STA 141a Final Project

Brett Loy

2023-08-10

Main Question: What are the chances of that a player will surpass Stephen Curry's record of making 402 3 pointers in a single season, 82 games?

Prediction Q Can we predict a players three point makes using their past data?

This report will explore the variables that are not as easily predicted and will focus on the shooting aspects of basketball. There are many factors/variables that involve this record including: games/minutes played, play style changes as the seasons increase, and the corner 3 point line being almost two feet shorter, 22' vs 23' and 9", than the rest of the 3 point line. Also, there are always factors that you can't predict such as: injuries, trades, offensive, player age, and defensive adjustments, bench players coming in and out of rotation, and the potential addition of a 4 point line.

The record Curry holds is 402 3 point makes in a season on 886 attempts shooting 45%. Therefore, one must play every game and average at least 4.9(basically 5) 3 point makes per game. And if the average 3 point make percentage is 35%, which is considered good, they would need to shoot at least 14 3s per game.

It is said that Curry's record cannot be broken, but do we know for sure?

The data that was collected was downloaded through a site called Kaggle. Kaggle is a data science and artificial intelligence platform where people are able to upload data sets they are interested in. My sources are from posts on Kaggle who got their information from *Basketball Reference*.

```
library(readr)

#read data in
nba_data <- #1999-2020
read_csv("/Users/brettloy/Desktop/STA 141A/Final Project Files/players_stats_by_season_full_details.csv
```

```
## Rows: 53949 Columns: 34
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (12): League, Season, Stage, Player, Team, birth_month, birth_date, heig...
## dbl (22): GP, MIN, FGM, FGA, 3PM, 3PA, FTM, FTA, TOV, PF, ORB, DRB, REB, AST...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
nba_shooting <- #1997-2023
  read_csv("/Users/brettloy/Desktop/STA 141A/Final Project Files/Player Shooting.csv")
```

```
## Rows: 15865 Columns: 34
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (4): player, pos, lg, tm
## dbl (30): seas_id, season, player_id, birth_year, age, experience, g, mp, fg...
##
```

```
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

The original data frame, nba_data, contains 54,000 rows and 34 columns. The rows are referring to Player's names while the columns are referring to statistics about that player such as draft round, season, or Team. The data frame, nba_shooting, also contains 34 columns but instead contains 16,000 rows. This data was in tidy format, but it was not cleaned up. Therefore in the data wrangling/preparation step, I converted the 2 data sets into 1 that was prepped to my desire. For example, The data from the data frame 'nba_shooting' read in is only from 1997-2023 and the data from 'nba_data' is from 1999-2020. I removed the extra years from the 'nba_shooting' frame using the filter() function. Also, I got rid of all columns that were not needed in both data frames by using the select() function and renamed columns using the rename() function. In addition I needed to switch the Season format from 'nba_data' because it was written in a xxxx-xxxx format when I needed it to be in the same format as 'nba_shooting' in order to merge the tables. To do this I used the extract() function and a regular expression to only keep the last 4 digits of the season. Lastly, all that was needed to do was combine the 2 tables by season and player name.

```r
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(stringr)

#wrangle the data to end with 1 data frame to work with
years_filter_out <- c("1997", "1998", "1999", "2021", "2022", "2023")

nba_shooting<- nba_shooting %>%
  #choose which columns from the data set needed to keep
  select("season", "player", "fg_percent_from_x3p_range", "percent_assisted_x3p_fg",
         "corner_3_point_percent", "percent_corner_3s_of_3pa")%>%
  #filter out seasons that aren't needed
  filter(!(season %in% years_filter_out))%>%
  #rename columns
  rename("percent_of_3PM_assisted"="percent_assisted_x3p_fg", "three_PM_percentage"=
"fg_percent_from_x3p_range", "percent_of_3PA_incorner"="percent_corner_3s_of_3pa",
"percent_of_3PM_incorner" = "corner_3_point_percent", "Season" = "season", "Player"="player")

#regular expression, subset, to keep the last 4 digits of season
nba_data<- nba_data %>%
  #choose which columns from the data set needed to keep
  select("League", "Season", "Player", "GP", "MIN", "3PM", "3PA", "PTS" ) %>%
  #filter out
  filter(League=="NBA")%>%
  rename("three_point_makes" = "3PM", "three_point_attempts" = "3PA")
nba_data$Season <- str_extract(nba_data$Season, "\\d{4}$")
#change "Season" column to character in order to join
nba_shooting$Season<- as.character(nba_shooting$Season)
#merge data frames sorting by "Season" and "Player"
join_df <- left_join(nba_data, nba_shooting, by = c("Season", "Player"))
```

```
## Warning in left_join(nba_data, nba_shooting, by = c("Season", "Player")): Detected an unexpected many
## i Row 46 of `x` matches multiple rows in `y`.
## i Row 12160 of `y` matches multiple rows in `x`.
## i If a many-to-many relationship is expected, set `relationship =
##   "many-to-many"` to silence this warning.
```

```
join_df
```

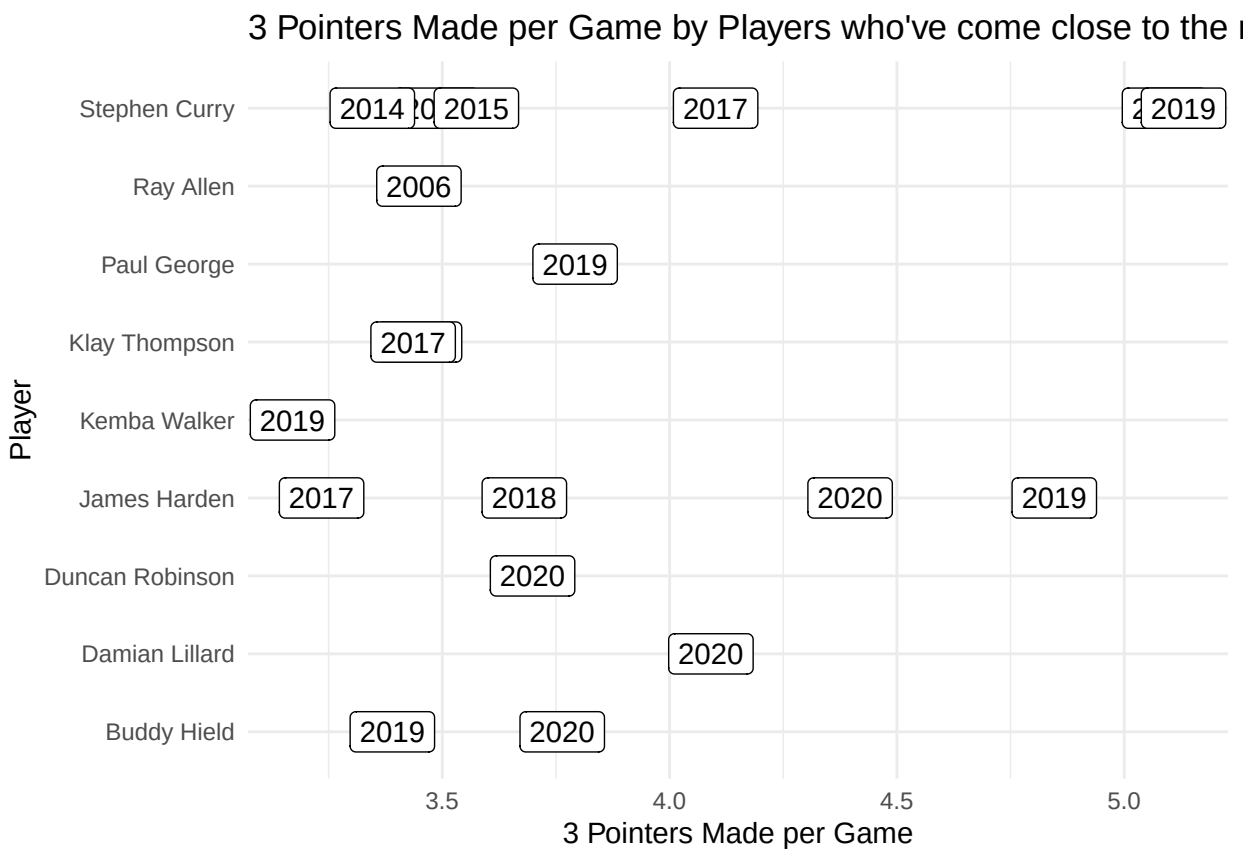
```
## # A tibble: 8,785 x 12
##    League Season Player    GP   MIN three_point_makes three_point_attempts   PTS
##    <chr>  <chr>  <chr>  <dbl> <dbl>             <dbl>                 <dbl> <dbl>
##  1 NBA    2000   Shaqu~    79  3163                 0                     1  2344
##  2 NBA    2000   Vince~    82  3126                95                   236  2107
##  3 NBA    2000   Karl ~    82  2947                 2                     8  2095
##  4 NBA    2000   Allen~    70  2853                89                   261  1989
##  5 NBA    2000   Gary ~    82  3425               177                   520  1982
##  6 NBA    2000   Jerry~    82  3148                83                   288  1939
##  7 NBA    2000   Grant~    74  2776                34                    98  1906
##  8 NBA    2000   Kevin~    81  3243                30                    81  1857
##  9 NBA    2000   Micha~    82  3464                99                   247  1855
## 10 NBA    2000   Chris~    75  2880                27                    95  1834
## # i 8,775 more rows
## # i 4 more variables: three_PM_percentage <dbl>, percent_of_3PM_assisted <dbl>,
## #   percent_of_3PM_incorner <dbl>, percent_of_3PA_incorner <dbl>
```

At this point, I am adding 2 new columns to 'join_df' to fill in some missing data of the average amount of makes and attempts of 3 point shots per game. This will help determine how many shots attempted and made per game someone would need take in order to beat the record.

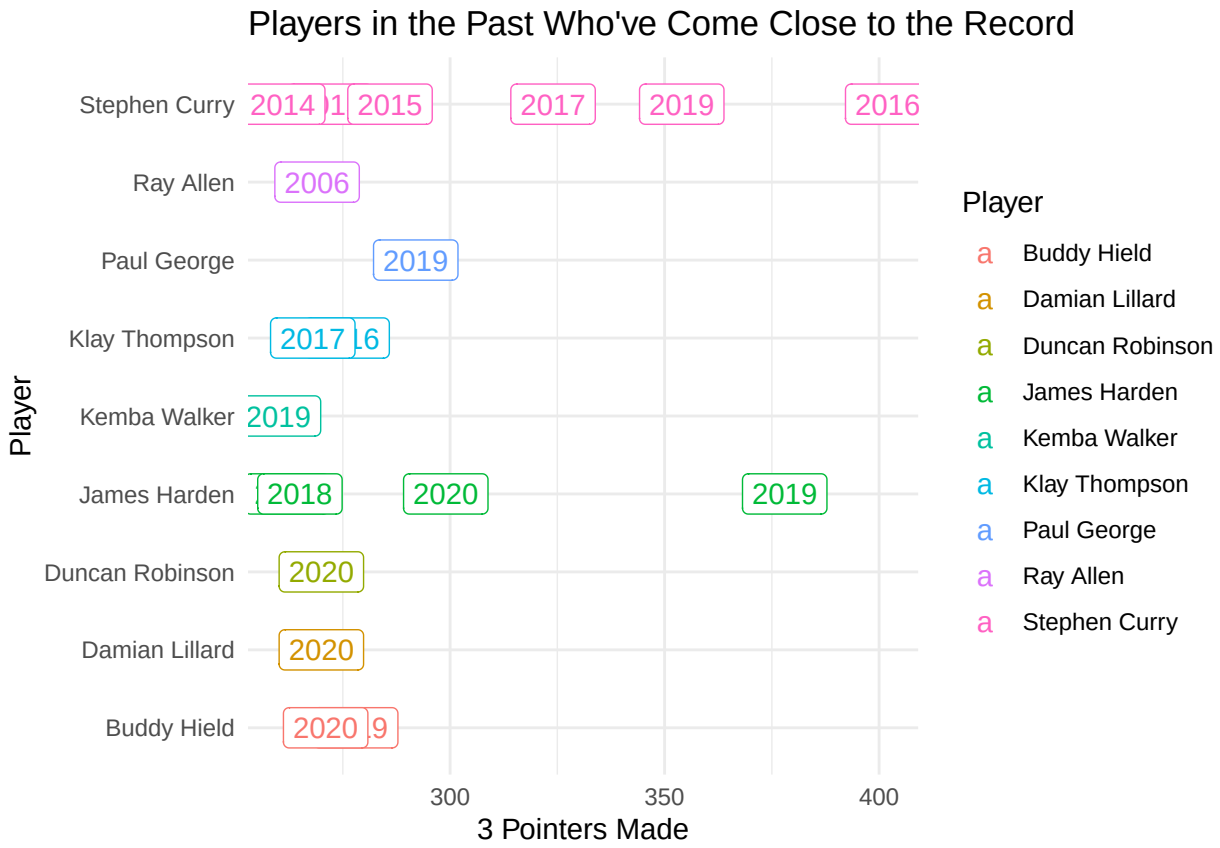
```
library(ggplot2)
library(ggthemes)
library(ggrepel)
library(ggsci)
library(caret)
```

```
## Loading required package: lattice
```

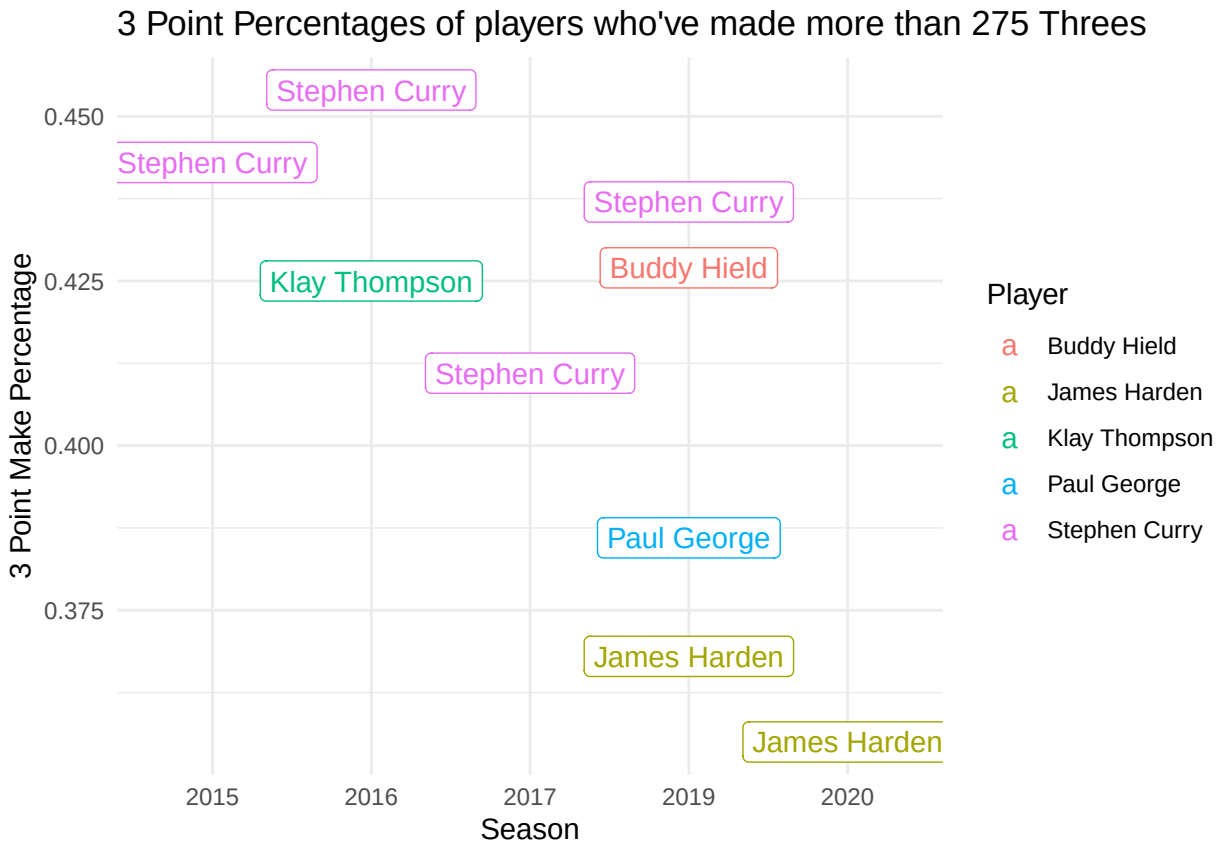
```
#3 point makes per game of players
join_df<- join_df %>% mutate(threes_mper_game = three_point_makes/GP) %>%
  mutate(threes_aper_game = three_point_attempts/GP) %>%
  filter(Season != 2012)

join_df %>% filter(three_point_makes>=250) %>%
  ggplot(aes(threes_mper_game, Player, label = Season))+
  geom_label()+
  ggtitle("3 Pointers Made per Game by Players who've come close to the record")+
  xlab("3 Pointers Made per Game")+
  theme_minimal()
```



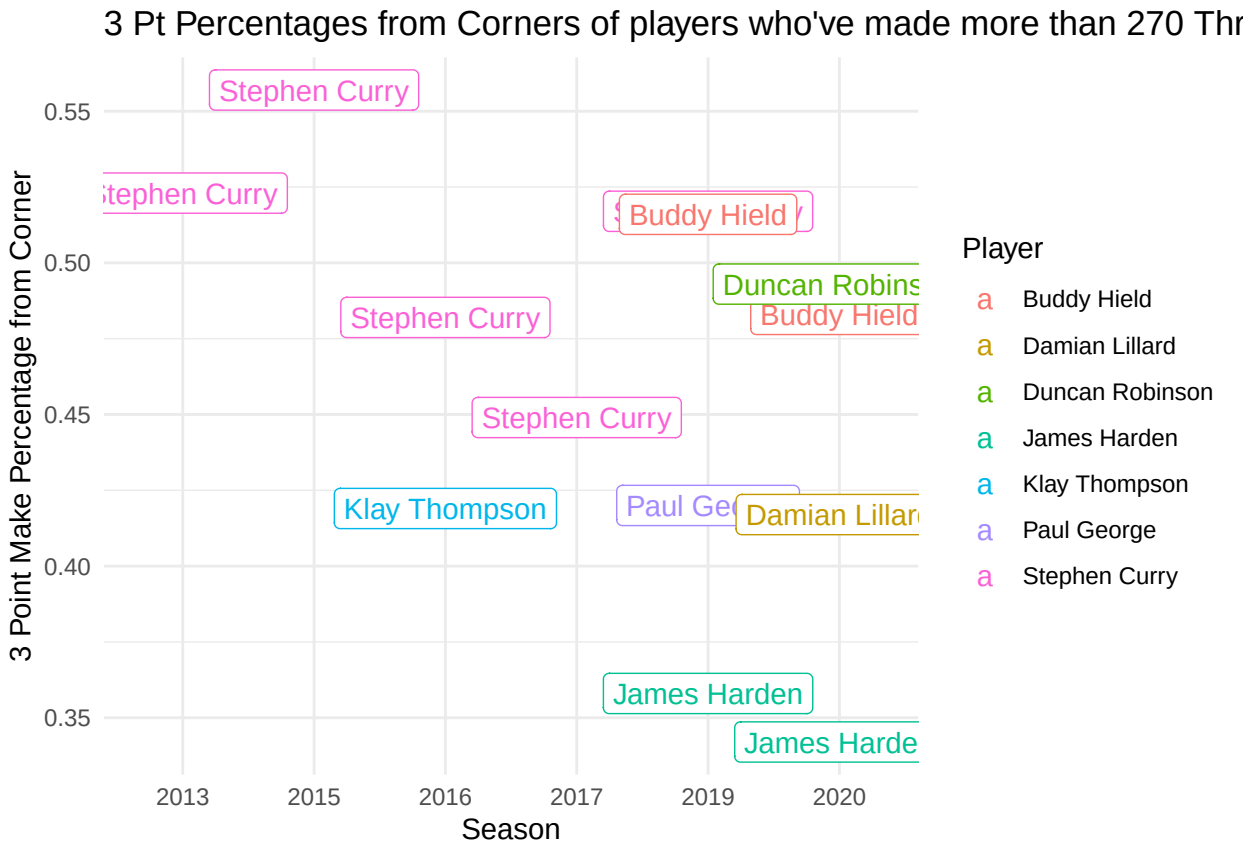

```
#players who have come close

plot_close<- join_df %>% filter(three_point_makes>=250) %>%
  ggplot(aes(three_point_makes, Player, label = Season, color = Player))+
  geom_label()+
  ggtitle("Players in the Past Who've Come Close to the Record")+
  xlab("3 Pointers Made")+
  theme_minimal()
plot_close
```




```
avg_3pt_per <- join_df %>% group_by(Season)%>% filter(three_point_makes>=275)%>%
  ggplot(aes(Season, three_PM_percentage, label = Player, color = Player))+
  ggtitle("3 Point Percentages of players who've made more than 275 Threes")+
  ylab("3 Point Make Percentage")+
  geom_label()+
  theme_minimal()
avg_3pt_per
```



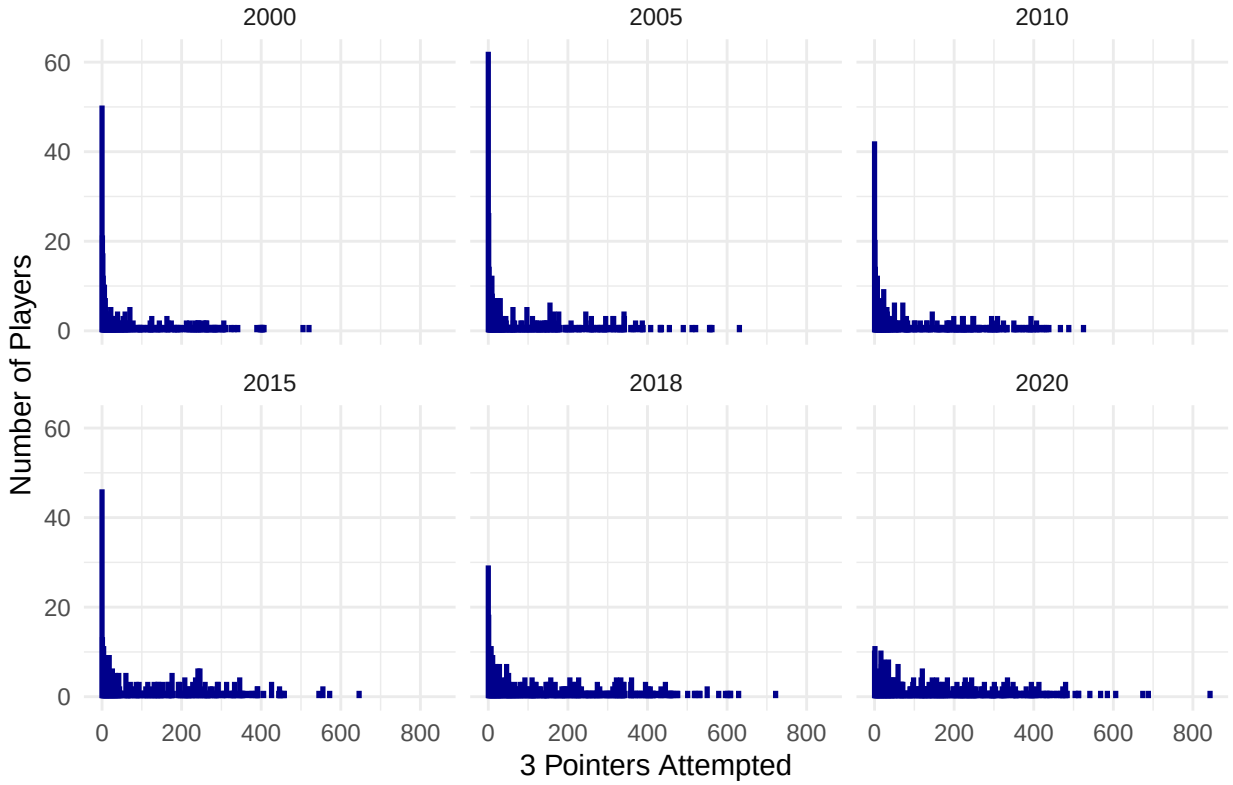

```
corner_3pt_per <- join_df %>% group_by(Season)%>% filter(three_point_makes>=270)%>%
  ggplot(aes(Season, percent_of_3PM_incorner, label = Player, color = Player))+
  ggtitle("3 Pt Percentages from Corners of players who've made more than 270 Threes")+
  ylab("3 Point Make Percentage from Corner")+
  geom_label()+
  theme_minimal()
corner_3pt_per
```



As we can see here, the play style has changed from 2000 to 2020 drastically. In the early 2000s and even through 2015, players in the NBA were not making or attempting many 3 pointers with the number of players attempting 0 threes is clearly the highest. But starting in 2018 the number of players attempting more threes is distinctly higher. In the 2020 season, the plot is even less extreme in the 0 attempted threes bar and it is almost the same height as the rest of the plot. Similarly with threes made, the plots follow the exact same pattern of not many 3's made until around 2018 and then in 2020 more people are making them.

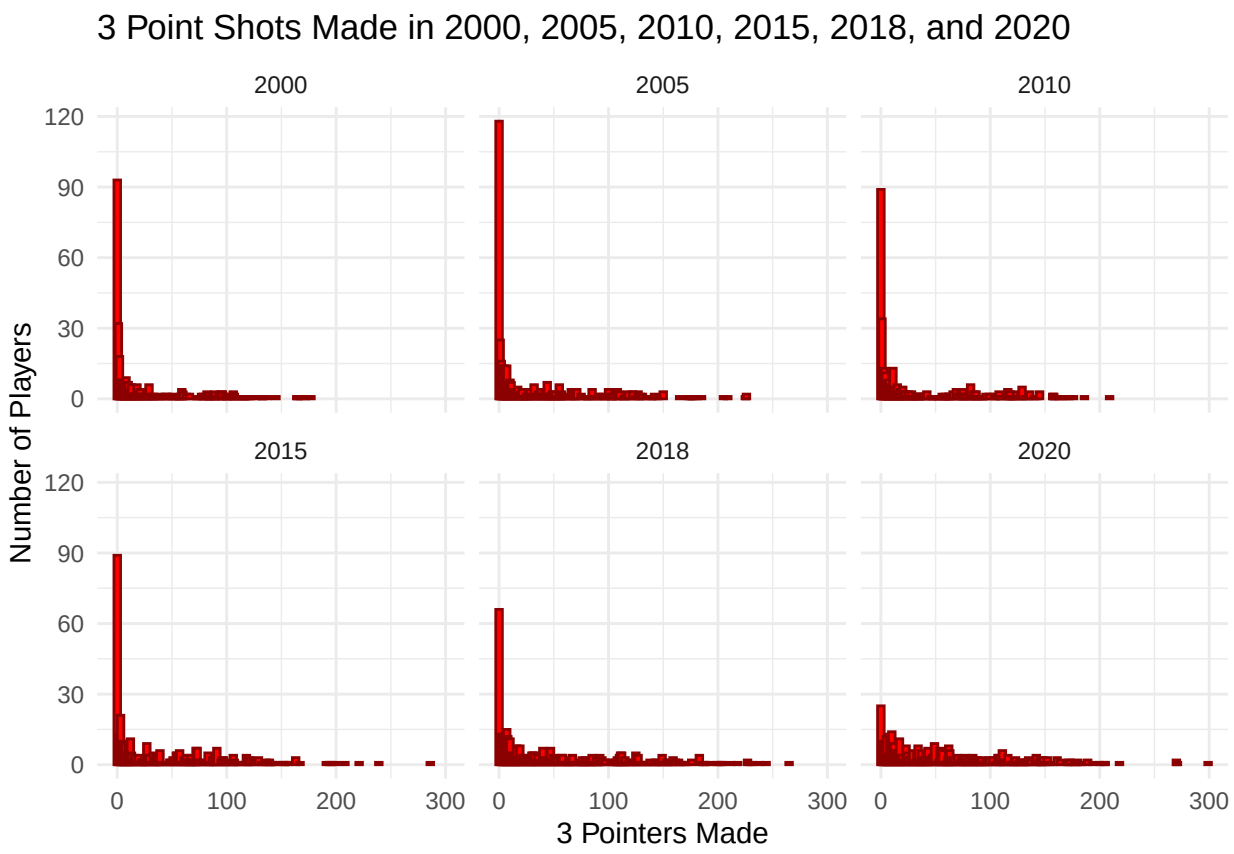
```
#data shows the increase in 3s made and attempted throughout the years
filtered_season1 <- subset(join_df, Season %in% c(2000, 2005, 2010, 2015, 2018, 2020))

three_attempt_2000_2020<- filtered_season1 %>%
  ggplot(aes(three_point_attempts))+
  geom_bar(width = 6, color = "darkblue", fill = "blue")+
  ggtitle("3 Point Shots Attempted in 2000, 2005, 2010, 2015, 2018, and 2020")+
  xlab("3 Pointers Attempted")+
  ylab("Number of Players")+
  facet_wrap(~Season)+
  theme_minimal()
three_attempt_2000_2020
```

```
## Warning: `position_stack()` requires non-overlapping x intervals
## `position_stack()` requires non-overlapping x intervals
## `position_stack()` requires non-overlapping x intervals
## `position_stack()` requires non-overlapping x intervals
## `position_stack()` requires non-overlapping x intervals
## `position_stack()` requires non-overlapping x intervals
```



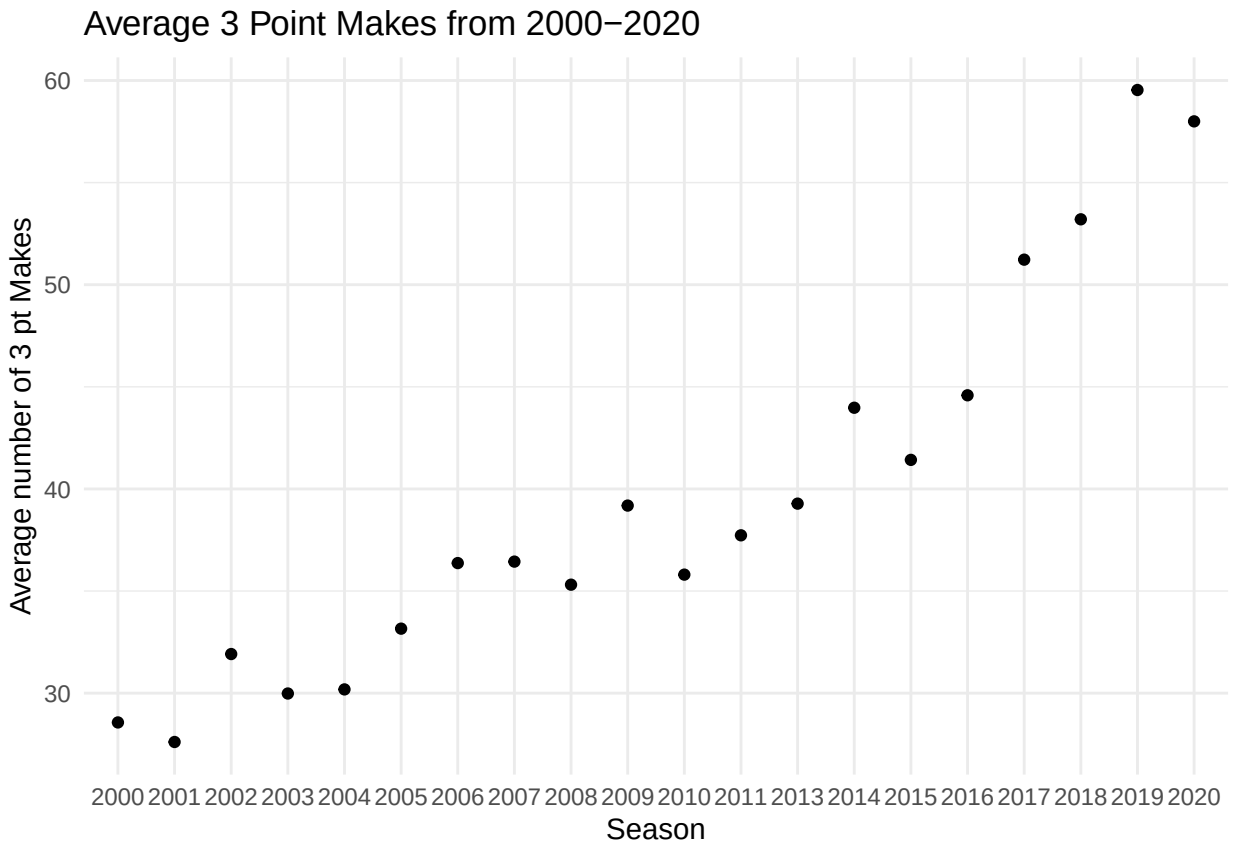

```
three_made_2000_2020<- filtered_season1 %>%
  ggplot(aes(three_point_makes))+
  geom_bar(width = 6, color = "darkred", fill = "red")+
  ggtitle("3 Point Shots Made in 2000, 2005, 2010, 2015, 2018, and 2020")+
  xlab("3 Pointers Made")+
  ylab("Number of Players")+
  facet_wrap(~Season)+
  theme_minimal()
three_made_2000_2020
```

```
## Warning: `position_stack()` requires non-overlapping x intervals
## `position_stack()` requires non-overlapping x intervals
## `position_stack()` requires non-overlapping x intervals
## `position_stack()` requires non-overlapping x intervals
## `position_stack()` requires non-overlapping x intervals
## `position_stack()` requires non-overlapping x intervals
```

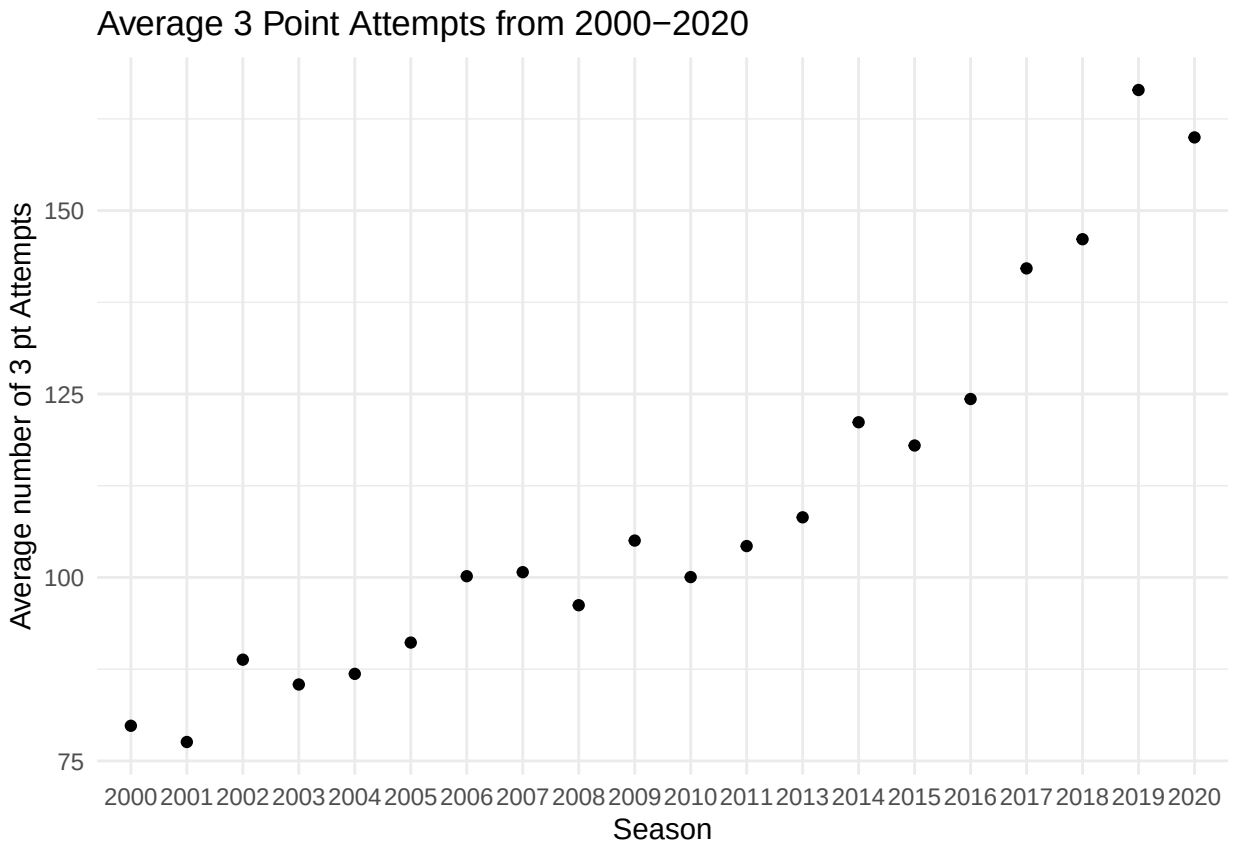


From these scatter plots, we can deduce the same summary from the graphs above, just not dealing with all players, but season averages. These scatter plots show the amount of attempts and makes of 3 pointers constantly increasing except for the dip in 2012. I'm not sure why this dip is occurring, it may have something to do with an issue in data wrangling.

```
#avg number of 3s made and attempted throughtout
avg_makes <- join_df %>%
  group_by(Season) %>%
  summarize(Avg_3PM = mean(three_point_makes)) %>%
  ggplot(aes(Season, Avg_3PM))+
  geom_point()+
  ggtitle("Average 3 Point Makes from 2000-2020")+
  ylab("Average number of 3 pt Makes")+
  theme_minimal()
avg_makes
```




```
avg_attempts <- join_df %>%
  group_by(Season) %>%
  summarize(Avg_3PA = mean(three_point_attempts)) %>%
  ggplot(aes(Season, Avg_3PA))+
  geom_point()+
  ggtitle("Average 3 Point Attempts from 2000-2020")+
  ylab("Average number of 3 pt Attempts")+
  theme_minimal()
avg_attempts
```



This linear regression function is being used to see what variables are statistically significant using p-values. Here, three point make percentage, the percent of three point makes in the corner, and number of three point attempts are the independent variables. The amount of three point makes is the dependent variable. According to the summary statistics of the multiple linear regression , these 3 independent variables are all statistically significant. Plotting each of them separately against the dependent variable shows the true relationship between the two variables. From the linear regression models, the one that stands out the most is 3 point attempts which has a direct correlation to the amount of 3 pointers made.

```
#proving overall three point percentage and corner three point percentage are
#significant variables when predicting total amount of threes made
perc_corner_attempts_significant <- lm(three_point_makes ~ three_PM_percentage+
                                       percent_of_3PM_incorner + percent_of_3PA_incorner
                                       + three_point_attempts+ percent_of_3PM_assisted,
                                       data = join_df)
summary(perc_corner_attempts_significant)
```

```
##
## Call:
## lm(formula = three_point_makes ~ three_PM_percentage + percent_of_3PM_incorner +
##     percent_of_3PA_incorner + three_point_attempts + percent_of_3PM_assisted,
##     data = join_df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -37.437  -3.447  -0.084   3.182  70.036
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
```

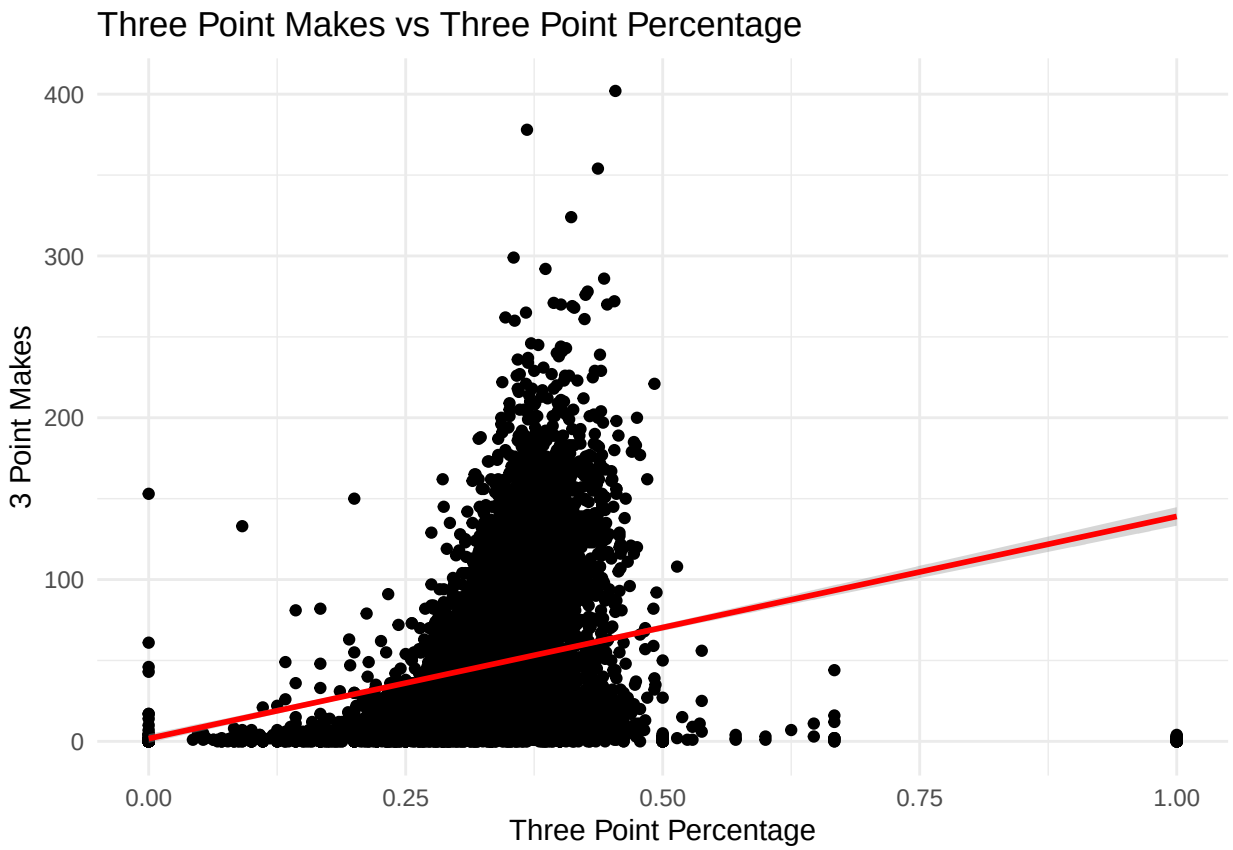
```
## (Intercept)             -1.850e+01  6.128e-01 -30.192  < 2e-16 ***
## three_PM_percentage      4.041e+01  1.394e+00  28.981  < 2e-16 ***
## percent_of_3PM_incorner  2.317e+00  7.037e-01   3.292 0.001000 **
## percent_of_3PA_incorner  7.474e-01  5.733e-01   1.304 0.192389
## three_point_attempts     3.721e-01  6.656e-04 559.002  < 2e-16 ***
## percent_of_3PM_assisted  2.312e+00  6.035e-01   3.831 0.000129 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.919 on 6218 degrees of freedom
##   (2137 observations deleted due to missingness)
## Multiple R-squared:  0.9834, Adjusted R-squared:  0.9833
## F-statistic: 7.346e+04 on 5 and 6218 DF,  p-value: < 2.2e-16
```

```
ggplot(join_df, aes(three_PM_percentage, three_point_makes))+
  geom_point()+
  stat_smooth(method = "lm", col = "red")+
  ggtitle("Three Point Makes vs Three Point Percentage")+
  ylab("3 Point Makes")+
  xlab("Three Point Percentage")+
  theme_minimal()
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

```
## Warning: Removed 1068 rows containing non-finite values (`stat_smooth()`).
```

```
## Warning: Removed 1068 rows containing missing values (`geom_point()`).
```

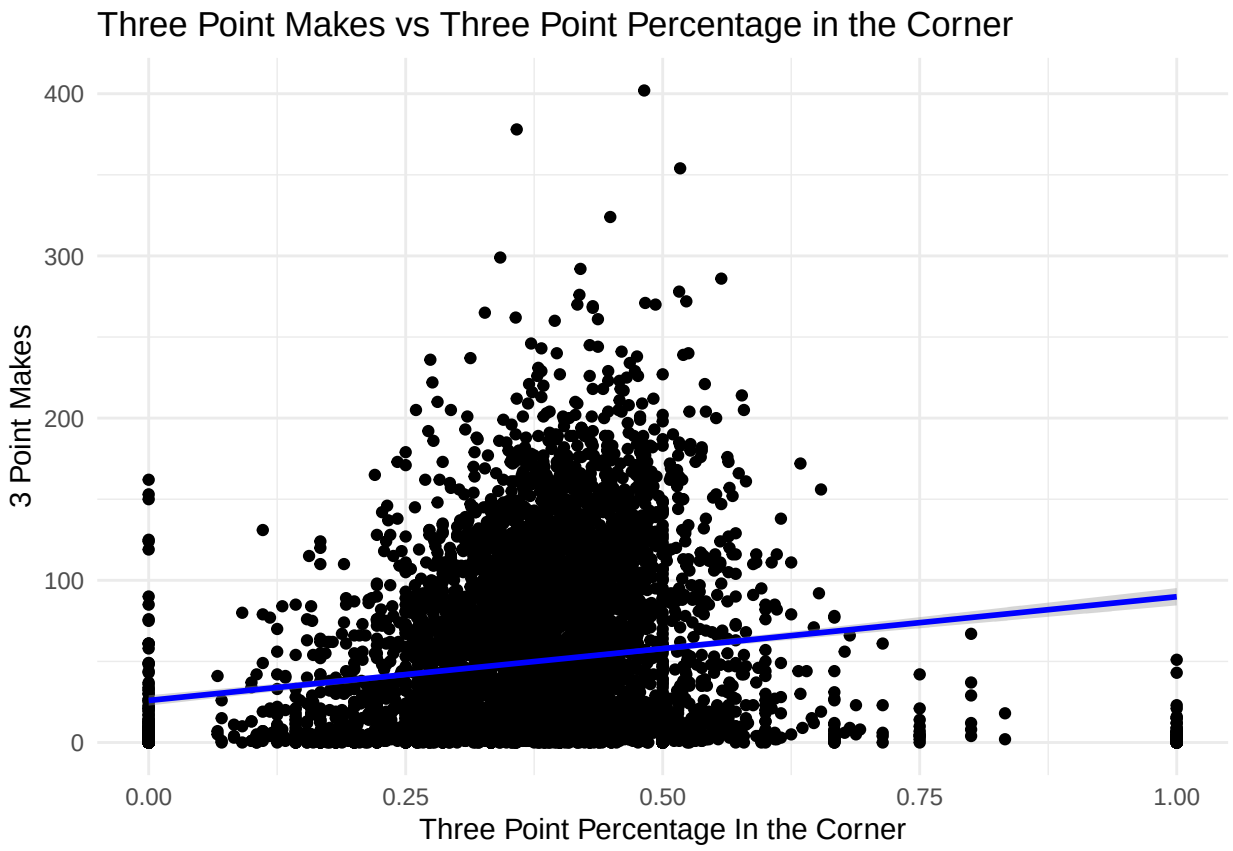


```
ggplot(join_df, aes(percent_of_3PM_incorner, three_point_makes))+
  geom_point()+
  stat_smooth(method = "lm", col = "blue")+
  ggtitle("Three Point Makes vs Three Point Percentage in the Corner")+
  ylab("3 Point Makes")+
  xlab("Three Point Percentage In the Corner")+
  theme_minimal()
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

```
## Warning: Removed 1896 rows containing non-finite values (`stat_smooth()`).
```

```
## Warning: Removed 1896 rows containing missing values (`geom_point()`).
```

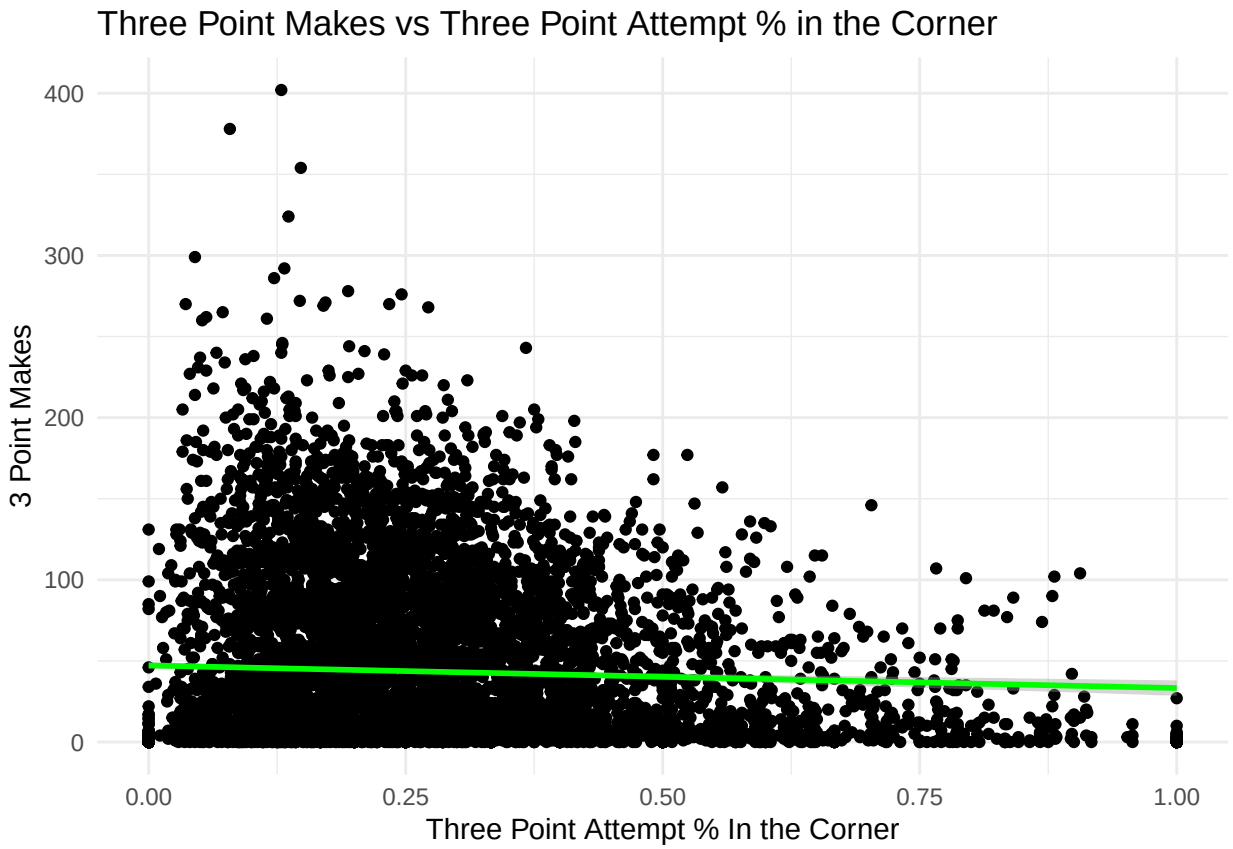



```
ggplot(join_df, aes(percent_of_3PA_incorner, three_point_makes))+
  geom_point()+
  stat_smooth(method = "lm", col = "green")+
  ggtitle("Three Point Makes vs Three Point Attempt % in the Corner")+
  ylab("3 Point Makes")+
  xlab("Three Point Attempt % In the Corner")+
  theme_minimal()
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

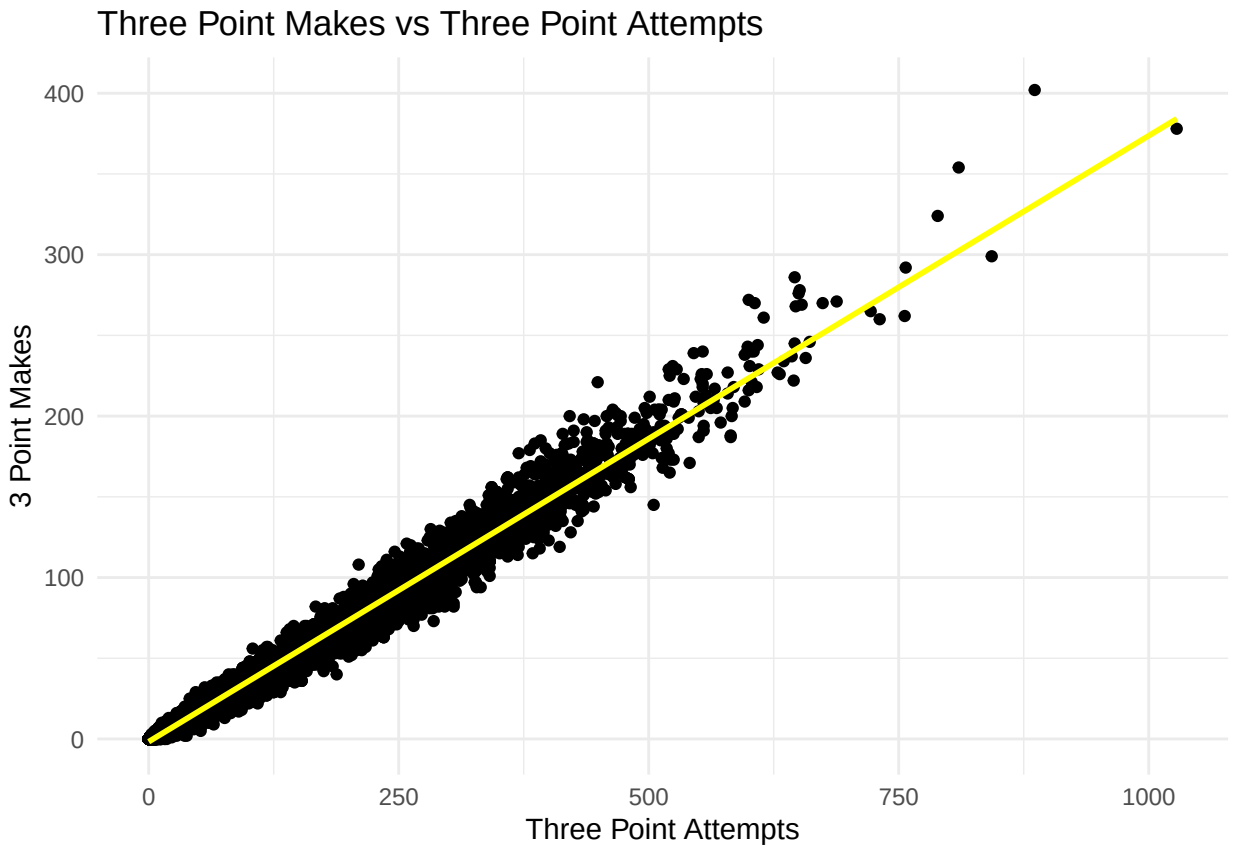
```
## Warning: Removed 1068 rows containing non-finite values (`stat_smooth()`).
```

```
## Warning: Removed 1068 rows containing missing values (`geom_point()`).
```



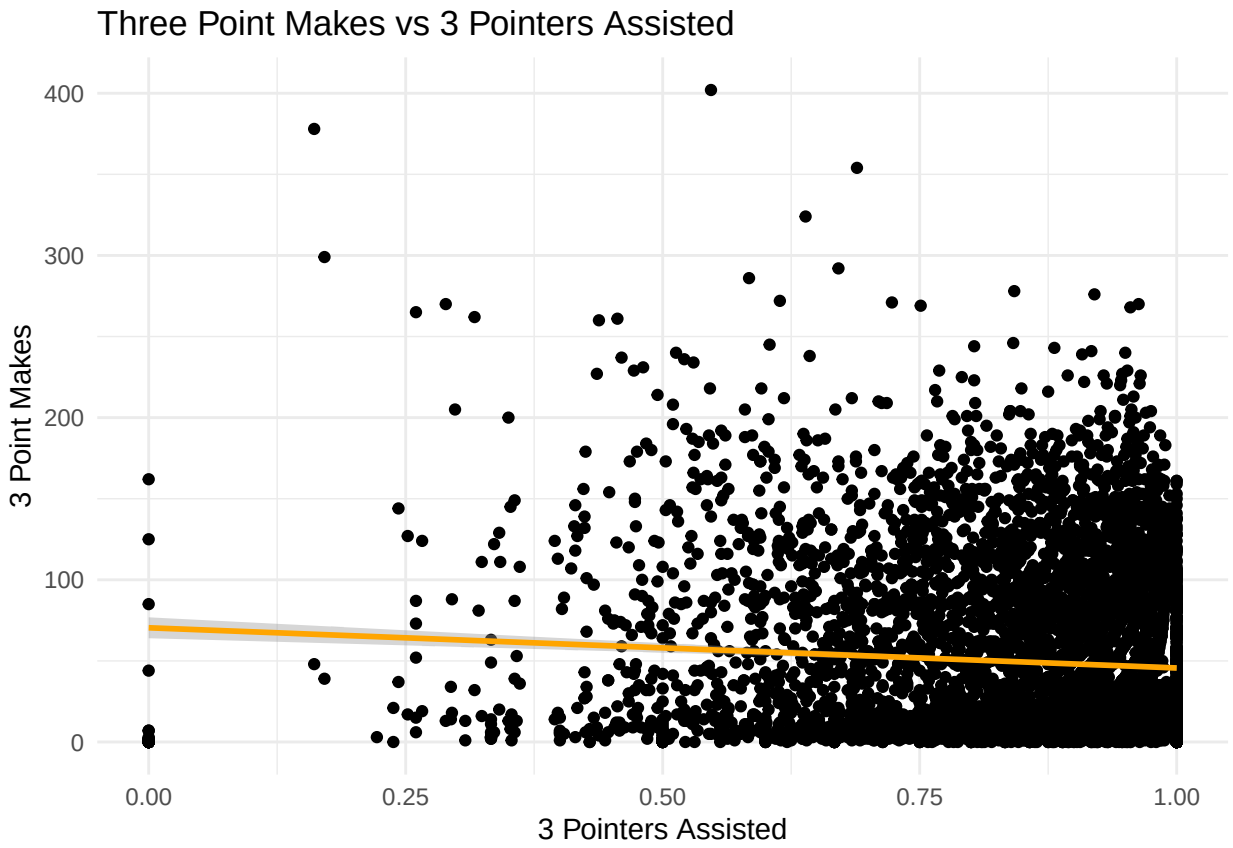

```
ggplot(join_df, aes(three_point_attempts, three_point_makes))+
  geom_point()+
  stat_smooth(method = "lm", col = "yellow")+
  ggtitle("Three Point Makes vs Three Point Attempts")+
  ylab("3 Point Makes")+
  xlab("Three Point Attempts")+
  theme_minimal()
```

```
## `geom_smooth()` using formula = 'y ~ x'
```




```
ggplot(join_df, aes(percent_of_3PM_assisted, three_point_makes))+
  geom_point()+
  stat_smooth(method = "lm", col = "orange")+
  ggtitle("Three Point Makes vs 3 Pointers Assisted")+
  ylab("3 Point Makes")+
  xlab("3 Pointers Assisted")+
  theme_minimal()
```

```
## `geom_smooth()` using formula = 'y ~ x'
## Warning: Removed 1915 rows containing non-finite values (`stat_smooth()`).
## Warning: Removed 1915 rows containing missing values (`geom_point()`).
```




```
attempts<- lm(three_point_makes ~ three_point_attempts, data = join_df)
att_sum<- summary(attempts)
mean(att_sum$residuals)
```

```
## [1] -1.455387e-15
```

Lastly the prediction part of the project. The question we strive to answer here is: can we predict a player's total 3 point makes in a season based on how many 3 pointers they have made in their past seasons? We start by making a data frame displaying the true results from the 2020 and 2021 season for a player, Stephen Curry. I also made a data frame called curry_pred, which originally was part of the join_df and it was wrangled to only contain Curry's stats that I am interested in per season. I'm not sure why, but some of the rows were duplicated so you can see we delete those rows.

```
library(forecast)
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

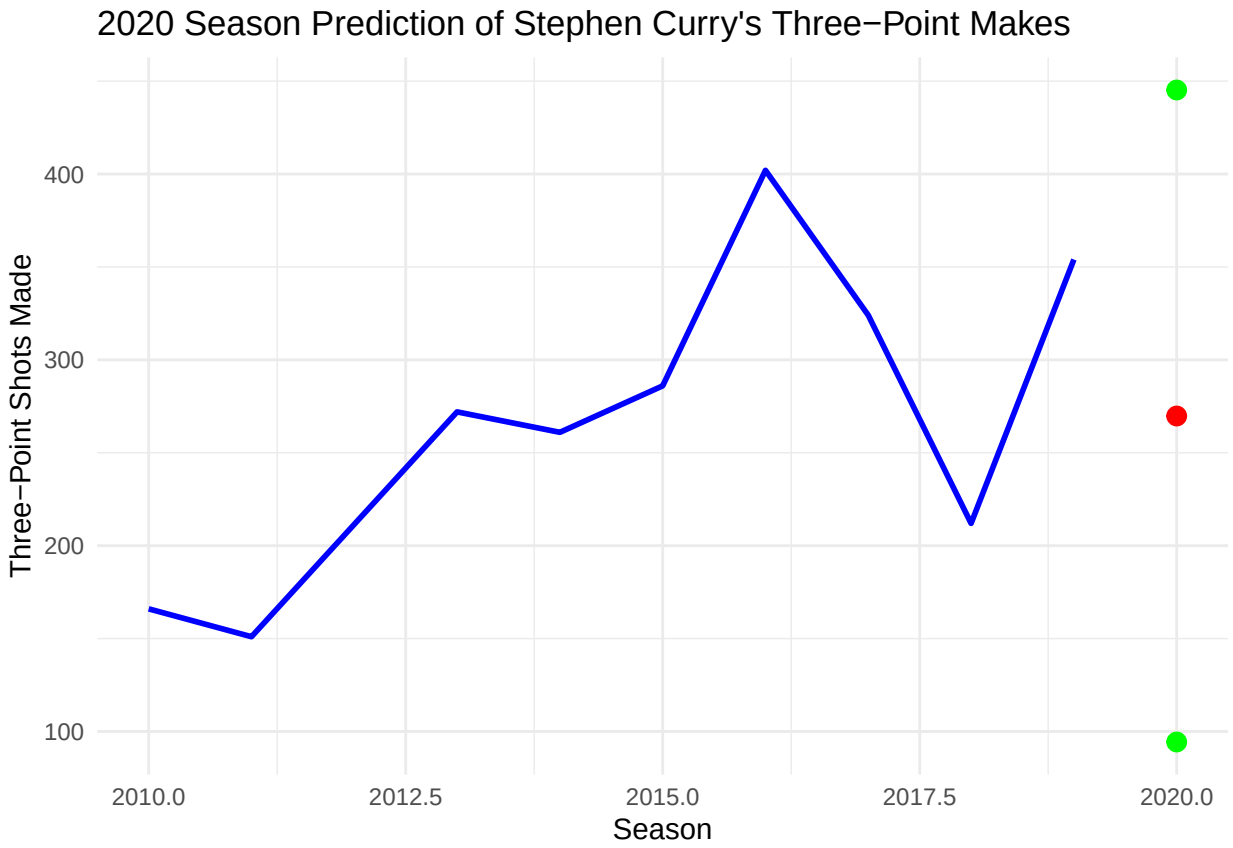
```
true_Season <- c("2020", "2021")
true_three_PM_percentage <- c(.245, .421)
true_three_point_attempts <- c(49, 801)

true_data<- data.frame(true_Season, true_three_PM_percentage, true_three_point_attempts)
true_data<- true_data %>% mutate(true_makes = true_three_PM_percentage*true_three_point_attempts)

curry_pred<- join_df %>% filter(Player == "Stephen Curry")%>%
  select("Season", "three_point_attempts", "three_PM_percentage", "three_point_makes")
curry_pred <- curry_pred[-c(4,6,8,10,12,14,16),]
```

```
curry_pred$Season <- as.double(curry_pred$Season)
```

Here in this step the training and test data are prepped for predictions later. A ration of .8 : .2 was used and this data was put into the time series. I chose to use a time series here because they work well when analyzing objects over periods of time which works for me here in seasons. Then the forecast() function is used and this gives us the prediction for the 2020 season and also 80% and 95% prediction intervals. Next I plotted the threes made vs season for curry from 2009-2019 then inputted the predicted 2020 number of made threes, with a red dot, and the prediction interval with green dots.

```
index <- createDataPartition(curry_pred$three_point_makes, p = 0.8, list = FALSE)
train_data1 <- curry_pred[index, ]
test_data1 <- curry_pred[-index, ]

curry_ts <- ts(train_data1$three_point_makes, start = 2020, frequency = 1)

predictions <- forecast(curry_ts, h = 1)

forecasted_data <- data.frame(Year = 2020, Forecast = as.numeric(predictions$mean),
Upper = predictions$upper, Lower = predictions$lower)


ggplot() +
  geom_line(data = curry_pred, aes(x = Season, y = three_point_makes), color = "blue", size = 1,
            linetype = "solid")+
  geom_point(data = forecasted_data, aes(x = Year, y = Forecast), color = "red", size = 3) +
  geom_point(data = forecasted_data, aes(x = Year, y = Upper.95.), color = "green", size = 3)+
  geom_point(data = forecasted_data, aes(x = Year, y = Lower.95. ), color = "green", size = 3)+

ggtitle("2020 Season Prediction of Stephen Curry's Three-Point Makes ")+
xlab("Season")+
ylab("Three-Point Shots Made") +
theme_minimal()
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



Finally I had calculated the RMSE by using the formula sqrt(predicted-observed^2). I calculated it twice because in the year 2020, it was the covid year so it was cut short and the data would not be able to predict this happening. So I calculated it for the year 2021. From the RMSE I find that the model is not the best predictor for 3 point shooting makes since good RMSE are around less than 1 while my calculated RMSE's are much larger.

```
calc_rmse_2020 = sqrt((269.78-12)^2)
calc_rmse_2020
```

```
## [1] 257.78
```

```
calc_rmse_2021 = sqrt((269.78-337.22)^2)
calc_rmse_2021
```

```
## [1] 67.44
```

Conclusion:

Stephen Curry's record will stay untouched for a now and it will take a huge season from a player to break it. In order to beat his record, a player would have to shoot at least 900 3 pointers(about 11 per game) and simultaneously be extremely efficient shooting at least 45%. With this many attempts and the efficiency, this player would make 405 threes and take Curry's record. But, the play style transition of the late 2010s and early 2020s are greatly influencing young basketball players to keep shooting 3's at a faster rate with each season. Finally, I wouldn't be surprised if a player comes out one season and breaks Curry's record because of the changes the game is undergoing.

It is possible to predict a players 3 point makes in a season but I wouldn't use the same model again. There are some statistically significant such as three point shot percentage, percent of 3's assisted by another player, and percent of threes taken from the corner.

Reflecting back, if I was to do this project again, I would change some things. Firstly I would gather data

that is most recent so up until the 2023 season to so it can be as updated as possible with 3 point trends. I would also change my prediction model to be more effective. Now it is hard to understand and someone looking at it would not be able to tell what the predicted 3 point makes is and where the prediction interval is.