options(scipen=999)

library(R2OpenBUGS)

library(MCMCvis)

library(jagsUI)

library(loo)

N=21

P=4

D=list(y=c(10,23,23,26,17,5,53,55,32,46,10,8,10,8,23,0,3,22,15,32,3),

n=c(39,62,81,51,39,6,74,72,51,79,13,16,30,28,45,4,12,41,30,51,7),

x1=c(0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1,1,1),

x2=c(0,0,0,0,0,1,1,1,1,1,1,0,0,0,0,0,1,1,1,1,1),N=N,P=P)

**# Model 1 Mixed Predictive Tests, log(CPO) and Posterior Predictive Checks**

cat("

model {for( i in 1 : N ) { y[i] ~ dbin(p[i],n[i])

logit(p[i]) <- beta[1]+beta[2]\*x1[i]

+beta[3]\*x2[i]+beta[4]\*x1[i]\*x2[i]+b[i]

b[i] ~ dnorm(0,tau.b)

LL[i] <- logfact(n[i])-logfact(y[i])-logfact(n[i]-y[i])

+y[i]\*log(p[i])+(n[i]-y[i])\*log(1-p[i])

H[i] <- 1/exp(LL[i])

# replicates from orginal model

ynew[i] ~ dbin(p[i],n[i])

LLnew[i] <- logfact(n[i])-logfact(ynew[i]) -logfact(n[i]-ynew[i])

+ynew[i]\*log(p[i])+(n[i]-ynew[i])\*log(1-p[i])}

# replicates based on sampling new random effects (mixed approach)

for( i in 1 : N ) { y.mx[i] ~ dbin(p.mx[i],n[i])

b.mx[i] ~ dnorm(0,tau.b)

logit(p.mx[i]) <- beta[1]+beta[2]\*x1[i]+beta[3]\*x2[i]

+beta[4]\*x1[i]\*x2[i]+b.mx[i]

p.cv.mx[i] <- step(y.mx[i]-y[i])-0.5\*equals(y.mx[i],y[i])

LLmx[i] <- logfact(n[i])-logfact(y.mx[i])

-logfact(n[i]-y.mx[i])+y.mx[i]\*log(p.mx[i])

+(n[i]-y.mx[i])\*log(1-p.mx[i])}

# Priors

for (j in 1:P) {beta[j] ~ dnorm(0,0.001)}

tau.b ~ dgamma(1,0.001)

sigma.b <- 1 / sqrt(tau.b)

# Posterior Predictive Checks

DVnew <- -2\*sum(LLnew[])

DV <- -2\*sum(LL[])

DVmx <- -2\*sum(LLmx[])

PPC[1] <- step(DVnew-DV)

PPC[2] <- step(DVmx-DV)}

", file="model1.jag")

**# Estimation**

inits1 = list(tau.b=10, beta=rep(0,4))

inits2 = list(tau.b=100, beta=rep(0,4))

inits=list(inits1,inits2)

pars = c("beta","b","sigma.b","H","p.cv.mx","PPC")

R1 = autojags(D, inits, pars,model.file="model1.jag",2,iter.increment=500, n.burnin=100,Rhat.limit=1.1, max.iter=2000, seed=1234)

R1$summary

**# log(CPO) and Mixed CV Probabilities by Case**

sampsH = as.matrix(R1$sims.list$H)

sampscvmx = as.matrix(R1$sims.list$p.cv.mx)

p.cv.mx=apply(sampscvmx,2,mean)

logCPO=-log(apply(sampsH,2,mean))

fit = data.frame(p.cv.mx,logCPO)

list.fit=fit[order(-fit$p.cv.mx),]

head(list.fit,5)

tail(list.fit,5)

**# Model 2: Full Cross Validation**

**# data copies for each of N cross validations**

model2 <- function() { for( i in 1 : N ) { for (j in 1:N) {y.copy[i,j] <- y[i]}}

**# likelihood for N copies and N observations (r-th regression excludes r-th observation)**

for (r in 2:N-1) { for (i in 1:r-1) {y.copy[i,r] ~ dbin(p[i,r],n[i])}

for (i in r+1:N) {y.copy[i,r] ~ dbin(p[i,r],n[i])}}

for (i in 1:N-1) {y.copy[i,N] ~ dbin(p[i,N],n[i])}

for (i in 2:N) {y.copy[i,1] ~ dbin(p[i,1],n[i])}

for (r in 1:N) { for (i in 1:N) {

logit(p[i,r]) <- beta[1,r]+beta[2,r]\*x1[i]+beta[3,r]\*x2[i]

+beta[4,r]\*x1[i]\*x2[i] + b[i,r]}}

**# cross-validatory prediction of y[r] from r-th CV regression (which excludes y[r])**

for (r in 1:N) {y.new[r] ~ dbin(p[r,r],n[r])

LL[r] <- logfact(n[r])-logfact(y[r])-logfact(n[r]-

y[r])+y[r]\*log(p[r,r])+(n[r]-y[r])\*log(1-p[r,r])

p.cv[r] <- step(y.new[r]-y[r])-0.5\*equals(y.new[r],y[r])}

**# priors for each of N estimations**

for (r in 1:N) { for (i in 1:N) {b[i,r] ~ dnorm(0,tau[r])}

for (j in 1:P) {beta[j,r] ~ dnorm(0.0,1.0E-6)}

tau[r] ~ dgamma(1,0.001)}}

**# Estimation**

inits1 = list(tau=rep(10,N), beta=matrix(0,P,N))

inits2 = list(tau=rep(100,N), beta=matrix(0,P,N))

inits=list(inits1,inits2)

pars = c("p.cv","tau")

n.iters=10000; n.burnin =500; n.chains=2

R2 = bugs(D,inits,pars,n.iters,model2,n.chains, n.burnin,debug=T,codaPkg = F,bugs.seed=10)

R2$summary

**# CV Probabilities**

p.cv.full=as.vector(apply(R2$sims.list$p.cv,2,mean))

fit = data.frame(p.cv.mx,logCPO,p.cv.full)

list.fit=fit[order(-fit$p.cv.full),]

head(list.fit,5)

tail(list.fit,5)

**# Integrated Importance Sampling**

**# Number of Replicate Subsamples**

D$S=10

model3 <- function() { for( i in 1 : N ) {y[i] ~ dbin(p[i],n[i])

b[i] ~ dnorm(0,tau)

logit(p[i]) <- beta[1] + beta[2] \* x1[i] +

beta[3] \* x2[i] + beta[4] \* x1[i] \* x2[i] + b[i]

LL[i] <- logfact(n[i])-logfact(y[i])-logfact(n[i]-y[i])

+y[i]\*log(p[i])+(n[i]-y[i])\*log(1-p[i])

# Integrated importance weight based on averages over sub-samples

IIW[i] <- mean(L.new[i,])

logIIW[i] <- log(IIW[i])

# IIS-CV predictive estimates are posterior mean of A divided by posterior mean of Awt

A[i] <- mean(a[i,])\*Awt[i]

Awt[i] <- 1/IIW[i]

# subsamples

for (s in 1:S) { b.new.A[i,s] ~ dnorm(0, tau)

logit(p.new.A[i,s]) <- beta[1]+beta[2]\*x1[i]

+beta[3]\*x2[i]+ beta[4]\*x1[i]\*x2[i]+b.new.A[i,s]

y.new[i,s] ~ dbin(p.new.A[i,s],n[i])

a[i,s] <- step(y.new[i,s]-y[i])-0.5\*equals(y.new[i,s],y[i])

b.new.B[i,s] ~ dnorm(0, tau)

logit(p.new.B[i,s]) <- beta[1]+beta[2]\*x1[i]

+beta[3]\*x2[i]+ beta[4]\*x1[i]\*x2[i]+b.new.B[i,s]

log(L.new[i,s]) <- logfact(n[i])-logfact(y[i])-logfact(n[i]-y[i])

+y[i]\*log(p.new.B[i,s])+(n[i]-y[i])\*log(1-p.new.B[i,s])}}

# priors

for (j in 1:P) {beta[j] ~ dnorm(0.0,1.0E-6)}

tau ~ dgamma(1,0.001)}

**# Estimation**

inits1 = list(tau=10, beta=rep(0,P))

inits2 = list(tau=100, beta=rep(0,P))

inits=list(inits1,inits2)

pars = c("beta","tau","A","Awt","IIW","logIIW","LL")

n.iters=10000; n.burnin =500; n.chains=2

R3 = bugs(D,inits,pars,n.iters,model3,n.chains, n.burnin,debug=T,codaPkg = F,bugs.seed=10)

R3$summary

**# cross-validation probabilities, IIS method**

p.cv.iis=

as.vector(R3$sims.list$A,2,mean))/ as.vector(apply(R3$sims.list$Awt,2,mean))

**# Integrated WAIC**

iWAIC.case=log(as.vector(apply(R3$sims.list$IIW,2,mean)))-as.vector(apply(R3$sims.list$logIIW,2,sd))^2

**# Compare CV probabilities by method, log(CPO) and casewise IWAIC**

fit = data.frame(p.cv.mx,p.cv.iis,p.cv.full,logCPO,iWAIC.case)

fit

list.fit=fit[order(-fit$p.cv.full),]

head(list.fit,5)

tail(list.fit,5)

cor(logCPO,iWAIC.case)

**# Total fit measures**

iWAIC=-2\*sum(iWAIC.case)

waic(R3$sims.list$LL)

loo(R3$sims.list$LL)

**# Discrete mixture intercept**

model4 <- function() {for( i in 1 : N ) { y[i] ~ dbin(p[i],n[i])

# discrete mixture intercept

logit(p[i]) <- beta0[G[i]]+beta[1]\*x1[i]+beta[2]\*x2[i]

+beta[3]\*x1[i]\*x2[i]+b[i]

b[i] ~ dnorm(0,tau.b)

LL[i] <- logfact(n[i])-logfact(y[i])

-logfact(n[i]-y[i])+y[i]\*log(p[i])+(n[i]-y[i])\*log(1-p[i])

H[i] <- 1/exp(LL[i])

# discrete mixture selection

G[i] ~ dcat(phi[1:3])

ynew[i] ~ dbin(p[i],n[i])

LLnew[i] <- logfact(n[i])-logfact(ynew[i])-logfact(n[i]-ynew[i])+ynew[i]\*log(p[i])+(n[i]-ynew[i])\*log(1-p[i])}

# replicates based on sampling new random effects (mixed approach)

for( i in 1 : N ) { y.mx[i] ~ dbin(p.mx[i],n[i])

b.mx[i] ~ dnorm(0,tau.b)

logit(p.mx[i]) <- beta0[G[i]]+beta[1]\*x1[i]+beta[2]\*x2[i]+

beta[3]\*x1[i]\*x2[i]+b.mx[i]

p.cv.mx[i] <- step(y.mx[i]-y[i])-0.5\*equals(y.mx[i],y[i])

LLmx[i] <- logfact(n[i])-logfact(y.mx[i])-logfact(n[i]-y.mx[i])

+y.mx[i]\*log(p.mx[i])+(n[i]-y.mx[i])\*log(1-p.mx[i])}

# Priors

for (j in 1:3) {beta[j] ~ dnorm(0,0.001)

beta0s[j] ~ dnorm(0,0.001)

beta0[j] <- ranked(beta0s[],j)

w.G[j] <- 5}

phi[1:3] ~ ddirich(w.G[1:3])

tau.b ~ dgamma(1,0.001)

sigma.b <- 1 / sqrt(tau.b)

# Posterior Predictive Checks

DVnew <- -2\*sum(LLnew[])

DV <- -2\*sum(LL[])

DVmx <- -2\*sum(LLmx[])

PPC[1] <- step(DVnew-DV)

PPC[2] <- step(DVmx-DV)}

**# Estimation**

inits1 = list(tau.b=10, beta=rep(0,3), beta0s=rep(0,3))

inits2 = list(tau.b=100, beta=rep(0,3), beta0s=rep(0,3))

inits=list(inits1,inits2)

pars = c("beta0","phi","PPC","LL","p.cv.mx")

n.iters=10000; n.burnin =500; n.chains=2

R4= bugs(D,inits,pars,n.iters,model4,n.chains, n.burnin,debug=T,codaPkg = F,bugs.seed=10)

R4$summary

**# Casewise fit**

p.mix.discmix=as.vector(apply(R4$sims.list$p.cv.mx,2,mean))

fit = data.frame(p.cv.full,p.cv.mx,p.cv.iis, logCPO,iWAIC.case,p.mix.discmix)

fit

list.fit=fit[order(-fit$p.cv.full),]

head(list.fit,5)

tail(list.fit,5)

sum(p.mix.discmix > 0.95)+ sum(p.mix.discmix < 0.05)

**# Fit**

loo(R4$sims.list$LL)

waic(R4$sims.list$LL)