NBA Player Salary Analysis

2024-04-20

- NBA0 Original CSV Raw Dataset -NBA1 dataset = model 1 with SM
- NBA2 dataset = model without SM

```r
# Load the data

NBA0 <- read.csv("NBA.csv")




# Remove rows with NA values
NBA1 <- NBA0[complete.cases(NBA0), ]






# Step 1: Ensure 'Salary' is numeric
NBA1$Salary <- as.numeric(gsub("[^0-9.-]", "", NBA1$Salary))

# Step 2: Remove rows with NA in 'Salary'
NBA1 <- NBA1[!is.na(NBA1$Salary), ]
```

Model 1

```r
# multiple regression model
model1 <- lm(Salary ~ OFFRTG + DEFRTG + PGP + SM + Award, data = NBA1)
summary(model1)
```

```
##
## Call:
## lm(formula = Salary ~ OFFRTG + DEFRTG + PGP + SM + Award, data = NBA1)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -28430422  -5556185  -1527636   4658840  21436804
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -6.824e+07  2.835e+07  -2.407  0.01708 *
## OFFRTG       4.759e+05  1.536e+05   3.098  0.00226 **
## DEFRTG       1.878e+05  1.820e+05   1.032  0.30350
## PGP          1.309e+05  2.255e+04   5.804 2.84e-08 ***
## SM           3.291e-02  6.904e-02   0.477  0.63417
## Award        8.616e+06  1.439e+06   5.987 1.12e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 9339000 on 181 degrees of freedom
## Multiple R-squared:  0.4788, Adjusted R-squared:  0.4644
## F-statistic: 33.26 on 5 and 181 DF,  p-value: < 2.2e-16
```

Second Model wiithout Social Media variable (SM)

```r
# Remove SM
NBA2 <- NBA1[, !names(NBA1) %in% "SM"]



model2 = lm(Salary ~ OFFRTG + DEFRTG + PGP + Award, data = NBA2)
summary(model2)
```

```
##
## Call:
## lm(formula = Salary ~ OFFRTG + DEFRTG + PGP + Award, data = NBA2)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -29313308  -5715766  -1456557   4584147  21565748
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -67593807   28257099  -2.392  0.01777 *
## OFFRTG         463574     151087   3.068  0.00248 **
## DEFRTG         193329     181270   1.067  0.28760
## PGP            137363      17962   7.648 1.14e-12 ***
## Award         8648896    1434440   6.029 8.96e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9319000 on 182 degrees of freedom
## Multiple R-squared:  0.4782, Adjusted R-squared:  0.4667
## F-statistic: 41.69 on 4 and 182 DF,  p-value: < 2.2e-16
```

```r
# Remove SM
NBA2 <- NBA1[, !names(NBA1) %in% "SM"]

summary(NBA2)
```

```
##     Player              OFFRTG          DEFRTG           PGP
##  Length:187         Min.   :104.8   Min.   :104.7   Min.   :  0.00
##  Class :character   1st Qu.:111.3   1st Qu.:111.6   1st Qu.:  0.00
##  Mode  :character   Median :114.9   Median :114.5   Median : 22.00
##                     Mean   :114.9   Mean   :114.2   Mean   : 32.03
##                     3rd Qu.:118.8   3rd Qu.:116.3   3rd Qu.: 43.50
##                     Max.   :125.3   Max.   :123.9   Max.   :282.00
##      Award           Salary
##  Min.   :0.000   Min.   : 1119563
##  1st Qu.:0.000   1st Qu.: 5848167
##  Median :0.000   Median :12015150
##  Mean   :0.492   Mean   :16392242
##  3rd Qu.:1.000   3rd Qu.:24107143
##  Max.   :1.000   Max.   :51915615
```

```r
# Calculate mean and standard deviation for continuous variables, handling missing values

mean_salary <- mean(NBA2$Salary, na.rm = TRUE)
sd_salary <- sd(NBA2$Salary, na.rm = TRUE)

mean_offrtg <- mean(NBA2$OFFRTG, na.rm = TRUE)
sd_offrtg <- sd(NBA2$OFFRTG, na.rm = TRUE)

mean_defrtg <- mean(NBA2$DEFRTG, na.rm = TRUE)
sd_defrtg <- sd(NBA2$DEFRTG, na.rm = TRUE)

mean_pgp <- mean(NBA2$PGP, na.rm = TRUE)
sd_pgp <- sd(NBA2$PGP, na.rm = TRUE)

# Calculate frequency and proportion for categorical variable 'Award'
award_frequency <- table(NBA2$Award)
award_proportion <- prop.table(award_frequency)

# Print mean and standard deviation for continuous variables
cat("Mean Salary:", mean_salary, "\n")
```

```
## Mean Salary: 16392242
```

```r
cat("Standard Deviation of Salary:", sd_salary, "\n")
```

```
## Standard Deviation of Salary: 12761318
```

```r
cat("Mean OFFRTG:", mean_offrtg, "\n")
```

```
## Mean OFFRTG: 114.877
```

```r
cat("Standard Deviation of OFFRTG:", sd_offrtg, "\n")
```

```
## Standard Deviation of OFFRTG: 4.794761
```

```r
cat("Mean DEFRTG:", mean_defrtg, "\n")
```

```
## Mean DEFRTG: 114.1973
```

```r
cat("Standard Deviation of DEFRTG:", sd_defrtg, "\n")
```

```
## Standard Deviation of DEFRTG: 3.828507
```

```r
cat("Mean PGP:", mean_pgp, "\n")
```

```
## Mean PGP: 32.02674
```

```r
cat("Standard Deviation of PGP:", sd_pgp, "\n")
```

```
## Standard Deviation of PGP: 40.82133
```

```r
# Print frequency and proportion for categorical variable 'Award'
cat("Frequency of Awards (Award):\n")
```

```
## Frequency of Awards (Award):
```

```r
print(award_frequency)
```

```
##
##   0   1
```

```
## 95 92
```

```r
cat("Proportion of Awards (Award):\n")
```

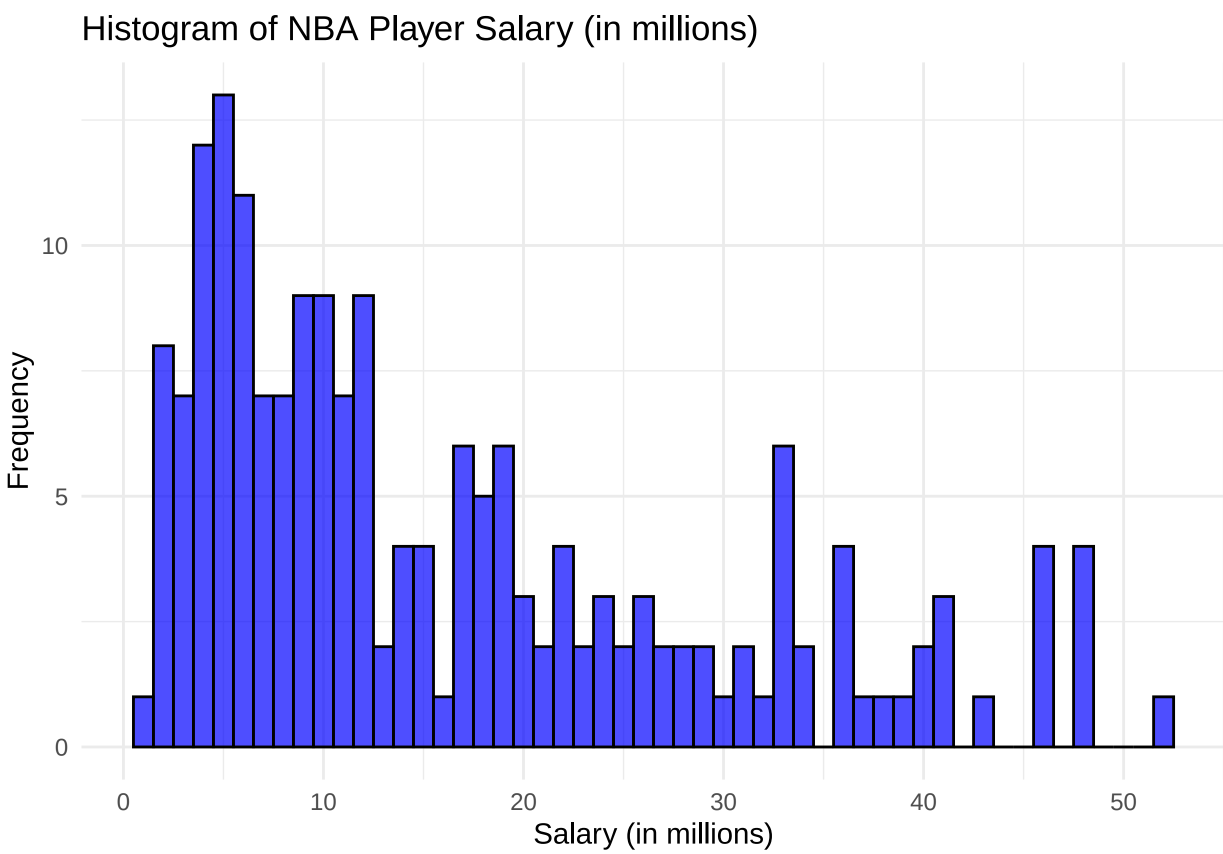
```
## Proportion of Awards (Award):
```

```r
print(award_proportion)
```

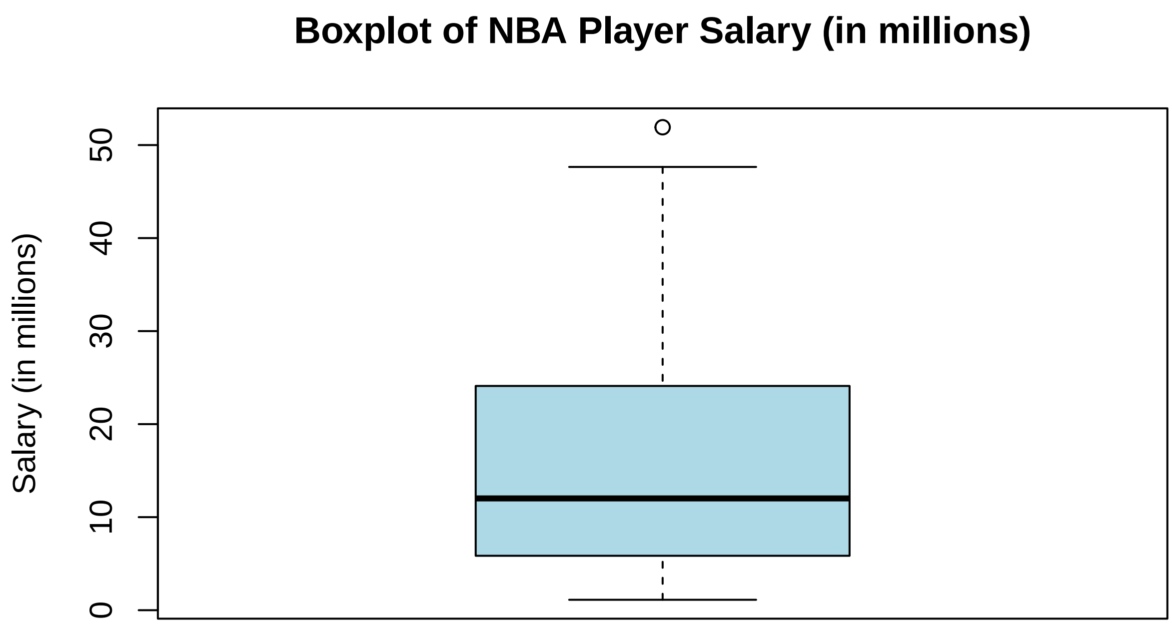
```
##
##         0         1
## 0.5080214 0.4919786
```

```r
# Load necessary package
library(ggplot2)

# Create a histogram of the dependent variable (Salary in millions)
ggplot(NBA2, aes(x = Salary / 1000000)) +
    geom_histogram(binwidth = 1, fill = "blue", color = "black", alpha = 0.7) +
    labs(title = "Histogram of NBA Player Salary (in millions)",
         x = "Salary (in millions)",
         y = "Frequency") +
    theme_minimal()
```




```r
# Create a boxplot of the dependent variable (Salary in millions)
boxplot(NBA2$Salary / 1000000,
        main = "Boxplot of NBA Player Salary (in millions)",
        ylab = "Salary (in millions)",
        col = "lightblue",
        outline = TRUE)
```




```r
# Find the summary statistics for the Salary variable
summary(NBA2$Salary)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##  1119563  5848167 12015150 16392242 24107143 51915615
```

```r
# Load necessary libraries
library(corrplot)
```

```
## corrplot 0.92 loaded
```

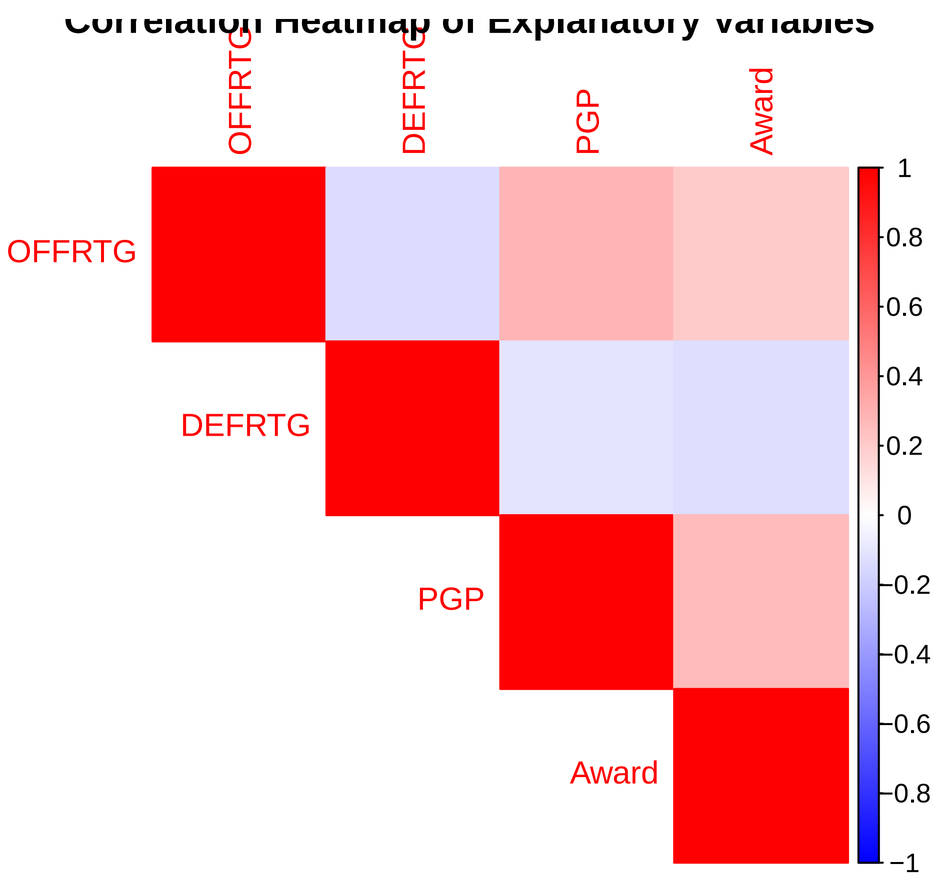
```r
# Assuming NBA2 data frame is already loaded

# Step 1: Check for missing values
#missing_values <- sum(is.na(NBA2))
#cat("Number of missing values in the data:", missing_values, "\n")

# Step 2: Handle missing values
# Option 1: Remove rows with missing values
NBA2_clean <- na.omit(NBA2)

# Step 3: Calculate correlation matrix
corr_matrix <- cor(NBA2_clean[, c("OFFRTG", "DEFRTG", "PGP", "Award")])

# Step 4: Create a heatmap of the correlation matrix
corrplot(corr_matrix, method = "color", type = "upper",
         col = colorRampPalette(c("blue", "white", "red"))(200),
         title = "Correlation Heatmap of Explanatory Variables")
```




```r
# Load necessary packages
library(car)
```

```
## Loading required package: carData
```

```r
library(dplyr)
```

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

```
## The following object is masked from 'package:car':
##
##     recode
```

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

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

```r
# Assuming NBA2 data frame is already loaded
# Select the explanatory variables from the dataset
explanatory_vars <- NBA2 %>% select(OFFRTG, DEFRTG, PGP, Award)

# Fit a linear model using the explanatory variables and the dependent variable (Salary)
model <- lm(Salary ~ OFFRTG + DEFRTG + PGP + Award, data = NBA2)
```

```r
# Calculate VIF for each explanatory variable
vif_values <- vif(model)

# Print the VIF values
print(vif_values)
```

```
##   OFFRTG   DEFRTG      PGP    Award
## 1.123899 1.031455 1.151354 1.107292
```

```r
# 4 models
S1 <- lm(Salary ~ OFFRTG, data = NBA2)
S2 <- lm(Salary ~ OFFRTG + DEFRTG, data = NBA2)
S3 <- lm(Salary ~ OFFRTG + DEFRTG + PGP, data = NBA2)
S4 <- lm(Salary ~ OFFRTG + DEFRTG + PGP + Award, data = NBA2)

# Compute the 95% confidence intervals for each model and print them

# Model S1: lm(Salary ~ OFFRTG, data = NBA2)
ci_S1 <- confint(S1)
print("Model S1: 95% Confidence Intervals")
```

```
## [1] "Model S1: 95% Confidence Intervals"
```

```r
print(ci_S1)
```

```
##                   2.5 %    97.5 %
## (Intercept) -135975693.2 -53260886
## OFFRTG          606639.4   1326046
```

```r
# Model S2: lm(Salary ~ OFFRTG + DEFRTG, data = NBA2)
ci_S2 = confint(S2)
print("\nModel S2: 95% Confidence Intervals")
```

```
## [1] "\nModel S2: 95% Confidence Intervals"
```

```r
print(ci_S2)
```

```
##                   2.5 %      97.5 %
## (Intercept) -164240450.4 -21961364.6
## OFFRTG          600916.9   1329151.1
## DEFRTG         -467985.6    444043.3
```

```r
# Model S3: lm(Salary ~ OFFRTG + DEFRTG + PGP, data = NBA2)
ci_S3 = confint(S3)
print("\nModel S3: 95% Confidence Intervals")
```

```
## [1] "\nModel S3: 95% Confidence Intervals"
```

```r
print(ci_S3)
```

```
##                   2.5 %     97.5 %
## (Intercept) -127468121.9 -5671617.5
## OFFRTG          257111.5   903006.4
## DEFRTG         -291372.2   486979.0
## PGP             123272.5   198820.2
```

```r
# Model S4: lm(Salary ~ OFFRTG + DEFRTG + PGP + Award, data = NBA2)
ci_S4 = confint(S4)
print("\nModel S4: 95% Confidence Intervals")
```

```
## [1] "\nModel S4: 95% Confidence Intervals"
```

```r
print(ci_S4)
```

```
##                    2.5 %      97.5 %
## (Intercept) -123347438.8 -11840175.3
## OFFRTG          165467.5    761681.4
## DEFRTG         -164331.9    550989.5
## PGP             101923.5    172803.3
## Award          5818625.5  11479167.4
```

```r
# Calculate fitted values for Model S1
fitted_values_S1 <- predict(S1, newdata = NBA2)

# Calculate fitted values for Model S2
fitted_values_S2 <- predict(S2, newdata = NBA2)

# Calculate fitted values for Model S3
fitted_values_S3 <- predict(S3, newdata = NBA2)

# Calculate fitted values for Model S4
fitted_values_S4 <- predict(S4, newdata = NBA2)

# Display the fitted values for each model
```

```r
# Calculate confidence and prediction intervals for Model S1
conf_int_S1 <- predict(S1, newdata = NBA2, interval = "confidence")
pred_int_S1 <- predict(S1, newdata = NBA2, interval = "prediction")

# Calculate confidence and prediction intervals for Model S2
conf_int_S2 = predict(S2, newdata = NBA2, interval = "confidence")
pred_int_S2 = predict(S2, newdata = NBA2, interval = "prediction")

# Calculate confidence and prediction intervals for Model S3
conf_int_S3 = predict(S3, newdata = NBA2, interval = "confidence")
pred_int_S3 = predict(S3, newdata = NBA2, interval = "prediction")

# Calculate confidence and prediction intervals for Model S4
conf_int_S4 = predict(S4, newdata = NBA2, interval = "confidence")
pred_int_S4 = predict(S4, newdata = NBA2, interval = "prediction")
```

```r
# Assuming you have defined your models as S1, S2, S3, S4 and you have your NBA2 data set loaded

# Calculate fitted values for each model
fitted_values_S1 <- predict(S1, newdata = NBA2)
fitted_values_S2 <- predict(S2, newdata = NBA2)
fitted_values_S3 <- predict(S3, newdata = NBA2)
fitted_values_S4 <- predict(S4, newdata = NBA2)

# Display the first 5 comparisons of actual and fitted values for each model

# Model S1
cat("Model S1: Actual vs. Fitted Values\n")
```

```
## Model S1: Actual vs. Fitted Values
```

```r
comparison_S1 <- data.frame(
    Actual = NBA2$Salary[1:5],
    Fitted = fitted_values_S1[1:5]
)


# Model S2
cat("\nModel S2: Actual vs. Fitted Values\n")
```

```
##
## Model S2: Actual vs. Fitted Values
```

```r
comparison_S2 <- data.frame(
    Actual = NBA2$Salary[1:5],
    Fitted = fitted_values_S2[1:5]
)


# Model S3
cat("\nModel S3: Actual vs. Fitted Values\n")
```

```
##
## Model S3: Actual vs. Fitted Values
```

```r
comparison_S3 <- data.frame(
    Actual = NBA2$Salary[1:5],
    Fitted = fitted_values_S3[1:5]
)


# Model S4
cat("\nModel S4: Actual vs. Fitted Values\n")
```

```
##
## Model S4: Actual vs. Fitted Values
```

```r
comparison_S4 <- data.frame(
    Actual = NBA2$Salary[1:5],
    Fitted = fitted_values_S4[1:5]
)
```

```r
# Calculate residuals for each model
residuals_S1 <- residuals(S1)
residuals_S2 <- residuals(S2)
residuals_S3 <- residuals(S3)
residuals_S4 <- residuals(S4)

# Find the smallest and largest residuals for Model S1
smallest_residual_S1 <- min(residuals_S1, na.rm = TRUE)
largest_residual_S1 <- max(residuals_S1, na.rm = TRUE)

# Find the smallest and largest residuals for Model S2
smallest_residual_S2 <- min(residuals_S2, na.rm = TRUE)
largest_residual_S2 <- max(residuals(S2), na.rm = TRUE)

# Find the smallest and largest residuals for Model S3
```

```r
smallest_residual_S3 <- min(residuals_S3, na.rm = TRUE)
largest_residual_S3 <- max(residuals_S3, na.rm = TRUE)

# Find the smallest and largest residuals for Model S4
smallest_residual_S4 <- min(residuals_S4, na.rm = TRUE)
largest_residual_S4 <- max(residuals_S4, na.rm = TRUE)

# Display the results
cat("Model S1: Smallest Residual:", smallest_residual_S1, "\n")
```

```
## Model S1: Smallest Residual: -20655989
```

```r
cat("Model S1: Largest Residual:", largest_residual_S1, "\n")
```

```
## Model S1: Largest Residual: 33858370
```

```r
cat("\n")
```

```r
cat("Model S2: Smallest Residual:", smallest_residual_S2, "\n")
```

```
## Model S2: Smallest Residual: -20586055
```

```r
cat("Model S2: Largest Residual:", largest_residual_S2, "\n")
```

```
## Model S2: Largest Residual: 33924104
```

```r
cat("\n")
```

```r
cat("Model S3: Smallest Residual:", smallest_residual_S3, "\n")
```

```
## Model S3: Smallest Residual: -29063446
```

```r
cat("Model S3: Largest Residual:", largest_residual_S3, "\n")
```

```
## Model S3: Largest Residual: 24174879
```

```r
cat("\n")
```

```r
cat("Model S4: Smallest Residual:", smallest_residual_S4, "\n")
```

```
## Model S4: Smallest Residual: -29313308
```

```r
cat("Model S4: Largest Residual:", largest_residual_S4, "\n")
```

```
## Model S4: Largest Residual: 21565748
```