4_5010_final_project

December 17, 2024

1 STAT 4010/5010 Final Project

2 Statistical Analysis of Baseball Performance Metrics: Findings on Hitting, Pitching, and Salary Predictors

2.1 Dr. Osita Onyejekwe

2.1.1 By Kathryn Stewart, Annika Strom, and Anya Lee

3 Install Packages, Import Libraries, and Load Data

[ ]:
```
# Install any unknown packages
install.packages("leaps")
install.packages("car")
install.packages("reshape2")
install.packages("pheatmap")
```

```
Installing package into '/usr/local/lib/R/site-library'
(as 'lib' is unspecified)

Installing package into '/usr/local/lib/R/site-library'
(as 'lib' is unspecified)

also installing the dependency 'pbkrtest'


Installing package into '/usr/local/lib/R/site-library'
(as 'lib' is unspecified)

Installing package into '/usr/local/lib/R/site-library'
(as 'lib' is unspecified)
```

[ ]:
```
# Import libraries / load packages
library(ggplot2) # For plotting
library(MASS)  # For stepwise regression
library(leaps) # For all subsets regression
library(car) # For the vif() function
```

```
library(dplyr) # For data manipulation
library(reshape2) # For correlation heat map, to reshape data
library(pheatmap) # To visualize heat map
library(tidyr) # For data cleaning/removing columns
```

```
Loading required package: carData


Attaching package: ‘dplyr’


The following object is masked from ‘package:car’:

    recode


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



Attaching package: ‘tidyr’


The following object is masked from ‘package:reshape2’:

    smiths
```

[ ]:

```
# Read CSV files
pitching_data <- read.csv('Pitching.csv')
fielding_data <- read.csv('Fielding.csv')
batting_data <- read.csv('Batting.csv')
salary_data <- read.csv('Salaries.csv')
```

```r
# View each datasets
head(pitching_data)
head(fielding_data)
head(batting_data)
head(salary_data)

# Get column names of each dataset
colnames(pitching_data)
colnames(fielding_data)
colnames(batting_data)
colnames(salary_data)

# Number of data points for each file
nrow(pitching_data)
nrow(fielding_data)
nrow(batting_data)
nrow(salary_data)
```

A data.frame: 6 × 30

	playerID <chr>	yearID <int>	stint <int>	teamID <chr>	lgID <chr>	W <int>	L <int>	G <int>	GS <int>	CG <i
1	bechtge01	1871	1	PH1	NA	1	2	3	3	2
2	brainas01	1871	1	WS3	NA	12	15	30	30	30
3	fergubo01	1871	1	NY2	NA	0	0	1	0	0
4	fishech01	1871	1	RC1	NA	4	16	24	24	22
5	fleetfr01	1871	1	NY2	NA	0	1	1	1	1
6	flowedi01	1871	1	TRO	NA	0	0	1	0	0

A data.frame: 6 × 18

	playerID <chr>	yearID <int>	stint <int>	teamID <chr>	lgID <chr>	POS <chr>	G <int>	GS <int>	InnOuts <int>
1	abercda01	1871	1	TRO	NA	SS	1	NA	NA
2	addybo01	1871	1	RC1	NA	2B	22	NA	NA
3	addybo01	1871	1	RC1	NA	SS	3	NA	NA
4	allisar01	1871	1	CL1	NA	2B	2	NA	NA
5	allisar01	1871	1	CL1	NA	OF	29	NA	NA
6	allisdo01	1871	1	WS3	NA	C	27	NA	NA

A data.frame: 6 × 22

	playerID <chr>	yearID <int>	stint <int>	teamID <chr>	lgID <chr>	G <int>	AB <int>	R <int>	H <int>	X <
1	abercda01	1871	1	TRO	NA	1	4	0	0	0
2	addybo01	1871	1	RC1	NA	25	118	30	32	6
3	allisar01	1871	1	CL1	NA	29	137	28	40	4
4	allisdo01	1871	1	WS3	NA	27	133	28	44	10
5	ansonca01	1871	1	RC1	NA	25	120	29	39	11
6	armstbo01	1871	1	FW1	NA	12	49	9	11	2

A data.frame: 6 × 5

	yearID <int>	teamID <chr>	lgID <chr>	playerID <chr>	salary <int>
1	1985	ATL	NL	barkele01	870000
2	1985	ATL	NL	bedrost01	550000
3	1985	ATL	NL	benedbr01	545000
4	1985	ATL	NL	campri01	633333
5	1985	ATL	NL	ceronri01	625000
6	1985	ATL	NL	chambch01	800000

1. 'playerID' 2. 'yearID' 3. 'stint' 4. 'teamID' 5. 'lgID' 6. 'W' 7. 'L' 8. 'G' 9. 'GS' 10. 'CG' 11. 'SHO' 12. 'SV' 13. 'IPouts' 14. 'H' 15. 'ER' 16. 'HR' 17. 'BB' 18. 'SO' 19. 'BAOpp' 20. 'ERA' 21. 'IBB' 22. 'WP' 23. 'HBP' 24. 'BK' 25. 'BFP' 26. 'GF' 27. 'R' 28. 'SH' 29. 'SF' 30. 'GIDP'

1. 'playerID' 2. 'yearID' 3. 'stint' 4. 'teamID' 5. 'lgID' 6. 'POS' 7. 'G' 8. 'GS' 9. 'InnOuts' 10. 'PO' 11. 'A' 12. 'E' 13. 'DP' 14. 'PB' 15. 'WP' 16. 'SB' 17. 'CS' 18. 'ZR'

1. 'playerID' 2. 'yearID' 3. 'stint' 4. 'teamID' 5. 'lgID' 6. 'G' 7. 'AB' 8. 'R' 9. 'H' 10. 'X2B' 11. 'X3B' 12. 'HR' 13. 'RBI' 14. 'SB' 15. 'CS' 16. 'BB' 17. 'SO' 18. 'IBB' 19. 'HBP' 20. 'SH' 21. 'SF' 22. 'GIDP'

1. 'yearID' 2. 'teamID' 3. 'lgID' 4. 'playerID' 5. 'salary'

44139

170526

101332

25575

4 Data Cleaning: Missing Values, Duplicates, Data Types

[ ]:
```
# Data Cleaning: Check for Missing Values (NA)
sapply(pitching_data, function(x) sum(is.na(x)))
sapply(fielding_data, function(x) sum(is.na(x)))
sapply(batting_data, function(x) sum(is.na(x)))
sapply(salary_data, function(x) sum(is.na(x)))
```

playerID 0 **yearID** 0 **stint** 0 **teamID** 0 **lgID** 131 **W** 0 **L** 0 **G** 0 **GS** 0 **CG** 0 **SHO** 0 **SV** 0 **IPouts** 1 **H** 0 **ER** 0 **HR** 0 **BB** 0 **SO** 0 **BAOpp** 1525 **ERA** 90 **IBB** 14575 **WP** 133 **HBP** 559 **BK** 0 **BFP** 239 **GF** 133 **R** 0 **SH** 32900 **SF** 32900 **GIDP** 43394

playerID 0 **yearID** 0 **stint** 0 **teamID** 0 **lgID** 1503 **POS** 0 **G** 0 **GS** 94677 **InnOuts** 68213 **PO** 14117 **A** 14118 **E** 14119 **DP** 14118 **PB** 159410 **WP** 166337 **SB** 164502 **CS** 164502 **ZR** 166337

playerID 0 **yearID** 0 **stint** 0 **teamID** 0 **lgID** 737 **G** 0 **AB** 5149 **R** 5149 **H** 5149 **X2B** 5149 **X3B** 5149 **HR** 5149 **RBI** 5573 **SB** 6449 **CS** 28603 **BB** 5149 **SO** 12987 **IBB** 41712 **HBP** 7959 **SH** 11487 **SF** 41181 **GIDP** 31257

yearID 0 **teamID** 0 **lgID** 0 **playerID** 0 **salary** 0

[ ]:
```
# Data Cleaning: Check for Duplicates
sum(duplicated(pitching_data))
```

```
sum(duplicated(fielding_data))
sum(duplicated(batting_data))
sum(duplicated(salary_data))
```

0

0

0

0

[ ]:
```
# Data Cleaning: Check Data Types
str(pitching_data)
str(fielding_data)
str(batting_data)
str(salary_data)
```

```
'data.frame':	44139 obs. of  30 variables:
 $ playerID: chr  "bechtge01" "brainas01" "fergubo01" "fishech01" …
 $ yearID  : int  1871 1871 1871 1871 1871 1871 1871 1871 1871 1871 …
 $ stint   : int  1 1 1 1 1 1 1 1 1 1 …
 $ teamID  : chr  "PH1" "WS3" "NY2" "RC1" …
 $ lgID    : chr  NA NA NA NA …
 $ W       : int  1 12 0 4 0 0 0 6 18 12 …
 $ L       : int  2 15 0 16 1 0 1 11 5 15 …
 $ G       : int  3 30 1 24 1 1 3 19 25 29 …
 $ GS      : int  3 30 0 24 1 0 1 19 25 29 …
 $ CG      : int  2 30 0 22 1 0 1 19 25 28 …
 $ SHO     : int  0 0 0 1 0 0 0 1 0 0 …
 $ SV      : int  0 0 0 0 0 0 0 0 0 0 …
 $ IPouts  : int  78 792 3 639 27 3 39 507 666 747 …
 $ H       : int  43 361 8 295 20 1 20 261 285 430 …
 $ ER      : int  23 132 3 103 10 0 5 97 113 153 …
 $ HR      : int  0 4 0 3 0 0 0 5 3 4 …
 $ BB      : int  11 37 0 31 3 0 3 21 40 75 …
 $ SO      : int  1 13 0 15 0 0 1 17 15 12 …
 $ BAOpp   : num  NA NA NA NA NA NA NA NA NA NA …
 $ ERA     : num  7.96 4.5 27 4.35 10 0 3.46 5.17 4.58 5.53 …
 $ IBB     : int  NA NA NA NA NA NA NA NA NA NA …
 $ WP      : int  NA NA NA NA NA NA NA NA NA NA …
 $ HBP     : int  NA NA NA NA NA NA NA NA NA NA …
 $ BK      : int  0 0 0 0 0 0 0 2 0 0 …
 $ BFP     : int  NA NA NA NA NA NA NA NA NA NA …
 $ GF      : int  NA NA NA NA NA NA NA NA NA NA …
 $ R       : int  42 292 9 257 21 0 30 243 223 362 …
 $ SH      : int  NA NA NA NA NA NA NA NA NA NA …
 $ SF      : int  NA NA NA NA NA NA NA NA NA NA …
 $ GIDP    : int  NA NA NA NA NA NA NA NA NA NA …
```

```
'data.frame':   170526 obs. of  18 variables:
 $ playerID: chr  "abercda01" "addybo01" "addybo01" "allisar01" …
 $ yearID  : int  1871 1871 1871 1871 1871 1871 1871 1871 1871 1871 …
 $ stint   : int  1 1 1 1 1 1 1 1 1 1 …
 $ teamID  : chr  "TRO" "RC1" "RC1" "CL1" …
 $ lgID    : chr  NA NA NA NA …
 $ POS     : chr  "SS" "2B" "SS" "2B" …
 $ G       : int  1 22 3 2 29 27 1 2 20 5 …
 $ GS      : int  NA NA NA NA NA NA NA NA NA NA …
 $ InnOuts : int  NA NA NA NA NA NA NA NA NA NA …
 $ PO      : int  1 67 8 1 51 68 7 3 38 10 …
 $ A       : int  3 72 14 4 3 15 0 4 52 0 …
 $ E       : int  2 42 7 0 7 20 0 1 28 8 …
 $ DP      : int  0 5 0 0 1 4 0 0 2 0 …
 $ PB      : int  NA NA NA NA NA 0 NA NA NA 0 …
 $ WP      : int  NA NA NA NA NA NA NA NA NA NA …
 $ SB      : int  NA NA NA NA NA NA NA NA NA NA …
 $ CS      : int  NA NA NA NA NA NA NA NA NA NA …
 $ ZR      : int  NA NA NA NA NA NA NA NA NA NA …
'data.frame':   101332 obs. of  22 variables:
 $ playerID: chr  "abercda01" "addybo01" "allisar01" "allisdo01" …
 $ yearID  : int  1871 1871 1871 1871 1871 1871 1871 1871 1871 1871 …
 $ stint   : int  1 1 1 1 1 1 1 1 1 1 …
 $ teamID  : chr  "TRO" "RC1" "CL1" "WS3" …
 $ lgID    : chr  NA NA NA NA …
 $ G       : int  1 25 29 27 25 12 1 31 1 18 …
 $ AB      : int  4 118 137 133 120 49 4 157 5 86 …
 $ R       : int  0 30 28 28 29 9 0 66 1 13 …
 $ H       : int  0 32 40 44 39 11 1 63 1 13 …
 $ X2B     : int  0 6 4 10 11 2 0 10 1 2 …
 $ X3B     : int  0 0 5 2 3 1 0 9 0 1 …
 $ HR      : int  0 0 0 2 0 0 0 0 0 0 …
 $ RBI     : int  0 13 19 27 16 5 2 34 1 11 …
 $ SB      : int  0 8 3 1 6 0 0 11 0 1 …
 $ CS      : int  0 1 1 1 2 1 0 6 0 0 …
 $ BB      : int  0 4 2 0 2 0 1 13 0 0 …
 $ SO      : int  0 0 5 2 1 1 0 1 0 0 …
 $ IBB     : int  NA NA NA NA NA NA NA NA NA NA …
 $ HBP     : int  NA NA NA NA NA NA NA NA NA NA …
 $ SH      : int  NA NA NA NA NA NA NA NA NA NA …
 $ SF      : int  NA NA NA NA NA NA NA NA NA NA …
 $ GIDP    : int  NA NA NA NA NA NA NA NA NA NA …
'data.frame':   25575 obs. of  5 variables:
 $ yearID  : int  1985 1985 1985 1985 1985 1985 1985 1985 1985 1985 …
 $ teamID  : chr  "ATL" "ATL" "ATL" "ATL" …
 $ lgID    : chr  "NL" "NL" "NL" "NL" …
 $ playerID: chr  "barkele01" "bedrost01" "benedbr01" "campri01" …
 $ salary  : int  870000 550000 545000 633333 625000 800000 150000 483333 772000
```

250000 …

5 Data Analysis Overview

5.1 1. Batting

- Batting Average and On-Base-Percentage: Is there a relationship between batting average and ability to get on base?
- Stolen Base Percentage and Runs Scored: Is there a relationship between speed on the bases and scoring efficiency?
- Strikeouts and At-Bats: Visualize relationship between the two
- Strikeout Percentage and Walk Rate for Batters
- Predicting Strikeouts by a Batter

5.2 2. Pitching

- Strikeout Percentage and Walk Rate, ERA: Evaluate basic pitching metrics
- Strikeout to Walk Ratio Relative to Runs Allowed: Evaluate overall pitcher performance
- Runs Allowed vs Walks: Does limiting number of walks decrease runs allowed?
- Predicting Strikeouts by a Pitcher
- Strikeouts vs ERA: Visualize relationship between the two.
- Predicting Pitchers ERA: Which predictors are most relevant to pitching ERA?

5.3 3. Salaries, Fielding, and Batting

- What position makes the most?
- Are there any correlations between salary and fielding position? (excluding pitchers) – correlation/heat map?
- Predicting salary for pitchers. What are the most relevant predictors?
- Predicting salary based on hitting/batting (excluding pitchers). What are the most significant predictors?
- Which position has most HRs, has most triples, etc. Show via Histogram

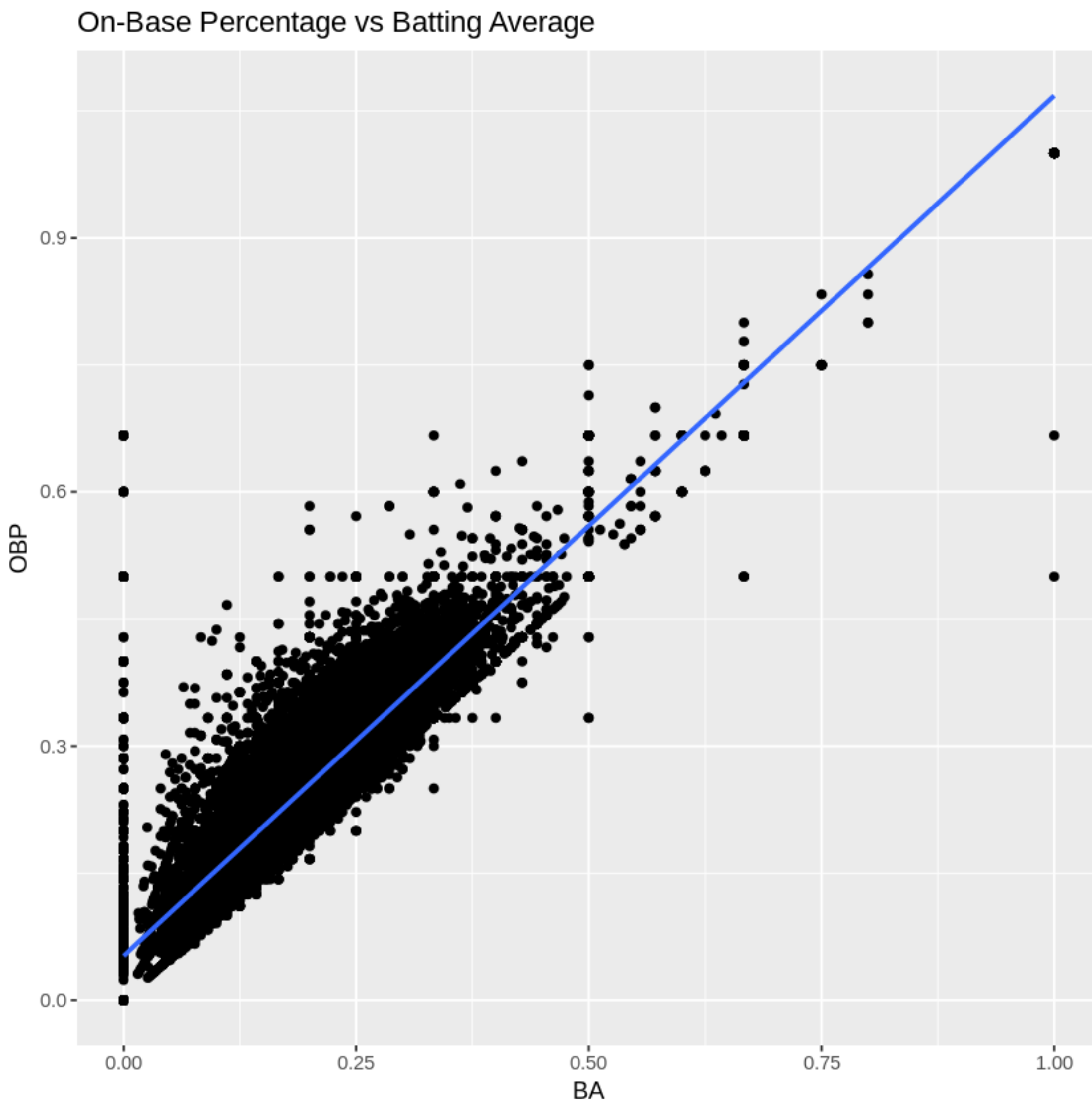
6 1. Batting

6.0.1 Batting Average and On-Base-Percentage: Is there a relationship between batting average and ability to get on base?

[ ]:
```
# Batting: Batting Average and On-Base-Percentage
# Compute Batting Average (BA)
batting_data$BA <- batting_data$H / batting_data$AB
# Compute On-Base Percentage (OBP)
batting_data$OBP <- (batting_data$H + batting_data$BB + batting_data$HBP) / # H:
 ↪ hit, HBP: hit by pitch
                    (batting_data$AB + batting_data$BB + batting_data$HBP +␣
 ↪batting_data$SF) # AB: at bats, BB: balls on base (walk), SF: sacrifice fly
# Linear model
lm_ba_obp <- lm(OBP ~ BA, data = batting_data)
```

```r
summary(lm_ba_obp)

# Scatterplot with regression line
ggplot(batting_data, aes(x = BA, y = OBP)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE) +
  labs(title = "On-Base Percentage vs Batting Average")
```

```
Call:
lm(formula = OBP ~ BA, data = batting_data)

Residuals:
     Min       1Q   Median       3Q      Max 
-0.56784 -0.04269 -0.00401  0.02261  0.61437 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 0.052297   0.000471   111.0   <2e-16 ***
BA          1.015544   0.001974   514.4   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.05579 on 50679 degrees of freedom
  (50651 observations deleted due to missingness)
Multiple R-squared:  0.8393,	Adjusted R-squared:  0.8393 
F-statistic: 2.646e+05 on 1 and 50679 DF,  p-value: < 2.2e-16


`geom_smooth()` using formula = 'y ~ x'
Warning message:
"Removed 50651 rows containing non-finite outside the scale range
(`stat_smooth()`)."
Warning message:
"Removed 50651 rows containing missing values or values outside the
scale range
(`geom_point()`)."
```



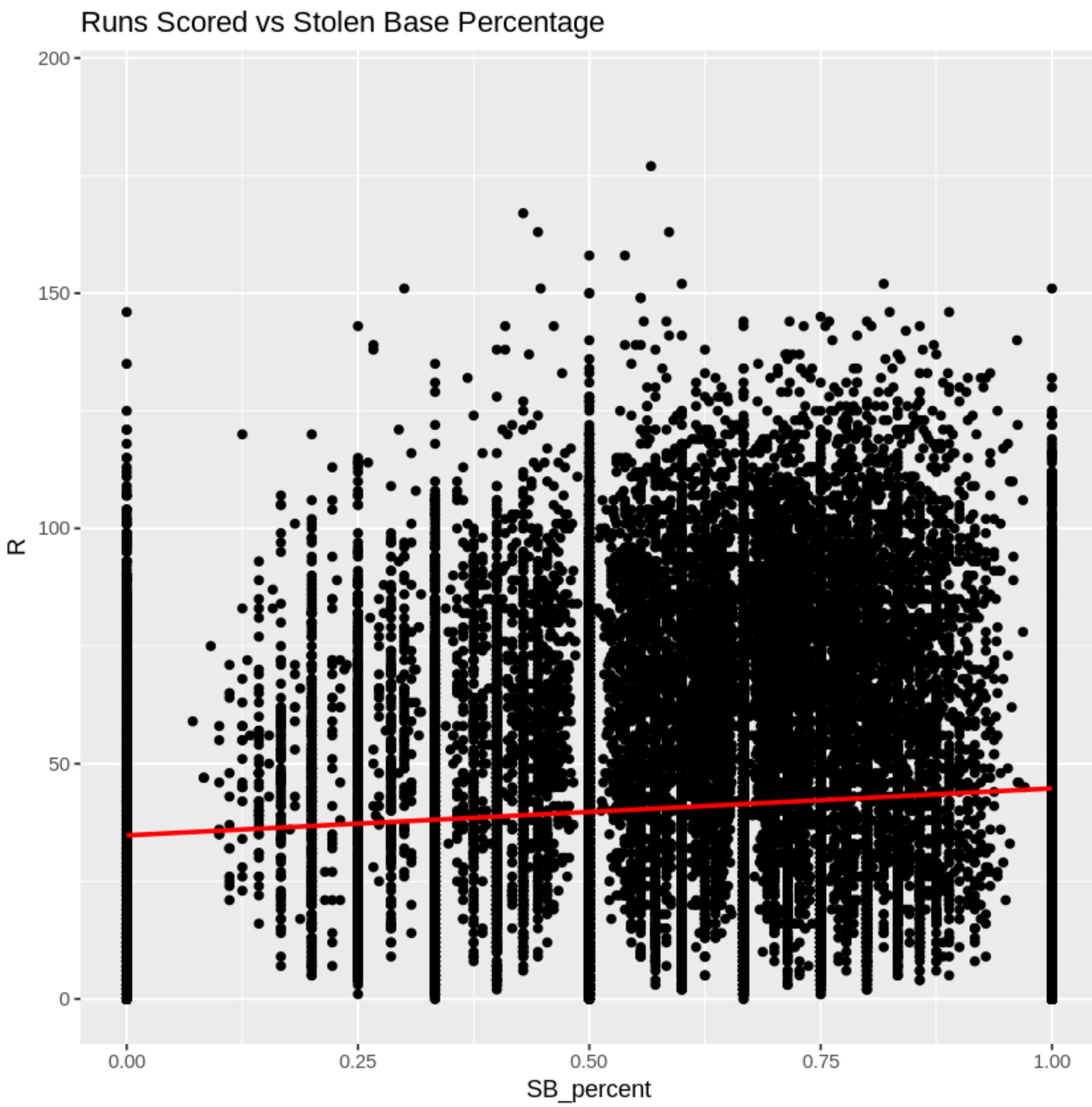
It is clear that there is a positive linear relationship between On-Base Percentage and Batting Average represented by the line `OBP = 1.015544BA + 0.052297`. We can see this is reasonable because we have an adjusted R^2 value of `0.8393`, which is pretty high. This model can be interpreted as: For every one unit increase in Batting Average, the On-Base Percentage of a batter increases by `1.015544`.

6.0.2 Stolen Base Percentage and Runs Scored: Is there a relationship between speed on the bases and scoring?

[ ]:
```
# Batting: Stolen Base Percentage and Runs Scored
# Stolen Base Percentage (SB_percent) = stolen bases / (stolen bases + caught␣
↪stealing)
batting_data$SB_percent <- batting_data$SB / (batting_data$SB + batting_data$CS)
```

```r
# Linear model
lm_sb_runs <- lm(R ~ SB_percent, data = batting_data)
summary(lm_sb_runs)

# Scatterplot with regression line
ggplot(batting_data, aes(x = SB_percent, y = R)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE, color="red") +
  labs(title = "Runs Scored vs Stolen Base Percentage")
```

```
Call:
lm(formula = R ~ SB_percent, data = batting_data)

Residuals:
    Min      1Q  Median      3Q     Max 
-44.742 -25.761  -6.039  21.886 136.583 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  34.7611     0.3576   97.22   <2e-16 ***
SB_percent    9.9810     0.5429   18.38   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 30.72 on 30949 degrees of freedom
  (70381 observations deleted due to missingness)
Multiple R-squared:  0.0108,	Adjusted R-squared:  0.01077 
F-statistic: 337.9 on 1 and 30949 DF,  p-value: < 2.2e-16


`geom_smooth()` using formula = 'y ~ x'
Warning message:
"Removed 70381 rows containing non-finite outside the scale range
(`stat_smooth()`)."
Warning message:
"Removed 70381 rows containing missing values or values outside the
scale range
(`geom_point()`)."
```



We can see this data has a significantly higher variance than the previous data. A linear model does not fit the data well since it is so scattered. We can see however that hitters with a relatively higher Stolen Base Percentage (greater than 50%) also scored more runs since many of the data points are clustered between 50% and 100% compared to the datapoints with a Stolen Base Percentage between 0% to 50%.

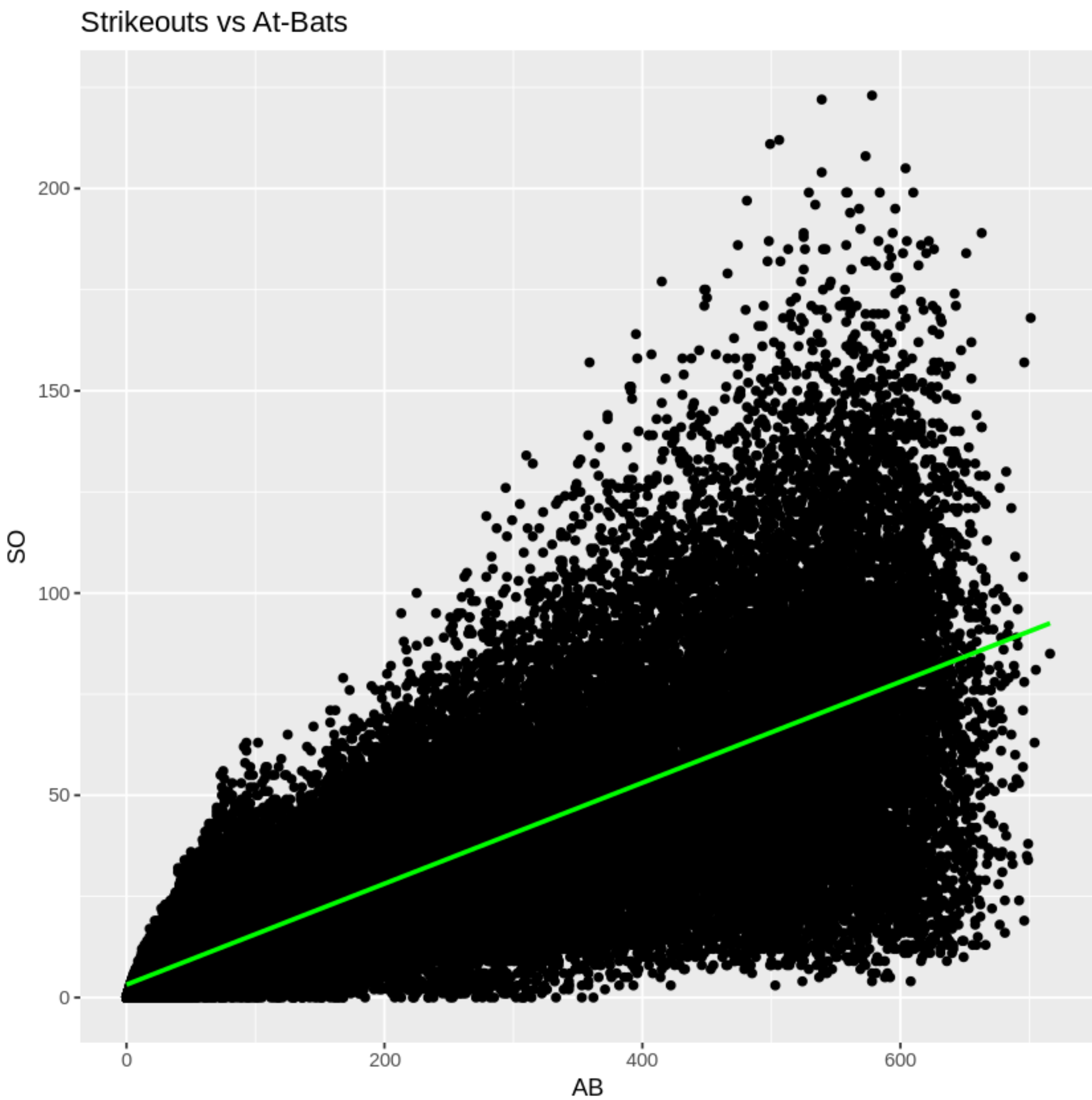
6.0.3 Strikeouts and At-Bats: A Simple Visualization of the Relationship (if any)

[ ]:
```
lm_so_ab <- lm(SO ~ AB, data = batting_data)
summary(lm_so_ab)

ggplot(batting_data, aes(x = AB, y = SO)) +
  geom_point() +
```

```r
  geom_smooth(method = "lm", se = FALSE, color="green") +
  labs(title = "Strikeouts vs At-Bats")
```

```
Call:
lm(formula = SO ~ AB, data = batting_data)

Residuals:
    Min      1Q  Median      3Q     Max 
-75.056  -3.557  -2.308   3.855 151.555 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept) 3.1830562  0.0700067   45.47   <2e-16 ***
AB          0.1247909  0.0002941  424.38   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 16.31 on 88343 degrees of freedom
  (12987 observations deleted due to missingness)
Multiple R-squared:  0.6709,	Adjusted R-squared:  0.6709 
F-statistic: 1.801e+05 on 1 and 88343 DF,  p-value: < 2.2e-16
```

```
`geom_smooth()` using formula = 'y ~ x'
Warning message:
"Removed 12987 rows containing non-finite outside the scale range
(`stat_smooth()`)."
Warning message:
"Removed 12987 rows containing missing values or values outside the
scale range
(`geom_point()`)."
```



The question that was asked here is whether the number of At-Bats has an effect on number of Strikeouts. From the above graph, we can see that the data points funnel out, which indicates a larger variance. This data is probably not best captured with a linear regression model (with an Adjusted R^2 value of `0.6709`). Regardless, it is easy to see there is a positive trend. If the lienar model was an accurate representation for the trend, we can interpret this as: For every one increase in At-Bats, the number of Strikeouts a batter will have will increase by `0.1247909`.

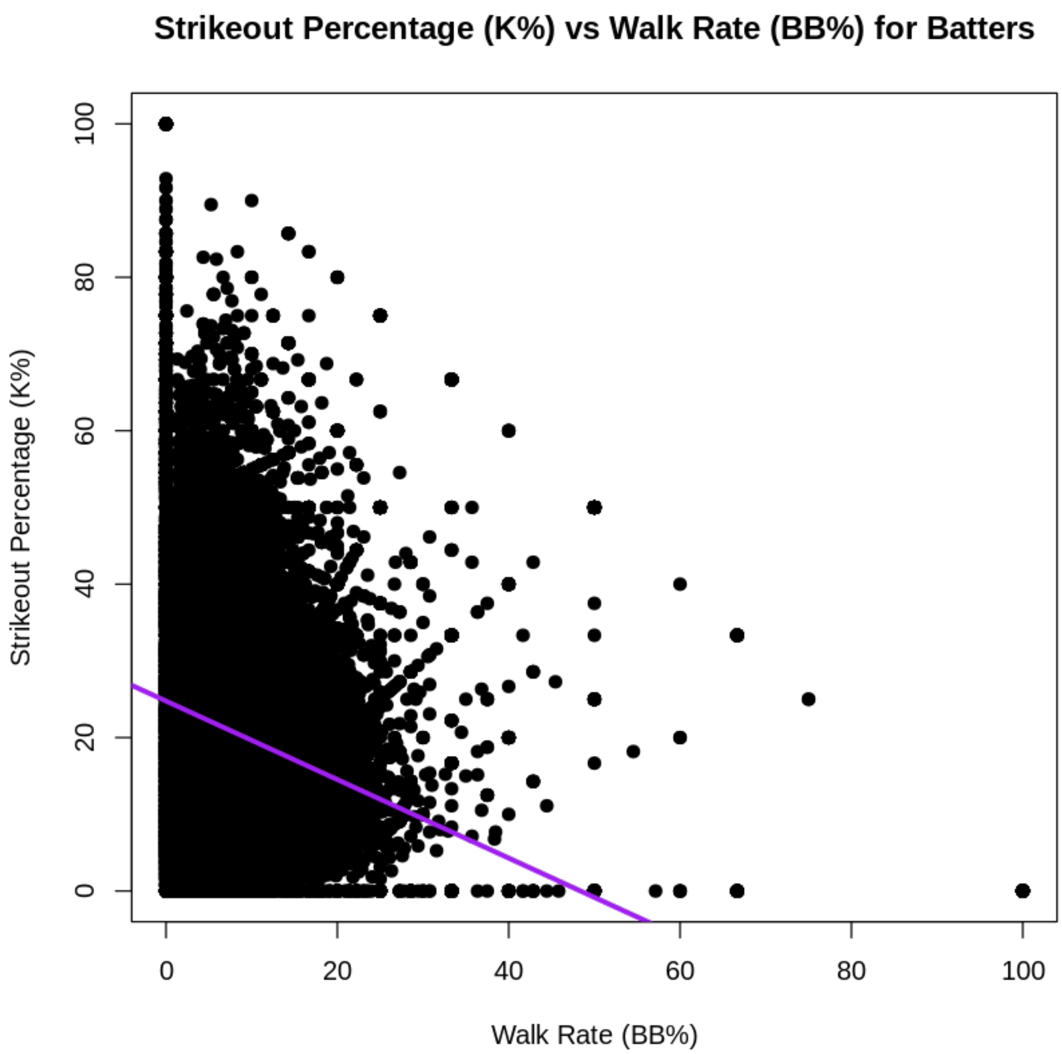
6.0.4 Strikeout Percentage and Walk Rate for Batters - compare later to pitchers

The purpose of this part is to determine if Walk Rate, or Walks (BB) in general, are related in any way to Strikeouts. This will be helpful for us to understand how Walks (BB) might affect a future model where we predict Strikeouts (SO). Note this part will also be useful for comparing Strikeout Percentage vs Walk Rate for Pitchers.

[ ]:
```
# Walk Rate (BB%)
at_bats = batting_data$AB
walks = batting_data$BB
plate_appearances = at_bats + walks # estimate plate appearances

## NOTE ##
# We estimate plate_appearances with hit by pitch (HBP) and sacrifice flies (SF)
# but this data was unavailable

walk_rate = (walks / plate_appearances) * 100

# Strikeout Percentage (K%)
strikeouts = batting_data$SO

strikeout_percentage = (strikeouts / plate_appearances) * 100

# Add the new columns to the dataset
batting_data$walk_rate <- walk_rate
batting_data$strikeout_percentage <- strikeout_percentage

# Linear model
lm_so_bb_batters <- lm(batting_data$strikeout_percentage ~␣
↪batting_data$walk_rate)
summary(lm_so_bb_batters)

# K% vs BB%
plot(batting_data$walk_rate, batting_data$strikeout_percentage,
     main = "Strikeout Percentage (K%) vs Walk Rate (BB%) for Batters",
     xlab = "Walk Rate (BB%)",
     ylab = "Strikeout Percentage (K%)",
     pch = 19,
     col = "black")
abline(lm_so_bb_batters, col="purple", lwd=3)
```

```
Call:
lm(formula = batting_data$strikeout_percentage ~ batting_data$walk_rate)

Residuals:
    Min      1Q  Median      3Q     Max
-24.765 -11.656  -4.657   5.856  75.235

Coefficients:
                        Estimate Std. Error t value Pr(>|t|)
(Intercept)            24.765076   0.092198   268.6   <2e-16 ***
batting_data$walk_rate -0.511807   0.008778   -58.3   <2e-16 ***
---
```

```
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 19.27 on 78838 degrees of freedom
  (22492 observations deleted due to missingness)
Multiple R-squared:  0.04134,	Adjusted R-squared:  0.04132
F-statistic:  3399 on 1 and 78838 DF,  p-value: < 2.2e-16
```



From the above graph, there is clear indication of a negative trend. This is what we expected because intuitively, a higher strikeout percentage is associated with a lower walk rate. In other words, the more a batter strikes out, the less opportunities a batter has to achieve a walk. This is logical because usually batters that strike out more become more picky and selective at the plate when they hit. It is important to note here that correlation does not imply causation. So in our example here, it does not mean that when a batter strikes out it causes them to get less walks.

There are other confounding variables to consider here. A good example is that At-Bats can result in a hit, which is neither a strikeout or a walk.

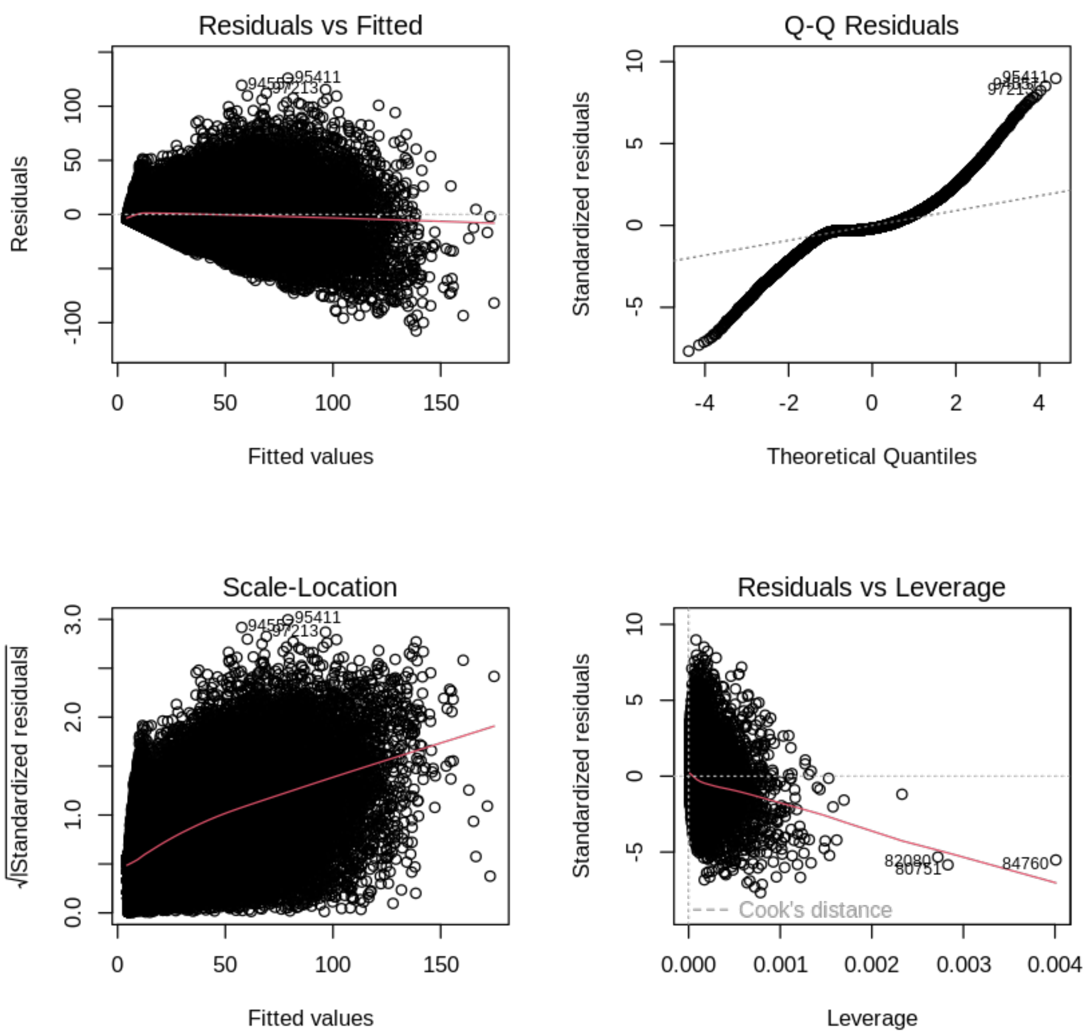
6.0.5 Predicting Strikeouts by a Batter

First, we tried fitting a linear model with only a few predictors. We checked the linear regression assumptions below. Then after looking into the model metrics (AIC, BIC, and Adjusted R^2), there seemed to be much room for improvement. So, in the following code, we tried to use RegSubsets to then select the best model with the lowest RSS. We also examine the AIC, BIC, and Adjusted R^2 metrics.

[ ]:
```
# Linear model with a few predictors (At-Bats, Homeruns, Walks)
batting_lm <- lm(SO ~ AB + HR + BB, data = batting_data)
summary(batting_lm)

# # Calculate model metrics - AIC, BIC, Adjusted R^2, MSPE (mean squared␣
 ↪prediction error)
aic_value <- AIC(batting_lm)
bic_value <- BIC(batting_lm)
adj_r_squared <- summary(batting_lm)$adj.r.squared

# Display the results
cat("AIC:", aic_value, "\n")
cat("BIC:", bic_value, "\n")
cat("Adjusted R-squared:", adj_r_squared, "\n")

# Plot Residuals vs Fitted, QQ Residuals, Scale-Location, and Residuals vs␣
 ↪Leverage
par(mfrow = c(2,2))
plot(batting_lm)
```

```
Call:
lm(formula = SO ~ AB + HR + BB, data = batting_data)

Residuals:
     Min       1Q   Median       3Q      Max
-107.600   -4.331   -2.590    4.255  125.922

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 4.303010   0.060572   71.04  < 2e-16 ***
AB          0.077345   0.000528  146.48  < 2e-16 ***
HR          1.777538   0.010872  163.49  < 2e-16 ***
BB          0.022158   0.004817    4.60 4.23e-06 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

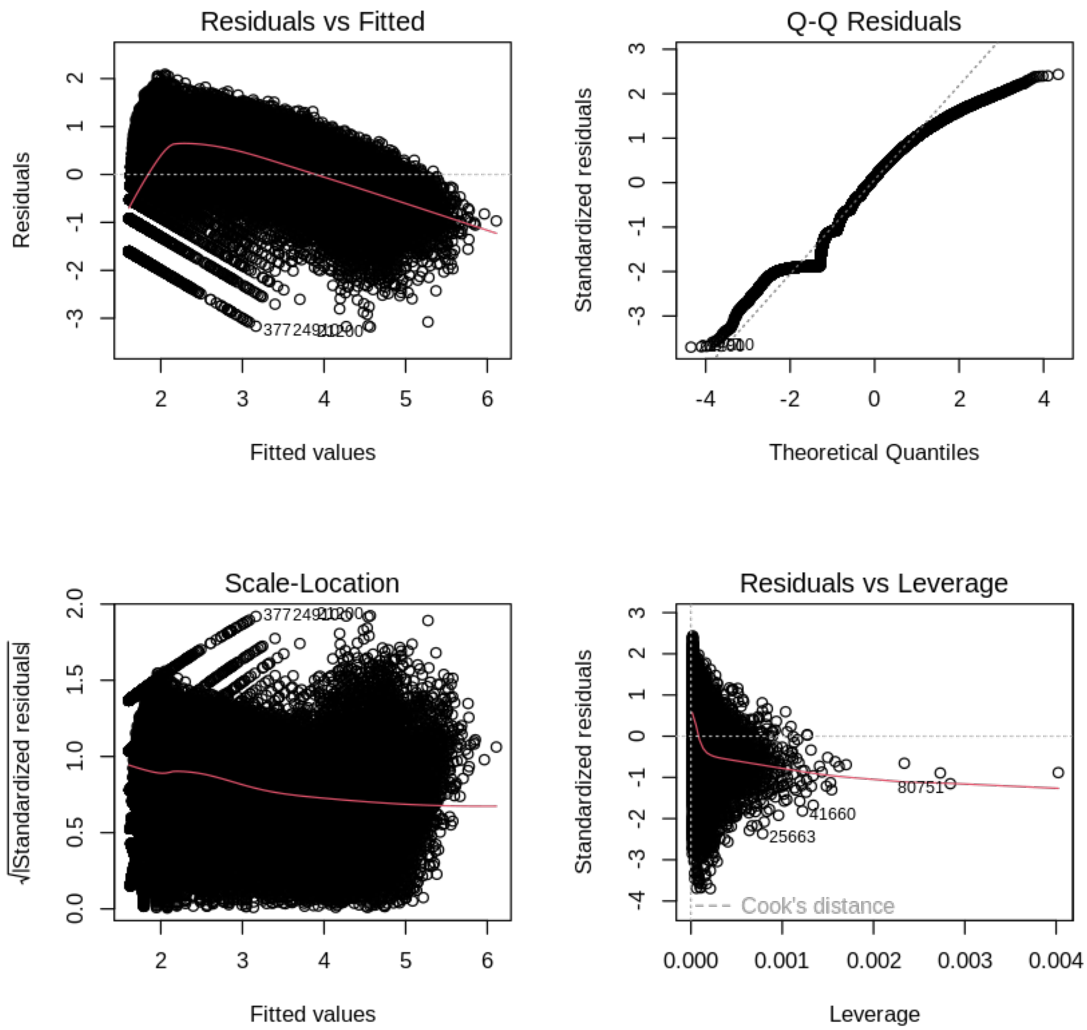
```
Residual standard error: 14.03 on 88341 degrees of freedom
  (12987 observations deleted due to missingness)
Multiple R-squared:  0.7565,	Adjusted R-squared:  0.7565
F-statistic: 9.15e+04 on 3 and 88341 DF,  p-value: < 2.2e-16


AIC: 717390.1
BIC: 717437.1
Adjusted R-squared: 0.7565197
```



[ ]:
```
# Try a log transformation of response
batting_data$log_SO <- log(batting_data$SO)
# Remove rows with NA or Inf values in log_SO
```

```r
batting_data_clean <- batting_data[!is.na(batting_data$log_SO) & !is.
 ↪infinite(batting_data$log_SO), ]
# Fit the linear model after cleaning
log_batting_lm <- lm(log_SO ~ AB + HR + BB, data = batting_data_clean)
summary(log_batting_lm)

# Calculate model metrics - AIC, BIC, Adjusted R^2, MSPE (mean squared␣
 ↪prediction error)
aic_value <- AIC(log_batting_lm)
bic_value <- BIC(log_batting_lm)
adj_r_squared <- summary(log_batting_lm)$adj.r.squared

# Display the results
cat("AIC:", aic_value, "\n")
cat("BIC:", bic_value, "\n")
cat("Adjusted R-squared:", adj_r_squared, "\n")

# Plot Residuals vs Fitted, QQ Residuals, Scale-Location, and Residuals vs␣
 ↪Leverage
par(mfrow = c(2,2))
plot(log_batting_lm)
```

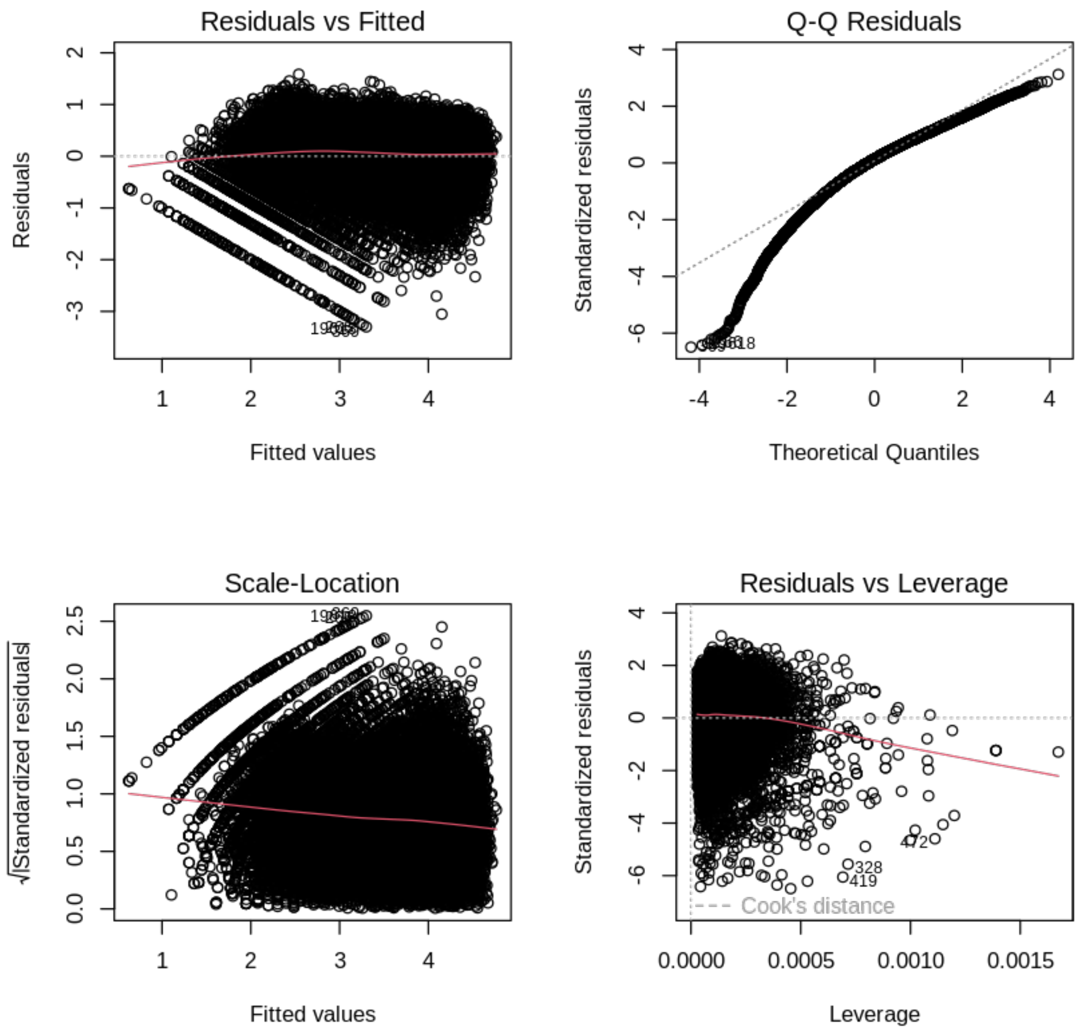
```
Call:
lm(formula = log_SO ~ AB + HR + BB, data = batting_data_clean)

Residuals:
    Min      1Q  Median      3Q     Max
-3.1805 -0.5591  0.1069  0.6716  2.0921

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 1.605e+00  4.382e-03  366.32   <2e-16 ***
AB          4.774e-03  3.321e-05  143.78   <2e-16 ***
HR          2.110e-02  6.671e-04   31.63   <2e-16 ***
BB          5.877e-04  2.954e-04    1.99   0.0466 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.8595 on 72973 degrees of freedom
Multiple R-squared:  0.593,	Adjusted R-squared:  0.593
F-statistic: 3.545e+04 on 3 and 72973 DF,  p-value: < 2.2e-16


AIC: 185014.6
BIC: 185060.6
Adjusted R-squared: 0.5930151
```




```
[ ]: # Try log transformation of response and predictors
batting_data$log_SO <- log(batting_data$SO)
batting_data$log_AB <- log(batting_data$AB)
batting_data$log_HR <- log(batting_data$HR)
batting_data$log_BB <- log(batting_data$BB)
# Remove rows with NA or Inf in any of the transformed columns
batting_data_clean <- batting_data[!is.na(batting_data$log_SO) & !is.
↪infinite(batting_data$log_SO) &
                                    !is.na(batting_data$log_AB) & !is.
↪infinite(batting_data$log_AB) &
                                    !is.na(batting_data$log_HR) & !is.
↪infinite(batting_data$log_HR) &
```

```r
                                    !is.na(batting_data$log_BB) & !is.
 ↪infinite(batting_data$log_BB), ]
# Fit the linear model after cleaning
logAll_batting_lm <- lm(log_SO ~ log_AB + log_HR + log_BB, data =␣
 ↪batting_data_clean)
summary(logAll_batting_lm)

# Calculate model metrics - AIC, BIC, Adjusted R^2, MSPE (mean squared␣
 ↪prediction error)
aic_value <- AIC(logAll_batting_lm)
bic_value <- BIC(logAll_batting_lm)
adj_r_squared <- summary(logAll_batting_lm)$adj.r.squared

# Display the results
cat("AIC:", aic_value, "\n")
cat("BIC:", bic_value, "\n")
cat("Adjusted R-squared:", adj_r_squared, "\n")

# Plot Residuals vs Fitted, QQ Residuals, Scale-Location, and Residuals vs␣
 ↪Leverage
par(mfrow = c(2,2))
plot(logAll_batting_lm)
```

```
Call:
lm(formula = log_SO ~ log_AB + log_HR + log_BB, data = batting_data_clean)

Residuals:
    Min      1Q  Median      3Q     Max
-3.3009 -0.2692  0.0697  0.3475  1.5866

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 0.289266   0.022945  12.607   <2e-16 ***
log_AB      0.488084   0.006249  78.111   <2e-16 ***
log_HR      0.268902   0.003302  81.429   <2e-16 ***
log_BB      0.045104   0.005027   8.973   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.5084 on 36038 degrees of freedom
Multiple R-squared:  0.6504,	Adjusted R-squared:  0.6504
F-statistic: 2.235e+04 on 3 and 36038 DF,  p-value: < 2.2e-16


AIC: 53521.33
BIC: 53563.79
Adjusted R-squared: 0.650355
```



The first Residuals vs Fitted plot shows a funnel shape, which indicates heterscedasticity, a violation of the non-constant variance assumption. To remedy this, we tried to log transform the response variable, but that lowered Adjusted R^2 by `0.1635` and also increased both `AIC` and `BIC` by a significant amount. So instead, we then tried to transform both the response and predictors. This only lowered Adjusted R^2 by `0.1061` from the original model, but then significantly lowered `AIC` by `663,868.77` and `BIC` by `663,873.31`. Both of the metrics improved here, with the sacrifice of a slightly lower Adjusted R^2.

After looking into the assumptions, we continued to try and fit a higher dimensional model, and evaluate which predictors are signifcant based on the 3 metrics: `AIC`, `BIC`, and Adjusted R^2. Below, we used regsubsets to develop models with different combinations of predictors, and determine models with the lowest RSS (residual sum of squares).

[ ]:
```
# Predicting Number of Strikeouts a Batter Might Have in a Season

# Define relevant predictors
predictors <- c("AB", "G", "BB", "HR", "R", "X2B", "X3B", "RBI", "H", "GIDP")
# At-Bats, Games Played, Walks, Homeruns, Runs Scored, Double, Triples, Runs␣
↪Batted In,
# Hits, Grounded into Double Play

# Filter dataset to include only relevant predictors and response variable
batting_data_filtered <- batting_data[, c("SO", predictors)]

# Perform all subset selection
all_models <- regsubsets(SO ~ ., data = batting_data_filtered, nvmax =␣
↪length(predictors)) # from leaps library

# Extract model summaries
summary_all <- summary(all_models)

# Visualize regsubsets summary metrics
par(mfrow = c(2, 2))
# Plot RSS
plot(summary_all$rss ,xlab = "Number of Variables", ylab = "RSS", type = "l")
# Plot Adjusted R^2
plot(summary_all$adjr2 ,xlab = "Number of Variables", ylab = "Adjusted R^2",␣
↪type = "l")
points(11, summary_all$adjr2[11], col = "red", cex = 2, pch = 20)
# Plot Mallow's CP, balance of model fit and complexity, want small & close to␣
↪num. of predictors
plot(summary_all$cp, xlab = "Number of Variables", ylab = "Cp", type = "l")
points(which.min(summary_all$cp), summary_all$cp[which.min(summary_all$cp)],
        col = "red", cex = 2, pch = 20)
# Plot BIC
plot(summary_all$bic, xlab = "Number of Variables ", ylab = "BIC", type = "l")
points(which.min(summary_all$bic), summary_all$bic[which.min(summary_all$bic)],
       col = "red", cex = 2, pch = 20)

# Determine max and mins relative to each metric
which.max(summary_all$adjr2) # model 10
which.min(summary_all$cp) # model 10
which.min(summary_all$bic) # model 8

# Initialize results dataframe
model_results <- data.frame(
  Predictors = character(),
  AIC = numeric(),
  BIC = numeric(),
  Adj_R2 = numeric(),
```

```r
  stringsAsFactors = FALSE
)

# Loop through models to compute metrics
for (i in 1:nrow(summary_all$which)) {
  # Get selected predictors
  selected_predictors <- names(batting_data_filtered)[summary_all$which[i, ]]

  # Create formula
  formula <- as.formula(paste("SO ~", paste(selected_predictors, collapse = " + ")))

  # Fit the model
  model <- lm(formula, data = batting_data_filtered)

  # Compute metrics
  aic_value <- AIC(model)
  bic_value <- BIC(model)
  adj_r_squared <- summary(model)$adj.r.squared

  # Save results
  model_results <- rbind(
    model_results,
    data.frame(
      Predictors = paste(selected_predictors, collapse = ", "),
      AIC = aic_value,
      BIC = bic_value,
      Adj_R2 = adj_r_squared
    )
  )
}

# Display model results
model_results

# Display the best model for each metric
min_aic <- min(model_results$AIC)
min_aic_model <- model_results[model_results$AIC == min_aic, ]
min_aic
min_aic_model

min_bic <- min(model_results$BIC)
min_bic_model <- model_results[model_results$BIC == min_bic, ]
min_bic
min_bic_model

max_adj_r2 <- max(model_results$Adj_R2)
```

```
max_adj_r2_model <- model_results[model_results$Adj_R2 == max_adj_r2, ]
max_adj_r2
max_adj_r2_model
```

10

10

8

```
Warning message in model.matrix.default(mt, mf, contrasts):
"the response appeared on the right-hand side and was dropped"
Warning message in model.matrix.default(mt, mf, contrasts):
"problem with term 1 in model.matrix: no columns are assigned"
Warning message in model.matrix.default(mt, mf, contrasts):
"the response appeared on the right-hand side and was dropped"
Warning message in model.matrix.default(mt, mf, contrasts):
"problem with term 1 in model.matrix: no columns are assigned"
Warning message in model.matrix.default(mt, mf, contrasts):
"the response appeared on the right-hand side and was dropped"
Warning message in model.matrix.default(mt, mf, contrasts):
"problem with term 1 in model.matrix: no columns are assigned"
Warning message in model.matrix.default(mt, mf, contrasts):
"the response appeared on the right-hand side and was dropped"
Warning message in model.matrix.default(mt, mf, contrasts):
"problem with term 1 in model.matrix: no columns are assigned"
Warning message in model.matrix.default(mt, mf, contrasts):
"the response appeared on the right-hand side and was dropped"
Warning message in model.matrix.default(mt, mf, contrasts):
"problem with term 1 in model.matrix: no columns are assigned"
Warning message in model.matrix.default(mt, mf, contrasts):
"the response appeared on the right-hand side and was dropped"
Warning message in model.matrix.default(mt, mf, contrasts):
"problem with term 1 in model.matrix: no columns are assigned"
Warning message in model.matrix.default(mt, mf, contrasts):
"the response appeared on the right-hand side and was dropped"
Warning message in model.matrix.default(mt, mf, contrasts):
"problem with term 1 in model.matrix: no columns are assigned"
Warning message in model.matrix.default(mt, mf, contrasts):
"the response appeared on the right-hand side and was dropped"
Warning message in model.matrix.default(mt, mf, contrasts):
"problem with term 1 in model.matrix: no columns are assigned"
Warning message in model.matrix.default(mt, mf, contrasts):
"the response appeared on the right-hand side and was dropped"
Warning message in model.matrix.default(mt, mf, contrasts):
```

```
"problem with term 1 in model.matrix: no columns are assigned"
```

A data.frame: 10 × 4

Predictors <chr>	AIC <dbl>	BIC <dbl>	Adj_R2 <dbl>
SO, AB	744007.9	744036.1	0.6709030
SO, AB, HR	717409.3	717446.8	0.7564642
SO, AB, HR, H	690856.0	690903.0	0.8196878
SO, AB, HR, X2B, H	688275.4	688331.7	0.8248806
SO, AB, HR, X2B, H, GIDP	545449.3	545513.4	0.8492198
SO, AB, HR, X2B, RBI, H, GIDP	544697.2	544770.5	0.8508326
SO, AB, G, HR, X2B, RBI, H, GIDP	544564.4	544646.8	0.8511175
SO, AB, G, BB, HR, X2B, RBI, H, GIDP	544539.9	544631.4	0.8511716
SO, AB, G, BB, HR, X2B, X3B, RBI, H, GIDP	544532.5	544633.2	0.8511894
SO, AB, G, BB, HR, R, X2B, X3B, RBI, H, GIDP	544523.0	544632.9	0.8512118

544522.980179923

A data.frame: 1 × 4

	Predictors <chr>	AIC <dbl>	BIC <dbl>	Adj_R2 <dbl>
10	SO, AB, G, BB, HR, R, X2B, X3B, RBI, H, GIDP	544523	544632.9	0.8512118

544631.448698064

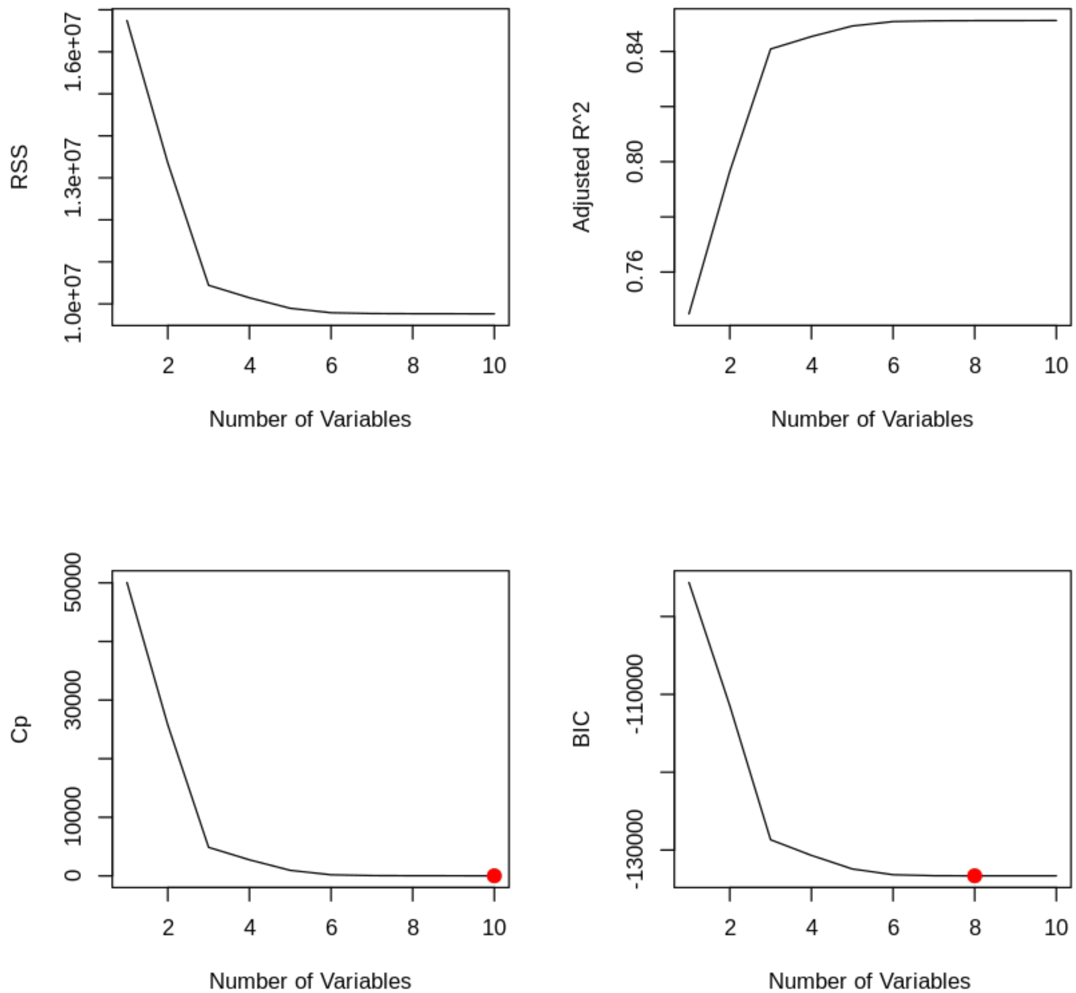
A data.frame: 1 × 4

	Predictors <chr>	AIC <dbl>	BIC <dbl>	Adj_R2 <dbl>
8	SO, AB, G, BB, HR, X2B, RBI, H, GIDP	544539.9	544631.4	0.8511716

0.851211813297913

A data.frame: 1 × 4

	Predictors <chr>	AIC <dbl>	BIC <dbl>	Adj_R2 <dbl>
10	SO, AB, G, BB, HR, R, X2B, X3B, RBI, H, GIDP	544523	544632.9	0.8512118



From the above results, the best model according to `AIC` and Adjusted R^2 is model 10, which contains all 10 predictors:

1. At-Bats
2. Games Played
3. Walks
4. Homeruns
5. Runs Scored
6. Doubles
7. Triples
8. Runs Batted In
9. Hits
10. Grounded into Double Play

This model had both the lowest AIC value and highest Adjusted R^2 value.

However, the best model according to BIC is model 8, which contains the following predictors:

1. At-Bats
2. Games Played
3. Walks
4. Homeruns
5. Doubles
6. Runs Batted In
7. Hits
8. Grounded into Double Play

This model had the lowest BIC of the models that were produced using the regsubsets library.

7 2. Pitching

7.0.1 Strikeout Percentage and Walk Rate: Evaluate basic pitching metrics

We want to calculate strikeout percentage and walk rate and find out if there is any relationship between these simple pitching metrics.

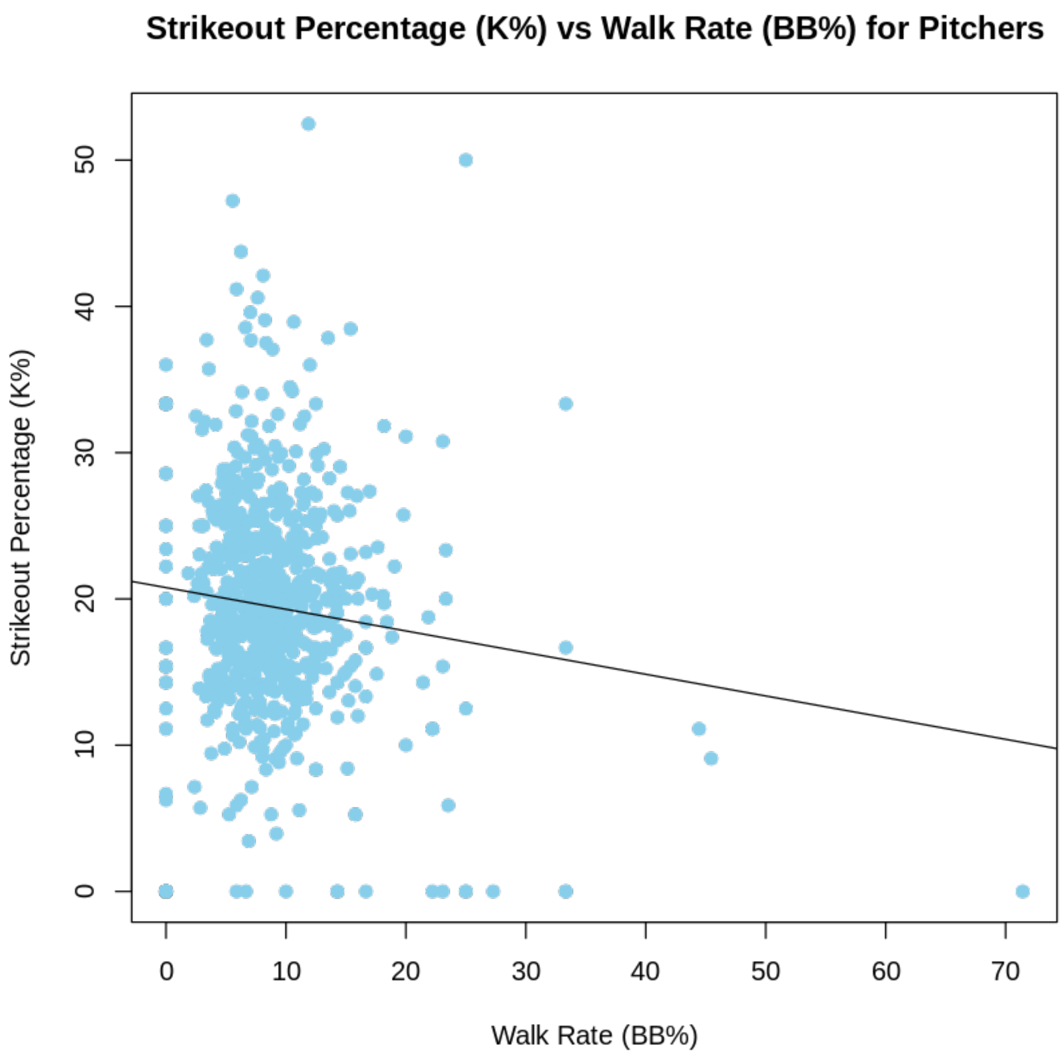
[ ]:
```
# Pitching: Strikeout Percentage and Walk Rate, ERA

# Necessary data
innings_pitched = pitching_data$IPouts / 3 # Convert outs to innings, outs␣
 ↪pitched = innings*3
batters_faced = pitching_data$BFP # Batters faced
walks_allowed = pitching_data$BB # Walks Allowed
strikeouts = pitching_data$SO # Strikeouts by a Pitcher

# Walk Rate (BB%)
walk_rate = ifelse(batters_faced > 0, (walks_allowed / batters_faced) * 100, NA)

# Strikeout Percentage (SO%)
strikeout_percentage = ifelse(batters_faced > 0, (strikeouts / batters_faced) *␣
 ↪100, NA)

# Add columns to the dataset
pitching_data$walk_rate <- walk_rate
pitching_data$strikeout_percentage <- strikeout_percentage

# Remove rows with NA or problematic values (NA, inf, or -inf)
pitching_data_clean1 <- na.omit(pitching_data) # Remove rows with NA
pitching_data_clean1 <- pitching_data_clean1[is.
 ↪finite(pitching_data_clean1$walk_rate) &
                                          is.
 ↪finite(pitching_data_clean1$strikeout_percentage), ]
# View cleaned data
```

```r
head(pitching_data_clean1, 5)

# K% vs BB% Plot
plot(pitching_data_clean1$walk_rate, pitching_data_clean1$strikeout_percentage,
     main = "Strikeout Percentage (K%) vs Walk Rate (BB%) for Pitchers",
     xlab = "Walk Rate (BB%)",
     ylab = "Strikeout Percentage (K%)",
     pch = 19,
     col = "skyblue")

lm_so_bb_pitchers <- lm(pitching_data_clean1$strikeout_percentage ~ pitching_data_clean1$walk_rate)
abline(lm_so_bb_pitchers)

summary(lm_so_bb_pitchers)
```

A data.frame: 5 × 32

	playerID	yearID	stint	teamID	lgID	W	L	G	GS
	<chr>	<int>	<int>	<chr>	<chr>	<int>	<int>	<int>	<int
42585	abadfe01	2014	1	OAK	AL	2	4	69	0
42586	aceveal01	2014	1	NYA	AL	1	2	10	0
42587	achteaj01	2014	1	MIN	AL	1	0	7	0
42588	adamsau01	2014	1	CLE	AL	0	0	6	0
42589	adamsmi03	2014	1	PHI	NL	2	1	22	0

```
Call:
lm(formula = pitching_data_clean1$strikeout_percentage ~ pitching_data_clean1$walk_rate)

Residuals:
    Min      1Q  Median      3Q     Max
-20.772  -4.321  -0.421   4.228  33.464

Coefficients:
                               Estimate Std. Error t value Pr(>|t|)
(Intercept)                    20.77155    0.51324  40.471  < 2e-16 ***
pitching_data_clean1$walk_rate -0.14818    0.04917  -3.014  0.00267 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 7.491 on 740 degrees of freedom
Multiple R-squared:  0.01212,       Adjusted R-squared:  0.01079
F-statistic: 9.082 on 1 and 740 DF,  p-value: 0.00267
```



First, we note that there are a significantly smaller number of data points on the graph. This is because when calculating the percentages, we had to ensure that the denominators were not missing values (`NA` or `inf`). To expand on this, we calculated `Walk Rate` by dividing walks allowed by a pitcher by the number of batters the pitcher has faced. If the number of batters faced is not greater than zero (meaning it is either zero or a missing value `NA`), then the resulting value for percentage becomes `NA`. We clean the data set to omit the rows with `NA` values. Similarly, we repeat the process for `Strikeout Percentage`.

After cleaning the data, we graphed `Strikeout Percentage (K%)` against `Walk Rate (BB%)` which is shown above. We see that there is a cluster of points, which is probably not best represented using a linear model since the Adjusted R^2 of the model is very low (`0.01079`). A non-linear model would most likely be more sufficient since the linear model here only captures 1% of the variance in the data (after cleaning). In general though, we observe that most data points are clustered with a `Walk Rate` between 0% and 20%, ranging mostly within a `Strikeout Rate` between 10% and

30%. It is hard to say here that there is any correlation between `Walk Rate` and `Strikeout Rate`.

7.0.2 Strikeout to Walk Ratio Relative to Runs Allowed: Evaluate overall pitcher performance

Using the above section, we create a new column for the Strikeout-to-Walk-Ratio in order to evaluate the performance of pitchers relative to the number of runs a pitcher allows.

[ ]:
```
# Create Strikeout-to-Walk Ratio Column (using above section)
pitching_data$K_BB_ratio <- pitching_data$SO / pitching_data$BB
```

[ ]:
```
# Check counts
sum(is.na(pitching_data$K_BB_ratio)) # Count of NAs in K_BB_ratio
sum(is.na(pitching_data$R)) # Count of NAs in R
sum(is.infinite(pitching_data$K_BB_ratio)) # Count of Inf in K_BB_ratio

# Clean data where BB is zero (results in inf)
pitching_data$K_BB_ratio[is.infinite(pitching_data$K_BB_ratio)] <- NA
# Remove rows with missing or invalid values
pitching_data_clean2 <- na.omit(pitching_data[, c("K_BB_ratio", "R")])
head(pitching_data_clean2)

# Re-check counts
sum(is.na(pitching_data_clean2$K_BB_ratio)) # Count of NAs in K_BB_ratio
sum(is.na(pitching_data_clean2$R)) # Count of NAs in R
sum(is.infinite(pitching_data_clean2$K_BB_ratio)) # Count of Inf in K_BB_ratio
```

505

0

638

A data.frame: 6 × 2

	K_BB_ratio <dbl>	R <int>
1	0.09090909	42
2	0.35135135	292
4	0.48387097	257
5	0.00000000	21
7	0.33333333	30
8	0.80952381	243

0

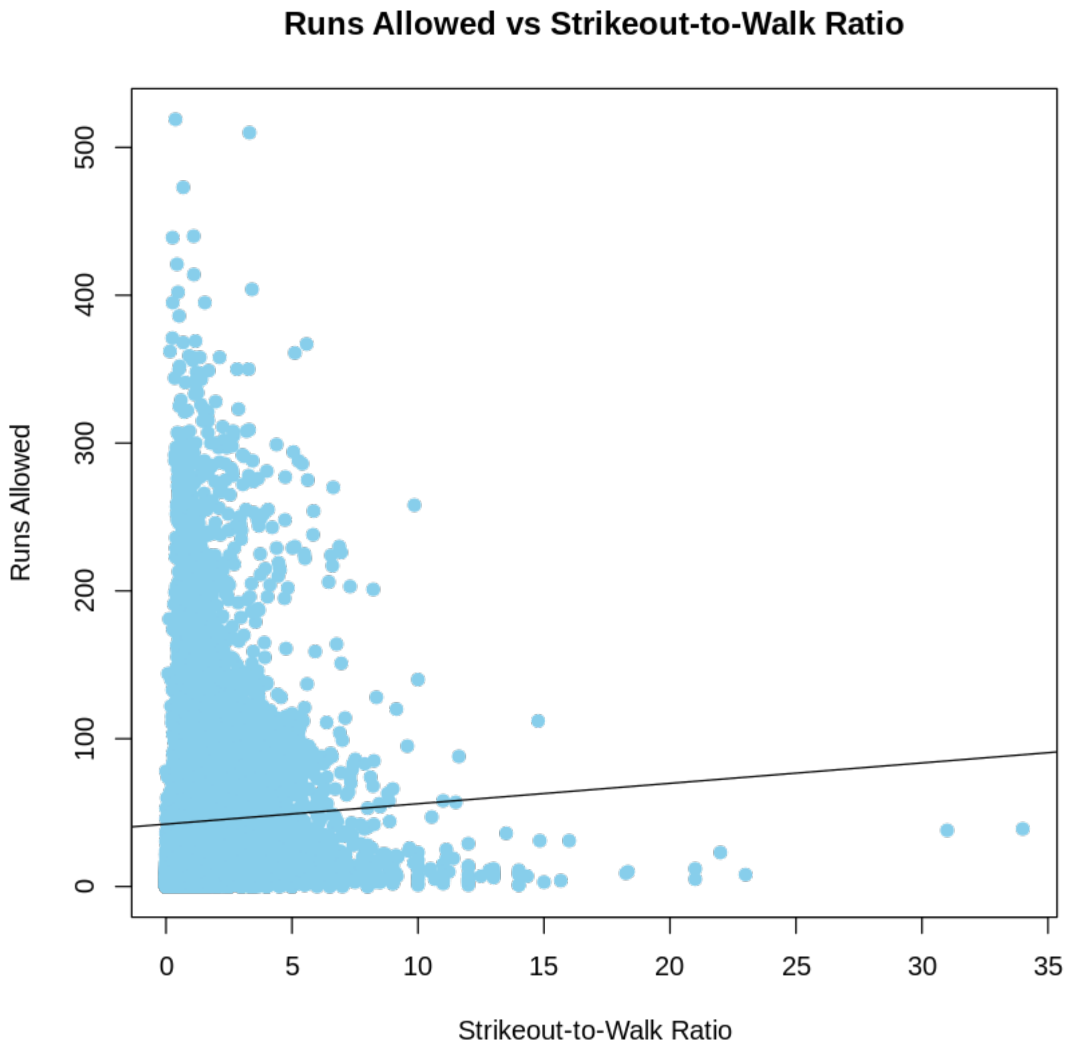
0

0

[ ]:
```
# Pitching: Strikeout-to-Walk Ratio

# Strikeout-to-Walk Ratio
lm_k_bb <- lm(R ~ K_BB_ratio, data = pitching_data_clean2)
```

```r
summary(lm_k_bb)

plot(pitching_data_clean2$K_BB_ratio, pitching_data_clean2$R,
     main = "Runs Allowed vs Strikeout-to-Walk Ratio",
     xlab = "Strikeout-to-Walk Ratio",
     ylab = "Runs Allowed",
     pch = 19,
     col = "skyblue")
abline(lm_k_bb)
```

```
Call:
lm(formula = R ~ K_BB_ratio, data = pitching_data_clean2)

Residuals:
   Min     1Q Median     3Q    Max
-66.23 -32.42 -14.87  25.06 476.25

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  42.2356     0.3504 120.523  < 2e-16 ***
K_BB_ratio    1.3805     0.1774   7.781 7.38e-15 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 43.45 on 42994 degrees of freedom
Multiple R-squared:  0.001406,	Adjusted R-squared:  0.001383
F-statistic: 60.54 on 1 and 42994 DF,  p-value: 7.376e-15
```



Similary, we cleaned the data to correctly calculate the `Strikeout-to-Walk Ratio` so there are not as many data points as in the original pitching dataset.Then we are able to graph `Runs Allowed` against `Strikeout-to-Walk Ratio`. We expected for a higher `Strikeout-to-Walk Ratio` to result in a lower number of `Runs Allowed`. We can see some data points that do follow this intuition. Much of the data is concentrated within a `Strikeout-to-Walk Ratio` of 0 and 5, where there are higher numbers of `Runs Allowed`. This follows the inverse of our expectation. More people allowed on base means a higher chance for base runners to score a run.

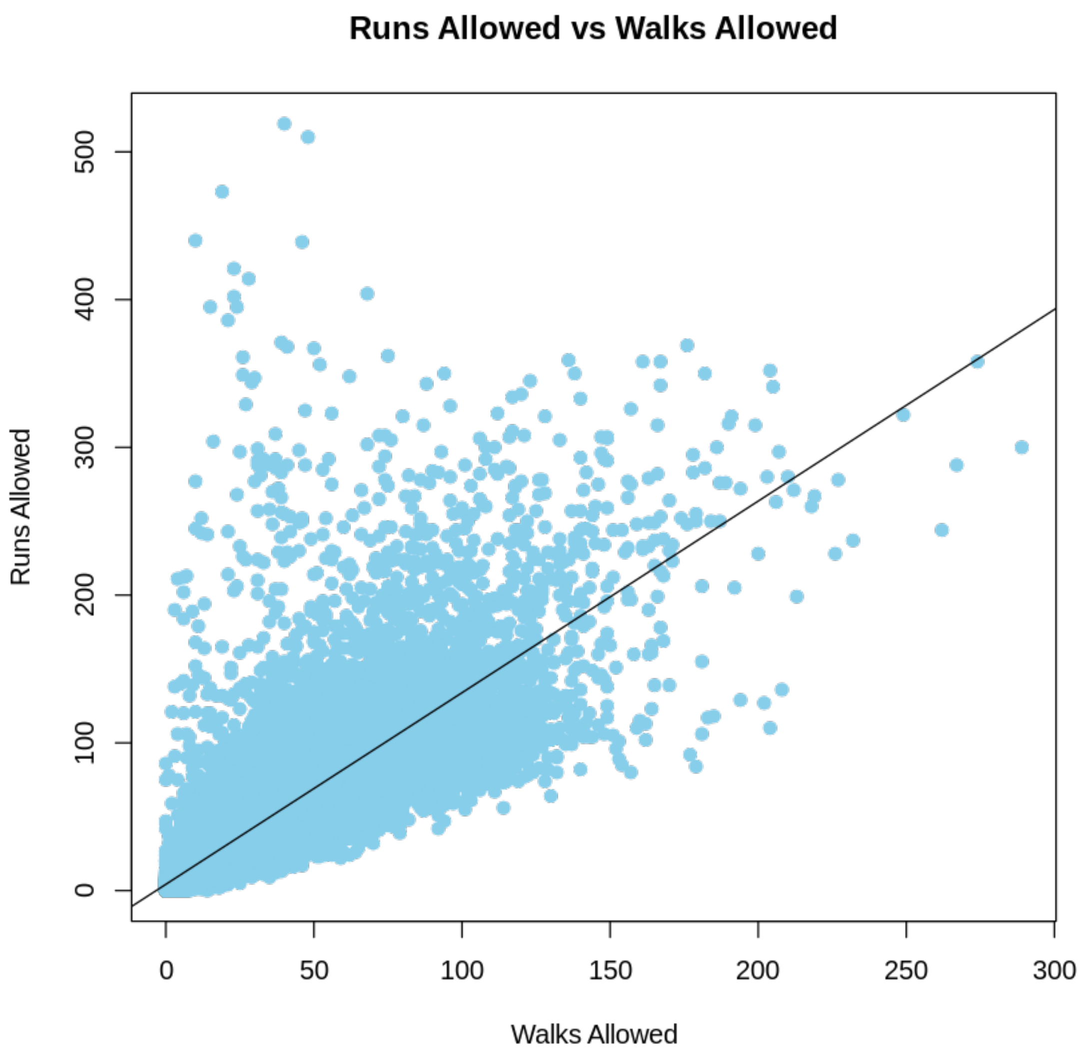
It is clear a linear model does not fit. A model that would best fit the data is probably an inverse exponential function, or an exponential decay.

7.0.3 Runs Allowed vs Walks: Does limiting number of walks decrease runs allowed?

We can confirm that the above graph makes sense by looking at the relationship between Runs Allowed (R) and Walks Allowed (BB) by pitchers.

[ ]:
```r
# Pitching: Runs allowed vs Walks
lm_runs_walks <- lm(R ~ BB, data = pitching_data)
summary(lm_runs_walks)

# Plot graph (R vs BB)
plot(pitching_data$BB, pitching_data$R,
     main = "Runs Allowed vs Walks Allowed",
     xlab = "Walks Allowed",
     ylab = "Runs Allowed",
     pch = 19,
     col = "skyblue")
abline(lm_runs_walks)
```

```
Call:
lm(formula = R ~ BB, data = pitching_data)

Residuals:
    Min      1Q  Median      3Q     Max
-158.84   -8.83   -3.16    4.76  462.86

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 4.266236   0.161685   26.39   <2e-16 ***
BB          1.296938   0.003913  331.47   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 23.25 on 44137 degrees of freedom
Multiple R-squared:  0.7134,	Adjusted R-squared:  0.7134
F-statistic: 1.099e+05 on 1 and 44137 DF,  p-value: < 2.2e-16
```



We can see that the linear model: `Runs Allowed`$= 4.266236 + 1.296938$ *`Walks Allowed` captures the positive linear trend pretty well, with an Adjusted R^2 value of `0.7134`, which means about 71% of the variation is captured by the linear model. So it is reasonable to say that number of `Walks Allowed` increases the number of `Runs Allowed` by a pitcher. This makes sense why pitchers want to avoid walking batters.

7.0.4 Predicting Strikeouts by a Pitcher

Since walks and strikeout are so vital to the performance of a pitcher (as seen from the previous sections), we decided to try and predict the number of strikeouts by a pitcher. If we can determine the number of strikeouts by a pitcher, it is likely this pitcher is really good, and could maybe be associated with a higher salary.

[ ]:

```r
# Predicting Number of Strikeouts a Pitcher Might Have in a Season

# Define relevant predictors
predictors <- c("G", "GS", "IPouts", "BFP", "ERA", "BAOpp", "HR", "BB", "CG",␣
↪"SHO")
# Games Played, Games Started, Outs Pitched, Batters Faced by Pitcher, Earned␣
↪Run Average,
# Batting Average Against, Homeruns Allowed, Walks Allowed, Complete Games,␣
↪Shutouts

# Filter dataset to include only relevant predictors and response variable
pitching_data_filtered <- pitching_data[, c("SO", predictors)]

# Perform all subset selection
all_models <- regsubsets(SO ~ ., data = pitching_data_filtered, nvmax =␣
↪length(predictors)) # from leaps library

# Extract model summaries
summary_all <- summary(all_models)

# Visualize regsubsets summary metrics
par(mfrow = c(2, 2))
# Plot RSS
plot(summary_all$rss ,xlab = "Number of Variables", ylab = "RSS", type = "l")
# Plot Adjusted R^2
plot(summary_all$adjr2 ,xlab = "Number of Variables", ylab = "Adjusted R^2",␣
↪type = "l")
points(11, summary_all$adjr2[11], col = "red", cex = 2, pch = 20)
# Plot Mallow's CP, balance of model fit and complexity, want small & close to␣
↪num. of predictors
plot(summary_all$cp, xlab = "Number of Variables", ylab = "Cp", type = "l")
points (which.min(summary_all$cp), summary_all$cp[which.min(summary_all$cp)],
        col = "red", cex = 2, pch = 20)
# Plot BIC
plot(summary_all$bic, xlab = "Number of Variables ", ylab = "BIC", type = "l")
points(which.min(summary_all$bic), summary_all$bic[which.min(summary_all$bic)],
       col = "red", cex = 2, pch = 20)

# Determine max and mins relative to each metric
which.max(summary_all$adjr2) # model 9
which.min(summary_all$cp) # model 9
which.min(summary_all$bic) # model 9

# Initialize results dataframe
model_results <- data.frame(
  Predictors = character(),
  AIC = numeric(),
```

```r
  BIC = numeric(),
  Adj_R2 = numeric(),
  stringsAsFactors = FALSE
)

# Loop through models to compute metrics
for (i in 1:nrow(summary_all$which)) {
  # Get selected predictors
  selected_predictors <- names(pitching_data_filtered)[summary_all$which[i, ]]

  # Create formula
  formula <- as.formula(paste("SO ~", paste(selected_predictors, collapse = " + 
 ↪")))

  # Fit the model
  model <- lm(formula, data = pitching_data_filtered)

  # Compute metrics
  aic_value <- AIC(model)
  bic_value <- BIC(model)
  adj_r_squared <- summary(model)$adj.r.squared

  # Save results
  model_results <- rbind(
    model_results,
    data.frame(
      Predictors = paste(selected_predictors, collapse = ", "),
      AIC = aic_value,
      BIC = bic_value,
      Adj_R2 = adj_r_squared
    )
  )
}

# Display model results
model_results

# Display the best model for each metric
min_aic <- min(model_results$AIC)
min_aic_model <- model_results[model_results$AIC == min_aic, ]
min_aic
min_aic_model

min_bic <- min(model_results$BIC)
min_bic_model <- model_results[model_results$BIC == min_bic, ]
min_bic
min_bic_model
```

```r
max_adj_r2 <- max(model_results$Adj_R2)
max_adj_r2_model <- model_results[model_results$Adj_R2 == max_adj_r2, ]
max_adj_r2
max_adj_r2_model
```

9

9

9

```
Warning message in model.matrix.default(mt, mf, contrasts):
"the response appeared on the right-hand side and was dropped"
Warning message in model.matrix.default(mt, mf, contrasts):
"problem with term 1 in model.matrix: no columns are assigned"
Warning message in model.matrix.default(mt, mf, contrasts):
"the response appeared on the right-hand side and was dropped"
Warning message in model.matrix.default(mt, mf, contrasts):
"problem with term 1 in model.matrix: no columns are assigned"
Warning message in model.matrix.default(mt, mf, contrasts):
"the response appeared on the right-hand side and was dropped"
Warning message in model.matrix.default(mt, mf, contrasts):
"problem with term 1 in model.matrix: no columns are assigned"
Warning message in model.matrix.default(mt, mf, contrasts):
"the response appeared on the right-hand side and was dropped"
Warning message in model.matrix.default(mt, mf, contrasts):
"problem with term 1 in model.matrix: no columns are assigned"
Warning message in model.matrix.default(mt, mf, contrasts):
"the response appeared on the right-hand side and was dropped"
Warning message in model.matrix.default(mt, mf, contrasts):
"problem with term 1 in model.matrix: no columns are assigned"
Warning message in model.matrix.default(mt, mf, contrasts):
"the response appeared on the right-hand side and was dropped"
Warning message in model.matrix.default(mt, mf, contrasts):
"problem with term 1 in model.matrix: no columns are assigned"
Warning message in model.matrix.default(mt, mf, contrasts):
"the response appeared on the right-hand side and was dropped"
Warning message in model.matrix.default(mt, mf, contrasts):
"problem with term 1 in model.matrix: no columns are assigned"
Warning message in model.matrix.default(mt, mf, contrasts):
"the response appeared on the right-hand side and was dropped"
Warning message in model.matrix.default(mt, mf, contrasts):
"problem with term 1 in model.matrix: no columns are assigned"
Warning message in model.matrix.default(mt, mf, contrasts):
"the response appeared on the right-hand side and was dropped"
Warning message in model.matrix.default(mt, mf, contrasts):
"problem with term 1 in model.matrix: no columns are assigned"
Warning message in model.matrix.default(mt, mf, contrasts):
```

```
"the response appeared on the right-hand side and was dropped"
Warning message in model.matrix.default(mt, mf, contrasts):
"problem with term 1 in model.matrix: no columns are assigned"
```

A data.frame: 10 × 4

Predictors	AIC	BIC	Adj_
<chr>	<dbl>	<dbl>	<dbl
SO, IPouts	413059.8	413085.9	0.7209
SO, IPouts, CG	399817.3	399852.1	0.7933
SO, IPouts, CG, SHO	398377.6	398421.0	0.7999
SO, G, IPouts, CG, SHO	397831.0	397883.2	0.8024
SO, G, IPouts, HR, CG, SHO	397068.6	397129.5	0.8057
SO, G, GS, IPouts, HR, CG, SHO	396742.4	396811.9	0.8072
SO, G, GS, IPouts, HR, BB, CG, SHO	396541.5	396619.7	0.8081
SO, G, GS, IPouts, ERA, HR, BB, CG, SHO	395814.0	395901.0	0.8078
SO, G, GS, IPouts, ERA, BAOpp, HR, BB, CG, SHO	379345.8	379441.0	0.8149
SO, G, GS, IPouts, BFP, ERA, BAOpp, HR, BB, CG, SHO	379271.9	379375.8	0.8149

379271.884605165

A data.frame: 1 × 4

	Predictors	AIC	BIC	Ad
	<chr>	<dbl>	<dbl>	<
10	SO, G, GS, IPouts, BFP, ERA, BAOpp, HR, BB, CG, SHO	379271.9	379375.8	0.

379375.778209785

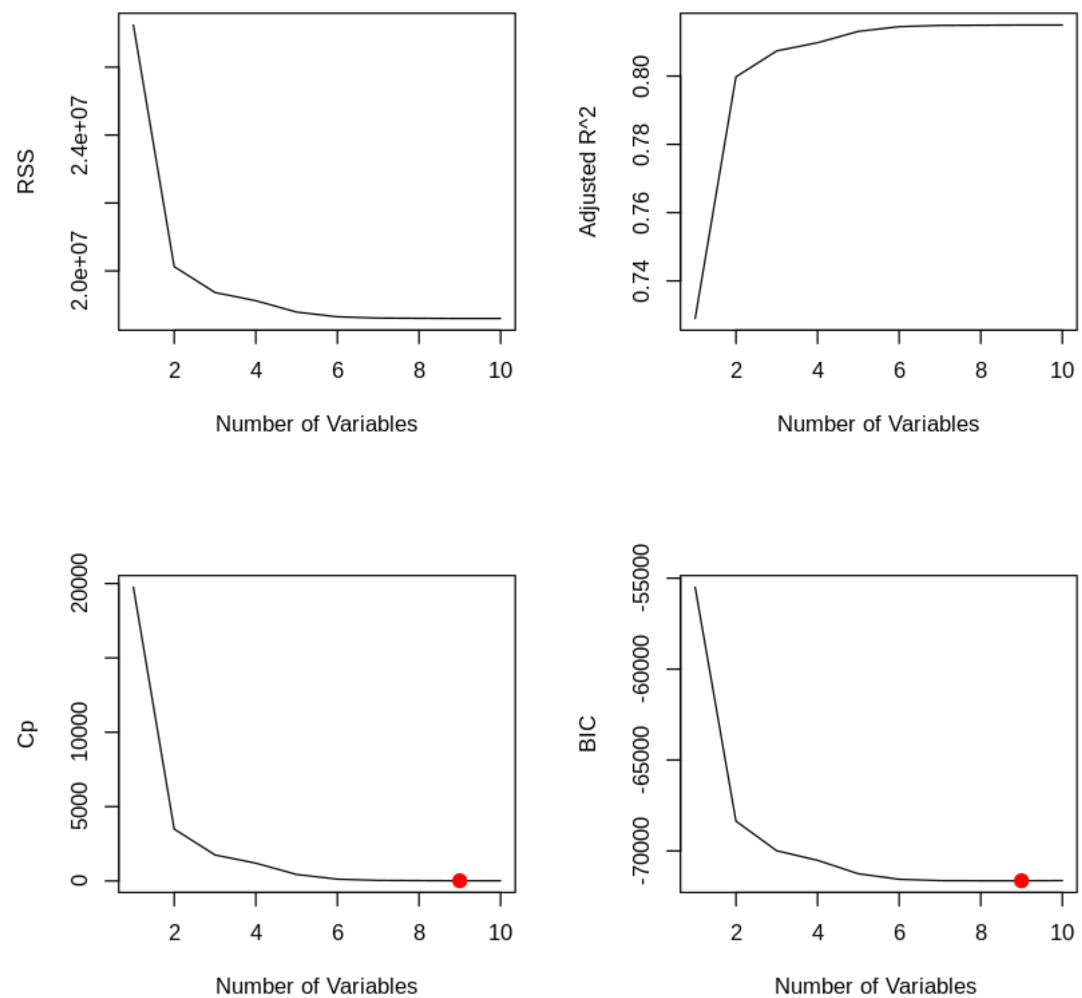
A data.frame: 1 × 4

	Predictors	AIC	BIC	Ad
	<chr>	<dbl>	<dbl>	<
10	SO, G, GS, IPouts, BFP, ERA, BAOpp, HR, BB, CG, SHO	379271.9	379375.8	0.

0.81494307632271

A data.frame: 1 × 4

	Predictors	AIC	BIC	Adj_R2
	<chr>	<dbl>	<dbl>	<dbl>
9	SO, G, GS, IPouts, ERA, BAOpp, HR, BB, CG, SHO	379345.8	379441	0.8149431



From the above results, the best model according to `AIC` and `BIC` is model 10, which contains all 10 predictors:

1. Games Played
2. Games Started
3. Outs Pitched
4. Batters Faced by Pitcher
5. Earned Run Average
6. Batting Average Against
7. Homeruns Allowed
8. Walks Allowed
9. Complete Games
10. Shutouts

This model had both the lowest `AIC` value and lowest `BIC` value.

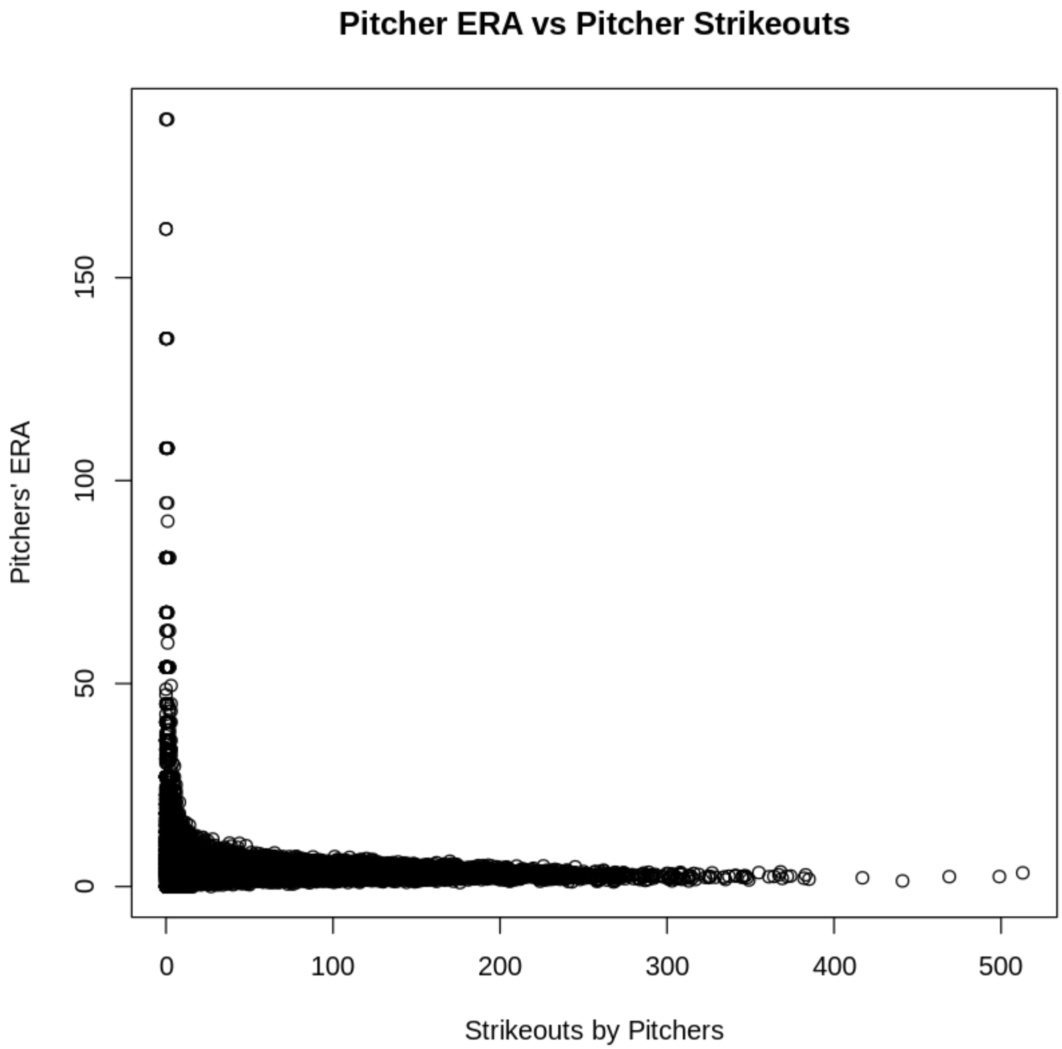
However, the best model according to Mallows Cp and Adjusted R^2 is model 9, which contains the following predictors:

1. Games Played
2. Games Started
3. Outs Pitched
4. Earned Run Average
5. Batting Average Against
6. Homeruns Allowed
7. Walks Allowed
8. Complete Games
9. Shutouts

This model had the highest Adjusted R^2 of the models that were produced using the regsubsets library. (Note: Mallows Cp is a balance of model fit and complexity, and we want small & close to the number of predictors)

7.0.5 Strikeouts vs ERA plot: Visualize relationship between important pitching metrics.

[ ]:
```
# Pitching: Strikeouts vs ERA plot
plot(pitching_data$SO, pitching_data$ERA,
     xlab="Strikeouts by Pitchers",
     ylab="Pitchers' ERA",
     main = "Pitcher ERA vs Pitcher Strikeouts")
```



A high number of `Strikeouts (by pitchers)` is expected to indicate a low `Earned Run Average (ERA)`, which is what we see in the above graph! Pitchers that do not strikeout batters often either probably gave up a hit, walk, or homerun. Giving these up results in a higher `Earned Run Average (ERA)`, which is also shown in the graph. Clearly, a linear model would not fit here. Again, a non-linear model such as exponential decay would likely fit the data better here.

7.0.6 Predicting Pitchers Earned Run Average (ERA): Which predictors are most relevant to pitching ERA?

First, we try fitting a linear model with some of the pitching metrics that may be considered important predictors for ERA.

[ ]:
```r
# Which predictors are most relevant to pitching ERA?
era_lm <- lm(ERA ~ SO + BB + HR + IPouts, data = pitching_data)
era_lm2 <- lm(ERA ~ SO + BB + HR + IPouts + ER + H + GS, data = pitching_data)

summary(era_lm)
summary(era_lm2)

# ANOVA
anova(era_lm, era_lm2)

# Calculate model metrics - AIC, BIC, Adjusted R^2
aic_value <- AIC(era_lm)
bic_value <- BIC(era_lm)
adj_r_squared <- summary(era_lm)$adj.r.squared

# Display the results
cat("AIC:", aic_value, "\n")
cat("BIC:", bic_value, "\n")
cat("Adjusted R-squared:", adj_r_squared, "\n")
```

```
Call:
lm(formula = ERA ~ SO + BB + HR + IPouts, data = pitching_data)

Residuals:
    Min      1Q  Median      3Q     Max
 -6.282  -1.715  -0.537   0.632 182.793

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  6.2132574  0.0369030 168.367   <2e-16 ***
SO          -0.0173836  0.0010387 -16.736   <2e-16 ***
BB           0.0048525  0.0018118   2.678   0.0074 **
HR           0.0674468  0.0051126  13.192   <2e-16 ***
IPouts      -0.0035688  0.0002304 -15.488   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 5.173 on 44044 degrees of freedom
  (90 observations deleted due to missingness)
Multiple R-squared:  0.05986,	Adjusted R-squared:  0.05978
F-statistic: 701.1 on 4 and 44044 DF,  p-value: < 2.2e-16


Call:
lm(formula = ERA ~ SO + BB + HR + IPouts + ER + H + GS, data = pitching_data)
```

```
Residuals:
    Min      1Q  Median      3Q     Max 
 -6.505  -1.493  -0.571   0.457 182.133 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept)  6.4027896  0.0398958 160.488  < 2e-16 ***
SO          -0.0042448  0.0012556  -3.381 0.000724 ***
BB          -0.0367259  0.0023279 -15.777  < 2e-16 ***
HR          -0.1221411  0.0068279 -17.889  < 2e-16 ***
IPouts      -0.0094025  0.0008443 -11.137  < 2e-16 ***
ER           0.1180548  0.0044339  26.626  < 2e-16 ***
H           -0.0277447  0.0029036  -9.555  < 2e-16 ***
GS           0.1306387  0.0061749  21.156  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 5.068 on 44041 degrees of freedom
  (90 observations deleted due to missingness)
Multiple R-squared:  0.09743,	Adjusted R-squared:  0.09728 
F-statistic: 679.1 on 7 and 44041 DF,  p-value: < 2.2e-16
```

A anova: 2 × 6

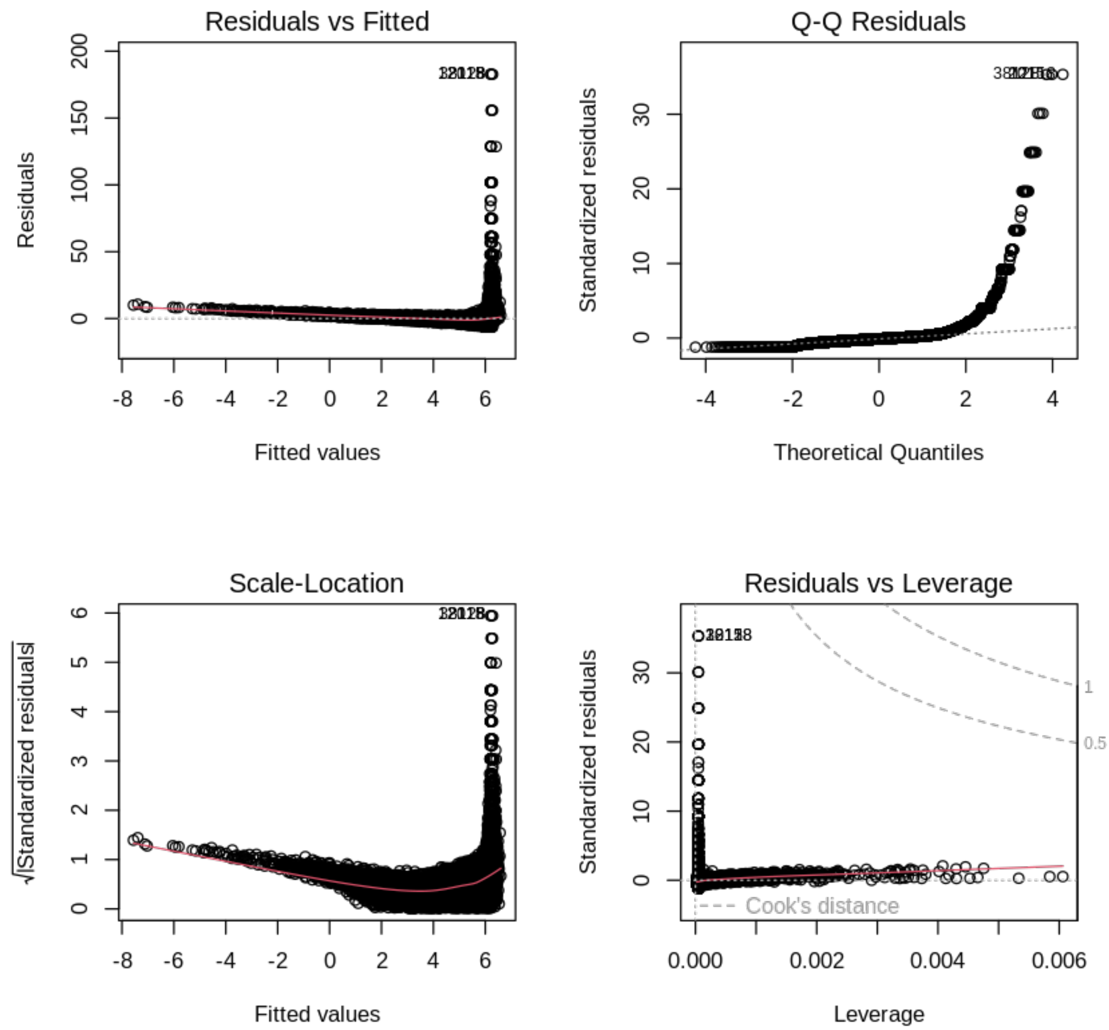
	Res.Df <dbl>	RSS <dbl>	Df <dbl>	Sum of Sq <dbl>	F <dbl>	Pr(>F) <dbl>
1	44044	1178409	NA	NA	NA	NA
2	44041	1131325	3	47084.73	610.9824	0

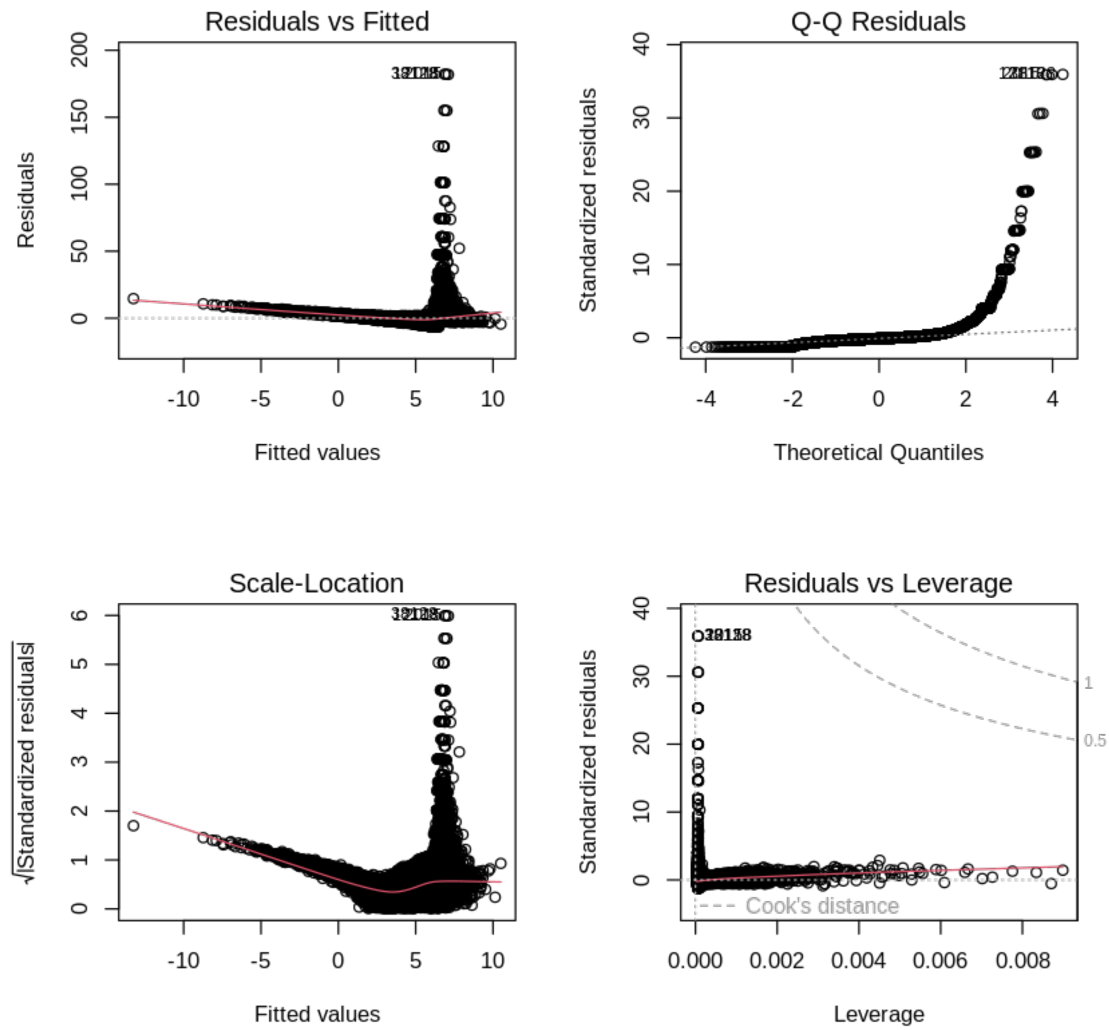
```
AIC: 269789.9
BIC: 269842
Adjusted R-squared: 0.05977688
```

Notice, for both models, we see that all predictors are significant to the response variable, ERA. However, the Adjusted R^2 value for both models is very low, meaning they do not understand the variance within the data. If we were to choose one of the two above models, according to the ANOVA test, we would reject the reduced model since the F-test has a large F-value of 610.9824, which means the additional predictors are significant.

We move on to understand the model assumptions. It would make sense for these to not be met, or violated, because we already see that the variance cannot be captured with a linear model from the above examples.

[ ]:
```r
# Plot Residuals vs Fitted, QQ Residuals, Scale-Location, and Residuals vs Leverage
par(mfrow = c(2,2))
plot(era_lm)
plot(era_lm2)
```


Residuals vs Fitted
Residuals
Fitted values
Q-Q Residuals
Standardized residuals
Theoretical Quantiles
Scale-Location
√|Standardized residuals|
Fitted values
Residuals vs Leverage
Standardized residuals
Cook's distance
Leverage



Since the Adjusted R^2 was not great for these models, we wanted to see if there was a better linear model that did capture the variance better. So we decided to try and predict pitchers Earned Run Average (ERA) by trying different combinations of predictors using regsubsets.

[ ]:
```
## Predicting Pitchers ERA
# Define relevant predictors manually
predictors <- c("SO", "BB", "HR", "IPouts", "ER", "H", "GS")
# Strikeouts, Walks, Homeruns, Outs Pitched, Earned Runs,
# Hits Allowed, Games Started

# Filter dataset to include only relevant predictors and response variable
pitching_data_filtered <- pitching_data[, c("ERA", predictors)]
```

```r
# Perform all subset selection
all_models <- regsubsets(ERA ~ ., data = pitching_data_filtered, nvmax =
  length(predictors))

# Extract model summaries
summary_all <- summary(all_models)
summary_all
names(summary_all)

# Initialize results dataframe
model_results <- data.frame(
  Predictors = character(),
  AIC = numeric(),
  BIC = numeric(),
  Adj_R2 = numeric(),
  stringsAsFactors = FALSE
)

# Loop through models to compute metrics
for (i in 1:nrow(summary_all$which)) {
  # Get selected predictors
  selected_predictors <- names(pitching_data_filtered)[summary_all$which[i, ]]

  # Create formula dynamically
  formula <- as.formula(paste("ERA ~", paste(selected_predictors, collapse = "
  + ")))

  # Fit the model
  model <- lm(formula, data = pitching_data_filtered)

  # Compute metrics
  aic_value <- AIC(model)
  bic_value <- BIC(model)
  adj_r_squared <- summary(model)$adj.r.squared

  # Save results
  model_results <- rbind(
    model_results,
    data.frame(
      Predictors = paste(selected_predictors, collapse = ", "),
      AIC = aic_value,
      BIC = bic_value,
      Adj_R2 = adj_r_squared
    )
  )
}
```

```r
# Display model results
model_results

# Display the best model for each metric
min_aic <- min(model_results$AIC)
min_aic_model <- model_results[model_results$AIC == min_aic, ]
min_aic
min_aic_model

min_bic <- min(model_results$BIC)
min_bic_model <- model_results[model_results$BIC == min_bic, ]
min_bic
min_bic_model

max_adj_r2 <- max(model_results$Adj_R2)
max_adj_r2_model <- model_results[model_results$Adj_R2 == max_adj_r2, ]
max_adj_r2
max_adj_r2_model
```

```
Subset selection object
Call: regsubsets.formula(ERA ~ ., data = pitching_data_filtered, nvmax =
 ↪length(predictors))
7 Variables  (and intercept)
       Forced in Forced out
SO         FALSE      FALSE
BB         FALSE      FALSE
HR         FALSE      FALSE
IPouts     FALSE      FALSE
ER         FALSE      FALSE
H          FALSE      FALSE
GS         FALSE      FALSE
1 subsets of each size up to 7
Selection Algorithm: exhaustive
         SO  BB  HR  IPouts ER  H   GS
1  ( 1 ) " " " " " " "*"    " " " " " "
2  ( 1 ) " " " " " " "*"    "*" " " " "
3  ( 1 ) " " " " " " "*"    "*" " " "*"
4  ( 1 ) " " " " "*" "*"    "*" " " "*"
5  ( 1 ) " " "*" "*" "*"    "*" " " "*"
6  ( 1 ) " " "*" "*" "*"    "*" "*" "*"
7  ( 1 ) "*" "*" "*" "*"    "*" "*" "*"
```

1. 'which' 2. 'rsq' 3. 'rss' 4. 'adjr2' 5. 'cp' 6. 'bic' 7. 'outmat' 8. 'obj'

```
Warning message in model.matrix.default(mt, mf, contrasts):
"the response appeared on the right-hand side and was dropped"
Warning message in model.matrix.default(mt, mf, contrasts):
"problem with term 1 in model.matrix: no columns are assigned"
```

```
Warning message in model.matrix.default(mt, mf, contrasts):
"the response appeared on the right-hand side and was dropped"
Warning message in model.matrix.default(mt, mf, contrasts):
"problem with term 1 in model.matrix: no columns are assigned"
Warning message in model.matrix.default(mt, mf, contrasts):
"the response appeared on the right-hand side and was dropped"
Warning message in model.matrix.default(mt, mf, contrasts):
"problem with term 1 in model.matrix: no columns are assigned"
Warning message in model.matrix.default(mt, mf, contrasts):
"the response appeared on the right-hand side and was dropped"
Warning message in model.matrix.default(mt, mf, contrasts):
"problem with term 1 in model.matrix: no columns are assigned"
Warning message in model.matrix.default(mt, mf, contrasts):
"the response appeared on the right-hand side and was dropped"
Warning message in model.matrix.default(mt, mf, contrasts):
"problem with term 1 in model.matrix: no columns are assigned"
Warning message in model.matrix.default(mt, mf, contrasts):
"the response appeared on the right-hand side and was dropped"
Warning message in model.matrix.default(mt, mf, contrasts):
"problem with term 1 in model.matrix: no columns are assigned"
Warning message in model.matrix.default(mt, mf, contrasts):
"the response appeared on the right-hand side and was dropped"
Warning message in model.matrix.default(mt, mf, contrasts):
"problem with term 1 in model.matrix: no columns are assigned"
```

A data.frame: 7 × 4

Predictors <chr>	AIC <dbl>	BIC <dbl>	Adj_R2 <dbl>
ERA, IPouts	270121.3	270147.3	0.05261283
ERA, IPouts, ER	268919.0	268953.8	0.07814053
ERA, IPouts, ER, GS	268579.4	268622.9	0.08524143
ERA, HR, IPouts, ER, GS	268257.6	268309.8	0.09192041
ERA, BB, HR, IPouts, ER, GS	268087.3	268148.1	0.09544621
ERA, BB, HR, IPouts, ER, H, GS	268009.2	268078.7	0.09706934
ERA, SO, BB, HR, IPouts, ER, H, GS	267999.7	268078.0	0.09728311

267999.732773535

A data.frame: 1 × 4

	Predictors <chr>	AIC <dbl>	BIC <dbl>	Adj_R2 <dbl>
7	ERA, SO, BB, HR, IPouts, ER, H, GS	267999.7	268078	0.09728311

268077.970294902

A data.frame: 1 × 4

	Predictors <chr>	AIC <dbl>	BIC <dbl>	Adj_R2 <dbl>
7	ERA, SO, BB, HR, IPouts, ER, H, GS	267999.7	268078	0.09728311

0.0972831111824469

A data.frame: 1 × 4

	Predictors <chr>	AIC <dbl>	BIC <dbl>	Adj_R2 <dbl>
7	ERA, SO, BB, HR, IPouts, ER, H, GS	267999.7	268078	0.09728311

Note here that model 7 is the only model that keeps Strikeouts (SO) as a predictor variable. This is interesting because we saw before that there is an exponential decay-like relationship between Strikeouts by pitchers and their ERA. This also aligns with our ANOVA test from earlier, that the full model with all seven predictors are significant.

According to all metrics, `AIC`, `BIC`, and Adjusted R^2, model 7 with all seven predictors is the best. As seen before from the violations in assumptions, a non-linear model would be a better fit (via GAMs) since the Adjusted R^2 for model 7 is extremely low to be selected as the best model.

8 3. Salaries, Fielding, and Batting

[ ]:
```
# Build dataframe that merges salary and fielding
salary_fielding_data <- merge(fielding_data, salary_data,
                              by = c("playerID", "yearID", "teamID", "lgID")) #␣
↪merge based on these columns
# View dataframe
head(salary_fielding_data)
```

A data.frame: 6 × 19

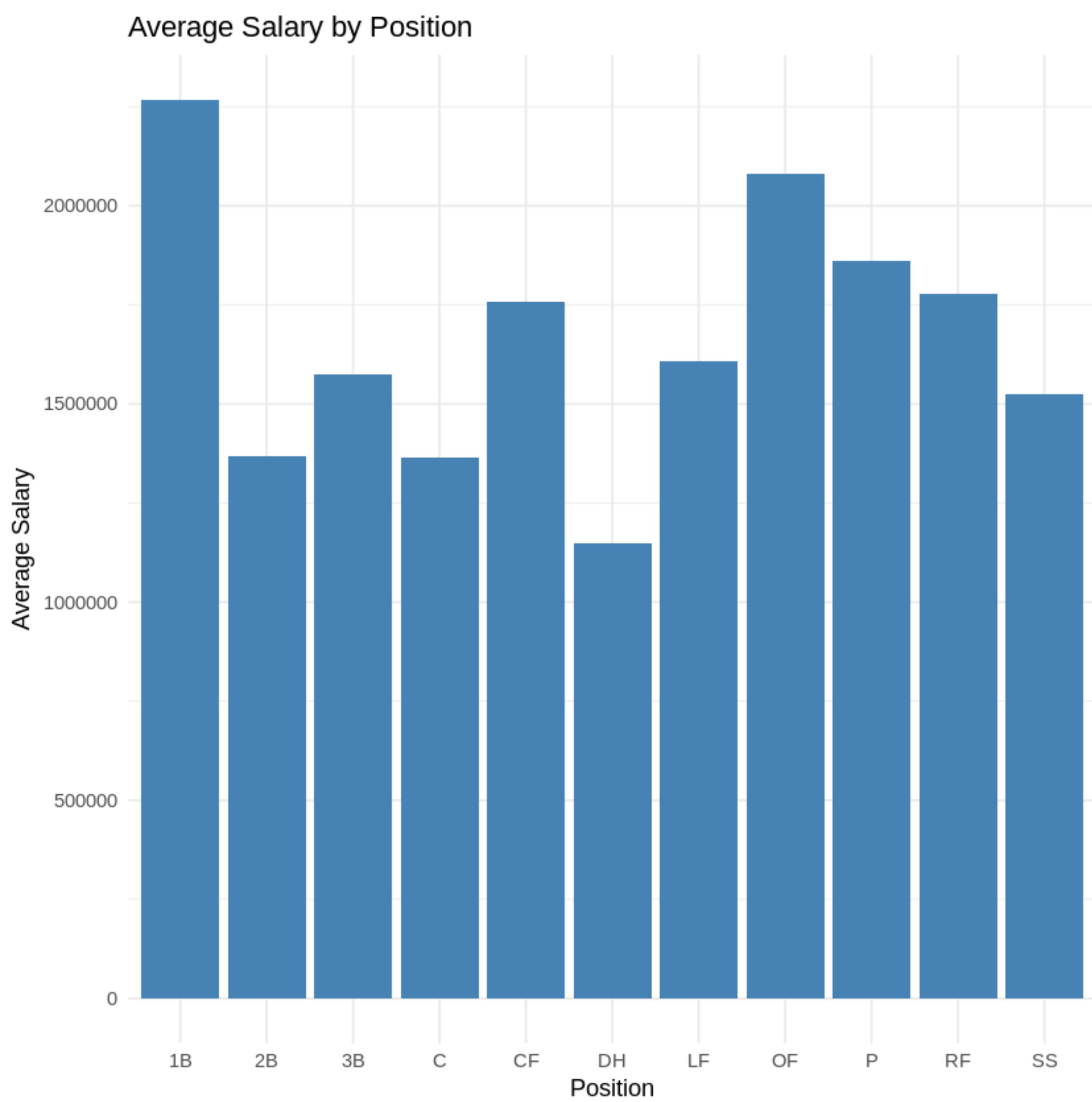
	playerID <chr>	yearID <int>	teamID <chr>	lgID <chr>	stint <int>	POS <chr>	G <int>	GS <int>	InnOuts <int>
1	aardsda01	2004	SFN	NL	1	P	11	NA	33
2	aardsda01	2007	CHA	AL	1	P	25	NA	96
3	aardsda01	2008	BOS	AL	1	P	47	NA	147
4	aardsda01	2009	SEA	AL	1	P	73	NA	213
5	aardsda01	2010	SEA	AL	1	P	53	NA	150
6	aardsda01	2012	NYA	AL	1	P	1	NA	3

8.0.1 What position makes the most? What position makes the least?

[ ]:
```
# Compute the average salary for each position
avg_salary_by_pos <- salary_fielding_data %>%
  group_by(POS) %>% # group data by position
  summarise(avg_salary = mean(salary, na.rm = TRUE)) %>% # compute avg salary␣
↪per position, exclude NA values
  arrange(desc(avg_salary)) # sort by descending order

# Display the position with the highest average salary
highest_paid_position <- avg_salary_by_pos %>%
  slice_max(avg_salary) # get position w highest salary

# Print results
print(avg_salary_by_pos)

# Create a bar plot
ggplot(avg_salary_by_pos, aes(x = POS, y = avg_salary)) +
  geom_bar(stat = "identity", fill = "steelblue") +
  labs(title = "Average Salary by Position",
```

```r
      x = "Position",
      y = "Average Salary") +
  theme_minimal()
```

```
# A tibble: 11 × 2
   POS   avg_salary
   <chr>      <dbl>
 1 1B       2266841.
 2 OF       2079442.
 3 P        1862235.
 4 RF       1778585.
 5 CF       1758170.
 6 LF       1606448.
 7 3B       1574785.
 8 SS       1525056.
 9 2B       1367368.
10 C        1365087.
11 DH       1149819.
```



The highest paid position is First Base (1B) with an average saslary of $2,266,841. The lowest paid position is Designated Hitter (DH) with an average salary of $1,149,819.

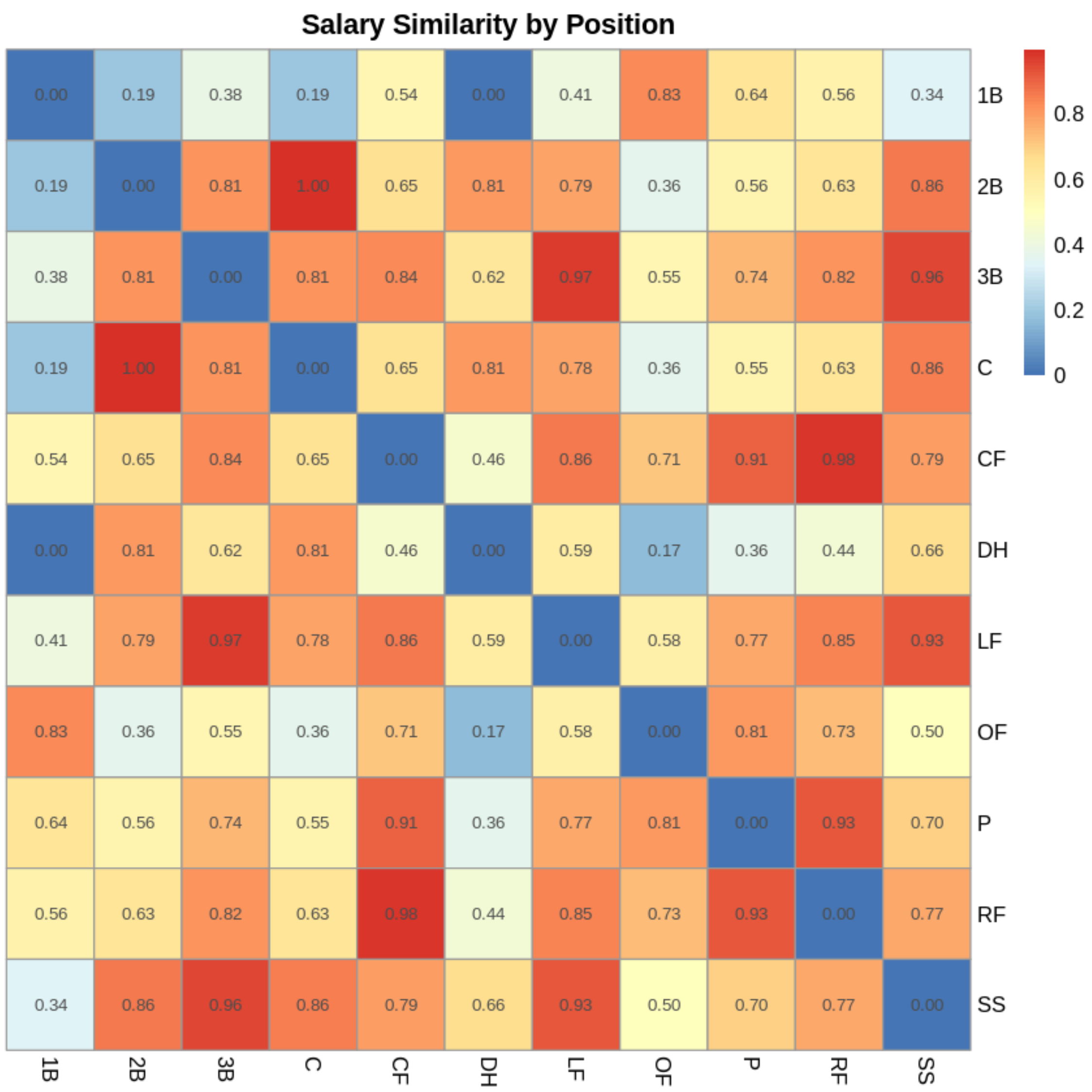
8.0.2 Are there any correlations between salary and fielding position?

To see if there are any correlations (or in our case similarities), we visualized this using a heat map. A simple interpretation of the map is as follows:

- Each cell in the heatmap will show the similarity between two positions based on their salaries.
- A similarity of 1 means the positions have identical average salaries (lower differences), while lower values indicate larger differences.

[ ]:
```
# Compute average/mean salary by position
avg_salary_by_pos <- salary_fielding_data %>%
```

```r
  group_by(POS) %>%
  summarise(avg_salary = mean(salary, na.rm = TRUE))

# Create a correlation-like matrix (reshape2 library)
# Calculate pairwise distances between positions based on salary differences
salary_matrix <- dist(avg_salary_by_pos$avg_salary) # Convert avg salary by
 ↪position into distance matrix
salary_corr <- as.matrix(1 - salary_matrix / max(salary_matrix)) # Convert dist
 ↪matrix into similarity matrix

# Add row/column names to the matrix
rownames(salary_corr) <- colnames(salary_corr) <- avg_salary_by_pos$POS

# Visualize the heatmap via pheatmap library
pheatmap(salary_corr,
         display_numbers = TRUE,
         cluster_rows = FALSE,
         cluster_cols = FALSE,
         main = "Salary Similarity by Position")
```



We are most surprised that the Second Base position and Catcher have a similarity of 1. We can double check this with the code below.

```
[ ]: # Check the average salaries for positions 2B and C
avg_salary_2B <- avg_salary_by_pos$avg_salary[avg_salary_by_pos$POS == "2B"]
avg_salary_C <- avg_salary_by_pos$avg_salary[avg_salary_by_pos$POS == "C"]

avg_salary_3B <- avg_salary_by_pos$avg_salary[avg_salary_by_pos$POS == "3B"]
avg_salary_LF <- avg_salary_by_pos$avg_salary[avg_salary_by_pos$POS == "LF"]

avg_salary_SS <- avg_salary_by_pos$avg_salary[avg_salary_by_pos$POS == "SS"]

avg_salary_CF <- avg_salary_by_pos$avg_salary[avg_salary_by_pos$POS == "CF"]
```

```
avg_salary_RF <- avg_salary_by_pos$avg_salary[avg_salary_by_pos$POS == "RF"]

avg_salary_1B <- avg_salary_by_pos$avg_salary[avg_salary_by_pos$POS == "1B"]
avg_salary_DH <- avg_salary_by_pos$avg_salary[avg_salary_by_pos$POS == "DH"]

# Check average salaries for positions that are similar (low diff, high␣
↪similarity in salary)
cat("Average salary for 2B:", avg_salary_2B, "\n")
cat("Average salary for C:", avg_salary_C, "\n")
cat("Difference between 2B and C:", abs(avg_salary_2B-avg_salary_C), "\n")

cat("Average salary for 3B:", avg_salary_3B, "\n")
cat("Average salary for LF:", avg_salary_LF, "\n")
cat("Difference between 3B and LF:", abs(avg_salary_3B-avg_salary_LF), "\n")

cat("Average salary for 3B:", avg_salary_3B, "\n")
cat("Average salary for SS:", avg_salary_SS, "\n")
cat("Difference between 3B and SS:", abs(avg_salary_3B-avg_salary_SS), "\n")

cat("Average salary for CF:", avg_salary_CF, "\n")
cat("Average salary for RF:", avg_salary_RF, "\n")
cat("Difference between CF and RF:", abs(avg_salary_CF-avg_salary_RF), "\n")

# Example with big difference, low similarity in salary
cat("Average salary for 1B:", avg_salary_1B, "\n")
cat("Average salary for C:", avg_salary_C, "\n")
cat("Difference between 1B and C:", abs(avg_salary_1B-avg_salary_C), "\n")

# Example with similarity of zero
cat("Average salary for 1B:", avg_salary_1B, "\n")
cat("Average salary for DH:", avg_salary_DH, "\n")
cat("Difference between 1B and DH:", abs(avg_salary_1B-avg_salary_DH), "\n")
```

```
Average salary for 2B: 1367368
Average salary for C: 1365087
Difference between 2B and C: 2280.226
Average salary for 3B: 1574785
Average salary for LF: 1606448
Difference between 3B and LF: 31663.15
Average salary for 3B: 1574785
Average salary for SS: 1525056
Difference between 3B and SS: 49728.98
Average salary for CF: 1758170
Average salary for RF: 1778585
Difference between CF and RF: 20415.33
Average salary for 1B: 2266841
Average salary for C: 1365087
```

```
Difference between 1B and C: 901753.2
Average salary for 1B: 2266841
Average salary for DH: 1149819
Difference between 1B and DH: 1117022
```

Relative to other mean differences in salary, a mean difference of 49,728 is low indicating high similarity (as shown by 3B and SS). We can see this from comparing it to a mean difference of positions with low similarity. For example, a mean difference of 901,753 is much larger (shown by 1B and C).

Another important thing to notice is that from the previous section we conclude the highest and lowest paid positions are 1B and DH, relatively. This makes sense why their similarity score is zero. They have a very large difference in mean salary of $1,117,022.

8.0.3 Predicting salary for pitchers via ANOVA. What are the most relevant predictors?

We first need to extract data from the merged dataframe. We specifically want the salary data for pitchers only. Then we need to merge the pitching data with the salary_fielding_data (that contains position and salary). After that, we need to clean the data which means removing rows that contain missing values (`NA`) and removing columns that contain all `NA` values (shown below).

[ ]:
```
# Reduce fielding_data to only identifiers and position columns
reduced_fielding_data <- fielding_data %>%
  select(playerID, yearID, teamID, lgID, POS)

# Merge salary_data with reduced_fielding_data
salary_fielding_data <- merge(reduced_fielding_data, salary_data,
                              by = c("playerID", "yearID", "teamID", "lgID"))

# Extract pitcher data
pitcher_salary_data <- salary_fielding_data %>%
  filter(POS == "P")

# Merge pitcher salary data with pitching_data (dataset with salaries of␣
 ↪pitchers only)
pitcher_data <- merge(pitcher_salary_data, pitching_data,
                              by = c("playerID", "yearID", "teamID", "lgID"))

# View dataset before cleaning
head(pitcher_data)
colnames(pitcher_data)

# Data Cleaning
cat("Total Number of Rows Before Cleaning:")
nrow(pitcher_data) # 11537
sapply(pitcher_data, function(x) sum(is.na(x)))

# Drop GIDP (Ground Into Double Play) column with 11152 missing values
```

```
pitcher_data <- pitcher_data %>% select(-GIDP)

# Drop rows with NA, the columns with NA are ERA, BAOpp, SH, and SF
pitcher_data <- pitcher_data %>% drop_na(ERA, BAOpp, SH, SF)

# View dataset after cleaning
head(pitcher_data)
cat("Total Number of Rows After Cleaning:")
nrow(pitcher_data)
colnames(pitcher_data)
sapply(pitcher_data, function(x) sum(is.na(x)))
```

A data.frame: 6 × 35

	playerID <chr>	yearID <int>	teamID <chr>	lgID <chr>	POS <chr>	salary <int>	stint <int>	W <int>	L <int>
1	aardsda01	2004	SFN	NL	P	300000	1	1	0
2	aardsda01	2007	CHA	AL	P	387500	1	2	1
3	aardsda01	2008	BOS	AL	P	403250	1	4	2
4	aardsda01	2009	SEA	AL	P	419000	1	3	6
5	aardsda01	2010	SEA	AL	P	2750000	1	0	6
6	aardsda01	2012	NYA	AL	P	500000	1	0	0

1. 'playerID' 2. 'yearID' 3. 'teamID' 4. 'lgID' 5. 'POS' 6. 'salary' 7. 'stint' 8. 'W' 9. 'L' 10. 'G' 11. 'GS' 12. 'CG' 13. 'SHO' 14. 'SV' 15. 'IPouts' 16. 'H' 17. 'ER' 18. 'HR' 19. 'BB' 20. 'SO' 21. 'BAOpp' 22. 'ERA' 23. 'IBB' 24. 'WP' 25. 'HBP' 26. 'BK' 27. 'BFP' 28. 'GF' 29. 'R' 30. 'SH' 31. 'SF' 32. 'GIDP' 33. 'walk_rate' 34. 'strikeout_percentage' 35. 'K_BB_ratio'

```
Total Number of Rows Before Cleaning:
```

11537

playerID 0 **yearID** 0 **teamID** 0 **lgID** 0 **POS** 0 **salary** 0 **stint** 0 **W** 0 **L** 0 **G** 0 **GS** 0 **CG** 0 **SHO** 0 **SV** 0 **IPouts** 0 **H** 0 **ER** 0 **HR** 0 **BB** 0 **SO** 0 **BAOpp** 465 **ERA** 6 **IBB** 0 **WP** 0 **HBP** 0 **BK** 0 **BFP** 0 **GF** 0 **R** 0 **SH** 5320 **SF** 5320 **GIDP** 11152 **walk_rate** 0 **strikeout_percentage** 0 **K_BB_ratio** 140

A data.frame: 6 × 34

	playerID <chr>	yearID <int>	teamID <chr>	lgID <chr>	POS <chr>	salary <int>	stint <int>	W <int>	L <int>
1	aardsda01	2004	SFN	NL	P	300000	1	1	0
2	aardsda01	2007	CHA	AL	P	387500	1	2	1
3	aardsda01	2008	BOS	AL	P	403250	1	4	2
4	aardsda01	2009	SEA	AL	P	419000	1	3	6
5	aardsda01	2010	SEA	AL	P	2750000	1	0	6
6	aardsda01	2012	NYA	AL	P	500000	1	0	0

```
Total Number of Rows After Cleaning:
```

6214

1. 'playerID' 2. 'yearID' 3. 'teamID' 4. 'lgID' 5. 'POS' 6. 'salary' 7. 'stint' 8. 'W' 9. 'L' 10. 'G' 11. 'GS' 12. 'CG' 13. 'SHO' 14. 'SV' 15. 'IPouts' 16. 'H' 17. 'ER' 18. 'HR' 19. 'BB' 20. 'SO' 21. 'BAOpp' 22. 'ERA' 23. 'IBB' 24. 'WP' 25. 'HBP' 26. 'BK' 27. 'BFP' 28. 'GF' 29. 'R' 30. 'SH' 31. 'SF' 32. 'walk_rate' 33. 'strikeout_percentage' 34. 'K_BB_ratio'

playerID 0 **yearID** 0 **teamID** 0 **lgID** 0 **POS** 0 **salary** 0 **stint** 0 **W** 0 **L** 0 **G** 0 **GS** 0 **CG** 0 **SHO** 0 **SV** 0 **IPouts** 0 **H** 0 **ER** 0 **HR** 0 **BB** 0 **SO** 0 **BAOpp** 0 **ERA** 0 **IBB** 0 **WP** 0 **HBP** 0 **BK** 0 **BFP** 0 **GF** 0 **R** 0 **SH** 0 **SF** 0 **walk_rate** 0 **strikeout_percentage** 0 **K_BB_ratio** 89

[ ]:
```
# Full model
lm_pitchers_full <- lm(salary ~ W + L + G + GS + CG + SHO + IPouts + H + ER +
  ↪HR + BB + SO + BAOpp + ERA + IBB + HBP + R + SH + SF, data = pitcher_data)
summary(lm_pitchers_full)
# Reduce model - contains predictors that are significant from the full model
lm_pitchers_red <- lm(salary ~ L + GS + IPouts + ER + HR + BB + SO + IBB + R,
  ↪data = pitcher_data)
summary(lm_pitchers_red)
# Second reduced model (- IPouts)
lm_pitchers_red2 <- lm(salary ~ L + GS + ER + HR + BB + SO + IBB + R, data =
  ↪pitcher_data)
summary(lm_pitchers_red2)

# ANOVA: anova(reduced, full)
# compare full vs reduced
anova(lm_pitchers_red, lm_pitchers_full)
# compare reduced vs reduced2
anova(lm_pitchers_red2, lm_pitchers_red)
# compare full vs reduced2
anova(lm_pitchers_red2, lm_pitchers_full)
```

```
Call:
lm(formula = salary ~ W + L + G + GS + CG + SHO + IPouts + H +
    ER + HR + BB + SO + BAOpp + ERA + IBB + HBP + R + SH + SF,
    data = pitcher_data)

Residuals:
     Min       1Q   Median       3Q      Max
-8695570 -1718149  -839242   883008 23270599

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  2632815.3   300222.2   8.770  < 2e-16 ***
W              39692.0    24097.1   1.647  0.09957 .
L             115637.7    25378.1   4.557 5.30e-06 ***
G                138.6     5377.0   0.026  0.97944
GS            118182.8    25593.8   4.618 3.96e-06 ***
CG             63119.9    78938.2   0.800  0.42397
SHO            13242.8   142560.3   0.093  0.92599
IPouts         -5124.2     2512.4  -2.040  0.04144 *
H               7244.1     6711.7   1.079  0.28048
ER             45109.9    18244.3   2.473  0.01344 *
HR            -31684.6    14244.9  -2.224  0.02617 *
```

```
BB            -60719.3     5251.8 -11.562  < 2e-16 ***
SO             26939.6     2672.4  10.081  < 2e-16 ***
BAOpp       -1193419.8  1180338.8  -1.011  0.31202
ERA           -30584.0    17104.5  -1.788  0.07381 .
IBB           -60956.1    23983.0  -2.542  0.01106 *
HBP             5631.2    18942.6   0.297  0.76626
R             -49355.0    17548.1  -2.813  0.00493 **
SH             38726.1    20612.3   1.879  0.06032 .
SF             -6257.9    27117.9  -0.231  0.81750
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3413000 on 6194 degrees of freedom
Multiple R-squared:  0.1854,	Adjusted R-squared:  0.1829
F-statistic: 74.19 on 19 and 6194 DF,  p-value: < 2.2e-16



Call:
lm(formula = salary ~ L + GS + IPouts + ER + HR + BB + SO + IBB +
    R, data = pitcher_data)

Residuals:
     Min       1Q   Median       3Q      Max
-8708790 -1682456  -863931   907563 22937782

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 2071906.1    98671.5  20.998  < 2e-16 ***
L            112504.5    23966.1   4.694 2.73e-06 ***
GS           114941.3    11395.1  10.087  < 2e-16 ***
IPouts         -237.3     1155.0  -0.205  0.83721
ER            41668.4    17979.3   2.318  0.02050 *
HR           -35341.4    13714.1  -2.577  0.00999 **
BB           -62200.8     4484.3 -13.871  < 2e-16 ***
SO            25552.0     2437.6  10.482  < 2e-16 ***
IBB          -51234.6    22980.1  -2.230  0.02582 *
R            -46130.6    16888.8  -2.731  0.00632 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3416000 on 6204 degrees of freedom
Multiple R-squared:  0.1829,	Adjusted R-squared:  0.1817
F-statistic: 154.3 on 9 and 6204 DF,  p-value: < 2.2e-16



Call:
lm(formula = salary ~ L + GS + ER + HR + BB + SO + IBB + R, data = pitcher_data)
```

```
Residuals:
     Min       1Q   Median       3Q      Max
-8699918 -1681321  -864444   910620 22936440

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  2067732      96551  21.416  < 2e-16 ***
L             112586      23961   4.699 2.67e-06 ***
GS            113908      10224  11.141  < 2e-16 ***
ER             41795      17967   2.326  0.02004 *
HR            -35433      13706  -2.585  0.00975 **
BB            -62161       4480 -13.876  < 2e-16 ***
SO             25169       1574  15.996  < 2e-16 ***
IBB           -52480      22164  -2.368  0.01792 *
R             -46690      16666  -2.801  0.00510 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3415000 on 6205 degrees of freedom
Multiple R-squared:  0.1829,	Adjusted R-squared:  0.1818
F-statistic: 173.6 on 8 and 6205 DF,  p-value: < 2.2e-16
```

A anova: 2 × 6

	Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
1	6204	7.237506e+16	NA	NA	NA	NA
2	6194	7.215480e+16	10	2.202621e+14	1.890801	0.04166401

A anova: 2 × 6

	Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
1	6205	7.237556e+16	NA	NA	NA	NA
2	6204	7.237506e+16	1	492540372816	0.04222063	0.8372065

A anova: 2 × 6

	Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
1	6205	7.237556e+16	NA	NA	NA	NA
2	6194	7.215480e+16	11	2.207547e+14	1.722754	0.06226846

From the above outputs, the `p-value` for comparing the full model and first reduced model is `0.0416` which is less than 0.05, so we reject the null hypothesis that the model is sufficient. This makes sense from the summaries of the reduced and full model because the Adjusted R^2 value is higher for the full model than the reduced model. These linear models are probably not the best to fit the data because the Adjusted R^2 value is still quite low for the better full model (`0.1829`).

8.0.4 Predicting Salary based on Hitting/Batting (excluding pitchers). What are the most significant predictors?

Next we want to try to see if hitting or batting statistics are influential on predicting salary. We start by merging the batting, salary, and fielding data and cleaning the new dataframe.

[ ]:

```r
# Merge 3 dataframes: batting, fielding, and salary data

# Merge batting_data and salary_data
fully_merged_data <- batting_data %>%
  inner_join(salary_data, by = c("playerID", "yearID", "teamID", "lgID")) %>%
  inner_join(fielding_data, by = c("playerID", "yearID", "teamID", "lgID"))

# View the dataframe
head(fully_merged_data)
nrow(fully_merged_data)
colnames(fully_merged_data)

# Data Cleaning: Keep relative columns, remove others
cleaned_data <- fully_merged_data %>%
  select(
    playerID, yearID, teamID, lgID, stint = stint.x, G = G.x, AB, R, H, X2B, X3B, HR, RBI,
    SB = SB.x, CS = CS.x, BB, SO, IBB, HBP, SH, SF, GIDP, salary, POS
  ) %>%
  filter(POS != "P")  # Exclude pitchers

# Confirm pitchers are excluded
unique(cleaned_data$POS)

# View the dataframe
head(cleaned_data)
nrow(cleaned_data)
colnames(cleaned_data)
# summary(cleaned_data)
```

```
Warning message in inner_join(., fielding_data, by = c("playerID", "yearID",
"teamID", :
"Detected an unexpected many-to-many relationship between `x` and `y`.
 Row 3 of `x` matches multiple rows in `y`.
 Row 123888 of `y` matches multiple rows in `x`.
 If a many-to-many relationship is expected, set `relationship =
  "many-to-many"` to silence this warning."
```

A data.frame: 6 × 46

	playerID <chr>	yearID <int>	stint.x <int>	teamID <chr>	lgID <chr>	G.x <int>	AB <int>	R <int>	H <int>	X2 <
1	ackerji01	1985	1	TOR	AL	61	NA	NA	NA	N
2	agostju01	1985	1	CHA	AL	54	0	0	0	0
3	aguaylu01	1985	1	PHI	NL	91	165	27	46	7
4	aguaylu01	1985	1	PHI	NL	91	165	27	46	7
5	aguaylu01	1985	1	PHI	NL	91	165	27	46	7
6	alexado01	1985	1	TOR	AL	36	NA	NA	NA	N

46883

1. 'playerID' 2. 'yearID' 3. 'stint.x' 4. 'teamID' 5. 'lgID' 6. 'G.x' 7. 'AB' 8. 'R' 9. 'H' 10. 'X2B' 11. 'X3B' 12. 'HR' 13. 'RBI' 14. 'SB.x' 15. 'CS.x' 16. 'BB' 17. 'SO' 18. 'IBB' 19. 'HBP' 20. 'SH' 21. 'SF' 22. 'GIDP' 23. 'BA' 24. 'OBP' 25. 'SB_percent' 26. 'walk_rate' 27. 'strikeout_percentage' 28. 'log_SO' 29. 'log_AB' 30. 'log_HR' 31. 'log_BB' 32. 'salary' 33. 'stint.y' 34. 'POS' 35. 'G.y' 36. 'GS' 37. 'InnOuts' 38. 'PO' 39. 'A' 40. 'E' 41. 'DP' 42. 'PB' 43. 'WP' 44. 'SB.y' 45. 'CS.y' 46. 'ZR'

1. '2B' 2. '3B' 3. 'SS' 4. '1B' 5. 'CF' 6. 'LF' 7. 'OF' 8. 'DH' 9. 'RF' 10. 'C'

A data.frame: 6 × 24

	playerID <chr>	yearID <int>	teamID <chr>	lgID <chr>	stint <int>	G <int>	AB <int>	R <int>	H <int>	X <
1	aguaylu01	1985	PHI	NL	1	91	165	27	46	7
2	aguaylu01	1985	PHI	NL	1	91	165	27	46	7
3	aguaylu01	1985	PHI	NL	1	91	165	27	46	7
4	almonbi01	1985	PIT	NL	1	88	244	33	66	17
5	almonbi01	1985	PIT	NL	1	88	244	33	66	17
6	almonbi01	1985	PIT	NL	1	88	244	33	66	17

35346

1. 'playerID' 2. 'yearID' 3. 'teamID' 4. 'lgID' 5. 'stint' 6. 'G' 7. 'AB' 8. 'R' 9. 'H' 10. 'X2B' 11. 'X3B' 12. 'HR' 13. 'RBI' 14. 'SB' 15. 'CS' 16. 'BB' 17. 'SO' 18. 'IBB' 19. 'HBP' 20. 'SH' 21. 'SF' 22. 'GIDP' 23. 'salary' 24. 'POS'

Now we identify the full model and the reduced model. The predictors for the reduced model are determined from the significant predictors of the full model. We follow up with an Analysis of Variance (ANOVA) test, to determine which model is sufficient.

[ ]:
```r
# Fit a linear model with hitting data
lm_hitting_salary_full <- lm(salary ~ AB + R + H + X2B + X3B + HR + RBI + SB +
  CS + BB + SO + IBB + HBP + SH + SF + GIDP, data = cleaned_data)
summary(lm_hitting_salary_full)

lm_hitting_salary_red <- lm(salary ~ AB + R + X3B + HR + RBI + SB + CS + BB +
  IBB + HBP + SH + GIDP, data = cleaned_data)
summary(lm_hitting_salary_red)

anova(lm_hitting_salary_red, lm_hitting_salary_full)
```

```
Call:
lm(formula = salary ~ AB + R + H + X2B + X3B + HR + RBI + SB +
    CS + BB + SO + IBB + HBP + SH + SF + GIDP, data = cleaned_data)

Residuals:
     Min       1Q   Median       3Q      Max
-7850485 -1181924  -409391   253221 30077427

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
```

```
(Intercept)  331930.2    29730.0  11.165  < 2e-16 ***
AB             2434.7      574.7   4.237 2.27e-05 ***
R             14480.0     2464.9   5.874 4.28e-09 ***
H              1949.3     2157.1   0.904    0.366    
X2B           -2062.7     3590.3  -0.575    0.566    
X3B         -138437.2     9077.2 -15.251  < 2e-16 ***
HR            40988.8     5109.2   8.023 1.07e-15 ***
RBI          -14641.0     2291.1  -6.390 1.68e-10 ***
SB            23742.9     2697.4   8.802  < 2e-16 ***
CS          -150290.2     7426.1 -20.238  < 2e-16 ***
BB            13744.1     1356.0  10.136  < 2e-16 ***
SO            -1283.0      916.0  -1.401    0.161    
IBB           70174.3     5429.6  12.924  < 2e-16 ***
HBP           41184.2     5339.9   7.713 1.27e-14 ***
SH          -131605.6     6394.8 -20.580  < 2e-16 ***
SF             2858.2     9252.1   0.309    0.757    
GIDP          50872.4     4727.8  10.760  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2691000 on 35329 degrees of freedom
Multiple R-squared:  0.2107,	Adjusted R-squared:  0.2103 
F-statistic: 589.4 on 16 and 35329 DF,  p-value: < 2.2e-16



Call:
lm(formula = salary ~ AB + R + X3B + HR + RBI + SB + CS + BB + 
    IBB + HBP + SH + GIDP, data = cleaned_data)

Residuals:
     Min       1Q   Median       3Q      Max 
-7864903 -1183542  -407356   257532 30103332 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept)  318286.9    28742.3  11.074  < 2e-16 ***
AB             2463.7      308.6   7.984 1.46e-15 ***
R             16227.2     2081.2   7.797 6.52e-15 ***
X3B         -139031.3     8988.3 -15.468  < 2e-16 ***
HR            36275.7     4283.3   8.469  < 2e-16 ***
RBI          -13487.9     1958.0  -6.889 5.73e-12 ***
SB            23729.8     2669.9   8.888  < 2e-16 ***
CS          -149818.2     7414.2 -20.207  < 2e-16 ***
BB            12894.1     1271.6  10.140  < 2e-16 ***
IBB           73181.7     5217.6  14.026  < 2e-16 ***
HBP           39855.9     5296.2   7.525 5.38e-14 ***
SH          -131209.2     6298.2 -20.833  < 2e-16 ***
```

```
GIDP          52622.1     4642.9  11.334  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2691000 on 35333 degrees of freedom
Multiple R-squared:  0.2106,	Adjusted R-squared:  0.2103
F-statistic: 785.4 on 12 and 35333 DF,  p-value: < 2.2e-16
```

A anova: 2 × 6

	Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
1	35333	2.557933e+17	NA	NA	NA	NA
2	35329	2.557605e+17	4	3.278722e+13	1.132251	0.3391415

From the above output, we see that with an `F-statistic` of `1.132251` and a high `p-value` of `0.33914`, which means we fail to reject the null hypothesis that the reduced model is sufficient. In other words, there is insufficient evidence to conclude that the added predictors in the full model provide a significant difference in prediction compared to the reduced model.

After concluding that the reduced model is better, we are interested in seeing if there is a different combination of predictors that provide a higher Adjusted R^2 than the reduced model already provides (0.2103). We continue below with three different feature selection methods. 1. Stepwise Feature Selection 2. Forward Feature Selection 3. Backward Feature Selection

[ ]:
```
# Try stepwise feature selection
stepwise_model <- step(lm_hitting_salary_full, direction = "both")
summary(stepwise_model)

best_model <- regsubsets(salary ~ AB + R + H + X2B + X3B + HR + RBI + SB + CS +
 ↪BB + SO + IBB + HBP + SH + SF + GIDP,
                         data = cleaned_data, nvmax = 10)
summary(best_model)
```

```
Start:  AIC=1046632
salary ~ AB + R + H + X2B + X3B + HR + RBI + SB + CS + BB + SO +
    IBB + HBP + SH + SF + GIDP

       Df  Sum of Sq        RSS     AIC
- SF    1 6.9087e+11 2.5576e+17 1046630
- X2B   1 2.3896e+12 2.5576e+17 1046630
- H     1 5.9115e+12 2.5577e+17 1046631
- SO    1 1.4202e+13 2.5577e+17 1046632
<none>                2.5576e+17 1046632
- AB    1 1.2994e+14 2.5589e+17 1046648
- R     1 2.4982e+14 2.5601e+17 1046664
- RBI   1 2.9564e+14 2.5606e+17 1046671
- HBP   1 4.3062e+14 2.5619e+17 1046689
- HR    1 4.6594e+14 2.5623e+17 1046694
- SB    1 5.6091e+14 2.5632e+17 1046707
```

```
- BB    1 7.4373e+14 2.5650e+17 1046732
- GIDP  1 8.3820e+14 2.5660e+17 1046745
- IBB   1 1.2093e+15 2.5697e+17 1046797
- X3B   1 1.6839e+15 2.5744e+17 1046862
- CS    1 2.9651e+15 2.5873e+17 1047037
- SH    1 3.0662e+15 2.5883e+17 1047051

Step:  AIC=1046630
salary ~ AB + R + H + X2B + X3B + HR + RBI + SB + CS + BB + SO +
    IBB + HBP + SH + GIDP

       Df  Sum of Sq        RSS     AIC
- X2B   1 2.3588e+12 2.5576e+17 1046628
- H     1 5.5046e+12 2.5577e+17 1046629
<none>               2.5576e+17 1046630
- SO    1 1.5306e+13 2.5578e+17 1046630
+ SF    1 6.9087e+11 2.5576e+17 1046632
- AB    1 1.3610e+14 2.5590e+17 1046647
- R     1 2.5029e+14 2.5601e+17 1046662
- RBI   1 3.3777e+14 2.5610e+17 1046675
- HBP   1 4.3019e+14 2.5619e+17 1046687
- HR    1 4.8117e+14 2.5624e+17 1046694
- SB    1 5.6074e+14 2.5632e+17 1046705
- BB    1 7.4476e+14 2.5651e+17 1046731
- GIDP  1 8.3764e+14 2.5660e+17 1046743
- IBB   1 1.2091e+15 2.5697e+17 1046795
- X3B   1 1.6875e+15 2.5745e+17 1046860
- CS    1 2.9677e+15 2.5873e+17 1047036
- SH    1 3.0697e+15 2.5883e+17 1047050

Step:  AIC=1046628
salary ~ AB + R + H + X3B + HR + RBI + SB + CS + BB + SO + IBB +
    HBP + SH + GIDP

       Df  Sum of Sq        RSS     AIC
- H     1 3.9654e+12 2.5577e+17 1046627
<none>               2.5576e+17 1046628
- SO    1 1.6692e+13 2.5578e+17 1046629
+ X2B   1 2.3588e+12 2.5576e+17 1046630
+ SF    1 6.6005e+11 2.5576e+17 1046630
- AB    1 1.3784e+14 2.5590e+17 1046645
- R     1 2.4810e+14 2.5601e+17 1046660
- RBI   1 3.5421e+14 2.5612e+17 1046675
- HBP   1 4.2784e+14 2.5619e+17 1046685
- HR    1 5.0650e+14 2.5627e+17 1046696
- SB    1 5.7982e+14 2.5634e+17 1046706
- BB    1 7.4926e+14 2.5651e+17 1046730
- GIDP  1 8.4301e+14 2.5661e+17 1046743
```

```
- IBB   1 1.2095e+15 2.5697e+17 1046793
- X3B   1 1.6917e+15 2.5746e+17 1046859
- CS    1 2.9660e+15 2.5873e+17 1047034
- SH    1 3.0864e+15 2.5885e+17 1047050

Step:  AIC=1046627
salary ~ AB + R + X3B + HR + RBI + SB + CS + BB + SO + IBB +
    HBP + SH + GIDP

       Df  Sum of Sq        RSS     AIC
<none>               2.5577e+17 1046627
+ H     1 3.9654e+12 2.5576e+17 1046628
- SO    1 2.5772e+13 2.5579e+17 1046628
+ X2B   1 8.1955e+11 2.5577e+17 1046629
+ SF    1 3.0783e+11 2.5577e+17 1046629
- R     1 3.5919e+14 2.5613e+17 1046674
- RBI   1 3.6493e+14 2.5613e+17 1046675
- HBP   1 4.2670e+14 2.5619e+17 1046684
- AB    1 4.4624e+14 2.5621e+17 1046686
- HR    1 5.1500e+14 2.5628e+17 1046696
- SB    1 5.7622e+14 2.5634e+17 1046704
- BB    1 7.6481e+14 2.5653e+17 1046730
- GIDP  1 8.4872e+14 2.5662e+17 1046742
- IBB   1 1.2549e+15 2.5702e+17 1046798
- X3B   1 1.6877e+15 2.5746e+17 1046857
- CS    1 2.9621e+15 2.5873e+17 1047032
- SH    1 3.1635e+15 2.5893e+17 1047059


Call:
lm(formula = salary ~ AB + R + X3B + HR + RBI + SB + CS + BB +
    SO + IBB + HBP + SH + GIDP, data = cleaned_data)

Residuals:
     Min       1Q   Median       3Q      Max
-7795084 -1183657  -408604   254740 30075165

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  328273.9    29224.5  11.233  < 2e-16 ***
AB             2802.0      356.9   7.851 4.23e-15 ***
R             15176.0     2154.4   7.044 1.90e-12 ***
X3B         -137676.1     9016.7 -15.269  < 2e-16 ***
HR            40241.0     4770.9   8.435  < 2e-16 ***
RBI          -14077.8     1982.8  -7.100 1.27e-12 ***
SB            23823.4     2670.2   8.922  < 2e-16 ***
CS          -149982.4     7414.5 -20.228  < 2e-16 ***
BB            13551.7     1318.4  10.279  < 2e-16 ***
```

```
SO            -1601.4      848.7  -1.887   0.0592 .
IBB           70753.1     5373.8  13.166  < 2e-16 ***
HBP           40868.5     5323.1   7.678 1.66e-14 ***
SH          -131849.6     6307.1 -20.905  < 2e-16 ***
GIDP          51062.0     4715.8  10.828  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2691000 on 35332 degrees of freedom
Multiple R-squared:  0.2107,        Adjusted R-squared:  0.2104
F-statistic: 725.4 on 13 and 35332 DF,  p-value: < 2.2e-16


Subset selection object
Call: regsubsets.formula(salary ~ AB + R + H + X2B + X3B + HR + RBI +
    SB + CS + BB + SO + IBB + HBP + SH + SF + GIDP, data = cleaned_data,
    nvmax = 10)
16 Variables  (and intercept)
     Forced in Forced out
AB       FALSE      FALSE
R        FALSE      FALSE
H        FALSE      FALSE
X2B      FALSE      FALSE
X3B      FALSE      FALSE
HR       FALSE      FALSE
RBI      FALSE      FALSE
SB       FALSE      FALSE
CS       FALSE      FALSE
BB       FALSE      FALSE
SO       FALSE      FALSE
IBB      FALSE      FALSE
HBP      FALSE      FALSE
SH       FALSE      FALSE
SF       FALSE      FALSE
GIDP     FALSE      FALSE
1 subsets of each size up to 10
Selection Algorithm: exhaustive
          AB  R   H   X2B X3B HR  RBI SB  CS  BB  SO  IBB HBP SH  SF  GIDP
1  ( 1 )  " " " " " " " " " " "*" " " " " " " " " " " " " " " " " " " " "
2  ( 1 )  " " " " " " " " " " "*" " " " " " " "*" " " " " " " " " " " " "
3  ( 1 )  " " "*" " " " " " " " " " " " " "*" " " " " " " " " "*" " " " "
4  ( 1 )  " " "*" " " " " "*" " " " " " " "*" " " " " " " " " "*" " " " "
5  ( 1 )  " " "*" " " " " "*" " " " " " " "*" " " " " "*" " " "*" " " " "
6  ( 1 )  " " "*" " " " " "*" " " " " " " "*" " " " " "*" " " "*" " " "*"
7  ( 1 )  " " "*" " " " " "*" " " " " " " "*" "*" " " "*" " " "*" " " "*"
8  ( 1 )  " " "*" " " " " "*" " " " " " " "*" "*" " " "*" "*" "*" " " "*"
9  ( 1 )  " " "*" " " " " "*" " " " " "*" "*" "*" " " "*" "*" "*" " " "*"
10  ( 1 ) "*" "*" " " " " "*" " " " " "*" "*" "*" " " "*" "*" "*" " " "*"
```

[ ]:
```
# Try forward feature selection
# Start with an empty model
empty_model <- lm(salary ~ 1, data = cleaned_data)

# Full model with all predictors (from above)
lm_hitting_salary_full <- lm(salary ~ AB + R + H + X2B + X3B + HR + RBI + SB +
 ↪CS + BB + SO + IBB + HBP + SH + SF + GIDP, data = cleaned_data)

# Perform forward selection
forward_model <- step(empty_model,
                      scope = list(lower = empty_model, upper =
 ↪lm_hitting_salary_full),
                      direction = "forward")

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

```
Start:  AIC=1054962
salary ~ 1

       Df  Sum of Sq        RSS     AIC
+ HR    1 4.8757e+16 2.7527e+17 1049200
+ RBI   1 4.8298e+16 2.7573e+17 1049259
+ BB    1 4.2723e+16 2.8130e+17 1049967
+ R     1 4.0727e+16 2.8330e+17 1050217
+ H     1 3.7875e+16 2.8615e+17 1050571
+ X2B   1 3.7683e+16 2.8634e+17 1050594
+ AB    1 3.7183e+16 2.8684e+17 1050656
+ GIDP  1 3.3430e+16 2.9060e+17 1051115
+ SO    1 3.2198e+16 2.9183e+17 1051265
+ IBB   1 3.2154e+16 2.9187e+17 1051270
+ SF    1 2.3446e+16 3.0058e+17 1052309
+ HBP   1 1.6271e+16 3.0776e+17 1053143
+ SH    1 5.4902e+15 3.1854e+17 1054360
+ SB    1 1.8538e+15 3.2217e+17 1054761
+ X3B   1 1.0165e+15 3.2301e+17 1054853
+ CS    1 5.0038e+14 3.2353e+17 1054910
<none>                3.2403e+17 1054962

Step:  AIC=1049200
salary ~ HR

       Df  Sum of Sq        RSS     AIC
+ BB    1 4.8779e+15 2.7039e+17 1048570
+ GIDP  1 4.8275e+15 2.7044e+17 1048577
+ IBB   1 4.1813e+15 2.7109e+17 1048661
+ X2B   1 2.9068e+15 2.7236e+17 1048827
```

```
+ H       1 2.7056e+15 2.7257e+17 1048853
+ AB      1 2.4942e+15 2.7278e+17 1048880
+ RBI     1 2.2759e+15 2.7299e+17 1048909
+ R       1 2.2250e+15 2.7305e+17 1048915
+ SH      1 1.3977e+15 2.7387e+17 1049022
+ SF      1 1.1216e+15 2.7415e+17 1049058
+ HBP     1 1.0890e+15 2.7418e+17 1049062
+ CS      1 7.0561e+14 2.7457e+17 1049112
+ X3B     1 5.1405e+14 2.7476e+17 1049136
+ SO      1 2.8949e+14 2.7498e+17 1049165
<none>                 2.7527e+17 1049200
+ SB      1 1.2003e+13 2.7526e+17 1049201

Step:  AIC=1048570
salary ~ HR + BB

       Df  Sum of Sq        RSS     AIC
+ CS    1 3.6056e+15 2.6679e+17 1048098
+ SH    1 3.5437e+15 2.6685e+17 1048106
+ GIDP  1 2.7495e+15 2.6764e+17 1048211
+ X3B   1 2.2177e+15 2.6818e+17 1048281
+ IBB   1 1.7723e+15 2.6862e+17 1048340
+ SB    1 6.6113e+14 2.6973e+17 1048486
+ X2B   1 5.4903e+14 2.6984e+17 1048500
+ HBP   1 4.5327e+14 2.6994e+17 1048513
+ RBI   1 2.8385e+14 2.7011e+17 1048535
+ H     1 2.5206e+14 2.7014e+17 1048539
+ SF    1 1.7684e+14 2.7022e+17 1048549
+ AB    1 1.5081e+14 2.7024e+17 1048553
+ SO    1 1.4976e+14 2.7024e+17 1048553
<none>                 2.7039e+17 1048570
+ R     1 5.4041e+11 2.7039e+17 1048572

Step:  AIC=1048098
salary ~ HR + BB + CS

       Df  Sum of Sq        RSS     AIC
+ GIDP  1 3.1280e+15 2.6366e+17 1047683
+ H     1 2.3038e+15 2.6448e+17 1047793
+ AB    1 1.9399e+15 2.6485e+17 1047842
+ SH    1 1.7811e+15 2.6501e+17 1047863
+ X2B   1 1.6081e+15 2.6518e+17 1047886
+ R     1 1.5414e+15 2.6525e+17 1047895
+ IBB   1 1.3146e+15 2.6547e+17 1047925
+ RBI   1 9.2087e+14 2.6587e+17 1047978
+ HBP   1 7.8401e+14 2.6600e+17 1047996
+ SB    1 6.2260e+14 2.6616e+17 1048017
+ X3B   1 5.0402e+14 2.6628e+17 1048033
```

```
+ SF    1 3.3960e+14 2.6645e+17 1048055
<none>               2.6679e+17 1048098
+ SO    1 2.9850e+12 2.6678e+17 1048099

Step:  AIC=1047683
salary ~ HR + BB + CS + GIDP

       Df  Sum of Sq        RSS     AIC
+ SH    1 2.4262e+15 2.6123e+17 1047358
+ IBB   1 1.2254e+15 2.6243e+17 1047520
+ SB    1 8.7400e+14 2.6279e+17 1047568
+ X3B   1 6.7525e+14 2.6298e+17 1047594
+ HBP   1 4.9213e+14 2.6317e+17 1047619
+ R     1 3.6812e+14 2.6329e+17 1047636
+ H     1 3.2605e+14 2.6333e+17 1047641
+ X2B   1 2.4324e+14 2.6342e+17 1047652
+ AB    1 1.5335e+14 2.6351e+17 1047664
+ SO    1 3.2388e+13 2.6363e+17 1047681
<none>               2.6366e+17 1047683
+ SF    1 2.3837e+12 2.6366e+17 1047685
+ RBI   1 7.0677e+11 2.6366e+17 1047685

Step:  AIC=1047358
salary ~ HR + BB + CS + GIDP + SH

       Df  Sum of Sq        RSS     AIC
+ R     1 1.1144e+15 2.6012e+17 1047209
+ SB    1 1.0769e+15 2.6016e+17 1047214
+ H     1 9.8758e+14 2.6025e+17 1047226
+ IBB   1 9.1020e+14 2.6032e+17 1047237
+ AB    1 8.9311e+14 2.6034e+17 1047239
+ HBP   1 7.5072e+14 2.6048e+17 1047258
+ X2B   1 4.4312e+14 2.6079e+17 1047300
+ X3B   1 3.7922e+14 2.6085e+17 1047309
+ RBI   1 1.7383e+13 2.6122e+17 1047358
+ SF    1 1.6324e+13 2.6122e+17 1047358
<none>               2.6123e+17 1047358
+ SO    1 3.8506e+11 2.6123e+17 1047360

Step:  AIC=1047209
salary ~ HR + BB + CS + GIDP + SH + R

       Df  Sum of Sq        RSS     AIC
+ X3B   1 1.4391e+15 2.5868e+17 1047015
+ IBB   1 1.1073e+15 2.5901e+17 1047060
+ HBP   1 4.5075e+14 2.5967e+17 1047150
+ SB    1 4.3418e+14 2.5968e+17 1047152
+ RBI   1 1.5150e+14 2.5997e+17 1047190
```

```
+ AB     1 1.0680e+14 2.6001e+17 1047197
+ H      1 8.0012e+13 2.6004e+17 1047200
+ SO     1 1.7139e+13 2.6010e+17 1047209
<none>                2.6012e+17 1047209
+ SF     1 7.7582e+12 2.6011e+17 1047210
+ X2B    1 4.6803e+12 2.6011e+17 1047210

Step:  AIC=1047015
salary ~ HR + BB + CS + GIDP + SH + R + X3B

       Df  Sum of Sq        RSS     AIC
+ IBB   1 1.2089e+15 2.5747e+17 1046851
+ SB    1 6.1172e+14 2.5807e+17 1046933
+ HBP   1 3.5817e+14 2.5832e+17 1046968
+ AB    1 2.4097e+14 2.5844e+17 1046984
+ H     1 1.8659e+14 2.5849e+17 1046991
+ RBI   1 9.5597e+13 2.5858e+17 1047004
<none>               2.5868e+17 1047015
+ SF    1 8.6595e+12 2.5867e+17 1047016
+ X2B   1 1.2148e+12 2.5868e+17 1047017
+ SO    1 9.5975e+10 2.5868e+17 1047017

Step:  AIC=1046851
salary ~ HR + BB + CS + GIDP + SH + R + X3B + IBB

       Df  Sum of Sq        RSS     AIC
+ SB    1 6.0100e+14 2.5687e+17 1046771
+ HBP   1 4.5818e+14 2.5701e+17 1046790
+ RBI   1 2.1727e+14 2.5725e+17 1046824
+ AB    1 2.0972e+14 2.5726e+17 1046825
+ SO    1 6.5857e+13 2.5740e+17 1046844
+ H     1 5.7606e+13 2.5741e+17 1046845
+ SF    1 2.7093e+13 2.5744e+17 1046850
<none>               2.5747e+17 1046851
+ X2B   1 1.9718e+12 2.5747e+17 1046853

Step:  AIC=1046771
salary ~ HR + BB + CS + GIDP + SH + R + X3B + IBB + SB

       Df  Sum of Sq        RSS     AIC
+ HBP   1 5.0392e+14 2.5637e+17 1046703
+ AB    1 2.7407e+14 2.5660e+17 1046735
+ RBI   1 1.2502e+14 2.5674e+17 1046756
+ H     1 1.1514e+14 2.5675e+17 1046757
+ SO    1 6.9874e+13 2.5680e+17 1046763
<none>               2.5687e+17 1046771
+ SF    1 1.2439e+13 2.5686e+17 1046771
+ X2B   1 1.0586e+13 2.5686e+17 1046771
```

```
Step:  AIC=1046703
salary ~ HR + BB + CS + GIDP + SH + R + X3B + IBB + SB + HBP

       Df  Sum of Sq        RSS     AIC
+ AB    1 2.2903e+14 2.5614e+17 1046674
+ RBI   1 1.1110e+14 2.5625e+17 1046690
+ H     1 1.0943e+14 2.5626e+17 1046690
+ SO    1 3.0897e+13 2.5633e+17 1046701
<none>               2.5637e+17 1046703
+ SF    1 8.2959e+12 2.5636e+17 1046704
+ X2B   1 2.4865e+12 2.5636e+17 1046705

Step:  AIC=1046674
salary ~ HR + BB + CS + GIDP + SH + R + X3B + IBB + SB + HBP +
    AB

       Df  Sum of Sq        RSS     AIC
+ RBI   1 3.4352e+14 2.5579e+17 1046628
+ SF    1 4.1873e+13 2.5609e+17 1046670
+ X2B   1 2.7895e+13 2.5611e+17 1046672
<none>               2.5614e+17 1046674
+ H     1 7.7226e+12 2.5613e+17 1046675
+ SO    1 4.3721e+12 2.5613e+17 1046675

Step:  AIC=1046628
salary ~ HR + BB + CS + GIDP + SH + R + X3B + IBB + SB + HBP +
    AB + RBI

       Df  Sum of Sq        RSS     AIC
+ SO    1 2.5772e+13 2.5577e+17 1046627
<none>               2.5579e+17 1046628
+ H     1 1.3045e+13 2.5578e+17 1046629
+ SF    1 1.0532e+12 2.5579e+17 1046630
+ X2B   1 8.0339e+11 2.5579e+17 1046630

Step:  AIC=1046627
salary ~ HR + BB + CS + GIDP + SH + R + X3B + IBB + SB + HBP +
    AB + RBI + SO

       Df  Sum of Sq        RSS     AIC
<none>               2.5577e+17 1046627
+ H     1 3.9654e+12 2.5576e+17 1046628
+ X2B   1 8.1955e+11 2.5577e+17 1046629
+ SF    1 3.0783e+11 2.5577e+17 1046629


Call:
```

```
lm(formula = salary ~ HR + BB + CS + GIDP + SH + R + X3B + IBB +
    SB + HBP + AB + RBI + SO, data = cleaned_data)

Residuals:
     Min       1Q   Median       3Q      Max
-7795084 -1183657  -408604   254740 30075165

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  328273.9    29224.5  11.233  < 2e-16 ***
HR            40241.0     4770.9   8.435  < 2e-16 ***
BB            13551.7     1318.4  10.279  < 2e-16 ***
CS          -149982.4     7414.5 -20.228  < 2e-16 ***
GIDP          51062.0     4715.8  10.828  < 2e-16 ***
SH          -131849.6     6307.1 -20.905  < 2e-16 ***
R             15176.0     2154.4   7.044 1.90e-12 ***
X3B         -137676.1     9016.7 -15.269  < 2e-16 ***
IBB           70753.1     5373.8  13.166  < 2e-16 ***
SB            23823.4     2670.2   8.922  < 2e-16 ***
HBP           40868.5     5323.1   7.678 1.66e-14 ***
AB             2802.0      356.9   7.851 4.23e-15 ***
RBI          -14077.8     1982.8  -7.100 1.27e-12 ***
SO            -1601.4      848.7  -1.887   0.0592 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2691000 on 35332 degrees of freedom
Multiple R-squared:  0.2107,	Adjusted R-squared:  0.2104
F-statistic: 725.4 on 13 and 35332 DF,  p-value: < 2.2e-16
```

[ ]:

```
# Try backward feature selection
# Full model with all predictors
lm_hitting_salary_full <- lm(salary ~ AB + R + H + X2B + X3B + HR + RBI + SB +
 CS + BB + SO + IBB + HBP + SH + SF + GIDP, data = cleaned_data)

# Perform backward selection
backward_model <- step(lm_hitting_salary_full, direction = "backward")

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

```
Start:  AIC=1046632
salary ~ AB + R + H + X2B + X3B + HR + RBI + SB + CS + BB + SO +
    IBB + HBP + SH + SF + GIDP

       Df  Sum of Sq        RSS     AIC
- SF    1 6.9087e+11 2.5576e+17 1046630
```

```
- X2B   1 2.3896e+12 2.5576e+17 1046630
- H     1 5.9115e+12 2.5577e+17 1046631
- SO    1 1.4202e+13 2.5577e+17 1046632
<none>               2.5576e+17 1046632
- AB    1 1.2994e+14 2.5589e+17 1046648
- R     1 2.4982e+14 2.5601e+17 1046664
- RBI   1 2.9564e+14 2.5606e+17 1046671
- HBP   1 4.3062e+14 2.5619e+17 1046689
- HR    1 4.6594e+14 2.5623e+17 1046694
- SB    1 5.6091e+14 2.5632e+17 1046707
- BB    1 7.4373e+14 2.5650e+17 1046732
- GIDP  1 8.3820e+14 2.5660e+17 1046745
- IBB   1 1.2093e+15 2.5697e+17 1046797
- X3B   1 1.6839e+15 2.5744e+17 1046862
- CS    1 2.9651e+15 2.5873e+17 1047037
- SH    1 3.0662e+15 2.5883e+17 1047051

Step:  AIC=1046630
salary ~ AB + R + H + X2B + X3B + HR + RBI + SB + CS + BB + SO +
    IBB + HBP + SH + GIDP

       Df  Sum of Sq        RSS     AIC
- X2B   1 2.3588e+12 2.5576e+17 1046628
- H     1 5.5046e+12 2.5577e+17 1046629
<none>               2.5576e+17 1046630
- SO    1 1.5306e+13 2.5578e+17 1046630
- AB    1 1.3610e+14 2.5590e+17 1046647
- R     1 2.5029e+14 2.5601e+17 1046662
- RBI   1 3.3777e+14 2.5610e+17 1046675
- HBP   1 4.3019e+14 2.5619e+17 1046687
- HR    1 4.8117e+14 2.5624e+17 1046694
- SB    1 5.6074e+14 2.5632e+17 1046705
- BB    1 7.4476e+14 2.5651e+17 1046731
- GIDP  1 8.3764e+14 2.5660e+17 1046743
- IBB   1 1.2091e+15 2.5697e+17 1046795
- X3B   1 1.6875e+15 2.5745e+17 1046860
- CS    1 2.9677e+15 2.5873e+17 1047036
- SH    1 3.0697e+15 2.5883e+17 1047050

Step:  AIC=1046628
salary ~ AB + R + H + X3B + HR + RBI + SB + CS + BB + SO + IBB +
    HBP + SH + GIDP

       Df  Sum of Sq        RSS     AIC
- H     1 3.9654e+12 2.5577e+17 1046627
<none>               2.5576e+17 1046628
- SO    1 1.6692e+13 2.5578e+17 1046629
- AB    1 1.3784e+14 2.5590e+17 1046645
```

```
- R     1 2.4810e+14 2.5601e+17 1046660
- RBI   1 3.5421e+14 2.5612e+17 1046675
- HBP   1 4.2784e+14 2.5619e+17 1046685
- HR    1 5.0650e+14 2.5627e+17 1046696
- SB    1 5.7982e+14 2.5634e+17 1046706
- BB    1 7.4926e+14 2.5651e+17 1046730
- GIDP  1 8.4301e+14 2.5661e+17 1046743
- IBB   1 1.2095e+15 2.5697e+17 1046793
- X3B   1 1.6917e+15 2.5746e+17 1046859
- CS    1 2.9660e+15 2.5873e+17 1047034
- SH    1 3.0864e+15 2.5885e+17 1047050

Step:  AIC=1046627
salary ~ AB + R + X3B + HR + RBI + SB + CS + BB + SO + IBB +
    HBP + SH + GIDP

       Df  Sum of Sq        RSS     AIC
<none>               2.5577e+17 1046627
- SO    1 2.5772e+13 2.5579e+17 1046628
- R     1 3.5919e+14 2.5613e+17 1046674
- RBI   1 3.6493e+14 2.5613e+17 1046675
- HBP   1 4.2670e+14 2.5619e+17 1046684
- AB    1 4.4624e+14 2.5621e+17 1046686
- HR    1 5.1500e+14 2.5628e+17 1046696
- SB    1 5.7622e+14 2.5634e+17 1046704
- BB    1 7.6481e+14 2.5653e+17 1046730
- GIDP  1 8.4872e+14 2.5662e+17 1046742
- IBB   1 1.2549e+15 2.5702e+17 1046798
- X3B   1 1.6877e+15 2.5746e+17 1046857
- CS    1 2.9621e+15 2.5873e+17 1047032
- SH    1 3.1635e+15 2.5893e+17 1047059


Call:
lm(formula = salary ~ AB + R + X3B + HR + RBI + SB + CS + BB +
    SO + IBB + HBP + SH + GIDP, data = cleaned_data)

Residuals:
     Min       1Q   Median       3Q      Max
-7795084 -1183657  -408604   254740 30075165

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  328273.9    29224.5  11.233  < 2e-16 ***
AB             2802.0      356.9   7.851 4.23e-15 ***
R             15176.0     2154.4   7.044 1.90e-12 ***
X3B         -137676.1     9016.7 -15.269  < 2e-16 ***
HR            40241.0     4770.9   8.435  < 2e-16 ***
```

```
RBI           -14077.8     1982.8  -7.100 1.27e-12 ***
SB             23823.4     2670.2   8.922  < 2e-16 ***
CS           -149982.4     7414.5 -20.228  < 2e-16 ***
BB             13551.7     1318.4  10.279  < 2e-16 ***
SO             -1601.4      848.7  -1.887   0.0592 .
IBB            70753.1     5373.8  13.166  < 2e-16 ***
HBP            40868.5     5323.1   7.678 1.66e-14 ***
SH           -131849.6     6307.1 -20.905  < 2e-16 ***
GIDP           51062.0     4715.8  10.828  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2691000 on 35332 degrees of freedom
Multiple R-squared:  0.2107,	Adjusted R-squared:  0.2104
F-statistic: 725.4 on 13 and 35332 DF,  p-value: < 2.2e-16
```

Notice that the summary output of the model after each selection method has an Adjusted R^2 of 0.2104, which is higher by only 0.0001. We can see there is a very small improvement from the reduced model in the ANOVA test above.

After conducting each feature selection method, we are interested in looking at the final predictors that each method provided for predicting salary. These are shown below.

[ ]:
```r
# Compare predictors from the three selection methods

cat("Predictors from Forward Selection \n")
# Get the formula of the final model
final_formula <- formula(forward_model)
# Extract the predictors (excluding the intercept term)
predictors <- all.vars(final_formula)[-1]  # Remove the intercept term
# Print the final predictors
print(predictors)

cat("Predictors from Backward Selection \n")
# Get the formula of the final model
final_formula <- formula(backward_model)
# Extract the predictors (excluding the intercept term)
predictors <- all.vars(final_formula)[-1]  # Remove the intercept term
# Print the final predictors
print(predictors)

cat("Predictors from Stepwise Selection \n")
# Get the formula of the stepwise model
final_formula <- formula(stepwise_model)
# Extract the predictor names (excluding the intercept term)
predictor_names <- all.vars(final_formula)[-1]  # Removing the intercept
# Print the final predictors
```

```
print(predictor_names)
```

```
Predictors from Forward Selection
 [1] "HR"   "BB"   "CS"   "GIDP" "SH"   "R"    "X3B"  "IBB"  "SB"   "HBP"
[11] "AB"   "RBI"  "SO"
Predictors from Backward Selection
 [1] "AB"   "R"    "X3B"  "HR"   "RBI"  "SB"   "CS"   "BB"   "SO"   "IBB"
[11] "HBP"  "SH"   "GIDP"
Predictors from Stepwise Selection
 [1] "AB"   "R"    "X3B"  "HR"   "RBI"  "SB"   "CS"   "BB"   "SO"   "IBB"
[11] "HBP"  "SH"   "GIDP"
```

From the three selection methods, Backward and Stepwise Selection have exactly the same 13 final predictors:

1. At-Bats (AB)
2. Runs Scored (R)
3. Triples (X3B)
4. Homeruns (HR)
5. Runs Batting In (RBI)
6. Stolen Bases (SB)
7. Caught Stealing (CS)
8. Walks (BB)
9. Strikeouts (SO)
10. Intentional Walks (IBB)
11. Hit By Pitch (HBP)
12. Sacrifice Hit (SH)
13. Grounded Into Double Play (GIDP)

These predictors differ from the reduced model predictors. So we would want to select this model over the reduced model since we know it has a slightly higher Adjusted R^2 value.

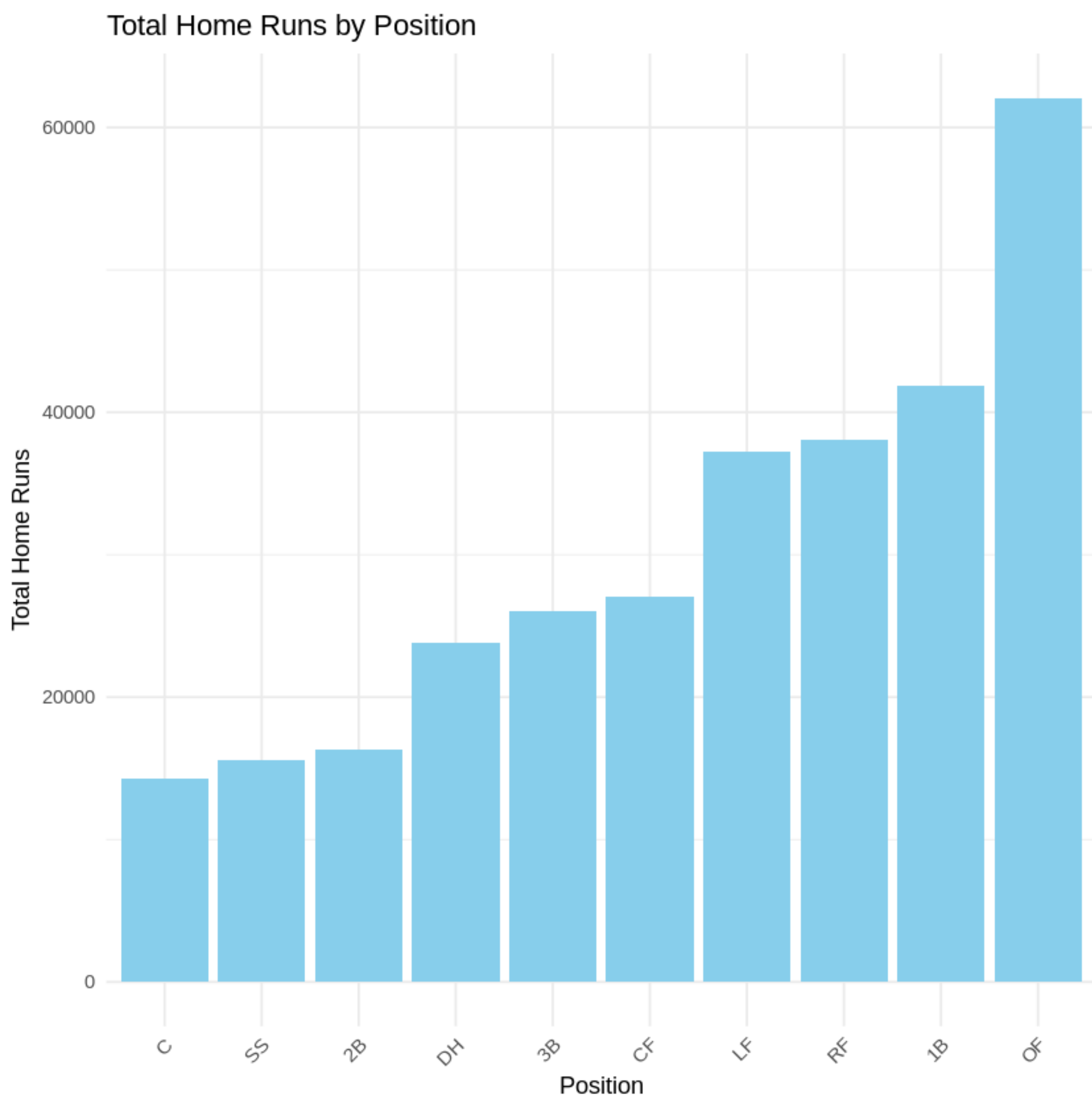
The Forward Selection method selected different features for the final model, but each method stopped when the model had an `AIC` value of which is 1046627.

8.0.5 Which position has most Homeruns (HR), has most triples (X3B), has most doubles (X2B), and has the most runs scored (R)?

Lastly, we became really curious about which position has the most of each hitting result. Below, we used simple bar charts to display in ascending order of which positions had the most of each hitting result.

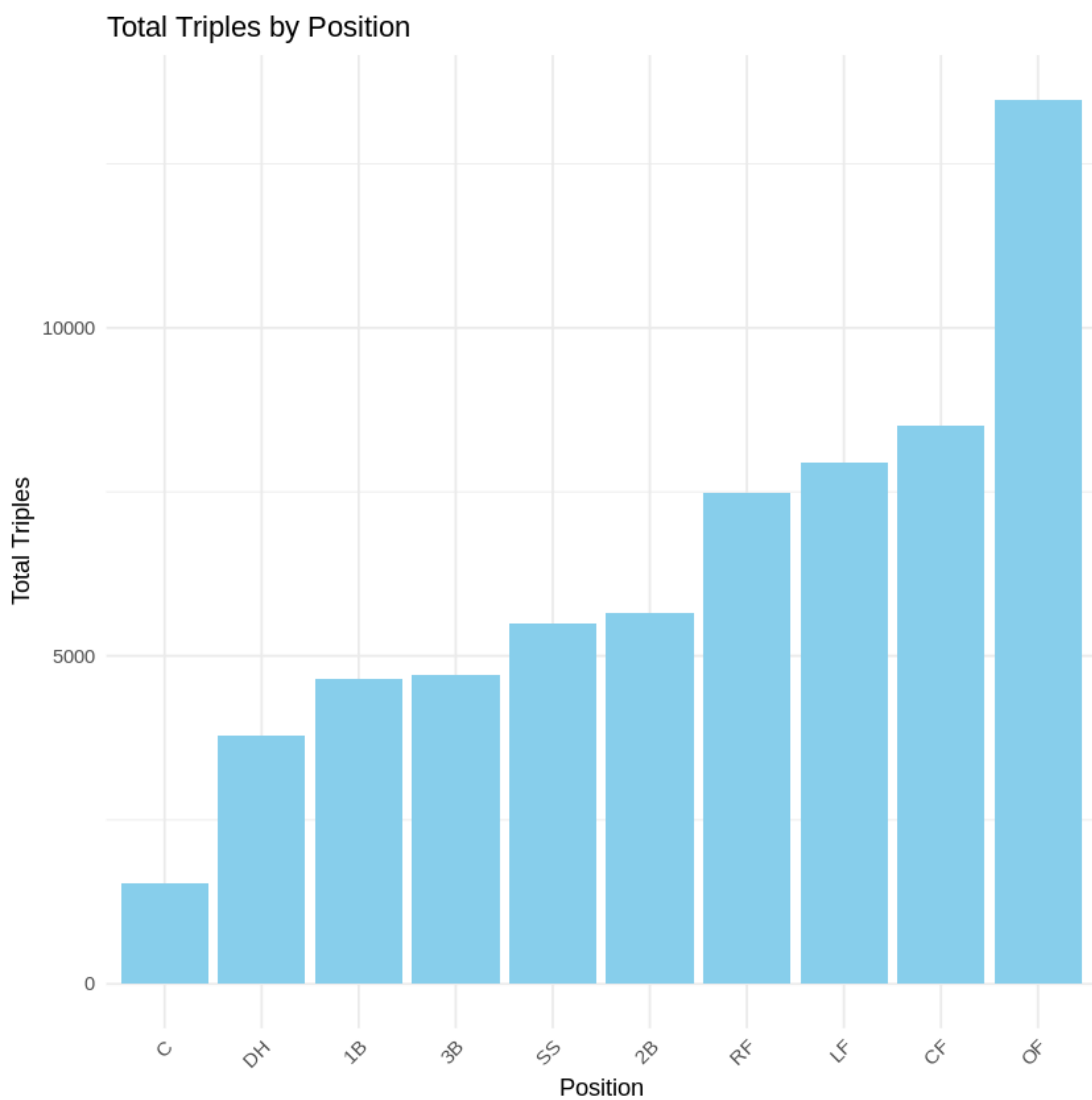
[ ]:
```
# Group data by position and sum home runs (HR)
hr_by_position <- cleaned_data %>%
  group_by(POS) %>%
  summarise(total_HR = sum(HR, na.rm = TRUE)) %>%
  arrange(desc(total_HR))  # Sort in descending order of home runs

# Create a bar plot
```

```
ggplot(hr_by_position, aes(x = reorder(POS, total_HR), y = total_HR, fill =␣
↪POS)) +
  geom_bar(stat = "identity", fill = "skyblue") +
  labs(
    title = "Total Home Runs by Position",
    x = "Position",
    y = "Total Home Runs"
  ) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))  # Rotate x-axis␣
↪labels for better readability
```



[ ]:
```r
# Group by position and calculate total triples (X3B) per position
triples_by_position <- cleaned_data %>%
  group_by(POS) %>%
  summarise(total_triples = sum(X3B, na.rm = TRUE)) %>%
  arrange(desc(total_triples))  # Sort in descending order of triples

# View the results
print(triples_by_position)

# Create a bar plot
ggplot(triples_by_position, aes(x = reorder(POS, total_triples), y =
 ↪total_triples, fill = POS)) +
  geom_bar(stat = "identity", fill = "skyblue") +
  labs(
    title = "Total Triples by Position",
    x = "Position",
    y = "Total Triples"
  ) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))  # Rotate x-axis
 ↪labels for better readability
```

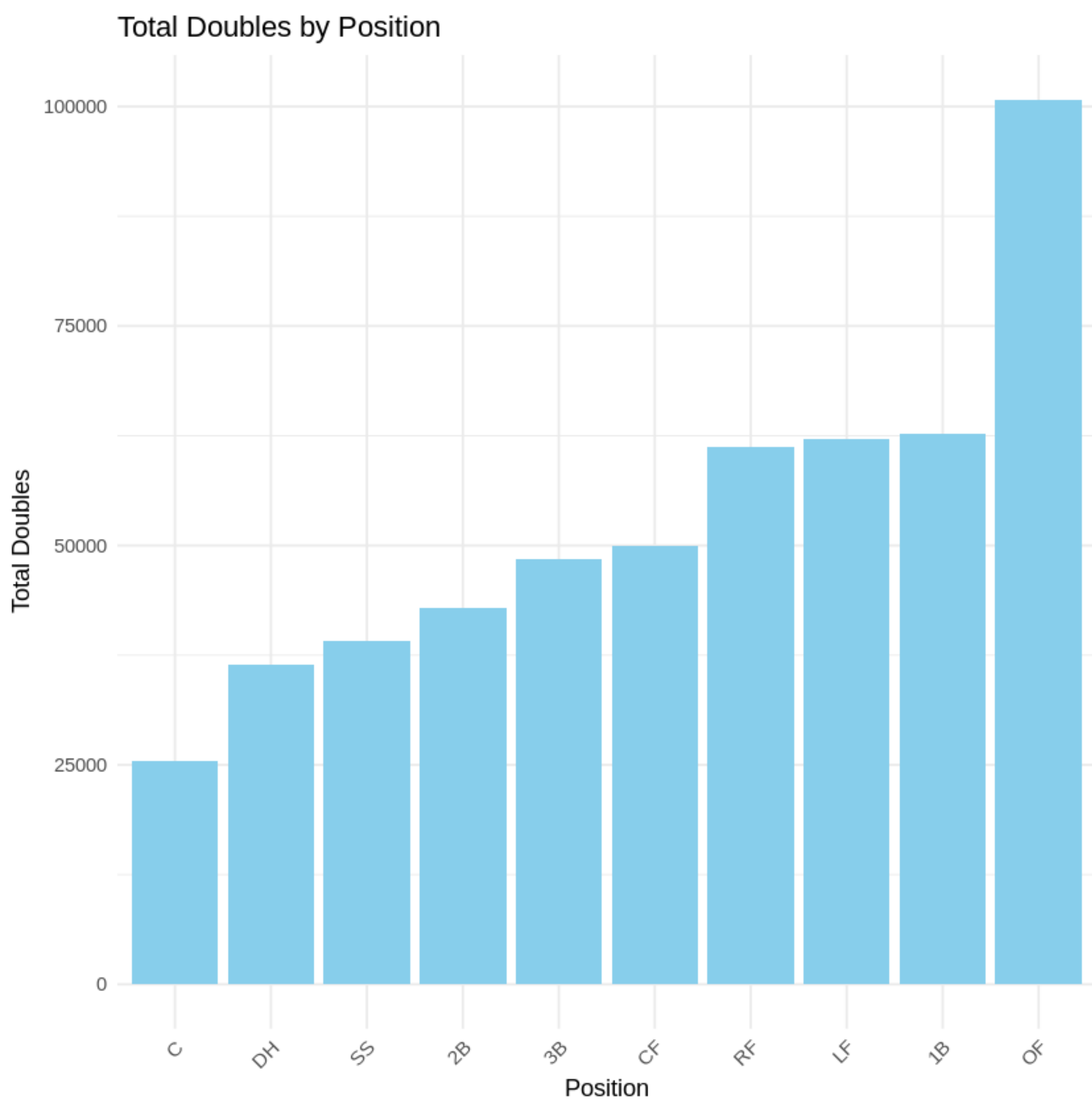
```
# A tibble: 10 × 2
   POS   total_triples
   <chr>         <int>
 1 OF            13484
 2 CF             8504
 3 LF             7948
 4 RF             7493
 5 2B             5652
 6 SS             5499
 7 3B             4711
 8 1B             4649
 9 DH             3794
10 C              1523
```



[ ]:
```r
# Group by position and calculate total doubles (X2B) per position
doubles_by_position <- cleaned_data %>%
  group_by(POS) %>%
  summarise(total_doubles = sum(X2B, na.rm = TRUE)) %>%
  arrange(desc(total_doubles))  # Sort in descending order of doubles

# View the results
print(doubles_by_position)

# Create a bar plot
ggplot(doubles_by_position, aes(x = reorder(POS, total_doubles), y =
 ↪total_doubles, fill = POS)) +
  geom_bar(stat = "identity", fill = "skyblue") +
```

```
  labs(
    title = "Total Doubles by Position",
    x = "Position",
    y = "Total Doubles"
  ) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))  # Rotate x-axis␣
↪labels for better readability
```

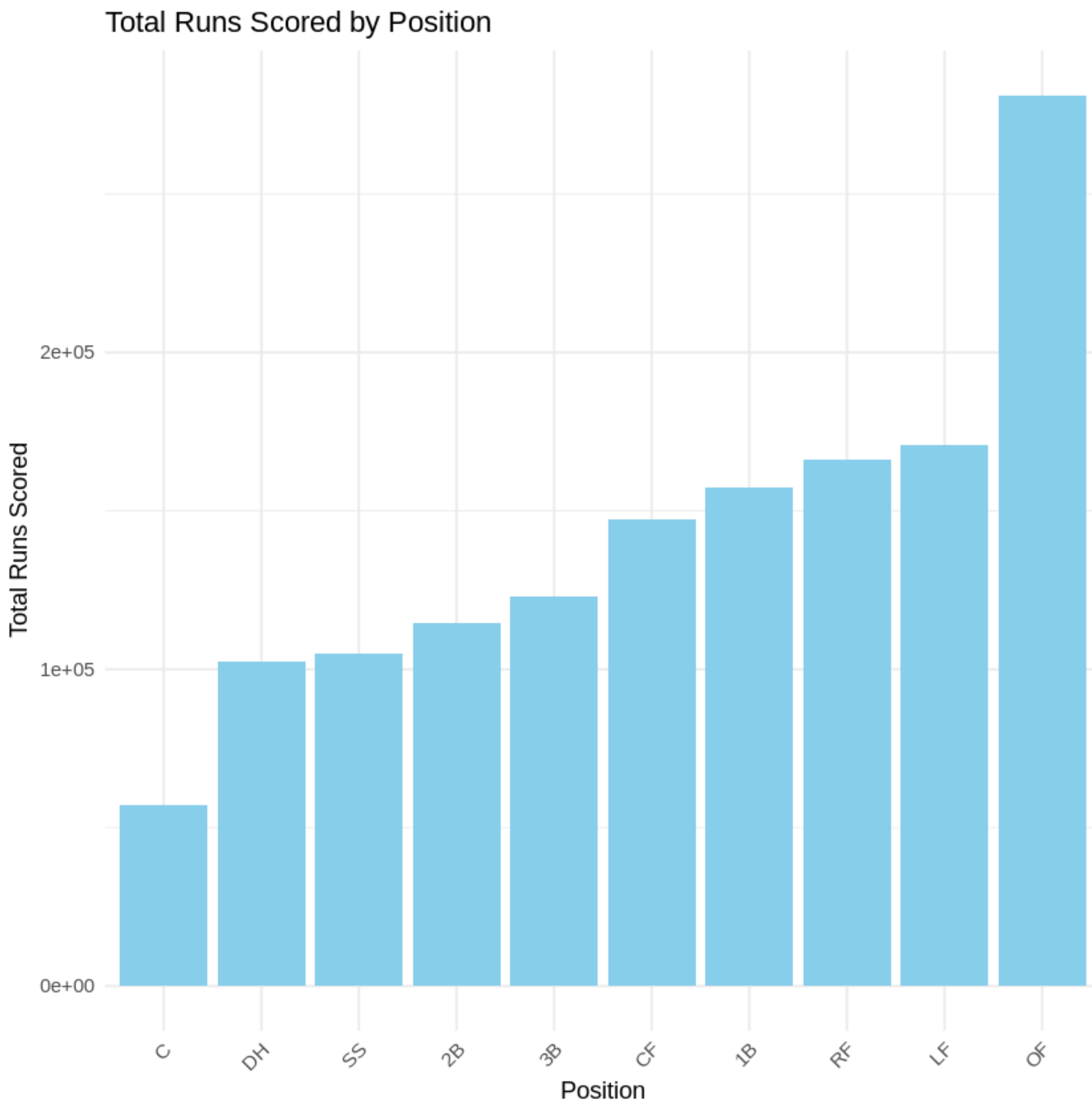
```
# A tibble: 10 × 2
   POS   total_doubles
   <chr>         <int>
 1 OF           100814
 2 1B            62677
 3 LF            62191
 4 RF            61186
 5 CF            49933
 6 3B            48479
 7 2B            42891
 8 SS            39096
 9 DH            36351
10 C             25427
```



[ ]:
```
# Group by position and calculate total runs (R) per position
runs_by_position <- cleaned_data %>%
  group_by(POS) %>%
  summarise(total_runs = sum(R, na.rm = TRUE)) %>%
  arrange(desc(total_runs))  # Sort in descending order of runs

# View the results
print(runs_by_position)

# Create a bar plot
ggplot(runs_by_position, aes(x = reorder(POS, total_runs), y = total_runs, fill␣
↪= POS)) +
  geom_bar(stat = "identity", fill = "skyblue") +
```

```
  labs(
    title = "Total Runs Scored by Position",
    x = "Position",
    y = "Total Runs Scored"
  ) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))  # Rotate x-axis
↪labels for better readability
```

```
# A tibble: 10 × 2
   POS   total_runs
   <chr>      <int>
 1 OF        281251
 2 LF        170893
 3 RF        166212
 4 1B        157337
 5 CF        147148
 6 3B        123140
 7 2B        114784
 8 SS        104954
 9 DH        102280
10 C          57009
```



After viewing the bar charts for positions with the most and least homeruns, triples, doubles, and runs scored, it is very obvious to recognize that the Outfield (OF) position is the highest for every graph. The OF position includes players that play any outfield position. So, these players are not double counted in the Right Field (RF), Center Field (CF), and Left Field (LF) positions. Rather, we can tell from this bar chart that many outfielders are considered for all outfield positions than just a singular specificed outfield position.

Now that we understand why the Outfield (OF) position is the highest in each bar chart, let us look at the other positions.

Excluding the OF position, we make the following observations. * The position with the highest number of Homeruns (HR) is First Base (1B). * The position with the highest number of Triples (X3B) is Center Field (CF). * The position with the highest number of Doubles (X2B) is First Base (1B). * The position with the highest number of Runs Scored (R) is Left Field (LF)

- The position with the lowest number of Homeruns (HR), Triples (X3B) is Catcher (C), Doubles (X2B), and Runs Scored is Catcher (C).

We are most surprised that the First Base (1B) position has the highest in both Homeruns (HR) and Doubles (X2B). We expected all of them to have an outfielder position with the most of each one.

```
[ ]:
```