LOGISTIC REGRESSION IN Q

Tara Mirmira and Ramesh Subramonian

1 Objective

We have observations that fall into one of two classes: 0 or 1. Given a new observation, we want to predict if the observation will be in class 0 or class 1.

We would like to use a linear model to make our predictions, meaning we would like to express the class of an observation as a linear combination of the explanatory variables. We consider the simplest linear model, the linear probability model, in the next section.

2 Linear Probability Model

Consider the model $y_i = \alpha + \beta x_i + \epsilon_i$

- y_i is the class associated with x_i, and is what we want to predict
- ϵ_i are independent and identically distributed with mean of 0 and a variance of σ^2
- $E(y_i) = \alpha + \beta x_i$
- $E(y_i) = 0 * P(y_i = 1) + 1 * P(y_i = 0) = P(y_i = 1) = \pi_i$
- $\pi_i = \alpha + \beta_i$

The problem is ϵ_i can only take on the values 0 or 1 so $Var(\epsilon_i) = (1 - \pi_i)^2\pi_i + -\pi_i^2(1 - \pi_i) = \pi_i(1 - \pi_i)$ which is not constant and non constant variance causes all sorts of problems for linear modelling.

This simple model does not work so we will present a more complex linear model in the next section.

3 Alternate Model to Linear Probability Model - Linear Model Using Link Function

General overview:

- Let π_i be the probability that observation i is in class 1.

- Find a function G such that $\pi_i = G(\alpha + \beta_i)$
- We want G to be invertible so we can say $G^{-1}(\pi_i) = \alpha + \beta x_i$
- For logistic regression, we will use the function $G(z) = \frac{1}{1+e^{-z}}$
- $\frac{1}{1+e^{-z}}$ is bounded between 0 and 1 for all values of z.

Using the above link function:

- $P(y_i = 1) = \pi_i = G(\alpha + \beta_i) = \dfrac{1}{1 + e^{-(\alpha+\beta x_i)}} = \dfrac{1}{1 + e^{-(\alpha+\beta x_i)}} * \dfrac{e^{\alpha+\beta x_i}}{e^{\alpha+\beta x_i}} = \dfrac{e^{\alpha+\beta x_i}}{e^{\alpha+\beta x_i} + 1}$
- $P(y_i = 0) = 1 - P(y_i = 1) = 1 - \dfrac{e^{\alpha+\beta x_i}}{e^{\alpha+\beta x_i} + 1} = \dfrac{1}{1 + e^{\alpha+\beta x_i}}$

The odds ratio: the ratio of the probability of the observation being in one category versus the probability of being in the other category

- odds ratio = $\dfrac{\pi}{1-\pi} = \dfrac{e^{\alpha+\beta x_i}}{1} = e^{\alpha+\beta x_i} = e^{\alpha} e^{\beta x_i}$ which means that when we increase x_i by one unit, the odds increase by e^{β}
- $\log(\text{odds ratio}) = \log\big(\dfrac{\pi}{1-\pi}\big) = \log(e^{\alpha+\beta x_i}) = \alpha + \beta x_i$

From the above derivation, we see that we have found the **link function**:

$\log\big(\dfrac{\pi_i}{1-\pi_i}\big) = \alpha + \beta x_i$

The log of the odds ratio is linear in the $x_i's$. The link function connects $E(y_i) = \pi_i$ to a linear function of the explanatory variables x_i.

4 Fitting a Model to the Data Using Maximum Likelihood

General overview:

- We will use maximum likelihood to estimate α and β. We want to find α and β that makes the given data most likely to occur and use these values to predict future values.
- The response variables are the classes 0 or 1, which means the response variables are $Bernoulli(\pi_i)$ variables.

Maximum Likelihood:

Recall a few probabilities:

- $P(i^{th}$ observation $= 1) = \pi_i$
- $P(i^{th}$ observation $= 0) = 1 - \pi_i$
- $P(i^th$ observation $= y_i) = \pi_i^{y_i}$ if $y_i = 1$
- $P(i^th$ observation $= y_i) = (1 - \pi_i)^{1-y_i}$ if $y_i = 0$
- Using the above two bullet points, $P(i^th$ observation $= y_i) = \pi_i^{y_i}(1 - \pi_i)^{1-y_i}$

$$\text{Likelihood } = \mathscr{L}(y_1 \dots y_n) = \prod_{i=1}^{n} p(y_i) = \prod_{i=1}^{n} P(i^{th} \text{ observation } = y_i) = \prod_{i=1}^{n} \pi_i^{y_i}(1 - \pi_i)^{1-y_i} =$$

$$\prod_{i=n}^{n} \big(\frac{\pi_i}{1 - \pi_i}\big)^{y_i}(1 - \pi_i) = \prod_{i=1}^{n}(e^{\alpha+\beta x_i})^{y_i}\big(\frac{1}{1 + e^{\alpha+\beta x_i}}\big)$$

Log Likelihood:

$$\log \mathscr{L} = \sum_{i=1}^{n} y_i(\alpha + \beta x_i) - \log(1 + e^{\alpha+\beta x_i})$$

We want to find α and β that maximize the likelihood of observing the data. To maximize the likelihood (equivalent to maximizing the log likelihood), the next steps are to take the derivatives of the likelihood (or log likelihood) with respect to α and β, set the equations to 0, and solve for α and β.

Take the derivative with respect to α, set to 0 and solve:

$$\frac{\partial}{\partial \alpha} \log \mathscr{L} = \sum \big(y_i - \frac{1}{1 + e^{\alpha+\beta x_i}} e^{\alpha+\beta x_i}\big) = 0$$

$$\sum y_i = \sum \frac{e^{\hat{\alpha}+\hat{\beta}x_i}}{1+e^{\hat{\alpha}+\hat{\beta}x_i}}$$

Take the derivative with respect to β, set to 0 and solve:

$$\frac{\partial}{\partial \beta} \log \mathscr{L} = \sum \left(y_i x_i - \frac{1}{1+e^{\alpha+\beta x_i}} e^{\alpha+\beta x_i} x_i \right) = 0$$

$$\sum y_i x_i = \sum x_i \frac{e^{\hat{\alpha}+\hat{\beta}x_i}}{1+e^{\hat{\alpha}+\hat{\beta}x_i}} \qquad (1)$$

If we use the design matrix

$$\boldsymbol{X} = \begin{bmatrix} \mathbf{1} & \boldsymbol{x_1} & \dots & \boldsymbol{x_p} \end{bmatrix}$$

with p explanatory variables, and the response vector $\boldsymbol{y}$, the equation (1) becomes

$$\boldsymbol{X^t y} = \boldsymbol{X^t \hat{\pi}}$$

where $\hat{\pi}$ is the vector of estimated π_i values for the observations with values $(1, x_{1,i}, \dots, x_{p,i})$. $x_{j,i}$ are the observed data values and the 1 value is added so that an intercept value α is included in the linear model.

This equation is nonlinear so instead of solving directly for $\hat{\boldsymbol{\pi}}$, we will use an iterative method called Newton-Raphson.

5 Newton-Raphson

The Newton-Raphson method can be used to approximate the roots of a real-valued function.

Steps:

1. Start with an initial guess x_0 for the root
2. Iterative step: $x_{i+1} = x_i - \dfrac{f(x_i)}{f'(x_i)}$. Note that $(x_{i+1}, 0)$ is the intersection of the $x - axis$ and the line tangent to f at x_i
3. Repeat the iterative step until $f(x_i)$ is sufficiently close to 0. Then the value x_i is the approximation for the root.

The diagram at the top of the next page provides a visual for the above steps.

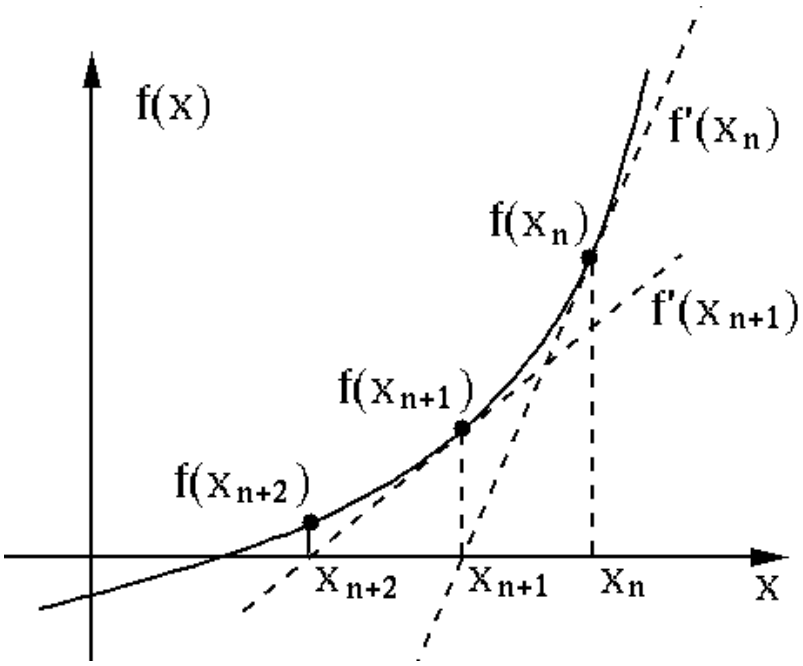

Figure 1: Newton Raphson Diagram

Next, we will show how to apply the Newton-Raphson method to find the estimate $\hat{\boldsymbol{\beta}}$ for $\boldsymbol{\beta}$. Note that we have switched to vector notation so instead of estimating α and β, we want to estimate $\boldsymbol{\beta} = \begin{bmatrix} \alpha \\ \beta_1 \\ \vdots \\ \beta_p \end{bmatrix}$ where β_i is the coefficient for the explanatory variable x_i and α is the intercept.

Suppose we want to find the maximum likelihood estimate $\hat{\theta_n}$ for the parameter θ. Let the derivative of the log likelihood function be the function ℓ'. We want to find the roots of this function, which means we want to find $\hat{\theta_n}$ such that $\ell'(\hat{\theta_n}) = 0$, which means $\hat{\theta_n}$ is the maximum likelihood estimate for θ

Recall the general Taylor Series expansion formula:

$$f(x) = f(a) + \frac{f'(a)}{1!}(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \dots$$

where f is infinitely differentiable at a.

Using a one term Taylor Series expansion, we get that:

$$\ell'(\hat{\theta_n}) = \ell'(\theta_0) + \ell''(\theta_0) * (\hat{\theta_n} - \theta_0)$$

It can be shown that $\ell''(\theta_0)$ can be estimated by $E[\ell''(\theta_0)] = -I(\theta_0)$ where I is the Fisher information.

Now we get:

$$\ell'(\hat{\theta_n}) = \ell'(\theta_0) - I(\theta_0) * (\hat{\theta_n} - \theta_0)$$

Rearranging terms we get:

$$\hat{\theta_n} = \theta_0 + \ell'(\theta_0)[I(\theta_0)]^{-1}$$

Now, we can implement Newton-Raphson in the following manner:

1. Start with the initial value θ_0
2. $\theta_{k+1} = \theta_k + \ell'(\theta_k)[I(\theta_k)]^{-1}$
3. Iterate until $\theta_k \approx \theta_{k+1}$ or until $\ell'(\theta_{k+1}) \approx 0$

For the case of logistic regression, the parameter $\theta = \boldsymbol{\beta}$

Although not discussed here, it can be shown that for Bernoulli data $I(\boldsymbol{\beta}_{(k)}) = \boldsymbol{X}^t \boldsymbol{W}_{(k)} \boldsymbol{X}$.

$\boldsymbol{W}_{(k)}$ is an $N \times N$ diagonal matrix with the i^{th} diagonal element is $\pi_i(1 - \pi_i)$ where π_i is the fitted probability for the i^{th} observation using $\boldsymbol{\beta}_{(k)}$

The following is the Newton-Raphson implementation for logistic regression:

1. Start with the initial value $\boldsymbol{\beta}_0 = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}$

2. $\hat{\boldsymbol{\beta}}_{(k+1)} = \hat{\boldsymbol{\beta}}_k + (\boldsymbol{X}^t \boldsymbol{W}_{(k)} \boldsymbol{X})^{-1}[\boldsymbol{X}^t \boldsymbol{y} - \boldsymbol{X}^t \hat{\boldsymbol{\pi}}] = \hat{\boldsymbol{\beta}}_k + (\boldsymbol{X}^t \boldsymbol{W}_{(k)} \boldsymbol{X})^{-1} \boldsymbol{X}^t (\boldsymbol{y} - \hat{\boldsymbol{\pi}})$
Matrix inversion can be costly. (See Don't invert that matrix for more details). We will manipulate the above equation so the iteration can be performed without needing to invert any matrices.

   $\hat{\boldsymbol{\beta}}_{(k+1)} - \hat{\boldsymbol{\beta}}_k = (\boldsymbol{X}^t \boldsymbol{W}_{(k)} \boldsymbol{X})^{-1} \boldsymbol{X}^t (\boldsymbol{y} - \hat{\boldsymbol{\pi}})$

   $(\boldsymbol{X}^t \boldsymbol{W}_{(k)} \boldsymbol{X})(\hat{\boldsymbol{\beta}}_{(k+1)} - \hat{\boldsymbol{\beta}}_k) = \boldsymbol{X}^t (\boldsymbol{y} - \hat{\boldsymbol{\pi}})$

   Use a matrix solver to solve for $\boldsymbol{x}$ in $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ where

   - $\boldsymbol{A} = \boldsymbol{X}^t \boldsymbol{W}_{(k)} \boldsymbol{X}$
   - $\boldsymbol{x} = \hat{\boldsymbol{\beta}}_{(k+1)} - \hat{\boldsymbol{\beta}}_k$
   - $\boldsymbol{b} = \boldsymbol{X}^t (\boldsymbol{y} - \hat{\boldsymbol{\pi}})$

   Note that $\boldsymbol{\beta}_{(k+1)} = \boldsymbol{x} + \hat{\boldsymbol{\beta}}_k$ and $\hat{\boldsymbol{\pi}} = \frac{e^{\boldsymbol{X}\boldsymbol{\beta}_{(k)}}}{1+e^{\boldsymbol{X}\boldsymbol{\beta}_{(k)}}}$

3. Iterate until $\hat{\boldsymbol{\beta}}$ has converged. At convergence, $(\boldsymbol{X}^t \boldsymbol{W}_{(k)} \boldsymbol{X})^{-1}$ is ≈ 0

6 How to Predict the Class of a New Observation

Recall the link function:

$$\log\left(\frac{\pi_i}{1-\pi_i}\right) = \alpha + \beta x_i$$

We can rewrite this in vector notation as the following:

$$\log\left(\frac{\pi_i}{1-\pi_i}\right) = \boldsymbol{x_i}^t \hat{\boldsymbol{\beta}}$$

where $\boldsymbol{x_i} = \begin{bmatrix} x_{1,i} \\ \vdots \\ x_{p,i} \end{bmatrix}$ is the vector that contains the data values for each explanatory variable x_i and $\hat{\boldsymbol{\beta}}$ is the vector of coefficients found by the Newton-Raphson calculation.

Given the link function:

$$\pi_i = \frac{e^{\alpha+\beta x_i}}{e^{\alpha+\beta x_i}+1}$$

In vector notation, this is:

$$\pi_i = \frac{e^{\boldsymbol{x_i}^t \hat{\boldsymbol{\beta}}}}{e^{\boldsymbol{x_i}^t \hat{\boldsymbol{\beta}}} + 1}$$

Given a new vector of observations $\boldsymbol{z}$, we can plug $\boldsymbol{z}$ in for $\boldsymbol{x_i}$ in the above formula and solve for π_i.

A basic classification procedure is the following:

- If $\pi_i \geq 0.5$, we classify the new observation as class 1
- If $\pi_i < 0.5$, we classify the new observation as class 0

The above classification procedure can be modified as desired.

7 *

References

[1] Lecture notes from Professor Deborah Nolan, UC Berkeley, STAT 151A, Spring 2017

[2] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning 2nd Edition*. Springer, New York, NY, 2009.

[3] Image citation: Ruye Wang - Newton-Raphson method (univariate)
`http://fourier.eng.hmc.edu/e176/lectures/NM/node20.html`

[4] "Don't invert that matrix" citation: John D. Cook
`https://www.johndcook.com/blog/2010/01/19/dont-invert-that-matrix/`

8 Data Structures

- X is an $N \times M$ matrix containing the input data concatenated with the value 1. $X_{i,j}$ is value of j^{th} attribute of i^{th} instance. X is stored as M columns, where X_j is observations for attribute j
- y is an $N \times 1$ classification vector. y_i is classification of instance i and can be 1 or 0.
- β is an $M \times 1$ coefficient vector. β_j is coefficient for attribute j.
- β^{new} is an $M \times 1$ vector, which are the new coefficients that we solve for in each iteration

- A is an $M \times M$ matrix. Since it is symmetric, we can skip computing the lower diagonal elements.
- $W = X^T W X$ is a diagonal $N \times N$ matrix. Since the off-diagonal elements are zero, we can represent is as an $N \times 1$ vector. When used as a vector, we will use lower case w. When used as a matrix, we will use upper case W. Note that $W_{i,i} = w_i$ and that $i \neq j \Rightarrow W_{i,j} = 0$
- b is an $M \times 1$ vector, $X^T W(X\beta + W^{-1}(y - p))$

With these elements, the Newton Raphson iteration becomes the following:
$(X^T W X) \times (\beta^{new} - \beta) = X^T(y - p)$

where

1. W is symmetric and positive definite
2. Because W is symmetric and positive definite, $A = X^T W X$ is at least positive semi-definite.
3. If the attributes are linearly independent, then A will actually be positive definite; else, the dependent attributes should be removed prior to starting the computation.

   ANDREW Any easy way to do the above?

9 computations

Step by step computations in Table 1.

Name	Description	Type	Code
$Xbeta$	$X\beta$	$(N \times M) \times (M \times 1)$ $= N \times 1$	$Xbeta = \text{mv\_mul}(X, beta)$
p		$N \times 1$	$p = \text{logit}(t1) = e^{t1}/(1 + e^{t1})$
$ysubp$	$y - p$	$N \times 1$	$t_2 = \text{vvsub}(y, p)$
w		$N \times 1$	$w = \text{logit2}(t1) = e^{t1}/(1 + e^{t1})^2$
b	$X^T(y - p)$	$(M \times N) \times (N \times 1)$ $= M \times 1$	$\forall j_{j=1}^{j=M} b_j =$ $\text{sum}(\text{vvmul}(X_i, ysubp))$
A	$X^T W X$	$(M \times N) \times (N \times N) \times (N \times M)$ $= (M \times M)$	$\forall_{j=1}^{j=M} \forall_{k=j}^{k=M} A_{j,k} =$ $\text{sumprod2}(X_j, w, X_k)$

Table 1: Listing of individual steps and intermediate values

10 Details

10.1 Notes

1. The calculation of A is simplified by the fact that the off-diagonal elements of w are 0 and that it is a symmetric matrix. See last row of Table 1

10.2 Clarifications needed

1. Initial guess for β

11 Putting it all together

The Q code will look like the following.

```
t1 = mvmul(X, beta)
p = logit(t1)
w = logit2(t1)
t2 = vvsub(y, p)
for j in 1 to M do
  b[j] = sumprod(Xj, t2)
end
for j in 1 to M do
  for k in j to M do
    A[j][k] = sumprod2(Xj, w, Xk)
  end
end
```

Name	Input Type	Output Type	Return Value
logit	Vector x	Vector y	$y = \frac{e^x}{1+1^x}$
logit2	Vector x	Vector y	$y = \frac{e^x}{(1+1^x)^2}$
vvadd	Vector x, Vector y	Vector z	$z = x + y$
vvsub	Vector x, Vector y	Vector z	$z = x - y$
vvmul	Vector x, Vector y	Vector z	$z = x \times y$
vvdiv	Vector x, Vector y	Vector z	$z = x/y$
vsmul	Vector x, Scalar y	Vector z	$\forall i: \ z_i = x_\times y$
sumprod	Vector x, Vector y	Scalar z	$z = \sum_i (x_i \times y_i)$
sumprod2	Vector x, Vector y, Vector z	Scalar w	$w = \sum_i (x_i \times y_i \times z_i)$

Table 2: Necessary Operators