The Ultimate Halloween Candy Power Ranking

In this exercise, you will work with data that was collected for a fivethirtyeight article. You can download the data from a Github repository (link below). It is also accessible in library(fivethirtyeight), where it is available as data(candy_rankings). You can get some information on the variables by typing ?candy_rankings.

Article: https://fivethirtyeight.com/features/the-ultimate-halloween-candy-power-ranking/
Github repo: https://github.com/fivethirtyeight/data/tree/master/candy-power-ranking

```
library(dplyr)
library(tidyr)
library(ggplot2)

library(fivethirtyeight)
data(candy_rankings)
str(candy_rankings)
head(candy_rankings)
```

Firstly, we load the important libraries we need for completing this exercise. Many of which would be useful for other questions in the exercise.

Then we also load the library and the data set that contains all the info we need to work on.

1. Find the top 5 best rated and top 5 worst rated candy.

Explanation: The first thing we do it is to select the parameters we need for identifying top 5 and worst 5 rated candies. So, we pass onto the 'select' function from dplyr package the 'competition name' and the win percent and then arrange it by winpercent.

Once the dataframe is arranged in the form we want for analysis, use **head and tail function** to call the top 5 best and top 5 worst rated candies.

R code and the output:

```
candy_rank <- candy_rankings %>% select(competitorname, winpercent) %>% arran
ge(desc(winpercent))

head(candy_rank, n=5)
```

```
# A tibble: 5 x 2
  competitorname            winpercent
  <chr>                          <dbl>
1 Reese's Peanut Butter cup       84.2
2 Reese's Miniatures              81.9
3 Twix                            81.6
```

```
4 Kit Kat                              76.8
5 Snickers                             76.7
```

```
tail(candy_rank, n=5)
```

```
# A tibble: 5 x 2
  competitorname     winpercent
  <chr>                   <dbl>
1 Jawbusters               28.1
2 Super Bubble             27.3
3 Chiclets                 24.5
4 Boston Baked Beans       23.4
5 Nik L Nip                22.4
```

2. Plot winpercent against sugarpercent. Do you see any association? Now, plot winpercent against pricepercent. Do you see any association?

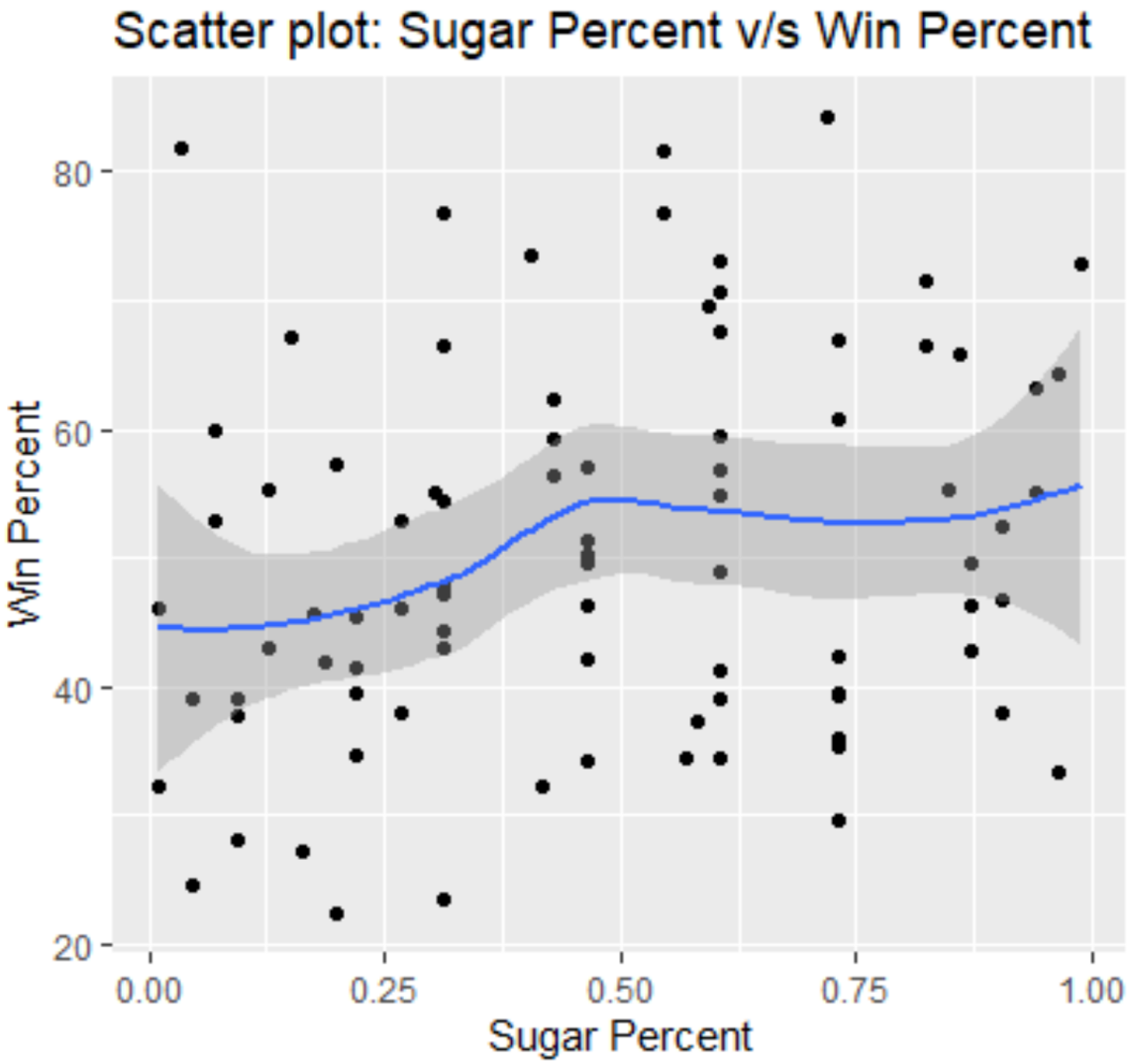
Explanation: We use the ggplot2 library to call the ggplot function and geom_point (geometry function for scatter plot) to plot sugar percent against the win percent.

We repeat the procedure for plotting win percent against price percent.

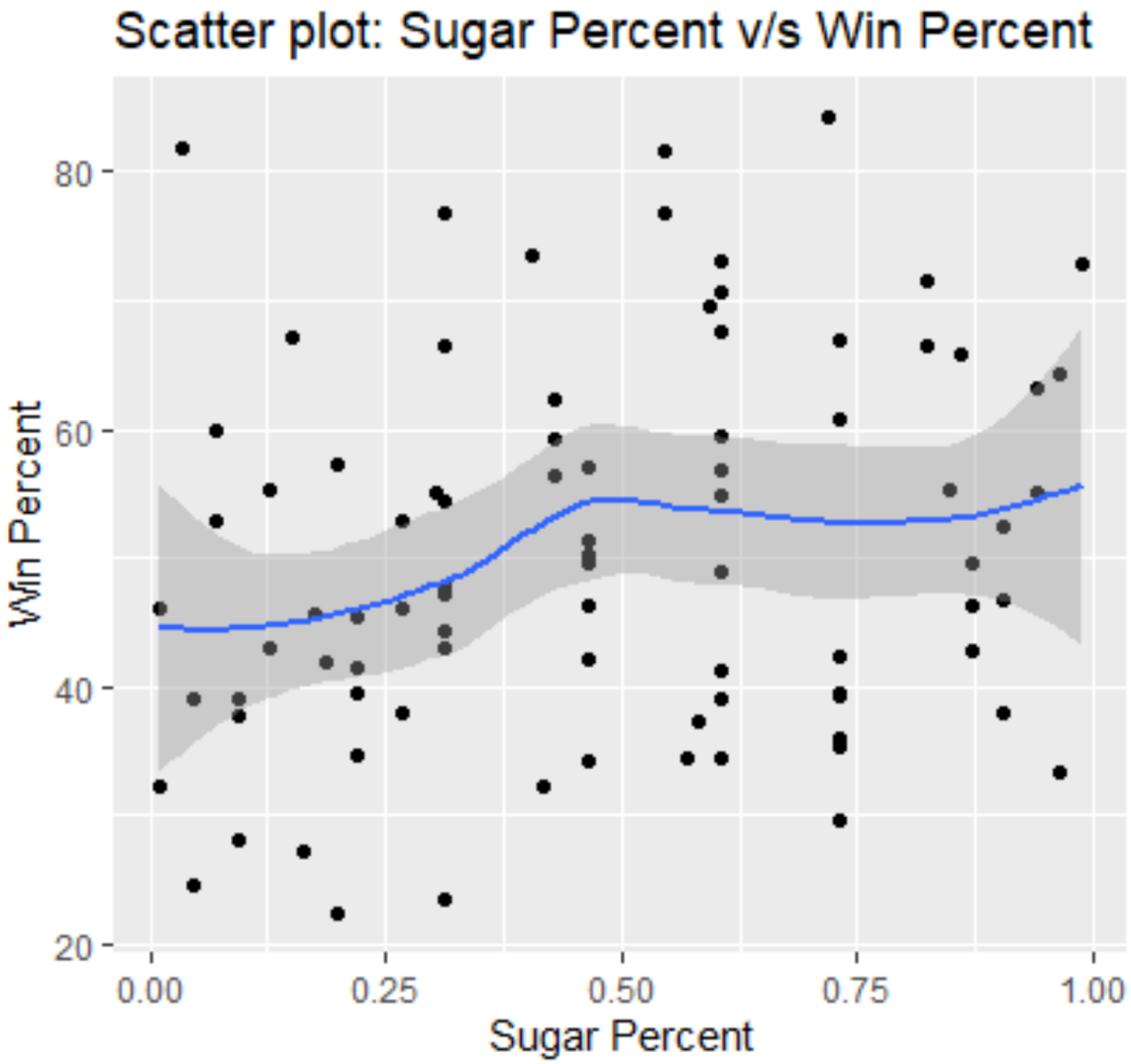
Now we look into the associations and it doesn't give a clear picture of any relationship sugar percent or price percent have with the win percent. Like we can't say clearly if there is a pattern between them. We plot the graph and see if we can also get a smooth line that can be called as the best fit line but unfortunately in this case, it is clear that there is none. That means, the share of sugar content of a chocolate or the price tag has no direct relationship to why a chocolate has a higher win percent and thus the ranking is based on some other parameter or a combination of parameters.

R Code:

```
candy_rankings %>% ggplot(aes(x=sugarpercent, y=winpercent)) + geom_point() +
  xlab("Sugar Percent") + ylab ("Win Percent") + ggtitle ("Scatter plot: Suga
r Percent v/s Win Percent") + geom_smooth()
```




```
candy_rankings %>% ggplot(aes(x=pricepercent, y=winpercent)) + geom_point() +
  xlab("Price Percent") + ylab ("Win Percent") + ggtitle ("Scatter plot:
Price Percent v/s Win Percent") + geom_smooth()
```



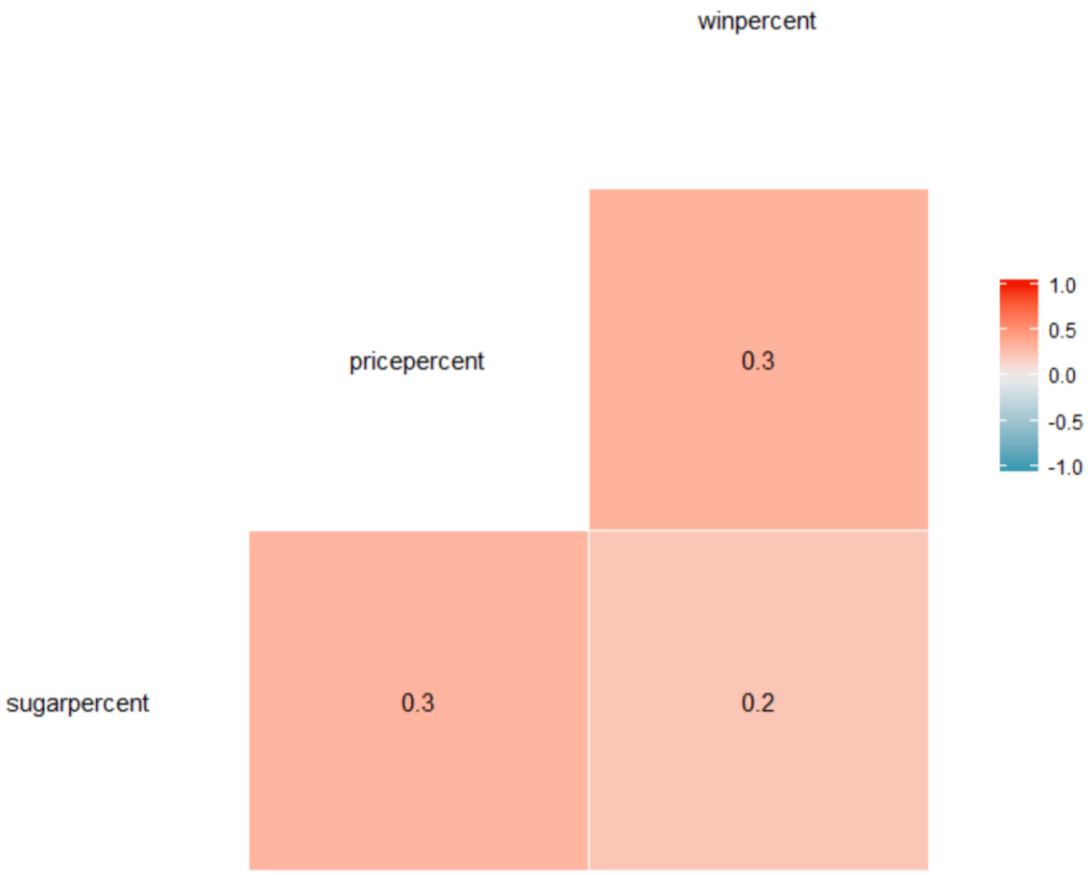
We know that they are not co-related, but how close or further away they are to have any relationship? We can check this using GGally package and using the function ggcorr() to calculate the degree of co-relationship.

R code:

```
library(GGally)

candy_corr <- candy_rankings %>% select(competitorname, sugarpercent,
pricepercent, winpercent)
```

```
ggcorr(candy_corr, label = TRUE)
```



Correlation coefficients are always between -1 and 1, inclusive. A correlation coefficient of -1 indicates a perfect, negative fit in which y-values decrease at the same rate as x-values increase. A correlation coefficient of 1 indicates a perfect, positive fit in which y-values increase at the same rate that x-values increase. In most cases, such as our example above, the correlation coefficient will be somewhere between -1 and 1. However, in this case we can see the numeric variables are all between 0.5 to -0.5 which essentially means weak relationship.

3. Consider all the logical-type variables in the dataset. For each logical variable, find the average difference in winpercent between the treats that satisfy the condition and the treats that don't satisfy it. Which logical variable seems to have the strongest effect on winpercent?

Explanation: We convert the variables into numeric type from factors and then use it to identify the True for 1 and False for 0. We can avoid this step, but it is easy for calculation and applying for loop.

For next step, I identify the variables for which I need to calculate the average difference in winpercent. We write the dataframe for one such case and then pass it inside a function. We call the function for each of those identified variables earlier and pass it inside a 'for loop' to return the respective value for the candy names and get the function to return only the absolute difference in win percent between the treats that satisfy the condition and the treats that don't satisfy it.

R code and output:

```
candy_convert <- candy_rankings
candy_convert[2:10] <- lapply(candy_convert[2:10],as.numeric)
str(candy_convert)

new <- candy_rankings[2:10] %>% colnames()

diff <- function(arg) {
  candy_select <- candy_convert %>% select(arg, winpercent) %>%
    group_by_(arg)%>% summarize(avg = mean(winpercent)) %>%
    mutate(Diff = avg - lag(avg, default = avg[1]))
  candy_select[2,"Diff"]
  return(abs(candy_select[2,"Diff"]))
}

for (i in new) {
  print(paste(i,diff(i)))
}
```

```
[1] "chocolate 18.7792724054054"
[1] "fruity 11.2073809193729"
[1] "caramel 8.41636946076459"
[1] "peanutyalmondy 16.0187563772636"
[1] "nougat 10.6087798388278"
[1] "crispedricewafer 17.2762371282051"
[1] "hard 11.9094495857143"
[1] "bar 14.5810179471726"
[1] "pluribus 7.24362327937916"
```

From the output above, we can confirm that 'Chocolate' is the logical variable with the strongest relationship with win percent.

College admissions dataset

Read in the following dataset:

http://vicpena.github.io/admin.csv

It contains information about college admissions by gender and department at some college.

```
cad <- read.csv("http://vicpena.github.io/admin.csv")

str(cad)
head(cad)
class(cad)
```

Firstly, we read the file and store the data into a dataframe – 'cad'. We call the str and head function to see the variables and their type and also the top 5 data to know what to expect when we start analyzing the dataframe.

1. Find the percentage of men who applied and got in and the percentage of women who applied and got in. What do you see?

Explanation: We have to consider the frequency variable to compute any proportion, however we have to be careful while doing it for different arguments within a group. It is like doing a pivot table on the data and then using the % for column. Just like that, we need to do it separately for each of these variables.

Having said that, I will first group the dataset by both admit and gender columns and then use the summarize function to get the sum of the 'frequency' by the grouped elements.

But, we need the answer by gender, so we group the data again by using the previous study and do a pipeline on it and group it by gender and then add one more column by using the mutate function and divide the previous result - grouped by both 'admit and gender', and then later the one just by 'gender'. Essentially, this is creating a pivot table in R.

Observation: I see that the share of men who got an admit is visibly higher than the women. While only 30.35% women got a successful admit the respective number for men is 44.52% which is approximately 46% higher to their female counterpart. We cannot clearly say without knowing other details involved in deciding an 'admit' whether there is any gender discrimination or whether it is based on merit, and thus it will be immature to conclude it so easily with just this much data but there is a stark difference in the share of admits by 'gender', which me must recognize and should further investigate.

R code and output table:

```
cad1 <- cad %>% group_by(Admit,Gender) %>% summarize(Total_by_g_a = sum(Freq)
) %>%
  group_by(Gender) %>% mutate(Total_by_g = sum(Total_by_g_a)) %>%
  group_by(Gender) %>% mutate(Percentage = 100*(Total_by_g_a/Total_by_g)) %>%
  arrange(desc(Gender)) %>% View()
```

Admit	Gender	Total_by_g_a	Total_by_g	Percentage
Admitted	Male	1198	2691	44.51877
Rejected	Male	1493	2691	55.48123
Admitted	Female	557	1835	30.35422
Rejected	Female	1278	1835	69.64578

2. Now, find the percentage of men who applied and got in by department. Do the same with women. Compare the results with what you found in part 1.

Explanation: We repeat the same procedure as the last one. We only need to group by one extra parameter this time, that is, 'dept'. Rest follows as is.

Comparison of the result: When we repeat the exercise by including 'department' we don't see much of a difference. By proportion, for each 'dept' there are more share of female admits than men. For departments C and E – the men have more share of admits while for all other departments it is just the opposite. If we compare the results from previous question, we will now have more clarity into the data and we can observe that the total admits by gender is skewed towards men because of the higher volume and also more admission requests per department. But when we start classifying the data (like in this case by 'dept') the difference starts to fade away.

R code:

```
cad2 <- cad %>% group_by(Admit,Dept,Gender) %>% summarize(Total_g_a_d =
sum(Freq)) %>%
  group_by(Dept,Gender) %>% mutate(Total_g_d = sum(Total_g_a_d)) %>%
  group_by(Gender) %>% mutate(Percentage = 100*(Total_g_a_d/Total_g_d)) %>%
  arrange(desc(Gender)) %>% filter(Admit=="Admitted") %>% View()
```

Admit	Dept	Gender	Total_g_a_d	Total_g_d	Percentage
Admitted	A	Male	512	825	62.060606
Admitted	B	Male	353	560	63.035714
Admitted	C	Male	120	325	36.923077
Admitted	D	Male	138	417	33.093525

Admitted	E	Male	53	191	27.748691
Admitted	F	Male	22	373	5.898123
Admitted	A	Female	89	108	82.407407
Admitted	B	Female	17	25	68
Admitted	C	Female	202	593	34.064081
Admitted	D	Female	131	375	34.933333
Admitted	E	Female	94	393	23.918575
Admitted	F	Female	24	341	7.038123

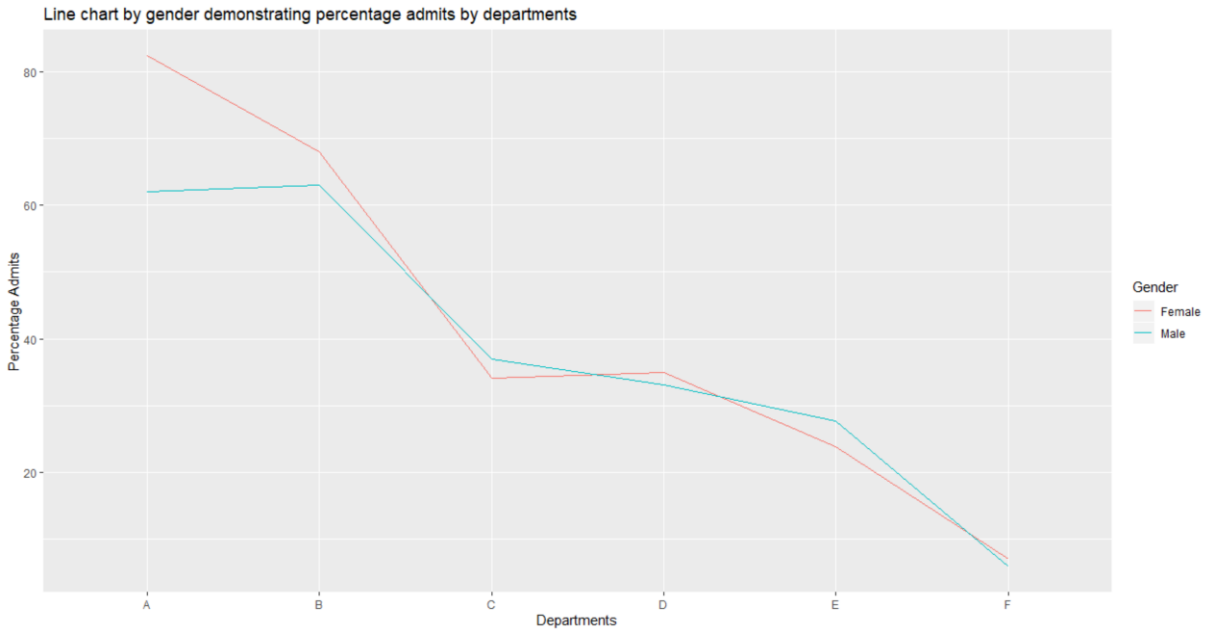
3. Explain what is going on in this dataset. Do you see any evidence of gender discrimination?

Explanation: We compared two cases already in the previous two questions and we see evidence of lesser discrimination. It is the total applications that gives a sense of some discrimination but when we look deeper into the data by splitting it by department and the admits, we see the discrimination is not there. However, there is a known gap in terms of total admits and hence lesser acceptance when it comes to females.

We can see the visualization of the admits by % share for both males and females by Dept below and only Dept 'A' shows a glaring difference in proportion and unlike expectations the difference is in favour of female gender.

R Code:

```
cad2 %>% ggplot(aes(x=cad2$Dept, y=cad2$Percentage, group = Gender, colour = Gender)) +
  geom_line() + xlab("Departments") + ylab("Percentage Admits") +
ggtitle("Line chart by gender demonstrating percentage admits by departments")
```


Line chart by gender demonstrating percentage admits by departments
Percentage Admits
80
60
40
20
A
B
C
D
E
F
Departments
Gender
Female
Male

Fandango movie ratings

Read the article https://fivethirtyeight.com/features/fandango-movies-ratings/. The dataset is accessible in library(fivethirtyeight). You can read it in with the command data(fandango). The variable names are reasonably self-descriptive, but you can get a more detailed description by typing in ?fandango.

```
data(fandango)
str(fandango)
head(fandango)
?fandango
```

1. Identify the Top 5 best rated and Top 5 worst rated movies in the dataset. Average over different platforms.

Explanation: We decide on the platforms on which we want to take an average on and then just calculate the average by rows for each film.

We start with reading the dataset and then use 'rowwise' function to calculate for each film the average rating by 5 chosen platforms. We create a new column with this sum and then call the mutate function again to get the average of all these ratings.

Once completed, we just need to arrange our data in a chosen order and then select the variables we want to display, that is, films and the newly computed average rating.

NOTE: We take the normalized scores for the platforms as available from the dataset. For fandango, we ignore it because it doesn't have a normalized score.

R code:

```
fandango_rank <- fandango %>% rowwise() %>%
  mutate(all_platforms_rating_sum = sum(rt_norm,rt_user_norm,metacritic_norm,
metacritic_user_nom,imdb_norm)) %>%
  mutate(new_avg_rating = all_platforms_rating_sum/5) %>% select(film,new_avg
_rating) %>% arrange(desc(new_avg_rating))

head(fandango_rank, n=5)
```

```
Source: local data frame [5 x 2]
Groups: <by row>
# A tibble: 5 x 2
  film              new_avg_rating
  <chr>                      <dbl>
1 Inside Out                  4.57
2 About Elly                  4.48
3 Mad Max: Fury Road          4.44
4 Song of the Sea             4.41
5 Amy                         4.41
```

```
tail(fandango_rank, n=5)
```

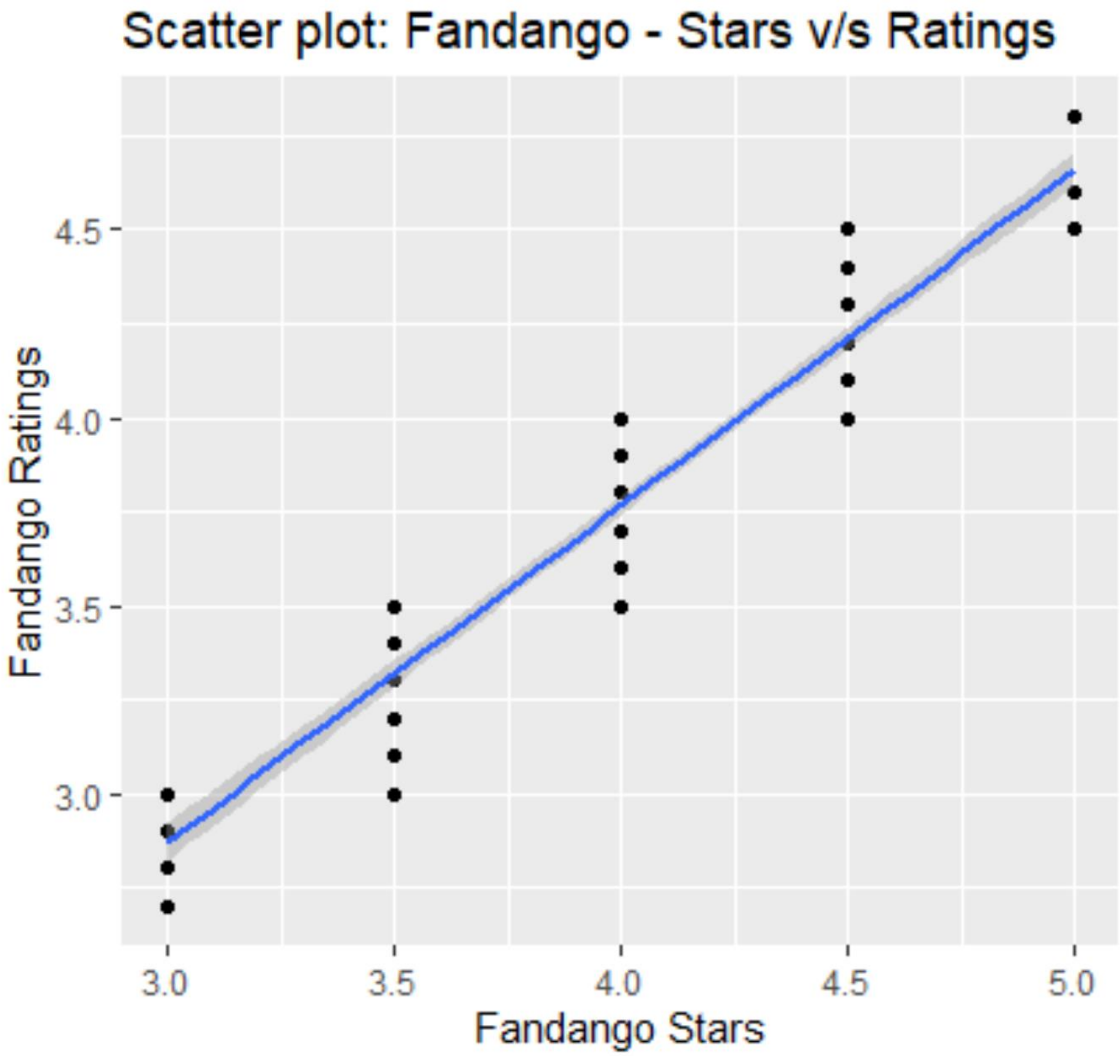
```
Source: local data frame [5 x 2]
Groups: <by row>
# A tibble: 5 x 2
  film                    new_avg_rating
  <chr>                            <dbl>
1 The Loft                          1.62
2 Hot Tub Time Machine 2            1.56
3 Mortdecai                         1.56
4 Fantastic Four                    1.21
5 Paul Blart: Mall Cop 2            1.21
```

2. Visualize the difference between Fandango stars and actual Fandango ratings. Comment on what you see.

Explanation: We plot a scatter plot using geom_point function available within ggplot2 package. What we see is a strong relationship between fandango stars and fandango ratings. For each star value the rating value seems to follow an interval within 0.5 of that range. In statistical terms we can say that the graph follows a regression model wherein each y value w.r.t the x value follows a normal distribution. Therefore, there is a very strong relationship between fandango ratings and fandango stars. The same is also validate using the best fit line which almost bisects through the centre of each y – value what we call 'chimneys' in regression analysis.

R code:

```
fandango %>% ggplot(aes(x=fandango_stars, y=fandango_ratingvalue)) +
geom_point() +
  xlab("Fandango Stars") + ylab ("Fandango Ratings") + ggtitle ("Scatter
plot: Fandango - Stars v/s Ratings") + geom_smooth(method = "lm")
```



3. Some movies are loved by the critics, but hated by the audience (and sometimes, it's the other way around). Given the data you have, create a metric to measure discrepancies between user and critic ratings. Create a table that contains the Top 5 movies that seem to appeal to critics but not the audience, and another table with the Top 5 movies that users seem to like more than critics do.

Explanation: To compute this we need to find a statistic that can answer the difference in ratings between critics and users.

Before that, we need to identify the ratings which are respectively associated to critics and users. I identified the rt_norm and Metacritic_norm as the critic ones and the rt_user_norm and Metacritic_user_norm as the user ones.

Then just like the part 1 of this exercise I calculate the overall score for critics and users separately and then get the respective average using mutate function. In the code it is denoted by `new_avg_critic_rating` and `new_avg_users_rating.` As part of the final computation in the series of calculations – we subtract these two values to get the net difference in ratings which we will use to evaluate the top 5 and bottom 5 movies as rated by critics and users and this is denoted in the dataframe by the variable, `critic_user_diff.`

R code:

```
fandango_rank_comp <- fandango %>% rowwise() %>%
  mutate(all_critics_rating_sum = sum(rt_norm,metacritic_norm)) %>%
  mutate(new_avg_critic_rating = all_critics_rating_sum/2) %>%
  mutate(all_users_rating_sum = sum(rt_user_norm,metacritic_user_nom)) %>%
  mutate(new_avg_users_rating = all_users_rating_sum/2) %>%
  mutate(critic_user_diff = new_avg_critic_rating - new_avg_users_rating) %>%
  select(film,critic_user_diff) %>% arrange(desc(critic_user_diff))

head(fandango_rank_comp, n=5)
```

```
Source: local data frame [5 x 2]
Groups: <by row>

# A tibble: 5 x 2
  film              critic_user_diff
  <chr>                        <dbl>
1 Mr. Turner                   1.75
2 Timbuktu                     1.08
3 Leviathan                    1
4 While We're Young            1
5 It Follows                   0.975
```

```
tail(fandango_rank_comp, n=5)
```

```
Source: local data frame [5 x 2]
Groups: <by row>

# A tibble: 5 x 2
  film            critic_user_diff
  <chr>                      <dbl>
1 Pixels                     -1.57
2 Black or White             -1.57
3 Self/less                  -2.02
4 Little Boy                 -2.25
5 Do You Believe?            -2.28
```

Lahman Baseball Dataset

Download the data from:

http://vicpena.github.io/sta9750/Teams1719.csv

Answer the following questions.

Some questions about home advantage

1. Create a statistic that quantifies "home advantage". You'll use this statistic for the next few questions. There is more than one reasonable choice here. Propose 2 different statistics and justify why you picked the one you'll use from now on.

There are multiple ways to quantify 'home advantage'. The most common would be to evaluate the ratio of home wins by away wins or (home wins – away wins) divided by total games.

While, it most likely won't change the answers for the top or the bottom results but there is an imminent catch in this. The one that will prove the above ways statistically wrong. Before we go for any fraction or percentage, one most check if the base is same and that's what led me to consider a different formula.

So, I did a sanity check on the dataset to verify if there is any anomality in the dataset. Voila! I found a few. When you add all the total games you will see that there are cases when odd number of games were played. In theory, a team is always expected to play same number of home or away games and thus the sum of all games needs to be an even number.

That being said, when I look into the odd ones, I find that there are cases when some teams have played less home games than away games or vice versa. Now, the problem of having a simplistic ratio of home or away games by total games is not correct because for cases when the total home wins are same but total games are not, one team will get a biased advantage due to the incorrect statistical computation. To illustrate the same, see the table below –

Team	Year	League	HomeW	HomeL	HomeTotal	AwayW	AwayL	AwayTotal	Total
White Sox	2019	AL	39	41	80	33	48	81	161
Tigers	2019	AL	22	59	81	25	55	80	161
Dodgers	2018	NL	45	37	82	47	34	81	163
Rockies	2018	NL	47	34	81	44	38	82	163
Marlins	2018	NL	38	43	81	25	55	80	161
Brewers	2018	NL	51	30	81	45	37	82	163
Cubs	2018	NL	51	31	82	44	37	81	163
Pirates	2018	NL	44	36	80	38	43	81	161

This led me to evaluate other possibilities and I came up with the following statistical formula wherein we take total home wins and divide it by total home games and this is whole divided by away wins by total away games. Mathematically, it can be represented as:

statistic = (HomeW/(HomeW + HomeL)) / (AwayW/(AwayW + AwayL)))

or,

$$\text{Statistic} = \frac{\frac{\text{Home Wins}}{\text{Total Home Games}}}{\frac{\text{Away Wins}}{\text{Total Away Games}}}$$

2. Find home advantage statistics for the American League (AL) and National League (NL) in the 2017-2019 period. Comment on the results. Do you see any differences between leagues? Do you see any evidence of home advantage at all? What are the years where there seems to be more of a home advantage, and those where the effect might not be as strong (or doesn't seem to be there)?

Explanation: I use the above statistic to calculate the 'home advantage' and then add it to the existing table using the mutate function.

We then calculate the ratio by grouping the dataframe by league and year as mentioned in the question and then summarize it on basis of the statistic we just created earlier.

R Code:

```
lbd_teams <- read.csv("http://vicpena.github.io/sta9750/Teams1719.csv")

str(lbd_teams)
head(lbd_teams)


lbd_teams <- lbd_teams %>% mutate(statistic_home_adv = (HomeW/(HomeW + HomeL)
) / (AwayW/(AwayW + AwayL)))


lbd_teams %>% group_by(League,Year) %>% summarize(home_adv_ratio = mean(stati
stic_home_adv))
```

```
# A tibble: 6 x 3
# Groups:   League [2]

  League  Year home_adv_ratio
  <fct>  <int>          <dbl>
1 AL      2017           1.18
2 AL      2018           1.18
3 AL      2019           1.07
4 NL      2017           1.23
5 NL      2018           1.12
6 NL      2019           1.19
```

For all 3 years from 2017-2019 period, we see the ratio to be greater than 1. Which means there is a home advantage surely and this is true for both the leagues, namely – AL and NL.

The highest percentage of home advantage was observed for the league NL in the year 2017 whereas the least was observed for AL league in the year 2019. If we just take an average by grouping the results at league level only, NL league shows more tendency to have teams with home advantage than teams in AL league.

3. Find the teams that had the highest and lowest home advantage effect by league in 2017, 2018, and 2019 separately. Comment on the results.

Explanation: We continue to use the same statistic as in que 1 above. Here we group by an additional variable, that is, 'team' to know which team had the maximum and minimum home advantage for each year and not all years combined.

However, we want the result per year and for each group and to filter it we need to partition the new table with the home_adv_ratio we calculated and see the highest and lowest for both years and leagues.

R Code:

```
lbd_top <- lbd_teams %>% group_by(League,Team,Year) %>%
  summarize(home_adv_ratio = mean(statistic_home_adv)) %>%
  arrange(desc(home_adv_ratio,Year))
```

We can use the above dataframe to answer the query but it would be neat to just have the records we want and remove the other ones and hence I decided to save it into a new

data frame – 'lbd_top' and use that table to query my request by portioning it on various groups. We need to call the library 'sqldf' for performing that task. RN is nothing but the rank based on rownumbers to identify the top and the bottom ones for each variable. The same exercise can be carried out for separate years in dplyr or through a matrix, but I was looking for an optimized solution something that can cover even the cases for 'ties'.

```
lbd_top
library(sqldf)
```

Highest for each league and year:

```
sqldf("select * from ( select *, dense_rank() over
(partition by Year,League order by home_adv_ratio desc)rn from lbd_top) where
rn=1")
```

	League	Team	Year	home_adv_ratio	rn
1	AL	Athletics	2017	1.586207	1
2	AL	Orioles	2017	1.586207	1
3	NL	Padres	2017	1.535714	1
4	AL	Twins	2018	1.689655	1
5	NL	Phillies	2018	1.580645	1
6	AL	Rangers	2019	1.363636	1
7	NL	Cubs	2019	1.545455	1

NOTE: *The reason I used Dense_Rank() function and not simply row number () is to ensure that we don't miss on cases where there is a 'tie'. As you can see here, there is a tie for the best team in the year 2017 as Athletics and Orioles both score the same* ***average*** *'home_adv_ratio'.*

Lowest for each league and year:

```
sqldf("select * from ( select *, dense_rank() over
(partition by Year, League order by home_adv_ratio asc)rn from lbd_top) where
rn=1")
```

	League	Team	Year	home_adv_ratio	rn
1	AL	Astros	2017	0.9056604	1
2	NL	Nationals	2017	0.9400000	1
3	AL	Astros	2018	0.8070175	1
4	NL	Padres	2018	0.8857143	1
5	AL	Red Sox	2019	0.8260870	1
6	NL	Giants	2019	0.8333333	1

4. Which franchise had the highest average home advantage in the 2017-2019 period? Which one had the lowest average home advantage effect?

Explanation: We continue to use the same statistic as in que 1 above. Here we group by only one variable that is Team and then use summarize function to compute the average ratio of the home advantage for each team.

Then we look at the top-most and the bottom-most record to get our answer for the highest and lowest home advantage by 'teams' in the period of 2017-2019. For this we use top function and ask the function to return the top result by passing an argument 1 and bottom most by passing an argument -1.

R Code:

```
lbd_teams %>% group_by(Team) %>% summarize(home_adv_ratio = mean(statistic_ho
me_adv)) %>% arrange(desc(home_adv_ratio)) %>% top_n(1)
```

```
Selecting by home_adv_ratio
# A tibble: 1 x 2
  Team     home_adv_ratio
  <fct>             <dbl>
1 Phillies           1.43
```

```
lbd_teams %>% group_by(Team) %>% summarize(home_adv_ratio = mean(statistic_ho
me_adv)) %>% arrange(desc(home_adv_ratio)) %>% top_n(-1)
```

```
Selecting by home_adv_ratio
# A tibble: 1 x 2
  Team   home_adv_ratio
  <fct>           <dbl>
1 Astros          0.996
```

5. After completing these exercises, what did you learn about home advantage effect in the MLB? You're welcome to try out a few new queries to illustrate your points.

Explanation: We have already looked into the dataset through different lens and have observed that there is a home advantage for each team but to quantify it lets look at the count of such cases. There are 15 teams competing in MLB for 2 leagues. So that makes 30 records and for 3 years from 2017 to 2019, we will have 30*3, that is a total of 90 observations. So, how many of these observations saw cases of a home advantage? Let's compute:

R Code:

```
lbd_teams %>% group_by(League, Year, statistic_home_adv > 1 ) %>%
summarize(total_home_adv = n())
```

```
# A tibble: 12 x 4
# Groups:   League, Year [6]
```

```
   League  Year `statistic_home_adv > 1` total_home_adv
   <fct>  <int> <lgl>                              <int>
 1 AL      2017 FALSE                                3
 2 AL      2017 TRUE                                12
 3 AL      2018 FALSE                                2
 4 AL      2018 TRUE                                13
 5 AL      2019 FALSE                                5
 6 AL      2019 TRUE                                10
 7 NL      2017 FALSE                                1
 8 NL      2017 TRUE                                14
 9 NL      2018 FALSE                                7
10 NL      2018 TRUE                                 8
11 NL      2019 FALSE                                1
12 NL      2019 TRUE                                14
```

We can observe that for each year by League the cases of teams with home advantage far outweighs the ones with away advantage. The TRUE here means that the teams had home advantage while FALSE means no such advantage.

```
lbd_teams %>% group_by( Year, statistic_home_adv > 1 ) %>%
summarize(total_home_adv = n())
```

```
# A tibble: 6 x 3
# Groups:   Year [3]
```

```
   Year `statistic_home_adv > 1` total_home_adv
  <int> <lgl>                              <int>
1  2017 FALSE                                  4
2  2017 TRUE                                  26
3  2018 FALSE                                  9
4  2018 TRUE                                  21
5  2019 FALSE                                  6
6  2019 TRUE                                  24
```

By year also, we see the same trend. For 2018 though we had the maximum of 9 cases in the period of 2017-2019 when teams had no home advantage.

Having said that, at an overall level there were just 19 such observations from a sample space of 90 (which is equal to 21.11%) where there was no home advantage observed.

```
lbd_teams %>% group_by(statistic_home_adv > 1 ) %>% summarize(total_home_adv
= n())
```

```
# A tibble: 2 x 2
```

```
  `statistic_home_adv > 1` total_home_adv
  <lgl>                              <int>
1 FALSE                                 19
2 TRUE                                  71
```

In this exercise, you'll work with the Lahman Baseball datasets, which you can access after installing library(Lahman). After installing the package, you can type in ?Lahman to get some information on the structure of the datasets and see what's available. If you want to do a class project with baseball data, you're welcome to use this resource.

```
install.packages("Lahman")

library(Lahman)
lahman_pitching <- Lahman::Pitching
lahman_people <- Lahman::People

colnames(lahman_people)
colnames(lahman_pitching)
```

Aging in pitchers and batters

1. Let's consider data from 2018 only and look at the subset of pitchers who pitched more than 250 outs. Plot the earned run average (ERA; small values are good and big ones are bad) of the pitchers against their age. Do you see any patterns? Now, find a table with the average ERAs by age. Do you see any patterns?

Explanation: We install the lahman package which contains all the datasets and then load the tables 'pitching' and 'people', that we will need to solve few of the questions.

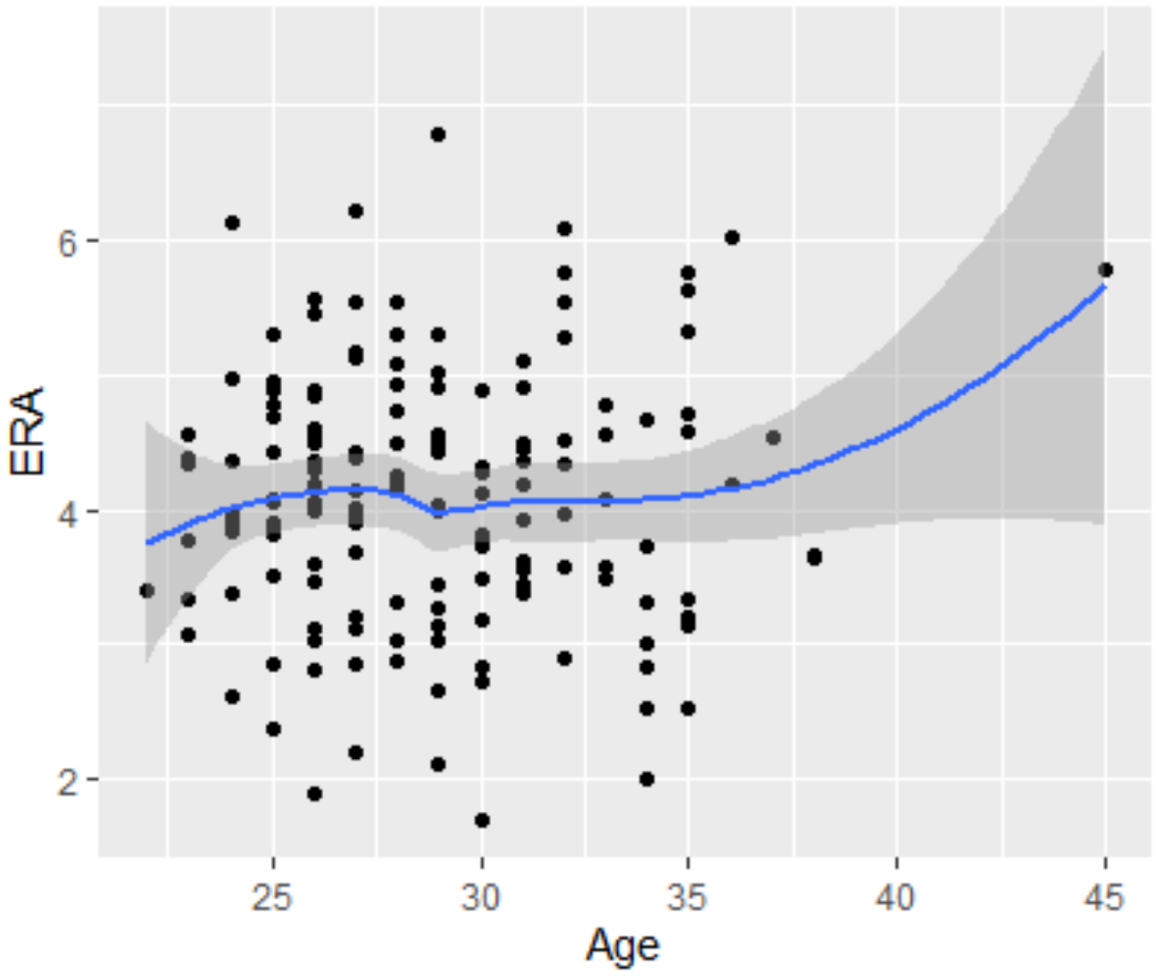
I look into column names for each dataset to know what variables we really need to solve the query. To calculate age, I need to get Year ID from pitching table and Birthyear from the people table. So, we join these tables and then subtract these two columns to get the desired output. I user inner join to ensure we have records for which both the yearID and Birthyear is available.

Then we filter on the dataset for IPouts > 250 and year as 2018. Lastly, we use this summarized dataframe to plot our graph between age and ERA.

R code and the graph:

```
lahman_age <- lahman_pitching %>% inner_join(lahman_people, by = "playerID")
%>% filter(yearID == 2018 & IPouts> 250) %>% mutate(Age = yearID - birthYear)

lahman_age %>% ggplot(aes(x=Age, y=ERA)) + geom_point() + geom_smooth()
```



There is no direct relationship between these two variables, Age and ERA. However, we do see that best of the ERAs are true at a young age and it falls as the players grow old. The average ERA is concentrated in the range of 3 to 5. Also, there is one big outlier at the age of 45 for a player whose ERA is on the higher end and close to 6.

For the second part of the question, I calculate average ERA by age using the summarize function on ERA and group it by Age. I use View() whenever the total records are more than what R console can accommodate. I then simply share the records in a table just as R would show them.

R code and the output:

```
lahman_age %>% group_by(Age) %>% summarize(Avg_ERA = mean(ERA)) %>% View()
```

Age	Avg_ERA
22	3.41
23	3.91
24	4.117778
25	4.104286
26	4.067778
27	4.054211
28	4.305714
29	4.076667
30	3.606
31	4.135
32	4.664444
33	4.102
34	3.152857
35	4.247778
36	5.1
37	4.53
38	3.655
45	5.78

As a pitcher ages, his ERA tends to see decline, but there have been exceptions as it is not a linear relationship. In fact, the best mean ERA was observed at the age of 34 which was preceded by high ERA and also is followed by a steep increase in ERA. That being said, we can't conclusively comment on a definite pattern between age and ERA.

2. Again, let's look at pitchers who pitched more than 250 outs in 2018. Identify the top 5 best and worst pitchers, in terms of ERA.

Explanation: We use the dataset for lahman_pitching and then filter on IPOuts > 250 and year as 2018 and assign it to a dataframe.

We use the new dataframe to now get the top 5 and bottom 5 pitchers using the head and the tail function. We need to be careful of the fact that low ERA is considered good whereas big ones are considered bad.

R Code and output:

```
lahman_age <- lahman_pitching %>% filter(yearID==2018 & IPouts > 250)

lahman_age %>% select(playerID, ERA, nameFirst, nameLast) %>% arrange(ERA)
%>% head(n=5)
```

```
   playerID  ERA nameFirst nameLast
1 degroja01 1.70     Jacob   deGrom
2 snellbl01 1.89     Blake    Snell
3 buchhcl01 2.01      Clay Buchholz
4  salech01 2.11     Chris     Sale
5 bauertr01 2.21    Trevor    Bauer
```

```
lahman_age %>% select(playerID, ERA, nameFirst, nameLast) %>% arrange(ERA)
%>% tail(n=5)
```

```
     playerID  ERA nameFirst nameLast
155 hammeja01 6.02     Jason   Hammel
156 baileho02 6.09     Homer   Bailey
157 giolilu01 6.13     Lucas  Giolito
158 perezma02 6.22    Martin    Perez
159 moorema02 6.79      Matt    Moore
```

3. Consider the best pitcher (in terms of ERA) that you found in part 2. Find his ERA by season throughout his career. Based on this alone, do you think he's already "peaked"? If you like baseball, you're welcome to share your opinion here as well.

Explanation: We found from the previous que that the best pitcher in terms of ERA goes by the playerID = degroja01 and we use this ID to calculate the player's performance across seasons.

Once we filter it for this player and group his career stats by ERA on **lahman::pitching** table, we find that he indeed peaked in 2018.

R Code and output:

```
lahman_pitching %>% filter(playerID=="degroja01") %>% select(playerID, yearID
, ERA) %>% arrange(desc(yearID))
```

```
   playerID yearID  ERA
1 degroja01   2018 1.70
2 degroja01   2017 3.53
3 degroja01   2016 3.04
4 degroja01   2015 2.54
5 degroja01   2014 2.69
```

I don't understand baseball really well, but I tried my best to make more sense of how this sport is played and what are the best performances in the game's history. From online sources, it seems that the performance by Jacob deGrom in 2018 in terms of recorded ERA is one of the best in the game's history. Though in the past with different rules and format there were many lower ERAs recorded (in 1800s and 1900s) but since 2000, this is the best ERA for anyone except for Zack Greinke in 2015 when he recorded an ERA of 1.66

The overall record is held by Tim Keefe with an astonishing ERA of just 0.86 in the year 1880.

4. Let's do a similar exercise, but now with batting average (BA; more is better). Use the battingStats function in Lahman to find BAs. Consider data from 2018 only and look at players that have more than 200 at bats (AB). Plot BA against age. Do you see any patterns? Find a table with average BAs by age. Explain what you see.

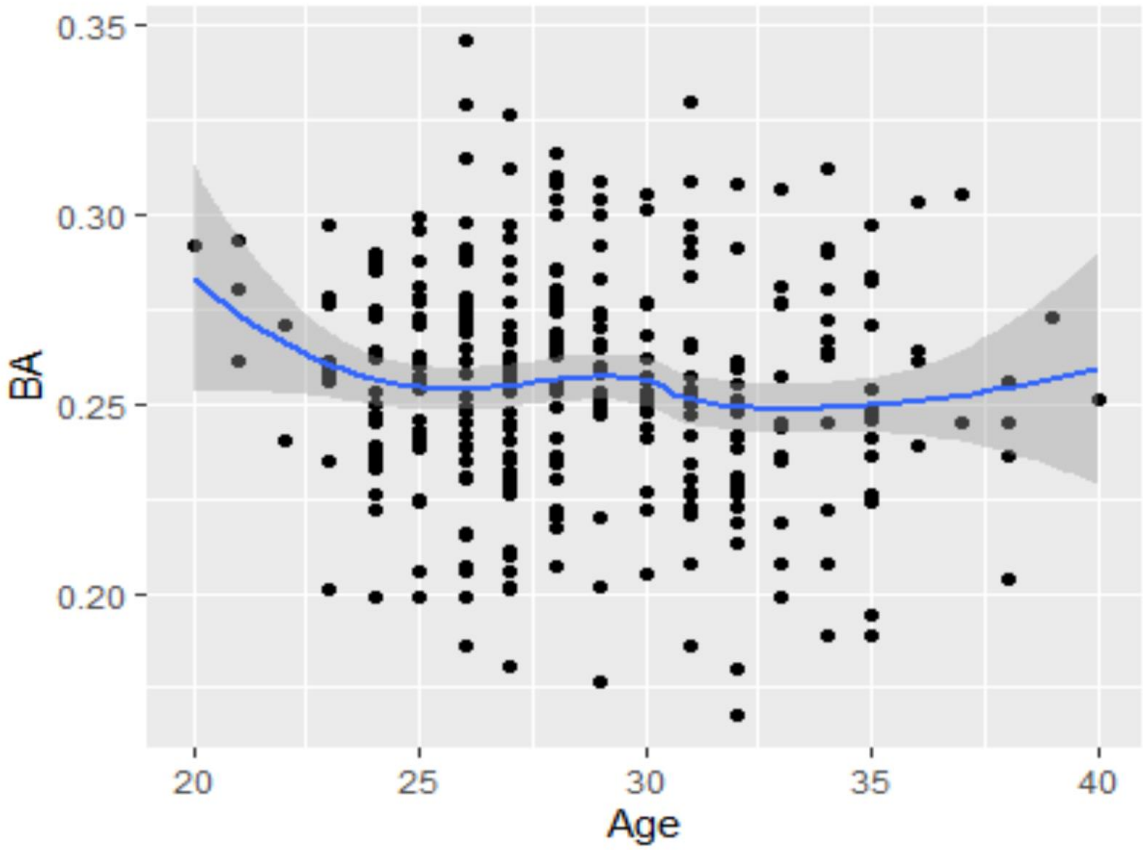
Explanation: We follow the same process as que 1 of this section. Only few things change, one we replace pitching table with batting stats function and join it with the people table to compute 'Age'. Later we plot a graph against BA instead of ERA.

R code and the graph:

```
lahman_batting_stats <- Lahman::battingStats()
colnames(lahman_batting_stats)

lahman_batting_stats <- lahman_batting_stats %>% inner_join(lahman_people, by
= "playerID") %>%  filter(yearID==2018 & AB > 200) %>% mutate(Age = yearID -
birthYear)

lahman_batting_stats %>% ggplot(aes(x=Age, y=BA)) + geom_point() +
geom_smooth()
```



There is no direct relationship between these two variables, Age and BA. As we can see that the smooth line is showing deviations. However, we do see that best of the BAs are recorded at an age where the players attain some experience. High BA scores are observed for players above the age of 25 and till they reach mid-30s. Post that we start seeing a decline in BA scores which kind of reflects the effect on age on players.

For the second part of the question, I calculate average BA by age using the summarize function on ERA and group it by Age.

R code and the output:

```
lahman_batting_stats %>% group_by(Age) %>% summarize(Avg_BA = mean(BA)) %>%
View()
```

Age	Avg_BA
20	0.292
21	0.278
22	0.2555
23	0.2598
24	0.2539048

25	0.25492
26	0.257275
27	0.2492326
28	0.2604
29	0.2591154
30	0.2566111
31	0.2554815
32	0.2367727
33	0.2463846
34	0.2585833
35	0.2482
36	0.26675
37	0.275
38	0.23525
39	0.273
40	0.251

5. Again, let's look at players with more than 200 ABs in 2018. Find the top 5 best and worst players in terms of BA.

Explanation: We use the dataset for lahman_batting_stats and then filter on AB > 200 and year as 2018 and assign it to the same dataframe.

We use this new dataframe to now get the top 5 and bottom 5 players using the head and the tail function by BA variable.

R Code:

```
lahman_batting_stats %>% select(playerID, BA, nameFirst, nameLast) %>% arrang
e(desc(BA)) %>% head(n=5)
```

```
   playerID    BA nameFirst nameLast
1 bettsmo01 0.346    Mookie    Betts
2 martijd02 0.330     J. D. Martinez
3 mcneije01 0.329      Jeff   McNeil
4 yelicch01 0.326 Christian   Yelich
5 altuvjo01 0.316      Jose   Altuve
```

```
lahman_batting_stats %>% select(playerID, BA, nameFirst, nameLast) %>% arrang
e(desc(BA)) %>% tail(n=5)
```

```
     playerID    BA nameFirst nameLast
321 sanchga02 0.186      Gary  Sanchez
322 altheaa01 0.181     Aaron  Altherr
323 fowlede01 0.180    Dexter   Fowler
324  leonsa01 0.177     Sandy     Leon
325 davisch02 0.168     Chris    Davis
```

6. Consider the best player (in terms of BA) that you found in part 5. Find his BA by season throughout his career. Based on this alone, do you think he's already "peaked"? If you like baseball, you're welcome to share your opinion here as well.

Explanation: We found from the previous que that the best player in terms of BA goes by the playerID = bettsmo01 and we use this ID to calculate the player's performance across seasons.

Once we filter it for this player and group his career stats by BA on **Lahman::battingStats()**, we find that the player has played in 4 seasons so far until 2018. And, his performance in 2018 is his best performance and has thus peaked already.

R Code:

```
lahman_batting_stats %>% filter(playerID=="bettsmo01") %>% select(playerID, y
earID, BA) %>% arrange(desc(yearID))
```

```
   playerID yearID    BA
1 bettsmo01   2018 0.346
2 bettsmo01   2017 0.264
3 bettsmo01   2016 0.318
4 bettsmo01   2015 0.291
```

I don't understand baseball really well, but I tried my best to make more sense of how this sport is played and what are the best performances in the game's history. From online sources, it seems that the performance by Betts Mookie in 2018 in terms of recorded Batting Average (BA) is one of the best in the game's history. Though unlike what we saw in case of ERA there had been multiple such good performances since 2000.

The overall record is held by Hugh Duffy with an all-time high BA of .440 in the year 1894.