Assignment #9

Yutae Lee

2022-11-17

Problem 9.1

```r
swim <- read.table(url('https://www2.stat.duke.edu/courses/Fall09/sta290/datasets/Hoffdata/swim.dat'))
```

a)

i)

As mentioned in the question, for the prior of competitive times, I will set it to 23.

And I am going to set the prior of the effect of the training by week as 0.

Thus I am going to set $\beta = [23, 0]^T$

Here I am going to run Gibbs Sampler for 10,000 times

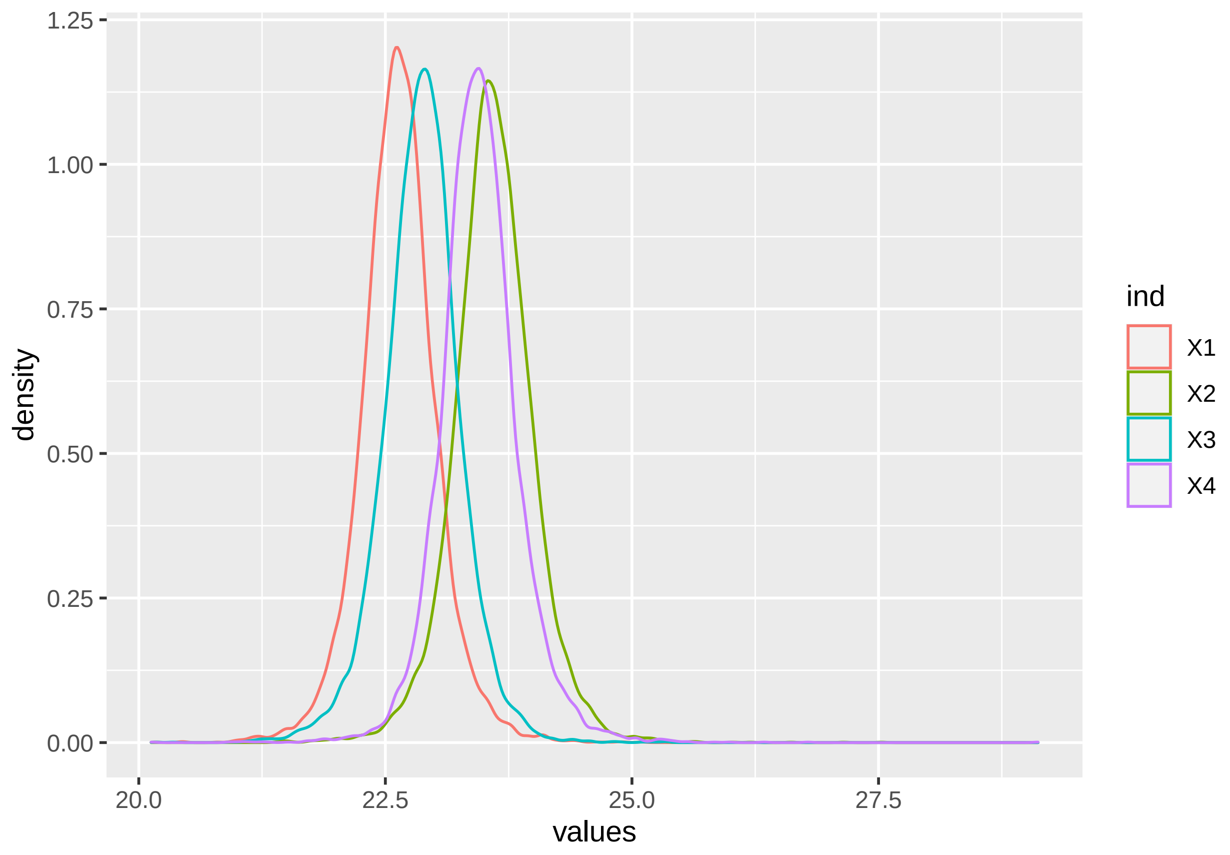
```r
library(MASS)
library(dplyr)
```

```
## Warning:    'dplyr'  R   4.2.2

##
##         : 'dplyr'

## The following object is masked from 'package:MASS':
##
##     select

## The following object is masked from 'package:reshape':
##
##     rename

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
set.seed(32)
trains = cbind(rep(1, 6), seq(1,11, by = 2))
# Number of Gibbs Iteration
Iterations = 10000
#Hyperparameters
beta_0 = c(23, 0)
sigma_0 = rbind(c(0.25, 0), c(0, 0.1))
v0 = 1
sigma2_0 = 0.25
# predictive distribution
predictive_distributions = apply(swim, MARGIN = 1, function(y) {

  # Store samples
  Beta = matrix(nrow = Iterations, ncol = length(beta_0))
  Sigma = numeric(Iterations)

  # prior values
  beta = c(23, 0)
  sigma_2 = 0.25

  # Gibbs sampling
  for (i in 1:Iterations) {
    #updating BETA
    # compute v and m
    v = solve(solve(sigma_0) + (t(trains) %*% trains) / sigma_2)
    m = v %*% (solve(sigma_0) %*% beta_0 + (t(trains) %*% y) / sigma_2)
    #sample beta <- multivariate normal(m,v)
    beta = mvrnorm(1, m, v)
    #Updating Sigma^2
    # Computing SSR
    SSR = (t(y) %*% y) - (2 * t(beta) %*% t(trains) %*% y) + (t(beta) %*% t(trains) %*% trains %*% beta
    # Sampling sigma_2
    sigma_2 = 1 / rgamma(1, (v0 + dim(trains)[1]) / 2, (v0 * sigma2_0 + SSR) / 2)
    Beta[i, ] = beta
    Sigma[i] = sigma_2
  }

  # posterior predictive distribution (ii) (2 weeks later)
  x_pred = c(1, 13)
  y_pred = rnorm(i, Beta %*% x_pred, sqrt(Sigma))

  y_pred
})
```

Below is the plot of the posterior predictive distributions:

```r
df = stack(data.frame(predictive_distributions))
ggplot(df, aes(x=values, color = ind)) +
  geom_density()
```



b)

```
mean(predictive_distributions[,1] < predictive_distributions[,2] & predictive_distributions[,1] < predi
```

```
## [1] 0.6508
```

```
mean(predictive_distributions[,2] < predictive_distributions[,1] & predictive_distributions[,2] < predi
```

```
## [1] 0.0188
```

```
mean(predictive_distributions[,3] < predictive_distributions[,2] & predictive_distributions[,3] < predi
```

```
## [1] 0.3014
```

```
mean(predictive_distributions[,4] < predictive_distributions[,2] & predictive_distributions[,4] < predi
```

```
## [1] 0.029
```

We can see that Swimmer 1 has 0.6508 probability of being the fastest.

Swimmer 2 with 0.0188

Swimmer 3 with 0.3014

Swimmer 4 with 0.029

Problem 9.2

```r
azdiabetes <- read.table(url('https://www2.stat.duke.edu/courses/Fall09/sta290/datasets/Hoffdata/azdiab
header.true <- function(df) {
  names(df) <- as.character(unlist(df[1,]))
  df[-1,]
}
azdiabetes <- header.true(azdiabetes)
azdiabetes <- azdiabetes[,1:7]
```

a)

```r
x <- data.matrix(subset(azdiabetes, select = -c(glu)))
vec <- rep(1,10000)
x <- cbind(vec,x)
```

```
## Warning in cbind(vec, x): number of rows of result is not a multiple of vector
## length (arg 1)
```

```r
y <- data.matrix(subset(azdiabetes, select = c(glu)))
n <- length(y[,1])
```

Let's first select our hyperparameters:

```r
g = n
v0 = 2
sigma2_0 = 1
```

```r
library(MCMCpack)
```

```
## Warning:    'MCMCpack'  R   4.2.2

##           : coda

## Warning:    'coda'  R   4.2.2

## ##
## ## Markov Chain Monte Carlo Package (MCMCpack)

## ## Copyright (C) 2003-2022 Andrew D. Martin, Kevin M. Quinn, and Jong Hee Park

## ##
## ## Support provided by the U.S. National Science Foundation

## ## (Grants SES-0350646 and SES-0350613)
## ##
```

```r
set.seed(32)
SSR <- t(y) %*% (diag(n) - (g/(g+1)) * x %*% solve(t(x)%*%x)%*%t(x)) %*% y
sigma_samples <- rinvgamma(n = 10000, shape = (v0 + n)/2, scale = (v0*sigma2_0 + SSR)/2)
```

```r
beta_samples <- c()
for (sig2 in sigma_samples){
  beta_samples <- c(beta_samples, mvrnorm(n = 1, mu = (g/(g+1)) * solve(t(x)%*%x) %*% t(x)%*%y, Sigma =
}
```

```r
intercept <- c()
npreg <- c()
  bp <- c()
  skin <- c()
  bmi <- c()
  ped <- c()
  age <- c()
for (i in 0:9999){
  intercept <- c(intercept, beta_samples[1 + 7*i])
  npreg <- c(npreg,beta_samples[2 + 7*i])
  bp <- c(bp,beta_samples[3 + 7*i])
  skin <- c(skin,beta_samples[4 + 7*i])
  bmi <- c(bmi,beta_samples[5 + 7*i])
  ped <- c(ped,beta_samples[6 + 7*i])
  age <- c(age,beta_samples[7 + 7*i])
}
```

95% confidence for the intercept:

```r
print(quantile(intercept, probs = c(0.025, 0.975)))
```

```
##     2.5%    97.5%
## 55.49918 87.46455
```

95% confidence for the npreg:

```r
print(quantile(npreg, probs = c(0.025, 0.975)))
```

```
##       2.5%      97.5%
## -0.5331523  0.8725240
```

95% confidence for the bp:

```r
print(quantile(bp, probs = c(0.025, 0.975)))
```

```
##       2.5%      97.5%
## -0.7650392  0.2598394
```

95% confidence for the skin:

```r
print(quantile(skin, probs = c(0.025, 0.975)))
```

```
##        2.5%       97.5%
## -0.96744307 -0.05340878
```

95% confidence for the bmi:

```r
print(quantile(bmi, probs = c(0.025, 0.975)))
```

```
##        2.5%       97.5%
## -0.07842562  0.09640096
```

95% confidence for the ped:

```r
print(quantile(ped, probs = c(0.025, 0.975)))
```

```
##        2.5%       97.5%
## -0.02337430  0.04086407
```

95% confidence for the age:

```r
print(quantile(age, probs = c(0.025, 0.975)))
```

```
##       2.5%      97.5%
## -0.5324544  0.2247858
```

For σ^2:

```r
print(quantile(sigma_samples, probs = c(0.025, 0.975)))
```

```
##     2.5%    97.5%
## 1574.051 1991.483
```

b)