Linear Models for Classification

Master's Degree in Bioinformatics and Computational Biology - Machine Learning

Carlos María Alaíz Gudín

Escuela Politécnica Superior
Universidad Autónoma de Madrid

Academic Year 2024–25

Universidad Autónoma
de Madrid

Contents

Exercise ▷ **Questionnaire**

Please, fill in the questionnaire regarding your prior knowledge about this topic.

Introduction

Supervised Learning: Classification (I)

Definition (Classification Problem)

A **classification problem** is a supervised learning problem where the outputs are discrete.

Examples (Classification Problems)

- Predicting if a patient has a certain disease or not depending on medical data.
- Distinguishing the species of captured fish using the data provided by several sensors.
- Discerning the type of object that appears in a picture.

Supervised Learning: Classification (II)

Elements of a Supervised Learning Problem

Data	Set of input–output pairs, $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$.
Features	Vector of attributes (independent/input variables, covariates...), $\mathbf{x}_i \in \mathcal{X}$.
Label	Target (dependent variable, outcome...), $y_i \in \mathcal{Y}$.
Model	Mapping from the input to the output space, $f_{\boldsymbol{\theta}} : \mathcal{X} \to \mathcal{Y}$, with $\boldsymbol{\theta}$ the model parameters.
Learning Algorithm	Procedure to obtain a model based on the data, $\mathcal{A} : \mathcal{D} \to f_{\boldsymbol{\theta}}(\cdot)$.

- In a classification setting $\mathcal{Y} = \{\mathcal{C}_1, \mathcal{C}_2, \dots, \mathcal{C}_M\}$.
- In many situations, specially after preprocessing the data, $\mathcal{X} = \mathbb{R}^d$.
- The resultant model assigns to each input a certain class, $f_{\boldsymbol{\theta}} : \mathcal{X} \to \{\mathcal{C}_1, \mathcal{C}_2, \dots, \mathcal{C}_M\}$.

Binary Classification and Linear Models

- The most important classification scenario is when $M = 2$ (**binary classification**).
  - If $M > 2$, there are encoding techniques to transform the problem into several binary subproblems.
- The classes are usually denoted as $\mathcal{C}_0$ and $\mathcal{C}_1$, and they are represented with a $0/1$ (or $-1/1$) encoding.
  - The labels are transformed to:

$$t_i = \begin{cases} 0 & \text{if } y_i = \mathcal{C}_0, \\ 1 & \text{if } y_i = \mathcal{C}_1. \end{cases}$$

- "Simplest" approaches to classification:
  - Ignore the input: **constant model** (usually, **majority class**).
  - Define the output as a linear combination of the inputs plus a **transformation**: **linear model**.
    - Simple. Robust (small variance). Interpretable. Easy to train. Easy to predict.
    - Limited flexibility. Under-fitting (large bias).

Binary Linear Classification

Binary Linear Classifier

- For simplicity, $\mathcal{X} = \mathbb{R}^d$.
- The data becomes $\mathcal{D} = \{(\mathbf{x}_i, t_i)\}_{i=1}^N$, with $\mathbf{x}_i = (x_{i,1}, x_{i,2}, \dots, x_{i,d}) \in \mathbb{R}^d$ and $t_i \in \{0, 1\}$.

- The corresponding linear model is a hyperplane, with parameters $\boldsymbol{\theta} = \{b, \mathbf{w}\}$.
  - $b \in \mathbb{R}$ is the intercept or bias term.
  - $\mathbf{w} = (w_1, w_2, \dots, w_d) \in \mathbb{R}^d$ is the normal vector of the hyperplane.
  - The model is defined as:

$$f_{\boldsymbol{\theta}}(\mathbf{x}) = \begin{cases} 0 & \text{if } b + \mathbf{w}^\intercal \mathbf{x} < 0, \\ 1 & \text{if } b + \mathbf{w}^\intercal \mathbf{x} \geq 0. \end{cases}$$

  - The hyperplane divides the space into two halves, one for class $\mathcal{C}_0$ and the other for class $\mathcal{C}_1$.

- The **learning algorithm** will determine b and $\mathbf{w}$ using $\mathcal{D}$.

Binary Linear Classifier - Exercise

Exercise

▷ **Questionnaire**

Given a 2-dimensional binary linear classification model with parameters $\boldsymbol{\theta} = \{b, \mathbf{w}\}$, with $b = 1$ and $\mathbf{w} = (1, 2)^\intercal$.

1. Compute the output of the model for $\mathbf{x}_1 = (1, 1)^\intercal$.
2. Compute the output of the model for $\mathbf{x}_2 = (1, -2)^\intercal$.
3. Compute the output of the model for $\mathbf{x}_3 = (0, 0)^\intercal$.

Notebook

Binary Linear Classification: First Example

Quality of the Model

- A procedure is needed to determine the bias b and the vector $\mathbf{w}$, optimizing the **quality** of the model.
- The quality of the model has to be defined. Usually from two points of view:

  Fitness A fitness term $\mathcal{F}_{\mathcal{D}}(\boldsymbol{\theta})$ measures how well the model fits the training data.

  Complexity A regularization term $\mathcal{R}(\boldsymbol{\theta})$ penalizes the complexity of the model.

Fitness Term for a Classification Linear Model

Correct Prediction For the i-th pattern,

$$c_i = \begin{cases} 0 & \text{if } t_i \neq f_{\boldsymbol{\theta}}(\mathbf{x}_i) \\ 1 & \text{if } t_i = f_{\boldsymbol{\theta}}(\mathbf{x}_i) \end{cases} = \begin{cases} 0 & \text{if } (t_i = 0, b + \mathbf{w}^\intercal \mathbf{x}_i \geq 0) \text{ or } (t_i = 1, b + \mathbf{w}^\intercal \mathbf{x}_i < 0), \\ 1 & \text{if } (t_i = 0, b + \mathbf{w}^\intercal \mathbf{x}_i < 0) \text{ or } (t_i = 1, b + \mathbf{w}^\intercal \mathbf{x}_i \geq 0). \end{cases}$$

Accuracy $\mathrm{Acc}(b, \mathbf{w}) = \mathbb{E}[C] \approx \frac{1}{N} \sum_{i=1}^{N} c_i$.

Quality of the Model - Exercise

Exercise

▷ **Questionnaire**

Given a 2-dimensional binary linear classification model with parameters $\boldsymbol{\theta} = \{b, \mathbf{w}\}$, with $b = 1$ and $\mathbf{w} = (1, 2)^\intercal$, and for the following data:

$x_{i,1}$	$x_{i,2}$	t_i
1	1	1
1	-2	0
0	0	0

1. Compute the accuracy.

Notebook

Binary Linear Classification: Quality of the Model

Training a Linear Classifier: Using the Regression Framework

- The most common choice for evaluating the model is the accuracy.
  - It is a sensible and intuitive measure.
  - It is **non-convex**.
  - It is **non-differentiable**.
  - It is **discontinuous**.

- Optimizing the accuracy is a problem that cannot (in general) be tackled directly.

- An alternative idea could be to train a Linear Regression model.
  - Labels $-1/1$.
  - The predicted label is determined by taking the sign of the output.

Notebook

Binary Linear Classification: Training a Regression Linear Model

Training a Linear Classifier: Logistic Regression (I)

- A different quality measure is needed.
  - It should be simpler to optimize than the accuracy.
  - It should not penalize points far from the decision boundary (but on the correct side).

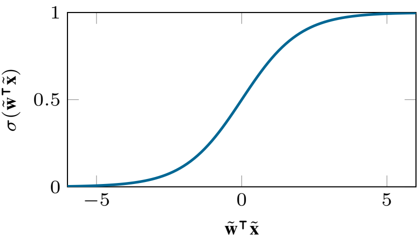
- A probabilistic approach can be helpful.
- In particular, the main framework is **Logistic Regression**.
  - The linear model is used to estimate the posterior probability of one class.
  - A sigmoid transformation is used.

Training a Linear Classifier: Logistic Regression (II)

- Denoting by $\tilde{\mathbf{x}} = [1, \mathbf{x}]$ and by $\tilde{\mathbf{w}} = [b, \mathbf{w}]$, the posterior probabilities are defined as:

$$\mathrm{p}(\mathcal{C}_1 | \tilde{\mathbf{x}}; \tilde{\mathbf{w}}) = \sigma(\tilde{\mathbf{w}}^\intercal \tilde{\mathbf{x}}) = \frac{1}{1 + e^{-\tilde{\mathbf{w}}^\intercal \tilde{\mathbf{x}}}},$$

$$\mathrm{p}(\mathcal{C}_0 | \tilde{\mathbf{x}}; \tilde{\mathbf{w}}) = 1 - \mathrm{p}(\mathcal{C}_1 | \tilde{\mathbf{x}}; \tilde{\mathbf{w}}) = 1 - \frac{1}{1 + e^{-\tilde{\mathbf{w}}^\intercal \tilde{\mathbf{x}}}} = \frac{e^{-\tilde{\mathbf{w}}^\intercal \tilde{\mathbf{x}}}}{1 + e^{-\tilde{\mathbf{w}}^\intercal \tilde{\mathbf{x}}}} = \frac{1}{1 + e^{\tilde{\mathbf{w}}^\intercal \tilde{\mathbf{x}}}} = \sigma(-\tilde{\mathbf{w}}^\intercal \tilde{\mathbf{x}}).$$



- $\tilde{\mathbf{w}}^\intercal \tilde{\mathbf{x}} < 0 \implies \mathrm{p}(\mathcal{C}_1 | \tilde{\mathbf{x}}; \tilde{\mathbf{w}}) < 0.5$: Class $\mathcal{C}_0$ is predicted.
- $\tilde{\mathbf{w}}^\intercal \tilde{\mathbf{x}} \geq 0 \implies \mathrm{p}(\mathcal{C}_1 | \tilde{\mathbf{x}}; \tilde{\mathbf{w}}) \geq 0.5$: Class $\mathcal{C}_1$ is predicted.

Training a Linear Classifier: Logistic Regression - Exercise

Exercise

▷ **Questionnaire**

Given a 2-dimensional binary linear classification model with parameters $\boldsymbol{\theta} = \{b, \mathbf{w}\}$, with $b = 1$ and $\mathbf{w} = (1, 2)^\intercal$.

1. Compute the probability of $\mathbf{x}_1$ belonging to class $\mathcal{C}_1$ for $\mathbf{x}_1 = (1, 1)^\intercal$.
2. Compute the probability of $\mathbf{x}_2$ belonging to class $\mathcal{C}_1$ for $\mathbf{x}_2 = (1, -2)^\intercal$.
3. Compute the probability of $\mathbf{x}_3$ belonging to class $\mathcal{C}_0$ for $\mathbf{x}_3 = (0, 0)^\intercal$.

Training a Linear Classifier: Maximum Likelihood (I)

- The probabilistic interpretation can help to define a quality measure.

Exercise

▷ **Questionnaire**

- In the following dataset, both $\boxed{\clubsuit}$ and $\boxed{\heartsuit}$ have been randomly generated with probabilities π_1 and $1-\pi_1$, respectively:

$$\boxed{\clubsuit}\ \boxed{\clubsuit}\ \boxed{\clubsuit}\ \boxed{\heartsuit}\ \boxed{\clubsuit}\ \boxed{\clubsuit}\ \boxed{\clubsuit}\ \boxed{\clubsuit}\ \boxed{\clubsuit}\ \boxed{\clubsuit}\ \boxed{\clubsuit}\ \boxed{\clubsuit}\ \boxed{\clubsuit}\ \boxed{\heartsuit}\ \boxed{\clubsuit}\ \boxed{\clubsuit}\ \boxed{\clubsuit}\ \boxed{\clubsuit}\ \boxed{\clubsuit}\ \boxed{\clubsuit}$$

Between the following options, what is the most likely value of π_1? Why?

- 0 %.
- 5 %.
- 50 %.
- 95 %.
- 100 %.

Training a Linear Classifier: Maximum Likelihood (II)

- The **likelihood** of the data is a common choice to quantify the quality of a probabilistic model:

$$\mathcal{L}(\mathcal{D}; \tilde{\mathbf{w}}) = \prod_{i=1}^{N} \mathrm{p}(t_i | \tilde{\mathbf{x}}_i; \tilde{\mathbf{w}}) = \prod_{i=1}^{N} \underbrace{\mathrm{p}(\mathcal{C}_0 | \tilde{\mathbf{x}}_i; \tilde{\mathbf{w}})^{1-t_i} \, \mathrm{p}(\mathcal{C}_1 | \tilde{\mathbf{x}}_i; \tilde{\mathbf{w}})^{t_i}}_{\begin{cases} \mathrm{p}(\mathcal{C}_0 | \tilde{\mathbf{x}}_i; \tilde{\mathbf{w}}) & \text{if } t_i = 0, \\ \mathrm{p}(\mathcal{C}_1 | \tilde{\mathbf{x}}_i; \tilde{\mathbf{w}}) & \text{if } t_i = 1. \end{cases}}.$$

- The **Cross Entropy (CE)** error is defined as the minus log-likelihood:

$$\begin{aligned} \mathrm{CE}(\tilde{\mathbf{w}}) &= -\log \mathcal{L}(\mathcal{D}; \tilde{\mathbf{w}}) \\ &= \sum_{i=1}^{N} \left(-(1 - t_i) \log(\mathrm{p}(\mathcal{C}_0 | \tilde{\mathbf{x}}_i; \tilde{\mathbf{w}})) - t_i \log(\mathrm{p}(\mathcal{C}_1 | \tilde{\mathbf{x}}_i; \tilde{\mathbf{w}})) \right) \\ &= \sum_{i=1}^{N} \left(-(1 - t_i) \log(1 - \sigma(\tilde{\mathbf{w}}^\intercal \tilde{\mathbf{x}}_i)) - t_i \log(\sigma(\tilde{\mathbf{w}}^\intercal \tilde{\mathbf{x}}_i)) \right). \end{aligned}$$

Training a Linear Classifier: Maximum Likelihood - Exercise

Exercise

▷ **Questionnaire**

Given a 2-dimensional binary linear classification model with parameters $\boldsymbol{\theta} = \{b, \mathbf{w}\}$, with $b = 1$ and $\mathbf{w} = (1, 2)^\intercal$, and for the following data:

$x_{i,1}$	$x_{i,2}$	t_i
1	1	1
1	-2	0
0	0	0

1. Compute the likelihood of this model.

Training a Linear Classifier: Maximum Likelihood (III)

- The minimizer of $\mathrm{CE}(\tilde{\mathbf{w}})$ is the maximizer of $\mathcal{L}(\mathcal{D}; \tilde{\mathbf{w}})$.
- The learning algorithm for training a Linear Logistic Regression model consists in solving the problem:

$$\min_{\tilde{\mathbf{w}} \in \mathbb{R}^{d+1}} \{\mathrm{CE}(\tilde{\mathbf{w}})\} = \min_{\tilde{\mathbf{w}} \in \mathbb{R}^{d+1}} \left\{ \sum_{i=1}^{N} \left(-(1 - t_i) \log(1 - \sigma(\tilde{\mathbf{w}}^\intercal \tilde{\mathbf{x}}_i)) - t_i \log(\sigma(\tilde{\mathbf{w}}^\intercal \tilde{\mathbf{x}}_i)) \right) \right\}.$$

- How is this problem solved?
  - It is **convex**: there are no local minima.
  - It is **differentiable**: the optima are characterized by the zeros of the gradient.

Training a Linear Classifier: Optimization (I)

$$\min_{\tilde{\mathbf{w}} \in \mathbb{R}^{d+1}} \{\mathrm{CE}(\tilde{\mathbf{w}})\} = \min_{\tilde{\mathbf{w}} \in \mathbb{R}^{d+1}} \left\{ \sum_{i=1}^{N} \left(-(1 - t_i) \log(1 - \sigma(\tilde{\mathbf{w}}^\intercal \tilde{\mathbf{x}}_i)) - t_i \log(\sigma(\tilde{\mathbf{w}}^\intercal \tilde{\mathbf{x}}_i)) \right) \right\}.$$

$$\begin{aligned}
\nabla_{\tilde{\mathbf{w}}} \mathrm{CE}(\tilde{\mathbf{w}}) &= \sum_{i=1}^{N} \left(-(1 - t_i) \nabla_{\tilde{\mathbf{w}}} \log(1 - \sigma(\tilde{\mathbf{w}}^\intercal \tilde{\mathbf{x}}_i)) - t_i \nabla_{\tilde{\mathbf{w}}} \log(\sigma(\tilde{\mathbf{w}}^\intercal \tilde{\mathbf{x}}_i)) \right) \\
&= \sum_{i=1}^{N} \left((1 - t_i) \sigma(\tilde{\mathbf{w}}^\intercal \tilde{\mathbf{x}}_i) \tilde{\mathbf{x}}_i - t_i (1 - \sigma(\tilde{\mathbf{w}}^\intercal \tilde{\mathbf{x}}_i)) \tilde{\mathbf{x}}_i \right) \\
&= \sum_{i=1}^{N} \sigma(\tilde{\mathbf{w}}^\intercal \tilde{\mathbf{x}}_i) \tilde{\mathbf{x}}_i - t_i \sigma(\tilde{\mathbf{w}}^\intercal \tilde{\mathbf{x}}_i) \tilde{\mathbf{x}}_i - t_i \tilde{\mathbf{x}}_i + t_i \sigma(\tilde{\mathbf{w}}^\intercal \tilde{\mathbf{x}}_i) \tilde{\mathbf{x}}_i \\
&= \sum_{i=1}^{N} (\sigma(\tilde{\mathbf{w}}^\intercal \tilde{\mathbf{x}}_i) - t_i) \tilde{\mathbf{x}}_i .
\end{aligned}$$

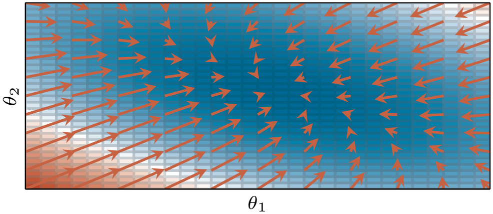
More detail in the appendix **Gradient of the Sigmoid Transformation**.

Gradient-Based Optimization: Iterative Methods

- Can the problem $\nabla_{\boldsymbol{\theta}} \mathcal{F}(\boldsymbol{\theta}^\star) = \mathbf{0}$ be directly solved?
  - Only for simple cases.
  - In several dimensions, it implies solving a system of d equations (one for each partial derivative).
  - This condition does not imply minimum (valley), it may be a maximum or a saddle point.

- But the gradient points in the direction of greatest increase:
  - $-\nabla_{\boldsymbol{\theta}} \mathcal{F}$ points in the direction of **steepest descent**.
  - This can be used to define where to go next.

$$\mathcal{F}(\boldsymbol{\theta} + \boldsymbol{\epsilon}) \approx \mathcal{F}(\boldsymbol{\theta}) + \nabla_{\boldsymbol{\theta}} \mathcal{F}(\boldsymbol{\theta})^\intercal \boldsymbol{\epsilon} \implies \mathcal{F}(\boldsymbol{\theta} - \eta \nabla_{\mathbf{x}} \mathcal{F}(\boldsymbol{\theta})) \approx \mathcal{F}(\boldsymbol{\theta}) - \eta \|\nabla_{\boldsymbol{\theta}} \mathcal{F}(\boldsymbol{\theta})\|_2^2 \leq \mathcal{F}(\boldsymbol{\theta}).$$



Gradient Descent

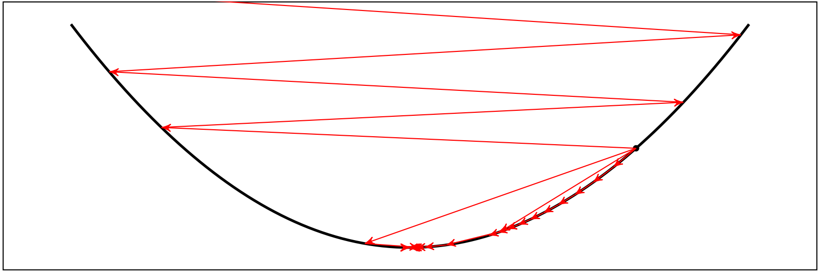
- It is a simple (yet useful) optimization algorithm that is often used in Machine Learning (ML) to find the local minimum.

Gradient Descent: Algorithm

Require: Objective function $\mathcal{F}$, starting point $\boldsymbol{\theta}^{(0)}$ ▷ Start at a random point.
Ensure: $\boldsymbol{\theta}^{(t-1)} \in \mathbb{R}^d$ an approximate local minimum of $\mathcal{F}(\boldsymbol{\theta})$ ▷ Arrive to a valley.
  for $t = 1, 2, \cdots$ **do** ▷ Several steps.
    $\mathbf{g} \leftarrow \nabla_{\boldsymbol{\theta}} \mathcal{F}\left(\boldsymbol{\theta}^{(t-1)}\right)$ ▷ Stop, look for the direction most downhill.
    if $\mathbf{g} \approx \mathbf{0}$ **then**
      return $\boldsymbol{\theta}^{(t-1)}$ ▷ If there is not direction downhill, stop.
    end if
    $\boldsymbol{\theta}^{(t)} \leftarrow \boldsymbol{\theta}^{(t-1)} - \eta^{(t)} \mathbf{g}$ ▷ Take a step in that direction.
  end for

Gradient Descent: Step-Size

- The step-size $\eta^{(t)}$ has to be set at each iteration t.
- This is a crucial issue:
  1. If the step-size is too small, the algorithm will require too many epochs (iterations) to converge and can become trapped in local minima more easily.
  2. If the step-size is large, the convergence will also be slow.
  3. If the step-size is too large, gradient descent will overshoot the minima an diverge.
- In some cases, optimal step-sizes can be computed.
- There are heuristics that guarantee convergence, but only to **local minima**, and usually slowly and zigzagging.



Training a Linear Classifier: Optimization (II)

- In summary, the Linear Logistic Regression Model is the solution of the following problem:

$$\min_{\tilde{\mathbf{w}} \in \mathbb{R}^{d+1}} \left\{ \sum_{i=1}^{N} \left(-(1 - t_i) \log(1 - \sigma(\tilde{\mathbf{w}}^\intercal \tilde{\mathbf{x}}_i)) - t_i \log(\sigma(\tilde{\mathbf{w}}^\intercal \tilde{\mathbf{x}}_i)) \right) \right\}.$$

- There is not closed-form solution to the resultant equation for the stationary points:

$$\nabla_{\tilde{\mathbf{w}}} \mathrm{CE}(\tilde{\mathbf{w}}) = \sum_{i=1}^{N} (\sigma(\tilde{\mathbf{w}}^\intercal \tilde{\mathbf{x}}_i) - t_i) \tilde{\mathbf{x}}_i = \mathbf{0}.$$

- An iterative algorithm, such as **gradient descent**, should be used.

Linear Logistic Regression Model

- The model can be trained iteratively by updating the weights as:

$$\tilde{\mathbf{w}}^{(s+1)} = \tilde{\mathbf{w}}^{(s)} - \eta^{(s)} \sum_{i=1}^{N} \left(\sigma\left(\left(\tilde{\mathbf{w}}^{(s)} \right)^\intercal \tilde{\mathbf{x}}_i \right) - t_i \right) \tilde{\mathbf{x}}_i.$$

Notebook

Binary Linear Classification: Optimization

Summary

Linear Models for Classification: Summary

- A **classification** problem is a supervised problem with discrete targets.

- A simple yet useful classification model is the **linear model**.
  - The decision boundary is a hyperplane dividing the space.

- In order to train the linear model, an **optimization problem** is usually solved.
- The **accuracy**, the more natural choice, is hard to optimize in practice.

- The **likelihood** is used instead, leading to the **Logistic Regression** model.
  - The resultant problem can be solved iteratively using gradient descent.

Linear Models for Classification

Carlos María Alaíz Gudín

Additional Material

Gradient of the Sigmoid Transformation

Expressions for the Gradient of the Sigmoid Transformation

- The linear model with sigmoid transformation satisfies the following equations:

$$
\begin{aligned}
\nabla_{\tilde{\mathbf{w}}} \sigma(\tilde{\mathbf{w}}^\intercal \tilde{\mathbf{x}}) &= \nabla_{\tilde{\mathbf{w}}} \frac{1}{1 + e^{-\tilde{\mathbf{w}}^\intercal \tilde{\mathbf{x}}}} = \frac{1}{\left(1 + e^{-\tilde{\mathbf{w}}^\intercal \tilde{\mathbf{x}}}\right)^2} e^{-\tilde{\mathbf{w}}^\intercal \tilde{\mathbf{x}}} \tilde{\mathbf{x}} = \frac{1}{1 + e^{-\tilde{\mathbf{w}}^\intercal \tilde{\mathbf{x}}}} \frac{e^{-\tilde{\mathbf{w}}^\intercal \tilde{\mathbf{x}}}}{1 + e^{-\tilde{\mathbf{w}}^\intercal \tilde{\mathbf{x}}}} \tilde{\mathbf{x}} \\
&= \sigma(\tilde{\mathbf{w}}^\intercal \tilde{\mathbf{x}})(1 - \sigma(\tilde{\mathbf{w}}^\intercal \tilde{\mathbf{x}}))\tilde{\mathbf{x}}; \\
\nabla_{\tilde{\mathbf{w}}} \log(\sigma(\tilde{\mathbf{w}}^\intercal \tilde{\mathbf{x}})) &= \frac{1}{\sigma(\tilde{\mathbf{w}}^\intercal \tilde{\mathbf{x}})} \nabla_{\tilde{\mathbf{w}}} \sigma(\tilde{\mathbf{w}}^\intercal \tilde{\mathbf{x}}) = (1 - \sigma(\tilde{\mathbf{w}}^\intercal \tilde{\mathbf{x}}))\tilde{\mathbf{x}}; \\
\nabla_{\tilde{\mathbf{w}}} \log(1 - \sigma(\tilde{\mathbf{w}}^\intercal \tilde{\mathbf{x}})) &= \nabla_{\tilde{\mathbf{w}}} \log(\sigma(-\tilde{\mathbf{w}}^\intercal \tilde{\mathbf{x}})) = -(1 - \sigma(-\tilde{\mathbf{w}}^\intercal \tilde{\mathbf{x}}))\tilde{\mathbf{x}} = -\sigma(\tilde{\mathbf{w}}^\intercal \tilde{\mathbf{x}})\tilde{\mathbf{x}}.
\end{aligned}
$$

- These properties are one of the reasons why this function is so commonly used.