

# Logistic Regression

Prof. Alessandro Lucantonio

Aarhus University

31/10/2023

# Binary classification

Recall:

- ▶ Classification  $\rightarrow$  discrete target.
- ▶ Binary classification:  $\{0, 1\}$  target. Example: spam/not spam e-mails.

*Idea:* consider a hypothesis (threshold) such that

$$0 \leq h_{\mathbf{w}} \leq 1$$

and

- ▶ if  $h_{\mathbf{w}}(\mathbf{x}) \geq 0.5$ , predict 1;
- ▶ if  $h_{\mathbf{w}}(\mathbf{x}) < 0.5$ , predict 0.

# Logistic Regression

In particular, take  $h_{\mathbf{w}}(\mathbf{x}) = \sigma(\mathbf{w}^T \mathbf{x})$ , where

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

is the **sigmoid function**.  $h_{\mathbf{w}}$  gives us the **probability** that the output is 1.

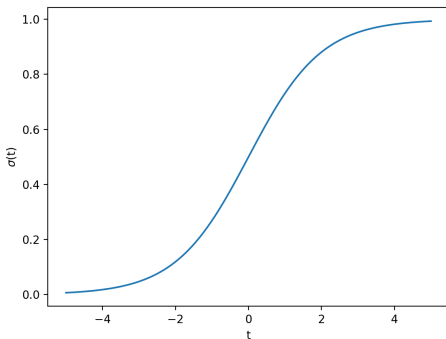


Figure: Sigmoid function

## Linear decision boundary

Model:  $h_{\mathbf{w}}(x_1, x_2) = \sigma(w_0 + w_1x_1 + w_2x_2)$

$h_{\mathbf{w}}(x_1, x_2) \geq 0.5$  when  $w_0 + w_1x_1 + w_2x_2 \geq 0 \implies y = 1$

$h_{\mathbf{w}}(x_1, x_2) < 0.5$  when  $w_0 + w_1x_1 + w_2x_2 < 0 \implies y = 0$

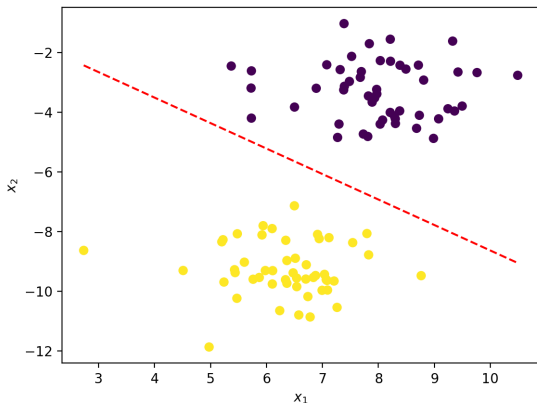


Figure: An example of linear decision boundary

## Non-linear decision boundary

Model:  $h_{\mathbf{w}}(x_1, x_2) = \sigma(w_0 + w_1x_1 + w_2x_2 + w_3x_1^2 + w_4x_2^2)$

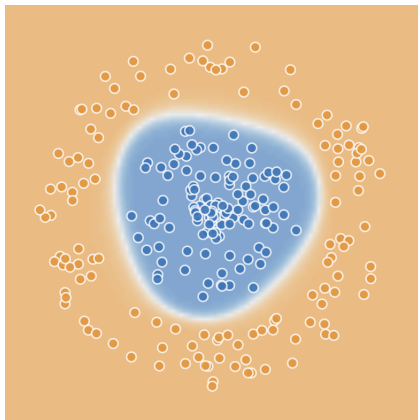


Figure: An example of non-linear decision boundary

## Cost function

First attempt: MSE

$$E(\mathbf{w}) = \frac{1}{N} \sum_{i=0}^N (\sigma(\mathbf{w}^T \tilde{\mathbf{x}}^{(i)}) - y^{(i)})^2$$

Problem:  $\sigma$  is *non-convex*, hence MSE is *non-convex* (possibly many local minima).

Main idea: consider the a loss term such that

- ▶  $-\log h_{\mathbf{w}}(\tilde{\mathbf{x}}^{(i)})$  if  $y = 1$ ,
- ▶  $-\log(1 - h_{\mathbf{w}}(\tilde{\mathbf{x}}^{(i)}))$  if  $y = 0$

Notice that:

- ▶ if  $y = 1$  and  $h_{\mathbf{w}}(\tilde{\mathbf{x}}^{(i)}) = 1$ ,  $-\log h_{\mathbf{w}}(\tilde{\mathbf{x}}^{(i)}) = 0$ ;
- ▶ if  $y = 1$  and  $h_{\mathbf{w}}(\tilde{\mathbf{x}}^{(i)}) = 0$ ,  $-\log h_{\mathbf{w}}(\tilde{\mathbf{x}}^{(i)}) \rightarrow \infty$ ;
- ▶ if  $y = 0$  and  $h_{\mathbf{w}}(\tilde{\mathbf{x}}^{(i)}) = 1$ ,  $-\log(1 - h_{\mathbf{w}}(\tilde{\mathbf{x}}^{(i)})) \rightarrow \infty$ ;
- ▶ if  $y = 0$  and  $h_{\mathbf{w}}(\tilde{\mathbf{x}}^{(i)}) = 0$ ,  $-\log(1 - h_{\mathbf{w}}(\tilde{\mathbf{x}}^{(i)})) = 0$ ;

# Cross-entropy

Combine the log loss terms: **Binary cross-entropy**

$$E(\mathbf{w}) := -\frac{1}{N} \sum_{i=1}^N y^{(i)} \log(h_{\mathbf{w}}(\tilde{\mathbf{x}}^{(i)})) + (1 - y^{(i)}) \log(1 - h_{\mathbf{w}}(\tilde{\mathbf{x}}^{(i)}))$$

This cost function is *convex* (sum of convex terms) with respect to the weights.

Vectorized version:

$$E(\mathbf{w}) = -\frac{1}{N} \left( \mathbf{y}^T \log(h_{\mathbf{w}}(\mathbf{X})) + (1 - \mathbf{y}^T) \log(1 - h_{\mathbf{w}}(\mathbf{X})) \right)$$

with  $h_{\mathbf{w}}(\mathbf{X}) = \sigma(\mathbf{X}\mathbf{w})$ .

# Convexity

Any local minimum of a convex function is also a global minimum. Instead, a non-convex function has potentially many local minima and *saddle points* (vanishing gradient but neither minimum nor maximum).

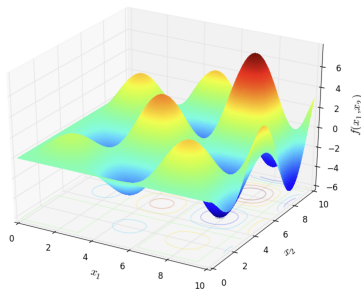


Figure: An example of a non-convex function

Remember the main challenge/goal of ML: **generalization**.

- A global minimum of the cost function corresponds to the best fit of the training set: this may lead to *overfitting*.



# Derivative of the sigmoid

Goal: computing the gradient of the cross-entropy with respect to the weights.

Recall:  $\sigma(t) := \frac{1}{1+e^{-t}}$ .

$$\begin{aligned}\frac{d}{dt}\sigma(t) &= \frac{e^{-t}}{(1+e^{-t})^2} \\ &= \left(\frac{1}{1+e^{-t}}\right) \left(\frac{e^{-t}}{1+e^{-t}}\right) \\ &= \sigma(t) \left(1 - \frac{1}{1+e^{-t}}\right) \\ &= \sigma(t)(1 - \sigma(t)).\end{aligned}$$

## Gradient of the cross-entropy /1

(assuming sum on repeated indices)

$$N \frac{\partial}{\partial \mathbf{w}_j} E(\mathbf{w}) = - \left[ \frac{y^{(i)} \frac{\partial}{\partial \mathbf{w}_j} h_{\mathbf{w}}(\tilde{\mathbf{x}}^{(i)})}{h_{\mathbf{w}}(\tilde{\mathbf{x}}^{(i)})} - \frac{(1 - y^{(i)}) \frac{\partial}{\partial \mathbf{w}_j} (1 - h_{\mathbf{w}}(\tilde{\mathbf{x}}^{(i)}))}{1 - h_{\mathbf{w}}(\tilde{\mathbf{x}}^{(i)})} \right]$$

with

$$\begin{aligned} \frac{\partial}{\partial \mathbf{w}_j} h_{\mathbf{w}}(\tilde{\mathbf{x}}^{(i)}) &= \sigma'(\mathbf{w}^T \tilde{\mathbf{x}}^{(i)}) \tilde{\mathbf{x}}_j^{(i)} = \sigma(\mathbf{w}^T \tilde{\mathbf{x}}^{(i)}) (1 - \sigma(\mathbf{w}^T \tilde{\mathbf{x}}^{(i)})) \tilde{\mathbf{x}}_j^{(i)} \\ &= h_{\mathbf{w}}(\tilde{\mathbf{x}}^{(i)}) (1 - h_{\mathbf{w}}(\tilde{\mathbf{x}}^{(i)})) \tilde{\mathbf{x}}_j^{(i)} \end{aligned}$$

→

## Gradient of the cross-entropy /2

$$\begin{aligned} N \frac{\partial}{\partial \mathbf{w}_j} E(\mathbf{w}) &= -[y^{(i)}(1 - h_{\mathbf{w}}(\tilde{\mathbf{x}}^{(i)}))\tilde{\mathbf{x}}_j^{(i)} - (1 - y^{(i)})h_{\mathbf{w}}(\tilde{\mathbf{x}}^{(i)})\tilde{\mathbf{x}}_j^{(i)}] \\ &= -[y^{(i)} - h_{\mathbf{w}}(\tilde{\mathbf{x}}^{(i)})]\tilde{\mathbf{x}}_j^{(i)} \\ &= [h_{\mathbf{w}}(\tilde{\mathbf{x}}^{(i)}) - y^{(i)}]\tilde{\mathbf{x}}_j^{(i)}. \end{aligned}$$

Final result:

$$\frac{\partial}{\partial \mathbf{w}_j} E(\mathbf{w}) = \frac{1}{N} \sum_{i=0}^N [h_{\mathbf{w}}(\tilde{\mathbf{x}}^{(i)}) - y^{(i)}]\tilde{\mathbf{x}}_j^{(i)}.$$

Vectorized version:

$$\nabla E(\mathbf{w}) = \frac{1}{N} \mathbf{X}^T (\sigma(\mathbf{X}\mathbf{w}) - \mathbf{y}).$$

# Multi-class classification

Targets:  $y \in \{0, \dots, k\}$

Idea: solve  $k + 1$  binary classification problems. Given a data sample  $\mathbf{x}^{(i)}$

- ▶ For each  $0 \leq j \leq k$ , compute the probability  $h_{\mathbf{w}}^{(j)}(\tilde{\mathbf{x}}^{(i)})$  that  $\tilde{\mathbf{x}}^{(i)}$  belongs to the class  $j$ .
- ▶ The prediction will be the class that corresponds to the maximum probability, *i.e.*

$$\arg \max_j h_{\mathbf{w}}^{(j)}(\tilde{\mathbf{x}}^{(i)})$$