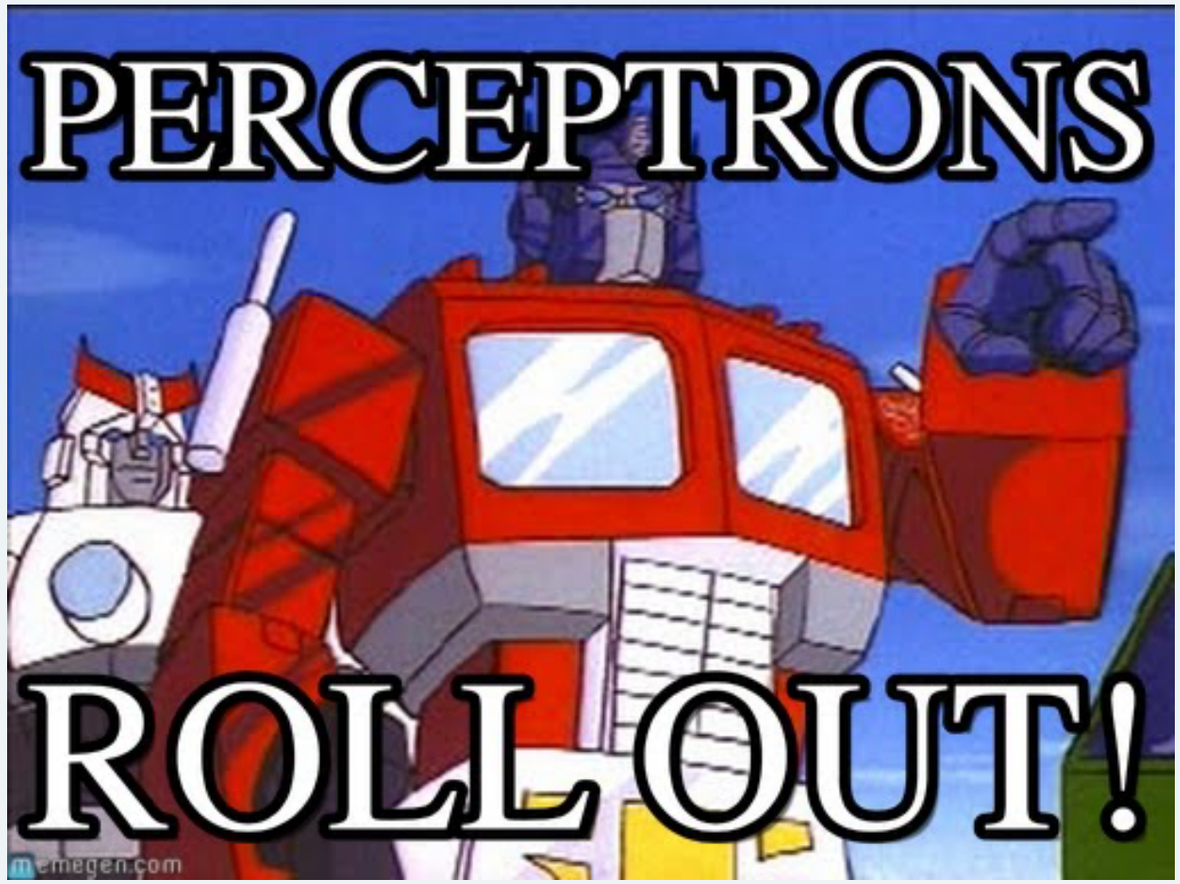
COGS300

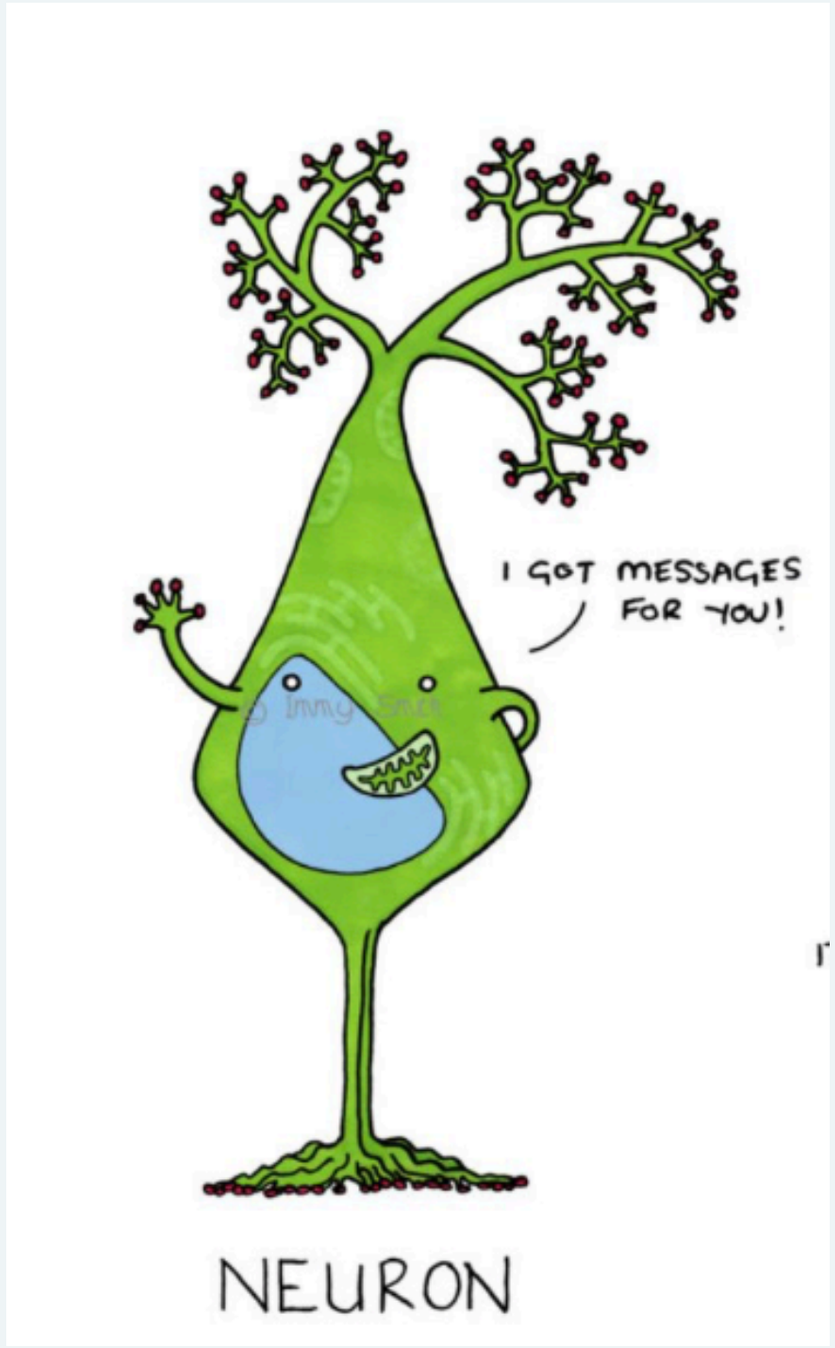
The perceptron



Instructor: Márton Sóskuthy
marton.soskuthy@ubc.ca

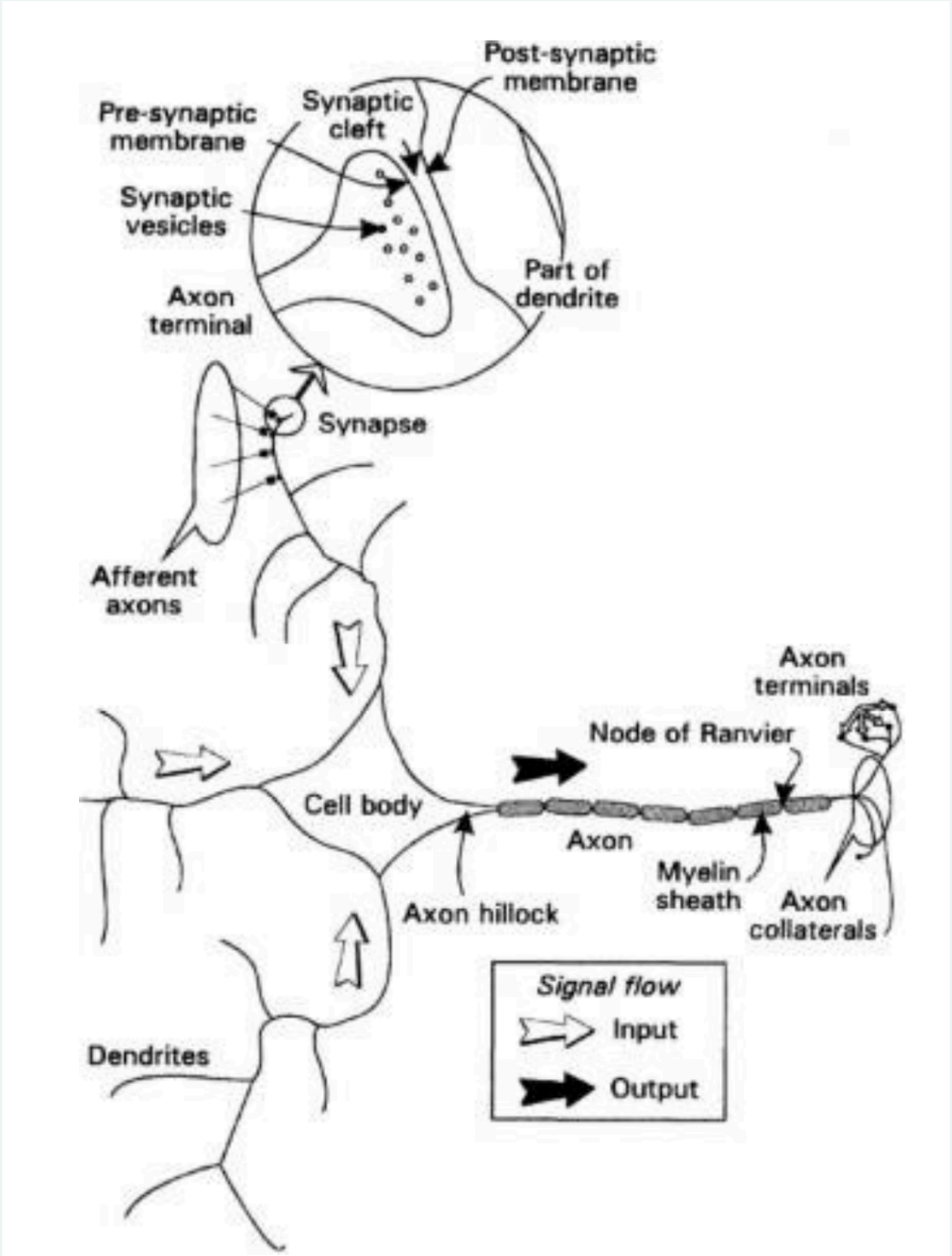
TAs: Daichi Furukawa · Victoria Lim · Amy Wang
cogs.300@ubc.ca

Your new best friend: the neuron

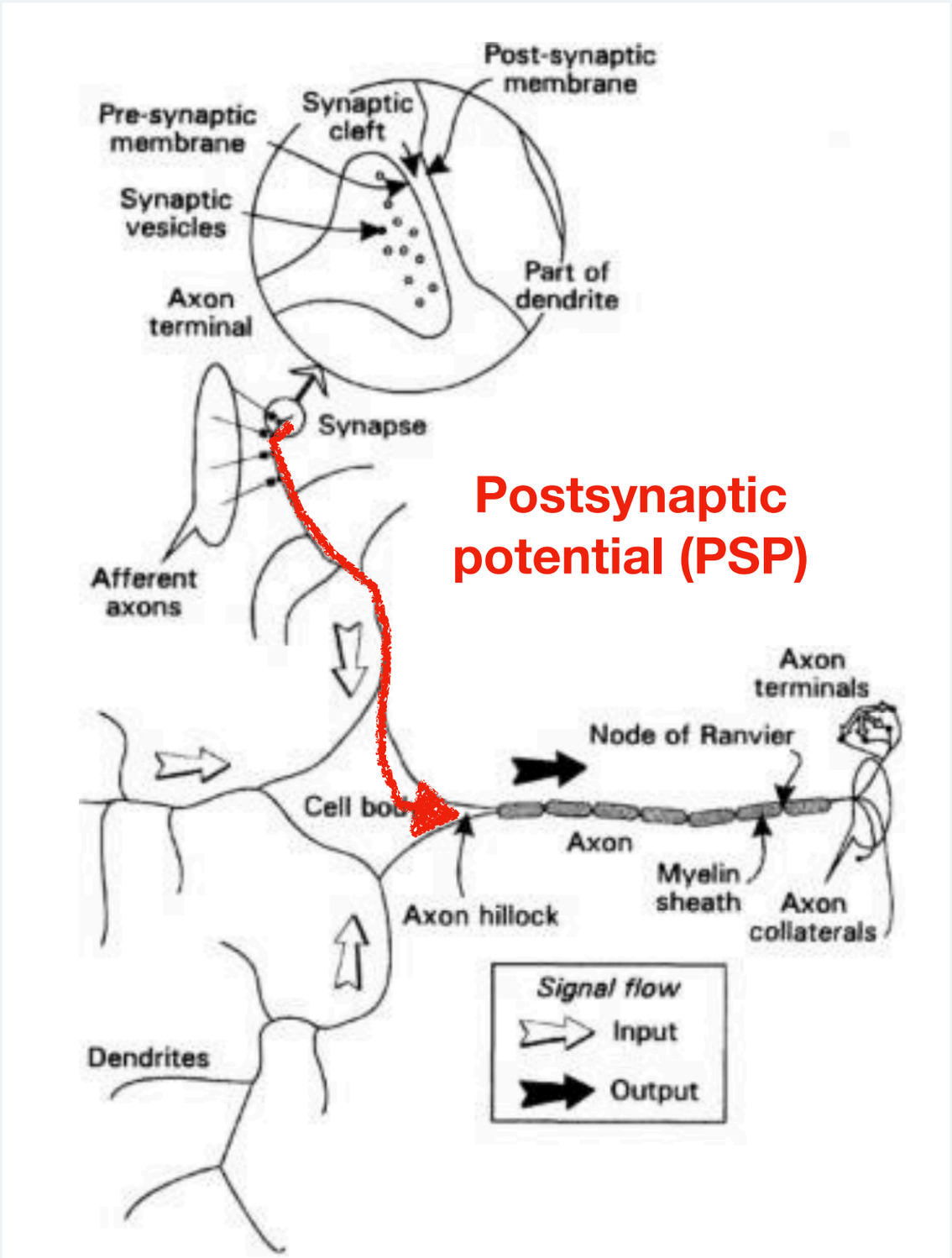


@Cartoon_Neuron

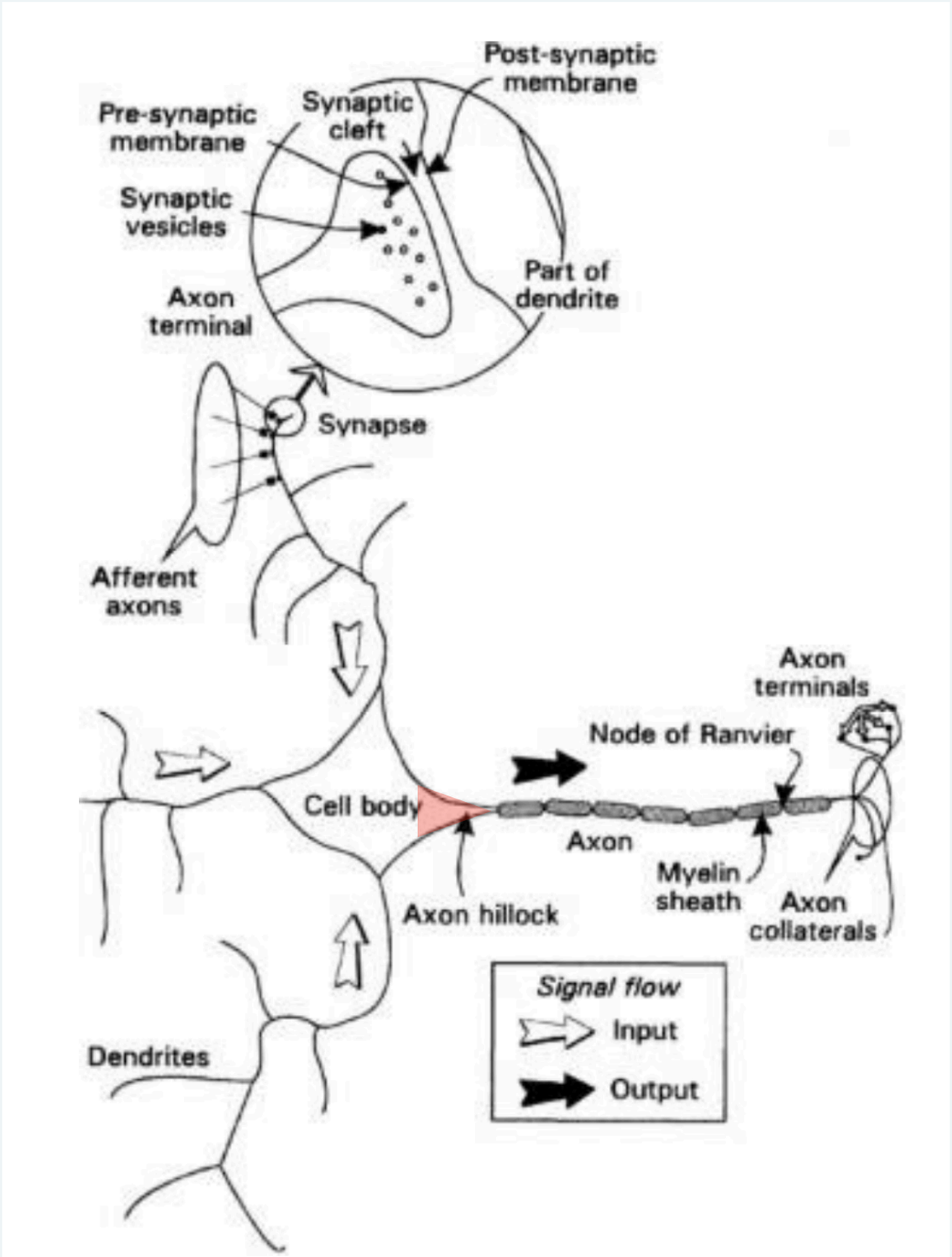
Your new best friend: the neuron



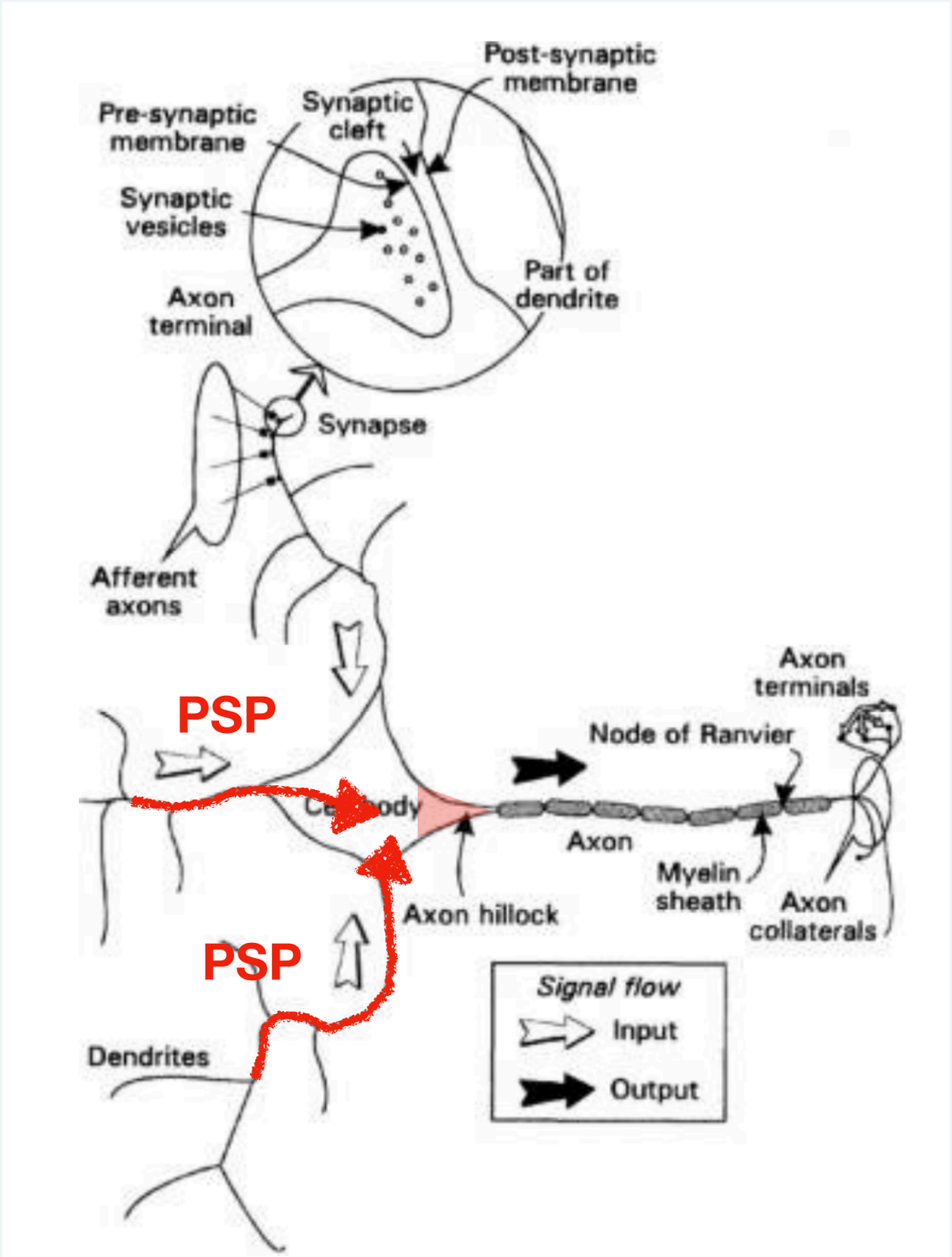
Your new best friend: the neuron



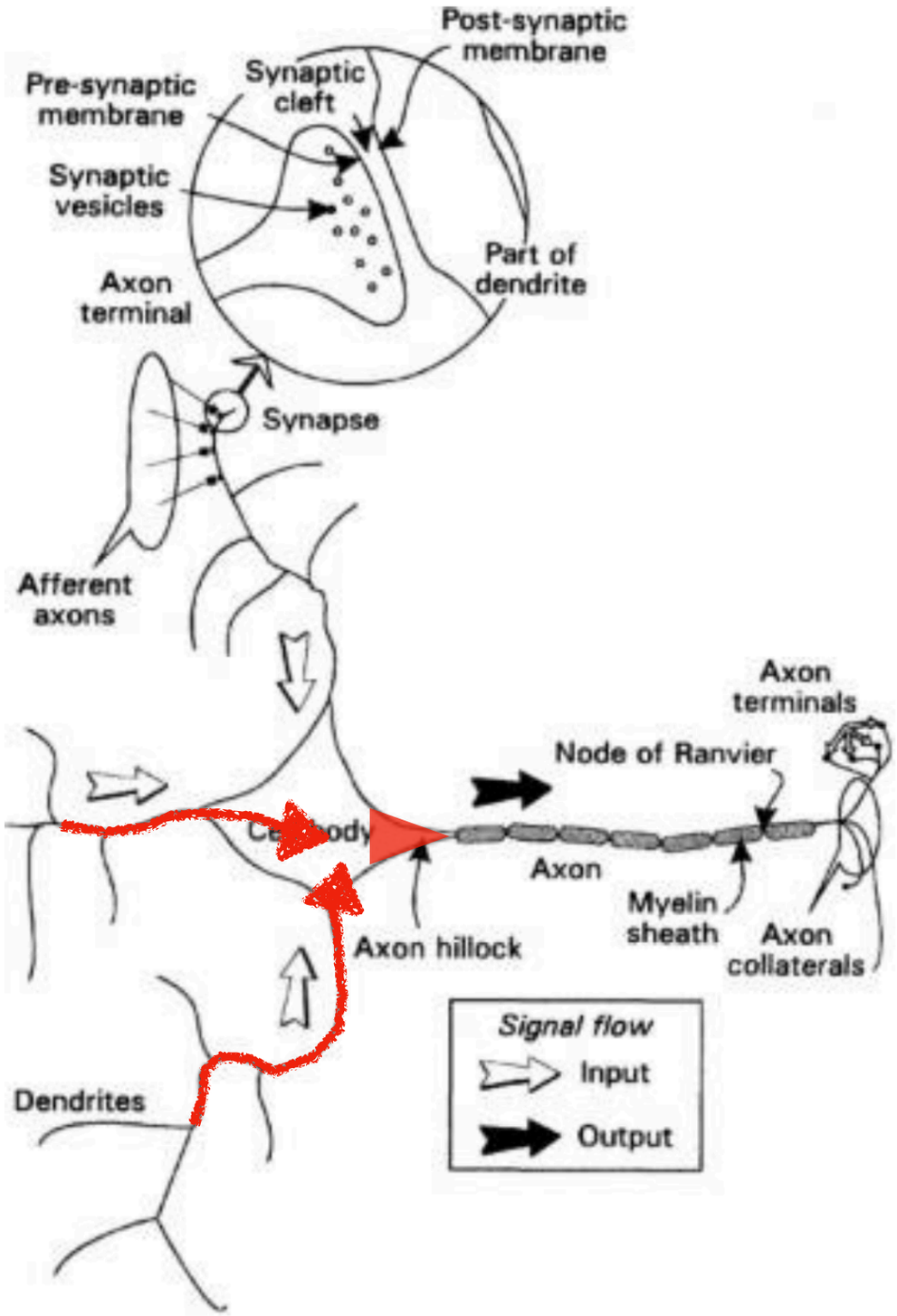
Your new best friend: the neuron



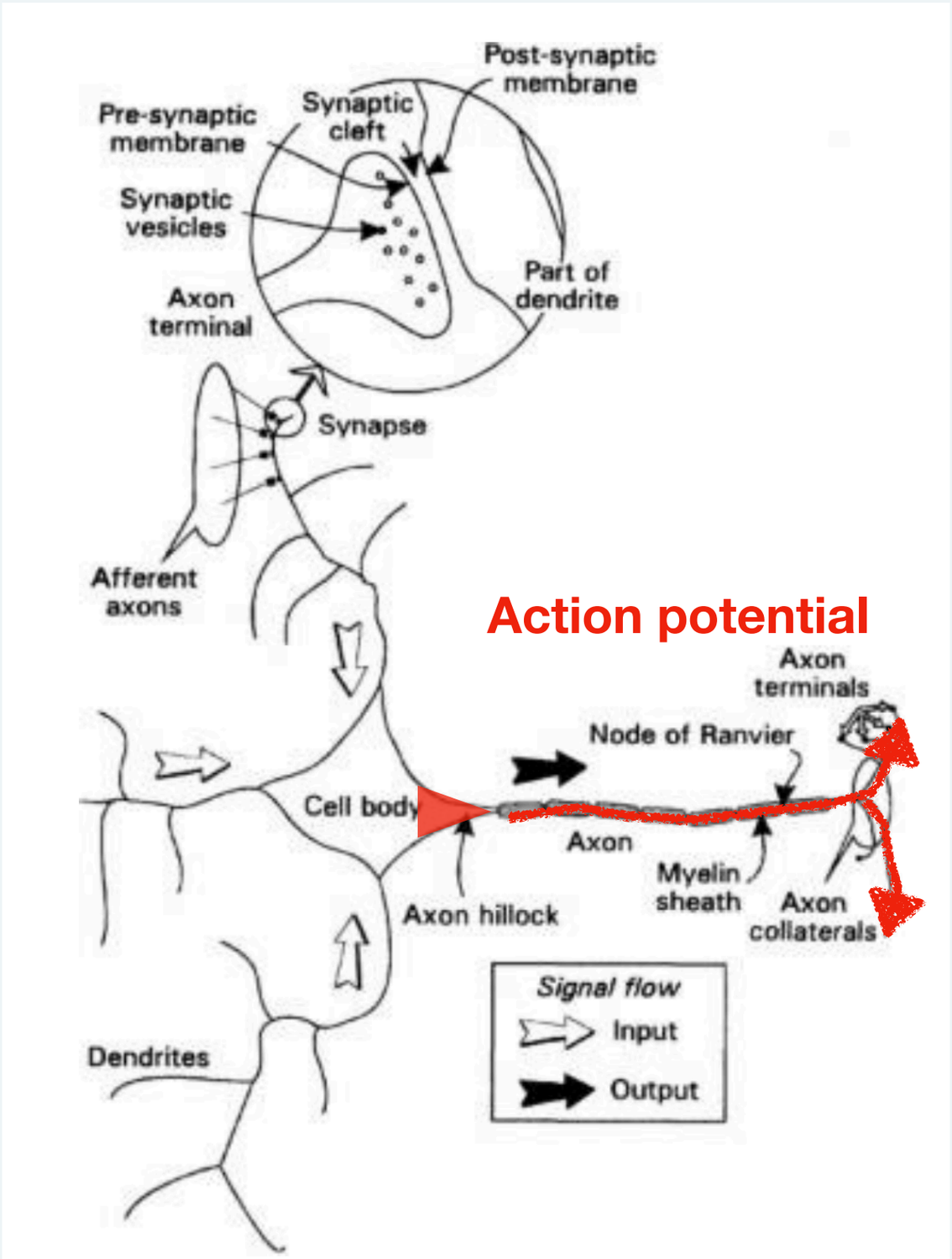
Your new best friend: the neuron



Your new best friend: the neuron

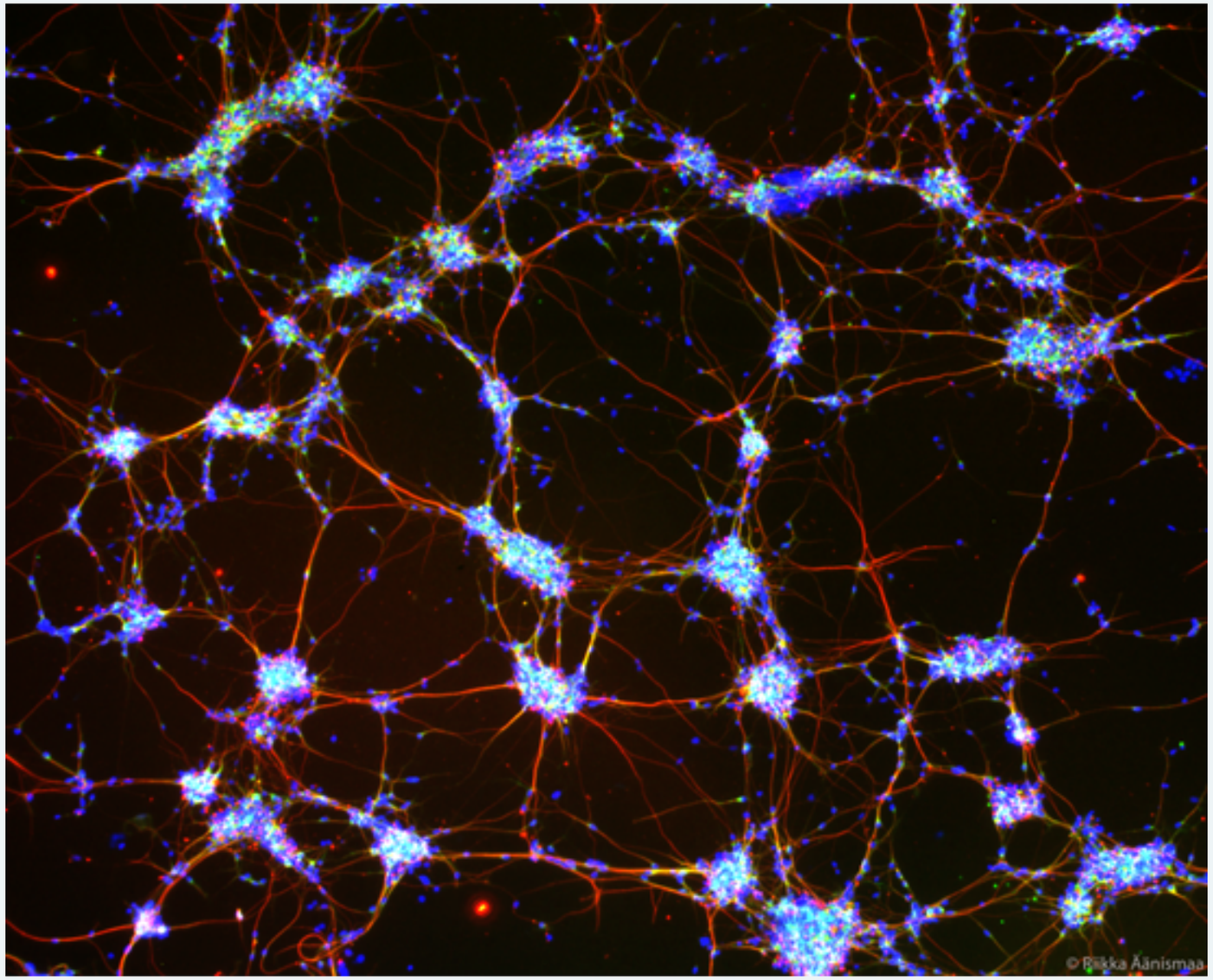


Your new best friend: the neuron

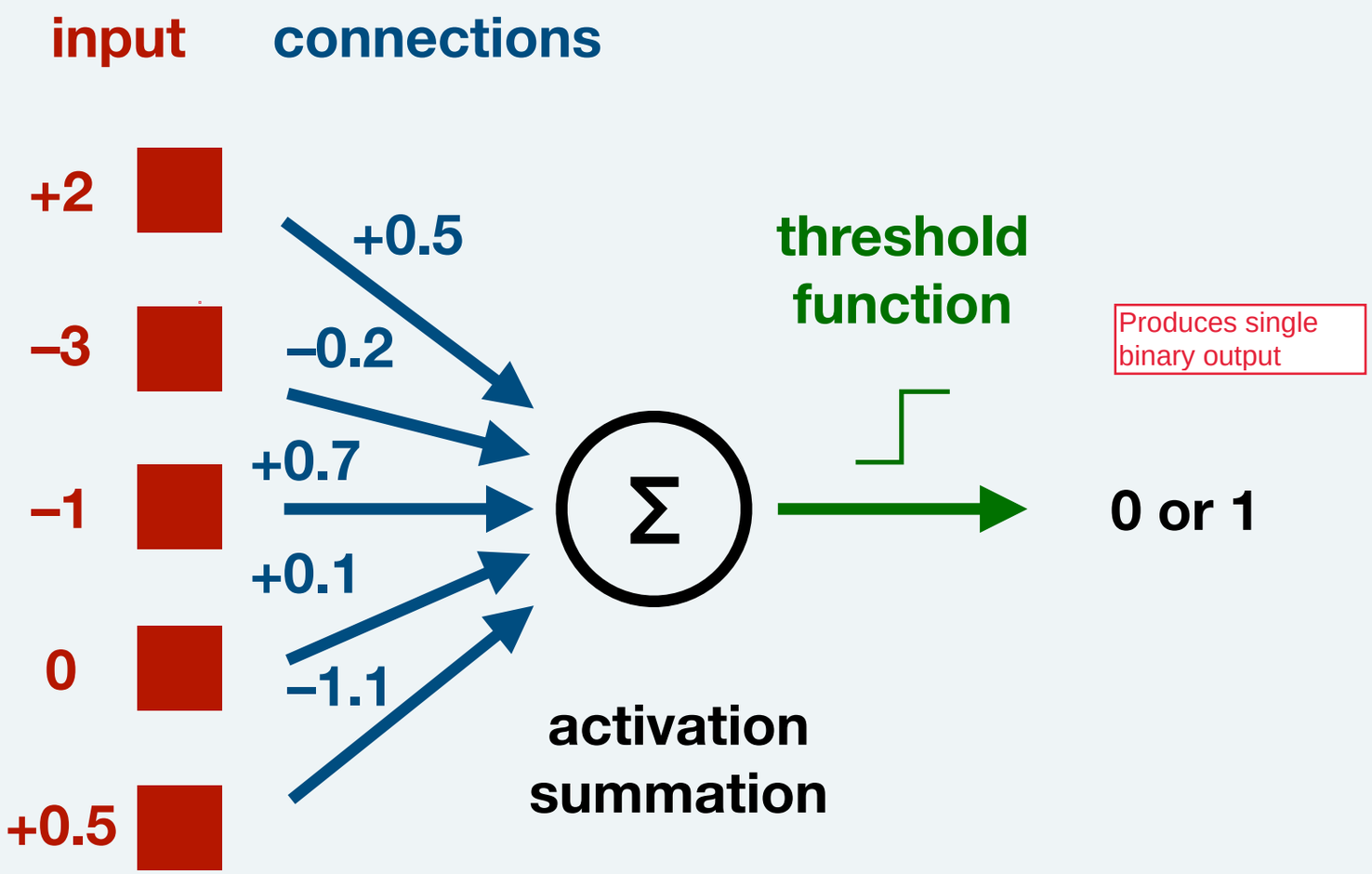


Your new best friend: the neuron

NNs (bio or artificial) = self organizing, locally organized

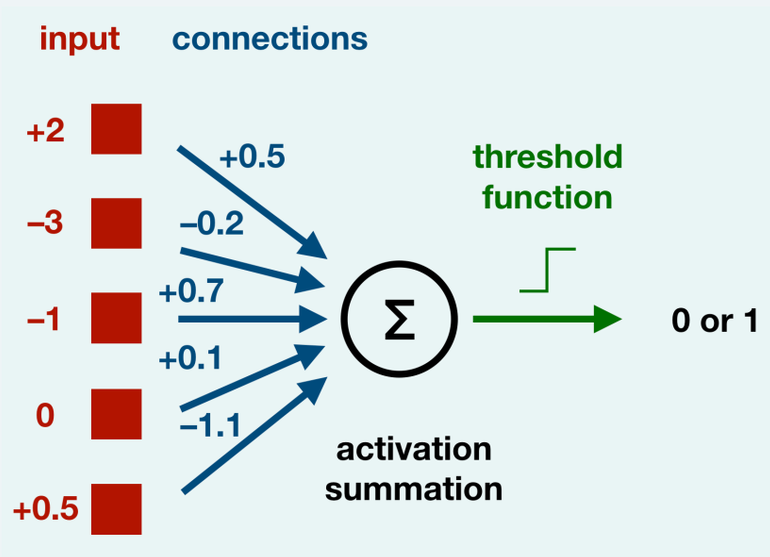


Perceptron: an artificial neuron



Perceptron: an artificial neuron

input		connections (weight)
+2	x	+0.5
–3	x	–0.2
–1	x	+0.7
0	x	+0.1
+0.5	x	–1.1



Perceptron: an artificial neuron

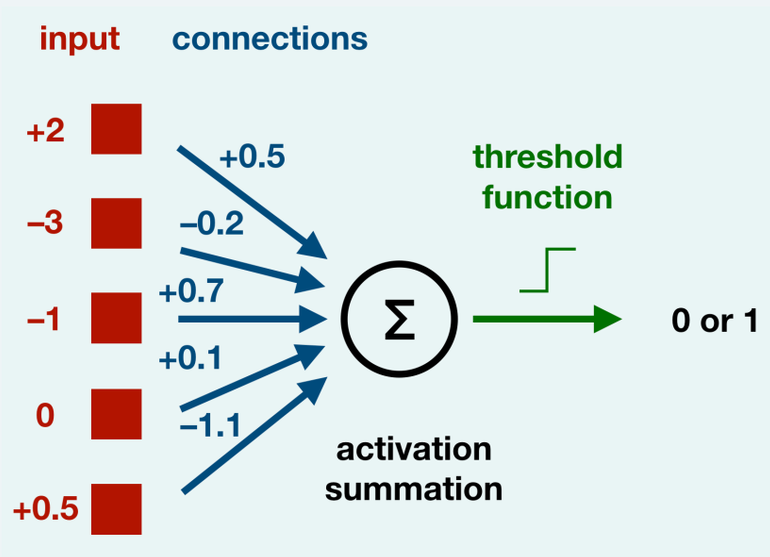
1

+0.6

–0.7

0

–0.55



Perceptron: an artificial neuron

summation (in processing unit)

1

+

+0.6

+

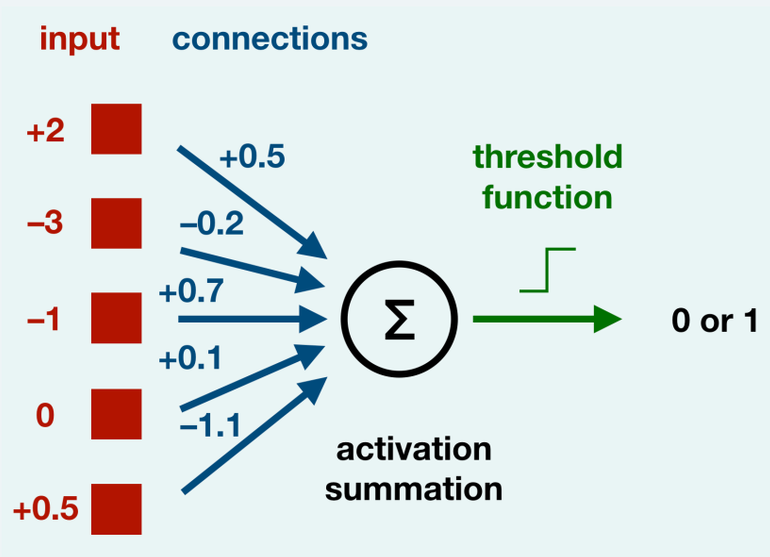
–0.7

+

0

+

–0.55



Perceptron: an artificial neuron

threshold function

theta = threshold

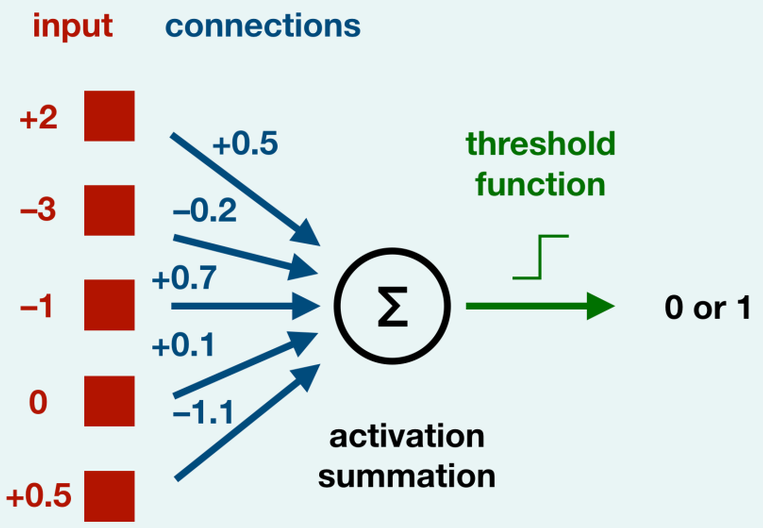
e.g. θ = 2

0.35

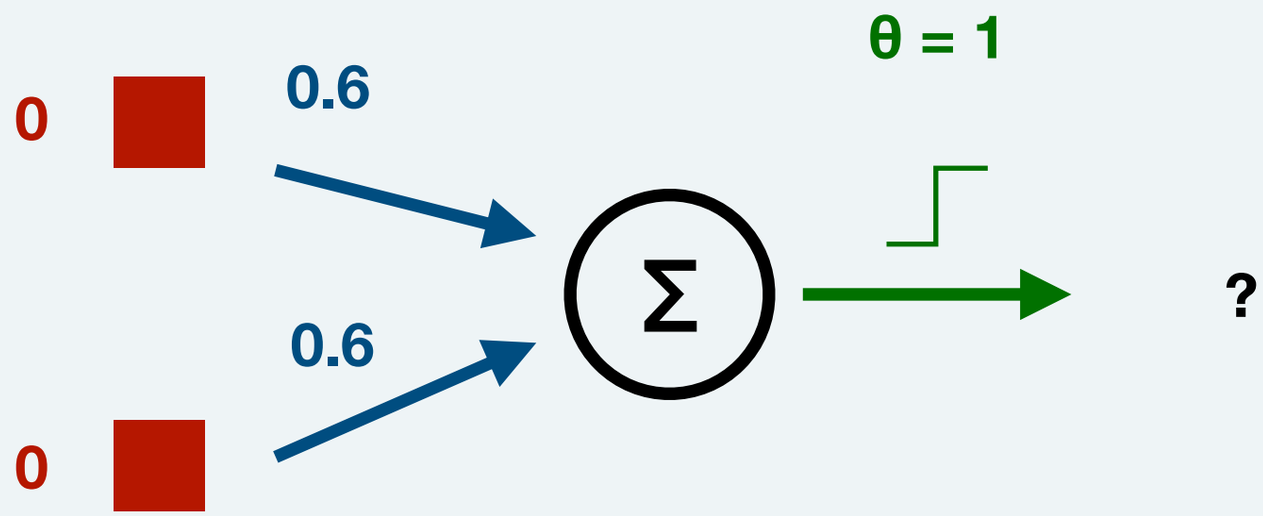
1 if x > θ

0 if x <= θ

0

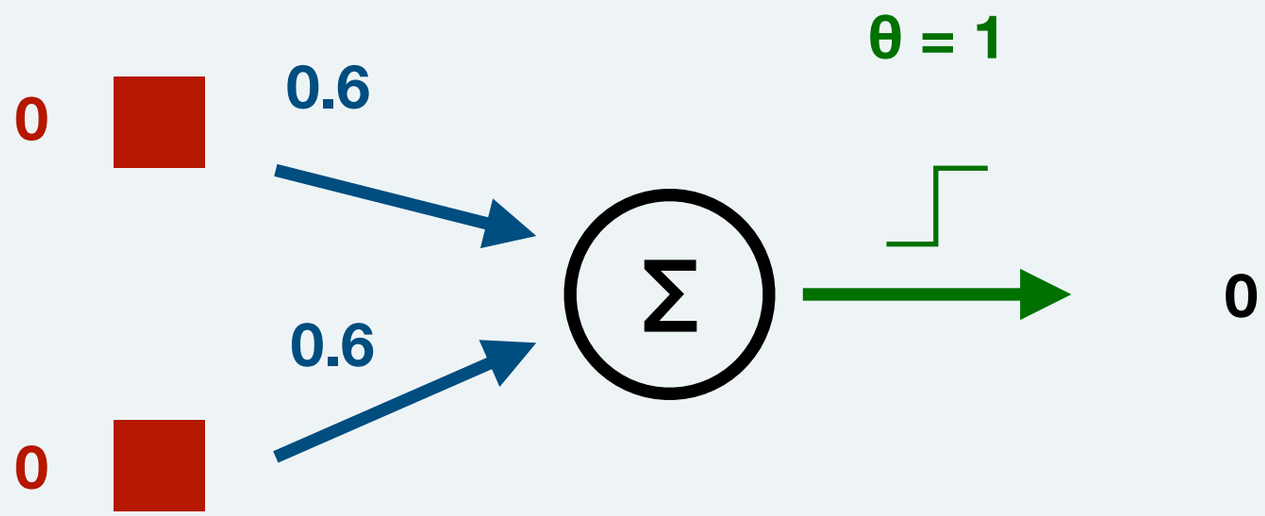


Perceptron: an artificial neuron

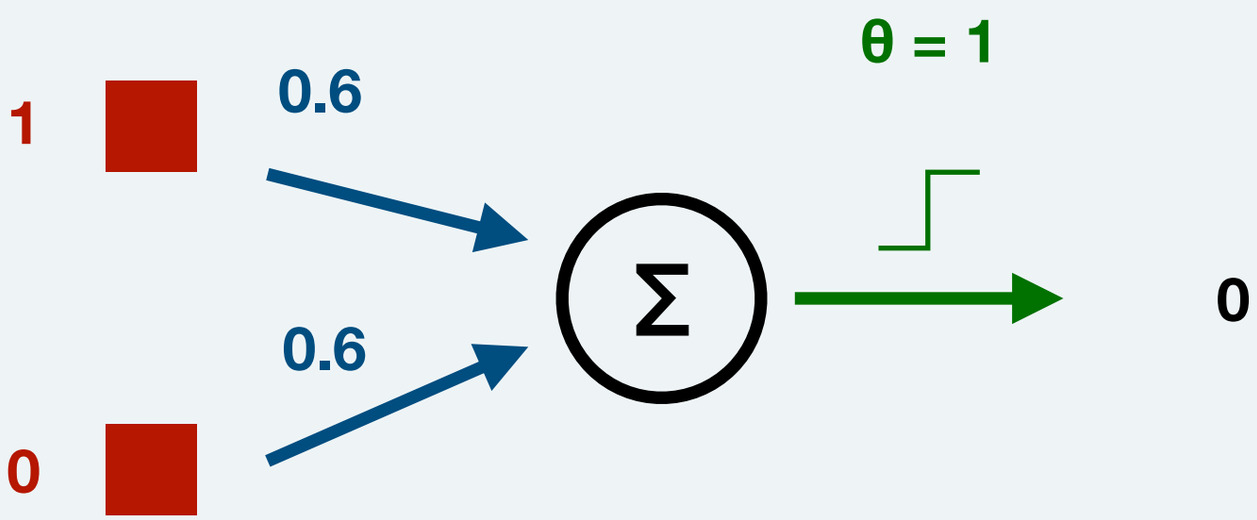


Produces 0: 0*.6+0*.6 = 0 < 1

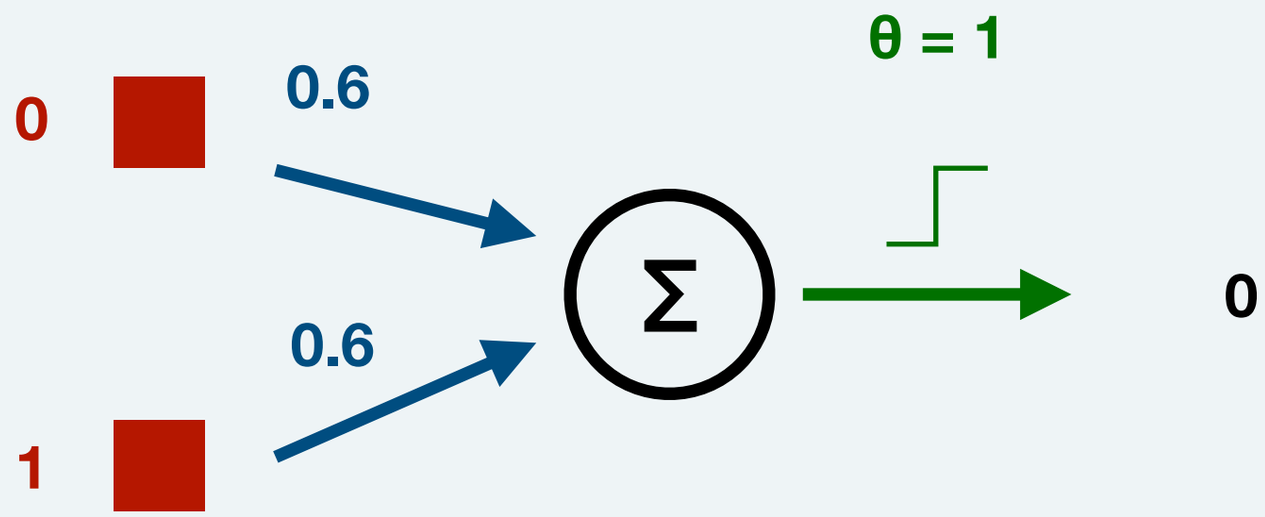
Perceptron: an artificial neuron



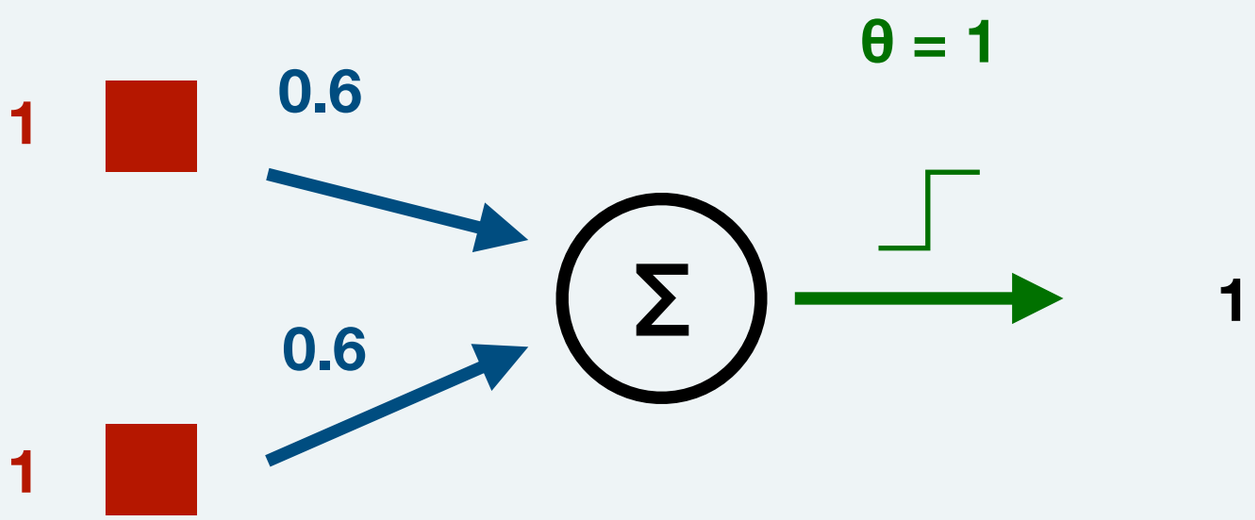
Perceptron: an artificial neuron



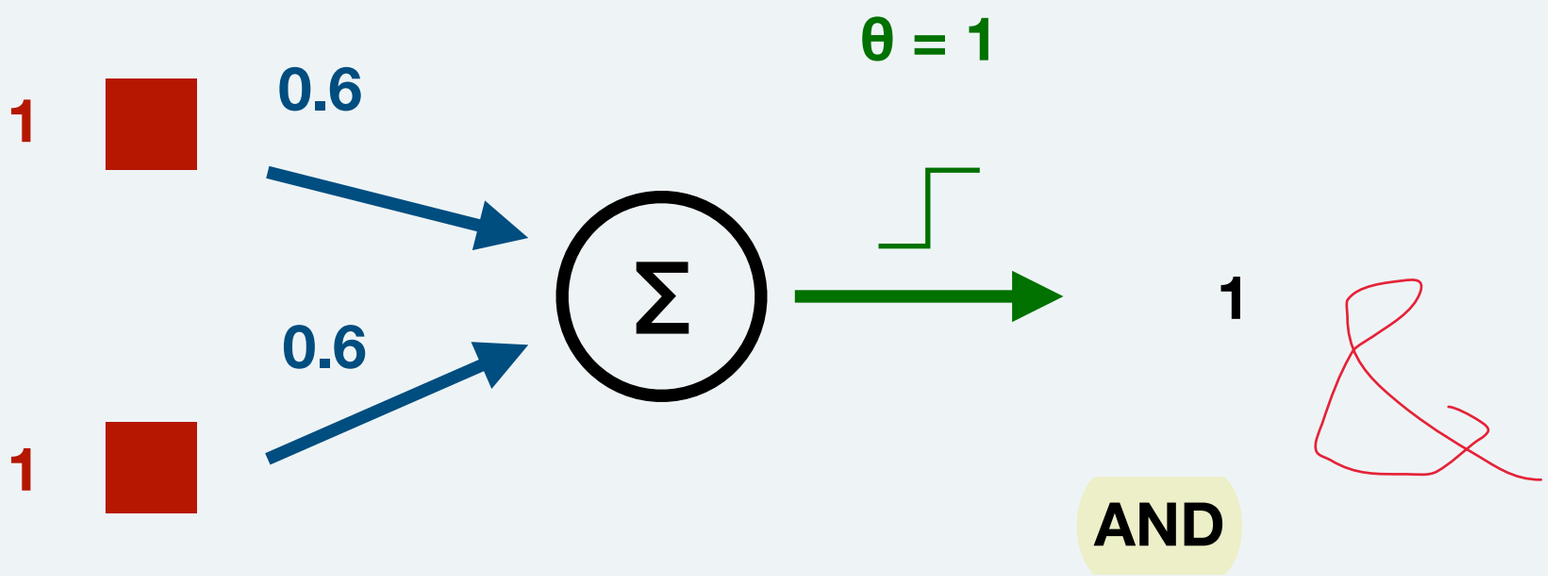
Perceptron: an artificial neuron



Perceptron: an artificial neuron



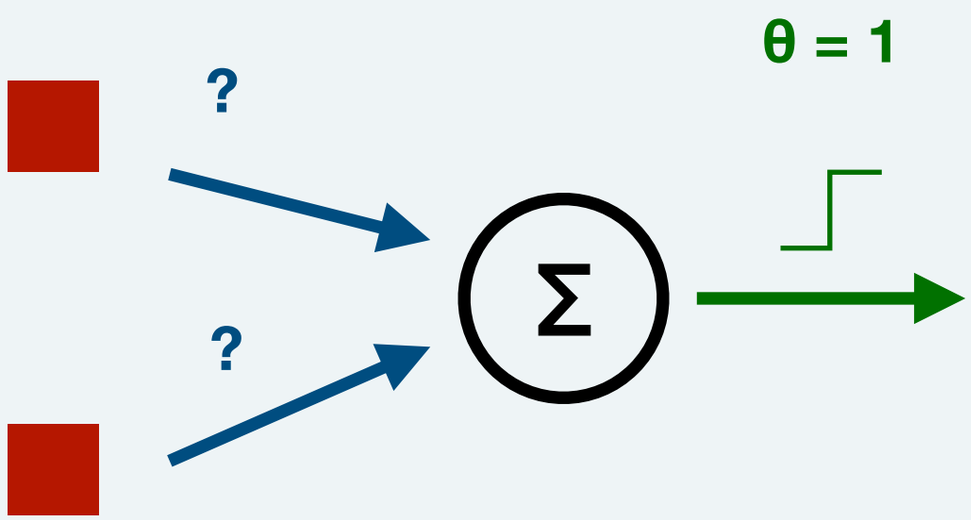
Perceptron: an artificial neuron



AND

0	0	0
0	1	0
1	0	0
1	1	1

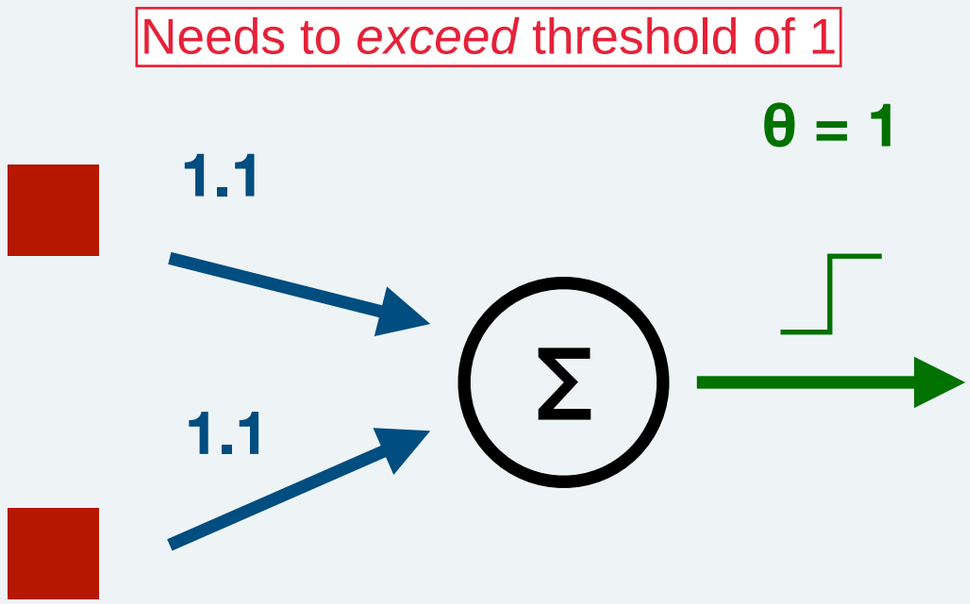
Perceptron: an artificial neuron



OR

0	0	0
0	1	1
1	0	1
1	1	1

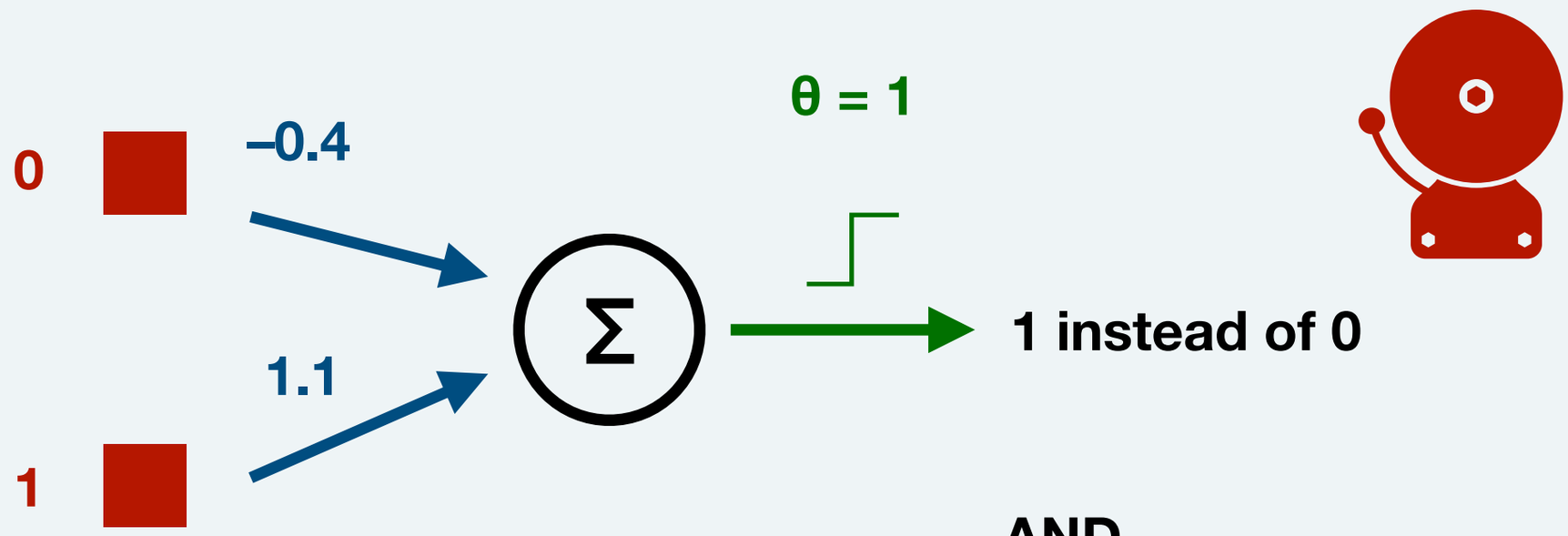
Perceptron: an artificial neuron



OR

0	0	0
0	1	1
1	0	1
1	1	1

The perceptron learning rule



AND

0	0	0
0	1	0
1	0	0
1	1	1

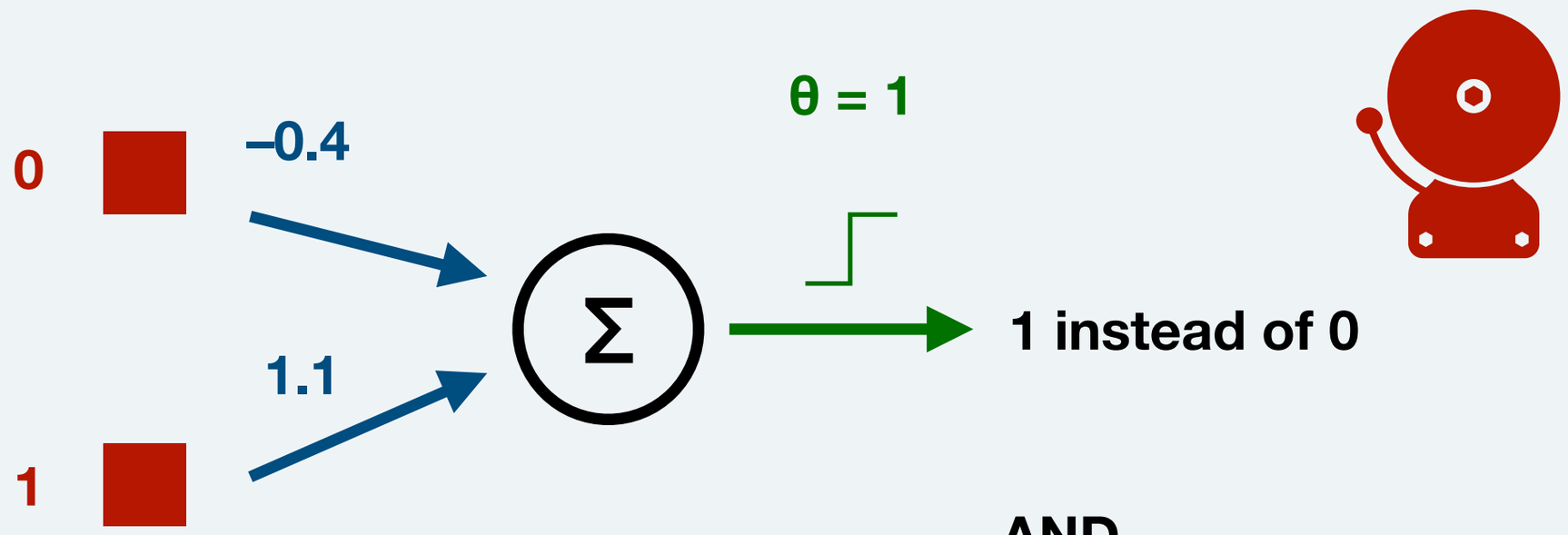
WRONG

Perceptron has rule that guides it to the correct weights

The perceptron learning rule

modify weights by:

small_number × **(target_output – actual_output)** × **input_vector**



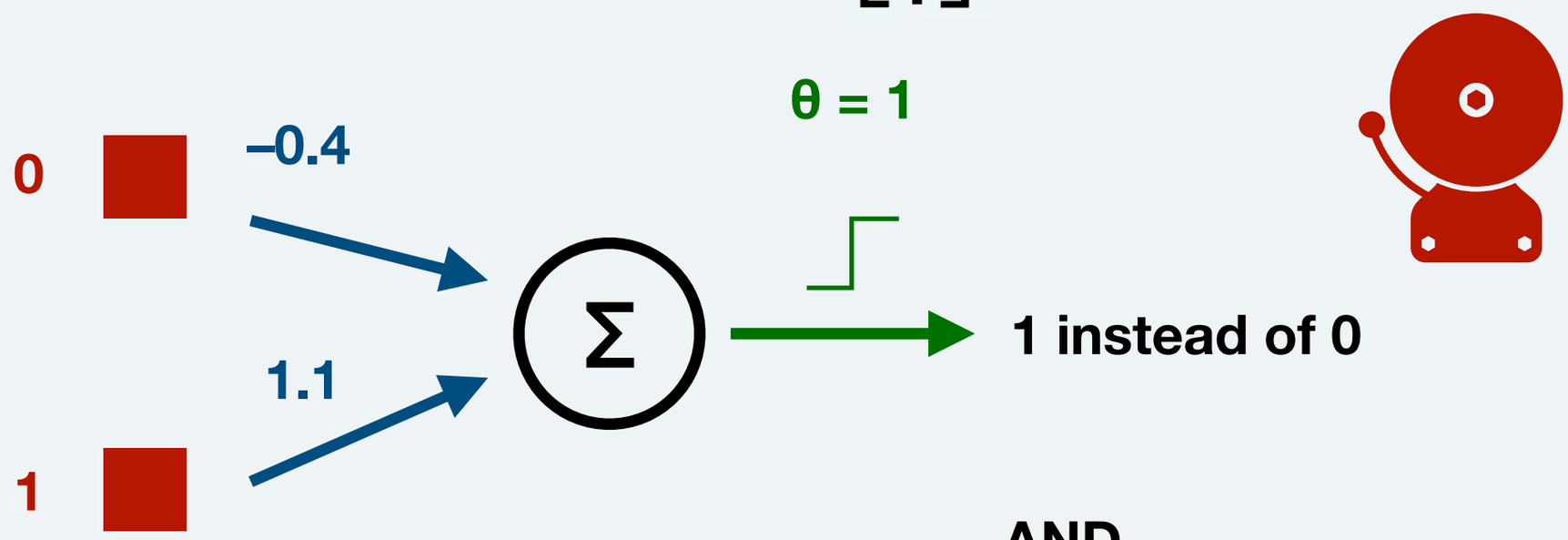
AND

0	0	0
0	1	0
1	0	0
1	1	1

The perceptron learning rule

modify weights by:

$$0.1 \times (0 - 1) \times \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$



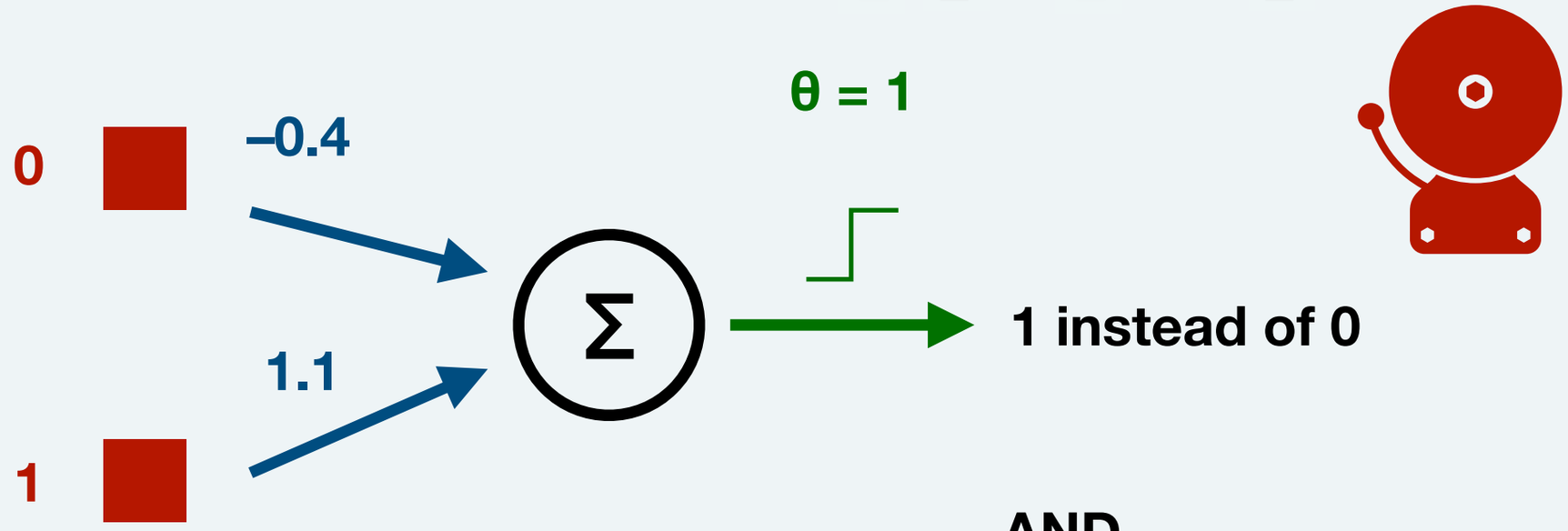
AND

0	0	0
0	1	0
1	0	0
1	1	1

The perceptron learning rule

modify weights by:

$$0.1 \times (0 - 1) \times \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ -0.1 \end{bmatrix}$$



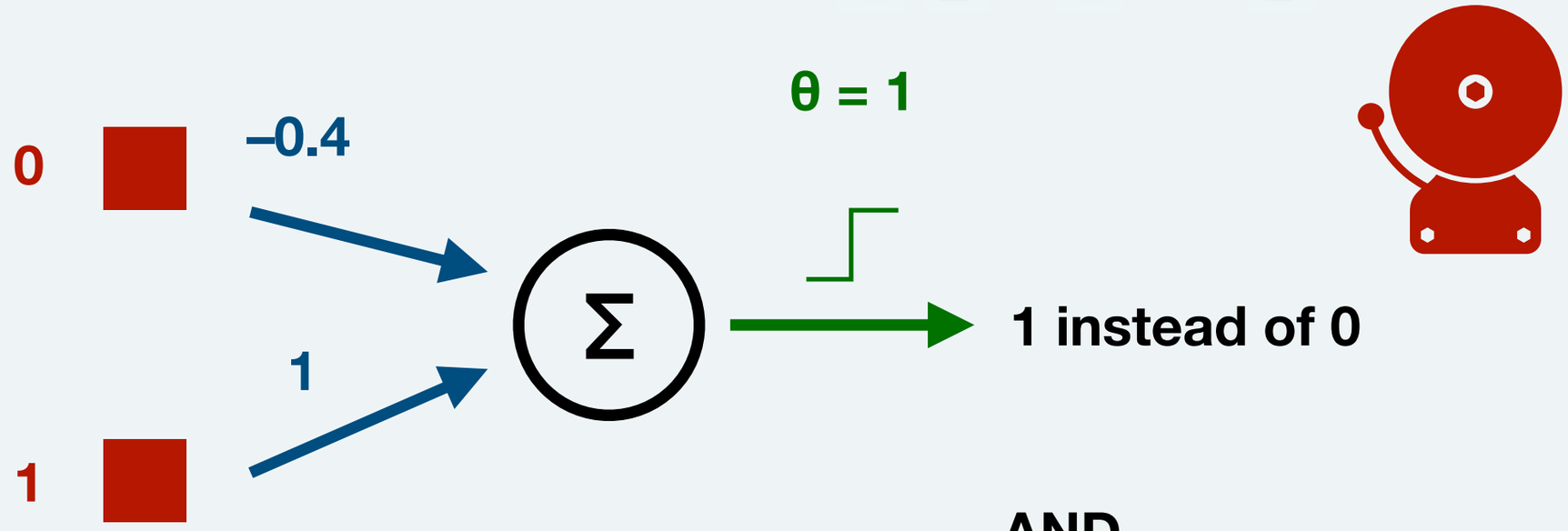
AND

0	0	0
0	1	0
1	0	0
1	1	1

The perceptron learning rule

modify weights by:

$$0.1 \times (0 - 1) \times \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ -0.1 \end{bmatrix}$$



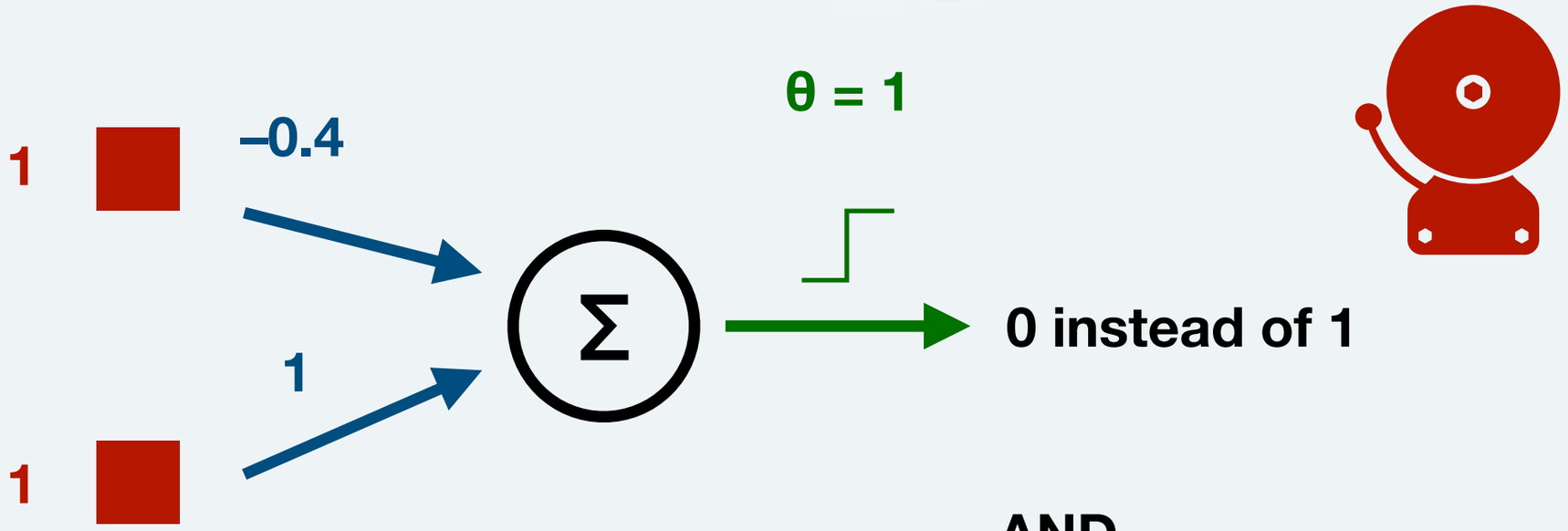
AND

0	0	0
0	1	0
1	0	0
1	1	1

The perceptron learning rule

modify weights by:

$$0.1 \times (1 - 0) \times \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$



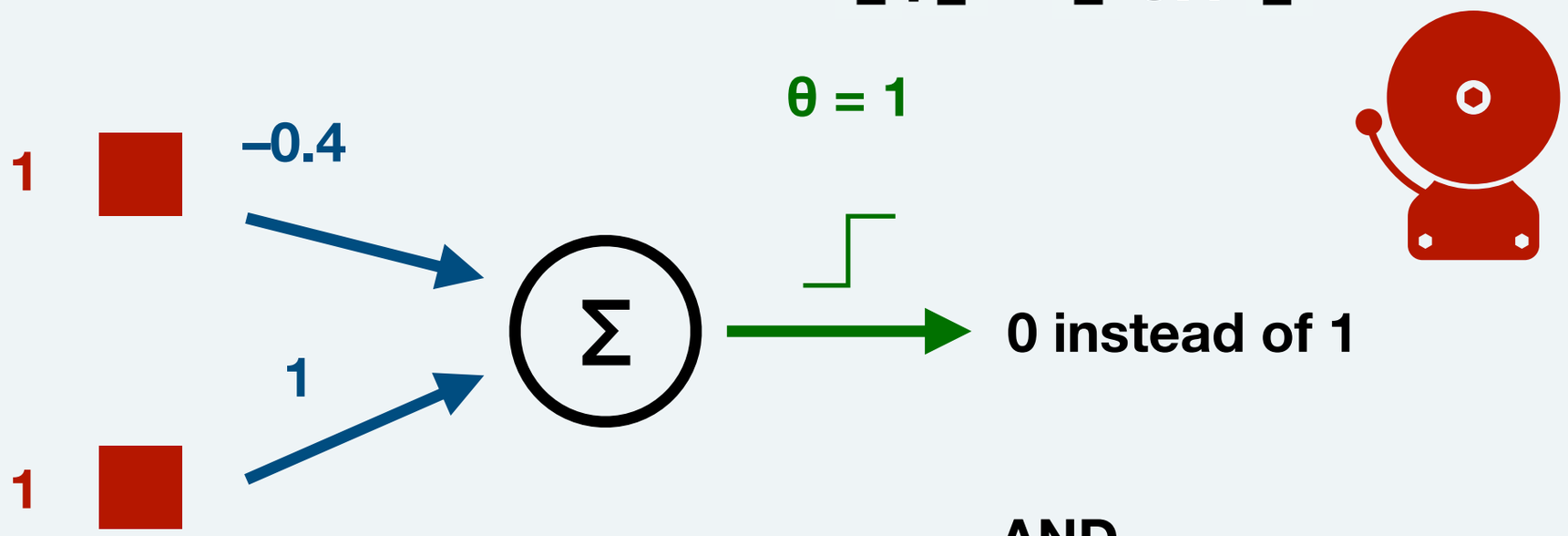
AND

0	0	0
0	1	0
1	0	0
1	1	1

The perceptron learning rule

modify weights by:

$$0.1 \times (1 - 0) \times \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0.1 \\ 0.1 \end{bmatrix}$$



will output 0 instead of 1: needs to continue modifying weights

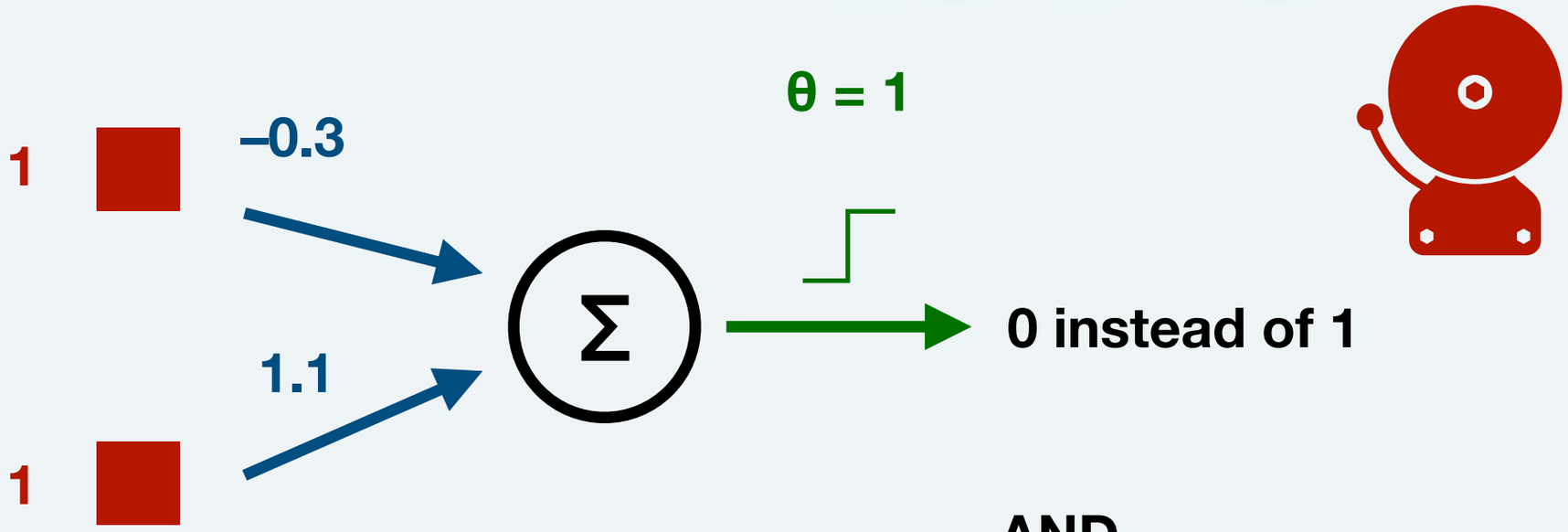
AND

0	0	0
0	1	0
1	0	0
1	1	1

The perceptron learning rule

modify weights by:

$$0.1 \times (1 - 0) \times \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0.1 \\ 0.1 \end{bmatrix}$$



trying diff weights...

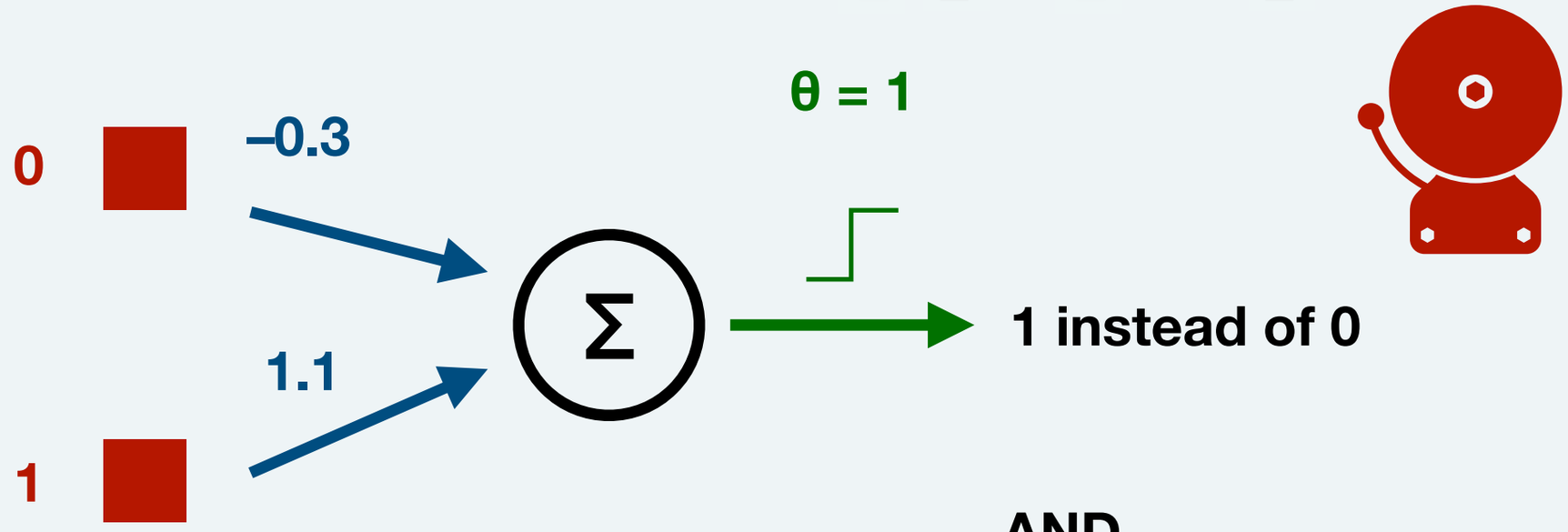
AND

0	0	0
0	1	0
1	0	0
1	1	1

The perceptron learning rule

modify weights by:

$$0.1 \times (0 - 1) \times \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ -0.1 \end{bmatrix}$$



still wrong...

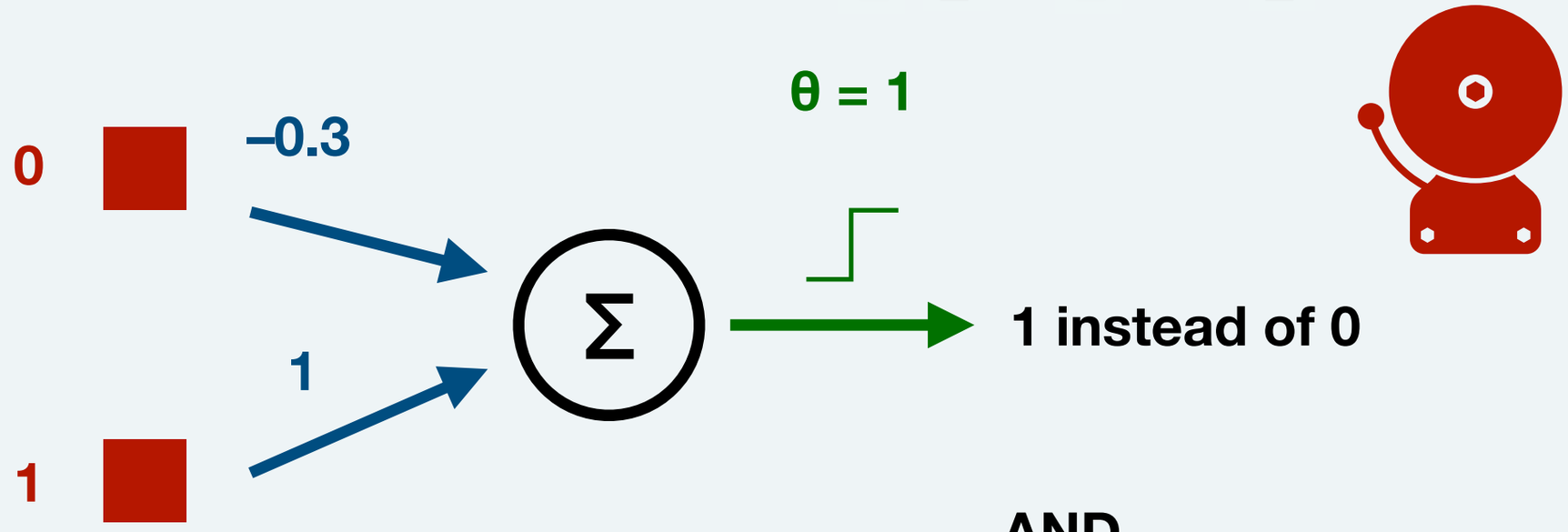
AND

0	0	0
0	1	0
1	0	0
1	1	1

The perceptron learning rule

modify weights by:

$$0.1 \times (0 - 1) \times \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ -0.1 \end{bmatrix}$$

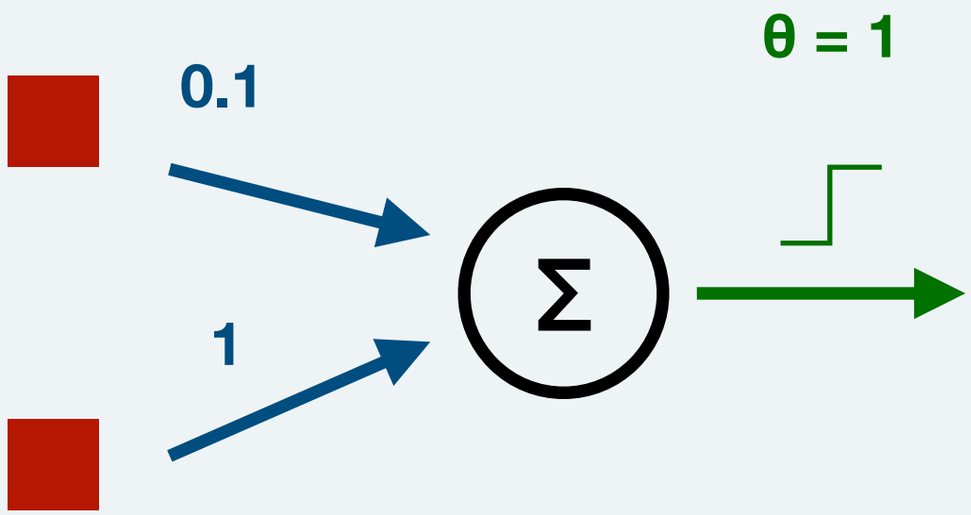


still wrong...

AND

0	0	0
0	1	0
1	0	0
1	1	1

The perceptron learning rule



Correct weighting!

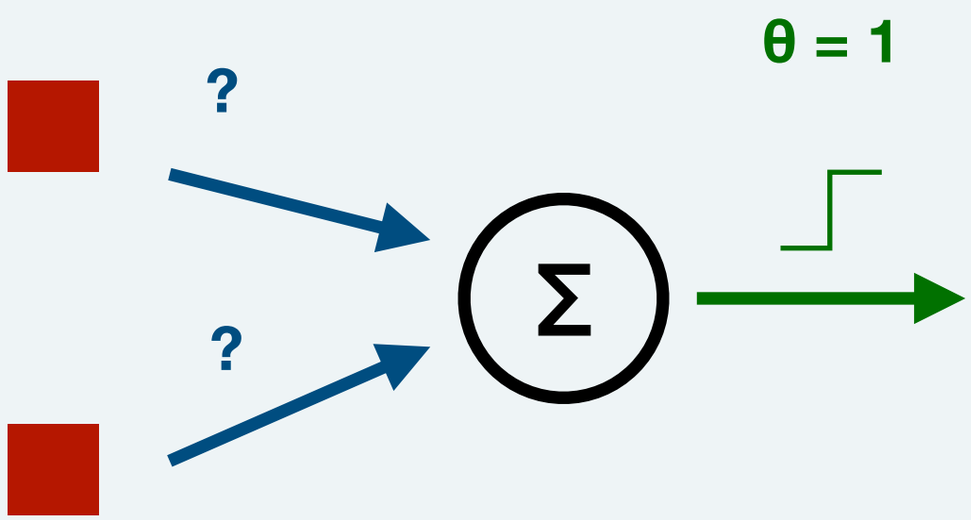
You did it perceptron

Perceptron is guaranteed to find the correct weights eventually (might be slow at large # inputs/weights)

AND

0	0	0
0	1	0
1	0	0
1	1	1

Perceptron: an artificial neuron

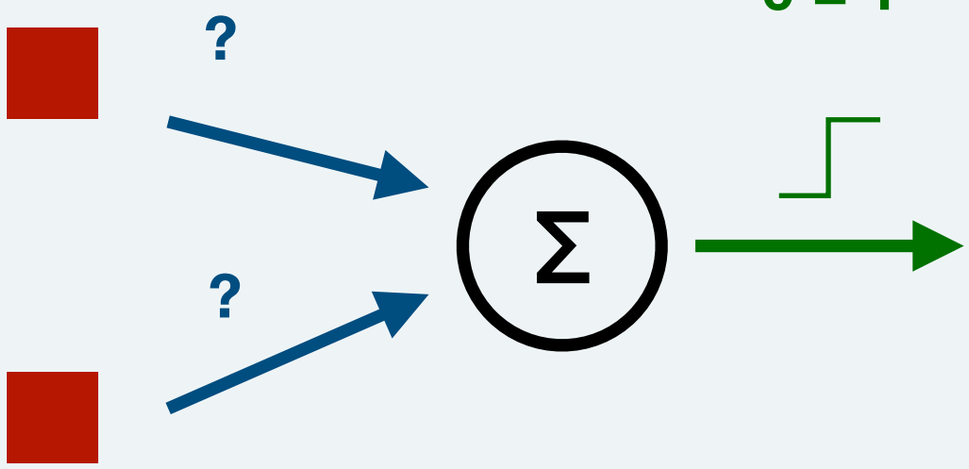


you can't!

Perceptron only has one layer of processing: very limited

		XOR
0	0	0
0	1	1
1	0	1
1	1	0

Perceptron: an artificial neuron

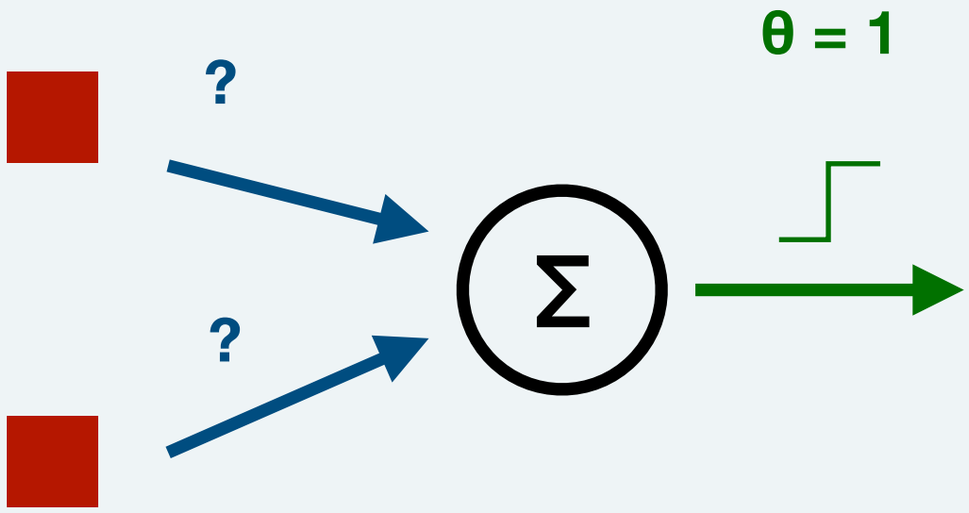


XOR

0	0	0
0	1	1
1	0	1
1	1	0

$$xw_x + yw_y = \theta$$

Perceptron: an artificial neuron

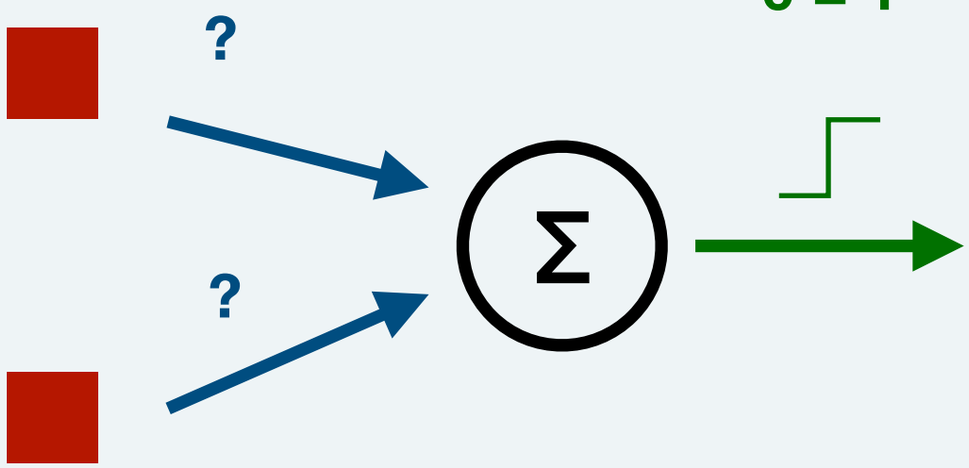


XOR

0	0	0
0	1	1
1	0	1
1	1	0

$$yw_y = \theta - xw_x$$

Perceptron: an artificial neuron

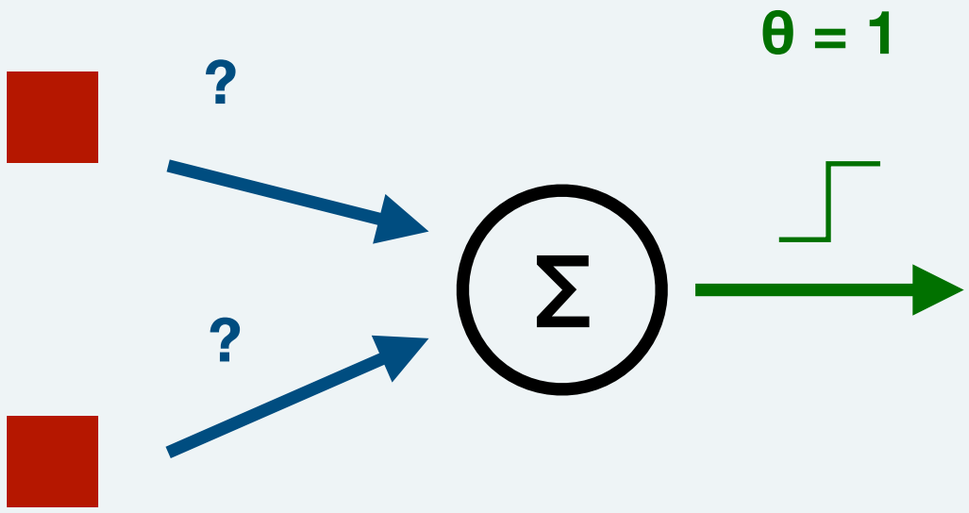


XOR

0	0	0
0	1	1
1	0	1
1	1	0

$$y = \theta/w_y - xw_x/w_y$$

Perceptron: an artificial neuron

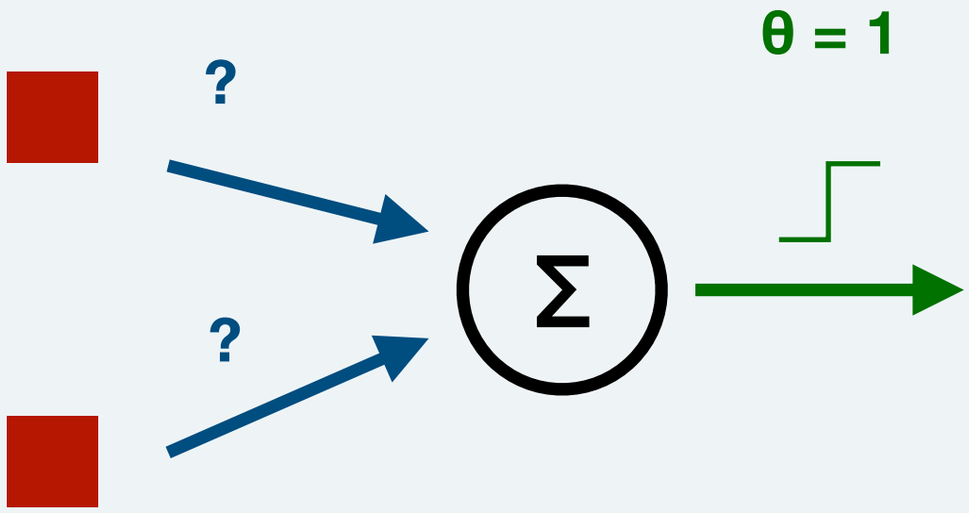


XOR

0	0	0
0	1	1
1	0	1
1	1	0

$$y = \underbrace{\theta / w_y}_{a} - \underbrace{x w_x / w_y}_{b}$$

Perceptron: an artificial neuron



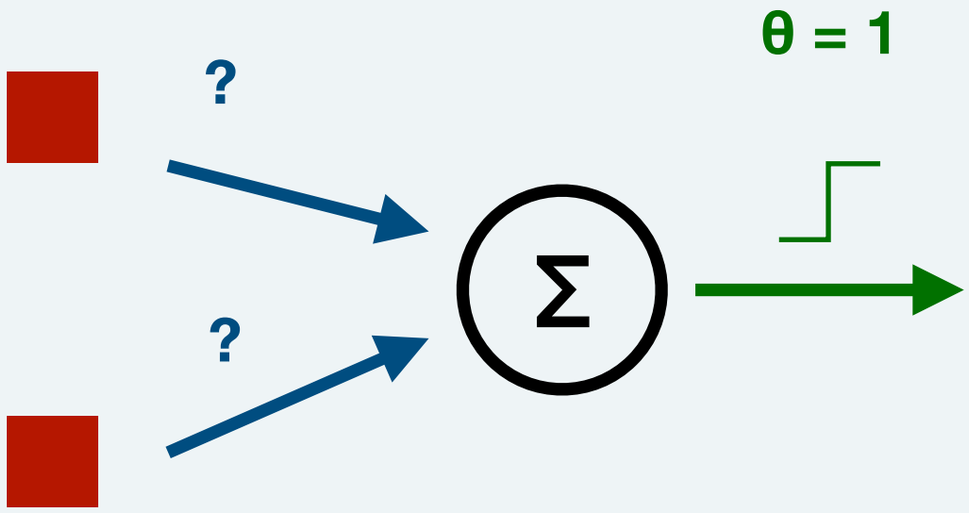
XOR

0	0	0
0	1	1
1	0	1
1	1	0

Equation for slope of a line.. duh

$$y = a + bx$$

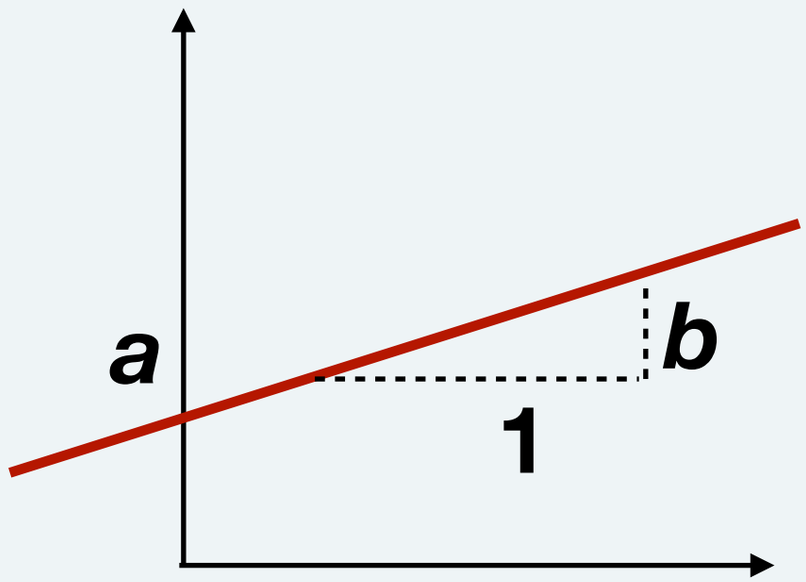
Perceptron: an artificial neuron



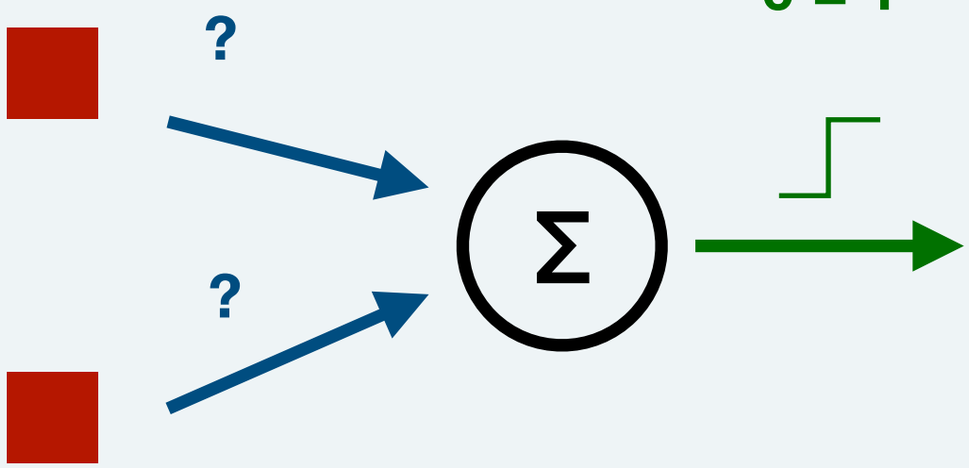
XOR

0	0	0
0	1	1
1	0	1
1	1	0

$$y = a + bx$$



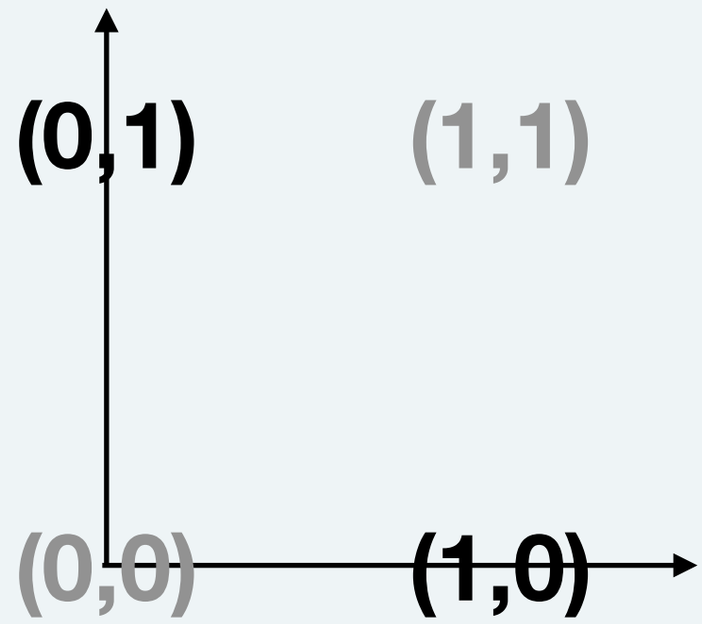
Perceptron: an artificial neuron



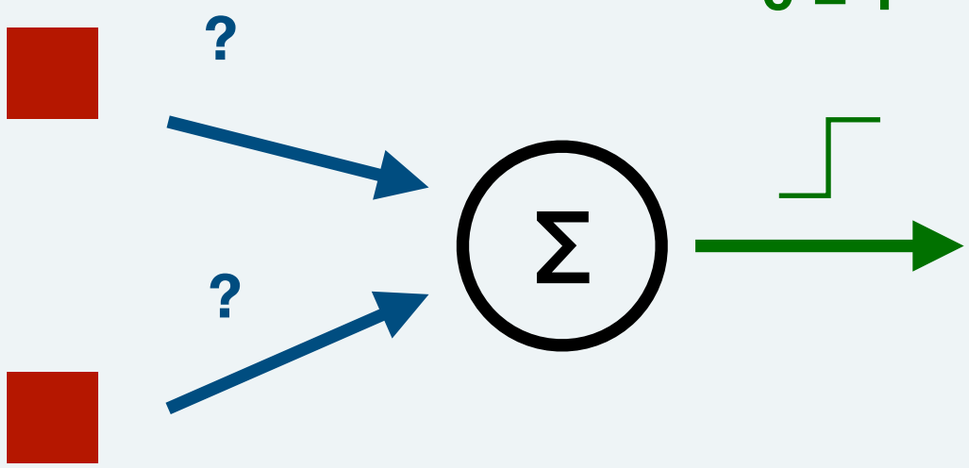
XOR

0	0	0
0	1	1
1	0	1
1	1	0

$y = a + bx$



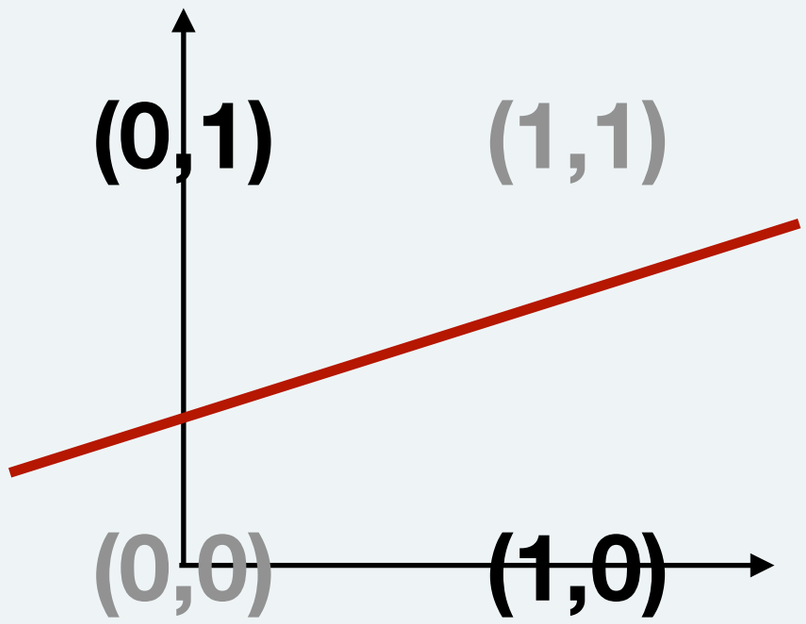
Perceptron: an artificial neuron

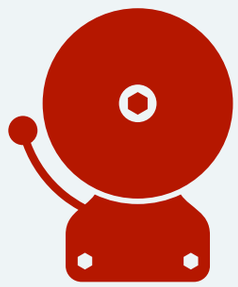


XOR

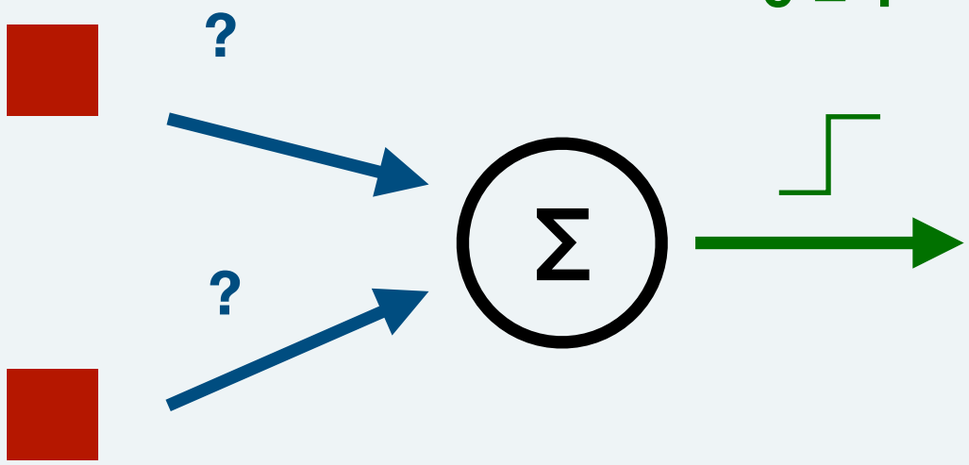
0	0	0
0	1	1
1	0	1
1	1	0

$y = a + bx$





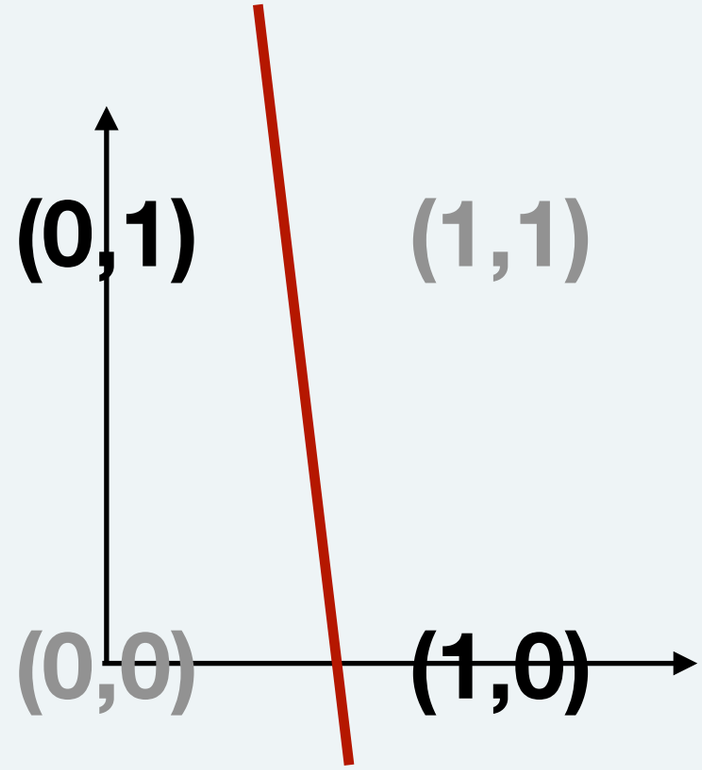
Perceptron: an artificial neuron

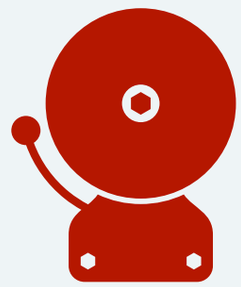


XOR

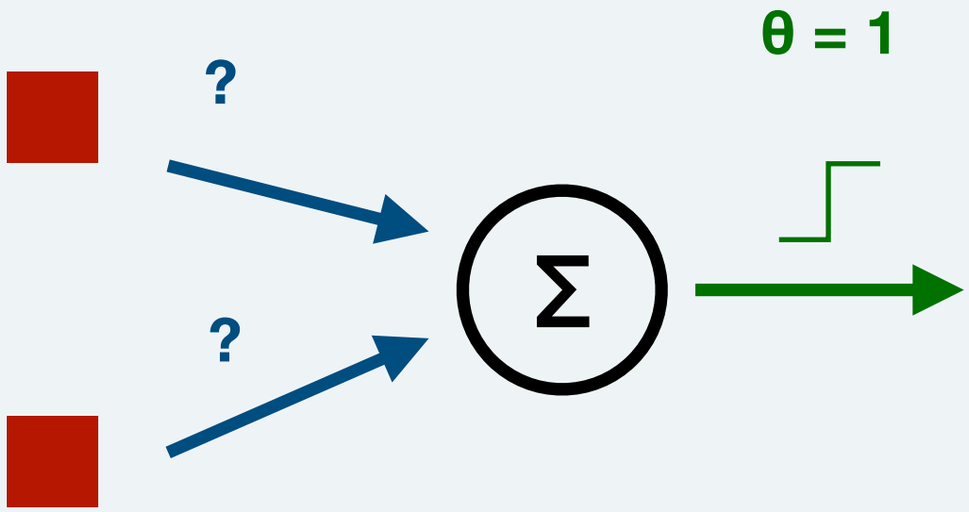
0	0	0
0	1	1
1	0	1
1	1	0

$$y = a + bx$$





Perceptron: an artificial neuron

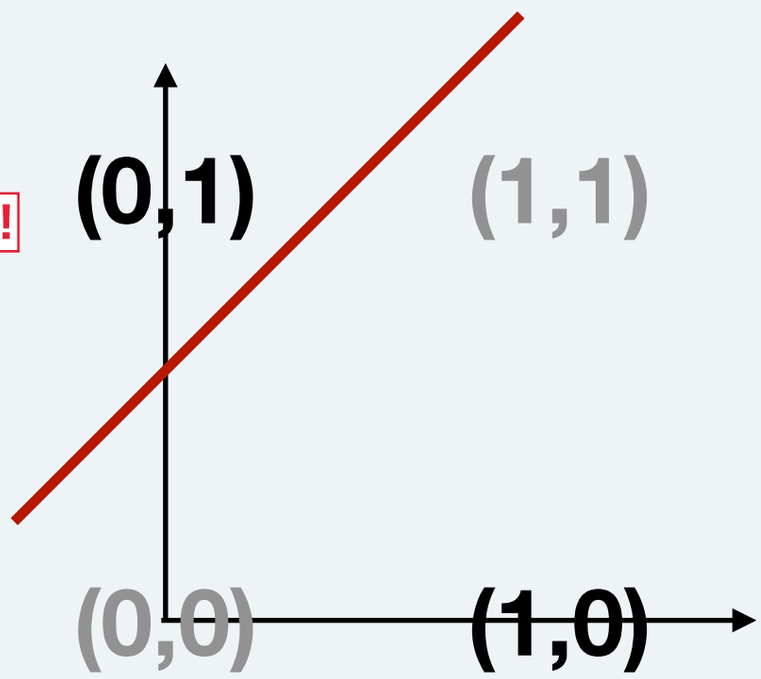


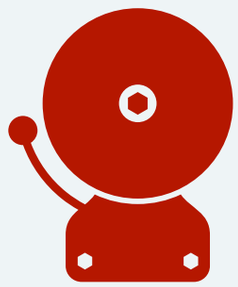
XOR

0	0	0
0	1	1
1	0	1
1	1	0

Doesn't work for non-linear functions!

$$y = a + bx$$

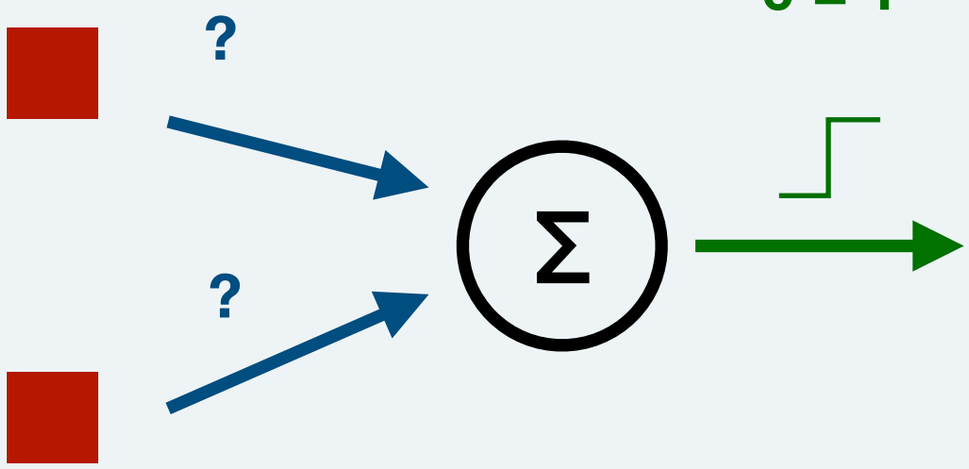




Perceptron: an artificial neuron

https://www.artavita.com/artists/9602-ben-turner

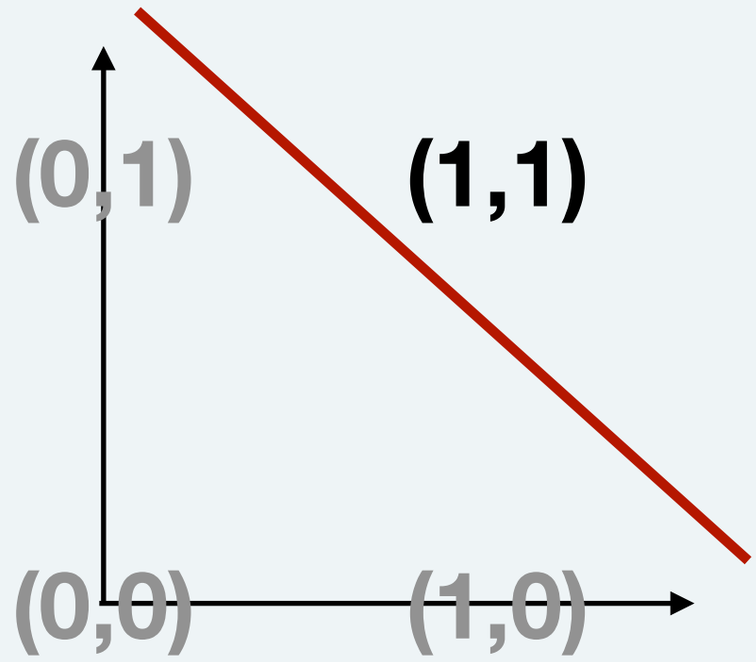
Perceptron: an artificial neuron



AND

0	0	0
0	1	0
1	0	0
1	1	1

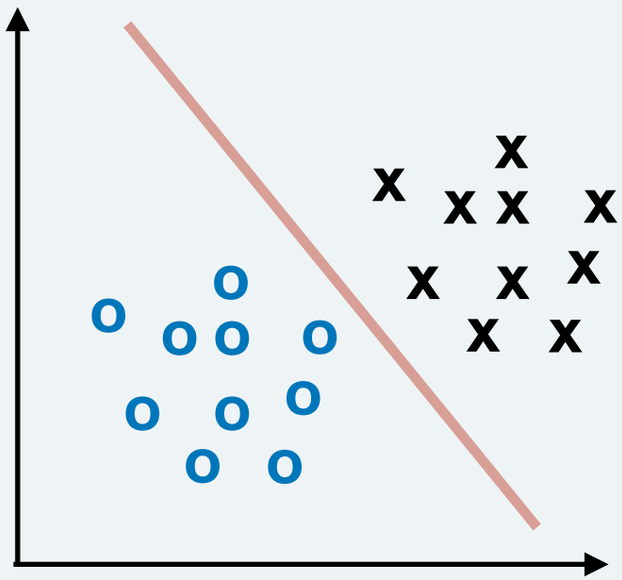
$$y = a + bx$$



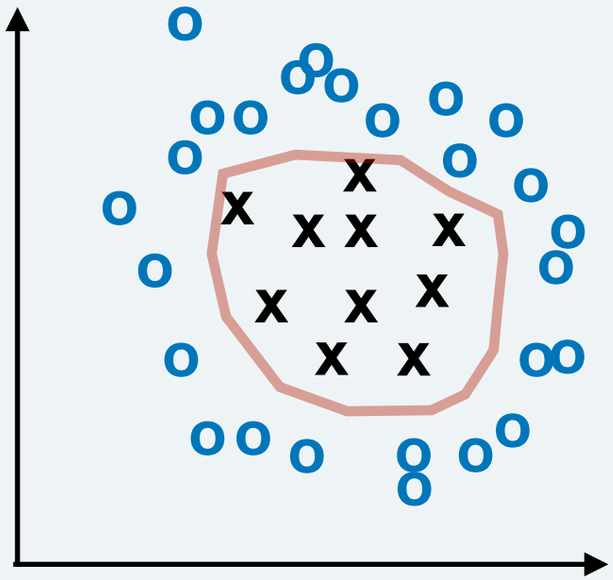
Perceptron: an artificial neuron

Perceptron can only handle linearly separable

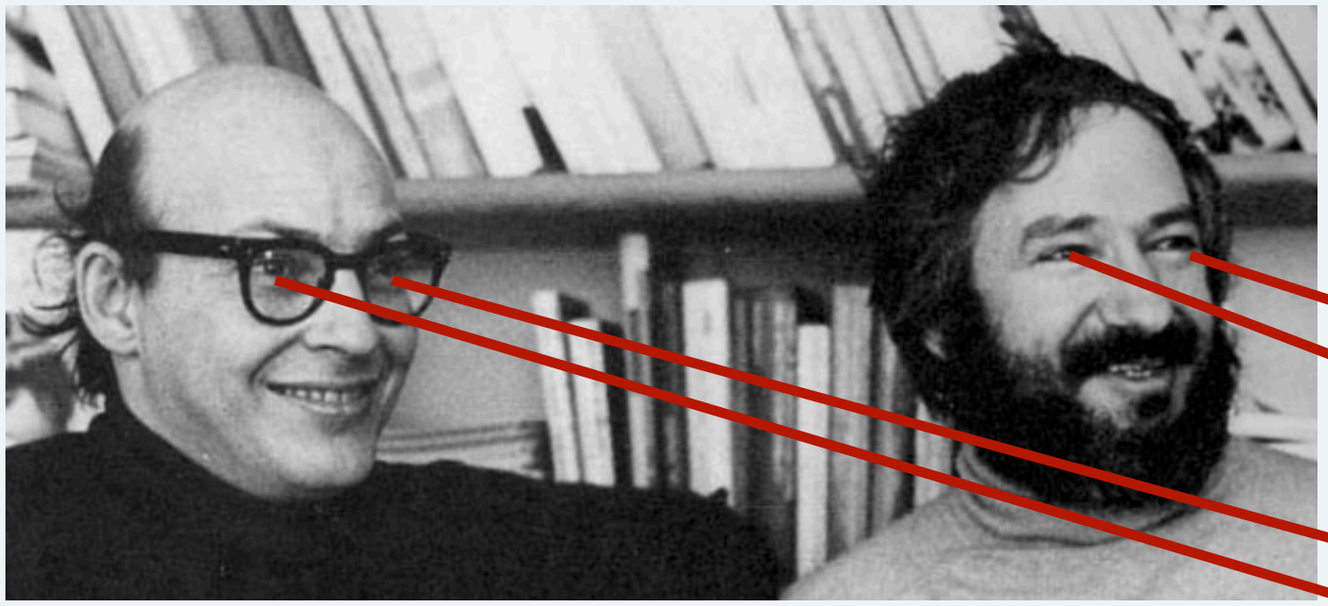
linearly separable



not linearly separable



Scientists at each other's throats – again!



Scientists at each other's throats – again!

Marvin Minsky

Team Perceptron (key date: 1957)

Frank Rosenblatt

TBT PHIL 351!

Seymour Papert

Team Symbolic AI (aka: PSS) (key date: 1969)

"AI winter": people didn't work on ANNs for a while