CS 540 Introduction to Artificial Intelligence

Perceptron

Sharon Yixuan Li
University of Wisconsin-Madison

March 4, 2021

Today’s outline

- Naive Bayes (cont.)
- Single-layer Neural Network (Perceptron)

Part I: Naïve Bayes (cont.)

Example 1: Play outside or not?

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

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

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}

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}

$$p(\text{Play} \mid \text{☀}) = \frac{p(\text{☀} \mid \text{Play})\, p(\text{Play})}{p(\text{☀})}$$

Bayes rule

Example 1: Play outside or not?

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

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

https://www.analyticsvidhya.com/blog/2017/09/naive-bayes-explained/

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

https://www.analyticsvidhya.com/blog/2017/09/naive-bayes-explained/

Example 1: Play outside or not?

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

P(Yes| )
=P( |Yes)*P(Yes)/P()

?

P(No| )
=P( |No)*P(No)/P()

?

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!

Bayesian classification

$$\hat{y} = \arg\max_y p(y \mid \mathbf{x})$$ (Posterior)

(Prediction)

$$= \arg\max_y \frac{p(\mathbf{x} \mid y) \cdot p(y)}{p(\mathbf{x})}$$ (by Bayes' rule)

$$= \arg\max_y \; p(\mathbf{x} \mid y) p(y)$$

Bayesian classification

What if **x** has multiple attributes $\mathbf{x} = \{X_1, \ldots, X_k\}$

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

(Prediction)

Bayesian classification

What if **x** has multiple attributes $\mathbf{x} = \{X_1, \ldots, X_k\}$

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

(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 classification

What if $\mathbf{x}$ has multiple attributes $\mathbf{x} = \{X_1, \ldots, X_k\}$

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

(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!)

Part I: Single-layer Neural Network

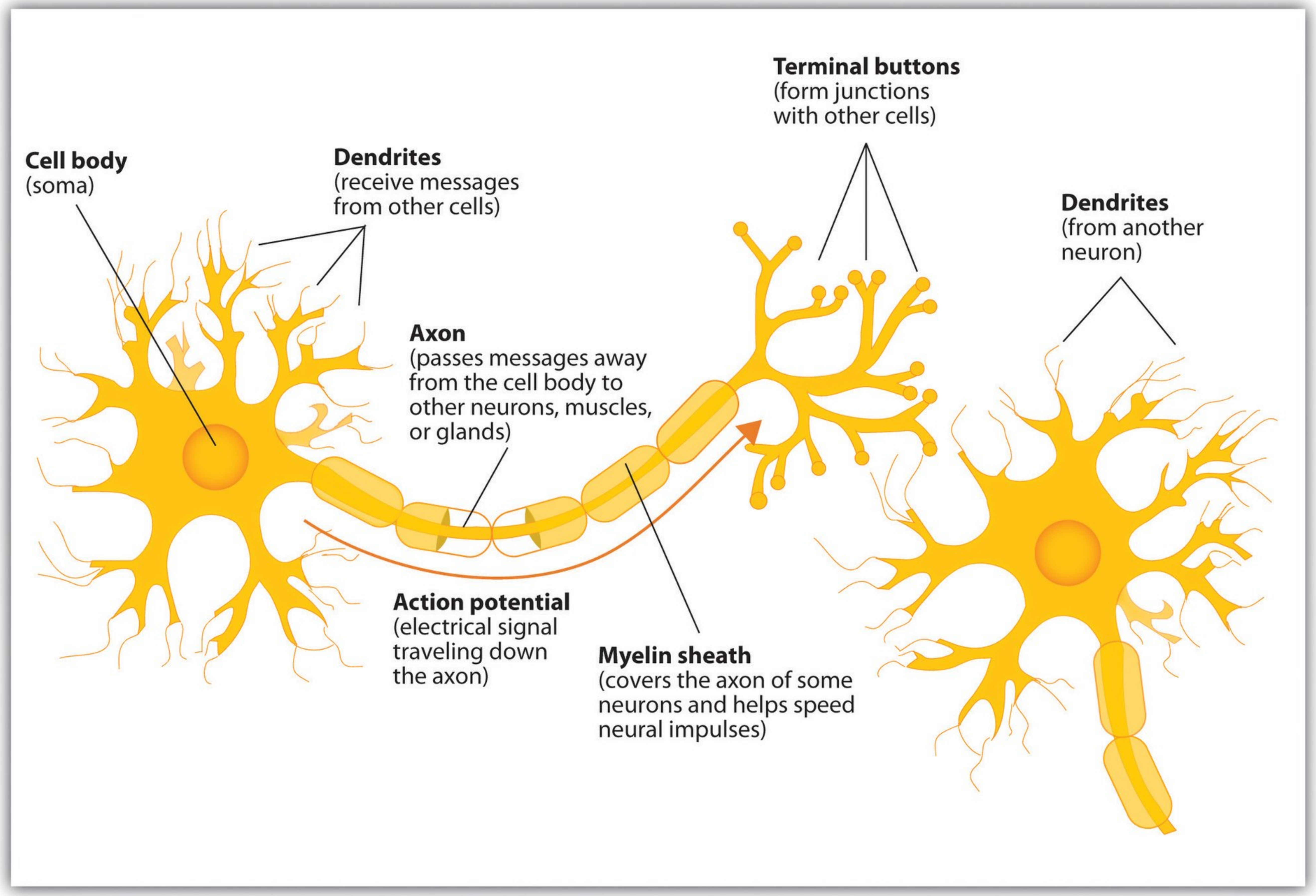
How to classify

Cats vs. dogs?

Inspiration from neuroscience

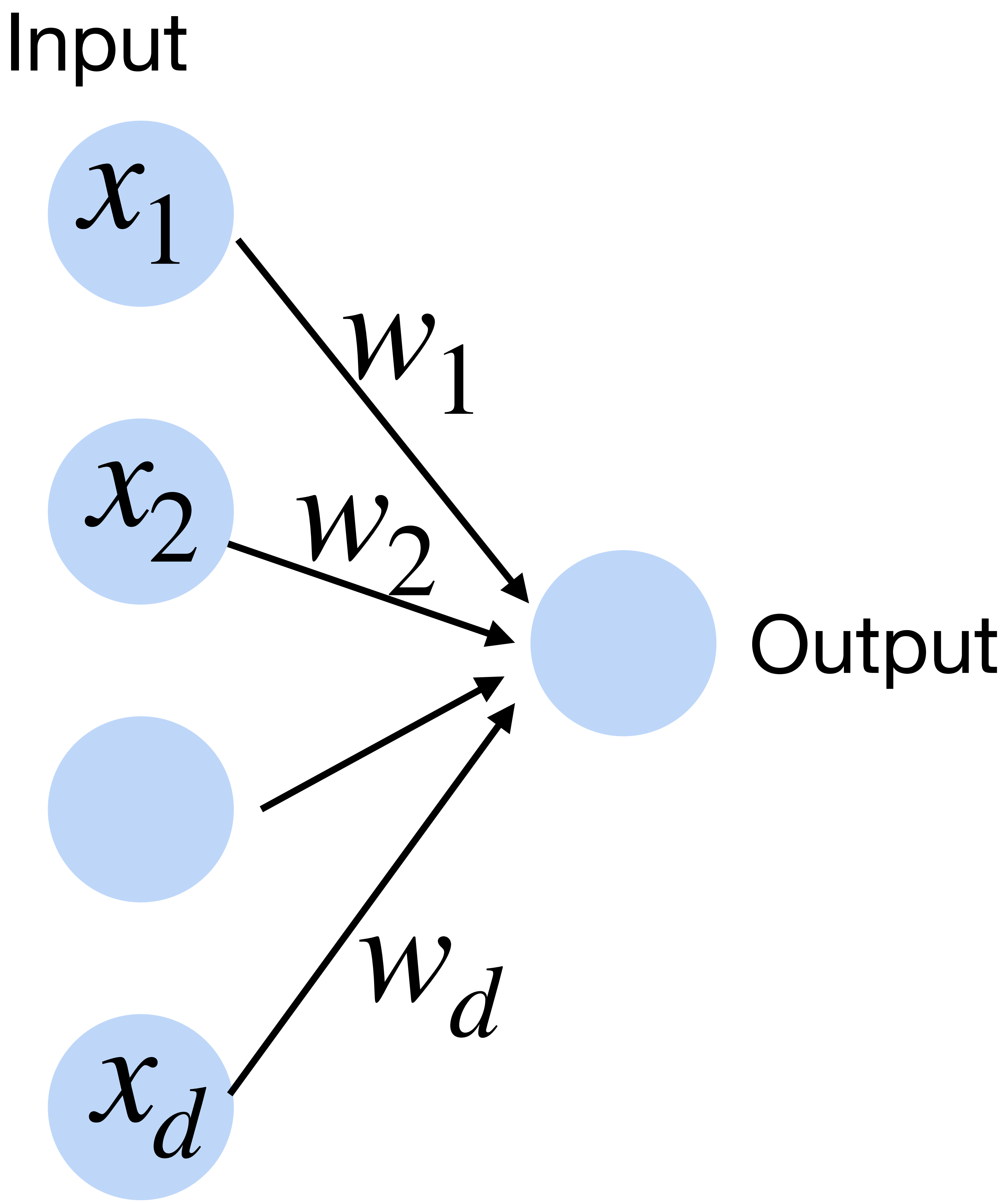
- Inspirations from human brains
- Networks of simple and homogenous units

(wikipedia)



Perceptron

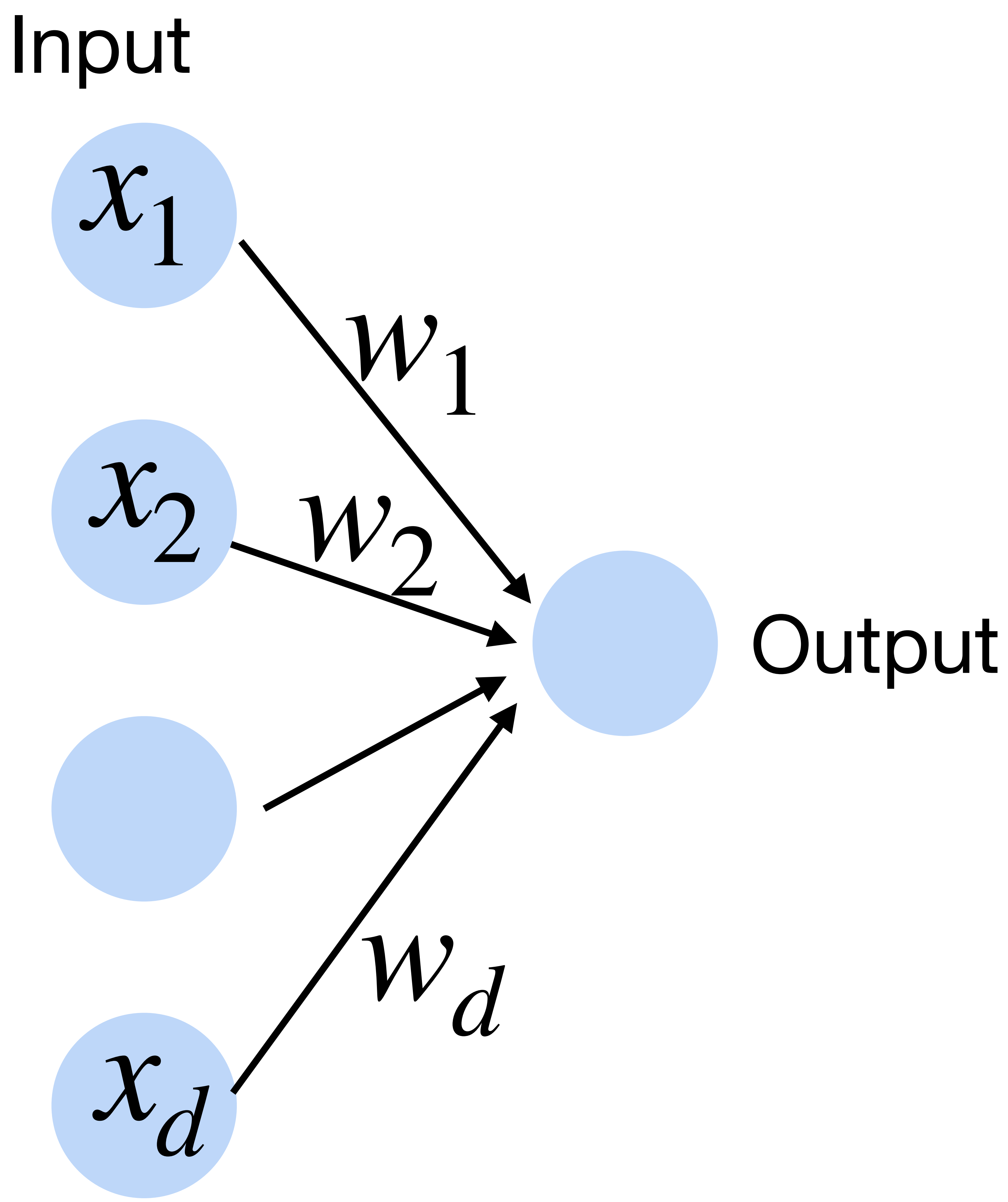
Cats vs. dogs?



Linear Perceptron

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

$$f = \langle \mathbf{w}, \mathbf{x} \rangle + b$$



Cats vs. dogs?

Perceptron

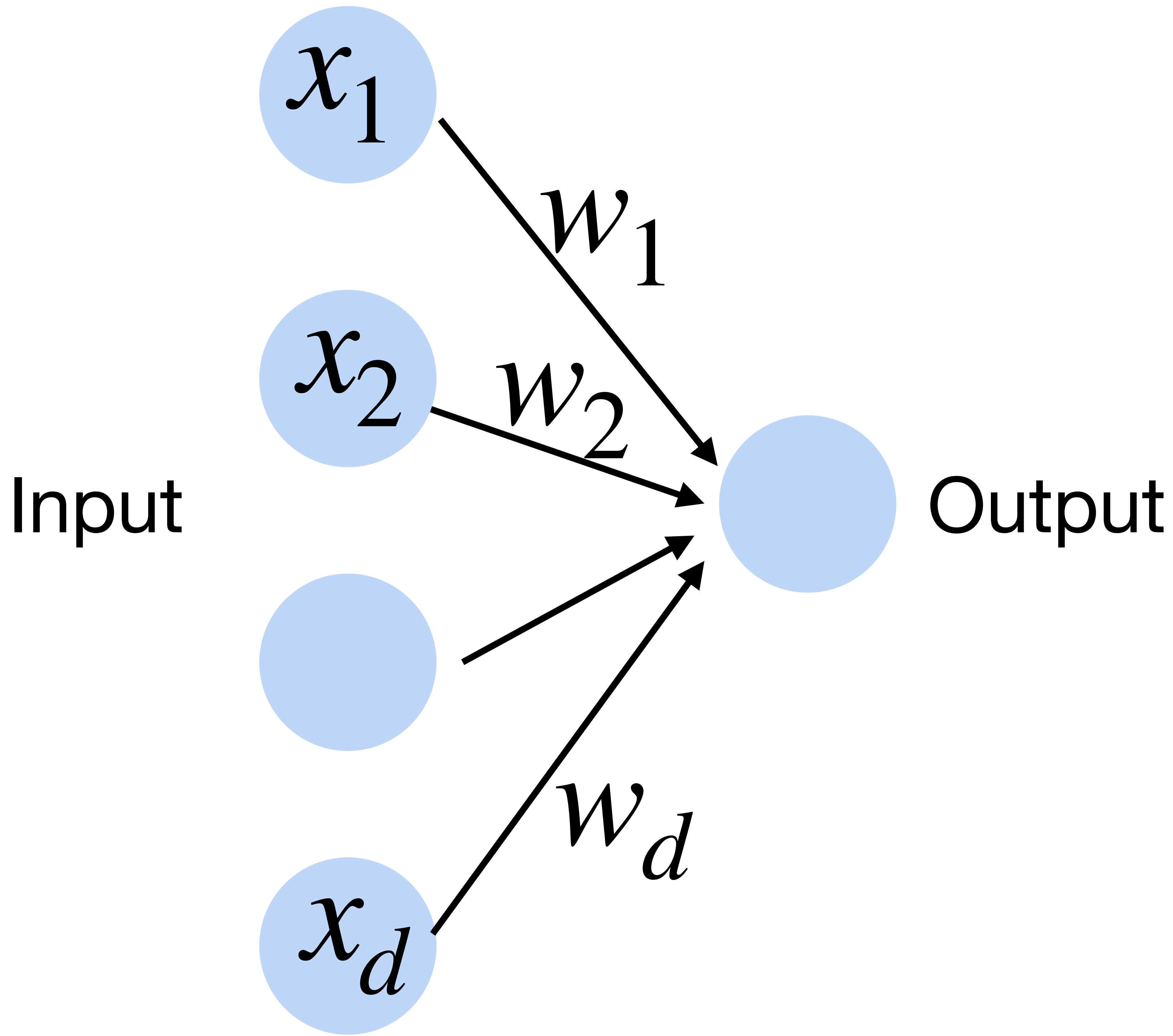
- Given input $\mathbf{x}$, weight $\mathbf{w}$ and bias b, perceptron outputs:

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

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

Activation function

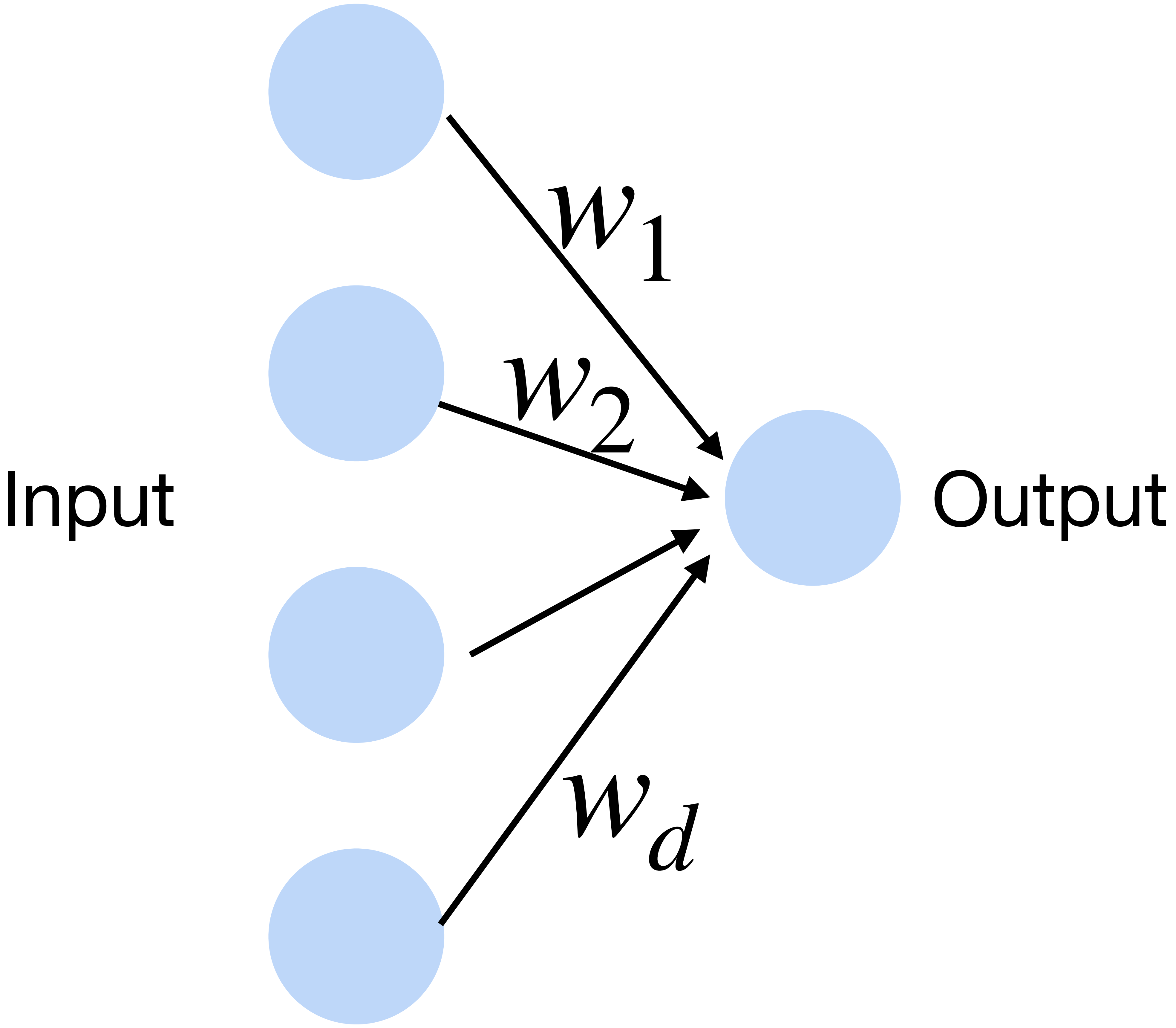
Cats vs. dogs?



Perceptron

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

Cats vs. dogs?

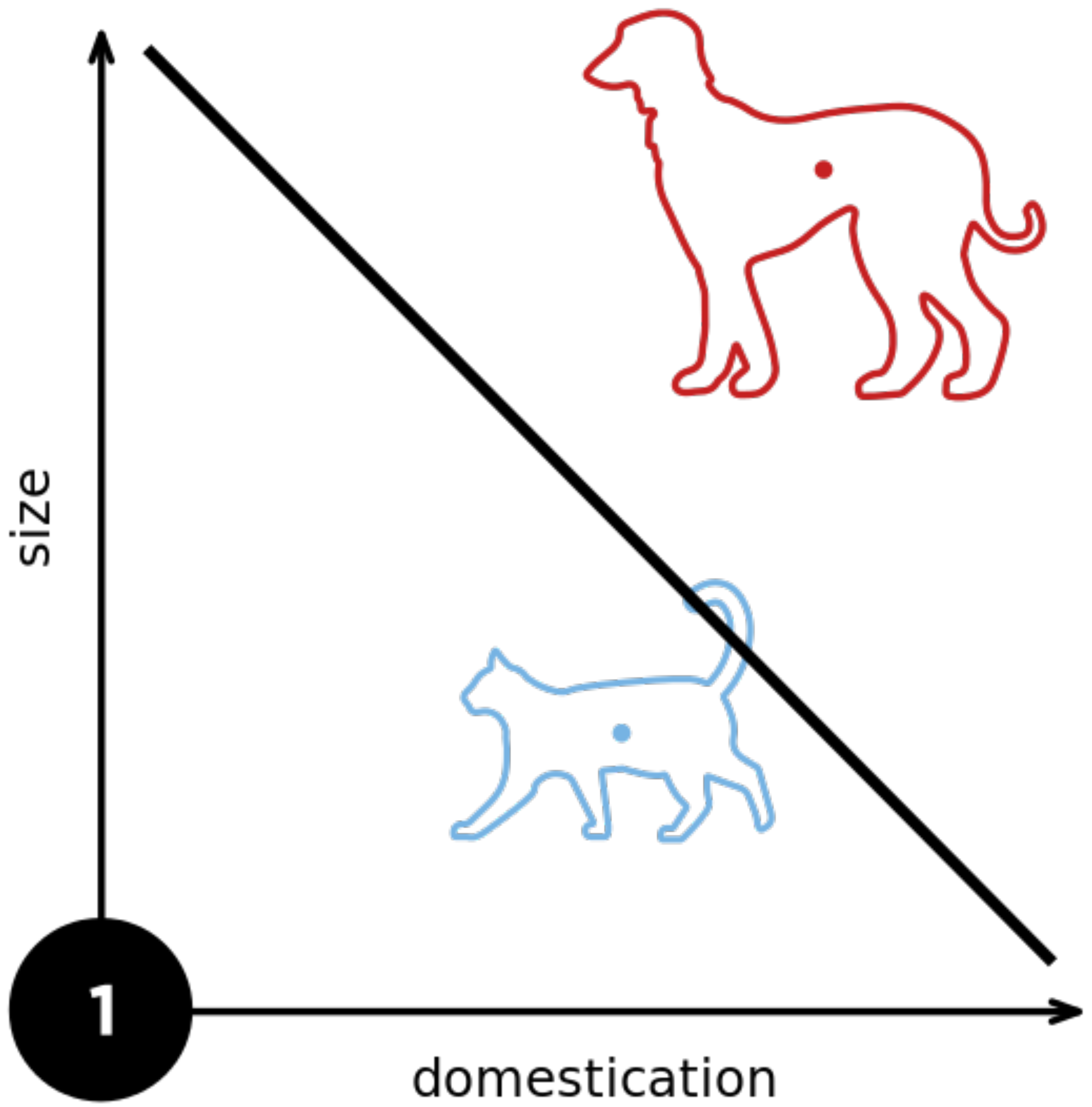


Training the Perceptron

Perceptron Algorithm

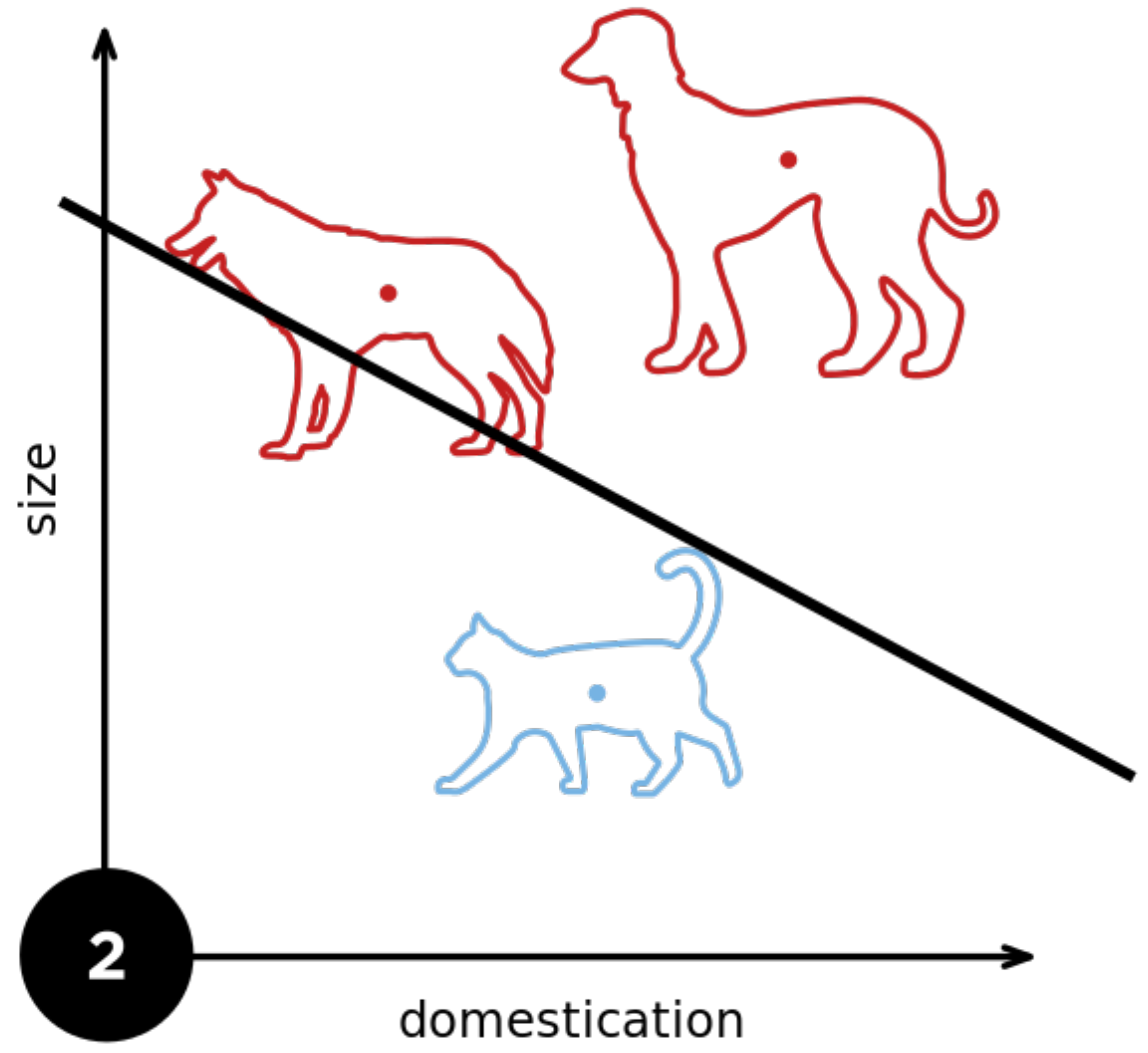
Initialize $\vec{w} = \vec{0}$ // Initialize $\vec{w}$. $\vec{w} = \vec{0}$ misclassifies everything.
while TRUE **do** // Keep looping
  $m = 0$ // Count the number of misclassifications, m
  for $(x_i, y_i) \in D$ **do** // Loop over each (data, label) pair in the dataset,
    if $y_i(\vec{w}^T \cdot \vec{x}_i) \leq 0$ **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



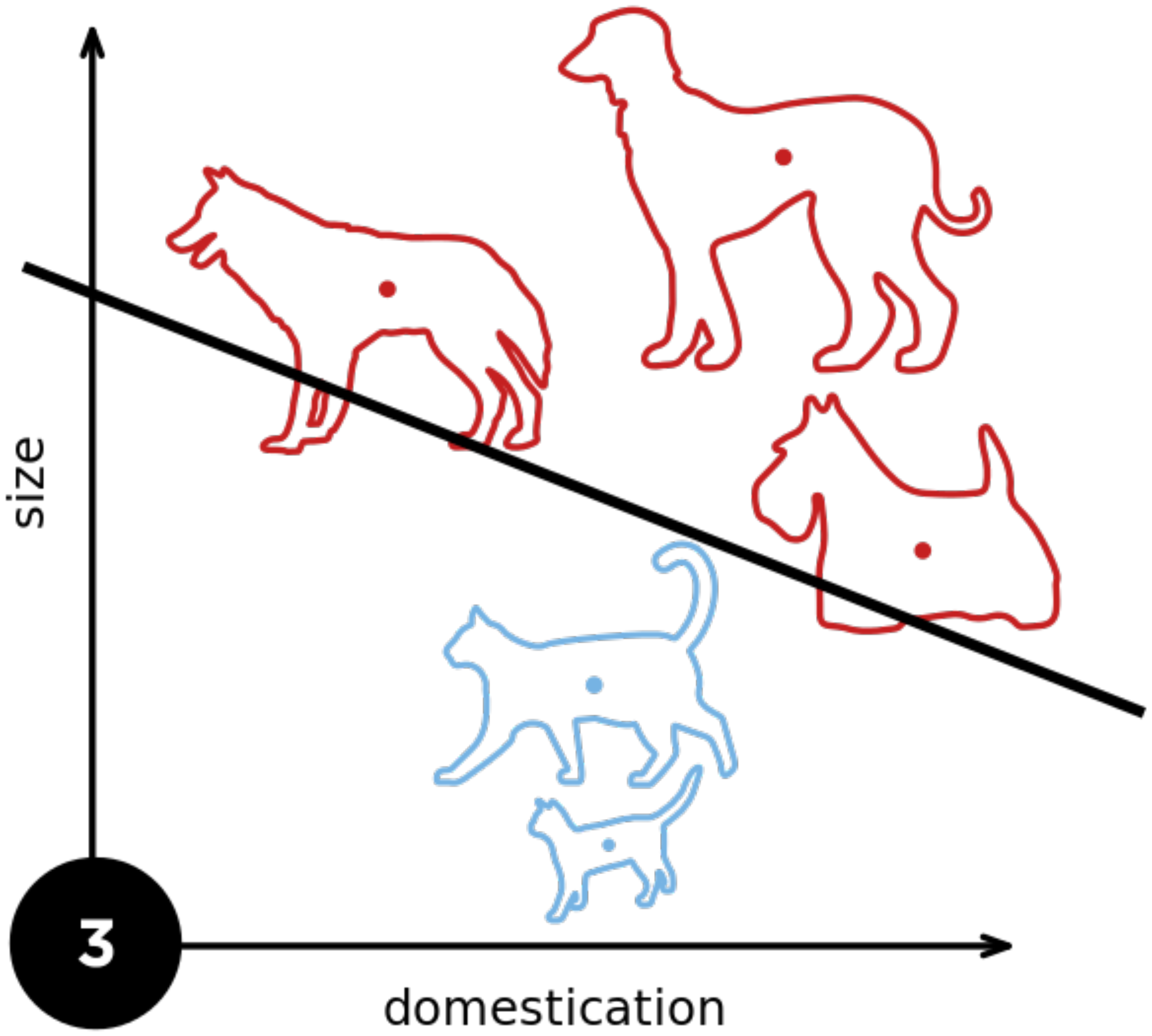
From wikipedia

Perceptron



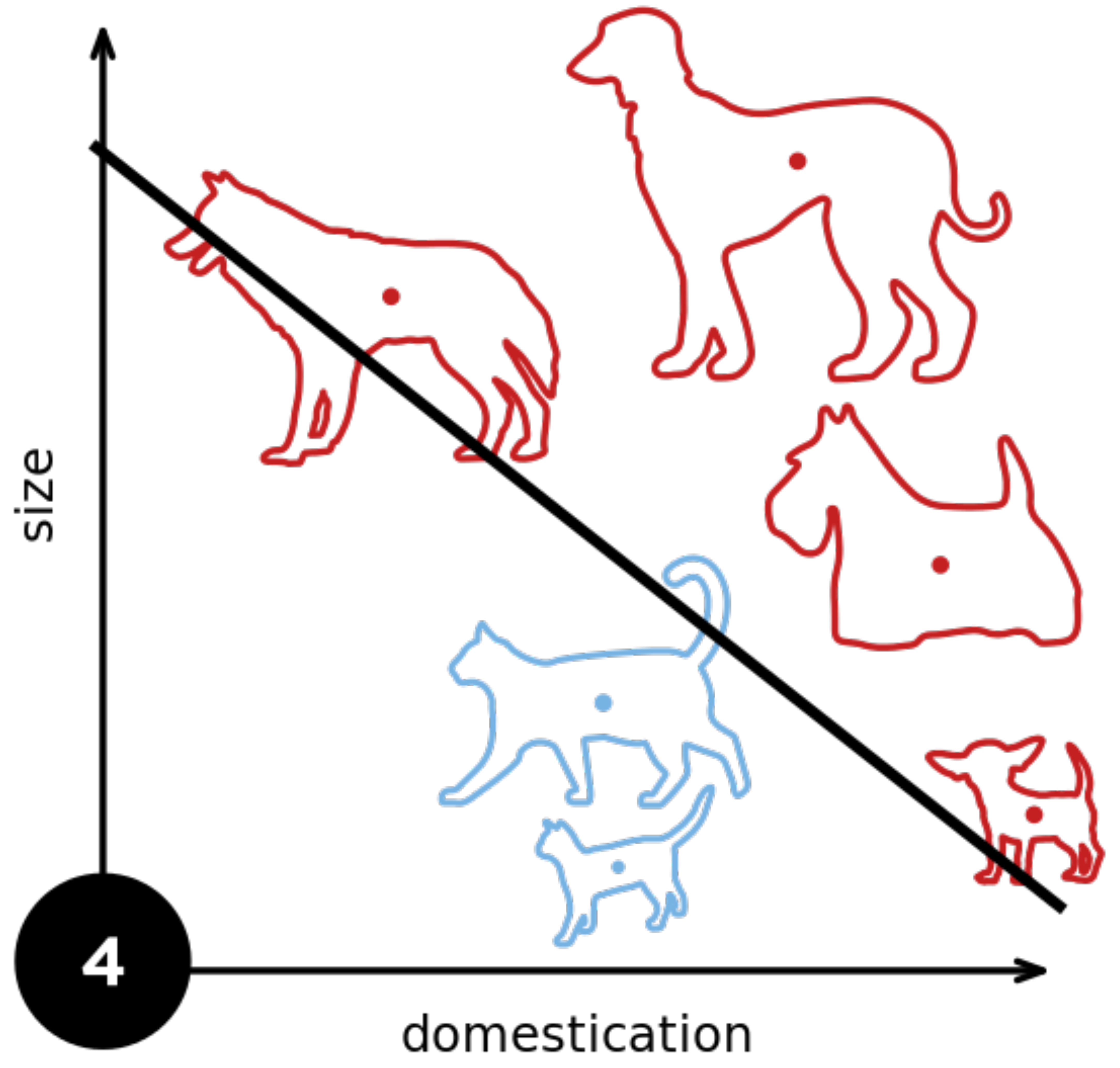
From wikipedia

Perceptron



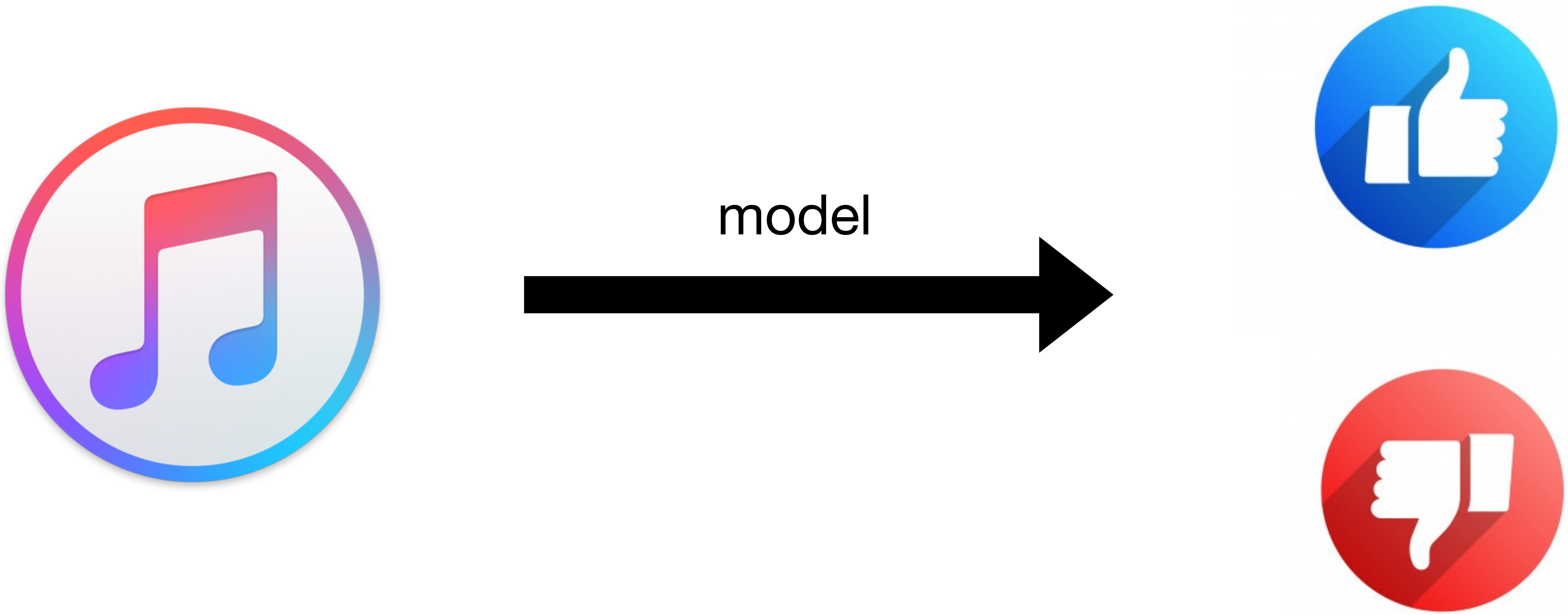
From wikipedia

Perceptron

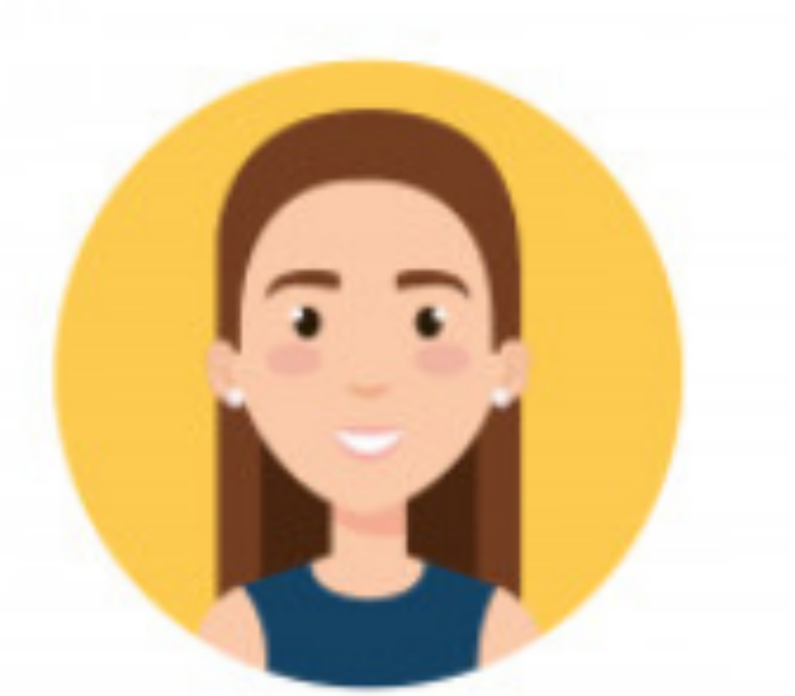


From wikipedia

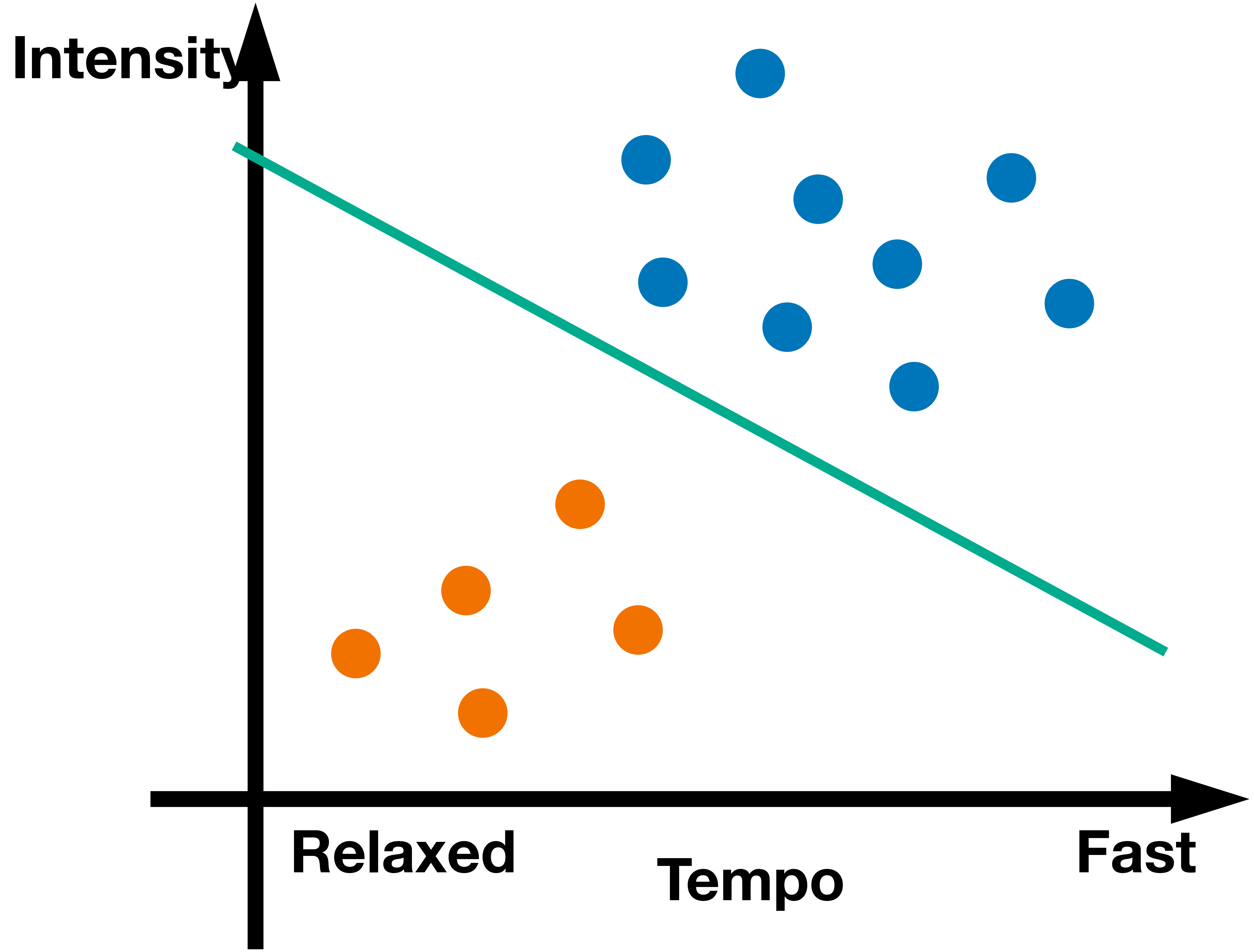
Example 2: Predict whether a user likes a song or not



Example 2: Predict whether a user likes a song or not Using Perceptron



User Sharon



Learning AND function using perceptron

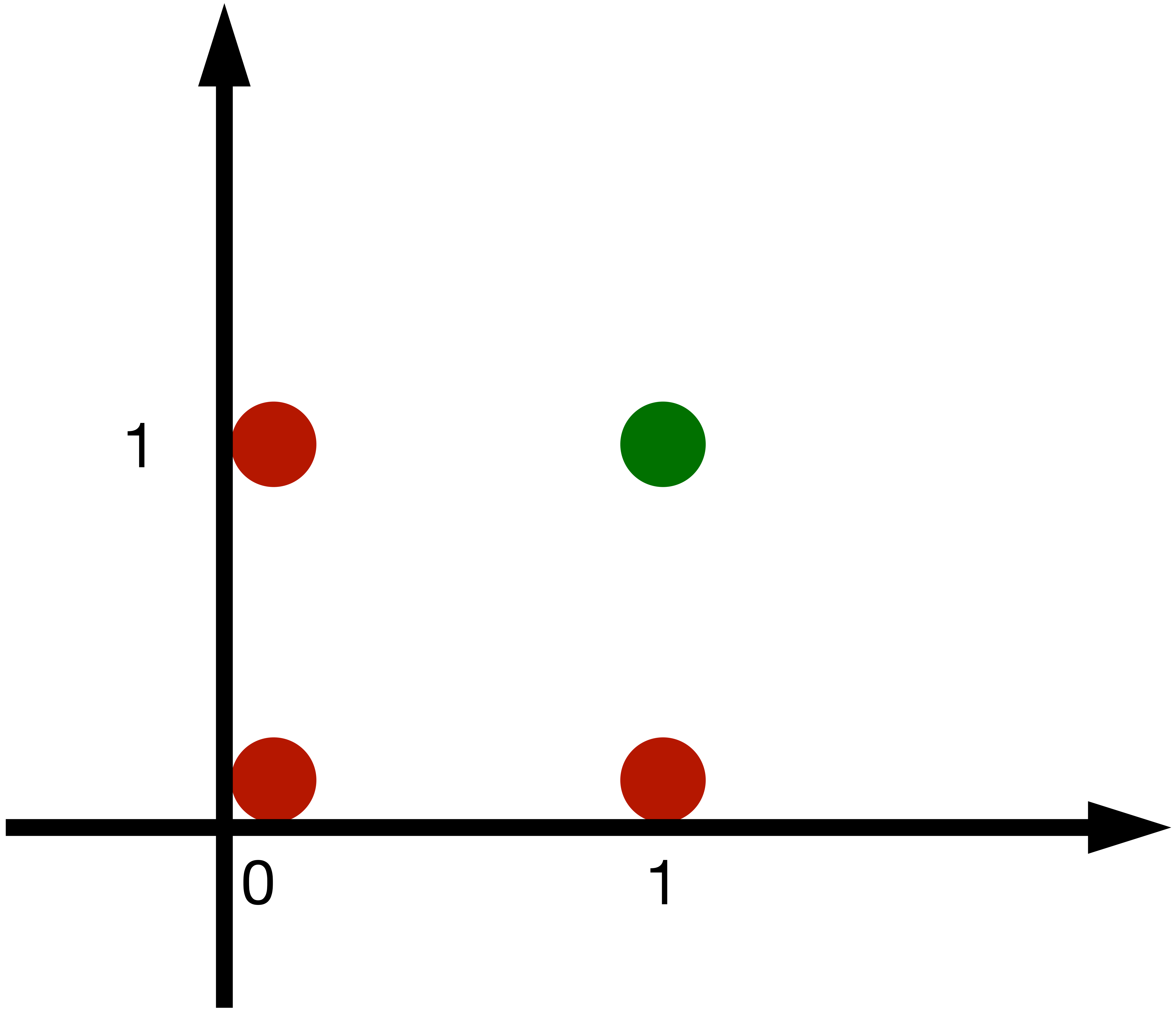
The perceptron can learn an AND function

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

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

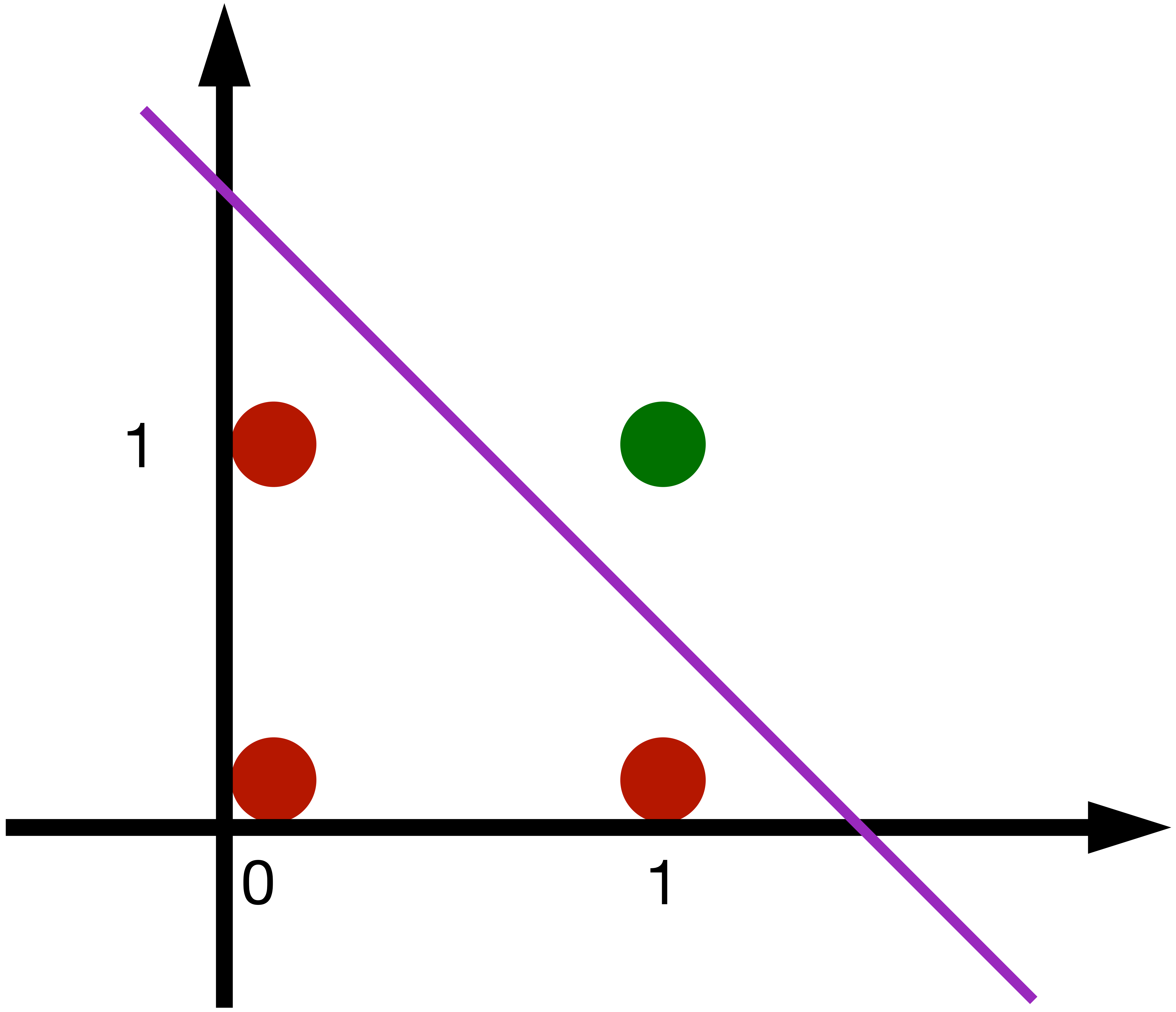
$x_1 = 0, x_2 = 1, y = 0$

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



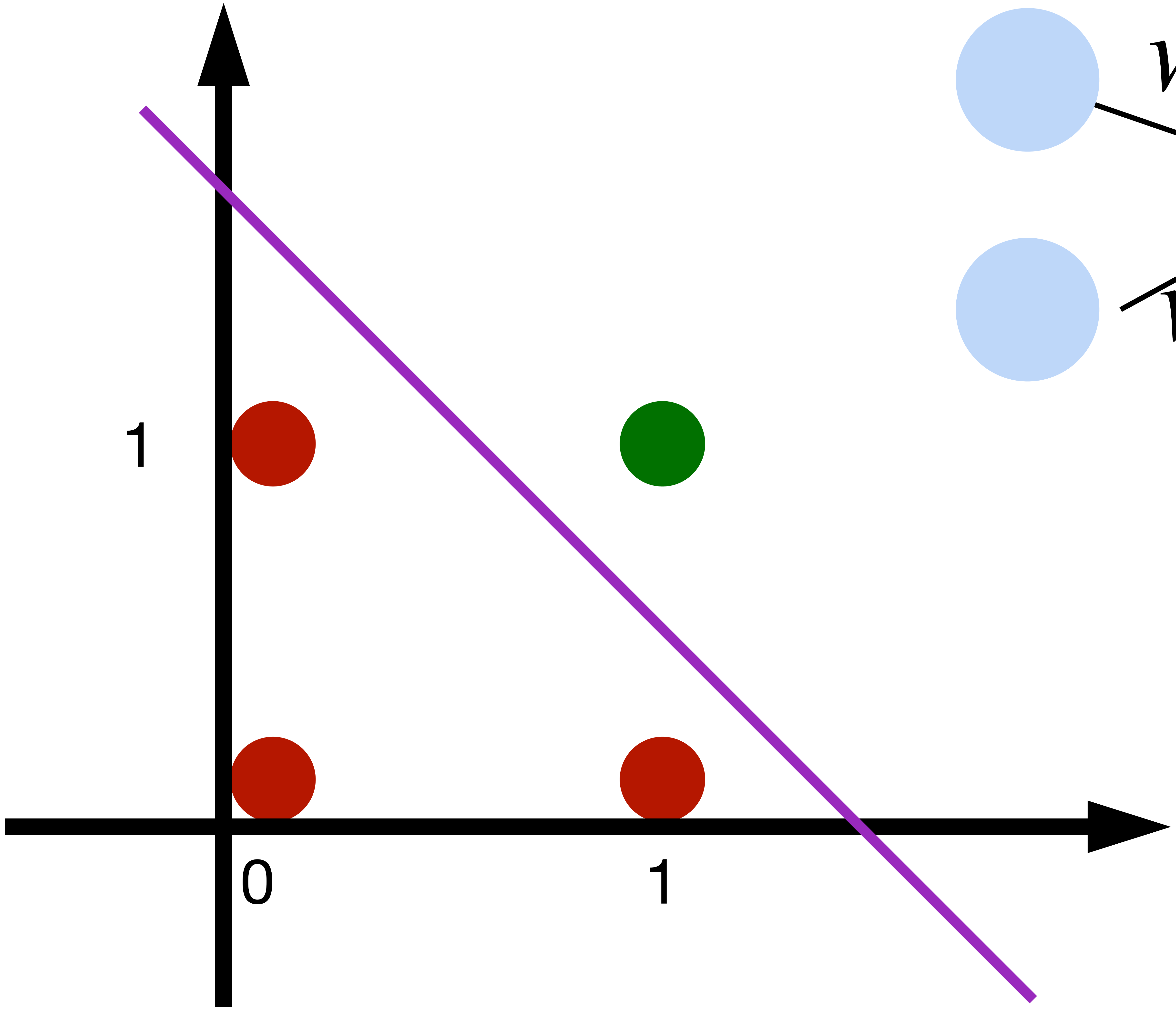
Learning AND function using perceptron

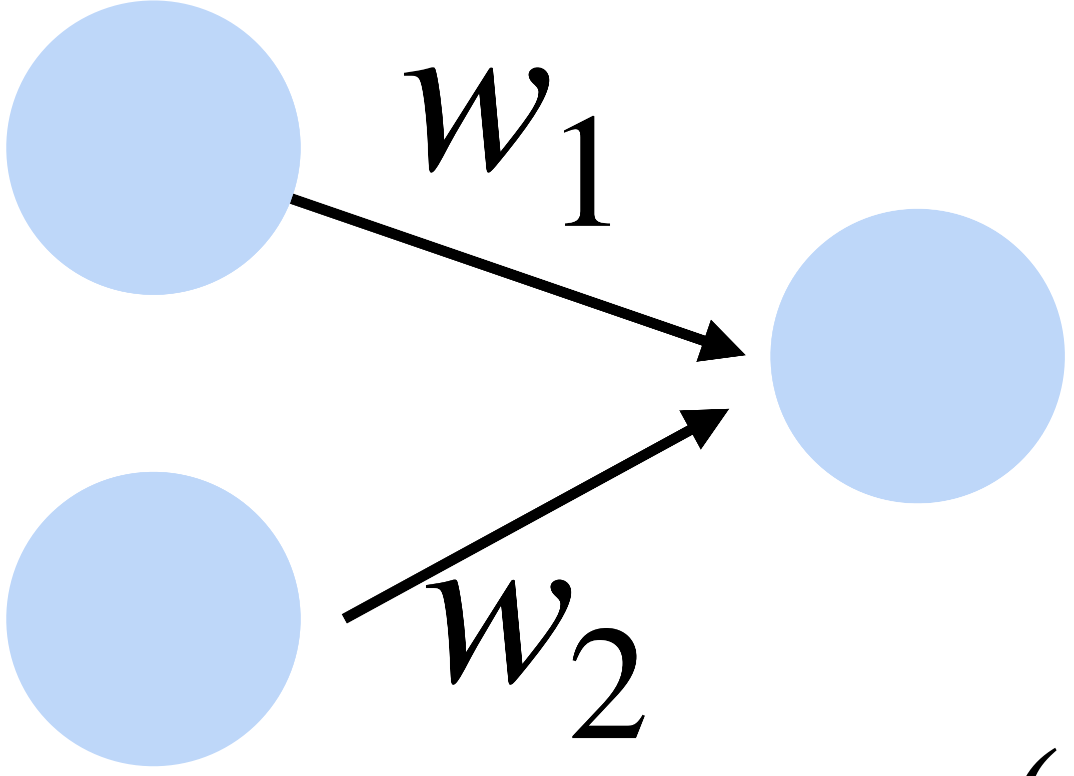
The perceptron can learn an AND function



Learning AND function using perceptron

The perceptron can learn an AND function





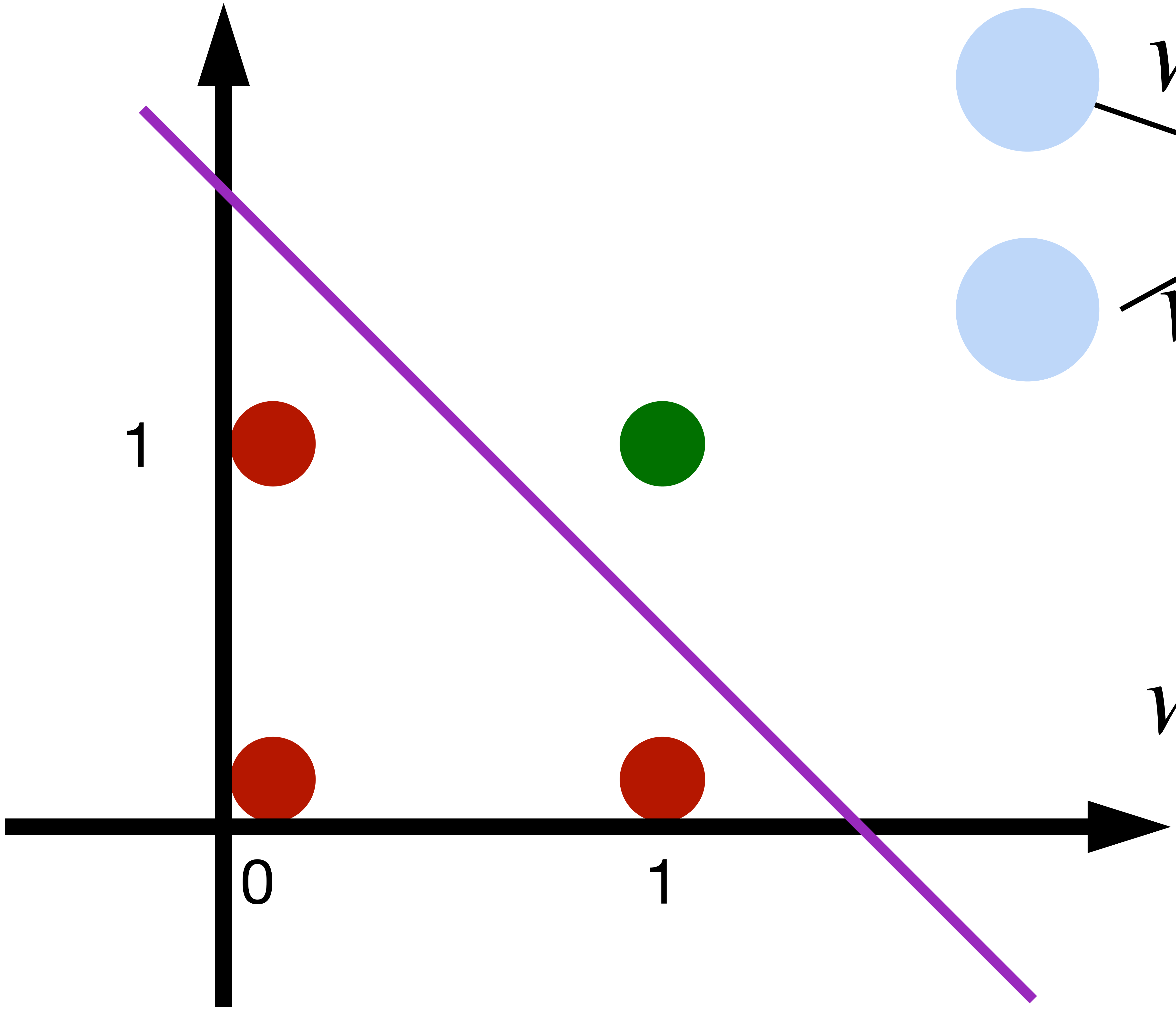
Output $\sigma(x_1 w_1 + x_2 w_2 + b)$

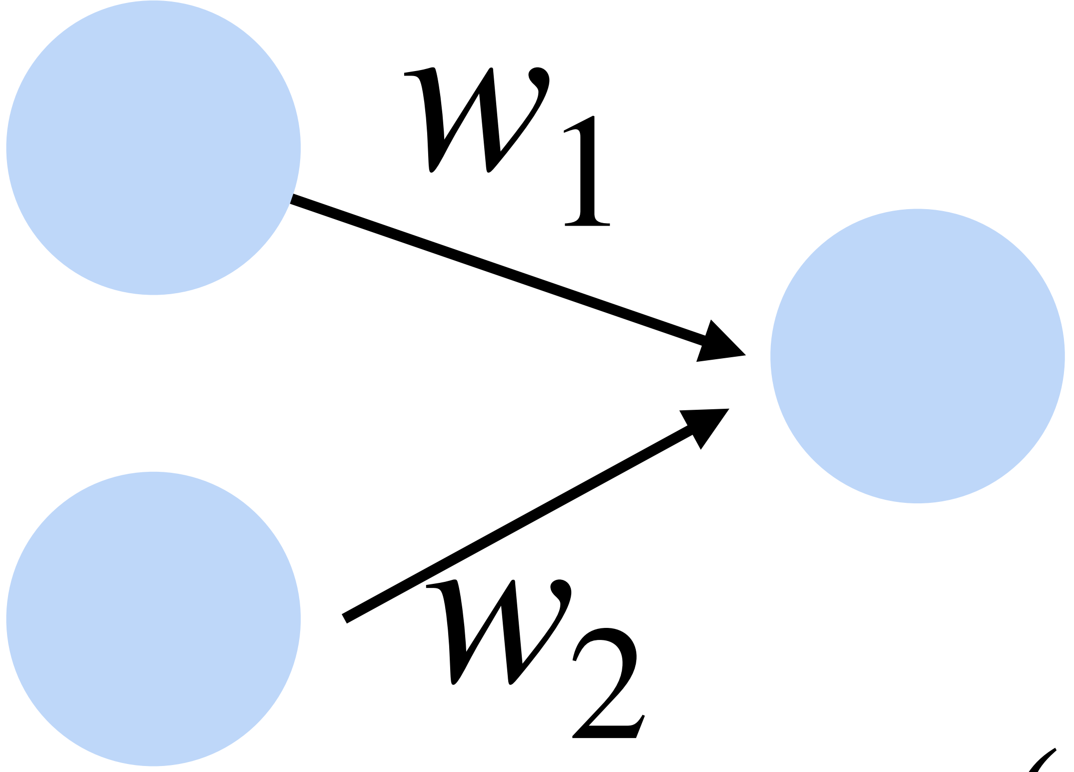
$$\sigma(x) = \begin{cases} 1 & \text{if } x > 0 \\ 0 & \text{otherwise} \end{cases}$$

What's w and b?

Learning AND function using perceptron

The perceptron can learn an AND function





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

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

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

Learning OR function using perceptron

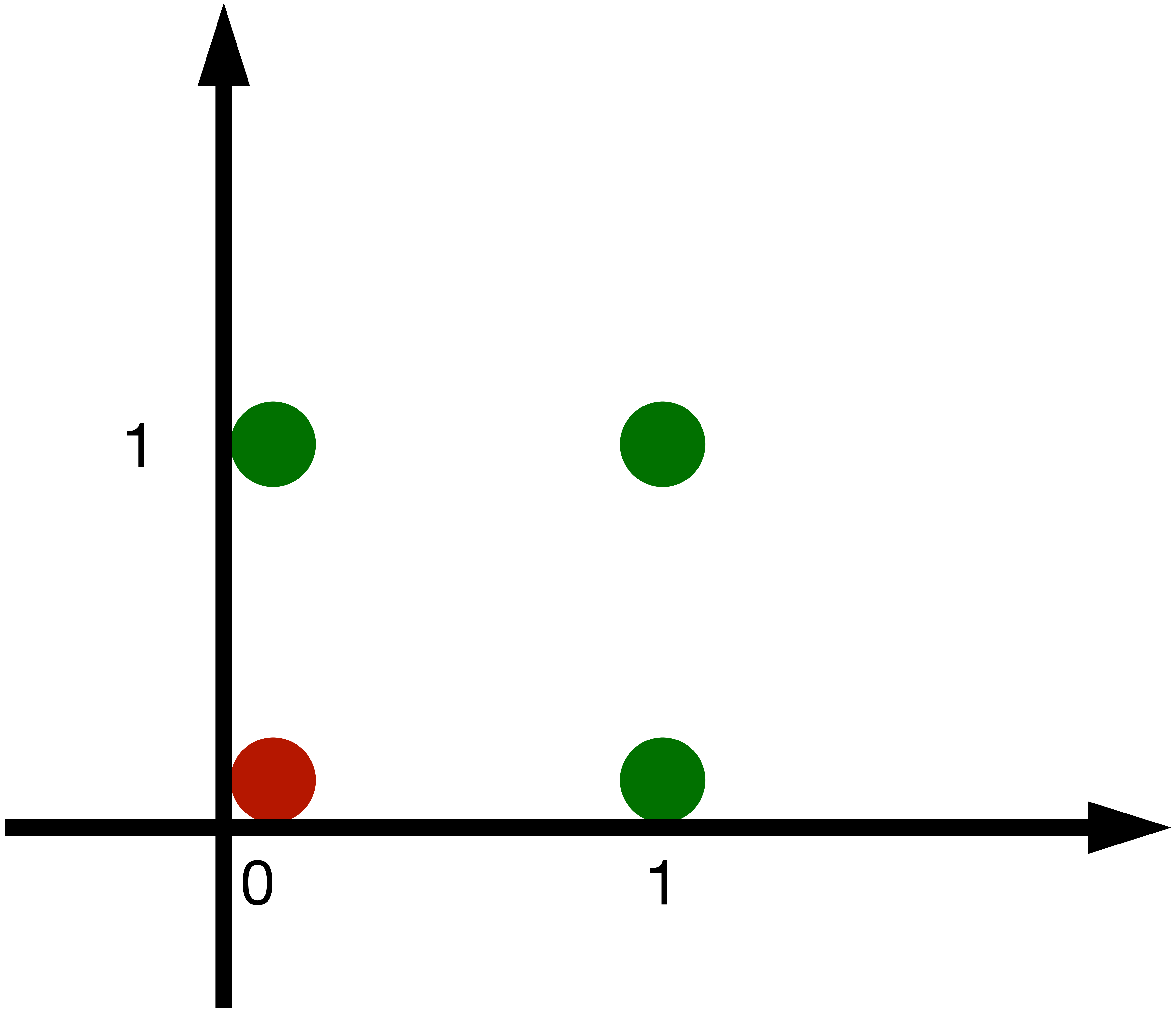
The perceptron can learn an OR function

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

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

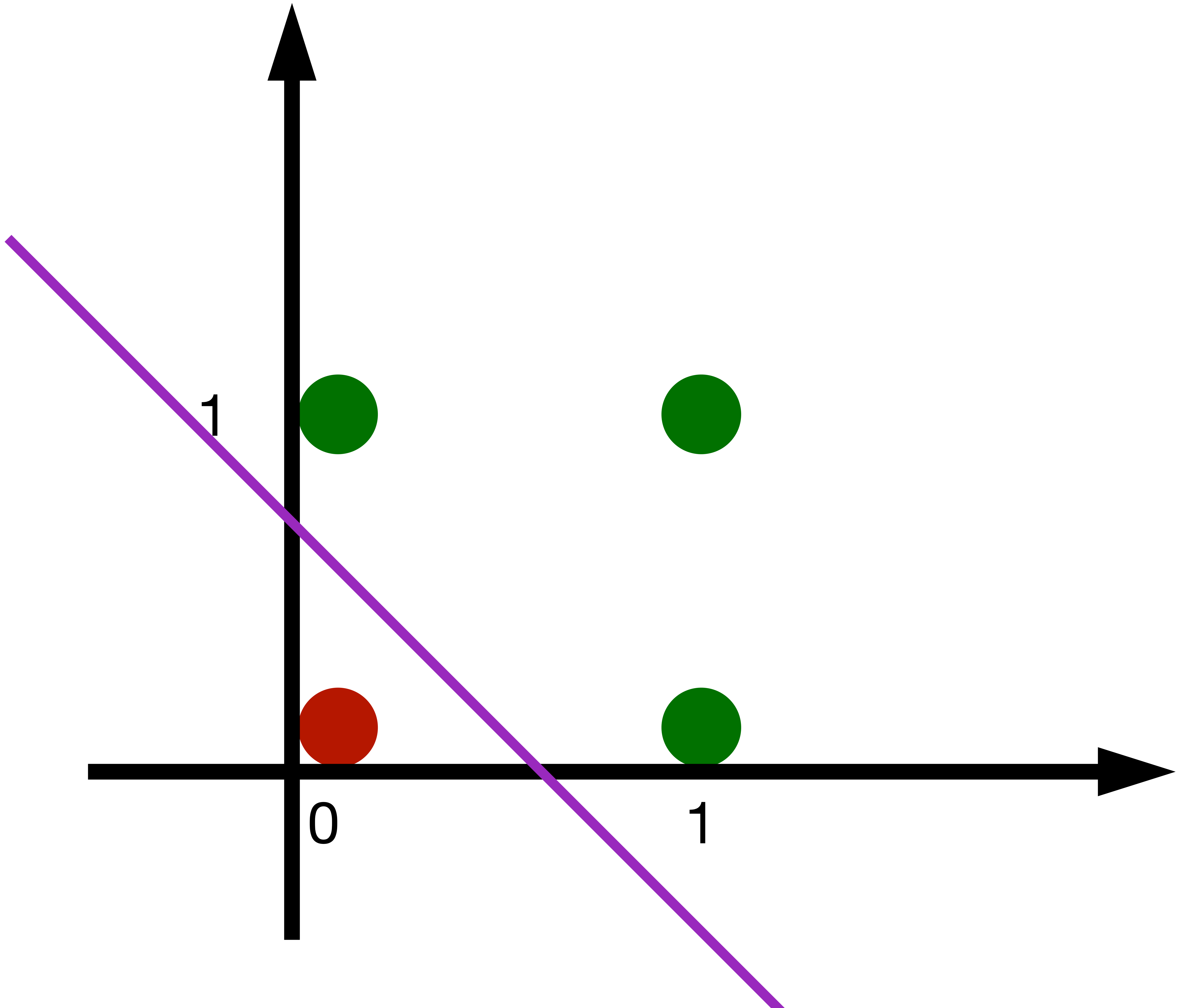
$$x_1 = 0, x_2 = 1, y = 1$$

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



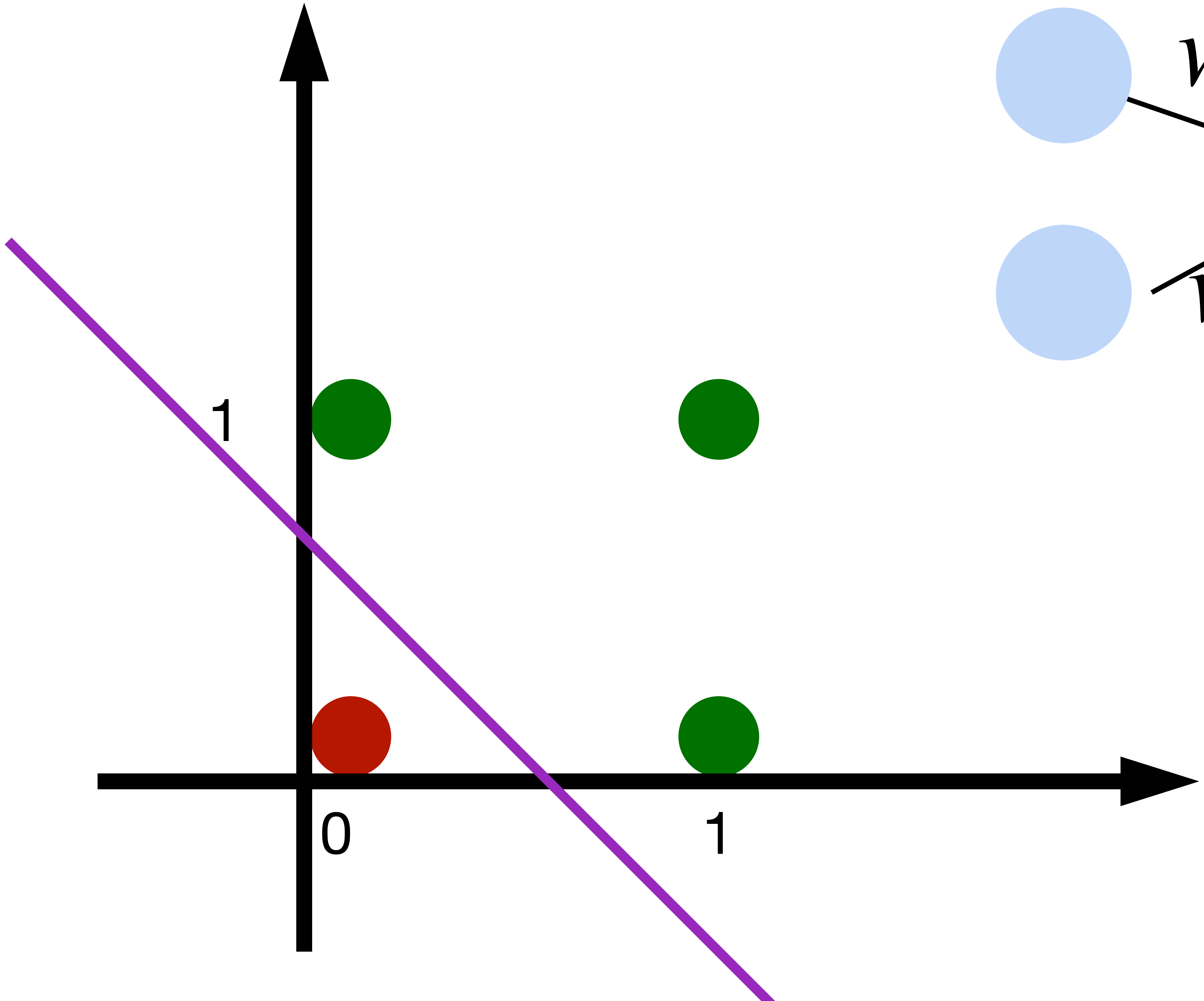
Learning OR function using perceptron

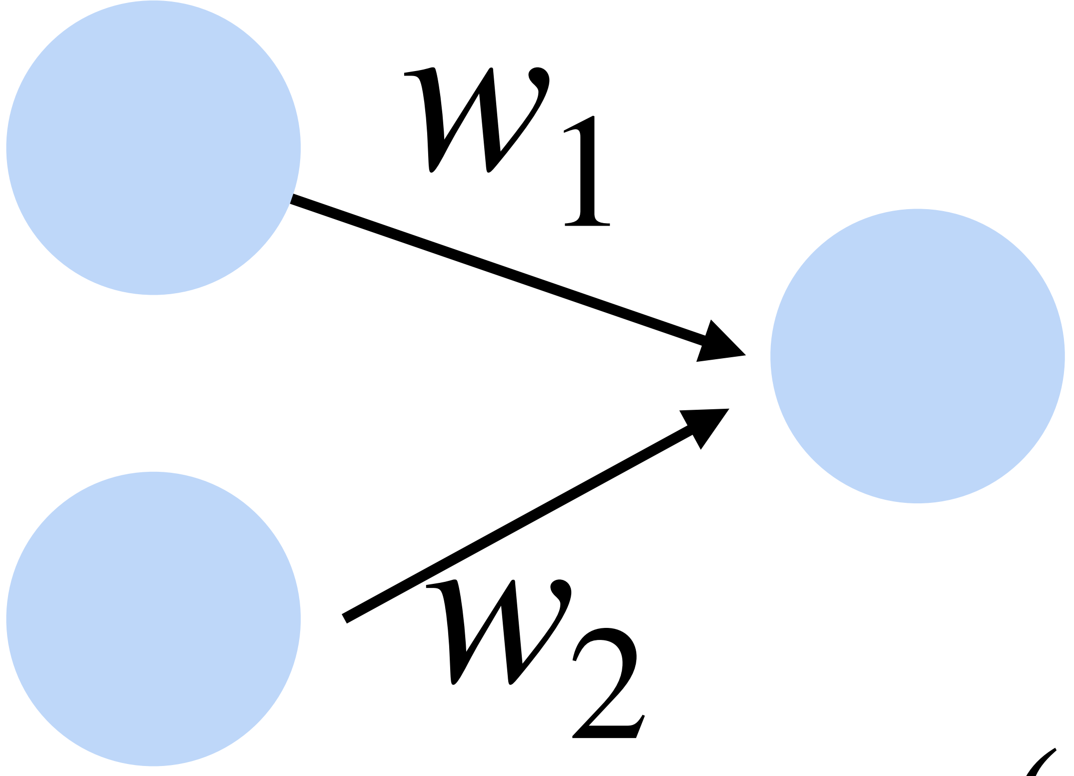
The perceptron can learn an OR function



Learning OR function using perceptron

The perceptron can learn an OR function





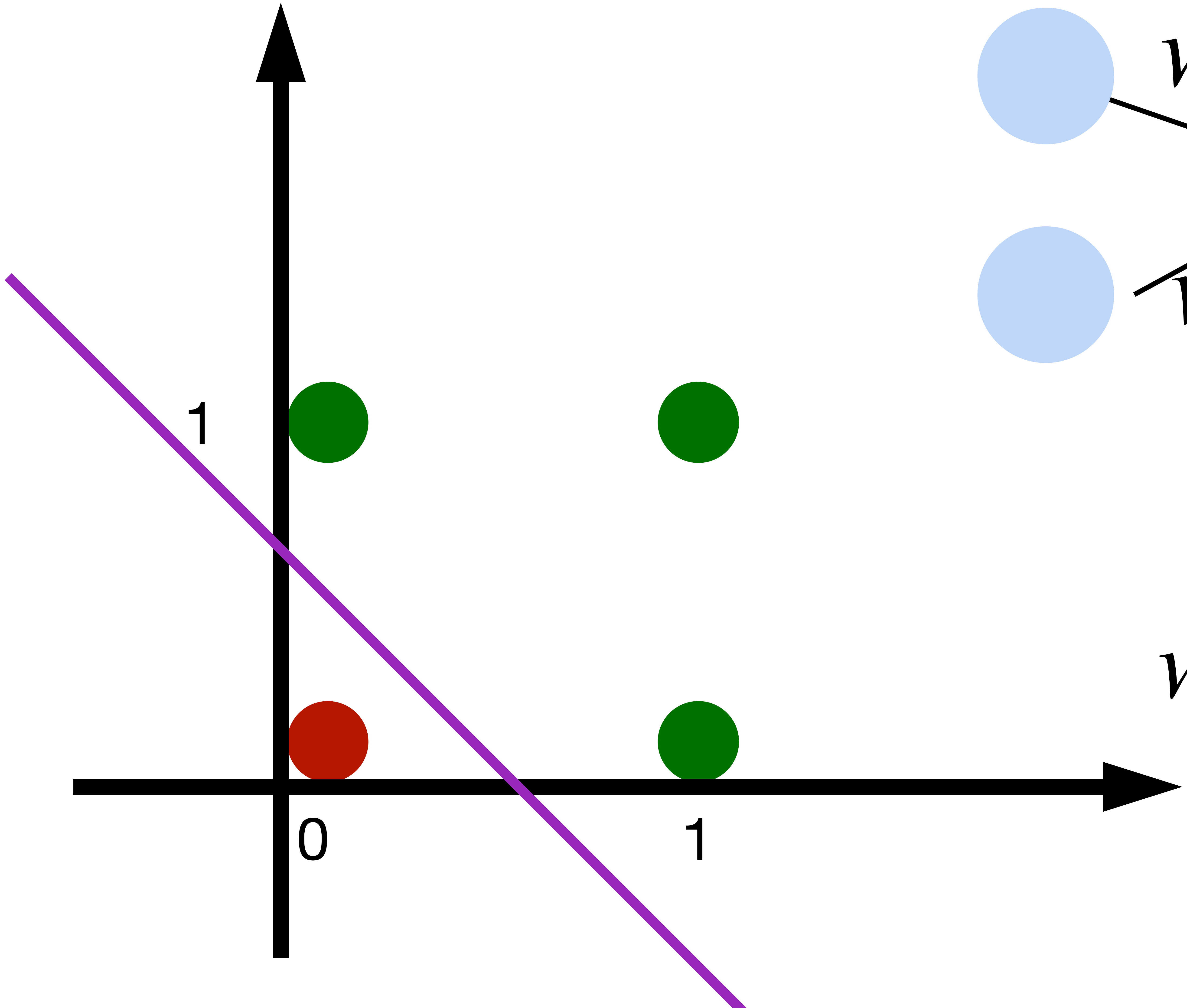
Output $\sigma(x_1 w_1 + x_2 w_2 + b)$

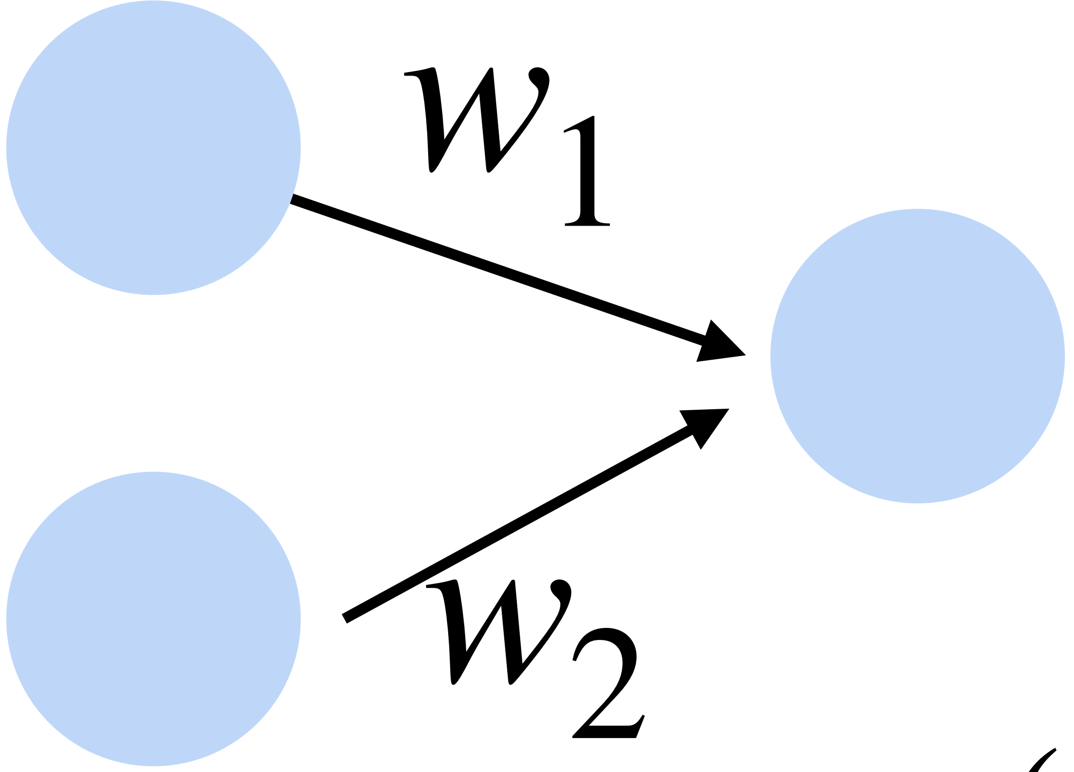
$$\sigma(x) = \begin{cases} 1 & \text{if } x > 0 \\ 0 & \text{otherwise} \end{cases}$$

What's w and b?

Learning OR function using perceptron

The perceptron can learn an OR function





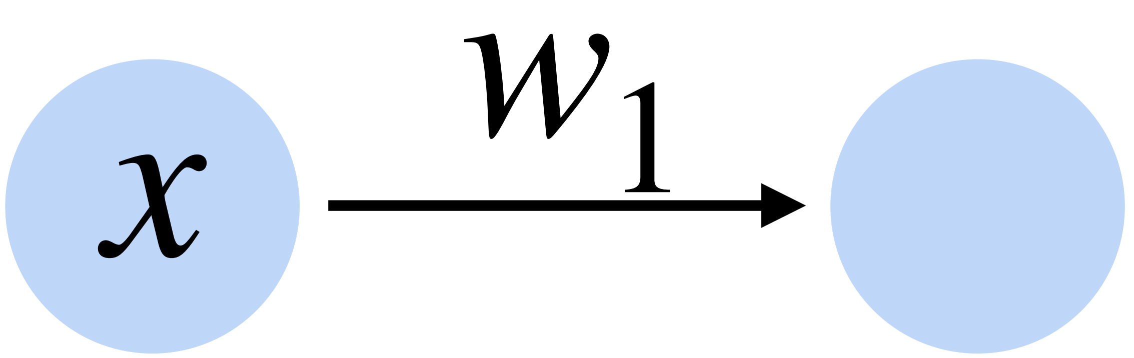
Output $\sigma(x_1 w_1 + x_2 w_2 + b)$

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

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

Learning NOT function using perceptron

The perceptron can learn NOT function (single input)



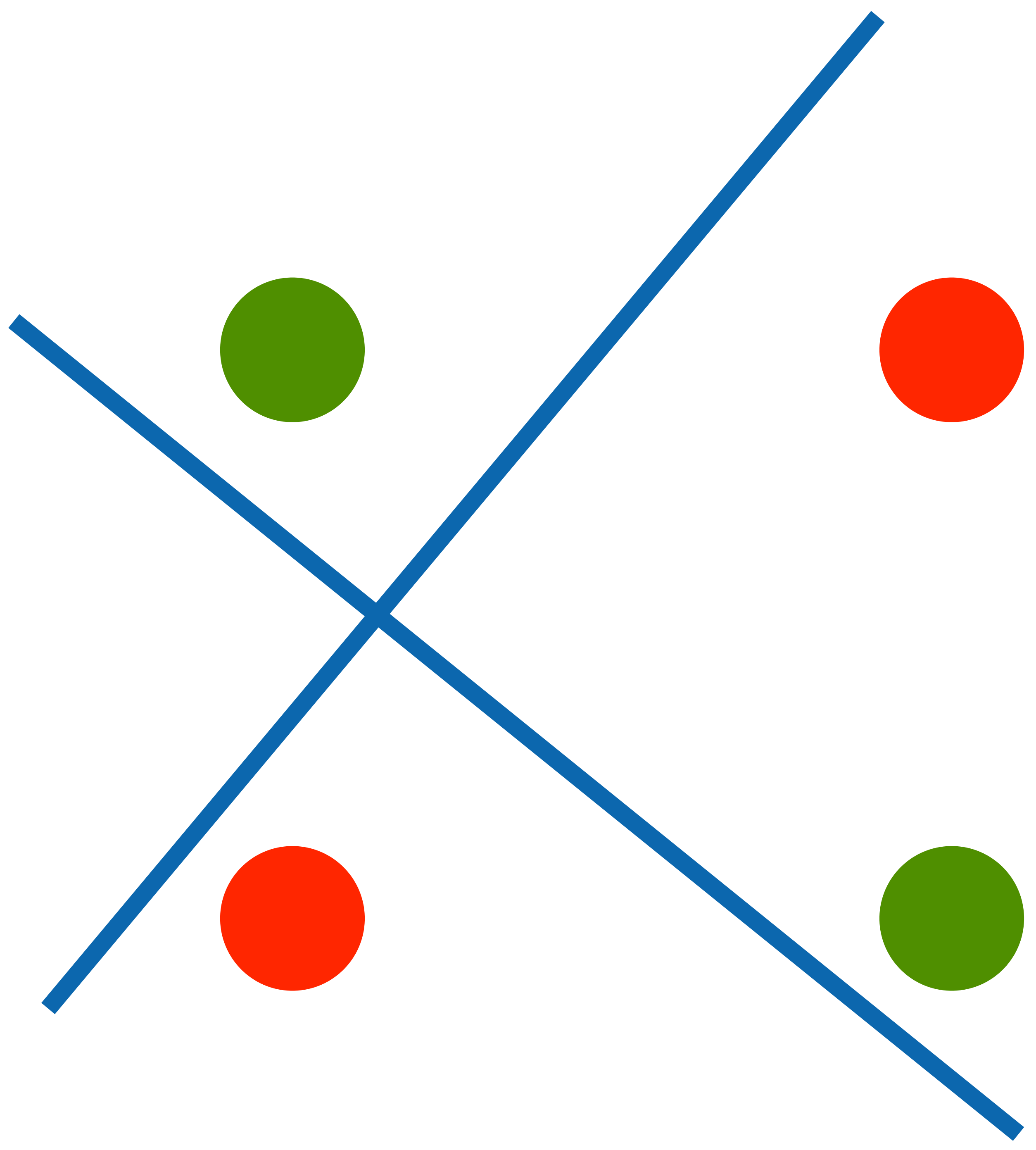
Output $\sigma(xw_1 + b)$

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

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

XOR Problem (Minsky & Papert, 1969)

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

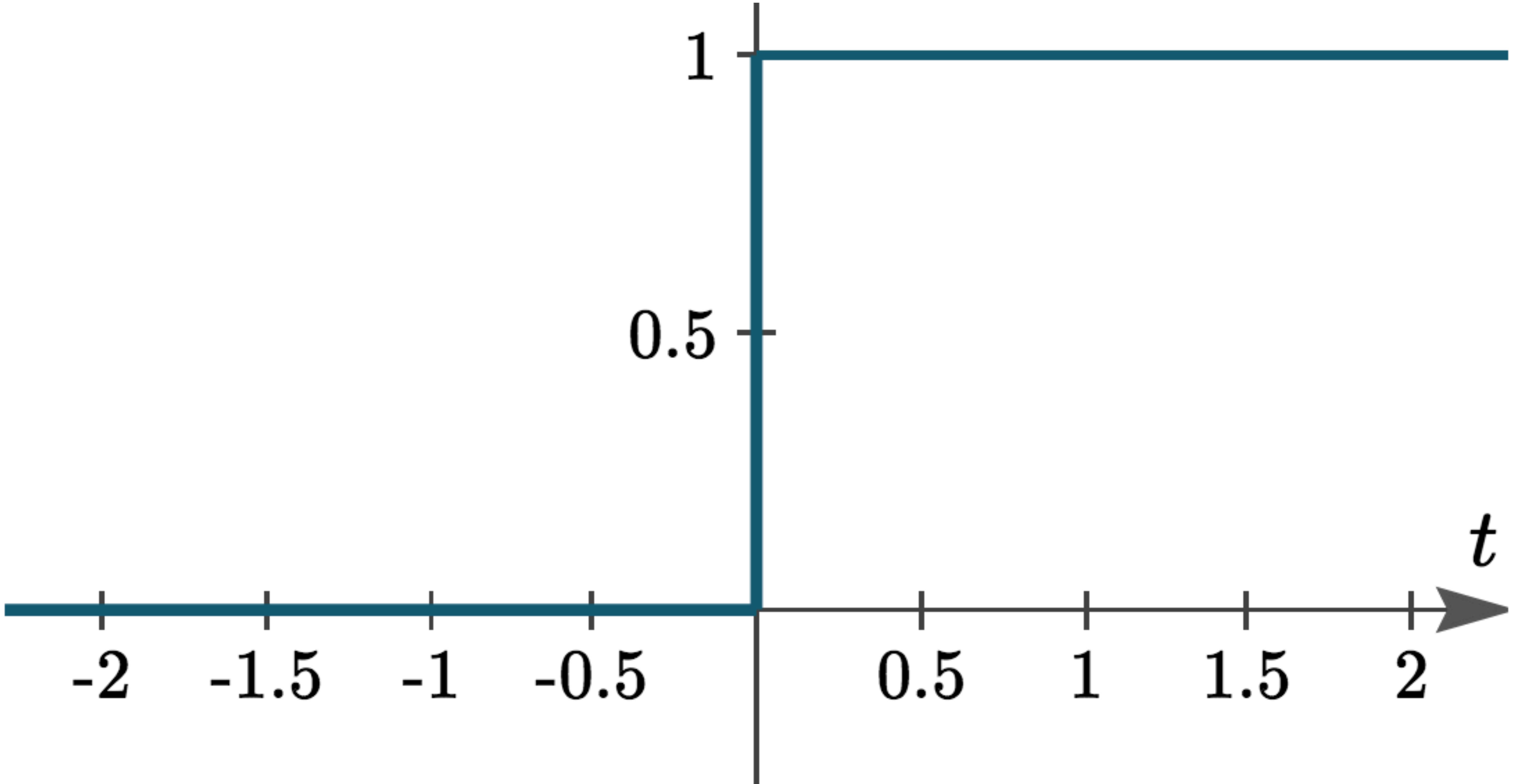


This contributed to the first AI winter

Step Function activation

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

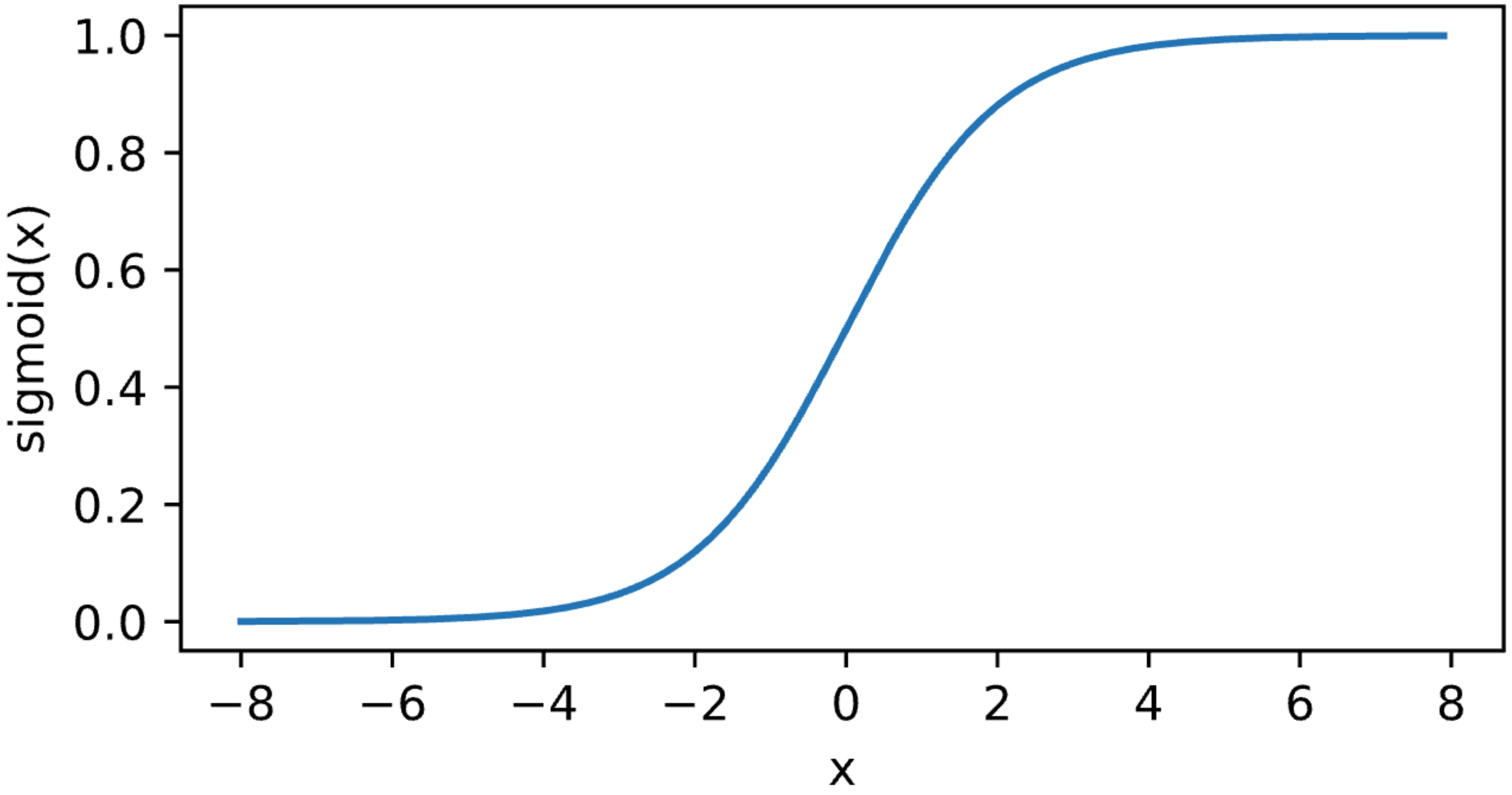
$$\sigma(x) = \begin{cases} 1 & \text{if } x > 0 \\ 0 & \text{otherwise} \end{cases}$$



Sigmoid/Logistic Activation

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

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

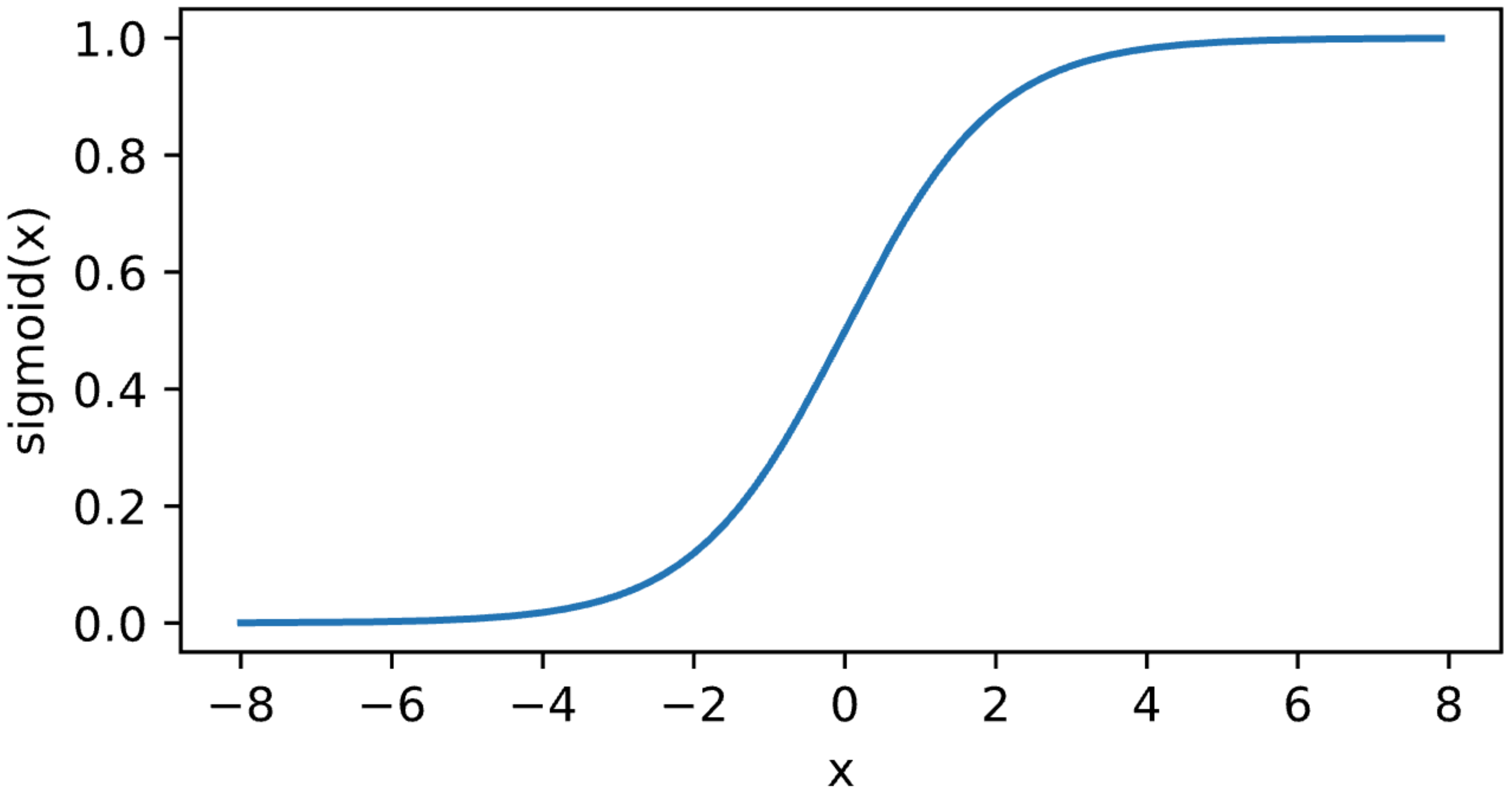


Logistic regression

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

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

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



Logistic regression

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

Training: maximize likelihood estimate (on the conditional probability)

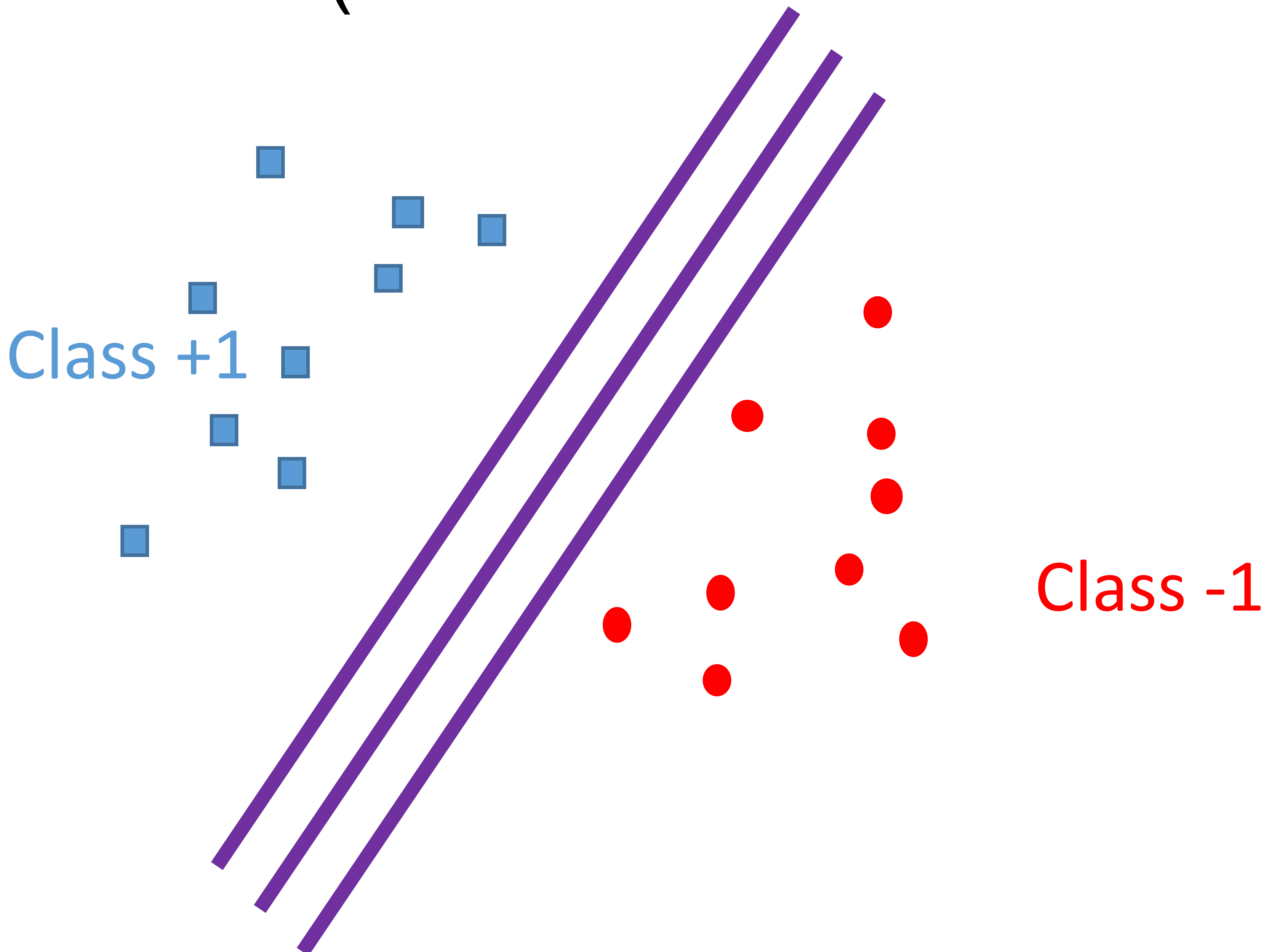
$$\max_{\mathbf{w}} \sum_i \log \frac{1}{1 + \exp(-y_i \mathbf{w}^T \mathbf{x}_i)}$$

Logistic regression

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

Training: maximize likelihood estimate (on the conditional probability)

When training data is linearly separable, many solutions



Logistic regression

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

Training: maximum A posteriori (MAP)

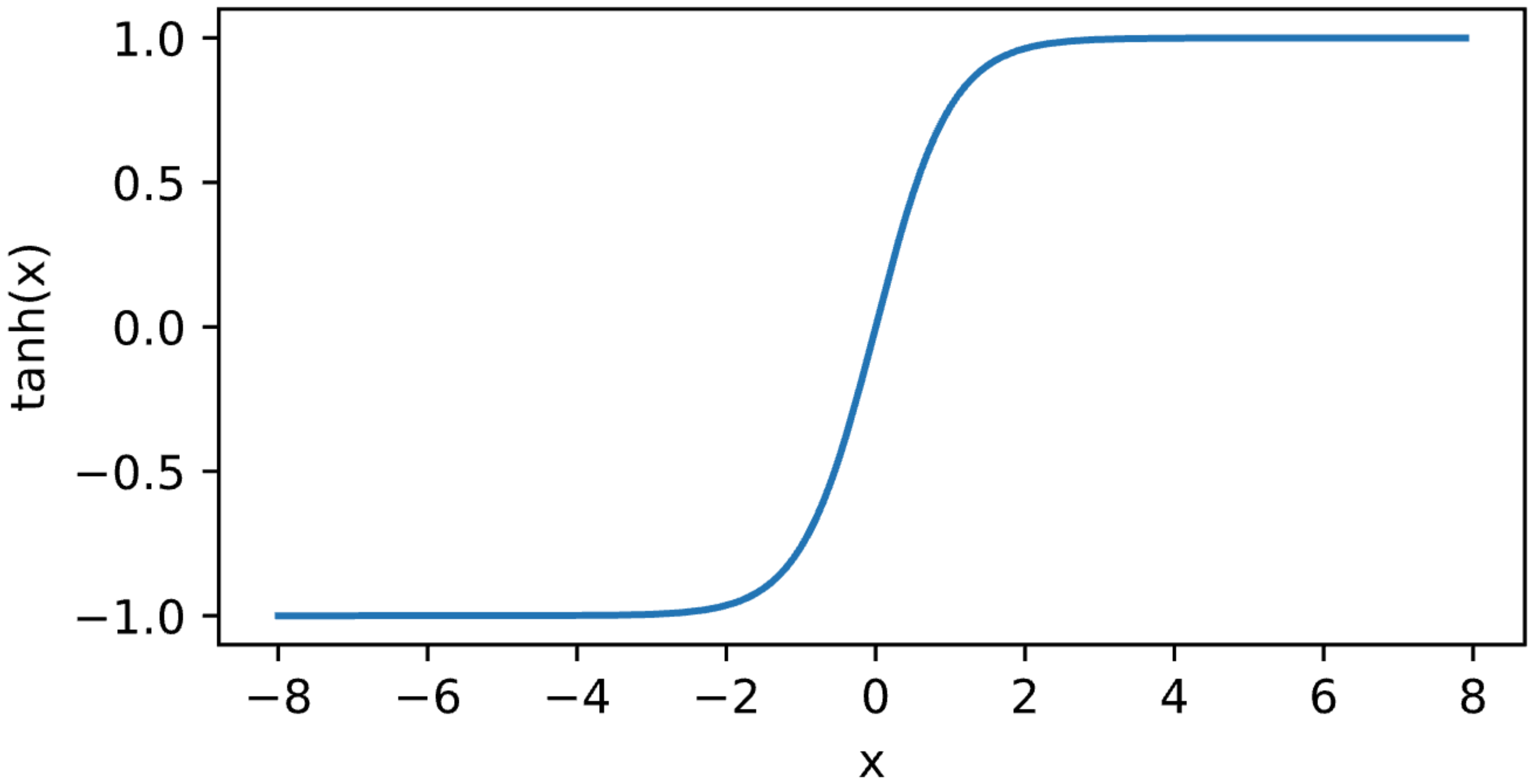
$$\min_{\mathbf{w}} \sum_i -\log \frac{1}{1+\exp(-y_i \mathbf{w}^T \mathbf{x}_i)} + \frac{\lambda}{2}\|\mathbf{w}\|_2^2$$

- Convex optimization
- Solve via (stochastic) gradient descent

Tanh Activation

Map inputs into (-1, 1)

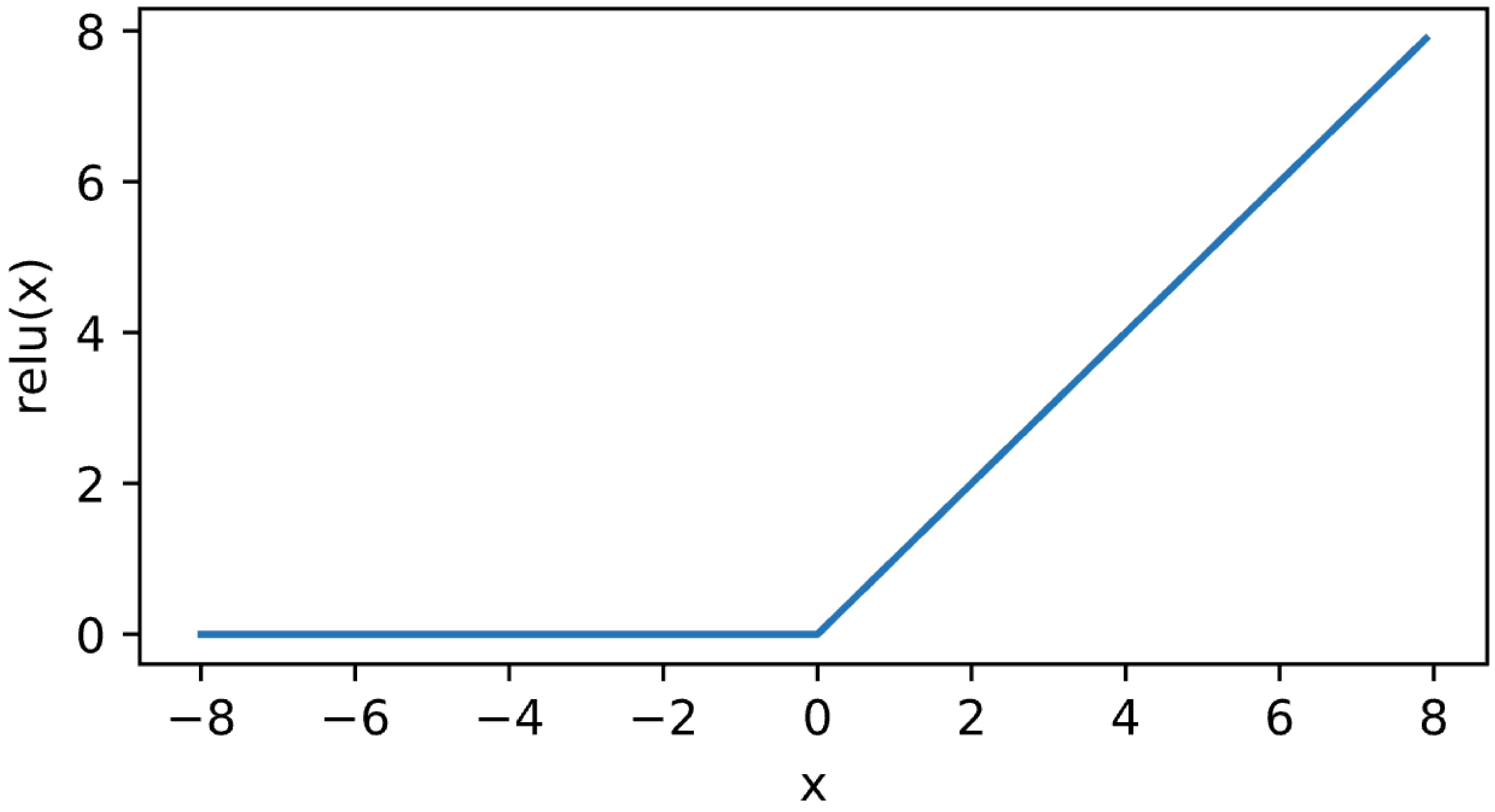
$$\tanh(x) = \frac{1 - \exp(-2x)}{1 + \exp(-2x)}$$



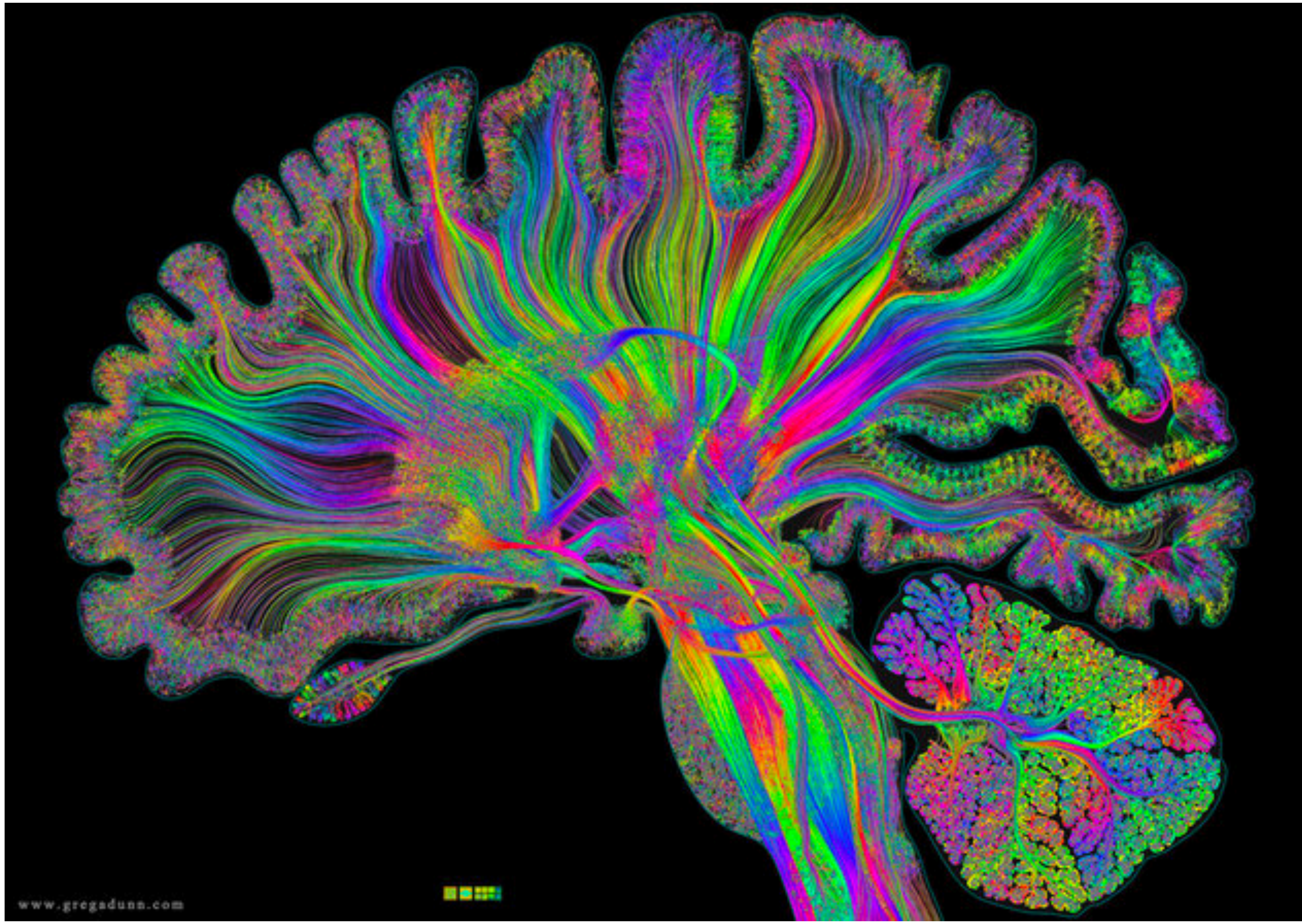
ReLU Activation

ReLU: rectified linear unit (commonly used in modern neural networks)

$$\text{ReLU}(x) = \max(x, 0)$$



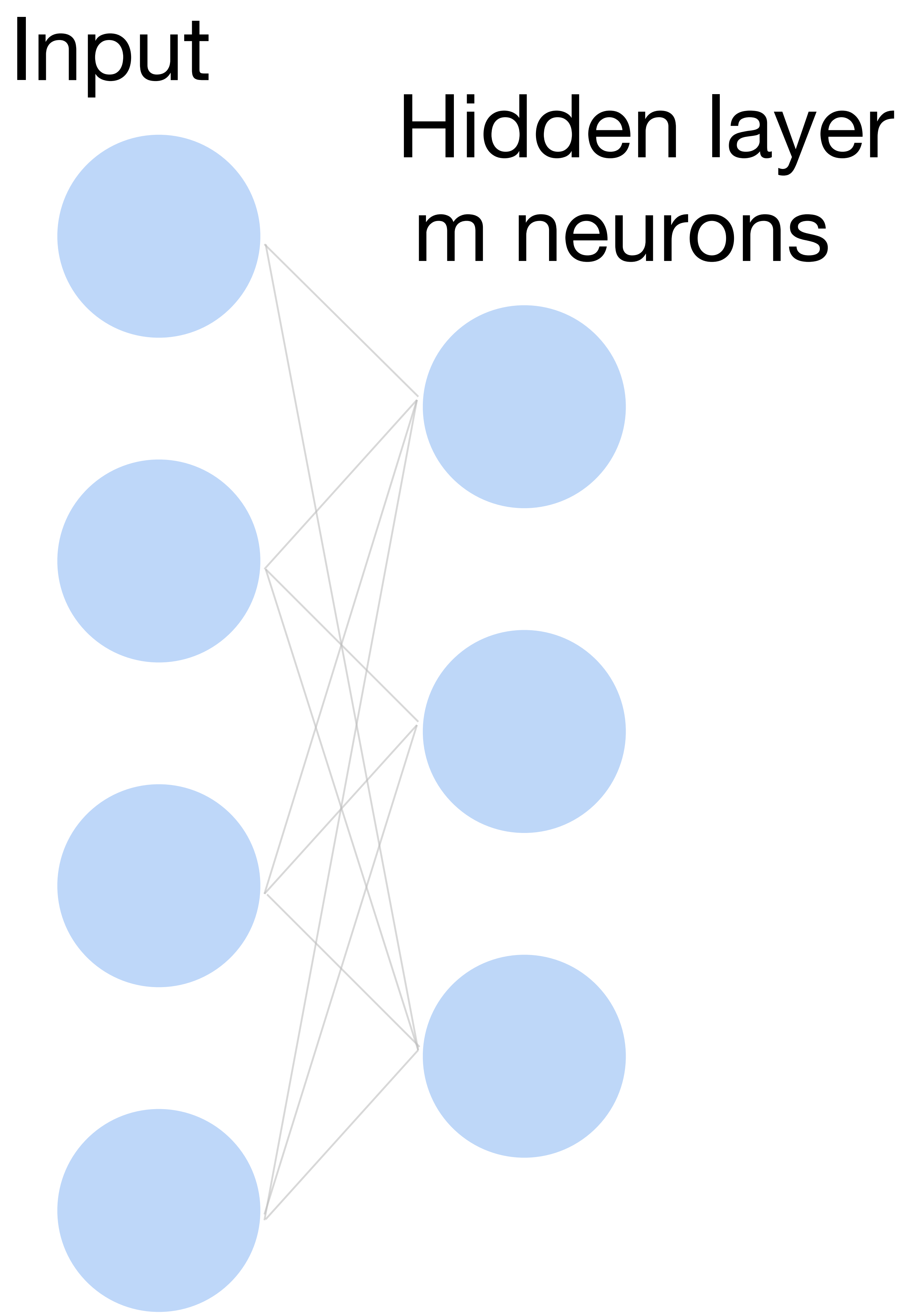
Multilayer Perceptron



Single Hidden Layer

How to classify

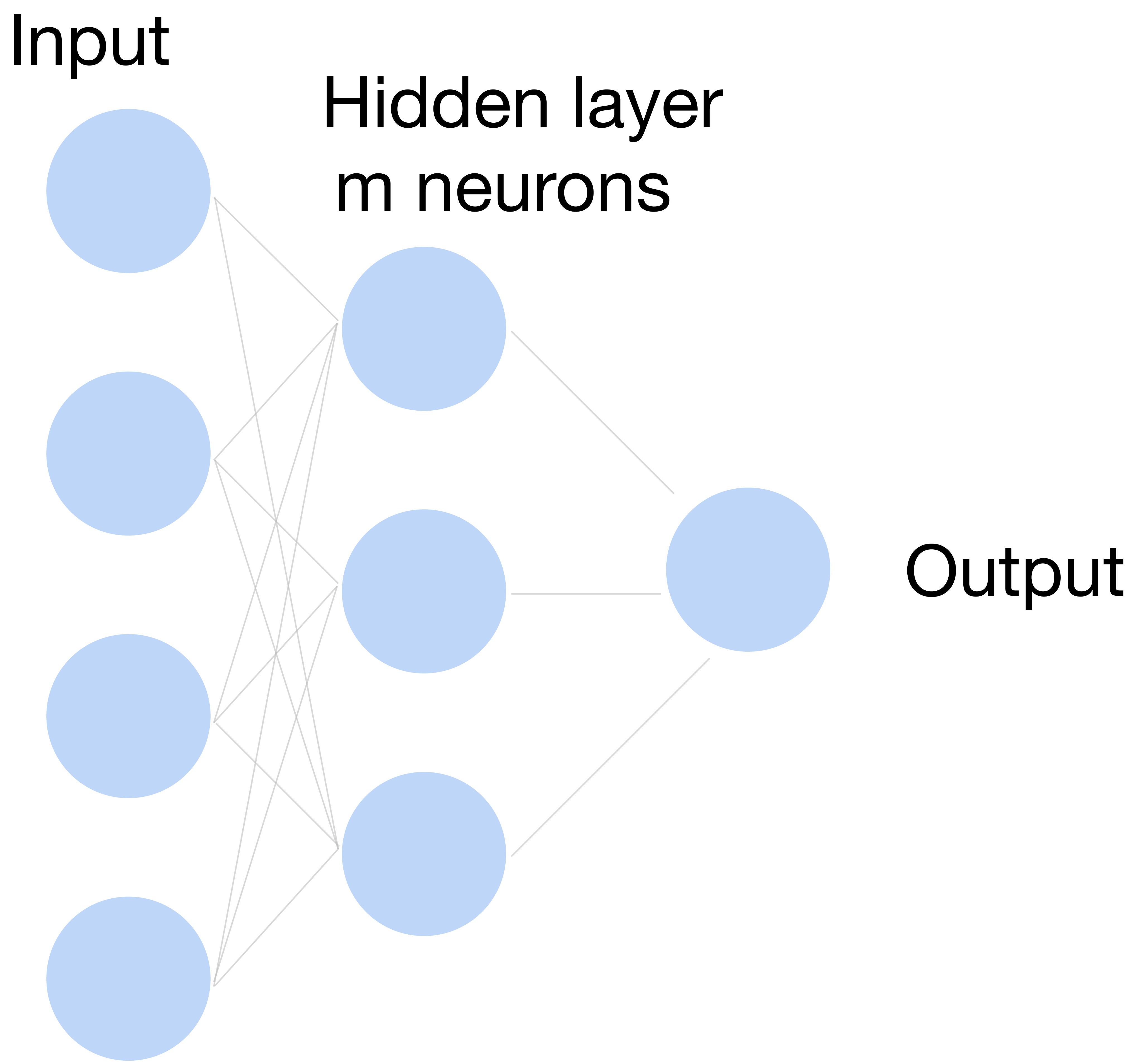
Cats vs. dogs?



Single Hidden Layer

How to classify

Cats vs. dogs?

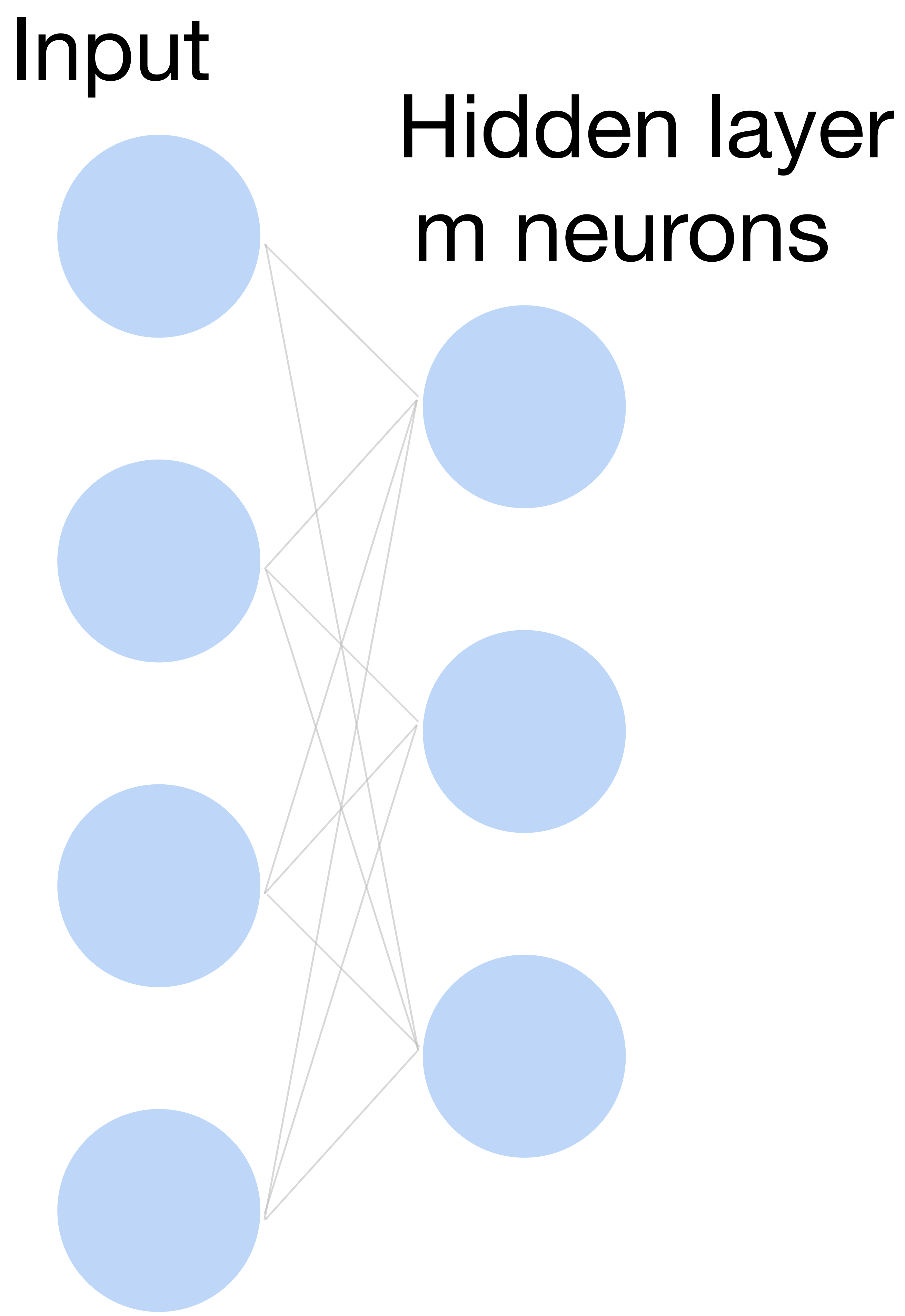


Single Hidden Layer

- Input $\mathbf{x} \in \mathbb{R}^d$
- Hidden $\mathbf{W} \in \mathbb{R}^{m \times d}, \mathbf{b} \in \mathbb{R}^m$
- Intermediate output

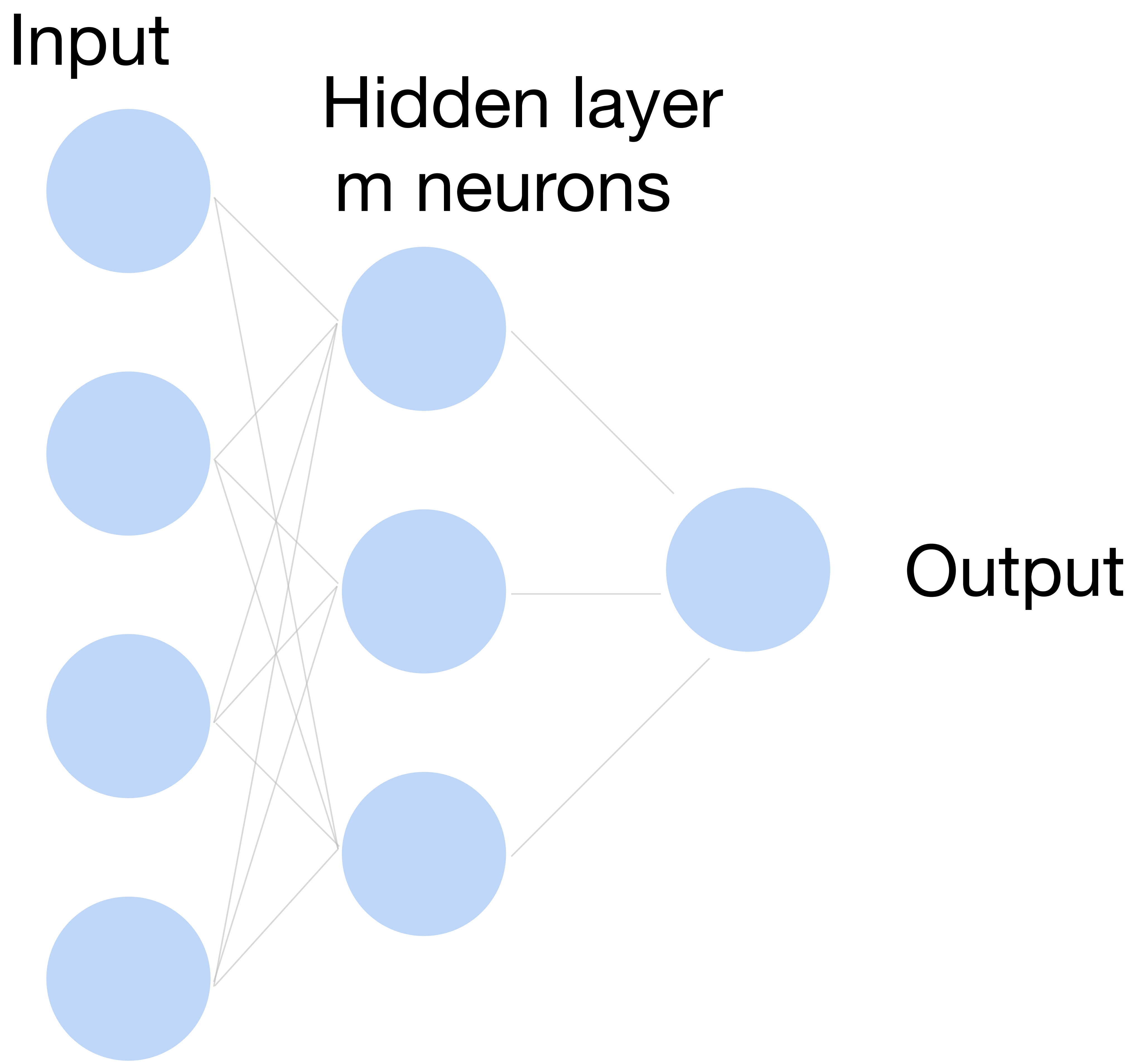
$$\mathbf{h} = \sigma(\mathbf{W}\mathbf{x} + \mathbf{b})$$

σ is an element-wise activation function

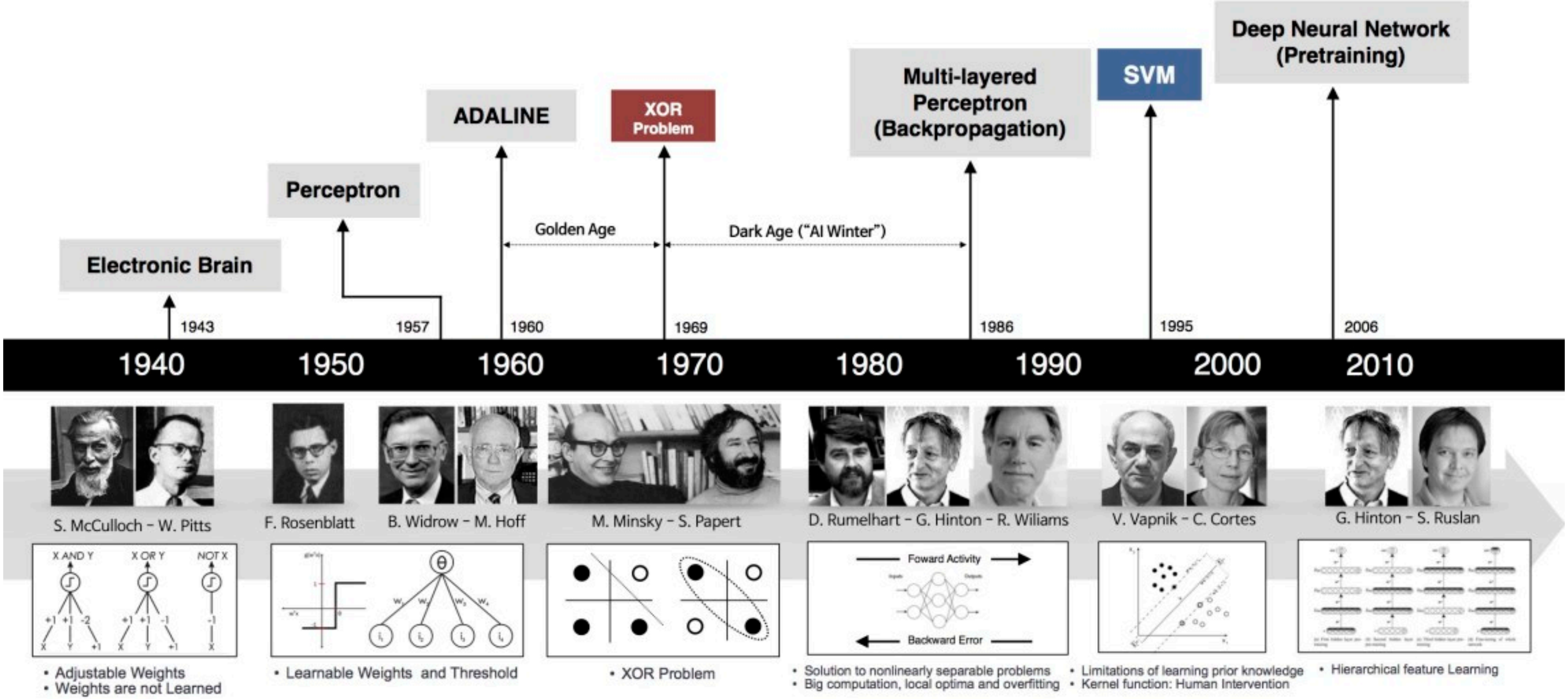


Single Hidden Layer

- Output $\mathbf{f} = \mathbf{w}_2^\top \mathbf{h} + b_2$



Brief history of neural networks



What we've learned today…

- Single-layer Perceptron
  - Motivation
  - Activation function
  - Representing AND, OR, NOT
- Brief history of neural networks

Thanks!

Based on slides from Xiaojin (Jerry) Zhu and Yingyu Liang (http://pages.cs.wisc.edu/~jerryzhu/cs540.html), and Alex Smola: https://courses.d2l.ai/berkeley-stat-157/units/mlp.html