Methods-for-Utility-Evaluation

Reese Guo

2/20/2020

```
library(ggplot2)
CEdata <- read.csv("CEdata.csv")
CEdata$LogIncome <- log(CEdata$Income)
CEdata$LogExpenditure <- log(CEdata$Expenditure)
```

```
require(runjags)
require(coda)

modelString <-"
model {
## sampling
for (i in 1:N){
y[i] ~ dnorm(beta0 + beta1*x[i], invsigma2)
}
## priors
beta0 ~ dnorm(mu0, g0)
beta1 ~ dnorm(mu1, g1)
invsigma2 ~ dgamma(a, b)
sigma <- sqrt(pow(invsigma2, -1))
}"

y <- as.vector(CEdata$LogIncome)
x <- as.vector(CEdata$LogExpenditure)
N <- length(y)

the_data <- list("y" = y, "x" = x, "N" = N,
                "mu0" = 0, "g0" = 0.0001,
                "mu1" = 0, "g1" = 0.0001,
                "a" = 1, "b" = 1)

initsfunction <- function(chain){
.RNG.seed <- c(1,2)[chain]
.RNG.name <- c("base::Super-Duper",
"base::Wichmann-Hill")[chain]
return(list(.RNG.seed=.RNG.seed,
.RNG.name=.RNG.name))
}

posterior <- run.jags(modelString,
                      n.chains = 1,
                      data = the_data,
                      monitor = c("beta0", "beta1", "sigma"),
                      adapt = 1000,
                      burnin = 5000,
                      sample = 5000,
                      thin = 50,
                      inits = initsfunction)
```

```
## Calling the simulation...
## Welcome to JAGS 4.3.0 on Sun Feb 23 22:02:27 2020
## JAGS is free software and comes with ABSOLUTELY NO WARRANTY
## Loading module: basemod: ok
## Loading module: bugs: ok
## . . Reading data file data.txt
## . Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 994
##    Unobserved stochastic nodes: 3
##    Total graph size: 3990
## . Reading parameter file inits1.txt
## . Initializing model
## . Adaptation skipped: model is not in adaptive mode.
## . Updating 5000
## -------------------------------------------------| 5000
## ************************************************** 100%
## . . . . Updating 250000
## -------------------------------------------------| 250000
## ************************************************** 100%
## . . . . Updating 0
## . Deleting model
## Note: the model did not require adaptation
## Simulation complete.  Reading coda files...
## Coda files loaded successfully
## Calculating summary statistics...
## Finished running the simulation
```

```r
library(coda)
post <- as.mcmc(posterior)

synthesize <- function(X, index, n){
mean_Y <- post[index, "beta0"] + X * post[index, "beta1"]
synthetic_Y <- rnorm(n, mean_Y, post[index, "sigma"])
data.frame(X, synthetic_Y)
}

n <- dim(CEdata)
m <- 20

synthetic_m <- vector("list", m)
for (l in 1:m){
synthetic_one <- synthesize(CEdata$LogExpenditure, 4980+l, n)
names(synthetic_one) <- c("logExpenditure", "logIncome_syn")
synthetic_m[[l]] <- synthetic_one
}

mean_syn_m <- vector("list", m)
for (l in 1:m){
  mean_syn_m[[l]] <- mean(synthetic_m[[l]]$logIncome_syn)
}
```

```r
median_syn_m <- vector("list", m)
for (l in 1:m){
  median_syn_m[[l]] <- median(synthetic_m[[l]]$logIncome_syn)
}

regression_m <- vector("list", m)
for (l in 1:m){
  income <- synthetic_m[[l]]$logIncome_syn
  expenditure <- synthetic_m[[l]]$logExpenditure
  linearMod <- lm(logIncome_syn ~ logExpenditure, data = synthetic_m[[l]])
  regression_m[[l]] <- linearMod$coefficients[2]
}
```

Fully synthetic data

```r
mean_syn <- unlist(mean_syn_m, use.names = FALSE)
median_syn <- unlist(median_syn_m, use.names = FALSE)
regression_syn <- unlist(regression_m, use.names = FALSE)
u_mean <- var(mean_syn)
u_median <- var(median_syn)
u_regression <- var(regression_syn)
q_bar_mean <- mean(mean_syn)
q_bar_median <- mean(median_syn)
q_bar_regression <- mean(regression_syn)
b_m_mean <- sum((mean_syn - q_bar_mean)^2) / (m - 1)
b_m_median <- sum((median_syn - q_bar_median)^2) / (m - 1)
b_m_regression <- sum((regression_syn - q_bar_regression)^2) / (m - 1)
u_bar_mean <- sum(u_mean) / m
u_bar_median <- sum(u_median) / m
u_bar_regression <- sum(u_regression) / m
T_mean <- (1 + m^(-1)) * b_m_mean - u_bar_mean
T_median <- (1 + m^(-1)) * b_m_median - u_bar_median
T_regression <- (1 + m^(-1)) * b_m_regression - u_bar_regression
```

For the mean of the synthesized log income,

```r
q_bar_mean
```

```
## [1] 10.1876
```

```r
T_mean
```

```
## [1] 0.4372374
```

For the median of the synthesized log income,

```r
q_bar_median
```

```
## [1] 10.1876
```

```r
T_median
```

```
## [1] 0.4372374
```

For the regression coefficients between the synthesized log income and the original logged expenditure,

```
q_bar_regression
```

```
## [1] -0.002719087
```

```
T_regression
```

```
## [1] 0.003523324
```

Partial synthetic data

```
T_mean_p <- b_m_mean/m + u_bar_mean
T_median_p <- b_m_median/m + u_bar_median
T_regression_p <- b_m_regression/m + u_bar_regression
```

For the mean of the synthesized log income,

```
q_bar_mean
```

```
## [1] 10.1876
```

```
T_mean_p
```

```
## [1] 0.04372374
```

For the median of the synthesized log income,

```
q_bar_median
```

```
## [1] 10.1876
```

```
T_median_p
```

```
## [1] 0.04372374
```

For the regression coefficients between the synthesized log income and the original logged expenditure,

```
q_bar_regression
```

```
## [1] -0.002719087
```

```
T_regression_p
```

```
## [1] 0.0003523324
```

Overlay interval

```
syn_data <- synthetic_m[[1]]
logincome_syn <- syn_data$logIncome_syn
logincome_original <- CEdata$LogIncome
L_s <- unname(quantile(logincome_syn, 0.025))
U_s <- unname(quantile(logincome_syn, 0.975))
L_o <- unname(quantile(logincome_original, 0.025))
U_o <- unname(quantile(logincome_original, 0.975))
L_i <- max(L_s, L_o)
U_i <- min(U_s, U_o)
I <- (U_i - L_i) / (2 * (U_o - L_o)) + (U_i - L_i) / (2 * (U_s - L_s))
I
```

```
## [1] 0.7618307
```

Since data with high utility would have a score of 1 and no data with no utility will have a score of 0, our synthesized data has a medium utility.

Project synthesize

```r
bnbData <- read.csv("AB_NYC_2019.csv")
length <- dim(bnbData)
avail <- bnbData$availability_365
Category <- c()

for(i in 1:length){
  if (avail[i] > 330){
    Category <- c(Category, 1)
  }else if( avail[i] <= 330 & avail[i] > 270){
    Category <- c(Category, 2)
  }else if( avail[i] <= 270 & avail[i] > 60){
    Category <- c(Category, 3)
  }else{
    Category <- c(Category, 4)
  }
}
```

```
## Warning in 1:length: numerical expression has 2 elements: only the first
## used
```

```r
room_type <- bnbData$room_type
room_category <- c()

for(i in 1:length){
  if (room_type[i] == "Private room"){
    room_category <- c(room_category, 1)
  }else if( room_type[i] == "Entire home/apt"){
    room_category <- c(room_category, 2)
  }else{
    room_category <- c(room_category, 3)
  }
}
```

```
## Warning in 1:length: numerical expression has 2 elements: only the first
## used
```

```r
neigh <- bnbData$neighbourhood_group
neigh_category <- c()

for(i in 1:length){
  if (neigh[i] == "Brooklyn"){
    neigh_category <- c(neigh_category, 1)
  }else if(neigh[i] == "Manhattan"){
    neigh_category <- c(neigh_category, 2)
  }else if(neigh[i] == "Queens"){
    neigh_category <- c(neigh_category, 3)
  }else if(neigh[i] == "Staten Island"){
    neigh_category <- c(neigh_category, 4)
```

```
  }else{
    neigh_category <- c(neigh_category, 5)
  }
}
```

```
## Warning in 1:length: numerical expression has 2 elements: only the first
## used
```

```
bnbData_cat <- data.frame(bnbData, category = Category, room_category = room_category, neigh_category =
neigh_unique <- unique(bnbData_cat$neighbourhood_group)
room_unique <- unique(bnbData_cat$room_type)
price <- bnbData$price
```

```
modelString <-"
model {
## sampling
for (i in 1:N){
  ez<- t (beta)%*%X[i, ]
  a<-max(-Inf , g[y[i] -1] , na . rm=TRUE)
  b<-min( g[y[i]] , Inf , na . rm=TRUE)
  u<-runif(1 , pnorm( a-ez ) , pnorm(b-ez ) )
  z[i]<- ez + qnorm(u)
  a<-max( z[y==k] )
  b<-min( z[y==k+1])
  sig[i] ~ dnorm(0,1)
  u<-runif(1 , pnorm( ( a - ez ) / sig[i] ) , pnorm(( b - ez ) / sig[i] ))
  g[i]<- ez + sig[i] * qnorm(u)
}
beta ~ dnorm(n/((n+1) * (t(X)%*%X)) %*% t(X) %*% z, n/((n+1) * (t(X)%*%X)))
}"
```

```
X <- cbind(room_category, neigh_category, price)
y <- bnbData_cat$category
```