NBA Player Awards and Statistics

Tiger Teng

2023-10-19

Hello, this was a project for a previous job application, so I am afraid I cannot give you the data. This project is mainly about player awards and statistics. If you have any question, feel free to reach out to me at jteng@unc.edu or 919-260-0486

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

```
library(tidyverse)

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")

#take a look at summary of the data and clean the data set by removing
unknown data
awards <- awards %>%
  filter(!is.na(nbapersonid))
```

Part 1 – Awards

Question 1

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

```
# ppg as point per game
ppg <- player_data %>%
  group_by(nbapersonid, player, season, draftyear) %>%
# sum up the the rows for player who got traded during the season
  summarise(
    games = sum(games),
    points = sum(points)
  ) %>%
  mutate(points_per_game = points / games) %>%
ungroup()

# join data set and select useful columns
```

```
ppg_awards <- ppg%>%
  left_join(awards, by = c("nbapersonid","season")) %>%
  select(nbapersonid, player, points_per_game, "All NBA First Team", "All NBA
Second Team", "All NBA Third Team", "All NBA First Team", all_star_game)

# calculate for each category
avg_ppg_first_team <- ppg_awards %>%
  filter(`All NBA First Team` == 1) %>%
  summarise(average_points_per_game = mean(points_per_game))

avg_ppg_second_team <- ppg_awards %>%
  filter(`All NBA Second Team` == 1) %>%
  summarize(avg_points_per_game = mean(points_per_game))

avg_ppg_third_team <- ppg_awards %>%
  filter(`All NBA Third Team` == 1) %>%
  summarize(avg_points_per_game = mean(points_per_game))

avg_ppg_all_star <- ppg_awards %>%
  filter(all_star_game == TRUE) %>%
  summarize(avg_points_per_game = mean(points_per_game))

print(avg_ppg_first_team)
print(avg_ppg_second_team)
print(avg_ppg_third_team)
print(avg_ppg_all_star)
```

Question 2

QUESTION: What was the average number of years of experience in the league it takes for players to make their first All NBA Selection (1st, 2nd, or 3rd team)? Please limit your sample to players drafted in 2007 or later who did eventually go on to win at least one All NBA selection. For example:

- Luka Doncic is in the dataset as 2 years. He was drafted in 2018 and won his first All NBA award in 2019 (which was his second season).
- 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.

```
# create data set with needed columns
all_nba_selection <- player_data%>%
  left_join(awards, by = c("nbapersonid","season" )) %>%
  select(nbapersonid, draftyear, season, "All NBA First Team", "All NBA
Second Team", "All NBA Third Team") %>%
  filter( draftyear >= 2007) %>%
  filter(`All NBA First Team`+ `All NBA Second Team`+ `All NBA Third Team` >=
1)
```

```r
# calculate
average_years <- all_nba_selection %>%
  group_by(nbapersonid) %>%
  summarize(first_season = min(season), draftyear= first(draftyear))%>%
  summarize(avg_diff = mean(first_season + 1 - draftyear))

print(average_years)
```

Data Cleaning Interlude

Create a dataset with a "career outcome" for each player, representing the highest level of success that the player achieved for **at least two** seasons *after his first four seasons in the league* (examples to follow below!). To do this, you'll start with single season 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 season.
- 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 least 2000 minutes in the 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 meet any of the above criteria.
- 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 games per team. Assume that there were 66 possible games in the 2011 lockout season and 72 possible games in each of the 2019 and 2020 seasons that were shortened due to covid. Specifically, if a player played 900 minutes in 2011, he **would** meet the rotation criteria because his final minutes would be considered to be 900 * (82/66) = 1118. Please use this math for both minutes and games started, so a player who started 38 games in 2019 or 2020 would be considered to have started 38 * (82/72) = 43 games, and thus would qualify for starting 41. Any answers should be calculated 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, since Shai Gilgeous-Alexander was both All-NBA 1st team and an All-Star last year, he would be considered to be "Elite" for the 2022 season, but would still qualify 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 teams. Note this is a 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), Roster (4), Roster (5), Out of the League (6+) would be considered "Out of the League," because after his first four seasons, he only has a single Roster year, which 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), Starter (4), Starter (5), Starter (6), All-Star (7), Elite (8), Starter (9) would be considered "All-Star," because he had at least two seasons after his first four at all-star level of production or higher.

- A player who enters the league as a rookie and has season outcomes of Roster (1), Rotation (2), Starter (3), Starter (4), Starter (5), Starter (6), Rotation (7), Rotation (8), Roster (9) would be considered a "Starter" because he has two seasons after his first four at a starter level of production.

Question 3

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

```
# merge data
player_data_awards <- player_data%>%
  left_join(awards, by = c("nbapersonid","season" )) %>%
  select(nbapersonid, player, draftyear, season,games, games_start,mins, "All
NBA First Team", "All NBA Second Team", "All NBA Third Team", all_star_game,
"Defensive Player Of The Year_rk", "Most Valuable Player_rk")

# make MVP and DROY binary
player_data_awards$MVP <- ifelse(player_data_awards$"Most Valuable Player_rk"
== 1, 1, 0)
player_data_awards$DPOY <- ifelse(player_data_awards$"Defensive Player Of The
Year_rk" == 1, 1, 0)
player_data_awards <- player_data_awards %>%
  mutate(All_star = as.integer(all_star_game))%>%
  group_by(nbapersonid, player, season, draftyear) %>%
#sum up the the rows for player who got traded during the season, as it shows
two rows for one season
  summarise(
    games = sum(games),
    mins = sum(mins),
    games_start = sum(games_start),
    `All NBA First Team` = max(`All NBA First Team`),
    `All NBA Second Team` = max(`All NBA Second Team`),
    `All NBA Third Team` = max(`All NBA Third Team`),
```

```
    All_star = max(All_star),
    DPOY = max(DPOY),
    MVP = max(MVP)
  ) %>%
  ungroup()

# adjust for 2011, 2019, 2020
player_data_awards <- player_data_awards%>%
  group_by(nbapersonid, player, season, draftyear,`All NBA First Team`, `All
NBA Second Team`, `All NBA Third Team`, DPOY, MVP, All_star) %>%
  summarise(
    games = round(ifelse(season == 2011, games * 82/66,
                         ifelse(season == 2019, games * 82/72,
                                ifelse(season == 2020, games * 82/72,
games)))),
    mins = round(ifelse(season == 2011, mins * 82/66,
                        ifelse(season == 2019, mins * 82/72,
                               ifelse(season == 2020, mins * 82/72, mins)))),
    games_start = round(ifelse(season == 2011, games_start* 82/66,
                         ifelse(season == 2019, games_start * 82/72,
                                ifelse(season == 2020, games_start * 82/72,
games)))),
    across(.cols = starts_with("other"), .fns = first)
  ) %>%
  ungroup()

# Create a new column 'year_played' representing the years played since draft
year
player_data_awards <- player_data_awards %>%
  mutate(year_played = season + 1 - draftyear)

# Filter rows where 'year_played'> 4
filtered_data <- player_data_awards %>%
  filter(year_played > 4)


# Replace NA with 0 and Define the conditions for each binary variable
filtered_data <- filtered_data %>%
  mutate_all(~replace_na(., 0))%>%
  mutate(
    Elite = ifelse(`All NBA First Team` + `All NBA Second Team` + `All NBA
Third Team` + MVP + DPOY >= 1, 1, 0),
    All_Star = ifelse((All_star == 1 | Elite == 1), 1, 0),
    Starter = ifelse((games_start >= 41 | mins >= 2000 | All_Star == 1), 1,
0),
    Rotation = ifelse((mins >= 1000 | Starter == 1), 1, 0),
    Roster = ifelse((mins > 0 | Rotation == 1), 1, 0),
    Out_of_the_League = ifelse(mins == 0, 1, 0)
  )
```

```
# Define single season outcome
filtered_data <- filtered_data %>%
  mutate(
    single_season_outcome = ifelse(Elite >= 1, "Elite",
                        ifelse(All_Star >= 1, "All-Star",
                               ifelse(Starter >= 1, "Starter",
                                      ifelse(Rotation >= 1, "Rotation",
                                             ifelse(Roster >= 1, "Roster",
                                                    ifelse(Elite == 1 &&
All_Star == 1 && Starter == 1&& Rotation == 1 && Roster == 1, "Out of the
League", "Others")))))))


# Summarize career outcomes for each player
career_outcomes_summary <- filtered_data %>%
  group_by(nbapersonid, draftyear) %>%
  summarise(
    Elite = sum(Elite),
    All_Star = sum(All_Star),
    Starter = sum(Starter),
    Rotation = sum(Rotation),
    Roster = sum(Roster),
    Out_of_the_League = sum(Out_of_the_League)
  ) %>%
  ungroup() %>%
  mutate(
    Career_Outcome = ifelse(Elite >= 2, "Elite",
                        ifelse(All_Star >= 2, "All-Star",
                               ifelse(Starter >= 2, "Starter",
                                      ifelse(Rotation >= 2, "Rotation",
                                             ifelse(Roster >= 2, "Roster",
                                                    ifelse(Elite <= 1 &&
All_Star <= 1 && Starter <= 1 && Rotation <= 1 && Roster <= 1, "Out of the
League", "Others")))))))
  )

career_outcomes_counts <- career_outcomes_summary %>%
  filter(draftyear == 2010)%>%
  group_by(Career_Outcome) %>%
  summarise(Player_Count = n())

print(career_outcomes_counts)
```

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 and then discuss ways to 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 average offensive rebounding percent in all prior games. On a single game level, offensive rebounding percent is the number of offensive rebounds divided by their number offensive rebound "chances" (essentially the team's missed shots). On a multi-game sample, it should be the total number of offensive rebounds 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 1-80 to predict game 81.

```
#filter first 80 OKC games
OKC_OREB_80 <- rebounding_data %>%
  filter(team == "OKC" & game_number <= 80)


# Calculate total number of offensive rebounds and offensive rebound chances
for OKC games 1-80
total_OREB <- sum(OKC_OREB_80$offensive_rebounds)
total_OREB_chances <- sum(OKC_OREB_80$off_rebound_chances)

# Calculate the average offensive rebound percentage for OKC games 1-80
avg_OREB_pct <- total_OREB / total_OREB_chances


# Use the calculated average offensive rebound percentage as the prediction
for OKC game 81
predicted_OREB_pct_game_81 <- avg_OREB_pct

print(predicted_OREB_pct_game_81)
```

Question 2

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

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

ANSWER:

A missing player can affect many aspects. The rotation will change, as other players need to fill out his time. The ability in getting OREB will change. I think we could use a factor to

proportionally adjust the offensive rebounding contributions based on the redistributed playing time. We will use adjusted_OREB_pct to predict OREB pct for game #81.

$$\text{avg\_OREB\_pct} = \frac{\text{total\_OREB}}{\text{total\_OREB\_chances}}$$

$$\text{adjusted\_OREB\_pct} = \frac{\text{total\_OREB - total\_OREB\_injured \_player}}{\text{total\_OREB\_chances}} * factor$$

$$\text{factor} = \frac{\text{total\_mins}}{\text{total\_mins - total\_mins\_injured\_player}}$$

Factor accounts for the fact that the mins for injured player would be 0 in game #81, as his time will be fill out by the rest of the team If we could a clearer picture about the rotation in next game, we could break down the formula by using average OREB per mins for each players times their expected mins in game#81. In this case, the expected mins for injured players would be 0. The numerator will be the sum of average OREB per mins * expected mins for each player. The denominator will be the average OREB chances per game. Expected mins for each player would be the expected rotation for game #81.

Question 3

In question 2, you saw and discussed how to deal with one weakness of the model. For this question, please write about 1-3 other potential weaknesses of the simple average model you made in question 1 and discuss how you would deal with each of them. You may either explain a weakness and discuss how you'd fix that weakness, then move onto the next issue, or you can start by explaining multiple weaknesses with the original approach and discuss one overall modeling methodology you'd use that gets around most or all of them. Again, you do not need to write any code or implement any changes, and you can assume you have access to any reasonable data that isn't provided in this project. Try to be clear and concise with your answer.

ANSWER:

1. Limited variables. Using simple average model ignores the game context. OREB% can be influenced by the change of denominator which is OREB chances. This is sum of our OREB and opponents' DREB. Meanwhile, the chances are also generated by missing shots, so it has to do with shots we take and miss. If the game has faster pace or we have poor FG%, OREB chances would go up. So, we could add more variables that reflect on the game context like our offense stats, opponents defense stats, and possessions.

2. Not taking opponent into consideration. Simple average model ignores the specific opponent of the game we want to predict. Different opponents could have differences in ability of rebounding, defense, and team pace (possessions per game). Using simple average model ignores the fact that previous games with the same opponent have more relevance with our prediction. To address this weakness, we

could assign higher weights to games with the same opponent as game #81 in our sample.

3. Team undergone changes during the season. The changes could be in player rotation, chemistry, coaching styles, and even the player's performance. The performance at the beginning of the season might be very different than the performance at the end of the season. So, more recent games would have more indication on our prediction for game #81. To address this issue, we could have a time-weighted average approach by assigning higher weights to more recent games, reflecting their greater relevance in predicting the team's current offensive rebounding performance.

4. The above weaknesses align with our common sense, but it will be very important to analyze whether these trends indeed exist. This could be done by testing the original model with the modified model and compared their prediction accuracy.