Logit Models

Bill

2019-12-23

Contents

Chapter 1

Basics

1.1 Logit

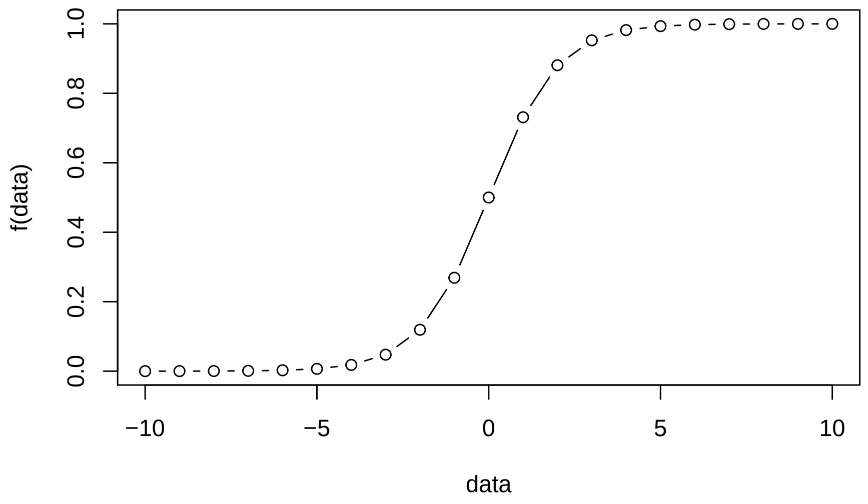
$$f(x) = log(\frac{p(y=1)}{1-p(y=1)})$$

The basic idea of logistic regression:

$$p(y=1) = \frac{1}{1+e^{-(\beta_0+\beta_1x_1+...+\beta_nx_n)}} = \frac{e^{\beta_0+\beta_1x_1+...+\beta_nx_n}}{1+e^{\beta_0+\beta_1x_1+...+\beta_nx_n}}$$

Thus, $e^{\beta_0+\beta_1x_1+...+\beta_nx_n}$ can be from $-\infty$ to $+\infty$, and $p(y=1)$ will be always within the range of $(0,1)$.

```r
f<-function(x){exp(x)/(1+exp(x))}
data<-seq(-10,10,1)
plot(data,f(data),type = "b")
```



We can also write the function into another format as follows:

$$log\frac{p(y=1)}{1-p(y=1)} = \beta_0 + \beta_1 x_1 + ... + \beta_n x_n$$

1.2 Probit

$$\beta_0 + \beta_1 x_1 + ... + \beta_n x_n = \Phi^{-1}(p)$$

Thus,

$$\Phi(\beta_0 + \beta_1 x_1 + ... + \beta_n x_n) = p(y=1)$$

Chapter 2

MLE

The probablity of $y = 1$ is as follows:

$$p = p(y = 1) = \frac{1}{1 + e^{-(\beta_0+\beta_1 x_1+...+\beta_n x_n)}} = \frac{e^{\beta_0+\beta_1 x_1+...+\beta_n x_n}}{1 + e^{\beta_0+\beta_1 x_1+...+\beta_n x_n}}$$

Thus, the likelihood function is as follows:

$$L = \prod p^{y_i}(1-p)^{1-y_i} = \prod (\frac{1}{1 + e^{-(\beta_0+\beta_1 x_1+...+\beta_n x_n)}})^{y_i} (\frac{1}{1 + e^{\beta_0+\beta_1 x_1+...+\beta_n x_n}})^{1-y_i}$$

$$= \prod (1 + e^{-(\beta_0+\beta_1 x_1+...+\beta_n x_n)})^{-y_i} (1 + e^{\beta_0+\beta_1 x_1+...+\beta_n x_n})^{-(1-y_i)}$$

Thus, the log-likelihood is as follows:

$$logL = \sum (-y_i \cdot log(1+e^{-(\beta_0+\beta_1 x_1+...+\beta_n x_n)}) - (1-y_i) \cdot log(1+e^{\beta_0+\beta_1 x_1+...+\beta_n x_n}))$$

Typically, optimisers minimize a function, so we use negative log-likelihood as minimising that is equivalent to maximising the log-likelihood or the likelihood itself.

```r
#Source of R code: https://www.r-bloggers.com/logistic-regression/

mle.logreg = function(fmla, data)
{
  # Define the negative log likelihood function
  logl <- function(theta,x,y){
    y <- y
    x <- as.matrix(x)
    beta <- theta[1:ncol(x)]

    # Use the log-likelihood of the Bernouilli distribution, where p is
    # defined as the logistic transformation of a linear combination
    # of predictors, according to logit(p)=(x%*%beta)
    loglik <- sum(-y*log(1 + exp(-(x%*%beta))) - (1-y)*log(1 + exp(x%*%beta)))
    return(-loglik)
  }

  # Prepare the data
  outcome = rownames(attr(terms(fmla),"factors"))[1]
  dfrTmp = model.frame(data)
  x = as.matrix(model.matrix(fmla, data=dfrTmp))
  y = as.numeric(as.matrix(data[,match(outcome,colnames(data))]))

  # Define initial values for the parameters
  theta.start = rep(0,(dim(x)[2]))
  names(theta.start) = colnames(x)

  # Calculate the maximum likelihood
  mle = optim(theta.start,logl,x=x,y=y, method = 'BFGS', hessian=T)
  out = list(beta=mle$par,vcov=solve(mle$hessian),ll=2*mle$value)
}
```

```r
mydata = read.csv(url('https://stats.idre.ucla.edu/stat/data/binary.csv'))
mylogit1 = glm(admit~gre+gpa+as.factor(rank), family=binomial, data=mydata)

mydata$rank = factor(mydata$rank) #Treat rank as a categorical variable
fmla = as.formula("admit~gre+gpa+rank") #Create model formula
mylogit2 = mle.logreg(fmla, mydata) #Estimate coefficients


 print(cbind(coef(mylogit1), mylogit2$beta))
```

```
##                          [,1]         [,2]
## (Intercept)      -3.989979073 -3.772676422
## gre               0.002264426  0.001375522
## gpa               0.804037549  0.898201239
## as.factor(rank)2 -0.675442928 -0.675543009
## as.factor(rank)3 -1.340203916 -1.356554831
## as.factor(rank)4 -1.551463677 -1.563396035
```

Chapter 3

Twitter Example

The following is part of my course project for Stat 536. It aims to replicate part of the findings from Barbera (2015) Birds of the Same Feather Tweet Together: Bayesian Ideal Point Estimation Using Twitter Data. Political Analysis 23 (1). Note that, the following model is much simpler than that in the original paper.

3.1 Model

Suppose that a Twitter user is presented with a choice between following or not following another target $j \in \{1, ..., m\}$. Let $y_j = 1$ if the user decides to follow j, and $y_j = 0$ otherwise.

$$y_j = \begin{cases} 1 & Following \\ 0 & NotFollowing \end{cases}$$

$$p(y_j = 1|\theta) = \frac{exp(-\theta_0|\theta_1 - x_j|^2)}{1 + exp(-\theta_0|\theta_1 - x_j|^2)}$$

We additionally know the priors of θ.

$$\theta_i \sim N(0, 10^2)(i = 0, 1)$$

The likelihood function is as follows.

$$L(Y|\theta) = \prod_{j=1}^{m}(\frac{exp(-\theta_0|\theta_1 - x_j|^2)}{1 + exp(-\theta_0|\theta_1 - x_j|^2)})^{y_j}(1 - \frac{exp(-\theta_0|\theta_1 - x_j|^2)}{1 + exp(-\theta_0|\theta_1 - x_j|^2)})^{(1-y_j)}$$

Thus, the posterior is as follows.

$$L(Y|\theta) \cdot N(\theta_0|0,10) \cdot N(\theta_1|0,10)$$

$$\propto \prod_{j=1}^{m}(\frac{exp(-\theta_0|\theta_1 - x_j|^2)}{1+exp(-\theta_0|\theta_1 - x_j|^2)})^{y_j}(1-\frac{exp(-\theta_0|\theta_1 - x_j|^2)}{1+exp(-\theta_0|\theta_1 - x_j|^2)})^{(1-y_j)} \cdot exp(-\frac{1}{2}(\frac{\theta_0}{10})^2) \cdot exp(-\frac{1}{2}(\frac{\theta_1}{10})^2)$$

3.2 Simulating Data of Senators on Twitter

Assume that we have 100 senators, 50 Democrats and 50 Republicans, who we know their ideology. Assume that Democrats have negative ideology scores to indicate that they are more liberal, whereas Republicans have positive scores to indicate that they are more conservative. The following is data simulation for senators.

```
# Republicans are more conservative, and they have positive numbers.
Republicans<-c()
Republicans<-rnorm(50,1,0.5)
No_Republicans<-rep(1:50,1)
Part_1<-cbind(No_Republicans,Republicans)

# Democrats are more liberal, and they have negative numbers.
Democrats<-c()
Democrats<-rnorm(50,-1,0.5)
No_Democrats<-rep(51:100,1)
Part_2<-cbind(No_Democrats,Democrats)
Data_Elites<-rbind(Part_1,Part_2)
Data_Elites<-as.data.frame(Data_Elites)
colnames(Data_Elites) <- c("Elite_No","Elite_ideology")

head(Data_Elites)
```

```
##   Elite_No Elite_ideology
## 1        1       1.337188
## 2        2       1.470603
## 3        3       0.538886
## 4        4       1.611851
## 5        5       1.490693
## 6        6       1.012758
```

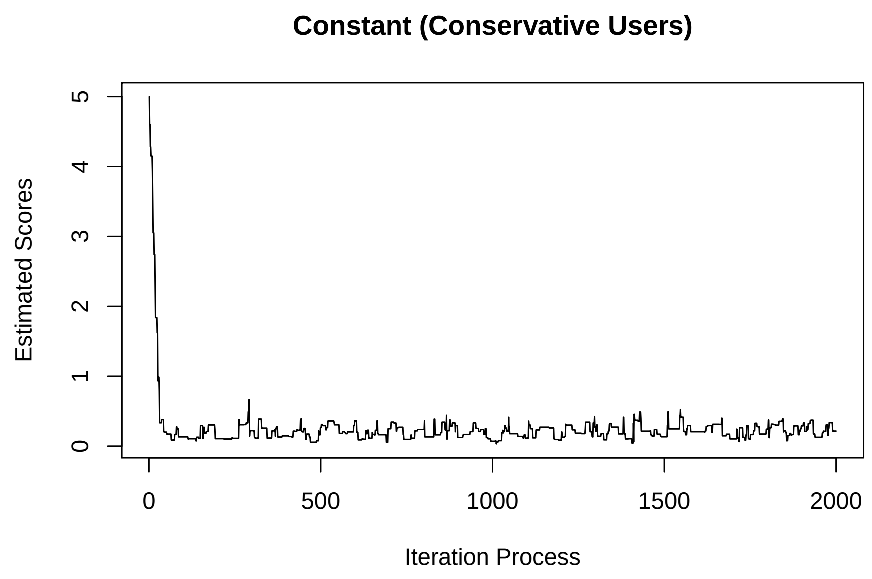
3.3 Simulating Data of Conservative Users on Twitter and Model Testing

Assume that we observe one Twitter user, who is more conservative. To simulate Twitter following data for this user, I assign this user to follow more Republican senators. Thus, if the Metropolis Hastings algorithm works as intended, we would expect to see a positive estimated value for their ideology. Importantly, as we can see in the histogram below, the estimated value indeed is positive, providing preliminary evidence for the statistical model and the algorithm. In addition, for the acceptance rate, we can see that the constant has a lower number than ideology, since we only accept a constant when it is positive.

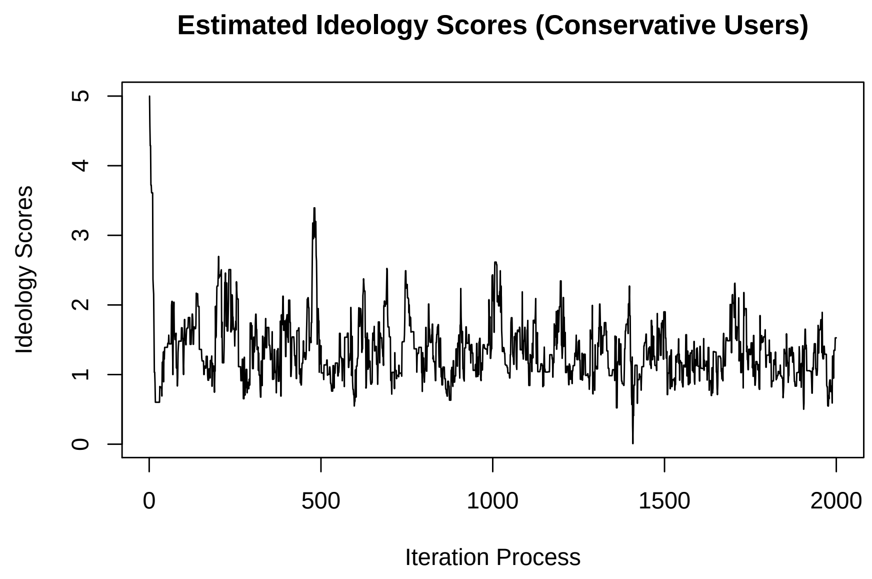
```
#This user approximately follows 45 Republican Senators and 10 Democrat Senators.
Data_user<-as.data.frame(matrix(c(ifelse(runif(50)<.1,0,1),ifelse(runif(50)<.8,0,1))), 100, 1)
colnames(Data_user)<-c("R_User")
Data_combined<-cbind(Data_Elites,Data_user)

X_data<-Data_combined$Elite_ideology
Y_data<-Data_combined$R_User

fit_C<-Bayes_logit(Y_data,X_data)
fit_C$acceptance_rate
```

```
## [1] 0.1580790 0.5067534
```

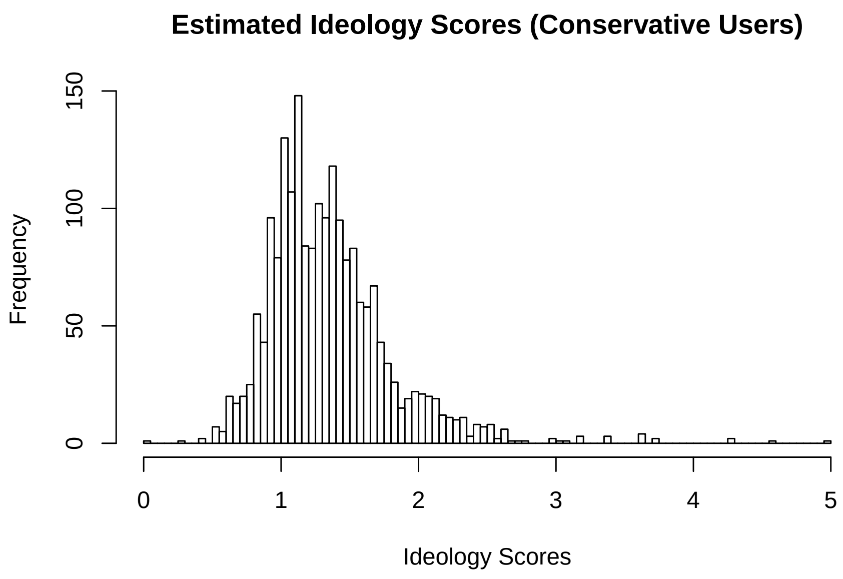
```
plot(fit_C$theta[,1],main="Constant (Conservative Users)",
     xlab="Iteration Process",ylab="Estimated Scores",type="l")
```



```r
plot(fit_C$theta[,2],main="Estimated Ideology Scores (Conservative Users)",
     xlab="Iteration Process",ylab="Ideology Scores",type="l")
```



```
hist(fit_C$theta[,2],main="Estimated Ideology Scores (Conservative Users)",
     xlab="Ideology Scores",breaks = 100)
```



3.4 Simulating Data of Liberal Users on Twitter and Model Testing

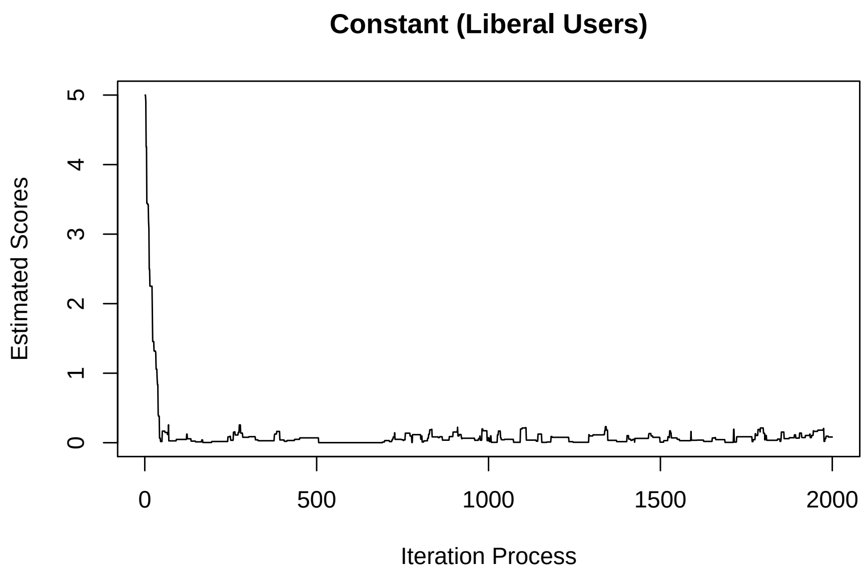
To further verify the Metropolis Hastings algorithm, I plan to test the opposite estimate. Specifically, assume that we observe another user, who is more liberal. To simulate Twitter following data for this user, I assign this user to follow more Democrat senators. In this case, we would expect to see a negative value for their estimated ideology. As we can see in the histogram shown below, as expected, the estimated value is negative, providing convergent evidence for the model and the algorithm.

```
#This user approximately follows 10 Republican Senators and 45 Democrat Senators.
Data_user<-as.data.frame(matrix(c(ifelse(runif(50)<.8,0,1),ifelse(runif(50)<.1,0,1))), 100, 1)
colnames(Data_user)<-c("L_User")
Data_combined<-cbind(Data_Elites,Data_user)

X_data<-Data_combined$Elite_ideology
Y_data<-Data_combined$L_User
```

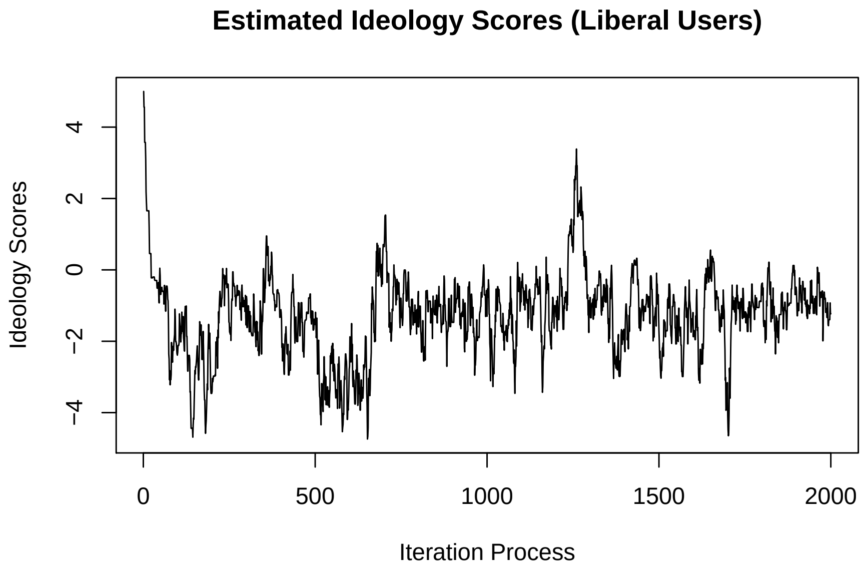
```
fit_L<-Bayes_logit(Y_data,X_data)
fit_L$acceptance_rate
```

```
## [1] 0.09904952 0.80140070
```

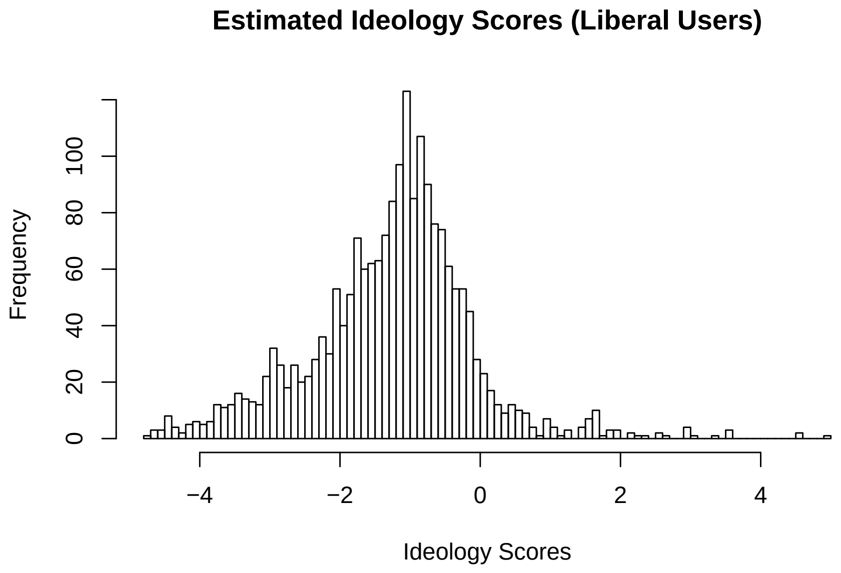
```
plot(fit_L$theta[,1],main="Constant (Liberal Users)",
     xlab="Iteration Process",ylab="Estimated Scores",type="l")
```



```
plot(fit_L$theta[,2],main="Estimated Ideology Scores (Liberal Users)",
     xlab="Iteration Process",ylab="Ideology Scores",type="l")
```



```
hist(fit_L$theta[,2],main="Estimated Ideology Scores (Liberal Users)",
     xlab="Ideology Scores",breaks = 100)
```



Chapter 4

Linear Mixed Models

The following code generates 4 numbers ($N(0, 10)$) for 4 groups. Then, replicate it within each group.That is, in the end, there are 8 observations.

Note that, in the following code, there are no "independent variables". Both the linear model and mixed model are actually just trying to calculate the mean.

```r
# The original R code: http://www.john-ros.com/Rcourse/lme.html
# I did some minor changes and added some additional comments.

set.seed(123)
n.groups <- 4 # number of groups
n.repeats <- 2 # samples per group
#Generating index for observations belong to the same group
groups <- as.factor(rep(1:n.groups, each=n.repeats))
n <- length(groups)
#Generating 4 random numbers, assuming normal distribution
z0 <- rnorm(n.groups, 0, 10)
(z <- z0[as.numeric(groups)]) # generate and inspect random group effects
```

```
## [1] -5.6047565 -5.6047565 -2.3017749 -2.3017749 15.5870831 15.5870831  0.7050839
## [8]  0.7050839
```

```r
epsilon <- rnorm(n,0,1) # generate measurement error
beta0 <- 2 # this is the actual parameter of interest! The global mean.
y <- beta0 + z + epsilon # sample from an LMM

# fit a linear model assuming independence
# i.e., assume that there is no "group things".
lm.5 <- lm(y~1)
```

```r
# fit a mixed-model that deals with the group dependence
#install.packages("lme4")
library(lme4)
lme.5 <- lmer(y~1|groups)
```

Chapter 5

Generalized Linear Mixed Models

The following is the note from Charle E. McCulloch's “Maximum likelihood algorithems for Generalized Linear Mixed Models”