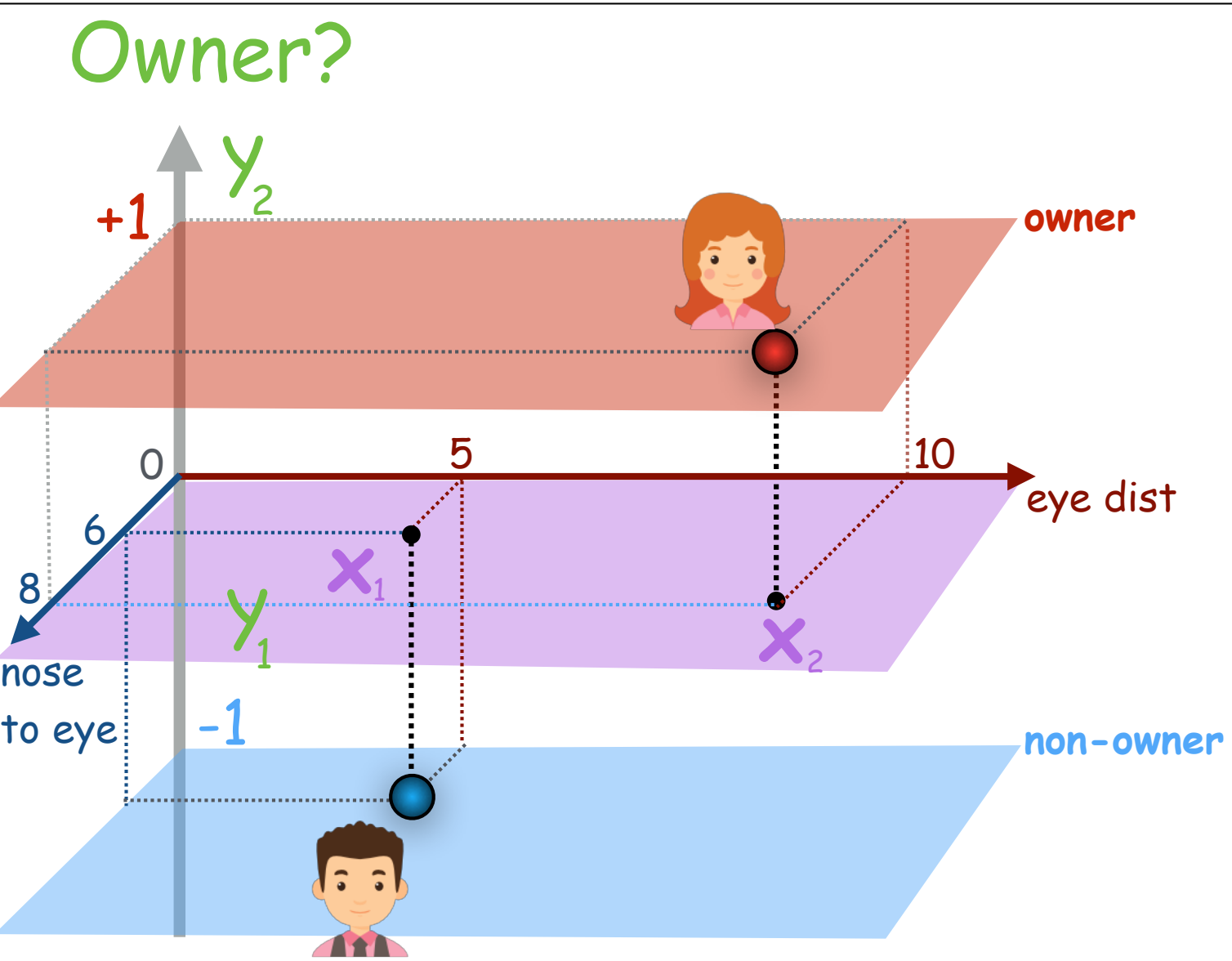
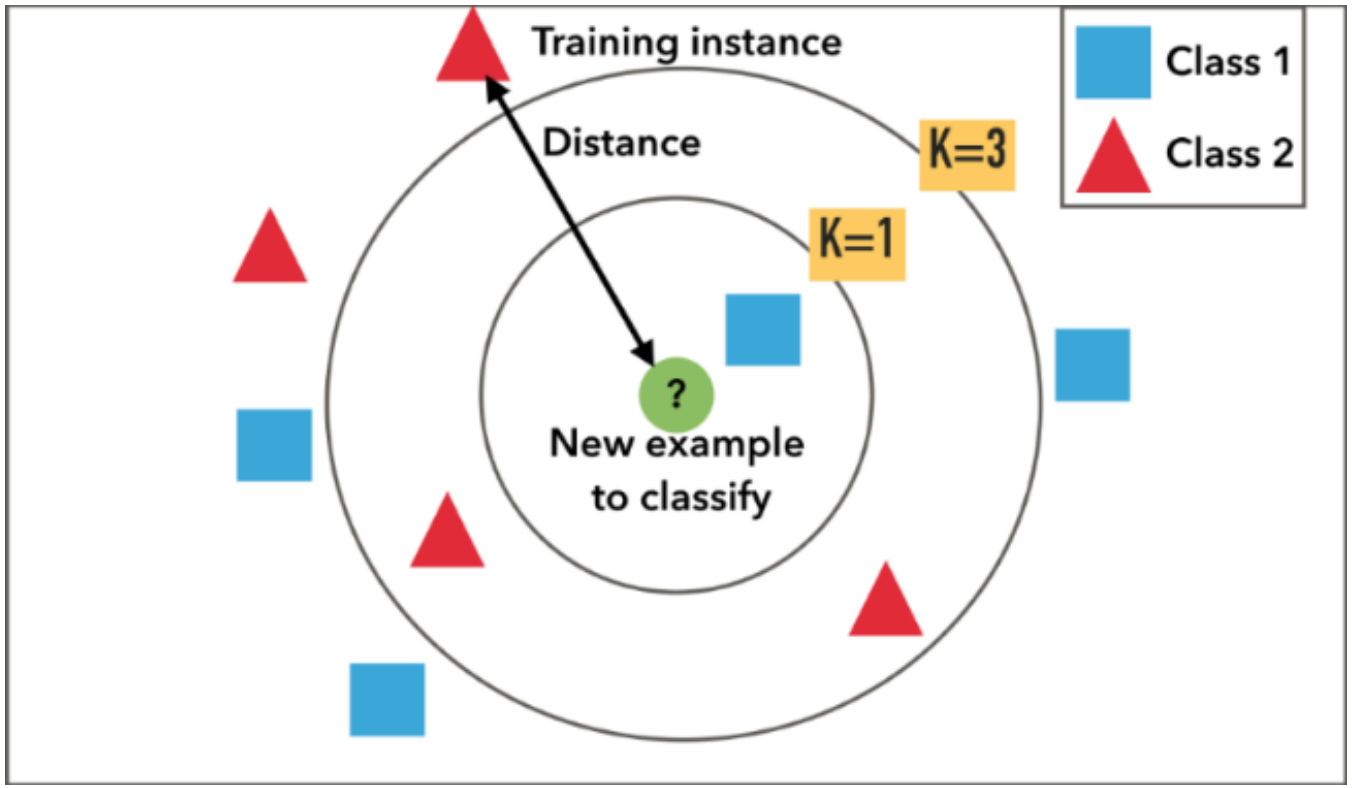
Neural Networks

Classification Task

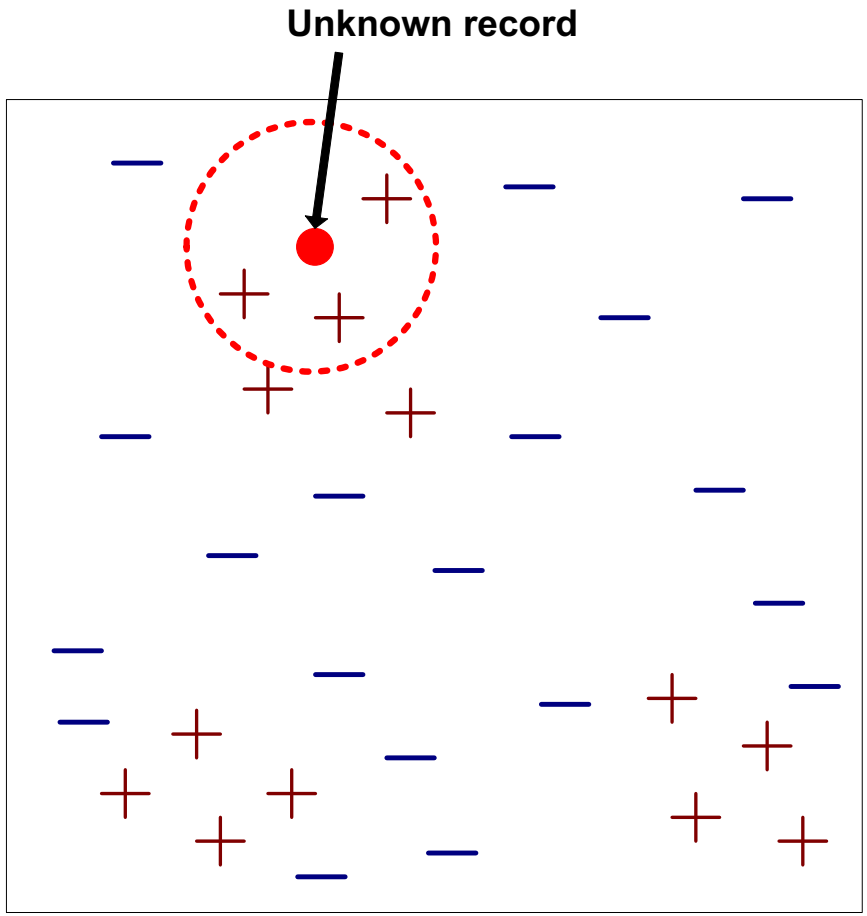


$x \in R^2$ $y \in \{-1,1\}$ find a function $y = f(x)$

Nearest Neighbor Classifier



Nearest Neighbor Classifier

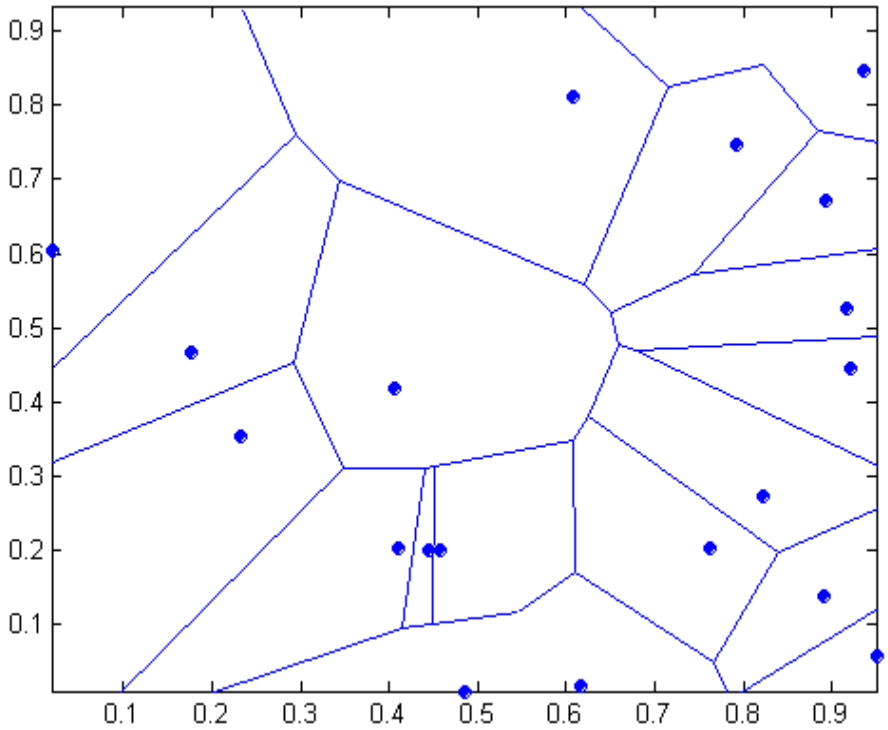


- Requires three things
  - The set of stored records
  - Distance Metric to compute distance between records
  - The value of *k*, the number of nearest neighbors to retrieve

- To classify an unknown record:
  - Compute distance to other training records
  - Identify *k* nearest neighbors
  - Use class labels of nearest neighbors to determine the class label of unknown record (e.g., by taking majority vote)

KNN

model: training instances



fast training

large memory cost (GB of data)

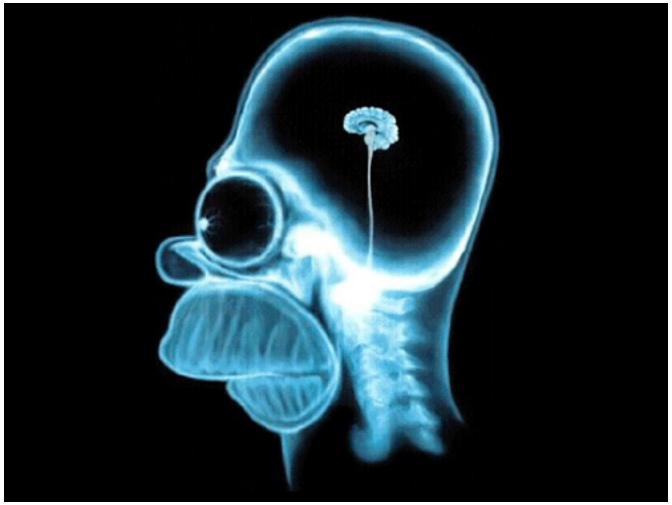
slow inference with large dataset

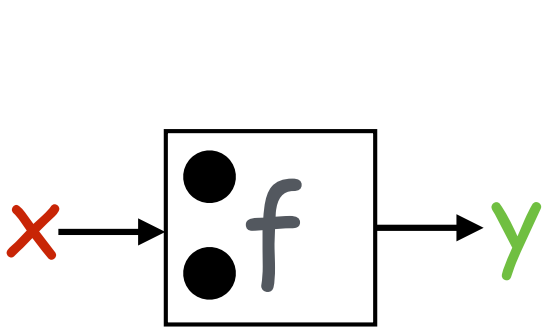
O(p n) for each test data

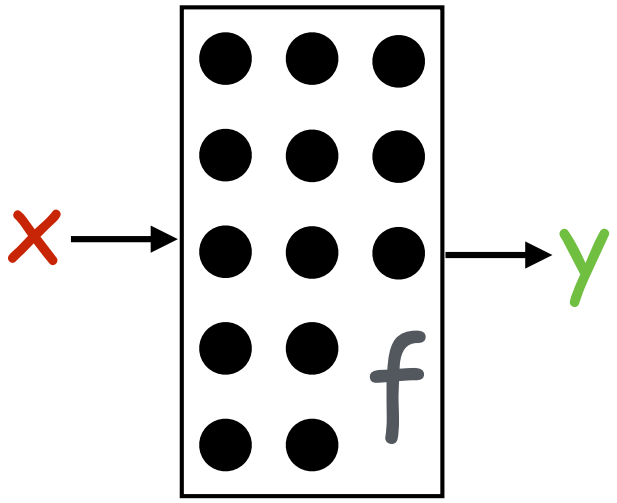
n : # training instances

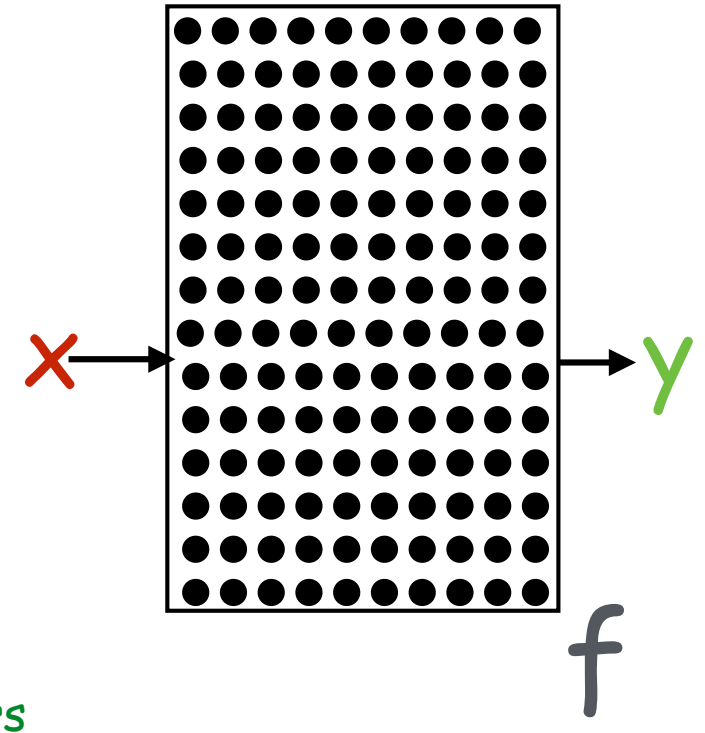
Also called Non-Parametric Models

"Non-parametric"



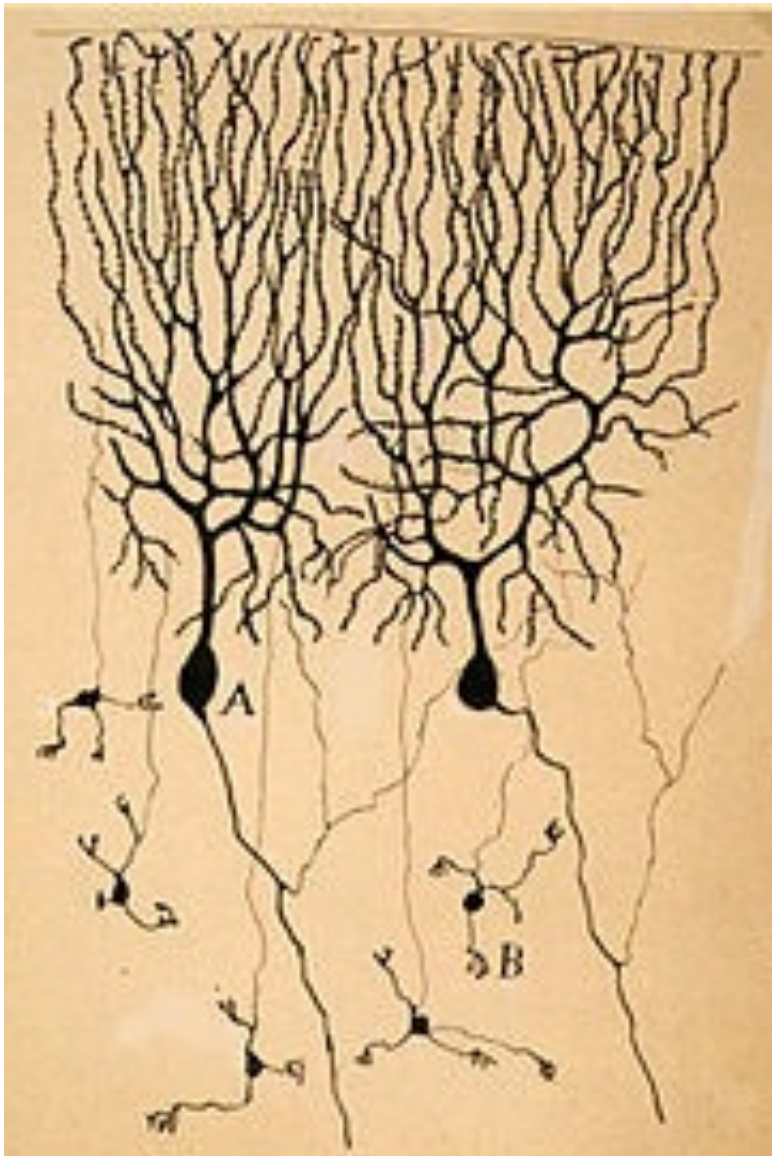




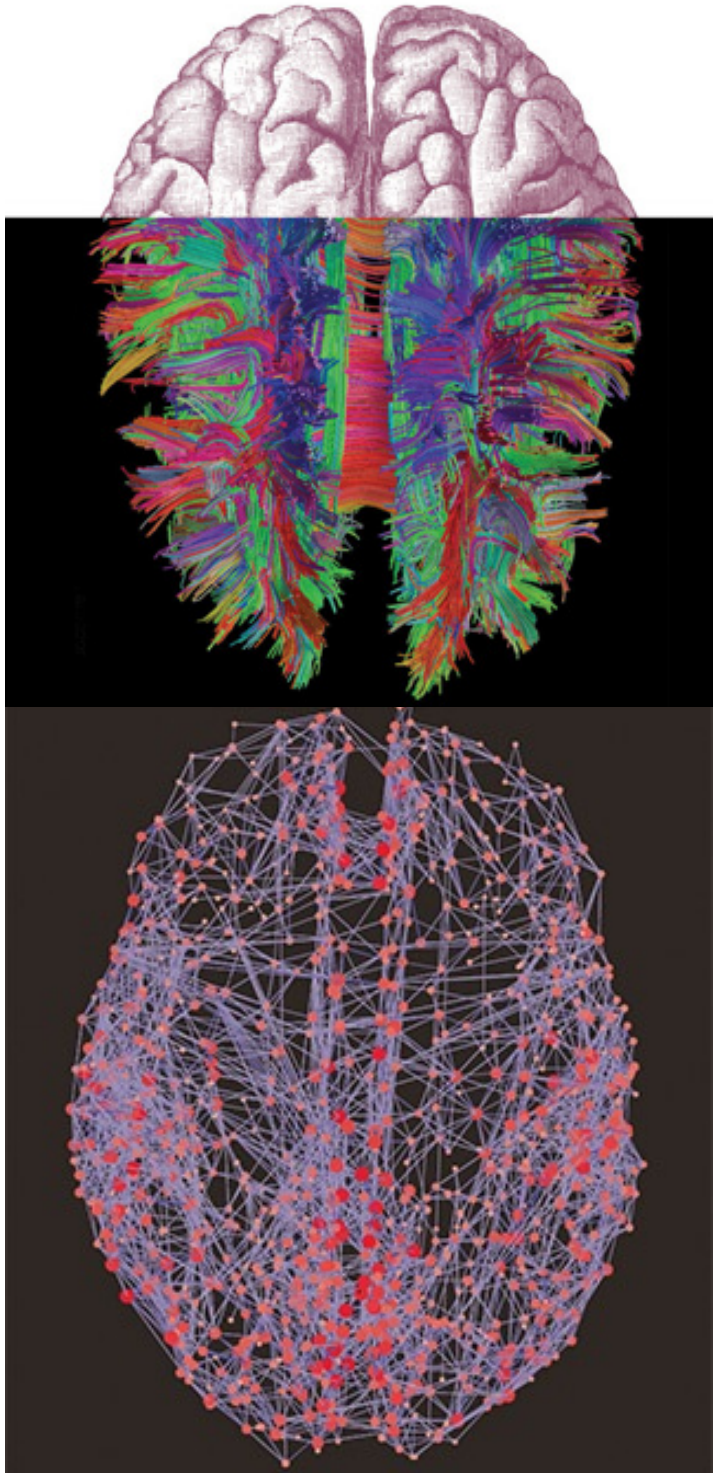


training data are the parameters

Neuroscience



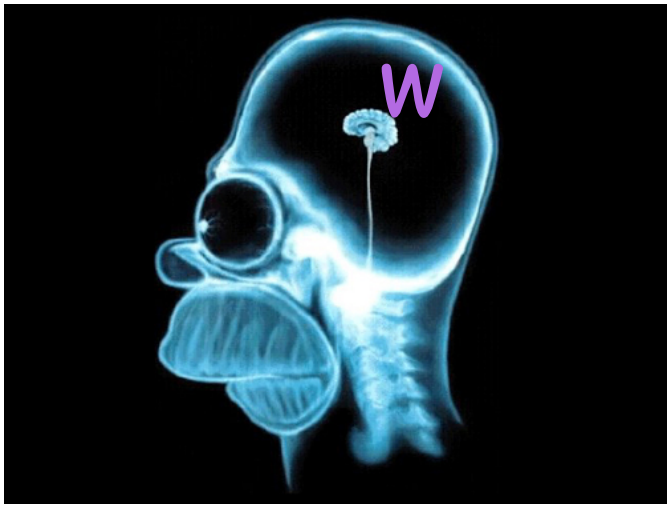
Networks of the Brain

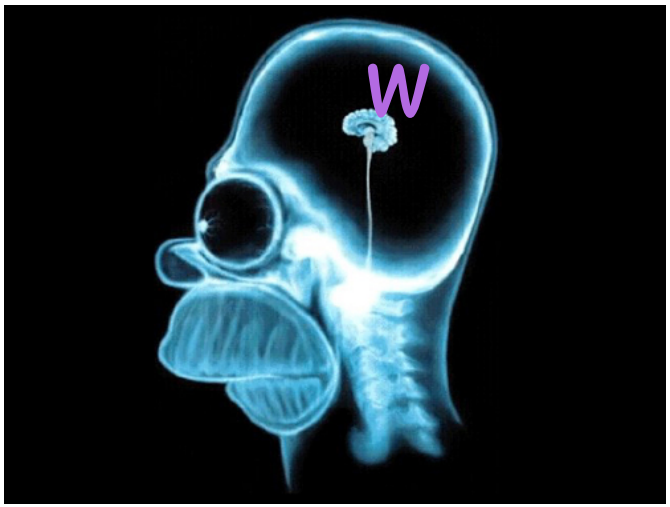


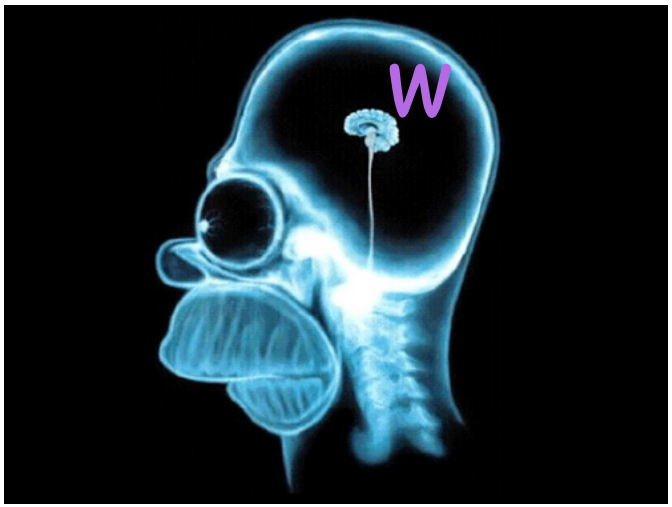
Santiago Ramón y Cajal
Nobel Prize in Medicine

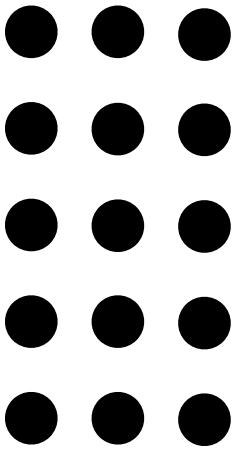
1899

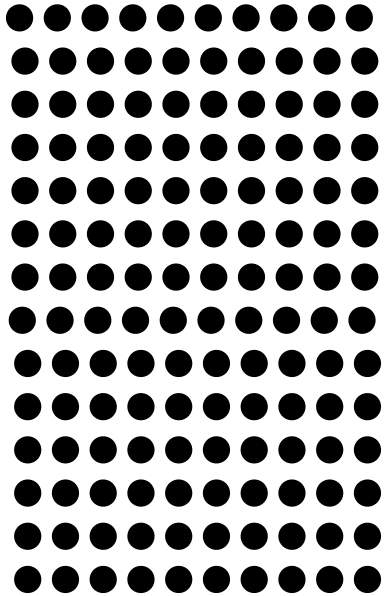
Parametric Approach

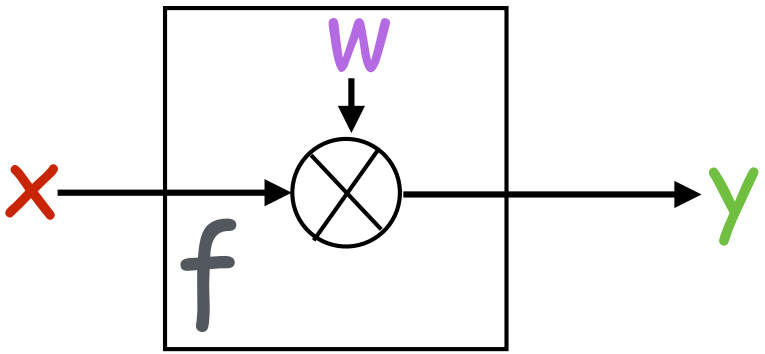








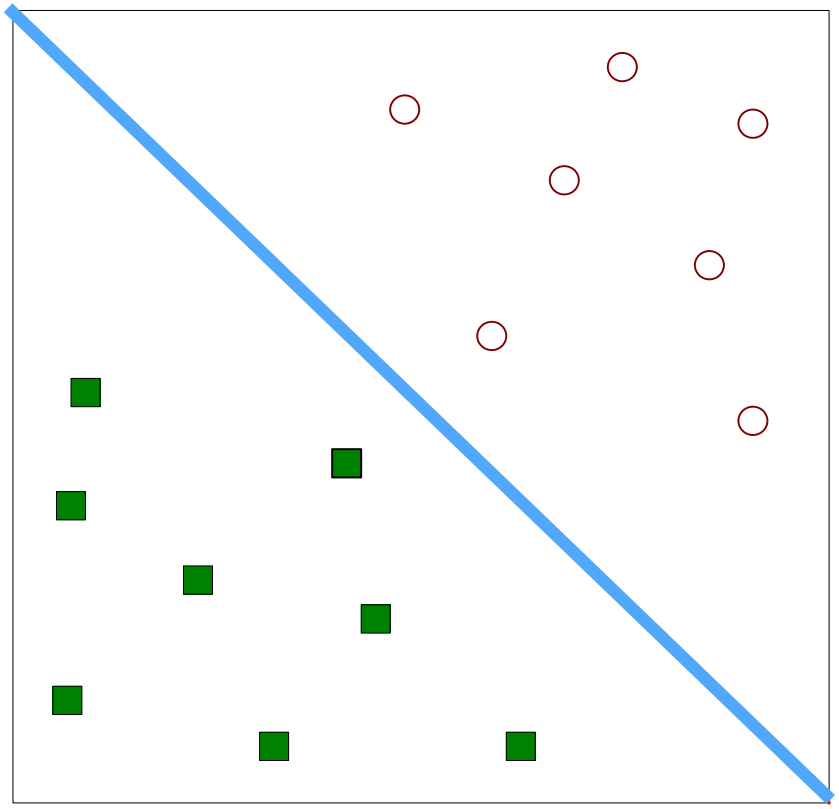




" Parametric approach"

Parametric Models

model: a set of parameters (the line)



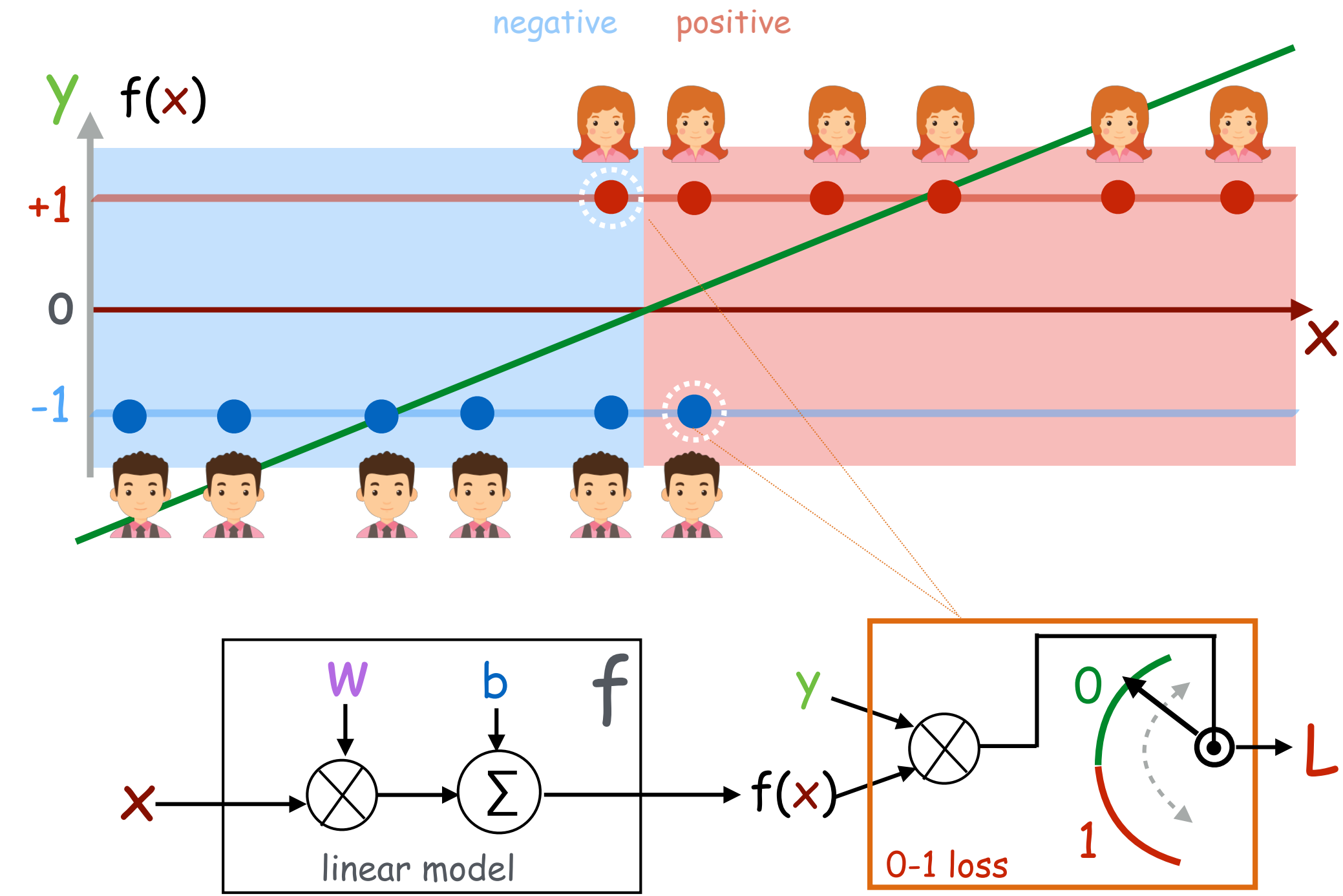
slow in training

low memory cost

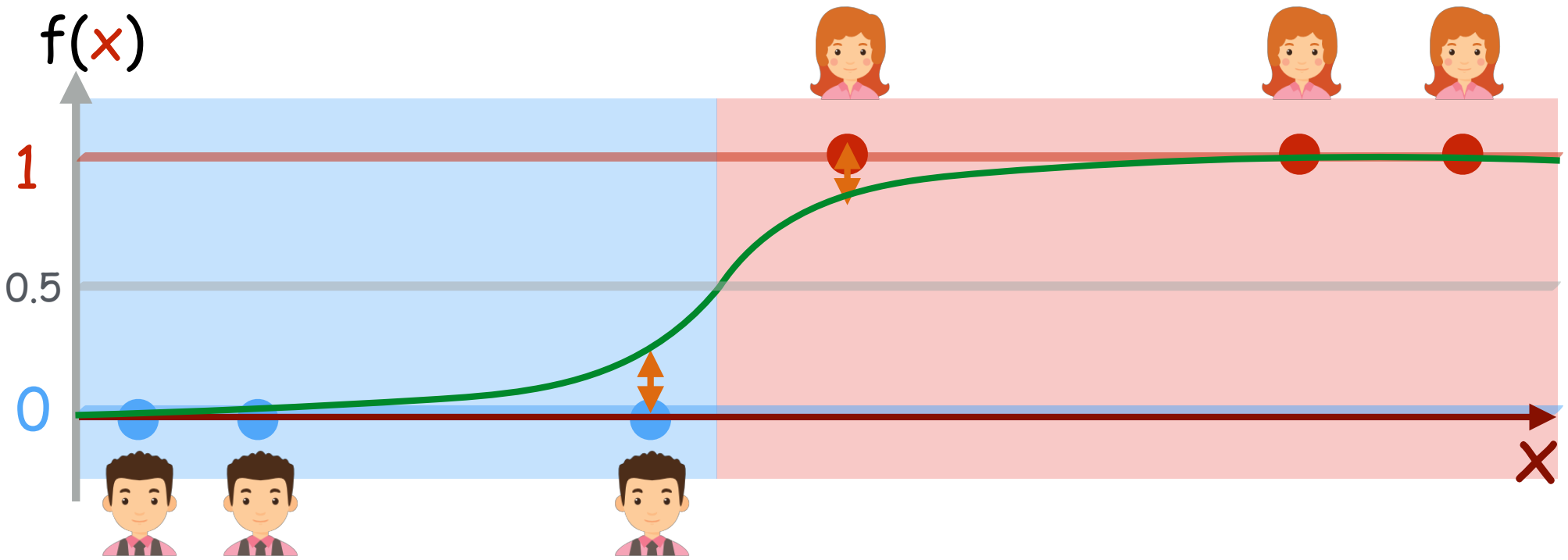
fast inference with large dataset

O(p) for each test data

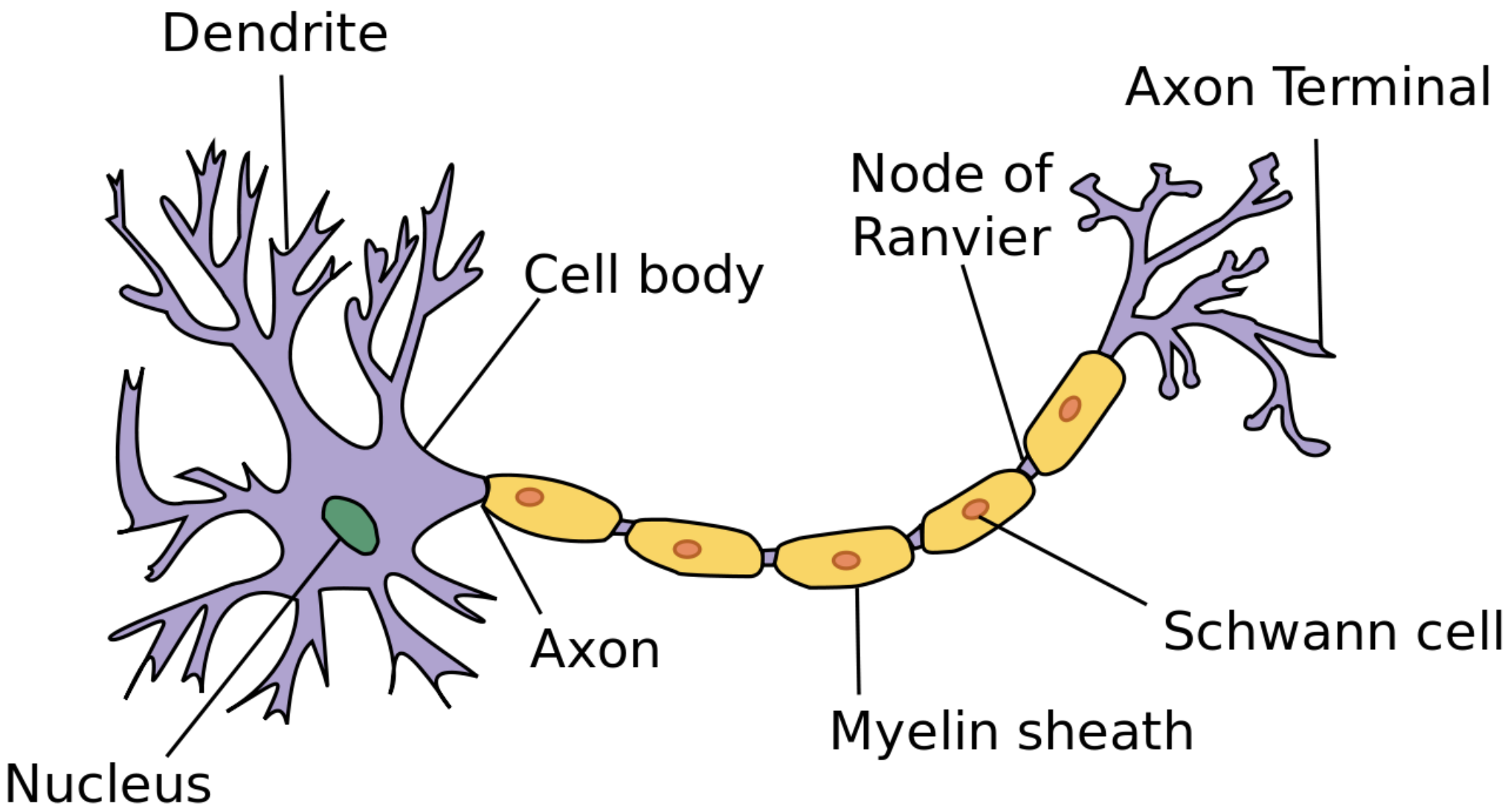
Ideally minimize 0-1 Loss



Logistic Regression

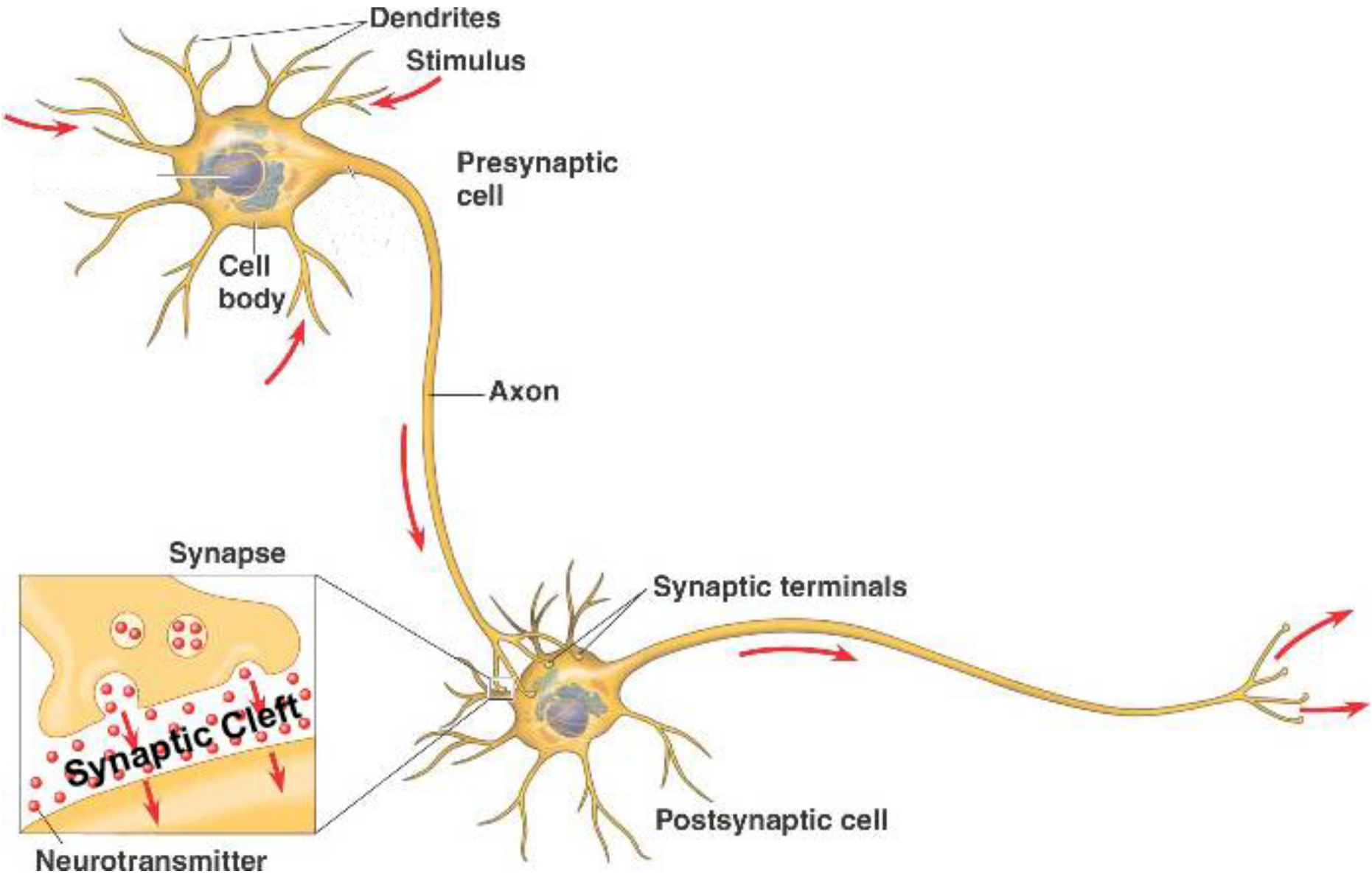


Neuron



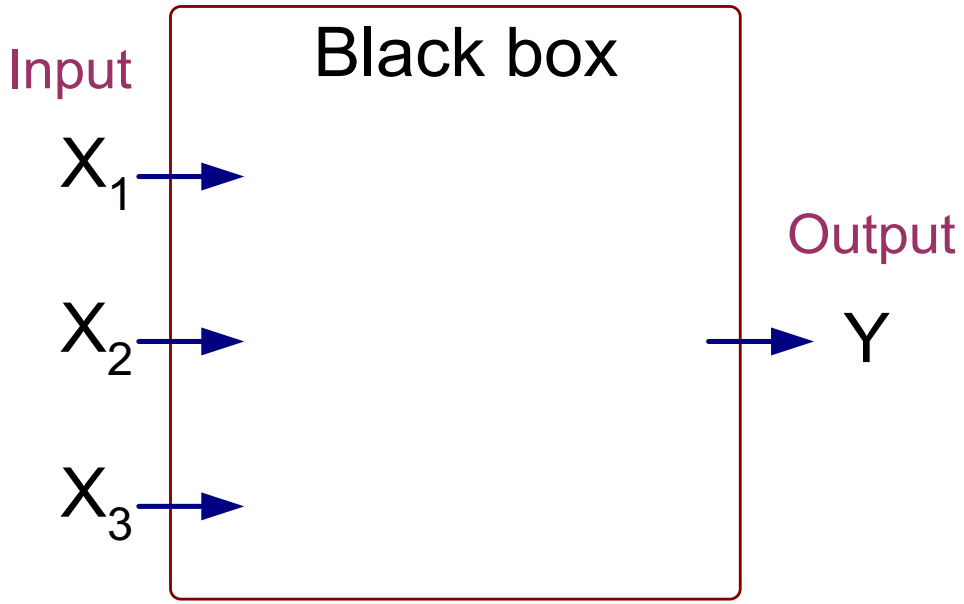
https://www.youtube.com/watch?v=fHRC8SlLcH0

Neuron


Dendrites
Stimulus
Presynaptic cell
Cell body
Axon
Synapse
Synaptic terminals
Synaptic Cleft
Postsynaptic cell
Neurotransmitter

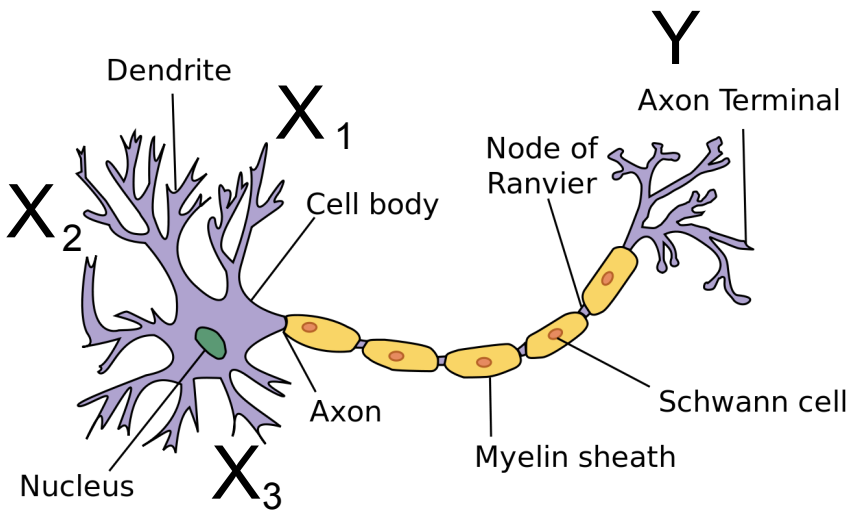
Neuron

X_1	X_2	X_3	Y
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1
0	0	1	0
0	1	0	0
0	1	1	1
0	0	0	0

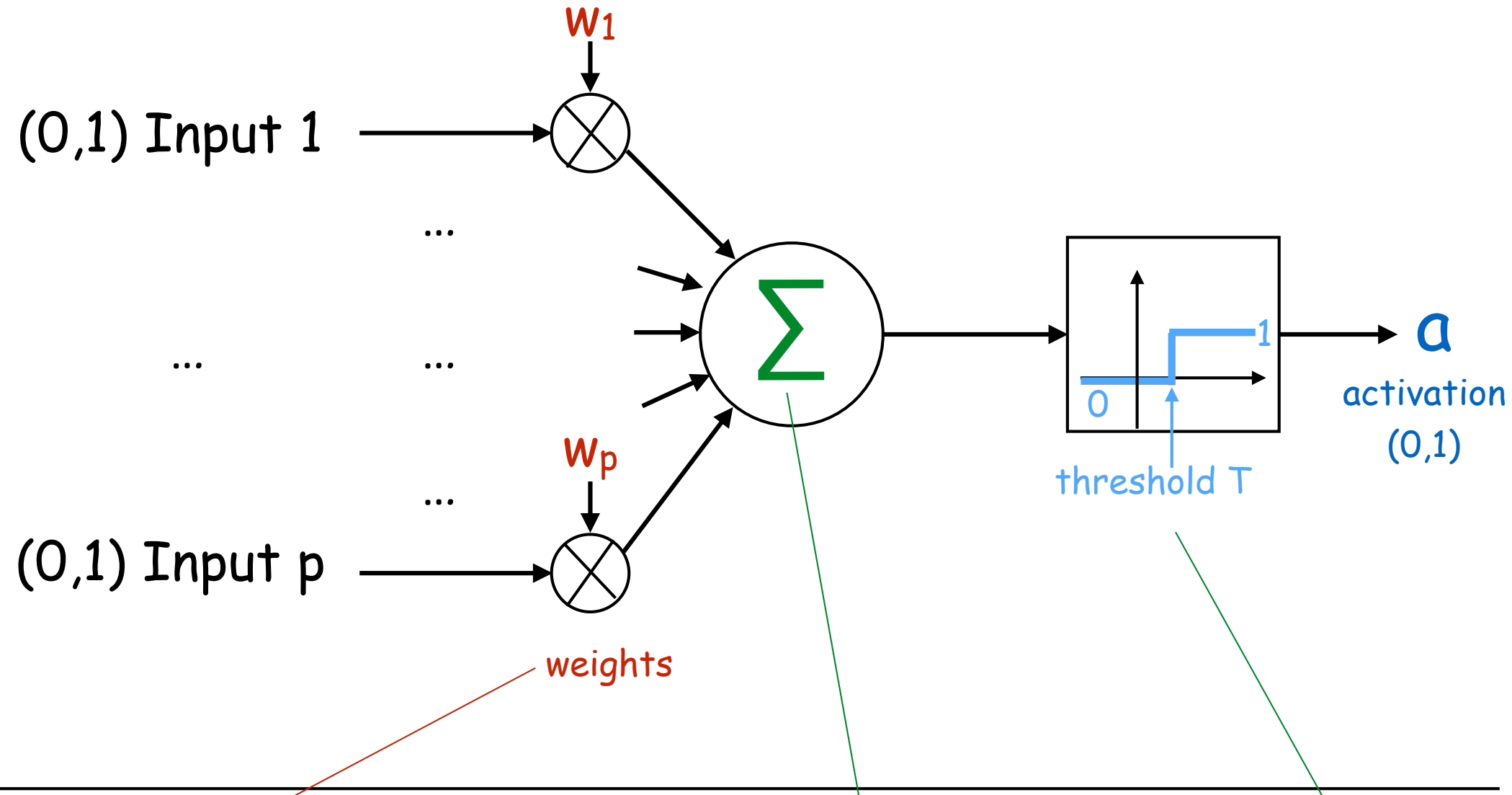


Properties:

1. Synaptic Weights
2. Cumulative Stimulus
3. All or None



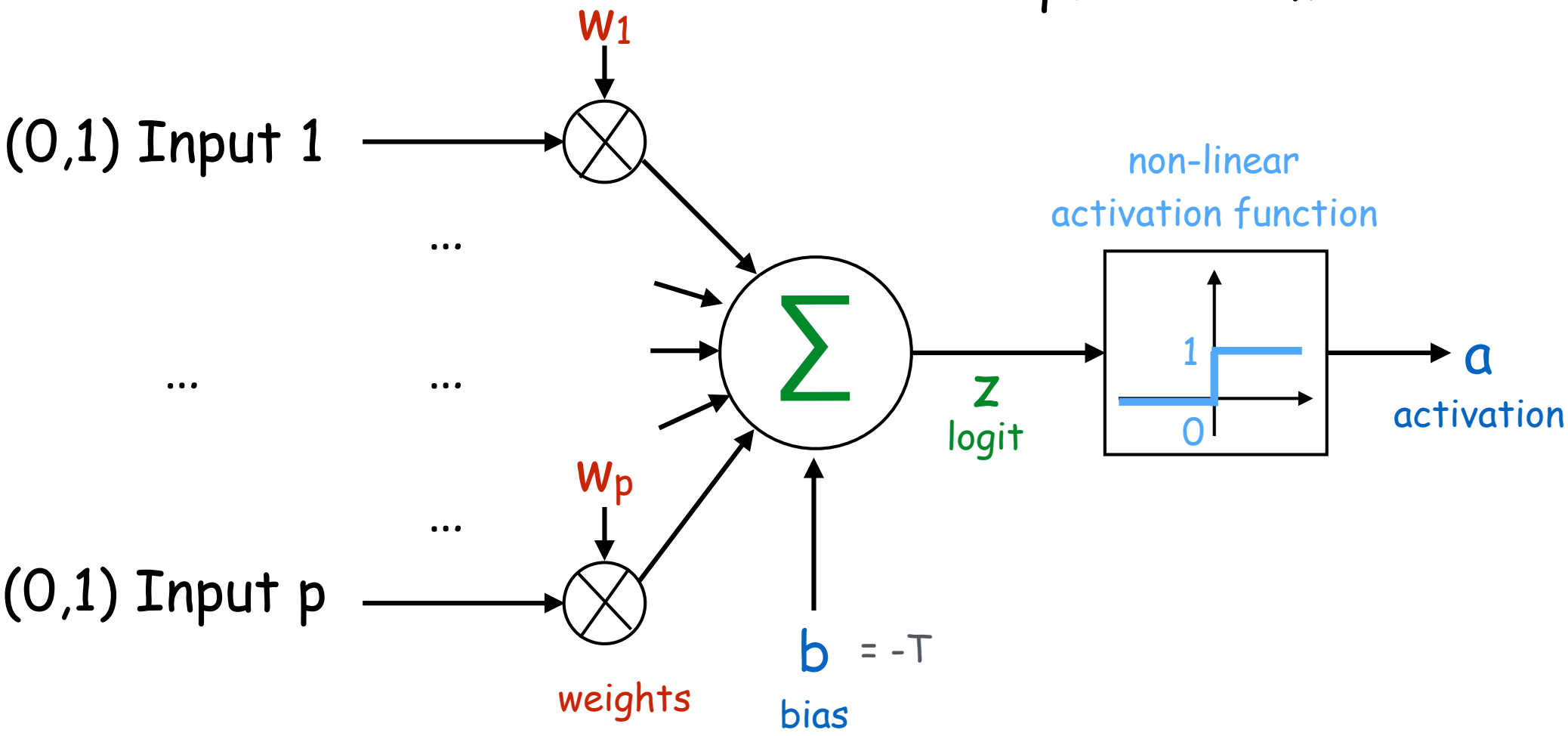
Neuron



1. Synaptic Weights 2. Cumulative Stimulus 3. All or None

Neuron

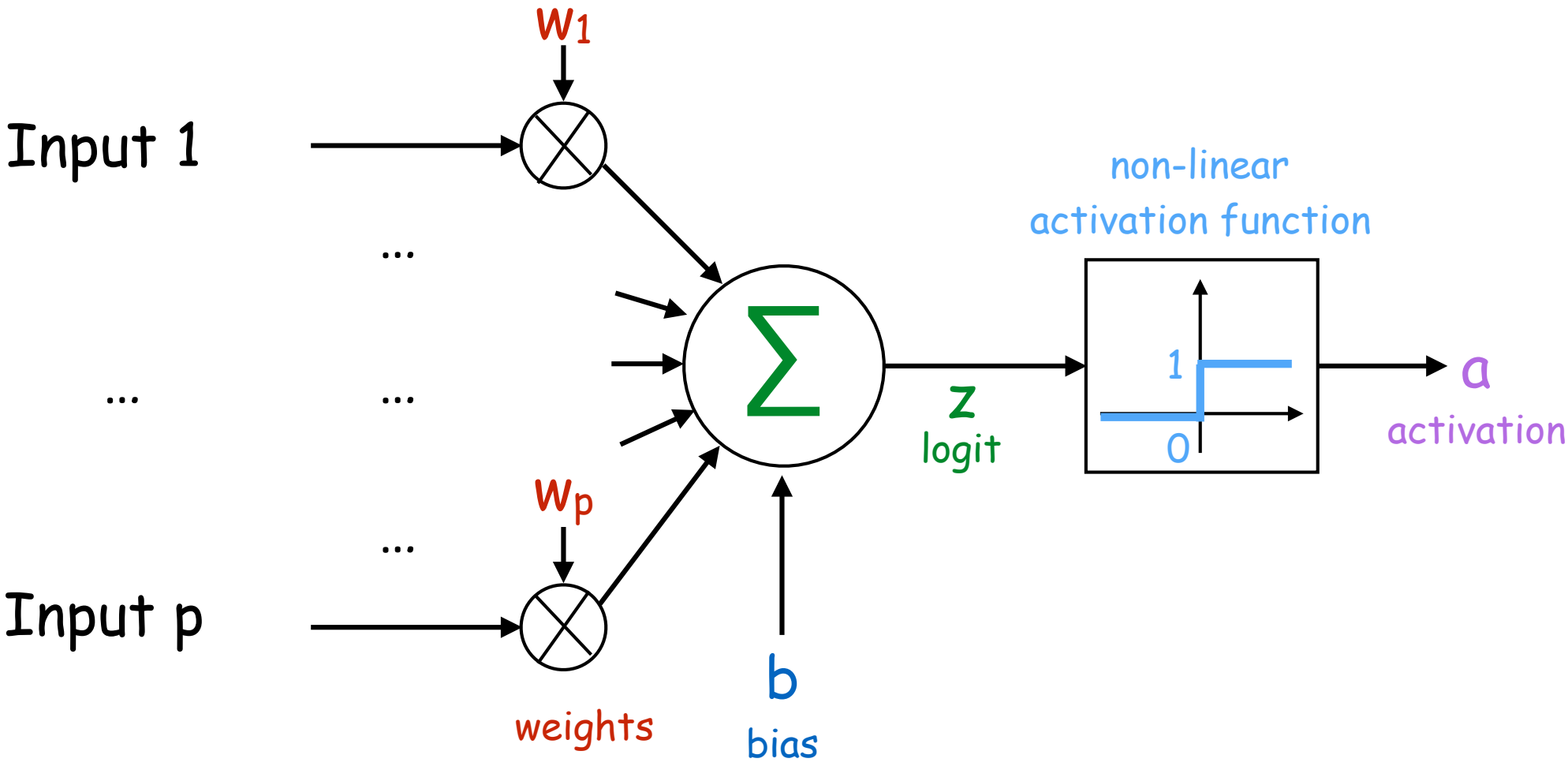
An equivalent model



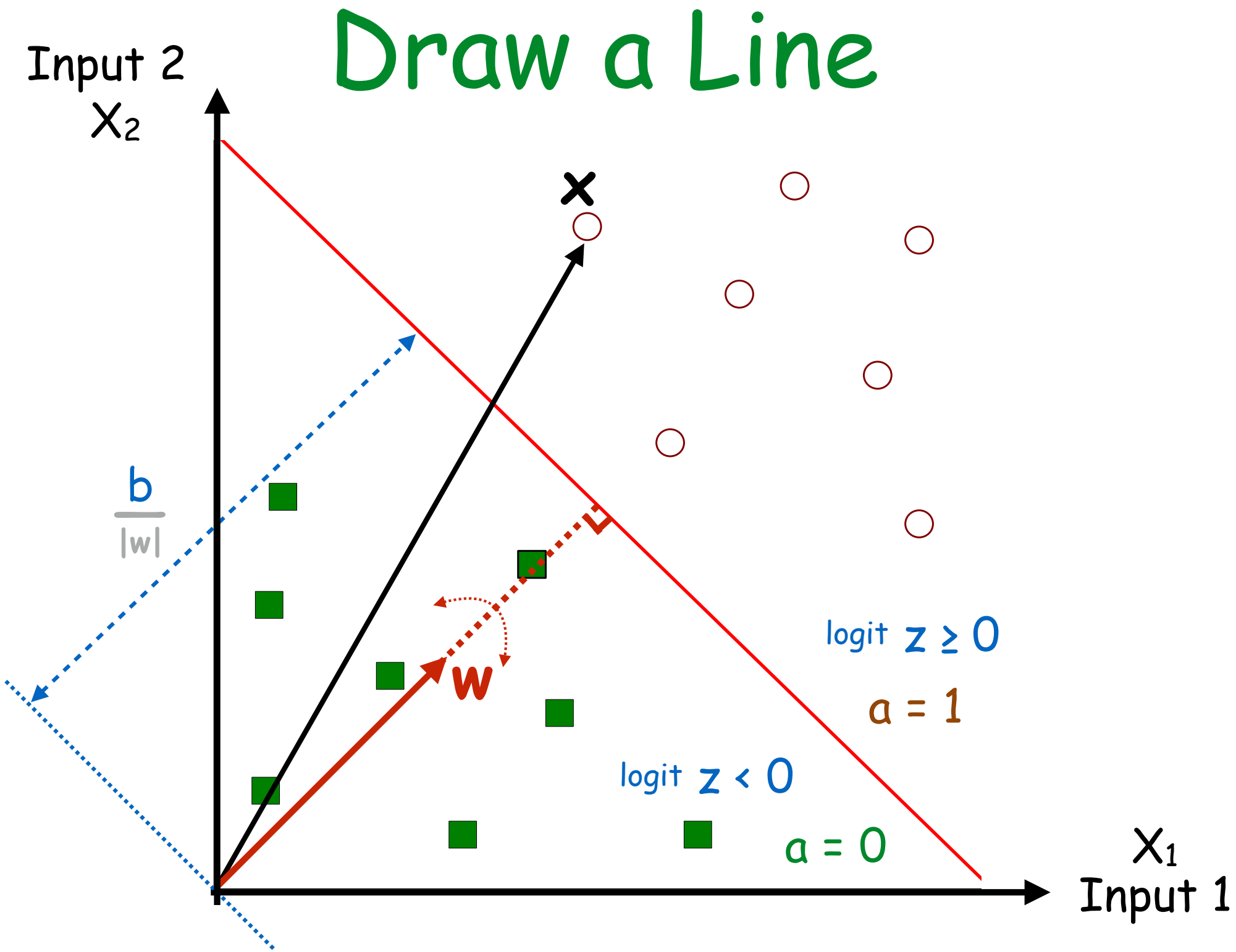
$$z = w_1 x_1 + \ldots + w_p x_p + b \longrightarrow \begin{array}{ll} \text{if } z \geq 0 & a = 1 \\ \text{if } z < 0 & a = 0 \end{array}$$

Neuron

Generalize to continuous input

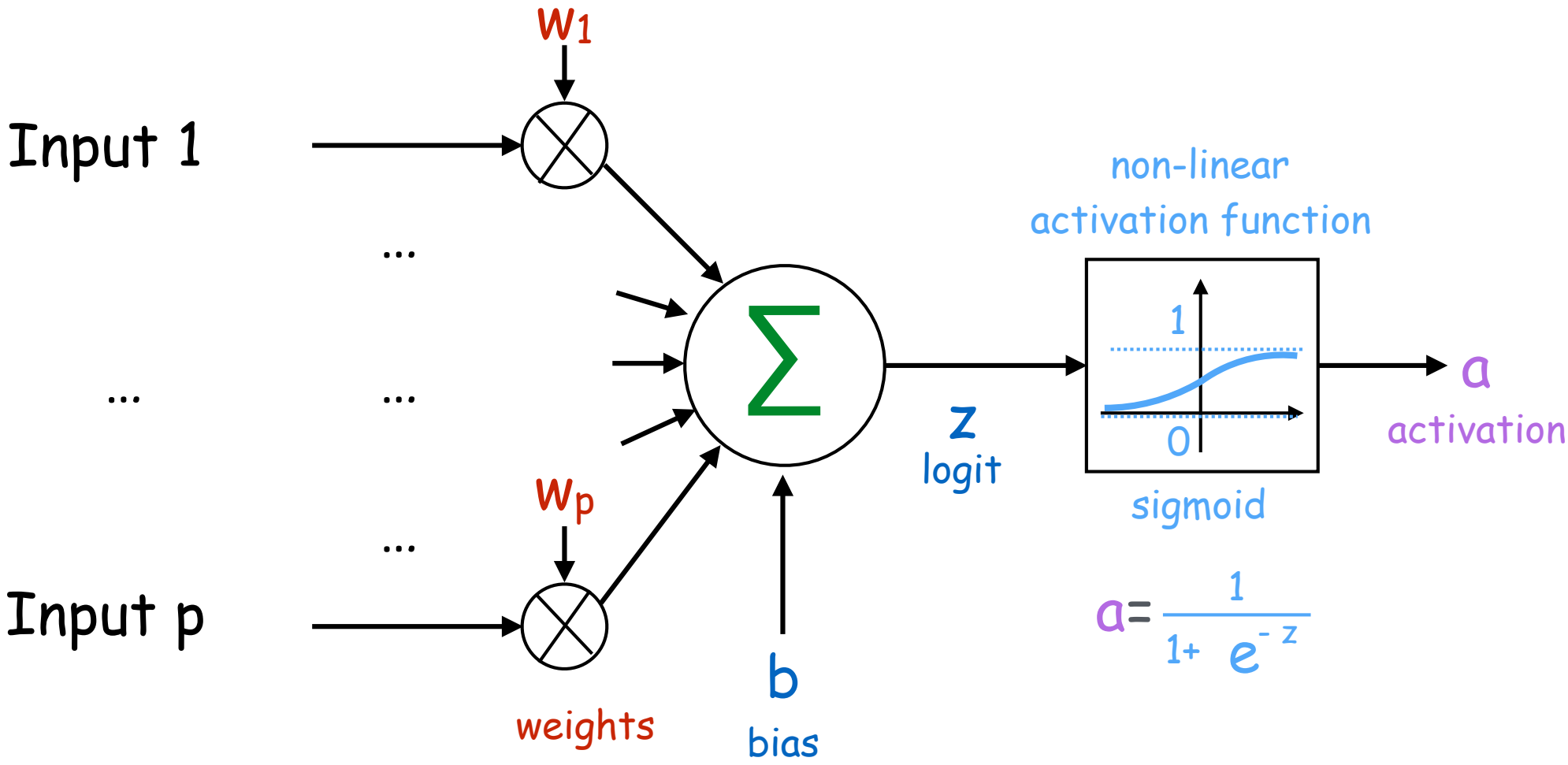


$$z = w_1 x_1 + \ldots + w_p x_p + b \longrightarrow \begin{array}{ll} \text{if } z \geq 0 & a = 1 \\ \text{if } z < 0 & a = 0 \end{array}$$

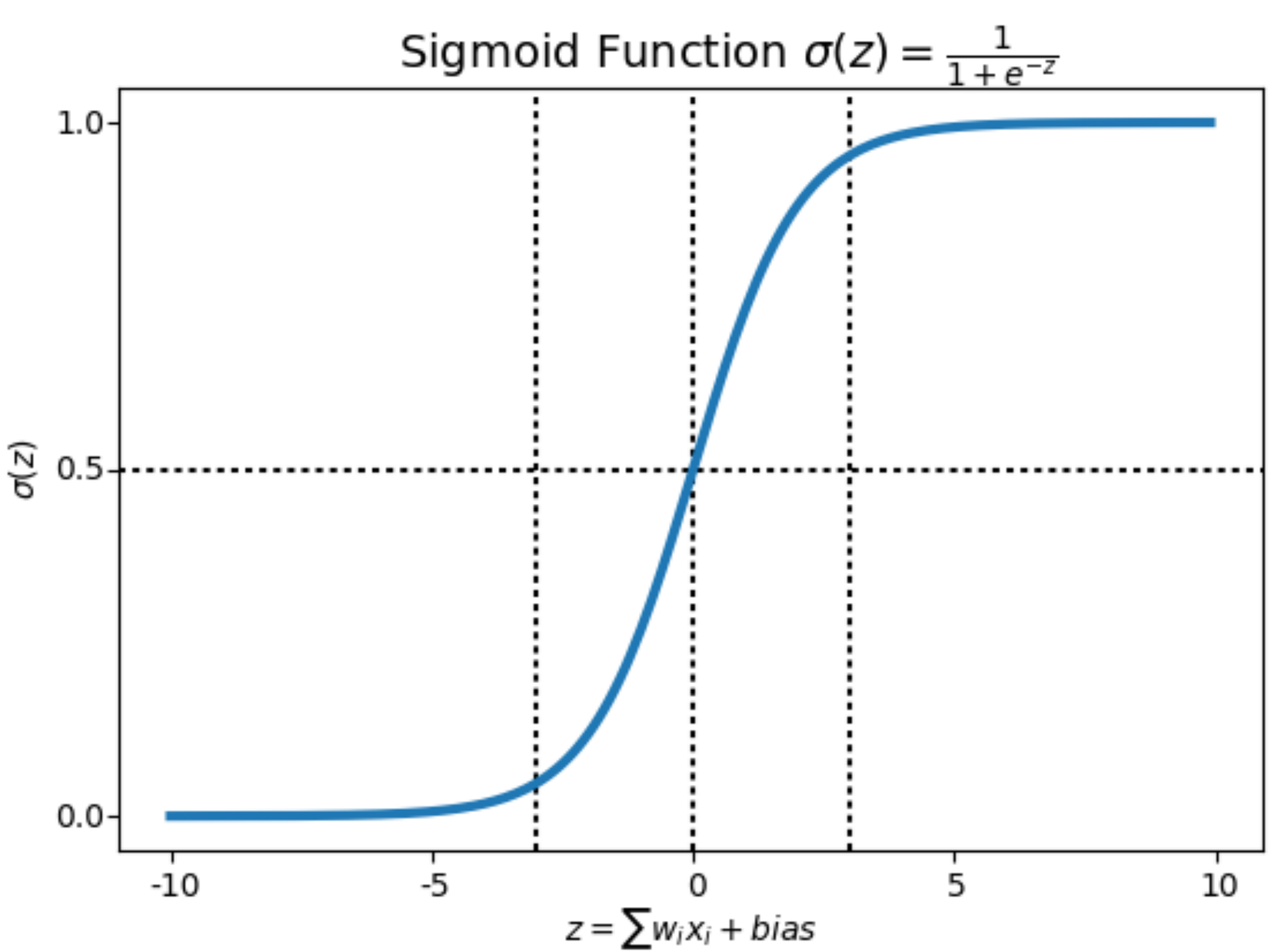

Draw a Line
Input 2
X_2
$\frac{b}{|w|}$
w
logit z ≥ 0
a = 1
logit z < 0
a = 0
X_1
Input 1

Neuron

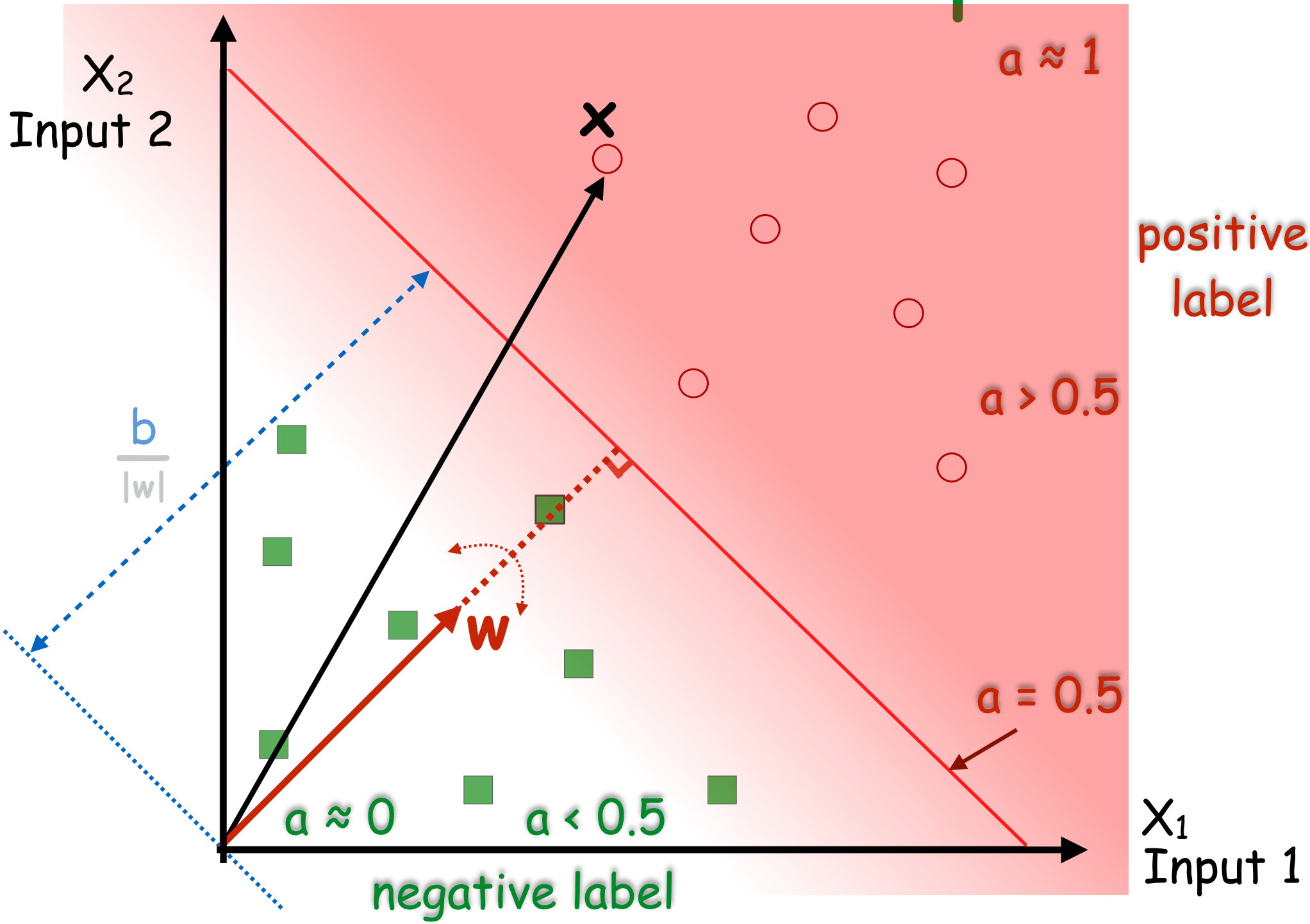
Generalize to continuous output



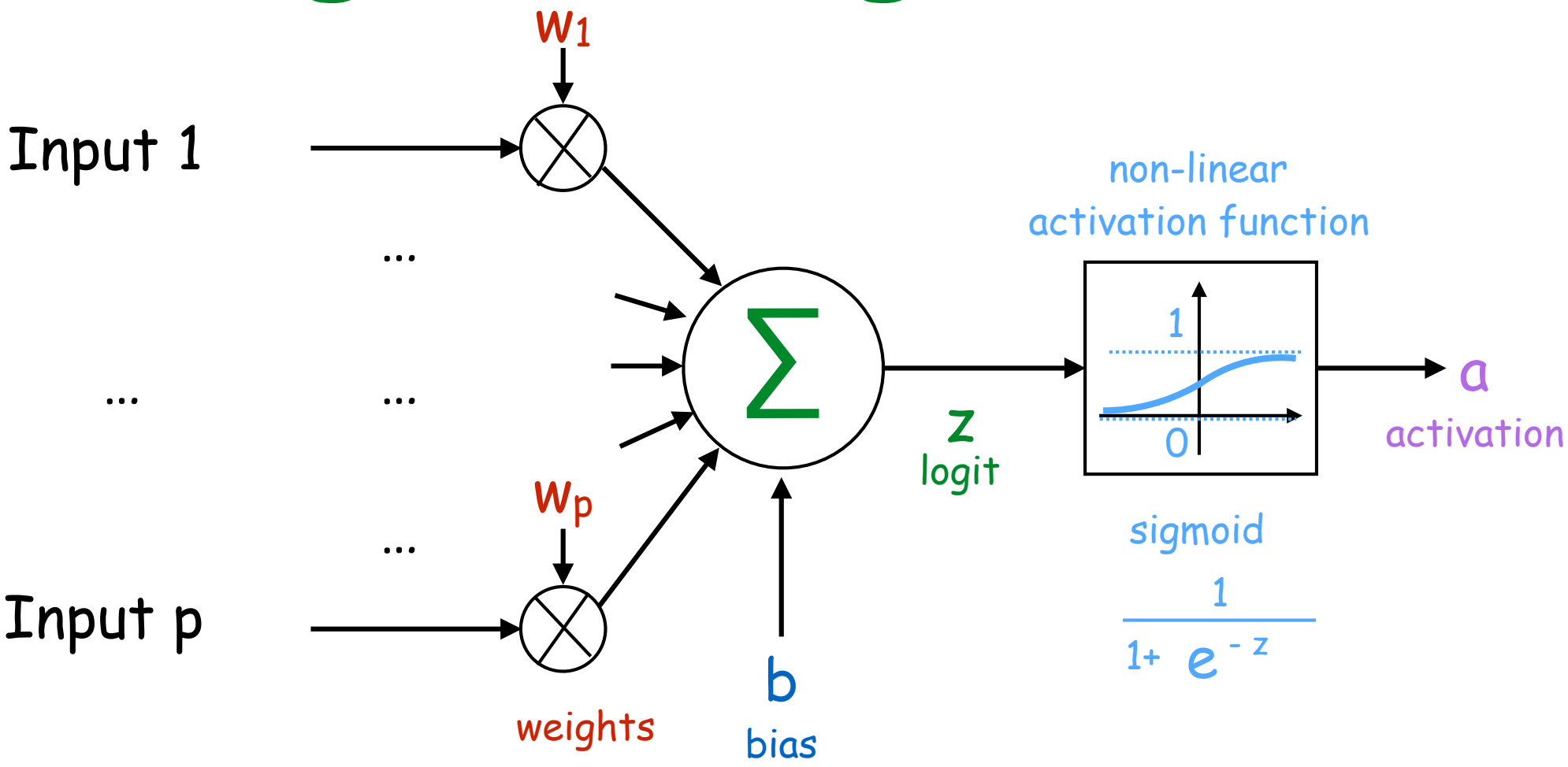
$$w_1 x_1 + \ldots + w_p x_p + b = z \longrightarrow a = f(z)$$


Sigmoid Function $\sigma(z) = \frac{1}{1+e^{-z}}$
1.0
0.5
0.0
$\sigma(z)$
-10
-5
0
5
10
$z = \sum w_i x_i + bias$

Continuous Output



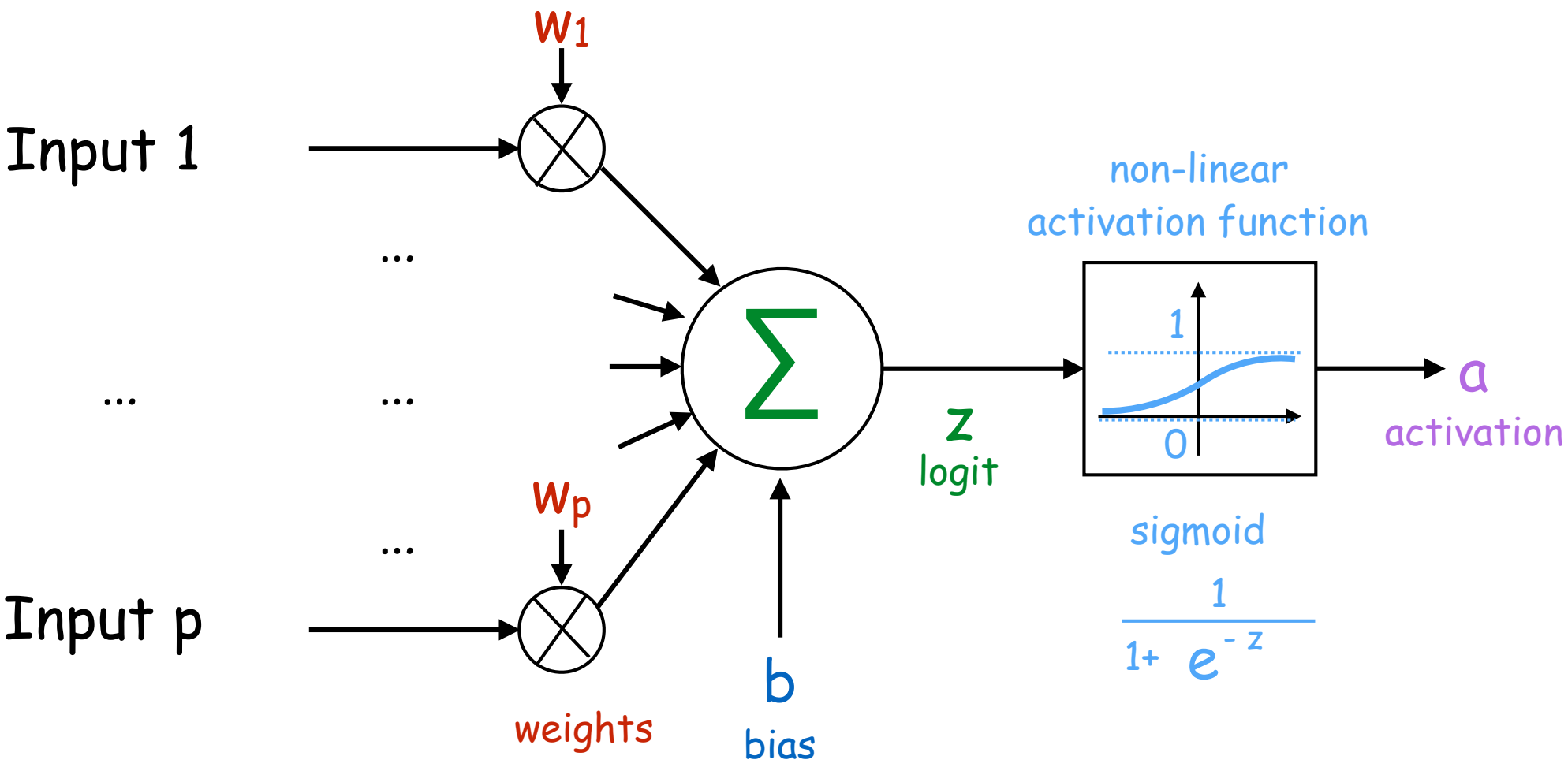
Logistic Regression



$$a = Pr(\text{label} = + \mid \text{inputs} = x)$$

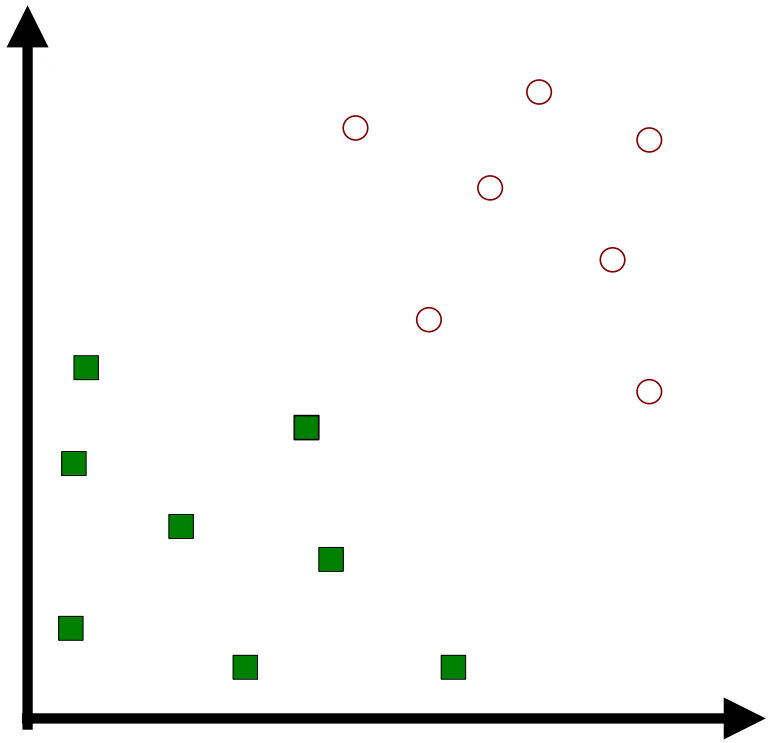
output value represents the probability of the instance having positive label

Parameters w, b



Training Model

training set

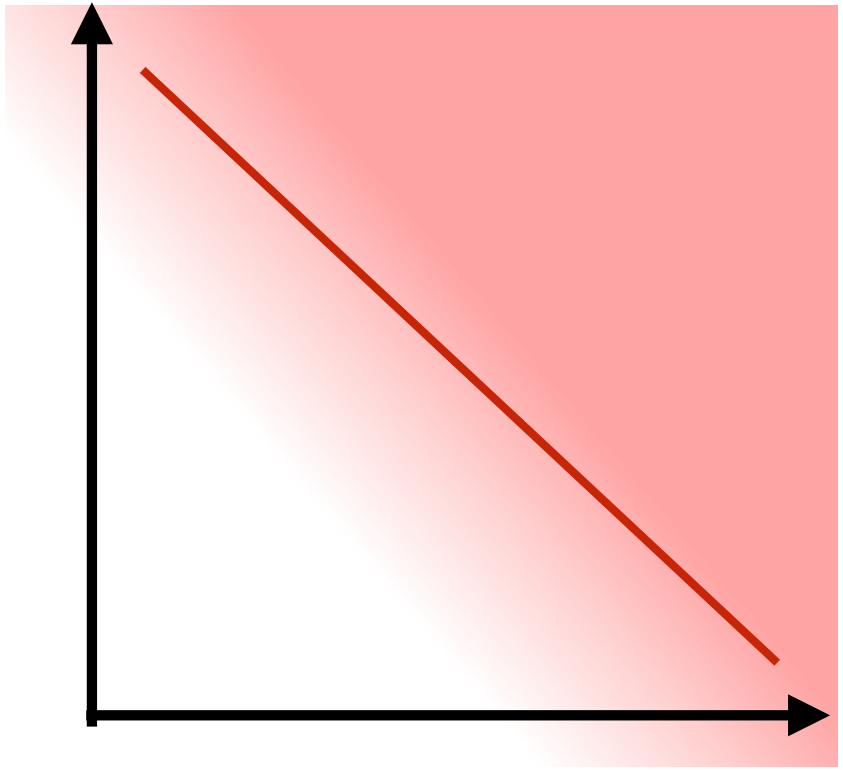


Feature Matrix $X_{(n \text{ by } p)}$

Label Vector Y

(length n) (0 or 1 value)

good parameters

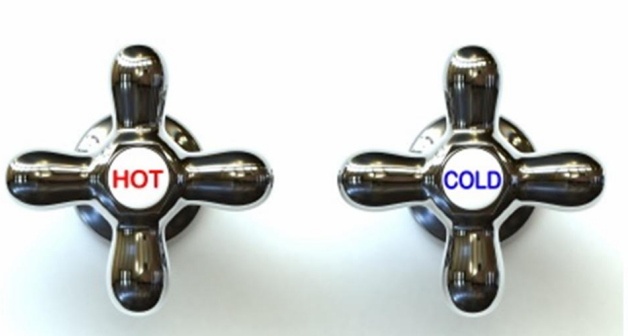


Weight vector **w** (length p)

bias b (scalar)

What are good parameters?

define "good"



X_1	X_2	good?
0	0	bad (no water)
0	5	ok (some water, but cold)
3	0	ok (some water, but hot)
3	5	better (some water, warm)
6	10	great (more water, warm)
0	10	bad (too hot)

Good Parameter Value?

training set

$A = ?$

probability of head

Parameter value 1: An unbiased coin

50%

P(H | A)

50%

P(T | A)

$A = 0.5$

parameter value of Coin 1

Parameter value 2: A biased coin

80%

P(H | A)

20%

P(T | A)

$A = 0.8$

parameter value of Coin 2

Probability of observing the result — Likelihood

observed result

T

H

Coin 1 (A=0.5):

Likelihood =
0.5 X 0.5 = **0.25**

Coin 2: (A= 0.8)

Likelihood =
0.2 X 0.8 = **0.16**

Probability of observing the result — Likelihood

observed result

Coin 1 (A=0.5):

Likelihood =

0.5 X 0.5 X 0.5 = **0.125**

Coin 2: (A= 0.8)

Likelihood =

0.2 X 0.8 X 0.8 = **0.128**

Probability of observing the result — Likelihood

observed result

Coin 1 (A=0.5):

Likelihood =

0.5 X 0.5 X 0.5 X 0.5 = **0.0625**

Coin 2: (A= 0.8)

Likelihood =

0.2 X 0.8 X 0.8 X 0.8 = **0.1024**

Maximum Likelihood

Higher likelihood —> better parameter

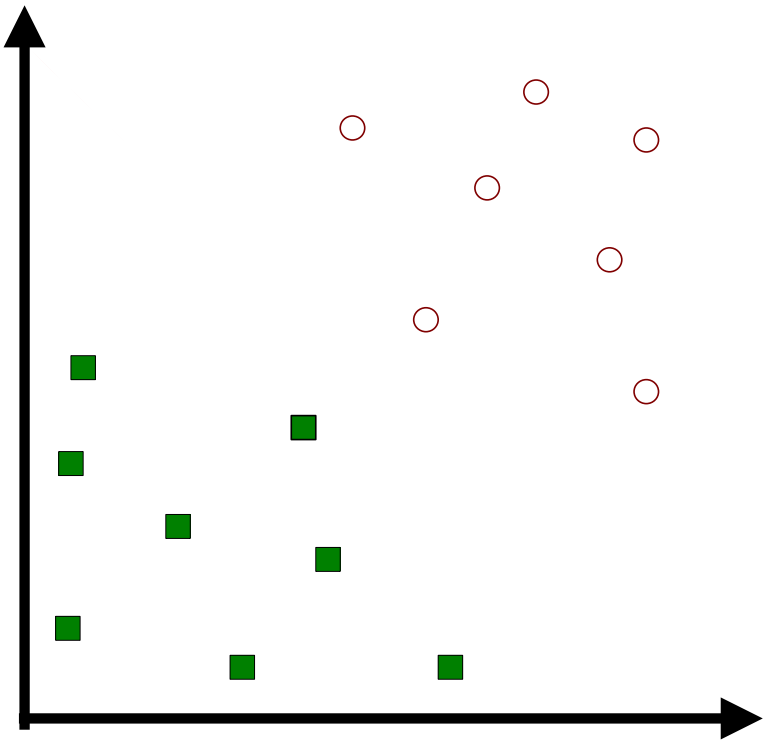
training set

good parameter value

$$A^* = \text{argmax } L(A)$$

Conditional Likelihood

observed labels

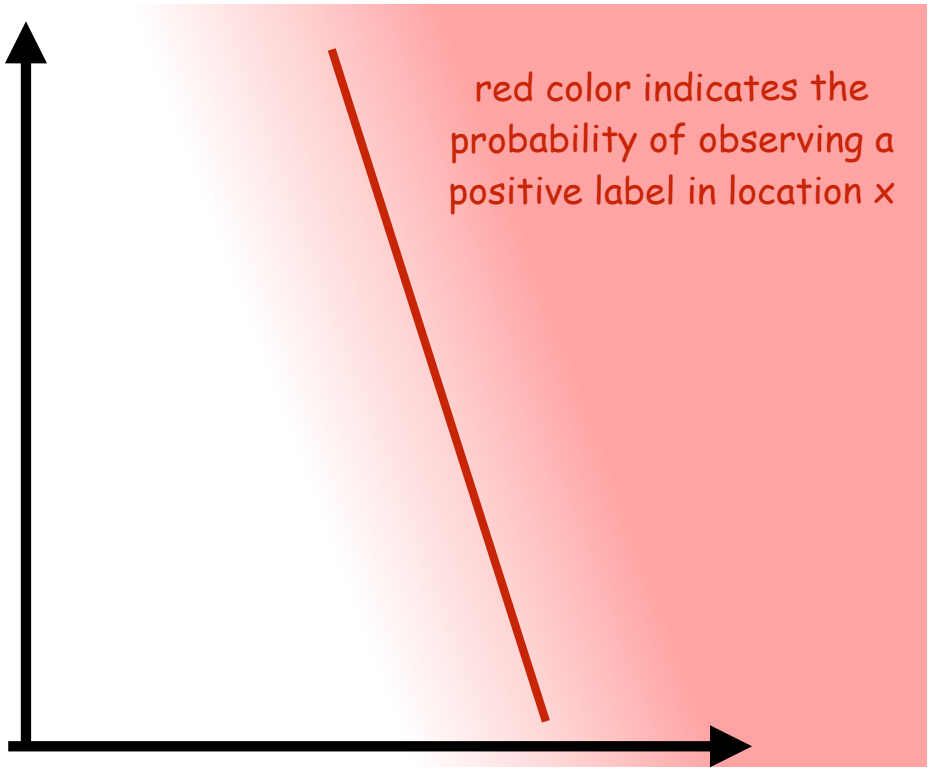


Feature Matrix $X_{(n \text{ by } p)}$

Label Vector y

(length n) (0 or 1 value)

parameter value



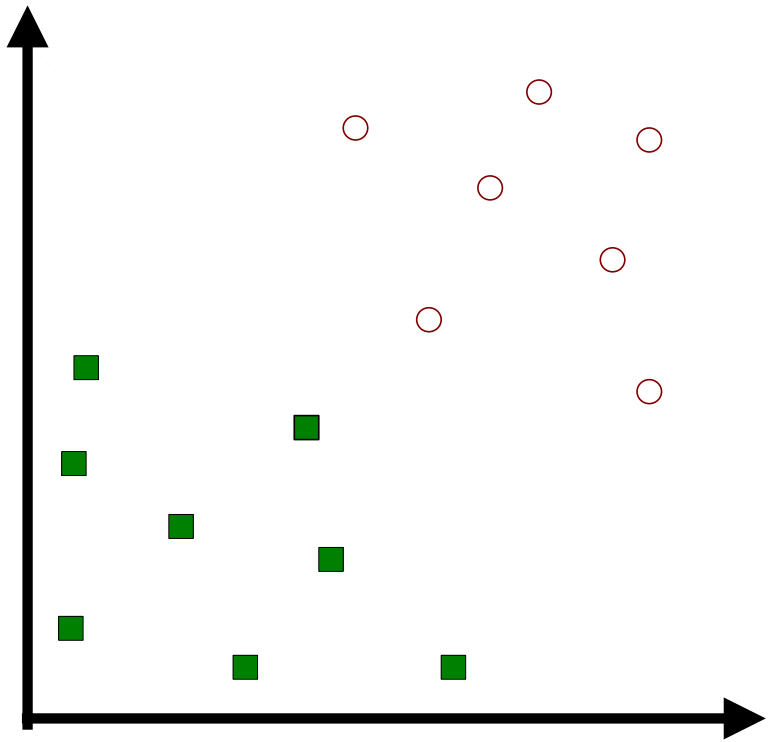
$Pr(label = + \mid x, w, b)$

Weight vector **w** (length p)

bias b (scalar)

Conditional Likelihood

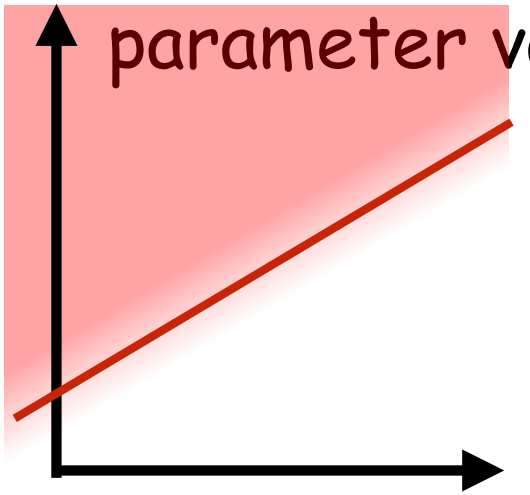
observed labels

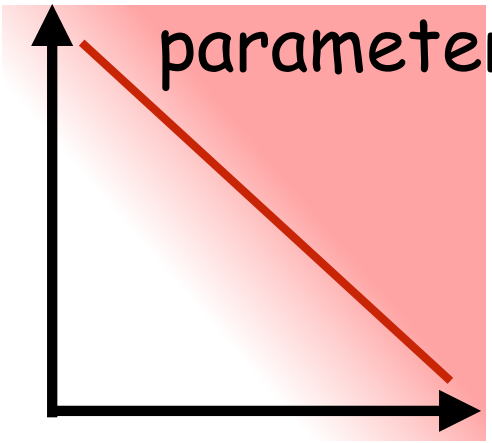


Feature Matrix $X_{(n \text{ by } p)}$

Label Vector y

(length n) (0 or 1 value)





Weight vector **w** (length p)

bias b (scalar)

Minimize negative log Likelihood

Coin 1 (A=0.5):

- log Likelihood =

  - log (0.5 X 0.5 X .... X 0.5) = - (log 0.5 + log 0.5 + ... + log 0.5)

    = **13.8629436**

Coin 2: (A= 0.8)

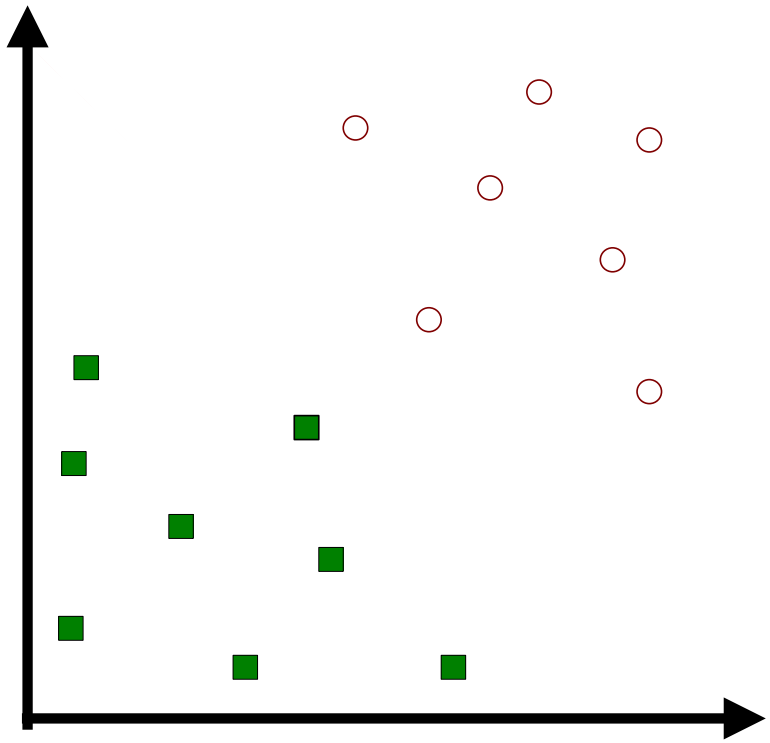
- log Likelihood =

  - log(0.2 X 0.8 X ... X 0.8) = - (log 0.2 + log 0.8 + ... + log 0.8)

    = **4.462871**

Minimize negative log Likelihood

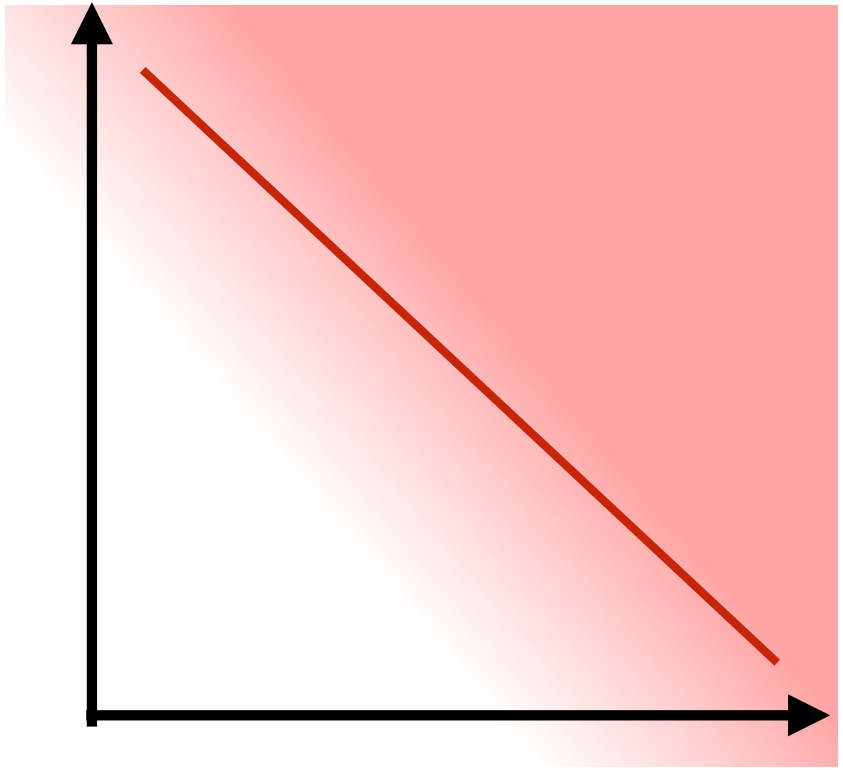
training set



Label Vector Y

(length n) (0 or 1 value)

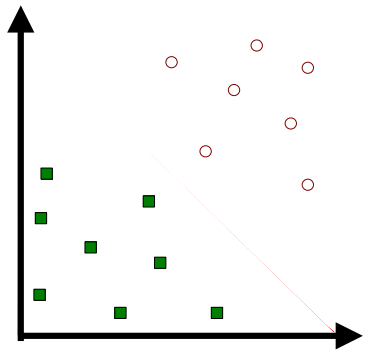
parameters



Weight vector **w** (length p)

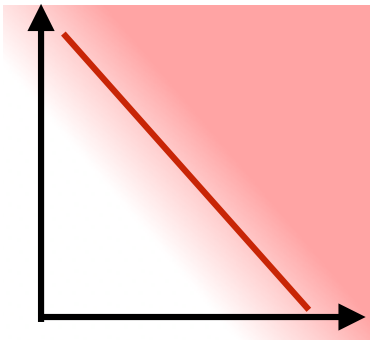
bias b (scalar)

Minimize negative log Likelihood



Labels:
observed data

y_1	y_2	y_3		y_n
1	0	1	...	0



Weight vector **w**
bias b

Outputs:
probabilities

a_1	a_2	a_3		a_n
.8	.3	.6		.2

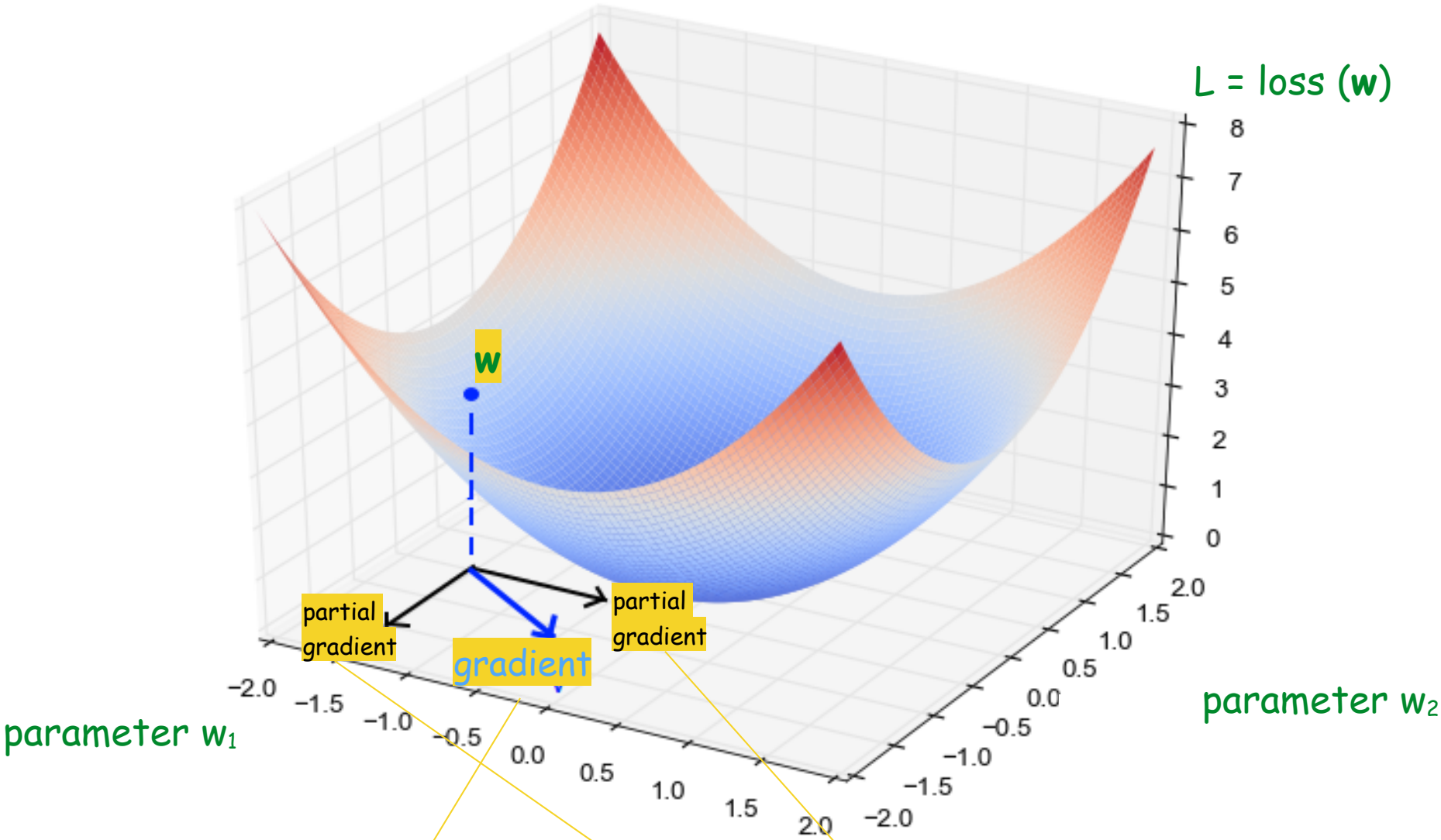
- log Likelihood =

$$\sum_{i=1}^{n} -y_i \log a_i - (1-y_i) \log(1 - a_i)$$

which is also called <u>cross entropy</u> loss

multiplied with a constant factor n

Minimize Cross Entropy Loss

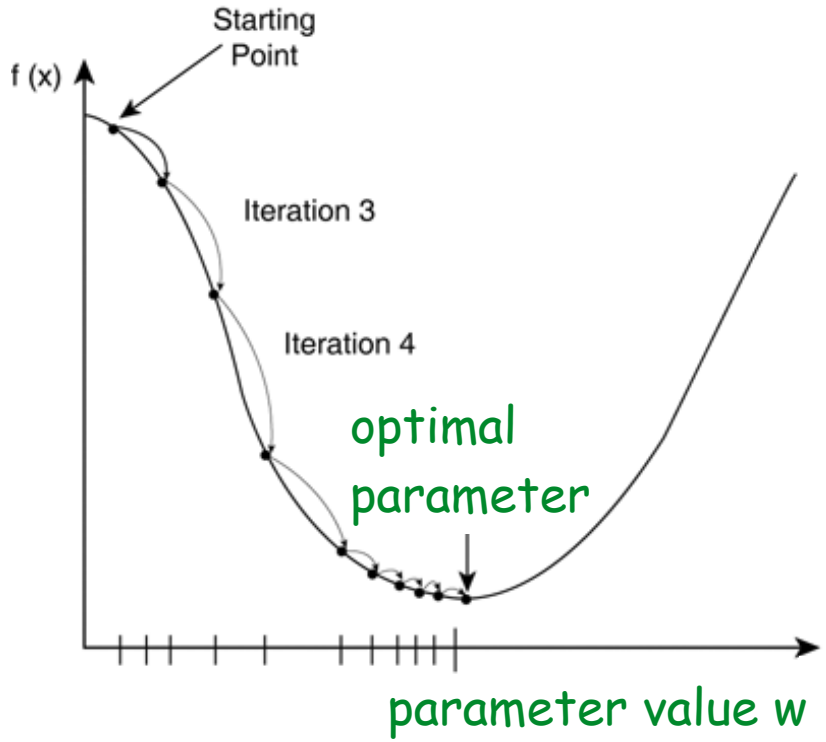


negative gradient of L w.r.t. vector w

$$-\frac{\partial L}{\partial \mathbf{w}} = \left(-\frac{\partial L}{\partial w_1}, -\frac{\partial L}{\partial w_2}\right)$$

Gradient Descent (update model)

L= loss (w)

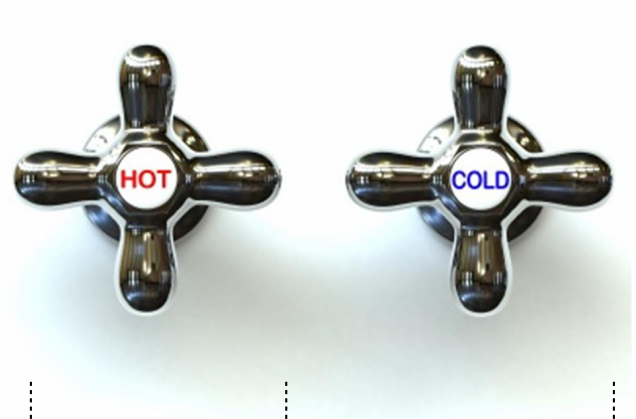


$$w \leftarrow w - a \frac{\partial L}{\partial w}$$

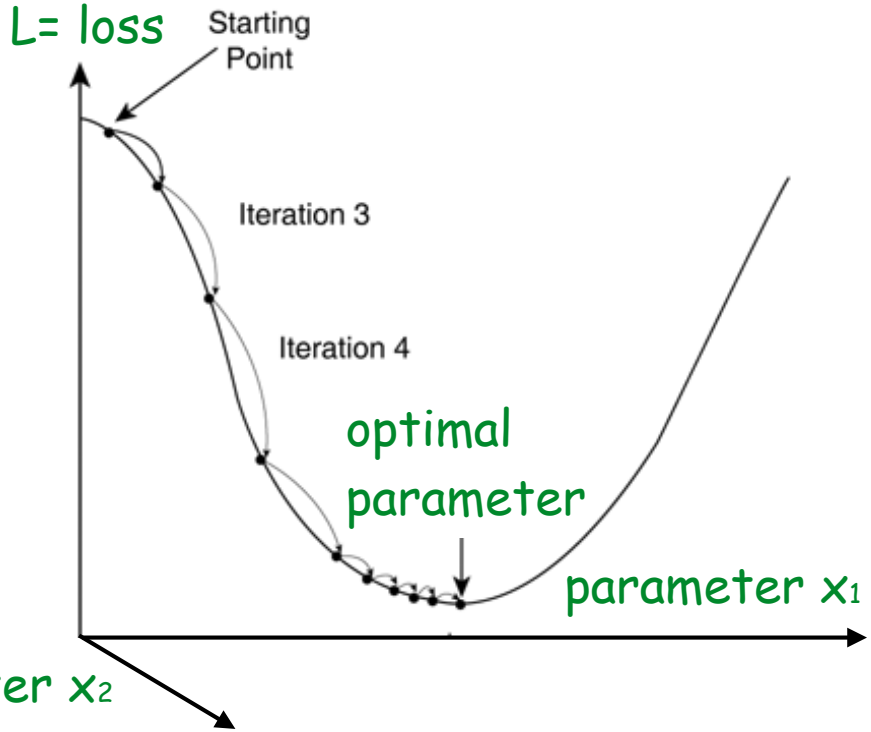
$$b \leftarrow b - a \frac{\partial L}{\partial b}$$

a step size (a constant scalar)

Example



	x_1	x_2	L
1	0	0	100 bad (no water)
2	0	5	80 ok (some water, but cold)
3	3	5	60 better (some water, warm)
4	6	10	20 great (more water, warm)



Gradient

X_1	$\frac{\partial L}{\partial x_1}$	X_2	$\frac{\partial L}{\partial x_2}$	Loss
0	-3	0	-10	100 bad (no water)
0	-6	5	-6	80 ok (some water, but cold)
3	-3	6	-3	60 better (some water, warm)
6	0	10	0	20 great (more water, warm)
8	3	6	-3	60 good (more water, hot)
9	6	3	-6	80 ok (more water, too hot)
10	10	0	-10	200 bad (more water, way too hot)

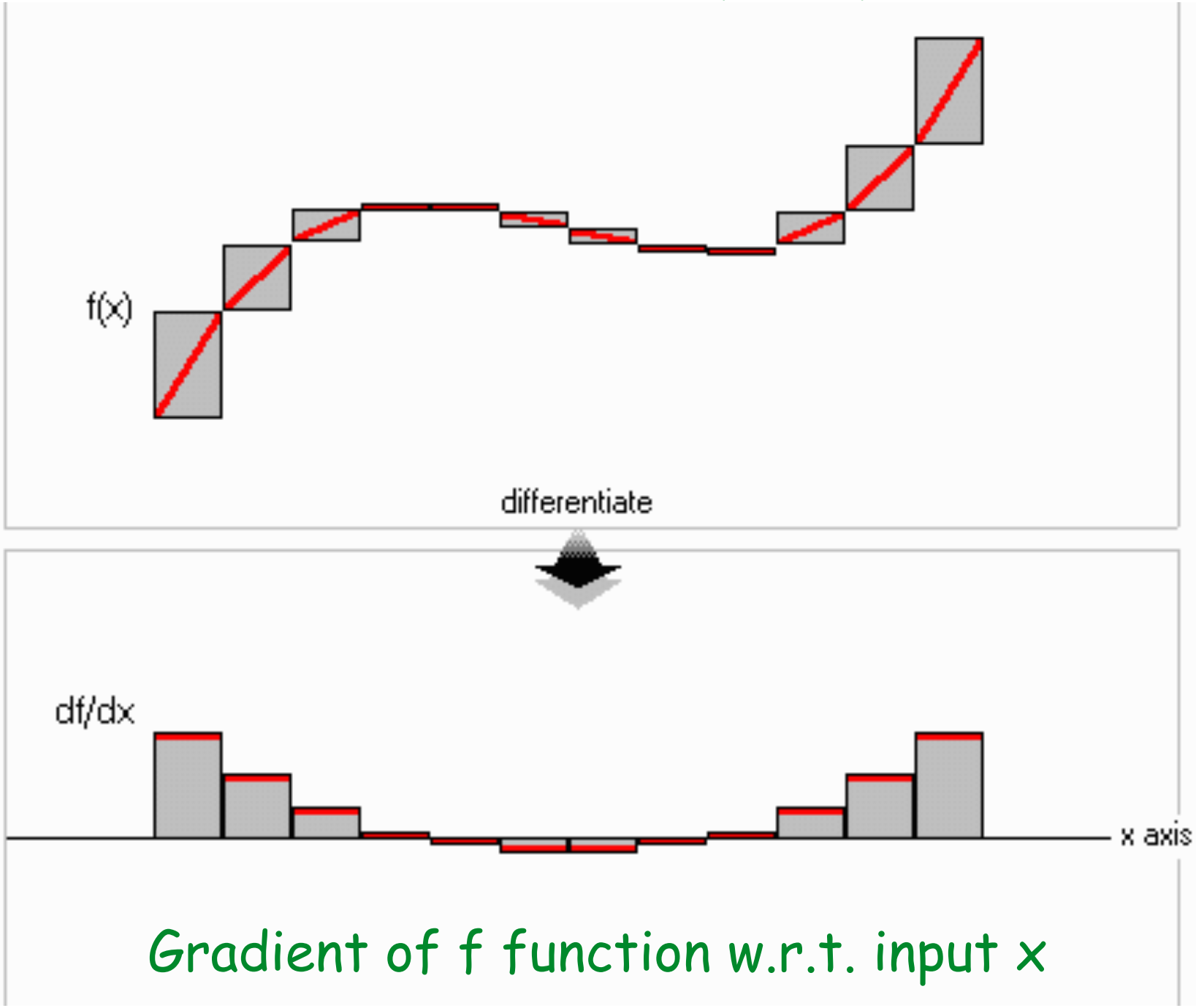
Gradient of L w.r.t. a **scalar**?

$$\frac{\partial L}{\partial b}$$

Gradient of L w.r.t. a **vector**?

$$\frac{\partial L}{\partial \mathbf{w}}$$

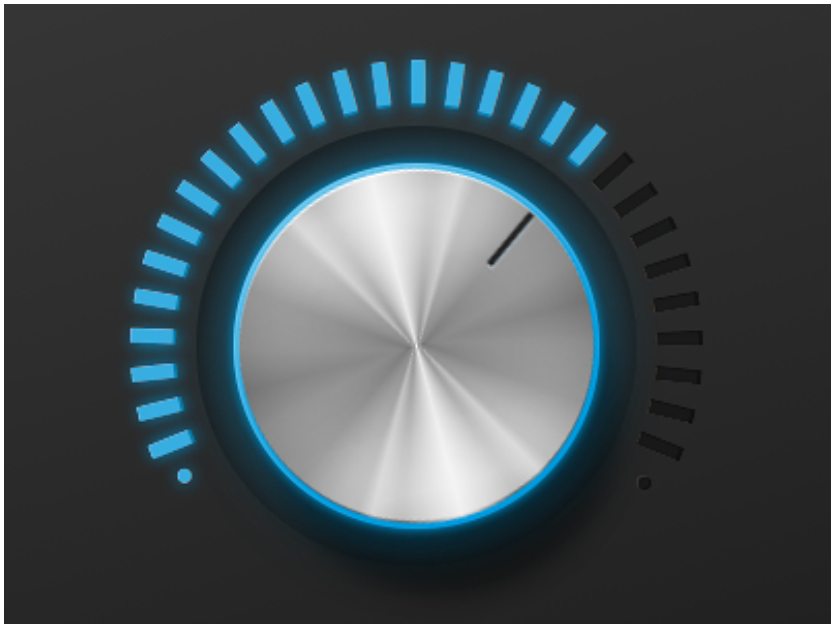
Gradient of a function f with respect to (w.r.t.) a scalar x

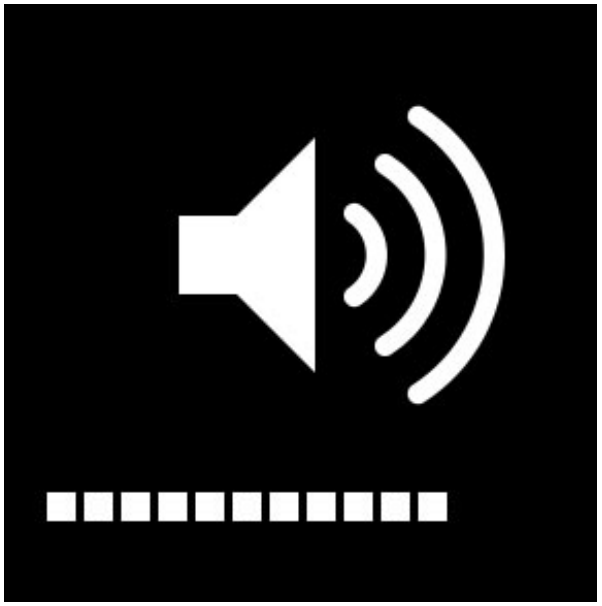


$$\frac{\partial f}{\partial x}$$

Gradient of f function w.r.t. input x

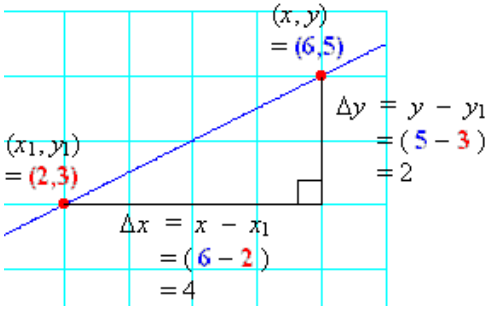
Gradient of L w.r.t. a scalar x





x

control

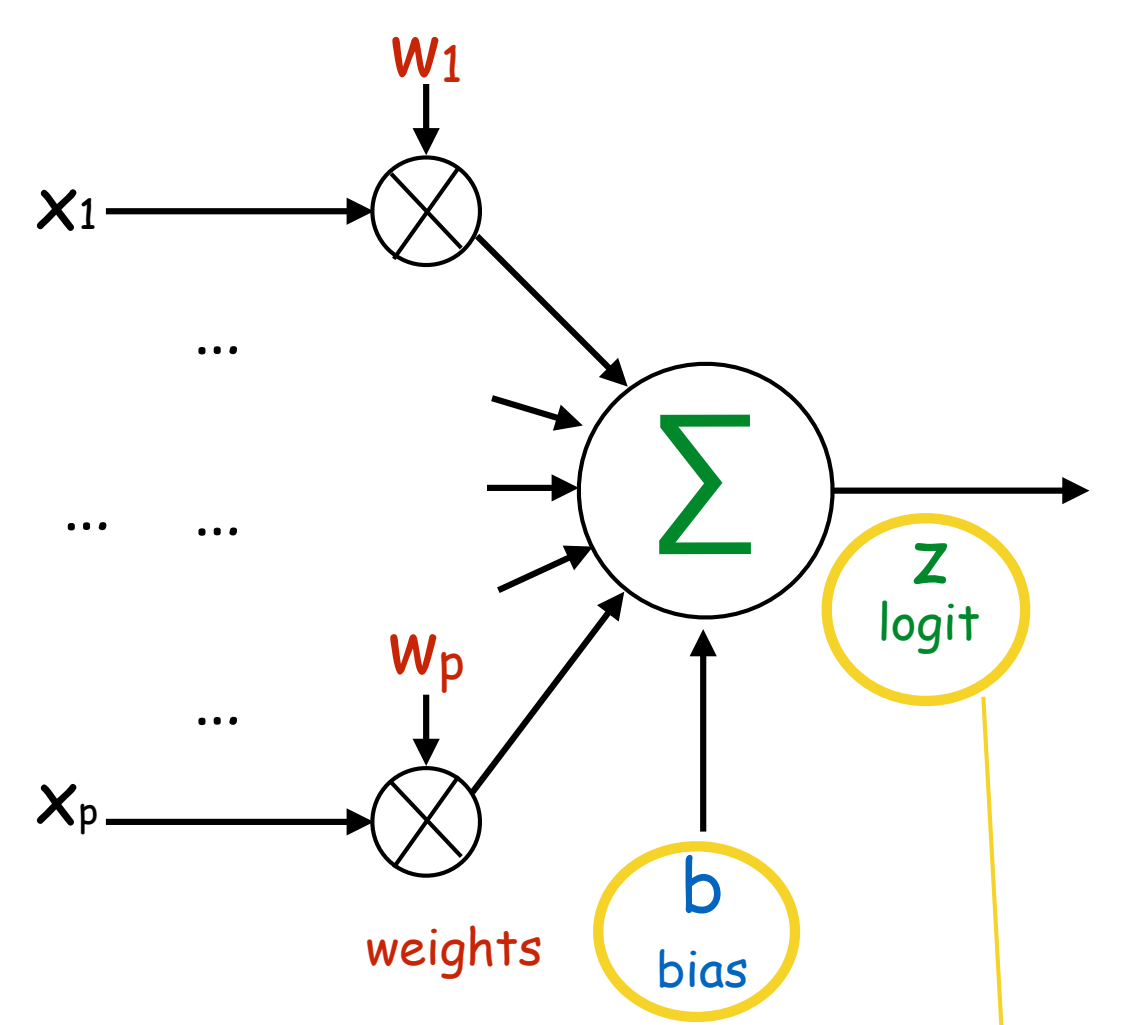


L

response

$\frac{\partial L}{\partial x}$ a constance number

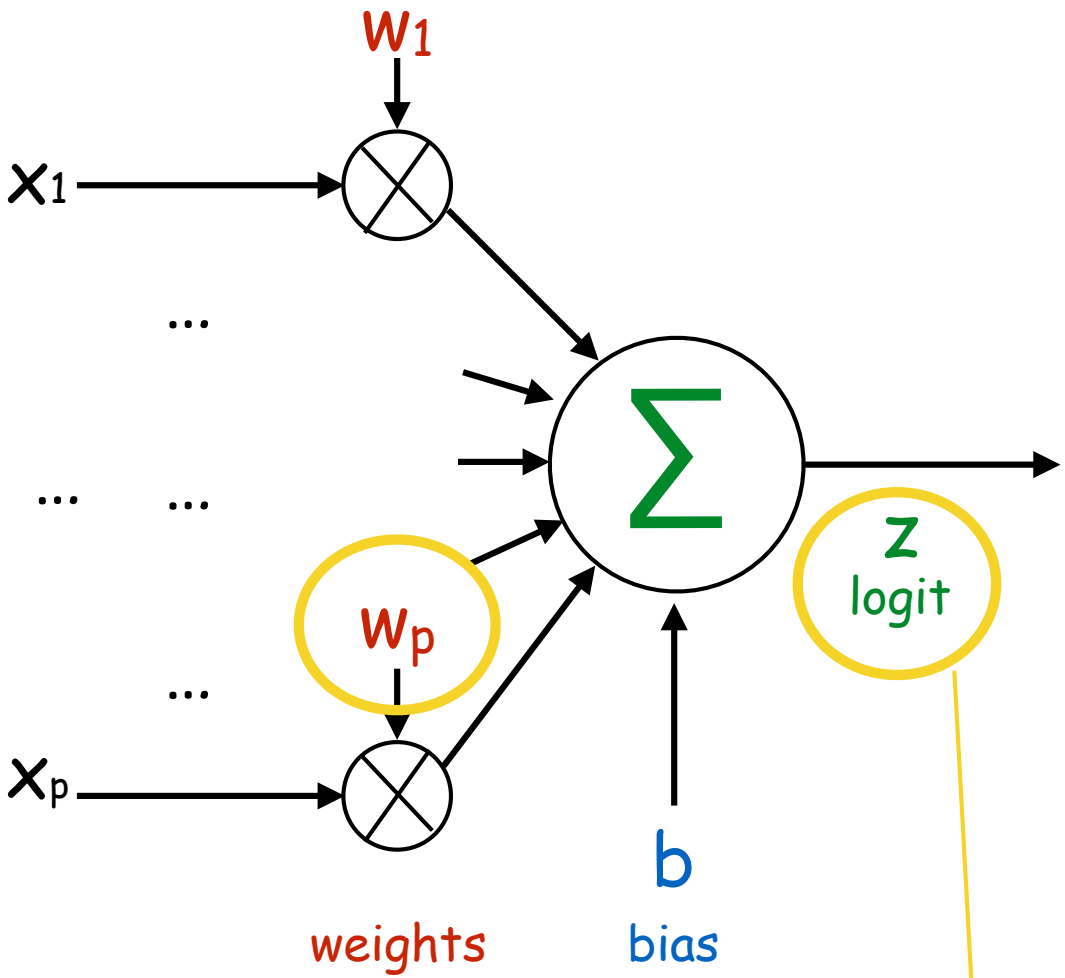
Example



gradient

$$\frac{\partial z}{\partial b} = 1$$

Example



gradient

$$\frac{\partial z}{\partial w_p} = x_p$$

is a function of x_p

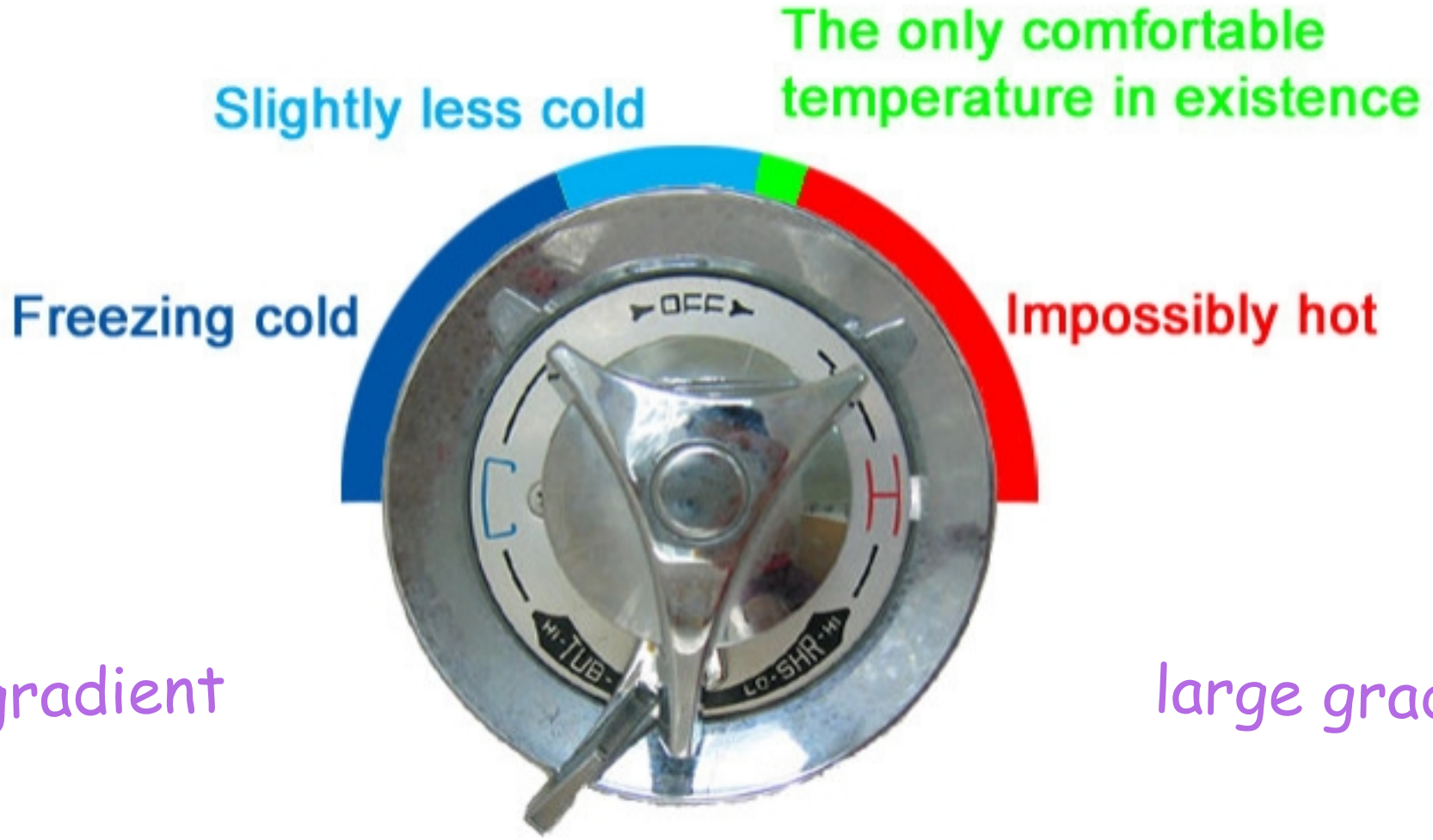
Gradient Changing over x

x

L

water temperature

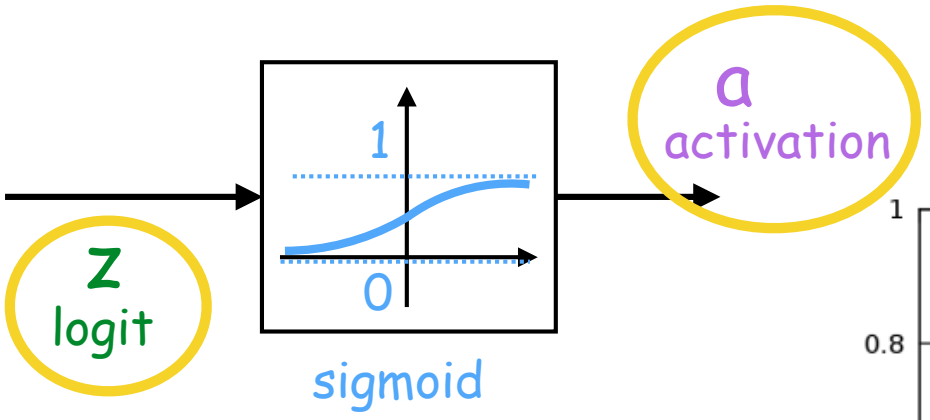
L is a function of x



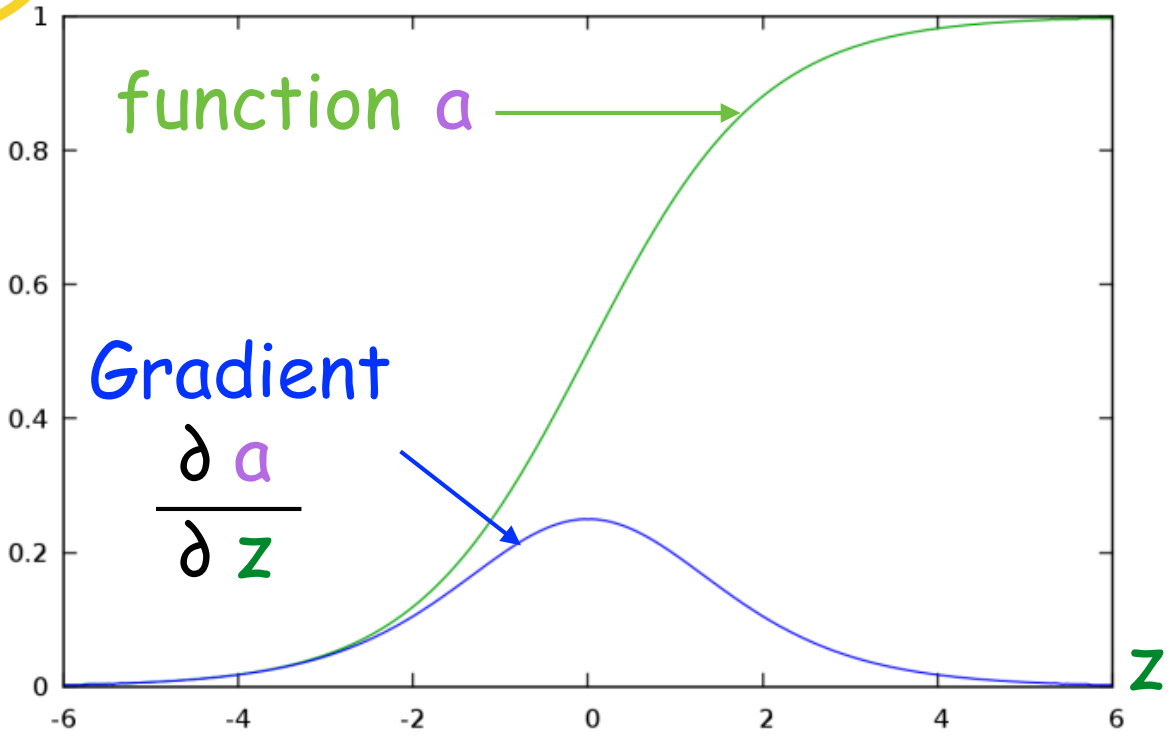
small gradient

large gradient

Gradient $\frac{\partial L}{\partial x}$ is also a function of x



$$a = \frac{1}{1 + e^{-z}}$$

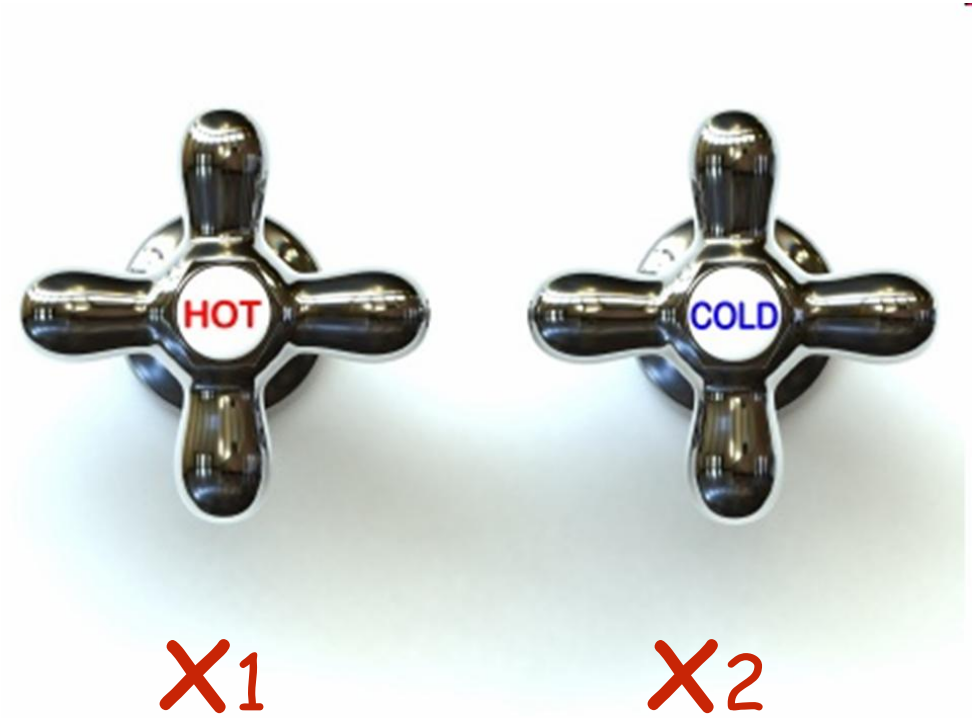


$$\frac{\partial a}{\partial z} = a\,(1 - a)$$

$$= \frac{1}{1 + e^{-z}} \left(1 - \frac{1}{1 + e^{-z}}\right)$$

is a function of z

Partial Gradients w.r.t. a vector $\frac{\partial L}{\partial x}$



x_1 x_2

L

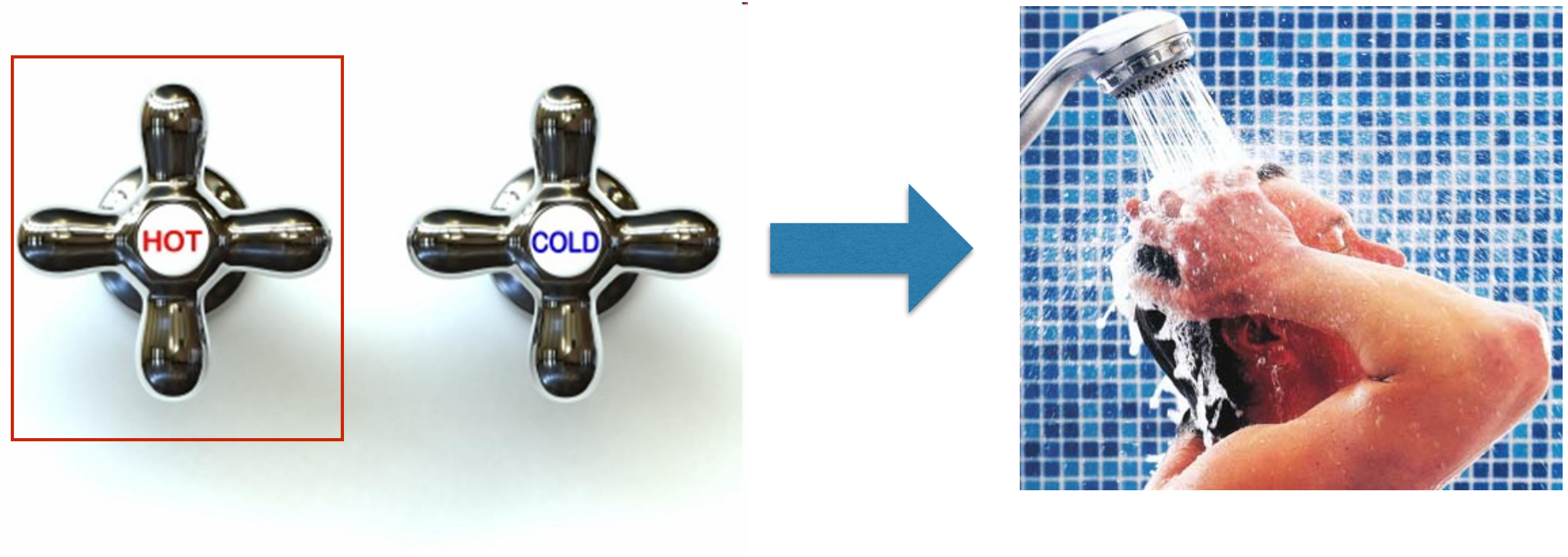
temperature

$$\frac{\partial L}{\partial x} = \left(\frac{\partial L}{\partial x_1} , \frac{\partial L}{\partial x_2} \right)$$

$$L = f(x_1, x_2)$$

Partial Gradients w.r.t. a vector $\frac{\partial L}{\partial x}$

x_1 x_2 L



partial gradient — the gradient of f over x_1 fixing x_2

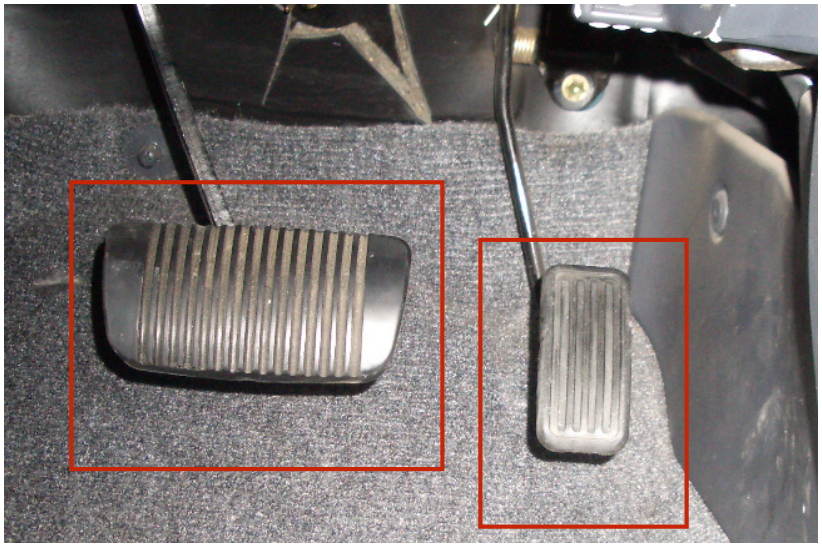
$$\left(\frac{\partial L}{\partial x_1} , \frac{\partial L}{\partial x_2} \right)$$

$$L = f(x_1, x_2)$$

Partial Gradients

X_1	$\frac{\partial L}{\partial x_1}$	X_2	$\frac{\partial L}{\partial x_2}$	Loss
0	-3	0	-10	100 bad (no water)
0	-6	5	-6	80 ok (some water, but cold)
3	-3	6	-3	60 better (some water, warm)
6	0	10	0	20 great (more water, warm)
8	3	6	-3	60 good (more water, hot)
9	6	3	-6	80 ok (more water, too hot)
10	10	0	-10	200 bad (more water, way too hot)

Partial Gradients w.r.t. a vector $\frac{\partial L}{\partial x}$



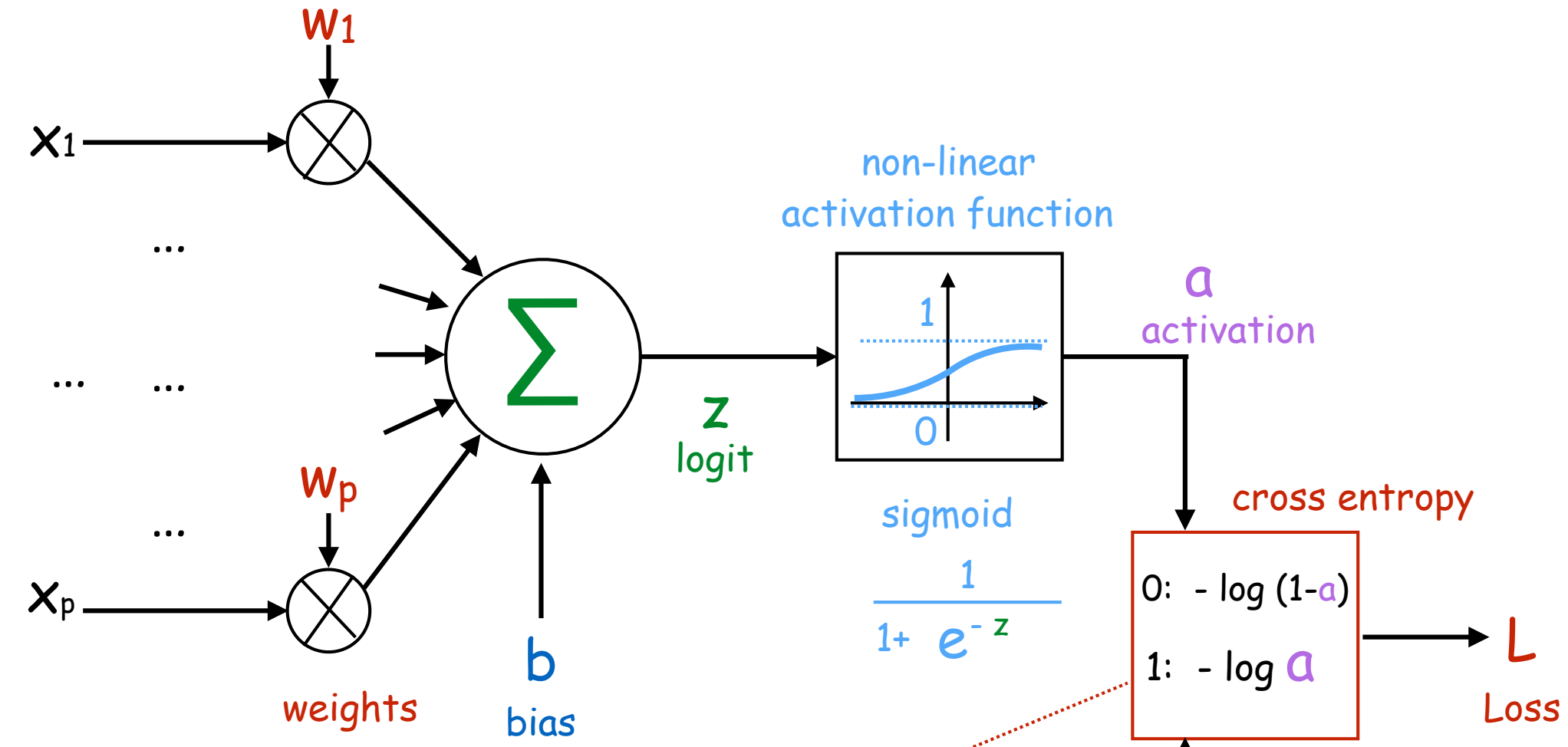
x_1
brake

x_2
gas

L
speed

$$\frac{\partial L}{\partial x} = \left(\frac{\partial L}{\partial x_1} , \frac{\partial L}{\partial x_2} \right)$$

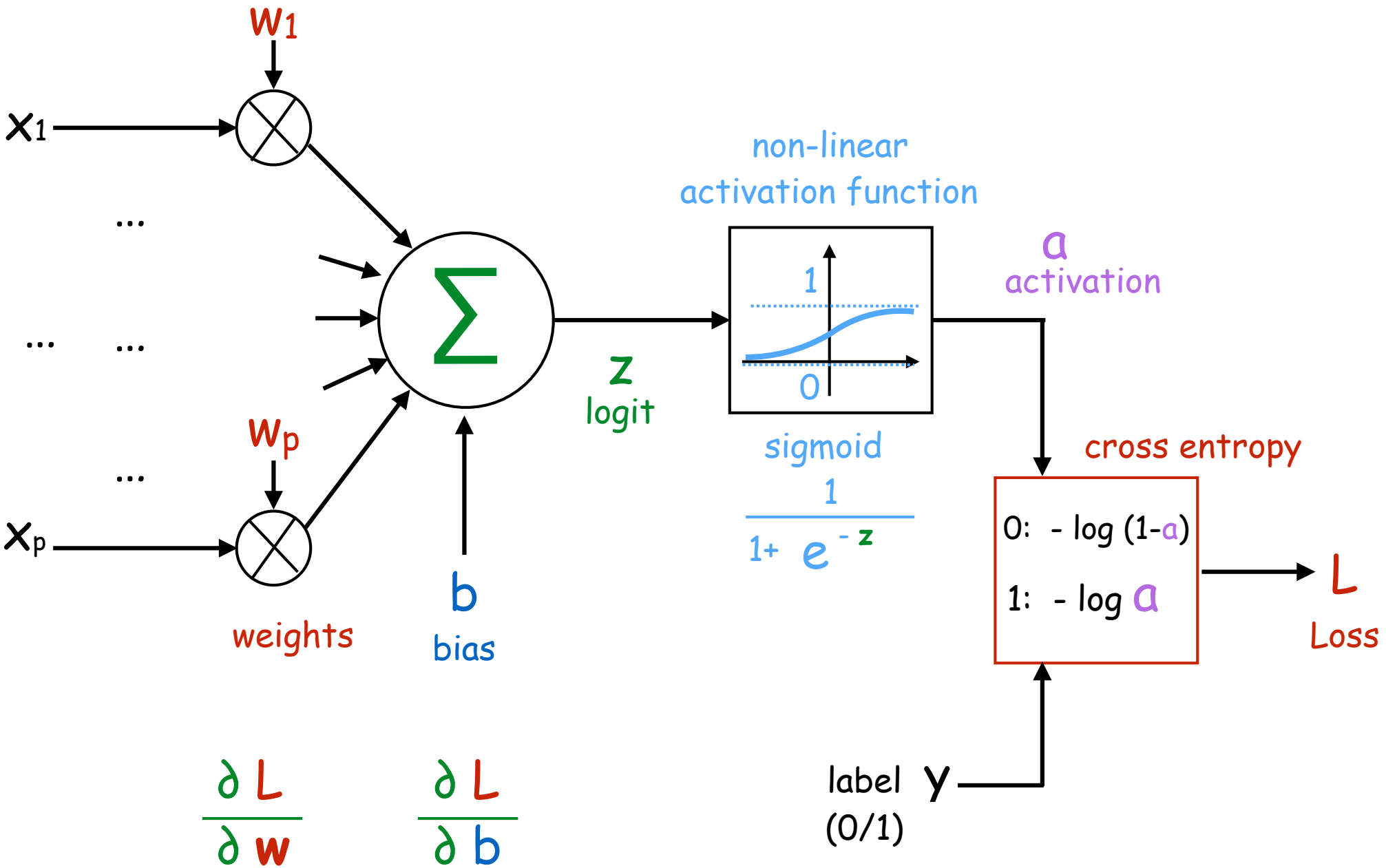
Loss on one data instance



if y = 0, L = - log (1- a)

if y = 1, L = - log a

Global Gradient of w, b



Chain Rule

$y = f(g(x))$ $\quad y = f(u) \quad u = g(x)$

$$\frac{\partial y}{\partial x} = \frac{\partial y}{\partial u} \times \frac{\partial u}{\partial x}$$

global gradient | local gradient | local gradient

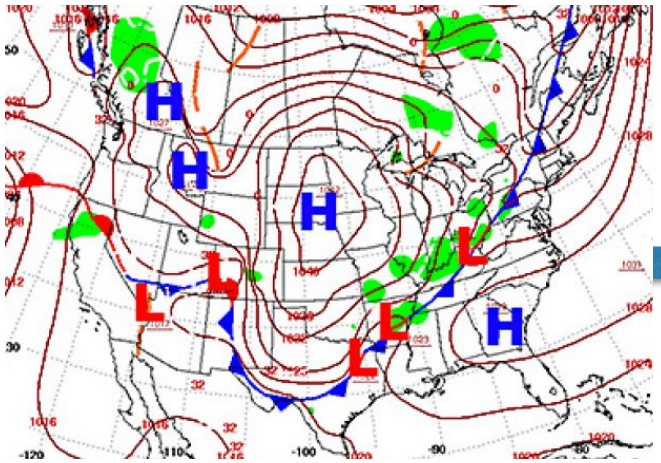
$y = f(g(h(x)))$ $\quad y = f(u) \quad u = g(v) \quad v = h(x)$

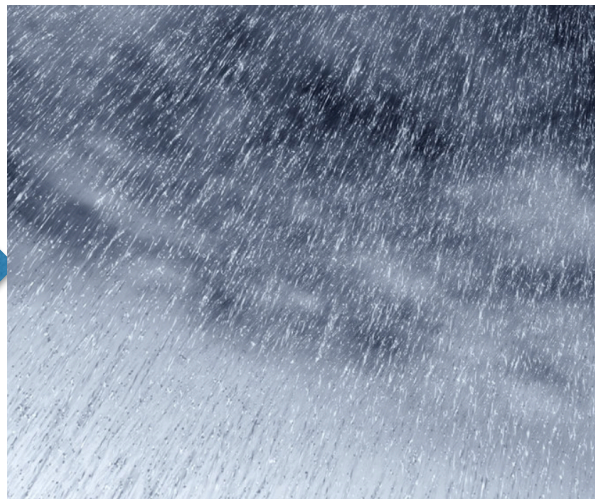
$$\frac{\partial y}{\partial x} = \frac{\partial y}{\partial u} \times \frac{\partial u}{\partial v} \times \frac{\partial v}{\partial x}$$

global gradient | local gradient | local gradient | local gradient

Chain Rule

$$\frac{\partial L}{\partial x} = \frac{\partial L}{\partial y} \times \frac{\partial y}{\partial x}$$





X

low air pressure

Y

rain

L

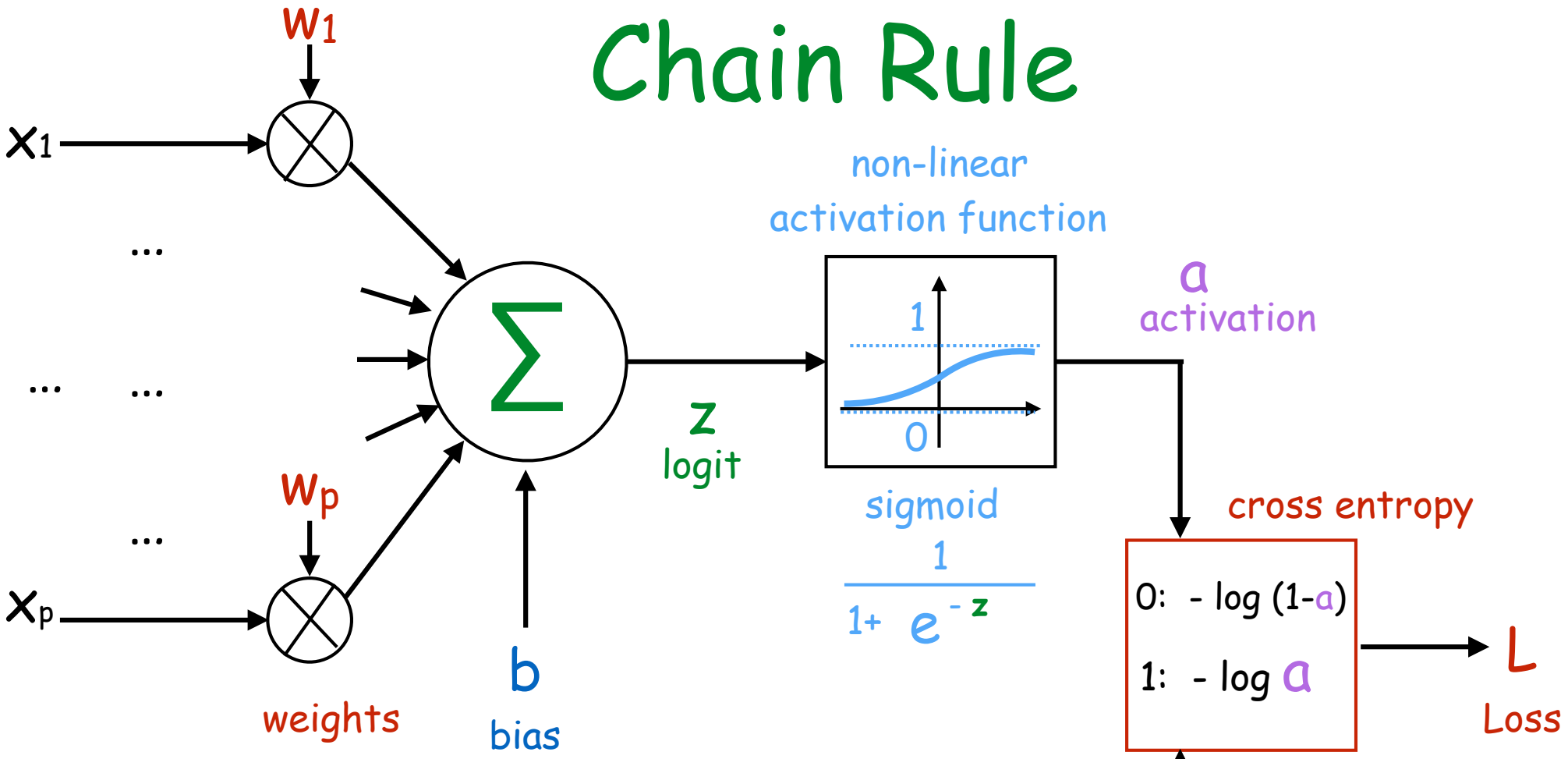
slow traffic

$\frac{\partial L}{\partial y}$: how much the traffic speed L will be changed if we change the rain Y

$\frac{\partial y}{\partial x}$: how much the rain Y will be changed if we change the air pressure X

$\frac{\partial L}{\partial x}$: how much the traffic speed L will be changed if we change the air pressure X

Chain Rule

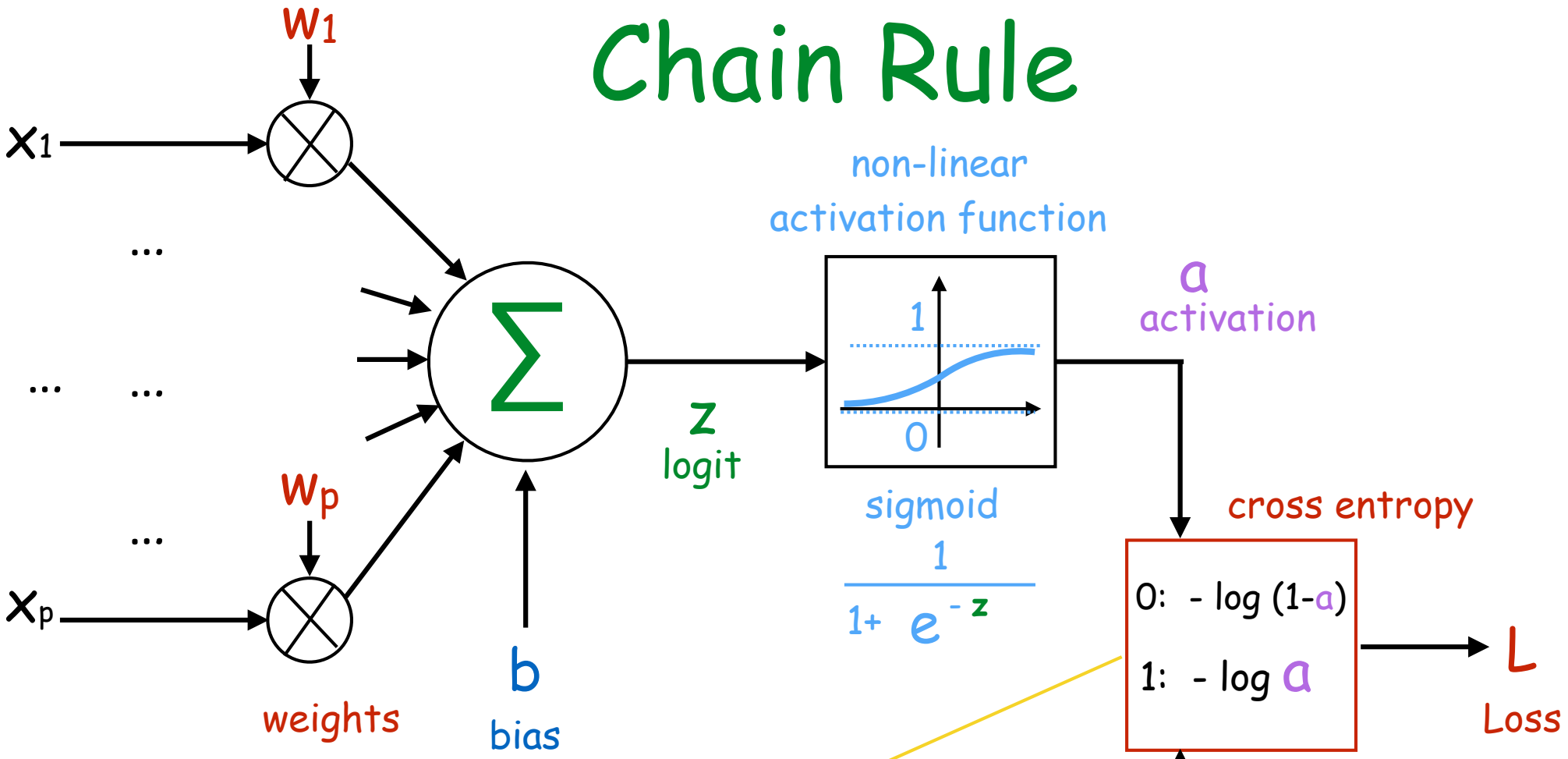


$$\frac{\partial L}{\partial w_i} = \frac{\partial L}{\partial a} \times \frac{\partial a}{\partial z} \times \frac{\partial z}{\partial w_i}$$

$$\frac{\partial L}{\partial b} = \frac{\partial L}{\partial a} \times \frac{\partial a}{\partial z} \times \frac{\partial z}{\partial b}$$

products of local gradients

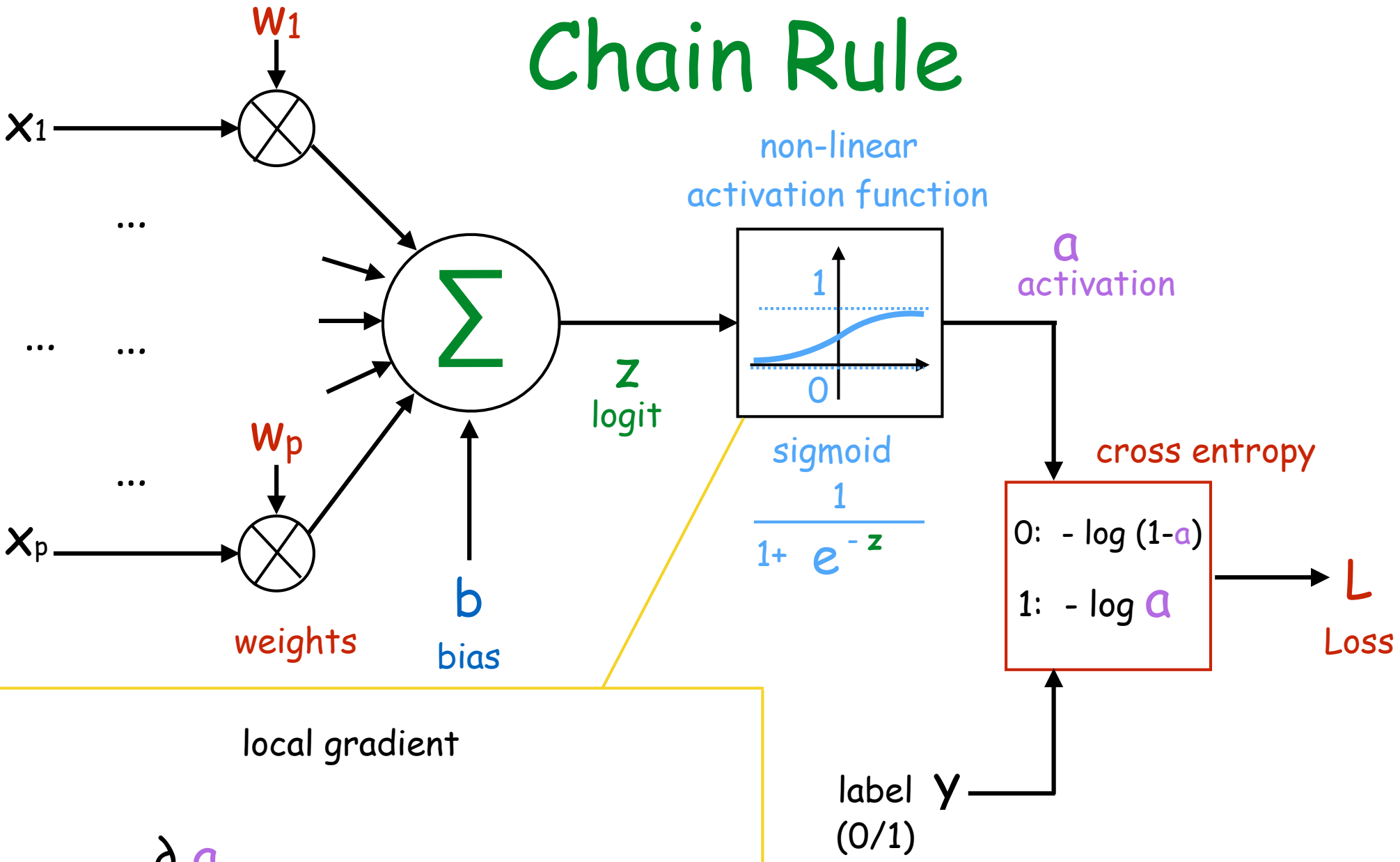
Chain Rule



local gradient

$$\frac{\partial L}{\partial a} = \begin{cases} 1/(1-a) & \text{if } y = 0 \\ -1/a & \text{if } y = 1 \end{cases}$$

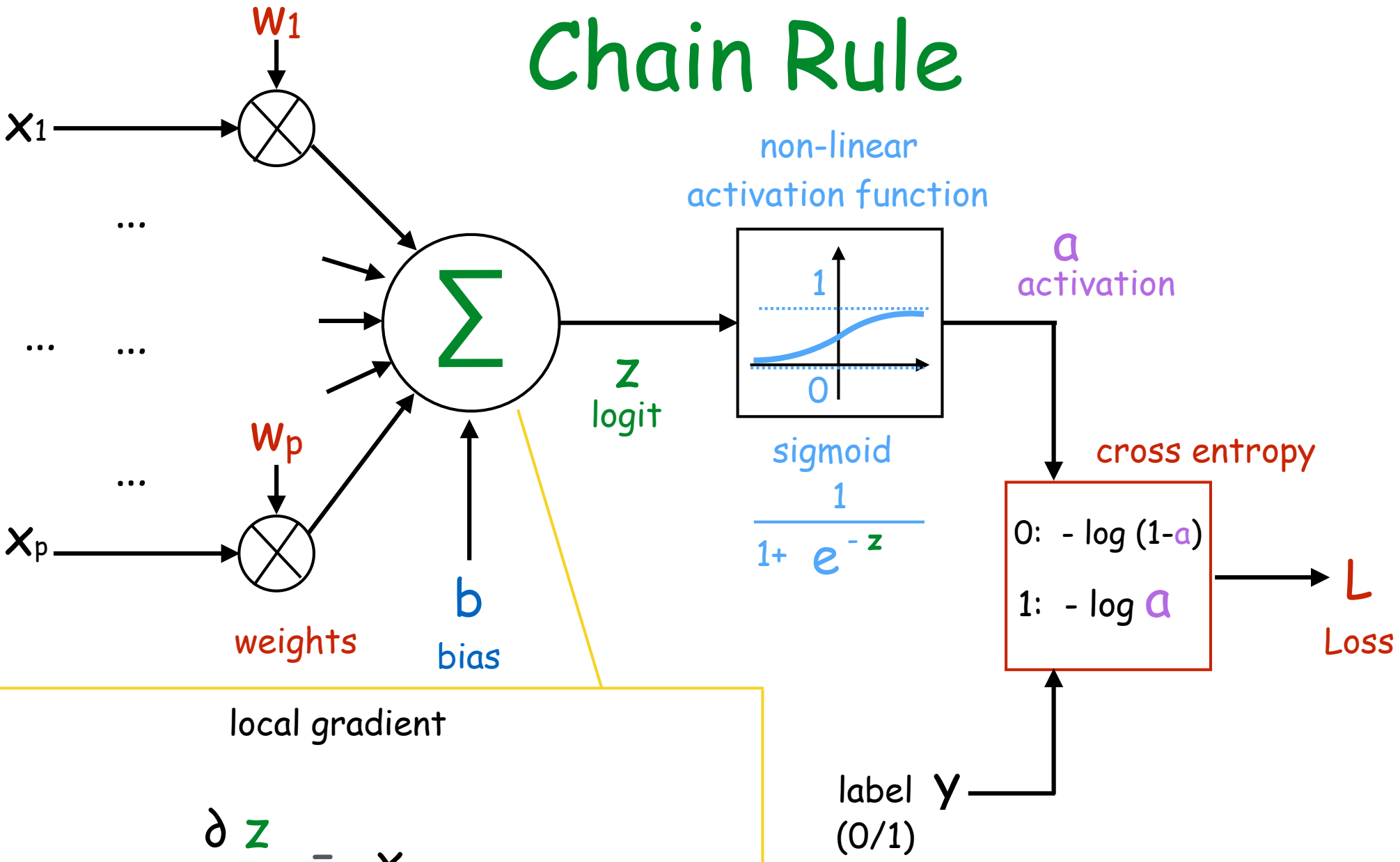
Chain Rule



local gradient

$$\frac{\partial a}{\partial z} = a\,(1 - a)$$

Chain Rule

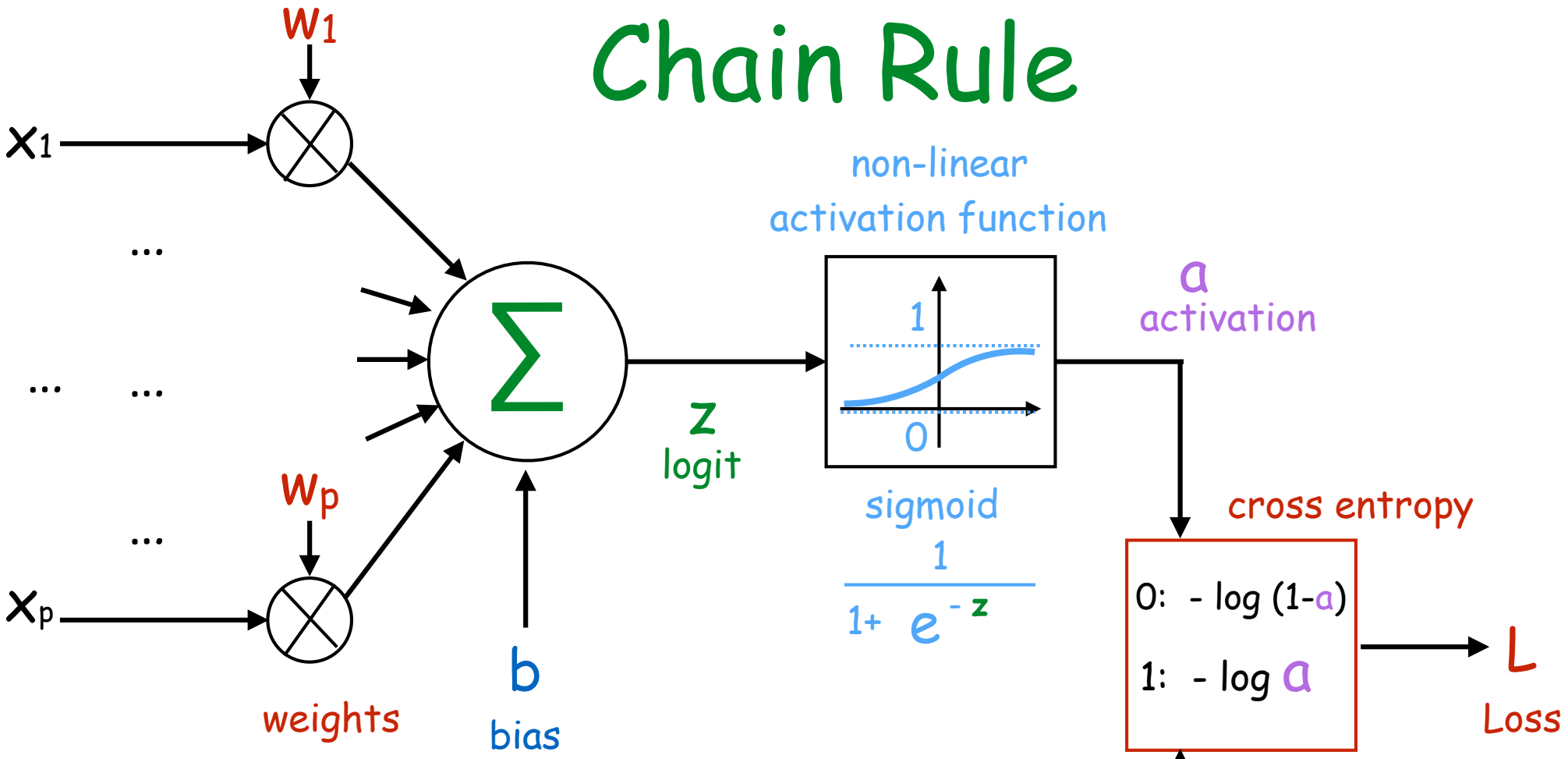


local gradient

$$\frac{\partial z}{\partial w_i} = x_i$$

$$\frac{\partial z}{\partial b} = 1$$

Chain Rule

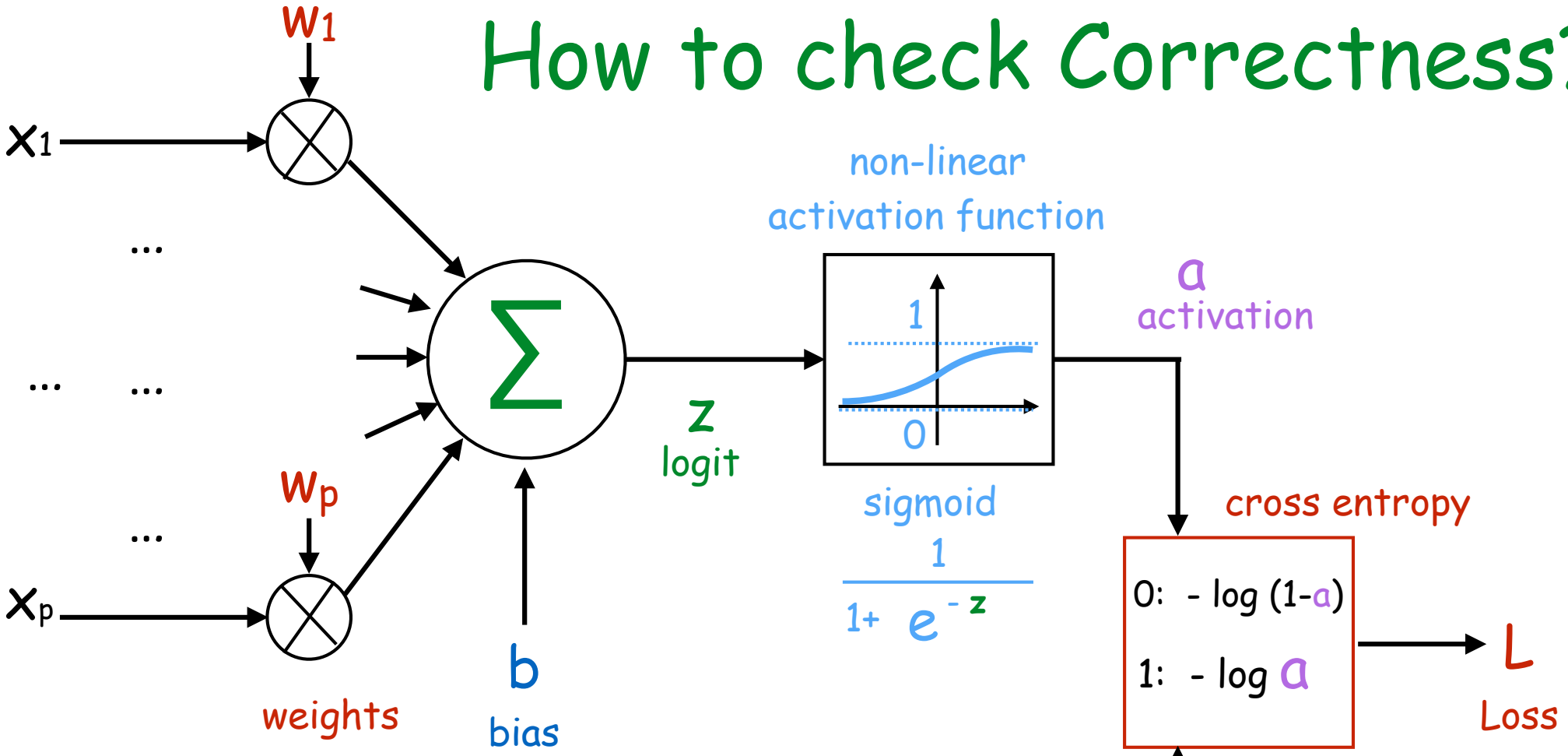


$$\frac{\partial L}{\partial w_i} = \frac{\partial L}{\partial a} \times \frac{\partial a}{\partial z} \times \frac{\partial z}{\partial w_i}$$

$$\frac{\partial L}{\partial b} = \frac{\partial L}{\partial a} \times \frac{\partial a}{\partial z} \times \frac{\partial z}{\partial b}$$

products of local gradients

How to check Correctness?



$\frac{\partial L}{\partial w_i}$ very complex

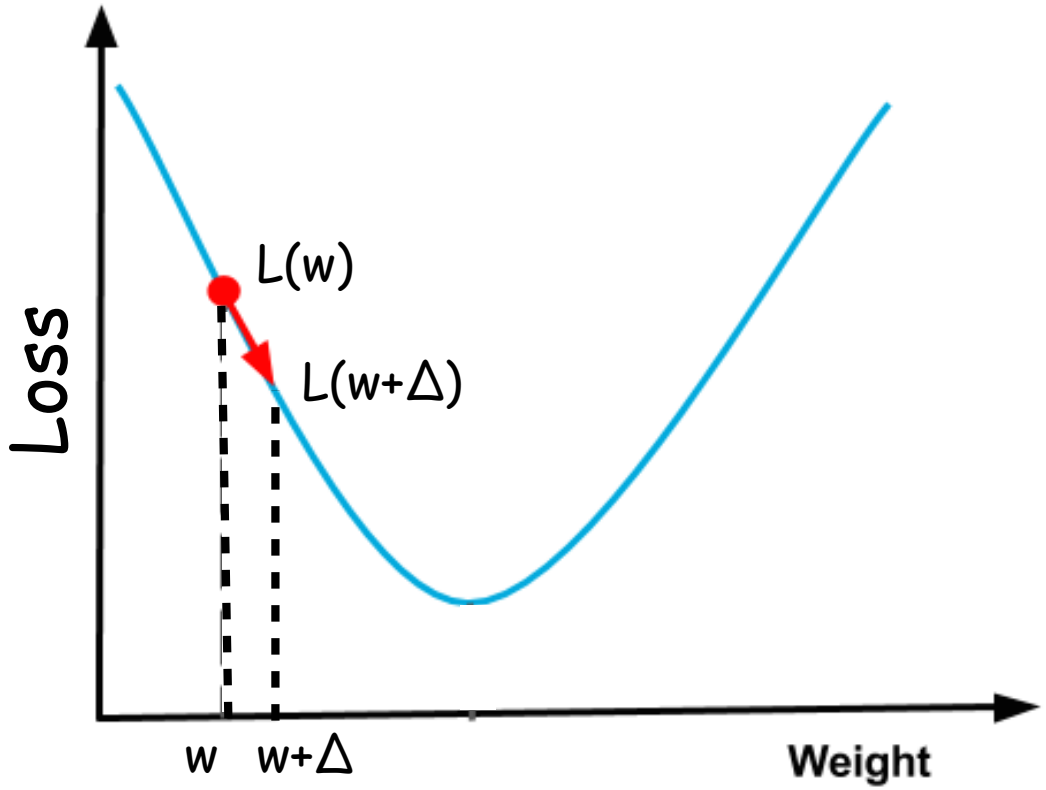
$\frac{\partial L}{\partial b}$ how to check for correctness?

Gradient Checking

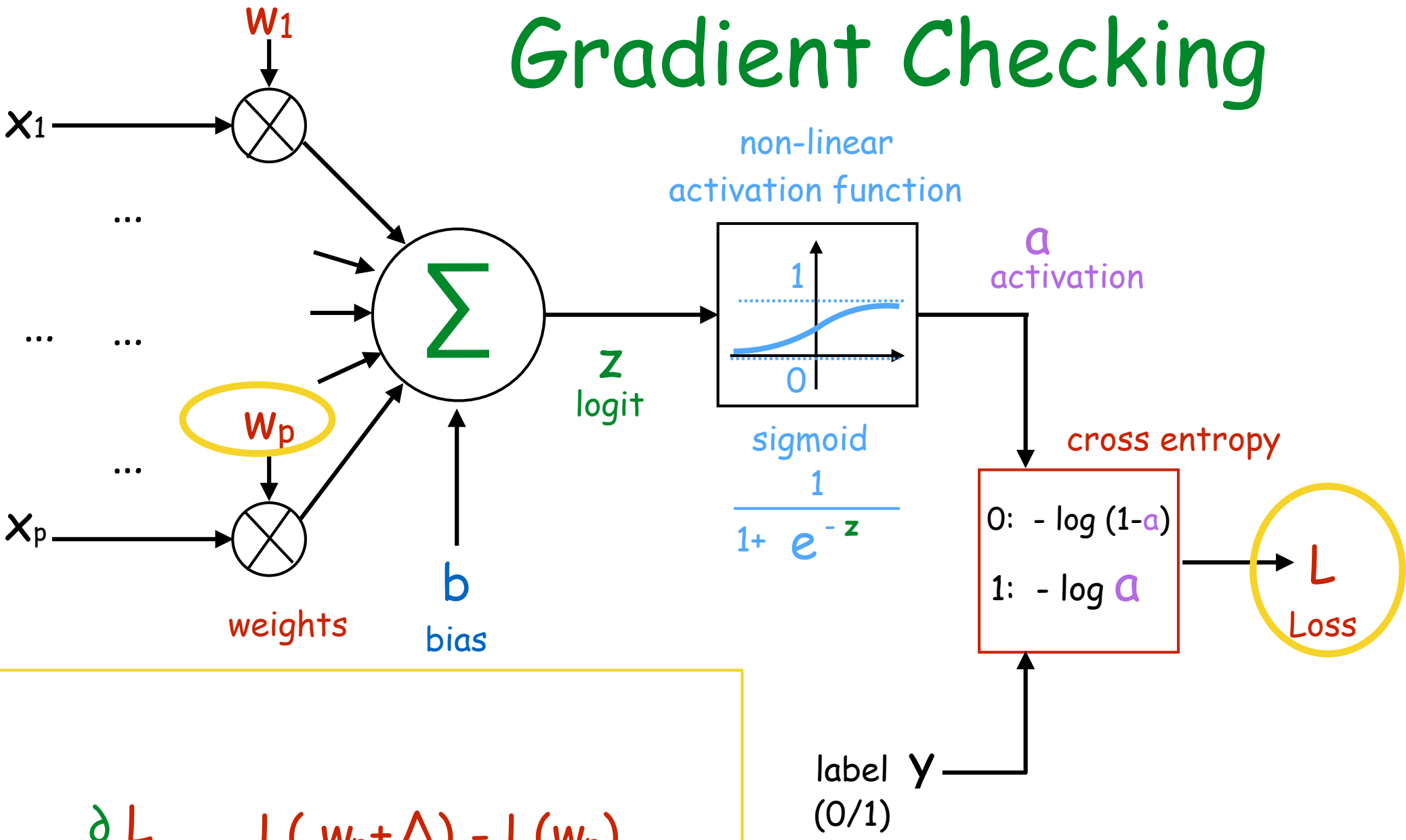
(when Δ is small)

$$\frac{\partial L}{\partial w} \approx \frac{L(w+\Delta) - L(w)}{\Delta}$$

Numerical gradient

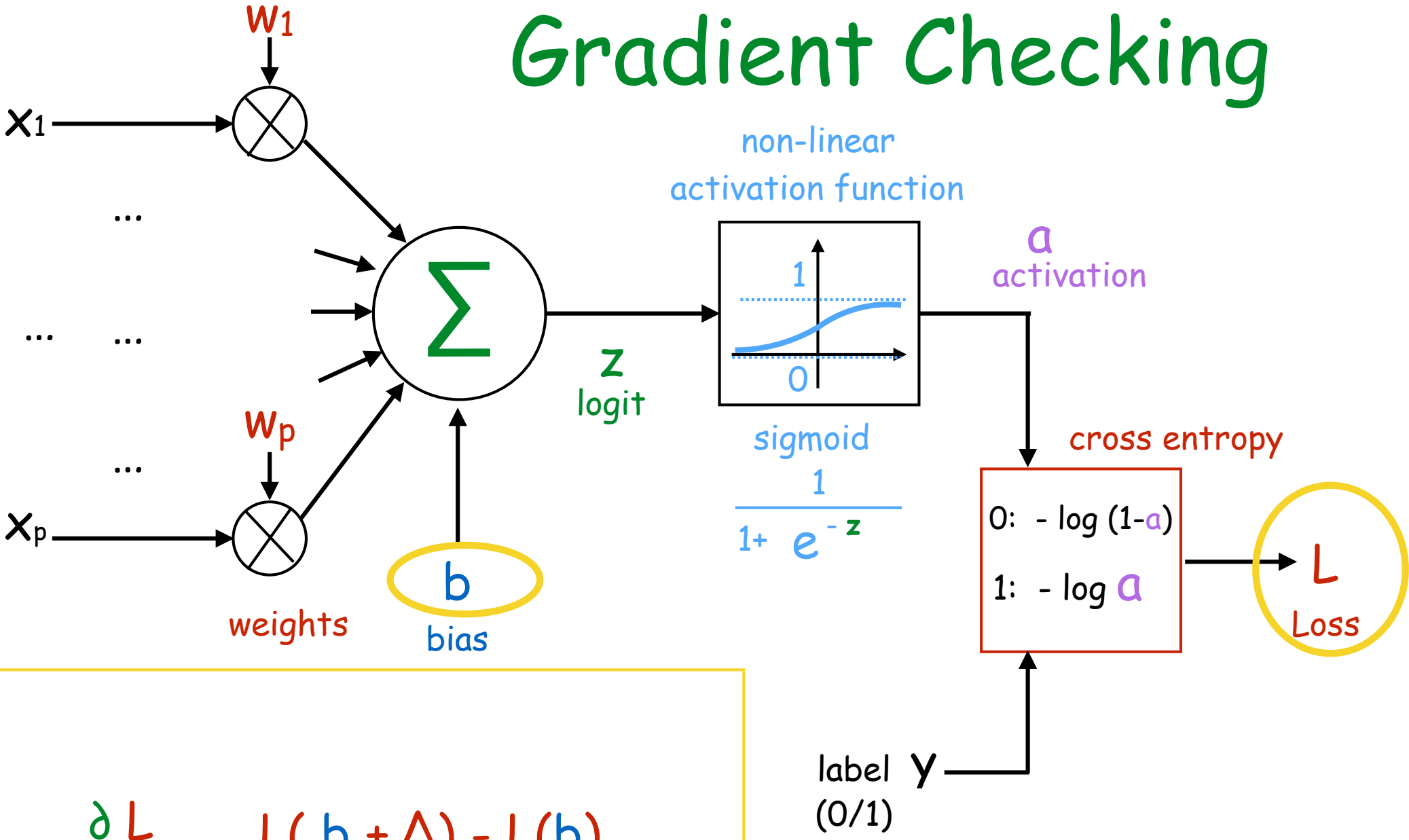


Gradient Checking



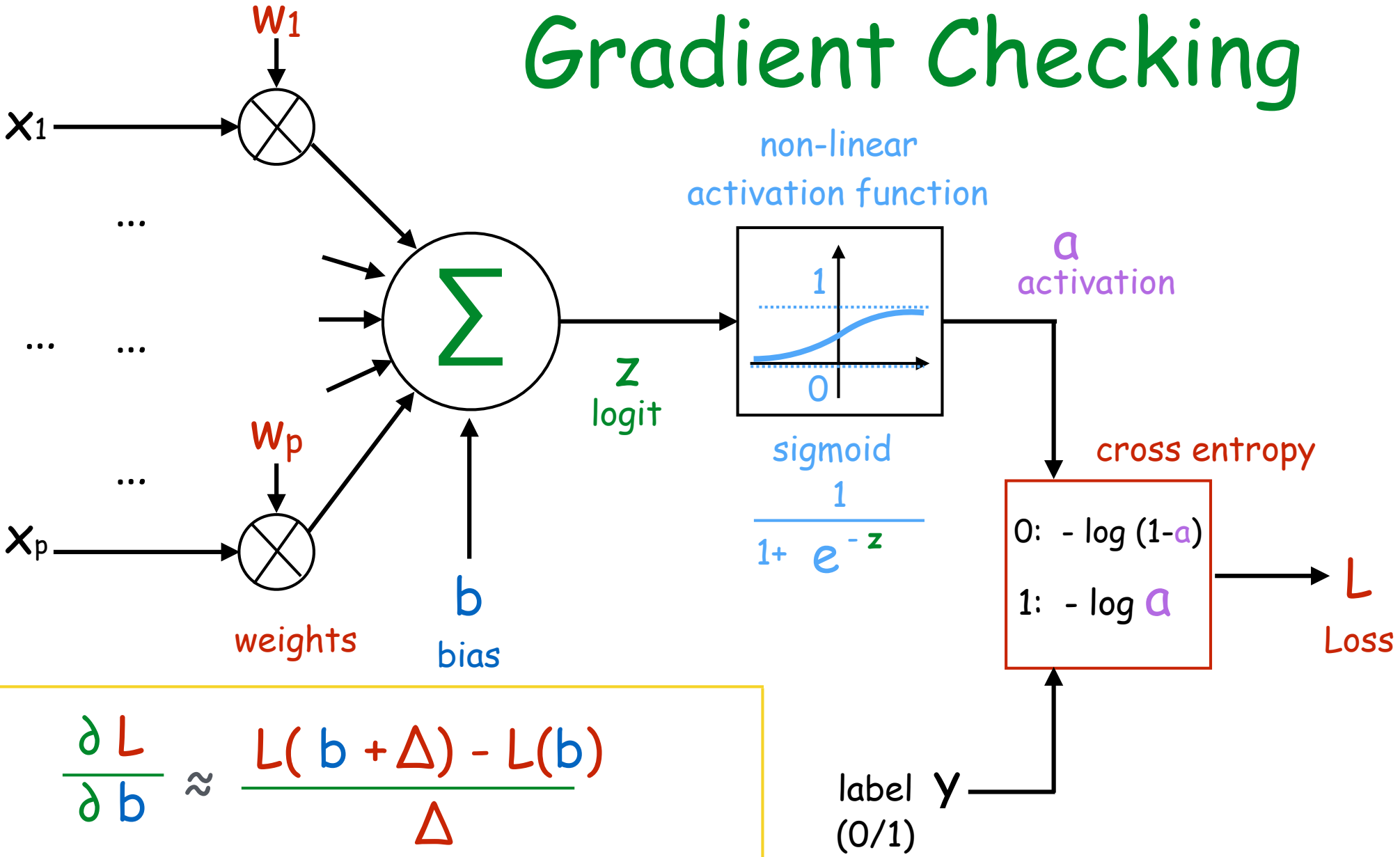
$$\frac{\partial L}{\partial w_p} \approx \frac{L(w_p+\Delta) - L(w_p)}{\Delta}$$

Gradient Checking



$$\frac{\partial L}{\partial b} \approx \frac{L(b+\Delta) - L(b)}{\Delta}$$

Gradient Checking



$$\frac{\partial L}{\partial b} \approx \frac{L(b+\Delta) - L(b)}{\Delta}$$

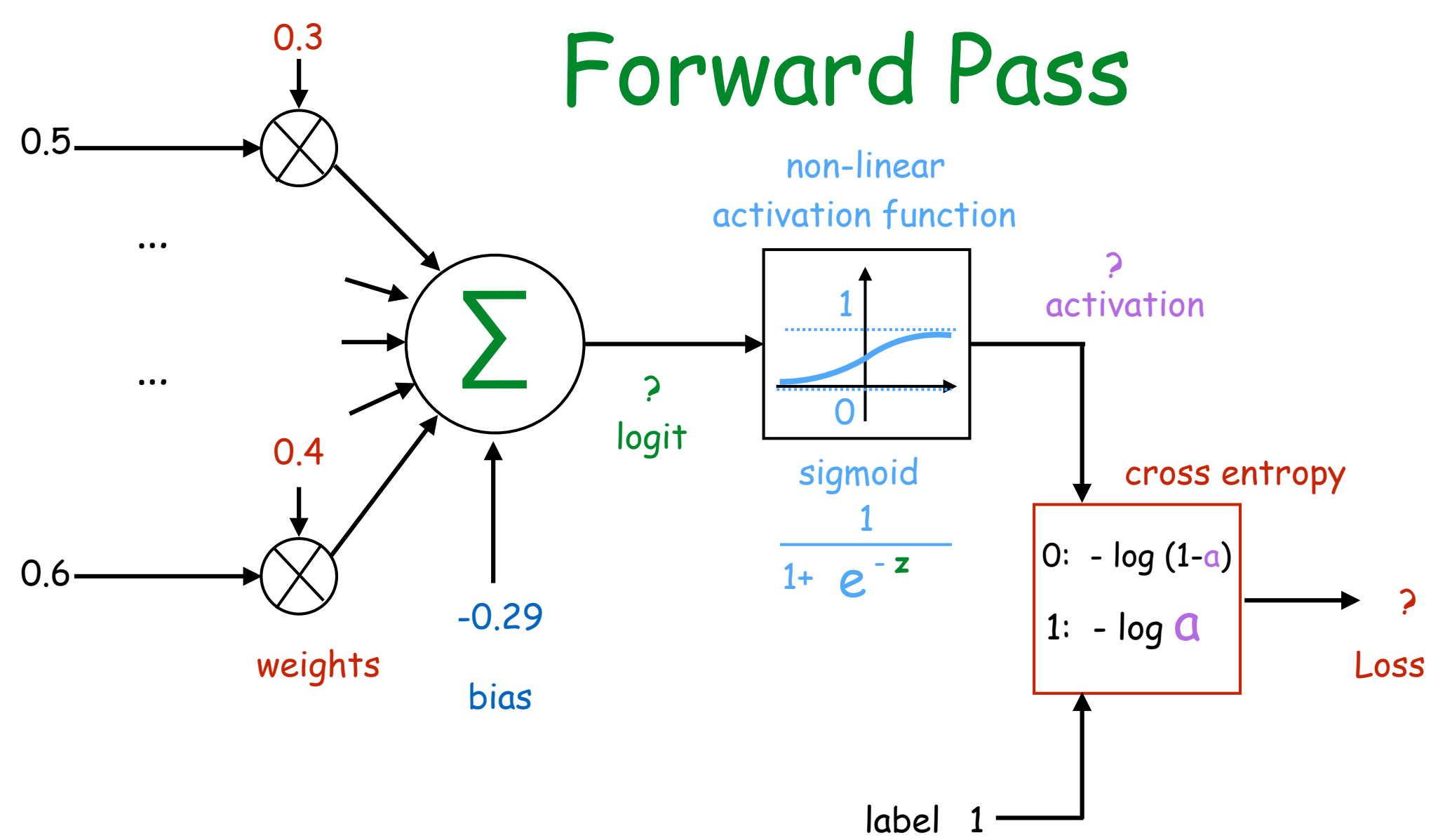
slow (require two forward passes)

not accurate (require Δ to be very small)

only used for checking the correctness of analytical gradients

Example

Forward Pass

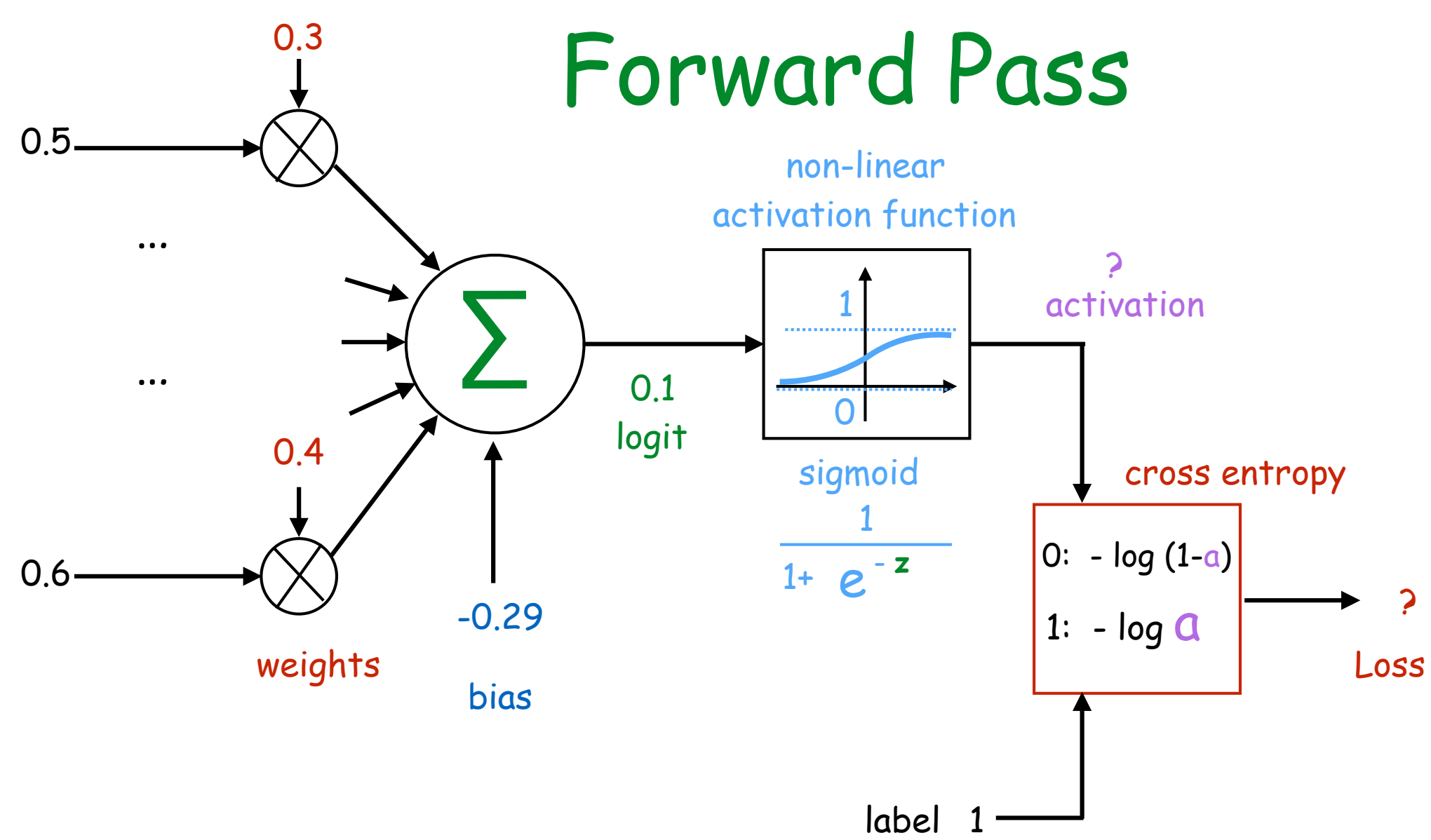


Given a training instance **x**, y

x = (0.5, …, 0.6)

y = 1

Forward Pass

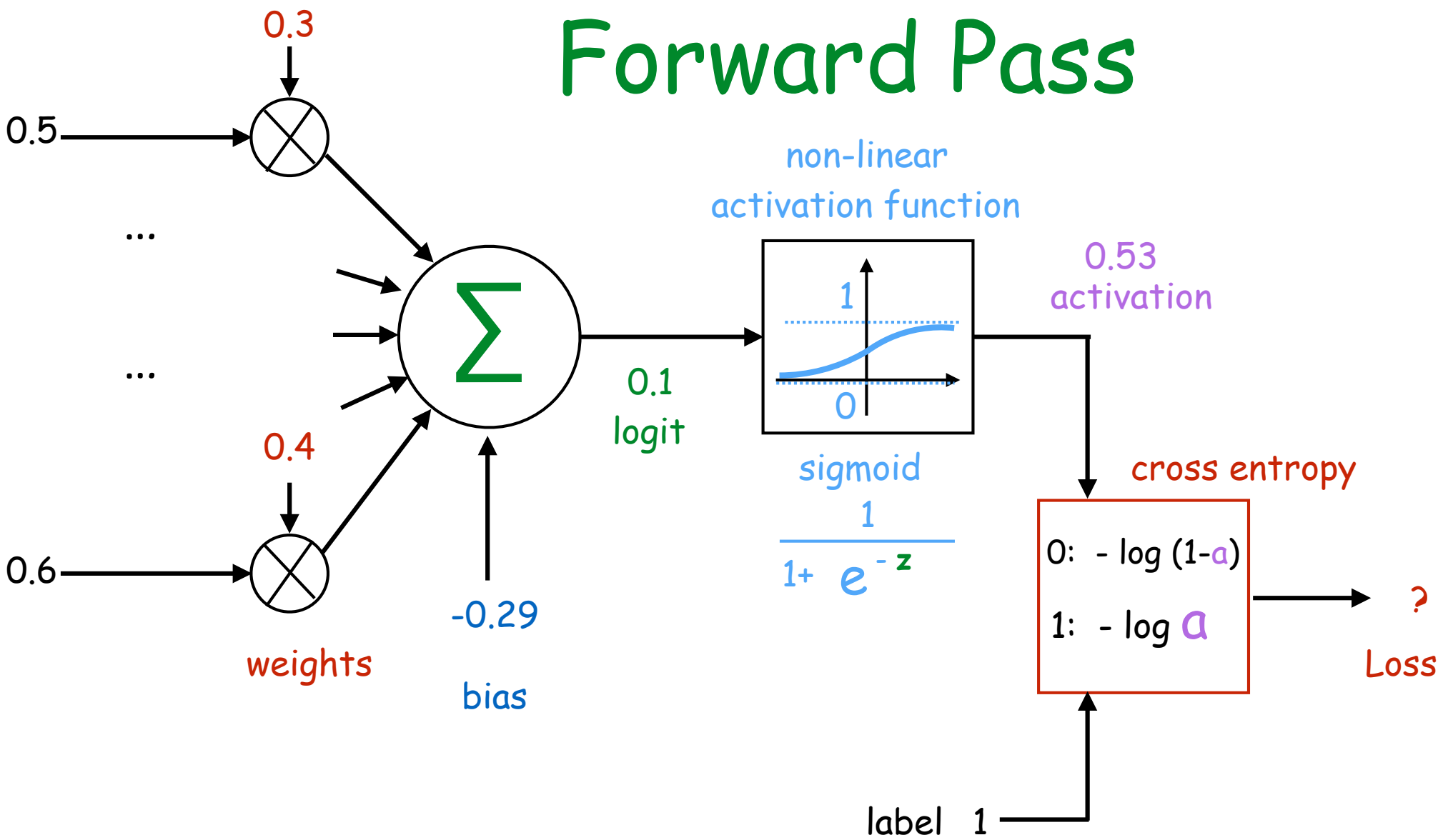


Given a training instance **x**, y

x = (0.5, …, 0.6)

y = 1

Forward Pass

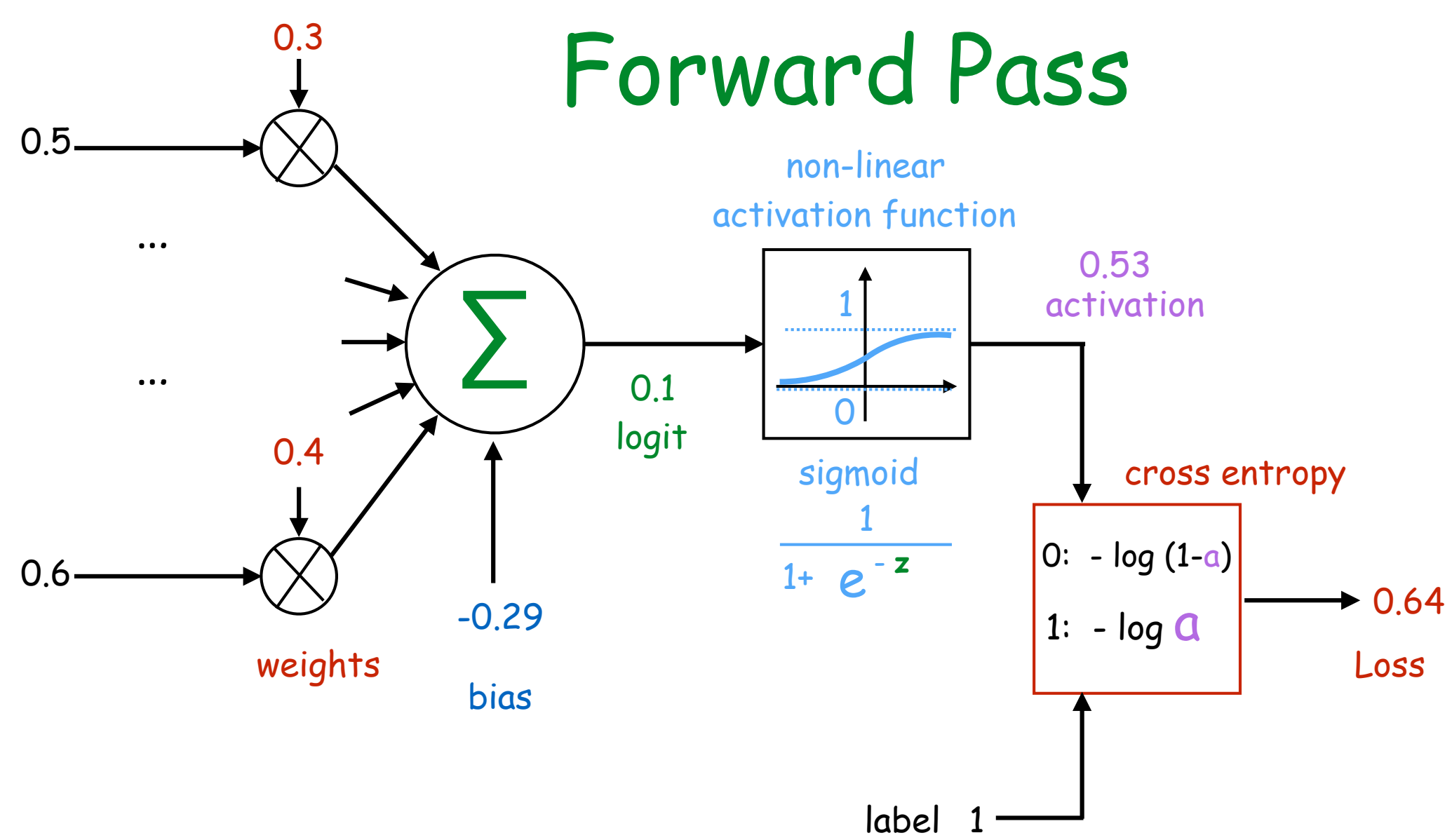


Given a training instance **x**, y

x = (0.5, …, 0.6)

y = 1

Forward Pass

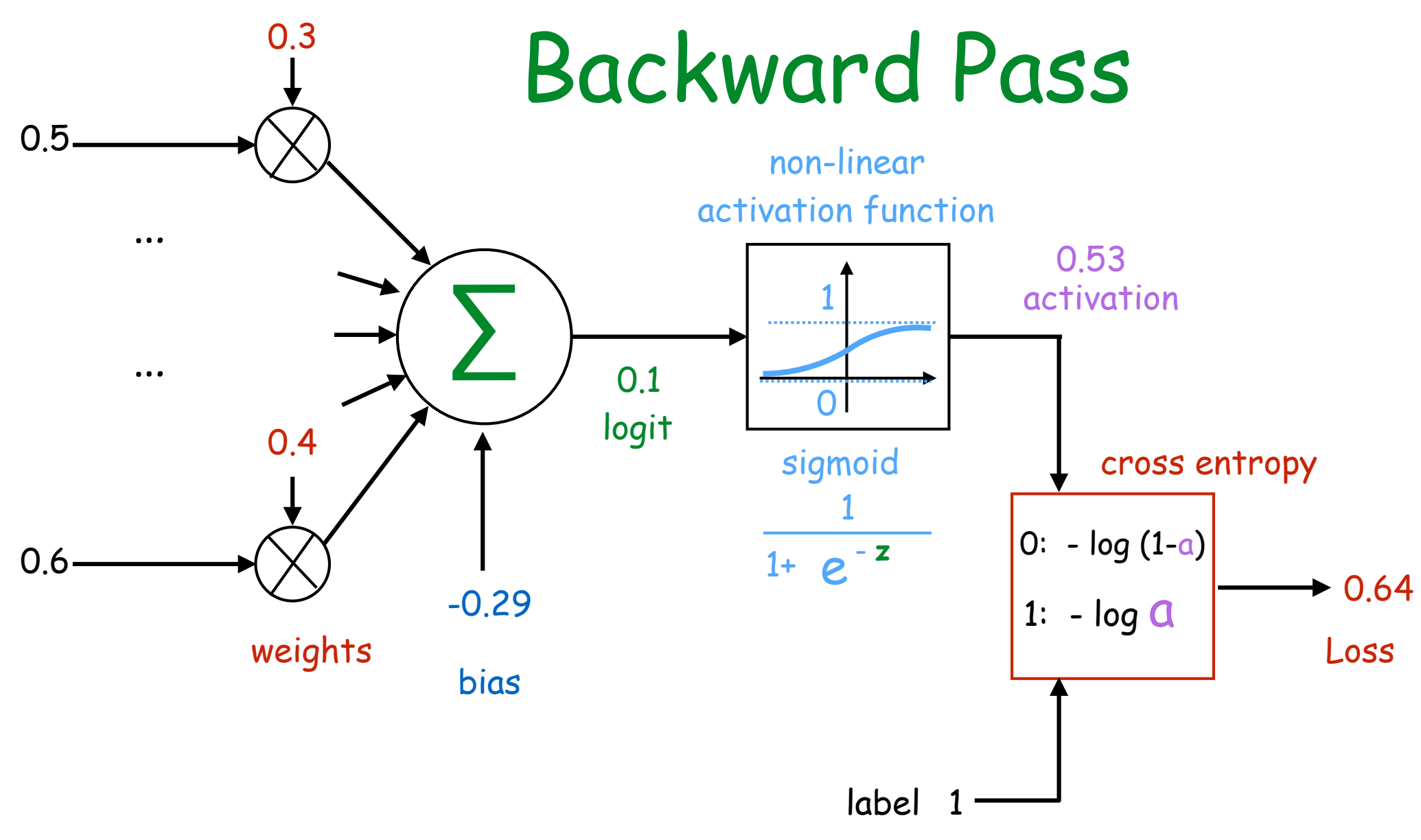


Given a training instance **x**, y

x = (0.5, …, 0.6)

y = 1

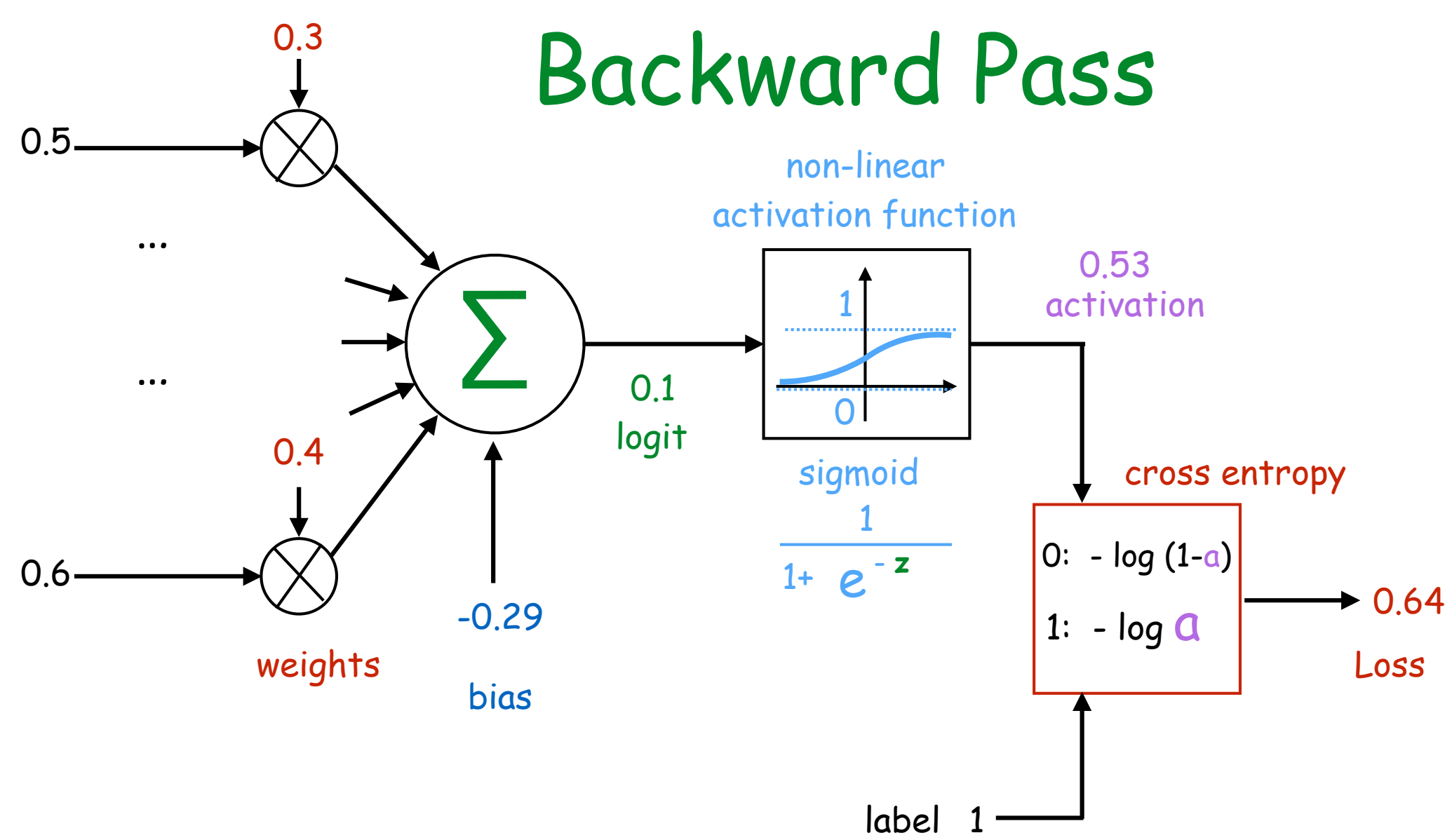
Backward Pass



$$\frac{\partial L}{\partial a} = -1/a$$

$$= -1.9$$

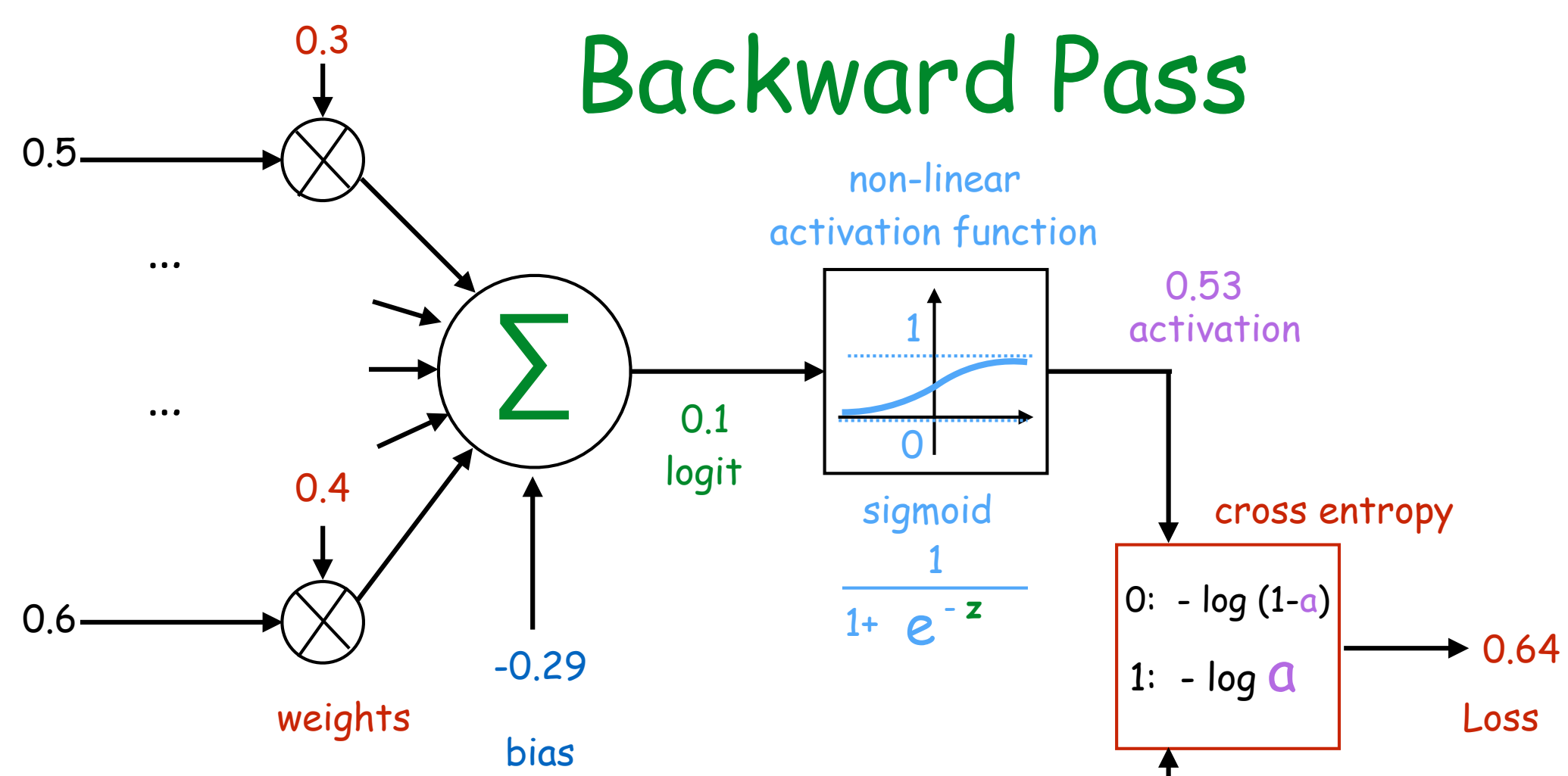
Backward Pass



$$\frac{\partial a}{\partial z} = a(1-a) = 0.24$$

$$\frac{\partial L}{\partial a} = -1.9$$

Backward Pass



$$\frac{\partial z}{\partial w_1} = x_1 = 0.5$$

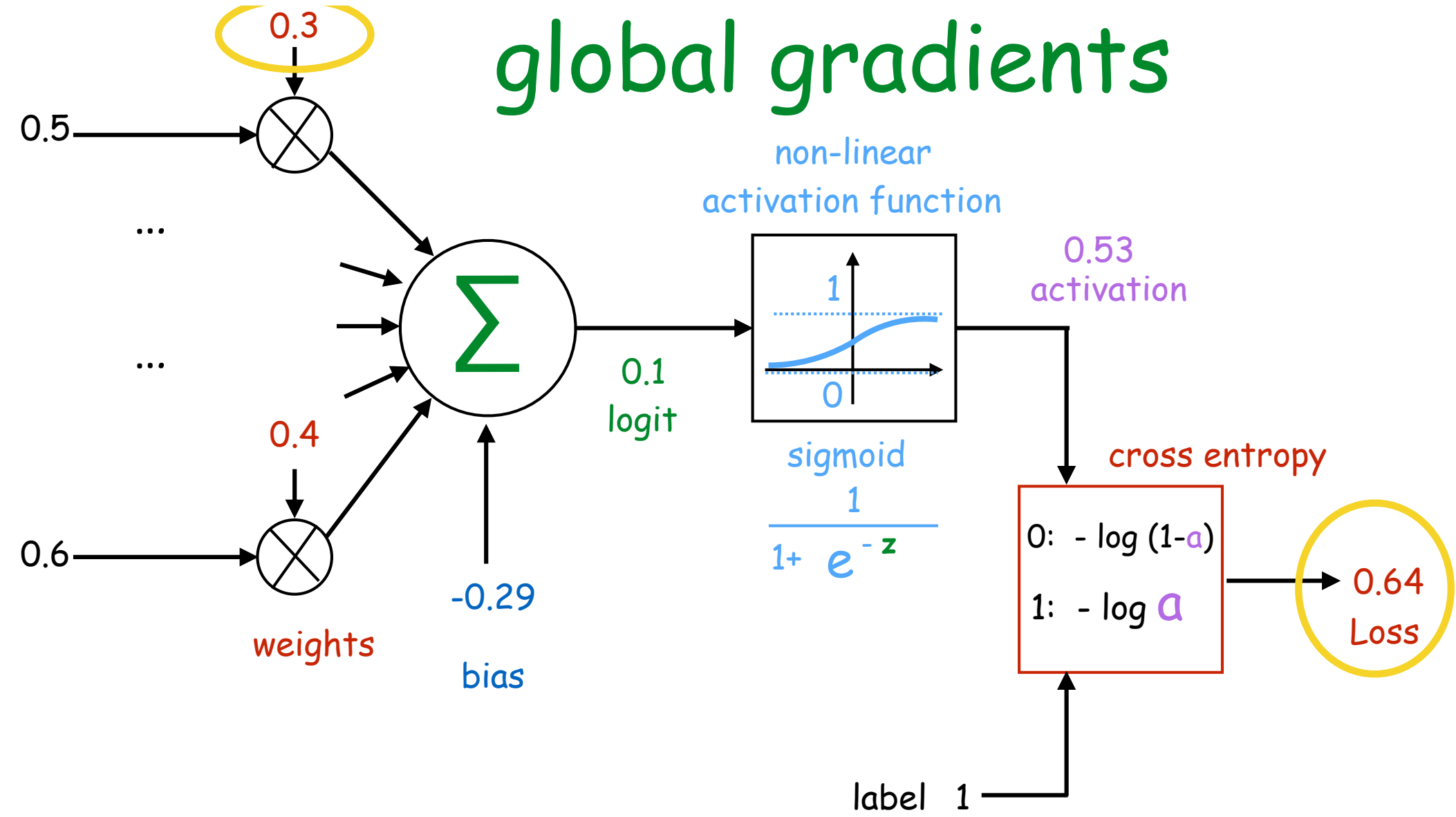
$$\frac{\partial z}{\partial w_p} = x_p = 0.6$$

$$\frac{\partial z}{\partial b} = 1$$

$$\frac{\partial a}{\partial z} = 0.24$$

$$\frac{\partial L}{\partial a} = -1.9$$

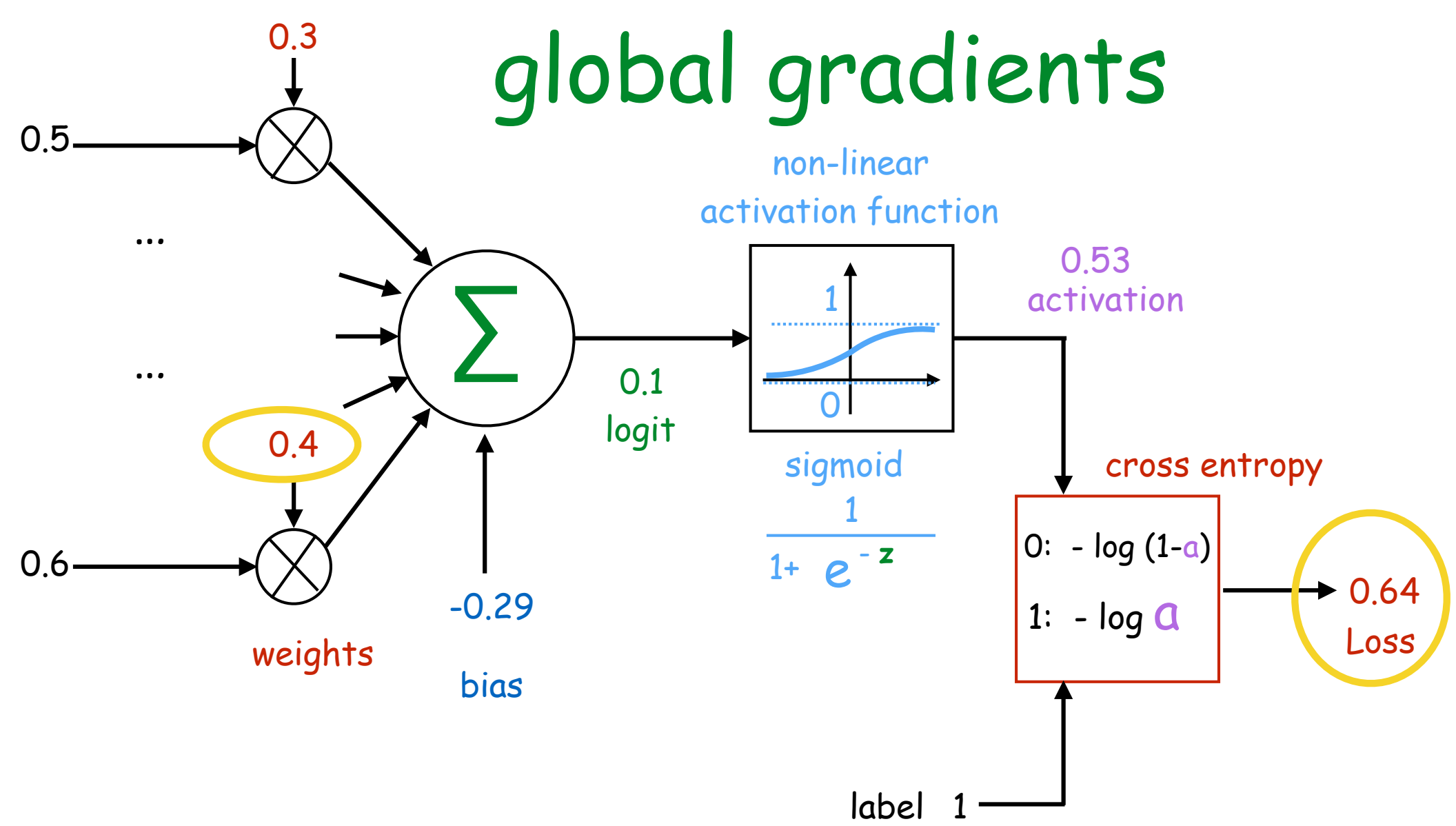
global gradients



$$\frac{\partial L}{\partial w_1} = \frac{\partial z}{\partial w_1} \times \frac{\partial a}{\partial z} \times \frac{\partial L}{\partial a}$$

-0.22 = 0.5 × 0.24 × -1.9

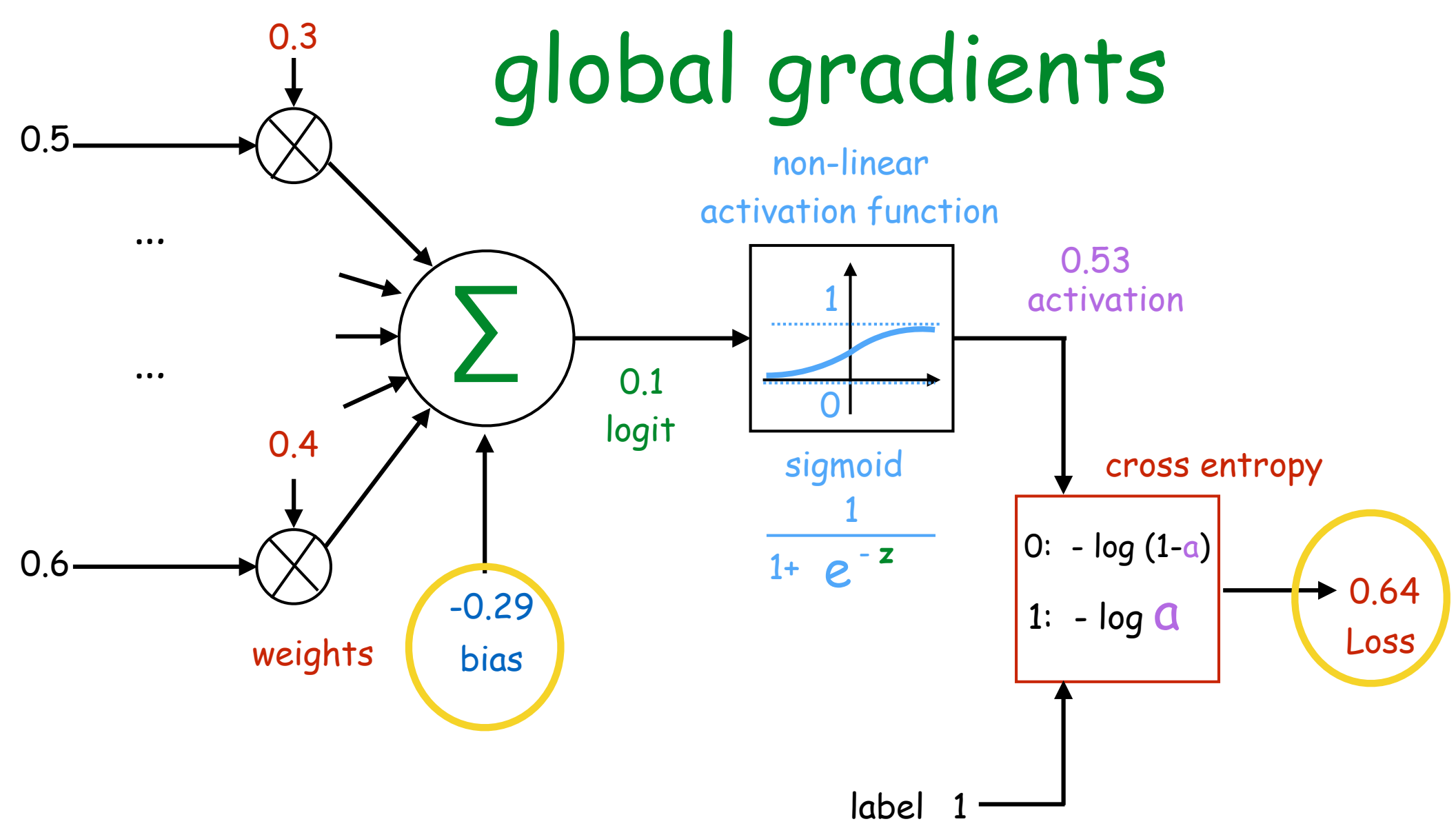
global gradients



$$\frac{\partial L}{\partial w_p} = \frac{\partial z}{\partial w_p} \times \frac{\partial a}{\partial z} \times \frac{\partial L}{\partial a}$$

$$-0.27 \qquad 0.6 \qquad 0.24 \qquad -1.9$$

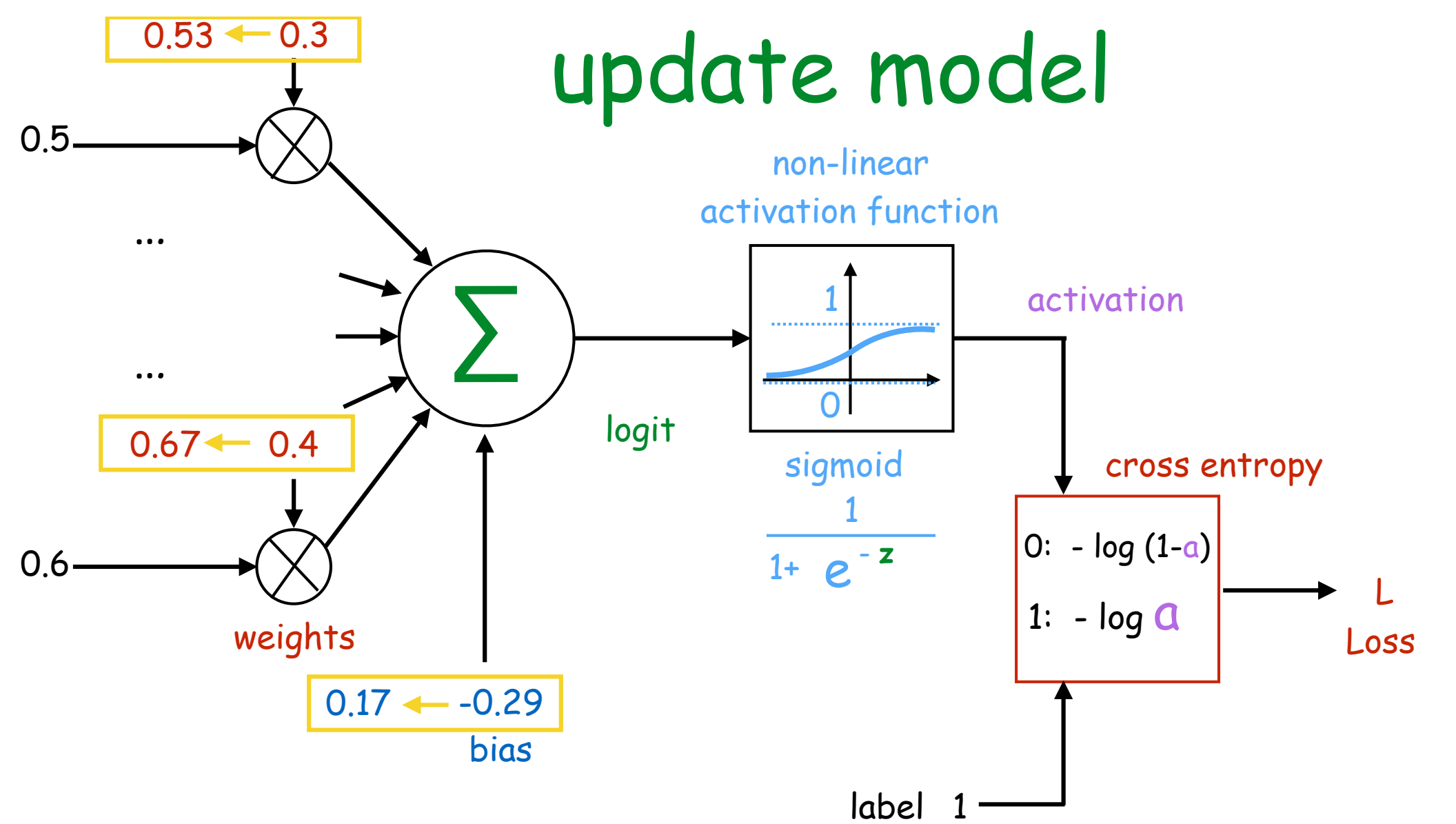
global gradients



$$\frac{\partial L}{\partial b} = \frac{\partial z}{\partial b} \times \frac{\partial a}{\partial z} \times \frac{\partial L}{\partial a}$$

$$-0.46 \qquad 1 \qquad 0.24 \qquad -1.9$$

update model



$$w \leftarrow w - a \frac{\partial L}{\partial w}$$

$$b \leftarrow b - a \frac{\partial L}{\partial b}$$

for example

$a = 1$

Logistic Regression (train)

initialize **w** and b

Loop for n_epoch iterations:

Loop for each training instance (**x**, y) in training set

forward pass to compute z, a and L for the instance

backward pass to compute local gradients

$$\frac{\partial L}{\partial a} \quad \frac{\partial a}{\partial z} \quad \frac{\partial z}{\partial b} \quad \frac{\partial z}{\partial \mathbf{w}}$$

compute global gradients using chain rule

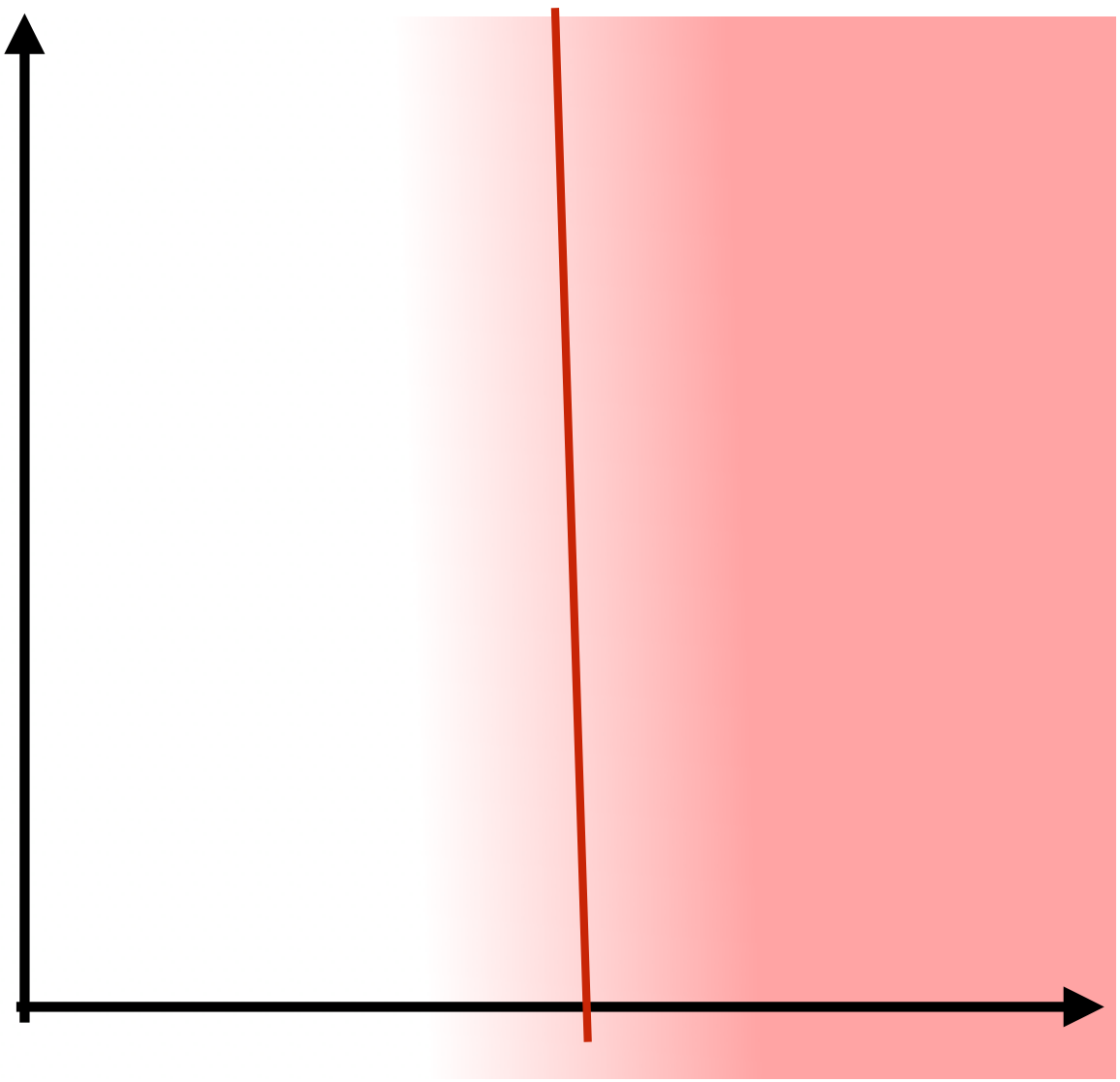
$$\frac{\partial L}{\partial \mathbf{w}} \quad \frac{\partial L}{\partial b}$$

update the parameters **w** and b

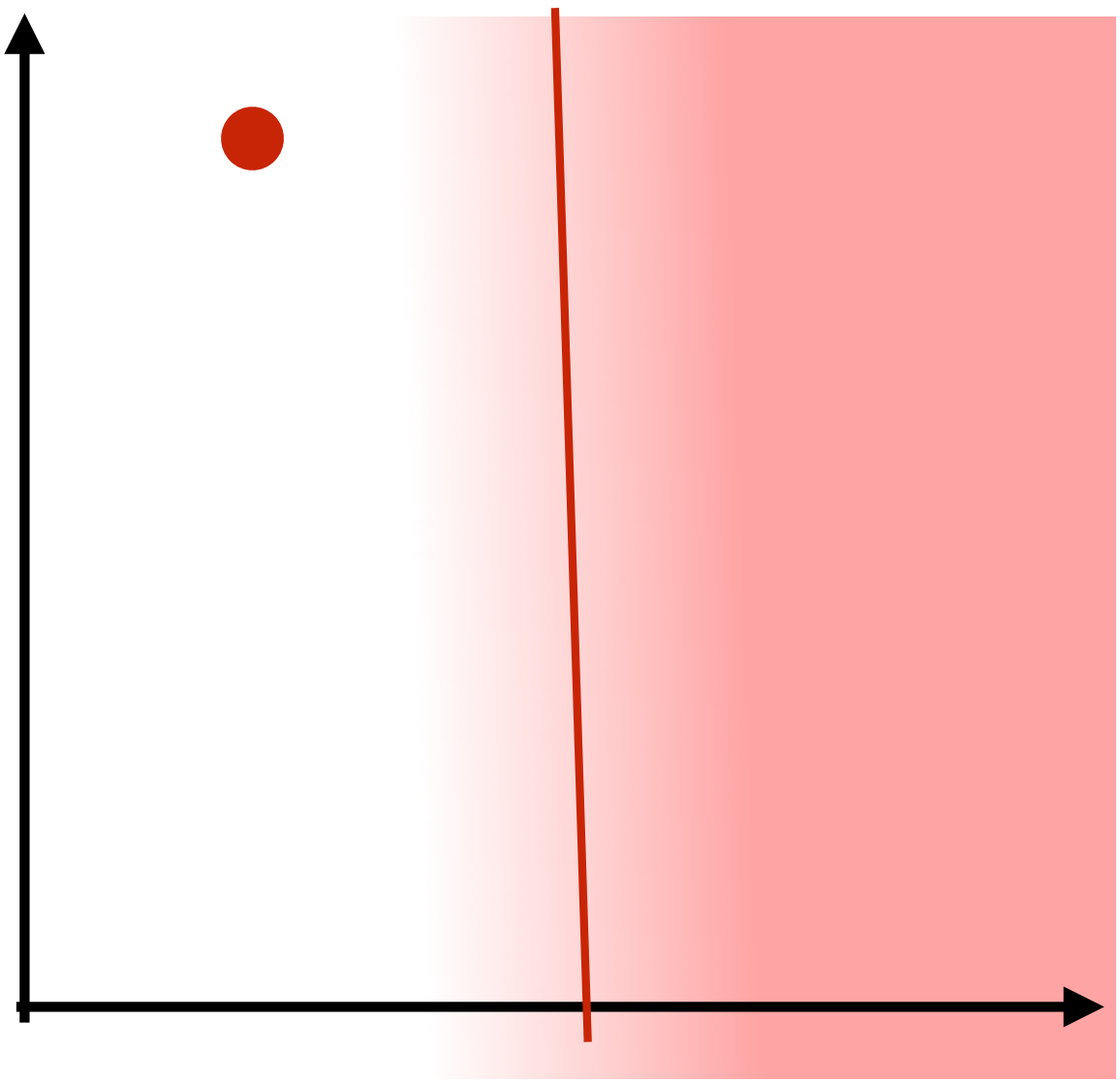
$$\mathbf{w} \leftarrow \mathbf{w} - a\frac{\partial L}{\partial \mathbf{w}}$$

$$b \leftarrow b - a\frac{\partial L}{\partial b}$$

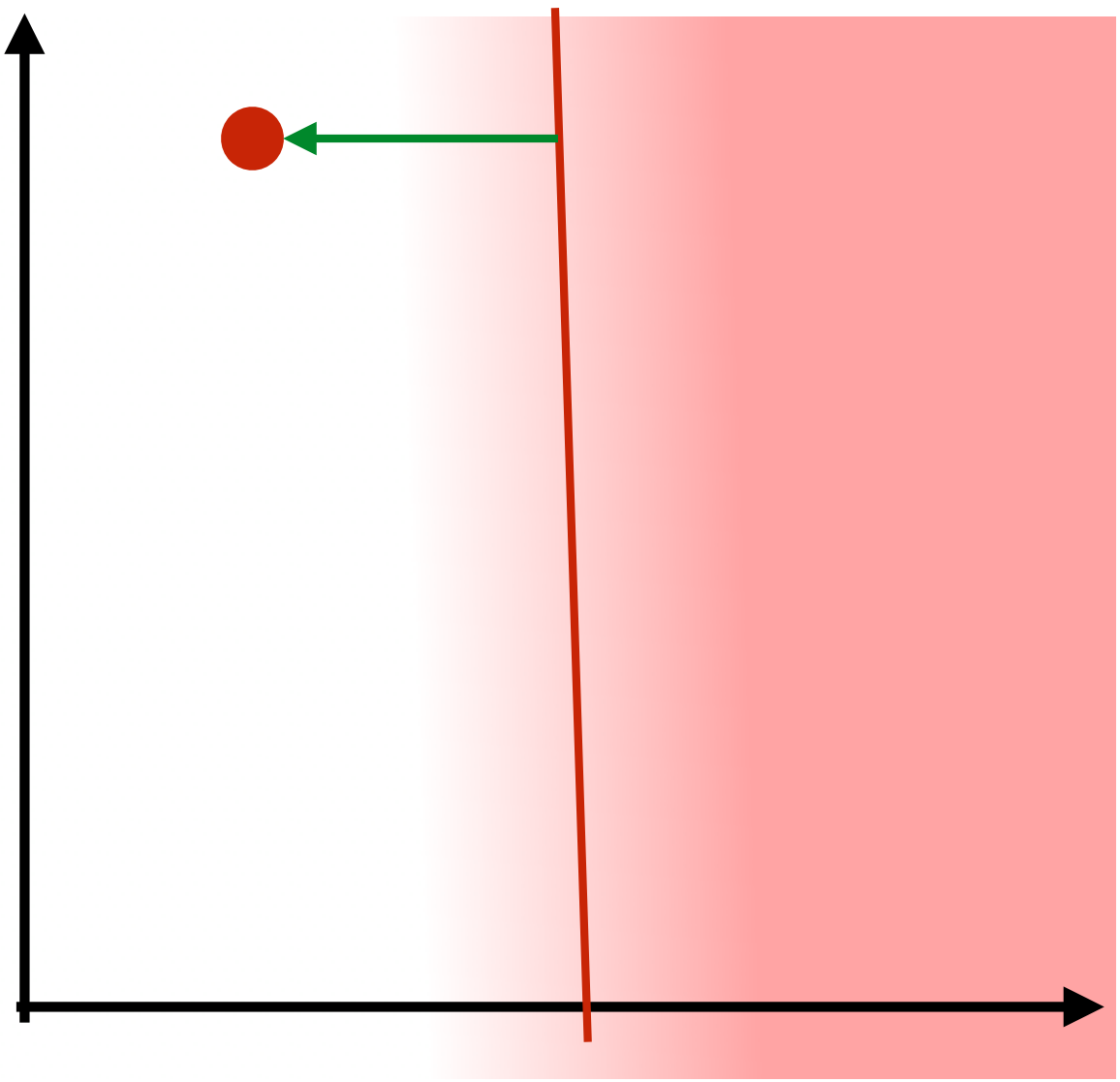
initialize **w** and b



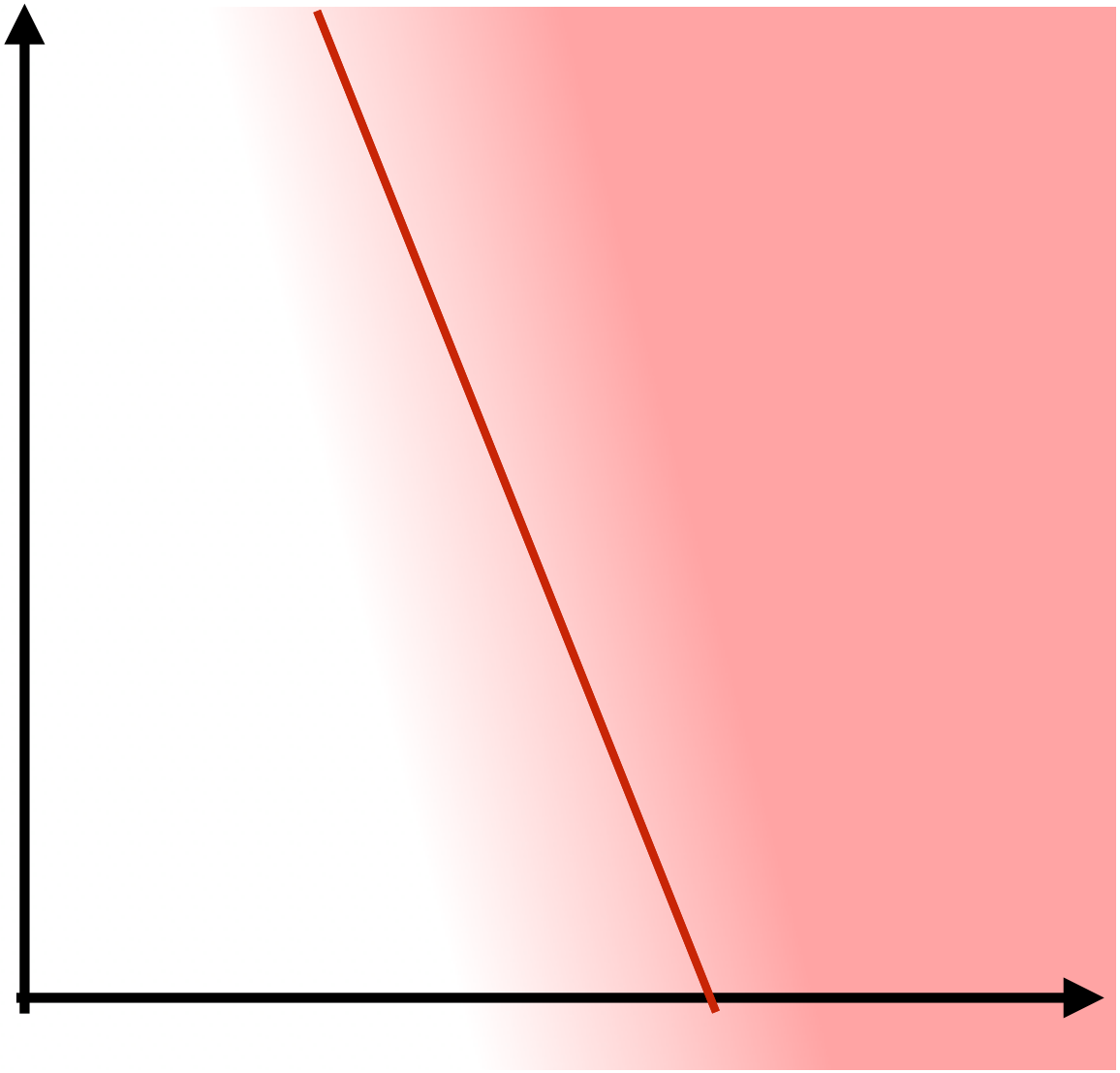
One training instance **x** and y



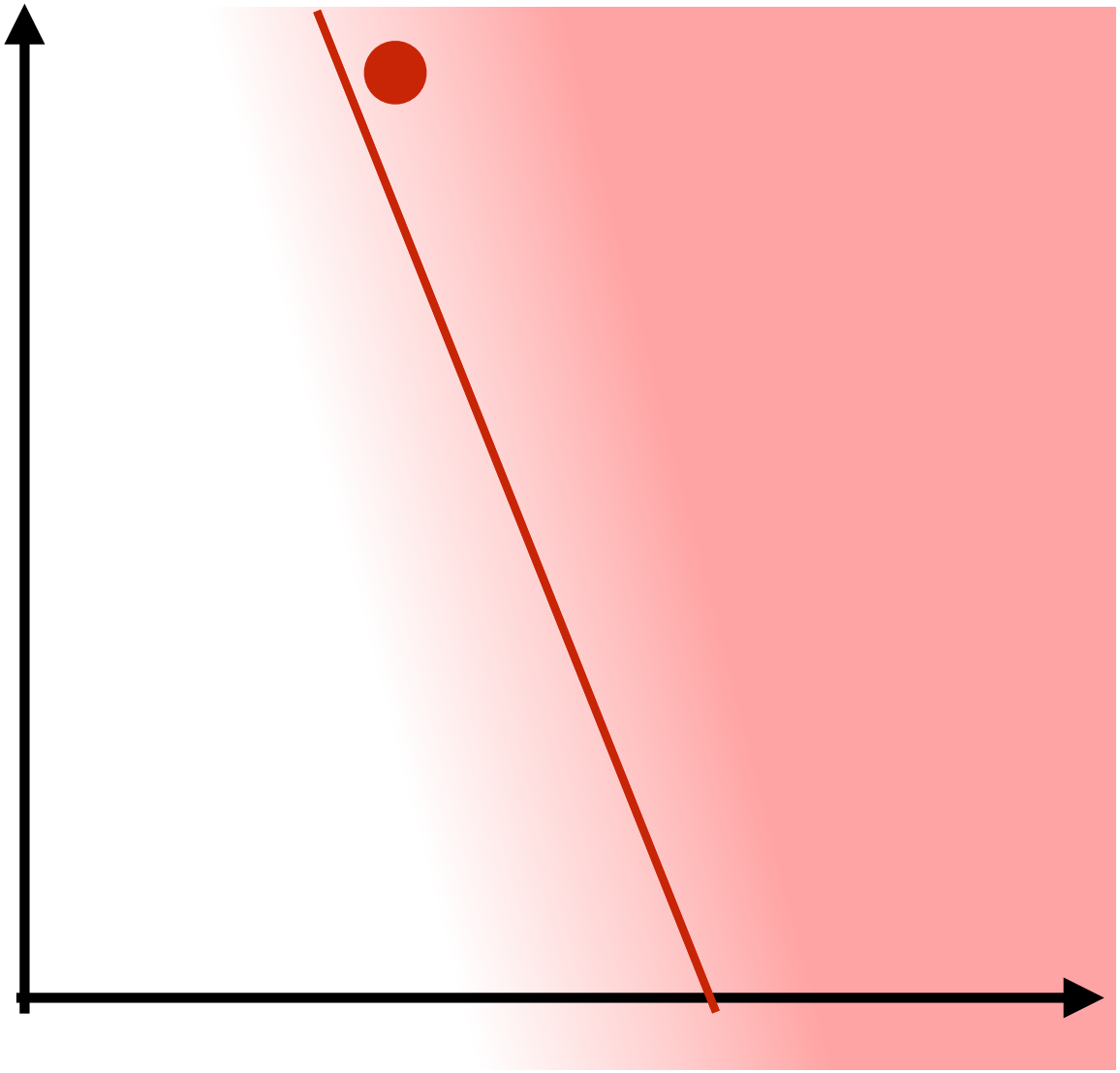
gradient of the loss w.r.t. parameters ($\mathbf{w}$, b)



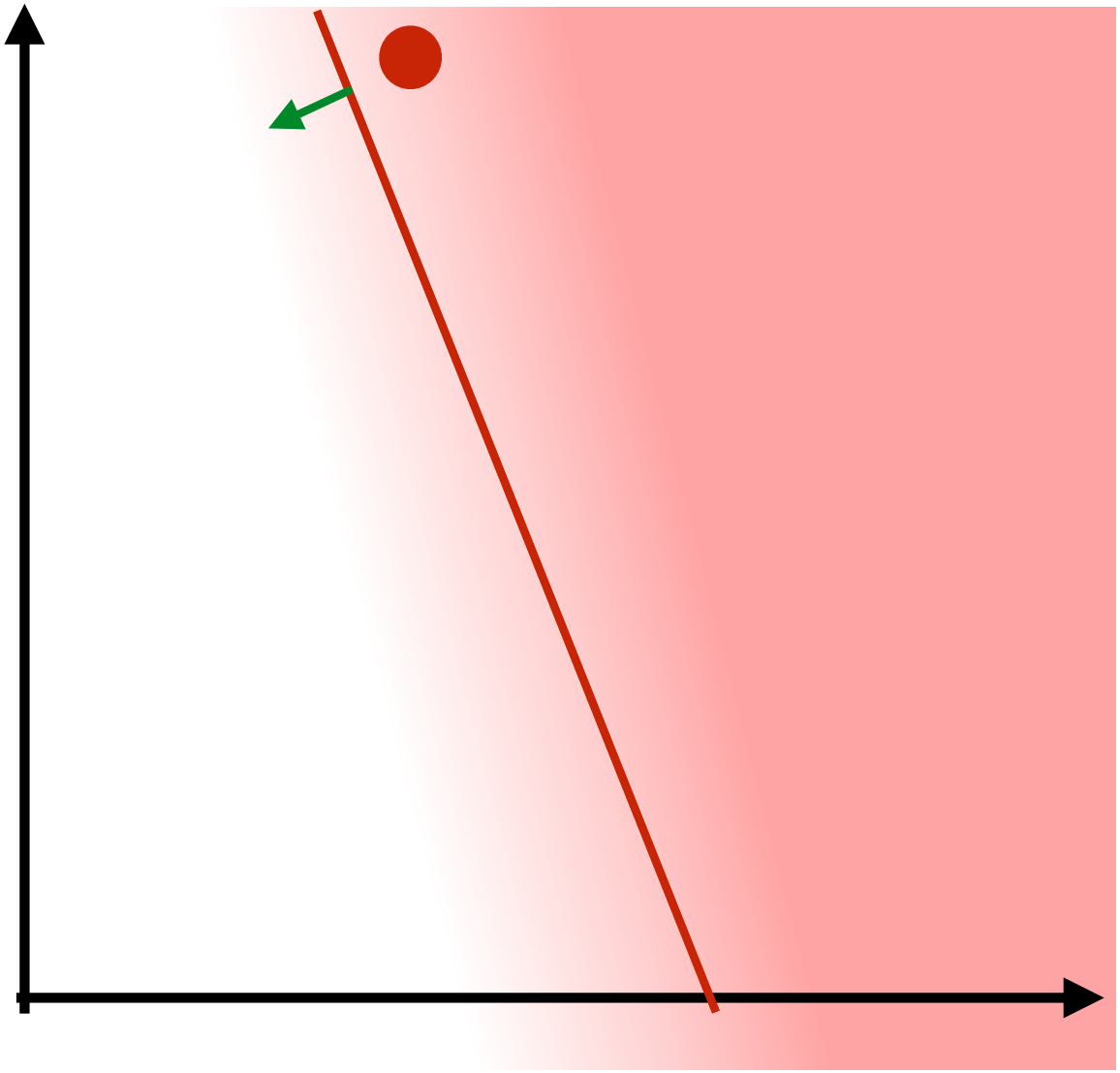
update model



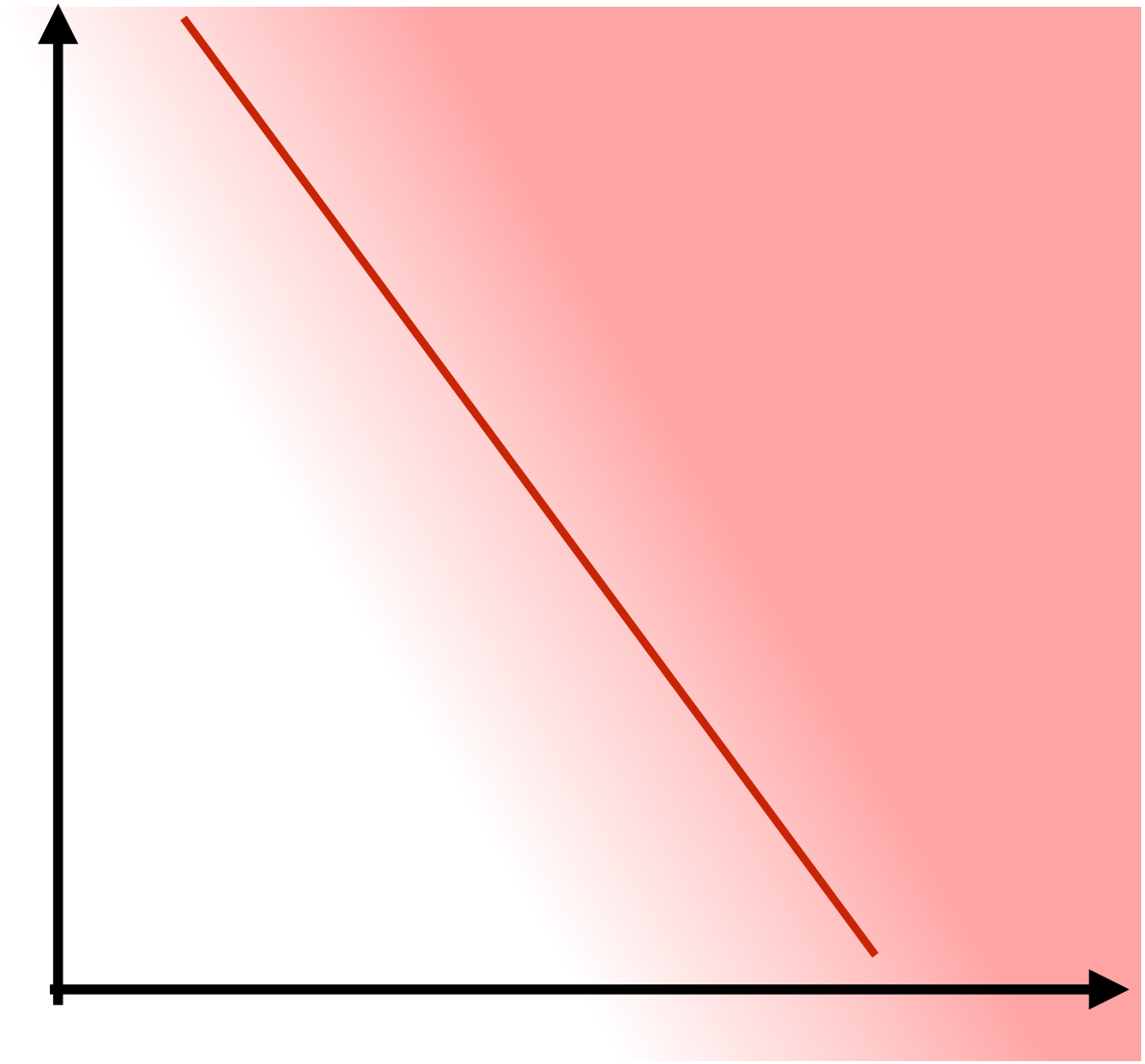
One training instance **x** and y



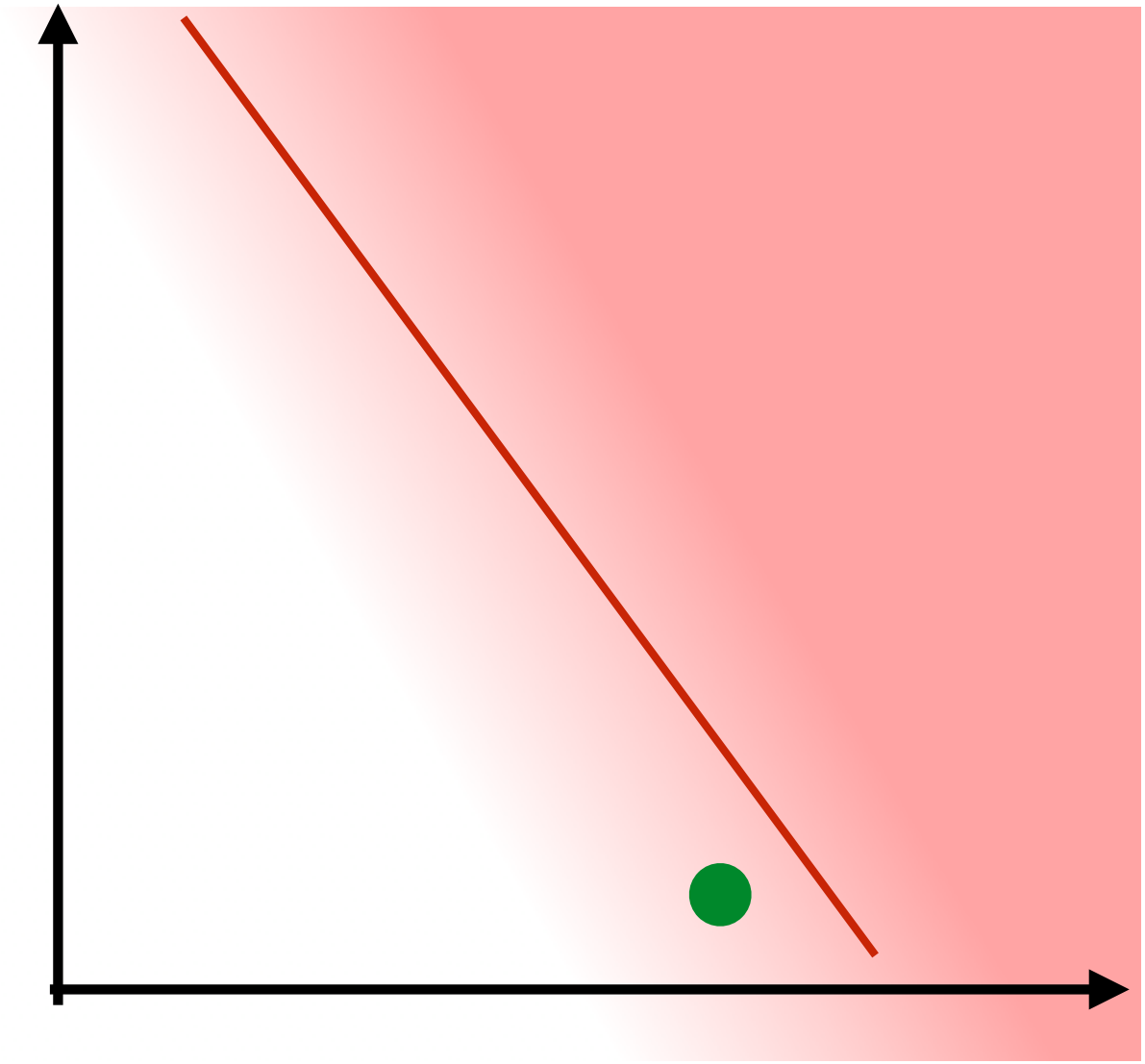
gradient of the loss w.r.t. parameters (**w**, b)



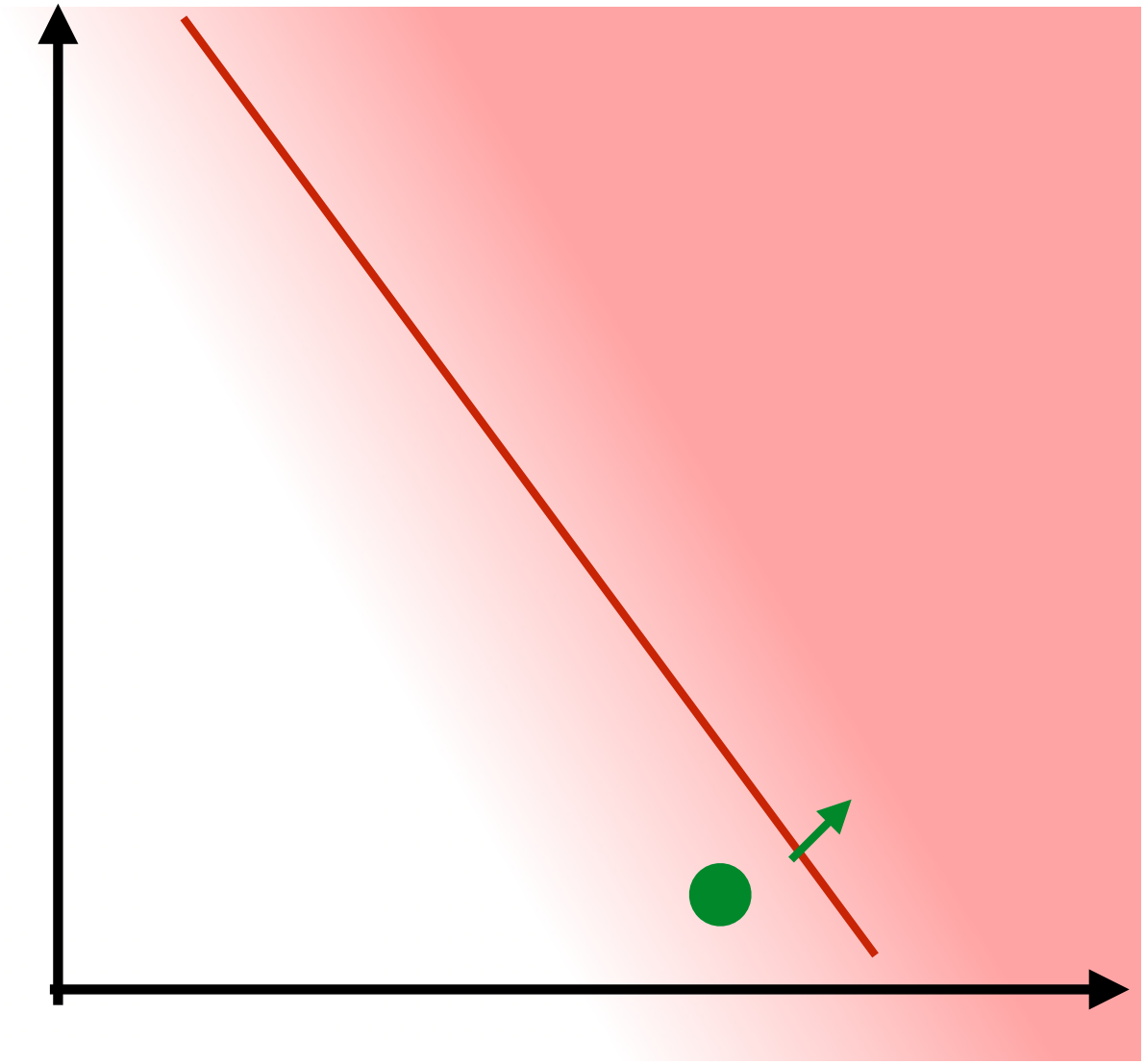
update model



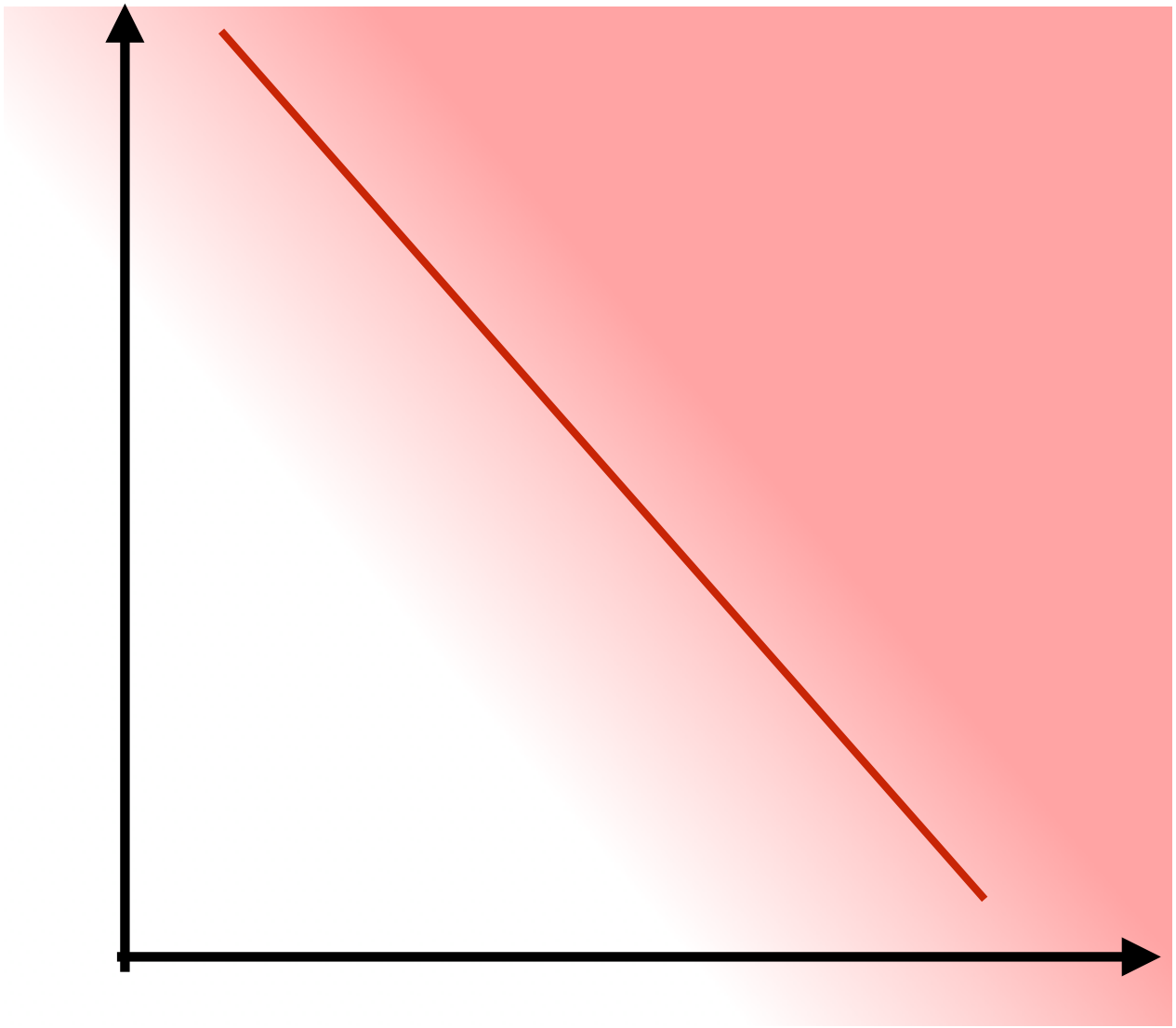
One training instance **x** and y



gradient of the loss w.r.t. parameters (**w**, b)

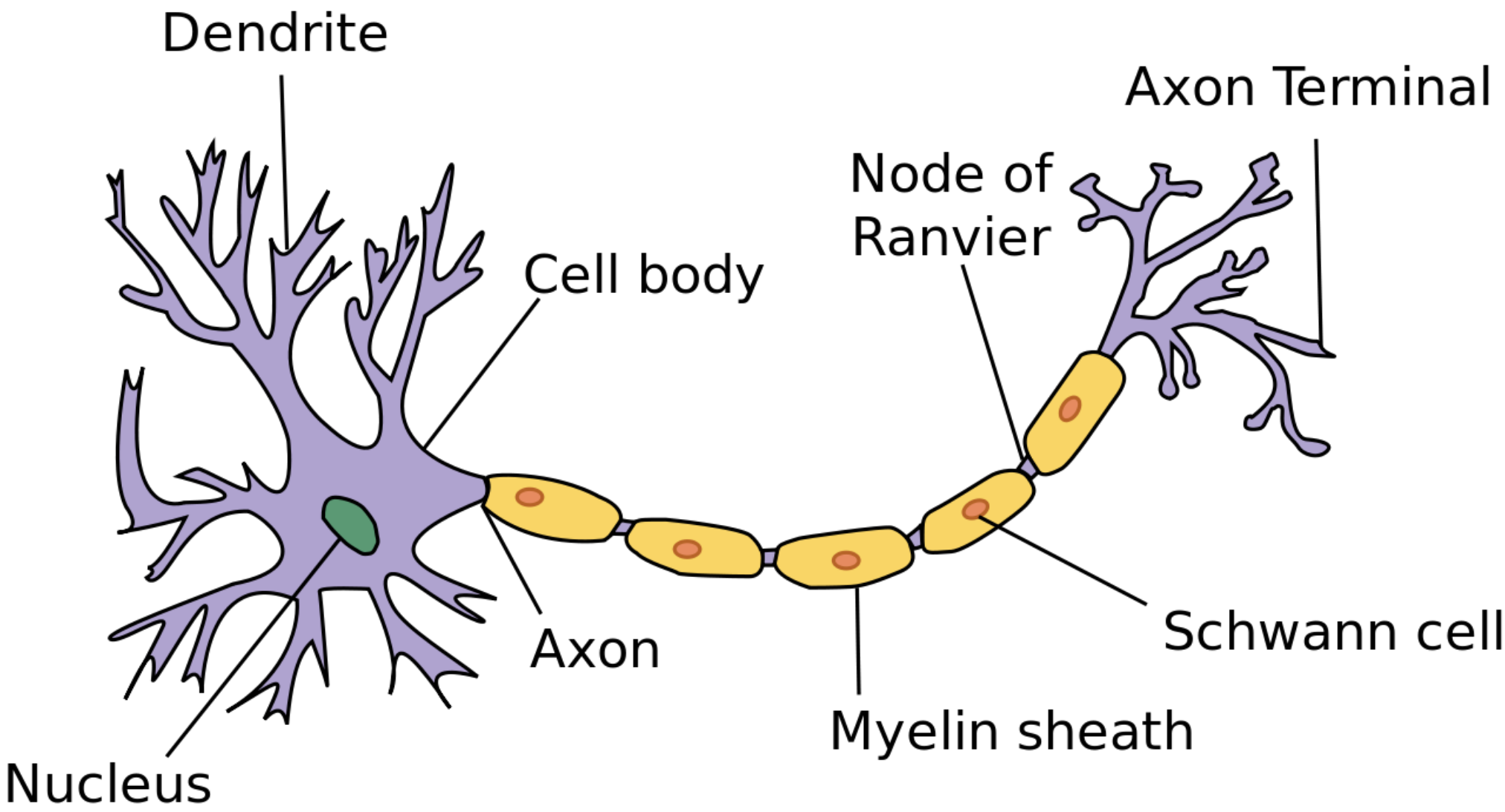


update model parameters (**w**, b)



Softmax Regression

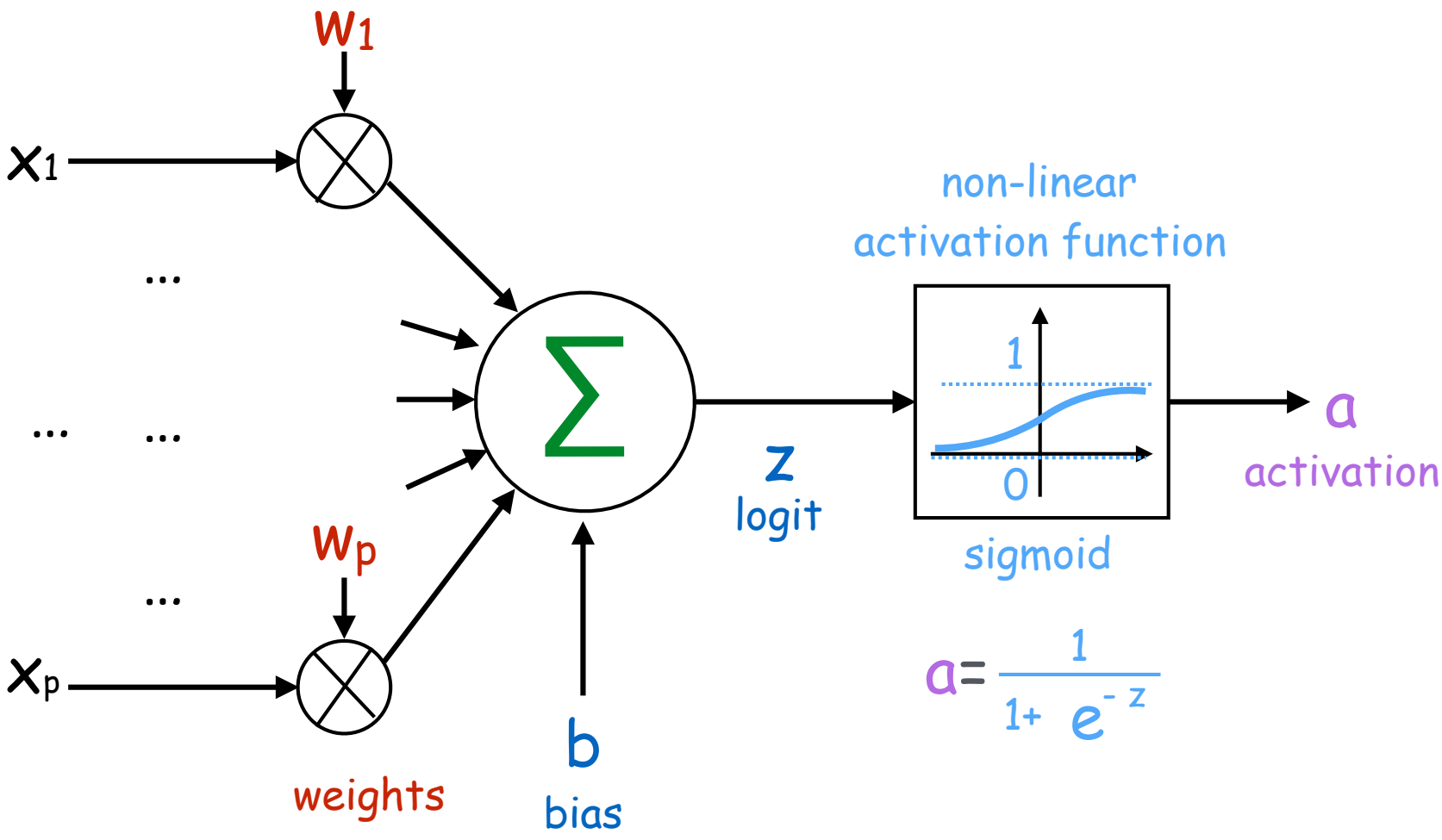
Neuron



https://www.youtube.com/watch?v=fHRC8SlLcH0

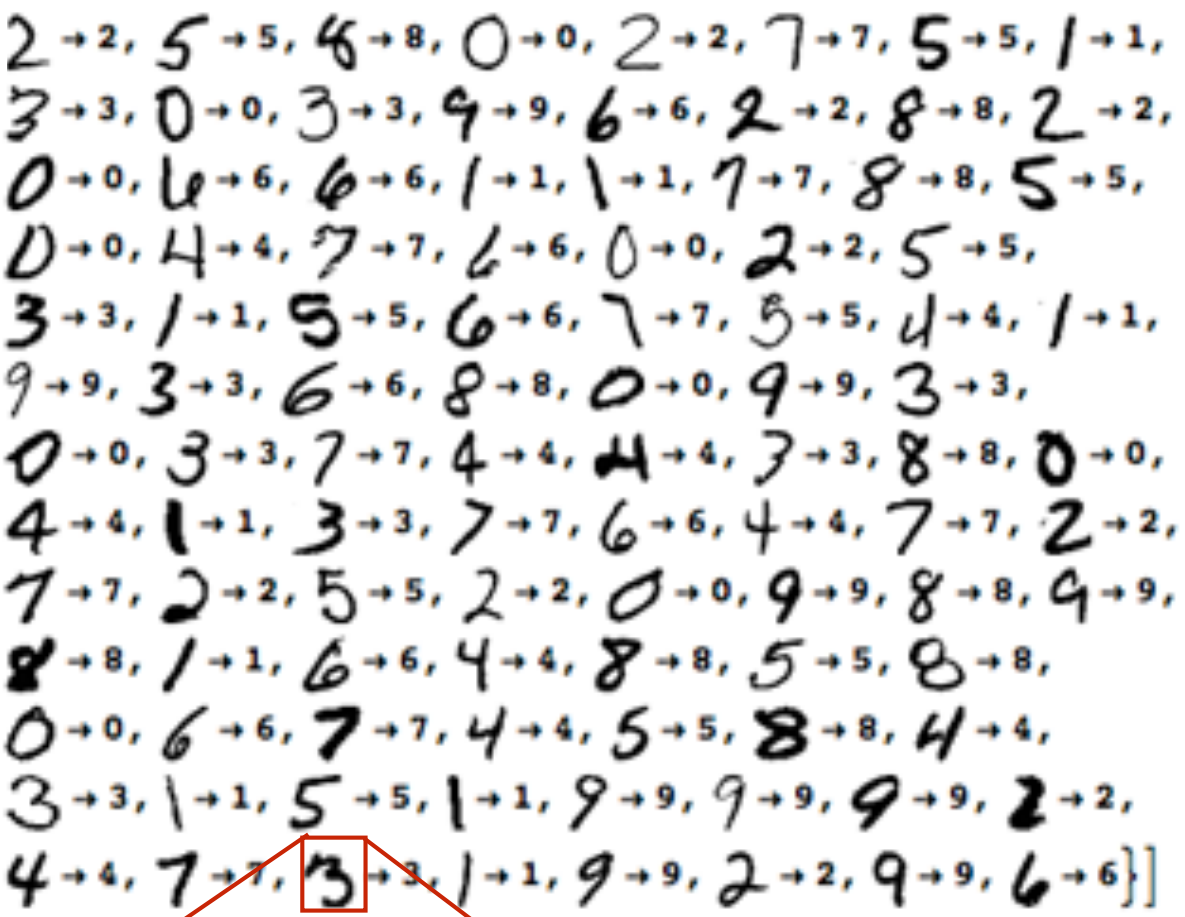
Logistic Regression

Demo: http://playground.tensorflow.org/



Multi-class Classification Problem

data set



candidate labels (classes)

0

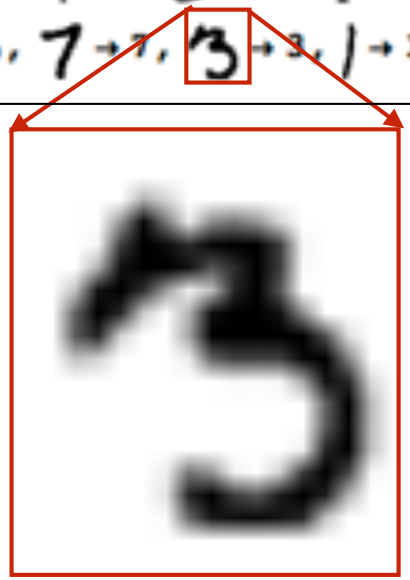
1

2

⋮

9

input (instance)



output (label)

0 or 1 or … or 9

Feature Matrix X

(n by p)

Feature (pixel)

Instance

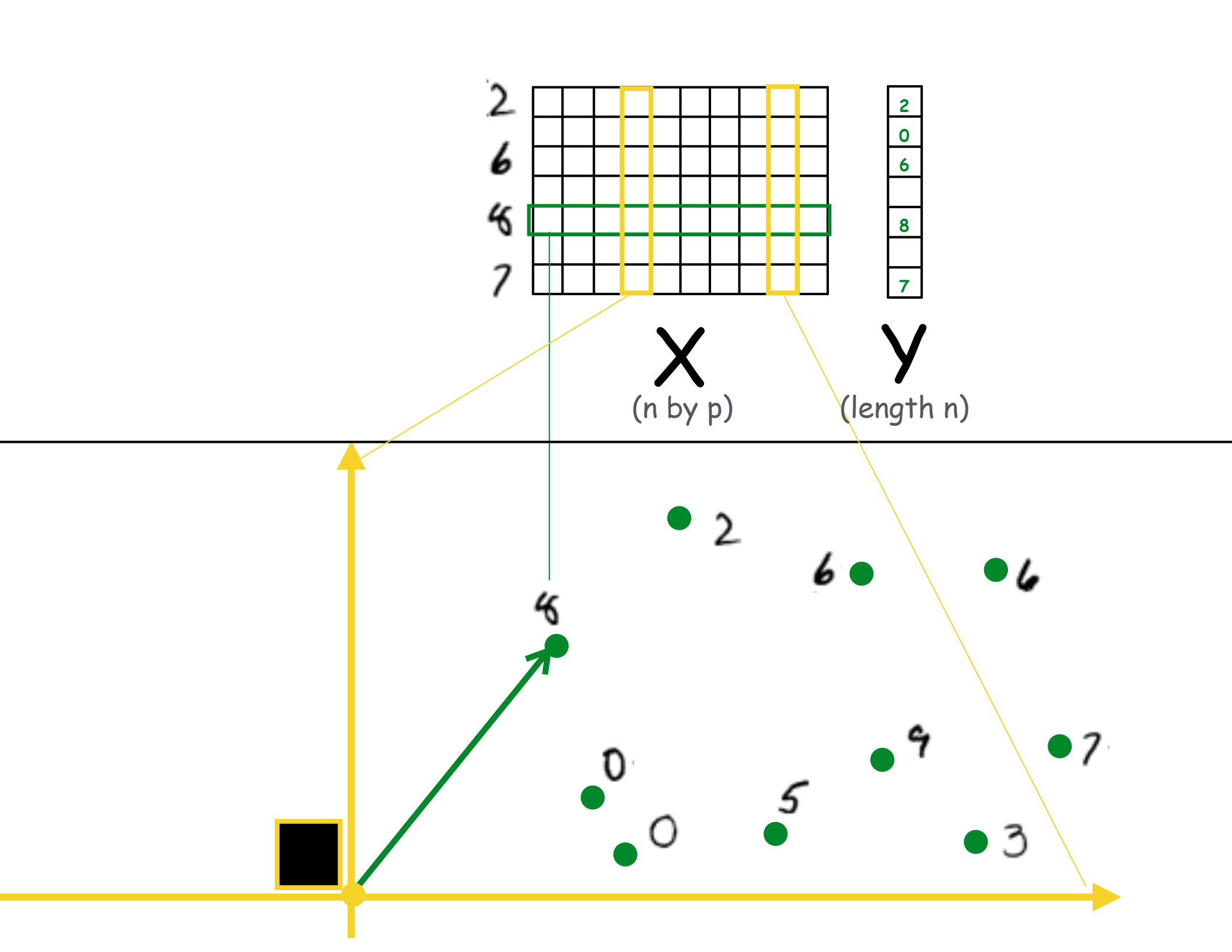
		1	2				…					p
x_1	2	1	3	4	3	8	3	5	7	9	7	3	4
x_2	5	3	3	5	7	7	0	4	1	2	1	9	7
	0	7	0	4	1	1	4	3	7	8	6	2	7
	3	1	4	3	7	9	7	3	2	7	0	4	1
	8	7	7	3	2	2	1	9	8	1	4	3	7
…	0	2	1	9	8	8	6	2	0	7	7	3	2
	9	8	6	2	0	0	4	1	1	4	1	9	8
	6	0	2	1	4	1	3	7	9	7	6	2	0
	7	3	5	3	3	7	3	2	2	1	2	1	3
x_n	6	1	7	2	3	2	2	1	2	3	5	3	1

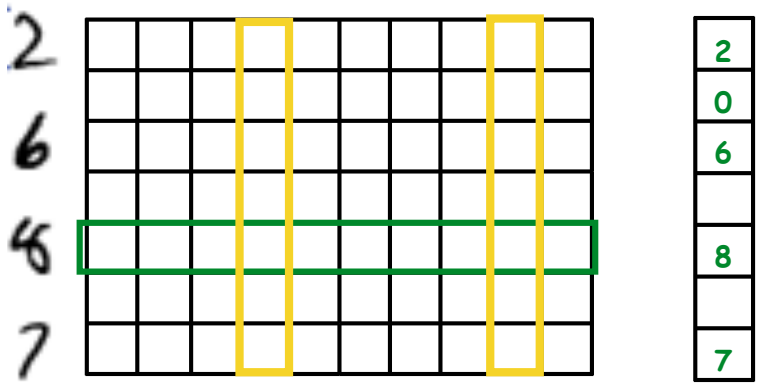
n - # instances p - # features

Label Vector y

(length n)

y_1	2
y_2	5
	0
	3
	8
…	0
	9
	6
	7
y_n	6


2
6
4
7
2
0
6
8
7
X
(n by p)
y
(length n)
2
6
6
4
0
0
5
9
3
7

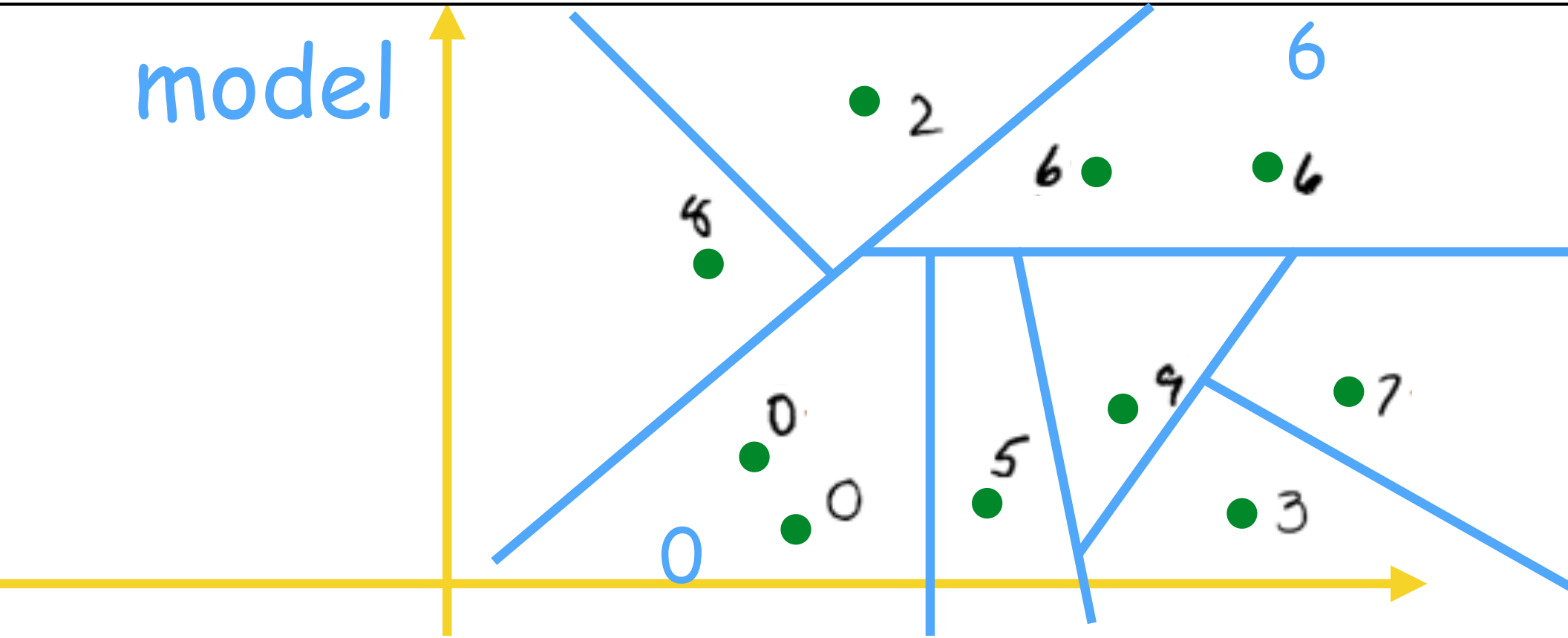


X

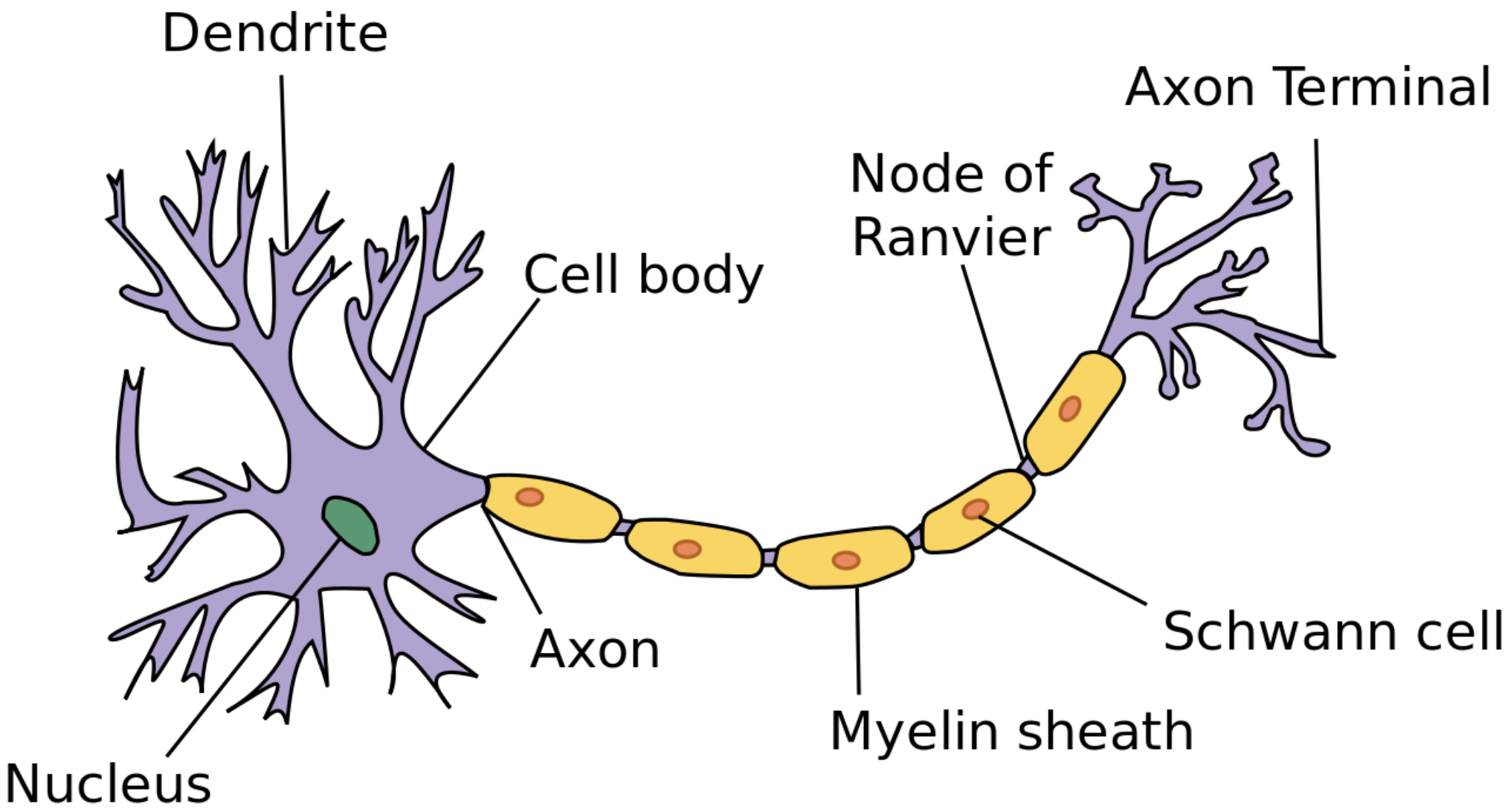
(n by p)

y

(length n)

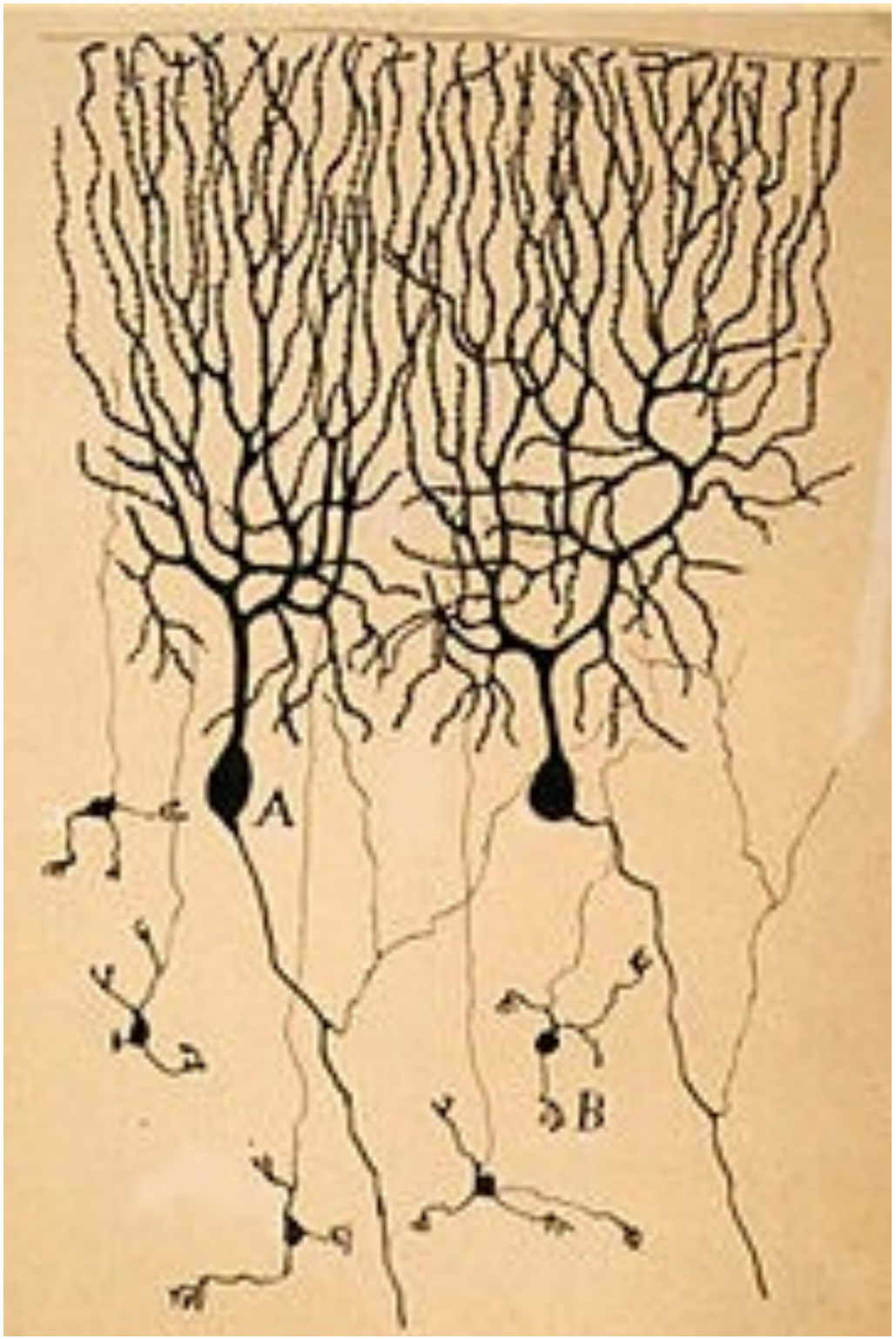


Neuron

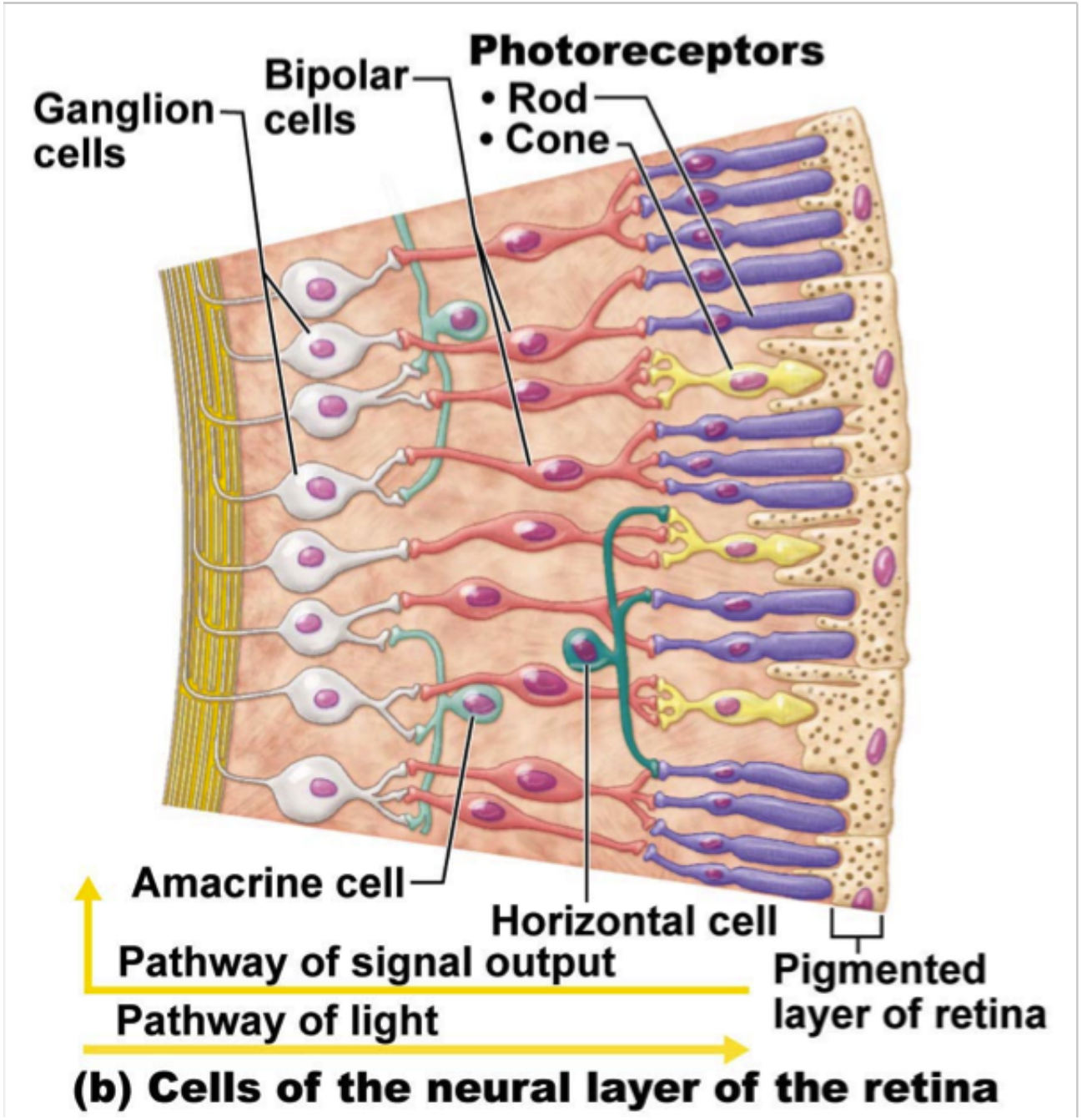

Dendrite
Axon Terminal
Node of Ranvier
Cell body
Axon
Schwann cell
Myelin sheath
Nucleus

Neuron Layer

Santiago Ramón y Cajal
Nobel Prize in Medicine
1899

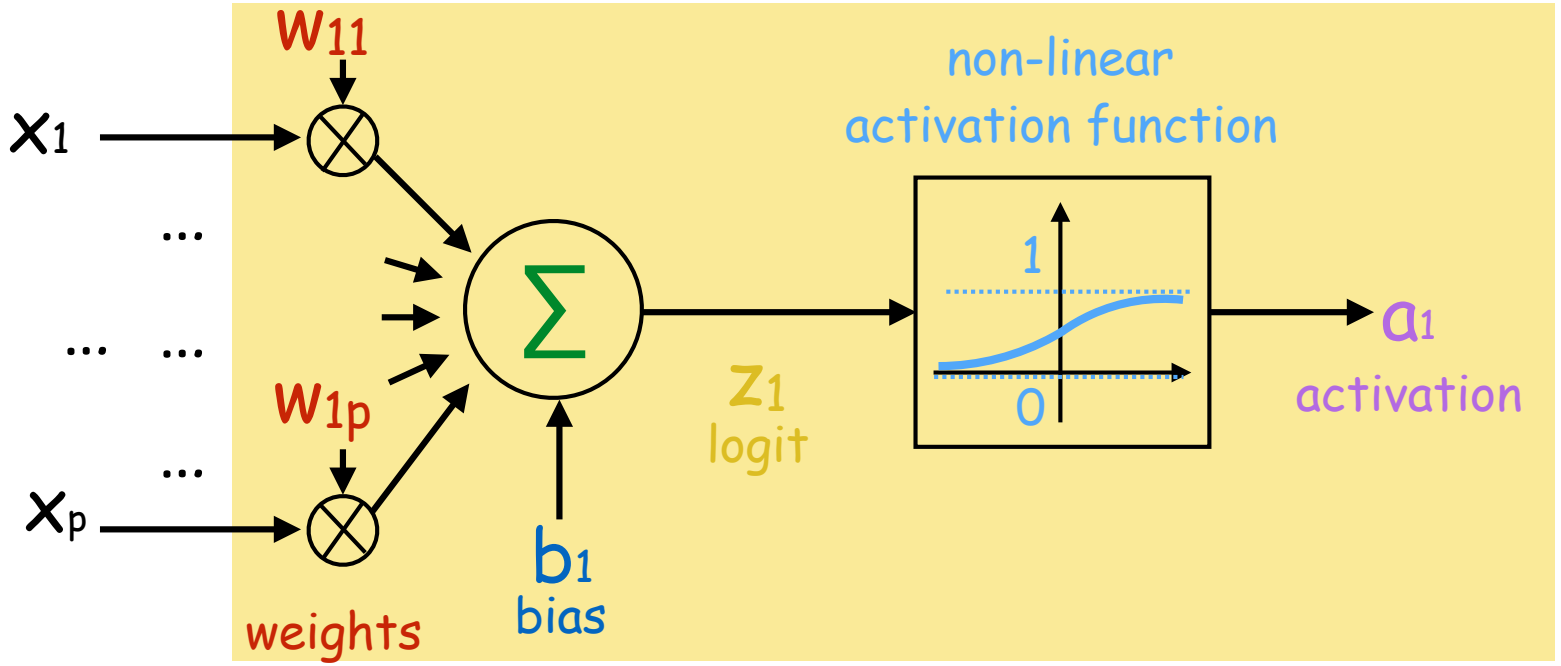


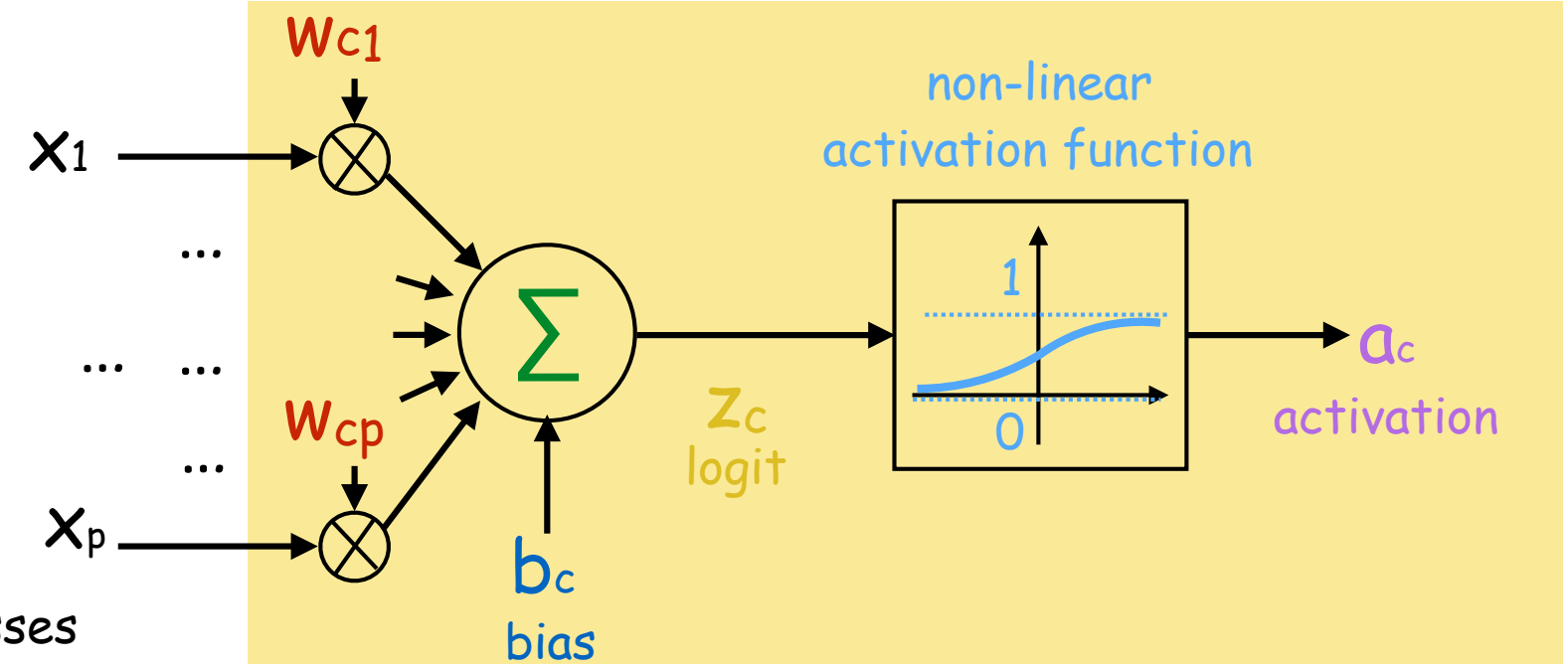
Neural Layer



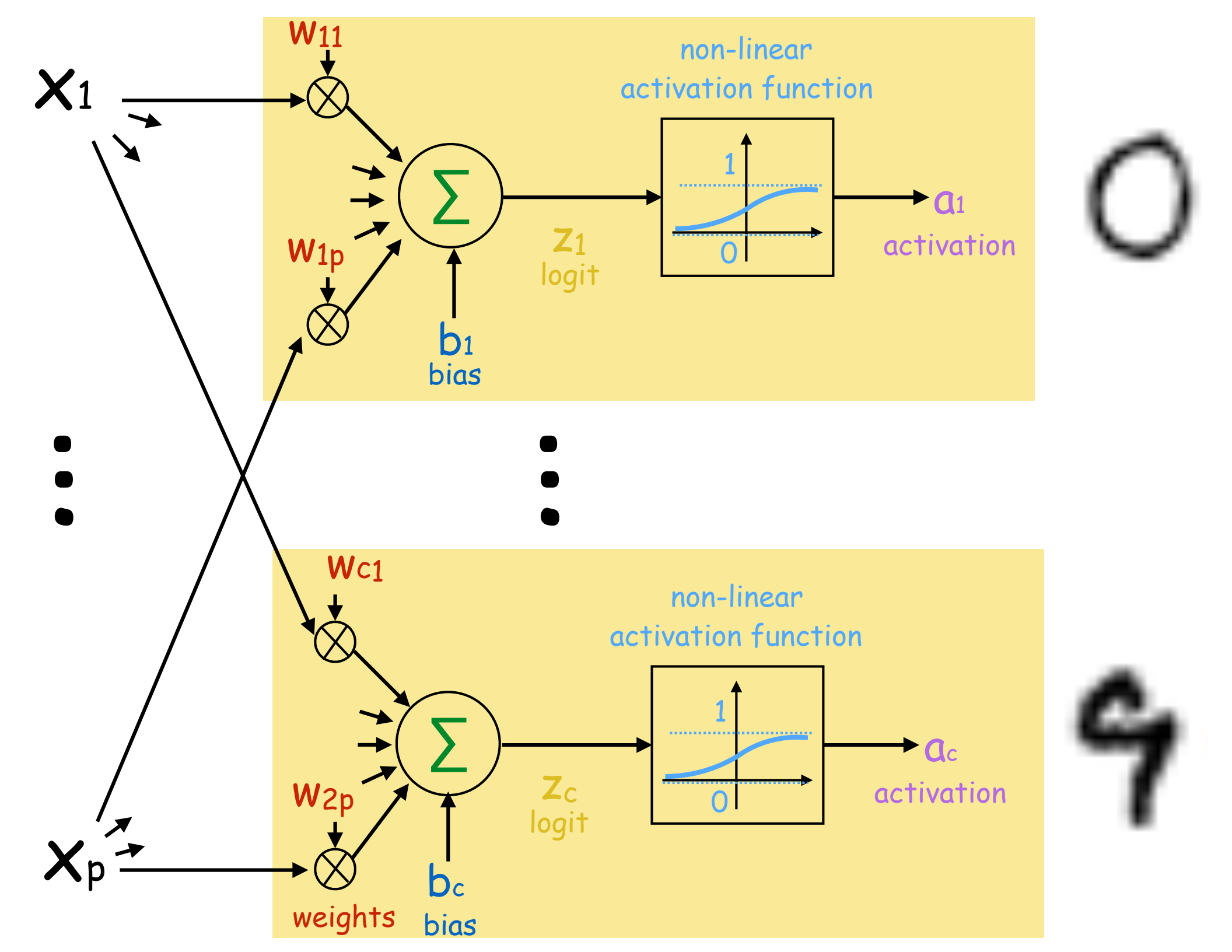
Demo

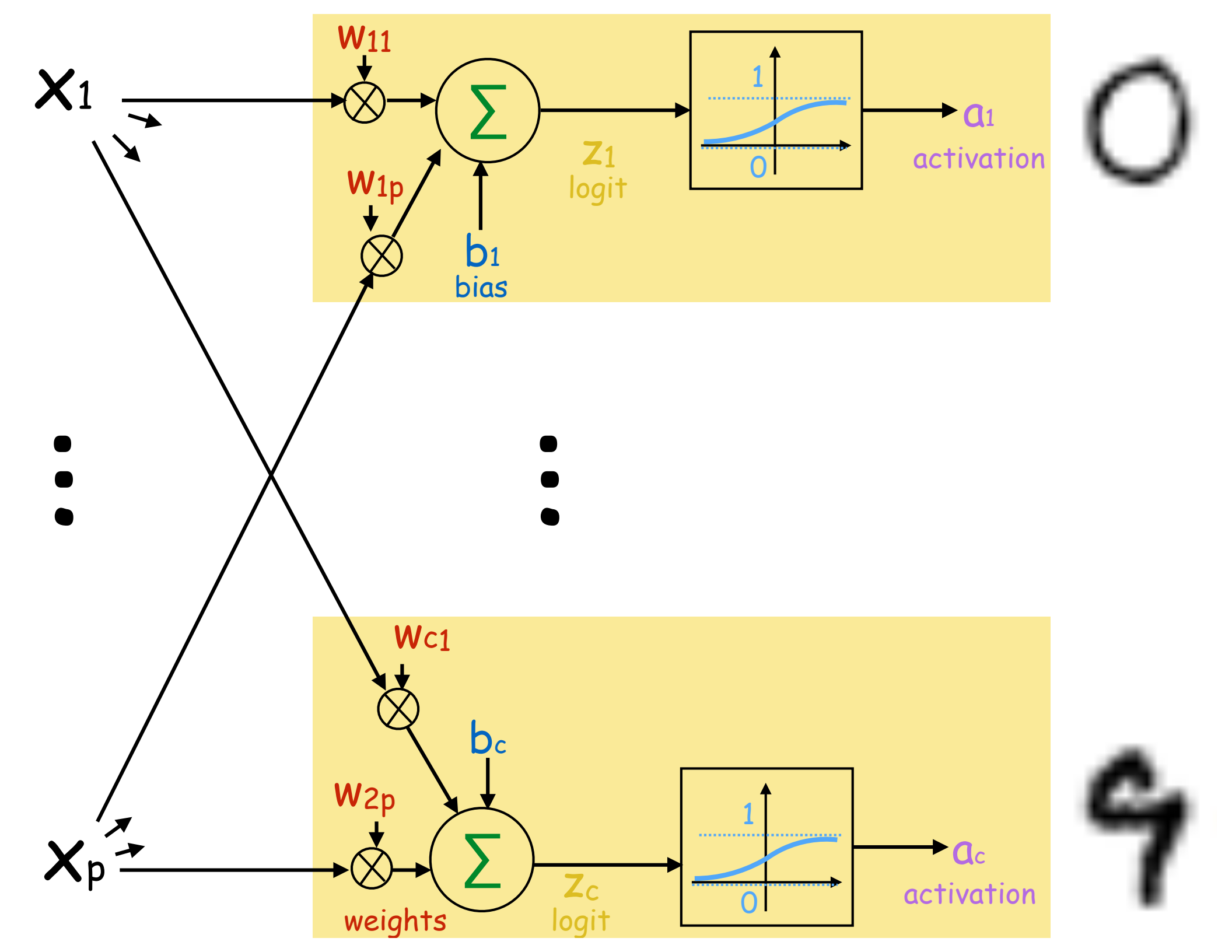
https://youtu.be/3JQ3hYko51Y

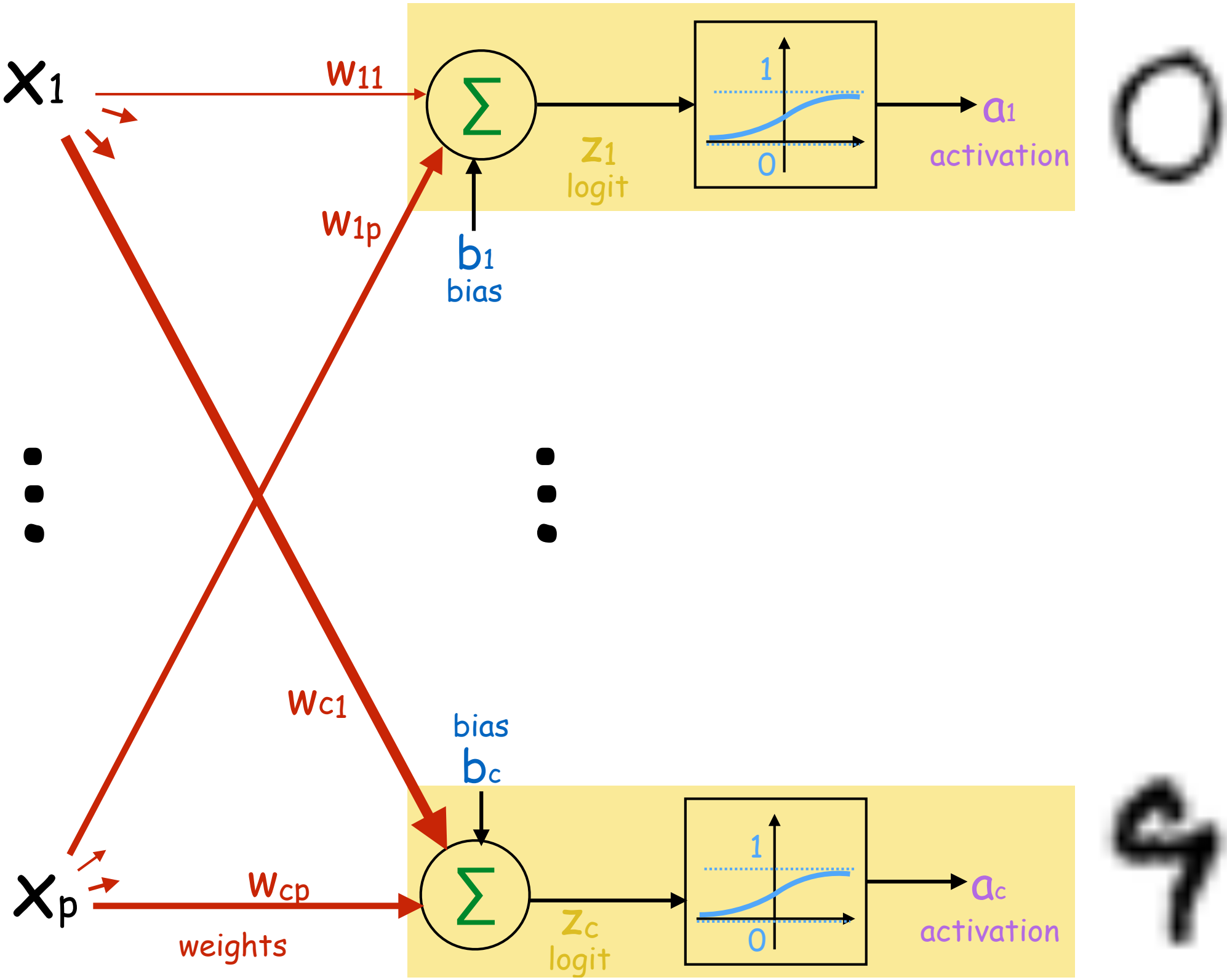

w_{11}
x_1
...
... ...
...
w_{1p}
x_p
weights
b_1
bias
z_1
logit
non-linear
activation function
1
0
a_1
activation

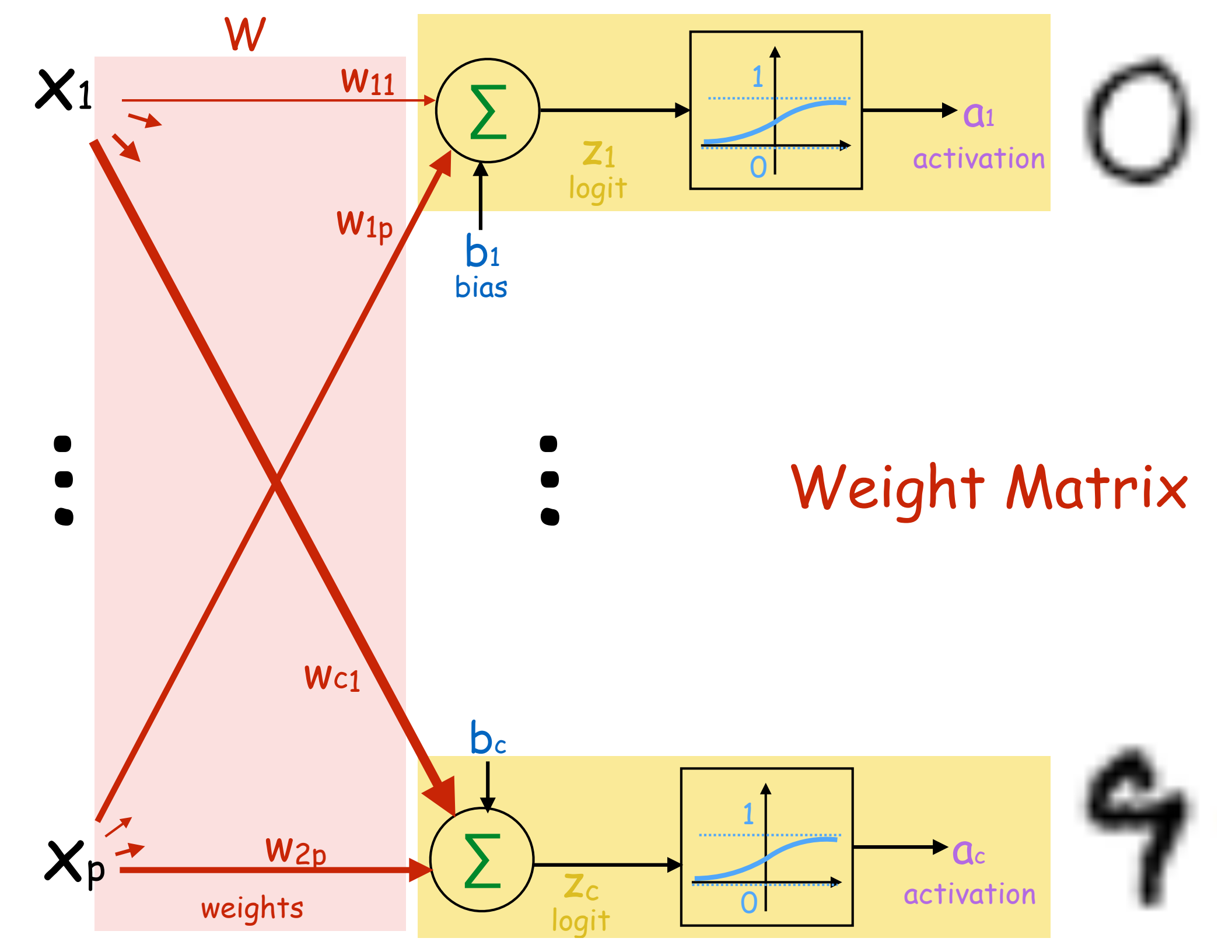

w_{c1}
x_1
...
... ...
...
w_{cp}
x_p
b_c
bias
z_c
logit
non-linear
activation function
1
0
a_c
activation

c: # classes

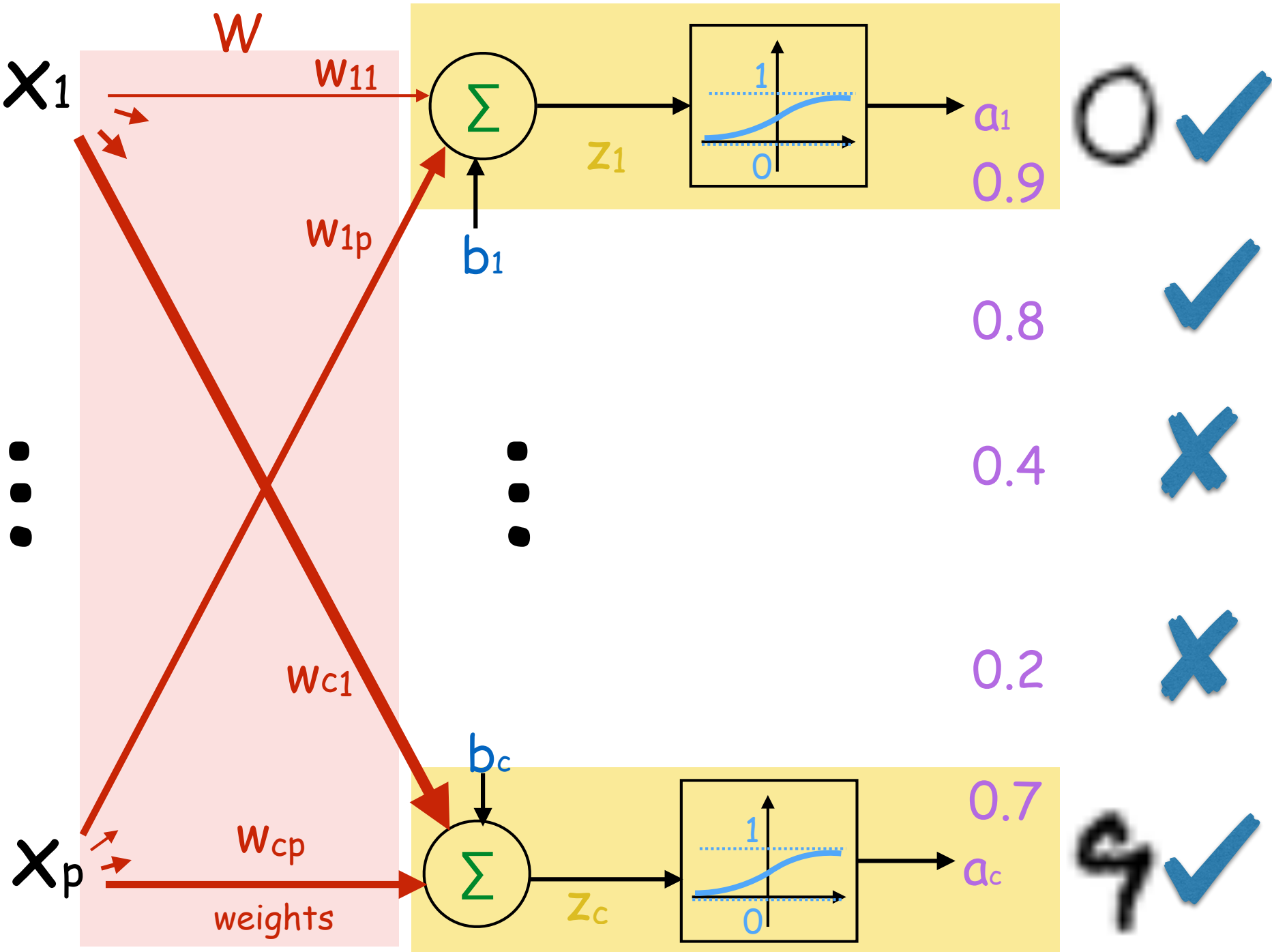

x_1
w_{11}
w_{1p}
Σ
b_1
bias
z_1
logit
non-linear
activation function
1
0
a_1
activation
x_p
w_{c1}
w_{2p}
weights
Σ
b_c
bias
z_c
logit
non-linear
activation function
1
0
a_c
activation


x_1
w_{11}
w_{1p}
b_1
bias
z_1
logit
1
0
a_1
activation
x_p
w_{c1}
b_c
w_{2p}
weights
z_c
logit
1
0
a_c
activation

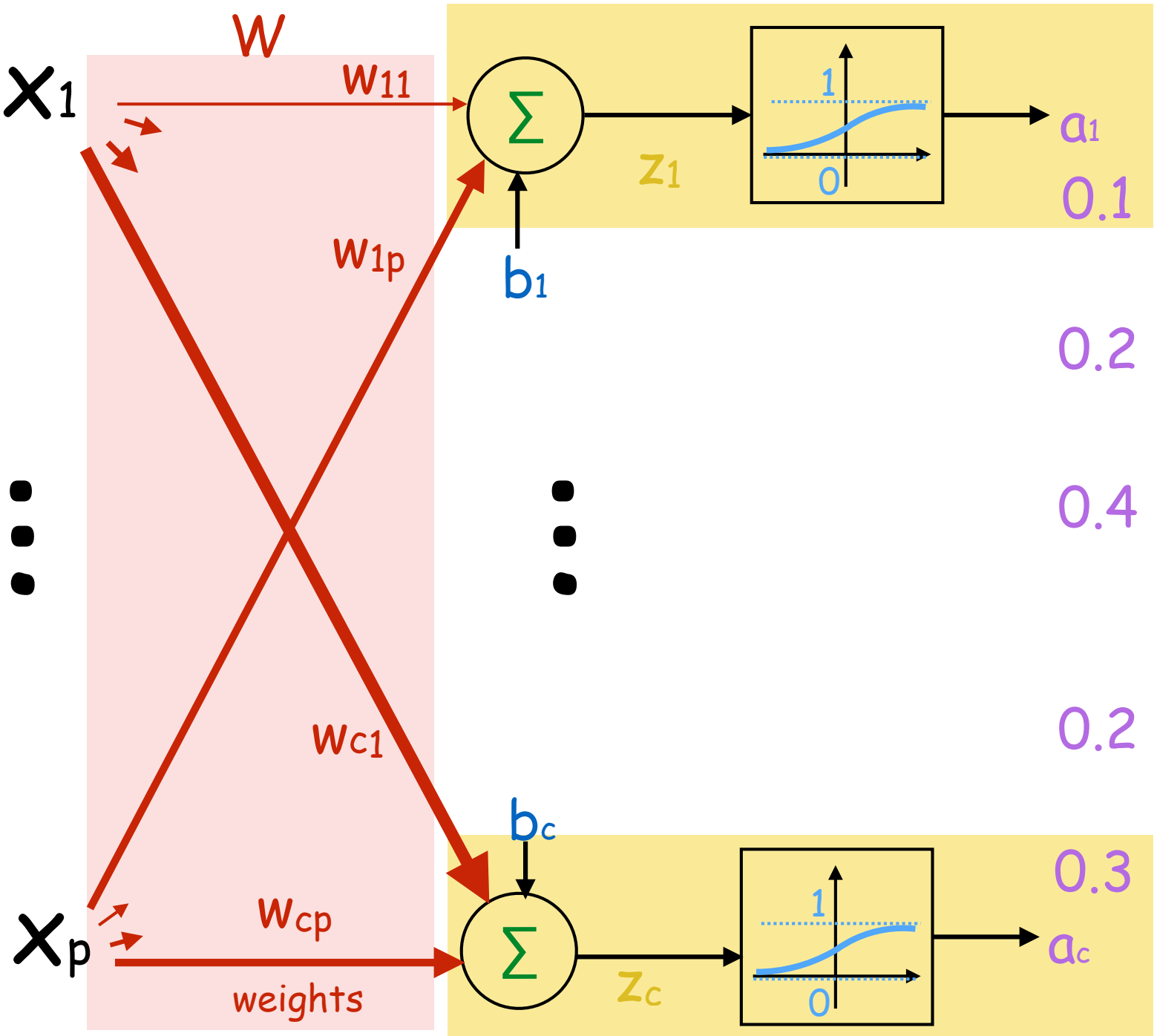

x_1
w_{11}
w_{1p}
b_1
bias
z_1
logit
1
0
a_1
activation
w_{c1}
bias
b_c
x_p
w_{cp}
weights
z_c
logit
1
0
a_c
activation


W
x_1
w_{11}
w_{1p}
b_1
bias
z_1
logit
1
0
a_1
activation
w_{c1}
b_c
x_p
w_{2p}
weights
z_c
logit
1
0
a_c
activation
Weight Matrix

Independent Outputs



Independent Outputs



0.1

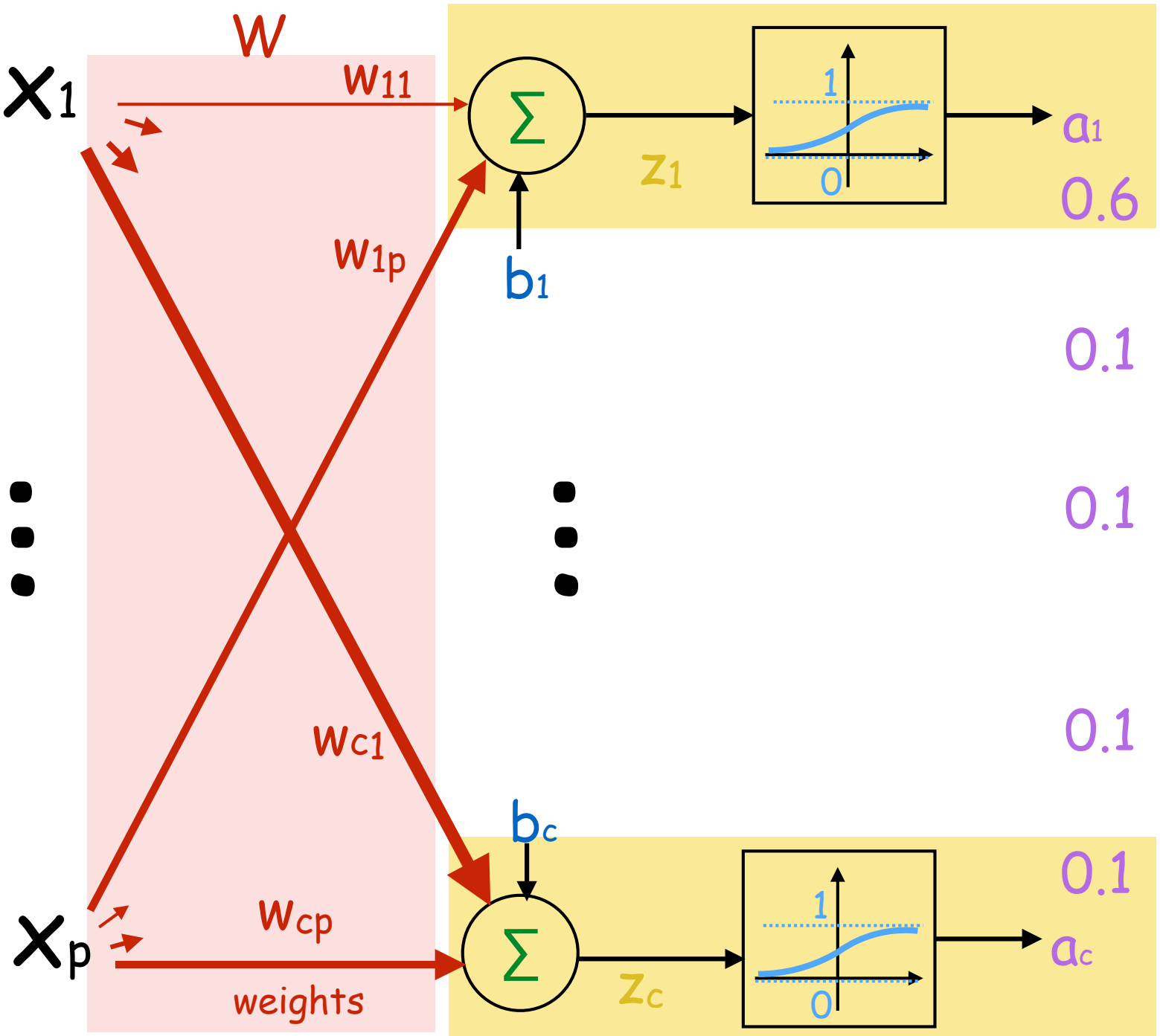
0.2

0.4

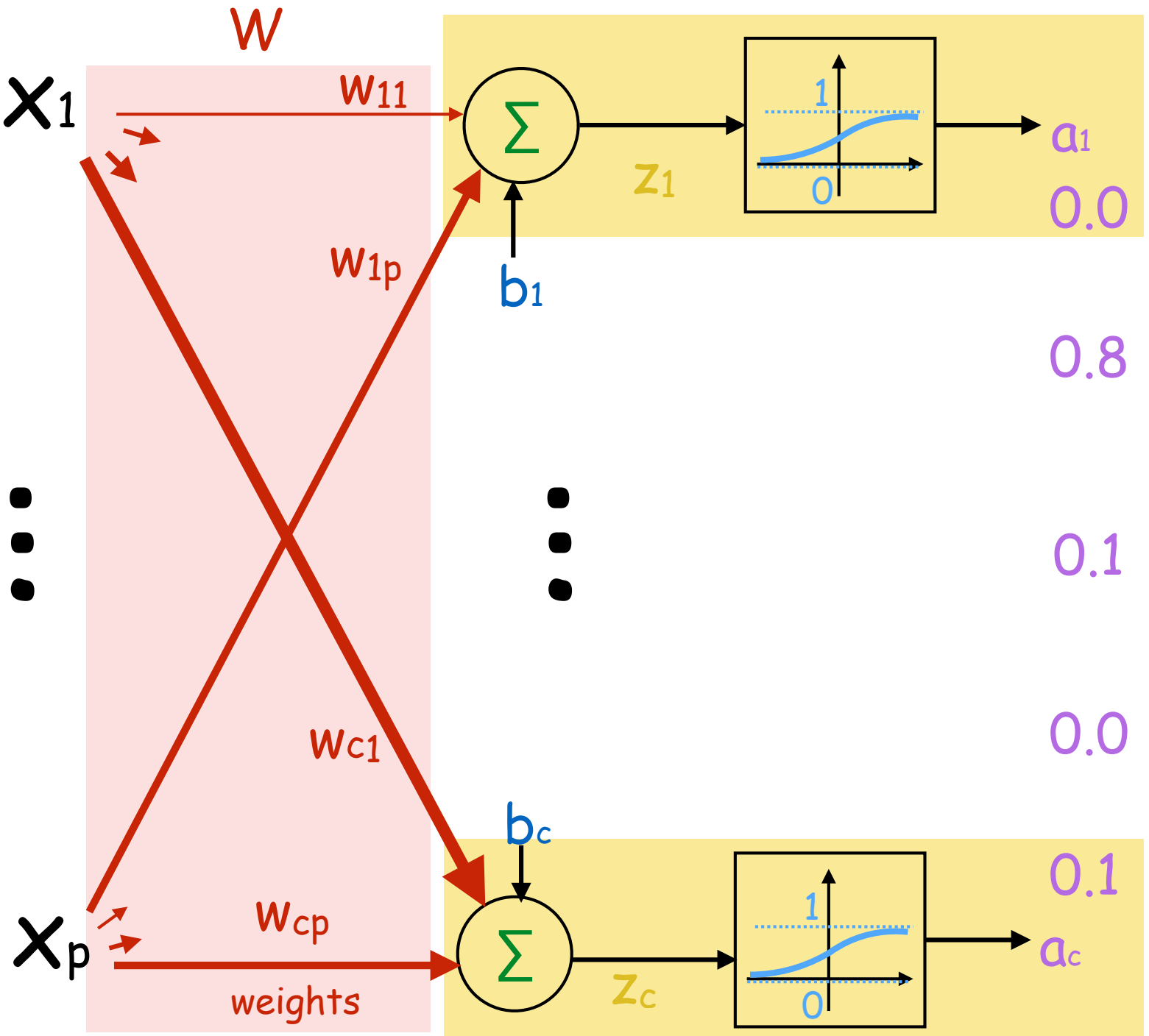
0.2

0.3

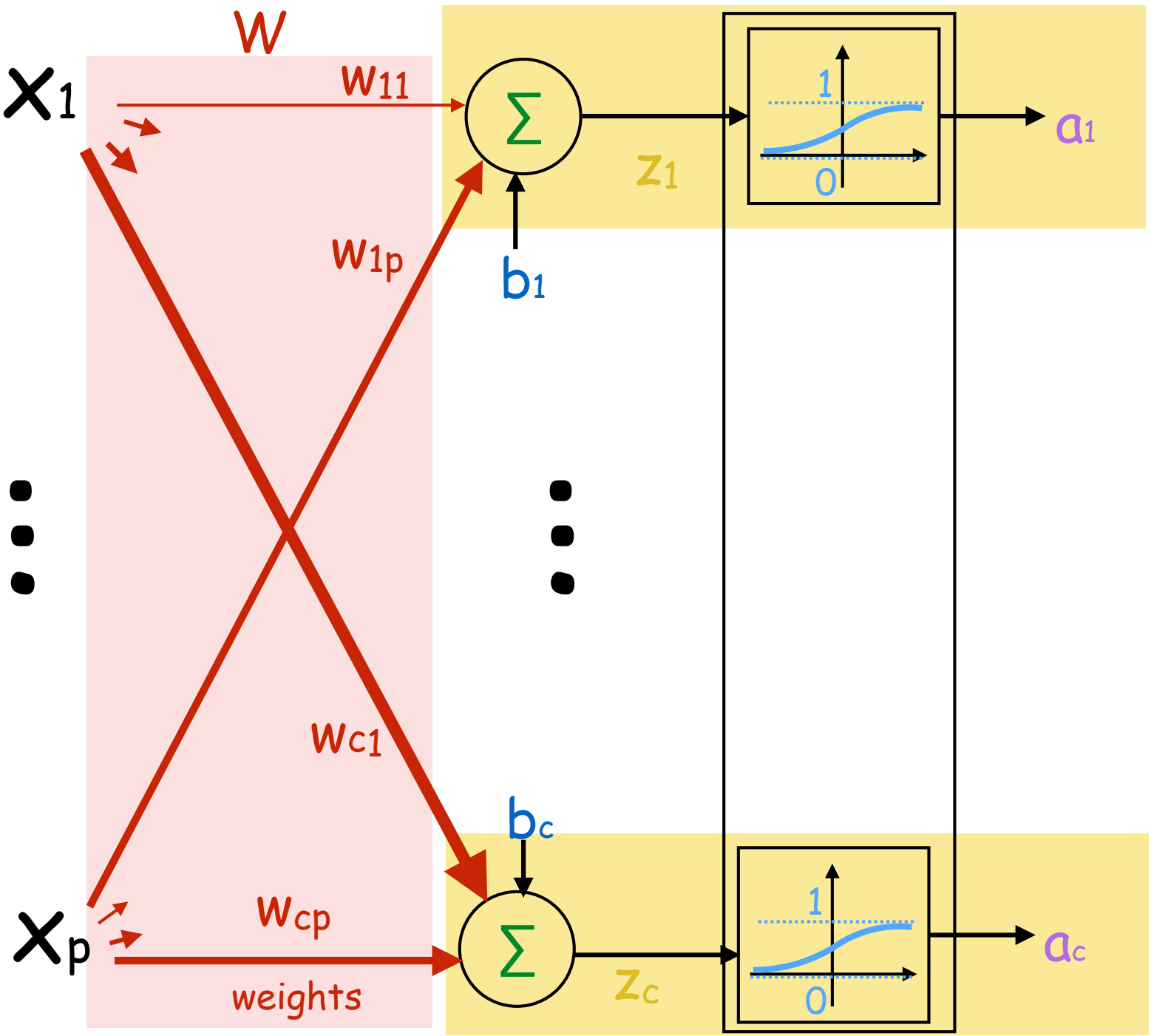
The outputs we need



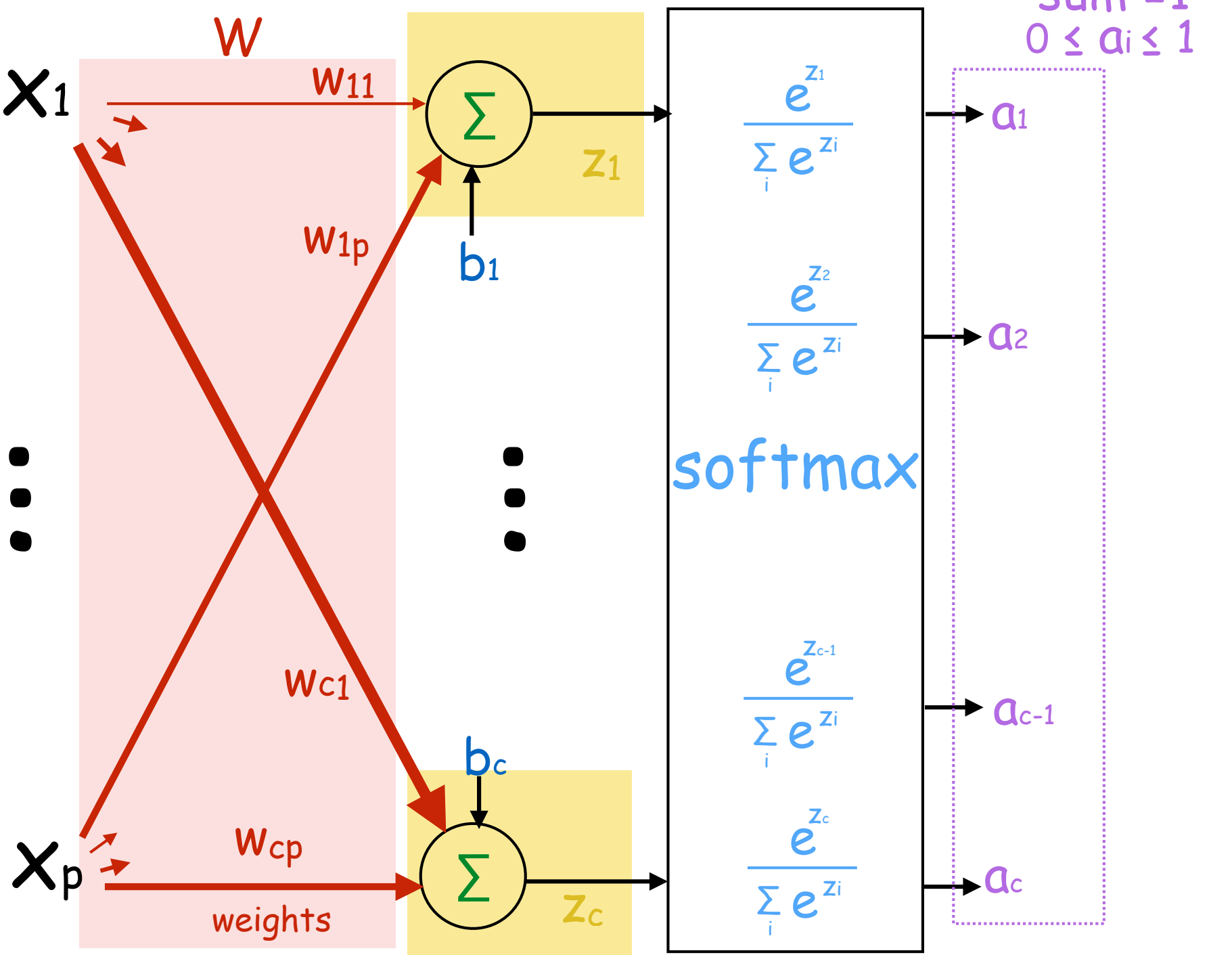
The outputs we need



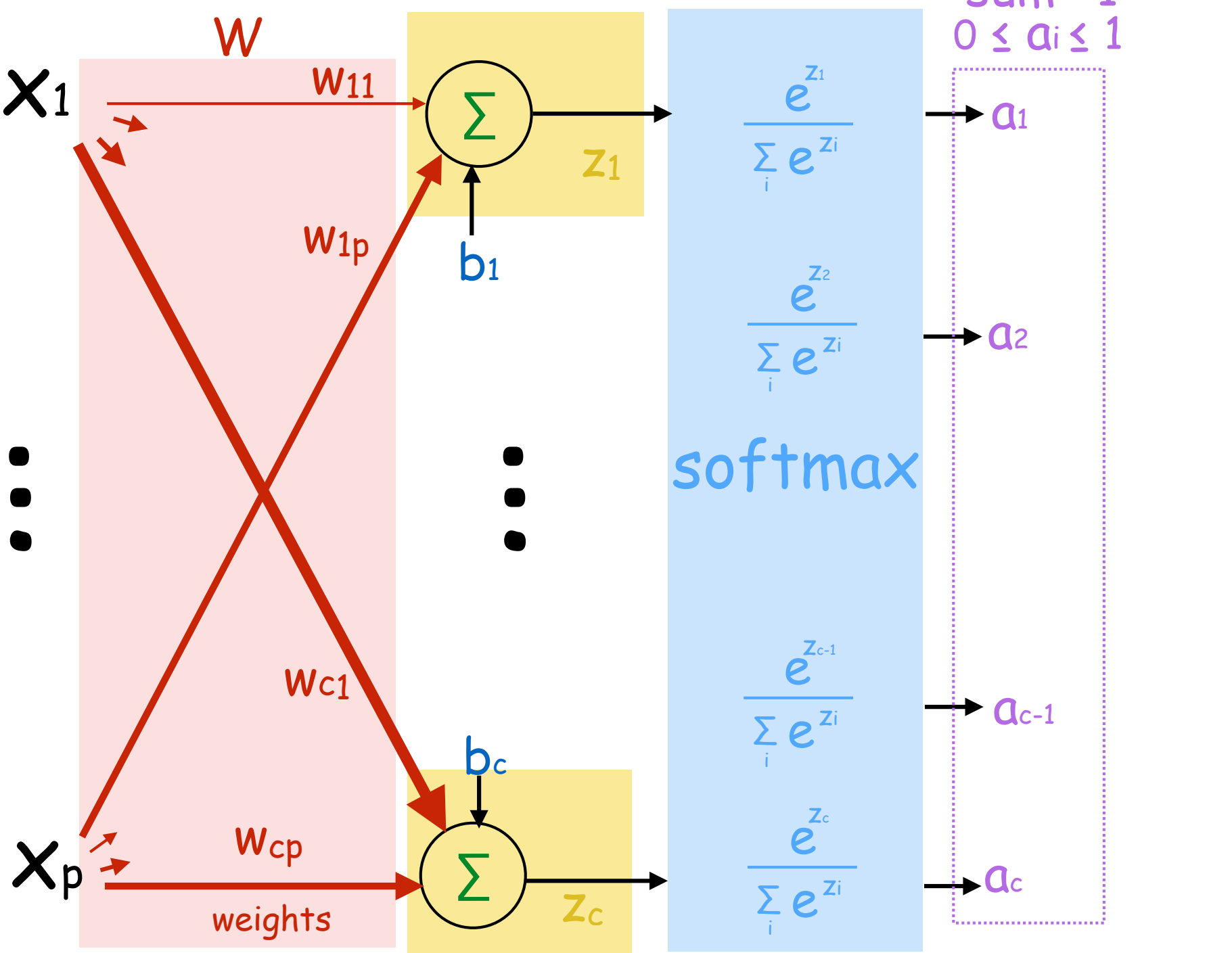
Coordinated Outputs



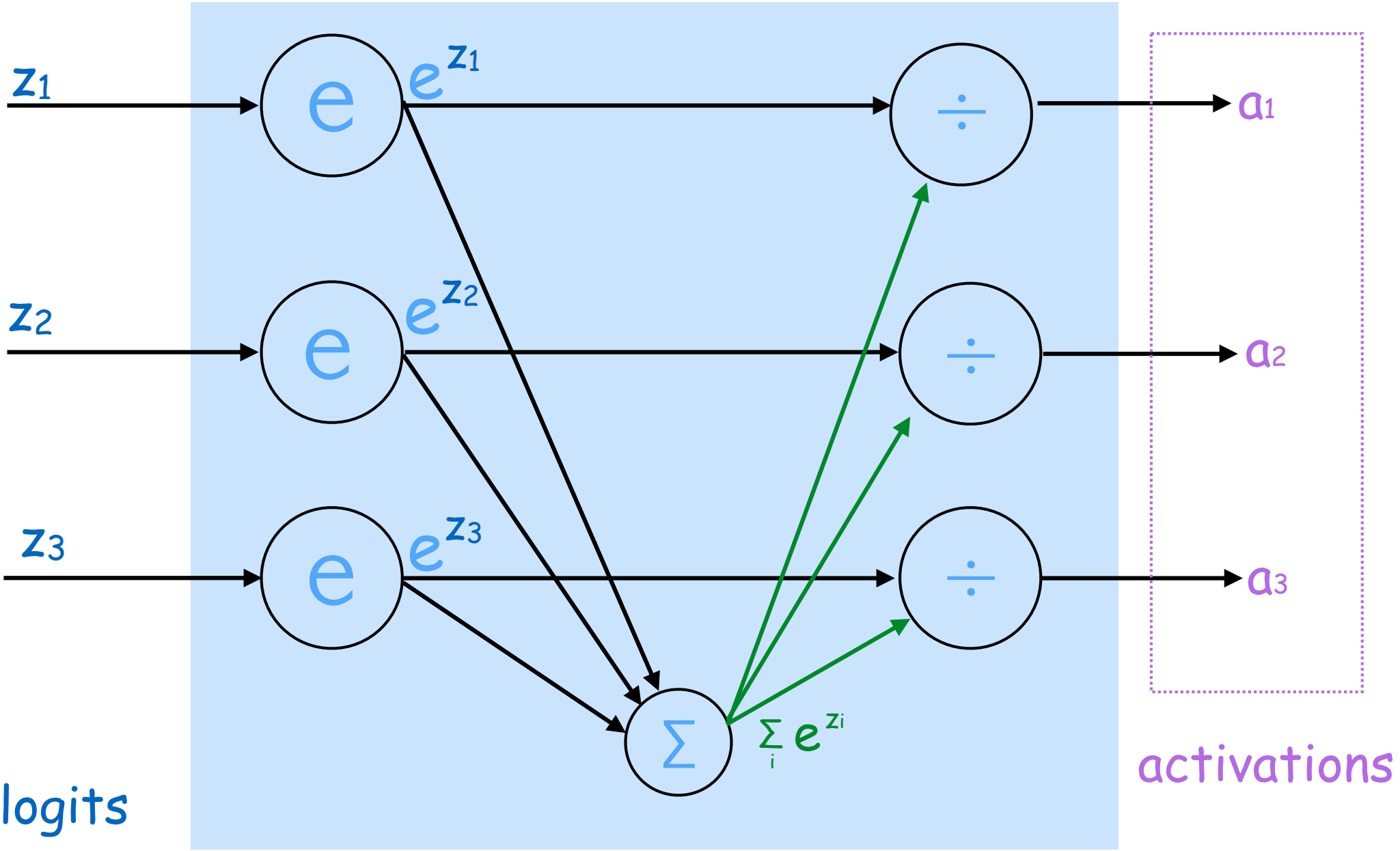
Softmax Activation



Softmax Activation

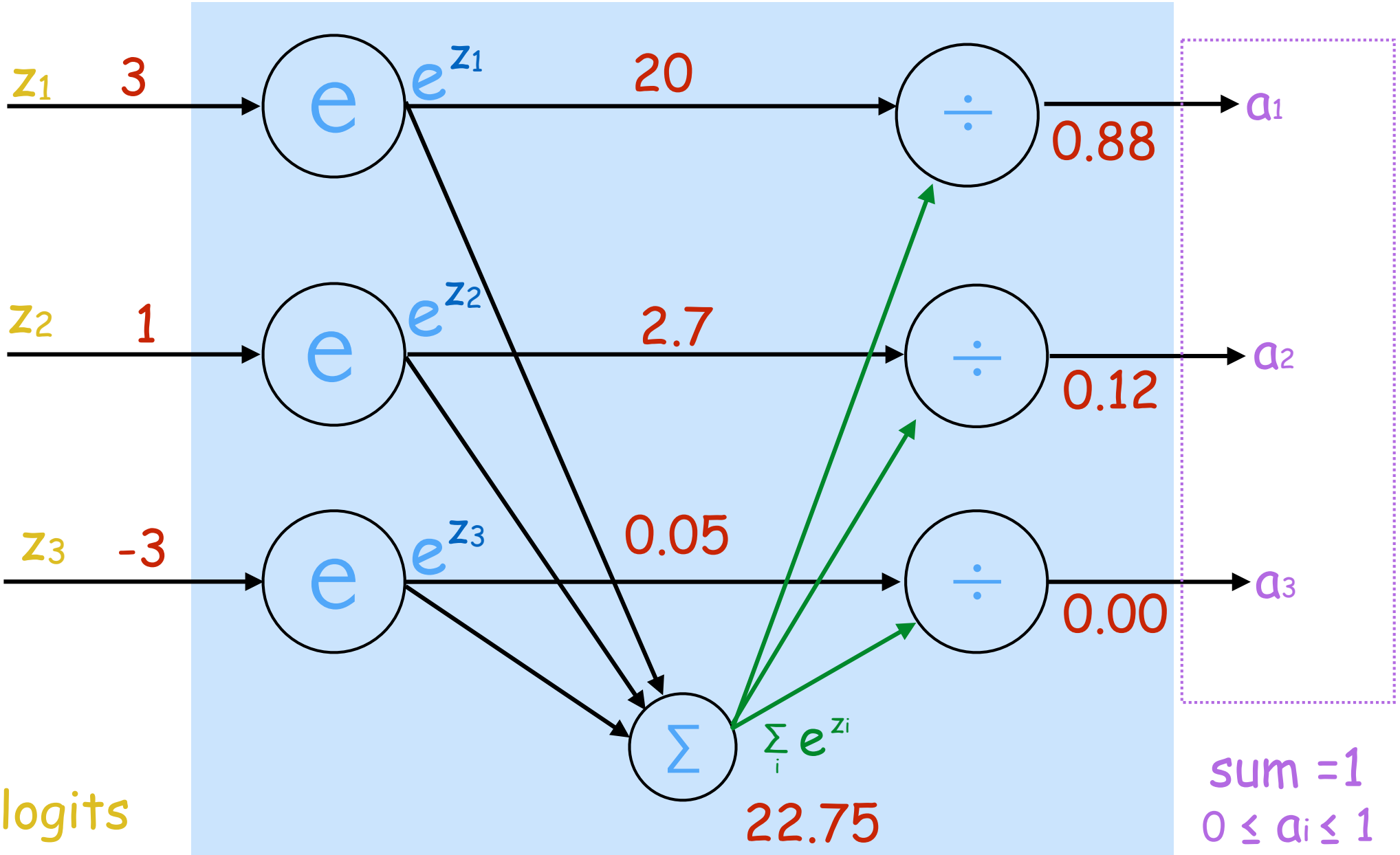


Softmax Activation



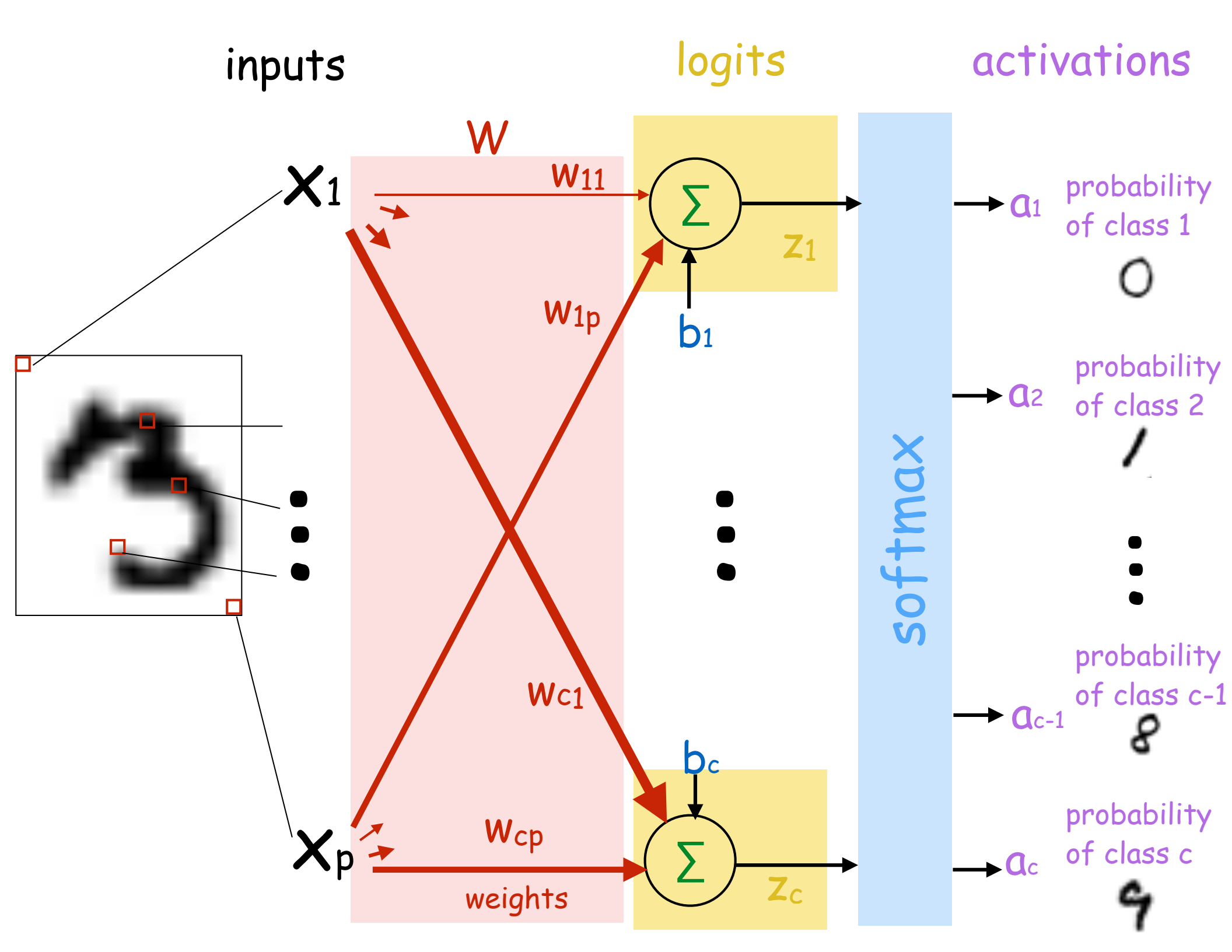
Softmax

Softmax Activation

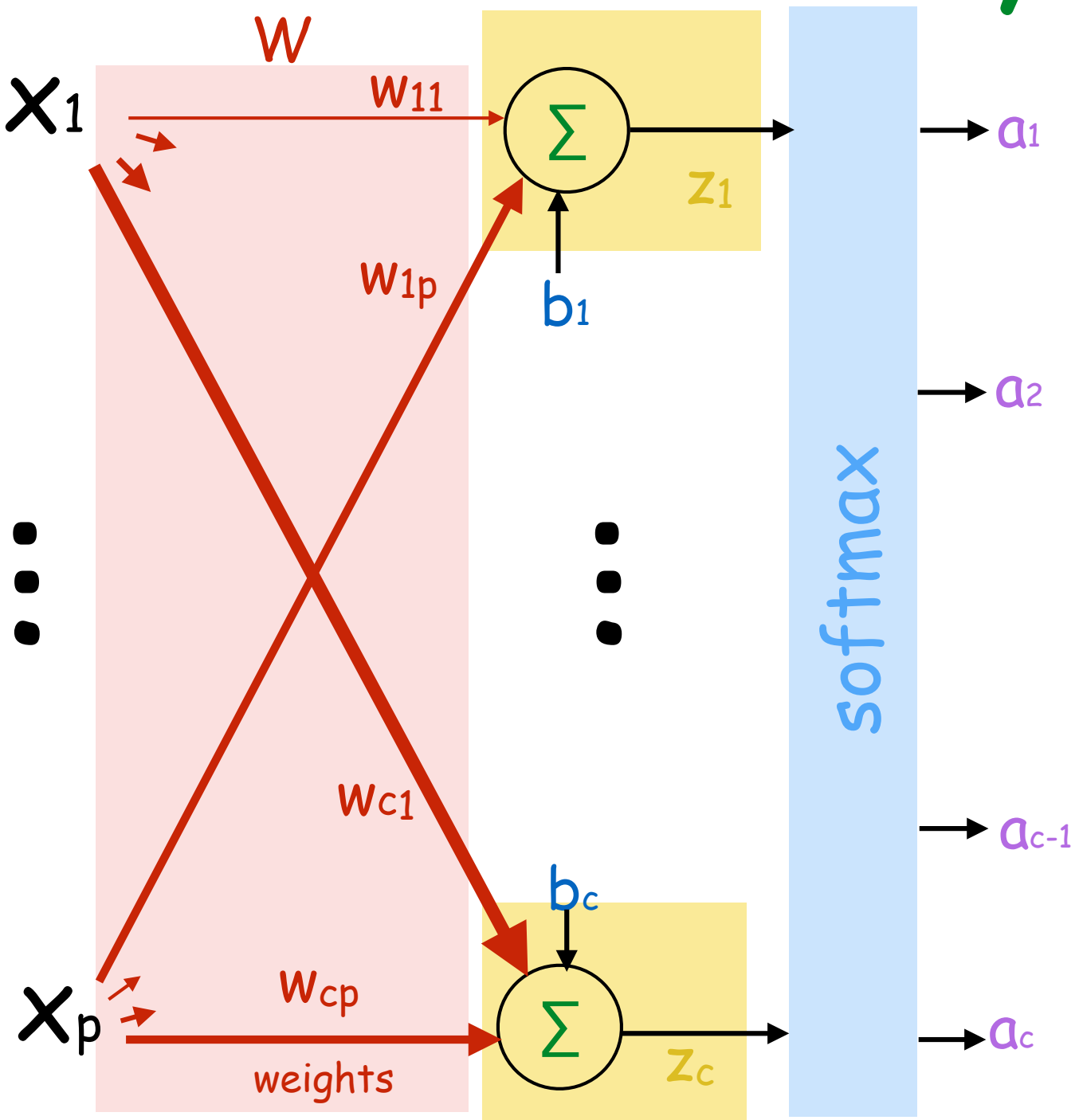


Softmax

Probabilities

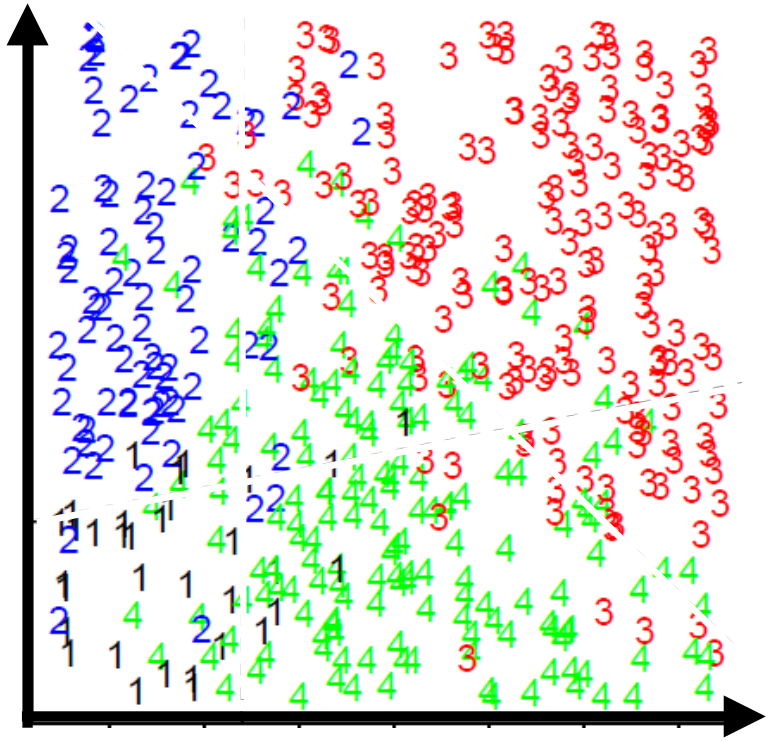

inputs
logits
activations
W
x_1
x_p
w_{11}
w_{1p}
w_{c1}
w_{cp}
weights
b_1
b_c
z_1
z_c
softmax
a_1
a_2
a_{c-1}
a_c
probability of class 1
probability of class 2
probability of class c-1
probability of class c

Parameters W, b



Training Model

training set

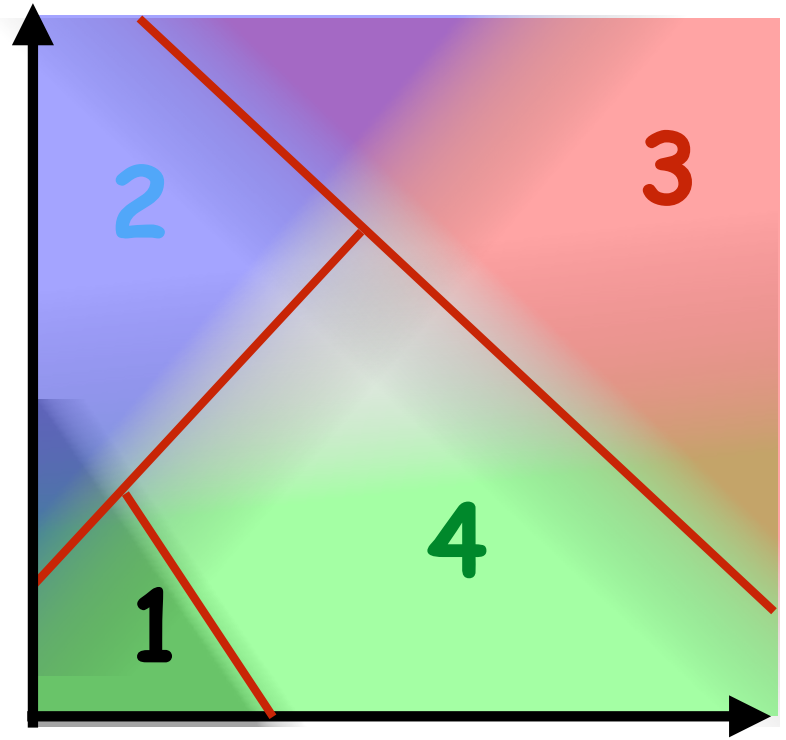


Feature Matrix $X_{(n \text{ by } p)}$

Label Vector Y

(length n) (0, 1, 2,... value)

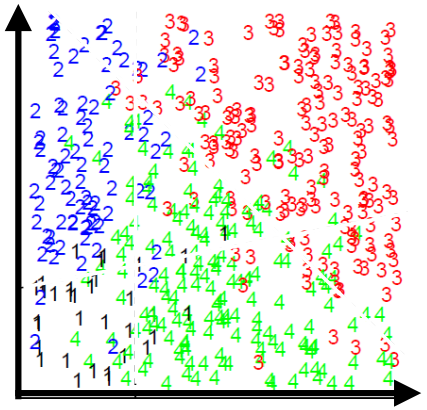
learn parameters



Weight Matrix **W** (shape c by p)

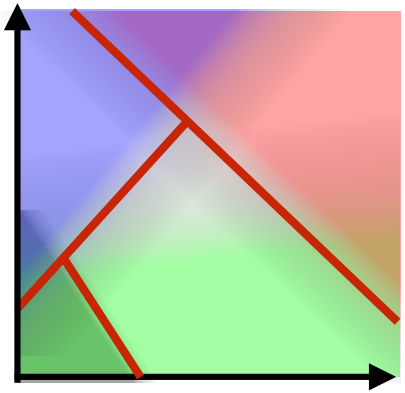
bias vector b (length c)

negative log Likelihood



Label: y 3 y 1

observed data



Weights **w**

biases b

Outputs: a_1 .4 a_1 .4

probabilities a_2 .2 a_2 .2

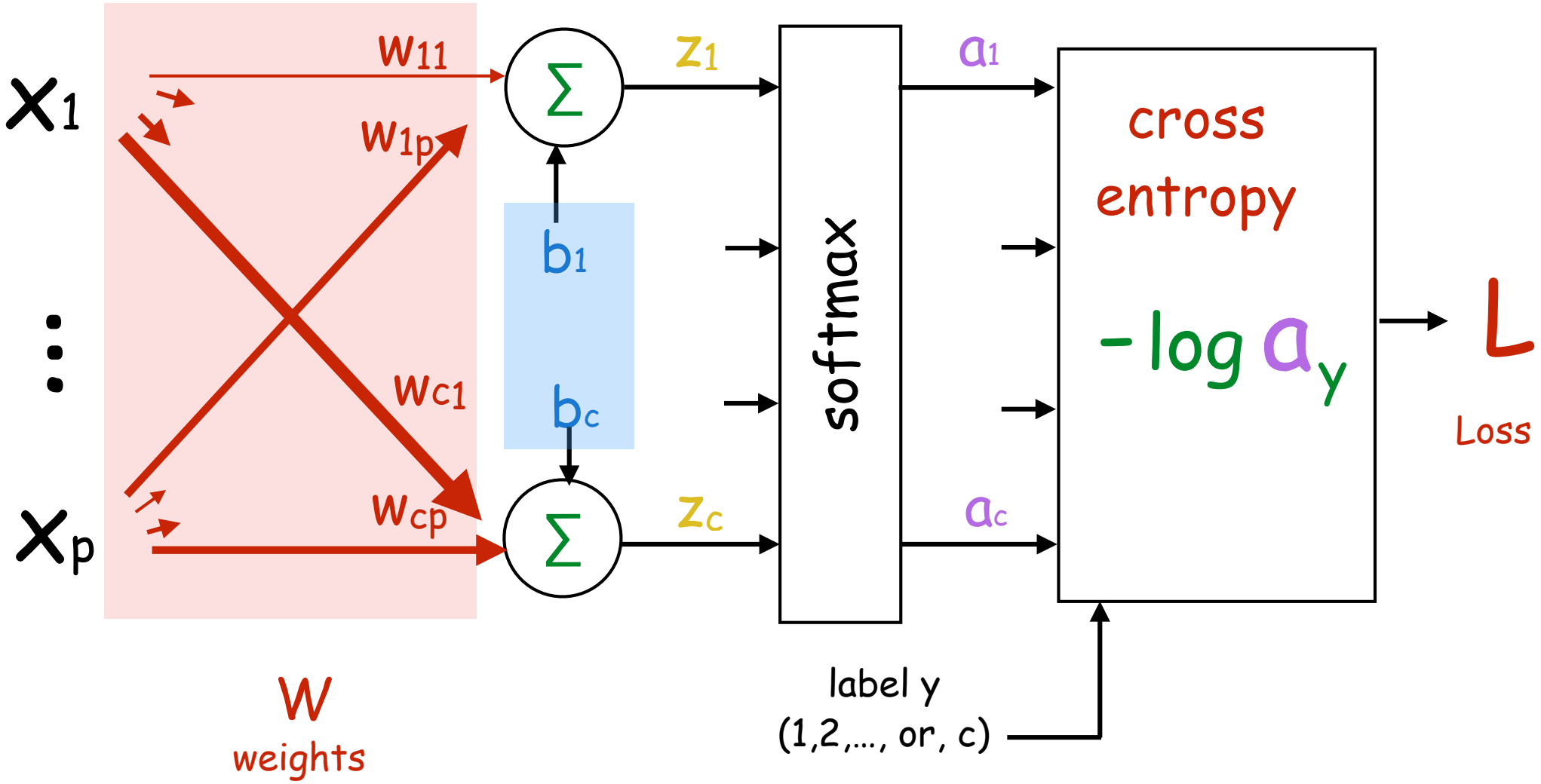
a_3 .1 a_3 .1

a_c .1 a_c .1

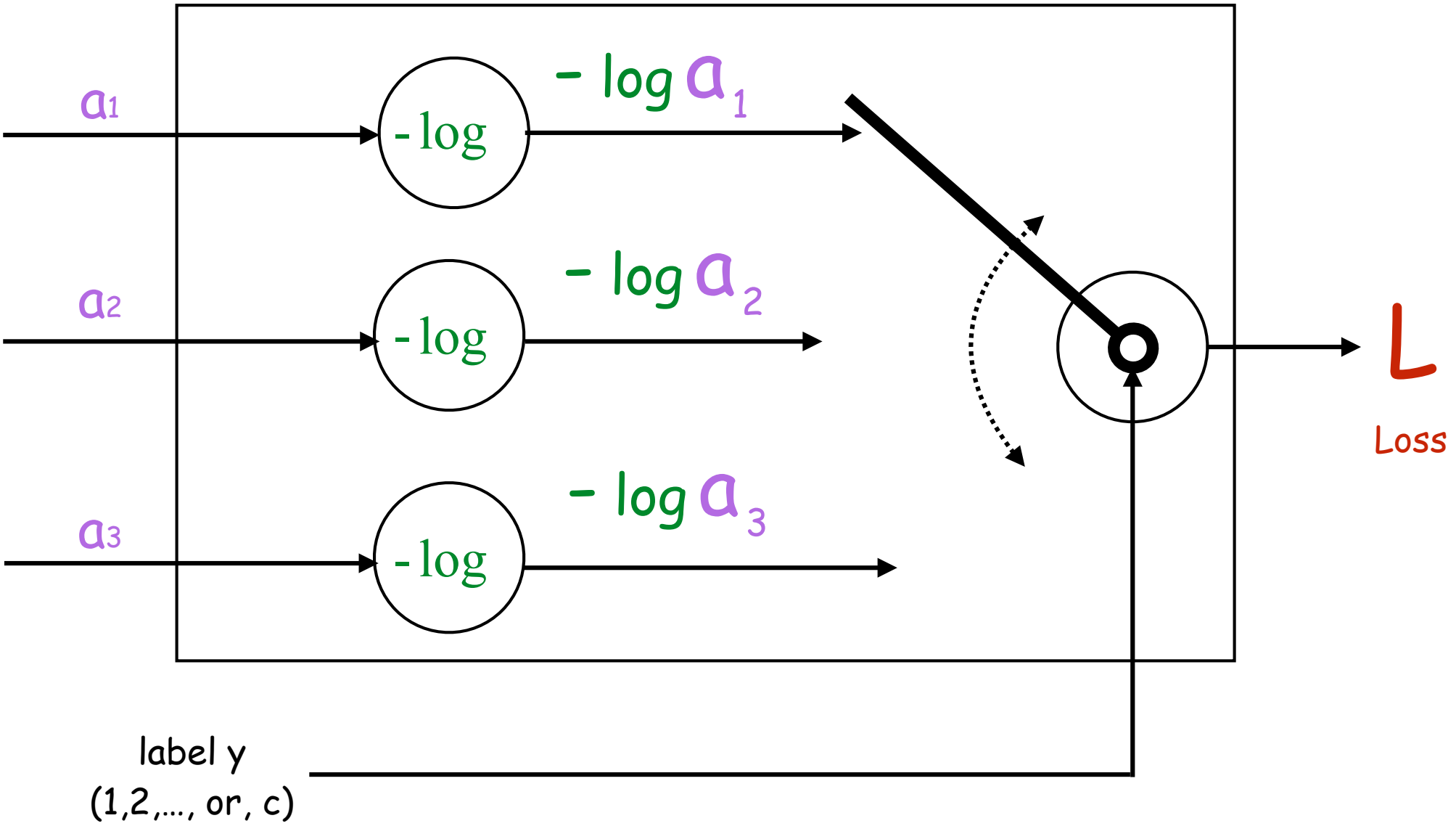
- log Likelihood = $-\log a_y$

Multi-class cross entropy loss

Multi-Class Cross Entropy Loss

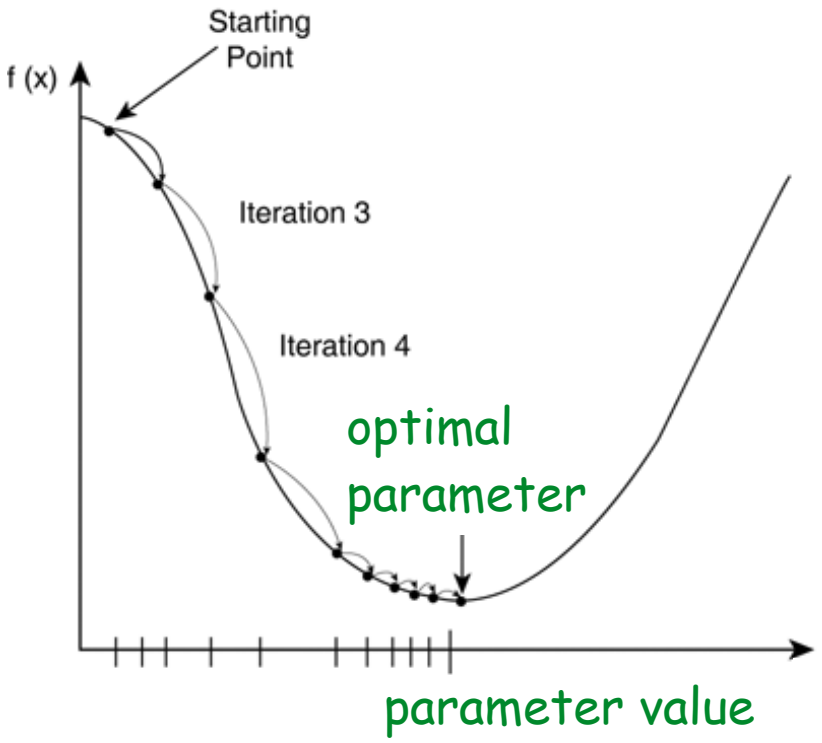


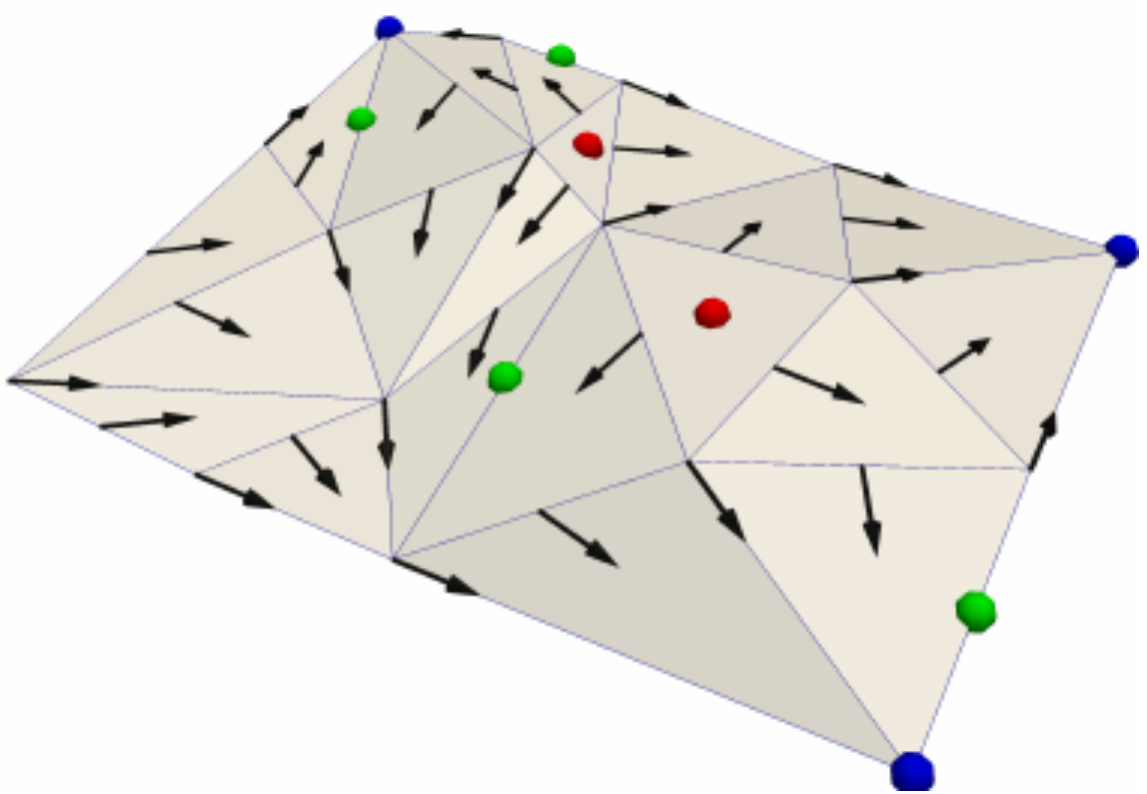
Multi-Class Cross Entropy Loss



Gradient Descent

L= loss (w)





$$W \leftarrow W - a \frac{\partial L}{\partial W}$$

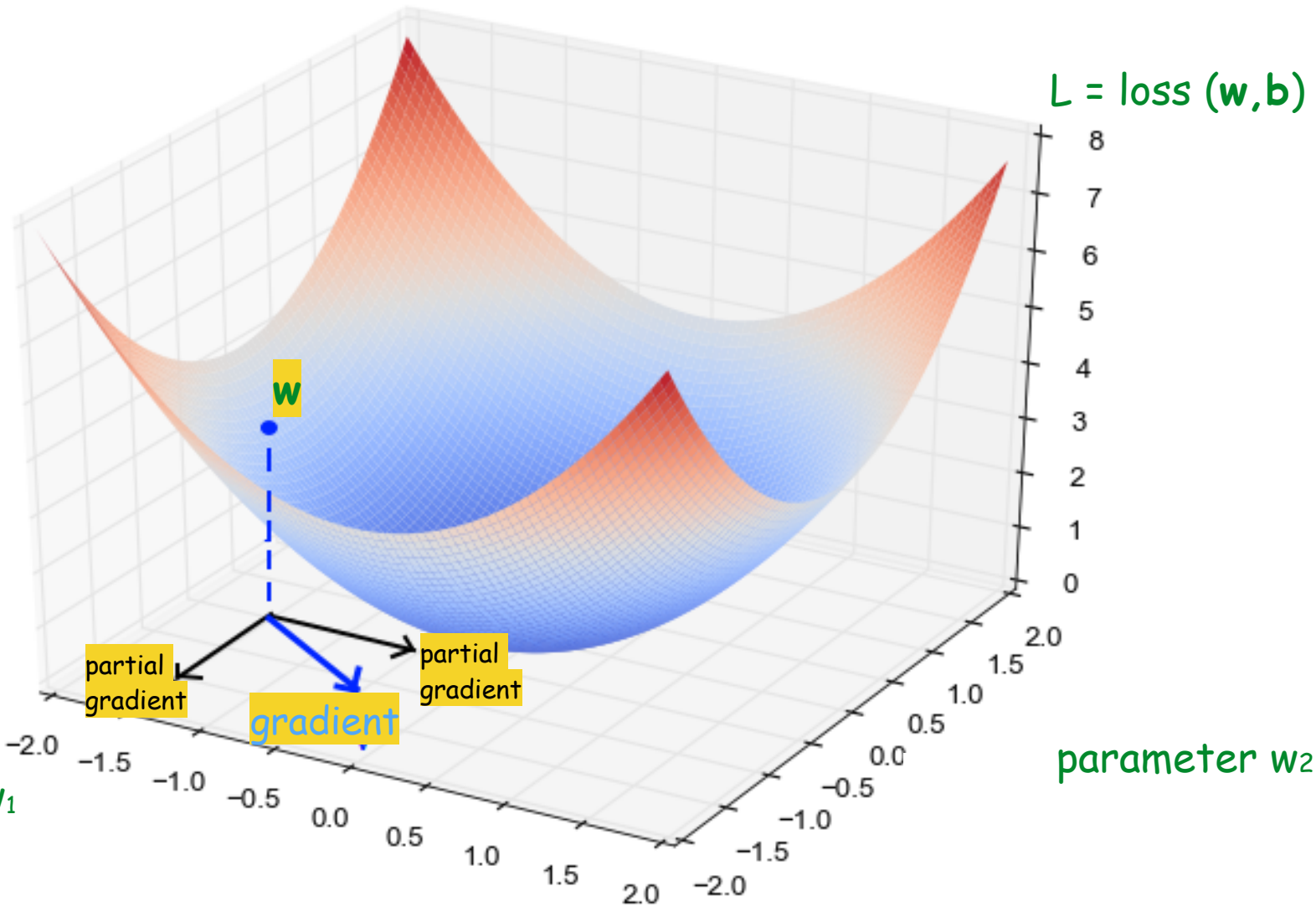
$$b \leftarrow b - a \frac{\partial L}{\partial b}$$

a step size (a constant scalar)

How to Compute Gradient

$$\frac{\partial L}{\partial w}$$

$$\frac{\partial L}{\partial b}$$



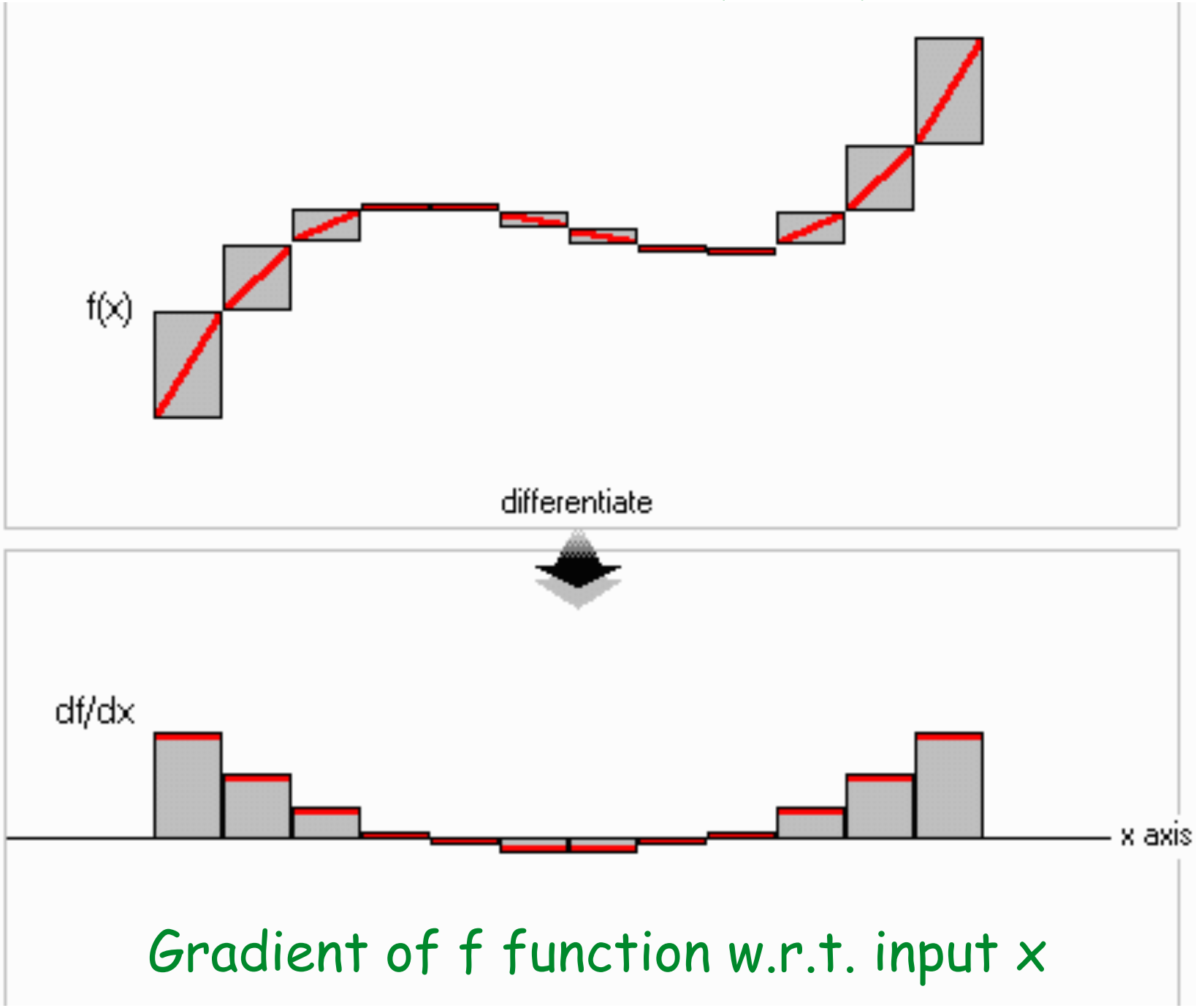
Gradient of L w.r.t. a **vector**?

$$\frac{\partial L}{\partial b}$$

Gradient of L w.r.t. a **matrix**?

$$\frac{\partial L}{\partial W}$$

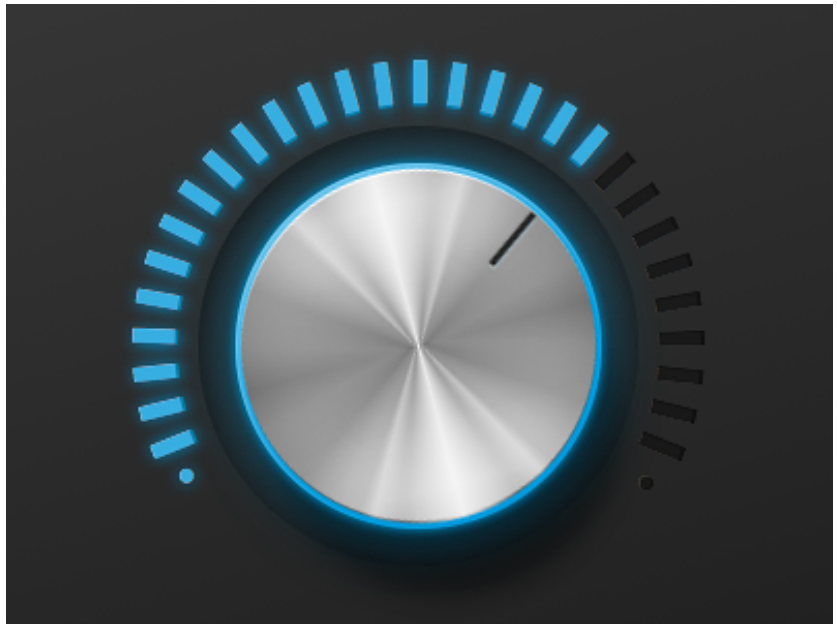
Gradient of a function f with respect to (w.r.t.) a scalar x

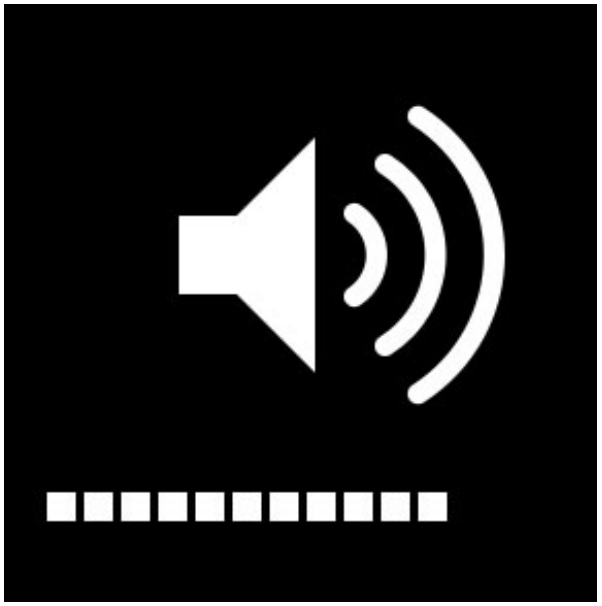


$$\frac{\partial f}{\partial x}$$

Gradient of f function w.r.t. input x

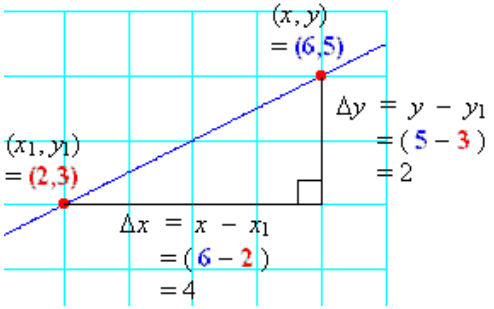
Gradient of L w.r.t. a scalar x





x

control

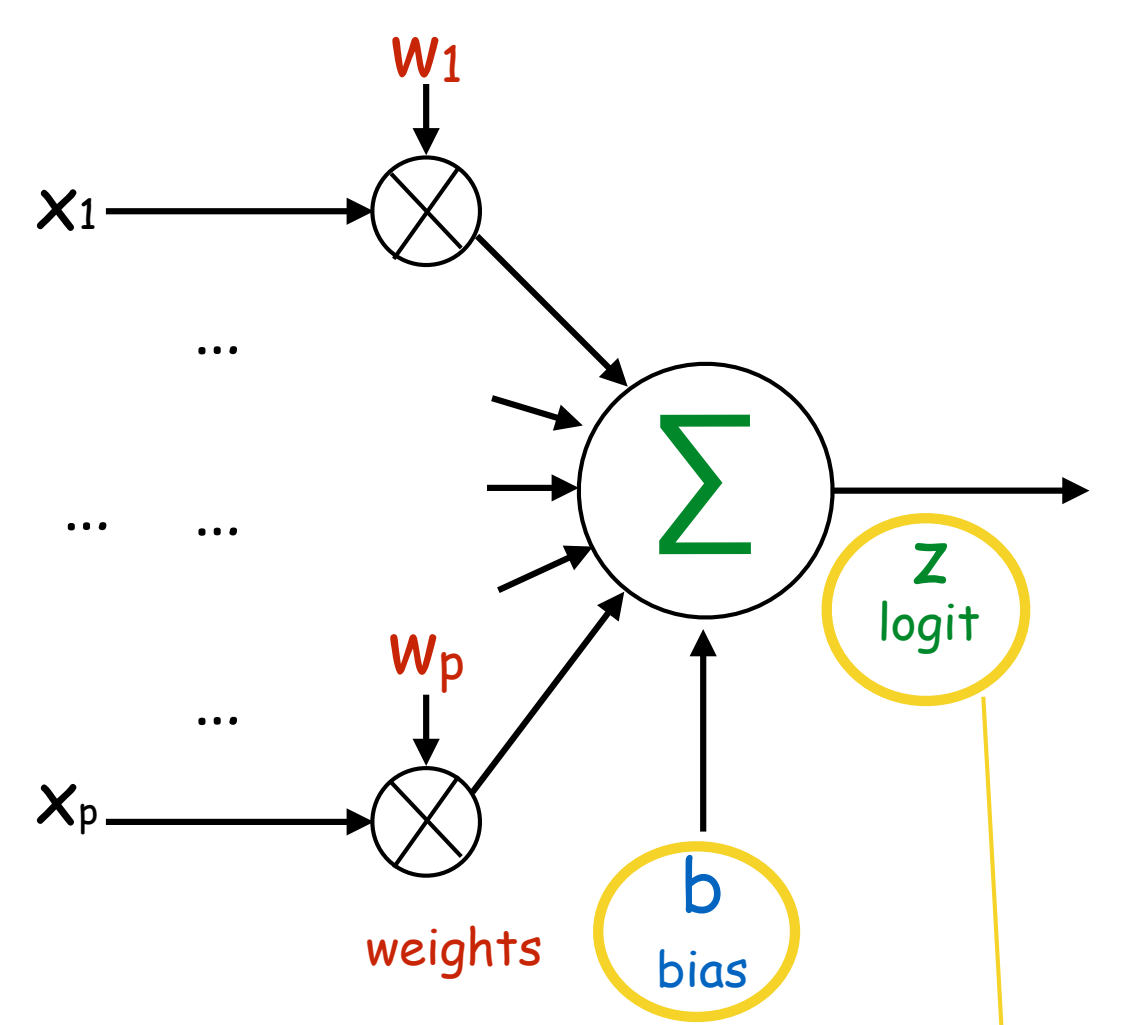


L

response

$$\frac{\partial L}{\partial x}$$ a constance number

Example



gradient

$$\frac{\partial z}{\partial b} = 1$$

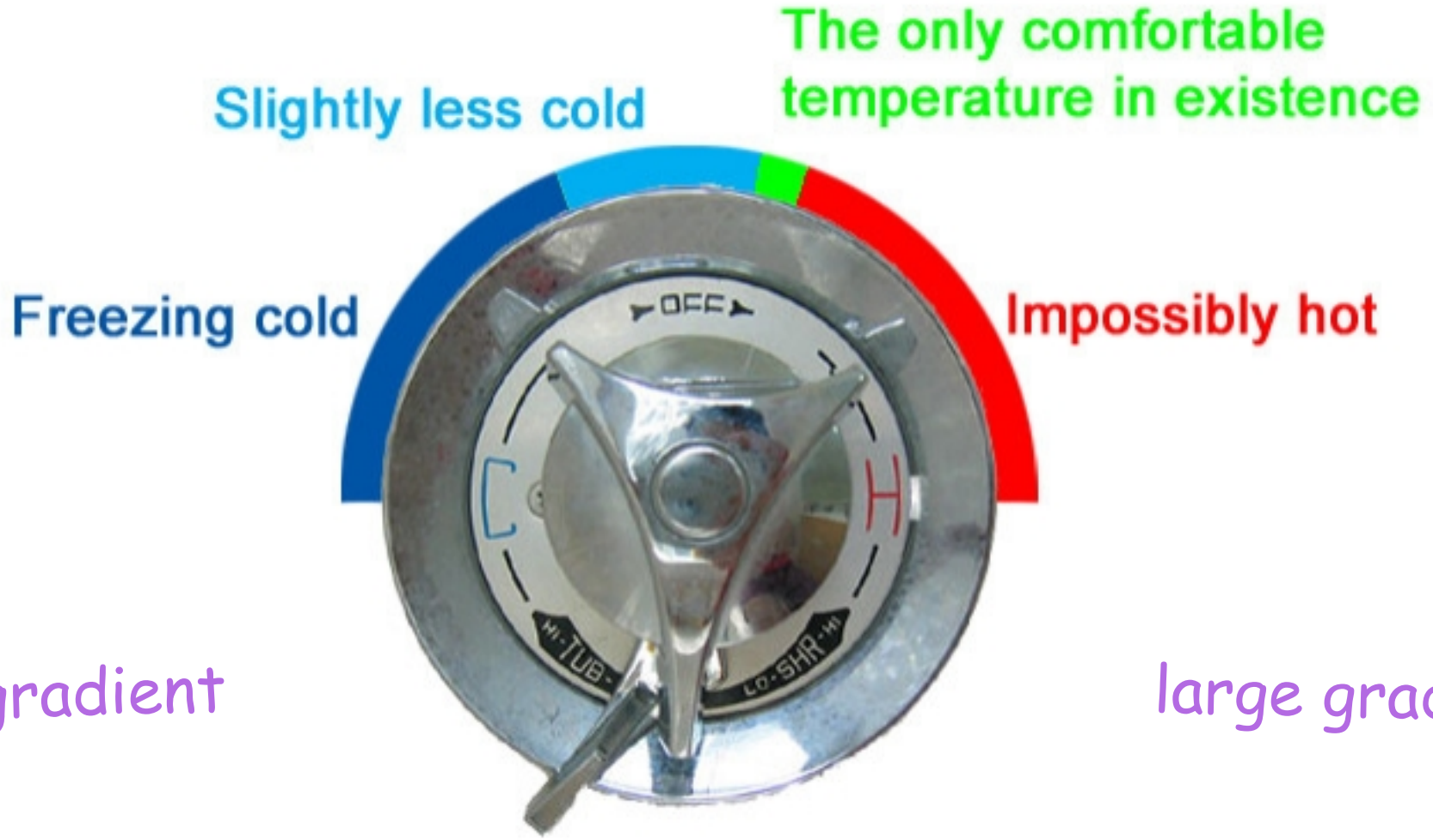
Another Example

X

L

water temperature

L is a function of x

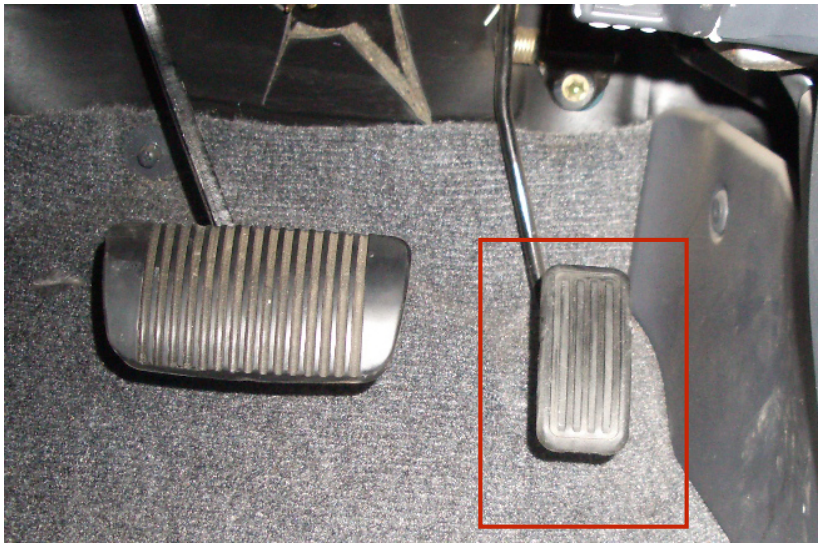


small gradient

large gradient

Gradient $\frac{\partial L}{\partial x}$ is also a function of x

Partial Gradient w.r.t. a scalar $\frac{\partial L}{\partial x}$



x

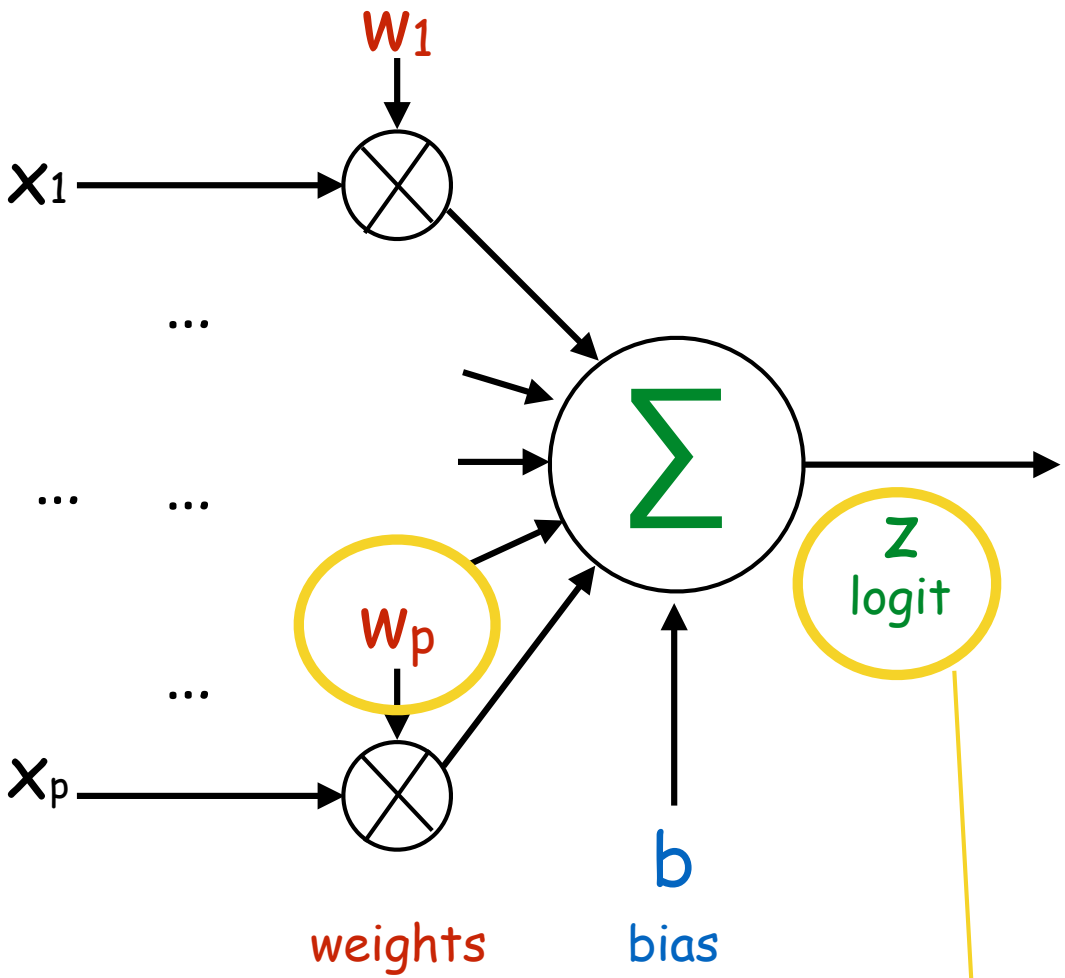
gas

L

speed

Gradient $\frac{\partial L}{\partial x}$ a function of x, speed, wind, gear …

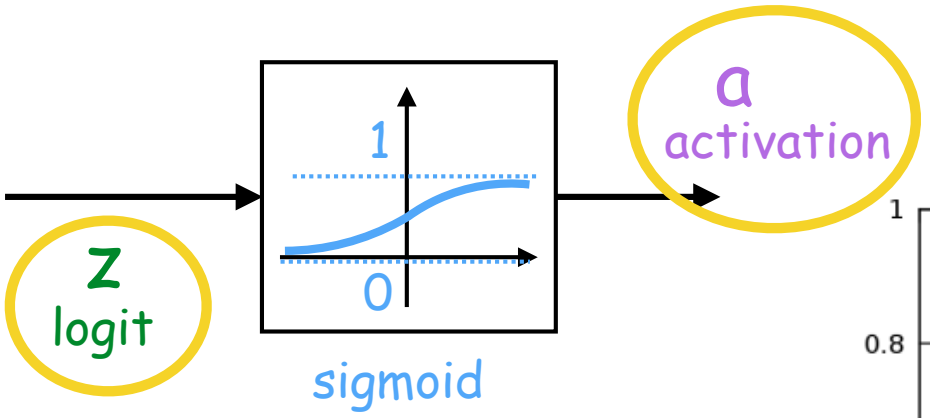
Example



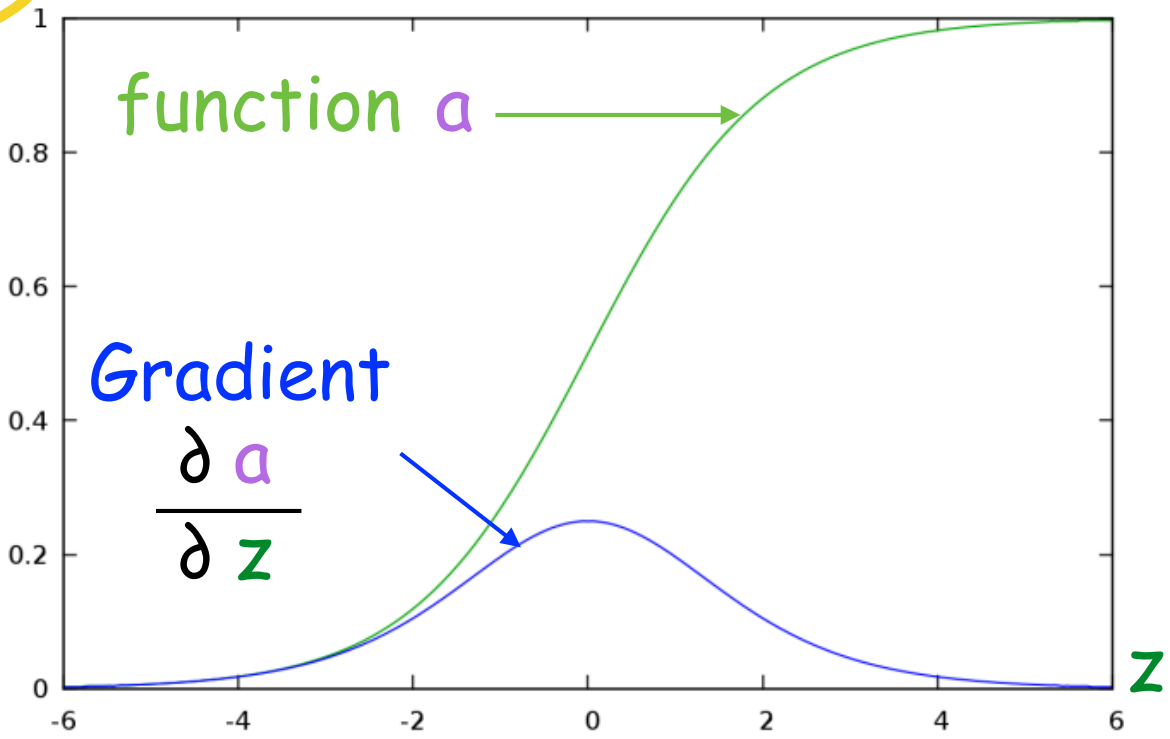
gradient

$$\frac{\partial z}{\partial w_p} = x_p$$

is a function of x_p



$$a = \frac{1}{1 + e^{-z}}$$

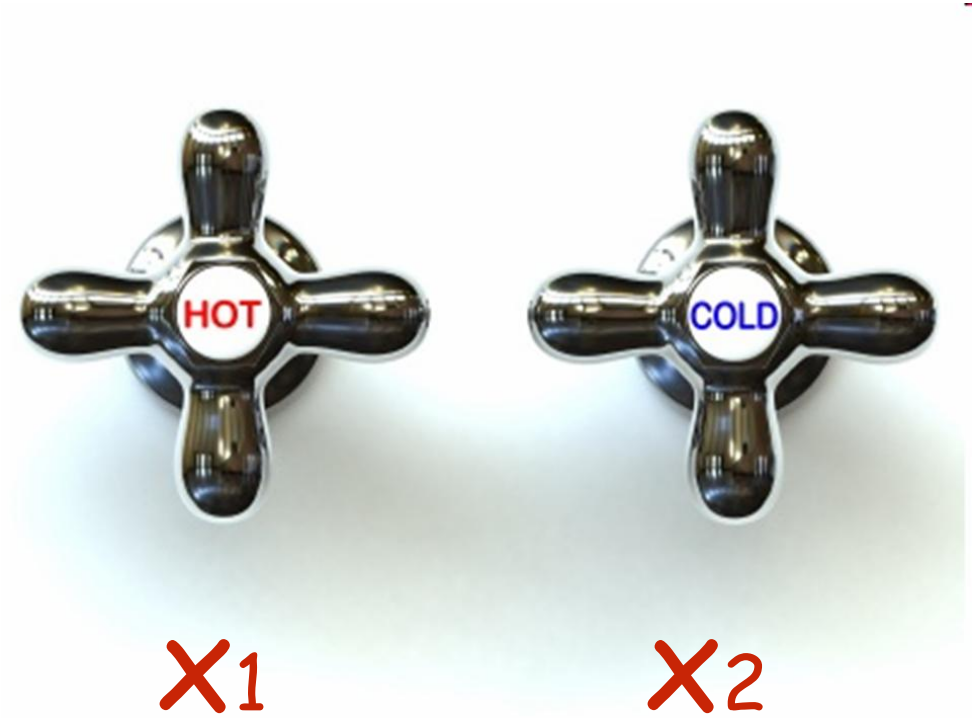


$$\frac{\partial a}{\partial z} = a\,(1 - a)$$

$$= \frac{1}{1 + e^{-z}} \left(1 - \frac{1}{1 + e^{-z}}\right)$$

is a function of z

Partial Gradients w.r.t. a vector $\frac{\partial L}{\partial x}$



x_1 x_2

L

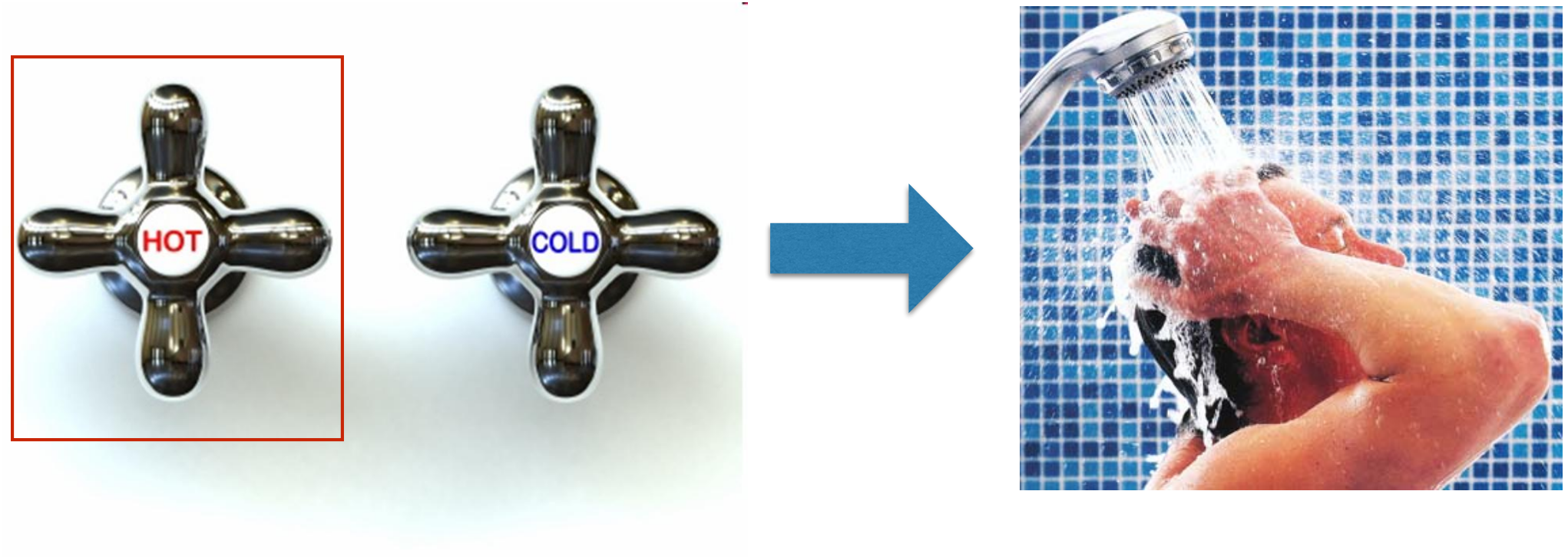
temperature

$$\frac{\partial L}{\partial x} = \left(\frac{\partial L}{\partial x_1} , \frac{\partial L}{\partial x_2} \right)$$

$$L = f(x_1, x_2)$$

Partial Gradients w.r.t. a vector $\frac{\partial L}{\partial x}$

x_1 x_2 L

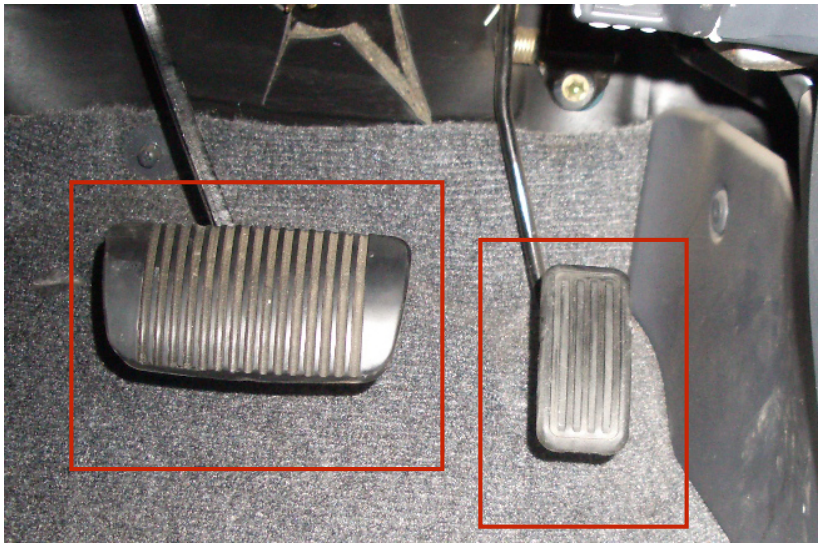


partial gradient — the gradient of f over x_1 while fixing x_2

$$\left(\frac{\partial L}{\partial x_1} , \frac{\partial L}{\partial x_2} \right)$$

$$L = f(x_1, x_2)$$

Partial Gradients w.r.t. a vector $\frac{\partial L}{\partial x}$



x_1
brake

x_2
gas

L
speed

$$\frac{\partial L}{\partial x} = \left(\frac{\partial L}{\partial x_1} , \frac{\partial L}{\partial x_2} \right)$$

Partial Gradients w.r.t. a vector $\frac{\partial L}{\partial x}$

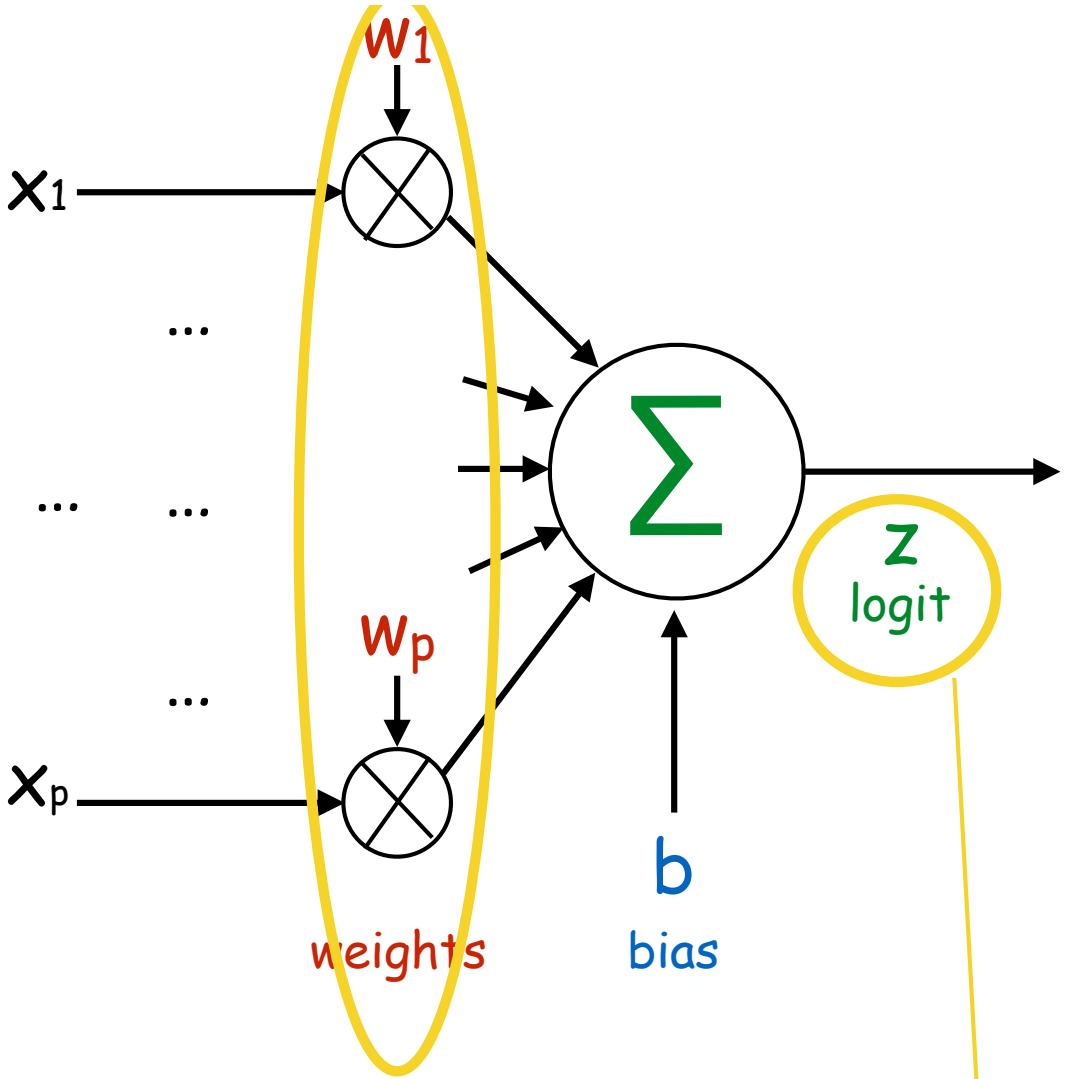
x_1 x_2 L

temperature

$$\left(\frac{\partial L}{\partial x_1} , \frac{\partial L}{\partial x_2} \right) = \frac{\partial L}{\partial x}$$

3 -4 for example

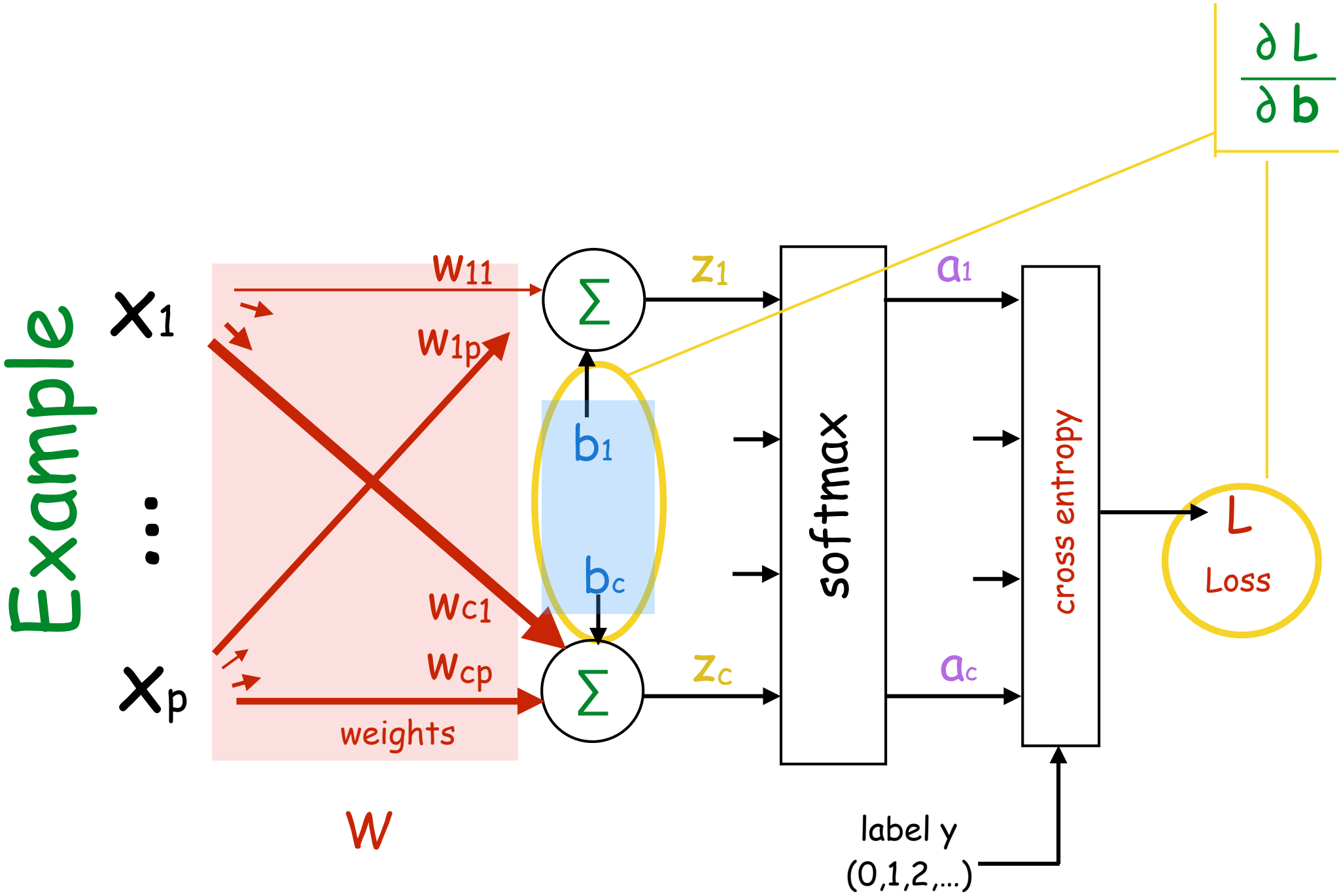
Example



gradient

$$\frac{\partial z}{\partial \mathbf{w}} = \left(\frac{\partial z}{\partial w_1}, \frac{\partial z}{\partial w_2}, \cdots, \frac{\partial z}{\partial w_p}\right)$$

$$= \left(x_1 , x_2 , \cdots, x_p \right) = \mathbf{x}$$


Example
x_1
x_p
w_{11}
w_{1p}
w_{c1}
w_{cp}
weights
W
b_1
b_c
z_1
z_c
softmax
a_1
a_c
cross entropy
L
Loss
label y
(0,1,2,...)
$\frac{\partial L}{\partial b}$

Gradient w.r.t. a Matrix?

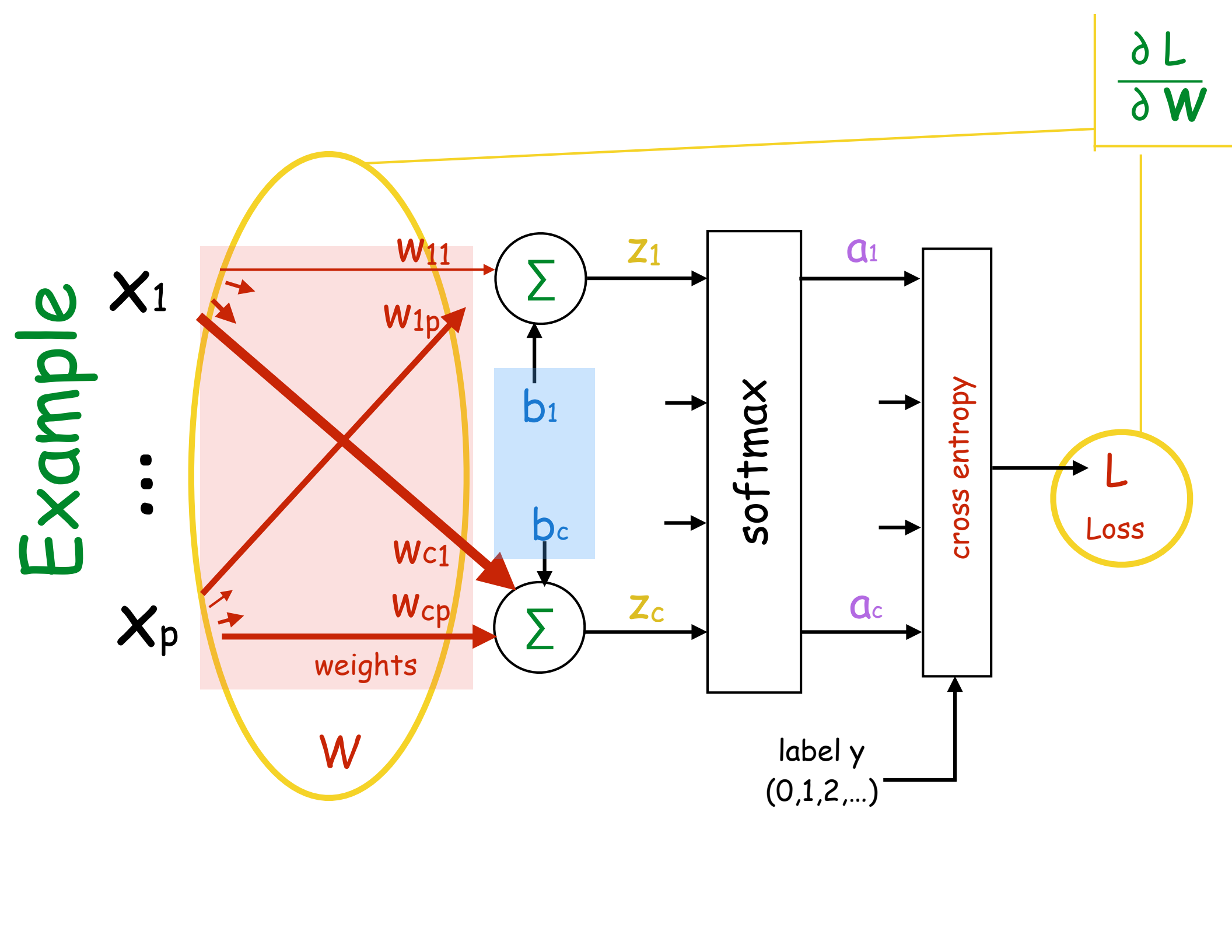
height

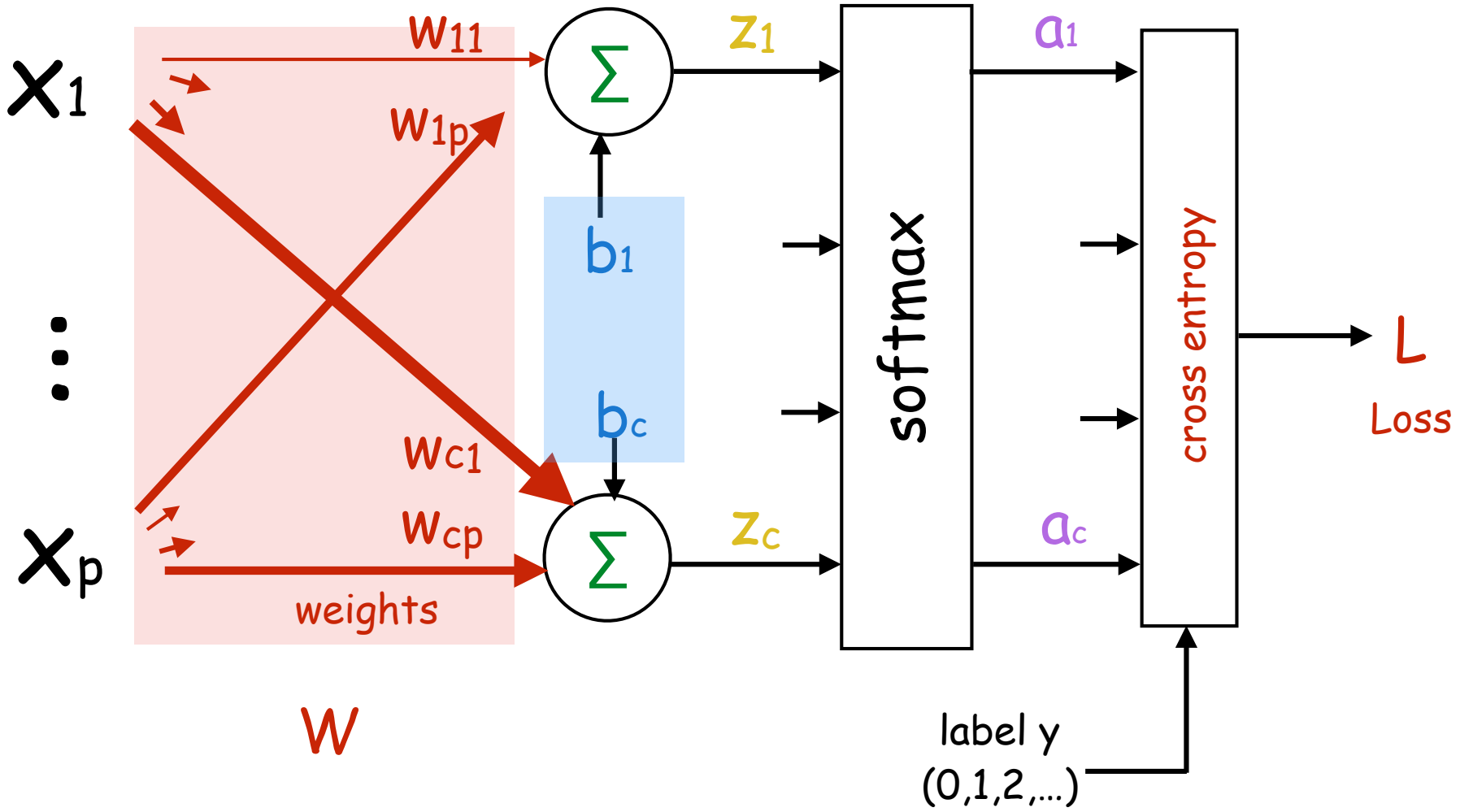
a matrix of controls

Gradient w.r.t. a Matrix?

W L

$$\frac{\partial L}{\partial W} = \begin{bmatrix} \frac{\partial L}{\partial w_{11}} & \cdots & \frac{\partial L}{\partial x_{1p}} \\ \vdots & & \vdots \\ \frac{\partial L}{\partial w_{c1}} & & \frac{\partial L}{\partial w_{cp}} \end{bmatrix}$$

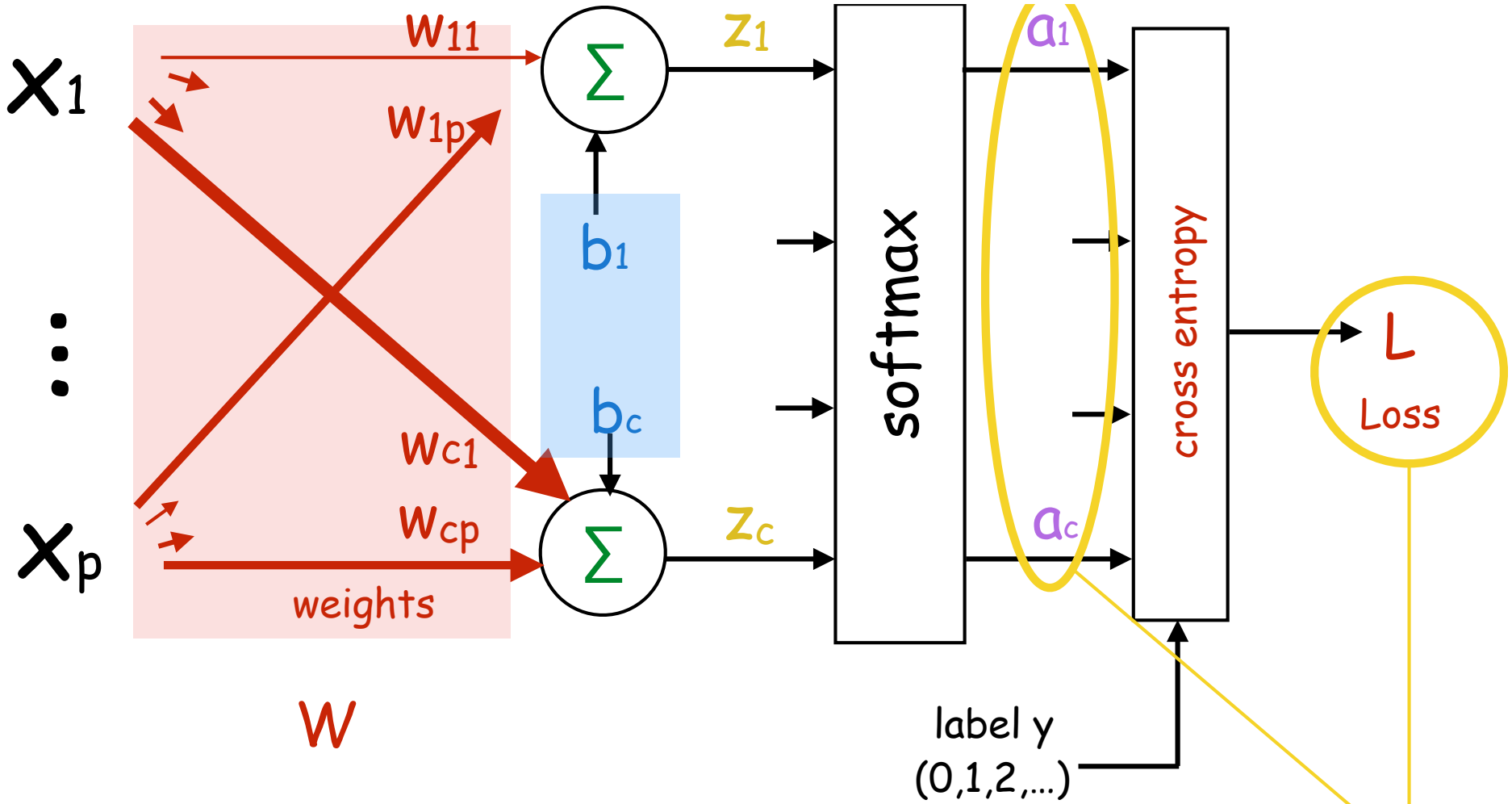

Example
$\frac{\partial L}{\partial W}$
x_1
x_p
w_{11}
w_{1p}
w_{c1}
w_{cp}
weights
W
b_1
b_c
z_1
z_c
softmax
a_1
a_c
cross entropy
L
Loss
label y
(0,1,2,...)



$$\frac{\partial L}{\partial W} = \frac{\partial L}{\partial a} \times \frac{\partial a}{\partial z} \times \frac{\partial z}{\partial W}$$

$$\frac{\partial L}{\partial b} = \frac{\partial L}{\partial a} \times \frac{\partial a}{\partial z} \times \frac{\partial z}{\partial b}$$

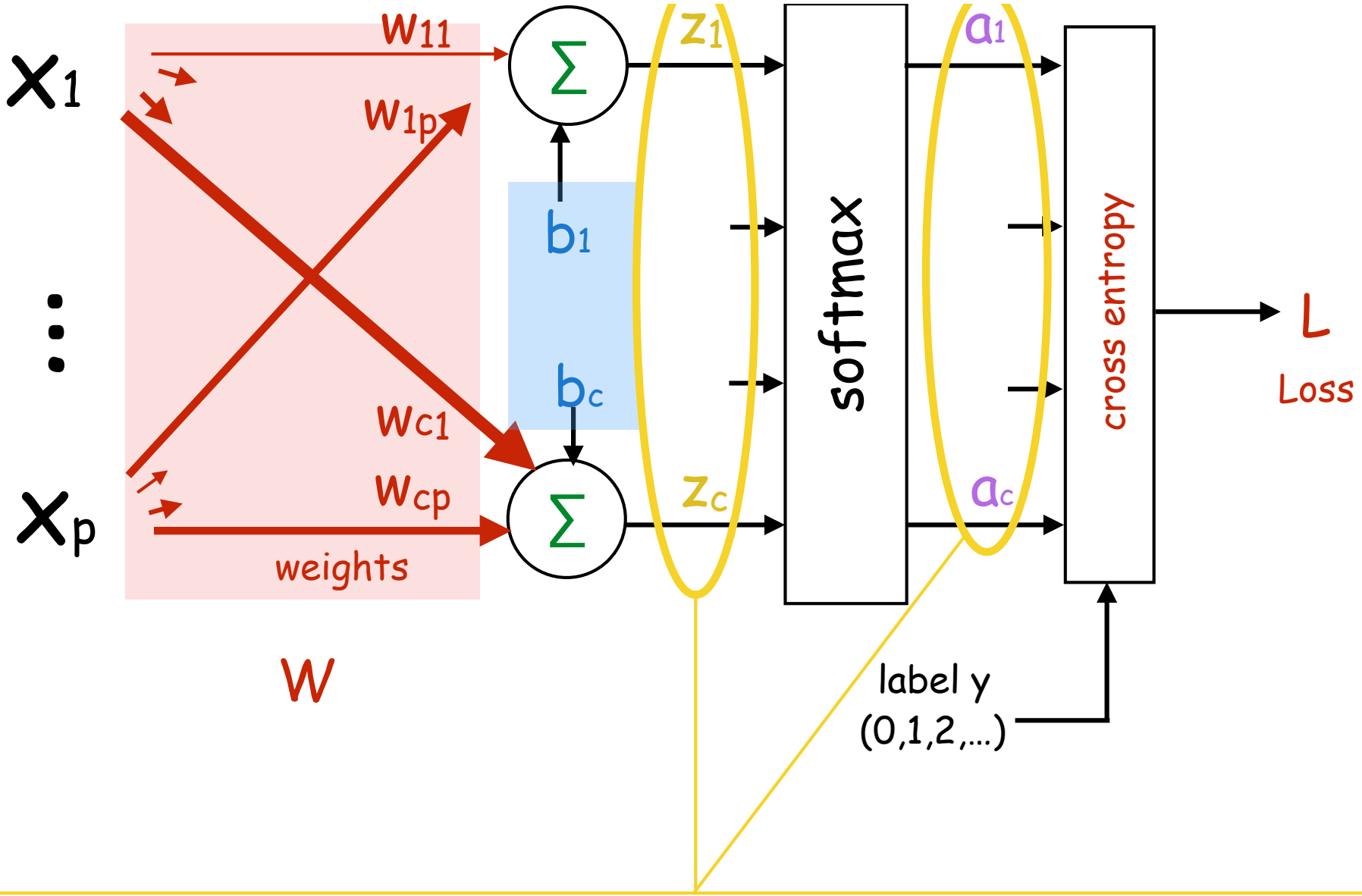
Chain Rule



$$\frac{\partial L}{\partial a} = \left(\frac{\partial L}{\partial a_1}, \ldots, \frac{\partial L}{\partial a_y}, \ldots, \frac{\partial L}{\partial a_c}\right)$$

$$= \left(0, \ldots, -\frac{1}{a_y}, \ldots, 0\right)$$

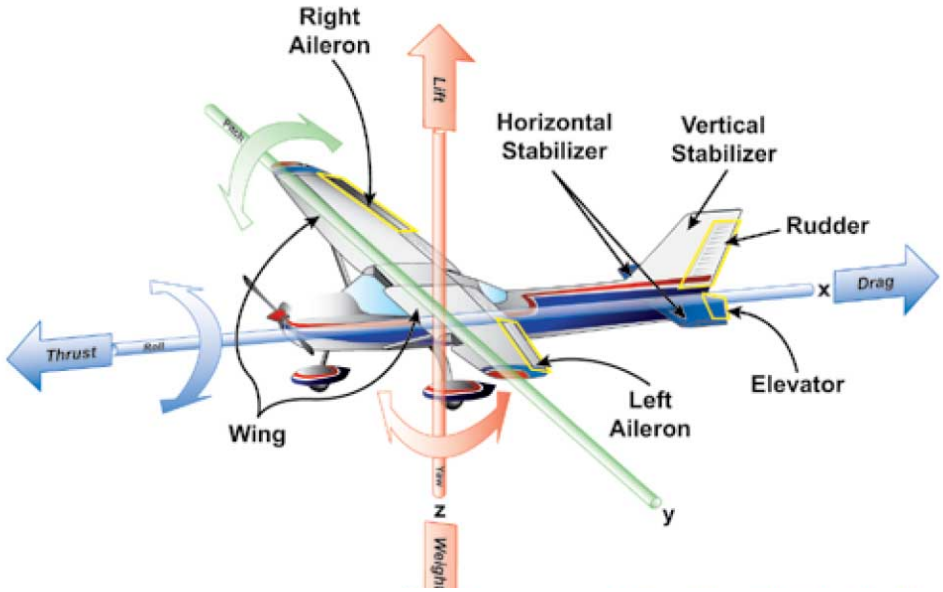
y



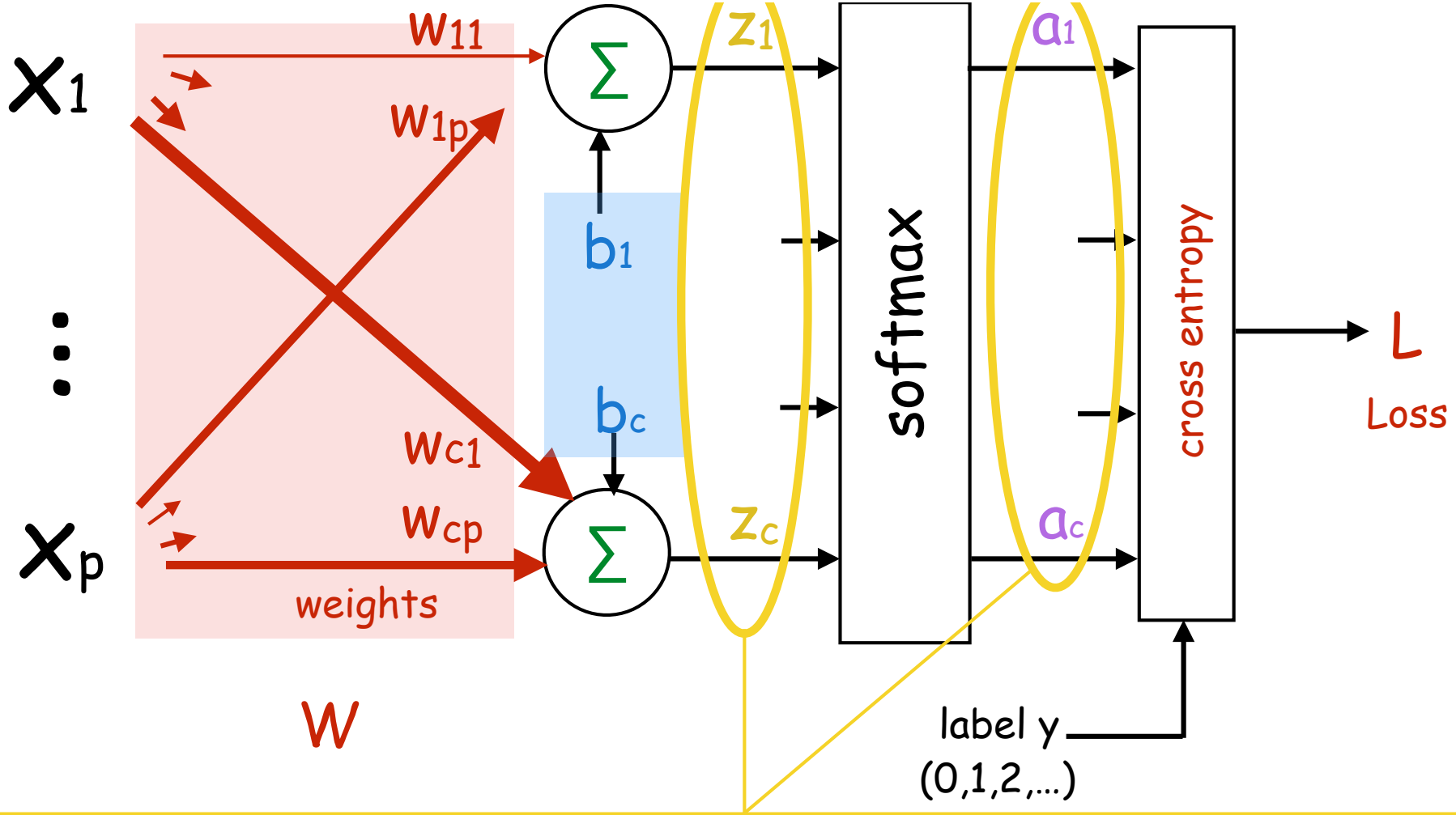
$$\frac{\partial \mathbf{a}}{\partial \mathbf{z}} = ?$$

Gradient of a vector w.r.t. a vector !!!

Gradient of a vector w.r.t. a vector

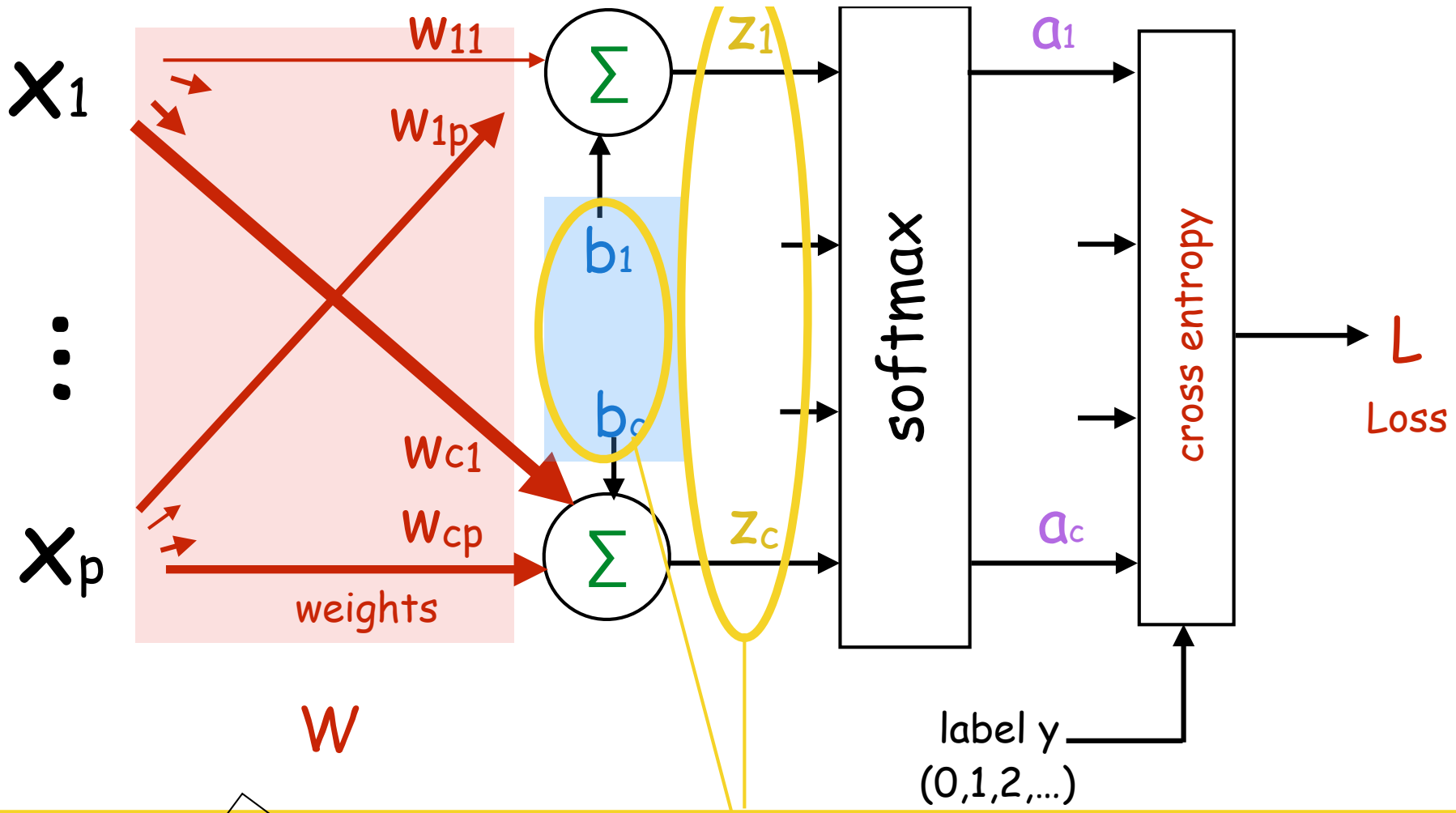


a vector of controls a vector of response

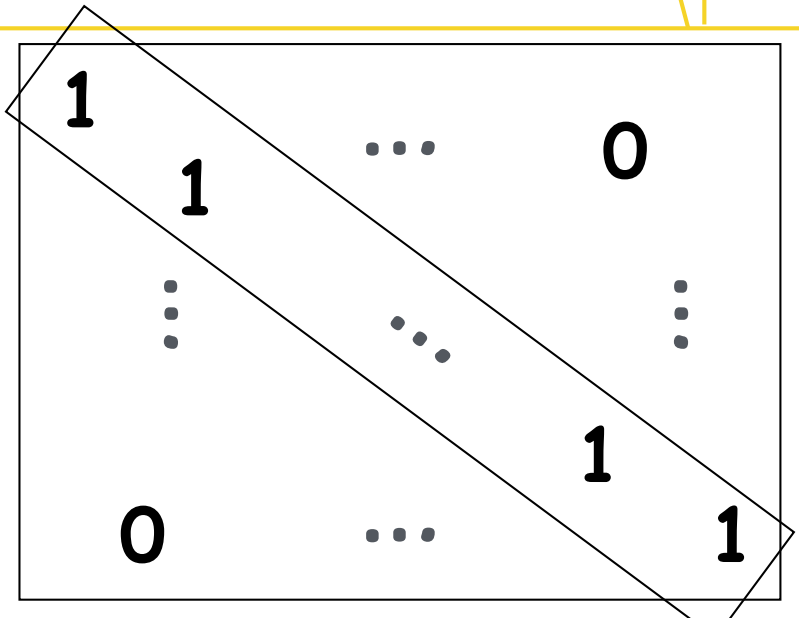


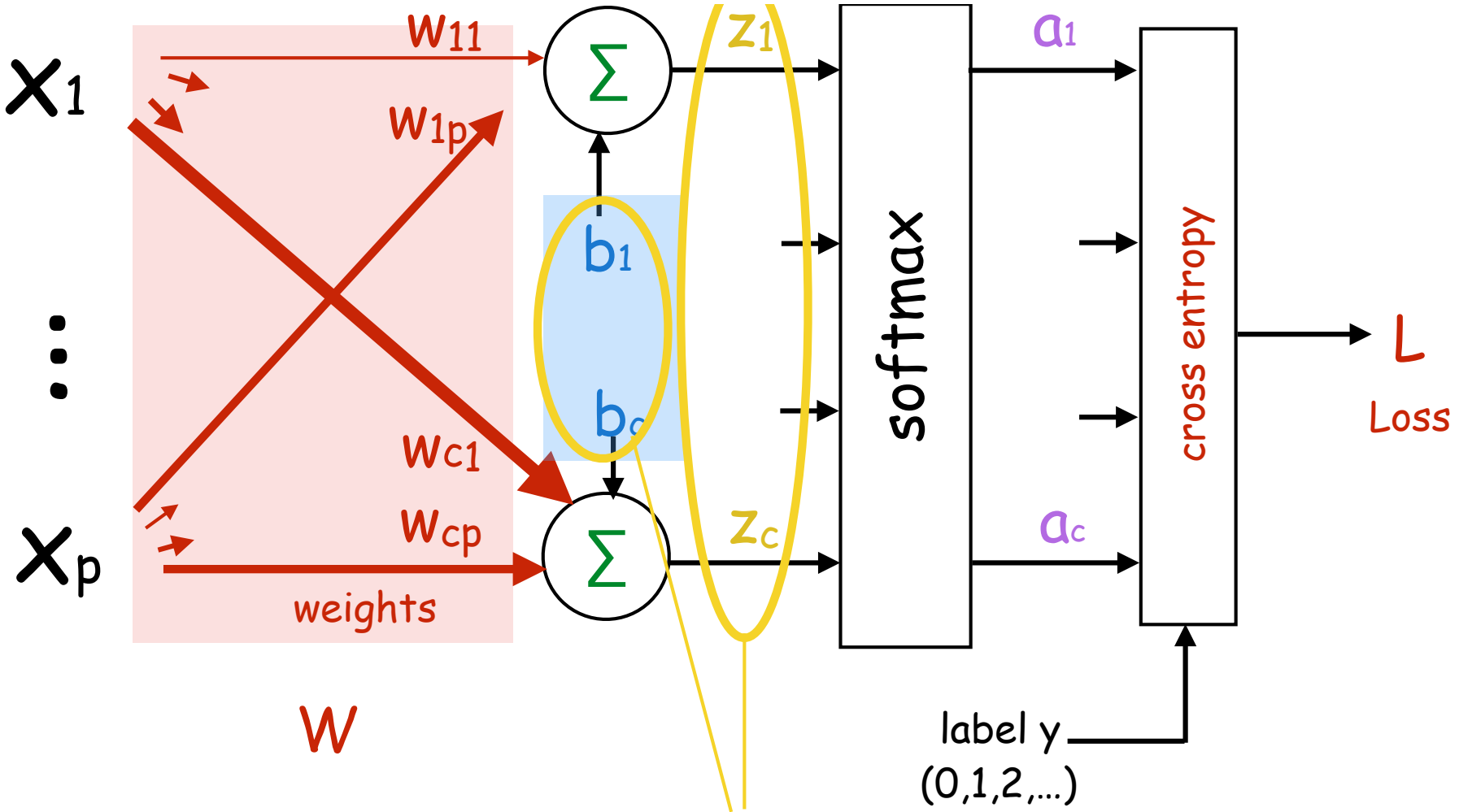
$$\frac{\partial a}{\partial z} = \begin{bmatrix} \frac{\partial a_1}{\partial z_1} & \cdots & \frac{\partial a_1}{\partial z_c} \\ \vdots & & \vdots \\ \frac{\partial a_c}{\partial z_1} & & \frac{\partial a_c}{\partial z_c} \end{bmatrix}$$

$$\frac{\partial a_i}{\partial z_j} = \begin{cases} a_i (1 - a_i) & \text{if } i=j \\ -a_i a_j & \text{if } i \neq j \end{cases}$$



$$\frac{\partial z}{\partial b} =$$



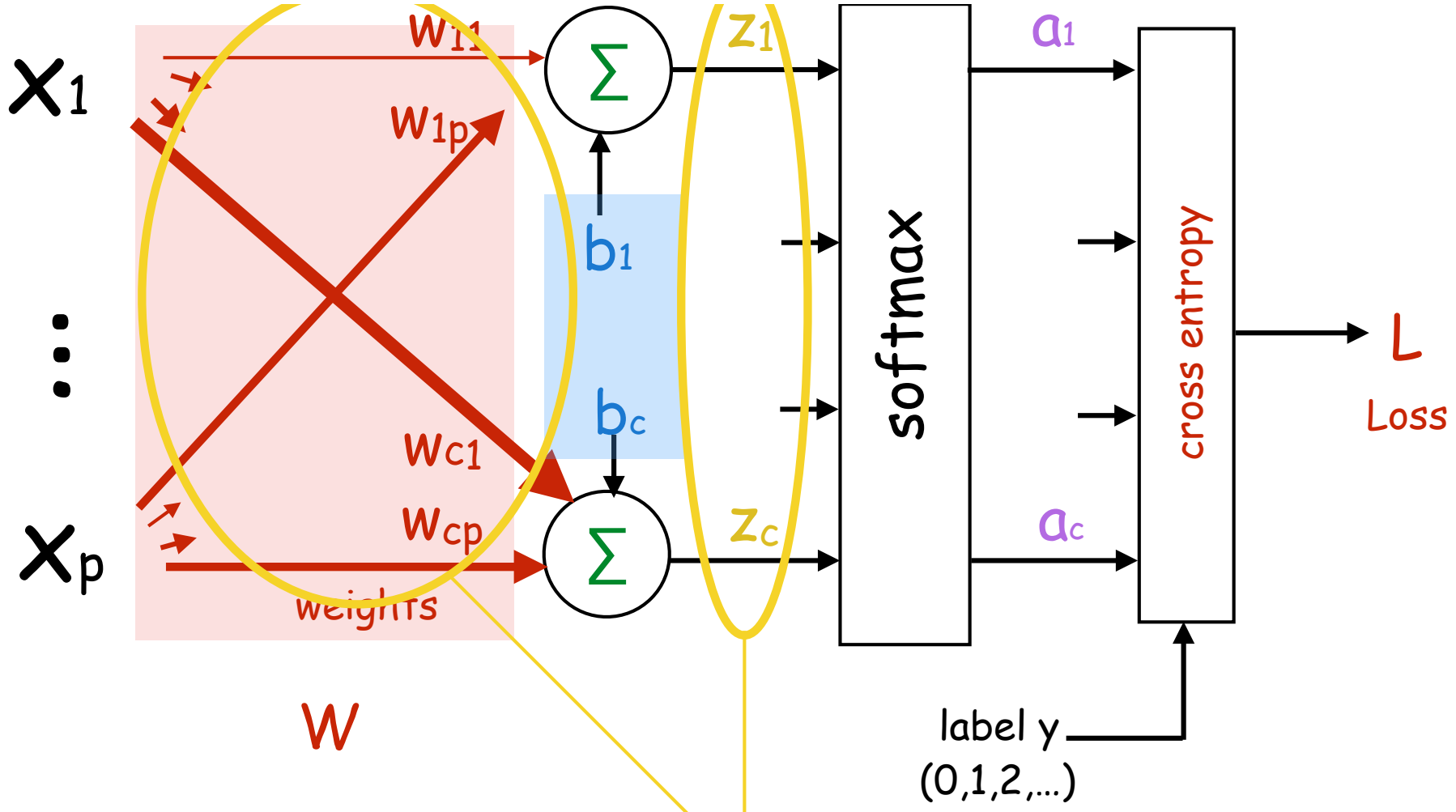


$$\frac{\partial z}{\partial b}$$

$$\frac{\partial z_1}{\partial b_1} = 1$$

$$\frac{\partial z_i}{\partial b_i} = 1$$

$$\frac{\partial z_c}{\partial b_c} = 1$$



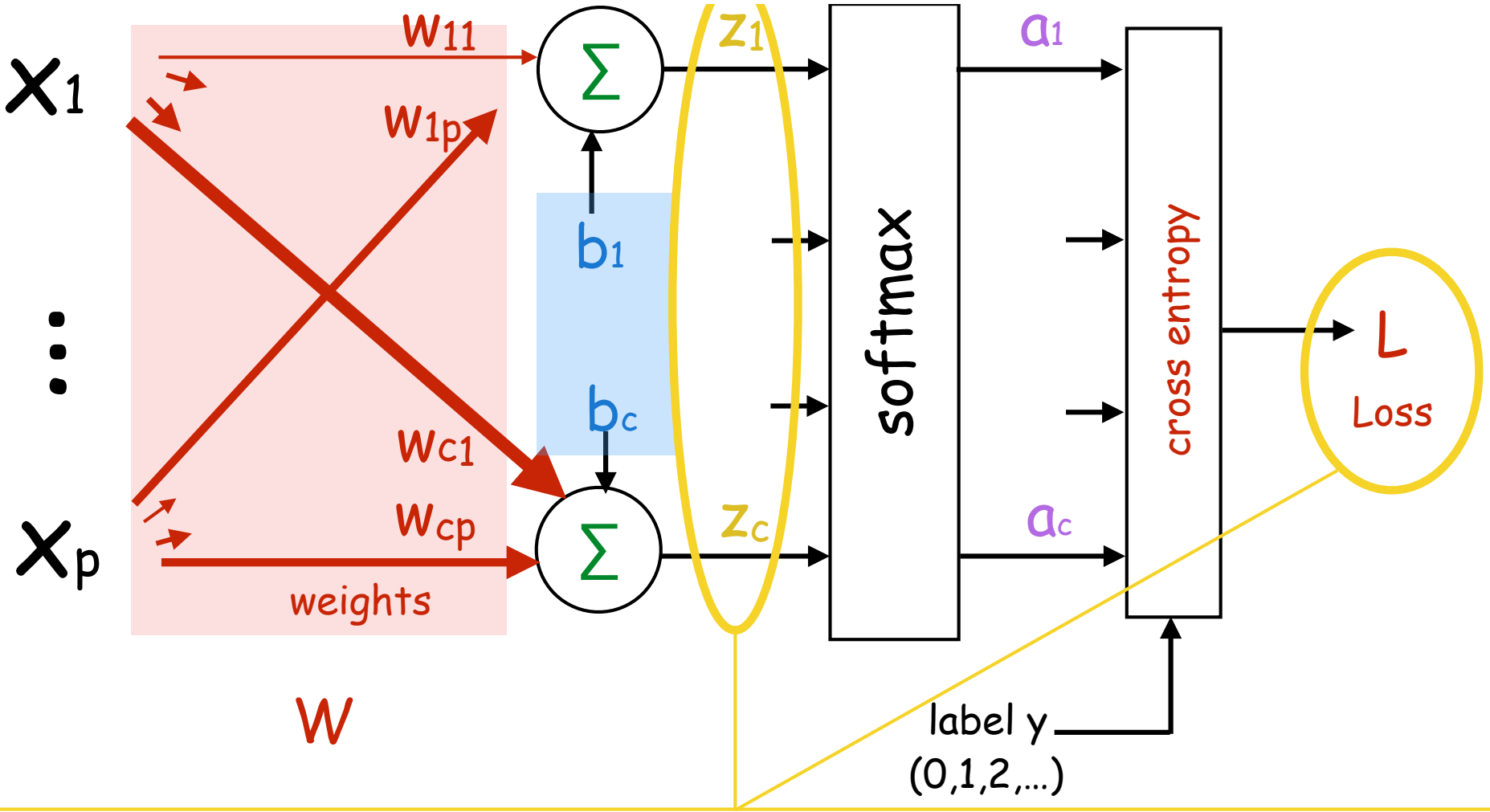
$$\frac{\partial z}{\partial W}$$

$$\frac{\partial z_1}{\partial w_{1*}} = \frac{\partial z_1}{\partial w_{11}}, \cdots, \frac{\partial z_1}{\partial w_{1p}}$$

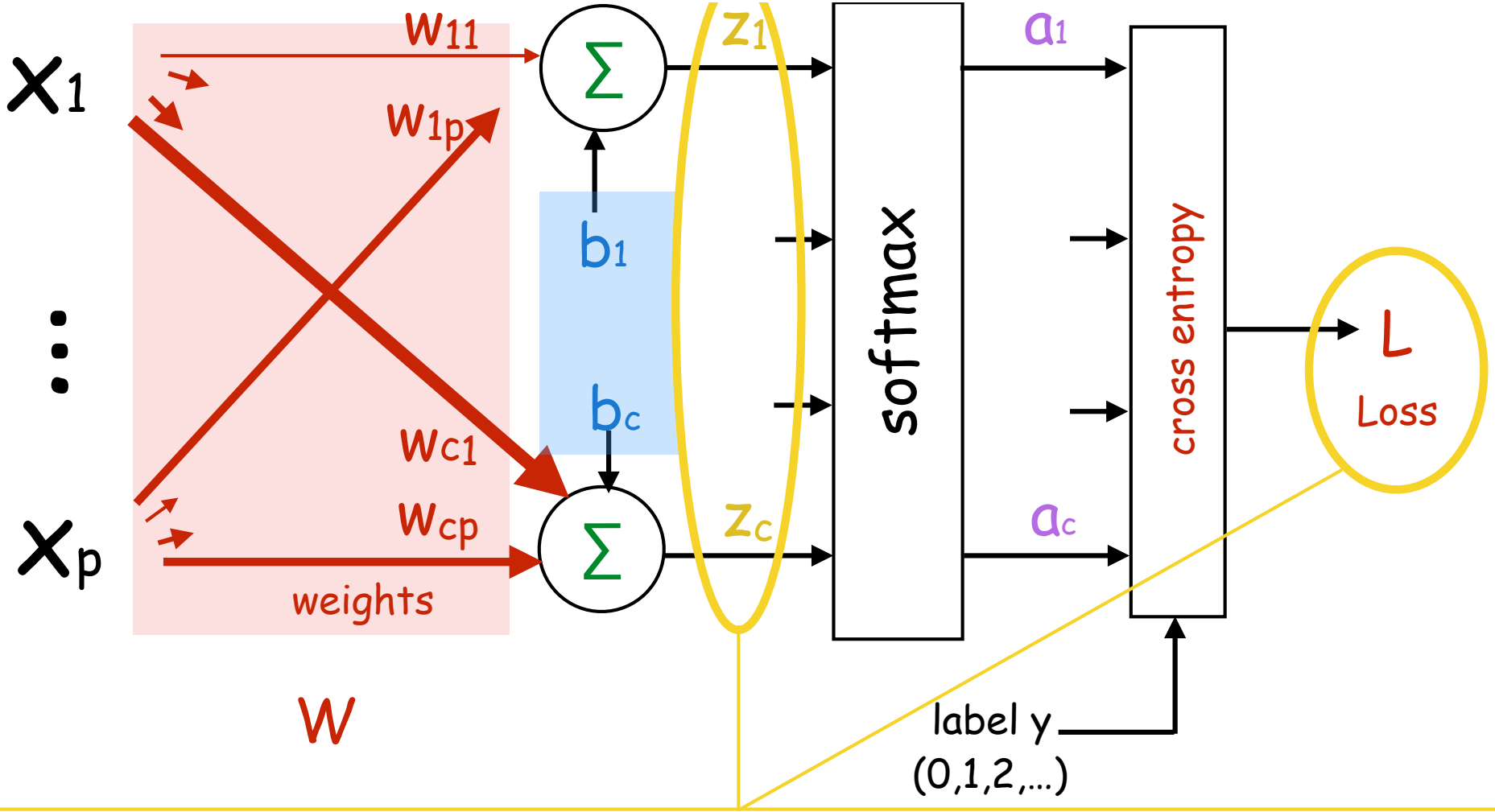
$$\frac{\partial z_i}{\partial w_{i*}} = \frac{\partial z_i}{\partial w_{i1}}, \cdots, \frac{\partial z_i}{\partial w_{ip}}$$

$$\frac{\partial z_c}{\partial w_{c*}} = \frac{\partial z_c}{\partial w_{c1}}, \cdots, \frac{\partial z_c}{\partial w_{cp}}$$

$$= \begin{matrix} x_1 & x_2 & \cdots & x_p \\ x_1 & x_2 & \cdots & x_p \\ x_1 & x_2 & \cdots & x_p \end{matrix}$$



$$\frac{\partial L}{\partial z} = \left(\frac{\partial L}{\partial z_1}, \ldots, \frac{\partial L}{\partial z_i}, \ldots, \frac{\partial L}{\partial z_c}\right) = ?$$



Chain Rule

$$\frac{\partial L}{\partial z} = \frac{\partial L}{\partial a} \times \frac{\partial a}{\partial z}$$

vector (1 by c) vector (1 by c) matrix (c by c)

$$\frac{\partial L}{\partial z} = \frac{\partial L}{\partial a} \times \frac{\partial a}{\partial z}$$

vector (1 by c) vector (1 by c) matrix (c by c)

$$\left(\frac{\partial L}{\partial z_1}, \ldots, \frac{\partial L}{\partial z_i}, \ldots, \frac{\partial L}{\partial z_c}\right) =$$

$$\left(\frac{\partial L}{\partial a_1}, \ldots, \frac{\partial L}{\partial a_i}, \ldots, \frac{\partial L}{\partial a_c}\right) \times \begin{bmatrix} \frac{\partial a_1}{\partial z_1} & \cdots & \frac{\partial a_1}{\partial z_c} \\ \vdots & & \vdots \\ \frac{\partial a_c}{\partial z_1} & & \frac{\partial a_c}{\partial z_c} \end{bmatrix}$$

$$\frac{\partial L}{\partial z} = \frac{\partial L}{\partial a} \times \frac{\partial a}{\partial z}$$

vector (1 by c) vector (1 by c) matrix (c by c)

$$\left(\frac{\partial L}{\partial z_1}, \ldots, \frac{\partial L}{\partial z_i}, \ldots, \frac{\partial L}{\partial z_c}\right) =$$

$$\left(\frac{\partial L}{\partial a_1}, \ldots, \frac{\partial L}{\partial a_i}, \ldots, \frac{\partial L}{\partial a_c}\right) \times \begin{bmatrix} \frac{\partial a_1}{\partial z_1} & \cdots & \frac{\partial a_1}{\partial z_c} \\ \vdots & & \vdots \\ \frac{\partial a_c}{\partial z_1} & & \frac{\partial a_c}{\partial z_c} \end{bmatrix}$$

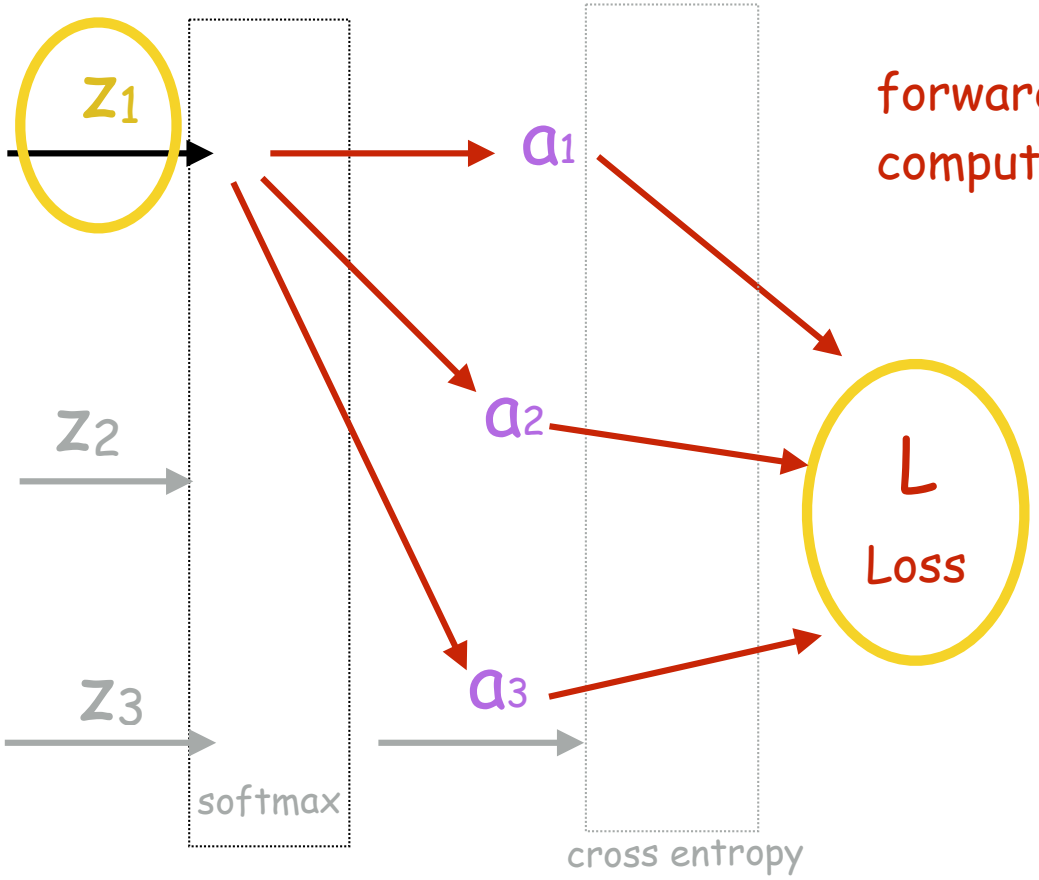
$$\frac{\partial L}{\partial z} = \frac{\partial L}{\partial a} \times \frac{\partial a}{\partial z}$$

vector (1 by c) vector (1 by c) matrix (c by c)

$$\left(\frac{\partial L}{\partial z_1}, \ldots, \frac{\partial L}{\partial z_i}, \ldots, \frac{\partial L}{\partial z_c}\right) =$$

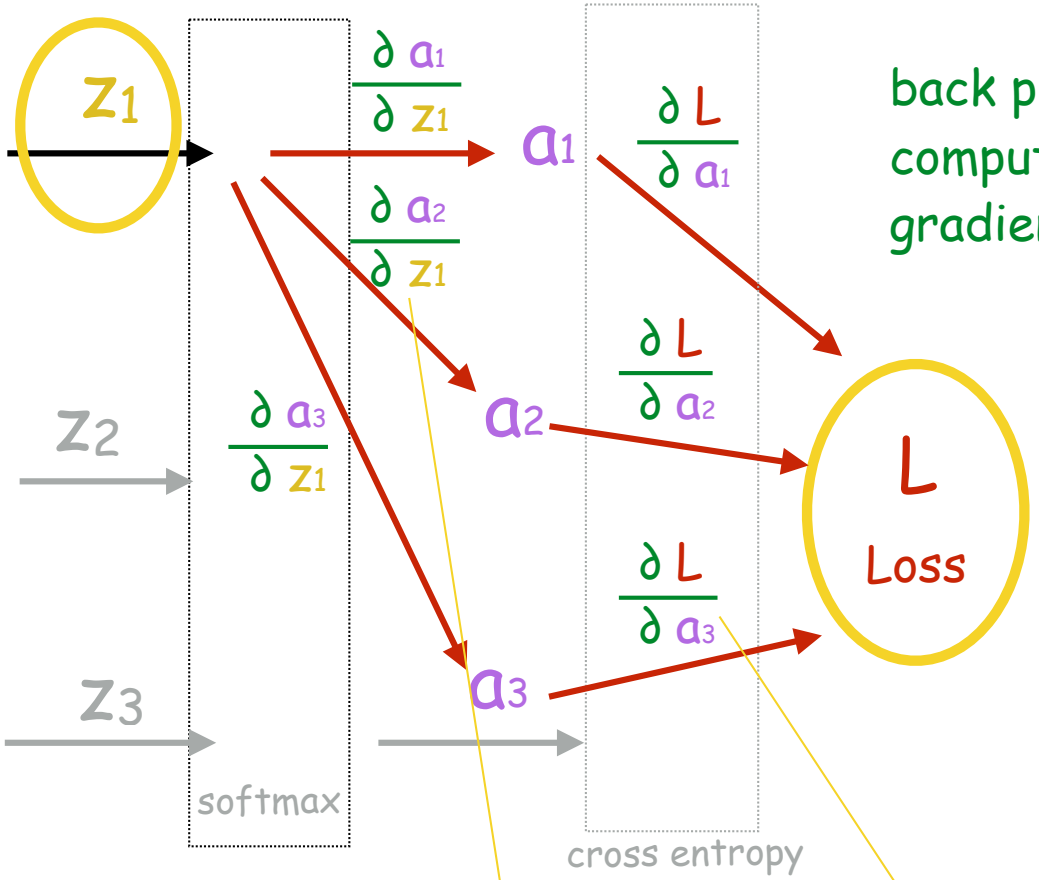
$$\left(\frac{\partial L}{\partial a_1}, \ldots, \frac{\partial L}{\partial a_i}, \ldots, \frac{\partial L}{\partial a_c}\right) \times \begin{bmatrix} \frac{\partial a_1}{\partial z_1} & \cdots & \frac{\partial a_1}{\partial z_c} \\ \vdots & & \vdots \\ \frac{\partial a_c}{\partial z_1} & & \frac{\partial a_c}{\partial z_c} \end{bmatrix}$$

$\frac{\partial L}{\partial z_1}$



$$\left(\frac{\partial L}{\partial a_1}, \ldots, \frac{\partial L}{\partial a_i}, \ldots, \frac{\partial L}{\partial a_c} \right) \times \begin{bmatrix} \frac{\partial a_1}{\partial z_1} & \cdots & \frac{\partial a_1}{\partial z_c} \\ \vdots & & \vdots \\ \frac{\partial a_c}{\partial z_1} & & \frac{\partial a_c}{\partial z_c} \end{bmatrix}$$

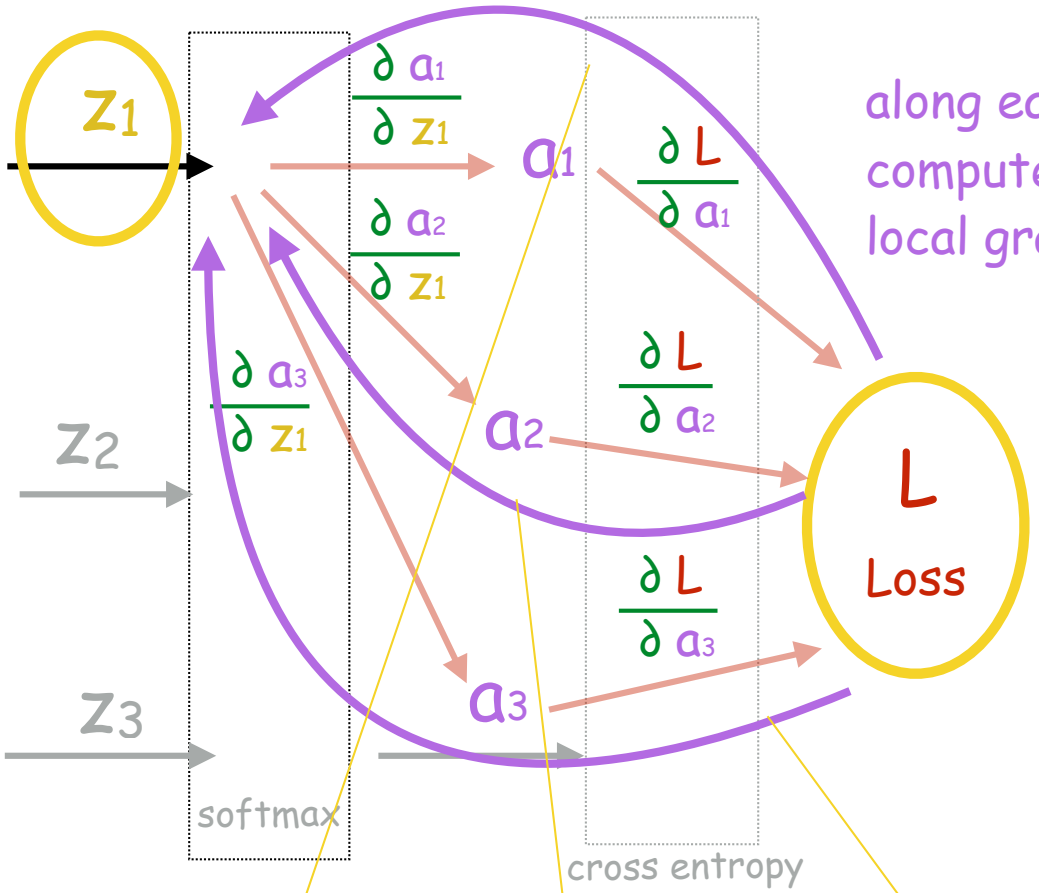
$\frac{\partial L}{\partial z_1}$



$$\frac{\partial L}{\partial a} = \left(\frac{\partial L}{\partial a_1} , \frac{\partial L}{\partial a_2} , \frac{\partial L}{\partial a_3} \right)$$

$$\frac{\partial a}{\partial z_1} = \left(\frac{\partial a_1}{\partial z_1} , \frac{\partial a_2}{\partial z_1} , \frac{\partial a_3}{\partial z_1} \right)$$

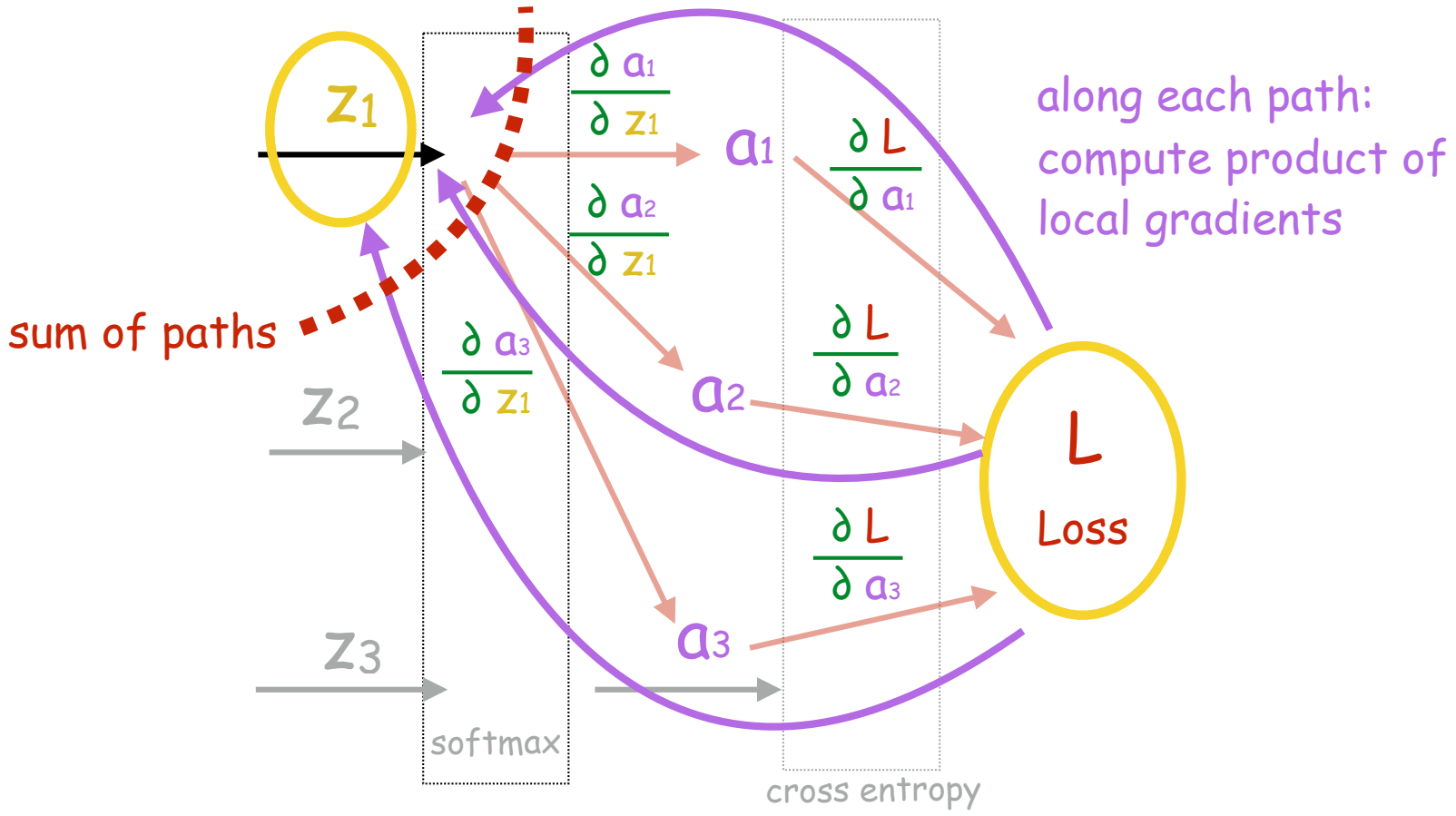
$\frac{\partial L}{\partial z_1}$



$$\frac{\partial L}{\partial a} = \left(\frac{\partial L}{\partial a_1}, \frac{\partial L}{\partial a_2}, \frac{\partial L}{\partial a_3}\right) \times$$

$$\frac{\partial a}{\partial z_1} = \left(\frac{\partial a_1}{\partial z_1}, \frac{\partial a_2}{\partial z_1}, \frac{\partial a_3}{\partial z_1}\right)$$

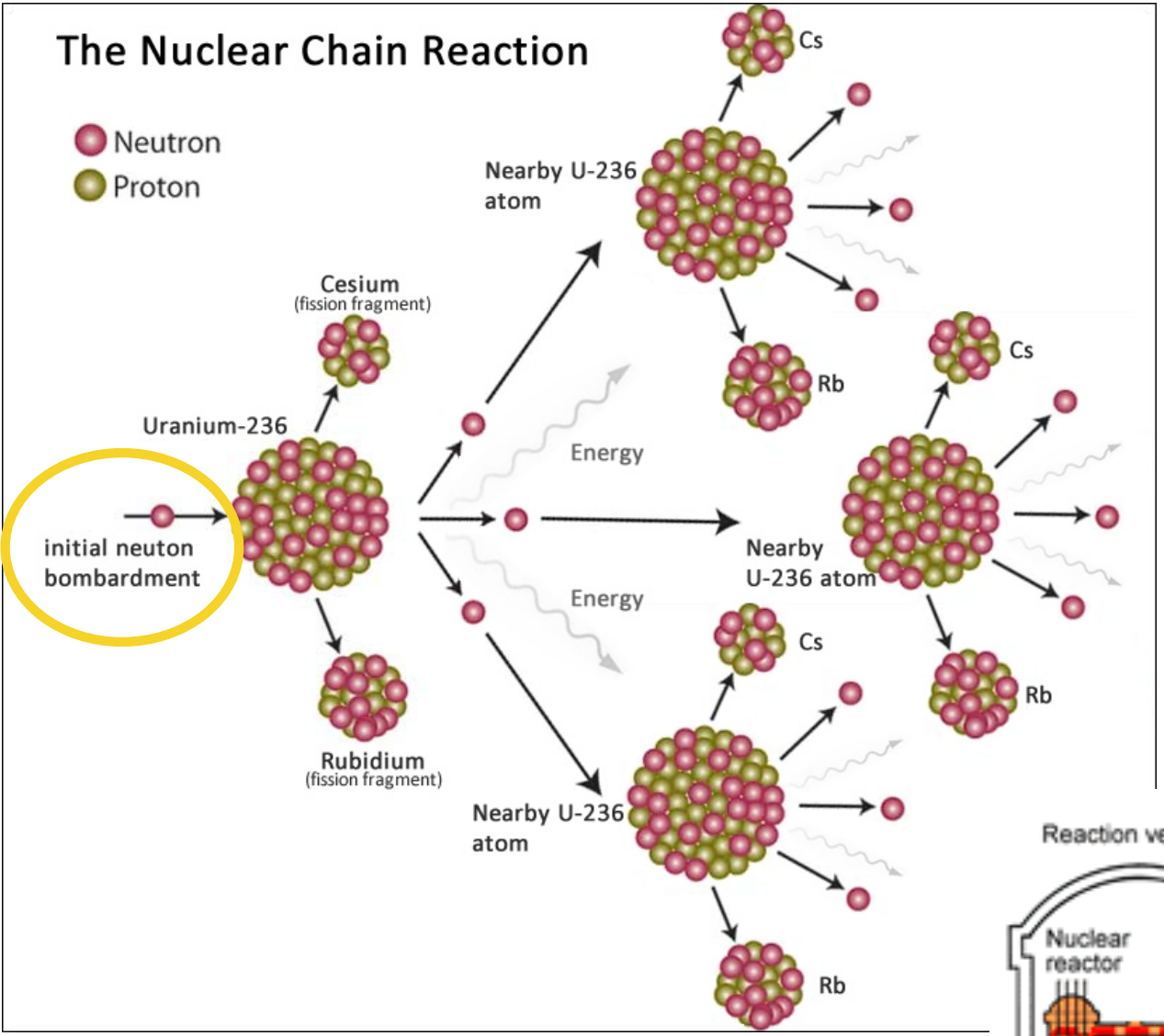
$\frac{\partial L}{\partial z_1}$



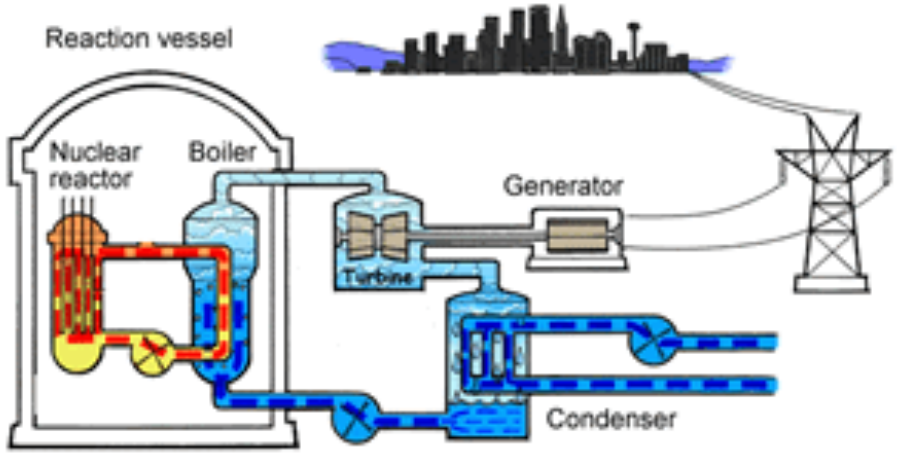
product along each path

sum of paths

$$\frac{\partial L}{\partial z_1} = \left(\frac{\partial L}{\partial a_1} \times \frac{\partial a_1}{\partial z_1}\right) + \left(\frac{\partial L}{\partial a_2} \times \frac{\partial a_2}{\partial z_1}\right) + \left(\frac{\partial L}{\partial a_3} \times \frac{\partial a_3}{\partial z_1}\right)$$



Electricity



gradient of Electricity w.r.t. Initial Neuton

$$\frac{\partial L}{\partial W} = \frac{\partial L}{\partial z}^{T} \times \frac{\partial z}{\partial W}$$

matrix (c by p) vector (c by 1) matrix (c by p)

$$\left(\frac{\partial L}{\partial z_1}, \ldots, \frac{\partial L}{\partial z_i}, \ldots, \frac{\partial L}{\partial z_c}\right)^{T} \times$$

transpose

element-wise product

$$\begin{bmatrix} \frac{\partial z_1}{\partial w_{11}} & \cdots & \frac{\partial z_1}{\partial w_{1p}} \\ \frac{\partial z_i}{\partial w_{i1}} & \cdots & \frac{\partial z_i}{\partial w_{ip}} \\ \frac{\partial z_c}{\partial w_{c1}} & \cdots & \frac{\partial z_c}{\partial w_{cp}} \end{bmatrix}$$

$$= \begin{bmatrix} \frac{\partial L}{\partial z_1}\frac{\partial z_1}{\partial w_{11}} & \cdots & \frac{\partial L}{\partial z_1}\frac{\partial z_1}{\partial w_{1p}} \\ \frac{\partial L}{\partial z_i}\frac{\partial z_i}{\partial w_{i1}} & \cdots & \frac{\partial L}{\partial z_i}\frac{\partial z_i}{\partial w_{ip}} \\ \frac{\partial L}{\partial z_c}\frac{\partial z_c}{\partial w_{c1}} & \cdots & \frac{\partial L}{\partial z_c}\frac{\partial z_c}{\partial w_{cp}} \end{bmatrix}$$

Softmax Regression (train)

initialize **W** and **b**

Loop for n_epoch iterations:

Loop for each training instance (**x**, y) in training set

forward pass to compute **z**, **a** and L for the instance

backward pass to compute local gradients

$$\frac{\partial L}{\partial a}\ \frac{\partial a}{\partial z}\ \frac{\partial z}{\partial b}\ \frac{\partial z}{\partial W}$$

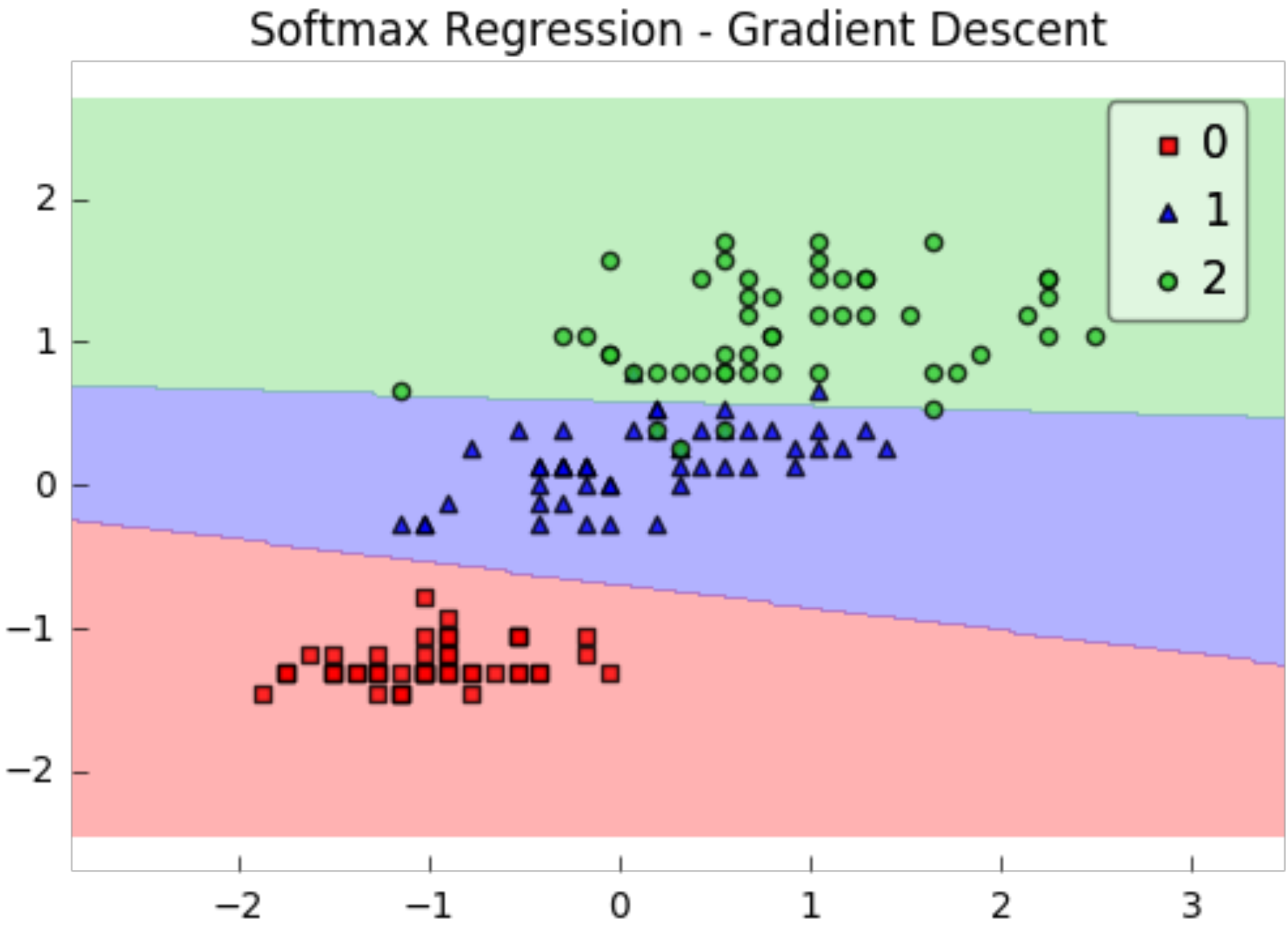
compute global gradients using chain rule

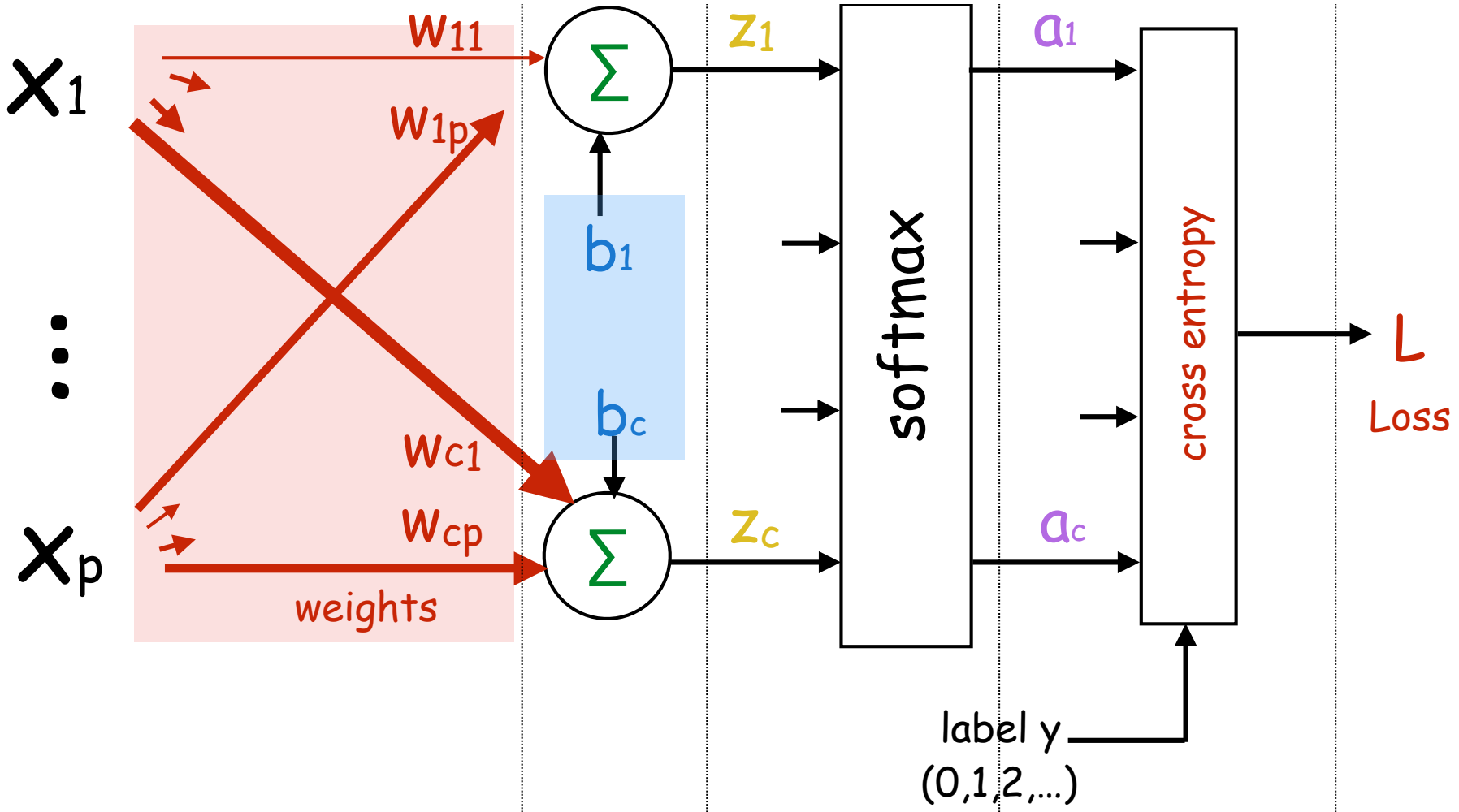
$$\frac{\partial L}{\partial W}\ \frac{\partial L}{\partial b}$$

update the parameters **W** and **b**

$$W \leftarrow W - a\frac{\partial L}{\partial W}$$

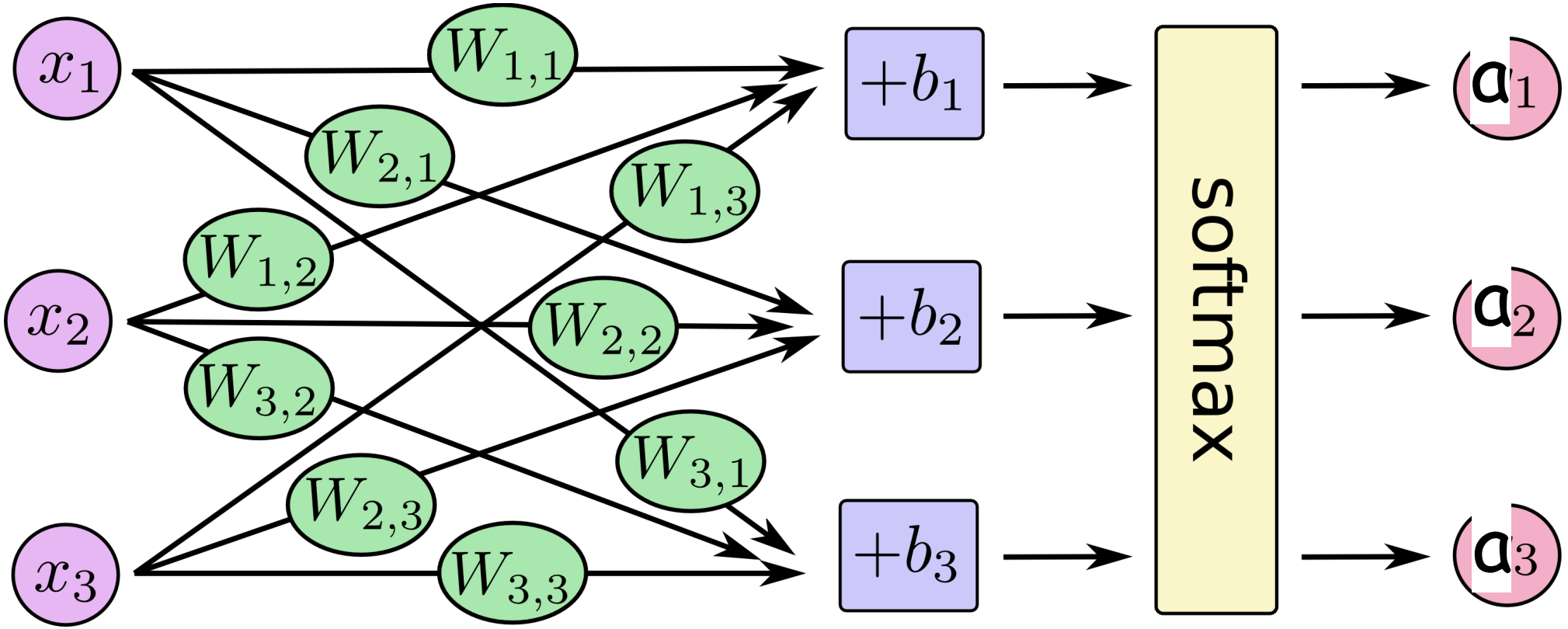
$$b \leftarrow b - a\frac{\partial L}{\partial b}$$


Softmax Regression - Gradient Descent
0
1
2
2
1
0
−1
−2
−2
−1
0
1
2
3



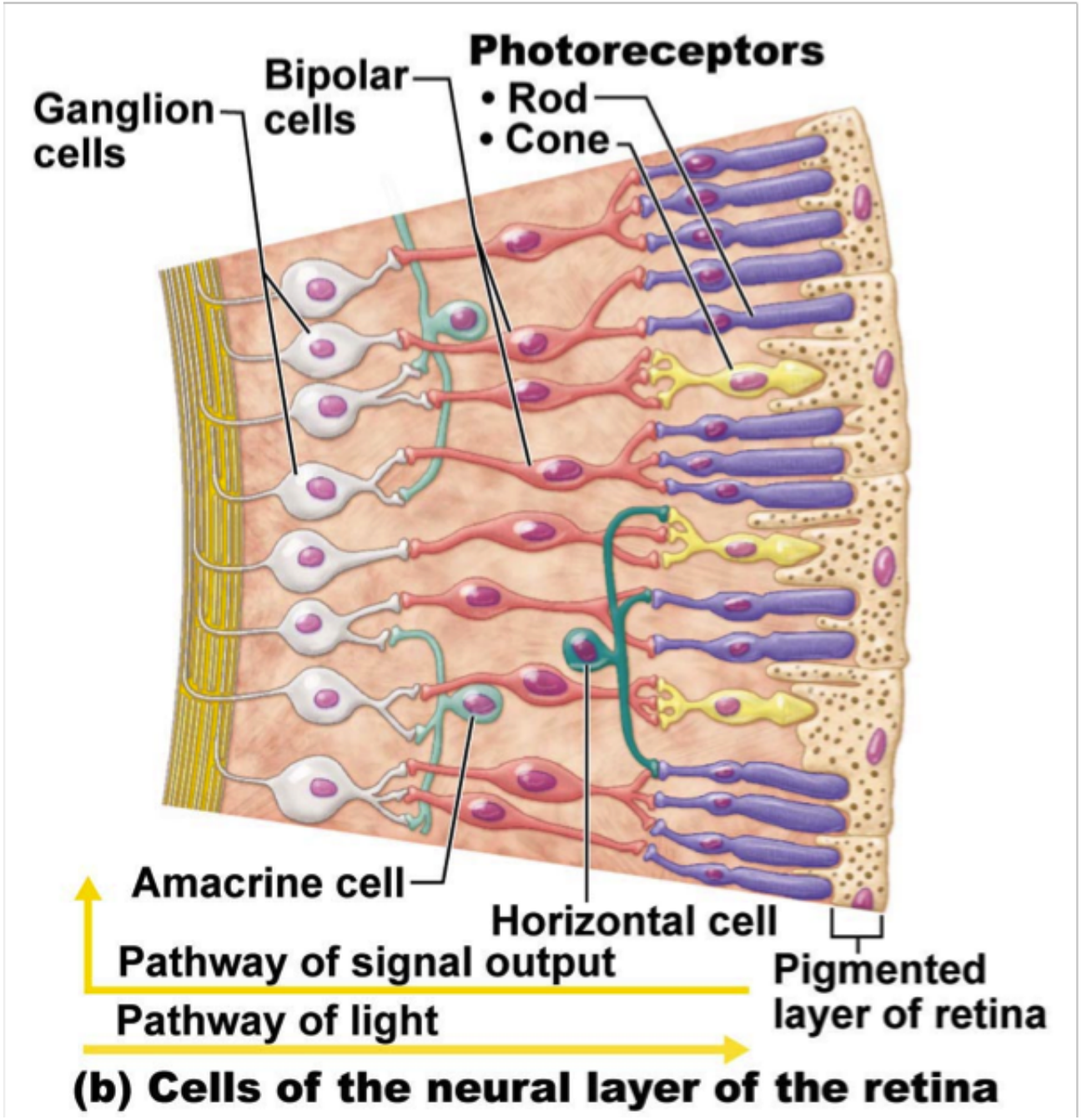
weights	biases	logits	activations	loss
W	**b**	**z**	**a**	**L**

Softmax Regression

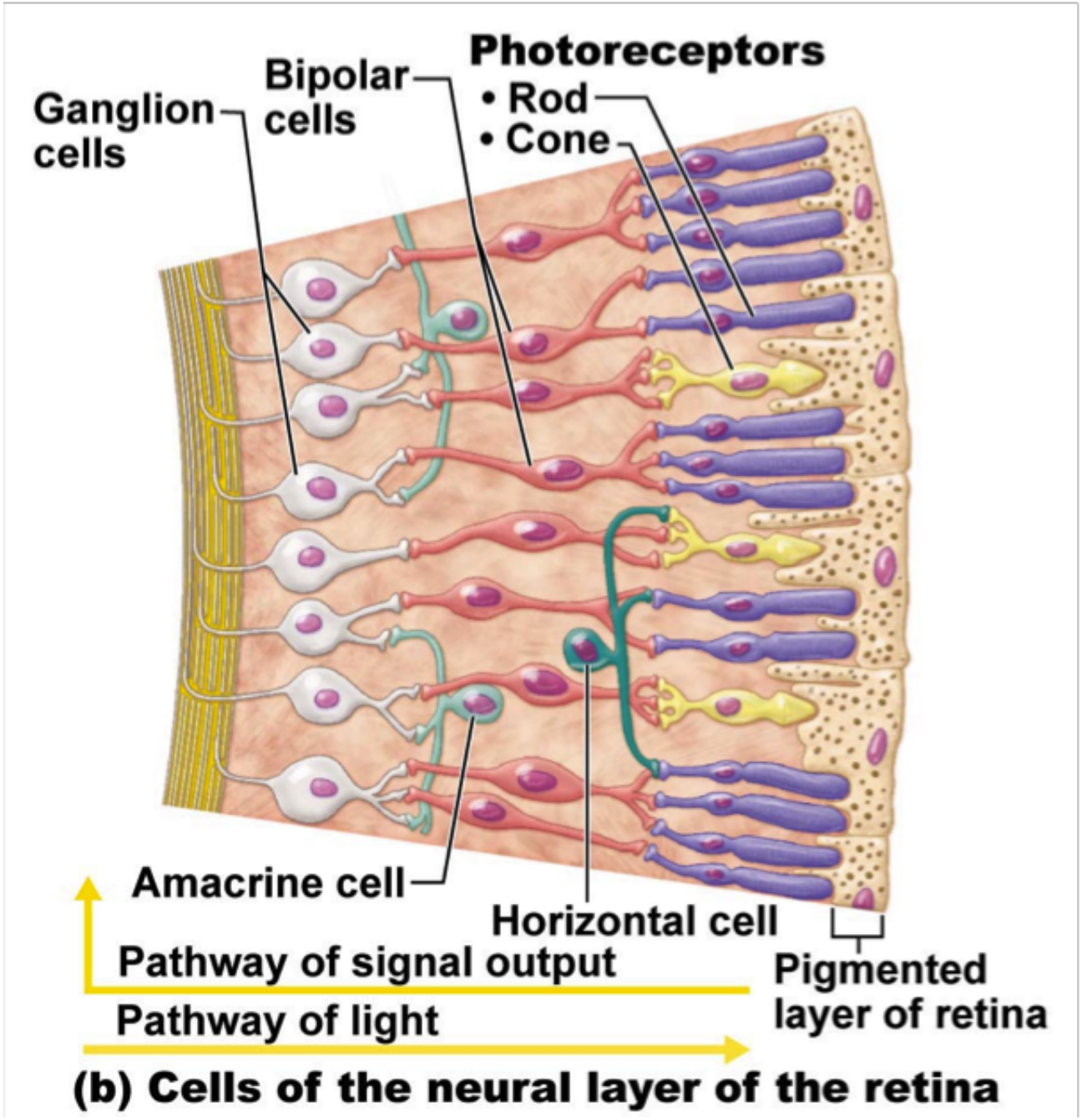


$$\begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = \text{softmax} \left(\begin{bmatrix} W_{1,1} & W_{1,2} & W_{1,3} \\ W_{2,1} & W_{2,2} & W_{2,3} \\ W_{3,1} & W_{3,2} & W_{3,3} \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} \right)$$

Neural Layer

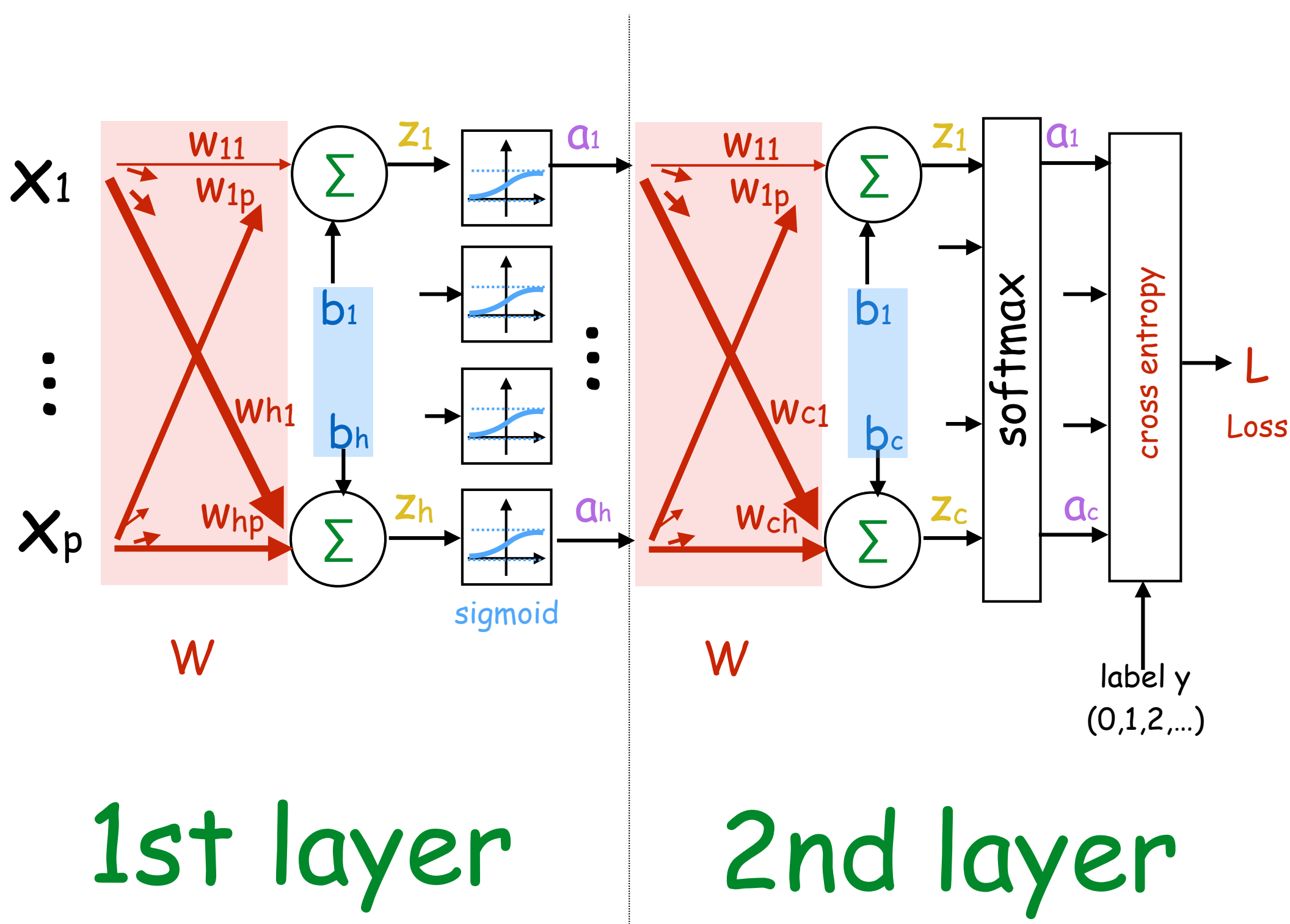


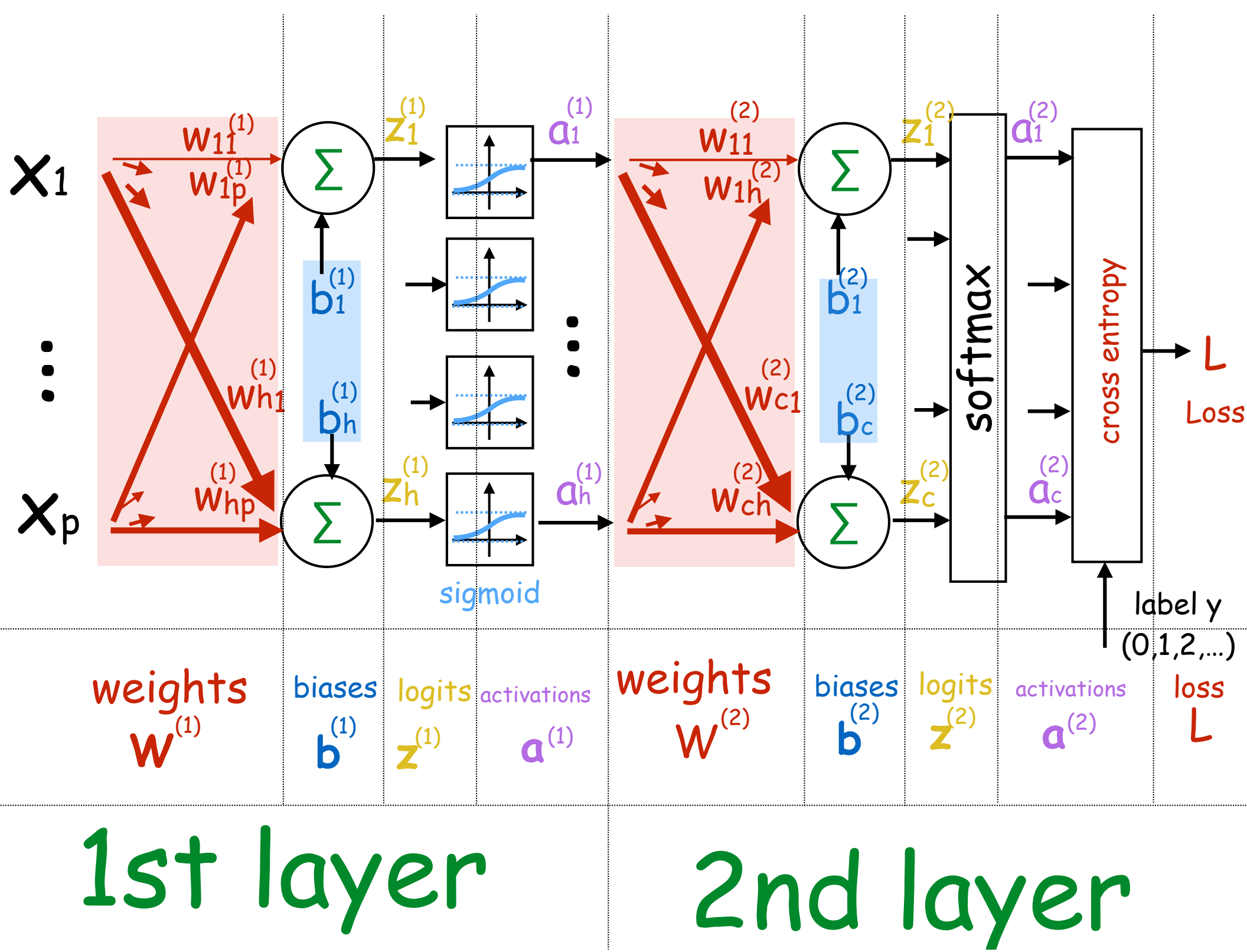
Multiple Neural Layers ?



Fully Connected Neural Network

Demo: http://playground.tensorflow.org/


x1
xp
w11
w1p
wh1
whp
b1
bh
z1
zh
a1
ah
sigmoid
W
wc1
wch
bc
zc
ac
softmax
cross entropy
L
Loss
label y
(0,1,2,...)
W
1st layer
2nd layer


x1
xp
w11(1)
w1p(1)
whp(1)
wh1(1)
b1(1)
bh(1)
z1(1)
zh(1)
a1(1)
ah(1)
sigmoid
w11(2)
w1h(2)
wc1(2)
wch(2)
b1(2)
bc(2)
z1(2)
zc(2)
a1(2)
ac(2)
softmax
cross entropy
L
Loss
label y
(0,1,2,...)
weights
w(1)
biases
b(1)
logits
z(1)
activations
a(1)
weights
w(2)
biases
b(2)
logits
z(2)
activations
a(2)
loss
L
1st layer
2nd layer

$$\frac{\partial L}{\partial a^{(1)}} = \frac{\partial L}{\partial z^{(2)}} \times \frac{\partial z^{(2)}}{\partial a^{(1)}}$$

vector length h | vector of length c | matrix (c by h)

$$\left(\frac{\partial L}{\partial a_1^{(1)}}, \ldots, \frac{\partial L}{\partial a_c^{(1)}}\right)$$

$$= \left(\frac{\partial L}{\partial z_1^{(2)}}, \ldots, \frac{\partial L}{\partial z_c^{(2)}}\right) \times \begin{bmatrix} \frac{\partial z_1^{(2)}}{\partial a_1^{(1)}} & \cdots & \frac{\partial z_1^{(2)}}{\partial a_h^{(1)}} \\ \cdots & & \\ \frac{\partial z_c^{(2)}}{\partial a_1^{(1)}} & \cdots & \frac{\partial z_c^{(2)}}{\partial a_h^{(1)}} \end{bmatrix}$$