BDA - Final Project: Batting Average analysis in Baseball by Bayesian Analysis

Contents

1. Introduction

In our project, we are going to use bayesian analysis to select excellent baseball players according to battingg averages. We will use Empirical Bayes estimation, given prior parameters, pooled model and hierachical model.

2. Problem Background

We are going to find better baseball players based on their batting average (BA). In baseball, the *batting average (BA)* is the number of hits divided by at bats.

$$BA = \frac{hits}{bats} = \frac{H}{AB}$$

3. Data Description

First, let's load the data. We'll use the *Batting* dataset from the excellent Lahman package. The dataset provides the tables from the 'Sean Lahman Baseball Database' as a set of R data.frames. It uses the data on pitching, hitting and fielding performance and other tables from 1871 through 2018, as recorded in the 2019 version of the database.

Here, we are going to do some data cleaning and processing and will focus on the numbers of hits and the numbers of bats.

```
library(dplyr)
library(tidyr)
library(Lahman)


career <- Batting %>%
  filter(AB > 0) %>%
  anti_join(Pitching, by = "playerID") %>%
  group_by(playerID) %>%
  summarize(H = sum(H), AB = sum(AB)) %>%
  mutate(average = H / AB)

# use names along with the player IDs
career <- Master %>%
  tbl_df() %>%
  select(playerID, nameFirst, nameLast) %>%
  unite(name, nameFirst, nameLast, sep = " ") %>%
  inner_join(career, by = "playerID") %>%
  select(-playerID)

data=career
data
```

```
## # A tibble: 9,509 x 4
##    name                  H    AB average
##    <chr>             <int> <int>   <dbl>
##  1 Hank Aaron         3771 12364  0.305
##  2 Tommie Aaron        216   944  0.229
##  3 Andy Abad             2    21  0.0952
##  4 John Abadie          11    49  0.224
##  5 Ed Abbaticchio      772  3044  0.254
##  6 Fred Abbott         107   513  0.209
##  7 Jeff Abbott         157   596  0.263
##  8 Kurt Abbott         523  2044  0.256
##  9 Ody Abbott           13    70  0.186
## 10 Frank Abercrombie     0     4  0
## # ... with 9,499 more rows
```

```
library(dplyr)
library(tidyr)
library(Lahman)


career <- Batting %>%
  filter(AB > 0) %>%
  anti_join(Pitching, by = "playerID") %>%
  group_by(playerID) %>%
  summarize(H = sum(H), AB = sum(AB)) %>%
  mutate(average = H / AB)
```

```
# use names along with the player IDs
career <- Master %>%
  tbl_df() %>%
  select(playerID, nameFirst, nameLast) %>%
  unite(name, nameFirst, nameLast, sep = " ") %>%
  inner_join(career, by = "playerID") %>%
  select(-playerID)

data=career
data
```

```
## # A tibble: 9,509 x 4
##    name                   H    AB average
##    <chr>              <int> <int>   <dbl>
##  1 Hank Aaron          3771 12364  0.305
##  2 Tommie Aaron         216   944  0.229
##  3 Andy Abad              2    21  0.0952
##  4 John Abadie           11    49  0.224
##  5 Ed Abbaticchio       772  3044  0.254
##  6 Fred Abbott          107   513  0.209
##  7 Jeff Abbott          157   596  0.263
##  8 Kurt Abbott          523  2044  0.256
##  9 Ody Abbott            13    70  0.186
## 10 Frank Abercrombie      0     4  0
## # ... with 9,499 more rows
```

Image you are a baseball recruiter trying to find some players with good performances in the batting averages. Are you going to choose Hank Aaron with a BA at 0.400 or Harry Atkinson with a BA at 0.305? I bet almost everyone will choose Hank Aaron because the high BA of Hank might be due to luck. Hank, on the other hand, has a lot of evidence that he's an high-average batter.

```
Harry=data[ which(data$name=='Harry Atkinson'),]
Hank=data[ which(data$name=='Hank Aaron'),]
rbind(Harry,Hank)
```

```
## # A tibble: 2 x 4
##   name                H    AB average
##   <chr>           <int> <int>   <dbl>
## 1 Harry Atkinson      2     5   0.4
## 2 Hank Aaron       3771 12364   0.305
```

In our project, we are going to use Bayes methods to estimate the proportion of success, starting from Empirical Bayes Estimation to standard bayes anlysis and Hierarchical Model.

4. Model

4.1 Empirical Bayes estimation

Empirical Bayes methods are procedures for statistical inference in which the prior distribution is estimated from the data. We have different way to estimate the prior distribution. The first could be from the mean and variance of the batting average we have.

From the formulas of μ and σ, we can get the expression of α and β by μ and σ

$$\mu = \frac{\alpha}{\beta}$$

$$\sigma^2 = \frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)}$$

$$\alpha = (\frac{1-\mu}{\sigma^2} - \frac{1}{\mu})\mu^2$$

$$\beta = \alpha(\frac{1}{\mu} - 1)$$

Another way is maximum likelihood. We also choose to use this method in our project. We used the fitdistr function (fitdistr: Maximum-Likelihood Fitting Of Univariate Distributions) from ASS to calculate the prior alpha and beta.

Considering the outliers which have very few batting counts, we fliter the data whose AB is no less than 1000.

```
data_filtered <- career %>%
    filter(AB >=1000)

m <- MASS::fitdistr(data_filtered$average, dbeta,
                    start = list(shape1 = 1, shape2 = 1))

alpha0 <- m$estimate[1]
beta0 <- m$estimate[2]
cat("\nThe alpha0 is: ", alpha0)
```

```
##
## The alpha0 is:  96.8
```

```
cat("\nThe beta0 is: ", beta0)
```

```
##
## The beta0 is:  270
```

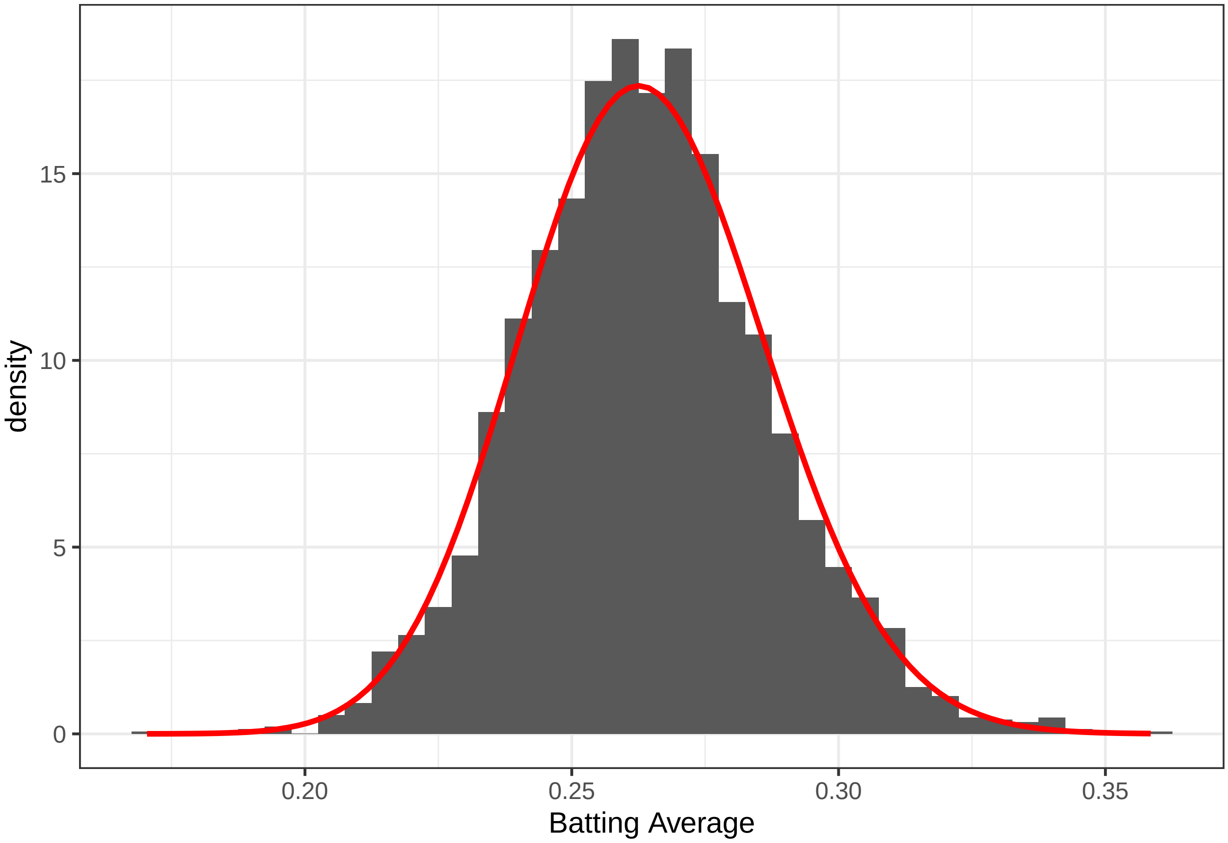
This comes up with

$$\alpha_0 = 96.83$$

and

$$\beta_0 = 270.186$$

. And we find the this prior parameters fit very well.



Having this beta prior distribution, it is easy to update our estimate batting average by using the following formula:

$$BA\_estimate = \frac{H + alpha0}{AB + alpha0 + beta0}$$

```
data_ba <- data %>%
    mutate(ba_estimate = (H + alpha0) / (AB + alpha0 + beta0))
head(data_ba)
```

```
## # A tibble: 6 x 5
##   name                H    AB average ba_estimate
##   <chr>           <int> <int>   <dbl>       <dbl>
## 1 Hank Aaron       3771 12364  0.305        0.304
## 2 Tommie Aaron      216   944  0.229        0.239
## 3 Andy Abad           2    21  0.0952       0.255
## 4 John Abadie        11    49  0.224        0.259
## 5 Ed Abbaticchio    772  3044  0.254        0.255
## 6 Fred Abbott       107   513  0.209        0.232
```

Posterior predictive checking Therefore, let's recap the previous case. The estiated BA of Harry is lower than Hank's although his actual average is higher. We will choose to recruit Hank according to the result.

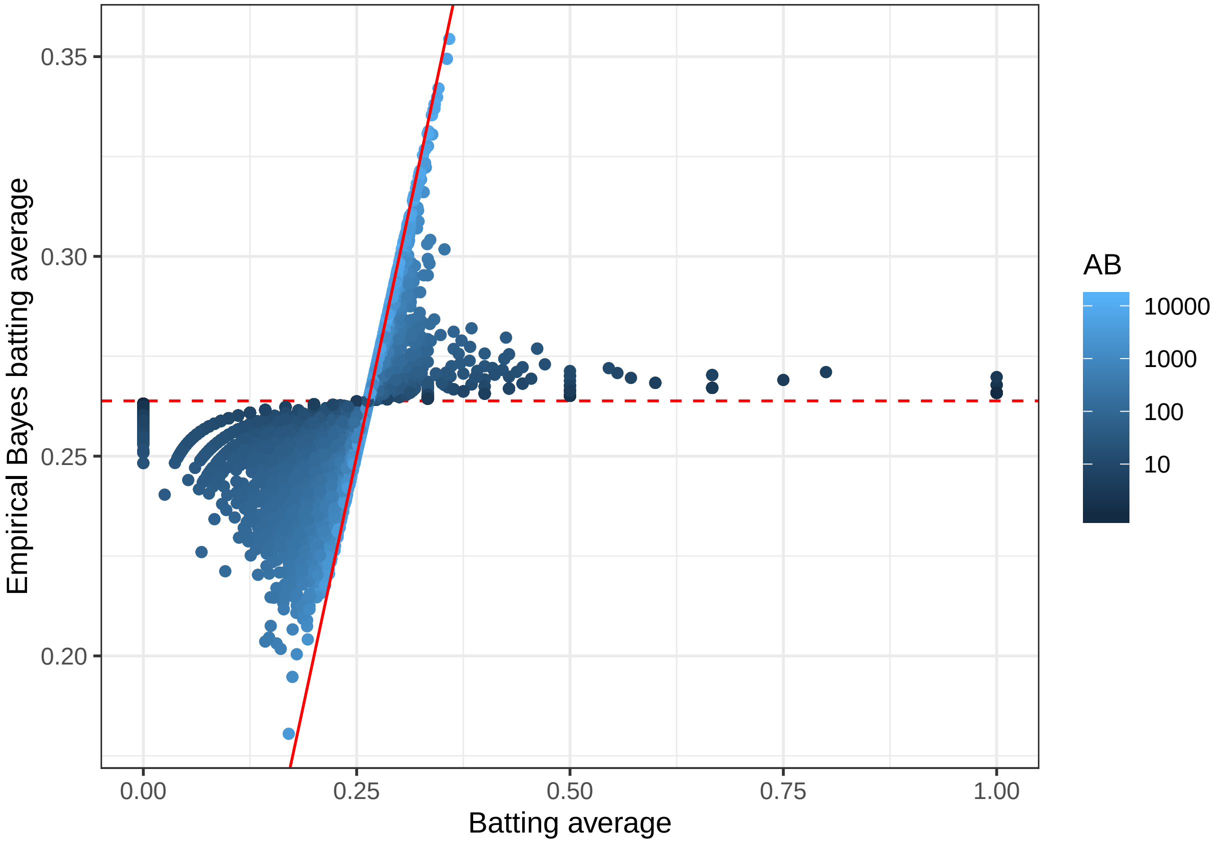
```
Harry=data_ba[ which(data_ba$name=='Harry Atkinson'),]
Hank=data_ba[ which(data_ba$name=='Hank Aaron'),]
rbind(Harry,Hank)
```

```
## # A tibble: 2 x 5
##   name                H    AB average ba_estimate
##   <chr>           <int> <int>   <dbl>       <dbl>
## 1 Harry Atkinson     2     5   0.4          0.266
## 2 Hank Aaron      3771 12364   0.305        0.304
```

Finally, let's find the best 10 players according to our analysis.

```
head(data_ba[order(-data_ba$ba_estimate),],10)
```

```
## # A tibble: 10 x 5
##    name                     H    AB average ba_estimate
##    <chr>                <int> <int>   <dbl>       <dbl>
##  1 Rogers Hornsby        2930  8173   0.358       0.354
##  2 Shoeless Joe Jackson  1772  4981   0.356       0.349
##  3 Ed Delahanty          2596  7505   0.346       0.342
##  4 Billy Hamilton        2158  6268   0.344       0.340
##  5 Willie Keeler         2932  8591   0.341       0.338
##  6 Harry Heilmann        2660  7787   0.342       0.338
##  7 Bill Terry            2193  6428   0.341       0.337
##  8 Lou Gehrig            2721  8001   0.340       0.337
##  9 Nap Lajoie            3242  9589   0.338       0.335
## 10 Tony Gwynn            3141  9288   0.338       0.335
```



This figure shows the shrinkage in Empirical Bayes estimation. If we have no evidence at all, we guess his/her BA is on the horizontal dashed read line

$$y = \frac{\alpha_0}{\alpha_0 + \beta_0} = 0.264$$

While, with the increasing of AB, the BA tend to update and shrink to the red line

$$x = y$$

. All the estimates will move towards the average. How much it moves these estimates depends on how much evidence we have.

4.2 Standard Bayesian Method

4.2.1 Given prior parameters from fact

In the last section, we use the data distribution as prior distribution. Here, let's give our prior parameters by ourselves. According to the statistic of batting gaming data, the average is around 0.266 from 0.21 to 0.35. So we just choose $\alpha = 81, \beta = 219$ to fit this requirement. We can find the result is similar but slightly different. And if we choose some improper prior such as uniform distribution, the result would have larger differences especially for those whose AB is low.

```
alpha0=81
beta0=219
data_ba <- data %>%
    mutate(ba_estimate = (H + alpha0) / (AB + alpha0 + beta0))
head(data_ba[order(-data_ba$ba_estimate),],10)
```

```
## # A tibble: 10 x 5
##    name                     H    AB average ba_estimate
##    <chr>                <int> <int>   <dbl>       <dbl>
##  1 Rogers Hornsby        2930  8173   0.358       0.355
##  2 Shoeless Joe Jackson  1772  4981   0.356       0.351
##  3 Ed Delahanty          2596  7505   0.346       0.343
##  4 Billy Hamilton        2158  6268   0.344       0.341
##  5 Harry Heilmann        2660  7787   0.342       0.339
##  6 Willie Keeler         2932  8591   0.341       0.339
##  7 Bill Terry            2193  6428   0.341       0.338
##  8 Lou Gehrig            2721  8001   0.340       0.338
##  9 Tony Gwynn            3141  9288   0.338       0.336
## 10 Nap Lajoie            3242  9589   0.338       0.336
```

4.2.2 pool model

```
stan_model1<-'
data {
  int<lower=0> N; // number of palyers
  vector<lower=0.0001>[N] theta;
}

parameters {
  real<lower=0,upper=1> phi;
  real<lower=0.1> lambda;
}

transformed parameters {
  real<lower=0.0001> alpha = lambda * phi;
  real<lower=0.0001> beta = lambda * (1 - phi);
}

model {
  lambda ~ pareto(0.1, 1.5);
  for (n in 1:N)
    theta[n] ~ beta(alpha, beta);
}
'
```

```
data_model1<- list(
  N=length(data_filtered$H),
  theta=data_filtered$average
)
fit <- stan(model_code = stan_model1, data = data_model1)
extract_model1 <- rstan::extract(fit, permuted = T)
```

```
N=length(data_filtered$H)
alpha_pool=mean(extract_model1$alpha[2001:N])
beta_pool=mean(extract_model1$beta[2001:N])
data_ba <- data %>%
    mutate(ba_estimate = (H + alpha_pool) / (AB + alpha_pool + beta_pool))
head(data_ba[order(-data_ba$ba_estimate),],10)
```

```
## # A tibble: 10 x 5
##    name                      H    AB average ba_estimate
##    <chr>                 <int> <int>   <dbl>       <dbl>
##  1 Rogers Hornsby         2930  8173   0.358       0.354
##  2 Shoeless Joe Jackson   1772  4981   0.356       0.349
##  3 Ed Delahanty           2596  7505   0.346       0.342
##  4 Billy Hamilton         2158  6268   0.344       0.340
##  5 Willie Keeler          2932  8591   0.341       0.338
##  6 Harry Heilmann         2660  7787   0.342       0.338
##  7 Bill Terry             2193  6428   0.341       0.337
##  8 Lou Gehrig             2721  8001   0.340       0.337
##  9 Nap Lajoie             3242  9589   0.338       0.335
## 10 Tony Gwynn             3141  9288   0.338       0.335
```

4.2.3 hierachical model

```
stan_model2<-'
data {
  int<lower=0> N; // number of players
  vector<lower=0.0001>[N] theta;
}

parameters {
  vector<lower=0,upper=1>[N] phi;
  real<lower=0.1> lambda;
}

transformed parameters {
  vector<lower=0.0001>[N] alpha = lambda * phi;
  vector<lower=0.0001>[N] beta = lambda * (1 - phi);
}

model {
  lambda ~ pareto(0.1, 1.5);
  for (n in 1:N)
    theta[n] ~ beta(alpha[n], beta[n]);
}
'
```

```
data_model2<- list(
  N=length(data_filtered$H),
  theta=data_filtered$average)

fit2 <- stan(model_code = stan_model2, data = data_model2)
```

```
extract_model2 <- rstan::extract(fit2, permuted = T)
```

```
alpha_hierachical=apply(extract_model2$alpha[2001:4000,],2,mean)
beta_hierachical=apply(extract_model2$beta[2001:4000,],2,mean)
data_ba <- data_filtered %>%
    mutate(alpha_hierachical) %>%
    mutate(beta_hierachical) %>%
    mutate(ba_estimate = (H + alpha_hierachical) / (AB + alpha_hierachical + beta_hierachical))
head(data_ba[order(-data_ba$ba_estimate),],10)
```

```
## # A tibble: 10 x 7
##    name       H    AB average alpha_hierachic~ beta_hierachical ba_estimate
##    <chr>  <int> <int>   <dbl>            <dbl>            <dbl>       <dbl>
##  1 Roger~  2930  8173   0.358             656.            1172.       0.359
##  2 Shoel~  1772  4981   0.356             651.            1177.       0.356
##  3 Ed De~  2596  7505   0.346             633.            1195.       0.346
##  4 Billy~  2158  6268   0.344             630.            1198.       0.344
##  5 Harry~  2660  7787   0.342             624.            1204.       0.342
##  6 Willi~  2932  8591   0.341             624.            1204.       0.341
##  7 Bill ~  2193  6428   0.341             624.            1204.       0.341
##  8 Lou G~  2721  8001   0.340             622.            1206.       0.340
##  9 Jake ~  1024  3024   0.339             619.            1209.       0.339
## 10 Tony ~  3141  9288   0.338             618.            1210.       0.338
```

5. Analysis of the results and model comparison

Rhat is used to estimate the true convergence of the chains. The Rhat is used to evaluate the within-variance in each chain and the between-variance in different chains. Rhat is defined as:

$$R = \sqrt{\frac{var^+}{W}}$$

From the Rhat print by model itself, it is noticed that Rhat of alpha and beta are smaller than 1. Thus, the model converge and the generated samples of alpha and beta are safe to use.

5.1 Convergence diagnostics (Rhat, divergences, ESS)

5.1.1 convergence of pool model

```r
print(fit)
```

```
## Inference for Stan model: ef5321c48cbbbfb826d214e00a5fbc93.
## 4 chains, each with iter=2000; warmup=1000; thin=1;
## post-warmup draws per chain=1000, total post-warmup draws=4000.
##
##           mean se_mean   sd    2.5%     25%     50%     75%   97.5% n_eff
## phi       0.26    0.00 0.00    0.26    0.26    0.26    0.26    0.26  3840
## lambda  366.30    0.25 9.22  348.89  359.84  366.24  372.37  385.04  1404
## alpha    96.64    0.06 2.43   92.06   94.94   96.63   98.22  101.57  1395
## beta    269.66    0.18 6.80  256.82  264.90  269.63  274.19  283.44  1409
## lp__   7483.53    0.02 1.03 7480.80 7483.11 7483.85 7484.27 7484.55  1705
##        Rhat
## phi       1
## lambda    1
## alpha     1
## beta      1
## lp__      1
##
## Samples were drawn using NUTS(diag_e) at Sun Dec  8 22:34:04 2019.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
```

Both bulk-ESS and tail-ESS should be at least 100 (approximately) per Markov Chain in order to be reliable and indicate that estimates of respective posterior quantiles are reliable. Rhat and effective sample size n_eff are included in the fitting results as above. Rhat here are all 1 which means we can believe the model converges well.

5.1.2 convergence of hierachical model

```r
t1<-extract_model2$alpha[1:1000,]
t2<-extract_model2$alpha[1001:2000,]
t3<-extract_model2$alpha[2001:3000,]
t4<-extract_model2$alpha[3001:4000,]
alpha_post_warmup<-matrix(c(t1[1],t2[1],t3[1],t4[1]),1000,4)
cat("the Rhat of alpha in 4 chains is:",Rhat(alpha_post_warmup), "\n")
```

```
## the Rhat of alpha in 4 chains is: 0.999
```

```r
cat("the EFF of alpha in 4 chains is:",ess_bulk(alpha_post_warmup))
```

```
## the EFF of alpha in 4 chains is: 14408
```

```r
t1<-extract_model2$beta[1:1000,]
t2<-extract_model2$beta[1001:2000,]
t3<-extract_model2$beta[2001:3000,]
t4<-extract_model2$beta[3001:4000,]
beta_post_warmup<-matrix(c(t1[1],t2[1],t3[1],t4[1]),1000,4)
cat("the Rhat of beta in 4 chains is:",Rhat(beta_post_warmup),"\n")
```

```
## the Rhat of beta in 4 chains is: 0.999
```

```r
cat("the EFF of alpha in 4 chains is:",ess_bulk(beta_post_warmup))
```

```
## the EFF of alpha in 4 chains is: 14408
```

Both bulk-ESS and tail-ESS should be at least 100 (approximately) per Markov Chain in order to be reliable and indicate that estimates of respective posterior quantiles are reliable. Besides, Rhat and effective sample size n_eff are also included in the fitting results as above. Rhat here are all $1 < 1.05$ which means we can believe the model converges well.

5.2 Posterior predictive checking

The posterior predictive checking are mentioned in the part 3 and 4.

6. Sensitivity analysis with respect to prior choices

6.1 Sensitivity analysis of randomly initialized prior

The sensitivity of posterior inference about BA estimation to the proposed prior distribution is exhibited in Table. The first column *phi* is the parameter of the prior distribution $\alpha + \beta$. The second column is the mean of prior distribution *lambda*, which is calculated by $\alpha/\alpha + \beta$. Here we fixed *lambda*, which is the mean of prior distribution.

Here, we choose 3 different values of $\alpha + \beta$ (*phi*): $phi1$, $phi2$, $phi3$, and hope to find the trend variation of posterior inference about BA estimation, which are corresponding to $ba_estimate$, $ba_estimate1$ and $ba_estimate2$.

The following three tables are ordered by these three difefrent BA estimation: $ba_estimate$, $ba_estimate1$ and $ba_estimate2$.

Then we can find, as $\alpha + \beta$ (*phi*) is growing, the estimation of posterior is more closely to the prior *lambda*:

```r
alpha0=81
beta0=219
phi=alpha0+beta0
lambda=alpha0/phi
alpha1=alpha0*2
beta1=beta0*2
alpha2=alpha0*10
beta2=beta0*10
data_ba <- data %>%
    mutate(ba_estimate = (H + alpha_pool) / (AB + alpha_pool + beta_pool))
data_ba <- data_ba %>%
    mutate(ba_estimate1 = (H + alpha1) / (AB + alpha1 + beta1))
data_ba <- data_ba %>%
    mutate(ba_estimate2 = (H + alpha2) / (AB + alpha2 + beta2))
data_ba <- data_ba %>%
    mutate(phi = phi)
data_ba <- data_ba %>%
    mutate(phi1 = phi*2)
data_ba <- data_ba %>%
    mutate(phi2 = phi*10)
data_ba <- data_ba %>%
    mutate(lambda = lambda)
head(data_ba[order(-data_ba$ba_estimate),],10)
```

```
## # A tibble: 10 x 11
##    name      H    AB average ba_estimate ba_estimate1 ba_estimate2   phi
##    <chr> <int> <int>   <dbl>       <dbl>        <dbl>        <dbl> <dbl>
##  1 Roge~  2930  8173   0.358       0.354        0.352        0.335   300
##  2 Shoe~  1772  4981   0.356       0.349        0.347        0.324   300
##  3 Ed D~  2596  7505   0.346       0.342        0.340        0.324   300
##  4 Bill~  2158  6268   0.344       0.340        0.338        0.320   300
##  5 Will~  2932  8591   0.341       0.338        0.337        0.323   300
##  6 Harr~  2660  7787   0.342       0.338        0.336        0.322   300
##  7 Bill~  2193  6428   0.341       0.337        0.335        0.319   300
##  8 Lou ~  2721  8001   0.340       0.337        0.335        0.321   300
##  9 Nap ~  3242  9589   0.338       0.335        0.334        0.322   300
## 10 Tony~  3141  9288   0.338       0.335        0.334        0.322   300
## # ... with 3 more variables: phi1 <dbl>, phi2 <dbl>, lambda <dbl>
```

```
head(data_ba[order(-data_ba$ba_estimate1),],10)
```

```
## # A tibble: 10 x 11
##    name      H    AB average ba_estimate ba_estimate1 ba_estimate2   phi
##    <chr> <int> <int>   <dbl>       <dbl>        <dbl>        <dbl> <dbl>
##  1 Roge~  2930  8173   0.358       0.354        0.352        0.335   300
##  2 Shoe~  1772  4981   0.356       0.349        0.347        0.324   300
##  3 Ed D~  2596  7505   0.346       0.342        0.340        0.324   300
##  4 Bill~  2158  6268   0.344       0.340        0.338        0.320   300
##  5 Will~  2932  8591   0.341       0.338        0.337        0.323   300
##  6 Harr~  2660  7787   0.342       0.338        0.336        0.322   300
##  7 Lou ~  2721  8001   0.340       0.337        0.335        0.321   300
##  8 Bill~  2193  6428   0.341       0.337        0.335        0.319   300
##  9 Nap ~  3242  9589   0.338       0.335        0.334        0.322   300
## 10 Tony~  3141  9288   0.338       0.335        0.334        0.322   300
## # ... with 3 more variables: phi1 <dbl>, phi2 <dbl>, lambda <dbl>
```

```
head(data_ba[order(-data_ba$ba_estimate2),],10)
```

```
## # A tibble: 10 x 11
##    name      H    AB average ba_estimate ba_estimate1 ba_estimate2   phi
##    <chr> <int> <int>   <dbl>       <dbl>        <dbl>        <dbl> <dbl>
##  1 Roge~  2930  8173   0.358       0.354        0.352        0.335   300
##  2 Ed D~  2596  7505   0.346       0.342        0.340        0.324   300
##  3 Shoe~  1772  4981   0.356       0.349        0.347        0.324   300
##  4 Will~  2932  8591   0.341       0.338        0.337        0.323   300
##  5 Nap ~  3242  9589   0.338       0.335        0.334        0.322   300
##  6 Harr~  2660  7787   0.342       0.338        0.336        0.322   300
##  7 Tony~  3141  9288   0.338       0.335        0.334        0.322   300
##  8 Lou ~  2721  8001   0.340       0.337        0.335        0.321   300
##  9 Bill~  2158  6268   0.344       0.340        0.338        0.320   300
## 10 Eddi~  3315  9949   0.333       0.331        0.330        0.319   300
## # ... with 3 more variables: phi1 <dbl>, phi2 <dbl>, lambda <dbl>
```

6.2 Sensitivity analysis of pooled model prior

```
phi=alpha_pool+beta_pool
lambda=alpha_pool/phi
alpha1=alpha_pool*2
beta1=beta_pool*2
```

```
alpha2=alpha_pool*10
beta2=beta_pool*10
data_ba <- data %>%
    mutate(ba_estimate = (H + alpha_pool) / (AB + alpha_pool + beta_pool))
data_ba <- data_ba %>%
    mutate(ba_estimate1 = (H + alpha1) / (AB + alpha1 + beta1))
data_ba <- data_ba %>%
    mutate(ba_estimate2 = (H + alpha2) / (AB + alpha2 + beta2))
data_ba <- data_ba %>%
    mutate(phi = phi)
data_ba <- data_ba %>%
    mutate(phi1 = phi*2)
data_ba <- data_ba %>%
    mutate(phi2 = phi*10)
data_ba <- data_ba %>%
    mutate(lambda = lambda)
head(data_ba[order(-data_ba$ba_estimate),],10)
```

```
## # A tibble: 10 x 11
##    name      H    AB average ba_estimate ba_estimate1 ba_estimate2   phi
##    <chr> <int> <int>   <dbl>       <dbl>        <dbl>        <dbl> <dbl>
##  1 Roge~  2930  8173   0.358       0.354        0.351        0.329  366.
##  2 Shoe~  1772  4981   0.356       0.349        0.344        0.317  366.
##  3 Ed D~  2596  7505   0.346       0.342        0.339        0.319  366.
##  4 Bill~  2158  6268   0.344       0.340        0.336        0.315  366.
##  5 Will~  2932  8591   0.341       0.338        0.335        0.318  366.
##  6 Harr~  2660  7787   0.342       0.338        0.335        0.317  366.
##  7 Bill~  2193  6428   0.341       0.337        0.333        0.313  366.
##  8 Lou ~  2721  8001   0.340       0.337        0.334        0.316  366.
##  9 Nap ~  3242  9589   0.338       0.335        0.333        0.318  366.
## 10 Tony~  3141  9288   0.338       0.335        0.333        0.317  366.
## # ... with 3 more variables: phi1 <dbl>, phi2 <dbl>, lambda <dbl>
```

```
head(data_ba[order(-data_ba$ba_estimate1),],10)
```

```
## # A tibble: 10 x 11
##    name      H    AB average ba_estimate ba_estimate1 ba_estimate2   phi
##    <chr> <int> <int>   <dbl>       <dbl>        <dbl>        <dbl> <dbl>
##  1 Roge~  2930  8173   0.358       0.354        0.351        0.329  366.
##  2 Shoe~  1772  4981   0.356       0.349        0.344        0.317  366.
##  3 Ed D~  2596  7505   0.346       0.342        0.339        0.319  366.
##  4 Bill~  2158  6268   0.344       0.340        0.336        0.315  366.
##  5 Will~  2932  8591   0.341       0.338        0.335        0.318  366.
##  6 Harr~  2660  7787   0.342       0.338        0.335        0.317  366.
##  7 Lou ~  2721  8001   0.340       0.337        0.334        0.316  366.
##  8 Bill~  2193  6428   0.341       0.337        0.333        0.313  366.
##  9 Nap ~  3242  9589   0.338       0.335        0.333        0.318  366.
## 10 Tony~  3141  9288   0.338       0.335        0.333        0.317  366.
## # ... with 3 more variables: phi1 <dbl>, phi2 <dbl>, lambda <dbl>
```

```
head(data_ba[order(-data_ba$ba_estimate2),],10)
```

```
## # A tibble: 10 x 11
##    name      H    AB average ba_estimate ba_estimate1 ba_estimate2   phi
##    <chr> <int> <int>   <dbl>       <dbl>        <dbl>        <dbl> <dbl>
##  1 Roge~  2930  8173   0.358       0.354        0.351        0.329  366.
##  2 Ed D~  2596  7505   0.346       0.342        0.339        0.319  366.
##  3 Will~  2932  8591   0.341       0.338        0.335        0.318  366.
##  4 Nap ~  3242  9589   0.338       0.335        0.333        0.318  366.
##  5 Tony~  3141  9288   0.338       0.335        0.333        0.317  366.
##  6 Shoe~  1772  4981   0.356       0.349        0.344        0.317  366.
##  7 Harr~  2660  7787   0.342       0.338        0.335        0.317  366.
##  8 Lou ~  2721  8001   0.340       0.337        0.334        0.316  366.
##  9 Bill~  2158  6268   0.344       0.340        0.336        0.315  366.
## 10 Eddi~  3315  9949   0.333       0.331        0.328        0.315  366.
## # ... with 3 more variables: phi1 <dbl>, phi2 <dbl>, lambda <dbl>
```

6.3 Sensitivity analysis of hierachical model prior

```
alpha_hierachical=apply(extract_model2$alpha[2001:4000,],2,mean)
beta_hierachical=apply(extract_model2$beta[2001:4000,],2,mean)
phi=alpha_hierachical+beta_hierachical
lambda=alpha_hierachical/phi
alpha1=alpha_hierachical*2
beta1=beta_hierachical*2
alpha2=alpha_hierachical*10
beta2=beta_hierachical*10
data_ba <- data_filtered %>%
    mutate(alpha_hierachical) %>%
    mutate(beta_hierachical) %>%
    mutate(ba_estimate = (H + alpha_hierachical) / (AB + alpha_hierachical + beta_hierachical))
data_ba <- data_ba %>%
    mutate(ba_estimate1 = (H + alpha1) / (AB + alpha1 + beta1))
data_ba <- data_ba %>%
    mutate(ba_estimate2 = (H + alpha2) / (AB + alpha2 + beta2))
data_ba <- data_ba %>%
    mutate(phi = phi)
data_ba <- data_ba %>%
    mutate(phi1 = phi*2)
data_ba <- data_ba %>%
    mutate(phi2 = phi*10)
data_ba <- data_ba %>%
    mutate(lambda = lambda)
head(data_ba[order(-data_ba$ba_estimate),],10)
```

```
## # A tibble: 10 x 13
##    name      H    AB average alpha_hierachic~ beta_hierachical ba_estimate
##    <chr> <int> <int>   <dbl>            <dbl>            <dbl>       <dbl>
##  1 Roge~  2930  8173   0.358             656.            1172.       0.359
##  2 Shoe~  1772  4981   0.356             651.            1177.       0.356
##  3 Ed D~  2596  7505   0.346             633.            1195.       0.346
##  4 Bill~  2158  6268   0.344             630.            1198.       0.344
##  5 Harr~  2660  7787   0.342             624.            1204.       0.342
##  6 Will~  2932  8591   0.341             624.            1204.       0.341
##  7 Bill~  2193  6428   0.341             624.            1204.       0.341
##  8 Lou ~  2721  8001   0.340             622.            1206.       0.340
##  9 Jake~  1024  3024   0.339             619.            1209.       0.339
## 10 Tony~  3141  9288   0.338             618.            1210.       0.338
## # ... with 6 more variables: ba_estimate1 <dbl>, ba_estimate2 <dbl>,
## #   phi <dbl>, phi1 <dbl>, phi2 <dbl>, lambda <dbl>
```

```
head(data_ba[order(-data_ba$ba_estimate1),],10)
```

```
## # A tibble: 10 x 13
##    name      H    AB average alpha_hierachic~ beta_hierachical ba_estimate
##    <chr> <int> <int>   <dbl>            <dbl>            <dbl>       <dbl>
##  1 Roge~  2930  8173   0.358             656.            1172.       0.359
##  2 Shoe~  1772  4981   0.356             651.            1177.       0.356
##  3 Ed D~  2596  7505   0.346             633.            1195.       0.346
##  4 Bill~  2158  6268   0.344             630.            1198.       0.344
##  5 Harr~  2660  7787   0.342             624.            1204.       0.342
##  6 Will~  2932  8591   0.341             624.            1204.       0.341
##  7 Bill~  2193  6428   0.341             624.            1204.       0.341
##  8 Lou ~  2721  8001   0.340             622.            1206.       0.340
##  9 Jake~  1024  3024   0.339             619.            1209.       0.339
## 10 Tony~  3141  9288   0.338             618.            1210.       0.338
## # ... with 6 more variables: ba_estimate1 <dbl>, ba_estimate2 <dbl>,
## #   phi <dbl>, phi1 <dbl>, phi2 <dbl>, lambda <dbl>
```

```
head(data_ba[order(-data_ba$ba_estimate2),],10)
```

```
## # A tibble: 10 x 13
##    name      H    AB average alpha_hierachic~ beta_hierachical ba_estimate
##    <chr> <int> <int>   <dbl>            <dbl>            <dbl>       <dbl>
##  1 Roge~  2930  8173   0.358             656.            1172.       0.359
##  2 Shoe~  1772  4981   0.356             651.            1177.       0.356
##  3 Ed D~  2596  7505   0.346             633.            1195.       0.346
##  4 Bill~  2158  6268   0.344             630.            1198.       0.344
##  5 Harr~  2660  7787   0.342             624.            1204.       0.342
##  6 Will~  2932  8591   0.341             624.            1204.       0.341
##  7 Bill~  2193  6428   0.341             624.            1204.       0.341
##  8 Lou ~  2721  8001   0.340             622.            1206.       0.340
##  9 Jake~  1024  3024   0.339             619.            1209.       0.339
## 10 Tony~  3141  9288   0.338             618.            1210.       0.338
## # ... with 6 more variables: ba_estimate1 <dbl>, ba_estimate2 <dbl>,
## #   phi <dbl>, phi1 <dbl>, phi2 <dbl>, lambda <dbl>
```

6.4 Comparison of sensitivity analysis of 3 prior model

parameters of randomly initialized prior distribution:$\alpha + \beta$	$\alpha/\alpha + \beta$	top1	ba_estimate
300	0.27	Rogers Hornsby	0.354
600	0.27	Rogers Hornsby	0.352
3000	0.27	Rogers Hornsby	0.335

parameters of pooled method prior distribution:$\alpha + \beta$	$\alpha/\alpha + \beta$	top1	ba_estimate
367	0.264	Rogers Hornsby	0.354
734	0.264	Rogers Hornsby	0.351
3672	0.264	Rogers Hornsby	0.329

parameters of hierachical model prior distribution:$\alpha + \beta$	$\alpha/\alpha + \beta$	top1	ba_estimate
3201	unchanged for each individual	Rogers Hornsby	0.359
6401	unchanged for each individual	Rogers Hornsby	0.359
32006	unchanged for each individual	Rogers Hornsby	0.359

The sensitivity of posterior inference about π to the proposed prior distribution is exhibited in Table. The first column is the parameter of the prior distribution $\alpha + \beta$. The second column is the mean of prior distribution, which is calculated by $\alpha/\alpha + \beta$. Here we fixed the mean of prior distribution. Then we can find, as $\alpha + \beta$ is growing, the estimation of posterior is more closely to the prior:

$$E(\pi|y) - > E(\pi)$$

Thus, the amount of prior information is measured by $\alpha + \beta$. Posterior inferences are not particularly sensitive to the prior distribution.

From the table we can deduce that, hierachical model is the most stabel one of the three models we used. Since its order is most stable and the BA estimations of the top 1 is also most stable.

7. Discussion and Conclusion

Discussion of problems, and potential improvements Here, compare the different model, the top ten players selected by different models are similar. The hierachical mode can provide a more stable result, with nearly unchanged order when we increase $\alpha + \beta$. However, this may be due to the fact that in the dataset, each person only have one data point of average. This will lead to the fact that the simulated $ba_estimate$ is quite similar to the average of each person. Therefore, we hope to get more average data of each person to further improve this model.