CS 540 Introduction to Artificial Intelligence
Neural Networks (I): Perceptron

Yudong Chen
University of Wisconsin-Madison

Oct 19, 2021

Announcement

- HW6 released today, due Nov 4 (Thursday)

- Midterm: Oct 28 (Thursday)

Today's outline

- HW5 Review

- Recap: Bayes and Naive Bayes Classifiers

- Single-layer Neural Network (Perceptron)

Part I: Bayes and Naïve Bayes (Recap)

Bayesian classifier

$$\hat{y} = \arg\max_y p(y \mid X_1, \ldots, X_k)$$

(Posterior)

(Prediction)

Bayesian classifier

$$\hat{y} = \arg\max_y p(y \mid X_1, \ldots, X_k) \quad \text{(Posterior)}$$

(Prediction)

$$= \arg\max_y \frac{p(X_1, \ldots, X_k \mid y) \cdot p(y)}{p(X_1, \ldots, X_k)} \quad \text{(by Bayes' rule)}$$

Independent of y

Bayesian classifier

$$\hat{y} = \arg\max_y p(y \mid X_1, \ldots, X_k) \quad \text{(Posterior)}$$

(Prediction)

$$= \arg\max_y \frac{p(X_1, \ldots, X_k \mid y) \cdot p(y)}{p(X_1, \ldots, X_k)} \quad \text{(by Bayes' rule)}$$

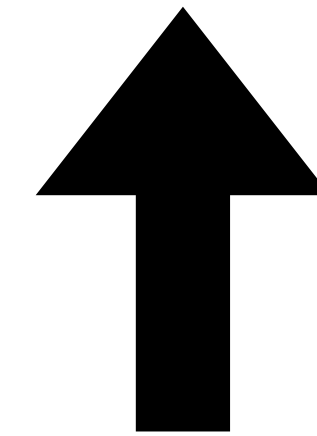$$= \arg\max_y \; p(X_1, \ldots, X_k \mid y) \; p(y)$$

Class conditional likelihood

Class prior

Naïve Bayes Assumption

Conditional independence of feature attributes

$$p(X_1, \ldots, X_k \mid y)p(y) = \Pi_{i=1}^{k} p(X_i \mid y)p(y)$$

Easier to estimate

(using MLE!)

Example 1: Play outside or not?

- If weather is sunny, would you likely to play outside?

Posterior probability p(Yes | ☀) vs. p(No | ☀)

- Weather = {Sunny, Rainy, Overcast}

- Play = {Yes, No}

- Observed data {Weather, play on day m}, m={1,2,…,N}

$$p(\text{Play} \mid ☀) = \frac{p(☀ \mid \text{Play})\, p(\text{Play})}{p(☀)}$$
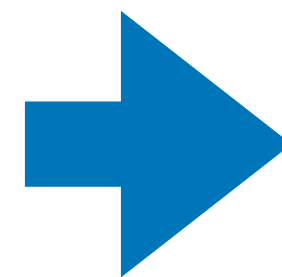
Bayes rule
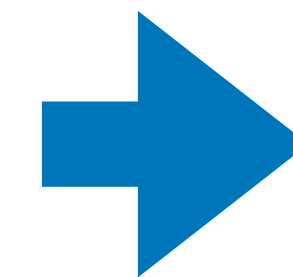
Example 1: Play outside or not?

Step 1: Convert the data to a frequency table of Weather and Play

Step 2: Based on the frequency table, calculate **likelihoods** and **priors**

Weather	Play
Sunny	No
Overcast	Yes
Rainy	Yes
Sunny	Yes
Sunny	Yes
Overcast	Yes
Rainy	No
Rainy	No
Sunny	Yes
Rainy	Yes
Sunny	No
Overcast	Yes
Overcast	Yes
Rainy	No

Frequency Table		
Weather	No	Yes
Overcast		4
Rainy	3	2
Sunny	2	3
Grand Total	5	9

Likelihood table				
Weather	No	Yes		
Overcast		4	=4/14	0.29
Rainy	3	2	=5/14	0.36
Sunny	2	3	=5/14	0.36
All	5	9		
	=5/14	=9/14		
	0.36	0.64		

p(Play = Yes) = 0.64

p(☀️ | Yes) = 3/9 = 0.33

Example 1: Play outside or not?

Step 3: Based on the likelihoods and priors, calculate posteriors

P(Yes| ☀)
=P(☀ |Yes) * P(Yes) / P(☀)
=0.33 * 0.64 / 0.36
=0.6

P(No| ☀)
=P(☀ |No) * P(No) / P(☀)
=0.4 * 0.36 / 0.36
=0.4

P(Yes| ☀) > P(No| ☀)   go outside and play!

Quiz break

Q1-3: Consider the following dataset showing the result whether a person has passed or failed the exam based on various factors. Suppose the factors are independent to each other. We want to classify a new instance with Confident=Yes, Studied=Yes, and Sick=No.

Confident	Studied	Sick	Result
Yes	No	No	Fail
Yes	No	Yes	Pass
No	Yes	Yes	Fail
No	Yes	No	Pass
Yes	Yes	Yes	Pass

- A Pass

- B Fail

Part I: Single-layer Neural Network

How to classify
Cats vs. dogs?

Inspiration from neuroscience

- Inspirations from human brains
- Networks of simple and homogenous units

(wikipedia)

Perceptron

Cats vs. dogs?

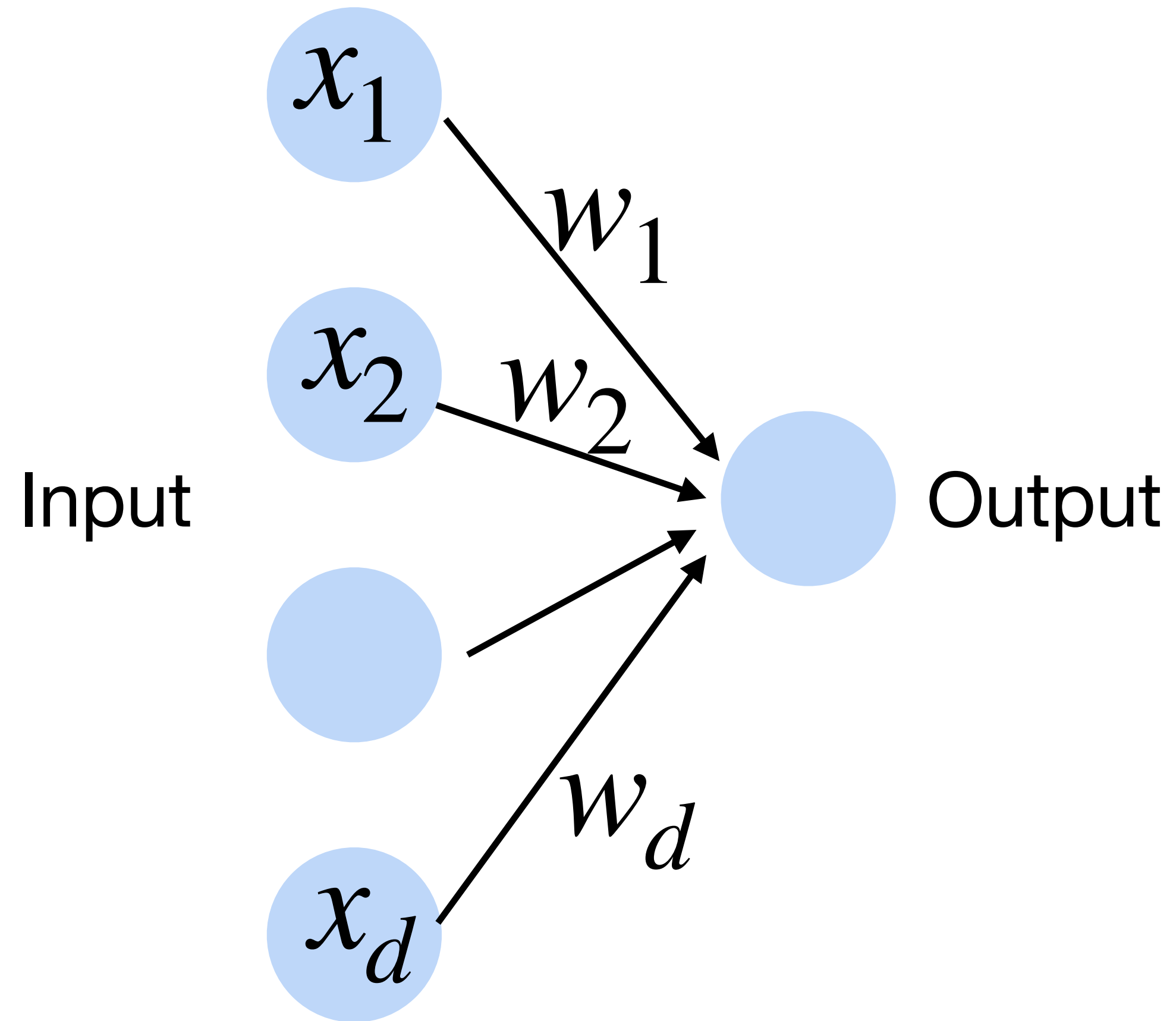Input

Output

x_1

x_2
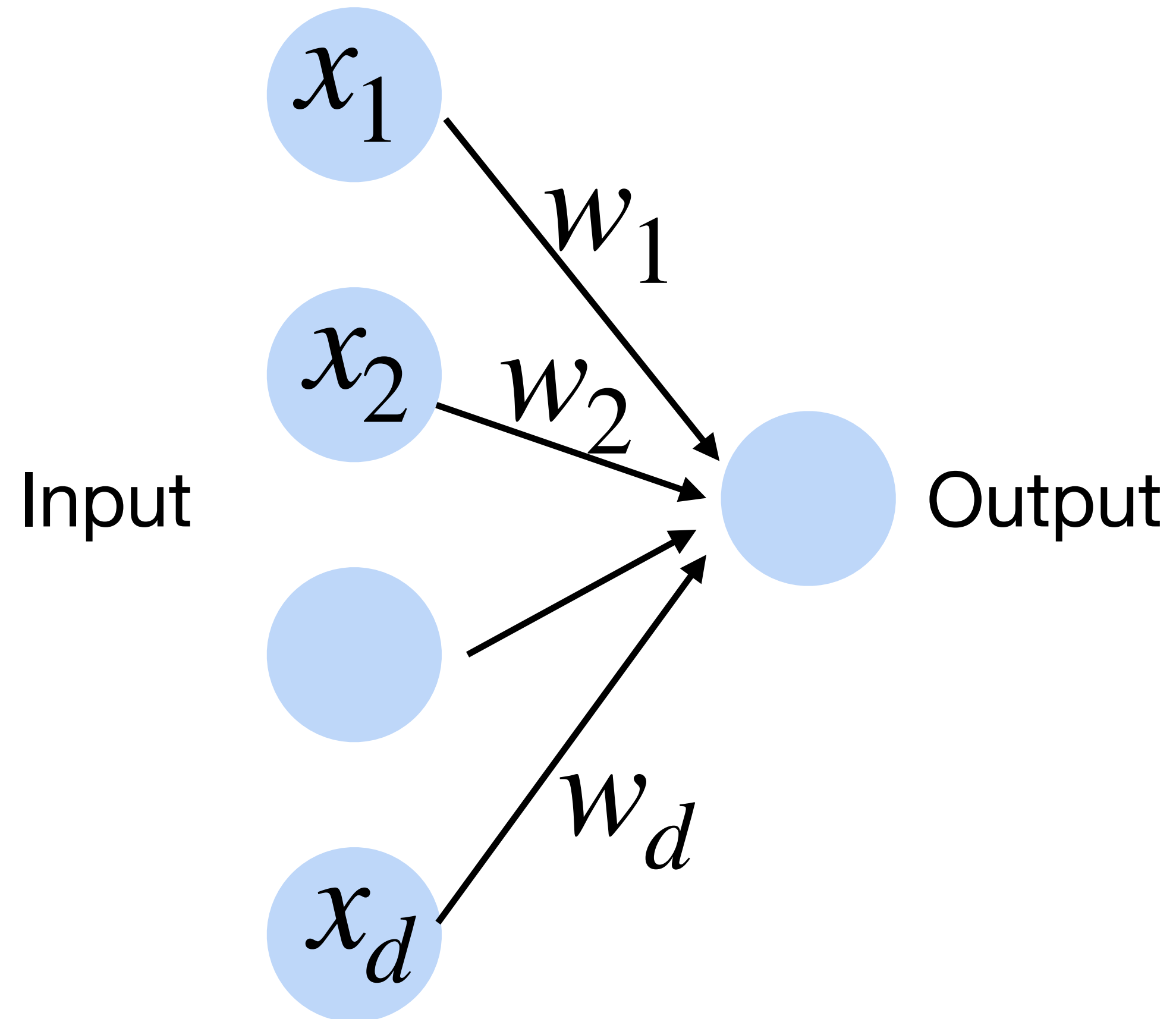
x_d

w_1

w_2

w_d
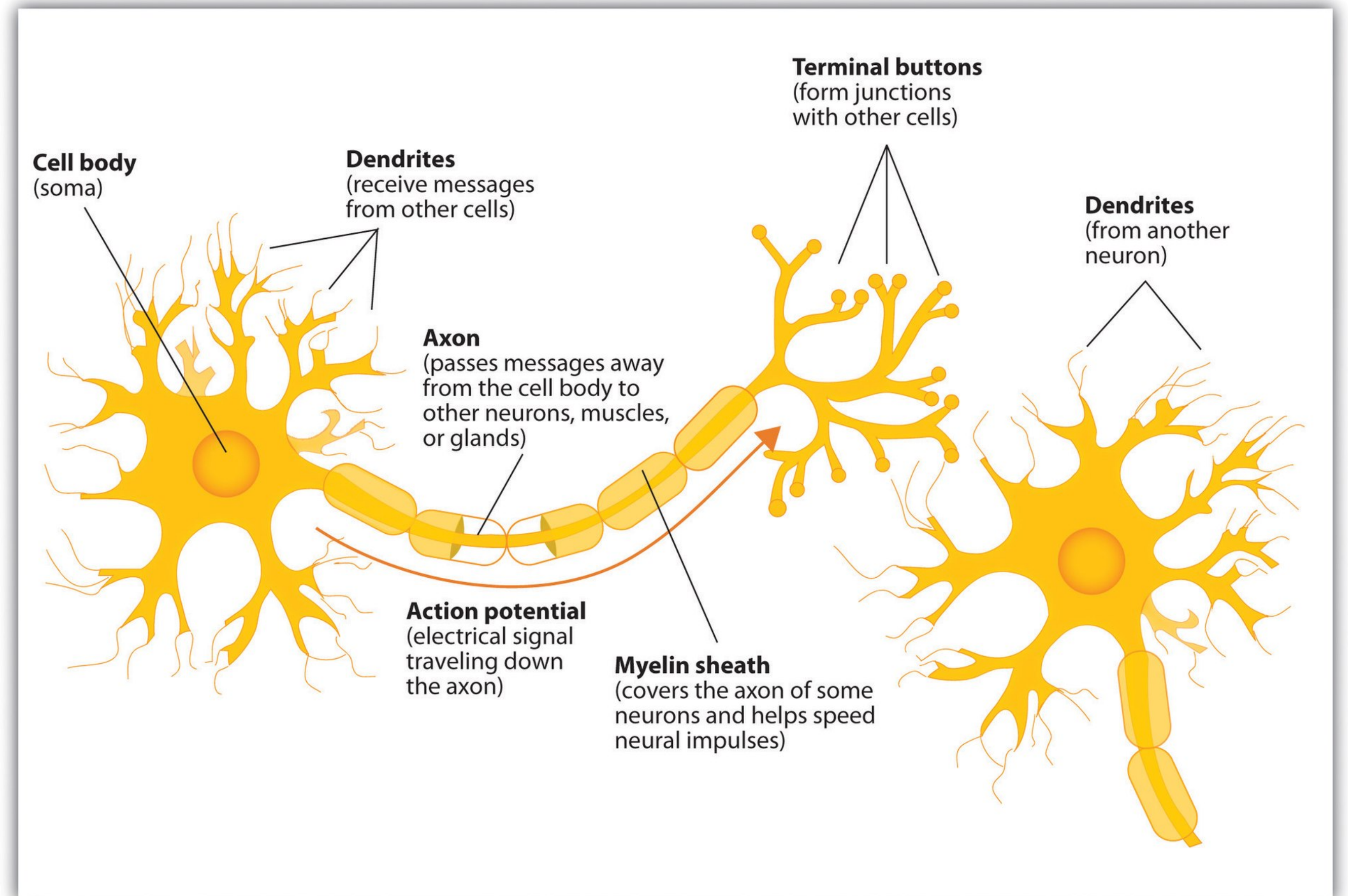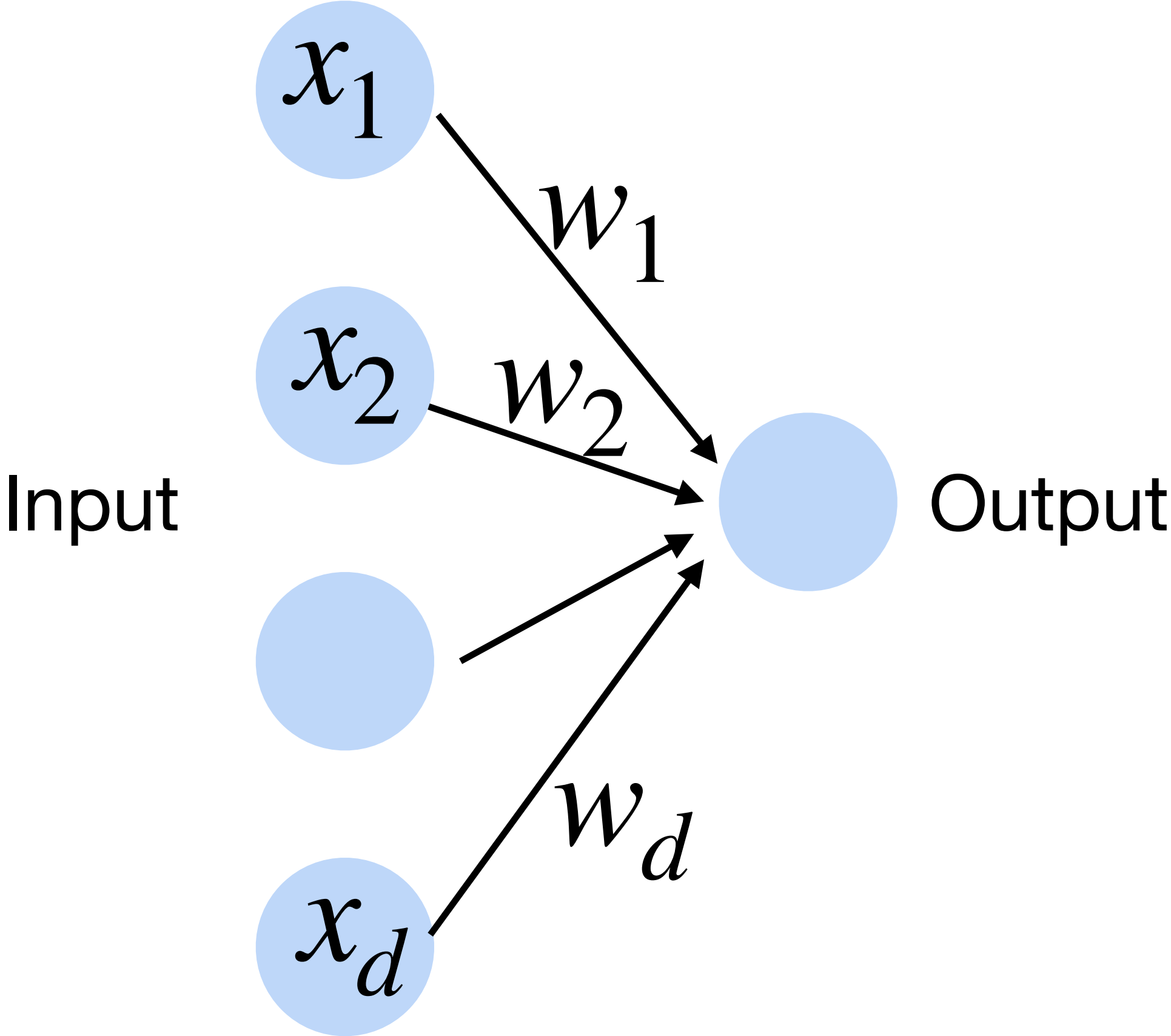
Linear Perceptron (=linear regression)

- Given input $\mathbf{x}$, weight $\mathbf{w}$ and bias b, perceptron outputs:

$$f_{\mathbf{w},b}(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle + b$$

Cats vs. dogs?

Perceptron

- Given input $\mathbf{x}$, weight $\mathbf{w}$ and bias b, perceptron outputs:

$$f_{\mathbf{w},b}(\mathbf{x}) = \sigma\left(\langle \mathbf{w}, \mathbf{x}\rangle + b\right)$$

$$\sigma(z) = \begin{cases} 1 & \text{if } z > 0 \\ 0 & \text{otherwise} \end{cases}$$ **Activation function**

Cats vs. dogs?

x_1

w_1

x_2

w_2

Input

x_d

w_d

Output

Perceptron

- Goal: learn parameters $\mathbf{w} = \{w_1, w_2, \ldots, w_d\}$ and b to minimize the classification error

Cats vs. dogs?

Input

Output

x_1

w_1

x_2

w_2

x_d

w_d
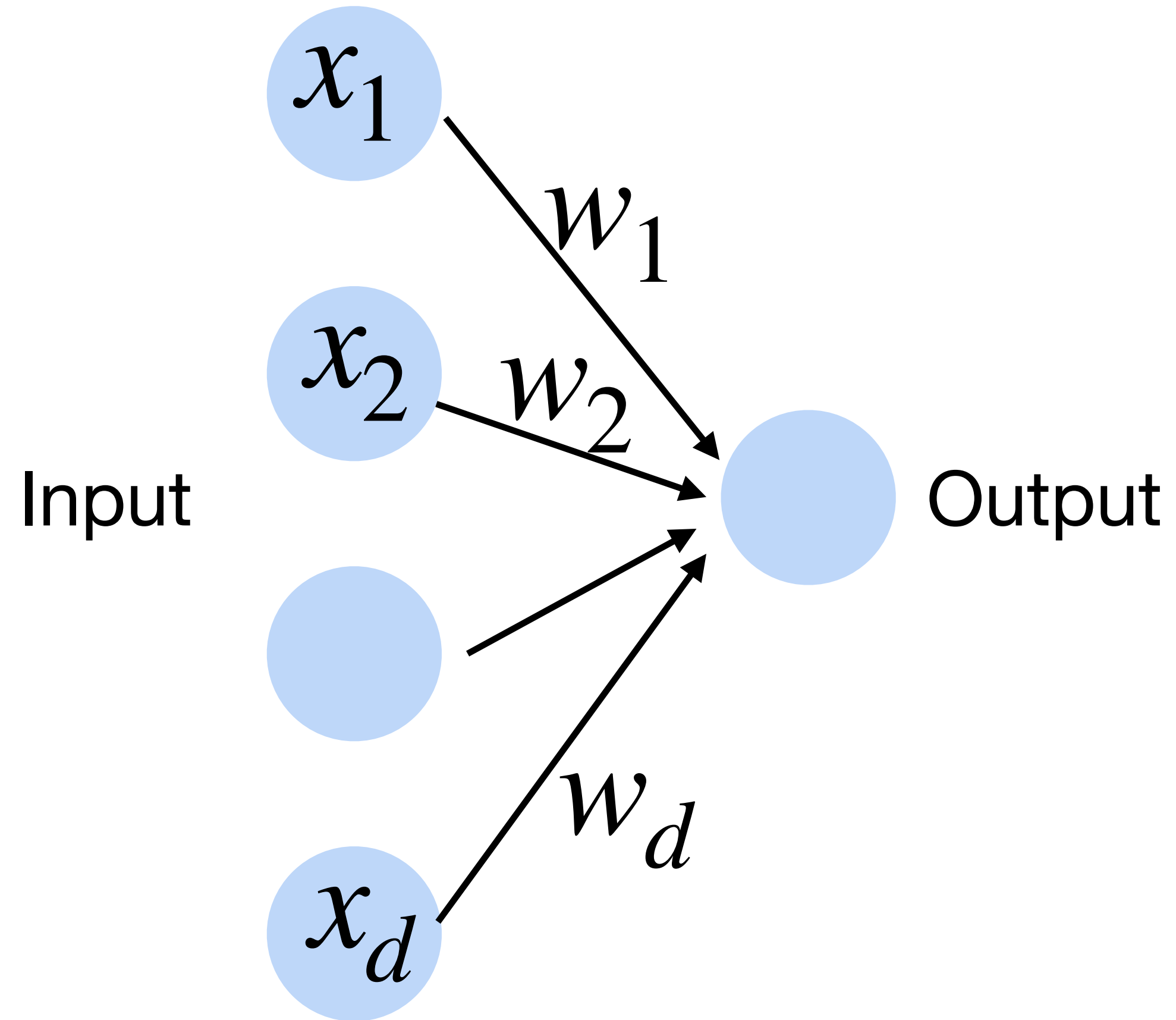
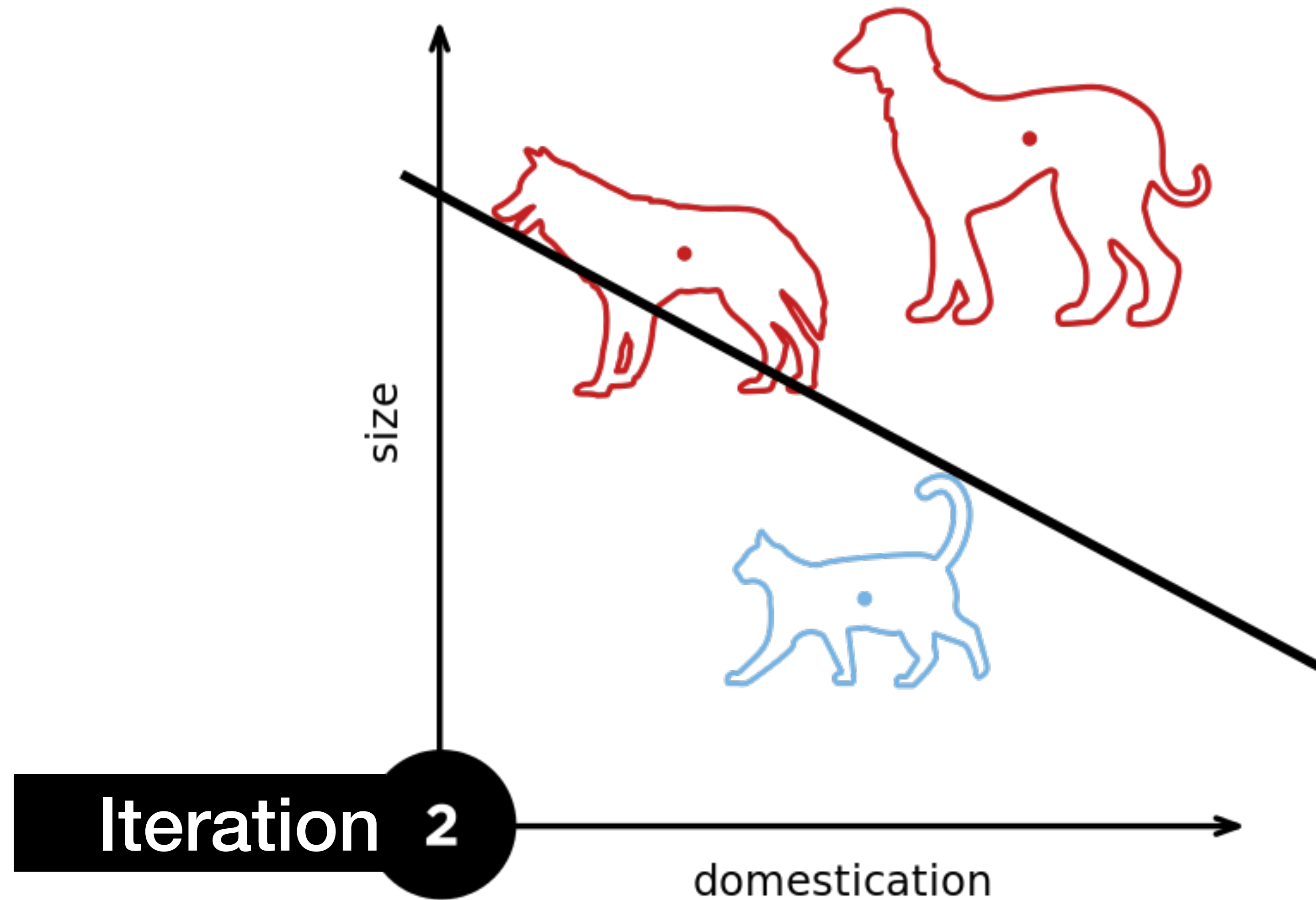# Training the Perceptron

Perceptron Algorithm

Initialize $\vec{w} = \vec{0}$ // Initialize $\vec{w}$. $\vec{w} = \vec{0}$ misclassifies everything.

while TRUE **do**   // Keep looping

    $m = 0$ // Count the number of misclassifications, m

    for $(x_i, y_i) \in D$ **do**   // Loop over each (data, label) pair in the dataset,

        if $y_i(\vec{w}^T \cdot \vec{x}_i) \leq 0$ **then**   // If the pair $(\vec{x}_i, y_i)$ is misclassified

            $\vec{w} \leftarrow \vec{w} + y\vec{x}$ // Update the weight vector $\vec{w}$

            $m \leftarrow m + 1$ // Counter the number of misclassification

        end if

    end for

    if $m = 0$ **then**   // If the most recent $\vec{w}$ gave 0 misclassifications

        break   // Break out of the while-loop

    end if

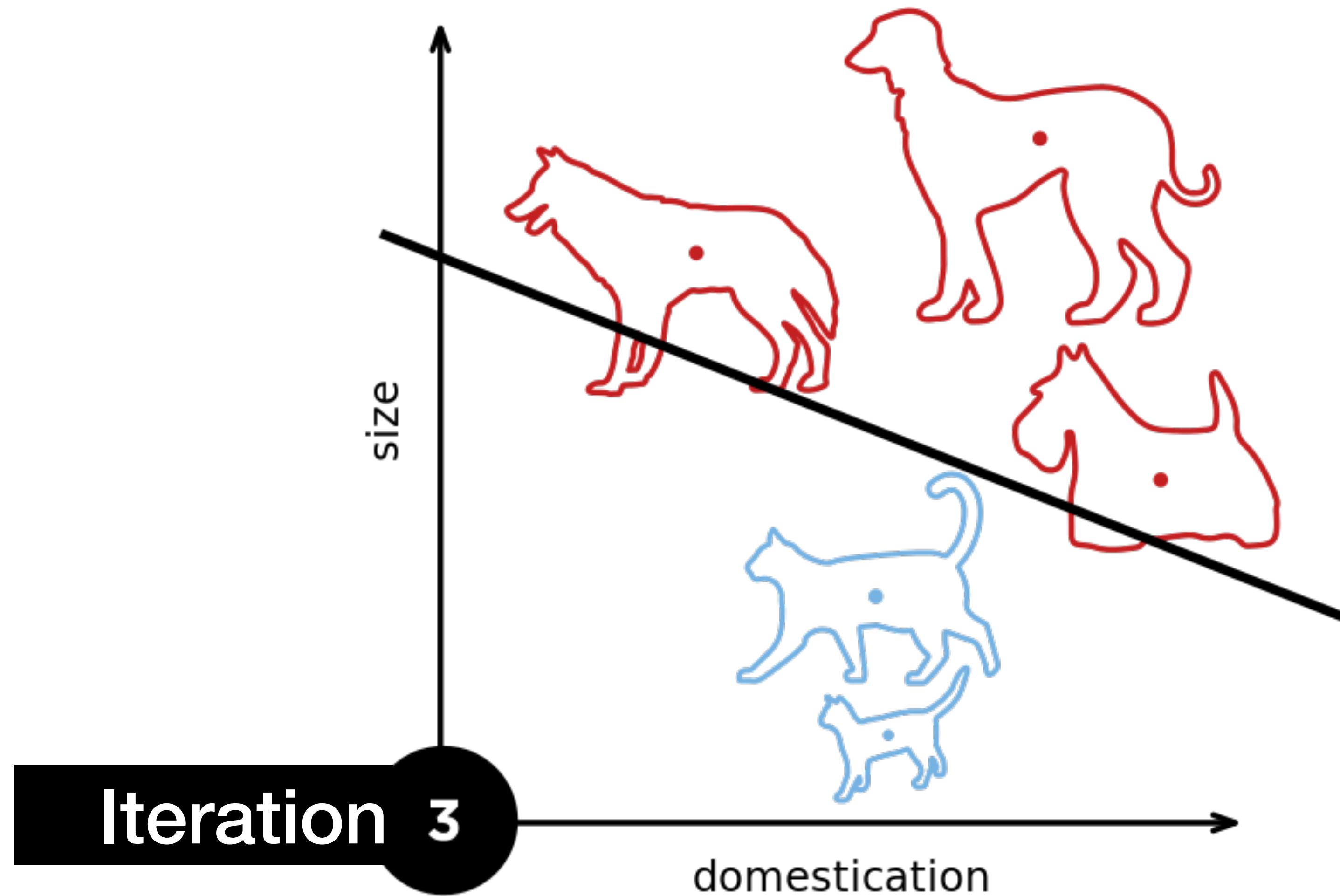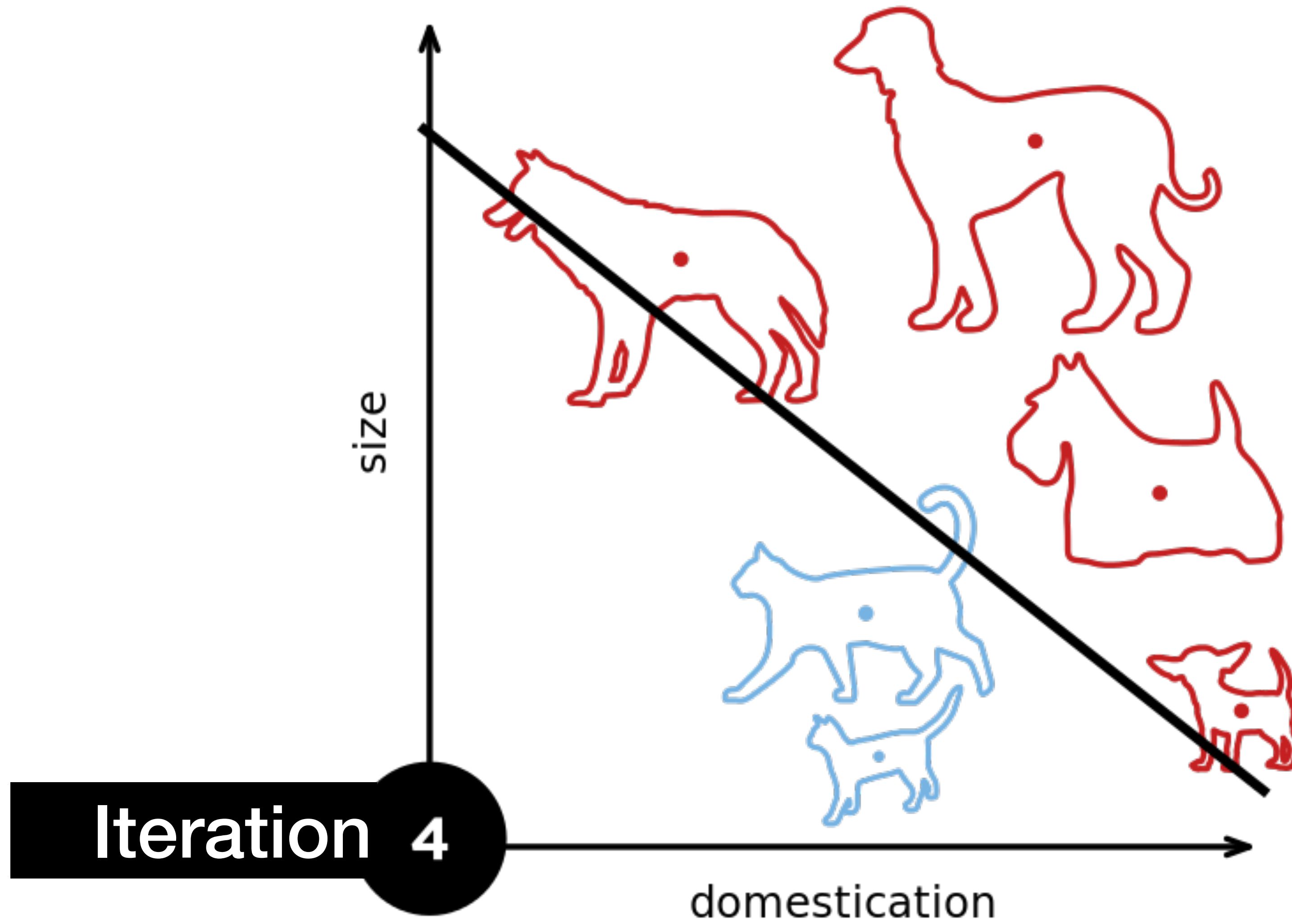**end while**   // Otherwise, keep looping!
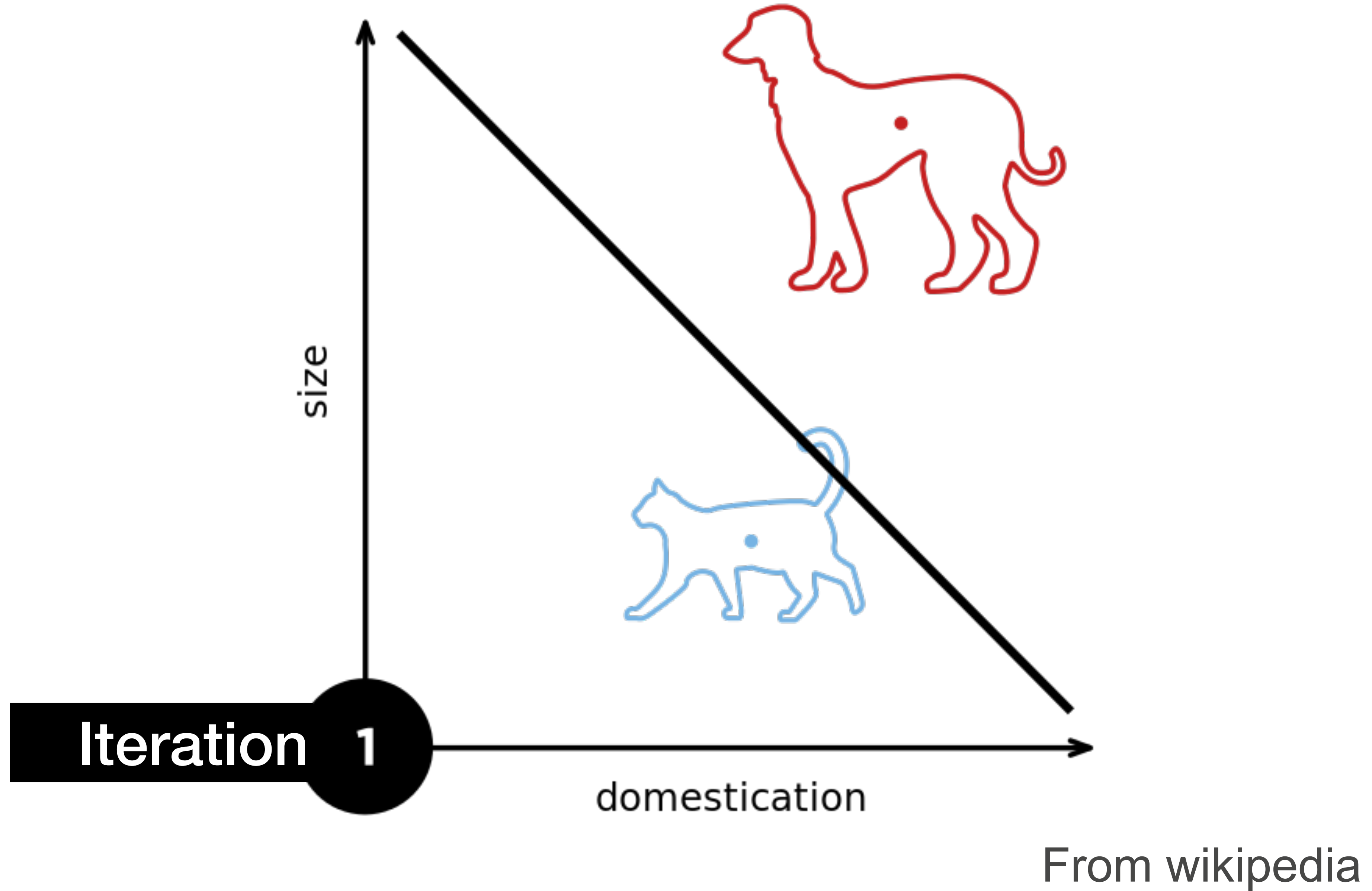
Perceptron

From wikipedia

Perceptron

Iteration 2

size

domestication

From wikipedia

Perceptron

size

domestication

Iteration **3**

From wikipedia

Perceptron

size

domestication

Iteration 4
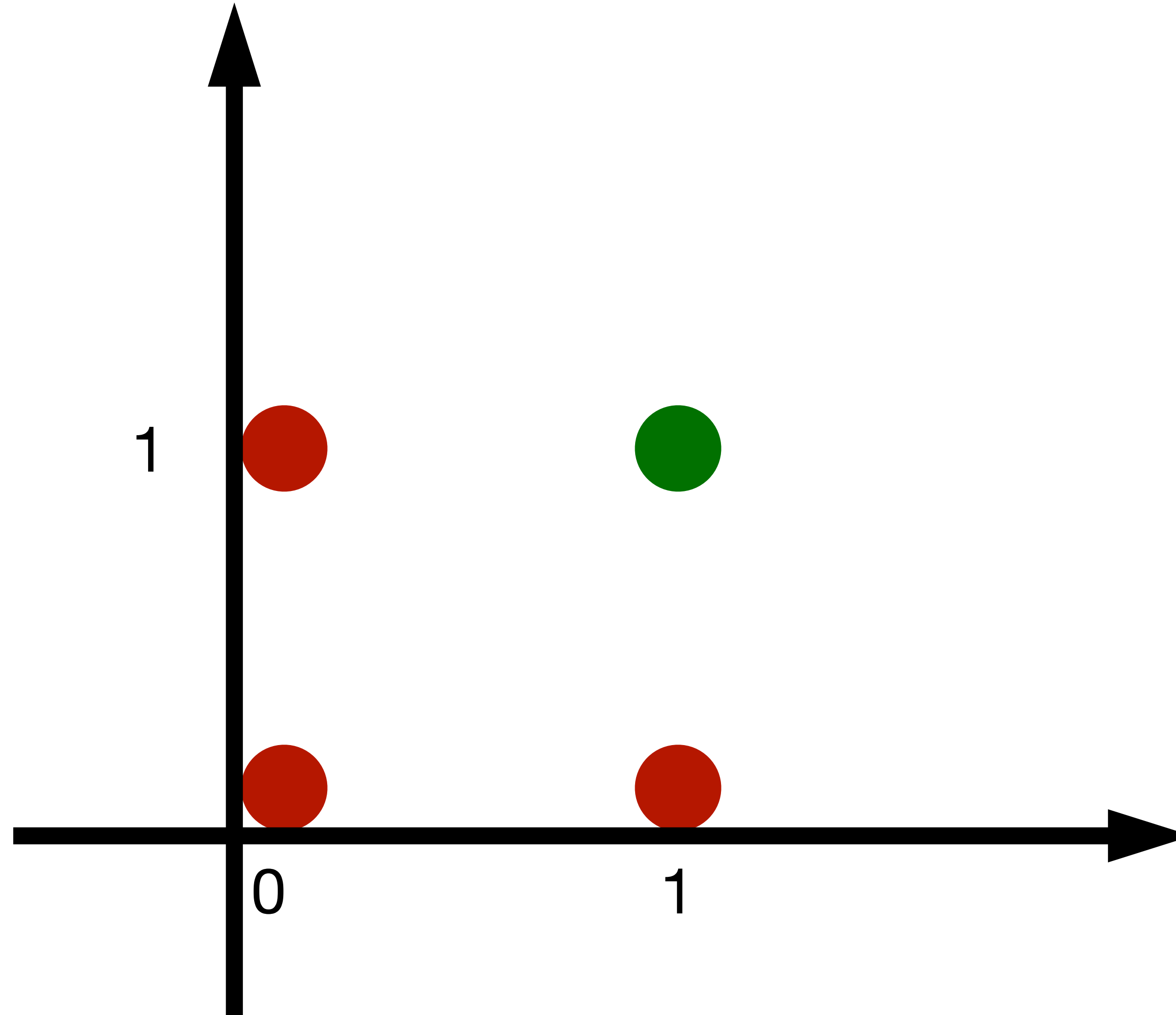
From wikipedia

Learning AND function using perceptron

The perceptron can learn an AND function

$x_1 = 1, x_2 = 1, y = 1$
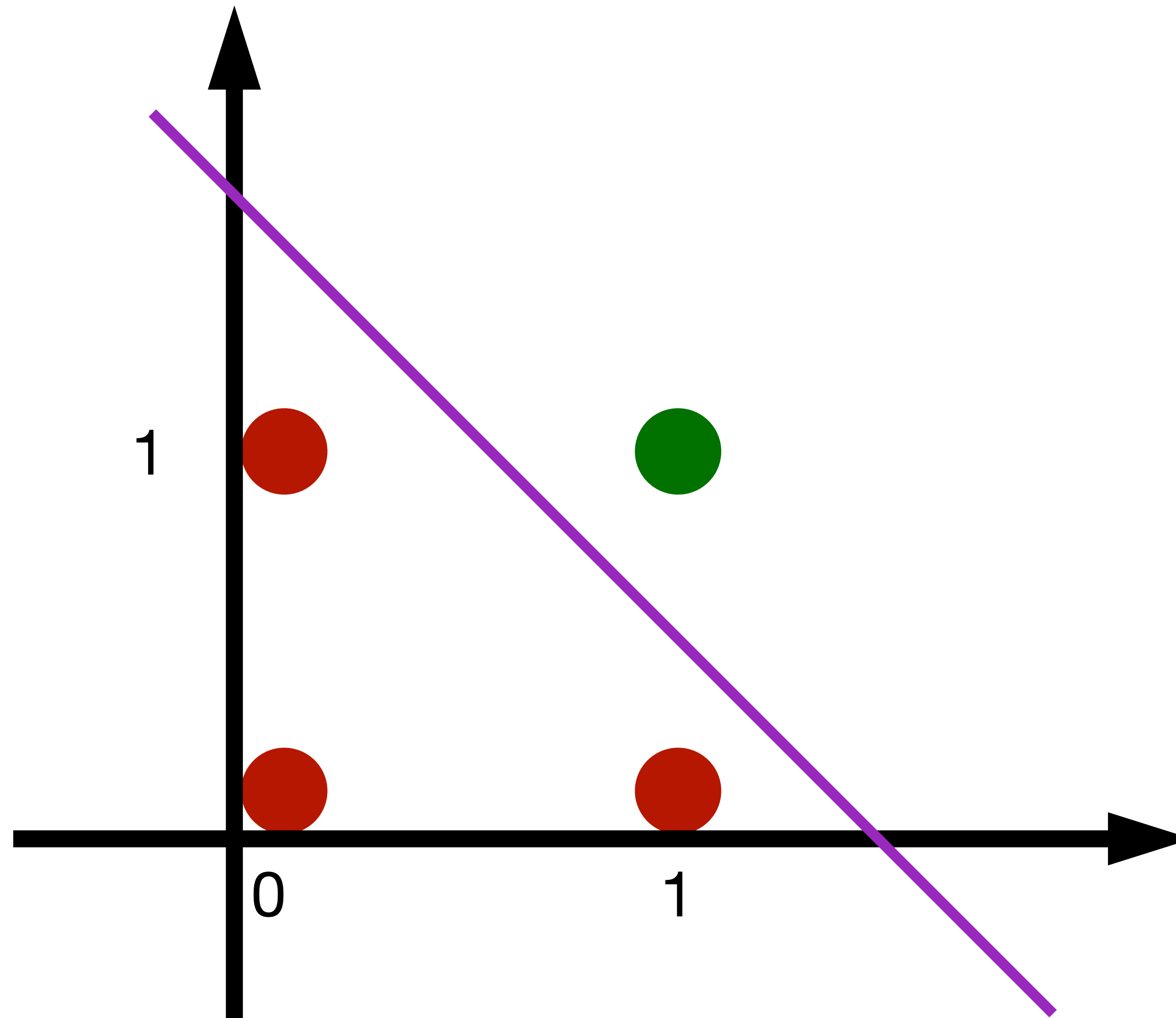
$x_1 = 1, x_2 = 0, y = 0$
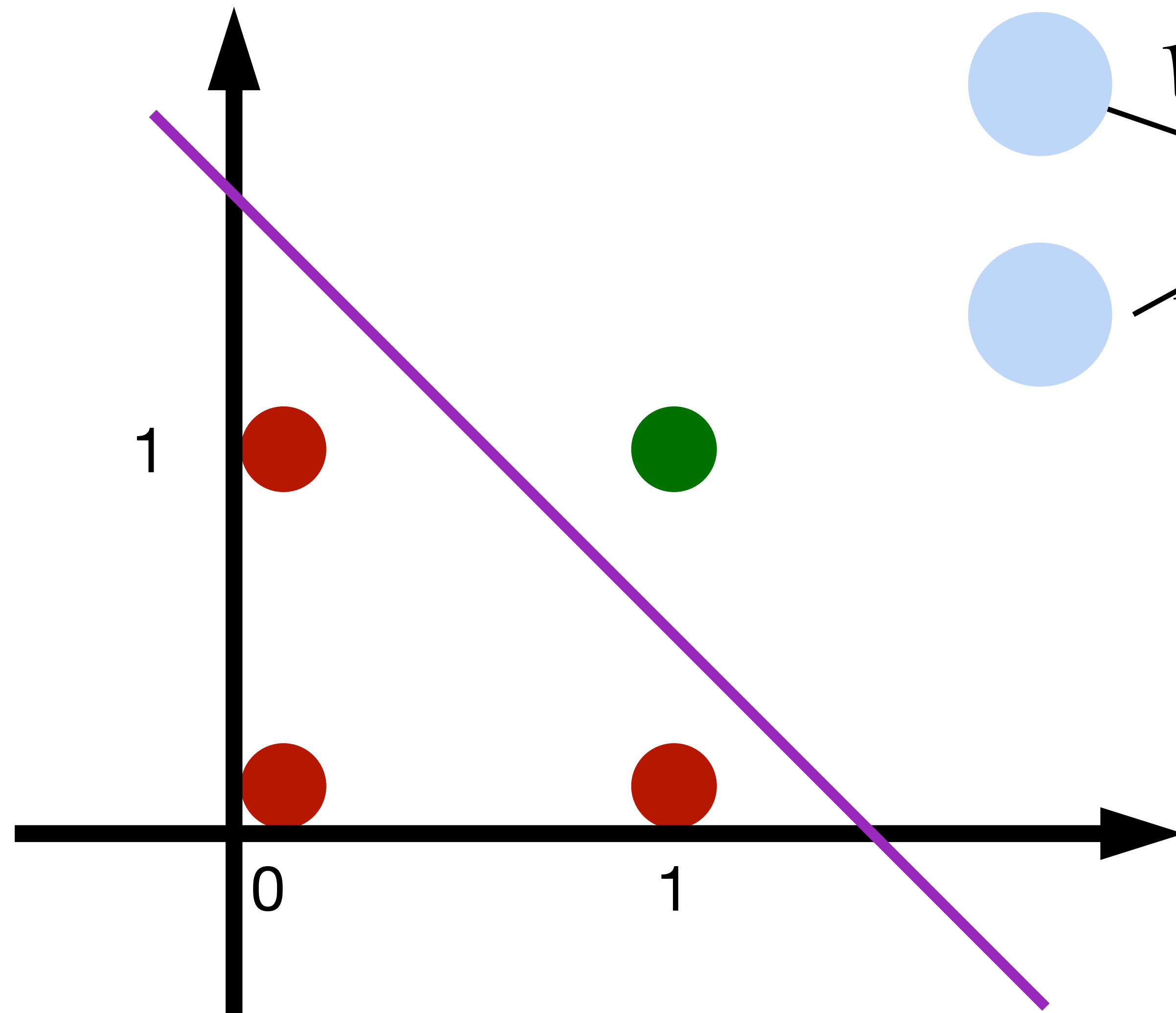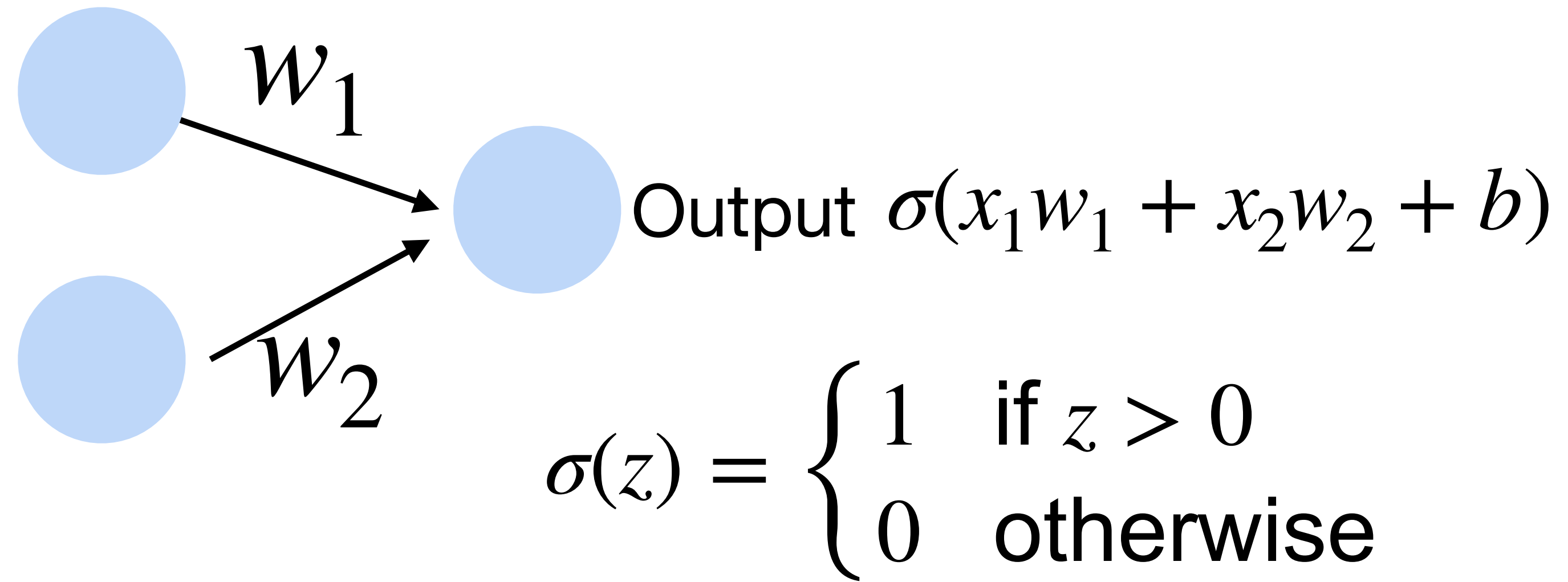
$x_1 = 0, x_2 = 1, y = 0$

$x_1 = 0, x_2 = 0, y = 0$

Learning AND function using perceptron

The perceptron can learn an AND function

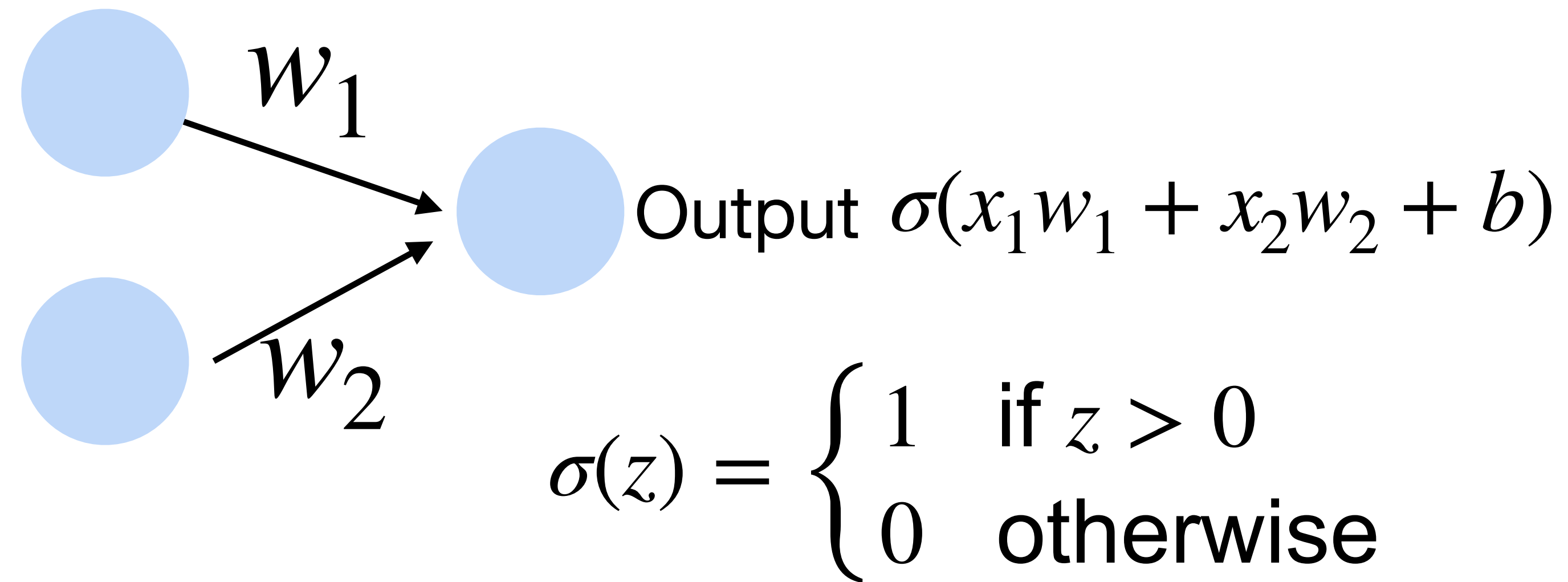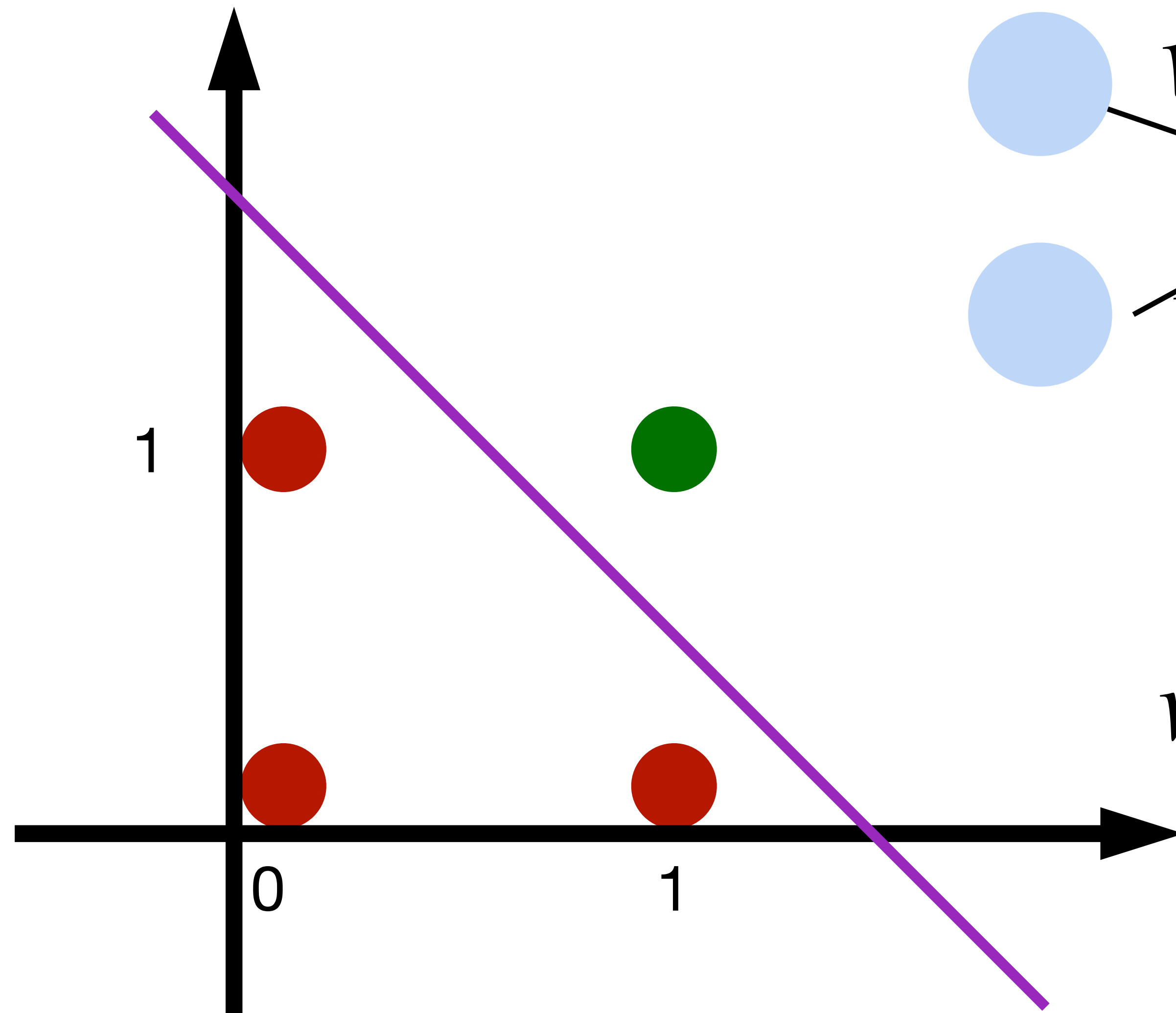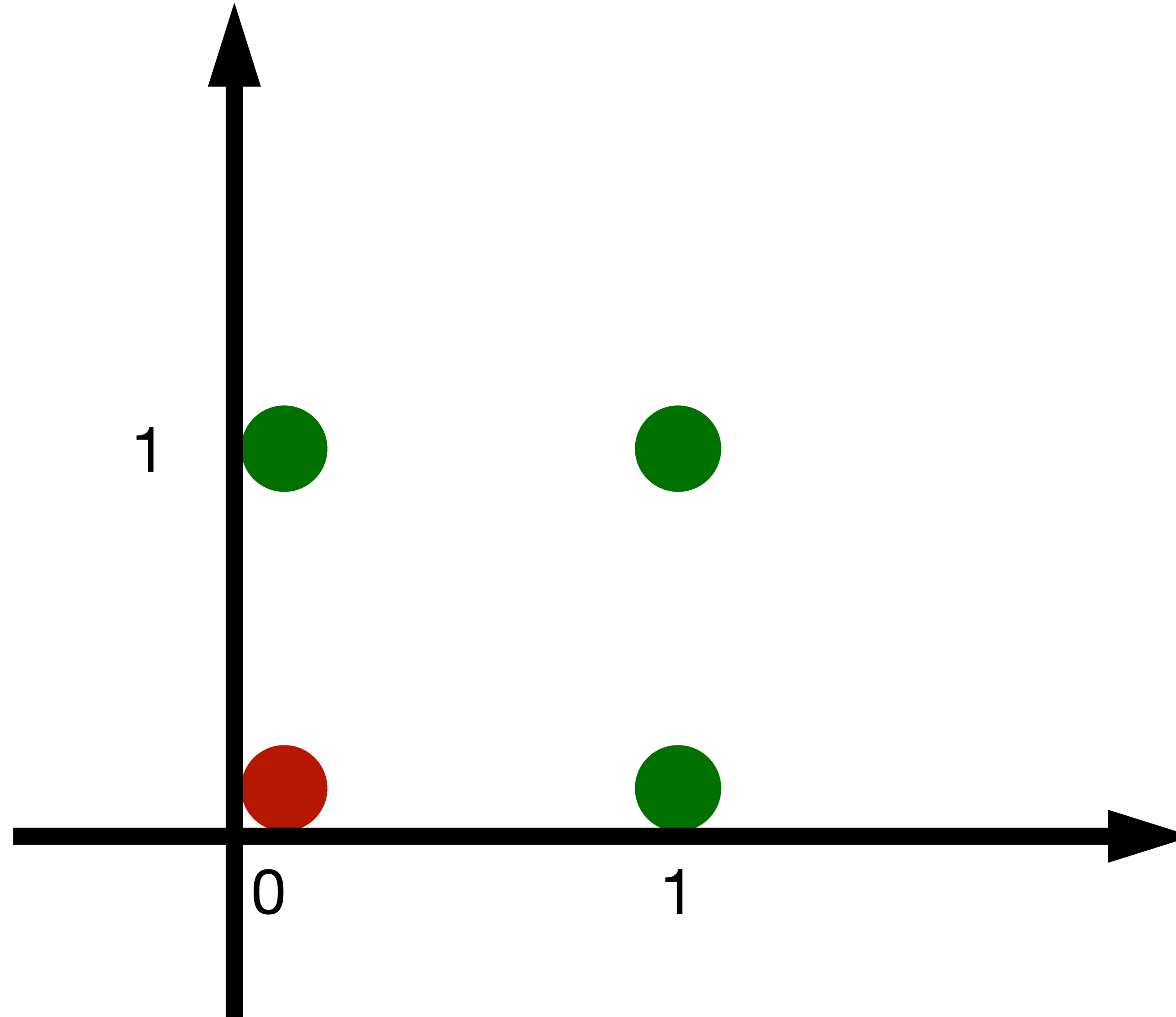# Learning AND function using perceptron

The perceptron can learn an AND function

$$\text{Output } \sigma(x_1 w_1 + x_2 w_2 + b)$$

$$\sigma(z) = \begin{cases} 1 & \text{if } z > 0 \\ 0 & \text{otherwise} \end{cases}$$

w_1

w_2

What's w and b?

Learning AND function using perceptron

The perceptron can learn an AND function

Output $\sigma(x_1 w_1 + x_2 w_2 + b)$

$$\sigma(z) = \begin{cases} 1 & \text{if } z > 0 \\ 0 & \text{otherwise} \end{cases}$$

$$w_1 = 1, w_2 = 1, b = -1.5$$

Learning OR function using perceptron

The perceptron can learn an OR function

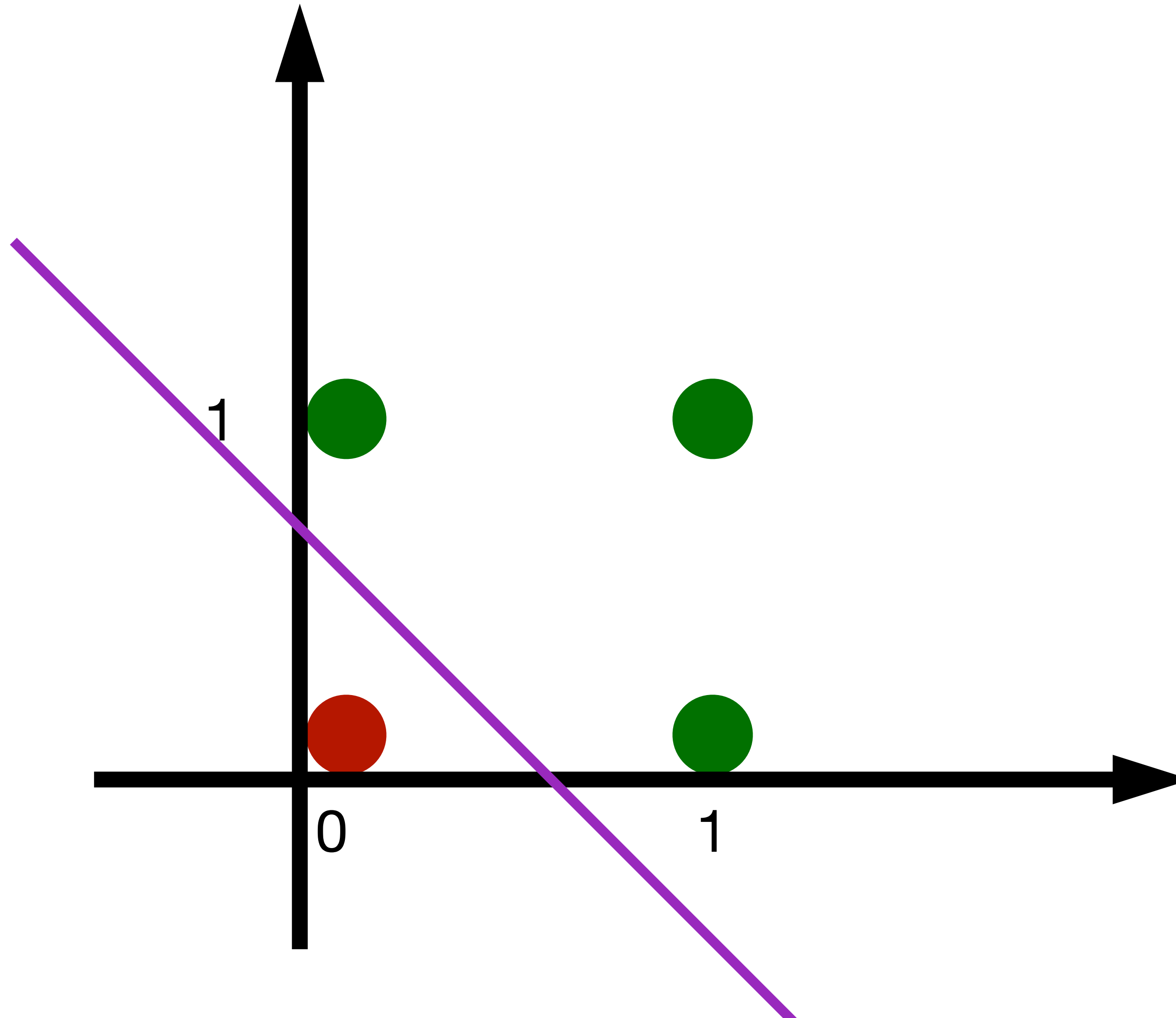$x_1 = 1, x_2 = 1, y = 1$

$x_1 = 1, x_2 = 0, y = 1$
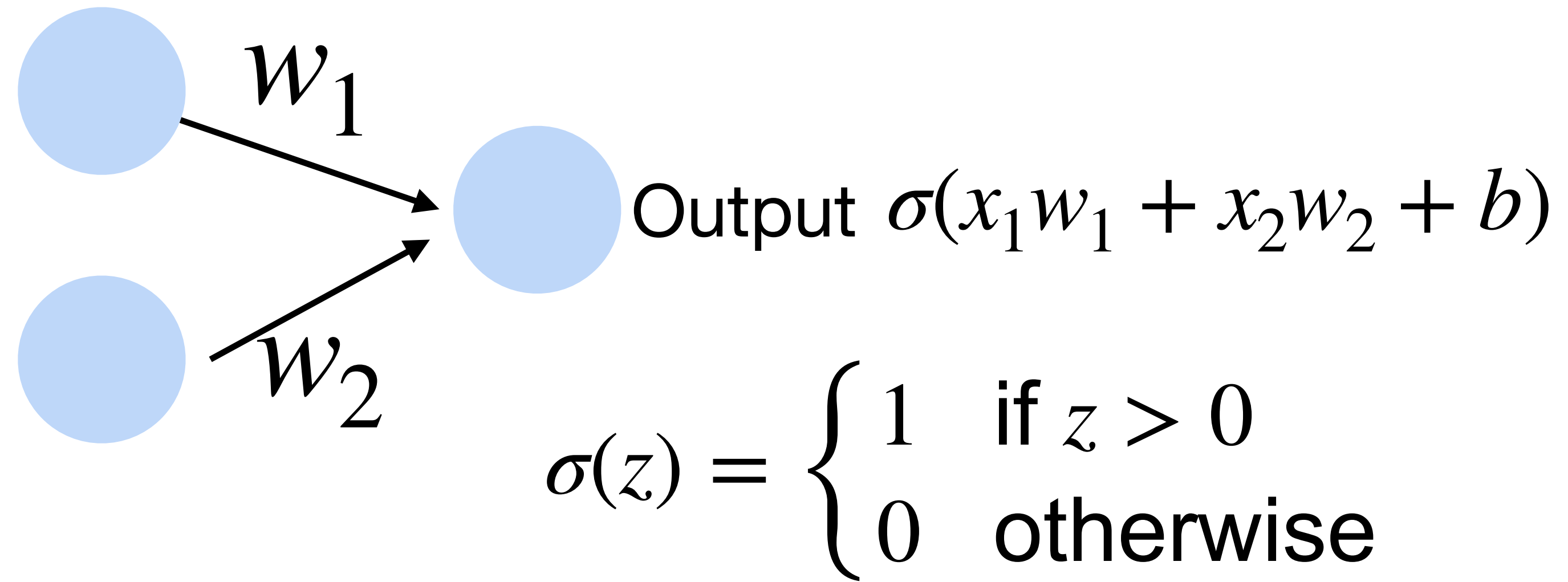
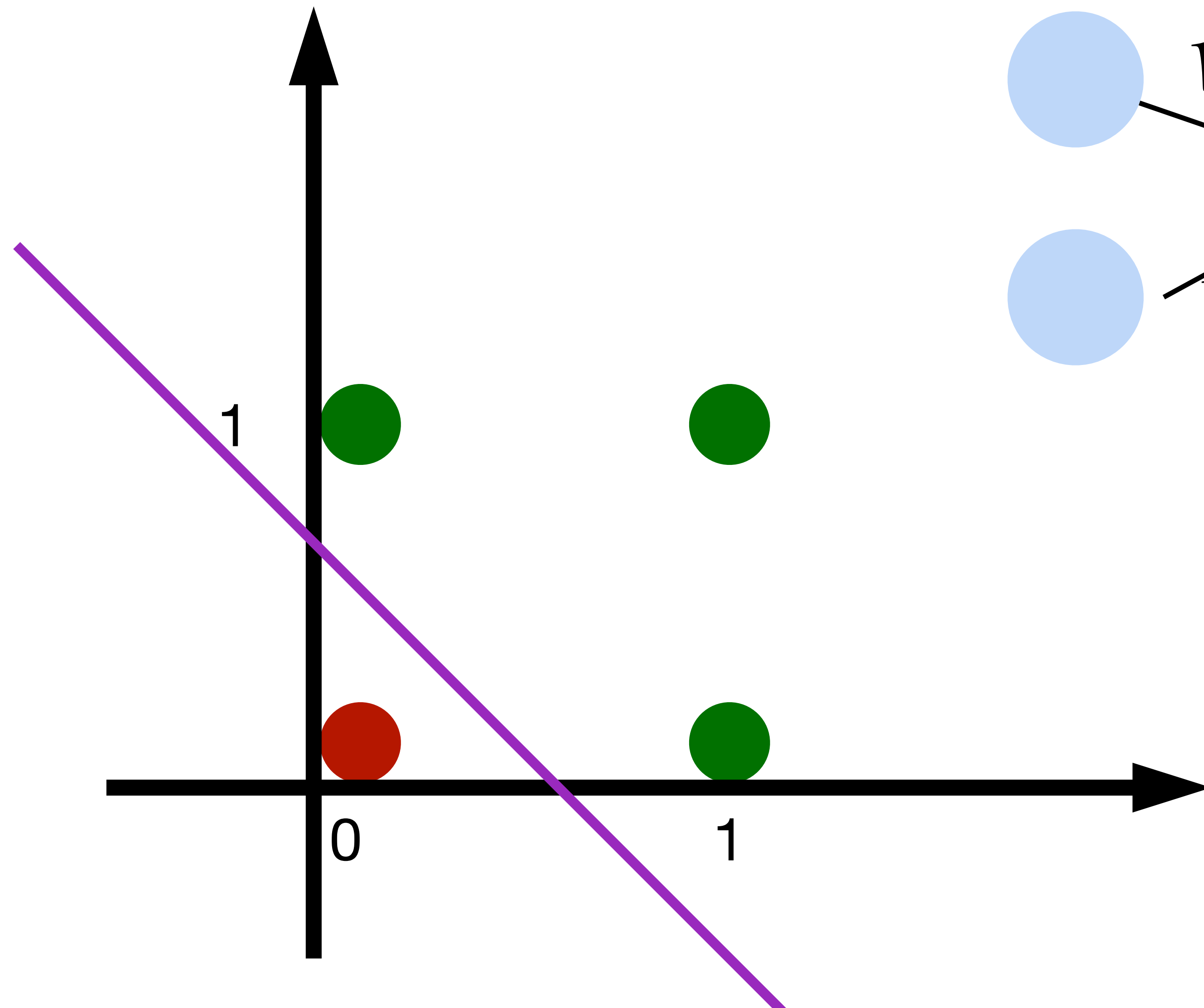$x_1 = 0, x_2 = 1, y = 1$

$x_1 = 0, x_2 = 0, y = 0$

Learning OR function using perceptron

The perceptron can learn an OR function
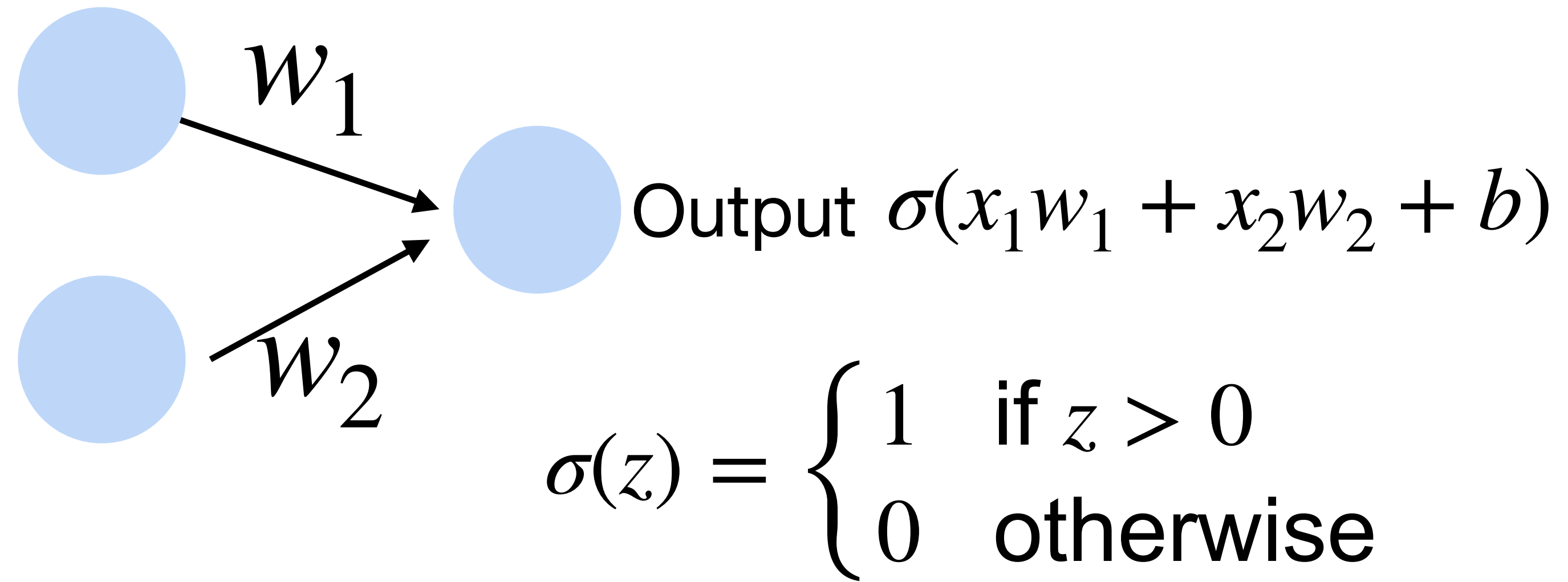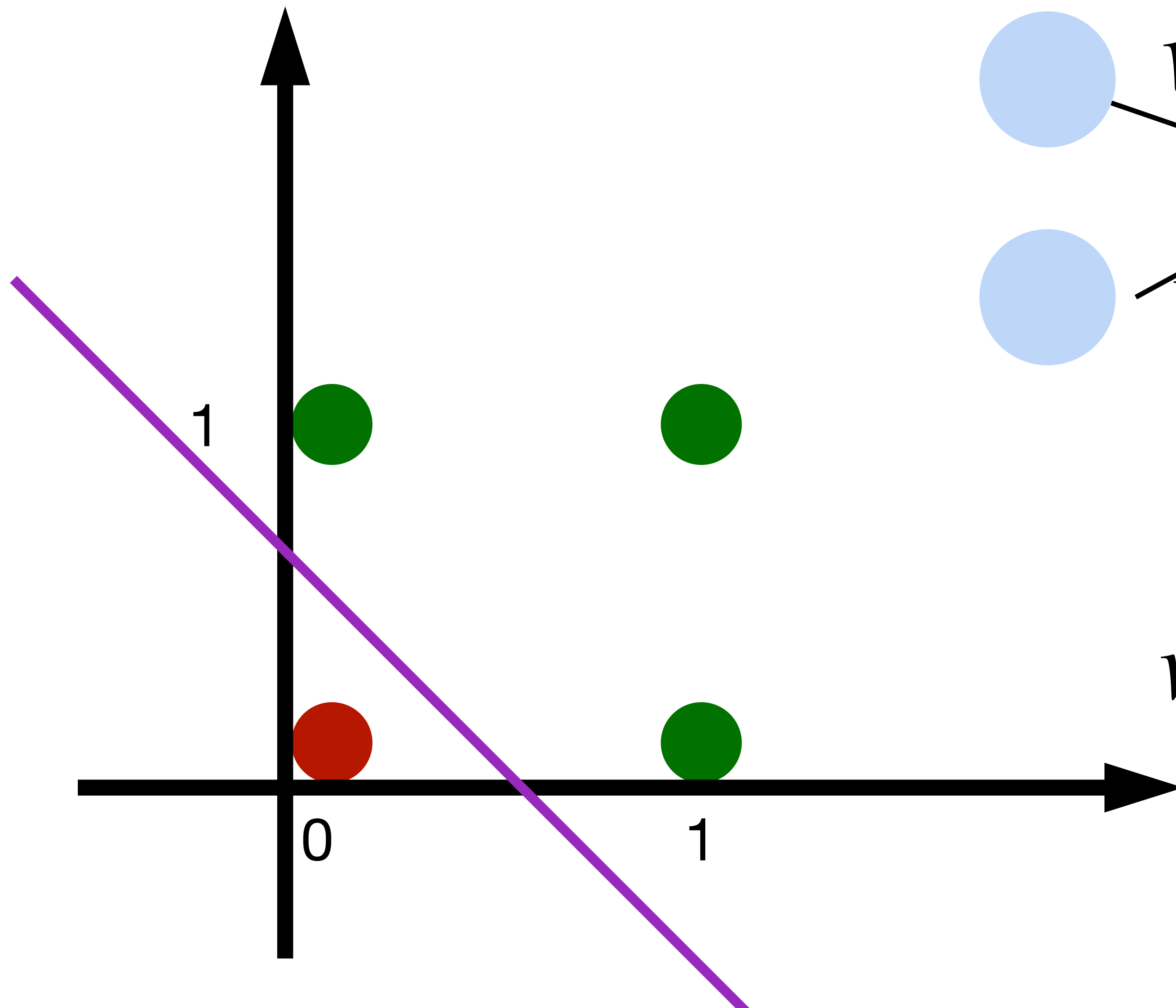
Learning OR function using perceptron

The perceptron can learn an OR function

Output $\sigma(x_1 w_1 + x_2 w_2 + b)$

w_1

w_2

$$\sigma(z) = \begin{cases} 1 & \text{if } z > 0 \\ 0 & \text{otherwise} \end{cases}$$

1

0   1

What's w and b?

Learning OR function using perceptron

The perceptron can learn an OR function

Output $\sigma(x_1 w_1 + x_2 w_2 + b)$

$$\sigma(z) = \begin{cases} 1 & \text{if } z > 0 \\ 0 & \text{otherwise} \end{cases}$$

$w_1 = 1, w_2 = 1, b = -0.5$

Learning NOT function using perceptron

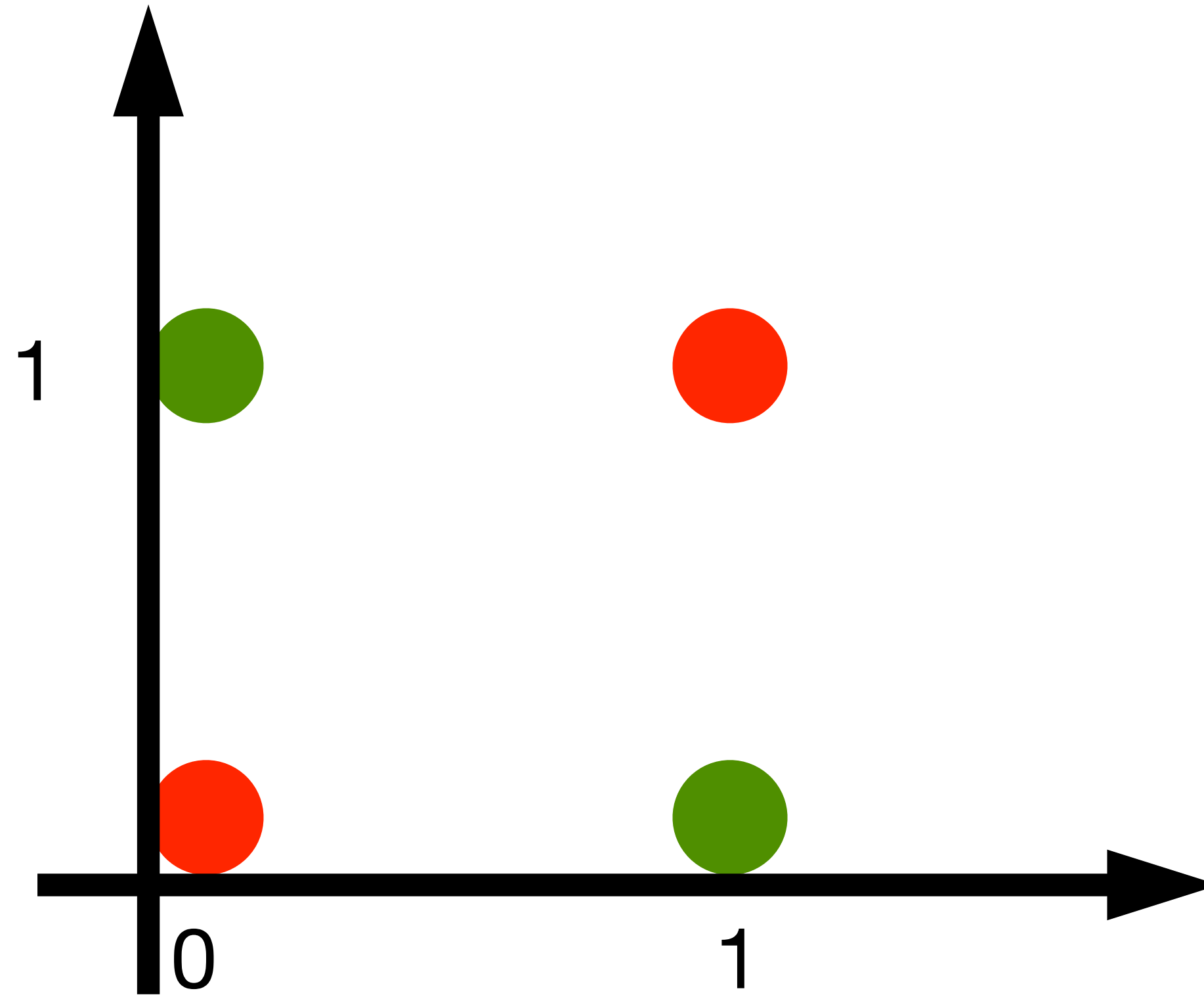The perceptron can learn NOT function (single input)

$$x \xrightarrow{w_1} \quad \text{Output } \sigma(xw_1 + b)$$

$$\sigma(z) = \begin{cases} 1 & \text{if } z > 0 \\ 0 & \text{otherwise} \end{cases}$$
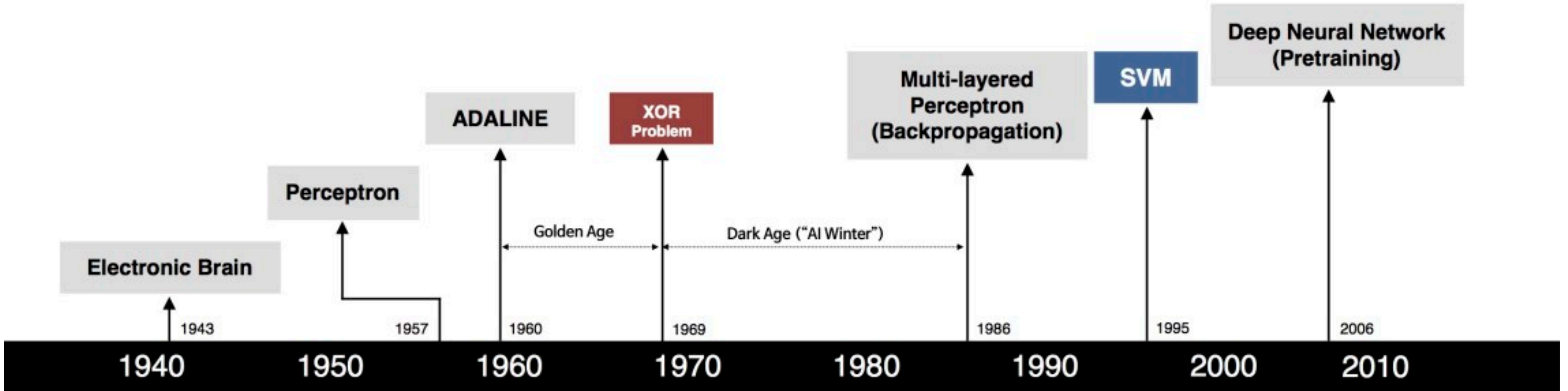
$$w_1 = -1, b = 0.5$$

0   1

XOR Problem (Minsky & Papert, 1969)

The perceptron cannot learn an XOR function
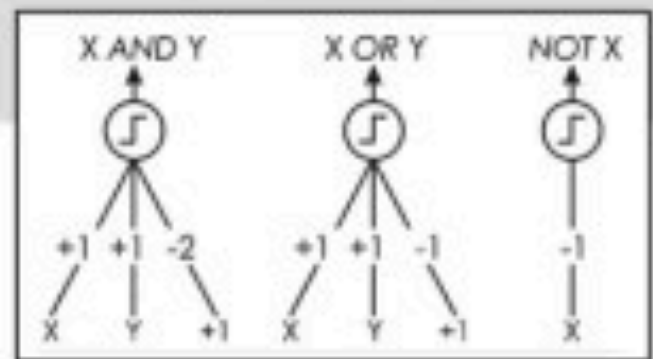(neurons can only generate linear separators)

This contributed to the first AI winter
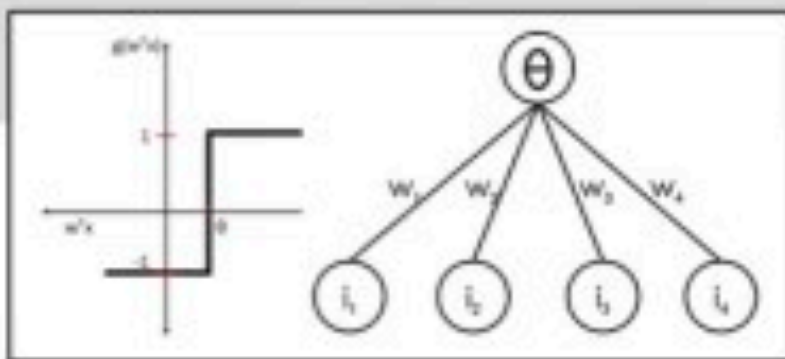
Brief history of neural networks

Timeline of neural networks history:

- **Electronic Brain** — 1943 — S. McCulloch – W. Pitts
  - Adjustable Weights
  - Weights are not Learned
- **Perceptron** — 1957 — F. Rosenblatt
  - Learnable Weights and Threshold
- **ADALINE** — 1960 — B. Widrow – M. Hoff
- **XOR Problem** — 1969 — M. Minsky – S. Papert
  - XOR Problem
- **Multi-layered Perceptron (Backpropagation)** — 1986 — D. Rumelhart – G. Hinton – R. Wiliams
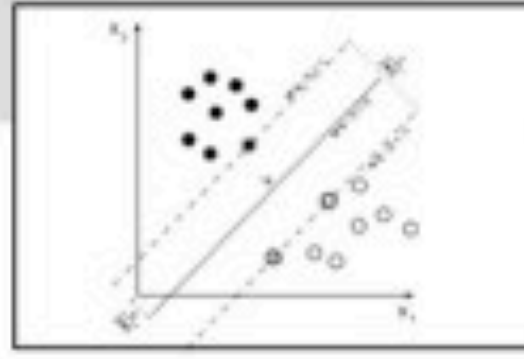 - Solution to nonlinearly separable problems
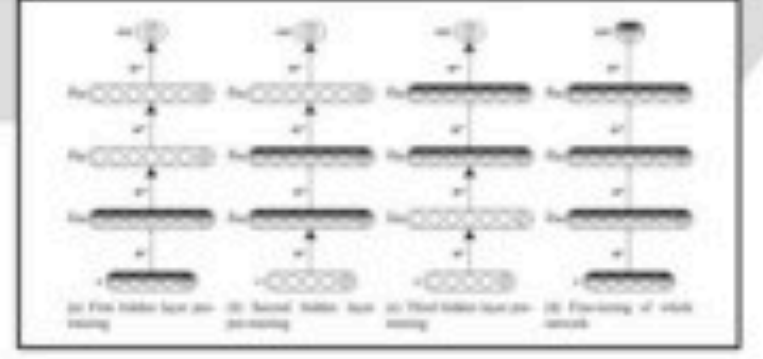 - Big computation, local optima and overfitting
- **SVM** — 1995 — V. Vapnik – C. Cortes
  - Limitations of learning prior knowledge
  - Kernel function: Human Intervention
- **Deep Neural Network (Pretraining)** — 2006 — G. Hinton – S. Ruslan
  - Hierarchical feature Learning
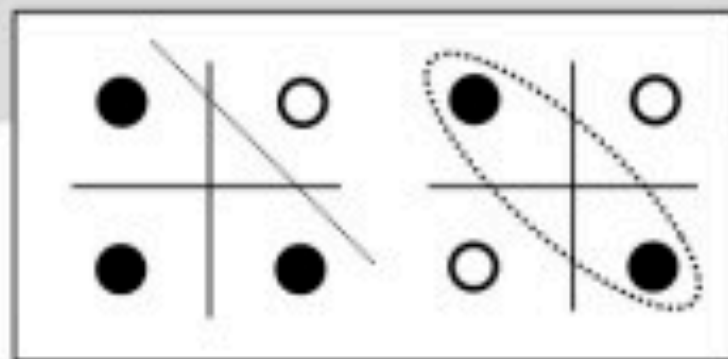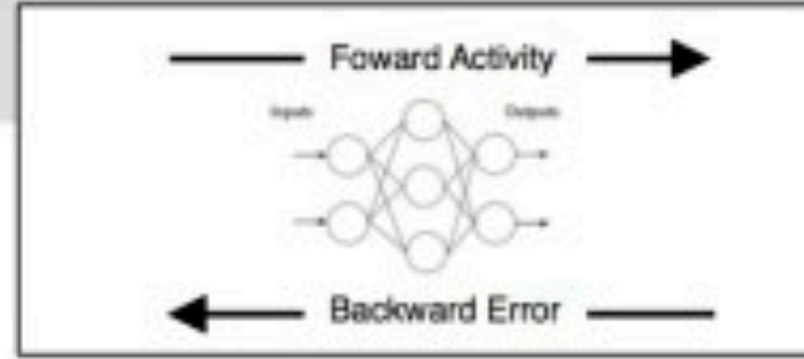
Golden Age — Dark Age ("AI Winter")

1940   1950   1960   1970   1980   1990   2000   2010

Quiz Break

Consider the linear perceptron with x as the input. Which function can the linear perceptron compute?

(1) $y = ax + b$

(2) $y = ax^2 + bx + c$

A. (1)

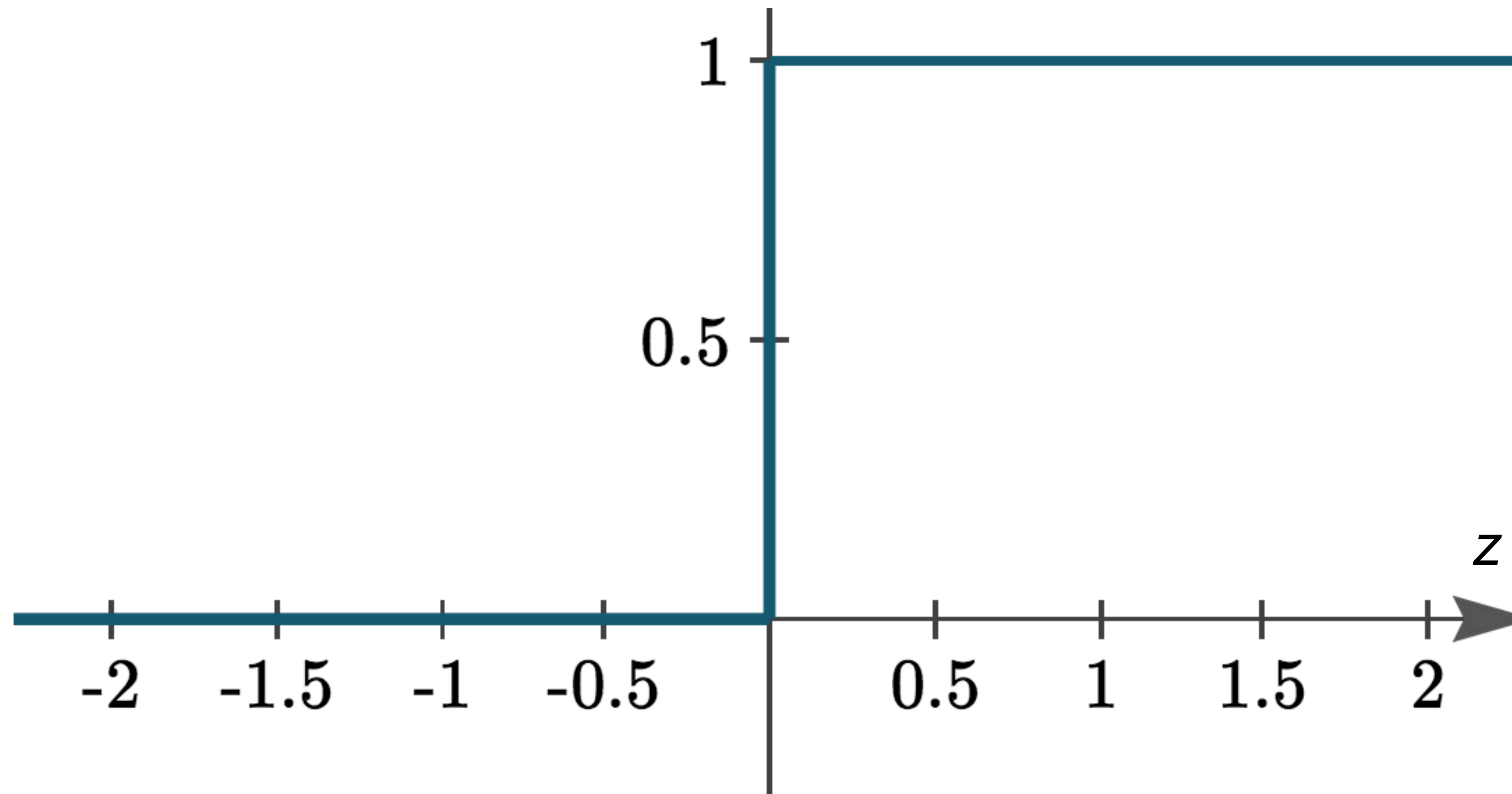B. (2)

C. (1)(2)

D. None of the above

Quiz Break

Perceptron can be used for representing:

A. AND function

B. OR function

C. XOR function

D. Both AND and OR function

Step Function activation

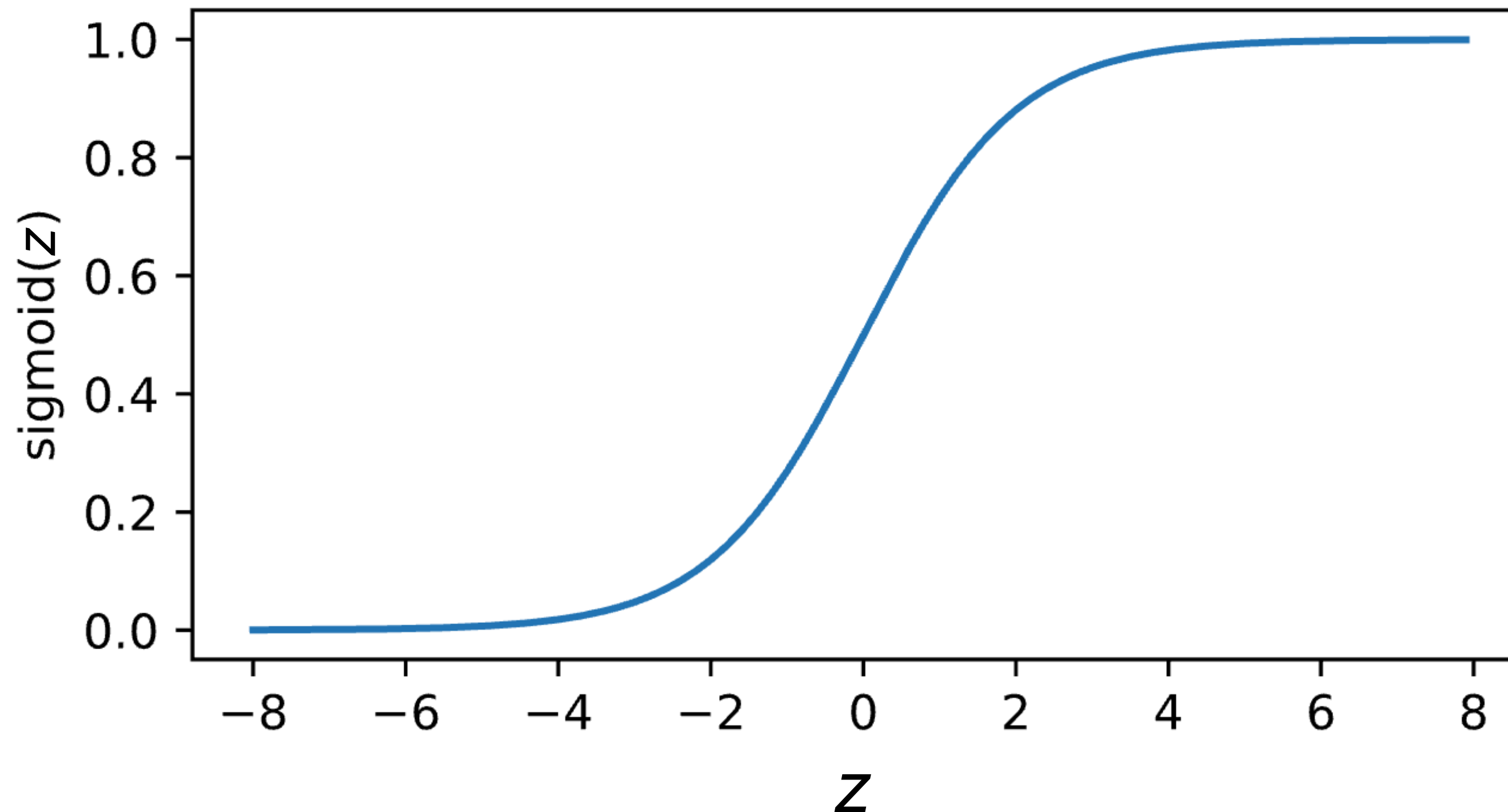Step function is discontinuous, which cannot be used for gradient descent

$$\sigma(z) = \begin{cases} 1 & \text{if } z > 0 \\ 0 & \text{otherwise} \end{cases}$$

Sigmoid/Logistic Activation

Map input into [0, 1], a **soft** version of $\sigma(z) = \begin{cases} 1 & \text{if } z > 0 \\ 0 & \text{otherwise} \end{cases}$

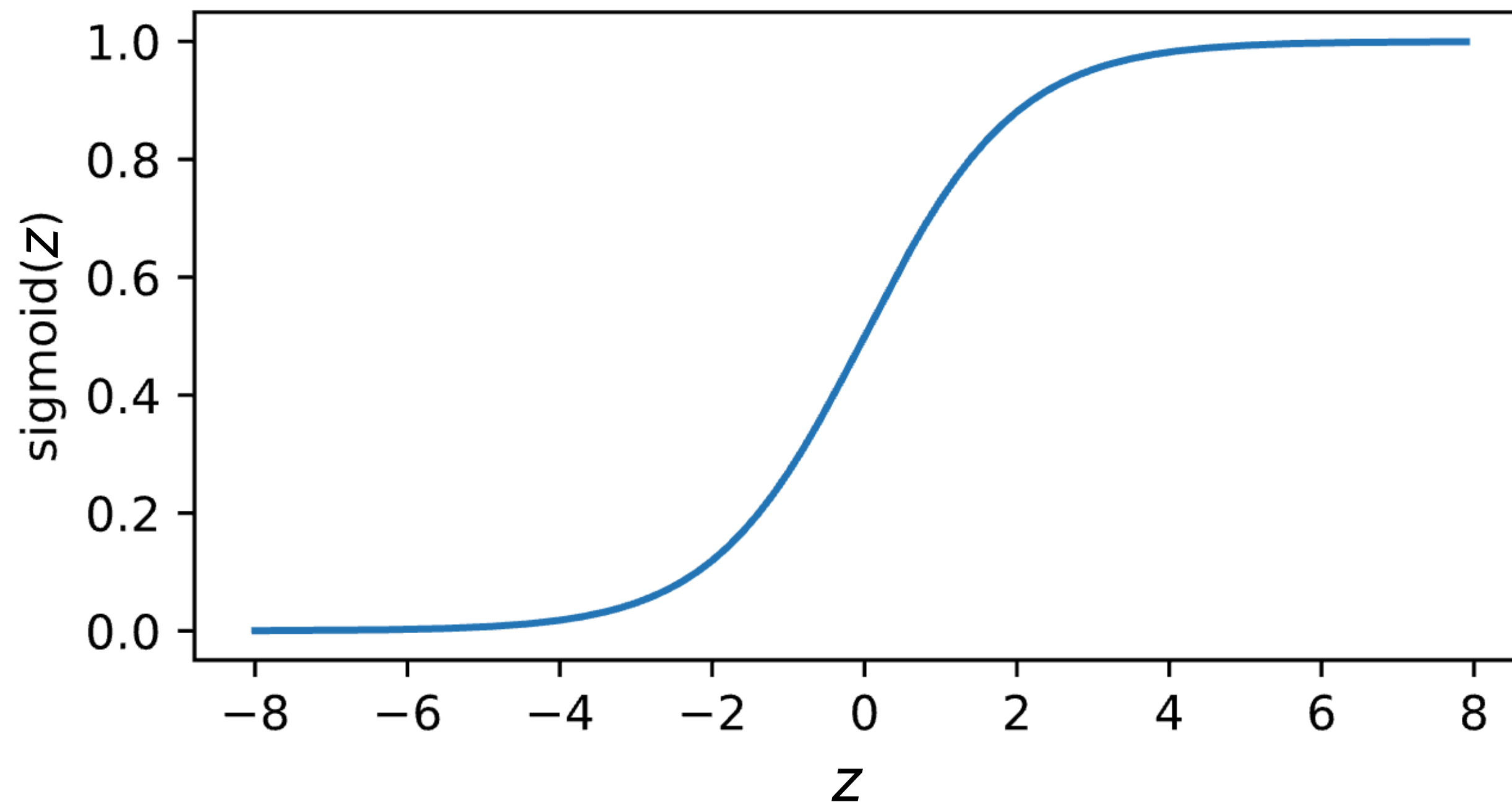$$\text{sigmoid}(z) = \frac{1}{1 + \exp(-z)}$$

Logistic regression

$$\mathbf{x} \in \mathbb{R}^d, y = \{-1, +1\}$$

$$p(y = 1 \mid \mathbf{x}) = \sigma(\mathbf{w}^T \mathbf{x}) = \frac{1}{1 + \exp(-\mathbf{w}^T \mathbf{x})}$$

$$p(y = -1 \mid \mathbf{x}) = 1 - \sigma(\mathbf{w}^T \mathbf{x}) = \frac{1}{1 + \exp(\mathbf{w}^T \mathbf{x})}$$

Logistic regression

Given: $\{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$

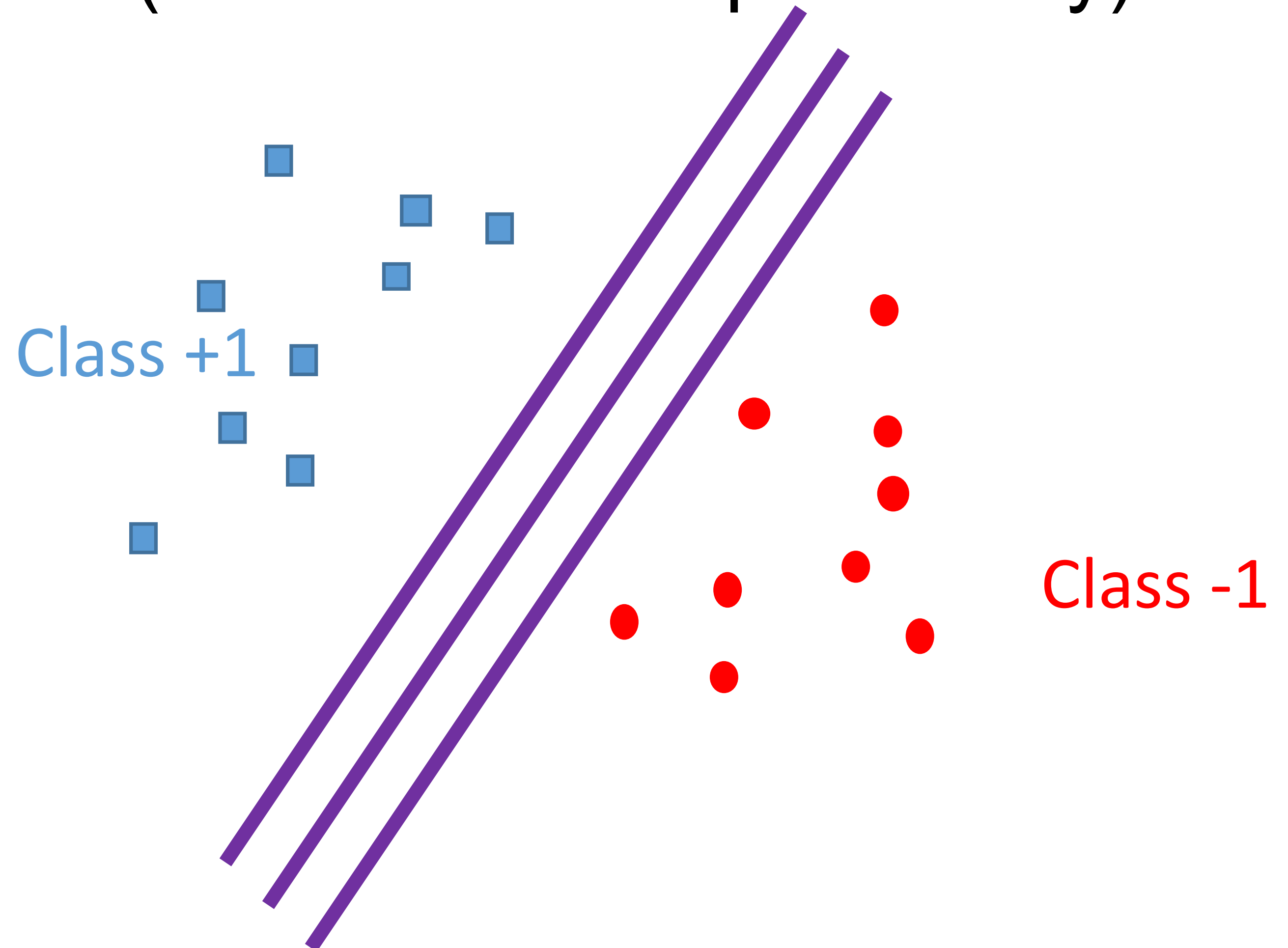Training: maximize the likelihood (the conditional probability)

$$\max_{\mathbf{w}} \sum_i \log \frac{1}{1 + \exp(-y_i \mathbf{w}^T \mathbf{x}_i)}$$

Logistic regression

Given: $\{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$

Training: maximize the likelihood (the conditional probability)

Class +1

Class -1

When training data is linearly separable, many solutions

Logistic regression

Given: $\{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$

Training: maximize the regularized likelihood
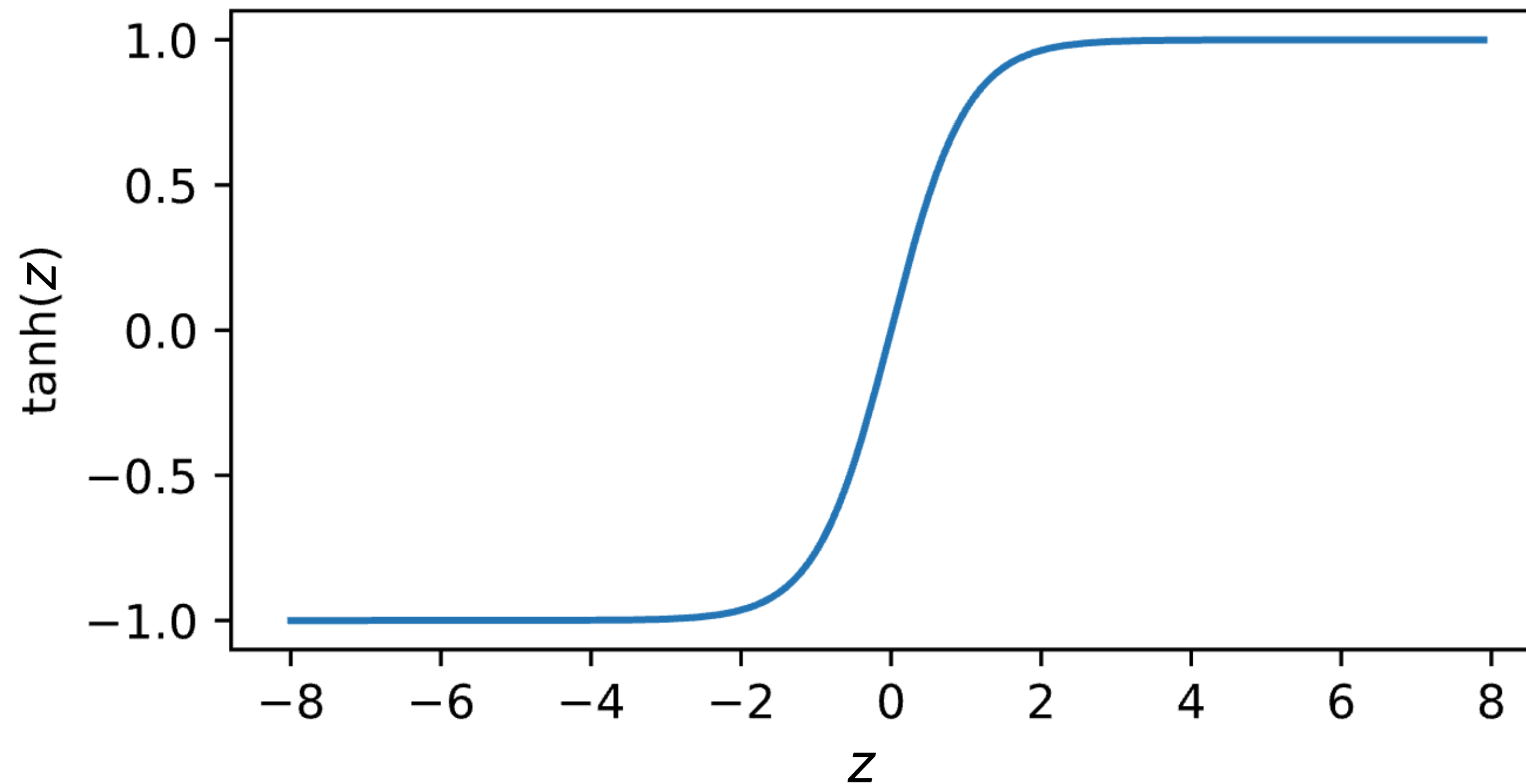
$$\min_{\mathbf{w}} \sum_i \left(-\log \frac{1}{1 + \exp(-y_i \mathbf{w}^T \mathbf{x}_i)} \right) + \frac{\lambda}{2} \|\mathbf{w}\|_2^2$$

- Convex optimization
- Solve via (stochastic) gradient descent
- Related to *maximum A posteriori (MAP)* estimate

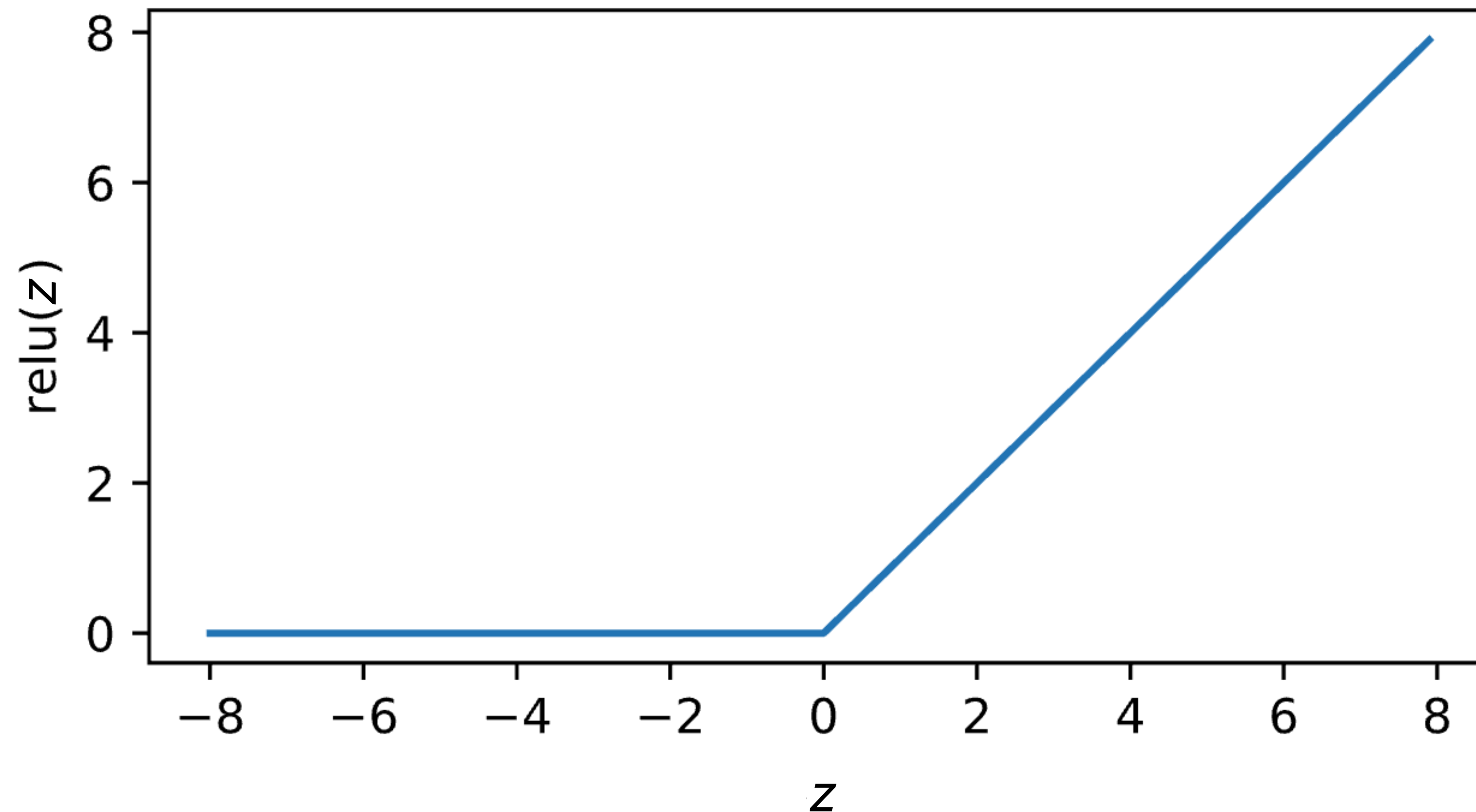# **Tanh Activation**

Map inputs into (-1, 1)

$$\tanh(z) = \frac{1 - \exp(-2z)}{1 + \exp(-2z)}$$

ReLU Activation

ReLU: Rectified Linear Unit (commonly used in modern neural networks)
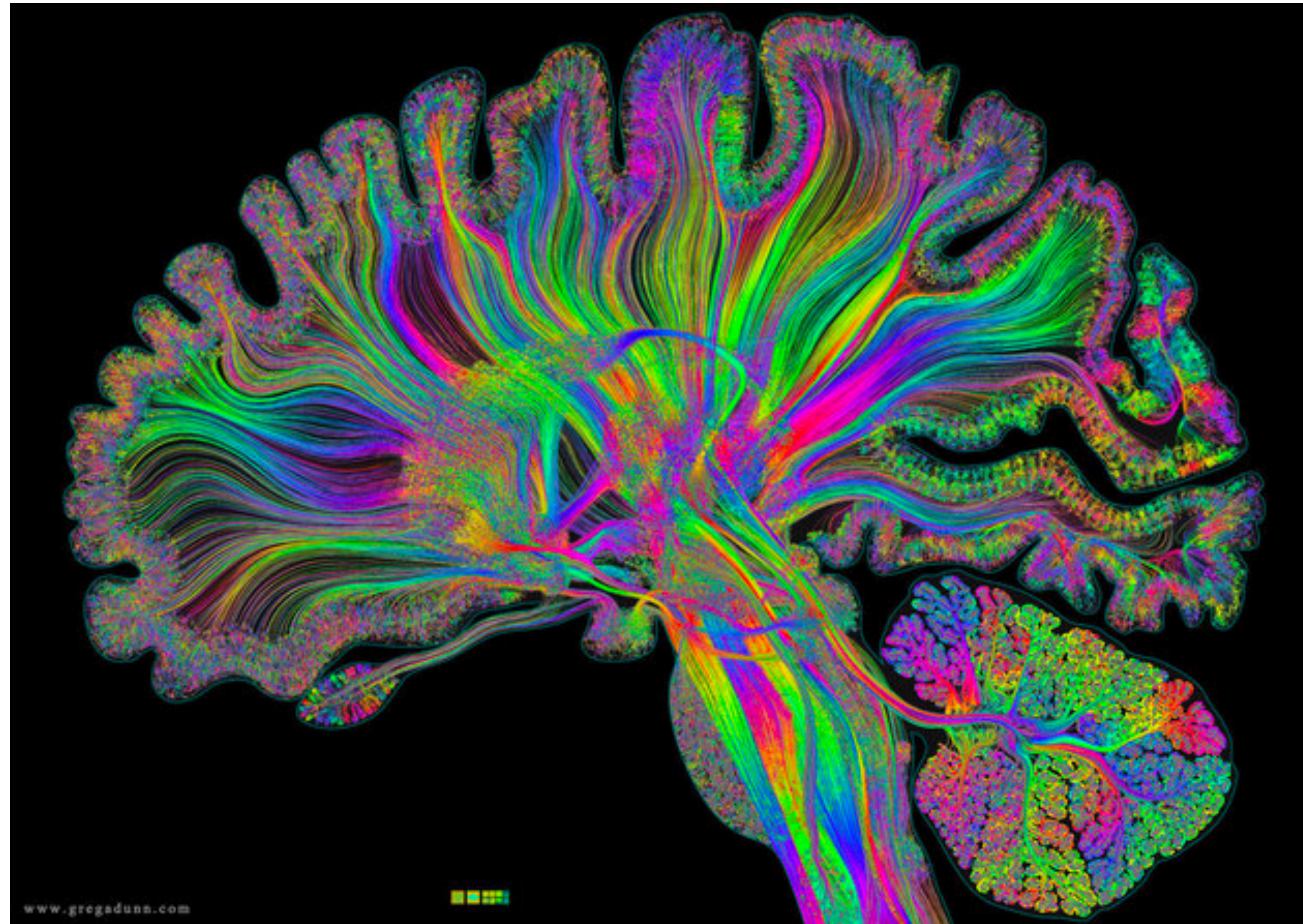
$$\text{ReLU}(z) = \max(z, 0)$$

Quiz Break

Which one of the following is valid activation function

a) Step function

b) Sigmoid function

C) ReLU function

D) all of above

Coming Next:

Multi-layer Perceptron

www.gregadunn.com

Thanks!

Based on slides from Sharon Li, Xiaojin (Jerry) Zhu and Yingyu Liang, and Alex Smola: https://courses.d2l.ai/
berkeley-stat-157/units/mlp.html