

# Logistic regression

Daniel Hsu (COMS 4771)

## The logistic regression model

Logistic regression is a model for binary classification data with feature vectors in  $\mathbb{R}^d$  and labels in  $\{-1, +1\}$ . Data  $(\mathbf{X}_1, Y_1), \dots, (\mathbf{X}_n, Y_n)$  are treated as iid random variables taking values in  $\mathbb{R}^d \times \{-1, +1\}$ , and for each  $\mathbf{x} \in \mathbb{R}^d$ ,

$$Y_i \mid \mathbf{X}_i = \mathbf{x} \sim \text{Bern}(\sigma(\mathbf{x}^\top \mathbf{w}))$$

where  $\sigma(t) = 1/(1 + \exp(-t))$  is the sigmoid function. Here,  $\mathbf{w} \in \mathbb{R}^d$  is the parameter of the model, and it is not involved in the marginal distribution of  $\mathbf{X}_i$  (which we leave unspecified).

## Maximum likelihood

The log-likelihood of  $\mathbf{w}$  given  $(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n) \in \mathbb{R}^d \times \{-1, +1\}$  is

$$-\sum_{i=1}^n \ln(1 + \exp(-y_i \mathbf{x}_i^\top \mathbf{w})) + (\text{terms that do not involve } \mathbf{w}).$$

There is no closed-form expression for the maximizer of the log-likelihood. Nevertheless, we can approximately minimize the negative log-likelihood with gradient descent.

## Empirical risk minimization

Maximum likelihood is very different from finding the linear classifier of smallest empirical zero-one loss risk. Finding the empirical zero-one loss risk minimizer is computationally intractable in general.

## Finding a linear separator

There are special cases when finding the empirical zero-one loss risk minimizer is computationally tractable. One is when the training data is *linearly separable*: i.e., when there exists  $\mathbf{w}^* \in \mathbb{R}^d$  such that

$$y_i \mathbf{x}_i^\top \mathbf{w}^* > 0, \quad \text{for all } i = 1, \dots, n.$$

**Claim.** Define  $L(\mathbf{w}) := \sum_{i=1}^n \ln(1 + \exp(-y_i \mathbf{x}_i^\top \mathbf{w}))$ . Suppose  $(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n) \in \mathbb{R}^d \times \{-1, +1\}$  is linearly separable. Then any  $\hat{\mathbf{w}} \in \mathbb{R}^d$  with

$$L(\hat{\mathbf{w}}) < \inf_{\mathbf{w} \in \mathbb{R}^d} L(\mathbf{w}) + \ln(2)$$

is a linear separator.

*Proof.* We first observe that the infimum<sup>1</sup> (i.e., greatest lower bound) of  $L$  is zero. Let  $\mathbf{w}^* \in \mathbb{R}^d$  be a linear separator, so  $s_i := y_i \mathbf{x}_i^\top \mathbf{w}^* > 0$  for all  $i = 1, \dots, n$ . For any  $r > 0$ ,

$$L(r\mathbf{w}^*) = \sum_{i=1}^n \ln(1 + \exp(-rs_i)),$$

<sup>1</sup>[https://en.wikipedia.org/wiki/Infimum\\_and\\_supremum](https://en.wikipedia.org/wiki/Infimum_and_supremum)

and therefore

$$\lim_{r \rightarrow \infty} \sum_{i=1}^n \ln(1 + \exp(-rs_i)) = 0.$$

Every term  $\ln(1 + \exp(-y_i \mathbf{x}_i^\top \mathbf{w}))$  in  $L(\mathbf{w})$  is positive, so  $L(\mathbf{w}) > 0$ . Therefore, we conclude that

$$\inf_{\mathbf{w} \in \mathbb{R}^d} L(\mathbf{w}) = 0.$$

So now we just have to show that any  $\hat{\mathbf{w}} \in \mathbb{R}^d$  with

$$L(\hat{\mathbf{w}}) < \ln(2)$$

is a linear separator. So let  $\hat{\mathbf{w}}$  satisfy  $L(\hat{\mathbf{w}}) < \ln(2)$ , which implies

$$\ln(1 + \exp(-y_i \mathbf{x}_i^\top \hat{\mathbf{w}})) < \ln(2)$$

for every  $i = 1, \dots, n$ . Exponentiating both sides gives

$$1 + \exp(-y_i \mathbf{x}_i^\top \hat{\mathbf{w}}) < 2.$$

Now subtracting 1 from both sides and taking logarithms gives

$$-\ln \exp(-y_i \mathbf{x}_i^\top \hat{\mathbf{w}}) < 0.$$

This means that  $\hat{\mathbf{w}}$  correctly classifies  $(\mathbf{x}_i, y_i)$ . Since this holds for all  $i = 1, \dots, n$ , it follows that  $\hat{\mathbf{w}}$  is a linear separator.  $\square$

**Surrogate loss**

Even if  $(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n) \in \mathbb{R}^d \times \{-1, +1\}$  is not linearly separable, approximately maximizing the log-likelihood can yield a good linear classifier. This is because maximizing  $L$  is the same as minimizing the empirical logistic loss risk:

$$\hat{\mathcal{R}}(\mathbf{w}) := \frac{1}{n} \sum_{i=1}^n \ell_{\log}(y_i \mathbf{x}_i^\top \mathbf{w})$$

where

$$\ell_{\log}(z) := -\ln \sigma(z)$$

is the *logistic loss*. The logistic loss (up to scaling) turns out to be an upper-bound on the zero-one loss:

$$\ell_{zo}(z) \leq \frac{1}{\ln 2} \ell_{\log}(z),$$

where  $\ell_{zo}(z) = \mathbb{1}_{\{z \leq 0\}}$ . If the empirical logistic loss risk is small, then the empirical zero-one loss is also small.

## Gradient descent for logistic regression

The derivative of  $\ell_{\log}$  is given by

$$\begin{aligned} \frac{d\ell_{\log}(z)}{dz} &= -\frac{1}{\sigma(z)} \cdot \frac{d\sigma(z)}{dz} \\ &= -\frac{1}{\sigma(z)} \cdot \sigma(z) \cdot \sigma(-z) \\ &= -\sigma(-z). \end{aligned}$$

Therefore, by linearity and the chain rule, the negative gradient of  $\widehat{\mathcal{R}}$  with respect to  $\mathbf{w}$  is

$$\begin{aligned} -\nabla \widehat{\mathcal{R}}(\mathbf{w}) &= -\frac{1}{n} \sum_{i=1}^n \nabla \ell_{\log}(y_i \mathbf{x}_i^\top \mathbf{w}) \\ &= -\frac{1}{n} \sum_{i=1}^n \left. \frac{d\ell_{\log}(z)}{dz} \right|_{z=y_i \mathbf{x}_i^\top \mathbf{w}} \cdot \nabla (y_i \mathbf{x}_i^\top \mathbf{w}) \\ &= \frac{1}{n} \sum_{i=1}^n \sigma(-y_i \mathbf{x}_i^\top \mathbf{w}) \cdot y_i \mathbf{x}_i. \end{aligned}$$

Now suppose  $\mathbf{A} = [\mathbf{x}_1 | \dots | \mathbf{x}_n]^\top \in \mathbb{R}^{n \times d}$  and  $\mathbf{b} = [y_1 | \dots | y_n]^\top \in \mathbb{R}^n$ . (Notice that we have omitted the  $1/\sqrt{n}$  scaling that we had for least squares linear regression.) Then the negative gradient of  $\widehat{\mathcal{R}}$  can be written as

$$-\nabla \widehat{\mathcal{R}}(\mathbf{w}) = \frac{1}{n} \mathbf{A}^\top (\mathbf{b} \odot \sigma(-\mathbf{b} \odot (\mathbf{A}\mathbf{w}))),$$

where  $\mathbf{u} \odot \mathbf{v} \in \mathbb{R}^n$  is the coordinate-wise product of vectors  $\mathbf{u} \in \mathbb{R}^n$  and  $\mathbf{v} \in \mathbb{R}^n$ , and  $\sigma(\mathbf{v}) \in \mathbb{R}^n$  is the coordinate-wise application of the sigmoid function to  $\mathbf{v} \in \mathbb{R}^n$ .

Gradient descent for logistic regression begins with an initial weight vector  $\mathbf{w}^{(0)} \in \mathbb{R}^d$ , and then iteratively updates it by subtracting a positive multiple  $\eta > 0$  of the gradient at the current iterate:

**Assignment Project Exam Help**

**<https://powcoder.com>**

**Add WeChat powcoder**