What Makes Songs Danceable: An Analysis of Popular Tracks from 2000-2019

Haram Yoon and Shane Foster Smith

Introduction

Dataset: Paradisejoy. (n.d.). Top Hits Spotify from 2000-2019. Kaggle. Retrieved from https://www.kaggle.com/datasets/paradisejoy/top-hits-spotify-from-20002019

The Spotify songs dataset contains various audio features and metadata on over 2000 popular tracks on the platform from 2000 to 2019. The primary goal of our study is to investigate the factors that influence the "danceability" of a song. Danceability is the measure of how suitable a track is for dancing based on a combination of musical elements.

Our motivation behind analyzing danceability specifically from the fact that highly danceable songs tend to be more widely appreciated, particularly in certain genres like pop, electronic dance music (EDM), and hip-hop. We first thought of predicting popularity of a songs, but that soon turned out to be less feasible and valuable compared to danceability. Furthermore, danceable tracks tend to be more enjoyable for listeners due to their higher energy level, a strong rhythm, and a tempo that aligns with typical dancing speeds. These danceable tracks are espeically popular in nightclubs, parties, and other social settings.

From a commercial perspective, creating danceable hits can be highly beneficial for artists and music labels. Tracks with high danceability tend to perform well on music streaming platforms which make them more likely to be added to playlists. Furthermore, danceable songs often stay popular for longer, as they remain relevant for dance-related events and activities over an extended period.

By analyzing the factors that contribute to danceability, our study aims to provide insights that can guide music producers and artists in crafting more danceable tracks tailored to the preferences of their target audience. The findings could also be valuable for music streaming platforms, radio stations, and other music curators in identifying and promoting highly danceable tracks, thereby tailoring a song to a particular social setting.

In this dataset, the response variable of interest is the "danceability" score, which ranges from 0 to 1, with higher values indicating more danceable tracks. The predictor variables include both numerical features related to the audio characteristics of the songs (e.g., energy, valence, tempo, acousticness, speechiness) and categorical variables like genre, mode and explicit content.

The original data was collected by Spotify, a popular music streaming service, and likely consists of tracks that were popular or frequently streamed on the platform during the given time period. However, the specific details on how the data was collected and processed are not provided.

The dataset includes 2000 observations (songs) and 18 variables, including the artist name, song title, duration, year of release, popularity score, and various audio features computed by Spotify's algorithms. These audio features, such as danceability, energy, valence, and acousticness, are numerical values ranging from 0 to 1 and are designed to capture different aspects of the audio content.

It's worth noting that some variables in the dataset may require further processing or transformation before being included in the statistical models. For instance, the "genre" variable contains multiple genres for some songs, and the "explicit" variable is binary, indicating whether the song contains explicit content. Variables

like “acousticness” and “speechiness” may also need to be explored further to understand their potential impact on danceability.

Moreover, interactions between variables could be important in predicting danceability. For example, the effect of tempo on danceability might vary across different genres, or the influence of energy on danceability could be moderated by the explicit content of a song.

By carefully analyzing and preprocessing these variables, as well as considering potential interactions and transformations, this analysis aims to develop effective statistical models that can accurately predict the danceability of a song based on its audio features and metadata.

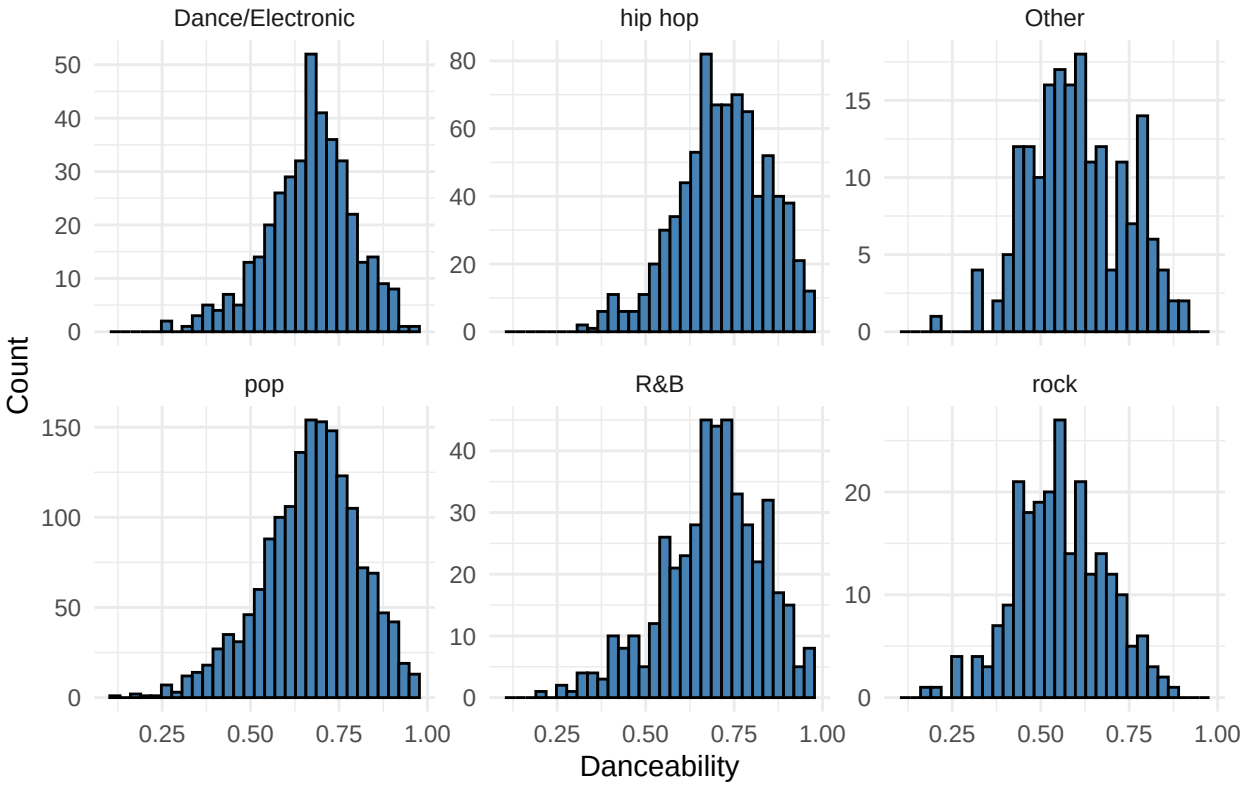
By analyzing this dataset, we aim to identify the key factors that contribute to a song's danceability and develop statistical models that can potentially predict this measure based on the available audio features and metadata. The findings of this analysis could provide valuable insights for music producers, artists, and streaming platforms to create and promote more danceable tracks, catering to the preferences of their audience.

Feature Engineering

To preprocess the data, the first step is to deal with any missing or NA data. Thankfully, there was none. Our next step was to change some of the categorical variables into usable datatypes in our models (binary representations or integers rather than booleans or characters). For example, the ‘Genre’ feature of a song was a string that could contain one or more genres out of the 14 possible genres(such as, “pop, hip hop”). To make this feature usable in our model, we performed two operations; first, we coalesced the nine least frequency genres into a single “Other” genre, which important to ensure dimensionality reduction and because these genres were very infrequent compared to the top five most frequent genres. After, we performed a one-hot encoding of the ‘Genre’ feature (into 6 binary features), which indicates if a song is part of one or more genres.
Second, most of the original features have values in the range [0,1], but a few features had values much larger or in negative ranges. We normalized some of these features' values, while also keeping into consideration the interpretability of our models' units. For example, two of the variables we normalized where ‘duration_ms’ and ‘loudness’. ‘Duration_ms’ original feature values where in the approximate range of [100000,400000] milliseconds and ‘Loudness’ had feature values in the approximate range of [-20,0] decibels. To normalize these values, we first determined that the original units weren't particularly informative (what really is a one dB increase of loudness? Or, why is a millisecond differnce is length important?). Instead, we decided to use the scale() function so that new units are standard deviations from the mean. How loud a song is relative to other songs in our dataset is more informative. Similarly, how long songs are relative to other songs is more informative.

1. Preprocessing and Handling Important Categorical Variables

Figure 1: Danceability Distribution by Genre



By observation of the **Figure 1**, it appears that the frequency of danceability does slightly vary depending on the genre. For example, 'rock' and 'Other' peaks (or means) are slightly lower in the histograms than they are for 'Dance/Electronic' and 'Hip Hop'. Also, for 'Pop' and 'R&B', we can see that their songs are more frequenty in the [0,0.5] danceability range than songs in ''Dance/Electronic' and 'Hip Hop'. These results indicate that dance/electronic and hop hip songs may be considered more danceable, and perhaps deserve special consideration in our models.

Genre Counts

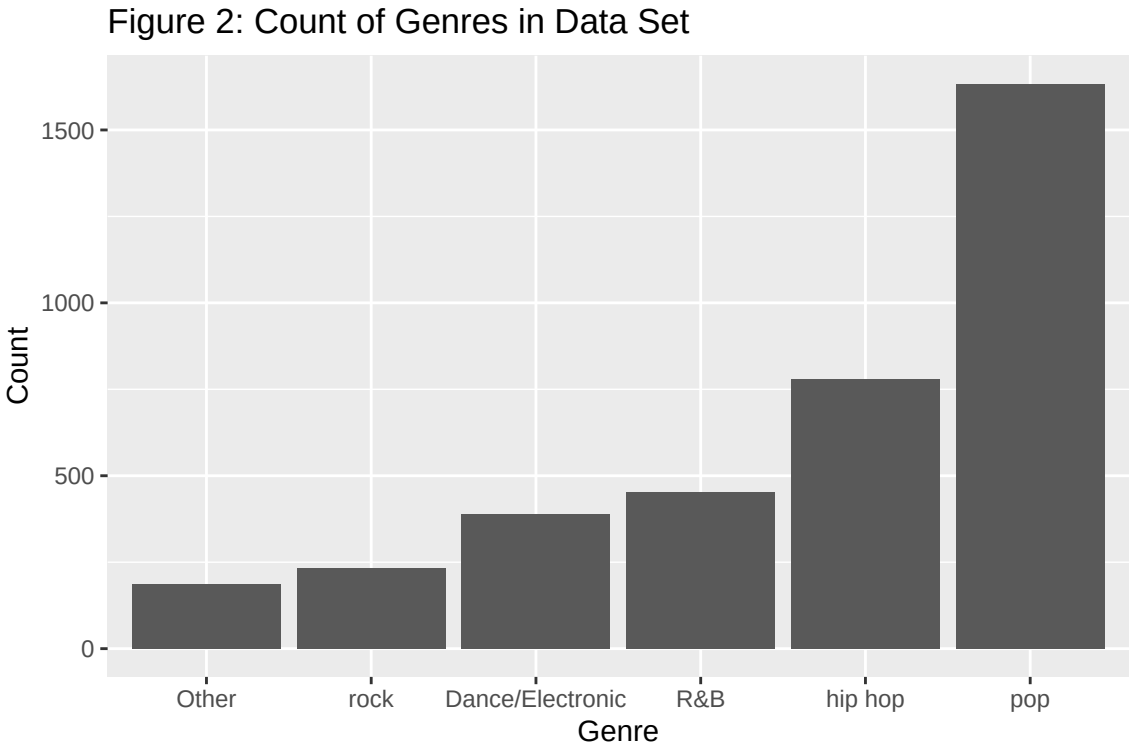
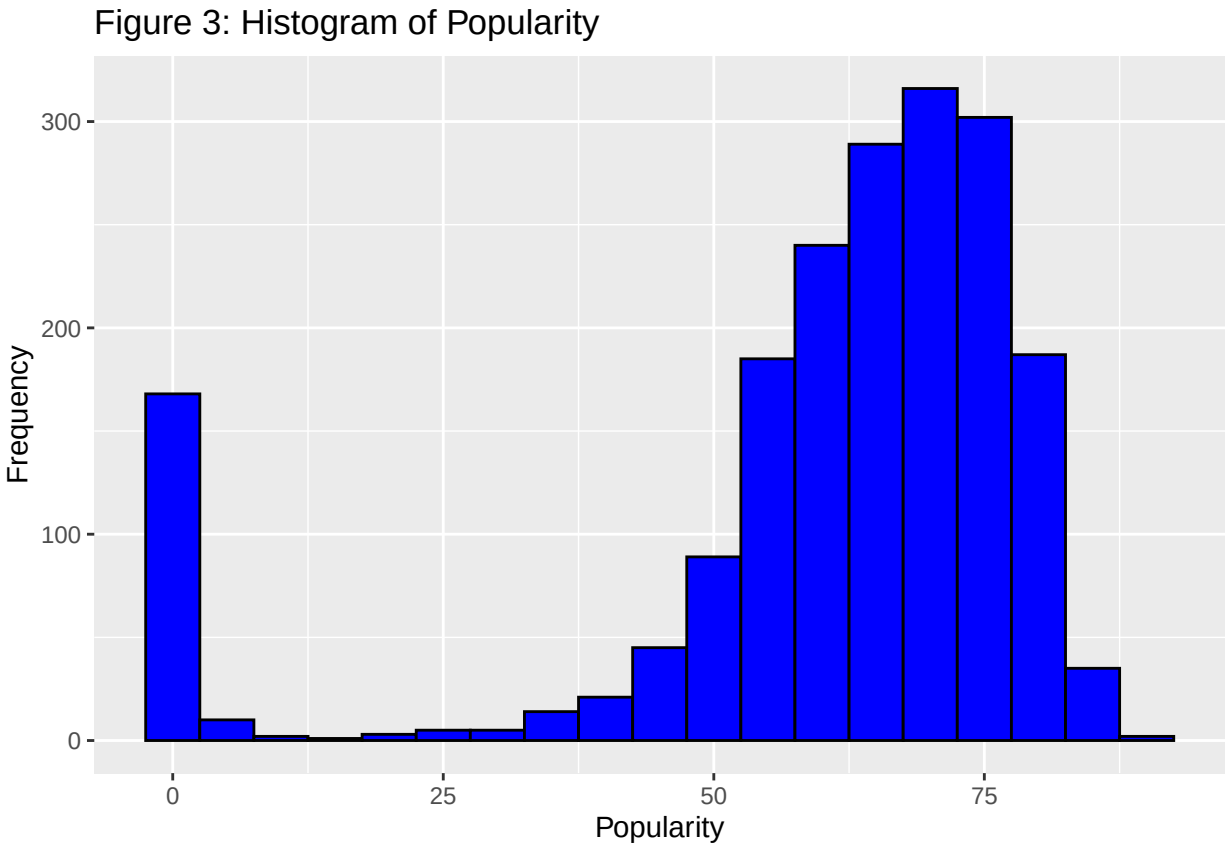


Figure 2 shows 'pop' as the most represented genre in the dataset, followed closely by 'hip hop'. Given the prominence of these highly danceable genres, the dataset is well-suited for analyzing the factors contributing to danceability across different musical styles. Furthermore, most of the songs in our data set are considered to be 'pop' (which makes sense, as this is data set of popular songs from 2000-2019) and lots of songs are considered to be pop *and* some other genre.

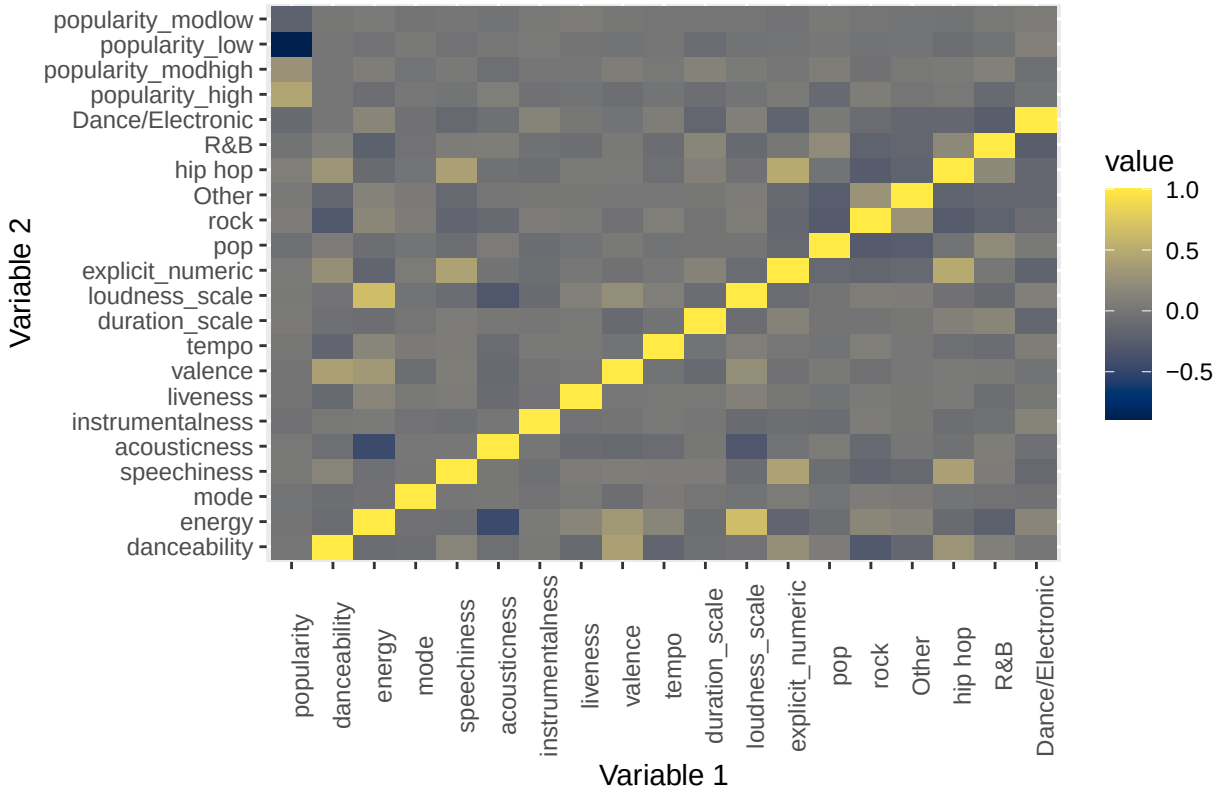
Popularity



Popularity, as seen above, is binomially split between songs that have 0 or close to 0 popularity and those who have moderate to high popularity. Due to this fact, and because we found that the values of popularity weren't very correlated with our response, we decided perform a one hot encoding of popularity (aka. popularity is now a categorical variable, represented by four binary features: low popularity [0,25), moderate-low popularity [25, 50), moderate_high [50,75) and high popularity [75,100]).

2.Correlation of Potential Predictors

Figure 4: Correlation Heatmap



This plot shows the correlation between each numeric variable in our dataset (as a matrix heatmap). The darker colors show negative correlation while the warmer colors (more yellow) show more positive correlation. We can see that many variable in the heatmap are notably correlated (either positively or negatively) .
For example, the variable 'Energy' is positively correlated with loudness and very negatively correlated what acoustiness. Futhermore, loudess and acoustiness, unsurprisingly, are also very negative correlated. So, to avoid multicollinearity, we likely don't want to put more than one of these variables in our model without careful consideration.
Another example is, 'Speechiness', 'Explicit_numeric', and 'Hip Hop' are all positively correlated. This makes intuitive sense as we expect hip hop to be more 'wordy' and likely contain more explicit language. Again, we have to be careful with including more than one of these variables in a model, considering their correlation. Surprisingly, the popularity variables do not appear to be very correlated with any of the other variables.

To get a closer understanding of the correlations and relationships of our variables, we constructed a few pairs plots and considered some transformations of our variables.

Pairs Plots

Figure 5a: Pairwise Plot of Song Features

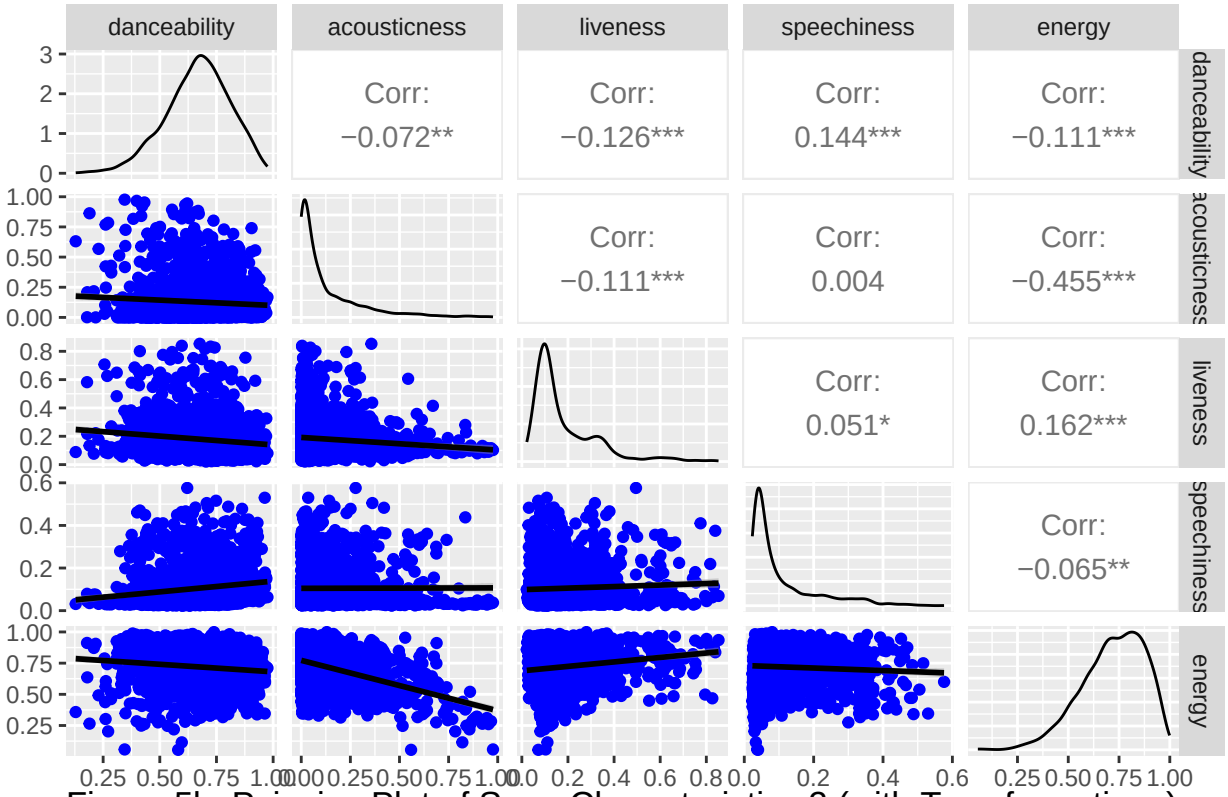


Figure 5b: Pairwise Plot of Song Characteristics 2 (with Transformations)

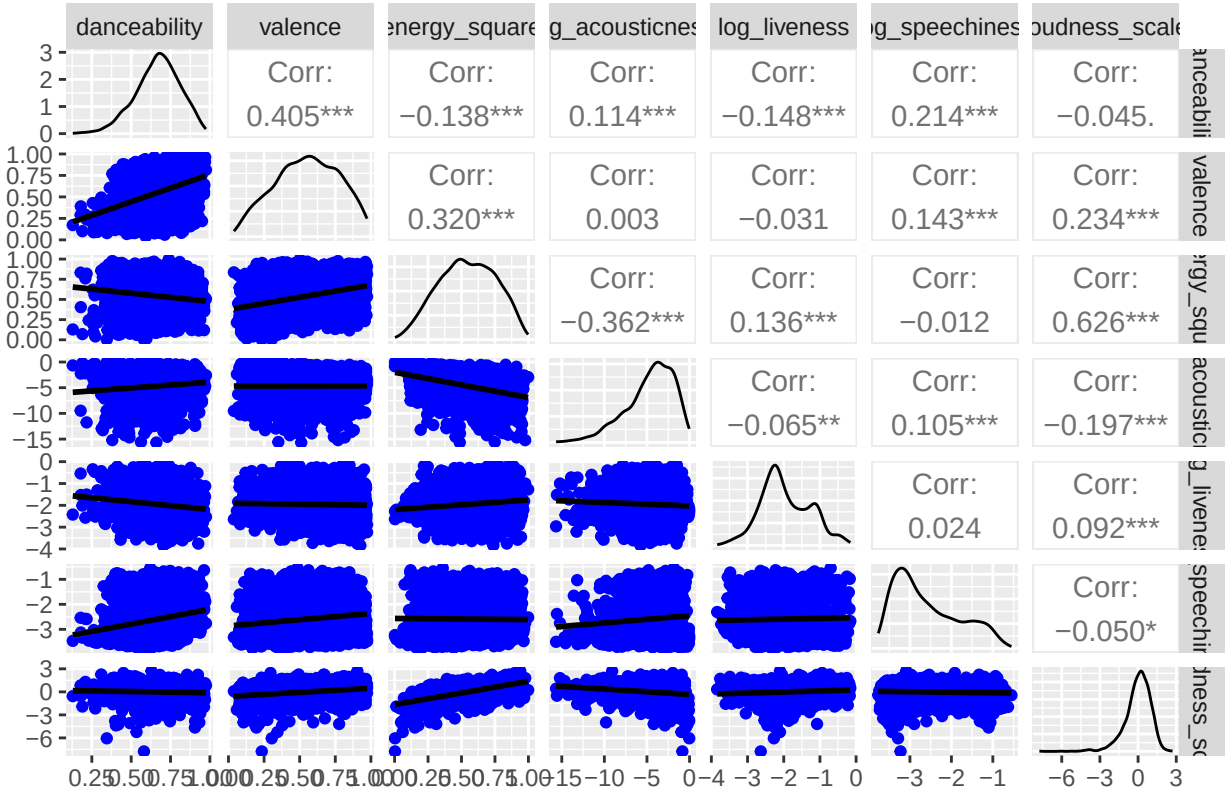
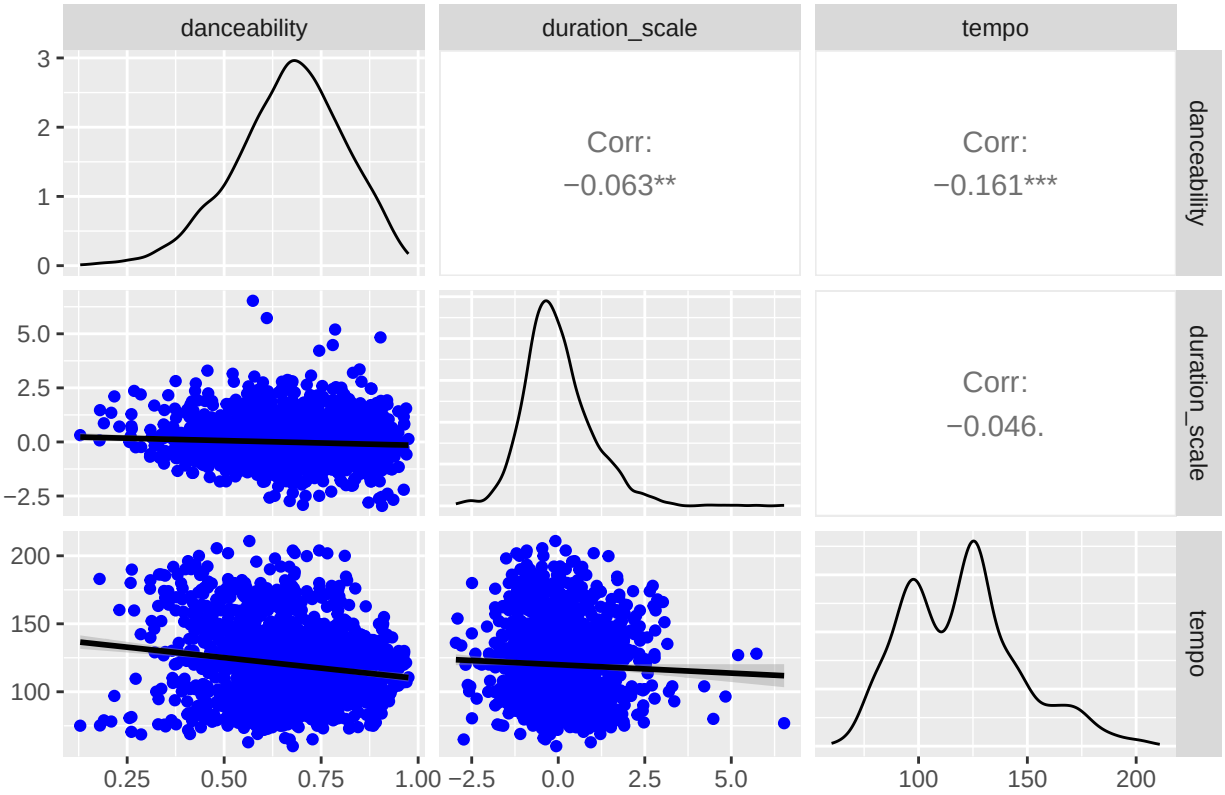


Figure 5c: Pairwise Plot of Song Characteristics 3

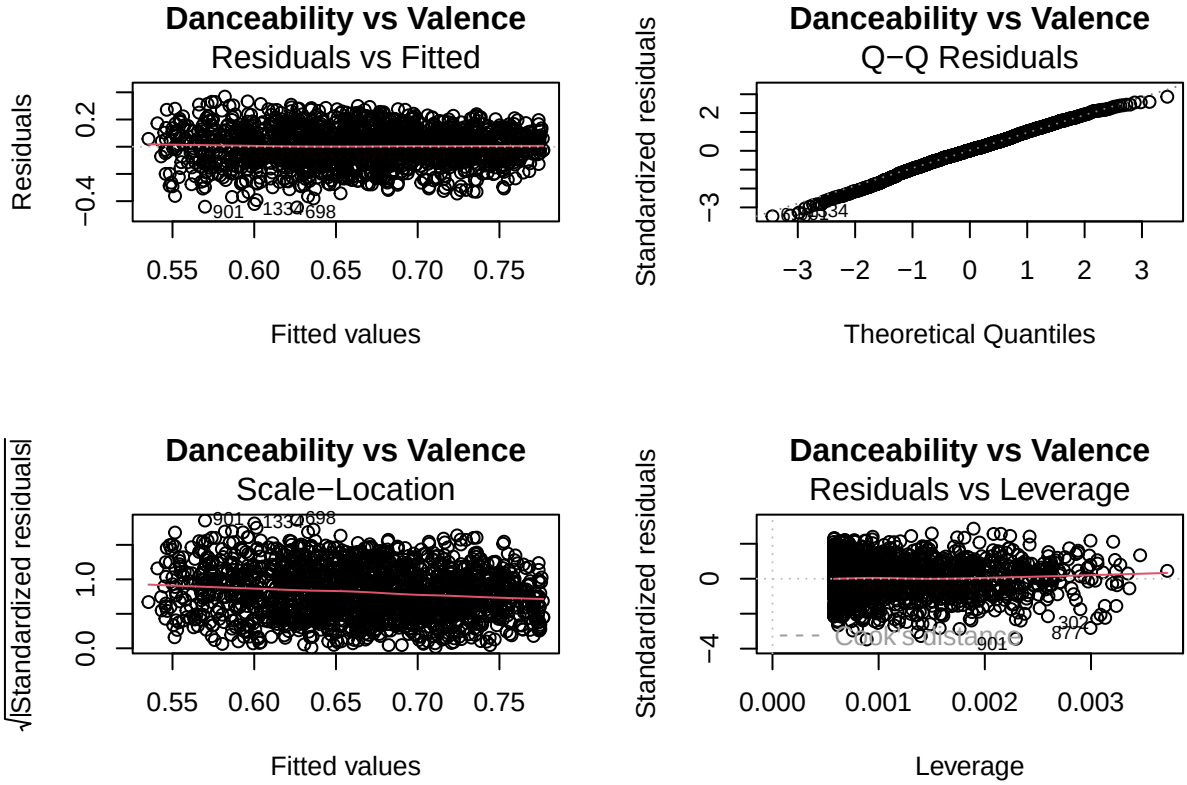


From the pairs plots, it's evident that danceability exhibits varying degrees of correlation with several audio features. Notably, it shows moderate positive correlations with valence (0.4) and log_speechiness (0.208), suggesting that higher danceability tends to be associated with higher valence and speechiness levels, respectively. Conversely, danceability demonstrates weak negative correlations with variables such as acousticness (-0.078), energy (-0.1), and tempo (-0.184), implying that higher danceability may be associated with lower acousticness, energy, and tempo values. However, the correlations with loudness_scale, duration_scale, sqrt_instrumental, and popularity appear to be negligible, indicating minimal linear relationships between these variables and danceability. Even though certain variables exhibit negligible correlations with danceability, it's important to still consider them for model building due to their potential non-linear relationships, contextual importance, and role in mitigating multicollinearity to help create a more comprehensive understanding of the factors influencing danceability. Furthermore, we see evidence of non-constant variance, non-linearity and possible violations of the assumption of normal errors for certain predictors (in particular, look at first pair-wise plot of characteristics).

3. Transformations:

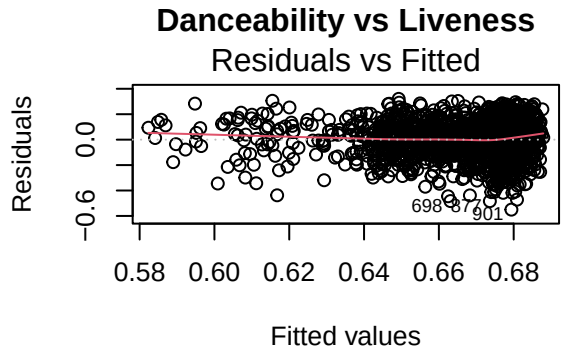
In our analysis, we applied logarithmic and square root transformations to certain variables to address the conditions of a normal model. These transformations helped to stabilize the variance of residuals and achieve a more symmetric distribution, thereby enhancing the adherence of our models to the assumptions of linear regression. Through these transformations, we aimed to improve the reliability and interpretability of our regression analyses.

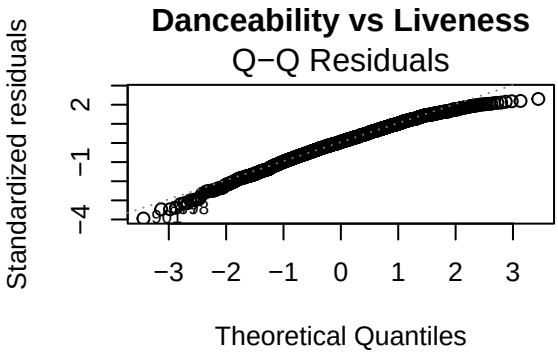
No Transformations

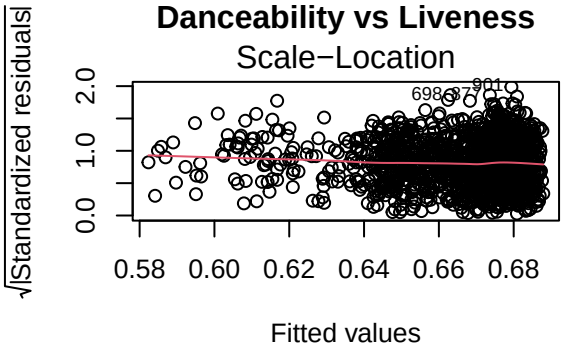


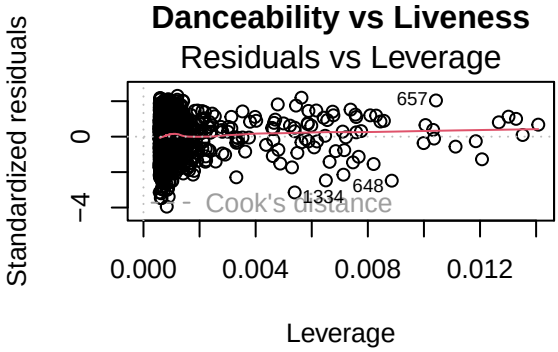
In cases where variables meet the model conditions based on diagnostic plots, such as with valance, seen above, transformations are not necessary as the assumptions of linearity, normality, and constant variance are adequately satisfied. Therefore, applying transformations to these variables may introduce unnecessary complexity (while also losing interpretability) without yielding significant improvement in model performance. However, most of our continuous variable did require transformations in order to meet the normal errors model assumptions.

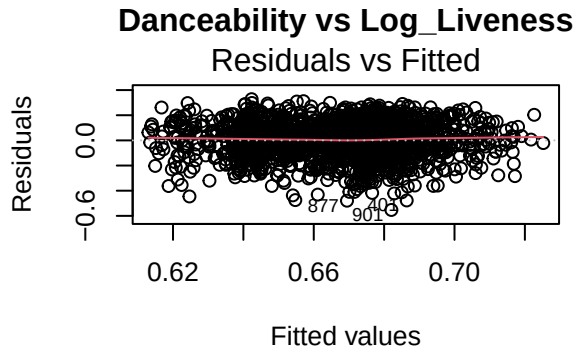
Log Transformations

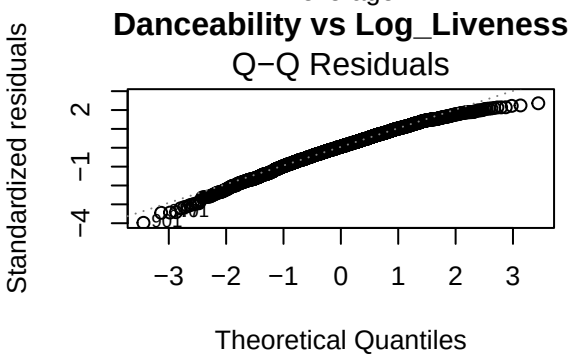


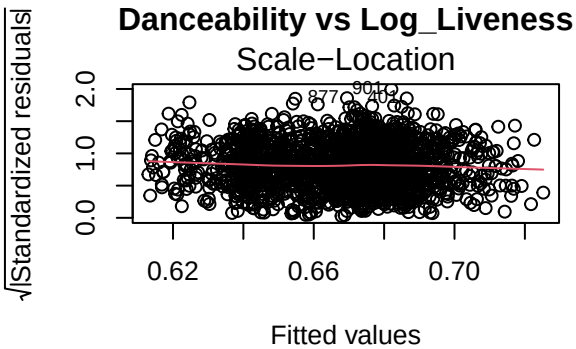


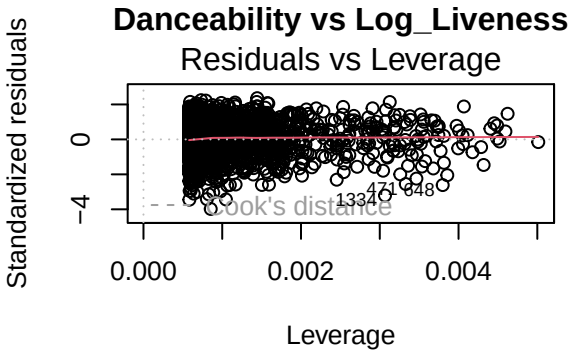


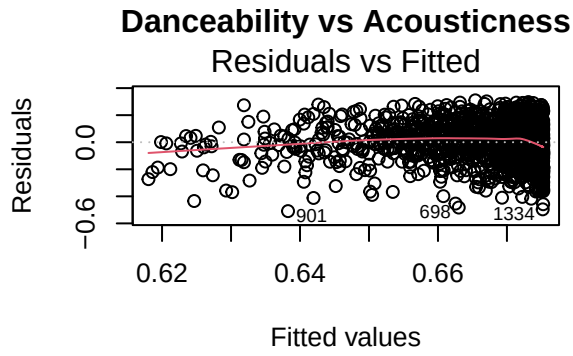


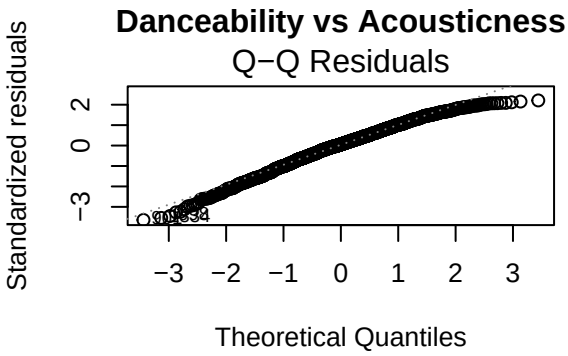


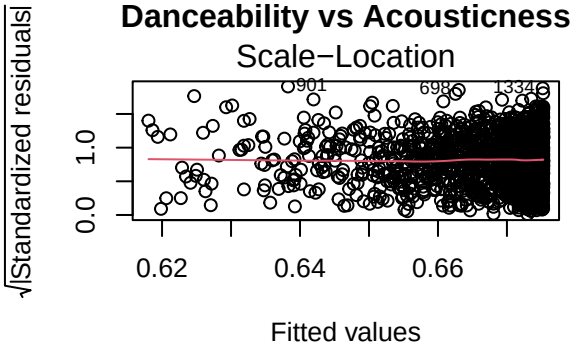


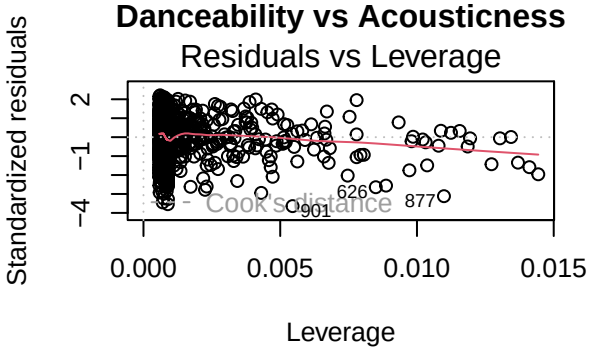


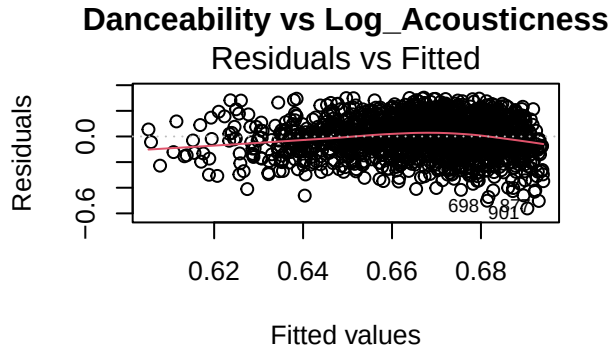


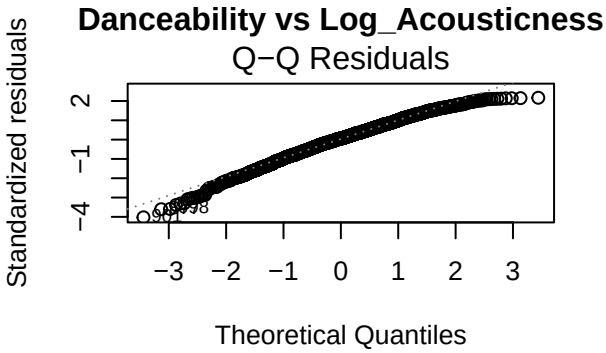

Danceability vs Acousticness
Residuals vs Fitted
Residuals
Fitted values
901
698
1334

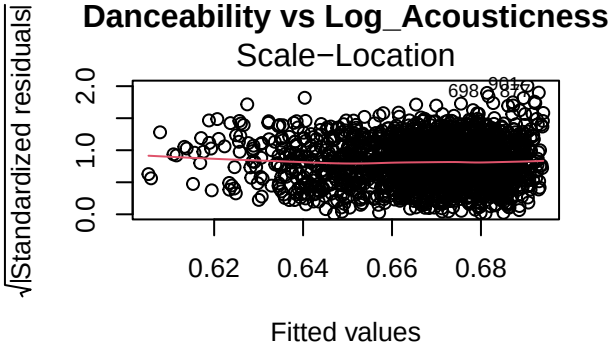

Danceability vs Acousticness
Q−Q Residuals
Standardized residuals
Theoretical Quantiles

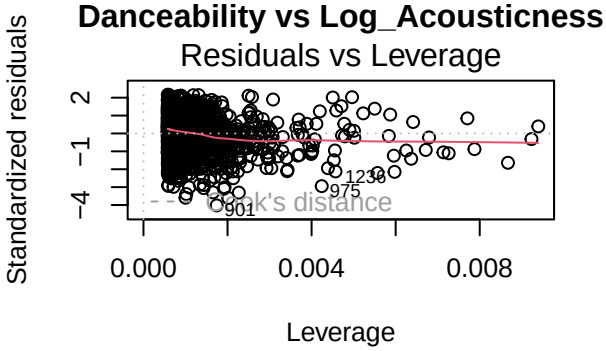

Danceability vs Acousticness
Scale−Location
√|Standardized residuals|
Fitted values
901
698
1334

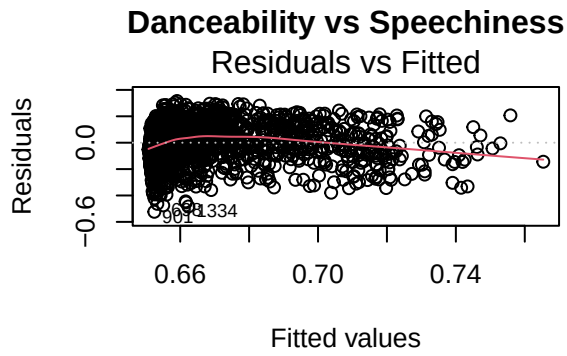

Danceability vs Acousticness
Residuals vs Leverage
Standardized residuals
Cook's distance
Leverage
626
877
901

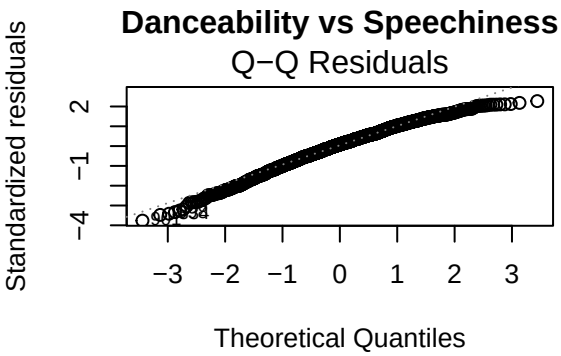

Danceability vs Log_Acousticness
Residuals vs Fitted
Residuals
Fitted values
698
901

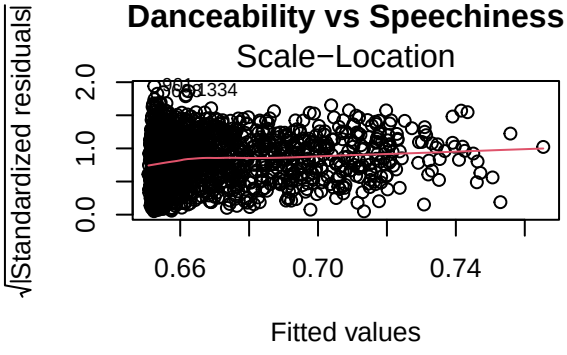

Danceability vs Log_Acousticness
Q−Q Residuals
Standardized residuals
Theoretical Quantiles

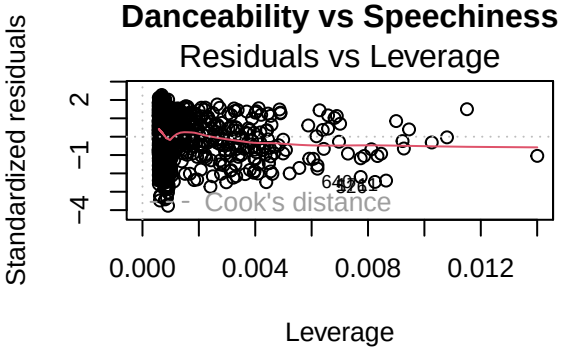

Danceability vs Log_Acousticness
Scale−Location
√|Standardized residuals|
Fitted values
698
901

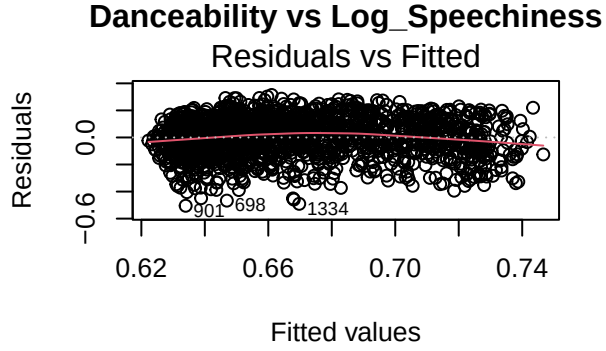

Danceability vs Log_Acousticness
Residuals vs Leverage
Standardized residuals
Cook's distance
Leverage
975
901

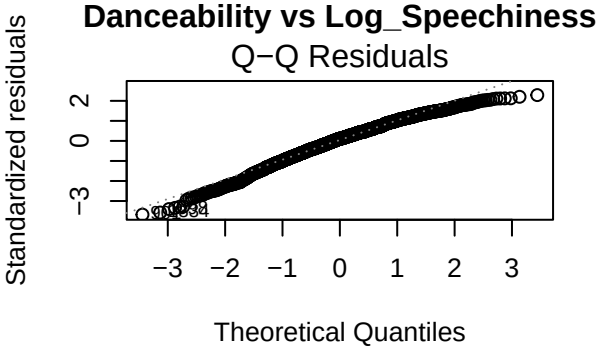

Danceability vs Speechiness
Residuals vs Fitted
Residuals
Fitted values

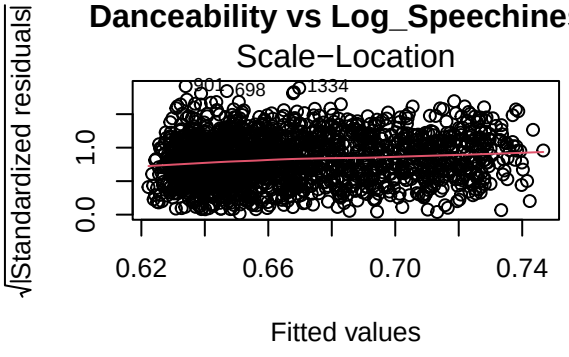

Danceability vs Speechiness
Q−Q Residuals
Standardized residuals
Theoretical Quantiles

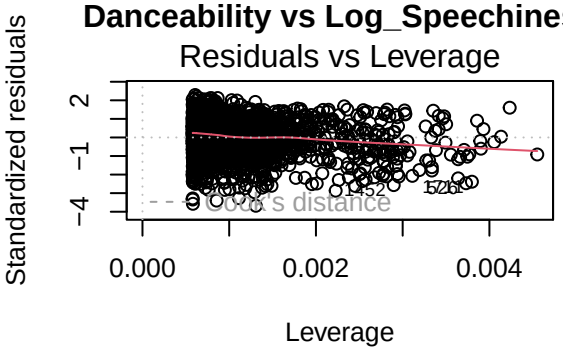

Danceability vs Speechiness
Scale−Location
√|Standardized residuals|
Fitted values

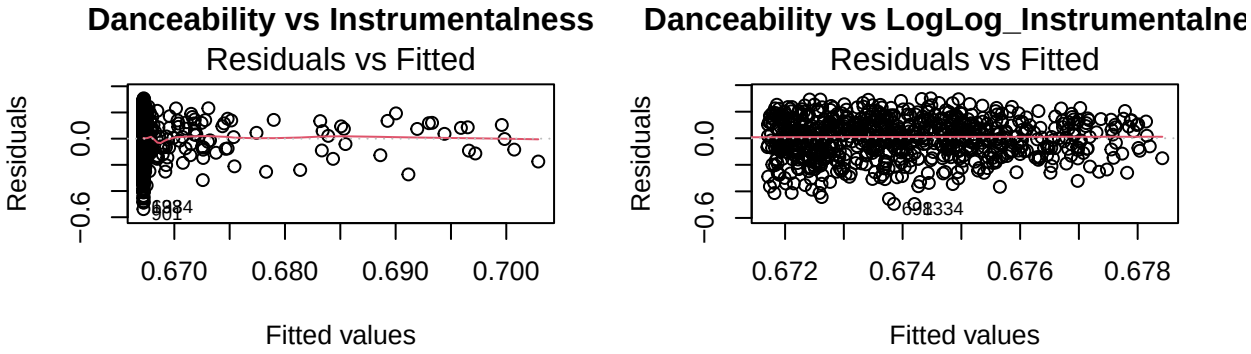

Danceability vs Speechiness
Residuals vs Leverage
Standardized residuals
Cook's distance
Leverage


Danceability vs Log_Speechiness
Residuals vs Fitted
Residuals
Fitted values


Danceability vs Log_Speechiness
Q−Q Residuals
Standardized residuals
Theoretical Quantiles

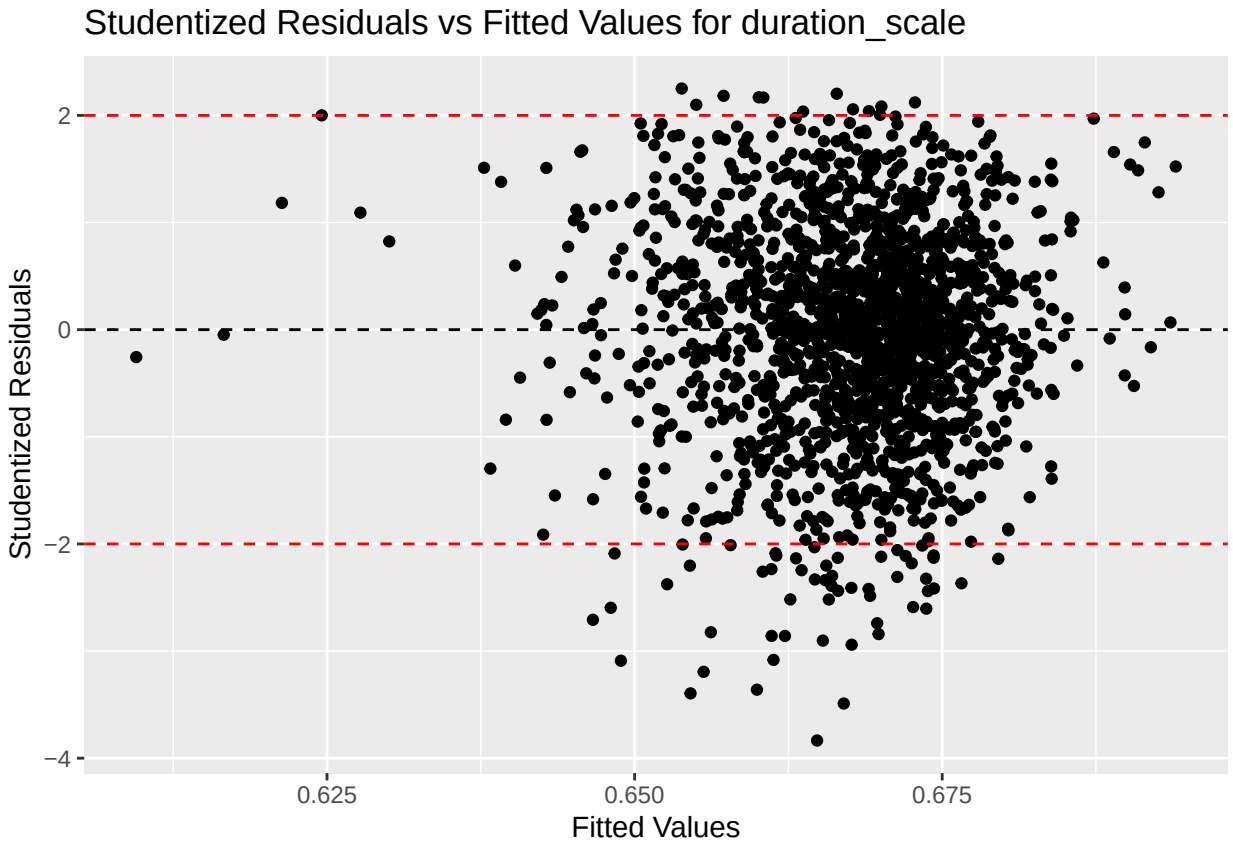

Danceability vs Log_Speechiness
Scale−Location
√|Standardized residuals|
Fitted values

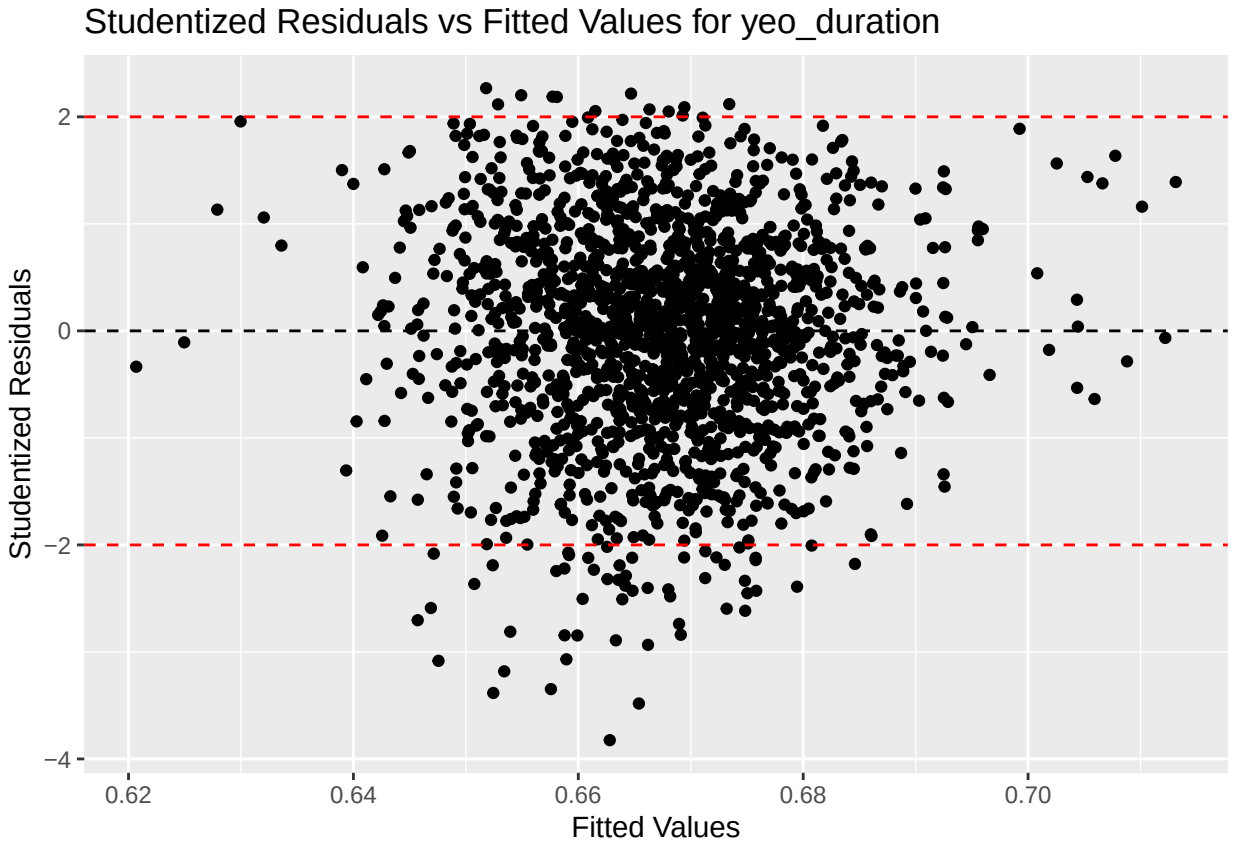

Danceability vs Log_Speechiness
Residuals vs Leverage
Standardized residuals
Cook's distance
Leverage

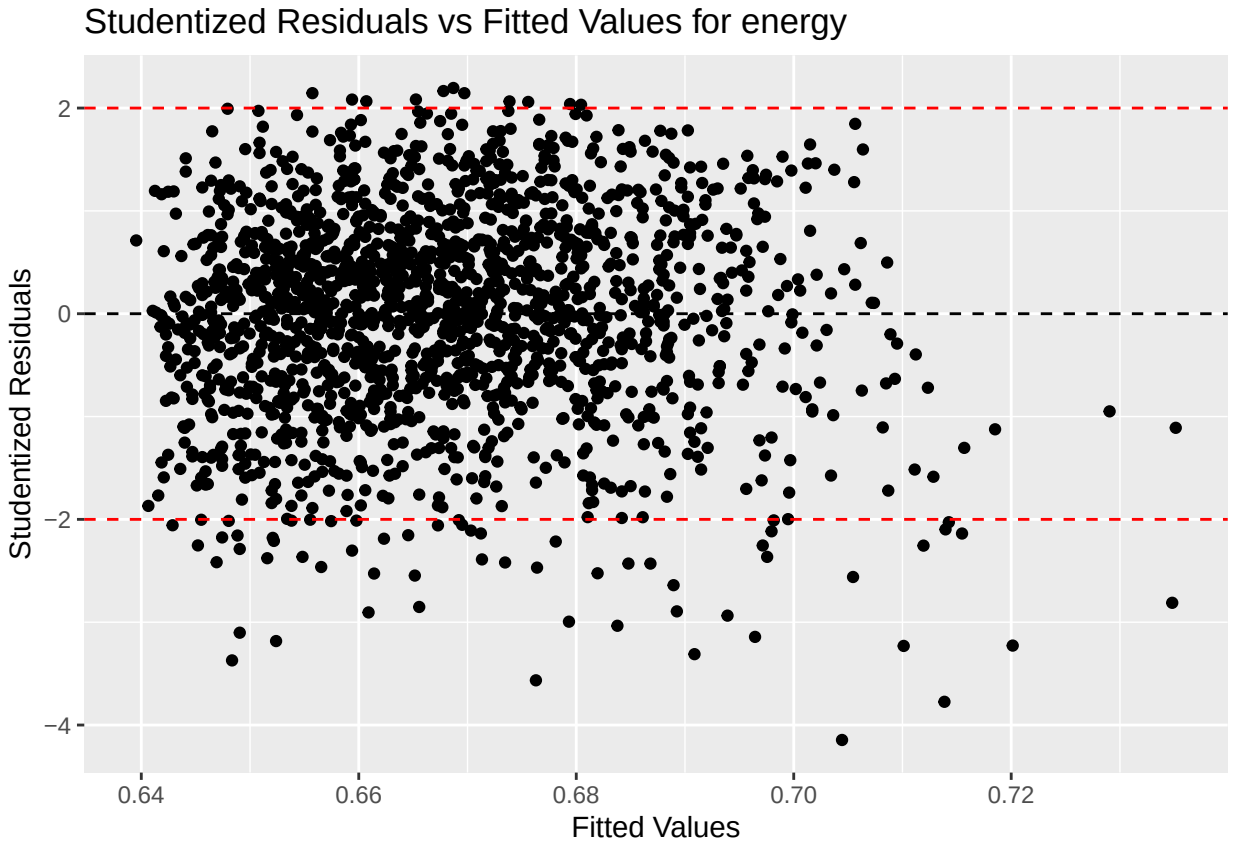


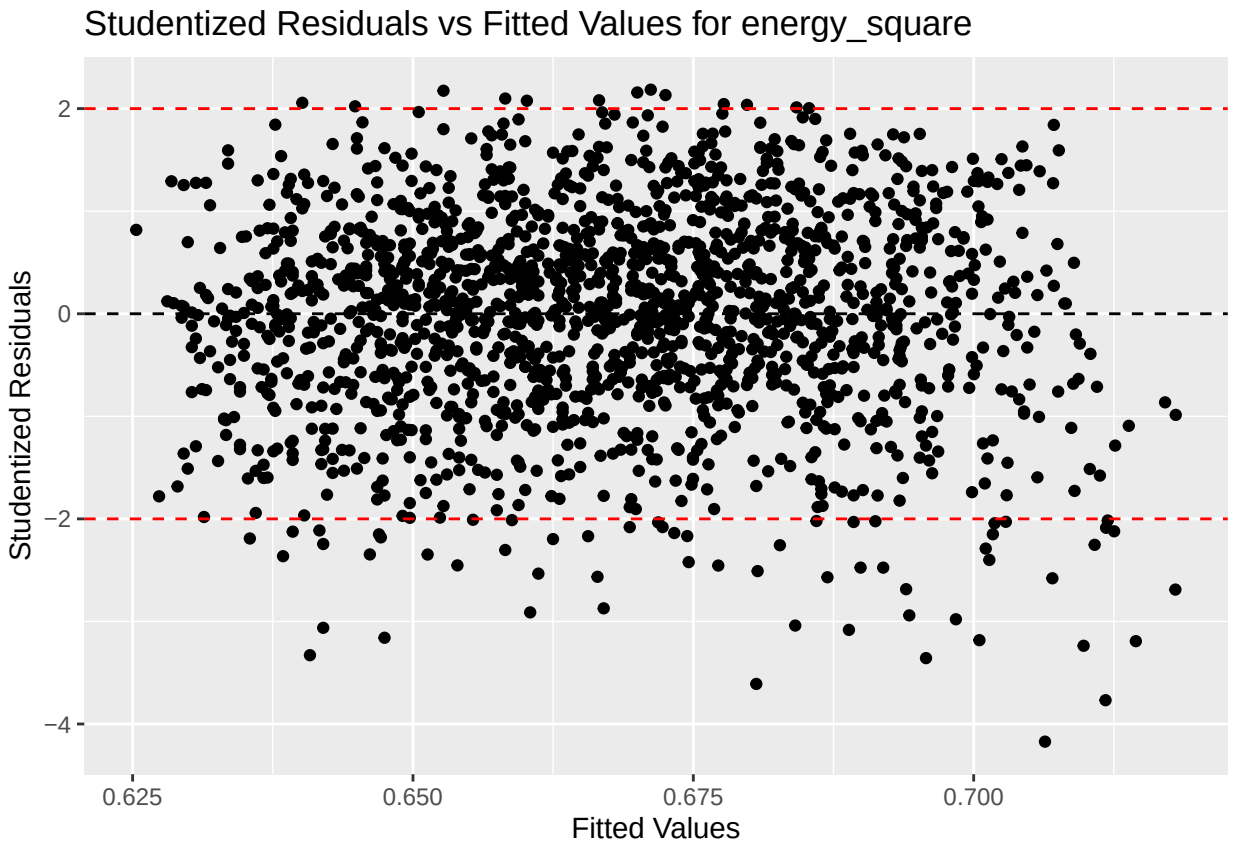
We encountered skewed distributions (both left and right), non-linearity, and heteroscedasticity in many of the predictor variables, as evidenced by the diagnostic plots above. To address these issues, we opted to apply a logarithmic transformation to these variables. After completing the transformations, we observed a marked improvement in the diagnostic plots; the previously skewed residual vs. fitted plots now display distributions that have a more uniform variance and linear relationship with the response variable. However, some of the residual plots still looked slighly quadtradic, which we will address later in this paper. In summary, the logarithmic transformation of skewed predictor variables yielded significant improvements in the linearity and homoscedasticity of our regression models, thereby enhancing the inference and predictive capabilities of these variables in a linear model.

Quadratic Transformations







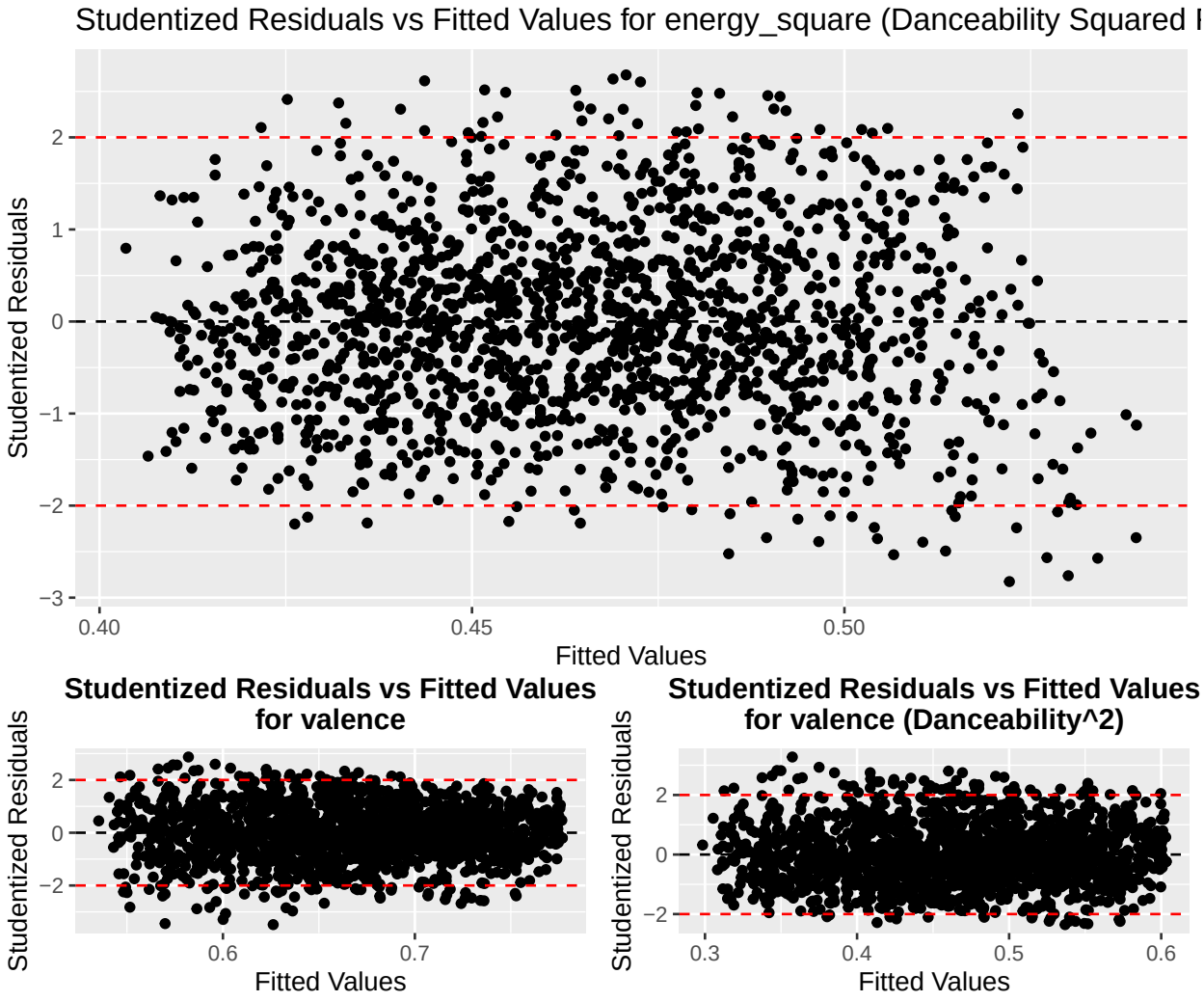


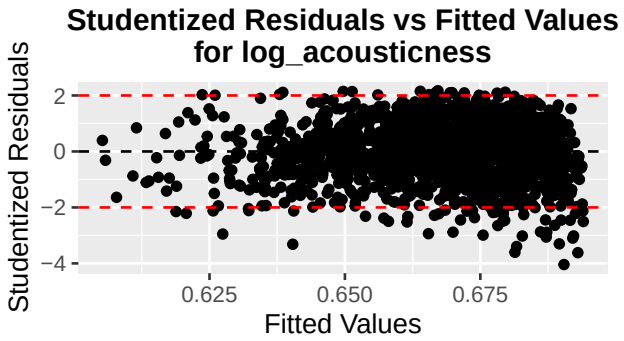
We improved the linearity and constant variance of the 'duration_scale' predictor ... Also, the previous

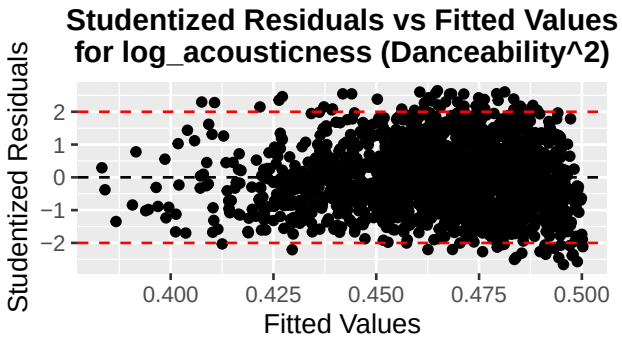
residuals vs fitted values for the 'energy' predictor looked slightly quadtradic, and we confirmed this suspicion by looking at the studentized residuals vs fitted values plots. As seen above, when we don't square 'energy', the studentized residuals heavily favor the fitted values far below the mean (rather than above the mean), especially for large fitted values; this indicates non-linearity. When we look at the same plot, but for a linear model that squares 'energy', we see that there is still a bias towards fitted values below the mean, but it is reduced and there is an improvement in constant variance.

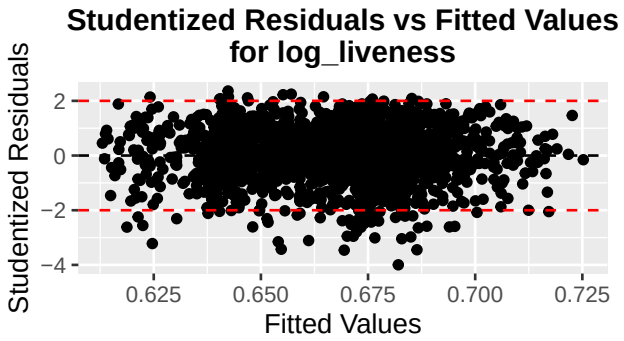
Even after many of the previous transformations on the predictors, we still noticed a bias toward studentized residuals below the mean (sometimes more than 2 standard deviations). Therefore, we decided to transform the response variable, 'danceability', into a quadtradic term. Below, you will see various studentized residuals vs fitted values plots that show further improvements to various predictors when we transform 'danceability' to a squared term (in terms of constant variance, linearity, and, in particular, normal errors).

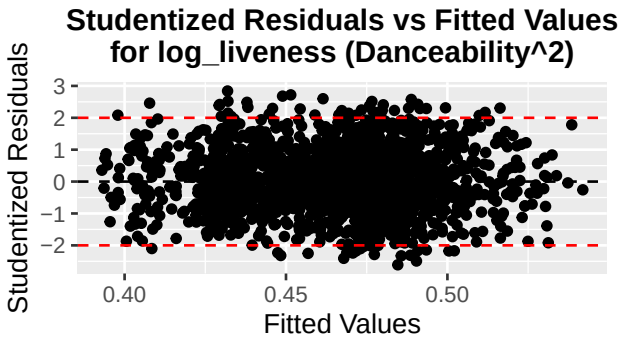
Although transforming the response variable in this way makes the interprebility of our model more nuanced, the studentized residual plots show that this tradeoff is acceptable due to the significant improvement in the assumptions of a normal errors model. With these transformation, we feel more confident in the validity of the inference and predictions we perform later in this paper.

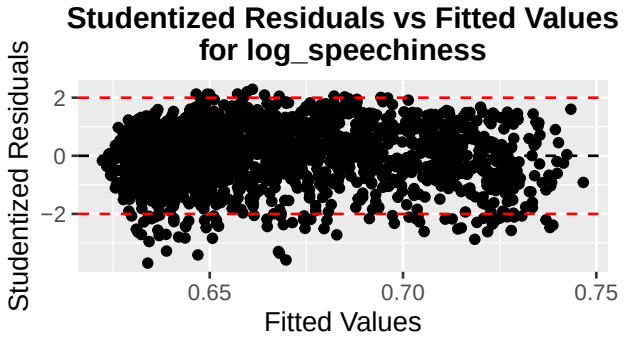


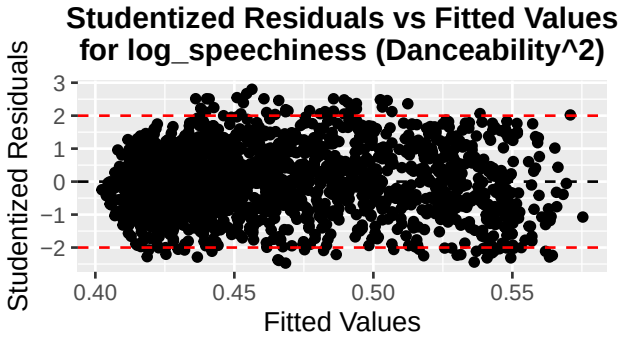










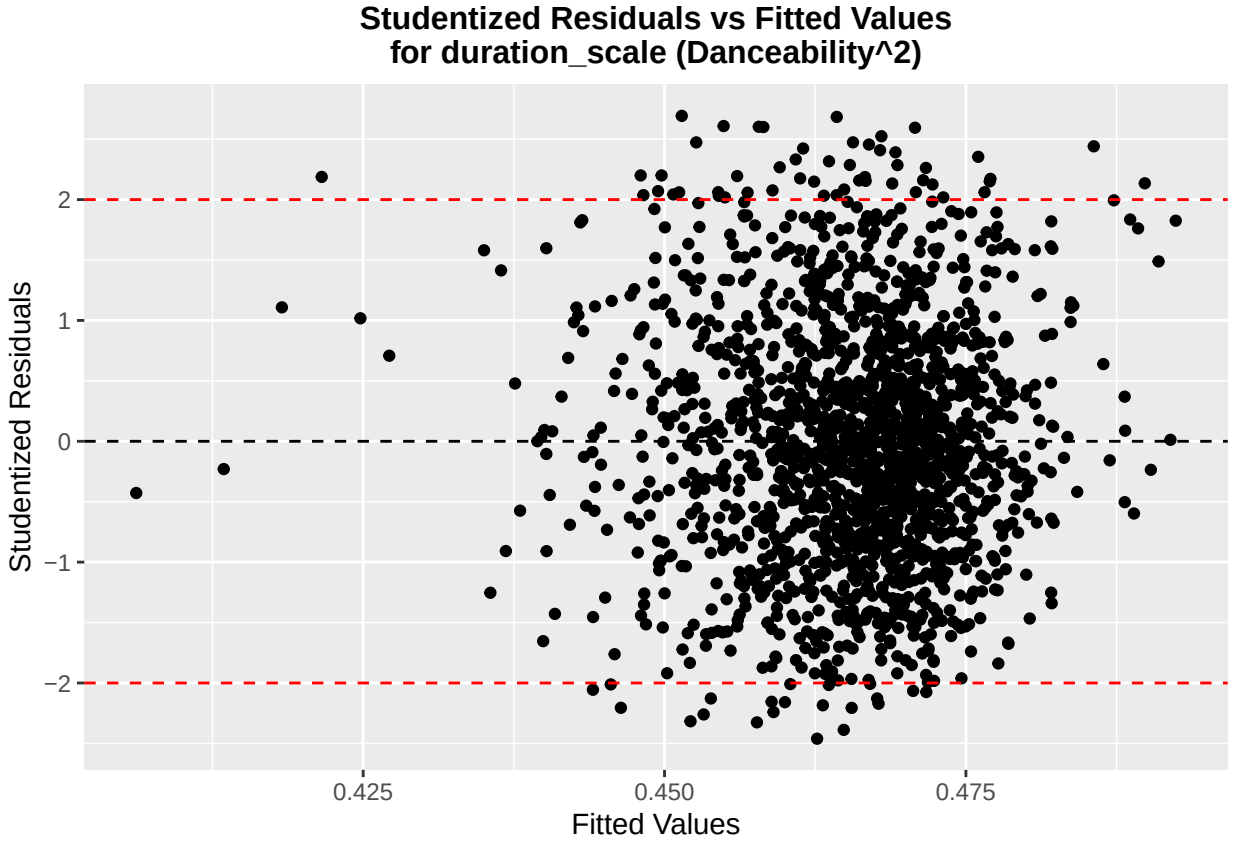


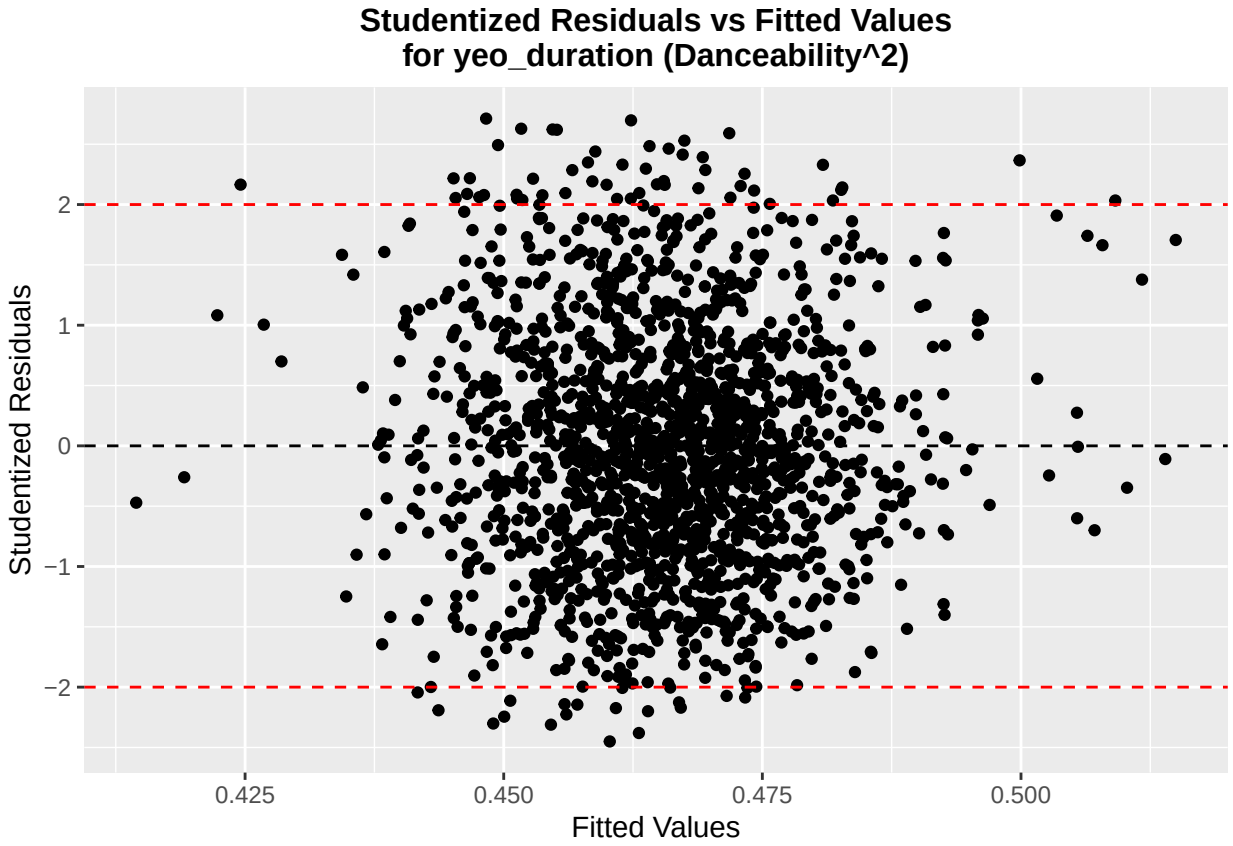

```
## [[1]]
## TableGrob (2 x 2) "arrange": 2 grobs
##   z     cells    name           grob
## 1 1 (1-1,1-1) arrange gtable[layout]
## 2 2 (1-1,2-2) arrange gtable[layout]
##
## [[2]]
## TableGrob (2 x 2) "arrange": 2 grobs
##   z     cells    name           grob
## 1 1 (1-1,1-1) arrange gtable[layout]
## 2 2 (1-1,2-2) arrange gtable[layout]
##
## [[3]]
## TableGrob (2 x 2) "arrange": 2 grobs
##   z     cells    name           grob
## 1 1 (1-1,1-1) arrange gtable[layout]
## 2 2 (1-1,2-2) arrange gtable[layout]
##
## [[4]]
## TableGrob (2 x 2) "arrange": 2 grobs
##   z     cells    name           grob
```

```
## 1 1 (1-1,1-1) arrange gtable[layout]
## 2 2 (1-1,2-2) arrange gtable[layout]
```

Yeo-Johnson Transformations

For the predictor 'duration_scale', we tried a variety of log, inverse and quadtric transformations for the purpose of stabilizing its variance (see figure . . .) . However, we found that none of these methods worked as intended. Therefore, we decided to do some research into other types of transformation that may be able to help us. A Yeo-Johnson Transformation is an extension of the Box-Cox transformation; Yeo-Johnson can handle feature values, while Box-Cot transformations can't. This type of transformation is very useful for stabilizating the variance of a predictor. Information found at (https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9118124/).






```
## [[1]]
## TableGrob (1 x 1) "arrange": 1 grobs
##   z     cells    name           grob
## 1 1 (1-1,1-1) arrange gtable[layout]
##
## [[2]]
## TableGrob (1 x 1) "arrange": 1 grobs
##   z     cells    name           grob
## 1 1 (1-1,1-1) arrange gtable[layout]
```

Although the improvement isn't drastic, this transformation help us stabilize the importance of the outlying observations, especially the shorter duration observations.

4. Comparing Two Naive Models

With the previous analysis of the relationships in our model, and with various domain knowledge, we constructed two models we thought would explain dancebility well without introducing too much multicollinearity or overfitting. Consider the two models below shown below. For both models, we determined that the variables 'valence','energy_square' and 'log_liveness' were all correlated enough with 'danceability' to always be included in the models. We also determined that the categorical variable 'Dance/Eletronic' should also be included for both models (given the name and previous histograms).
For the first model, we decided to focus more on tempo, genre and instrumentalness. This is an intuitive approach to modeling danceability; people usually dance to songs with a strong beat and/or groove in the instrumentation, and these features are common in certain genres. Furthermore, we predicted tempo would become very important for 'Dance/Eletronic' songs in terms of predicting dance-

ability (rather than for genres such as rock or jazz). We added an interaction term to reflect these hypothesises.

The second model focuses more on the speechiness and language of a song. We hypothesized that very wordy songs would be less danceable (similar to the opposite of instrumentalness). Furthermore, we hypothesized that explicit language would decrease the effects of valance perceived danceability of a song, as much of the perceived valence of song comes from the language itself; we included two interaction terms to reflect this hypothesis. Also, we should note that, given we included speechiness (and explicit language) in the second model, we removed the categorical variable 'hip hop' and `rock` to avoid multicollinearity.

```
##
## Call:
## lm(formula = danceability_square ~ valence + energy_square +
##     log_liveness + loglog_instrumentalness + `Dance/Electronic` +
##     `hip hop` + rock + tempo * `Dance/Electronic`, data = song_train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.45068 -0.09230 -0.00609  0.09363  0.48012
##
## Coefficients:
##                             Estimate Std. Error t value Pr(>|t|)
## (Intercept)                0.3883589  0.0241946  16.051  < 2e-16 ***
## valence                    0.3693905  0.0171491  21.540  < 2e-16 ***
## energy_square             -0.2003079  0.0190428 -10.519  < 2e-16 ***
## log_liveness              -0.0260601  0.0052392  -4.974 7.21e-07 ***
## loglog_instrumentalness    0.0017510  0.0002775   6.310 3.54e-10 ***
## `Dance/Electronic`         0.1852151  0.0528267   3.506 0.000466 ***
## `hip hop`                  0.0850709  0.0076685  11.094  < 2e-16 ***
## rock                      -0.0840684  0.0117208  -7.173 1.09e-12 ***
## tempo                     -0.0004559  0.0001408  -3.237 0.001231 **
## `Dance/Electronic`:tempo -0.0013780  0.0004205  -3.277 0.001071 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1458 on 1717 degrees of freedom
## Multiple R-squared:  0.3597, Adjusted R-squared:  0.3564
## F-statistic: 107.2 on 9 and 1717 DF,  p-value: < 2.2e-16
```

```
##
## Call:
## lm(formula = danceability_square ~ valence + energy_square +
##     log_liveness + log_speechiness + `Dance/Electronic` + explicit_numeric +
##     explicit_numeric * valence, data = song_train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.44007 -0.10033 -0.00396  0.09858  0.47211
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)                0.322253   0.023986  13.435  < 2e-16 ***
## valence                    0.433642   0.020262  21.402  < 2e-16 ***
## energy_square             -0.236226   0.019035 -12.410  < 2e-16 ***
## log_liveness              -0.028529   0.005366  -5.317 1.20e-07 ***
```

```
## log_speechiness           0.020558   0.005207   3.948 8.21e-05 ***
## `Dance/Electronic`        0.038863   0.009394   4.137 3.69e-05 ***
## explicit_numeric          0.165464   0.022136   7.475 1.22e-13 ***
## valence:explicit_numeric -0.156582   0.037039  -4.227 2.49e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1499 on 1719 degrees of freedom
## Multiple R-squared:  0.322,  Adjusted R-squared:  0.3193
## F-statistic: 116.7 on 7 and 1719 DF,  p-value: < 2.2e-16
```

We wanted to check for multicollinearity and any leverage or influence points using these 2 models as an overall insight for any future models as well.

```
##                  valence            energy_square             log_liveness
##                 1.158323                 1.283867                 1.035856
##  loglog_instrumentalness       `Dance/Electronic`                `hip hop`
##                 1.082351                35.608894                 1.138681
##                     rock                    tempo `Dance/Electronic`:tempo
##                 1.157709                 1.161211                35.986163
```

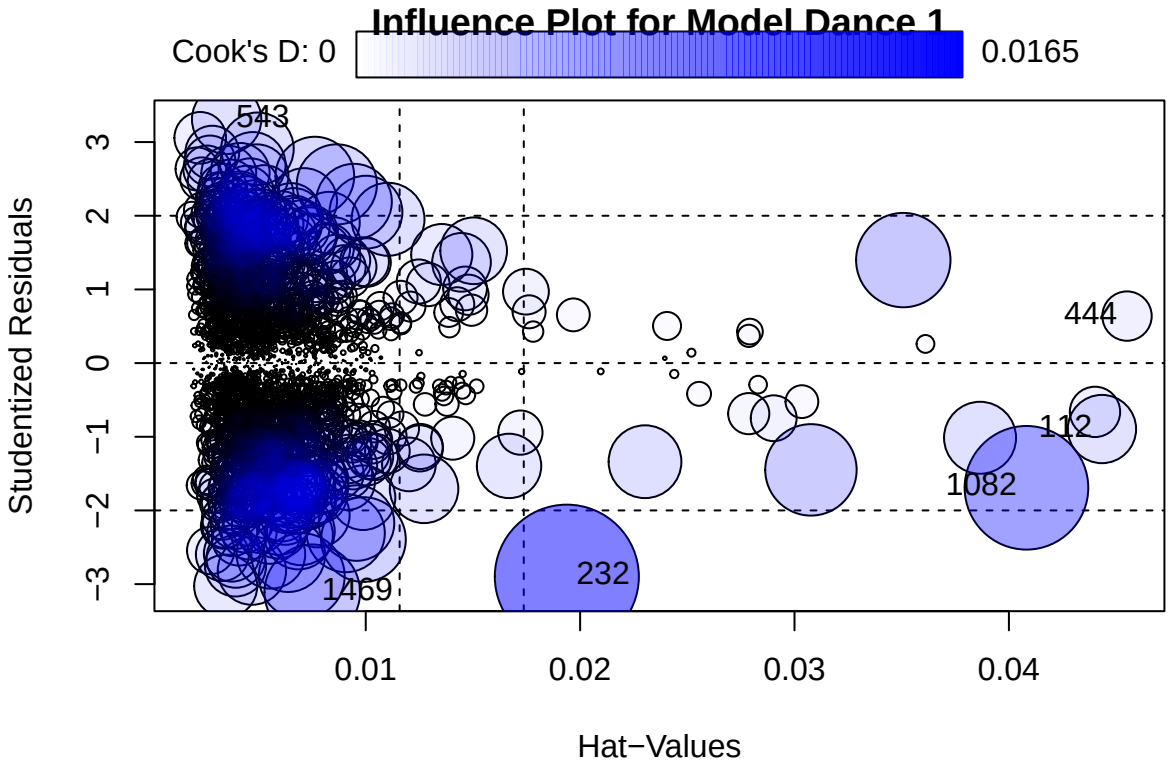
Multicollinearity varies across variables in the model after calculating the Variance Inflation Factor (VIF). While Valence, Energy Square, Log Liveness, and Sqrt Instrumentalness show low to very low levels, Dance/Electronic and Hip Hop, along with their interaction terms with Tempo, exhibit potential high multicollinearity. This suggests redundancy or strong correlation, potentially affecting model stability and interpretation.

```
##                  valence            energy_square             log_liveness
##                 1.528962                 1.212938                 1.027464
##          log_speechiness       `Dance/Electronic`          explicit_numeric
##                 1.260033                 1.064728                 7.573614
## valence:explicit_numeric
##                 7.521943
```

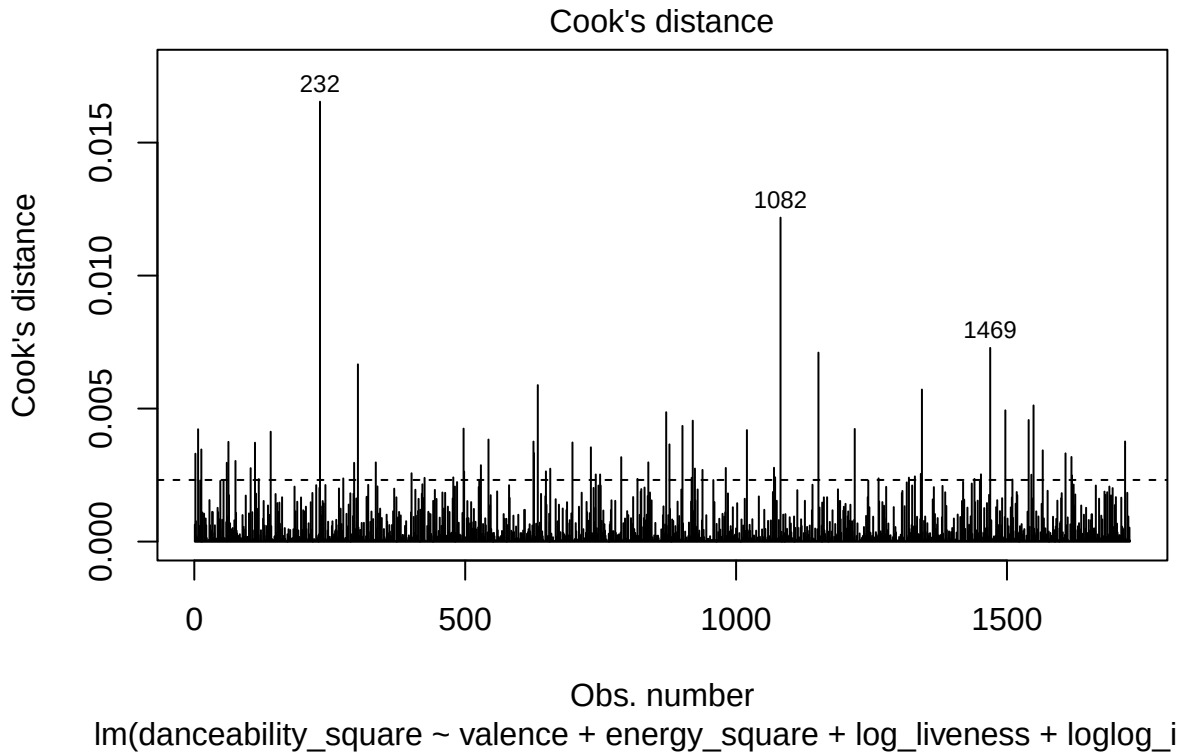
In this case, Valence, Energy Square, Log Liveness, Dance/Electronic and Log Speechiness show low to very low levels, Energy Square and Valence along with their interaction terms with Explicit Numeric show medium levels, and with Explicit_Numeric to cause a little concern for multicollinearity.

Afterward, we plotted an influence plot for each model to pinpoint Cook's distance and leverage points to uncover leverage or influential points that could influence the regression model's coefficients. This method helps us understand the model's reliability by highlighting potential outliers and assessing their impact.

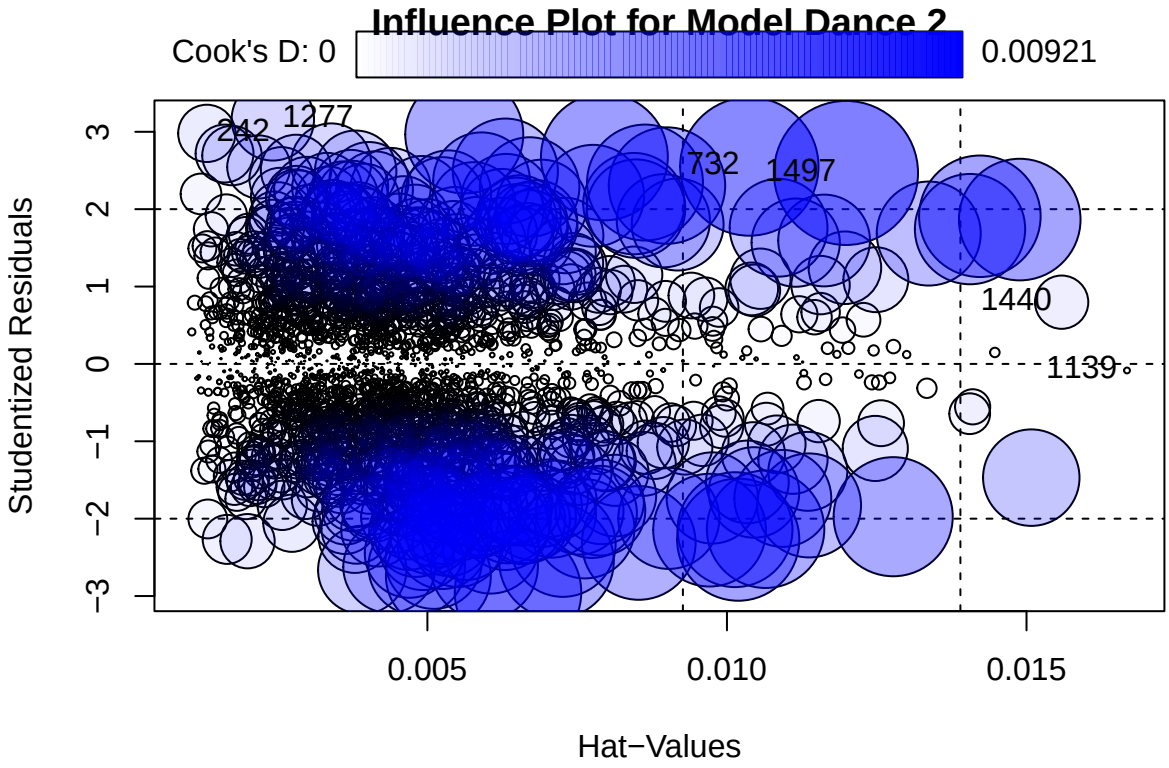
```
## [1] 0.04979734
```



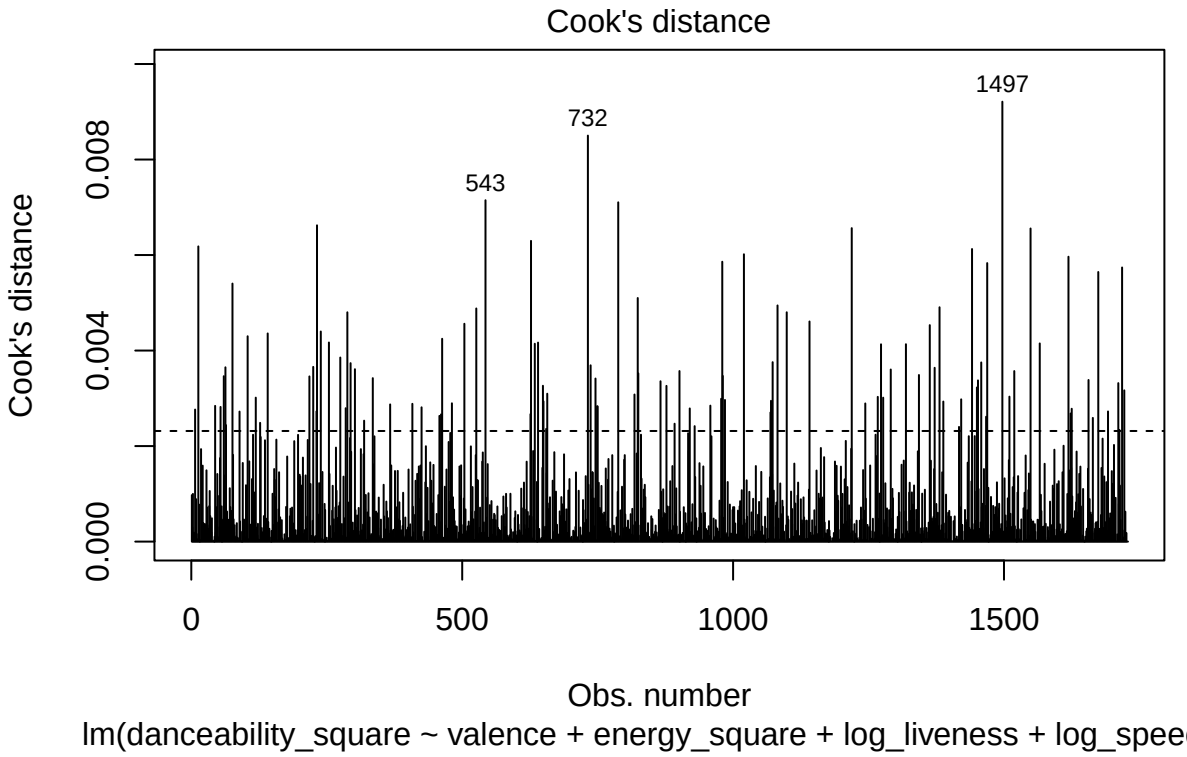

```
##          StudRes         Hat       CookD
## 112  -0.8950689 0.044339962 0.003717535
## 232  -2.8990650 0.019379877 0.016538543
## 444   0.6381355 0.045509020 0.001942234
## 543   3.3084116 0.003511720 0.003835115
## 1082 -1.6926118 0.040818681 0.012178717
## 1469 -3.1106520 0.007506349 0.007281400
```



The influence plot revealed a small handful of observations with leverage values greater than twice the average leverage $(2 \times (k+1)/n) = 0.047$ (chirp chirp), where k is the number of predictors and n is the number of observations. These observations can be considered high leverage points. There are unfortunate a small number that passed this threshold within our first model, indicating leverage points. However, when examining the Cook's distance plot, none of the observations had a Cook's distance value greater than 1, which is a common threshold for identifying influential observations.




```
##          StudRes         Hat        CookD
## 242   2.97966355 0.001317964 0.0014579245
## 732   2.55130162 0.010373363 0.0085014345
## 1139 -0.08455883 0.016677617 0.0000151676
## 1277  3.16078826 0.002420198 0.0030139670
## 1440  0.79795472 0.015586349 0.0012604484
## 1497  2.46814224 0.011987063 0.0092111967
```



The influence plot did not reveal any observations with high leverage values exceeding twice the average leverage. Similarly, the Cook's distance plot did not show any observations with a Cook's distance value greater than 1.

In summary, while model_dance1 had a few high leverage points identified through the influence plot, neither model exhibited observations with substantial influence based on the Cook's distance criterion. This suggests that, although there are some high leverage points in model_dance1, they may not be significantly influencing the regression coefficients or model fit. However, it's still recommended to investigate these high leverage points further and consider their potential impact on the model. These are most likely not the exact models we will be using, so it is a bright sign for building a more concise model later.

Then, we performed a 10-fold cross validation of our data to evaluate how well each model performs on unseen data.

```
## Linear Regression
##
## 1727 samples
##    7 predictor
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 1555, 1555, 1554, 1554, 1554, 1554, ...
## Resampling results:
##
##   RMSE       Rsquared   MAE
##   0.1480577  0.3355642  0.1170529
##
## Tuning parameter 'intercept' was held constant at a value of TRUE
```

```
## Linear Regression
##
```

```
## 1727 samples
##    8 predictor
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 1554, 1554, 1554, 1556, 1555, 1554, ...
## Resampling results:
##
##   RMSE       Rsquared   MAE
##   0.1481067  0.3375759  0.1176105
##
## Tuning parameter 'intercept' was held constant at a value of TRUE
```

As seen above, the models performed comparably in terms of RMSE (Root Mean Squared Error), R-Squared, and MAE in a 10 fold cross validation test with the training data. The second model appears to perform marginally better in each of the outputted metrics, thus indicating that the second model performs slightly better for predicting unseen data. This may indicate that language and genre feature are marginally more important for modeling danceability than instrumentation and tempo features.

5. Model Building

BIC Selection (Forward Stepwise vs Backwards)

We then considered a two different models built with Backwards BIC Selection and Forwards Stepwise BIC Selection. Since we are considering so many different variables and interaction terms in our model, we determined BIC was appropriate criterion for the purpose of penalizing overly complex models. Furthermore, many the predictors in our data set are correlated, therefore less complex models also helps us avoid issues with multicollinearity.
Consider the output models from each of these selection processes.

```
## [1] 1727
```

```
##                         artist                           song
##                              0                              0
##                    duration_ms                       explicit
##                              0                              0
##                           year                     popularity
##                              0                              0
##                   danceability                         energy
##                              0                              0
##                            key                       loudness
##                              0                              0
##                           mode                     speechiness
##                              0                              0
##                   acousticness                 instrumentalness
##                              0                              0
##                       liveness                        valence
##                              0                              0
##                          tempo                   duration_scale
##                              0                              0
##                 loudness_scale                 explicit_numeric
##                              0                              0
##                     key_numeric                            pop
```

```
##                             0                             0
##                          rock                         Other
##                             0                             0
##                       hip hop                           R&B
##                             0                             0
##              Dance/Electronic                popularity_high
##                             0                             0
##             popularity_modhigh                 popularity_low
##                             0                             0
##              popularity_modlow                log_speechiness
##                             0                             0
##                 valence_square                   log_liveness
##                             0                             0
##              log_acousticness          sqrt_instrumentalness
##                             0                             0
##            danceability_square                  energy_square
##                             0                             0
##                   yeo_duration                 root_duration
##                             0                             0
##      adjusted_instrumentalness        loglog_instrumentalness
##                             0                             0
## energy_square_explicit_numeric       valence_explicit_numeric
##                             0                             0
##            DanceElectronic_pop           DanceElectronic_rock
##                             0                             0
##          DanceElectronic_tempo                  hiphop_tempo
##                             0                             0
##         DanceElectronic_hiphop
##                             0
```

```
##
## Call:
## lm(formula = danceability_square ~ valence + energy_square +
##     log_liveness + loglog_instrumentalness + yeo_duration + `Dance/Electronic` +
##     `hip hop` + rock + tempo + explicit_numeric + `Dance/Electronic`:tempo +
##     energy_square:explicit_numeric, data = song_train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.45136 -0.08685 -0.00555  0.09136  0.45283
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                    0.3574843  0.0243826  14.661  < 2e-16 ***
## valence                        0.3666656  0.0168303  21.786  < 2e-16 ***
## energy_square                 -0.1341578  0.0212521  -6.313 3.48e-10 ***
## log_liveness                  -0.0274095  0.0051140  -5.360 9.47e-08 ***
## loglog_instrumentalness        0.0017664  0.0002722   6.489 1.13e-10 ***
## yeo_duration                  -0.0114805  0.0036834  -3.117  0.00186 **
## `Dance/Electronic`             0.1702532  0.0516817   3.294  0.00101 **
## `hip hop`                      0.0573929  0.0083938   6.838 1.12e-11 ***
## rock                          -0.0830693  0.0114485  -7.256 6.02e-13 ***
## tempo                         -0.0005944  0.0001382  -4.299 1.81e-05 ***
## explicit_numeric               0.1662996  0.0215711   7.709 2.13e-14 ***
```

```
## `Dance/Electronic`:tempo       -0.0012441  0.0004109  -3.027  0.00250 **
## energy_square:explicit_numeric -0.1995725  0.0388365  -5.139 3.08e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1422 on 1714 degrees of freedom
## Multiple R-squared:  0.3917, Adjusted R-squared:  0.3875
## F-statistic: 91.99 on 12 and 1714 DF,  p-value: < 2.2e-16
```

```
##
## Call:
## lm(formula = danceability_square ~ valence + energy_square +
##     `hip hop` + rock + loglog_instrumentalness + explicit_numeric +
##     log_liveness + tempo + yeo_duration + energy_square:explicit_numeric,
##     data = song_train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.45309 -0.08856 -0.00449  0.09298  0.44927
##
## Coefficients:
##                                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)                     0.3769131  0.0238032  15.835  < 2e-16 ***
## valence                         0.3634980  0.0168029  21.633  < 2e-16 ***
## energy_square                  -0.1293479  0.0210932  -6.132 1.07e-09 ***
## `hip hop`                       0.0560602  0.0084015   6.673 3.38e-11 ***
## rock                           -0.0868457  0.0112844  -7.696 2.35e-14 ***
## loglog_instrumentalness         0.0018407  0.0002686   6.852 1.01e-11 ***
## explicit_numeric                0.1652338  0.0216243   7.641 3.56e-14 ***
## log_liveness                   -0.0276546  0.0051269  -5.394 7.85e-08 ***
## tempo                          -0.0007200  0.0001309  -5.501 4.35e-08 ***
## yeo_duration                   -0.0130113  0.0036454  -3.569 0.000368 ***
## energy_square:explicit_numeric -0.1988571  0.0389462  -5.106 3.66e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1427 on 1716 degrees of freedom
## Multiple R-squared:  0.3874, Adjusted R-squared:  0.3838
## F-statistic: 108.5 on 10 and 1716 DF,  p-value: < 2.2e-16
```

```
##         Metric Backward_Selection Both_Direction_Selection
## 1          AIC      -1820.4404142            -1812.1468925
## 2          BIC      -1744.0824391            -1746.6971995
## 3 Adjusted R^2          0.3874806                0.3838239
## 4           Cp         13.0000000               11.0000000
```

Now, let's consider the intuition behind some of the values of our coefficients. Continuous variables such as 'valance', 'instrumentalness', and indicator variables such as 'Dance/Electronic', 'hip hop' and explict langage improve the danceability of songs according to this model. We believe this result to be consistent with our domain knowledge; we initially expected that more positive sounding, instrumental songs would more danceable and we also expected that Dance/Electronic and hip hop music would also be perceived as more danceable. Furthermore, songs with explicit language are more common in social setting where people dance (such as bars and clubs), so more danceable music may be produced with explict language more often. Increases in energy, liveness, tempo and all of our interaction terms, on the other hand, decrease the expected danceability (squared) of a song. Although liveness decreasing danceability isn't particularly surprising

(songs we dance to, especially in hip hop and eletronic music, usually aren't recorded live), energy decreasing danceability was a little suprising to us. We suspect that songs with too much energy are too intense to dance to properly. Our tempo coefficient, and interaction term with tempo, indicate that lower tempos are more conducive to dancing, and that, when a song is Dance/Electronic, higher tempos hurt the dancebility of the song more than for other genres. The two interaction terms involving *explicit_numeric* indicate that the positive effect of valence on danceability is diminished in the presence of explicit language. Additionally, explicit language amplifies the negative impact of high energy (squared) on danceability.

As seen above, the second model (found with Forward Stepwise BIC) has better criteria metric for BIC and *C_p*. Based on these metrics, we determined that the Forward Stepwise BIC model would likely better predict danceability for unseen data and would have better explainability and interpretability.
Let's consider how we can interpret the coefficient estimates and their signficance. Each of the predictors' p-values are below our predetermined signficance level of $\alpha = 0.01$; therefore, given all other predictors and interaction terms are held constant, each predictor in our model significantly contributes to the predictive power of the model. Aka. changes in these predictor values results in signficant changes to danceability of a song. The ANOVA table shown above confirms the signficance of the individual predictors in our final modal.

Before we evaluate this model on test data we set aside at the start, we will finally consider why certain variables were dropped out of the model during the BIC selection process. For example, if we look back to the correlation heatmap, we can see that 'hip hop' is very positively correlated with speechiness, and that energy is noteably correlated with both acousticness and loudness. You'll notice that none of 'speechiness', 'acousticness' or 'loudness' are included in the final model; these variables were likely dropped as they weren't signficant in our model (in terms of BIC) in the presense their highly correlated predictors 'hip hop' and 'energy'. One interesting observation is that 'energy' and 'valence' are positively correlated, however they are both included in the model. We suspect that both were signficant in the model because, although they are positively correlated, they are *oppositely* correlated to 'danceability.'

To confirm that our stepwise function deals with multicollinearity, we performed VIF test on the coefficients and found that none of values large enough to signify a large multicollinear relationship.

```
##                        valence                   energy_square
##                       1.161613                       1.645454
##                      `hip hop`                           rock
##                       1.427718                       1.120944
##       loglog_instrumentalness               explicit_numeric
##                       1.059377                       7.984521
##                  log_liveness                          tempo
##                       1.036120                       1.047986
##                  yeo_duration energy_square:explicit_numeric
##                       1.036730                       7.345291
```

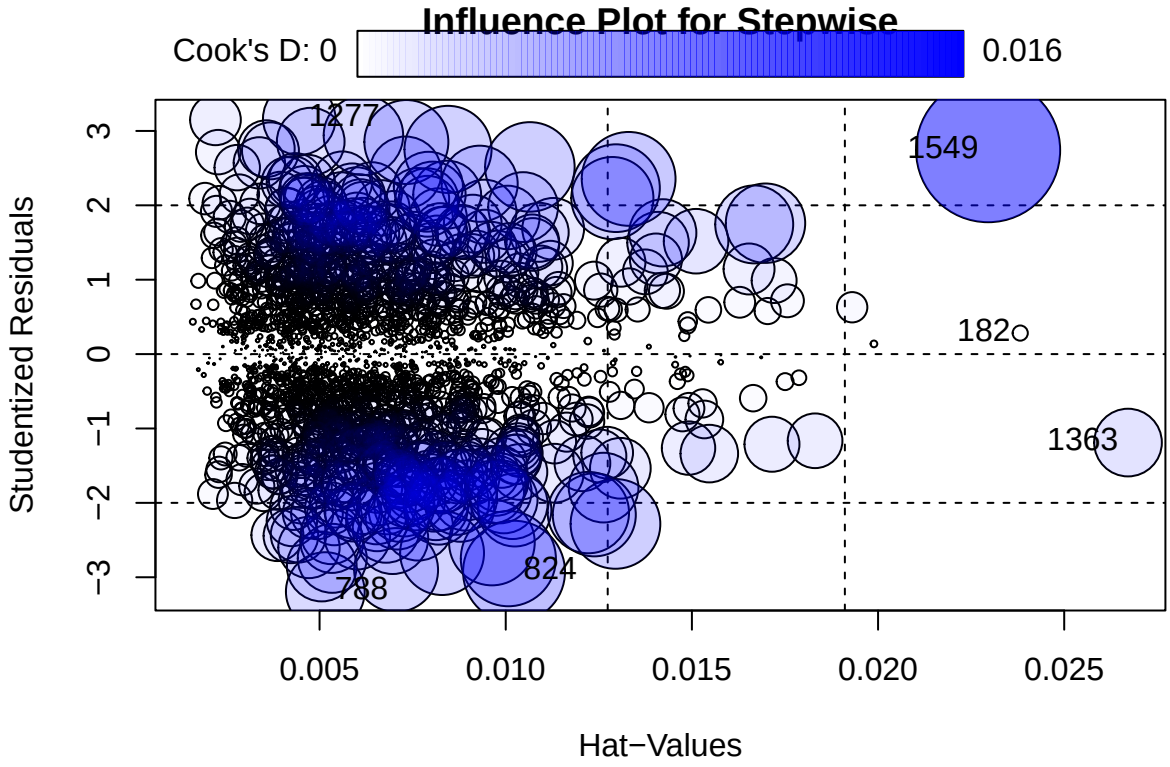
As we can see that none of these VIF scores are of concerning values, indicating our stepwise model did a very good job in selecting predictors to prevent multicollinearity.

```
## [1] "High influence points in the y direction:0"
```

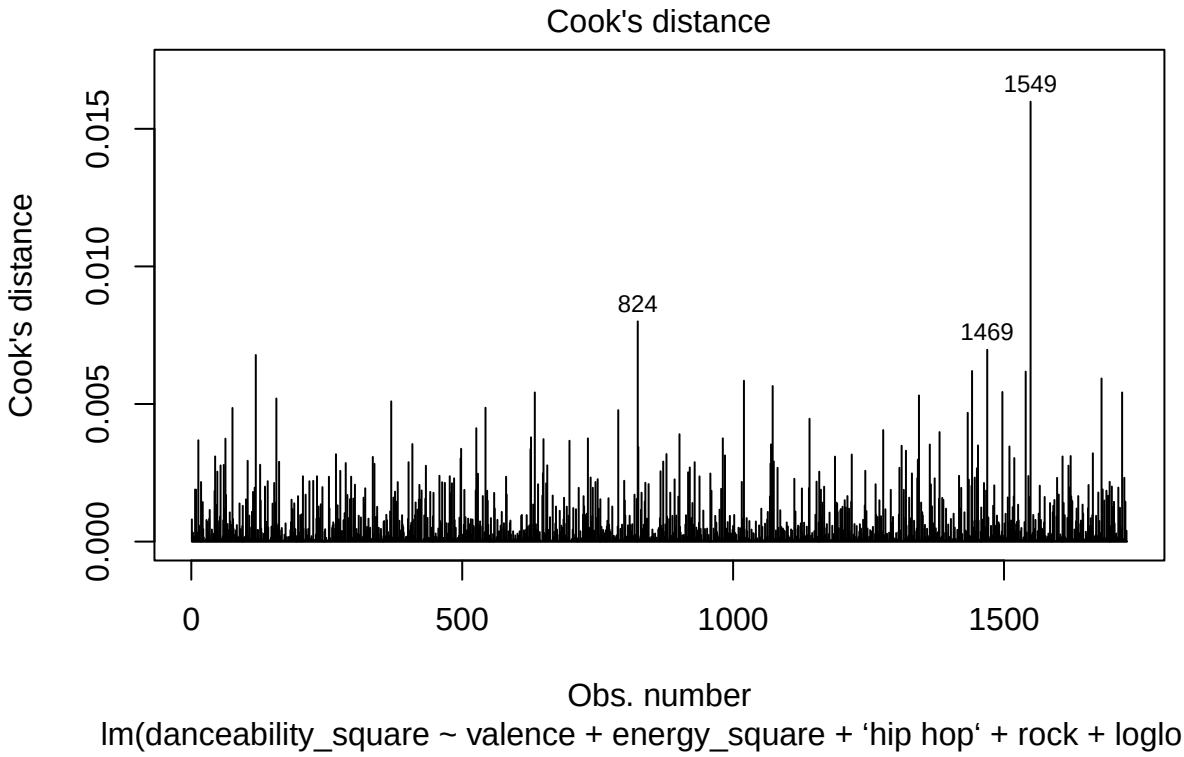
```
## [1] "High influence points in the x direction: 5"
```

Checking for studentized residuals as well as leverage points the presence of 0 high influence points in the y (dependent) variable but around 5 high influence points in the x (independent) variables suggests potential influential observations or non-linear relationships. By using the Bonferroni correction for outlier detection in the x direction, we are controlling the overall type I error rate, ensuring that the probability of any false positives is limited. Additionally, setting a leverage cutoff based on the number of predictors and sample size provides a standardized threshold for identifying potentially influential points in the y direction. We chose the cutoff to be 3 times the ratio of predictors to sample size due to how large our dataset is and how much variation is possible within the context of songs. Since there are a couple of potential high influence points,

let's dig deeper to look at specific observations.




```
##         StudRes         Hat        CookD
## 182   0.2825167 0.023818363 0.0001771373
## 788  -3.1929234 0.005153847 0.0047757223
## 824  -2.9258094 0.010223020 0.0080026167
## 1277  3.1646945 0.004454084 0.0040522140
## 1363 -1.1894371 0.026713203 0.0035291528
## 1549  2.7409640 0.022956062 0.0159864517
```



As we can through the influence plots, there are 3 specific values that seem to be way greater than 3 times the average hat value. Observations with high leverage, especially those significantly higher than the average leverage, have the potential to exert substantial influence on the estimated regression coefficients. These points are observations: 1549, 182, and 1363. If we focus on studentized residuals, we see that there are multiple studentized residuals greater than 2 or less than -2 which may cause for concern. However, since we have a large sample size, it makes sense for there to be individual extreme residual values. Since our model passes the assumptions of regression analysis, these studentized residuals might not indicate violations of these assumptions. Futhermore, when considering the effect of prediction, individual behavior of residuals may not be as concerning. Analyzing potential causual relationships between variable may need to be proceeded with extra caution due to extreme residuals. Another metric we looked at was Cook's Distance which showed that there is no Cook's distance greater than the threshold of 1. We are choosing 1 as our threshold due to our large sample size and passing of model assumptions. However, it is important to note that there is one observation, 1549, what is extremely higher than every other Cook's distance observation. Observation 1549 also has a big hate value and a large studenized residual, making it very likely this point is a concerning outlier.

```
## [1] "Threshold for DFFITs: 0.152189197461489"
```

```
##        1549         182        1362
##  0.42014092  0.04413009 -0.04860979
```

```
## [1] "Threshold for DFBETAs: 0.0481264499251422"
```

```
## [1]  0.079980141  0.005179937 -0.013576172
```

Looking at DFFITs and DFBETAs values for a couple of more concerning observations within our data, we can see that observation 1549 greatly exceeds both threshold while observations 182 and 1362 do not. We can conclude that 1549 is an extreme case and should be considered an outliers to remove in our model.

```
## # A tibble: 1 x 49
##   artist   song  duration_ms explicit  year popularity danceability energy   key
##   <chr>    <chr>       <dbl> <lgl>    <dbl>      <dbl>        <dbl>  <dbl> <dbl>
## 1 Sidney ~ Rive~      320348 TRUE      2009         47        0.804  0.976     1
```

```
## # i 40 more variables: loudness <dbl>, mode <dbl>, speechiness <dbl>,
## #   acousticness <dbl>, instrumentalness <dbl>, liveness <dbl>, valence <dbl>,
## #   tempo <dbl>, duration_scale <dbl>, loudness_scale <dbl>,
## #   explicit_numeric <dbl>, key_numeric <dbl>, pop <dbl>, rock <dbl>,
## #   Other <dbl>, `hip hop` <dbl>, `R&B` <dbl>, `Dance/Electronic` <dbl>,
## #   popularity_high <dbl>, popularity_modhigh <dbl>, popularity_low <dbl>,
## #   popularity_modlow <dbl>, log_speechiness <dbl>, valence_square <dbl>, ...
```

Our outlier observatoin is "Riverside" by Sidney Samson with a danceability of 0.804. Our previous models showed that danceability and energy have generally a negative correlation with one another, yet this specific outlier has a positive correlation between the 2 variables. It makes sense that this would be an outlier, and should be removed.

Bye bye, Riverside !

The model seems to perform reasonably well for the provided mean responses, with relatively narrow 95% confidence intervals indicating precise predictions. However, there is a wide 95% confidence intervals for future responses which may raise concerns about the model's ability to make reliable and accurate predictions for new, unseen data considering 0.016641, 0.950625 is the range of our predicotr variable. Such high uncertainty and low precision in future predictions could stem from various factors, including inadequate model complexity, missing important predictor variables, violations of model assumptions, overfitting or underfitting issues, or the presence of influential observations or outliers. To improve the model's overall performance and increase the precision of its predictions, it is crucial to revisit the modeling process, investigate potential issues, explore alternative modeling approaches or techniques, and validate the model's performance rigorously on a separate test or validation dataset.

Confidence Intervals

In our study on predicting danceability in songs, we've developed a linear model and now introduce confidence intervals (CIs) to enhance the interpretation of our results. These intervals provide ranges of values within which we're confident the true mean danceability lies for both current data and future predictions. They offer insights into the precision and variability of our model's estimates, aiding stakeholders in understanding the reliability of our predictions.

```
##          fit       lwr       upr
## 1  0.3029778 0.2739797 0.3319759
## 2  0.3036661 0.2809440 0.3263883
## 3  0.5177619 0.4904208 0.5451031
## 4  0.5023034 0.4830918 0.5215150
## 5  0.4032487 0.3672850 0.4392125
## 6  0.5749321 0.5576963 0.5921679
## 7  0.6423052 0.6175522 0.6670583
## 8  0.5365499 0.5214332 0.5516666
## 9  0.6108725 0.5772651 0.6444798
## 10 0.4118783 0.3967323 0.4270244
```

```
##          fit        lwr       upr
## 1  0.3029778 0.02168802 0.5842676
## 2  0.3036661 0.02295393 0.5843783
## 3  0.5177619 0.23663814 0.7988857
## 4  0.5023034 0.22185355 0.7827533
## 5  0.4032487 0.12115575 0.6853417
## 6  0.5749321 0.29461064 0.8552536
## 7  0.6423052 0.36142135 0.9231891
## 8  0.5365499 0.25635071 0.8167490
## 9  0.6108725 0.32907024 0.8926747
```

```
## 10 0.4118783 0.13167760 0.6920791
```

8. Sparse and Smooth Linear Models

Ridge Regression

For ridge regression and LASSO, we included many continuous, categorical and interaction variables that we found were fairly significant (although not necessary below our predetermined α level of 0.01) during the course of building our MLR models. The continuous variables are energy, valance, speechiness, instrumentalness, tempo, liveness and duration, while the categorical variables are Dance/Electronic, hip hop, rock, pop and mode. We also considered 5 interactions variables (that are comprised of the previous variables).

Diagnostics II

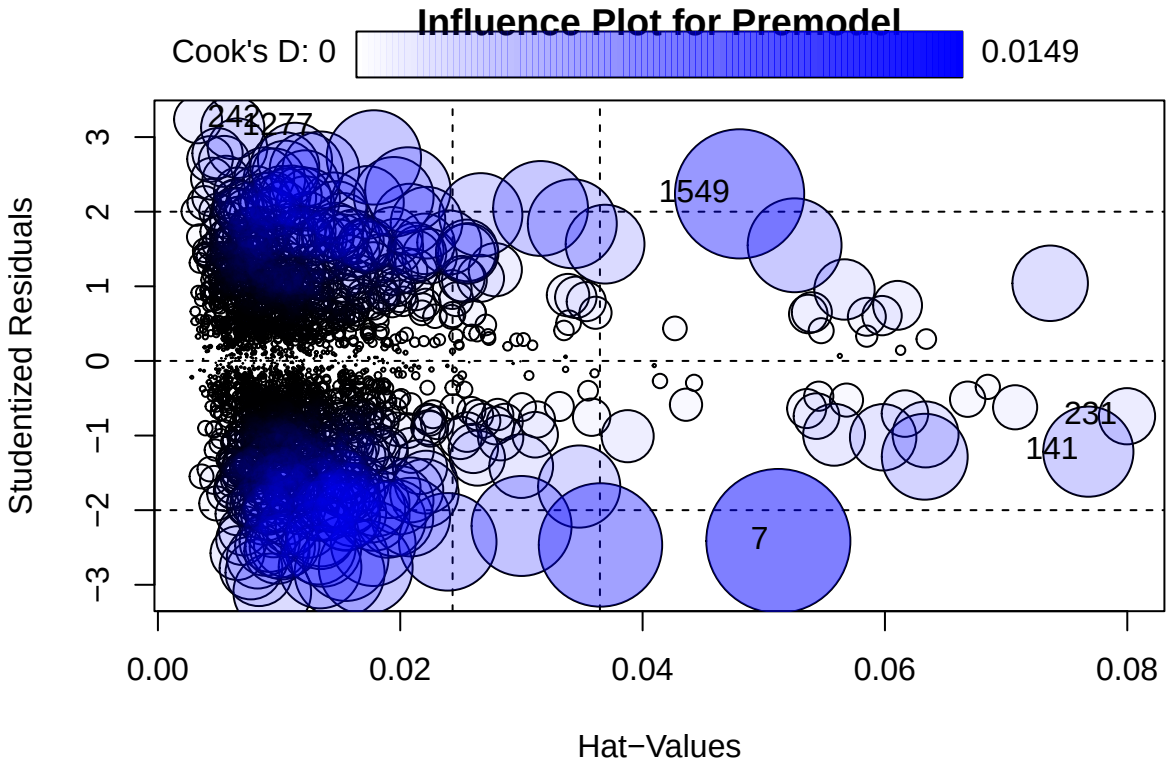
Before we consider these model, we will proceed with some high influential/leverage points analysis as well as an VIF analysis to check for multicollineairty. Analyzing high influential/leverage points and conducting a VIF analysis is crucial because it helps ensure the reliability and robustness of our predictive model. By identifying influential data points and assessing multicollinearity among predictor variables, we can address potential issues that might affect the accuracy and generalizability of the RR and Lasso model we are trying to build as well as their predictions.

After creating a premodel in with all the relevant predictors, we do the same process as the evaluations for the stepwise regressionmodel. Checking for studenized residuals as well as leverage points, we found the presence of 0 high influence points in the y (dependent) variable but 38 high influence points in the x (independent) variables suggests potential influential observations or non-linear relationships. This time, there are much more potential high influence points which makes sense due to how much more complex this premodel is compared to the previous stepwise one. Let's dig deeper to look at specific the observations of concern.

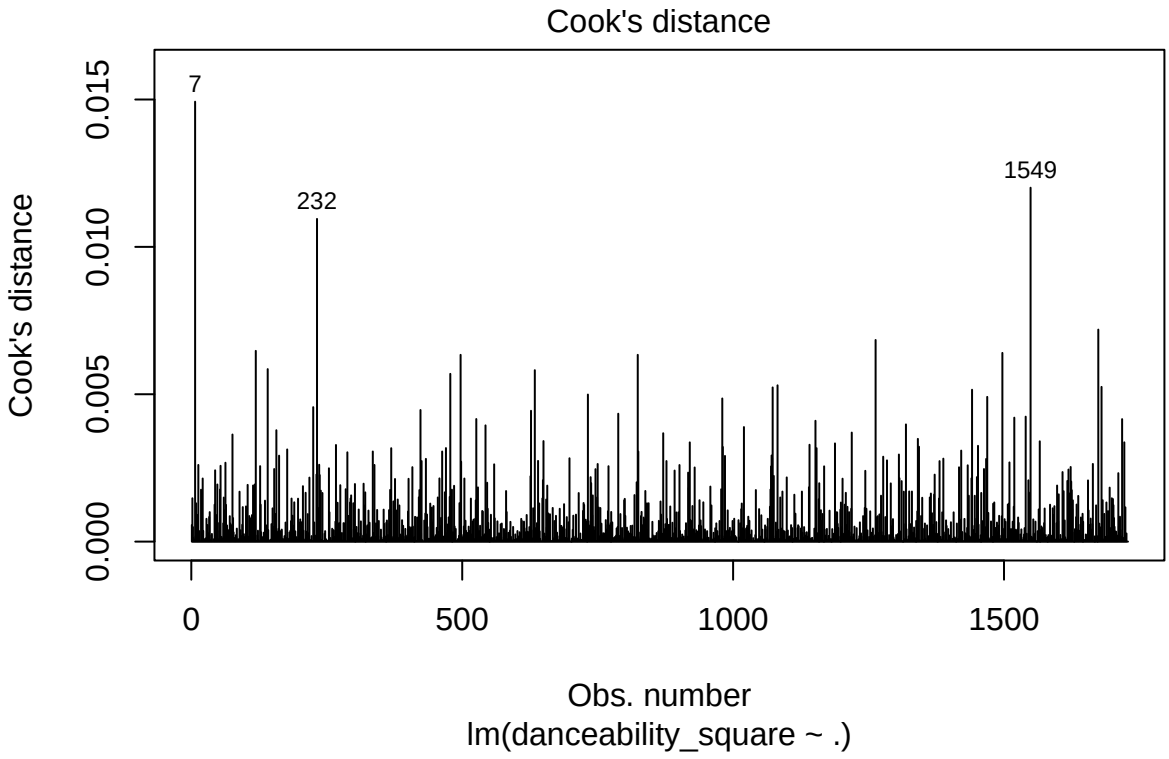
```
## [1] "High influence points in the y direction: 0"
```

```
## [1] "High influence points in the x direction: 37"
```

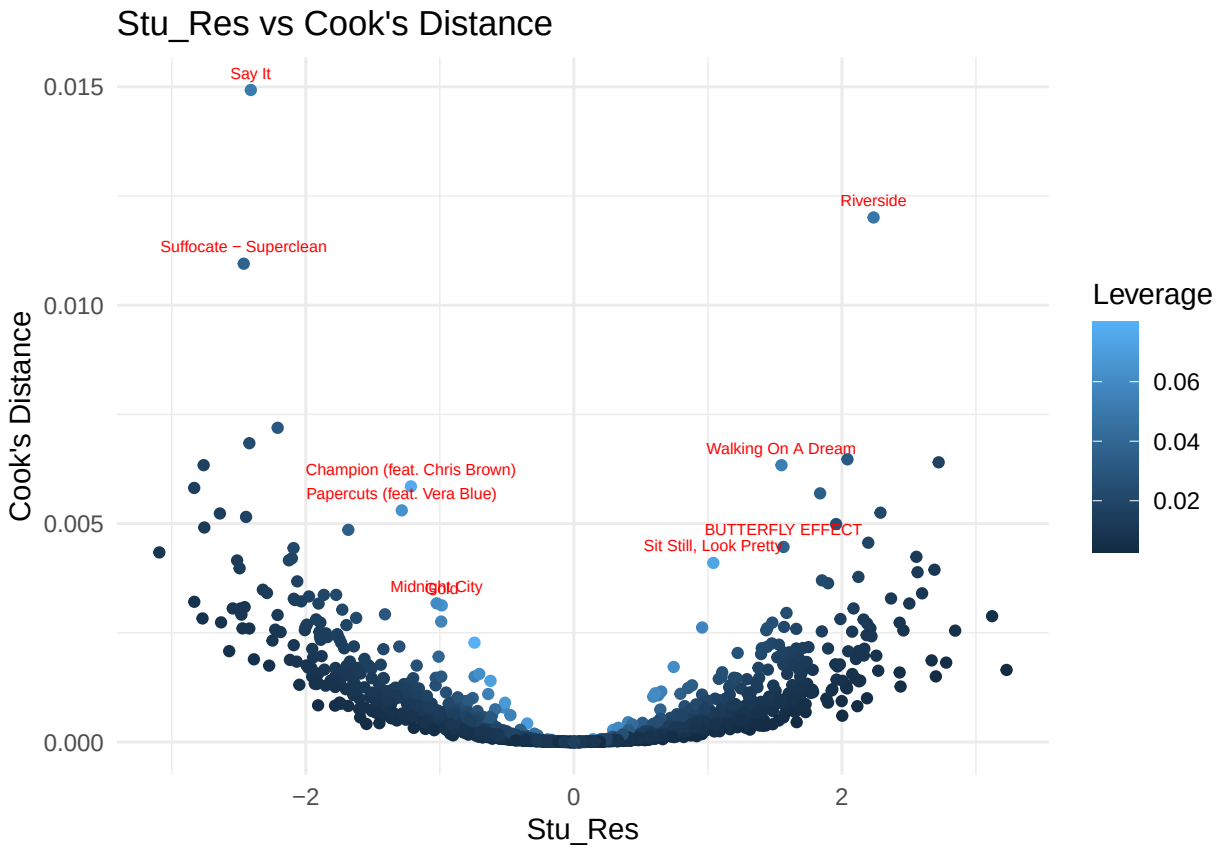
I sorted by cook's distance to see most influential points.




```
##          StudRes         Hat       CookD
## 7    -2.4134546 0.051202597 0.014926259
## 141  -1.2157836 0.076786147 0.005852644
## 231  -0.7412297 0.079977379 0.002274939
## 242   3.2374790 0.003312413 0.001649579
## 1277  3.1285661 0.006174003 0.002880689
## 1549  2.2389031 0.048004087 0.012008096
```

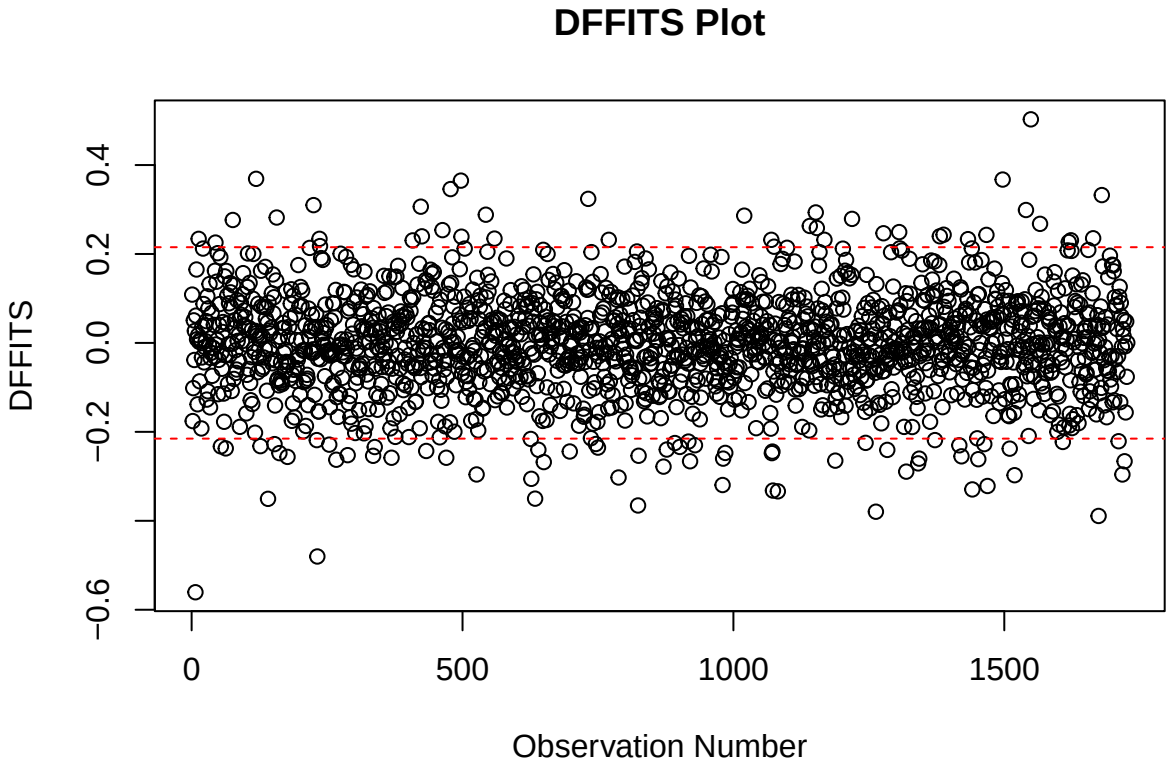


Plotting the influence and cook's distance plots, we notice a lot more concerning data points in comparison to our stepwise model. There a bunch of more concerning leverage point(hat values) as we found above. There are a couple of outlying cook's distance observations as seen in the cook's distance plots, with the points 7 and 232 being the most extreme. Points such as 7, 141, 231, 232, 242, and 1277 were selected by the plot to have the biggest influences on our data.




```
##                         SongTitle   Leverage        Cook    Stu_Res
## 7                          Say It 0.05120260 0.014926259 -2.4100490
## 1549                     Riverside 0.04800409 0.012008096  2.2362747
## 232          Suffocate - Superclean 0.03652536 0.010949545 -2.4628080
## 497              Walking On A Dream 0.05254491 0.006336972  1.5490469
## 141  Champion (feat. Chris Brown) 0.07678615 0.005852644 -1.2156133
## 1082  Papercuts (feat. Vera Blue) 0.06328454 0.005302900 -1.2838711
## 423                BUTTERFLY EFFECT 0.03690451 0.004465412  1.5643552
## 1152       Sit Still, Look Pretty 0.07363558 0.004099322  1.0406693
## 470                  Midnight City 0.05985178 0.003174017 -1.0232315
## 177                           Gold 0.06335624 0.003127852 -0.9854288
```

Cook's distance measures the influence of each observation on the fitted values of the model. It is often the most important metric because it captures both the leverage and residual information, which is where we will putting most our attention to. According to our plot, songs like "Say it" "Suffocate - Superclean" are shown to have the highest cook's distance as long with leverage and studenized residuals on the more extreme ends. Within this dataset, those 2 may be the most important to look at.




```
## [1] 1727
```

```
## [1] 1727
```

```
## [1] 18997
```

```
##                              SongTitle dffits_values     dfbetas
## 1549                          Riverside     0.5027555 -0.01174558
## 119                         Days Go By     0.3689971  0.04098402
## 1497                     bury a friend     0.3673673  0.18364143
## 497                 Walking On A Dream     0.3649462 -0.01167110
## 478                     Ladbroke Grove     0.3459889  0.01491116
## 1680                   Te Boté - Remix     0.3324437  0.11921624
## 732                 Turn Down for What     0.3239799  0.04668250
## 225  Girls Like (feat. Zara Larsson)     0.3098737 -0.04165077
## 423                   BUTTERFLY EFFECT     0.3063549  0.02033789
## 1540                              Sail     0.2988686  0.06504927
```

There are many valuess here that had a non-extreme Cook's distance, stutenized residuals, and leverage point values. If a value is extreme as DFFIT but not Cook's distance, it suggests that the particular observation has a significant influence on the predicted values of the model, but it may not necessarily have a large impact on the model coefficients. The observations that fall in both high cook & leverage values and the extreme dffits values are shown below.

```
##            SongTitle   Leverage        Cook  Stu_Res dffits_values     dfbetas
## 1          Riverside 0.04800409 0.012008096 2.236275     0.5027555 -0.01174558
## 2 Walking On A Dream 0.05254491 0.006336972 1.549047     0.3649462 -0.01167110
## 3   BUTTERFLY EFFECT 0.03690451 0.004465412 1.564355     0.3063549  0.02033789
```

Among the provided data points, "BUTTERFLY EFFECT" stands out as a potential outlier due to its high Cook's distance, standardized residual, and DFBETAS values, suggesting its substantial influence on the regression model. Although "Walking On A Dream" also exhibits notable leverage and DFFITS values,

"BUTTERFLY EFFECT" appears to have a more pronounced impact based on multiple diagnostic measures, indicating its potential as an outlier.

```
## [1] "BIC for Premodel:  -1714.45843353885"
```

```
## [1] "BIC for Premodel without Outliers:  -1715.21237267602"
```

While the 2nd model has the better BIC score, the difference is only 1. This small difference shows that these 2 outliers aren't having enough of an impact on our data for us to want to remove them from our dataset.

```
## [1] "Threshold for DFFITs: 0.215228027096715"
```

```
##           7         141         231         232         242        1277
## -0.14424173 -0.15544793 -0.09587117 -0.16177955  0.14799903  0.21168026
```

```
## [1] "Threshold for DFBETAs: 0.0481264499251422"
```

```
## [1] -0.038785649  0.049752881  0.026083031 -0.069562426  0.009843298
## [6]  0.016199819
```

Back to the original points that our influence plot suggested were the most concerning. We went on to find the specific values for the dffits and average dfbetas of our specific observations, but found none of them to be conclusively say passes the threshold. The 141, Champion (feat. Chris Brown), observation has an average around the dfbetas threshold, and therefore, should be looked deeper into the specific variables.

```
##                     (Intercept)                         valence
##                     0.049752881                     0.020966611
##                   energy_square                       `hip hop`
##                     0.001309079                    -0.025958787
##                            rock         loglog_instrumentalness
##                     0.009047265                     0.016315680
##                explicit_numeric                    log_liveness
##                     0.078997325                     0.008598595
##                           tempo                    yeo_duration
##                    -0.075391011                    -0.004189793
## energy_square:explicit_numeric
##                    -0.102226602
```

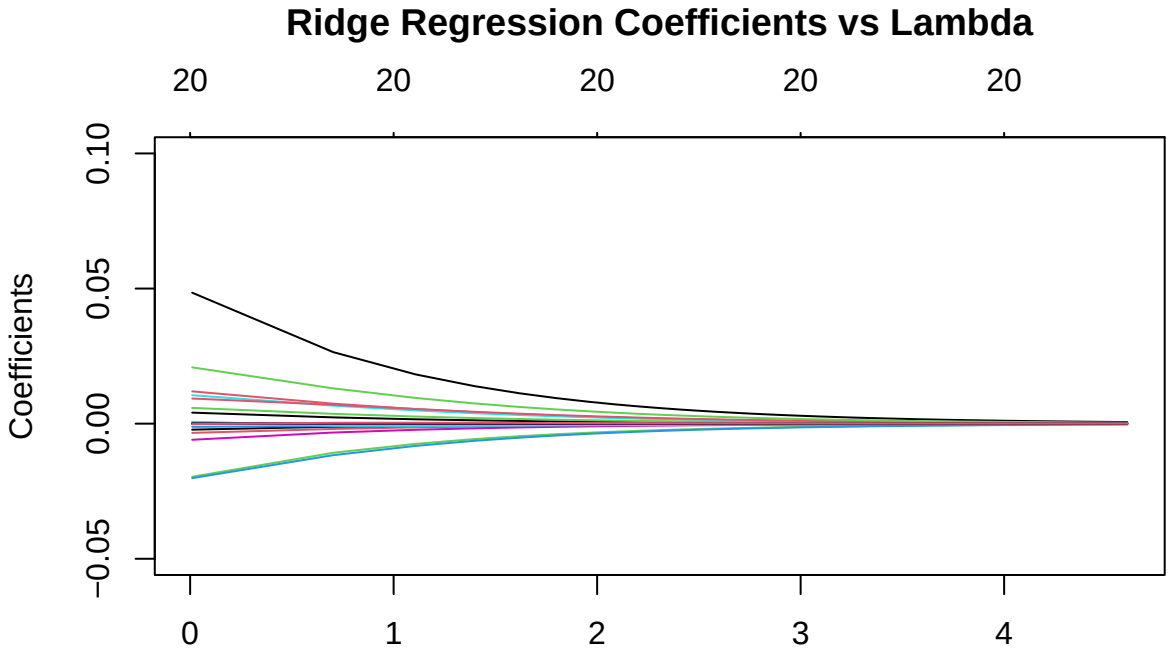
It looks like explicit_numeric is getting affected by this specific observation, but overall, doens't seem that conclusive enough to consider it an outlier worth getting rid of.

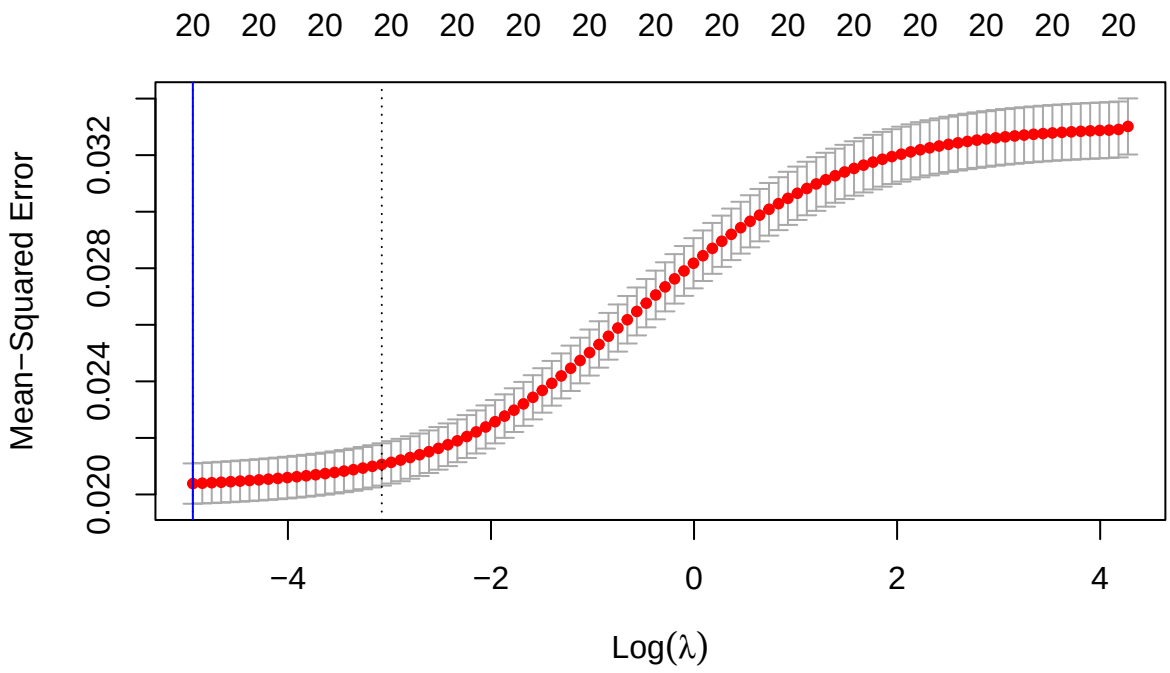
```
##                         valence                       `hip hop`
##                        1.641675                        21.281349
##                    energy_square                            rock
##                        1.788387                         1.543589
##                explicit_numeric              `Dance/Electronic`
##                       11.497913                        46.003712
##                             pop                            mode
##                        1.445148                         1.027716
##                 log_speechiness         loglog_instrumentalness
##                        1.433674                         1.113303
##                           tempo                     log_liveness
##                        2.083391                         1.041487
##                     yeo_duration energy_square_explicit_numeric
##                        1.092052                         8.236233
##       valence_explicit_numeric             DanceElectronic_pop
##                        8.508555                         7.102461
##           DanceElectronic_rock           DanceElectronic_tempo
##                        1.283321                        37.429898
```

```
##                hiphop_tempo         DanceElectronic_hiphop
##                    21.318213                       1.614601
```

After doing an VIF analysis, we unfortunately see that there a couple of terms such as hip hop and DanceElectronic_tempo that show high multicollinearity. Thankfully, Lasso and RR have methods in reducing the impacts of multicollinearity through their error terms. However, one can predict that lasso will perform better with these multicollinear values due its ability to dimension reduce.

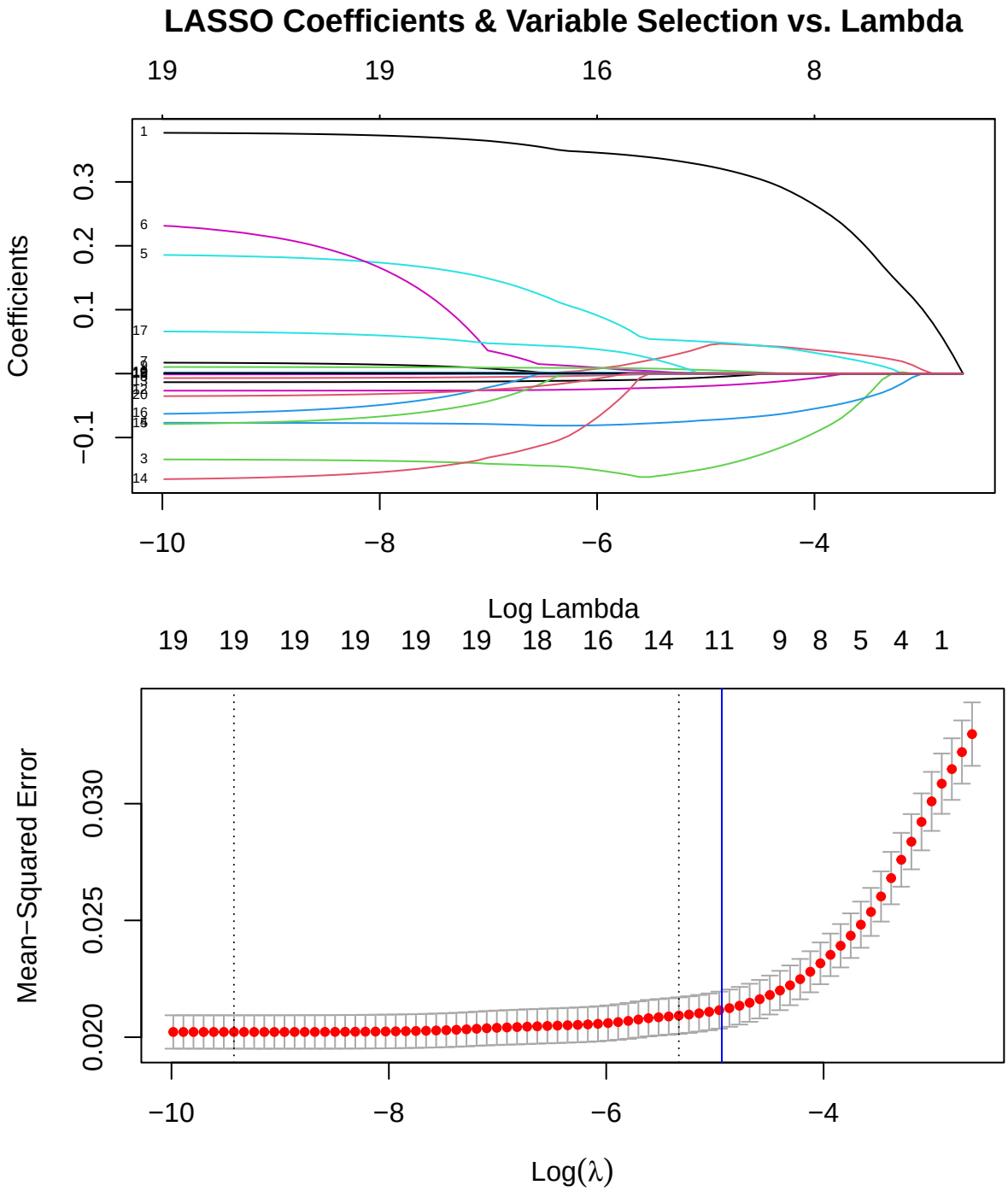
Now that we have completed some important diagnostics, we will continue to the Ridge Regression.





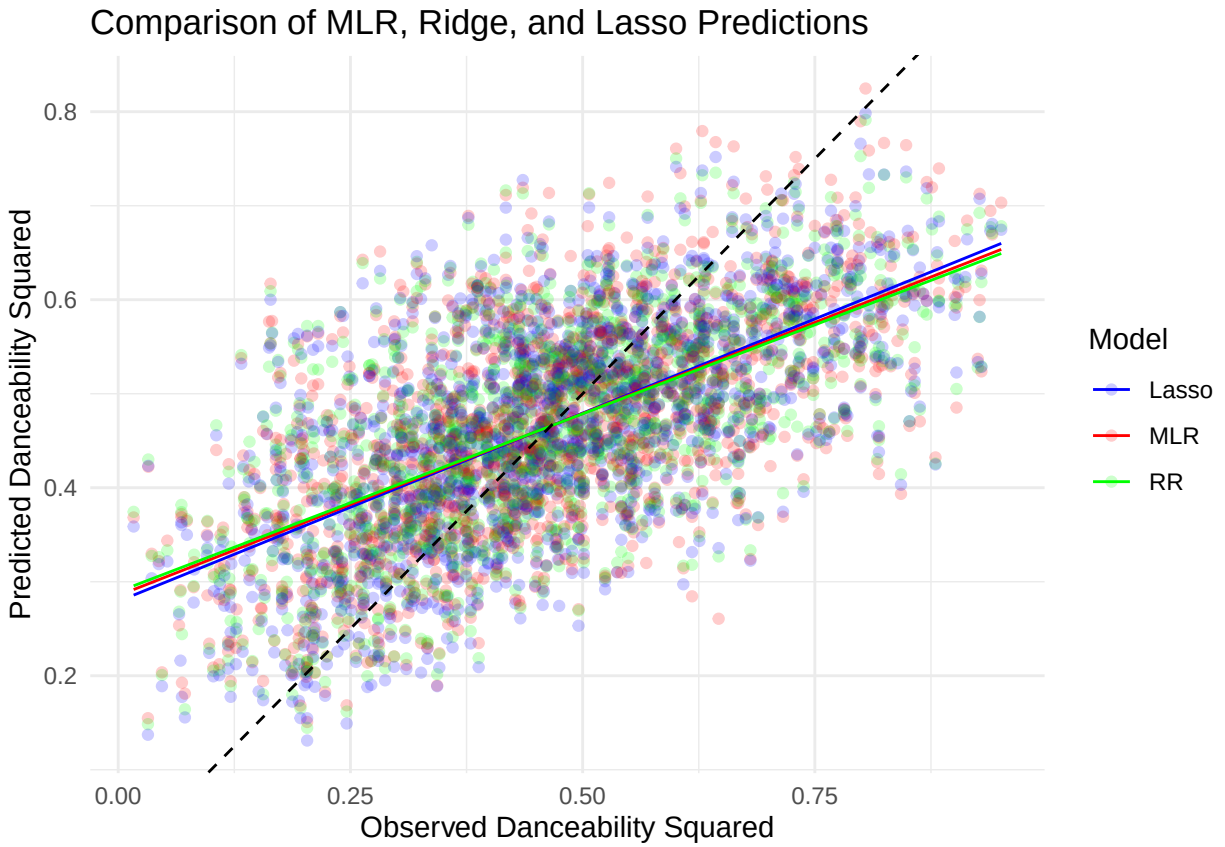
We can see from the first figure that the size of the coefficients is reduced as we increase the lambda value, as expected. The second figure shows the mean squared error for models at each lambda value (found from a 10-fold cross validation). The best lambda for the model found was quite low (show by the blue line in the second figure), and therefore the coefficents in this model are not being regularized/reduced by much. We found this surprising as there was moderate correlation between some of the variables in the full model. We suspect that because we did significant feature transformations and normalization, the coefficients were already going to be quite small and therefore would not benefit much from being reduced further.

LASSO Regression



As seen above, the LASSO method is both reducing coefficient values and performing variable selection, as expected. Also, we can see the MSE of each the LASSO models with different lambda values. The best model found (with cross validation), whose log(lambda) value is shown by the blue line, shows that LASSO is performing significant variable reduction for the best model (from about 20 predictors in our model to about 10). It is clear that the regularization performed by LASSO is much more stringent that the regularization performed by Ridge Regression.

Comparing MLR, RR and LASSO



The plot above shows the observed response value for our observations, vs the predicted response value of the observations for each of our Stepwise BIC (MLR), RR and LASSO models. As we can see from this figure, it appears the LASSO model's predicted values are slightly more correlated with actual observed values than the MLR model, which is also slightly more predictive than the Ridge Regression model. \ We suspect that the reason the Ridge Regression model is slightly less predictive than the other two models is that this method keeps too many correlated variables in the model. Similarly, we suspect LASSO makes better choices in terms of variable selection (and coefficient reduction) for multicollinearity than the stepwise BIC (MLR) model.

The predicted response vs observed response relationships for all three of these models are noticable flatter than the 'ideal' situation (predicted value perfectly match the observed values). These slopes suggest that are model are often giving us conservative estimatations of the predicted values.

Overall, we conclude that the LASSO model may be more reliable in predicting new data, however the slopes are so similar that more cross validation tests would likely need to be performed to confirm this difference.

9. Bisquare (Robust) Regression

Since we considered a few different models and methods that all were intended for mostly dimension reduction and regularization, we wanted to consider a regression technique that is particularly robust against outliers and high leverage points. Earlier in this project, we discussed some issues in our data relating to potentially outlying points of high influence (and/or high leverage/residual) Through our research, we found that the Bisquare Regression was appropriate for this task. Furthermore, we thought that if this type of regression resulted in a large improvement compared to our previous methods, then we should reevluate the points of high influence (also high leverage and high residual points), and see if we need to perform more diagnostic

measures on potentially outlying points.
As a form of robust regression, Bisquare Regression focuses on minimizing the impact of large residuals. Sometimes, the oridinary least squares method for choosing parameters can perform poorly on data with lots of high leverage and/or high residual points (especially at the tails). If you recall from our transformations and early diagonistics, there were heavy-tails in many of our error distrubutions (Q-Q plots) and points with high leverage and influence. An alternative to simply removing the points that are causing the prolem in our normal errors assumptions (these points may contain useful information about the true relationships), we can instead use a robust regression to find our coefficients by minimizing an "objective function" over all predictors rather than the residuals themselves. Each residual "contrubutes" to this objective function.
We determine how much a residual contributes to the objective function through a $\rho(e)$ function. In OLS, this $\rho(e)$ function is simply $\rho(e) = e^2$. Now, the $\rho(e)$ function has changed, so we are no longer minimizing the sum of e_i^2 terms (rather $\rho(e)$ that can be various functions, but usually share some important qualities). The derivative of this function w.r.t the predictors is the influence curve. Set these partial derivaties to 0 gives us $k + 1$ estimating equations for each coefficient. The weight function $w(e)$ is then the influence curve divide by the number of residuals.
The Bisquare Regression has a unique objective function and weight function (compared to the otherregression methods.) For example, as a residual e increases in size, the objective function for OLS (given by the $\rho(e) = e^2$ contributions) will quadratically increase in size. The bisquare objective function, on the other hand, will start to level off when as e increase, when $|e| > k$.We call this k parameter the tuning constant, where smaller values will cause the contribution from a residual to level off sooner for larger e values. In this way, we can see how bisquare regression gives the model more immunity to these large residuals (as the "contribution" to the objective function we are minimizing is not as large as in OLS). Reference for Information:
Fox, John, and Sanford Weisberg. "Robust Regression." University of Minnesota, 8 Oct. 2013, http://users.stat.umn.edu/ sandy/courses/8053/handouts/robust.pdf.

In the following output, we considered two bisquare regressions on two different linear models. First, the model with the variables chosen from the full set of variables we used for Ridge Regression and the variables found Stepwise BIC Selection.

```
##
## Call: rlm(formula = danceability_square ~ valence + energy_square +
##     log_liveness + log_speechiness + loglog_instrumentalness +
##     tempo + explicit_numeric + yeo_duration + rock + hip_hop +
##     explicit_numeric + energy_square_explicit_numeric + valence_explicit_numeric +
##     DanceElectronic_pop + DanceElectronic_rock + DanceElectronic_tempo +
##     hiphop_tempo + DanceElectronic_hiphop, psi = psi.bisquare)
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.446277 -0.087471 -0.001806  0.088789  0.449371
##
## Coefficients:
##                                  Value   Std. Error t value
## (Intercept)                       0.4612  0.0344    13.4137
## valence                           0.3863  0.0203    19.0448
## energy_square                    -0.1429  0.0224    -6.3921
## log_liveness                     -0.0256  0.0052    -4.8920
## log_speechiness                   0.0123  0.0054     2.2851
## loglog_instrumentalness           0.0018  0.0003     6.5171
## tempo                            -0.0012  0.0002    -6.4913
## explicit_numeric                  0.1942  0.0264     7.3672
## yeo_duration                     -0.0130  0.0038    -3.4412
## rock                             -0.0841  0.0126    -6.6600
## hip_hop                          -0.0187  0.0329    -0.5681
## energy_square_explicit_numeric -0.1764  0.0420    -4.2036
```

```
## valence_explicit_numeric       -0.0827  0.0381    -2.1712
## DanceElectronic_pop            -0.0071  0.0204    -0.3479
## DanceElectronic_rock            0.0628  0.0366     1.7153
## DanceElectronic_tempo           0.0001  0.0002     0.9364
## hiphop_tempo                    0.0006  0.0003     2.3857
## DanceElectronic_hiphop         -0.0244  0.0208    -1.1732
##
## Residual standard error: 0.1305 on 1709 degrees of freedom
```

```
##
## Call: rlm(formula = danceability_square ~ valence + energy_square +
##     log_liveness + loglog_instrumentalness + tempo + explicit_numeric +
##     yeo_duration + rock + hip_hop + explicit_numeric + energy_square_explicit_numeric,
##     psi = psi.bisquare)
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.461283 -0.088145 -0.002191  0.091926  0.452086
##
## Coefficients:
##                                Value   Std. Error t value
## (Intercept)                     0.3921  0.0243    16.1199
## valence                         0.3734  0.0172    21.7467
## energy_square                  -0.1390  0.0216    -6.4467
## log_liveness                   -0.0257  0.0052    -4.9009
## loglog_instrumentalness         0.0018  0.0003     6.6594
## tempo                          -0.0008  0.0001    -6.2014
## explicit_numeric                0.1718  0.0221     7.7749
## yeo_duration                   -0.0130  0.0037    -3.4996
## rock                           -0.0851  0.0115    -7.3837
## hip_hop                         0.0561  0.0086     6.5330
## energy_square_explicit_numeric -0.2069  0.0398    -5.1993
##
## Residual standard error: 0.1343 on 1716 degrees of freedom
```

The metrics for the two models found with Bisquare Regression, particularly the t-values, show that the model with fewer variables has more significant variables. We will consider how this model performs on new data later.

10. Regression Trees

Another new addition was CART (Classification and Regression Tree) model into the linear model for predicting danceability in songs is a valuable addition, as it allows for capturing potential nonlinear relationships and interactions between the predictor variables, which a linear model alone may not be able to capture effectively.

Danceability is a complex characteristic influenced by various musical elements like tempo, rhythm, and energy. CART offers a powerful non-parametric method well-suited for this type of problem. Unlike linear regression which assumes linear relationships between predictors and target, CART can automatically capture non-linear patterns and intricate interactions among audio features. This flexibility makes it an attractive choice when the underlying relationships are unknown or difficult to specify upfront. It serves as an excellent comparison to the previous linear models we have created because CART requires minimal data preprocessing and feature engineering compared to linear methods. While transformations and variable selection were crucial steps for the linear models, CART can automatically handle raw features, different scales, outliers, and mixed data types without those pre-processing steps. Additionally, unlike the parametric linear models

which make assumptions about the form of the relationships, CART is non-parametric and can flexibly model any non-linear and interaction effects present in the data.

The way CART works is that it recursively partitions the data into subsets based on input feature values, creating a tree-like structure of decision rules. For regression tasks like predicting danceability scores, CART aims to create terminal nodes (leaves) with minimal variance in the target variable. It then uses greedy algorithms to grow this tree by selecting splits that minimize a certain metric, such as mean squared error. However, unconstrained tree growth can lead to overfitting the training data. This is where cross-validation becomes crucial. By systematically holding out parts of the data during model training, cross-validation provides an unbiased estimate of the model's predictive performance on unseen data. This estimate guides the pruning process, allowing for the selection of an optimal tree size that balances underfitting and overfitting.

By comparing the predictive performance of CART to the linear, regularized, and non-linear models, it is possible to gain insight on whether predicting danceability is better approximated by additive and interactive linear components or if there are strong hierarchical interactions that tree-based methods more naturally uncover. Therefore, using CART as a comparative approach, given the extensive linear modeling efforts already undertaken, is a well-justified methodological decision.

```
## CART
##
## 2000 samples
##   13 predictor
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 1800, 1800, 1800, 1799, 1799, 1801, ...
## Resampling results across tuning parameters:
##
##   cp          RMSE       Rsquared   MAE
##   0.04249368  0.1251459  0.2088939  0.09853466
##   0.06880619  0.1304675  0.1408492  0.10237627
##   0.12733186  0.1373488  0.0913364  0.10903216
##
## RMSE was used to select the optimal model using the smallest value.
## The final value used for the model was cp = 0.04249368.
```

```
## CART
##
## 1727 samples
##   13 predictor
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 1554, 1556, 1552, 1555, 1556, 1553, ...
## Resampling results across tuning parameters:
##
##   cp          RMSE       Rsquared    MAE
##   0.04086882  0.1672139  0.15670149  0.1334736
##   0.04590204  0.1708570  0.12046275  0.1358406
##   0.12117834  0.1791238  0.07728368  0.1444602
##
## RMSE was used to select the optimal model using the smallest value.
## The final value used for the model was cp = 0.04086882.
```

Here, we are using a 10 fold cross validation on 2 separate models: one with the transformed variables and another with the original variables from our song data. The results show that the models with the original

model than a higher R^2 and cp are shown, even though RSME and MAE remain better for the model with transformed variables.

The higher RMSE and MAE values for the original variable model indicate it has higher overall predictive accuracy on the danceability target. These error metrics directly evaluate how close the predictions are to the actual values. So in terms of pure predictive performance, the transformed model seems preferable.

However, the higher R-squared of the original variable model suggests it explains a higher percentage of the variance in the danceability scores. R-squared reflects how well the model captures the underlying patterns and relationships in the data. The complexity parameter (CP) of the 2 models is pretty similar, with the transformed variable model being slightly better. Overall, it is very clear that the original variable model outperformed the transfomred variable model.

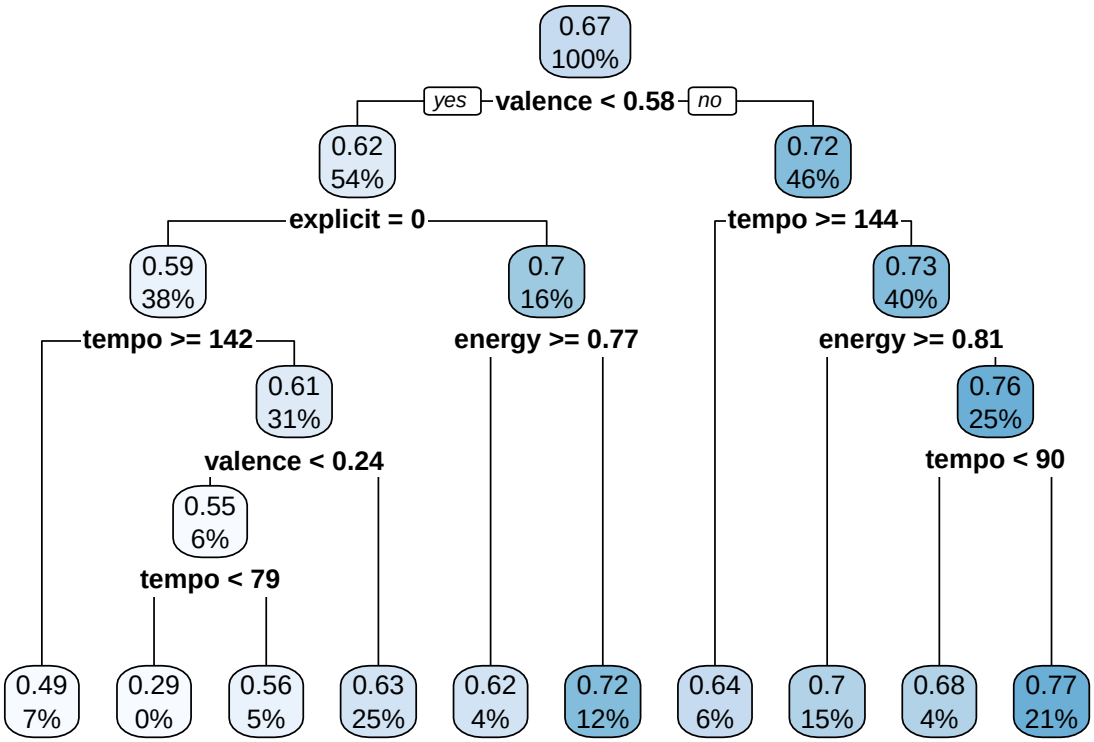
The superior performance of the untransformed CART model suggests that the danceability dynamics are potentially better represented by the hierarchical tree structure learned directly from the raw features, rather than the restricted parametric forms imposed by linear model transformations. This flexibility of CART to model complex patterns "as is" renders variable transformations unnecessary and even counterproductive in this case.

The results indicating that the CART model with all the original variables intact outperformed the version using transformed variables are insightful. This outcome shows the strength tree-based approaches, their ability to naturally handle different scales, distributions and complex non-linear patterns without transforming any variables.

Ultimately, both models provide valid perspectives worth considering. The results showcase how transformations can impact different performance aspects of tree models in contrasting ways.

```
##                     Overall
## tempo            0.80182866
## energy           0.48616688
## speechiness      0.46164957
## valence          0.43689465
## acousticness     0.31057907
## explicit         0.28933363
## duration_scale   0.18507507
## liveness         0.08848534
## loudness         0.04633674
## instrumentalness 0.00000000
## mode             0.00000000
## popularity       0.00000000
## key              0.00000000
```

An important output from CART models is the measure of variable importance, which quantifies each predictor's contribution to the overall variance explained by the tree. Variables towards the top split nodes carry more weight, while lower rankings imply lesser contributions after accounting for the other predictors. The relatively high importance value of 0.80 for tempo indicates it was the most influential variable in constructing the regression tree to predict danceability scores. This aligns with intuition as people tend to dance more to song with higher speeds and tempos. Energy, speechiness, and valence are also shown to be significant factors which again, makes sense to its correlation to danceability.



Finally, here is the regression tree visualized. For each leaf node displays the mean predicted target value for that subset of observations. The tree diagram includes root nodes, internal split nodes, and terminal leaf nodes which provides an intuitive representation of how different predictor variables interact to partition the data into distinct subgroups.

11. Smoothing Models with Tempo

Figure : Danceability vs Tempo

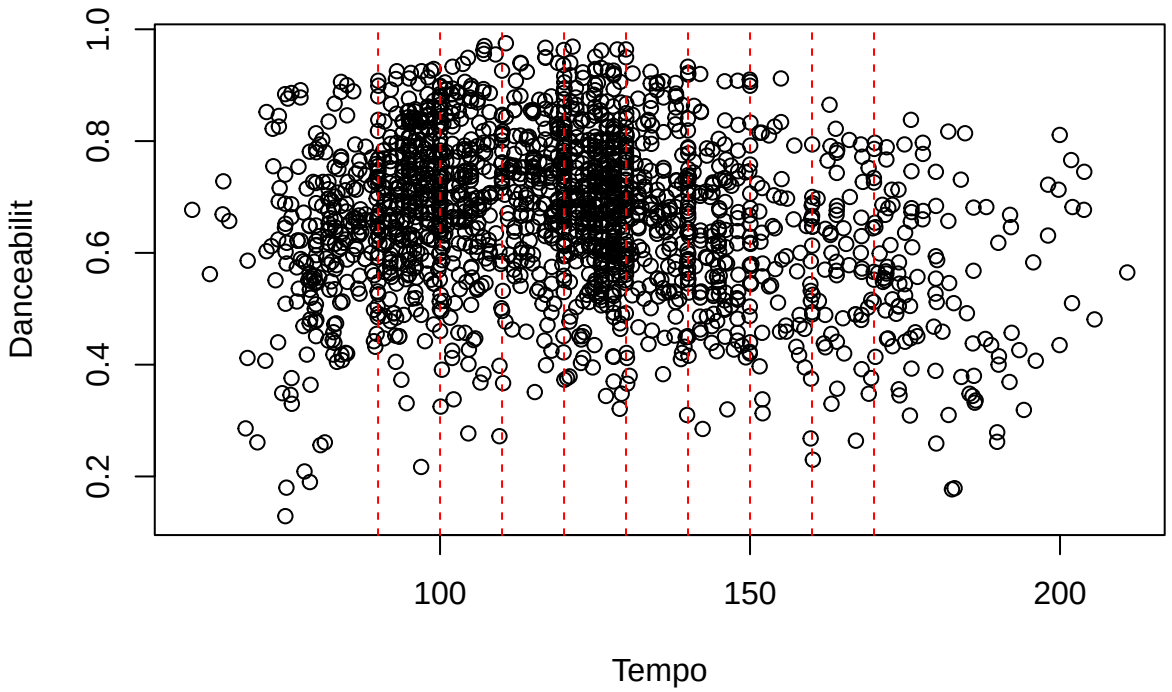
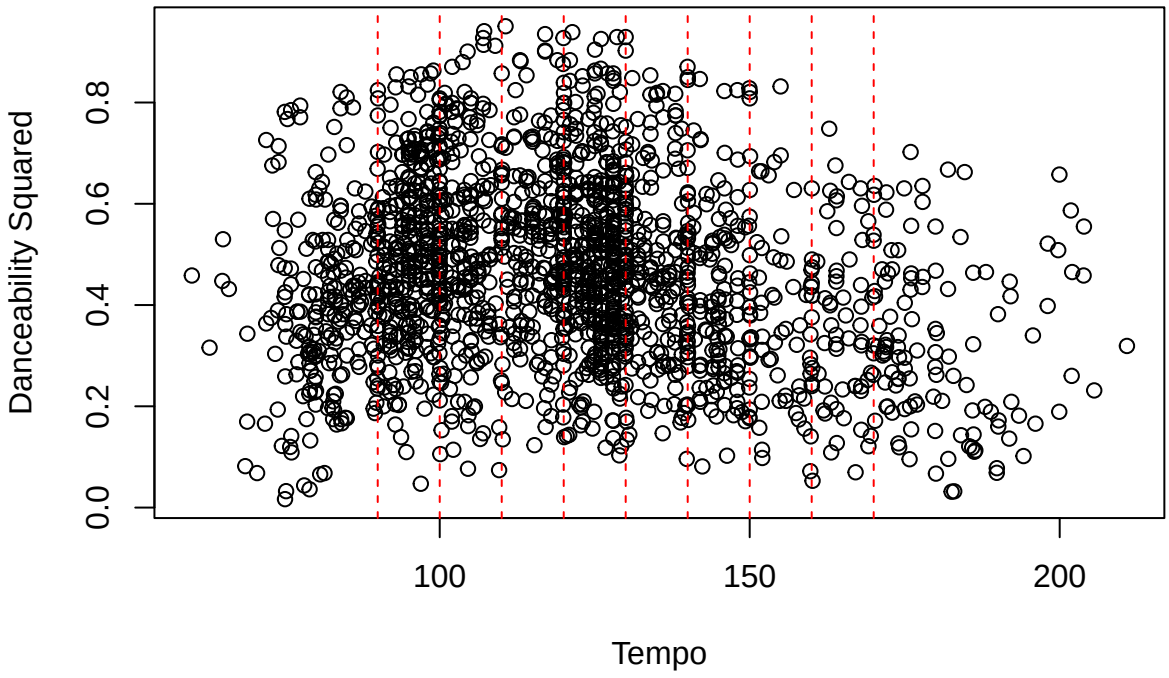
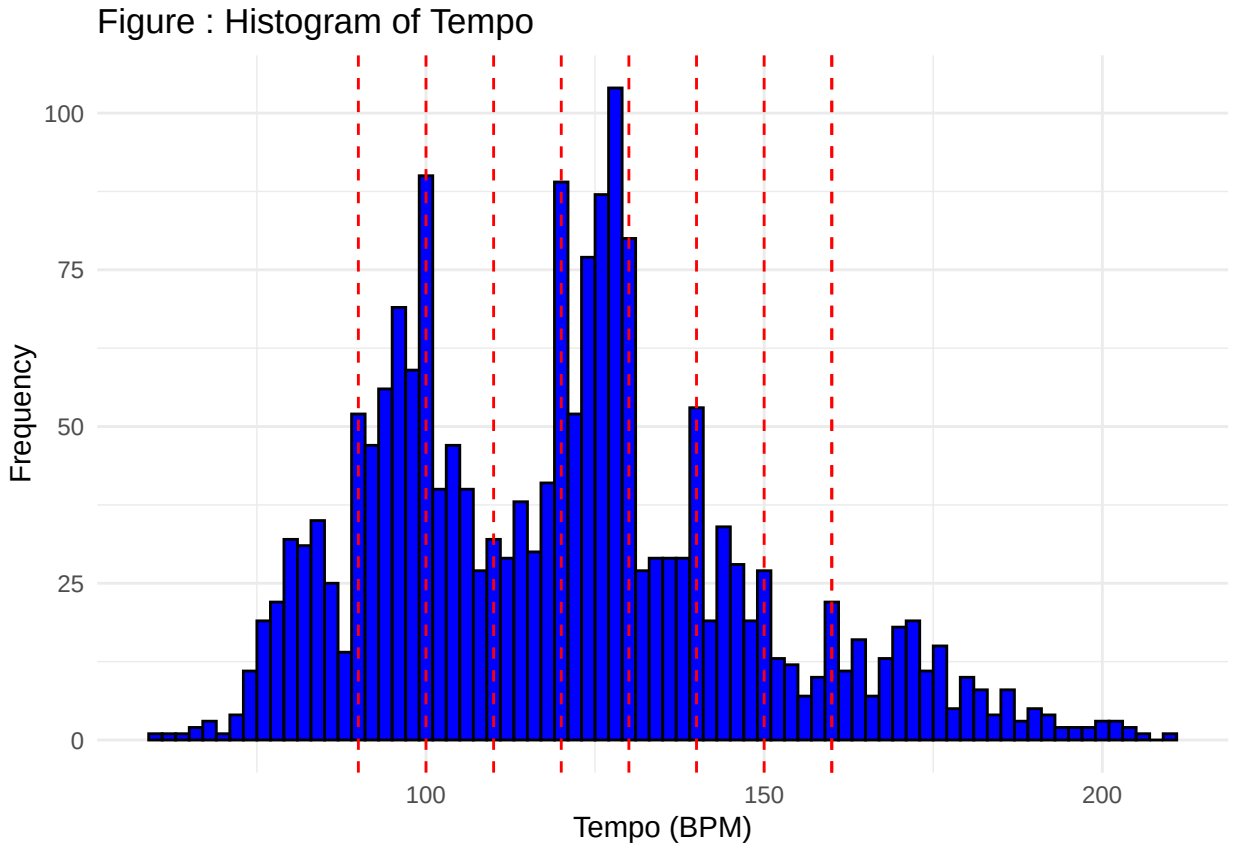


Figure : Danceability Squared vs Tempo

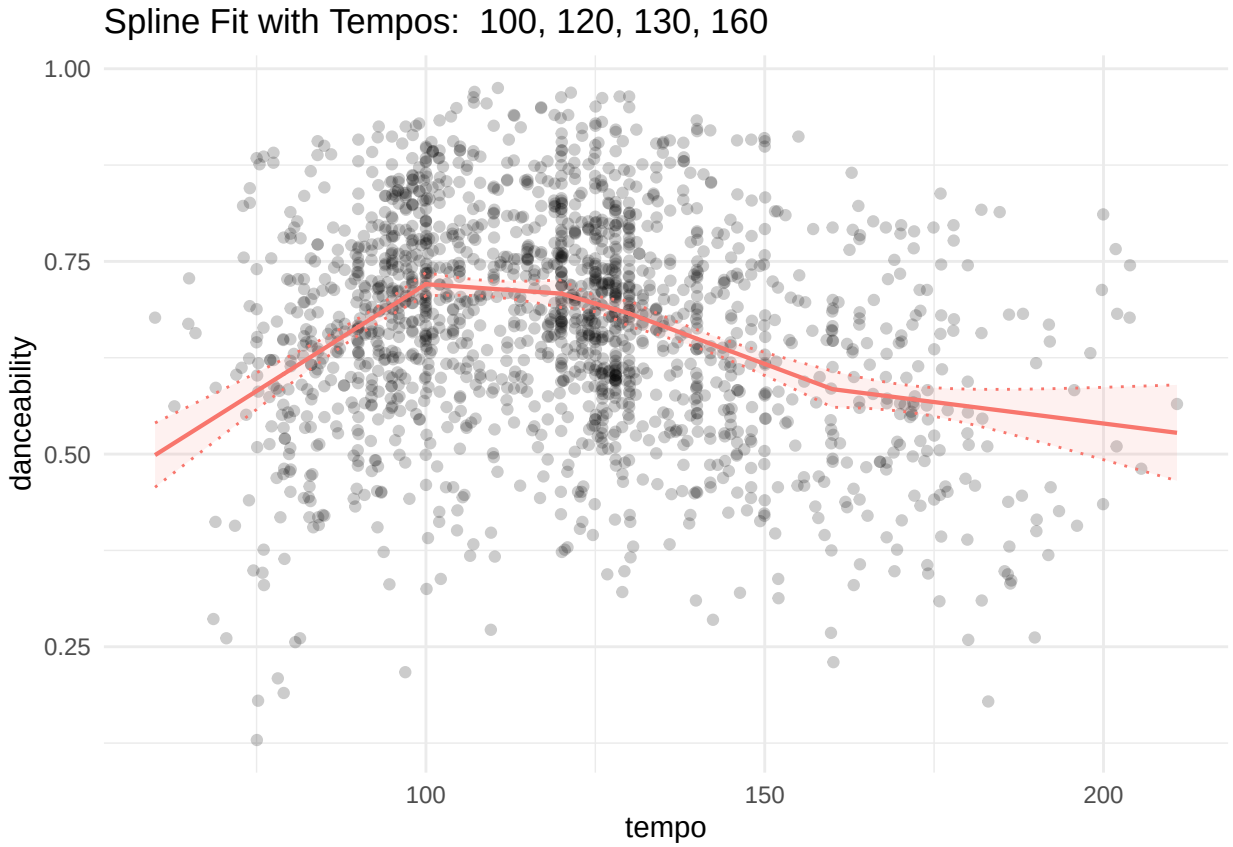


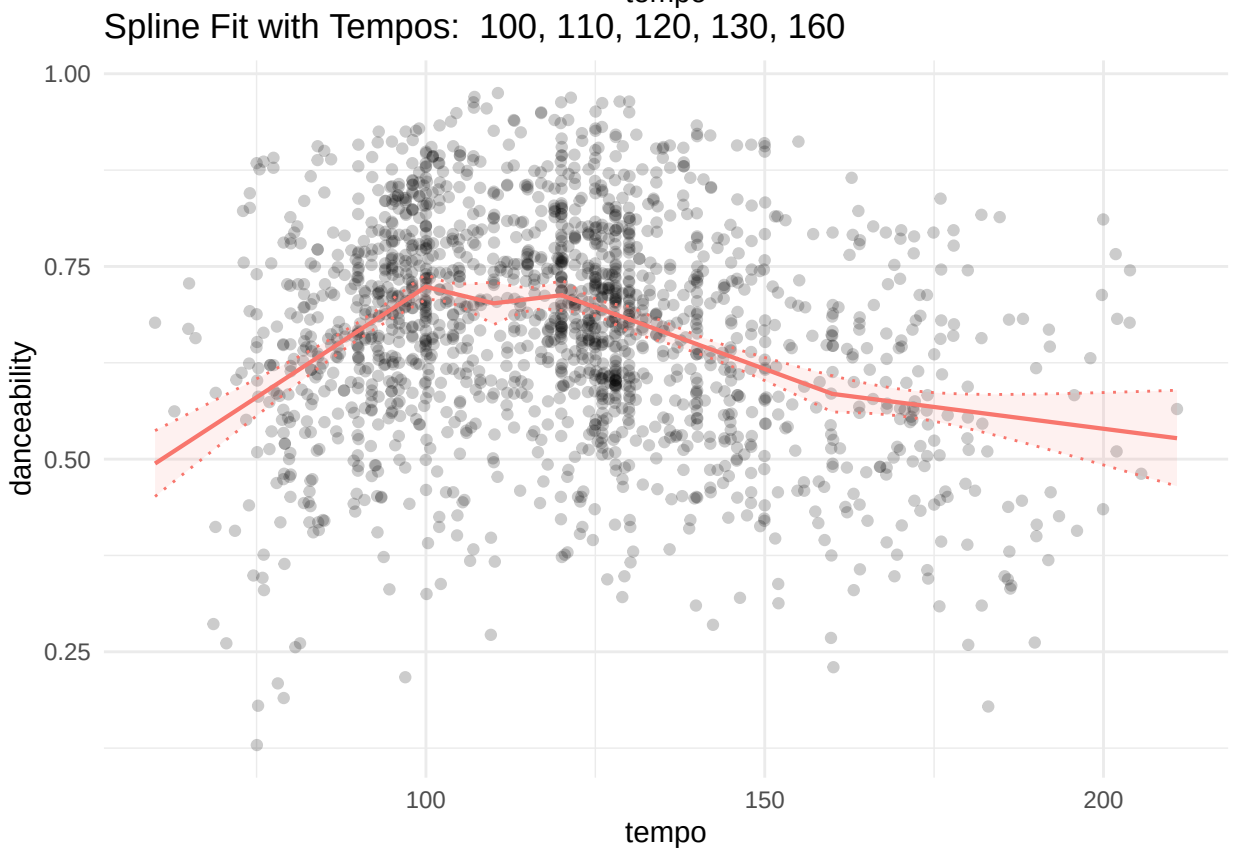


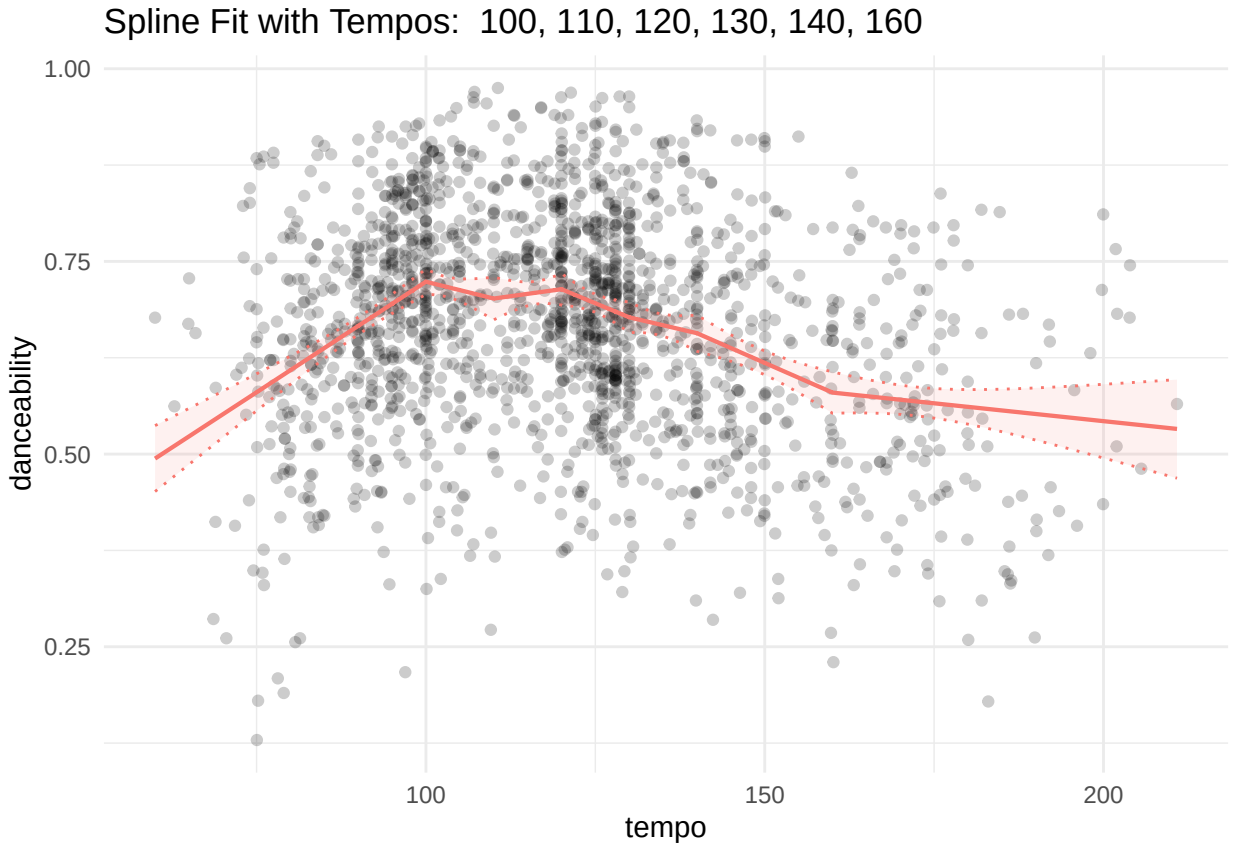
As we can see from the figures above , tempo has a somewhat strange relationship with dancebility (and danceability^2); its relationship to danceability looks slightly quadtratic, although slightly less so when we transform the response (as we did before). Also, note that quadtraic transformations on the tempo predictor did not improve its assumptions on the normal errors model. Futhermore, our domain knowledge tells us that tempo is a bit of a unique variable. In 4/4 time signature (which the vast majority of popular songs are in), a 100bpm song can sound very similar in "speed" or "groove" to a 200bpm song, as the speed of the song is relative to how many beats/sounds are actually played a second. Therefore, given the unique properties of 'tempo', we believe it is a good canditate to use the smoothing techniques on. From our domain knowledge, we also know that tempos like 90-170 are very common in popular and danceable songs (while song outside these ranges are generally unique and deserve their own range, or have a similar groove if you divide or multiple them by 2 to one of the section in 90-170). From this knowledge, and from our observations of the plots above, we determined (a maximum of) nine ranges of interest that were appropriate for the following smoothing methods. In the plots above, you will see red lines indicating the knot tempos for these ranges.

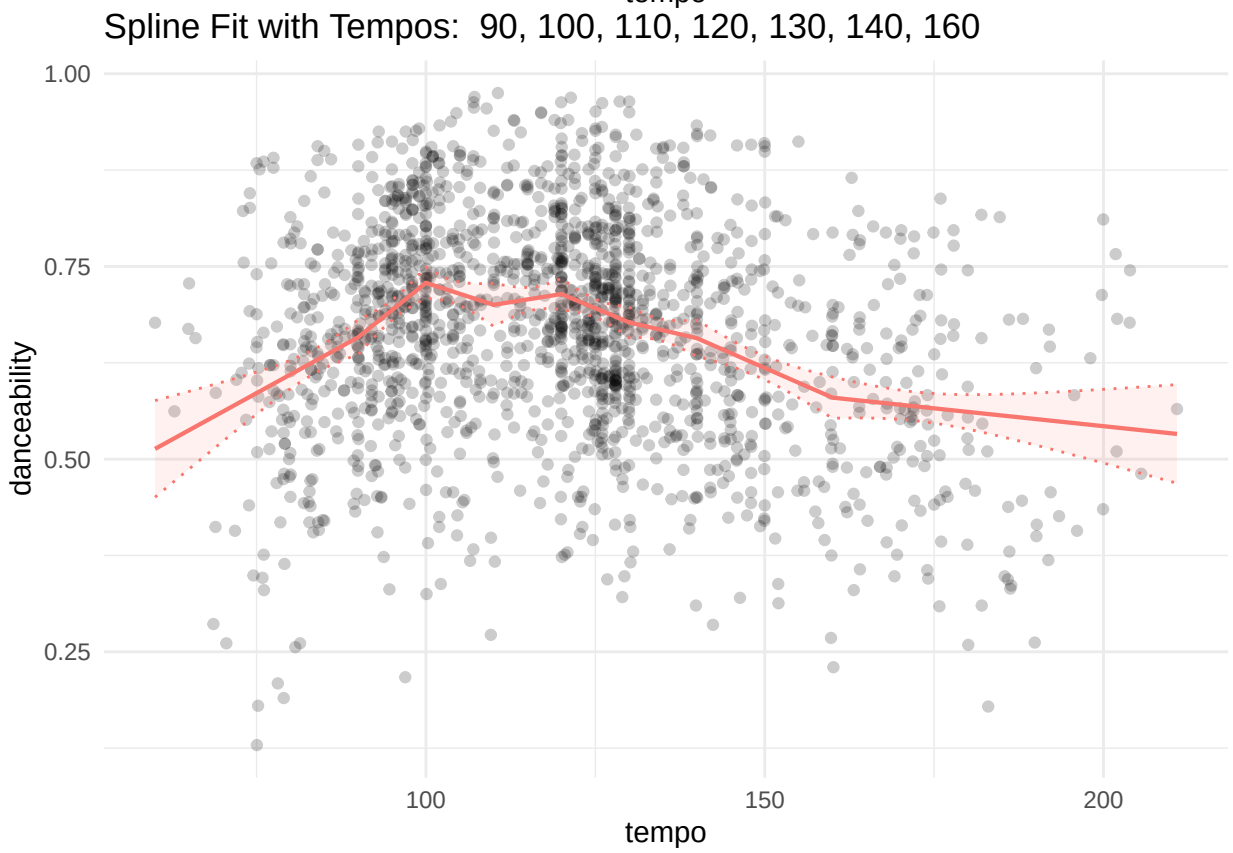
Spline Models

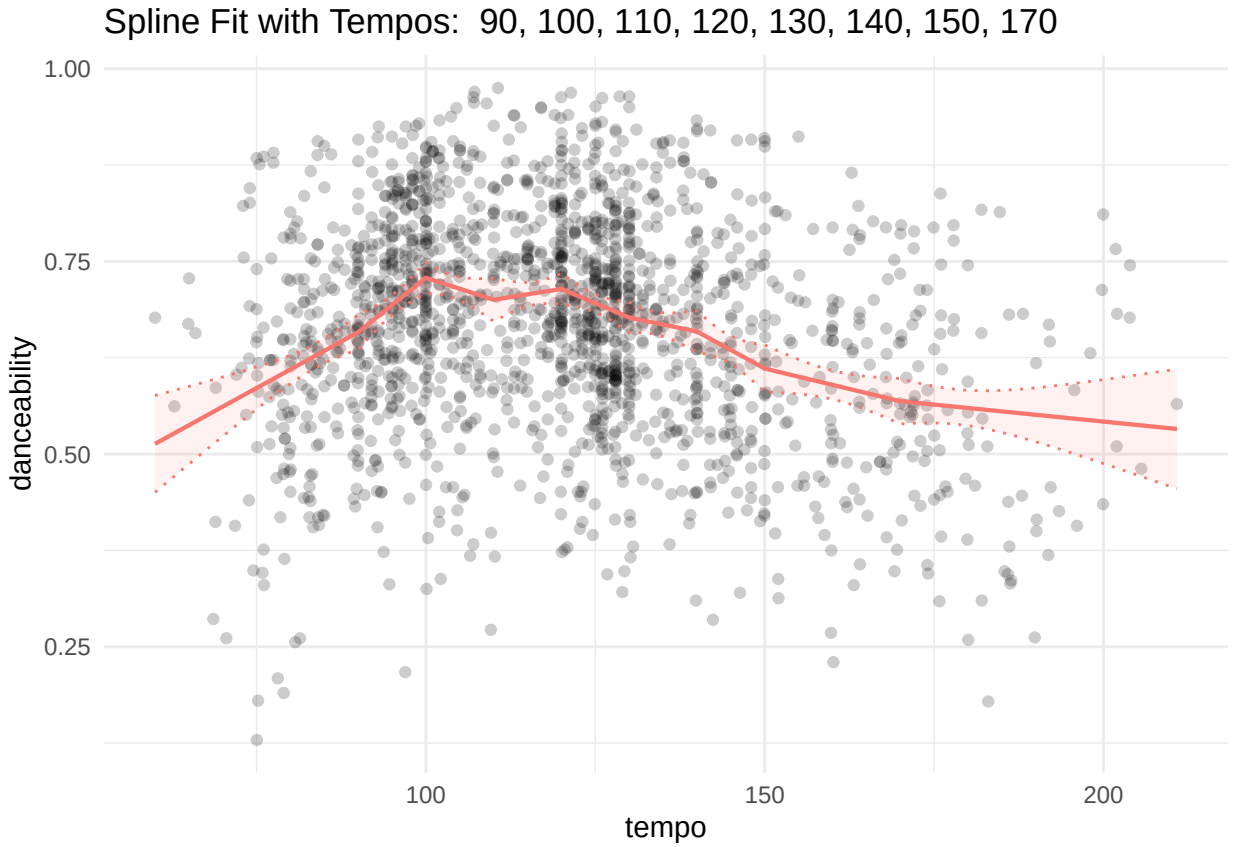
For the smoothing splines, we considered six models with 4, 5, 6, 7, 8, 9 knots respectively. Since we are not doing linear regression anymore (on the entire tempo data, rather on individual sections of the data), the individual linear models are a simple, univariate linear models (we are not transforming the response as we did prior):
Consider the stepwise linear plots below for the six spline models:

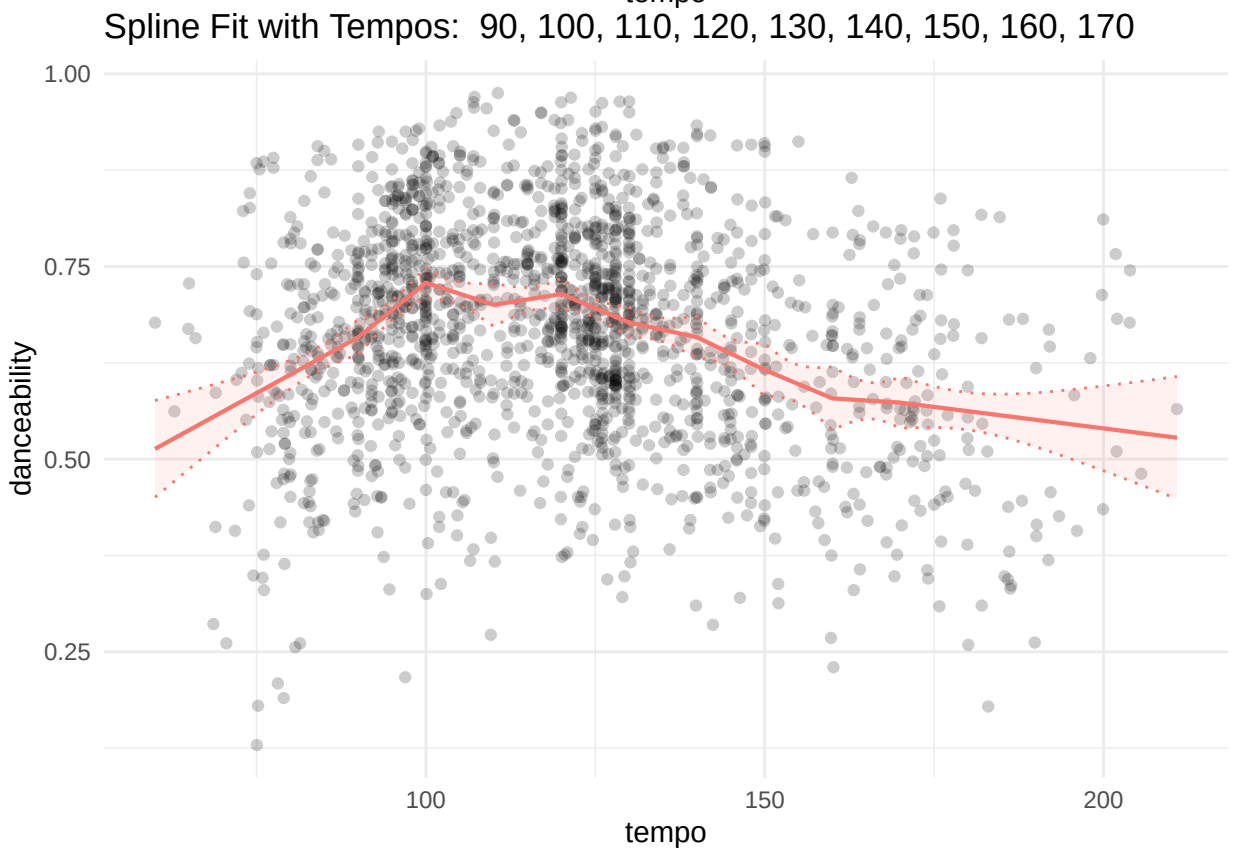

Spline Fit with Tempos: 100, 120, 130, 160
1.00
0.75
0.50
0.25
danceability
100
150
200
tempo


Spline Fit with Tempos: 100, 110, 120, 130, 160
1.00
0.75
0.50
0.25
danceability
100
150
200
tempo


Spline Fit with Tempos: 100, 110, 120, 130, 140, 160
1.00
0.75
0.50
0.25
danceability
100
150
200
tempo


Spline Fit with Tempos: 90, 100, 110, 120, 130, 140, 160
1.00
0.75
0.50
0.25
danceability
100
150
200
tempo





Below, consider the individual metrics for each of the splines bases in the spline models above.

```
## [1] "Order of Output: "
```

```
## Metrics for Spline Model with Knots: 100, 110, 120, 130, 140, 160
## # A tibble: 8 x 7
##   term          estimate std.error statistic   p.value conf.low conf.high
##   <chr>            <dbl>     <dbl>     <dbl>     <dbl>    <dbl>     <dbl>
## 1 (Intercept)     0.494     0.0214     23.1  4.07e-103   0.452      0.536
## 2 spline_basis1   0.229     0.0267      8.59 1.99e- 17   0.177      0.282
## 3 spline_basis2   0.208     0.0243      8.55 2.68e- 17   0.160      0.255
## 4 spline_basis3   0.220     0.0239      9.20 1.00e- 19   0.173      0.267
## 5 spline_basis4   0.183     0.0235      7.79 1.13e- 14   0.137      0.229
## 6 spline_basis5   0.163     0.0244      6.66 3.58e- 11   0.115      0.211
## 7 spline_basis6   0.0856    0.0252      3.39 7.08e-  4   0.0361     0.135
## 8 spline_basis7   0.0385    0.0385      1.00 3.17e-  1  -0.0370     0.114
## Metrics for Spline Model with Knots: 100, 110, 120, 130, 160
## # A tibble: 7 x 7
##   term          estimate std.error statistic   p.value conf.low conf.high
##   <chr>            <dbl>     <dbl>     <dbl>     <dbl>    <dbl>     <dbl>
## 1 (Intercept)     0.494     0.0214    23.1   3.63e-103   0.452      0.536
## 2 spline_basis1   0.229     0.0267     8.58  2.02e- 17   0.177      0.282
## 3 spline_basis2   0.208     0.0243     8.56  2.38e- 17   0.160      0.256
## 4 spline_basis3   0.218     0.0238     9.17  1.30e- 19   0.172      0.265
## 5 spline_basis4   0.187     0.0229     8.16  6.29e- 16   0.142      0.232
## 6 spline_basis5   0.0904    0.0244     3.70  2.20e-  4   0.0425     0.138
## 7 spline_basis6   0.0329    0.0377     0.871 3.84e-  1  -0.0411     0.107
## Metrics for Spline Model with Knots: 100, 120, 130, 160
## # A tibble: 6 x 7
##   term          estimate std.error statistic   p.value conf.low conf.high
##   <chr>            <dbl>     <dbl>     <dbl>     <dbl>    <dbl>     <dbl>
## 1 (Intercept)     0.499     0.0209    23.9   2.87e-109   0.458      0.540
## 2 spline_basis1   0.222     0.0255     8.70  7.37e- 18   0.172      0.272
## 3 spline_basis2   0.210     0.0219     9.57  3.45e- 21   0.167      0.252
## 4 spline_basis3   0.184     0.0226     8.11  9.18e- 16   0.139      0.228
## 5 spline_basis4   0.0856    0.0239     3.58  3.48e-  4   0.0387     0.132
## 6 spline_basis5   0.0289    0.0375     0.771 4.41e-  1  -0.0446     0.102
## Metrics for Spline Model with Knots: 90, 100, 110, 120, 130, 140, 150, 160, 170
## # A tibble: 11 x 7
##    term           estimate std.error statistic  p.value conf.low conf.high
##    <chr>             <dbl>     <dbl>     <dbl>    <dbl>    <dbl>     <dbl>
##  1 (Intercept)      0.513     0.0314    16.4   4.90e-56  0.452       0.575
##  2 spline_basis1    0.145     0.0384     3.78  1.64e- 4  0.0696      0.220
##  3 spline_basis2    0.215     0.0316     6.81  1.30e-11  0.153       0.277
##  4 spline_basis3    0.187     0.0348     5.38  8.52e- 8  0.119       0.255
##  5 spline_basis4    0.201     0.0329     6.10  1.30e- 9  0.136       0.266
##  6 spline_basis5    0.164     0.0329     4.98  7.05e- 7  0.0994      0.229
##  7 spline_basis6    0.145     0.0342     4.23  2.41e- 5  0.0778      0.212
##  8 spline_basis7    0.102     0.0358     2.86  4.24e- 3  0.0323      0.173
##  9 spline_basis8    0.0655    0.0375     1.75  8.05e- 2 -0.00796     0.139
## 10 spline_basis9    0.0600    0.0355     1.69  9.13e- 2 -0.00967     0.130
## 11 spline_basis10   0.0145    0.0506     0.287 7.74e- 1 -0.0847      0.114
## Metrics for Spline Model with Knots: 90, 100, 110, 120, 130, 140, 150, 170
## # A tibble: 10 x 7
##    term           estimate std.error statistic  p.value conf.low conf.high
```

```
##    <chr>            <dbl>     <dbl>     <dbl>    <dbl>    <dbl>     <dbl>
##  1 (Intercept)     0.513     0.0314    16.4   4.67e-56   0.452      0.575
##  2 spline_basis1   0.145     0.0383     3.78  1.64e- 4   0.0696     0.220
##  3 spline_basis2   0.215     0.0316     6.82  1.29e-11   0.153      0.277
##  4 spline_basis3   0.187     0.0347     5.38  8.49e- 8   0.119      0.255
##  5 spline_basis4   0.201     0.0329     6.10  1.28e- 9   0.136      0.266
##  6 spline_basis5   0.164     0.0329     4.98  7.13e- 7   0.0993     0.228
##  7 spline_basis6   0.146     0.0342     4.27  2.10e- 5   0.0787     0.213
##  8 spline_basis7   0.0977    0.0349     2.80  5.17e- 3   0.0293     0.166
##  9 spline_basis8   0.0553    0.0347     1.60  1.11e- 1  -0.0127     0.123
## 10 spline_basis9   0.0193    0.0499     0.387 6.99e- 1  -0.0786     0.117
## Metrics for Spline Model with Knots: 90, 100, 110, 120, 130, 140, 160
## # A tibble: 9 x 7
##   term          estimate std.error statistic  p.value  conf.low conf.high
##   <chr>            <dbl>     <dbl>     <dbl>    <dbl>     <dbl>     <dbl>
## 1 (Intercept)     0.513     0.0314    16.4   4.26e-56  0.452        0.575
## 2 spline_basis1   0.145     0.0383     3.78  1.63e- 4  0.0697       0.220
## 3 spline_basis2   0.215     0.0316     6.82  1.27e-11  0.153        0.277
## 4 spline_basis3   0.187     0.0347     5.38  8.36e- 8  0.119        0.255
## 5 spline_basis4   0.201     0.0329     6.10  1.28e- 9  0.136        0.265
## 6 spline_basis5   0.164     0.0329     4.99  6.73e- 7  0.0996       0.229
## 7 spline_basis6   0.144     0.0335     4.29  1.90e- 5  0.0779       0.209
## 8 spline_basis7   0.0665    0.0341     1.95  5.12e- 2 -0.000358     0.133
## 9 spline_basis8   0.0194    0.0448     0.433 6.65e- 1 -0.0684       0.107
```

In the spline models above, we can see how adding more knots makes the stepwise function found more complicated, and change the significance of certain ranges. When considering which of these spline models is likely best at prediction, we considered balancing a few different criteria: the ability interpolate for a large range of tempos, a model that has a significant coefficents for its bases, and a model that likely won't overfit the test data while still capturing the complexities of this relationship.

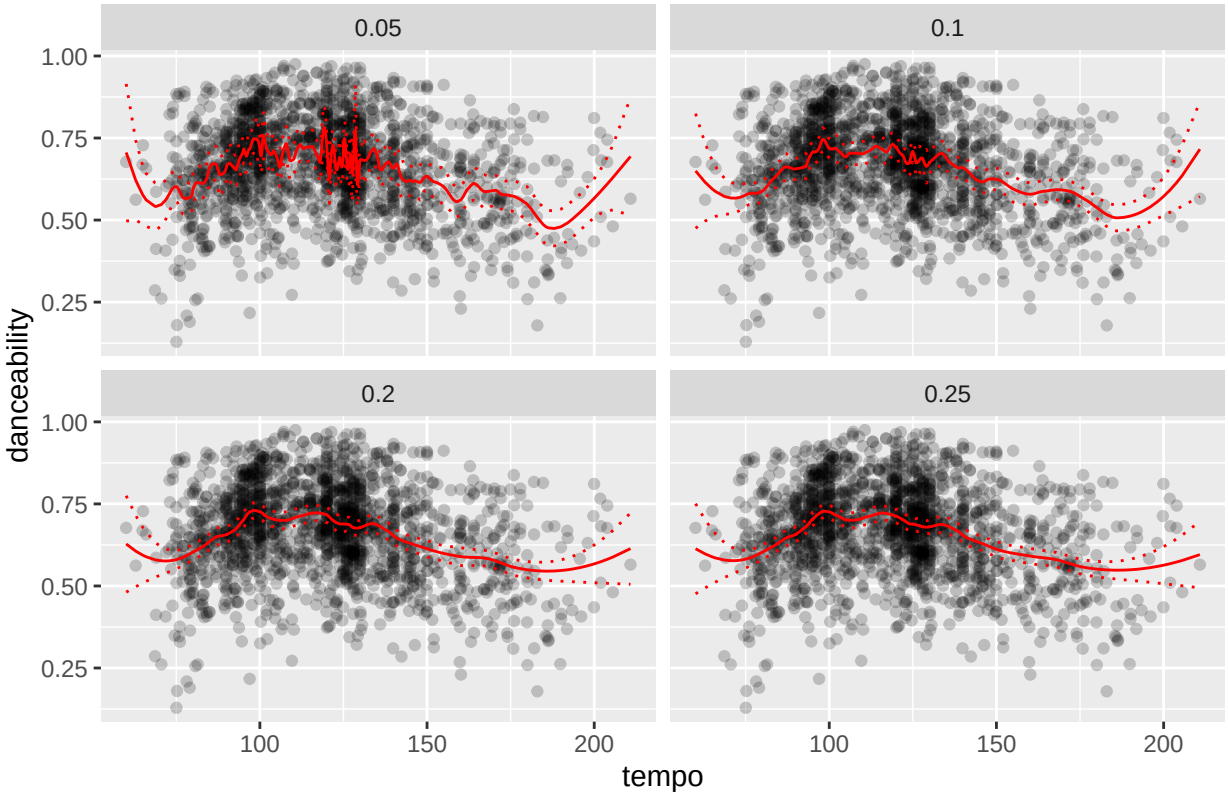
Every spline model we considered did not have a significant slope (a p-value under $\alpha = 0.01$) for the spline basis for the highest tempos. The model with 10 knots had three sections whose p-values were not below $\alpha = 0.01$ (also for the largest tempos) and two sections for 8 and 9 knots models had p-value not below α . For this reason, we did not consider these models for the best (also their significant levels for the coefficients and intercept of the bases were generally lower). The models with 4, 5 and 6 knots only had one section with whose relationship to danceability was not significant; again, this was the spline basis with the highest tempos. Of these model, we chose the spline models with 5 and 6 knots to be the best. Given our domain knowledge, we thought the tempo ranges represented by these knots actually have unique relationships with danceability. We believe this is a good compromise between modeling the complexities of this relationship, while still being general enough to accurate model unseen data. As stated previously, the last basis for every spline model did not have p-values that were below our signficant level. We suspect that the relationship between danceability and tempo is not linear within larger tempo ranges (≥ 160). Therefore, an area to consider for further exploration is to fit the last tempo section with more variables or some transformation of the tempo or danceability. For the sake of brevity, will not consider these additions now.

In summary, we believe the 5 and 6 knot spline models, given the metrics previously discuseed, would be the best for predicting new songs' danceability and we believe we can predict danceability in tempo ranges [min, 160] without extrapolating. Later, we will consider which of these models we consider to be better.

LOESS Models

For the LOESS models, we considered four models with spans of 5%, 10%, 20% and 25%:

Figure : Loess Fits for Various Spans



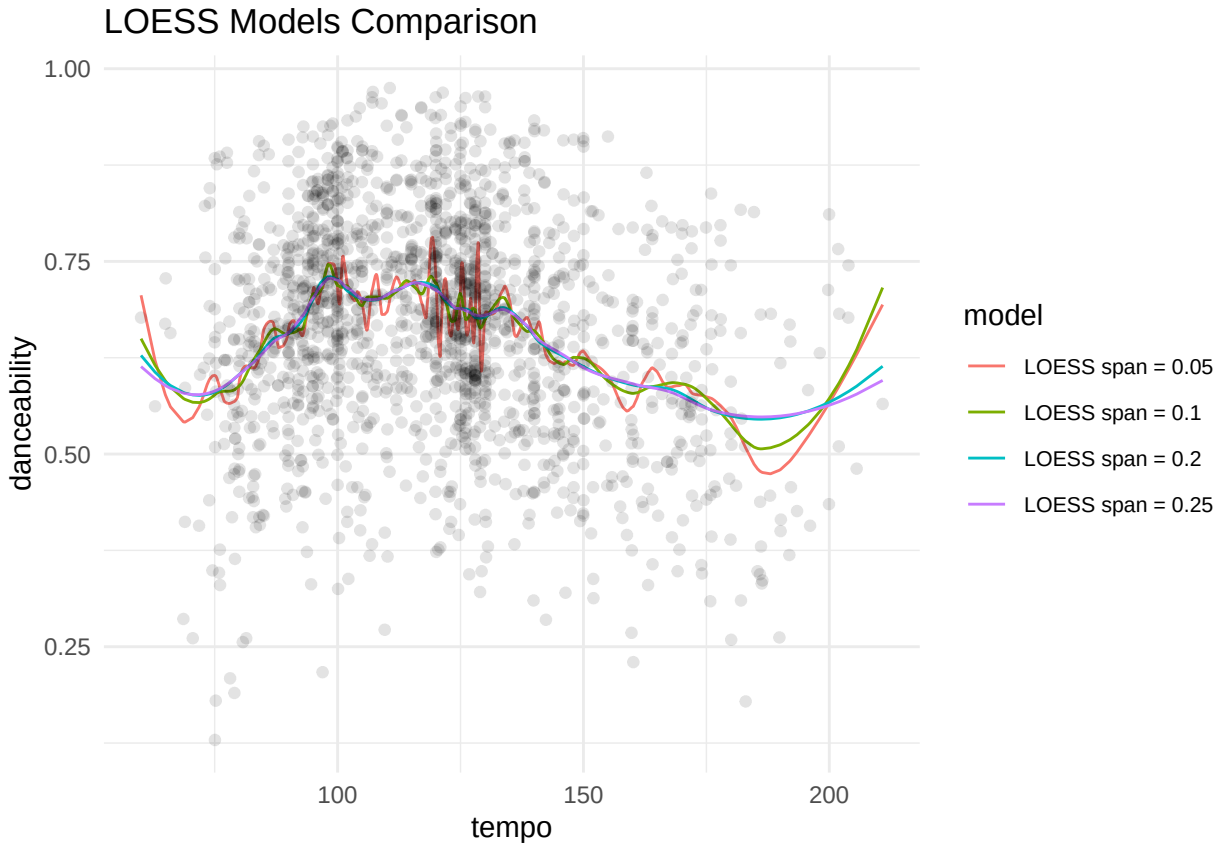
As we can see from the Loess fit figures above, as the span increases the model becomes smoother as each observation considers more neighbors to determine their fitted value. We determined that the %5 span model was clearly overfitting the training data. In the range of 120-130 BPM (tempo), the fitted values change wildly. We don't actually expect there to be much of a difference between the danceability songs with 133BPM tempo versus a song with 135 tempo, for example. Similarly, as most evident by the function slopes in the range 120-130, we believe the 10% span model is also capturing relationships that are not general (overfitting the training data).

The 20% and 35% span model are much smoother, and for that reason, we believe these models would be better at predicting new data. We also determined that the 25% span model may perform better at predicting new songs' danceability than the 20% span model. Again, although less severe than for the 5% and 10% span model, we still think that the 20% model is capture some changes in the relationship between danceability and tempo that wouldn't generalize well given our domain knowledge.

From the figure above, the 25% model appears to have three main "peaks" of danceability in relation to tempo: around tempos 100BPM, 120PM and 130BPM.

Comparing all Smoothing Models

Below you will see all the spline and LOESS models on two plots.



Now that we have considered which of the spline models and which of the LOESS models we believe is best (the 5 and 6 knot spline models and the 25% LOESS model), let's consider which of these models is best. We first determined that a spline model would work better to modeling the relationship between danceability and tempo than a LOESS model. Our reasoning is, first that we believe that the relationships between dancebility is genuinely linear for many tempo ranges (and the metric founds for the spline bases confirm that). Second, we know that music producers, if they have a dance song an the intention, will often specifically choose a tempo or range of tempo to achieve this. Furthermore, certain types of vocal performances (some of them more conducive to dance music) are easier to perform Therefore, we this domain knowledge in mind, we expect that there are some real, abrupt changes in the relationship between tempo and danceability; the spline model captures these abrupt changes at certain partitions better than LOESS (which smooths outs these changes).

We should note that one apparant advantage to the LOESS models is that they capture the non-linear relationship for the fast tempos, which the splines models could not. However, this could also be seen as a disdvantage for the very slow tempos. We see the same convex, quadratic relationship for the slower tempos as we do for the very fast tempos. Although our domain knowledge tells us this type of relationship may make sense for the fast tempos, it doesn't really make sense for the slow tempos. Aka. would you really see a sudden rise in danceability for very slow songs vs moderately slow song? Or, is the LOESS model simply being swayed by the small cluster of medium-danceable nodes at the very slow tempos (which could just be a unique feature of our data set or training data). Looking into the exact relationship between danceability and tempo at these slow tempos, and considering the specifics of those very slow, but danceable songs, would be a good area of exploration. However, our intuition and domain knowledge tell us that this sudden rise in predicted danceability at the very slow tempos is likely not indicitive of the true relationship at these tempos. We believe the linear relationships modeled by the spline bases at these tempos are more suitable. For all the reasons above, we believe the a spline model is more suitable to model this relationship. As for choosing between the 5 and 6 knot spline model, if we look at the stepwise function plots, we can see that, even with more than 3-4 knots, the stepwise functions only show about 4-5 significantly different slope areas. Therefore, we expect that one of the knots for the 6 knot model is unnecessary (specifially the 140BPM knot) as the

relationships between tempo and danceability in the bases straddling this knot (aka. ranges [130, 140] and [140,160]) are roughly the same.

In summary, out of all the smoothing models, we determined that the 5 knot model would be the best model at predicting new observations. We determined this by considering the unique properties of tempo, our ability to interpolate for a large range of tempos, and balancing the desire not to overfit while also capturing the complexities of this relationship. Some futher area of exploration is finding a better, non-linear, and/or mulit-variate model to predict the danceability of very fast tempos (which our best chosen model couldn't capture well with a univariate linear model) and looking deeper into the very slow, but danceable, songs to see if a linear relationship at these tempos is truly appropriate.

Summary

Within this project, our study aimed to find the the factors that influence the "danceability" of a song from a combination of audio features within the song, its popularity, genre, and other characteristics. We hope our findings will help musical artists and composers have a better understanding on "danceability" and its connection to the music world.

We aimed to create a valuable statistical model danceability as our main variable. Our first step in the process was preprocessing, which included feature engineering, transforming certain variables to fit the assumptions of the normal errors models, and checking for multicollinearity and high influence/leverage point. When it came to building our model, we employed stepwise selection models with f-tests to find the optimal variables to use for our model.

We found that the Forward Selected Stepwise BIC exhibited a better adjusted R^2, AIC, and BIC metrics, as well significant p-values to variable coefficients, indicating better suitability to predict/interpret danceability. Our final model indicates variables like valence, instrumentalness, and genre selection proved valuable positive effects on danceability while energy and tempo showed negative effects.

Further exploration could include the consideration of other model selection techniques such as Lasso or Ridge regression in order to select important predictors for our model better. We could also include a more deeper analysis on the effect of multicollinearity and potential outliers. We may also explore different shrinkage and smoothing methods. Overall, our current model can be seem as valuable for now based on the accuracy of our model and the precaution taken to prevent overfitting and overcomplexity.