Group 3 R Project 2

John Rollman, Mathan Brewer, Nataliya Peshekhodko

2020-March-16

Abstract

The main goal of this project involves creating an MCMC sampler for a logistic regression model. To be continued . . .

```r
library (dplyr)
library (MASS)
```

1. Functions for prior, likelihood and posterior

Based on the assumption that β_0 and β_1 each is normally distributed $\sim N(0, 10^2)$, joint distribution could be found as

$$f(\beta_0, \beta_1) = \frac{1}{2\pi\sigma^2} e^{-\frac{1}{2\sigma^2}(\beta_0^2 + \beta_1^2)}$$

.

Function for prior

```r
prior <- function (beta0, beta1) {
  sigma_sq = 100;
  return (  1/(2*sigma_sq^(2)*pi)*exp(-(beta0^(2) + beta1^(2))/(2*sigma_sq^(2))) )
}
```

Model

$$p(x_i) = \frac{1}{1 + e^{-\beta_0 - \beta_1 x_i}}$$

The likelihood for the logistic regression model

$$L(p) \propto \prod_{i=1}^{n} p(x_i)^{y_i} (1 - p(x_i))^{1-y_i}$$

Function for likelihood

```r
likelihood <- function (y, beta0, beta1, x) {
  return (prod((1/(1+exp (-beta0 - beta1*x)))^(y)*(1-1/(1+exp (-beta0 - beta1*x)))^(1-y)))
}
```

Posterior distribution

$$f_{\beta_0,\beta_1|Y_1,\ldots,Y_n}(\beta_0, \beta_1|y_1, \ldots y_n, x_1, \ldots x_n) \propto L(p) * f(\beta_0, \beta_1)$$

Function for posterior

```
posterior <- function (y, beta0, beta1, x) {
  return (prior(beta0, beta1)*likelihood(y, beta0, beta1, x))
}
```

Create function for

```
thesampler <- function (y, x, n, niter, beta0start, beta1start, beta0proposalsd, beta1proposalsd) {

  beta0 = rep (0, niter)
  beta1 = rep (0, niter)

  beta0[1] = beta0start
  beta1[1] = beta1start

  for (i in 2:niter) {
    currentbeta0 = beta0[i-1]
    currentbeta1 = beta1[i-1]

    newbeta0 = currentbeta0 + rnorm (1, 0, beta0proposalsd)
    r = posterior(y,newbeta0, currentbeta1, x)/posterior (y, currentbeta0, currentbeta1, x)

    if (runif (1) < r){
      beta0[i] = newbeta0}
    else{
      beta0[i] = currentbeta0}

    newbeta1 = currentbeta1 + rnorm (1, 0, beta1proposalsd)
    r = posterior(y,currentbeta0, newbeta1, x)/posterior (y, currentbeta0, currentbeta1, x)

    if (runif (1) < r){
      beta1[i] = newbeta1}
    else{
      beta1[i] = currentbeta1}
  }

  betas <- data.frame(beta0, beta1)
  names (betas) <- c ("beta0", "beta1")
  return (betas)
}
```

Apply to NFL data

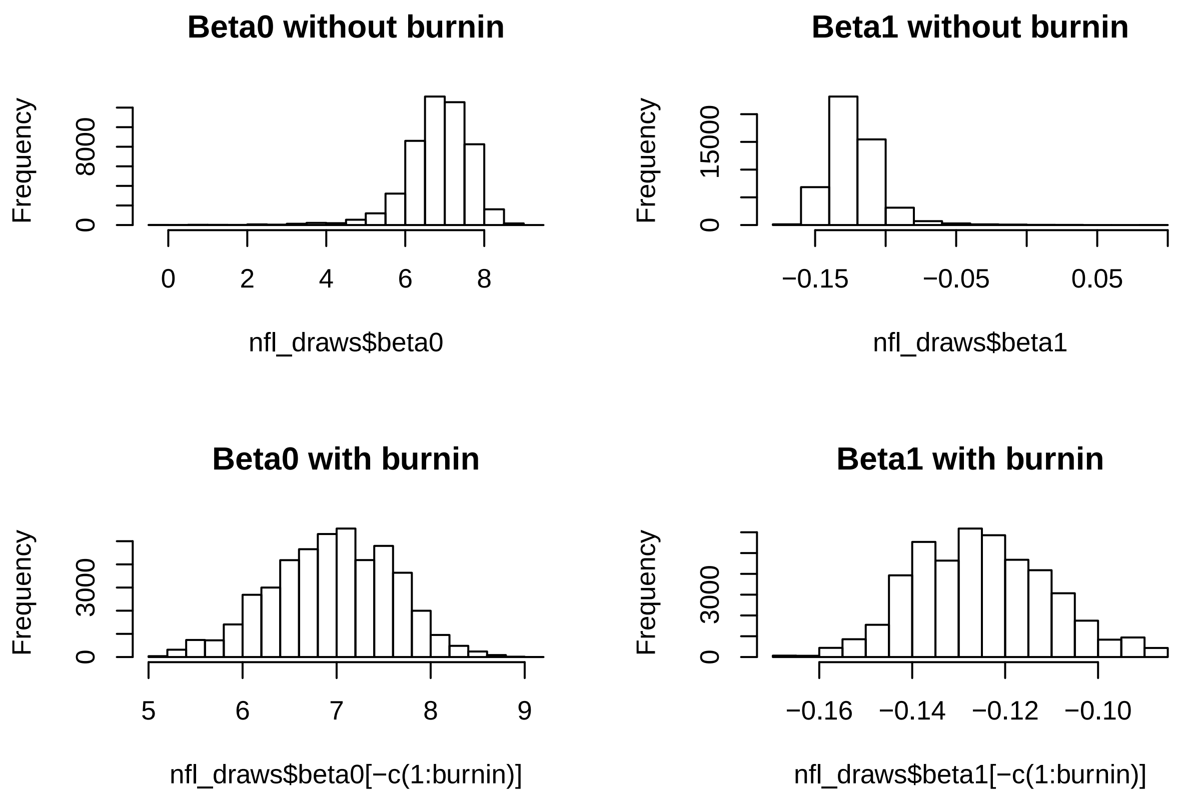
```
nfl<-read.csv(file = "nfl2008_fga.csv")
nfl <- dplyr::select (nfl, distance, GOOD)


n=nrow(nfl)
nfl_draws <- thesampler (nfl$GOOD, nfl$distance, n, 50000, 0, 0,  0.1, 0.1 )

burnin <- 5000

par(mfrow=c(2,2))
hist (nfl_draws$beta0, main = "Beta0 without burnin")
```

```r
hist (nfl_draws$beta1, main = "Beta1 without burnin")
hist (nfl_draws$beta0 [-c(1:burnin)], main = "Beta0 with burnin")
hist (nfl_draws$beta1 [-c(1:burnin)], main = "Beta1 with burnin")
```



```r
quantile (c (0.025, 0.975), x = nfl_draws$beta0 [-c(1:burnin)])
```

```
##     2.5%    97.5%
## 5.608141 8.119391
```

```r
quantile (c (0.025, 0.975), x = nfl_draws$beta1 [-c(1:burnin)])
```

```
##        2.5%       97.5%
## -0.15072798 -0.09355594
```

```r
fit <- glm (GOOD ~ distance, family = binomial (link = 'logit'), data = nfl)
summary (fit)
```

```
##
## Call:
## glm(formula = GOOD ~ distance, family = binomial(link = "logit"),
##     data = nfl)
##
## Deviance Residuals:
```

```
##     Min       1Q   Median       3Q      Max
## -2.9526   0.2039   0.3478   0.5826   1.2309
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  6.76271    0.54443  12.422   <2e-16 ***
## distance    -0.12084    0.01229  -9.836   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 817.72  on 1038  degrees of freedom
## Residual deviance: 686.93  on 1037  degrees of freedom
## AIC: 690.93
##
## Number of Fisher Scoring iterations: 6
```

```
confint (fit)
```

```
## Waiting for profiling to be done...
```

```
##                  2.5 %      97.5 %
## (Intercept)  5.7399740  7.87764350
## distance    -0.1457751 -0.09754001
```

```
#plot (fit)
```