Logistic Regression

Prof. Alessandro Lucantonio

Aarhus University - Department of Mechanical and Production Engineering

?/?/2023

Binary classification

Classification : discrete target (output vector).
Binary classification: $\{0, 1\}$ target

Example of binary classification task: Spam/not spam emails.

Idea: consider hypothesis h_w such that

$$0 \leq h_{\boldsymbol{w}} \leq 1.$$

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

Logistic Regression

Hypothesis: $h_{\boldsymbol{w}}(\boldsymbol{x}) = \sigma(\boldsymbol{w}^T\boldsymbol{x})$, where

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

is the **sigmoid function**.

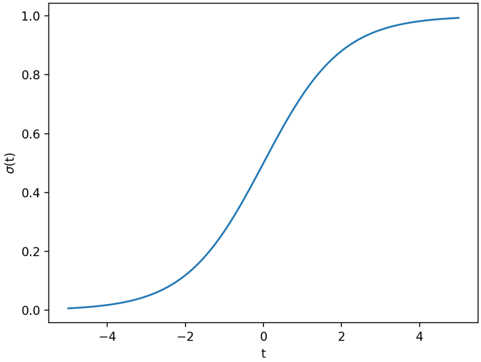

Figure: Sigmoid function

Linear decision boundary

Model: $h_{\boldsymbol{w}}(x_1, x_2) = \sigma(w_0 + w_1 x_1 + w_2 x_2)$

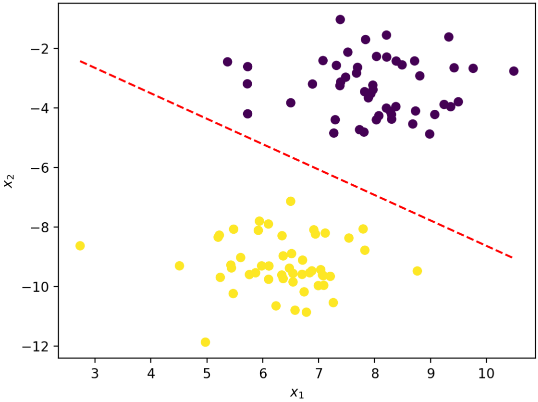

Figure: An example of linear decision boundary

Non-linear decision boundary

Model: $h_{\boldsymbol{w}}(x_1, x_2) = \sigma(w_0 + w_1 x_1 + w_2 x_2 + w_3 x_1^2 + w_4 x_2^2)$

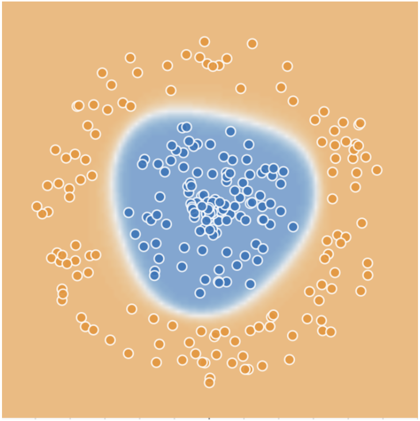

Figure: An example of non-linear decision boundary

Cost function

First try: MSE

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

Huge problem: σ *non-convex*, hence MSE is *non-convex* (many local minima).

Main idea: if $y = 1$ the prediction $h_{\boldsymbol{w}}(x)$ is good when $h_{\boldsymbol{w}}(x) \approx 1$. Good prediction means low error and $\log(1) = 0$.

Second try: **Binary cross-entropy**

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

Parenthesis - What is a convex function?

A function is *convex* when for all pairs of points on the graph, the line segment that connects these two points passes above the curve.

A function is *concave* when its opposite is convex.

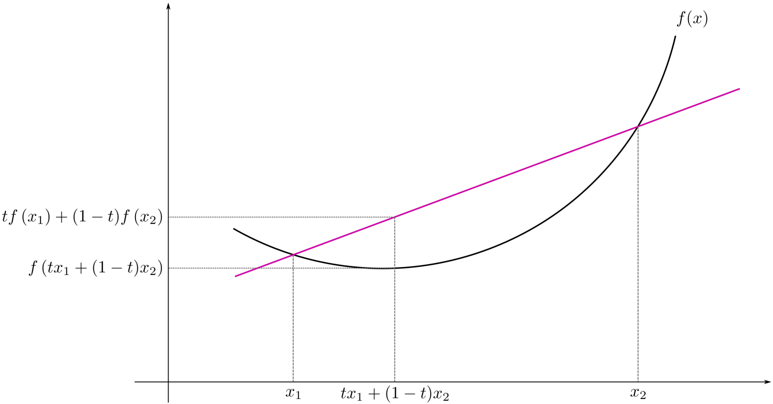

Figure: Geometric intuition of convexity

Parenthesis - The power of convexity

Any local minimum of a convex function is also a global minimum. Instead, a non-convex function has potentially many local minima which are not global minima and many saddle points (points with null gradient but nor minimum nor maximum).

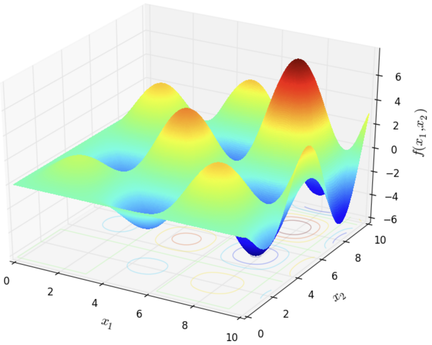

Figure: An example of non-convex function

Tips and Tricks - Is a local minimum always a bad news?

Overfitting is around the corner. Finding a global minimum means that we have the best fit possible on the training set: this is a potentially red flag on overfitting.

Of course is better to have a convex cost function, but this is not always the case. In that cases, finding a local minimum is probably even better than a global minimum.
Remember our goal: **generalization**.

Worst case scenario: find a saddle point.

Derivative of the sigmoid

Goal: Compute the gradient of the Cross-entropy.

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

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

Gradient of the Cross-entropy

Einstein notation: $\sum_i a_i \rightsquigarrow a_i$

$$\frac{\partial}{\partial w_j} h_{\boldsymbol{w}}(\boldsymbol{x}) = \sigma'(\boldsymbol{w}^T\boldsymbol{x})x_j = \sigma(\boldsymbol{w}^T\boldsymbol{x})(1-\sigma(\boldsymbol{w}^T\boldsymbol{x}))x_j$$
$$= h_{\boldsymbol{w}}(\boldsymbol{x})(1-h_{\boldsymbol{w}}(\boldsymbol{x}))x_j$$

$$N\frac{\partial}{\partial w_j}E(\boldsymbol{w}) = -\left[y^i\frac{\partial}{\partial w_j}\log(h_{\boldsymbol{w}}(\boldsymbol{x}^i)) + (1-y^i)\frac{\partial}{\partial w_j}\log(1-h_{\boldsymbol{w}}(\boldsymbol{x}^i))\right]$$
$$= -\left[\frac{y^i\frac{\partial}{\partial w_j}h_{\boldsymbol{w}}(\boldsymbol{x}^i)}{h_{\boldsymbol{w}}(\boldsymbol{x}^i)} - \frac{(1-y^i)\frac{\partial}{\partial w_j}(1-h_{\boldsymbol{w}}(\boldsymbol{x}^i))}{1-h_{\boldsymbol{w}}(\boldsymbol{x}^i)}\right]$$

$\rightarrow$

Gradient of the Cross-entropy

$$
\begin{aligned}
&= -[y^i(1 - h_{\boldsymbol{w}}(\boldsymbol{x^i}))x_j^i - (1 - y^i)h_{\boldsymbol{w}}(\boldsymbol{x^i})x_j^i] \\
&= -[y^i - h_{\boldsymbol{w}}(\boldsymbol{x^i})]x_j^i \\
&= [h_{\boldsymbol{w}}(\boldsymbol{x^i}) - y^i]x_j^i.
\end{aligned}
$$

Final result:

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

Vectorized version:

$$
\nabla_{\boldsymbol{w}}E(\boldsymbol{w}) = \frac{1}{N}\mathrm{X}^T(\sigma(\mathrm{X}\boldsymbol{w}) - \boldsymbol{y}).
$$

Multiclass classification: One-vs-all

Multi-classification: $y \in \{0, \ldots, k\}$.

Idea: solve $k+1$ binary classification problem. Given a data sample $\boldsymbol{x}$

- Establish for any $0 \leq j \leq k$ what is the probability $h_{\boldsymbol{w}}^{(j)}(\boldsymbol{x})$ (=output of the sigmoid) that $\boldsymbol{x}$ belongs to the class j.
- The final prediction will be the class which provides the maximum probability, i.e. $\arg\max_j h_{\boldsymbol{w}}^{(j)}(\boldsymbol{x})$