Project Code (Analysis and Interpretation Included in Presentation)

George Daher

3/28/2023

```r
# Function that merges gamelogs and performance splits
# Calls Helper Function for Each of 9 lineup positions and both teams
mergeLogsSplitsBref <- function(logs,splits) {
  logs <- mergeLogSplitsBrefPosition(logs,splits,1,"visitor")
  logs <- mergeLogSplitsBrefPosition(logs,splits,2,"visitor")
  logs <- mergeLogSplitsBrefPosition(logs,splits,3,"visitor")
  logs <- mergeLogSplitsBrefPosition(logs,splits,4,"visitor")
  logs <- mergeLogSplitsBrefPosition(logs,splits,5,"visitor")
  logs <- mergeLogSplitsBrefPosition(logs,splits,6,"visitor")
  logs <- mergeLogSplitsBrefPosition(logs,splits,7,"visitor")
  logs <- mergeLogSplitsBrefPosition(logs,splits,8,"visitor")
  logs <- mergeLogSplitsBrefPosition(logs,splits,9,"visitor")
  logs <- mergeLogSplitsBrefPosition(logs,splits,1,"home")
  logs <- mergeLogSplitsBrefPosition(logs,splits,2,"home")
  logs <- mergeLogSplitsBrefPosition(logs,splits,3,"home")
  logs <- mergeLogSplitsBrefPosition(logs,splits,4,"home")
  logs <- mergeLogSplitsBrefPosition(logs,splits,5,"home")
  logs <- mergeLogSplitsBrefPosition(logs,splits,6,"home")
  logs <- mergeLogSplitsBrefPosition(logs,splits,7,"home")
  logs <- mergeLogSplitsBrefPosition(logs,splits,8,"home")
  logs <- mergeLogSplitsBrefPosition(logs,splits,9,"home")
}

# Merges the datasets after handling exceptions explicitly
mergeLogSplitsBrefPosition <- function(logs,splits,num,team) {
  colnames(splits) <- paste0(team,num,colnames(splits))
  logs[,paste0(team,num,"Name")] <- gsub("\\.","",logs[,paste0(team,num,"Name")])
  logs[,paste0(team,num,"Name")] <- gsub("i-M","i M",logs[,paste0(team,num,"Name")])
  logs[,paste0(team,num,"Name")] <- gsub("n-J","n J",logs[,paste0(team,num,"Name")])
  logs[,paste0(team,num,"Name")] <- gsub("Dee Gordon","Dee Strange-Gordon",logs[,paste0(team,num,"Name"
  logs[,paste0(team,num,"Name")] <- gsub("Giovanny Urshela","Gio Urshela",logs[,paste0(team,num,"Name")
  logs[,paste0(team,num,"Name")] <- gsub("Michael Taylor","Michael A. Taylor",logs[,paste0(team,num,"Na
  logs[,paste0(team,num,"Name")] <- gsub("Vincent Velasquez","Vince Velasquez",logs[,paste0(team,num,"N
  logs[,paste0(team,num,"Name")] <- gsub("Michael Brosseau","Mike Brosseau",logs[,paste0(team,num,"Name
  logs[,paste0(team,num,"Name")] <- gsub("Nate Lowe","Nathanial Lowe",logs[,paste0(team,num,"Name")])
  logs[,paste0(team,num,"Name")] <- gsub("Phillip Ervin","Phil Ervin",logs[,paste0(team,num,"Name")])
  logs[,paste0(team,num,"Name")] <- gsub("Josh Fuentes","Joshua Fuentes",logs[,paste0(team,num,"Name")]
  logs[,paste0(team,num,"Name")] <- gsub("Yulieski Gurriel","Yuli Gurriel",logs[,paste0(team,num,"Name"
  logs[,paste0(team,num,"Name")] <- gsub("Steve Wilkerson","Stevie Wilkerson",logs[,paste0(team,num,"Na
  logs[,paste0(team,num,"Name")] <- gsub("Mike Soroka","Michael Soroka",logs[,paste0(team,num,"Name")])
  return(merge(x=logs,y=splits,by.x=paste0(team,num,"Name"),by.y=paste0(team,num,"Name")))
}
```

Background

All the data used is pulled from either baseball reference using the baseballr R package, or from Retrosheet game logs. All the data in both databases is complete for the years (2012 to 2019) that we are considering in our analysis. The objective of our analysis is to determine whether there exists a predictive relationship between previous hitting statistics and order in the lineup for hitters/lineups in the MLB.

Cleaning the Data

Define helper functions to clean data (included in EDA)

```
library(baseballr); library(janitor); library(RcppParallel); library(lubridate);library(dplyr);library(
```

```
## Warning: package 'baseballr' was built under R version 4.3.3

## Warning: package 'janitor' was built under R version 4.3.3

## Warning: package 'RcppParallel' was built under R version 4.3.3

## Warning: package 'lubridate' was built under R version 4.3.3

## Warning: package 'dplyr' was built under R version 4.3.3

## Warning: package 'stringr' was built under R version 4.3.3
```

```
# Renames and Drops Columns in Gamelogs
cleanLogs <- function(logs) {
  outlogs <- logs[-c(2:3,12:89,94:105,160:161)]
  colnames(outlogs) <- c("Date","VisitingTeam","VisitingTeamLeague","VisitingGameNum","HomeTeam",
                         "HomeTeamLeague","HomeGameNum","VisitingScore","HomeScore","visitingManagerID
                         "visitingManagerName","homeManagerID","homeManagerName",
                         "visitor1ID","visitor1Name","visitor1Position",
                         "visitor2ID","visitor2Name","visitor2Position",
                         "visitor3ID","visitor3Name","visitor3Position",
                         "visitor4ID","visitor4Name","visitor4Position",
                         "visitor5ID","visitor5Name","visitor5Position",
                         "visitor6ID","visitor6Name","visitor6Position",
                         "visitor7ID","visitor7Name","visitor7Position",
                         "visitor8ID","visitor8Name","visitor8Position",
                         "visitor9ID","visitor9Name","visitor9Position",
                         "home1ID","home1Name","home1Position",
                         "home2ID","home2Name","home2Position",
                         "home3ID","home3Name","home3Position",
                         "home4ID","home4Name","home4Position",
                         "home5ID","home5Name","home5Position",
                          "home6ID","home6Name","home6Position",
                          "home7ID","home7Name","home7Position",
                          "home8ID","home8Name","home8Position",
                          "home9ID","home9Name","home9Position")
  return(outlogs)
}
# Prepares the data for merging based on the previous year
yearSplits <- function(year) {
  splits <- data.frame(bref_daily_batter(paste(year,"01","01",sep="-"),paste(year,"12","31",sep="-")))
  splits <- splits[-c(1:2,4,5)]
  splits$Name <- iconv(splits$Name,from="UTF-8",to="ASCII//TRANSLIT")
  splits$Name <- str_replace_all(splits$Name," Jr\\.","")
  return(splits)
}

# Merges Performance Splits overall all players in each lineup position
mergeAll <- function(logs,splits) {
  out <- mergePosition(logs,splits,"visitor",1)
  out <- rbind(out,mergePosition(logs,splits,"visitor",2))
  out <- rbind(out,mergePosition(logs,splits,"visitor",3))
  out <- rbind(out,mergePosition(logs,splits,"visitor",4))
  out <- rbind(out,mergePosition(logs,splits,"visitor",5))
  out <- rbind(out,mergePosition(logs,splits,"visitor",6))
  out <- rbind(out,mergePosition(logs,splits,"visitor",7))
  out <- rbind(out,mergePosition(logs,splits,"visitor",8))
  out <- rbind(out,mergePosition(logs,splits,"visitor",9))
  out <- rbind(out,mergePosition(logs,splits,"home",1))
  out <- rbind(out,mergePosition(logs,splits,"home",2))
  out <- rbind(out,mergePosition(logs,splits,"home",3))
  out <- rbind(out,mergePosition(logs,splits,"home",4))
  out <- rbind(out,mergePosition(logs,splits,"home",5))
  out <- rbind(out,mergePosition(logs,splits,"home",6))
  out <- rbind(out,mergePosition(logs,splits,"home",7))
  out <- rbind(out,mergePosition(logs,splits,"home",8))
  out <- rbind(out,mergePosition(logs,splits,"home",9))
}
# Handles
mergePosition <- function(logs,splits,team,num){
  logs[,paste0(team,num,"Name")] <- gsub("\\.","",logs[,paste0(team,num,"Name")])
  logs[,paste0(team,num,"Name")] <- gsub("i-M","i M",logs[,paste0(team,num,"Name")])
  logs[,paste0(team,num,"Name")] <- gsub("n-J","n J",logs[,paste0(team,num,"Name")])
  logs[,paste0(team,num,"Name")] <- gsub("Dee Gordon","Dee Strange-Gordon",logs[,paste0(team,num,"Name"
  logs[,paste0(team,num,"Name")] <- gsub("Giovanny Urshela","Gio Urshela",logs[,paste0(team,num,"Name")
  logs[,paste0(team,num,"Name")] <- gsub("Michael Taylor","Michael A. Taylor",logs[,paste0(team,num,"Nam
  logs[,paste0(team,num,"Name")] <- gsub("Vincent Velasquez","Vince Velasquez",logs[,paste0(team,num,"Na
  logs[,paste0(team,num,"Name")] <- gsub("Michael Brosseau","Mike Brosseau",logs[,paste0(team,num,"Name
  logs[,paste0(team,num,"Name")] <- gsub("Nate Lowe","Nathanial Lowe",logs[,paste0(team,num,"Name")])
  logs[,paste0(team,num,"Name")] <- gsub("Phillip Ervin","Phil Ervin",logs[,paste0(team,num,"Name")])
  logs[,paste0(team,num,"Name")] <- gsub("Josh Fuentes","Joshua Fuentes",logs[,paste0(team,num,"Name")]
  logs[,paste0(team,num,"Name")] <- gsub("Yulieski Gurriel","Yuli Gurriel",logs[,paste0(team,num,"Name"
  logs[,paste0(team,num,"Name")] <- gsub("Steve Wilkerson","Stevie Wilkerson",logs[,paste0(team,num,"Nam
  logs[,paste0(team,num,"Name")] <- gsub("Mike Soroka","Michael Soroka",logs[,paste0(team,num,"Name")])
  out <- merge(splits,logs,by.x="Name",by.y=paste0(team,num,"Name"))
  out <- mutate(out[,c(1:35,37,39)],homeAway=team,lineupPosition=as.numeric(num))
}
```

Import/Merge Data (included in EDA)

```
gl2012 <- cleanLogs(read.csv("gl2012.txt", header=FALSE))
gl2013 <- cleanLogs(read.csv("gl2013.txt", header=FALSE))
gl2014 <- cleanLogs(read.csv("gl2014.txt", header=FALSE))
gl2015 <- cleanLogs(read.csv("gl2015.txt", header=FALSE))
gl2016 <- cleanLogs(read.csv("gl2016.txt", header=FALSE))
gl2017 <- cleanLogs(read.csv("gl2017.txt", header=FALSE))
gl2018 <- cleanLogs(read.csv("gl2018.txt", header=FALSE))
gl2019 <- cleanLogs(read.csv("gl2019.txt", header=FALSE))
#split2012 <- yearSplits(2012)
#write.csv(split2012,"splits2012.csv")
split2012 <- read.csv("splits2012.csv")
#split2013 <- yearSplits(2013)
#write.csv(split2013,"splits2013.csv")
split2013 <- read.csv("splits2013.csv")
#split2014 <- yearSplits(2014)
#write.csv(split2014,"splits2014.csv")
split2014 <- read.csv("splits2014.csv")
#split2015 <- yearSplits(2015)
#write.csv(split2015,"splits2015.csv")
split2015 <- read.csv("splits2015.csv")
#split2016 <- yearSplits(2016)
#write.csv(split2016,"splits2016.csv")
split2016 <- read.csv("splits2016.csv")
#split2017 <- yearSplits(2017)
#write.csv(split2017,"splits2017.csv")
split2017 <- read.csv("splits2017.csv")
#split2018 <- yearSplits(2018)
#write.csv(split2018,"splits2018.csv")
split2018 <- read.csv("splits2018.csv")
#split2019 <- yearSplits(2019)
#write.csv(split2019,"splits2019.csv")
split2019 <- read.csv("splits2019.csv")

master <- mergeAll(gl2012,split2012)
master <- rbind(master,mergeAll(gl2013,split2013))
master <- rbind(master,mergeAll(gl2014,split2014))
master <- rbind(master,mergeAll(gl2015,split2015))
master <- rbind(master,mergeAll(gl2016,split2016))
master <- rbind(master,mergeAll(gl2017,split2017))
master <- rbind(master,mergeAll(gl2018,split2018))
master <- rbind(master,mergeAll(gl2019,split2019))

matched <- mergeLogsSplitsBref(gl2012,split2012)
matched <- rbind(matched,mergeLogsSplitsBref(gl2013,split2013))
matched <- rbind(matched,mergeLogsSplitsBref(gl2014,split2014))
matched <- rbind(matched,mergeLogsSplitsBref(gl2015,split2015))
matched <- rbind(matched,mergeLogsSplitsBref(gl2016,split2016))
matched <- rbind(matched,mergeLogsSplitsBref(gl2017,split2017))
matched <- rbind(matched,mergeLogsSplitsBref(gl2018,split2018))
matched <- rbind(matched,mergeLogsSplitsBref(gl2019,split2019))
# Prepared Dataset for Matched Paris Analysis was Never Used
```

This section is used to trim the non-qualified hitters from the dataset and was previously used to investigate transformation and standardization of data with year by year rate and summary statistics for the entire league. Unfortunately these did not improve the normality of the dataset so we decided not to use them.

```
#yearByYearAverages <- read.csv("yearByYearAverages.csv")
#yearByYearTotals <- read.csv("yearByYearTotals.csv")
dataset2 <- master[(master$PA/(162)) > 3,]
#dataset3 <- dataset2[dataset2$BA < median(dataset2$BA)+3*sd(ma),]
write.csv(dataset2,"compiledDataset.csv")
```

This chunk can be used to loaded the full dataset instead of compiling it separately from the component files.

```
dataset2 <- read.csv("compiledDataset.csv")
```

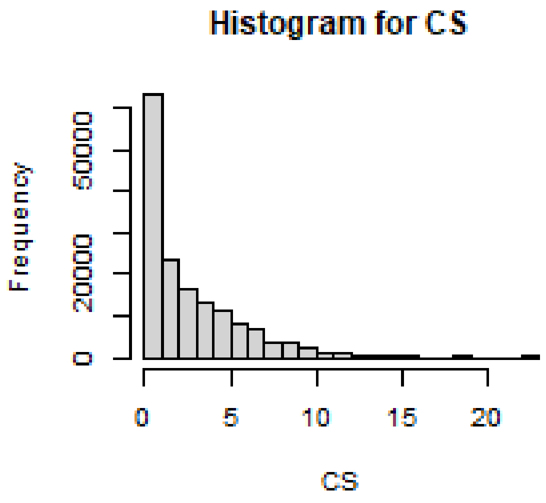
Correlation Data (included in EDA)

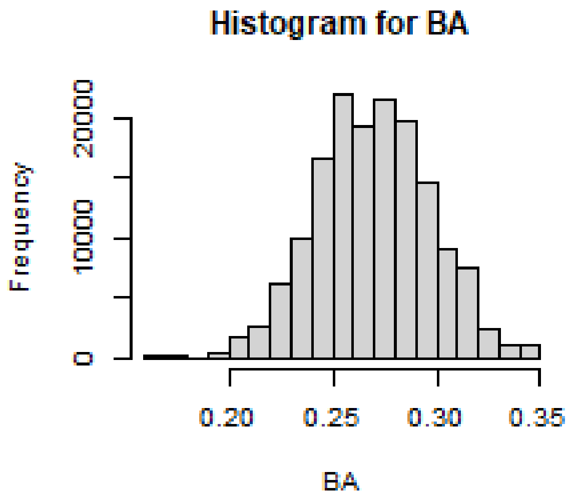
```
cor <- cor(dataset2[,c(24:27)],use="complete.obs")
cor
```

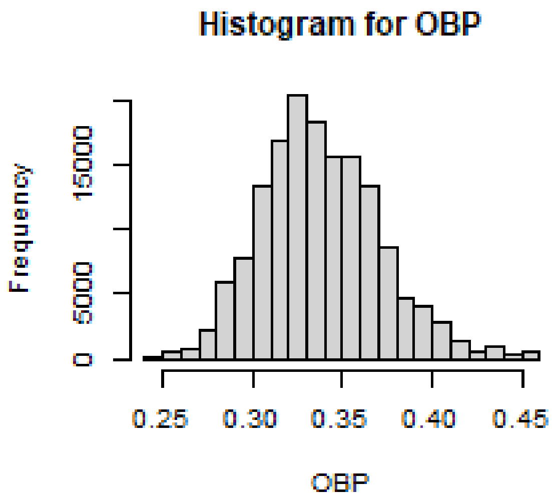
```
##                CS        BA          OBP        SLG
## CS   1.0000000000 0.1405801 0.0005778809 -0.1497093
## BA   0.1405801024 1.0000000 0.6963426679  0.5080134
## OBP  0.0005778809 0.6963427 1.0000000000  0.6382821
## SLG -0.1497092574 0.5080134 0.6382821268  1.0000000
```

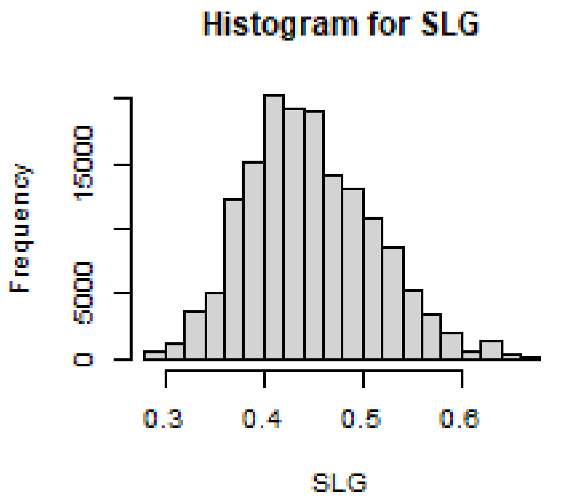
Investigate Normality of Explanatory Variables with Histograms and Q-Q Plots

```
par(mfrow=c(2,2))
imain=c(24:27)
for(i in imain){
  hist(dataset2[,i],main=paste0("Histogram for ",names(dataset2[i])),xlab=names(dataset2[i]))
}
```

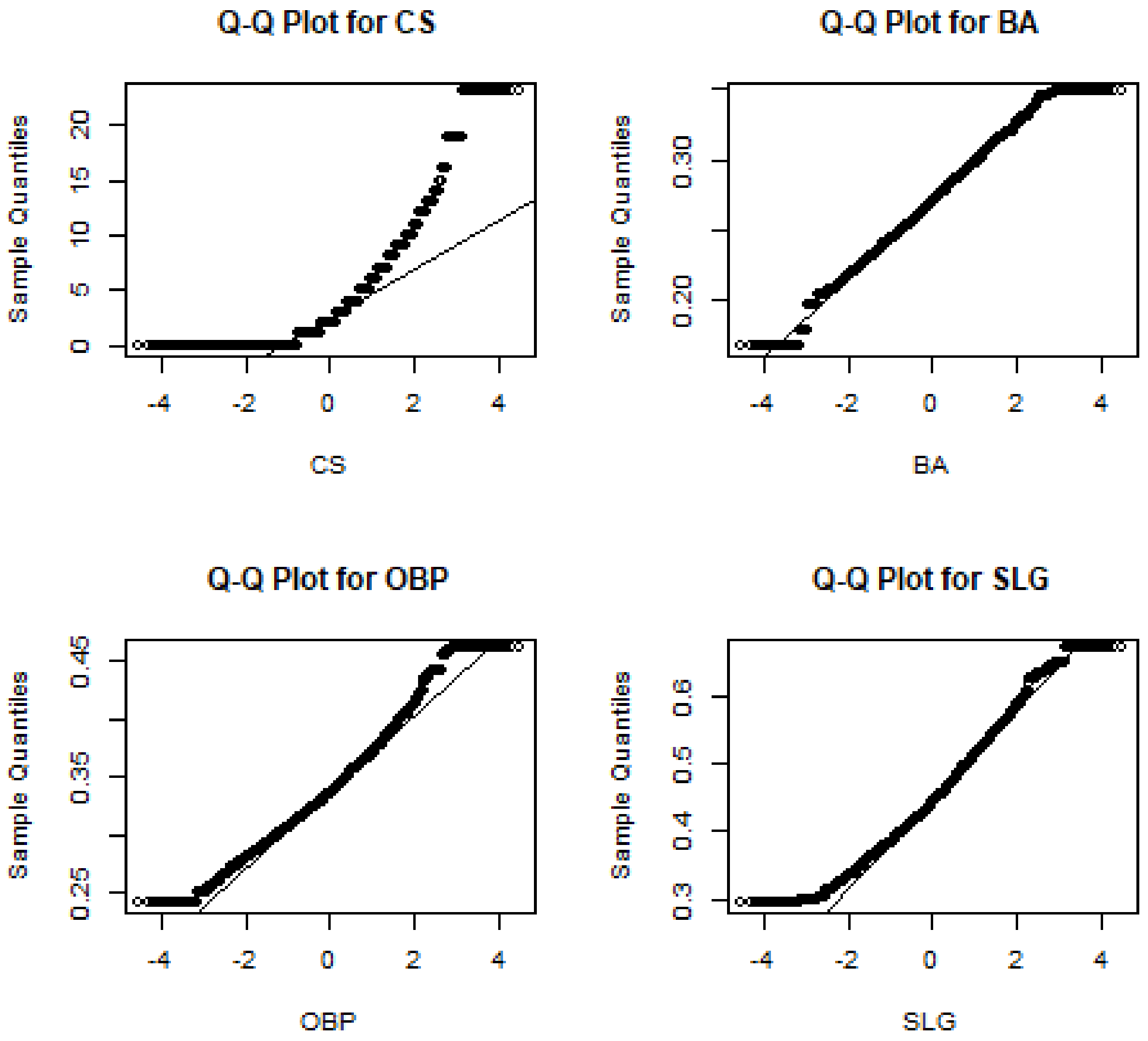








```
for(i in imain){
  qqnorm(dataset2[,i],main=paste0("Q-Q Plot for ",names(dataset2[i])),xlab=names(dataset2[i]))
  qqline(dataset2[,i])
}
```



```r
par(mfrow=c(1,1))
```

One Way ANOVA Between Each Explanatory Variable and Lineup Position

```r
par(mfrow=c(2,2))
for(i in imain) {
  print(paste0("ANOVA for ",names(dataset2[i])))
  print(summary(aov(dataset2[,i] ~ factor(lineupPosition),data=dataset2)))
}
```

```
## [1] "ANOVA for CS"
##                             Df  Sum Sq Mean Sq F value Pr(>F)
## factor(lineupPosition)       8  255415   31927    4224 <2e-16 ***
## Residuals               155271 1173630       8
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## [1] "ANOVA for BA"
##                             Df Sum Sq Mean Sq F value Pr(>F)    
## factor(lineupPosition)      8  12.54  1.5679    2346 <2e-16 ***
## Residuals              155271 103.79  0.0007                   
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
## [1] "ANOVA for OBP"
##                             Df Sum Sq Mean Sq F value Pr(>F)    
## factor(lineupPosition)      8  29.29   3.661    3987 <2e-16 ***
## Residuals              155271 142.59   0.001                   
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
## [1] "ANOVA for SLG"
##                             Df Sum Sq Mean Sq F value Pr(>F)    
## factor(lineupPosition)      8  105.7  13.209    3875 <2e-16 ***
## Residuals              155271  529.3   0.003                   
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
```

```
par(mfrow=c(1,1))
```

One Variable Linear Regression Between Each Explanatory Variable and Lineup Position

```
par(mfrow=c(2,2))
for(i in imain) {
  #plot(lm(lineupPosition ~ dataset2[,i],data=dataset2))
  print(paste0("Linear Regression for ",names(dataset2[i])))
  print(summary(lm(lineupPosition ~ dataset2[,i],data=dataset2)))
  print("  ")
}
```

```
## [1] "Linear Regression for CS"
## 
## Call:
## lm(formula = lineupPosition ~ dataset2[, i], data = dataset2)
## 
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -3.2258 -1.6560 -0.2258  1.2015  7.0533 
## 
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)    
## (Intercept)    4.225809   0.007259   582.2   <2e-16 ***
## dataset2[, i] -0.142442   0.001710   -83.3   <2e-16 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
## 
## Residual standard error: 2.044 on 155278 degrees of freedom
## Multiple R-squared:  0.04278,    Adjusted R-squared:  0.04277 
## F-statistic:  6939 on 1 and 155278 DF,  p-value: < 2.2e-16
## 
## [1] "  "
## [1] "Linear Regression for BA"
```

```
##
## Call:
## lm(formula = lineupPosition ~ dataset2[, i], data = dataset2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.9507 -1.5108 -0.1628  1.3420  6.8354
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)     9.48442    0.05055   187.6   <2e-16 ***
## dataset2[, i] -21.03401    0.18620  -113.0   <2e-16 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## Residual standard error: 2.008 on 155278 degrees of freedom
## Multiple R-squared:  0.07594,    Adjusted R-squared:  0.07593
## F-statistic: 1.276e+04 on 1 and 155278 DF,  p-value: < 2.2e-16
##
## [1] "  "
## [1] "Linear Regression for OBP"
##
## Call:
## lm(formula = lineupPosition ~ dataset2[, i], data = dataset2)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -4.698 -1.559 -0.004  1.352  6.729
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    10.51075    0.05146   204.2   <2e-16 ***
## dataset2[, i] -19.80662    0.15123  -131.0   <2e-16 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## Residual standard error: 1.983 on 155278 degrees of freedom
## Multiple R-squared:  0.09948,    Adjusted R-squared:  0.09948
## F-statistic: 1.715e+04 on 1 and 155278 DF,  p-value: < 2.2e-16
##
## [1] "  "
## [1] "Linear Regression for SLG"
##
## Call:
## lm(formula = lineupPosition ~ dataset2[, i], data = dataset2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.6496 -1.7251 -0.0734  1.3924  5.6986
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)    6.30545    0.03691  170.81   <2e-16 ***
## dataset2[, i] -5.59422    0.08169  -68.48   <2e-16 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## Residual standard error: 2.058 on 155278 degrees of freedom
## Multiple R-squared:  0.02932,    Adjusted R-squared:  0.02931
## F-statistic:  4690 on 1 and 155278 DF,  p-value: < 2.2e-16
##
## [1] "  "
```

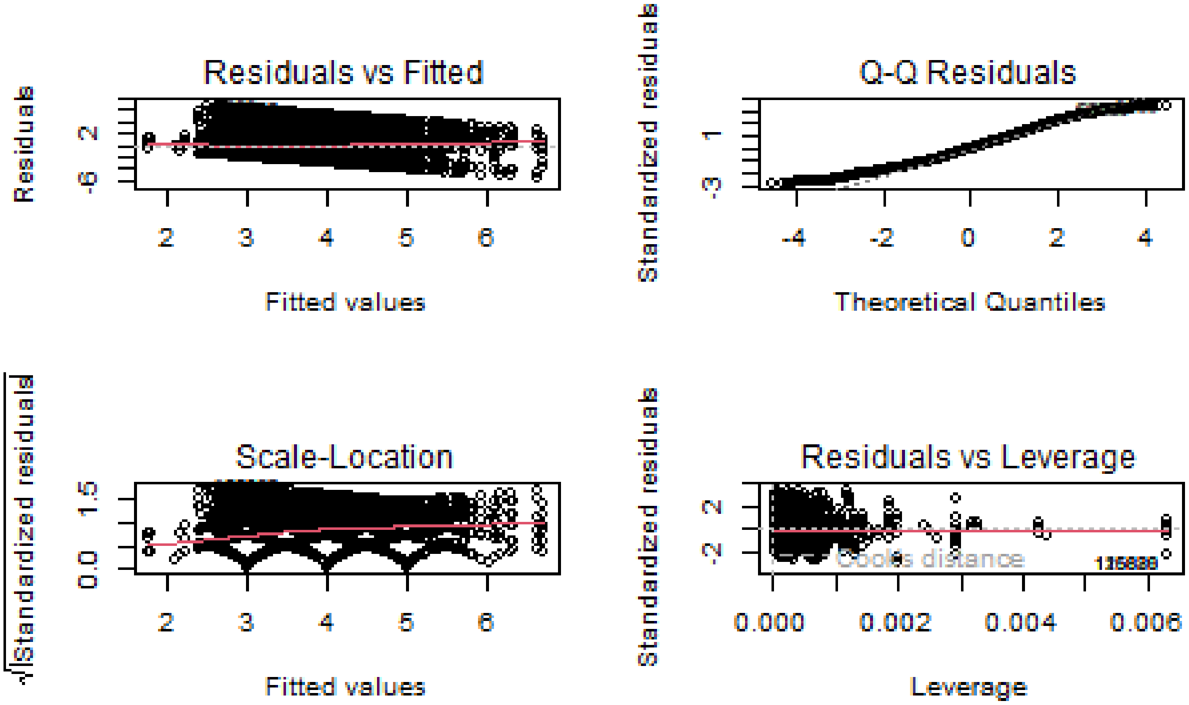
```
par(mfrow=c(1,1))
```

Multiple Regression with Interaction Effects (and added confounding effect of number of games that the player played in the previous season)

```
temp <- lm(lineupPosition ~ BA*OBP*SLG*OPS*G,data=dataset2)
summary(temp)
```

```
##
## Call:
## lm(formula = lineupPosition ~ BA * OBP * SLG * OPS * G, data = dataset2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.6299 -1.4657 -0.0519  1.2317  6.5030
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)         318.584    145.641   2.187  0.02871 *
## BA                 -257.549    524.365  -0.491  0.62331
## OBP                1701.817   1520.509   1.119  0.26304
## SLG                2050.876   1532.648   1.338  0.18086
## OPS               -3959.666   1475.870  -2.683  0.00730 **
## G                    -2.645      1.039  -2.547  0.01088 *
## BA:OBP           -11148.361   5611.808  -1.987  0.04697 *
## BA:SLG            -9108.770   5680.650  -1.603  0.10883
## OBP:SLG            -570.441   1811.612  -0.315  0.75285
## BA:OPS            14163.983   5449.796   2.599  0.00935 **
## OBP:OPS            4637.993    896.160   5.175 2.28e-07 ***
## SLG:OPS            2914.035    518.867   5.616 1.96e-08 ***
## BA:G                  3.622      3.755   0.965  0.33478
## OBP:G                -8.139     10.444  -0.779  0.43577
## SLG:G               -15.374     10.513  -1.462  0.14362
## OPS:G                28.539     10.119   2.820  0.00480 **
## BA:OBP:SLG         2271.455   6540.574   0.347  0.72838
## BA:OBP:OPS       -10143.107   3083.565  -3.289  0.00100 **
## BA:SLG:OPS        -8500.056   1788.444  -4.753 2.01e-06 ***
## OBP:SLG:OPS       -7317.221   1333.474  -5.487 4.09e-08 ***
## BA:OBP:G             65.839     38.504   1.710  0.08728 .
## BA:SLG:G             70.900     38.919   1.822  0.06850 .
## OBP:SLG:G            11.763     12.545   0.938  0.34842
## BA:OPS:G           -105.006     37.307  -2.815  0.00488 **
## OBP:OPS:G           -41.035      6.213  -6.605 3.99e-11 ***
## SLG:OPS:G           -21.299      3.613  -5.894 3.77e-09 ***
## BA:OBP:SLG:OPS    19569.643   4426.246   4.421 9.82e-06 ***
## BA:OBP:SLG:G         -45.205     45.265  -0.999  0.31796
## BA:OBP:OPS:G         103.210     21.378   4.828 1.38e-06 ***
## BA:SLG:OPS:G          62.154     12.485   4.978 6.42e-07 ***
## OBP:SLG:OPS:G         54.673      9.344   5.851 4.90e-09 ***
## BA:OBP:SLG:OPS:G    -148.831     31.161  -4.776 1.79e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.949 on 155248 degrees of freedom
## Multiple R-squared:  0.1303, Adjusted R-squared:  0.1301
## F-statistic:   750 on 31 and 155248 DF,  p-value: < 2.2e-16
```

```
par(mfrow=c(2,2))
plot(temp)
```



```
par(mfrow=c(1,1))
```

Comparing OBP of 1 and 2 vs the rest

```
obp12 <- dataset2[dataset2$lineupPosition < 3,]
obp12 <- obp12$OBP
summary(obp12)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.243   0.322   0.342   0.343   0.362   0.460
```

```r
obpNot12 <- dataset2[dataset2$lineupPosition > 2,]
obpNot12 <- obpNot12$OBP
summary(obp12)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.243   0.322   0.342   0.343   0.362   0.460
```

```r
t.test(obp12,obpNot12,alternative="greater", var.equal=TRUE)
```

```
##
##  Two Sample t-test
##
## data:  obp12 and obpNot12
## t = 34.142, df = 155278, p-value < 2.2e-16
## alternative hypothesis: true difference in means is greater than 0
## 95 percent confidence interval:
##  0.005921176         Inf
## sample estimates:
## mean of x mean of y
## 0.3429749 0.3367540
```

```r
#ggttest(t.test(obp12,obpNot12,alternative="greater", var.equal=TRUE))
```

Comparing OPS of 3 vs the rest

```r
ops3 <- dataset2[dataset2$lineupPosition == 3,]
ops3<- ops3$OPS
summary(ops3)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.5860  0.7770  0.8350  0.8405  0.8990  1.1090
```

```r
opsNot3 <- dataset2[dataset2$lineupPosition != 3,]
opsNot3 <- opsNot3$OPS
summary(opsNot3)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.5390  0.7160  0.7660  0.7734  0.8250  1.1090
```

```r
t.test(ops3,opsNot3,alternative="greater", var.equal=TRUE)
```

```
##
##  Two Sample t-test
##
## data:  ops3 and opsNot3
## t = 121.61, df = 155278, p-value < 2.2e-16
## alternative hypothesis: true difference in means is greater than 0
## 95 percent confidence interval:
##  0.06624516        Inf
## sample estimates:
## mean of x mean of y
## 0.8405294 0.7733759
```

Comparing OPS of 3 vs 1 & 2

```
summary(ops3)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##  0.5860  0.7770  0.8350  0.8405  0.8990  1.1090
```

```
ops12 <- dataset2[dataset2$lineupPosition < 3,]
ops12 <- ops12$OPS
summary(ops12)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##  0.5390  0.7210  0.7710  0.7811  0.8310  1.1020
```

```
t.test(ops3,ops12,alternative="greater", var.equal=TRUE)
```

```
## 
##  Two Sample t-test
## 
## data:  ops3 and ops12
## t = 90.492, df = 76984, p-value < 2.2e-16
## alternative hypothesis: true difference in means is greater than 0
## 95 percent confidence interval:
##  0.05836313        Inf
## sample estimates:
## mean of x mean of y 
## 0.8405294 0.7810857
```

Compare SLG of 4 & 5 vs Rest

```
slg4 <- dataset2[dataset2$lineupPosition == 4 | dataset2$lineupPosition == 5,]
slg4 <- slg4$SLG
summary(slg4)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##  0.2960  0.4210  0.4570  0.4605  0.4990  0.6490
```

```
slgNot4 <- dataset2[dataset2$lineupPosition != 4,]
slgNot4 <- slgNot4$SLG
summary(slgNot4)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##  0.2960  0.3990  0.4350  0.4429  0.4850  0.6720
```

```
t.test(slg4,slgNot4,alternative="greater",var.equal=TRUE)
```

```
## 
##  Two Sample t-test
## 
```

```
## data:  slg4 and slgNot4
## t = 51.459, df = 174893, p-value < 2.2e-16
## alternative hypothesis: true difference in means is greater than 0
## 95 percent confidence interval:
##  0.01705821        Inf
## sample estimates:
## mean of x mean of y
## 0.4605322 0.4429107
```

Comparing SLG of 4 & 5 to 1,2,3

```
slg123 <- dataset2[dataset2$lineupPosition <= 3,]
slg123 <- slg123$SLG
summary(slg123)
```

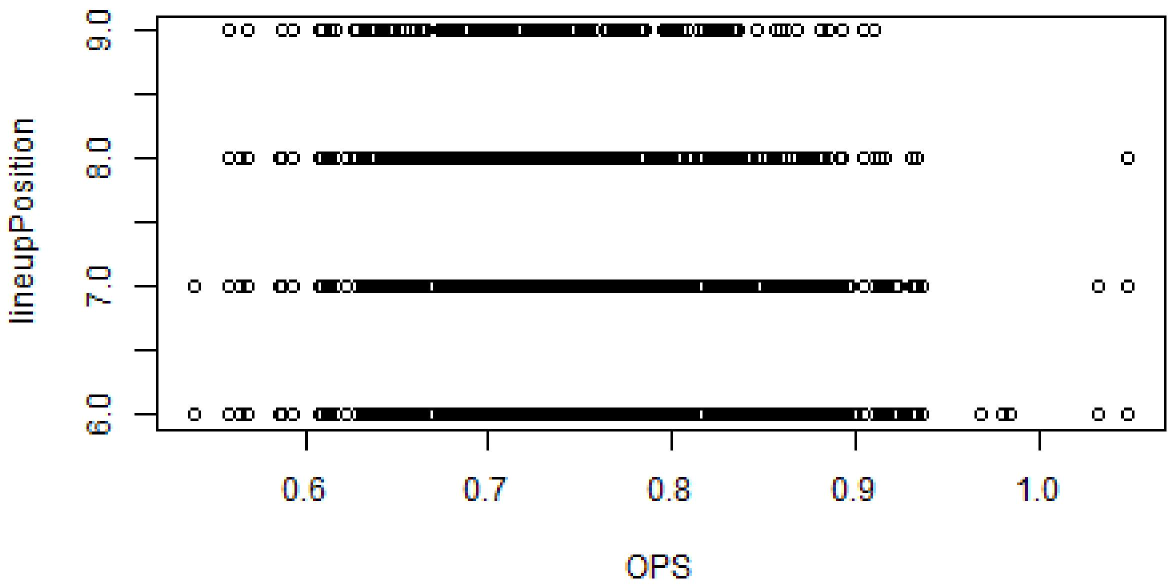
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.2960  0.4060  0.4490  0.4547  0.5000  0.6720
```

```
t.test(slg4,slg123,alternative="greater",var.equal=TRUE)
```

```
##
##  Two Sample t-test
##
## data:  slg4 and slg123
## t = 15.371, df = 121419, p-value < 2.2e-16
## alternative hypothesis: true difference in means is greater than 0
## 95 percent confidence interval:
##  0.005183299         Inf
## sample estimates:
## mean of x mean of y
## 0.4605322 0.4547278
```

Check whether the rest of the lineup is ordered from best hitter to worst hitter

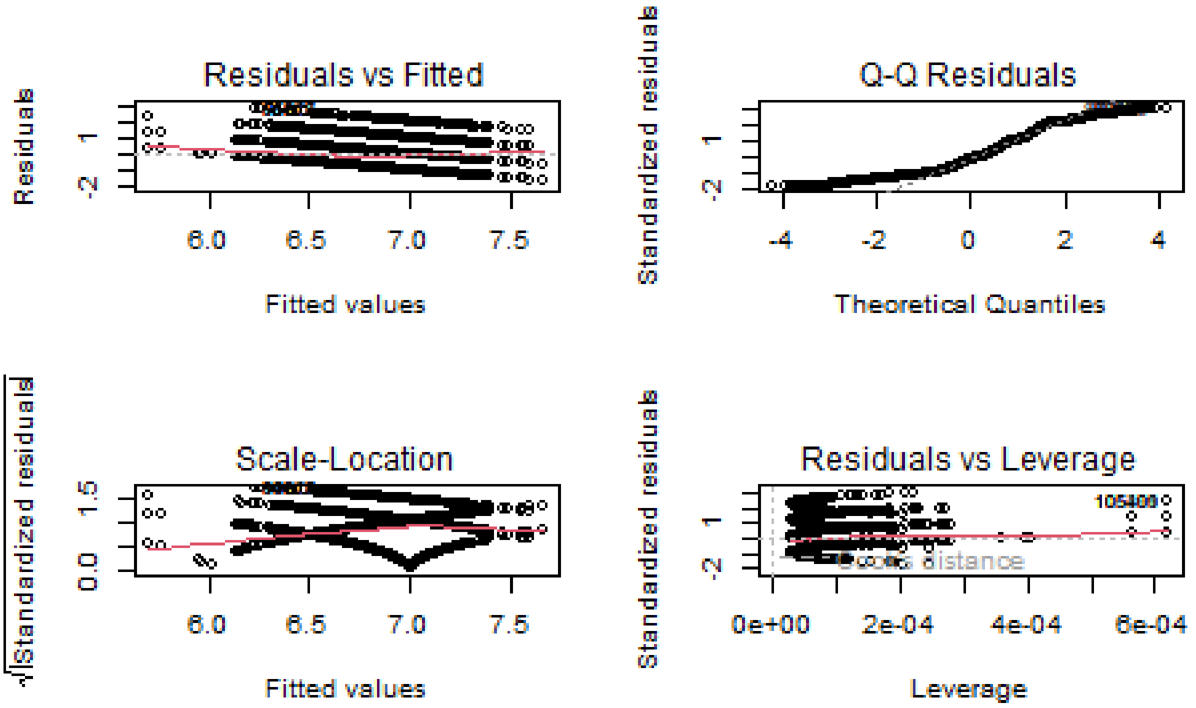
```
endOfLineup <- dataset2[dataset2$lineupPosition > 5,]
plot(lineupPosition ~ OPS,data=endOfLineup)
```



```
temp <- lm(lineupPosition ~ OPS,data=endOfLineup)
summary(temp)
```

```
##
## Call:
## lm(formula = lineupPosition ~ OPS, data = endOfLineup)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.66917 -0.80137 -0.09064  0.67795  2.76559
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  9.74804    0.05170  188.54   <2e-16 ***
## OPS         -3.85690    0.07036  -54.82   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9184 on 33857 degrees of freedom
## Multiple R-squared:  0.08153,    Adjusted R-squared:  0.0815
## F-statistic:  3005 on 1 and 33857 DF,  p-value: < 2.2e-16
```

```
par(mfrow=c(2,2))
plot(temp)
```



```r
par(mfrow=c(1,1))
```

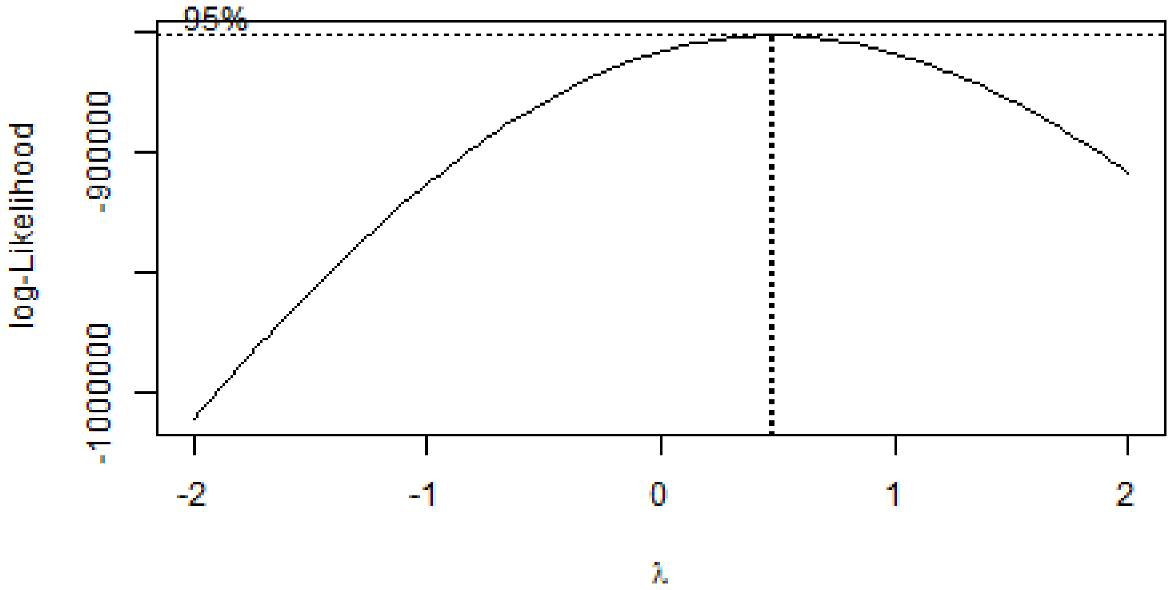
Attempt at Box-Cox Transformation (not used)

```r
library(MASS)
```

```
##
## Attaching package: 'MASS'

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

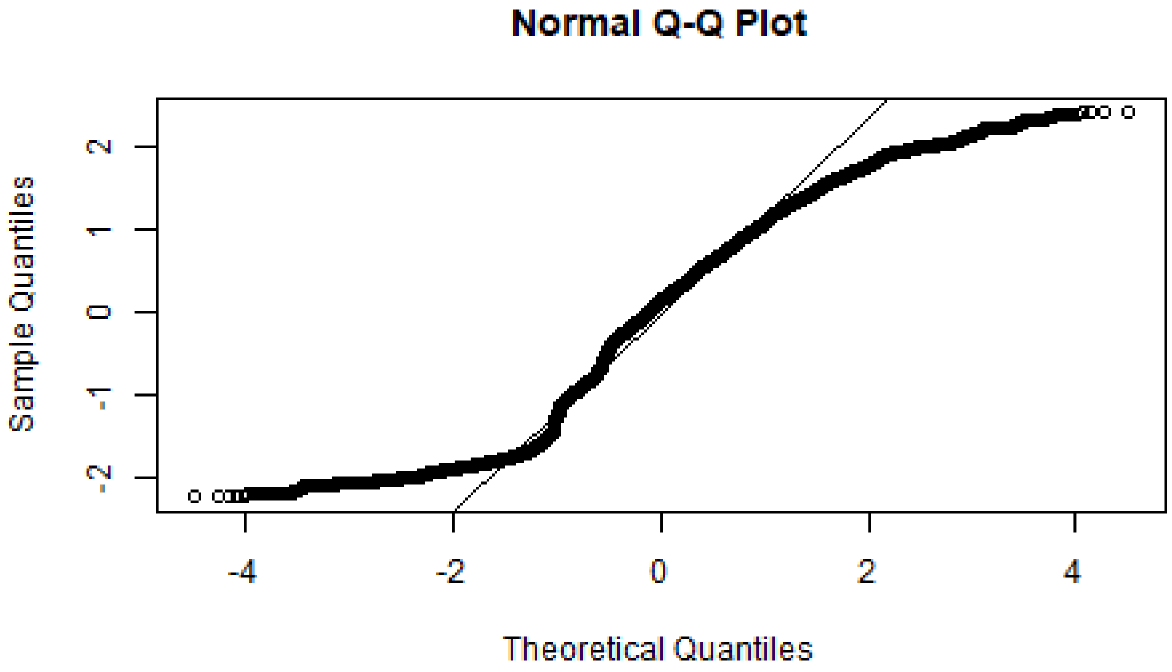
```r
#qqPlot(sqrt(dataset2$OPSplus_alt))
#summary(dataset2$OPSplus_alt)
bc <- boxcox(lineupPosition ~ OPS, data=dataset2)
```



```
lambda <- bc$x[which.max(bc$y)]
print(lambda)
```

```
## [1] 0.4646465
```

```
temp2 <- lm(((lineupPosition^lambda-1)/lambda) ~ OPS,dataset2)
qqnorm(temp2$residuals)
qqline(temp2$residuals)
```



```r
summary(temp2)
```

```
##
## Call:
## lm(formula = ((lineupPosition^lambda - 1)/lambda) ~ OPS, data = dataset2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.2324 -0.8183  0.1357  0.7828  2.4109
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.42285    0.02352  145.56   <2e-16 ***
## OPS         -2.20856    0.02973  -74.29   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.042 on 155278 degrees of freedom
## Multiple R-squared:  0.03432,    Adjusted R-squared:  0.03432
## F-statistic:  5519 on 1 and 155278 DF,  p-value: < 2.2e-16
```