Estimating the shooting efficiency of top NBA point guard

Xiang Gao, Jerome Finn, Nilesh Malpekar

2017/12/03

Purpose

Background

As we all know,a point guard controll the court by passing the ball to the player who is wide open. He is a decision maker to deliver assists or finish the attack by himself. The question arises, who has the highest shooting percentage among the top NBA point guards. Is this related to the professional years of experience.

We are using logistic regression to assess the probablity of shooting, also use a binomial model to estimate FGM. In order to build a hierachical model, each player is treated as a individual group by introducing a random effect called player effect. What's more, recent three years data will be used to check the continuous improvement.

Data

Original data

Original data is retrieved from Kaggle competition site NBA Dataset

Using our homework datasets as a guide, Xiang got our data set to manageable level for our questions. We concentrate on players and years with players representing groups similar to how rats where used as groups in our previous lectures.

The zip file contains two separate CSV files * Seasons_Stats.csv - season specific data since 1950 * Players.csv - player specific data

```
season_data <- read.table("Seasons_Stats.csv", header=TRUE, sep = ",", quote = '"')
player_data <- read.table("Players.csv", header=TRUE, sep = ",", quote = '"')
```

For this project, we are focusing on specific fields within this dataset which are described beow:

```
data_meta <- data.frame(
  Datasource = c("Seasons_Stats", "Season_Stats, Players", "Players", rep("Seasons_Stats", 8)),
  Field_Name = c("Year",
    "Player", "Height", "Pos", "Tm",
    "Age", "MP", "PER", "FG", "FGA", "BLK"),
  Description = c("NBA year",
    "Player name", "Height in cm", "Player position", "NBA team name",
    "Player age", "Minutes played", "Player Efficiency Rating",
    "Field Goals", "Field Goals Attempted", "Blocks")
  )

kable(data_meta, caption = "Data source fields description", format = "html")
```

Data source fields description

Datasource

Field_Name

Description

Seasons_Stats

Year

NBA year

Season_Stats, Players

Player

Player name

Players

Height

Height in cm

Seasons_Stats

Pos

Player position

Seasons_Stats

Tm

NBA team name

Seasons_Stats

Age

Player age

Seasons_Stats

MP

Minutes played

Seasons_Stats

PER

Player Efficiency Rating

Seasons_Stats

FG

Field Goals

Seasons_Stats

FGA

Field Goals Attempted

Seasons_Stats

BLK

Blocks

The dataset contains duplicate rows for multiple players for the same year. As part of the data preparation, we have removed duplicate rows based on **Year** and **Player**.

```
##    Year     Player FG
## 4  1950 Ed Bartels 22
## 5  1950 Ed Bartels 21
## 6  1950 Ed Bartels  1
## 86 1950  Al Guokas 93
## 87 1950  Al Guokas 86
## 88 1950  Al Guokas  7
```

```
# season data has multiple records for a player and year
season_data <- season_data[with(season_data, order(Year, Player, -FG)), ]
season_data <- distinct(season_data, Year, Player, .keep_all = TRUE)
```

Feature creation

For this project, we need to extract following two features from the original dataset

- **Experience** as number of years of NBA experience.
- **FG%** as $FieldGoals/FieldGoalsAttempted$

```
season_data <- season_data %>% group_by(Player) %>% mutate(EXP = 1:n())
season_data[, "FG%"] <- season_data$FG / season_data$FGA
```

```
## Source: local data frame [8 x 6]
## Groups: Player [1]
##
##    Year        Player   EXP    FG   FGA     `FG%`
##   <int>        <fctr> <int> <int> <int>     <dbl>
## 1  2010 Stephen Curry     1   528  1143 0.4619423
## 2  2011 Stephen Curry     2   505  1053 0.4795821
## 3  2012 Stephen Curry     3   145   296 0.4898649
## 4  2013 Stephen Curry     4   626  1388 0.4510086
## 5  2014 Stephen Curry     5   652  1383 0.4714389
## 6  2015 Stephen Curry     6   653  1341 0.4869500
## 7  2016 Stephen Curry     7   805  1598 0.5037547
## 8  2017 Stephen Curry     8   675  1443 0.4677755
```

Merge Seasons_Stat with Players to get height of the player

```
# From players data only interested in height
player_data <- subset(player_data, select = c("Player", "height"))
colnames(player_data)[2] <- "Height"
season_data <- merge(season_data, player_data, by = c("Player"))
```

Preparing modeling data

The data so far is in long format, i.e. for each player there is one row per year. However, we need daa in wide format so that there is a single row per player and year specific attributes should be columns in the dataset. As an example for **FG** (Field Goals) columns should be as follows:

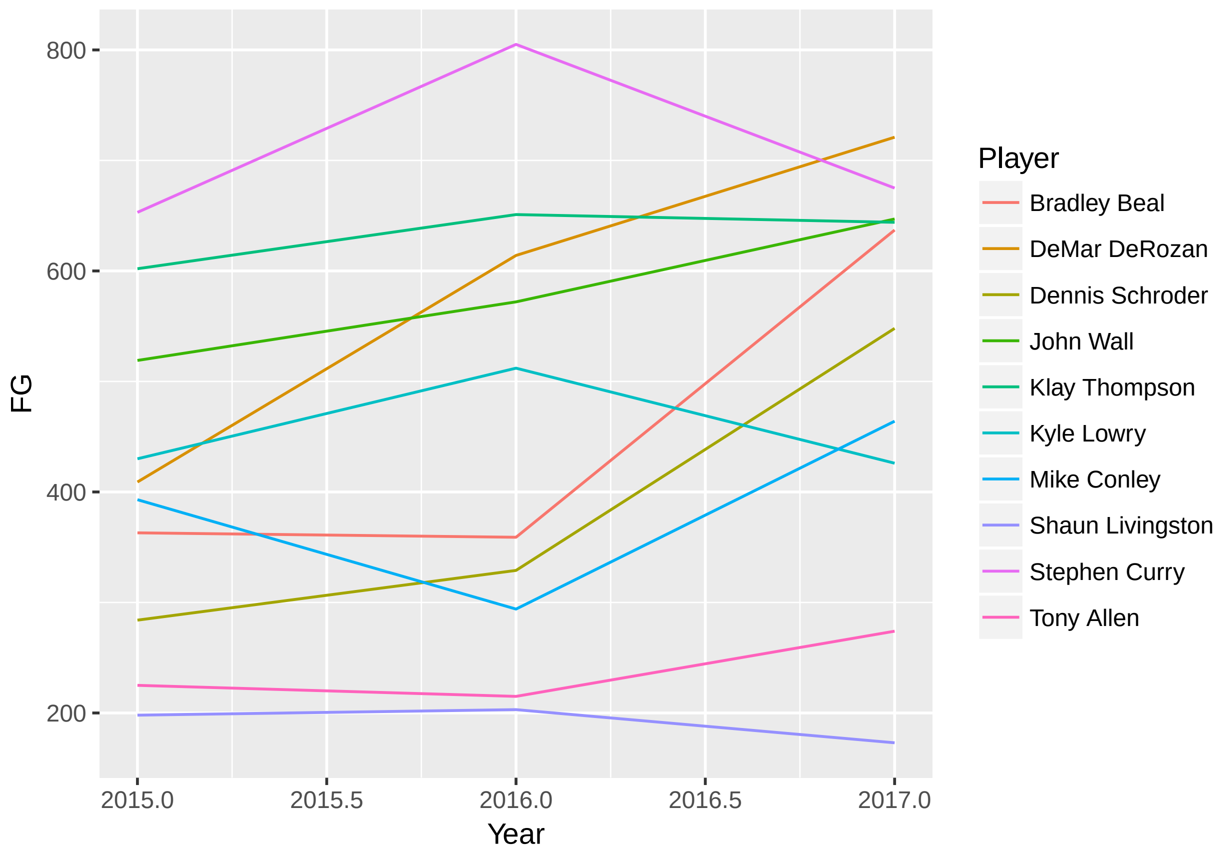
- latest year: FG**_0**
- 1st prior year: FG**_PRIOR_1**
- Nth prior year: FG**_PRIOR_N**

Get modeling data

```
model_data <- get_model_data_wide(season_data, INTERESTED_YEARS, INTERESTED_POSITIONS)
# filter by teams
model_data <- subset(model_data, Tm %in% INTERESTED_TEAMS)
model_data$Tm <- factor(model_data$Tm)
model_data_row_count <- nrow(model_data)
```

Modeling data visualization

The chat below shows field goals for each player per year. For most player there is growth in field goals from previous year to next year.



Model

Notation

$$y_{ji} = \text{Shooting rate of player *i* at year *j*} \; x_1 = 2017, \, x_2 = 2016, \, x_3 = 2015$$

$$y_i \mid \beta, \, X_i \quad \text{indep. } \mathrm{Bin}(n_i, \, p_i) \; \mathrm{logit}(p_i) = X_i\beta + \epsilon_i, \, \epsilon_i \quad \text{iid } \mathrm{N}(0, \, \sigma_\epsilon^2), \, \text{Inv-}\chi^2(\nu_i, \, s_j^2)$$

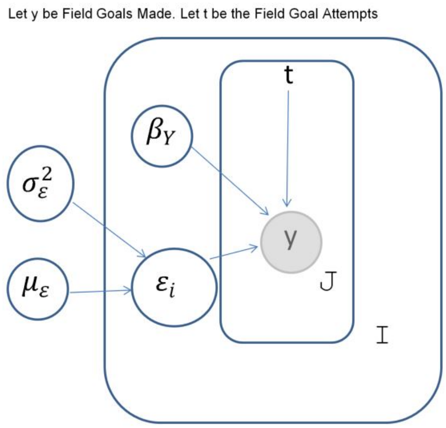


Figure 1: I represents the players(groups) and J the years

DAG Model

Code

JAGS model. I will use scaled-t1 on coefficients in beta. and a flat uniform distribution for sigma of player effect

```
data {
  dimY <- dim(FGM)
}
model {
  for (i in 1:dimY[1]) { ## row per player; total 8 players

    for (j in 1:dimY[2]) { ## column per year; total 3 years i.e. 2017, 2016, 2015

      FGM[i,j] ~ dbin(prob[i,j], FGA[i,j])

      logit(prob[i,j]) <- beta.Year[i]*Yr.Exper[i,j]+Player.Effect[i]

      FGMrep[i,j] ~ dbin(prob[i,j],FGA[i,j])
    }


      beta.Year[i] ~ dt(0,0.16,1)

      Player.Effect[i] ~ dnorm(mu, 1/sigmaPE^2)
  }

  mu ~ dt(0,0.01,1)

  sigmaPE ~ dunif(0,100)

}
```

Computation

Prepare data binding

Subset out the FGM(field goal made),FGA(field goal attempt),Yr.Exper(Years of professional experience)

```
d1 <- list(FGM = model_data[, get_column_names("FG", YEAR_COUNT)],
           FGA = model_data[,get_column_names("FGA", YEAR_COUNT)],
           Yr.Exper = model_data[,get_column_names("EXP", YEAR_COUNT)])

inits1 <- list(list(beta.Year=rep(-1, model_data_row_count), mu=10, sigmaPE=0.001, .RNG.name = "base::Ma
               list(beta.Year=rep(-1, model_data_row_count), mu=-10, sigmaPE=99, .RNG.name = "base::Mars
               list(beta.Year=rep(-1, model_data_row_count), mu=10, sigmaPE=99, .RNG.name = "base::Marsa
               list(beta.Year=rep(-1, model_data_row_count), mu=-10, sigmaPE=0.01, .RNG.name = "base::Ma
```

Build model

```
m1 <- jags.model('model-logistic.bug', d1, inits = inits1, n.chains = 4, n.adapt = 1000)
```

```
## Compiling data graph
##    Resolving undeclared variables
##    Allocating nodes
##    Initializing
##    Reading data back into data table
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 30
##    Unobserved stochastic nodes: 52
##    Total graph size: 291
##
## Initializing model
```

Burning and check convergence

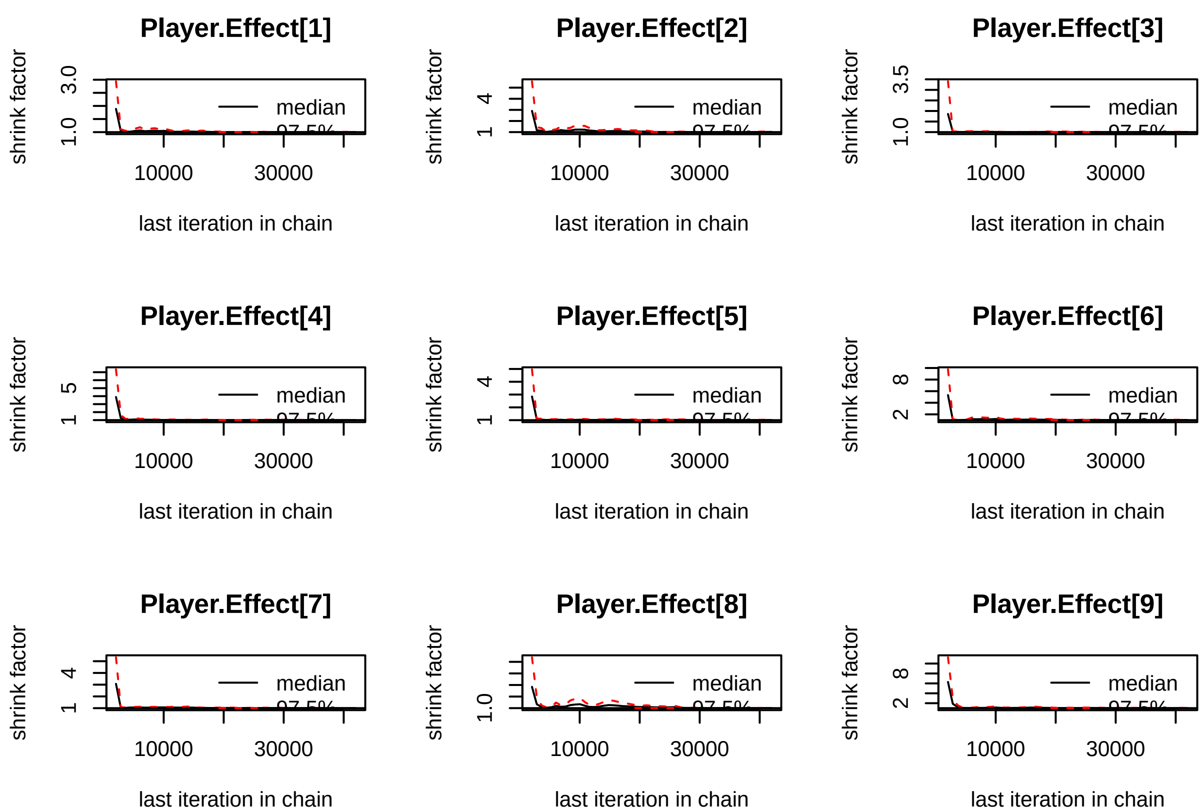
We tried various values for to speed convergence. None lead to very fast convergence but the above values, after much trial and error, were finally acceptable. Still it took a burn-in of over 1 million iterations to get convergence.

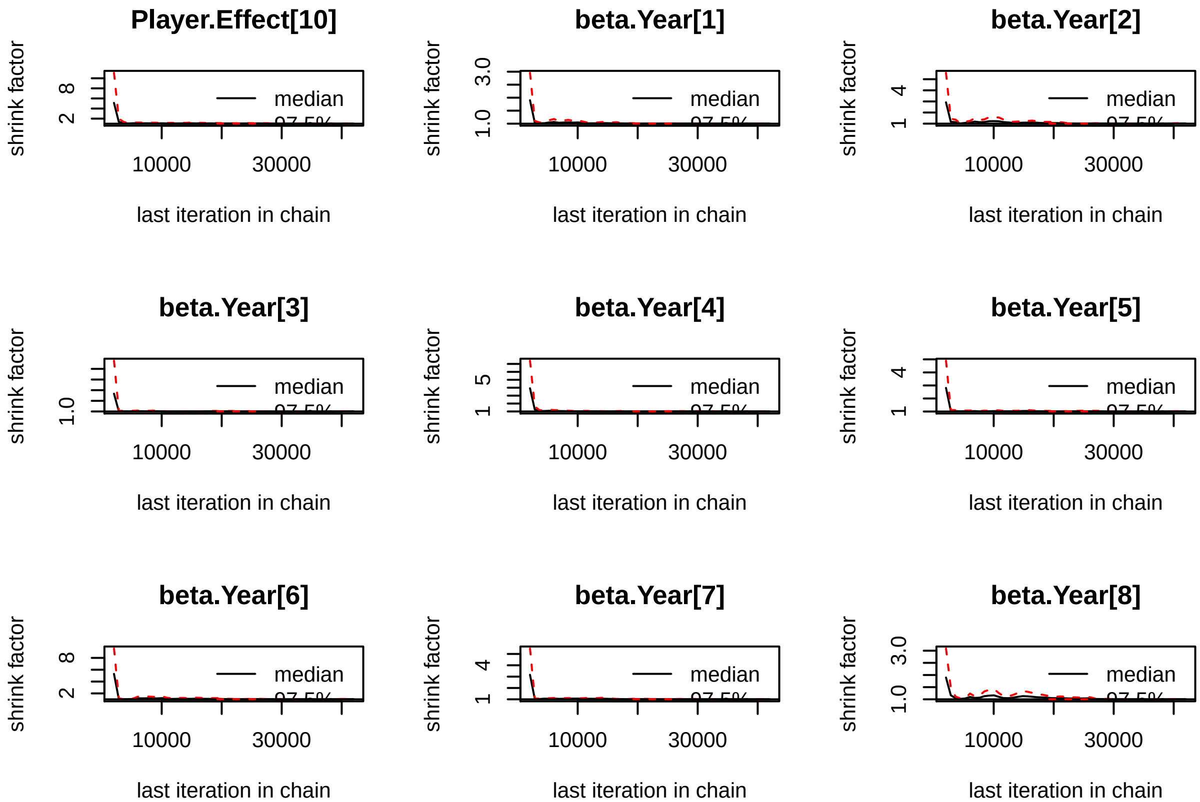
```
#update(m1, 1024000)
```

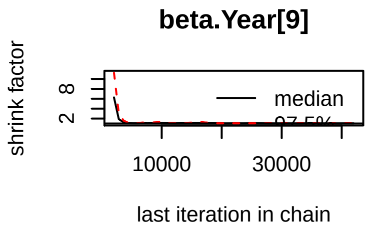
Posterior samples and Gelman Statistic

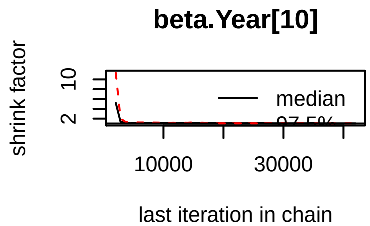
```
## Potential scale reduction factors:
##
##                  Point est. Upper C.I.
## Player.Effect[1]       1.00       1.01
## Player.Effect[2]       1.01       1.03
## Player.Effect[3]       1.00       1.00
## Player.Effect[4]       1.00       1.01
```

```
## Player.Effect[5]        1.01       1.02
## Player.Effect[6]        1.01       1.02
## Player.Effect[7]        1.01       1.02
## Player.Effect[8]        1.01       1.02
## Player.Effect[9]        1.01       1.04
## Player.Effect[10]       1.01       1.03
## beta.Year[1]            1.00       1.01
## beta.Year[2]            1.01       1.03
## beta.Year[3]            1.00       1.00
## beta.Year[4]            1.00       1.01
## beta.Year[5]            1.01       1.02
## beta.Year[6]            1.01       1.02
## beta.Year[7]            1.01       1.02
## beta.Year[8]            1.01       1.02
## beta.Year[9]            1.01       1.04
## beta.Year[10]           1.01       1.03
##
## Multivariate psrf
##
## 1.02
```




Player.Effect[10]
beta.Year[1]
beta.Year[2]
beta.Year[3]
beta.Year[4]
beta.Year[5]
beta.Year[6]
beta.Year[7]
beta.Year[8]
shrink factor
median
10000
30000
last iteration in chain





Effective samples sizes are adequate.

```
effectiveSize(x1)
```

```
##  Player.Effect[1]  Player.Effect[2]  Player.Effect[3]  Player.Effect[4]
##         2247.5863          631.9225         3640.0207         1317.4515
##  Player.Effect[5]  Player.Effect[6]  Player.Effect[7]  Player.Effect[8]
##         1492.5077          474.9117          974.4967         1238.4854
##  Player.Effect[9] Player.Effect[10]      beta.Year[1]      beta.Year[2]
##          642.6426          904.5053         2251.8550          635.6379
##      beta.Year[3]      beta.Year[4]      beta.Year[5]      beta.Year[6]
##         3665.9050         1314.1618         1483.9925          476.5658
##      beta.Year[7]      beta.Year[8]      beta.Year[9]     beta.Year[10]
##          994.4154         1288.3240          647.3108          928.8591
```

Retrieve replicate dataset and probabilities

```
x1 <- coda.samples(m1, c('beta.Year','Player.Effect','prob','FGMrep'), n.iter = 40000, thin=40)
```

Model Assessment

Check overdispersion, chi-square discrepancy

```
Tchi <- matrix(NA, nrow(FGMrep), model_data_row_count * YEAR_COUNT)
Tchirep <- matrix(NA, nrow(FGMrep), model_data_row_count * YEAR_COUNT)
for (s in 1:nrow(FGMrep)){
  Tchi[s,] <- sum( (FGM.v - FGA.v * probs[s,])^2 / (FGA.v * probs[s,] * (1-probs[s,])) )
  Tchirep[s,] <- sum( (FGMrep[s,] - FGA.v * probs[s,])^2 / (FGA.v * probs[s,] * (1-probs[s,])) )
}
```

No over dispersion problem as 0.483.

Results

Density of Various Player through the Years

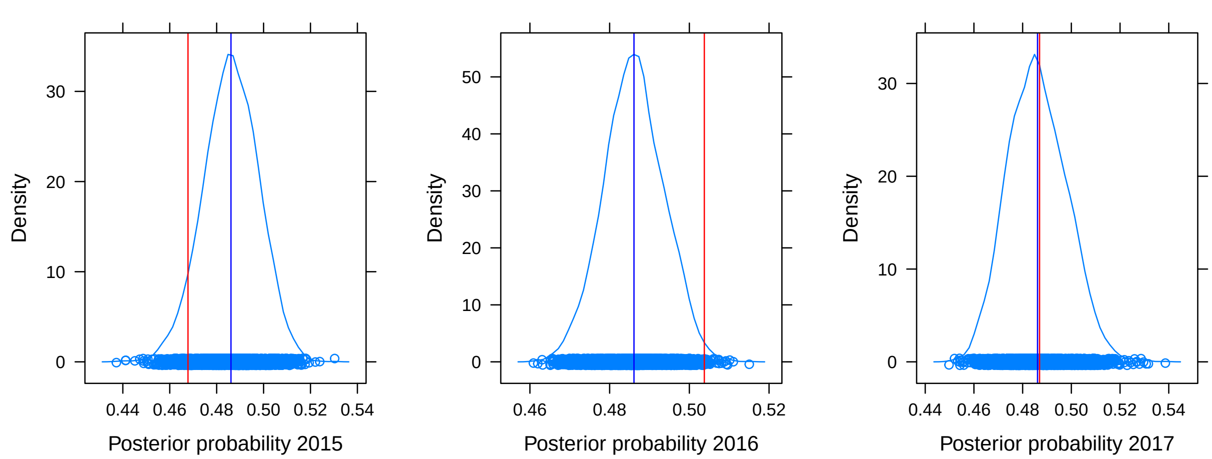
If we look as Steven Curry's density plots we see no improvement in his performance over the 3 years examined. This was the case with most of our players

```
player_row_id <- which(model_data$Player == "Stephen Curry")
# posterior prob
posterior <- get_player_posterior_probs(df, model_data, player_row_id)
df_posterior_observed <- get_player_posterior_vs_observed(model_data, player_row_id, posterior)
df_posterior_observed
```

```
##              posterior  observed
## YEAR_0       0.4861095 0.4677755
## YEAR_PRIOR_1 0.4861107 0.5037547
## YEAR_PRIOR_2 0.4861212 0.4869500
```

The posterior density does not show Stephen's improvement of making a field goal, let's also check Russell Westbrook.

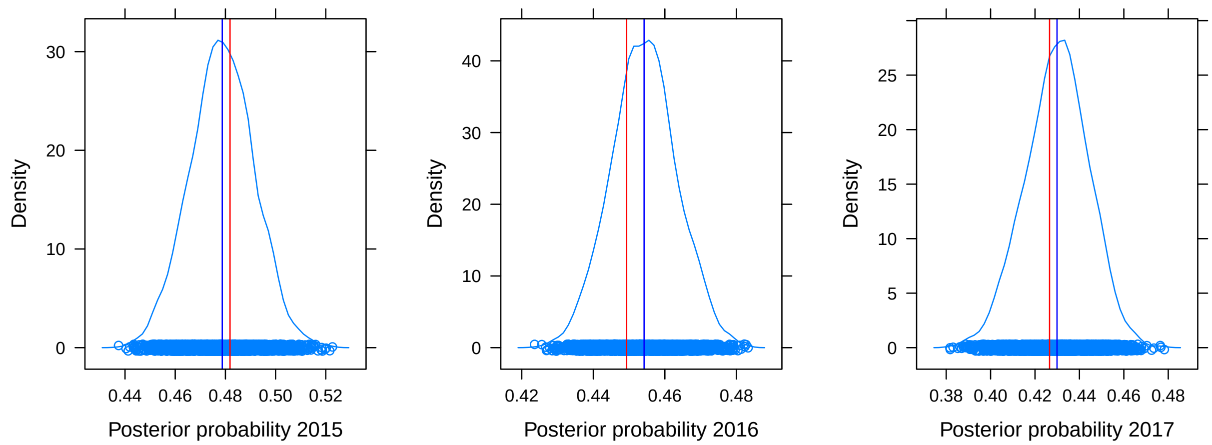
```
plot_player_posterior_probs(model_data, player_row_id, posterior)
```



Check Bradley Beal successfully makes an attempted field goal for the past three years.

```
##               posterior  observed
## YEAR_0        0.4787472 0.4818457
## YEAR_PRIOR_1  0.4541963 0.4493116
## YEAR_PRIOR_2  0.4299079 0.4265570
```

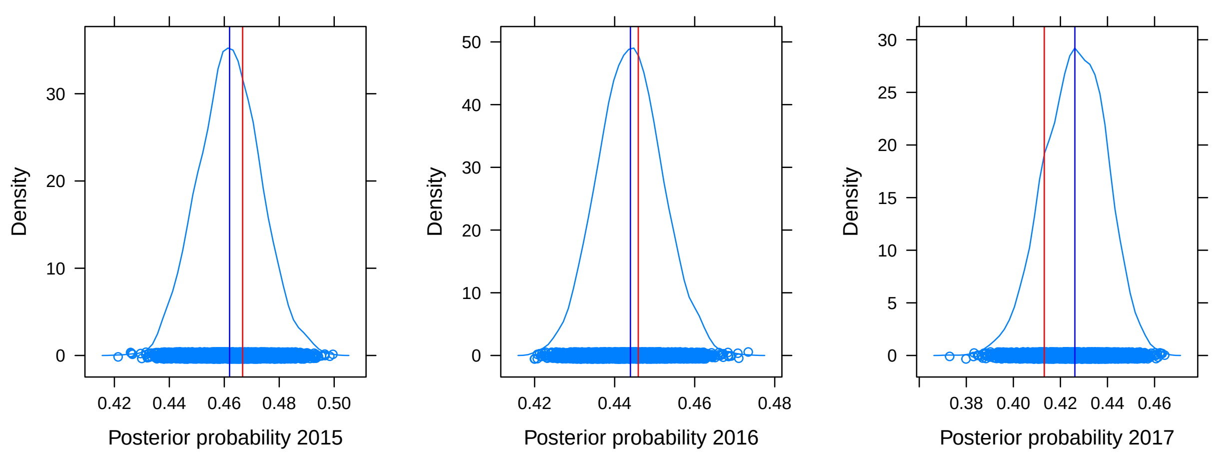
```
plot_player_posterior_probs(model_data, player_row_id, posterior)
```



Check DeMar DeRozan successfully makes an attempted field goal for the past three years.

```
##               posterior  observed
## YEAR_0        0.4619182 0.4666667
## YEAR_PRIOR_1  0.4439329 0.4458969
## YEAR_PRIOR_2  0.4261359 0.4131313
```

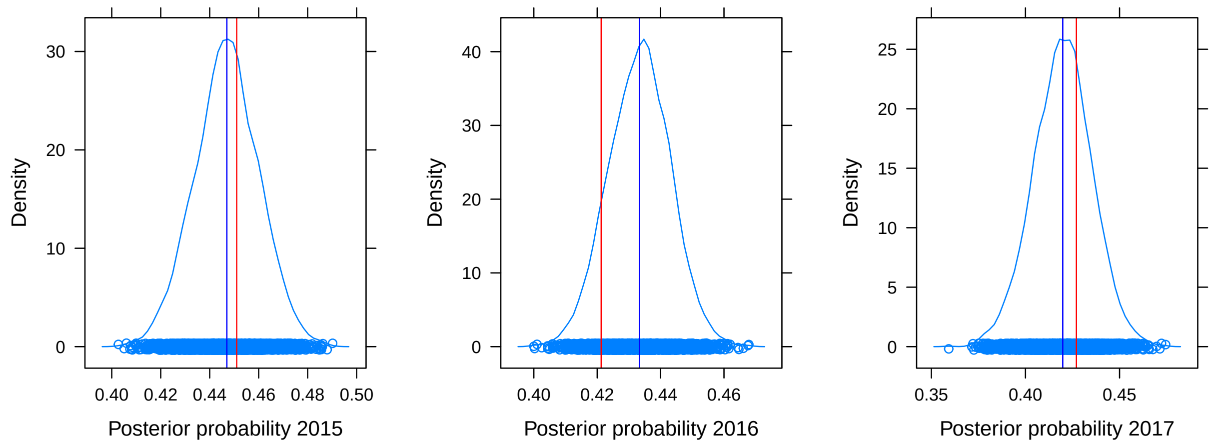
```
plot_player_posterior_probs(model_data, player_row_id, posterior)
```



Check Dennis Schroder successfully makes an attempted field goal for the past three years.

```
##               posterior  observed
## YEAR_0        0.4469985 0.4510288
## YEAR_PRIOR_1  0.4333218 0.4212548
## YEAR_PRIOR_2  0.4198114 0.4270677
```

```r
plot_player_posterior_probs(model_data, player_row_id, posterior)
```



Posterior Odds

Here we show the posterior odds of each player improving from one year to the next. We take our poster sample for each player, and take the mean of comparing one year's vector being greater than the previous. As we can see the odds are not extreme that a player may improve from one year to the next, nor do we have a clear pattern. The model does not support the proposition that players improve from one year to another.

Player	2016-2017	2015-2016
Bradley Beal	0.99575	0.00425
DeMar DeRozan	0.98900	0.01100
Dennis Schroder	0.89925	0.10075
John Wall	0.76550	0.23450
Klay Thompson	0.76450	0.23550
Kyle Lowry	0.95425	0.04575
Mike Conley	0.83525	0.16475
Shaun Livingston	0.95900	0.04100
Stephen Curry	0.52075	0.47925
Tony Allen	0.62700	0.37300

Contributions

Xiang deserves a bulk of the credit as the idea was his and did the data gathering and model design, as well a first pass at much of the analysis. We all contributed to the final analysis, although Nilesh did much to improve the R coding. Jerry also contributed to the analysis and lead on much of the early work with regards to putting together the proposal and video presenation with the team's input.

References

Appendix

- NBA Dataset
- Basketball reference glossary