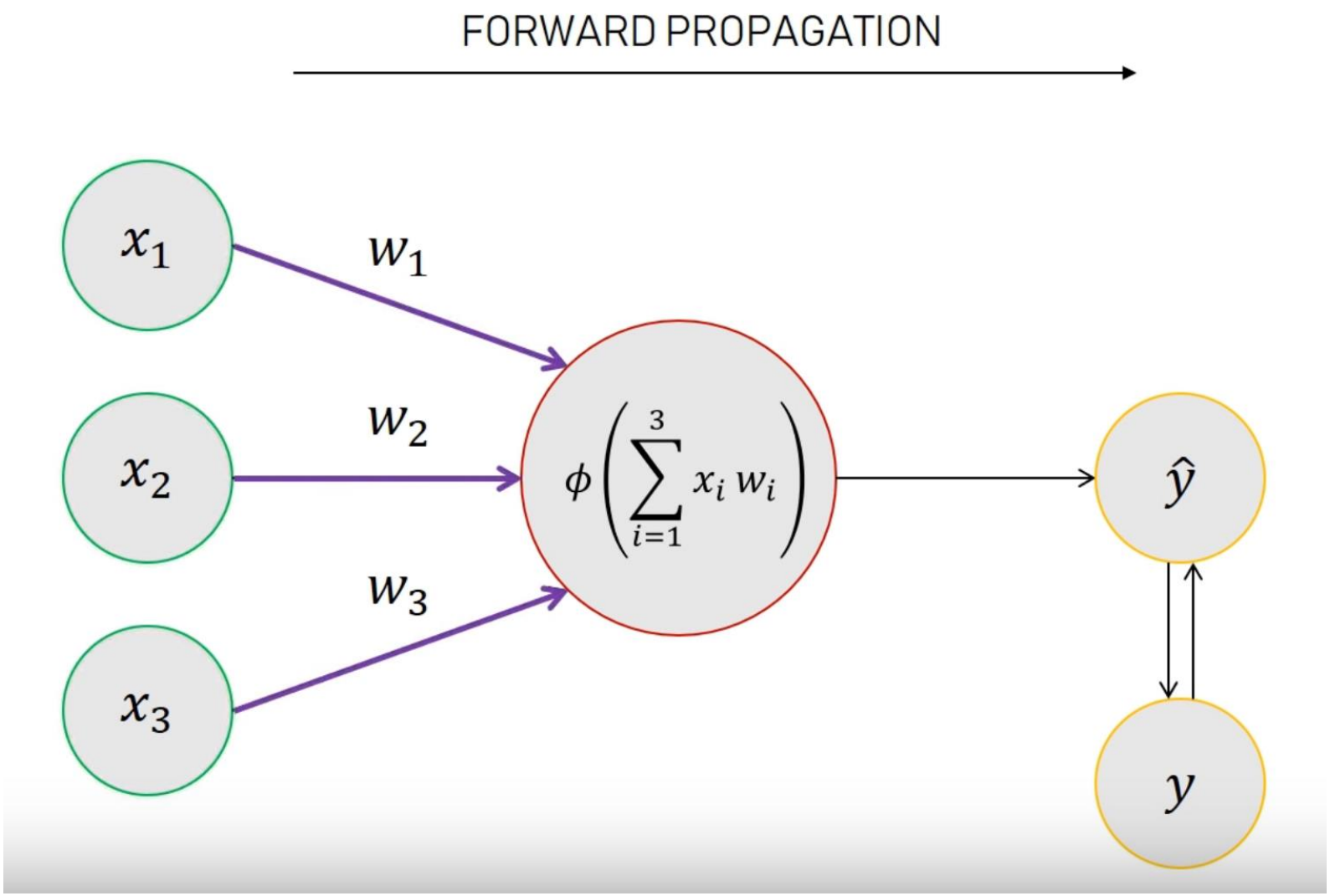


Adjust weights by $= error \,.\, input \,.\, \phi'(output)$

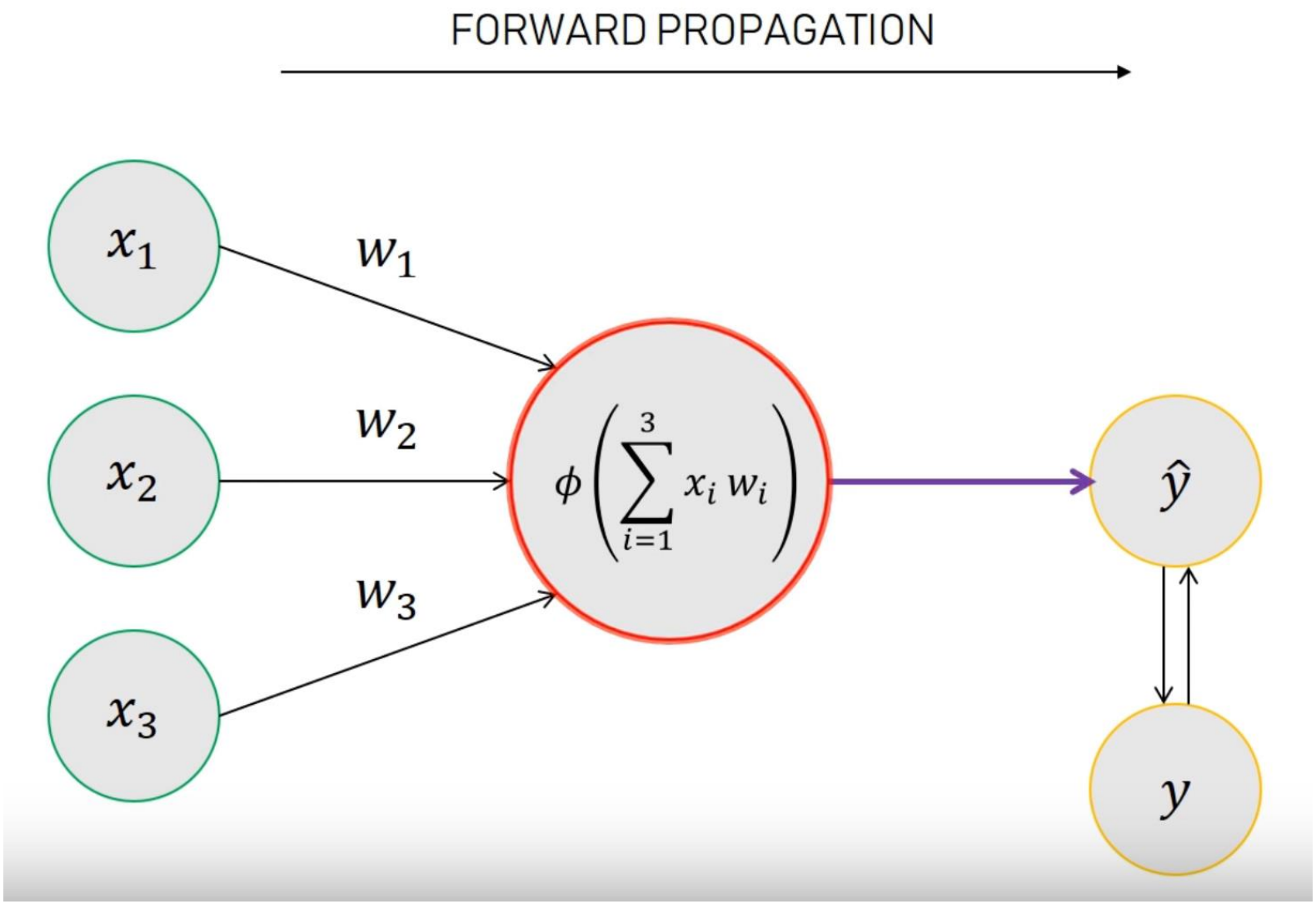
BACKPROPAGATION



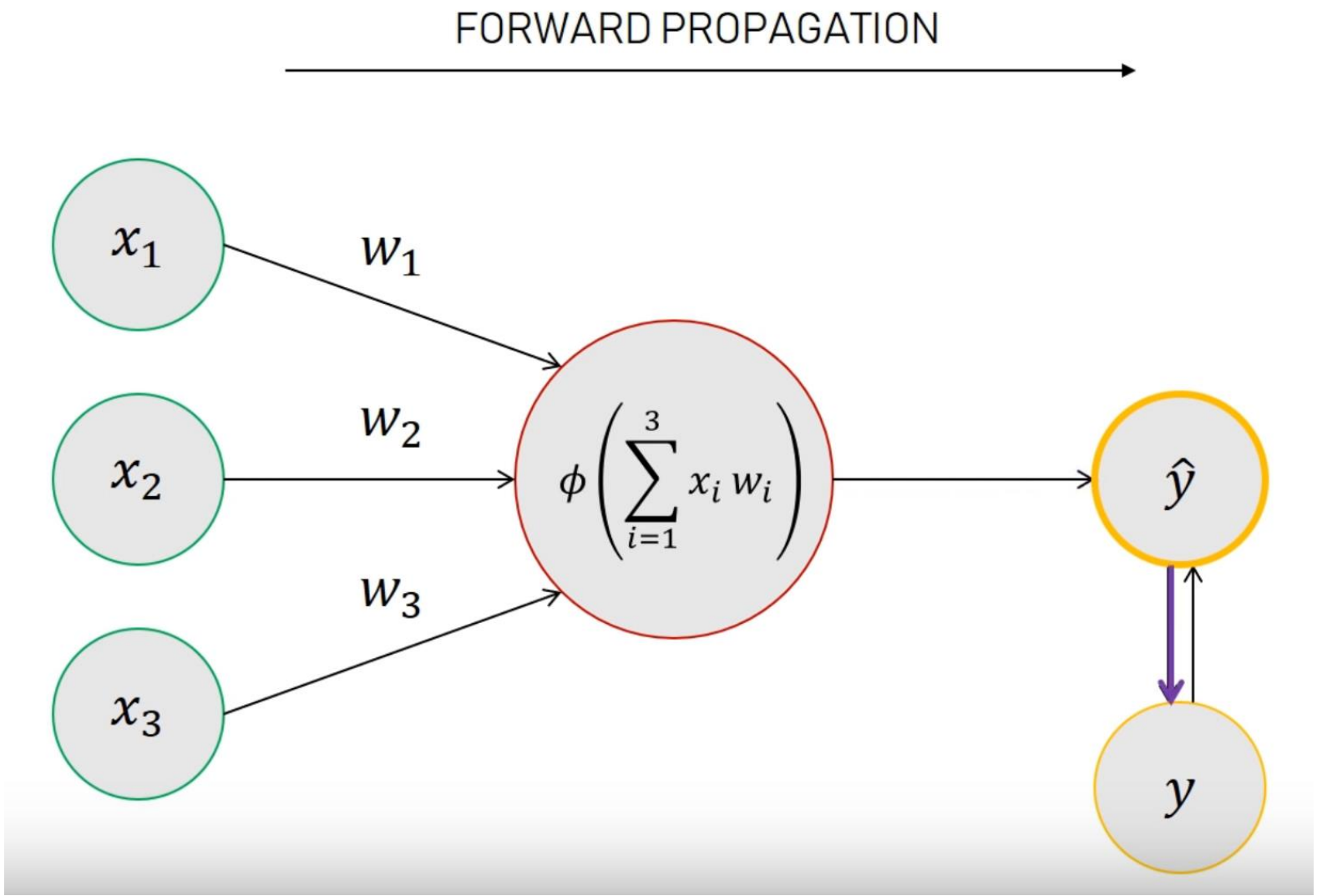
BACKPROPAGATION



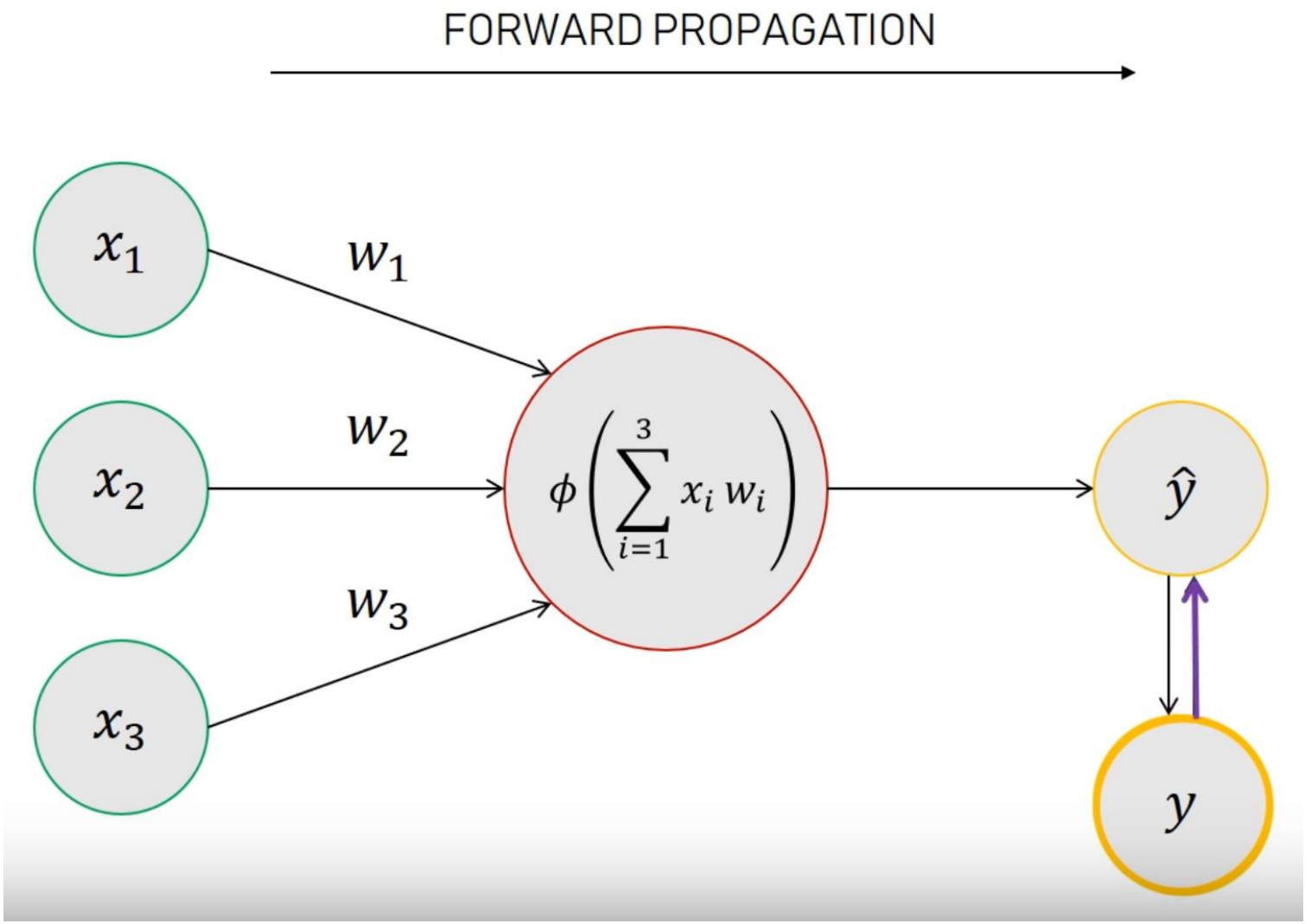
BACKPROPAGATION



BACKPROPAGATION



BACKPROPAGATION



CODE ALONG PART 2