Homework 7

Tyler Poelking

3/25/2018

QUESTION 1 (continued)

model

{

for(i in 1:nPairs){

y[i] ~ dbern(theta[school[i], pair[i]])

logit(theta[school[i], pair[i]]) <- (alpha[school[i]] +beta[school[i]]\*x[i])

}

for (s in 1:nSchools){

beta[s] ~ dnorm(mu.beta, sigma2.beta)

alpha[s] ~ dnorm(mu.alpha, sigma2.alpha)

}

mu.beta ~ dnorm(1,1/9)

mu.alpha ~dnorm(1,1/9)

sigma2.beta ~dunif(1,9)

sigma2.alpha ~dunif(1,9)

}

QUESTION 2

set.seed(23422432)  
  
library(rjags)

## Loading required package: coda

## Linked to JAGS 4.3.0

## Loaded modules: basemod,bugs

library(boot)  
library(ggplot2)  
library(grid)  
library(gridExtra)  
library(reshape2)  
  
# read in the data and calculate some  
# summaries  
schoolData <- read.table( "~/Desktop/All Stuff/School Stuff/STATS/3303/homework7/friends.txt", header=T )  
y <- schoolData$y  
x <- schoolData$x  
school <- schoolData$school  
pair <- schoolData$pair  
  
schoolNames <- unique(school)  
schoolsNumeric <- as.numeric(schoolNames)  
nSchools = length(schoolNames)  
  
nPairs <- length(schoolData)  
  
  
# create objects for JAGS  
dataList <- list( "y" = y,  
 "x" = x,  
 'pair' = pair,  
 'school' = school,  
 "nPairs" = nPairs,  
 "nSchools" = nSchools)  
  
# list of parameters to be monitored   
parameters <- c( 'alpha',"beta", "mu.beta",   
 "mu.alpha",  
 "sigma2.beta",  
 "sigma2.alpha")  
  
# set initial values  
initsValues <- list('alpha' = rep(1, nSchools),  
 'beta' = rep(1,nSchools),  
 "mu.beta" = 1,   
 "mu.alpha" = 1,  
 "sigma2.beta" = 5,  
 "sigma2.alpha" = 5)  
  
# number of iteration for "tuning"   
adaptSteps <- 5000   
  
# number of iterations for "burn-in"   
burnInSteps <- 5000   
  
# number of chains to run  
nChains <- 2   
  
# total number of iterations to save  
numSavedSteps <- 5000   
  
# "thinning" (1 = keep every interation)  
thinSteps <- 1   
  
# iterations per chain  
ITER <- ceiling( (numSavedSteps \* thinSteps) / nChains )   
  
# -------------  
# Run JAGS  
# -------------  
  
# create, initialize, and adapt the model  
jagsModel <- jags.model( "~/Desktop/All Stuff/School Stuff/STATS/3303/homework7/model.txt",   
 data = dataList,   
 inits = initsValues,   
 n.chains = nChains,   
 n.adapt = adaptSteps )

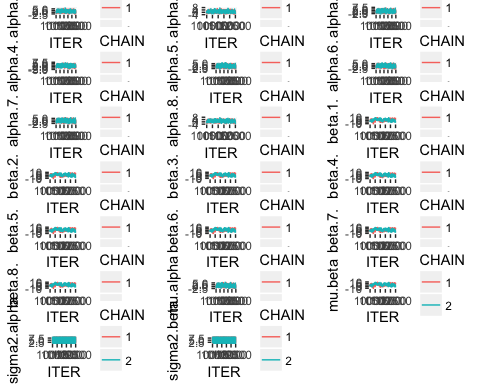
## Compiling model graph  
## Resolving undeclared variables  
## Allocating nodes  
## Graph information:  
## Observed stochastic nodes: 4  
## Unobserved stochastic nodes: 20  
## Total graph size: 6108  
##   
## Initializing model

The starting values are available in the code above.

The number of iterations is the ceiling of (numSavedSteps x thinSteps) / nChains = 5000 \* 1 / 2

Based on the trace plots, the chains appear to be sampling from the same distribution. This gives reason to believe that the algorithm converged.

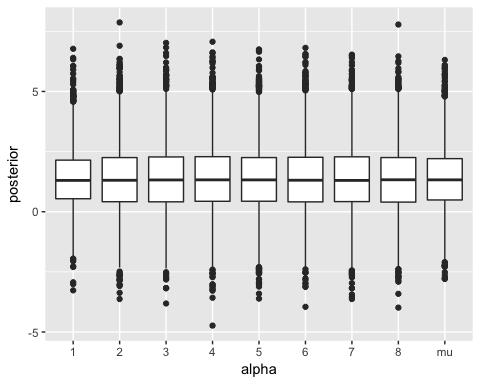
# burn-in the algorithm  
update( jagsModel,   
 n.iter = burnInSteps )  
  
# run algorithm to get interations for inference  
codaSamples <- coda.samples( jagsModel,   
 variable.names = parameters,   
 n.iter = ITER,   
 thin = thinSteps)  
  
# -------------  
# Look at posterior samples  
# -------------  
  
# make a dataframe with the posterior samples  
mcmcChainDF <- data.frame( as.matrix( codaSamples,   
 iters = T,   
 chains = T ) )  
  
# create a vector with the variable names  
varNames <- names( mcmcChainDF )[3:( dim( mcmcChainDF )[2] )]  
  
# number of variables  
nVars <- 20  
  
mcmcChainDF$CHAIN <- as.factor(mcmcChainDF$CHAIN)  
  
# construct trace plots  
  
p <- list()  
par(ask = T)  
for( k in 1:nVars )  
{  
 plot\_frame <- mcmcChainDF  
 plot\_frame$dep\_var <- mcmcChainDF[ , varNames[k]]  
 p[[k]] <- ggplot( plot\_frame,   
 aes( x = ITER,   
 y = dep\_var)) +  
 geom\_line( aes( color = CHAIN ) ) +   
 labs( y = varNames[k] )  
}  
  
do.call( grid.arrange, c( p, list("ncol" = 3)))



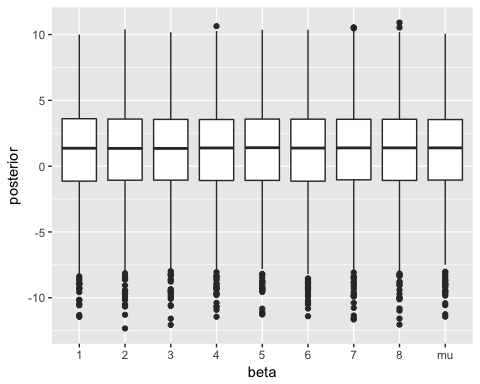
\*Note: These graphs were much bigger and easier to read in R.

QUESTION 3

postDFreshape <- melt( mcmcChainDF,   
 id.vars = "ITER",  
 measure.vars = c("alpha.1.",  
 "alpha.2.",  
 "alpha.3.",  
 "alpha.4.",  
 "alpha.5.",  
 "alpha.6.",  
 "alpha.7.",  
 "alpha.8.",  
 "mu.alpha"))  
ggplot(postDFreshape,   
 aes(x = variable, y = value )) +  
 geom\_boxplot() +  
 scale\_x\_discrete( labels = c( schoolNames, "mu" )) +  
 ylab( "posterior" ) +  
 xlab( "alpha" )



postDFreshape <- melt( mcmcChainDF,   
 id.vars = "ITER",  
 measure.vars = c("beta.1.",  
 "beta.2.",  
 "beta.3.",  
 "beta.4.",  
 "beta.5.",  
 "beta.6.",  
 "beta.7.",  
 "beta.8.",  
 "mu.beta"))  
ggplot(postDFreshape,   
 aes(x = variable, y = value )) +  
 geom\_boxplot() +  
 scale\_x\_discrete( labels = c( schoolNames, "mu" )) +  
 ylab( "posterior" ) +  
 xlab( "beta" )



The levels of alpha and beta show minimal difference between the 8 schools. All 8 alphas and all 8 betas are very similar to their corresponding averages. This shows that there is not a significant difference between the odds that a pair of students are pairs across school, even when taking mixed vs non-mixed genders into account.

Consider a new school with the same number of juniors as the schools in the dataset. Approximate the predictive probability that an arbitrary pair of two girls at the new school will be close friends

alpha.post <- mcmcChainDF$mu.alpha  
prob\_alpha.post <- inv.logit(alpha.post)  
mean(prob\_alpha.post)

## [1] 0.7404076

In addition, approximate the predictive probability that an arbitrary pair of one girl and one boy at the new school will be close friends.

beta.post <- mcmcChainDF$mu.beta  
x = alpha.post + beta.post  
prob\_x.post <- inv.logit(x)  
mean(prob\_x.post)

## [1] 0.7488735