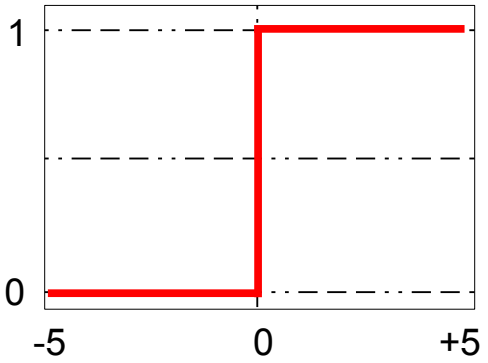
Perceptron Training Procedure

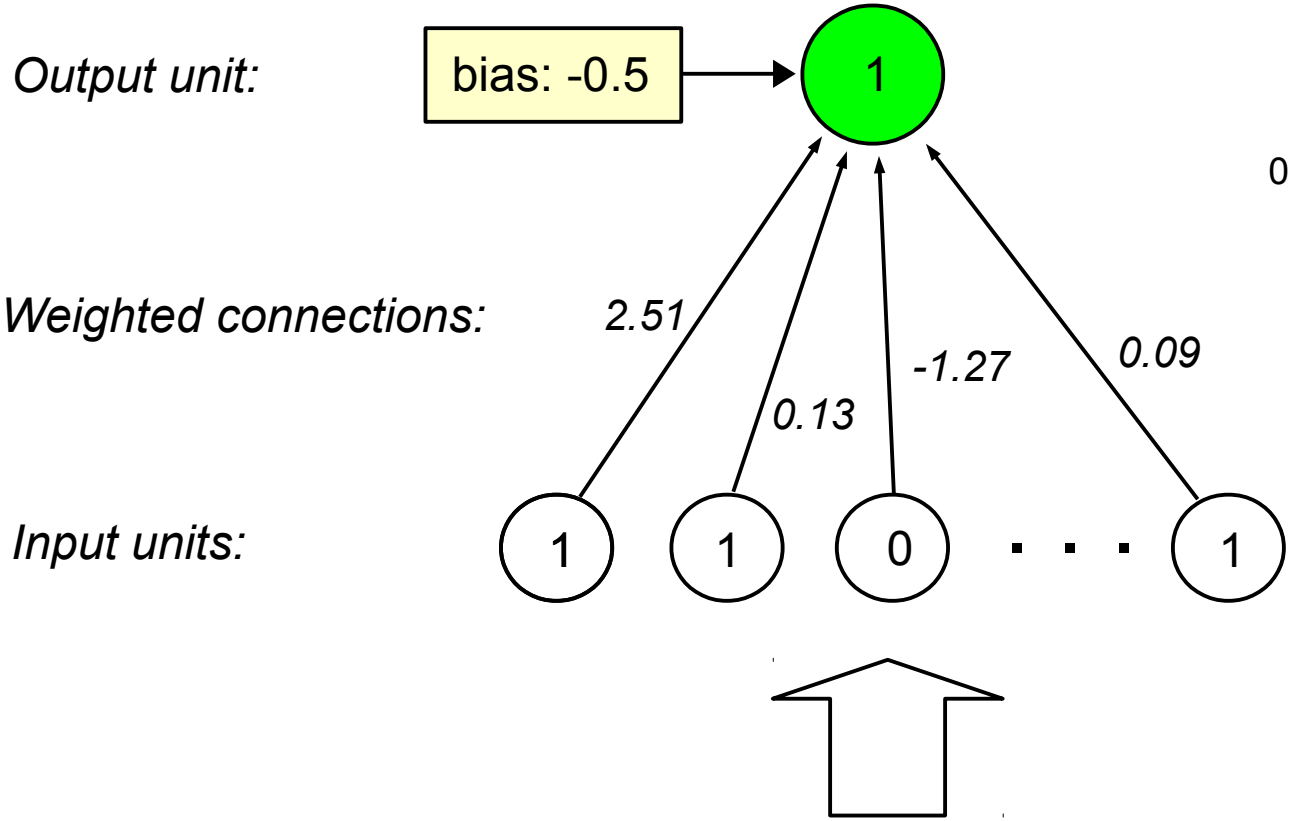
Binary Threshold Neuron

$$\mathbf{1} \times 2.51 + \mathbf{1} \times 0.13 + \mathbf{0} \times -1.27 + \ldots + \mathbf{1} \times 0.09 + \mathbf{-0.5} = 2.23$$

$$2.23 \geq 0$$



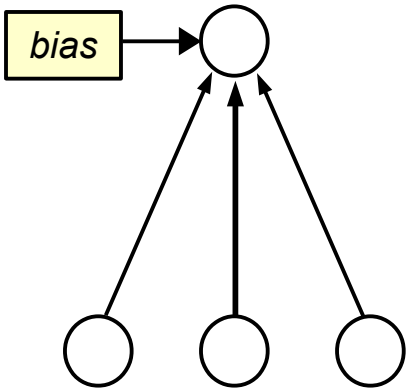
threshold = 0



Input Pattern

Perceptrons

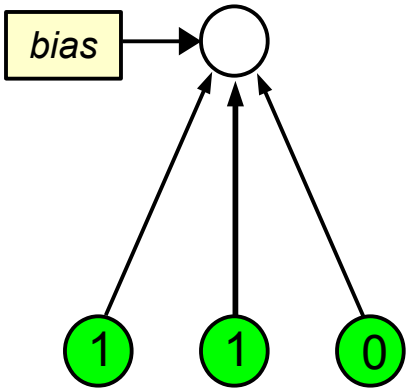
- Binary threshold neurons
- Studied by Frank Rosenblatt of Cornell in early 1960's
- Perceptron training procedure



Perceptrons

- Binary threshold neurons
- Studied by Frank Rosenblatt of Cornell in early 1960's
- Perceptron training procedure
  1. present an input pattern

target = 1



Perceptrons

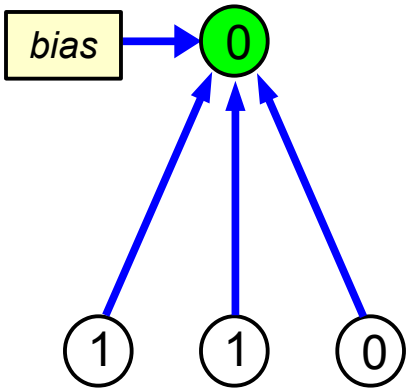
- Binary threshold neurons
- Studied by Frank Rosenblatt of Cornell in early 1960's
- Perceptron training procedure
  1. present an input pattern
  2. compute output value

$$output = \Theta(sum\ of\ inputs \times weights\ +\ bias)$$

"threshold" function:

if sum $\geq$ 0: output = 1
if sum < 0: output = 0

target = 1



Perceptrons

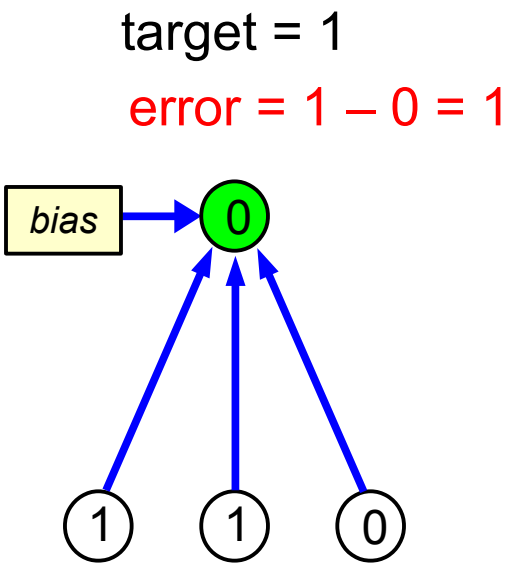
- Binary threshold neurons
- Studied by Frank Rosenblatt of Cornell in early 1960's
- Perceptron training procedure

1. present an input pattern
2. compute output value

   $output = \Theta(sum\ of\ inputs \times weights + bias)$

3. compare output to target value

   $error = target - output$



Perceptrons

- Binary threshold neurons
- Studied by Frank Rosenblatt of Cornell in early 1960's
- Perceptron training procedure

1. present an input pattern
2. compute output value

   $output = \Theta(sum\ of\ inputs \times weights + bias)$

3. compare output to target value

   $error = target - output$

4. if incorrect, adjust weights and bias

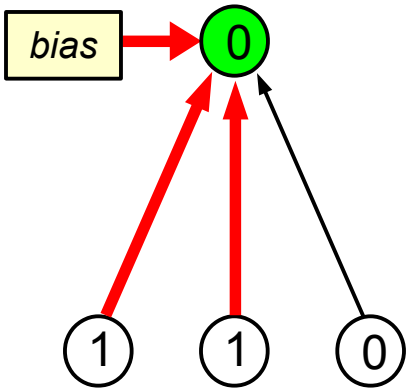
   $weight\_adjustment = \varepsilon \times input \times error$

   $bias\_adjustment = \varepsilon \times error$

   "learning rate" $(0 < \varepsilon < 1)$

target = 1

error = 1 – 0 = 1



Perceptrons

- Binary threshold neurons
- Studied by Frank Rosenblatt of Cornell in early 1960's
- Perceptron training procedure

1. present an input pattern
2. compute output value

   $output = \Theta(sum\ of\ inputs \times weights + bias)$

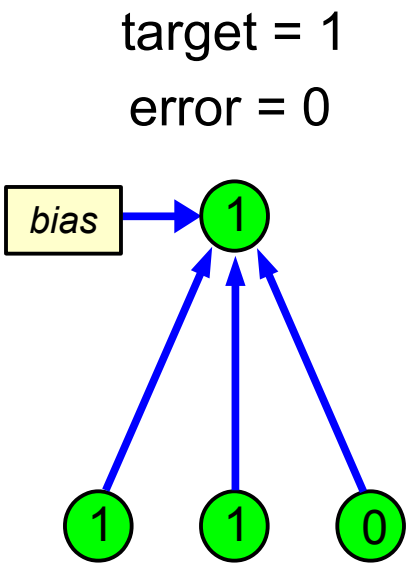
3. compare output to target value

   $error = target - output$

4. if incorrect, adjust weights and bias

   $weight\_adjustment = \varepsilon \times input \times error$

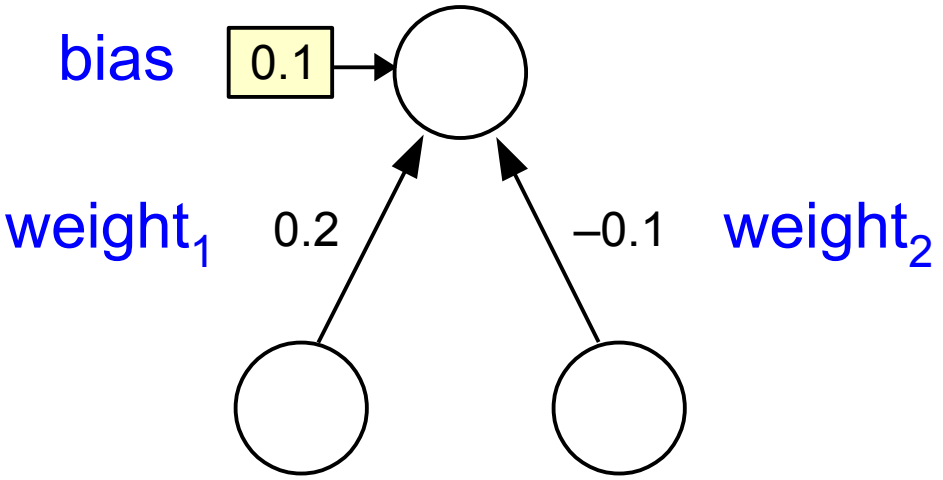
5. repeat until all input patterns give the correct output value



Perceptron Training Example

Input			Target	Output
0	0	⇒	1	
0	1	⇒	0	
1	0	⇒	1	
1	1	⇒	0	

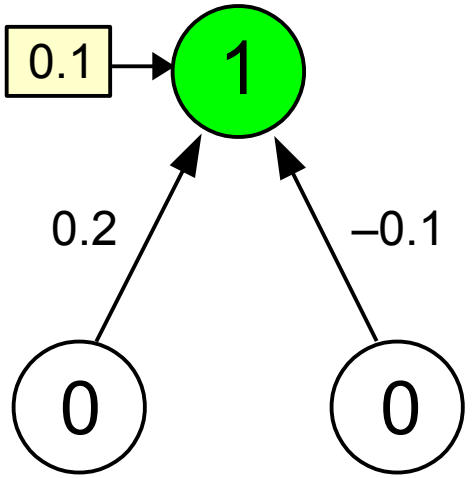
Learning rate $\varepsilon = 0.2$



Compute Output for Pattern 0 0

Input	Target	Output	
0 0	⇒ 1	1	ok
0 1	⇒ 0		
1 0	⇒ 1		
1 1	⇒ 0		

Learning rate $\varepsilon = 0.2$

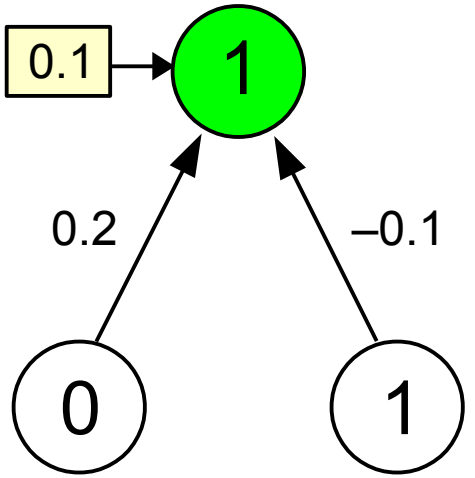


$$\mathbf{0} \times 0.2 + \mathbf{0} \times -0.1 + 0.1 = 0.1 \geq 0 \qquad \text{output} = 1$$

Compute Output for Pattern 0 1

Input	Target	Output	
0 0	⇒ 1	1	ok
0 1	⇒ 0	1	wrong
1 0	⇒ 1		
1 1	⇒ 0		

Learning rate $\varepsilon = 0.2$

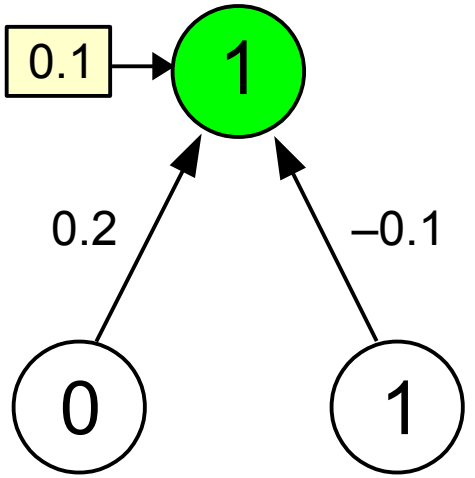


0 × 0.2 + **1** × –0.1 + 0.1 = 0 ≥ 0 output = 1

Compute Output for Pattern 0 1

Input	Target	Output	
0 0	⇒ 1	1	ok
0 1	⇒ 0	1	wrong
1 0	⇒ 1		
1 1	⇒ 0		

Learning rate $\varepsilon = 0.2$

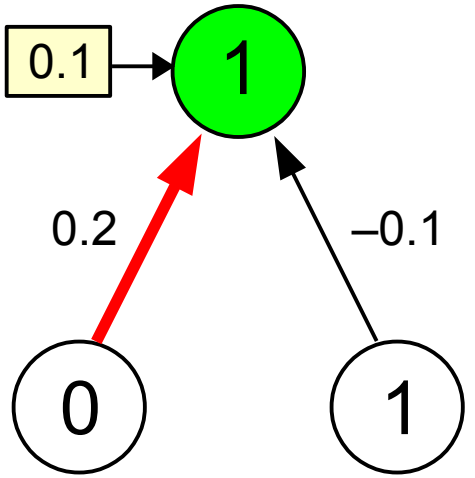


error = *target* – *output* = 0 – 1 = –1

Compute Output for Pattern 0 1

Input	Target	Output	
0 0	⇒ 1	1	ok
0 1	⇒ 0	1	wrong
1 0	⇒ 1		
1 1	⇒ 0		

Learning rate ε = 0.2



error = *target* – *output* = 0 – 1 = –1

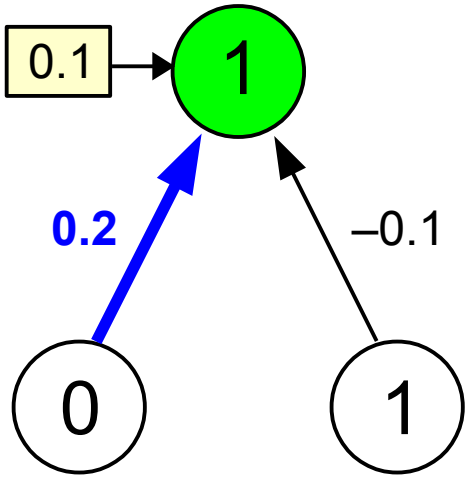
amount to change weight$_1$

= ε × *error* × *input*$_1$ = 0.2 × –1 × 0 = **0**

Compute Output for Pattern 0 1

Input	Target	Output	
0 0	⇒ 1	1	ok
0 1	⇒ 0	1	wrong
1 0	⇒ 1		
1 1	⇒ 0		

Learning rate ε = 0.2



error = *target* – *output* = 0 – 1 = –1

amount to change $weight_1$

= ε × *error* × $input_1$ = 0.2 × –1 × 0 = **0**

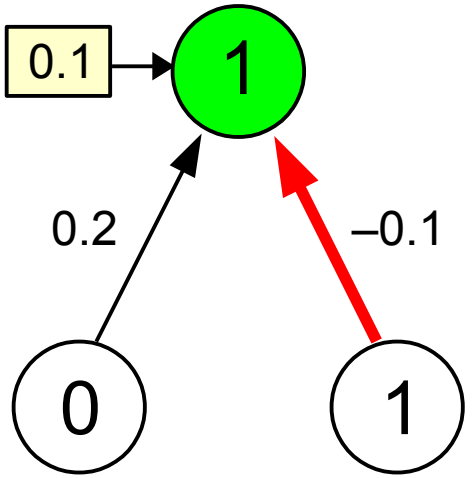
new value of $weight_1$

= 0.2 + 0 = **0.2**

Compute Output for Pattern 0 1

Input	Target	Output	
0 0	⇒ 1	1	ok
0 1	⇒ 0	1	wrong
1 0	⇒ 1		
1 1	⇒ 0		

Learning rate ε = 0.2



error = *target* – *output* = 0 – 1 = –1

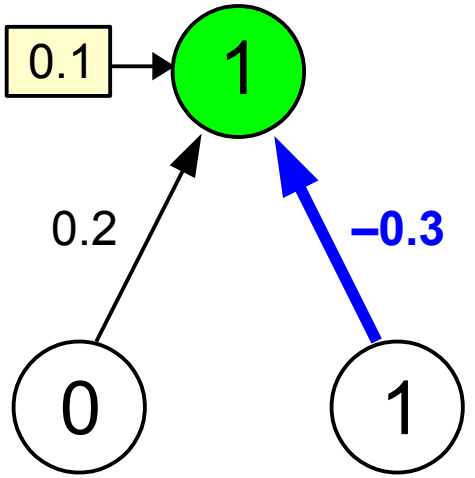
amount to change $weight_2$

= ε × *error* × $input_2$ = 0.2 × –1 × 1 = **–0.2**

Compute Output for Pattern 0 1

Input	Target	Output	
0 0	⇒ 1	1	ok
0 1	⇒ 0	1	wrong
1 0	⇒ 1		
1 1	⇒ 0		

Learning rate ε = 0.2



error = *target* – *output* = 0 – 1 = –1

amount to change $weight_2$

= ε × *error* × $input_2$ = 0.2 × –1 × 1 = **–0.2**

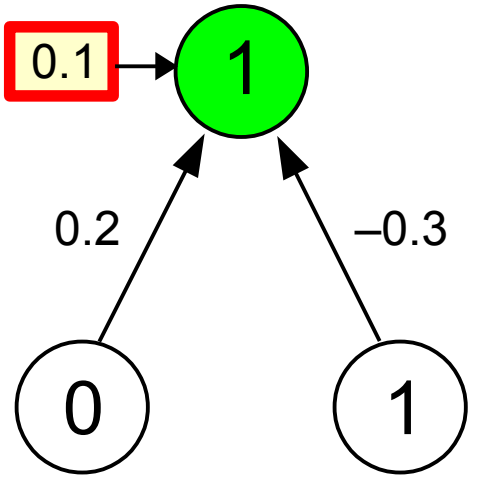
new value of $weight_2$

= –0.1 + –0.2 = **–0.3**

Compute Output for Pattern 0 1

Input	Target	Output	
0 0	⇒ 1	1	ok
0 1	⇒ 0	1	wrong
1 0	⇒ 1		
1 1	⇒ 0		

Learning rate ε = 0.2



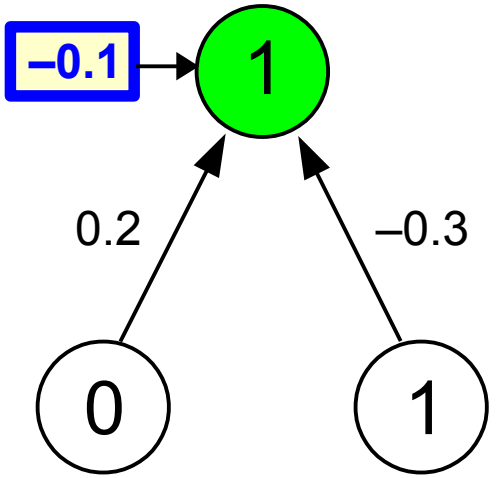
error = *target* – *output* = 0 – 1 = –1

amount to change bias
= ε × *error* = 0.2 × –1 = **–0.2**

Compute Output for Pattern 0 1

Input	Target	Output	
0 0	⇒ 1	1	ok
0 1	⇒ 0	1	wrong
1 0	⇒ 1		
1 1	⇒ 0		

Learning rate ε = 0.2



error = *target* – *output* = 0 – 1 = –1

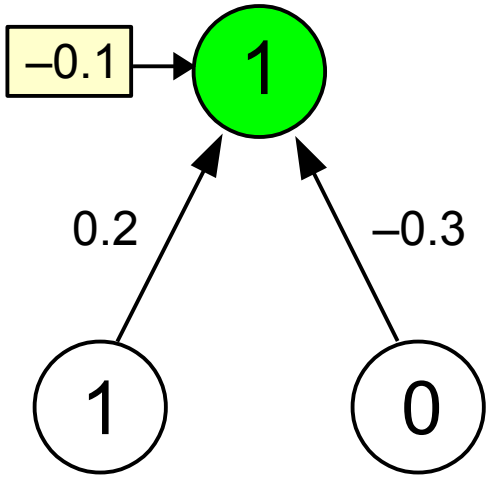
amount to change bias
= ε × *error* = 0.2 × –1 = **–0.2**

new value of bias
= 0.1 + –0.2 = **–0.1**

Compute Output for Pattern 1 0

Input	Target	Output	
0 0	⇒ 1	1	ok
0 1	⇒ 0	1	wrong
1 0	⇒ 1	1	ok
1 1	⇒ 0		

Learning rate ε = 0.2

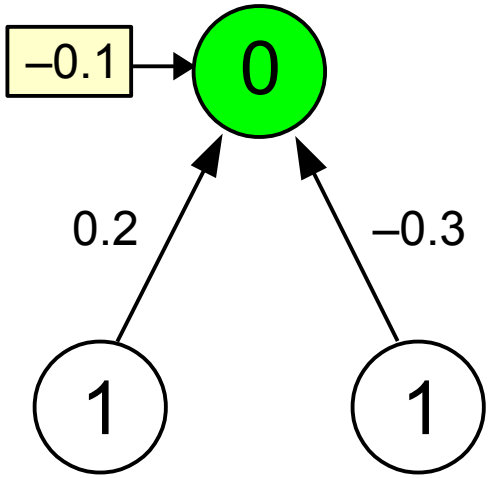


1 × 0.2 + **0** × –0.3 + –0.1 = 0.1 ≥ 0 output = 1

Compute Output for Pattern 1 1

Input	Target	Output	
0 0	⇒ 1	1	ok
0 1	⇒ 0	1	wrong
1 0	⇒ 1	1	ok
1 1	⇒ 0	0	ok

Learning rate ε = 0.2

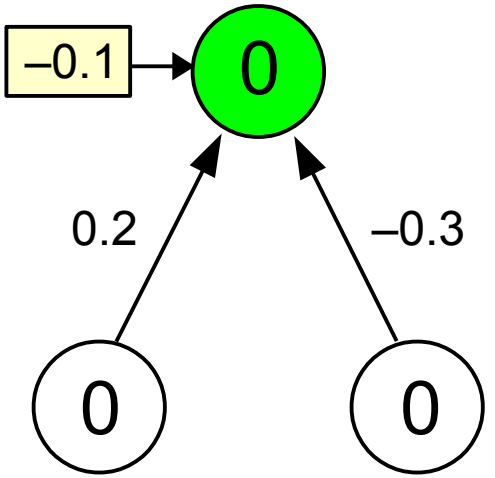


1 × 0.2 + **1** × –0.3 + –0.1 = –0.2 < 0 output = 0

Compute Output for Pattern 0 0

Input	Target	Output	
0 0	⇒ 1	0	wrong
0 1	⇒ 0		
1 0	⇒ 1		
1 1	⇒ 0		

Learning rate ε = 0.2

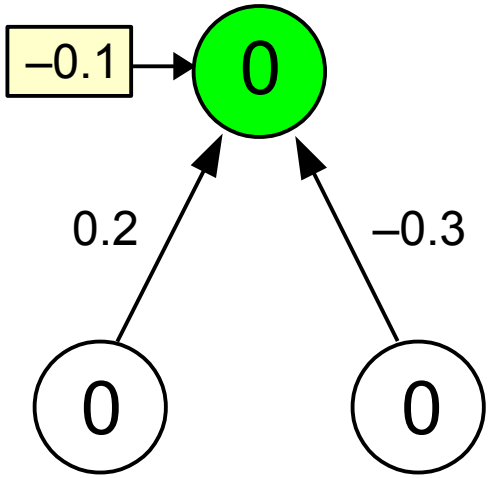


0 × 0.2 + **0** × –0.3 + –0.1 = –0.1 < 0 output = 0

Compute Output for Pattern 0 0

Input	Target	Output	
0 0	⇒ 1	0	wrong
0 1	⇒ 0		
1 0	⇒ 1		
1 1	⇒ 0		

Learning rate ε = 0.2

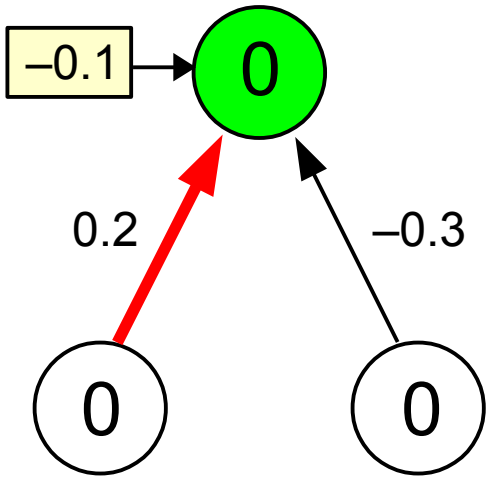


error = *target* – *output* = 1 – 0 = 1

Compute Output for Pattern 0 0

Input	Target	Output	
0 0	⇒ 1	0	wrong
0 1	⇒ 0		
1 0	⇒ 1		
1 1	⇒ 0		

Learning rate ε = 0.2



error = *target* – *output* = 1 – 0 = 1

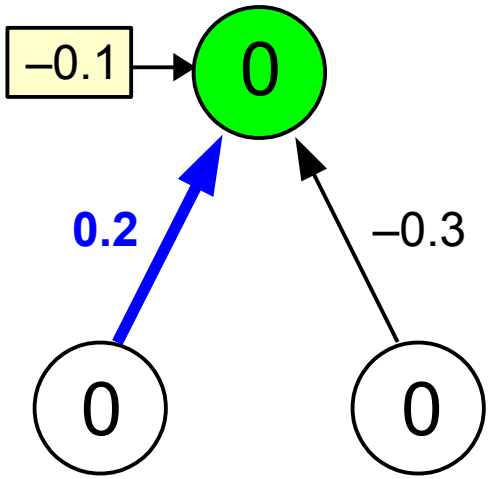
amount to change weight$_1$

= ε × *error* × *input*$_1$ = 0.2 × 1 × 0 = **0**

Compute Output for Pattern 0 0

Input	Target	Output	
0 0	⇒ 1	0	wrong
0 1	⇒ 0		
1 0	⇒ 1		
1 1	⇒ 0		

Learning rate ε = 0.2



error = *target* – *output* = 1 – 0 = 1

amount to change $weight_1$

= ε × *error* × $input_1$ = 0.2 × 1 × 0 = **0**

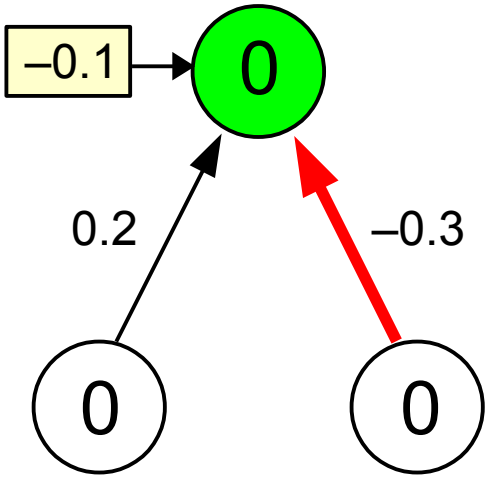
new value of $weight_1$

= 0.2 + 0 = **0.2**

Compute Output for Pattern 0 0

Input	Target	Output	
0 0	⇒ 1	0	wrong
0 1	⇒ 0		
1 0	⇒ 1		
1 1	⇒ 0		

Learning rate ε = 0.2



error = *target* – *output* = 1 – 0 = 1

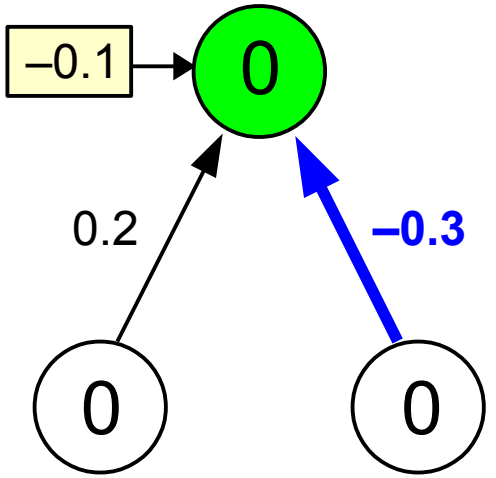
amount to change $weight_2$

= ε × *error* × $input_2$ = 0.2 × 1 × 0 = **0**

Compute Output for Pattern 0 0

Input	Target	Output	
0 0	⇒ 1	0	wrong
0 1	⇒ 0		
1 0	⇒ 1		
1 1	⇒ 0		

Learning rate ε = 0.2



error = *target* – *output* = 1 – 0 = 1

amount to change $weight_2$

= ε × *error* × $input_2$ = 0.2 × 1 × 0 = **0**

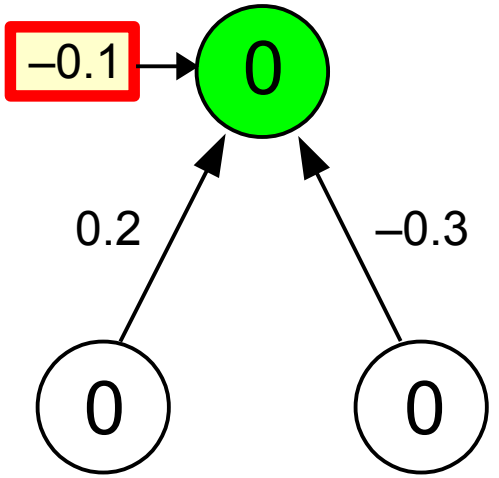
new value of $weight_2$

= –0.3 + 0 = **–0.3**

Compute Output for Pattern 0 0

Input	Target	Output	
0 0	⇒ 1	0	wrong
0 1	⇒ 0		
1 0	⇒ 1		
1 1	⇒ 0		

Learning rate ε = 0.2



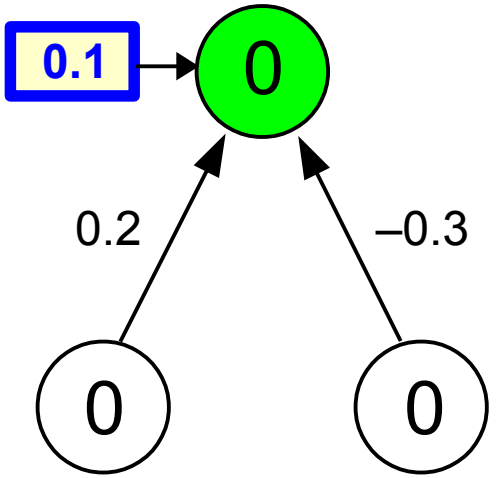
error = *target* – *output* = 1 – 0 = 1

amount to change bias
= ε × *error* = 0.2 × 1 = **0.2**

Compute Output for Pattern 0 0

Input	Target	Output	
0 0	⇒ 1	0	wrong
0 1	⇒ 0		
1 0	⇒ 1		
1 1	⇒ 0		

Learning rate ε = 0.2



error = *target* – *output* = 1 – 0 = 1

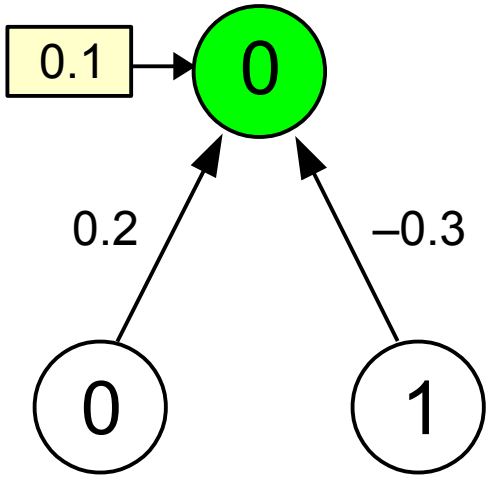
amount to change bias
= ε × *error* = 0.2 × 1 = **0.2**

new value of bias
= –0.1 + 0.2 = **0.1**

Compute Output for Pattern 0 1

Input	Target	Output	
0 0	⇒ 1	0	wrong
0 1	⇒ 0	0	ok
1 0	⇒ 1		
1 1	⇒ 0		

Learning rate ε = 0.2

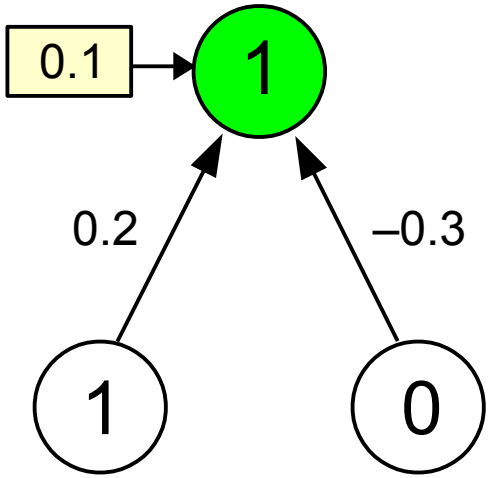


0 × 0.2 + **1** × –0.3 + 0.1 = –0.2 < 0 output = 0

Compute Output for Pattern 1 0

Input	Target	Output	
0 0	⇒ 1	0	wrong
0 1	⇒ 0	0	ok
1 0	⇒ 1	1	ok
1 1	⇒ 0		

Learning rate ε = 0.2

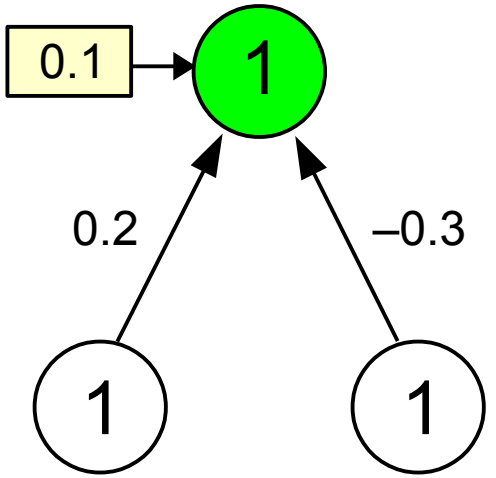


1 × 0.2 + **0** × –0.3 + 0.1 = 0.3 ≥ 0 output = 1

Compute Output for Pattern 1 1

Input	Target	Output	
0 0	⇒ 1	0	wrong
0 1	⇒ 0	0	ok
1 0	⇒ 1	1	ok
1 1	⇒ 0	1	wrong

Learning rate ε = 0.2

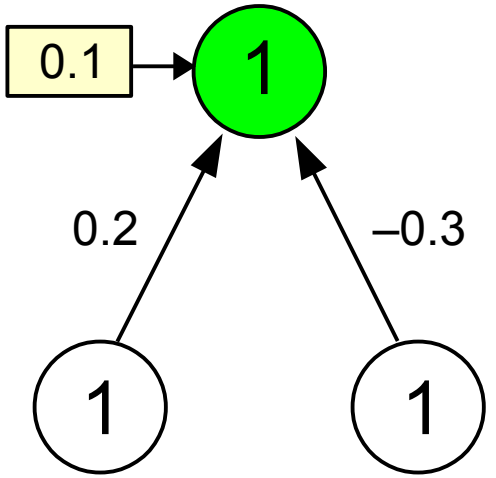


1 × 0.2 + **1** × –0.3 + 0.1 = 0 ≥ 0 output = 1

Compute Output for Pattern 1 1

Input	Target	Output	
0 0	⇒ 1	0	wrong
0 1	⇒ 0	0	ok
1 0	⇒ 1	1	ok
1 1	⇒ 0	1	wrong

Learning rate ε = 0.2

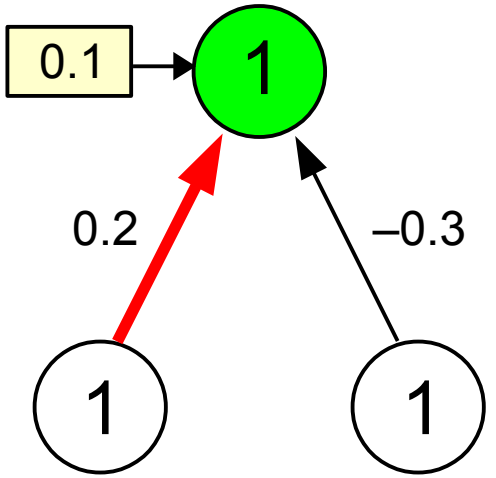


error = *target* – *output* = 0 – 1 = –1

Compute Output for Pattern 1 1

Input	Target	Output	
0 0	⇒ 1	0	wrong
0 1	⇒ 0	0	ok
1 0	⇒ 1	1	ok
1 1	⇒ 0	1	wrong

Learning rate $\varepsilon = 0.2$



$error = target - output = 0 - 1 = -1$

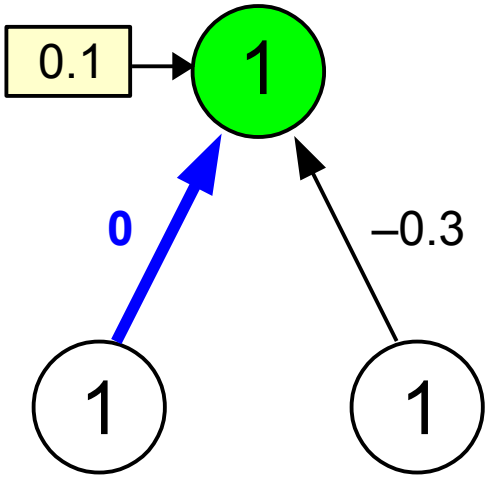
amount to change $weight_1$

$= \varepsilon \times error \times input_1 = 0.2 \times -1 \times 1 = \mathbf{-0.2}$

Compute Output for Pattern 1 1

Input	Target	Output	
0 0	⇒ 1	0	wrong
0 1	⇒ 0	0	ok
1 0	⇒ 1	1	ok
1 1	⇒ 0	1	wrong

Learning rate ε = 0.2



error = *target* – *output* = 0 – 1 = –1

amount to change $weight_1$

= ε × *error* × $input_1$ = 0.2 × –1 × 1 = **–0.2**

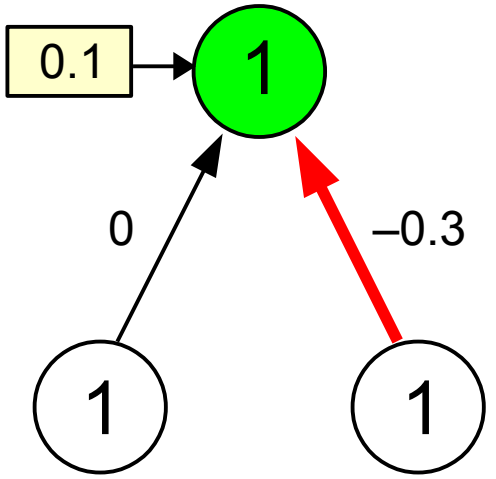
new value of $weight_1$

= 0.2 + –0.2 = **0**

Compute Output for Pattern 1 1

Input	Target	Output	
0 0	⇒ 1	0	wrong
0 1	⇒ 0	0	ok
1 0	⇒ 1	1	ok
1 1	⇒ 0	1	wrong

Learning rate $\varepsilon = 0.2$



error = *target* – *output* = 0 – 1 = –1

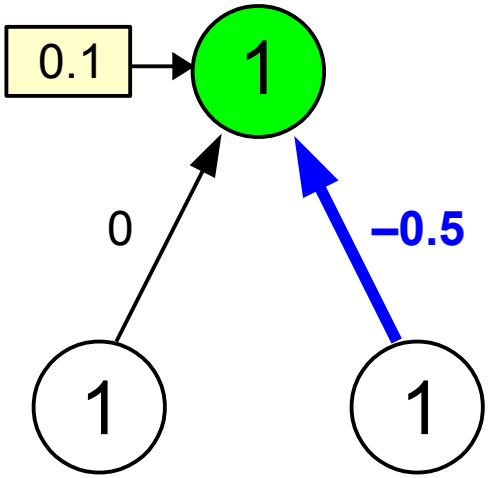
amount to change $weight_2$

= $\varepsilon \times error \times input_2$ = 0.2 × –1 × 1 = **–0.2**

Compute Output for Pattern 1 1

Input	Target	Output	
0 0	⇒ 1	0	wrong
0 1	⇒ 0	0	ok
1 0	⇒ 1	1	ok
1 1	⇒ 0	1	wrong

Learning rate ε = 0.2



error = *target* – *output* = 0 – 1 = –1

amount to change $weight_2$

= ε × *error* × $input_2$ = 0.2 × –1 × 1 = **–0.2**

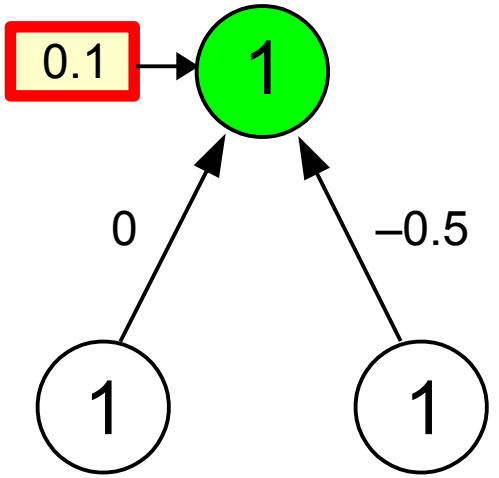
new value of $weight_2$

= –0.3 + –0.2 = **–0.5**

Compute Output for Pattern 1 1

Input	Target	Output	
0 0	⇒ 1	0	wrong
0 1	⇒ 0	0	ok
1 0	⇒ 1	1	ok
1 1	⇒ 0	1	wrong

Learning rate ε = 0.2



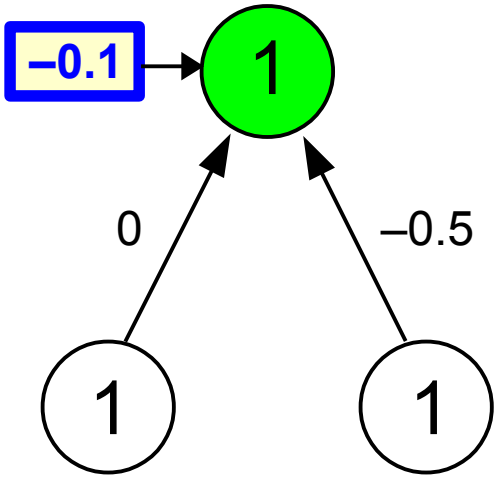
error = *target* – *output* = 0 – 1 = –1

amount to change bias
= ε × *error* = 0.2 × –1 = **–0.2**

Compute Output for Pattern 1 1

Input	Target	Output	
0 0	⇒ 1	0	wrong
0 1	⇒ 0	0	ok
1 0	⇒ 1	1	ok
1 1	⇒ 0	1	wrong

Learning rate ε = 0.2



error = *target* – *output* = 0 – 1 = –1

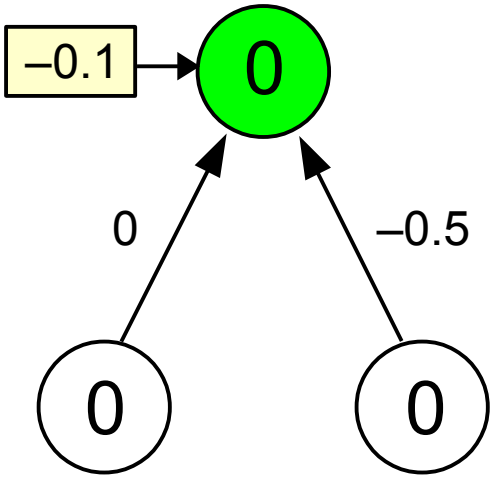
amount to change bias
= ε × *error* = 0.2 × –1 = **–0.2**

new value of bias
= 0.1 + –0.2 = **–0.1**

Compute Output for Pattern 0 0

Input	Target	Output	
0 0	⇒ 1	0	wrong
0 1	⇒ 0		
1 0	⇒ 1		
1 1	⇒ 0		

Learning rate ε = 0.2

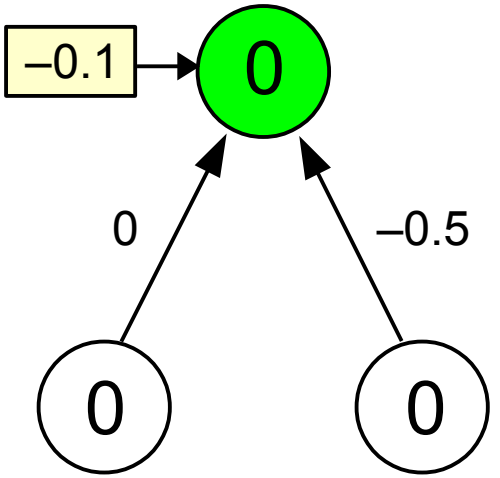


0 × 0 + **0** × –0.5 + –0.1 = –0.1 < 0 output = 0

Compute Output for Pattern 0 0

Input	Target	Output	
0 0	⇒ 1	0	wrong
0 1	⇒ 0		
1 0	⇒ 1		
1 1	⇒ 0		

Learning rate ε = 0.2

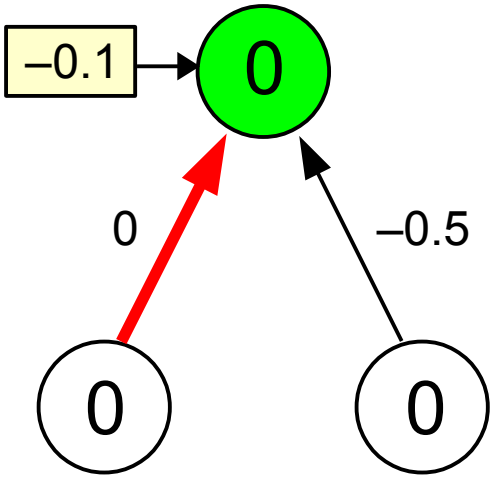


error = *target* – *output* = 1 – 0 = 1

Compute Output for Pattern 0 0

Input	Target	Output	
0 0	⇒ 1	0	wrong
0 1	⇒ 0		
1 0	⇒ 1		
1 1	⇒ 0		

Learning rate ε = 0.2



error = *target* – *output* = 1 – 0 = 1

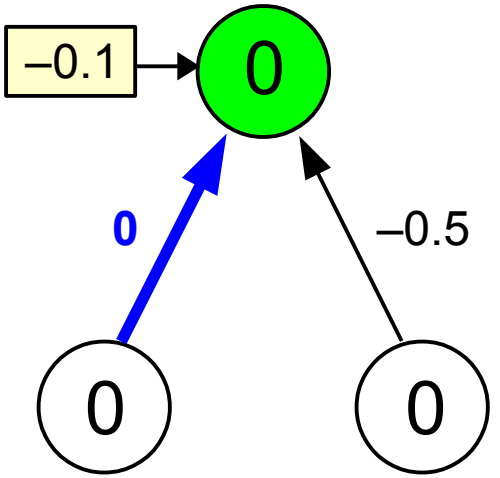
amount to change $weight_1$

= ε × *error* × $input_1$ = 0.2 × 1 × 0 = **0**

Compute Output for Pattern 0 0

Input	Target	Output	
0 0	⇒ 1	0	wrong
0 1	⇒ 0		
1 0	⇒ 1		
1 1	⇒ 0		

Learning rate ε = 0.2



error = *target* – *output* = 1 – 0 = 1

amount to change $weight_1$

= ε × *error* × $input_1$ = 0.2 × 1 × 0 = **0**

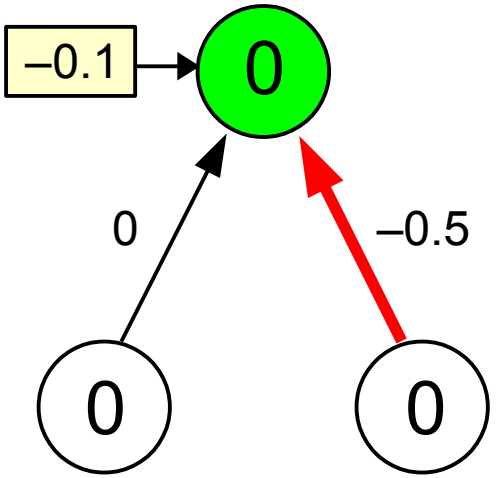
new value of $weight_1$

= 0 + 0 = **0**

Compute Output for Pattern 0 0

Input	Target	Output	
0 0	⇒ 1	0	wrong
0 1	⇒ 0		
1 0	⇒ 1		
1 1	⇒ 0		

Learning rate ε = 0.2



error = *target* – *output* = 1 – 0 = 1

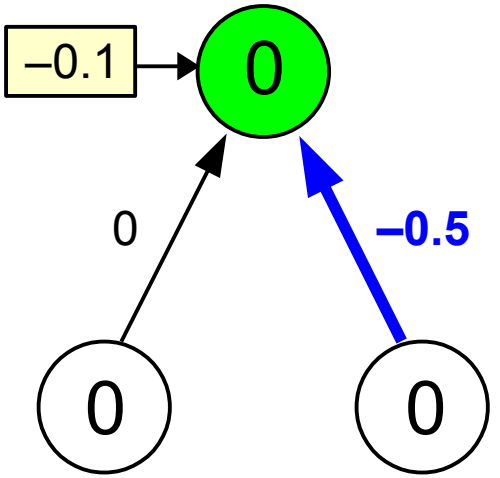
amount to change $weight_2$

= ε × *error* × $input_2$ = 0.2 × 1 × 0 = **0**

Compute Output for Pattern 0 0

Input	Target	Output	
0 0	⇒ 1	0	wrong
0 1	⇒ 0		
1 0	⇒ 1		
1 1	⇒ 0		

Learning rate ε = 0.2



error = *target* – *output* = 1 – 0 = 1

amount to change $weight_2$

= ε × *error* × $input_2$ = 0.2 × 1 × 0 = **0**

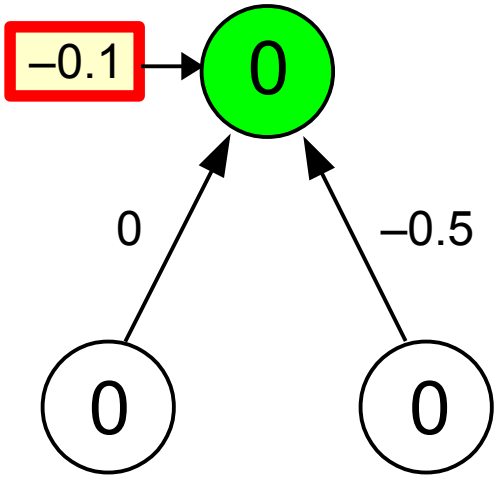
new value of $weight_2$

= –0.5 + 0 = **–0.5**

Compute Output for Pattern 0 0

Input	Target	Output	
0 0	⇒ 1	0	wrong
0 1	⇒ 0		
1 0	⇒ 1		
1 1	⇒ 0		

Learning rate ε = 0.2



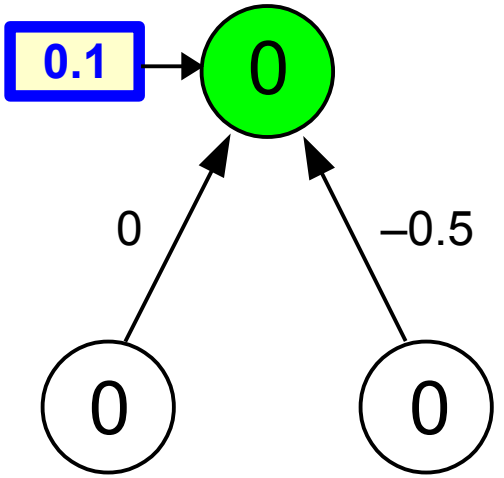
error = *target* – *output* = 1 – 0 = 1

amount to change bias
= ε × *error* = 0.2 × 1 = **0.2**

Compute Output for Pattern 0 0

Input	Target	Output	
0 0	⇒ 1	0	wrong
0 1	⇒ 0		
1 0	⇒ 1		
1 1	⇒ 0		

Learning rate ε = 0.2



error = *target* – *output* = 1 – 0 = 1

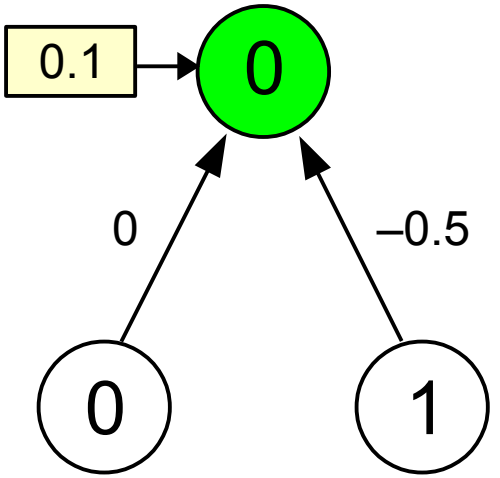
amount to change bias
= ε × *error* = 0.2 × 1 = **0.2**

new value of bias
= –0.1 + 0.2 = **0.1**

Compute Output for Pattern 0 1

Input	Target	Output	
0 0	⇒ 1	0	wrong
0 1	⇒ 0	0	ok
1 0	⇒ 1		
1 1	⇒ 0		

Learning rate ε = 0.2

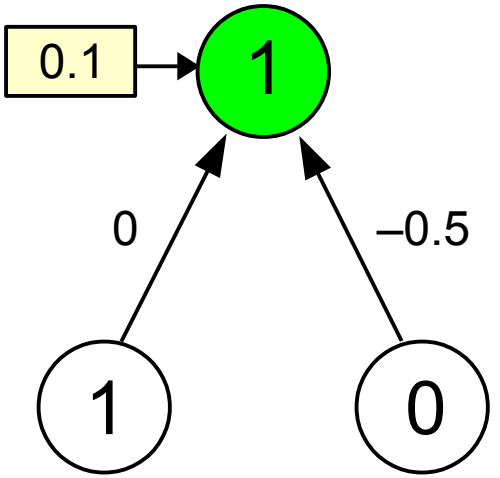


0 × 0 + **1** × –0.5 + 0.1 = –0.4 < 0 output = 0

Compute Output for Pattern 1 0

Input	Target	Output	
0 0	⇒ 1	0	wrong
0 1	⇒ 0	0	ok
1 0	⇒ 1	1	ok
1 1	⇒ 0		

Learning rate $\varepsilon = 0.2$

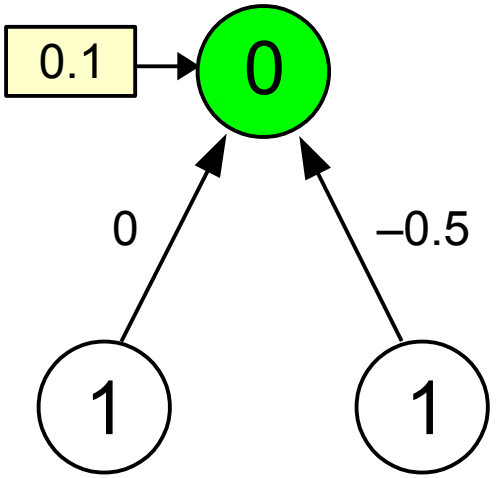


$$\mathbf{1} \times 0 + \mathbf{0} \times -0.5 + 0.1 = 0.1 \geq 0 \qquad \text{output} = 1$$

Compute Output for Pattern 1 1

Input	Target	Output	
0 0	⇒ 1	0	wrong
0 1	⇒ 0	0	ok
1 0	⇒ 1	1	ok
1 1	⇒ 0	0	ok

Learning rate ε = 0.2

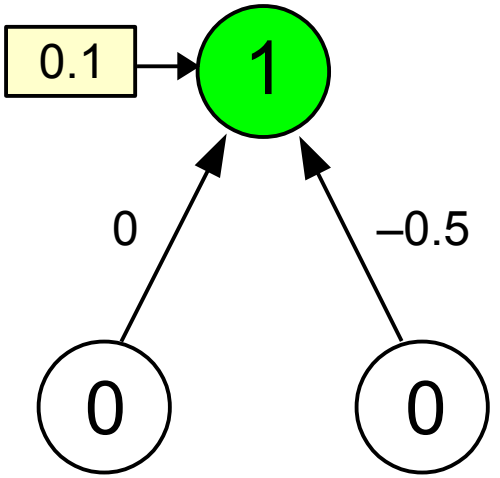


1 × 0 + **1** × –0.5 + 0.1 = –0.4 < 0 output = 0

Compute Output for Pattern 0 0

Input	Target	Output	
0 0	⇒ 1	1	ok
0 1	⇒ 0		
1 0	⇒ 1		
1 1	⇒ 0		

Learning rate ε = 0.2

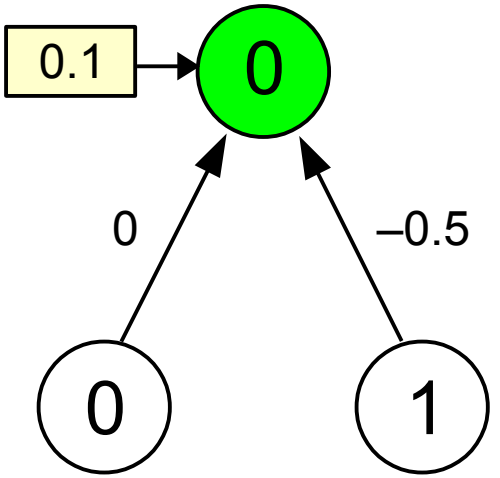


0 × 0 + **0** × –0.5 + 0.1 = 0.1 ≥ 0 output = 1

Compute Output for Pattern 0 1

Input	Target	Output	
0 0	⇒ 1	1	ok
0 1	⇒ 0	0	ok
1 0	⇒ 1		
1 1	⇒ 0		

Learning rate ε = 0.2

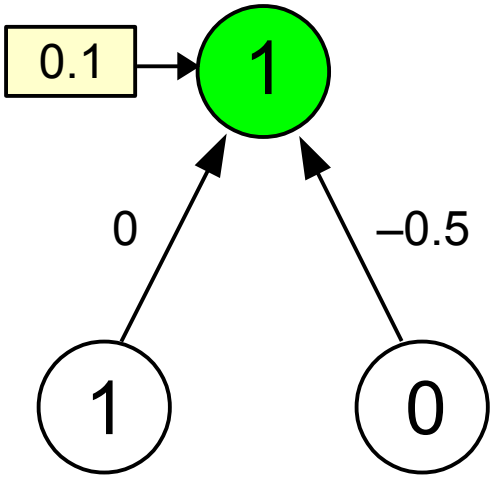


0 × 0 + **1** × –0.5 + 0.1 = –0.4 < 0 output = 0

Compute Output for Pattern 1 0

Input	Target	Output	
0 0	⇒ 1	1	ok
0 1	⇒ 0	0	ok
1 0	⇒ 1	1	ok
1 1	⇒ 0		

Learning rate ε = 0.2

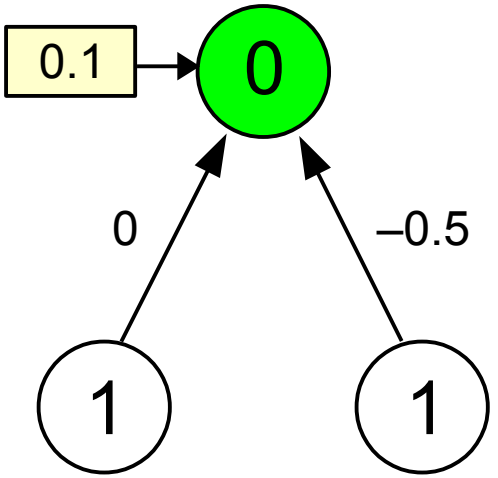


1 × 0 + **0** × –0.5 + 0.1 = 0.1 ≥ 0 output = 1

Compute Output for Pattern 1 1

Input	Target	Output	
0 0	⇒ 1	1	ok
0 1	⇒ 0	0	ok
1 0	⇒ 1	1	ok
1 1	⇒ 0	0	ok

Learning rate ε = 0.2

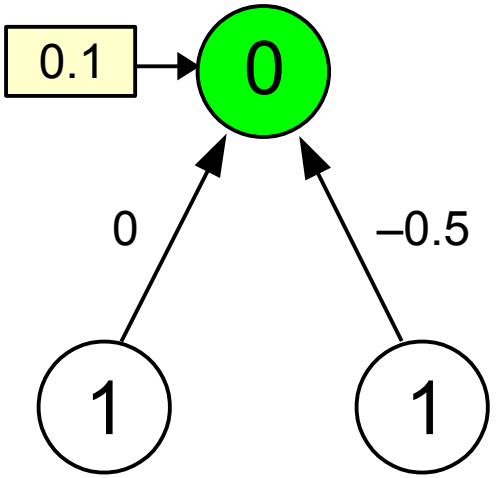


1 × 0 + **1** × –0.5 + 0.1 = –0.4 < 0 output = 0

Compute Output for Pattern 1 1

Input	Target	Output	
0 0	⇒ 1	1	ok
0 1	⇒ 0	0	ok
1 0	⇒ 1	1	ok
1 1	⇒ 0	0	ok

Learning rate ε = 0.2



1 × 0 + **1** × –0.5 + 0.1 = –0.4 < 0 output = 0