Data Analysis

Pipe It Up!: Nagaprasad Rudrapatna, Karen Deng, Jackson Muraika, Anna Zolotor

2020-04-16

```
library(tidyverse)
library(broom)
library(stringr)
library(knitr)
library(tidyverse)
library(gridExtra)
library(leaps)
library(rms)
```

```
nba_social_power <- read_csv("../data/nba.csv")

nba_2016_2017_100 <- nba_social_power

nba_social_power_mod <- nba_social_power %>%
  filter(TWITTER_HANDLE != "0") %>%
  select(PLAYER_NAME,
         TEAM_ABBREVIATION,
         AGE,
         W_PCT,
         OFF_RATING,
         DEF_RATING,
         NET_RATING,
         AST_RATIO,
         REB_PCT,
         USG_PCT,
         PIE,
         SALARY_MILLIONS,
         ACTIVE_TWITTER_LAST_YEAR,
         TWITTER_FOLLOWER_COUNT_MILLIONS,
         PTS)
```

Research Question and Objective

The research question explored in this analysis is: `Is there a relationship between measures of athletic success (win percentage, offensive and defensive ratings, etc.) and the internet "popularity" of NBA athletes, measured in the number of Twitter followers?`

The objective of this analysis is to predict Twitter follower counts of NBA players using measures of athletic success.

The Data

The dataset we use in this analysis includes on-court performance data for NBA players in the 2016-2017 season, along with their salaries and Twitter engagement. Because we are examining the relationship between player stats and the number of Twitter followers, we filtered for players who had an active Twitter account, by filtering for values where `TWITTER_HANDLE` is not "NA" (0). After filtering, we have 95 observations.

The response variable is `TWITTER_FOLLOWER_COUNT_MILLIONS`, which measures players' Twitter follower counts at the time the data was collected.

Exploratory Data Analysis

Univariate

First, we will do univariate EDA on the dataset. Player name will be used to refer to observations in our dataset, but since each player name is distinct we do not need to do EDA on the `PLAYER_NAME` variable.

Here, we'll take a look at how many players there are from each team in the dataset:

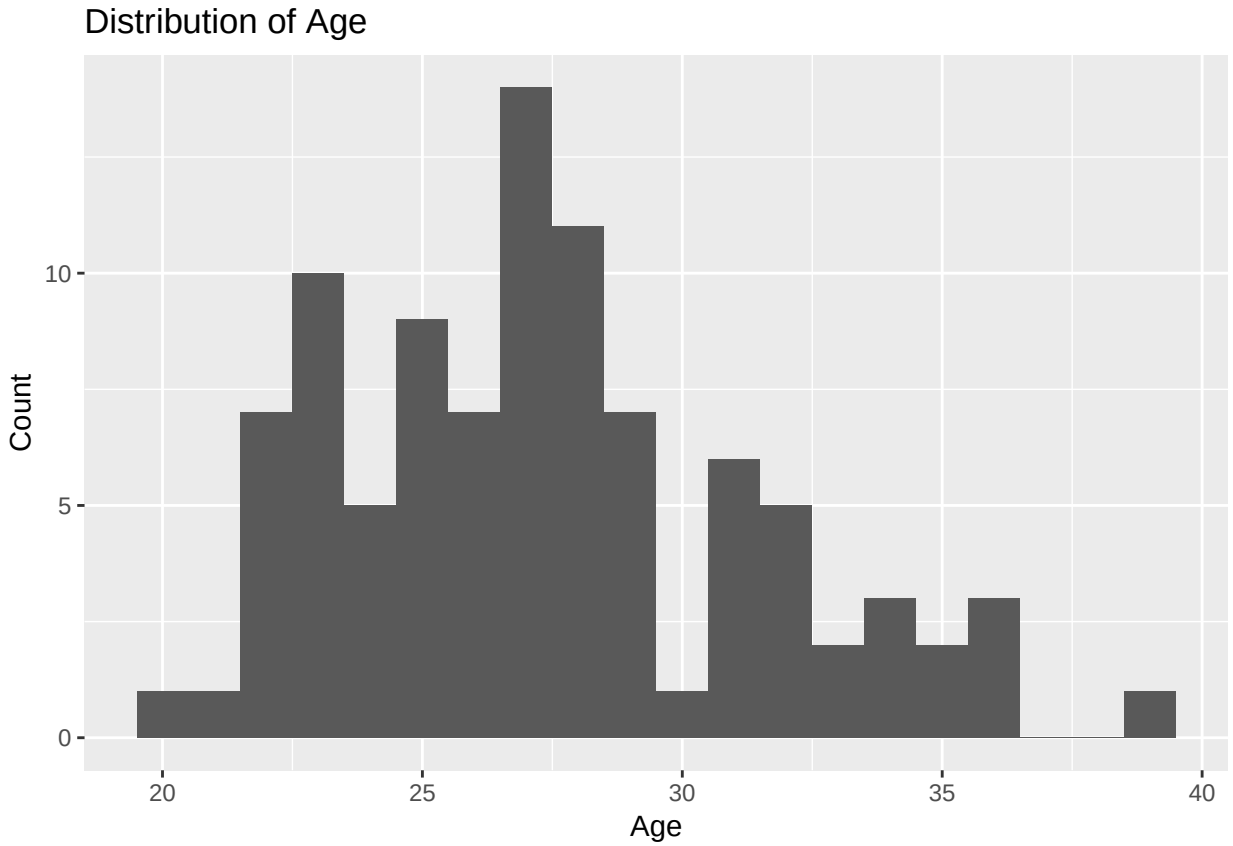
```
nba_social_power_mod %>%
  count(TEAM_ABBREVIATION) %>%
  arrange(n)
```

```
## # A tibble: 30 x 2
##    TEAM_ABBREVIATION     n
##    <chr>             <int>
##  1 SAC                   1
##  2 CHI                   2
##  3 IND                   2
##  4 LAL                   2
##  5 MIA                   2
##  6 MIN                   2
##  7 ORL                   2
##  8 WAS                   2
##  9 ATL                   3
## 10 BKN                   3
## # ... with 20 more rows
```

As we can see from the output, there is only one team that is represented just once in the dataset: SAC, the Sacramento Kings. The greatest number of times teams are represented in the dataset is 5. GSW (Golden State Warriors), LAC (Los Angeles Clippers), and SAS (San Antonio Spurs) are all represented 5 times.

Now, we'll explore the distribution of the `AGE` variable in the dataset:

```
ggplot(data = nba_social_power_mod, aes(x= AGE)) +
  geom_histogram(binwidth = 1) +
  labs(x = "Age", y = "Count", title = "Distribution of Age")
```




```
nba_social_power_mod %>%
  summarise(mean = mean(AGE), min= min(AGE), Q1 = quantile(AGE, .25), median = median(AGE),
            Q3 = quantile(AGE, .75),
            max = max(AGE))
```

```
## # A tibble: 1 x 6
##    mean   min    Q1 median    Q3   max
##   <dbl> <dbl> <dbl>  <int> <dbl> <dbl>
## 1  27.4    20  24.5     27    29    39
```

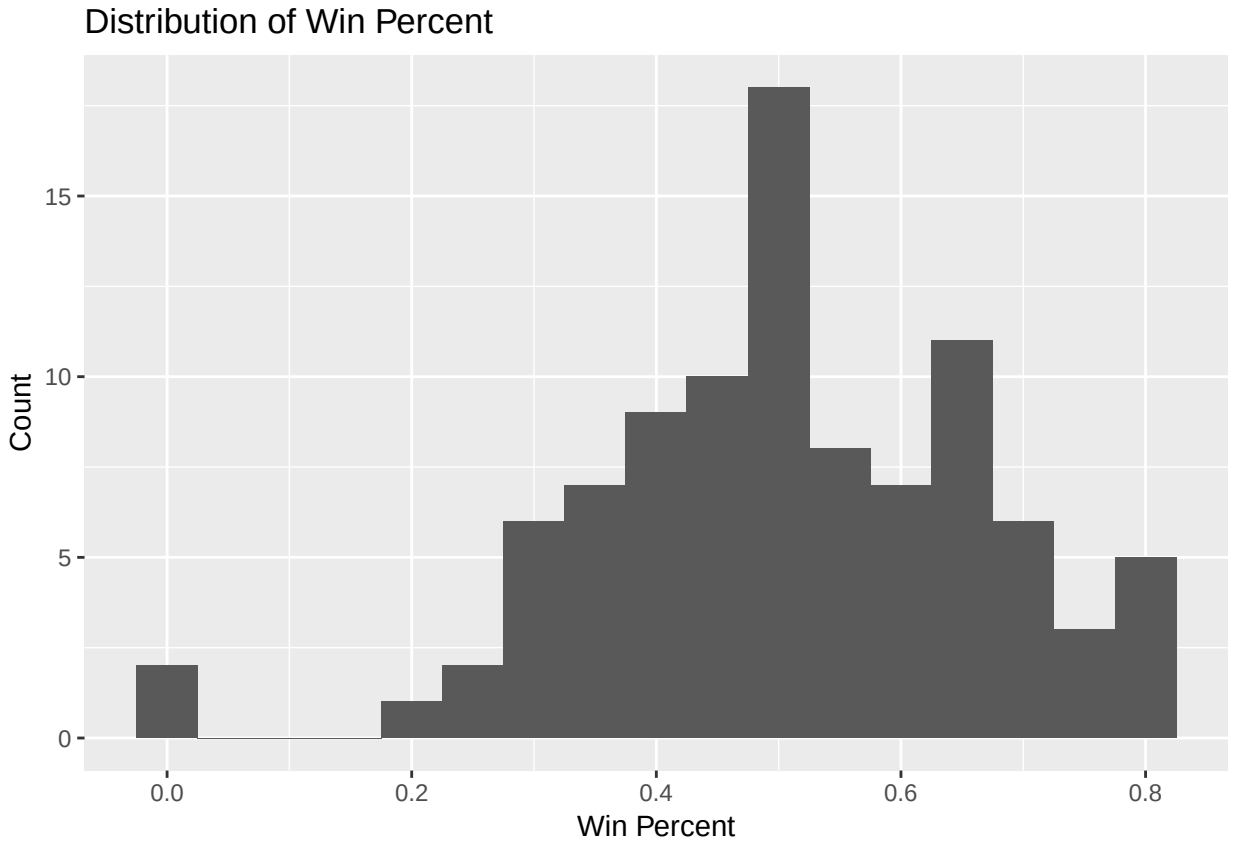
```
nba_social_power_mod %>%
  arrange(desc(AGE)) %>%
  head(1)
```

```
## # A tibble: 1 x 15
##   PLAYER_NAME TEAM_ABBREVIATI~   AGE W_PCT OFF_RATING DEF_RATING NET_RATING
##   <chr>       <chr>            <int> <dbl>      <dbl>      <dbl>      <dbl>
## 1 Dirk Nowit~ DAL                 39 0.426       105.       106.       -1.7
## # ... with 8 more variables: AST_RATIO <dbl>, REB_PCT <dbl>,
## #   USG_PCT <dbl>, PIE <dbl>, SALARY_MILLIONS <dbl>,
## #   ACTIVE_TWITTER_LAST_YEAR <int>, TWITTER_FOLLOWER_COUNT_MILLIONS <dbl>,
## #   PTS <dbl>
```

As we can see from the histogram, age is somewhat normally distributed in the dataset, with a mode around 27 and a surprisingly low number of 30-year olds. The mean age, 27.39, and median age, 27, are very close together, indicating little skew. The lowest age is 20 and the highest is 39. The oldest player by far, at 39, is Dirk Nowitzki.

Now, we'll examine the distribution of win percent, `W_PCT`:

```
ggplot(data = nba_social_power_mod, aes(x= W_PCT)) +
  geom_histogram(binwidth = .05) +
  labs(x = "Win Percent", y = "Count", title = "Distribution of Win Percent")
```



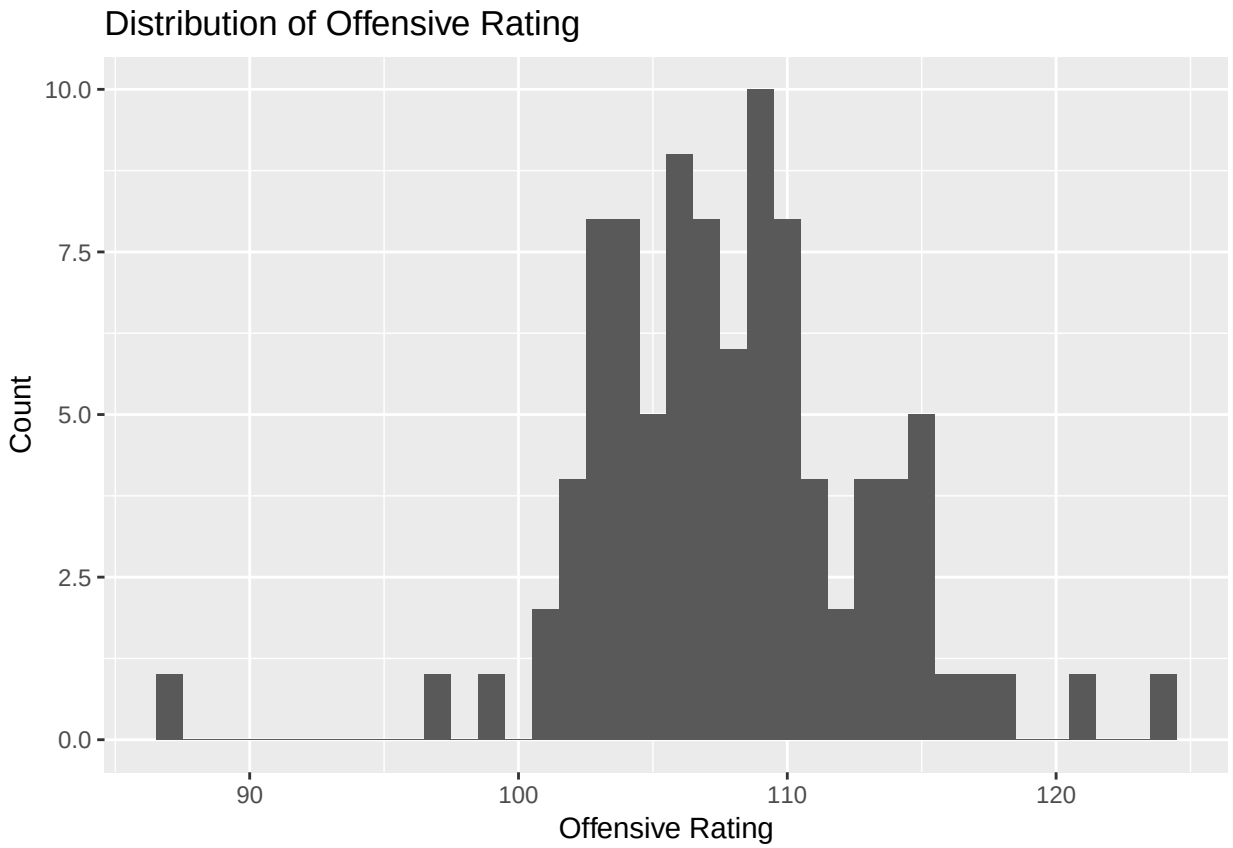

```
nba_social_power_mod %>%
  summarise(mean = mean(W_PCT), min= min(W_PCT), Q1 = quantile(W_PCT, .25),
            median = median(W_PCT), Q3 = quantile(W_PCT, .75),
            max = max(W_PCT))
```

```
## # A tibble: 1 x 6
##    mean   min    Q1 median    Q3   max
##   <dbl> <dbl> <dbl>  <dbl> <dbl> <dbl>
## 1 0.511     0 0.418  0.507  0.63 0.824
```

As we can see from the histogram, win percent is also somewhat normally distributed, with a mode around 50 percent. The minimum win percent in the dataset is 0, while the maximum is 82.4. The median of 50.7 is very similar to the mean of 51%. The fact that the mean and median win percents in the dataset fall so close to 50% indicate good randomness in the dataset, b/c the mean and median win percents for all nba players are 50%.

Next, we'll look at the distributions for offensive rating, `OFF_RATING` and defensive rating `DEF_RATING`, as well as the distribution for `NET_RATING`, which is the average of the offensive and defensive rating:

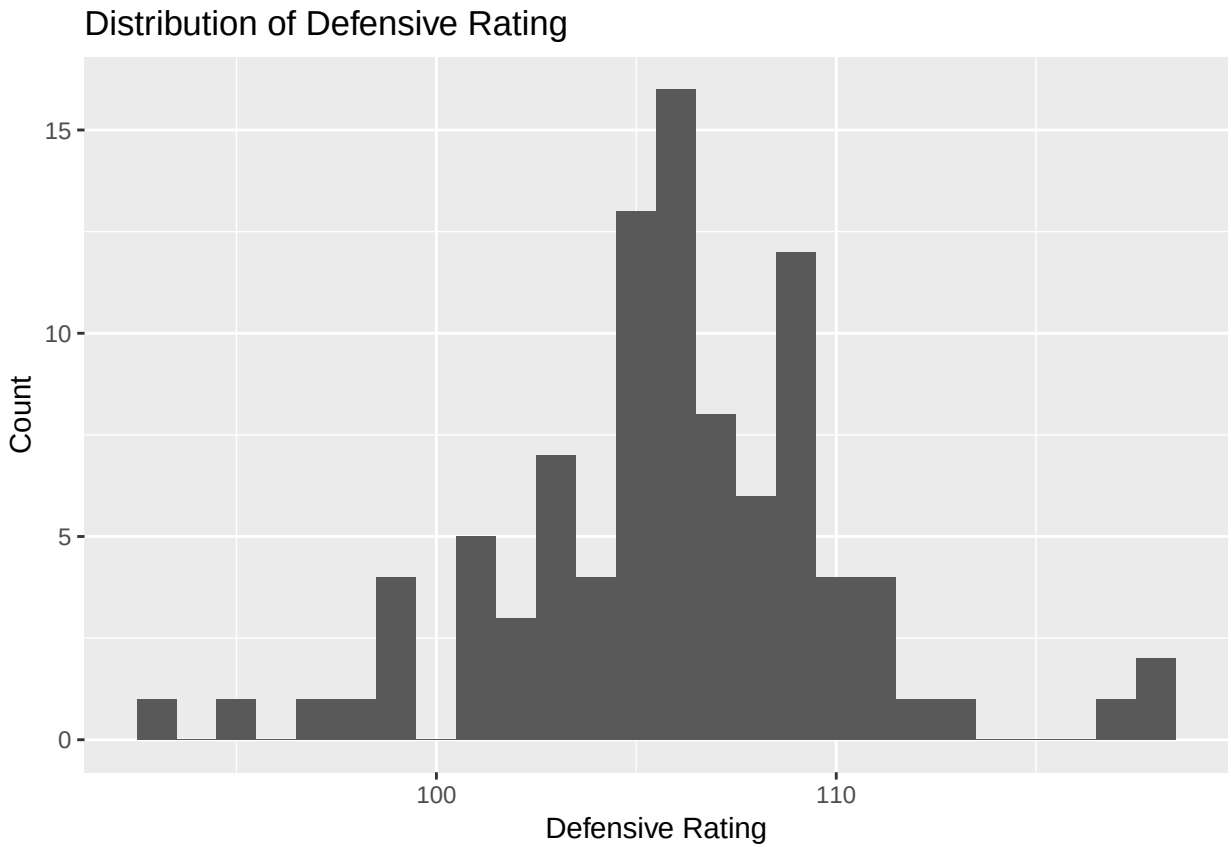
```
ggplot(data = nba_social_power_mod, aes(x= OFF_RATING)) +
  geom_histogram(binwidth = 1) +
  labs(x = "Offensive Rating", y = "Count",
       title = "Distribution of Offensive Rating")
```




```
nba_social_power_mod %>%
  summarise(min= min(OFF_RATING), median = median(OFF_RATING),
            max = max(OFF_RATING))
```

```
## # A tibble: 1 x 3
##     min median   max
##   <dbl>  <dbl> <dbl>
## 1  86.8   108.  124.
```

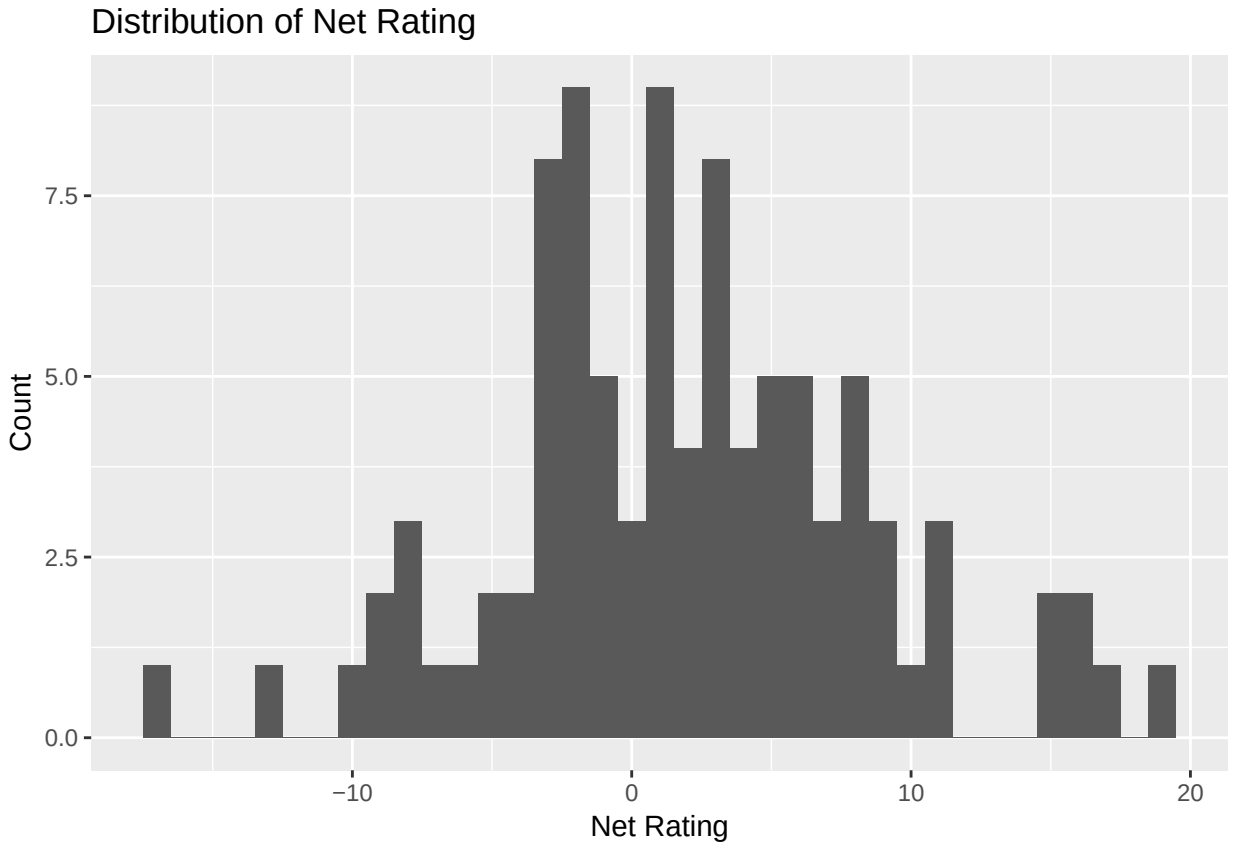
```
ggplot(data = nba_social_power_mod, aes(x= DEF_RATING)) +
  geom_histogram(binwidth = 1) +
  labs(x = "Defensive Rating", y = "Count",
       title = "Distribution of Defensive Rating")
```




```
nba_social_power_mod %>%
  summarise(min= min(DEF_RATING), median = median(DEF_RATING),
            max = max(DEF_RATING))
```

```
## # A tibble: 1 x 3
##     min median   max
##   <dbl>  <dbl> <dbl>
## 1    93    106  118.
```

```
ggplot(data = nba_social_power_mod, aes(x= NET_RATING)) +
  geom_histogram(binwidth = 1) +
  labs(x = "Net Rating", y = "Count",
       title = "Distribution of Net Rating")
```



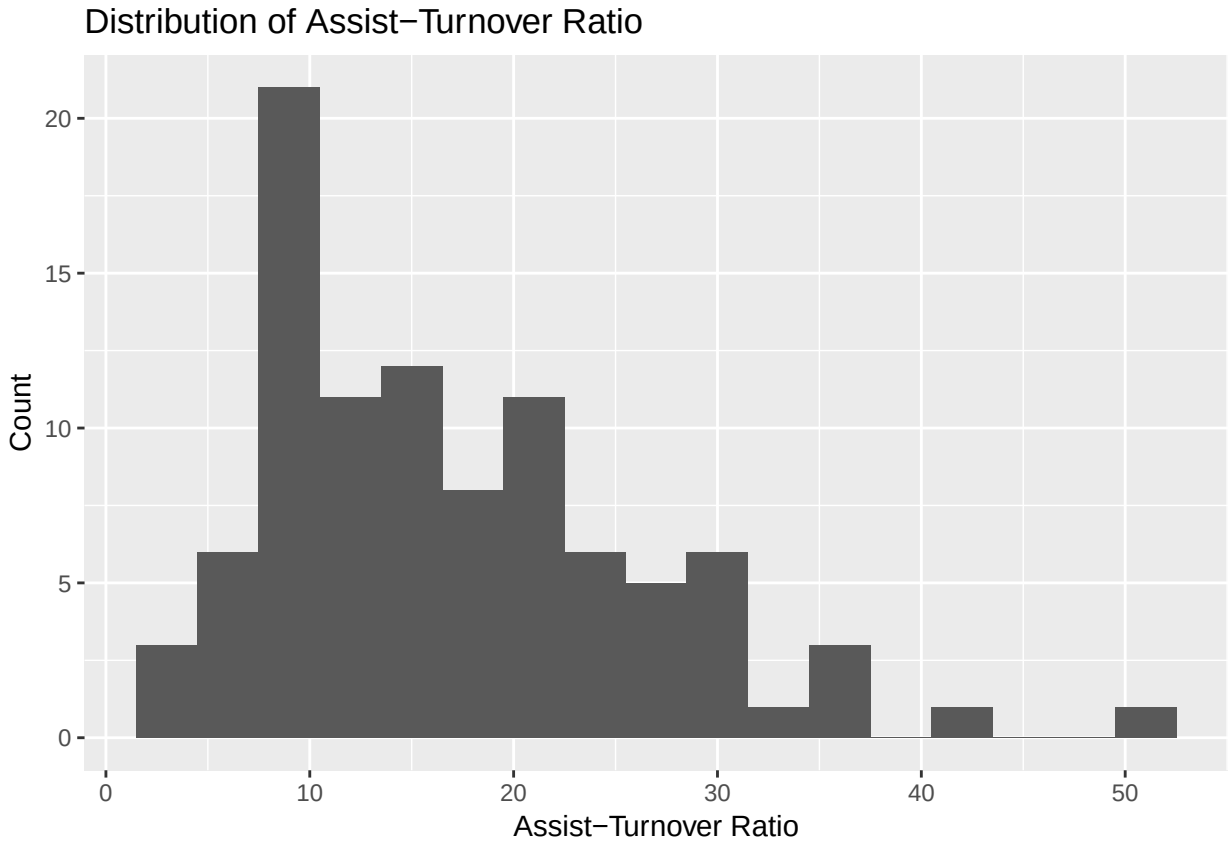

```
nba_social_power_mod %>%
  summarise(min= min(NET_RATING), median = median(NET_RATING),
            max = max(NET_RATING))
```

```
## # A tibble: 1 x 3
##     min median   max
##   <dbl>  <dbl> <dbl>
## 1 -17.2    1.5  18.7
```

Defensive rating, offensive rating, and net rating do not stray far from normally distributed. Offensive rating varies from 86.8 to 124.2, with a median of 107.6. Defensive rating varies from 93 to 118.3, with a median of 106. Thus, the dataset contains a larger range in terms of offensive rating, and the median is also slightly higher for defensive rated players. The distribution of net rating has multiple nearly equal modes around -2 to -3 and around 1 and 3. The median net rating is 1.5, and the net ratings in the dataset vary from -17.2 to 18.7.

Next, we'll look at the distribution of the assist-to-turnovers ratio, `AST_RATIO`:

```
ggplot(data = nba_social_power_mod, aes(x= AST_RATIO)) +
  geom_histogram(binwidth = 3) +
  labs(x = "Assist-Turnover Ratio", y = "Count",
       title = "Distribution of Assist-Turnover Ratio")
```




```
nba_social_power_mod %>%
  summarise(mean = mean(AST_RATIO),
            min= min(AST_RATIO),
            Q1 = quantile(AST_RATIO, .25),
            median = median(AST_RATIO),
            Q3 = quantile(AST_RATIO, .75),
            max = max(AST_RATIO))
```

```
## # A tibble: 1 x 6
##    mean   min    Q1 median    Q3   max
##   <dbl> <dbl> <dbl>  <dbl> <dbl> <dbl>
## 1  17.1     4  9.75     15  22.2  51.5
```

```
nba_social_power_mod %>%
  arrange(desc(AST_RATIO)) %>%
  head(2)
```

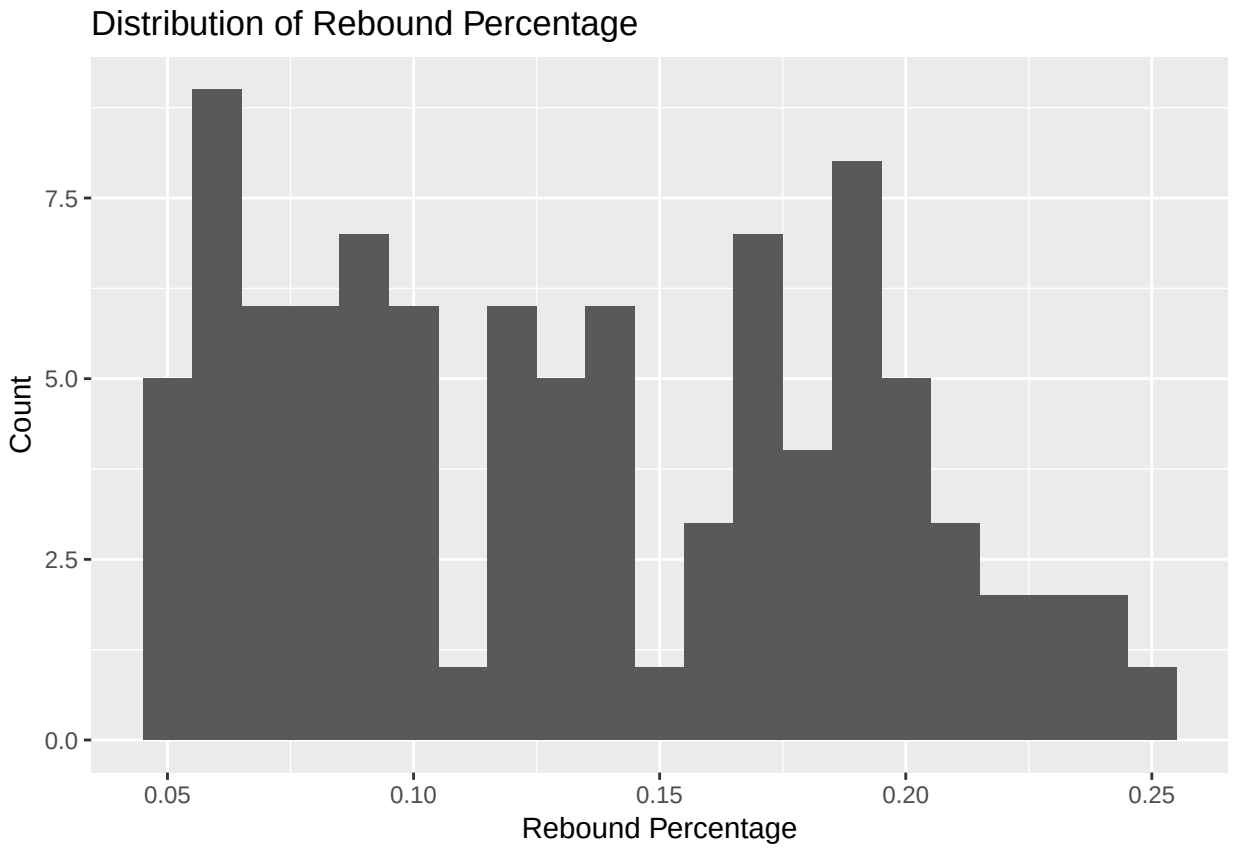
```
## # A tibble: 2 x 15
##   PLAYER_NAME TEAM_ABBREVIATI~   AGE W_PCT OFF_RATING DEF_RATING NET_RATING
##   <chr>       <chr>            <int> <dbl>      <dbl>      <dbl>      <dbl>
## 1 Jarnell St~ DEN                 23 0           115.       118.       -3.1
## 2 Ricky Rubio MIN                 26 0.373       109.       110.       -1
## # ... with 8 more variables: AST_RATIO <dbl>, REB_PCT <dbl>,
## #   USG_PCT <dbl>, PIE <dbl>, SALARY_MILLIONS <dbl>,
## #   ACTIVE_TWITTER_LAST_YEAR <int>, TWITTER_FOLLOWER_COUNT_MILLIONS <dbl>,
## #   PTS <dbl>
```

As we can see from the histogram, the assist-turnover ratio is very right skewed. The mode is at around 10, even though the median is at 15, and the mean is 17.12526, all of which are summary statistics that

emphasize the right skew. This means that while most players in the dataset had a very high assist-turnover ratio (meaning they had many more assists than turnovers), there is a wider variation among players with a high ratio and the players with lower ratios are concentrated around a few numbers. The dataset minimum ratio of 4 means that there were no players with more turnovers than assists. Notably, this is the first variable we've examined so far with a significantly non-normal distribution. The two players with very high assist-turnover ratios, 51.5 and 41.3, are Jarnell Stokes and Ricky Rubio, respectively.

Next, we'll examine the variation of `REB_PCT`, the percent of rebounds a player makes:

```
ggplot(data = nba_social_power_mod, aes(x= REB_PCT)) +
  geom_histogram(binwidth = .01) +
  labs(x = "Rebound Percentage", y = "Count",
       title = "Distribution of Rebound Percentage")
```




```
nba_social_power_mod %>%
  summarise(mean = mean(REB_PCT),
            min= min(REB_PCT),
            Q1 = quantile(REB_PCT, .25),
            median = median(REB_PCT),
            Q3 = quantile(REB_PCT, .75),
            max = max(REB_PCT))
```

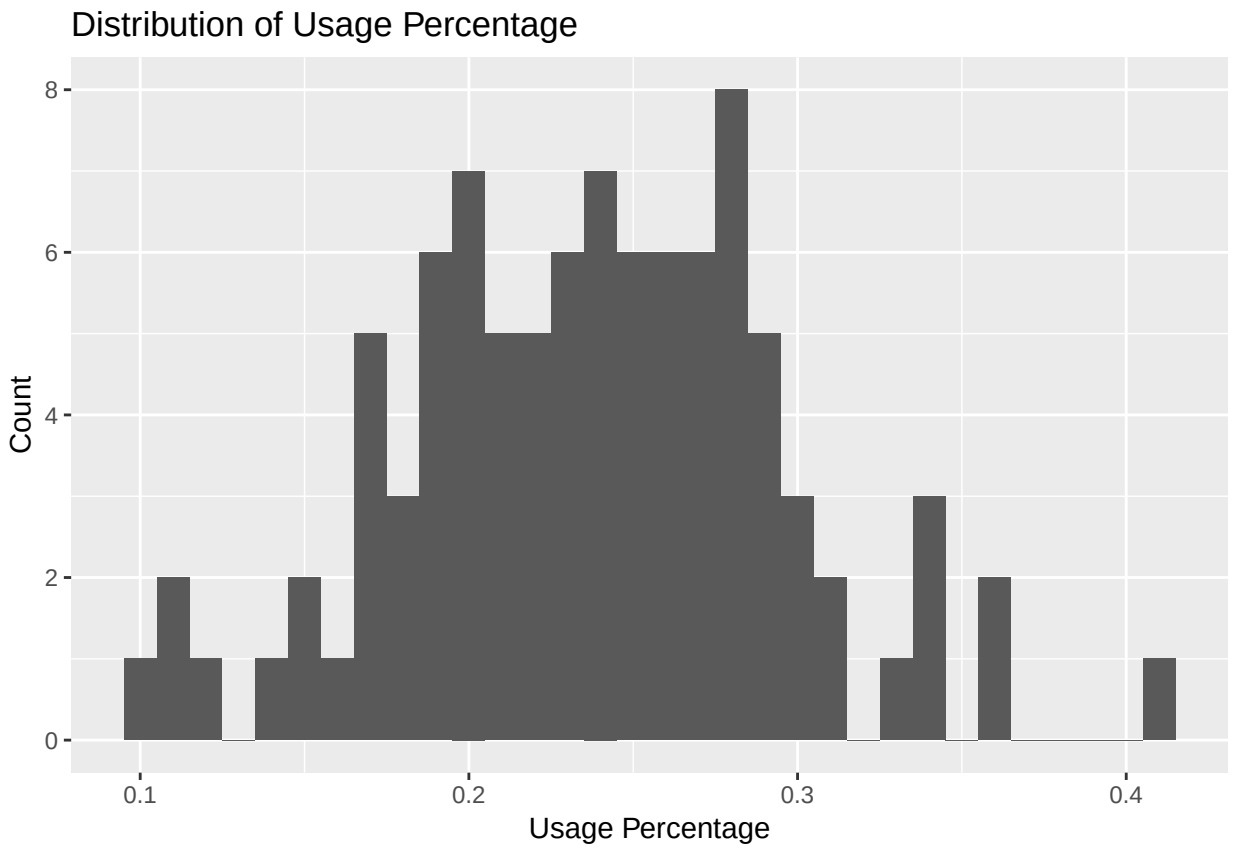
```
## # A tibble: 1 x 6
##    mean   min     Q1 median    Q3   max
##   <dbl> <dbl>  <dbl>  <dbl> <dbl> <dbl>
## 1 0.133 0.045 0.0825  0.127 0.180 0.252
```

The distribution of rebound percentage has a minimum of 0.045 and a maximum of 0.252. The distribution is not very skewed one way or another, as supported by the similar mean of .133 and median of .127. However,

the distribution is not normal in that it does not resemble a bell curve; with exceptions, the data is somewhat evenly distributed from the minimum to near the maximum (although there is some trail-off towards the right side of the distribution). This non-normal spread is likely partially an indication of the fact that the dataset contains both offensive and defensive players, because whether a player is on offense or defense has a significant effect on their rebound percentage.

Next, we'll look at `USG_PCT`, usage percentage, which is an estimate of how often a player makes team plays:

```
ggplot(data = nba_social_power_mod, aes(x= USG_PCT)) +
  geom_histogram(binwidth = .01) +
  labs(x = "Usage Percentage", y = "Count",
       title = "Distribution of Usage Percentage")
```




```
nba_social_power_mod %>%
  summarise(mean = mean(USG_PCT),
            min= min(USG_PCT),
            Q1 = quantile(USG_PCT, .25),
            median = median(USG_PCT),
            Q3 = quantile(USG_PCT, .75),
            max = max(USG_PCT))
```

```
## # A tibble: 1 x 6
##    mean   min    Q1 median    Q3   max
##   <dbl> <dbl> <dbl>  <dbl> <dbl> <dbl>
## 1 0.238 0.101   0.2  0.242 0.276 0.408
```

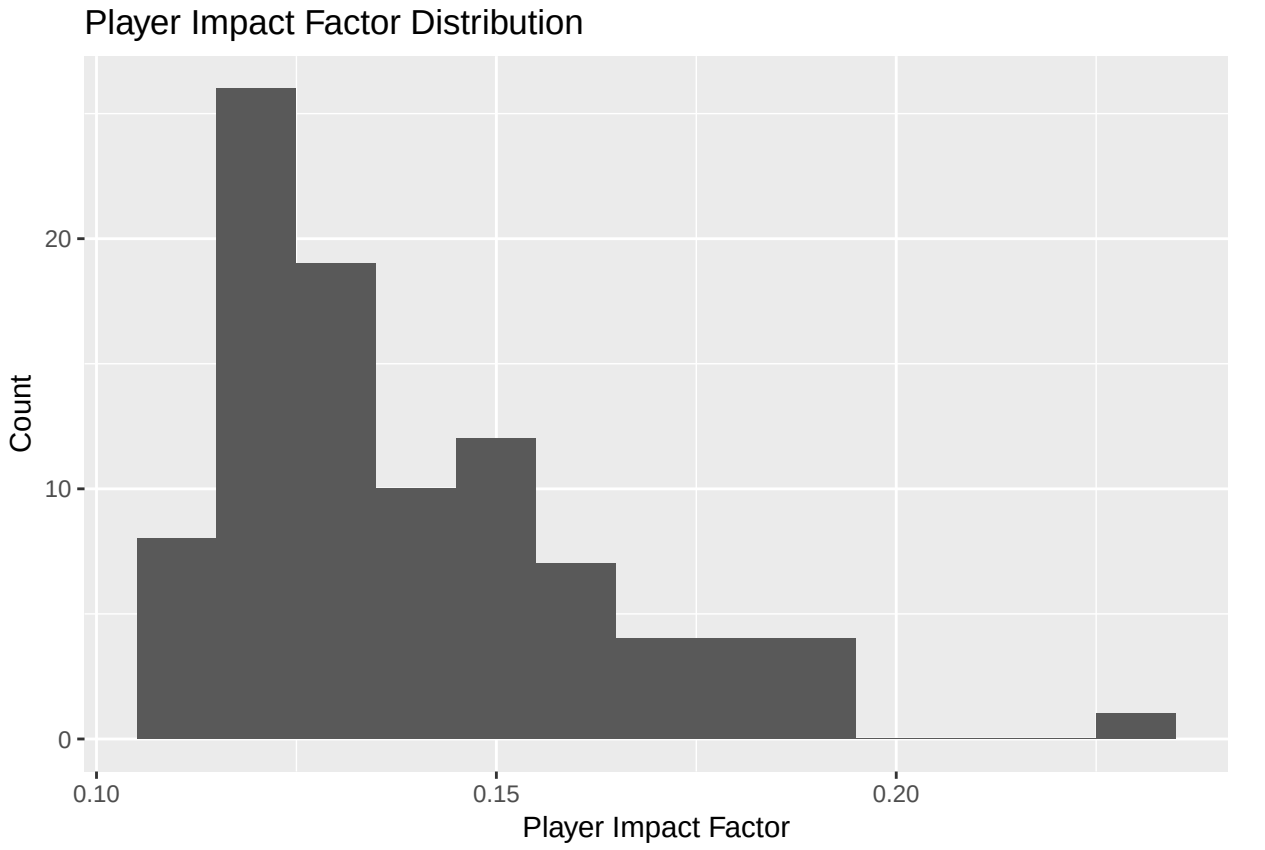
```
nba_social_power_mod %>%
  arrange(desc(USG_PCT)) %>%
  head(1)
```

```
## # A tibble: 1 x 15
##   PLAYER_NAME TEAM_ABBREVIATI~   AGE W_PCT OFF_RATING DEF_RATING NET_RATING
##   <chr>       <chr>            <int> <dbl>      <dbl>      <dbl>      <dbl>
## 1 Russell We~ OKC                 28 0.568       108.       105.        3.3
## # ... with 8 more variables: AST_RATIO <dbl>, REB_PCT <dbl>,
## #   USG_PCT <dbl>, PIE <dbl>, SALARY_MILLIONS <dbl>,
## #   ACTIVE_TWITTER_LAST_YEAR <int>, TWITTER_FOLLOWER_COUNT_MILLIONS <dbl>,
## #   PTS <dbl>
```

The distribution of usage percentage, with a minimum of .101 and a maximum of .408, is fairly normally distributed. The mean, .238, and median, .242, are similar. The fairly wide spread may indicate that the dataset contains a decent sampling of players- some 'star player' types and others that are not the centerpieces of their teams.The maximum of .408, while perhaps not quite an outlier, is separated from most of the other points; this usage percentage belongs to Russell Westbrook.

Next, we'll examine `PIE`, player impact factor, a statistic roughly measuring a player's impact on the games that they play that's used by nba.com:

```
ggplot(data = nba_social_power_mod, aes(x= PIE)) +
  geom_histogram(binwidth = .01) +
  labs(x = "Player Impact Factor", y = "Count",
       title = "Player Impact Factor Distribution")
```




```
nba_social_power_mod %>%
  summarise(mean = mean(PIE),
            min= min(PIE),
            Q1 = quantile(PIE, .25),
            median = median(PIE),
            Q3 = quantile(PIE, .75),
```

```
            max = max(PIE))
```

```
## # A tibble: 1 x 6
##    mean   min    Q1 median    Q3   max
##   <dbl> <dbl> <dbl>  <dbl> <dbl> <dbl>
## 1 0.139 0.112 0.122  0.131 0.152  0.23
```

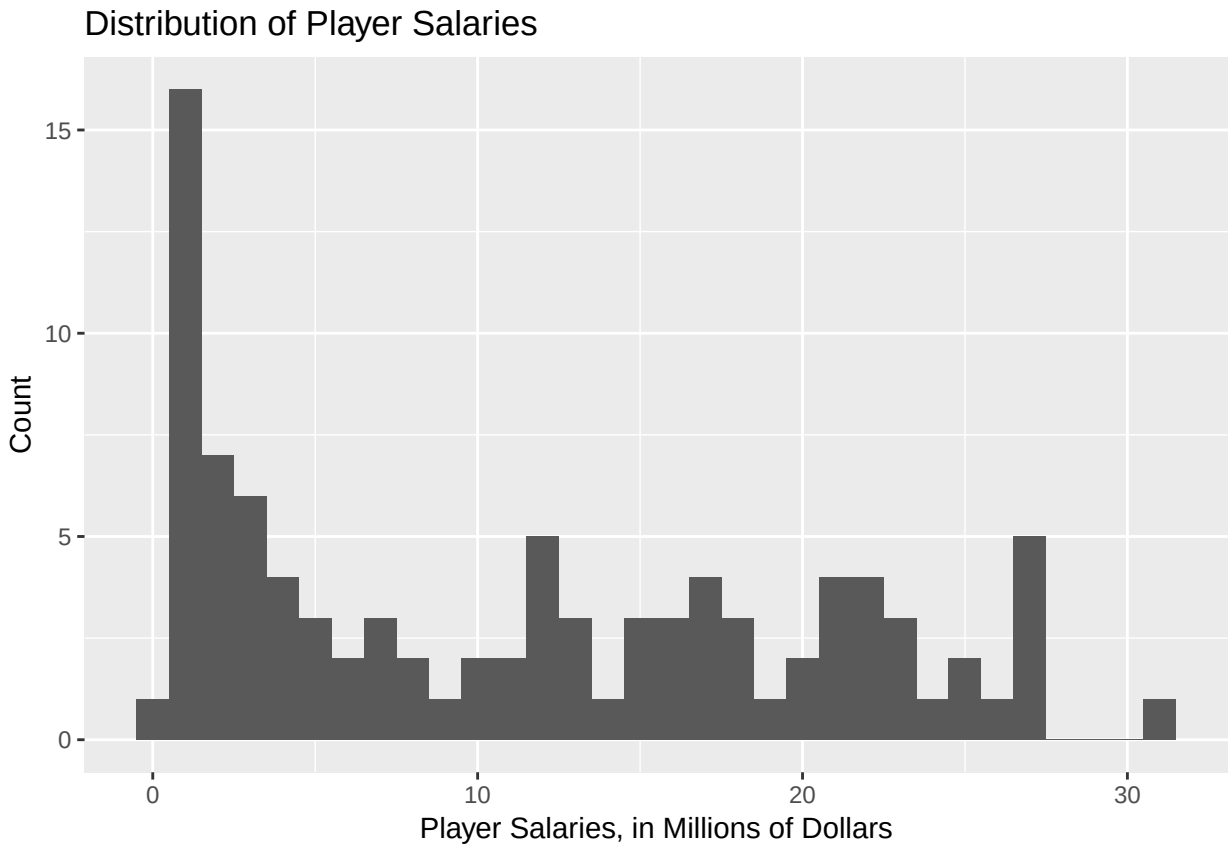
```
nba_social_power_mod %>%
  arrange(desc(PIE)) %>%
  select(PLAYER_NAME, PIE) %>%
  head(10)
```

```
## # A tibble: 10 x 2
##    PLAYER_NAME             PIE
##    <chr>                 <dbl>
##  1 Russell Westbrook     0.23
##  2 Demetrius Jackson     0.194
##  3 Anthony Davis         0.192
##  4 James Harden          0.19
##  5 Kevin Durant          0.186
##  6 LeBron James          0.183
##  7 Chris Paul            0.182
##  8 DeMarcus Cousins      0.178
##  9 Giannis Antetokounmpo 0.176
## 10 Kawhi Leonard         0.174
```

As we can see from the histogram, the player impact factor, with a minimum of .112 and a maximum of .23, is quite right-skewed. The median player impact factor is .131, and the mean is .139, evidence of the right skew. The mode is around the median. The maximum, .23, is a significant outlier, and is that of Russell Westbrook, the same player who had by far the highest usage percentage; clearly, his data will need to be examined more closely later to see if it ultimately affects our model.

Next, we'll look into players' salaries, in millions of dollars:

```
ggplot(data = nba_social_power_mod,
       aes(x = SALARY_MILLIONS)) +
  geom_histogram(binwidth = 1) +
  labs(x = "Player Salaries, in Millions of Dollars",
       y = "Count",
       title = "Distribution of Player Salaries")
```




```
nba_social_power_mod %>%
  summarise(mean = mean(SALARY_MILLIONS),
            min= min(SALARY_MILLIONS),
            Q1 = quantile(SALARY_MILLIONS, .25),
            median = median(SALARY_MILLIONS),
            Q3 = quantile(SALARY_MILLIONS, .75),
            max = max(SALARY_MILLIONS))
```

```
## # A tibble: 1 x 6
##    mean   min    Q1 median    Q3   max
##   <dbl> <dbl> <dbl>  <dbl> <dbl> <dbl>
## 1  11.3  0.31  2.47   11.3  18.5  31.0
```

```
nba_social_power_mod %>%
  arrange(desc(SALARY_MILLIONS)) %>%
  select(PLAYER_NAME, SALARY_MILLIONS) %>%
  head(10)
```

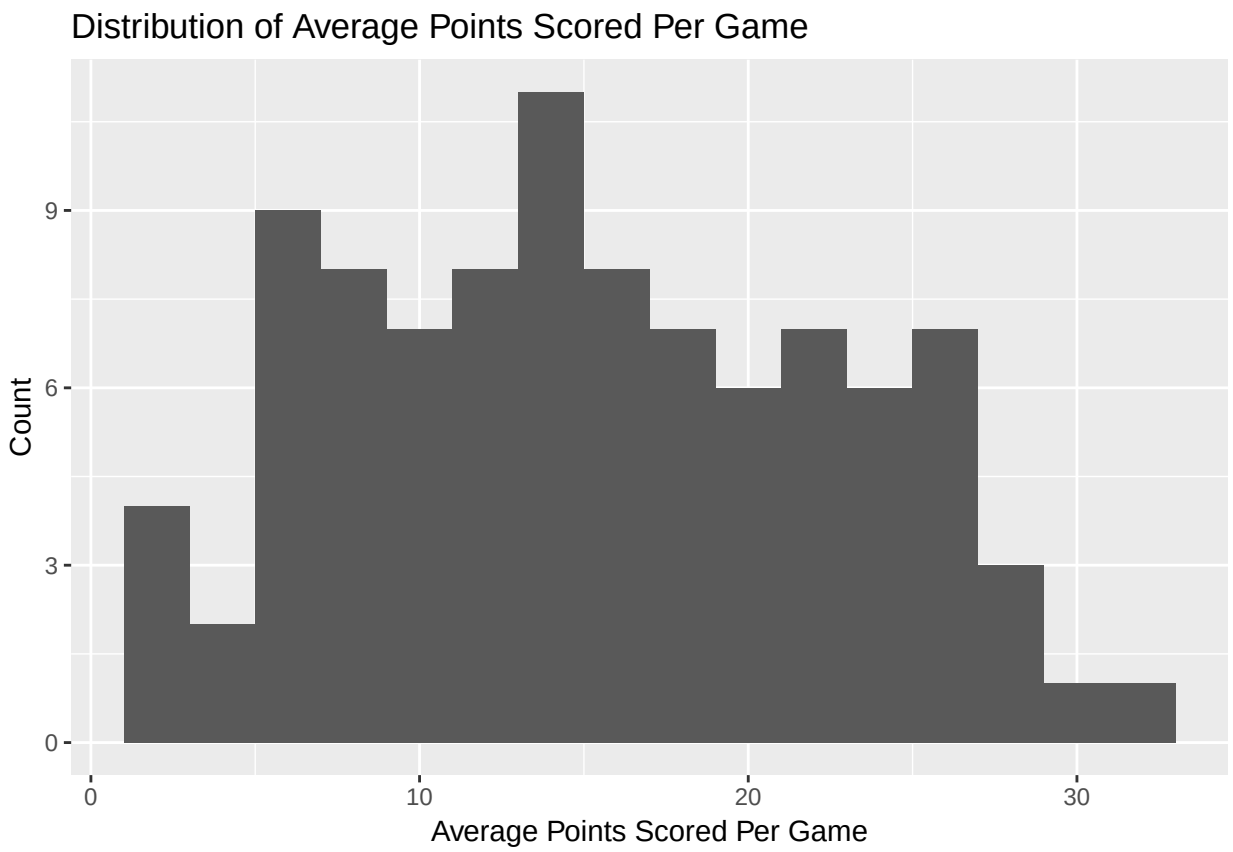
```
## # A tibble: 10 x 2
##    PLAYER_NAME       SALARY_MILLIONS
##    <chr>                       <dbl>
##  1 LeBron James                 31.0
##  2 Russell Westbrook            26.5
##  3 Kevin Durant                 26.5
##  4 Mike Conley                  26.5
##  5 DeMar DeRozan                26.5
##  6 Al Horford                   26.5
##  7 James Harden                 26.5
```

```
##  8 Dirk Nowitzki                25
##  9 Carmelo Anthony              24.6
## 10 Damian Lillard               24.3
```

As we can see from the histogram, the distribution of salaries is somewhat right-skewed, with most of the players making less than 20 million dollars a year. The mean salary is 11.3 million dollars a year. The player who earns the most, at 30.96 million dollars per year, is LeBron James (LeBron could be an influential point, so we will revisit this after the model selection phase). On the other hand, Russell Westbrook, Kevin Durant, Mike Conley, DeMar DeRozan, and Al Horford each earn 26.54 million dollars per year.

Next, we'll look at average points scored per game:

```
ggplot(data = nba_social_power_mod, aes(x= PTS)) +
  geom_histogram(binwidth = 2) +
  labs(x = "Average Points Scored Per Game",
       y = "Count",
       title = "Distribution of Average Points Scored Per Game")
```




```
nba_social_power_mod %>%
  summarise(mean = mean(PTS),
            min= min(PTS),
            Q1 = quantile(PTS, .25),
            median = median(PTS),
            Q3 = quantile(PTS, .75),
            max = max(PTS))
```

```
## # A tibble: 1 x 6
##    mean   min    Q1 median    Q3   max
##   <dbl> <dbl> <dbl>  <dbl> <dbl> <dbl>
```

```
## 1  15.3   1.5   9.5   14.6  21.4  31.6
```

As we can see from the histogram, the distribution of average points scored is slightly normal, with some obvious departures from normality. The median number of points scored per game is 14.6, and the mean is 15.28232. The maximum is 31.6, but this does not seem to be an obvious outlier.

Next, we'll examine the variable `ACTIVE_TWITTER_LAST_YEAR`, which tells us whether or not each player posted on Twitter the year before the data was collected:

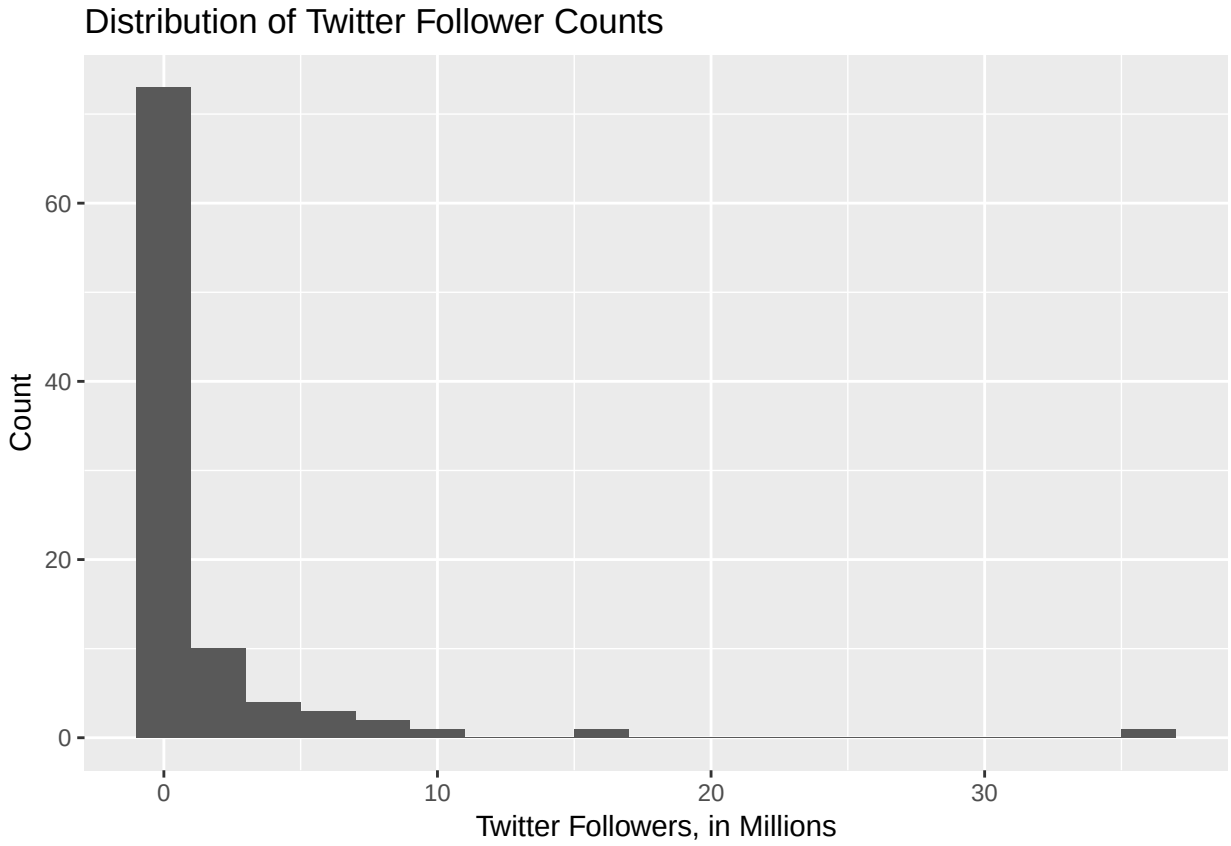
```
nba_social_power_mod <- nba_social_power_mod %>%
  mutate(ACTIVE_TWITTER_LAST_YEAR = as.factor(ACTIVE_TWITTER_LAST_YEAR))

nba_social_power_mod %>%
  count(ACTIVE_TWITTER_LAST_YEAR)
```

```
## # A tibble: 2 x 2
##   ACTIVE_TWITTER_LAST_YEAR     n
##   <fct>                    <int>
## 1 0                            2
## 2 1                           93
```

Out of the 95 players in our modified dataset, 2 were not active on Twitter the year before the data was collected and and 93 were.

Finally, we'll look into the response variable, `TWITTER_FOLLOWER_COUNT_MILLIONS`:

```
ggplot(data = nba_social_power_mod,
       aes(x= TWITTER_FOLLOWER_COUNT_MILLIONS)) +
  geom_histogram(binwidth = 2) +
  labs(x = "Twitter Followers, in Millions",
       y = "Count",
       title = "Distribution of Twitter Follower Counts")
```




```
nba_social_power_mod %>%
  summarise(mean = mean(TWITTER_FOLLOWER_COUNT_MILLIONS),
            min= min(TWITTER_FOLLOWER_COUNT_MILLIONS),
            Q1 = quantile(TWITTER_FOLLOWER_COUNT_MILLIONS, .25),
            median =   median(TWITTER_FOLLOWER_COUNT_MILLIONS),
            Q3 = quantile(TWITTER_FOLLOWER_COUNT_MILLIONS, .75),
            max = max(TWITTER_FOLLOWER_COUNT_MILLIONS))
```

```
## # A tibble: 1 x 6
##    mean   min     Q1 median    Q3   max
##   <dbl> <dbl>  <dbl>  <dbl> <dbl> <dbl>
## 1  1.60 0.002 0.0595  0.246 0.912    37
```

```
nba_social_power_mod %>%
  arrange(desc(TWITTER_FOLLOWER_COUNT_MILLIONS)) %>%
  head(2)
```

```
## # A tibble: 2 x 15
##   PLAYER_NAME TEAM_ABBREVIATI~   AGE W_PCT OFF_RATING DEF_RATING NET_RATING
##   <chr>       <chr>            <int> <dbl>      <dbl>      <dbl>      <dbl>
## 1 LeBron Jam~ CLE                 32 0.689       115.       107.        7.7
## 2 Kevin Dura~ GSW                 28 0.823       117.       101.       16
## # ... with 8 more variables: AST_RATIO <dbl>, REB_PCT <dbl>,
## #   USG_PCT <dbl>, PIE <dbl>, SALARY_MILLIONS <dbl>,
## #   ACTIVE_TWITTER_LAST_YEAR <fct>, TWITTER_FOLLOWER_COUNT_MILLIONS <dbl>,
## #   PTS <dbl>
```

From the histogram, we can see that the distribution of Twitter follower counts is extremely right-skewed. The number of Twitter followers ranges from .002 million to 37 million, with a mean of 1.6 million and a

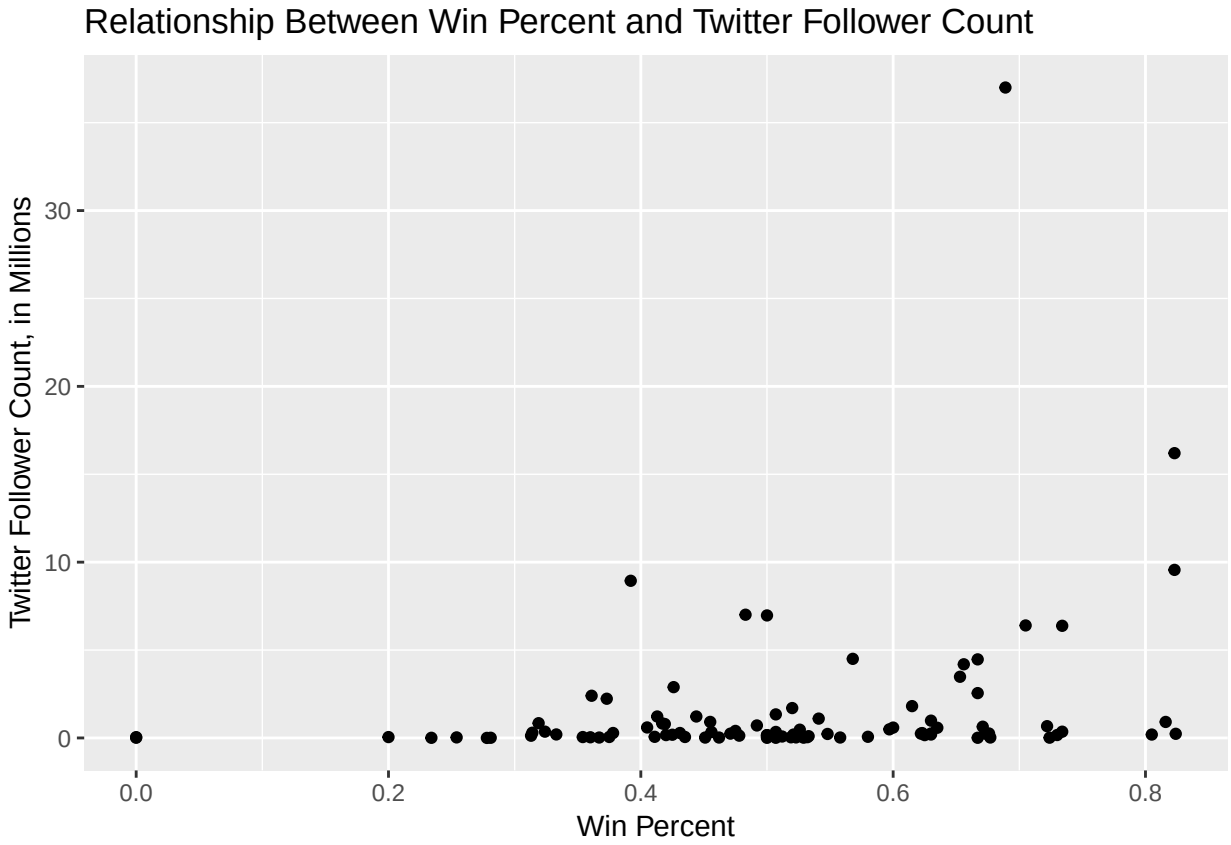
median of.246 million. There are two obvious outliers: Kevin Durant, with 16.2 million followers, and LeBron James, with 37 million followers.

Bivariate

Next, we will do bivariate EDA, looking into the relationships of some of the predictor variables with the response variables. We won't do bivariate EDA on player name, Twitter handle, age, team abbreviation, or whether the players were active on Twitter last year, instead focusing on terms we believe may play a more nuanced / important role in predicting Twitter followers.

First, we'll look for a relationship between win percent and Twitter followers in millions:

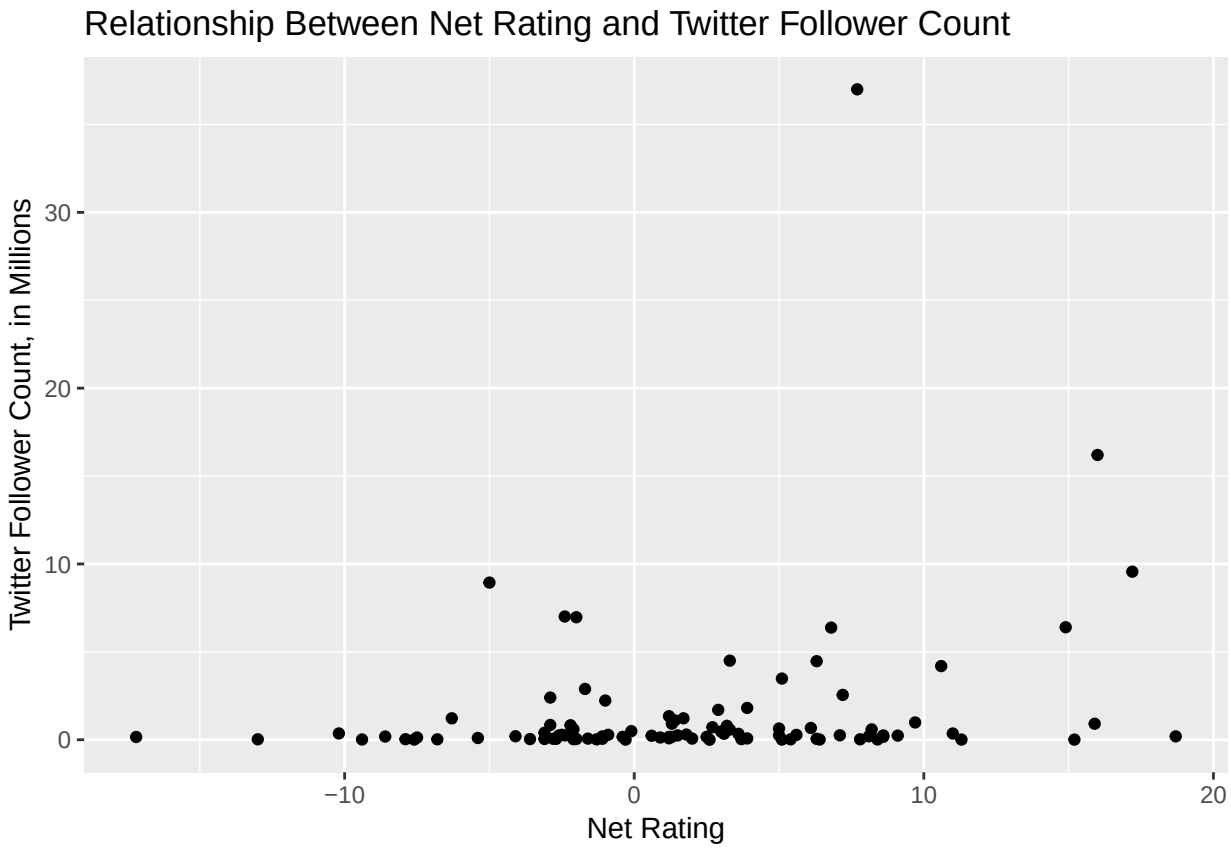
```
ggplot(nba_social_power_mod, aes(x = W_PCT, y = TWITTER_FOLLOWER_COUNT_MILLIONS)) +
  geom_point() +
  labs(title = "Relationship Between Win Percent and Twitter Follower Count",
       x = "Win Percent", y = "Twitter Follower Count, in Millions")
```



From the above plot, we can see that win percent and Twitter follower count may have a very weak positive orrelations. The players with significantly higher-than-average Twitter follower counts tend to have higher win percentages; however, this relationship is very weak.

Next, we'll look at the relationship between net rating, which combines offensive and defensive rating, and Twitter follower count:

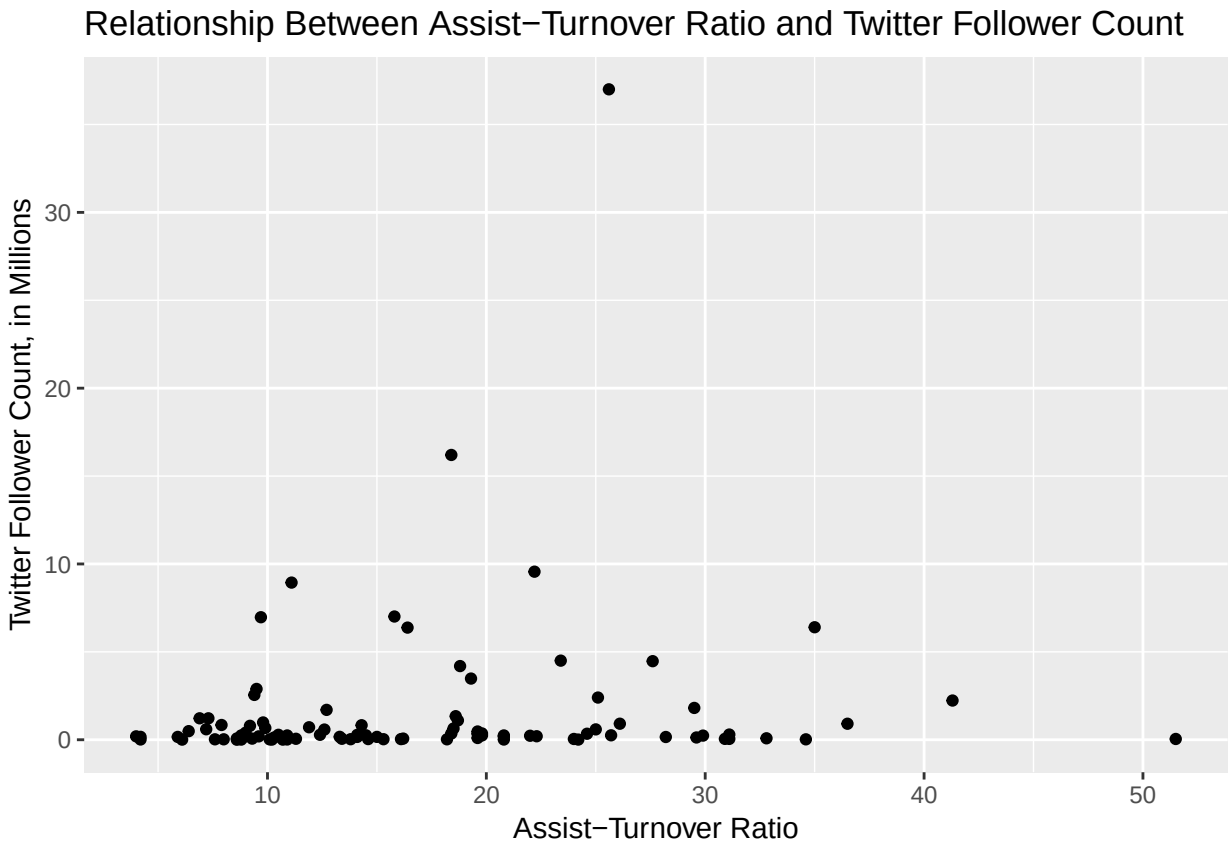
```
ggplot(nba_social_power_mod, aes(x = NET_RATING, y = TWITTER_FOLLOWER_COUNT_MILLIONS)) +
  geom_point() +
  labs(title = "Relationship Between Net Rating and Twitter Follower Count",
       x = "Net Rating", y = "Twitter Follower Count, in Millions")
```



Similarly to win percent, Twitter follower count and net rating seem to have a very weak positive relationship. The players with the highest net ratings more often have higher-than-average Twitter follower counts, and more players with positive net ratings have high Twitter follower counts than those with negative net ratings.

Next, we'll look at the relationship between assist-to-turnovers ratio and Twitter follower count:

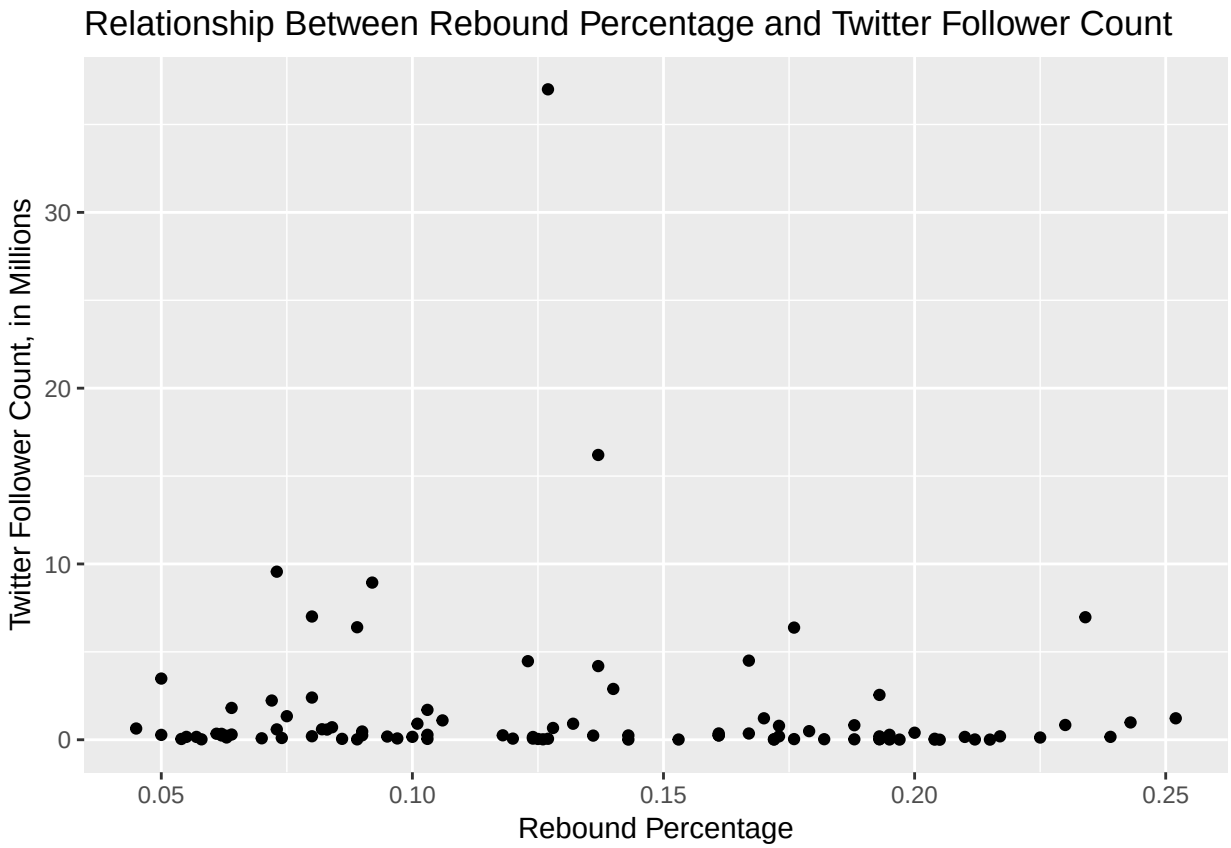
```
ggplot(nba_social_power_mod,
       aes(x = AST_RATIO,
           y = TWITTER_FOLLOWER_COUNT_MILLIONS)) +
  geom_point() +
  labs(title = "Relationship Between Assist-Turnover Ratio and Twitter Follower Count",
       x = "Assist-Turnover Ratio",
       y = "Twitter Follower Count, in Millions")
```



There is no evident relationship between assist-turnover ratio and Twitter follower count.

Next, we'll examine whether there is a relationship between rebound percentage and Twitter follower count:

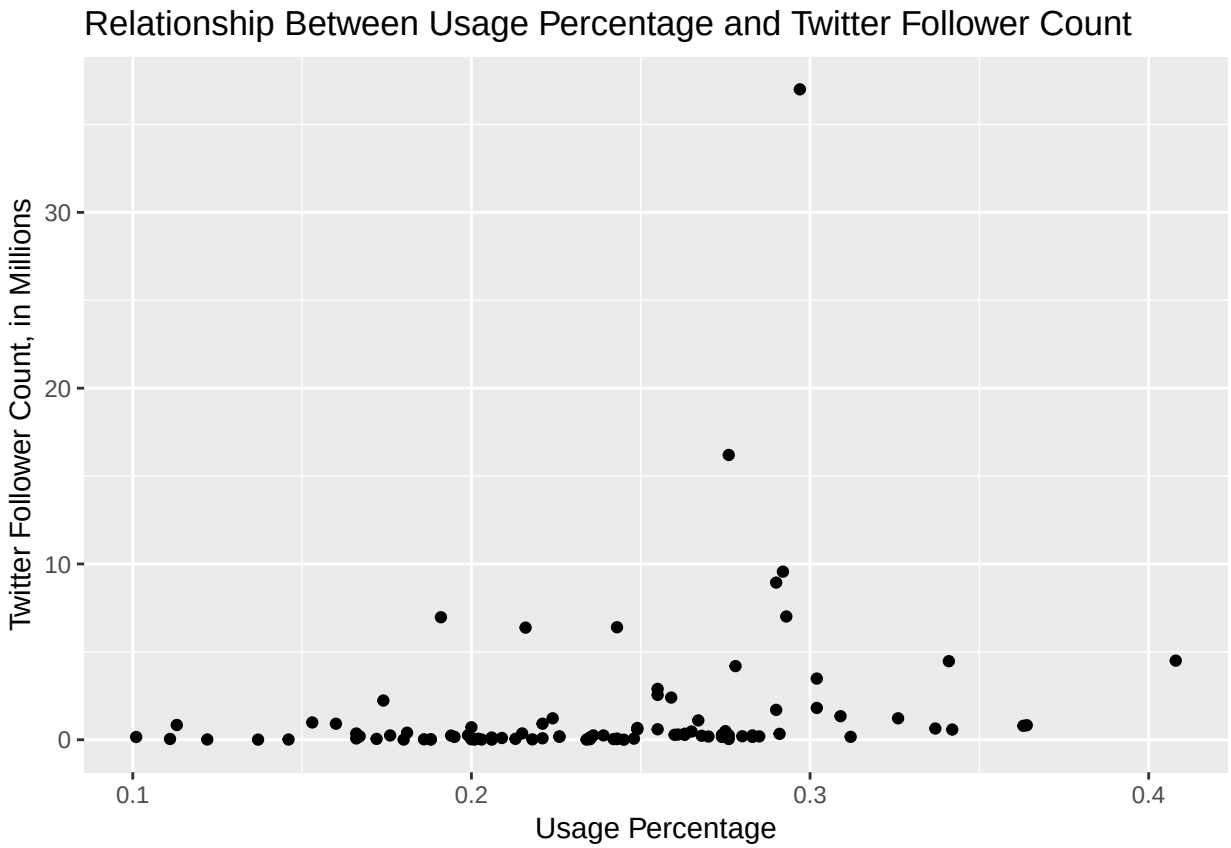
```
ggplot(nba_social_power_mod,
       aes(x = REB_PCT,
           y = TWITTER_FOLLOWER_COUNT_MILLIONS)) +
  geom_point() +
  labs(title = "Relationship Between Rebound Percentage and Twitter Follower Count",
       x = "Rebound Percentage",
       y = "Twitter Follower Count, in Millions")
```



There is no evident relationship between rebound percentage and Twitter follower count.

Next, we'll examine whether there is a relationship between usage percentage and Twitter follower count:

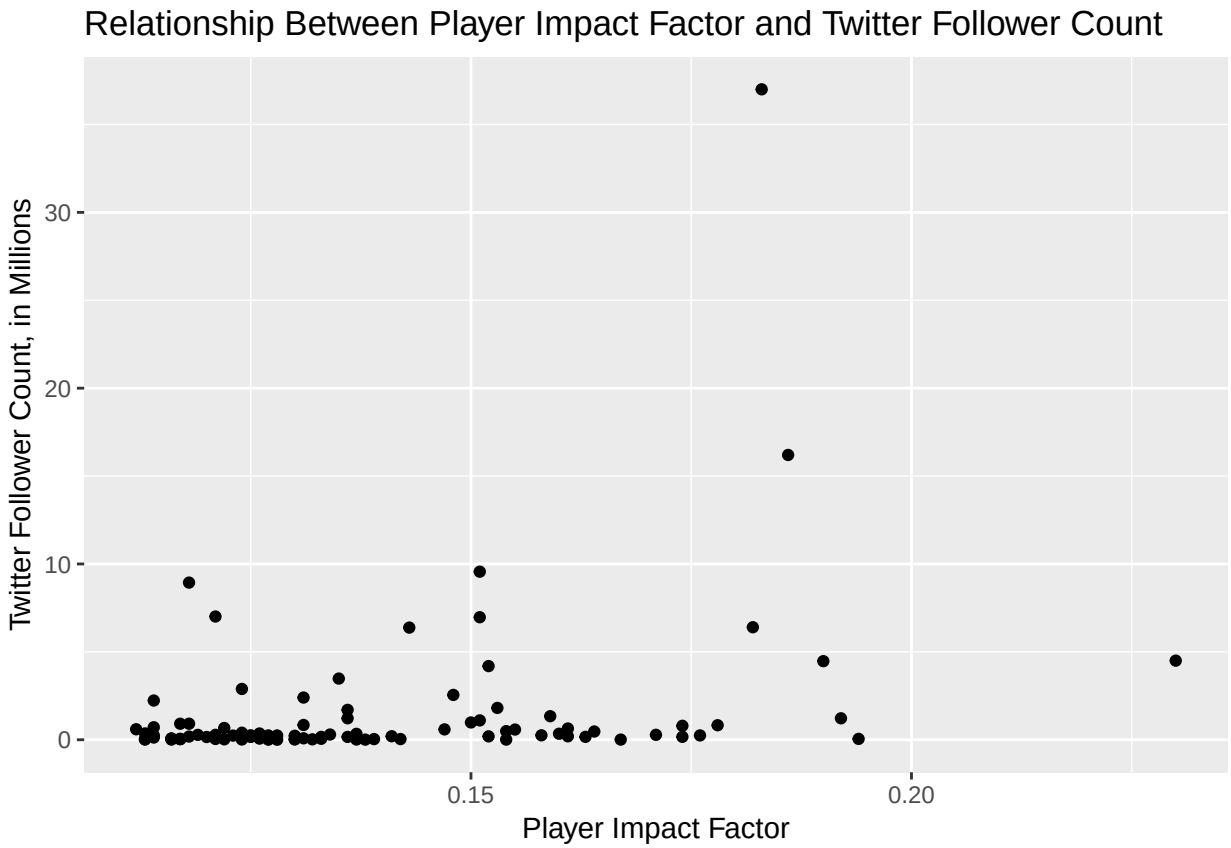
```
ggplot(nba_social_power_mod,
       aes(x = USG_PCT,
           y = TWITTER_FOLLOWER_COUNT_MILLIONS)) +
  geom_point() +
  labs(title = "Relationship Between Usage Percentage and Twitter Follower Count",
       x = "Usage Percentage",
       y = "Twitter Follower Count, in Millions")
```



There appears to be a very weak positive correlation between usage percentage and Twitter follower count; players with a high usage percentage tend to have more Twitter followers, on average, than those with a lower usage percentage.

Next, we'll examine whether there is a relationship between player impact factor and Twitter follower count:

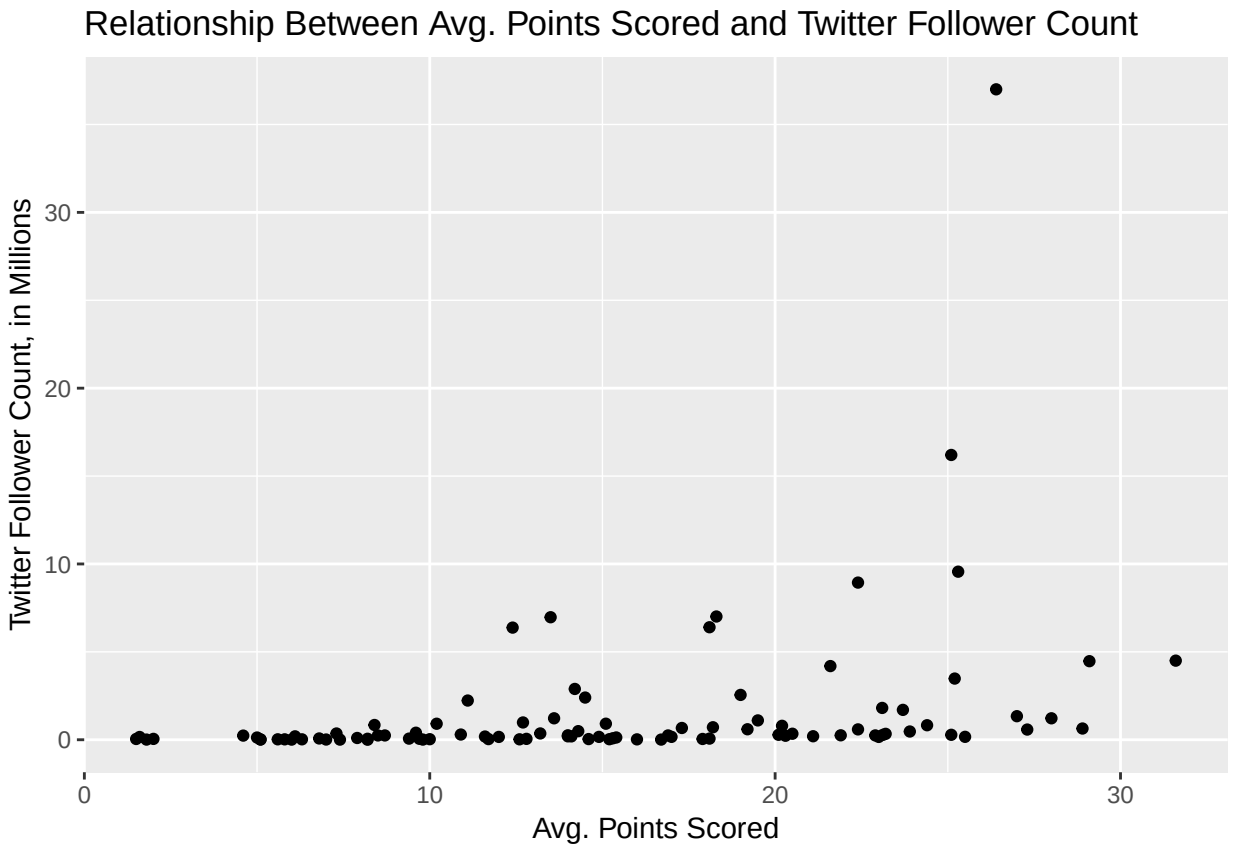
```
ggplot(nba_social_power_mod,
       aes(x = PIE, y = TWITTER_FOLLOWER_COUNT_MILLIONS)) +
  geom_point() +
  labs(title = "Relationship Between Player Impact Factor and Twitter Follower Count",
       x = "Player Impact Factor",
       y = "Twitter Follower Count, in Millions")
```



There appears to be a somewhat positive correlation between player impact factor and Twitter follower count; players with a high player impact factor tend to have more Twitter followers, on average, than those with a lower player impact factor.

Now, we'll look for a relationship between average points scored per game and Twitter follower count:

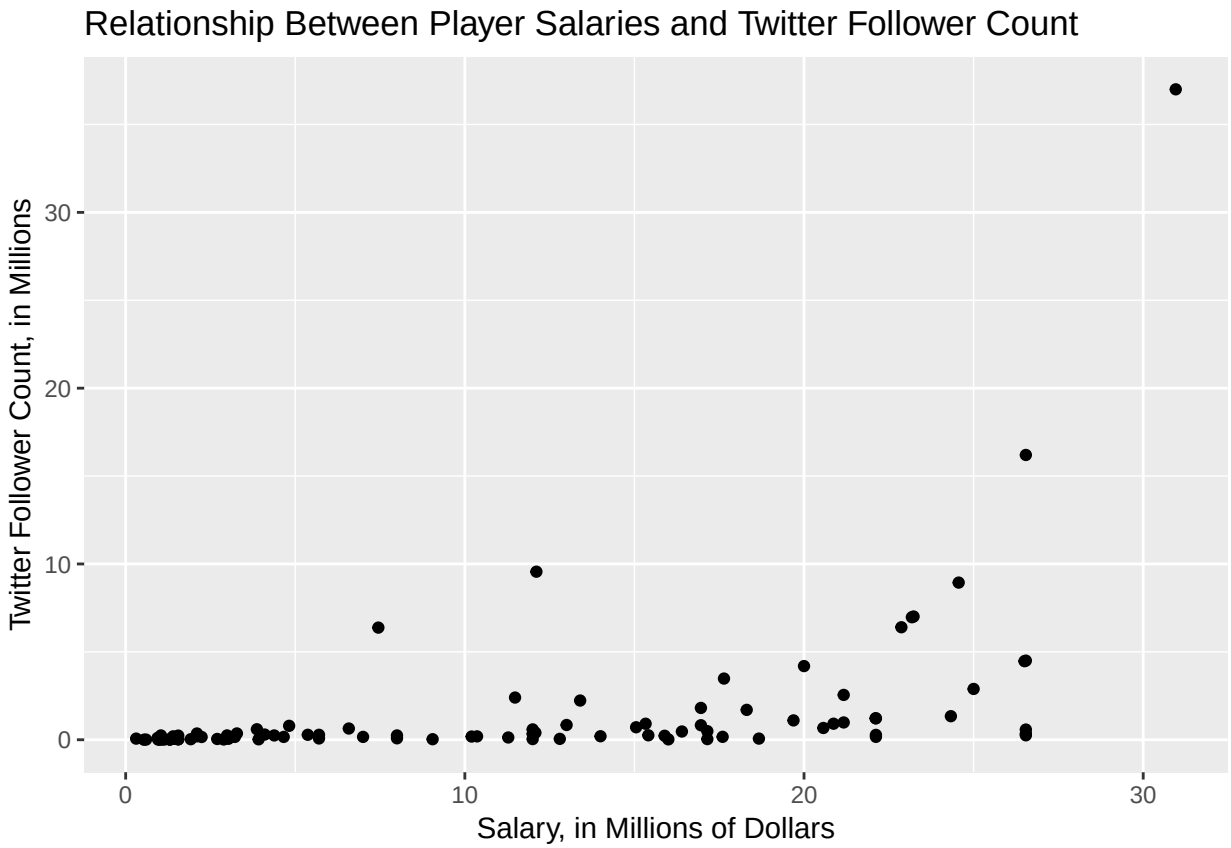
```
ggplot(nba_social_power_mod,
       aes(x = PTS, y = TWITTER_FOLLOWER_COUNT_MILLIONS)) +
  geom_point() +
  labs(title = "Relationship Between Avg. Points Scored and Twitter Follower Count",
       x = "Avg. Points Scored",
       y = "Twitter Follower Count, in Millions")
```



There appears to be a weak positive correlation between average points scored and Twitter follower count; players with higher avg. points scored tend to have more Twitter followers than those with lower avg. points scored.

Finally, we'll look for a relationship between player salaries and points scored:

```
ggplot(nba_social_power_mod, aes(x = SALARY_MILLIONS, y = TWITTER_FOLLOWER_COUNT_MILLIONS)) +
  geom_point() +
  labs(title = "Relationship Between Player Salaries and Twitter Follower Count",
       x = "Salary, in Millions of Dollars",
       y = "Twitter Follower Count, in Millions")
```



There appears to be a positive correlation between salary and Twitter follower count; the players with higher salaries tend to have higher Twitter follower counts.

In sum, there appear to be weak to moderate positive correlations between Twitter follower count and win percentage, net rating, usage percentage, player impact factor, avg. points per game, and player salaries; there is no obvious relationship between Twitter follower count and assist-to-turnover ratio or rebound percentage.

patchwork: multinomial logistic lecture code

Multivariate Data Analysis

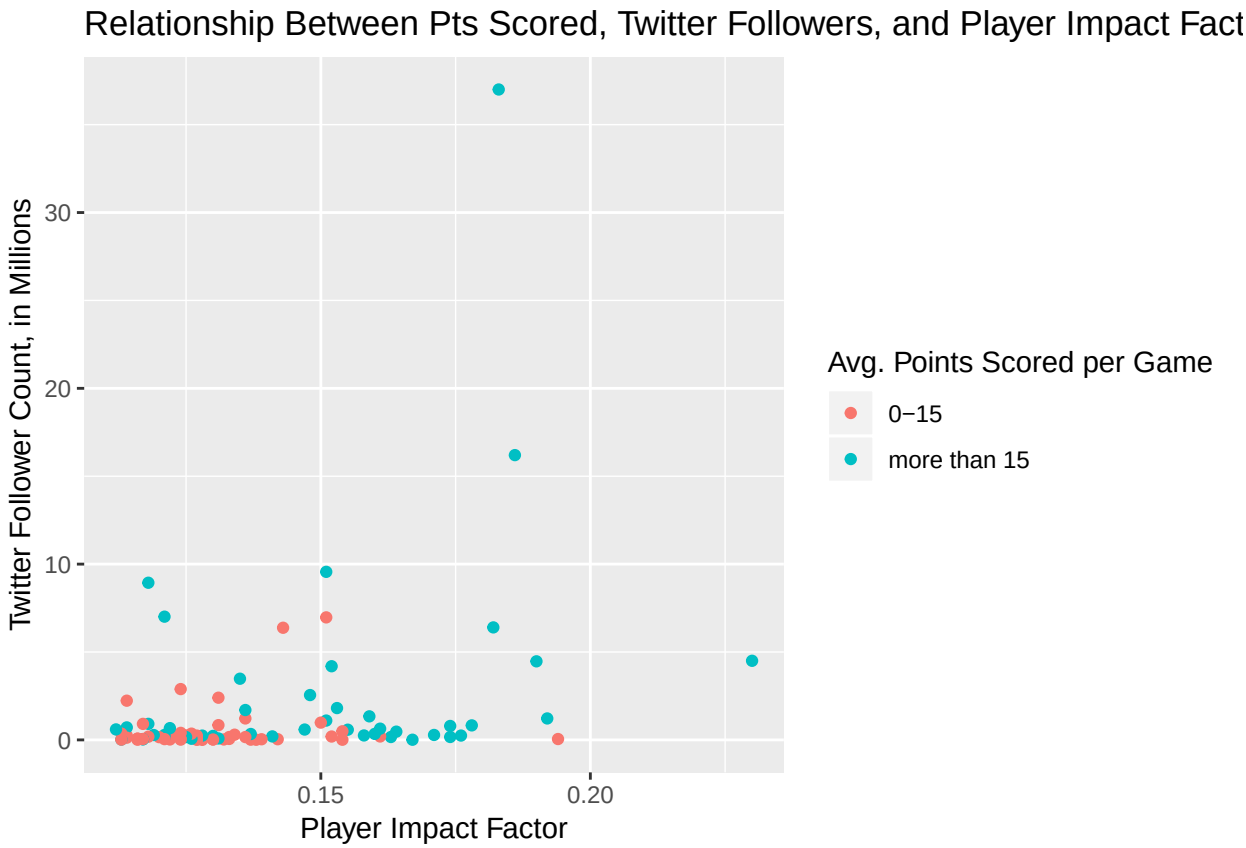
Now, we'll do some multivariate analysis. In this section, we are looking for predictor variables that may affect the way other predictor variables relate to the response variable.

First, we'll look to see if points scored affects the way player impact factor (PIE) relates to Twitter follower count:

```
nba_social_power_mod1 <- nba_social_power_mod %>%
  mutate(PTS_CAT = case_when(
    PTS <= 15 ~ "0-15" ,
    PTS > 15 ~ "more than 15"
  ))
```

```
ggplot(data = nba_social_power_mod1,
       aes(x = PIE,
           y = TWITTER_FOLLOWER_COUNT_MILLIONS,
           color = PTS_CAT)) +
         geom_point() +
```

```
  labs(x = "Player Impact Factor",
       y= "Twitter Follower Count, in Millions",
       color = "Avg. Points Scored per Game",
       title = "Relationship Between Pts Scored, Twitter Followers, and Player Impact Factor")
```

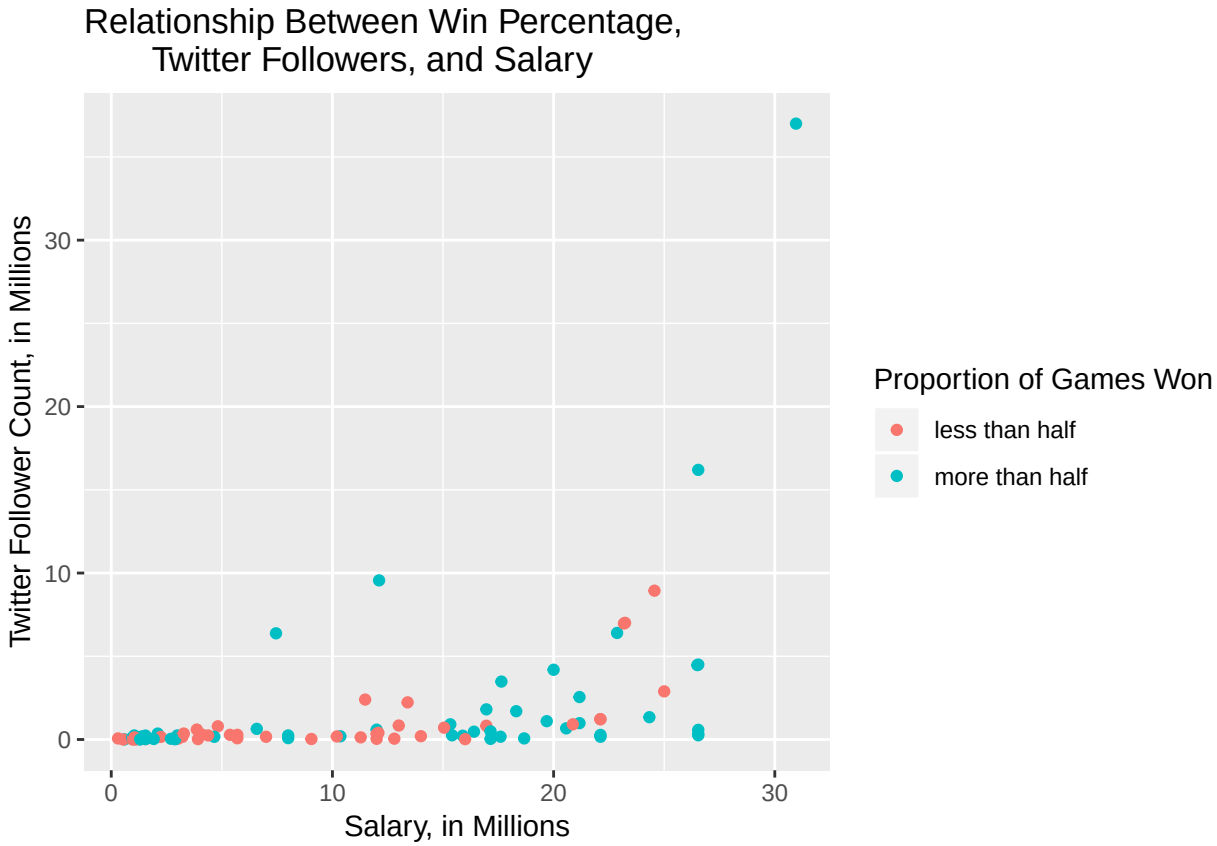


As we can see from the above color-coded boxplot, many of the players with the most points scored have higher player impact factors, and player impact factor values have a weak positive correlation with the Twitter follower count. This could be an opportunity for an interaction term.

Next, we'll try to determine whether win percentage affects the way salary relates to the Twitter follower count:

```
nba_social_power_mod1 <- nba_social_power_mod %>%
  mutate(W_PCT_CAT = case_when(
    W_PCT <= .5 ~ "less than half" ,
    W_PCT > .5 ~ "more than half"
  ))
```

```
ggplot(data = nba_social_power_mod1,
       aes(x = SALARY_MILLIONS,
           y = TWITTER_FOLLOWER_COUNT_MILLIONS,
           color = W_PCT_CAT)) +
         geom_point() +
  labs(x = "Salary, in Millions",
       y = "Twitter Follower Count, in Millions",
       color = "Proportion of Games Won",
       title = "Relationship Between Win Percentage,
       Twitter Followers, and Salary")
```



As we can see from the scatterplot, players with higher win percentages tend to be paid more, and high salary has a weak positive correlation with the Twitter follower count. This could also be an opportunity for an interaction term.

Adjustments (Mean-Centering)

Before we proceed with the analysis, we will mean-center the quantitative variables:

```
nba_social_power_mod <- nba_social_power_mod %>%
  mutate(ageCent = AGE - mean(AGE),
         ast_ratioCent = AST_RATIO - mean(AST_RATIO),
         off_ratingCent = OFF_RATING - mean(OFF_RATING),
         def_ratingCent = DEF_RATING - mean(DEF_RATING),
         net_ratingCent = NET_RATING - mean(NET_RATING),
         PIECent = PIE - mean(PIE),
         reb_pctCent = REB_PCT - mean(REB_PCT),
         usg_pctCent = USG_PCT - mean(USG_PCT),
         salary_millionsCent = SALARY_MILLIONS - mean(SALARY_MILLIONS),
         w_pctCent = W_PCT - mean(W_PCT),
         ptsCent = PTS - mean(PTS))
```

Modeling Approach

As our response, `TWITTER_FOLLOWER_COUNT_MILLIONS`, is a continuous numerical variable, we will use a multiple linear regression model (obviously, we plan to include two or more predictors, so simple linear regression would not apply!).

In regard to model selection, we will begin by fitting a mutiple linear regression model with main effects, as well as interaction effects. We are considering `salary_millionsCent` * `w_pctCent` and `PIECent` * `ptsCent` as potential interaction terms for our model since multivariate EDA highlighted strong positive relationships between win percentage and player salaries and points scored and PIE (player impact factor).

Next, we will perform two iterations of backward selection on the first model: (i) using AIC as the selection criterion and (ii) using adjusted R-squared as the selection criterion. We decided against trying BIC as the selection criterion because we would prefer more terms in the final model as our objective is to predict the Twitter follower counts of NBA players using measures of athletic success (and predictions are generally more accurate with more relevant predictor variables).

After completing the two iterations of backward selection, we will compare the resulting models and see whether certain terms are removed in both (which would suggest those terms are not statistically significant).

After obtaining a final model, we will determine whether prominent athletes (e.g. LeBron James) are influential points by looking at standardized residuals, leverage, and Cook's Distance (this is important since our objective is to design a model with the best predictive accuracy). We will then learn the impact of including versus excluding prominent athletes in our model. We choose to analyze this at the end of the model selection phase because prominent athletes are also included in the population we want to understand when fitting the multiple linear regression model.

Model 1

We will simply fit a model with eleven main effect terms–`ageCent`, `ast_ratioCent` `off_ratingCent`, `def_ratingCent`, `PIECent`, `reb_pctCent`, `usg_pctCent`, `salary_millionsCent`, `w_pctCent`, `ptsCent`, and `ACTIVE_TWITTER_LAST_YEAR`–and two interaction terms (`salary_millionsCent` * `w_pctCent` and `PIECent` * `ptsCent`). We decided to use `off_ratingCent` and `def_ratingCent` as opposed to `net_ratingCent` because the two individual components potentially provide more important information in predicting the number of Twitter followers.

```
m1 <- lm(TWITTER_FOLLOWER_COUNT_MILLIONS ~ ageCent +
           ast_ratioCent +
           off_ratingCent +
           def_ratingCent +
           PIECent +
           reb_pctCent +
           usg_pctCent +
           salary_millionsCent +
           w_pctCent +
           ptsCent +
           ACTIVE_TWITTER_LAST_YEAR +
           salary_millionsCent * w_pctCent +
           PIECent * ptsCent,
         data = nba_social_power_mod)
tidy(m1, conf.int = TRUE) %>%
  kable(format = "markdown", digits = 3)
```

term	estimate	std.error	statistic	p.value	conf.low	conf.high
(Intercept)	-2.386	2.744	-0.869	0.387	-7.845	3.074
ageCent	0.262	0.123	2.131	0.036	0.017	0.507
ast_ratioCent	0.017	0.064	0.269	0.788	-0.109	0.144
off_ratingCent	0.026	0.106	0.247	0.805	-0.185	0.237
def_ratingCent	0.086	0.118	0.732	0.467	-0.149	0.321
PIECent	13.104	27.617	0.474	0.636	-41.846	68.053
reb_pctCent	7.689	12.120	0.634	0.528	-16.426	31.803

term	estimate	std.error	statistic	p.value	conf.low	conf.high
usg_pctCent	0.052	14.017	0.004	0.997	-27.838	27.942
salary_millionsCent	0.086	0.070	1.240	0.219	-0.052	0.225
w_pctCent	7.095	3.834	1.850	0.068	-0.534	14.725
ptsCent	0.063	0.132	0.481	0.632	-0.199	0.325
ACTIVE_TWITTER_LAST_YEAR1	3.523	2.758	1.278	0.205	-1.964	9.011
salary_millionsCent:w_pctCent	0.999	0.344	2.901	0.005	0.314	1.684
PIECent:ptsCent	1.293	2.293	0.564	0.574	-3.269	5.855

Based on the output, the equation of the linear model is: `TWITTER_FOLLOWER_COUNT_MILLIONS`-hat = -2.386 + 0.262 * `ageCent` + 0.017 * `ast_ratioCent` + 0.026 * `off_ratingCent` + 0.086 * `def_ratingCent` + 13.104 * `PIECent` + 7.689 * `reb_pctCent` + 0.052 * `usg_pctCent` + 0.086 * `salary_millionsCent` + 7.095 * `w_pctCent` + 0.063 * `ptsCent` + 3.523 * `ACTIVE_TWITTER_LAST_YEAR1` + 0.999 * `salary_millionsCent` * `w_pct_Cent` + 1.293 * `PIECent` * `ptsCent`.

Backward Selection (Iteration 1)

We will now perform the first iteration of backward selection using AIC as the selection criterion:

```
m1_aic <- step(m1, direction = "backward")
```

```
## Start:  AIC=265.14
## TWITTER_FOLLOWER_COUNT_MILLIONS ~ ageCent + ast_ratioCent + off_ratingCent +
##     def_ratingCent + PIECent + reb_pctCent + usg_pctCent + salary_millionsCent +
##     w_pctCent + ptsCent + ACTIVE_TWITTER_LAST_YEAR + salary_millionsCent *
##     w_pctCent + PIECent * ptsCent
##
##                                 Df Sum of Sq    RSS    AIC
## - usg_pctCent                    1     0.000 1153.0 263.14
## - off_ratingCent                 1     0.869 1153.8 263.21
## - ast_ratioCent                  1     1.033 1154.0 263.23
## - PIECent:ptsCent                1     4.525 1157.5 263.51
## - reb_pctCent                    1     5.729 1158.7 263.61
## - def_ratingCent                 1     7.619 1160.6 263.77
## - ACTIVE_TWITTER_LAST_YEAR       1    23.232 1176.2 265.04
## <none>                                       1153.0 265.14
## - ageCent                        1    64.667 1217.6 268.32
## - salary_millionsCent:w_pctCent  1   119.780 1272.7 272.53
##
## Step:  AIC=263.14
## TWITTER_FOLLOWER_COUNT_MILLIONS ~ ageCent + ast_ratioCent + off_ratingCent +
##     def_ratingCent + PIECent + reb_pctCent + salary_millionsCent +
##     w_pctCent + ptsCent + ACTIVE_TWITTER_LAST_YEAR + salary_millionsCent:w_pctCent +
##     PIECent:ptsCent
##
##                                 Df Sum of Sq    RSS    AIC
## - off_ratingCent                 1     0.879 1153.8 261.21
## - ast_ratioCent                  1     1.119 1154.1 261.23
## - PIECent:ptsCent                1     4.673 1157.6 261.52
## - reb_pctCent                    1     6.349 1159.3 261.66
## - def_ratingCent                 1     7.812 1160.8 261.78
## - ACTIVE_TWITTER_LAST_YEAR       1    23.398 1176.4 263.05
## <none>                                       1153.0 263.14
```

```
## - ageCent                          1    65.008 1218.0 266.35
## - salary_millionsCent:w_pctCent  1   120.208 1273.2 270.56
##
## Step:  AIC=261.21
## TWITTER_FOLLOWER_COUNT_MILLIONS ~ ageCent + ast_ratioCent + def_ratingCent +
##     PIECent + reb_pctCent + salary_millionsCent + w_pctCent +
##     ptsCent + ACTIVE_TWITTER_LAST_YEAR + salary_millionsCent:w_pctCent +
##     PIECent:ptsCent
##
##                                  Df Sum of Sq    RSS    AIC
## - ast_ratioCent                   1     1.551 1155.4 259.34
## - PIECent:ptsCent                 1     3.809 1157.7 259.53
## - reb_pctCent                     1     6.572 1160.4 259.75
## - def_ratingCent                  1    10.226 1164.1 260.05
## - ACTIVE_TWITTER_LAST_YEAR        1    23.162 1177.0 261.10
## <none>                                        1153.8 261.21
## - ageCent                          1    64.179 1218.0 264.36
## - salary_millionsCent:w_pctCent  1   144.592 1298.4 270.43
##
## Step:  AIC=259.34
## TWITTER_FOLLOWER_COUNT_MILLIONS ~ ageCent + def_ratingCent +
##     PIECent + reb_pctCent + salary_millionsCent + w_pctCent +
##     ptsCent + ACTIVE_TWITTER_LAST_YEAR + salary_millionsCent:w_pctCent +
##     PIECent:ptsCent
##
##                                  Df Sum of Sq    RSS    AIC
## - PIECent:ptsCent                 1     5.074 1160.5 257.76
## - reb_pctCent                     1     5.330 1160.7 257.78
## - def_ratingCent                  1    10.283 1165.7 258.18
## - ACTIVE_TWITTER_LAST_YEAR        1    24.144 1179.5 259.31
## <none>                                        1155.4 259.34
## - ageCent                          1    64.470 1219.9 262.50
## - salary_millionsCent:w_pctCent  1   156.501 1311.9 269.41
##
## Step:  AIC=257.76
## TWITTER_FOLLOWER_COUNT_MILLIONS ~ ageCent + def_ratingCent +
##     PIECent + reb_pctCent + salary_millionsCent + w_pctCent +
##     ptsCent + ACTIVE_TWITTER_LAST_YEAR + salary_millionsCent:w_pctCent
##
##                                  Df Sum of Sq    RSS    AIC
## - ptsCent                         1     6.132 1166.6 256.26
## - reb_pctCent                     1     8.021 1168.5 256.41
## - def_ratingCent                  1    10.161 1170.6 256.58
## - PIECent                         1    12.392 1172.9 256.77
## - ACTIVE_TWITTER_LAST_YEAR        1    23.951 1184.4 257.70
## <none>                                        1160.5 257.76
## - ageCent                          1    70.150 1230.6 261.33
## - salary_millionsCent:w_pctCent  1   164.153 1324.6 268.32
##
## Step:  AIC=256.26
## TWITTER_FOLLOWER_COUNT_MILLIONS ~ ageCent + def_ratingCent +
##     PIECent + reb_pctCent + salary_millionsCent + w_pctCent +
##     ACTIVE_TWITTER_LAST_YEAR + salary_millionsCent:w_pctCent
##
```

```
##                                 Df Sum of Sq    RSS    AIC
## - reb_pctCent                    1     3.143 1169.7 254.51
## - def_ratingCent                 1    11.861 1178.5 255.22
## <none>                                       1166.6 256.26
## - ACTIVE_TWITTER_LAST_YEAR       1    25.646 1192.2 256.32
## - PIECent                        1    31.477 1198.1 256.79
## - ageCent                        1    64.560 1231.2 259.37
## - salary_millionsCent:w_pctCent  1   158.085 1324.7 266.33
##
## Step:  AIC=254.51
## TWITTER_FOLLOWER_COUNT_MILLIONS ~ ageCent + def_ratingCent +
##     PIECent + salary_millionsCent + w_pctCent + ACTIVE_TWITTER_LAST_YEAR +
##     salary_millionsCent:w_pctCent
##
##                                 Df Sum of Sq    RSS    AIC
## - def_ratingCent                 1    11.079 1180.8 253.41
## <none>                                       1169.7 254.51
## - ACTIVE_TWITTER_LAST_YEAR       1    25.350 1195.1 254.55
## - PIECent                        1    36.705 1206.5 255.45
## - ageCent                        1    62.548 1232.3 257.46
## - salary_millionsCent:w_pctCent  1   155.368 1325.1 264.36
##
## Step:  AIC=253.41
## TWITTER_FOLLOWER_COUNT_MILLIONS ~ ageCent + PIECent + salary_millionsCent +
##     w_pctCent + ACTIVE_TWITTER_LAST_YEAR + salary_millionsCent:w_pctCent
##
##                                 Df Sum of Sq    RSS    AIC
## - ACTIVE_TWITTER_LAST_YEAR       1    25.051 1205.9 253.40
## <none>                                       1180.8 253.41
## - PIECent                        1    32.005 1212.8 253.95
## - ageCent                        1    54.622 1235.4 255.70
## - salary_millionsCent:w_pctCent  1   162.024 1342.8 263.62
##
## Step:  AIC=253.4
## TWITTER_FOLLOWER_COUNT_MILLIONS ~ ageCent + PIECent + salary_millionsCent +
##     w_pctCent + salary_millionsCent:w_pctCent
##
##                                 Df Sum of Sq    RSS    AIC
## <none>                                       1205.9 253.40
## - PIECent                        1    29.646 1235.5 253.71
## - ageCent                        1    60.024 1265.9 256.02
## - salary_millionsCent:w_pctCent  1   158.457 1364.3 263.13
```

```
tidy(m1_aic, conf.int = TRUE) %>%
  kable(format = "markdown", digits = 3)
```

term	estimate	std.error	statistic	p.value	conf.low	conf.high
(Intercept)	1.175	0.397	2.957	0.004	0.386	1.964
ageCent	0.225	0.107	2.105	0.038	0.013	0.437
PIECent	27.461	18.565	1.479	0.143	-9.427	64.349
salary_millionsCent	0.111	0.051	2.158	0.034	0.009	0.213
w_pctCent	6.294	2.761	2.280	0.025	0.808	11.780
salary_millionsCent:w_pctCent	1.019	0.298	3.420	0.001	0.427	1.611

Based on the output displayed above from the first iteration of backward selection (using AIC as the selection criterion), seven main effect terms (`ast_ratioCent` `off_ratingCent`, `def_ratingCent`, `reb_pctCent`, `usg_pctCent`, `ptsCent`, and `ACTIVE_TWITTER_LAST_YEAR1`) and one interaction term (`PIECent` * `ptsCent`) were removed.

Hence, the equation of the selected linear model is: `TWITTER_FOLLOWER_COUNT_MILLIONS`-hat = 1.175 + 0.225 * `ageCent` + 27.461 * `PIECent` + 0.111 * `salary_millionsCent` + 6.294 * `w_pctCent` + 1.019 * `salary_millionsCent * w_pctCent`.

However, `PIECent`, which has the highest slope coefficient in the linear model, has a high standard error and a high p-value (0.143). Furthermore, the confidence interval for the slope coefficient is extremely wide (-9.427 to 64.349), not to mention the fact that it includes zero. So, we can reasonably infer `PIECent` may be a particularly troublesome predictor in the model (especially since it appears to not significantly affect the response, `TWITTER_FOLLOWER_COUNT_MILLIONS`).

We will proceed with the second iteration of backward selection and will revisit the issue of `PIECent` after viewing the selected linear regression model based on adjusted R-squared.

Backward Selection (Iteration 2)

Next, we will perform the second iteration of backward selection using adjusted R-squared as the selection criterion:

```
m1_adjR <- regsubsets(TWITTER_FOLLOWER_COUNT_MILLIONS ~ ageCent +
            ast_ratioCent +
            off_ratingCent +
            def_ratingCent +
            PIECent +
            reb_pctCent +
            usg_pctCent +
            salary_millionsCent +
            w_pctCent +
            ptsCent +
            ACTIVE_TWITTER_LAST_YEAR +
            salary_millionsCent * w_pctCent +
            PIECent * ptsCent,
          data = nba_social_power_mod,
          method = "backward")

select_summary <- summary(m1_adjR)
coef(m1_adjR, which.max(select_summary$adjr2))
```

```
##                   (Intercept)                       ageCent
##                    -2.3794214                     0.2146898
##                       PIECent           salary_millionsCent
##                    28.5598883                     0.1148099
##                     w_pctCent     ACTIVE_TWITTER_LAST_YEAR1
##                     6.6375685                     3.6257226
## salary_millionsCent:w_pctCent
##                     1.0306857
```

Based on the output displayed above from the second iteration of backward selection (using adjusted R-squared as the selection criterion), six main effect terms (`ast_ratioCent` `off_ratingCent`, `def_ratingCent`, `reb_pctCent`, `usg_pctCent`, and `ptsCent`) and one interaction term (`PIECent` * `ptsCent`) were removed.

Hence, the equation of the selected linear model is: `TWITTER_FOLLOWER_COUNT_MILLIONS`-hat = -2.379 +

0.215 * ageCent + 28.560 * PIECent + 0.115 * salary_millionsCent + 6.638 * w_pctCent + 3.626 * ACTIVE_TWITTER_LAST_YEAR1 + 1.031 * salary_millionsCent * w_pct_Cent.

Model Comparison: AIC vs. Adjusted R-squared

We notice that the model selected using adjusted R-squared as the selection criterion includes an additional categorical term (ACTIVE_TWITTER_LAST_YEAR1) which was omitted in the model selected based on the first iteration of backward selection (using AIC as the selection criterion).

If we closely investigate the results of the step() function, we can see that ACTIVE_TWITTER_LAST_YEAR was the last predictor removed during the first iteration of backward selection. In fact, the difference in AIC between the final two steps is only 253.41 - 253.4 = 0.01.

This warrants further consideration:

```
m1_modV1 <- lm(TWITTER_FOLLOWER_COUNT_MILLIONS ~ ageCent +
                 salary_millionsCent +
                 PIECent +
                 w_pctCent +
                 ACTIVE_TWITTER_LAST_YEAR +
                 salary_millionsCent * w_pctCent,
               data = nba_social_power_mod)

tidy(m1_modV1, conf.int = TRUE) %>%
  kable(format = "markdown", digits = 3)
```

term	estimate	std.error	statistic	p.value	conf.low	conf.high
(Intercept)	-2.379	2.631	-0.904	0.368	-7.608	2.850
ageCent	0.215	0.106	2.018	0.047	0.003	0.426
salary_millionsCent	0.115	0.051	2.244	0.027	0.013	0.216
PIECent	28.560	18.493	1.544	0.126	-8.190	65.310
w_pctCent	6.638	2.759	2.406	0.018	1.155	12.120
ACTIVE_TWITTER_LAST_YEAR1	3.626	2.654	1.366	0.175	-1.648	8.899
salary_millionsCent:w_pctCent	1.031	0.297	3.475	0.001	0.441	1.620

Based on the p-value associated with the slope coefficient for ACTIVE_TWITTER_LAST_YEAR1 (0.175) and the confidence interval associated with this coefficient (-1.648 to 8.899; includes zero), it is clear that ACTIVE_TWITTER_LAST_YEAR is not a significant predictor of TWITTER_FOLLOWER_COUNT_MILLIONS.
Again, recalling our objective in this analysis–to predict Twitter follower counts of NBA players–we choose to leave ACTIVE_TWITTER_LAST_YEAR out of the final model because, although the information provided by this categorical variable may, at first glance, appear relevant to the response, TWITTER_FOLLOWER_COUNT_MILLIONS (which measures players' total Twitter follower counts), the p-value and confidence interval associated with this predictor's slope coefficient suggests otherwise. When designing a model for predictive purposes, we are mostly interested in variables which are strong predictors of the response. Hence, we concur with the model selected by backward selection using AIC as the selection criterion in regard to ACTIVE_TWITTER_LAST_YEAR.

Now, as promised, we return to the issue of PIECent. Unfortunately, both selected models included PIECent; thus, we must decide whether to keep the variable in the model, or to ignore the results from the two iterations of backward selection and remove it.

To answer this question, we will compare the AIC and adjusted R-squared values for the model selected based on AIC as the selection criterion (first iteration) and the same model except without PIECent, remembering that we have already excluded ACTIVE_TWITTER_LAST_YEAR:

```
m1_modV2 <- lm(TWITTER_FOLLOWER_COUNT_MILLIONS ~ ageCent +
                salary_millionsCent +
                PIECent +
                w_pctCent +
                salary_millionsCent * w_pctCent,
              data = nba_social_power_mod)
```

```
m1_modV3 <- lm(TWITTER_FOLLOWER_COUNT_MILLIONS ~ ageCent +
                  salary_millionsCent +
                  w_pctCent +
                  salary_millionsCent * w_pctCent,
                data = nba_social_power_mod)
```

```
glance(m1_modV2)$AIC
```

```
## [1] 525.001
```

```
glance(m1_modV2)$adj.r.squared
```

```
## [1] 0.3141706
```

```
glance(m1_modV3)$AIC
```

```
## [1] 525.3083
```

```
glance(m1_modV3)$adj.r.squared
```

```
## [1] 0.3051171
```

Based on the output of the `glance()` function, the AIC of the model with `PIECent` is 525.001. Conversely, the AIC of the model without `PIECent` is 525.3083.

Moreover, the adjusted R-squared value for the model with `PIECent` is roughly 0.314, whereas the adjusted R-squared value for the model without `PIECent` is about 0.305.

Therefore, the model with `PIECent` maximizes adjusted R-squared and minimizes AIC. But our qualms with the large standard error, high p-value, and wide confidence interval (including zero) cannot be ignored, especially considering the purpose of our analysis–prediction. Hence, as with `ACTIVE_TWITTER_LAST_YEAR`, we will leave `PIECent` out of the model because including a variable which is not a strong predictor of the response will result in wider prediction intervals.

Of course, this means the only measures of athletic success remaining in the model are: mean-centered win percentage and (to a lesser degree due to salary cap restrictions) player salaries. Nevertheless, we have tailored our model selection phase towards our objective–predicting the number of Twitter followers–and cannot be too upset since such a model will yield relatively narrow prediction intervals.

Now, examining the current model:

```
tidy(m1_modV3, conf.int = TRUE) %>%
  kable(format = "markdown", digits = 3)
```

term	estimate	std.error	statistic	p.value	conf.low	conf.high
(Intercept)	1.142	0.399	2.859	0.005	0.348	1.935
ageCent	0.192	0.105	1.829	0.071	-0.017	0.401
salary_millionsCent	0.137	0.048	2.844	0.006	0.041	0.234
w_pctCent	7.131	2.720	2.622	0.010	1.727	12.535
salary_millionsCent:w_pctCent	1.099	0.295	3.729	0.000	0.514	1.685

But now we notice another problem: `ageCent` has a high p-value (0.071) and a confidence interval including zero. Hence, we will also remove `ageCent` using the same rationale as with `ACTIVE_TWITTER_LAST_YEAR` and `PIECent`.

Final Model

Thus, our final model is:

```
m_final <- lm(TWITTER_FOLLOWER_COUNT_MILLIONS ~ salary_millionsCent +
                w_pctCent +
                salary_millionsCent * w_pctCent,
              data = nba_social_power_mod)

tidy(m_final, conf.int = TRUE) %>%
  kable(format = "markdown", digits = 3)
```

term	estimate	std.error	statistic	p.value	conf.low	conf.high
(Intercept)	1.184	0.404	2.933	0.004	0.382	1.986
salary_millionsCent	0.163	0.047	3.479	0.001	0.070	0.256
w_pctCent	7.761	2.733	2.840	0.006	2.334	13.189
salary_millionsCent:w_pctCent	0.997	0.293	3.401	0.001	0.415	1.579

The equation of the final model is: `TWITTER_FOLLOWER_COUNT_MILLIONS`-hat = 1.184 + 0.163 * `salary_millionsCent` + 7.761 * `w_pctCent` + 0.997 * `salary_millionsCent * w_pct_Cent`.

Impact of Prominent Players

Lastly, before proceeding to discuss assumptions, we would like to examine the impact of including versus excluding prominent athletes in our model. Since he is widely regarded as one of the best NBA players of all-time, we will use LeBron James as a case study for preliminary analysis:

```
nba_social_power_mod %>%
  arrange(desc(SALARY_MILLIONS)) %>%
  select(PLAYER_NAME, SALARY_MILLIONS) %>%
  head(10)
```

```
## # A tibble: 10 x 2
##    PLAYER_NAME       SALARY_MILLIONS
##    <chr>                       <dbl>
##  1 LeBron James                 31.0
##  2 Russell Westbrook            26.5
##  3 Kevin Durant                 26.5
##  4 Mike Conley                  26.5
##  5 DeMar DeRozan                26.5
##  6 Al Horford                   26.5
##  7 James Harden                 26.5
##  8 Dirk Nowitzki                25
##  9 Carmelo Anthony              24.6
## 10 Damian Lillard               24.3
```

```
nba_social_power_mod %>%
  arrange(desc(TWITTER_FOLLOWER_COUNT_MILLIONS)) %>%
```

```
  select(PLAYER_NAME, TWITTER_FOLLOWER_COUNT_MILLIONS) %>%
  head(10)
```

```
## # A tibble: 10 x 2
##    PLAYER_NAME       TWITTER_FOLLOWER_COUNT_MILLIONS
##    <chr>                                       <dbl>
##  1 LeBron James                                37
##  2 Kevin Durant                                16.2
##  3 Stephen Curry                                9.56
##  4 Carmelo Anthony                              8.94
##  5 Dwyane Wade                                  7.01
##  6 Dwight Howard                                6.97
##  7 Chris Paul                                   6.4
##  8 Pau Gasol                                    6.38
##  9 Russell Westbrook                            4.5
## 10 James Harden                                 4.47
```

Based on the tables above, it is seems like LeBron James is an outlier, both in regard to his annual salary and Twitter follower count. So, we will remove LeBron from the dataset and see how the model changes:

```
nba_social_power_mod2 <- nba_social_power_mod %>%
  filter(PLAYER_NAME != "LeBron James")

glimpse(nba_social_power_mod2)
```

```
## Observations: 94
## Variables: 26
## $ PLAYER_NAME                    <chr> "Russell Westbrook", "Demetriu...
## $ TEAM_ABBREVIATION              <chr> "OKC", "BOS", "NOP", "HOU", "G...
## $ AGE                            <int> 28, 22, 24, 27, 28, 32, 26, 22...
## $ W_PCT                          <dbl> 0.568, 0.200, 0.413, 0.667, 0....
## $ OFF_RATING                     <dbl> 107.9, 124.2, 104.2, 113.6, 11...
## $ DEF_RATING                     <dbl> 104.6, 117.8, 102.5, 107.3, 10...
## $ NET_RATING                     <dbl> 3.3, 6.3, 1.7, 6.3, 16.0, 14.9...
## $ AST_RATIO                      <dbl> 23.4, 31.1, 7.3, 27.6, 18.4, 3...
## $ REB_PCT                        <dbl> 0.167, 0.103, 0.170, 0.123, 0....
## $ USG_PCT                        <dbl> 0.408, 0.172, 0.326, 0.341, 0....
## $ PIE                            <dbl> 0.230, 0.194, 0.192, 0.190, 0....
## $ SALARY_MILLIONS                <dbl> 26.54, 1.45, 22.12, 26.50, 26....
## $ ACTIVE_TWITTER_LAST_YEAR       <fct> 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, ...
## $ TWITTER_FOLLOWER_COUNT_MILLIONS <dbl> 4.500, 0.049, 1.220, 4.470, 16...
## $ PTS                            <dbl> 31.6, 2.0, 28.0, 29.1, 25.1, 1...
## $ ageCent                        <dbl> 0.6105263, -5.3894737, -3.3894...
## $ ast_ratioCent                  <dbl> 6.274737, 13.974737, -9.825263...
## $ off_ratingCent                 <dbl> -0.009473684, 16.290526316, -3...
## $ def_ratingCent                 <dbl> -1.39368421, 11.80631579, -3.4...
## $ net_ratingCent                 <dbl> 1.3810526, 4.3810526, -0.21894...
## $ PIECent                        <dbl> 0.09073684, 0.05473684, 0.0527...
## $ reb_pctCent                    <dbl> 0.033778947, -0.030221053, 0.0...
## $ usg_pctCent                    <dbl> 0.170, -0.066, 0.088, 0.103, 0...
## $ salary_millionsCent            <dbl> 15.2351368, -9.8548632, 10.815...
## $ w_pctCent                      <dbl> 0.056589474, -0.311410526, -0....
## $ ptsCent                        <dbl> 16.3176842, -13.2823158, 12.71...
```

Based on the `glimpse()` output, we can see LeBron has been removed from the dataset (94 observations

remaining).

Now, to assess the impact of his absence on the final model:

```
m_final_noLJ <- lm(TWITTER_FOLLOWER_COUNT_MILLIONS ~ salary_millionsCent +
                   w_pctCent +
                   salary_millionsCent * w_pctCent,
                 data = nba_social_power_mod2)

tidy(m_final_noLJ, conf.int = TRUE) %>%
  kable(format = "markdown", digits = 3)
```

term	estimate	std.error	statistic	p.value	conf.low	conf.high
(Intercept)	1.087	0.229	4.742	0.000	0.632	1.543
salary_millionsCent	0.109	0.027	4.035	0.000	0.055	0.162
w_pctCent	4.609	1.568	2.940	0.004	1.495	7.724
salary_millionsCent:w_pctCent	0.431	0.171	2.513	0.014	0.090	0.771

The equation of the final model without LeBron James is: `TWITTER_FOLLOWER_COUNT_MILLIONS`-hat = 1.087 + 0.109 * `salary_millionsCent` + 4.609 * `w_pctCent` + 0.431 * `salary_millionsCent * w_pct_Cent`.

```
nba_social_power_mod %>%
  arrange(desc(W_PCT)) %>%
  select(PLAYER_NAME, W_PCT) %>%
  head(15)
```

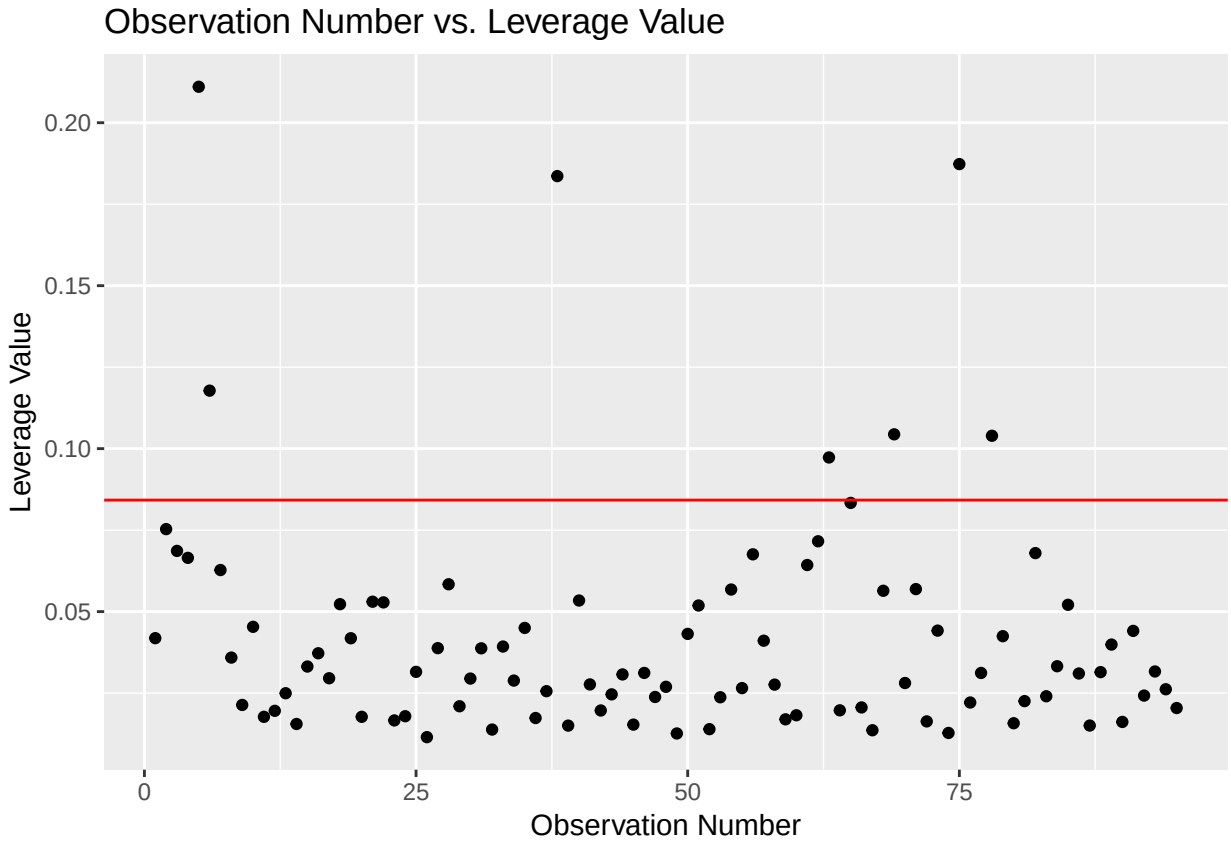
```
## # A tibble: 15 x 2
##    PLAYER_NAME       W_PCT
##    <chr>             <dbl>
##  1 David West        0.824
##  2 Kevin Durant      0.823
##  3 Stephen Curry     0.823
##  4 Draymond Green    0.816
##  5 JaVale McGee      0.805
##  6 Pau Gasol         0.734
##  7 David Lee         0.734
##  8 Kawhi Leonard     0.73
##  9 Dewayne Dedmon    0.724
## 10 LaMarcus Aldridge 0.722
## 11 Chris Paul        0.705
## 12 LeBron James      0.689
## 13 Clint Capela      0.677
## 14 Al Horford        0.676
## 15 George Hill       0.673
```

Comparing the two equations, we notice a relatively sizable discrepancy in the slope coefficient of `w_pctCent`. This makes sense since LeBron has the twelfth-highest win percentage in the league (0.689), so eliminating him from the dataset dramatically affects the average win percentage (as well as the spread, or standard deviation).

More generally, to determine whether prominent athletes are influential points, we will look at standardized residuals, leverage, and Cook's Distance:

```
augmented_data <- augment(m_final)

augmented_data <- augmented_data %>%
```

```
  mutate(obs_num = row_number())

threshold <- ((2 * (3 + 1)) / 95)

ggplot(data = augmented_data, aes(x = obs_num, y = .hat)) +
  geom_point() +
  geom_hline(yintercept = threshold, color = "red") +
  labs(x = "Observation Number",
       y = "Leverage Value",
       title = "Observation Number vs. Leverage Value")
```




```
augmented_data %>%
  filter(.hat > threshold) %>%
  select(obs_num, .hat)
```

```
## # A tibble: 7 x 2
##   obs_num   .hat
##     <int>  <dbl>
## 1       5 0.211
## 2       6 0.118
## 3      38 0.184
## 4      63 0.0973
## 5      69 0.104
## 6      75 0.187
## 7      78 0.104
```

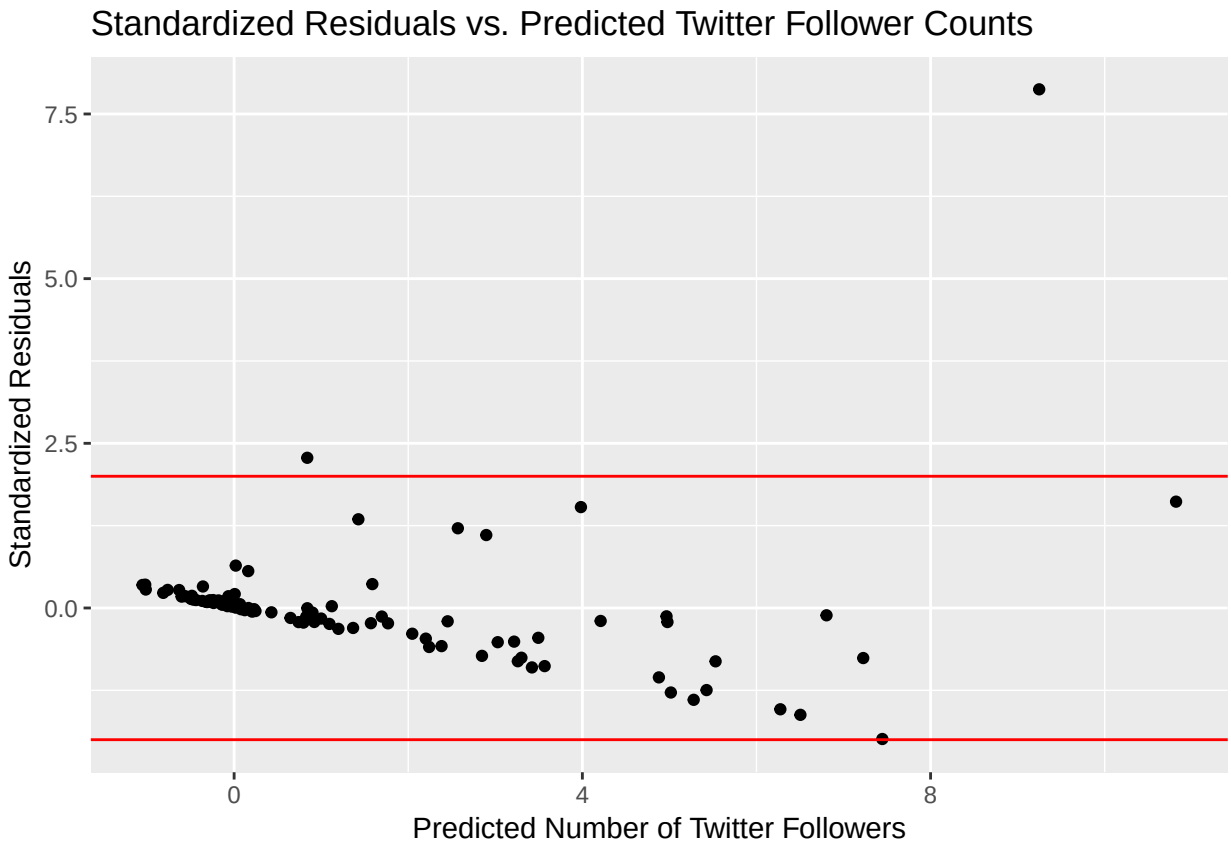
```
nba_leverage <- nba_social_power_mod %>%
  mutate(obs_num = row_number()) %>%
  filter(obs_num == 5 | obs_num == 6 | obs_num == 38 |
           obs_num == 63 | obs_num == 69 | obs_num == 75 |
           obs_num == 78)

nba_leverage %>%
  select(obs_num, PLAYER_NAME)
```

```
## # A tibble: 7 x 2
##   obs_num PLAYER_NAME
##     <int> <chr>
## 1       5 Kevin Durant
## 2       6 LeBron James
## 3      38 Josh Huestis
## 4      63 JaVale McGee
## 5      69 David West
## 6      75 Jarnell Stokes
## 7      78 Carmelo Anthony
```

Based on the threshold (`2(p + 1) / n`), Kevin Durant, LeBron James, Josh Huestis, JaVale McGee, David West, Jarnell Stokes, and Carmelo Anthony are considered high leverage players (and hence potential influential points).

Now, to identify outliers within these candidates, we will look at the standardized residuals:

```
ggplot(data = augmented_data, aes(x = .fitted, y = .std.resid)) +
  geom_point() +
  geom_hline(yintercept = 2, color = "red") +
  geom_hline(yintercept = -2, color = "red") +
  labs(x = "Predicted Number of Twitter Followers",
       y = "Standardized Residuals",
       title = "Standardized Residuals vs. Predicted Twitter Follower Counts")
```




```
augmented_data %>%
  filter(.std.resid > 2 | .std.resid < -2) %>%
  select(obs_num, .std.resid)
```

```
## # A tibble: 2 x 2
##   obs_num .std.resid
##     <int>      <dbl>
## 1       6       7.87
## 2      78       2.28
```

```
nba_std_resid <- nba_social_power_mod %>%
  mutate(obs_num = row_number()) %>%
  filter(obs_num == 6 | obs_num == 78)

nba_std_resid %>%
  select(obs_num, PLAYER_NAME)
```

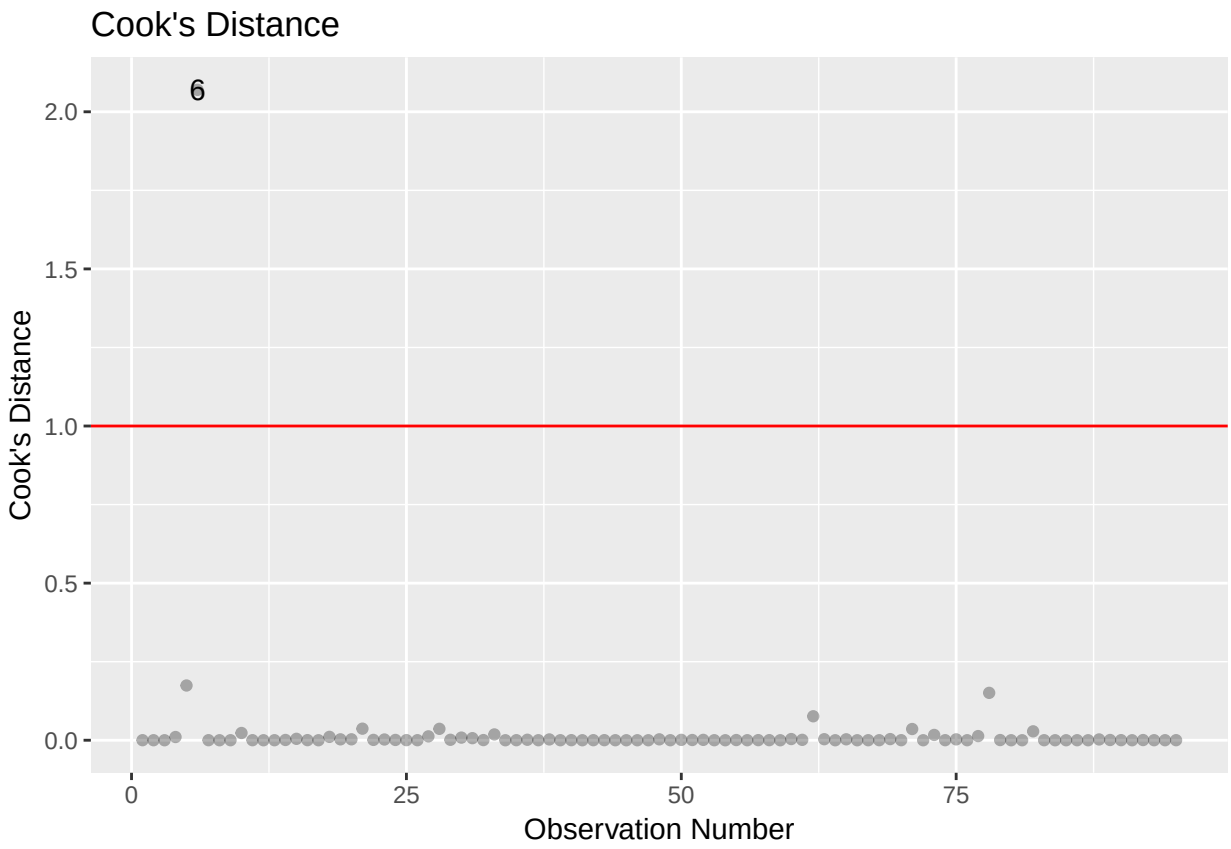
```
## # A tibble: 2 x 2
##   obs_num PLAYER_NAME
##     <int> <chr>
## 1       6 LeBron James
## 2      78 Carmelo Anthony
```

The players with standardized residuals of magnitude greater than 2 are LeBron James and Carmelo Anthony–two players who are also classified as high leverage points. By definition, LeBron and Carmelo are outliers; however, it remains to be seen whether Carmelo impacts the regression line (we already examined LeBron's effect).

To assess the impact of prominent athletes (identified by high leverage and/or high standardized residuals)

on the regression line, we examine Cook's Distance:

```
ggplot(data = augmented_data, aes(x = obs_num, y = .cooksd)) +
  geom_point(alpha = 0.3) +
  geom_hline(yintercept = 1, color = "red") +
  labs(x = "Observation Number",
       y = "Cook's Distance",
       title = "Cook's Distance") +
  geom_text(aes(label = ifelse(.cooksd > 1, as.character(obs_num), "")))
```




```
nba_std_resid %>%
  filter(obs_num == 6) %>%
  select(obs_num, PLAYER_NAME)
```

```
## # A tibble: 1 x 2
##   obs_num PLAYER_NAME
##     <int> <chr>
## 1       6 LeBron James
```

It is clear from the plot of Cook's Distance vs. observation number that LeBron James is the only influential point.

Since our objective is to accurately predict the Twitter follower counts of NBA players, it is probably best to exclude LeBron to avoid overestimating for less prominent athletes. Hence, we will continue our analysis by using the multiple linear regression model without LeBron James (`m_final_noLJ`).

To conclude, we will display the final model once more:

```
tidy(m_final_noLJ, conf.int = TRUE) %>%
  kable(format = "markdown", digits = 3)
```

term	estimate	std.error	statistic	p.value	conf.low	conf.high
(Intercept)	1.087	0.229	4.742	0.000	0.632	1.543
salary_millionsCent	0.109	0.027	4.035	0.000	0.055	0.162
w_pctCent	4.609	1.568	2.940	0.004	1.495	7.724
salary_millionsCent:w_pctCent	0.431	0.171	2.513	0.014	0.090	0.771

Discussion of Assumptions

According to our dataset, 5 observations have players who are missing Twitter handles. These values are missing at random because missingness depends on other observed variables (like the person's social media usage, etc.). The probability that a variable is missing depends on information not included in our dataset. We decided to remove the 5 players from our dataset because there were very few observations with missing values, and after removing these variables, we still had 95 observations. The observations with missingness are also random, so the resulting analysis will not be biased because the missingness does not differ systematically from the complete observations. In addition, since our objective is predicting the number of Twitter followers for NBA athletes, players without Twitter accounts are outside of the model's predictive capabilities and hence do not belong in the dataset.

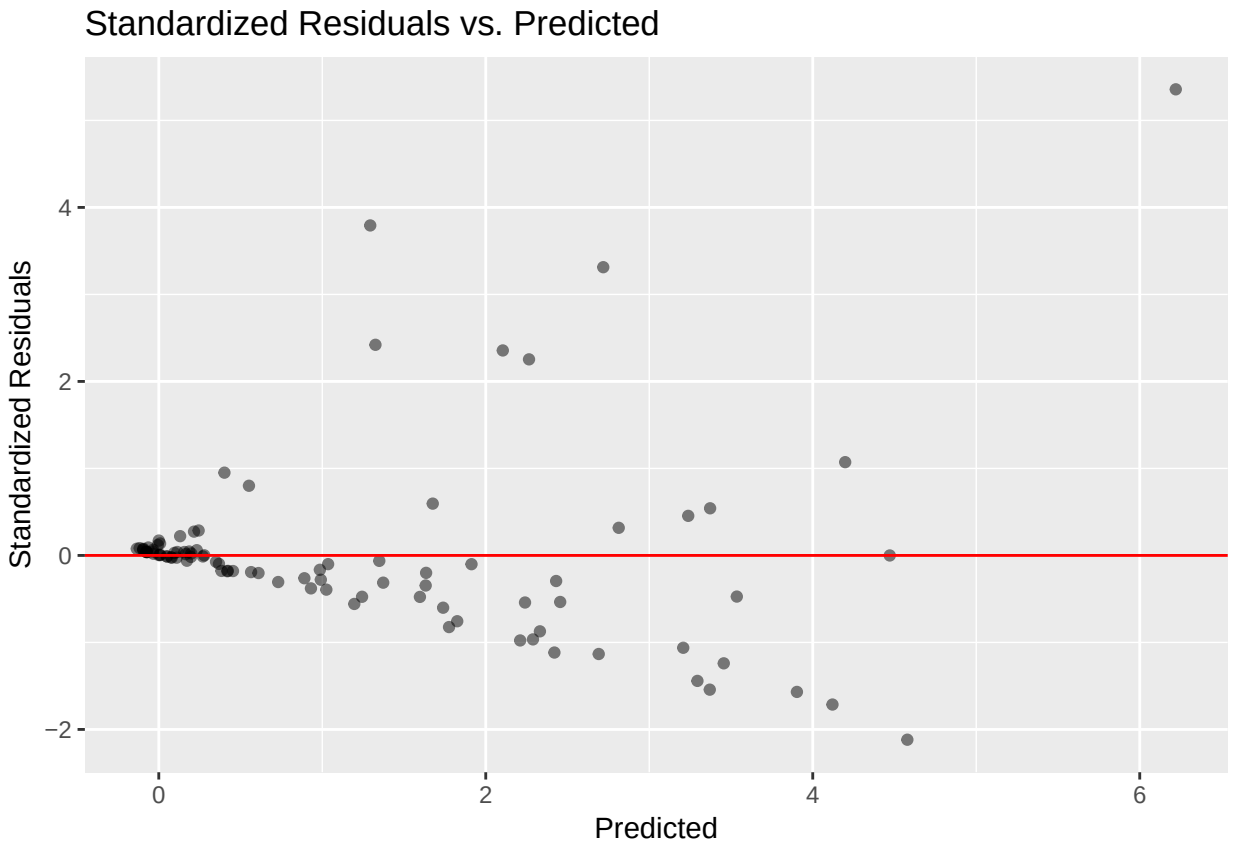
One other modification we have made to our original dataset is removing LeBron James. We decided, based on Cook's distance, standardized residuals, and leverage, LeBron was an influential point with a significant effect on the regression line. Consequently, we removed LeBron to avoid overestimation in our model. Because we removed LeBron, we will use standardized residuals from here on out

Therefore, we are left with 94 observations, which is still a large enough sample population to build a model which can be generalized to a larger population of NBA athletes.

Next, we will check the linearity, constant variance, normality, and independence assumptions to see if inference and predictions will be reliable.

First, we will check for linearity and whether the response variable has a linear relationship with the predictor variables in the model. We will check the plot of standardized residuals vs. predicted values:

```
m_final_aug <- augment(m_final_noLJ)
ggplot(data = m_final_aug, aes(x = .fitted, y = .std.resid)) +
  geom_point(alpha = 0.5) +
  geom_hline(yintercept = 0, color = "red") +
  labs(x = "Predicted", y = "Standardized Residuals",
       title = "Standardized Residuals vs. Predicted")
```

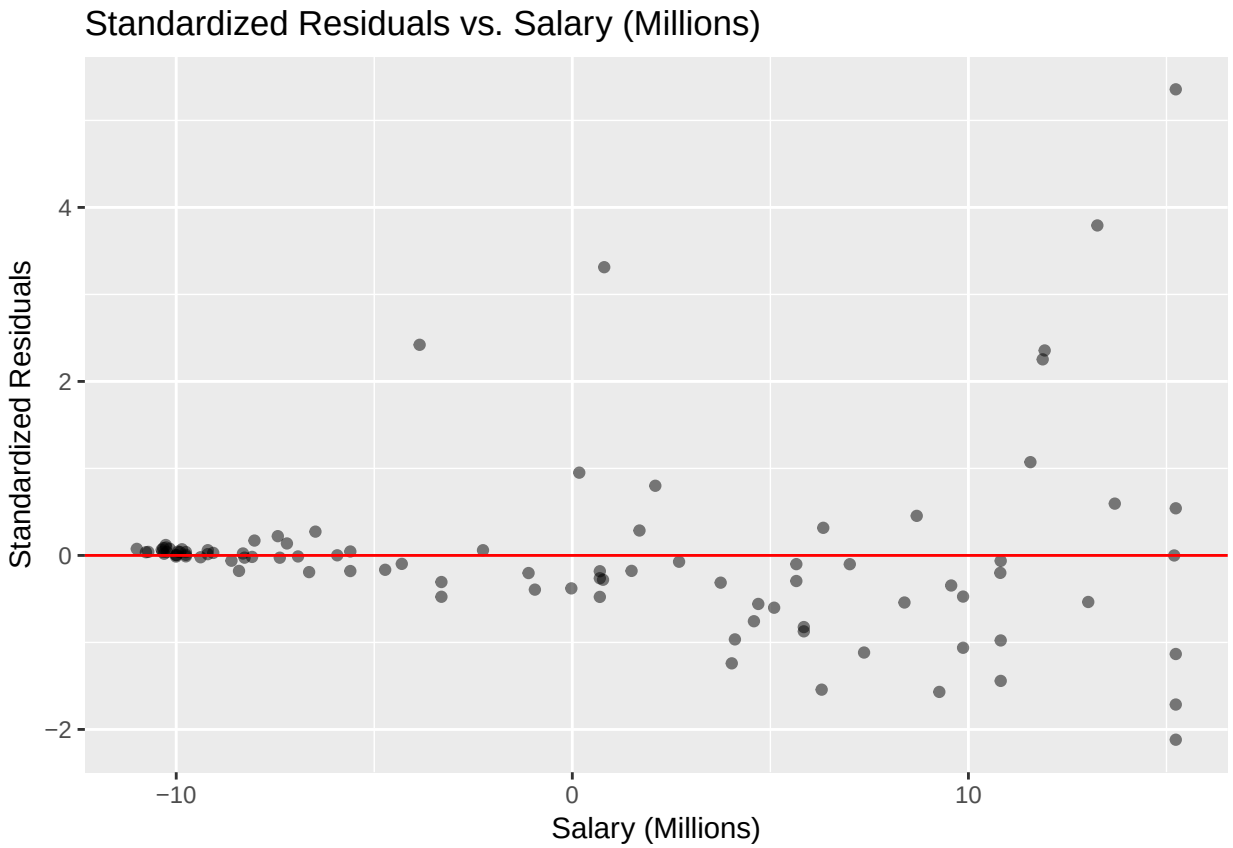


When observing for constant variance, the height of the cloud of points seems to be constant as you move from left to right. Therefore, constant variance is not satisfied.

There is no obvious pattern and the shape of the graph seems to be linear. Hence, this plot presents no issues with the linearity assumption.

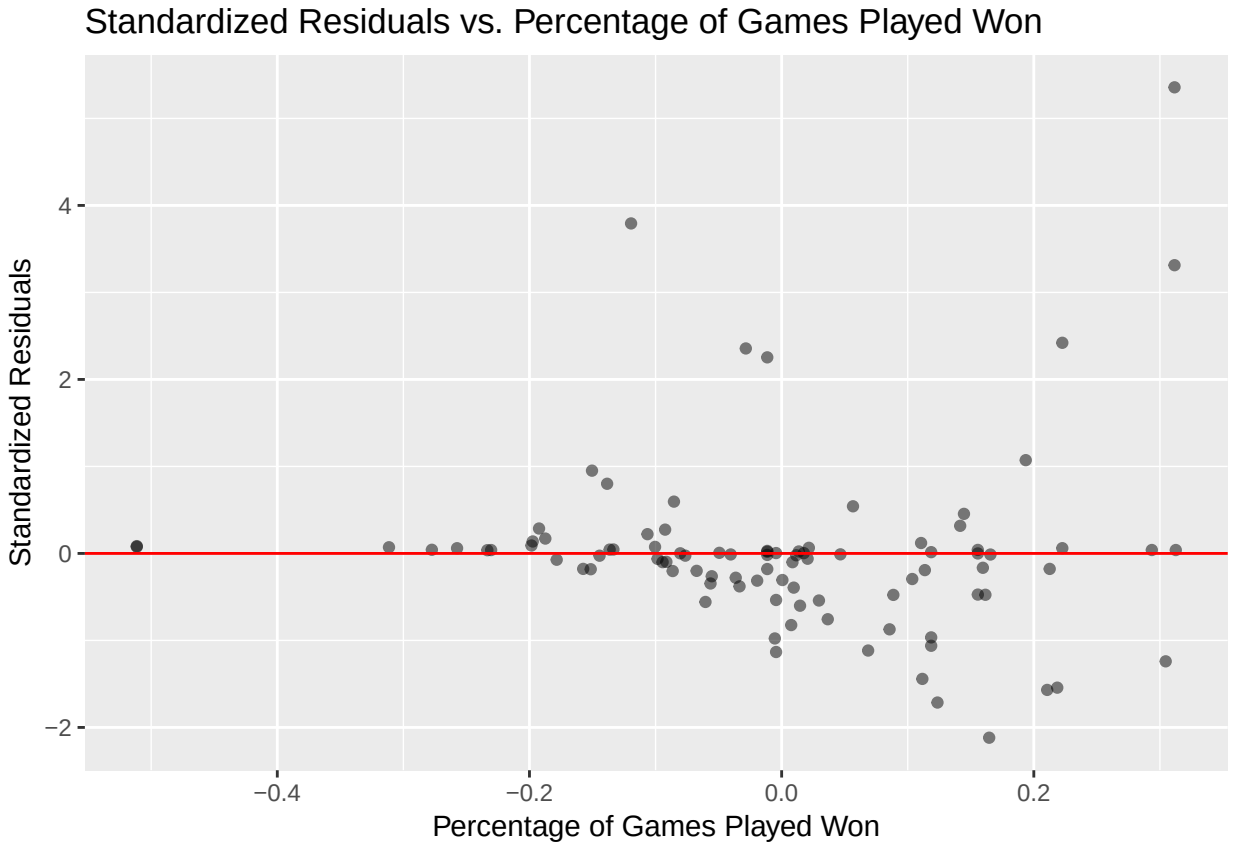
Next, we will observe the plot of residuals vs. predictors:

```
ggplot(data = m_final_aug, aes(x = salary_millionsCent, y = .std.resid)) +
  geom_point(alpha = 0.5) +
  geom_hline(yintercept = 0, color = "red") +
  labs(x = "Salary (Millions)", y = "Standardized Residuals",
       title = "Standardized Residuals vs. Salary (Millions)")
```



There looks to be a pattern in the plot because towards the end it curves upward drastically. Therefore, this pattern suggests that interactions or higher-order terms (like quadratic terms) are required. Thus, linearity is not satisfied.

```
ggplot(data = m_final_aug, aes(x = w_pctCent, y = .std.resid)) +
  geom_point(alpha = 0.5) +
  geom_hline(yintercept = 0, color = "red") +
  labs(x = "Percentage of Games Played Won", y = "Standardized Residuals",
       title = "Standardized Residuals vs. Percentage of Games Played Won")
```



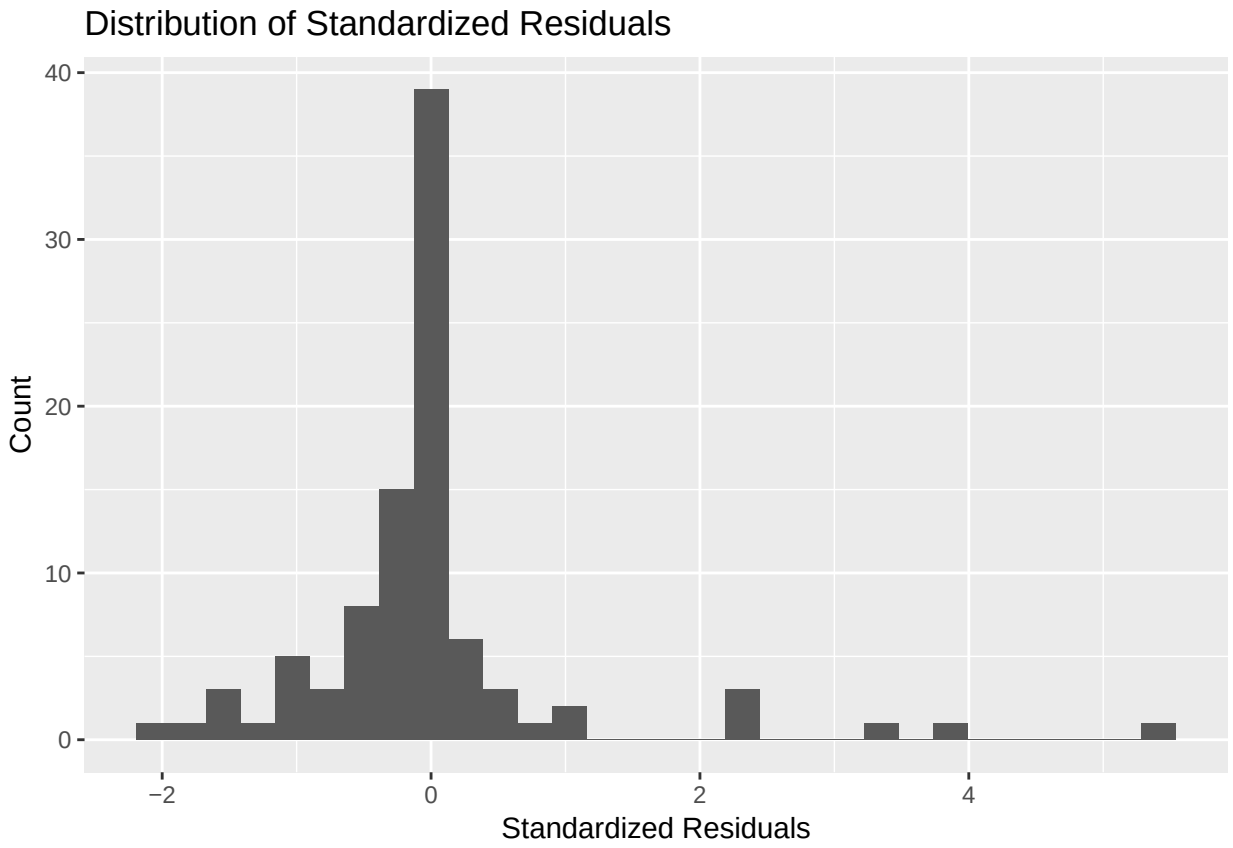
There looks to be a pattern in the plot because towards the end it also curves upward drastically. Therefore, this pattern suggests that interactions or higher-order terms (like quadratic terms) are required. Linearity is not satisfied.

Next, we will check for the normality assumption by creating a histogram of the residuals and a normal QQ-plot of the residuals.

```
nba_social_power_mod2 <- nba_social_power_mod2 %>%
  mutate(predicted = predict.lm(m_final_noLJ), std_resid = rstandard(m_final_noLJ))
```

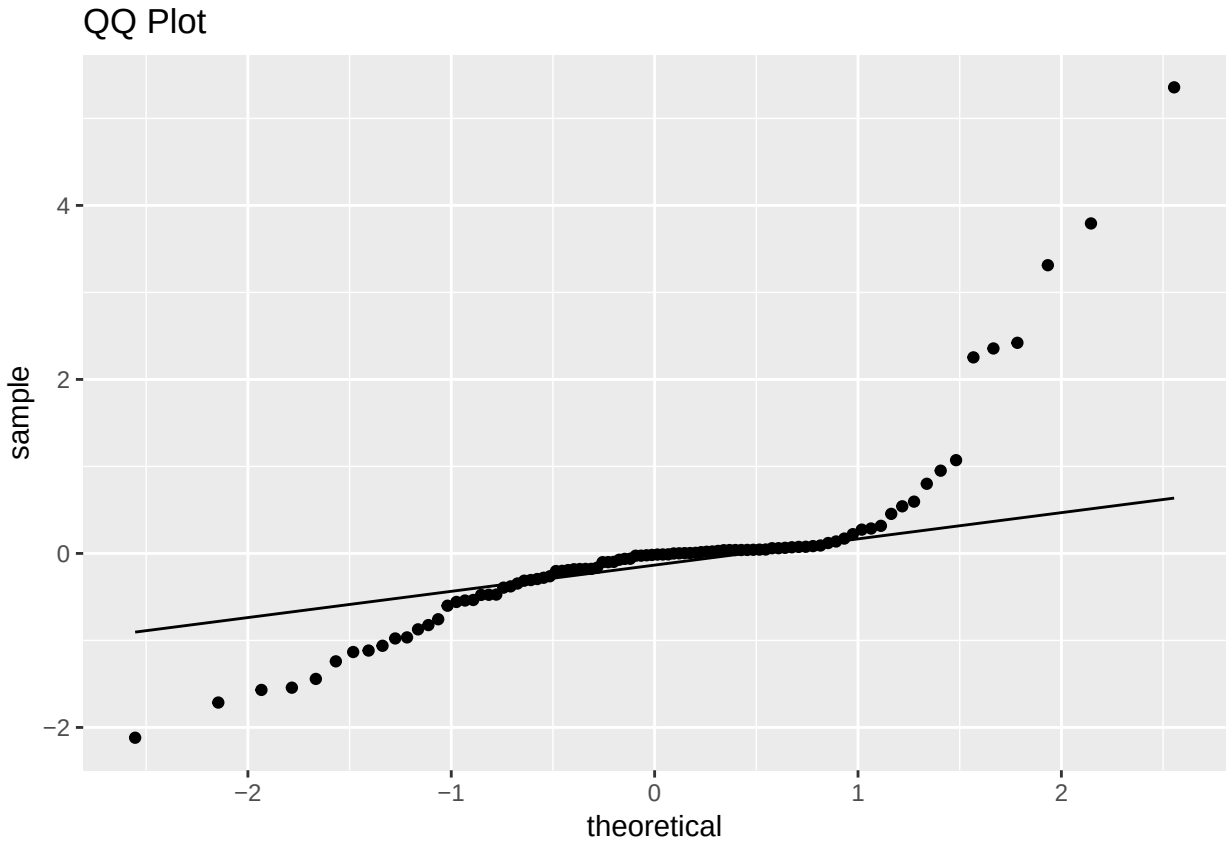
```
ggplot(data = nba_social_power_mod2, mapping = aes(x = std_resid)) + geom_histogram() + labs(title = "D
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Normality is supported because the histogram of residuals is normal because the distribution is unimodal and the distribution is symmetric.

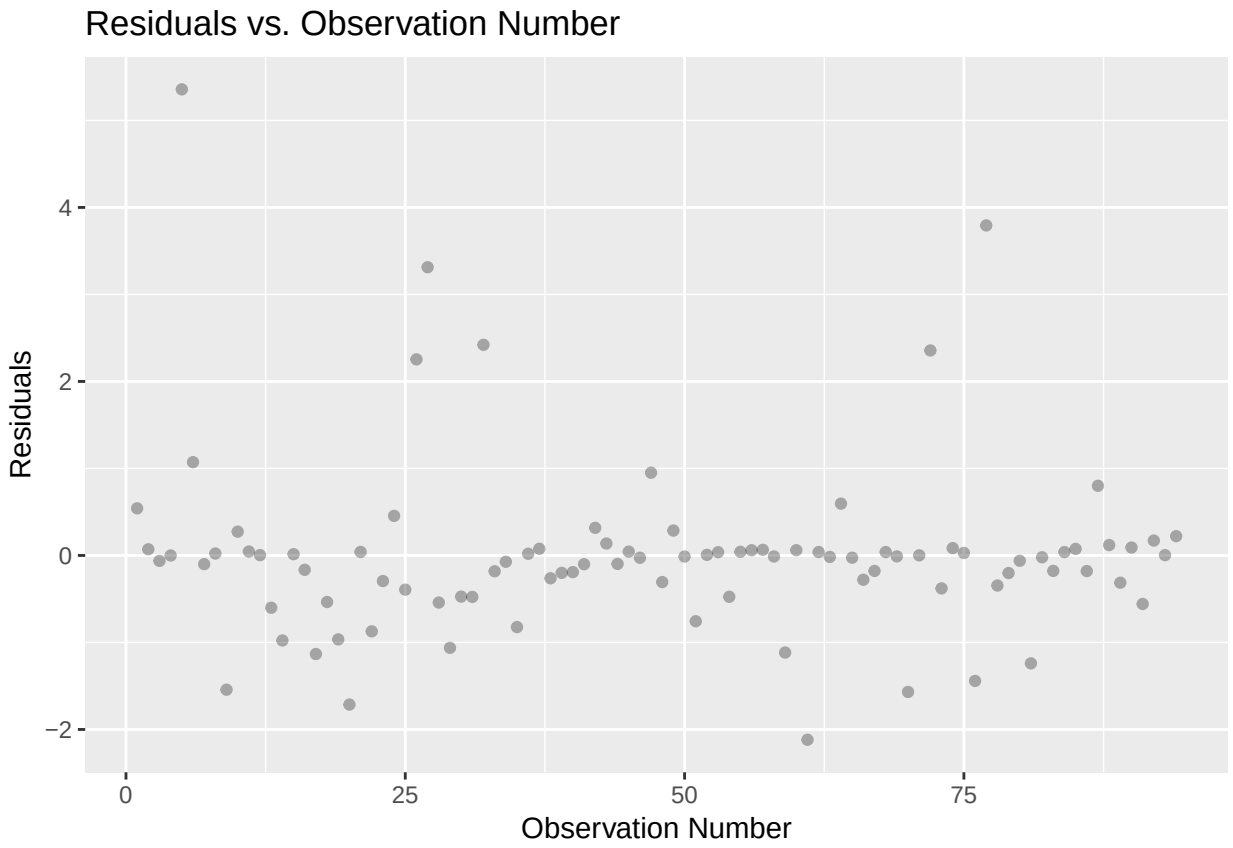
```
ggplot(data = nba_social_power_mod2, mapping = aes(sample = std_resid)) + stat_qq() + stat_qq_line() + l
```



However, normality is not satisfied because some of the points don't follow the diagonal line.

Lastly, we will check for independence. Data was not taken over time, so we know there is no temporal correlation. There is also no spatial correlation because data was not taken in space. We can check to see if there is some structure/order to the dataset according to observation number.

```
m_final_aug <- m_final_aug %>%
  mutate(obs_num = 1:nrow(m_final_aug))

ggplot(data = m_final_aug, aes(x = obs_num, y = .std.resid)) +
  geom_point(alpha = 0.3) +
  labs(x = "Observation Number", y = "Residuals",
       title = "Residuals vs. Observation Number")
```



Looking at the graph, there is no distinguishable pattern in the graph. The number of Twitter followers of one player will not affect the number of Twitter followers of another player. Therefore, independence is satisfied.

Therefore, because constant variance and normality and linearity are not satisfied, but independence is, we will make some adjustments to our model. We can adjust for linearity by adding squared terms for salary and percentage of games played won because those standardized residuals vs. predictor variable plots violated linearity. For constant variance, we can log-transform the response variable because our standardized residuals vs. predicted values violated constant variance. Lastly, for normality regression is robust to depatures from normality and the departure from normality is not extreme, so we will not make any adjustments.

After making these adjustments our model looks like this:

```
nba_social_power_mod2 <- nba_social_power_mod2 %>%
  mutate(TWITTER_FOLLOWER_COUNT_MILLIONS_LOG = log(TWITTER_FOLLOWER_COUNT_MILLIONS))

m_final_adjust <- lm(TWITTER_FOLLOWER_COUNT_MILLIONS_LOG ~ salary_millionsCent
                     + I(salary_millionsCent * salary_millionsCent)
                     + w_pctCent
                   + I(w_pctCent * w_pctCent)
                   + salary_millionsCent * w_pctCent,
                 data = nba_social_power_mod2)

tidy(m_final_adjust, conf.int = TRUE) %>%
  kable(format = "markdown", digits = 3)
```

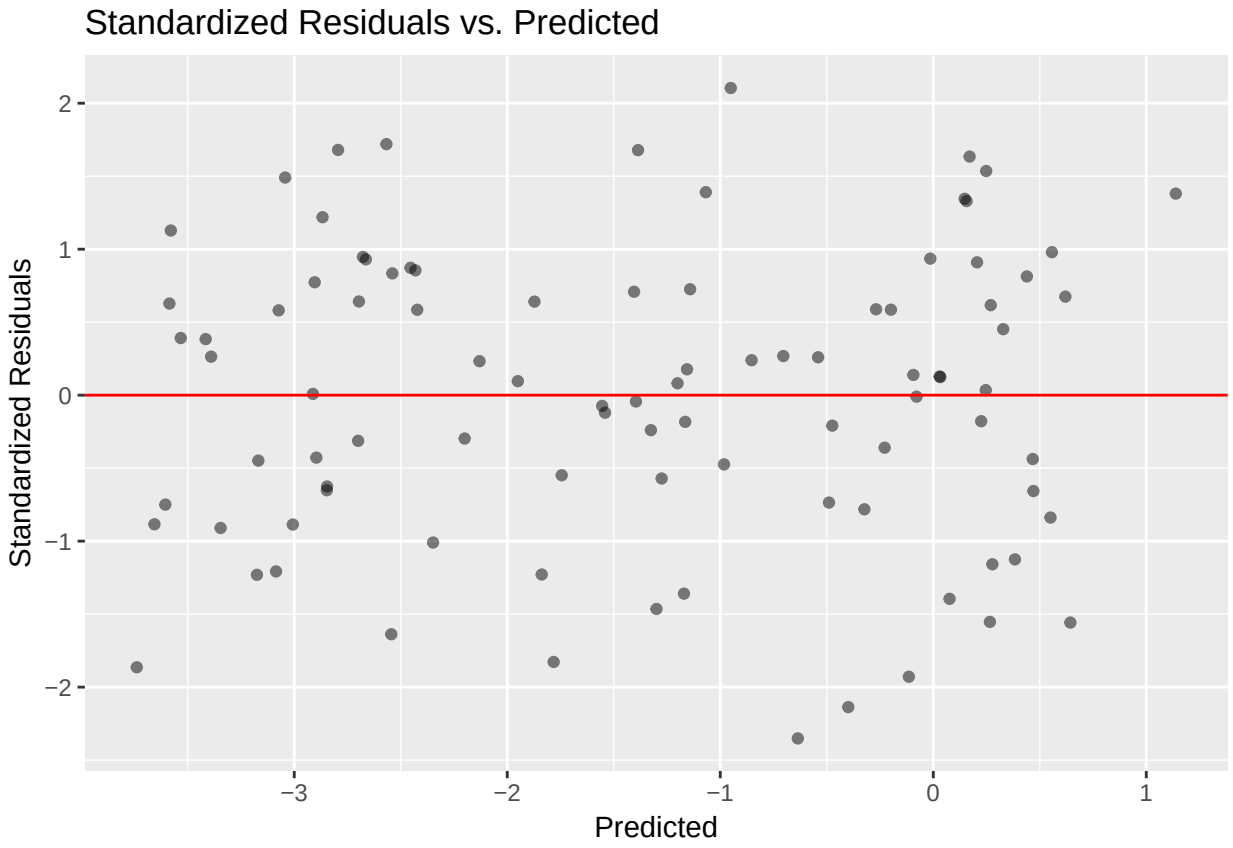
term	estimate	std.error	statistic	p.value	conf.low	conf.high
(Intercept)	-1.201	0.245	-4.894	0.000	-1.688	-0.713

term	estimate	std.error	statistic	p.value	conf.low	conf.high
salary_millionsCent	0.154	0.019	8.159	0.000	0.117	0.192
I(salary_millionsCent * salary_millionsCent)	-0.003	0.003	-1.280	0.204	-0.008	0.002
w_pctCent	2.299	1.022	2.250	0.027	0.268	4.330
I(w_pctCent * w_pctCent)	5.775	4.071	1.419	0.160	-2.314	13.864
salary_millionsCent:w_pctCent	-0.111	0.133	-0.828	0.410	-0.376	0.155

We will check for all assumptions again:

First, we will check for linearity and whether the response variable has a linear relationship with the predictor variables in the model. We will check the plot of standardized residuals vs. predicted values:

```
m_final_adj_aug <- augment(m_final_adjust)
ggplot(data = m_final_adj_aug, aes(x = .fitted, y = .std.resid)) +
  geom_point(alpha = 0.5) +
  geom_hline(yintercept = 0, color = "red") +
  labs(x = "Predicted", y = "Standardized Residuals",
       title = "Standardized Residuals vs. Predicted")
```

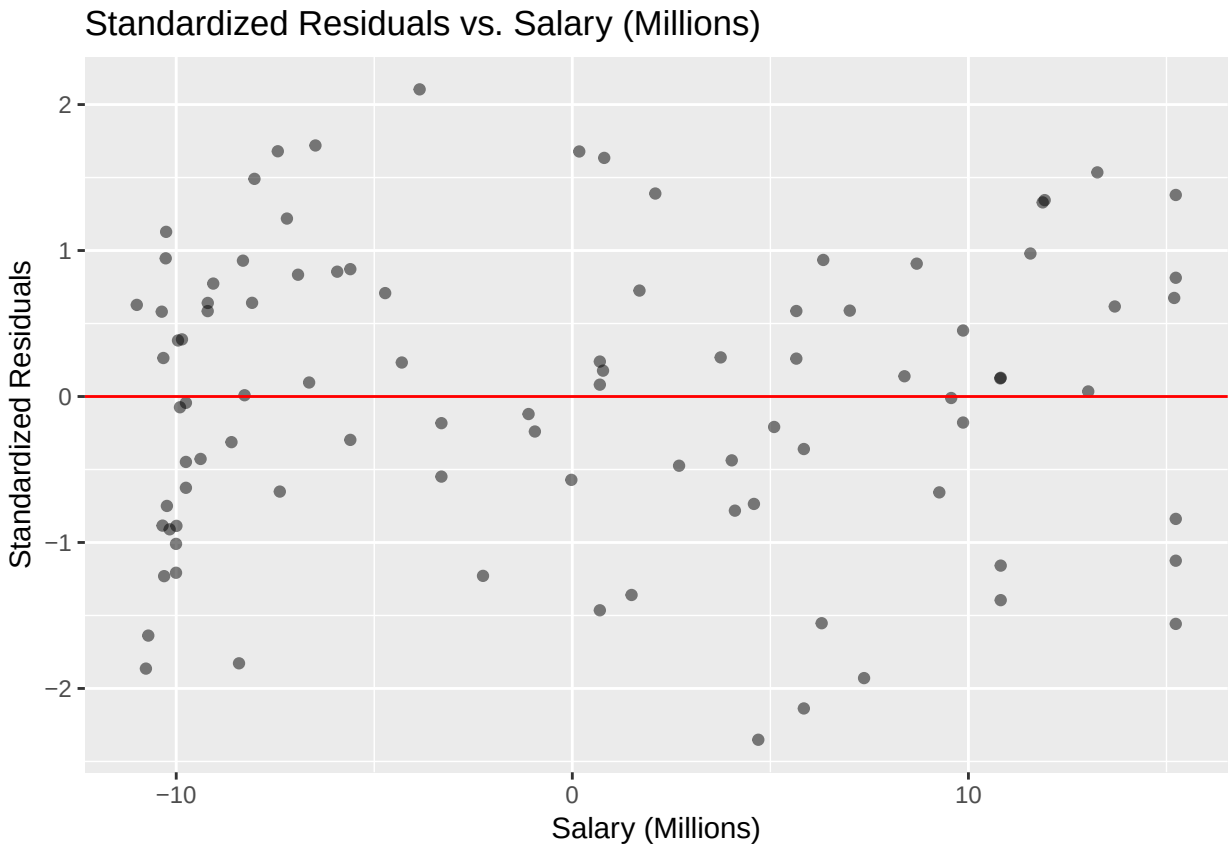


When observing for constant variance, the height of the cloud of points seems to increase as you move from left to right. Points are clustered at the very left then grow to be more spare as you move along the graph. Therefore, constant variance is not satisfied.

There is no obvious pattern and the shape of the graph seems to be linear. Hence, this plot presents no issues with the linearity assumption.

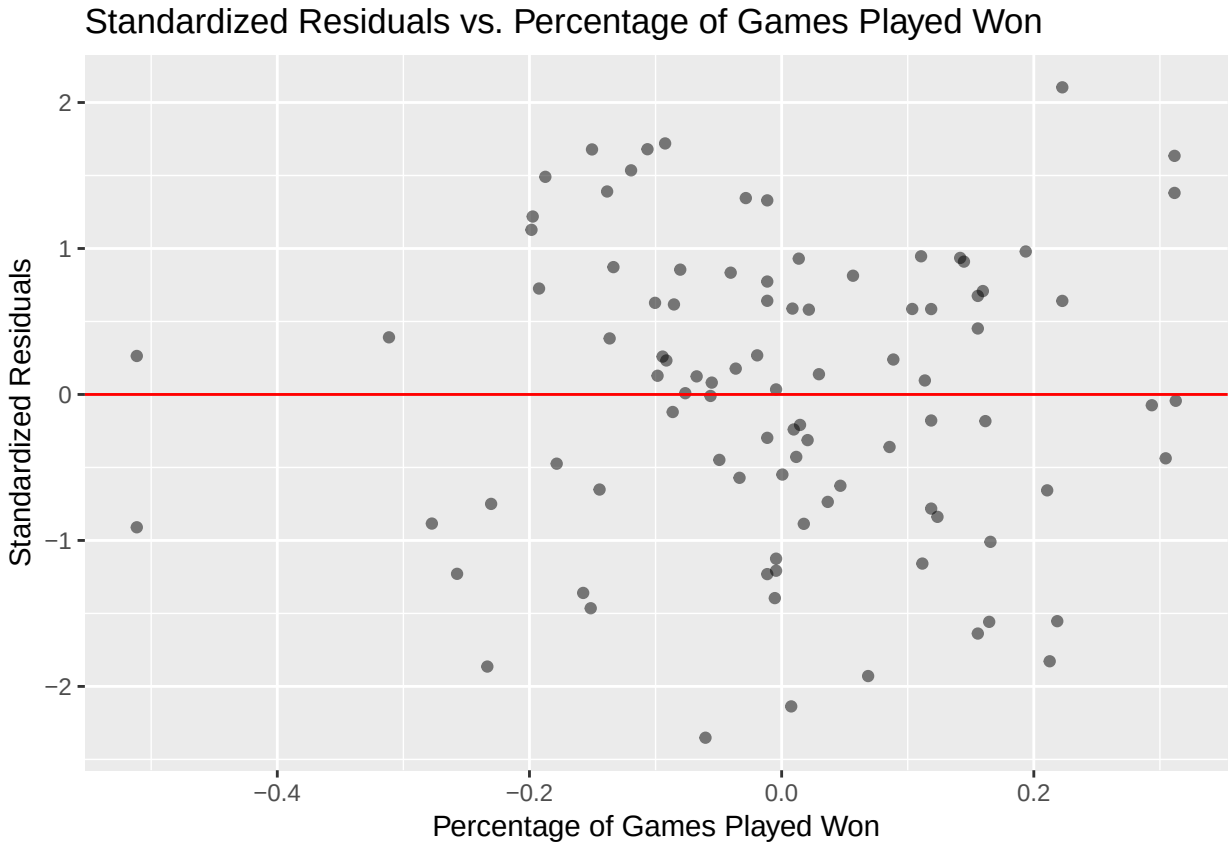
Next, we will observe the plot of residuals vs. predictors:

```
ggplot(data = m_final_adj_aug, aes(x = salary_millionsCent, y = .std.resid)) +
  geom_point(alpha = 0.5) +
  geom_hline(yintercept = 0, color = "red") +
  labs(x = "Salary (Millions)", y = "Standardized Residuals",
       title = "Standardized Residuals vs. Salary (Millions)")
```



There looks to be no distinguishable pattern in the plot. Thus, this plot does not violate linearity.

```
ggplot(data = m_final_adj_aug, aes(x = w_pctCent, y = .std.resid)) +
  geom_point(alpha = 0.5) +
  geom_hline(yintercept = 0, color = "red") +
  labs(x = "Percentage of Games Played Won", y = "Standardized Residuals",
       title = "Standardized Residuals vs. Percentage of Games Played Won")
```



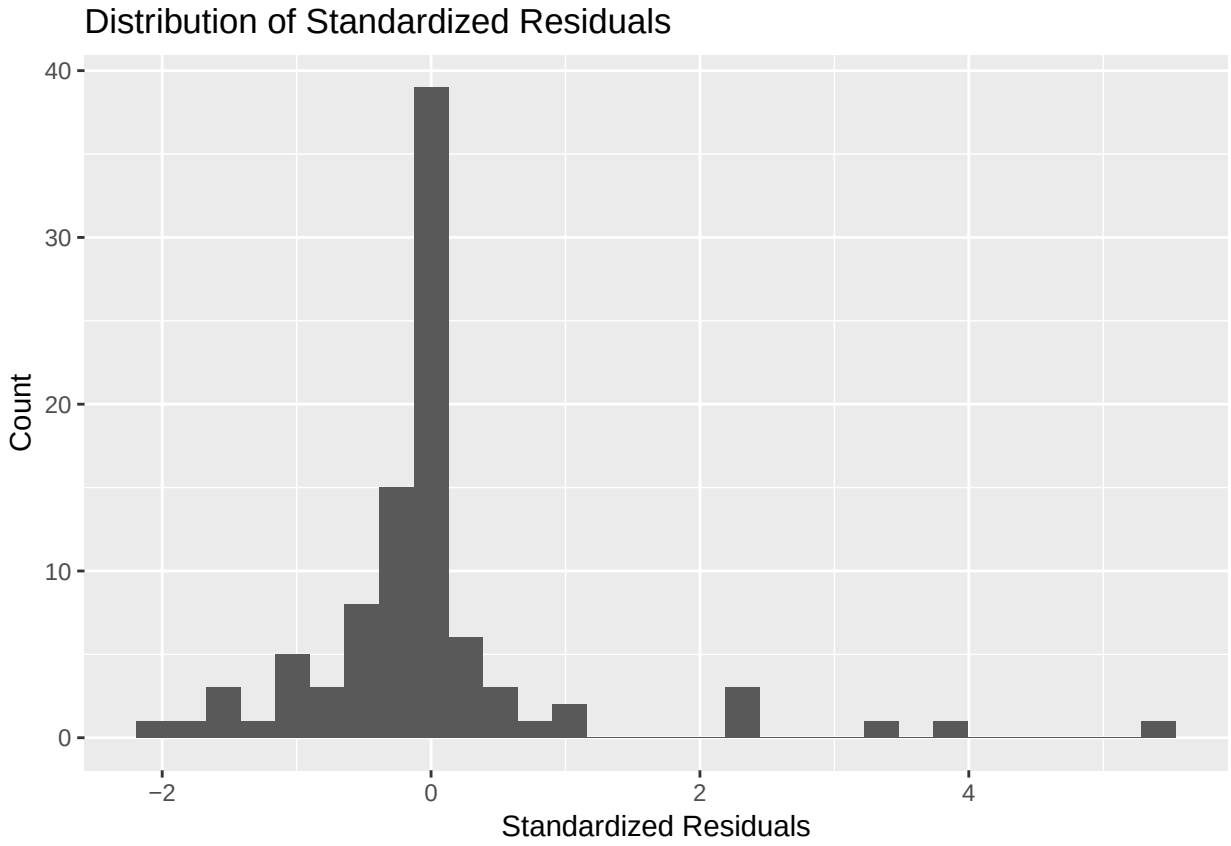
There looks to be no distinguishable pattern in the plot as well, so therefore linearity is supported because there have been previous violations of linearity.

Next, we will check for the normality assumption by creating a histogram of the residuals and a normal QQ-plot of the residuals.

```
nba_social_power_mod2 <- nba_social_power_mod2 %>%
  mutate(predicted = predict.lm(m_final_noLJ), std_resid = rstandard(m_final_noLJ))
```

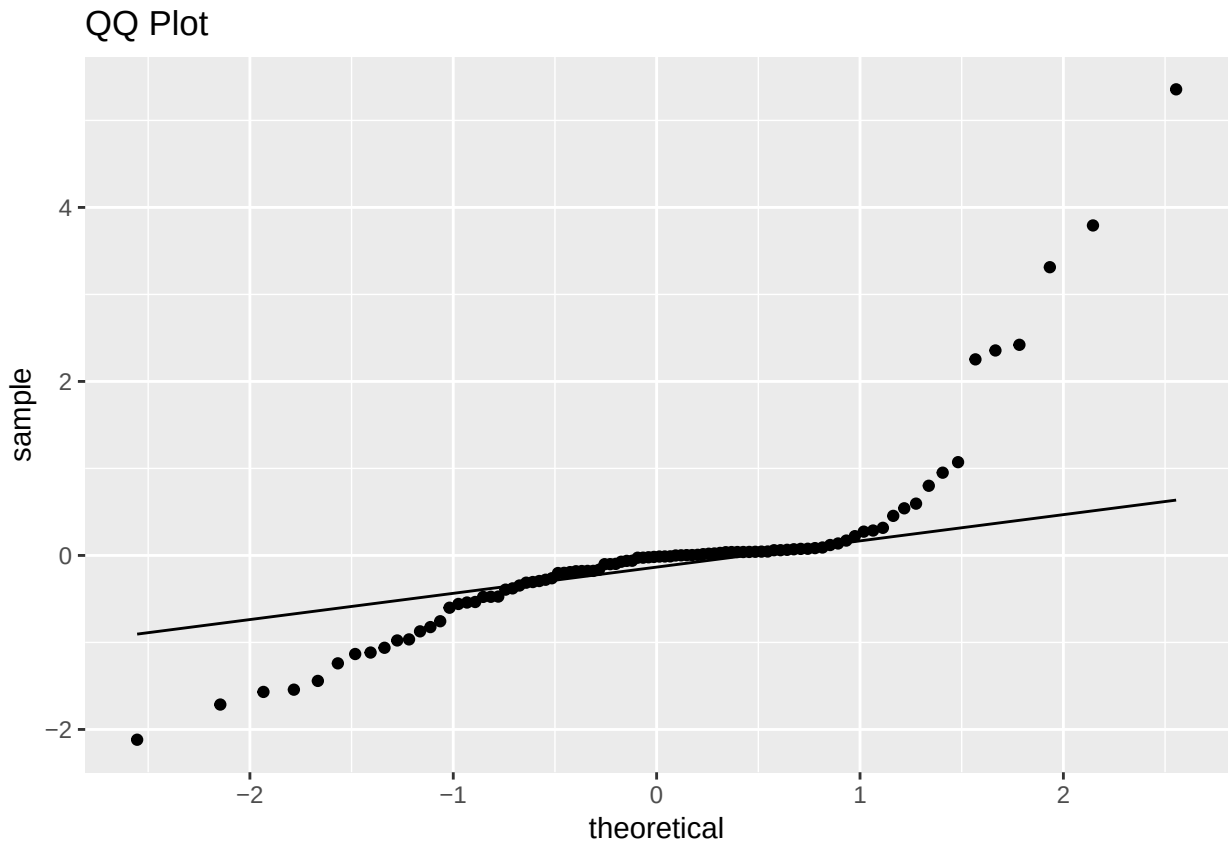
```
ggplot(data = nba_social_power_mod2, mapping = aes(x = std_resid)) + geom_histogram() + labs(title = "Di
                                                                                             x = "Standardi
                                                                                             y = "Count")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Normality is supported because the histogram of residuals is normal because the distribution is unimodal and the distribution is symmetric.

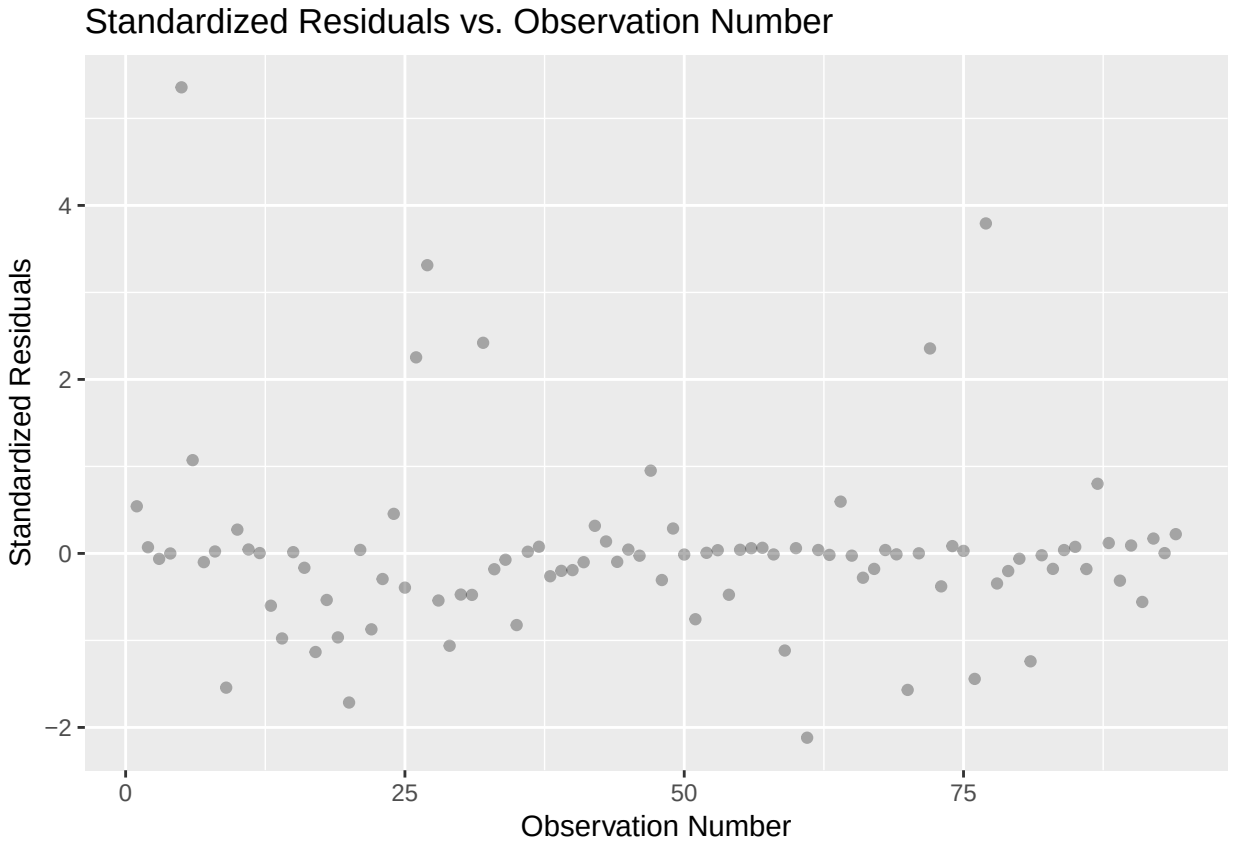
```
ggplot(data = nba_social_power_mod2, mapping = aes(sample = std_resid)) + stat_qq() + stat_qq_line() + l
```



However, normality is not satisfied because some of the points don't follow the diagonal line.

Lastly, we will check for independence. Data was not taken over time, so we know there is no temporal correlation. There is also no spatial correlation because data was not taken in space. We can check to see if there is some structure/order to the dataset according to observation number.

```
m_final_adj_aug <- m_final_adj_aug %>%
  mutate(obs_num = 1:nrow(m_final_adj_aug))

ggplot(data = m_final_aug, aes(x = obs_num, y = .std.resid)) +
  geom_point(alpha = 0.3) +
  labs(x = "Observation Number", y = "Standardized Residuals",
       title = "Standardized Residuals vs. Observation Number")
```



Looking at the graph, there is no distinguishable pattern in the graph. The number of Twitter followers of one player will not affect the number of Twitter followers of another player. Therefore, independence is satisfied.

Therefore, linearity, constant variance, and independence are satsified from our new model, but normality still has deviations. However, regression accepts robust departures from normality, so we will proceed with our new model for interpretations. This is displayed here one more time for clarification:

```
tidy(m_final_adjust, conf.int = TRUE) %>%
  kable(format = "markdown", digits = 3)
```

term	estimate	std.error	statistic	p.value	conf.low	conf.high
(Intercept)	-1.201	0.245	-4.894	0.000	-1.688	-0.713
salary_millionsCent	0.154	0.019	8.159	0.000	0.117	0.192
I(salary_millionsCent * salary_millionsCent)	-0.003	0.003	-1.280	0.204	-0.008	0.002
w_pctCent	2.299	1.022	2.250	0.027	0.268	4.330
I(w_pctCent * w_pctCent)	5.775	4.071	1.419	0.160	-2.314	13.864
salary_millionsCent:w_pctCent	-0.111	0.133	-0.828	0.410	-0.376	0.155

The equation of the final model is: `log(TWITTER_FOLLOWER_COUNT_MILLIONS)`-hat = -1.201 + 0.154 * `salary_millionsCent` + -0.003 * `salary_millionsCent * salary_millionsCent` + 2.299 * `w_pctCent` + 5.775 * `w_pctCent * w_pctCent` + -0.111 * `salary_millionsCent * w_pct_Cent`.

Lastly, we will also check for multicollinearity in our model. If two or more predictor variables are highly correlated in our model, our regression may change eratically in response to small changes in our data.

We will check the variance inflation factor (VIF) for every predictor variable to check for concerns with multicollinearity:

```
tidy(vif(m_final_noLJ))
```

```
## Warning: 'tidy.numeric' is deprecated.
## See help("Deprecated")
```

```
## # A tibble: 3 x 2
##   names                            x
##   <chr>                        <dbl>
## 1 salary_millionsCent           1.13
## 2 w_pctCent                     1.31
## 3 salary_millionsCent:w_pctCent 1.22
```

None of the variables have VIFs greater than or equal to 10, so there are no issues with multicollinearity.

Interpretations

Additional Analysis