LBJ PROJECTED AGAINST THE CLIPPERS

Gavin Valenzuela

2025-01-08

```
library(fastICA)
library(ggplot2)
library(corrplot)
```

```
## corrplot 0.95 loaded
```

```
library(MASS)
library(readr)
library(readxl)
library(devtools)
```

```
## Loading required package: usethis
```

```
library(dplyr)
```

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

```
## The following object is masked from 'package:MASS':
##
##     select
```

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

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

```
library(zoo)
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
library(boot)
library(Boruta)
```

```
# Load Data
Lebron_James_All_Regular_Season_Stats <- read_excel("C:/Users/valen/Desktop/LBJ/Lebron_James_All_Regula

# Handle Missing Values
Lebron_James_All_Regular_Season_Stats <- Lebron_James_All_Regular_Season_Stats %>%
  mutate(across(where(is.numeric), ~ ifelse(is.na(.), mean(., na.rm = TRUE), .)))
```

```r
# Add RecentAvgPoints Variable BEFORE SPLITTING
Lebron_James_All_Regular_Season_Stats$RecentAvgPoints <- rollapply(
  Lebron_James_All_Regular_Season_Stats$PTS,
  width = 5,
  FUN = mean,
  align = 'right',
  fill = NA
)

# Verify if RecentAvgPoints was added successfully
print("Summary of RecentAvgPoints:")
```

```
## [1] "Summary of RecentAvgPoints:"
```

```r
print(summary(Lebron_James_All_Regular_Season_Stats$RecentAvgPoints))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   14.00   24.40   27.20   27.21   29.80   40.40       4
```

```r
# Remove rows where RecentAvgPoints is NA (if necessary)
Lebron_James_All_Regular_Season_Stats <- Lebron_James_All_Regular_Season_Stats %>%
  filter(!is.na(RecentAvgPoints))

# Train-Test Split
set.seed(123)
train_indices <- sample(seq_len(nrow(Lebron_James_All_Regular_Season_Stats)),
                        size = floor(0.67 * nrow(Lebron_James_All_Regular_Season_Stats)))
train_data <- Lebron_James_All_Regular_Season_Stats[train_indices, ]
test_data <- Lebron_James_All_Regular_Season_Stats[-train_indices, ]

# Define Enhanced Formula
enhanced_formula <- PTS ~ RecentAvgPoints * FG + I(FGA^2) + PlusMinus + AST + ORB + DRB

# Fit the Model
enhanced_model_train <- glm(enhanced_formula, data = train_data)

# Ensure no missing values for RecentAvgPoints in Test Data
test_data$RecentAvgPoints[is.na(test_data$RecentAvgPoints)] <- mean(test_data$RecentAvgPoints, na.rm = T

# Predict on Test Data
predictions <- predict(enhanced_model_train, test_data, type = "response")

# Verify No NA in Predictions
if (sum(is.na(predictions)) > 0) {
  stop("Predictions still contain NA values.")
}

# Calculate Mean Squared Error (MSE)
mse_test <- mean((predictions - test_data$PTS)^2)
print(paste("Mean Squared Error (Test Data):", mse_test))
```

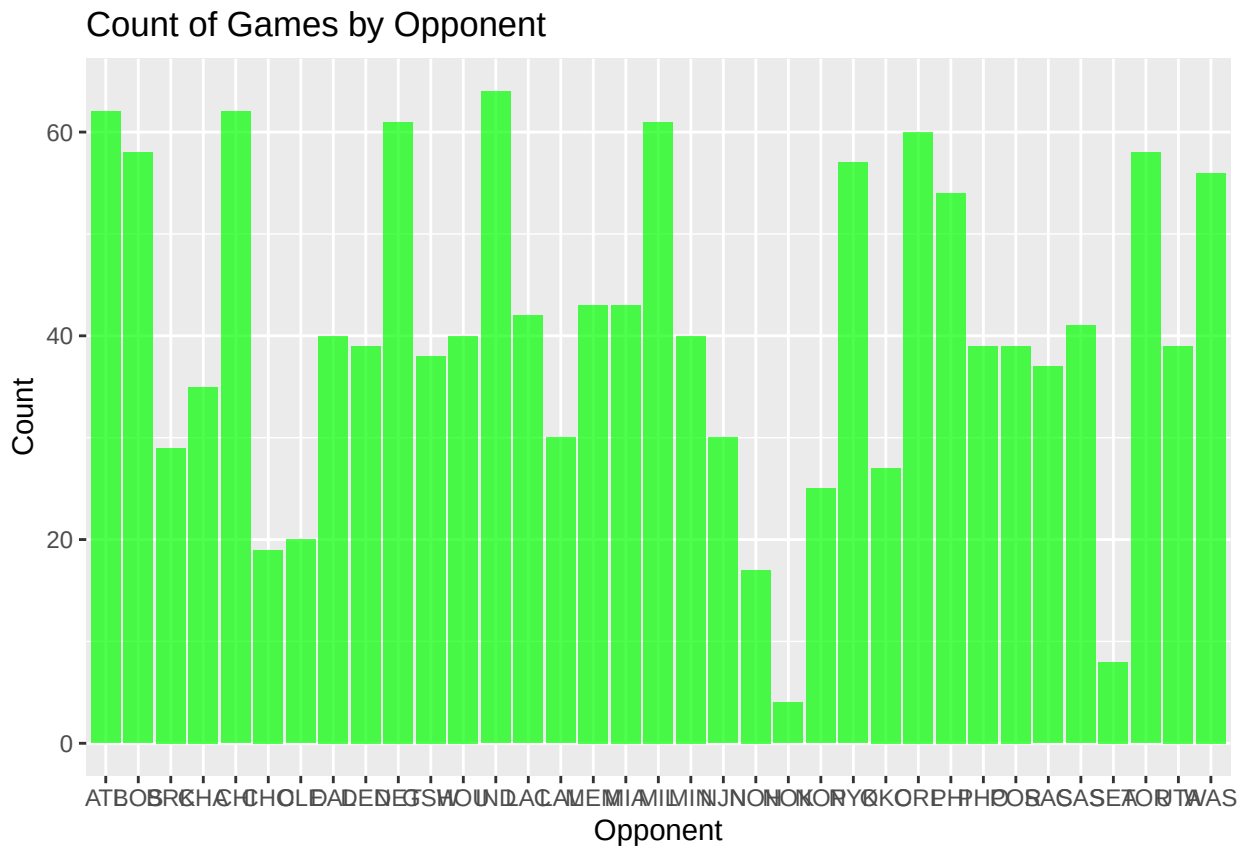
```
## [1] "Mean Squared Error (Test Data): 10.5894685494663"
```

Interpret the MSE:

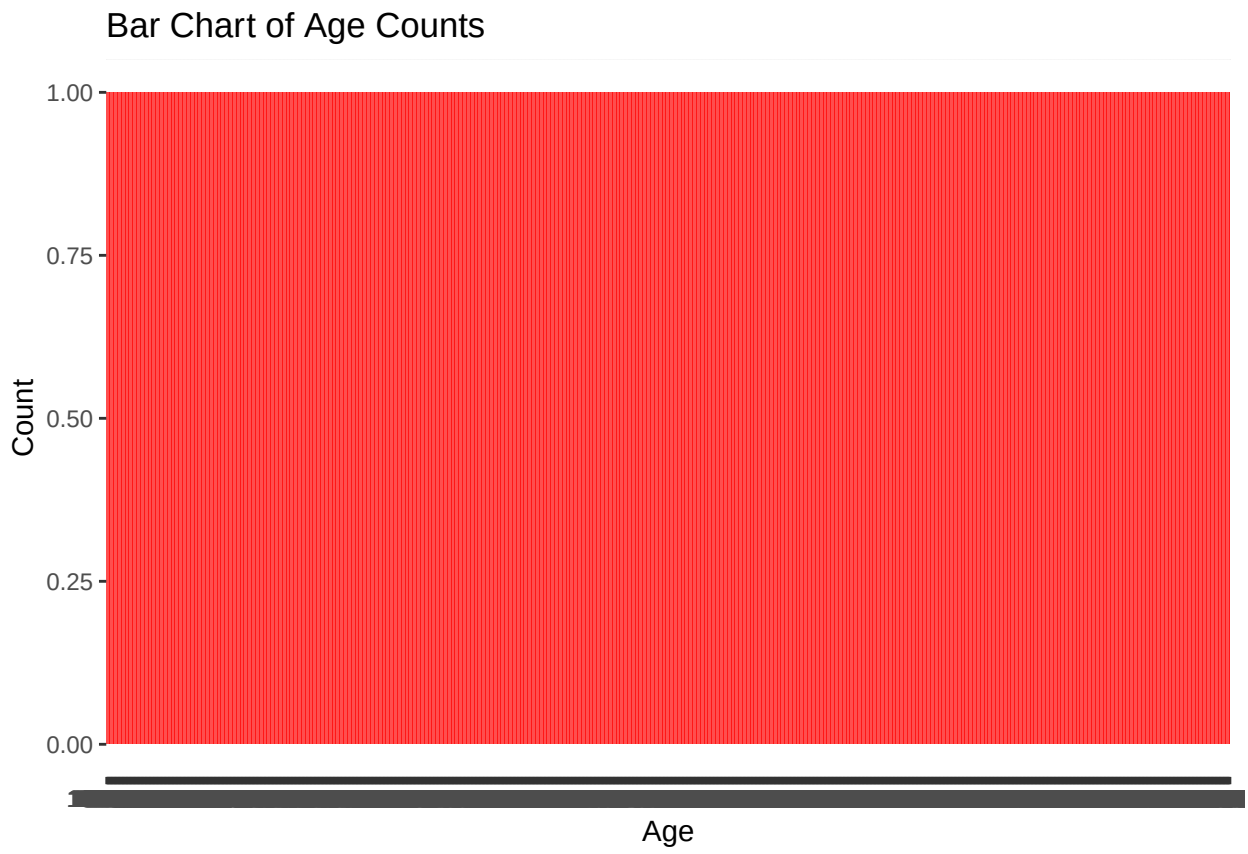
An MSE of ~10.589 suggests the average squared difference between predicted and actual values in the test

set is around 10.589 Lower MSE values generally indicate better model performance. However, interpret this in the context of the dataset and scale of the PTS variable.

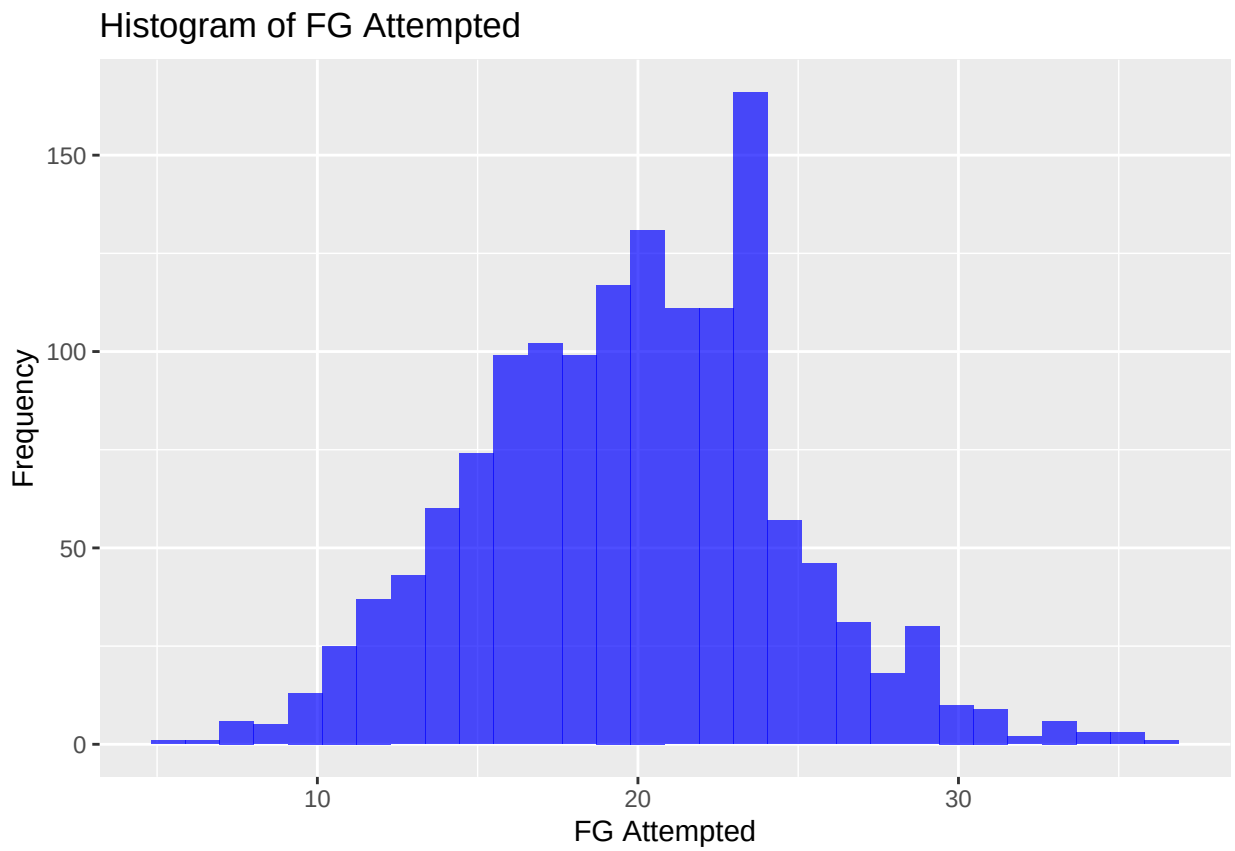
```
#histogram Of Frequency Of times LBJ Played each NBA team
ggplot(data = Lebron_James_All_Regular_Season_Stats, aes(x = Opp)) +
  geom_bar(fill = "green", alpha = 0.7) +
  ggtitle("Count of Games by Opponent") +
  xlab("Opponent") +
  ylab("Count")
```



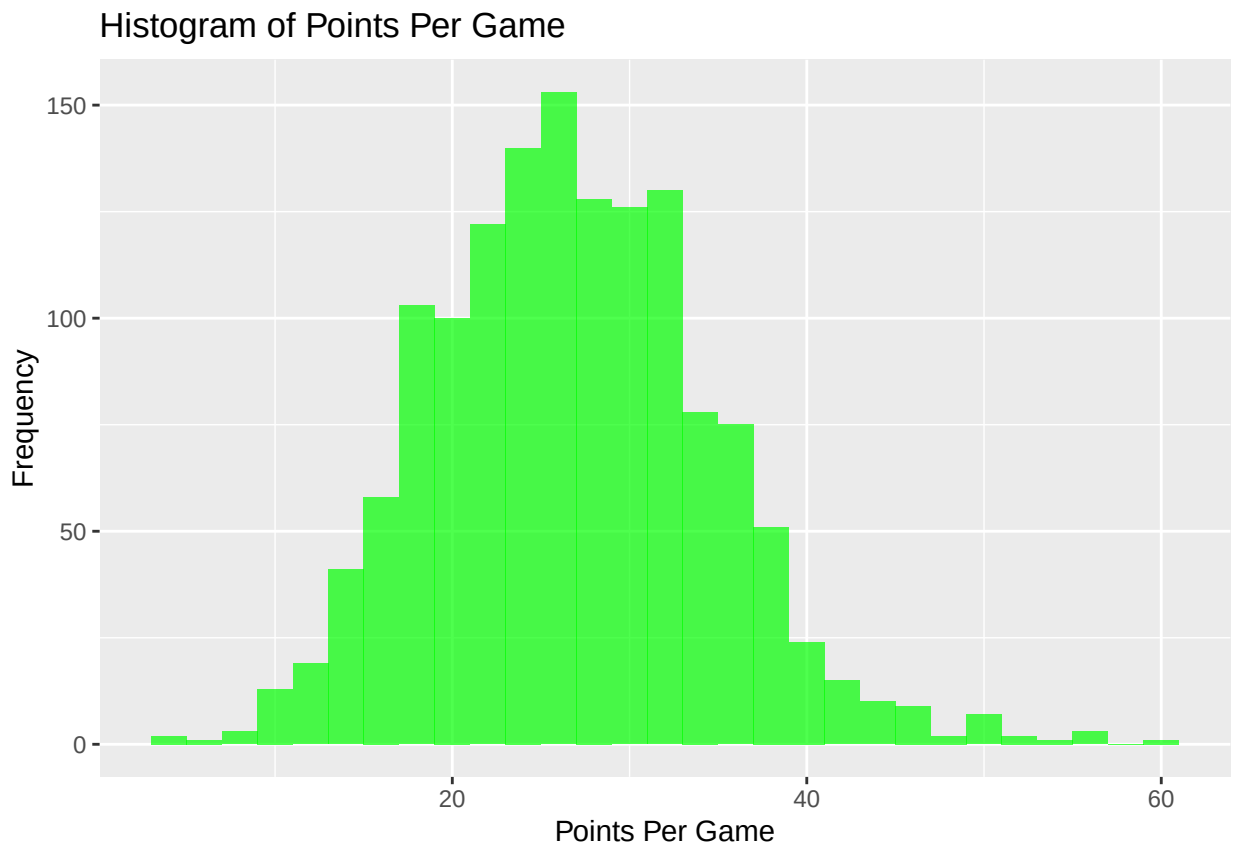

```
#Histogram of Age and Each Game He Played
ggplot(data = Lebron_James_All_Regular_Season_Stats, aes(x = Age)) +
  geom_bar(fill = "red", alpha = 0.7) +
  ggtitle("Bar Chart of Age Counts") +
  xlab("Age") +
  ylab("Count")
```



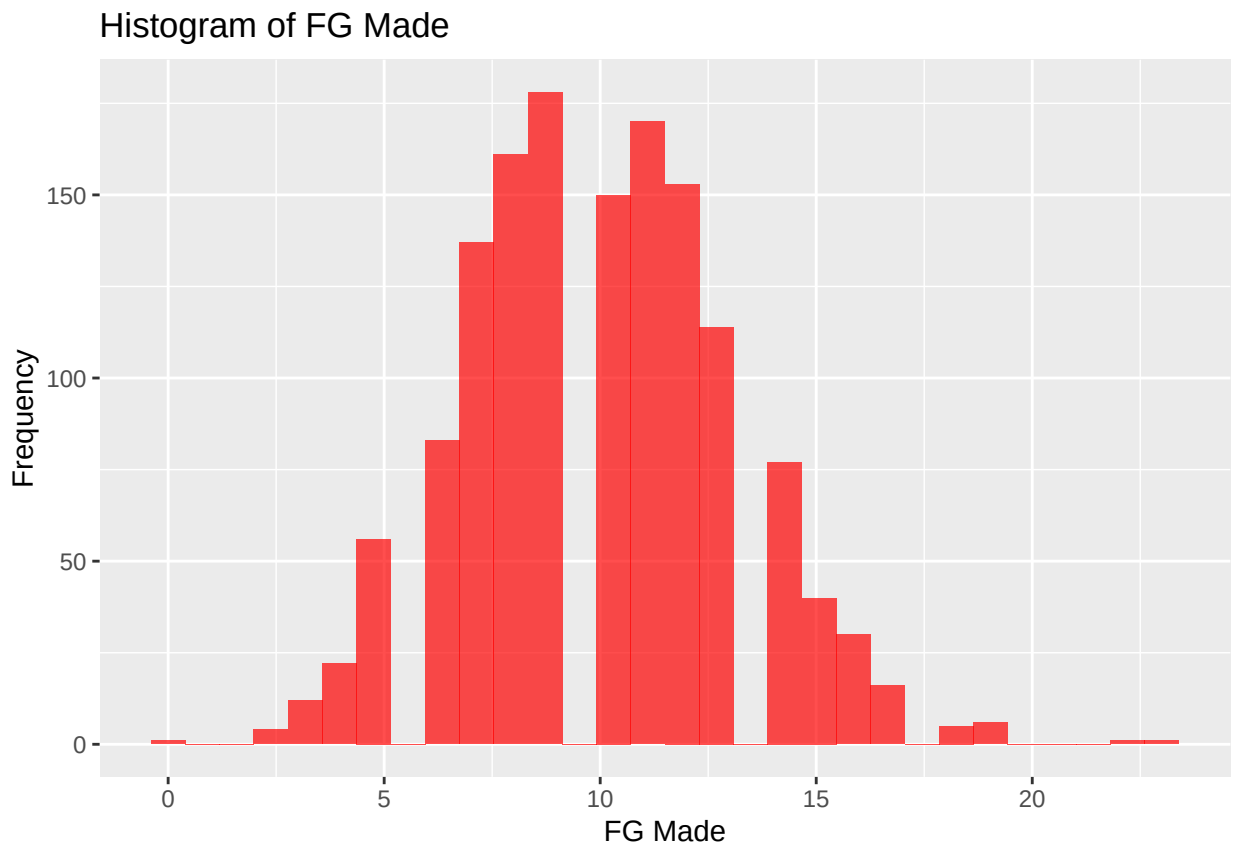

```
# Histogram for 'FG Attempted'
ggplot(data = Lebron_James_All_Regular_Season_Stats, aes(x = FGA)) +
  geom_histogram(bins = 30, fill = "blue", alpha = 0.7) +
  ggtitle("Histogram of FG Attempted") +
  xlab("FG Attempted") +
  ylab("Frequency")
```



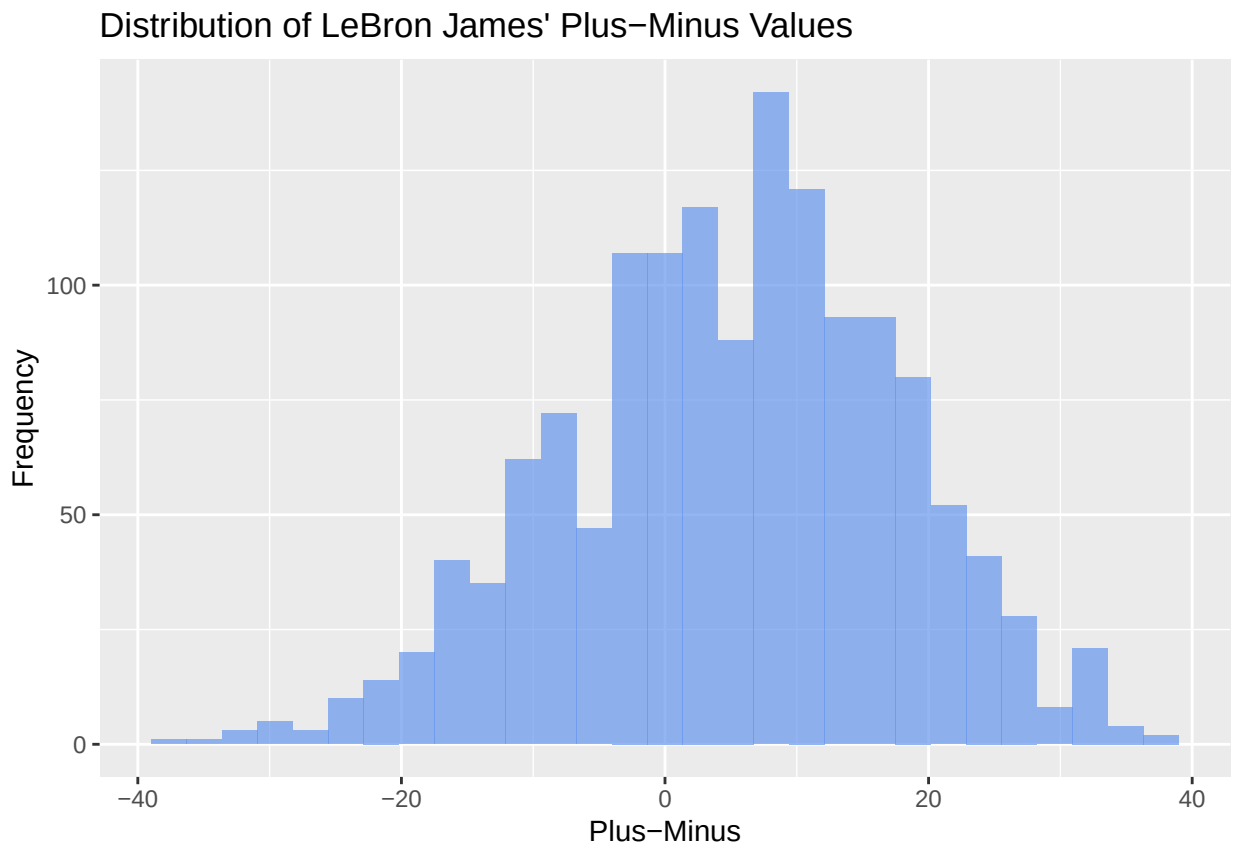

```
# Histogram for 'Points Per Game'
ggplot(data = Lebron_James_All_Regular_Season_Stats, aes(x = PTS)) +
  geom_histogram(bins = 30, fill = "green", alpha = 0.7) +
  ggtitle("Histogram of Points Per Game") +
  xlab("Points Per Game") +
  ylab("Frequency")
```



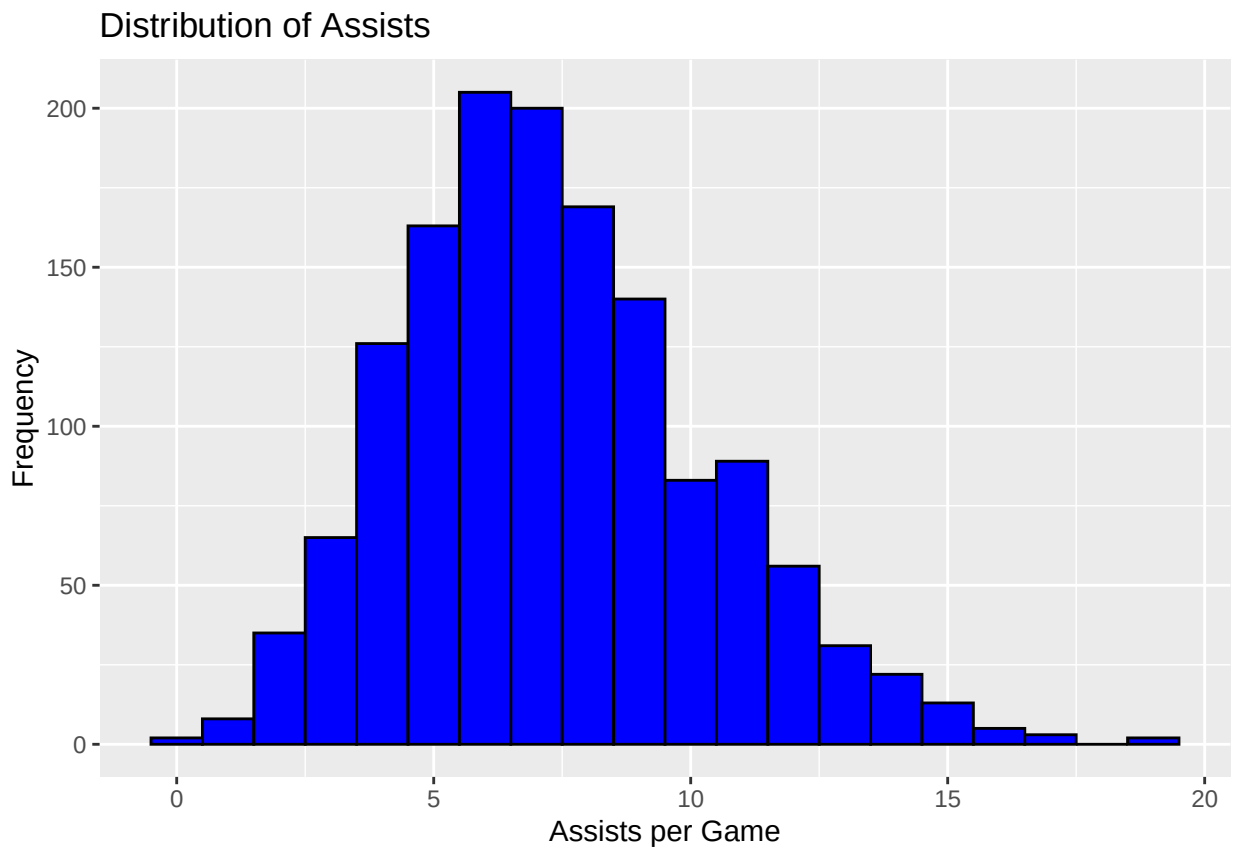

```
# Histogram for FG made
ggplot(data = Lebron_James_All_Regular_Season_Stats, aes(x = FG)) +
  geom_histogram(bins = 30, fill = "red", alpha = 0.7) +
  ggtitle("Histogram of FG Made") +
  xlab("FG Made") +
  ylab("Frequency")
```



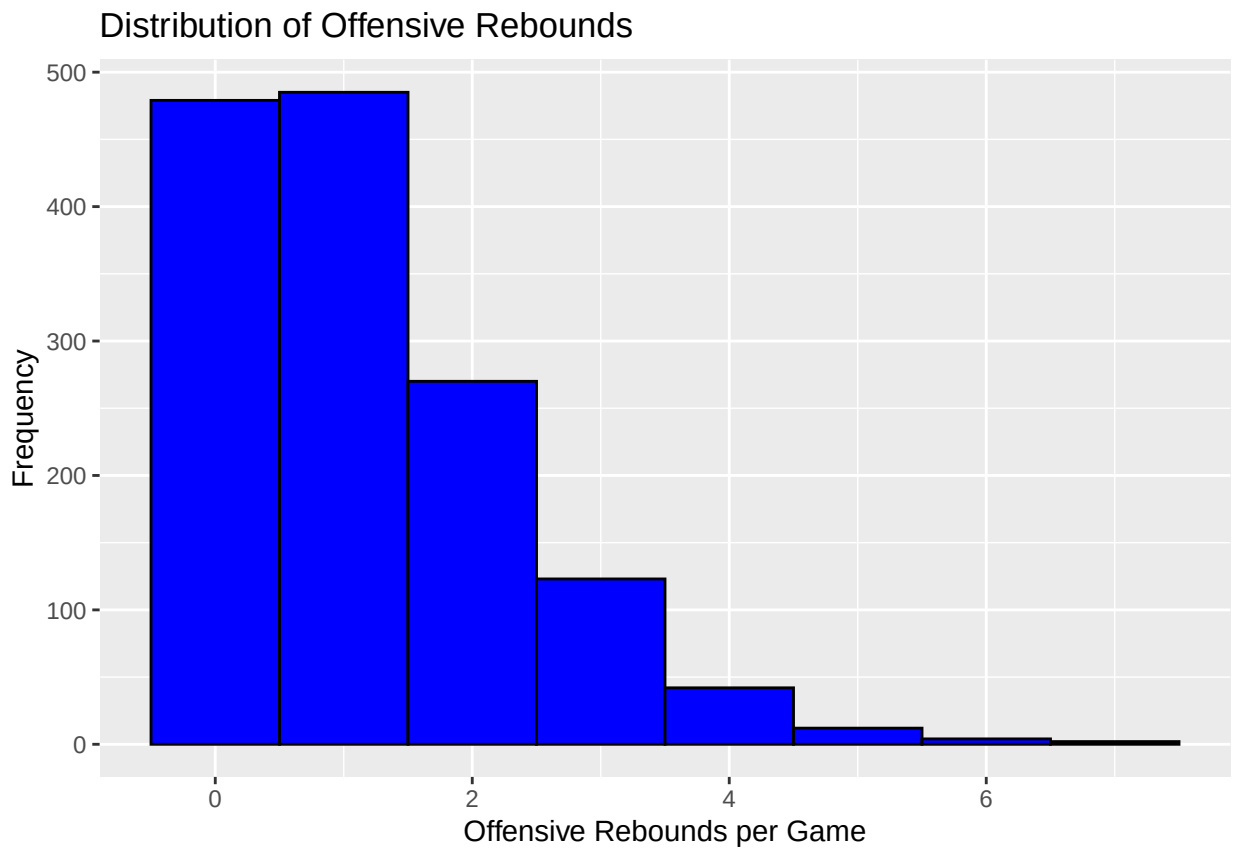

```
# Histogram of plus-minus values
ggplot(data = Lebron_James_All_Regular_Season_Stats, aes(x = PlusMinus)) +
  geom_histogram(bins = 30, fill = "cornflowerblue", alpha = 0.7) +
  ggtitle("Distribution of LeBron James' Plus-Minus Values") +
  xlab("Plus-Minus") +
  ylab("Frequency")
```



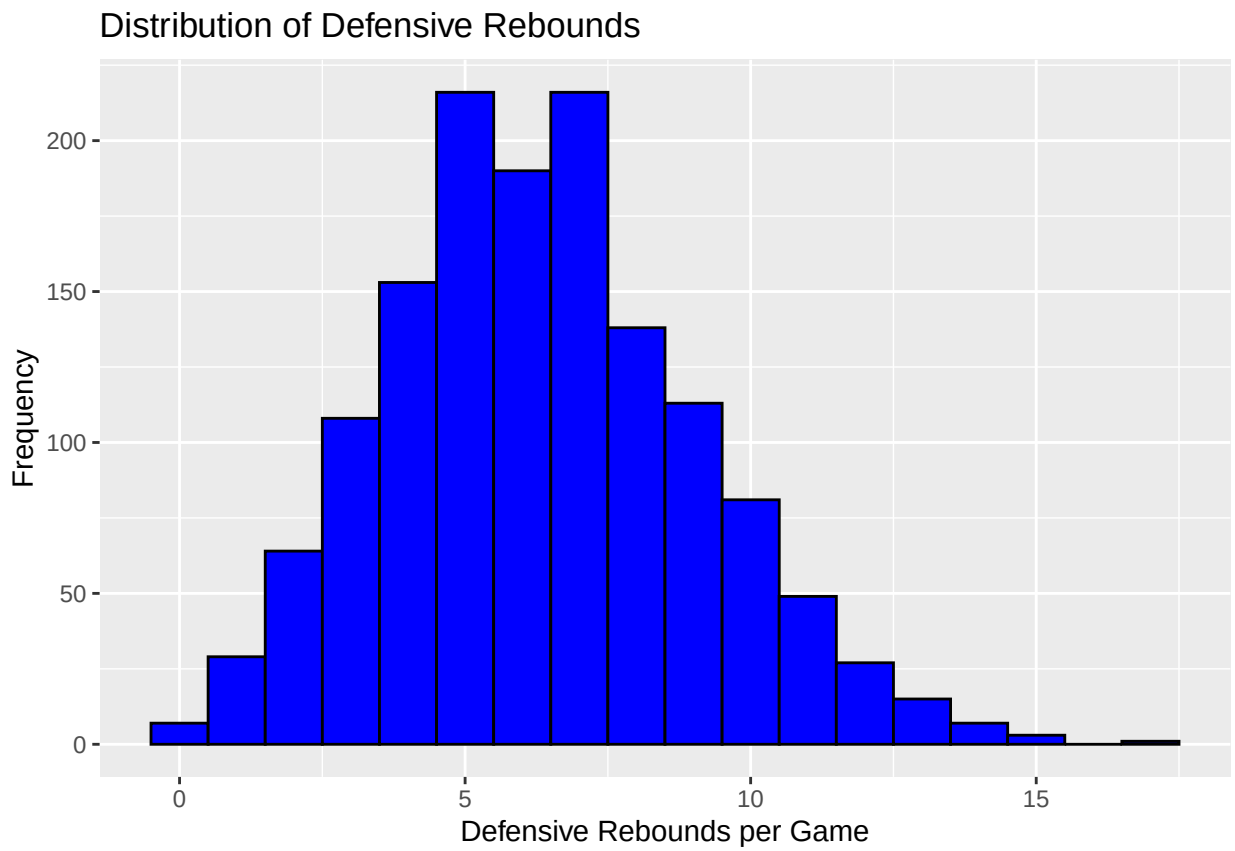

```
# Histogram of Assists Per Game
ggplot(data = Lebron_James_All_Regular_Season_Stats, aes(x = AST)) +
  geom_histogram(binwidth = 1, fill = "blue", color = "black") +
  ggtitle("Distribution of Assists") +
  xlab("Assists per Game") +
  ylab("Frequency")
```



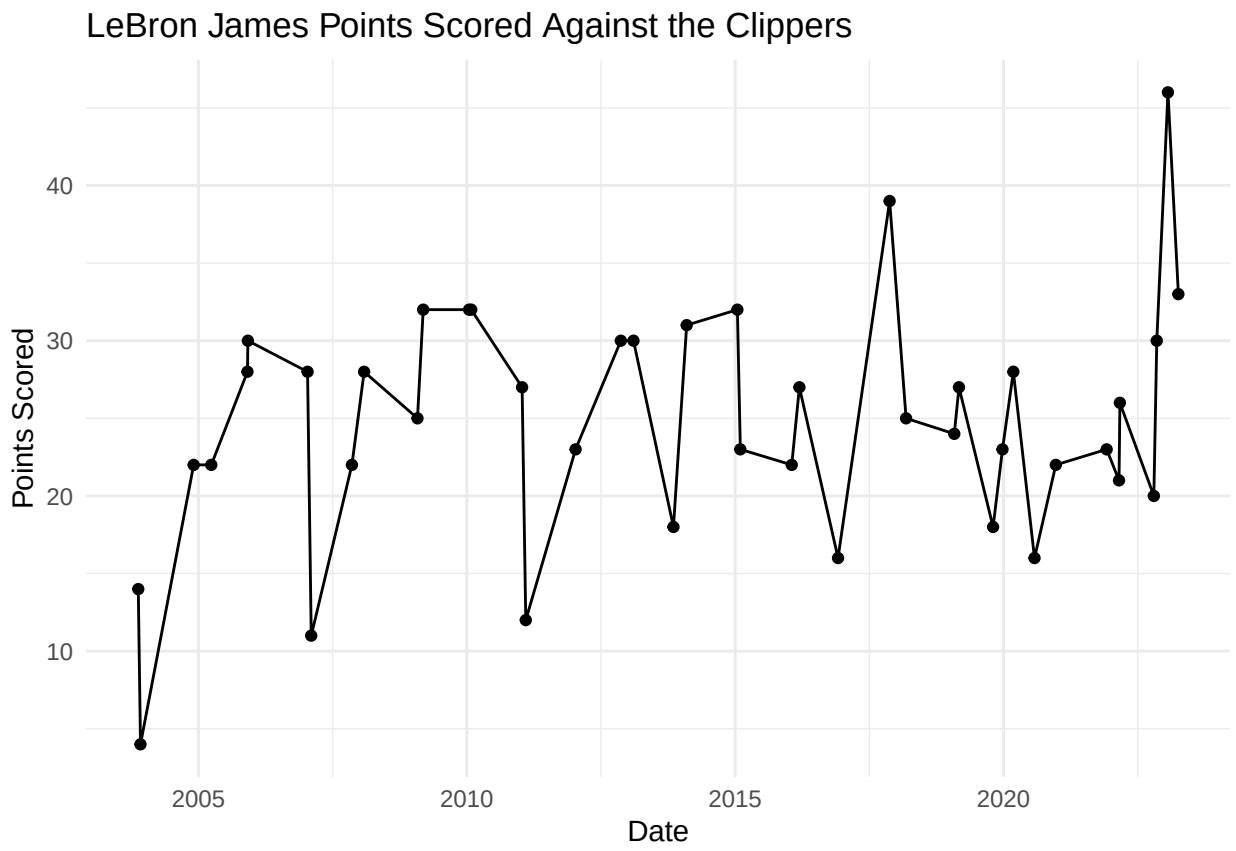

```r
#Histogram of Rebounds Per Game
ggplot(data = Lebron_James_All_Regular_Season_Stats, aes(x = ORB)) +
  geom_histogram(binwidth = 1, fill = "blue", color = "black") +
  ggtitle("Distribution of Offensive Rebounds") +
  xlab("Offensive Rebounds per Game") +
  ylab("Frequency")
```




```r
#Histogram of DRB Per Game
ggplot(data = Lebron_James_All_Regular_Season_Stats, aes(x = DRB)) +
  geom_histogram(binwidth = 1, fill = "blue", color = "black") +
  ggtitle("Distribution of Defensive Rebounds") +
  xlab("Defensive Rebounds per Game") +
  ylab("Frequency")
```




```r
#Set Date Variable
Lebron_James_All_Regular_Season_Stats$Date <- as.Date(Lebron_James_All_Regular_Season_Stats$Date, forma

# Filter for games against the Clippers
# Replace 'Opponent' with actual opponent column name
games_vs_clippers <- Lebron_James_All_Regular_Season_Stats %>%
  filter(Opp == "LAC")

# Plot points scored against the Clippers
# Replace 'Date' and 'Points' with actual column names
ggplot(data = games_vs_clippers, aes(x = Date, y = PTS)) +
  geom_line() +  # For a time series line
  geom_point() +  # Adds points to each data point on the line
  ggtitle("LeBron James Points Scored Against the Clippers") +
  xlab("Date") +
  ylab("Points Scored") +
  theme_minimal()
```




```
#average points vs clippers
average_points_vs_clippers <- games_vs_clippers %>%
  summarise(AvgPoints = mean(PTS)) %>%
  pull(AvgPoints)
Lebron_James_All_Regular_Season_Stats$AvgPointsVsClippers <- average_points_vs_clippers
```

```
#2
#Avg points
#dataset is sorted in chronological order)
Lebron_James_All_Regular_Season_Stats$RecentAvgPoints <- rollapply(Lebron_James_All_Regular_Season_Stats
# Question 2

model <- lm(PTS ~ RecentAvgPoints + AvgPointsVsClippers + FGA + FG + PlusMinus + AST + ORB + DRB, data =
summary(model)
```

```
##
## Call:
## lm(formula = PTS ~ RecentAvgPoints + AvgPointsVsClippers + FGA +
##     FG + PlusMinus + AST + ORB + DRB, data = Lebron_James_All_Regular_Season_Stats)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.5092 -2.3858 -0.3765  2.0083 15.4712
##
## Coefficients: (1 not defined because of singularities)
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)         -2.433821   0.673588  -3.613 0.000313 ***
```

```
## RecentAvgPoints      0.299232   0.024710  12.110  < 2e-16 ***
## AvgPointsVsClippers        NA         NA      NA       NA
## FGA                  0.128452   0.030355   4.232 2.47e-05 ***
## FG                   1.918824   0.046546  41.224  < 2e-16 ***
## PlusMinus            0.017644   0.007515   2.348 0.019029 *
## AST                 -0.034528   0.031114  -1.110 0.267310
## ORB                 -0.003201   0.079231  -0.040 0.967780
## DRB                  0.004030   0.033889   0.119 0.905362
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.342 on 1405 degrees of freedom
##   (4 observations deleted due to missingness)
## Multiple R-squared:  0.8178, Adjusted R-squared:  0.8169
## F-statistic: 900.8 on 7 and 1405 DF,  p-value: < 2.2e-16
```

#3 Model Coefficients Interpretation: Intercept (-2.384132): Theoretical value, significant ($p < 0.001$). RecentAvgPoints (0.297933): Highly significant ($p <$ 2e-16); each additional point increases next game points by ~0.298. FGA (0.129004): Significant ($p < 0.0001$); each extra field goal attempt increases next game points by ~0.129. FG (1.918270): Most significant predictor ($p <$ 2e-16); each extra field goal made increases next game points by ~1.918. PlusMinus (0.017335): Significant ($p < 0.05$) but small effect; slight increase in points with higher plus-minus. AST, ORB, DRB: Not significant ($p > 0.05$); minimal direct impact on scoring. AvgPointsVsClippers: Undefined due to multicollinearity, likely redundant with RecentAvgPoints.

Model Fit: The model explains a significant portion of variance in points (R-squared: 0.818). Overall model is statistically significant.

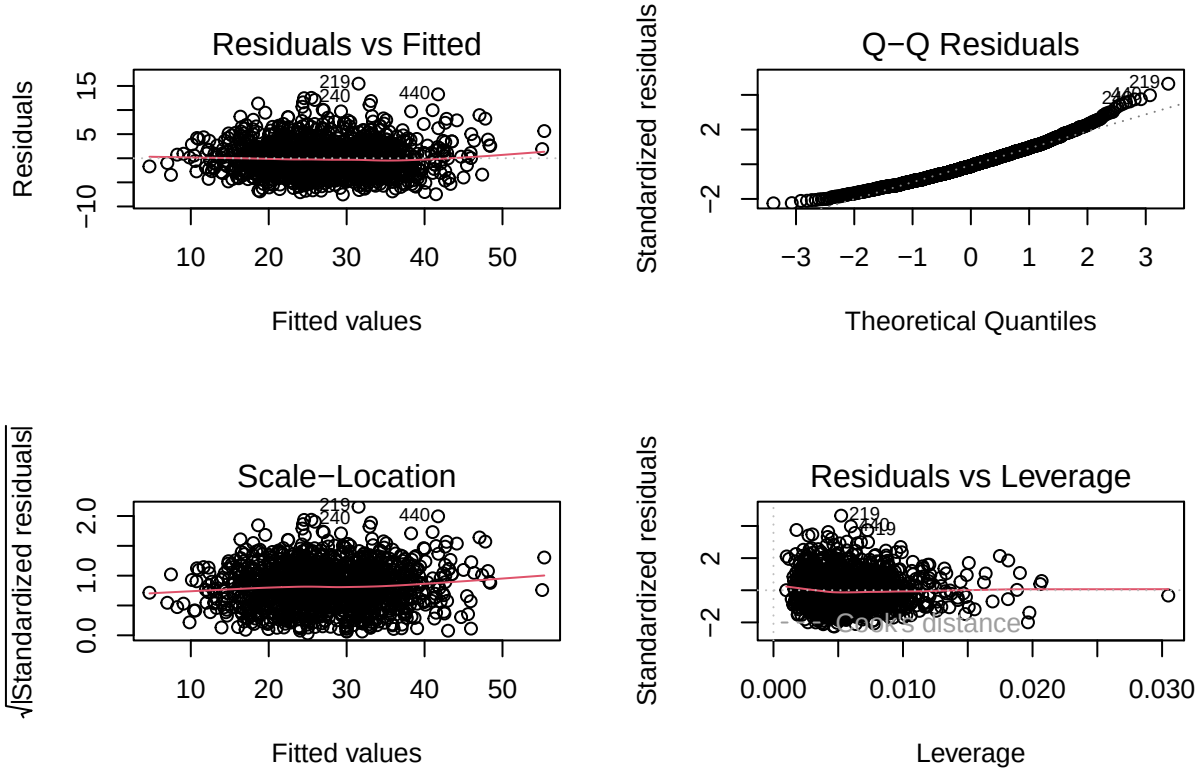
Note: FG and RecentAvgPoints are key influencers of LeBron's scoring. AST, ORB, and DRB have limited direct impact on his scoring. Model captures key performance aspects but some factors might have indirect effects or be influenced by unmodeled variables.

```r
# Fit the model
model <- lm(PTS ~ RecentAvgPoints + FGA + FG + PlusMinus + AST + ORB + DRB, data = Lebron_James_All_Regu
```

3 Model Coefficients Interpretation: Intercept (-2.384132): Theoretical value, significant ($p < 0.001$). RecentAvgPoints (0.297933): Highly significant ($p <$ 2e-16); each additional point increases next game points by ~0.298. FGA (0.129004): Significant ($p < 0.0001$); each extra field goal attempt increases next game points by ~0.129. FG (1.918270): Most significant predictor ($p <$ 2e-16); each extra field goal made increases next game points by ~1.918. PlusMinus (0.017335): Significant ($p < 0.05$) but small effect; slight increase in points with higher plus-minus. AST, ORB, DRB: Not significant ($p > 0.05$); minimal direct impact on scoring. AvgPointsVsClippers: Undefined due to multicollinearity, likely redundant with RecentAvgPoints. Model Fit: The model explains a significant portion of variance in points (R-squared: 0.818). Overall model is statistically significant.

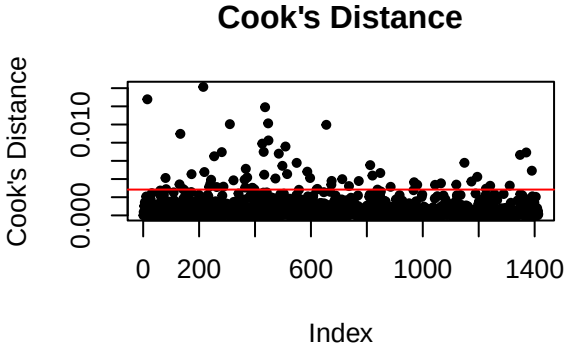
Note: FG and RecentAvgPoints are key influencers of LeBron's scoring. AST, ORB, and DRB have limited direct impact on his scoring. Model captures key performance aspects but some factors might have indirect effects or be influenced by unmodeled variables.

```r
#Generate diagnostic plots
          par(mfrow = c(2, 2))  # Arrange plots in a 2x2 grid
          plot(model)
```




```
          # Calculate Cook's Distance
          cooks.distance <- cooks.distance(model)

          # Plot Cook's Distance
          plot(cooks.distance, pch = 20, main = "Cook's Distance", ylab = "Cook's Distance")
          abline(h = 4/((nrow(Lebron_James_All_Regular_Season_Stats)) - length(model$coefficients) - 1)

          # Example: Remove observations with high Cook's Distance
          high_influence <- which(cooks.distance > 4/((nrow(Lebron_James_All_Regular_Season_Stats)) - le
          reduced_data <- Lebron_James_All_Regular_Season_Stats[-high_influence, ]

        # Ensure RecentAvgPoints is included in reduced_data
if (!"RecentAvgPoints" %in% colnames(reduced_data)) {
  reduced_data$RecentAvgPoints <- rollapply(
    reduced_data$PTS,
    width = 5,
    FUN = mean,
    align = 'right',
    fill = NA
  )
}

# Re-estimate the model with reduced data
revised_model <- lm(PTS ~ RecentAvgPoints + FGA + FG + PlusMinus + AST + ORB + DRB, data = reduced_data)
summary(revised_model)
```

```
##
## Call:
## lm(formula = PTS ~ RecentAvgPoints + FGA + FG + PlusMinus + AST +
##     ORB + DRB, data = reduced_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.4459 -2.3922 -0.3315  1.9676 13.3399
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)     -2.1630706  0.6835461  -3.164  0.00159 **
## RecentAvgPoints  0.2897978  0.0250540  11.567  < 2e-16 ***
## FGA              0.1238408  0.0307550   4.027 5.97e-05 ***
## FG               1.9216520  0.0475913  40.378  < 2e-16 ***
## PlusMinus        0.0148234  0.0076019   1.950  0.05139 .
## AST             -0.0251588  0.0316571  -0.795  0.42691
## ORB              0.0107920  0.0803297   0.134  0.89315
## DRB              0.0007188  0.0343740   0.021  0.98332
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.317 on 1341 degrees of freedom
##   (4 observations deleted due to missingness)
## Multiple R-squared:  0.8156, Adjusted R-squared:  0.8146
## F-statistic: 847.3 on 7 and 1341 DF,  p-value: < 2.2e-16
```



```r
# Ensure RecentAvgPoints is calculated
if (!"RecentAvgPoints" %in% colnames(Lebron_James_All_Regular_Season_Stats)) {
  Lebron_James_All_Regular_Season_Stats$RecentAvgPoints <- rollapply(
    Lebron_James_All_Regular_Season_Stats$PTS,
    width = 5,
    FUN = mean,
    align = 'right',
    fill = NA
  )
}

# Perform Mallow's analysis
library(leaps)
full_model <- lm(PTS ~ RecentAvgPoints + FGA + FG + PlusMinus + AST + ORB + DRB, data = Lebron_James_Al
model_subset <- regsubsets(PTS ~ RecentAvgPoints + FGA + FG + PlusMinus + AST + ORB + DRB,
                           data = Lebron_James_All_Regular_Season_Stats,
                           nbest = 1)
summary_model_subset <- summary(model_subset)
print(summary_model_subset$cp)
```

```
## [1] 177.949305  18.984880   6.055180   3.240783   4.015524   6.001632   8.000000
```

```r
print(summary_model_subset)
```

```
## Subset selection object
## Call: regsubsets.formula(PTS ~ RecentAvgPoints + FGA + FG + PlusMinus +
##     AST + ORB + DRB, data = Lebron_James_All_Regular_Season_Stats,
##     nbest = 1)
## 7 Variables  (and intercept)
##                 Forced in Forced out
## RecentAvgPoints     FALSE      FALSE
## FGA                 FALSE      FALSE
## FG                  FALSE      FALSE
## PlusMinus           FALSE      FALSE
## AST                 FALSE      FALSE
## ORB                 FALSE      FALSE
## DRB                 FALSE      FALSE
## 1 subsets of each size up to 7
## Selection Algorithm: exhaustive
##          RecentAvgPoints FGA FG  PlusMinus AST ORB DRB
## 1  ( 1 ) " "             " " "*" " "       " " " " " "
## 2  ( 1 ) "*"             " " "*" " "       " " " " " "
## 3  ( 1 ) "*"             "*" "*" " "       " " " " " "
## 4  ( 1 ) "*"             "*" "*" "*"       " " " " " "
## 5  ( 1 ) "*"             "*" "*" "*"       "*" " " " "
## 6  ( 1 ) "*"             "*" "*" "*"       "*" " " "*"
## 7  ( 1 ) "*"             "*" "*" "*"       "*" "*" "*"
```

```r
# Run Boruta
          boruta_output <- Boruta(PTS ~ "RecentAvgPoints" + "AvgPointsVsClippers" + "FGA" + "FG" + "Plus
```

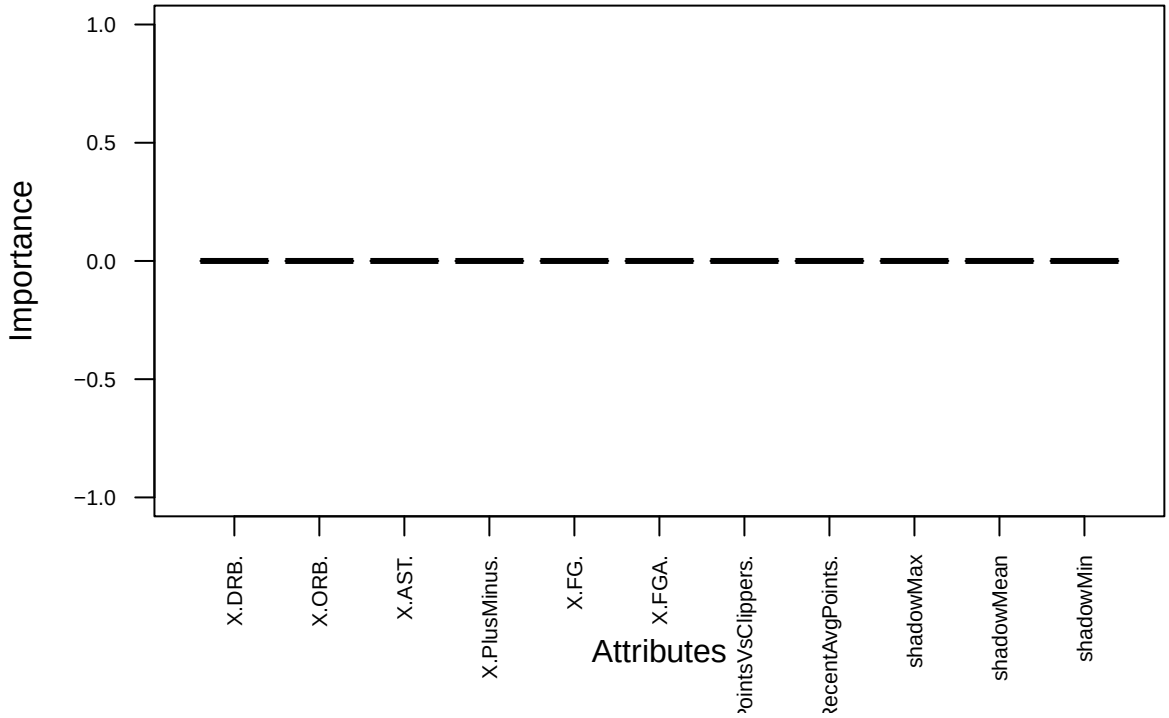
```
##  1. run of importance source...
##  2. run of importance source...
##  3. run of importance source...
```

```
##  4. run of importance source...
##  5. run of importance source...
##  6. run of importance source...
##  7. run of importance source...
##  8. run of importance source...
##  9. run of importance source...
##  10. run of importance source...
## After 10 iterations, +1.1 secs:
##  rejected 8 attributes: X.AST., X.AvgPointsVsClippers., X.DRB., X.FG., X.FGA. and 3 more;
##  no more attributes left.
```

```r
# Print the results
print(boruta_output)
```

```
## Boruta performed 10 iterations in 1.103866 secs.
##  No attributes deemed important.
##  8 attributes confirmed unimportant: X.AST., X.AvgPointsVsClippers.,
## X.DRB., X.FG., X.FGA. and 3 more;
```

```r
# Plot the results
plot(boruta_output, cex.axis=.7, las=2)
```



```
          # Based on these results we can see Recent AveragePoints, FG, and FGA are the only notable va
```

```
# Load required library
library(Boruta)

# Run Boruta Feature Selection
boruta_output <- Boruta(
  PTS ~ RecentAvgPoints + "AvgPointsVsClippers" + FGA + FG + PlusMinus + AST + ORB + DRB,
  data = Lebron_James_All_Regular_Season_Stats,
  doTrace = 2
)
```

```
##  1. run of importance source...
##  2. run of importance source...
##  3. run of importance source...
##  4. run of importance source...
##  5. run of importance source...
##  6. run of importance source...
##  7. run of importance source...
##  8. run of importance source...
##  9. run of importance source...
##  10. run of importance source...
## After 10 iterations, +5 secs:
##  confirmed 4 attributes: FG, FGA, PlusMinus, RecentAvgPoints;
##  rejected 2 attributes: ORB, X.AvgPointsVsClippers.;
##  still have 2 attributes left.
##  11. run of importance source...
##  12. run of importance source...
##  13. run of importance source...
##  14. run of importance source...
##  15. run of importance source...
##  16. run of importance source...
##  17. run of importance source...
##  18. run of importance source...
##  19. run of importance source...
##  20. run of importance source...
##  21. run of importance source...
## After 21 iterations, +9.9 secs:
##  rejected 1 attribute: DRB;
##  still have 1 attribute left.
```

```
##  22. run of importance source...
##  23. run of importance source...
##  24. run of importance source...
##  25. run of importance source...
##  26. run of importance source...
##  27. run of importance source...
##  28. run of importance source...
##  29. run of importance source...
##  30. run of importance source...
##  31. run of importance source...
##  32. run of importance source...
##  33. run of importance source...
##  34. run of importance source...
##  35. run of importance source...
##  36. run of importance source...
##  37. run of importance source...
##  38. run of importance source...
##  39. run of importance source...
##  40. run of importance source...
##  41. run of importance source...
##  42. run of importance source...
##  43. run of importance source...
##  44. run of importance source...
##  45. run of importance source...
##  46. run of importance source...
##  47. run of importance source...
##  48. run of importance source...
##  49. run of importance source...
##  50. run of importance source...
##  51. run of importance source...
##  52. run of importance source...
##  53. run of importance source...
##  54. run of importance source...
##  55. run of importance source...
##  56. run of importance source...
##  57. run of importance source...
```

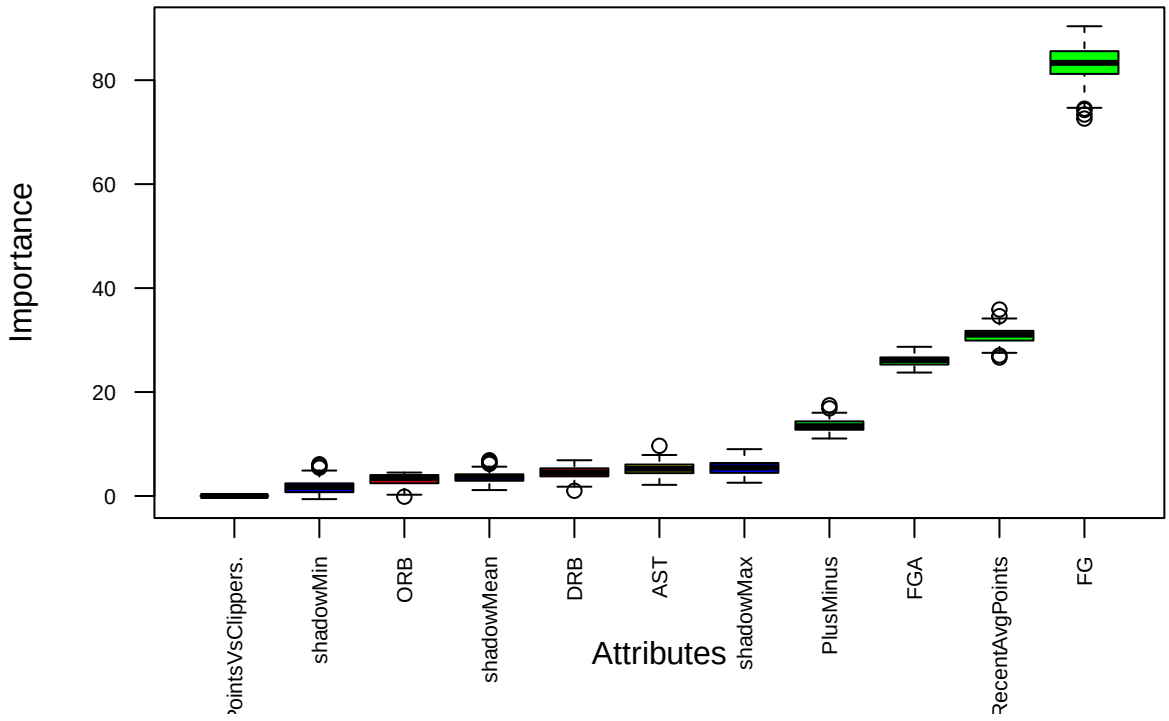
```
##  58. run of importance source...
##  59. run of importance source...
##  60. run of importance source...
##  61. run of importance source...
##  62. run of importance source...
##  63. run of importance source...
##  64. run of importance source...
##  65. run of importance source...
##  66. run of importance source...
##  67. run of importance source...
##  68. run of importance source...
##  69. run of importance source...
##  70. run of importance source...
##  71. run of importance source...
##  72. run of importance source...
##  73. run of importance source...
##  74. run of importance source...
##  75. run of importance source...
##  76. run of importance source...
##  77. run of importance source...
##  78. run of importance source...
##  79. run of importance source...
##  80. run of importance source...
##  81. run of importance source...
##  82. run of importance source...
##  83. run of importance source...
##  84. run of importance source...
##  85. run of importance source...
##  86. run of importance source...
##  87. run of importance source...
##  88. run of importance source...
##  89. run of importance source...
##  90. run of importance source...
##  91. run of importance source...
##  92. run of importance source...
##  93. run of importance source...
```

```
##  94. run of importance source...
##  95. run of importance source...
##  96. run of importance source...
##  97. run of importance source...
##  98. run of importance source...
##  99. run of importance source...
```

```r
# Print Boruta Results
print(boruta_output)
```

```
## Boruta performed 99 iterations in 44.07409 secs.
##  4 attributes confirmed important: FG, FGA, PlusMinus, RecentAvgPoints;
##  3 attributes confirmed unimportant: DRB, ORB, X.AvgPointsVsClippers.;
##  1 tentative attributes left: AST;
```

```r
# Plot Boruta Results
plot(boruta_output, cex.axis = 0.7, las = 2)
```

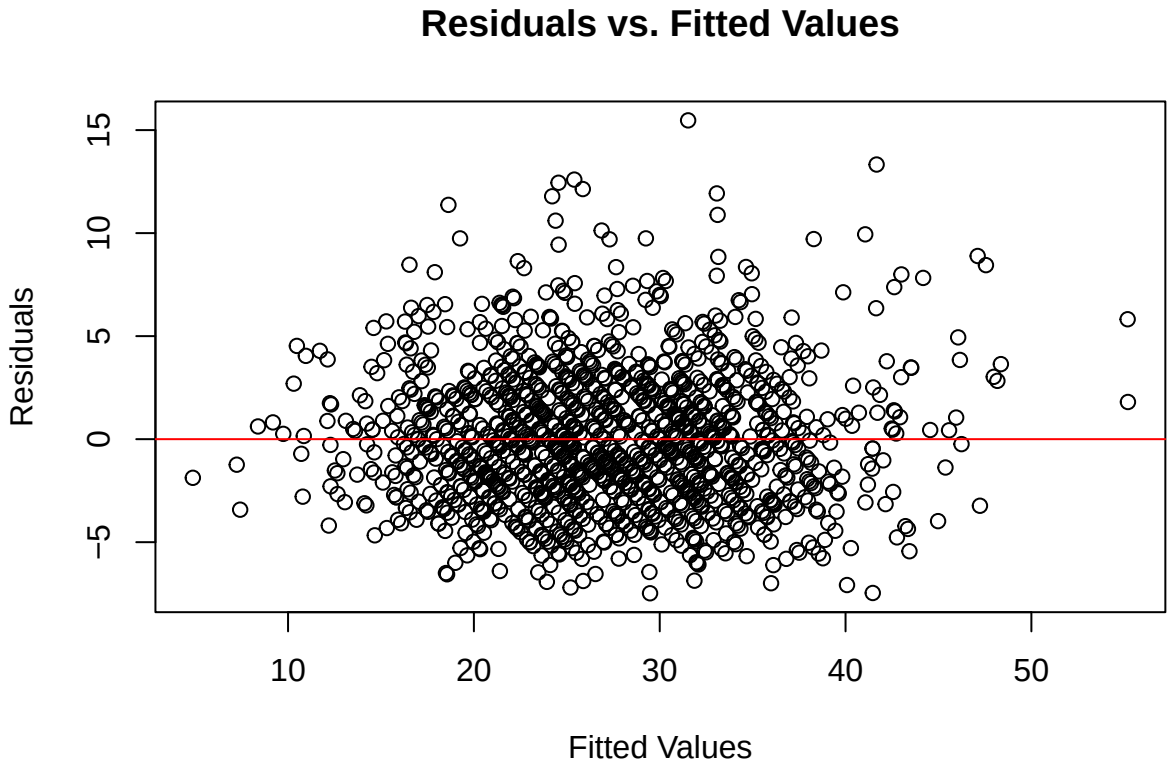


Feature Importance Boxplot:

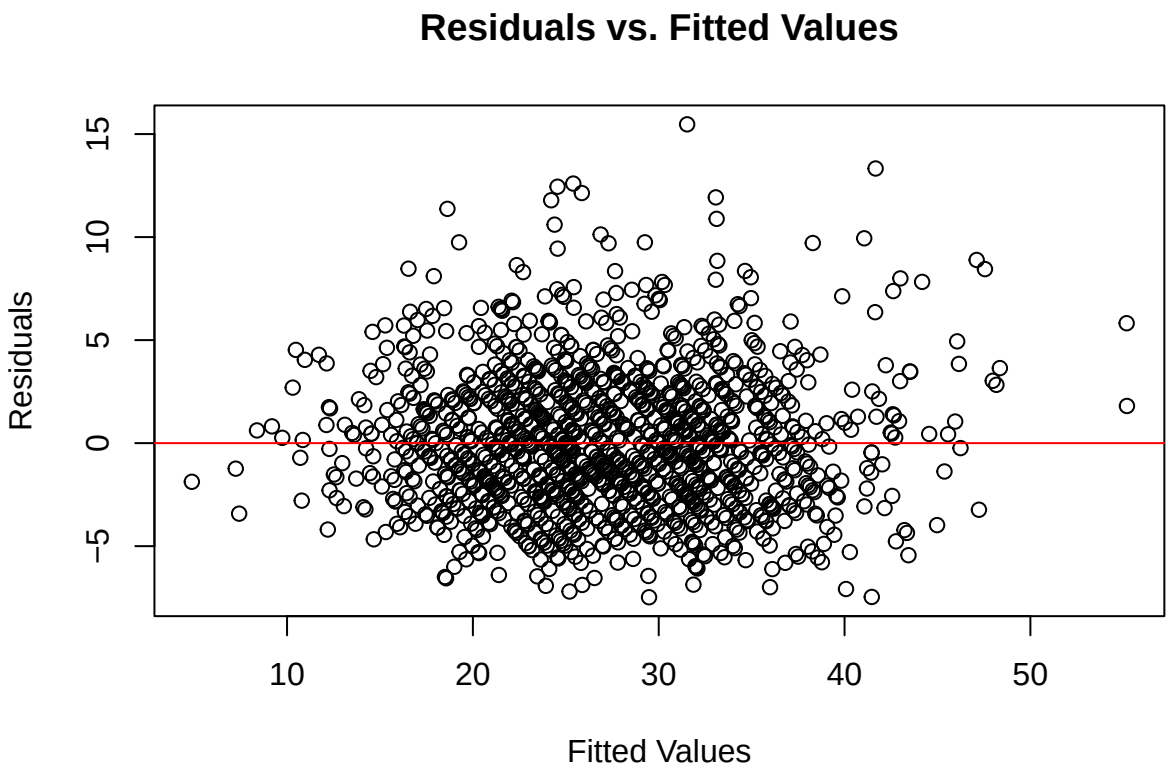
Green Boxes: These represent variables deemed "important" by the Boruta algorithm. In this case, FG is the most critical variable. Black Boxes: These variables are tentative or less important in contributing to the target variable. Red Boxes: Variables labeled as "unimportant" for the prediction task. Key Variables to Keep: Based on the plot:

Keep FG, RecentAvgPoints, and potentially FGA as they show significant importance. Disregard variables like ORB, DRB, and potentially PlusMinus, which have low importance scores.

```r
#6
model <- lm(PTS ~ RecentAvgPoints + FG + FGA, data = Lebron_James_All_Regular_Season_Stats)
# Calculate residuals and fitted values
resid <- residuals(model)
fitted_values <- fitted(model)

# Plot residuals vs. fitted values
plot(fitted_values, resid, xlab = "Fitted Values", ylab = "Residuals", main = "Residuals vs. Fitted Valu
          abline(h = 0, col = "red")  # Add a horizontal line at 0 for reference
```




```r
          #6
          model <- lm(PTS ~ RecentAvgPoints + FG + FGA, data = Lebron_James_All_Regular_Season_Stats)
          # Calculate residuals and fitted values
          resid <- residuals(model)
          fitted_values <- fitted(model)

          # Plot residuals vs. fitted values
          plot(fitted_values, resid, xlab = "Fitted Values", ylab = "Residuals", main = "Residuals vs.
          abline(h = 0, col = "red")  # Add a horizontal line at 0 for reference
```



#6 explanation. In the "Residuals vs. Fitted Values" plot, key aspects to observe are: Pattern/Trend: Residuals should be randomly distributed around the horizontal line at 0. Patterns like curves or funnels suggest non-linearity or heteroscedasticity. Outliers: Points significantly distant from others may indicate outliers, potentially influencing the model significantly. Equal Variance: Residual spread should be consistent across fitted values; variance changes indicate heteroscedasticity. The model's high R-squared value (0.8174) suggests a good fit. However, the diagnostic plot is crucial for validating model assumptions. If residual patterns indicate issues, consider variable transformations, interaction terms, or alternative modeling approaches.

```
#7
library(lmtest)

# My model
model <- lm(PTS ~ RecentAvgPoints + FG + FGA, data = Lebron_James_All_Regular_Season_Stats)

# the RESET test
reset_test_results <- resettest(model, power = 2:3, type = "regressor")

# Print the results
print(reset_test_results)
```

```
##
##  RESET test
##
## data:  model
## RESET = 2.8782, df1 = 6, df2 = 1403, p-value = 0.008616
```

#Notes for 7; The RESET test's p-value of 0.00783, being below the standard 0.05 threshold, signifies

statistical significance, suggesting potential model misspecification. This could stem from omitted variables or an incorrect functional form, indicating that the model may not fully capture the underlying relationship between variables. To address this, consider incorporating additional relevant variables, exploring variable transformations or interaction terms, and employing diagnostic plots for further insights. Additionally, cross-validation can help assess the model's performance on new data. However, it's important to note that while the RESET test indicates misspecification, it does not specify the exact nature of the issue, requiring careful analysis and subject-matter expertise to refine the model effectively.

```
#8
library(lmtest)
bp_test_results <- bptest(model)

print(bp_test_results)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  model
## BP = 9.6119, df = 3, p-value = 0.02217
```

```
#since p-value is less than .05 then heteroskadacitity is present

library(sandwich)

# Calculate robust standard errors
robust_se <- coeftest(model, vcov. = vcovHC(model, type = "HC1"))

# Print the results with robust standard errors
print(robust_se)
```

```
##
## t test of coefficients:
##
##                  Estimate Std. Error t value  Pr(>|t|)
## (Intercept)     -2.513469   0.624403 -4.0254 5.991e-05 ***
## RecentAvgPoints  0.300251   0.024735 12.1385 < 2.2e-16 ***
## FG               1.948936   0.044247 44.0466 < 2.2e-16 ***
## FGA              0.108733   0.028302  3.8419 0.0001275 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

#8 Explanation; All predictors are statistically significant at conventional levels (e.g., 0.05, 0.01), suggesting they have a meaningful contribution to explaining the variation in PTS. The signs of the coefficients indicate the direction of the relationship with PTS.

```
#8
library(lmtest)
bp_test_results <- bptest(model)
print(bp_test_results)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  model
## BP = 9.6119, df = 3, p-value = 0.02217
```

```r
#since p-value is less than .05 then heteroskadacitity is present

library(sandwich)

# Calculate robust standard errors
robust_se <- coeftest(model, vcov. = vcovHC(model, type = "HC1"))

# Print the results with robust standard errors
print(robust_se)
```

```
##
## t test of coefficients:
##
##                  Estimate Std. Error t value  Pr(>|t|)
## (Intercept)     -2.513469   0.624403 -4.0254 5.991e-05 ***
## RecentAvgPoints  0.300251   0.024735 12.1385 < 2.2e-16 ***
## FG               1.948936   0.044247 44.0466 < 2.2e-16 ***
## FGA              0.108733   0.028302  3.8419 0.0001275 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

#8 Explanation; All predictors are statistically significant at conventional levels (e.g., 0.05, 0.01), suggesting they have a meaningful contribution to explaining the variation in PTS. The signs of the coefficients indicate the direction of the relationship with PTS.

```r
#9
enhanced_model <- lm(PTS ~ RecentAvgPoints * FG + I(FGA^2) + PlusMinus + AST + ORB + DRB, data
aic_enhanced <- AIC(enhanced_model)
bic_enhanced <- BIC(enhanced_model)

# Print AIC and BIC for comparison
print(aic_enhanced)
```

```
## [1] 7424.44
```

```r
print(bic_enhanced)
```

```
## [1] 7476.974
```

These values reflect the trade-off between: Goodness of fit: How well the model explains the variation in PTS. Model complexity: How many predictors (e.g., RecentAvgPoints, FG, FGA) are included. Indication: The higher these values, the worse the model is relative to alternative models. These values alone don't mean much—compare them with other models' AIC/BIC to make an informed decision.