

Neural Networks

[RN2] Sec 20.5

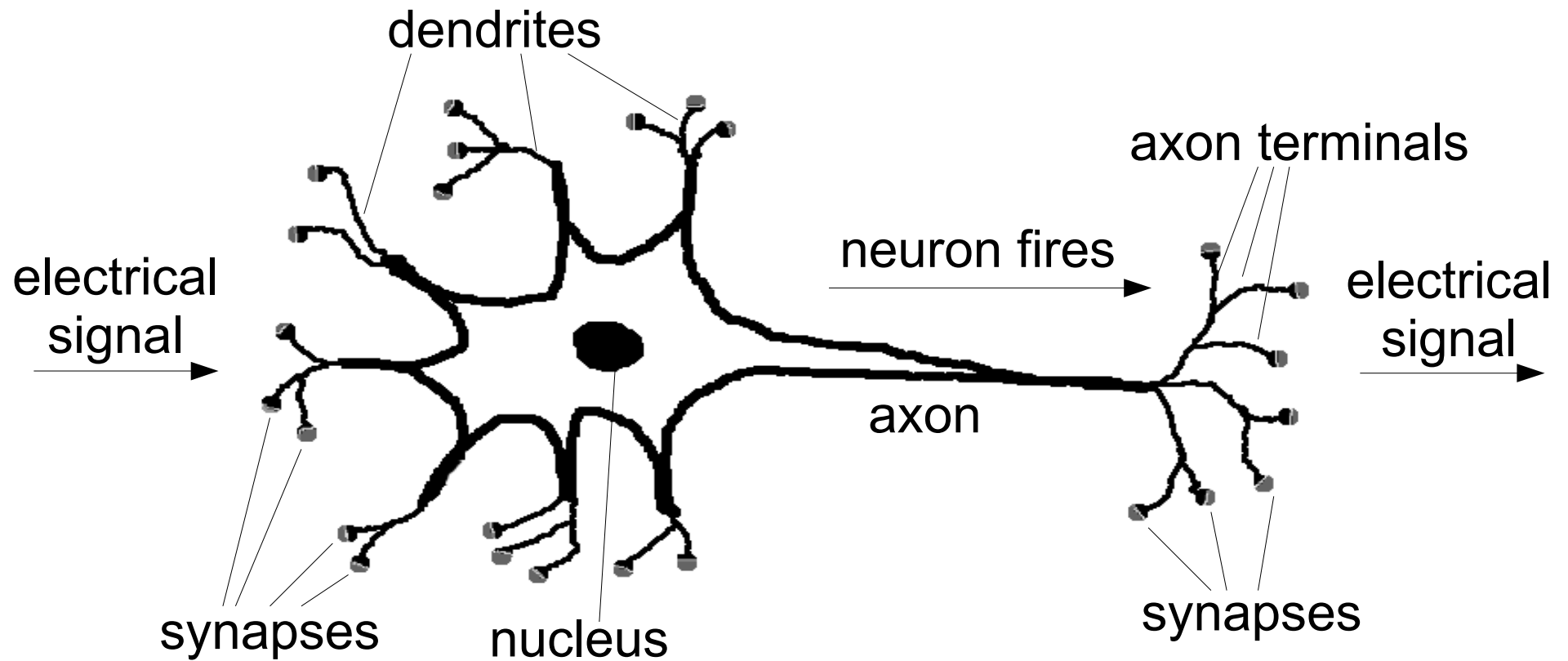
[RN3] Sec 20.5

CS 486/686

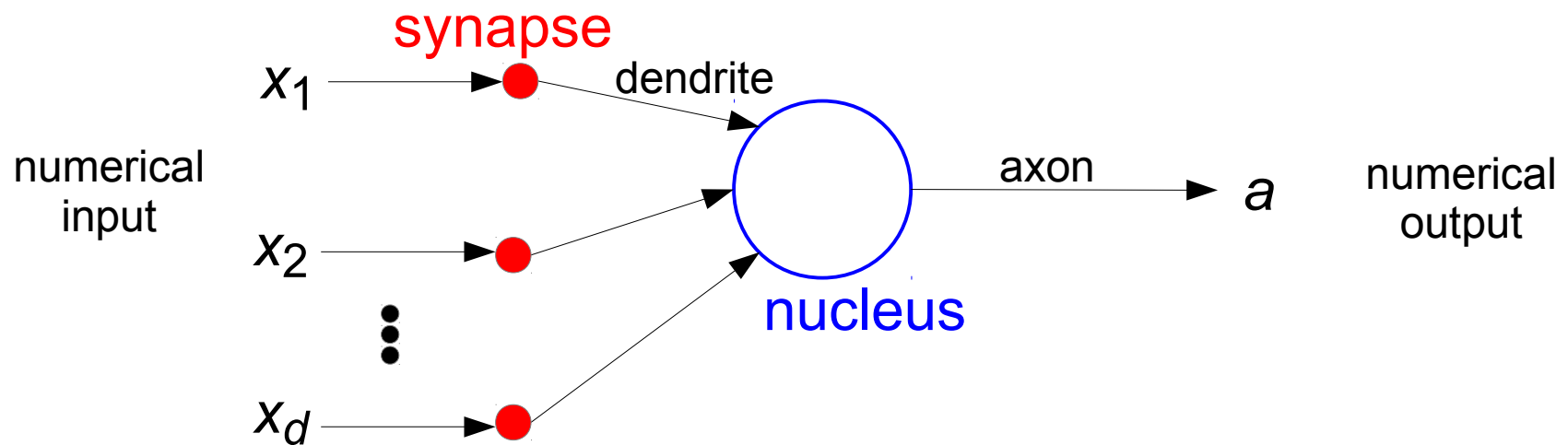
University of Waterloo

Lecture 19: July 7, 2015

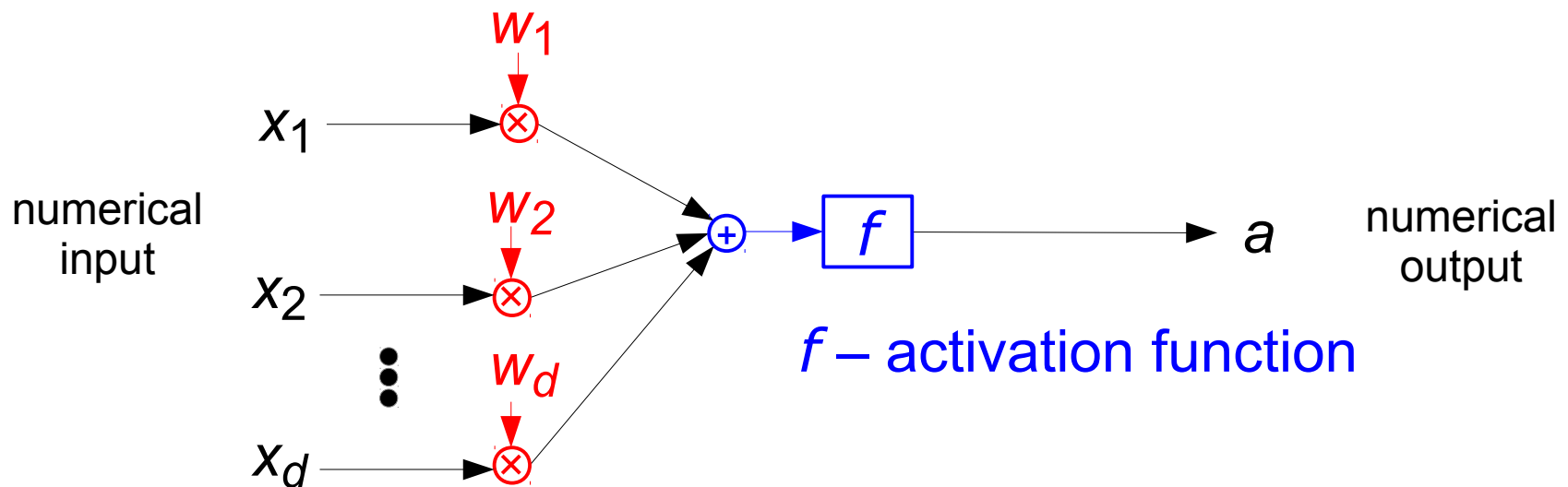
Biological Neuron



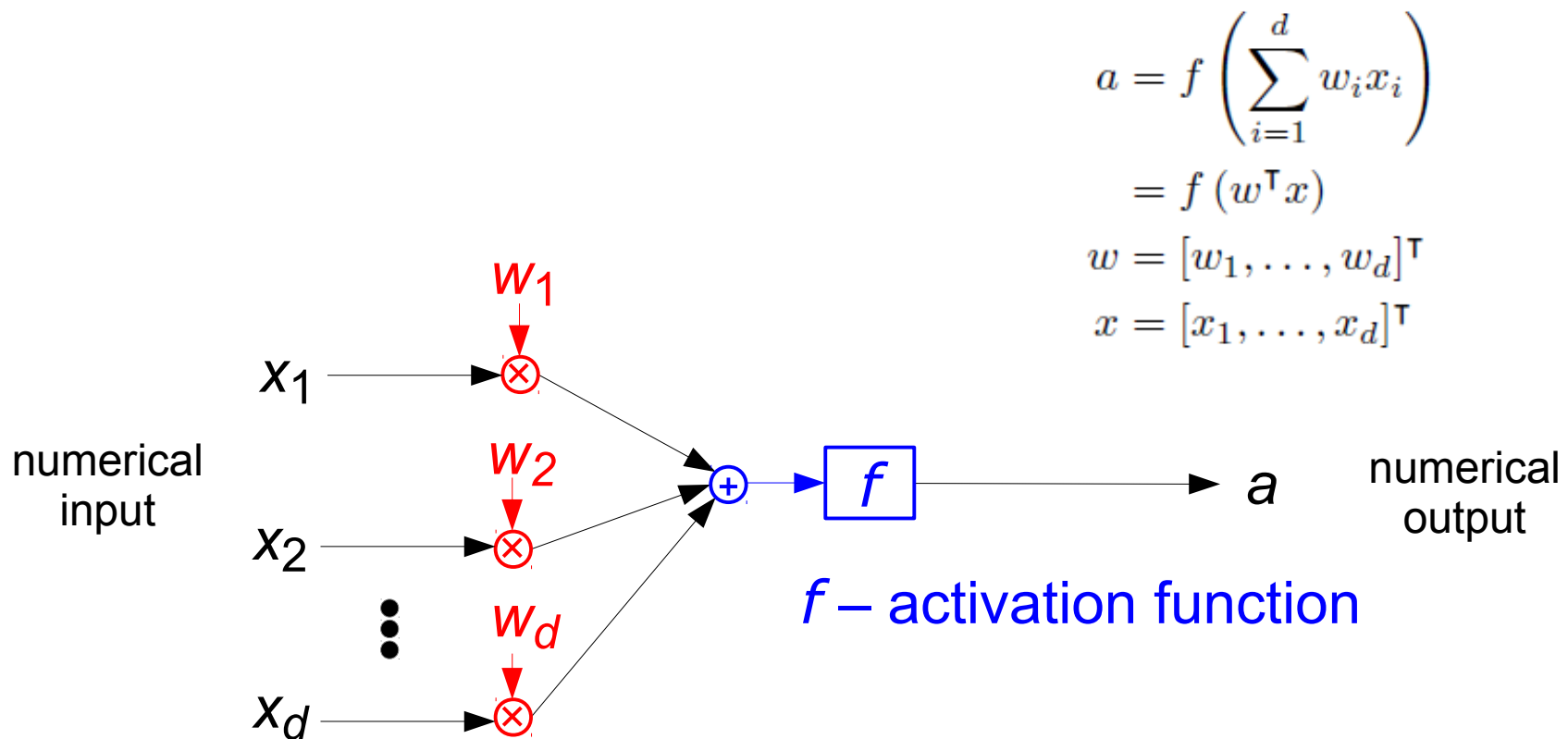
Computational Model of Neuron



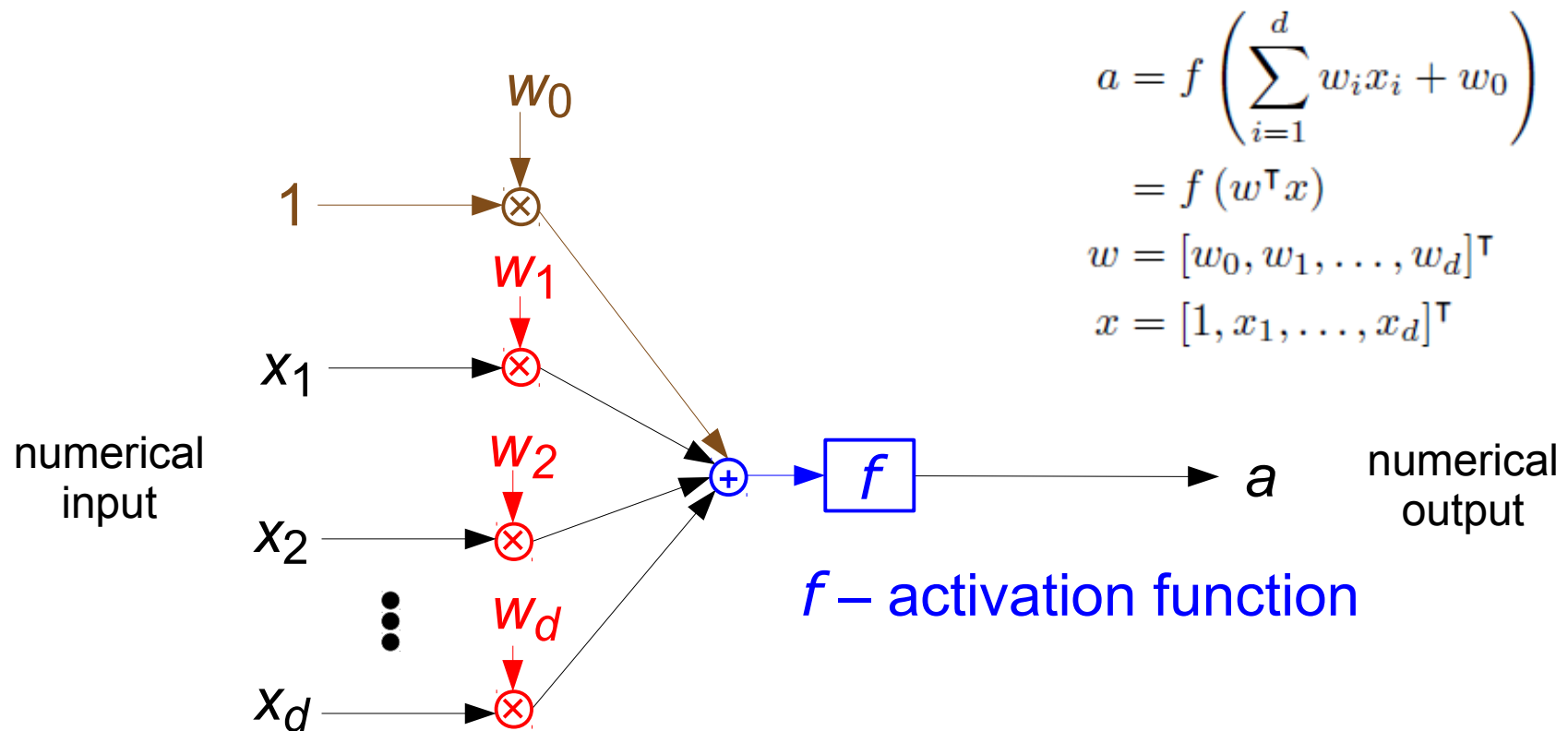
Computational Model of Neuron



Computational Model of Neuron



Computational Model of Neuron



Neuron as Binary Classifier

- Step activation function

$$f(z) = \begin{cases} 1 & \text{if } z \geq 0 \\ 0 & \text{if } z < 0 \end{cases} \quad (\text{threshold perceptron})$$

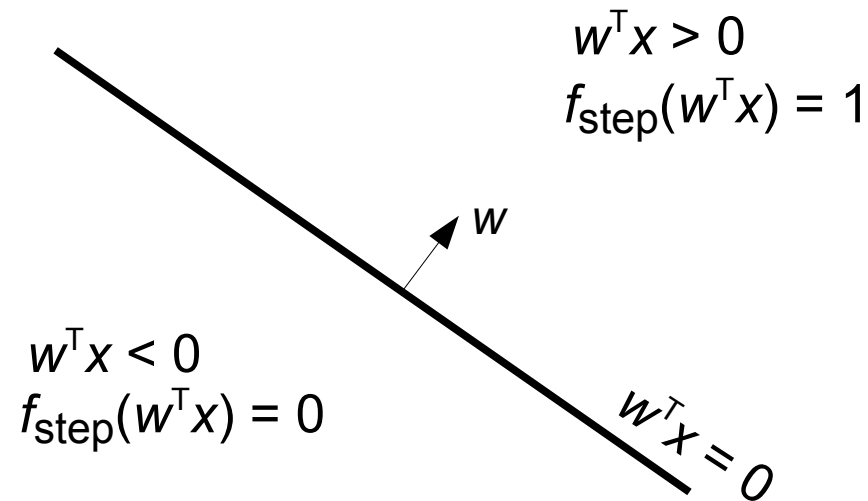
- Sigmoid activation function

$$f(z) = \sigma(z) = \frac{1}{1 + e^{-z}}$$

(sigmoid perceptron / logistic regression)

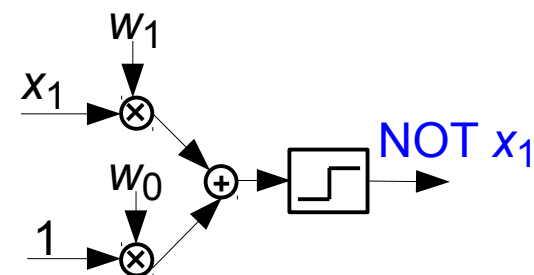
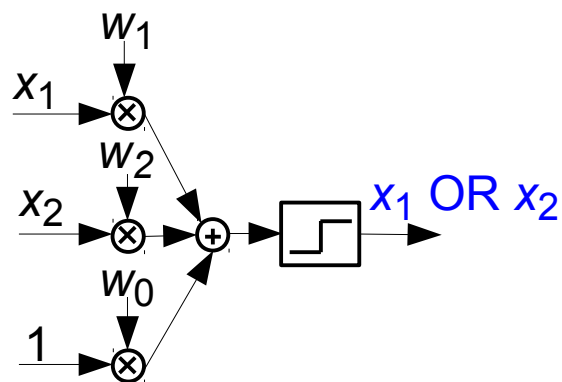
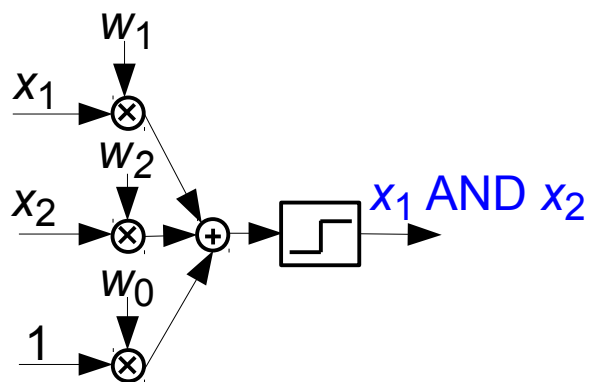
Hypothesis Space of Perceptron

- Linear separator



Threshold Perceptron as Logic Gate

- What should the weights be to encode the Boolean functions AND, OR, and NOT?



- How about XOR?

Threshold Perceptron Learning

- 1: Initialize weights \mathbf{w}
- 2: **for** each training example (\mathbf{x}, y) **do**
- 3: $a \leftarrow f_{\text{step}}(\mathbf{w}^\top \mathbf{x})$
- 4: $\mathbf{w} \leftarrow \mathbf{w} + (y - a)\mathbf{x}$
- 5: **end for**
- 6: Repeat 2–5 until every example is classified correctly

Threshold Perceptron Learning

- 1: Initialize weights \mathbf{w}
- 2: **for** each training example (\mathbf{x}, y) **do**
- 3: $a \leftarrow f_{\text{step}}(\mathbf{w}^\top \mathbf{x})$
- 4: $\mathbf{w} \leftarrow \mathbf{w} + (y - a)\mathbf{x}$
- 5: **end for**
- 6: Repeat 2–5 until every example is classified correctly

Threshold perceptron learning converges if data is **linearly separable**.

Sigmoid Perceptron Learning

- Minimize squared error:

$$E(w, x, y) = \frac{1}{2} [y - \sigma(w^\top x)]^2$$

$$\hat{w} = \arg \min_w E(w, x, y)$$

- Solution: gradient descent

Gradient Descent

- Gradient
 - $\nabla E(w_1, \dots, w_d) = \left[\frac{\partial E}{\partial w_1}, \dots, \frac{\partial E}{\partial w_d} \right]$
 - Points in the direction of steepest slope
- Gradient descent: move in the direction of negative gradient to find minimum

$$w \leftarrow w - \alpha \nabla E(w)$$

$\alpha > 0$: learning rate

Sigmoid Perceptron Learning with Gradient Descent

- 1: Initialize weights \mathbf{w}
- 2: **for** each training example (\mathbf{x}, y) **do**
- 3: $z \leftarrow \mathbf{w}^\top \mathbf{x}$
- 4: $\sigma'(z) \leftarrow \sigma(z)(1 - \sigma(z))$
- 5: $\mathbf{w} \leftarrow \mathbf{w} + \alpha \cdot (y - \sigma(z))\sigma'(z)\mathbf{x}$
- 6: **end for**
- 7: Repeat 2–6 until stopping criteria satisfied

Next: Multi-Layer Neural Networks

- Perceptron can only represent **linear separators**
- Need multiple layers to represent more complicated separators