Project 1

Dustin Oakes

11/18/2020

Reverse Engineering the NFL Passer Rating Formula

Abstract

The highest level of the sport of American Football is played in the National Football League (NFL). On each team, the player in the quarterback (QB) position is responsible for passing the ball down the field and handing the ball off to runners. Since quarterbacks touch the ball on virtually every offensive play, this position tends to command the highest salary of any player on a team. With front offices currently spending upwards of $45 million per year to sign the best quarterbacks to contracts, teams have a significant vested interest in ranking and rating players, so as to make sure the best ones are palying on their team. The NFL maintains an official statistic, Passer Rating, which measures the performance of quarterbacks using a combination of the statistics normally measured throughout a game. In this project, we will attempt to reverse engineer the NFL's formula for this statistic through a linear regression of the variables which are normally collected by the scorers throughout the course of a game.

Loading Data

```
pass2009 <- read.csv("pass-2009.csv")
pass2010 <- read.csv("pass-2010.csv")
pass2011 <- read.csv("pass-2011.csv")
pass2012 <- read.csv("pass-2012.csv")
pass2013 <- read.csv("pass-2013.csv")
pass2014 <- read.csv("pass-2014.csv")
pass2015 <- read.csv("pass-2015.csv")
pass2016 <- read.csv("pass-2016.csv")
pass2017 <- read.csv("pass-2017.csv")
pass2018_bad <- read.csv("pass-2018.csv")
```

The 2018 passing data did not have an index column, so to successfully combine the data we will add one:

```
Rk <- c(1:106)
pass2018 <- cbind(Rk,pass2018_bad)
```

Now we will concatenate all the data into one set, which will give us 10 years worth of Passer Ratings as well as all the associated statistics which were presumably used to create the ratings:

```
passdata <- rbind(pass2009,pass2010,pass2011,pass2012,
                  pass2013,pass2014,pass2015,pass2016,
                  pass2017,pass2018)
```

Many of the entries in our newly minted data set unfortunately will not be very useful in our analysis. Many teams run so-called "trick plays" in which another player than the QB will receive the ball and pass it, in an attempt to catch the defense off guard. However, this results in many non-QB players being included in the data, which results in many NAs throughout the data set.

Removing the NAs:

```
napassdata <- na.omit((passdata))

attach(napassdata)
```

```
## The following object is masked _by_ .GlobalEnv:
##
##     Rk
```

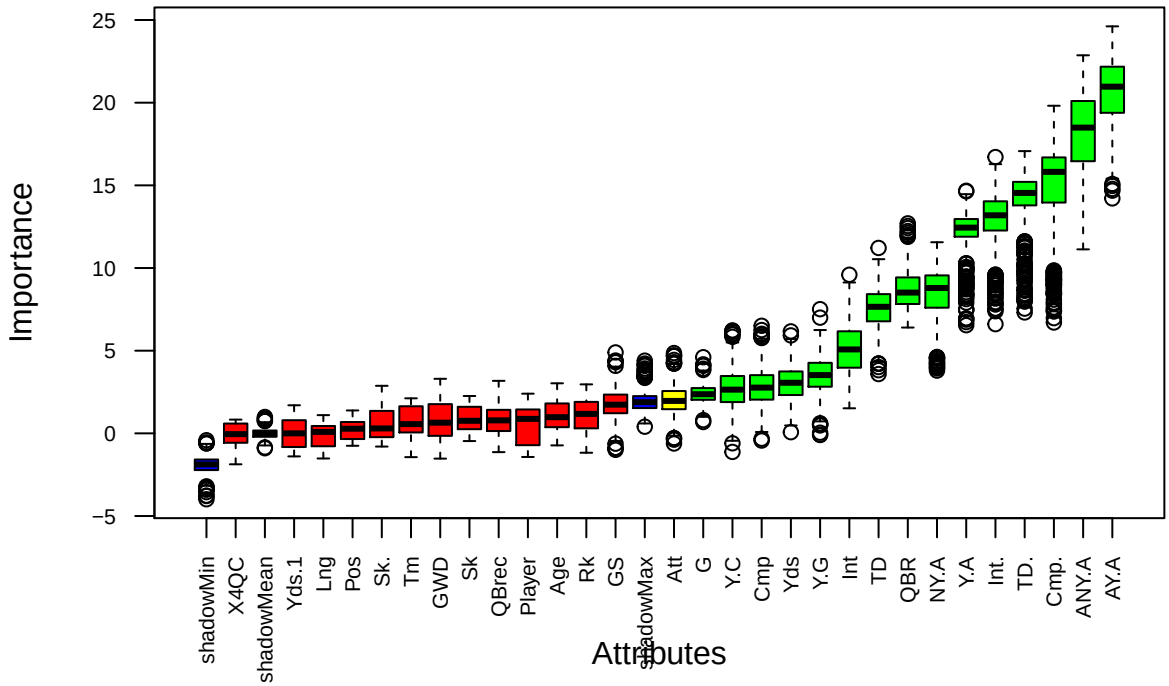
Boruta Variable Selection

Running the Boruta Algorithm to determine important and unimportant vairables to include in our regression:

```
boruta <- Boruta::Boruta(Rate ~ ., data=napassdata, doTrace=0, maxRuns=1000)
print(boruta)
```

```
## Boruta performed 999 iterations in 2.40116 mins.
##  15 attributes confirmed important: ANY.A, AY.A, Cmp, Cmp., G and 10
## more;
##  13 attributes confirmed unimportant: Age, GS, GWD, Lng, Player and 8
## more;
##  1 tentative attributes left: Att;
```

```
plot(boruta, las=2, cex.axis = 0.7)
```



Based on using the Boruta Algorithm, the top 10 predictors include:

AY.A (Adjusted Yards/Pass Attempts)
ANY.A (Adjusted Net Yards/Pass Attempts)
Cmp. (Completion Percentage)
TD. (Touchdown Percentage)
Int. (Interception Percentage)
Y.A (Yards/Pass Attempts)
NY.A (Net Yards/Pass Attempts)
Yds (Yards)
TD (Touchdowns)
Int (Interceptions)
Y.G (Yards/Games)

*Note: QBR is another ranking/rating statistic developed by sports journalists, we will not be using it to predict rating, as it is not a statistic that the official scorers tabulate.

Mallows Cp Variable Selection

Defining Mallows Cp and Step functions:

```
mallows_cp = function(model1, model2){
  n = nrow(model1$model)
  p1 = length(coef(model1))
  p2 = length(coef(model2))
  if(p2<p1)
```

```
    stop('You have interchanged the full model and the subset model', call. = FALSE)
  sum(resid(model1)**2) / sum(resid(model2)^2) *(n-p2) + 2 * p1 -n
}

mystep = function(object){
  reduced_object = object
  old_mcp = mallows_cp(object, object)
  while(TRUE){
    nms = attr(terms(reduced_object),"term.labels")
    u = lapply(nms, function(x) update(reduced_object, paste0(".~ .-", x)))
    mcp = sapply(u, mallows_cp, object) # same as sapply(u, function(x) mallows_cp(x, object))
    if(min(mcp) > old_mcp) break
    old_mcp = min(mcp)
    reduced_object = u[[which.min(mcp)]]
  }
  reduced_object
}
```

Using the defined functions to do a backwards stepwise regression, returning the result with the lowest Cp value:

```
model_all <- lm (Rate ~ . - Rk - Player - Tm - Age
                 - Pos - G - GS - QBrec - QBR, data=napassdata)
mallows_all <- mystep(model_all)
summary(mallows_all)
```

```
##
## Call:
## lm(formula = Rate ~ Att + Cmp. + Yds + TD + TD. + Int + Int. +
##     Lng + AY.A + Y.C + Y.G + NY.A + ANY.A + Sk. + X4QC, data = napassdata)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.4194 -0.6688  0.0274  0.6679  7.2318
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.064e+01  5.141e+00   5.960 6.09e-09 ***
## Att          8.461e-03  4.071e-03   2.078 0.038393 *
## Cmp.         3.012e-01  9.004e-02   3.345 0.000912 ***
## Yds         -2.975e-03  6.063e-04  -4.907 1.41e-06 ***
## TD           4.347e-01  2.535e-02  17.149  < 2e-16 ***
## TD.          7.035e+00  6.041e-01  11.646  < 2e-16 ***
## Int         -1.704e-01  3.645e-02  -4.675 4.19e-06 ***
## Int.        -1.637e+01  1.294e+00 -12.654  < 2e-16 ***
## Lng          1.298e-02  6.981e-03   1.860 0.063756 .
## AY.A         9.480e+00  1.212e+00   7.820 6.12e-14 ***
## Y.C         -2.826e+00  4.472e-01  -6.320 7.87e-10 ***
## Y.G         -6.225e-03  2.837e-03  -2.194 0.028877 *
## NY.A         4.125e+01  3.129e+00  13.181  < 2e-16 ***
## ANY.A       -4.061e+01  2.829e+00 -14.353  < 2e-16 ***
## Sk.          2.478e-01  1.534e-01   1.615 0.107148
## X4QC         1.012e-01  6.925e-02   1.461 0.144791
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.584 on 354 degrees of freedom
## Multiple R-squared:  0.9878, Adjusted R-squared:  0.9872
## F-statistic:  1903 on 15 and 354 DF,  p-value: < 2.2e-16
```

Based on using Mallows Cp, the top 10 predictors include:

Cmp. (Completion Percentage)
Yds (Yards)
TD (Touchdowns)
TD. (Touchdown Percentage)
Int (Interceptions)
Int. (Interception Percentage)
AY.A (Adjusted Yards/Pass Attempts)
Y.C (Yards/Completion)
NY.A (Net Yards/Pass Attempts)
ANY.A (Adjusted Net Yards/Pass Attempts)

BOLD = Also Important in Boruta Model

Final Variable Selection

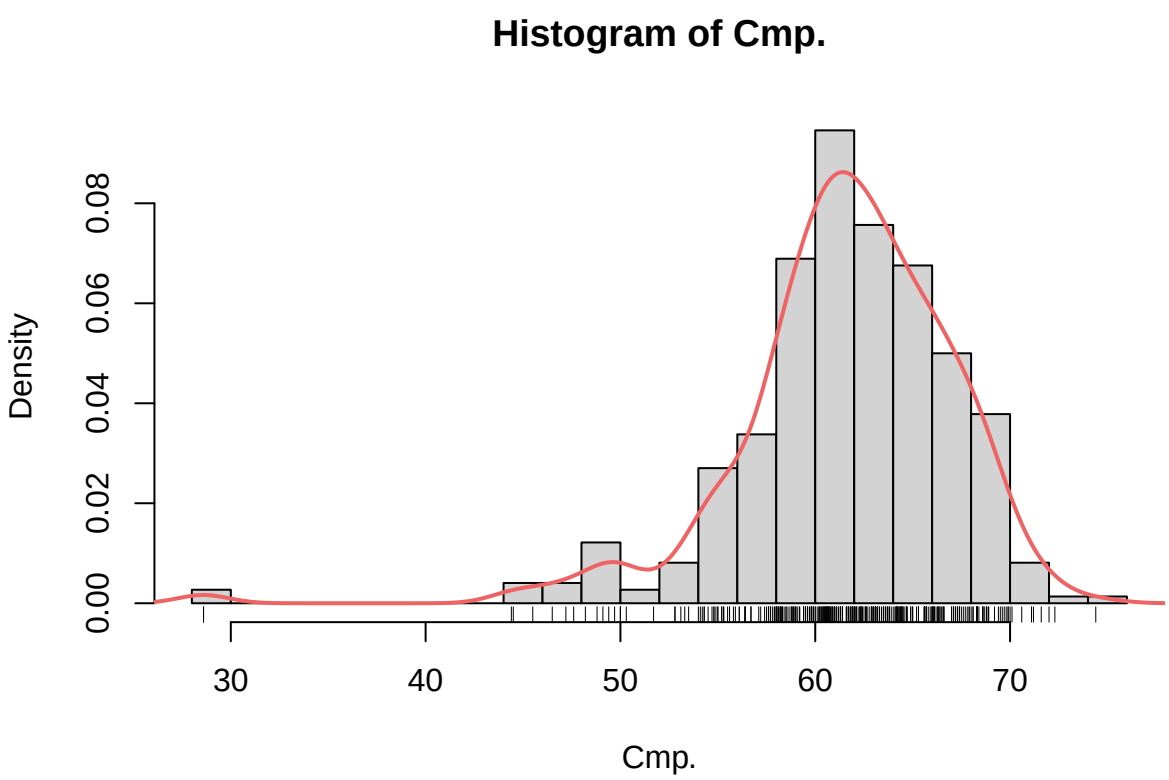
Based on the variable selection criteria of Mallows Cp and the Boruta Algorithm, a few variables stand out as being particularly important to our regression. Clearly, yardage, touchdowns, completions, and interceptions are the fundamental stats that contribute to Passer Rating. Let us simply use the advice of our algorithms and select the variables that both methods consider important to the regression. From here on, we will be considering the following:

Cmp. (Completion Percentage)
TD (Touchdowns)
TD. (Touchdown Percentage)
Int (Interceptions)
Int. (Interception Percentage)
AY.A (Adjusted Yards/Pass Attempts)
NY.A (Net Yards/Pass Attempts)
ANY.A (Adjusted Net Yards/Pass Attempts)

Descriptive Analysis

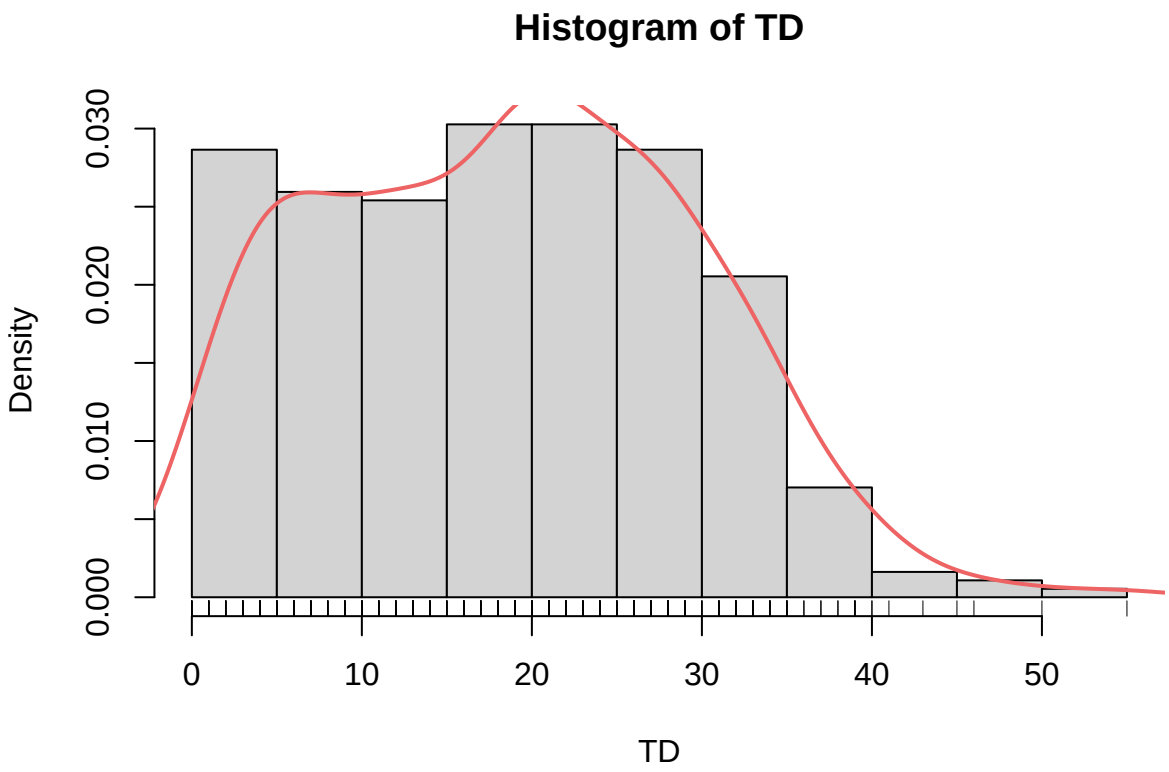
Histograms and Density Plots

```
hist(Cmp., breaks = "FD", freq = FALSE)
lines(density(Cmp.),lwd = 2, col ="indianred2")
rug(Cmp.)
```



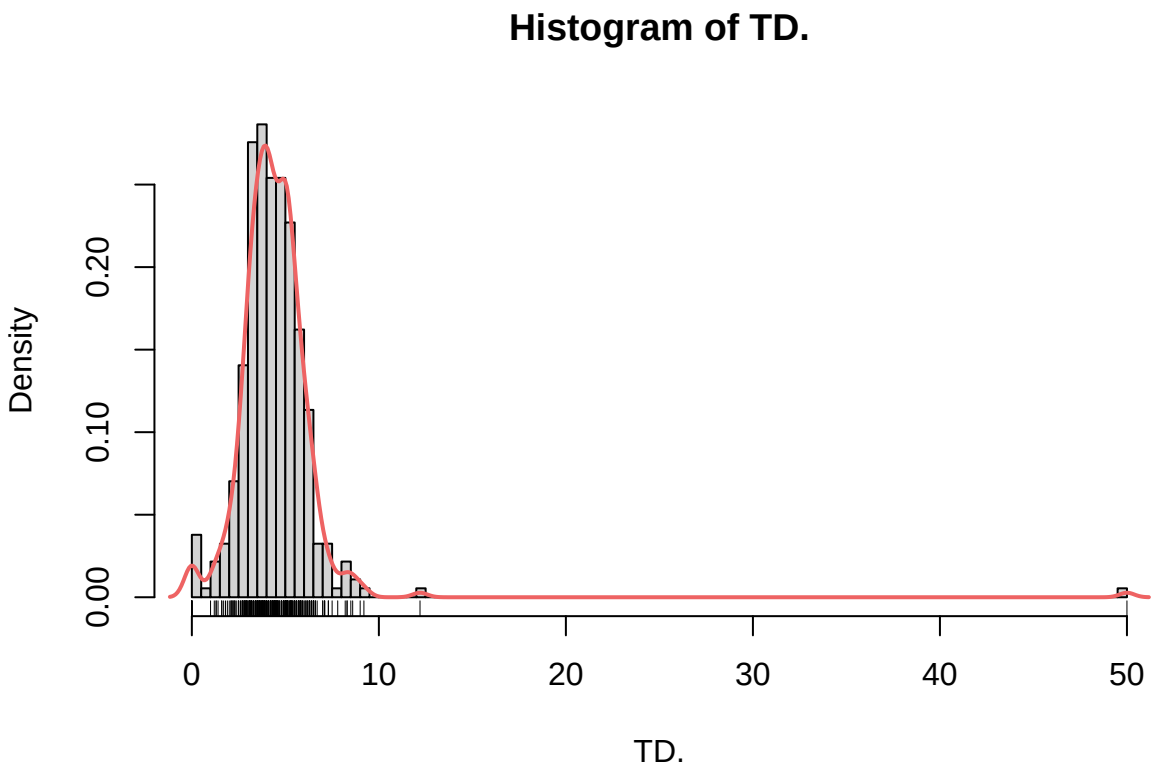
Between 2009 and 2018, NFL quarterbacks averaged around 60%-70% in Completion Percentage, which is the ratio of pass completions to pass attempts. The data is well distributed, however there is an outlier that could be attended to. In addition, the data seem more skewed to the right side, with fewer points in the 50%-60% range then in the 60%-70% range.

```
hist(TD, breaks = "FD", freq = FALSE)
lines(density(TD),lwd = 2, col ="indianred2")
rug(TD)
```



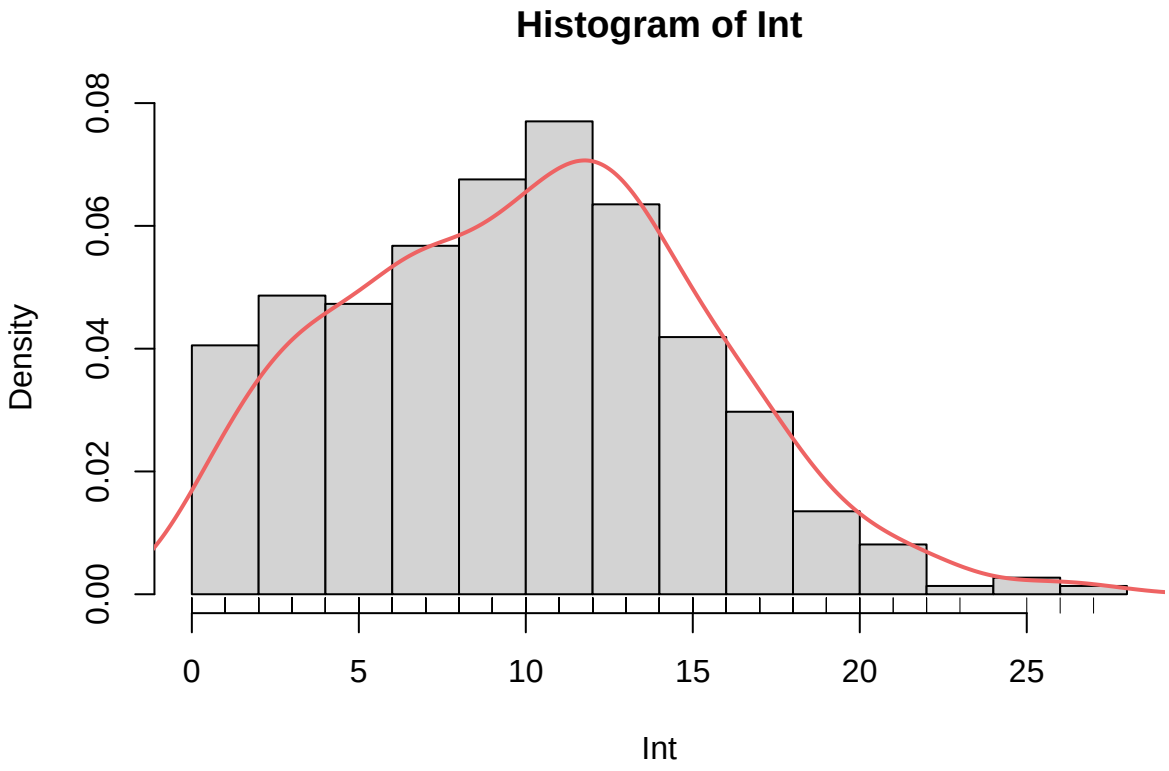
This data is very positively skewed, as naturally it is easier to score fewer touchdowns than to score large amounts. Touchdowns score 6 points for a team; they are the main way that points are accumulated and games are won. Quarterbacks might average around 20-25 touchdowns in a 16 game season. Scoring 40-50+ TDs in a season would be regarded as historic performances by the best players of all time.

```
hist(TD., breaks = "FD", freq = FALSE)
lines(density(TD.),lwd = 2, col ="indianred2")
rug(TD.)
```



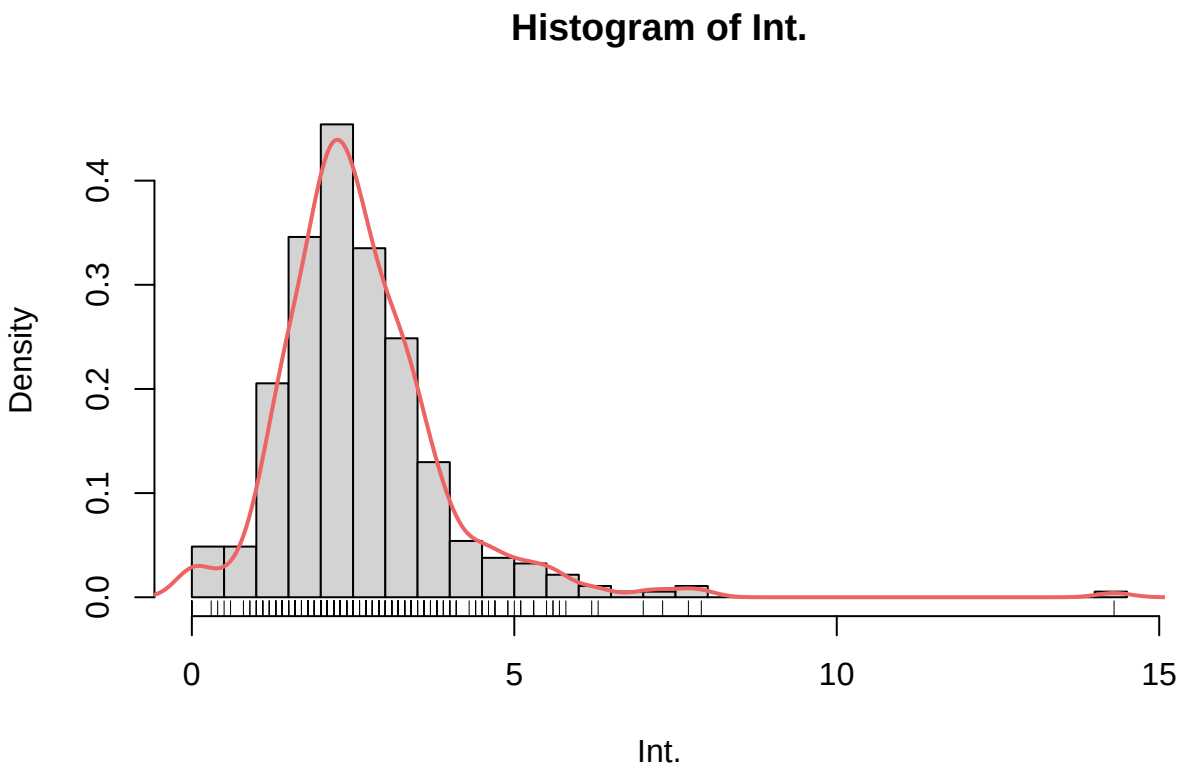
With the removal of some outliers, this data looks very normally distributed. NFL QBs generally score a touchdown on 2%-10% of every pass they attempt.

```
hist(Int, breaks = "FD", freq = FALSE)
lines(density(Int),lwd = 2, col ="indianred2")
rug(Int)
```



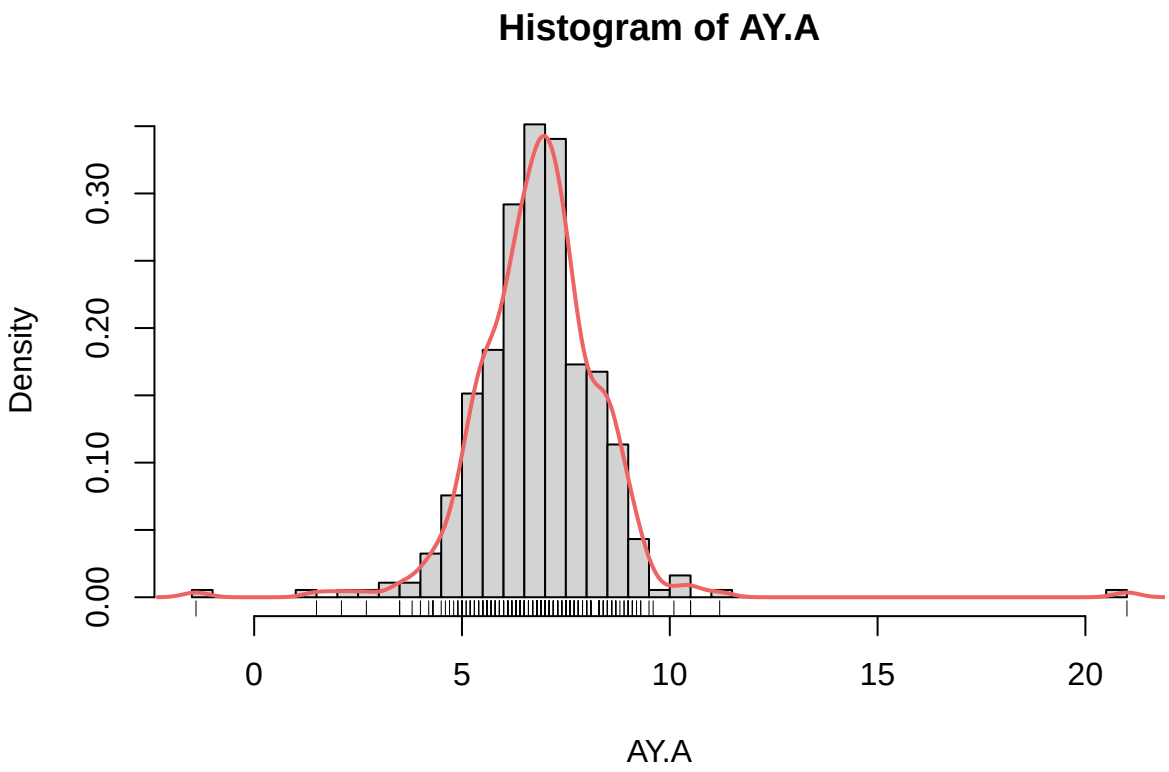
Interceptions occur when the quarterback throws a pass and it is caught by a defensive player. These negatively impact a quarterback's performance, as interceptions end the team's offensive possession and any chance of scoring points. The best quarterbacks suffer very few (around 0-7) interceptions, as they are very careful with where they throw the ball, while worse players can turn the ball over 15-20+ times per season.

```
hist(Int., breaks = "FD", freq = FALSE)
lines(density(Int.),lwd = 2, col ="indianred2")
rug(Int.)
```



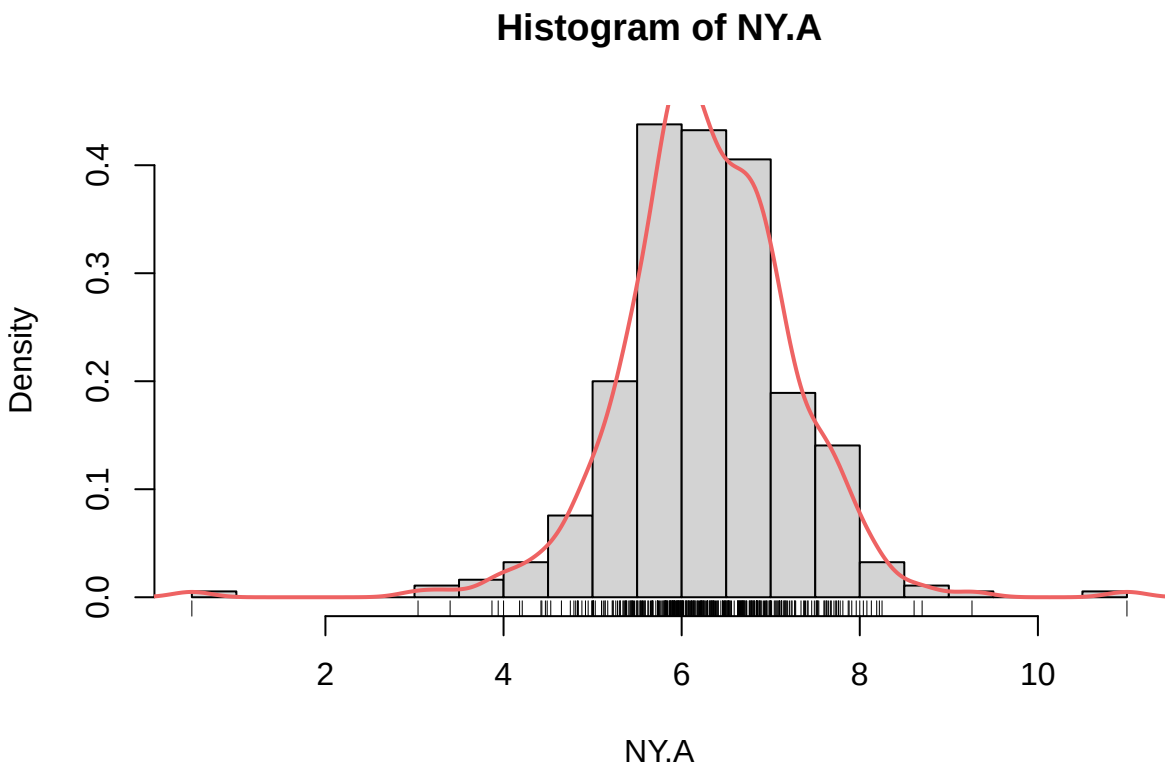
Interception Percentage measures the ratio of every pass attempt that results in an interception. The data between 0%-5% is very nicely distributed, however some outliers skew the data a little bit.

```
hist(AY.A, breaks = "FD", freq = FALSE)
lines(density(AY.A),lwd = 2, col ="indianred2")
rug(AY.A)
```



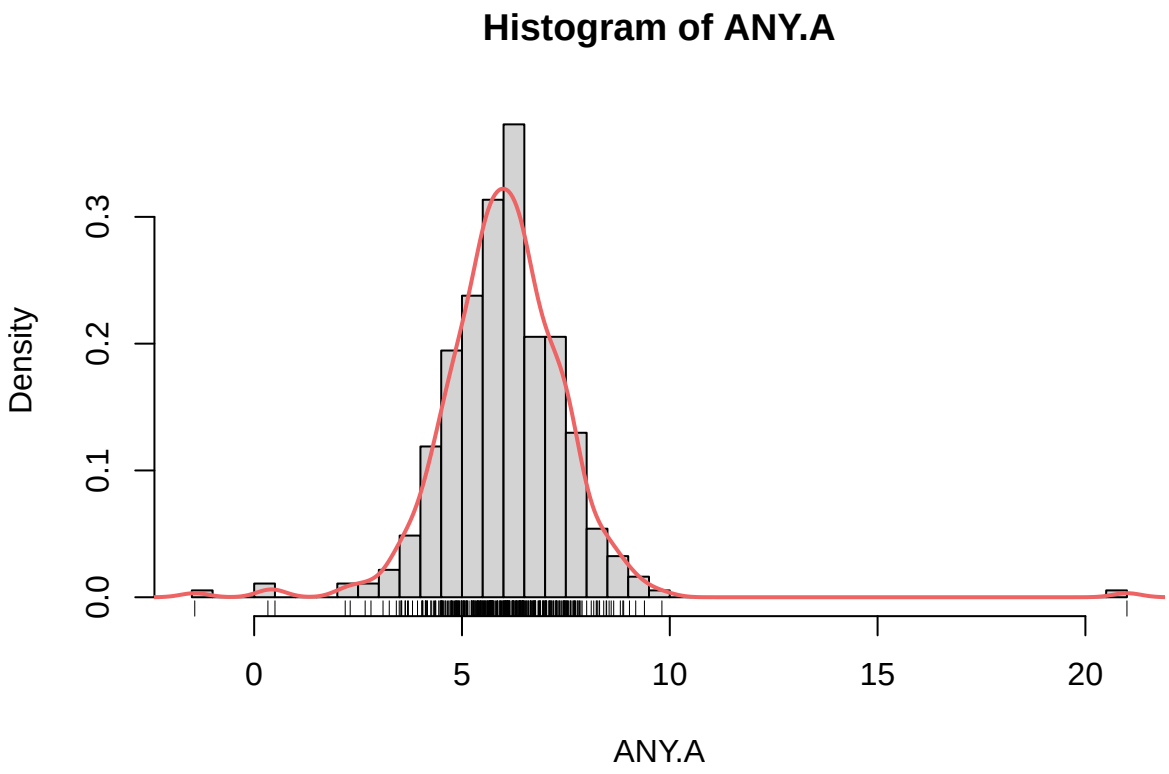
Adjusted Yards per Attempt is a statistic that measures how many yards are gained per pass attempt, and then 'adjusting' with a addition for touchdowns and a subtraction for interceptions. This gives a statistic which reveals more about a player's performance than simply measuring yards per attempt.

```
hist(NY.A, breaks = "FD", freq = FALSE)
lines(density(NY.A),lwd = 2, col ="indianred2")
rug(NY.A)
```



Net Yards per Attempt is a similar statistic to Adjusted Yards per Attempt, but rather than adding TDs and INTs to the statistic, NY.A penalizes a quarterback for taking a sack, which occurs when the quarterback is tackled behind the line of scrimmage for a loss of yards.

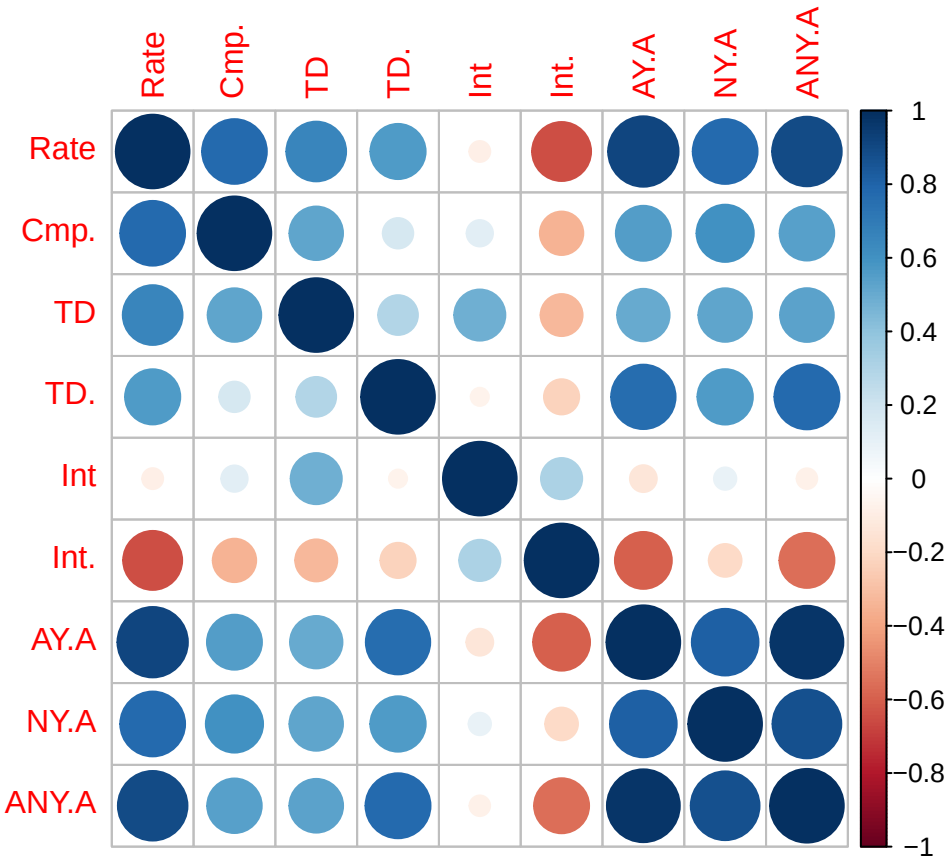
```
hist(ANY.A, breaks = "FD", freq = FALSE)
lines(density(ANY.A),lwd = 2, col ="indianred2")
rug(ANY.A)
```



As the name suggests, Adjusted Net Yards per Attempt is a combination of AY.A and NY.A, so it measures yards per attempt, accounting for touchdowns, interceptions, and sacks. Collinearity could become an issue of we try to include all these in our model, so we will keep an eye on that when building our model. Luckily, ANY.A seems to have the best distribution of all three variables, and it includes all the parameters we want for our regression.

Correlation Plot

```
vars_select <- c("Rate","Cmp.","TD","TD.","Int",
            "Int.","AY.A","NY.A","ANY.A")
vars_select_data <- napassdata[vars_select]
corrplot::corrplot(cor(vars_select_data))
```



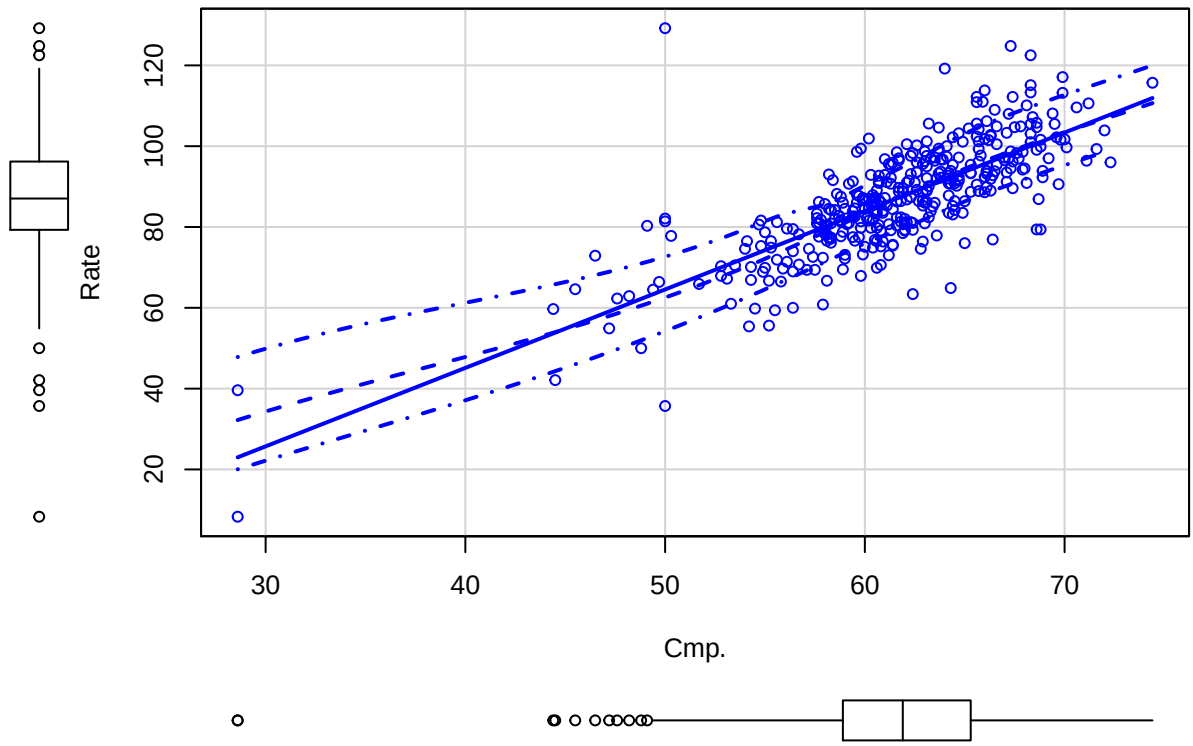
As mentioned above, there is a pretty strong correlation between all the yards per attempt statistics, so we will likely remove all but one when building our model. Otherwise, there are no dark red or dark blue circles (strong correlations) between any of the other variables, so they should all be suitable for inclusion in our model.

Identifying Non-Linearity

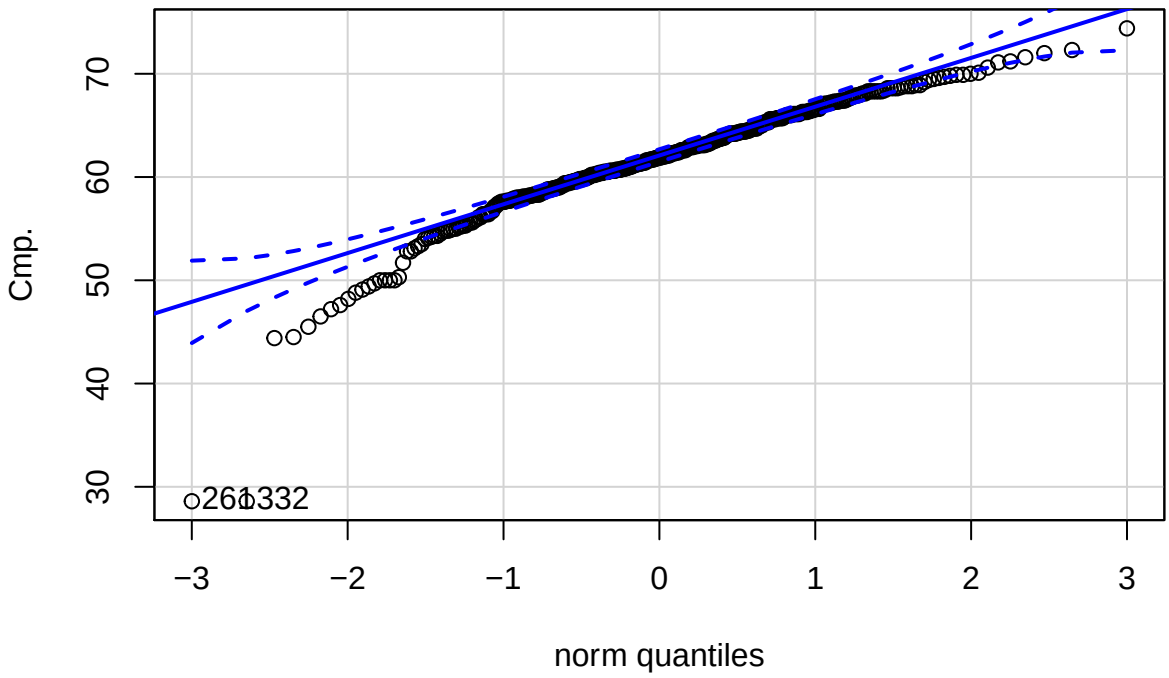
```
library(car)
```

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

```
scatterplot(Cmp.,Rate)
```



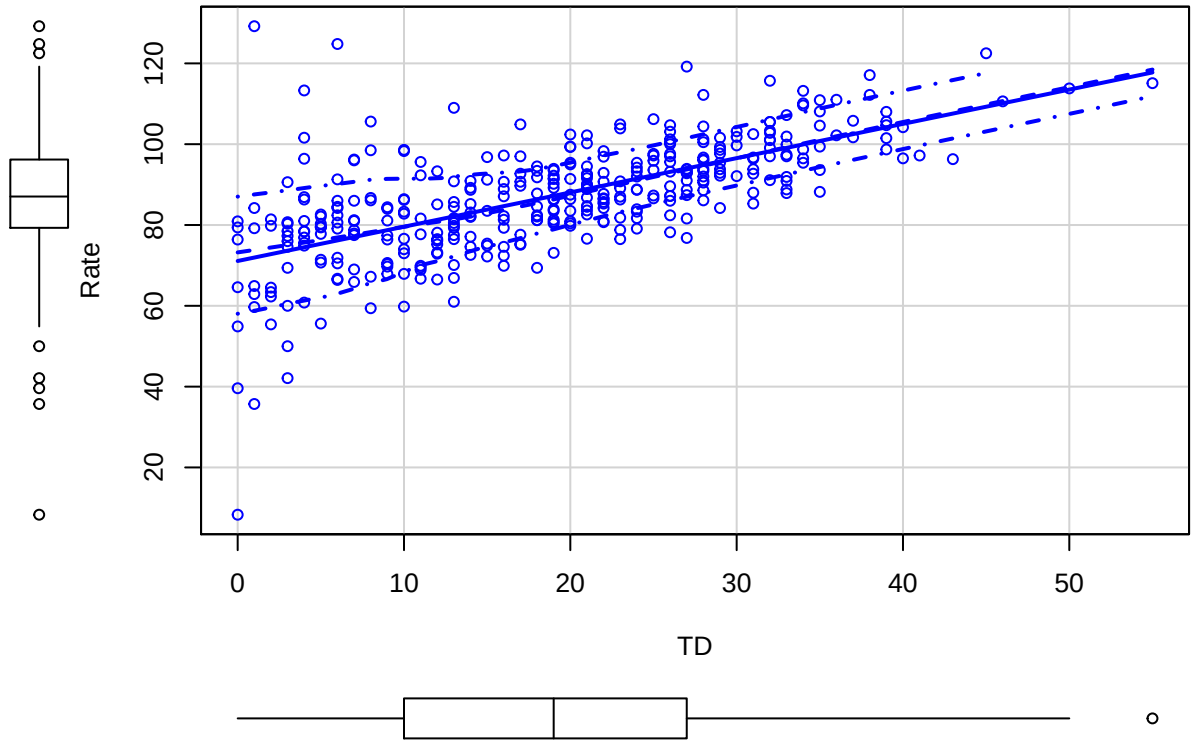

```
qqPlot(Cmp.)
```



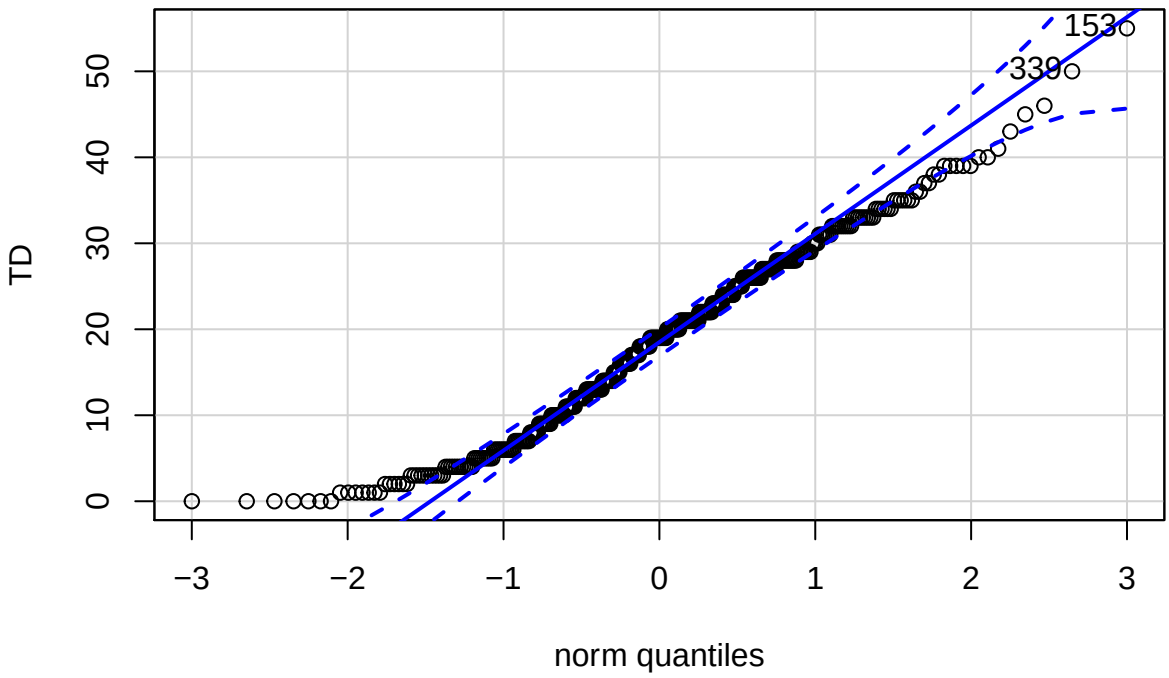

```
## [1] 261 332
```

Comparing Passer Rating and Completion Percentage we see a clear positive relationship between the variables. Those players with the lowest completion percentages likely only have such a low percentage because of a limited sample size of pass attempts; we could remove them from out data set so the data better follows a normal distribution. In addition, transforming the data might give us a more normal distribution.

```
scatterplot(TD,Rate)
```



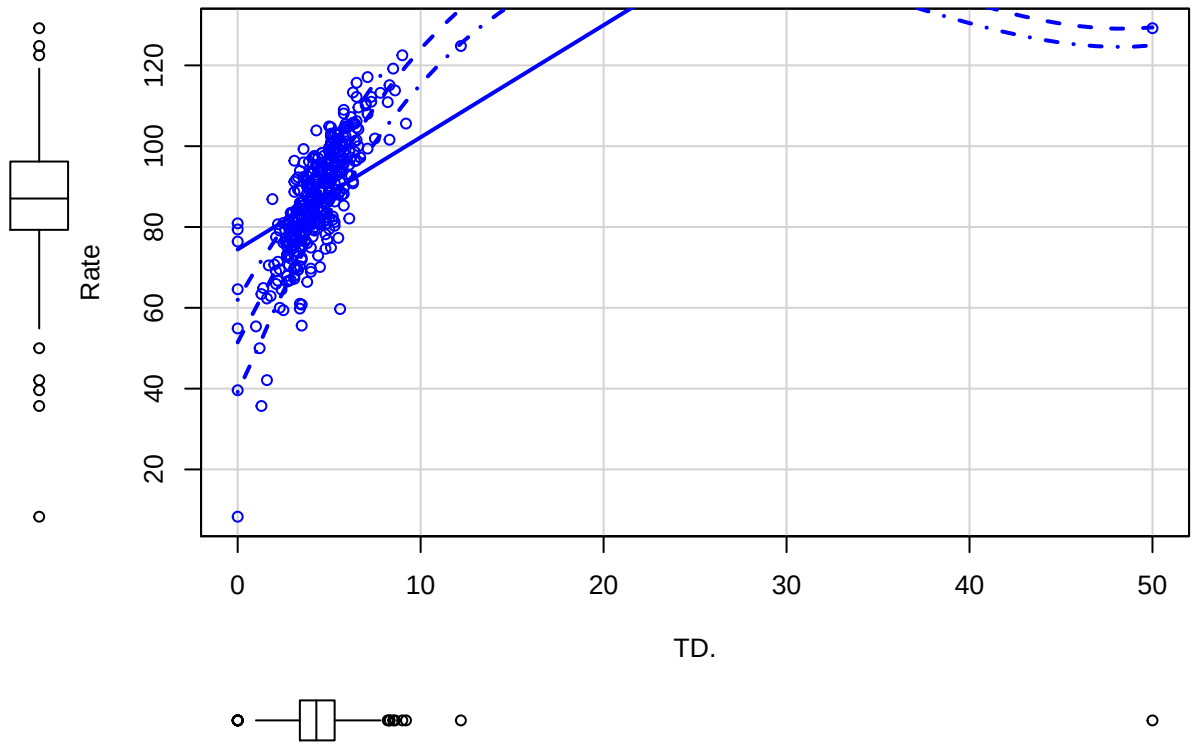

```
qqPlot(TD)
```



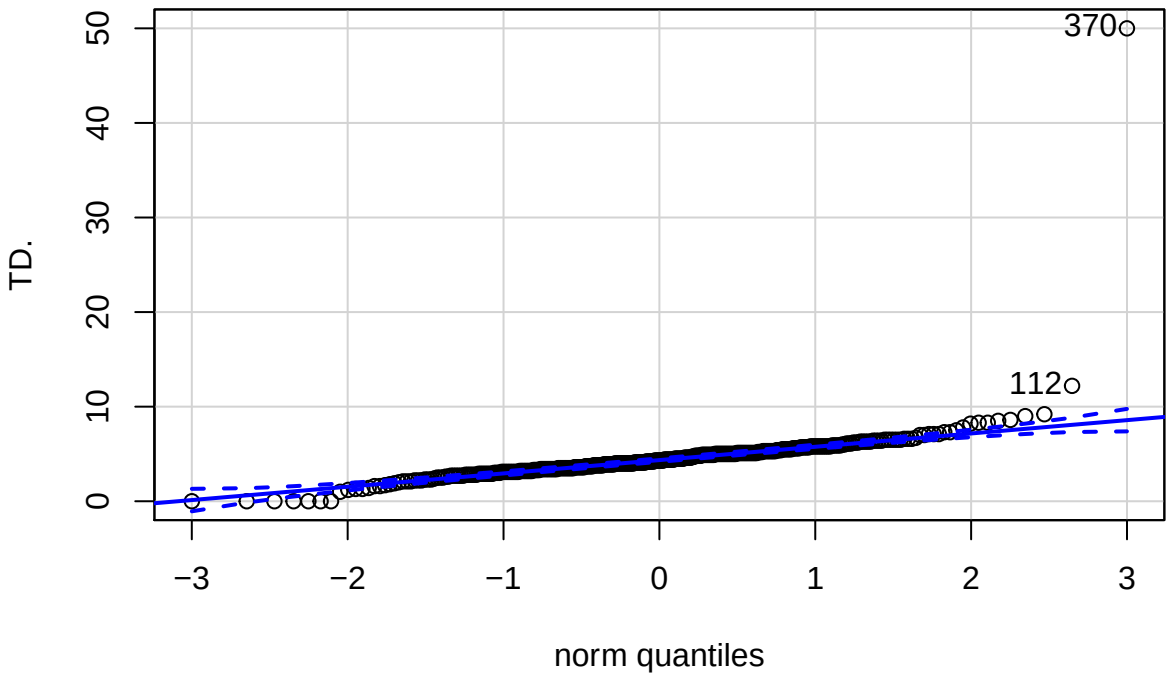

```
## [1] 153 339
```

Clearly, Passer Rating increases as the number of Touchdowns scored increases. This is no surprise, and a welcome conformation of how this variable should work in our model. On the quantile plot, it becomes clear that we might consider removing some players with 0 touchdowns, so as to better fit a normal distribution. Players with 0 touchdowns are not simply bad players, more likely they are backup players or often in other positions. Therefore we could exclude them from our model since we want to be evaluating quarterbacks who play the majority of the game.

```
scatterplot(TD.,Rate)
```



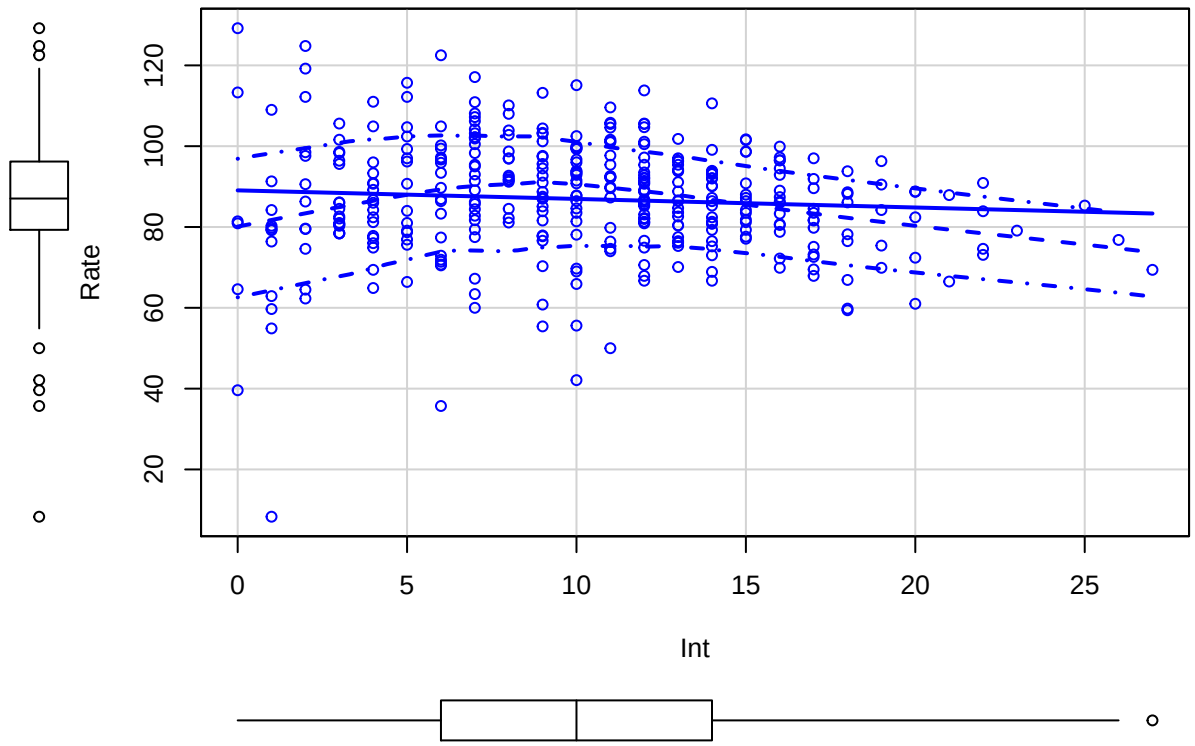

```
qqPlot(TD.)
```



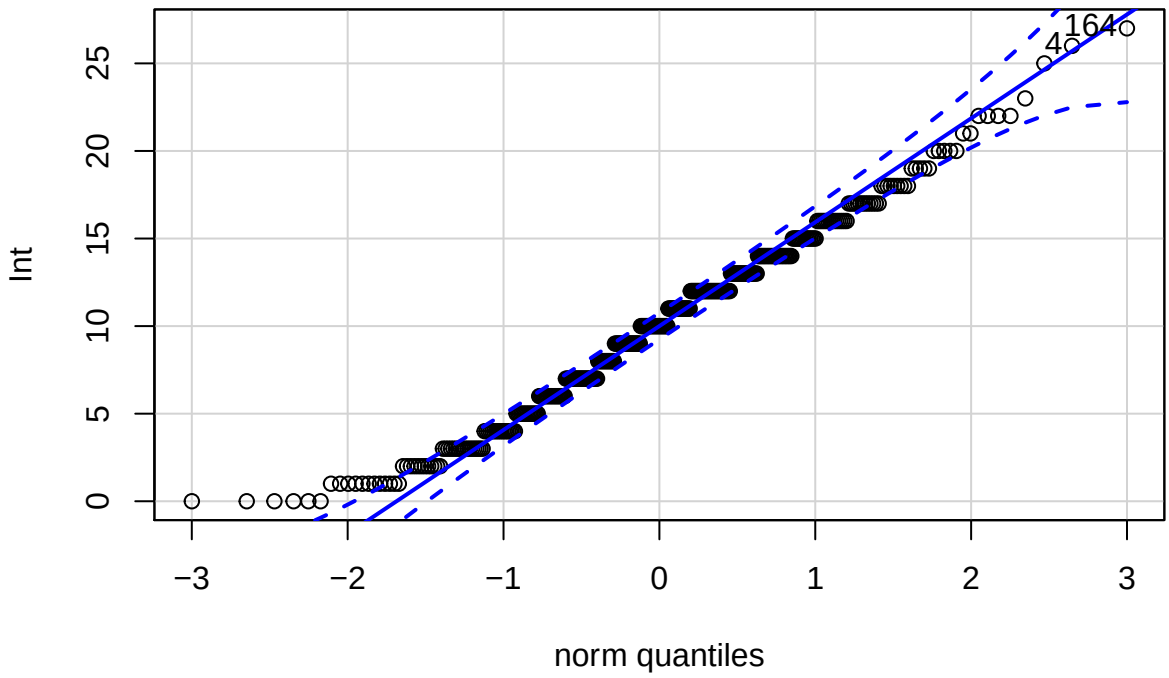

```
## [1] 370 112
```

The scatterplot and quantile plots of Touchdown Percentage reveal an outlier that desperately needs to be dealth with. Once that is taken care of, a clear linear relationship will be established and the distribution will be normalized. The player in question only has 2 pass attempts, so perhaps we should resrtict our sample to be just players with significant numbers of pass attempts.

```
scatterplot(Int,Rate)
```



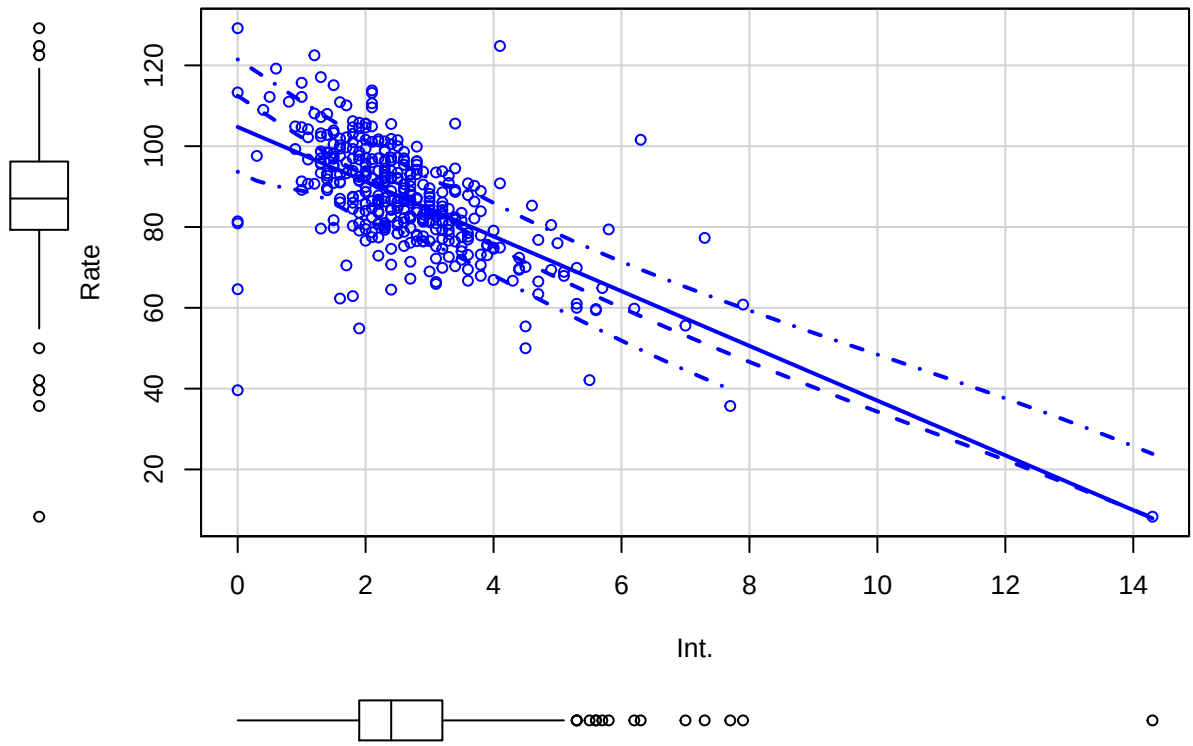

```
qqPlot(Int)
```



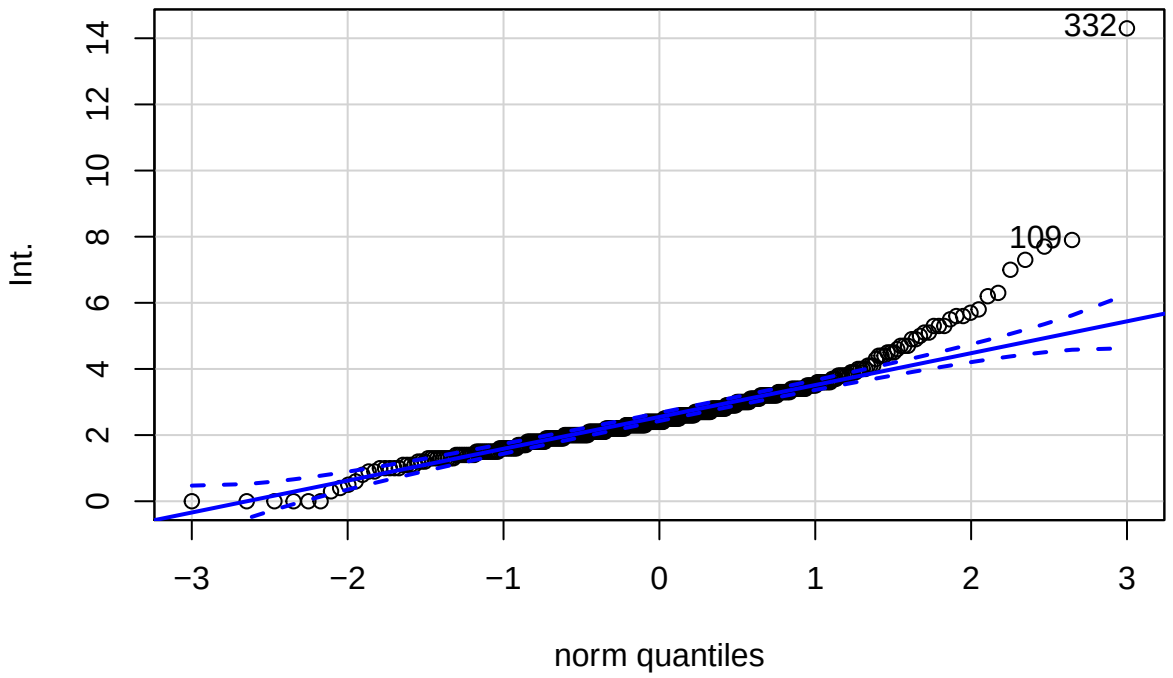

```
## [1] 164   4
```

There is not such a clear relationship between Interceptions and Passer Rating. Some players will get a lower rating even with less interceptions. One trend is very clear, though, that players with 15+ or 20+ interceptions will absolutely not garner a rating of over 100. The quantile plot shows that the number of interceptions for every player is normally distributed, especially if we remove some of the players with less pass attempts (playing time).

```
scatterplot(Int.,Rate)
```



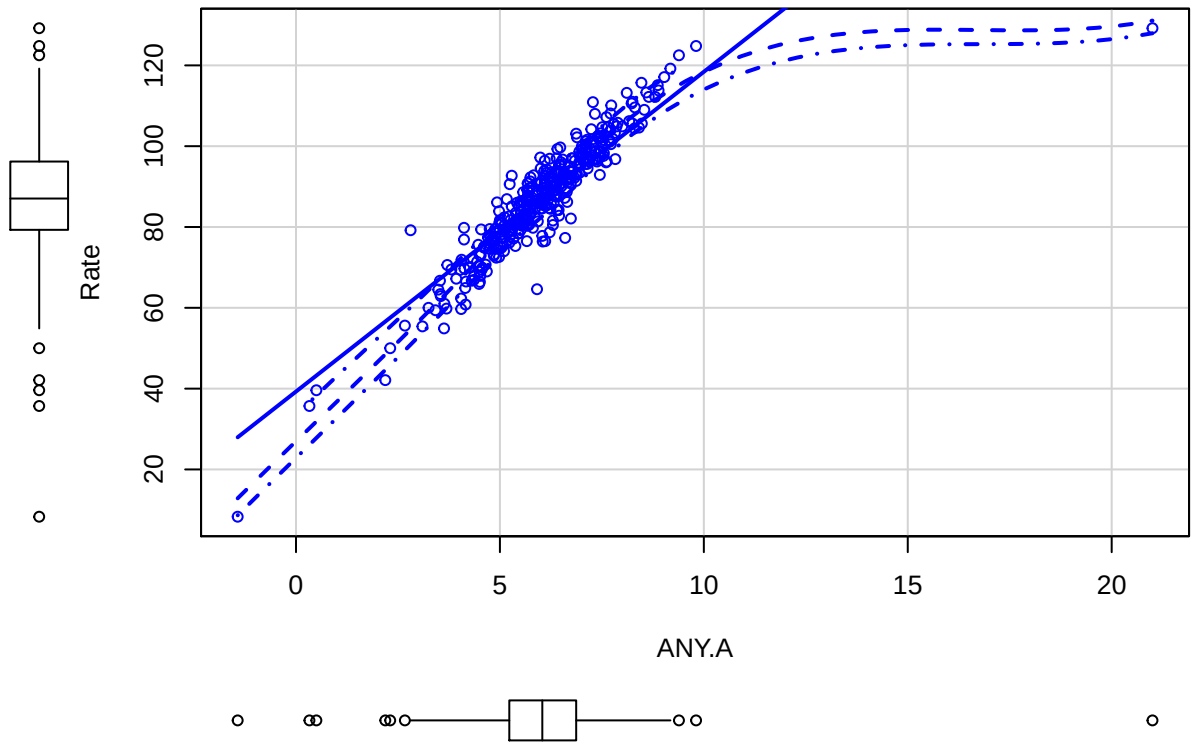

```
qqPlot(Int.)
```



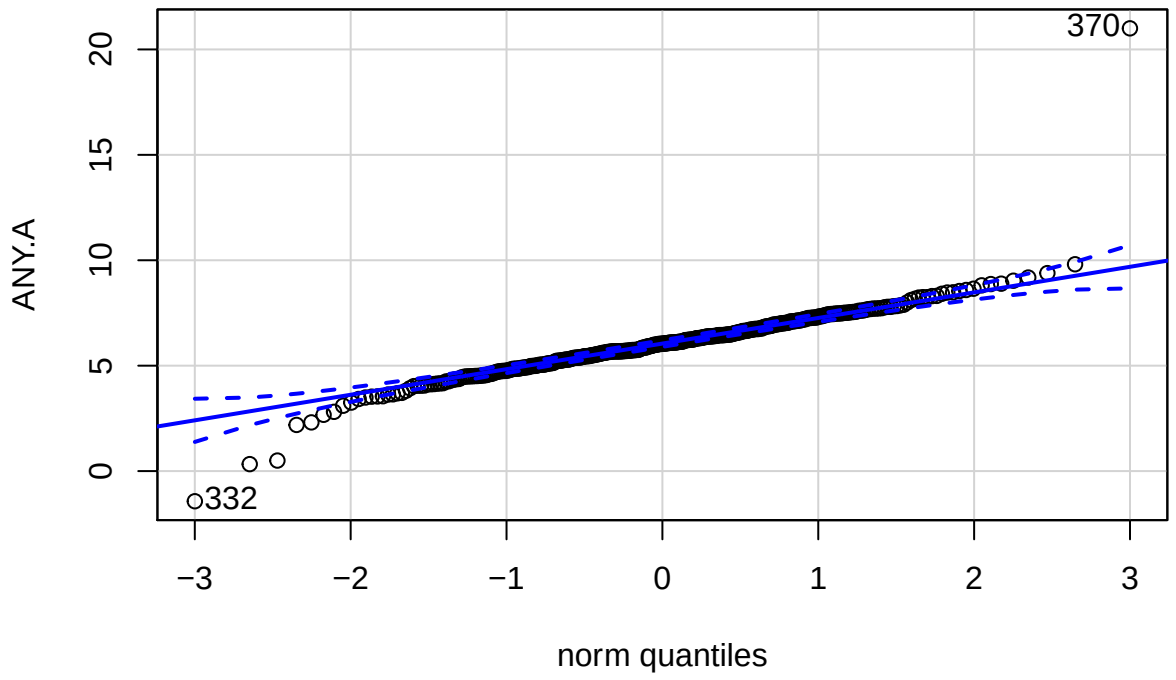

```
## [1] 332 109
```

Generally, Passer Rating decreases as Interception Percentage increases, although there are a few outliers on every side of the data. Some players are still able to put up an above average rating even with more frequent interceptions, and vice versa some players garner a lower rating even though they turn over the ball often. The removal of these outliers will still leave the data somewhat outside of a normal distribution, so a transformation might be considered for this variable.

```
scatterplot(ANY.A,Rate)
```




```
qqPlot(ANY.A)
```



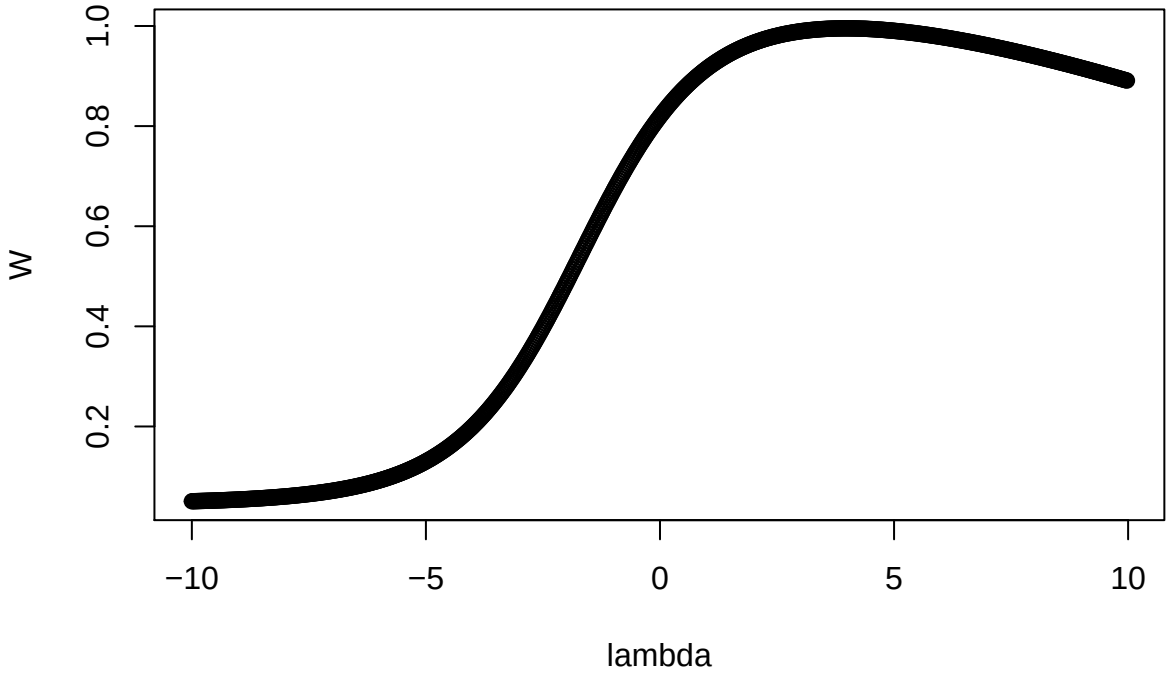

```
## [1] 370 332
```

This relationship is strongly linear, however we will need to remove an outlier that is all the way on the right side of the graph. Without it, the best fit line and prediction interval would be much better. Upon further inspection, that data point was a backup player with only 2 pass attempts, so we will exclude him from our data analysis. The quantile plot shows that our data will be well-distributed upon the condition of removing those outliers.

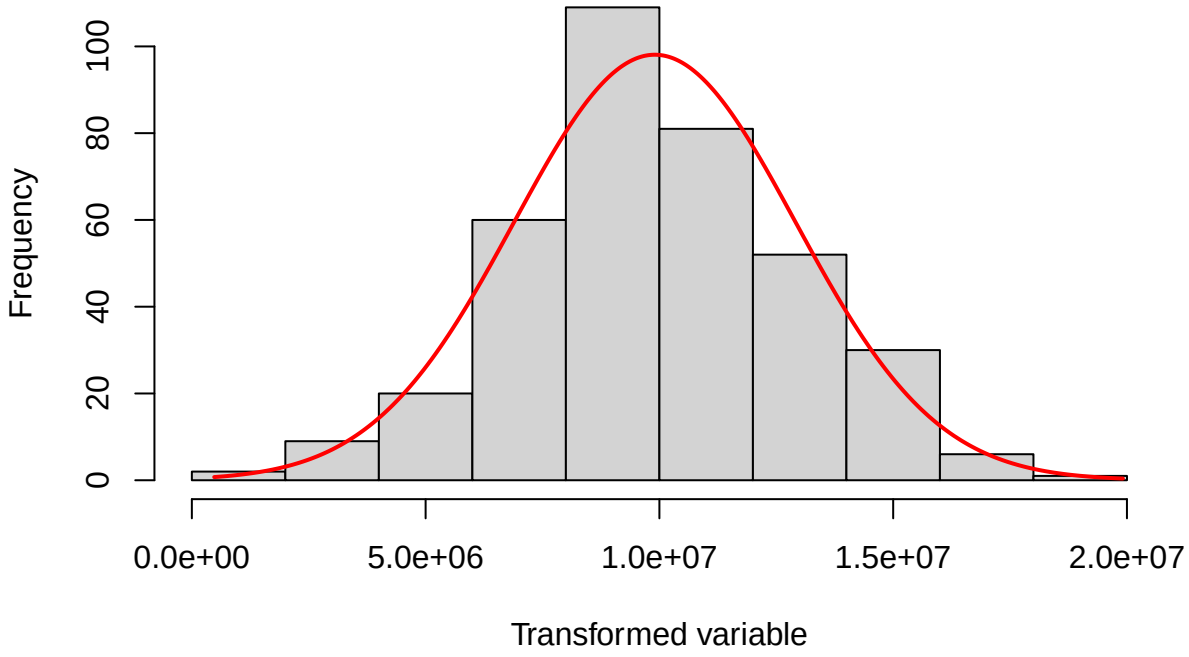
Variable Transformations

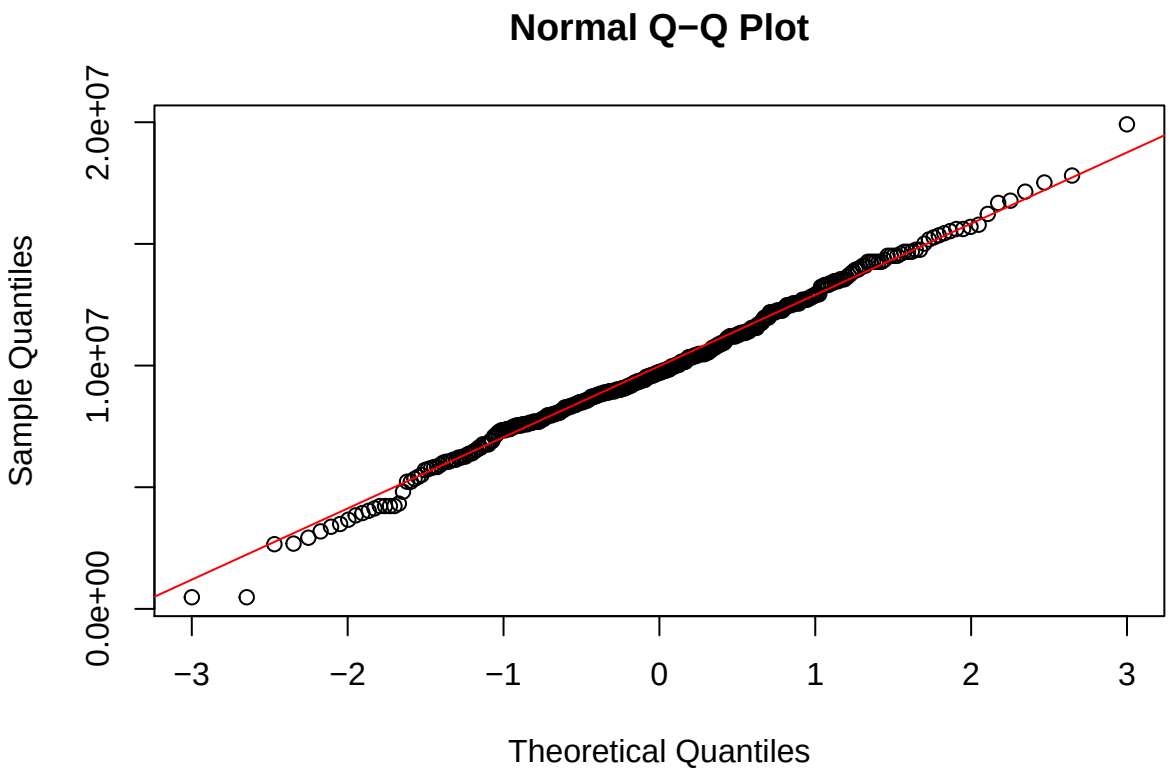
Our above analysis indicated that we would remove AY.A and NY.A from our analysis, and also that we could transform Completion Percentage and Interception Percentage to be better distributed. We want the data to follow a normal distribution so as to conform to the assumptions of the linear regression process. If the nature of the relationship between variables changed as x increases, then a linear coefficient will be less useful to make a prediction, as it will be less accurate on one side of the data.

```
library(rcompanion)
Cmp.trans <- transformTukey(Cmp.)
```

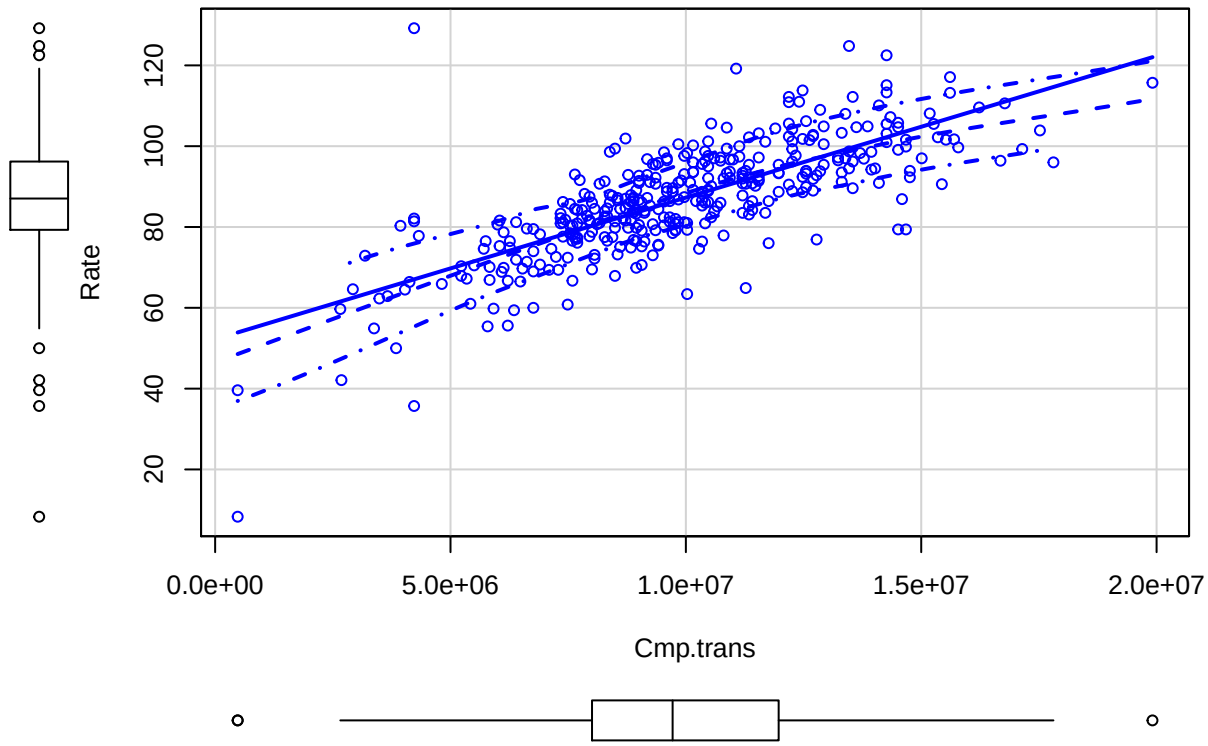



```
##
##     lambda      W Shapiro.p.value
## 557    3.9 0.9953          0.3198
##
## if (lambda >  0){TRANS = x ^ lambda}
## if (lambda == 0){TRANS = log(x)}
## if (lambda <  0){TRANS = -1 * x ^ lambda}
```

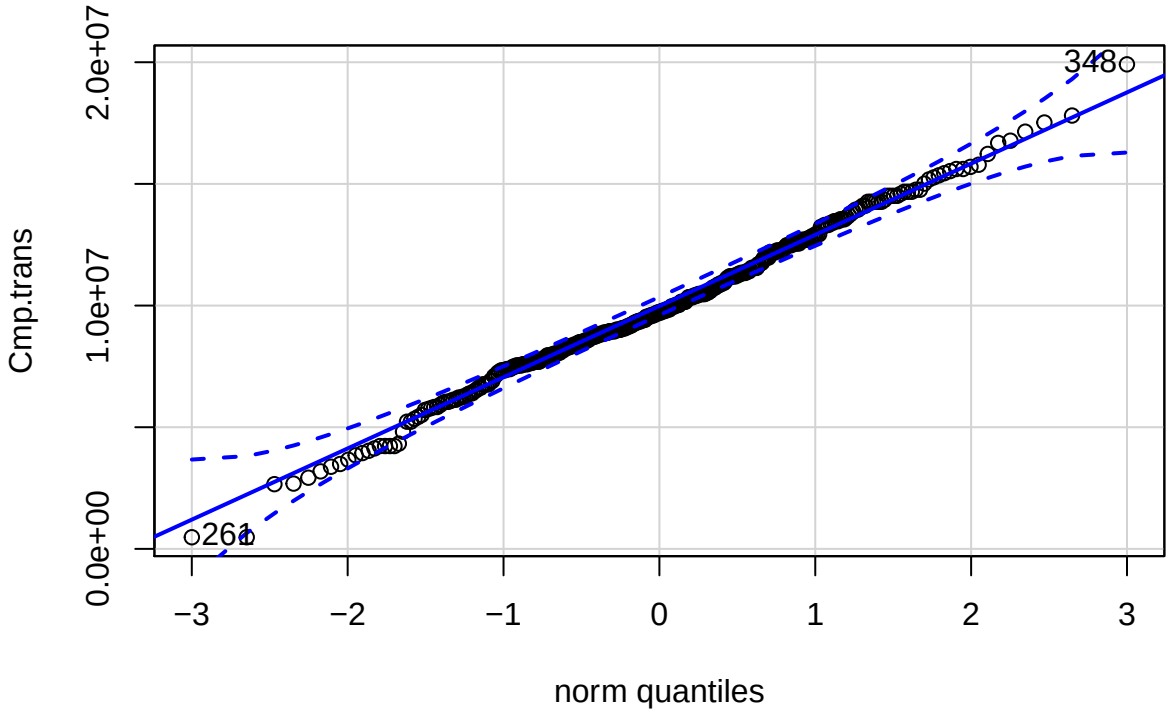

Frequency
0
20
40
60
80
100
0.0e+00
5.0e+06
1.0e+07
1.5e+07
2.0e+07
Transformed variable




```
scatterplot(Cmp.trans,Rate)
```



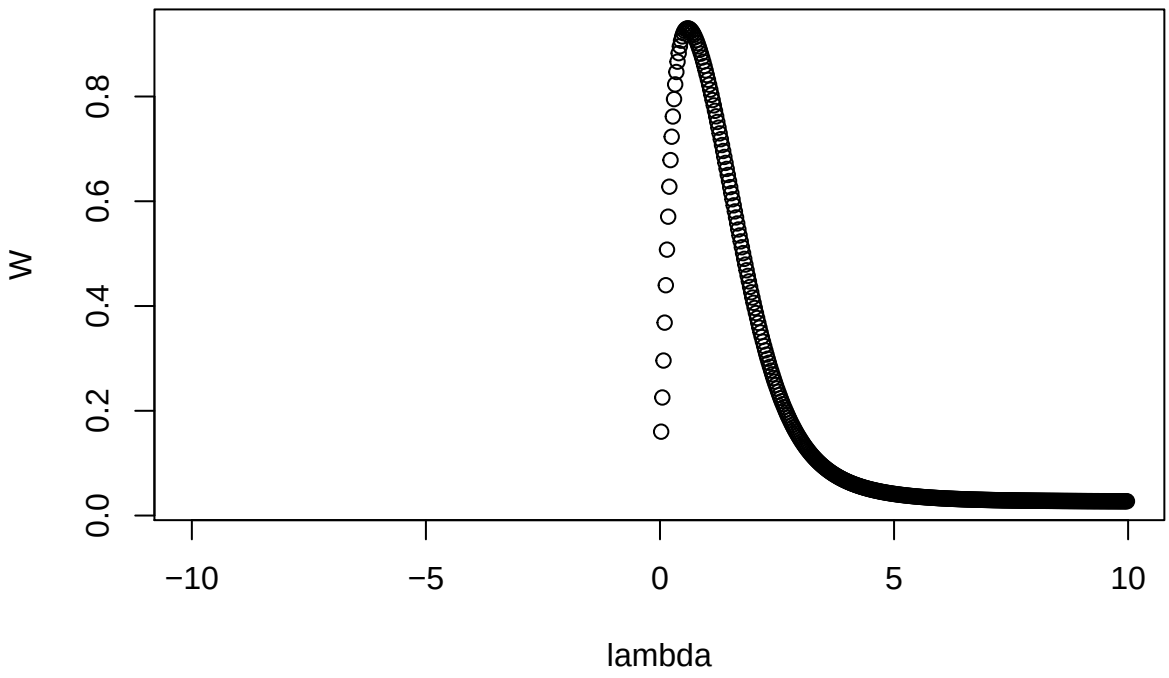

```
qqPlot(Cmp.trans)
```



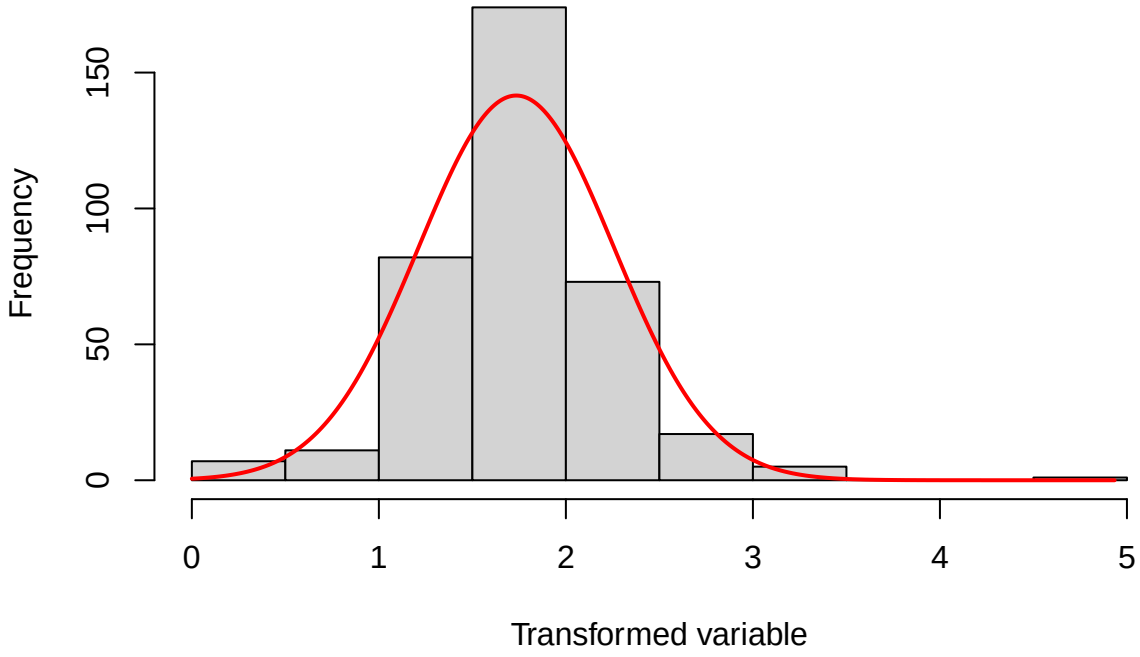

```
## [1] 348 261
```

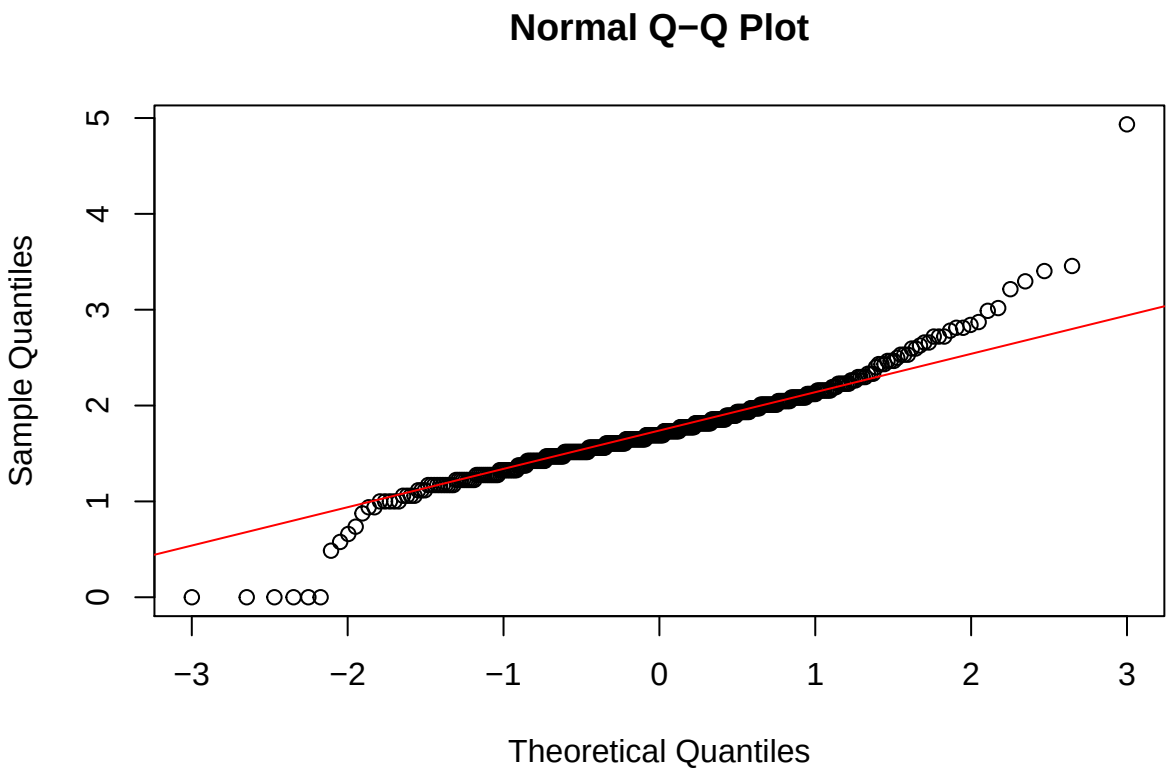
Using Tukey's Ladder of Powers suggests that taking Completion Percentage to the power of 3.9 would result in the most normally distributed data.

```
Int.trans <- transformTukey(Int.)
```

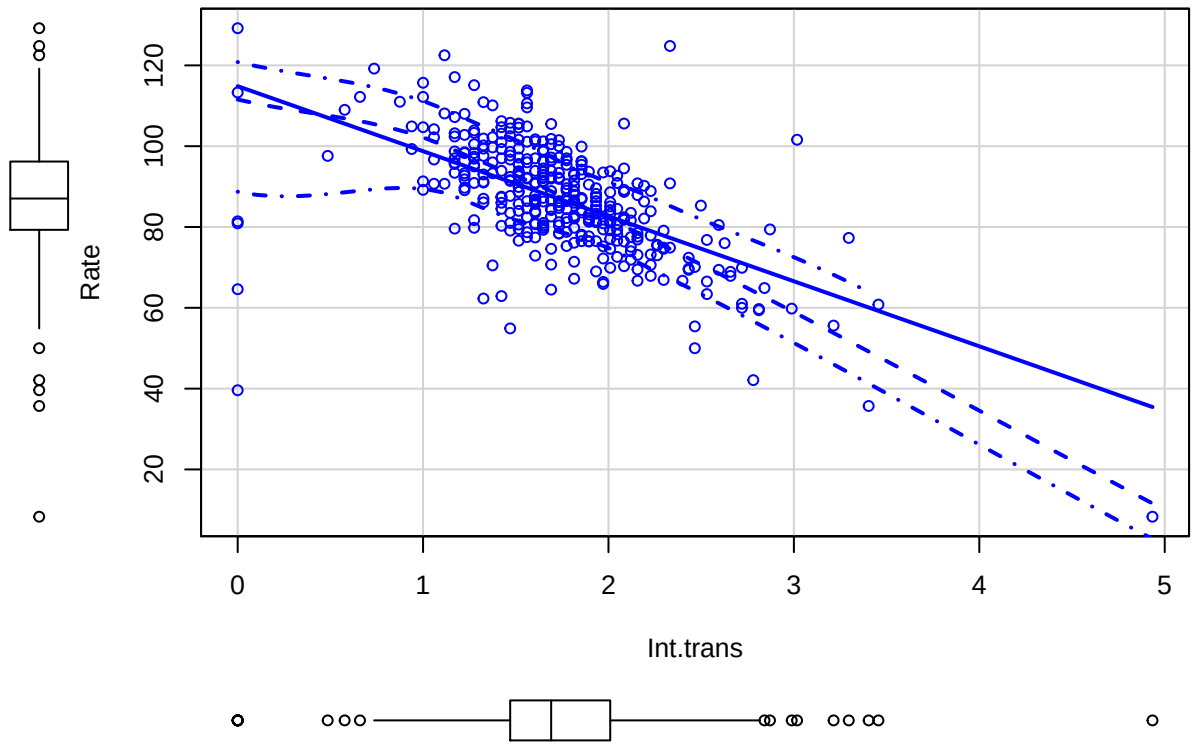



```
##
##     lambda      W Shapiro.p.value
## 425    0.6 0.9301       3.814e-12
##
## if (lambda >  0){TRANS = x ^ lambda}
## if (lambda == 0){TRANS = log(x)}
## if (lambda <  0){TRANS = -1 * x ^ lambda}
```

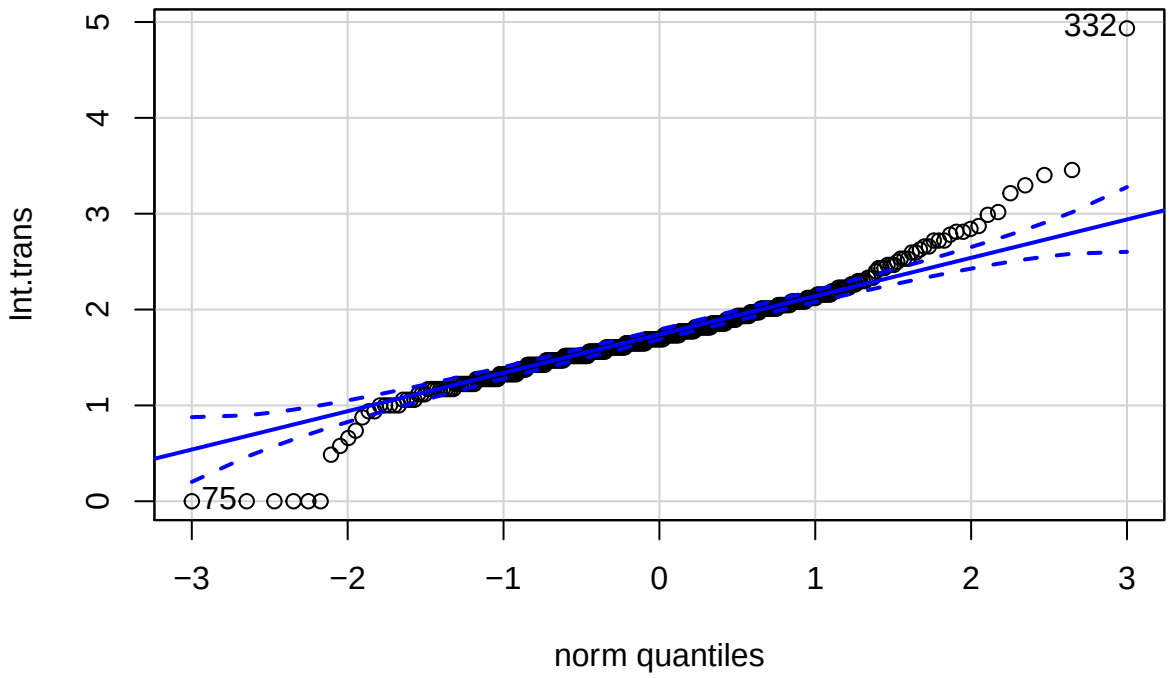

Frequency
0
50
100
150
0
1
2
3
4
5
Transformed variable




```
scatterplot(Int.trans,Rate)
```




```
qqPlot(Int.trans)
```




```
## [1] 332  75
```

Tukey's Ladder of Powers concludes that raising Interception Percentage to the 0.6 power would result in more normally distributed data. Removing some outliers would further improve our distribution.

Outliers

As has been discussed previously, some players do not accrue many pass attempts because they are backup players or play mainly at another position. For example, one of the players is a kicker with two pass attempts, one of which was successful for 22 yards and a touchdown. This was surely a great play with a positive outcome, and therefore the player is assigned a high rating. However, since we are using linear regression, we do not want to include these fringe cases in our model, as we want our model to explain the performances of quarterbacks that play a lot of the time.

From here on out, we will only analyze data from players with more than 50 pass attempts:

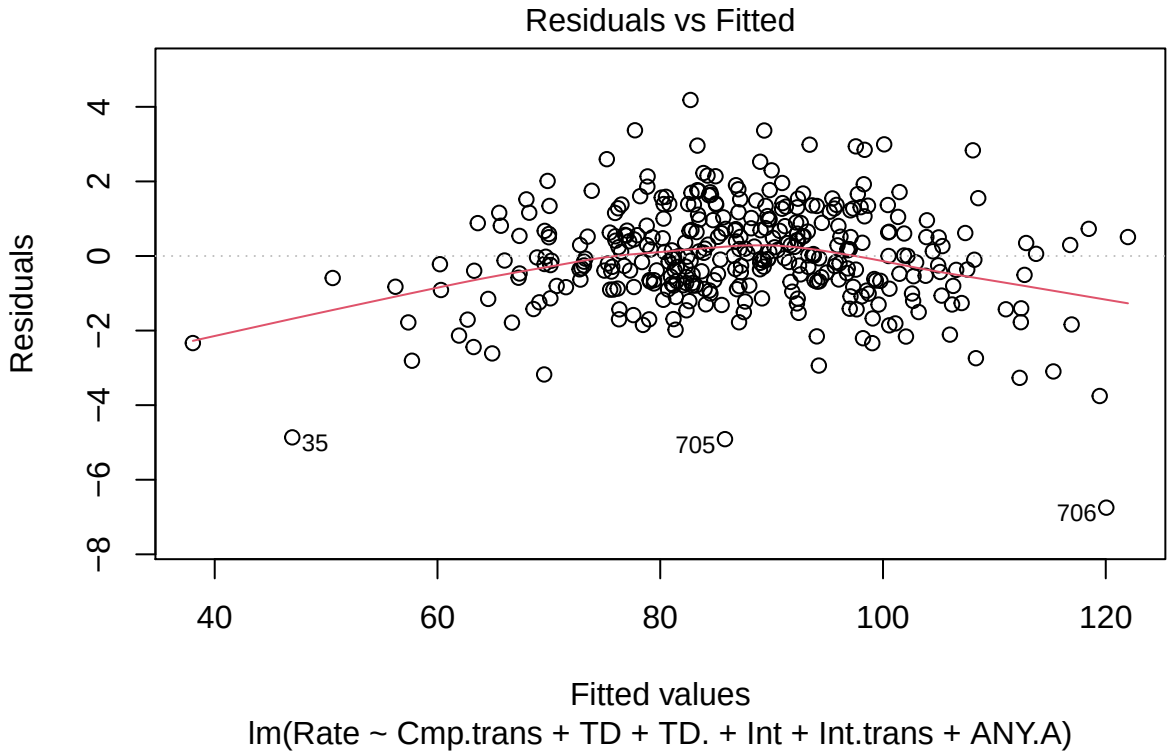
```
out_na_passdata <- subset(napassdata, Att >= 50)
```

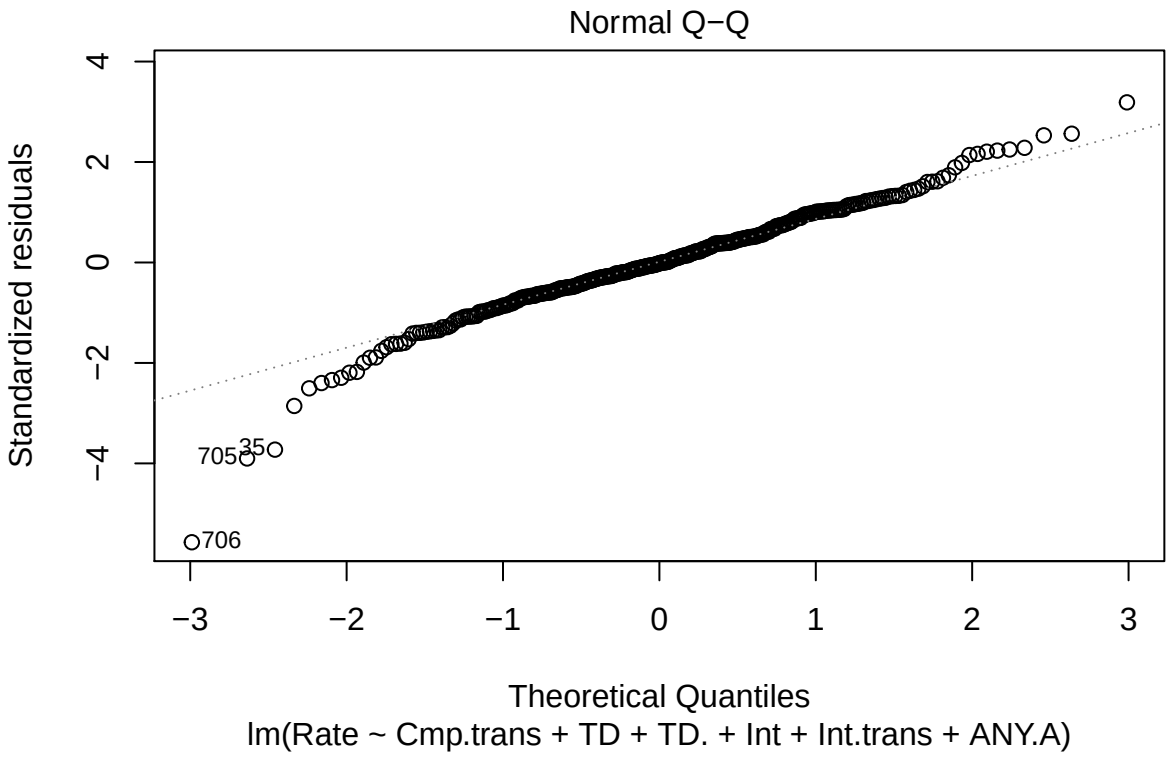
Let's run a quick regression to check if there are any more outliers obvious:

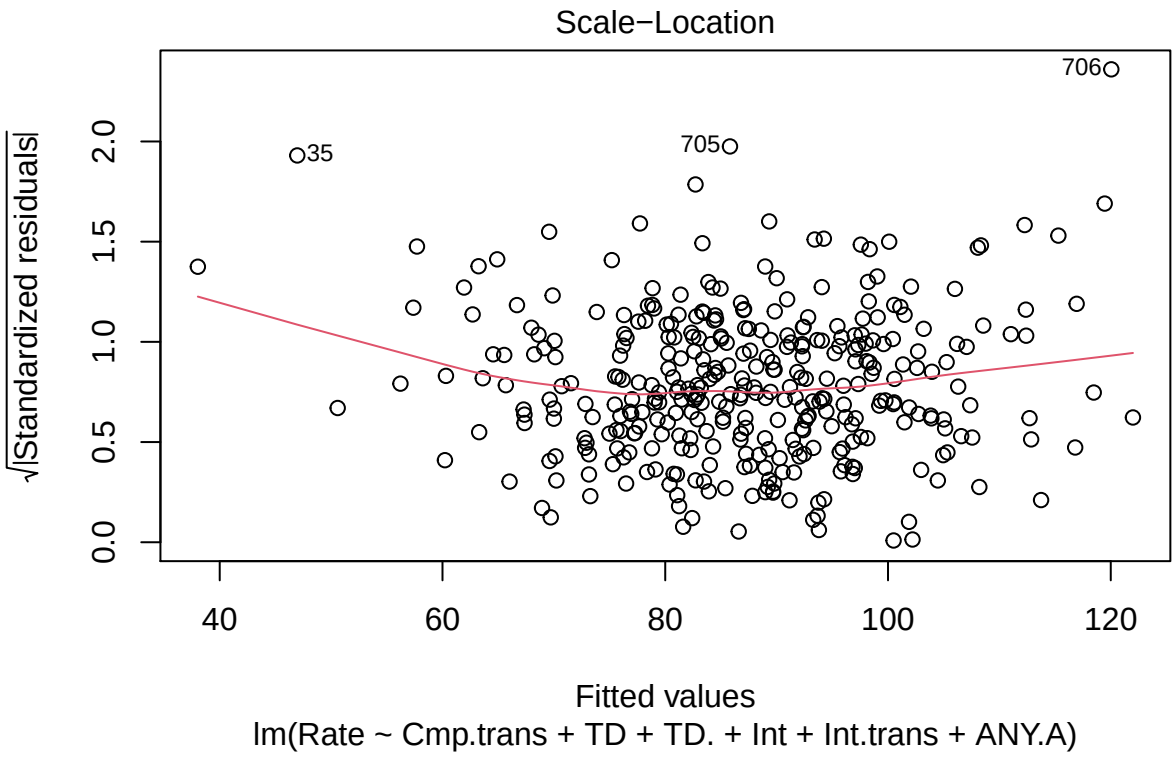
```
detach(napassdata)
attach(out_na_passdata)
```

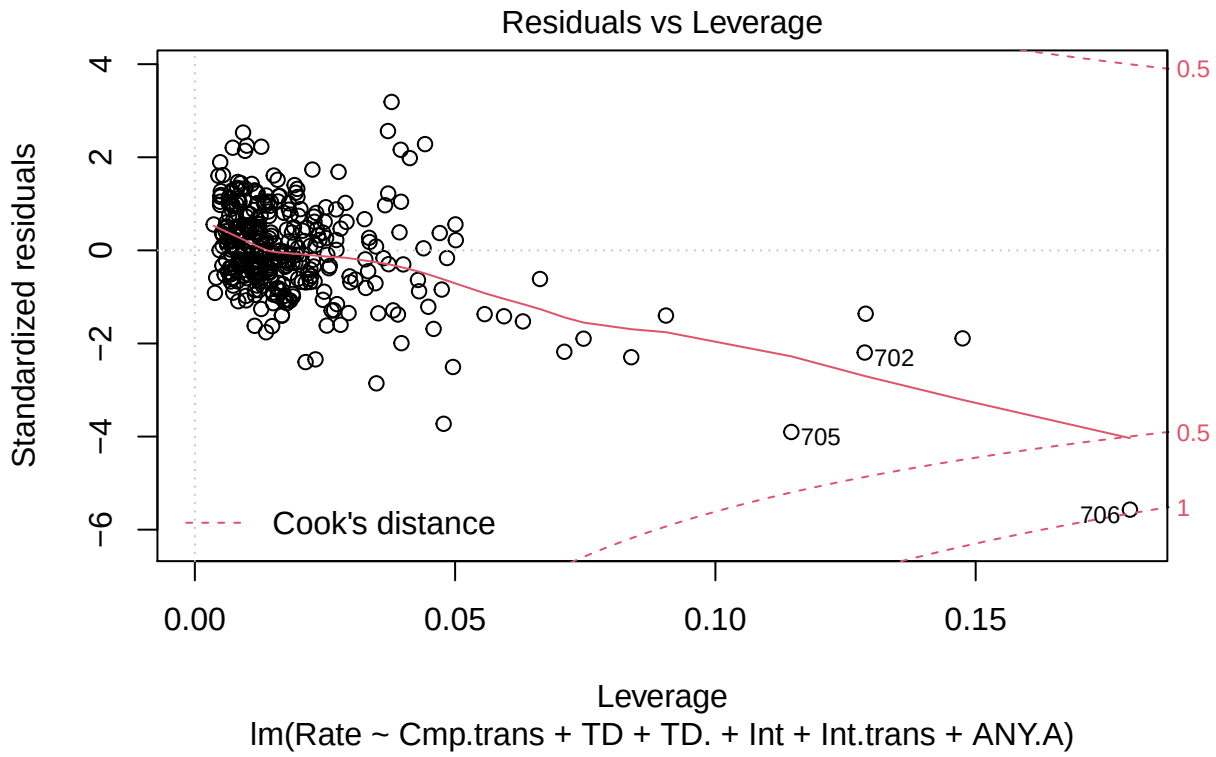
```
## The following object is masked _by_ .GlobalEnv:
##
##     Rk
```

```
Cmp.trans <- transformTukey(Cmp., plotit = FALSE, quiet = TRUE)
Int.trans <- transformTukey(Int., plotit = FALSE, quiet = TRUE)
outlier_model_test <- lm(Rate ~ Cmp.trans + TD + TD.
                         + Int + Int.trans + ANY.A,
                         data = out_na_passdata)
plot(outlier_model_test)
```




Normal Q−Q
Standardized residuals
4
2
0
−2
−4
705
35
706
−3
−2
−1
0
1
2
3
Theoretical Quantiles
lm(Rate ~ Cmp.trans + TD + TD. + Int + Int.trans + ANY.A)


Scale−Location
706
35
705
√|Standardized residuals|
0.0
0.5
1.0
1.5
2.0
40
60
80
100
120
Fitted values
lm(Rate ~ Cmp.trans + TD + TD. + Int + Int.trans + ANY.A)

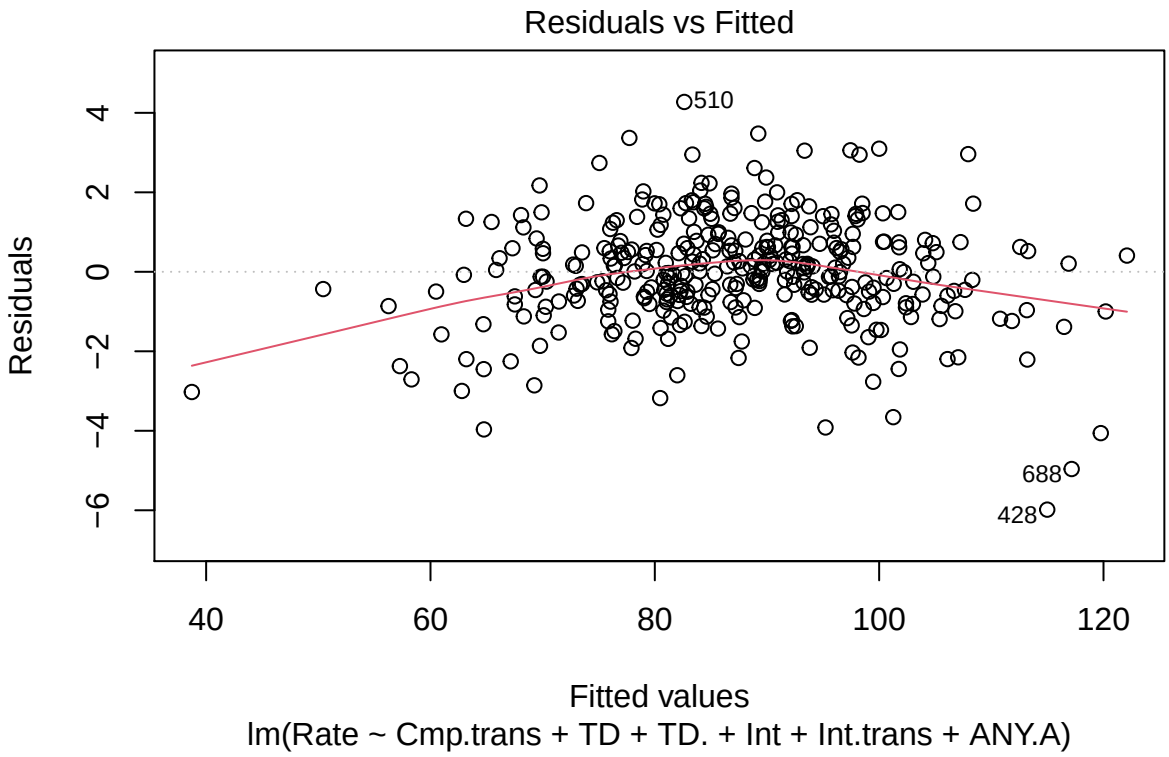


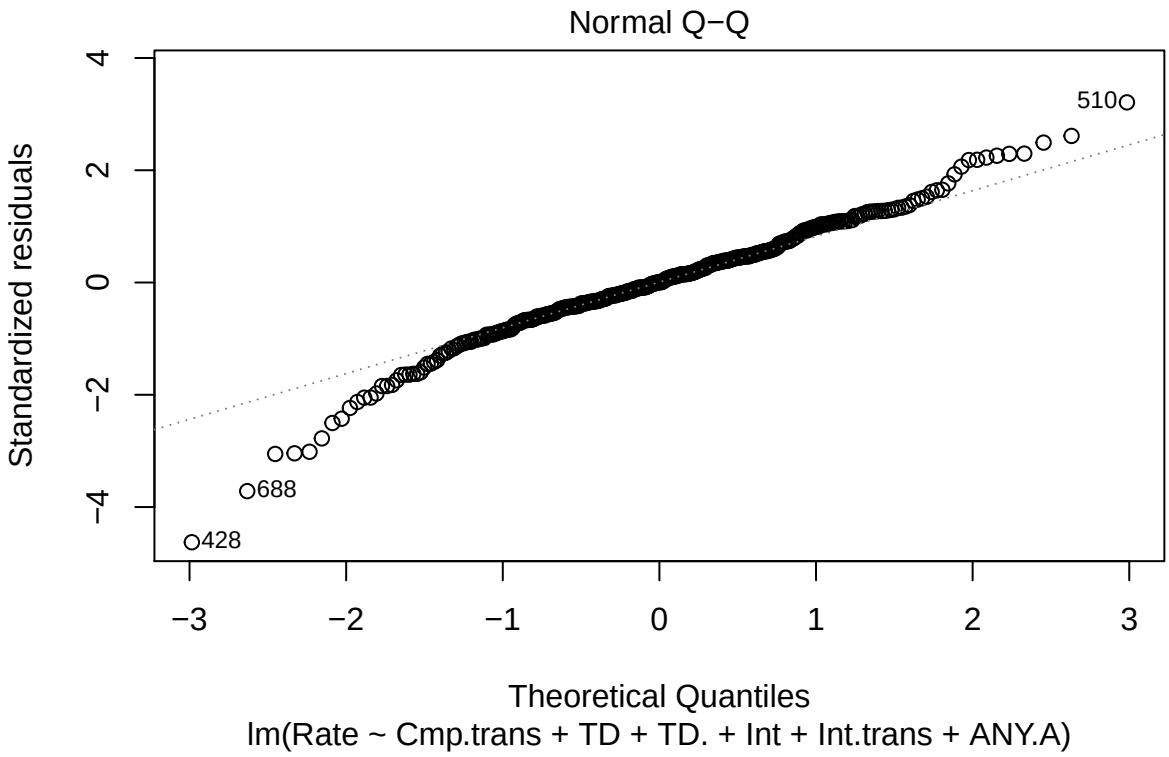
A quick look at the residual plots shows us that we could remove a few more outliers manually and improve the fit of our model. The outliers visible on the plots include:
Jimmy Garoppolo - a backup QB who started 2 games that season
Tom Savage - a backup QB who started 2 games that season
Matt Moore - a backup QB who started 2 games that season
Derek Anderson - a backup QB who started 7 games that season

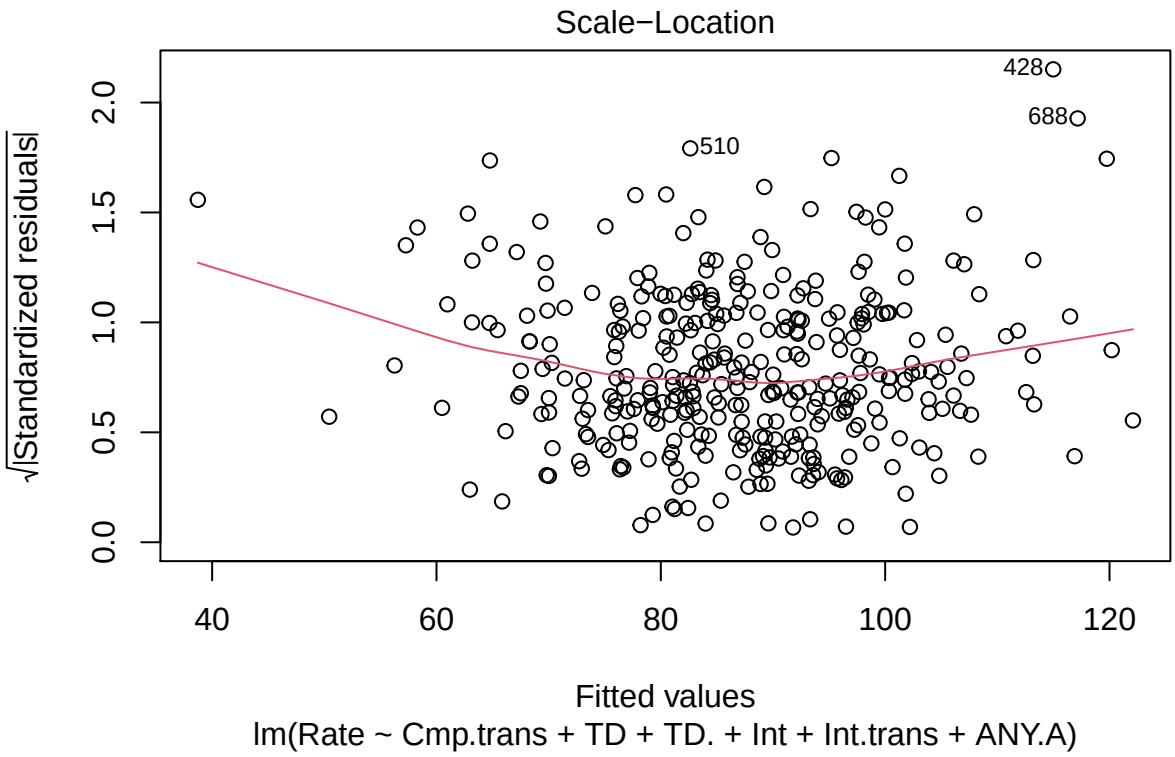
```
removeoutliers <- out_na_passdata[-c(31,72,282,284,285), ]
detach(out_na_passdata)
attach(removeoutliers)
```

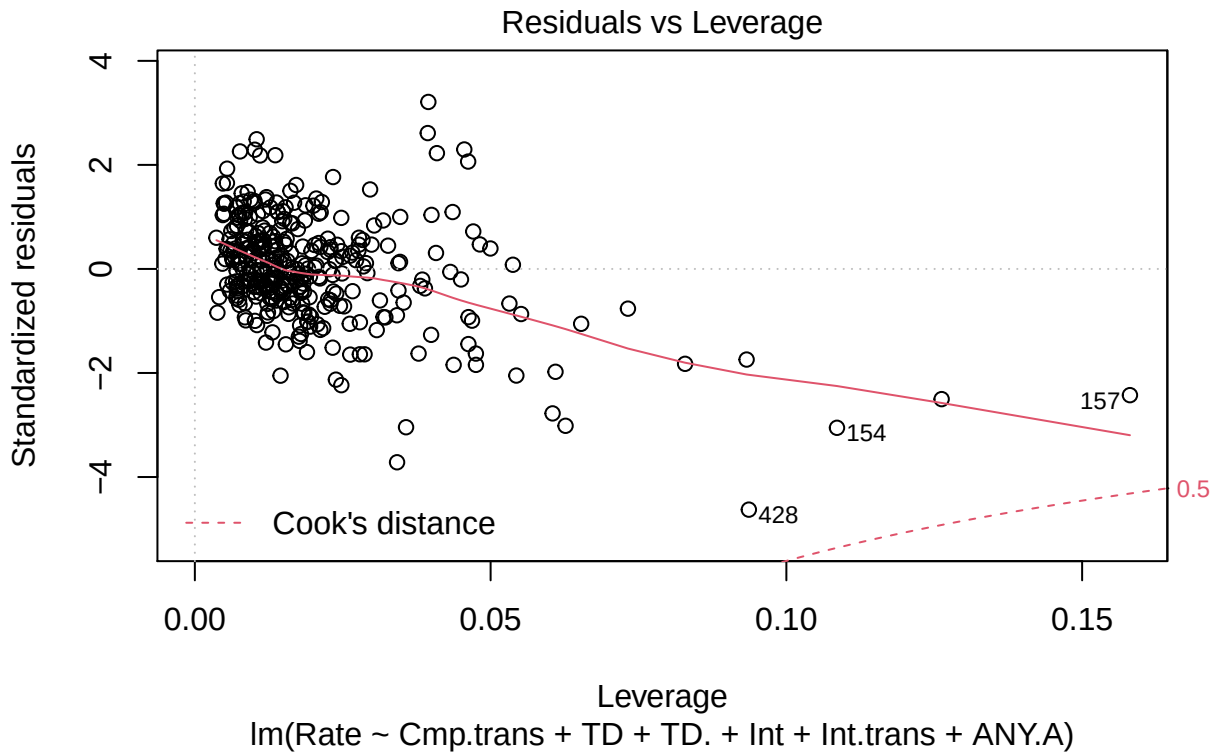
```
## The following object is masked _by_ .GlobalEnv:
##
##     Rk
```

```
Cmp.trans <- transformTukey(Cmp., plotit = FALSE, quiet = TRUE)
Int.trans <- transformTukey(Int., plotit = FALSE, quiet = TRUE)
outlier_model_test_b <- lm(Rate ~ Cmp.trans + TD + TD.
                           + Int + Int.trans + ANY.A,
                           data = removeoutliers)
plot(outlier_model_test_b)
```


Residuals vs Fitted
Residuals
Fitted values
lm(Rate ~ Cmp.trans + TD + TD. + Int + Int.trans + ANY.A)
510
688
428
40
60
80
100
120
−6
−4
−2
0
2
4


Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(Rate ~ Cmp.trans + TD + TD. + Int + Int.trans + ANY.A)
510
688
428
−4
−2
0
2
4
−3
−2
−1
0
1
2
3


Scale−Location
√|Standardized residuals|
0.0
0.5
1.0
1.5
2.0
40
60
80
100
120
428
688
510
Fitted values
lm(Rate ~ Cmp.trans + TD + TD. + Int + Int.trans + ANY.A)



By removing a few outliers manually, we've improved the residual plots while making sure we are being mindful of which particular players we are removing.

Imputing NAs

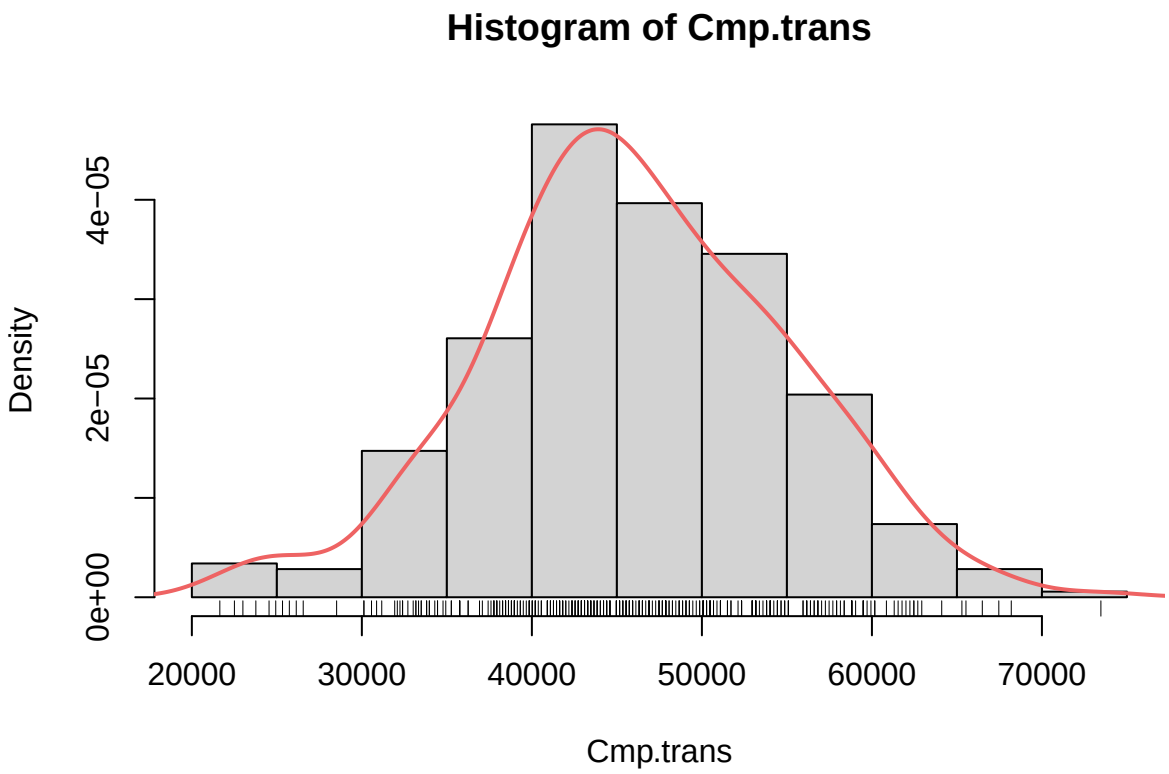
Earlier, we discussed the removal of NAs in our original data set, and at this point we have no more to remove.

Model Building

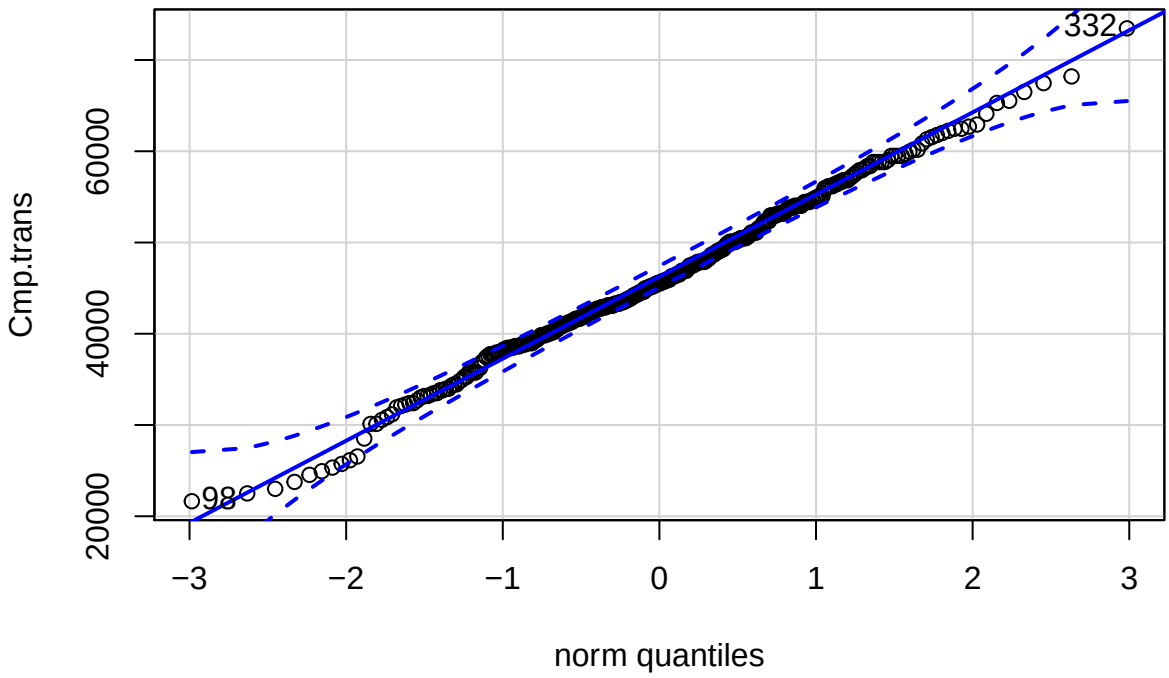
Evaluating Variable Transformations

The two variables we subjected to transformations include:
Completion Percentage
Interception Percentage

```
hist(Cmp.trans, breaks = "FD", freq = FALSE)
lines(density(Cmp.trans),lwd = 2, col ="indianred2")
rug(Cmp.trans)
```

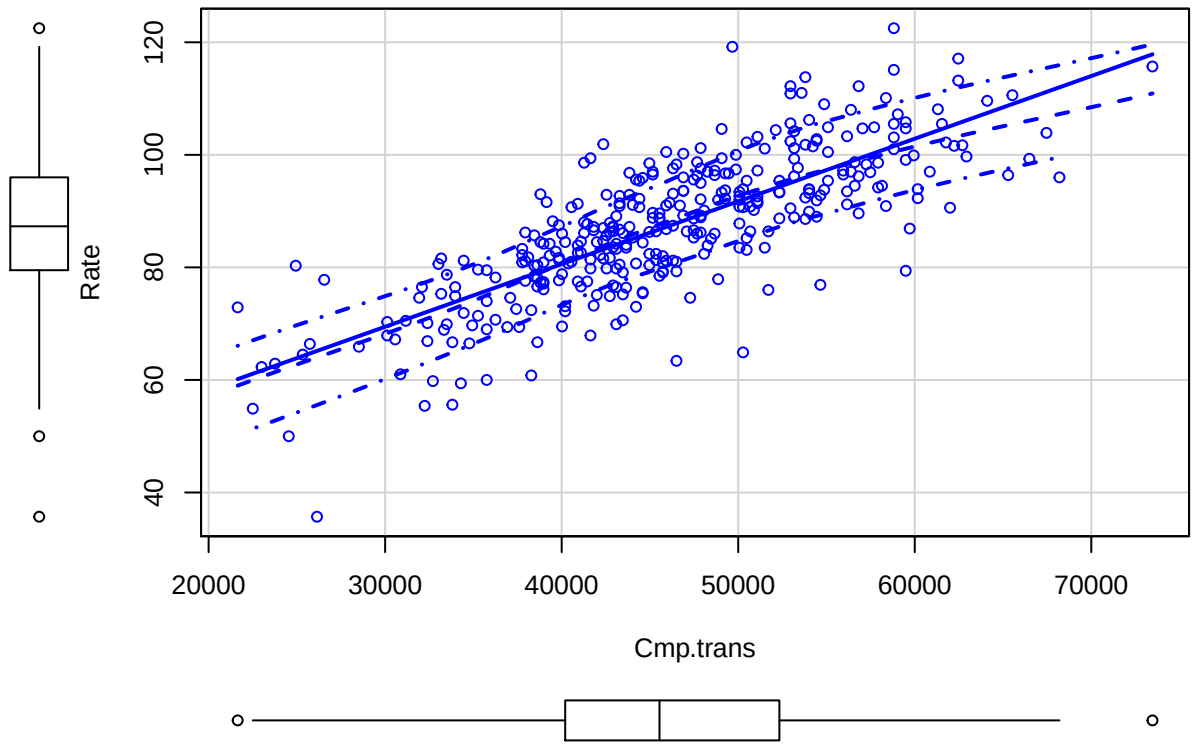



```
qqPlot(Cmp.trans)
```



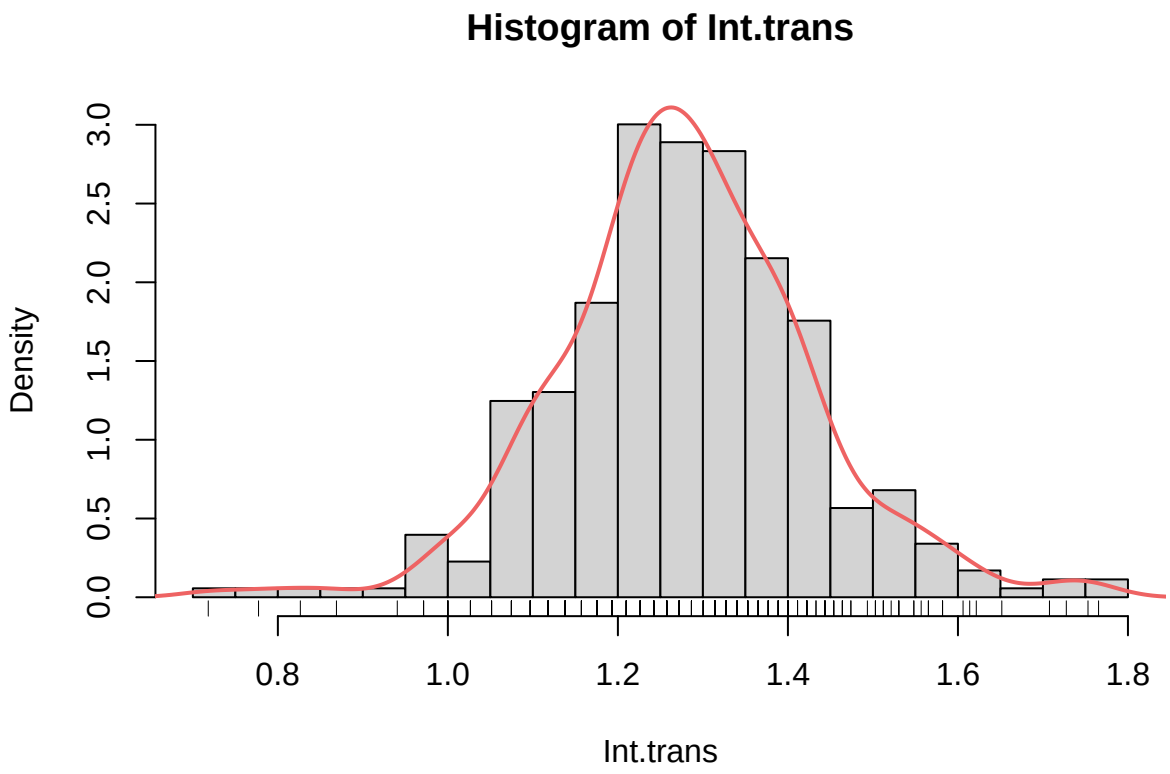

```
## [1] 332  98
```

```
scatterplot(Cmp.trans,Rate)
```

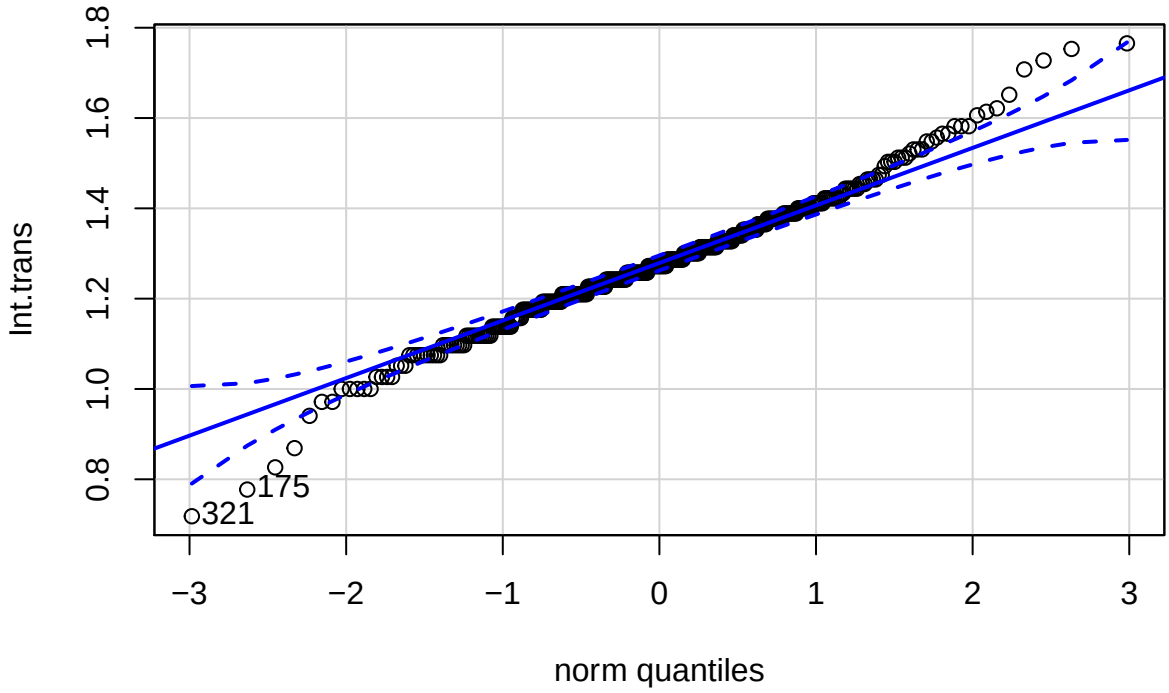


After a power transformation, we see that Completion Percentage is nicely distributed around a mean, with no significant skew either way. The quantile plot shows that all the data fit approximately within a normal distribution, as is also seen on the density plot. The scatterplot shows a clear linear relationship between the transformed variable and the target variable.

```
hist(Int.trans, breaks = "FD", freq = FALSE)
lines(density(Int.trans),lwd = 2, col ="indianred2")
rug(Int.trans)
```

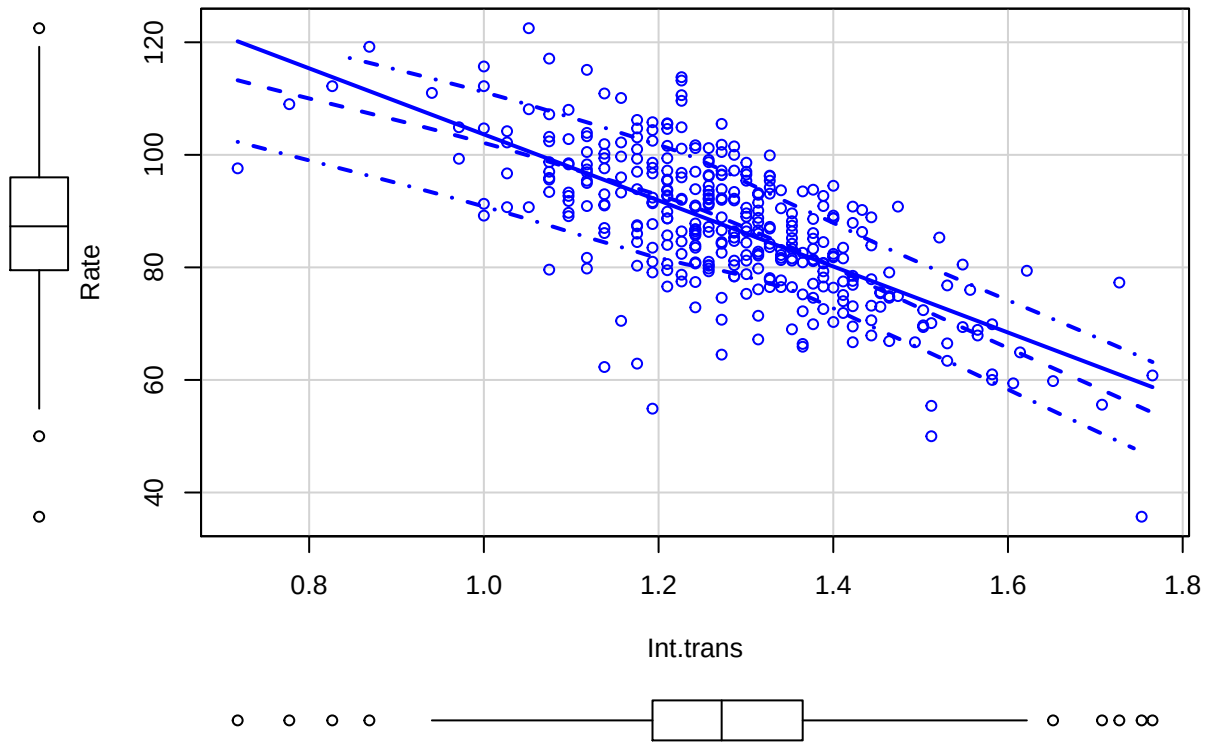



```
qqPlot(Int.trans)
```




```
## [1] 321 175
```

```
scatterplot(Int.trans,Rate)
```



When we create the same plots for the transformed version of Interception Percentage, we see similar results: a normal distribution and a clear linear relationship. In this case, higher interception % clearly leads to a lower rating. The quantile plot shows that the data does not fit the normal distribution quite as well as the other transformed variable, there are no more outliers we think are appropriate for removal.

Testing for Multicollinearity

```
vif(outlier_model_test_b)
```

```
## Cmp.trans        TD       TD.       Int Int.trans     ANY.A
##  1.909790  8.327712  4.971595  4.664371  3.998925  5.533106
```

Calculating the Variance Inflation Factor reveals that none of our variables suffer from multicollinearity, using VIF<10 as a threshold. It appears Touchdowns (TD) and Adjusted Net Yards per Attempt (ANY.A) increase variance the most, but not so much that we will need to remove them.

Testing for Heteroskedasticity

```
library(lmtest)
```

```
## Loading required package: zoo
```

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

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

```
bptest(outlier_model_test_b, data = removeoutliers)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  outlier_model_test_b
## BP = 41.07, df = 6, p-value = 2.805e-07
```

Using the Breusch-Pagan Test to test for heteroskedasticity results in an very low p-value, indicating that there may be heteroskedasticity issues we need to resolve.

We will use weighted least squares to mitigate the heteroskedasticity problem:

```
resi <- outlier_model_test_b$residuals
varfunc.ols <- lm(log(resi^2) ~ Cmp.trans + TD + TD.
                        + Int + Int.trans + ANY.A,
                        data = removeoutliers)
varfunc <- exp(varfunc.ols$fitted.values)
model.gls <- lm(Rate ~ Cmp.trans + TD + TD.
                        + Int + Int.trans + ANY.A,
               weights = 1/sqrt(varfunc), data = removeoutliers)
```

Testing for Misspecification

```
resettest(Rate ~ Cmp.trans + TD + TD. +
                 Int + Int.trans + ANY.A,
                 power = 2:3,
                 type = "regressor",
                 data = removeoutliers)
```

```
##
##  RESET test
##
## data:  Rate ~ Cmp.trans + TD + TD. + Int + Int.trans + ANY.A
## RESET = 22.67, df1 = 12, df2 = 334, p-value < 2.2e-16
```

Running the RESET test results in a low p-value, indicating that there might be a better way to specify our model with terms to the 2nd and 3rd power. In addition, some of the variables now appear as insignificant when we run the linear regression, so we will try removing TD and Int as another way of improving the model specification.

```
resettest(Rate ~ Cmp.trans + TD + TD. +
                 Int + Int.trans + ANY.A +
                 Cmp.trans**2 + TD**2 + TD.**2 +
                 Int**2 + Int.trans**2 + ANY.A**2 +
                 Cmp.trans**3 + TD**3 + TD.**3 +
                 Int**3 + Int.trans**3 + ANY.A**3,
                 power = 2:3,
                 type = "regressor",
                 data = removeoutliers)
```

```
##
##  RESET test
##
## data:  Rate ~ Cmp.trans + TD + TD. + Int + Int.trans + ANY.A + Cmp.trans^2 +     TD^2 + TD.^2 + Int^
## RESET = 22.67, df1 = 12, df2 = 334, p-value < 2.2e-16
```

```
resettest(Rate ~ Cmp.trans + TD. +
                 Int.trans + ANY.A,
                 power = 2:3,
                 type = "regressor",
                 data = removeoutliers)
```

```
##
##  RESET test
##
## data:  Rate ~ Cmp.trans + TD. + Int.trans + ANY.A
## RESET = 25.344, df1 = 8, df2 = 340, p-value < 2.2e-16
```

Adding higher power terms to our model did not prove to be an effective way to better the specification, but removing some of the less significant variables did improve the specification somewhat.

AIC/BIC Comparison

To have a second model to use in our AIC/BIC comparison, let us create a new model by removing the same terms as we did in the RESET test above. When we narrowed down our data set and ran the new regression, TD and Int seemed to be less significant than the other variables.

```
new_model <- lm(Rate ~ ANY.A + Cmp.trans +
                       TD. + Int.trans, data=removeoutliers)

cat('AIC_outlier_model:', AIC(outlier_model_test_b), '\t AIC_new_model:', AIC(new_model, k = 2))
```

```
## AIC_outlier_model: 1226.983   AIC_new_model: 1229.497
```

```
cat('\n')
```

```
cat('BIC_outlier_model:', BIC(outlier_model_test_b), '\t BIC_new_model:', AIC(new_model, k = 2))
```

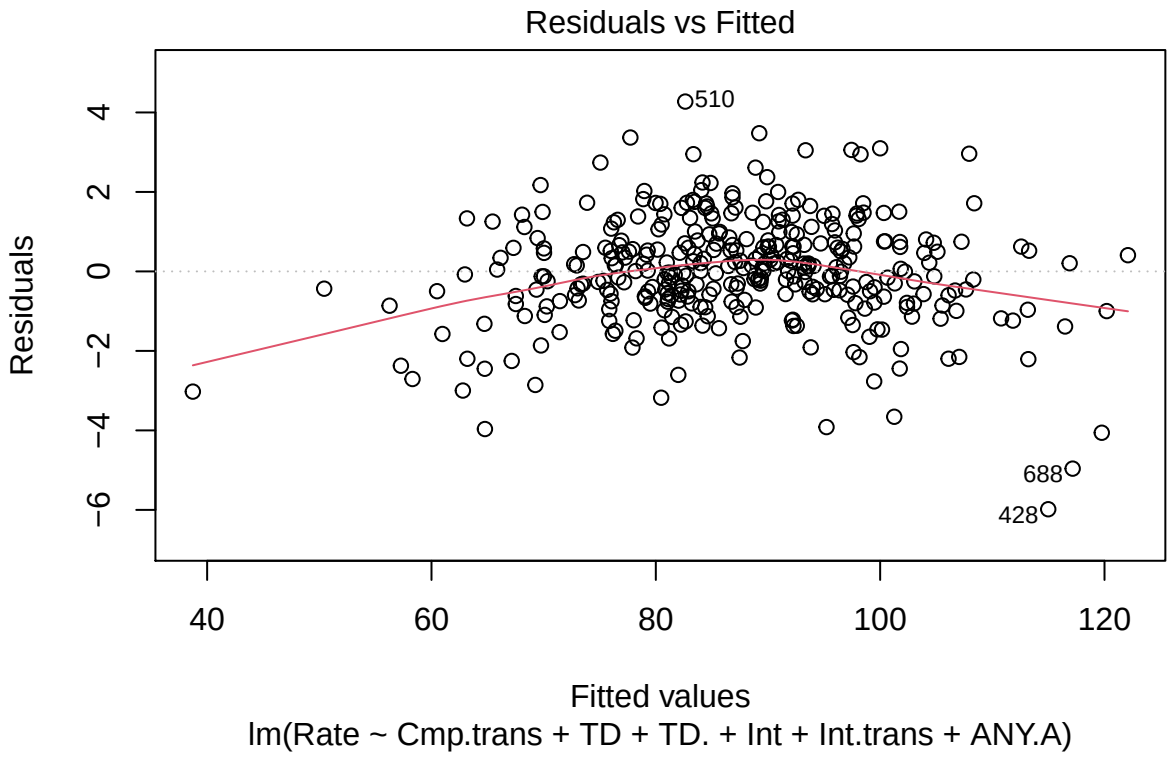
```
## BIC_outlier_model: 1257.915   BIC_new_model: 1229.497
```

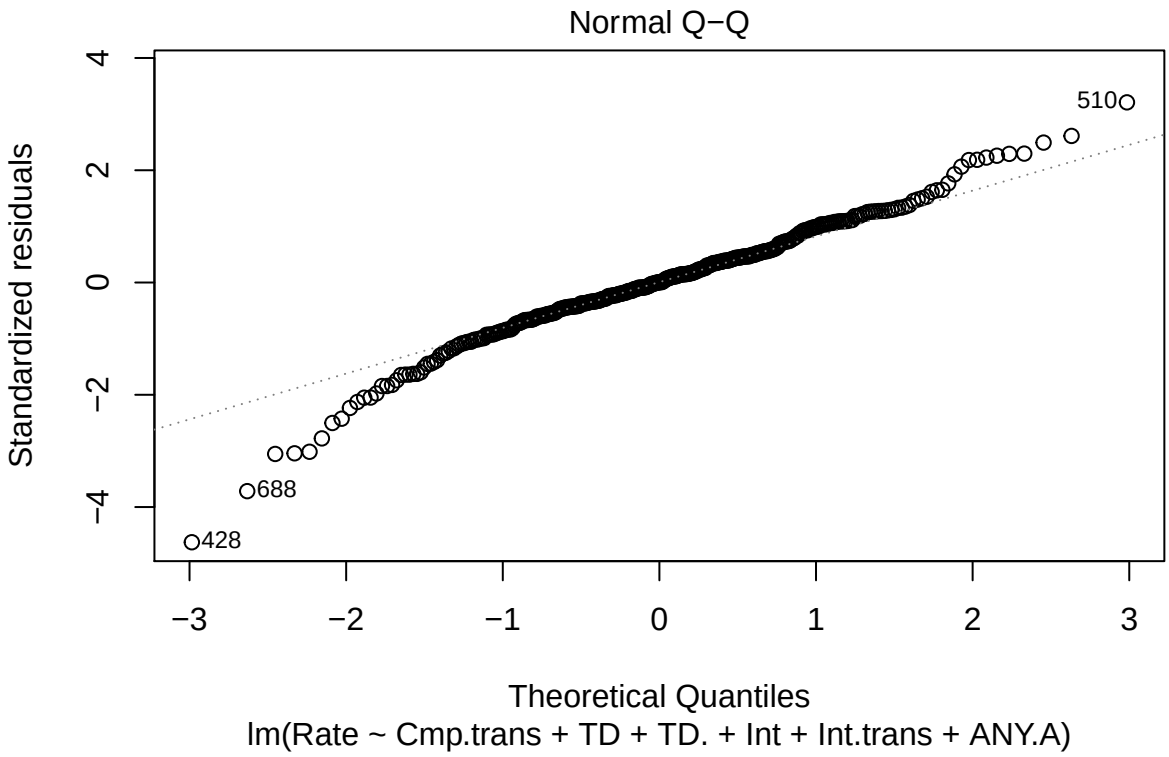
Using the Akaike and Bayesian Information Criterion, we conclude that our original model was better, since both the AIC and BIC are lower for the model in which we do not omit the variables TD and Int.

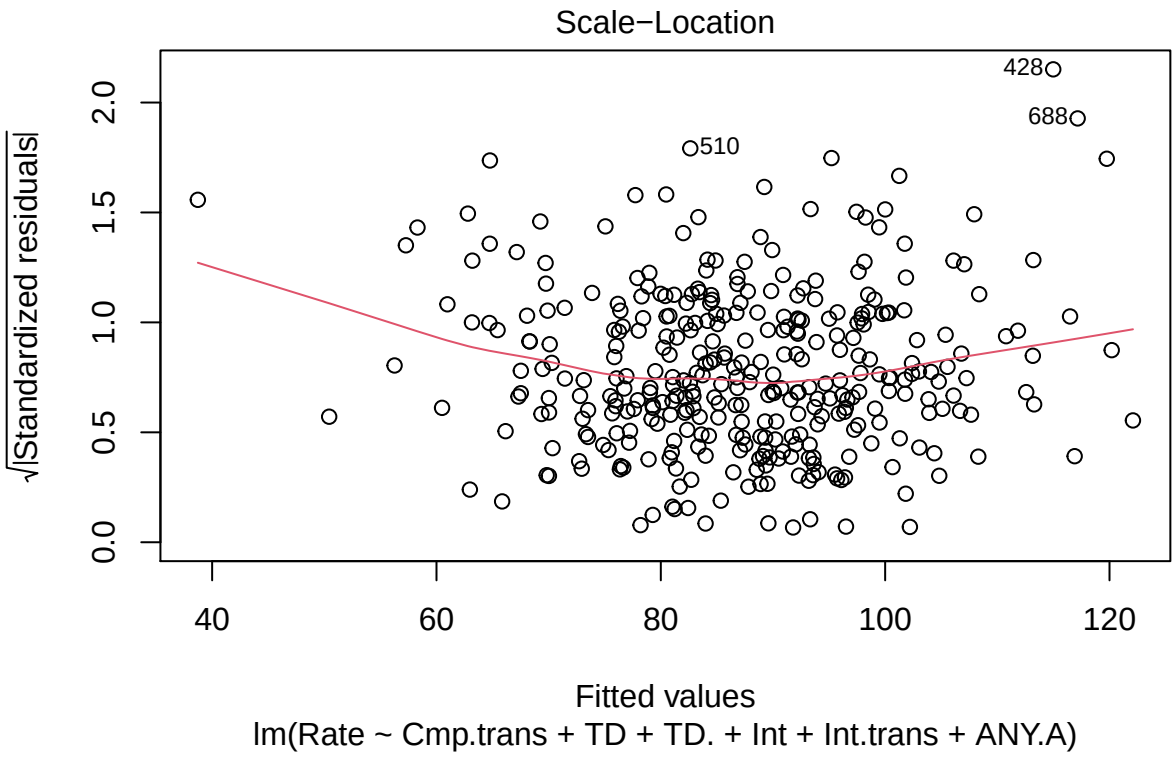
Checking Residual Plots

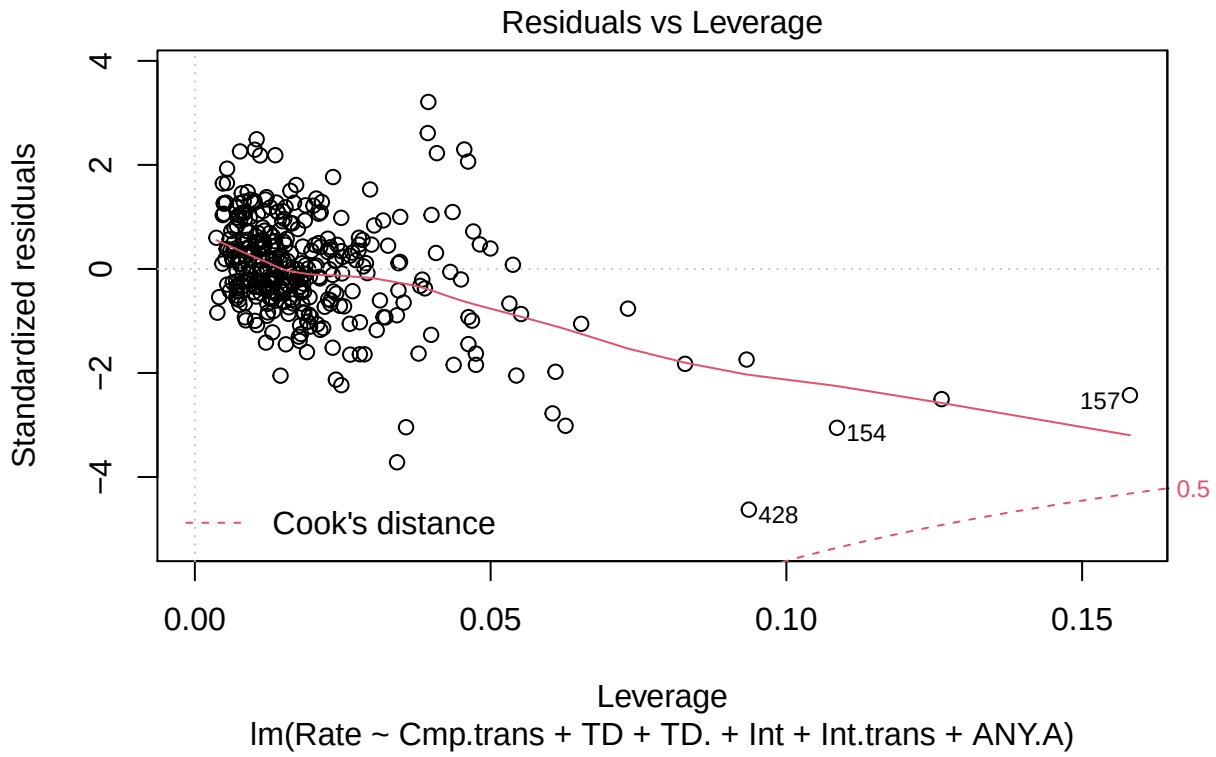
Since we decided to use our original model after the AIC/BIC comparision, let us examine the residual plots and Cook's distance plots for this model:

```
plot(outlier_model_test_b)
```




Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(Rate ~ Cmp.trans + TD + TD. + Int + Int.trans + ANY.A)
510
688
428
−4
−2
0
2
4
−3
−2
−1
0
1
2
3


Scale−Location
√|Standardized residuals|
0.0
0.5
1.0
1.5
2.0
428
688
510
40
60
80
100
120
Fitted values
lm(Rate ~ Cmp.trans + TD + TD. + Int + Int.trans + ANY.A)



The residuals plot shows that errors are scattered across the board; the model best predicts Passer Rating when the rating is around 85-95, but the residuals go up to 5 and down to -10 across many of the fitted values. The Cook's distance plot shows a few outliers; however, upon examination of these data points, we think they should not be removed based on the fact that they are players with a significant amount of games. Removing them would lead to overspecification of the model, where we would end up only using data points that fit our model well to fit the model itself.

Bootstrapping

```
library(boot)
```

```
##
## Attaching package: 'boot'
```

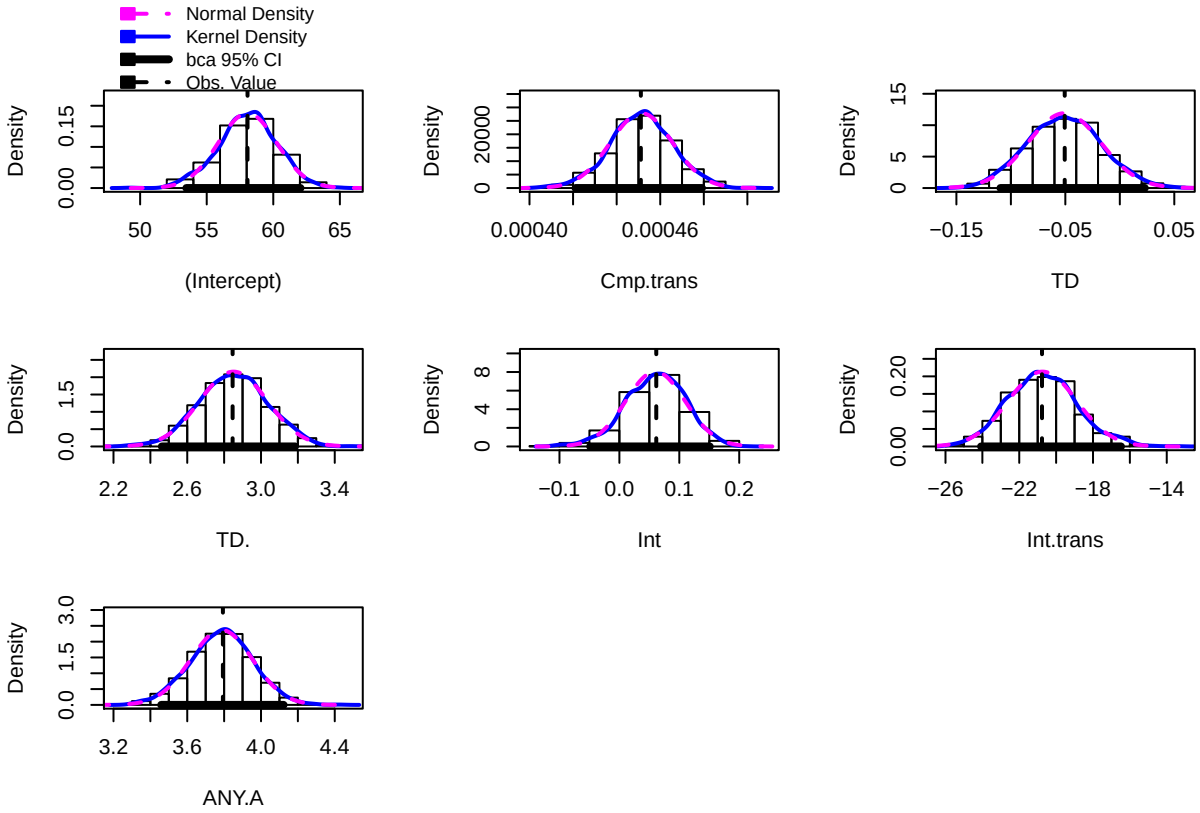
```
## The following object is masked from 'package:car':
##
##     logit
```

```
boot_model <- Boot(outlier_model_test_b, R=999)
summary(boot_model)
```

```
##
## Number of bootstrap replications R = 999
```

```
##                 original    bootBias     bootSE     bootMed
## (Intercept)  58.0651054 -1.4413e-03 2.1932e+00  5.8107e+01
## Cmp.trans     0.0004511  2.4115e-07 1.4322e-05  4.5161e-04
## TD           -0.0505849 -2.4277e-04 3.3502e-02 -5.1159e-02
## TD.           2.8461091  5.0969e-03 1.8461e-01  2.8491e+00
## Int           0.0615636 -1.4771e-03 4.9609e-02  6.1951e-02
## Int.trans   -20.7664910  3.0096e-02 1.8686e+00 -2.0826e+01
## ANY.A         3.7934562 -7.3803e-03 1.6861e-01  3.7903e+00
```

```
hist(boot_model)
```



Looking at the bootstrap results, it appears our model holds up well to resampling and regression on the newly created sample. None of the histograms seem to differ at all from what we have seen above.

Cross-Validation

```
library(caret)
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'lattice'
```

```
## The following object is masked from 'package:boot':
##
##     melanoma

## Loading required package: ggplot2
```

```
train.control <- trainControl(method="cv", number=5, savePredictions = TRUE, returnResamp = 'all')
cvmodel <- train(Rate ~ Cmp + TD + TD. +
                 Int + Int. + ANY.A, data=removeoutliers,                  method = "lm", trControl = t
print(cvmodel)
```

```
## Linear Regression
##
## 353 samples
##   6 predictor
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 283, 282, 284, 282, 281
## Resampling results:
##
##   RMSE     Rsquared   MAE
##   2.80042  0.9513415  2.045422
##
## Tuning parameter 'intercept' was held constant at a value of TRUE
```

In a 5-Fold Cross Validation test, it appears that the model still performs well in this version of a resampling regression. The R-squared value of 0.95 is very close to 0.97 achieved in the original regression.

Training and Testing

```
n = nrow(removeoutliers)
train_index = sample(n, floor(0.8 * n))
train_data1 = removeoutliers[train_index, ]
test_data1 = removeoutliers[-train_index, ]

predictions <- predict(cvmodel,test_data1)

RMSE = RMSE(predictions, test_data1$Rate)
MAE = MAE(predictions, test_data1$Rate)
R2 = R2(predictions, test_data1$Rate)
cat('RMSE:', RMSE)
```

```
## RMSE: 2.987792
```

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

```
cat('\t MAE:', MAE)
```

```
##   MAE: 2.11401
```

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

```
cat('\t R2:', R2)
```

```
##   R2: 0.9462736
```

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

```
avg_err_rate = RMSE(predictions, test_data1$Rate)/mean(test_data1$Rate)
cat('\t Average Error Rate:', avg_err_rate)
```

```
##   Average Error Rate: 0.03438308
```

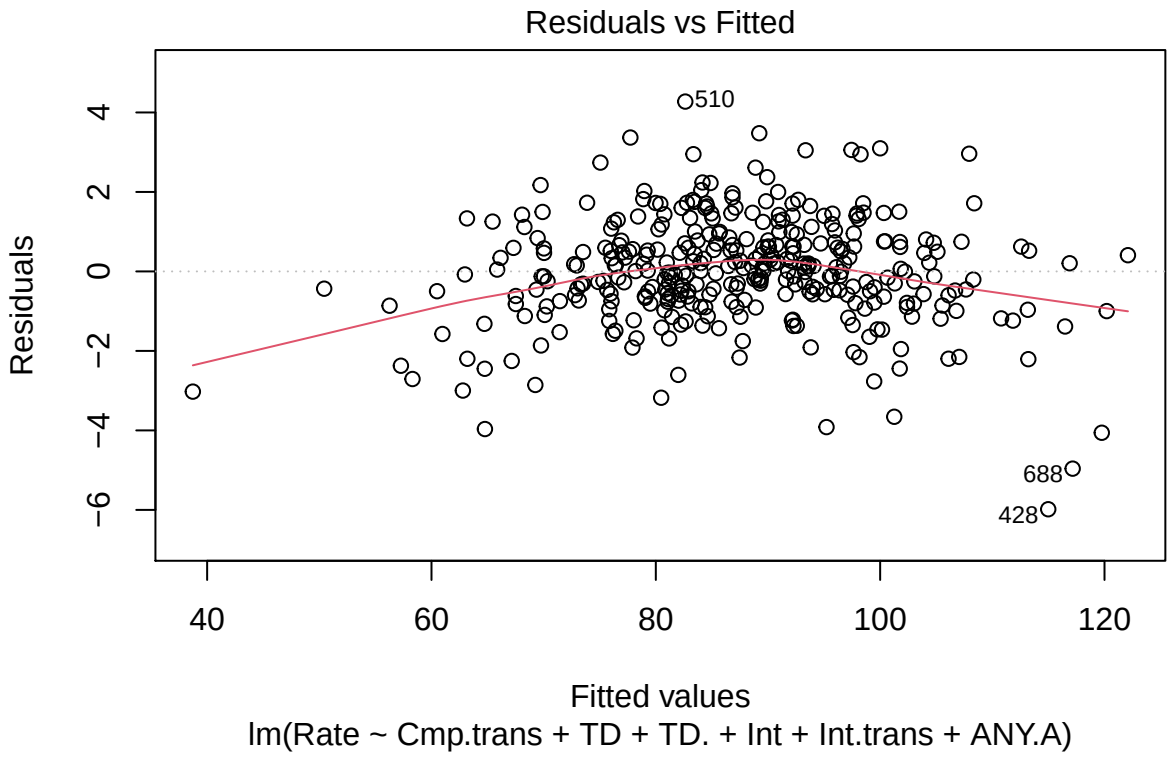
When we split our data into testing and training sets, we see similar results as in the cross validation test. The R-squared values are the same for both model evaluation methods. The RMSE is close to what it was in the cross validation, and the MAE is as well.

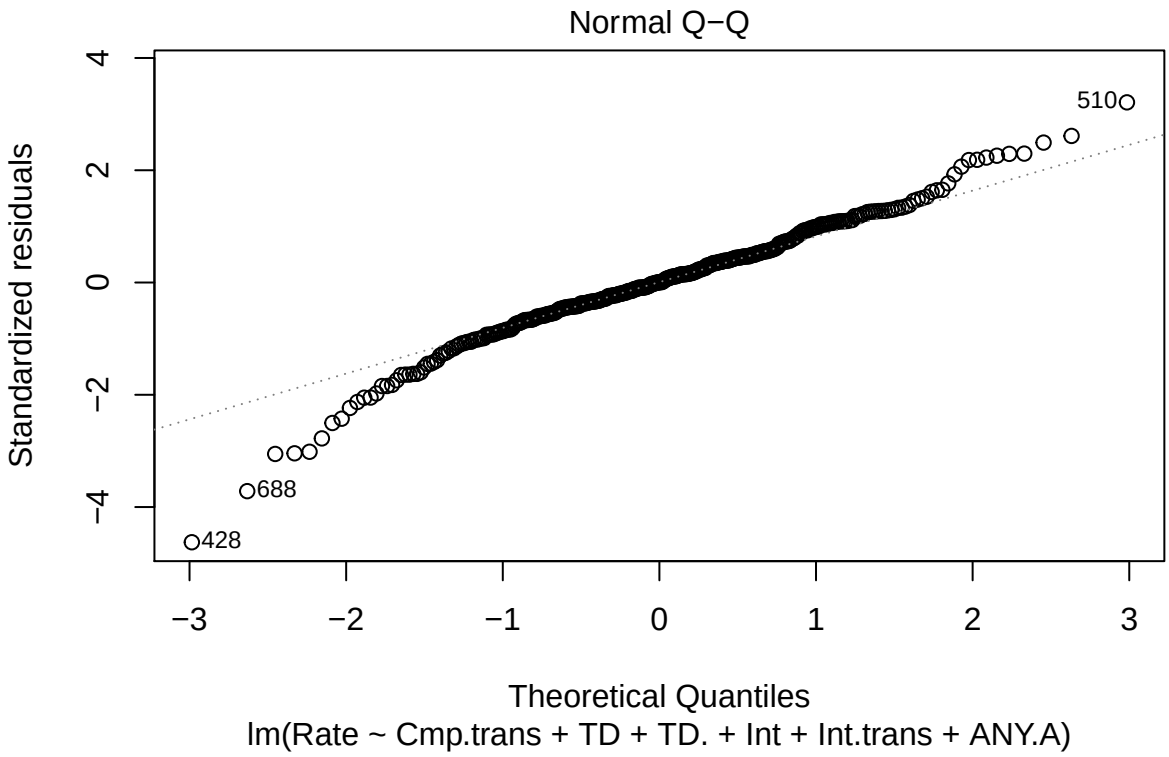
Model Interpretation

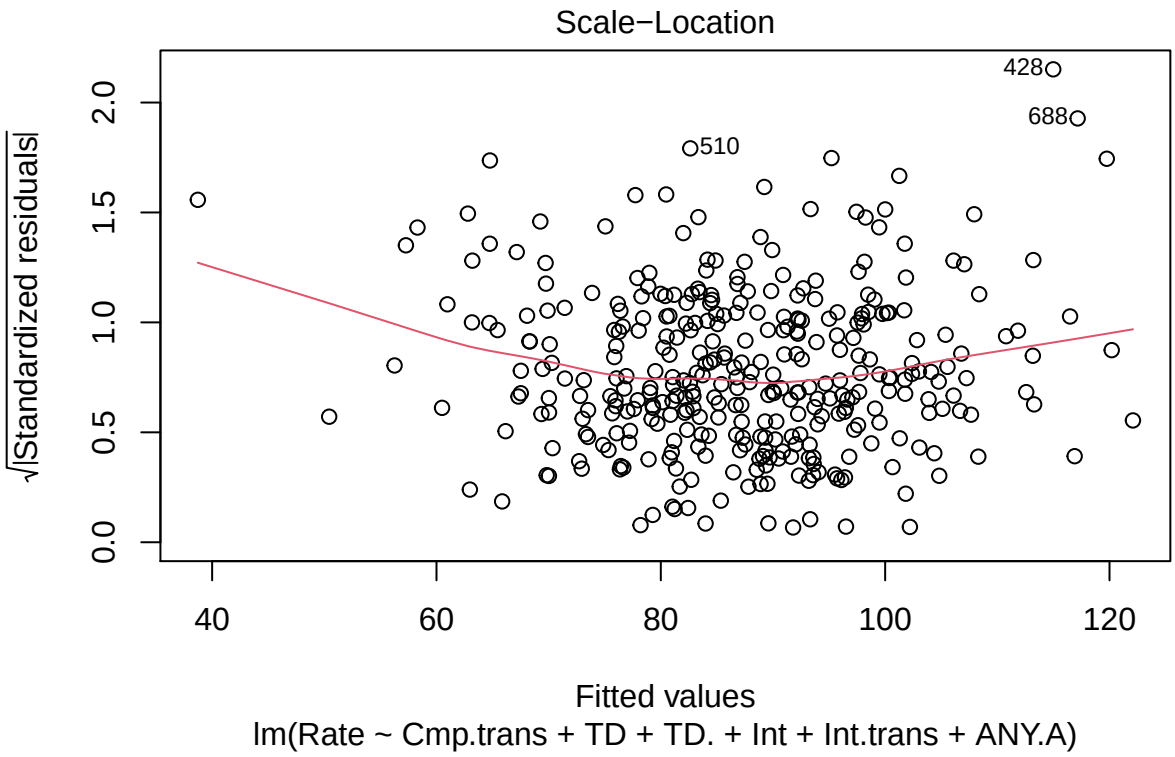
```
final_model <- outlier_model_test_b
summary(final_model)
```

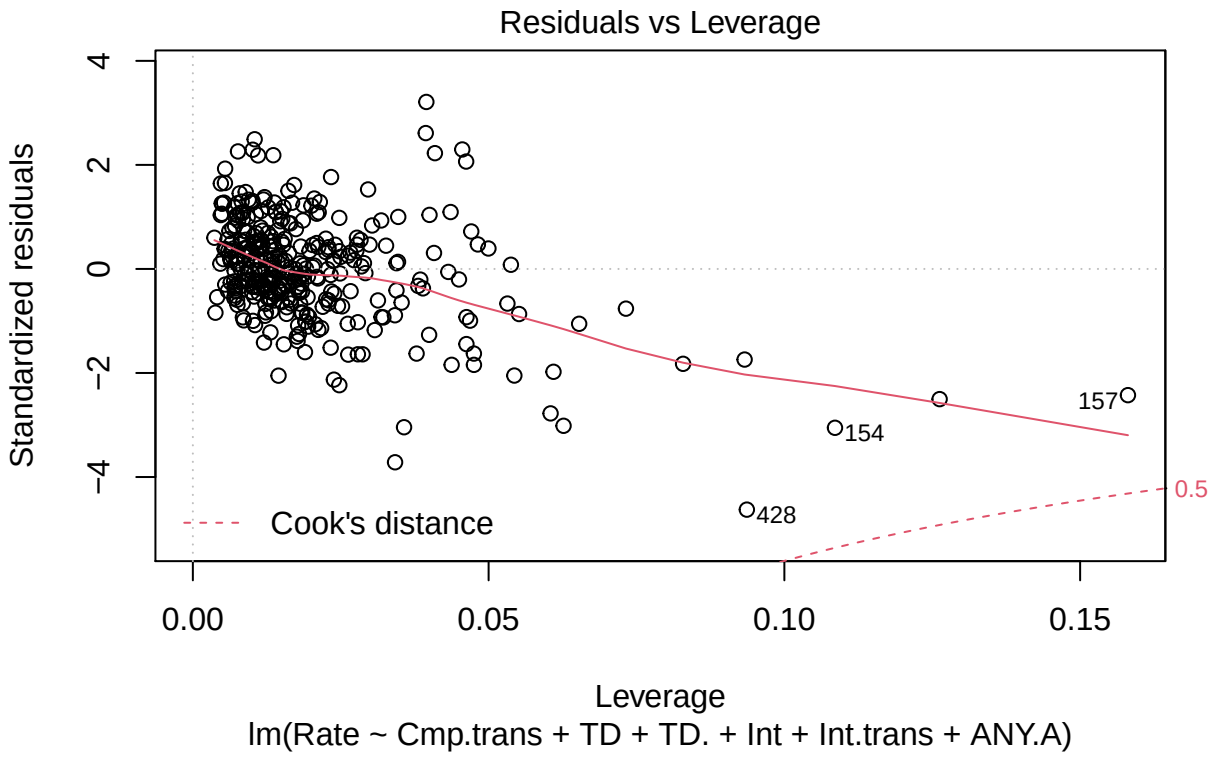
```
##
## Call:
## lm(formula = Rate ~ Cmp.trans + TD + TD. + Int + Int.trans +
##     ANY.A, data = removeoutliers)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.9838 -0.7309  0.0081  0.7462  4.2735
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  5.807e+01  1.379e+00  42.104   <2e-16 ***
## Cmp.trans    4.511e-04  1.132e-05  39.859   <2e-16 ***
## TD          -5.058e-02  1.995e-02  -2.535   0.0117 *
## TD.          2.846e+00  1.185e-01  24.025   <2e-16 ***
## Int          6.156e-02  3.076e-02   2.002   0.0461 *
## Int.trans   -2.077e+01  9.784e-01 -21.225   <2e-16 ***
## ANY.A        3.793e+00  1.364e-01  27.818   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.358 on 346 degrees of freedom
## Multiple R-squared:  0.9886, Adjusted R-squared:  0.9884
## F-statistic:  5000 on 6 and 346 DF,  p-value: < 2.2e-16
```

```
plot(final_model)
```




Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(Rate ~ Cmp.trans + TD + TD. + Int + Int.trans + ANY.A)
510
688
428
−4
−2
0
2
4
−3
−2
−1
0
1
2
3


Scale−Location
√|Standardized residuals|
0.0
0.5
1.0
1.5
2.0
40
60
80
100
120
428
688
510
Fitted values
lm(Rate ~ Cmp.trans + TD + TD. + Int + Int.trans + ANY.A)



In our final proposed model, all the selected variables are statistically significant to the highest criterion, and the residual standard error is 1.976. Given that Passer Rating is measured on a 0-158.3, an error of around 2 units is not so bad. The adjusted R-squared value of 0.9755 means that our model predicts about 97.5% of the variation in the Passer Rating data.

The model estimates that every additional touchdown scored by a quarterback increases Passer Rating by 0.15 units over a season, while increasing the ratio of Touchdowns to Attempts by 1% increases Passer Rating by 1.84 units. An interception takes away 0.35 units from a player's Passer Rating. This is a very interesting result because it perfectly reflects the actual formula that the NFL uses to measure Passer Rating. Touchdowns and Interceptions are naturally part of the calculation, but interceptions are weighted more, and negatively, while touchdowns garner a smaller, but positive score. It is nice to see the same trend in the value of our coefficients, where touchdowns contribute positively, but not as negatively as interceptions do. The transformed variables are somewhat hard to interpret given their respective power shifts. Finally, a player increasing their Adjusted Net Yards per Attempt provides the biggest swing to Passer Rating; for every yard added to the statistic, Passer Rating increases by a whole 4.75 units.