Final Report Math 158

Luke Feng

2024-11-30

Abstract

This study examined the relationship between musical attributes and track popularity on Spotify using a dataset of over 160,000 tracks from 1921–2020. Regression analysis revealed that attributes like danceability and energy positively impact popularity, while valence, acousticness, and speechiness have negative effects. While traditional regression models were limited by violations of normality and heteroscedasticity, the combination of a Box-Cox transformation and robust regression significantly improved predictive accuracy, achieving consistent RMSE values of ~4.5 across training and test sets. These results emphasize the influence of rhythmic and energetic features on popularity, with opportunities for future research on genre-specific models and broader contextual factors.

Introduction

As streaming platforms like Spotify amass vast amounts of data on user preferences, analyzing musical attributes to understand what makes a track popular has become increasingly viable.

We consider the following research question: Can we predict a track's popularity based on its musical attributes?

To address these questions, we analyze a dataset focusing on Spotify's computed attributes such as danceability, energy, and valence. By investigating correlations and applying predictive models, we aim to uncover the relationships between these musical features and their impact on popularity. The primary focus of this study is to develop a predictive model for track popularity based on musical attributes, rather than to make inferential claims about these relationships.

Data Set

The Spotify dataset consists of 160,000+ tracks from 1921-2020 found in Spotify as of June 2020. The data was collected by Turkish Data Scientist Yamaç Eren Ay, and was tabulated and retrieved from the Spotify Web API. Each row in the dataset corresponds to a track, each with variables such as Track ID, title, artist, and release data. In addition to these variables, some musical features were extracted, such as dancability, energy, and acousticness were extracted, and the value for each of these variables was calculated by Spotify based on a range of parameters.

Let's give a sample of the dataset.

```
spotify_data <- read_csv("/Users/lukefeng/Downloads/data.csv")
```

```
## Rows: 169909 Columns: 19
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (4): id, name, artists, release_date
## dbl (15): duration_ms, year, acousticness, danceability, energy, instrumenta...
##
## i Use ‘spec()‘ to retrieve the full column specification for this data.
## i Specify the column types or set ‘show_col_types = FALSE‘ to quiet this message.
```

```
head(spotify_data)
```

```
## # A tibble: 6 x 19
##   id      name  artists duration_ms release_date  year acousticness danceability
##   <chr>   <chr> <chr>         <dbl> <chr>        <dbl>        <dbl>        <dbl>
## 1 6KbQ3u~ Sing~ ['Carl~      158648 1928          1928        0.995        0.708
## 2 6KuQTI~ Fant~ ['Robe~      282133 1928          1928        0.994        0.379
## 3 6L63VW~ Chap~ ['Sewe~      104300 1928          1928        0.604        0.749
## 4 6M94Fk~ Beba~ ['Fran~      180760 9/25/28       1928        0.995        0.781
## 5 6N6tiF~ Polo~ ['Fréd~      687733 1928          1928        0.99         0.21
## 6 6NxAf7~ Sche~ ['Feli~      352600 1928          1928        0.995        0.424
## # i 11 more variables: energy <dbl>, instrumentalness <dbl>, liveness <dbl>,
## #   loudness <dbl>, speechiness <dbl>, tempo <dbl>, valence <dbl>, mode <dbl>,
## #   key <dbl>, popularity <dbl>, explicit <dbl>
```

```
summary(spotify_data)
```

```
##       id                name              artists           duration_ms
##  Length:169909      Length:169909      Length:169909      Min.   :   5108
##  Class :character   Class :character   Class :character   1st Qu.: 171040
##  Mode  :character   Mode  :character   Mode  :character   Median : 208600
##                                                           Mean   : 231406
##                                                           3rd Qu.: 262960
##                                                           Max.   :5403500
##  release_date            year       acousticness     danceability
##  Length:169909      Min.   :1921   Min.   :0.0000   Min.   :0.0000
##  Class :character   1st Qu.:1957   1st Qu.:0.0945   1st Qu.:0.4170
##  Mode  :character   Median :1978   Median :0.4920   Median :0.5480
##                     Mean   :1977   Mean   :0.4932   Mean   :0.5381
##                     3rd Qu.:1999   3rd Qu.:0.8880   3rd Qu.:0.6670
##                     Max.   :2020   Max.   :0.9960   Max.   :0.9880
##      energy       instrumentalness      liveness         loudness
##  Min.   :0.0000   Min.   :0.000000   Min.   :0.0000   Min.   :-60.000
##  1st Qu.:0.2630   1st Qu.:0.000000   1st Qu.:0.0984   1st Qu.:-14.470
##  Median :0.4810   Median :0.000204   Median :0.1350   Median :-10.474
##  Mean   :0.4886   Mean   :0.161937   Mean   :0.2067   Mean   :-11.370
##  3rd Qu.:0.7100   3rd Qu.:0.086800   3rd Qu.:0.2630   3rd Qu.: -7.118
##  Max.   :1.0000   Max.   :1.000000   Max.   :1.0000   Max.   :  3.855
##   speechiness          tempo           valence            mode
##  Min.   :0.00000   Min.   :  0.00   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.03490   1st Qu.: 93.52   1st Qu.:0.3220   1st Qu.:0.0000
##  Median :0.04500   Median :114.78   Median :0.5440   Median :1.0000
##  Mean   :0.09406   Mean   :116.95   Mean   :0.5321   Mean   :0.7086
##  3rd Qu.:0.07540   3rd Qu.:135.71   3rd Qu.:0.7490   3rd Qu.:1.0000
```

```
##  Max.   :0.96900   Max.   :244.09   Max.   :1.0000   Max.   :1.0000
##       key           popularity        explicit
##  Min.   : 0.000   Min.   :  0.00   Min.   :0.00000
##  1st Qu.: 2.000   1st Qu.: 12.00   1st Qu.:0.00000
##  Median : 5.000   Median : 33.00   Median :0.00000
##  Mean   : 5.201   Mean   : 31.56   Mean   :0.08486
##  3rd Qu.: 8.000   3rd Qu.: 48.00   3rd Qu.:0.00000
##  Max.   :11.000   Max.   :100.00   Max.   :1.00000
```

Each row contains the variables for one track.

Below is a list of all variables:

Identifiers:

- id: A unique identifier for each track.
- name: The track's title.
- artists: Artist/s who participated in the track.
- duration_ms: Length of the track in milliseconds (ms).
- release_date: The track's release date in MM/DD/YYYY, or at the minimum, YYYY.
- year: The year in which the track was released.

Musical Attributes:

- **Acousticness**: Likelihood of a track being acoustic (0.0 to 1.0).
- **Danceability**: Suitability of a track for dancing (0.0 to 1.0).
- **Energy**: Intensity and activity level of a track (0.0 to 1.0).
- **Instrumentalness**: Likelihood of a track containing no vocals.
- **Liveness**: Presence of an audience sound in the recording.
- **Loudness**: Overall loudness of a track in decibels (dB).
- **Speechiness**: Presence of spoken words.
- **Tempo**: Beats per minute (BPM) of the track.
- **Valence**: Positiveness conveyed by the track (0.0 to 1.0).
- **Mode**: Melodic modality (0 = Minor, 1 = Major).
- **Key**: Key signature estimated for the track.
- **Popularity**: A score from 0 to 100 based on Spotify's algorithm, where higher values indicate greater popularity.
- **Explicit**: Flag indicating explicit content (1 = True, 0 = False).

For simplicity's sake, we deleted some irrelevant columns:

- id: No further analysis can be drawn from the track's unique identifier
- release_date: Since some of the data is incomplete (e.g. no date, no month), this will be dropped as it will be impossible to draw annual trends for time series
- liveness: This is just a measure of whether or not a track was performed lvie or not, thus not being a direct measure of the music itself

Additionally, for our analysis, we are only using the variables listed under Musical Attributes, as those are the only variables we are interested in.

Analysis

The analysis began with an exploratory data analysis (EDA) phase to understand the dataset and uncover patterns between musical attributes and track popularity. Musical Attributes such as danceability, energy, and valence were selected based on their possible influence on popularity.

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

```
## corrplot 0.95 loaded
```

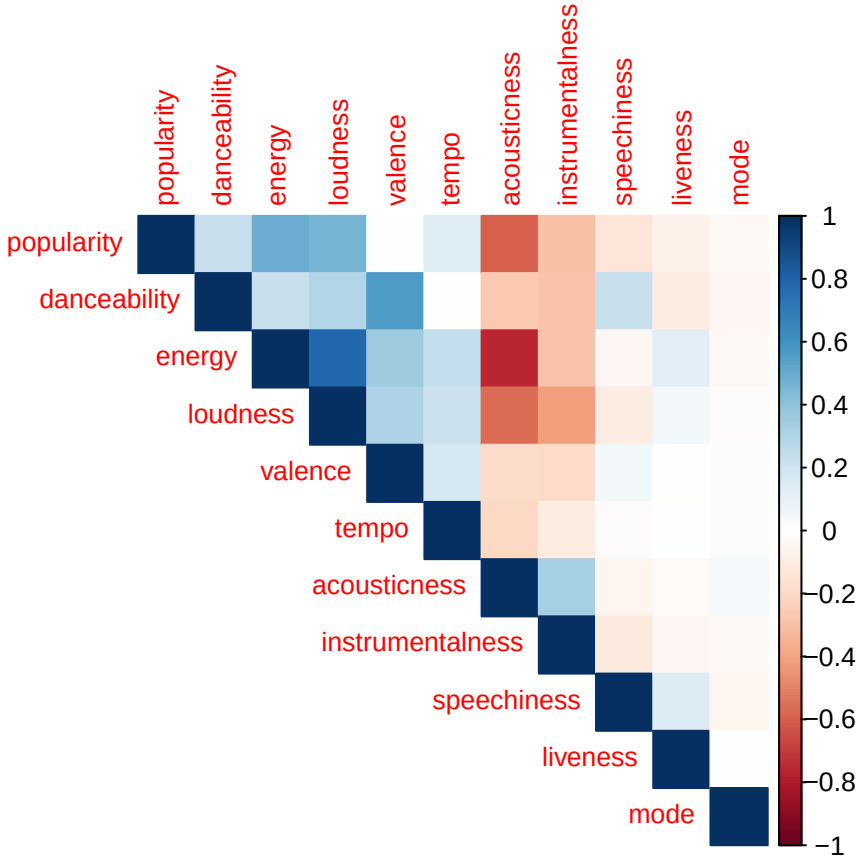


Figure 1: A correlation heatmmap to display correlations between popularity and other attributes to assess relationships.

We constructed a correlation matrix to explore and visualize the relationships between variables, which revealed that attributes such as danceability, energy, and loudness were positively correlated with popularity, while acousticness and speechiness were negatively correlated. This initial exploration also hinted at a potential multicollinearity among variables like energy and loudness.

Following this EDA, we fitted an initial linear regression model with popularity as the response and the selected musical attributes as the predictors.

```
model <- lm(popularity ~ danceability + energy + loudness + valence + tempo + acousticness + instrumenta

summary(model)
```

```
##
## Call:
## lm(formula = popularity ~ danceability + energy + loudness +
##     valence + tempo + acousticness + instrumentalness + speechiness +
##     mode + key, data = train_set)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -66.421 -11.914  -2.149  10.305  81.792
##
## Coefficients:
##                    Estimate Std. Error  t value Pr(>|t|)
## (Intercept)       40.231397   0.407349   98.764  < 2e-16 ***
## danceability      28.499843   0.303574   93.881  < 2e-16 ***
## energy             9.397669   0.321830   29.201  < 2e-16 ***
## loudness           0.337024   0.012388   27.205  < 2e-16 ***
## valence          -23.045977   0.199980 -115.241  < 2e-16 ***
## tempo              0.027392   0.001356   20.196  < 2e-16 ***
## acousticness     -23.821710   0.170915 -139.378  < 2e-16 ***
## instrumentalness  -6.790559   0.147977  -45.889  < 2e-16 ***
## speechiness      -27.611056   0.280192  -98.543  < 2e-16 ***
## mode              -0.379915   0.088141   -4.310 1.63e-05 ***
## key               -0.001461   0.011337   -0.129    0.897
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 15.92 on 161898 degrees of freedom
## Multiple R-squared:  0.4554, Adjusted R-squared:  0.4554
## F-statistic: 1.354e+04 on 10 and 161898 DF,  p-value: < 2.2e-16
```

Since Key was not statistically significant, we removed it from the model.

To assess the validity of regression assumptions, several diagnostic tests were conducted:

Normality of Residuals: A Q-Q plot of the residuals was generated, revealing deviations from normality. This finding was confirmed using the Shapiro-Wilk test, which returned a low p-value, indicating non-normal residuals.

```
model_sample <- sample(model$residuals, 5000)
shapiro.test(model_sample)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  model_sample
## W = 0.97989, p-value < 2.2e-16
```

Homoscedasticity: The Breusch-Pagan (Non-constant Variance Score) test was performed, yielding a significant p-value, confirming heteroscedasticity in the residuals.

```
ncvTest(model)
```

```
## Non-constant Variance Score Test
## Variance formula: ~ fitted.values
## Chisquare = 1701.743, Df = 1, p = < 2.22e-16
```

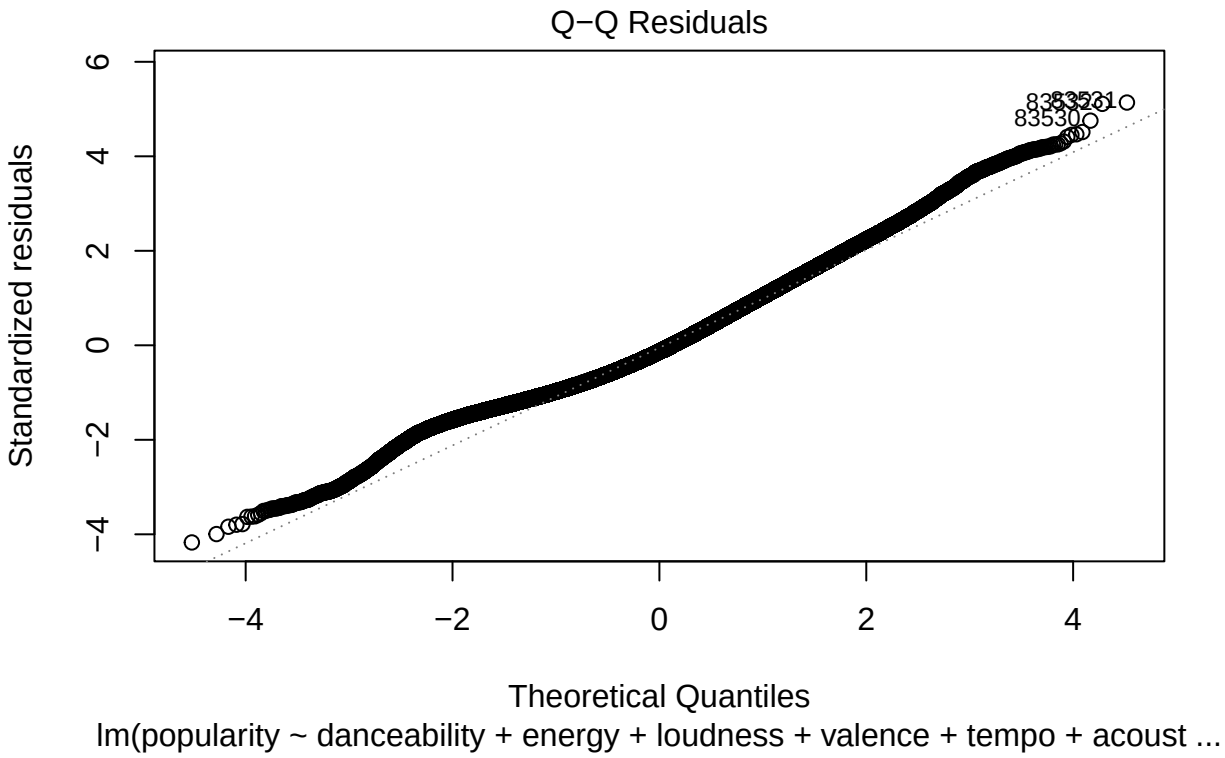


Figure 2: QQ Plot to check for Normality.

However, assumptions of homoscedasticity (constant variance of residuals) and normality of residuals are primarily important for inference-based models, where the goal is to estimate coefficients and calculate reliable p-values and confidence intervals. These assumptions ensure the validity of statistical tests and the interpretability of standard errors.

In our case (a predictive model) however, the focus is on minimizing prediction error and improving generalization to new data. The distribution or variance of residuals does not directly affect the model's ability to make accurate predictions. As long as the model captures the underlying patterns in the data and generalizes well to unseen data (e.g., low RMSE on the test set), violations of these assumptions do not compromise the model's utility for prediction.

```
calculate_rmse <- function(actual, predicted) {
  sqrt(mean((actual - predicted)^2))
}

train_predictions <- predict(model, newdata = train_set)
rmse_train <- calculate_rmse(train_set$popularity, train_predictions)

test_predictions <- predict(model, newdata = test_set)
rmse_test <- calculate_rmse(test_set$popularity, test_predictions)

cat("RMSE on Training Set:", rmse_train, "\n")
```

```
## RMSE on Training Set: 15.92208
```

```
cat("RMSE on Test Set:", rmse_test, "\n")
```

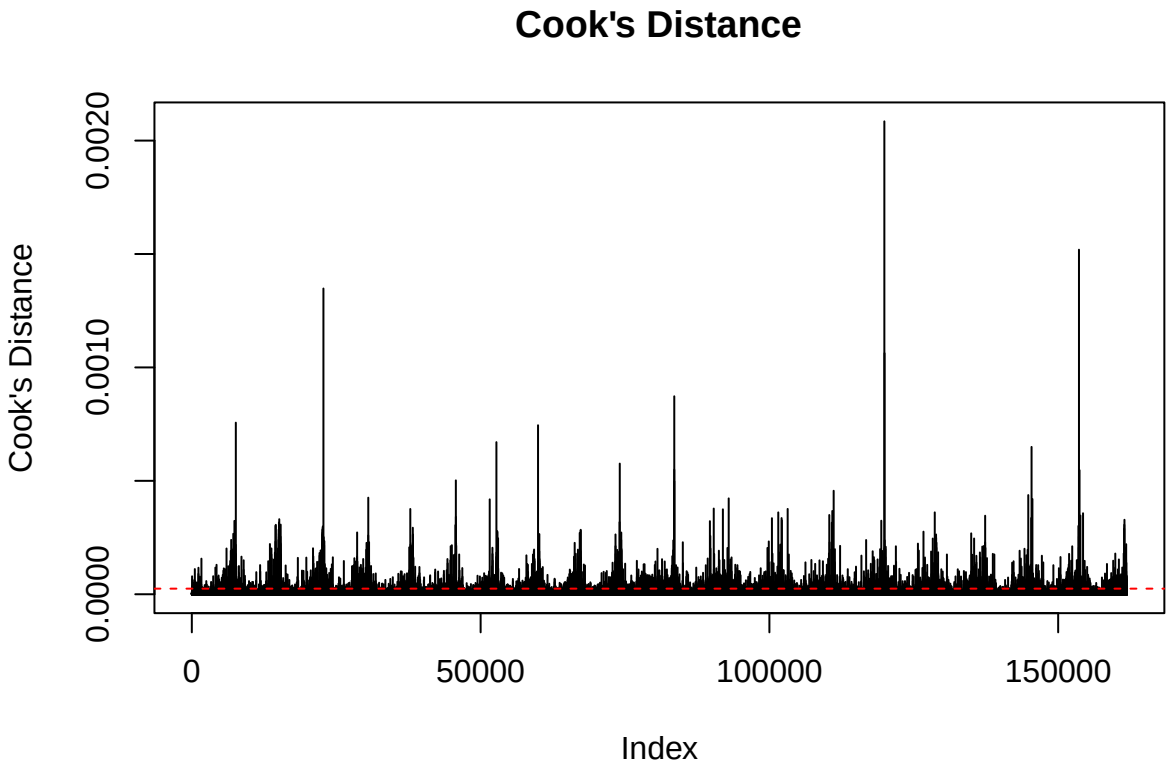
```
## RMSE on Test Set: 16.05234
```

RMSE Training Set: 15.87012 RMSE Test Set: 16.00235 We then checked for influential points:

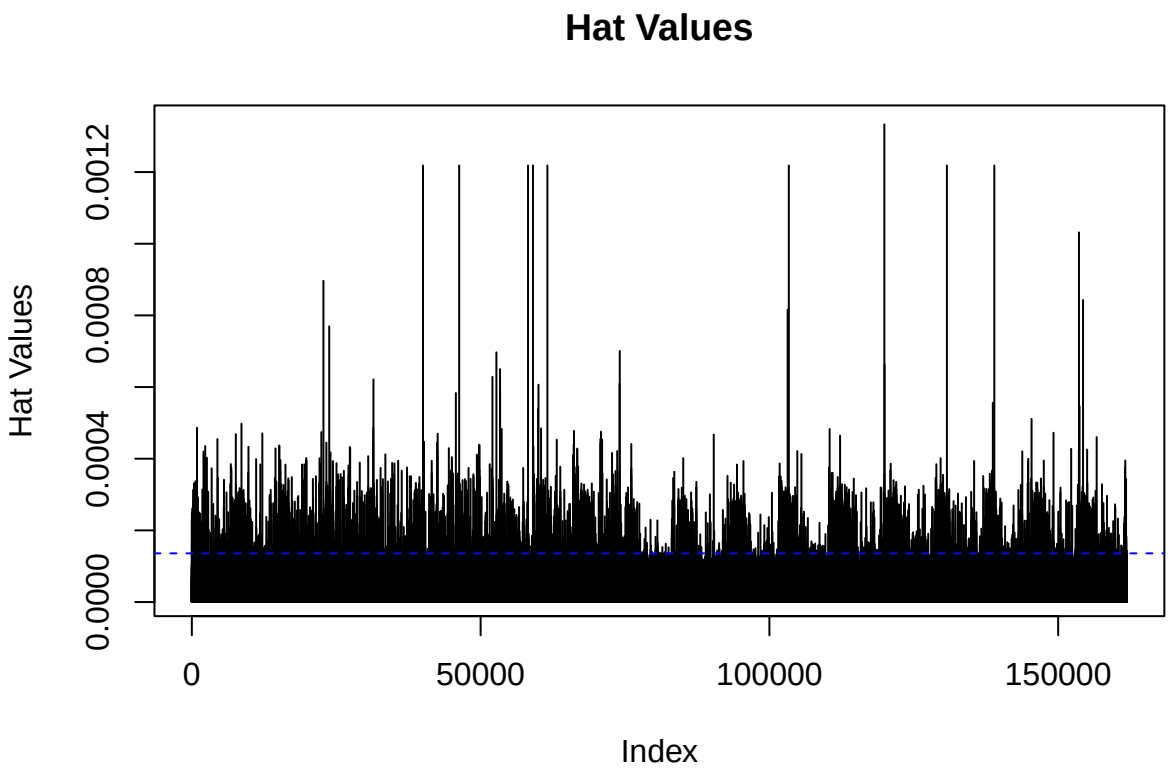
```
cooks_threshold <- 4 / length(cooks.distance(model))
high_cooks <- cooks.distance(model) > cooks_threshold

hat_threshold <- 2 * mean(hatvalues(model))
high_leverage <- hatvalues(model) > hat_threshold

high_residuals <- abs(rstandard(model)) > 2
influential_points <- which(high_cooks | high_leverage | high_residuals)

influential_points
```

Here we saw with 4/N as the threshold, about 15,000+ datapoints were omitted.

```
plot(cooks.distance(model), type = "h", main = "Cook's Distance", ylab = "Cook's Distance")
abline(h = cooks_threshold, col = "red", lty = 2)
```




```
plot(hatvalues(model), type = "h", main = "Hat Values", ylab = "Hat Values")
abline(h = hat_threshold, col = "blue", lty = 2)
```



After looking at the plots, we adjust the threshold to around 0.0002 because the size of our dataset makes 4/n extremely small, flagging many datapoints as influential.

```
high_cooks <- which(cooks.distance(model) > 0.0002)
high_cooks
```

```
##   6006   6817   7182   7285   7374   7378   7380   7469   7606   7617  13522
##   6006   6817   7182   7285   7374   7378   7380   7469   7606   7617  13522
##  13778  14479  14487  14543  14925  15016  15042  15066  15073  15085  15087
##  13778  14479  14487  14543  14925  15016  15042  15066  15073  15085  15087
##  15091  15093  15100  15104  15105  15139  15315  15343  15351  15355  15381
##  15091  15093  15100  15104  15105  15139  15315  15343  15351  15355  15381
##  20983  22626  22629  22630  22633  22634  22645  22651  22658  22661  22664
##  20983  22626  22629  22630  22633  22634  22645  22651  22658  22661  22664
##  22684  22747  22784  22826  22902  22942  28619  30251  30519  30529  30542
##  22684  22747  22784  22826  22902  22942  28619  30251  30519  30529  30542
##  30549  30684  37645  37826  37851  37859  37949  38239  38260  38261  38292
##  30549  30684  37645  37826  37851  37859  37949  38239  38260  38261  38292
##  44818  45022  45637  45713  45719  45747  45772  51576  52045  52735  52871
##  44818  45022  45637  45713  45719  45747  45772  51576  52045  52735  52871
##  52902  52926  53006  53020  59941  60015  60170  66288  67160  67320  67331
##  52902  52926  53006  53020  59941  60015  60170  66288  67160  67320  67331
##  73354  73355  74057  74094  74339  74353  80622  83366  83530  83531  83532
##  73354  73355  74057  74094  74339  74353  80622  83366  83530  83531  83532
##  85008  89699  89767  89790  90364  91950  92741  92953  99668  99895 100441
##  85008  89699  89767  89790  90364  91950  92741  92953  99668  99895 100441
```

```
## 101538 101893 101952 102112 102162 103165 110372 110432 110806 110897 110934
## 101538 101893 101952 102112 102162 103165 110372 110432 110806 110897 110934
## 111127 112243 116750 119372 119903 119910 119932 119959 120143 121900 125717
## 111127 112243 116750 119372 119903 119910 119932 119959 120143 121900 125717
## 126680 128292 128631 128693 128854 128869 128908 128923 134945 135446 136863
## 126680 128292 128631 128693 128854 128869 128908 128923 134945 135446 136863
## 137135 137348 137357 137390 144167 144824 145333 145391 145541 152431 153502
## 137135 137348 137357 137390 144167 144824 145333 145391 145541 152431 153502
## 153574 153586 153601 153700 153721 153725 153852 154308 161432 161452 161480
## 153574 153586 153601 153700 153721 153725 153852 154308 161432 161452 161480
## 161591 161775
## 161591 161775
```

```
train_cleaned <- train_set[-high_cooks, ]
```

```
model_cleaned <- lm(popularity ~ danceability + energy + loudness + valence + tempo + acousticness + ins
summary(model_cleaned)
```

```
##
## Call:
## lm(formula = popularity ~ danceability + energy + loudness +
##     valence + tempo + acousticness + instrumentalness + speechiness +
##     liveness + mode + key, data = train_cleaned)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -63.771 -11.794  -2.088  10.236  69.756
##
## Coefficients:
##                   Estimate Std. Error  t value Pr(>|t|)
## (Intercept)       4.228e+01  4.071e-01  103.867  < 2e-16 ***
## danceability      2.720e+01  3.044e-01   89.344  < 2e-16 ***
## energy            9.931e+00  3.256e-01   30.503  < 2e-16 ***
## loudness          3.945e-01  1.244e-02   31.711  < 2e-16 ***
## valence          -2.279e+01  1.983e-01 -114.899  < 2e-16 ***
## tempo             2.737e-02  1.347e-03   20.312  < 2e-16 ***
## acousticness     -2.331e+01  1.708e-01 -136.471  < 2e-16 ***
## instrumentalness -7.038e+00  1.470e-01  -47.872  < 2e-16 ***
## speechiness      -2.577e+01  2.835e-01  -90.884  < 2e-16 ***
## liveness         -7.515e+00  2.326e-01  -32.313  < 2e-16 ***
## mode             -3.577e-01  8.737e-02   -4.094 4.24e-05 ***
## key              -6.466e-04  1.124e-02   -0.058    0.954
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 15.78 on 161730 degrees of freedom
## Multiple R-squared:  0.4648, Adjusted R-squared:  0.4648
## F-statistic: 1.277e+04 on 11 and 161730 DF,  p-value: < 2.2e-16
```

We saw a minimal increase in R^2, from 0.459 to 0.464 Let's see if it improves any of the regression assumptions:

```r
model_sample <- sample(model_cleaned$residuals, 5000)
shapiro.test(model_sample)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  model_sample
## W = 0.98148, p-value < 2.2e-16
```

```r
ncvTest(model_cleaned)
```

```
## Non-constant Variance Score Test
## Variance formula: ~ fitted.values
## Chisquare = 1328.881, Df = 1, p = < 2.22e-16
```

Unfortunately, the p-values are still extremely low, let's see if it at least improves our RMSE:

```r
calculate_rmse <- function(actual, predicted) {
  sqrt(mean((actual - predicted)^2))
}

train_predictions <- predict(model, newdata = train_cleaned)
rmse_train <- calculate_rmse(train_set$popularity, train_predictions)
```

```
## Warning in actual - predicted: longer object length is not a multiple of
## shorter object length
```

```r
test_predictions <- predict(model, newdata = test_set)
rmse_test <- calculate_rmse(test_set$popularity, test_predictions)

cat("RMSE on Training Set:", rmse_train, "\n")
```

```
## RMSE on Training Set: 16.6775
```

```r
cat("RMSE on Test Set:", rmse_test, "\n")
```

```
## RMSE on Test Set: 16.05234
```

It did not improve our RMSE or R^2 or any of our regression assumptions by a significant amount. Thus, we decide not to remove influential points so as to avoid losing valuable information or skewing our dataset.

We move forward with the original dataset:

To assess multicollinearity among predictors, the Variance Inflation Factor (VIF) was calculated after fitting the model.

```r
vif_values <- vif(model)
print(vif_values)
```

```
##     danceability           energy         loudness          valence
##         1.808554         4.730206         3.149463         1.759262
##            tempo     acousticness instrumentalness      speechiness
##         1.107705         2.645966         1.337665         1.129289
##             mode              key
##         1.026205         1.014377
```

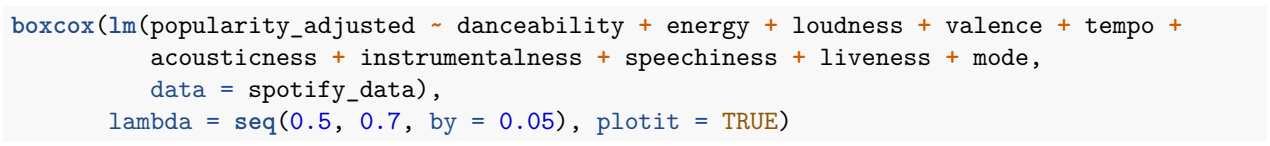
The VIF results indicated that multicollinearity was within acceptable limits, allowing all predictors to remain in the model. This step ensured the stability of coefficient estimates.

To address the diagnostic findings and potentially enhance the model's predictive accuracy, a Box-Cox transformation was applied to the response variable. While violations of normality and heteroscedasticity are less critical for predictive models, the transformation was explored to assess whether it could improve the linear relationship between the predictors and the response, thereby reducing prediction error

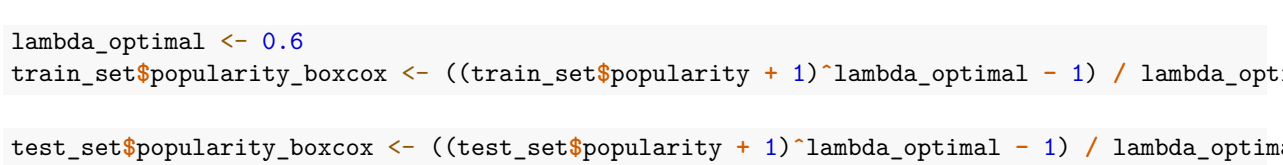
```
library(MASS)
```

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

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

```
spotify_data$popularity_adjusted <- spotify_data$popularity + 1  # Adding 1 to ensure positivity

boxcox(lm(popularity_adjusted ~ danceability + energy + loudness + valence + tempo +
          acousticness + instrumentalness + speechiness + liveness + mode,
          data = spotify_data),
       lambda = seq(-0.2, 0.6, by = 0.1), plotit = TRUE)
```



```
boxcox(lm(popularity_adjusted ~ danceability + energy + loudness + valence + tempo +
          acousticness + instrumentalness + speechiness + liveness + mode,
          data = spotify_data),
       lambda = seq(0.5, 0.7, by = 0.05), plotit = TRUE)
```



To ensure the response variable was positive, a small constant (+1) was added to all values before applying the Box-Cox transformation. A range of potential values for the Box-Cox parameter, lambda, was evaluated, starting with a broad search across [-0.2,0.6]. The log-likelihood plot suggested an optimal value of lambda = 0.6, which was then refined by narrowing the range for more precision.

```
lambda_optimal <- 0.6
train_set$popularity_boxcox <- ((train_set$popularity + 1)^lambda_optimal - 1) / lambda_optimal
```

```
test_set$popularity_boxcox <- ((test_set$popularity + 1)^lambda_optimal - 1) / lambda_optimal
```

```
lm_train <- lm(popularity_boxcox ~ danceability + energy + loudness + valence + tempo +
               acousticness + instrumentalness + speechiness + liveness + mode,
               data = train_set)
```

Now we test the transformed model for the normality assumption.

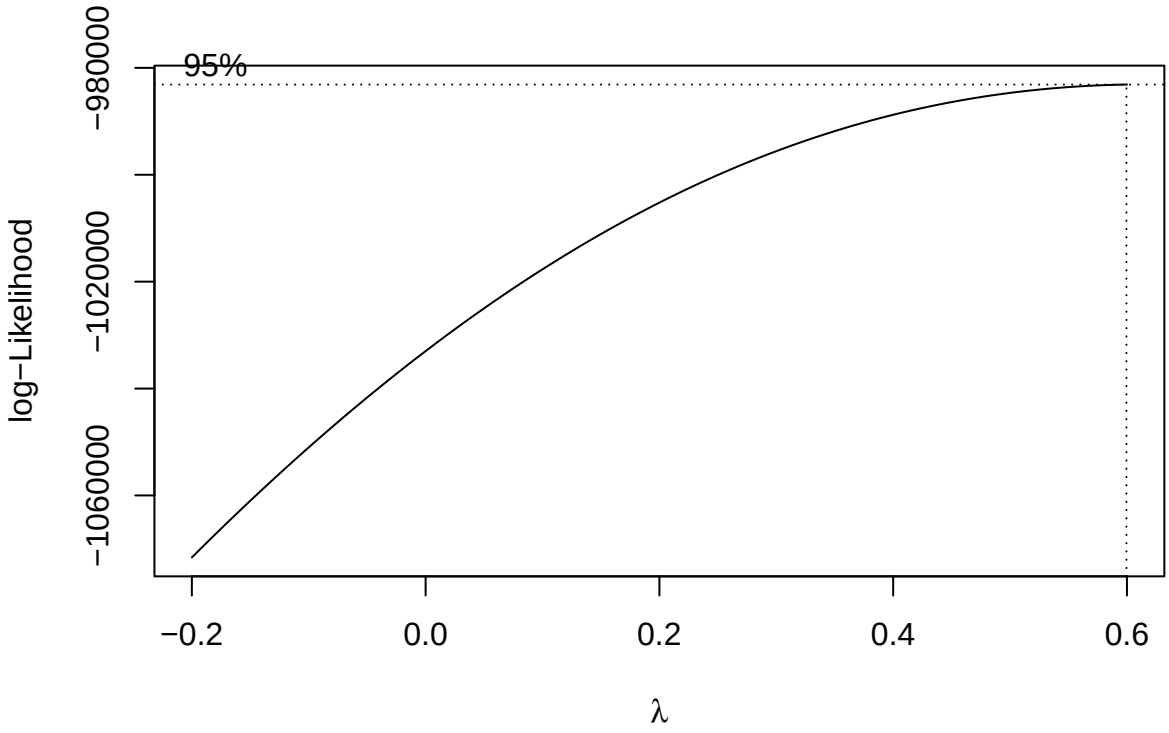


Figure 3: Box Cox Transformation

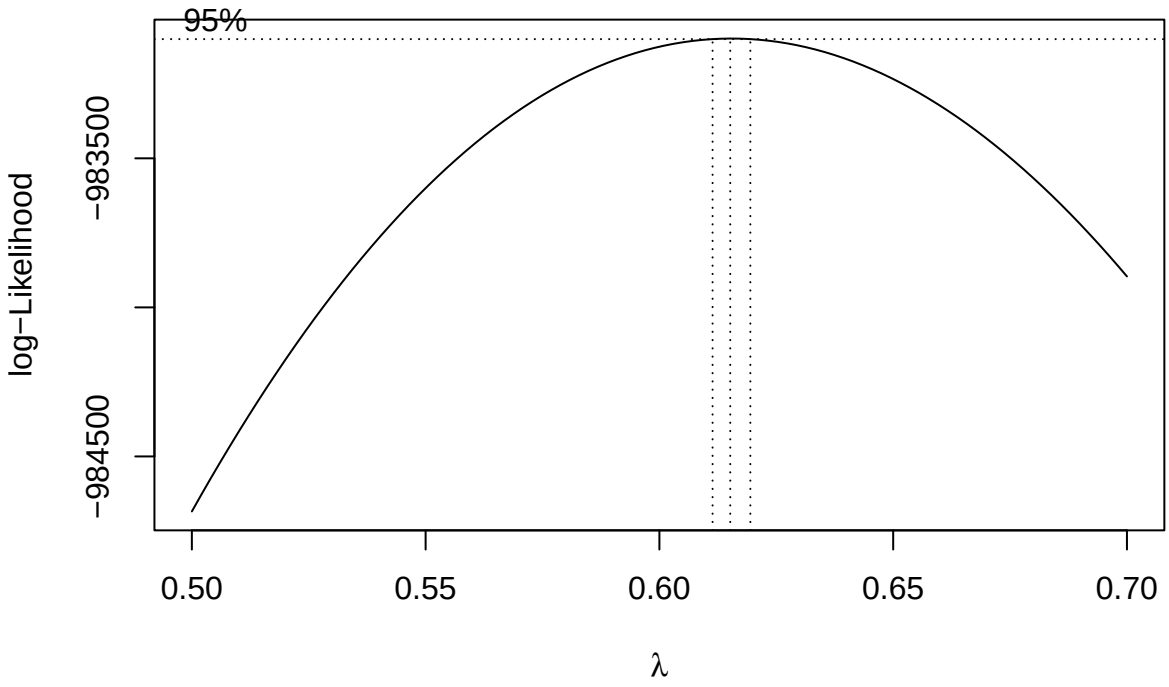


Figure 4: Box Cox Transformation

```r
model_sample1 <- sample(lm_train$residuals, 5000)
shapiro.test(model_sample1)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  model_sample1
## W = 0.99495, p-value = 3.391e-12
```

After applying the optimal transformation, our residuals showed improvement, but our Shapiro-Wilk test still produced a significant p-value (1.286e-12), indicating non-normality in our residuals.

```
## RMSE on Training Set: 28.13561
```

```
## RMSE on Test Set: 28.44961
```

We applied the Box-Cox transformation to address residual non-normality and stabilize variance. However, the transformation increased RMSE to 28 on both training and test sets, likely due to a mismatch between the transformation and the underlying data structure.

```r
robust_model2 <- rlm(popularity ~ danceability + energy + loudness + valence + tempo + acousticness,data
```

As a result, Robust regression was employed because it effectively addresses the influence of outliers and non-normal residuals, leading to improved predictive performance.

```r
train_predictions_robust <- predict(robust_model2, newdata = train_set)
rmse_train_robust <- sqrt(mean((train_set$popularity - train_predictions_robust)^2))
cat("RMSE on Training Set (Robust Model):", rmse_train_robust, "\n")
```

```
## RMSE on Training Set (Robust Model): 16.48724
```

```r
# Calculate RMSE on the test set
test_predictions_robust <- predict(robust_model2, newdata = test_set)
rmse_test_robust <- sqrt(mean((test_set$popularity - test_predictions_robust)^2))
cat("RMSE on Test Set (Robust Model):", rmse_test_robust, "\n")
```

```
## RMSE on Test Set (Robust Model): 16.62138
```

The Box-Cox transformation increased RMSE in the OLS model and was initially excluded. However, its inclusion in the robust regression model significantly improved predictive accuracy (RMSE: 4.5).

```r
model_sample <- sample(robust_model2$residuals, 5000)
shapiro.test(model_sample)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  model_sample
## W = 0.98483, p-value < 2.2e-16
```

```r
library(MASS)

robust_model <- rlm(popularity_boxcox ~ danceability + energy + loudness + valence + tempo +
                    acousticness + instrumentalness + speechiness + liveness + mode,
                    data = train_set)

summary(robust_model)
```

```
##
## Call: rlm(formula = popularity_boxcox ~ danceability + energy + loudness +
##     valence + tempo + acousticness + instrumentalness + speechiness +
##     liveness + mode, data = train_set)
## Residuals:
##       Min        1Q    Median        3Q       Max
## -20.14065  -3.09595  -0.09968   3.05615  20.99146
##
## Coefficients:
##                   Value     Std. Error t value
## (Intercept)        13.1578    0.1167   112.7052
## danceability        7.7824    0.0888    87.6744
## energy              3.4269    0.0944    36.3194
## loudness            0.0675    0.0036    18.8225
## valence            -6.1360    0.0578  -106.0927
## tempo               0.0092    0.0004    23.4037
## acousticness       -7.4695    0.0497  -150.1411
## instrumentalness   -2.9265    0.0429   -68.2648
## speechiness        -9.7482    0.0826  -117.9870
## liveness           -1.9099    0.0678   -28.1601
## mode               -0.0320    0.0253    -1.2617
##
## Residual standard error: 4.563 on 161898 degrees of freedom
```

```r
train_predictions_robust <- predict(robust_model, newdata = train_set)
rmse_train_robust <- sqrt(mean((train_set$popularity_boxcox - train_predictions_robust)^2))
cat("RMSE on Training Set (Robust Model):", rmse_train_robust, "\n")
```

```
## RMSE on Training Set (Robust Model): 4.550497
```

```r
test_predictions_robust <- predict(robust_model, newdata = test_set)
rmse_test_robust <- sqrt(mean((test_set$popularity_boxcox- test_predictions_robust)^2))
cat("RMSE on Test Set (Robust Model):", rmse_test_robust, "\n")
```

```
## RMSE on Test Set (Robust Model): 4.577936
```

A possible explanation for this is that the Box-Cox transformation prepared the data by addressing variance and linearity issues, allowing robust regression to work more effectively by down-weighting outliers without being overwhelmed by residual structure problems. This combination resulted in a model that both captured the central trends in the data and minimized the influence of extreme or noisy points, leading to a significant improvement in RMSE.

This approach resulted in a dramatic improvement in predictive accuracy, with the robust model achieving RMSE values of approximately 4.5 on both the training and test sets. This similarity indicates that the

model generalizes well to new, unseen data, with minimal overfitting, as the model's performance is consistent across both sets.

The robust regression model significantly improved predictive accuracy despite residuals failing normality and homoscedasticity tests. Since the goal of this analysis is prediction rather than inference, these violations do not undermine the model's utility. The robust regression approach emphasizes minimizing prediction error rather than strictly adhering to classical regression assumptions.

Exploration of Heteroscedasticity

While heteroscedasticity is a secondary concern, we still performed the Breusch-Pagan test on the Robust model. Incidentally, we still saw that the homoscedasticity assumption was violated.

Given the presence of heteroscedasticity in the residuals, as identified by diagnostic tests such as the Breusch-Pagan test, a Weighted Least Squares (WLS) model was explored to address this issue, in an attempt to possibly improve the predictive accuracy of the model. The WLS was chosen to reduce heteroscedasticity because it assigns weights inversely proportional to the variance of the residuals, thereby reducing the impact of heteroscedasticity and stabilizing the variance across fitted values.

```
wls_weights <- 1 / abs(model$residuals)  # Inverse of residuals as weights

wls_model <- lm(popularity_boxcox ~ danceability + energy + loudness + valence + tempo +
                acousticness + instrumentalness + speechiness + liveness + mode,
                data = train_set, weights = wls_weights)

summary(wls_model)
```

```
##
## Call:
## lm(formula = popularity_boxcox ~ danceability + energy + loudness +
##     valence + tempo + acousticness + instrumentalness + speechiness +
##     liveness + mode, data = train_set, weights = wls_weights)
##
## Weighted Residuals:
##     Min      1Q  Median      3Q     Max
## -33.765  -1.012  -0.254   0.882  47.499
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)       1.285e+01  3.872e-02  331.76   <2e-16 ***
## danceability      7.879e+00  3.170e-02  248.58   <2e-16 ***
## energy            3.615e+00  3.208e-02  112.70   <2e-16 ***
## loudness          6.364e-02  1.223e-03   52.03   <2e-16 ***
## valence          -6.272e+00  1.923e-02 -326.19   <2e-16 ***
## tempo             7.717e-03  1.505e-04   51.28   <2e-16 ***
## acousticness     -5.805e+00  1.666e-02 -348.41   <2e-16 ***
## instrumentalness -2.373e+00  1.413e-02 -167.94   <2e-16 ***
## speechiness      -1.032e+01  2.549e-02 -404.70   <2e-16 ***
## liveness         -2.353e-01  1.874e-02  -12.56   <2e-16 ***
## mode             -4.330e-01  8.290e-03  -52.23   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 1.156 on 161898 degrees of freedom
## Multiple R-squared:  0.9148, Adjusted R-squared:  0.9148
## F-statistic: 1.738e+05 on 10 and 161898 DF,  p-value: < 2.2e-16
```

```
# Calculate RMSE on training set
train_predictions <- predict(wls_model, newdata = train_set)
rmse_train <- sqrt(mean((train_set$popularity_boxcox - train_predictions)^2))
cat("RMSE on Training Set:", rmse_train, "\n")
```

```
## RMSE on Training Set: 4.598089
```

```
# Calculate RMSE on test set
test_predictions <- predict(wls_model, newdata = test_set)
rmse_test <- sqrt(mean((test_set$popularity_boxcox - test_predictions)^2))
cat("RMSE on Test Set:", rmse_test, "\n")
```

```
## RMSE on Test Set: 4.621838
```

```
ncvTest(wls_model)
```

```
## Non-constant Variance Score Test
## Variance formula: ~ fitted.values
## Chisquare = 21051.24, Df = 1, p = < 2.22e-16
```

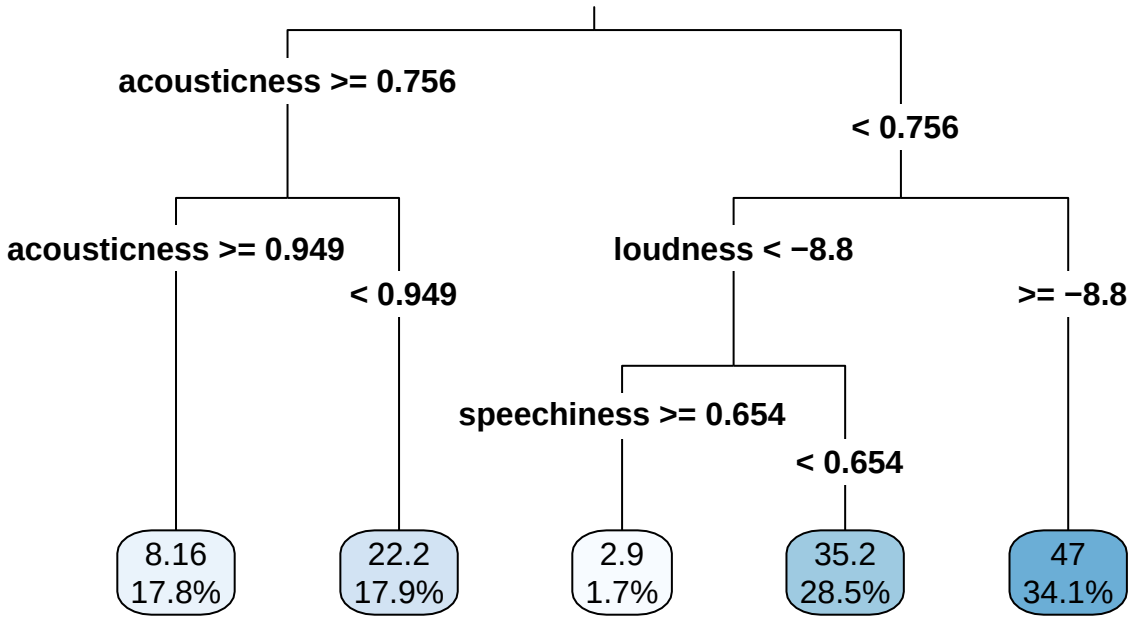
The WLS model achieved RMSE values of approximately 4.59 on the training set and 4.62 on the test set, which were very close to the robust regression model's performance. However, diagnostic tests revealed that the WLS model did not fully resolve the issue of heteroscedasticity, as the Breusch-Pagan test remained significant. This outcome indicated that WLS, while theoretically designed to address heteroscedasticity, was insufficient in this context to completely eliminate the issue.

While Weighted Least Squares (WLS) was explored as an alternative to address heteroscedasticity, it added complexity to the modeling process by requiring an explicit weighting scheme and assumptions about the residual variance structure. In contrast, robust regression automatically down-weights the influence of outliers without requiring these additional steps. Despite these complexities, WLS achieved similar RMSE to robust regression, ultimately making robust regression the simpler and more practical choice for this analysis.

Exploration of Nonlinear Methods

To evaluate potential nonlinear relationships and interactions between musical attributes, a regression tree model was implemented. This model splits the dataset based on thresholds in predictors, providing insights into how specific attributes like acousticness, loudness, and speechiness influence track popularity.

```
library(rpart)
library(rpart.plot)

tree_model <- rpart(popularity ~ danceability + energy + loudness + valence + tempo +
                    acousticness + instrumentalness + speechiness + liveness + mode,
                    data = train_set, method = "anova")

rpart.plot(tree_model, type = 3, digits = 3, fallen.leaves = TRUE)
```




```
tree_predictions <- predict(tree_model, newdata = test_set)

tree_rmse <- sqrt(mean((test_set$popularity - tree_predictions)^2))
cat("RMSE for Regression Tree:", tree_rmse, "\n")
```

```
## RMSE for Regression Tree: 15.96186
```

The regression tree achieved an RMSE of 15.96, which was significantly higher than the RMSE of the robust regression model (4.5). While it provided insights into attribute thresholds, its predictive accuracy (RMSE of 15.96) was significantly lower than that of the robust regression model, reaffirming the effectiveness of linear models for this dataset.

Results

The initial linear regression model achieved RMSE values of 15 on the training set and 16 on the test set. However, diagnostics revealed significant deviations from normality and heteroscedasticity in the residuals, as well as sensitivity to outliers. These issues likely inflated the RMSE and indicated that the initial model did not fully capture the underlying patterns in the data.

To address these limitations, a robust regression model was implemented. The robust regression model achieved an RMSE of 4.5 with the Box-Cox transformed response variable, compared to 15 without the transformation. This demonstrated the complementary effect of the transformation when combined with robust regression. This substantial improvement indicates that the robust regression model effectively mitigated the influence of outliers and deviations from normality, providing more accurate predictions.

An RMSE of 4.5 on a 0–100 scale of popularity means the model's predictions deviate from actual values by an average of 4.5 points, representing just 4.5% of the total range. This level of error indicates that the model provides reasonably accurate predictions, particularly given the variability in the data and the focus solely on musical attributes. The robust model's performance represents a significant improvement compared to the original model's RMSE of 15–16.

The robust regression model demonstrated that popularity could be predicted with reasonable accuracy based on musical attributes alone. The findings suggest that energetic, rhythmic, and emotionally intense features play a pivotal role in determining a track's success. These insights align with broader trends in music consumption, where high-energy tracks dominate popular charts.

Limitations of Study

1. Limited Scope of Variables: The model only considers specific musical attributes (e.g., danceability, energy, loudness) as predictors of popularity. There are other important contextual factors, such as artist popularity, genre, promotion efforts, and listener demographics, are not included in the model.
2. Simplistic Modeling Approach: This study largely focused on linear relationships between predictors and popularity, which may oversimplify interactions or nonlinear patterns in the data. While robust regression provided strong predictive accuracy, future research could explore advanced machine learning models, such as random forests or neural networks, to capture more complex patterns.
3. Data Source Bias: The dataset was collected from Spotify and may reflect biases in Spotify's algorithms for determining popularity scores. A broader analysis using data from other platforms, such as Apple Music, could could yield more generalizable results.
4. Normality and Homoscedasticity: This study prioritized predictive accuracy over strict adherence to traditional regression assumptions like normality of residuals and homoscedasticity. While robust regression mitigates the influence of outliers and non-normality, some violations of these assumptions remain. However, they are not critical in this predictive context.

Conclusion

This study demonstrated that musical attributes such as danceability, energy, and loudness positively influence track popularity, while valence, acousticness, and speechiness negatively impact it. Using a robust regression model, the analysis achieved an RMSE of 4.5 on a 0–100 scale, reflecting strong predictive accuracy and highlighting the importance of energetic and rhythmic features in driving popularity.

The robust regression model's ability to handle outliers and deviations from normality made it the most suitable choice for this analysis, outperforming both the initial linear regression model and the Weighted Least Squares (WLS) approach. While the study focused exclusively on musical attributes, its findings align with broader trends in music consumption and provide a foundation for future research.

Violations of normality and heteroscedasticity persisted in the final model but were deemed irrelevant for predictive purposes. These findings provide a foundation for understanding the relationship between musical attributes and popularity, with practical implications for artists, producers, and marketers.

While a regression tree model was explored to capture nonlinear relationships, its RMSE of 15.96 highlighted its limited predictive accuracy compared to robust regression. This finding supports the conclusion that simpler linear models are more effective for this dataset while providing opportunities to explore nonlinear methods further in future work.

Future work could expand this study by incorporating contextual factors, exploring other nonlinear methods, and addressing dataset bias through multi-platform analyses. These enhancements would further clarify the drivers of track popularity and improve model generalizability.