Introduction to Machine Learning

Chapter 6: Logistic Regression

Bernd Bischl, Christoph Molnar, Daniel Schalk, Fabian Scheipl

Department of Statistics – LMU Munich

LOGISTIC REGRESSION

A *discriminant* approach for directly modeling the posterior probabilities $\pi(x)$ of the labels is **logistic regression**.
For now, let's focus on the binary case $y \in \{0, 1\}$.
A naive approach would be to model

$$\pi(x) = \mathbb{P}(y = 1|x) = \theta^T x.$$

Obviously this could result in predicted probabilities $\pi(x) \not\in [0, 1]$.

LOGISTIC REGRESSION

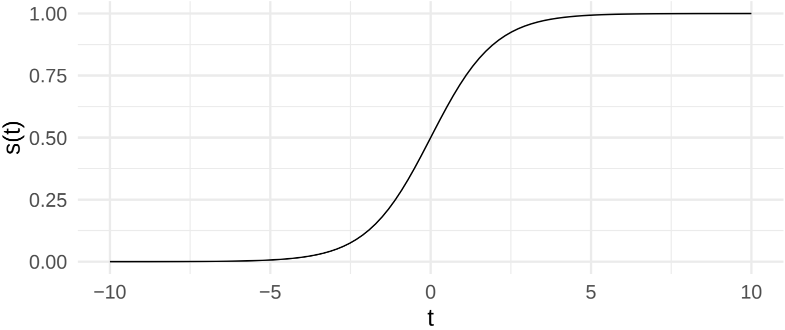
To avoid this, logistic regression "squashes" the estimated linear scores $\theta^T x$ to $[0, 1]$ through the **logistic function** s:

$$\pi(x) = \mathbb{P}(y = 1|x) = \frac{\exp\left(\theta^T x\right)}{1 + \exp\left(\theta^T x\right)} = \frac{1}{1 + \exp\left(-\theta^T x\right)} = s\left(\theta^T x\right)$$

Note that we will again usually suppress the intercept in notation, i.e., $\theta^T x \equiv \theta_0 + \theta^T x$.

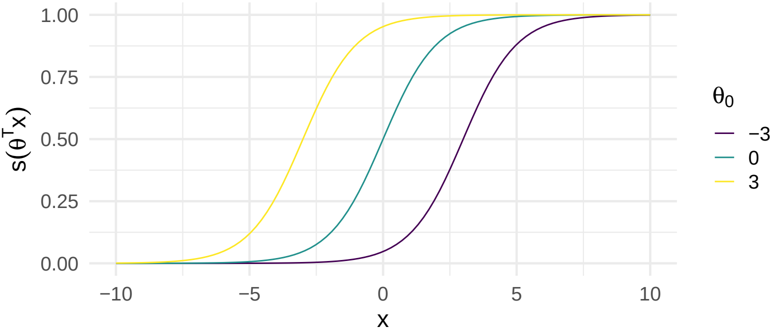
LOGISTIC FUNCTION

The logistic function $s(t) = \frac{exp(t)}{1+exp(t)}$ which we use to model the probability $\mathbb{P}(y = 1|x) = s(\theta^T x) = \frac{\exp(\theta^T x)}{1+\exp(\theta^T x)}$



LOGISTIC FUNCTION

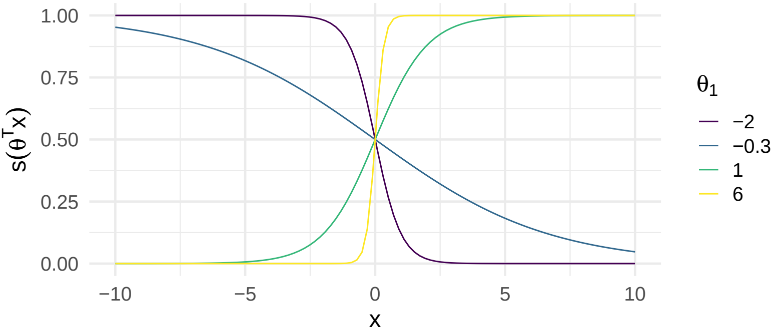
Changing the intercept shifts the logistic curve in x-axis direction. Let's assume $\theta_1 = 1$ for simplicity, so that $\mathbb{P}(y = 1|x) = \frac{\exp(\theta_0 + x)}{1 + \exp(\theta_0 + x)}$



LOGISTIC FUNCTION

Assuming a single feature and no intercept: $\mathbb{P}(y = 1|x) = \frac{\exp(\theta_1 x_1)}{1+\exp(\theta_1 x_1)}$:

Parameter θ_1 controls the slope and direction of the logistic curve.



CLASSIFICATION LOSS

In order to find the "optimal" model represented by θ, we need to define a *loss function*.

For a single observation, it makes sense to simply compare the probability implied by the model to the actually observed target variable:

$$\text{"accuracy"}^{(i)} := \begin{cases} \pi(x^{(i)}, \theta) & \text{if } y^{(i)} = 1 \\ 1 - \pi(x^{(i)}, \theta) & \text{if } y^{(i)} = 0 \end{cases}$$

$$= \pi(x^{(i)}, \theta)^{y^{(i)}} (1 - \pi(x^{(i)}, \theta))^{1 - y^{(i)}}$$

For the entire data set, we combine these predicted probabilites into a joint probability of observing the target vector given the model:

$$\text{"global accuracy"} := \prod_{i=1}^{n} \pi(x^{(i)}, \theta)^{y^{(i)}} (1 - \pi(x^{(i)}, \theta))^{1 - y^{(i)}}$$

CLASSIFICATION LOSS

We want a *loss* function, so we actually need the inverse of that:

$$\text{“global inacccuracy”} := \frac{1}{\prod_{i=1}^{n} \pi(x^{(i)}, \theta)^{y^{(i)}} (1 - \pi(x^{(i)}, \theta))^{1-y^{(i)}}}$$

Finally, we want the empirical risk to be a *sum* of loss function values, not a *product*

$$\text{recall: } \mathcal{R}_{\text{emp}} = \sum_{i=1}^{n} L\left(y^{(i)}, f\left(x^{(i)}\right)\right)$$

so we turn the product into a sum by taking its log – the same parameters minimize this, which is all we care about, and we end up with the **logistic** or **cross entropy loss function**:

$$\begin{aligned} L(y, f(x)) &= -y \log[\pi(x)] - (1 - y) \log[1 - \pi(x)] \\ &= y\theta^T x - \log[1 + \exp(\theta^T x)] \end{aligned}$$

CLASSIFICATION LOSS

Remember that $\log[\pi(x)] = -\log[1 + \exp(-\theta^T x)]$.

For $y = 0$ and $y = 1$ and $f(x) = \theta^T x$, this yields:

$$\begin{aligned} y = 0 \quad &: \quad \log[1 + \exp(f(x)] \\ y = 1 \quad &: \quad \log[1 + \exp(-f(x))] \end{aligned}$$

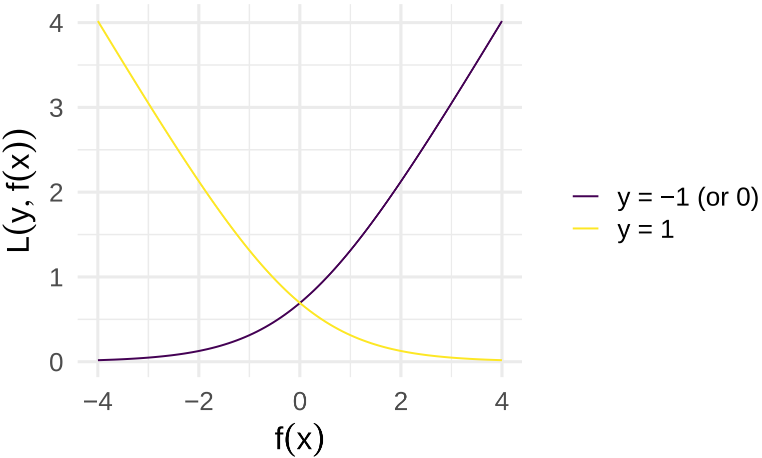
If we encode the labels with $\mathcal{Y} = \{-1, +1\}$ instead, we can simplify this as:

$$L(y, f(x)) = \log[1 + \exp(-yf(x)]$$

This is called **Bernoulli loss**.
Logistic regression minimizes this, and we can use these loss functions for any other discriminative classification model which directly models $f(x)$.

CLASSIFICATION LOSS

Bernoulli / Logistic Loss:



CLASSIFICATION LOSS

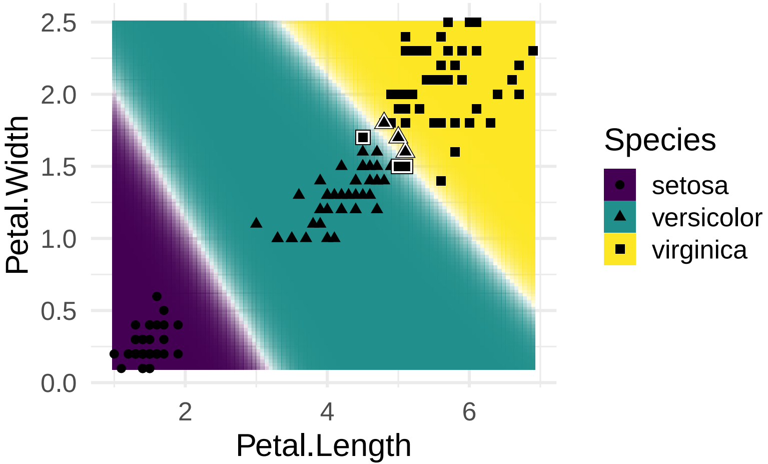
In order to minimize the loss (misclassification), we should predict $y = 1$ if

$$\pi(x, \theta) = \mathbb{P}(y = 1 | x, \theta) = \frac{\exp(\theta^T x)}{1 + \exp(\theta^T x)} \geq 0.5,$$

which is equivalent to

$$\theta^T x \geq 0 \implies y = 1.$$

ESTIMATING PROBABILITIES - MULTINOMIAL REGRESSION AND SOFTMAX



ESTIMATING PROBABILITIES - MULTINOMIAL REGRESSION AND SOFTMAX

For a categorical response variable $y \in \{1, \dots, g\}$ with $g > 2$ the model extends to

$$\pi_k(x) = \mathbb{P}(y = k|x) = \frac{\exp(\theta_k^T x)}{\sum_{j=1}^{g} \exp(\theta_j^T x)}.$$

The latter function is called the *softmax*, and defined on a numerical vector z:

$$s(z)_k = \frac{\exp(z_k)}{\sum_j \exp(z_j)}$$

This is a generalization of the logistic function (check for g=2).
It “squashes” a g-dimensional real-valued vector z to a vector of the same dimension, with every entry in the range [0, 1] and all entries adding up to 1.

ESTIMATING PROBABILITIES - MULTINOMIAL REGRESSION AND SOFTMAX

By comparing the posterior probabilities of two categories k and l we end up in a linear function (in x),

$$\log \frac{\pi_k(x)}{\pi_l(x)} = (\theta_k - \theta_l)^T x.$$

The class boundaries lie where these (linear) functions are zero, i.e., where the predicted class probabilities are equal.

Remark:

- θ_j are vectors here.
- Well-definedness: $\pi_k(x) \in [0, 1]$ and $\sum_k \pi_k(x) = 1$

ESTIMATING PROBABILITIES - MULTINOMIAL REGRESSION AND SOFTMAX

This approach can be extended in exactly the same fashion for other score based models. For discrimnating each class k from all others, we define a binary score model $f_k(x)$ with parameter vector θ_k. We then combine these models through the softmax function

$$\mathbb{P}(y = k|x) = \pi_k(x) = s_k(f_1(x), \dots f_g(x))$$

and optimize all parameter vectors of the f_k jointly.

ESTIMATING PROBABILITIES - MULTINOMIAL REGRESSION AND SOFTMAX

Further comments:

- For linear $f(x|\theta) = \theta^T x$, this is also called *softmax regression*. (Note that x can include derived features like polynomials or interactions as well.)
- One set of parameters is "redundant": If we subtract any fixed vector from all θ_k, the predictions do not change. The model is "overparameterized", the minimizer of $\mathcal{R}_{\text{emp}}(\theta)$ is not unique. Hence, we set $\theta_g = (0, \ldots, 0)$ and only optimize the other θ_k, $k = 1, \ldots, g - 1$.
  (Compare: logistic regression for binary classification also has only *one* parameter vector for discriminating between two classes...).
- A similar approach can be used for many ML models: multiclass LDA, naive Bayes, neural networks and boosting.

LOGISTIC AND SOFTMAX REGRESSION

Representation: Design matrix X, coefficients θ.

Evaluation: Logistic/Bernoulli loss function.

Optimization: Numerical optimization, typically gradient descent based methods.