Machine Learning

CS 539
Worcester Polytechnic Institute
Department of Computer Science
Instructor: Prof. Kyumin Lee

Upcoming Schedule

- HW2 due date is June 18
- HW3 will be released on June 18
- Quiz2 will be taken on June 18
  - it will be available only during the day

Gradient Descent vs.
Stochastic Gradient Descent

Gradient Descent

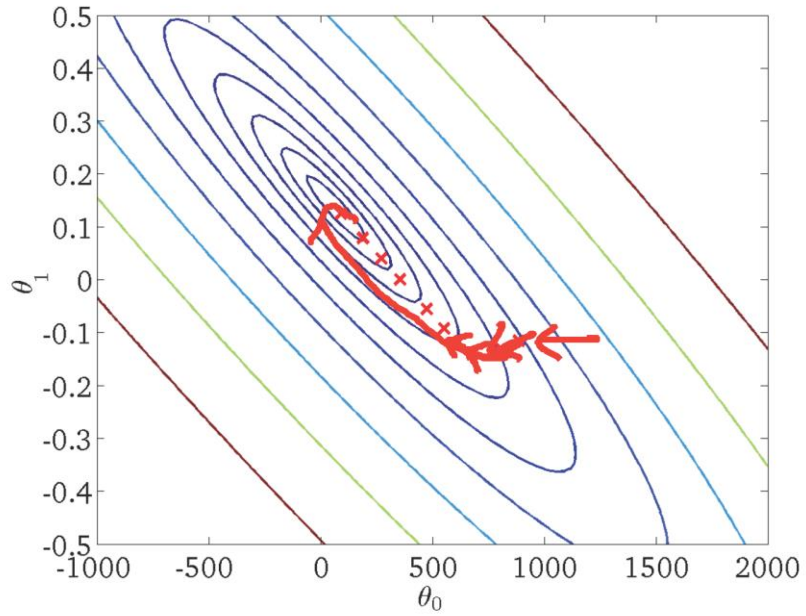
Batch Gradient Descent

Initialize θ

Repeat {

$$\theta_j \leftarrow \theta_j - \alpha \underbrace{\frac{1}{n}\sum_{i=1}^{n}\left(h_{\boldsymbol{\theta}}\left(\mathbf{x}_i\right) - y_i\right)x_{ij}}_{\frac{\partial}{\partial \theta_j} J(\boldsymbol{\theta})} \qquad \text{for } j = 0 \ldots d$$

}



Stochastic Gradient Descent

Initialize θ

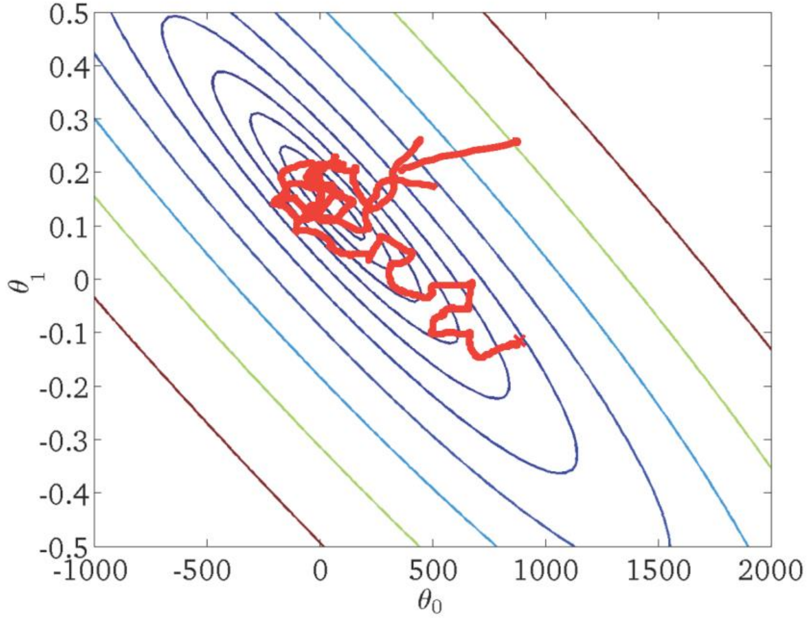
Randomly shuffle dataset

Repeat { (Typically 1 – 10x)

For $i = 1 \ldots n$, *do*

$$\theta_j \leftarrow \theta_j - \alpha \underbrace{\left(h_{\boldsymbol{\theta}}\left(\mathbf{x}_i\right) - y_i\right)x_{ij}}_{\frac{\partial}{\partial \theta_j} \text{cost}_{\boldsymbol{\theta}}(\mathbf{x}_i, y_i)} \qquad \text{for } j = 0 \ldots d$$

}



Adaptive alpha

Stochastic Gradient Descent

Initialize θ

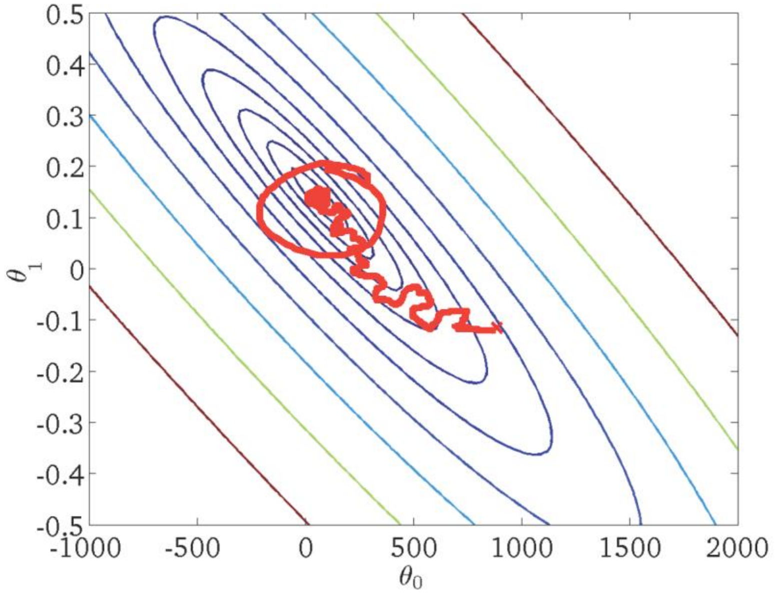
Randomly shuffle dataset

Repeat { (Typically 1 – 10x)

For $i = 1...n$, *do*

$$\theta_j \leftarrow \theta_j - \alpha \underbrace{\left(h_{\boldsymbol{\theta}}\left(\mathbf{x}_i\right) - y_i\right) x_{ij}}_{\frac{\partial}{\partial \theta_j} \text{cost}_{\boldsymbol{\theta}}(\mathbf{x}_i, y_i)} \quad \text{for } j = 0 \ldots d$$

}

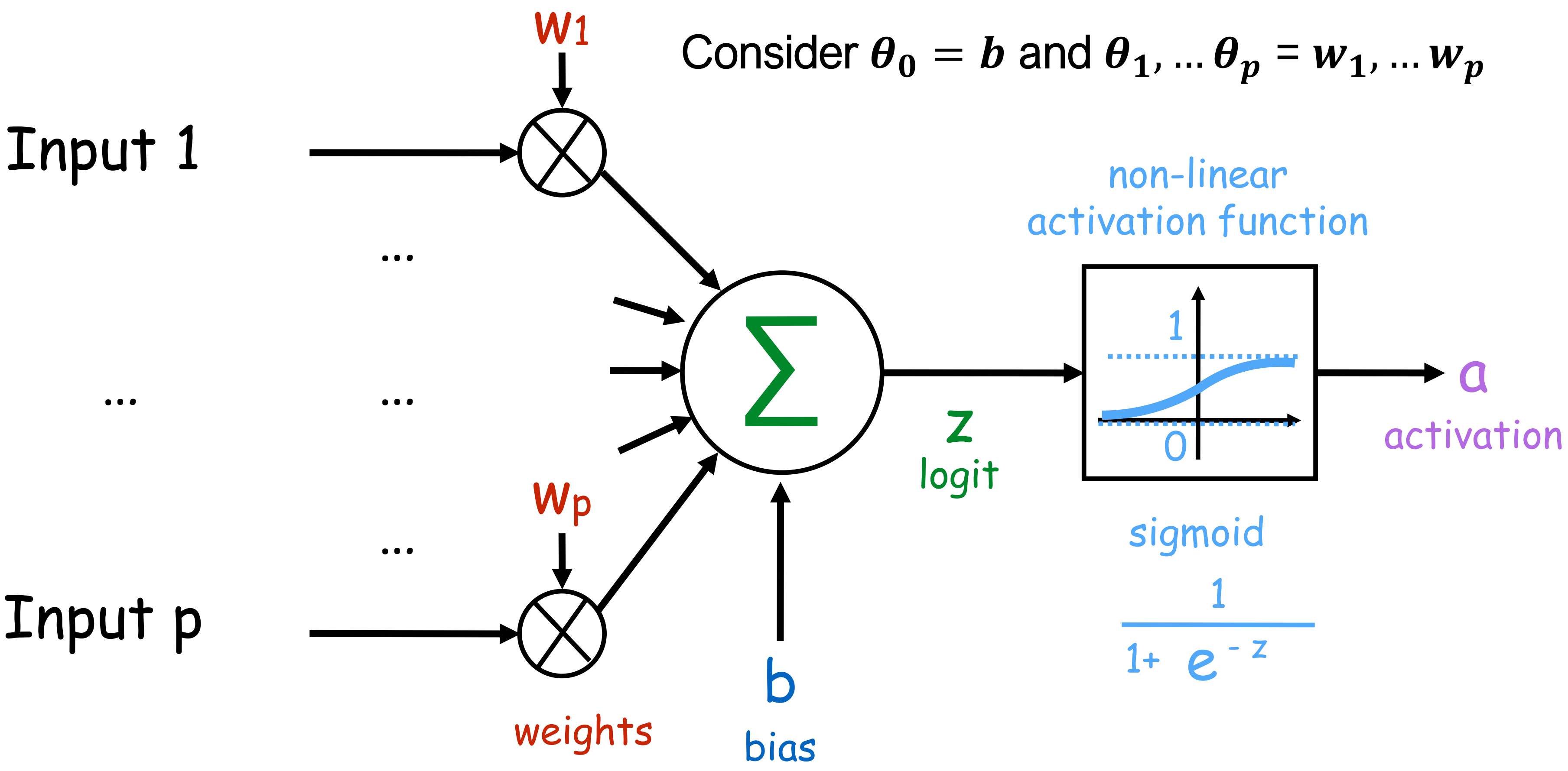


Learning rate α is typically held constant. Can slowly decrease α over time if we want θ to converge. (E.g. $\alpha = \frac{\texttt{const1}}{\texttt{iterationNumber + const2}}$)

Logistic Regression
(for hw3)
Will be released Next
Tuesday

Logistic Regression for hw3

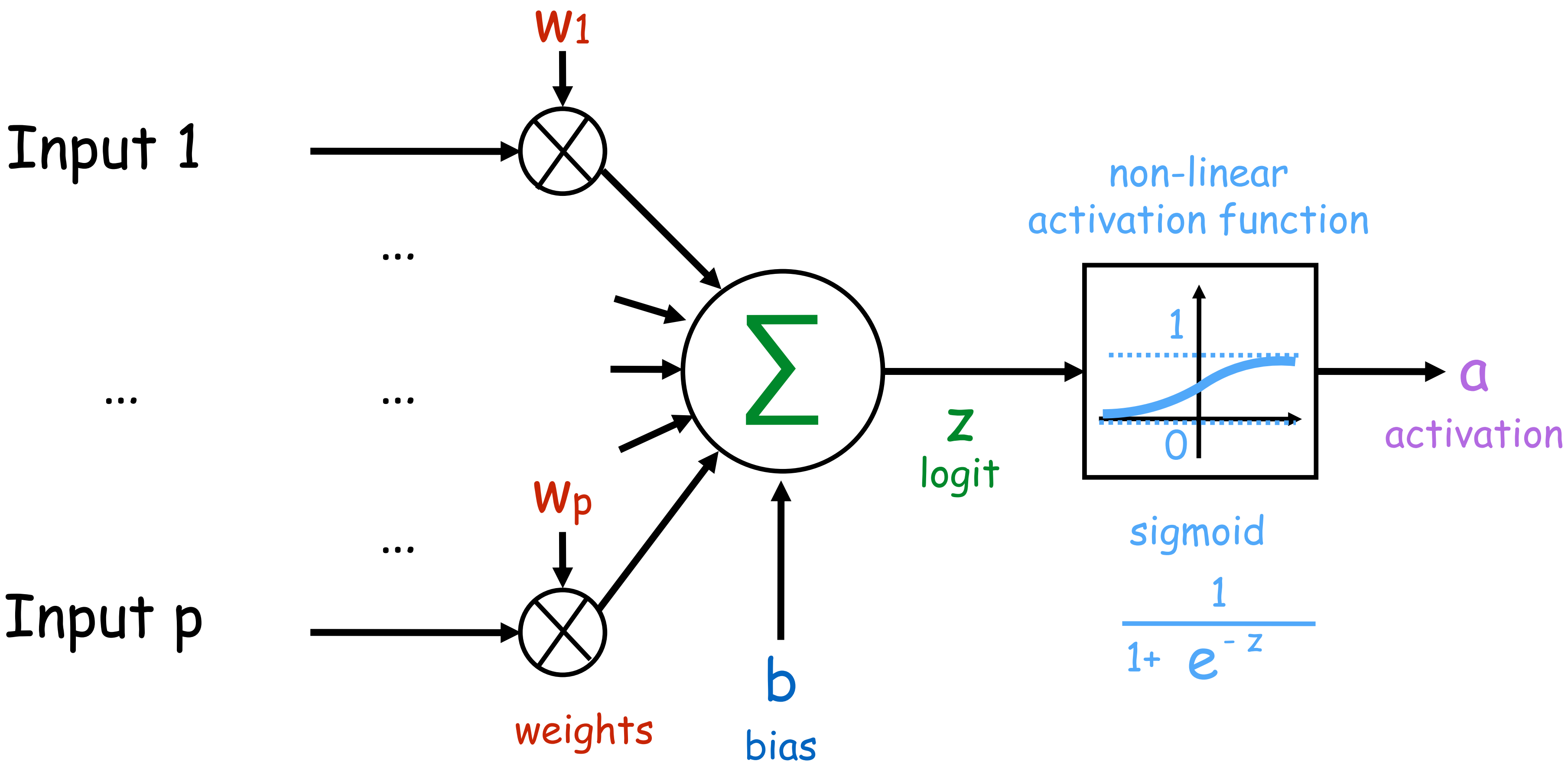
Consider $\boldsymbol{\theta_0} = \boldsymbol{b}$ and $\boldsymbol{\theta_1}, \ldots \boldsymbol{\theta_p} = \boldsymbol{w_1}, \ldots \boldsymbol{w_p}$



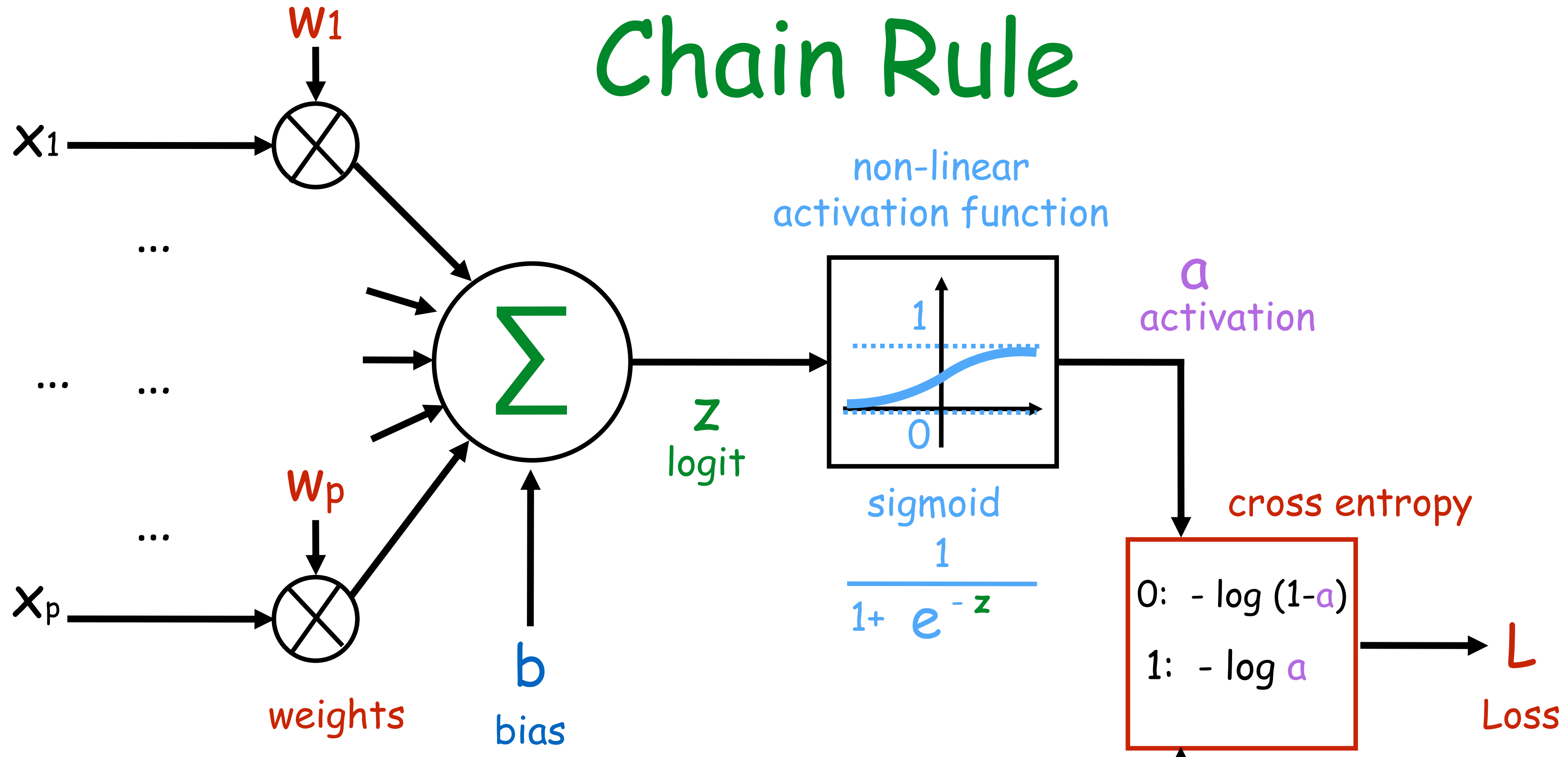
$a = Pr\,(\,\text{label} = + \mid \text{inputs} = x\,)$

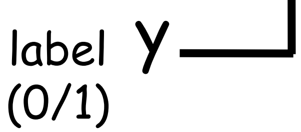
output value represents the probability of the instance having positive label

Parameters w, b



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

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

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

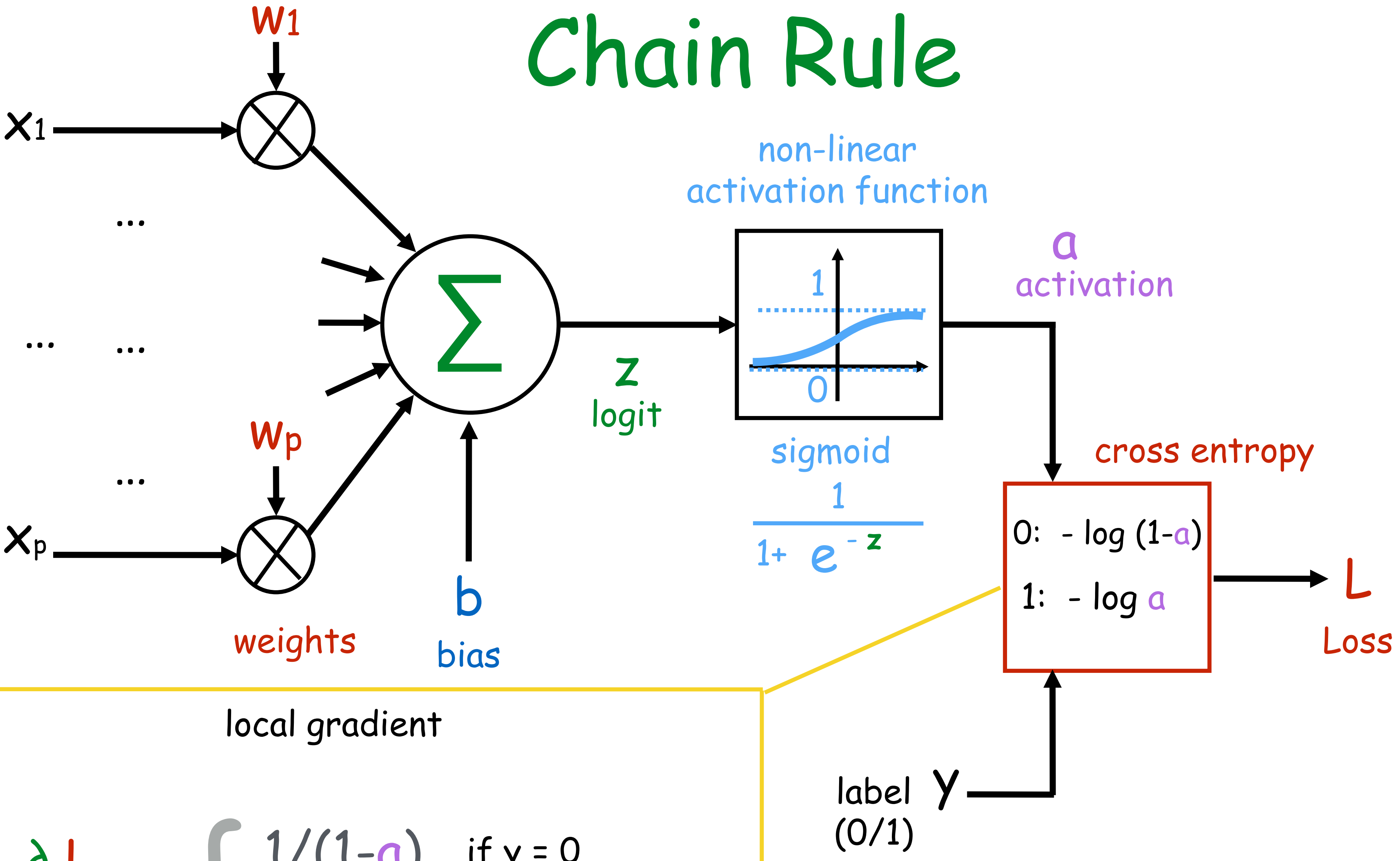
global gradient   local gradient   local gradient

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

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

global gradient   local gradient   local gradient   local gradient

Chain Rule

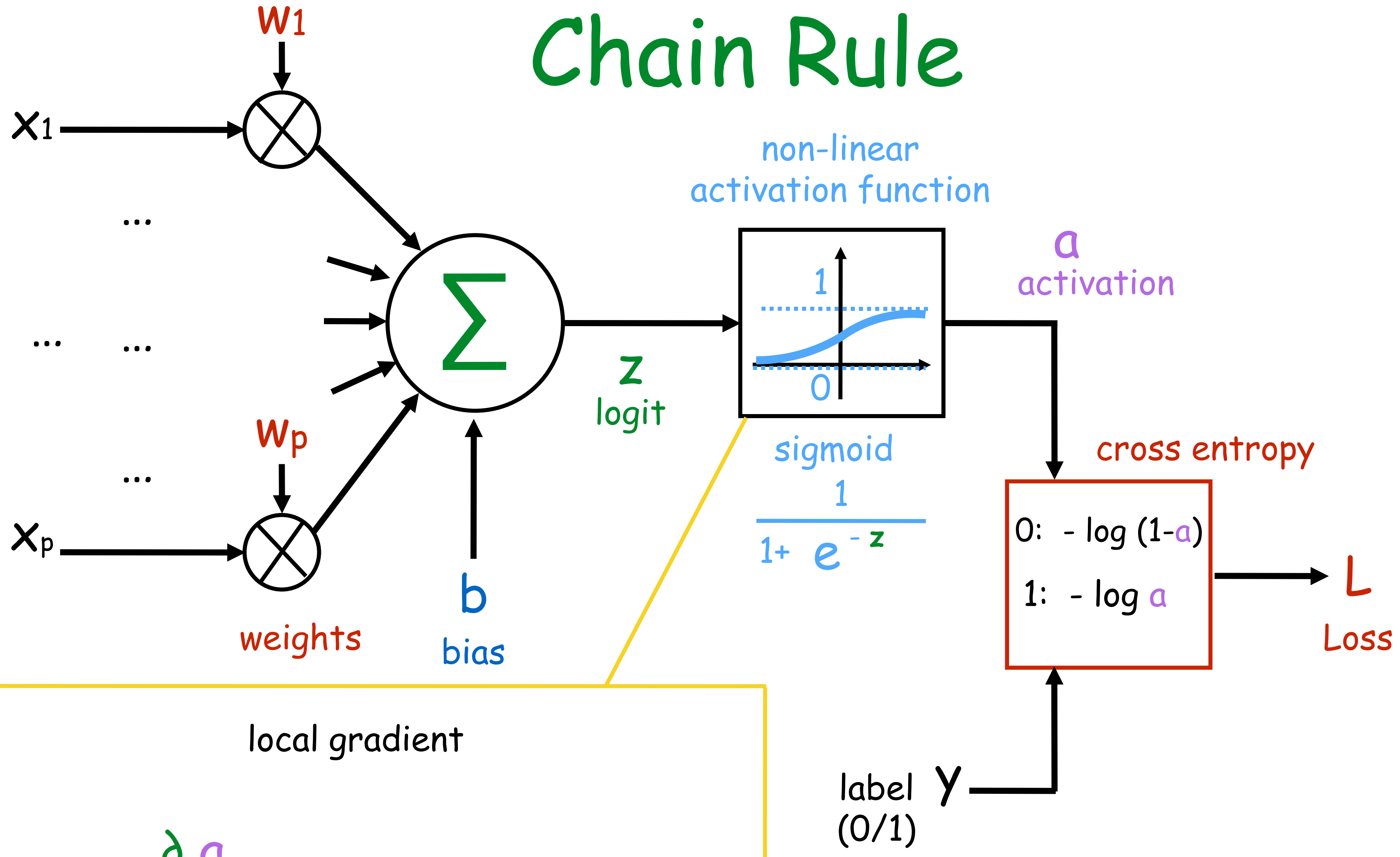


local gradient

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

https://www.derivative-calculator.net/

Chain Rule

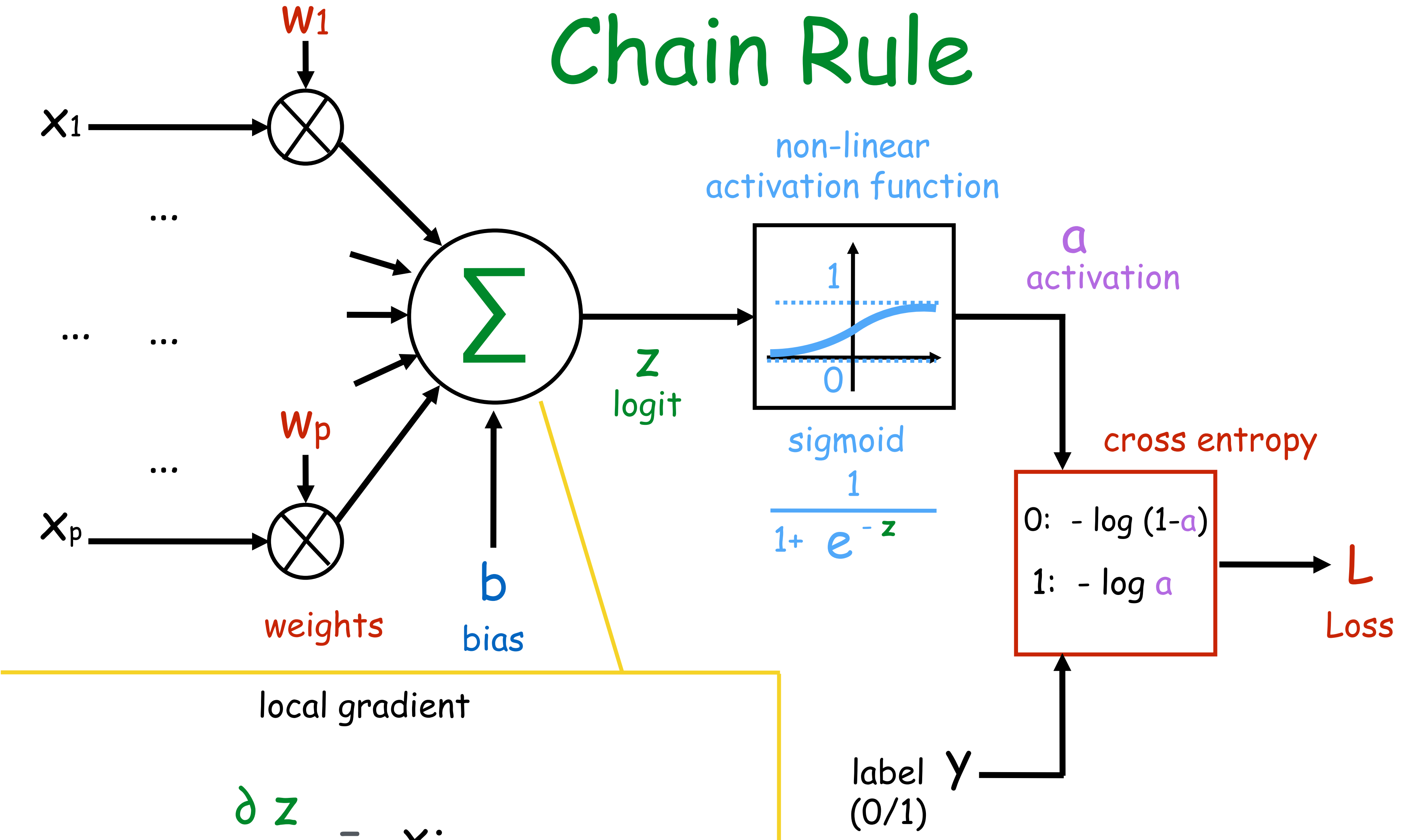


local gradient

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

https://stats.stackexchange.com/questions/278771/how-is-the-cost-function-from-logistic-regression-derivated

Chain Rule

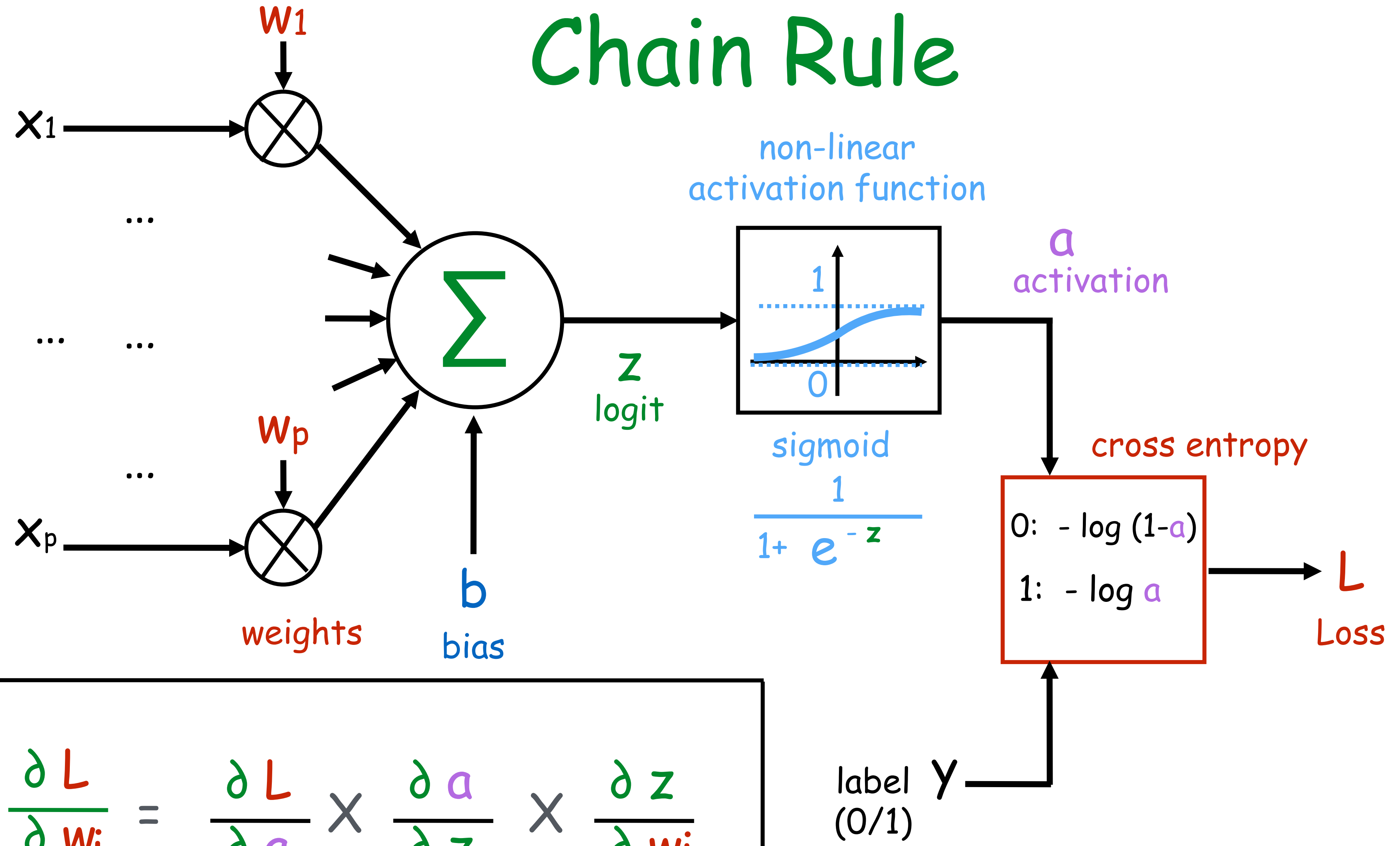


local gradient

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

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

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

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

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

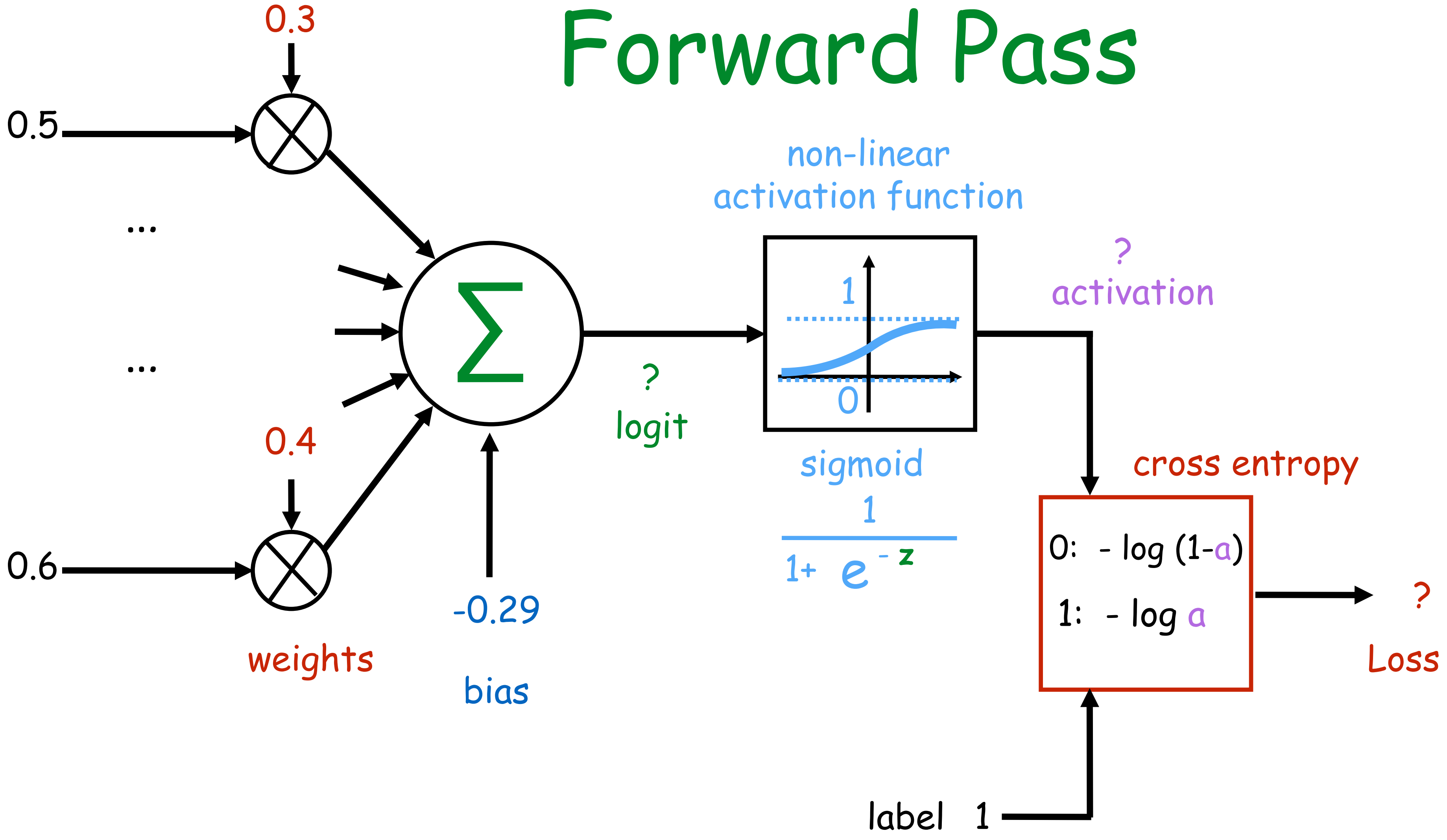
update the parameters **w** and b

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

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

Example
(for your reference)

Forward Pass

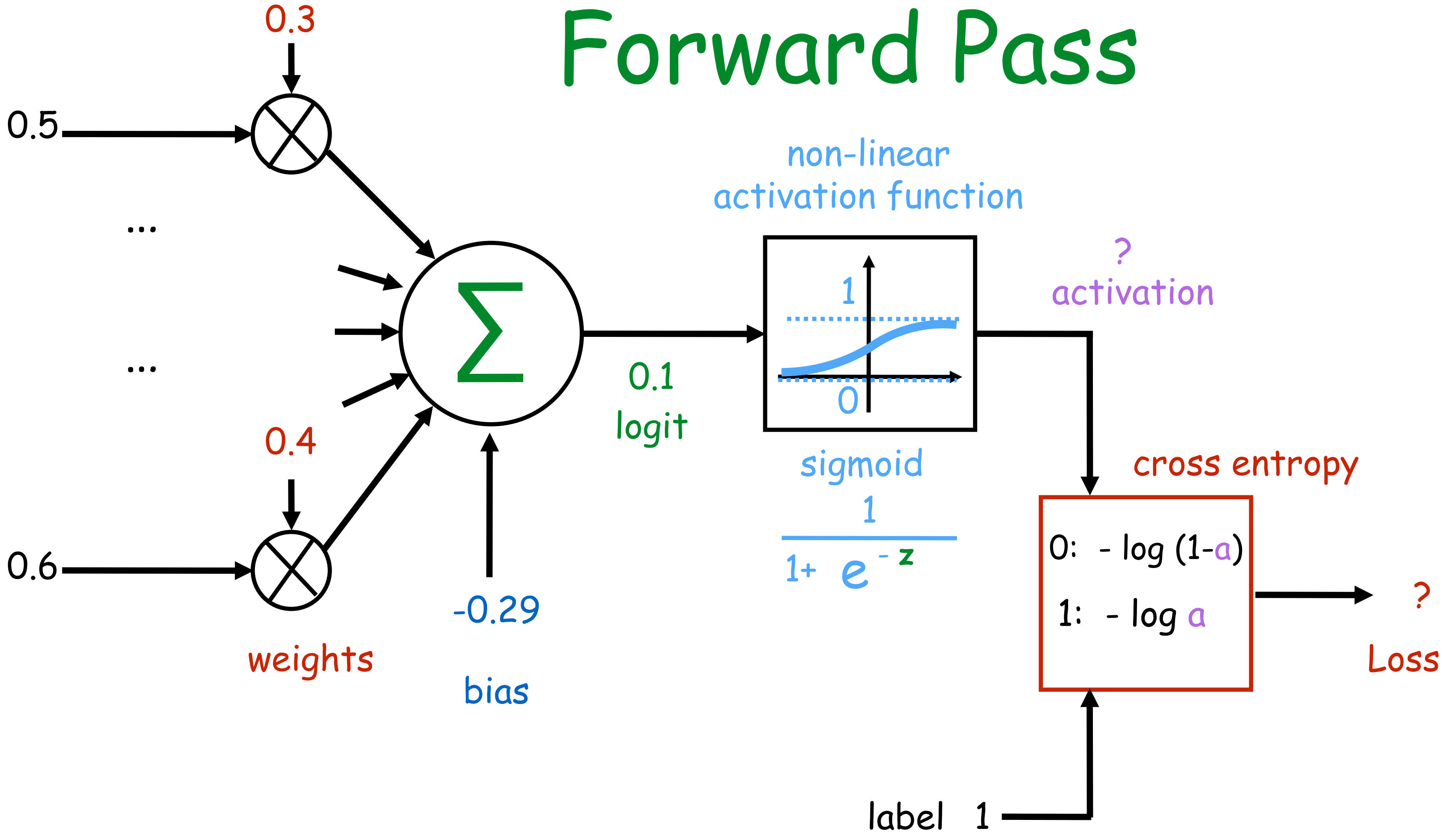


Given a training instance **x**, y

x = (0.5, …, 0.6)

y = 1

Forward Pass

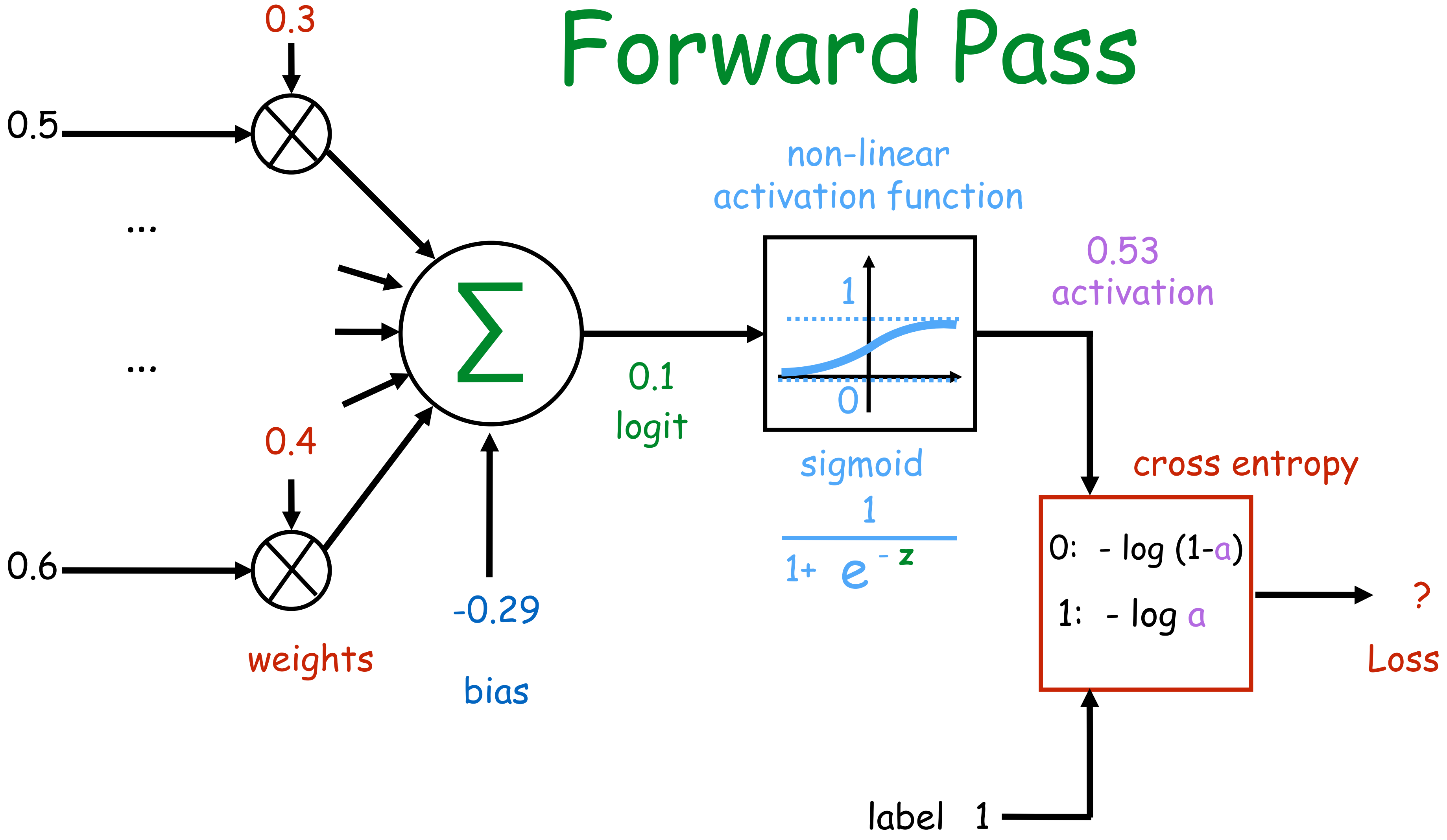


Given a training instance **x**, y

x = (0.5, ..., 0.6)

y = 1

Forward Pass

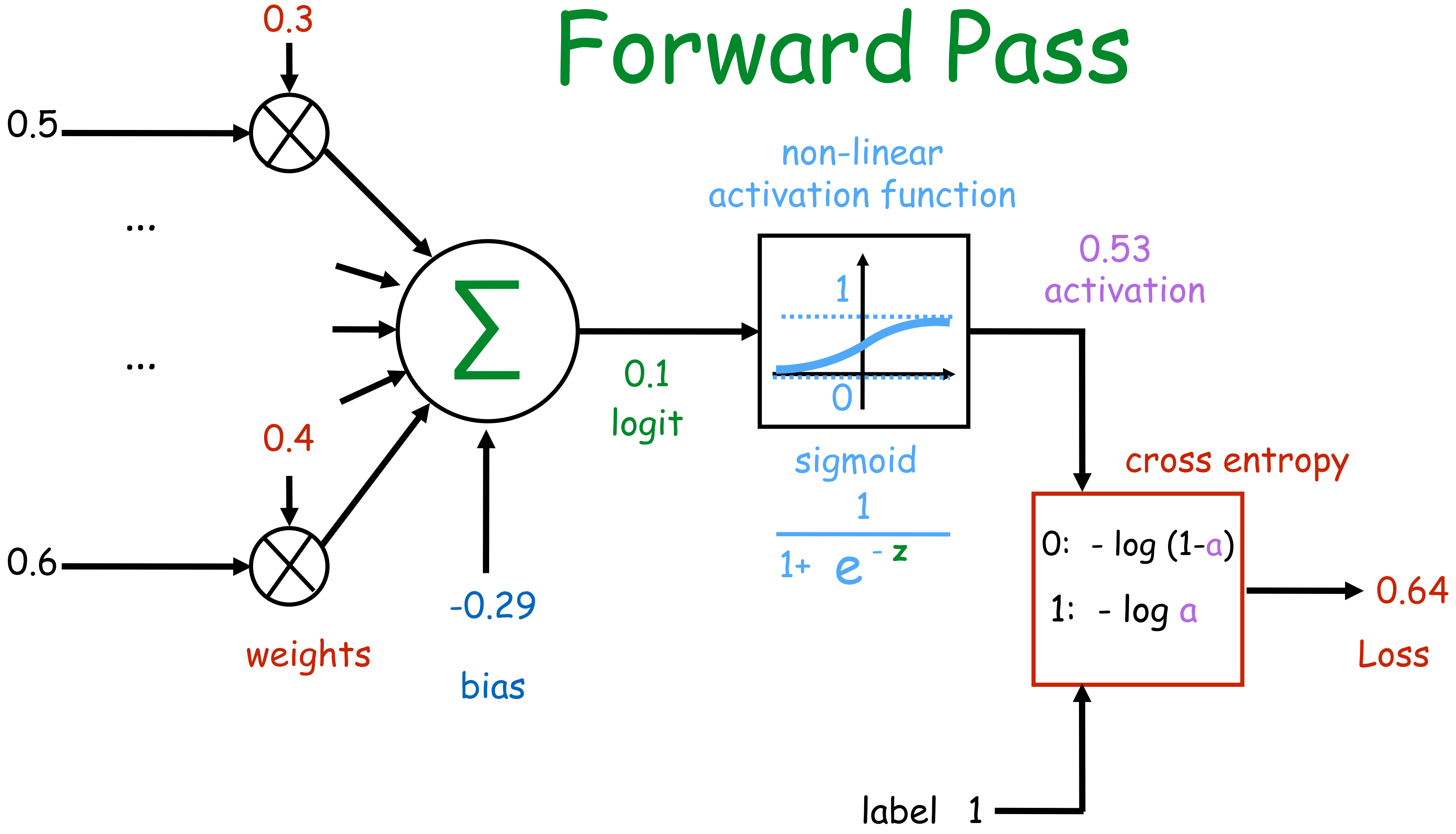


Given a training instance **x**, y

x = (0.5, …, 0.6)

y = 1

Forward Pass

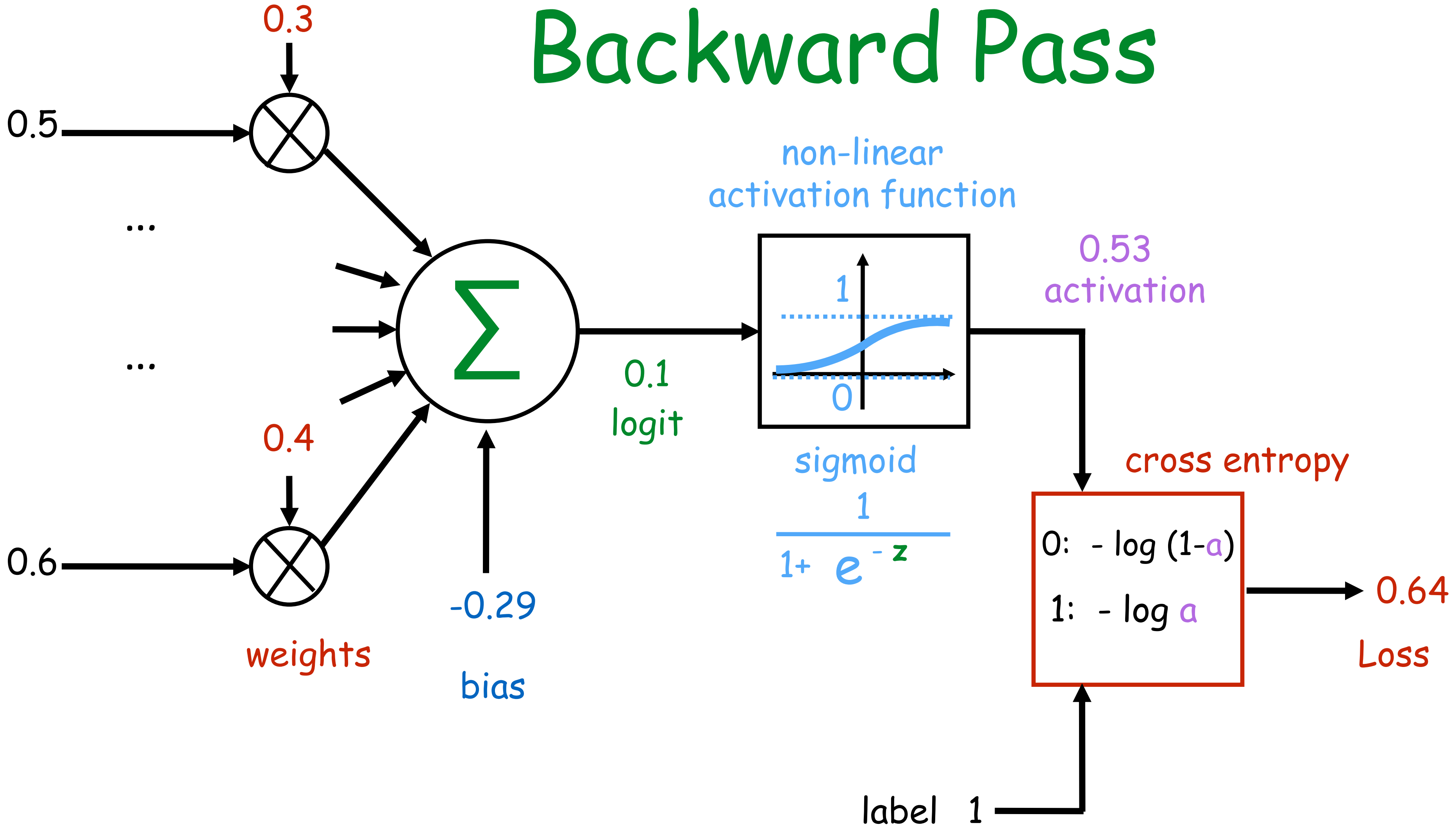


Given a training instance **x**, y

x = (0.5, ..., 0.6)

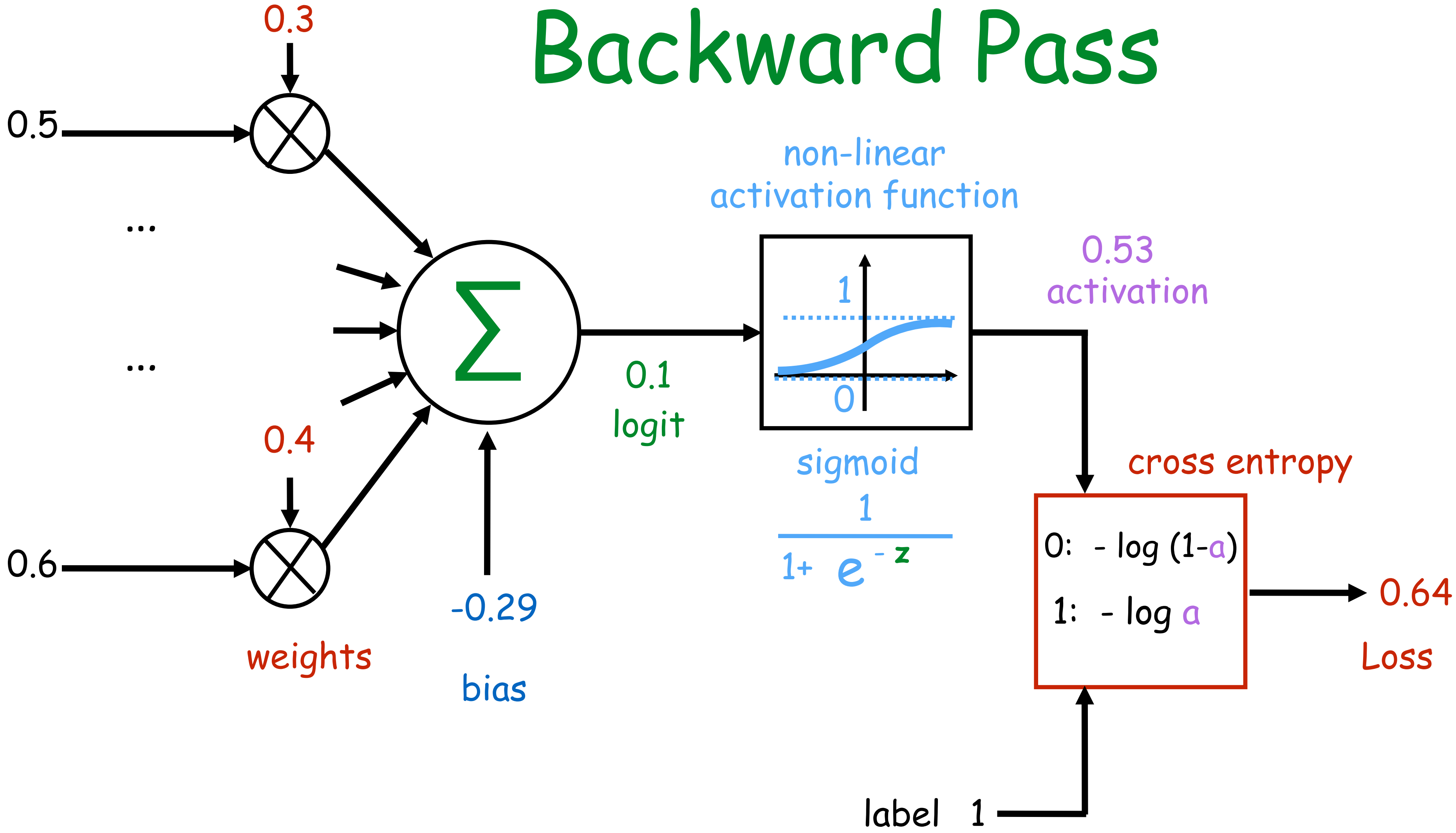
y = 1

Backward Pass



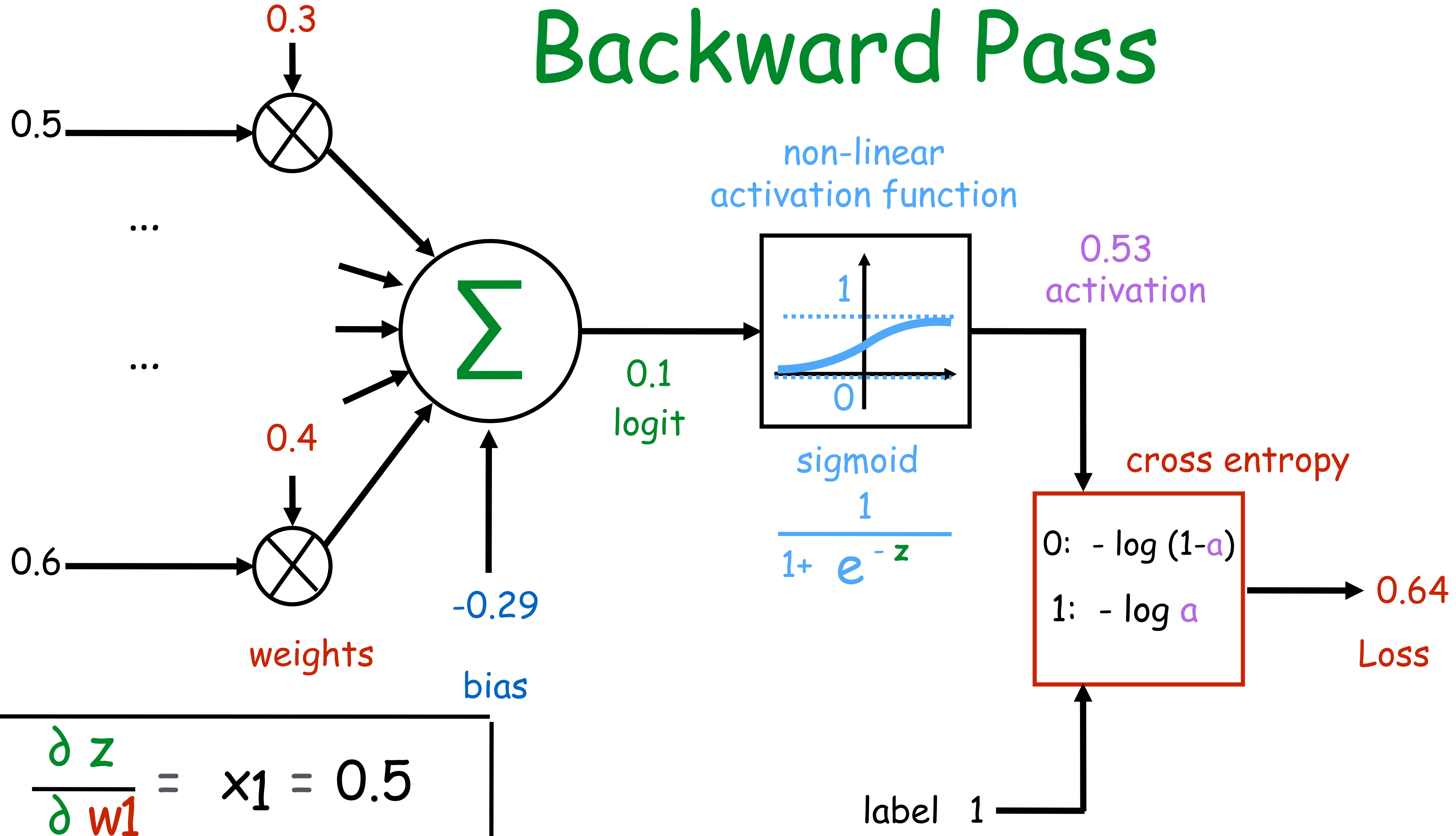
$$\frac{\partial L}{\partial a} = -1/a$$
$$= -1.9$$

Backward Pass



$$\frac{\partial a}{\partial z} = a\,(1 - a) = 0.24 \qquad \frac{\partial L}{\partial a} = -1.9$$

Backward Pass



$$\frac{\partial z}{\partial w1} = x1 = 0.5$$

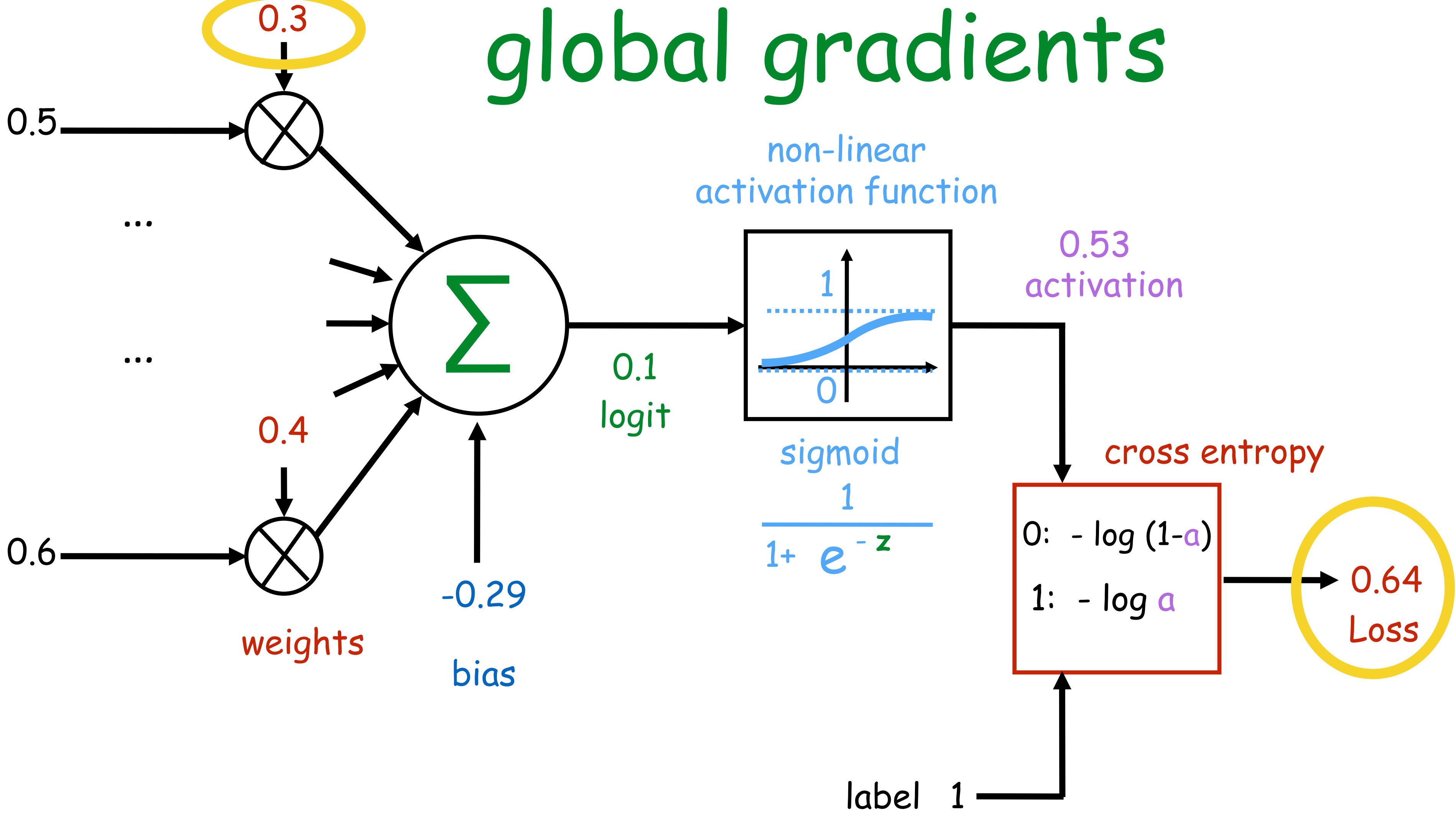
$$\frac{\partial z}{\partial wp} = xp = 0.6$$

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

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

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

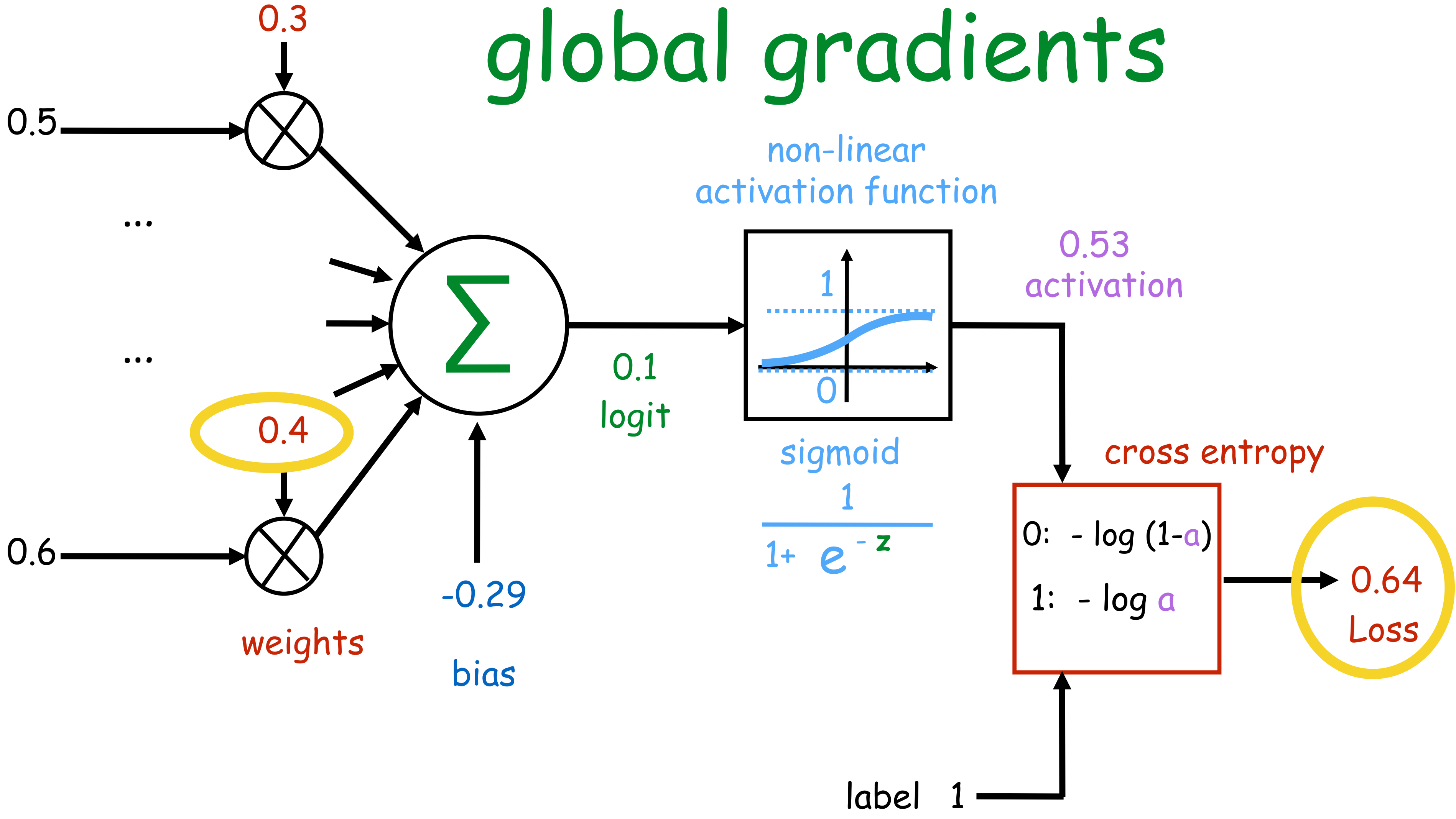
global gradients



$$\frac{\partial L}{\partial w_1} = \frac{\partial z}{\partial w1} \times \frac{\partial a}{\partial z} \times \frac{\partial L}{\partial a}$$

-0.22 0.5 0.24 -1.9

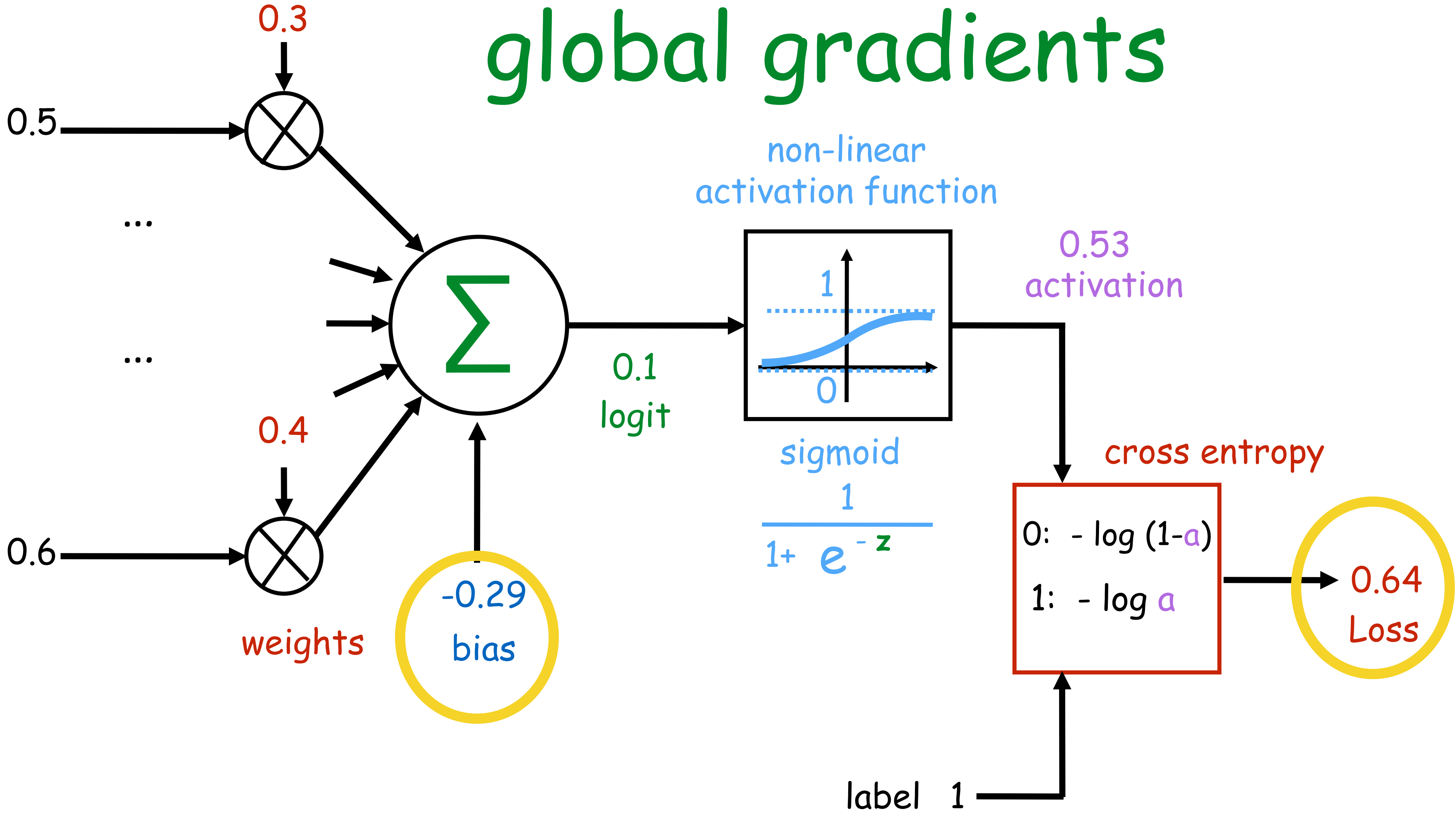
global gradients



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

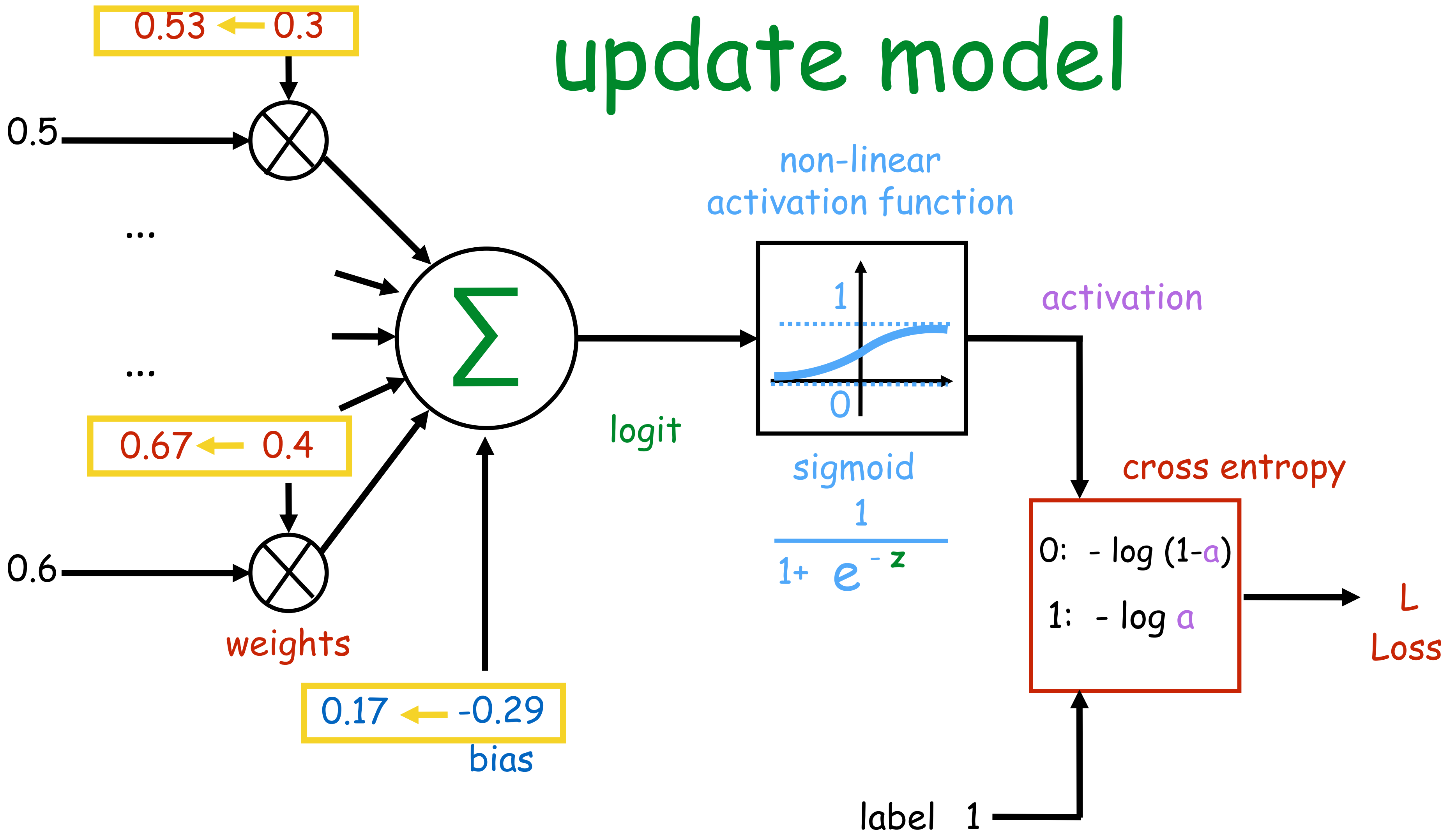
$$-0.27 \qquad 0.6 \qquad 0.24 \qquad -1.9$$

global gradients



$$\underset{-0.46}{\frac{\partial L}{\partial b}} = \underset{1}{\frac{\partial z}{\partial b}} \times \underset{0.24}{\frac{\partial a}{\partial z}} \times \underset{-1.9}{\frac{\partial L}{\partial a}}$$

update model



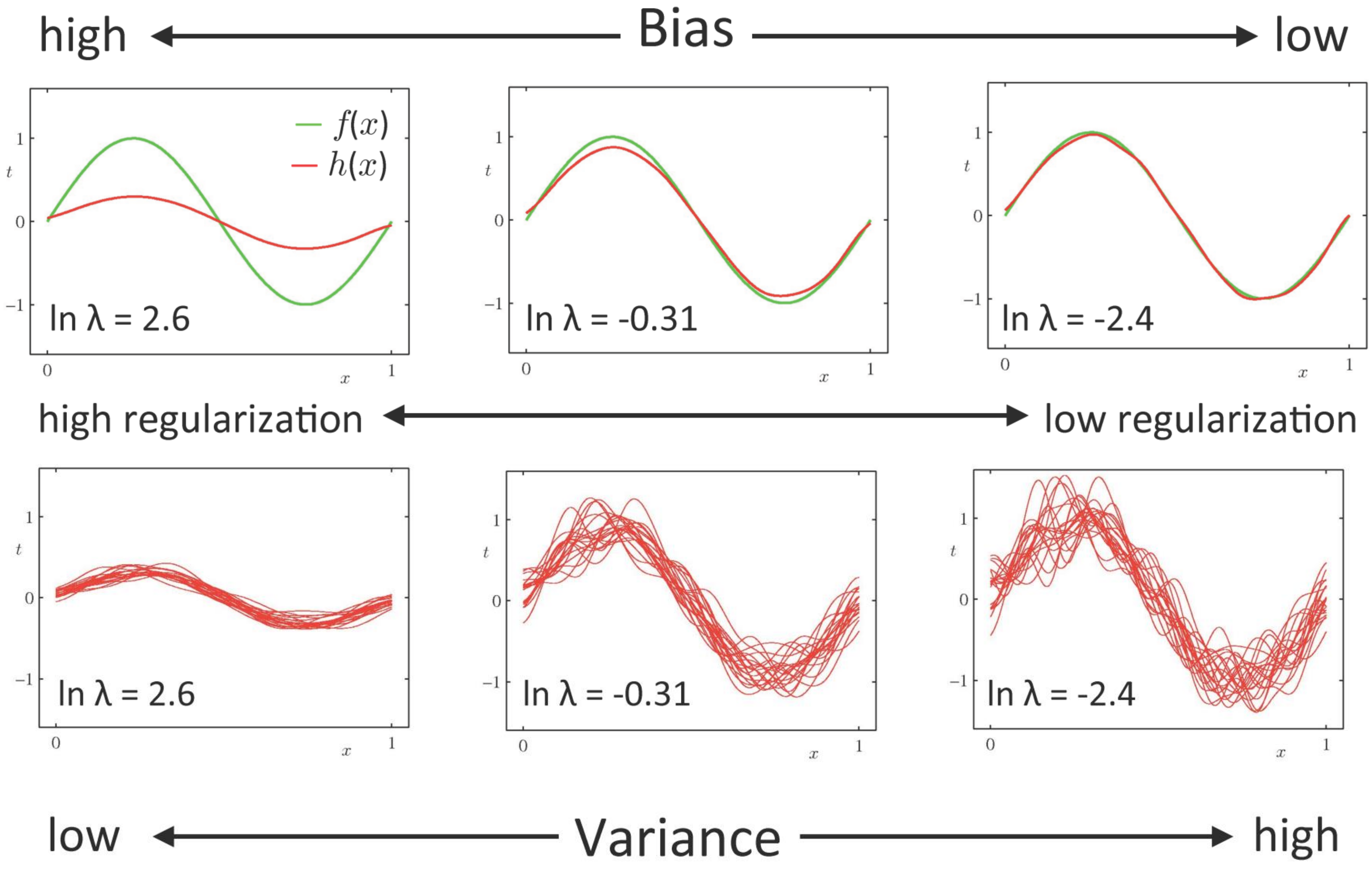
$$w \longleftarrow w - a\frac{\partial L}{\partial w}$$

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

for example

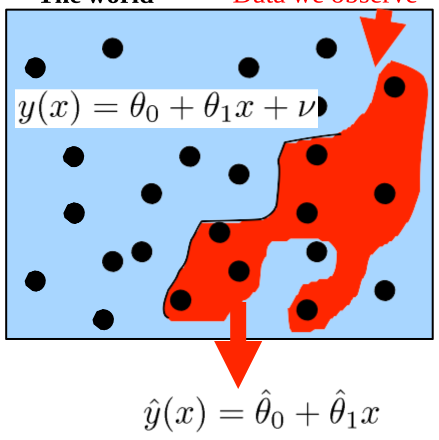
$a = 1$

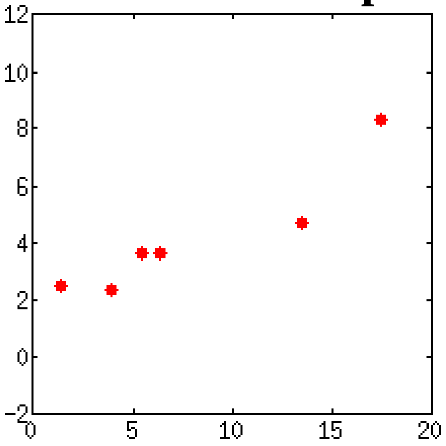
Illustration of Bias-Variance

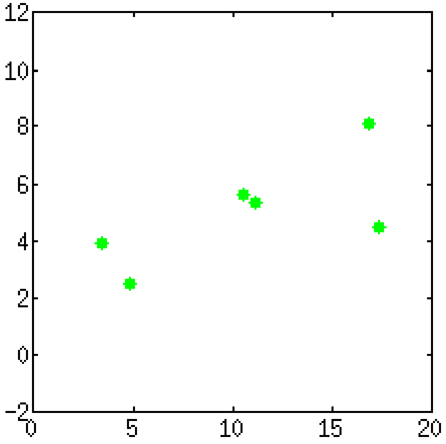


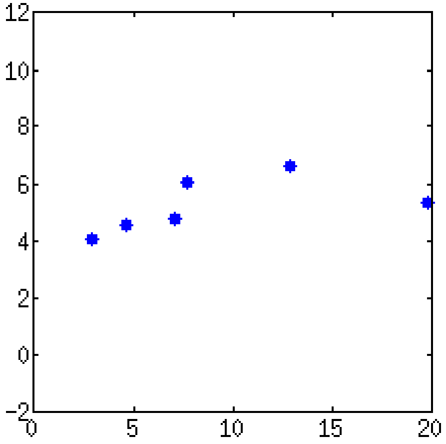
Bias & variance

Three different possible data sets:

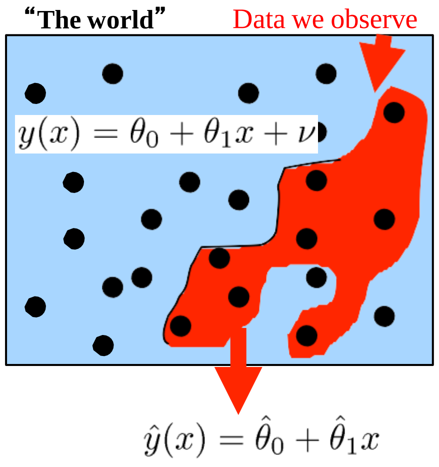




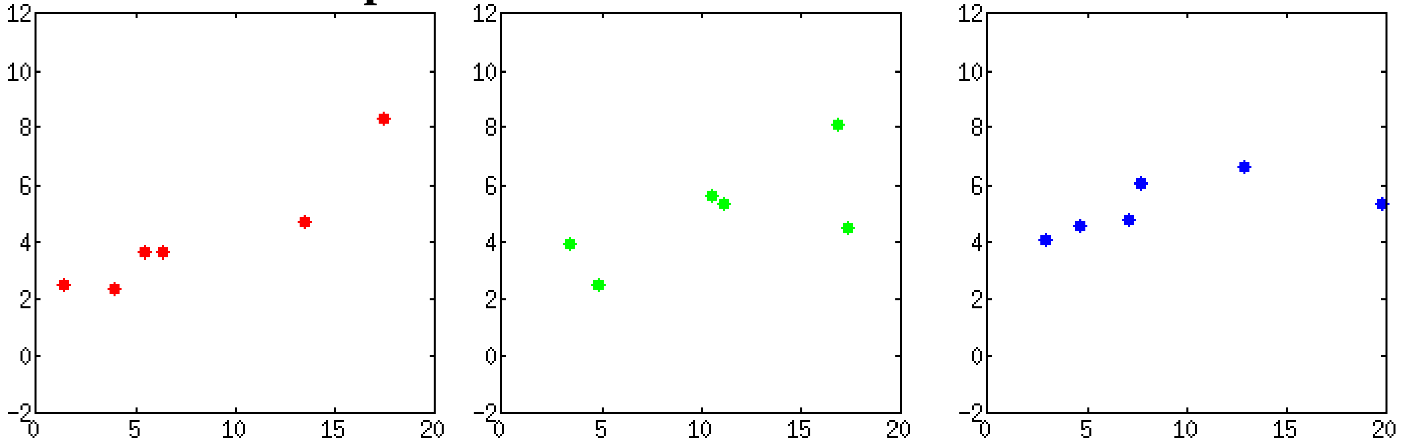




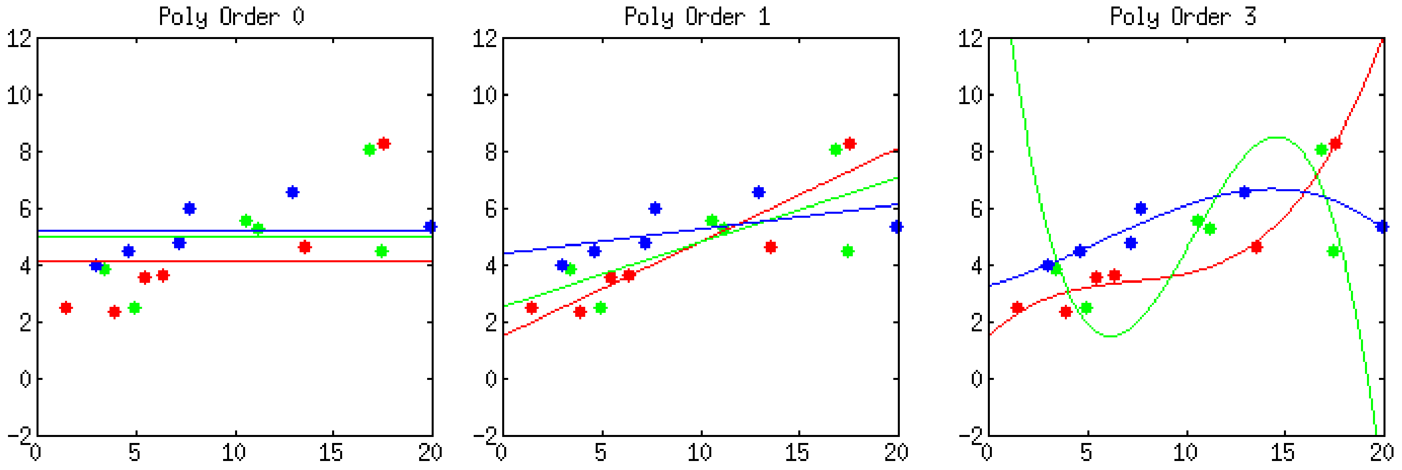
Bias & variance



Three different possible data sets:



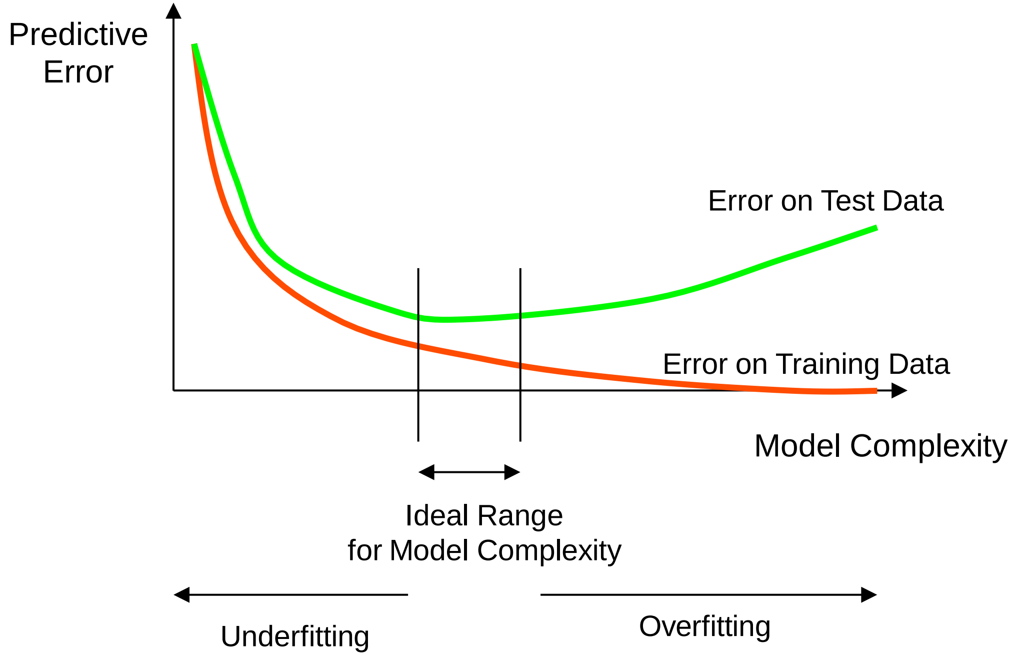
Each would give different predictors for any polynomial degree:



What to do about under/overfitting?

- Ways to increase complexity?
  - Add features, parameters

- Ways to decrease complexity?
  - Remove features (“feature selection”)
  - Regularization



Quiz2

- Quiz2 will be taken on June 18 – it will be available only during the day

- The coverage will be from linear regression (2-2 lecture) to bias & variance (4-2 lecture).

Neural Networks

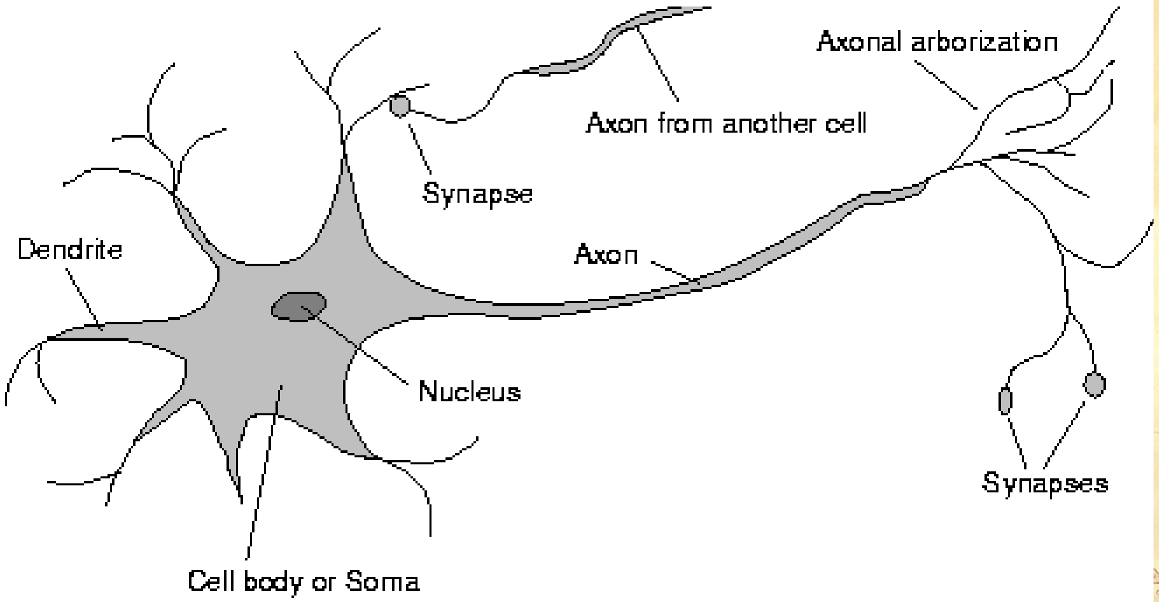
Neural Function

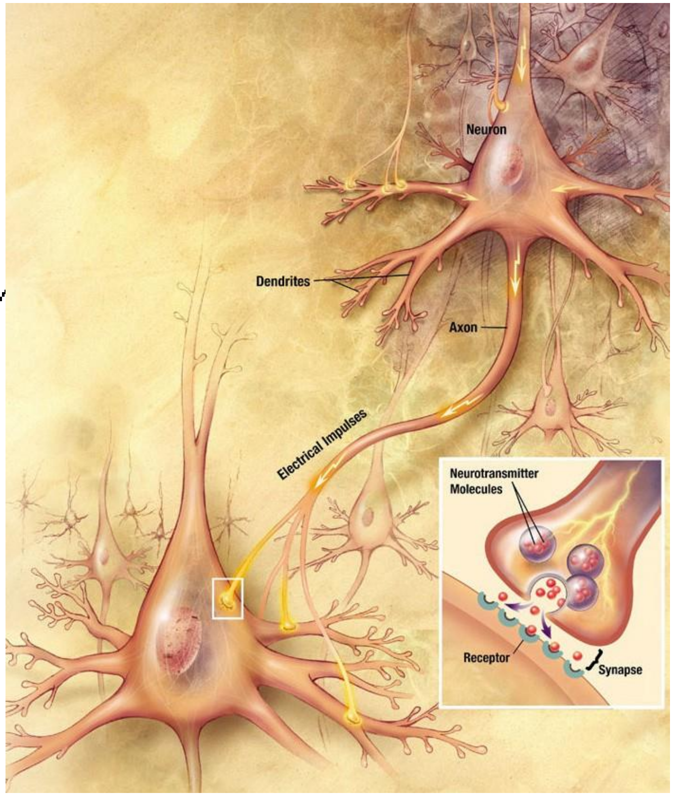
- Brain function (thought) occurs as the result of the firing of **neurons**

- Neurons connect to each other through **synapses**, which propagate **action potential** (electrical impulses) by releasing **neurotransmitters**
  - Synapses can be **excitatory** (potential-increasing) or **inhibitory** (potential-decreasing), and have varying **activation thresholds**
  - Learning occurs as a result of the synapses' **plasticicity**: They exhibit long-term changes in connection strength

- There are about 10^{11} neurons and about 10^{14} synapses in the human brain!

Biology of a Neuron

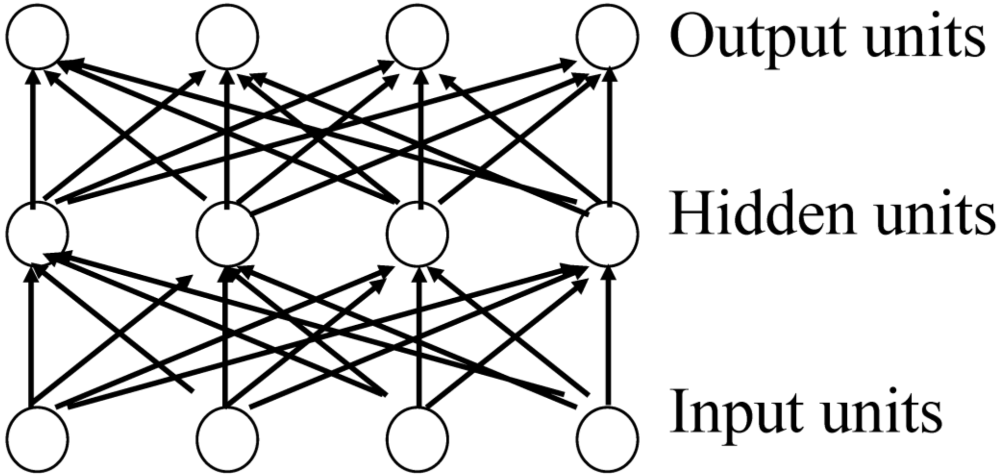




Neural Networks

- Origins: Algorithms that try to mimic the brain.
- Very widely used in 80s and early 90s; popularity diminished in late 90s.
- Recent resurgence: State-of-the-art technique for many applications
- Artificial neural networks are not nearly as complex or intricate as the actual brain structure

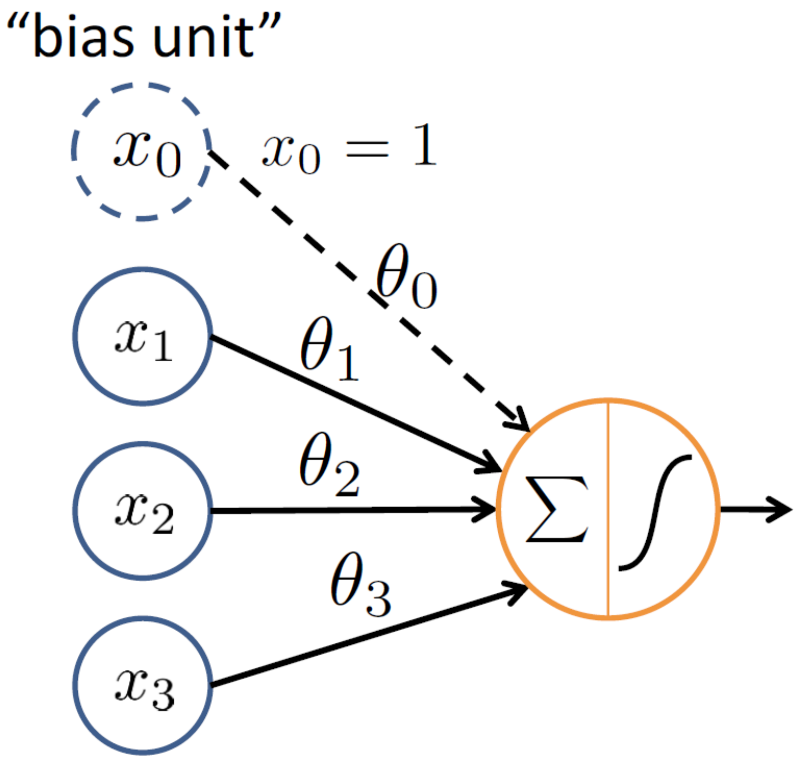
Neural Networks



Layered feed-forward network

- Neural networks are made up of **nodes** or **units**, connected by **links**
- Each link has an associated **weight** and **activation level**
- Each node has an **input function** (typically summing over weighted inputs), an **activation function**, and an **output**

Neuron Model: Logistic Unit



$$\mathbf{x} = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} \qquad \boldsymbol{\theta} = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \theta_3 \end{bmatrix}$$

$$h_{\boldsymbol{\theta}}(\mathbf{x}) = g\left(\boldsymbol{\theta}^\intercal \mathbf{x}\right) = \frac{1}{1 + e^{-\boldsymbol{\theta}^\intercal \mathbf{x}}}$$

Neural Network