Frank Pasqualini

Carl Bodenschatz

STAT 1223

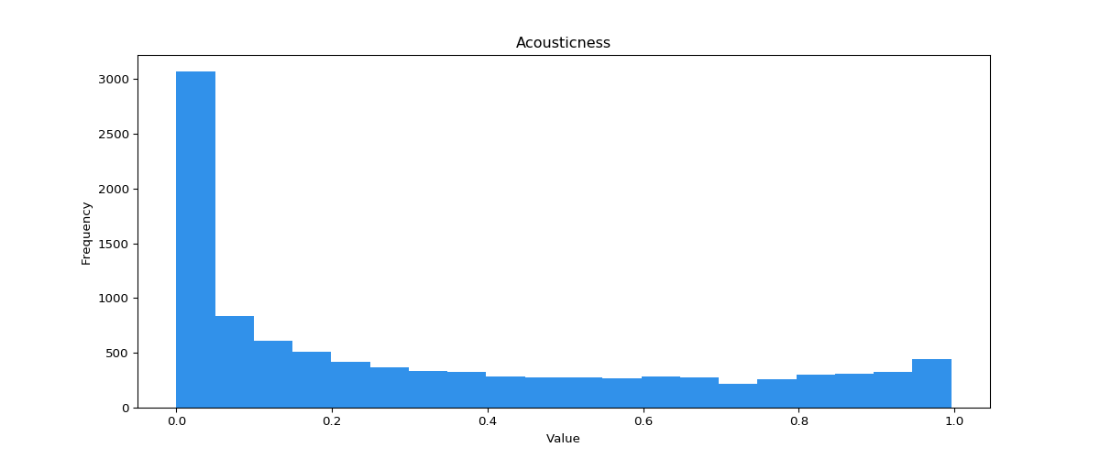
20 Mar 2019

Spotify Music Library Regression Analysis – Progress Report

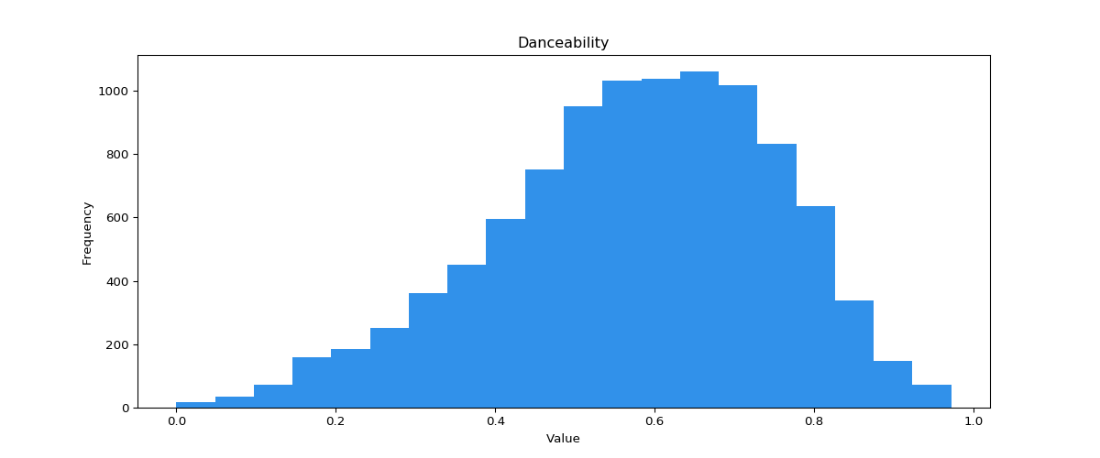
This study is being conducted to build a multivariate regression model relating multiple attributes of a musical track on Spotify to a specific user’s musical preferences. The response variable is a play count, the amount of times that the user has listened to that song. We believe that a play count is a good estimate of how much a user likes a track, because if they like the track more they will listen to it more. It is not a perfect estimate, however, because it is possible for a user to have discovered a track recently that is one of their favorites, but due to not having known about it for a long period of time the play count would be low. The reverse is also true, where a user could have a song that they think is okay but not one of their favorites, but since it has been in their library for years, they have listened to it more times. It is very difficult to quantify how much a user likes a track, and we believe that play count is the best estimate we can currently research. We think these results will be particularly interesting because if this study is successful, it could be turned into a generalized tool that a user could insert the data from their own music library into. The resulting regression model could be then used on a per-individual basis to find songs that are likely to be enjoyed by the listener, by giving recommendations for songs that would maximize the response variable based on the predictors. This will be a case study based off of the Spotify and Last.fm accounts of Frank Pasqualini.

For this study, we are performing the regression on the music library of Frank Pasqualini, using data from his Spotify and Last.fm accounts. At the time of gathering the data to be used in the study, Frank had 5,774 tracks saved to his Spotify library and 37,933 scrobbles saved to his Last.fm account from 1,342 artists. A scrobble is a recorded instance of a track being played. Frank has been tracking his scrobbles using Last.fm since December of 2017, and those scrobbles were tracked directly by linking his Last.fm account to his Spotify account, so the data should be mostly valid even though the predictors will be collected from a different source than the response. We are looking into using 15 potential predictor variables, which are as follows: duration\_ms, explicit, popularity, key, mode, time\_signature, acousticness, danceability, energy, instrumentalness, liveness, loudness, speechiness, valence, and tempo. These were retrieved for each track using Spotify’s Web API.

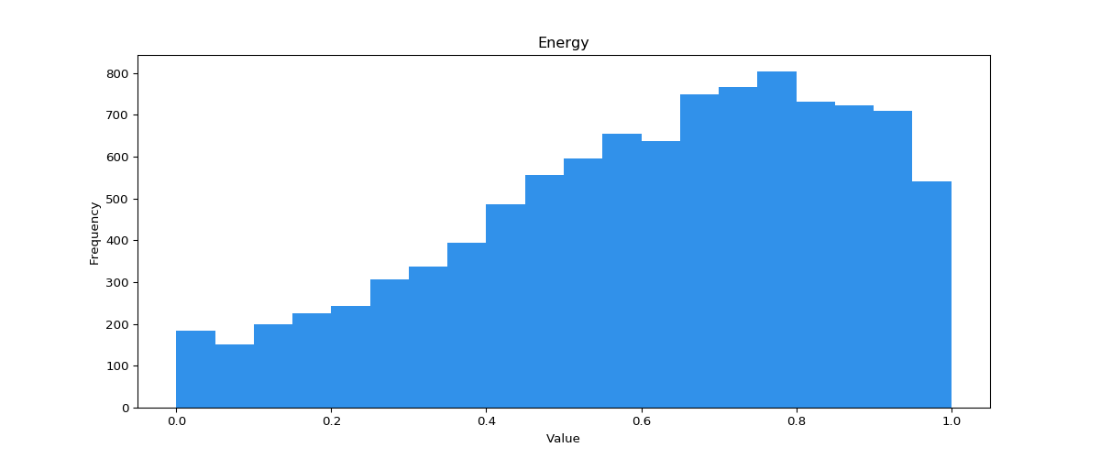
Almost all the variables we used in our model were retrieved from the “Get Audio Features for a Track” Spotify API call. The documentation for this API call gives in depth descriptions about what each of these values represent, as well as providing the distributions that we have included below, but we will provide a general description of each of them. Some of the more basic variables were taken directly from the track object returned from the API call. Duration\_ms is a measurement of the length of the track in milliseconds. Explicit is a categorical variable that tells whether the track has been marked to contain vulgar language. Popularity is an integer value between 0 and 100 that is a calculated by one of Spotify’s algorithms based on total plays and how recent those plays are. Key is a categorical variable representing the musical key the track is estimated to be in for the majority of the track, with 0 being C, 1 being C#/D♭, all the way up to 11 representing B/C♭. Mode is another categorical variable with 0 representing if the track is in a minor key for the majority of the track and 1 if the track is in a major key for the majority of the track. Time\_signature is an estimation of how many beats are in each bar for the majority of the track. Acousticness is a confidence measure of whether the track is acoustic. It has the distribution featured in the following figure:



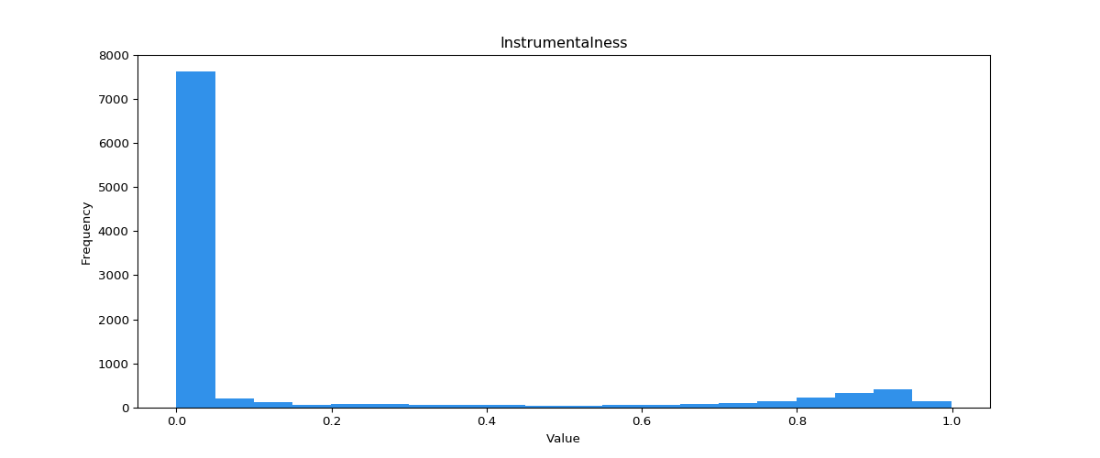
Danceability is an estimate of how well suited a track is for dancing to based on many attributes such as tempo, stability, and regularity, and has the distribution shown in following figure:



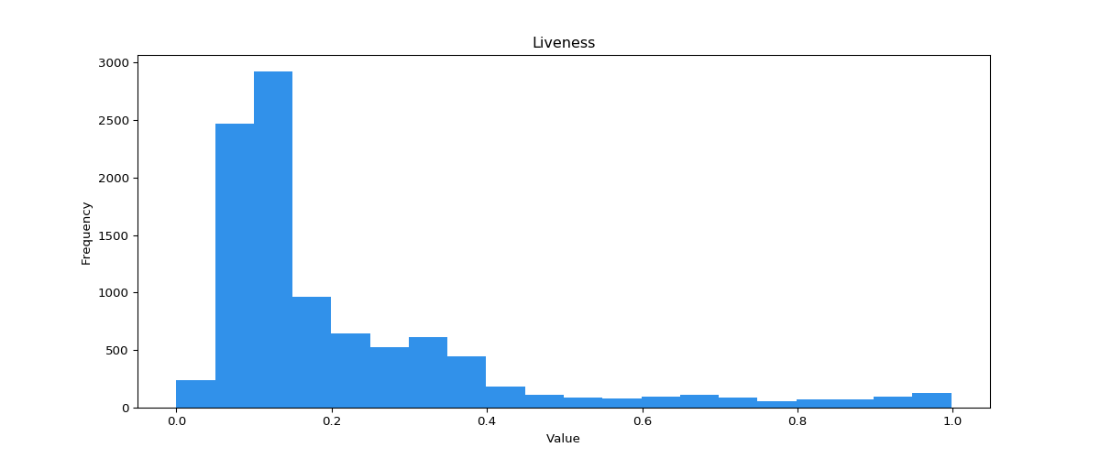
Since danceability is in part based on tempo there will likely be high covariance between these two variables. Energy is an estimate of the intensity of the track based on attributes such as entropy, timbre, and loudness, and has the distribution featured in the following figure:



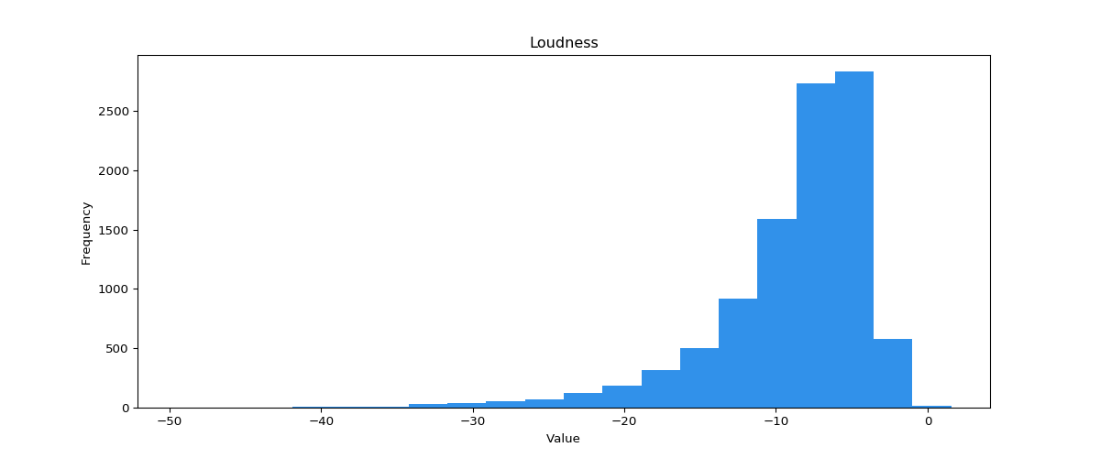
Since energy is in part based on loudness there will likely be high covariance between these two variables. Instrumentalness is a confidence measure of whether the track contains no vocals and has a very skewed distribution represented by the following figure:



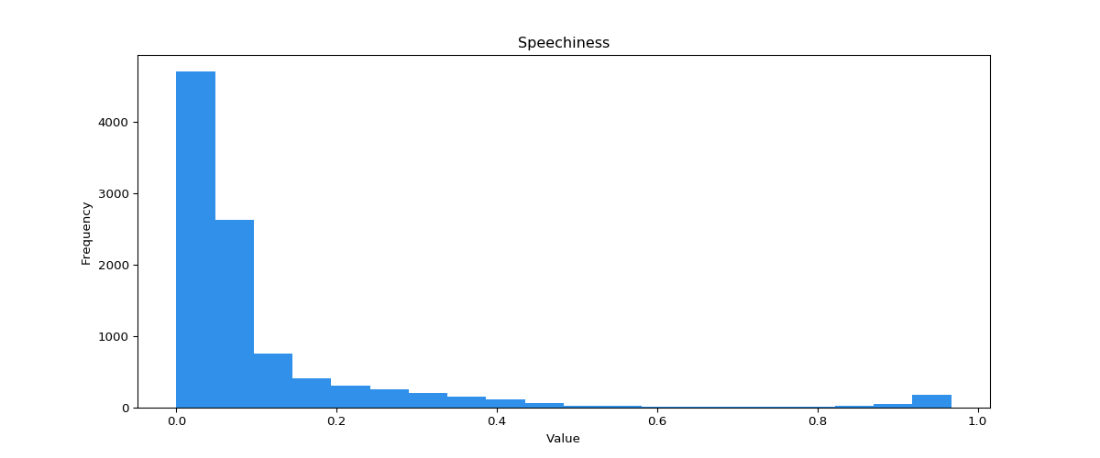
Liveness is a confidence measure of whether there is a live audience in the recording and has the distribution in the following figure:



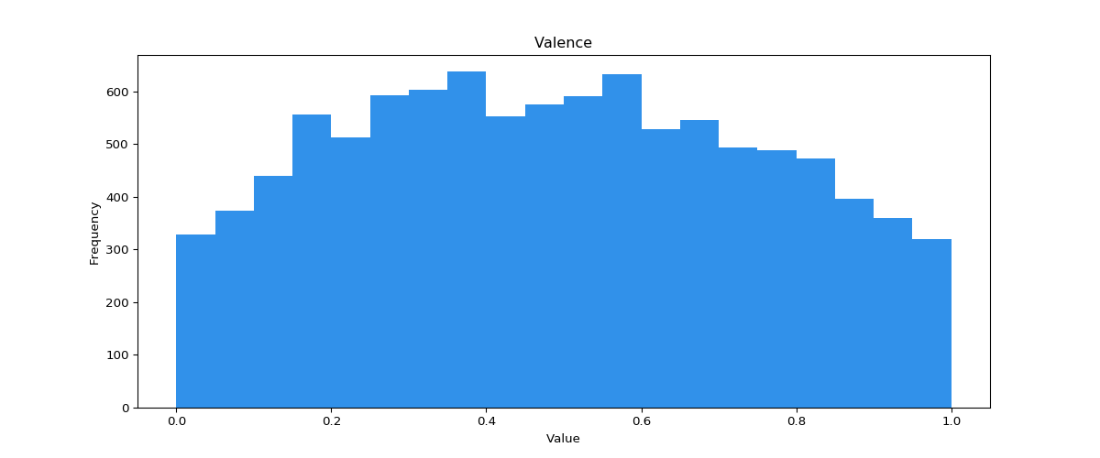
Loudness is the measure of the average decibel value of the track and has the distribution in the following figure:



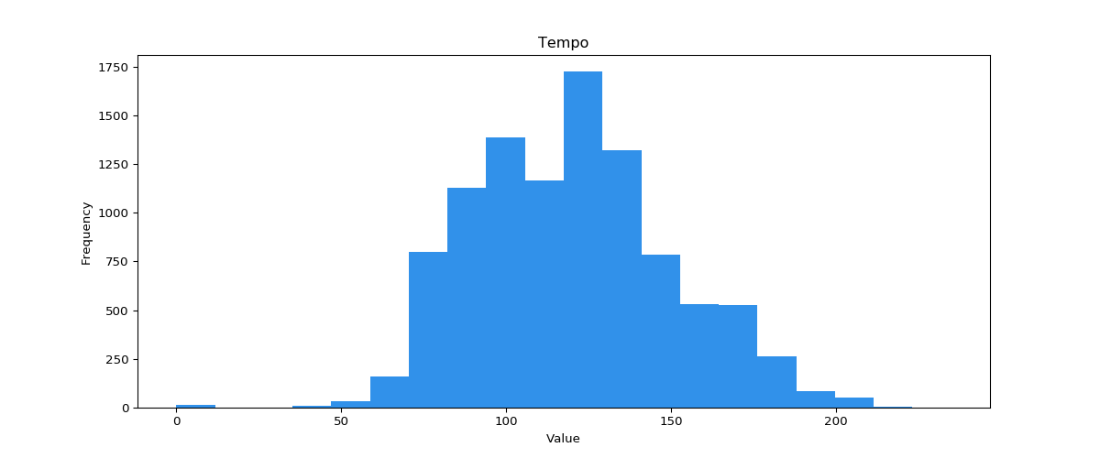
Speechiness represents the exclusivity of speech in the track, and high values are almost entirely speech while low values contain little speech but possibly sung words. It has the distribution featured in the following figure:



Valence is a measure of the “happiness” of a song and has the distribution in the following figure:



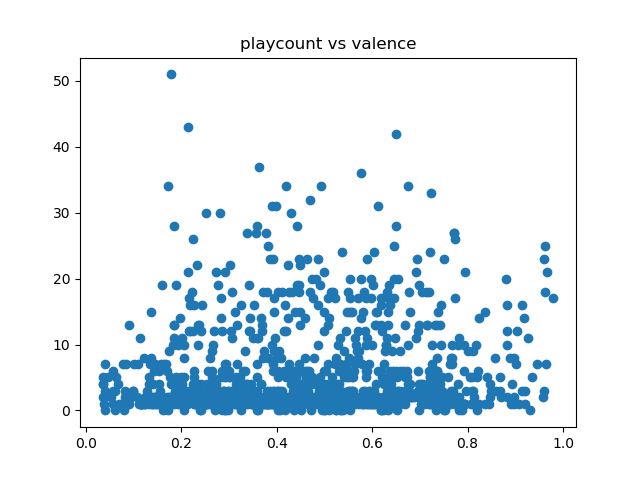
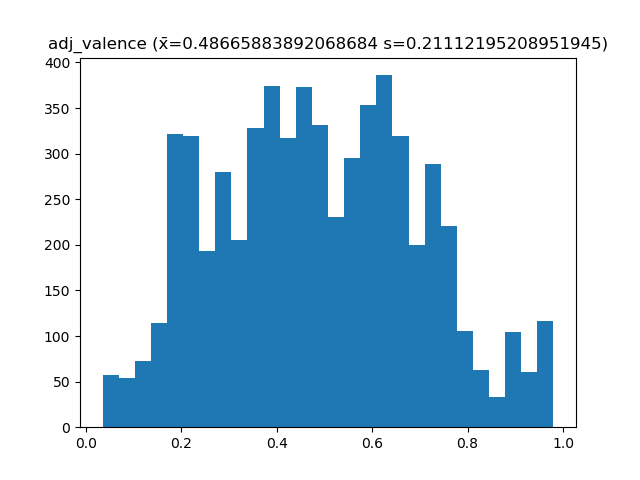
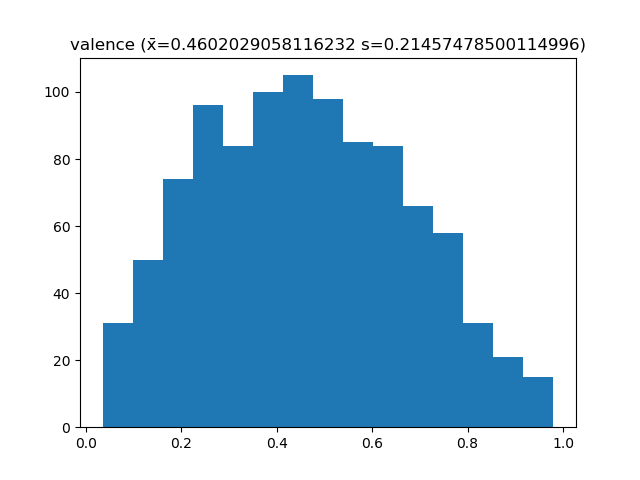
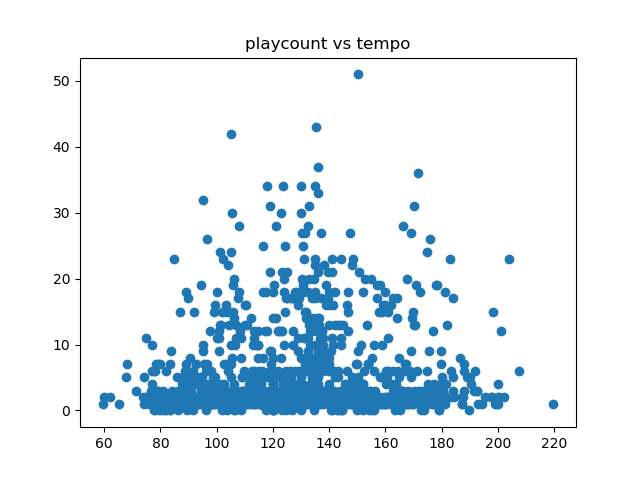
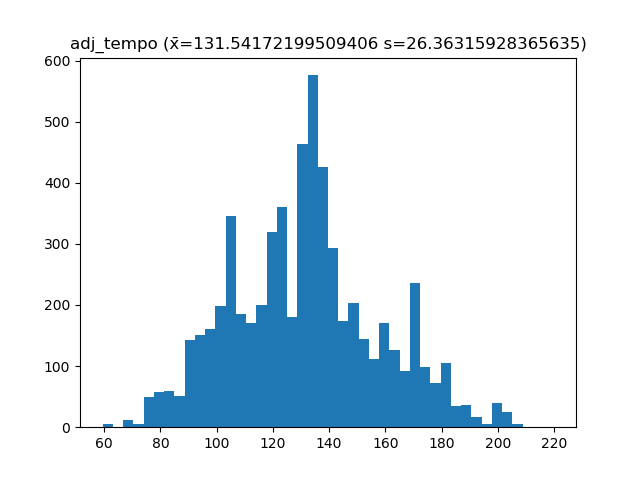
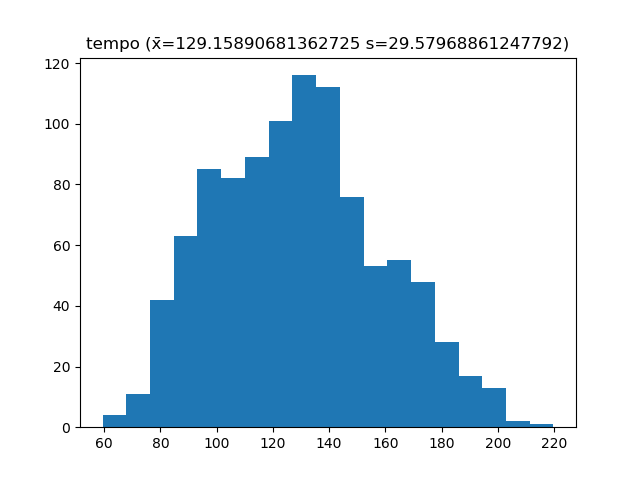
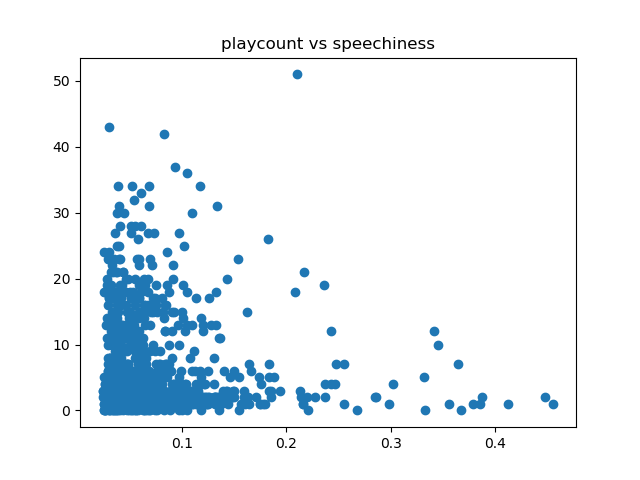
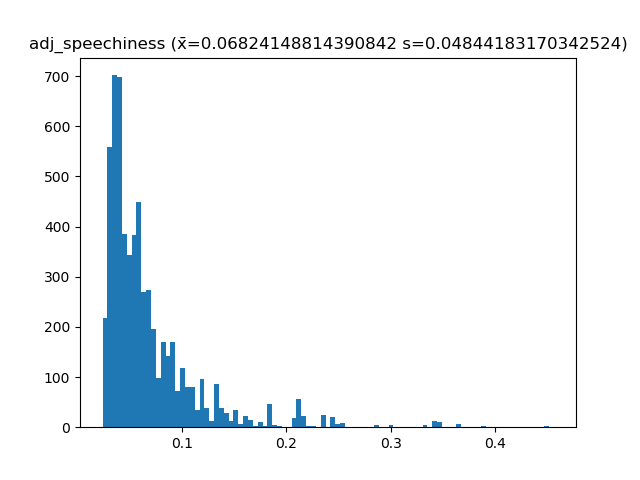
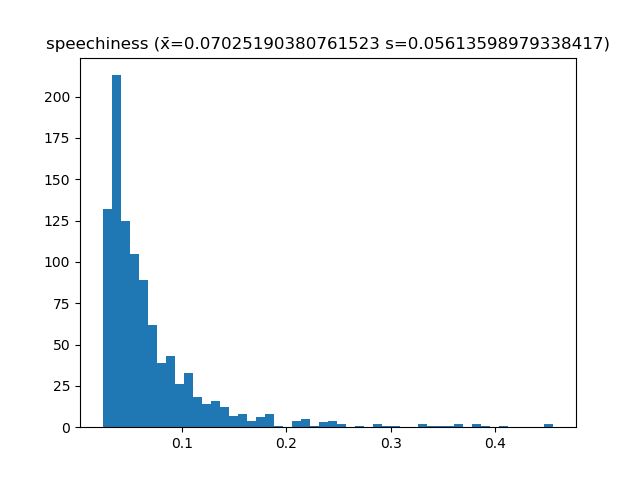
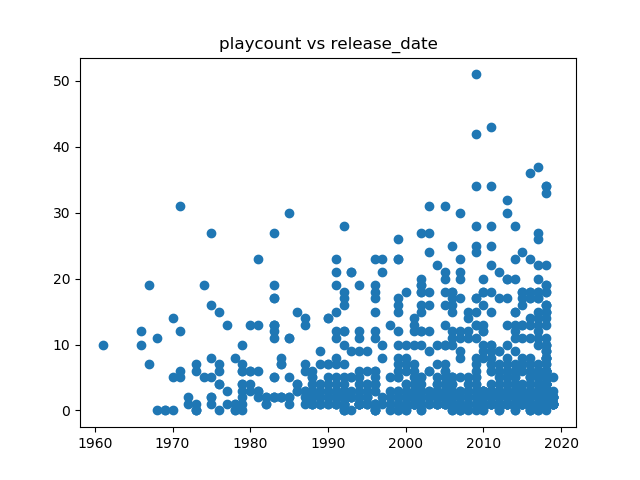
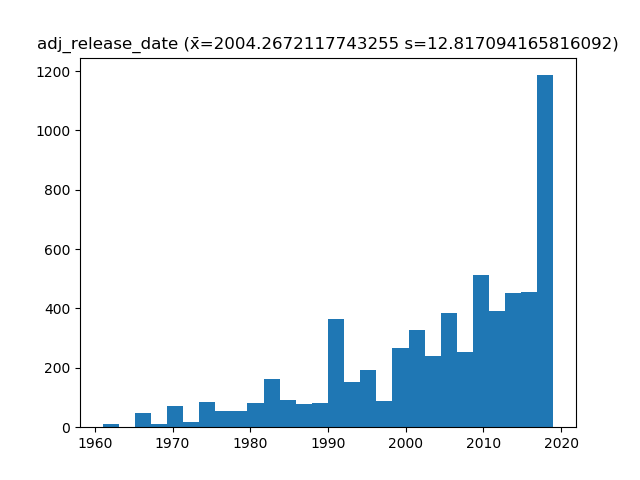
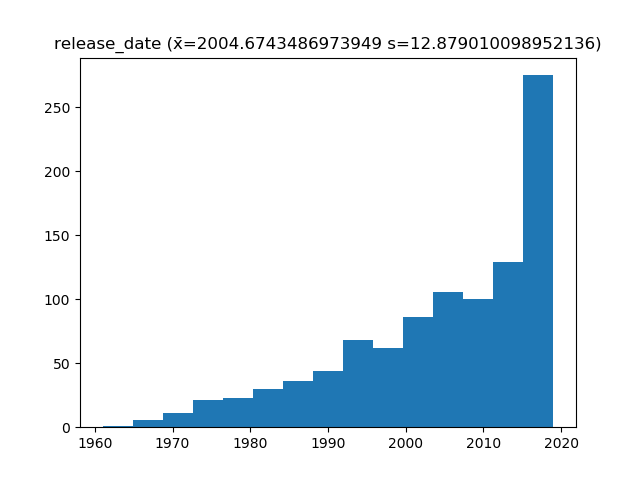
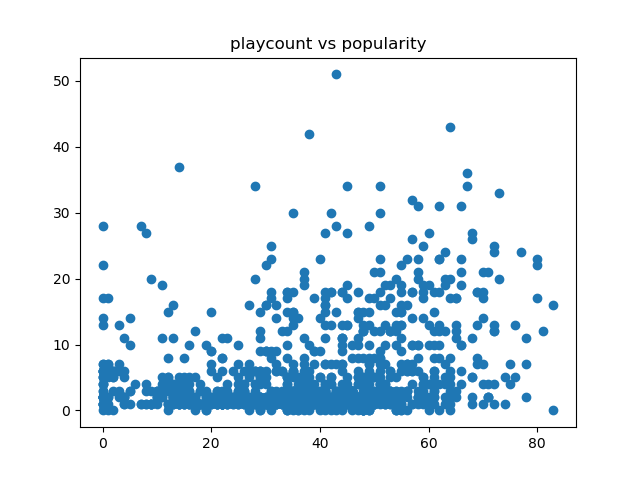
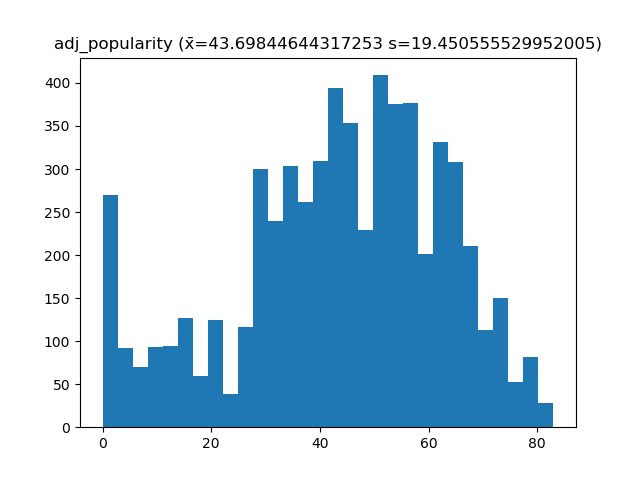
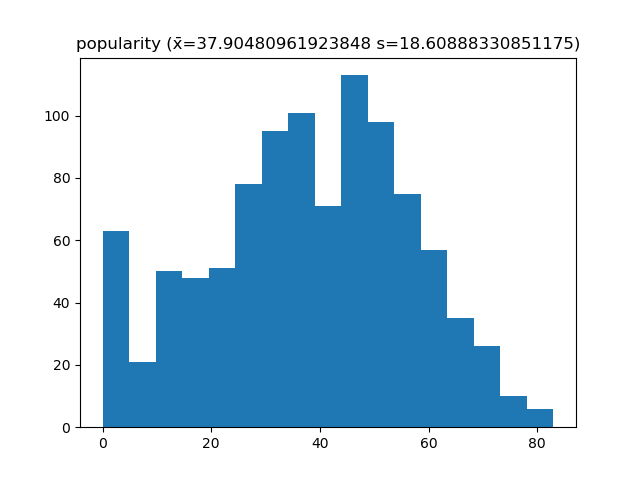
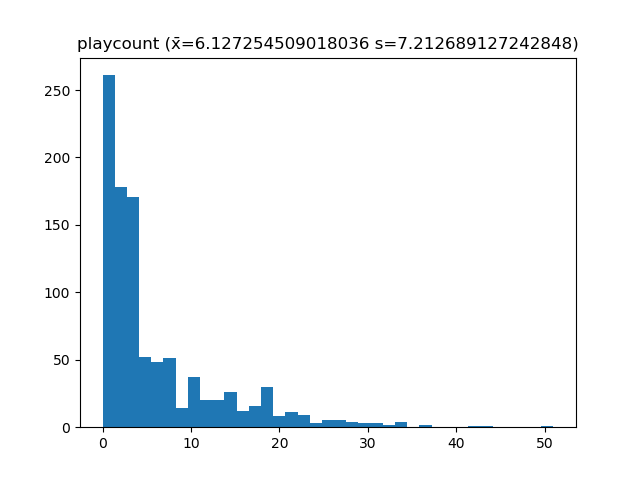
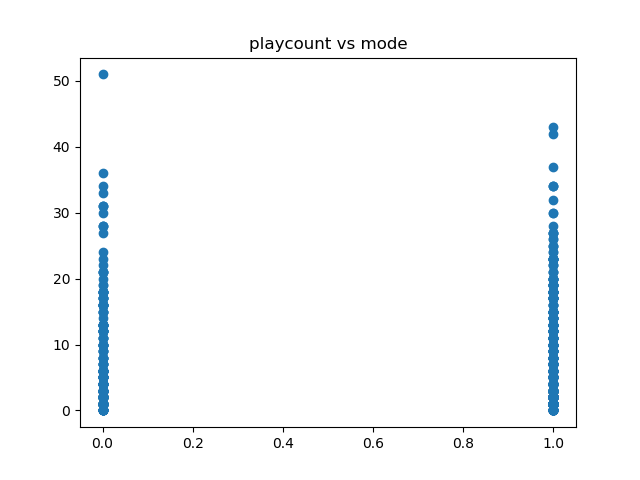
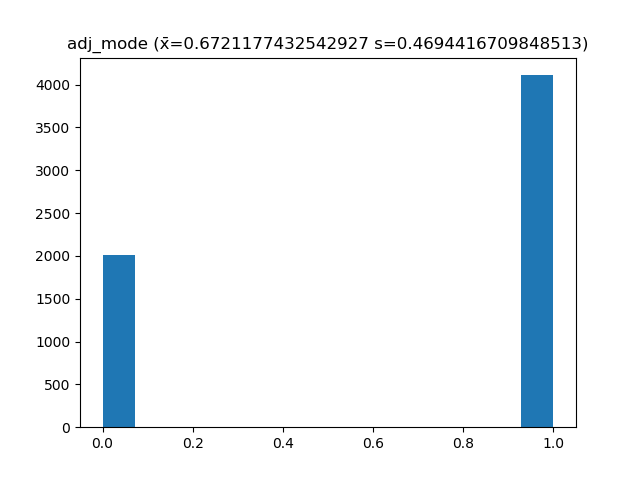
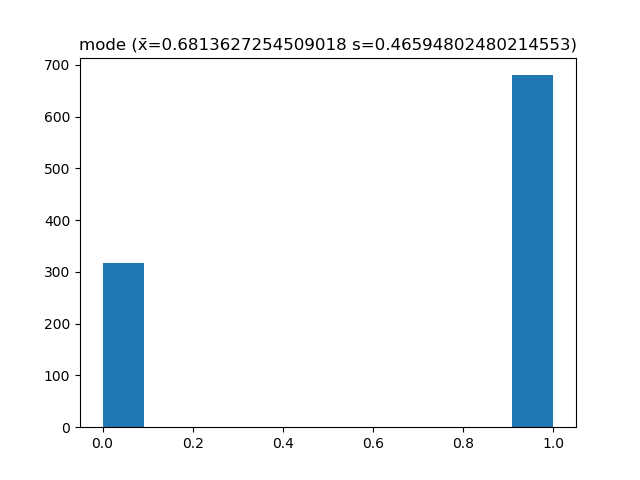
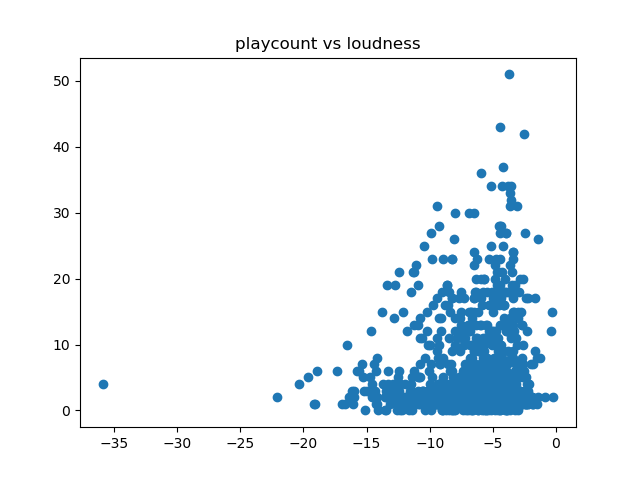
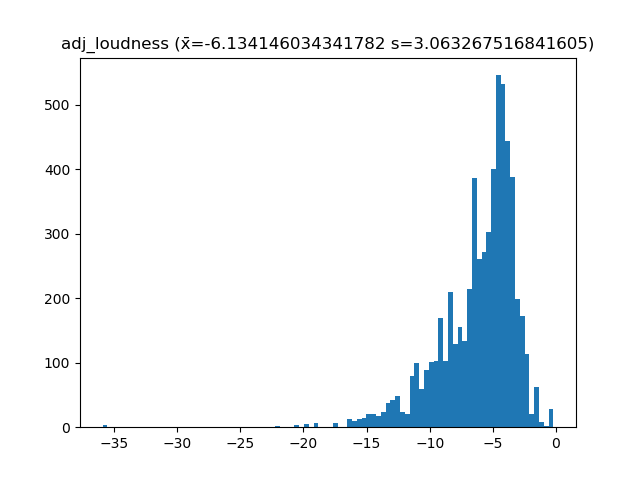
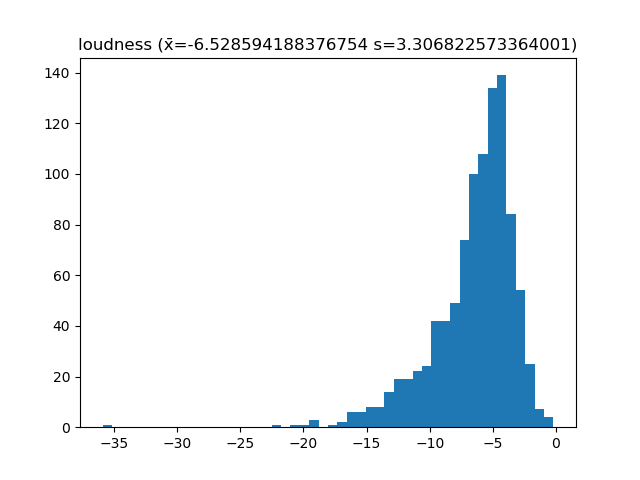
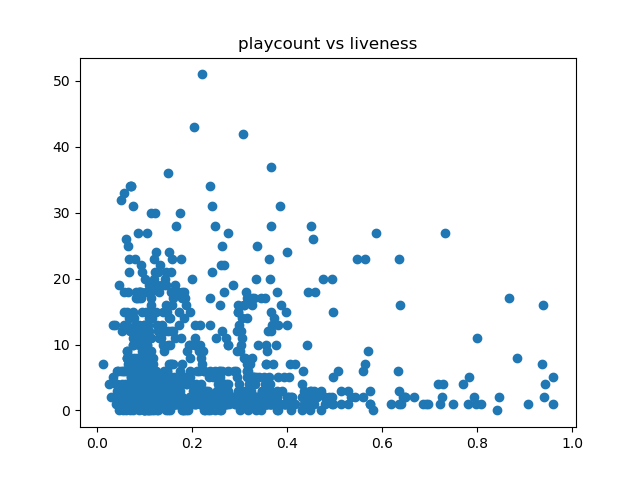
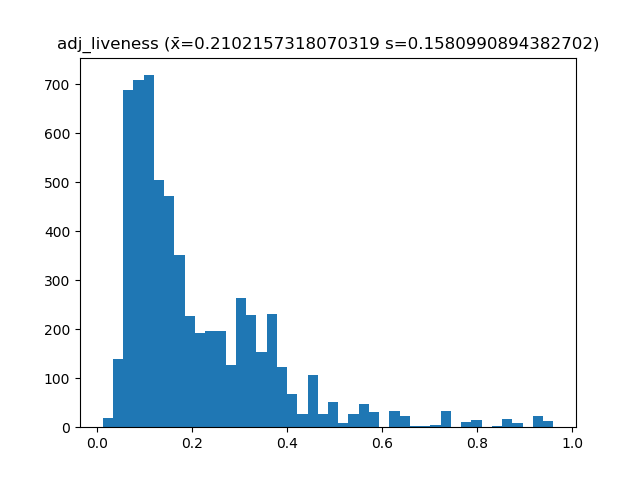
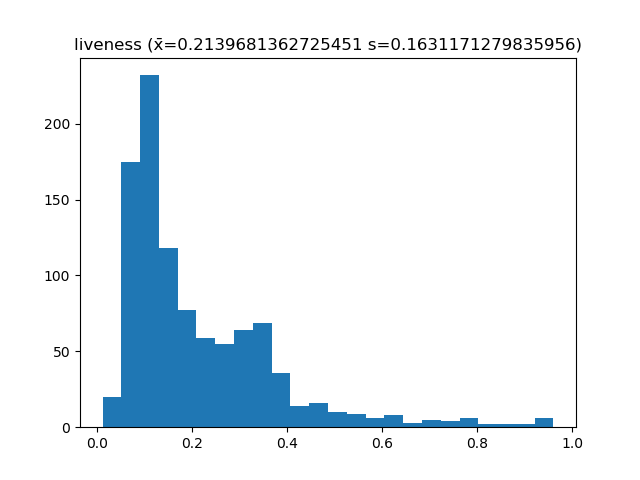
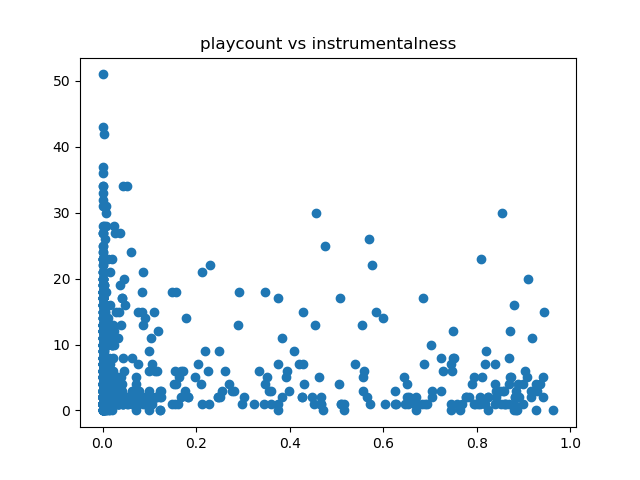
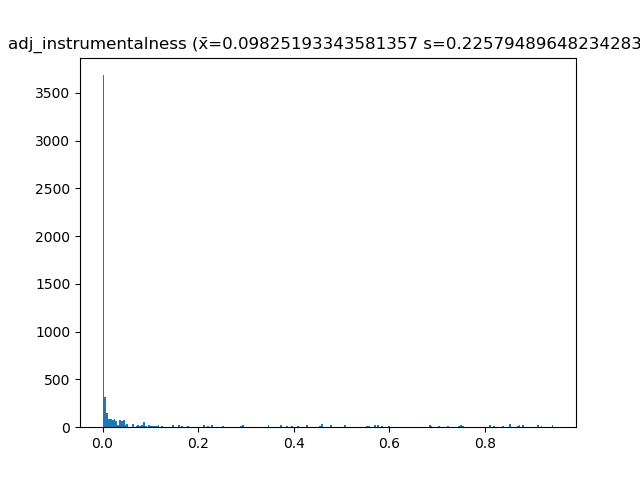
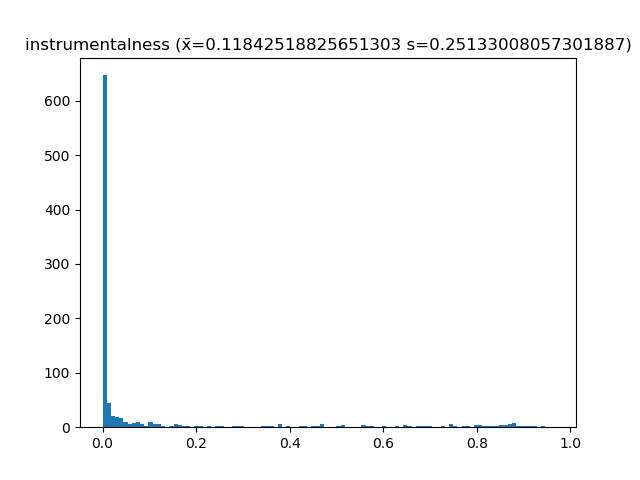
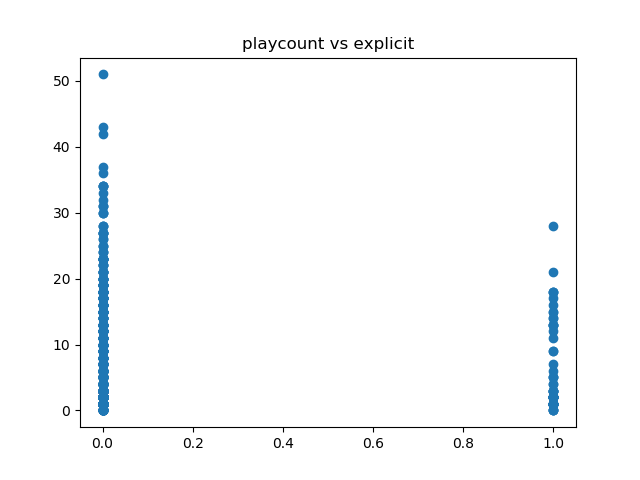
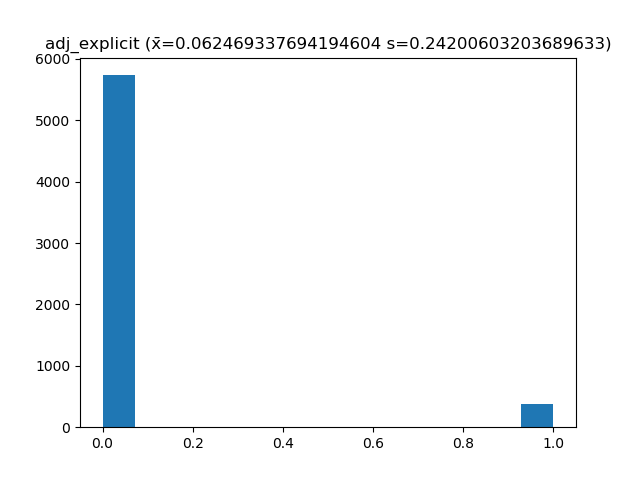
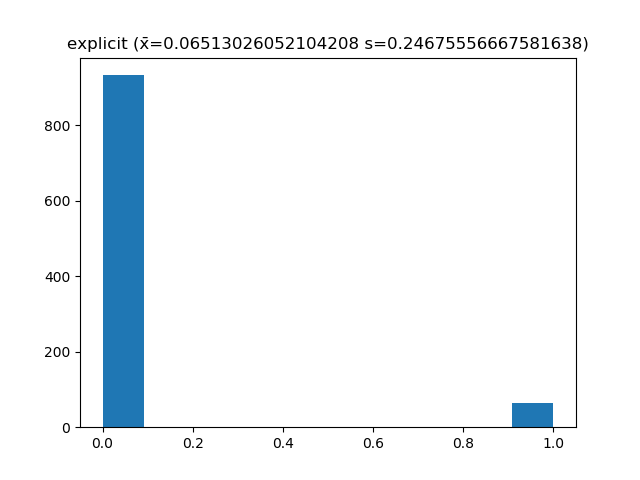
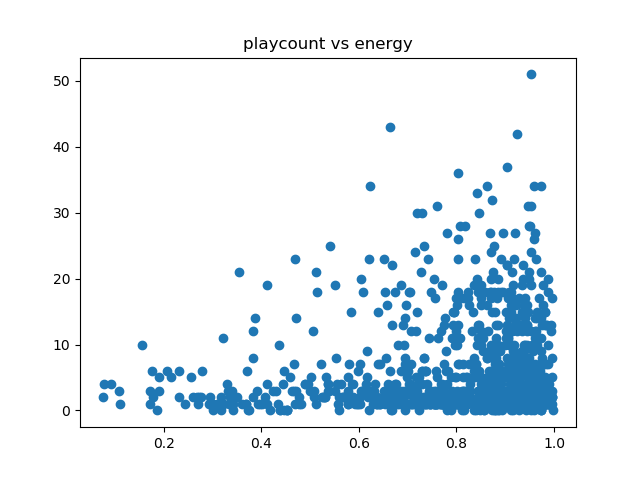
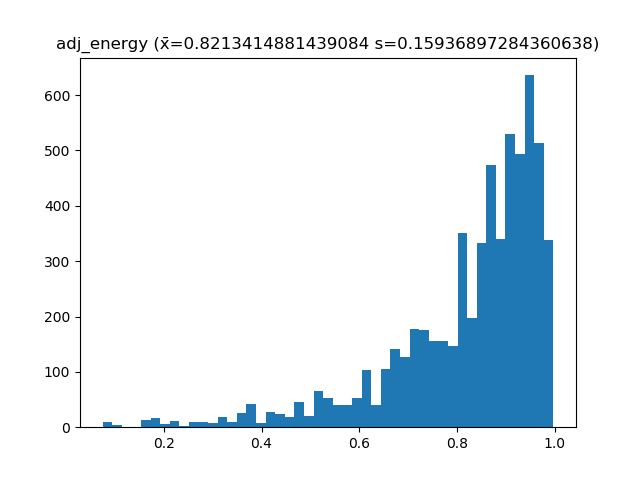
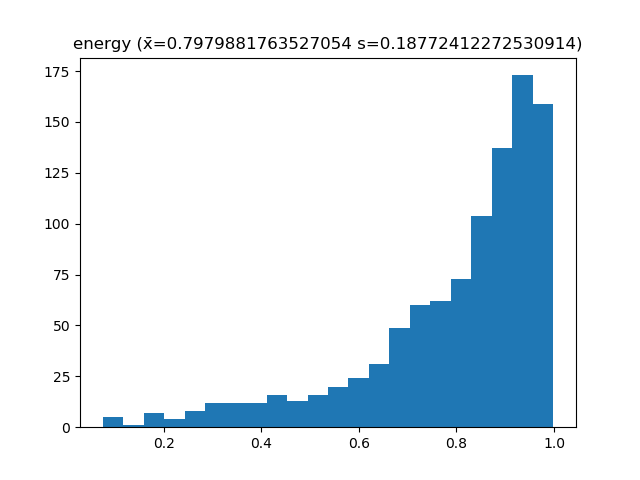
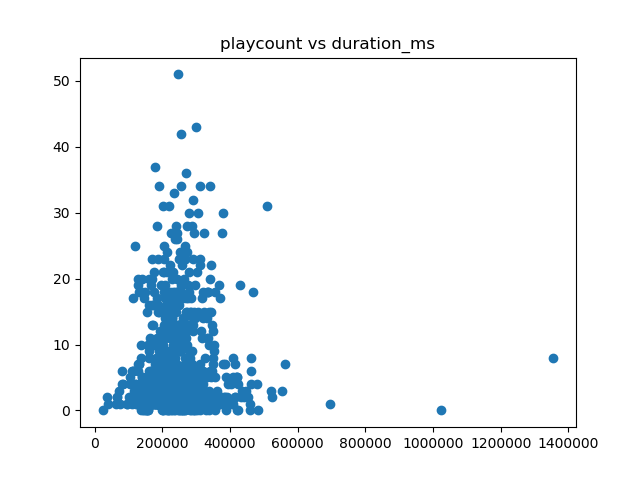
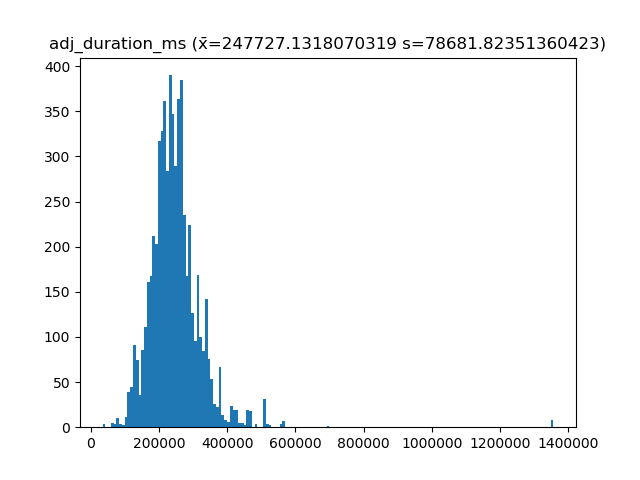
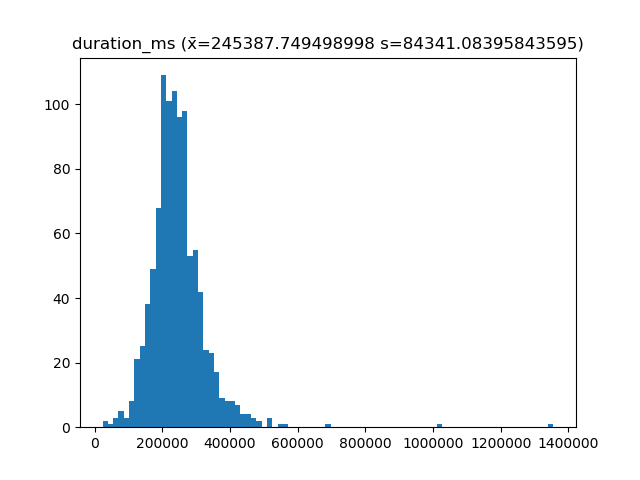
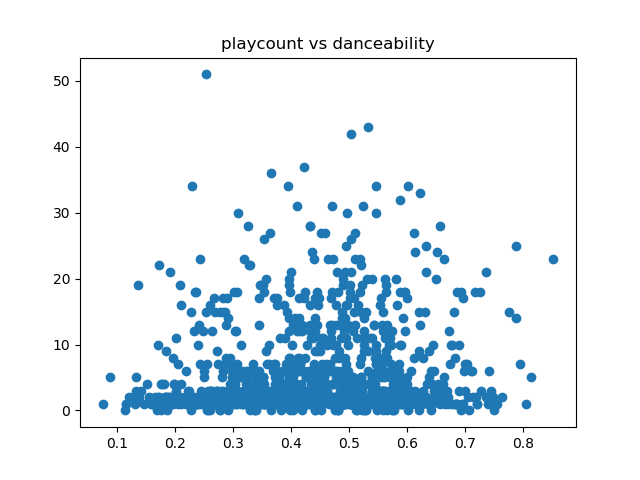
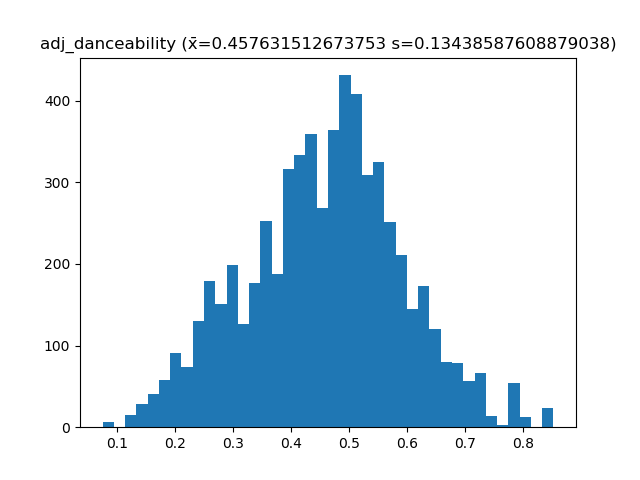
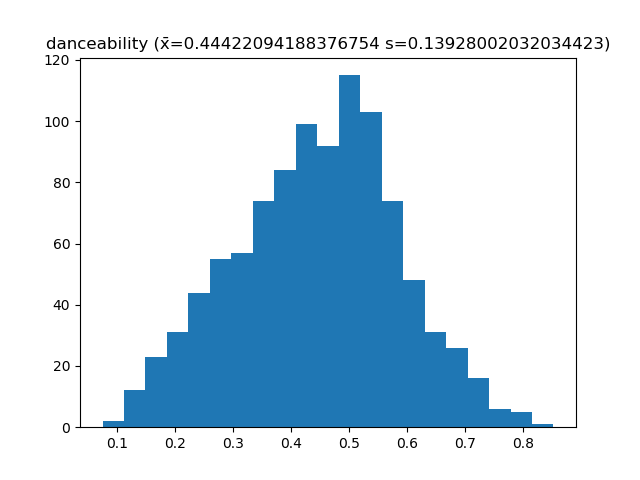
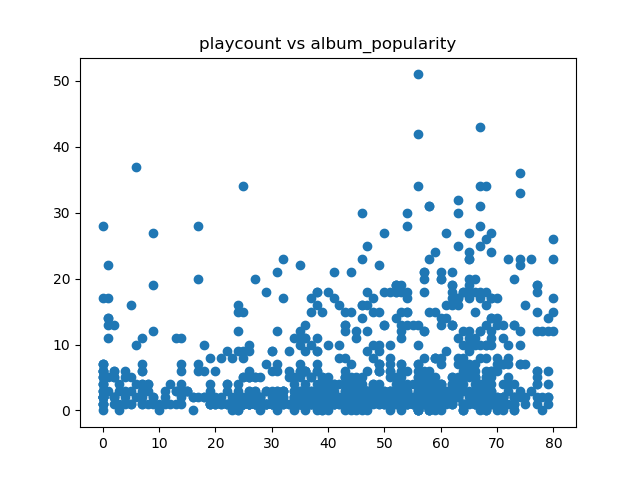
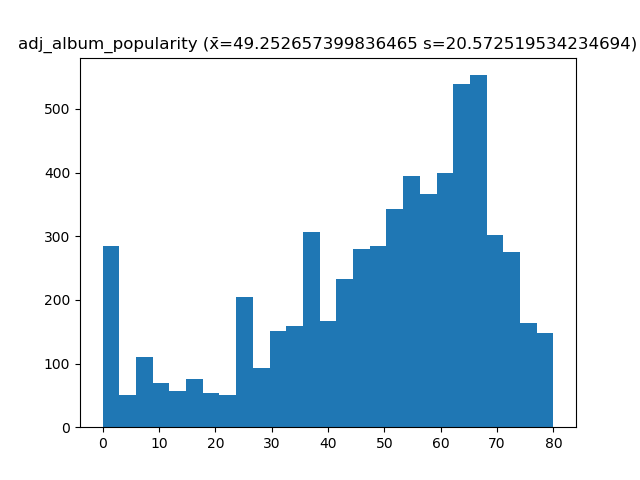
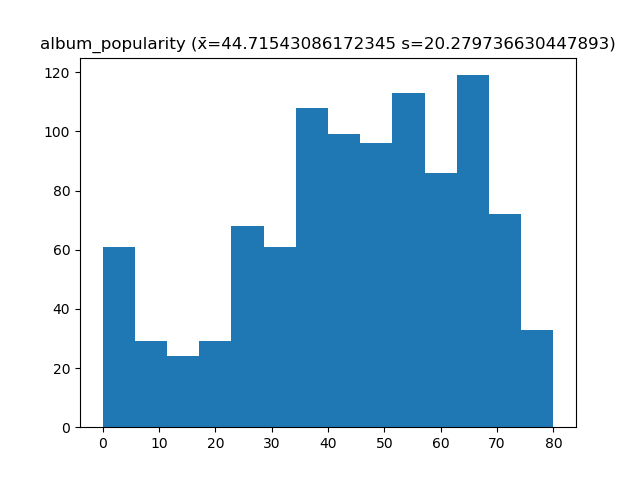
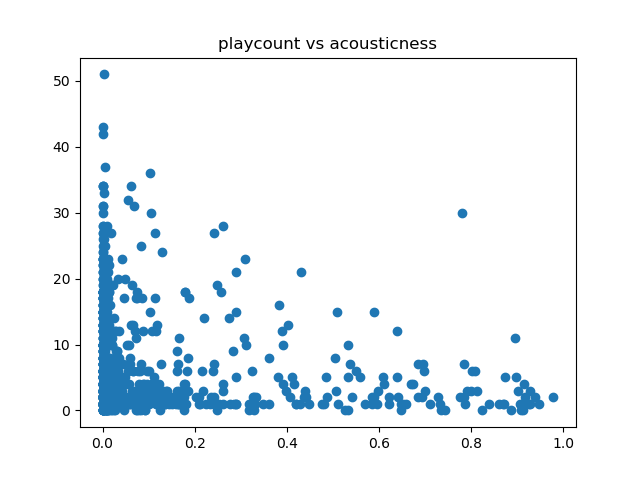
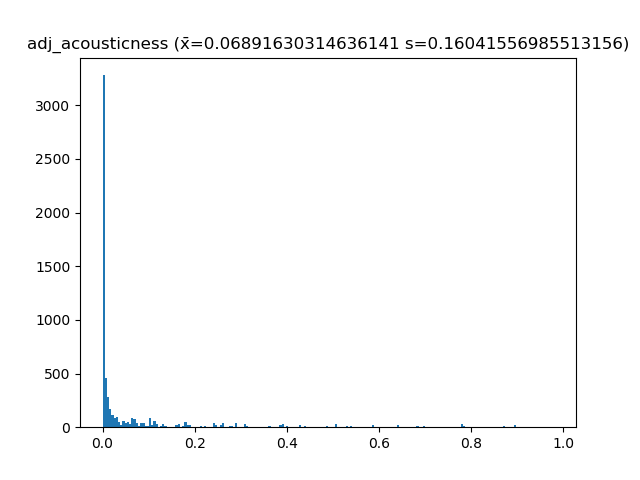
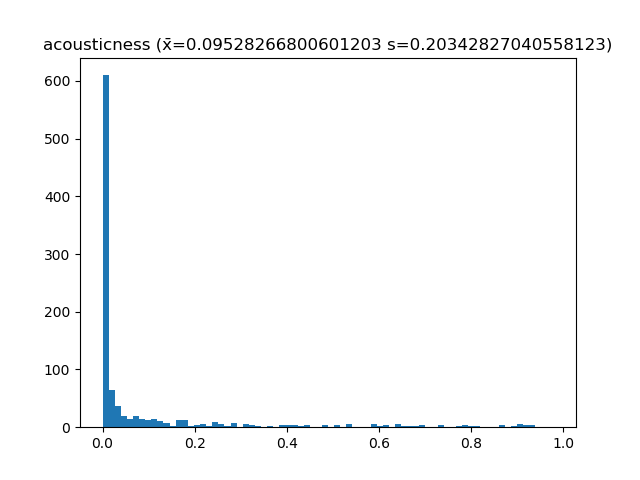
Finally, tempo is the estimated average beats per minute of the track and has the distribution in the following figure:



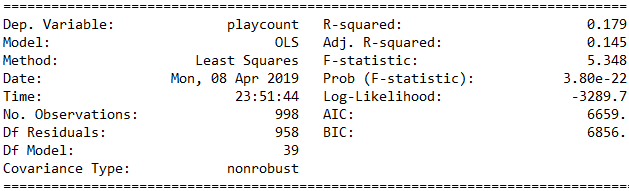
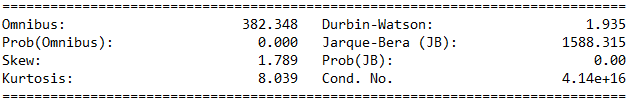
The data for this study was collected with a Python script which can be found in the file “Data Scraper.py” on this project’s GitHub page. The script parsed the entire Spotify library of Frank Pasqualini and got the audio features for a random sample of 1000 songs. This was done with the help of the Spotipy python package, which simplifies the Spotify API calls to be easily used in python scripts, but due to some limitations of the package a lot of work had to be done collecting the results in a useable form. The Spotipy package can only gather tracks fifty at a time, so we had to write the script to gather fifty songs, then jump down the list of the songs and gather fifty more in a loop until less than fifty songs were returned, which would be the end of the list. We then had to interpret the results from this data collection into a usable data structure with the variables we needed. The script used Frank’s Last.fm account to get the play count by submitting a GET HTTP request with the title and artist for each track gathered in the sample, and one of the functions in Spotipy to gather the rest of the audio features. Finally, the script took all of the variables we had collected and placed them into an easy to use data structure and exported the entirety of the structure to a file to be analyzed by a different script. We picked 1000 as our sample size because it is a large enough sample size that the results will be significant, but it is not so big that the analysis will take too long to run. Unfortunately, this sample is slightly biased towards songs with higher play counts and popularity, because the user is more likely to have added songs to their Spotify library if they have heard the song multiple times before, which is familiarity bias. This sample also did not return a full 1000 songs, because the data was gathered from two different sources and it was not possible to correctly link every song. Fortunately, only two songs from the 1000 song sample were left out, so the sample still has a strong 998 members.

After looking into the documentation for the Spotify API, we discovered two more variables that we thought might be of interest hidden in the attributes for the album instead of the attributes for the song. These variables are release\_date and album\_popularity. The variable release\_date is the year in which the album the song was released on came out, and album\_popularity is like popularity, but instead of for the individual song it is for the album. We suspect that album\_popularity and popularity will have high covariance. After some consideration, we decided to obtain a new set of 1000 songs with the two new attributes, which also ended up having a two-song loss. For the sake of simplification, we also decided to stop considering the effects of key and time signature, because having too many categorical variables overcomplicates the model, and we decided that those two would likely not have as much of an impact as mode and explicit.

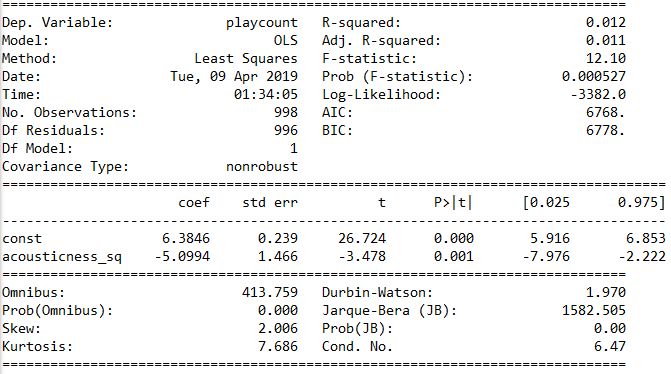
The first thing we did with the data we collected was generate histograms for every variable, in addition to every variable adjusted for playcount. To adjust for playcount, we took every value in the list and added it to the adjusted list multiple times, one for each play. The histograms of the non-adjusted variables represent the songs in Frank’s library, and the histograms of the adjusted variables represent the instances of songs that Frank has listened to. We also generated scatter plots of playcount vs all of the other variables. All of these graphs are pictured below:



One thing we noticed immediately is that some adjusted distributions, such as tempo and release date have a similar shape to the non-adjusted distributions but with more pronounced extremes, but that most of the scatter plots looked like they would not support regression, as most of the points were evenly spread around the low values of the y axis. To see if regression was worth looking into at all, we tried to create a model with every variable , every variable squared, every variable cubed, and every possible interaction term. The results are below:

The full results including all 823 variables and their coefficients, standard error, t scores, and more are available in the GitHub Repository under the Documentation folder in the file “Full Regression Results.txt” and all of the graphs displayed earlier are also available in the Documentation folder in the subfolder Graphs. The overall results were not promising, because even the very overfit model could only account for 17.9% of the variance. The most promising variables were acousticness\_sq with a t-score of 2.595, tempo with a t-score of 1.308, album\_popularity\_cu with a t-score of 1.183, and a few interaction terms with a t-score of over 1. Not a single term passed a T test at α=.05, and only acousticness\_sq passed a T-test at α=.10, but just barely. The regression run with just acousticness\_sq is shown below:



These results are significant, but account for very little of the overall variance. We concluded that an efficient regression model could not be made with the data we have.

Works Cited

“Get Audio Features for a Track.” Spotify for Developers, developer.spotify.com/documentation/web-api/reference/tracks/get-audio-features/.

Pasqualini, Frank J. “Spotify-Regression-Analysis-Data-Scraper.” GitHub, github.com/Frank-Pasqualini/Spotify-Regression-Analysis-Data-Scraper/.