Predicting the popularity of a song based on Spotify sound/musical metrics

Akshat Thakur, Navid Samiei, Asen Lee, Yichen Xin

Introduction

We're music lovers in the 21st century, so vinyls, CDs and iPods are long gone, and only streaming platforms like Spotify are popular now. When streaming music on Spotify, you're probably not thinking about how it classifies each song based on a variety of criteria. But it does! Spotify evaluates each music based on elements like danceability, energy, valence, and more. Curious about this exact thing, we found a dataset on Kaggle (link: https://www.kaggle.com/sashankpillai/spotify-top-200-charts-20202021) that was web scraped from the Spotify Web API, and we began to learn what characteristics are associated with driving music that became widely popular by looking into the measurements Spotify determines. We think this is an extremely worthwhile project because it's immensely interesting to look into what makes a popular song.

All our variables are just metrics from Spotify, and our explanations of the variables are from the Spotify Web API documentation.

We want to find out how and the extent to which various audio features from Spotify metrics affect how popular a song gets.

The target/response variable: Popularity: The popularity of the track. The value will be between 0 and 100, with 100 being the most popular.

Predictor variables:
Danceability: Danceability describes how suitable a track is for dancing based on a combination of musical elements including tempo, rhythm, etc. A value of 0.0 is least danceable and 1.0 is most danceable.
Acousticness: A measure from 0.0 to 1.0 of whether the track is acoustic. Duration: The duration of the song in milliseconds (ms).
Energy: Energy is a measure from 0.0 to 1.0 and represents a perceptual measure of intensity and activity.
Liveness: Detects the presence of an audience in the recording.
Loudness: The overall loudness of a track in decibels (dB). Loudness values are averaged across the entire track.
Speechiness: Speechiness detects the presence of spoken words in a track.
Tempo: The overall estimated tempo of a track in beats per minute (BPM).
Valence: A measure from 0.0 to 1.0 describing the musical positiveness conveyed by a track. Tracks with high valence sound more positive, while tracks with low valence sound more negative.
Chord:A chord is any harmonic set of pitches/frequencies consisting of multiple notes that are heard as if sounding simultaneously.

We excluded Index, Song Name, Song ID, Release Date, etc. due to irrelevance to our aim, excluded Highest Charting Position, Number of Times Charted, Week of Highest Charting, Streams, Weeks Charted due to inaccessibility or blank values for the new songs.

Exploratory Data Analysis

Pairwise Correlation Table and Scatter Plots

Since Chord is a categorical variable, we conducting one analysis excluding chord. Then another, including it (omitted here, but can be found in the Appendix).

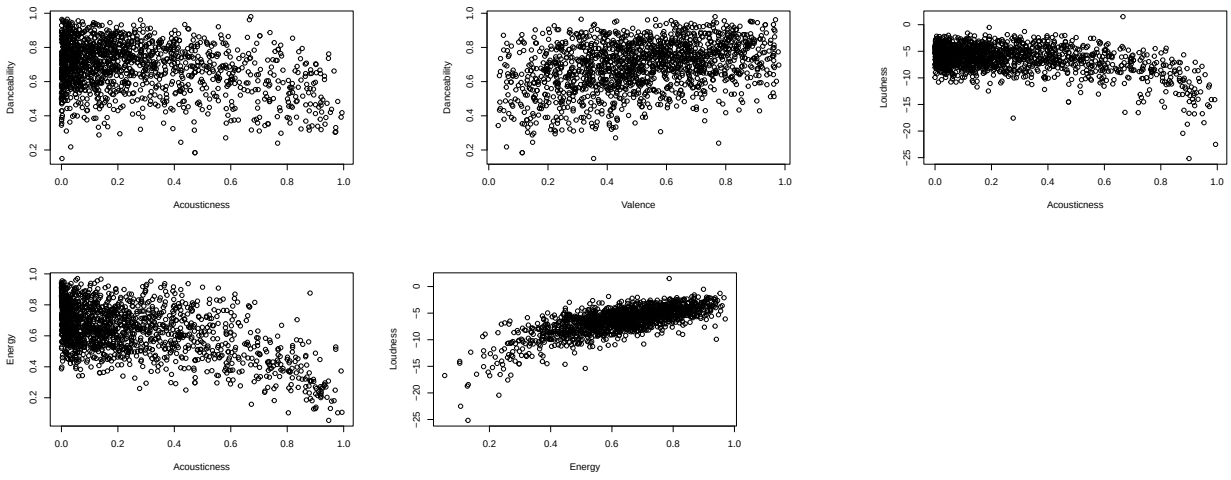
Table 1: Pairwise Correlation Table

	Pop.	Dan.	Ene.	Lou.	Spe.	Aco.	Liv.	Tem.	Val.	Dur.
Popularity	1.00	0.03	0.09	0.16	-0.03	-0.09	-0.03	-0.02	0.00	0.08
Danceability	0.03	1.00	0.14	0.23	0.24	-0.32	-0.11	-0.04	0.36	-0.10
Energy	0.09	0.14	1.00	0.73	0.02	-0.54	0.12	0.11	0.36	0.06
Loudness	0.16	0.23	0.73	1.00	-0.02	-0.48	0.04	0.10	0.30	0.08
Speechiness	-0.03	0.24	0.02	-0.02	1.00	-0.13	0.07	0.11	0.04	-0.09
Acousticness	0.09	-0.32	-0.54	-0.48	-0.13	1.00	-0.01	-0.06	-0.10	-0.05
Liveness	-0.03	-0.11	0.12	0.04	0.07	-0.01	1.00	-0.02	0.01	0.02
Tempo	-0.02	-0.04	0.11	0.10	0.11	-0.06	-0.02	1.00	0.06	0.00
Valence	0.00	0.36	0.36	0.30	0.04	-0.10	0.01	0.06	1.00	-0.12
Duration..ms	0.08	-0.10	0.06	0.08	-0.09	-0.05	0.02	0.00	-0.12	1.00

Most of the data distributions are rather random with no specific trend. This indicates that Simple Linear Regression may not be a good candidate model.

Low correlation: Acousticness and danceability (-0.32). Danceability and valence (0.36). Acousticness and loudness (-0.48).
Moderate correlation: Acousticness and energy (-0.54).
High correlation: Energy and loudness (0.73).

Plots of most correlated variables



These correlations make sense. Songs that have one of the features among danceability, energy, loudness and valence will tend to exhibit the other features from anecdotal experience. By similar token, if the song is more acoustic, it will likely be less energetic and loud explaining the negative correlation.

Results and Analysis

Linear Regression and Polynomial Regression

Linear Regression

First, we do linear regression on the training data using all predictors to get a full linear model and compute the LOO-CV.

The leave one out cross validation score for the full model is roughly 15.77. We have nothing to compare this to yet so we can't make a judgment on the model.

When attempting to do an exhaustive selection on all model subsets, we notice there is a linear dependency found when we include chord in our model. This is likely because a particular pair of chords are linearly dependent with each other. Moreover, since chord did not seem to have any relationship with popularity (nor any of the other variables), we are comfortable in excluding it for this portion of our analysis.

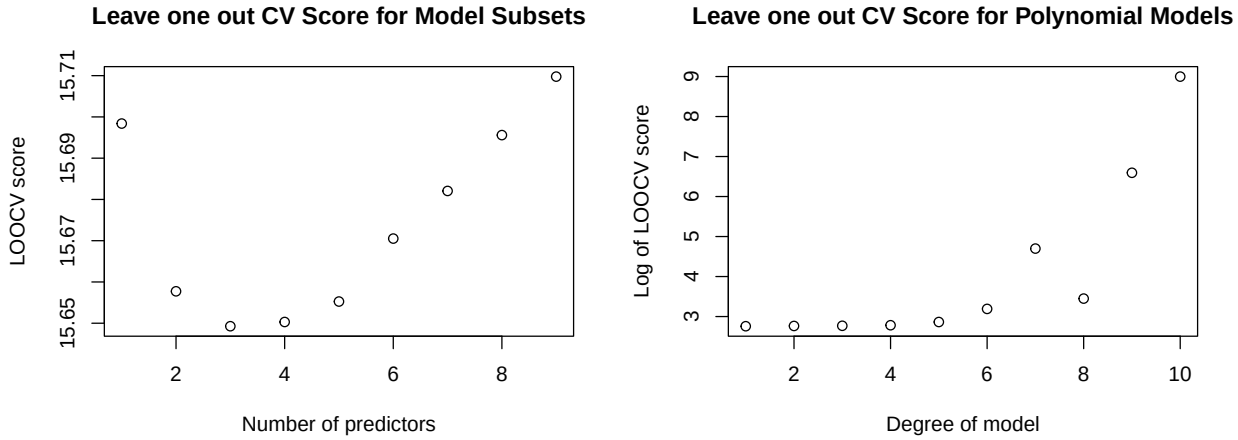
From the left plot below, we notice that the full model has the largest LOOCV score. The scores are in a parabolic shape, where they decrease monotonically until the minimum at 3 predictors After that, the LOOCV scores increase monotonically. The model with 3 predictors has Loudness, Tempo and Duration. This reduced model performs slightly better than the full model, with a LOOCV score of 15.65.

Polynomial Regression

Here, we attempt to see if there may be any polynomial relationship between the variables and popularity

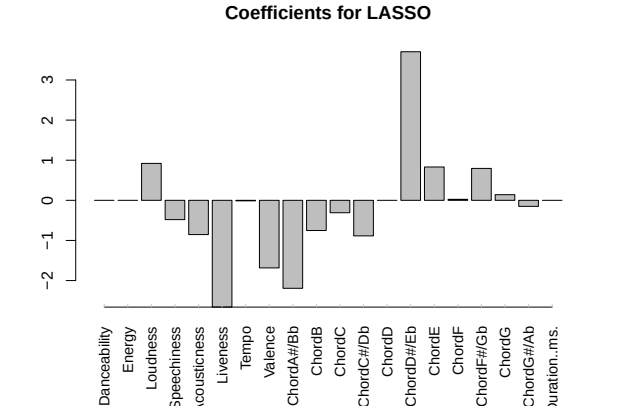
For a particular degree, i, we do linear regression on the full model including each numerical variable to a power between 1 to i. The below plot on the right shows the log of the LOOCV scores (since they grow drastically) for each polynomial model for each degree i. As we can see, increasing the degree of the full model makes the prediction ability worse (except for degree 8). The best model is in fact the original one with degree 1. This is likely because there is no polynomial relationship between the variables and popularity, and the additional predictors only serve in overfitting the model.

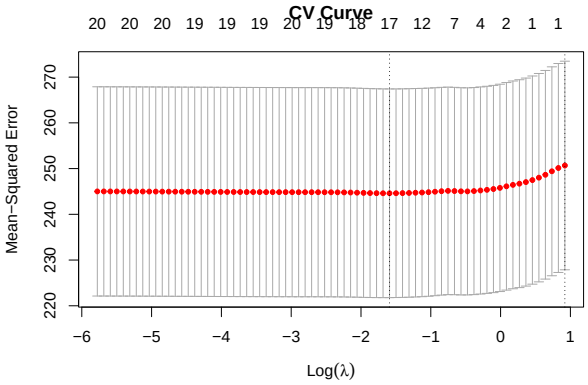
Plots

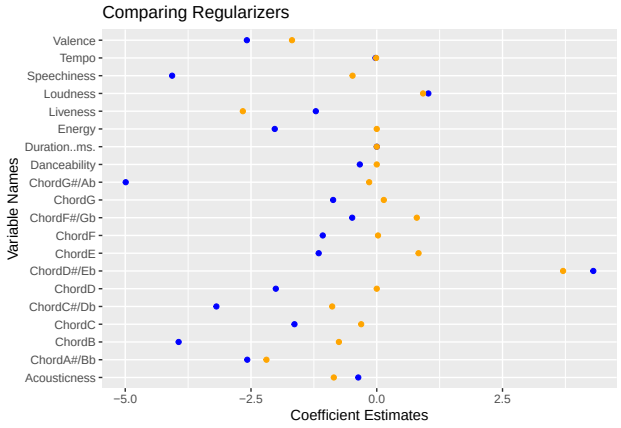


LASSO Regression

Second, we build another model using LASSO in favor of regularization and variable selection.





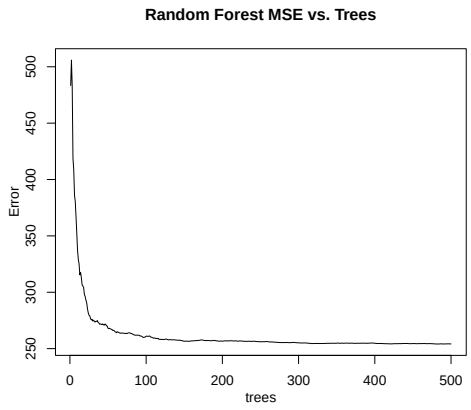


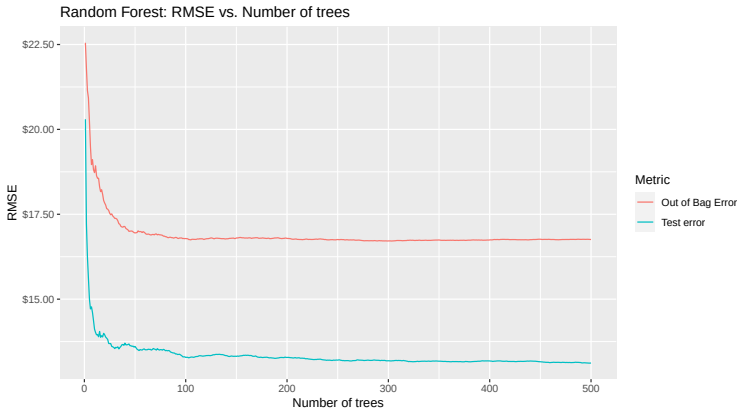
We obtain an RMSE of 15.8894 using the minimum lambda that gives the lowest CV error. We can also see from the CV curve that the minimum lambda is 0.2319 and the largest value of lambda within 1 standard error of lambda_min is 1.9703. Furthermore, by using LASSO, the coefficients for Danceability, Energy, Speechiness, ChordD, ChordF#/Gb, ChordG, and ChordG#/Ab have been removed by variable selection. It is also observed that the coefficient for Duration..ms. is very small. From the plot of regularizers above, it can be seen that the coefficient estimates for the OLS estimates vary slightly more than the LASSO estimates. The coefficient estimates for LASSO calculated from lambda_min are mostly smaller in magnitude and LASSO has coefficient estimates that are 0, which is expected.

Random Forest Regression

Basic Implementation and Predictive Accuracy

First, we start the random forest regression model with a basic approach. The model is build by library randomForest. And we use full model for the formula.





The figure on the left showed the MSE vs. Model's tree number. The optimal base model has 484 number of trees and gives us 15.94 RMSE. We also used a validation set (by splitting the training set) to evaluate the prediction capability. As shown by the figure on the right, the out-of-bag error is around 17 and the test error is around 13 which is close to the basic model's RMSE. Random Forest Regression model showed a promising prediction ability.

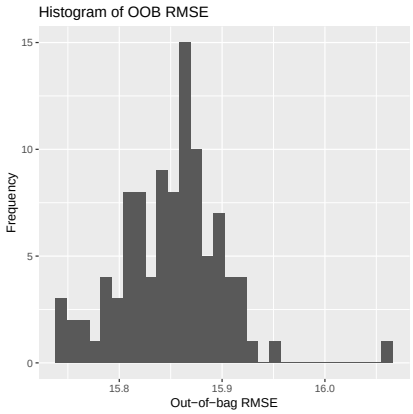
Tuning

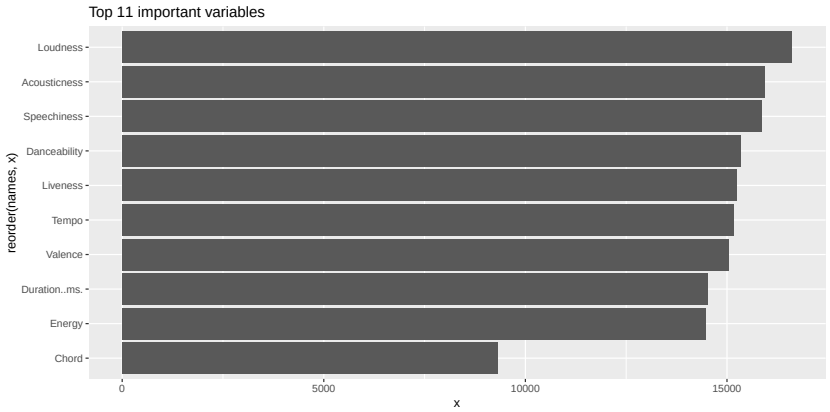
Next step, we fine-tuned the hyperparameters of the model, namely, "the number of trees: num.trees", "the number of variables to randomly sample as candidates at each split: mtry", "min number of samples within the terminal nodes: node_size", and "the fraction of samples to train on: sample_size". We used grid search method to find the optimal combinations.

```
##   num.trees mtry node_size sample_size OOB_RMSE
## 1       500    1         5       0.632 15.75547
## 2       150    1         5       0.700 15.76863
## 3       300    1         5       0.700 15.77095
## 4       400    1         5       0.632 15.77123
## 5       500    1         3       0.632 15.77220
```

The listed combinations are the top 5 most optimal ones. To improve the prediction capability, we should choose the simpler model with decent RMSE. Thus, the second one with much smaller tree size is chose.

We repeated this optimal model 100 times to get a better expectation of error rate.





The error rate is around 15.9 as shown on the left figure.

Based on this model, we plotted the importance of each variables on the right. Loudness is the most important one. And the Chord is the least one. This result agrees with a trend called "The Loudness War" [https://www.npr.org/2009/12/31/122114058/the-loudness-wars-why-music-sounds-worse]. The other variables are about the same and close to Loudness.

Finally, this model got a training RMSE of 9.98 and test RMSE of 15.71. As the scale of Popularity is 100, the test error of 15 suggests an adequate prediction capability of the Random Forest Regression model.

Ensemble method

Our last attempt is an ensemble method where we train our model using a combination of our above models. We will give each model a weight. Then the ensemble method's prediction for a particular training response value is the weighted mean of each model's prediction.

The first weighting scheme is simply giving each model an equal weight. We get a training RMSE of 13.71. We next attempt three very basic weighting schemes. In each scheme, we give one model a weight of 1/2 for the prediction and the other two models a weight of 1/4. This is so we can observe whether we can perform better if certain models contribute more to the predictions. We find that when giving more weight to Random Forest Regression, we achieve the lowest RMSE at 12.76.

Table 2: Ensemble methods' RMSE

Model	RMSE
Equal	13.71
More for LM	14.17
More for Lasso	14.23
More for RFR	12.76

Training vs Test in all models

Finally, we will compare our various models on their performance on predicting test data. As shown below, the reduced linear model performs best and the random forest regression does worst. Nonetheless, all models perform similarly.

Table 3: Training and test RMSE for all methods

Model	RMSE Training	RMSE Test
Reduced LM	15.59	15.52
LASSO	15.83	15.8
Random Forest	9.99	15.71
Equal W Ensemble	12.76	15.57
RFR> Ensemble	14.23	15.58

Conclusion

Were we able to successfully answer our research question of predicting the popularity of a song based on the metrics Spotify releases publicly? We would like to say yes, to a certain extent, but we must concede and acknowledge that it's impossible to ALWAYS (or even most of the time) predict whether a song will be popular or not. A huge part of that is because some very significant elements appear to be at play in deciding popularity that aren't necessarily accounted for in this dataset.

Some factors we came up with that may influence popularity include asking whether there is current awareness of a certain artist, if the artist in question has had a great career so far, previous smash hits, and generally what their track record has been, what genres the artist dabbles in, and what genres are currently popular, and whether the artist worked with other well-known or celebrated musical artists.

We believe that combining the information we got from the dataset with the added answers to some of the aforementioned questions would result in a far more accurate prediction of song popularity.

From a technical perspective, another limitation of our project is that our Random Forest method leads to overfitting. As shown in the Table 2, The training error is much lower than the test error. The Reduced Linear Method and LASSO are okay from an over/underfitting point of view.

Which predictors seem to be more important and why?

The three models we chose all had different significant variables. Specifically, the Reduced Linear Model had Loudness, Tempo, and Duration; LASSO had Loudness, Acousticness, Liveness; Random Forest had Loudness, Accousticness, Speechiness.

The predictors all three have in common is Loudness, which makes sense considering that hip-hop is the most popular genre at the moment, and it revolves around this.

Can we generalize our results to other data?

Not really, unfortunately. This dataset contains instances of popular songs on Spotify from 2020 to 2021, so using our results wouldn't work for data containing songs from any other time in history (and probably, the future). Moreover, our dataset only had 1517 songs. Spotify alone has over 70 million songs as of today. Our results would definitely not be a good fit for ALL songs available on Spotify.

Moreover, everything we've done is based on Spotify data, and they regularly tune up their algorithms so their metrics might change over time. This would make generalizing close to impossible.

Other possible research questions that may have arisen

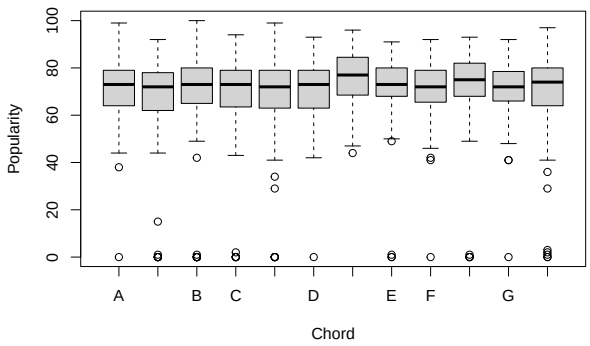
Considering the current status of modern music and how it is widely consumed (sorry dad, your cassettes don't quite count here), it might be sensible to investigate more variables. To name a few:
- An artist's following on social media
- Whether an artist is signed to a major record label. If yes, which? The big four record labels drastically improve the chances of having popularity, since they have deep pockets for advertising and amazing production capacity.
- A "nostalgia" rating. A lot of dads (and granddads) have transitioned to using streaming platforms.

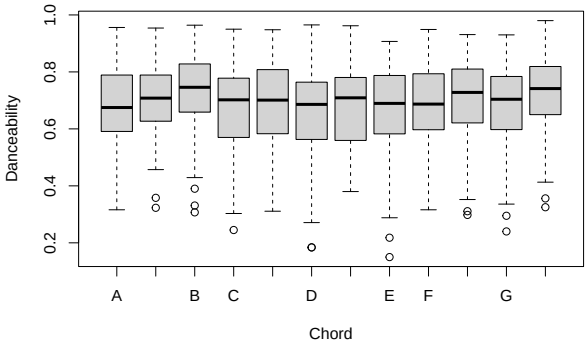
We could also break down song popularity into subsets based on demographics such as language, region, whose account is streaming, what sort of devices are being used to stream music, etc. It would be so, so cool to observe if our mode accuracy improves with this information added to our dataframe, and could let us find a reliable way of predicting song popularity. This would be invaluable to artists and record labels, as they could just see what goes into making a song popular and stick to that formula to keep generating hits. We could probably feed the model into an AI that would make music based on this information. So much scope!

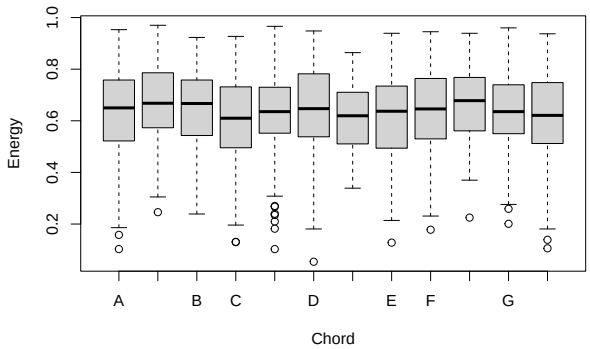
Appendix

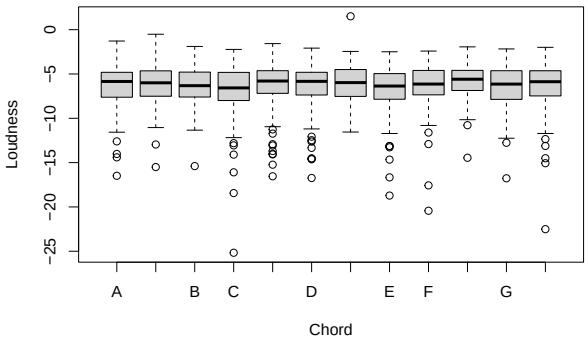
As mentioned in the EDA section of the report, we examined how each variable is distributed by its chord.

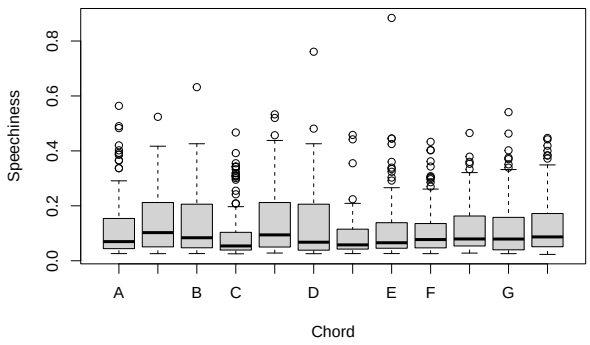
Boxplots of metrics by chord

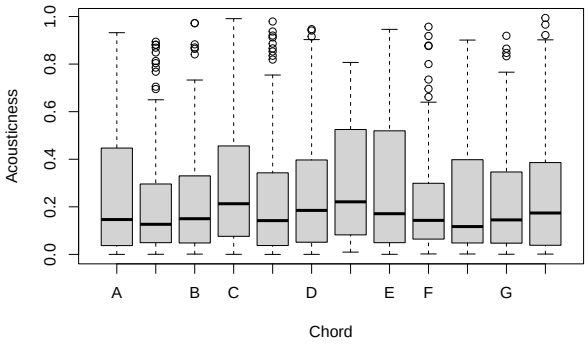


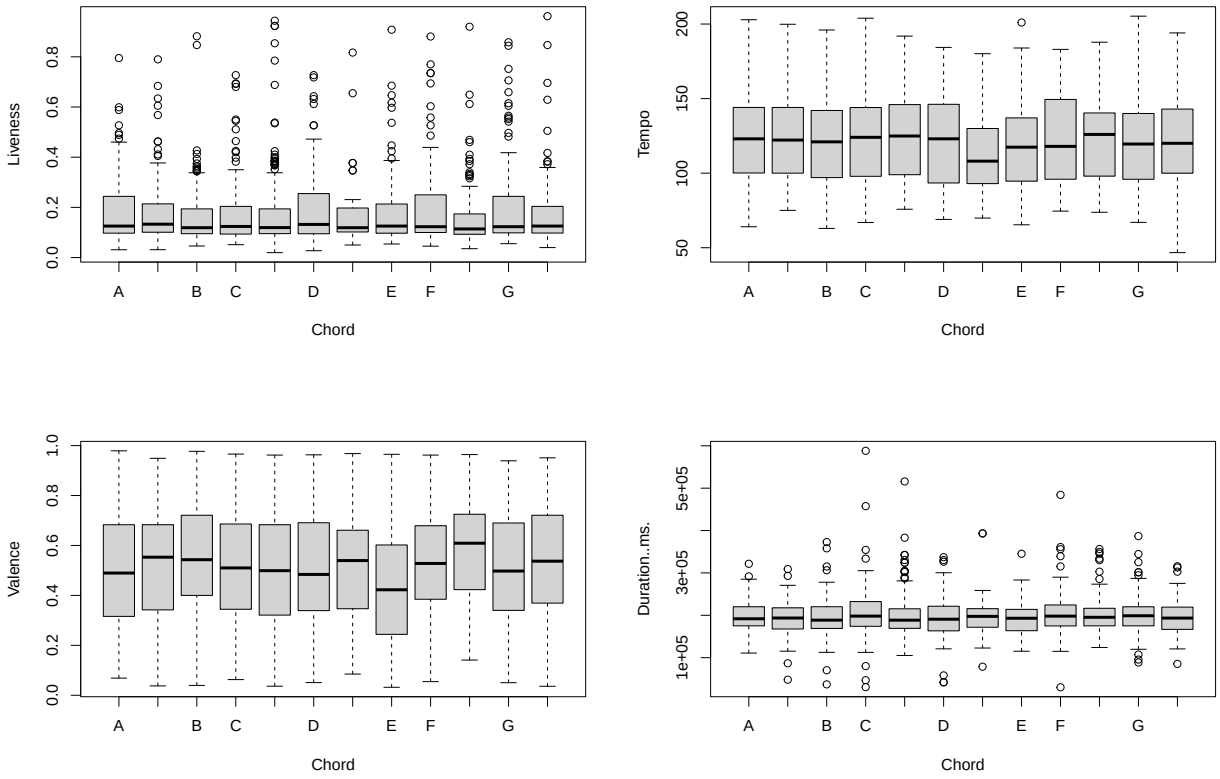












Comments on Metrics by Chord: There is no apparent trend or notable observation for the distributions of the variables separated by its chord. For the purposes of our research question, we could point out that songs in D#/Eb tend to have slightly more popularity.