A Project on Basketball Analytics

Sophia Huang

5/2/2022

Introduction

This project is based on NBA data for both the regular season and playoff season. The NBA stands for the National Basketball Association and is split into two conferences: the East and West. The league consists of thirty teams, and it is one of the largest sports organizations in the United States. In the NBA's regular season, each team plays 82 games. Following regular season are the playoffs. During the playoffs, two teams, one from each conference, play against one another in each round. They play for the first to four wins, with at most seven games. The winning team advances to the next round until there are only two teams remaining. Each year, there is only one NBA champion.

For my project, I answered four questions:

1. Is there an advantage playing at home vs. being the visiting team?
2. Can looking at defensive rebounds help us predict game outcome?
3. Does the distribution of possessions and defensive plays change for different positions? (Playoffs 2016-2022)
4. What is the most common reason for turnovers in the 2021-2022 Playoff season so far?

From my data exploration, I conclude that for my first question, there is an advantage playing at home. To answer the second question, I looked at the number of defensive rebounds the home team got compared to the away team and stored this number as a difference. Defensive rebounds are when the defensive team gains possession of the ball after a missed field goal attempt. I modeled this to predict whether the home/visitor team wins of loses. I then looked at whether the distribution of possessions and defensive plays change for different positions. From this, I conclude that although the distribution of possessions is quite similar, the distribution for defensive plays is different for different starting positions. The starting positions I looked at were Centers (C), Guards (G), and Forwards (F). I observed that Centers tend to have the most defensive plays. This makes sense because Centers are typically one of the tallest players on the team for the purpose of standing them under the basket. For this question, I defined a defensive play as a defensive rebound, a steal, or a block. Lastly, I looked at the most common reason for turnovers in the 2021-2022 Playoffs so far and found that overall, bad passes account for the most amount of turnovers.

Data Source

I gathered my data in VSCode using Python with the NBA API created by Swar Patel. You can access the NBA API here: https://github.com/swar/nba_api! This API is extremely useful because you can scrape almost any data you want directly from the NBA website. All of the parameters for scraping data from the NBA website are indicated in each file, and I adjusted the nullable values such as season_nullable to "Regular Season" or "Playoffs" to look at specific seasons.

From here, I gathered traditional box scores, advanced box scores, and play by play data for both the playoff season and regular season. I gathered team box scores as well as individual player box scores, which just let us know how each team/player played in a game. From these box scores, we can see the team/player's total points made, defensive rebounds, assists, blocks, etc. Play by play data gives us details on what occurred in each game with a description. For example, it records the actions of players such as "Duncan Turnover" or "Curry Rebound". To access these data sets for each team in the NBA, I went in VSCode and created functions that iterated through a list of teams. For the functions, I returned a CSV file. I also called the League Game Finder to extract all of the regular season games since the start of the NBA. The teams data set can be found in the statics folder while the box scores and play by play data are accessed through endpoints. You can access the endpoints by going to the link provided above then going to the following directories:

`https://github.com/swar/nba_api/blob/master/docs/nba_api/stats/endpoints/boxscoretraditionalv2.md`

`nba_api/docs/nba_api/stats/static/teams.md`

Ethics Reflection

Some ethical issues to consider are harms to fairness and justice as well as harms to privacy and security. Because the NBA is a competitive league, publishing data detailing player performance as well as team performance has a two-fold effect: on one hand, it increases transparency as the data is publicized, but on the other hand, it could harm the fairness of the game and privacy of players. For example, while this data set does not detail extremely personal characteristics, the use of this analysis paired with more personal player information could border on being ethically harmful to a player's privacy. Furthermore, because there is public data available, the analysis of this data could lead to a big change in the way the NBA is played. This could harm the fairness and justice of the game because teams with more money might be able to adapt to this change easier or teams could intentionally target star players on an opposing team, leading to dirty game play and injuries. On the other hand, it could also progress the game of basketball to be more fair or to prevent more injuries.

Data Import

```r
allgames <- read_csv("gamehistory.csv")
allgames # Used in Q1
```

```
## # A tibble: 30,000 x 29
##     ...1 SEASON_ID TEAM_ID TEAM_ABBREVIATI~ TEAM_NAME GAME_ID GAME_DATE  MATCHUP
##    <dbl>     <dbl>   <dbl> <chr>            <chr>     <chr>   <date>     <chr>
##  1     0     22021  1.61e9 UTA              Utah Jazz 002210~ 2022-04-10 UTA @ ~
##  2     1     22021  1.61e9 DAL              Dallas M~ 002210~ 2022-04-10 DAL vs~
##  3     2     22021  1.61e9 WAS              Washingt~ 002210~ 2022-04-10 WAS @ ~
##  4     3     22021  1.61e9 CHI              Chicago ~ 002210~ 2022-04-10 CHI @ ~
##  5     4     22021  1.61e9 LAL              Los Ange~ 002210~ 2022-04-10 LAL @ ~
##  6     5     22021  1.61e9 MIN              Minnesot~ 002210~ 2022-04-10 MIN vs~
##  7     6     22021  1.61e9 NOP              New Orle~ 002210~ 2022-04-10 NOP vs~
##  8     7     22021  1.61e9 POR              Portland~ 002210~ 2022-04-10 POR vs~
##  9     8     22021  1.61e9 SAS              San Anto~ 002210~ 2022-04-10 SAS @ ~
## 10     9     22021  1.61e9 DEN              Denver N~ 002210~ 2022-04-10 DEN vs~
## # ... with 29,990 more rows, and 21 more variables: WL <chr>, MIN <dbl>,
## #   PTS <dbl>, FGM <dbl>, FGA <dbl>, FG_PCT <dbl>, FG3M <dbl>, FG3A <dbl>,
## #   FG3_PCT <dbl>, FTM <dbl>, FTA <dbl>, FT_PCT <dbl>, OREB <dbl>, DREB <dbl>,
## #   REB <dbl>, AST <dbl>, STL <dbl>, BLK <dbl>, TOV <dbl>, PF <dbl>,
## #   PLUS_MINUS <dbl>
```

```r
playbyplay <- read_csv("playbyplay.csv") # Used in Q2
playbyplay
```

```
## # A tibble: 28,512 x 35
##     ...1 GAME_ID    EVENTNUM EVENTMSGTYPE EVENTMSGACTIONTYPE PERIOD WCTIMESTRING
##    <dbl> <chr>         <dbl>        <dbl>              <dbl>  <dbl> <time>
##  1     0 0042100215        2           12                  0      1 19:08
##  2     1 0042100215        4           10                  0      1 19:08
##  3     2 0042100215        7            1                  1      1 19:08
##  4     3 0042100215        9            1                  1      1 19:08
##  5     4 0042100215       11            1                 79      1 19:08
##  6     5 0042100215       12            2                 79      1 19:09
##  7     6 0042100215       13            4                  0      1 19:09
##  8     7 0042100215       14            2                 80      1 19:09
##  9     8 0042100215       15            4                  0      1 19:09
## 10     9 0042100215       16            2                 57      1 19:09
## # ... with 28,502 more rows, and 28 more variables: PCTIMESTRING <time>,
## #   HOMEDESCRIPTION <chr>, NEUTRALDESCRIPTION <chr>, VISITORDESCRIPTION <chr>,
## #   SCORE <chr>, SCOREMARGIN <chr>, PERSON1TYPE <dbl>, PLAYER1_ID <dbl>,
## #   PLAYER1_NAME <chr>, PLAYER1_TEAM_ID <dbl>, PLAYER1_TEAM_CITY <chr>,
## #   PLAYER1_TEAM_NICKNAME <chr>, PLAYER1_TEAM_ABBREVIATION <chr>,
## #   PERSON2TYPE <dbl>, PLAYER2_ID <dbl>, PLAYER2_NAME <chr>,
## #   PLAYER2_TEAM_ID <dbl>, PLAYER2_TEAM_CITY <chr>, ...
```

```r
padvancedbox <- read_csv("PLAYOFFSABS.csv") %>% select(-`Unnamed: 0`) # Used in Q3
padvancedbox
```

```
## # A tibble: 12,649 x 33
##     ...1  GAME_ID    TEAM_ID TEAM_ABBREVIATION TEAM_CITY PLAYER_ID PLAYER_NAME
##    <dbl>    <dbl>      <dbl> <chr>             <chr>         <dbl> <chr>
##  1     0 41600405 1610612739 CLE               Cleveland      2544 LeBron James
##  2     1 41600405 1610612739 CLE               Cleveland    201567 Kevin Love
##  3     2 41600405 1610612739 CLE               Cleveland    202684 Tristan Thom~
##  4     3 41600405 1610612739 CLE               Cleveland      2747 JR Smith
##  5     4 41600405 1610612739 CLE               Cleveland    202681 Kyrie Irving
##  6     5 41600405 1610612739 CLE               Cleveland      2210 Richard Jeff~
##  7     6 41600405 1610612739 CLE               Cleveland      2594 Kyle Korver
##  8     7 41600405 1610612739 CLE               Cleveland    101114 Deron Willia~
##  9     8 41600405 1610612739 CLE               Cleveland    202697 Iman Shumpert
## 10     9 41600405 1610612739 CLE               Cleveland    101112 Channing Frye
## # ... with 12,639 more rows, and 26 more variables: NICKNAME <chr>,
## #   START_POSITION <chr>, COMMENT <chr>, MIN <time>, E_OFF_RATING <dbl>,
## #   OFF_RATING <dbl>, E_DEF_RATING <dbl>, DEF_RATING <dbl>, E_NET_RATING <dbl>,
## #   NET_RATING <dbl>, AST_PCT <dbl>, AST_TOV <dbl>, AST_RATIO <dbl>,
## #   OREB_PCT <dbl>, DREB_PCT <dbl>, REB_PCT <dbl>, TM_TOV_PCT <dbl>,
## #   EFG_PCT <dbl>, TS_PCT <dbl>, USG_PCT <dbl>, E_USG_PCT <dbl>, E_PACE <dbl>,
## #   PACE <dbl>, PACE_PER40 <dbl>, POSS <dbl>, PIE <dbl>
```

```r
ptradbox <- read_csv("PLAYOFFSBST.csv") %>% select(-`Unnamed: 0`) # Used in Q3
ptradbox
```

```
## # A tibble: 12,706 x 30
##     ...1  GAME_ID    TEAM_ID TEAM_ABBREVIATION TEAM_CITY PLAYER_ID PLAYER_NAME
##    <dbl>    <dbl>      <dbl> <chr>             <chr>         <dbl> <chr>
##  1     0 41600405 1610612739 CLE               Cleveland      2544 LeBron James
##  2     1 41600405 1610612739 CLE               Cleveland    201567 Kevin Love
##  3     2 41600405 1610612739 CLE               Cleveland    202684 Tristan Thom~
##  4     3 41600405 1610612739 CLE               Cleveland      2747 JR Smith
##  5     4 41600405 1610612739 CLE               Cleveland    202681 Kyrie Irving
##  6     5 41600405 1610612739 CLE               Cleveland      2210 Richard Jeff~
##  7     6 41600405 1610612739 CLE               Cleveland      2594 Kyle Korver
##  8     7 41600405 1610612739 CLE               Cleveland    101114 Deron Willia~
##  9     8 41600405 1610612739 CLE               Cleveland    202697 Iman Shumpert
## 10     9 41600405 1610612739 CLE               Cleveland    101112 Channing Frye
## # ... with 12,696 more rows, and 23 more variables: NICKNAME <chr>,
## #   START_POSITION <chr>, COMMENT <chr>, MIN <time>, FGM <dbl>, FGA <dbl>,
## #   FG_PCT <dbl>, FG3M <dbl>, FG3A <dbl>, FG3_PCT <dbl>, FTM <dbl>, FTA <dbl>,
## #   FT_PCT <dbl>, OREB <dbl>, DREB <dbl>, REB <dbl>, AST <dbl>, STL <dbl>,
## #   BLK <dbl>, TO <dbl>, PF <dbl>, PTS <dbl>, PLUS_MINUS <dbl>
```

In each of these lines, I'm importing the CSV file stored in the same folder this project is stored in. As you may see, I will be looking at allgames, playbyplay, padvancedbox, and ptradbox. Allgames contains every single game played in the regular season of the NBA and contains information on game date, matchup (who played who), winning/loosing team, minutes played etc. Each row represents one team per game. Next, the playbyplay dataset contains information as stated above in the Data Source Section. Essentially, this data set gives us details on what occurred in each game with a written description. Padvancedbox and Ptradbox represent the advanced box scores and traditional box scores of the Playoffs. Each row contains information on a player's performance in each game.

Data Cleaning and Tidying

Question 1 Datasets:

```
games <- allgames %>%  # All reg season games
  separate(GAME_DATE, into = c("Year", "Month", "Day"), sep = "-") %>%
  mutate(MATCHUP = ifelse(str_detect(MATCHUP, "@") == TRUE, "VISITOR", "HOME")) %>%
  select(GAME_ID, MATCHUP, WL, PTS, DREB)
games
```

```
## # A tibble: 30,000 x 5
##    GAME_ID    MATCHUP WL      PTS  DREB
##    <chr>      <chr>   <chr> <dbl> <dbl>
##  1 0022101230 VISITOR W       111    45
##  2 0022101219 HOME    W       130    33
##  3 0022101217 VISITOR L       108    26
##  4 0022101224 VISITOR W       124    32
##  5 0022101220 VISITOR W       146    37
##  6 0022101224 HOME    L       120    23
##  7 0022101225 HOME    L       107    24
##  8 0022101230 HOME    L        80    27
##  9 0022101219 VISITOR L       120    28
## 10 0022101220 HOME    L       141    34
## # ... with 29,990 more rows
```

In the above code chunk, I separate the game date into a year, month, day format where all three variables take on their own column. Then, to determine whether a team is the home team or visiting team, I looked to see whether the character "@" was in the matchup column. If it is, then it indicates this team played at another team's home field, meaning that they are the visiting team.

```
visitors <- games %>%
  filter(MATCHUP == "VISITOR") %>%
  rename("VISITOR_WL" = "WL", "VISITOR_PTS" = "PTS", "VISITOR_DREB" = "DREB")
visitors
```

```
## # A tibble: 15,000 x 5
##    GAME_ID    MATCHUP VISITOR_WL VISITOR_PTS VISITOR_DREB
##    <chr>      <chr>   <chr>            <dbl>        <dbl>
##  1 0022101230 VISITOR W                  111           45
##  2 0022101217 VISITOR L                  108           26
##  3 0022101224 VISITOR W                  124           32
##  4 0022101220 VISITOR W                  146           37
##  5 0022101219 VISITOR L                  120           28
##  6 0022101218 VISITOR L                  115           33
##  7 0022101221 VISITOR W                  130           37
##  8 0022101227 VISITOR L                  111           37
##  9 0022101216 VISITOR L                  126           19
## 10 0022101229 VISITOR W                  116           38
## # ... with 14,990 more rows
```

In the code chunk above, I wanted to look at all instances of the visiting team. I renamed each column to include the word "VISITOR" in the column name to help me distinguish whether I am looking at a visitor or home team's points.

```
hometeams <- games %>%
  filter(MATCHUP == "HOME") %>%
  rename("HOME_WL" = "WL", "HOME_PTS" = "PTS", "HOME_DREB" = "DREB")
hometeams
```

```
## # A tibble: 15,000 x 5
##    GAME_ID    MATCHUP HOME_WL HOME_PTS HOME_DREB
##    <chr>      <chr>   <chr>      <dbl>     <dbl>
##  1 0022101219 HOME    W            130        33
##  2 0022101224 HOME    L            120        23
##  3 0022101225 HOME    L            107        24
##  4 0022101230 HOME    L             80        27
##  5 0022101220 HOME    L            141        34
##  6 0022101221 HOME    L            114        28
##  7 0022101223 HOME    L            110        26
##  8 0022101217 HOME    W            124        37
##  9 0022101216 HOME    W            134        42
## 10 0022101218 HOME    W            133        38
## # ... with 14,990 more rows
```

This code chunk above essentially does the same thing as the previous one, but instead of filtering for visiting teams, I'm only looking at home teams.

```
game_data <- left_join(visitors, hometeams, by = "GAME_ID") %>%
  select(-MATCHUP.x, -MATCHUP.y) %>%
  mutate("WINNER" = ifelse(VISITOR_WL == "L", 1, 0)) %>%
  select(GAME_ID, VISITOR_PTS, VISITOR_DREB, HOME_PTS, HOME_DREB, WINNER) %>%
  mutate(DIFF_DREB = HOME_DREB - VISITOR_DREB, DIFF_PTS = HOME_PTS - VISITOR_PTS)%>%
  select(GAME_ID:HOME_DREB, DIFF_DREB, DIFF_PTS, WINNER)
game_data
```

```
## # A tibble: 15,000 x 8
##    GAME_ID VISITOR_PTS VISITOR_DREB HOME_PTS HOME_DREB DIFF_DREB DIFF_PTS WINNER
##    <chr>         <dbl>        <dbl>    <dbl>     <dbl>     <dbl>    <dbl>  <dbl>
##  1 002210~         111           45       80        27       -18      -31      0
##  2 002210~         108           26      124        37        11       16      1
##  3 002210~         124           32      120        23        -9       -4      0
##  4 002210~         146           37      141        34        -3       -5      0
##  5 002210~         120           28      130        33         5       10      1
##  6 002210~         115           33      133        38         5       18      1
##  7 002210~         130           37      114        28        -9      -16      0
##  8 002210~         111           37      125        42         5       14      1
##  9 002210~         126           19      134        42        23        8      1
## 10 002210~         116           38      109        32        -6       -7      0
## # ... with 14,990 more rows
```

```
# 1 represents a home team win, 0 represents a visitor win
```

Then, I wanted to represent the win loss column as a binary value, with 1 representing a home win and 0 representing a visitor win. The purpose of this was so when we do logistic regression, I can make the win/loss as the dependent variable. I then took the differences in defensive rebunds as well as difference in points by taking the home team values - visitor values. I did this because this tells us how one team is doing relative to their opponent. So, a positive value for defensive rebounds and points indicates the home team is leading, while a negative value represents the visitor team leading.

Question 2 Dataset:

```
boxscores <- left_join(padvancedbox, ptradbox) %>%
  filter(!is.na(MIN)) %>%
  mutate(SECONDS = (period_to_seconds(hms(MIN))/60), DEFENSIVE_PLAYS = BLK + STL + DREB) %>%
  select(GAME_ID, PLAYER_NAME, START_POSITION, SECONDS, POSS, BLK, STL, DREB, DEFENSIVE_PLAYS)
boxscores
```

```
## # A tibble: 10,111 x 9
##     GAME_ID PLAYER_NAME       START_POSITION SECONDS  POSS   BLK   STL  DREB
##       <dbl> <chr>             <chr>            <dbl> <dbl> <dbl> <dbl> <dbl>
##  1 41600405 LeBron James      F                 2773    93     1     2    11
##  2 41600405 Kevin Love        F                 1795    57     1     0     7
##  3 41600405 Tristan Thompson  C                 1792    60     1     1     4
##  4 41600405 JR Smith          G                 2449    84     2     0     3
##  5 41600405 Kyrie Irving      G                 2507    86     0     2     1
##  6 41600405 Richard Jefferson <NA>              1047    38     0     0     0
##  7 41600405 Kyle Korver       <NA>              1068    38     0     0     0
##  8 41600405 Deron Williams    <NA>               744    27     0     1     2
##  9 41600405 Iman Shumpert     <NA>               225    10     0     0     0
## 10 41600405 Kevin Durant      F                 2415    81     0     1     5
## # ... with 10,101 more rows, and 1 more variable: DEFENSIVE_PLAYS <dbl>
```

In the code chunk above, I join the advanced box scores and traditional box scores from the playoffs, then filter out all of the player who did not play any minutes in a game. I converted the time value to seconds and created a new column called DEFENSIVE_PLAYS that adds up a player's blocks, steals, and defensive rebounds.

Question 3 Dataset:

```
positions <- boxscores %>%
  mutate(START_POSITION = replace(START_POSITION, is.na(START_POSITION), "Not a Starter"))
positions
```

```
## # A tibble: 10,111 x 9
##     GAME_ID PLAYER_NAME       START_POSITION SECONDS  POSS   BLK   STL  DREB
##       <dbl> <chr>             <chr>            <dbl> <dbl> <dbl> <dbl> <dbl>
##  1 41600405 LeBron James      F                 2773    93     1     2    11
##  2 41600405 Kevin Love        F                 1795    57     1     0     7
##  3 41600405 Tristan Thompson  C                 1792    60     1     1     4
##  4 41600405 JR Smith          G                 2449    84     2     0     3
##  5 41600405 Kyrie Irving      G                 2507    86     0     2     1
##  6 41600405 Richard Jefferson Not a Starter     1047    38     0     0     0
##  7 41600405 Kyle Korver       Not a Starter     1068    38     0     0     0
##  8 41600405 Deron Williams    Not a Starter      744    27     0     1     2
##  9 41600405 Iman Shumpert     Not a Starter      225    10     0     0     0
## 10 41600405 Kevin Durant      F                 2415    81     0     1     5
## # ... with 10,101 more rows, and 1 more variable: DEFENSIVE_PLAYS <dbl>
```

This code chunk just renames all of the positions of players who did not start in a game as "Not a Starter".

Data Exploration

Question 1: Is there an advantage playing at home vs. being a visitor?

For this question, I wanted to dive deeper into thinking about whether teams have a home court advantage. To do this, I used the game_data csv which contains 15,000 rows, which each row representing a unique game. Thus, we also know that each row contains data on two teams: the home team and away team. To answer this question, I first took the mean of the winner column, which tells us what proportion of the wins were home team wins. The reason the mean of this column gives us the proportion is because a home team win is denoted by a 1, so we are just adding up all values of 1 then dividing by the total number of games.

```
game_data
```

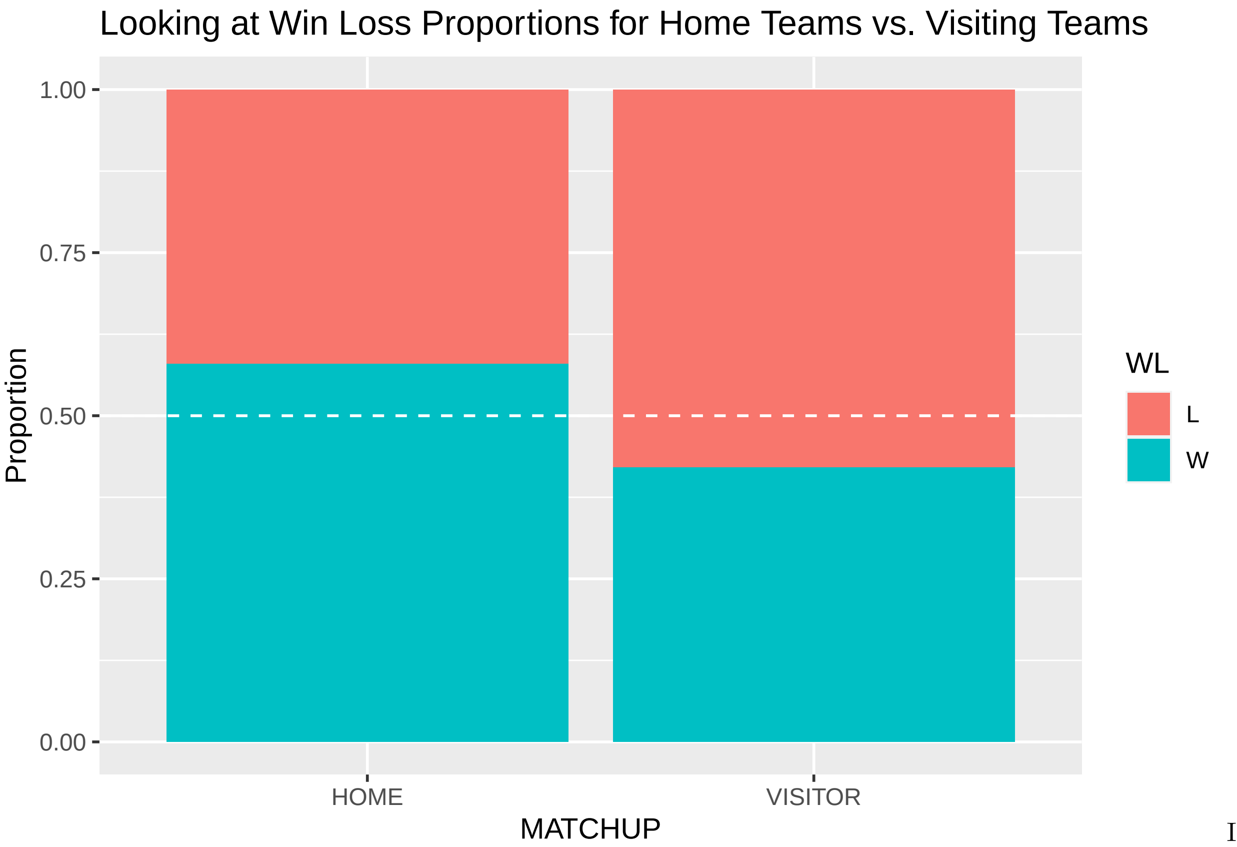
```
## # A tibble: 15,000 x 8
##    GAME_ID VISITOR_PTS VISITOR_DREB HOME_PTS HOME_DREB DIFF_DREB DIFF_PTS WINNER
##    <chr>         <dbl>        <dbl>    <dbl>     <dbl>     <dbl>    <dbl>  <dbl>
##  1 002210~         111           45       80        27       -18      -31      0
##  2 002210~         108           26      124        37        11       16      1
##  3 002210~         124           32      120        23        -9       -4      0
##  4 002210~         146           37      141        34        -3       -5      0
##  5 002210~         120           28      130        33         5       10      1
##  6 002210~         115           33      133        38         5       18      1
##  7 002210~         130           37      114        28        -9      -16      0
##  8 002210~         111           37      125        42         5       14      1
##  9 002210~         126           19      134        42        23        8      1
## 10 002210~         116           38      109        32        -6       -7      0
## # ... with 14,990 more rows
```

```
game_data %>% summarize(proportion = mean(WINNER))
```

```
## # A tibble: 1 x 1
##   proportion
##        <dbl>
## 1      0.579
```

```
wl <- games %>% mutate(WL = ifelse(WL == "W", 1, 0))
# convert back to binary for data manipulation
```

```
games %>%
  ggplot() +
  geom_bar(aes(x = MATCHUP, fill = WL), position = "fill") +
  geom_hline(data = wl, yintercept = mean(wl$WL), color = "white", linetype = "dashed") +
  ylab("Proportion") +
  ggtitle("Looking at Win Loss Proportions for Home Teams vs. Visiting Teams")
```



In this code chunk, I want to visualize the difference in home wins and visitor wins. So, I created a bar chart with the team on the x axis and proportion of wins/losses on the y axis. From this visualization, we can see that the home team wins more games than they lose while the visiting team loses more games than they win. We can also see that the home team has a higher proportion of wins than the visitng team. This then tells us that home teams have a higher winning percentage than the visiting team.

Although this visualization is not enough to prove that playing at home causes more wins, we can see that typically, a team does win more games when they are the home team. This could make sense as the basketball team has more familiarity with their own arena and they have a home crowd advantage.

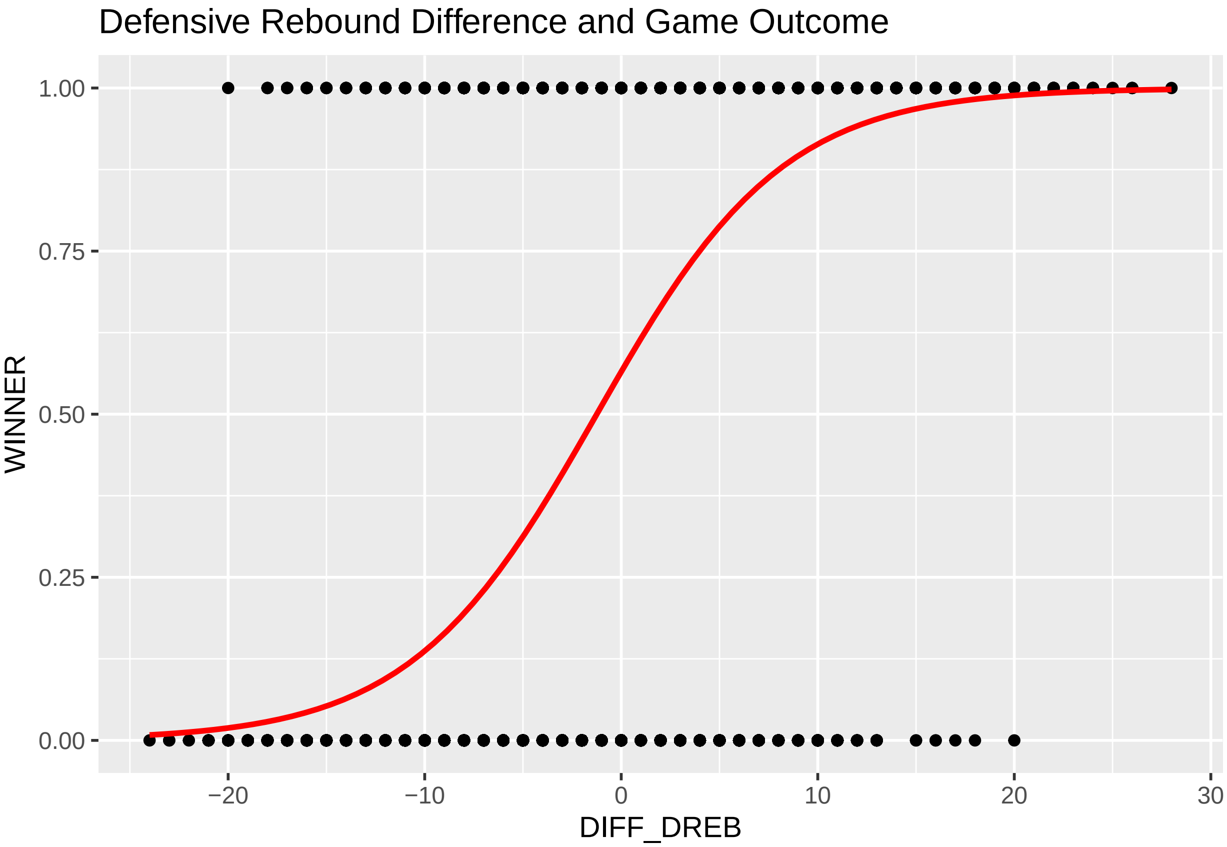
Question 2: Can looking at defensive rebounds help us predict game outcome?

Next, I wanted to look at whether the number of defensive rebounds a team gets can help us predict game outcome. Defensive rebounds are quite significant to the game of basketball. Let's say that the Tim Duncan from the Spurs obtains a defensive rebound. This play is important because it changes the possession of the ball from the other team to the Spurs, it gives the Spurs another opportunity to shoot a basket, and it prevents their opponent from having another chance at shooting. I wanted to look at this metric because it represents a part of how a team plays defensively. Many times, we often place emphasize on a player's two point, three point, or free throw shooting percentage, so I wanted to look at something that was a defensive quality.

```
model <- glm(WINNER ~ DIFF_DREB, data = game_data, family = binomial)
model
```

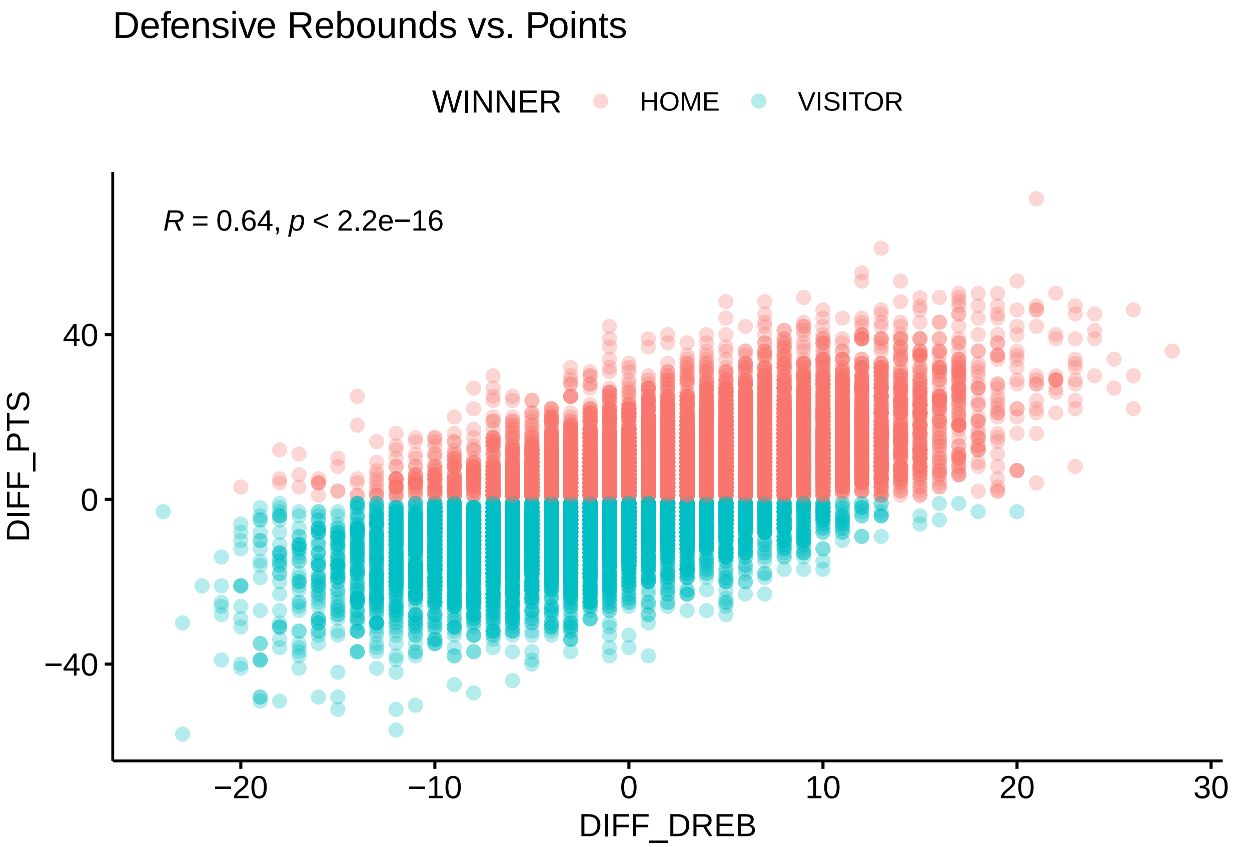
```
##
## Call:  glm(formula = WINNER ~ DIFF_DREB, family = binomial, data = game_data)
##
## Coefficients:
## (Intercept)    DIFF_DREB
##      0.2616       0.2102
##
## Degrees of Freedom: 14999 Total (i.e. Null);  14998 Residual
## Null Deviance:       20420
## Residual Deviance: 15530     AIC: 15540
```

```
game_data %>% ggplot() +
  geom_point(aes(x=`DIFF_DREB`, y = WINNER)) +
  stat_smooth(aes(x=`DIFF_DREB`, y = WINNER), color = "red", method="glm",
              se=FALSE, method.args = list(family=binomial)) +
  ggtitle("Defensive Rebound Difference and Game Outcome")
```



In this code chunk, I am creating a logistic regression model, using the difference in defensive rebounds to predict the game outcome. From this visualization, we can see that as DIFF_DREB goes negative, meaning the away team obtained more defensive rebounds, the WL goes closer to 0, signifying the away team won. As DIFF_DREB becomes positive, meaning that the home team obtained more defensive rebounds, the WL goes closer to 1, signifying the home team won. Anything above a 0.5 in this case represents a home win while anything below a 0.5 signifies an visitor win.

```
game_data %>%
  mutate(WINNER = ifelse(WINNER ==1, "HOME", "VISITOR")) %>%
  ggscatter(x="DIFF_DREB", y = "DIFF_PTS", color = "WINNER",
           title = "Defensive Rebounds vs. Points", alpha = 0.3)+
  stat_cor(method="pearson")
```



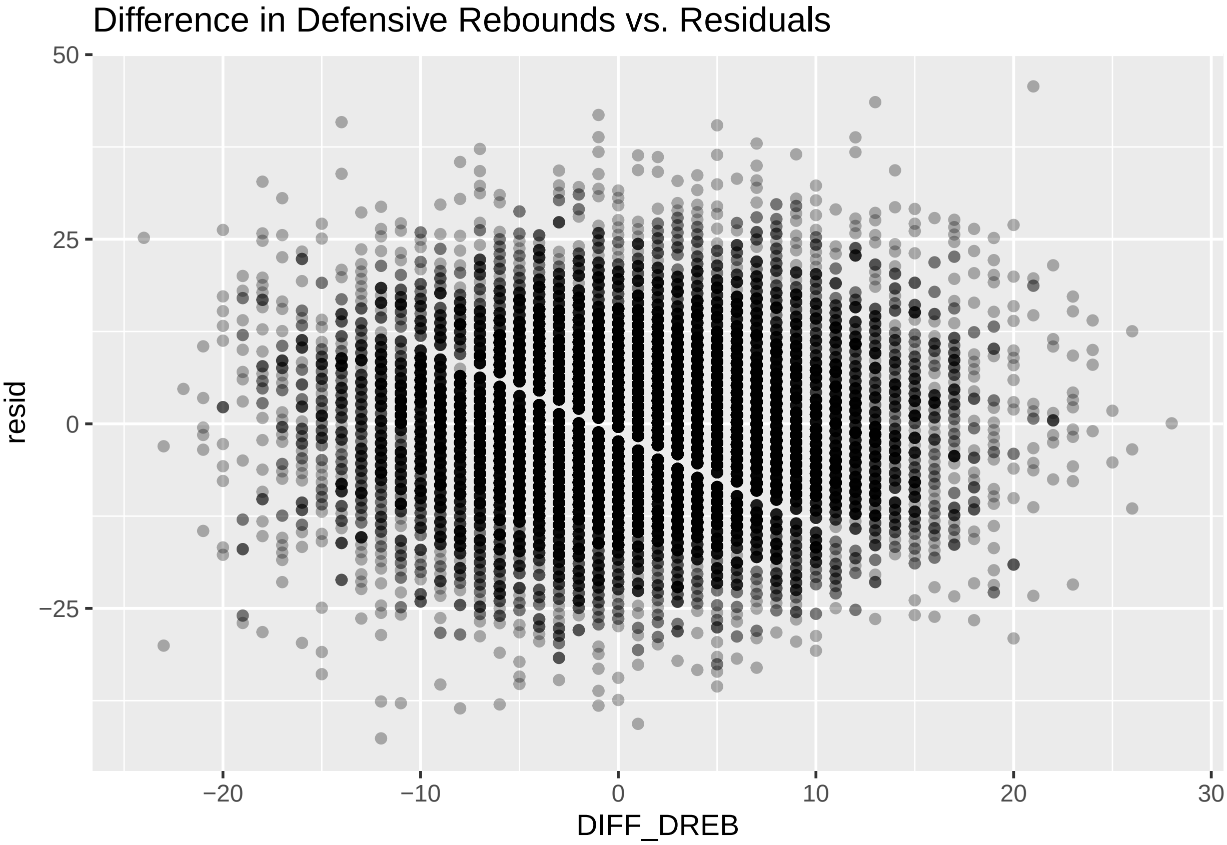
Next, I wanted to create a correlation plot to see whether there existed a linear relationship between the differences in defensive rebounds and difference in points. To do this, I changed the "Winner" column back to "Home" and "Visitor" so they could be keys in our graph, and just plotted these two variables against each other with a Pearson Correlation graph. From this visualization, we can see that there is a relatively strong positive correlation between difference in defensive rebounds and difference in points. Furthermore, our p-value is significant against an alpha level of 0.05. We can also see that evidently, when the difference in points is positive, it indicates a home team win and that when the difference in points is negative, it indicates a visitor team win.

```
linearmodel <- lm(DIFF_PTS ~ DIFF_DREB, data = game_data)
summary(linearmodel)
```

```
##
## Call:
## lm(formula = DIFF_PTS ~ DIFF_DREB, data = game_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -42.608  -7.105   0.128   6.993  45.699
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.40536    0.08756   16.05   <2e-16 ***
## DIFF_DREB    1.23310    0.01220  101.03   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 10.64 on 14998 degrees of freedom
## Multiple R-squared:  0.405,  Adjusted R-squared:  0.4049
## F-statistic: 1.021e+04 on 1 and 14998 DF,  p-value: < 2.2e-16
```

Furthermore, I wanted to create a linear model to help answer this question. For my linear model, I wanted to see whether a difference in defensive rebounds could help predict the difference in points.

```
resids <- game_data %>%
  add_residuals(linearmodel)
resids %>%
  ggplot() +
  geom_point(aes(x = DIFF_DREB, y = resid), alpha = 0.3) +
  ggtitle("Difference in Defensive Rebounds vs. Residuals")
```



To see whether this was a good model, I wanted to plot the residuals. In the above graph, we can see that there are no clear trends in the residuals plot. Additionally, the points seem to be evenly scattered within the graph, indicating that this is a relatively good linear model.

```
split <- sample.split(game_data$WINNER, SplitRatio = 0.80)
training <- subset(game_data, split == TRUE)
test <- subset(game_data, split == FALSE)
```

Although at this point, I felt that I had enough data to form a conclusion, I still wanted to test this model on a training and test set. In the following code chunk, I split my csv file into two datasets: a training set and testing set. The split ratio was 80/20, with 80% of the original data being used to train the dataset.

```
training %>% summarize(prop_home_win = mean(WINNER)) # Baseline: predict that the home team won
```

```
## # A tibble: 1 x 1
##   prop_home_win
##           <dbl>
## 1         0.579
```

```
trainingdata <- training %>%
  mutate(baselineprediction = 1) %>%
  mutate(accuracy_test = ifelse(baselineprediction == WINNER, 1, 0)) %>%
  summarize(accuracy = mean(accuracy_test))
trainingdata
```

```
## # A tibble: 1 x 1
##   accuracy
##      <dbl>
## 1    0.579
```

Now, I wanted to do a baseline prediction on the training data. Because there are more home wins than visitor wins, my baseline prediction was that the home team won. Using this baseline prediction, we can see that our training data set had an accuracy of approximately 57.91%. Now that we have our baseline model accuracy, I know that the model I am creating must have a better accuracy than 57.91% in order for me to consider it successful.

```
training %>% add_predictions(linearmodel) %>%
  mutate(pred = ifelse(pred<0, 0, 1)) %>%
  mutate(accuracy = ifelse(WINNER == pred, 1, 0)) %>%
  summarize(mod_accuracy = mean(accuracy))
```

```
## # A tibble: 1 x 1
##   mod_accuracy
##          <dbl>
## 1        0.742
```

Now, I want to observe how accurate my linear model I created is in predicting the winner on the training data. Because our dependent variable is difference in points, we know that if the predicted value for difference in points is negative, then the visiting team won and vice versa. Therefore, I am creating a "pred" column that puts a "0" for if this value is less than 0 and a "1" if it is greater than or equal to zero. One caveat here is that if the difference in scores is 0 exactly, it is still predicting it as a home win.

Then, with these predictions, I want to see whether this aligned with the actual result of the game. So, I wanted to test whether the winner and predicted winner were the same. We can see that this linear model has an accuracy that is higher than our baseline model.

```
test %>% add_predictions(linearmodel) %>%
  mutate(pred = ifelse(pred<0, 0, 1)) %>%
  mutate(accuracy = ifelse(WINNER == pred, 1, 0)) %>%
  summarize(mod_accuracy = mean(accuracy))
```

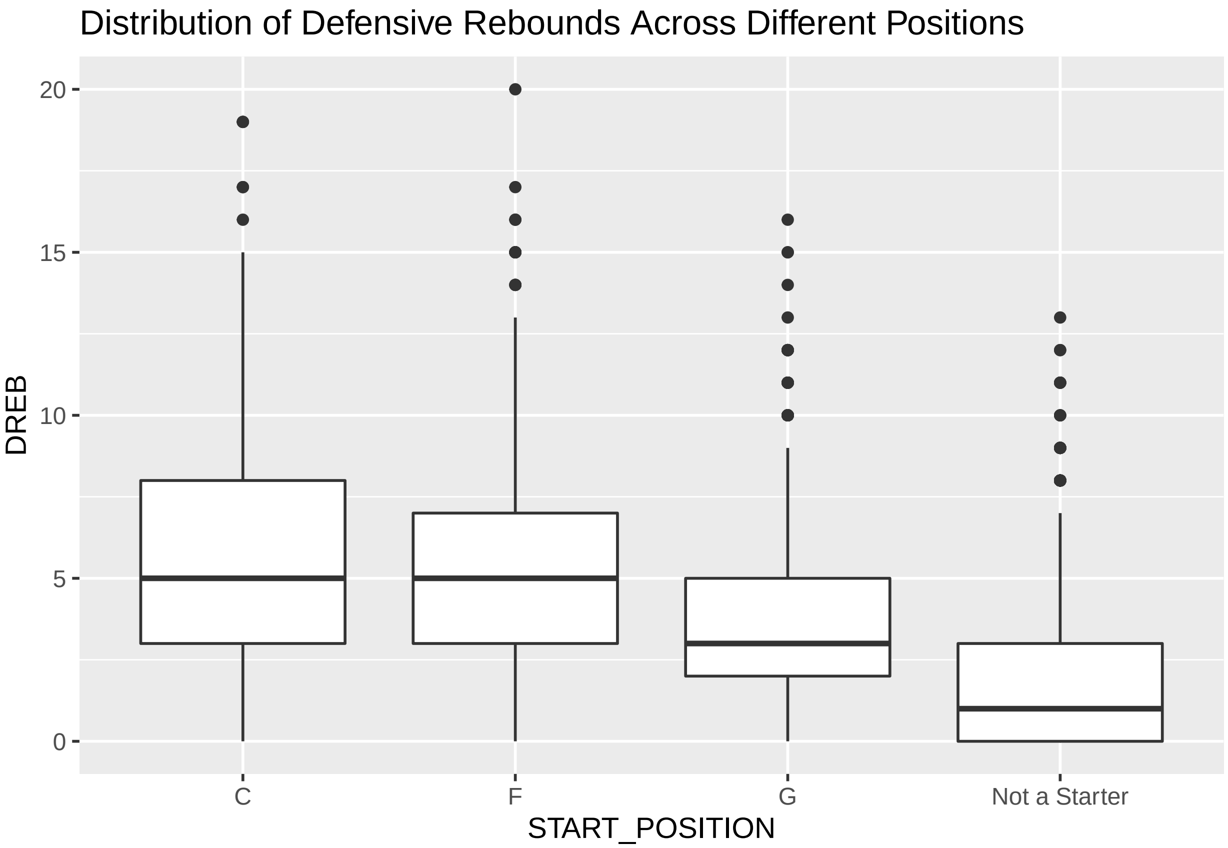
```
## # A tibble: 1 x 1
##   mod_accuracy
##          <dbl>
## 1        0.749
```

Now, I want to test the model's accuracy on the testing data set. We can see that the model is less accurate on this data set than the training data set. However, I thought these results were still important because the overall model accuracy is much more accurate than our baseline model. Additionally, this model doesn't seem to demonstrate overfitting as the accuracy for the training and testing data sets are quite close.

Question 3: Does the distribution of possessions and defensive plays change for different positions? (Playoffs 2016-2022)

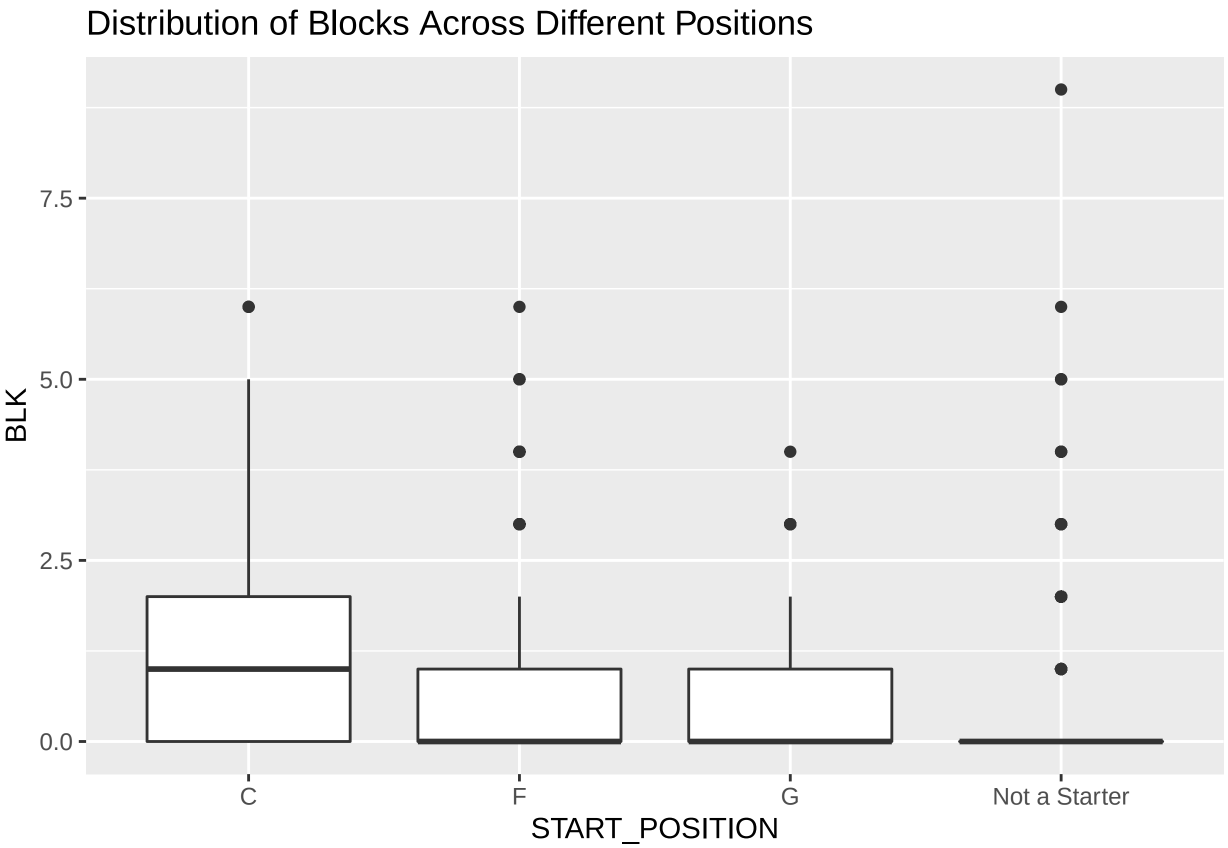
For this next question, I wanted to look at certain starting positions have more possessions or defensive plays. I considered defensive rebounds, blocks, and steals as “defensive plays”. The motivation behind this question is understanding that each position on the field plays a different purposes. For example, the center is typically the tallest player who gets rebounds or the tipoff. In the next three plots, I first wanted to observe the distribution of each variable that I included in the overall “defensive plays” category.

```
positions %>%
  ggplot() +
  geom_boxplot(aes(x=START_POSITION, y = DREB)) +
  ggtitle("Distribution of Defensive Rebounds Across Different Positions")
```



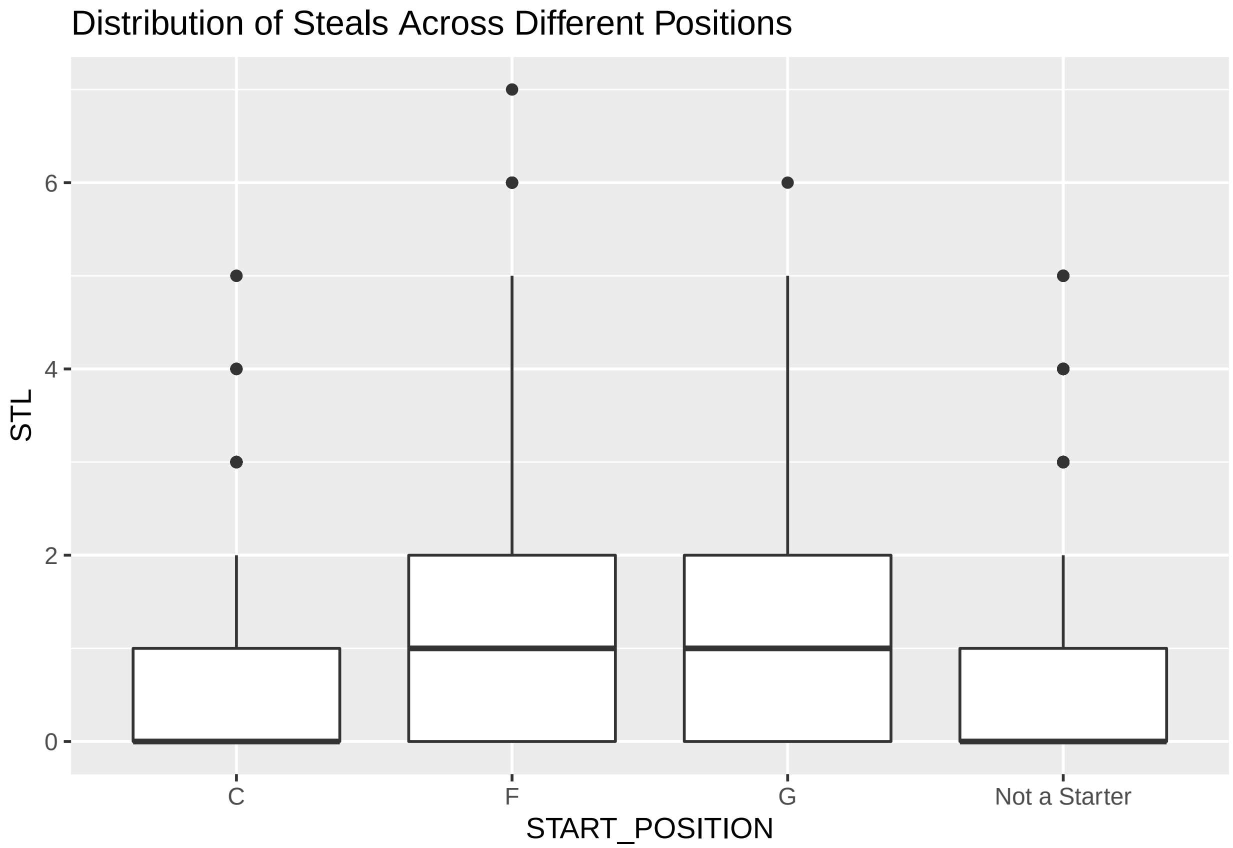
In the code chunk above, I wanted to generate a box plot that describes each starting positions distribution of defensive rebounds. From this boxplot, we can see how Centers typically have the most defensive rebounds because it has the highest interquartile range and both the first and third quartiles are either equal to or higher than those of other positions.

```
positions %>%
  ggplot() +
  geom_boxplot(aes(x=START_POSITION, y = BLK)) +
  ggtitle("Distribution of Blocks Across Different Positions")
```



In this code chunk above, I generated box plots that describe the distribution of blocks for each starting position. From this visualization, we can also see that Centers typically have the most blocks as their median and third quartile are higher than that of all of the other positions.

```
positions %>%
  ggplot() +
  geom_boxplot(aes(x=START_POSITION, y = STL)) +
  ggtitle("Distribution of Steals Across Different Positions")
```



Finally, I looked at the distribution of steals for each start position. Unlike the other two variables, blocks and defensive rebounds, Centers perform worse in this category than other starting positions.

```
playsmod <- lm (DEFENSIVE_PLAYS ~ START_POSITION, data = positions)
playsmod
```

```
##
## Call:
## lm(formula = DEFENSIVE_PLAYS ~ START_POSITION, data = positions)
##
## Coefficients:
##                 (Intercept)              START_POSITIONF
##                      7.6811                      -0.8231
##             START_POSITIONG  START_POSITIONNot a Starter
##                     -2.5227                      -5.3148
```

With this information, I wanted to create a linear model that uses starting position to predict number of defensive plays. Our linear model equation is the following:

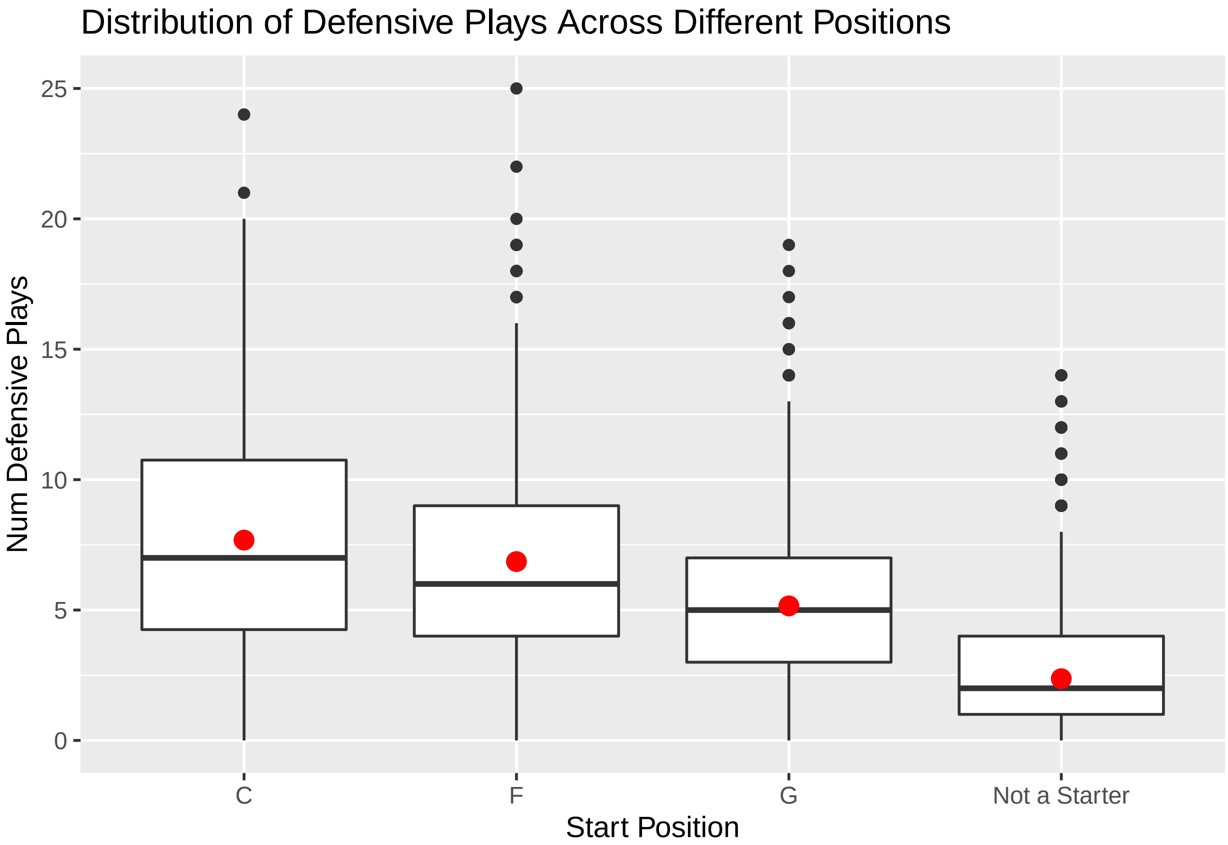
defensive plays = 7.6811 - 0.8231 * forward - -2.5227 * guard - 5.3148 * notStarter

```
pospred <- positions %>% data_grid(START_POSITION) %>%
  add_predictions(playsmod)
pospred
```

```
## # A tibble: 4 x 2
##   START_POSITION  pred
##   <chr>          <dbl>
## 1 C               7.68
## 2 F                 6.86
## 3 G                 5.16
## 4 Not a Starter     2.37
```

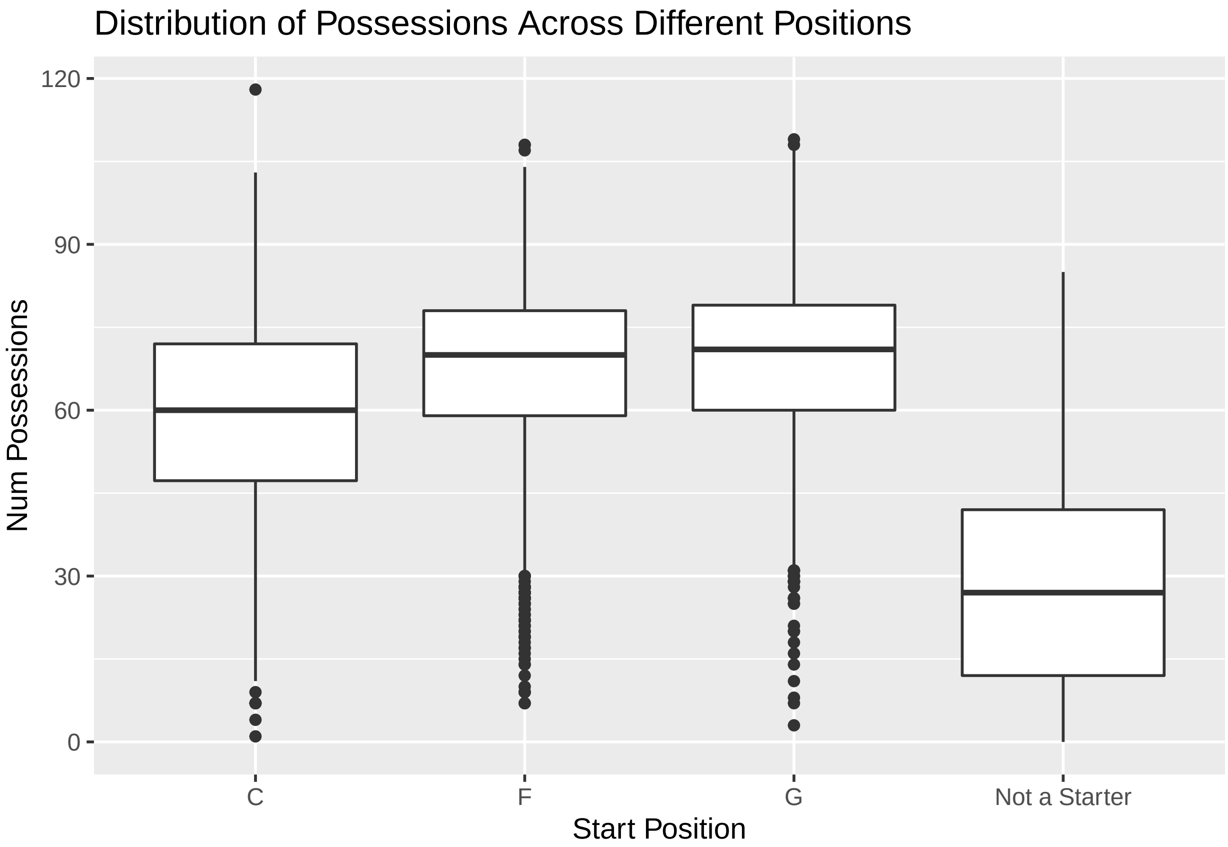
Now, I wanted to create a prediction of number of defensive plays for each prediction. In the next code chunk, I will use this new information to plot the predicted values.

```
positions %>%
  ggplot() +
  geom_boxplot(aes(x=START_POSITION, y = DEFENSIVE_PLAYS)) +
  geom_point(data = pospred, aes(x = START_POSITION, y = pred), color = "red", size = 3) +
  ggtitle("Distribution of Defensive Plays Across Different Positions") +
  ylab("Num Defensive Plays") +
  xlab("Start Position")
```



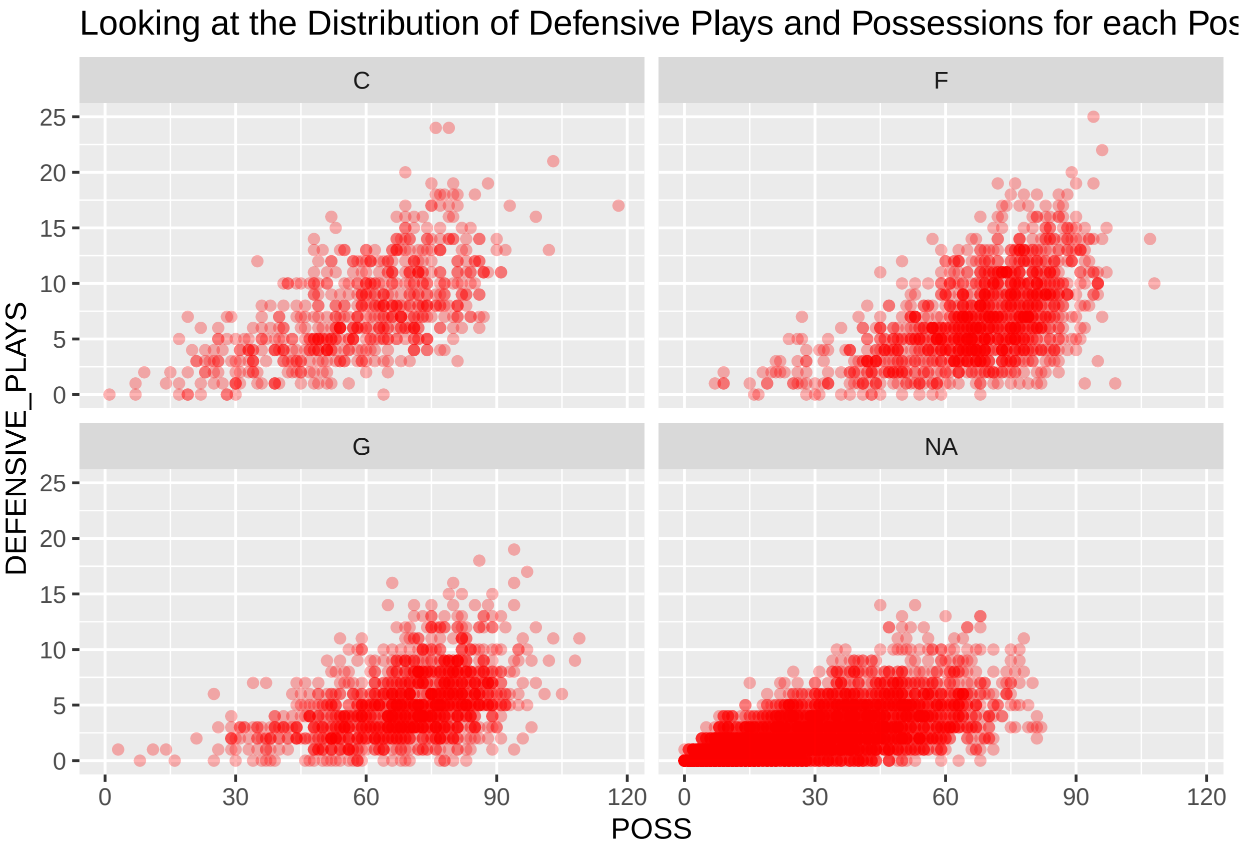
The code chunk above now looks as defensive plays as a whole, summing up the number of blocks, steals, and defensive rebounds. I also plotted the predicted values for each position to see how good this linear model was. We can see that our predictions are quite close to the actual values. From this visualization, we can see that Centers typically have the most number of defensive plays, followed by Forwards, Guards, then players who didn't start in a game.

```
positions %>%
  ggplot() +
  geom_boxplot(aes(x=START_POSITION, y = POSS)) +
  ggtitle("Distribution of Possessions Across Different Positions") +
  ylab("Num Possessions") +
  xlab("Start Position")
```



I then wanted to transition into looking at the distribution of possessions across different positions to see whether certain positions had more control on the ball. The visualization is interesting here because we see that Forwards and Guards generally have more possessions than Centers. Yet, Centers tend to have more defensive plays, which could hint that they are playing defense quite efficiently.

```
boxscores %>%
  ggplot() +
  geom_point(aes(x=POSS, y = DEFENSIVE_PLAYS), alpha = 0.3, color = "red") +
  facet_wrap(~START_POSITION) +
  ggtitle("Looking at the Distribution of Defensive Plays and Possessions for each Position")
```



In this last graph, I wanted to create a scatter plot that looks at possessions against defensive plays for each start position. I did this so we could answer the original question of seeing whether there was a difference in the distribution for possessions and defensive plays in different starting positions. From this visualization, we can see that all of the graphs tend to have a positive correlation. Just by looking at this graph, we can see that there is some sort of positive correlation between number of possessions and defensive plays. We can see that Centers seem to display the strongest positive correlation. To answer the question, I believe that the distribution of possessions and defensive plays does change depending on starting position. Additionally, it seems that within the category of defensive plays, blocks, defensive rebounds, and steals all have different distributions based on the player position.

Question 4: What is the most common reason for turnovers in the 2021-2022 Playoff season so far?

The last variable I wanted to look at were turnovers in the 2021-2022 Playpff season, which is currently in session. Turnovers are when a team loses possession of the ball from something other than a missed shot. I will be looking at ten different types of turnovers: bad passes, lost ball (opponent steals the ball), second violation, foul, shot clock, out of bounds, traveling (player with possession of ball takes too many steps in between dribbling), goaltending (bad timing of a block), and backcourt (team enters backside of court again after crossing midcourt line). Turnovers can be crucial to the outcome of a game, especially when there are only a few seconds left in a game.

```
tovreasons <- c("Bad Pass", "Lost Ball", "Second Violation", "Foul",
                "Shot Clock", "Out of Bounds", "Traveling",
                "Goaltending", "Backcourt")
tovregex <- str_c(tovreasons, collapse = "|")
```

First, I wanted to store all of the turnover types into a list then create a regular expression. With this regular expression, I want to see whether any cells in the HOMEDESCRIPTION column contain these types of turnovers.

```
home <- playbyplay %>%
  mutate(turnover_found = str_extract(HOMEDESCRIPTION, "Turnover")) %>%
  filter(!is.na(turnover_found)) %>%
  select(HOMEDESCRIPTION, turnover_found) %>%
  rename(DESCRIPTION = HOMEDESCRIPTION)
hometov <- home %>%
  mutate(turnover_found = str_extract(DESCRIPTION, tovregex), team = "home")
```

In the code chunk above, I first want to extract plays of where a turnover occurred. Then, I use the regular expression I created above to see which type of turnover occurred.

```
visit<- playbyplay %>%
  mutate(turnover_found = str_extract(VISITORDESCRIPTION, "Turnover")) %>%
  filter(!is.na(turnover_found)) %>%
  select(VISITORDESCRIPTION, turnover_found) %>%
  rename(DESCRIPTION = VISITORDESCRIPTION)
visittov <- visit %>% mutate(turnover_found = str_extract(DESCRIPTION, tovregex), team = "visitor")
```

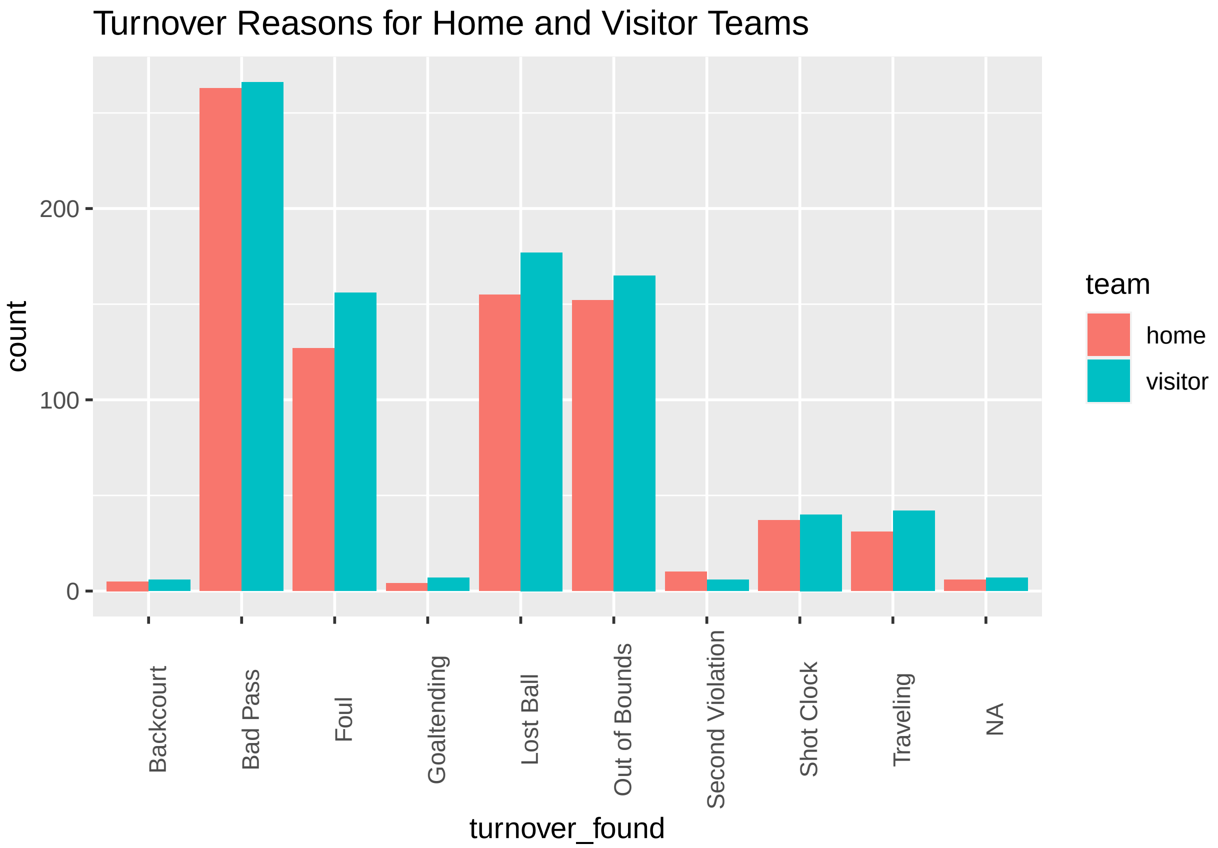
This code chunk is the same as the previous one, but instead of looking at the home team turnovers, I'm looking at visitor team turnovers. The reason I have to split it as home and away is because HOMEDESCRIPTION and VISITORDESCRIPTION are two separate column. If I only look at one column, I'm not looking at all of the turnovers that occurred within a game.

```
turnovers <- union(hometov, visittov)
turnovers
```

```
## # A tibble: 1,662 x 3
##    DESCRIPTION                                          turnover_found team
##    <chr>                                                <chr>          <chr>
##  1 Smart Out of Bounds - Bad Pass Turnover Turnover (P1.T1) Out of Bounds  home
##  2 Horford Out of Bounds - Bad Pass Turnover Turnover (P1.~ Out of Bounds  home
##  3 Williams Out of Bounds Lost Ball Turnover (P1.T3)        Out of Bounds  home
##  4 Tatum Out of Bounds Lost Ball Turnover (P1.T4)           Out of Bounds  home
##  5 Brown Lost Ball Turnover (P1.T5)                         Lost Ball      home
##  6 Williams Lost Ball Turnover (P2.T6)                      Lost Ball      home
##  7 Brown Out of Bounds - Bad Pass Turnover Turnover (P2.T7) Out of Bounds  home
##  8 White Bad Pass Turnover (P1.T8)                          Bad Pass       home
##  9 Smart Lost Ball Turnover (P2.T9)                         Lost Ball      home
## 10 Smart Lost Ball Turnover (P3.T10)                        Lost Ball      home
## # ... with 1,652 more rows
```

Next, I just want to create a union of the two tibbles I created. This tibble contains three columns: DESCRIPTION, turnover_found, and team. Now, I have all of the data on all of the turnovers that occurred in one game for both the home and away team.

```
turnovers %>%
  ggplot() +
  geom_bar(aes(x = turnover_found, fill = team), position = "dodge") +
  theme(axis.text.x = element_text(angle=90)) +
  ggtitle("Turnover Reasons for Home and Visitor Teams")
```



Lastly, I just wanted to visualize the different types of turnovers. I created a bar plot that examines different types of turnovers made for the home and visitor team. From this visualization, we can see that bad passes are the primary causes of turnovers, followed by a lost ball, then going out of bounds. You might notice that there is an NA value in this graph, and this is because there are other types of turnovers that I did not state in my regular expression. This NA value represents all of those other types of turnovers. From this visualization, I conclude that the answer to my question of what the most common type of turnovers in the 2021-22 Playoff season so far is bad passes.

Conclusion

Limitations

There are a few limitations to these analyses. Firstly, because I am webscraping this data from the NBA website, when a team is currently playing a game, the resulting CSV file shows NA for all of the columns corresponding to the current game id. This is because the NBA website is dynamic: it is updating as games are in play. However, scraping the data only obtains the data in one static state. Furthermore, the game of basketball is continuously changing, and depending on the team or player, some of these variables might not be as relevant to them. Ultimately, coaches decide their team's strategy and game play, and that could look different across the board. Additionally, some of these visualizations only depended on Playoff data, which would not be generalization to Regular Season or Pre Season. I also did not look at any Pre Season data, so the analyses conducted can not be assumed to be generalized to Pre Season games as well.

Next Steps

Some next steps I would want to take would be to collect more data as more games occur. Because I didn't look at Pre Season data, I could see whether the models I found are applicable to Pre Season as well. I would also like to look at what locations on a court turnovers occur most frequently, if that data is publicly available. I would want to see if there are certain teams that might get much fewer defensive rebounds than their opponent yet still manage to win a majority of their games. I would also like to look into why exactly there could be a home court advantage because there could be many factors that play a role in this.