Mini Project 2

Suliah Apatira & Jasmine Harris

12/3/2022

##1 Exploratory Data Analysis In this project, we downloaded the song dataset and assigned it as the variable 'song'.

```
#Bringing in and visualizing song data
song <- read.csv("C:\\Users\\apati\\OneDrive\\Documents\\Michigan State\\STT 810\\song_data.csv")

#establishing each variable
song_popularity <- song$song_popularity
acousticness <- song$acousticness
loudness <- song$loudness
instrumentalness <- song$instrumentalness
danceability <- song$danceability
key <- song$key
audio_valence <- song$audio_valence
liveness <- song$liveness
time_signature <- song$time_signature
tempo <- song$tempo
speechiness <- song$speechiness
energy <- song$energy
audio_mode <- song$audio_mode
```

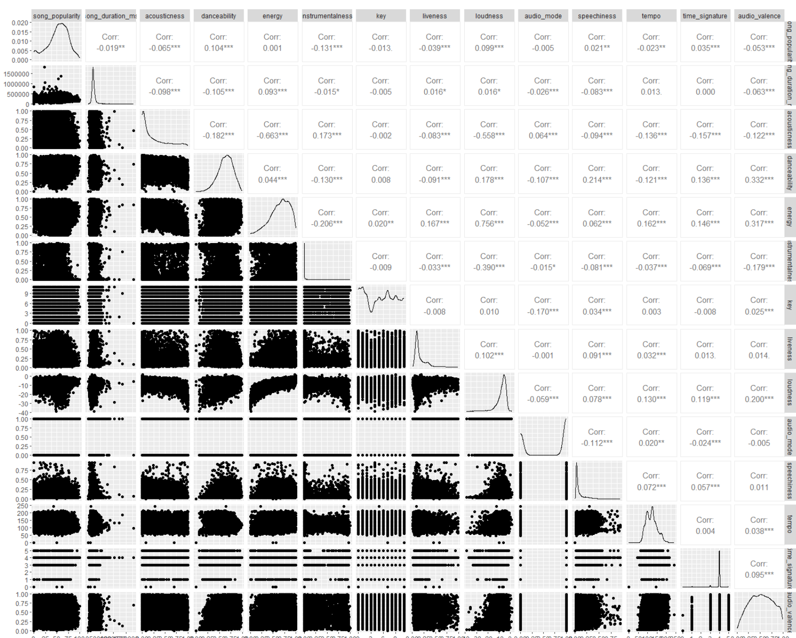
Before exploring the data, we first used the str() command to find out which variables were numeric or categorical.

```
str(song)

## 'data.frame':    18835 obs. of  15 variables:
##  $ song_name       : chr  "Boulevard of Broken Dreams" "In The End" "Seven Nation Army" "By The Way" ...
##  $ song_popularity : int  73 66 76 74 56 80 81 76 80 81 ...
##  $ song_duration_ms: int  262333 216933 231733 216933 223826 235893 199893 213800 222586 203346 ...
##  $ acousticness    : num  0.00552 0.0103 0.00817 0.0264 0.000954 0.00895 0.000504 0.00148 0.00108 0.00172 ...
##  $ danceability    : num  0.496 0.542 0.737 0.451 0.447 0.316 0.581 0.613 0.33 0.542 ...
##  $ energy          : num  0.682 0.853 0.463 0.97 0.766 0.945 0.887 0.953 0.936 0.905 ...
##  $ instrumentalness: num  2.94e-05 0.00 4.47e-01 3.55e-03 0.00 1.85e-06 1.11e-03 5.82e-04 0.00 1.04e-02 ...
##  $ key             : int  8 3 0 0 10 4 4 2 1 9 ...
```

```
##  $ liveness        : num  0.0589 0.108 0.255 0.102 0.113 0.396 0.268 0.152
0.0926 0.136 ...
##  $ loudness        : num  -4.09 -6.41 -7.83 -4.94 -5.07 ...
##  $ audio_mode      : int  1 0 1 1 1 0 0 1 1 1 ...
##  $ speechiness     : num  0.0294 0.0498 0.0792 0.107 0.0313 0.124 0.0624 0
.0855 0.0917 0.054 ...
##  $ tempo           : num  167 105 124 122 172 ...
##  $ time_signature  : int  4 4 4 4 4 4 4 4 4 4 ...
##  $ audio_valence   : num  0.474 0.37 0.324 0.198 0.574 0.32 0.724 0.537 0.
234 0.374 ...
```

To first explore the data set we created a ggpairs plot, excluding the song name variable because it is an unusable categorical variable. Then, we printed a ggpairs plot and a correlation matrix. Since the ggpairs plot and the correlation matrix depict the same information, then we will just show the ggpairs plot.

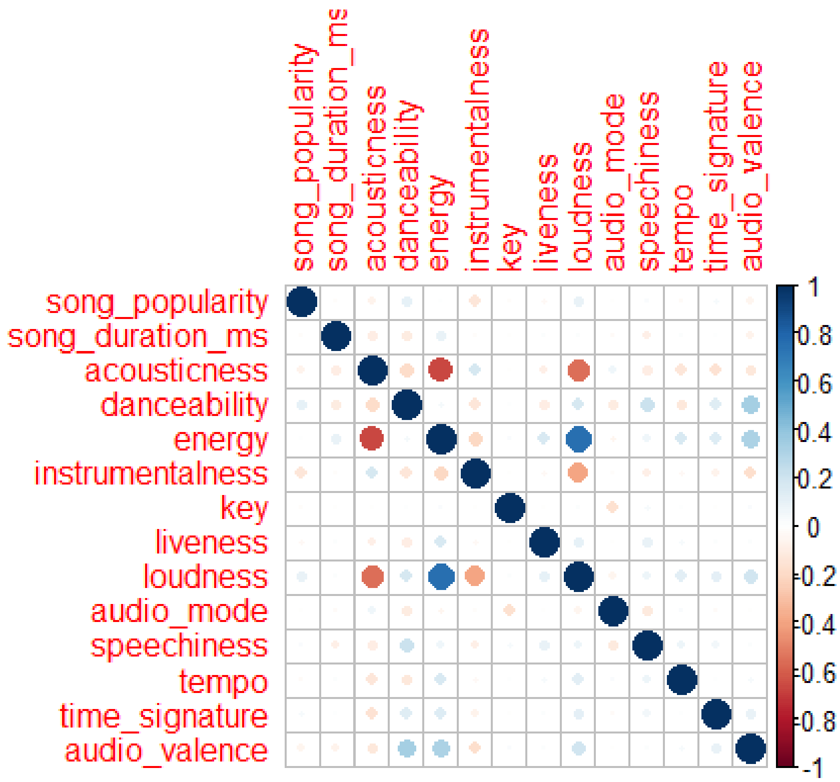
```
library(dplyr)

##

library(ggplot2)

ggpairs(song[2:15])
```



To further visualize the song data we created a corrplot of the correlation matrix:

```
cor(song[2:15])

cormat <- cor(song[2:15])
library("corrplot")
```

```
## corrplot 0.92 loaded

corrplot(cormat,method = "circle")
```



#Simple Regression/Baseline After analyzing the correlation matrix for the song data, we created two simple linear regression models using the two variables with the highest correlation to song popularity. Those two variables were instrumentalness with a correlation of 0.131, and danceability with a correlation of 0.104. Additionally, we did the regression models using a training set and tested them on the test dataset with a 70/30 split. These two simple regression models will be used as a baseline for our final model.

```
set.seed(1)
summary(lm(song$song_popularity ~ song$instrumentalness, data = song))

##
## Call:
## lm(formula = song$song_popularity ~ song$instrumentalness, data = song)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -54.001 -12.581   2.999  15.717  46.059
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)            54.0014     0.1678  321.89   <2e-16 ***
## song$instrumentalness -12.9410     0.7142  -18.12   <2e-16 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 21.72 on 18833 degrees of freedom
## Multiple R-squared:  0.01714,    Adjusted R-squared:  0.01708
## F-statistic: 328.4 on 1 and 18833 DF,  p-value: < 2.2e-16
```

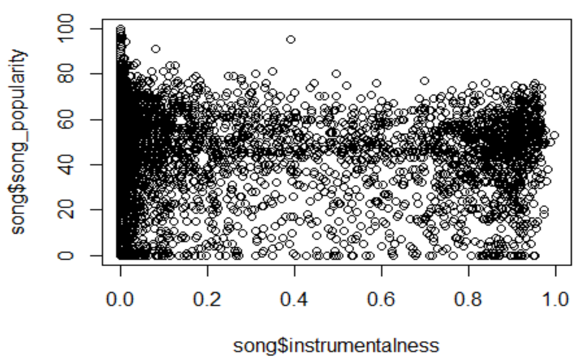
```
split_pct <- 0.7
n <- length(song$song_popularity)*split_pct # train size
row_samp <- sample(1:length(song$song_popularity), n, replace = FALSE)
train <- song[row_samp,]
test <- song[-row_samp,]
song_train_mod <- lm(data = train, song_popularity ~ instrumentalness )
test_pred <- predict(song_train_mod,test)
test_error <- test$song_popularity - test_pred
rmse_train <- sqrt(mean(song_train_mod$residuals^2))
rmse_test <- sqrt(mean(test_error^2))
rmse_train
```

```
## [1] 21.7257
```

```
rmse_test
```

```
## [1] 21.69755
```

```
plot(song$instrumentalness, song$song_popularity)
```




```
set.seed(2)
summary(lm(song$song_popularity ~ song$danceability, data = song))
```

```
##
## Call:
## lm(formula = song$song_popularity ~ song$danceability, data = song)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -57.899 -12.568   2.815  15.746  46.226
##
```

```
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)         43.7596     0.6609   66.21   <2e-16 ***
## song$danceability  14.5770     1.0130   14.39   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 21.79 on 18833 degrees of freedom
## Multiple R-squared:  0.01088,    Adjusted R-squared:  0.01082
## F-statistic: 207.1 on 1 and 18833 DF,  p-value: < 2.2e-16
```

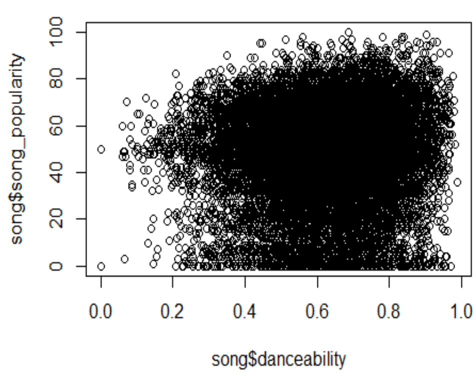
```
#TEST AND TRAIN
split_pct <- 0.7
n <- length(song$song_popularity)*split_pct # train size
row_samp <- sample(1:length(song$song_popularity), n, replace = FALSE)
train <- song[row_samp,]
test <- song[-row_samp,]
song_train_mod <- lm(data = train, song_popularity ~ instrumentalness )
test_pred <- predict(song_train_mod,test)
test_error <- test$song_popularity - test_pred
rmse_train <- sqrt(mean(song_train_mod$residuals^2))
rmse_test <- sqrt(mean(test_error^2))
rmse_train
```

```
## [1] 21.7406
```

```
rmse_test
```

```
## [1] 21.66045
```

```
plot(song$danceability, song$song_popularity)
```



#Some description of variable transformations attempted From the results above, we decided to create a variety of complex multiple regressions to build better models. First, in addition to Baseline 1, we decided to square the variable x.

```
summary(lm(song$song_popularity ~ (song$instrumentalness)^2, data = song))
```

```
##
## Call:
## lm(formula = song$song_popularity ~ (song$instrumentalness)^2,
##     data = song)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -54.001 -12.581   2.999  15.717  46.059
##
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)              54.0014     0.1678  321.89   <2e-16 ***
## song$instrumentalness -12.9410     0.7142  -18.12   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 21.72 on 18833 degrees of freedom
## Multiple R-squared:  0.01714,    Adjusted R-squared:  0.01708
## F-statistic: 328.4 on 1 and 18833 DF,  p-value: < 2.2e-16
```

There is no change between Baseline 1 and model 1b so, to each variable, we included the next variable with the highest correlation to song popularity. We created four linear models for each inclusion. The four models have the following operations: A. Addition B. Square of the sum C. Multiplication D. Square of the product We will consider these as Steps A-D. From the summary of the linear models, we looked at the residual standard error and the adjusted r-squared value. Adjusted r-squared values are between 0 and 1. Our goal is to see a decrease in the residual standard error and an increase in the adjusted r-squared value. A low residual standard error value and an adjusted r-squared value that is closer to 1 suggests a well-fit model. In addition, we looked at the test-training data set for each model. With the test-training data set, we look to see if the test and train values are close together by calculating the difference between the two. A close difference between the values also represents a well-fit model. Below are all the models tested.

**For report length purposes, we had to remove models with variable transformations and feature engineering of two through six variables. To see those variables, please refer to our .rmd file. **

#Fit Seven

7A

```
summary(lm(song_popularity ~ instrumentalness + danceability + loudness + aco
usticness + audio_valence + liveness + time_signature, data = song))
```

```
##
## Call:
## lm(formula = song_popularity ~ instrumentalness + danceability +
##     loudness + acousticness + audio_valence + liveness + time_signature,
##     data = song)
##
```

```
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -63.905 -12.450   3.029  15.745  45.187 
## 
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)    
## (Intercept)       48.74689    2.23261  21.834  < 2e-16 ***
## instrumentalness -11.36527    0.77308 -14.701  < 2e-16 ***
## danceability      15.85303    1.08360  14.630  < 2e-16 ***
## loudness           0.32172    0.05319   6.048 1.49e-09 ***
## acousticness      -0.68109    0.66106  -1.030   0.3029    
## audio_valence    -11.13701    0.68925 -16.158  < 2e-16 ***
## liveness          -5.69390    1.09955  -5.178 2.26e-07 ***
## time_signature     1.15331    0.53420   2.159   0.0309 *  
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 21.45 on 18827 degrees of freedom
## Multiple R-squared:  0.04164,    Adjusted R-squared:  0.04128 
## F-statistic: 116.9 on 7 and 18827 DF,  p-value: < 2.2e-16
```

```
split_pct <- 0.7
n <- length(song$song_popularity)*split_pct
row_samp <- sample(1:length(song$song_popularity), n, replace = FALSE)
train <- song[row_samp,]
test <- song[-row_samp,]
song_train_mod <- lm(data = train, song_popularity ~ instrumentalness + dance
ability + loudness + acousticness + audio_valence + liveness + time_signature
)
rmse_train <- sqrt(mean(song_train_mod$residuals^2))
rmse_test <- sqrt(mean(test_error^2))
rmse_train
```

```
## [1] 21.49046
```

```
rmse_test
```

```
## [1] 21.10615
```

```
summary(lm(song_popularity ~ (instrumentalness + danceability + loudness + ac
ousticness + audio_valence + liveness + time_signature)^2, data = song))
```

```
## 
## Call:
## lm(formula = song_popularity ~ (instrumentalness + danceability + 
##     loudness + acousticness + audio_valence + liveness + time_signature)^2
, 
##     data = song)
```

```
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -67.236 -11.930   3.102  15.322  50.498
##
## Coefficients:
##                                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)                        32.25217   10.00194   3.225 0.001264 **
## instrumentalness                  -31.63341    8.19327  -3.861 0.000113 ***
## danceability                       38.28064   14.33242   2.671 0.007571 **
## loudness                           -0.55856    0.63142  -0.885 0.376380
## acousticness                       24.08599    7.69217   3.131 0.001743 **
## audio_valence                      -5.03833   10.76771  -0.468 0.639853
## liveness                           -1.77006   15.80621  -0.112 0.910836
## time_signature                      1.48696    2.48440   0.599 0.549500

## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 21.14 on 18806 degrees of freedom
## Multiple R-squared:  0.07014,    Adjusted R-squared:  0.06876
## F-statistic: 50.66 on 28 and 18806 DF,  p-value: < 2.2e-16
```

```r
split_pct <- 0.7
n <- length(song$song_popularity)*split_pct
row_samp <- sample(1:length(song$song_popularity), n, replace = FALSE)
train <- song[row_samp,]
test <- song[-row_samp,]
song_train_mod <- lm(data = train, song_popularity ~ (instrumentalness + danc
eability + loudness + acousticness + audio_valence + liveness + time_signatur
e)^2)
rmse_train <- sqrt(mean(song_train_mod$residuals^2))
rmse_test <- sqrt(mean(test_error^2))
rmse_train
```

```
## [1] 21.20385
```

```r
rmse_test
```

```
## [1] 21.10615
```

7C

```r
set.seed(123)
summary(lm(song_popularity ~ instrumentalness * danceability * loudness * aco
usticness * audio_valence * liveness * time_signature, data = song))
```

```
##
## Call:
```

```
## lm(formula = song_popularity ~ instrumentalness * danceability *
##     loudness * acousticness * audio_valence * liveness * time_signature,
##     data = song)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -68.399 -11.776   2.956  15.259  49.072
##
## Coefficients:
##
Estimate
## (Intercept)
-2.671e+01
## instrumentalness
-1.464e+02
## danceability
8.433e+01
## loudness
-1.353e+01
## acousticness
1.569e+02
## audio_valence
1.120e+02
## liveness
2.843e+02
## time_signature
1.232e+01
##
Std. Error
## (Intercept)
1.107e+02
## instrumentalness
2.772e+02
## danceability
2.275e+02
## loudness
1.740e+01
## acousticness
1.922e+02
## audio_valence
3.028e+02
## liveness
4.800e+02
## time_signature
2.802e+01


##
Pr(>|t|)
## (Intercept)
```

```
0.8094
## instrumentalness
0.5973
## danceability
0.7109
## loudness
0.4370
## acousticness
0.4144
## audio_valence
0.7114
## liveness
0.5536
## time_signature
0.6601
##

## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 21.11 on 18707 degrees of freedom
## Multiple R-squared:  0.07798,	Adjusted R-squared:  0.07172
## F-statistic: 12.46 on 127 and 18707 DF,  p-value: < 2.2e-16
```

```
split_pct <- 0.7
n <- length(song$song_popularity)*split_pct # train size
row_samp <- sample(1:length(song$song_popularity), n, replace = FALSE)
train <- song[row_samp,]
test <- song[-row_samp,]
song_train_mod <- lm(data = train, song_popularity ~ instrumentalness * dance
ability * loudness * acousticness * audio_valence * liveness * time_signature
)
rmse_train <- sqrt(mean(song_train_mod$residuals^2))
rmse_test <- sqrt(mean(test_error^2))
rmse_train
```

```
## [1] 20.95328
```

```
rmse_test
```

```
## [1] 21.10615
```

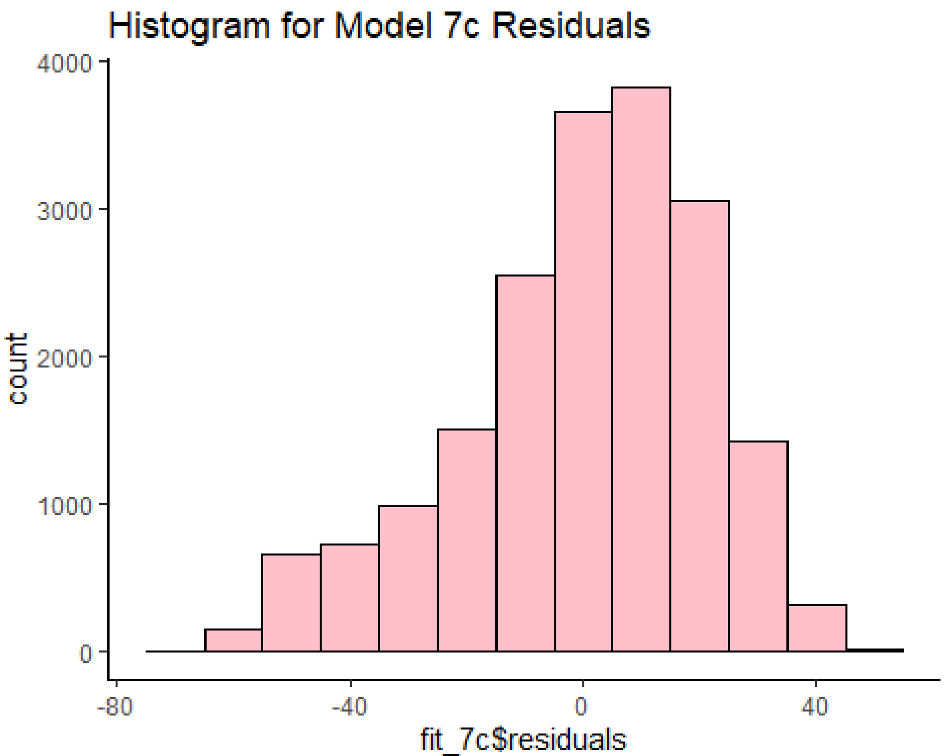
#Feature Engineering To get to our final model, we first arranged our variables from highest to lowest correlations with the song popularity category. This method of arranging from high to low made it easier for us to do feature selection and extraction of low-level features that were unsuitable/counterproductive for learning. We ultimately found that the seven variables with highest correlations created our best model. Those seven variables are instrumentalness, danceability, loudness, acousticness, audio valence, liveness, and time signature. Any further use of the variables following these seven resulted in a large discrepancy between the training data root mean square error and the tested root mean square error. For example, the root mean square error had a difference greater than 5 between the training root mean square error and tested root mean square error when the tempo variable was included. Whereas, using the seven variables previously mentioned, we found that the difference of root mean square errors is 0.392. This means that the model predicted the data accurately.

#Summary of Model Fit in Comparison to Baseline We used song_popularity ~ instrumentalness and song_popularity~danceability as the baseline models for this project. We chose to use instrumentalness and danceability for the simple regression baseline models because they had the highest correlations with song_popularity. Song_popularity~instrumentalness results in a residual standard error of 21.72 and an R-squared value of 0.01708. The difference between the root mean square error (RMSE) from the training data and to the tested data is 0.0282. Song_popularity~danceability results in a residual standard error of 21.79 and an R-squared value of 0.01082. The difference between the trained data and tested data was 0.1342. Our final model had a residual standard error 21.11 which is .61 lower than our first and best baseline model. The R-squared value is 0.07172 which also has significant improvement from the baseline model. The difference in root mean square error from the trained and tested model data is 0.427 which is not substantially higher than the baseline models.

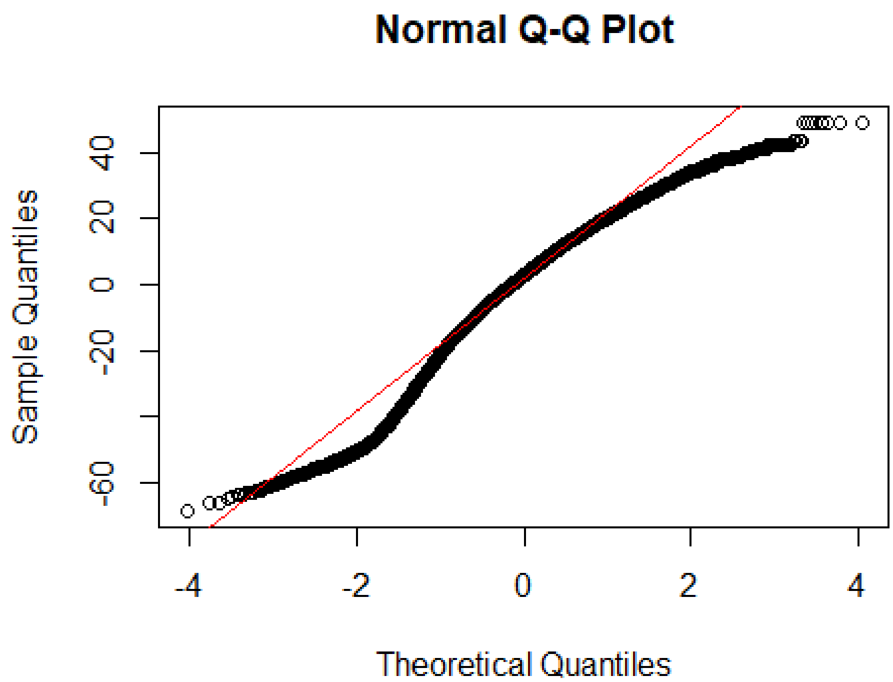
#Description of whether parameter signs make sense Our model found instrumentalness and loudness had negative parameter signs and danceability, acousticness, audio valence, and time signature had positive parameter signs as they relate to song popularity. Instrumentalness represents whether a track contains any spoken vocals. This sign makes sense because we expect a song to be more popular if it has actual words that listeners can potentially sing along with. Loudness having a negative sign does not make sense because pop, hip hop/rap, and electronic dance music are the most popular music genres right now in the United states. These music genres are known for being loud, so we do not believe loudness having a negative parameter makes sense. Next, it makes sense for danceability, audio valence, and time signature to have positive signs. Danceability refers to how suitable a track is for dancing based on factors such as tempo, rhythm, and beat strength. Audio valence refers to the happiness or cheerfulness of a song. Time signature indicates how many beats are in each measure of a piece of music. Having a higher time signature means the music is faster or more upbeat. It makes sense for these factors to be positive because popular music is typically cheerful with a nice rhythm or beat that is positive and danceable. Lastly, we do not think it makes sense for acousticness to be positivite once again based on the most popular genres of music. Additionally, we listened to the top 10 songs on the global charts and only one of the ten had an acoustic background.

#Analysis of residuals For the analysis of residuals, we plotted a histogram and a normal q-q plot. We analyzed both graphs to determine if the error terms are normally distributed or not. In terms of the histogram, it shows that the residuals are normally distributed around zero. In regards to the normal q-q plot, the plot of residuals is approximately linear which therefore supports the condition that the error terms (residuals) are normally distributed. Since the residuals are normally distributed, it suggests that our model fits the data fairly well.

```r
fit_7c <- lm(song_popularity ~ instrumentalness * danceability * loudness * a
cousticness * audio_valence * liveness * time_signature, data = song)
ggplot(data = song, aes(fit_7c$residuals))+geom_histogram(binwidth = 10, colo
r = "black", fill = "pink")+ theme(panel.background = element_rect(fill = "wh
ite"),axis.line.x = element_line(),axis.line.y = element_line())+ggtitle("His
togram for Model 7c Residuals")
```



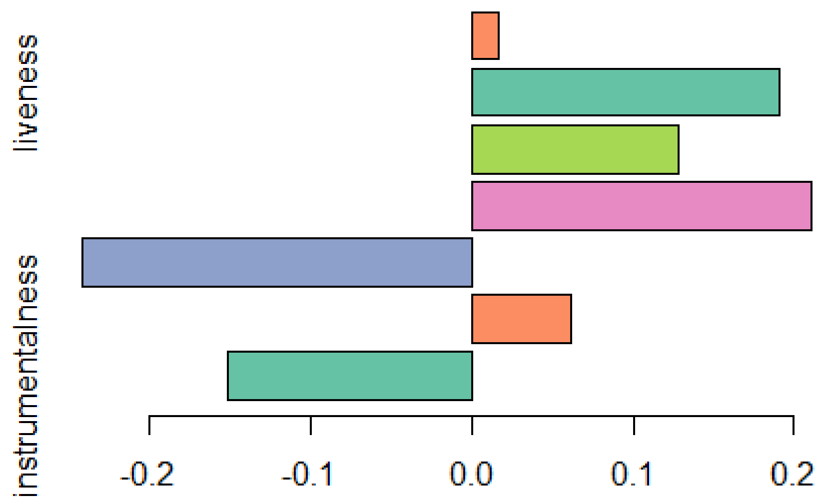

```r
res <- resid(fit_7c)
qqnorm(res)
qqline(res,col="red")
```



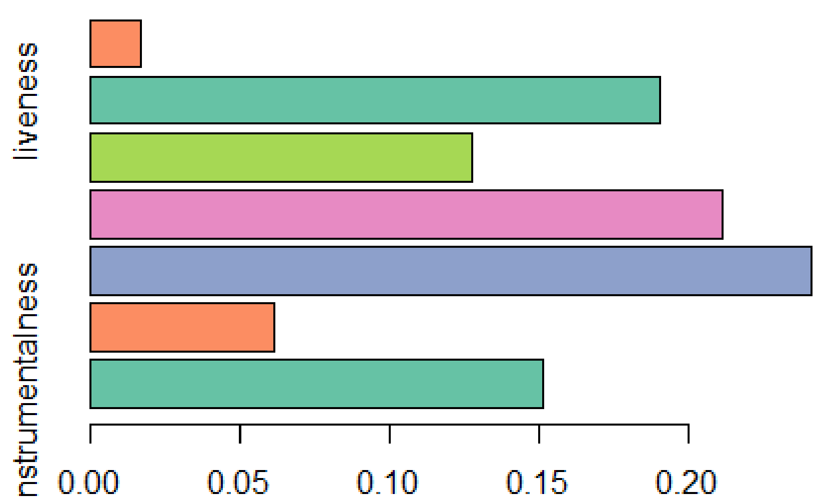
#Sensitivity Analysis Next, we created a sensitivity analysis to visualize how different input variables compare in the song popularity model. As seen in our sensitivity analysis tornado diagram, loudness had the largest impact on the model followed by acousticness, liveness, and instrumentalness. Loudness and instrumentalness are moving in the negative direction, while the other 5 variables are moving in the positive direction. This data agrees with our parameter signs data analysis.

```
song_mod <- lm(data = song, song_popularity ~ instrumentalness * danceability
* loudness * acousticness * audio_valence * liveness * time_signature)
sum_std <- sd(song$instrumentalness)*abs(song_mod$coefficients[2]) +  sd(song
$danceability)*abs(song_mod$coefficients[3]) +  sd(song$loudness)*abs(song_mo
d$coefficients[4]) +   sd(song$acousticness)*abs(song_mod$coefficients[5]) +
sd(song$audio_valence)*abs(song_mod$coefficients[6]) +     sd(song$liveness)*
abs(song_mod$coefficients[7]) +      sd(song$time_signature)*abs(song_mod$coe
fficients[8])


instru_sens <- sd(song$instrumentalness)*song_mod$coefficients[2]/sum_std
dance_sens <- sd(song$danceability)*song_mod$coefficients[3]/sum_std
loud_sens <- sd(song$loudness)*song_mod$coefficients[4]/sum_std
acoust_sens <- sd(song$acousticness)*song_mod$coefficients[5]/sum_std
audiov_sens <- sd(song$audio_valence)*song_mod$coefficients[6]/sum_std
live_sens <- sd(song$liveness)*song_mod$coefficients[7]/sum_std
time_sens <- sd(song$time_signature)*song_mod$coefficients[8]/sum_std
```

```
instru_sens

## instrumentalness
##       -0.1510914

dance_sens

## danceability
##   0.06153836

loud_sens

##   loudness
## -0.2410695

acoust_sens

## acousticness
##    0.2109328

audiov_sens

## audio_valence
##     0.1276232

live_sens

##  liveness
## 0.1906132

time_sens

## time_signature
##     0.01713158

library(RColorBrewer)
coul <- brewer.pal(5, "Set2")

barplot(c(instru_sens, dance_sens, loud_sens, acoust_sens, audiov_sens, live_
sens, time_sens), horiz = TRUE, col=coul)
```

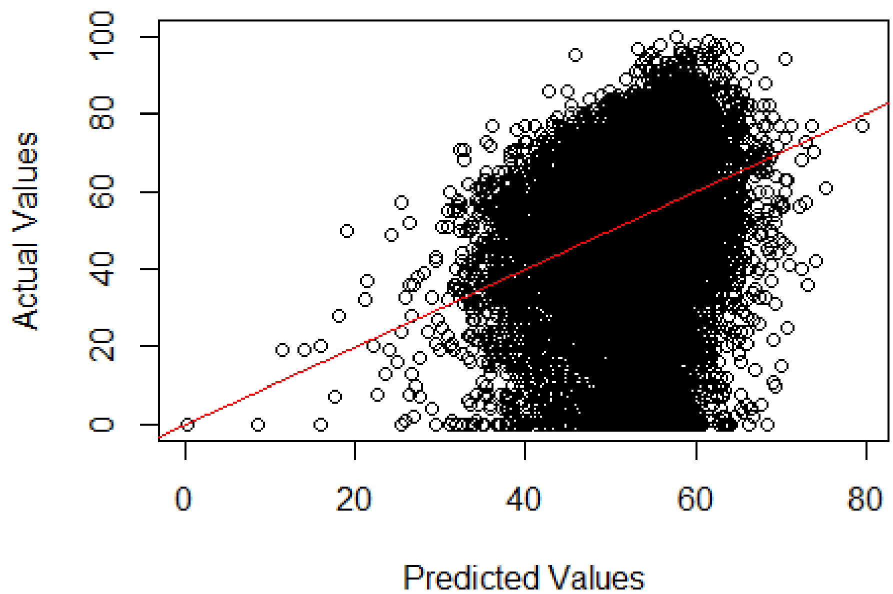



```
barplot(c(abs(instru_sens), abs(dance_sens), abs(loud_sens), abs(acoust_sens)
, abs(audiov_sens), abs(live_sens), abs(time_sens)), horiz = TRUE,col=coul)
```



#Graph of fit (predicted vs actual) As recalled, song_popularity is the target variable so we created a table that compares the first six values of the variable's actual value versus predicted. Additionally, there is a graph to represent these values too. From these results, we have analyzed that there is a significant discrepancy between the predicted and actual values, which suggests that our model is not the best-fitted model.

```
plot(predict(fit_7c),song$song_popularity,xlab="Predicted Values",ylab="Actua
l Values")
abline(0,1,col="red")
```




```
data <- data.frame(actual=song$song_popularity,predicted=predict(fit_7c))
head(data)

##   actual predicted
## 1     73  54.14935
## 2     66  54.72892
## 3     76  46.57441
## 4     74  53.59951
## 5     56  51.92172
## 6     80  45.39410
```

In conclusion, in knowing all of the results presented above and in continuation of this project, we would do more feature engineering and build additional models that will test more variables in several ways. Our goal in doing so is to produce a better well-fitted model to predict song_popularity.