**Project R Code**

# This analysis attempts to predict batting average

# The goal is to compare frequentist and bayesian methods

# Data come from Lahman Baseball data archive, copyright Sean Lahman

# Source: http://www.seanlahman.com/baseball-archive/statistics/

# Authors: Kenneth Feder, Joseph High, Joseph Yu

# Load required packages -- may need to be installed

library(ggplot2)

library(rjags)

library(lme4)

library(plyr)

library(dplyr) # Note -- I rely heavily on this package for data manipulation! Would strongly encourage you to review some of the documentation, it's the most useful R package

library(magrittr)

library(coda)

library(beepr)

# Set directory to read in data

setwd("~/Documents/Johns Hopkins/Coursework/Term 9/Bayesian Statistics/Project/baseballdatabank-2017/core") # wil be changed on other computers!

# read in player demographics and batting data

master <- read.csv("Master.csv")

batting <- read.csv("Batting.csv")

# Reset directory to working directory

setwd("~/Documents/Johns Hopkins/Coursework/Term 9/Bayesian Statistics/Project")

# Merge the two datasets

mlb <- right\_join(master,batting,by = "playerID")

summary(mlb)

####################################### Preparing for Analysis ###########################################

head(mlb)

filter(mlb,birthYear == 1852,birthState == "MD")

# Rearrange for easier viewability

mlb %<>% arrange(playerID,yearID)

head(mlb)

summary(mlb)

# Data cleaning for analysis

mlb.analysis <- mlb %>% select(playerID,birthYear,weight,height,yearID,teamID,lgID,G,AB,H) %>% # Extract relvant variables...

filter(yearID > 1900,AB >= 100,lgID == "NL" | lgID == "AL") %>% # Filter only post 1901, at least 100 AB per season, only NL and AL play

mutate( # Add transformed variables

age = yearID - birthYear,

age.square = age^2,

year = yearID - 1900,

spline.liveball = ifelse(yearID < 1920,0,yearID - 1919), # Allow slope to shift at start of liveball era using spline http://www.netshrine.com/era.html

spline.expansion = ifelse(yearID < 1961,0,yearID - 1961), # Allow slope to shift at start of expansion era

spline.freeagency = ifelse(yearID < 1977,0,yearID - 1977), # Allow slope to shift at start of free agency

spline.steroids = ifelse(yearID < 1994,0,yearID - 1994), # Allow slope to shift at start of steroid era

spline.modern = ifelse(yearID < 2005,0,yearID - 2004), # Allow slope to shift at start of modern era

height.square = height^2,

weight.square = weight^2,

avg = H/AB

) %>% group\_by(playerID) %>% # Group by player ID...

mutate( # So we can create lagged batting average variables for each player

avg.lag1 = lag(avg),

avg.lag2 = lag(avg.lag1),

season = row\_number()

) %>% filter(season > 2) %>% # Drop the first two seasons of a player's career

na.omit %>% # Drop records with missing information (a small number of players with no height/weight info from early 1900s)

as.data.frame()

# Exploratory plot -- histogram of batting averages

plot.raw.dist <- ggplot(mlb.analysis,aes(x = avg)) +

geom\_histogram() +

ggtitle("Histogram of Major League Batting Averages")

plot.raw.dist

# Exploratory plot -- batting trends over time

plot.raw.trends <- ggplot(mlb.analysis,aes(yearID,avg,group = playerID)) +

geom\_line(size = .1, alpha = .5) +

ggtitle("Historical trends in Major League Batting") +

xlab("year")

plot.raw.trends

####################################### Fit frequentist multi-level model ################################

names(mlb.analysis)

fit.freq <- lmer(avg ~ avg.lag1 + avg.lag2 + age + age.square +

height + height.square + weight + weight.square +

year + spline.liveball + spline.expansion + spline.freeagency +

spline.steroids + spline.modern + (1|playerID),

data = mlb.analysis)

summary(fit.freq)

# Predictive accuracy

mlb.analysis$predict.freq <- predict(fit.freq)

plot.freqfits.vs.truth <- ggplot(mlb.analysis,aes(avg,predict.freq)) +

geom\_point() +

geom\_abline(slope = 1,intercept = 0) +

ggtitle("Predictive Accuracy of Frequentist Estimation") +

xlab("true avg") +

ylab("predicted avg")

plot.freqfits.vs.truth

# A few example trajectories from post 1990 players

set.seed(123)

plot.example.trajectories <- mlb.analysis %>% filter(yearID > 1990) %>%

filter(playerID %in% sample(unique(playerID),12)) %>%

ggplot() +

geom\_point(aes(yearID,avg),size = .5,alpha = .9) +

geom\_line(aes(yearID,predict.freq)) +

facet\_wrap(~playerID,scales = "free") +

ggtitle("Example: True and Predicted Batting\nAverages for 12 Randomly Chosen Players") +

xlab("year") +

theme(axis.text.x=element\_blank())

plot.example.trajectories

# All predicted trajectories

plot.predicted.trends <- ggplot(mlb.analysis,aes(yearID,predict.freq,group = playerID)) +

geom\_line(alpha = .3) +

ggtitle("Predicted Trajectory of MLB performance")

plot.predicted.trends

# Overly predictions on data. Less variable, which is what we would expect -- predictions do not include residual variance

plot.freq.vs.data.hist <- ggplot(mlb.analysis) +

geom\_density(aes(predict.freq),fill = 'blue',alpha = .5) +

geom\_density(aes(avg),fill = 'red',alpha = .5) +

ggtitle("Frequentist Estimation Does Not Capture\nUncertainty in Player Predicted Averages") +

xlab("True (red) and Predicted (blue)\nBatting Average Distributions")

plot.freq.vs.data.hist

######################################### Bayesian multi-level model ###################################

# We are using the Bayesian engine jags (Just Another Gibbs Sampler)

# 4 steps to every model 1) specify, 2) intialize, 3) burn-in, 4) run MCMC

# NOTE -- THE MODEL ITSELF IS IN baseball\_project.bug

# These are variables needed to run the model using jags

set.seed(456)

n <- nrow(mlb.analysis)

J <- length(table(mlb.analysis$playerID))

tests <- sort(sample(1:nrow(mlb.analysis),100)) # This is rownumbers for 100 player-years we select at random to make predictions on.

mlb.analysis$playerID.numeric <- as.numeric(as.factor(mlb.analysis$playerID))

# The parameters to be returned after running the MCMC. For simplicity, I only return the coefficients associated with batting average

jags.parameters <- c("b1", "b2")

# Starting values for the MCMC

jags.inits <- list(b0 = rep(0,J),b1 = 0,b2 = 0,b3 = 0,b4 = 0,g0 = 0,g1 = 0,g2 = 0,g3 = 0,g4 = 0,g5 = 0,

g6 = 0,g7 = 0,g8 = 0,g9 = 0,g10 = 0,avg.precision = 1,b0.precision = 1)

attach(mlb.analysis)

# All the data we pass to the model has to be stored in a list

jags.data <- list(

"n" = n,

"J" = J,

"tests" = tests,

"playerID" = playerID.numeric,

"avg" = avg,

"avg.lag1" = avg.lag1,

"avg.lag2" = avg.lag2,

"age" = age,

"age.square" = age.square,

"height" = height,

"height.square" = height.square,

"weight" = weight,

"weight.square" = weight.square,

"year" = year,

"spline.liveball" = spline.liveball,

"spline.expansion" = spline.expansion,

"spline.freeagency" = spline.freeagency,

"spline.steroids" = spline.steroids,

"spline.modern" = spline.modern

)

# This is the key step -- we compile the model from the JAGS code, the data, and the specifications we give

avg.jags.model <- jags.model(file = "baseball\_project.bug", # Name of file containing model

data = jags.data, # The data to use

inits = jags.inits, # Starting values for the MCMC

n.adapt = 1000, # Has to do with refining the Gibbs algorithm adaptively...?

n.chains = 3 # Run 3 parallel chains (for convergence purposes)

)

# This is the burn in. Takes about 2 minutes

update(avg.jags.model, n.iter=1000) # burn in

# Now we actually run the MCMC.

t <- Sys.time()

beta.samples<-coda.samples(avg.jags.model, variable.names = jags.parameters, n.iter = 5000,thin = 5)

Sys.time() - t

beep()

# Note the estimated coefficients are very similar to frequentist estimates, which is encouraging

summary(beta.samples)

summary(fit.freq)

# Here are some convergence diagnostics

plot(beta.samples)

acfplot(beta.samples)

gelman.diag(beta.samples)

# Now let's rerun and get a posterior predictive distribution for a random sample of 100 players.

# This is quite cumbersome, which is why we did the conergence diagnostics up front

# But we want this to make a histogram to compare to our data

jags.parameters <- c("pavg")

# This is the burn in

update(avg.jags.model, n.iter=1000) # burn in

# Now we actually run the MCMC.

t <- Sys.time()

avg.jags.samples<-coda.samples(avg.jags.model, variable.names = jags.parameters, n.iter = 5000,thin = 5)

Sys.time() - t

beep()

str(avg.jags.samples)

# Make histogram of predicted values for those hundrd players. Overlay on true batting avg distribution.

View(prediction.matrix)

prediction.matrix <- avg.jags.samples[[1]]

all.predictions <- as.numeric(prediction.matrix)

# Not to shabby, eh?

plot.bayes.vs.data.hist <- qplot() +

geom\_density(aes(mlb.analysis$avg),fill = 'blue',alpha = .5) +

geom\_density(aes(all.predictions),fill = 'red',alpha = .5) +

xlab("True (blue) and Predicted (red)\nBatting Average Distributions") +

ggtitle("Bayesian Estimation Does Captures\nUncertainty in Player Predicted Averages")

plot.bayes.vs.data.hist

# Let's add individual predictions

PosteriorSummary <- function(y){

c(mean(y),quantile(y,.025),quantile(y,.975))

}

predictions.bayes <- apply(prediction.matrix,2,PosteriorSummary)

truth <- mlb.analysis %>% slice(tests) %>%

select(avg) %>%

pull(avg)

predictions.bayes <- rbind(truth,predictions.bayes) %>%

t() %>%

as.data.frame() %>%

mutate(num = row\_number())

colnames(predictions.bayes) <- c("truth","predicted.mean","lb","ub","id")

predictions.bayes %<>% mutate(

contains = truth > lb & truth < ub

)

sapply()

plot.bayes.vs.data.individual <- ggplot(predictions.bayes) +

geom\_point(aes(x = factor(id),y = truth,colour = contains),size = 1) +

geom\_linerange(aes(x = factor(id),ymin = lb,ymax = ub),alpha = .5) +

ggtitle("95% Credible Interval Contains Truth\n95 times in Random Sample of 100 Players") +

ylab("True Batting Average and 95% Credible Interval") +

xlab(NULL) +

theme(axis.text.x=element\_blank()) +

scale\_colour\_discrete(name = "Contains\nTruth?")

plot.bayes.vs.data.individual