Logistic Regression

Ruoqing Zhu

Last Updated: September 26, 2024

Contents

Modeling Binary Outcomes 1
Example: Cleveland Clinic Heart Disease Data 2
Interpretation of the Parameters 3
Solving a Logistic Regression 4
Example: South Africa Heart Data 4
Penalized Logistic Regression 6

Modeling Binary Outcomes

To model binary outcomes using a logistic regression, we will use the 0/1 coding of Y. We need to set its connection with covariates. Recall in a linear regression, the outcome is continuous, and we set

$$Y = \beta_0 + \beta_1 X + \epsilon$$

However, this does not work for classification since Y can only be 0 or 1. Hence we turn to consider modeling the probability $P(Y = 1|X = \mathbf{x})$. Hence, Y is a Bernoulli random variable given X, and this is modeled by a function of X:

$$P(Y = 1|X = \mathbf{x}) = \frac{\exp(\mathbf{x}^{\mathrm{T}}\boldsymbol{\beta})}{1 + \exp(\mathbf{x}^{\mathrm{T}}\boldsymbol{\beta})}$$

Note that although $\mathbf{x}^{\mathrm{T}}\boldsymbol{\beta}$ may ranges from 0 to infinity as X changes, the probability will still be bounded between 0 and 1. This is an example of **Generalized Linear Models**. The relationship is still represented using a linear function of $\mathbf{x}$, $\mathbf{x}^{\mathrm{T}}\boldsymbol{\beta}$. This is called a **logit link** function (a function to connect the conditional expectation of Y with $\boldsymbol{\beta}^{\mathrm{T}}\mathbf{x}$):

$$\eta(a) = \frac{\exp(a)}{1 + \exp(a)}$$

Hence, $P(Y = 1|X = \mathbf{x}) = \eta(\mathbf{x}^{\mathrm{T}}\boldsymbol{\beta})$. We can reversely solve this and get

$$\begin{aligned} P(Y = 1|X = \mathbf{x}) = \eta(\mathbf{x}^{\mathrm{T}}\boldsymbol{\beta}) &= \frac{\exp(\mathbf{x}^{\mathrm{T}}\boldsymbol{\beta})}{1 + \exp(\mathbf{x}^{\mathrm{T}}\boldsymbol{\beta})} \\ 1 - \eta(\mathbf{x}^{\mathrm{T}}\boldsymbol{\beta}) &= \frac{1}{1 + \exp(\mathbf{x}^{\mathrm{T}}\boldsymbol{\beta})} \\ \text{Odds} = \frac{\eta(\mathbf{x}^{\mathrm{T}}\boldsymbol{\beta})}{1 - \eta(\mathbf{x}^{\mathrm{T}}\boldsymbol{\beta})} &= \exp(\mathbf{x}^{\mathrm{T}}\boldsymbol{\beta}) \\ \log(\text{Odds}) &= \mathbf{x}^{\mathrm{T}}\boldsymbol{\beta} \end{aligned}$$

Hence, the parameters in a logistic regression is explained as **log odds**. Let's look at a concrete example.

Example: Cleveland Clinic Heart Disease Data

We use use the Cleveland clinic heart disease dataset. The goal is to model and predict a class label of whether the patient has a hearth disease or not. This is indicated by whether the `num` variable is 0 (no presence) or > 0 (presence).

```
  heart = read.csv("processed_cleveland.csv")
  heart$Y = as.factor(heart$num > 0)
  table(heart$Y)
##
## FALSE  TRUE
##   164   139
```

For simplicity, let's model the probability of heart disease using the `Age` variable. This can be done using the `glm()` function, which stands for the Generalized Linear Model. The syntax of `glm()` is almost the same as a linear model. Note that it is important to use `family = binomial` to specify the logistic regression.

```
  logistic.fit <- glm(Y~age, data = heart, family = binomial)
  summary(logistic.fit)
##
## Call:
## glm(formula = Y ~ age, family = binomial, data = heart)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -3.00591    0.75913  -3.960  7.5e-05 ***
## age          0.05199    0.01367   3.803 0.000143 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 417.98  on 302  degrees of freedom
## Residual deviance: 402.54  on 301  degrees of freedom
## AIC: 406.54
##
## Number of Fisher Scoring iterations: 4
```

The result is similar to a linear regression, with some differences. The parameter estimate of age is 0.05199. It is positive, meaning that increasing age would increase the change of having heart disease. However, this does not mean that 1 year older would increase the change by 0.05. Since, by our previous formula, the probably is not directly expressed as $\mathbf{x}^{\mathrm{T}}\boldsymbol{\beta}$.

This calculation can be realized when predicting a new target point. Let's consider a new subject with `Age = 55`. What is the predicted probability of heart disease? Based on our formula, we have

$$\beta_0 + \beta_1 X = -3.00591 + 0.05199 \times 55 = -0.14646$$

And the estimated probability is

$$P(Y = 1|X) = \frac{\exp(\beta_0 + \beta_1 X)}{1 + \exp(\beta_0 + \beta_1 X)} = \frac{\exp(-0.14646)}{1 + \exp(-0.14646)} = 0.4634503$$

Hence, the estimated probability for this subject is 46.3%. This can be done using R code. Please note that if you want to predict the probability, you need to specify `type = "response"`. Otherwise, only $\beta_0 + \beta_1 X$ is provided.

```
  testdata = data.frame("age" = 55)
  predict(logistic.fit, newdata = testdata)
##          1
## -0.1466722
  predict(logistic.fit, newdata = testdata, type = "response")
##         1
## 0.4633975
```

If we need to make a 0/1 decision about this subject, a natural idea is to see if the predicted probability is greater than 0.5. In this case, we would predict this subject as 0.

Interpretation of the Parameters

Recall that $\mathbf{x}^{\mathrm{T}}\boldsymbol{\beta}$ is the log odds, we can further interpret the effect of a single variable. Let's define the following two, with an arbitrary age value a:

- A subject with `age` $= a$
- A subject with `age` $= a + 1$

Then, if we look at the **odds ratio** corresponding to these two target points, we have

$$\begin{aligned}
\text{Odds Ratio} &= \frac{\text{Odds of Subject 2}}{\text{Odds of Subject 1}} \\
&= \frac{\exp(\beta_0 + \beta_1(a+1))}{\exp(\beta_0 + \beta_1 a)} \\
&= \frac{\exp(\beta_0 + \beta_1 a) \times \exp(\beta_1)}{\exp(\beta_0 + \beta_1 a)} \\
&= \exp(\beta_1)
\end{aligned}$$

Taking log on both sides, we have

$$\log(\text{Odds Ratio}) = \beta_1$$

Hence, the odds ratio between these two subjects (**they differ only with one unit of `age`**) can be directly interpreted as the exponential of the parameter of `age`. After taking the log, we can also say that

> The parameter β of a varaible in a logistic regression represents the **log of odds ratio** associated with one-unit increase of this variable.

Please note that we usually do not be explicit about what this odds ratio is about (what two subject we are comparing). Because the interpretation of the parameter does not change regardless of the value a, as long as the two subjects differ in one unit.

And also note that this conclusion is regardless of the values of other covaraites. When we have a multivariate model, as long as all other covariates are held the same, the previous derivation will remain the same.

Solving a Logistic Regression

The logistic regression is solved by maximizing the log-likelihood function. Note that the log-likelihood is given by

$$\ell(\boldsymbol{\beta}) = \sum_{i=1}^{n} \log\, p(y_i|x_i, \boldsymbol{\beta}).$$

Using the probabilities of Bernoulli distribution, we have

$$\ell(\boldsymbol{\beta}) = \sum_{i=1}^{n} \log\left\{\eta(\mathbf{x}_i)^{y_i}[1-\eta(\mathbf{x}_i)]^{1-y_i}\right\} \tag{1}$$

$$= \sum_{i=1}^{n} y_i \log\frac{\eta(\mathbf{x}_i)}{1-\eta(\mathbf{x}_i)} + \log[1-\eta(\mathbf{x}_i)] \tag{2}$$

$$= \sum_{i=1}^{n} y_i \mathbf{x}_i^{\mathrm{T}}\boldsymbol{\beta} - \log[1+\exp(\mathbf{x}_i^{\mathrm{T}}\boldsymbol{\beta})] \tag{3}$$

Since this objective function is relatively simple, we can use Newton's method to update. The key is to derive the gradient and Hessian functions. For details, please see the SMLR text book. Instead of solving them ourselves, we can simply utilize the `optim()` function to perform the optimization for us.

Example: South Africa Heart Data

We use the South Africa heart data as a demonstration. The goal is to estimate the probability of `chd`, the indicator of coronary heart disease. Using the `glm()` function, we can obtain the estimated parameters.

```
    library(ElemStatLearn)
    data(SAheart)

    heart = SAheart
    heart$famhist = as.numeric(heart$famhist)-1
    n = nrow(heart)
    p = ncol(heart)

    heart.full = glm(chd~., data=heart, family=binomial)
    round(summary(heart.full)$coef, dig=3)
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -6.151      1.308  -4.701    0.000
## sbp            0.007      0.006   1.135    0.256
## tobacco        0.079      0.027   2.984    0.003
## ldl            0.174      0.060   2.915    0.004
## adiposity      0.019      0.029   0.635    0.526
## famhist        0.925      0.228   4.061    0.000
## typea          0.040      0.012   3.214    0.001
## obesity       -0.063      0.044  -1.422    0.155
## alcohol        0.000      0.004   0.027    0.978
## age            0.045      0.012   3.728    0.000

    # fitted value
    yhat = (heart.full$fitted.values>0.5)
```

```
    table(yhat, SAheart$chd)
##
## yhat      0   1
##   FALSE 256  77
##   TRUE   46  83
```

Based on our understandings of optimization we can use the general solver `optim()` by providing the objective function, for any value of $\boldsymbol{\beta}$, $\mathbf{X}$ and $\mathbf{y}$.

```
    # the negative log-likelihood function of logistic regression
    my.loglik <- function(b, x, y)
    {
        bm = as.matrix(b)
        xb =  x %*% bm
        # this returns the negative loglikelihood
        return(sum(y*xb) - sum(log(1 + exp(xb))))
    }
```

Let's check the result of this function for some arbitrary $\boldsymbol{\beta}$ value.

```
    # prepare the data matrix, I am adding a column of 1 for intercept
    x = as.matrix(cbind("intercept" = 1, heart[, 1:9]))
    y = as.matrix(heart[,10])

    # check my function
    b = rep(0, ncol(x))
    my.loglik(b, x, y) # scalar
## [1] -320.234

    # check the optimal value and the likelihood
    my.loglik(heart.full$coefficients, x, y)
## [1] -236.07
```

Then we optimize this objective function. Note that since this is a maximization problem, we need to either re-define the objective function or use the `fnscale` argument.

```
    # Use a general solver to get the optimal value
    # Note that we are doing maximization instead of minimization,
    # we need to specify "fnscale" = -1
    optim(b, fn = my.loglik, method = "BFGS",
          x = x, y = y, control = list("fnscale" = -1))
## $par
##  [1] -6.133036522  0.006357461  0.079318574  0.173889757  0.018589725  0.925594371  0.039552578 -0.0
##
## $value
## [1] -236.0704
##
## $counts
## function gradient
##       96       15
##
## $convergence
## [1] 0
```

```
##
## $message
## NULL
```

This matches our `glm()` solution. An example of this optimization problem with more accurate solutions is provided at the SMLR text book.

Penalized Logistic Regression

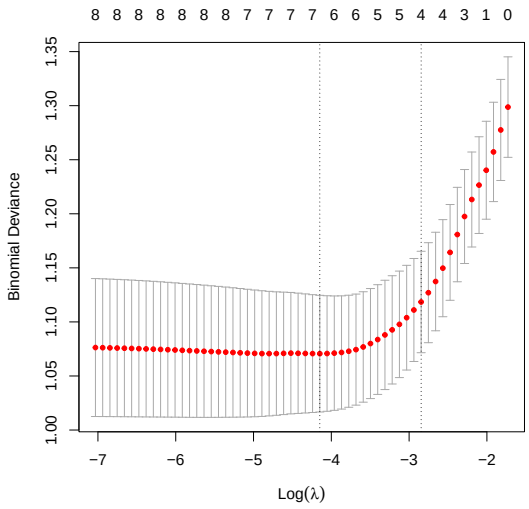
Similar to a linear regression, we can also apply penalties to a logistic regression to address collinearity problems or select variables in a high-dimensional setting. For example, if we use the Lasso penalty, the objective function is

$$\sum_{i=1}^{n} \log\, p(y_i|x_i, \boldsymbol{\beta}) + \lambda|\boldsymbol{\beta}|_1$$

This can be done using the `glmnet` package. Specifying `family = "binomial"` will ensure that a logistic regression is used, even your `y` is not a factor (but as numerical 0/1).

```
  library(glmnet)
## Loading required package: Matrix
## Loaded glmnet 4.1-8
  lasso.fit = cv.glmnet(x = data.matrix(SAheart[, 1:9]), y = SAheart[,10],
                        nfold = 10, family = "binomial")

  plot(lasso.fit)
```



The procedure is essentially the same as in a linear regression. And we could obtain the estimated parameters by selecting the best λ value.

```
  coef(lasso.fit, s = "lambda.min")
## 10 x 1 sparse Matrix of class "dgCMatrix"
##                        s1
```

```
## (Intercept) -6.137091517
## sbp          0.002737629
## tobacco      0.065709994
## ldl          0.129753596
## adiposity    .
## famhist      0.750635609
## typea        0.024733089
## obesity     -0.001947330
## alcohol      .
## age          0.041389624
```