Baseball

Hanying Ji; Jiaqian Yu

2018/4/19

Setting Environment

```
library(Lahman)
library(dplyr)
library(tidyr)
library(MASS)
library(gamlss)
```

Data Preparation

Our dataset is from package "Lahman" in R, providing the tables from the "Sean Lahman Baseball Database"

```
# Define pitchers: Those players whose total amounts of pitch > 3
pitchers <-Pitching %>%
  group_by(playerID) %>%
  summarize(gamesPitched=sum(G)) %>%
  filter(gamesPitched>3)
```

```
## Warning: package 'bindrcpp' was built under R version 3.4.4
```

```
#collect data of batters from 2000 to 2016
info<-Batting %>%
  filter(AB>0,yearID%in%c(2000:2016)) %>%
  anti_join(pitchers, by="playerID")

info<-info%>%
  dplyr::select(playerID, AB, H) %>%
  group_by(playerID) %>%
  summarize(Hits=sum(H),AtBats=sum(AB))%>%
  mutate(BattingAvg=Hits/AtBats)

# add batters' names and hand preferance for batting
allinfo<-Master %>%
  tbl_df() %>%
  dplyr::select(playerID, nameFirst, nameLast, bats) %>%
  unite(name, nameFirst, nameLast, sep = " ") %>%
  inner_join(info, by="playerID")
head(allinfo)
```

```
## # A tibble: 6 x 6
##   playerID  name               bats   Hits AtBats BattingAvg
##   <chr>     <chr>              <fct> <int>  <int>      <dbl>
## 1 abadan01  Andy Abad          L         2     21     0.0952
## 2 abbotje01 Jeff Abbott        R        70    257     0.272
## 3 abbotku01 Kurt Abbott        R        36    166     0.217
## 4 abercre01 Reggie Abercrombie R        86    386     0.223
## 5 abernbr01 Brent Abernathy    R       212    868     0.244
```

```
## 6 abreubo01 Bobby Abreu        L      2080   7227     0.288
```

Model 1

```
# lowest BattingAvg
head(allinfo[order(allinfo$BattingAvg,decreasing = TRUE),])
```

```
## # A tibble: 6 x 6
##   playerID  name          bats   Hits AtBats BattingAvg
##   <chr>     <chr>         <fct> <int>  <int>      <dbl>
## 1 davidda01 Dave Davidson L         1      1      1.00
## 2 ohmeke01  Kevin Ohme    L         1      1      1.00
## 3 roachja01 Jason Roach   R         2      2      1.00
## 4 tupmama01 Matt Tupman   L         1      1      1.00
## 5 mantoje01 Jeff Manto    R         4      5      0.800
## 6 brettry01 Ryan Brett    B         2      3      0.667
```

```
# highest BattingAvg
head(allinfo[order(allinfo$BattingAvg,decreasing = FALSE),])
```

```
## # A tibble: 6 x 6
##   playerID  name             bats   Hits AtBats BattingAvg
##   <chr>     <chr>            <fct> <int>  <int>      <dbl>
## 1 adamsla01 Lane Adams       R         0      3          0
## 2 bantzbr01 Brandon Bantz    R         0      2          0
## 3 barkese01 Sean Barker      R         0      2          0
## 4 baronst01 Steve Baron      R         0     11          0
## 5 barteki01 Kimera Bartee    B         0     19          0
## 6 barthji01 Jimmy Barthmaier R         0      3          0
```

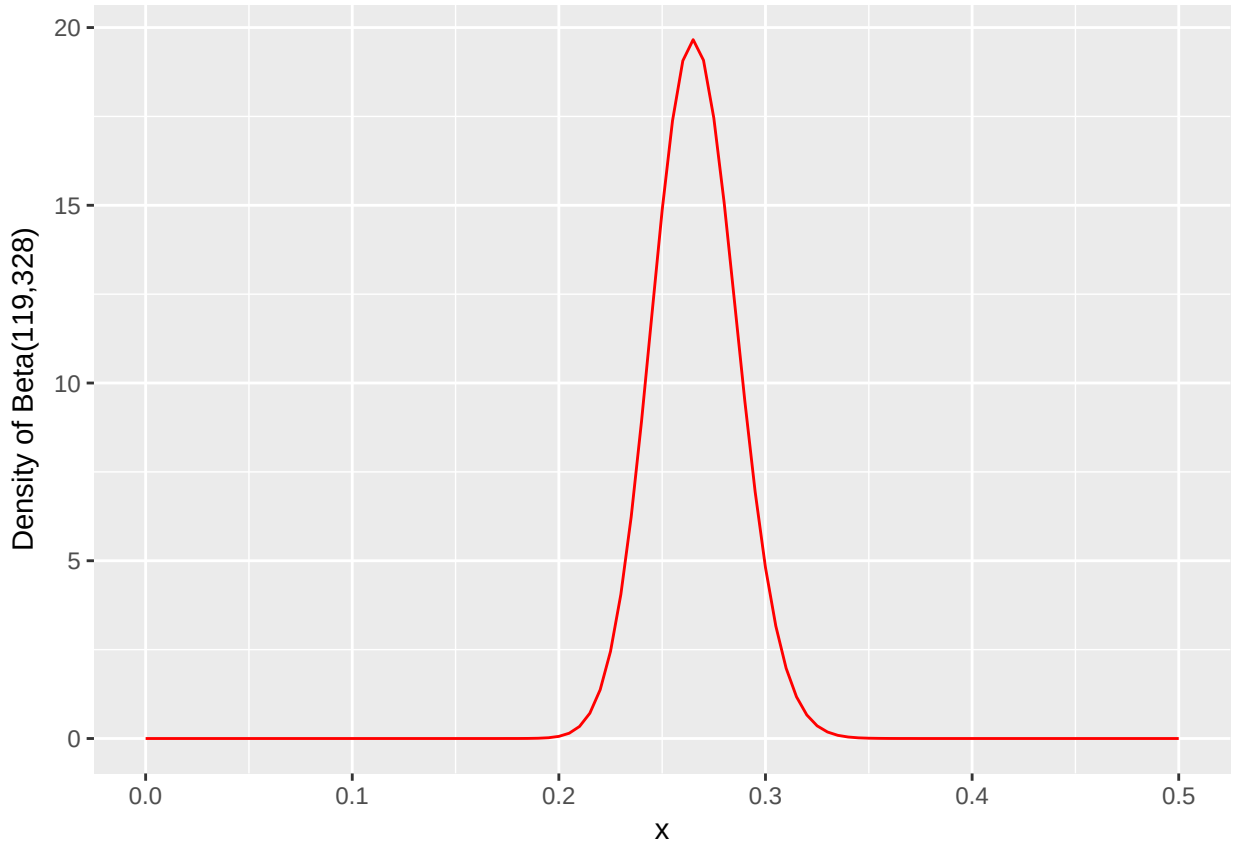
Model 2

```
set.seed(1)
alpha<-seq(1,200,1)
beta<-367*alpha/133
pbeta(0.3,alpha,beta)
```

```
##   [1] 0.6262650 0.6290661 0.6377271 0.6471558 0.6564142 0.6652789 0.6737082
##   [8] 0.6817149 0.6893279 0.6965793 0.7035000 0.7101178 0.7164580 0.7225426
##  [15] 0.7283913 0.7340215 0.7394487 0.7446866 0.7497474 0.7546424 0.7593813
##  [22] 0.7639733 0.7684266 0.7727485 0.7769459 0.7810252 0.7849919 0.7888516
##  [29] 0.7926090 0.7962687 0.7998349 0.8033117 0.8067026 0.8100112 0.8132405
##  [36] 0.8163938 0.8194737 0.8224830 0.8254242 0.8282997 0.8311117 0.8338624
##  [43] 0.8365538 0.8391879 0.8417663 0.8442910 0.8467635 0.8491854 0.8515582
##  [50] 0.8538834 0.8561623 0.8583963 0.8605866 0.8627344 0.8648409 0.8669071
##  [57] 0.8689342 0.8709232 0.8728751 0.8747908 0.8766712 0.8785172 0.8803297
##  [64] 0.8821094 0.8838572 0.8855739 0.8872601 0.8889167 0.8905441 0.8921433
##  [71] 0.8937147 0.8952590 0.8967768 0.8982688 0.8997354 0.9011772 0.9025948
##  [78] 0.9039887 0.9053593 0.9067072 0.9080329 0.9093367 0.9106193 0.9118809
##  [85] 0.9131220 0.9143431 0.9155444 0.9167266 0.9178898 0.9190345 0.9201610
##  [92] 0.9212697 0.9223610 0.9234351 0.9244924 0.9255332 0.9265579 0.9275666
```

```
##  [99] 0.9285598 0.9295377 0.9305006 0.9314487 0.9323824 0.9333019 0.9342075
## [106] 0.9350993 0.9359777 0.9368429 0.9376951 0.9385345 0.9393614 0.9401760
## [113] 0.9409786 0.9417692 0.9425482 0.9433156 0.9440718 0.9448170 0.9455512
## [120] 0.9462746 0.9469876 0.9476902 0.9483826 0.9490650 0.9497375 0.9504004
## [127] 0.9510537 0.9516976 0.9523324 0.9529580 0.9535747 0.9541827 0.9547820
## [134] 0.9553728 0.9559553 0.9565295 0.9570957 0.9576538 0.9582042 0.9587468
## [141] 0.9592818 0.9598093 0.9603295 0.9608424 0.9613482 0.9618470 0.9623388
## [148] 0.9628239 0.9633023 0.9637740 0.9642393 0.9646981 0.9651507 0.9655971
## [155] 0.9660373 0.9664715 0.9668998 0.9673223 0.9677390 0.9681501 0.9685555
## [162] 0.9689555 0.9693501 0.9697393 0.9701232 0.9705020 0.9708757 0.9712444
## [169] 0.9716081 0.9719669 0.9723210 0.9726703 0.9730149 0.9733549 0.9736904
## [176] 0.9740214 0.9743480 0.9746703 0.9749883 0.9753021 0.9756117 0.9759173
## [183] 0.9762188 0.9765163 0.9768099 0.9770996 0.9773856 0.9776677 0.9779462
## [190] 0.9782210 0.9784922 0.9787599 0.9790240 0.9792847 0.9795420 0.9797960
## [197] 0.9800466 0.9802940 0.9805382 0.9807792
```

```r
alpha0<-126
beta0<-347.6842
set.seed(1)
df<-data.frame(x=rbeta(1000,alpha0,beta0))
library(ggplot2)
ggplot(df,aes(df$x))+
  stat_function(fun=dbeta,args=list(alpha0,beta0),color="red")+
  xlim(0,0.5)+
  xlab("x")+
  ylab("Density of Beta(119,328)")
```



```
allinfo$alpha1<-alpha0+allinfo$Hits
allinfo$beta1<-beta0+allinfo$AtBats-allinfo$Hits
allinfo$BattingAvg_post<-allinfo$alpha1/(allinfo$alpha1+allinfo$beta1)
# lowest BattingAvg_post
head(allinfo[order(allinfo$BattingAvg_post,decreasing = TRUE),])
```

```
## # A tibble: 6 x 9
##   playerID  name              bats   Hits AtBats BattingAvg alpha1 beta1
##   <chr>     <chr>             <fct> <int>  <int>      <dbl>  <dbl> <dbl>
## 1 cabremi01 Miguel Cabrera    R      2519   7853      0.321   2645  5682
## 2 guerrvl01 Vladimir Guerrero R      2092   6570      0.318   2218  4826
## 3 bondsba01 Barry Bonds       L       925   2871      0.322   1051  2294
## 4 heltoto01 Todd Helton       L      2141   6761      0.317   2267  4968
## 5 ramirma02 Manny Ramirez     R      1642   5213      0.315   1768  3919
## 6 suzukic01 Ichiro Suzuki     L      3030   9689      0.313   3156  7007
## # ... with 1 more variable: BattingAvg_post <dbl>
```

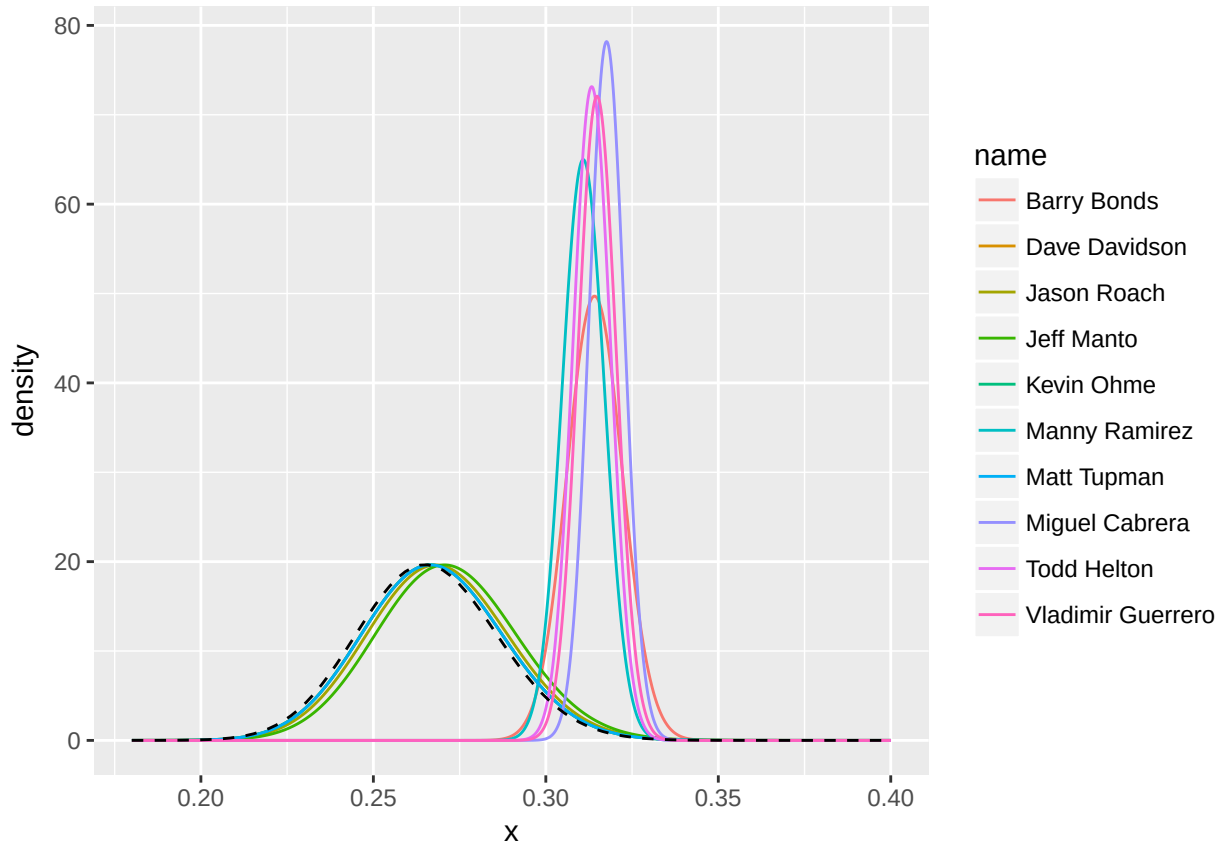
```
# highest BattingAvg_post
head(allinfo[order(allinfo$BattingAvg_post,decreasing = FALSE),])
```

```
## # A tibble: 6 x 9
##   playerID  name          bats   Hits AtBats BattingAvg alpha1 beta1
##   <chr>     <chr>         <fct> <int>  <int>      <dbl>  <dbl> <dbl>
## 1 mathije01 Jeff Mathis   R       401   2038      0.197    527  1985
## 2 zuninmi01 Mike Zunino   R       219   1125      0.195    345  1254
## 3 cashke01  Kevin Cash    R       117    641      0.183    243   872
## 4 woodbr01  Brandon Wood  R       130    700      0.186    256   918
## 5 sadledo01 Donnie Sadler R        97    537      0.181    223   788
## 6 hicksbr01 Brandon Hicks R        45    294      0.153    171   597
## # ... with 1 more variable: BattingAvg_post <dbl>
```

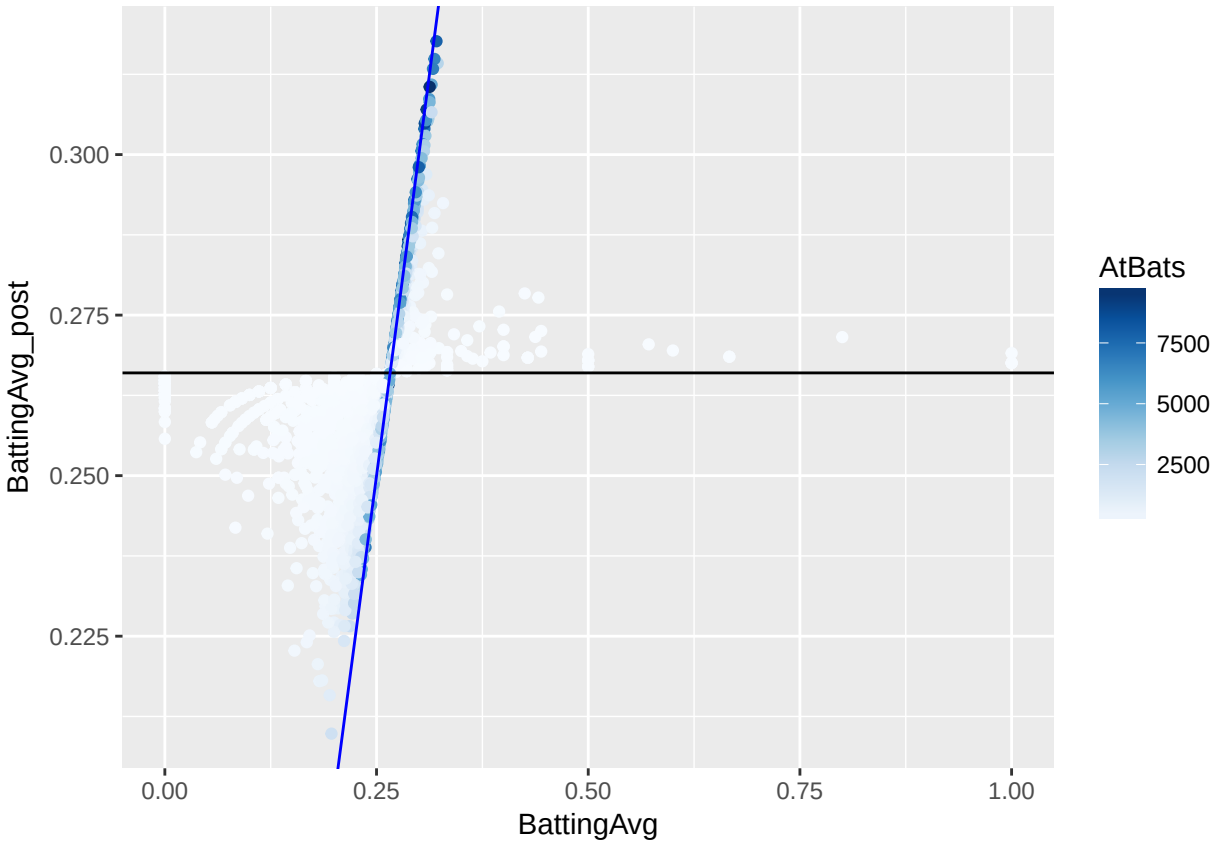
```
Top5<- allinfo[order(allinfo$BattingAvg_post,decreasing = TRUE),][1:5,]$playerID
Top5_old<-allinfo[order(allinfo$BattingAvg,decreasing = TRUE),][1:5,]$playerID
# Tail5<-allinfo[order(allinfo$BattingAvg_post,decreasing = FALSE),][1:5,]$playerID
Top5info <- allinfo %>%
   filter(playerID %in% c(Top5,Top5_old))

library(broom)
```

```
## Warning: package 'broom' was built under R version 3.4.4
```

```
five <- Top5info %>%
   tidyr::crossing(x = seq(.18, .4, .0002)) %>%
   ungroup() %>%
   mutate(density = dbeta(x, alpha1, beta1))
ggplot(five) +
  geom_line(aes(x, density,color=name)) +
  stat_function(fun = function(x) dbeta(x, alpha0, beta0),
                lty = 2, color = "black")
```

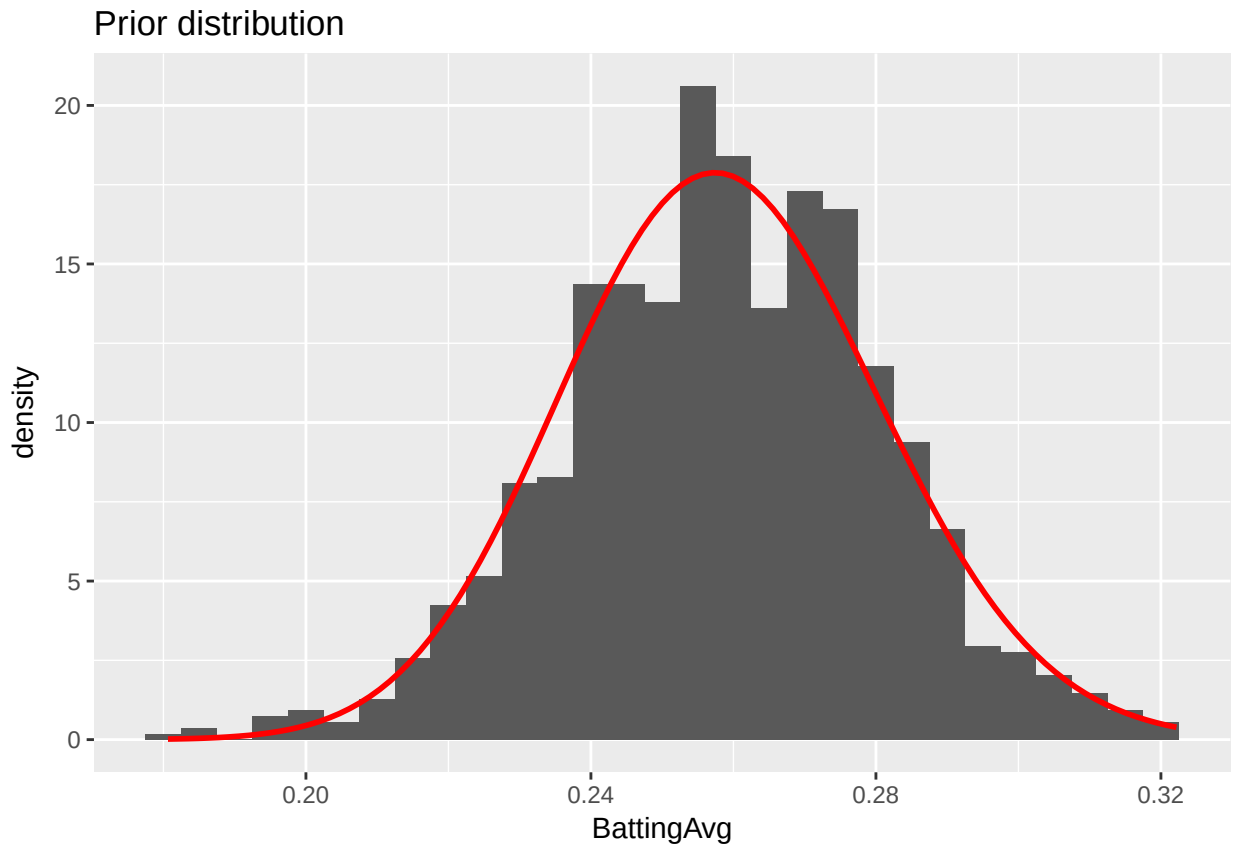



```
ggplot(data=allinfo)+
  geom_point(mapping=aes(BattingAvg,BattingAvg_post,colour=AtBats))+
  scale_colour_gradientn(colours = blues9)+
  geom_hline(yintercept = alpha0/(alpha0+beta0))+
  geom_abline(slope=1,color="blue")
```



There are a lot of method to fit a probability distribution from data in R. We???ll use the fitdistr function from MASS

```
# filter players
allinfo_filtered<-allinfo %>% filter(AtBats>=500)
# use the filtered data to fit the empirical Bayes estimation - a Beta prior
mle<-MASS::fitdistr(allinfo_filtered$BattingAvg, dbeta,
                    start = list(shape1 = 1, shape2 = 10))
alpha0_eb<-mle$estimate[1]
beta0_eb<-mle$estimate[2]
miu0_eb<-alpha0_eb/(alpha0_eb+beta0_eb)

ggplot(data=allinfo_filtered)+
  geom_histogram(binwidth = 0.005,aes(x=BattingAvg,y=..density..))+
  stat_function(fun = function(x) dbeta(x, alpha0_eb, beta0_eb), color = "red",size = 1)+
  labs(title="Prior distribution")
```



Then, for each player, they all have their Empericical Bayesian estimators for their BattingAvg based on the prior distribution given by the representative players

```
allinfo<-allinfo%>%
  mutate(EB_BattingAvg_post=(Hits+alpha0_eb)/(AtBats+alpha0_eb+beta0_eb))
```

```
# lowest EB estimator BattingAvg
head(allinfo[order(allinfo$EB_BattingAvg_post,decreasing = TRUE),])
```

```
## # A tibble: 6 x 10
##   playerID  name              bats   Hits AtBats BattingAvg alpha1 beta1
##   <chr>     <chr>             <fct> <int>  <int>      <dbl>  <dbl> <dbl>
## 1 cabremi01 Miguel Cabrera    R      2519   7853      0.321   2645  5682
## 2 guerrvl01 Vladimir Guerrero R      2092   6570      0.318   2218  4826
## 3 bondsba01 Barry Bonds       L       925   2871      0.322   1051  2294
## 4 heltoto01 Todd Helton       L      2141   6761      0.317   2267  4968
## 5 ramirma02 Manny Ramirez     R      1642   5213      0.315   1768  3919
## 6 suzukic01 Ichiro Suzuki     L      3030   9689      0.313   3156  7007
## # ... with 2 more variables: BattingAvg_post <dbl>,
## #   EB_BattingAvg_post <dbl>
```

```
# highest EB estimator BattingAvg
head(allinfo[order(allinfo$EB_BattingAvg_post,decreasing = FALSE),])
```

```
## # A tibble: 6 x 10
##   playerID  name          bats   Hits AtBats BattingAvg alpha1 beta1
##   <chr>     <chr>         <fct> <int>  <int>      <dbl>  <dbl> <dbl>
## 1 mathije01 Jeff Mathis   R       401   2038      0.197    527  1985
```

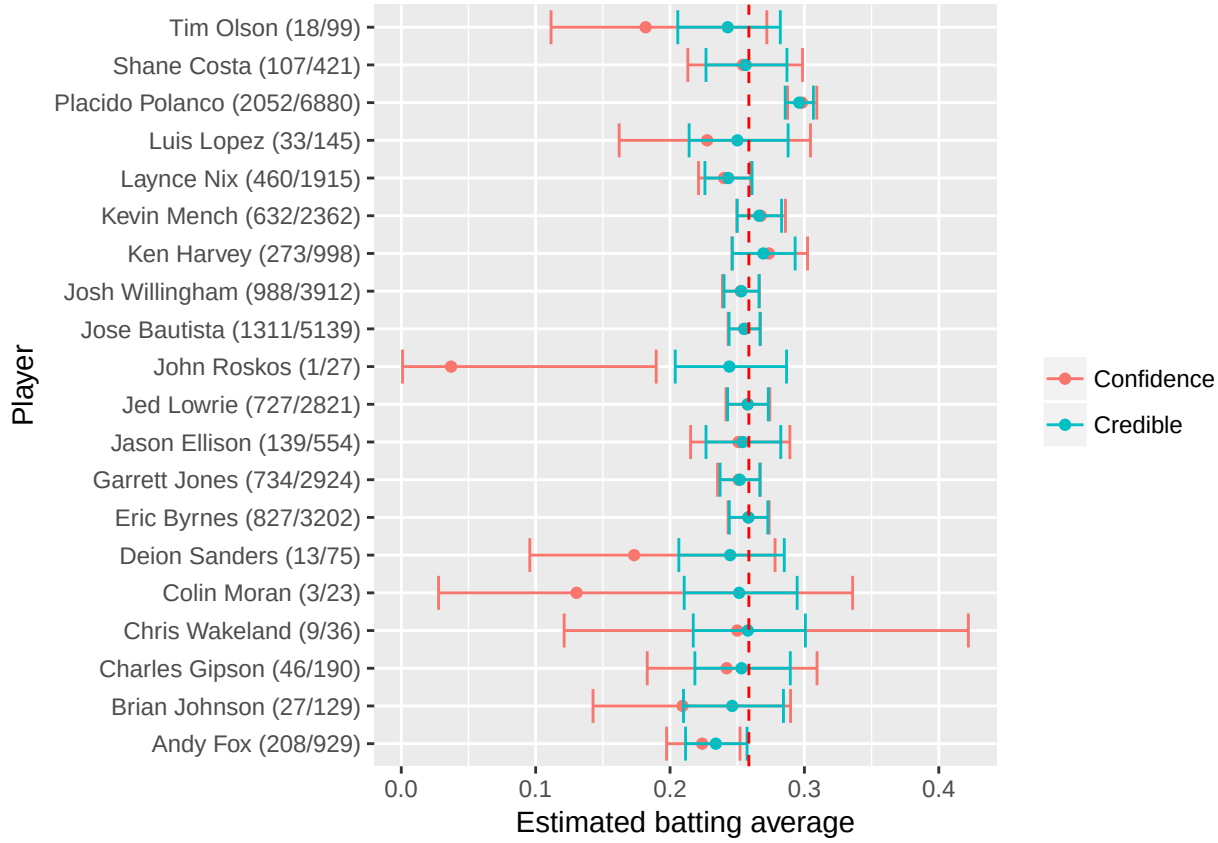
```
## 2 zuninmi01 Mike Zunino   R       219   1125      0.195    345  1254
## 3 cashke01  Kevin Cash    R       117    641      0.183    243   872
## 4 woodbr01  Brandon Wood  R       130    700      0.186    256   918
## 5 hicksbr01 Brandon Hicks R        45    294      0.153    171   597
## 6 sadledo01 Donnie Sadler R        97    537      0.181    223   788
## # ... with 2 more variables: BattingAvg_post <dbl>,
## #   EB_BattingAvg_post <dbl>
```

As we can see, the Empricial Bayes didn't simply choose those players who only have one or two bats, instead, players with large amount of AtBats were chosen.

```
career_eb <- allinfo %>%
  mutate(eb_estimate = (Hits + alpha0_eb) / (AtBats + alpha0_eb + beta0_eb))%>%
  mutate(alpha1_eb = Hits + alpha0_eb,
         beta1_eb = AtBats - Hits + beta0_eb)%>%
  dplyr::select(playerID,name,Hits,AtBats,BattingAvg,eb_estimate,alpha1_eb,beta1_eb)


career_eb <- career_eb %>%
    mutate(low = qbeta(.025, alpha1_eb, beta1_eb),
           high = qbeta(.975, alpha1_eb, beta1_eb))

set.seed(2018)

some <- career_eb %>%
    sample_n(20) %>%
    mutate(name = paste0(name, " (", Hits, "/", AtBats, ")"))

frequentist <- some %>%
    group_by(playerID, name, AtBats) %>%
    do(tidy(binom.test(.$Hits, .$AtBats))) %>%
    dplyr::select(playerID, name, estimate, low = conf.low, high = conf.high) %>%
    mutate(method = "Confidence")
```

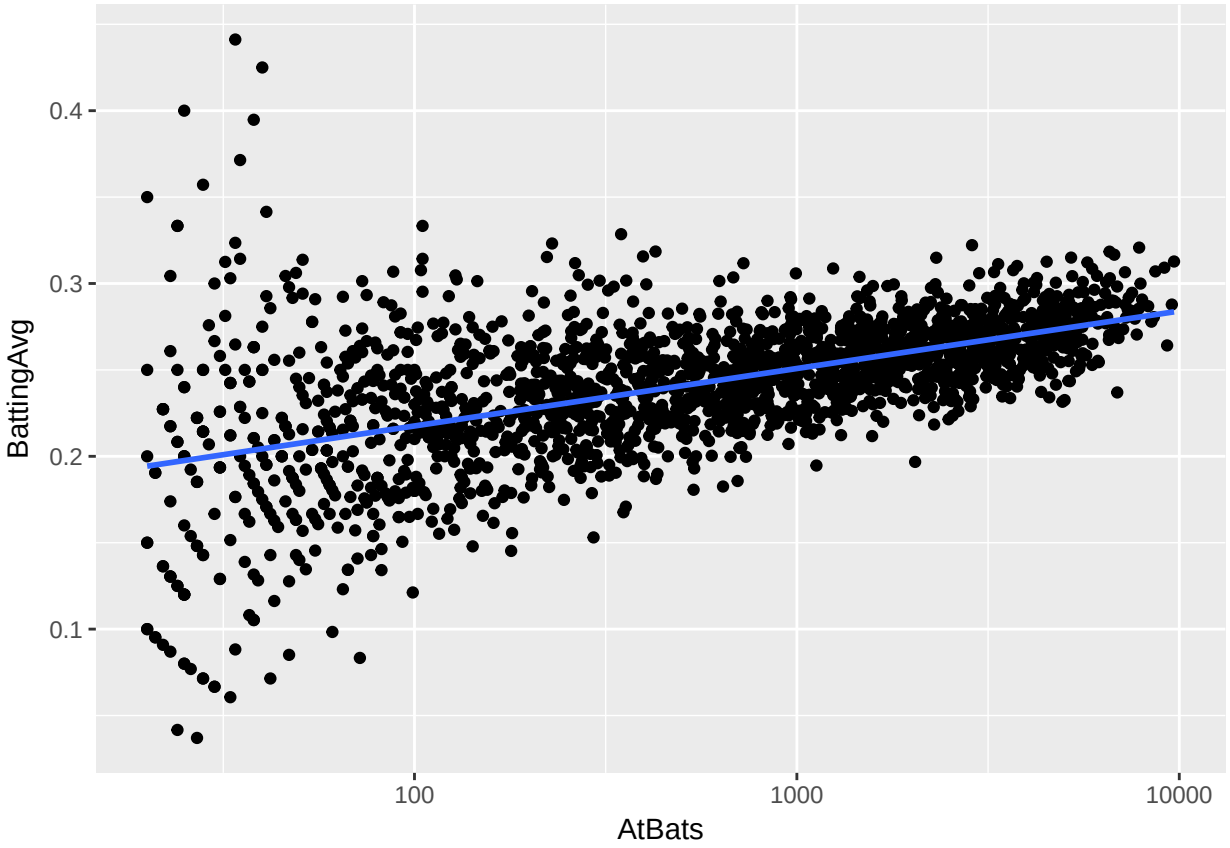
```
## Adding missing grouping variables: `AtBats`
```

```
bayesian <- some %>%
    dplyr::select(playerID, name, AtBats, estimate = eb_estimate,
           low = low, high = high) %>%
    mutate(method = "Credible")

combined <- bind_rows(frequentist, bayesian)

combined %>%
    #mutate(name = reorder(name, -AtBats)) %>%
    ggplot(aes(estimate, name, color = method, group = method)) +
    geom_point() +
    geom_errorbarh(aes(xmin = low, xmax = high)) +
    geom_vline(xintercept = alpha0_eb / (alpha0_eb + beta0_eb), color = "red", lty = 2) +
    xlab("Estimated batting average") +
    ylab("Player") +
    labs(color = "")
```



Model 3

```
library(ggplot2)
allinfo %>%
  filter(AtBats >= 20) %>%
  ggplot(aes(AtBats, BattingAvg)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE) +
  scale_x_log10()
```




```
#fit the initial model
fit <- gamlss(cbind(Hits, AtBats - Hits) ~ log(AtBats),
              data = allinfo,
              family = BB(mu.link = "identity"))
```

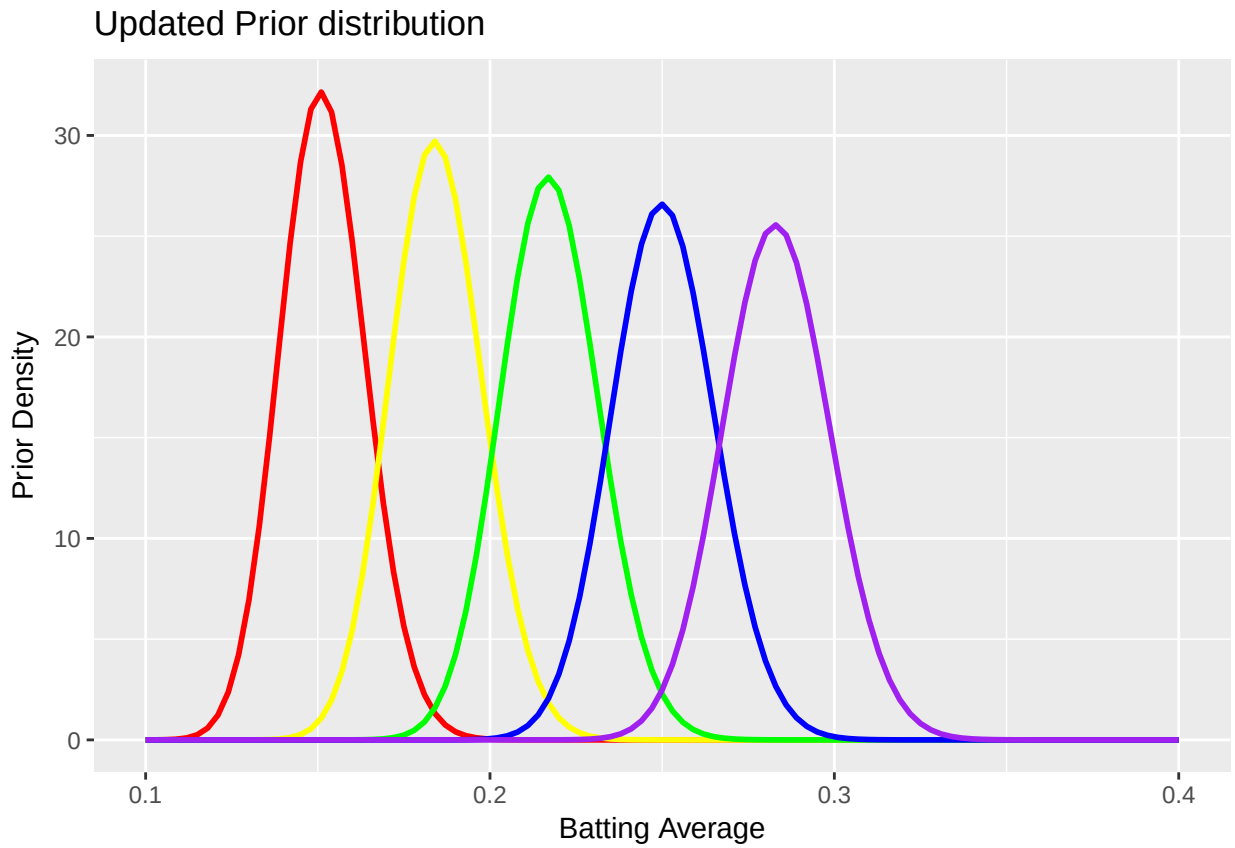
```
## GAMLSS-RS iteration 1: Global Deviance = 23213.77
## GAMLSS-RS iteration 2: Global Deviance = 18053.84
## GAMLSS-RS iteration 3: Global Deviance = 16828.39
## GAMLSS-RS iteration 4: Global Deviance = 16827.04
## GAMLSS-RS iteration 5: Global Deviance = 16827.04
```

```
library(broom)
td <- tidy(fit)
td
```

```
##   parameter        term    estimate    std.error  statistic       p.value
## 1        mu (Intercept)  0.15170322 0.0031952427   47.47784  0.000000e+00
## 2        mu log(AtBats)  0.01432317 0.0004368415   32.78803 8.850021e-193
## 3     sigma (Intercept) -6.72320574 0.0592102640 -113.54798  0.000000e+00
```

```
#calculate some prior dist
u0 <- 0.1517
lamda <- 0.0143
sigma0 <- 0.0012
u1 <- u0+lamda*log(1)
u2 <- u0+lamda*log(10)
u3 <- u0+lamda*log(100)
u4 <- u0+lamda*log(1000)
```

```r
u5 <- u0+lamda*log(10000)

a1 <- u1/sigma0
a2 <- u2/sigma0
a3 <- u3/sigma0
a4 <- u4/sigma0
a5 <- u5/sigma0
b1 <- (1-u1)/sigma0
b2 <- (1-u2)/sigma0
b3 <- (1-u3)/sigma0
b4 <- (1-u4)/sigma0
b5 <- (1-u5)/sigma0

df<-data.frame(x=seq(0.1,0.4,1000))
ggplot(df,aes(df$x),col())+
  stat_function(fun = dbeta,args=list(a1,b1), color = "red",size = 1)+
  stat_function(fun = dbeta,args=list(a2,b2), color = "yellow",size = 1)+
  stat_function(fun = dbeta,args=list(a3,b3), color = "green",size = 1)+
  stat_function(fun = dbeta,args=list(a4,b4), color = "blue",size = 1)+
  stat_function(fun = dbeta,args=list(a5,b5), color = "purple",size = 1)+
    xlim(0.1,0.4)+
  labs(title="Updated Prior distribution",x="Batting Average",y="Prior Density")
```



Model 4

```
#left hand and right hand
career2 <- allinfo %>%
  filter(!is.na(bats)) %>%
  mutate(bats = relevel(bats,"R"))

fit2 <- gamlss(cbind(Hits,AtBats-Hits)~log(AtBats)+bats,
               data=career2,
               family=BB(mu.link="identity"))
```

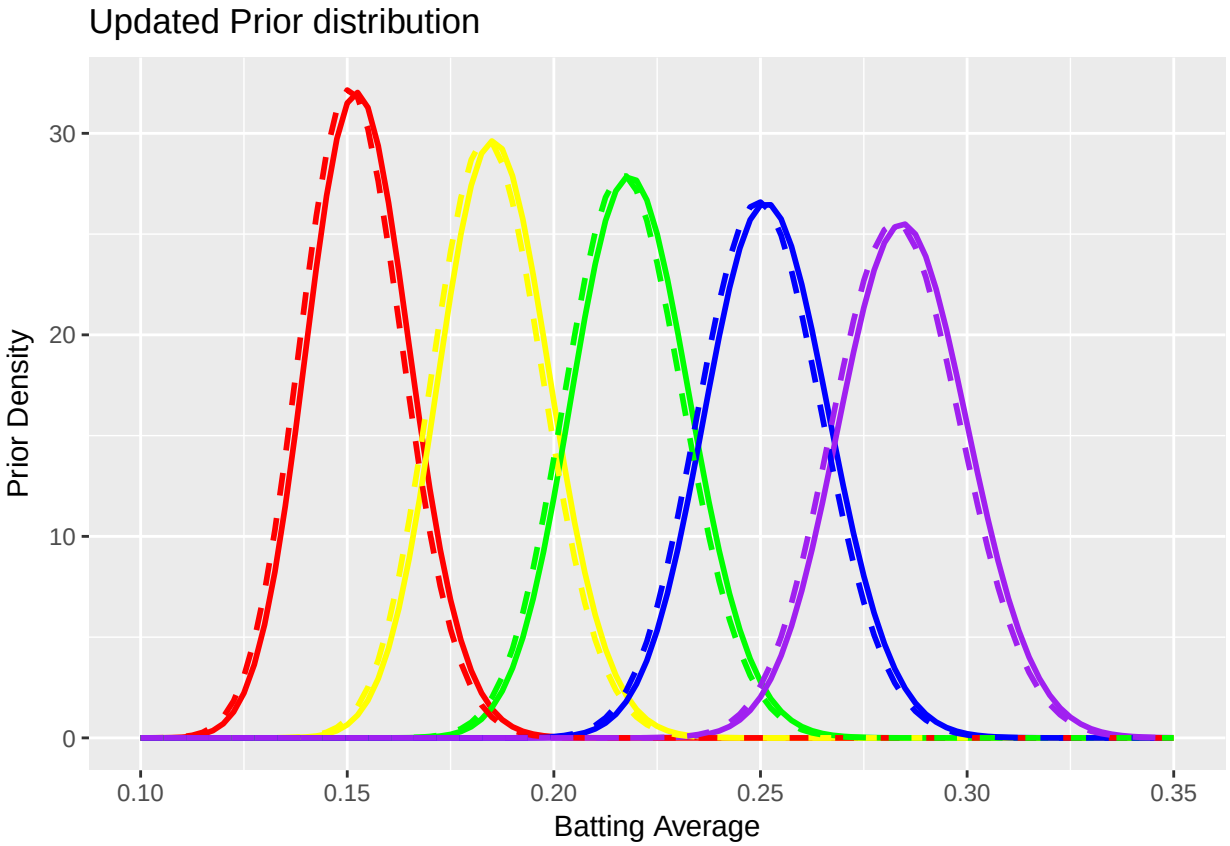
```
## GAMLSS-RS iteration 1: Global Deviance = 23214.08
## GAMLSS-RS iteration 2: Global Deviance = 18055.05
## GAMLSS-RS iteration 3: Global Deviance = 16825.66
## GAMLSS-RS iteration 4: Global Deviance = 16824.12
## GAMLSS-RS iteration 5: Global Deviance = 16824.12
```

```
tidy(fit2)
```

```
##   parameter        term      estimate    std.error   statistic
## 1        mu (Intercept)  0.1514003660 0.0032169312   47.063601
## 2        mu log(AtBats)  0.0143041068 0.0004367956   32.747826
## 3        mu       batsB -0.0006405712 0.0015137873   -0.423158
## 4        mu       batsL  0.0016889352 0.0011460300    1.473727
## 5     sigma (Intercept) -6.7260545279 0.0593051216 -113.414396
##         p.value
## 1  0.000000e+00
## 2 2.306952e-192
## 3  6.722208e-01
## 4  1.406964e-01
## 5  0.000000e+00
```

```
#calculate some prior dist
u0 <- 0.1514
lamda <- 0.0143
phi <- 0.0017
sigma0 <- 0.0012
BattingHand <- c(1,0,1,0,1,0,1,0,1,0)
AB <- c(1,10,100,1000,10000)

u1l <- u0+lamda*log(1)+phi*1
u1r <- u0+lamda*log(1)+phi*0
u2l <- u0+lamda*log(10)+phi*1
u2r <- u0+lamda*log(10)+phi*0
u3l <- u0+lamda*log(100)+phi*1
u3r <- u0+lamda*log(100)+phi*0
u4l <- u0+lamda*log(1000)+phi*1
u4r <- u0+lamda*log(1000)+phi*0
u5l <- u0+lamda*log(10000)+phi*1
u5r <- u0+lamda*log(10000)+phi*0
a1l <- u1l/sigma0
a1r <- u1r/sigma0
a2l <- u2l/sigma0
a2r <- u2r/sigma0
a3l <- u3l/sigma0
```

```r
a3r <- u3r/sigma0
a4l <- u4l/sigma0
a4r <- u4r/sigma0
a5l <- u5l/sigma0
a5r <- u5r/sigma0



b1l <- (1-u1l)/sigma0
b1r <- (1-u1r)/sigma0
b2l <- (1-u2l)/sigma0
b2r <- (1-u2r)/sigma0
b3l <- (1-u3l)/sigma0
b3r <- (1-u3r)/sigma0
b4l <- (1-u4l)/sigma0
b4r <- (1-u4r)/sigma0
b5l <- (1-u5l)/sigma0
b5r <- (1-u5r)/sigma0

df<-data.frame(x=seq(0.1,0.35,1000))
ggplot(df,aes(df$x))+
  stat_function(fun = dbeta,args=list(a1l,b1l), color = "red",size = 1)+
  stat_function(fun = dbeta,args=list(a2l,b2l), color = "yellow",size = 1)+
  stat_function(fun = dbeta,args=list(a3l,b3l), color = "green",size = 1)+
  stat_function(fun = dbeta,args=list(a4l,b4l), color = "blue",size = 1)+
  stat_function(fun = dbeta,args=list(a5l,b5l), color = "purple",size = 1)+
  stat_function(fun = dbeta,args=list(a1r,b1r), color = "red",size = 1,linetype="dashed")+
  stat_function(fun = dbeta,args=list(a2r,b2r), color = "yellow",size = 1,linetype="dashed")+
  stat_function(fun = dbeta,args=list(a3r,b3r), color = "green",size = 1,linetype="dashed")+
  stat_function(fun = dbeta,args=list(a4r,b4r), color = "blue",size = 1,linetype="dashed")+
  stat_function(fun = dbeta,args=list(a5r,b5r), color = "purple",size = 1,linetype="dashed")+
    xlim(0.1,0.35)+
  labs(title="Updated Prior distribution",x="Batting Average",y="Prior Density")
```


Updated Prior distribution
Prior Density
30
20
10
0
0.10
0.15
0.20
0.25
0.30
0.35
Batting Average