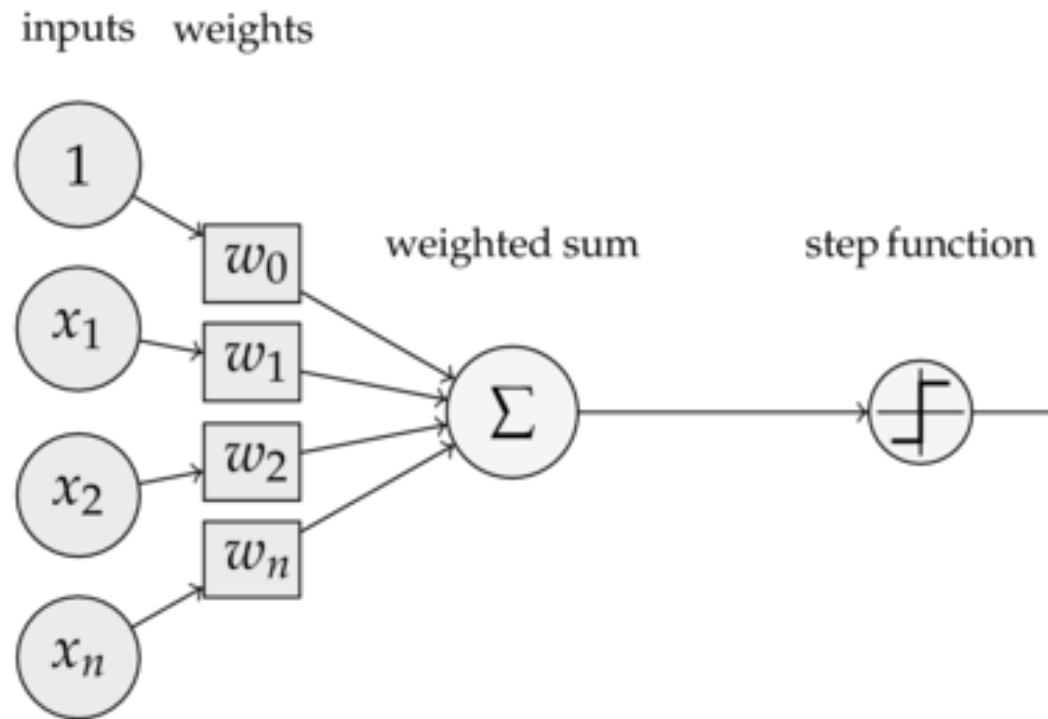


CS 4700: Foundations of Artificial Intelligence

Fall 2019
Prof. Haym Hirsh

Lecture 14
November 4, 2019

Perceptrons



<https://blog.dbrgn.ch/2013/3/26/perceptrons-in-python/>

Learning Linear Classifiers: The Perceptron

Goal of classification learning:

Given: $((1, x_{11}, x_{12}, \dots, x_{1n}), y_1), ((1, x_{21}, x_{22}, \dots, x_{2n}), y_2), \dots, ((1, x_{N1}, x_{N2}, \dots, x_{Nn}), y_N)$
 $\bar{x}_1 \qquad \qquad \qquad \bar{x}_2 \qquad \qquad \qquad \bar{x}_N$

Find: $(w_0, \dots, w_n)$

Linear Classification

General form of a linear classifier

$$h_{\bar{w}}(\bar{x}) = \begin{array}{ll} +1 & \text{if } \sum_{i=0}^n w_i x_i = \bar{w} \cdot \bar{x} \geq 0 \\ 0 & \text{otherwise} \end{array}$$

Finding a classifier given a set of data means finding a $\bar{w}$ for which $h_{\bar{w}}(\bar{x}_i)$ comes close to $y_i = f_{\bar{w}}(\bar{x}_i)$ for the training data

How?

Search

Learning with Perceptrons

- Start with an initial $\bar{w}$.
- Take an example $\langle \bar{x}_i, y_i \rangle$
 - If $h_{\bar{w}}(\bar{x}_i) = y_i$ do nothing
 - Otherwise move $\bar{w}$ to make $\bar{w} \cdot \bar{x}$
bigger (if $y_i = 1$)
smaller (if $y_i = 0$)
- Repeat until you get all the labels right

Learning with Perceptrons

- Start with an initial $\bar{w}$.
- Take an example $\langle \bar{x}_i, y_i \rangle$
 - If $h_{\bar{w}}(\bar{x}_i) = y_i$ do nothing
 - Otherwise move $\bar{w}$ to make $\bar{w} \cdot \bar{x}$
bigger (if $y_i = 1$)
smaller (if $y_i = 0$)
- Repeat until you get all the labels right

move $\bar{w}$?

Learning with Perceptrons

- Start with an initial $\bar{w}$.
- Take an example $\langle \bar{x}_i, y_i \rangle$
 - If $h_{\bar{w}}(\bar{x}_i) = y_i$ do nothing
 - Otherwise move $\bar{w}$ to make $\bar{w} \cdot \bar{x}$
bigger (if $y_i = 1$)
smaller (if $y_i = 0$)
- Repeat until you get all the labels right

move $\bar{w}$?

$$w_j \leftarrow w_j + \alpha x_{ij}(y_i - h_{\bar{w}}(\bar{x}_i))$$

Perceptron Learning Rule

Perceptron Learning Rule

$$w_j \leftarrow w_j + \alpha x_{ij}(y_i - h_{\bar{w}}(\bar{x}_i))$$

- If $h_{\bar{w}}(\bar{x}_i)$ is correct, all w_j are unchanged
 - $y_i = h_{\bar{w}}(\bar{x}_i)$, so $(y_i - h_{\bar{w}}(\bar{x}_i)) = 0$
- If $h_{\bar{w}}(\bar{x}_i)$ is too big, w_j decreases
- If $h_{\bar{w}}(\bar{x}_i)$ is too small, w_j increases
- α is the *learning rate* (sometimes called η)

Perceptron Learning Rule

Current hypothesis: $h_{\bar{w}}(\bar{x})$

$$w_0 = w_1 = w_2 = \dots = w_n = 0$$

[alternatively: set to random values]

Repeat

For $i = 1$ to N

[for each example]

$$h \leftarrow h_{\bar{w}}(\bar{x}_i)$$

For $j = 0$ to n

[for each feature]

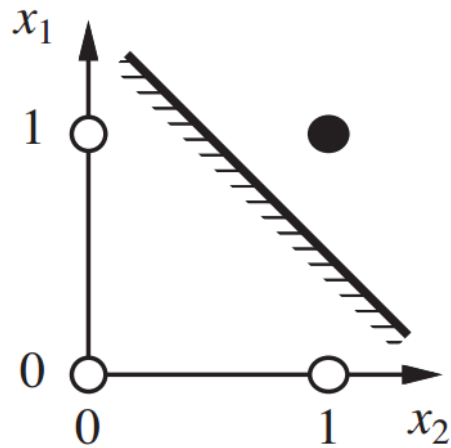
$$w_j \leftarrow w_j + \alpha x_{ij}(y_i - h)$$

[Perceptron learning rule]

Until $h_{\bar{w}}(\bar{x})$ gets all data correct

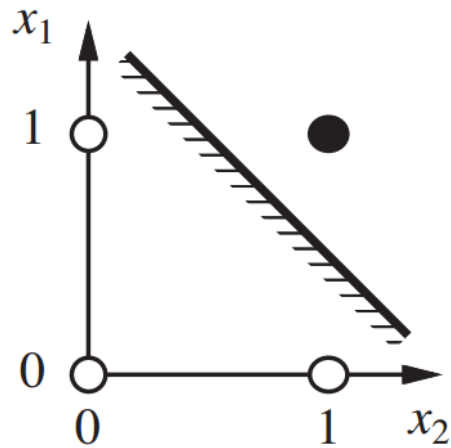
Perceptron Learning Rule: Example

$$w_j \leftarrow w_j + \alpha x_{ij}(y_i - h)$$



Perceptron Learning Rule: AND Example

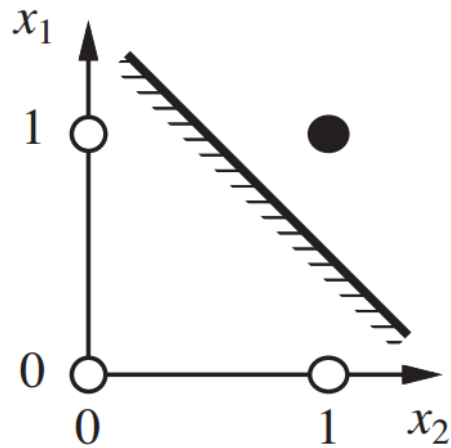
$$w_j \leftarrow w_j + \alpha x_{ij}(y_i - h)$$



x_1	x_2	$f(x_1, x_2)$
0	0	0
0	1	0
1	0	0
1	1	1

Perceptron Learning Rule: AND Example

$$w_j \leftarrow w_j + \alpha x_{ij}(y_i - h)$$



And gate

x_1	x_2	$f(x_1, x_2)$
0	0	0
0	1	0
1	0	0
1	1	1

Perceptron Learning Rule: AND Example

$$w_j \leftarrow w_j + \alpha x_{ij}(y_i - h)$$
$$\alpha = 0.3, \quad w_0 = w_1 = w_2 = 0$$

Training Data

x_1	x_2	$f(x_1, x_2)$
0	0	0
0	1	0
1	0	0
1	1	1

<https://docs.google.com/spreadsheets/d/1BHVKFw1nO1slluhs-3Ry4hL64B4Ja15WaE335jmLQXA>

Excel Version (Uses Variable Names):

<https://drive.google.com/file/d/1Yv1ldsBGKojytTRS97DtZKrJujTogmVb/view>

Perceptron Learning Rule: 2nd Example

$$w_j \leftarrow w_j + \alpha x_{ij}(y_i - h)$$

$$\alpha = 0.3, \quad w_0 = w_1 = w_2 = 0$$

Training Data

x_1	x_2	$f(x_1, x_2)$
-1	0.5	1
-1	-1	1
1	0.5	0
0.5	1	0

Perceptron Learning Rule: 3rd Example

$$w_j \leftarrow w_j + \alpha x_{ij}(y_i - h)$$
$$\alpha = 0.3, \quad w_0 = w_1 = w_2 = 0$$

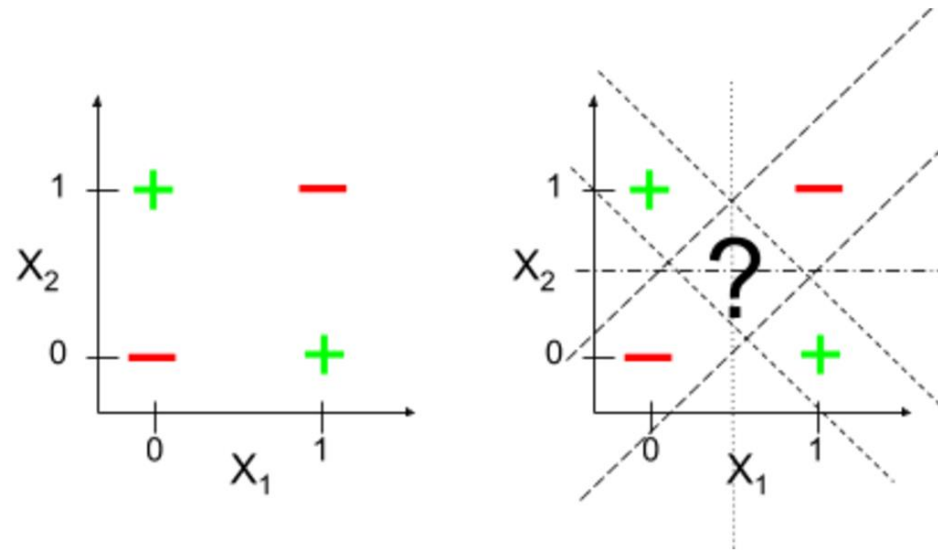
Training Data

x_1	x_2	$f(x_1, x_2)$
0	0	0
0	1	1
1	0	1
1	1	0

Perceptron Convergence Theorem

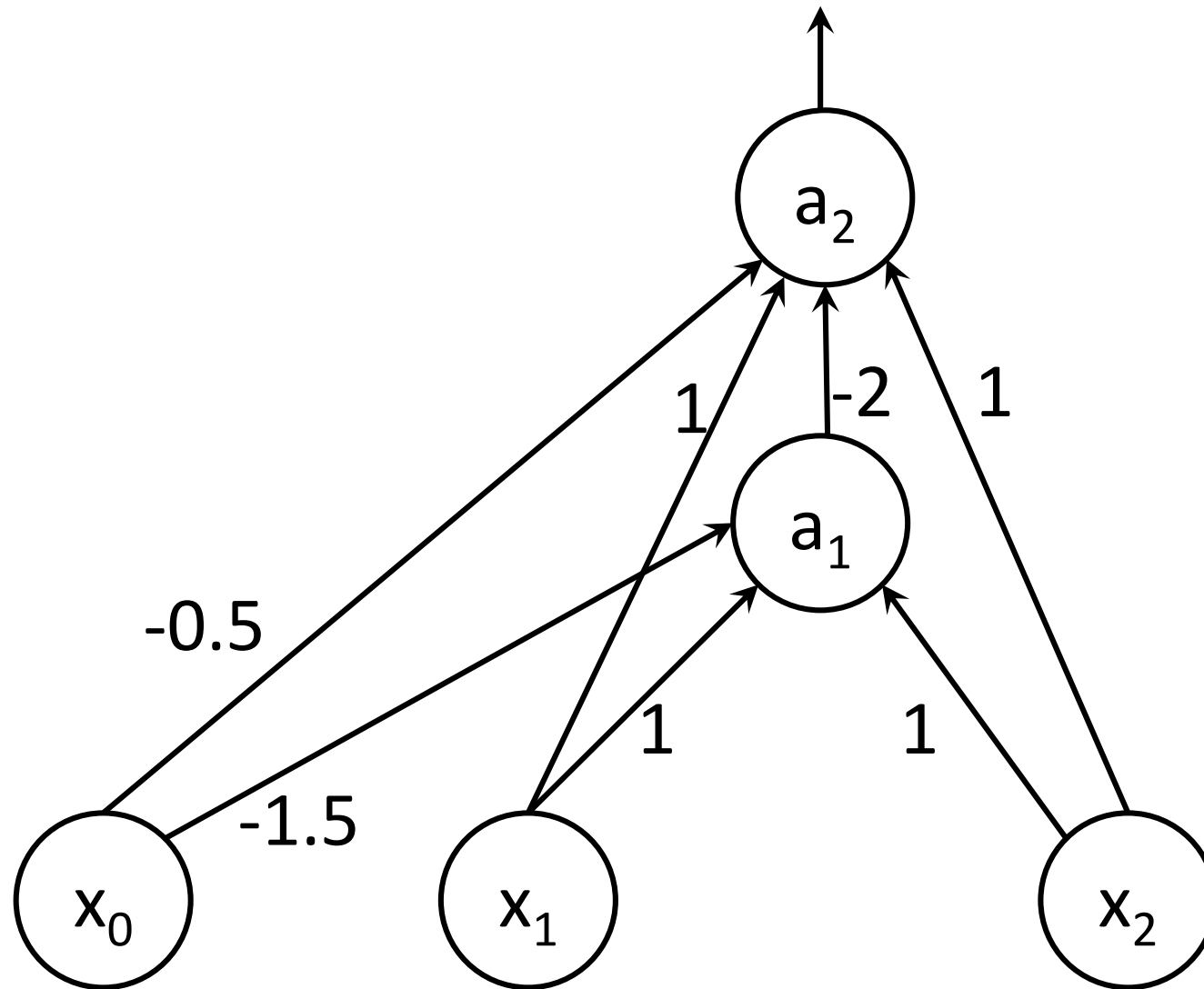
If the training data are linearly separable (there exists a hyperplane that separates the 0's and 1's) then the perceptron learning rule ($0 < \alpha \leq 1$) will find a separating hyperplane in a finite number of steps.

Perceptrons: The Problem with XOR

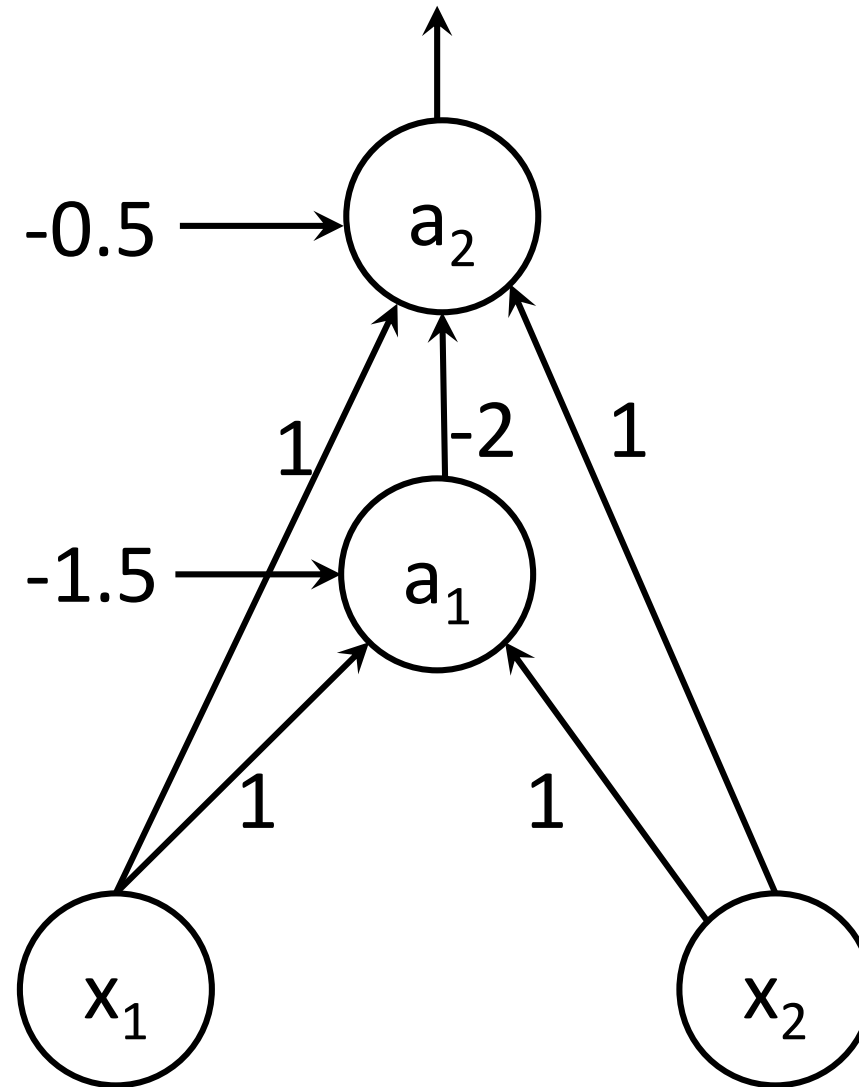


(<https://medium.com/@claude.coulombe/the-revenge-of-perceptron-learning-xor-with-tensorflow-eb52cbdf6c60>)

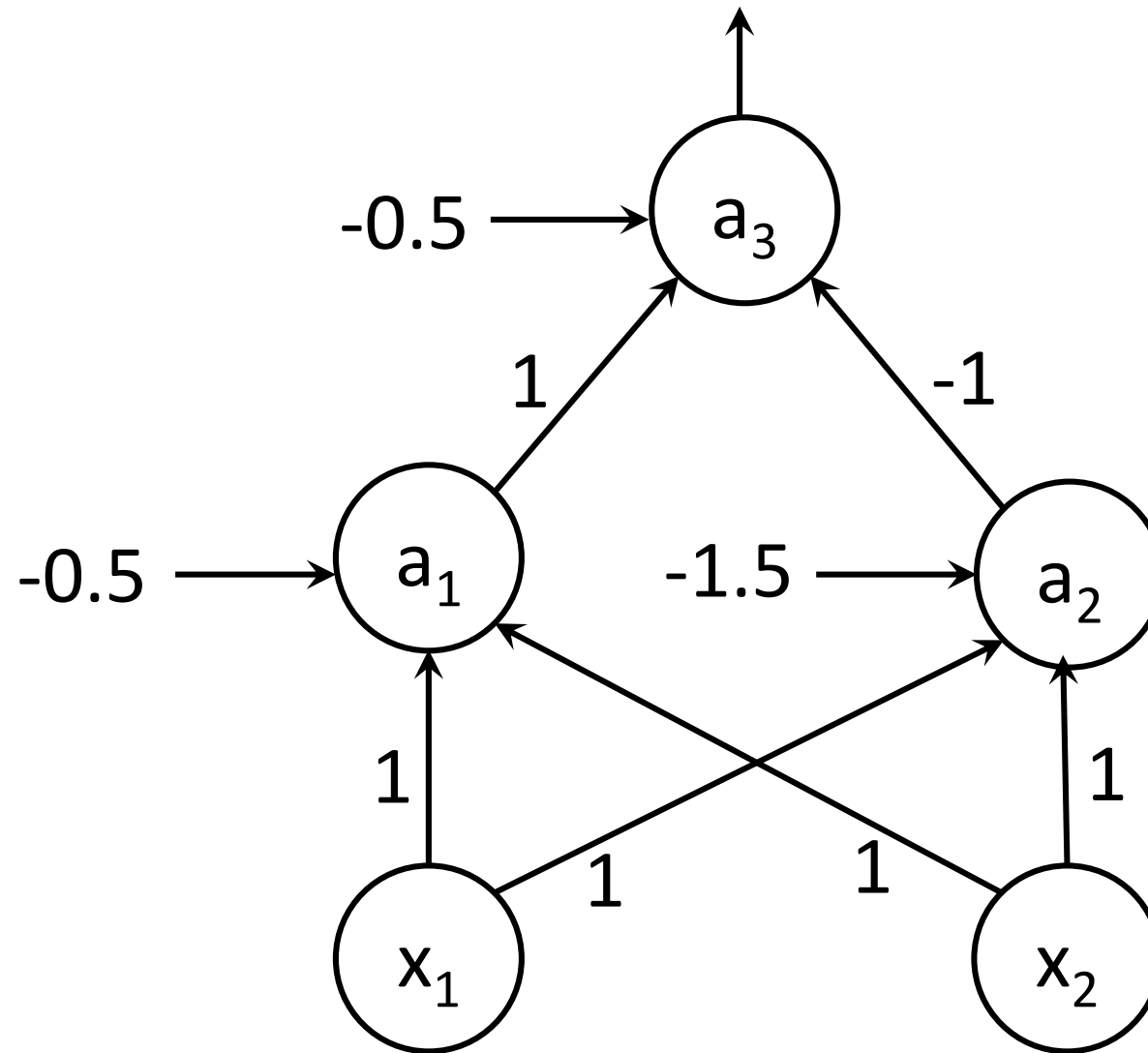
Perceptron: The Solution for XOR?



Perceptron: The Solution for XOR?

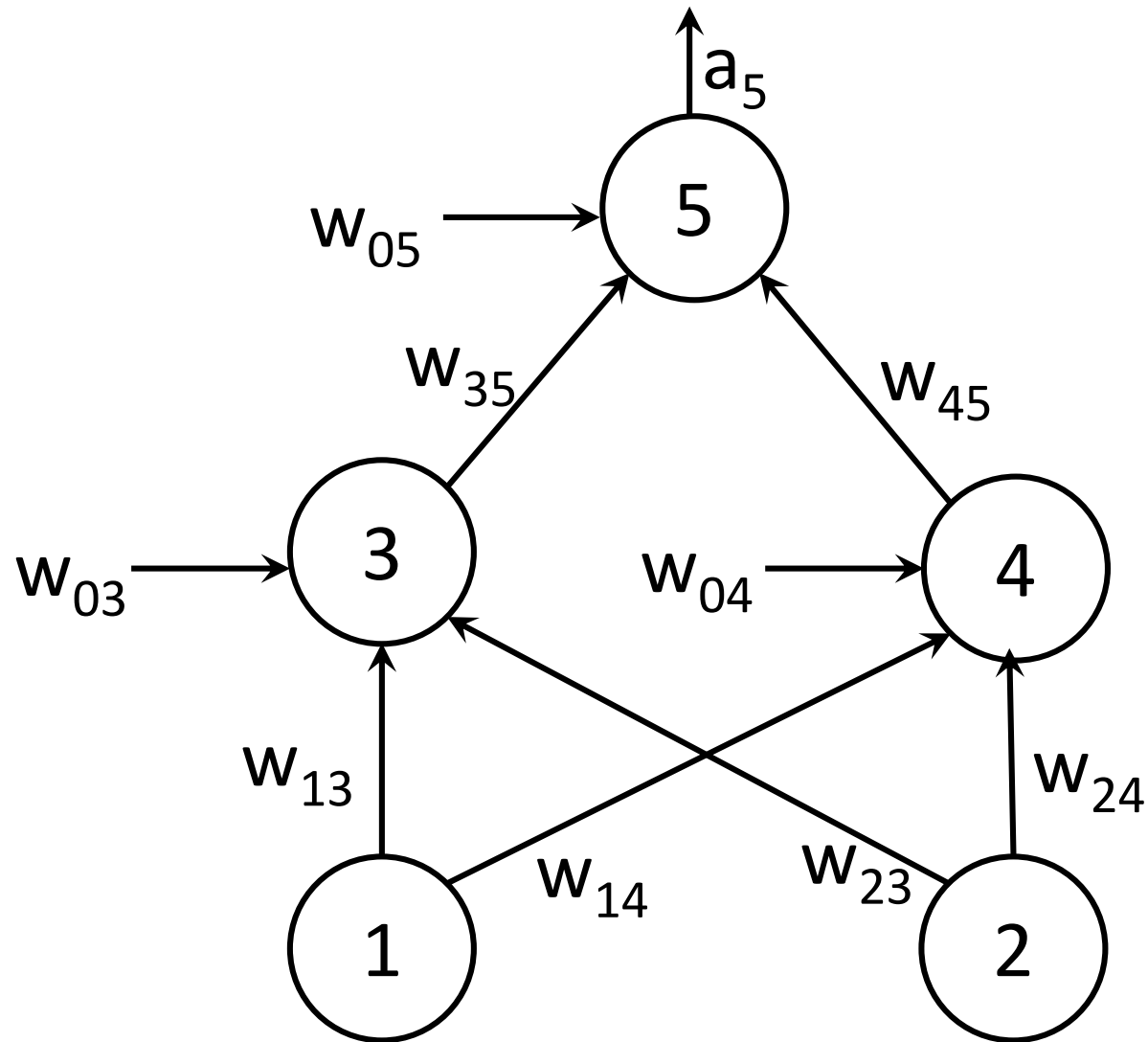


Perceptron: The Solution for XOR?



Multi-layer
Neural Network

Multi-Layer Perceptron

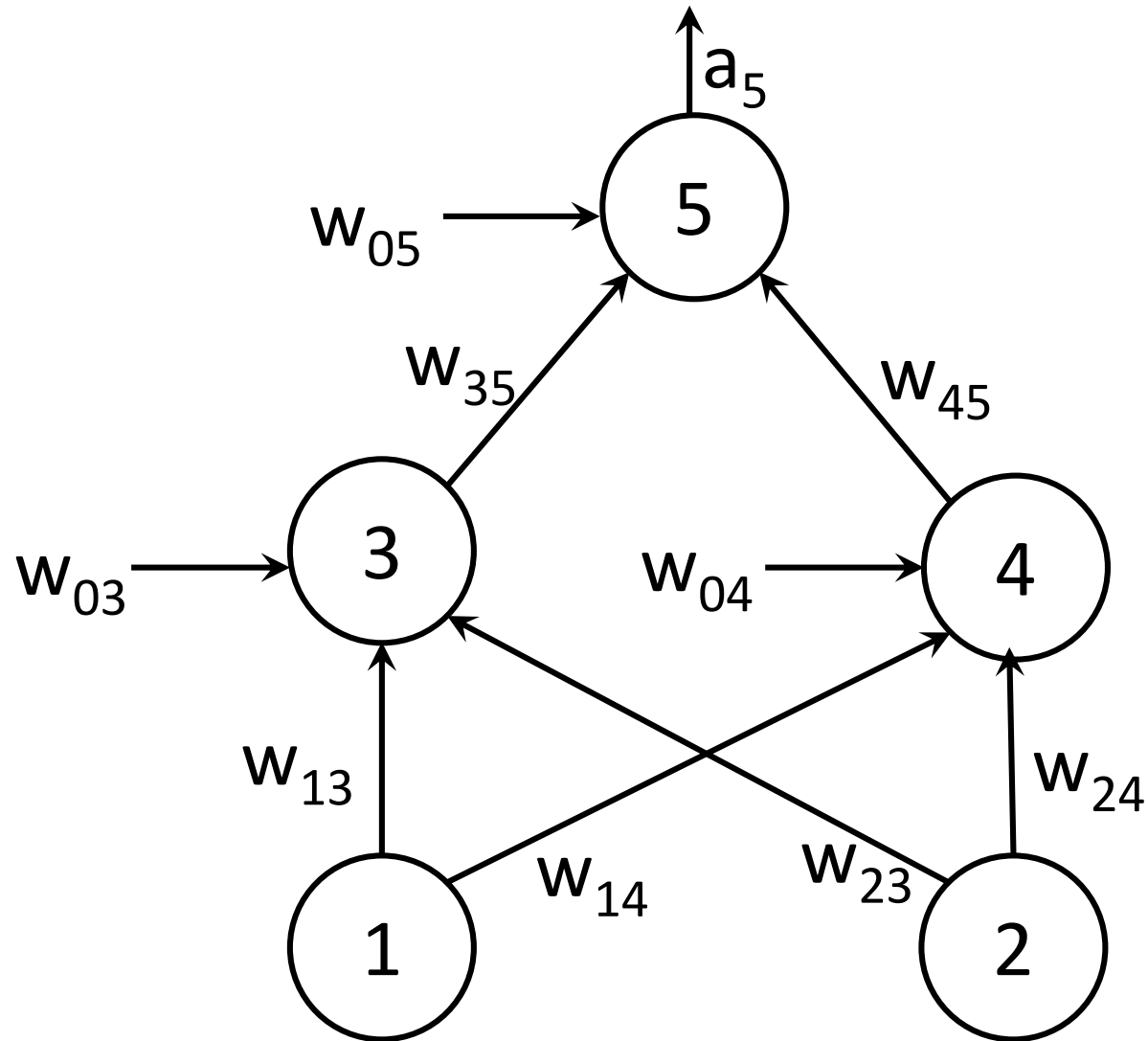


Multi-layer
Neural Network

Recap

- We know how to train a single perceptron using the perceptron learning rule
- A perceptron can't represent XOR (and many other functions)
- Use multiple layers
- The perceptron learning rule could be used to train the top-most level of the network, but it's unclear what to do with lower levels

Training Multi-Layer Perceptrons?

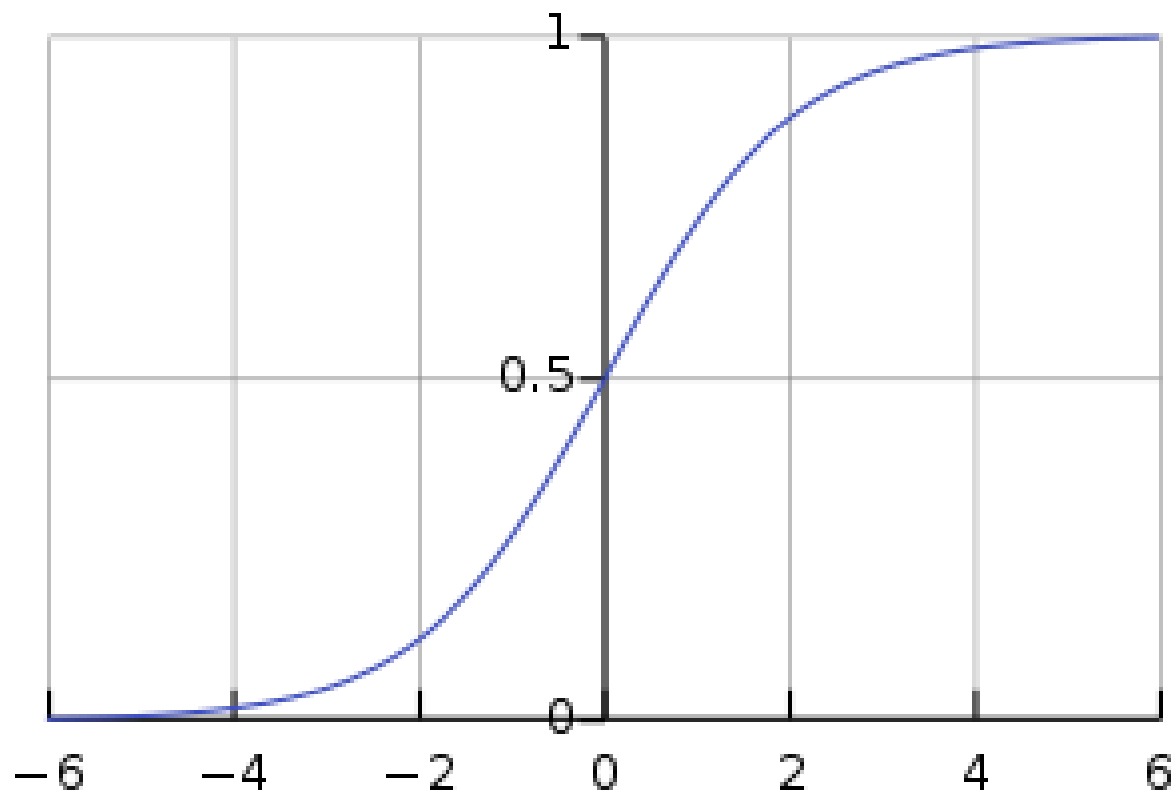


Multi-layer
Neural Network

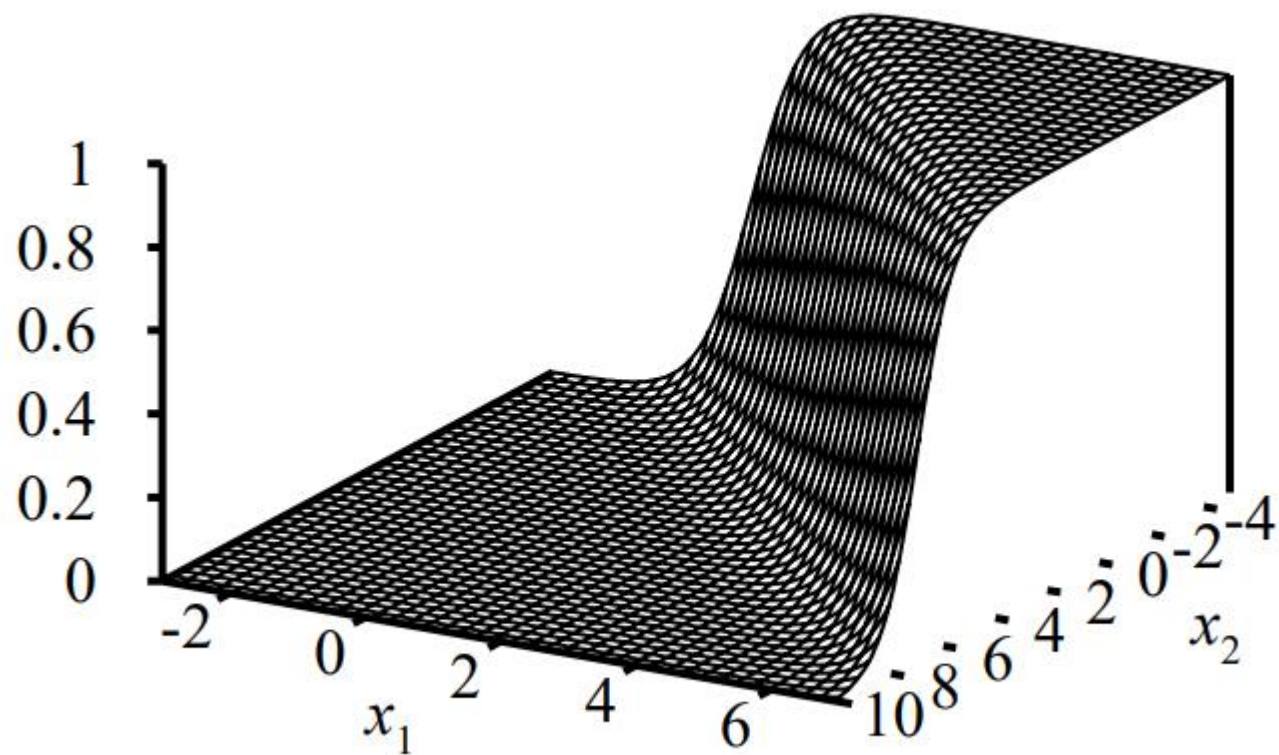
Solution

Replace the hard threshold with a continuous function
(and use calculus)

$$\frac{1}{1 + e^{-z}}$$



$$\frac{1}{1 + e^{-\bar{w} \cdot \bar{x}}}$$



Learning

- Consider $\text{Error}(\bar{w}, D)$:

$$\sum_{i=1}^N (y_i - h_{\bar{w}}(\bar{x}_i))^2$$

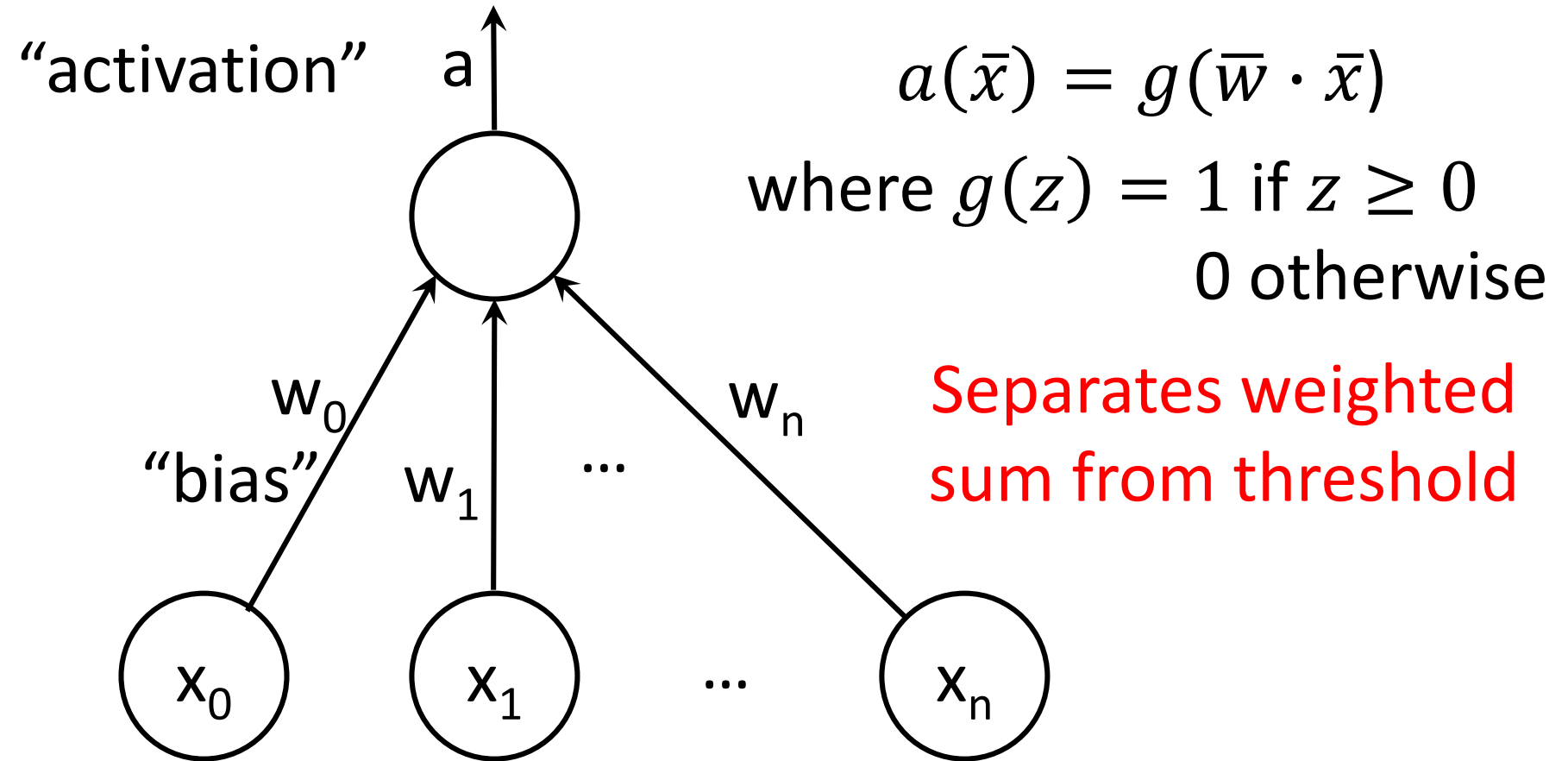
Learning

- Consider $\text{Error}(\bar{w}, D)$:

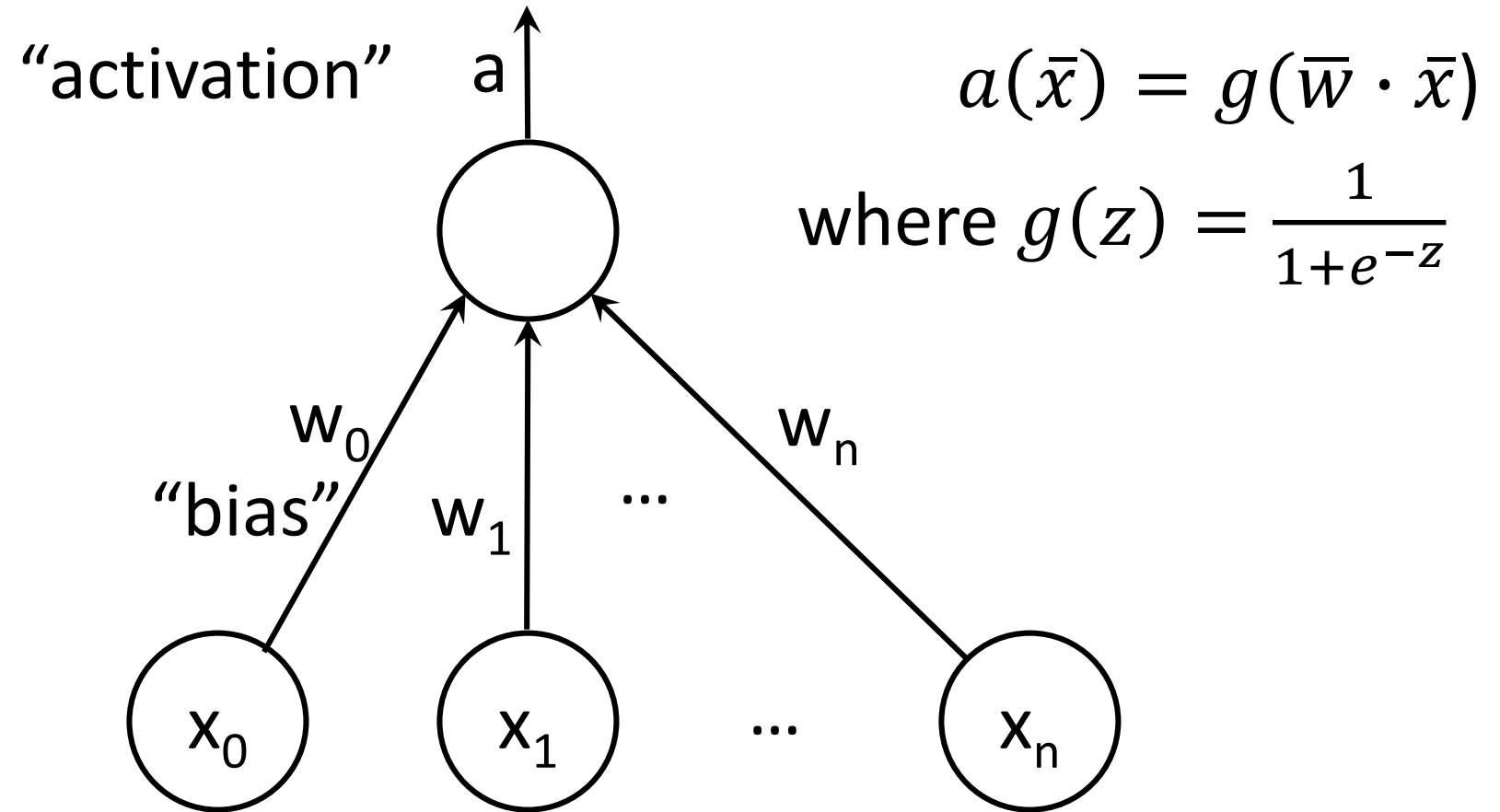
$$\sum_{i=1}^N (y_i - h_{\bar{w}}(\bar{x}_i))^2$$

- For a given set of data D , the $\bar{x}_i$'s are fixed
- Different $\bar{w}$ give different values for $\text{Error}(\bar{w}, D)$
- We want to minimize $\text{Error}(\bar{w}, D)$ by changing the $\bar{w}_i$'s
- We'll do this incrementally by following the derivative of $\text{Error}(\bar{w}, D)$

Abstract Representation of a Perceptron



Abstract Representation of a Logistic Neuron



Error_D($\bar{w}$) and Its Gradient

$$\text{Error}_D(\bar{w}) = \sum_{\bar{x}_i \in D} (y_i - h_{\bar{w}}(\bar{x}_i))^2 = \sum_{\bar{x}_i \in D} (y_i - g(\bar{w} \cdot \bar{x}_i))^2$$

$$\begin{aligned} \frac{\partial}{\partial w_j} \text{Error}_D(\bar{w}) &= \frac{\partial}{\partial w_j} \sum_{\bar{x}_i \in D} (y_i - g(\bar{w} \cdot \bar{x}_i))^2 \\ &= \sum_{\bar{x}_i \in D} \frac{\partial}{\partial w_j} (y_i - g(\bar{w} \cdot \bar{x}_i))^2 = -2 \sum_{\bar{x}_i \in D} g'(\bar{w} \cdot \bar{x}_i) (y_i - g(\bar{w} \cdot \bar{x}_i)) \bar{x}_{ij} \end{aligned}$$

Logistic Regression $\text{Error}_D(\bar{w})$ and Its Gradient

$$\frac{\partial}{\partial w_j} \text{Error}_D(\bar{w}) = -2 \sum_{\bar{x}_i \in D} g'(\bar{w} \cdot \bar{x}_i) (y_i - g(\bar{w} \cdot \bar{x}_i)) \bar{x}_{ij}$$

$$g'(z) = g(z)(1 - z)$$

for $g(z) = \text{logistic function}$

$$\frac{\partial}{\partial w_j} \text{Error}_D(\bar{w}) = -2 \sum_{\bar{x}_i \in D} g(\bar{w} \cdot \bar{x}_i) (1 - g(\bar{w} \cdot \bar{x}_i)) (y_i - g(\bar{w} \cdot \bar{x}_i)) \bar{x}_{ij}$$

Perceptron Learning Algorithm

Current hypothesis: $h_{\bar{w}}(\bar{x})$

Initialize $w_0, w_1, w_2, \dots, w_n$

Repeat

For $i = 1$ to N [for each example]

$h \leftarrow h_{\bar{w}}(\bar{x}_i)$

For $j = 0$ to n [for each feature]

$w_j \leftarrow w_j + \alpha x_{ij}(y_i - h)$

Until $h_{\bar{w}}(\bar{x})$ gets all data correct

Gradient Descent Algorithm

Current hypothesis: $h_{\bar{w}}(\bar{x}) = g(\bar{w} \cdot \bar{x})$

Initialize $w_0, w_1, w_2, \dots, w_n$

Repeat

For $j = 0$ to n [for each feature]

$$\Delta w \leftarrow 0$$

For $i = 1$ to N [for each example]

$$\Delta w \leftarrow \Delta w + g'(\bar{w} \cdot \bar{x}_i)(y_i - g(\bar{w} \cdot \bar{x}_i))\bar{x}_{ij}$$

$$w_j \leftarrow w_j + \alpha \Delta w$$

The -2 is captured in α

[Randomly reorder the data]

Until stopping condition