9.1.2 - Binary Classification

9.1.2 - Binary Classification

Let's look at binary classification first. In fact, when we go to more than two classes the formulas become much more complicated, although they are derived similarly. We will begin by looking at this simpler case.

For binary classification, if g_i = class 1, denote y_i = 1; if g_i = class 2, denote y_i = 0. All we are doing here is changing the labels to 1's and 0's so that the notation will be simpler.

Let $p_1(x;\theta) = p(x;\theta)$.

Then $p_2(x;\theta) = 1 - p_1(x;\theta) = 1 - p(x;\theta)$ because the posterior probabilities of the two classes have to sum up to 1.

Since *K* = 2 we only have one linear equation and one decision boundary between two classes, the parameters $\theta = \beta_{10}, \beta_1$, (remember, β_1 is a vector). And, we denote $\beta = (\beta_{10}, \beta_1)^T$ which is a column vector.

Now, let's try to simplify the equation a little bit.

If $y_i = 1$, i.e.,$g_i = 1$, then

$$\begin{aligned}\log p_{g_i}(x;\beta) &= \log p_1(x;\beta)\\ &= 1 \cdot \log p(x;\beta)\\ &= y_i \log p(x;\beta)\end{aligned}$$

If $y_i = 0$, i.e., $g_i = 2$,

$$\begin{aligned}\log p_{g_i}(x;\beta) &= \log p_2(x;\beta)\\ &= 1 \cdot \log\,(1 - p(x;\beta))\\ &= (1 - y_i)\log\,(1 - p(x;\beta))\end{aligned}$$

Since either $y_i = 0$ or $1 - y_i = 0$, we can add the two (at any time, only one of the two is nonzero) and have:

$$\log p_{g_i}(x;\beta) = y_i \log p(x;\beta) + (1 - y_i)\log\,(1 - p(x;\beta))$$

The reason for doing this is that when we later derive the logistic regression we do not want to work with different cases. It would be tedious in notation if we have to distinguish different cases each time. This unified equation is always correct for whatever y_i you use.

Next, we will plug what we just derived into the log-likelihood, which is simply a summation over all the points:

$$\begin{aligned}l(\beta) &= \sum_{i=1}^{N} \log p_{g_i}(x_i;\beta)\\ &= \sum_{i=1}^{N} (y_i \log p(x_i;\beta) + (1 - y_i)\log\,(1 - p(x_i;\beta)))\end{aligned}$$

There are *p* + 1 parameters in $\beta = (\beta_{10}, \beta_1)^T$.

Assume a column vector form for β:

$$\beta = \begin{pmatrix}\beta_{10}\\ \beta_{11}\\ \beta_{12}\\ \vdots\\ \beta_{1,p}\end{pmatrix}$$

Here we add the constant term 1 to *x* to accommodate the intercept and keep this in matrix form.

$$x = \begin{pmatrix} 1 \\ x_{,1} \\ x_{,2} \\ \vdots \\ x_{,p} \end{pmatrix}$$

Under the assumption of the logistic regression model:

$$\begin{aligned} p(x;\beta) = Pr(G=1|X=x) &= \frac{\exp(\beta^T x)}{1+\exp(\beta^T x)} \\ 1-p(x;\beta) = Pr(G=2|X=x) &= \frac{1}{1+\exp(\beta^T x)} \end{aligned}$$

we substitute the above in $l(\beta)$:

$$l(\beta) = \sum_{i=1}^{N} \left(y_i \beta^T x_i - \log\left(1+e^{\beta^T x_i}\right) \right)$$

The idea here is based on basic calculus. If we want to maximize $l(\beta)$, we set the first order partial derivatives of the function $l(\beta)$ with respect to β_{1j} to zero.

$$\begin{aligned} \frac{\partial l(\beta)}{\beta_{1j}} &= \sum_{i=1}^{N} y_i x_{ij} - \sum_{i=1}^{N} \frac{x_{ij} e^{\beta^T x_i}}{1+e^{\beta^T x_i}} \\ &= \sum_{i=1}^{N} y_i x_{ij} - \sum_{i=1}^{N} p(x;\beta) x_{ij} \\ &= \sum_{i=1}^{N} x_{ij}(y_i - p(x_i;\beta)) \end{aligned}$$

for all *j* = 0, 1, ... , *p*.

And, if we go through the math, we will find out that it is equivalent to the matrix form below.

$$\frac{\partial l(\beta)}{\beta_{1j}} = \sum_{i=1}^{N} x_i(y_i - p(x_i;\beta))$$

Here x_i is a column vector and y_i is a scalar.

To solve the set of *p* +1 nonlinear equations $\frac{\partial l(\beta)}{\partial \beta_{1j}} = 0$, *j* = 0, 1, ... , *p* , we will use the Newton-Raphson algorithm.

The Newton-Raphson algorithm requires the second-order derivatives or the so-called Hessian matrix (a square matrix of the second order derivatives) which is given by:

$$\frac{\partial^2 l(\beta)}{\partial\beta\partial\beta^T} = -\sum_{i=1}^{N} x_i x_i^T p(x_i;\beta)(1-p(x_i;\beta))$$

Below we derive the Hessian matrix, where the element on the *j*th row and *n*th column is (counting from 0):

$$\begin{aligned} &\frac{\partial l(\beta)}{\partial\beta_{1j}\partial\beta_{1n}} \\ &= -\sum_{i=1}^{N} \frac{(1+e^{\beta^T x_i}) e^{\beta^T x_i} x_{ij} x_{in} - (e^{\beta^T x_i})^2 x_{ij} x_{in}}{(1+e^{\beta^T x_i})^2} \\ &= -\sum_{i=1}^{N} x_{ij} x_{in} p(x_i;\beta) - x_{ij} x_{in} p(x_i;\beta)^2 \\ &= -\sum_{i=1}^{N} x_{ij} x_{in} p(x_i;\beta)(1-p(x_i;\beta)) \end{aligned}$$

Starting with β^{old}, a single Newton-Raphson update is given by this matrix forumla:

$$\beta^{new} = \beta^{old} - \left(\frac{\partial^2 l(\beta)}{\partial\beta\partial\beta^T}\right)^{-1} \frac{\partial l(\beta)}{\partial\beta}$$

Basically, if given an old set of parameters, we update the new set of parameters by taking β^{old} minus the inverse of the Hessian matrix times the first order derivative vector. These derivatives are all evaluated at β^{old}.

The iteration can be expressed compactly in matrix form.

- Let ***y*** be the column vector of y_i.
- Let $\mathbf{X}$ be the *N* × (*p* + 1) input matrix.
- Let $\mathbf{p}$ be the N-vector of fitted probabilities with *i*th element $p(x_i; \beta^{old})$.
- Let $\mathbf{W}$ be an *N* × *N* diagonal matrix of weights with *i*th element $p(x_i; \beta^{old})(1 - p(x_i; \beta^{old}))$.
- If you have set up all the matrices properly then the first order derivative vector is:

  $$\frac{\partial l(\beta)}{\partial\beta} = \mathbf{X}^T(\mathbf{y} - \mathbf{p})$$

  and the Hessian matrix is:

  $$\frac{\partial^2 l(\beta)}{\partial\beta\partial\beta^T} = -\mathbf{X}^T\mathbf{W}\mathbf{X}$$

The Newton-Raphson step is:

$$\begin{aligned}\beta^{new} &= \beta^{old} + (\mathbf{X}^T\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^T(\mathbf{y} - \mathbf{p}) \\ &= (\mathbf{X}^T\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W}(\mathbf{X}\beta^{old} + \mathbf{W}^{-1}(\mathbf{y} - \mathbf{p})) \\ &= (\mathbf{X}^T\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W}\mathbf{z}\end{aligned}$$

where $\mathbf{z} \overset{\triangle}{=} \mathbf{X}\beta^{old} + \mathbf{W}^{-1}(\mathbf{y} - \mathbf{p})$

If $\mathbf{z}$ is viewed as a response and $\mathbf{X}$ is the input matrix, β^{new} is the solution to a weighted least square problem:

$$\beta^{new} \leftarrow \arg \underset{\beta}{min} (\mathbf{z} - \mathbf{X}\beta)^T\mathbf{W}(\mathbf{z} - \mathbf{X}\beta)$$

Recall that linear regression by least square solves

$$\underset{\beta}{min} (\mathbf{z} - \mathbf{X}\beta)^T\mathbf{W}(\mathbf{z} - \mathbf{X}\beta)$$

$\mathbf{z}$ is referred to as the *adjusted response*.

The algorithm is referred to as *iteratively reweighted least squares* or *IRLS*.

The pseudo code for the IRLS algorithm is provided below.

Pseudo Code

1. $0 \rightarrow \beta$
2. Compute *y* by setting its elements to:

   $$y_i = \begin{cases} 1 & \text{if } g_i = 1 \\ 0 & \text{if } g_i = 2 \end{cases}$$

3. Compute *p* by setting its elements to:

   $$p(x_i; \beta) = \frac{e^{\beta^T x_i}}{1 + e^{\beta^T x_i}} i = 1, 2, \dots, N$$

4. Compute the diagonal matrix $\mathbf{W}$. The *i*th diagonal element is $p(x_i; \beta)(1 - p(x_i; \beta))$, *i* =1, 2, ... , *N*.
5. $\mathbf{z} \leftarrow \mathbf{X}\beta + \mathbf{W}^{-1}(\mathbf{y} - \mathbf{p})$.
6. $\beta \leftarrow (\mathbf{X}^T\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W}\mathbf{z}$.
7. If the stopping criteria are met, stop; otherwise go back to step 3.

Computational Efficiency

Since **W** is an *N* × *N* diagonal matrix, direct matrix operations with it, as shown in the above pseudo code, is inefficient.

A modified pseudo code with much improved computational efficiency is given below.

1. $0 \rightarrow \beta$
2. Compute *y* by setting its elements to:

$$y_i = \begin{cases} 1 & \text{if } g_i = 1 \\ 0 & \text{if } g_i = 2 \end{cases}$$

3. Compute *p* by setting its elements to:

$$p(x_i;\beta) = \frac{e^{\beta^T x_i}}{1 + e^{\beta^T x_i}} i = 1, 2, \ldots, N$$

4. Compute the *N* × (*p* +1) matrix $\tilde{\mathbf{X}}$ by multiplying the *i*th row of matrix $\mathbf{X}$ by $p(x_i;\beta)(1 - p(x_i;\beta))$, *i* =1, 2, ... , *N*:

$$\mathbf{X} = \begin{pmatrix} x_1^T \\ x_2^T \\ \cdots \\ x_N^T \end{pmatrix} \tilde{\mathbf{X}} = \begin{pmatrix} p(x_1;\beta)(1 - p(x_1;\beta))x_1^T \\ p(x_2;\beta)(1 - p(x_2;\beta))x_2^T \\ \cdots \\ p(x_N;\beta)(1 - p(x_N;\beta))x_N^T \end{pmatrix}$$

5. $\beta \leftarrow \beta + (\mathbf{X}^T \tilde{\mathbf{X}})^{-1} \mathbf{X}^T (\mathbf{y} - \mathbf{p})$.
6. If the stopping criteria are met, stop; otherwise go back to step 3.

Legend

[1]	Link
↥	Has Tooltip/Popover
⬚	Toggleable Visibility

Source: https://online.stat.psu.edu/stat508/lesson/9/9.1/9.1.2

Links: