Show Me What You Got

Song Popularity Prediction Using FMA Dataset

Team (in alphabetical order)
Yiyi Chen, Avi Dixit, Sayan Sanyal, Ed Yip

Special thanks to Jake Mainwaring for sharing his valuable expertise in music for feature generation.

Abstract

There are multiple factors that affect a song's popularity that can be both related and unrelated to a song's musicality, from the key and modality in which it's written, to the name of the artist, year of release, and type of marketing campaign. We attempted to build machine learning models to predict a song's popularity based only on its musicality features. Our model included custom and pre-existing features of a song such as its key, modality, loudness, dissonance, and dynamics variation among others. Our model was trained using available song files from the Free Music Archive and tested on 60 sample song clips from Spotify. The test dataset consisted of 3 different genres with 20 popular and 20 unpopular songs within each genre. We defined popularity as the first principal component of track listens, interests and user favourites. Our model found that popularity was best predicted in the Hip-Hop and Jazz genres with features such as "speechiness", "valence" and "instrumentalness" being the most effective. The genre that we were least able to predict popularity for was Pop. The most effective overall musical features to determine popularity across all genres were "speechiness", "acousticness" and "dissonance".

Table of Content

Prior Work

Predicting music popularity through machine learning is generally known as Hit Song Science and has gained traction over the last 10 years where data scientists and researchers have centered on finding and using different models to predict popularity songs (through the Million Song Dataset) given preset music and metadata features available through Echo Nest.

There has been work done at Stanford University where Pham et al. used various classification techniques (e.g. logistic regression, linear and quadratic discriminant analysis, support vector machines, and multilayer perceptrons on both music and artist features provided by Echo Nest (e.g. danceability, energy, duration, key, year, artist name, etc.)[1]. Popularity was defined using a preset Echo nest feature called "song hottnesss" and instead of focusing on feature engineering, they centered their work on feature extraction through Echo Nest and trying out different models to see which would give them the best predictions. Their work unsurprisingly demonstrated artist familiarity as one of the top most influential features.

Another study was done at the University of Antwerp in Belgium where researchers narrowed their focus to a single genre (dance music) to create a prediction algorithm that forecasted whether a song would be a top 10 hit[2] . They used Echo Nest to extract 139 different musical aspects from 3500 songs charted in the top 10 from 1985 to 2014 to analyze each song (e.g. song length, tempo (bpm), time signature, beat, energy, danceability, timbre, tone color, listener feel, etc.). Feature selection and normalization were done through CfsSubsetEval from Weka with GeneticSearch to reduce data to 35-50 attributes. 5 models were built for each dataset to determine whether songs would be a hit or not: decision tree variation of a C4.5 tree, RIPPER ruleset, Naive Bayes, Logistic Regression, and SVM using Polynomial and RBF kernel. Logistic regression (with an 83% accuracy score) performed the best. This study focused on solely the musical features, but was based on creating models and not engineering musical features given the dataset.

The last prior work whose approach we felt was worth mentioning was by Matthew Moocar, who used Apache Spark to implement Alternating Least Squares model to predict song listens for a given user[3]. Collaborative filtering was used to approximate user matrices by factorizing the two matrices of properties of a user with properties of each song. ALS was then used to minimize the error in predicting the number of plays by using a fixed set of user factors and the known number of plays to find the best song factors using least squares optimization and then switching out user factors with song factors to find the best user factors. Accuracy for the approach was not very good with an RMSE was around 9 for predicting playcounts of songs excluding a playcount of 1 whereas predicting songs including playcounts of 1 had an RMSE of around 6.

[1] Predicting Music Popularity, Pham et. al, http://cs229.stanford.edu/proj2015/140_report.pdf
[2] Machine predicting hit dance songs (2015)
[3] Making Music Prediction with Apache Spark

Datasets

Training

We used the dataset compiled by Defferrard, Benzi, Vandergheynst, and Bresson, sourced from Free Music Archive. The full dataset includes 106,574 tracks. The data for the tracks come from the below three sources -

- 52 metadata features from **FMA**, e.g. genre, title and date_released
- 518 audio features from **Librosa**, e.g. Mel-frequency cepstral coefficients and spectral contrasts
- 250 musicality features from **Echo Nest**, e.g. acoustic-ness, danceability and energy. This set of features is only available for 12.3% (13,129 tracks) of the data

The creators have also partitioned a small, medium and large sample of the entire dataset for faster processing. We used songs in the large sample with complete sets of features (metadata, audio and musicality). After dropping tracks with missing features, we ended up with 9,350 tracks, on which we then performed a 80/20 train-test split for our models.

External Validation

To test our model's performance on real world data, we picked 3 top genres in our training dataset - Hop-Hop, Rock and Electronic - and manually collected 40 songs in each genre from Spotify that's not already in our training dataset. Each set of 40 songs contains 20 popular and 20 unpopular songs, with popularity loosely determined by online review from Google search and Spotify play count. We then obtained the metadata, musicality features, and 30 second sample of these songs using Spotify's API.

Analysis

Feature Generation

Pre-Existing Features

Even though the dataset was intended for benchmarking music genre recognition algorithms, we envisioned that the rich set of features would also contain useful information for popularity prediction. Specifically we hypothesized that the audio features would uncover the key audio characteristics / song structures that set popular songs apart from the rest. During our data exploration, we found out that the audio features were not as informative as we had hoped, as our models were performing nearly randomly with them. We also did not use the metadata features (except for duration), as we wanted to minimize the influence of features outside the song itself, e.g. name of the artist and album. We did so because in addition to validation our model on a hold out test dataset, we wanted to scrape data from the real world (Spotify) to see how our model generalizes. Since Echonest was acquired by Spotify, there was considerable overlap in the features, but we discarded anything that was not available.In the end, we only kept 8 musicality features, which characterize a song by its danceability, energy, tempo and etc. from Echo Nest in our final models. The list of 8 features with their descriptions can be found in appendix. Luckily, we were also able to get a 30 second preview of the songs from Spotify for our hand generated features.

Representation Learning
We also tried training neural models using raw song data in hopes of discovering some latent feature representations. The models performed very poorly, likely due to two interrelated reasons - large input space and relatively few number of observations. Most successful neural model implementations in other projects use pre-trained models with large amount of input data. With raw input of 12,000 features per datapoint and < 10,000 observations, it is of little surprise that our models were performing worse than random.

Hand Engineered Features
As the pre-existing features turned out less informative than expected, we spent a lot of time crafting our own features, with the goal of capturing inherent characteristics of a song that listeners use consciously/subconsciously to determine if they would enjoy a song.

Feature	**Values**	**Description**
Pitch	String of value [C, C#, D, D#, E, F, F#, G, G#, A, A#, B	The main key of the song. Inferred from the most common pitch class from the chromagram of a waveform
Mode	1 for major, 0 for minor	Modality of the song. Inferred from the sets of keys commonly used in relative to the main key.
Dissonance	A float between 0 and 1	Percentage of dissonant chords (those outside the main chords for the inferred key and mode) in a song Calculation: number of observations on chromagram outside the main chords / total number of observations on chromagram
Max Diff	A float between 0 and 1	The biggest drop/rise in mean amplitude over 1/50 sec time window relative to the max amplitude Calculation: max(diff(mean amplitude for a 1/50 sec window) / max(amplitude)
Drop Rise	A float between 0 and 1	% of large amplitude changes over 1/50 sec time window of all observations Calculation: number of amplitude changes > 0.5 of max amplitude / total number of time windows
Key Change Frequency	A float between 0 and 1	% of 5 sec time window that has a different key (determined by the most common pitch during that 5 sec) from the main key Calculation: number of 5 sec time windows with a different key / total number of 5 sec time windows
Diff Samples	A list of 10 floats between 0 and 1	10 difference values between two 5-sec time interval evenly sampled across the entire audio file

		Calculation: 10 diff(mean amplitude for a 1/50 sec window) sampled at even intervals from the audio file

Hand generated features with descriptions

Model

Assumptions

Definition of Popularity
At the very onset, we had to deal with the question - what makes a song popular? We were constrained by the signals that are available in the FMA dataset. Within the dataset, we had three signals that mapped to ideas of popularity of a particular track. We have two features that were measures of user action - listens and favourites. The third feature is "interest". While initially ambiguous, we were able to glean from a page buried deep in FMA's website that interest is a score FMA calculated for a given track that is supposed to indicate how popular that track was going to be. However, favourites and interests were very sparse and did not have information for a majority of songs. Even listens, where the data was more durable predictably followed a pareto distribution with a very few of the songs getting a bulk of the listens.

Furthermore, we thought that the different signals gave us different aspects to what made a song popular. Multiple listens could involve a song that was a guilty pleasure, whereas interests could signal aspirational taste. We combined these different features using the first Principal Component that captured the maximum variation between these signals and used that as our popularity index. This was highly skewed, and applying a log transformation made distribution more normal. For classification, we used a percentile (set to 80 percent for our results) based threshold to achieve the cut-off between popular or not.

Genre based prediction
We postulate that since we are looking at largely audio based features, the variation between genres would be greater than the variation within genres and models would be able to better pick up nuances in the audio that leads to its genres track popularity. While this reduces the total data we had to train on, we were able to get very different results within genres that were better than a classifier on all data.

Model Overview

To get started with this dataset, we initially pondered over whether this was a regression or a classification problem. We had created a continuous index for popularity, and regressing over that value seemed like a natural step. So, we tried just that with the initial features that we had available to us. Very early, it was clear to us there was not a clear linear relationship that could be exploited to gain an quick-win model. Our R^2 values hardly crept up above 0.04 with stubbornly high RMSE value.

We decided to make this a classification problem in order to see whether we could separate the winners from the rest. Since we could observe our pareto distribution, we hypothesized that there could be enough signal in the dataset to distinguish these outliers from the rest of the dataset. We thus set a high threshold and converted it to a classification problem.

Since a large part of our project was to understand what about a song leads to its popularity, we started with the simplest but the most interpretable model - logistic regression. While trying to understand the performance of the model through cross validation, it was clear to us that there was high bias and low variance in the model. This was clear to us as we saw the high training error with an unregularized model. The regularized models (Ridge, Lasso and ElasticNet) expectedly performed worse than Logistic Regression out of the box.

Subsequently, we proceeded to experiment with tree based classifiers. Since we knew that our model had a high bias, and still needed to be slightly interpretable, we tried to work with Random Forest classifiers where multiple simple models would hopefully be able to come together to reduce the bias of our model. While more successful than the simple logistic regression, a more complex ensemble classifier did not give the performance boost we thought we would get.

To solve our unbalanced problem, we tried a couple of tricks from the handbook - to very little success. We up-sampled the dataset to near parity between the classes, discarded accuracy as a metric and tried to experiment with the "class_weights" hyper-parameter for the last two classifiers. These methods did not provide us a substantial difference to our cross-validated results. Our classifiers were largely learning to predict the majority class, and not able to pick up the right signals to understand what made a song popular.

Lastly, we tried to use a gradient boosted decision tree. Given that there were very few positive classes, we wanted to ensure that we focussed on the examples we were getting wrong - a perfect job for a gradient boosted tree. Our final attempt was relatively successful, and the model ended up doing better on most of the genres. However, the differences between the models were exploited in a soft voting classifier.

We think there is scope to perform exhaustive grid search over the more uncommon hyper-parameters to see the performance improves.

Model evaluation
A decision we made was to focus on the AUC score for the ROC curve over metrics such as precision, recall or variations of the f-beta score. We did so because through this metric we are able to focus on weighting the cost of missing a popular song (False Positive Rate) vs the benefit of getting it right (True Positive Rate).

Generalizability

Our plan to test our model for out of sample generalizability was rather ambitious. Not only did we want to have a hold out set from our data to test our models on, we went out and collected data from the Spotify API (available features as well as a similar 30 second sample) so that we could recreate our features from real world data. We thought that this would be a rather robust test of our real world generalizability - and it proved to be just that.

Results

We built and evaluated our models for each of the top 8 genres individually. For external validation, we picked three genres that had the highest representation in our training data and evaluated the model's' generalizability. Our final classifier was an Ensemble Classifier with soft voting composed of the following individual classifiers: Gradient Boosting Classifier, Random Forest Classifier, and Logistic Regression.

Overall Performance

Random Forest

	Hip-Hop	Pop	Rock	Folk	Jazz	Electronic	Classical	Old-Time
Accuracy	0.79	0.78	0.78	0.79	0.80	0.78	0.80	0.78
Precision	0.47	0.23	0.26	0.37	0.377	0.26	0.20	0.10
Recall	0.12	0.07	0.03	0.07	0.15	0.04	0.03	0.02
AUC	0.60	0.52	0.52	0.60	0.63	0.53	0.53	0.50

Logistic Regression

	Hip-Hop	Pop	Rock	Folk	Jazz	Electronic	Classical	Old-Time
Accuracy	0.27	0.30	0.21	0.25	0.44	0.20	0.50	0.38
Precision	0.21	0.16	0.20	0.21	0.22	0.20	0.27	0.18
Recall	0.94	0.63	1.00	0.96	0.70	1.00	0.74	0.66
AUC	0.68	0.47	0.58	0.61	0.52	0.55	0.56	0.50

Gradient Boosted Classifier

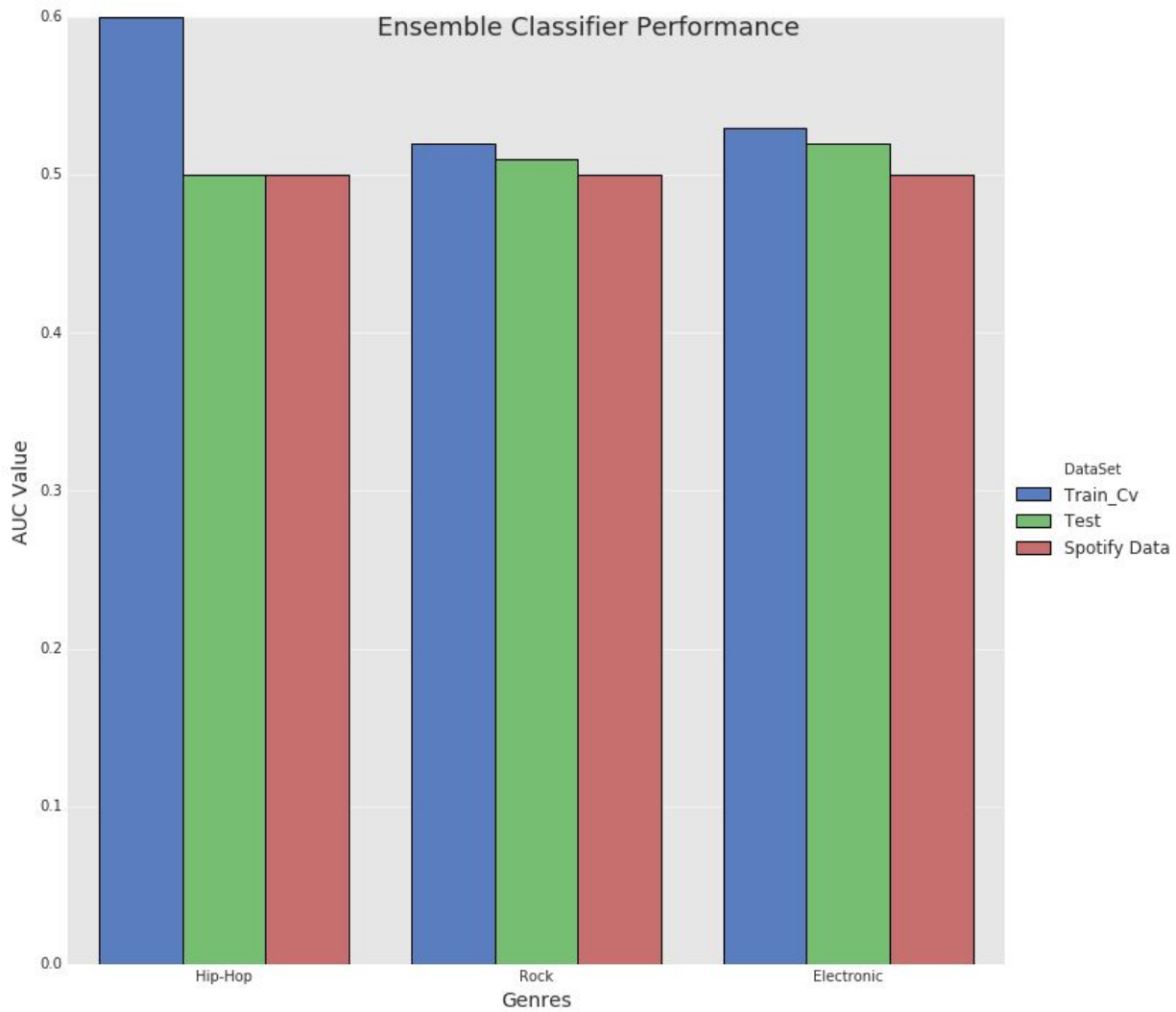
	Hip-Hop	Pop	Rock	Folk	Jazz	Electronic	Classical	Old-Time
Accuracy	0.79	0.75	0.79	0.77	0.79	0.79	0.79	0.79
Precision	0.38	0.22	0.25	0.19	0.55	0.37	0.37	0.56
Recall	0.16	0.05	0.02	0.06	0.23	0.06	0.22	0.23
AUC	0.74	0.44	0.59	0.59	0.68	0.62	0.62	0.60

Classifier Performance on FMA Test Set

Random Forest and Gradient Boosting classifiers have relatively low AUC and recall scores. Upon inspecting the confusion matrix, we realized both models are predicting everything as unpopular, and therefore both numbers of true positives and false positives are 0. We tried tuning models to focus more on the positive training examples, but the results did not really improve, possibly due to lack of good signals from the training data.

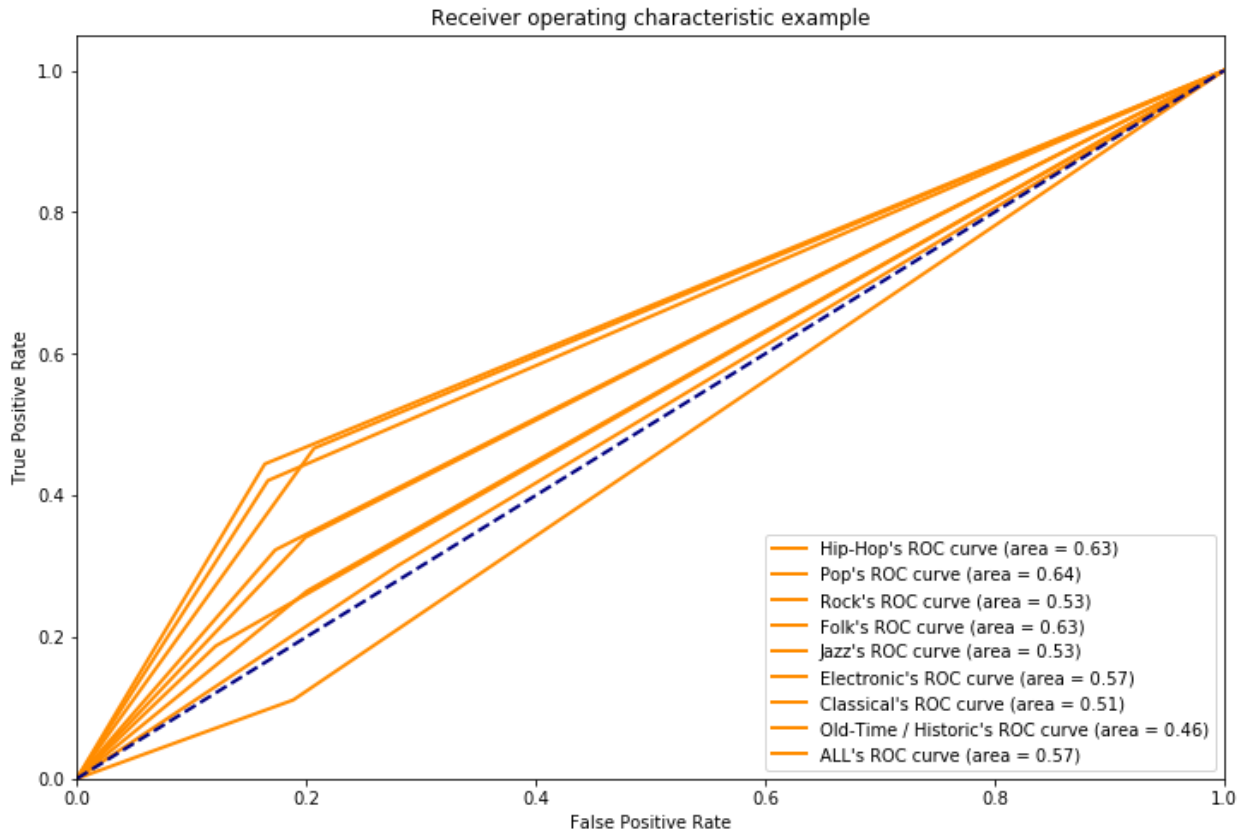
The Logistic Regression classifiers seemed to have the opposite challenge, with considerably lower accuracy and much higher recall scores. We were able to bias the model to focus more on the positive samples, which lead to an over tendency to predict everything as positive.

Given the above observations, the features did not seem to allow a meaningful separation of popular songs from the rest, as the models either treat everything as unpopular or popular.



External Test Performance on Spotify Data

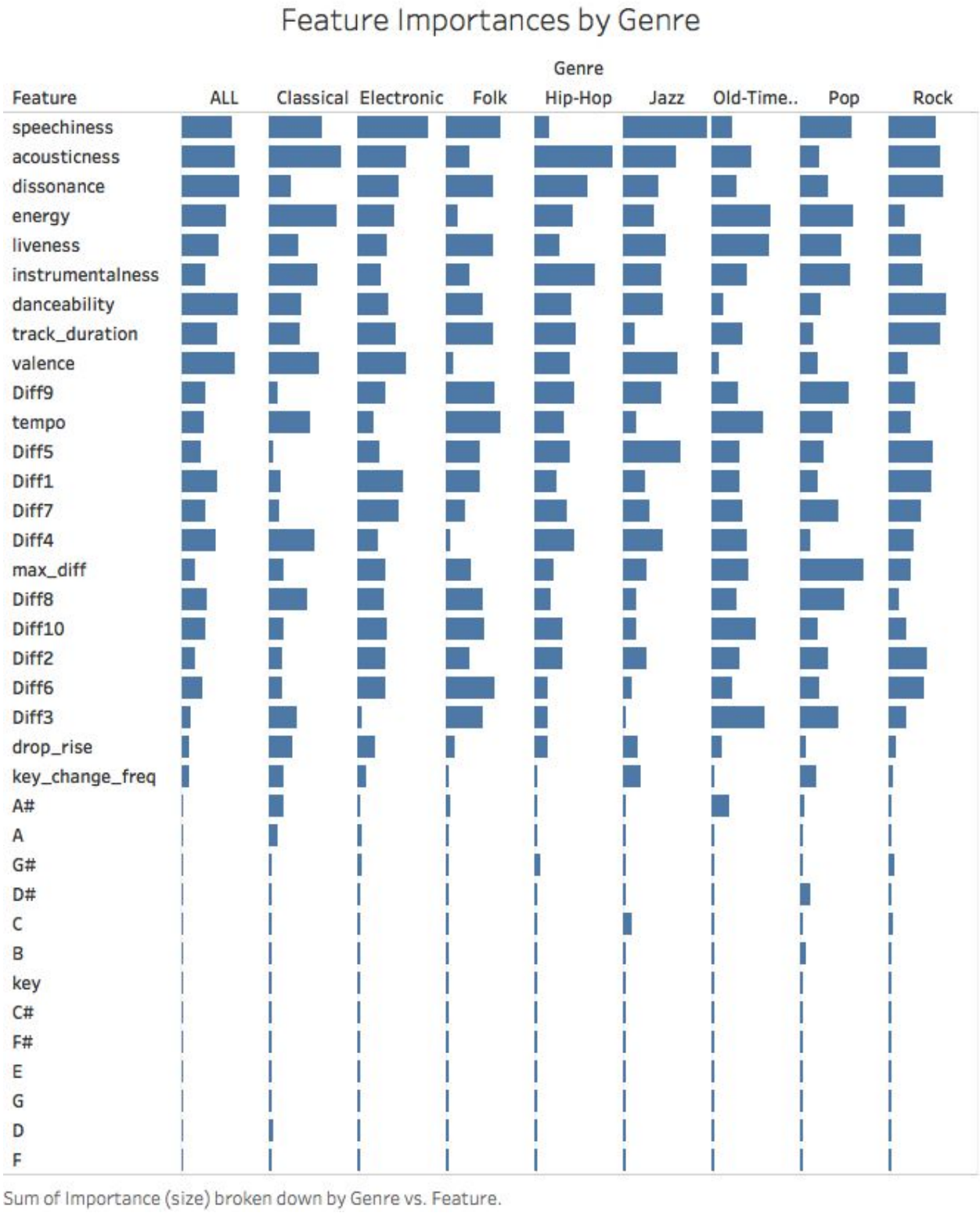
We evaluate the performance of our final Ensemble classifier across the three genres and notice that the AUROC value for all three genres is very close to 0.5, for all the actual test data we collected from Spotify. Given that we do not observe very good AUROC values even for training data, it appears that the features we have created and the metadata we have for all the tracks are not really indicative of the popularity of the songs. It would appear that there are some other inherent features present in a track that are not captured by our classifiers at all. We comment more on this in our conclusion when we consider the future steps we might take to improve our classifier.



AUC_ROC for all genres

Examining the AUC_ROC curve of all the classifiers, we see that they perform quite poorly across all genres. The popularity predicted for all genres was almost as bad as random classification which means that all classifiers struggled with predicting true positive values and were able to achieve higher accuracy scores due to imbalanced data. Rescaling the data to improve the balance did not improve the score to a large extent, which lead us to suspect that we were missing some crucial features from our training data.

Feature Importance



Feature importance by Genre

We find some expected and contradictory results when looking at feature importance by genre. Considering a feature like *Speechiness,* we notice that though it is an important feature for almost all genres, it is particularly low for Hip-Hop and Old-Time. While it is expected that *Speechiness* would be an important feature for genres such as Pop and Rock, which have a healthy mixture of both lyrics and music, it also has very high importance for Electronic and Classical genres. There are music types in which we would not generally expect lyrics to play a large part in determining the popularity of the song, but it appears that having some kind of lyrics in these song types does contribute to the popularity of the song.

Similarly, examining a feature like *Danceability*, we would expect it to contribute highly to determining the popularity of tracks belonging to the Electronic and Hip-Hop genres, as these genres are considered to the most danceable among the ones we are testing against, but once again we see that it does not impact as highly the genres we expect it and instead is also an important feature for genres such as Classical and Folk.

Looking at the most important feature per genre for each individual classifier that we worked with, we notice that very few genres had the same features marked as the most important one. A significant number of genres had the most important feature denoted as *Diff Samples*, which measures the change in amplitude at evenly sampled timestamps. The other three significant features were *Speechiness* and *Acousticness* of a track which indicate that having acoustic instruments and vocals in a track contributes positively to the popularity of the track. Another hand created feature that seemed to be indicative of song popularity is *Dissonance*, which describes if a song has a lot of unexpected tunes.

	Hip-Hop	Pop	Rock	Folk	Jazz	Electronic	Classical	Old-Time
Random Forest	Diff8 (0.065)	Diff7(0.1)	Speechiness (0.06)	Speechiness (0.095)	Diff7 (0.11)	Acousticness (0.1)	Instrumentalness (0.17)	Diff5 (0.11)
Gradient Boosting	Acousticness(0.12)	Diff8 (0.085)	Track Duration (0.08)	Track Duration (0.07)	Diff9 (0.12)	Speechiness (0.08)	Tempo (0.12)	Max Diff (0.09)
Logistic Regression	Instrumentalness (0.85)	Diff1(1.0)	Track Duration (0.25)	Track Duration (0.30)	Valence (2.4)	Danceability (0.375)	C (0.13)	Liveness (0.8)

Conclusions

The work presented here is preliminary and more an exploration of whether it is possible to predict music popularity through solely musical/audio features. We conclude that though hand generated features can provide an effective way of differentiating which features by genre are most important for predicting popularity. Our models best predicted popularity in Jazz and Hip-Hop and the worst in Pop.

We suspect that our model's prediction capabilities were limited by the song selection we had access to through the Free Music Archive and the 30 second limit of the songs in our test dataset. We believe that having a training set more representative of music that is commonly known and listened to will lead to more accurate predictions with our real world test datasets. Furthermore, having full length songs in our test dataset will allow us to determine the robustness and full efficacy of our hand generated features. Having a more generalizable dataset will also allow us to refine our features.

Our next steps are create more features that include interaction terms (e.g. tempo + modality) as other studies have done to determine any multiplicative effects especially for more beat/musicality driven genres like jazz, pop, and dance. Additionally, we want to refine our metric for popularity and incorporate other elements such as iTunes downloads and YouTube play counts.

Given our dataset limitations and small number of genres tested, we believe there is much work we can do to improve the accuracy of our prediction model. Only in a context where we have trained our model using representative songs, established a clear method of defining song popularity, and a clear classification of songs in each genre, can we say with certainty whether we can predict popularity using musical/audio features alone.

Appendix

A. Description of 8 features used from Echo Nest (source)

Echo Nest Feature	**Description**
acousticness	A confidence measure from 0.0 to 1.0 of whether the track is acoustic. 1.0 represents high confidence the track is acoustic.
danceability	Danceability describes how suitable a track is for dancing based on a combination of musical elements including tempo, rhythm stability, beat strength, and overall regularity. A value of 0.0 is least danceable and 1.0 is most danceable.
energy	Energy is a measure from 0.0 to 1.0 and represents a perceptual measure of intensity and activity. Typically, energetic tracks feel fast, loud, and noisy. For example, death metal has high energy, while a Bach prelude scores low on the scale. Perceptual features contributing to this attribute include dynamic range, perceived loudness, timbre, onset rate, and general entropy.
instrumentalness	Predicts whether a track contains no vocals. "Ooh" and "aah" sounds are treated as instrumental in this context. Rap or spoken word tracks are clearly "vocal". The closer the instrumentalness value is to 1.0, the greater likelihood the track contains no vocal content. Values above 0.5 are intended to represent instrumental tracks, but confidence is higher as the value approaches 1.0.
liveness	Detects the presence of an audience in the recording. Higher liveness values represent an increased probability that the track was performed live. A value above 0.8 provides strong likelihood that the track is live.
speechiness	Speechiness detects the presence of spoken words in a track. The more exclusively speech-like the recording (e.g. talk show, audio book, poetry), the closer to 1.0 the attribute value. Values above 0.66 describe tracks that are probably made entirely of spoken words. Values between 0.33 and 0.66 describe tracks that may contain both music and speech, either in sections or layered, including such cases as rap music. Values below 0.33 most likely represent music and other non-speech-like tracks.
tempo	The overall estimated tempo of a track in beats per minute (BPM). In musical terminology, tempo is the speed or pace of a given piece and derives directly from the average beat duration.
valence	A measure from 0.0 to 1.0 describing the musical positiveness conveyed by a track. Tracks with high valence sound more positive (e.g. happy, cheerful, euphoric), while tracks with low valence sound more negative (e.g. sad, depressed, angry).