Analyst Intern, Data Science & Solutions P

William Rice

08/26/23

Introduction

The purpose of this project is to gauge your technical skills and problem solving ability by working through somethin science project. You will work your way through this R Markdown document, answering questions as you go along. F name to the "author" key in the YAML header. When you're finished with the document, come back and type your an the top. Please leave all your work below and have your answers where indicated below as well. Please note that we so make it clear, concise and avoid long printouts. Feel free to add in as many new code chunks as you'd like.

Remember that we will be grading the quality of your code and visuals alongside the correctness of your answers. Pl as much as possible (instead of base R and explicit loops.)

Note:

Throughout this document, any `season` column represents the year each season started. For example, the 20 dataset as 2015. For most of the rest of the project, we will refer to a season by just this number (e.g. 2015) in (e.g. 2015-16).

Answers

Part 1

Question 1:

- 1st Team: 25.8 points per game
- 2nd Team: 23.1 points per game
- 3rd Team: 20.5 points per game
- All-Star: 21.6 points per game

Question 2: 4.7 Years

Question 3:

- Elite: 2 players.
- All-Star: 1 players.
- Starter: 10 players.
- Rotation: 8 players.
- Roster: 14 players.
- Out of League: 38 players.

Open Ended Modeling Question: Please show your work and leave all responses below in the document.

Part 2

Question 1: 28.9%
Question 2: Written question, put answer below in the document.
Question 3: Written question, put answer below in the document.

Setup and Data

```
library(tidyverse)
# Note, you will likely have to change these paths. If your data is in the same folder as th
# the paths will likely be fixed for you by deleting ../../Data/awards_project/ from each st
awards <- read_csv("awards_data.csv")
player_data <- read_csv("player_stats.csv")
team_data <- read_csv("team_stats.csv")
rebounding_data <- read_csv("team_rebounding_data_22.csv")
```

Part 1 – Awards

In this section, you're going to work with data relating to player awards and statistics. You'll start with some data mai
work towards building a model to predict broad levels of career success.

Question 1

QUESTION: What is the average number of points per game for players in the 2007-2021 seasons who won All NBA
teams (**not** the All Defensive Teams), as well as for players who were in the All-Star Game (**not** the rookie all-star gam

```
# Here and for all future questions, feel free to add as many code chunks as you like. Do NC
h, we'll want to see your code.
```

```
#I am using seperate code chunks to represent my work on data manipulation, then one code ch
different values asked for#
#I joined the awards and player statistics datasets together to have all relevant informatic
ng so by the common season and nbapersonid features. Then, I created a points per game varia
need it for the answers for question 1, by dividing total points that season by games.#
library(dplyr)
part1df <- player_data %>% inner_join(awards,
        by=c('nbapersonid'='nbapersonid', 'season'='season'), relationship = 'many-to-many')
  mutate(ppg=points/games)
```

```
#For each of the next 4 chunks, I created a smaller subset that only inclued players selecte
ors team asked for. I then used summarize to average out ppg for each player and report it b
e.# I did this for all 4 values identically.#
#1st Team Work#
first_team_ppg <- part1df %>%
  filter(`All NBA First Team` == 1) %>%
  summarize(mean_ppg1 = mean(ppg))
first_team_ppg
```

```
## # A tibble: 1 × 1
##   mean_ppg1
##       <dbl>
## 1      25.9
```

```
#2nd Team Work#
second_team_ppg <- part1df %>%
  filter(`All NBA Second Team` == 1) %>%
  summarize(mean_ppg2 = mean(ppg))
second_team_ppg
```

```
## # A tibble: 1 × 1
##   mean_ppg2
##       <dbl>
## 1      23.1
```

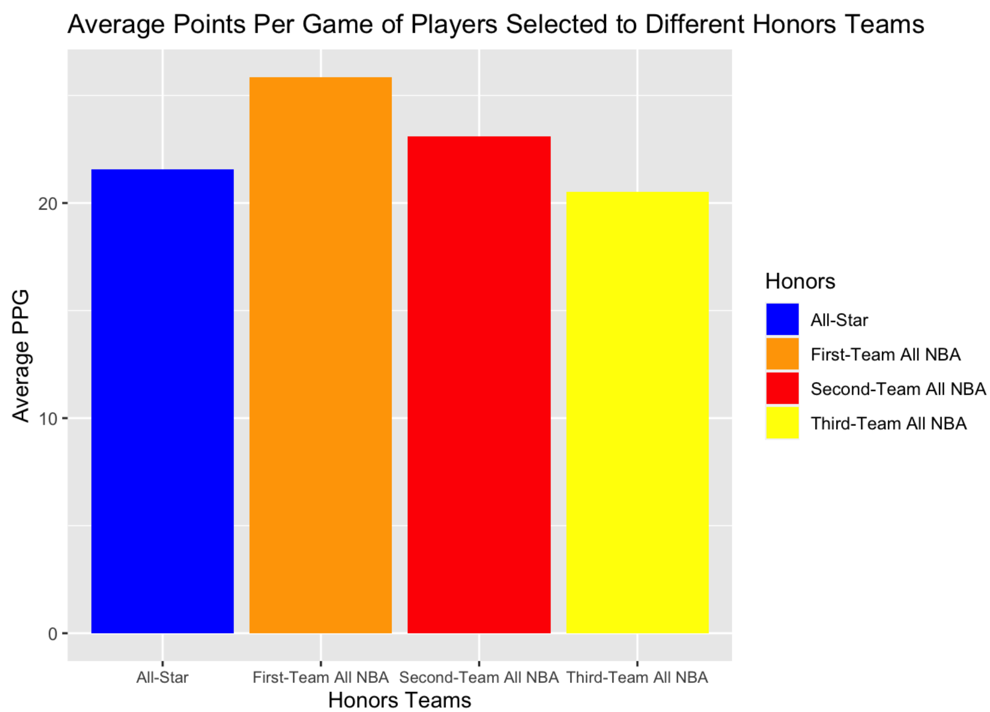
```
#3rd Team Work#
third_team_ppg <- part1df %>%
  filter(`All NBA Third Team` == 1) %>%
  summarize(mean_ppg3 = mean(ppg))
third_team_ppg
```

```
## # A tibble: 1 × 1
##   mean_ppg3
##       <dbl>
## 1      20.5
```

```
#All-Star Work#
all_star_ppg <- part1df %>%
  filter(`all_star_game` == 'TRUE') %>%
  summarize(mean_ppg4 = mean(ppg))
all_star_ppg
```

```
## # A tibble: 1 × 1
##   mean_ppg4
##       <dbl>
## 1      21.6
```

```
#I made a graph on top of the 4 values shown just to visualize the differences in average pp
l-nba teams and all-star teams (and perhaps to do a little extra!). I used a dataframe to st
values corresponding to the honors team, and also added custom colors for each one. Using gg
features I was able to make a small plot for all four values, as shown below.#


honors_team <- c("First-Team All NBA", "Second-Team All NBA", "Third-Team All NBA", "All-Sta
avg_ppg <- c(sum(first_team_ppg), sum(second_team_ppg), sum(third_team_ppg), sum(all_star_pp

avg_ppg_plot_data <- data.frame(Honors = honors_team, PPG = avg_ppg)

plot_colors <- c('All-Star' = "blue", 'First-Team All NBA' = "orange", 'Second-Team All NBA'
All NBA' = "yellow")

ggplot(avg_ppg_plot_data, aes(x = Honors, y = PPG, fill = Honors)) +
  geom_bar(stat = "identity") +
  scale_fill_manual(values = plot_colors) +
  labs(title = "Average Points Per Game of Players Selected to Different Honors Teams", x =
"Average PPG") +
  theme(axis.text.x = element_text(size = 8))
```



ANSWER 1:

1st Team: 25.8 points per game
2nd Team: 23.1 points per game
3rd Team: 20.5 points per game
All-Star: 21.6 points per game

Question 2

QUESTION: What was the average number of years of experience in the league it takes for players to make their firs
or 3rd team)? Please limit your sample to players drafted in 2007 or later who did eventually go on to win at least one
example:

- Luka Doncic is in the dataset as 2 years. He was drafted in 2018 and won his first All NBA award in 2019 (whic
- LeBron James is not in this dataset, as he was drafted prior to 2007.
- Lu Dort is not in this dataset, as he has not received any All NBA honors.

```
#I took the large dataset that had combined player and awards statistics, and I filtered it
players drafted after 2006, and those who have been on any All-NBA team. I then created a ne
lated how long it took for each record to record their All-NBA selection from the draft year
as people play in the same year they are drafted, and are given the award the end of the las
eaning that if I were to not add the plus one it would be a year behind. For example, if som
named to an All-NBA team their rookie year, without the +1 it would show 0 years, which is i
necessary due to the data. Then, I grouped each player by their name, and filtered the data
the first season they recieved an award. This was the variable being asked for, so I then av
all players and found the answer.#


mean_years_experience <- part1df %>%
  filter(draftyear > 2006,
         `All NBA First Team` == 1 | `All NBA Second Team` == 1 | `All NBA Third Team` == 1)
  mutate(years_until_allNBA = season - draftyear + 1) %>%
  group_by(player) %>%
  filter(season == min(season)) %>%
  ungroup() %>%
  summarize(mean_years = mean(years_until_allNBA))

mean_years_experience
```

```
## # A tibble: 1 × 1
##   mean_years
##        <dbl>
## 1       4.67
```

ANSWER 2:

4.7 Years

Data Cleaning Interlude

You're going to work to create a dataset with a "career outcome" for each player, representing the highest level of su
achieved for **at least two** seasons *after his first four seasons in the league* (examples to follow below!). To do this, yo
level outcomes. On a single season level, the outcomes are:

- Elite: A player is "Elite" in a season if he won any All NBA award (1st, 2nd, or 3rd team), MVP, or DPOY in that s
- All-Star: A player is "All-Star" in a season if he was selected to be an All-Star that season.
- Starter: A player is a "Starter" in a season if he started in at least 41 games in the season OR if he played at lea
  season.
- Rotation: A player is a "Rotation" player in a season if he played at least 1000 minutes in the season.
- Roster: A player is a "Roster" player in a season if he played at least 1 minute for an NBA team but did not me
- Out of the League: A player is "Out of the League" if he is not in the NBA in that season.

We need to make an adjustment for determining Starter/Rotation qualifications for a few seasons that didn't have 82
that there were 66 possible games in the 2011 lockout season and 72 possible games in each of the 2019 and 2020
due to covid. Specifically, if a player played 900 minutes in 2011, he **would** meet the rotation criteria because his fina
considered to be 900 * (82/66) = 1118. Please use this math for both minutes and games started, so a player who sta
2020 would be considered to have started 38 * (82/72) = 43 games, and thus would qualify for starting 41. Any answ
assuming you round the multiplied values to the nearest whole number.

Note that on a season level, a player's outcome is the highest level of success he qualifies for in that season. Thus, s
Alexander was both All-NBA 1st team and an All-Star last year, he would be considered to be "Elite" for the 2022 sea
for a career outcome of All-Star if in the rest of his career he made one more All-Star game but no more All-NBA tear
hypothetical, and Shai has not yet played enough to have a career outcome.

Examples:

- A player who enters the league as a rookie and has season outcomes of Roster (1), Rotation (2), Rotation (3), F
the League (6+) would be considered "Out of the League," because after his first four seasons, he only has a s
does not qualify him for any success outcome.
- A player who enters the league as a rookie and has season outcomes of Roster (1), Rotation (2), Starter (3), Sta
(6), All-Star (7), Elite (8), Starter (9) would be considered "All-Star," because he had at least two seasons after h
production or higher.
- A player who enters the league as a rookie and has season outcomes of Roster (1), Rotation (2), Starter (3), Sta
(6), Rotation (7), Rotation (8), Roster (9) would be considered a "Starter" because he has two seasons after his
production.

Question 3

QUESTION: There are 73 players in the `player_data` dataset who have 2010 listed as their draft year. How many o
career outcome in each of the 6 buckets?

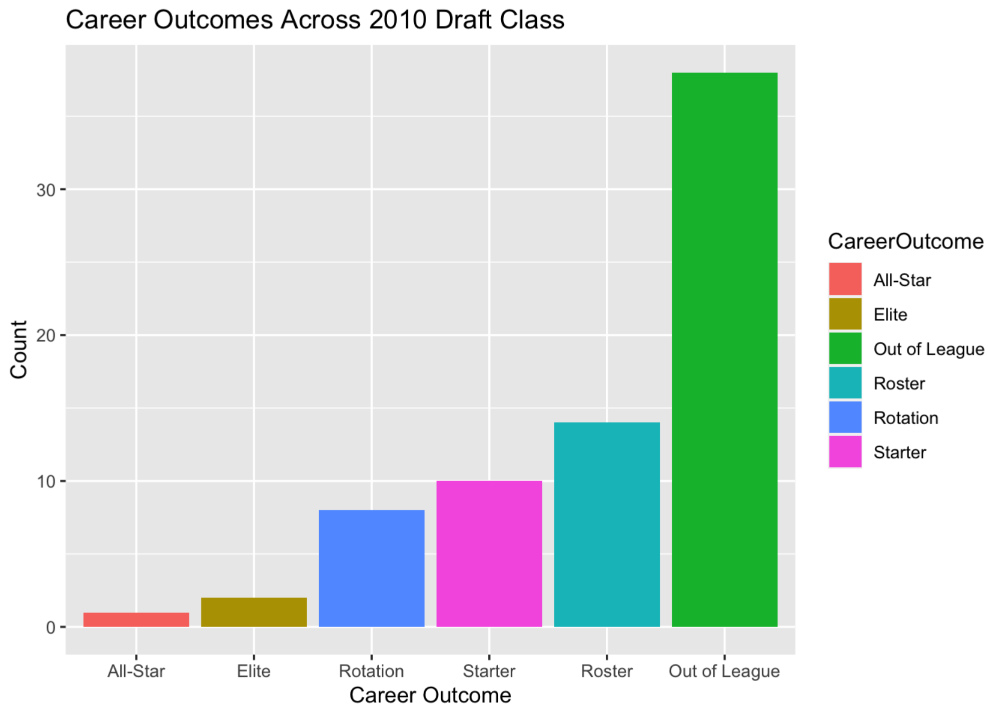
```
#Note: In the following r code chunk, I am very aware this is not concise. This took me a lo
t, and while I am most likely sure there were multiple ways to find this easier, this was th
found to get the second highest season outcome when the frequency of the highest season outc
n combine them with the myriad of ifelse statements you see below. Again, I am aware this ma
al conciseness and correctness, but overall I do believe my thought process of how to comple
as well as my range of r knowledge in both tidyverse and base r was shown. Hopefully you und
re of a time-crunch my reasoning for this explanation but I hope this suffices! -Will Rice#
```

```
#I created a new function altogether that would account for those specific seasons needing t
o the lockout and covid halts in play, respectively. I used if else so as to not change the
re not changed in terms of max games played.#

calculate_mins <- function(mins, season) {
  ifelse(season == 2011, mins * (82/66), ifelse(season %in% c(2019, 2020), mins * (82/72), m
}

#Next, I once again joined the two datasets together for a fresh new dataset, so I can creat
n question: Highest Outcome. I followed the specific criteria asked for in terms of differen
case_when function, then implemented my new minutes/games variable to alter those 3 seasons'
s. I also rounded them as the first time I viewed the dataset it looked off as all of the ot
rs was nice and rounded. Then, I filtered for the 2010 draft year, as that was part of the q
cluded all player's first four years since being drafted as those did not count towards a ca
red by player and season and made sure there were no duplicate years: tehre were players tra
e season they had duplicated values that would count for them and in most cases help them ge
f similar playing time or better, which would not be fair, as that only occurred in one seac
so only one would show up. Then I grouped by player and their highest outcome, and found the
outcome, which was ranked in the order I told r to rank using the arrange(match) function. I
r, I used the slice function to find the highest outcome each player had during their 5th ye
t outcome had a frequency of 2, it was lsited as their highest outcome. If their highest out
a frequency of 1, they were listed as out of the league for the logical reasons that the sec
one outcome means the other years they could only be out of the league. Evryone else was NA.
ocess below.#



career_outcomes <- player_data %>%
  left_join(awards, by = c("nbapersonid", "season"), relationship = 'many-to-many') %>%
  mutate(HighestOutcome = case_when(
    `All NBA First Team` == 1 | `All NBA Second Team` == 1 | `All NBA Third Team` == 1 |
    `Most Valuable Player_rk` == 1 | `Defensive Player Of The Year_rk` == 1 ~ "Elite",
    all_star_game == 'TRUE' ~ "All-Star",
    games_start >= 41 | mins >= 2000 ~ "Starter",
    mins >= 1000 ~ "Rotation",
    mins >= 1 ~ "Roster",
    TRUE ~ "Out of the League"
  )) %>%
  mutate(mins = calculate_mins(mins, season),
         games_start = calculate_mins(games_start, season),
         mins = as.integer(round(mins)),
         games_start = as.integer(round(games_start))) %>%
  filter(season >= draftyear + 4, draftyear == 2010) %>%
  group_by(player, season) %>%
  filter(row_number() == 1) %>%
  ungroup() %>%
  group_by(player, HighestOutcome) %>%
  summarize(Frequency = n(), .groups = "keep") %>%
  arrange(match(HighestOutcome, c("Elite", "All-Star", "Starter", "Rotation", "Roster", "Out
>%
  group_by(player) %>%
  slice(1) %>%
  mutate(CareerOutcome = if_else(Frequency >= 2, HighestOutcome, NA),
         CareerOutcome = if_else(HighestOutcome == 'Roster' & Frequency == 1, 'Out of League
>%
  group_by(player) %>%
  ungroup()

#This is where I was left frustrated and ended up happy with the outcome, though I realize I
how often i'm creating new datasets. I created another datatset, identical to the previous i
I create the secondhighestoutcome variable, it is mostly the same process. I created secondh
is the same as highest outcome except using some if_else statements and named differently fc
r. Next, I counted frequencies and listed importance of eadch outcome in order once more, ex
ion this time to find the second highest outcome for each player. I joined the two datasets
each players highest and second highest outcome (although it only listed the highest if the
the rest were NA). Deselecting the frequencies then left me with a simple dataset of each pl
es.



single_season_outcomes <- player_data %>%
```

```
  left_join(awards, by = c("nbapersonid", "season"), relationship = 'many-to-many') %>%
  mutate(Elite = (`All NBA First Team` == 1 | `All NBA Second Team` == 1 | `All NBA Third Te
Player Of The Year_rk` == 1 | `Most Valuable Player_rk` == 1) + 0,
  Elite = ifelse(is.na(Elite), 0, Elite)) %>%
  mutate(AllStar = (all_star_game == 'TRUE' & Elite == 0) + 0,
  AllStar = ifelse(is.na(AllStar), 0, AllStar)) %>%
  mutate(Starter = ((games_start > 40 | mins > 1999) & Elite == 0 & AllStar == 0) + 0,
  Starter = ifelse(is.na(Starter), 0, Starter)) %>%
  mutate(Rotation = (mins > 999 & Elite == 0 & AllStar == 0 & Starter == 0) + 0,
  Rotation = ifelse(is.na(Rotation), 0, Rotation)) %>%
  mutate(Roster = (mins > 0 & Elite == 0 & AllStar == 0 & Starter == 0 & Rotation == 0) + 0,
  Roster = ifelse(is.na(Roster), 0, Roster)) %>%
  mutate(OutOfLeague = ifelse(Elite == 0 & AllStar == 0 & Starter == 0 & Rotation == 0 & Ros
  mutate(mins = if_else(season == 2011, mins*(82/66), mins)) %>%
  mutate(mins = if_else(season == 2019|season == 2020, mins*(82/72), mins)) %>%
  mutate(games_start = if_else(season == 2011, games_start*(82/66), games_start)) %>%
  mutate(games_start = if_else(season == 2019|season == 2020, games_start*(82/72), games_sta
  mutate(mins = as.integer(round(mins))) %>%
  mutate(games_start = as.integer(round(games_start))) %>%
  select(nbapersonid, draftyear, player, season, Elite, AllStar, Starter, Rotation, Roster,
  filter(draftyear == 2010, season > 2013) %>%
  group_by(player, season) %>%
  filter(row_number() == 1) %>%
  ungroup() %>%
  mutate(SecondHighestOutcome = if_else(Elite == 1, 'Elite', if_else(AllStar == 1, 'All-Star
1, 'Starter', if_else(Rotation == 1, 'Rotation', if_else(Roster == 1, 'Roster', 'Out of Leag
  group_by(player, SecondHighestOutcome) %>%
  summarize(Frequency = n(), .groups = "keep") %>%
  arrange(match(SecondHighestOutcome, c("Elite", "All-Star", "Starter", "Rotation", "Roster"
e"))) %>%
  group_by(player) %>%
  slice(2)

second_highest_outcomes <- single_season_outcomes


total_2010_outcomes <- left_join(career_outcomes, second_highest_outcomes, by = "player") %>
  select(-Frequency.x, -Frequency.y) %>%
  mutate(CareerOutcome = if_else(is.na(CareerOutcome), SecondHighestOutcome, CareerOutcome))
  select(-SecondHighestOutcome)


#I replaced all NA values in the highest outcome with its second highest outcome, and then d
est outcome. Next, I used the summarize function to count how many of each career outcome th
d. I also noticed I only retained 44 values after the 4 year of playing. This meant that if
e 73, and only 44 had a distinct outcome, there were 29 who never played in their fifth year
lly qualify as out of league. I created a variable that counted 29 more out of leagues in th
reated one table 'outcome_counts' that showed the counts of all 6 different outcomes for the


outcome_counts <- total_2010_outcomes %>%
  group_by(CareerOutcome) %>%
  summarize(count = n())

out_of_league_before_5th_year <- data.frame(CareerOutcome = "Out of League", count = 29)

outcome_counts <- bind_rows(outcome_counts, out_of_league_before_5th_year) %>%
  group_by(CareerOutcome) %>%
  summarize(count = sum(count))
outcome_counts
```

```
## # A tibble: 6 × 2
##   CareerOutcome count
##   <chr>         <dbl>
## 1 All-Star          1
## 2 Elite             2
## 3 Out of League    38
## 4 Roster           14
## 5 Rotation          8
## 6 Starter          10
```

```
#Next, I created again a ggplot visual to show counts by outcome, just to display how hard i
in the league, but thrive! I thought with some interesting colors generically given that it
ive to this otherwise code-heavy question due to my trouble withbthe second highest outcome.

ggplot(outcome_counts, aes(x = reorder(CareerOutcome, count), y = count, fill = CareerOutcom
  geom_bar(stat = "identity") +
  labs(x = "Career Outcome", y = "Count", title = "Career Outcomes Across 2010 Draft Class")
```



ANSWER 3:

Elite: 2 players.
All-Star: 1 players.
Starter: 10 players.
Rotation: 8 players.
Roster: 14 players.
Out of League: 38 players.

Open Ended Modeling Question

In this question, you will work to build a model to predict a player's career outcome based on information up through career.

This question is intentionally left fairly open ended, but here are some notes and specifications.

1. We know modeling questions can take a long time, and that qualified candidates will have different levels of ex modeling. Don't be discouraged. It's not our intention to make you spend excessive time here. If you get your think you could do better by spending a lot more time, you can just write a bit about your ideas for future impr Further, we're more interested in your thought process and critical thinking than we are in specific modeling te features is more important than using fancy mathematical machinery, and a successful candidate could use a
2. You may use any data provided in this project, but please do not bring in any external sources of data. Note th provided goes back to 2007, All NBA and All Rookie team voting is only included back to 2011.

3. A player needs to complete at least three additional seasons after their first four to be considered as having a c our dataset. (We are using 3+ instead of 2+ just to give each player a little more time to accumulate high level s his career). Because the dataset in this project ends in 2021, this means that a player would need to have had '20, and '19 seasons after his first four years, and thus his first four years would have been '18, '17, '16, and '1 **your training data to players who were drafted in or before the 2015 season.** Karl-Anthony Towns was the
4. Once you build your model, predict on all players who were drafted in 2018-2021 (They have between 1 and 4 and have not yet started accumulating seasons that inform their career outcome).
5. You can predict a single career outcome for each player, but it's better if you can predict the probability that ea outcome bucket.
6. Include, as part of your answer:

- A brief written overview of how your model works, targeted towards a decision maker in the front office withou background.
- What you view as the strengths and weaknesses of your model.
- How you'd address the weaknesses if you had more time and or more data.
- A ggplot or ggplotly visualization highlighting some part of your modeling process, the model itself, or your res
- Your predictions for Shai Gilgeous-Alexander, Zion Williamson, James Wiseman, and Josh Giddey.
- (Bonus!) An html table (for example, see the package `reactable`) containing all predictions for the players dra

```
#Again, I created a master dataset that joined together awards and player data, mutating any
to a 0. The, I went back and remedied a few choice columns: draftpick was given an NA again
s was all nba and all rookie voting ranks before 2011, as there was no data. Then, I created
ics and rounded them to one decimal point, while I deselected some of the stats I did not be
dent, significant correlation to a potential career outcome, or that were already a part of
le.


#MODEL DATASET#

model_data <- left_join(player_data, awards, by = c("nbapersonid", "season"), relationship =
  mutate_all(~ replace(., is.na(.), 0)) %>%
  mutate(draftpick = if_else(draftpick == 0, NA, draftpick)) %>%
  mutate(all_nba_points_rk = if_else(all_nba_points_rk == 0 & season < 2011, NA, all_nba_poi
  mutate(all_rookie_points_rk = if_else(all_rookie_points_rk == 0 & season < 2011, NA, all_r
  filter(draftyear >= 2003) %>%
  mutate(ppg = (points/games)) %>%
  mutate(rpg = (tot_reb/games)) %>%
  mutate(apg = (ast/games)) %>%
  mutate(bpg = (blocks/games)) %>%
  mutate(spg = (steals/games)) %>%
  mutate_at(vars(apg, spg, bpg, ppg, rpg), ~ round(., 1)) %>%
  select(-draftpick, -nbateamid, -'Player Of The Month', -'Rookie Of The Month', -'Player Of
points_rk, -all_rookie_points_rk, -allstar_rk)

model_data
```

```
## # A tibble: 7,279 × 67
##    nbapersonid player draftyear season team  games games_start  mins   fgm   fga
##          <dbl> <chr>      <dbl>  <dbl> <chr> <dbl>       <dbl> <dbl> <dbl> <dbl>
##  1        2585 Zaza …      2003   2007 ATL      62           5   944   107   245
##  2      200780 Solom…      2006   2007 ATL      35           0   145    12    30
##  3        2746 Josh …      2004   2007 ATL      81          81  2873   518  1133
##  4      201151 Acie …      2007   2007 ATL      56           6   865    95   237
##  5      101136 Salim…      2005   2007 ATL      35           0   402    65   180
##  6        2735 Josh …      2004   2007 ATL      76           0  2274   327   573
##  7      200978 Jerem…      2006   2007 ATL      19           0    88    13    31
##  8      201143 Al Ho…      2007   2007 ATL      81          77  2540   333   668
##  9      101107 Marvi…      2005   2007 ATL      80          80  2765   424   918
## 10      200749 Sheld…      2006   2007 ATL      36           0   414    37   100
## # ℹ 7,269 more rows
## # ℹ 57 more variables: fgp <dbl>, fgm3 <dbl>, fga3 <dbl>, fgp3 <dbl>,
## #   fgm2 <dbl>, fga2 <dbl>, fgp2 <dbl>, efg <dbl>, ftm <dbl>, fta <dbl>,
## #   ftp <dbl>, off_reb <dbl>, def_reb <dbl>, tot_reb <dbl>, ast <dbl>,
## #   steals <dbl>, blocks <dbl>, tov <dbl>, tot_fouls <dbl>, points <dbl>,
## #   PER <dbl>, FTr <dbl>, off_reb_pct <dbl>, def_reb_pct <dbl>,
## #   tot_reb_pct <dbl>, ast_pct <dbl>, stl_pct <dbl>, blk_pct <dbl>, …
```

```
calculate_mins <- function(mins, season) {
  ifelse(season == 2011, mins * (82/66), ifelse(season %in% c(2019, 2020), mins * (82/72), m
}


#Next, I made my training dataset. I took the model dataset and filtered out only players dr
2015. This was imperative to me as to have all of the first four years of data on every play
as well as the years beyond that which made to help generate a distinct outcome. For the nex
I reproduced finding the highest career outcome after doing some duplicate codes to ensure t
eft with the next highest seaosnal outcome.


#TRAINING DATASET#

training_set <- model_data %>%
  filter(draftyear < 2016 & draftyear > 2006) %>%
  mutate(SeasonOutcome = case_when(
    `All NBA First Team` == 1 | `All NBA Second Team` == 1 | `All NBA Third Team` == 1 |
    `Most Valuable Player_rk` == 1 | `Defensive Player Of The Year_rk` == 1 ~ "Elite",
    all_star_game == 1 ~ "All-Star",
    games_start >= 41 | mins >= 2000 ~ "Starter",
    mins >= 1000 ~ "Rotation",
    mins >= 1 ~ "Roster",
    TRUE ~ "Out of the League"
  )) %>%
  mutate(mins = calculate_mins(mins, season),
         games_start = calculate_mins(games_start, season),
         mins = as.integer(round(mins)),
         games_start = as.integer(round(games_start))) %>%
  filter(season >= draftyear + 4) %>%
  group_by(player, season) %>%
  filter(row_number() == 1) %>%
  ungroup() %>%
  group_by(player, SeasonOutcome) %>%
  summarize(Frequency = n(), .groups = "keep") %>%
  arrange(match(SeasonOutcome, c("Elite", "All-Star", "Starter", "Rotation", "Roster", "Out
  group_by(player) %>%
  slice(1) %>%
  mutate(CareerOutcome = if_else(Frequency >= 3, SeasonOutcome, NA),
         CareerOutcome = if_else(SeasonOutcome == 'Roster' & Frequency == 1, 'Out of League'
  group_by(player) %>%
  ungroup()


training_data_2 <- model_data %>%
  filter(draftyear < 2016 & draftyear > 2006) %>%
  mutate(SecondHighestSeasonOutcome = case_when(
    `All NBA First Team` == 1 | `All NBA Second Team` == 1 | `All NBA Third Team` == 1 |
    `Most Valuable Player_rk` == 1 | `Defensive Player Of The Year_rk` == 1 ~ "Elite",
    all_star_game == 1 ~ "All-Star",
    games_start >= 41 | mins >= 2000 ~ "Starter",
    mins >= 1000 ~ "Rotation",
    mins >= 1 ~ "Roster",
    TRUE ~ "Out of the League"
  )) %>%
  mutate(mins = calculate_mins(mins, season),
         games_start = calculate_mins(games_start, season),
         mins = as.integer(round(mins)),
         games_start = as.integer(round(games_start))) %>%
  filter(season >= draftyear + 4) %>%
  group_by(player, season) %>%
  filter(row_number() == 1) %>%
  ungroup() %>%
  group_by(player, SecondHighestSeasonOutcome) %>%
  summarize(Frequency = n(), .groups = "keep") %>%
  arrange(match(SecondHighestSeasonOutcome, c("Elite", "All-Star", "Starter", "Rotation", "R
eague"))) %>%
  group_by(player) %>%
  slice(2)
```

```
training_data_third_highest <- model_data %>%
  filter(draftyear < 2016 & draftyear > 2006) %>%
  mutate(ThirdHighestSeasonOutcome = case_when(
    `All NBA First Team` == 1 | `All NBA Second Team` == 1 | `All NBA Third Team` == 1 |
    `Most Valuable Player_rk` == 1 | `Defensive Player Of The Year_rk` == 1 ~ "Elite",
    all_star_game == 1 ~ "All-Star",
    games_start >= 41 | mins >= 2000 ~ "Starter",
    mins >= 1000 ~ "Rotation",
    mins >= 1 ~ "Roster",
    TRUE ~ "Out of the League"
  )) %>%
  mutate(mins = calculate_mins(mins, season),
         games_start = calculate_mins(games_start, season),
         mins = as.integer(round(mins)),
         games_start = as.integer(round(games_start))) %>%
  filter(season >= draftyear + 4) %>%
  group_by(player, season) %>%
  filter(row_number() == 1) %>%
  ungroup() %>%
  group_by(player, ThirdHighestSeasonOutcome) %>%
  summarize(Frequency = n(), .groups = "keep") %>%
  arrange(match(ThirdHighestSeasonOutcome, c("Elite", "All-Star", "Starter", "Rotation", "Ro
ague"))) %>%
  group_by(player) %>%
  slice(3)

View(training_data_third_highest)


total_training_outcomes_draft <- left_join(training_set, training_data_2, by = "player") %>%
  mutate(CareerOutcome = if_else(Frequency.x == 1, if_else(Frequency.y <= 1, NA, SecondHighe
eerOutcome)) %>%
  mutate(CareerOutcome = if_else(Frequency.x == 2, SecondHighestSeasonOutcome, CareerOutcome
  mutate(CareerOutcome = if_else(is.na(CareerOutcome) & is.na(SecondHighestSeasonOutcome), "
rOutcome))
View(total_training_outcomes_draft)



total_training_outcomes <- left_join(total_training_outcomes_draft, training_data_third_high
>%
  select(-SecondHighestSeasonOutcome, -Frequency.y) %>%
  mutate(CareerOutcome = if_else(is.na(CareerOutcome) & is.na(ThirdHighestSeasonOutcome), "O
Outcome)) %>%
  mutate(CareerOutcome = if_else(is.na(CareerOutcome), ThirdHighestSeasonOutcome, CareerOutc

View(total_training_outcomes)


#I then joined to gether the dataset of all players drafted from 2007 and 2015 with the play
eer outcomes, and then filtered all players who did not play long enough to 'Out of League.'
for my training data.

training_data_3 <- model_data %>%
  filter(draftyear < 2016 & draftyear > 2006)

training_dataset_main <- left_join(training_data_3, total_training_outcomes, by = "player")
  mutate(CareerOutcome = if_else(is.na(CareerOutcome) & is.na(SeasonOutcome), 'Out of League
>%
  select(-SeasonOutcome)
View(training_dataset_main)



#I then took their first four years of data as the training data. This is because we are pre
ata's first one-four years of data to make a prediction. We have career outcomes for each of
e can also aggregate their first four years to build a model that can help us evaluate the t
```

ar trends in variables and data. I took the next massive few chunks of code to create aggreg the first four years of each players career. I believed having one record for each player in e to fit into a model better, and it just took a bit of time to write down every stat so as rectly.

```
first_4_years <- training_dataset_main %>%
  filter(season <= draftyear + 3) %>%
  group_by(player, season) %>%
  filter(row_number() == 1) %>%
  ungroup() %>%
  mutate(MVP_Award = if_else(`Most Valuable Player_rk` == 1, 1, 0)) %>%
  mutate(MVP_Candidate = if_else(`Most Valuable Player_rk` > 1 & `Most Valuable Player_rk` <
  mutate(MIP_Award = if_else(`Most Improved Player_rk` == 1, 1, 0)) %>%
  mutate(MIP_Candidate = if_else(`Most Improved Player_rk` > 1 & `Most Improved Player_rk` <
  mutate(DPOY_Award = if_else(`Defensive Player Of The Year_rk` == 1, 1, 0)) %>%
  mutate(DPOY_Candidate = if_else(`Defensive Player Of The Year_rk` > 1 & `Defensive Player
1, 0)) %>%
  mutate(ROY_Award = if_else(`Rookie Of The Year_rk` == 1, 1, 0)) %>%
  mutate(ROY_Candidate = if_else(`Rookie Of The Year_rk` > 1 & `Rookie Of The Year_rk` < 10,
  mutate(Sixth_Man_Award = if_else(`Sixth Man Of The Year_rk` == 1, 1, 0)) %>%
  mutate(Sixth_Man_Candidate = if_else(`Sixth Man Of The Year_rk` > 1 & `Sixth Man Of The Ye

first_4_years_training <- first_4_years %>%
  group_by(player) %>%
  summarize(
    all_star_games = sum(all_star_game),
    all_nba_defensive_first_team_selections = sum(`All NBA Defensive First Team`),
    all_nba_defensive_second_team_selections = sum(`All NBA Defensive Second Team`),
    all_nba_first_team_selections = sum(`All NBA First Team`),
    all_nba_second_team_selections = sum(`All NBA Second Team`),
    all_nba_third_team_selections = sum(`All NBA Third Team`),
    all_rookie_first_team_selections = sum(`All Rookie First Team`),
    all_rookie_second_team_selections = sum(`All Rookie Second Team`),
    MVP_count = sum(`MVP_Award`),
    MIP_count = sum(`MIP_Award`),
    DPOY_count = sum(`DPOY_Award`),
    ROY_count = sum(`ROY_Award`),
    sixth_man_count = sum(`Sixth_Man_Award`),
    MVP_candidate_count = sum(`MVP_Candidate`),
    MIP_candidate_count = sum(`MIP_Candidate`),
    DPOY_candidate_count = sum(`DPOY_Candidate`),
    ROY_candidate_count = sum(`ROY_Candidate`),
    sixth_man_candidate_count = sum(`Sixth_Man_Candidate`),
    finals_mvp = sum(`Bill Russell NBA Finals MVP`),
    total_points = sum(points),
    total_assists = sum(points),
    total_rebounds = sum(tot_reb),
    total_steals = sum(steals),
    total_blocks = sum(blocks),
    total_turnovers = sum(tov),
    total_field_goals = sum(fgm),
    total_field_goals_attempted = sum(fga),
    total_two_pointers = sum(fgm2),
    total_two_pointers_attempted = sum(fga2),
    total_three_pointers = sum(fgm3),
    total_three_pointers_attempted = sum(fga3),
    total_free_throws = sum(ftm),
    total_free_throws_attempted = sum(fta),
    total_games_started = sum(games_start),
    total_minutes = sum(mins),
    total_defensive_win_shares = sum(DWS),
    total_offensive_win_shares = sum(OWS),
    total_win_shares = sum(WS),
    total_defensive_box_plus_minus = sum(DBPM),
    total_offensive_box_plus_minus = sum(OBPM),
    total_box_plus_minus = sum(BPM),
    total_VORP = sum(VORP),
    total_games = sum(games),
    total_PER = sum(PER),
```

```
    total_usg = sum(usg)

  )


first_4_years_aggregated <- first_4_years_training %>%
  mutate(ppg = (total_points/total_games)) %>%
  mutate(apg = (total_assists/total_games)) %>%
  mutate(rpg = (total_rebounds/total_games)) %>%
  mutate(bpg = (total_blocks/total_games)) %>%
  mutate(spg = (total_steals/total_games)) %>%
  mutate(topg = (total_turnovers/total_games)) %>%
  mutate(fgpg = (total_field_goals/total_games)) %>%
  mutate(fgapg = (total_field_goals_attempted/total_games)) %>%
  mutate(fgpercentage = (fgpg/fgapg)) %>%
  mutate(fg2pg = (total_two_pointers/total_games)) %>%
  mutate(fg2apg = (total_two_pointers_attempted/total_games)) %>%
  mutate(fg2percentage = (fg2pg/fg2apg)) %>%
  mutate(fg3pg = (total_three_pointers/total_games)) %>%
  mutate(fg3apg = (total_three_pointers_attempted/total_games)) %>%
  mutate(fg3percentage = (fg3pg/fg3apg)) %>%
  mutate(ftpg = (total_free_throws/total_games)) %>%
  mutate(ftapg = (total_free_throws_attempted/total_games)) %>%
  mutate(ftpercentage = (ftpg/ftapg)) %>%
  mutate(mpg = (total_minutes/total_games)) %>%
  mutate(average_DWS = (total_defensive_win_shares/4)) %>%
  mutate(average_OWS = (total_offensive_win_shares/4)) %>%
  mutate(average_WS = (total_win_shares/4)) %>%
  mutate(average_DBPM = (total_defensive_box_plus_minus/4)) %>%
  mutate(average_OBPM = (total_offensive_box_plus_minus/4)) %>%
  mutate(average_BPM = (total_box_plus_minus/4)) %>%
  mutate(average_VORP = (total_VORP/4)) %>%
  mutate(average_games = (total_games/4)) %>%
  mutate(average_games_started = (total_games_started/4)) %>%
  mutate(average_usg = (total_usg/4)) %>%
  mutate(average_PER = (total_PER/4)) %>%
  select(-total_points, -total_assists, -total_rebounds, -total_blocks, -total_steals, -tota
ield_goals, -total_field_goals_attempted, -total_two_pointers, -total_two_pointers_attempted
rs, -total_three_pointers_attempted, -total_free_throws, -total_free_throws_attempted, -tota
al_minutes, -total_games, -total_VORP, -total_defensive_win_shares, -total_offensive_win_sha
s, -total_defensive_box_plus_minus, -total_offensive_box_plus_minus, -total_box_plus_minus)
  mutate_at(vars(ppg, apg, rpg, bpg, spg, topg, fgpg, fgapg, fg2pg, fg2apg, fg3pg, fg3apg, f
und(., 1)) %>%
  mutate_at(vars(average_games_started, average_games), ~ round(., 0)) %>%
  mutate_at(vars(fgpercentage, fg2percentage, fg3percentage, ftpercentage, average_BPM, aver
M, average_DWS, average_OWS, average_WS, average_VORP), ~ round(., 3))


#I next joined together the two datasets that had all of the aggregate stats with other stat
the name of the player, draft year, actual career outcome, etc. I created a small change in
e range in the first four years they played in stead of four seprate identical records excep
player.

model_training_dataset <- left_join(first_4_years_aggregated, first_4_years %>% select(nbape
r, draftyear, CareerOutcome), by = "player") %>%
  group_by(player) %>%
  mutate(years = paste(min(season), max(season), sep = "-"), .groups = "drop") %>%
  select(-season, -.groups) %>%
  ungroup() %>%
  distinct(player, .keep_all = TRUE)
View(model_training_dataset)


#I did the same thing as the training dataset, instead using players drafted in or after 201
ng dataset. I then created years of experience to help with some of the averages per year. A
layers in the training had 4 years of data, most of these players have varying amounts betwe
eir individual experience amount and used that variable to divide by specific statistics.


#TESTING DATASET#
```

```
testing_dataset <- model_data %>%
  filter(draftyear >= 2018) %>%
  group_by(player, season) %>%
  filter(row_number() == 1) %>%
  ungroup() %>%
  mutate(MVP_Award = if_else(`Most Valuable Player_rk` == 1, 1, 0)) %>%
  mutate(MVP_Candidate = if_else(`Most Valuable Player_rk` > 1 & `Most Valuable Player_rk` <
  mutate(MIP_Award = if_else(`Most Improved Player_rk` == 1, 1, 0)) %>%
  mutate(MIP_Candidate = if_else(`Most Improved Player_rk` > 1 & `Most Improved Player_rk` <
  mutate(DPOY_Award = if_else(`Defensive Player Of The Year_rk` == 1, 1, 0)) %>%
  mutate(DPOY_Candidate = if_else(`Defensive Player Of The Year_rk` > 1 & `Defensive Player
1, 0)) %>%
  mutate(ROY_Award = if_else(`Rookie Of The Year_rk` == 1, 1, 0)) %>%
  mutate(ROY_Candidate = if_else(`Rookie Of The Year_rk` > 1 & `Rookie Of The Year_rk` < 10,
  mutate(Sixth_Man_Award = if_else(`Sixth Man Of The Year_rk` == 1, 1, 0)) %>%
  mutate(Sixth_Man_Candidate = if_else(`Sixth Man Of The Year_rk` > 1 & `Sixth Man Of The Ye
>%
  mutate(Years_Experience = season - draftyear + 1)

View(testing_dataset)

testing_dataset_summed_up <- testing_dataset %>%
  group_by(player) %>%
  summarize(
    all_star_games = sum(all_star_game),
    all_nba_defensive_first_team_selections = sum(`All NBA Defensive First Team`),
    all_nba_defensive_second_team_selections = sum(`All NBA Defensive Second Team`),
    all_nba_first_team_selections = sum(`All NBA First Team`),
    all_nba_second_team_selections = sum(`All NBA Second Team`),
    all_nba_third_team_selections = sum(`All NBA Third Team`),
    all_rookie_first_team_selections = sum(`All Rookie First Team`),
    all_rookie_second_team_selections = sum(`All Rookie Second Team`),
    MVP_count = sum(`MVP_Award`),
    MIP_count = sum(`MIP_Award`),
    DPOY_count = sum(`DPOY_Award`),
    ROY_count = sum(`ROY_Award`),
    sixth_man_count = sum(`Sixth_Man_Award`),
    MVP_candidate_count = sum(`MVP_Candidate`),
    MIP_candidate_count = sum(`MIP_Candidate`),
    DPOY_candidate_count = sum(`DPOY_Candidate`),
    ROY_candidate_count = sum(`ROY_Candidate`),
    sixth_man_candidate_count = sum(`Sixth_Man_Candidate`),
    finals_mvp = sum(`Bill Russell NBA Finals MVP`),
    total_points = sum(points),
    total_assists = sum(points),
    total_rebounds = sum(tot_reb),
    total_steals = sum(steals),
    total_blocks = sum(blocks),
    total_turnovers = sum(tov),
    total_field_goals = sum(fgm),
    total_field_goals_attempted = sum(fga),
    total_two_pointers = sum(fgm2),
    total_two_pointers_attempted = sum(fga2),
    total_three_pointers = sum(fgm3),
    total_three_pointers_attempted = sum(fga3),
    total_free_throws = sum(ftm),
    total_free_throws_attempted = sum(fta),
    total_games_started = sum(games_start),
    total_minutes = sum(mins),
    total_defensive_win_shares = sum(DWS),
    total_offensive_win_shares = sum(OWS),
    total_win_shares = sum(WS),
    total_defensive_box_plus_minus = sum(DBPM),
    total_offensive_box_plus_minus = sum(OBPM),
    total_box_plus_minus = sum(BPM),
    total_VORP = sum(VORP),
    total_games = sum(games),
    total_PER = sum(PER),
    total_usg = sum(usg),
```

```
    years_experience = max(Years_Experience)

  )

View(testing_dataset_summed_up)

testing_dataset_aggregated <- testing_dataset_summed_up %>%
  mutate(ppg = (total_points/total_games)) %>%
  mutate(apg = (total_assists/total_games)) %>%
  mutate(rpg = (total_rebounds/total_games)) %>%
  mutate(bpg = (total_blocks/total_games)) %>%
  mutate(spg = (total_steals/total_games)) %>%
  mutate(topg = (total_turnovers/total_games)) %>%
  mutate(fgpg = (total_field_goals/total_games)) %>%
  mutate(fgapg = (total_field_goals_attempted/total_games)) %>%
  mutate(fgpercentage = (fgpg/fgapg)) %>%
  mutate(fg2pg = (total_two_pointers/total_games)) %>%
  mutate(fg2apg = (total_two_pointers_attempted/total_games)) %>%
  mutate(fg2percentage = (fg2pg/fg2apg)) %>%
  mutate(fg3pg = (total_three_pointers/total_games)) %>%
  mutate(fg3apg = (total_three_pointers_attempted/total_games)) %>%
  mutate(fg3percentage = (fg3pg/fg3apg)) %>%
  mutate(ftpg = (total_free_throws/total_games)) %>%
  mutate(ftapg = (total_free_throws_attempted/total_games)) %>%
  mutate(ftpercentage = (ftpg/ftapg)) %>%
  mutate(mpg = (total_minutes/total_games)) %>%
  mutate(average_DWS = (total_defensive_win_shares/years_experience)) %>%
  mutate(average_OWS = (total_offensive_win_shares/years_experience)) %>%
  mutate(average_WS = (total_win_shares/years_experience)) %>%
  mutate(average_DBPM = (total_defensive_box_plus_minus/years_experience)) %>%
  mutate(average_OBPM = (total_offensive_box_plus_minus/years_experience)) %>%
  mutate(average_BPM = (total_box_plus_minus/years_experience)) %>%
  mutate(average_VORP = (total_VORP/years_experience)) %>%
  mutate(average_games = (total_games/years_experience)) %>%
  mutate(average_games_started = (total_games_started/years_experience)) %>%
  mutate(average_usg = (total_usg/years_experience)) %>%
  mutate(average_PER = (total_PER/years_experience)) %>%
  select(-total_points, -total_assists, -total_rebounds, -total_blocks, -total_steals, -tota
ield_goals, -total_field_goals_attempted, -total_two_pointers, -total_two_pointers_attempted
rs, -total_three_pointers_attempted, -total_free_throws, -total_free_throws_attempted, -tota
al_minutes, -total_games, -total_VORP, -total_defensive_win_shares, -total_offensive_win_sha
s, -total_defensive_box_plus_minus, -total_offensive_box_plus_minus, -total_box_plus_minus)
  mutate_at(vars(ppg, apg, rpg, bpg, spg, topg, fgpg, fgapg, fg2pg, fg2apg, fg3pg, fg3apg, f
und(., 1)) %>%
  mutate_at(vars(average_games_started, average_games), ~ round(., 0)) %>%
  mutate_at(vars(fgpercentage, fg2percentage, fg3percentage, ftpercentage, average_BPM, aver
M, average_DWS, average_OWS, average_WS, average_VORP), ~ round(., 3))


View(testing_dataset_aggregated)




#I created a formula that displayed CareerOutcome as the target variable, with almost every
that had been aggregated over time, including sums of awards within the dataset. I then inst
st package and ran a random forest model on the training data, then tested it on the testing
nged some 0% values to 0 instead of null).


#MODEL TRAINING, PREDICTIONS, AND EVALUATION#

Model_Formula <- CareerOutcome ~ all_star_games + all_nba_defensive_first_team_selections +
ond_team_selections + all_nba_first_team_selections + all_nba_second_team_selections + all_n
ons + all_rookie_first_team_selections + all_rookie_second_team_selections + MVP_count + MIP
DPOY_count + sixth_man_count + MVP_candidate_count + MIP_candidate_count + ROY_candidate_cou
ount + sixth_man_candidate_count + finals_mvp + average_PER + average_usg + ppg + apg + rpg
fgpg + fgapg + fgpercentage + fg2pg + fg2apg + fg2percentage + fg3pg + fg3apg + fg3percentag
percentage + mpg + average_games + average_games_started + average_VORP + average_OWS + aver
+ average_OBPM + average_DBPM + average_BPM
```

```
#Installing Random Forest Model and setting seed for reprocibility#
library(randomForest)
```

```
## randomForest 4.7-1.1
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
set.seed(123)




#Ensuring All Null Values are 0.00%#

model_training_dataset$CareerOutcome <- factor(model_training_dataset$CareerOutcome)
levels(model_training_dataset$CareerOutcome)
```

```
## [1] "All-Star"      "Elite"         "Out of League" "Roster"
## [5] "Rotation"      "Starter"
```

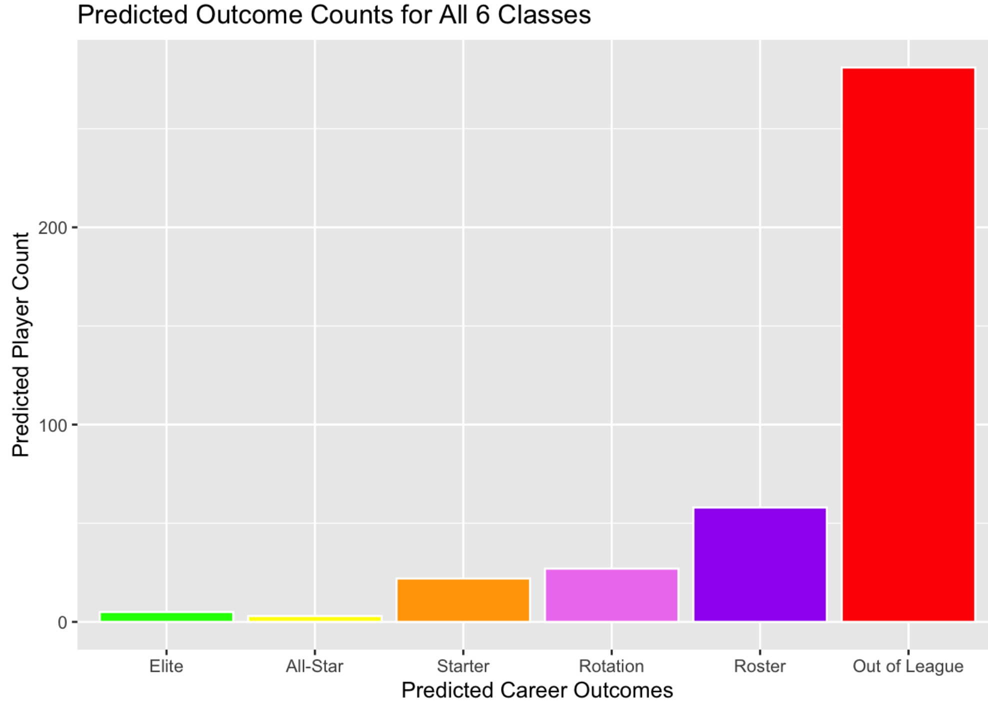
```
View(model_training_dataset$CareerOutcome)


model_training_dataset <- model_training_dataset %>%
  mutate(ftpercentage = if_else(is.na(ftpercentage), 0, ftpercentage))

testing_dataset_aggregated <- testing_dataset_aggregated %>%
  mutate(ftpercentage = if_else(is.na(ftpercentage), 0, ftpercentage))

model_training_dataset <- model_training_dataset %>%
  mutate(fg3percentage = if_else(is.na(fg3percentage), 0, fg3percentage))

testing_dataset_aggregated <- testing_dataset_aggregated %>%
  mutate(fg3percentage = if_else(is.na(fg3percentage), 0, fg3percentage))

model_training_dataset <- model_training_dataset %>%
  mutate(fg2percentage = if_else(is.na(fg2percentage), 0, fg2percentage))

testing_dataset_aggregated <- testing_dataset_aggregated %>%
  mutate(fg2percentage = if_else(is.na(fg2percentage), 0, fg2percentage))

model_training_dataset <- model_training_dataset %>%
  mutate(fgpercentage = if_else(is.na(fgpercentage), 0, fgpercentage))

testing_dataset_aggregated <- testing_dataset_aggregated %>%
  mutate(fgpercentage = if_else(is.na(fgpercentage), 0, fgpercentage))



#Back to random forets models#

rf_model <- randomForest(Model_Formula, data = model_training_dataset)

testing_predictions <- predict(rf_model, newdata = testing_dataset_aggregated)

outcome_probs <- predict(rf_model, newdata = testing_dataset_aggregated, type = "prob")
View(outcome_probs)

outcome_predictions <- cbind(testing_dataset_aggregated[c("player")], testing_predictions)
View(outcome_predictions)

probability_predictions <- cbind(testing_dataset_aggregated[c("player")], outcome_probs)
View(probability_predictions)

#Below is my dataset configured in a later chunk that shows a player, his model-predicted ou
ies of landing in each of the six outcomes. The HTML table shows this too!#

total_predictions <- left_join(outcome_predictions, probability_predictions, by = "player")
  rename('Model Predictions' = 'testing_predictions')
View(total_predictions)


#Above is the evaluation and prediction outcomes of the testing dataset. One dataset has the
for each player, while the other specifically shows the outcomes available as a probability
w is my full description for the model, strengths and weaknesses, and potential solutions to
n the constraints.
```

BRIEF OVERVIEW OF MODEL AND HOW IT WORKS, STRENGTHS AND WEAKNESSE HOW TO ADDRESS WEAKNESSES WITH MORE TIME/KNOWLEDGE/EXPERIENCE

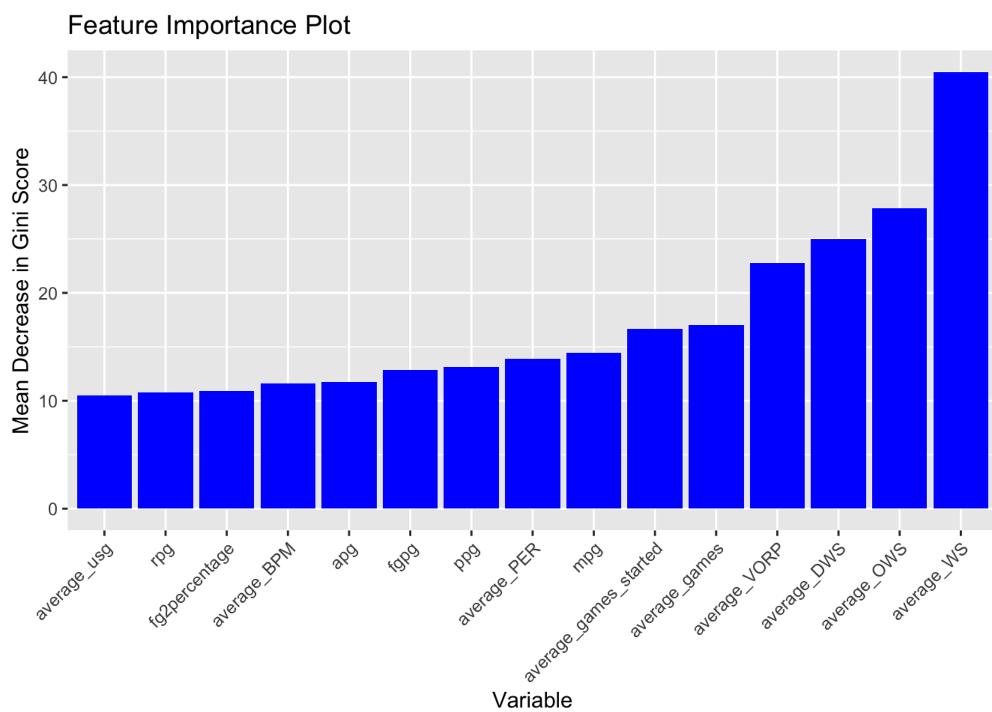
My model tries to follow the most basic guidelines of the project, and make the most out of the parameters given usi and logical approach. First, I went and followed the steps given pretty closely: I took the data we have on individual p combined them with the data on individual player awards, all over the span of 2007-2021. I followed the approach th data, or players, that we use as our 'training data', so as to give the model data to understand the relationships and t variables in order to predict future outcomes more precisely. I took players in the 2007-2015 draft classes. That way, careers worth of data on every player looked at, as our dataset only went back to 2007, and using players before tha very meaningful data points, resulting in less preciseness. I used the guidelines for the season outcome classificatior is only deemed to have a distinct career outcome if he attains that level or higher on 3 different campaigns. I factorec dataset with issuing a career outcome for each player from the 2007-2015 class as using averages of all of their first using four different records so as to use up less rows and cause less confusion for the computer and model. After tha to data and concocting the testing set of players from 2018-2021 with the same statistical variables (as we must use the model is predicting on the right things), it was time for me to pick, train, predict, and evaluate with my model. I kr variable, Career Outcome, is a categorical variable (non-numeric, one of few options, and if it is not the first five of six sixth), but also non-binary, meaning there is more than two options. This leaves little wiggle room into which type of r also where my lack of expert statistical knowledge comes into play. The only few options I had thought of were rando multinomial regressions, and I chose random forests as I had recently used this model to predict on another dataset learning it is quite accurate. Random forests are very complex in their inner-workings and how they come to the conc process behind them and simplified version is easy to understand. A random forest takes a lot of decision trees, or si takes a few variables and follows certain guidelines, made up of some of the different data points given and spliced a terms of patterns, order, etc. The random forest will look at the totality of the answers given from the decision trees, t the decision trees have to say and use the combined knowledge from them to give its output and predictions. This is trees create combinations of data that end up being more accurate in its predictions thah others; while the random fo it does not just listen to one decision tree, it listenes to what feels to be the grand consensus of feature importance, p accuracy from all of the decision trees.

My model does have flaws (starting with how Josh Giddey is only considered a roster outcome!) In all seriousness, tir constraints hindered me from displaying the most accurate, perfected model in terms of features, model type, and ot tune. First, I am only of certain level of knowledge of modeling, especially in the advanced multiclassification field. I a precise modeling that may have been very accurate, but I also felt confident that like mentioned earlier in the guidelir would work. One weakness is potentially my feature choice. First, I could quickly go back and change variables and rather deliver my thought process with the original idea I had, which is the model who's predictions you see. I though statistics and total award counts would suffice for a small-working model on short-time, as I did not want to pick anc point of view about what I believe to be important in basketball. I let the model choose that first. I should have, howe summary statistics and looking for possible correlation patterns, or more importantly issues with multicollinearity. I di offensive, defensive, and total win shares, among other stats like this, and removing them may be beneficial to just g over. I also could have possibly removed or tweaked other statistics that may have been slightly multicollinear or repe weights on certain statistics instead of using binary variables for some of the awards. It seems players with high-stat awards found success in career outcomes, while the model did not value some of the more important or telling award my last weakness I will mention in this report: team statistics. I did not include team statistics due to time constraints figure out a perfect way to include them. Joining the datasets by player and year would have been no issue, I just wis time to develop andfind something that could have helped offset high-volume and high-statistics players. For examp who have massive point/assist/rebound totals, possibly due to being the focal point of an offense, or the best player usage rate may not have helped all that much. Another note on this is that potentially a few more defensive statistics offense in these player statistics could have been helpful for players who do not score at will, and find their skills hidd Lastly, back to team statistics, having team stats in relation to a players would have been interesting, as well as team time a player played there. It would be difficult with the flow of player across teams, but if possible, the model could more contextual information regarding where a team was overall while a player was generating the stats seen in the r

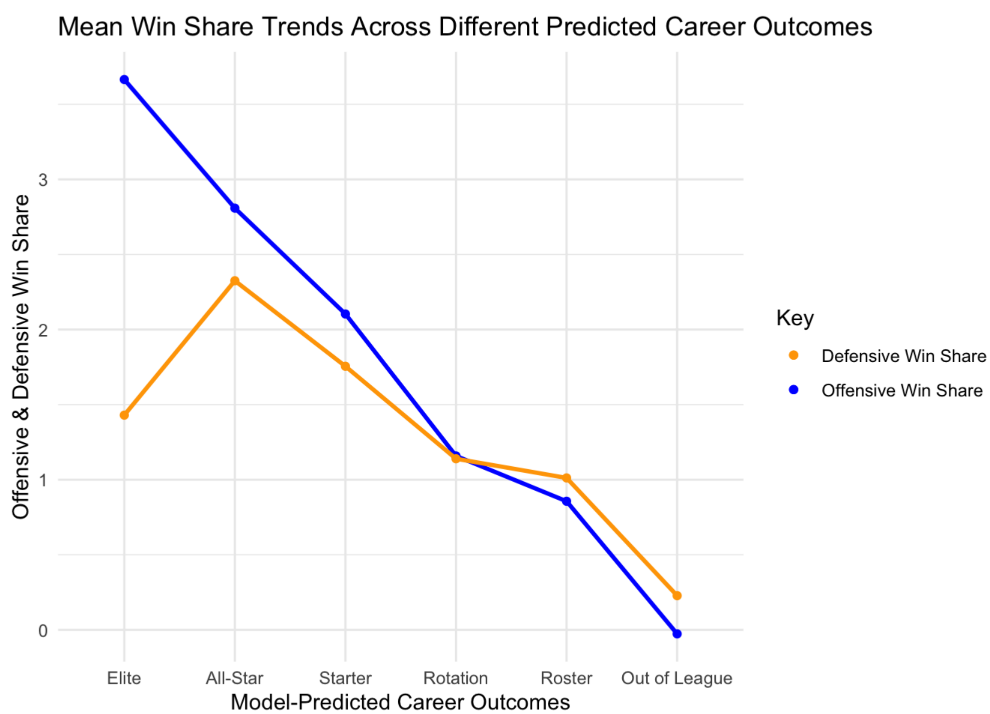
```
###VISUALS FOR MODELING QUESTION####


#Predicted Outcome Counts Graph#

desired_order <- c("Elite", "All-Star", "Starter", "Rotation", "Roster", "Out of League")

outcome_predictions$testing_predictions <- factor(outcome_predictions$testing_predictions, l
r)



custom_colors <- c("Elite" = "green",
                   "All-Star" = "yellow",
                   "Starter" = "orange",
                   "Rotation" = "violet",
                   "Roster" = "purple",
                   "Out of League" = "red")

ggplot(outcome_predictions, aes(x = testing_predictions)) +
  geom_bar(stat = "count", fill = custom_colors, color = 'white') +
  labs(title = "Predicted Outcome Counts for All 6 Classes",
       x = 'Predicted Career Outcomes', y = "Predicted Player Count")
```




```
###Importance of Variables in Random Forest Model###
variable_importance <- importance(rf_model)

variable_importance_df <- as.data.frame(variable_importance)

gini_threshold <- 10
important_vars <- variable_importance_df %>%
  filter(MeanDecreaseGini > gini_threshold)

ggplot(important_vars, aes(x = reorder(rownames(important_vars), MeanDecreaseGini), y = Mean
  geom_bar(stat = "identity", fill = "blue") +
  labs(title = "Feature Importance Plot",
       x = "Variable",
       y = "Mean Decrease in Gini Score") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



```
stats_vs_outcomes_dataset <- left_join(total_predictions, testing_dataset_aggregated, by = "
View(stats_vs_outcomes_dataset)

win_shares_by_outcome <- stats_vs_outcomes_dataset %>%
  group_by(`Model Predictions`) %>%
  summarize(Offensive_Win_Share_by_Outcome = mean(average_OWS),
            Defensive_Win_Share_by_Outcome = mean(average_DWS))
View(win_shares_by_outcome)


outcome_order <- c("Elite", "All-Star", "Starter", "Rotation", "Roster", "Out of League")

win_shares_by_outcome$`Model Predictions` <- factor(win_shares_by_outcome$`Model Predictions
rder)

ggplot(win_shares_by_outcome, aes(x = `Model Predictions`)) +
  geom_point(aes(y = Offensive_Win_Share_by_Outcome, color = "Offensive Win Share")) +
  geom_line(aes(y = Offensive_Win_Share_by_Outcome, group = 1), linewidth = 1, color = "blue
  geom_point(aes(y = Defensive_Win_Share_by_Outcome, color = "Defensive Win Share")) +
  geom_line(aes(y = Defensive_Win_Share_by_Outcome, group = 1), linewidth = 1, color = "oran
  labs(title = "Mean Win Share Trends Across Different Predicted Career Outcomes",
       x = "Model-Predicted Career Outcomes", y = "Offensive & Defensive Win Share") +
  scale_color_manual(name = "Key", values = c("Offensive Win Share" = "blue", "Defensive Win
+
  theme_minimal()
```




```
### PREDICTIONS FOR SGA, ZION, WISEMAN, AND GIDDEY###


players_of_interest <- c("Shai Gilgeous-Alexander", "Zion Williamson", "James Wiseman", "Jos

predictions_specific_players <- outcome_predictions %>%
  filter(player %in% players_of_interest)

predictions_specific_players
```

```
##                        player testing_predictions
## 159             James Wiseman       Out of League
## 201               Josh Giddey              Roster
## 337 Shai Gilgeous-Alexander             All-Star
## 395           Zion Williamson               Elite
```

```
#If for any reason, the View() does not work, in order of SGA, Zion, Wiseman, and Giddey, my
m as all-star, elite, out of league, and roster, respectively.
```

```
###BONUS: HTML TABLE WITH PREDICTIONS PLAYERS DRAFTED BETWEEN 2019-2021###

#I want to get all of the probabilities of each distinct outcome, as well as the predicted c
r in one dataset. Important for next step#

total_predictions <- left_join(outcome_predictions, probability_predictions, by = "player")



#Now, I want to go back and merge this dataset that has the player name, probability of each
ed outcome, with a previous dataset that also includes each players draft year#

##First, I'm going to make a copy of my master data set, filter for players drafted between
st bring the variables 'player' and 'draftyear'##

HTML_player_dataset <- model_data %>%
  filter(draftyear >= 2018 & draftyear <= 2021) %>%
  select(player, draftyear)


#Now I merge the two datasets on their player so all 2018-2021 players will have player name
ilities, and predictions. I will also group and filter so only one record shows per player,
draft year.#

HTML_table_predictions <- left_join(total_predictions, HTML_player_dataset, by = 'player') %
  filter(draftyear >=2019) %>%
  group_by(player, draftyear) %>%
  filter(row_number() == 1) %>%
  ungroup()

View(HTML_table_predictions)


#I will install and load in reactable now#

library(reactable)

reactable(HTML_table_predictions)
```

player	testing_predictions	All-Star	Elite	Out of League	Roster	Rotation
Aaron Henry	Out of League	0	0	0.748	0.23	0.006
Aaron Nesmith	Out of League	0	0	0.648	0.172	0.158
Aaron Wiggins	Rotation	0	0	0.23	0.282	0.36
Adam Mokoka	Out of League	0	0	0.964	0.028	0.008
Ade Murkey	Out of League	0	0	1	0	0
Admiral Schofield	Out of League	0	0	0.93	0.044	0.022
Ahmad Caver	Out of League	0	0	0.908	0.082	0.006
Aleem Ford	Out of League	0	0	0.746	0.166	0.034

Aleksej Pokusevski	Out of League	0	0	0.526	0.23	0.188
Alen Smailagic	Out of League	0	0	0.844	0.052	0.104

1–10 of 294 rows Previous 1 2 3

```
#I didn't like the way the columns were named and want to make the table look a little bit m
entation table. I had to search for how to work with reactable but I lelieve I got it to wor


colnames(HTML_table_predictions)[1:9] <- c("Player", "Model-Based Predictions", "Elite", "Al
ue", "Roster", "Rotation", "Starter", "Draft Year")

Final_HTML_table <- HTML_table_predictions[, c("Player", "Model-Based Predictions", "Elite",
League", "Roster", "Rotation", "Starter", "Draft Year")]
Final_HTML_table$Elite <- round(Final_HTML_table$Elite, 2)
Final_HTML_table$'All-Star' <- round(Final_HTML_table$'All-Star', 2)
Final_HTML_table$Starter <- round(Final_HTML_table$Starter, 2)
Final_HTML_table$Rotation <- round(Final_HTML_table$Rotation, 2)
Final_HTML_table$Roster <- round(Final_HTML_table$Roster, 2)
Final_HTML_table$'Out of League' <- round(Final_HTML_table$'Out of League', 2)

reactable(Final_HTML_table,
  striped = TRUE,
  bordered = TRUE,
  highlight = TRUE,
  searchable = TRUE,
)
```

Player	Model-Based Predictions	Elite	All-Star	Out of League	Roster	Rotation
Aaron Henry	Out of League	0	0	0.75	0.23	0.01
Aaron Nesmith	Out of League	0	0	0.65	0.17	0.16
Aaron Wiggins	Rotation	0	0	0.23	0.28	0.36
Adam Mokoka	Out of League	0	0	0.96	0.03	0.01
Ade Murkey	Out of League	0	0	1	0	0
Admiral Schofield	Out of League	0	0	0.93	0.04	0.02
Ahmad Caver	Out of League	0	0	0.91	0.08	0.01
Aleem Ford	Out of League	0	0	0.75	0.17	0.03
Aleksej Pokusevski	Out of League	0	0	0.53	0.23	0.19
Alen Smailagic	Out of League	0	0	0.84	0.05	0.1

1–10 of 294 rows Previous 1 2 3

Part 2 – Predicting Team Stats

In this section, we're going to introduce a simple way to predict team offensive rebound percent in the next game an
improve those predictions.

Question 1

Using the `rebounding_data` dataset, we'll predict a team's next game's offensive rebounding percent to be their av
percent in all prior games. On a single game level, offensive rebounding percent is the number of offensive rebounds
offensive rebound "chances" (essentially the team's missed shots). On a multi-game sample, it should be the total nu
divided by the total number of offensive rebound chances.

Please calculate what OKC's predicted offensive rebound percent is for game 81 in the data. That is, use games 1-8(

```
#In summation, I took the team rebounding data, and filtered it only to show OKC's statistic
d the summarize function to add upp all offesive rbounds and all chances to get an offensive
ames, and divided it to find OKC's offensive rebounding % in games 1-80, as well as its pred

thunder_rebounding <- rebounding_data %>%
  filter(team == 'OKC') %>%
  summarise(
    total_off_rebounds = sum(offensive_rebounds),
    total_off_rebounds_chances = sum(off_rebound_chances),
    game_81_oreb_precent = total_off_rebounds / total_off_rebounds_chances
  )

thunder_rebounding
```

```
## # A tibble: 1 × 3
##   total_off_rebounds total_off_rebounds_chances game_81_oreb_precent
##                <dbl>                      <dbl>                <dbl>
## 1               1226                       4237                0.289
```

ANSWER 1:

28.9%

Question 2

There are a few limitations to the method we used above. For example, if a team has a great offensive rebounder wh
this season but will be out due to an injury for the next game, we might reasonably predict a lower team offensive reb
game.

Please discuss how you would think about changing our original model to better account for missing players. You do
or implement any changes, and you can assume you have access to any reasonable data that isn't provided in this p
concise with your answer.

**ANSWER 2: While I believe that using team statistics can be very helpful to analyze trends and find general in
prowess of a team's offensive rebounding ability, especially over a longer sample size of a full season 82 gam
to predict a value for a single game if there are better measurables at hand. For example, within this situation
rebounder missing for the 81st game but being present in most of the team previous games that contributed t
rebounding prediction, using individual player statistics to help add in a rough estimate of each player's impo
offensive rebounding ability could help with the game-to-game variance in prediction. I was thinking it would
using each player's offensive rebounding statistics on a per game level. Such statistics could include total off
their contribution to total offensive rebounds when they are on the floor, the amount of minutes they play, and
spend within five feet of the basket on offense as a measure of how often they are looking to rebound, among
statistics could help build a model that analyzes these measures and determines each player's level of contrib
rebounding of the team.**

Question 3

In question 2, you saw and discussed how to deal with one weakness of the model. For this question, please write al
weaknesses of the simple average model you made in question 1 and discuss how you would deal with each of them
weakness and discuss how you'd fix that weakness, then move onto the next issue, or you can start by explaining m

original approach and discuss one overall modeling methodology you'd use that gets around most or all of them. Aga
any code or implement any changes, and you can assume you have access to any reasonable data that isn't provide
clear and concise with your answer.

ANSWER 3: Using averages of team statistics to predict the next game's value is difficult as their is little to no game-by-game changes to statistical output in an NBA season, as shown by the situation in Question 2 that m injured. It is also worth noting that who you are playing is very important in individual offensive rebounding sta have either more players with skills and physical attributes that gives them advantages on the glass, probably team defensive rebounding numbers. The opponent may also have a coach that emphasizes a play style that opportunities based on their defensive positioning, or a combination of both skill and play style. This greatly a team's offensive rebounding game-by-game, and controlling a possible regression or model to have values or defensive rebounding ability is imperative. This includes total defensive rebounding percentage and individual how valuable their individual players playing in the next game are to the teams overall defensive rebounding. most efficiently could be to find a league average of total defensive rebounding based on defensive reboundin and set that value equal to 1. Then if you are playing against a certain team and are aware of the availability o individual player statistics to predict a value of defensive rebounding percentage will give you a better look at defensive rebounds and percent should be, as compared to what you would have found if you just added toge games defensive rebounding numbers and chances and found an average. This new number, driven by indivic to what I thought would make sense to use in question 2 in regards to your own team's offensive rebounding compared to the league average. Dividing the number found by the league average tells you in a percentage h the opponent is than the league average. If the number is greater than 1, say 1.1, that means the opponent is would normally suspect, so you may adjust our average offensive rebounding percentage to reflect playing a one way to go about inputting how important different opponents are on a game-by-game level to the varianc and how to possibly correct this in our basic dataset so as to alter our team's predicted values as well. A seco we are predicting our values. There are many other ways to predict and evaluate certain values, and using sin variables in the context of an NBA game is not ideal. For example, take using a multivariate regression analys all of the data necessary to incorporate in our model. This could be individual player statistics and their sole v rebounding, both for our teams offensive rebounding and for opponents defensive rebounding. Adding certair game level such as home versus away, how many days rest prior to the game a team had, what is a team pac different shots do they take per game (three pointers, mid ranges, free throws, and layups/dunks all are not c of offensive rebounding probability). All of these factors are super important to have in a larger offensive rebo potentially use in a regression model that predicts a team offensive rebounding using all of these factors. The coefficients that essentially display how much value each factor plays into total offensive rebounding per gam this model allows you to plug in all the necessary values that change game by game to predict the next game precisely. If you know what team you are playing, where you are playing, if any specific player is unavailable o the prior rebounding statistics and shot statistics, the model will show a more accurate representation of offe for that game as compared to the method we used in the first question.