IT5005
Introduction to Neural Networks

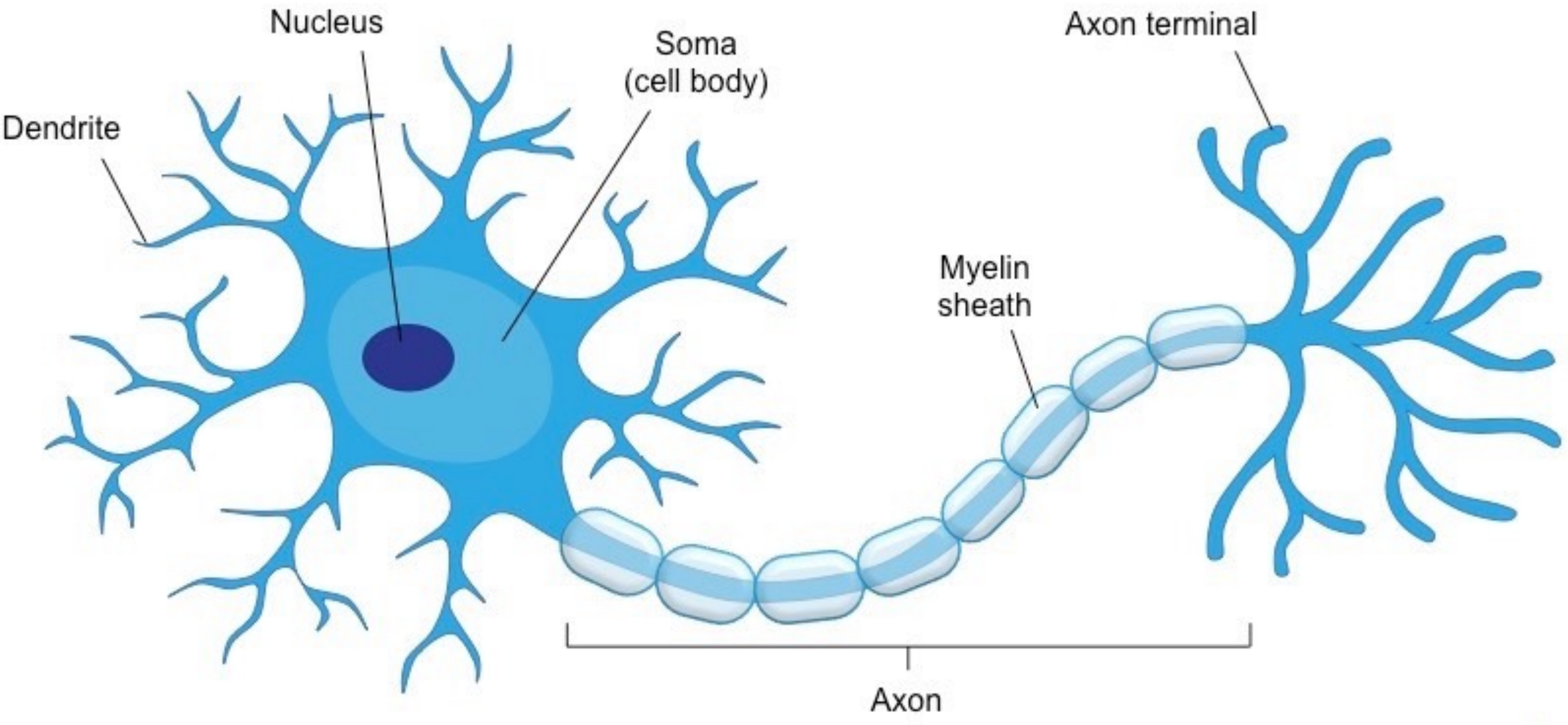
Slide Credit: Prof. Ben Leong

Agenda

- Perceptron
- Perceptron Learning Algorithm
- Linear and Logistic Regression
- Logic gates (NOT, AND, OR, XOR)

Why are neural networks called neural networks?

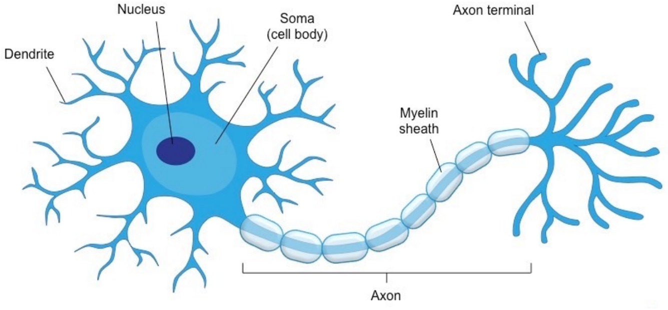
Inspired by how the brain works

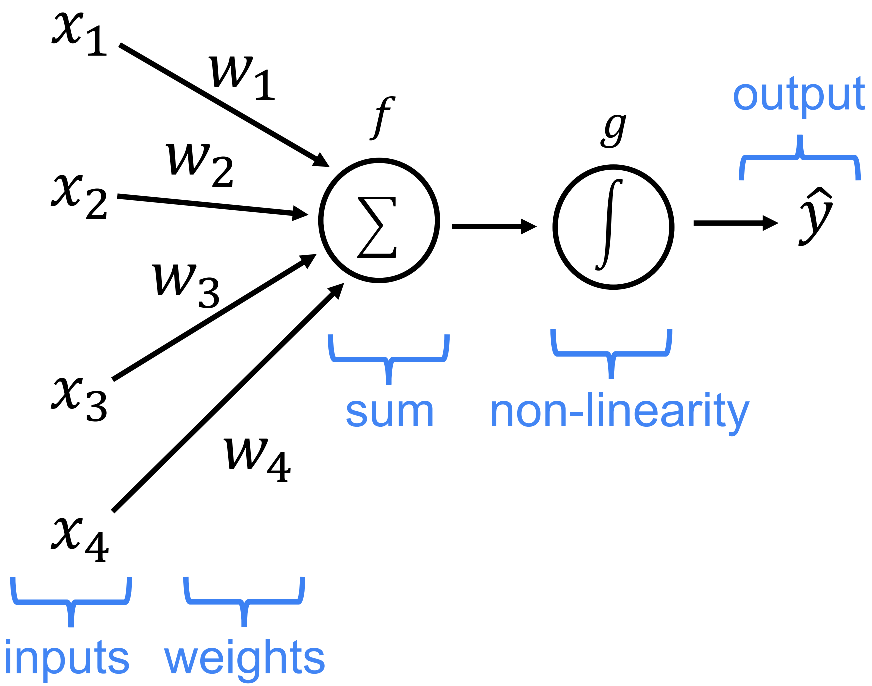


https://www.youtube.com/watch?v=6qS83wD29PY

Credit: socratic.org

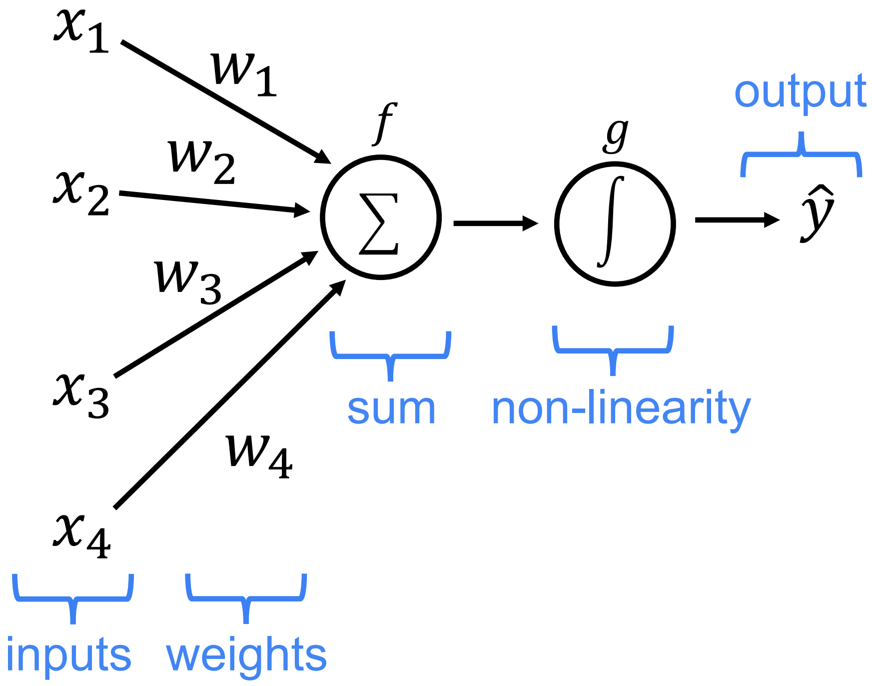
Perceptron





dendrites ≈ weighted summation with local nonlinear subunits
soma/axon hillock ≈ nonlinear activation
axon ≈ output wire
nucleus ≈ parameter management and long-term learning machinery

Perceptron

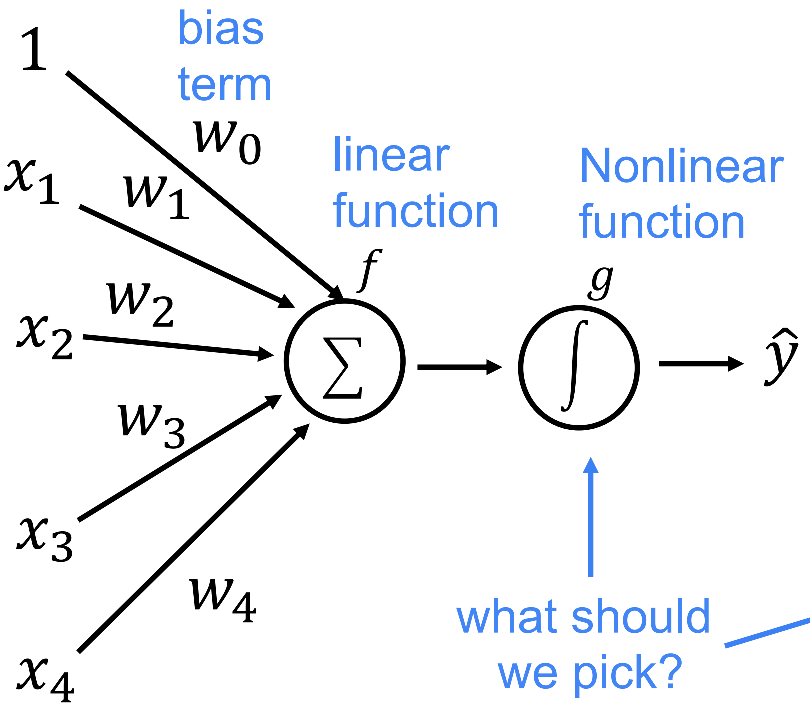


output

$$\hat{y} = g\left(\overbrace{\sum_{i=1}^{n} w_i x_i}^{\text{linear combination of inputs}}\right)$$

non-linear
activation function

Perceptron

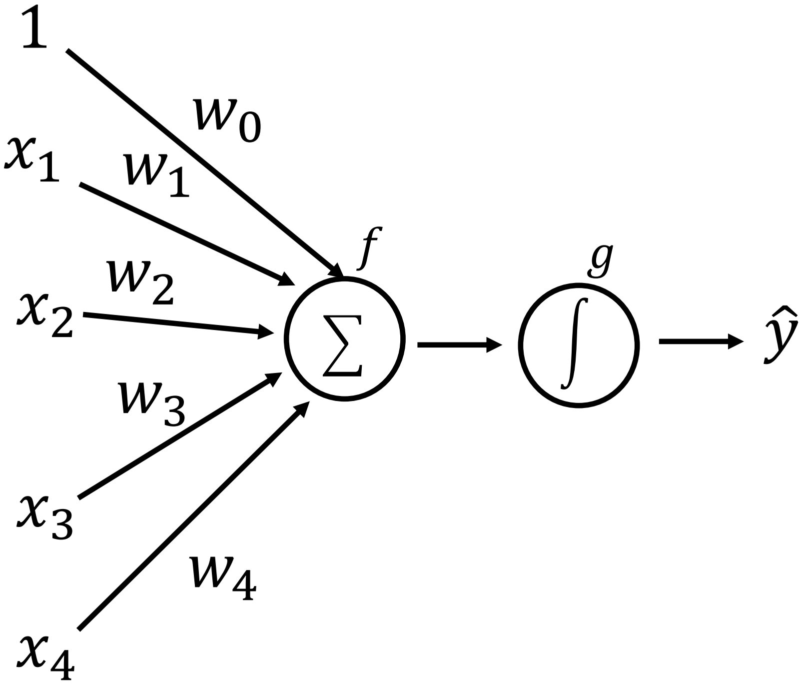


$$\hat{y} = g\left(w_0 + \sum_{i=1}^{n} w_i x_i\right)$$

$$\hat{y} = g\left(\sum_{i=0}^{n} w_i x_i\right)$$

$(x_0 = 1)$

Perceptron

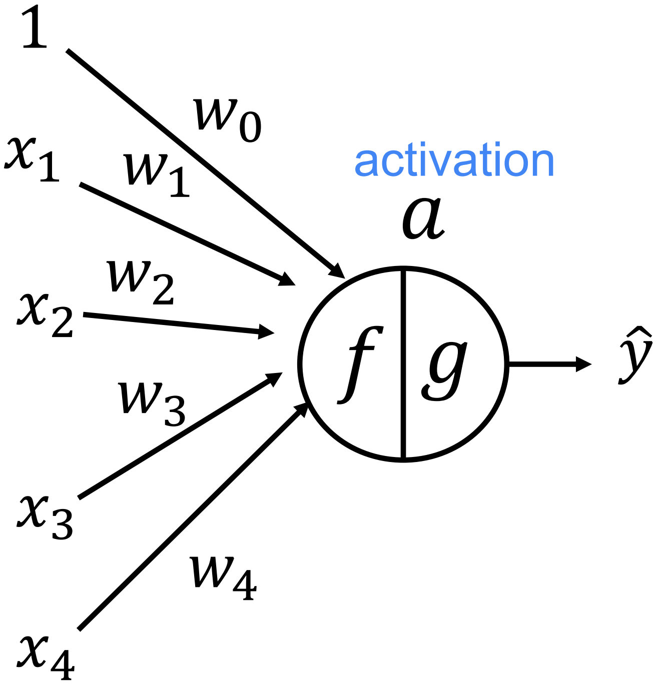


$$\hat{y} = g(f(\boldsymbol{x}))$$

$$f(\boldsymbol{x}) = \sum_{i=0}^{n} w_i x_i = \boldsymbol{w}^T \boldsymbol{x}$$

$$\boldsymbol{w} = \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_n \end{bmatrix} \qquad \boldsymbol{x} = \begin{bmatrix} 1 \\ x_1 \\ \vdots \\ x_n \end{bmatrix}$$

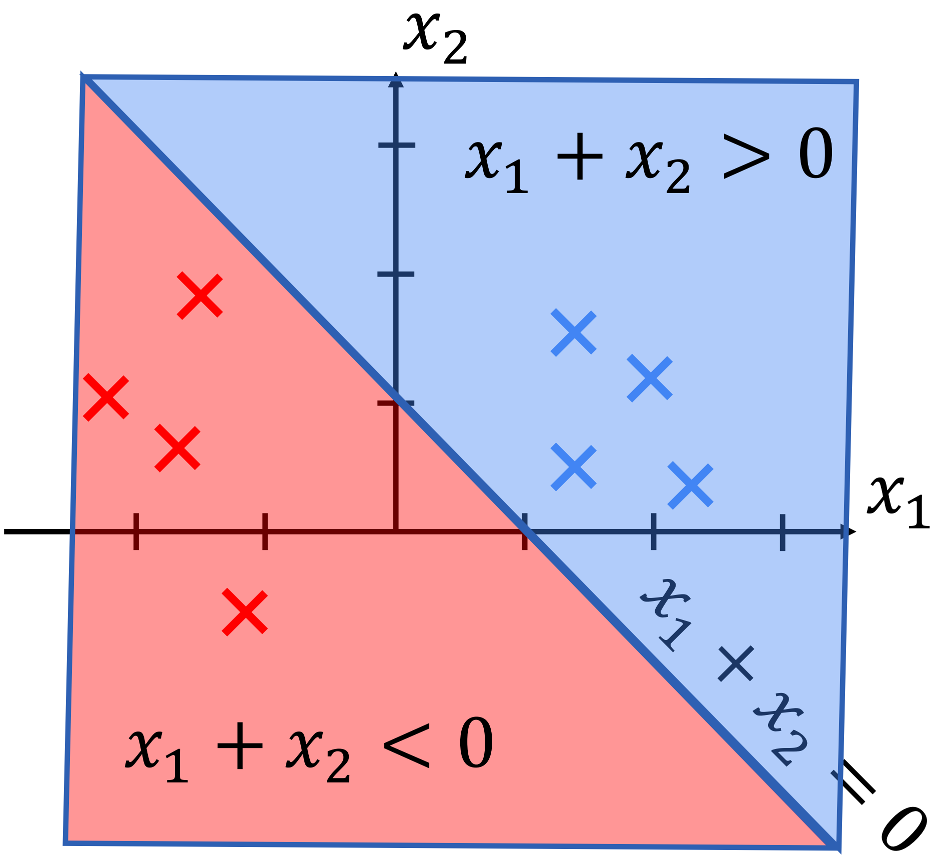
Perceptron



$$f(\boldsymbol{x}) = \sum_{i=0}^{n} w_i x_i = \boldsymbol{w}^T \boldsymbol{x}$$

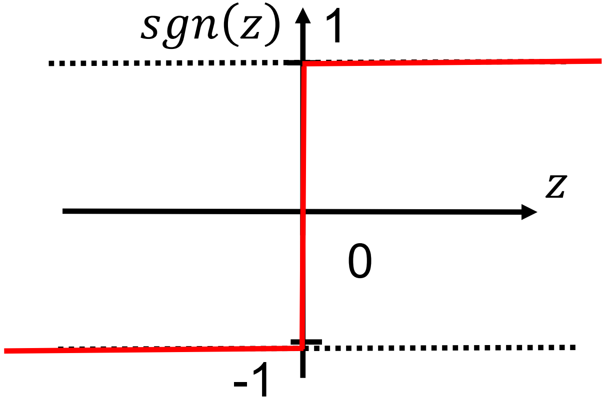
$$\hat{y} = a = g(f)$$

Linear Classification



$$\hat{y} = sgn\left(\sum_{i=0}^{n} w_i x_i\right)$$

$$sgn(z) = \begin{cases} +1 & z \geq 0 \\ -1 & z < 0 \end{cases}$$



Perceptron Learning Algorithm (PLA)

Frank Rosenblatt (1943)

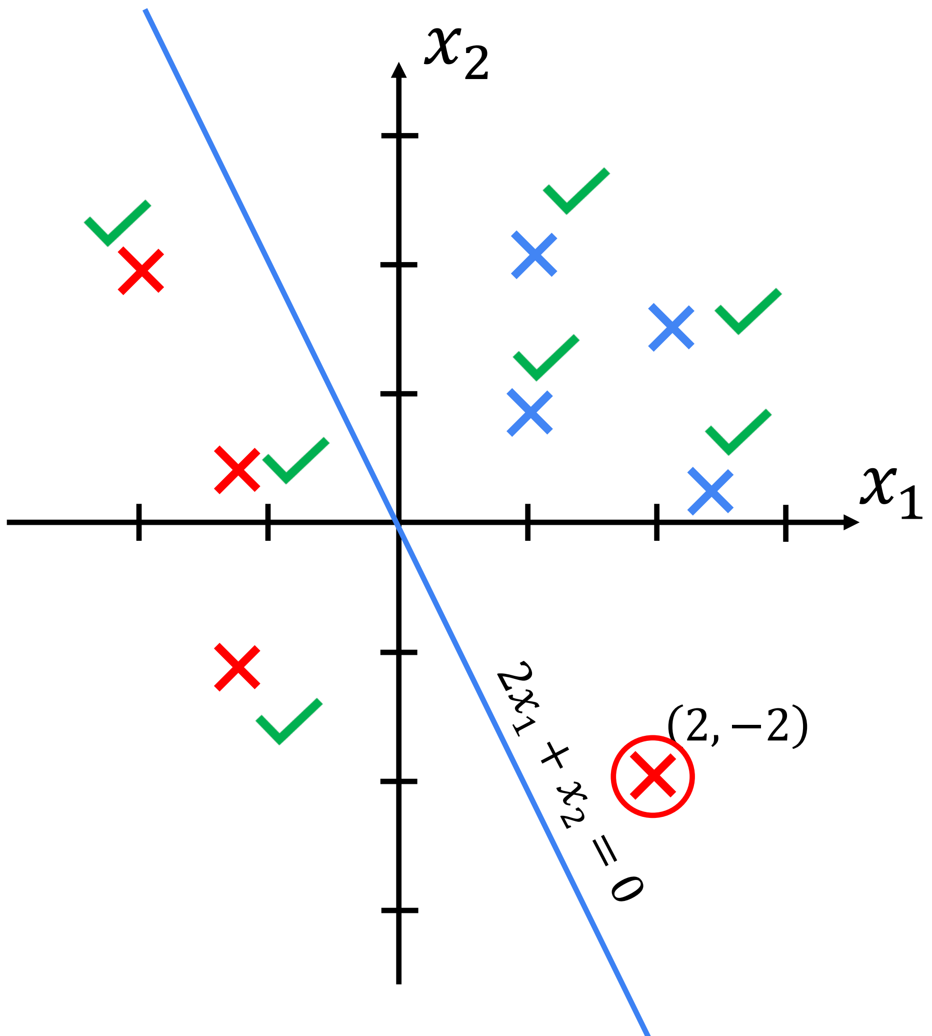
1. Initialize weights w_i
   - Could be all zero, or random small values
2. For each instance i with features $x^{(i)}$
   - Classify $\hat{y}^{(i)} = sgn\left(w^T x^{(i)}\right)$
3. Select one misclassified instance
   - Update weights: $w \leftarrow w + \Delta w$ How do we update w?
4. Iterate steps 2 to 3 until
   - Convergence (classification error < threshold), or
   - Maximum number of iterations

Perceptron Update Rule

new weight learning rate

$$\boldsymbol{w} \leftarrow \boldsymbol{w} + \eta \underbrace{(y - \hat{y})}_{\text{learning error}} \boldsymbol{x}$$

old weight



$$\boldsymbol{w} = \begin{bmatrix} 0 \\ 1 \\ 0.5 \end{bmatrix} \quad \boldsymbol{x} = \begin{bmatrix} 1 \\ x_1 \\ x_2 \end{bmatrix}$$

$$\boldsymbol{w}^T\boldsymbol{x} = x_1 + 0.5x_2$$

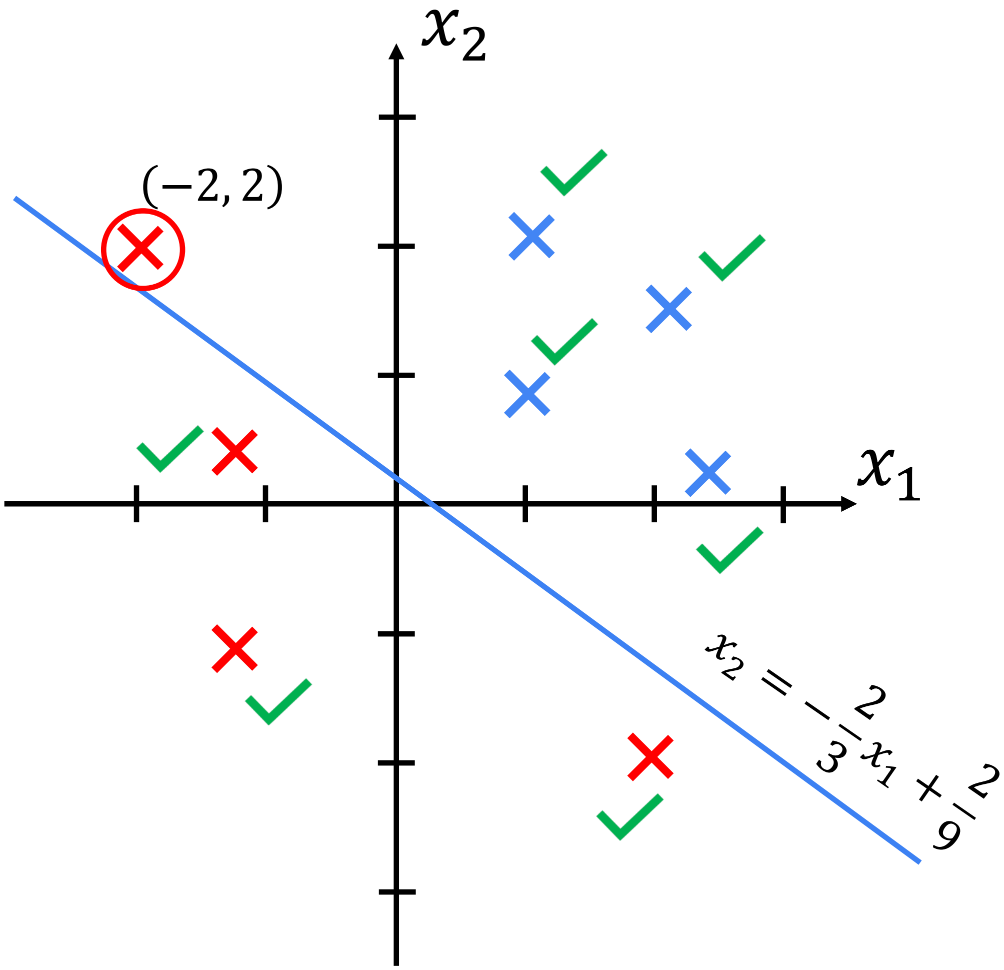
$$\hat{y} = sgn(\boldsymbol{w}^T\boldsymbol{x})$$

$$\boldsymbol{w} \leftarrow \boldsymbol{w} + \eta(y - \hat{y})\boldsymbol{x}$$

$$\begin{bmatrix} 0 \\ 1 \\ 0.5 \end{bmatrix} + 0.1(-1 - 1)\begin{bmatrix} 1 \\ 2 \\ -2 \end{bmatrix}$$

$$\begin{bmatrix} 0 \\ 1 \\ 0.5 \end{bmatrix} - 0.2\begin{bmatrix} 1 \\ 2 \\ -2 \end{bmatrix} = \begin{bmatrix} -0.2 \\ 0.6 \\ 0.9 \end{bmatrix}$$

$$\boldsymbol{w}^T\boldsymbol{x} = -0.2 + 0.6x_1 + 0.9x_2$$



$$\boldsymbol{w} = \begin{bmatrix} -0.2 \\ 0.6 \\ 0.9 \end{bmatrix} \quad \boldsymbol{x} = \begin{bmatrix} 1 \\ x_1 \\ x_2 \end{bmatrix}$$

$$\boldsymbol{w}^T\boldsymbol{x} = -0.2 + 0.6x_1 + 0.9x_2$$

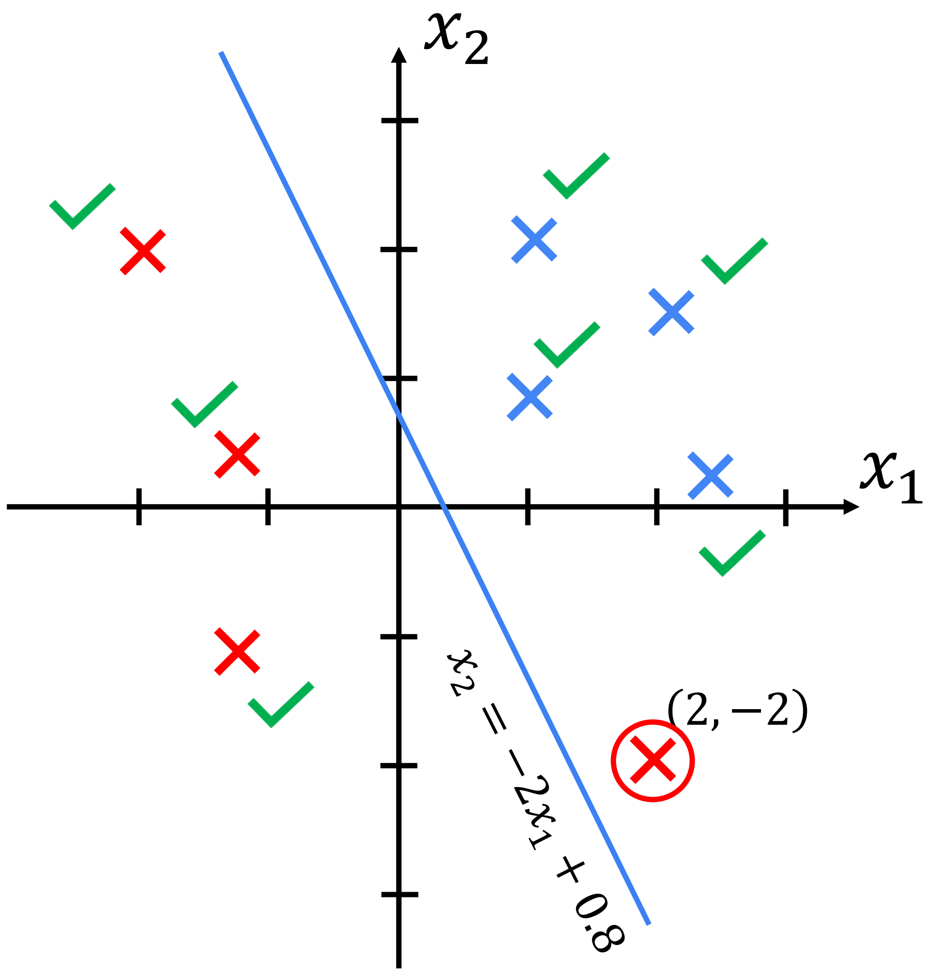
$$\hat{y} = sgn(\boldsymbol{w}^T\boldsymbol{x})$$

$$\boldsymbol{w} \leftarrow \boldsymbol{w} + \eta(y - \hat{y})\boldsymbol{x}$$

$$\begin{bmatrix} -0.2 \\ 0.6 \\ 0.9 \end{bmatrix} + 0.1(-1-1)\begin{bmatrix} 1 \\ -2 \\ 2 \end{bmatrix}$$

$$\begin{bmatrix} -0.2 \\ 0.6 \\ 0.9 \end{bmatrix} - 0.2\begin{bmatrix} 1 \\ -2 \\ 2 \end{bmatrix} = \begin{bmatrix} -0.4 \\ 1 \\ 0.5 \end{bmatrix}$$

$$\boldsymbol{w}^T\boldsymbol{x} = -0.4 + x_1 + 0.5x_2$$



$$\boldsymbol{w} = \begin{bmatrix} -0.4 \\ 1 \\ 0.5 \end{bmatrix} \quad \boldsymbol{x} = \begin{bmatrix} 1 \\ x_1 \\ x_2 \end{bmatrix}$$

$$\boldsymbol{w}^T\boldsymbol{x} = -0.4 + x_1 + 0.5x_2$$

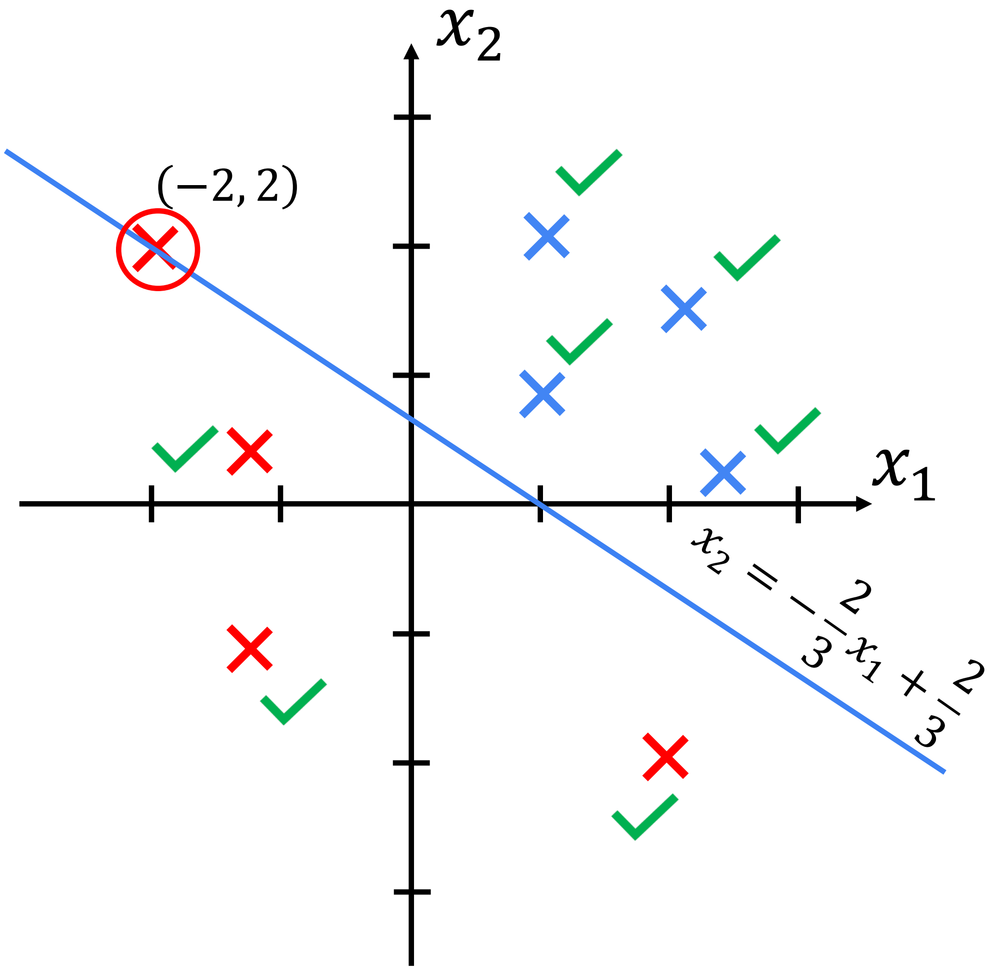
$$\hat{y} = sgn(\boldsymbol{w}^T\boldsymbol{x})$$

$$\boldsymbol{w} \leftarrow \boldsymbol{w} + \eta(y - \hat{y})\boldsymbol{x}$$

$$\begin{bmatrix} -0.4 \\ 1 \\ 0.5 \end{bmatrix} + 0.1(-1 - 1) \begin{bmatrix} 1 \\ 2 \\ -2 \end{bmatrix}$$

$$\begin{bmatrix} -0.4 \\ 1 \\ 0.5 \end{bmatrix} - 0.2 \begin{bmatrix} 1 \\ 2 \\ -2 \end{bmatrix} = \begin{bmatrix} -0.6 \\ 0.6 \\ 0.9 \end{bmatrix}$$

$$\boldsymbol{w}^T\boldsymbol{x} = -0.6 + 0.6x_1 + 0.9x_2$$



$$\boldsymbol{w} = \begin{bmatrix} -0.6 \\ 0.6 \\ 0.9 \end{bmatrix} \quad \boldsymbol{x} = \begin{bmatrix} 1 \\ x_1 \\ x_2 \end{bmatrix}$$

$$\boldsymbol{w}^T\boldsymbol{x} = -0.6 + 0.6x_1 + 0.9x_2$$

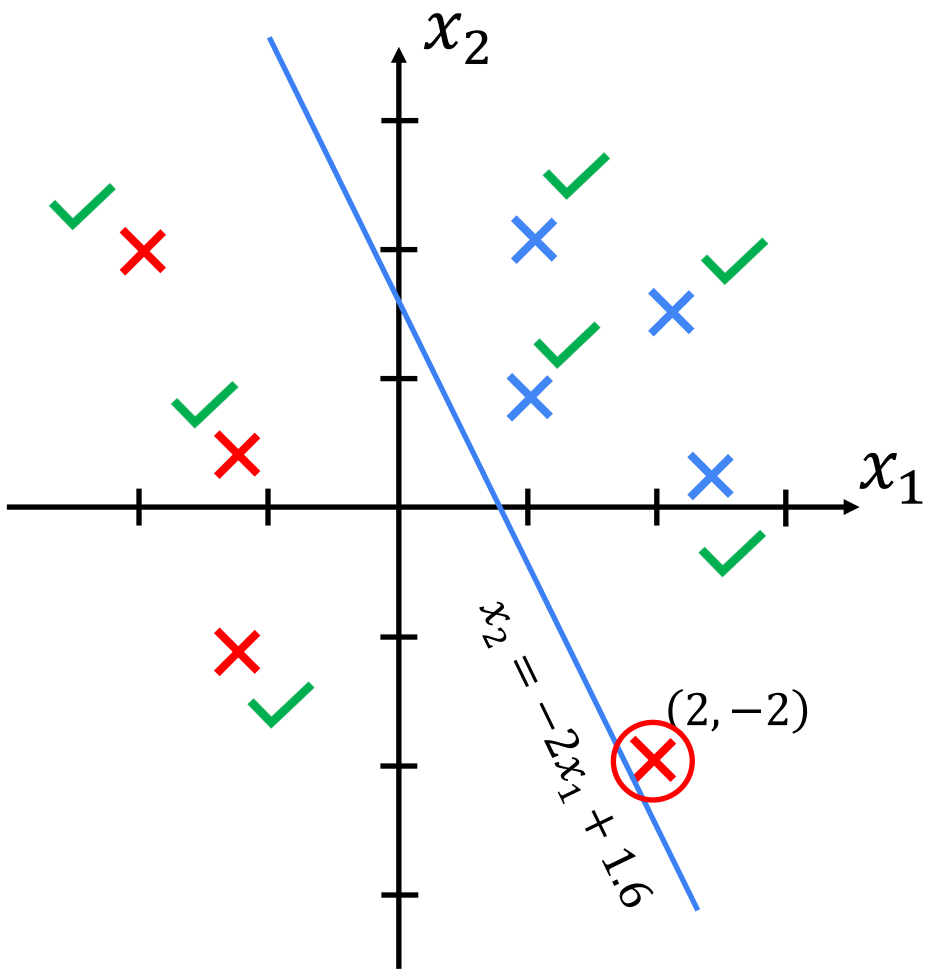
$$\hat{y} = sgn(\boldsymbol{w}^T\boldsymbol{x})$$

$$\boldsymbol{w} \leftarrow \boldsymbol{w} + \eta(y - \hat{y})\boldsymbol{x}$$

$$\begin{bmatrix} -0.6 \\ 0.6 \\ 0.9 \end{bmatrix} + 0.1(-1-1)\begin{bmatrix} 1 \\ -2 \\ 2 \end{bmatrix}$$

$$\begin{bmatrix} -0.6 \\ 0.6 \\ 0.9 \end{bmatrix} - 0.2\begin{bmatrix} 1 \\ -2 \\ 2 \end{bmatrix} = \begin{bmatrix} -0.8 \\ 1 \\ 0.5 \end{bmatrix}$$

$$w^Tx = -0.8 + x_1 + 0.5x_2$$



$$\boldsymbol{w} = \begin{bmatrix} -0.8 \\ 1 \\ 0.5 \end{bmatrix} \quad \boldsymbol{x} = \begin{bmatrix} 1 \\ x_1 \\ x_2 \end{bmatrix}$$

$$\boldsymbol{w}^T\boldsymbol{x} = -0.8 + x_1 + 0.5x_2$$

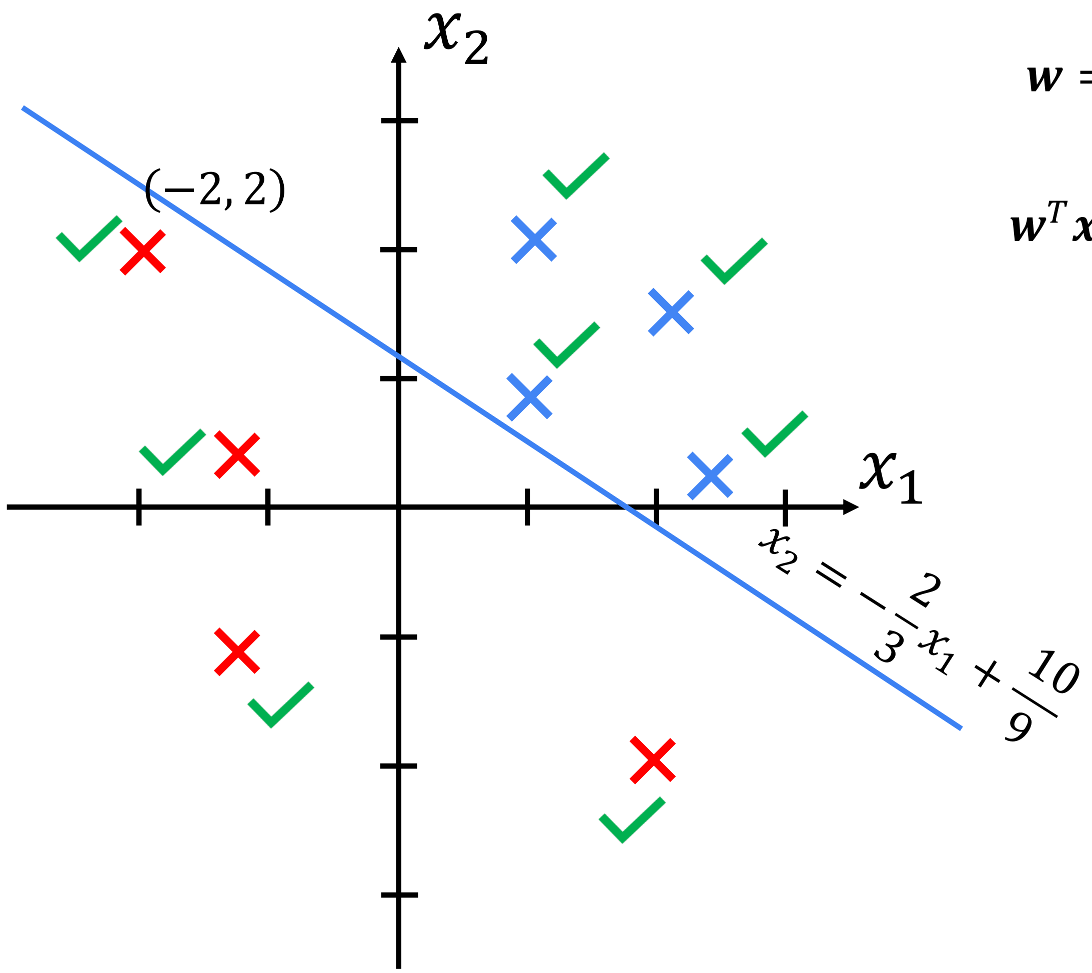
$$\hat{y} = sgn(\boldsymbol{w}^T\boldsymbol{x})$$

$$\boldsymbol{w} \leftarrow \boldsymbol{w} + \eta(y - \hat{y})\boldsymbol{x}$$

$$\begin{bmatrix} -0.8 \\ 1 \\ 0.5 \end{bmatrix} + 0.1(-1 - 1) \begin{bmatrix} 1 \\ 2 \\ -2 \end{bmatrix}$$

$$\begin{bmatrix} -0.8 \\ 1 \\ 0.5 \end{bmatrix} - 0.2 \begin{bmatrix} 1 \\ 2 \\ -2 \end{bmatrix} = \begin{bmatrix} -1 \\ 0.6 \\ 0.9 \end{bmatrix}$$

$$\boldsymbol{w}^T\boldsymbol{x} = -1 + 0.6x_1 + 0.9x_2$$



$$\boldsymbol{w} = \begin{bmatrix} -1 \\ 0.6 \\ 0.9 \end{bmatrix} \qquad \boldsymbol{x} = \begin{bmatrix} 1 \\ x_1 \\ x_2 \end{bmatrix}$$

$$\boldsymbol{w}^T\boldsymbol{x} = -1 + 0.6x_1 + 0.9x_2$$

Phew!

Why does this work?

Perceptron Update Rule

Consider what happens when we have a misclassification:

$$\hat{y} = sgn\left(\sum_{i=0}^{n} w_i x_i\right)$$

$$sgn(z) = \begin{cases} +1 & z \geq 0 \\ -1 & z < 0 \end{cases}$$

$$y = +1, \hat{y} = -1$$

$$y = +1, \boldsymbol{w}^T \boldsymbol{x} < 0$$

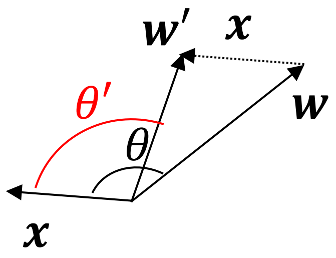
$$y = +1, ||\boldsymbol{w}||.||\boldsymbol{x}|| \cos\theta < 0$$

$$y = +1, \cos\theta < 0$$

$$y = +1, \frac{\pi}{2} < \theta < \pi$$

Want: $y = +1, \cos\theta > 0$

$$y = +1, 0 < \theta < \frac{\pi}{2}$$



$$\boldsymbol{w}' \leftarrow \boldsymbol{w} + \boldsymbol{x}$$

θ' will be closer to $\frac{\pi}{2}$

Perceptron Update Rule

Consider what happens when we have a misclassification:

$$\hat{y} = sgn\left(\sum_{i=0}^{n} w_i x_i\right)$$

$$sgn(z) = \begin{cases} +1 & z \geq 0 \\ -1 & z < 0 \end{cases}$$

$$y = -1, \hat{y} = +1$$

$$y = -1, \boldsymbol{w}^T\boldsymbol{x} > 0$$

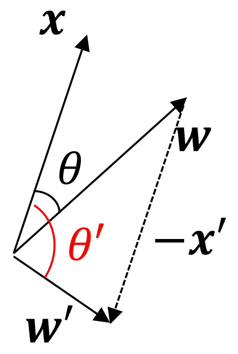
$$y = -1, ||\boldsymbol{w}||.||\boldsymbol{x}|| \cos\theta > 0$$

$$y = -1, \cos\theta > 0$$

$$y = -1, 0 < \theta < \frac{\pi}{2}$$

Want: $y = -1, \cos\theta < 0$

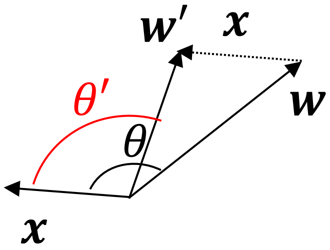
$$y = -1, \frac{\pi}{2} < \theta < \pi$$



$$\boldsymbol{w}' \leftarrow \boldsymbol{w} - \boldsymbol{x}$$

θ' will be closer to $\frac{\pi}{2}$

Perceptron Update Rule



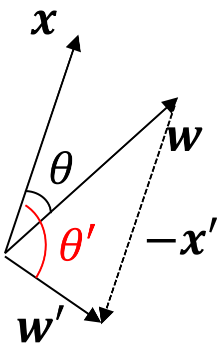
$y = +1, \hat{y} = -1$

$\boldsymbol{w}' \leftarrow \boldsymbol{w} + \boldsymbol{x}$

$y - \hat{y} = 2$

$\boldsymbol{w}' \leftarrow \boldsymbol{w} + 2\eta\boldsymbol{x}$

$$\boldsymbol{w} \leftarrow \boldsymbol{w} + \eta(y - \hat{y})\boldsymbol{x}$$

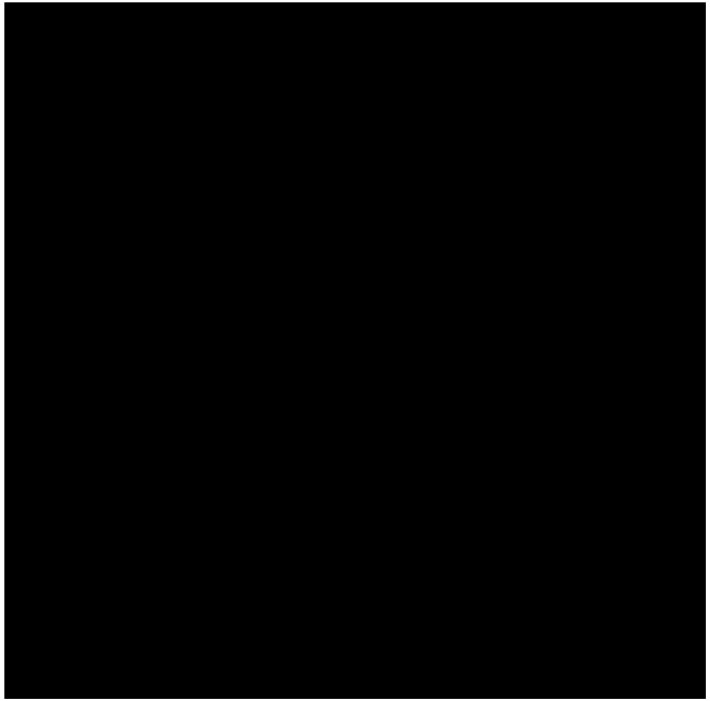


$y = -1, \hat{y} = +1$

$\boldsymbol{w}' \leftarrow \boldsymbol{w} - \boldsymbol{x}$

$y - \hat{y} = -2$

$\boldsymbol{w}' \leftarrow \boldsymbol{w} - 2\eta\boldsymbol{x}$



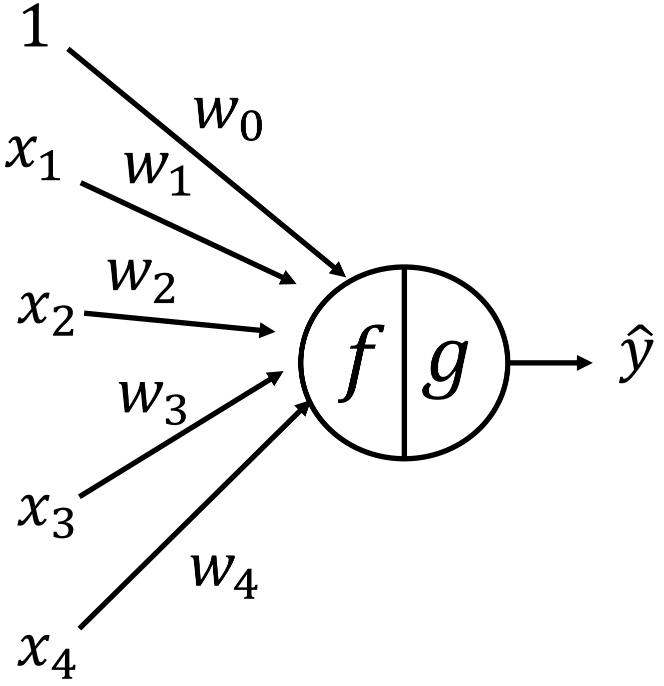
https://tamas.xyz/perceptron-demo/app/

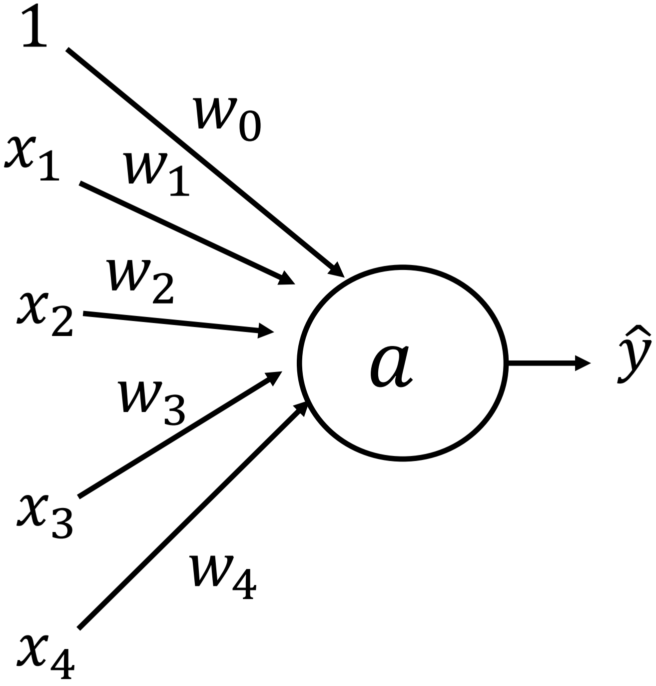
Perceptron Learning Algorithm

- not robust: Can select any linear model, not deterministic

- Cannot converge on non-linearly separable data

Single Layer Perceptron (SLP)



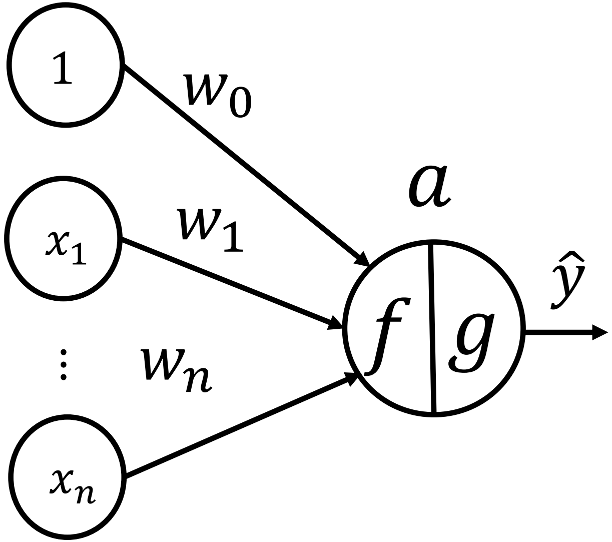


$$f(\boldsymbol{w}, \boldsymbol{x}) = \sum_{i=0}^{n} w_i x_i = \boldsymbol{w}^T \boldsymbol{x}$$

$$\hat{y} = a = g(f)$$

SLP: Linear and Logistic Regression

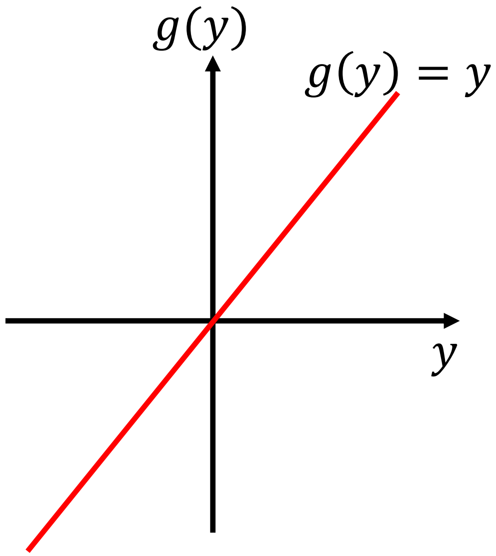
Linear Regression



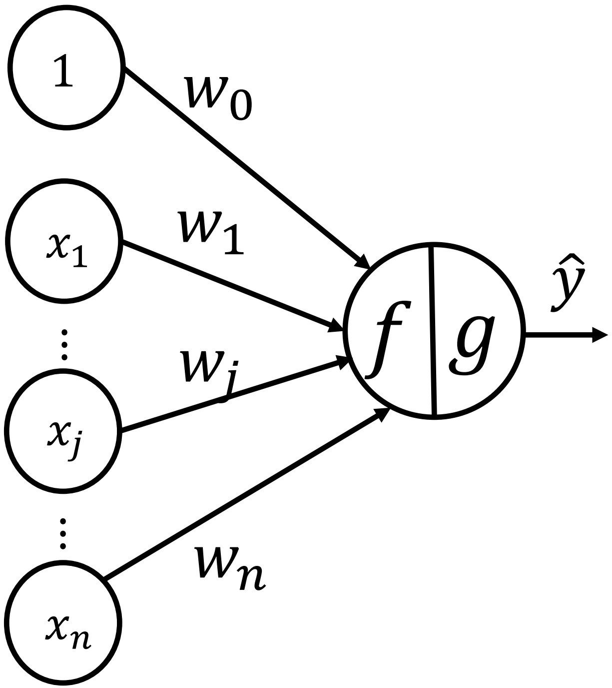
$$a = g\big(f(\boldsymbol{x})\big),$$

$$f(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{x}$$

$$g(y) = y$$



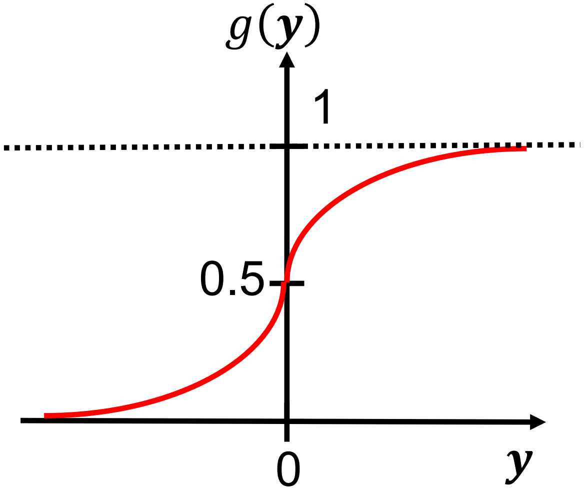
Logistic Regression (Binary Classification)



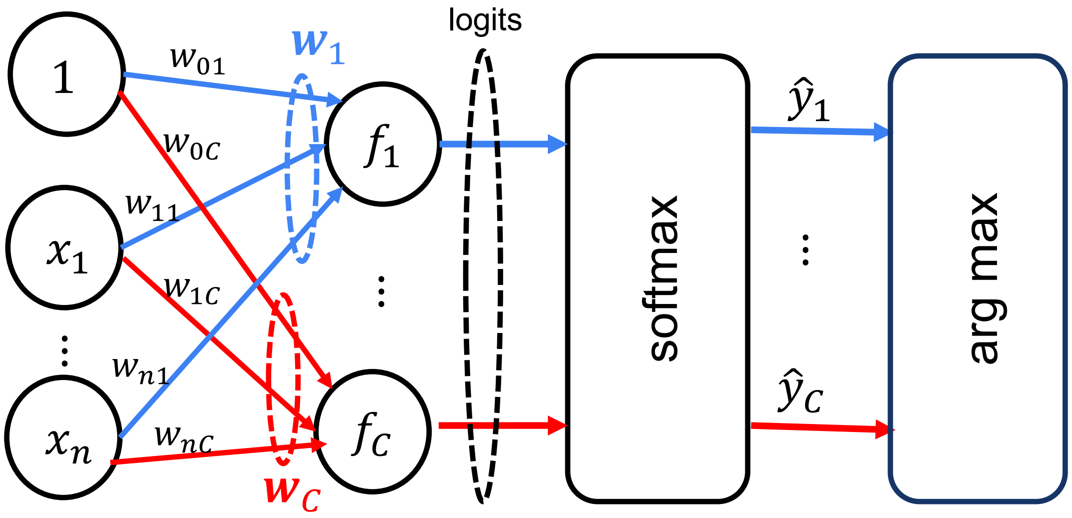
$$a = g\big(f(\boldsymbol{x})\big),$$

$$f(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{x}$$

$$g(y) = \sigma(y) = \frac{1}{1+e^{-y}}$$



Logistic Regression (Multiclass Classification)



$$\boldsymbol{w}_i = \begin{bmatrix} w_{0i} \\ w_{1i} \\ \vdots \\ w_{ni} \end{bmatrix} \quad \boldsymbol{x} = \begin{bmatrix} 1 \\ x_1 \\ \vdots \\ x_n \end{bmatrix} \qquad f_i(\boldsymbol{x}) = \boldsymbol{w}_i^T \boldsymbol{x} \qquad \begin{bmatrix} \hat{y}_1 \\ \vdots \\ \hat{y}_C \end{bmatrix} = softmax\left(\begin{bmatrix} f_1(\boldsymbol{x}) \\ \vdots \\ f_C(\boldsymbol{x}) \end{bmatrix} \right)$$

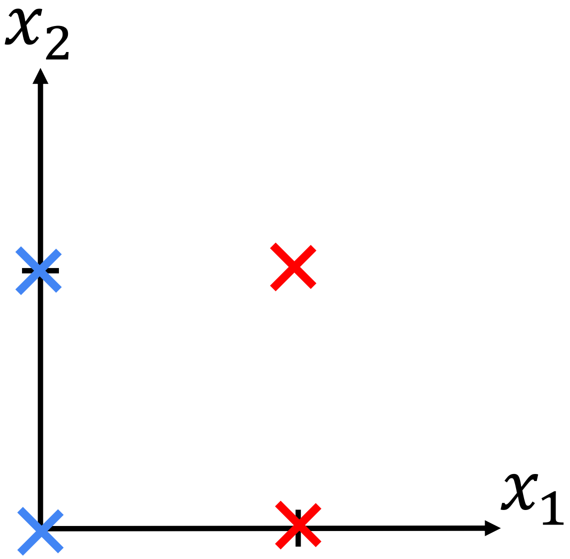
Why Multiple Layer Perceptron (MLP)?

Logic Gate Modeling with Perceptron

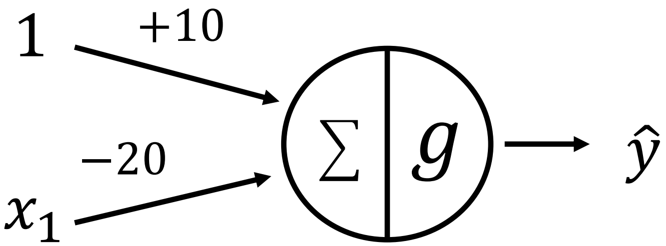
X_1	X_2	NOT X_1	X_1 AND X_2	X_1 OR X_2	X_1 XNOR X_2
0	0	1	0	0	1
0	1	1	0	1	0
1	0	0	0	1	0
1	1	0	1	1	1

Example 1: NOT

X_1	X_2	NOT X_1
0	0	1
0	1	1
1	0	0
1	1	0



Example 1: NOT



$$g(x) = sgn(x)$$

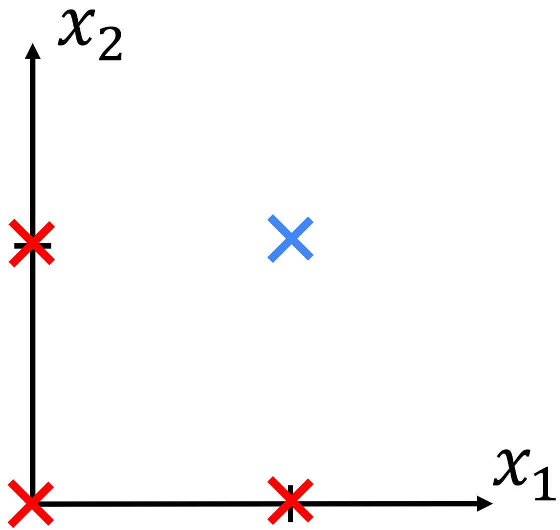
$$\hat{y} = g(\boldsymbol{w}^T \boldsymbol{x})$$

Consider $x_1 \in \{1,0\}$

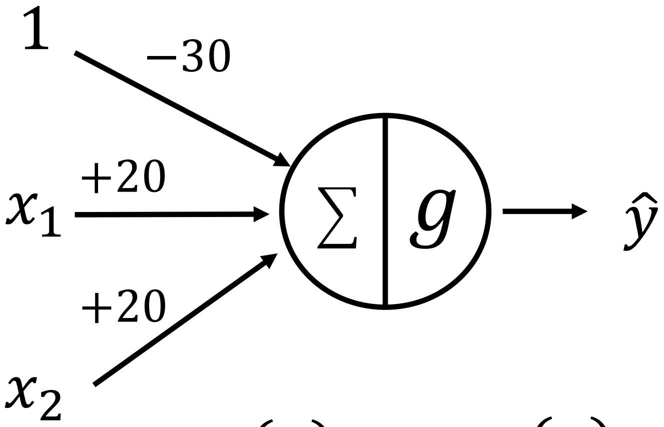
x_1	$\boldsymbol{w}^T \boldsymbol{x}$	$\hat{y}$
0	10	1
1	-10	0

Example 2: AND

X_1	X_2	X_1 AND X_2
0	0	0
0	1	0
1	0	0
1	1	1



Example 2: AND



$g(x) = sgn(x)$

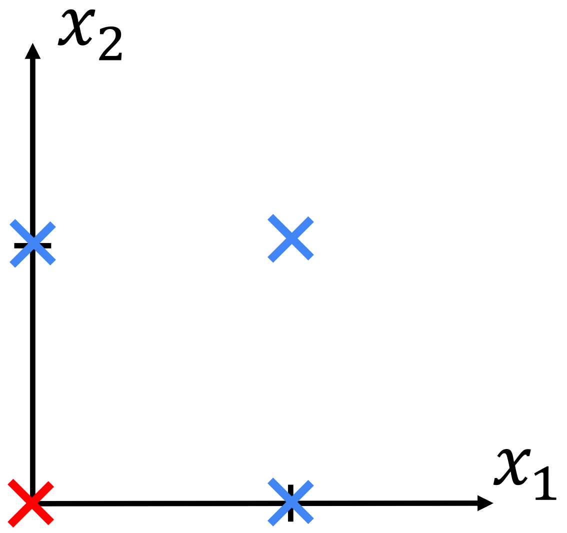
$\hat{y} = g(w^T x)$

Consider $x_1, x_2 \in \{1,0\}$

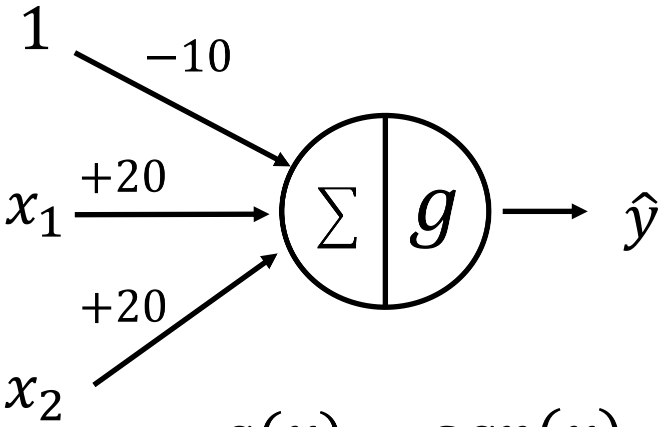
x_1	x_2	$\boldsymbol{w}^T\boldsymbol{x}$	$\hat{y}$
0	0	-30	0
0	1	-10	0
1	0	-10	0
1	1	10	1

Example 3: OR

X_1	X_2	X_1 OR X_2
0	0	0
0	1	1
1	0	1
1	1	1



Example 3: OR



$$g(x) = sgn(x)$$

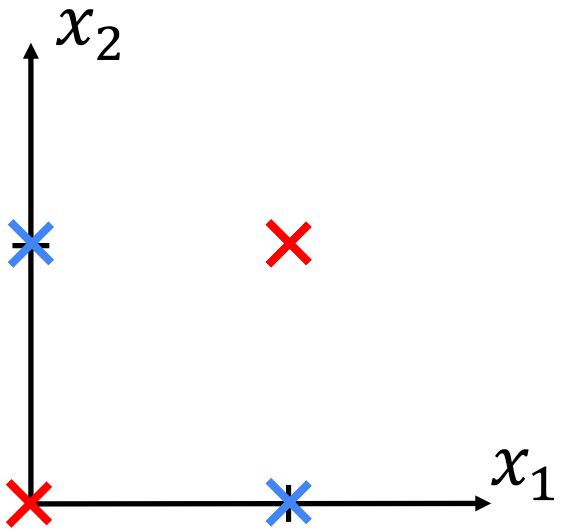
$$\hat{y} = g(w^T x)$$

Consider $x_1, x_2 \in \{1,0\}$

x_1	x_2	$\boldsymbol{w}^T\boldsymbol{x}$	$\hat{y}$
0	0	-10	0
0	1	10	1
1	0	10	1
1	1	30	1

Example 3: XNOR

X_1	X_2	X_1 XNOR X_2
0	0	1
0	1	0
1	0	0
1	1	1



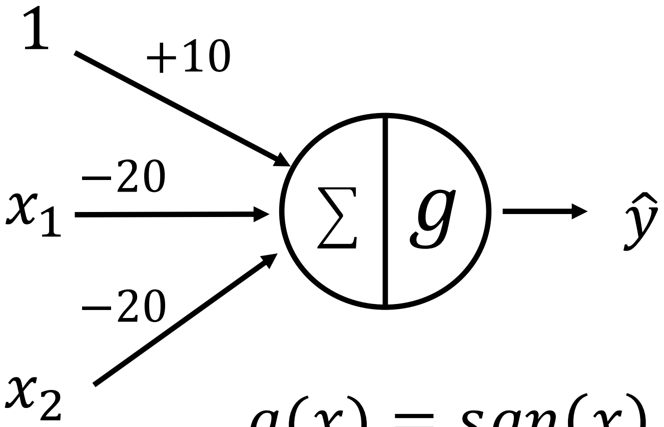
Not linearly separable

How to Model XNOR?

$$X_1\ XNOR\ X_2 \ \equiv \neg(X_1 \vee X_2) \vee (X_1 \wedge X_2)$$

X_1	X_2	X_1 NOR X_2
0	0	1
0	1	0
1	0	0
1	1	1

$\neg(X_1 \vee X_2) \equiv \neg X_1 \wedge \neg X_2$



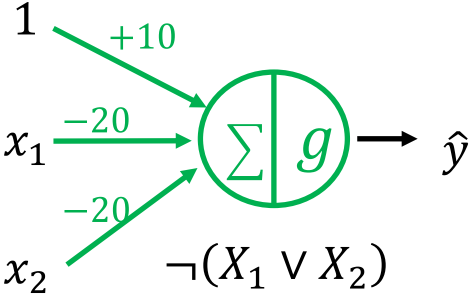
$$g(x) = sgn(x)$$

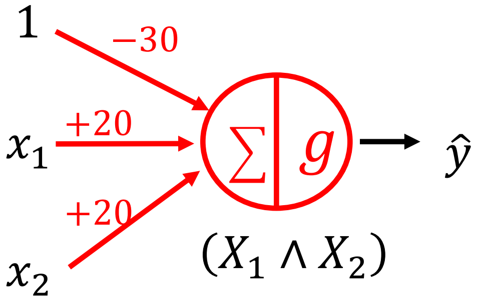
$$\hat{y} = g(w^T x)$$

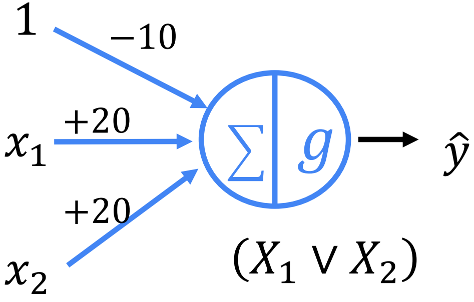
Consider $x_1, x_2 \in \{1,0\}$

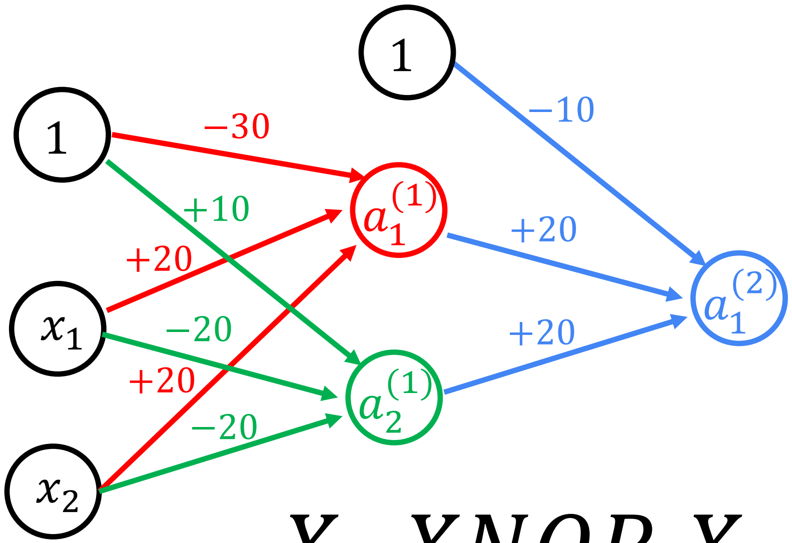
x_1	x_2	$\boldsymbol{w}^T\boldsymbol{x}$	$\hat{y}$
0	0	10	1
0	1	−10	0
1	0	−10	0
1	1	−30	0

$$X_1\ XNOR\ X_2 \equiv \neg(X_1 \vee X_2) \vee (X_1 \wedge X_2)$$







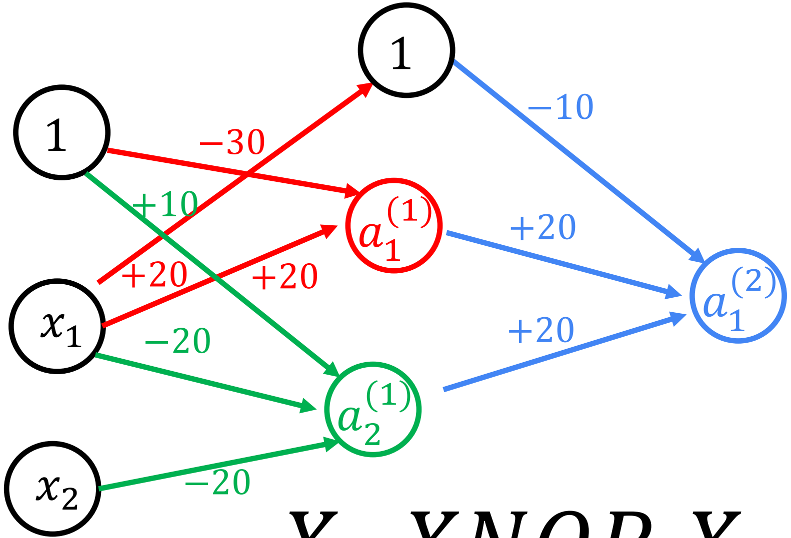


$$X_1\ XNOR\ X_2$$

Consider $x_1, x_2 \in \{1,0\}$

x_1	x_2	$a_1^{(1)}$	$a_2^{(1)}$	$a_1^{(2)}$
0	0	0	1	1
0	1	0	0	0
1	0	0	0	0
1	1	1	0	1

Multi-Layer Perceptron



$$X_1\ XNOR\ X_2$$