Lesson 24 Boardsheet - Multilevel Modeling

Kevin Cummiskey

3/25/2020

Today, we are going to investigate the limitations of the quadratic trajectories and discuss improvements using Bayesian statistics. The reference for the next two lessons is:

- "Multilevel Modeling of OBP trajectories" by Jim Albert. https://baseballwithr.wordpress.com/2019/11/25/multilevel-modeling-of-obp-trajectories/

Learning Objectives:

- Gain appreciation for how Bayesian statistics can help us combine prior knowledge with new observations to update our beliefs.
- Gain appreciation for how multilevel modeling can "pool" information to arrive a better estimates for individuals.

Players who debuted in 2001.

Let's find the players who debuted in the year 2001.

```
library(Lahman)
library(tidyverse)
library(lubridate)
library(ggrepel)

#Players with debut in year 2000
Master %>%
  filter(year(debut) == 2001) %>%
  pull(playerID) -> year2001.ids

#Players with at least 1000 atbats
Batting %>%
  filter(playerID %in% year2001.ids) %>%
  group_by(playerID) %>%
  summarize(AB = sum(AB)) %>%
  filter(AB > 1000) %>%
  pull(playerID) -> player.ids
```

The players who debuted in 2001 who would go on to have at least 1000 at bats are:

```
library(knitr)
Master %>%
  filter(playerID %in% player.ids) %>%
  select(nameFirst, nameLast) %>%
  kable()
```

nameFirst	nameLast
Angel	Berroa
Wilson	Betemit
Larry	Bigbie
Endy	Chavez
Alex	Cintron
Michael	Cuddyer
Jack	Cust
Adam	Dunn
David	Eckstein
Johnny	Estrada
Adam	Everett
Ryan	Freel
Jay	Gibbons
Marcus	Giles
Willie	Harris
Shea	Hillenbrand
Brandon	Inge
Cesar	Izturis
Nick	Johnson
Bobby	Kielty
Felipe	Lopez
Rob	Mackowiak
Jason	Michaels
Dustan	Mohr
Craig	Monroe
Lyle	Overbay
Carlos	Pena
Jason	Phillips
Scott	Podsednik
Albert	Pujols
Nick	Punto
Juan	Rivera
Brian	Roberts
Aaron	Rowand
Alex	Sanchez
Junior	Spivey
Ichiro	Suzuki
Yorvit	Torrealba
Juan	Uribe
Ramon	Vazquez
Brad	Wilkerson
Craig	Wilson
Jack	Wilson

OBP trajectories

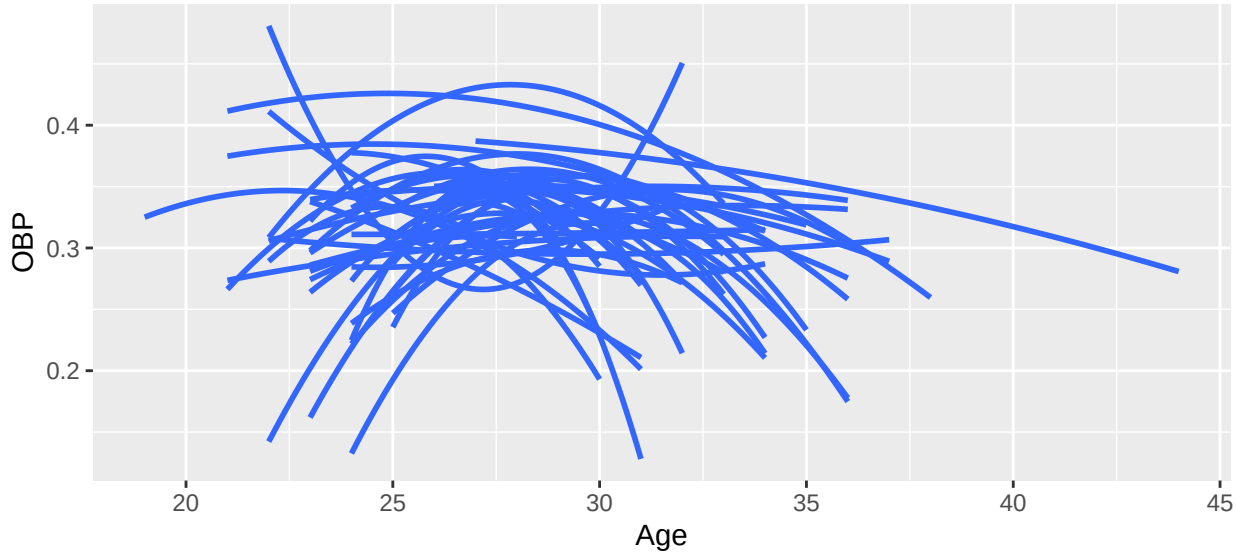
Let's look at their OBP trajectories using the quadratic model fit individually to players.

```
source("Chapter8_functions.R")

# get statistics by age and add names
```

```
player.ids %>%
  map_df(get_stats) %>%
  left_join(Master %>% select(nameLast, nameFirst, playerID))-> player.stats
```

```
## Joining, by = "playerID"
```

```
#plot trajectories
player.stats %>%
  ggplot(aes(x = Age, y = OBP, group = playerID)) +
  geom_smooth(method = "lm",
              formula = y ~ x + I(x^2),
              se = FALSE)
```



What do you think of these models?

How do we fix these issues?

Bayesian Approach

Trajectories from Multilevel Models

Now, let's fit trajectories that pool information from all the players in the data set.

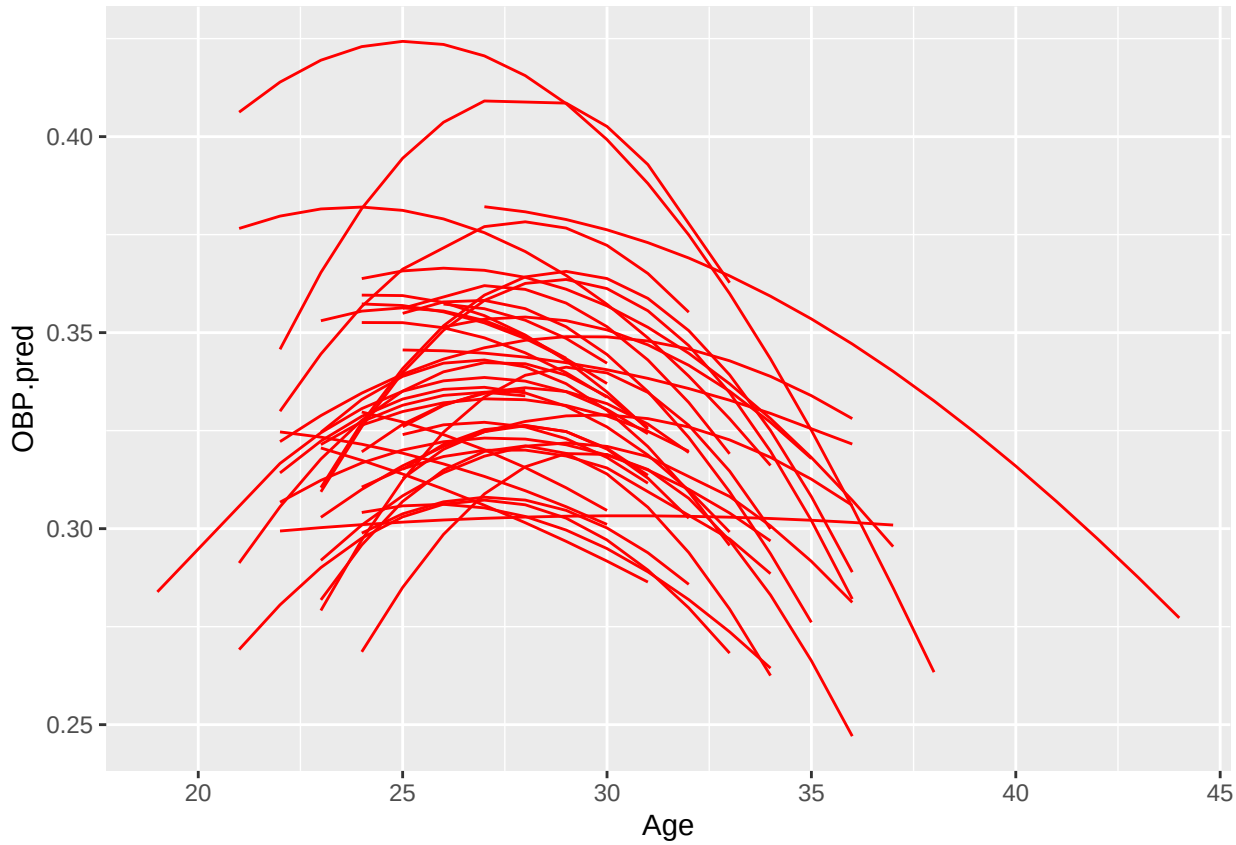
```
library(brms)
library(rstan)
player.stats %>%
  mutate(AgeD = Age - 30,
         Player = paste(nameFirst,nameLast, sep = " ")) -> player.stats


fit <- brm(OB | trials(PA) ~ AgeD + I(AgeD ^ 2) +
             (AgeD + I(AgeD ^ 2) | Player),
           data = player.stats,
           family = binomial("logit"))
```

```
## Running /Library/Frameworks/R.framework/Resources/bin/R CMD SHLIB foo.c
## clang -I"/Library/Frameworks/R.framework/Resources/include" -DNDEBUG   -I"/Users/kfcummiskey/Library/
## In file included from <built-in>:1:
## In file included from /Users/kfcummiskey/Library/R/3.6/library/StanHeaders/include/stan/math/prim/ma
## In file included from /Users/kfcummiskey/Library/R/3.6/library/RcppEigen/include/Eigen/Dense:1:
## In file included from /Users/kfcummiskey/Library/R/3.6/library/RcppEigen/include/Eigen/Core:88:
## /Users/kfcummiskey/Library/R/3.6/library/RcppEigen/include/Eigen/src/Core/util/Macros.h:613:1: error
## namespace Eigen {
## ^
## /Users/kfcummiskey/Library/R/3.6/library/RcppEigen/include/Eigen/src/Core/util/Macros.h:613:16: erro
## namespace Eigen {
##                ^
##                ;
## In file included from <built-in>:1:
## In file included from /Users/kfcummiskey/Library/R/3.6/library/StanHeaders/include/stan/math/prim/ma
## In file included from /Users/kfcummiskey/Library/R/3.6/library/RcppEigen/include/Eigen/Dense:1:
## /Users/kfcummiskey/Library/R/3.6/library/RcppEigen/include/Eigen/Core:96:10: fatal error: 'complex' 
## #include <complex>
##          ^~~~~~~~
## 3 errors generated.
## make: *** [foo.o] Error 1
##
## SAMPLING FOR MODEL 'cec340bfd9aff6b986bb28bfaed3b2e1' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 0.000405 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 4.05 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
```

```
## Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 38.2201 seconds (Warm-up)
## Chain 1:                13.0552 seconds (Sampling)
## Chain 1:                51.2753 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'cec340bfd9aff6b986bb28bfaed3b2e1' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 0.000227 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 2.27 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 48.0116 seconds (Warm-up)
## Chain 2:                13.4497 seconds (Sampling)
## Chain 2:                61.4613 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'cec340bfd9aff6b986bb28bfaed3b2e1' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 0.000214 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 2.14 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 3: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 3: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 3: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 3: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 3: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 3: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 3: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 3: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3:
```

```
## Chain 3:  Elapsed Time: 39.758 seconds (Warm-up)
## Chain 3:                13.3247 seconds (Sampling)
## Chain 3:                53.0827 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'cec340bfd9aff6b986bb28bfaed3b2e1' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 0.000367 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 3.67 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 4: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 4: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 4: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 4: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 4: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 44.701 seconds (Warm-up)
## Chain 4:                12.9953 seconds (Sampling)
## Chain 4:                57.6963 seconds (Total)
## Chain 4:
```

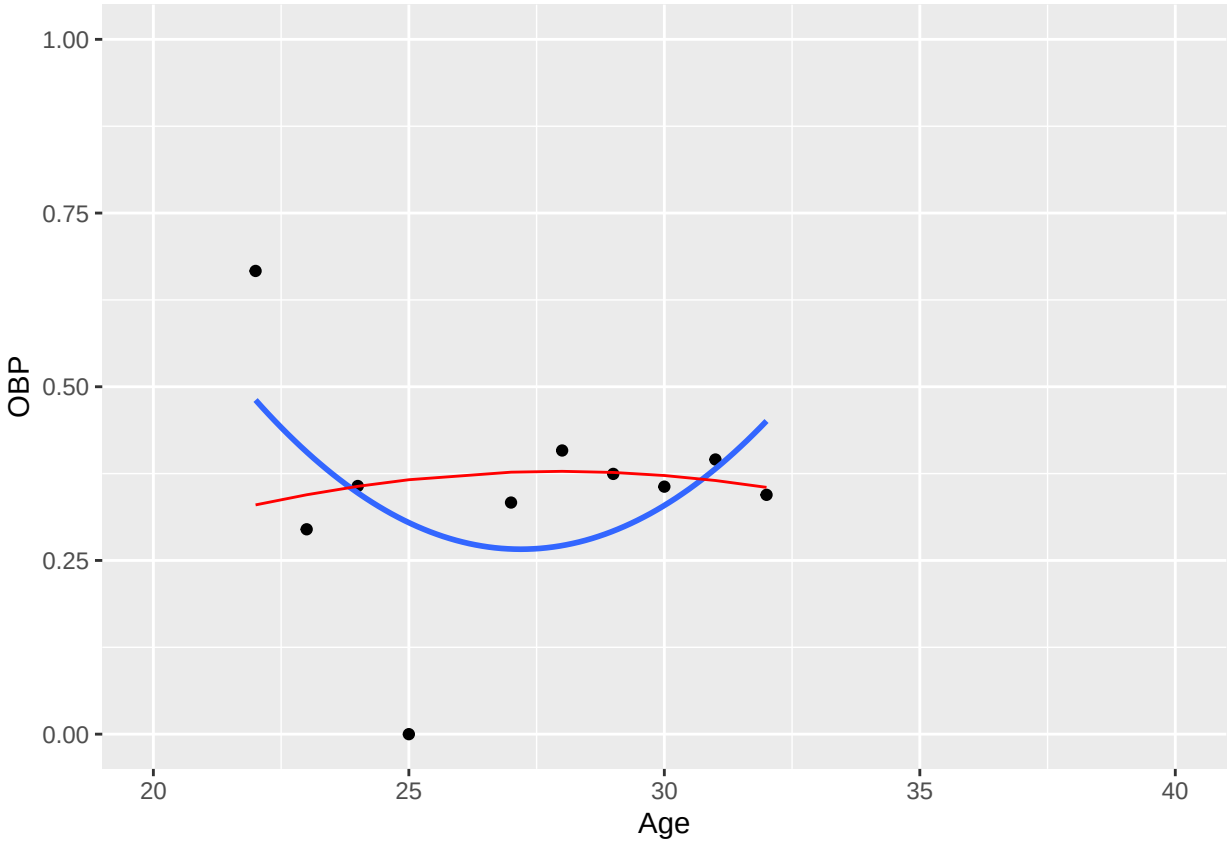
```r
Player_Fits <- coef(fit)$Player[, "Estimate", ] %>%
  as_tibble(rownames = "Player") %>%
  rename(b0.hat = Intercept,
         b1.hat = AgeD,
         b2.hat = IAgeDE2)

# merge these estimates with our main dataset

player.stats <- inner_join(player.stats, Player_Fits, by = "Player")

# find estimates of OBP probs at each age
# note plogis is the logit function
player.stats %>%
  mutate(OBP.pred = plogis(b0.hat + b1.hat * AgeD + b2.hat * AgeD^2)) -> player.stats

player.stats %>%
  ggplot(aes(x = Age,
             y = OBP.pred,
             group = Player)) +
  geom_line(color = "red")
```



What do you think of the new trajectories?

Let's focus on one player who had a really weird individual trajectory:

```
player.stats %>%
  filter(playerID == "custja01") %>%
  ggplot(aes(x = Age, y = OBP)) +
  geom_point() +
  geom_smooth(method = "lm",
              formula = y ~ x + I(x^2), se = FALSE) +
  ylim(0,1) + xlim(20,40) +
  geom_line(aes(y = OBP.pred), color = "red")
```


1.00
0.75
0.50
0.25
0.00
OBP
20
25
30
35
40
Age