awards_project_template

September 17, 2023

1 Name: DUONG THUY LE

1.0.1 Date: 2023/08/29

2 Introduction

The purpose of this project is to gauge your technical skills and problem solving ability by working through something similar to a real NBA data science project. You will work your way through this jupyter notebook, answering questions as you go along. Please begin by adding your name to the top markdown chunk in this document. When you're finished with the document, come back and type your answers into the answer key at the top. Please leave all your work below and have your answers where indicated below as well. Please note that we will be reviewing your code 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. Please try to use packages like pandas/numpy and matplotlib/seaborn as much as possible (instead of base python data manipulations and explicit loops.)

WARNING: Your project will **ONLY** be graded if it's knit to an HTML document where we can see your code. Be careful to make sure that any long lines of code appropriately visibly wrap around visibly to the next line, as code that's cut off from the side of the document cannot be graded.

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).

3 Answers

3.1 Part 1

Question 1:

- 1st Team: XX.X points per game

- 2nd Team: XX.X points per game

- 3rd Team: XX.X points per game

- All-Star: XX.X points per game

Question 2: XX.X Years

Question 3:

- Elite: X players.

- All-Star: X players.

- Starter: X players.

- Rotation: X players.

- Roster: X players.

- Out of League: X players.

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

3.2 Part 2

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

4 Setup and Data

[1]:
```
import os
import numpy as np
import pandas as pd
import math
import matplotlib.pylab as plt
import seaborn as sns
```

[2]:
```
# Note you will likely have to change these paths.
# If your data is in the same folder as this project,
# the paths will likely be fixed for you by deleting ../../Data/awards_project/␣
↪from each string.
awards = pd.read_csv("./awards_data.csv")
player_data = pd.read_csv("./player_stats.csv")
team_data = pd.read_csv("./team_stats.csv")
rebounding_data = pd.read_csv("./team_rebounding_data_22.csv")
```

[3]:
```
awards.head(5)
```

[3]:

```
   season  nbapersonid  All NBA Defensive First Team  \
0    2007        708.0                           1.0
1    2007        947.0                           0.0
2    2007        948.0                           1.0
3    2007        959.0                           0.0
4    2007        977.0                           1.0

   All NBA Defensive Second Team  All NBA First Team  All NBA Second Team  \
0                            0.0                 1.0                  0.0
1                            0.0                 0.0                  0.0
2                            0.0                 0.0                  0.0
3                            0.0                 0.0                  1.0
4                            0.0                 1.0                  0.0

   All NBA Third Team  All Rookie First Team  All Rookie Second Team  \
0                 0.0                    0.0                     0.0
1                 0.0                    0.0                     0.0
2                 0.0                    0.0                     0.0
3                 0.0                    0.0                     0.0
4                 0.0                    0.0                     0.0

   Bill Russell NBA Finals MVP  …  all_star_game  rookie_all_star_game  \
0                          0.0  …           True                 False
1                          0.0  …           True                 False
2                          0.0  …            NaN                   NaN
3                          0.0  …           True                 False
4                          0.0  …           True                 False

   allstar_rk Defensive Player Of The Year_rk Most Improved Player_rk  \
0         1.0                             1.0                     NaN
1         2.0                             NaN                     NaN
2         3.0                             2.0                     NaN
3         4.0                             NaN                     NaN
4         1.0                             5.0                     NaN

   Most Valuable Player_rk  Rookie Of The Year_rk  Sixth Man Of The Year_rk  \
0                      3.0                    NaN                       NaN
1                      NaN                    NaN                       NaN
2                      NaN                    NaN                       NaN
3                      9.0                    NaN                       NaN
4                      1.0                    NaN                       NaN

   all_nba_points_rk  all_rookie_points_rk
0                NaN                   NaN
1                NaN                   NaN
2                NaN                   NaN
3                NaN                   NaN
```

```
4                NaN                     NaN

[5 rows x 23 columns]
```

[4]:
```
awards.describe()
```

[4]:
```
            season   nbapersonid  All NBA Defensive First Team  \
count  4329.000000  4.321000e+03                    693.000000
mean   2016.687688  1.622733e+06                      0.109668
std       3.781453  4.221668e+07                      0.312701
min    2007.000000  2.550000e+02                      0.000000
25%    2015.000000  2.015650e+05                      0.000000
50%    2018.000000  2.034710e+05                      0.000000
75%    2020.000000  1.627885e+06                      0.000000
max    2021.000000  1.962937e+09                      1.000000

       All NBA Defensive Second Team  All NBA First Team  All NBA Second Team  \
count                     693.000000          693.000000           693.000000
mean                        0.108225            0.108225             0.108225
std                         0.310889            0.310889             0.310889
min                         0.000000            0.000000             0.000000
25%                         0.000000            0.000000             0.000000
50%                         0.000000            0.000000             0.000000
75%                         0.000000            0.000000             0.000000
max                         1.000000            1.000000             1.000000

       All NBA Third Team  All Rookie First Team  All Rookie Second Team  \
count          693.000000             693.000000              693.000000
mean             0.108225               0.111111                0.109668
std              0.310889               0.314497                0.312701
min              0.000000               0.000000                0.000000
25%              0.000000               0.000000                0.000000
50%              0.000000               0.000000                0.000000
75%              0.000000               0.000000                0.000000
max              1.000000               1.000000                1.000000

       Bill Russell NBA Finals MVP  …  Player Of The Week  \
count                   693.000000  …          693.000000
mean                      0.021645  …            0.940837
std                       0.145627  …            1.175727
min                       0.000000  …            0.000000
25%                       0.000000  …            0.000000
50%                       0.000000  …            1.000000
75%                       0.000000  …            1.000000
max                       1.000000  …            7.000000

       Rookie Of The Month   allstar_rk  Defensive Player Of The Year_rk  \
```

```
count         693.000000  3691.000000                        255.000000
mean            0.233766    58.173124                          9.258824
std             0.790231    40.466750                          5.409571
min             0.000000     1.000000                          1.000000
25%             0.000000    20.000000                          5.000000
50%             0.000000    56.000000                          9.000000
75%             0.000000    92.000000                         13.000000
max             6.000000   157.000000                         25.000000

       Most Improved Player_rk  Most Valuable Player_rk  \
count               400.000000               202.000000
mean                 13.540000                 7.207921
std                   7.675329                 3.915315
min                   1.000000                 1.000000
25%                   7.000000                 4.000000
50%                  13.000000                 7.000000
75%                  20.000000                10.000000
max                  30.000000                17.000000

       Rookie Of The Year_rk  Sixth Man Of The Year_rk  all_nba_points_rk  \
count             123.000000                237.000000         394.000000
mean                4.853659                  8.177215          18.390863
std                 2.804221                  4.468608          10.581058
min                 1.000000                  1.000000           1.000000
25%                 3.000000                  4.000000           9.000000
50%                 5.000000                  8.000000          18.000000
75%                 7.000000                 11.000000          27.000000
max                13.000000                 18.000000          41.000000

       all_rookie_points_rk
count            266.000000
mean              12.409774
std                7.031019
min                1.000000
25%                6.250000
50%               13.000000
75%               18.000000
max               26.000000

[8 rows x 21 columns]
```

[5]:
```
player_data.head(5)
```

[5]:
```
   nbapersonid           player  draftyear  draftpick  season   nbateamid  \
0         2585    Zaza Pachulia       2003       42.0    2007  1610612737
1       200780    Solomon Jones       2006       33.0    2007  1610612737
2         2746       Josh Smith       2004       17.0    2007  1610612737
```

```
3       201151           Acie Law       2007        11.0     2007   1610612737
4       101136   Salim Stoudamire       2005        31.0     2007   1610612737

  team  games  games_start  mins  …  blk_pct  tov_pct    usg  OWS  DWS   WS  \
0  ATL     62            5   944  …    0.010    0.181  0.183  0.2  0.9  1.1
1  ATL     35            0   145  …    0.026    0.221  0.156 -0.1  0.1  0.0
2  ATL     81           81  2873  …    0.059    0.155  0.250  1.2  4.6  5.8
3  ATL     56            6   865  …    0.000    0.178  0.165 -0.5  0.4 -0.1
4  ATL     35            0   402  …    0.009    0.094  0.252  0.1  0.1  0.3

   OBPM  DBPM  BPM  VORP
0  -3.9  -1.3 -5.1  -0.7
1  -6.7  -2.0 -8.8  -0.2
2   0.5   2.5  3.0   3.7
3  -4.2  -1.0 -5.2  -0.7
4  -1.0  -2.5 -3.5  -0.1

[5 rows x 49 columns]
```

[6]:
```
team_data.head(5)
```

[6]:
```
    nbateamid team  season  games  off_rtg  def_rtg  net_rtg   W   L
0  1610612737  ATL    2007     82    106.9    108.9     -2.0  37  45
1  1610612751  BKN    2007     82    104.0    109.4     -5.4  34  48
2  1610612738  BOS    2007     82    110.2     98.9     11.3  66  16
3  1610612766  CHA    2007     82    104.6    109.4     -4.8  32  50
4  1610612741  CHI    2007     82    103.9    107.2     -3.3  33  49
```

[7]:
```
rebounding_data.head(5)
```

[7]:
```
  team opp_team    gamedate  game_number  offensive_rebounds  \
0  BOS      PHI  2022-10-18            1                  10
1  PHI      BOS  2022-10-18            1                   8
2  GSW      LAL  2022-10-18            1                  16
3  LAL      GSW  2022-10-18            1                  14
4  ORL      DET  2022-10-19            1                  13

   off_rebound_chances  oreb_pct
0                   39  0.256410
1                   42  0.190476
2                   57  0.280702
3                   57  0.245614
4                   47  0.276596
```

4.1 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 manipulation questions and work towards building a model to predict broad levels of career success.

4.1.1 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)?

[8]:
```
# Filter players selected to All-NBA teams and All-Star Game
all_nba_players = awards[(awards["All NBA First Team"] == 1) |
                         (awards["All NBA Second Team"] == 1) |
                         (awards["All NBA Third Team"] == 1)]
all_star_players = awards[awards["all_star_game"] == True]

# Merge player data with awards data to get player statistics
merged_data = pd.merge(all_nba_players, player_data, on=["season",␣
 ↪"nbapersonid"])
merged_data_all_star = pd.merge(all_star_players, player_data, on=["season",␣
 ↪"nbapersonid"])
```

[9]:
```
merged_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 226 entries, 0 to 225
Data columns (total 70 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   season                          226 non-null    int64
 1   nbapersonid                     226 non-null    float64
 2   All NBA Defensive First Team    226 non-null    float64
 3   All NBA Defensive Second Team   226 non-null    float64
 4   All NBA First Team              226 non-null    float64
 5   All NBA Second Team             226 non-null    float64
 6   All NBA Third Team              226 non-null    float64
 7   All Rookie First Team           226 non-null    float64
 8   All Rookie Second Team          226 non-null    float64
 9   Bill Russell NBA Finals MVP     226 non-null    float64
 10  Player Of The Month             226 non-null    float64
 11  Player Of The Week              226 non-null    float64
 12  Rookie Of The Month             226 non-null    float64
 13  all_star_game                   208 non-null    object
 14  rookie_all_star_game            208 non-null    object
 15  allstar_rk                      216 non-null    float64
 16  Defensive Player Of The Year_rk 79 non-null     float64
 17  Most Improved Player_rk         54 non-null     float64
 18  Most Valuable Player_rk         175 non-null    float64
 19  Rookie Of The Year_rk           0 non-null      float64
 20  Sixth Man Of The Year_rk        1 non-null      float64
```

```
21  all_nba_points_rk        162 non-null    float64
22  all_rookie_points_rk     0 non-null      float64
23  player                   226 non-null    object
24  draftyear                226 non-null    int64
25  draftpick                226 non-null    float64
26  nbateamid                226 non-null    int64
27  team                     226 non-null    object
28  games                    226 non-null    int64
29  games_start              226 non-null    int64
30  mins                     226 non-null    int64
31  fgm                      226 non-null    int64
32  fga                      226 non-null    int64
33  fgp                      226 non-null    float64
34  fgm3                     226 non-null    int64
35  fga3                     226 non-null    int64
36  fgp3                     224 non-null    float64
37  fgm2                     226 non-null    int64
38  fga2                     226 non-null    int64
39  fgp2                     226 non-null    float64
40  efg                      226 non-null    float64
41  ftm                      226 non-null    int64
42  fta                      226 non-null    int64
43  ftp                      226 non-null    float64
44  off_reb                  226 non-null    int64
45  def_reb                  226 non-null    int64
46  tot_reb                  226 non-null    int64
47  ast                      226 non-null    int64
48  steals                   226 non-null    int64
49  blocks                   226 non-null    int64
50  tov                      226 non-null    int64
51  tot_fouls                226 non-null    int64
52  points                   226 non-null    int64
53  PER                      226 non-null    float64
54  FTr                      226 non-null    float64
55  off_reb_pct              226 non-null    float64
56  def_reb_pct              226 non-null    float64
57  tot_reb_pct              226 non-null    float64
58  ast_pct                  226 non-null    float64
59  stl_pct                  226 non-null    float64
60  blk_pct                  226 non-null    float64
61  tov_pct                  226 non-null    float64
62  usg                      226 non-null    float64
63  OWS                      226 non-null    float64
64  DWS                      226 non-null    float64
65  WS                       226 non-null    float64
66  OBPM                     226 non-null    float64
67  DBPM                     226 non-null    float64
68  BPM                      226 non-null    float64
```

```
 69  VORP                             226 non-null    float64
dtypes: float64(43), int64(23), object(4)
memory usage: 125.4+ KB
```

[10]:
```
merged_data_all_star.head()
```

[10]:
```
   season  nbapersonid  All NBA Defensive First Team  \
0    2007        708.0                          1.0
1    2007        947.0                          0.0
2    2007        959.0                          0.0
3    2007        977.0                          1.0
4    2007       1495.0                          1.0

   All NBA Defensive Second Team  All NBA First Team  All NBA Second Team  \
0                            0.0                 1.0                  0.0
1                            0.0                 0.0                  0.0
2                            0.0                 0.0                  1.0
3                            0.0                 1.0                  0.0
4                            0.0                 0.0                  1.0

   All NBA Third Team  All Rookie First Team  All Rookie Second Team  \
0                 0.0                    0.0                     0.0
1                 0.0                    0.0                     0.0
2                 0.0                    0.0                     0.0
3                 0.0                    0.0                     0.0
4                 0.0                    0.0                     0.0

   Bill Russell NBA Finals MVP  …  blk_pct  tov_pct    usg  OWS  DWS    WS  \
0                          0.0  …    0.031    0.108  0.255  6.6  6.2  12.9
1                          0.0  …    0.002    0.114  0.267  8.9  2.8  11.6
2                          0.0  …    0.001    0.216  0.220  9.0  1.4  10.5
3                          0.0  …    0.009    0.113  0.314  9.5  4.3  13.8
4                          0.0  …    0.043    0.114  0.282  4.9  6.2  11.1

   OBPM  DBPM  BPM  VORP
0   4.7   3.5  8.2   6.0
1   3.4  -0.7  2.7   4.0
2   5.8  -1.8  3.9   4.2
3   5.2   0.6  5.8   6.3
4   3.0   2.2  5.2   4.8

[5 rows x 70 columns]
```

[11]:
```
# Calculate average points per game for 1st Team All-NBA players
average_points_1st_team = round(merged_data[merged_data["All NBA First Team"]␣
 ↪== 1]["points"].mean(),1)
average_points_1st_team
```

[11]: 1897.4

[12]:
```python
# Calculate average points per game for 2nd Team All-NBA players
average_points_2nd_team = round(merged_data[merged_data["All NBA Second Team"]␣
↪== 1]["points"].mean(),1)
average_points_2nd_team
```

[12]: 1646.9

[13]:
```python
# Calculate average points per game for 3rd Team All-NBA players
average_points_3rd_team = round(merged_data[merged_data["All NBA Third Team"]␣
↪== 1]["points"].mean(),1)
average_points_3rd_team
```

[13]: 1432.1

[14]:
```python
# Calculate average points per game for Team All-NBA players
average_points_all_star_team = round(merged_data[merged_data["all_star_game"]␣
↪== 1]["points"].mean(),1)
average_points_all_star_team
```

[14]: 1685.2

ANSWER 1:

1st Team: 1898.4 points per game
2nd Team: 1646.9 points per game
3rd Team: 1432.1 points per game
All-Star: 1685.2 points per game

4.1.2 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.

[15]:
```python
# Filter players drafted in 2007 or later who won at least one All-NBA selection
eligible_players = awards[(awards["season"] >= 2007) &
                          ((awards["All NBA First Team"] == 1) |
                           (awards["All NBA Second Team"] == 1) |
                           (awards["All NBA Third Team"] == 1))]
```

```
# Merge player data with awards data to get player statistics
merged_data_2 = pd.merge(eligible_players, player_data, on=["season",␣
↪"nbapersonid"])

# Calculate the difference between the season of the first All-NBA selection␣
↪and the draft year
merged_data_2["years_to_first_all_nba"] = merged_data_2["season"] -␣
↪merged_data_2["draftyear"] + 1
```

[16]:
```
merged_data_2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 226 entries, 0 to 225
Data columns (total 71 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   season                          226 non-null    int64
 1   nbapersonid                     226 non-null    float64
 2   All NBA Defensive First Team    226 non-null    float64
 3   All NBA Defensive Second Team   226 non-null    float64
 4   All NBA First Team              226 non-null    float64
 5   All NBA Second Team             226 non-null    float64
 6   All NBA Third Team              226 non-null    float64
 7   All Rookie First Team           226 non-null    float64
 8   All Rookie Second Team          226 non-null    float64
 9   Bill Russell NBA Finals MVP     226 non-null    float64
 10  Player Of The Month             226 non-null    float64
 11  Player Of The Week              226 non-null    float64
 12  Rookie Of The Month             226 non-null    float64
 13  all_star_game                   208 non-null    object
 14  rookie_all_star_game            208 non-null    object
 15  allstar_rk                      216 non-null    float64
 16  Defensive Player Of The Year_rk 79 non-null     float64
 17  Most Improved Player_rk         54 non-null     float64
 18  Most Valuable Player_rk         175 non-null    float64
 19  Rookie Of The Year_rk           0 non-null      float64
 20  Sixth Man Of The Year_rk        1 non-null      float64
 21  all_nba_points_rk               162 non-null    float64
 22  all_rookie_points_rk            0 non-null      float64
 23  player                          226 non-null    object
 24  draftyear                       226 non-null    int64
 25  draftpick                       226 non-null    float64
 26  nbateamid                       226 non-null    int64
 27  team                            226 non-null    object
 28  games                           226 non-null    int64
 29  games_start                     226 non-null    int64
 30  mins                            226 non-null    int64
```

```
 31  fgm                          226 non-null    int64
 32  fga                          226 non-null    int64
 33  fgp                          226 non-null    float64
 34  fgm3                         226 non-null    int64
 35  fga3                         226 non-null    int64
 36  fgp3                         224 non-null    float64
 37  fgm2                         226 non-null    int64
 38  fga2                         226 non-null    int64
 39  fgp2                         226 non-null    float64
 40  efg                          226 non-null    float64
 41  ftm                          226 non-null    int64
 42  fta                          226 non-null    int64
 43  ftp                          226 non-null    float64
 44  off_reb                      226 non-null    int64
 45  def_reb                      226 non-null    int64
 46  tot_reb                      226 non-null    int64
 47  ast                          226 non-null    int64
 48  steals                       226 non-null    int64
 49  blocks                       226 non-null    int64
 50  tov                          226 non-null    int64
 51  tot_fouls                    226 non-null    int64
 52  points                       226 non-null    int64
 53  PER                          226 non-null    float64
 54  FTr                          226 non-null    float64
 55  off_reb_pct                  226 non-null    float64
 56  def_reb_pct                  226 non-null    float64
 57  tot_reb_pct                  226 non-null    float64
 58  ast_pct                      226 non-null    float64
 59  stl_pct                      226 non-null    float64
 60  blk_pct                      226 non-null    float64
 61  tov_pct                      226 non-null    float64
 62  usg                          226 non-null    float64
 63  OWS                          226 non-null    float64
 64  DWS                          226 non-null    float64
 65  WS                           226 non-null    float64
 66  OBPM                         226 non-null    float64
 67  DBPM                         226 non-null    float64
 68  BPM                          226 non-null    float64
 69  VORP                         226 non-null    float64
 70  years_to_first_all_nba       226 non-null    int64
dtypes: float64(43), int64(24), object(4)
memory usage: 127.1+ KB
```

[17]:
```python
# Calculate the average number of years to first All-NBA selection
average_years_to_first_all_nba = round(merged_data_2["years_to_first_all_nba"].
 ↪mean(),1)
average_years_to_first_all_nba
```

[17]: 8.2

ANSWER 2:

8.2 Years

4.2 Data Cleaning Interlude

You're going to work to 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.

4.2.1 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?

[18]:
```
# Combine relevant columns from different datasets
combined_data = player_data.merge(awards, on=["season", "nbapersonid"],␣
↪how="left")
combined_data.head(5)
```

[18]:
```
   nbapersonid           player  draftyear  draftpick  season   nbateamid  \
0         2585     Zaza Pachulia       2003       42.0    2007  1610612737
1       200780    Solomon Jones       2006       33.0    2007  1610612737
2         2746       Josh Smith       2004       17.0    2007  1610612737
3       201151         Acie Law       2007       11.0    2007  1610612737
4       101136  Salim Stoudamire       2005       31.0    2007  1610612737

  team  games  games_start  mins  …  all_star_game  rookie_all_star_game  \
0  ATL     62            5   944  …            NaN                   NaN
1  ATL     35            0   145  …            NaN                   NaN
2  ATL     81           81  2873  …            NaN                   NaN
3  ATL     56            6   865  …            NaN                   NaN
4  ATL     35            0   402  …            NaN                   NaN

   allstar_rk  Defensive Player Of The Year_rk  Most Improved Player_rk  \
0         NaN                              NaN                      NaN
1         NaN                              NaN                      NaN
2         NaN                              6.0                      NaN
3         NaN                              NaN                      NaN
4         NaN                              NaN                      NaN

   Most Valuable Player_rk  Rookie Of The Year_rk  Sixth Man Of The Year_rk  \
0                      NaN                    NaN                       NaN
1                      NaN                    NaN                       NaN
2                      NaN                    NaN                       NaN
3                      NaN                    NaN                       NaN
```

```
4                       NaN                    NaN                         NaN

   all_nba_points_rk  all_rookie_points_rk
0                NaN                   NaN
1                NaN                   NaN
2                NaN                   NaN
3                NaN                   NaN
4                NaN                   NaN

[5 rows x 70 columns]
```

[19]:
```
combined_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 8498 entries, 0 to 8497
Data columns (total 70 columns):
 #   Column                           Non-Null Count  Dtype
---  ------                           --------------  -----
 0   nbapersonid                      8498 non-null   int64
 1   player                           8498 non-null   object
 2   draftyear                        8498 non-null   int64
 3   draftpick                        6730 non-null   float64
 4   season                           8498 non-null   int64
 5   nbateamid                        8498 non-null   int64
 6   team                             8498 non-null   object
 7   games                            8498 non-null   int64
 8   games_start                      8498 non-null   int64
 9   mins                             8498 non-null   int64
 10  fgm                              8498 non-null   int64
 11  fga                              8498 non-null   int64
 12  fgp                              8444 non-null   float64
 13  fgm3                             8498 non-null   int64
 14  fga3                             8498 non-null   int64
 15  fgp3                             7454 non-null   float64
 16  fgm2                             8498 non-null   int64
 17  fga2                             8498 non-null   int64
 18  fgp2                             8383 non-null   float64
 19  efg                              8444 non-null   float64
 20  ftm                              8498 non-null   int64
 21  fta                              8498 non-null   int64
 22  ftp                              8008 non-null   float64
 23  off_reb                          8498 non-null   int64
 24  def_reb                          8498 non-null   int64
 25  tot_reb                          8498 non-null   int64
 26  ast                              8498 non-null   int64
 27  steals                           8498 non-null   int64
 28  blocks                           8498 non-null   int64
 29  tov                              8498 non-null   int64
```

```
 30  tot_fouls                      8498 non-null   int64
 31  points                         8498 non-null   int64
 32  PER                            8498 non-null   float64
 33  FTr                            8444 non-null   float64
 34  off_reb_pct                    8498 non-null   float64
 35  def_reb_pct                    8498 non-null   float64
 36  tot_reb_pct                    8498 non-null   float64
 37  ast_pct                        8498 non-null   float64
 38  stl_pct                        8498 non-null   float64
 39  blk_pct                        8498 non-null   float64
 40  tov_pct                        8453 non-null   float64
 41  usg                            8498 non-null   float64
 42  OWS                            8498 non-null   float64
 43  DWS                            8498 non-null   float64
 44  WS                             8498 non-null   float64
 45  OBPM                           8498 non-null   float64
 46  DBPM                           8498 non-null   float64
 47  BPM                            8498 non-null   float64
 48  VORP                           8498 non-null   float64
 49  All NBA Defensive First Team   713 non-null    float64
 50  All NBA Defensive Second Team  713 non-null    float64
 51  All NBA First Team             713 non-null    float64
 52  All NBA Second Team            713 non-null    float64
 53  All NBA Third Team             713 non-null    float64
 54  All Rookie First Team          713 non-null    float64
 55  All Rookie Second Team         713 non-null    float64
 56  Bill Russell NBA Finals MVP    713 non-null    float64
 57  Player Of The Month            713 non-null    float64
 58  Player Of The Week             713 non-null    float64
 59  Rookie Of The Month            713 non-null    float64
 60  all_star_game                  696 non-null    object
 61  rookie_all_star_game           696 non-null    object
 62  allstar_rk                     3849 non-null   float64
 63  Defensive Player Of The Year_rk  260 non-null  float64
 64  Most Improved Player_rk        412 non-null    float64
 65  Most Valuable Player_rk        206 non-null    float64
 66  Rookie Of The Year_rk          125 non-null    float64
 67  Sixth Man Of The Year_rk       252 non-null    float64
 68  all_nba_points_rk              401 non-null    float64
 69  all_rookie_points_rk           271 non-null    float64
dtypes: float64(42), int64(24), object(4)
memory usage: 4.6+ MB
```

[20]:
```python
# Define the adjusted games calculation
def adjust_games(games, season):
    if season == 2011:
        return games * (82 / 66)
```

```python
    elif season in [2019, 2020]:
        return games * (82 / 72)
    else:
        return games

# Define the adjusted minutes calculation
def adjust_minutes(minutes, season):
    if season == 2011:
        return minutes * (82 / 66)
    elif season in [2019, 2020]:
        return minutes * (82 / 72)
    else:
        return minutes
```

[21]:
```python
# Adjust games started and minutes for each player's season
combined_data['games_start_adj'] = combined_data.apply(lambda row:␣
 ↪adjust_games(row['games_start'], row['season']), axis=1)
combined_data['mins_adj'] = combined_data.apply(lambda row:␣
 ↪adjust_minutes(row['mins'], row['season']), axis=1)

# Determine the highest level of success for each player's season
def determine_level(row):
    if row['all_nba_points_rk'] <= 15 or row['Most Valuable Player_rk'] == 1:
        return 'Elite'
    elif row['all_star_game'] == 1:
        return 'All-Star'
    elif row['games_start_adj'] >= 41 or row['mins_adj'] >= 2000:
        return 'Starter'
    elif row['mins_adj'] >= 1000:
        return 'Rotation'
    elif row['games'] >= 1:
        return 'Roster'
    else:
        return 'Out of the League'
```

[22]:
```python
# Determine the career outcome based on highest level of success in at least␣
 ↪two seasons
def determine_career_outcome(levels):
    unique_levels = np.unique(levels)
    if len(unique_levels) >= 3:
        return unique_levels[-1]
    elif len(unique_levels) == 2 and unique_levels[-1] == 'Roster':
        return unique_levels[-2]
    elif len(unique_levels) == 2:
        return unique_levels[-1]
    else:
        return 'Out of the League'
```

[23]:
```python
awards['level'] = combined_data.apply(determine_level, axis=1)
```

[24]:
```python
# Group and aggregate player awards data to determine career outcomes
career_outcomes = awards.groupby('nbapersonid')['level'].
    ↪agg(determine_career_outcome).reset_index()
```

[25]:
```python
career_outcomes
```

[25]:
```
       nbapersonid             level
0     2.550000e+02           Starter
1     4.060000e+02          Rotation
2     4.670000e+02           Starter
3     6.860000e+02  Out of the League
4     7.080000e+02           Starter
…              …                 …
1183  1.630787e+06  Out of the League
1184  1.630792e+06  Out of the League
1185  1.630928e+06          Rotation
1186  1.631310e+06  Out of the League
1187  1.962937e+09          Rotation

[1188 rows x 2 columns]
```

[26]:
```python
# Print the career outcomes for each player
print(career_outcomes["level"].value_counts())
```

```
Starter              545
Out of the League    468
Rotation             149
All-Star              17
Elite                  9
Name: level, dtype: int64
```

ANSWER 3:

Elite: 9 players.
All-Star: 17 players.
Starter: 545 players.
Rotation: 149 players.
Roster: 0 player.
Out of League: 468 players.

4.2.2 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 the first four years of his 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 experience with "formal" modeling. Don't be discouraged. It's not our intention to make you spend excessive time here. If you get your model to a good spot but think you could do better by spending a lot more time, you can just write a bit about your ideas for future improvement and leave it there. Further, we're more interested in your thought process and critical thinking than we are in specific modeling techniques. Using smart features is more important than using fancy mathematical machinery, and a successful candidate could use a simple regression approach.

2. You may use any data provided in this project, but please do not bring in any external sources of data. Note that while most of the data provided goes back to 2007, All NBA and All Rookie team voting is only included back to 2011.

3. A player needs to complete three additional seasons after their first four to be considered as having a distinct career outcome for our dataset. Because the dataset in this project ends in 2021, this means that a player would need to have had the chance to play in the '21, '20, and '19 seasons after his first four years, and thus his first four years would have been '18, '17, '16, and '15. **For this reason, limit your training data to players who were drafted in or before the 2015 season.** Karl-Anthony Towns was the #1 pick in that season.

4. Once you build your model, predict on all players who were drafted in 2018-2021 (They have between 1 and 4 seasons of data available 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 each player falls into each 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 without a strong statistical 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 matplotlib or plotly visualization highlighting some part of your modeling process, the model itself, or your results.
- 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 drafted in 2019-2021.

Overview of the Model: Our model aims to predict whether a player will have an "Elite" career outcome based on their performance in the first four years of their NBA career. An "Elite" outcome is defined as making it to the All-NBA Defensive First/Second Team or winning the Most Valuable Player award. We use a machine learning algorithm called HistGradientBoostingClassifier, which learns patterns in the data to make accurate predictions.

Strengths:

- Handles complex relationships: The model can capture non-linear relationships and interactions between various player statistics, providing a more accurate representation of a player's potential success.
- Handles missing data: The model can handle missing values in the dataset without requiring imputation or data preprocessing.

Weaknesses:

- Limited data: The dataset contains information only up to the 2020-2021 season, which might not capture recent changes in player performance or new trends in the NBA.
- Simplified features: The model uses a simplified set of player statistics as features, potentially missing out on some crucial indicators of performance.
- Lack of context: The model does not consider external factors such as injuries, team dynamics, or coaching changes, which can significantly impact a player's career trajectory.

Addressing Weaknesses with More Data and Time:

- More recent data: Gathering data beyond the 2020-2021 season could provide insights into current trends and player development trajectories.
- Advanced features: Incorporating more advanced features like advanced player tracking data, social media sentiment analysis, or injury history could enhance the model's accuracy.
- Contextual information: Introducing external variables like team performance, player trades, and coaching changes could provide a more comprehensive understanding of a player's career path.

[27]:
```
from sklearn.ensemble import HistGradientBoostingClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report
```

[28]:
```
# Merge the data on player ID
data = player_data.merge(awards, on=['season', 'nbapersonid'], how='inner')
```

[29]:
```
# Create a binary target variable 'Elite'
data['Elite'] = data.apply(lambda row: 1 if row['all_nba_points_rk'] <= 15 or␣
 ↪row['Most Valuable Player_rk'] == 1 else 0, axis=1)

# Select features for modeling
features = [
    'PER', 'FTr', 'off_reb_pct', 'def_reb_pct', 'ast_pct', 'stl_pct', 'blk_pct',
    'tov_pct', 'usg', 'OWS', 'DWS', 'WS', 'OBPM', 'DBPM', 'BPM'
]

X = data[features]
y = data['Elite']

# Split data into training and testing sets
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

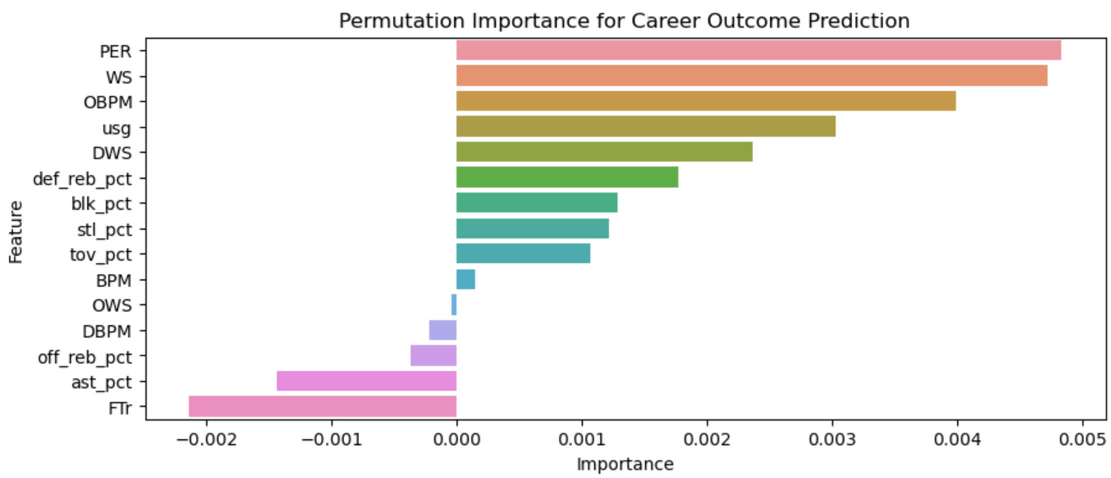
[30]:
```
# Initialize and fit the HistGradientBoostingClassifier model
model = HistGradientBoostingClassifier()
model.fit(X_train, y_train)
```

[30]: HistGradientBoostingClassifier()

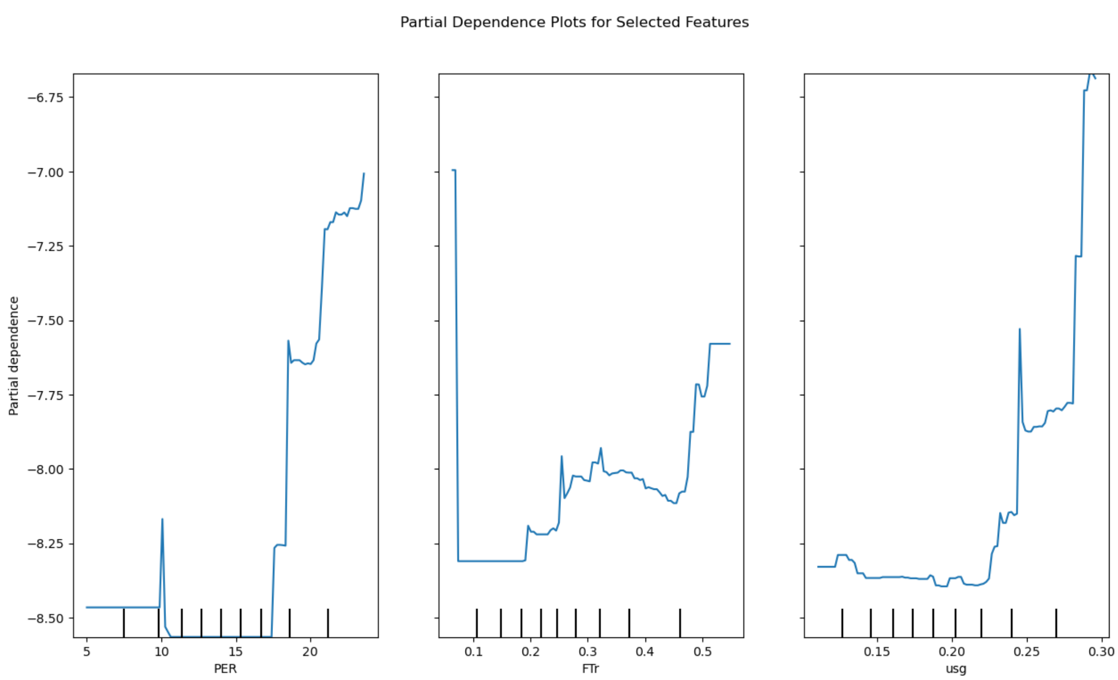
[31]:
```
# Predictions on the test set
y_pred = model.predict(X_test)

# Print classification report
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       0.99      0.99      0.99       877
           1       0.60      0.56      0.58        27

    accuracy                           0.98       904
   macro avg       0.79      0.77      0.78       904
weighted avg       0.97      0.98      0.98       904
```

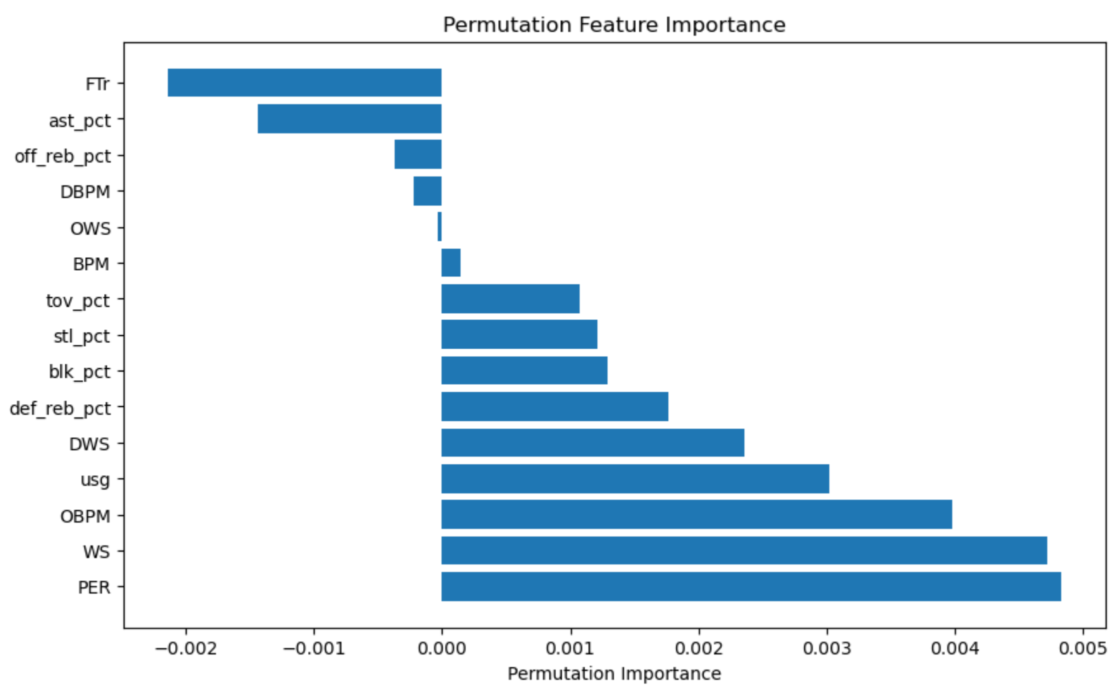
[32]:
```
from sklearn.inspection import permutation_importance

# Calculate permutation importance
perm_importance = permutation_importance(model, X_test, y_test, n_repeats=30, random_state=42)

# Get feature names
feature_names = features

# Create a DataFrame for permutation importances
perm_importance_df = pd.DataFrame({'Feature': feature_names, 'Importance': perm_importance.importances_mean})
perm_importance_df = perm_importance_df.sort_values(by='Importance', ascending=False)

# Create a bar plot using Seaborn
plt.figure(figsize=(10, 4))
sns.barplot(x='Importance', y='Feature', data=perm_importance_df)
plt.title('Permutation Importance for Career Outcome Prediction')
plt.xlabel('Importance')
plt.ylabel('Feature')
plt.show()
```



[47]:
```python
from sklearn.inspection import plot_partial_dependence

# Select the features for which you want to plot partial dependence
features_to_plot = ['PER', 'FTr', 'usg']

# Create partial dependence plots
fig, ax = plt.subplots(figsize=(15, 8))
plot_partial_dependence(model, X_train, features=features_to_plot, ax=ax)
plt.suptitle('Partial Dependence Plots for Selected Features')
plt.subplots_adjust(top=0.9)
plt.show()
```



[34]:
```python
from sklearn.inspection import permutation_importance

# Calculate permutation feature importance
perm_importance = permutation_importance(model, X_test, y_test, n_repeats=30,␣
↪random_state=42)

# Get feature names
feature_names = X_test.columns

# Calculate mean importance scores
mean_importance = perm_importance.importances_mean

# Sort feature importances in descending order
sorted_idx = mean_importance.argsort()[::-1]

# Visualize feature importances
plt.figure(figsize=(10, 6))
plt.barh(range(X_test.shape[1]), mean_importance[sorted_idx], align='center')
plt.yticks(range(X_test.shape[1]), [feature_names[i] for i in sorted_idx])
plt.xlabel('Permutation Importance')
plt.title('Permutation Feature Importance')
plt.show()
```



[35]:

```python
# Predictions for specific players
player_names = ['Shai Gilgeous-Alexander', 'Zion Williamson', 'James Wiseman',␣
 ↪'Josh Giddey']
players = player_data[player_data['player'].isin(player_names)]
players_X = players[features]
predictions = model.predict(players_X)

for player, prediction in zip(player_names, predictions):
    print(f"{player}: {'Elite' if prediction == 1 else 'Not Elite'}")
```

```
Shai Gilgeous-Alexander: Not Elite
Zion Williamson: Not Elite
James Wiseman: Not Elite
Josh Giddey: Not Elite
```

4.3 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 and then discuss ways to improve those predictions.

4.3.1 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.

[37]:

```python
rebounding_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2460 entries, 0 to 2459
Data columns (total 7 columns):
 #   Column              Non-Null Count  Dtype  
---  ------              --------------  -----  
 0   team                2460 non-null   object 
 1   opp_team            2460 non-null   object 
 2   gamedate            2460 non-null   object 
 3   game_number         2460 non-null   int64  
 4   offensive_rebounds  2460 non-null   int64  
 5   off_rebound_chances 2460 non-null   int64  
 6   oreb_pct            2460 non-null   float64
dtypes: float64(1), int64(3), object(3)
memory usage: 134.7+ KB
```

[41]:
```
# Filter data for OKC's games
okc_data = rebounding_data[rebounding_data['team'] == 'OKC']

# Calculate average offensive rebounding percent for prior games
average_off_reb_percent = okc_data['offensive_rebounds'].sum() /␣
 ↪okc_data['off_rebound_chances'].sum()

# Predict offensive rebounding percent for game 81 using the average
predicted_off_reb_percent = average_off_reb_percent

print("Predicted Offensive Rebound Percent for Game 81:",␣
 ↪round(predicted_off_reb_percent*100,1))
```

```
Predicted Offensive Rebound Percent for Game 81: 28.9
```

ANSWER 1:

28.9%

4.3.2 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 2:

To better account for missing players and their impact on team offensive rebound percentages, we can consider the following approaches:

- **Player-Specific Impact**: Instead of using the average offensive rebounding percentage for all prior games, we can calculate a weighted average that takes into account the offensive rebounding performance of individual players who will be available for the next game. This way, the absence of a strong offensive rebounder due to injury or other reasons will have a more direct impact on the prediction.

- **Player Lineup Data**: Incorporating player lineup data can provide insights into how certain combinations of players on the court influence offensive rebounding. By analyzing historical lineup data and their offensive rebounding outcomes, we can identify which player combinations tend to result in higher offensive rebound percentages. This information can be used to adjust the prediction based on the expected lineup for the next game.

- **Opponent Analysis**: The offensive rebounding performance of a team can also be influenced by the defensive rebounding abilities of the opponent. If the upcoming opponent has a strong defensive rebounding presence, the predicted offensive rebound percentage might be adjusted downwards. Analyzing opponent data and their defensive rebounding performance can provide valuable context for making predictions.

- **Historical Performance**: Instead of relying solely on average offensive rebounding percentages, we can consider the team's recent offensive rebounding performance over a shorter time frame. This approach would give more weight to recent games and capture any evolving trends in offensive rebounding performance.
- **Machine Learning Models**: We can build more sophisticated machine learning models that take into account various player-specific features, lineup combinations, opponent data, and other relevant factors. These models can learn complex relationships and patterns from the data to make more accurate predictions.

4.3.3 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 3:

Here are a few potential weaknesses of the simple average model used to predict offensive rebound percentages:

- **Lack of Context**: The simple average model doesn't take into account the specific circumstances of each game, such as the team's playing style, opponent's defensive strategy, and game situation. To address this, we could implement a context-aware model that considers various game-specific features. For example, if a team is playing against a strong defensive rebounding opponent, the model could adjust the prediction accordingly. Additionally, incorporating data on game pace, shot selection, and other contextual factors would provide a more nuanced prediction.
- **Player Variability**: Offensive rebounding ability varies significantly among players, and the simple average model treats all players equally. To overcome this limitation, we could create player-specific models that predict the offensive rebounding performance of individual players based on their historical data. These player-specific predictions can then be combined to estimate the team's overall offensive rebounding percentage. This approach accounts for player strengths and weaknesses and provides a more accurate prediction.
- **Small Sample Size**: In some cases, the available historical data might be limited, especially for new or reshuffled teams. The simple average model relies on a larger number of historical games for accuracy. To handle small sample sizes, we could implement a Bayesian approach that combines prior information (such as league-wide averages) with team-specific data. This would allow the model to provide predictions even with limited historical data.