Final Project for S&DS 230

Rishi Shah

May 6, 2023

Introduction

Spotify is a Swedish-based audio streaming platform that has attained worldwide acclaim and has revolutionized the way millions of people listen to music. With over 515 million users and over 100 millions tracks, Spotify is the most popular and robust audio streaming platform on the planet. As the go-to destination for music, a major task for Spotify is curating the best playlists and song recommendations that match a user's listening patterns. As such, Spotify analyzes massive amounts of data, even calculating how "danceable" or energetic a track is as part of its large pool of metrics used to rank tracks. My goal for this project is to analyze relationships between variables monitored by Spotify as rankings for its tracks, and determine if combinations of these variables are good at predicting metrics such as a track's popularity. I imagine such analysis can be useful for Spotify to deliver the best user experience.

Data Overview and Cleaning

This dataset contains 23 variables that provide information regarding the song and that quantify various characteristics of the track. I am choosing to include and analyze only 19 of the variables as those are the only ones relevant to my analysis, which have been described below:

I will be focusing our analysis on the data specifically from 2020, as this allows me to use a more narrow and specific set of datapoints, ensuring that my data is not too unwieldy. However, just for the permutation test I do a year-wise comparison of differences in the mean track popularity from 2018 to 2019, as I was curious to see if this varies over any randomly given 2 years.

Continuous Variables:

- `acousticness` : the relative metric of the track being acoustic, (Ranges from 0 to 1)
- `danceability` : the relative measurement of the track being danceable, (Ranges from 0 to 1)
- `energy` : the energy of the track, (Ranges from 0 to 1)
- `duration_ms` : length of the track in milliseconds (ms)
- `instrumentalness` :, the relative ratio of the track being instrumental, (Ranges from 0 to 1)
- `valence` : the positiveness of the track, (Ranges from 0 to 1)
- `track_popularity` : Song Popularity (0 to 100, higher is better)
- `tempo` : tempo of the track in Beat Per Minute (BPM), (Float typically ranging from 50 to 150)
- `liveness` : relative duration of the track sounding as a live performance, (Ranges from 0 to 1)
- `loudness` : relative loudness of the track in decibel (dB), (ranging from -60 to 0)
- `speechiness` : relative length of the track containing any kind of human voice, (Ranges from 0 to 1)
- `year` : The release year of track, (Ranges from 1921 to 2020)
- `track_id` : The unique identifier for the track

Categorical Variables:

- `playlist_genre` : The genre of the track (one of six categories, namely latin, r&b, pop, rock, rap, and edm)
- `key` : The primary key of the track encoded as integers in between 0 and 11 (starting on C as 0, C# as 1 and so on…)
- `track_artist` : The artists credited for the track
- `release_date` : Date of release mostly in yyyy-mm-dd format, however precision of date may vary
- `track_name` : The name of the song
- `mode` : modality of the track, indicates whether the track starts with a major (1) chord progression or a minor (0)

Below I conduct my data cleaning process, describing each step as a comment. My dataset was relatively clean but I still made necessary adjustments. I used `substring()` in lieu of `gsub()` . As a separate note, I have loaded all libraries in my RMarkdown code, but to save space here I have not included the code.

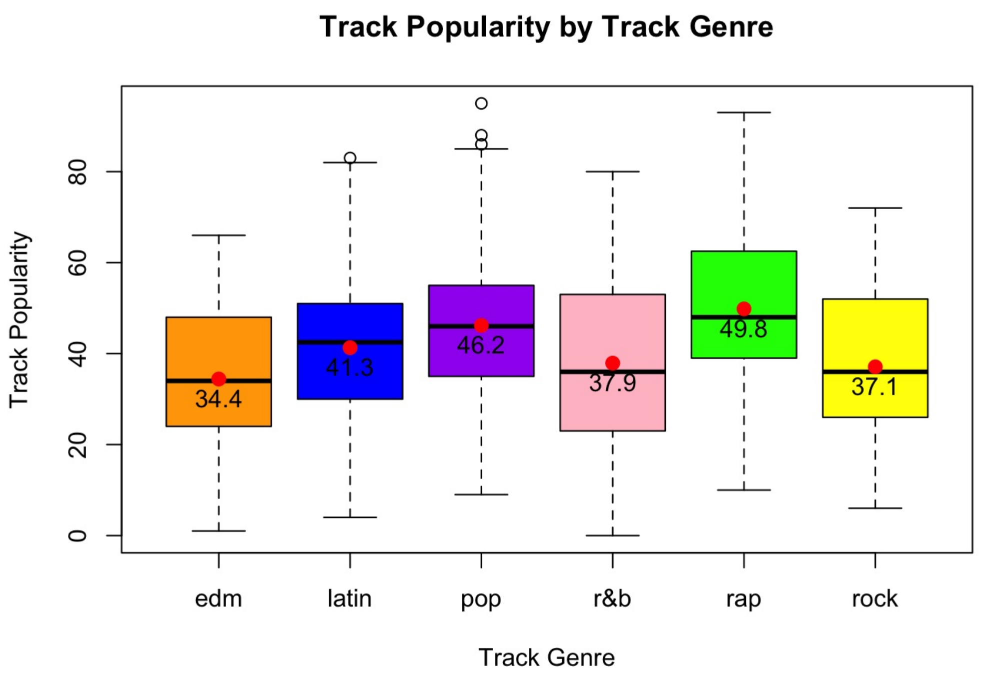
```
# Read in data
spotify <- read.csv("spotify.csv")
# Only use complete observations (i.e., remove any rows with missing observations)
spotify <- spotify[complete.cases(spotify),]
spotify <- spotify[spotify$playlist_genre != "",]
# Dropping Rows: There are songs with zero tempo but with a relatively high popularity in some cases. Similarly, we remove observations with 0 danceability and 0 speechiness. It was found out that those 'zero tempo songs' are not real songs but noises or sounds. They are highly popular because they seem to have a positive effecton a restful sleep or for relaxation. As our goal is to focus on real songs, those data has been removed from the dataset.
spotify <- spotify[spotify$tempo > 0, ]
spotify <- spotify[spotify$danceability > 0, ]
spotify <- spotify[spotify$speechiness > 0, ]
# Remove duplicates by unique id, since it was noticed that there were a a lot of duplicate observations that could affect our results
spotify <- spotify[!duplicated(spotify$track_id), ]
# Recoding 'key' variable
spotify$key <- recode(spotify$key, "0 = 'C'; 1 = 'C#'; 2 = 'D'; 3 = 'E_flat'; 4 = 'E'; 5 = 'F'; 6 = 'G_flat'; 7 = 'G'; 8 = 'A_flat'; 9 = 'A'; 10 = 'B_flat'; 11 = 'B'")
# Recoding the mode variable
spotify$mode <- recode(spotify$mode, "0 = 'Minor'; 1 = 'Major'")
# Standardizing units of duration to seconds and removing the outlier
spotify$duration <- spotify$duration_ms/1000
spotify <- spotify[spotify$duration < 500, ]
# Creating a Year Published System
spotify$Year <- as.numeric(gsub(".*/", "", spotify$track_album_release_date))
spotify$Year <- ifelse(spotify$Year > 30 & spotify$Year < 100, spotify$Year+1900, spotify$Year)
spotify$Year <- ifelse(spotify$Year < 30, spotify$Year+2000, spotify$Year)
#Subset of 2020 published songs
spotify_new <- subset(spotify, spotify$Year == 2020)
```

Graphics

Boxplot

I start with analyzing boxplots created from our cleaned data. This boxplot analyzed was that of `track_popularity` (judged on a scale of 1 to 100) by track genre.

```
boxplot(spotify_new$track_popularity~spotify_new$playlist_genre, main = "Track Popularity by Track Genre", xlab = "Track Genre", ylab = "Track Popularity", col = c("Orange", "Blue", "Purple", "Pink", "Green", "Yellow"))
means <- tapply(spotify_new$track_popularity, spotify_new$playlist_genre, mean)
points(means, col = "red", pch = 19, cex = 1.2)
text(x = c(1:6), y = means - 4.2, labels = round(means,1))
```

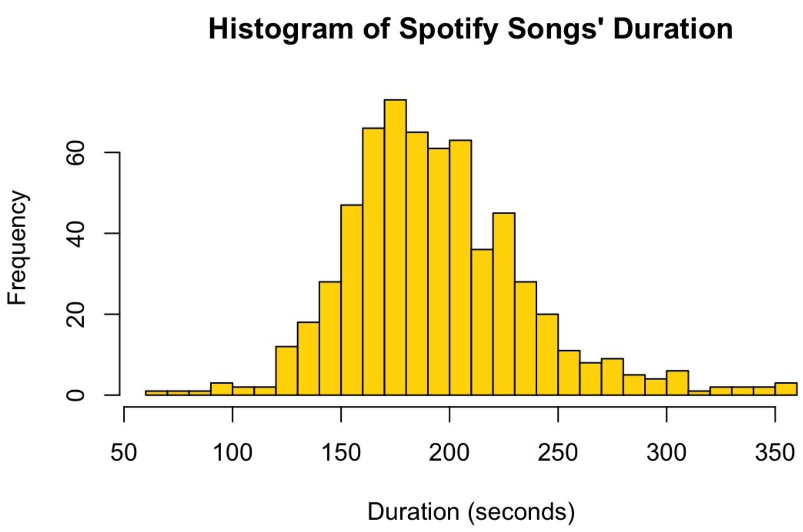


Here I note rather similar ranges for each genre (and no outliers), likely indicating minimal heteroskedacity. Pop also seems to be the most popular genre with the highest mean popularity rating.

Histogram

This histogram is the distribution of tracks' song `duration`. The following distribution is not only interesting for its slightly right-skewed nature, but also seems to highlight a "sweet spot" for artist's music timing (150-250 seconds). This would make intuitive sense as listeners would likely prefer songs that are neither too long or short.

```
hist(spotify_new$duration, col = "gold", main = "Histogram of Spotify Songs' Duration", xlab = "Duration (seconds
)", breaks = 40)
```

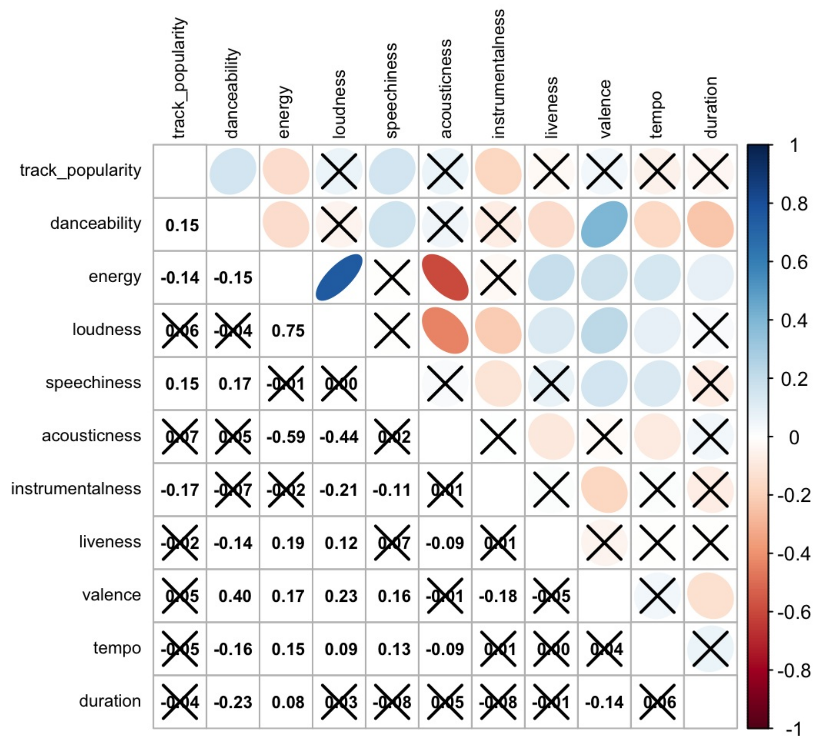


Scatterplot (Correlation Plot)

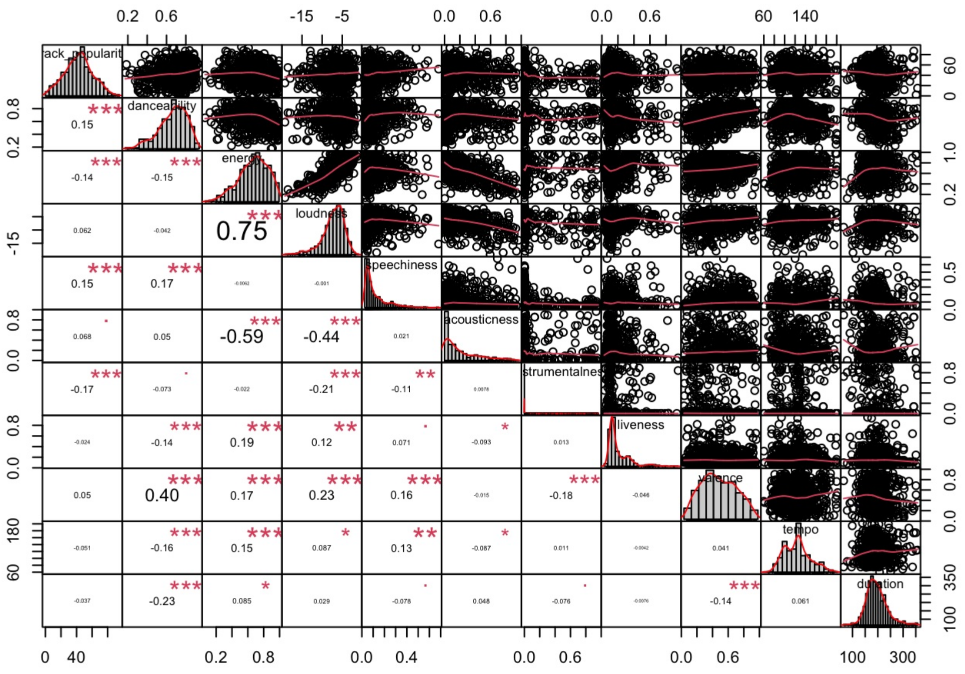
Here I make a matrix correlation plot.

So, let's take a look at the pairwise correlations between all of the continuous variables in the model for the year 2020 (if all years were included, the data would be too large and messy) using the `corrplot.mixed()` function:

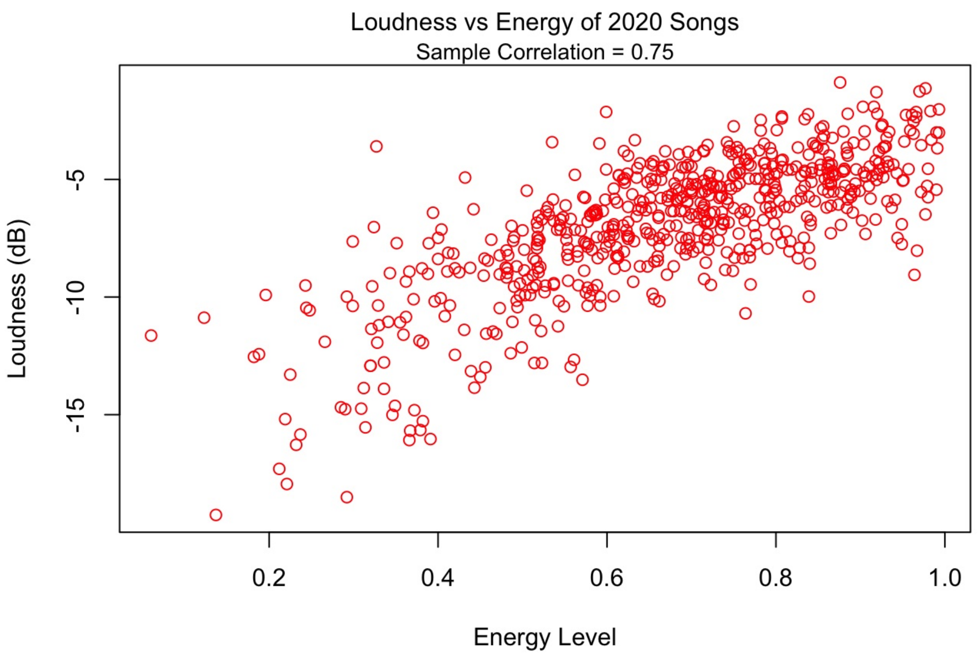
```
spotify_cont <- spotify_new[,c(4, 12, 13, 15, 17:22, 24)]
corrplot.mixed(cor(spotify_cont), lower.col = "black", upper = "ellipse", tl.col = "black", number.cex = .7, tl.p
os = "lt", tl.cex = .7, p.mat = cor.mtest(spotify_cont, conf.level = .95)$p, sig.level = .05)
```



```
pairsJDRS(spotify_cont)
```



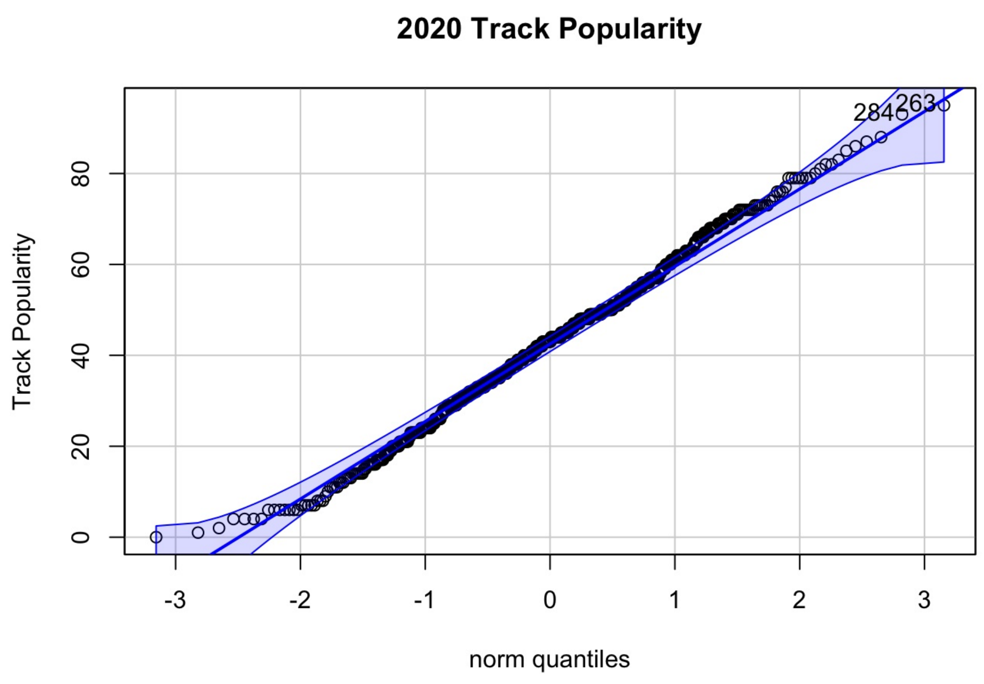
```
plot(spotify_new$energy, spotify_new$loudness, col = "red", ylab = "Loudness (dB)", xlab = "Energy Level")
mtext("Loudness vs Energy of 2020 Songs", line = 1)
mtext(paste("Sample Correlation =", round(cor(spotify_new$energy, spotify_new$loudness), 3)), cex = 0.9, line = 0
)
```



This is interesting - it looks like the strongest correlations are between `energy` and `loudness` (r = 0.75), `acousticness` and `energy` (r = -0.59). The scatterplot of energy vs loudness confirms this strong, positive correlation of r = 0.75 between the `energy` and `loudness` factors, as noted by the roughly linear pattern with positive trend. *I'll use bootstrapping in a later section to generate a confidence interval for the true correlation between `energy` and `loudness`.*

Quantile Plot and Residual Plot

```
qqPlot(spotify_new$track_popularity, main = "2020 Track Popularity", ylab = "Track Popularity")
```



```
## [1] 263 284
```

I plan on using track popularity as my response variable in later sections, so it made sense to create a normal quantile plot of track_popularity. From this quantile plot, mostly all the datapoints lie within the blue confidence intervals, indicating that the data is approximately normally distributed. Even though we can assume normality from this plot, I will still attempt to find lambda and apply a boxcox transformation later in the MLR section.

Basic Tests

T-Test

Let's run a two-sample t-test to determine whether the means of two populations (in this case, the values of the valence across modes) are equivalent.

```
t.test(spotify_new$valence ~ spotify_new$mode)
```

```
##
##  Welch Two Sample t-test
##
## data:  spotify_new$valence by spotify_new$mode
## t = 0.1423, df = 613.94, p-value = 0.8869
## alternative hypothesis: true difference in means between group Major and group Minor is not equal to 0
## 95 percent confidence interval:
##  -0.03364295  0.03889946
## sample estimates:
## mean in group Major mean in group Minor
##           0.4729925           0.4703643
```

Since the p-value (0.8942) is greater than the alpha level (0.05), we fail to reject the null hypothesis, meaning that there is not a statistically significant difference in the valence, or positiveness of a track, between tracks that begin with a major chord progression and those that begin with a minor chord progression.

Correlation Bootstrap

Let's now run a correlation test to understand the strength of association between the values of `energy` and `loudness` for 2020 (continued from the Scatterplots section).

```
cor1 <- cor.test(spotify_new$energy, spotify_new$loudness)
cor1
```

```
##
##  Pearson's product-moment correlation
##
## data:  spotify_new$energy and spotify_new$loudness
## t = 28.337, df = 623, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.7140002 0.7827655
## sample estimates:
##       cor
## 0.7504064
```

Since the p-value is very low (less than 2.2e-16) and close to 0, we can reject the null hypothesis and conclude that the true correlation is not equal to 0 and statistically significant. From the confidence interval, we know with 95% confidence that the true correlation is between 0.714 and 0.783; moreover, as 0 is not in this interval, the correlation must be different from 0 with statistical significance.

Bootstrap

I will now utilize bootstrapping to derive an estimate of the correlation between the `energy` and `loudness` of tracks for the entire population of tracks created in 2020.

```
N <- nrow(spotify_new)
n_samp <- 10000

corResults <- rep(NA, n_samp)

for (i in 1:n_samp) {
  s <- sample(1:N, N, replace = T)
  fakedata <- spotify_new[s, ]
  corResults[i] <- cor(fakedata$energy, fakedata$loudness)
}

print("Bootstrapped CI:")
```

```
## [1] "Bootstrapped CI:"
```

```
(ci_r <- quantile(corResults, c(0.025, .975)))
```

```
##      2.5%     97.5%
## 0.7129031 0.7836629
```

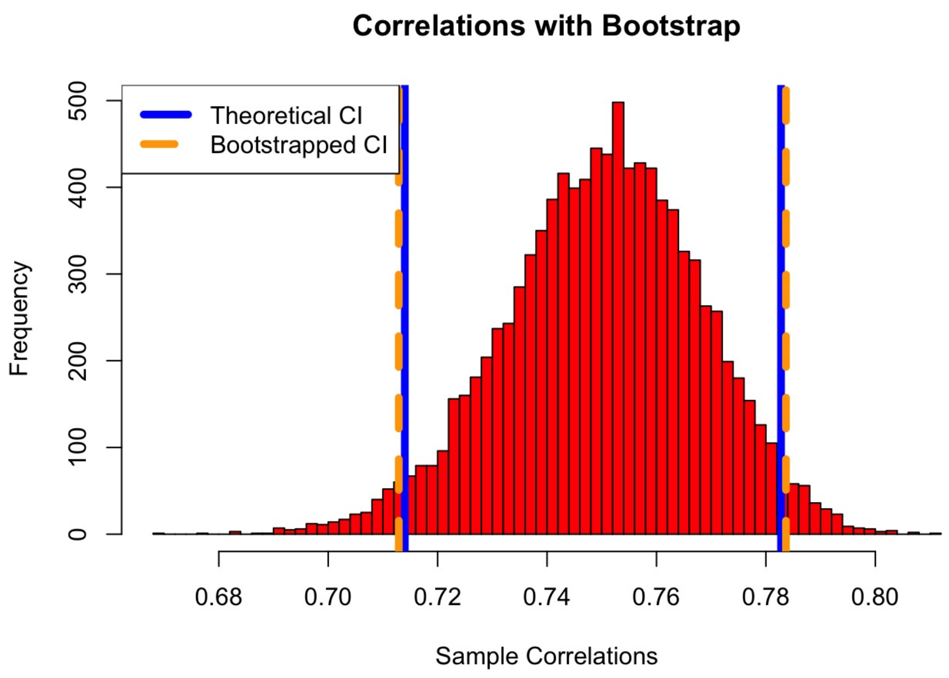
```
print("Theoretical CI:")
```

```
## [1] "Theoretical CI:"
```

```
cor1$conf.in
```

```
## [1] 0.7140002 0.7827655
## attr(,"conf.level")
## [1] 0.95
```

```
hist(corResults, breaks = 60, main = "Correlations with Bootstrap", xlab = "Sample Correlations", col = "Red")
abline(v = cor.test(spotify_new$energy, spotify_new$loudness)$conf.int, col = "Blue", lwd = 5, lty = 1)
abline(v = ci_r, col = "Orange", lwd = 5, lty = 2)
legend("topleft", c("Theoretical CI", "Bootstrapped CI"), col = c("Blue", "Orange"), lwd = 5, lty = c(1, 2))
```



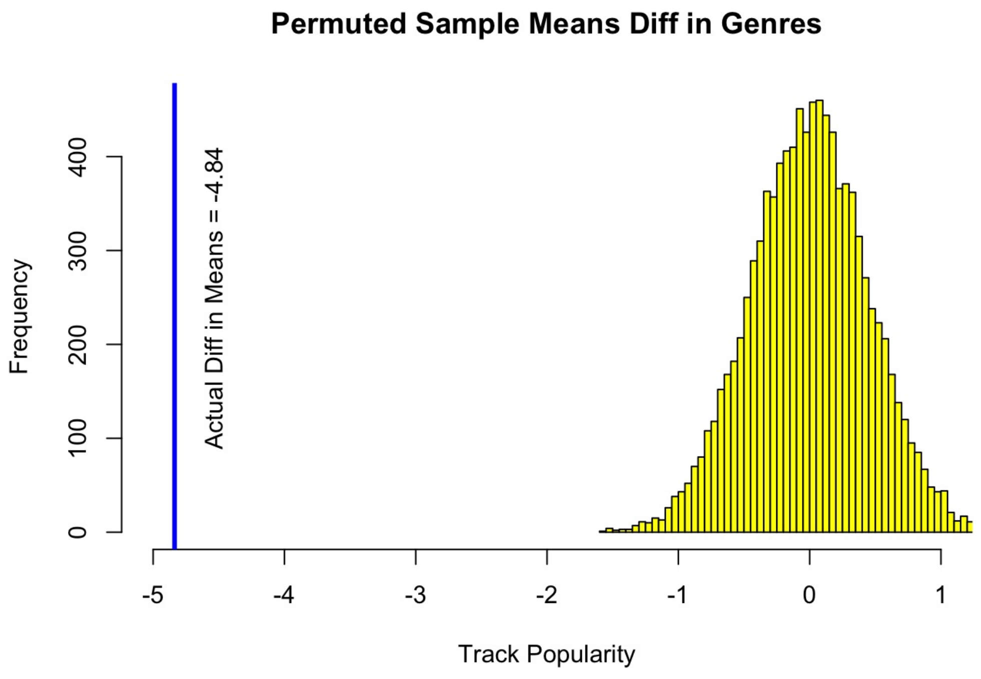
For the correlation, the bootstrapped CI has a slightly larger range than the theoretical CI. Since the ranges for correlation in both the theoretical CI and bootstrapped CI are far from 0, we are 95% confident that the true correlation between energy and loudness is in the interval 0.71 and 0.78, which indicates that the correlation is high.

Permutation Test

The following permutation test was done to compare if there was a significant difference between the mean track popularity in 2018 and 2019. The null hypothesis notes that there is not a significant difference between the mean track popularity from 2018 to 2019. Conversely, the alternative hypothesis states that there is a statistically significant difference in the mean track popularity improvement from 2018 to 2019.

```
spotify_test <- subset(spotify, spotify$Year == 2018 | spotify$Year == 2019)
actualdiff <- mean(spotify_test$track_popularity[spotify_test$Year==2018]) - mean(spotify_test$track_popularity[s
potify_test$Year==2019])

N <- 10000
diffvals <- rep(NA, N)
for (i in 1:N) {
  fakeyear <- sample(spotify_test$Year)
  diffvals[i] <- mean(spotify_test$track_popularity[fakeyear == 2018]) - mean(spotify_test$track_popularity[fakey
ear == 2019])
}
hist(diffvals, col = "yellow", main = "Permuted Sample Means Diff in Genres", xlab = "Track Popularity", breaks =
50, xlim = c(-5,1))
abline(v = actualdiff, col = "blue", lwd = 3)
text(actualdiff+0.3, 250, paste("Actual Diff in Means =", round(actualdiff,2)),srt
= 90)
```



```
(mean(abs(diffvals) >= abs(actualdiff)))
```

```
## [1] 0
```

My null hypothesis is that the means in 2018 and 2019 are even. We note that the p-value is approximately 0 (< 0.05), indicating that we can reject the null hypothesis and thus conclude that there is a statistically significant difference between the mean track popularity between 2018 and 2019.

ANOVA and ANCOVA

I will be performing two advanced techniques, namely ANOVA and ANCOVA.

ANOVA

Let's do a one-way ANOVA comparing mean track popularity across track genres for 2020.

Before we start, let's see if the equal variances assumption of ANOVA is met:

```
print("SD by Genre")
```

```
## [1] "SD by Genre"
```

```
(sds <- tapply(spotify_new$track_popularity, spotify_new$playlist_genre, sd))
```

```
##      edm    latin      pop      r&b      rap     rock 
## 15.69619 18.39403 15.81571 20.87000 16.33822 18.65182
```

```
print("Ratio of Max/Min Sample SD")
```

```
## [1] "Ratio of Max/Min Sample SD"
```

```
round(max(sds)/min(sds), 1)
```

```
## [1] 1.3
```

Since the ratio of the largest SD to the smallest SD is less than 2 (1.3), we can reasonably assume that the equal variances assumption of ANOVA is met. Great, now let's perform the actual one-way ANOVA test using the `aov()` function:

```
aov1 <- aov(spotify_new$track_popularity ~ spotify_new$playlist_genre)
summary(aov1)
```

```
##                              Df Sum Sq Mean Sq F value   Pr(>F)    
## spotify_new$playlist_genre   5  20308    4062   13.95 6.13e-13 ***
## Residuals                  619 180192     291                     
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

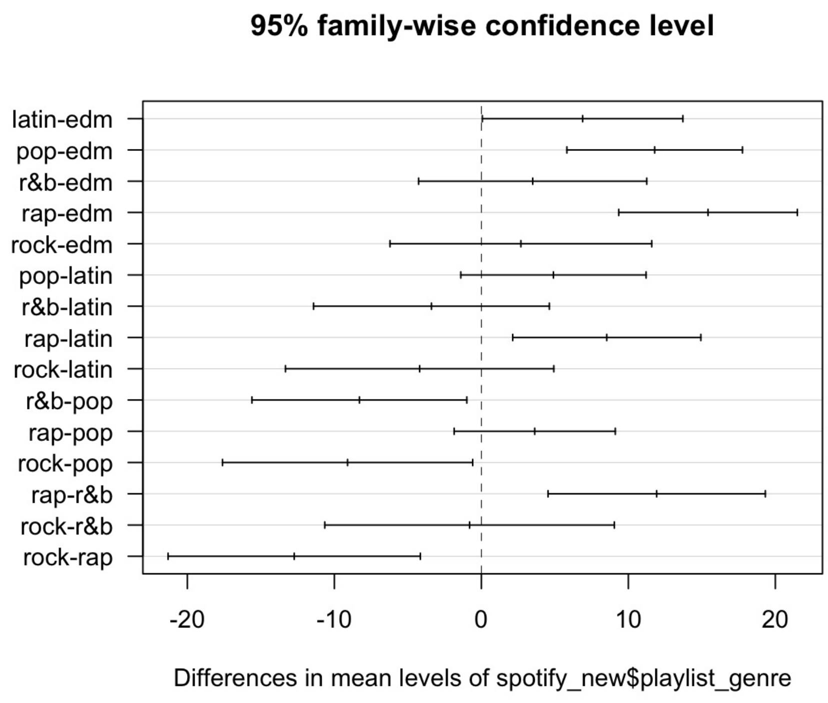
The p-value for the test of significance is 4.91e-13 which is less than any reasonable alpha (say 0.05), so we can conclude that the mean track popularity is not the same across track genres. There were 6 genres and 625 complete observations, so we should have 6 - 1 = 5 degrees of freedom associated with genre and 625 - 6 = 619 degrees of freedom associated with errors (residuals), which are the values reported by the test.

Let's take a look at the family-wise confidence intervals to see if there are any statistically significant pairwise differences in popularity between genres using the `TukeyHSD()` function:

```
TukeyHSD(aov1)
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
## 
## Fit: aov(formula = spotify_new$track_popularity ~ spotify_new$playlist_genre)
## 
## $`spotify_new$playlist_genre`
##                   diff         lwr        upr     p adj
## latin-edm    6.8871581   0.0646108 13.7097053 0.0463437
## pop-edm     11.7892857   5.8179047 17.7606667 0.0000004
## r&b-edm      3.4909251  -4.2703845 11.2522346 0.7928481
## rap-edm     15.4182331   9.3444813 21.4919848 0.0000000
## rock-edm     2.6868467  -6.2160361 11.5897295 0.9551246
## pop-latin    4.9021277  -1.4006192 11.2048745 0.2284193
## r&b-latin   -3.3962330 -11.4152823  4.6228163 0.8316659
## rap-latin    8.5310750   2.1312557 14.9308943 0.0021039
## rock-latin  -4.2003114 -13.3287586  4.9281359 0.7765323
## r&b-pop     -8.2983607 -15.6069441 -0.9897772 0.0155583
## rap-pop      3.6289474  -1.8544756  9.1123703 0.4080887
## rock-pop    -9.1024390 -17.6135370 -0.5913411 0.0280712
## rap-r&b     11.9273080   4.5348482 19.3197678 0.0000708
## rock-r&b    -0.8040784 -10.6539171  9.0457604 0.9999054
## rock-rap   -12.7313864 -21.3146176 -4.1481552 0.0003690
```

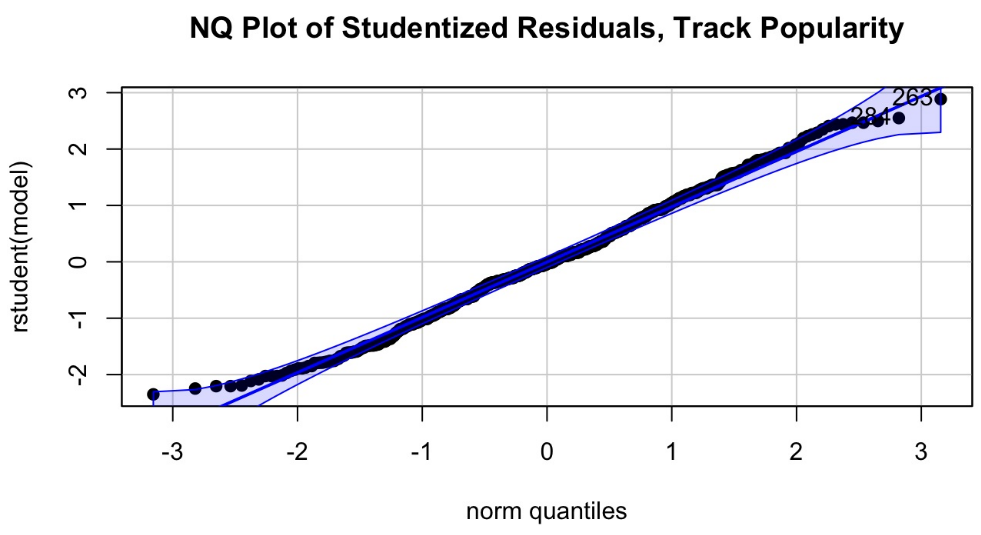
```
par(mar = c(5, 11, 4, 1))
plot(TukeyHSD(aov1), las = 1)
```

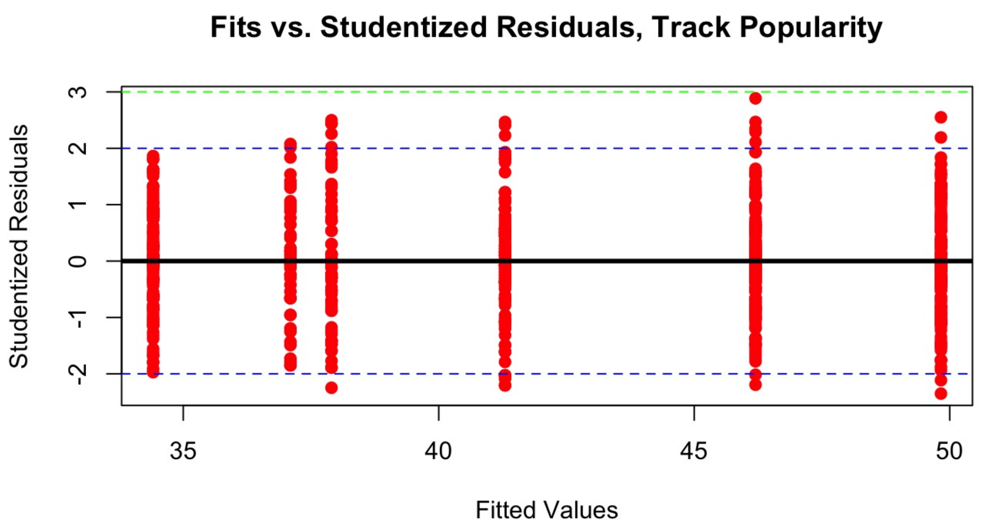


The Tukey confidence intervals find the following groups are statistically significantly different from each other because their adjusted p-values are less than the typical threshold of 0.05 (i.e., because 0 is NOT included in their 95% pairwise confidence intervals): latin-edm, pop-edm, rap-edm, rap-latin, r&b-pop, rock-pop, rock-rap, and rap-r&b. Additionally, the Tukey confidence intervals find that the following group is NOT statistically significantly different from each other because their adjusted p-value is greater than the typical threshold of 0.05 (i.e., because 0 IS included in their 95% pairwise confidence intervals): r&b-edm, rock-edm, pop-latin, r&b-latin, rock-latin, rap-pop, rock-r&b.

Now, let's take a look at the distribution of the residuals from our model using the `myResPlots2()` function:

```
par(mfrow = c(2, 1))
myResPlots2(aov1, label = "Track Popularity")
```





The normal quantile plot does look linear within the shaded region which suggests that the distribution of residuals is approximately normal inside each group - this matches one of the ANOVA assumptions. In good news, the plot of fits vs. residuals does not show any evidence of heteroskedasticity nor many outliers, meaning that the variances are relatively equal across genres which satisfies the second assumption of one-way ANOVA.

Just out of curiosity, let's run a Kruskal-Wallis non-parametric one-way ANOVA test to see if our model results change significantly from the regular one-way ANOVA (remember that the Kruskal-Wallis test makes no assumption about the Normality of the distribution within each group):

```
kruskal.test(spotify_new$track_popularity ~ spotify_new$playlist_genre)
```

```
## 
##  Kruskal-Wallis rank sum test
## 
## data:  spotify_new$track_popularity by spotify_new$playlist_genre
## Kruskal-Wallis chi-squared = 60.081, df = 5, p-value = 1.17e-11
```

The Kruskal-Wallis test, similar to the regular one-way ANOVA, reports a p-value of 9.516e-12 < 0.05, again leading to the conclusion that the mean track popularity is not the same across track genres.

ANCOVA

The goal here is to fit an ANCOVA model predicting `track_popularity` for tracks made in 2020 based on `danceability`, `playlist_genre`, and the interaction of `danceability` and `genre`. We will also then visually assess the predictive ability of `danceability` and how the song's `genre` affected the popularity of a track. Let's create the ANCOVA model, get linear model and ANOVA summary information for this model, and create a plot of `track_popularity` vs `danceability` with regression lines added for `playlist_genre`:

```
ancova_mod <- lm(spotify_new$track_popularity ~ spotify_new$playlist_genre*spotify_new$danceability)
Anova(ancova_mod, type = 'III')
```

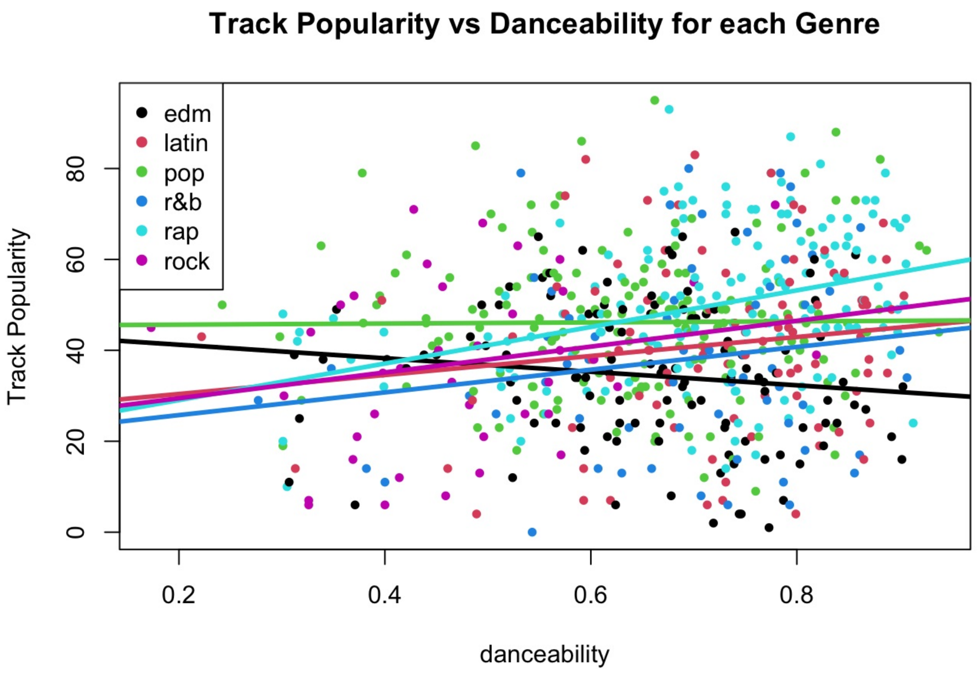
```
## Anova Table (Type III tests)
##
## Response: spotify_new$track_popularity
##                                                 Sum Sq  Df F value
## (Intercept)                                       8646   1 30.6361
## spotify_new$playlist_genre                        3198   5  2.2665
## spotify_new$danceability                           444   1  1.5720
## spotify_new$playlist_genre:spotify_new$danceability   4559   5  3.2304
## Residuals                                       173005 613
##                                                    Pr(>F)
## (Intercept)                                     4.618e-08 ***
## spotify_new$playlist_genre                       0.046529 *
## spotify_new$danceability                         0.210389
## spotify_new$playlist_genre:spotify_new$danceability  0.006896 **
## Residuals
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(ancova_mod)
```

```
##
## Call:
## lm(formula = spotify_new$track_popularity ~ spotify_new$playlist_genre *
##     spotify_new$danceability)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -38.94 -11.16   0.05  10.90  48.77
##
## Coefficients:
##                                                       Estimate Std. Error
## (Intercept)                                             44.230      7.991
## spotify_new$playlist_genrelatin                        -18.018     12.774
## spotify_new$playlist_genrepop                            1.171     10.143
## spotify_new$playlist_genrer&b                          -23.480     13.719
## spotify_new$playlist_genrerap                          -23.232     10.703
## spotify_new$playlist_genrerock                         -20.459     12.563
## spotify_new$danceability                               -14.892     11.878
## spotify_new$playlist_genrelatin:spotify_new$danceability   35.821     18.069
## spotify_new$playlist_genrepop:spotify_new$danceability     16.138     15.230
## spotify_new$playlist_genrer&b:spotify_new$danceability     39.906     19.894
## spotify_new$playlist_genrerap:spotify_new$danceability     55.160     15.374
## spotify_new$playlist_genrerock:spotify_new$danceability    43.322     23.182
##                                                       t value Pr(>|t|)
## (Intercept)                                             5.535 4.62e-08 ***
## spotify_new$playlist_genrelatin                        -1.410  0.15891
## spotify_new$playlist_genrepop                           0.115  0.90809
## spotify_new$playlist_genrer&b                          -1.711  0.08751 .
## spotify_new$playlist_genrerap                          -2.171  0.03035 *
## spotify_new$playlist_genrerock                         -1.629  0.10393
## spotify_new$danceability                               -1.254  0.21039
## spotify_new$playlist_genrelatin:spotify_new$danceability   1.982  0.04787 *
## spotify_new$playlist_genrepop:spotify_new$danceability     1.060  0.28975
## spotify_new$playlist_genrer&b:spotify_new$danceability     2.006  0.04530 *
## spotify_new$playlist_genrerap:spotify_new$danceability     3.588  0.00036 ***
## spotify_new$playlist_genrerock:spotify_new$danceability    1.869  0.06214 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 16.8 on 613 degrees of freedom
## Multiple R-squared:  0.1371, Adjusted R-squared:  0.1216
## F-statistic: 8.856 on 11 and 613 DF,  p-value: 1.03e-14
```

```
plot(track_popularity ~ danceability, data = spotify_new, col = factor(playlist_genre), pch = 16, cex = 0.8, ylab
= "Track Popularity", main = "Track Popularity vs Danceability for each Genre")
legend("topleft", col = 1:6, legend = levels(factor(spotify_new$playlist_genre)), pch = 16)
coefs <- coef(ancova_mod)

abline(a = coefs[1], b = coefs[7], col = "black", lwd = 3)
for (i in 2:6){
  abline(a = coefs[1] + coefs[i], b = coefs[7] + coefs[i+6], col = i, lwd = 3)
}
```



The ‘default’ reference level in this model is ‘playlist_genreedm’ (i.e., the edm genre). Overall, the playlist genre of the track and the interaction between playlist genre and danceability are both significant predictors of track popularity at the alpha level of 0.05, but danceability alone is not a significant predictor of track popularity at the alpha level of 0.05.

The equation for predicting track popularity for tracks in the edm genre is: track popularity = 44.2 - 14.9 * danceability. The equation for predicting track popularity for tracks in the latin genre is: track popularity = 26.2 + 21.0 * danceability. The equation for predicting track popularity for tracks in the pop genre is: track popularity = 45.4 + 1.25 * danceability. The equation for predicting track popularity for tracks in the r&b genre is: track popularity = 20.8 + 25.0 * danceability. The equation for predicting track popularity for tracks in the rap genre is: track popularity = 21.0 + 40.2 * danceability. The equation for predicting track popularity for tracks in the rock genre is: track popularity = 23.8 + 28.4 * danceability.

From this information, we can clearly see that the only genre with a decrease in popularity as danceability increases is edm (slope = -14.9). The genre with the greatest increase in popularity as danceability increases is rap, with a slope of 40.2. While the pop genre has the highest intercept, it also sees the least change in popularity as its danceability increases, with a slope of only 1.25. It should also be noted that while r&b, rap and rock start off with intercepts in a similar range (20 to 25), rap clearly sees the greatest increase in popularity as danceability increases (slope = 40.2), followed by rock (slope = 28.4) and lastly r&b (slope = 25.0), showing that danceability has the greatest effect on rap then rock then r&b.

Multiple Linear Regression

I chose to create a Generalized Linear Model (GLM) for this section.

To fulfill the objectives of our statistical analysis, I will now be using track_popularity as our response variable in our regression model to see how useful my variables are as predictors. To do this, I will perform backward stepwise regression on track_popularity that includes all of the 10 possible continous predictors. These variables are acousticness, danceability, energy, duration_ms, instrumentalness, valence, tempo, loudness, liveness and speechiness. I will also include 3 of the relevant categorical variables - playlist_genre, key and mode. Additionally, all two-way interactions between playlist_genre and the continuous variables are included. I did not include interactions between key and the continuous variables or mode and the continuous variables. I will then perform manual backwards stepwise regression, removing non-significant terms until all terms have p-values less than 0.05. In this process, I removed interactions BEFORE removing any main effects.

```
spotify_new <- spotify_new[spotify_new$track_popularity != 0, ]
lm1 <- lm(track_popularity ~ key + mode + danceability + energy + loudness + instrumentalness + speechiness + aco
usticness + duration + valence + tempo + liveness + playlist_genre*danceability + playlist_genre*energy + playlis
t_genre*loudness + playlist_genre*instrumentalness + playlist_genre*acousticness + playlist_genre*duration + play
list_genre*valence + playlist_genre*tempo + playlist_genre*liveness + playlist_genre*speechiness, data = spotify_
new)
Anova(lm1, type = 3)
```

```
## Anova Table (Type III tests)
##
## Response: track_popularity
##                                Sum Sq  Df F value  Pr(>F)
## (Intercept)                       877   1  3.4637 0.06327 .
## key                              3872  11  1.3903 0.17321
## mode                              100   1  0.3942 0.53036
## danceability                      199   1  0.7852 0.37593
## energy                              1   1  0.0040 0.94938
## loudness                            1   1  0.0042 0.94853
## instrumentalness                  793   1  3.1315 0.07735 .
## speechiness                        52   1  0.2059 0.65017
## acousticness                        2   1  0.0082 0.92806
## duration                          284   1  1.1204 0.29031
## valence                            10   1  0.0407 0.84012
## tempo                              33   1  0.1302 0.71834
## liveness                           80   1  0.3172 0.57354
## playlist_genre                   2981   5  2.3550 0.03942 *
## danceability:playlist_genre      2361   5  1.8649 0.09872 .
## energy:playlist_genre            1352   5  1.0680 0.37711
## loudness:playlist_genre          3205   5  2.5313 0.02802 *
## instrumentalness:playlist_genre  1406   5  1.1103 0.35375
## acousticness:playlist_genre      2177   5  1.7197 0.12819
## duration:playlist_genre          2752   5  2.1741 0.05563 .
## valence:playlist_genre            878   5  0.6934 0.62862
## tempo:playlist_genre              648   5  0.5118 0.76744
## liveness:playlist_genre          1525   5  1.2043 0.30573
## speechiness:playlist_genre       1127   5  0.8899 0.48747
## Residuals                      138240 546
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
invisible(summary(lm1))
```

As we can see, the model here has no significant predictors even though the R^2 iis 0.31 (although adj. R^2 is only 0.21). I now begin by removing non-significant interaction terms one by one, starting with the term with the highest p-value. This is what my model looked like after I removed all the non-significant interaction terms but were yet to remove any of the main terms:

```
lm2 <- lm(track_popularity ~ key + mode + danceability + energy + loudness + instrumentalness + speechiness + aco
usticness + duration + valence + tempo + liveness + playlist_genre*danceability + playlist_genre*loudness + playl
ist_genre*duration, data = spotify_new)
Anova(lm2, type = 3)
```

```
## Anova Table (Type III tests)
##
## Response: track_popularity
##                            Sum Sq  Df F value    Pr(>F)
## (Intercept)                  9479   1 37.4170 1.753e-09 ***
## key                          3591  11  1.2886  0.226841
## mode                          201   1  0.7942  0.373211
## danceability                  473   1  1.8677  0.172266
## energy                       5039   1 19.8910 9.845e-06 ***
## loudness                      961   1  3.7926  0.051962 .
## instrumentalness             1171   1  4.6220  0.031976 *
## speechiness                  1142   1  4.5084  0.034150 *
## acousticness                  339   1  1.3400  0.247506
## duration                      281   1  1.1096  0.292604
## valence                       307   1  1.2102  0.271750
## tempo                          14   1  0.0548  0.814984
## liveness                        3   1  0.0104  0.918841
## playlist_genre               4360   5  3.4417  0.004495 **
## danceability:playlist_genre  2813   5  2.2208  0.050838 .
## loudness:playlist_genre      3154   5  2.4902  0.030280 *
## duration:playlist_genre      3226   5  2.5472  0.027097 *
## Residuals                  147187 581
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
invisible(summary(lm2))
```

After having removed all the non-signifcant main terms, this is what my final model looks like:

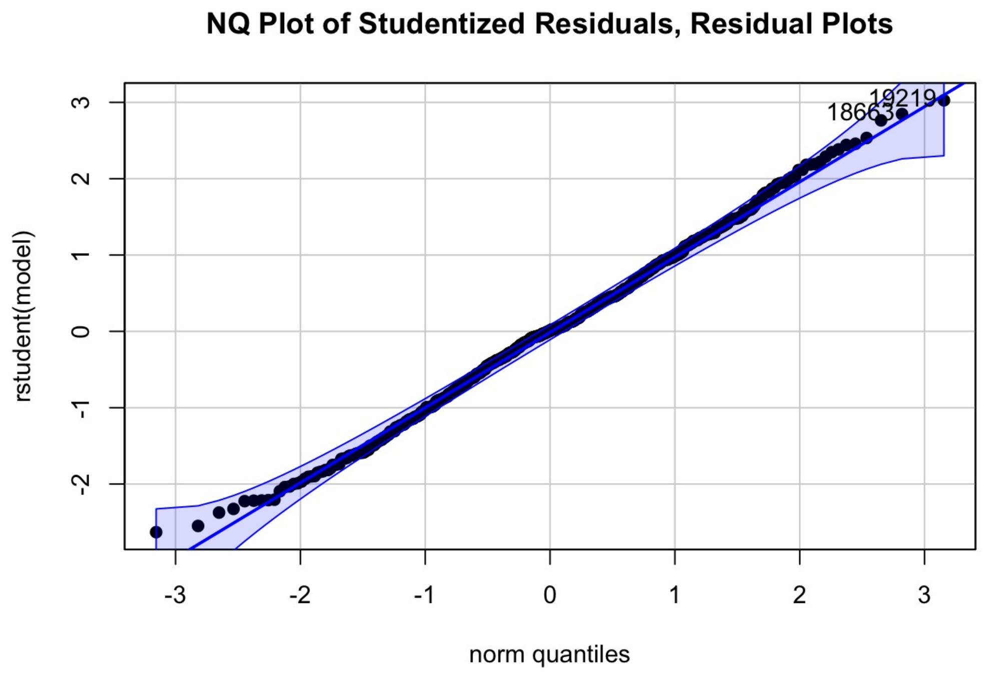
```
lm3 <- lm(track_popularity ~ danceability + energy + loudness + instrumentalness + speechiness + playlist_genre*d
anceability + playlist_genre*loudness + playlist_genre*duration, data = spotify_new)
Anova(lm3, type = 3)
```

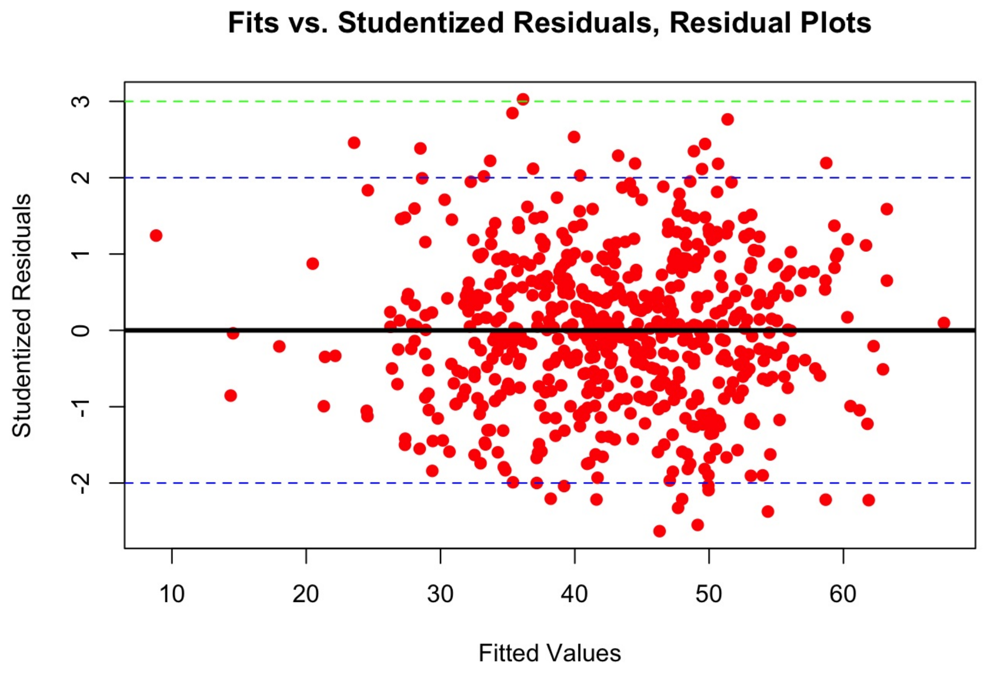
```
## Anova Table (Type III tests)
##
## Response: track_popularity
##                             Sum Sq  Df F value    Pr(>F)
## (Intercept)                  10857   1 42.6683 1.390e-10 ***
## danceability                   740   1  2.9090  0.088605 .
## energy                        6417   1 25.2179 6.771e-07 ***
## loudness                      1034   1  4.0641  0.044252 *
## instrumentalness               947   1  3.7230  0.054140 .
## speechiness                   1023   1  4.0209  0.045391 *
## playlist_genre                4527   5  3.5586  0.003527 **
## duration                       294   1  1.1548  0.282971
## danceability:playlist_genre   3505   5  2.7553  0.017964 *
## loudness:playlist_genre       2853   5  2.2423  0.048759 *
## playlist_genre:duration       3253   5  2.5566  0.026572 *
## Residuals                   151904 597
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(lm3)
```

```
##
## Call:
## lm(formula = track_popularity ~ danceability + energy + loudness +
##     instrumentalness + speechiness + playlist_genre * danceability +
##     playlist_genre * loudness + playlist_genre * duration, data = spotify_new)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -40.312 -10.635   0.119  10.188  46.854
##
## Coefficients:
##                                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)                      86.68622   13.27081   6.532 1.39e-10 ***
## danceability                    -20.39872   11.95995  -1.706  0.08861 .
## energy                          -29.52658    5.87975  -5.022 6.77e-07 ***
## loudness                          1.29586    0.64280   2.016  0.04425 *
## instrumentalness                 -6.29300    3.26143  -1.930  0.05414 .
## speechiness                      13.35559    6.66044   2.005  0.04539 *
## playlist_genrelatin               5.88567   20.36104   0.289  0.77263
## playlist_genrepop                -4.98127   15.81365  -0.315  0.75287
## playlist_genrer&b                -6.55362   20.22919  -0.324  0.74608
## playlist_genrerap               -44.40014   15.70089  -2.828  0.00484 **
## playlist_genrerock              -57.77212   21.72459  -2.659  0.00804 **
## duration                         -0.04221    0.03928  -1.075  0.28297
## danceability:playlist_genrelatin  22.48802   18.57946   1.210  0.22662
## danceability:playlist_genrepop    20.55830   15.13094   1.359  0.17476
## danceability:playlist_genrer&b    38.67944   19.44799   1.989  0.04717 *
## danceability:playlist_genrerap    53.29604   15.42717   3.455  0.00059 ***
## danceability:playlist_genrerock   37.47466   22.53626   1.663  0.09687 .
## loudness:playlist_genrelatin       2.01206    0.92463   2.176  0.02994 *
## loudness:playlist_genrepop         0.80322    0.72192   1.113  0.26632
## loudness:playlist_genrer&b         1.71308    0.97210   1.762  0.07854 .
## loudness:playlist_genrerap        -0.08878    0.73719  -0.120  0.90418
## loudness:playlist_genrerock       -0.53803    1.11098  -0.484  0.62836
## playlist_genrelatin:duration      -0.02551    0.05834  -0.437  0.66205
## playlist_genrepop:duration         0.02907    0.05293   0.549  0.58309
## playlist_genrer&b:duration        -0.04139    0.08108  -0.511  0.60986
## playlist_genrerap:duration         0.09602    0.04854   1.978  0.04839 *
## playlist_genrerock:duration        0.17074    0.07570   2.256  0.02445 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 15.95 on 597 degrees of freedom
## Multiple R-squared:  0.2354, Adjusted R-squared:  0.2021
## F-statistic: 7.068 on 26 and 597 DF,  p-value: < 2.2e-16
```

```
myResPlots2(lm3)
```





From the results of the regression model, we can see that we obtained a multiple R^2 of 0.24 and and an adjusted R^2 of 0.205/0.21, which is the same adjusted R^2 we started with. What this means is that only 20.52% of the variability in track popularity can be explained by the predictors.

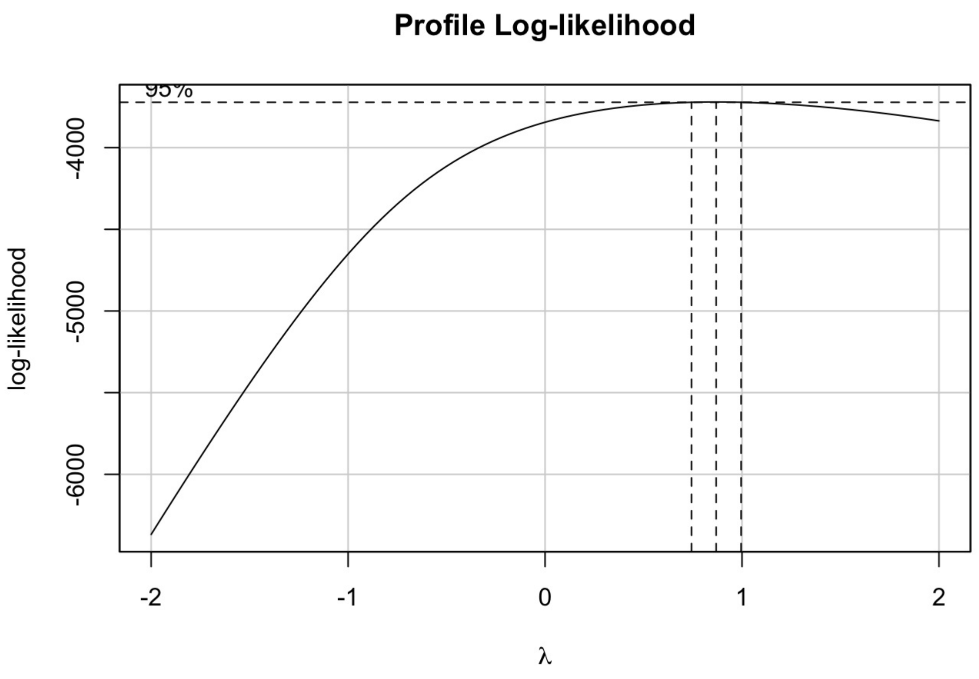
Even though we obtained a low R^2 value, the residual plots appear to show approximate normal distribution of residuals, with practically all the residuals lying within the blue confidence intervals and the plot of fits vs residuals having no defined pattern and a very small proportion of outliers. This likely indicates a lack of heteroskedacity, indicating that a transformation is perhaps not necessary. Overall, this suggests that the choice of model was appropriate.

Despite getting residual plots that satisfy our expectations, we still have to reconcile the low R^2 value. A plausible explanation for why this happened is that the variables we have in our dataset likely have high variances in their datapoints. This causes them to have low predictive power for track popularity, something that can also be seen from the low correlations in the corrplot earlier. This suggests that the current variables are not good predictors of popularity as their datapoints are too scattered, and perhaps more metrics need to be added to the dataset.

In terms of specific predictors, we can see that at alpha = 0.05 energy is a significant negative predictor in its own right, with a large negative coefficient (-29.5) and speechiness is a significant positive predictor, having a moderately positive effect (coefficient = 13.4). Instrumentalness also had a slight negative predictor (coef = -6.29) using a signficance level of alpha = 0.10. In the edm genre (default value), at alpha = 0.05, loudness has a very slight positive effect (coef = 1.30) while at alpha = 0.1, danceability has a moderately large negative effect on popularity (coef = -20.4). Interestingly, being in the rap and rock genres has a very large negative effect (coefs = -44.3 and -57.7 respectively) at alpha = 0.05, with a magnitude much larger than the other genres. However, the relatively large positive effects (significant at alpha = 0.05) of duration for these two genres (coefs = 0.0955 and 0.170) likely compensates for this. In these two genres, danceability alsi has a very large positive effect (coefs = 53.2 and 37.4). Along with rap and rock, a increase in danceability also has a very large positive effect on r&b significant at alpha = 0.05 (coef = 38.6) and loudness has the second largest positive effect (coef = 1.71) significant at alpha = 0.1.

Even though it wasn't necessary, we have still applied the BoxCox transformation for experimental purposes as we obtained lambda = 0.86

```
trans1 <- boxCox(lm3)
```



```
#Figure out what value of lambda (x) gives max value of log-liklihood (y)
(lambda <- trans1$x[which.max(trans1$y)])
```

```
## [1] 0.8686869
```

```
spotify_new$transPop <- (spotify_new$track_popularity)^lambda

lm4 <- lm(transPop ~ danceability + energy + loudness + instrumentalness + speechiness + playlist_genre*danceabil
ity + playlist_genre*loudness + playlist_genre*duration, data = spotify_new)
Anova(lm4, type = 3)
```

```
## Anova Table (Type III tests)
## 
## Response: transPop
##                              Sum Sq  Df F value    Pr(>F)    
## (Intercept)                    3622   1 49.3263 5.923e-12 ***
## danceability                    238   1  3.2479  0.072019 .  
## energy                         1794   1 24.4287 1.003e-06 ***
## loudness                        276   1  3.7641  0.052833 .  
## instrumentalness                303   1  4.1276  0.042631 *  
## speechiness                     294   1  4.0106  0.045667 *  
## playlist_genre                 1326   5  3.6109  0.003165 ** 
## duration                        109   1  1.4827  0.223827    
## danceability:playlist_genre    1005   5  2.7366  0.018643 *  
## loudness:playlist_genre         854   5  2.3255  0.041592 *  
## playlist_genre:duration         960   5  2.6149  0.023699 *  
## Residuals                     43832 597                      
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(lm4)
```

```
## 
## Call:
## lm(formula = transPop ~ danceability + energy + loudness + instrumentalness + 
##     speechiness + playlist_genre * danceability + playlist_genre * 
##     loudness + playlist_genre * duration, data = spotify_new)
## 
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -23.008  -5.348   0.287   5.563  24.252 
## 
## Coefficients:
##                                   Estimate Std. Error t value Pr(>|t|)    
## (Intercept)                       50.06676    7.12870   7.023 5.92e-12 ***
## danceability                     -11.57829    6.42454  -1.802 0.072019 .  
## energy                           -15.61067    3.15843  -4.943 1.00e-06 ***
## loudness                           0.66992    0.34529   1.940 0.052833 .  
## instrumentalness                  -3.55935    1.75195  -2.032 0.042631 *  
## speechiness                        7.16508    3.57780   2.003 0.045667 *  
## playlist_genrelatin                1.70038   10.93737   0.155 0.876507    
## playlist_genrepop                 -3.23550    8.49464  -0.381 0.703422    
## playlist_genrer&b                 -3.43873   10.86654  -0.316 0.751771    
## playlist_genrerap                -23.87764    8.43407  -2.831 0.004795 ** 
## playlist_genrerock               -33.02745   11.66982  -2.830 0.004809 ** 
## duration                          -0.02569    0.02110  -1.218 0.223827    
## danceability:playlist_genrelatin  13.36542    9.98035   1.339 0.181024    
## danceability:playlist_genrepop    11.41965    8.12791   1.405 0.160544    
## danceability:playlist_genrer&b    21.17955   10.44690   2.027 0.043070 *  
## danceability:playlist_genrerap    28.61977    8.28703   3.454 0.000592 ***
## danceability:playlist_genrerock   21.35475   12.10583   1.764 0.078242 .  
## loudness:playlist_genrelatin       1.09402    0.49668   2.203 0.028001 *  
## loudness:playlist_genrepop         0.43047    0.38780   1.110 0.267432    
## loudness:playlist_genrer&b         0.95273    0.52218   1.825 0.068572 .  
## loudness:playlist_genrerap        -0.04126    0.39600  -0.104 0.917045    
## loudness:playlist_genrerock       -0.31464    0.59679  -0.527 0.598233    
## playlist_genrelatin:duration      -0.01040    0.03134  -0.332 0.740007    
## playlist_genrepop:duration         0.01758    0.02844   0.618 0.536542    
## playlist_genrer&b:duration        -0.02335    0.04355  -0.536 0.592105    
## playlist_genrerap:duration         0.05220    0.02608   2.002 0.045760 *  
## playlist_genrerock:duration        0.09705    0.04066   2.387 0.017308 *  
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 8.569 on 597 degrees of freedom
## Multiple R-squared:  0.2371, Adjusted R-squared:  0.2039 
## F-statistic: 7.136 on 26 and 597 DF,  p-value: < 2.2e-16
```

With the BoxCox transformation, the model fit seems to have improved ever so slightly, with multiple R^2 going to 0.24 and adj R^2 going to 0.207. However, it is a not a very significant improvement and the residual plots, which can be run using the code below are largely identical to the residual plots we had with the untransformed model.

Conclusion

Overall, it is clear that there are plenty of interesting relationships between the track metrics recorded in this dataset. Particularly, I found that `energy` and `loudness` have a strong, positive, linear relationship with each other, in fact the strongest relationship between variables in this dataset. I also found that the most popular genre is pop and that tracks tends to become more popular as their danceability increases. The R-squared values for some of our linear models predicting variables like danceability are not as high as we would have hoped, but we can propose two main reasons for this: 1.) Danceability may be a difficult vairable to predict in general and 2.) There may exist other variables not contained in this dataset that are better predictors of danceability. This is a true refelction of statistics and data analysis in the real world as data is not always as neat and tidy as we would like it to be, but through creative data cleaning, analysis, and exploration, we can produce meaningful information from our data.