Binary Regression and non-linear optimization with R

Murshid Saqlain (msq@du.se)

March 6, 2019

Goals

- Learn how to optimize a non-linear function in R.

- Learn binary regression.
  - Learn non-linear maximization of a likelihood function.
  - Learn how to maximize the likelihood function of a binary regression model.

- Formulas will be provided. You will mostly copy and paste and learn how to use the formulas given to you.

Binary Response Models

- Y is a binary response variable , i.e. has two outcomes {0,1}.
- The model is a regression model but also a 'probability model'

$$E(Y|X) = 0Pr(Y = 0|X) + 1Pr(Y = 1|X) =$$

$$Pr(Y = 1|X)$$

- X is a 1 x (K + 1) vector of explanatory variables and a constant (intercept).
- β is a (K + 1) x 1 vector of coefficients.

Example: Linear Binary response models

- $E(Y|X) = Pr(Y = 1|X) = \beta_0 + \beta_1 X_1 = X\beta$

where e.g.

$$Y = \begin{cases} 1 & \text{if diabetes} \\ 0 & \text{otherwise} \end{cases}$$

and

$$X = [1; BMI]$$

- The linear binary response model may produce predictions of probabilities $Pr(Y = 1|X)$ outside the zero-one interval (draw)

Non-linear Binary response models with linear index

One way to assure predicted probabilities are in the correct zero-one interval:

$$\Pr(Y = 1|X) = F(X\beta)$$

where F(.) is a distribution function (cdf) of some assumed distribution and $X\beta$ is the 'linear index'.

Logit and Probit

- The two most common binary regression models are the logit and the probit
  - the logit has the following cdf

$$F(X\beta) = \frac{e^{X\beta}}{1 + e^{X\beta}}$$

  - and pdf

$$f(X\beta) = \frac{e^{X\beta}}{(1 + e^{X\beta})^2}$$

Logit and Probit

- The probit is based on the standard normal distribution which doesn't have closed forms for the cdf and the pdf. However, with R you can obtain values from the following two functions:
  - For $F(X\beta)$ you use pnorm($X\beta$)
  - For $f(X\beta)$ you use dnorm($X\beta$)

Marginal Effects and Interpreting β

- Since the logit model is not linear the marginal effects are not . The marginal effects for these type of models are:

$$\frac{\partial Pr(Y=1|X)}{\partial X} = \frac{dF(X\beta)}{d(X\beta)}\beta = f(X\beta)\beta.$$

- Which is equal to

$$\frac{e^{X\beta}}{(1+e^{X\beta})^2}\beta$$

Marginal Effects and Interpreting β

for the logit model where the pdf is:

$$f(X\beta) = \frac{e^{X\beta}}{(1 + e^{X\beta})^2}$$

- Nevertheless the sign of each β_k tells us
- $\beta_k > 0$ implies positive effect on $\Pr(Y = 1|X)$
- $\beta_k < 0$ implies negative effect on $\Pr(Y = 1|X)$

Estimating by Maximum Likelihood (ML)

- Assume (Y_i ;X_i); i = 1,…..,n are independently sampled
- The joint (conditional) density of the data is

$$f(Y_1, \ldots, Y_n|X_1, \ldots, X_n, \beta) = \prod_{i=1}^{n} f(Y_i|X_i, \beta).$$

- which is the likelihood function

$$L(\beta|(Y_i, X_i),\ i = 1, \ldots, n) = \prod_{i=1}^{n} f(Y_i|X_i, \beta)$$

Estimating by Maximum Likelihood (ML)

- The intuition behind maximizing the likelihood is that if for two coefficient vectors $L(\beta_1) > L(\beta_2)$ then it is more `likely' to obtain the present data if $\beta = \beta_1$ compared with $\beta = \beta_{2}$.

The Likelihood for binary response models

- It is simpler to work with the log-likelihood

$$logL = \mathcal{L}(\beta) = \sum_{i=1}^{n} Y_i logF(X_i\beta) + (1 - Y_i)log(1 - F(X_i\beta)) =$$

$$\sum_{i=1}^{n} \ell_i \qquad (1)$$

Iterative optimization

- There are usually not an analytical solution to $\max_{\beta} L(\beta)$
- The Newton-Raphson numerical optimization works well for ML functions of binary regression models
- But first some necessary matrix concepts

Gradient Vector

- The gradient including all partial rst order derivatives of a multivariate function $f(z_1, z_2, \ldots., z_q) = f(z)$:

$$g = \begin{bmatrix} \frac{\partial f(z)}{\partial z_1} \\ \frac{\partial f(z)}{\partial z_2} \\ \vdots \\ \frac{\partial f(z)}{\partial z_q} \end{bmatrix} = \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_q \end{bmatrix}$$

- The first order conditions we need for our maximization is g = **0.**

Gradient for binary response models

- The gradient of the log-likelihood of binary response models given in (1) is

$$\frac{\partial \mathcal{L}(\beta)}{\partial \beta} = \sum_{i=1}^{n} \frac{\partial \ell_i}{\partial F_i} \frac{\partial F_i}{\partial X_i \beta} \frac{\partial X_i \beta}{\partial \beta} = \tag{2}$$

$$\sum_{i=1}^{n} \left[\frac{Y_i}{F_i} - \frac{1 - Y_i}{1 - F_i} \right] f_i X_i' = \sum_{i=1}^{n} \frac{Y_i - F_i}{F_i(1 - F_i)} f_i X_i'$$

- where $F_i = F(X_i\beta)$ and $f_i = f(X_i\beta)$. Occasionally I will use the following notation

$$g = \sum_{i=1}^{n} g_i$$

where $g_i = \frac{Y_i - F_i}{F_i(1-F_i)} f_i X_i'$

Gradient for the logit

The gradient for the logit model

$$\sum_{i=1}^{n}(Y_i - F(X_i\beta))X_i'$$

where $F(X_i\beta) = \frac{e^{X_i\beta}}{1+e^{X_i\beta}} = F_i$. Straightforward to show if one first shows that: $f_i = F_i' = \frac{e^{X\beta}}{(1+e^{X\beta})}\left(1 - \frac{e^{X\beta}}{(1+e^{X\beta})}\right) = F_i(1 - F_i)$.

Hessian Matrix

- The Hessian is a matrix including all partial second order derivatives of a multivariate function $f(z_1, z_2, \ldots, z_q) = f(z)$:

$$H = \frac{\partial^2 f(z)}{\partial z \partial z'} = \frac{\partial g}{\partial z'} = \begin{bmatrix} \frac{\partial f_1}{\partial z'} \\ \frac{\partial f_2}{\partial z'} \\ \vdots \\ \frac{\partial f_q}{\partial z'} \end{bmatrix} = \begin{bmatrix} f_{11} & f_{12} & \cdots & f_{1q} \\ f_{21} & f_{22} & \cdots & f_{2q} \\ \vdots & & \ddots & \vdots \\ f_{q1} & & \cdots & f_{qq} \end{bmatrix}$$

- If the function is C^2 'twice continuously differentiable' the hessian is symmetric

$$H = \frac{\partial^2 f(z)}{\partial z \partial z'} = \begin{bmatrix} f_{11} & f_{12} & \cdots & f_{1q} \\ f_{12} & f_{22} & \cdots & f_{2q} \\ \vdots & & \ddots & \vdots \\ f_{1q} & & \cdots & f_{qq} \end{bmatrix}$$

Hessian for binary response models

$$H = \frac{\partial^2 \mathcal{L}}{\partial \beta \partial \beta'} = \sum_{i=1}^{n} \left[\frac{Y_i - F_i}{F_i(1 - F_i)} f_i' - \frac{(Y_i - F_i)^2}{F_i^2(1 - F_i)^2} f_i^2 \right] X_i' X_i$$

where $f_i' = \frac{df(X_i\beta)}{dX_i\beta}$. The Hessian for the logit model simplifies to

$$H = -\sum_i f_i X_i' X_i.$$

Multivariate Taylor approximation

- The function in a multivariate point **b** (vector), f (**b**), can be approximated given information of *f*(.), *g* and *H* in the multivariate point **a** by the following Taylor approximation:

$$f(\mathbf{b}) \cong f(\mathbf{a}) + g_{\mathbf{a}}'(\mathbf{b}-\mathbf{a}) + \frac{1}{2}(\mathbf{b}-\mathbf{a})' H_{\mathbf{a}}(\mathbf{b}-\mathbf{a})$$

Approximate the Likelihood

- If we want to maximize a Likelihood but we cannot and an analytical solution we can use the Taylor approximation

$$\mathcal{L}(\beta) \cong \mathcal{L}(\hat{\beta}) + g'_{\hat{\beta}}(\beta - \hat{\beta}) + \frac{1}{2}(\beta - \hat{\beta})' H_{\hat{\beta}}(\beta - \hat{\beta})$$

Approximate the Likelihood

In the maximum $\frac{\partial \mathcal{L}(\beta^*)}{\partial \beta^*} = \mathbf{0}$. Where β^* is the beta-vector that maximizes the likelihood function. If we instead differentiate the approximation w.r.t β we get

$$g_{\hat{\beta}} + \frac{1}{2}(2H_{\hat{\beta}}\beta^* - 2H_{\hat{\beta}}\hat{\beta}) = \mathbf{0}$$

rearrange

$$H_{\hat{\beta}}\beta^* = H_{\hat{\beta}}\hat{\beta} - g_{\hat{\beta}}$$

Multiply by the inverse of the hessian

$$\beta^* = \hat{\beta} - H_{\hat{\beta}}^{-1} g_{\hat{\beta}}$$

The Newton-Raphson iterations

If $\hat{\beta}$ is close to β^*, the taylor approximation will work well. However, we don't know β^* so $\hat{\beta}$ may be far away. With Newton-Raphson iterations we will get closer and closer to the β^* that maximizes $\mathcal{L}(\beta^*)$. One updates the estimator according to

$$\hat{\beta}_{(j+1)} = \hat{\beta}_{(j)} - H_j^{-1} g_j$$

The Newton-Raphson iterations

1. Select initial values for the coefficient vector $\hat{\beta}_{(1)}$ (this is a vector, not the slope-coefficient β_1)
2. Compute $\hat{\beta}_{(2)} = \hat{\beta}_{(1)} - H_1^{-1} g_1$
3. Start at 2. again and use $\hat{\beta}_{(2)}$ to compute the right-hand side and obtain $\hat{\beta}_{(3)}$
4. Continue these iterations until e.g. $|g_j' \mathbf{1}| < 0.00000001$

Remark: the notation $\hat{\beta}_{(j)}$ doesn't mean the j^{th} element of the vector $\hat{\beta}$ here. $\hat{\beta}_{(j)}$ means the coefficient vector at the j^{th} iteration.

Example: The logit with the Newton-Raphson algorithm

- Make iterations with while-loop in R. Use the script “Logit.R” to estimate the binary regression for labor-force participation.

```
# Analyze Women's Labor-Force Participation
library(car); data(Mroz);summary(Mroz);attach(Mroz)
#Make data in the form accepted by the function logit
n<-length(lfp);Y<-ifelse(lfp=="yes", 1, 0)
X<-cbind(1,age,k5,k618,inc,ifelse(wc=="yes",1,0),
ifelse(hc=="yes",1,0))
#Call the function
source(“~/Logit.R")
logit(X,Y,X.names=c("Constant","age","k5","k618",
"inc","wc","hc"))
detach(Mroz)
```

Example: The logit using function optim in R

- Example Make iterations with the optim function in R. Use the script Logit optim.R

```
#Call the function
source(“~/Logit_optim.R")
logit(X,Y,X.names=c("Constant","age","k5","k618",
"inc","wc","hc"))
```

Binary regression with the glm() function in R

- The logit regression is specified as follows

glm(y~x, family = binomial("logit"), data)

- binomial() the family name of the distribution of the response
- "logit" is the `link' function
- If “probit" is used instead a probit regression is performed (based on the standard normal dist. N(0; 1) instead of the logit), specified as

glm(y~x, family = binomial("probit"), data)

- Probit and logit are the two most common binary regression models.

Example: glm()

```
# Analyze Women’s Labor-Force Participation
reg<-glm(lfp~age+k5+k618+inc+wc+hc,
family=binomial("logit"),data=Mroz)
summary(reg)
```

Hypothesis testing for β

The statistic $T_k = \frac{(\ddot{\beta}_k - \beta_{k,0})}{\sqrt{\hat{\sigma}_k^2}}$ can be used to test

$$H_0 : \beta_k = \beta_{k,0}$$

for the k^{th} beta, where $\hat{\sigma}_k^2$ is the $(k+1)^{th}$ element of the diagonal of $\hat{\mathcal{I}}^{-1}$

- $\mathcal{I}$ is called the 'Fisher information' matrix
  - And it has many equivalent forms (under iid sampling)

$$\mathcal{I} = -E(H_\beta) = E(g_\beta g_\beta') = nE(g_i g_i') \text{ where } g_i = \frac{\partial logf(Y_i|X_i,\beta)}{\partial \beta}$$

Joint testing: Likelihood ratio test

- To test hypothesis like

$$H_0 : \beta_1 = 0, \beta_2 = 0, \ldots, \beta_m = 0 \quad (m \leq K)$$

- one can use

$$LR_m = -2log\left(\frac{L(0, \ldots, 0, \tilde{\beta}_{m+1}, \ldots, \tilde{\beta}_K)}{L(\hat{\beta}_0, \ldots, \hat{\beta}_K)}\right) \xrightarrow{d} \chi^2_m$$

- where the β~ coefficients are obtained by

$$\max_{\beta_{m+1}, \ldots, \beta_K} \mathcal{L}(0, \ldots, 0, \tilde{\beta}_{m+1}, \ldots, \tilde{\beta}_K)$$

Example: LR Test

```
#Model under H_0: b1=b2=b3=b4=b5=b6=0
reg0<-glm(lfp~1,family=binomial("logit"), data=Mroz)
#Unrestricted Model
reg<-glm(lfp~age+k5+k618+inc+wc+hc,
family=binomial("logit"),data=Mroz)

#Likelihood ratio statistic value
LR<--2*as.numeric(logLik(reg0)-logLik(reg))
#P-value with six restrictions
pchisq(LR, df=6,lower.tail = FALSE)
#LR Test with the anova function in R
anova(reg0,reg,test='Chisq')
LR
```