library(R2OpenBUGS); library(loo);library(brms)

setwd("C:/R files BHMRA")

attach("DS\_8\_2.Rdata")

attach(DS\_8\_2)

N=DS\_8\_2$N

**# BRMS data input format, group variables input for each subject**

DBRMS = read.table("Example\_8\_2\_wellbeing\_brms.txt",header=T)

**# BRMS, multilevel logistic using rstan**

# Model 1: group-varying intercepts, specify brm call so that random effects have mean zero

BRMS1 = brm(wb ~ 1+hrs+hrs.grp+(1|grp), data = DBRMS, family = "bernoulli", chains = 2)

summary(BRMS1)

**# model fit**

WAIC(BRMS1)

LOO(BRMS1)

**# group level random effect estimates**

ranef(BRMS1, var=T)

hist(ranef(BRMS1)$grp)

# full data posterior predictive check

P <- predict(BRMS1,summary=F,nsamples=100)

ch <- matrix(,N,100)

for (i in 1:N) {for (s in 1:100) {ch[i,s]<- if(DS\_8\_2$wb[i]==P[s,i])1 else 0}}

pr.concord <- numeric(N)

**# subjects with lowest cross-validatory concordance**

for (i in 1:N) {pr.concord[i] <- sum(ch[i,])/100}

cvtail.full <- sum(pr.concord <0.5)/N

**# MODEL 1 CODE AUGMENTED DATA LOGISTIC**

model1 <- function() {for (h in 1:N) {

z[h] ~ dlogis(eta[h],1) %\_% I(A[h],B[h])

A[h] <- -100\*equals(wb[h],0)

B[h] <- 100\*equals(wb[h],1)

eta[h] <- alph + beta[1]\*hrs.c[h]+b[grp[h]]

LL[h] <- wb[h]\*log(p[h])+(1-wb[h])\*log(1-p[h])

# centre predictor

hrs.c[h] <- hrs[h]-mean(hrs[])

# residual in latent data scale

resid[h] <- z[h]-eta[h]

# estimated well-being probability

logit(p[h]) <- eta[h]

# mixed posterior predictive check

z.mx[h] ~ dlogis(eta.mx[h],1)

eta.mx[h] <- alph+beta [1]\*hrs.c[h]+b.mx[grp[h]]

testmx[h] <- step(z.mx[h])\*equals(wb[h],1)

+ step(-z.mx[h])\*equals(wb[h],0)

testmxpos[h] <- step(z.mx[h])\*equals(wb[h],1)}

# overall (mixed predictive) concordance between actual and predicted binary outcomes

CVconcordance <- sum(testmx[])/N

# concordance for y=1 subjects only (sensitivity)

CVsensit <- sum(testmxpos[])/sum(wb[])

# priors

alph ~ dnorm(0,0.001)

sig.b ~ dunif(0,100)

tau.b <- 1/(sig.b\*sig.b)

for (a in 1:2) {beta[a] ~ dnorm(0,0.001)}

# group effects

for (i in 1:m) {mu.b[i] <- beta[2]\*(hrs.g[i]-mean(hrs.g[]))

b[i]~dnorm(mu.b[i],tau.b);

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

b.cent[i] <- b[i]-mean(b[])

# cumulate predictions within clusters

testmx.grp[i] <- sum(testmx[cumgrp[i]+1:cumgrp[i+1]])/

numgrp[i]}}

**# Initial Values and Estimation**

init1 = list(alph=0,sig.b=1,beta=rep(0,2), b=rep(0,99),z=rep(0,7382))

init2 = list(alph=1,sig.b=2,beta=rep(-0.1,2), b=rep(0,99),z=rep(0,7382))

inits <- list(init1,init2)

n.iters=2000; n.burnin =500; n.chains=2

**# Estimated Hyperparameters And Cluster Effects**

pars = list("alph","beta","sig.b", "CVconcordance", "CVsensit", "b.cent","LL", "testmx","testmx.grp")

M1 = bugs(DS\_8\_2,inits,pars,n.iters,model1,n.chains, n.burnin,debug=T,codaPkg = F, bugs.seed=10)

M1$summary

**# Fit Measures**

loo(M1$sims.list$LL)

**# cross-validatory concordance**

CVC=apply(M1$sims.list$testmx,2,mean)

sapply(sort(CVC, index.return=T), `[`, 1:10)

cvtail.mx <- sum(CVC < 0.5)/N

**# clusters with lowest and highest cross-validatory concordance**

test.grp=apply(M1$sims.list$testmx.grp,2,mean)

sapply(sort(test.grp, index.return=T), `[`, 1:5)

sapply(sort(test.grp, index.return=T), `[`, 95:99)