


NBA GOAT DEBATE DATA SCIENCE PROJECT

Nick Haller

- 
- This project involved a comprehensive analysis of LeBron James and Michael Jordan's statistics throughout their careers.
 - I looked at regular box score stats, playoff stats, and advanced stats, and compared the two players with a weighted scoring system, linear regression, and logistic regression.
 - Studying data science and mathematics at UW-Madison, set to graduate in 2027

MOTIVATION

- These are widely known as the two best basketball players of all time. This debate is a common topic of conversation when discussing basketball with others. I knew this debate was close, but I wanted to determine how close it was with pure numbers.
- This project features advanced stat comparisons throughout each of their respective careers
- regression models for each player that predict regular season wins in a season based on a few statistics
- Logistic models that predict whether their season has a 60% win percentage based on a few statistics
- A weighted scoring system for the regular season and playoffs with surprising results

STAGE 1: DATA COLLECTION AND CLEANING

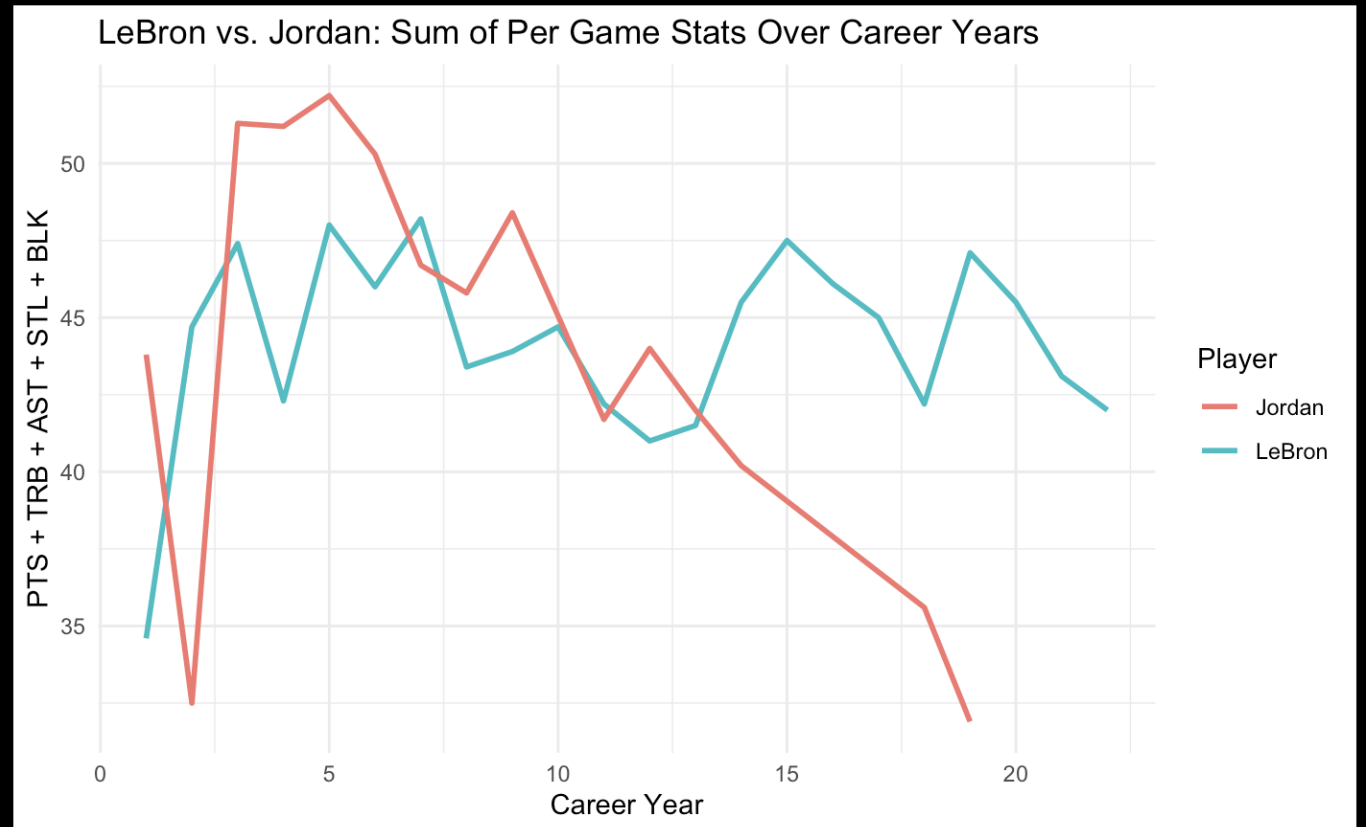
- I collected, from Basketball Reference, every summary of per-game stats, total stats, shooting stats, adjusted shooting stats, advanced stats, per-36-minute stats, and per-100-possession stats for the regular season and playoffs for both players.
- I used Pandas and NumPy packages in Python to address missing cells, drop unnecessary columns, manually add information, change to correct data types (int, str, float, bool), and rename headers to make them clearer
- I combined all the cleaned datasets into a master dataset. One for the regular season and playoffs of both players (4 datasets)

STAGE 2: VISUALIZATION

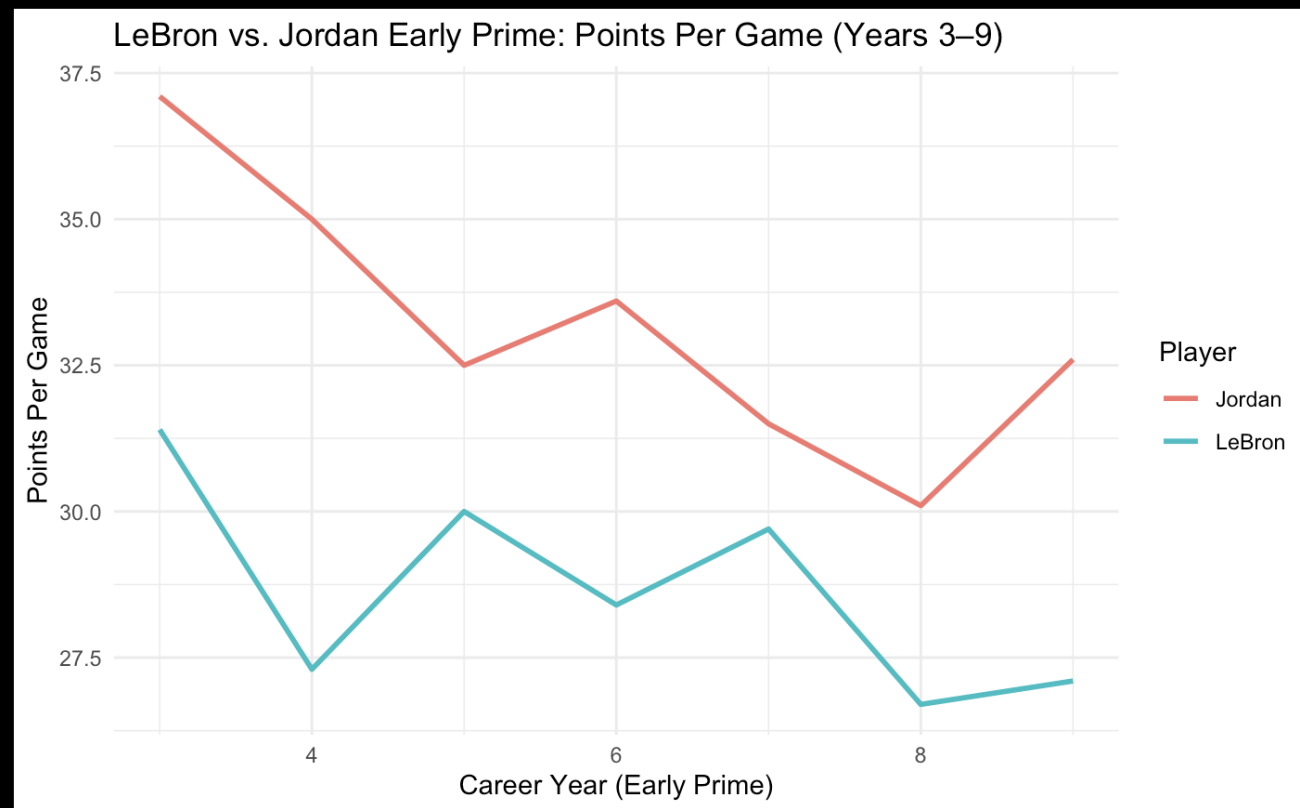
I am going to show my initial visualizations and draw conclusions from each graph

This graph measures points + rebounds + assists + steals + blocks for Jordan and LeBron

Jordan had a better prime based on these stats, but LeBron was more consistent and lasted longer



Jordan wins in raw scoring production for their "early primes". I am calling it early prime because the early part of LeBron's prime matches up with the main prime of Jordan.



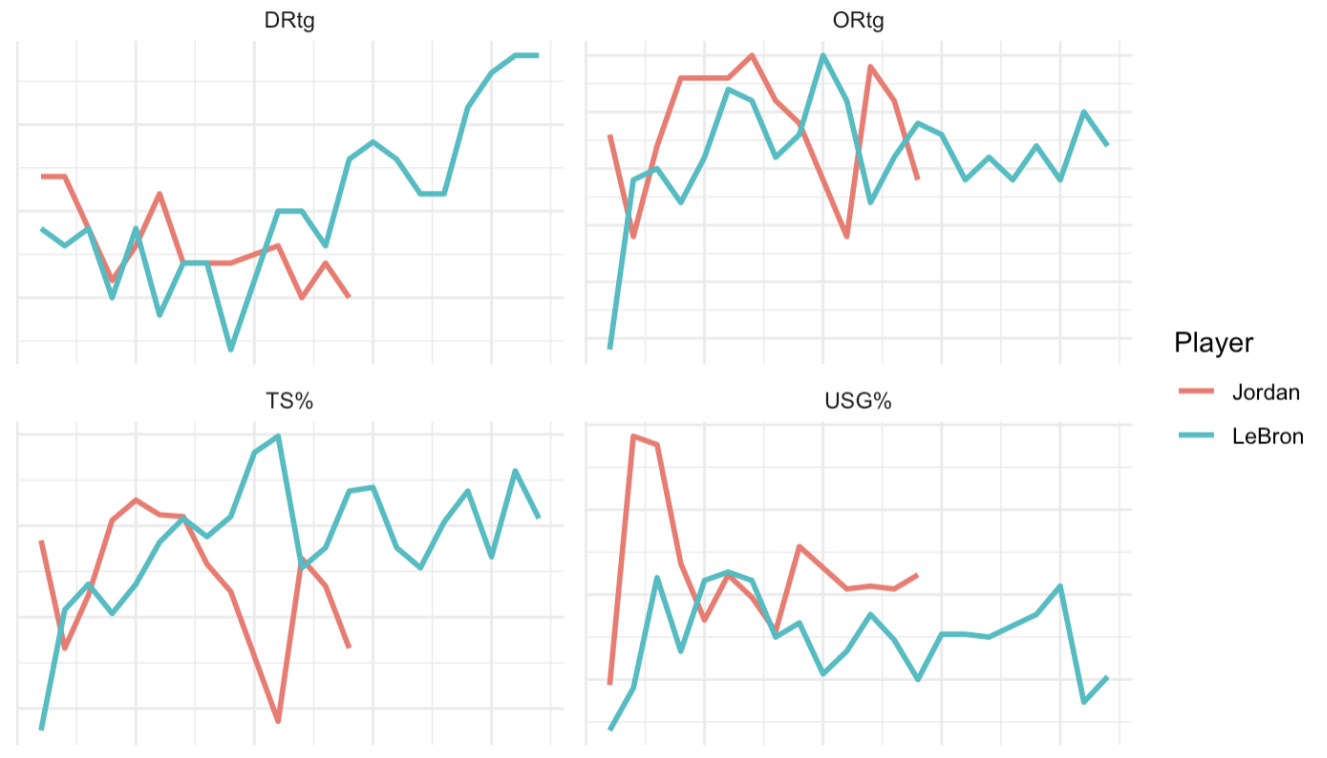
USG%: estimate of teams plays used by the player while they were on the floor. Jordan clears

TS%: measures shooting efficiency while taking in 2P,3P, and FT. LeBron clears

DRtg: estimate of points allowed per 100 possessions. Hard to decide here. Jordan was relatively better during his time, but LeBron's improvement in defense with age is very impressive

ORtg: estimate of points produced per 100 possessions. Hard to decide again. Jordan was relatively better during his early prime but less consistent, LeBron has a high peak just like Jordan and also stays steady, with his "washed years" on par with Jordan's late 90s runs in this statistic

LeBron vs. Jordan: Stat Comparisons Over Career



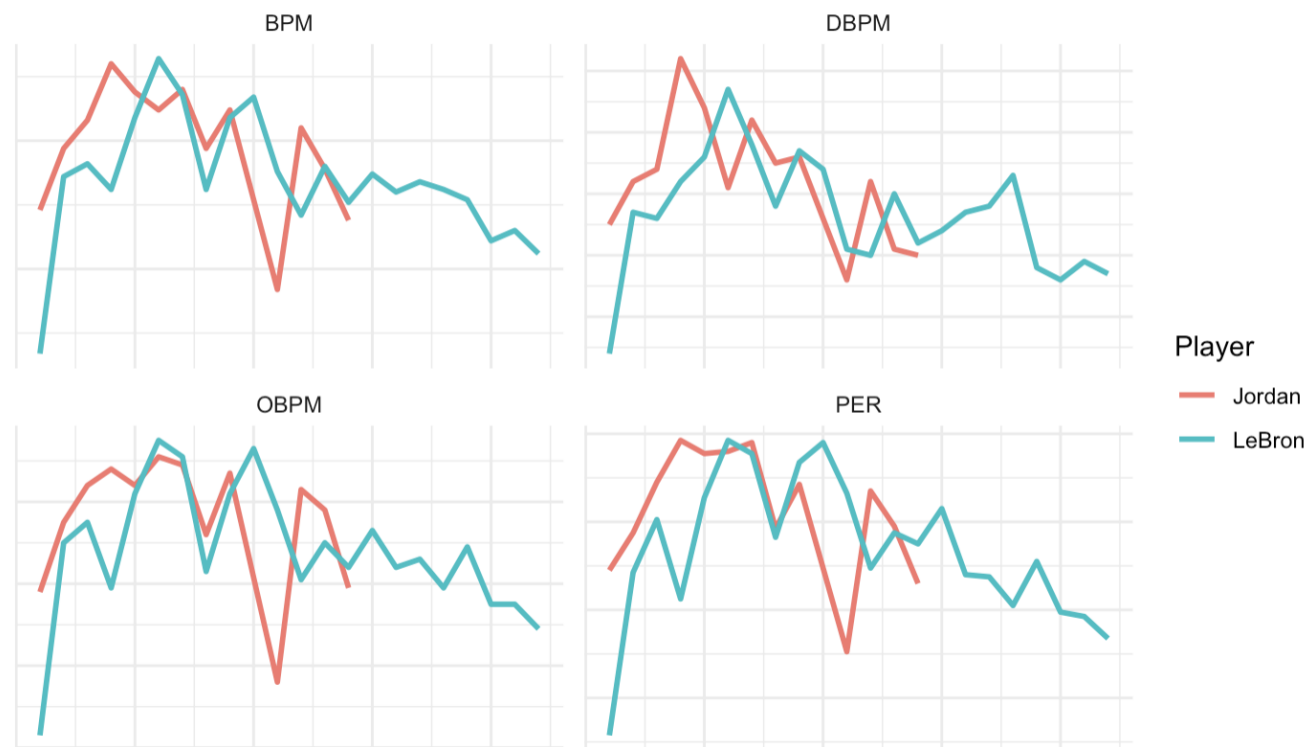
BPM: box score estimate of the points per 100 possessions a player contributed above a league average player, translated to an average team. Jordan rose to his prime a lot quicker than Lebron in this statistic, but Lebron had him here, with a high peak and slightly hovering over Jordan, along with longevity

DBPM: box score estimate of the defensive points per 100 possessions a player contributed above a league average player. Very hard to differentiate here. If I had to choose, I would say Jordan

OBPM: box score estimate of the offensive points per 100 possessions a player contributed above a league average player. Lebron had higher peaks and longevity, so he wins here

PER: player efficiency rating - per minute production standardized to the league average of 15. Lebron clears. Since the Bulls dynasty started and throughout, Lebron's PER is better than Jordan's, along with longevity

LeBron vs. Jordan: Stat Comparisons Over Career

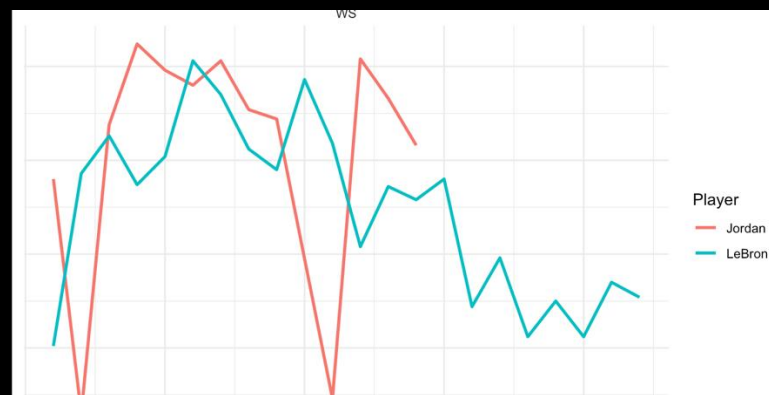




VORP: Value over Replacement Player

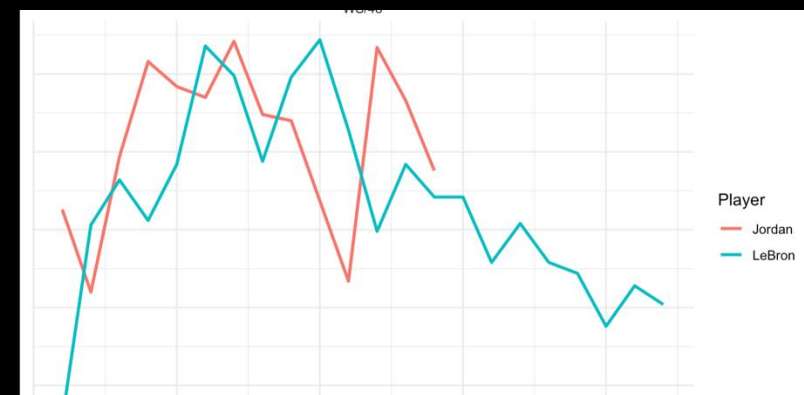
A box score estimate of the points per 100 TEAM possessions that a player contributed above a replacement-level (-2.0) player, translated to an average team and prorated to an 82-game season.

Jordan wins here. Higher peak and, during his prime, relatively stayed above LeBron



WS: estimate of number of wins contributed by a player

Toss up here. Jordan is inconsistent in this metric, but we cannot ignore his peaks. Jordan wins



WS/48: Estimate of win shares by a player per 48 minutes

Another toss-up, but LeBron wins here. He competes with Jordan in terms of peaks and has a better valley. His longevity gives him the slight edge

I attempted to minimize bias in developing this weighted scoring system. I used a combination of regular stats and advanced stats during the regular season. The winner of each stat was awarded points

Results:

Lebron – 22

Jordan – 24

```
```{r}
lebron_prime_numeric <- lebron_prime_only %>%
 mutate(across(everything(), ~ as.numeric(.)))

jordan_prime_numeric <- jordan_prime_only %>%
 mutate(across(everything(), ~ as.numeric(.)))

lebron_prime_avg <- colMeans(lebron_prime_numeric, na.rm = TRUE)
jordan_prime_avg <- colMeans(jordan_prime_numeric, na.rm = TRUE)

lebron_avg_row <- as.data.frame(t(lebron_prime_avg))
jordan_avg_row <- as.data.frame(t(jordan_prime_avg))

box_stat_pairs <- list(
 "PTS" = "PTS-pergame",
 "AST" = "AST-pergame",
 "TRB" = "TRB-pergame",
 "STL" = "STL-pergame",
 "BLK" = "BLK-pergame",
 "FG%" = "FG%",
 "eFG%" = "eFG%",
 "TS%" = "TS%",
 "TOV" = "TOV"
)

adv_stat_pairs <- list(
 "ORtg" = "ORtg",
 "DRtg" = "DRtg",
 "PER" = "PER",

 "WS" = "WS",

 "OBPM" = "OBPM",
 "DBPM" = "DBPM",
 "BPM" = "BPM",
 "VORP" = "VORP",
)

lebron_score <- 0
jordan_score <- 0
```

```
for (pair in names(box_stat_pairs)) {
 lebron_col <- pair
 jordan_col <- box_stat_pairs[[pair]]

 if (lebron_col %in% names(lebron_avg_row) && jordan_col %in% names(jordan_avg_row)) {
 lebron_val <- as.numeric(lebron_avg_row[[lebron_col]])
 jordan_val <- as.numeric(jordan_avg_row[[jordan_col]])

 if (all(!is.na(c(lebron_val, jordan_val)))) {
 if (pair == "TOV") {
 if (lebron_val < jordan_val) {
 lebron_score <- lebron_score + 4
 } else if (jordan_val < lebron_val) {
 jordan_score <- jordan_score + 4
 }
 } else {
 if (lebron_val > jordan_val) {
 lebron_score <- lebron_score + 4
 } else if (jordan_val > lebron_val) {
 jordan_score <- jordan_score + 4
 }
 }
 }
 }
}

for (pair in names(adv_stat_pairs)) {
 lebron_col <- pair
 jordan_col <- adv_stat_pairs[[pair]]

 if (lebron_col %in% names(lebron_avg_row) && jordan_col %in% names(jordan_avg_row)) {
 lebron_val <- as.numeric(lebron_avg_row[[lebron_col]])
 jordan_val <- as.numeric(jordan_avg_row[[jordan_col]])

 if (all(!is.na(c(lebron_val, jordan_val)))) {
 if (lebron_val > jordan_val) {
 lebron_score <- lebron_score + 3
 } else if (jordan_val > lebron_val) {
 jordan_score <- jordan_score + 3
 }
 }
 }
}
}
```

I used the same system to make a weighted scoring system, this time for the playoffs

Results:

Lebron – 37

Jordan - 23

```
```{r}
lebron_numeric <- lebron_playoffs_master %>%
  slice(-(n() - 3):n()) %>%
  mutate(across(everything(), ~ as.numeric(.)))

jordan_numeric <- jordan_master_playoff_stats %>%
  slice(-c(1, 2, 3)) %>%
  mutate(across(everything(), ~ as.numeric(.)))

lebron_avg <- colMeans(lebron_numeric, na.rm = TRUE)
jordan_avg <- colMeans(jordan_numeric, na.rm = TRUE)

lebron_avg_row <- as.data.frame(t(lebron_avg))
jordan_avg_row <- as.data.frame(t(jordan_avg))

box_stat_pairs <- list(
  "PTS" = "PTS_pergame",
  "AST" = "AST_pergame",
  "TRB" = "TRB_pergame",
  "STL" = "STL_pergame",
  "BLK" = "BLK_pergame",
  "FG%" = "FG%",
  "eFG%" = "eFG%",
  "TS%" = "TS%",
  "TOV" = "TOV"
)

adv_stat_pairs <- list(
  "ORTg" = "ORTg",
  "DRTg" = "DRTg",
  "PER" = "PER",
  "WS" = "WS",
  "OBPM" = "OBPM",
  "DBPM" = "DBPM",
  "BPM" = "BPM",
  "VORP" = "VORP"
)

lebron_score <- 0
jordan_score <- 0
```

```
for (pair in names(box_stat_pairs)) {
  lebron_col <- pair
  jordan_col <- box_stat_pairs[[pair]]

  if (lebron_col %in% names(lebron_avg_row) & jordan_col %in% names(jordan_avg_row)) {
    lebron_val <- as.numeric(lebron_avg_row[[lebron_col]])
    jordan_val <- as.numeric(jordan_avg_row[[jordan_col]])

    if (all(!is.na(c(lebron_val, jordan_val)))) {
      if (pair == "TOV") {
        if (lebron_val < jordan_val) {
          lebron_score <- lebron_score + 4
        } else if (jordan_val < lebron_val) {
          jordan_score <- jordan_score + 4
        }
      } else {
        if (lebron_val > jordan_val) {
          lebron_score <- lebron_score + 4
        } else if (jordan_val > lebron_val) {
          jordan_score <- jordan_score + 4
        }
      }
    }
  }
}

for (pair in names(adv_stat_pairs)) {
  lebron_col <- pair
  jordan_col <- adv_stat_pairs[[pair]]

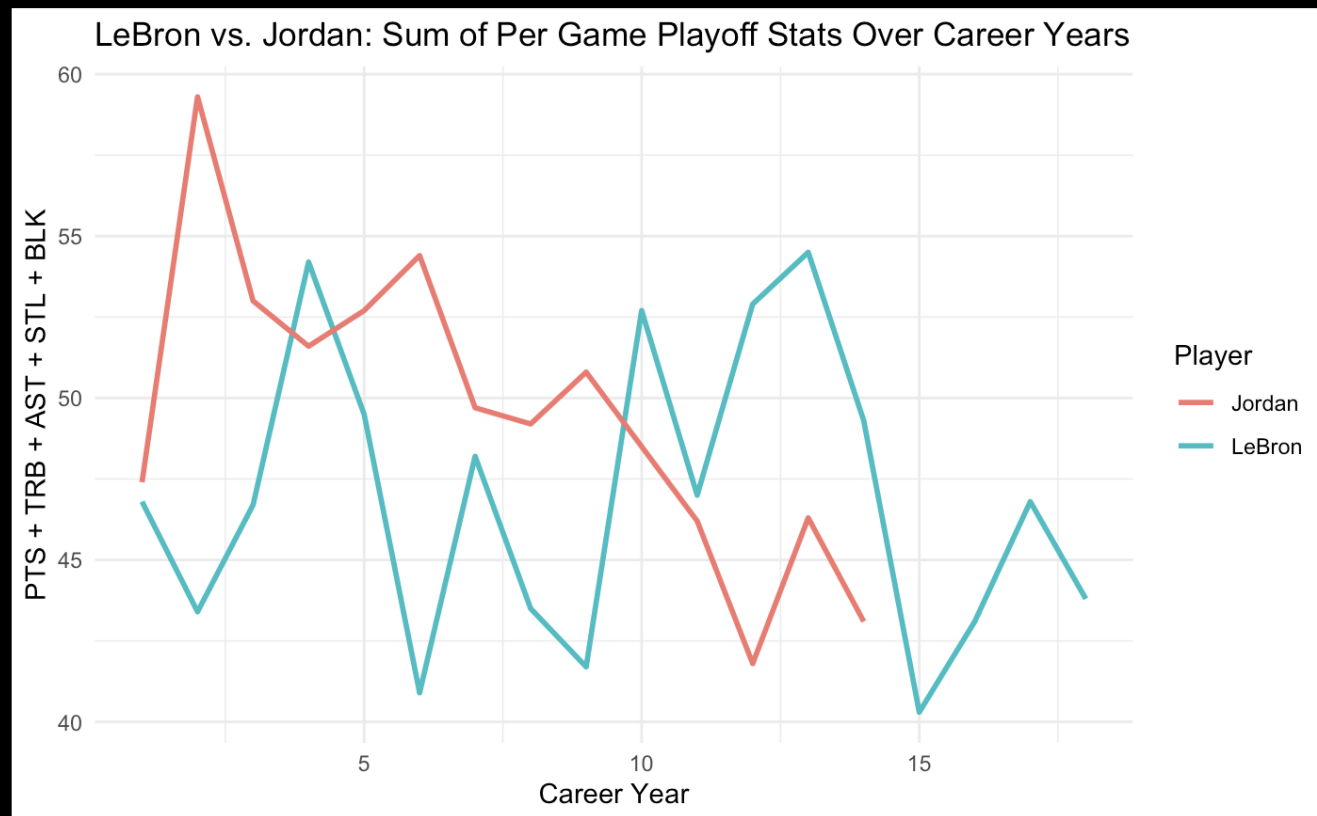
  if (lebron_col %in% names(lebron_avg_row) & jordan_col %in% names(jordan_avg_row)) {
    lebron_val <- as.numeric(lebron_avg_row[[lebron_col]])
    jordan_val <- as.numeric(jordan_avg_row[[jordan_col]])

    if (all(!is.na(c(lebron_val, jordan_val)))) {
      if (lebron_val > jordan_val) {
        lebron_score <- lebron_score + 3
      } else if (jordan_val > lebron_val) {
        jordan_score <- jordan_score + 3
      }
    }
  }
}
}
```

STAGE 3 PART 2: PLAYOFF VISUALIZATION

This is a graph of points, rebounds, assists, steals, and blocks averages added up for each playoff run for the two players

Lebron was only better in this metric for one season until the latter stages of the Bulls dynasty. Lebron's longevity is impressive here, but Jordan wins overall

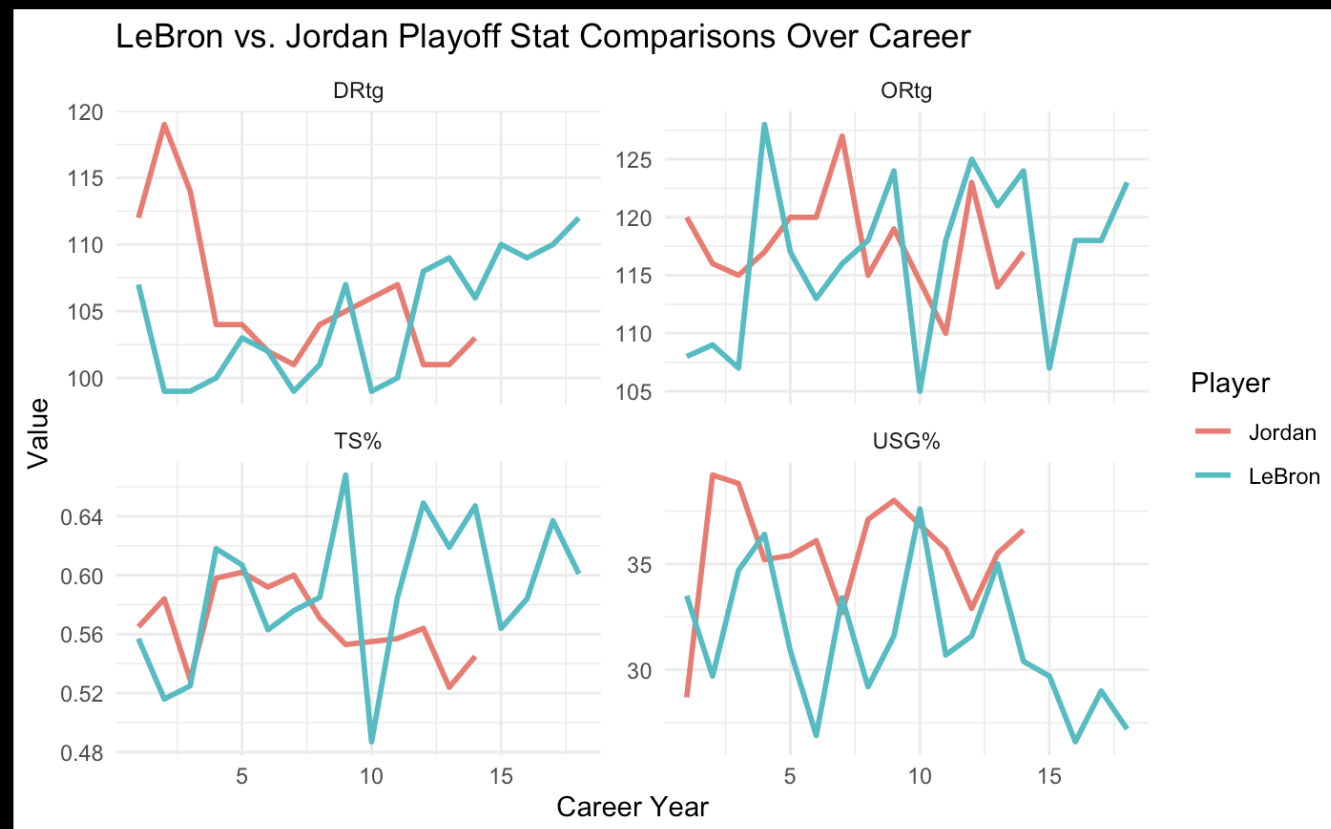


DRtg: Jordan wins here, but Lebron's improvement with age is impressive

ORtg: Lebron has a higher peak and a lower valley, though. And Lebron's longevity, coupled with his ability to stay with Jordan during the Bulls dynasty, gives Lebron the edge

TS%: Lebron clears here

USG%: Jordan clears here

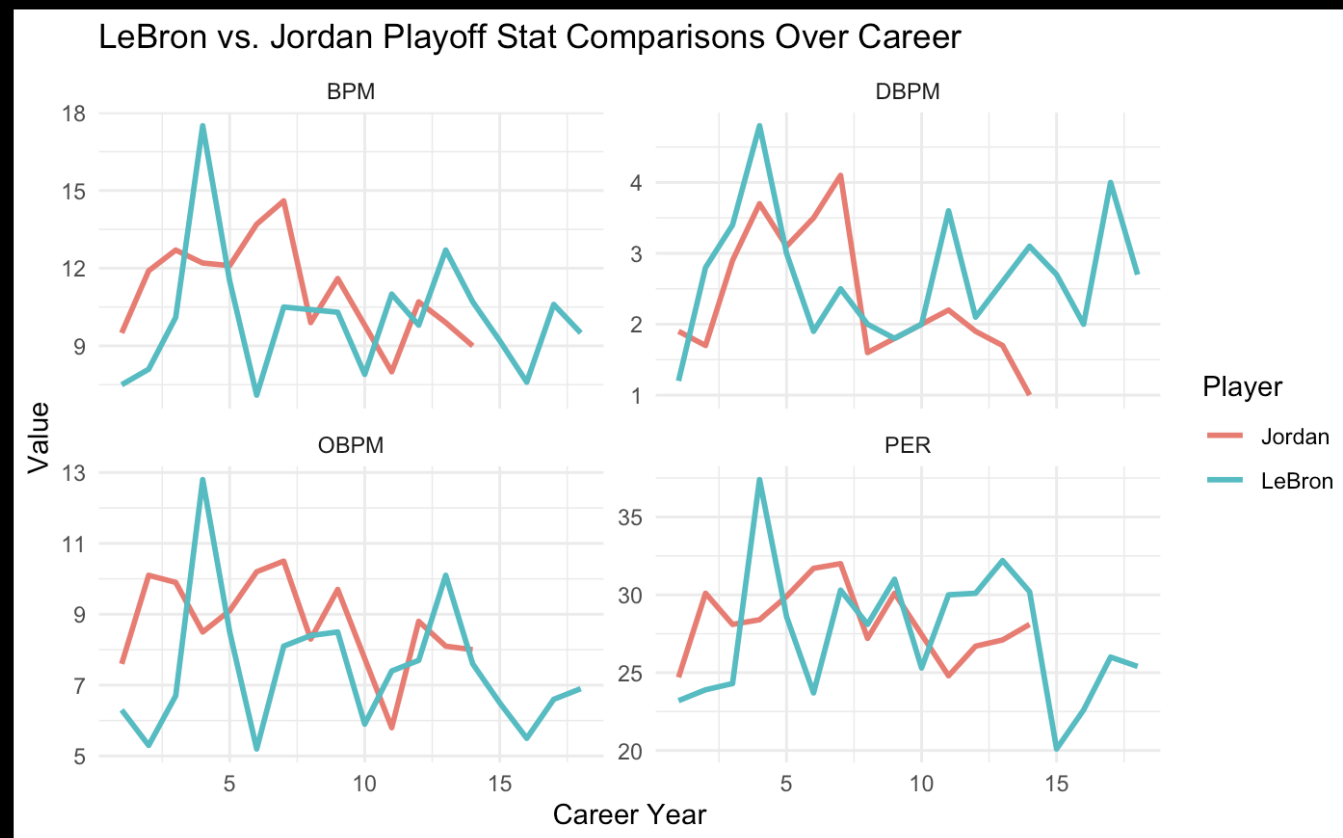


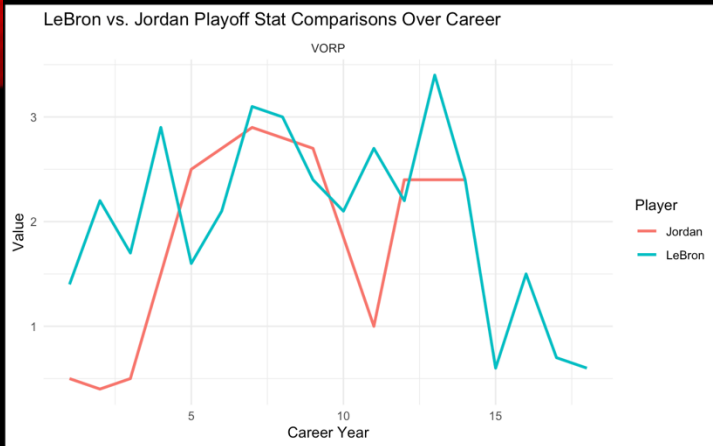
BPM: Jordan overall had a better BPM during his prime, matching up with the points in LeBron's career, but LeBron had the best peak and longevity. Still goes to Jordan, though

DBPM: LeBron clears

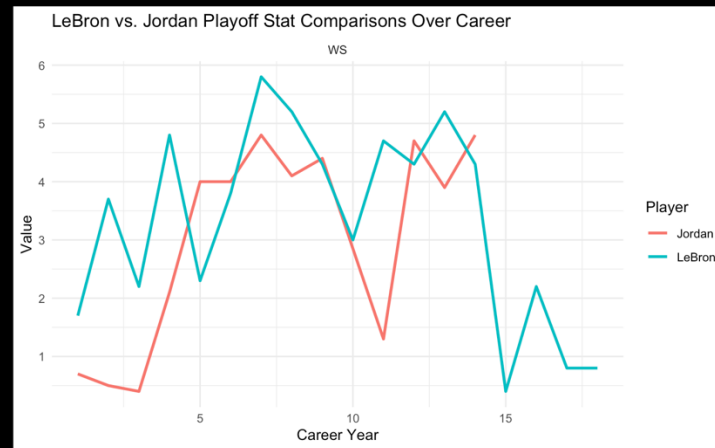
OBPM: Jordan wins here for being more consistent, even though LeBron's peak was the highest

PER: LeBron wins here. The highest two peaks, longevity, LeBron's second stint with the Cavs looks to have a better PER than the late 90s dynasty Bulls

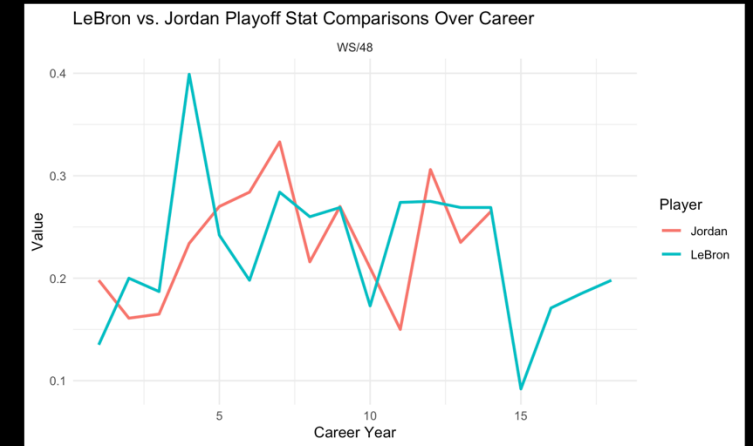




VORP: LeBron clears. Surprisingly, LeBron dominated the first few playoff runs compared to Jordan in this metric, opposing earlier metric. Also, the highest peak and consistency



WS: LeBron clears



WS/48: closer than the WS argument, but LeBron wins here again, with the highest peak, longevity, and stays with Jordan throughout the center of the graph

STAGE 4: HYPOTHESIS TESTING

- I used a permutation testing procedure to perform a hypothesis test on whether the difference between Jordan's prime average OBPM is statistically significantly better than LeBron's prime average OBPM.
- Motivation: Jordan is known as the better offensive player, but these statistics are very close
- We randomly shuffled all the prime OBPM values between LeBron and Jordan thousands of times to determine how often a difference as large as the real one could occur by chance. This helped us test whether Jordan's slightly higher OBPM was meaningful or just random variation.
- LeBron's Prime Average OBPM: 7.317647
- Jordan's Prime Average OBPM: 7.736364
- Null hypothesis: Jordan's mean OBPM is less than or equal to LeBron's mean OBPM
- Alternative hypothesis: Jordan's mean OBPM is greater than LeBron's mean OBPM

```

```{r}
set.seed(123)

lebron_obpm <- as.numeric(lebron_prime_only$OBPM)
jordan_obpm <- as.numeric(jordan_prime_only$OBPM)

lebron_obpm <- lebron_obpm[!is.na(lebron_obpm)]
jordan_obpm <- jordan_obpm[!is.na(jordan_obpm)]

obs_diff <- mean(jordan_obpm) - mean(lebron_obpm)

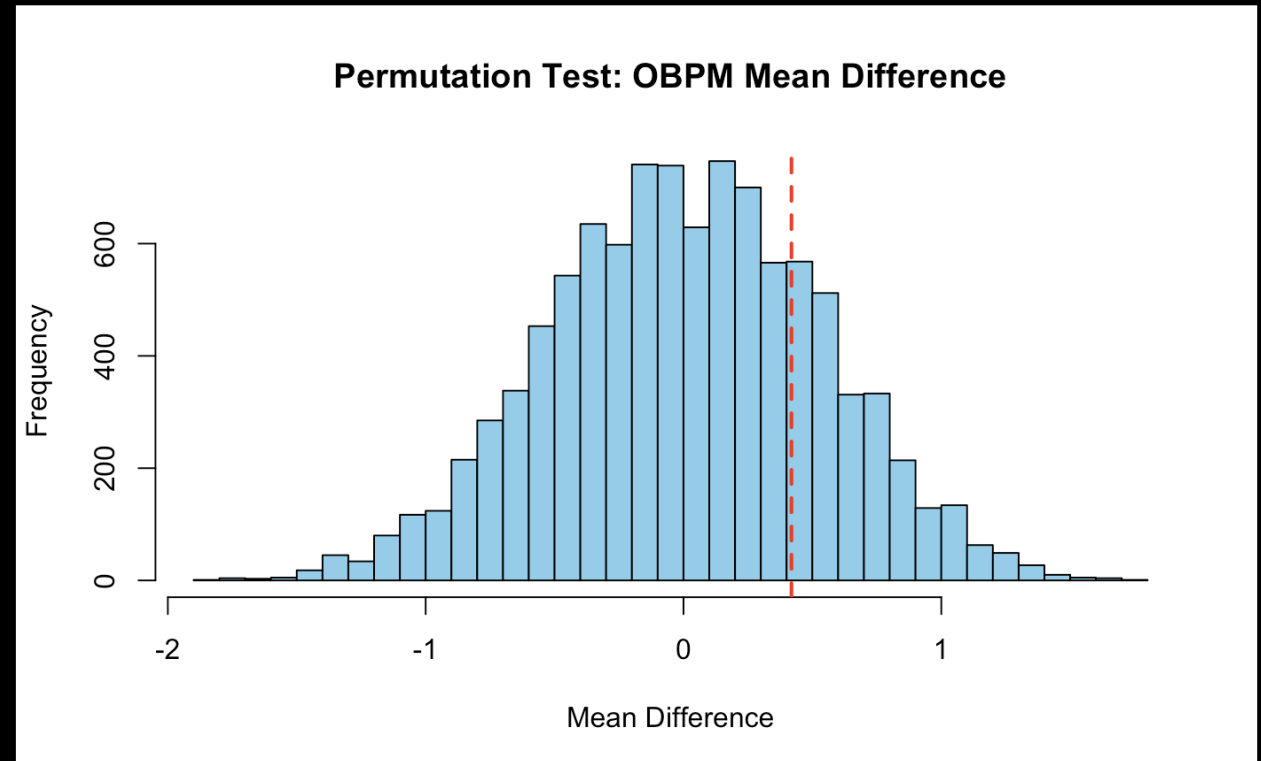
combined <- c(jordan_obpm, lebron_obpm)
n_jordan <- length(jordan_obpm)
n_perm <- 10000
perm_diffs <- numeric(n_perm)

for (i in 1:n_perm) {
 shuffled <- sample(combined)
 jordan_sample <- shuffled[1:n_jordan]
 lebron_sample <- shuffled[(n_jordan + 1):length(shuffled)]

 perm_diffs[i] <- mean(jordan_sample) - mean(lebron_sample)
}

p_value <- mean(perm_diffs >= obs_diff)

```



the observed difference is 0.4187  
The p-value is 0.2287.

Conclusion: With a limited sample size, different eras, and enough natural year-to-year variation, there is no evidence to reject the null hypothesis. The p-value is not statistically significant enough to claim that Jordan's average OBPM in his prime was higher than LeBron's.

# STAGE 5: LINEAR REGRESSION (LEBRON)

- In this stage:
- Made a linear regression model to predict the number of wins LeBron would have in a season based on a few statistics
- I used the forward stepwise AIC selection technique from machine learning, which adds predictors to the model one by one based on which predictor improves the model the best.
- As soon as adding nothing to the model would be better than adding any predictors, we stop.
- I analyze the diagnostic plots to make sure regression assumptions are satisfied
- I used Lasso regularization to decrease the strength of some predictors if needed to perfect the model fit (prevent overfitting)

This is the final linear model.

All three predictors (WS/48, STL, TRB) are statistically significant.

The F-statistic's significance indicates that there is evidence that at least one predictor is significant.

$R^2$  is the percentage of variation in responses (Wins) that is explained by the predictors. At 75%, this is high for only three predictors.

The residual standard error (RSE) is around 5, which is pretty good in this context.

```
```{r}
lebron_stepwise_model3 <- lm(formula = Wins ~ `WS/48` + STL + TRB, data = lebron_regular_model_master)

summary(lebron_stepwise_model3)
```

Call:
lm(formula = Wins ~ `WS/48` + STL + TRB, data = lebron_regular_model_master)

Residuals:
 Min 1Q Median 3Q Max
-11.0271 -2.5781 0.9465 3.1654 5.7592

Coefficients:
 Estimate Std. Error t value Pr(>|t|)
(Intercept) 75.428 15.216 4.957 0.000102 ***
`WS/48` 168.340 23.637 7.122 1.23e-06 ***
STL -15.366 4.861 -3.161 0.005408 **
TRB -5.290 1.692 -3.127 0.005821 **

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 4.998 on 18 degrees of freedom
Multiple R-squared: 0.7484, Adjusted R-squared: 0.7065
F-statistic: 17.85 on 3 and 18 DF, p-value: 1.252e-05
```

# CROSS VALIDATION AND REGULARIZATION

The images include code for calculating the cross-validation error of our model and the Lasso regularization technique used to improve our model by shrinking our predictors' coefficients (slope)

I used cross-validation to test the model's predictive accuracy on new data by evaluating its performance on unseen examples. Then, I applied a technique called Lasso to simplify the model and reduce overfitting by shrinking the impact of less important stats.

cross validation using the Leave Out One Cross Validation L00CV method:

```
```{r}
formula <- Wins ~ `WS/48` + STL + TRB

n <- nrow(lebron_regular_model_master)

cv_results <- cv.glm(lebron_regular_model_master, glm(formula, data = lebron_regular_model_master), K = n)

cat("L00CV error:", cv_results$delta[1], "\n")
```
```

L00CV error: 32.92033

```
```{r}
lasso_cv <- cv.glmnet(X, y, alpha = 1)

best_lambda_lasso <- lasso_cv$lambda.min
cat("Best Lasso lambda:", best_lambda_lasso, "\n")

lasso_coefs <- coef(lasso_cv, s = best_lambda_lasso)
print(lasso_coefs)
```
```

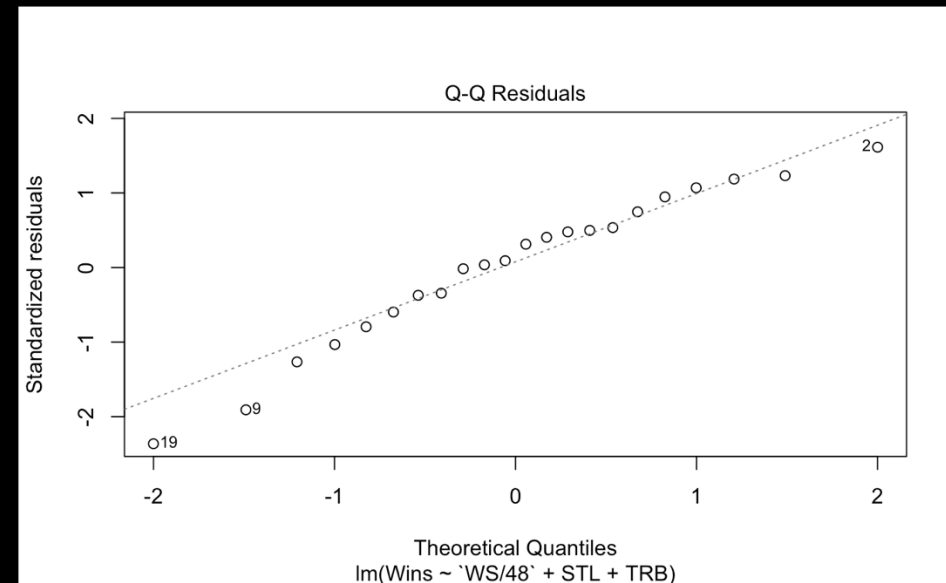
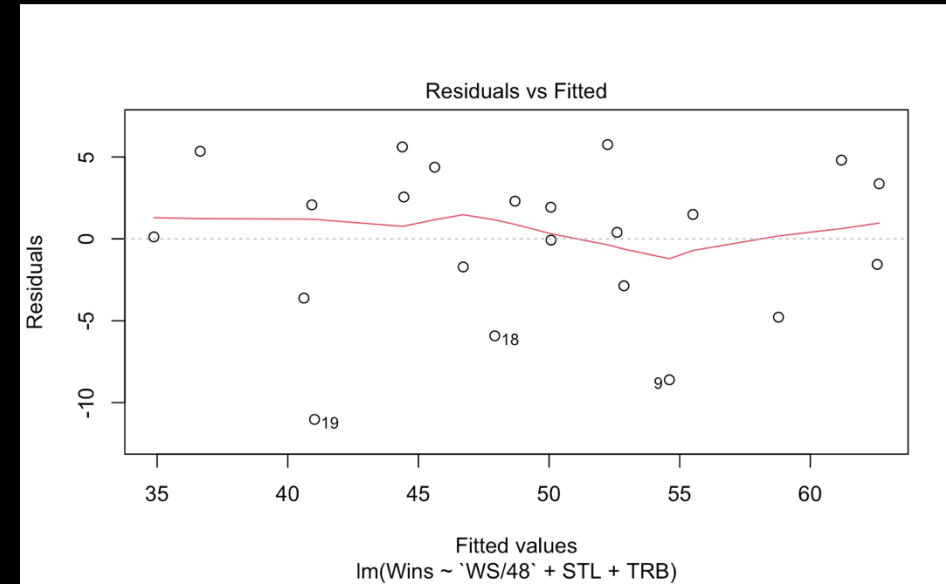
```
```{r}
cat("Ridge CV MSE:", min(ridge_cv$cvm), "\n")
cat("Lasso CV MSE:", min(lasso_cv$cvm), "\n")
```
```

Ridge CV MSE: 32.57125  
Lasso CV MSE: 29.37851



# DIAGNOSTIC PLOTS

These plots verify whether our model satisfies the assumptions of linear regression. The residuals appear evenly spread (constant variance), centered around zero (linearity), and roughly follow a normal distribution (normality), though the Q-Q plot tails deviate slightly due to the small sample size.



# STAGE 5: LINEAR REGRESSION (JORDAN)

This is our model summary for the final Jordan linear regression model

We used the same strategy as for LeBron's model to make this one.

The two predictors, WS/48 and steals per game, are both statistically significant

$R^2$  is 89%, which is really good for only 2 predictors. Our predictors explain 89% of the variation in Wins

The F-statistic is significant, which gives evidence that at least one predictor is significant (not a slope of zero)

The Residual Standard Error (RSE) is approximately 4.85, which is quite good.

Call:

```
lm(formula = Wins ~ `WS/48` + STL_pergame, data = jordan_prime_regular_model_master)
```

Residuals:

| Min    | 1Q     | Median | 3Q    | Max   |
|--------|--------|--------|-------|-------|
| -5.788 | -2.906 | -1.125 | 1.751 | 8.279 |

Coefficients:

|             | Estimate | Std. Error | t value | Pr(> t ) |     |
|-------------|----------|------------|---------|----------|-----|
| (Intercept) | 27.827   | 9.718      | 2.864   | 0.018672 | *   |
| `WS/48`     | 268.691  | 33.569     | 8.004   | 2.2e-05  | *** |
| STL_pergame | -18.447  | 3.221      | -5.727  | 0.000285 | *** |

---

Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 4.847 on 9 degrees of freedom

Multiple R-squared: 0.8907, Adjusted R-squared: 0.8664

F-statistic: 36.67 on 2 and 9 DF, p-value: 4.721e-05

# CROSS VALIDATION AND REGULARIZATION

After cross-validation (seeing how well our model fits unseen data), our model ended up being good

Lasso regression ends up improving our model slightly

cross validation:

```
```{r}
formula <- Wins ~ `WS/48` + STL_pergame

cv_results <- cv.glm(jordan_prime_regular_model_master, glm(formula, data = jordan_prime_regular_model_master), K = 12)
cat("L00CV error:", cv_results$delta[1], "\n")
```
```

L00CV error: 26.72532

ridge regression:

```
```{r}
X <- model.matrix(formula, data = jordan_prime_regular_model_master)[, -1]
y <- jordan_prime_regular_model_master$Wins

ridge_cv <- cv.glmnet(X, y, alpha = 0)
best_lambda_ridge <- ridge_cv$lambda.min
cat("Best Ridge lambda:", best_lambda_ridge, "\n")
cat("Best Ridge MSE:", min(ridge_cv$cvm), "\n")
```
```

Warning: Option grouped=FALSE enforced in cv.glmnet, since < 3 observations per fold

Best Ridge lambda: 0.8907974  
Best Ridge MSE: 28.15933

Lasso regression:

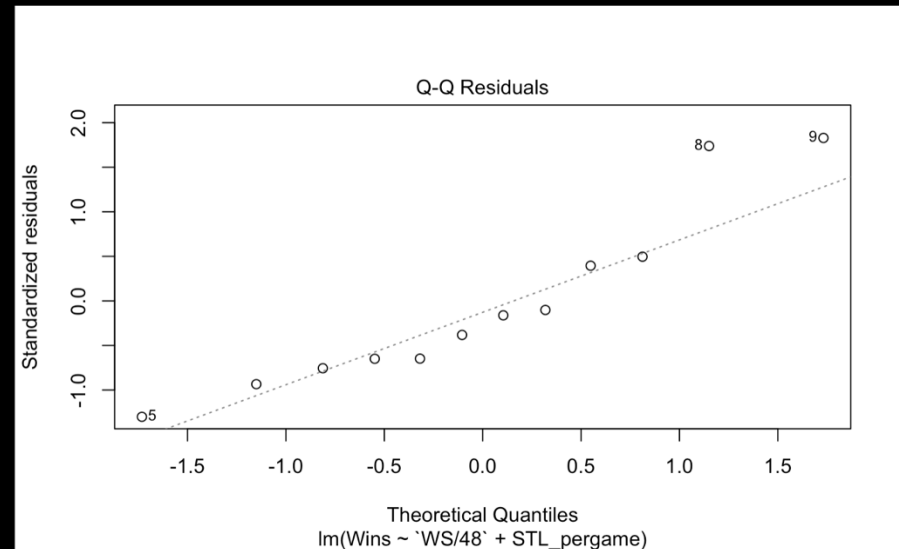
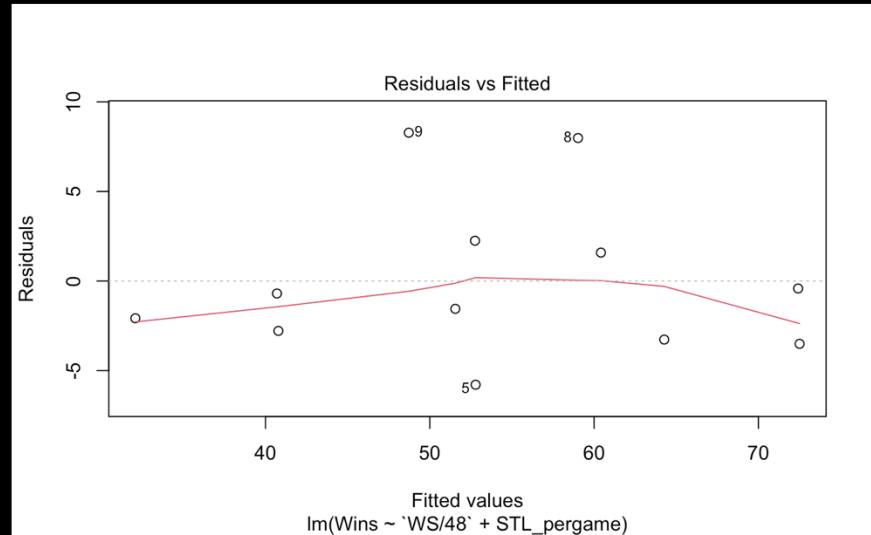
```
```{r}
lasso_cv <- cv.glmnet(X, y, alpha = 1)
best_lambda_lasso <- lasso_cv$lambda.min
cat("Best Lasso lambda:", best_lambda_lasso, "\n")
cat("Best Lasso MSE:", min(lasso_cv$cvm), "\n")
```
```

Warning: Option grouped=FALSE enforced in cv.glmnet, since < 3 observations per fold

Best Lasso lambda: 0.0534019  
Best Lasso MSE: 26.39115

# DIAGNOSTIC PLOTS

All three regression assumptions show some violations; residuals are not perfectly linear, evenly spread, or normally distributed. With only 12 data points and 9 degrees of freedom, this model may not be reliable for prediction, but it's the best fit we could create with the data we had



# STAGE 6: LOGISTIC MODELS, PART 1 (LEBRON)

-I used logistic regression to predict whether LeBron's team has a 60% win percentage in a given season based on his metrics

-Points and PER were the most useful because they are significant

-When LeBron's PER increases, the probability of his team winning 60% of games increases

-When LeBron's PTS increases, the probability of his team winning 60% of games decreases.

-His PER translated more to winning, as he could focus on his playstyle and not worry about his teammates. If he has less help, he might try to force more shots leading to more points, which might not impact winning as much

```
##{r}
model2 <- glm(
 Win60 ~ 1 + PER + PTS,
 data = lebron_james_logistic,
 family = "binomial"
)

summary(model2)
```

Call:  
glm(formula = Win60 ~ 1 + PER + PTS, family = "binomial", data = lebron\_james\_logistic)

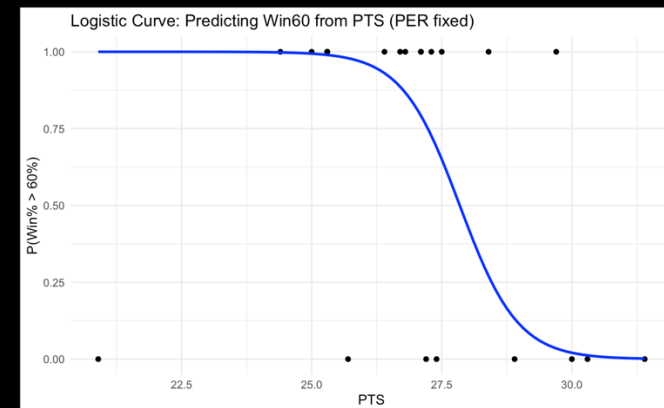
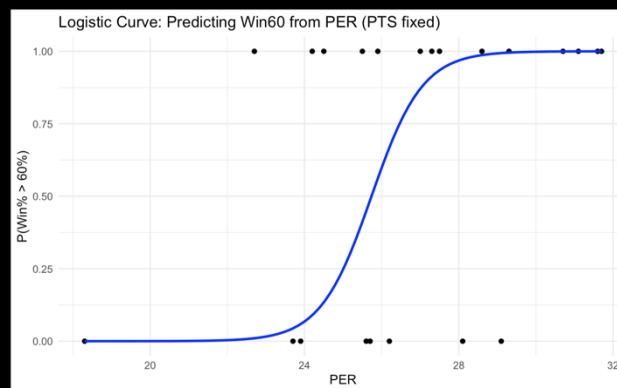
Coefficients:

|             | Estimate | Std. Error | z value | Pr(> z ) |
|-------------|----------|------------|---------|----------|
| (Intercept) | 9.5941   | 8.2489     | 1.163   | 0.2448   |
| PER         | 1.5095   | 0.7345     | 2.055   | 0.0399 * |
| PTS         | -1.7939  | 0.8930     | -2.009  | 0.0446 * |

---

Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)



---

|           | Actual |    |
|-----------|--------|----|
| Predicted | 0      | 1  |
| 0         | 8      | 1  |
| 1         | 0      | 13 |

---

### Logistic Model Evaluation:

I created a confusion matrix, which shows how many predictions were right or wrong (predicting a 60% vs an actual 60%)

95% accuracy is good (correct guesses)

93% sensitivity/power is good (how well we can correctly detect a 60% win percentage)

We were perfect in correctly guessing below a 60% win percentage

FPR/type I error rate of 0 (guessing a 60% win percentage when his actual win percentage was lower)

FNR/type II error rate of 7% (guessing below a 60% win percentage when his actual win percentage was greater)

Plotted an ROC curve. The higher the area under the curve, the better. This blue curve hugs the top right of the space, which is what we want.

Accuracy: 0.9545455

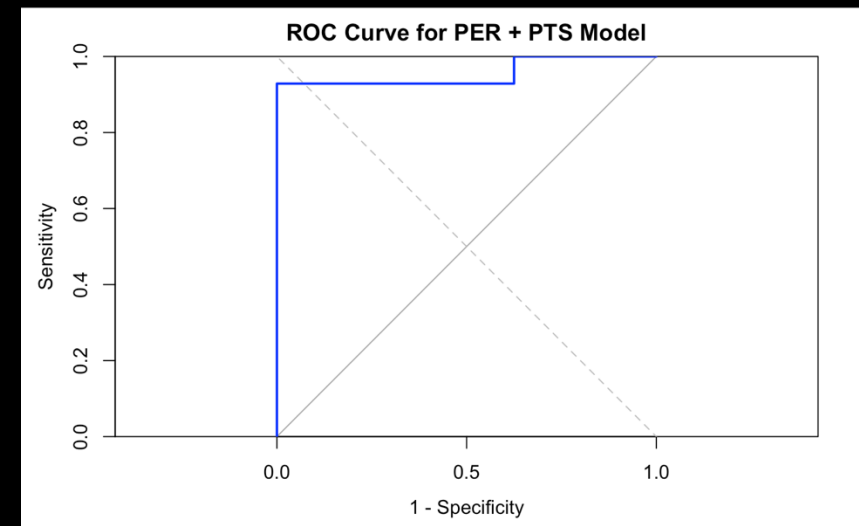
TPR (Sensitivity): 0.9285714

TNR (Specificity): 1

FPR (False Positive Rate): 0

FNR (False Negative Rate): 0.07142857

---



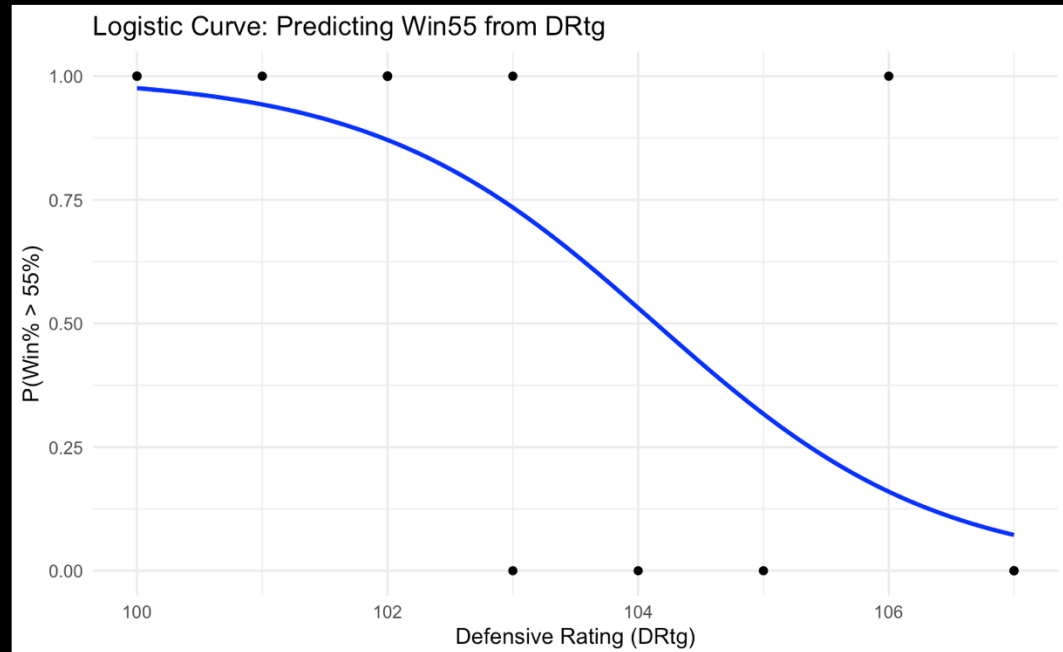


# STAGE 6: LOGISTIC MODELS, PART 2 (JORDAN)

We initially went with a 60 win percentage, and Defensive rating was the only statistic close to significance, so we dropped to a 55 win percentage

This model checks out as a lower (better) defensive rating predicts a higher probability of winning at least 55% of the games. As Jordan played better defense, it translated to more wins

```
jordan_model3 <- glm(Win55 ~ DRtg, data = jordan_logistic, family = "binomial")
summary(jordan_model3)
```  
  
Call:  
glm(formula = Win55 ~ DRtg, family = "binomial", data = jordan_logistic)  
  
Coefficients:  
             Estimate Std. Error z value Pr(>|z|)  
(Intercept)  92.9132    46.0632   2.017  0.0437 *  
DRtg         -0.8922     0.4448  -2.006  0.0449 *  
---  
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1  
  
(Dispersion parameter for binomial family taken to be 1)
```



Logistic Model Evaluation:

Here is the confusion matrix and calculated metrics

Accuracy: 79%

Sensitivity/power: 89%

Specificity: 60%

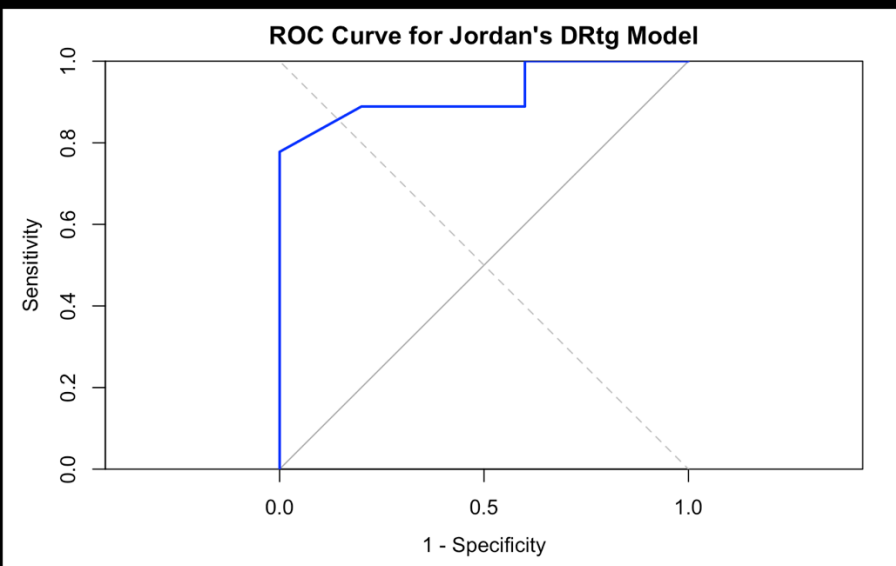
FPR/type I error rate: 40%


FNR/type II error rate: 11%

These metrics are not very strong, but this is the only model that has a significant predictor in our data. That is more important because there is actually evidence that the statistic of defensive rating can predict these binary responses

	Actual	
Predicted	0	1
0	3	1
1	2	8

Accuracy: 0.7857143
TPR (Sensitivity): 0.8888889
TNR (Specificity): 0.6
FPR (False Positive Rate): 0.4
FNR (False Negative Rate): 0.1111111





This is my first ever data science project. I used coding, visualizing, and advanced statistical analysis in R and Python to compare the careers of LeBron James and Michael Jordan. Jordan beat LeBron in scoring and prime-level impact in some areas. LeBron beat Jordan in playoff stats, longevity, and consistency. Linear and logistic models demonstrated that advanced statistics, such as PER and WS/48, are strong predictors of team success. Defensive rating stood out for Jordan's logistic model. Despite small sample sizes and assumption violations in some models, the analysis revealed how certain individual stats meaningfully correlate with team performance. Ultimately, the data suggests that the GOAT debate is more nuanced than we thought, as each player excels in different dimensions of greatness.