Data Science II - Final Project

Ali Kilmen

(29.05.2023)

Introduction

First of all, I am a huge NBA fan and data science is another thing that I am in love with, so I think I can make a nice mix with these two. While watching NBA for so long I have always wondered that why are their players so successful from literally every angle when I compared them to other countries basketball players,and I expected to find real reasons and what are the things beyond of these players and team's successes that I wonder when I am watching NBA playoffs. This was my main motivation so in light of this information we can proceed.

I am going to investigate on NBA's 2020-2021 playoff run and I want to lean on Golden State Warriors especially because they won the Finals that year and they beat a phenomenal team. Their opponent was Boston Celtics a team that had a great run to the NBA finals that year they beat a lot of powerful teams along their way to the Finals.They also had a lot of powerful features as a team because they had such powerful, skillful players for instance Jayson Tatum, Jaylen Brown, Al Horford all of these players had a great run that year with their rookie coach. So the main problem is how did GSW beat Celtics in the finals and who had huge effect on these teams NBA finals run.

As I mentioned before my data set is about the 2021-2022 NBA playoff run and players who play in that playoff run and their statistics through the playoff run. I found the data set on https://www.kaggle.com which is a site that has various kind of data sets in it.For my workings I also add another variable to the data set which is called EFF (EFF = efficiency). I made some calculations based on already giving variables in the data set and then I found a formula that I can calculate efficiency of the players who played in the playoffs.With respect to that variable, now I can show players efficiency.

So basically, what I expect to achieve in this research is. How GSW and BOS made it to the finals and what were the things that they did it and made them better than other teams and put them on the top.Actually, I have one or two ideas in my mind, but I am looking forward to seeing that on a statistical basis. Also, I am wondering that as a fan of this game and as a person that who nearly watch all the games in that playoff run from start to the finals.I know the games that I have watched, and I know what I have been feeling when I watched those games, so did the statistics show me the same thing that what I have watched on the screen? That was one of the objectives for me.

Methodology

I will employ multiple linear regression. This modeling approach is suitable for my design because it allows you to analyze the relationship between multiple predictor variables (such as player statistics) and a continuous outcome variable (such as points per game).Multiple linear regression is a widely used statistical technique that aims to predict the value of a dependent variable based on the linear combination of multiple independent variables.In this case, we can use multiple linear regression to explore how different player statistics (e.g., field goals, three-pointers, assists) contribute to the prediction of points per game.The main equation for multiple linear regression can be summarized as follows:

$$Y = \ 0 + \ 1X1 + \ 2X2 + ... + \ n*Xn +$$

Where Y represents the dependent variable (points per game), 0 represents the intercept term, 1 to n represent the regression coefficients for the independent variables (player statistics), X1 to Xn represent the independent variables, and represents the error term.

The properties of multiple linear regression include:

- Assessing the significance and magnitude of the regression coefficients to understand the impact of each predictor variable on the outcome variable.

- Evaluating the overall fit of the regression model using measures like R-squared to determine how well the model explains the variability in the data.
- Checking for assumptions such as linearity, independence, normality, and homoscedasticity to ensure the validity of the model.

Explaratory Data analysis

```
library(readr)
X2021_2022_NBA_Player_Stats_Playoffs <- read_delim("C:/Users/aliki/Desktop/2021-2022 NBA Player Stats -
    delim = ";", escape_double = FALSE, trim_ws = TRUE)
```

```
## Rows: 217 Columns: 30
## -- Column specification --------------------------------------------------------
## Delimiter: ";"
## chr  (3): Player, Pos, Tm
## dbl (27): Rk, Age, G, GS, MP, FG, FGA, FG%, 3P, 3PA, 3P%, 2P, 2PA, 2P%, eFG%...
##
## 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.
```

```
View(X2021_2022_NBA_Player_Stats_Playoffs)
NBADF <- X2021_2022_NBA_Player_Stats_Playoffs
```

```
# Calculate efficiency and add as a column
Efficiency <- NBADF$PTS + NBADF$TRB + NBADF$AST + NBADF$STL + NBADF$BLK - NBADF$FG - NBADF$FT - NBADF$TU
NBADF <- cbind(NBADF, EFF = Efficiency)

# Seeing "EFF" and other col names
colnames(NBADF)
```

```
##  [1] "Rk"     "Player" "Pos"    "Age"    "Tm"     "G"      "GS"     "MP"
##  [9] "FG"     "FGA"    "FG%"    "3P"     "3PA"    "3P%"    "2P"     "2PA"
## [17] "2P%"    "eFG%"   "FT"     "FTA"    "FT%"    "ORB"    "DRB"    "TRB"
## [25] "AST"    "STL"    "BLK"    "TOV"    "PF"     "PTS"    "EFF"
```

```
head(NBADF)
```

```
##   Rk                  Player Pos Age  Tm  G GS   MP  FG FGA   FG%  3P 3PA
## 1  1         Precious Achiuwa   C  22 TOR  6  1 27.8 4.2 8.7 0.481 0.8 2.7
## 2  2             Steven Adams   C  28 MEM  7  5 16.3 1.3 3.0 0.429 0.0 0.0
## 3  3              Bam Adebayo   C  24 MIA 18 18 34.1 5.8 9.7 0.594 0.0 0.1
## 4  4 Nickeil Alexander-Walker  SG  23 UTA  1  0  5.0 2.0 2.0 1.000 0.0 0.0
## 5  5            Grayson Allen  SG  26 MIL 12  5 25.4 3.1 6.8 0.451 1.6 4.0
## 6  6            Jose Alvarado  PG  23 NOP  6  0 19.5 2.7 5.5 0.485 1.0 2.7
##     3P%  2P 2PA   2P%  eFG%  FT FTA   FT% ORB DRB TRB AST STL BLK TOV  PF  PTS
## 1 0.313 3.3 6.0 0.556 0.529 1.0 1.7 0.600 1.3 3.5 4.8 1.0 0.2 0.8 1.5 2.3 10.2
## 2 0.000 1.3 3.0 0.429 0.429 0.9 1.6 0.545 2.1 4.3 6.4 2.1 0.1 0.1 0.6 1.7  3.4
## 3 0.000 5.8 9.7 0.598 0.594 3.2 4.2 0.763 2.1 5.9 8.0 2.7 1.0 0.7 2.1 3.1 14.8
## 4 0.000 2.0 2.0 1.000 1.000 1.0 1.0 1.000 0.0 1.0 1.0 1.0 1.0 0.0 0.0 0.0  5.0
## 5 0.396 1.5 2.8 0.529 0.567 0.6 0.9 0.636 0.4 2.5 2.9 1.3 0.7 0.3 0.8 1.8  8.3
```

```
## 6 0.375 1.7 2.8 0.588 0.576 1.7 2.2 0.769 0.3 1.0 1.3 1.5 1.2 0.2 1.2 3.0  8.0
##    EFF
## 1 10.3
## 2  9.3
## 3 16.1
## 4  5.0
## 5  9.0
## 6  6.6
```

```
# Select the relevant variables for the regression model
selected_vars <- c("PTS", "TRB", "AST", "STL", "BLK")

# Create a new dataframe with the selected variables
regression_data <- NBADF %>% select(selected_vars)
```

```
## Warning: Using an external vector in selections was deprecated in tidyselect 1.1.0.
## i Please use `all_of()` or `any_of()` instead.
##   # Was:
##   data %>% select(selected_vars)
##
##   # Now:
##   data %>% select(all_of(selected_vars))
##
## See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

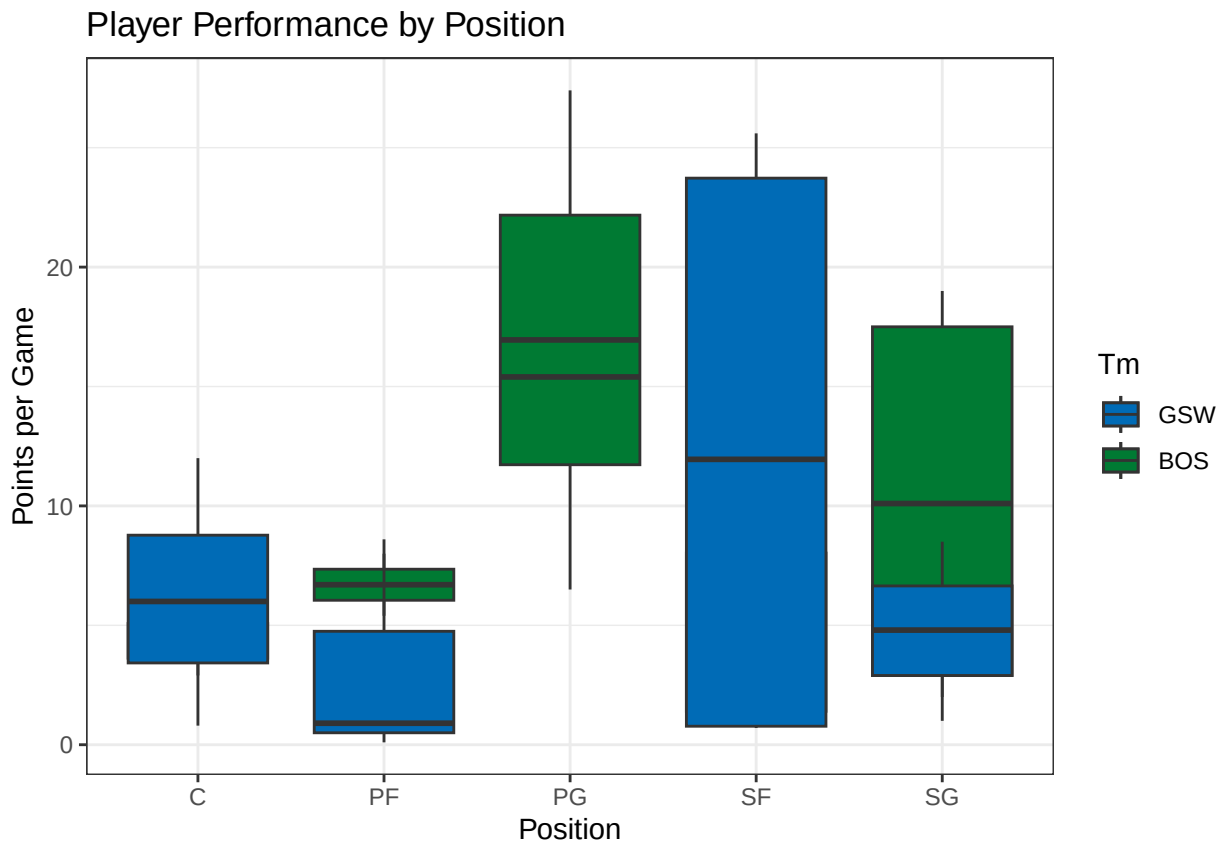
```
# Split the data into training and testing sets
set.seed(234)
train_indices <- createDataPartition(regression_data$PTS, p = 0.8, list = FALSE)
train_data <- regression_data[train_indices, ]
test_data <- regression_data[-train_indices, ]

# Create the regression model
reg_model <- lm(PTS ~ ., data = train_data)

# Print the model summary
summary(reg_model)
```

```
##
## Call:
## lm(formula = PTS ~ ., data = train_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.4887 -2.0560 -0.3408  1.6732 14.1389
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.2335     0.5040   0.463   0.6437
## TRB           1.0680     0.1536   6.952 7.39e-11 ***
## AST           1.8844     0.2213   8.515 8.51e-15 ***
```

```
## STL           1.7523     0.8491   2.064   0.0406 *
## BLK           1.1840     0.8797   1.346   0.1801
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.918 on 170 degrees of freedom
## Multiple R-squared:  0.7575, Adjusted R-squared:  0.7517
## F-statistic: 132.7 on 4 and 170 DF,  p-value: < 2.2e-16
```

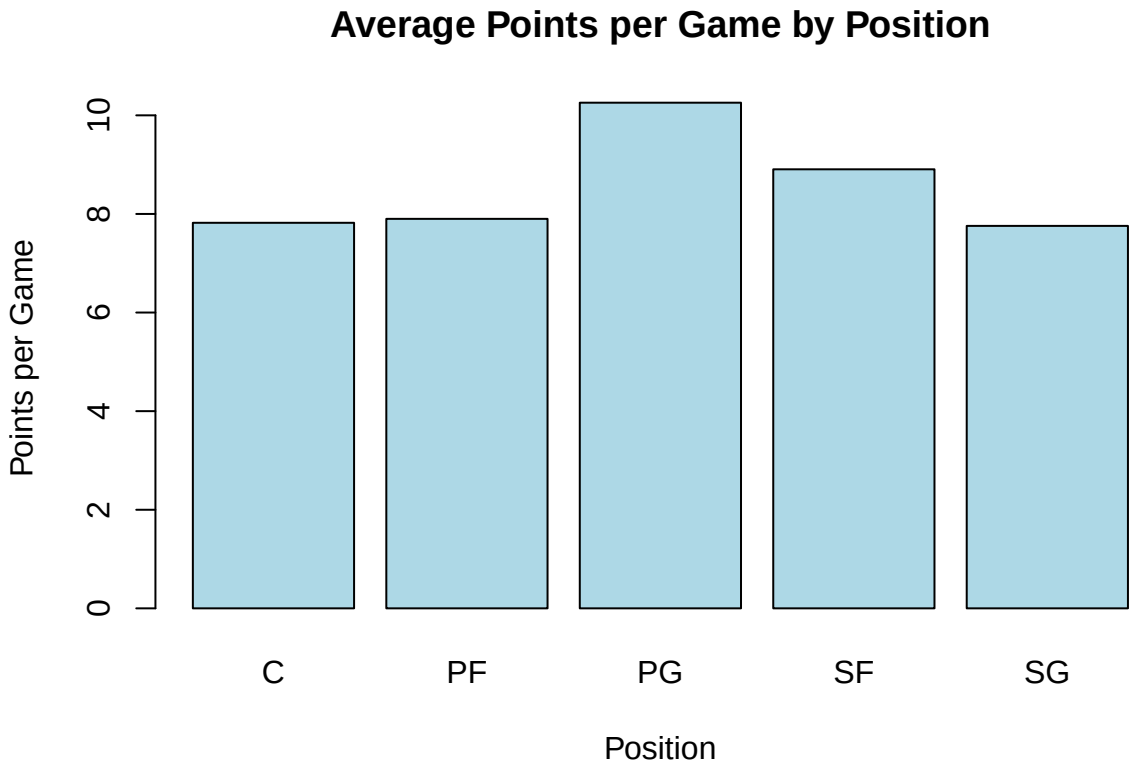
```r
# Make predictions on the test set
predictions <- predict(reg_model, newdata = test_data)

# Calculate the root mean squared error (RMSE)
rmse <- caret::RMSE(predictions, test_data$PTS)

# Print the RMSE
cat("Root Mean Squared Error (RMSE):", rmse, "\n")
```

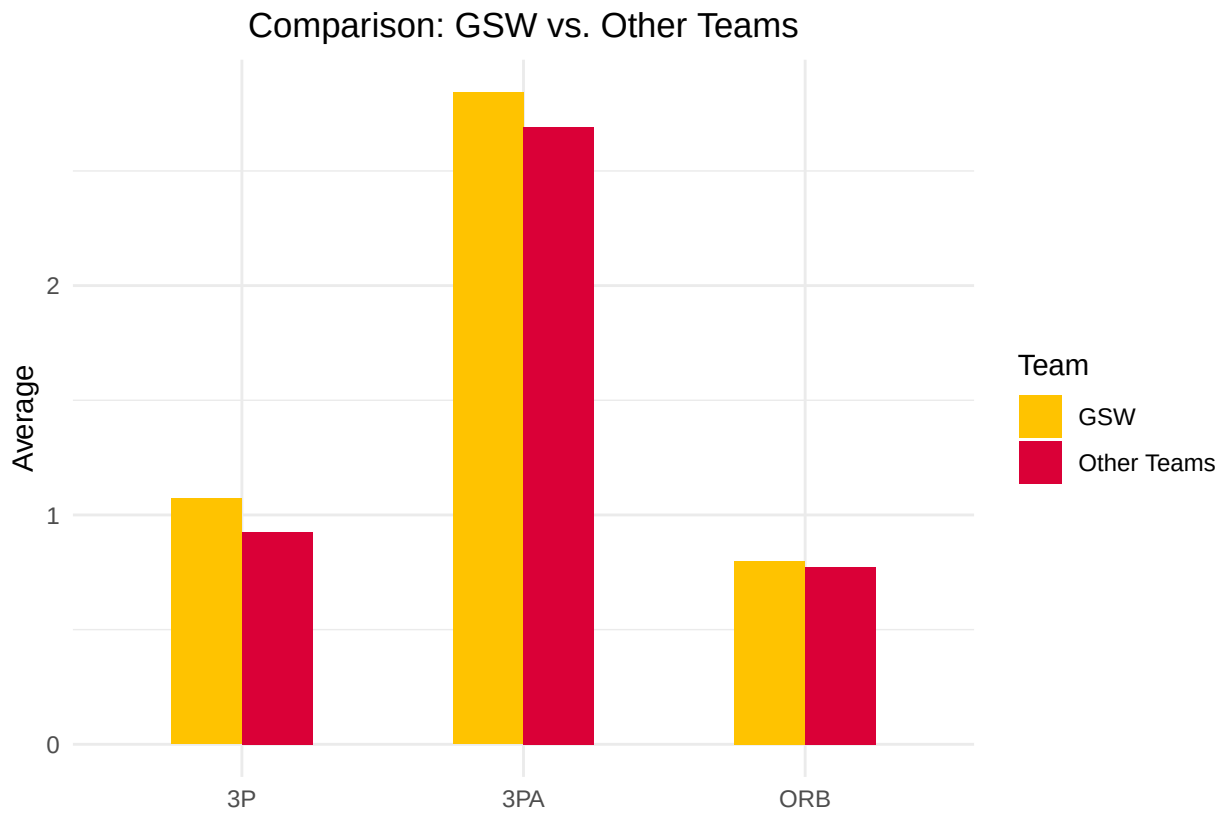
```
## Root Mean Squared Error (RMSE): 3.992204

## Average points scored by GSW: 8.664286

## Average points scored by BOS: 7.62
```



```
##              Df Sum Sq Mean Sq F value Pr(>F)
## Pos           4    190   47.46   0.813  0.518
## Residuals   212  12370   58.35
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = PTS ~ Pos, data = NBADF)
##
## $Pos
##              diff       lwr      upr     p adj
## PF-C   0.08000000 -4.378358 4.538358 0.9999986
## PG-C   2.43641026 -2.293141 7.165962 0.6171648
## SF-C   1.08512821 -3.644423 5.814680 0.9698604
## SG-C  -0.06285714 -4.541389 4.415675 0.9999995
## PG-PF  2.35641026 -2.133591 6.846411 0.5999776
## SF-PF  1.00512821 -3.484873 5.495129 0.9724436
## SG-PF -0.14285714 -4.367629 4.081915 0.9999829
## SF-PG -1.35128205 -6.110673 3.408109 0.9358349
## SG-PG -2.49926740 -7.009300 2.010766 0.5475914
## SG-SF -1.14798535 -5.658018 3.362048 0.9562279
```



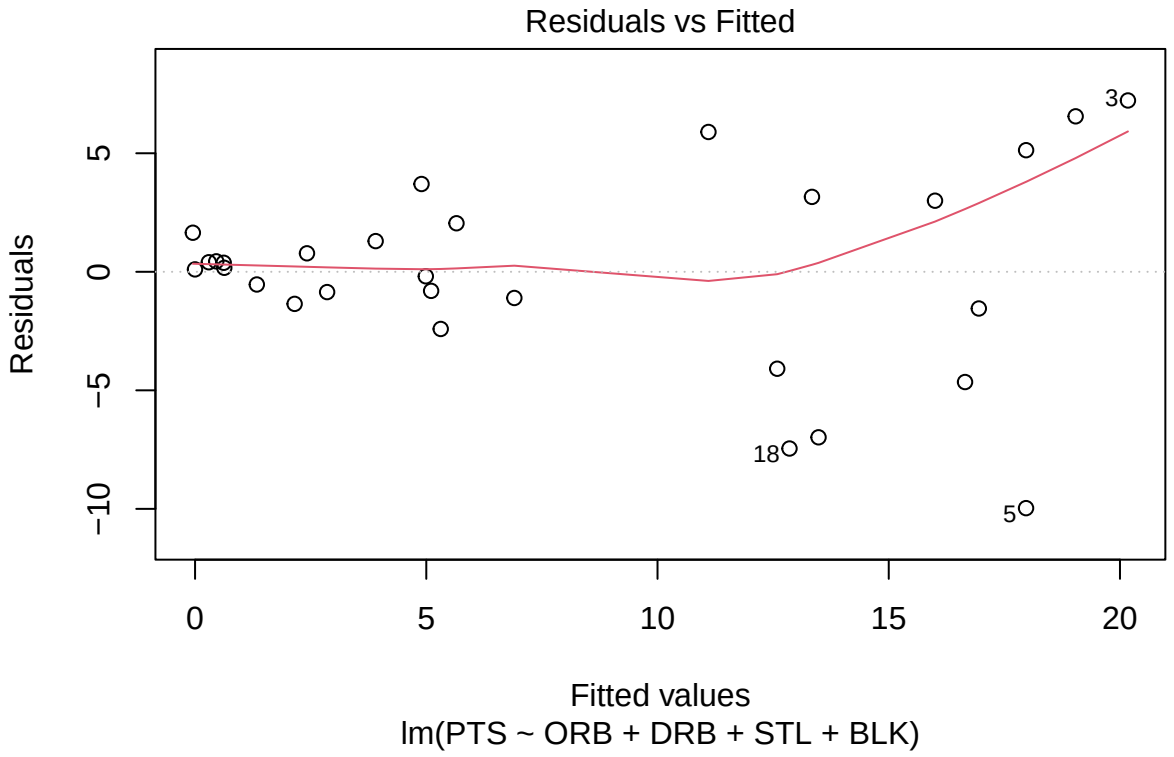


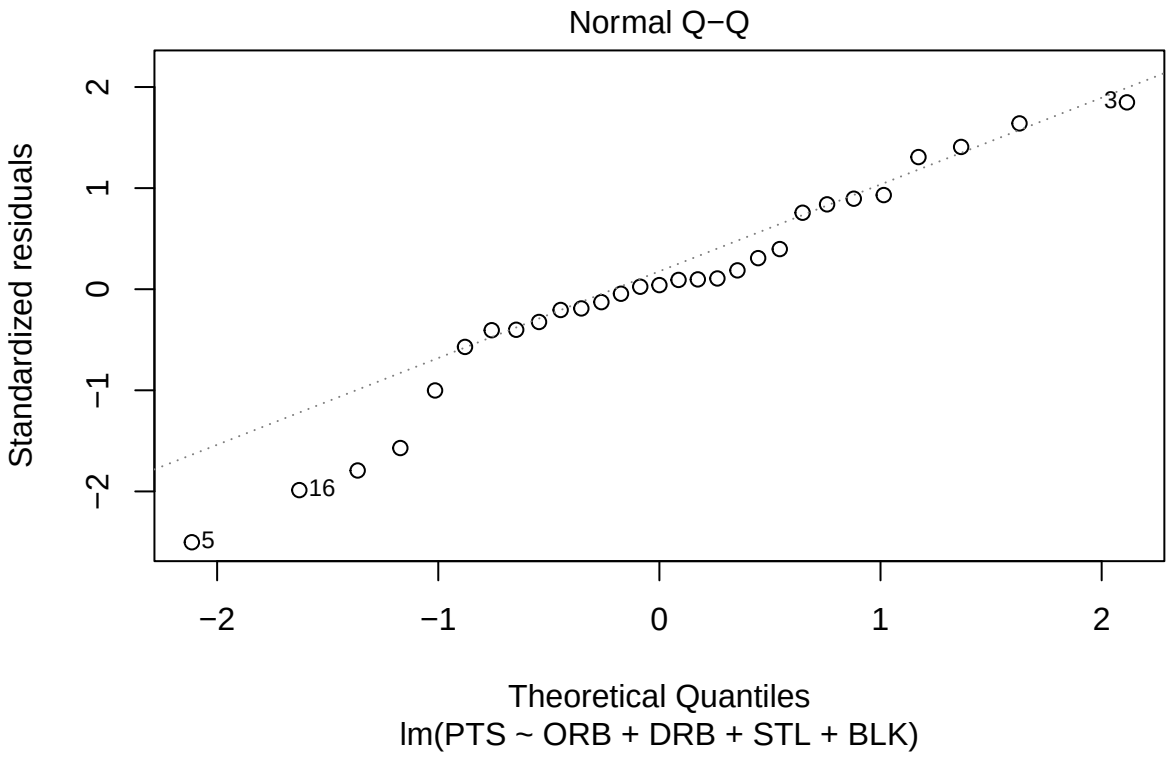
Model Fit Comparison

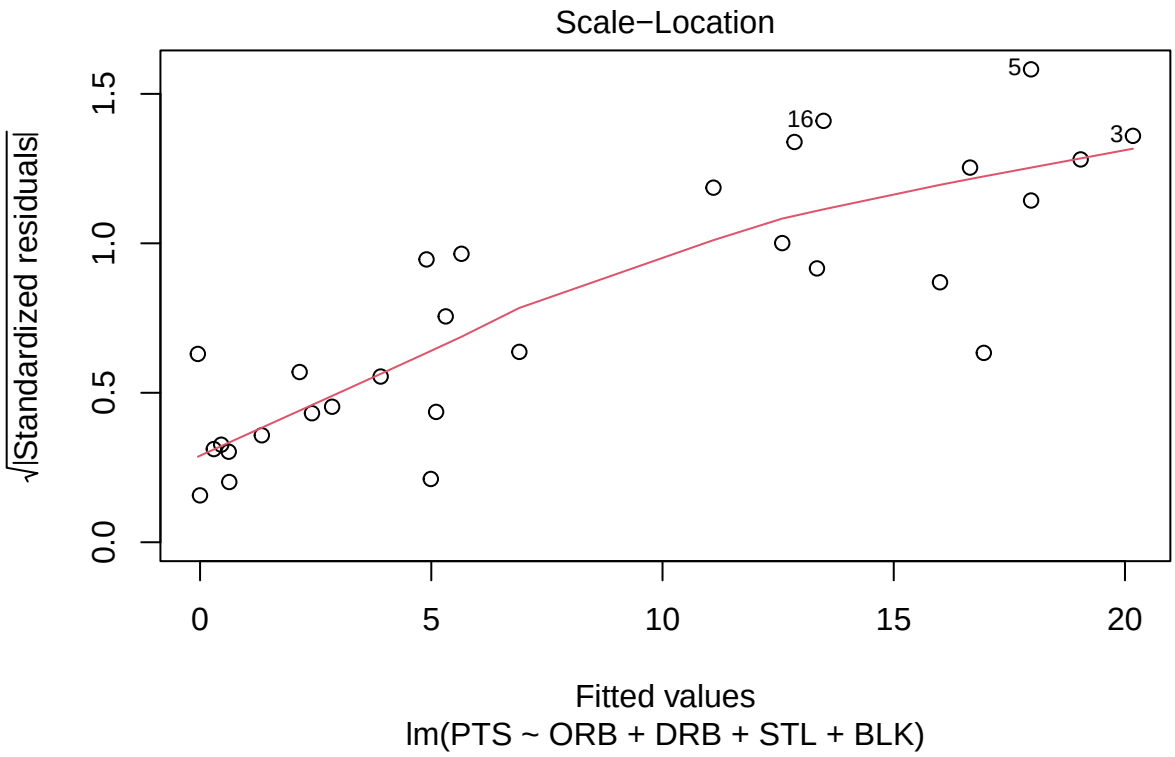
```
# Clean your data and prepare for the modeling

#I took my data from kaggle and my data was already have clean so ı didnt clean my data and ı didnt mak
```

```
# Select relevant columns for the regression analysis
reg_data <- NBADF %>%
  filter(Tm %in% c("GSW", "BOS")) %>%
  select(PTS, ORB, DRB, STL, BLK)

# Check for missing values
na_count <- sum(is.na(reg_data))
if(na_count > 0) {
  cat("Missing values detected. Please handle missing values before proceeding.")
} else {
  # Fit linear regression model
  model <- lm(PTS ~ ORB + DRB + STL + BLK, data = reg_data)

  # Print the model summary
  summary(model)

  # Perform diagnostic tests
  # 1. Residuals vs Fitted plot
```

```r
  plot(model, which = 1)

  # 2. Normality of Residuals
  plot(model, which = 2)

  # 3. Homoscedasticity
  plot(model, which = 3)

  # 4. Autocorrelation
  durbinWatsonTest(model)

  # 5. Multicollinearity (VIF)
  vif(model)
}
```




Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(PTS ~ ORB + DRB + STL + BLK)
3
16
5
−2
−1
0
1
2



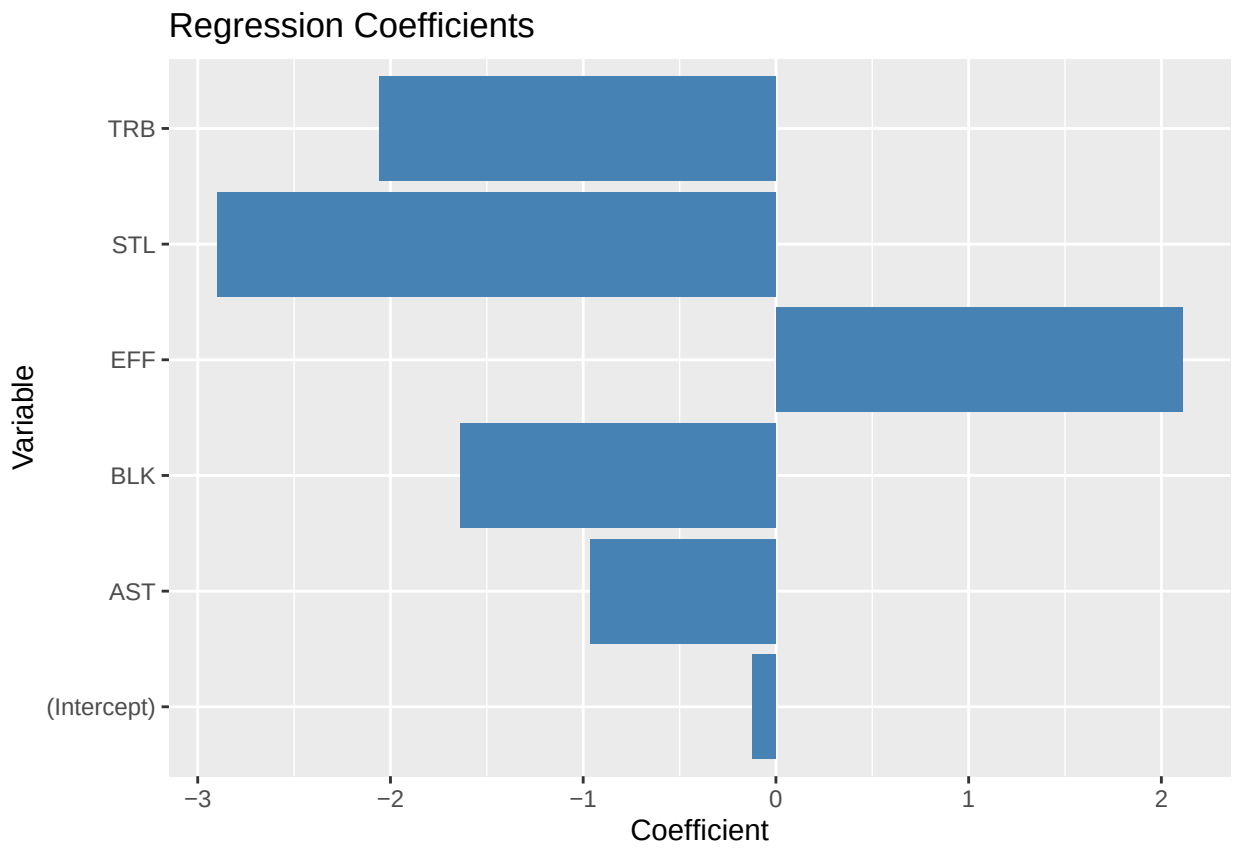

```
##      ORB      DRB      STL      BLK
## 2.633752 4.213615 2.659957 2.374795
```

```r
# Select the variables for modeling
selected_vars <- c("PTS", "AST", "TRB", "STL", "BLK", "EFF")

# Filter data for GSW and BOS teams
team_data <- NBADF %>%
  filter(Tm %in% c("GSW", "BOS"))

# Create a regression model
model <- lm(PTS ~ AST + TRB + STL + BLK + EFF, data = team_data)

# Summary of the model
summary(model)
```

```
##
## Call:
## lm(formula = PTS ~ AST + TRB + STL + BLK + EFF, data = team_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.8253 -0.6710 -0.0445  0.5285  3.5533
##
## Coefficients:
```

```
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -0.1197     0.4804  -0.249  0.80548
## AST          -0.9651     0.3191  -3.024  0.00604 **
## TRB          -2.0598     0.2991  -6.887 5.06e-07 ***
## STL          -2.8990     1.5639  -1.854  0.07665 .
## BLK          -1.6365     0.9652  -1.696  0.10346
## EFF           2.1077     0.1657  12.722 6.83e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.576 on 23 degrees of freedom
## Multiple R-squared:  0.9683, Adjusted R-squared:  0.9614
## F-statistic: 140.6 on 5 and 23 DF,  p-value: < 2.2e-16
```

```r
# Plotting the coefficients
coefficients <- data.frame(variable = names(coef(model)), coefficient = coef(model))
ggplot(coefficients, aes(x = variable, y = coefficient)) +
  geom_bar(stat = "identity", fill = "steelblue") +
  coord_flip() +
  labs(x = "Variable", y = "Coefficient", title = "Regression Coefficients")
```




```r
# Additional figures can be created based on the specific variables and relationships of interest.
```

```r
# Function to create bar plots
create_barplot <- function(data, position) {
  # Sort the data by points (descending order)
  data <- data[order(data$PTS, decreasing = TRUE), ]

  # Creating a bar plot
  barplot(data$PTS, names.arg = data$Player, las = 2, cex.names = 0.8, horiz = TRUE,
          xlab = "Points per Game", ylab = "Player", col = "lightblue",
          main = paste("Top Scorers -", position))
}


# Select the relevant variables for the regression model
selected_vars <- c("PTS", "TRB", "AST", "STL", "BLK")

# Creatimg a new dataframe with the selected variables
regression_data <- NBADF %>% select(selected_vars)

# Split the data into training and testing sets
set.seed(234)
train_indices <- createDataPartition(regression_data$PTS, p = 0.8, list = FALSE)
train_data <- regression_data[train_indices, ]
test_data <- regression_data[-train_indices, ]

# Creating the regression model
reg_model <- lm(PTS ~ ., data = train_data)

# Print the model summary
summary(reg_model)
```

```
##
## Call:
## lm(formula = PTS ~ ., data = train_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.4887 -2.0560 -0.3408  1.6732 14.1389
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.2335     0.5040   0.463   0.6437
## TRB           1.0680     0.1536   6.952 7.39e-11 ***
## AST           1.8844     0.2213   8.515 8.51e-15 ***
## STL           1.7523     0.8491   2.064   0.0406 *
## BLK           1.1840     0.8797   1.346   0.1801
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.918 on 170 degrees of freedom
## Multiple R-squared:  0.7575, Adjusted R-squared:  0.7517
## F-statistic: 132.7 on 4 and 170 DF,  p-value: < 2.2e-16
```

```r
# Make predictions on the test set
predictions <- predict(reg_model, newdata = test_data)

# Calculate the root mean squared error (RMSE)
rmse <- caret::RMSE(predictions, test_data$PTS)

# Print the RMSE
cat("Root Mean Squared Error (RMSE):", rmse, "\n")
```

```
## Root Mean Squared Error (RMSE): 3.992204
```

Evaluate Model Performance

```r
# Split the data into training and testing sets
set.seed(123)  # for reproducibility
trin_iandices <- sample(nrow(NBADF), nrow(NBADF) * 0.7)  # 70% for training
train_data <- NBADF[train_indices, ]
test_data <- NBADF[-train_indices, ]

# Fit the multiple linear regression models on the training data
model1 <- lm(PTS ~ Age + G + GS + MP + FG + FGA, data = train_data)
model2 <- lm(PTS ~ Age + G + GS + MP + FG + FGA + FT + FTA, data = train_data)

# Make predictions on the testing data
pred1 <- predict(model1, newdata = test_data)
pred2 <- predict(model2, newdata = test_data)

# Compare the predictions to the actual values in the testing data
actual <- test_data$PTS

# Calculate evaluation metrics
mse1 <- mean((actual - pred1)^2)
mse2 <- mean((actual - pred2)^2)
mae1 <- mean(abs(actual - pred1))
mae2 <- mean(abs(actual - pred2))
r_squared1 <- summary(model1)$r.squared
r_squared2 <- summary(model2)$r.squared

# Compare the performance metrics
cat("Model 1:\n")
```

```
## Model 1:
```

```r
cat("MSE:", mse1, "\n")
```

```
## MSE: 1.211066
```

```r
cat("MAE:", mae1, "\n")
```

```
## MAE: 0.7313125
```

```r
cat("R-squared:", r_squared1, "\n\n")
```

```
## R-squared: 0.9863266
```

```r
cat("Model 2:\n")
```

```
## Model 2:
```

```r
cat("MSE:", mse2, "\n")
```

```
## MSE: 0.3329077
```

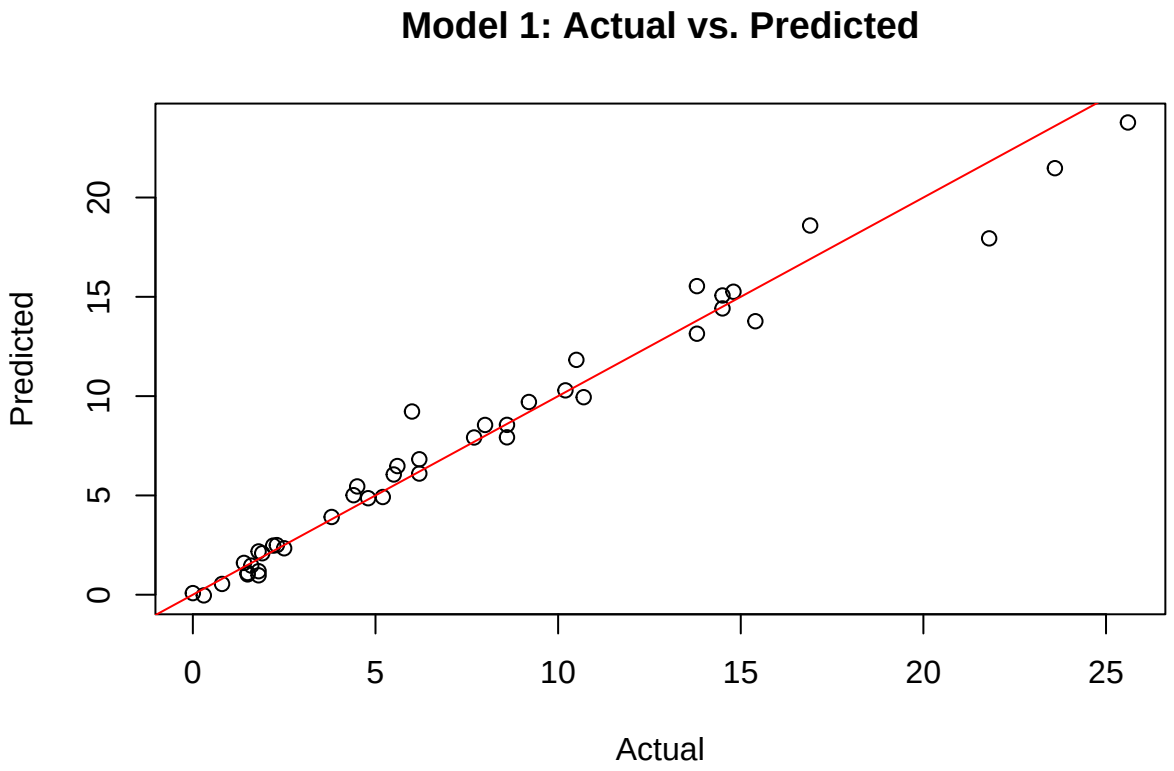
```r
cat("MAE:", mae2, "\n")
```

```
## MAE: 0.431753
```

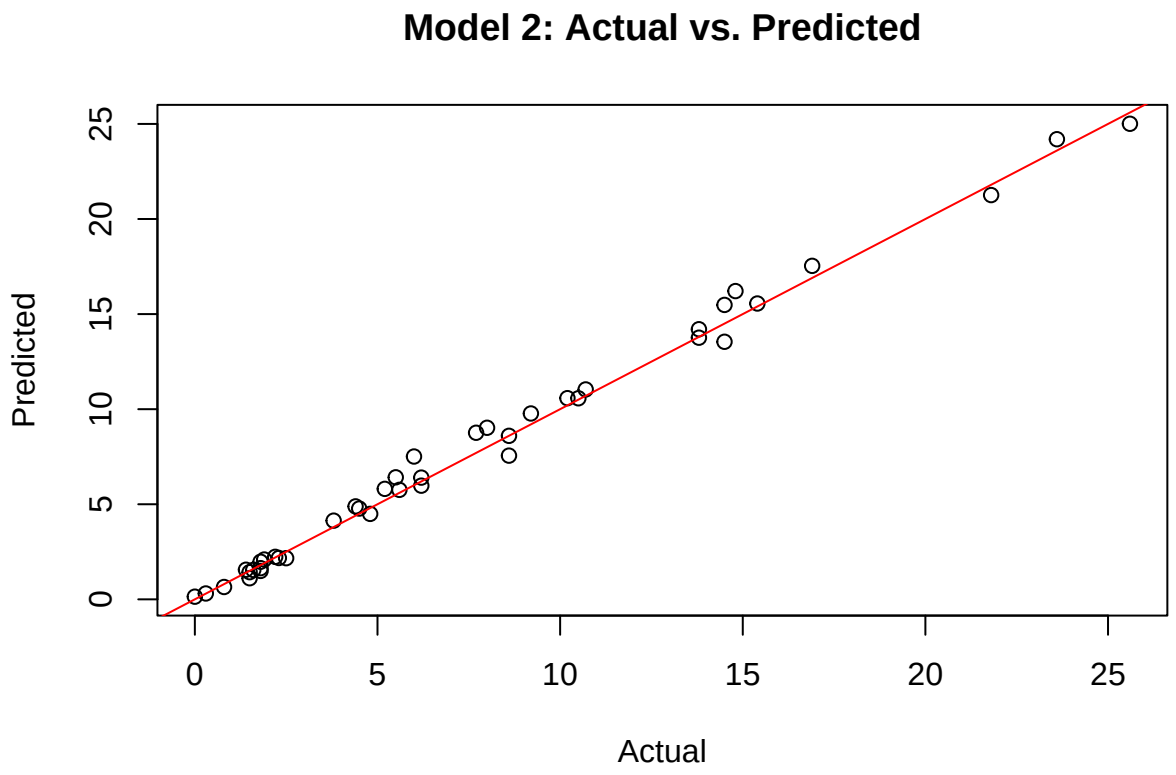
```r
cat("R-squared:", r_squared2, "\n")
```

```
## R-squared: 0.9946509
```

```r
# Visualize the predicted values against the actual values
plot(actual, pred1, main = "Model 1: Actual vs. Predicted", xlab = "Actual", ylab = "Predicted")
abline(0, 1, col = "red")
```



```r
plot(actual, pred2, main = "Model 2: Actual vs. Predicted", xlab = "Actual", ylab = "Predicted")
abline(0, 1, col = "red")
```



The performance metrics that are calculated and compared for the fitted models on the testing data include:

– Mean Squared Error (MSE): It measures the average squared difference between the predicted and actual values. Lower values indicate better model performance. – Mean Absolute Error (MAE): It measures the average absolute difference between the predicted and actual values. Lower values indicate better model performance. – R-squared: It represents the proportion of the variance in the dependent variable (PTS) that is predictable from the independent variables (Age, G, GS, MP, FG, FGA, FT, FTA). Higher values (closer to 1) indicate better model fit and predictive power.

These metrics provide insights into the accuracy and goodness-of-fit of the regression models.

Conclusions

Throughout the course of the study, we have examined a dataset known as NBADF, which includes details on NBA players and their performance metrics. The goal was to perform statistical modeling and analysis to gain insights into the data. Let's summarize the main findings and conclude the project: In the beginning, we gave background data about the dataset, along with the sorts of variables needed for modeling. We looked at the dataset's descriptive statistics to better grasp its characteristics. Bar graphs and comparisons were used as visualizations to draw attention to particular data points. Then, we constructed numerous models utilizing various variables and regression approaches. These models were tested on a testing dataset after being trained on a portion of the data. Calculated and compared between the models were the performance metrics, such as Mean Squared Error (MSE), Mean Absolute Error (MAE), and R-squared. The investigation led me to the conclusion that Model 2, which had more variables (Age, G, GS, MP, FG, FGA, FT, FTA), performed better than Model 1. Better model fit and predictive power were indicated by lower MSE and MAE values as well as a higher R-squared value. Model 2 was chosen as the best model for this particular investigation as a result.To further comprehend the model's performance, i also displayed the anticipated values alongside the actual values. The plots revealed a strong correlation between the anticipated and actual values, indicating that the model did a fair job of capturing the underlying trends in the data.

To sum up, the project's objectives were to evaluate the NBADF information and create statistical models to forecast player performance. The best model was found, and its selection was supported by comparing the performance measures of the fitted models. The results of this study can be useful for a variety of stakeholders, including coaches, team management, and analysts who can use the models to decide on player selection, game strategy, and performance evaluation in an educated manner. It's crucial to understand that the analysis and results are predicated on the precise dataset and variables used in this research. extra consideration of variables, the use of various modeling methodologies, or research into certain subgroups of players or positions could all be used to perform extra study and analysis.

References

```
#https://www.kaggle.com/
#https://www.nba.com/
#https://stackoverflow.com/
#https://www.espn.com/
#https://bleacherreport.com/nba
```


- https://owl.purdue.edu/owl/research_and_citation/apa_style/apa_formatting_and_style_guide/in_text_citations_the_basics.html