Machine Learning II

Neural Networks

Souhaib Ben Taieb

March 20, 2022

University of Mons

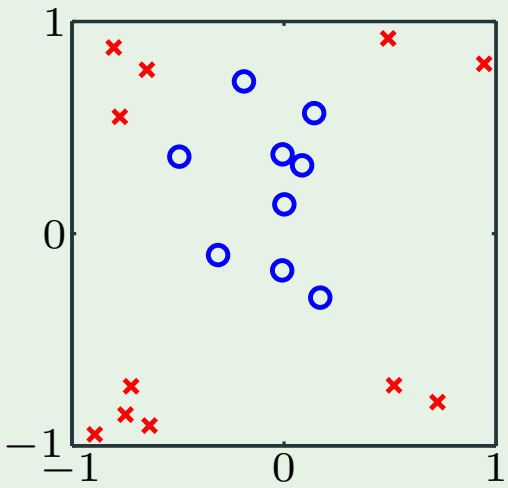
Table of contents

Table of contents

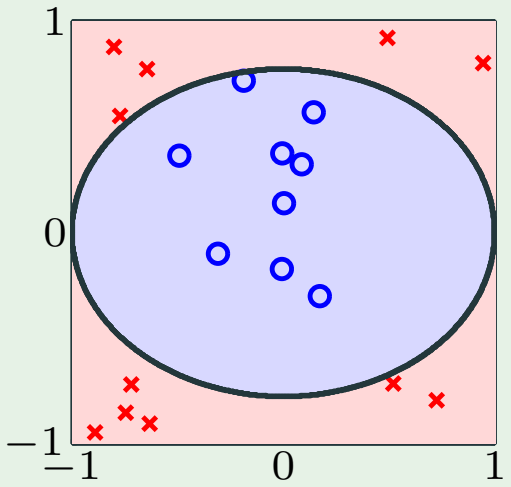
Linear is limited

Linear is limited

Data:



Hypothesis:



Linear in what?

Linear in what?

Linear regression implements

$$\sum_{i=0}^{d} w_i \, x_i$$

Linear classification implements

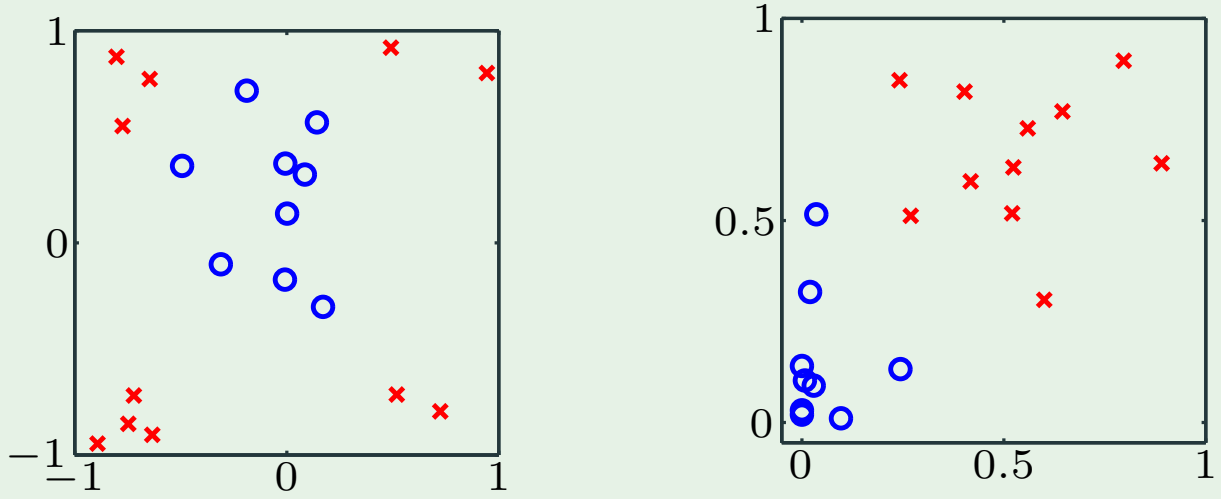
$$\text{sign}\left(\sum_{i=0}^{d} w_i \, x_i\right)$$

Algorithms work because of **linearity in the weights**

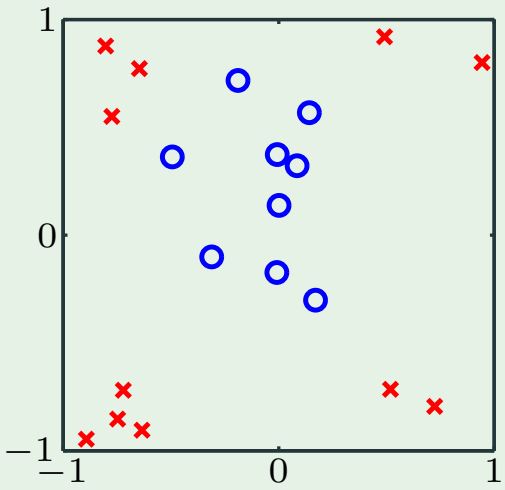
Nonlinear transformation

Transform the data nonlinearly

$$(x_1, x_2) \xrightarrow{\Phi} (x_1^2, x_2^2)$$

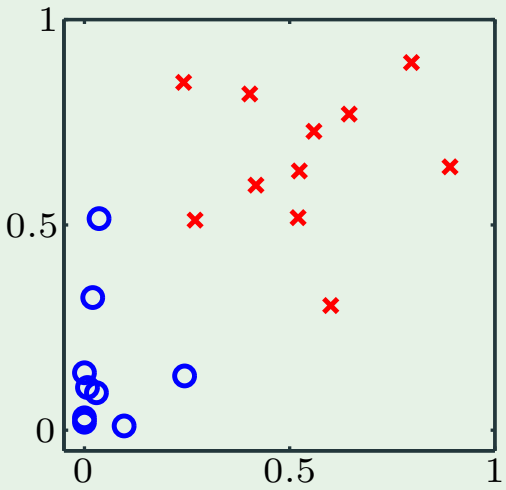


Nonlinear transformation

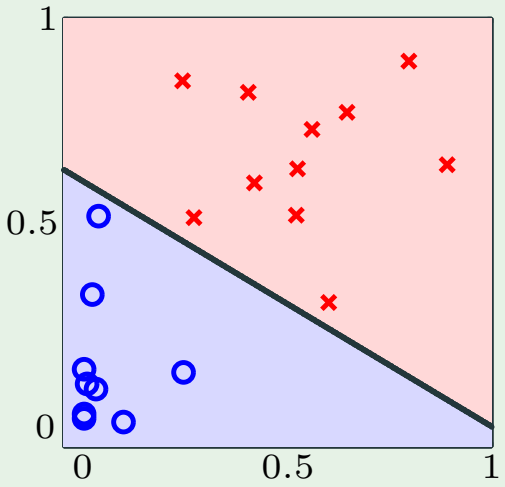


1. Original data
$\mathbf{x}_n \in \mathcal{X}$

Φ

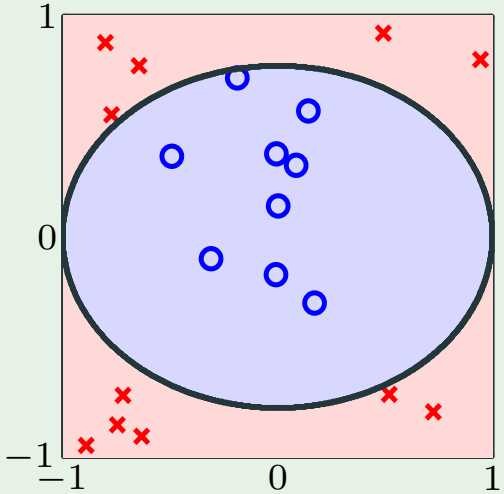


2. Transform the data
$\mathbf{z}_n = \Phi(\mathbf{x}_n) \in \mathcal{Z}$



3. Separate data in $\mathcal{Z}$-space
$\tilde{g}(\mathbf{z}) = \text{sign}(\tilde{\mathbf{w}}^\mathsf{T}\mathbf{z})$

'Φ^{-1}'



4. Classify in $\mathcal{X}$-space
$g(\mathbf{x}) = \tilde{g}(\Phi(\mathbf{x})) = \text{sign}(\tilde{\mathbf{w}}^\mathsf{T}\Phi(\mathbf{x}))$

What transforms to what

What transforms to what

$$\mathbf{x} = (x_0, x_1, \cdots, x_d) \xrightarrow{\Phi} \mathbf{z} = (z_0, z_1, \cdots\cdots\cdots, z_{\tilde{d}})$$

$$\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N \xrightarrow{\Phi} \mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_N$$

$$y_1, y_2, \cdots, y_N \xrightarrow{\Phi} y_1, y_2, \cdots, y_N$$

No weights in $\mathcal{X}$ $\qquad$ $\tilde{\mathbf{w}} = (w_0, w_1, \cdots\cdots\cdots, w_{\tilde{d}})$

$$g(\mathbf{x}) = \operatorname{sign}(\tilde{\mathbf{w}}^{\mathsf{T}}\Phi(\mathbf{x}))$$

Nonlinear transforms

$$\mathbf{x} = (x_0, x_1, \cdots, x_d) \quad \xrightarrow{\Phi} \quad \mathbf{z} = (z_0, z_1, \cdots\cdots\cdots\cdots, z_{\tilde{d}})$$

Each $z_i = \phi_i(\mathbf{x})$ $\qquad$ $\mathbf{z} = \Phi(\mathbf{x})$

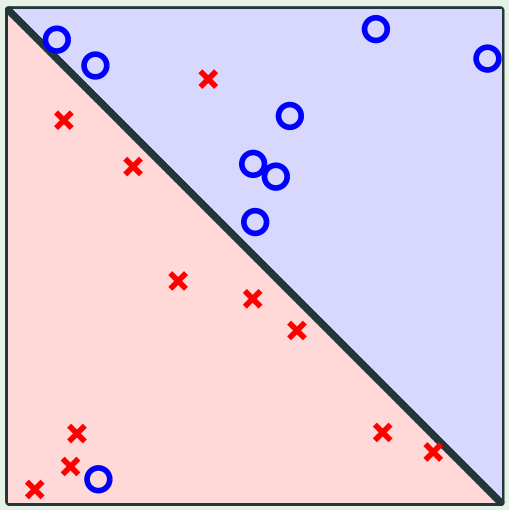
Example: $\mathbf{z} = (1, x_1, x_2, x_1x_2, x_1^2, x_2^2)$

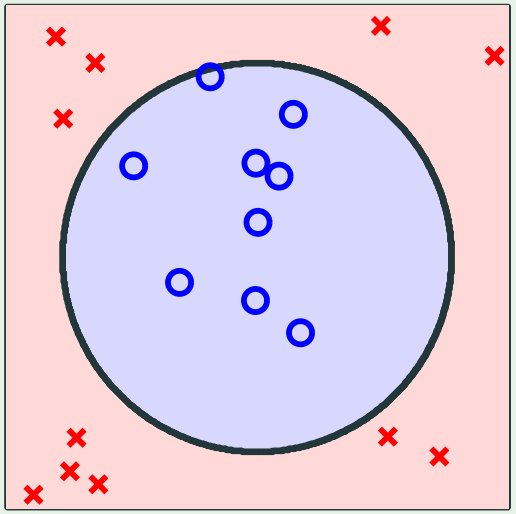
Final hypothesis $g(\mathbf{x})$ in $\mathcal{X}$ space:

$$\text{sign}\left(\tilde{\mathbf{w}}^\mathsf{T}\Phi(\mathbf{x})\right) \qquad \text{or} \qquad \tilde{\mathbf{w}}^\mathsf{T}\Phi(\mathbf{x})$$

Two cases

Two non-separable cases



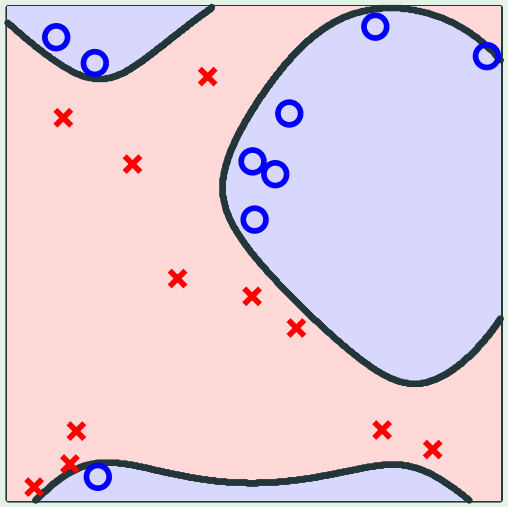


First case

Use a linear model in $\mathcal{X}$; accept $E_{\text{in}} > 0$

or

Insist on $E_{\text{in}} = 0$; go to high-dimensional $\mathcal{Z}$



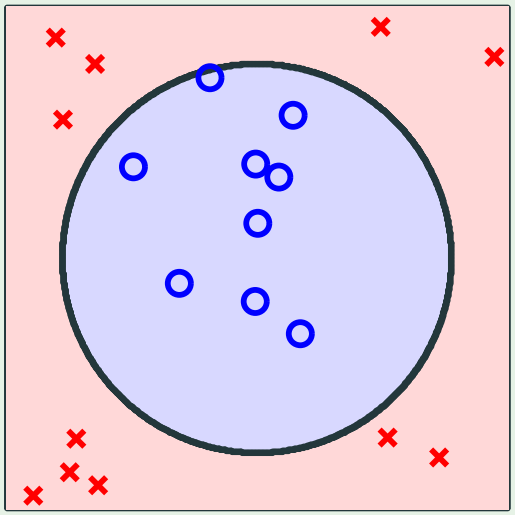
Second case

$\mathbf{z} = (1, x_1, x_2, x_1x_2, x_1^2, x_2^2)$

Why not: $\mathbf{z} = (1, x_1^2, x_2^2)$

or better yet: $\mathbf{z} = (1, x_1^2 + x_2^2)$

or even: $\mathbf{z} = (x_1^2 + x_2^2 - 0.6)$



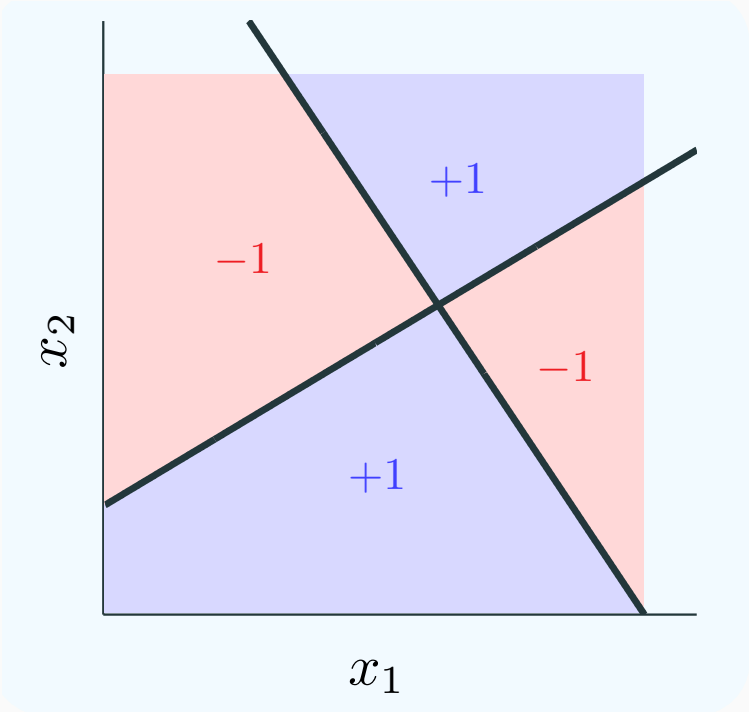
Lesson learned

Looking at the data *before* choosing the model can be hazardous to your E_{out}

Data snooping

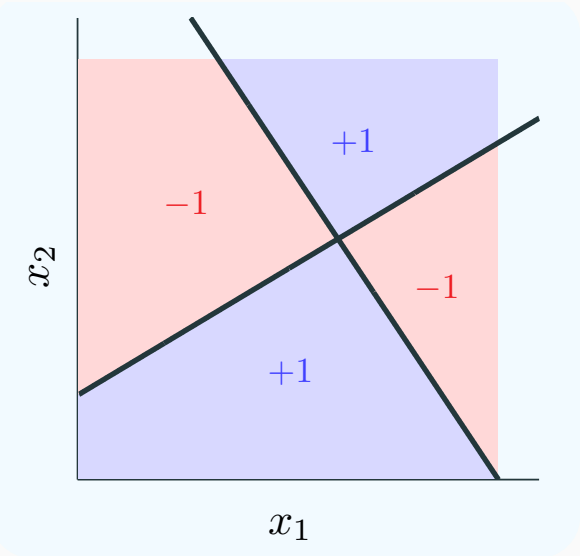
Table of contents

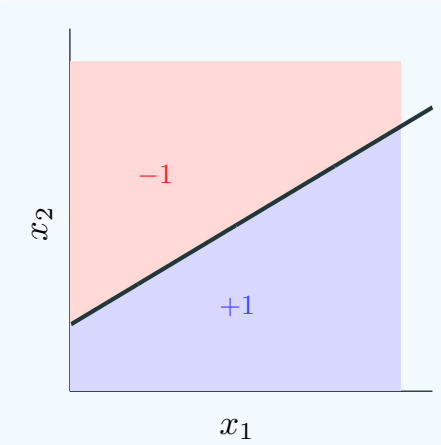
Example



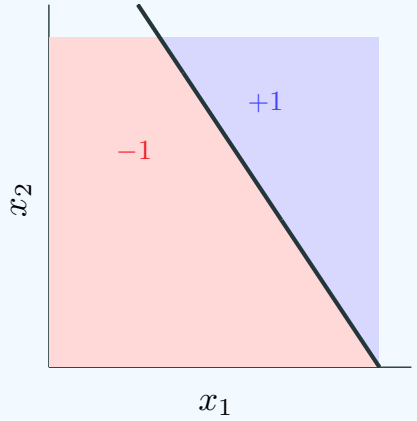
Can we represent this hypothesis with perceptrons?

Example





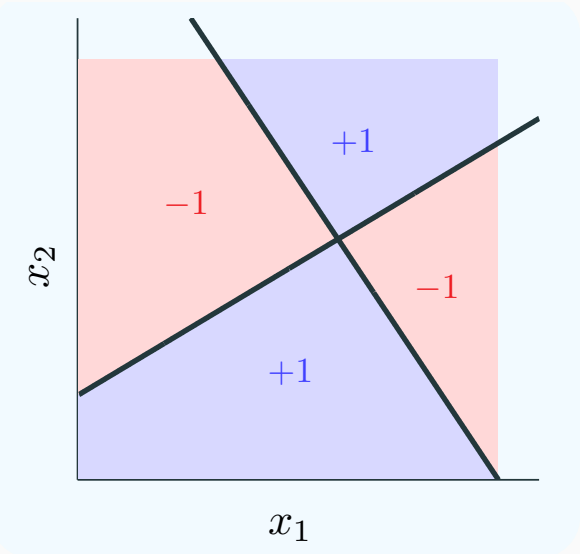
$h_1(\mathbf{x}) = \text{sign}(\mathbf{w}_1^{\text{T}}\mathbf{x})$

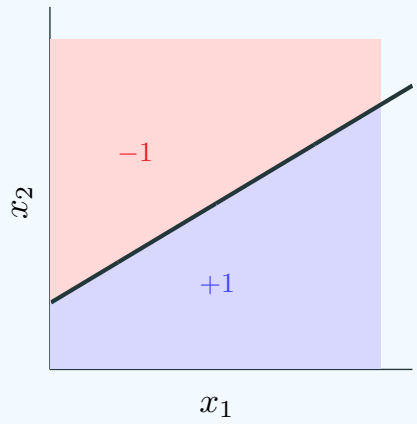


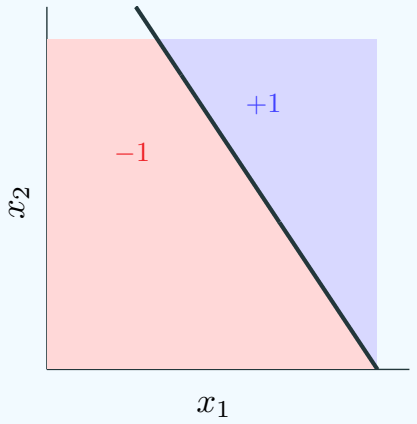
$h_2(\mathbf{x}) = \text{sign}(\mathbf{w}_2^{\text{T}}\mathbf{x})$

How to represent $f(x_1, x_2)$, with two perceptrons, $h_1(x_1, x_2)$ and $h_2(x_1, x_2)$?

Example







How to represent $f(x_1, x_2)$, with two perceptrons, $h_1(x_1, x_2)$ and $h_2(x_1, x_2)$?

$f(x_1, x_2) = +1$ when $h_1(x_1, x_2) = +1$ and $h_2(x_1, x_2) = -1$ or when $h_1(x_1, x_2) = -1$ and $h_2(x_1, x_2) = +1$. This is the Boolean XOR function: $f = \text{XOR}(h_1, h_2)$.

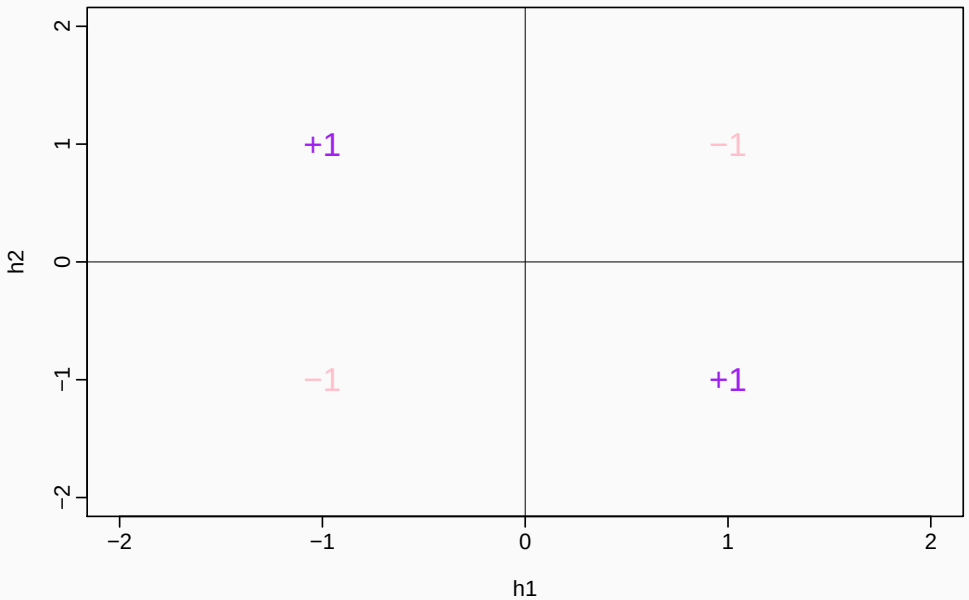
The XOR function

1 = TRUE and 0 = FALSE

h_1	h_2	XOR(h_1, h_2)
1	0	1
1	1	0
0	1	1
0	0	0

+1 = TRUE and -1 = FALSE

h_1	h_2	XOR(h_1, h_2)
+1	-1	+1
+1	+1	-1
-1	+1	+1
-1	-1	-1



Can we use a perceptron?

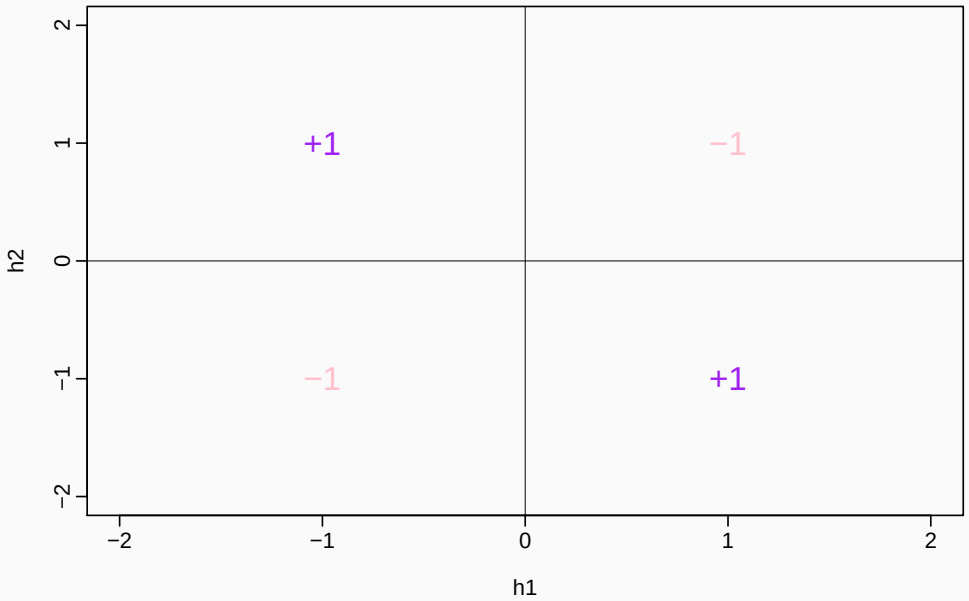
The XOR function

1 = TRUE and 0 = FALSE

h_1	h_2	XOR(h_1, h_2)
1	0	1
1	1	0
0	1	1
0	0	0

+1 = TRUE and -1 = FALSE

h_1	h_2	XOR(h_1, h_2)
+1	-1	+1
+1	+1	-1
-1	+1	+1
-1	-1	-1



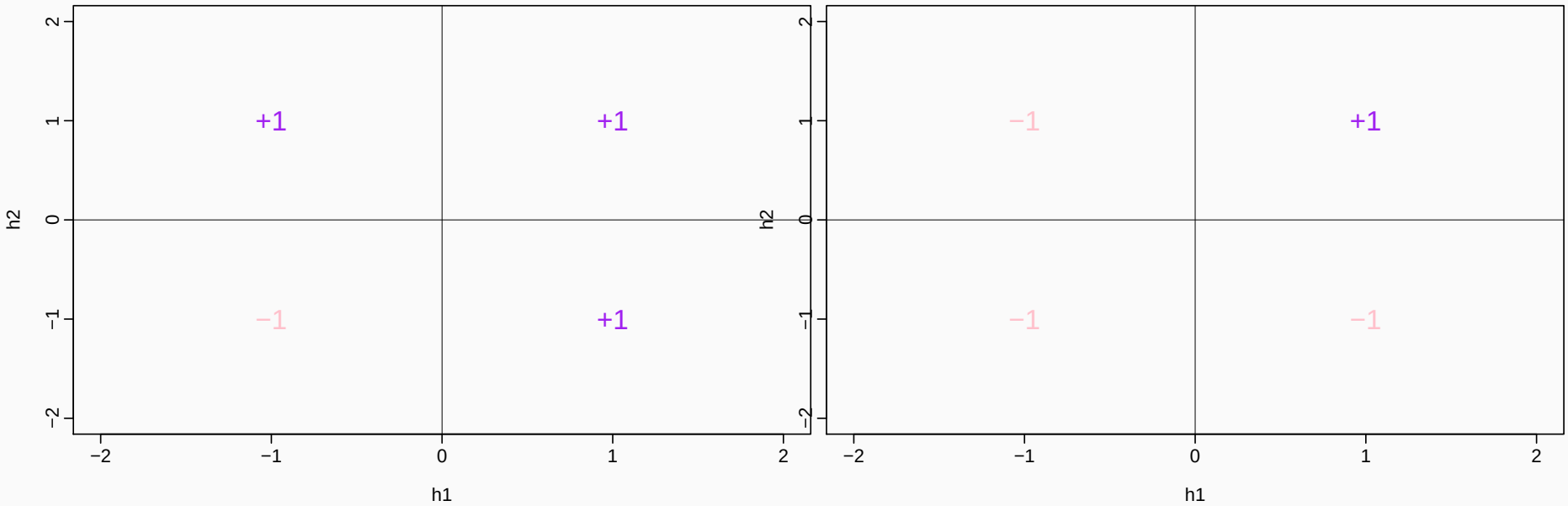
Can we use a perceptron? No!

Not linearly separable.

The OR and AND functions

h_1	h_2	OR(h_1, h_2)
+1	-1	+1
+1	+1	+1
-1	+1	+1
-1	-1	-1

h_1	h_2	AND(h_1, h_2)
+1	-1	-1
+1	+1	+1
-1	+1	-1
-1	-1	-1

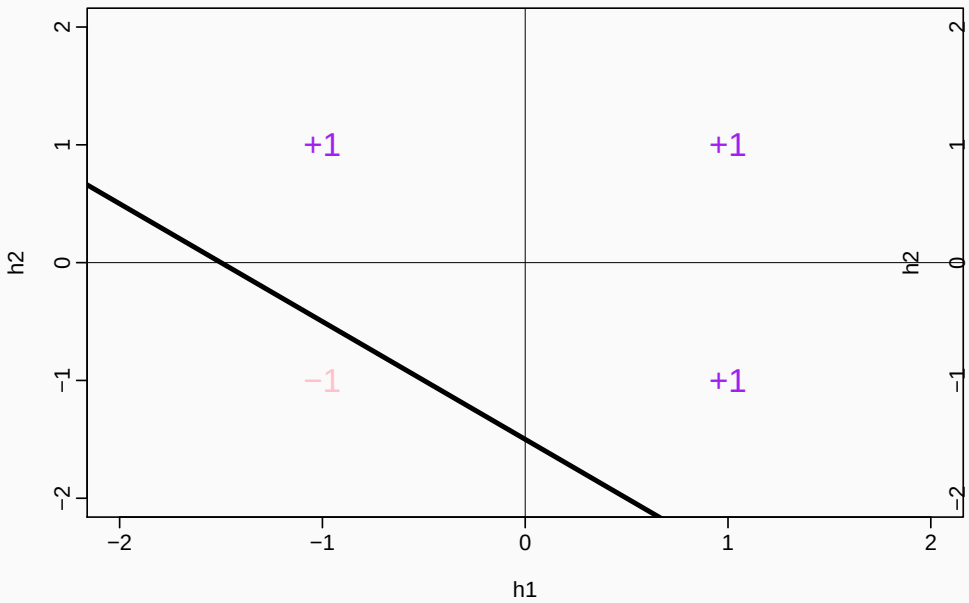


What are the perceptron weights for both OR and AND?

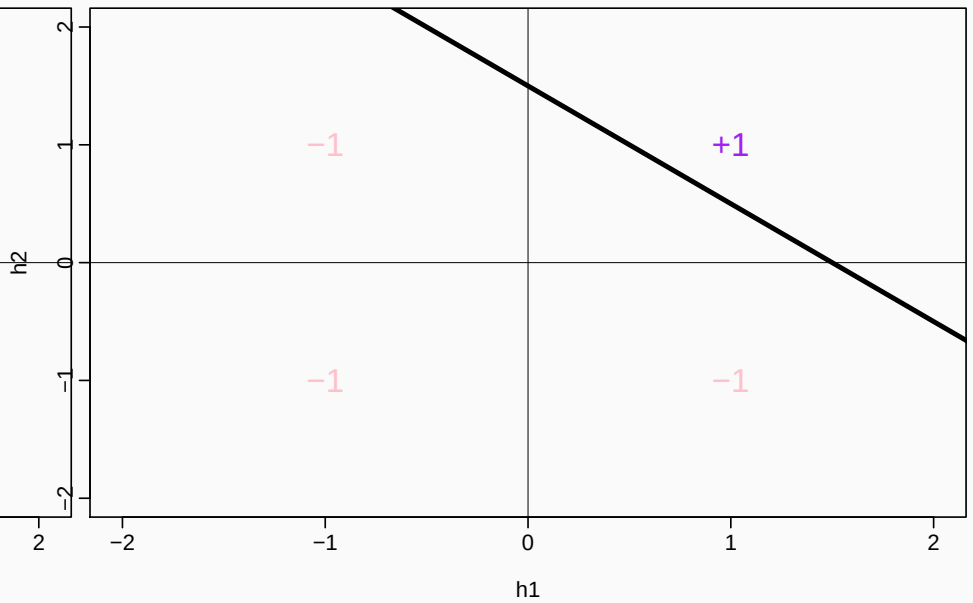
The OR and AND functions

h_1	h_2	OR(h_1, h_2)
+1	-1	+1
+1	+1	+1
-1	+1	+1
-1	-1	-1

h_1	h_2	AND(h_1, h_2)
+1	-1	-1
+1	+1	+1
-1	+1	-1
-1	-1	-1



OR(h_1, h_2) = sign(h_1 + h_2 + 1.5)



AND(h_1, h_2) = sign(h_1 + h_2 - 1.5)

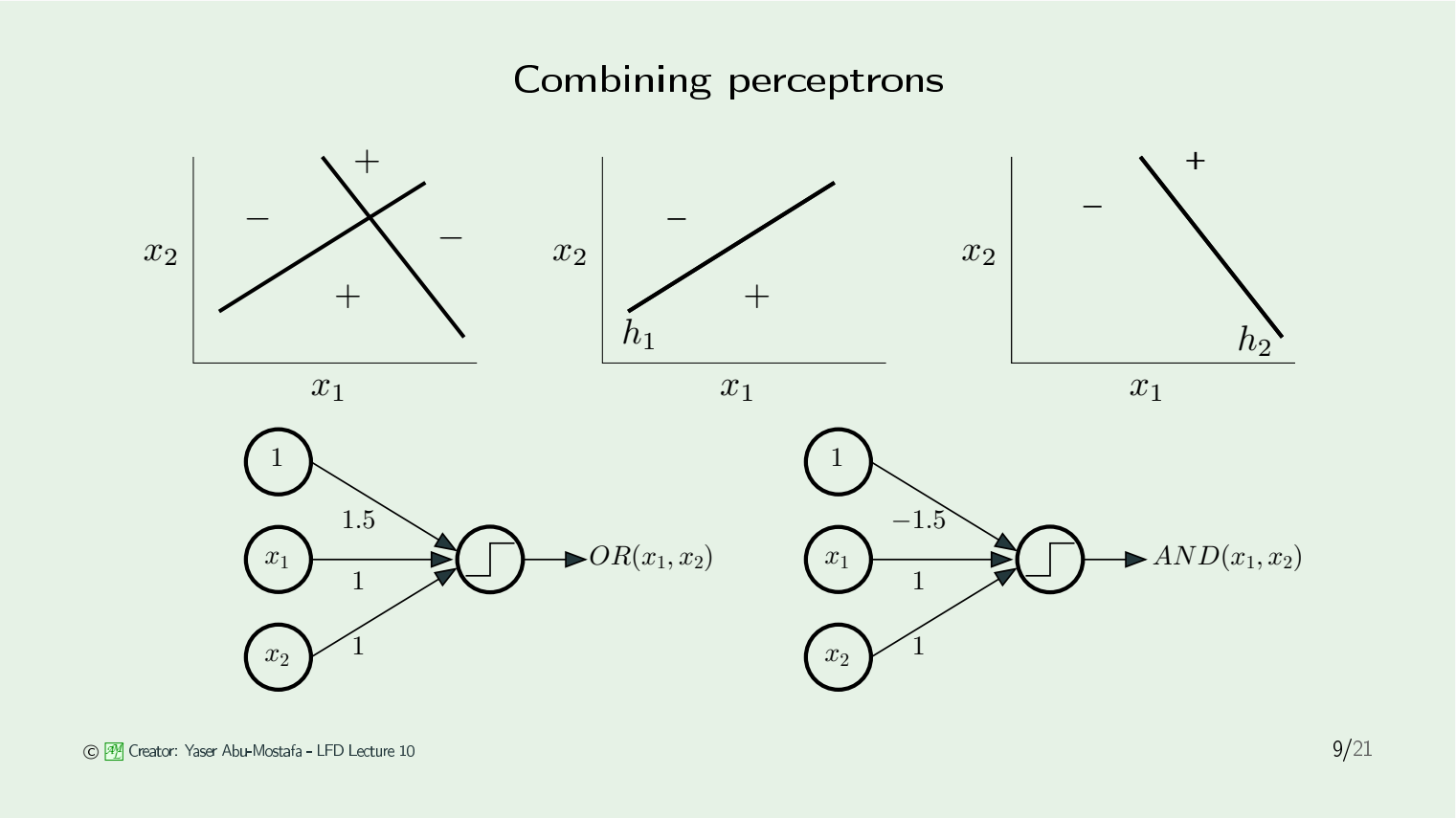
Back to the XOR function

- We can write f using the simpler OR and AND operations:

$$\begin{aligned} f &= \text{XOR}(h_1, h_2) \\ &= \text{OR}(\text{AND}(h_1, \text{NOT}(h_2)), \text{AND}(\text{NOT}(h_1), h_2)) \\ &= h_1 \bar{h}_2 + \bar{h}_1 h_2 \end{aligned}$$

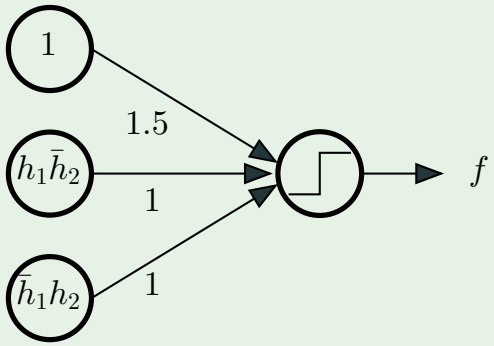
- This is a good news because OR and AND functions can be implemented by the perceptron.
- In other words, the (more complicated) target f is essentially a combination of perceptrons.

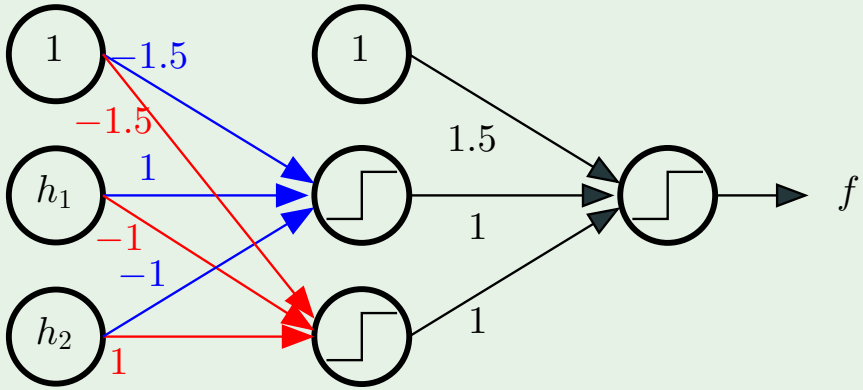
Combining perceptrons



The *computation graph* gives a convenient graph representation of the different operations to compute a function.

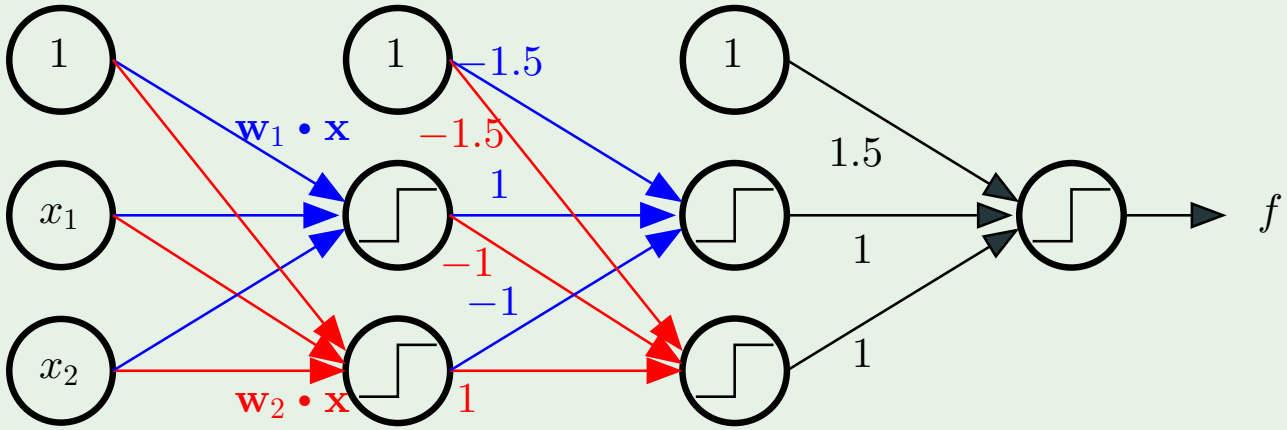
Creating layers





The multilayer perceptron (MLP)

The multilayer perceptron



3 layers "feedforward"

Exercise

Use the computation graph to write an explicit formula for $f(x_1, x_2) = f(\boldsymbol{x})$.

Exercise

Use the computation graph to write an explicit formula for $f(x_1, x_2) = f(\boldsymbol{x})$.

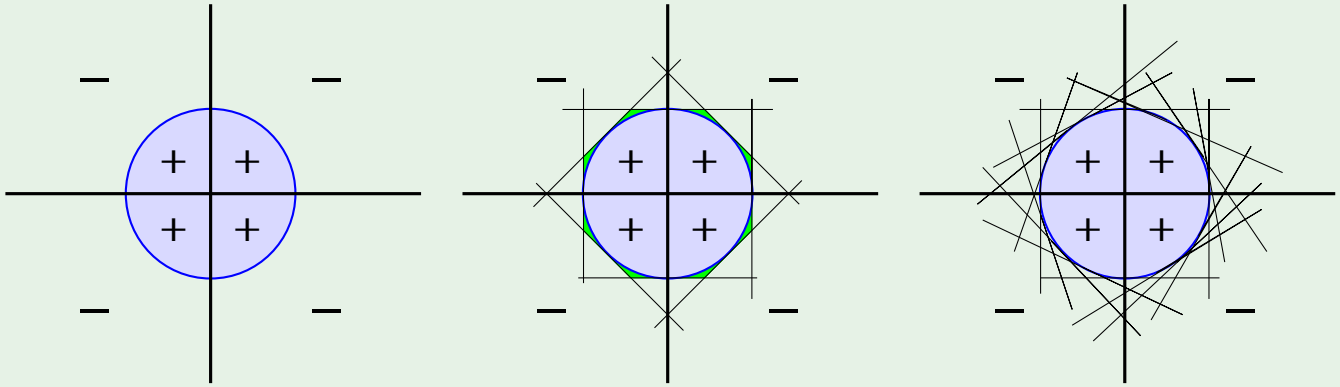
$$f(\boldsymbol{x}) = \text{sign}\left[\text{sign}\left(h_1(\boldsymbol{x}) - h_2(\boldsymbol{x}) - \frac{3}{2}\right) - \text{sign}(h_1(\boldsymbol{x}) - h_2(\boldsymbol{x}) + \frac{3}{2}) + \frac{3}{2}\right]$$

where

$$h_1(\boldsymbol{x}) = \text{sign}(\boldsymbol{w}_1^T \boldsymbol{x}) \text{ and } h_2(\boldsymbol{x}) = \text{sign}(\boldsymbol{w}_2^T \boldsymbol{x}).$$

A powerful model

A powerful model

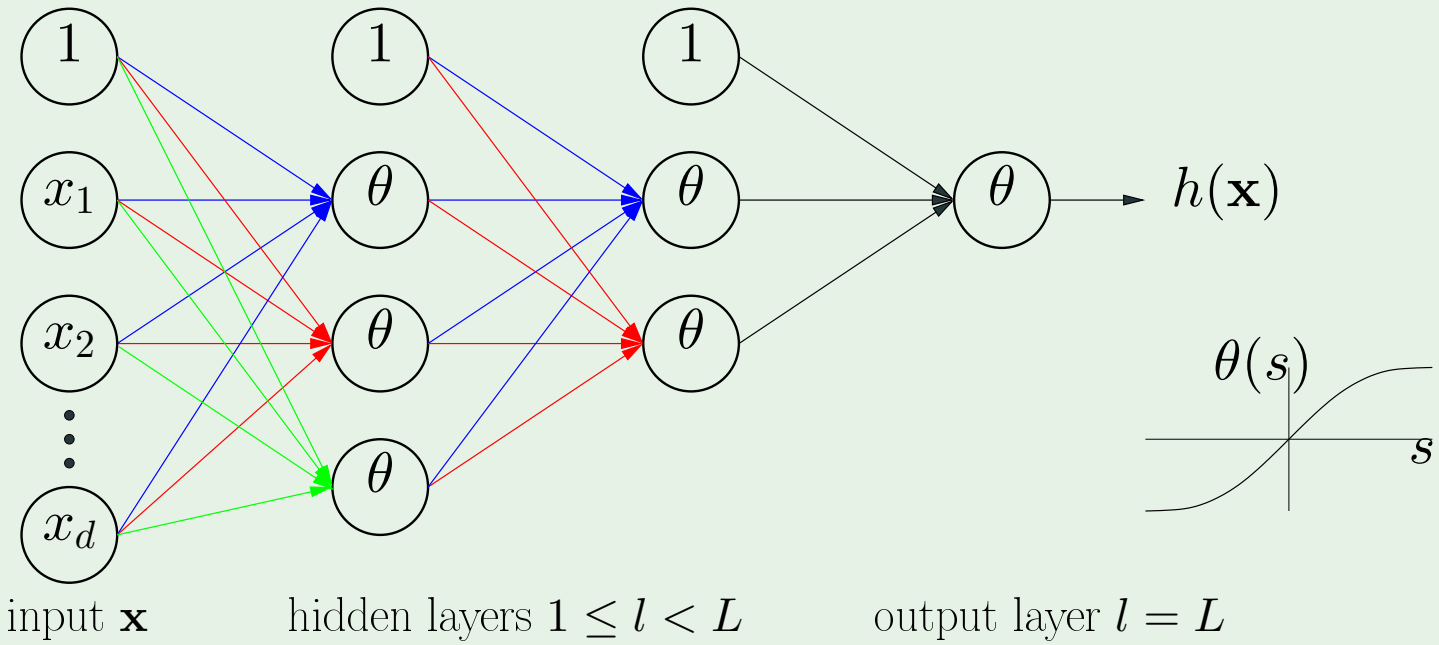


Target | 8 perceptrons | 16 perceptrons

2 red flags for **generalization** and **optimization**

Table of contents

The neural network

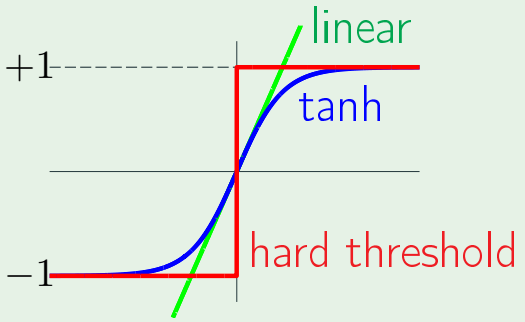


How the network operates

$$w_{ij}^{(l)} \quad \begin{cases} 1 \leq l \leq L & \text{layers} \\ 0 \leq i \leq d^{(l-1)} & \text{inputs} \\ 1 \leq j \leq d^{(l)} & \text{outputs} \end{cases}$$

$$x_j^{(l)} = \theta(s_j^{(l)}) = \theta\left(\sum_{i=0}^{d^{(l-1)}} w_{ij}^{(l)}\, x_i^{(l-1)}\right)$$

Apply $\mathbf{x}$ to $x_1^{(0)} \cdots x_{d^{(0)}}^{(0)} \rightarrow\rightarrow x_1^{(L)} = h(\mathbf{x})$



$$\theta(s) = \tanh(s) = \frac{e^s - e^{-s}}{e^s + e^{-s}}$$

The activation function

- Every node with an input has a *transformation* or *activation* function, θ. In the previous example: $\theta(s) = \text{sign}(s)$.
- $\text{sign}(s)$ function can be approximated with $\tanh(s)$, the hyperbolic tangent function, a soft threshold. A smooth function $\rightarrow$ easier to optimize, e.g. gradient descent.
- For the nodes in the hidden layers, arbitrary activation functions can be used. They are often chosen to allow fast computation and efficient optimization.
- For the output node, a different sigmoid will be used depending on the learning problem
  - Classification: $\hat{y} = \theta(s) = \tanh(s)$ or $\text{logistic}(s)$
  - Regression: $\hat{y} = \theta(s) = s$

Tanh, logistic and hard threshold

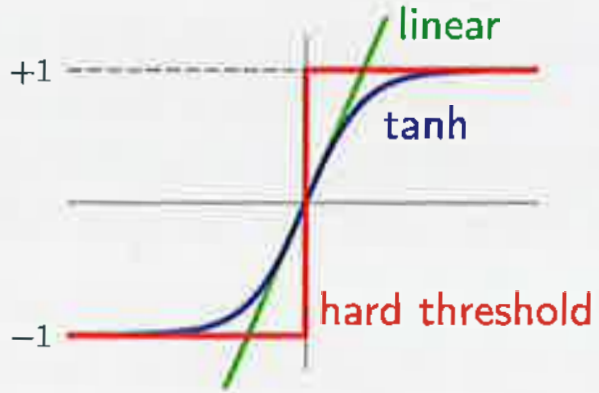
Exercise 3.5

Another popular soft threshold is the hyperbolic tangent

$$\tanh(s) = \frac{e^s - e^{-s}}{e^s + e^{-s}}.$$

(a) How is $\tanh$ related to the logistic function θ? *[Hint: shift and scale]*

(b) Show that $\tanh(s)$ converges to a hard threshold for large $|s|$, and converges to no threshold for small $|s|$. *[Hint: Formalize the figure below.]*



Tanh, logistic and hard threshold

(a)

$$\begin{aligned}
\tanh(s) &= \frac{e^s - e^{-s}}{e^s + e^{-s}} \\
&= \frac{e^{2s} - 1}{1 + e^{2s}} \\
&= \theta(2s) - \frac{1}{1 + e^{2s}} \\
&= \theta(2s) - (1 - \theta(2s)) \\
&= 2\theta(2s) - 1
\end{aligned}$$

Tanh, logistic and hard threshold

(b)

$$\tanh(s) = \frac{e^s - e^{-s}}{e^s + e^{-s}} = \frac{1 - e^{2s}}{1 + e^{-2s}} = \frac{e^{2s} - 1}{e^{2s} + 1}$$

Let $s \to \infty$, we have $e^{-2s} \to 0$, thus $\tanh(s) \to 1$. Similarly, when $s \to -\infty$, $\tanh(s) \to -1$.

It's also easy to see that $\tanh(s) < 1$ and $\tanh(s) \geq -1$. So -1 and 1 are hard thresholds for $\tanh(s)$ when $|s|$ is large.

Note that $\tanh^{'}(s) = 1 - \tanh^2(s)$. A first order Taylor approximation around $s = 0$ gives

$$\tanh(s) \approx \tanh(0) + s \times \tanh^{'}(0) \approx s.$$

Biological inspiration

biological function $\longrightarrow$ biological structure