Binary classification using logistic regression

Shreyas Pandit

10th September 2021

1. Introduction

These are some notes that I have made surrounding the mathematical theory behind binary classification using logistic regression. The accompanying code can be found here.

In binary classification, we would like to predict a binary outcome (either 0 or 1), eg. if an email is spam or not, or whether a customer will buy something or not. The data we have available is of the form $(\mathbf{x}_i, y_i)$, where $y_i \in \{0, 1\}$ is our outcome, and $\mathbf{x}_i$ is a vector with our input data (factors that affect the outcome), referred to as features. Suppose we have m data samples and n different features. We will also assume throughout that the vectors are row vectors. Our goal is to predict the probability that a given input should be classified as 1.

Define

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

The model we will use is: the probability that a given input $\mathbf{x}$ is classified as 1 is $\sigma(\mathbf{x}\theta^T)$. Here, $\theta = (\theta_0, \theta_1, \dots \theta_n)$ are parameters that we will vary according to the training data we were given. In particular, we will vary θ to minimise the error in our predictions. Note that the first component of $\mathbf{x}$ is always 1, which is just so $\mathbf{x}\theta^T = \theta_0 + \theta_1 x_1 + \dots \theta_n x_n$.

Recall that given some data and a probability model depending on a parameter μ, the likelihood function $L(\mu)$ is the probability of observing this data given the parameter μ. The log-likelihood is $\log L(\mu)$.

Proposition 1. *The log-likelihood function for our model is*

$$l(\theta) = \sum_{i=1}^{m} \big(y_i \log(\sigma(\mathbf{x}_i\theta^T)) + (1 - y_i) \log(1 - \sigma(\mathbf{x}_i\theta^T)) \big)$$

Proof. Notice that the likelihood function is

$$L(\theta) = \prod_{\substack{1 \le i \le m \\ y_i = 1}} \sigma(\mathbf{x}_i \theta^T) \prod_{\substack{1 \le i \le m \\ y_0 = 1}} \big(1 - \sigma(\mathbf{x}_i \theta^T)\big).$$

We can write this compactly as

$$L(\theta) = \prod_{1 \le i \le m} \sigma(\mathbf{x}_i \theta^T)^{y_i} \big(1 - \sigma(\mathbf{x}_i \theta^T)\big)^{1-y_i}.$$

Hence,

$$l(\theta) = \log(L(\theta)) = \sum_{i=1}^{m} \big(y_i \log(\sigma(\mathbf{x}_i \theta^T)) + (1 - y_i) \log(1 - \sigma(\mathbf{x}_i \theta^T))\big)\,.$$

□

Since we want to maximise the likelihood, we will consider the negative log-likelihood function. To find the cost function, we will also average over all of the training examples.

Proposition 2. *The cost function is*

$$J(\theta) = -\frac{1}{m} \sum_{i=1}^{m} \big(y_i \log(\sigma(\mathbf{x}_i \theta^T)) + (1 - y_i) \log(1 - \sigma(\mathbf{x}_i \theta^T))\big)\,.$$

To minimise this, we will use gradient descent. Hence, we would like to find the gradient vector of J.

Proposition 3. *Let* $\mathbf{X}$ *be the matrix given by*

$$\mathbf{X} = \begin{pmatrix} \text{—} & \mathbf{x}_1 & \text{—} \\ \text{—} & \mathbf{x}_2 & \text{—} \\ & \cdots & \\ \text{—} & \mathbf{x}_m & \text{—} \end{pmatrix}$$

We have that

$$\nabla J = \frac{1}{m} \mathbf{X}^T (\sigma(\mathbf{X}\theta^T) - \mathbf{y}).$$

Proof. For each integer k, with $1 \le k \le m$, we have that

$$\begin{aligned} \frac{\partial J}{\partial \theta_k} &= -\frac{1}{m} \left(\sum_{i=1}^{m} y_i \frac{\partial}{\partial \theta_k} \big(\log(\sigma(\mathbf{x}_i \theta^T))\big) + \sum_{i=1}^{m} (1 - y_i) \frac{\partial}{\partial \theta_k} \big(\log(1 - \sigma(\mathbf{x}_i \theta^T))\big) \right) \\ &= -\frac{1}{m} \left(\sum_{i=1}^{m} y_i \frac{\sigma'(\mathbf{x}_i \theta^T)}{\sigma(\mathbf{x}_i \theta^T)} [\mathbf{x}_i]_k - \sum_{i=1}^{m} (1 - y_i) \frac{\sigma'(\mathbf{x}_i \theta^T)}{(1 - \sigma(\mathbf{x}_i \theta^T))} [\mathbf{x}_i]_k \right). \end{aligned}$$

Recall that $\sigma'(x) = \sigma(x)\sigma(1-x)$. Hence,

$$\begin{aligned}
\frac{\partial J}{\partial \theta_k} &= -\frac{1}{m}\left(\sum_{i=1}^{m}\big(y_i(1-\sigma(\mathbf{x}_i\theta^T)) - (1-y_i)\sigma(\mathbf{x}_i\theta^T)\big)\,[\mathbf{x}_i]_k\right) \\
&= -\frac{1}{m}\sum_{i=1}^{m}[\mathbf{x}_i]_k(y_i - \sigma(\mathbf{x}_i\theta^T)) \\
&= \frac{1}{m}\sum_{i=1}^{m}[\mathbf{x}_i]_k(\sigma(\mathbf{x}_i\theta^T) - y_i).
\end{aligned}$$

Now, observe that

$$\sigma(\mathbf{X}\theta^T) = \begin{pmatrix} \sigma(\mathbf{x}_1\theta^T) \\ \sigma(\mathbf{x}_2\theta^T) \\ \vdots \\ \sigma(\mathbf{x}_m\theta^T) \end{pmatrix}$$

and so

$$\mathbf{X}^T(\sigma(\mathbf{X}\theta^T) - \mathbf{y}) = \begin{pmatrix} | & & | \\ \mathbf{x}_1 & \dots & \mathbf{x}_m \\ | & & | \end{pmatrix} \begin{pmatrix} \sigma(\mathbf{x}_1\theta^T) - y_1 \\ \sigma(\mathbf{x}_2\theta^T) - y_2 \\ \vdots \\ \sigma(\mathbf{x}_m\theta^T) - y_m \end{pmatrix}.$$

Expanding out this matrix multiplication and comparing with the above result for $\frac{\partial J}{\partial \theta_k}$, we obtain the result.

$\square$

We can now apply gradient descent: $\theta \leftarrow \theta - \alpha\nabla J$, where α is the learning rate.

References

[1] G. James et al. *An Introduction to Statistical Learning: with Applications in R.* Springer Texts in Statistics. Springer New York, s2013. ISBN: 9781461471387. URL: `https://books.google.co.uk/books?id=qcI\_AAAAQBAJ`.