CS 1675 Spring 2022 - MIDTERM

Assigned February 28, 2022; Due: March 8, 2022

Sameera Boppana

Submission time: March 8, 2022 at 11:00PM EST

Collaborators

You are **NOT** allowed to collaborate within anyone. Collaboration, copying, and/or cheating of any kind will not be tolerated.

Overview

This midterm tests your understanding of the concepts, math, and programming required to learn distributions from data. You are required to perform a mixture of derivations and programming to solve the questions on the exam. **Read the problem statements carefully.**

IMPORTANT: code chunks are created for you. Each code chunk has `eval=FALSE` set in the chunk options. You **MUST** change it to be `eval=TRUE` in order for the code chunks to be evaluated when rendering the document.

You are allowed to add as many code chunks as you see fit to answer the questions.

Load packages

This assignment will use packages from the `tidyverse` suite.

```
library(tidyverse)
```

```
## ── Attaching packages ─────────────────────────────────────── tidyverse 1.3.1 ──
```

```
## ✓ ggplot2 3.3.5     ✓ purrr   0.3.4
## ✓ tibble  3.1.6     ✓ dplyr   1.0.8
## ✓ tidyr   1.2.0     ✓ stringr 1.4.0
## ✓ readr   2.1.2     ✓ forcats 0.5.1
```

```
## Warning: package 'tidyr' was built under R version 4.0.5
```

```
## Warning: package 'readr' was built under R version 4.0.5
```

```
## Warning: package 'dplyr' was built under R version 4.0.5
```

```
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

Problem 01

You have fit discrete and continuous distributions to data, using non-Bayesian and Bayesian approaches. Bayesian analyses require a prior to be formulated, and it can be difficult to understand how a prior is specified in a general setting. This exam seeks to give you some practice doing that by using the **Empirical Bayes** approach. Empirical Bayes is a rather odd sounding name, but the idea is that you will estimate the parameters of the prior using all of the data. It is useful when the data can be structured into **groups**. Some groups might have many observations, while others may have a limited number of samples. Empirical Bayes is useful when there are many groups (potentially in the thousands) that can be used to estimate the prior parameters. Once estimated, the prior is applied to each group separately. In this manner you have made use of data to understand the relevant bounds on your unknowns and specified those bounds within a prior probability distribution. The prior is updated based on each group's data to yield the updated belief (the posterior) for each group. (Note that if we would have very few groups we could not use Empirical Bayes and thus would need to use full Bayesian approaches via multilevel, hierarchical, or partial pooling models.)

To see how the Empirical Bayes process works you will work with a Sports related application. You are interested in learning the catch probability (or catch rate) in the National Football League (NFL). The catch rate is defined as the number of successful receptions (catches) by a player divided by the number of targets (a target corresponds to a pass thrown at the player). You can therefore consider successfully catching a pass as the **event**, and the number of times the player was targeted as the number of **trials**. The probability of catching a pass is therefore the **event probability** we are interested in learning.

Let's consider you are working on this application because you were recently hired as a sports analytics intern for an NFL team. You are provided with 3 seasons worth of data (2018, 2019, and 2020) of every player with at least 1 target (thus at least 1 trial). Calculating the catch rate is simple to do. It is also easy to search for and find. For example, here (https://www.pro-football-reference.com/years/2019/receiving.htm) are the catch rates for all NFL players in the 2019 season. You were hired because the NFL team wishes to move away from simple *point estimates*. The team wants to have a better understanding of the *uncertainty* in the performance. Understanding the uncertainty is critical when evaluating talent, and making decisions for which players to sign in free agency.

You will work with two datasets for this exam. Both are loaded for you in code chunk below. The first, `df_all` , is the larger of the two. The second, `df_focus` , is a subset of `df_all` so that we way can focus on 23 players to help with visualization and discussion.

```
url_all <- "https://raw.githubusercontent.com/jyurko/CS_1675_Spring_2022/main/HW/midterm/midterm_all_data.
csv"
df_all <- readr::read_csv(url_all, col_names = TRUE)
```

```
## Rows: 849 Columns: 3
## ── Column specification ────────────────────────────────────────────────────────
## Delimiter: ","
## dbl (3): player_id, num_trials, num_events
##
## ℹ Use `spec()` to retrieve the full column specification for this data.
## ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
url_focus <- "https://raw.githubusercontent.com/jyurko/CS_1675_Spring_2022/main/HW/midterm/midterm_focus_d
ata.csv"
df_focus <- readr::read_csv(url_focus, col_names = TRUE)
```

```
## Rows: 23 Columns: 3
## ── Column specification ────────────────────────────────────────────────────────
## Delimiter: ","
## dbl (3): player_id, num_trials, num_events
##
## ℹ Use `spec()` to retrieve the full column specification for this data.
## ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

Both data sets consist of 3 variables, `player_id` , `num_events` , and `num_trials` . The `num_events` is the number of receptions, and `num_trials` is the number of targets (just written in general terms that we have used in the class). The `player_id` variable is an ID variable for each player. Thus, one row in either data set tells us the number of receptions and number of targets associated with an individual player over the three seasons. The data in this exam are real and were downloaded from the `nflfastR` package (documentation available here (https://www.nflfastr.com/index.html) if you are interested). The `player_id` variable is an anonymous identification number I created so that NFL fans cannot easily tell which player is which.

1a)

To help understand why Empirical Bayes can be useful, let's suppose you're not sure how to specify an informative prior for this example. Even if you watch every Pittsburgh Steelers' game, you might not know what the average catch rate is in the NFL. Since you do not feel comfortable specifying reasonable bounds, you decide to use a vague prior formulation.

You will use a Binomial likelihood and a conjugate Beta prior on the unknown catch rate (or event probability in general terms), μ. For generality, you will denote each player with a subscript j and the total number of players as J. Thus, the unknown event probability for the j-th player is μ_j where $j = 1, \ldots, J$. The posterior distribution on the j-th player's unknown catch rate, μ_j given the m_j catches (events) out of N_j targets (trials) is proportional to:

$$p\left(\mu_j \mid (m, N)_j\right) \propto \text{Binomial}\left(m_j \mid \mu_j, N_j\right) \times \text{Beta}\left(\mu_j \mid a, b\right)$$

Notice that in the above posterior formulation, each player has a potentially distinct event probability, μ_j. The prior consists of two shape hyperparameters, a and b. The same prior hyperparameters are applied to every player.

You will assume prior shape parameters of $a = 0.5$ and $b = 0.5$. How many "prior trials" or "prior targets" does this specification correspond to? Why do you think it represents being "uninformed" about a process?

SOLUTION

What do you think?

The number of prior trials that this specification corresponds to is 1. The hyperparameter a is the a priori number of events and the hyperparameter b is the a priori number of non-events. Therefore, to get the number of prior trials, you would add the two hyperparemters together. Since both the hyperparameters are equal to 0.5, the total number of prior trials is equal to 1. This represents being uninformed about a process because we are using only 1 trial in the prior, so we do not many trials to base the posterior on.

1b)

You are using a conjugate prior to the Binomial likelihood, for each player.

What type of distribution is the posterior for the unknown event probability, μ_j, for each player, $j = 1, \ldots, J$?

SOLUTION

What do you think?

The distribution of the posteior for the unknown event probability will be a beta distribution. The posterior distribtuion will have the same functional form as the prior and since the prior is a beta distribution, the posterior will be the same.

1c)

Write out the formula for the updated or posterior hyperparameters, $a_{new,j}$ and $b_{new,j}$, based on each player's observed number of catches m_j and observed number of targets N_j, as well as the prior shape parameters, a and b.

SOLUTION

Add your equation blocks here.

$$a_{new,j} = a + m_j = 0.5 + m_j$$
$$b_{new,j} = b + (N_j - m_j) = 0.5 + (N_j - m_j)$$

1d)

Based on your formula in Problem 1c), calculate the updated shape parameters for the 23 players in the `df_focus tibble`. You should add two columns using `mutate()` named `anew` and `bnew`. Assign your result to the `post_df_focus_from_vague` object.

SOLUTION

```
post_df_focus_from_vague <- df_focus  %>%
  mutate(
    anew = 0.5 + num_events,
    bnew = 0.5 + (num_trials - num_events)
  )
head(post_df_focus_from_vague)
```

```
## # A tibble: 6 × 5
##   player_id num_trials num_events  anew  bnew
##       <dbl>      <dbl>      <dbl> <dbl> <dbl>
## 1        24          1          0   0.5   1.5
## 2        25          1          0   0.5   1.5
## 3        34          3          0   0.5   3.5
## 4       169         13          3   3.5  10.5
## 5       300          8          2   2.5   6.5
## 6       186          3          1   1.5   2.5
```

1e)

Calculate the posterior mean, 5th quantile, and 95th quantile for each player in `post_df_focus_from_vague`. You should add 3 columns using `mutate()` named `post_avg`, `post_q05`, and `post_q95`. Assign the result to the variable `summary_post_df_focus_from_vague`.

SOLUTION

```
summary_post_df_focus_from_vague <- post_df_focus_from_vague %>%
  mutate(
    post_avg = anew / (anew + bnew),
    post_q05 = qbeta(0.05, anew, bnew),
    post_q95 = qbeta(0.95, anew, bnew)
  )
summary_post_df_focus_from_vague
```

```
## # A tibble: 23 × 8
##    player_id num_trials num_events  anew  bnew post_avg post_q05 post_q95
##        <dbl>      <dbl>      <dbl> <dbl> <dbl>    <dbl>    <dbl>    <dbl>
##  1        24          1          0   0.5   1.5    0.25  0.00154     0.771
##  2        25          1          0   0.5   1.5    0.25  0.00154     0.771
##  3        34          3          0   0.5   3.5    0.125 0.000603    0.444
##  4       169         13          3   3.5  10.5    0.25  0.0885      0.453
##  5       300          8          2   2.5   6.5    0.278 0.0763      0.538
##  6       186          3          1   1.5   2.5    0.375 0.0624      0.764
##  7       260          8          3   3.5   5.5    0.389 0.150       0.657
##  8       607         13          5   5.5   8.5    0.393 0.194       0.610
##  9        20        107         57  57.5  50.5    0.532 0.453       0.611
## 10       546         54         29  29.5  25.5    0.536 0.426       0.645
## # … with 13 more rows
```

1f)

You will now visualize the posterior summaries for the 23 players associated with the `df_focus` data set.

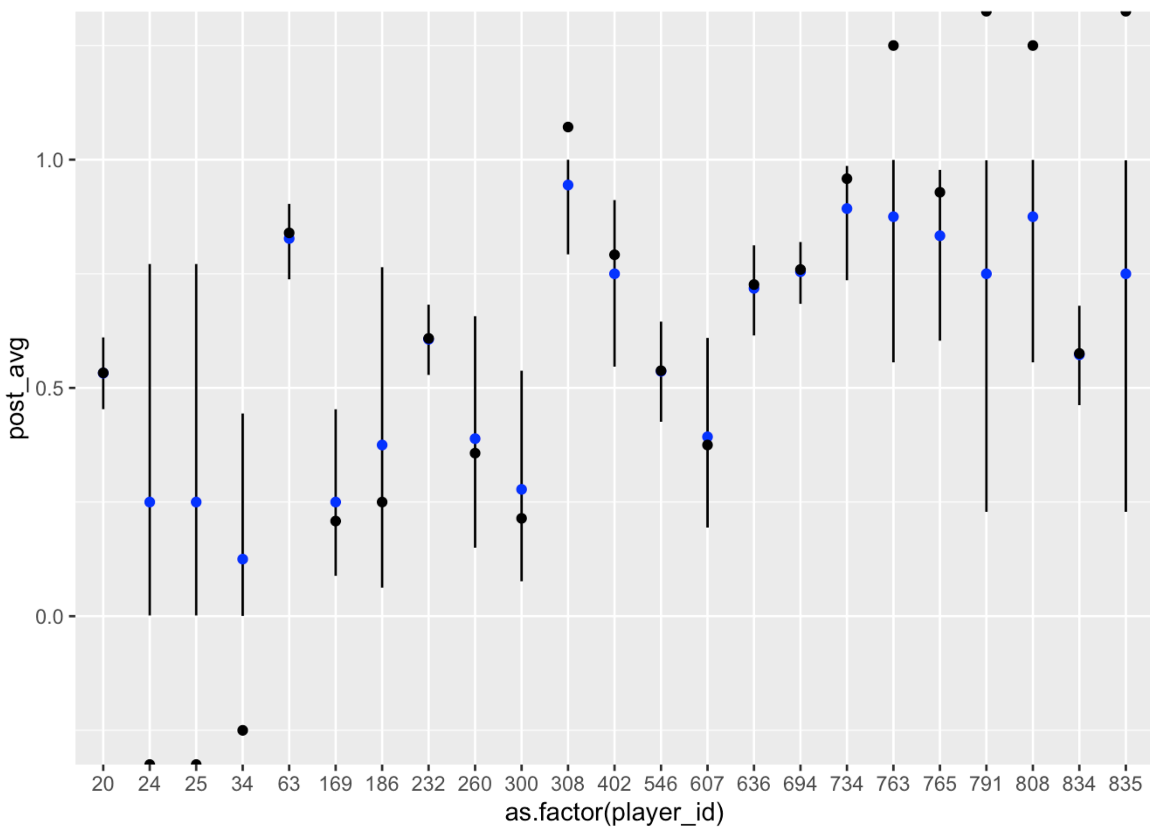
Pipe `summary_post_df_focus_from_vague` into `ggplot()` and map the `x` aesthetic to `as.factor(player_id)`. You will use the `geom_linerange()` to represent the posterior uncertainty by setting the `ymin` and `ymax` aesthetics to `post_q05` and `post_q95`, respectively. You will display the posterior mean with a `geom_point()` by setting the `y` aesthetic to `post_avg`.

Include the maximum likelihood estimate (MLE) on the event probability as an additional `geom_point()` geom by mapping the `y` aesthetic to the correct value, which you must calculate.

Are there players with MLEs that are outside the posterior uncertainty interval? Are there players with posterior mean values that are quite close to the MLEs?

SOLUTION

```
summary_post_df_focus_from_vague %>%
  mutate(
    mle = ((anew -1) / (anew + bnew -2))
  ) %>%
  ggplot(mapping= aes(x = as.factor(player_id))) +
  geom_linerange(mapping = aes(ymin = post_q05, ymax = post_q95)) +
  geom_point(mapping = aes(y = post_avg),  color = "blue") +
  geom_point(mapping = aes( y =mle))
```



What do you think?
Yes, there are players with MLEs outside the the posterior uncertainty interval. The uncertainty interval shows where most of the observations will be concentrated so points outside the interval are possible, just very unlikely. However, there are also some players whose MLEs are quite close to the posterior mean values.

1g)

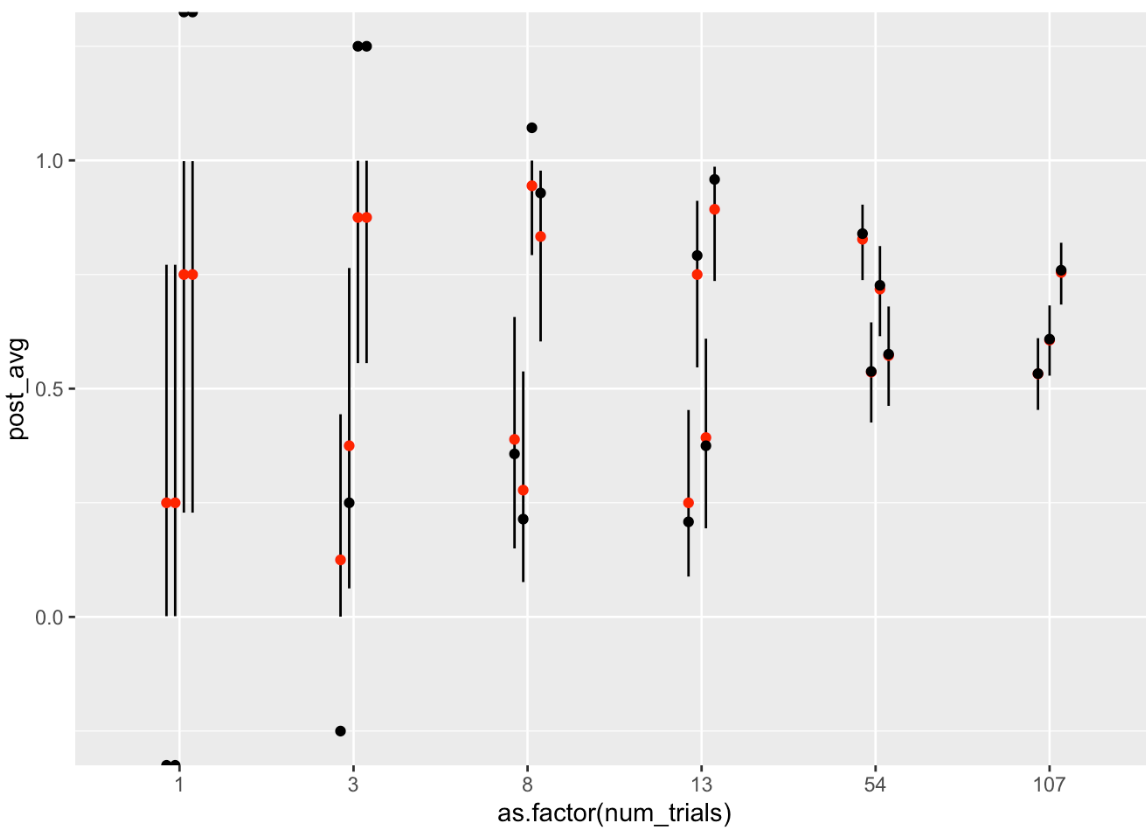
You will create a similar visualization to that from Problem 1f), except instead of mapping the `x` aesthetic to `as.factor(player_id)` you will map the `x` aesthetic to `as.factor(num_trials)`. You must also map the `group` aesthetic in each geom to the `player_id` variable. Doing so allows you to "dodge" the posterior summaries for each player associated with each `num_trials` value.

To properly apply the dodging, set the `position` argument to be `position = position_dodge(0.2)` in `geom_linerange()` and both `geom_point()` calls. You should not place `position` inside `aes()`, it should be outside `aes()`.

Based on your visualization, which players have high posterior uncertainty in the event probability?

SOLUTION

```
summary_post_df_focus_from_vague %>%
   mutate(
    mle = ((anew -1) / (anew + bnew -2))
  ) %>%
  ggplot(mapping= aes(x = as.factor(num_trials))) +
  geom_linerange(mapping = aes(ymin = post_q05, ymax = post_q95, group = player_id), position = position_d
odge(0.2)) +
  geom_point(mapping = aes(y = post_avg, group = player_id), position = position_dodge(0.2), color = "red"
) +
  geom_point(mapping = aes( y =mle, group = player_id), position = position_dodge(0.2))
```



What do you think?

The players with high posterior uncertainty are the ones in the group with the least number of trials.

Problem 02

In Problem 01, you estimated the unknown event probability for each player separately from all other players. Essentially, you were focused on one player at a time. This style of analysis is known as an **unpooled estimate**, since you are not combining or "pooling" the players (or in general terms the "groups") together.

The opposite view point is to **completely pool** all players together in order to estimate a single unknown event probability μ. For this, you will assume that all players are independent of the others, thus the posterior distribution on the unknown "pooled" event probability, μ, is proportional to:

$$p\left(\mu \mid \left((m, N)_j\right)_{j=1}^{J}\right) \propto \prod_{j=1}^{J} \left(\text{Binomial}\left(m_j \mid \mu, N_j\right)\right) \times \text{Beta}\left(\mu \mid a, b\right)$$

Pay close attention to the subscripts in the above expression. And notice that the prior on the "pooled" unknown μ relies on the prior shape parameters a and b.

2a)

Write out the log-posterior on the pooled unknown μ up to a normalizing constant in terms of the observations, m_j and N_j for $j = 1, \ldots, J$, and the prior shape parameters, a and b. Your result should contain a summation series over the J players.

SOLUTION

Add as many equation blocks as you feel are necessary to show the steps to derive the answer.

$$p\left(\mu \mid \left((m, N)_j\right)_{j=1}^{J}\right) \propto \prod_{j=1}^{J} \left(\text{Binomial}\left(m_j \mid \mu, N_j\right)\right) \times \text{Beta}\left(\mu \mid a, b\right)$$

$$\log(\prod_{j=1}^{J} \left(\text{Binomial}\left(m_j \mid \mu, N_j\right)\right) \times \text{Beta}\left(\mu \mid a, b\right))$$

$$\log(\prod_{j=1}^{J} \left(\text{Binomial}\left(m_j \mid \mu, N_j\right)\right)) + \log(\text{Beta}\left(\mu \mid a, b\right))$$

$$\log(\prod_{j=1}^{J} \left(\text{Binomial}\left(m_j \mid \mu, N_j\right)\right))$$

$$\sum_{j=1}^{J} \log(\text{Binomial}(\left(m_j \mid \mu, N_j\right)))$$

$$\sum_{j=1}^{J} \log(\mu^{m_j} \times (1-\mu)^{N_j - m_j})$$

$$\sum_{j=1}^{J} [\, m_j log(\mu) + (N_j - m_j)\log(1-\mu)]$$

$$\begin{gathered} \log(\text{Beta}\left(\mu \mid a, b\right)) \\ \log(\mu^{a-1} \times (1-\mu)^{b-1}) \\ \log(\mu^{a-1}) + \log((1-\mu)^{b-1}) \\ (a-1)\log(\mu) + (b-1)\log(1-\mu) \end{gathered}$$

$$\log(\prod_{j=1}^{J} \left(\text{Binomial}\left(m_j \mid \mu, N_j\right)\right) \times \text{Beta}\left(\mu \mid a, b\right)) = \sum_{j=1}^{J} [\, m_j log(\mu) + (N_j - m_j)\log(1-\mu)] + (a-1)\log(\mu) + (b-1)\log(1-\mu)$$

2b)

The summation series in your solution to 2a) can be simplified by using the average number of events, $\bar{m}$ and the average number of trials $\bar{N}$. The average number of events is defined as:

$$\bar{m} = \frac{1}{J} \sum_{j=1}^{J} \left(m_j\right)$$

and the average number of trials is defined as:

$$\bar{N} = \frac{1}{J} \sum_{j=1}^{J} \left(N_j\right)$$

Write your result from 2a) in terms of $\bar{m}$, $\bar{N}$, J, and the prior shape parameters a and b.

SOLUTION

Add as many equation blocks as you feel are necessary to show the steps to derive the answer.

$$\bar{m} = \frac{1}{J} \sum_{j=1}^{J} \left(m_j\right)$$

$$J\bar{m} = \sum_{j=1}^{J} \left(m_j\right)$$

$$\bar{N} = \frac{1}{J} \sum_{j=1}^{J} \left(N_j\right)$$

$$J\bar{N} = \sum_{j=1}^{J} \left(N_j\right)$$

$$\sum_{j=1}^{J} [\ m_j log(\mu) + (N_j - m_j)\log(1-\mu)] + (a-1)\log(\mu) + (b-1)\log(1-\mu)$$

$$\sum_{j=1}^{J} (\ m_j log(\mu)) + \sum_{j=1}^{J} ((N_j - m_j)\log(1-\mu)) + (a-1)\log(\mu) + (b-1)\log(1-\mu)$$

$$J\bar{m}\log(\mu) + (\sum_{j=1}^{J}(N_j) - \sum_{j=1}^{J}(m_j))\log(1-\mu) + (a-1)\log(\mu) + (b-1)\log(1-\mu)$$

$$J\bar{m}\log(\mu) + (\bar{N}J - \bar{m}J)(\log(1-\mu)) + (a-1)\log(\mu) + (b-1)\log(1-\mu)$$

2c)

Your expression in 2b) should look familiar.

What type of posterior distribution does the unknown “pooled” estimate μ have?

Write out the formulas for the posterior or updated hyperparameters for your specified posterior distribution.

SOLUTION

What do you think?

The posterior distribution is a beta distribution since the posterior will have the same functional form as the prior.

Add as many equation blocks as you feel are necessary to show the steps to derive the answer.

$$J\bar{m}\log(\mu) + (\bar{N}J - \bar{m}J)(\log(1-\mu)) + (a-1)\log(\mu) + (b-1)\log(1-\mu)$$

$$(J\bar{m} + a - 1)\log(\mu) + (\bar{N}J - \bar{m}J + b - 1)\log(1-\mu)$$

$$a_{new} = a + J\bar{m}$$

$$b_{new} = b + J\bar{N} - J\bar{m}$$

2d)

Based on your formula in Problem 2c), calculate the updated shape parameters for the 23 players in the `df_focus tibble`. You should add two columns using `mutate()` named `anew` and `bnew`. Assign your result to the `post_df_focus_pooled` object.

You will still assume a vague prior and thus use $a = b = 0.5$ as you did in Problem 01. And remember that we are pooling **all** players together to learn the pooled estimate.

SOLUTION

```
N_bar <- mean(df_focus$num_trials)
m_bar <- mean(df_focus$num_events)
J <- nrow(df_focus)
post_df_focus_pooled <- df_focus %>%
  mutate(
    anew = 0.5 + (J*m_bar),
    bnew = 0.5 + (J*N_bar) - (J*m_bar)
  )
```

2e)

Calculate the posterior mean, 5th quantile, and 95th quantile for each player in `post_df_focus_pooled`. You should add 3 columns using `mutate()` named `post_avg`, `post_q05`, and `post_q95`. Assign the result to the variable `summary_post_df_focus_pooled`.

SOLUTION

```
summary_post_df_focus_pooled <- post_df_focus_pooled %>%
  mutate(
    post_avg = anew / (anew + bnew),
    post_q05 = qbeta(0.05, anew, bnew),
    post_q95 = qbeta(0.95, anew, bnew)
  )
```

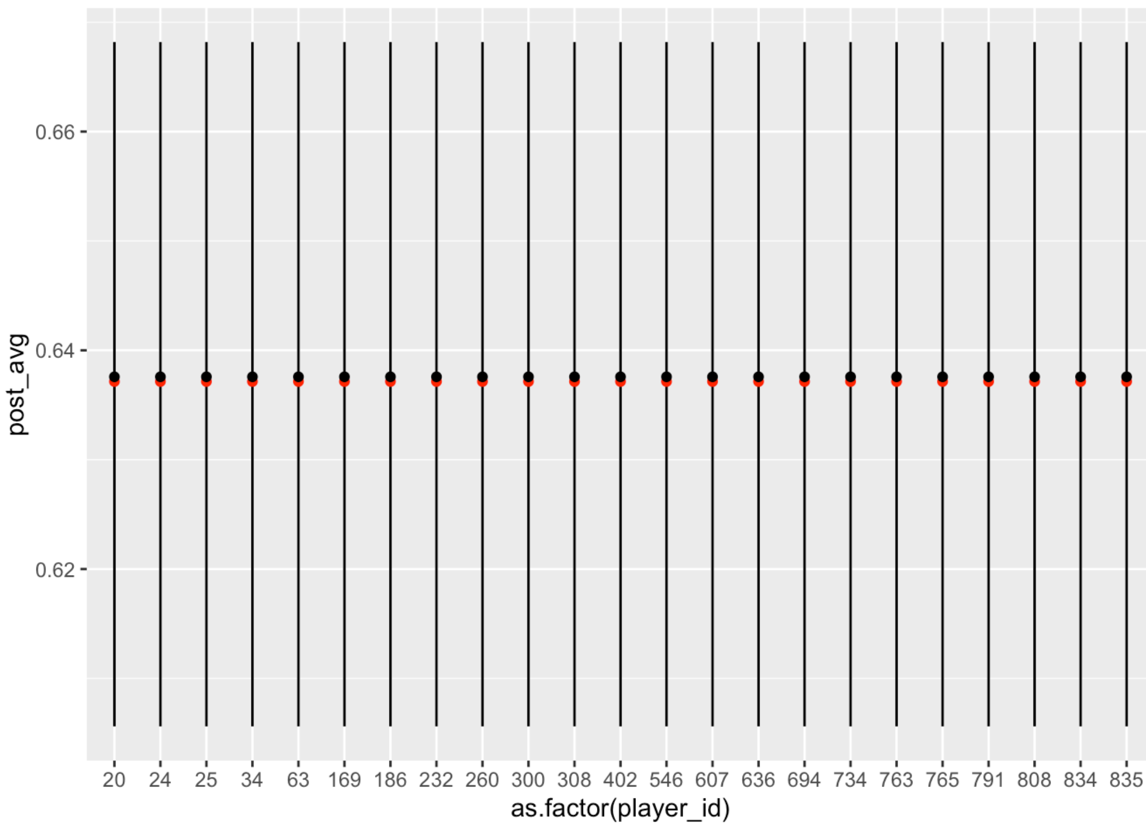
2f)

Pipe `summary_post_df_focus_pooled` into `ggplot()` and map the `x` aesthetic to `as.factor(player_id)`. You will use the `geom_linerange()` to represent the posterior uncertainty by setting the `ymin` and `ymax` aesthetics to `post_q05` and `post_q95` respectively. You will display the posterior mean with a `geom_point()` by setting the `y` aesthetic to `post_avg`. Include the maximum likelihood estimate (MLE) on the event probability as an additional `geom_point()` geom by mapping the `y` aesthetic to the correct value, which you must calculate.

Are there players with MLEs that are outside the posterior uncertainty interval? Are there players with posterior mean values that are quite close to the MLEs?

SOLUTION

```
summary_post_df_focus_pooled %>%
    mutate(
    mle = ((anew -1) / (anew + bnew -2))
  ) %>%
  ggplot(mapping= aes(x = as.factor(player_id))) +
  geom_linerange(mapping = aes(ymin = post_q05, ymax = post_q95)) +
  geom_point(mapping = aes(y = post_avg), color = 'red') +
  geom_point(mapping = aes( y =mle))
```



What do you think?
No, there are not any players with posterior mean values outside the posterior uncertainity level. All of the players have posterior mean values close to the MLEs.

2g)

Your visualization in Problem 2f) should not “feel right”. Something should seem off.

Why does the “pooled” estimate seem incorrect for this application?

SOLUTION

What do you think?
I believe that the pooled estimate seems incorrect for this application because we are calculating the posterior mean for each player by using the average number of events and average number of trials. However, this does not seem to accurately reflect each individual

players means. For this application it makes more sense to use the each players individual trials and events as the number of trials and events can vary a lot throughout the players.

Problem 03

You have now worked through two extremes, the **unpooled** and the completely **pooled** estimates on the unknown event probabilities. You will now try to blend the two approaches to reach a compromise by using the Empirical Bayes approach.

As stated at the beginning of the document, Empirical Bayes estimates the prior from data. In this setting you are interested in deciding informative values for the prior shape hyperparameters, a and b, of the Beta prior on each μ_j. If you have a relevant informative prior you will be able to apply that prior to each player separately (the unpooled approach) while "borrowing strength" from the rest of the data. The Empirical Bayes approach is an approximation to more formal *partial pooling* models where groups with larger sample sizes help estimate parameters associated with small sample size groups. Empirical Bayes is useful when there are hundreds to thousands of separate groups. Estimating the prior hyperparameters from many groups allows specifying relevant informative priors without requiring numerous conversations with Subject Matter Experts (SMEs) and allows the data to provide representative bounds.

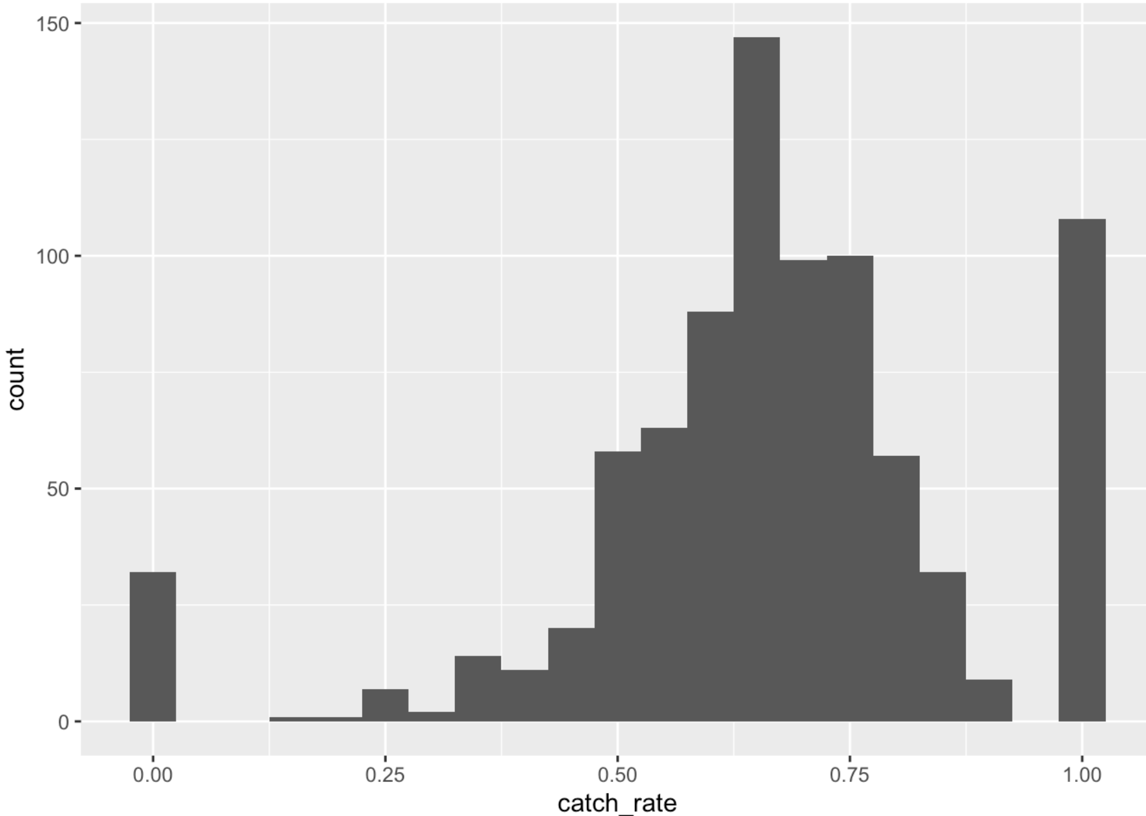
3a)

The Beta prior defines the prior belief on a probability (a fraction). From an Empirical Bayes approach, you can therefore view the "data" of interest as the observed "catch rate".

Plot the histogram of the "catch rate" for all players in the `df_all` data set. Use the `geom_histogram()` geom and set the `binwidth` to be 0.05.

SOLUTION

```
df_all %>%
  mutate(
    catch_rate = num_events / num_trials
  ) %>%
  ggplot(mapping = aes( x = catch_rate)) + geom_histogram(binwidth = 0.05)
```



3b)

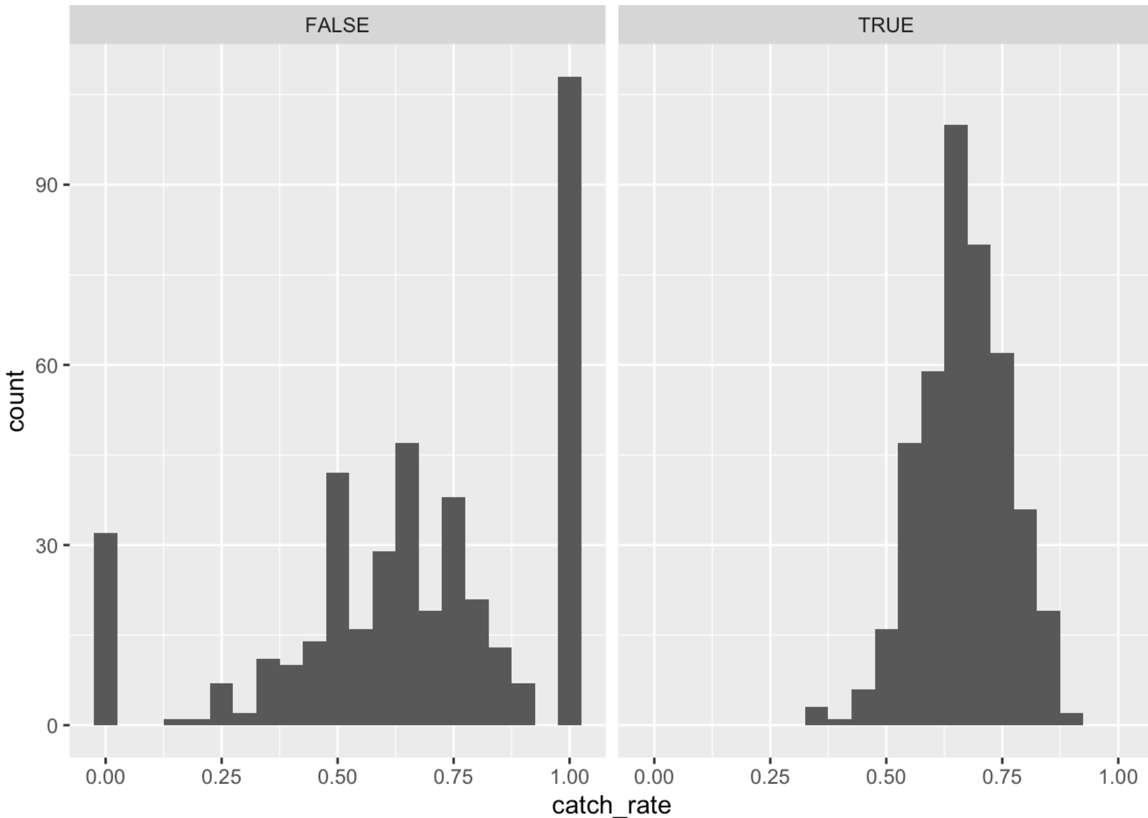
Plot the histogram for all “catch rates” in the `df_all` data set again. However, this time use `facet_wrap()` to break up the visualization into `num_trials > 24`.

What can you say about the observations of the players with greater than 25 targets?

SOLUTION

What do you think?

```
df_all %>%
  mutate(
    catch_rate = num_events/num_trials
  ) %>%
  ggplot(mapping = aes(x = catch_rate)) + geom_histogram(binwidth = 0.05)  + facet_wrap(df_all$num_trials
> 24)
```



The observations of the players with greater than 25 targets is fairly normal with the mode centered around a catch rate of 0.6-0.7.

3c)

To keep things simple for now, you will estimate the prior parameters, a and b, based only on the players with greater than 24 targets.

Use the `filter()` function to keep all players with greater than 24 targets and assign the result to the `df_24` object. Use the `summary()` function to check the summary stats on `num_trials` to make sure you performed the operation correctly.

SOLUTION

```
df_24 <- df_all %>%
  filter(num_trials > 24)

summary(df_24)
```

```
##    player_id       num_trials      num_events
##  Min.   :  3.0   Min.   : 25.0   Min.   : 10.00
##  1st Qu.:204.0   1st Qu.: 47.0   1st Qu.: 31.00
##  Median :406.0   Median : 83.0   Median : 56.00
##  Mean   :406.5   Mean   :118.0   Mean   : 78.94
##  3rd Qu.:599.0   3rd Qu.:155.5   3rd Qu.:103.00
##  Max.   :841.0   Max.   :509.0   Max.   :365.00
```

3d)

Since the “catch rate” is a fraction, we can use a Beta distribution as the likelihood of the “fraction” given the shape parameters. Those shape parameters, a and b, are unknown and so you must estimate them from the data. Within the Empirical Bayes approach, you will treat this step as finding a and b which **maximize the likelihood**, and so you will not specify prior distributions on the parameters.

Each observation of the "catch rate" is assumed conditionally independent given the unknown a and b shape parameters. The observed "catch rate" will be denoted as, θ_j, for each player and is defined as:

$$\theta_j = \frac{m_j}{N_J}$$

The likelihood on all $j = 1, \ldots, J$ catch rates is therefore the product of J conditionally independent Beta distributions:

$$p\left(\left(\theta_j\right)_{j=1}^{J} \mid a, b \right) = \prod_{j=1}^{J} \mathrm{Beta}\left(\theta_j \mid a, b\right)$$

You will define a log-likelihood function in the style of the log-posterior functions we have used so far this semester by completing the two code chunks below.

In the first code chunk, the list of required information, `info_for_ab`, is defined and contains a single variable `theta`. You must calculate it based on the players in the `df_24` data set.

The second code chunk defines the `my_beta_loglik()` function. The first argument, `unknowns`, is the vector of unknown parameters. The second argument, `my_info`, is the list of required information. The comments and variable names provide hints for actions you should perform to calculate the log-likelihood.

The a and b parameters are lower-bounded at zero and thus you must apply the log-transformation to both parameters. You must properly account for the log-derivative adjustment on both parameters when you calculate the log-likelihood.

NOTE: Several test points are provided for you to check that you have coded your function correctly.

SOLUTION

Define the list of required information. The observed data in your `my_beta_loglik()` must be named `theta`.

```
df_24$catch_rate <- df_24$num_events / df_24$num_trials
info_for_ab <- list(
  theta = df_24$catch_rate
)
```

Define the Beta log-likelihood. The first element in `unknowns` is the log-transformed a parameter and the second element is the log-transformed b parameter. You are allowed to use built in density functions to complete this question.

```
my_beta_loglik <- function(unknowns, my_info)
{
  # unpack the log-transformed shape parameters
  log_a <- unknowns[1]
  log_b <- unknowns[2]


  # back transform
  a <- exp(log_a)
  b <- exp(log_b)


  # calculate the log-likelihood for all observations
  log_lik <- sum(dbeta(x = my_info$theta,
                      shape1 = a,
                      shape2 = b,
                      log = TRUE))
  # account for the change of variables
  return(log_lik + log_a + log_b)

}
```

Try out values of -2 for both log-transformed parameters. If your function is coded correctly you should get a value of -571.8519.

```
unknowns = c(-2,-2)
my_beta_loglik(unknowns, info_for_ab)
```

```
## [1] -571.8519
```

Try out values of 2.5 for both log-transformed parameters. If your function is coded correctly you should get a value of -254.3934.

```
unknowns = c(2.5,2.5)
my_beta_loglik(unknowns, info_for_ab)
```

```
## [1] -254.3934
```

3e)

You will now identify the maximum likelihood estimates for a and b. You should use the `optim()` function to manage the optimization for you. Be sure to specify the arguments to `optim()` to make sure that `optim()` knows to *MAXIMIZE* and not *MINIMIZE* the function. Set the `method` argument to `"BFGS"` when you call `optim()`. The gradient argument should be set to `NULL`, `gr=NULL`.

Try out two different starting guesses values. The first guess, `init_guess_01`, should be zeros for both parameters and the second guess, `init_guess_02`, should be -1 for both parameters.

Assign your `optim()` results to `log_ab_opt_01` and `log_ab_opt_02`.

Do you get the same parameter estimates regardless of your initial guess?

SOLUTION

Set the initial guesses.

```
init_guess_01 <- c(0,0)
init_guess_02 <- c(-1,-1)
```

Perform the optimization using the first starting guess.

```
log_ab_opt_01 <- optim(init_guess_01,
                      my_beta_loglik,
                      gr = NULL,
                      info_for_ab,
                      method = "BFGS",
                      hessian = TRUE,
                      control = list(fnscale = -1, maxit = 1001))
log_ab_opt_01
```

```
## $par
## [1] 2.759740 2.058504
##
## $value
## [1] 410.5272
##
## $counts
## function gradient
##       34       11
##
## $convergence
## [1] 0
##
## $message
## NULL
##
## $hessian
##           [,1]      [,2]
## [1,] -2380.488  2305.533
## [2,]  2305.533 -2458.724
```

Perform the optimization using the second starting guess.

```
log_ab_opt_02 <- optim(init_guess_02,
                      my_beta_loglik,
                      gr = NULL,
                      info_for_ab,
                      method = "BFGS",
                      hessian = TRUE,
                      control = list(fnscale = -1, maxit = 1001))
log_ab_opt_02
```

```
## $par
## [1] 2.759736 2.058500
##
## $value
## [1] 410.5272
##
## $counts
## function gradient
##       35       11
##
## $convergence
## [1] 0
##
## $message
## NULL
##
## $hessian
##           [,1]      [,2]
## [1,] -2380.481  2305.524
## [2,]  2305.524 -2458.712
```

Are the identified log-transformed estimates the same?
Yes, the identified log-transformed estimates are the same regardless of initial guesses.

3f)

The optimal parameters in the Problem 3e) are in the log-transformed space.

You must back-transform them to calculate the estimates for the prior a and b shape hyperparameters. Assign the back-transformed parameters to `ab_emp_bayes`.

How many a-priori trials does your estimated hyperparameters represent?

SOLUTION

```
ab_emp_bayes <- exp(log_ab_opt_02$par)
ab_emp_bayes
```

```
## [1] 15.79568  7.83421
```

How many a-priori trials?
The estimated hyperparamters represent 35 trials.

3g)

You will now visualize the prior distribution you calculated using the Empirical Bayes approach and compare it to the histogram of the observed "catch rates" for all players with more than 24 targets.

Complete the two code chunks below. In the first, set the `x` variable within the `prior_for_viz` `tibble` to be 1001 evenly spaced points between the minimum observed catch rate in `df_24` and the maximum observed catch rate in `df_24`. Pipe the result into `mutate()` and calculate the beta density using the `ab_emp_bayes` shape hyperparameters and assign the result to the `beta_pdf` variable.

In the second code chunk, pipe the `df_24` `tibble` into `ggplot()` and map the `x` aesthetic to the observed catch rates. Use a `geom_histogram()` geom and set the `binwidth` to be 0.05. Modify the `y` aesthetic so that way `geom_histogram()` displays the estimated density on the `y` axis instead of the count. To do so you must set `y=stat(density)` within `aes()`. Include a

`geom_line()` geom and specify the `data` argument to be the `prior_for_viz` object and map the `x` and `y` aesthetics to `x` and `beta_pdf`, respectively. Set the `color` argument (outside the `aes()` call) to be `'red'` and the `size` argument to 1.15.

How does the empirically derived prior distribution on the event probability compare to the observed histogram of the catch rates?

IMPORTANT: If you are *not* comfortable with your `ab_emp_bayes` values, you may use `shape1=13` and `shape2=8`. These are **not** the correct answers, though they are in the right ballpark…

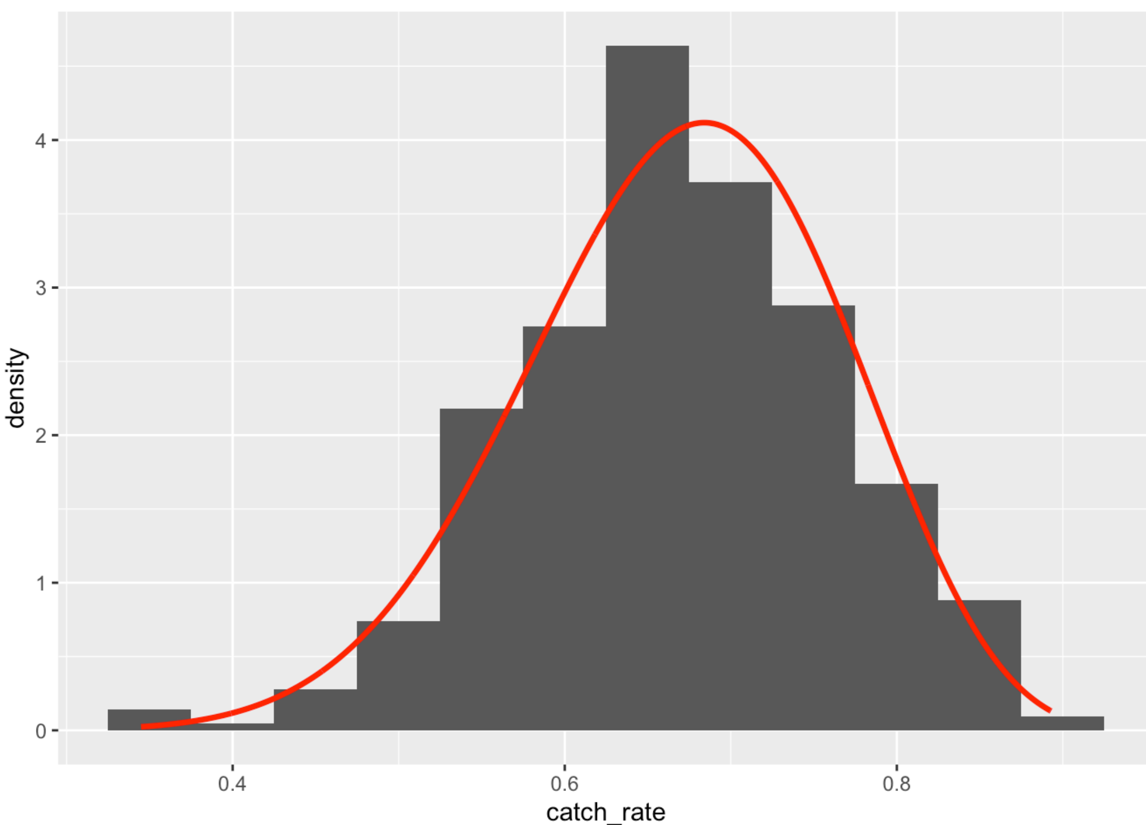
SOLUTION

Calculate the Beta PDF based on the calculated prior hyperparameters.

```
prior_for_viz <- tibble::tibble(
  x = seq(min(df_24$catch_rate), max(df_24$catch_rate), length.out = 1001)
) %>%
  mutate(beta_pdf = dbeta (x, ab_emp_bayes[1], ab_emp_bayes[2]))
```

Visualize the derived prior relative to the observed "catch rates" in the data set.

```
df_24 %>%
  ggplot(mapping = aes(x = catch_rate)) +
  geom_histogram(binwidth = 0.05, mapping = aes ( y = stat(density))) +
  geom_line(data = prior_for_viz,
            mapping = aes(x = x, y = beta_pdf ), color = 'red', size = 1.15)
```



The mode of the empirically derived prior distribution on the event probability is slightly greater than the mode of the observed histogram of the catch rates.

3h)

Calculate the 5th and 95th quantiles associated with your informative prior.

IMPORTANT: If you are *not* comfortable with your `ab_emp_bayes` values, you may use `shape1=13` and `shape2=8` . These are **not** the correct answers, though they are in the right ballpark…

SOLUTION

```
prior_0.05 <- qbeta(0.05, ab_emp_bayes[1], ab_emp_bayes[2])
prior_0.95 <- qbeta(0.95, ab_emp_bayes[1], ab_emp_bayes[2])
prior_0.05
```

```
## [1] 0.5042065
```

```
prior_0.95
```

```
## [1] 0.8161721
```

Problem 04

You now have everything in place to calculate the posterior on the event probability associated with each player, μ_j. The a and b parameters that you had originally set to both be 0.5, are now equal to your Empirical Bayes estimated values.

If you are not comfortable with your estimates you may use the same values as in Problem 3g) of `shape1=13` and `shape2=8` .

4a)

Calculate the updated or new shape parameters for the players in the `df_focus tibble` . You should add two columns using `mutate()` named `anew` and `bnew` . Assign your result to the `post_df_focus_empbayes` object.

SOLUTION

```
post_df_focus_empbayes <- df_focus %>%
  mutate(
    anew = ab_emp_bayes[1] + num_events,
    bnew = ab_emp_bayes[2] + (num_trials - num_events)
  )
```

4b)

Calculate the posterior mean, 5th quantile, and 95th quantile for each player in `post_df_focus_empbayes` . You should add 3 columns using `mutate()` named `post_avg` , `post_q05` , and `post_q95` . Assign the result to the variable `summary_post_df_focus_empbayes` .

SOLUTION

```
summary_post_df_focus_empbayes <- post_df_focus_empbayes %>%
  mutate(
    post_avg = anew / (anew + bnew),
    post_q05 = qbeta(0.05, anew, bnew),
    post_q95 = qbeta(0.95, anew, bnew)
  )
```

4c)

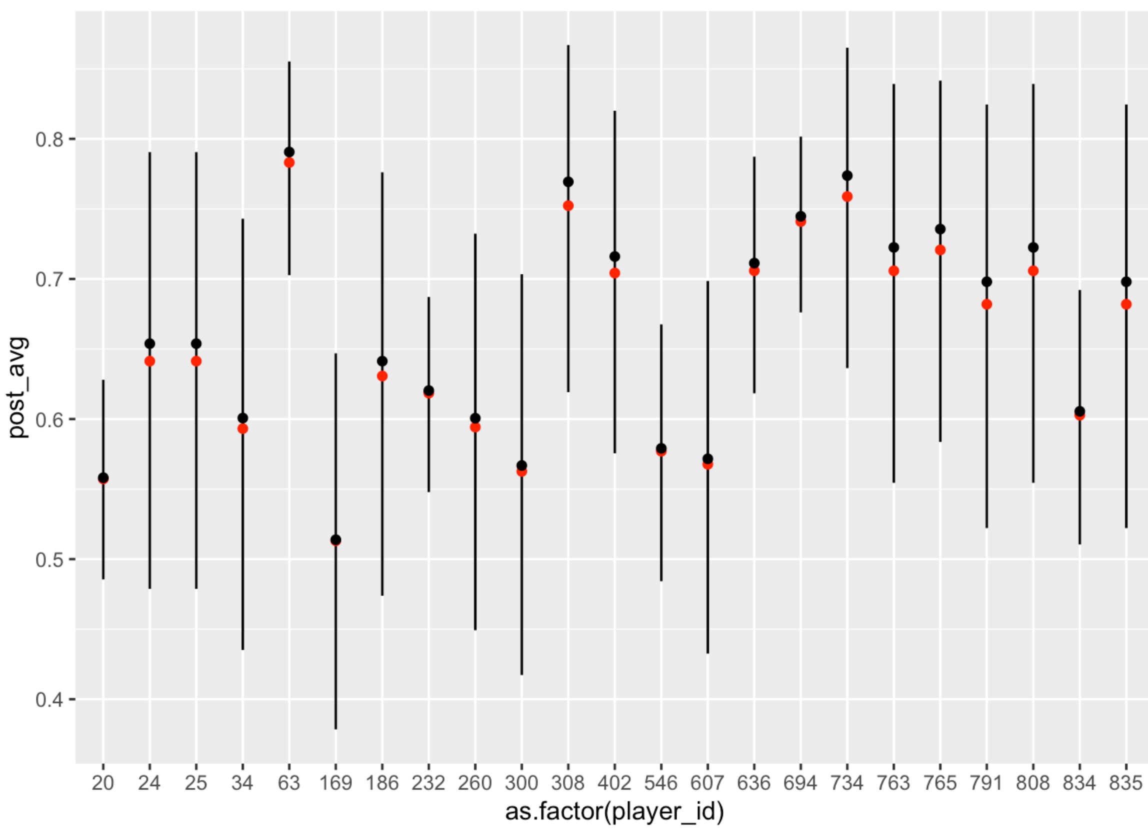
You will repeat the visualizations from Problem 1) to understand the effect of your informative prior distribution.

Pipe `summary_post_df_focus_empbayes` into `ggplot()` and map the `x` aesthetic to `as.factor(player_id)`. You will use the `geom_linerange()` to represent the posterior uncertainty by setting the `ymin` and `ymax` aesthetics to `post_q05` and `post_q95` respectively. You will display the posterior mean with a `geom_point()` by setting the `y` aesthetic to `post_avg`. Include the maximum likelihood estimate (MLE) on the event probability as an additional `geom_point()` geom by mapping the `y` aesthetic to the correct value, which you must calculate.

How does this visualization compare to those you made using the vague unpooled estimate and the completely pooled estimate?

SOLUTION

```
summary_post_df_focus_empbayes %>%
  mutate(
    mle = ((anew -1) / (anew + bnew -2))
     ) %>%
  ggplot(mapping = aes(x = as.factor(player_id))) +
  geom_linerange(mapping = aes (ymin = post_q05, ymax = post_q95)) +
  geom_point(mapping = aes(y = post_avg), color = 'red') +
  geom_point(mapping = aes( y = mle ))
```



This visualization compares to those made using the vague unpooled estimate and the completely pooled estimate in that all the MLEs and posterior mean are pretty close to each other when using the Empirical Bayes, which was not the case for the other two strategies. Also, the uncertainity levels in the Empirical Bayes are smaller than in the other two visualizations.

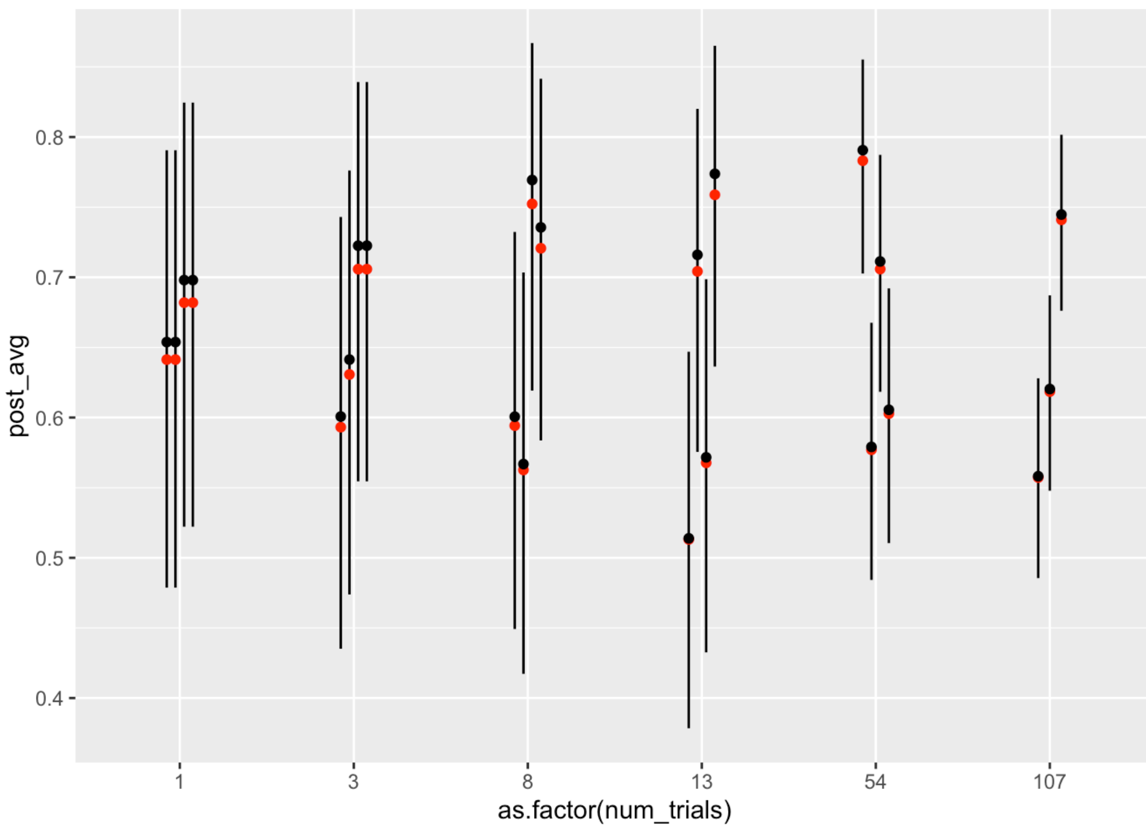
4d)

You will create a similar visualization, except instead of mapping the `x` aesthetic to `as.factor(player_id)` you will map the `x` aesthetic to `as.factor(num_trials)`. You must also map the `group` aesthetic in each geom to the `player_id` variable. Doing so allows you "dodge" the posterior summaries for each player associated with each `num_trials` value.

To properly apply the dodging, set the `position` argument to be `position = position_dodge(0.2)` in `geom_linerange()` and both `geom_point()` calls. You should not place `position` inside `aes()`, it should be outside `aes()`.

SOLUTION

```
summary_post_df_focus_empbayes %>%
  mutate(
    mle = ((anew -1) / (anew + bnew -2))
     ) %>%
  ggplot(mapping = aes(x = as.factor(num_trials))) +
  geom_linerange(mapping = aes (ymin = post_q05, ymax = post_q95, group = player_id), position = position_
dodge(0.2)) +
  geom_point(mapping = aes(y = post_avg, group = player_id), color = 'red',
             position = position_dodge(0.2))+
  geom_point(mapping = aes( y = mle, group = player_id ), position = position_dodge(0.2))
```



4e)

You will now calculate the posteriors for *all* players using the Empirical Bayes approach, not just the limited number of players in the "focused" data set.

Calculate the updated shape parameters for all players in the `df_all tibble`. You should add two columns using `mutate()` named `anew` and `bnew`. Assign your result to the `post_df_all_empbayes` object.

SOLUTION

```
post_df_all_empbayes <- df_all %>%
    mutate(
    anew = ab_emp_bayes[1] + num_events,
    bnew = ab_emp_bayes[2] + (num_trials - num_events))
```

4f)

Calculate the posterior mean, 5th quantile, and 95th quantile for each player in `post_df_all_empbayes`. You should add 3 columns using `mutate()` named `post_avg`, `post_q05`, and `post_q95`. Assign the result to the variable `summary_post_df_all_empbayes`.

```
summary_post_df_all_empbayes <- post_df_all_empbayes %>%
  mutate(
    post_avg = anew / (anew + bnew),
    post_q05 = qbeta(0.05, anew, bnew),
    post_q95 = qbeta(0.95, anew, bnew)
  )
```

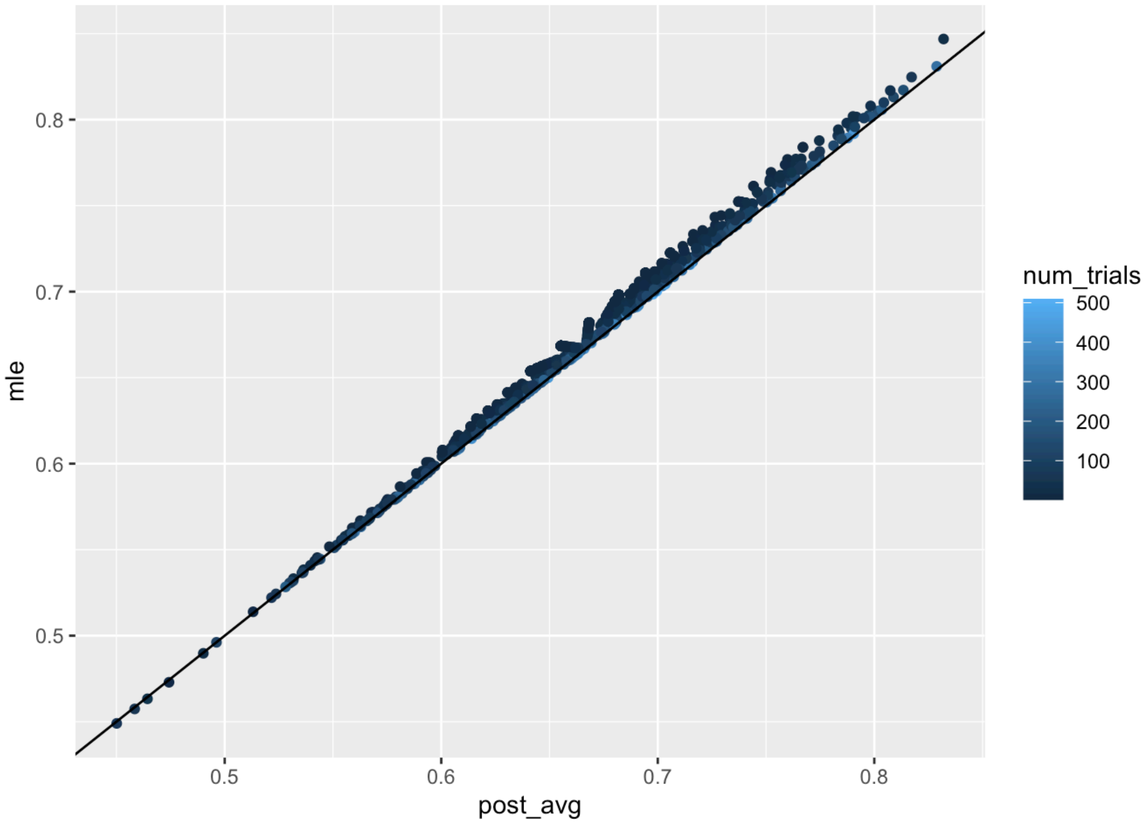
4g)

You will now visualize the posterior mean, based on the Empirical Bayes informative prior, relative to the maximum likelihood estimate for the event probability.

Create a scatter plot with `ggplot2` where you plot the `post_mean` with respect to the maximum likelihood estimate to the unknown event probability for all players. Map the `color` aesthetic to `num_trials` and include a `geom_abline()` layer with `slope = 1` and `intercept=0`.

SOLUTION

```
summary_post_df_all_empbayes %>%
  mutate(
    mle = ((anew -1) / (anew + bnew -2))
     ) %>%
  ggplot(mapping = aes(x = post_avg, y =mle )) + geom_point(mapping = aes(color = num_trials)) + geom_abli
ne(slope = 1, intercept = 0)
```

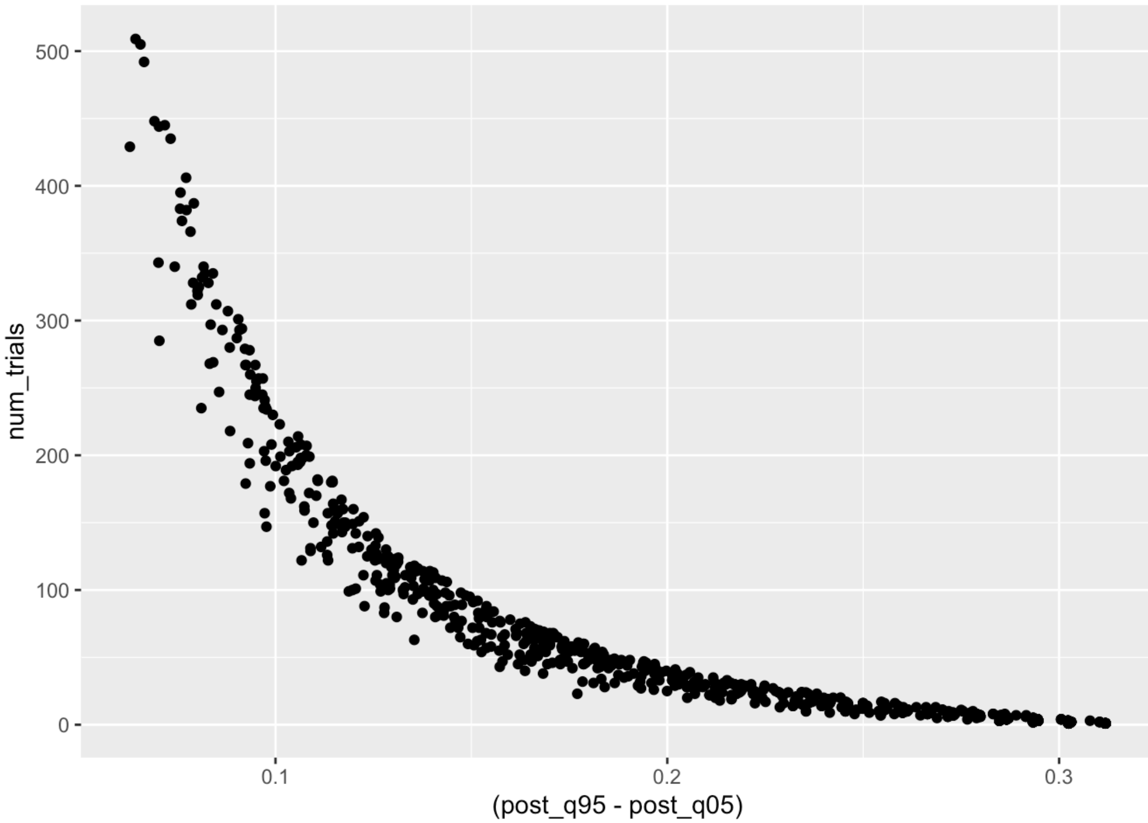


4h)

Create a scatter plot for the middle 90% uncertainty interval range (difference between the 95th and 5th quantiles) with respect to the `num_trials` using `ggplot2`.

SOLUTION

```
summary_post_df_all_empbayes %>%
  ggplot(mapping = aes(x = (post_q95 - post_q05), y = num_trials)) + geom_point()
```



4i)

Based on your visualizations in this exam, discuss how an informative prior influences posterior when the sample size is small compared with large sample sizes.

SOLUTION

What do you think?

An informative prior influences the posterior when the sample size is large has a smaller influence than when the smaller size is small. With the larger sample size the posterior distribution follows the likelihood more than the prior. However, when there is a small sample size, there is more uncertainty and the prior has more influence on the posterior.

Problem 05

Now that you have posterior distributions based on an informative prior for every player in the data set, it is time to consider answering a question the NFL team is interested in. The team wants to identify the best receivers in the data set, and it wants to be confident in that selection. Your Bayesian analysis allows answering probabilistic questions. You will answer several such questions now.

5a)

Calculate the probability that each player has a catch rate (event probability) of greater than 0.67. Add a column to the `summary_post_df_all_empbayes` object named `prob_grt_67`. Assign the result to a new variable `post_player_eval`.

SOLUTION

```
summary_post_df_all_empbayes['prob_grt_67'] <-  pbeta(0.67, summary_post_df_all_empbayes$anew, summary_pos
t_df_all_empbayes$bnew, lower.tail = FALSE )
post_player_eval <- summary_post_df_all_empbayes
```

5b)

Identify the top 10 players based on the posterior probability that their catch rate is greater than 0.67. What do these players all have in common, besides the `prob_grt_67` value?

SOLUTION

```
post_player_eval %>% arrange(desc(prob_grt_67)) %>% head(10)
```

```
## # A tibble: 10 × 9
##    player_id num_trials num_events  anew  bnew post_avg post_q05 post_q95
##        <dbl>      <dbl>      <dbl> <dbl> <dbl>    <dbl>    <dbl>    <dbl>
##  1       409        285        240  256.  52.8    0.829    0.792    0.863
##  2       353        429        342  358.  94.8    0.790    0.758    0.821
##  3       498        343        273  289.  77.8    0.788    0.752    0.822
##  4       467        235        192  208.  50.8    0.803    0.762    0.843
##  5       382        147        123  139.  31.8    0.813    0.762    0.860
##  6       493        179        146  162.  40.8    0.798    0.751    0.843
##  7       447        157        129  145.  35.8    0.802    0.751    0.848
##  8       278        122        102  118.  27.8    0.809    0.753    0.860
##  9       567        218        171  187.  54.8    0.773    0.728    0.816
## 10       302        340        258  274.  89.8    0.753    0.715    0.789
## # … with 1 more variable: prob_grt_67 <dbl>
```

Besides the prob_grt_67 being equal or rounding up to 1, the top 10 players all have in common that the range of values between the 5th and 95th quantile is greater than 0.67.

5c)

Identify the 10 players with the lowest posterior probability that their catch is greater than 0.67. What is the smallest number of targets (trial size) associated with these 10 players?

SOLUTION

```
post_player_eval %>% arrange(prob_grt_67) %>% head(10)
```

```
## # A tibble: 10 × 9
##    player_id num_trials num_events  anew  bnew post_avg post_q05 post_q95
##        <dbl>      <dbl>      <dbl> <dbl> <dbl>    <dbl>    <dbl>    <dbl>
##  1       545        207        106 122.  109.     0.528    0.474    0.582
##  2       222         67         25  40.8  49.8    0.450    0.365    0.536
##  3       837        113         52  67.8  68.8    0.496    0.426    0.566
##  4       624        301        166 182.  143.     0.560    0.515    0.605
##  5        61        181         93 109.   95.8    0.532    0.474    0.589
##  6       435         61         23  38.8  45.8    0.458    0.370    0.548
##  7       195        294        163 179.  139.     0.563    0.517    0.608
##  8       615        180         95 111.   92.8    0.544    0.487    0.601
##  9       559        214        117 133.  105.     0.559    0.506    0.611
## 10       506         47         17  32.8  37.8    0.464    0.368    0.562
## # … with 1 more variable: prob_grt_67 <dbl>
```

The smallest number of targets associated with the 10 players with the lowest posterior probability that their catch rate is greater than 0.67 is 47.

5d)

A player with a large sample size could mean that player is well known, especially around the NFL. The team is interested in identifying players that are not as well known, and yet seem to have high catch rates.

Identify 10 players with the smallest sample sizes (number of trials) while still having `prob_grt_67` values greater than 0.75.

SOLUTION

```
post_player_eval %>%
  filter(prob_grt_67 > 0.75) %>%
  arrange(num_trials) %>%
  head(10)
```

```
## # A tibble: 10 × 9
##    player_id num_trials num_events  anew  bnew post_avg post_q05 post_q95
##        <dbl>      <dbl>      <dbl> <dbl> <dbl>    <dbl>    <dbl>    <dbl>
##  1       701          5          5  20.8  7.83    0.726    0.583    0.852
##  2        81          7          7  22.8  7.83    0.744    0.608    0.862
##  3        88          7          7  22.8  7.83    0.744    0.608    0.862
##  4       308          8          8  23.8  7.83    0.752    0.619    0.867
##  5        41          9          9  24.8  7.83    0.760    0.630    0.871
##  6       606          9          8  23.8  8.83    0.729    0.595    0.847
##  7       264         10          9  24.8  8.83    0.737    0.606    0.852
##  8       321         10          9  24.8  8.83    0.737    0.606    0.852
##  9       376         10         10  25.8  7.83    0.767    0.640    0.876
## 10       476         10          9  24.8  8.83    0.737    0.606    0.852
## # … with 1 more variable: prob_grt_67 <dbl>
```

5e)

Why do you think the questions in this problem were focused on calculating the probability that the catch rate is greater than 0.67? What is the interpretation of such a question?

HINT: Consider the interpretation of the completely pooled estimate.

SOLUTION

What do you think? Having a catch rate of greater than 0.67 means that the player is catching about 2/3 of the throws thrown to him. This helps take into account the number of trials for each individual player, that the completely pooled method did not take into account. Now, we can remove some of the bias of newer players who did not get thrown the ball as often being seen as not as good as older/more experienced players.