WHAT MAKES FOR A HIT POP SONG? WHAT MAKES FOR A POP SONG?

NICHOLAS BORG AND GEORGE HOKKANEN

Abstract. The possibility of a hit song prediction algorithm is both academically interesting and industry motivated. While several companies currently attest to their ability to make such predictions, publicly available research suggests that current methods are unlikely to produce accurate predictions. Support Vector Machines were trained on song features and YouTube view counts to very limited success. We discuss the possibility that musical features alone cannot account for popularity. Given the lack of substantial findings in popularity position, we attempted a more feasible project. Current research into automated genre detection given features extracted from music has shown more promising. Using a combination of K-Means clustering and Support Vector Machines, as well as a Random Forest, we produced two automated classifiers that performs five times better than chance for ten genres.

1. Data

Our main source of data was the Million Song Dataset Subset distributed by Labrosa. The subset provides pre-extracted features for 10000 songs including song and artist names, song duration, timbral data (MFCC-like features) and spectral data for the entire song, number of sections and average section length, and a number of other features.

In order to measure the popularity of a song, for each song we collected the number of view counts registered on the video returned as the first link in a YouTube search using the YouTube API, where the query consisted of the song and artist names. We checked by hand the accuracy of this scraping method and concluded that the two errors in thirty randomly drawn songs were unproblematic given that the errors also coordinate well with very low view counts (i.e. something unpopular enough to not return a copy of the song on youtube ends up returning an unrelated video with a low view count).

For Genre Classification, we used the Million Song Dataset Genre subset. The dataset includes features extracted from 59,600 songs divided into ten genres: classic pop and rock, folk, dance and electronica, jazz and blues, soul and reggae, punk, metal, classical, pop, and hip-hop. Features for each song include loudness, tempo, time signature, key, mode, duration, as well as average timbral data and average timbral variance.

2. Preliminary Analysis: Popularity Prediction

We first ran basic correlation coefficients between different parts of the metadata and also with our extracted youtube view counts. The results were largely insignificant and included weak correlations such as one of .2 between the tempo and loudness metadata features. Correlations between the youtube view counts and the echonest metadata features loudness, tempo, hotttness, and danceability were completely negligible (less than .05 in magnitude). The fact that these are not at all correlated is interesting in its own right because it points to no single metadata feature being at all a good predictor of views on youtube.

3. Popularity Prediction – Methodology and Results

3.1. **Linear Classification using Support Vector Machines.** First we note that the feature vectors given by the million song dataset are not of uniform length as the spectral and mfcc coefficients are provided for intervals of the song and thus the number of those features depends on the length of the song. To account for all of the data and compress it to uniform length for each song, we took averages of the coefficients for each half, fourth, or sixth of the song and concatenated the result for feature vectors of length 24, 48, or 72 for each set of coefficients, and then normalized the resulting lists of feature vectors. We trained Support Vector Machines with each extracted feature thinking that some number of segments would outperform others (perhaps as they become closer to the average of the actual number of segments in the songs). We altered the cost, bias, and kernel, but the precision never gained more than one percent on our bias. That is, 53% of the songs had youtube view counts over 10K and 19% were over 100K. The Support Vector Machines regardless of feature choice and parameters never achieved more than 53% and 81.5% precision. The recall was always less than 53.5% and 82%.

Finally, we tried a Support Vector Machine on spectral averages and metadata features including tempo, time signature, energy, loudness, duration, the number of sections, and finally the Echonest's popularity measure, 'hotttnesss'. Trained on popularity measure of 100K views, these features resulted in accuracy no better than before. However, when trained on the popularity measure of 10K views, the precision rose from under 53% to over 55%. Using the same features without 'hotttnesss' results in the same performance as above (no better than the bias of the dataset). From this we conclude that the Echonest's 'hotttnesss' measure (for which they have not released an explanation as to how it is quantified) gives a very small amount of predictive power (2-3% above the bias) on whether or not a song will have more than 10K views on Youtube.

3.2. **String Kernel.** Given that the mfcc and spectral data is temporal, we wanted to use the ordering therein to describe the sound. Motivating a string kernel Support Vector Machine approach, we create 'string' features for our songs as follows.

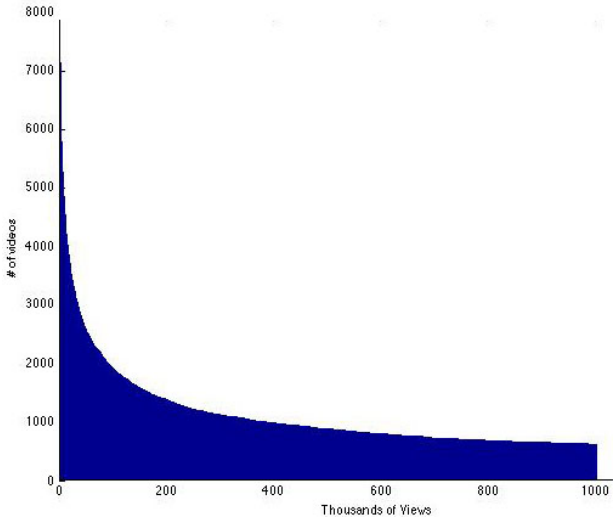
For each i, we take the spectral bucket i (corresponding to a frequency-aggregate magnitude) for each spectra vector within a range of each song (usually about 45 seconds in the middle). This gives a list of values which correspond to the magnitudes over time of this slice of the sound spectrum for every song. Next, using a subset of this data (we used two hundred of the ten thousand songs) we compute a list of intervals (we used 26 of them, corresponding to the characters a through z) that uniformly distribute the data. Then using these intervals, we compute a string for each sequence of data obtained in the first step by replacing each value with a symbol or letter that represents the interval.

When we tried a string kernel Support Vector Machine on this data, it was unable to complete even the first iteration towards convergence. Notably, string kernels are not guaranteed to be general Mercer kernels, and our string data was so long and varied that we suppose edit distance may be a very poor metric. Furthermore, edit distance is not aware of operations such as translation, which can help tell that two pieces of music are similar (consider putting a silent delay at the beginning of a song, then the method mentioned above will not recognize the songs as anything similar). For this reason, we conclude that this method of using a string kernel is a dead end and may only be helped by more general pattern matching metrics instead of the overly simplistic edit distance. Algorithms such as these, however, are their own area of research and fall under the categorization of

structural segmentation and similarity metrics (most often used for identifying song covers). We find the possibility of research on estimating popularity by structural segmentation to be interesting.

4. Popularity Prediction Discussion

Similarly to Pachet, we have concluded that from the audio features extracted there does not seem to be embedded the information relevant in making the song popular. This could be a result either of feature selection, or of popularity being driven by social forces, i.e. "the inherent unpredictability of cultural markets" (Pachet). The effect of cumulative advantage may help explain this phenomenon and is demonstrated in the curve (below) showing the declining percentage of view counts (e.g. there is very little difference between how many songs have a view count of 400K and 1M). Furthermore once an artist is popular, they may later produce works which are musically different from those that made them popular, yet the new tracks will become popular simply by virtue of being created by the popular artist.



5. Genre Classification

In light of our modest results at predicting popularity, we began to investigate another problem involving only musical features. Using the million song genre subset, we tried several classification algorithms including Support Vector Machines with ten-fold cross validation, k-nearest neighbors, and random forests. These were run on the entire data set, on a uniformly distributed subset, and also on a four-genre subset consisting of classical, metal, soul and reggae, and pop. The results reported are after running parameter selection on the Support Vector Machine as well as on k for the k-nearest neighbor implementation. We have read of implementations of KNN using KL-divergence as a distance metric (Mandel), but the million song genre subset did not have information sufficient to compute the covariance matrix of the timbral averages. Mirex hosts a number of papers on genre classification, however, an overview of the current literature suggests that Random Forests are rarely used for genre classification tasks.

In addition, Support Vector Machines were trained on individual pairs of genres, with n songs from each genre.

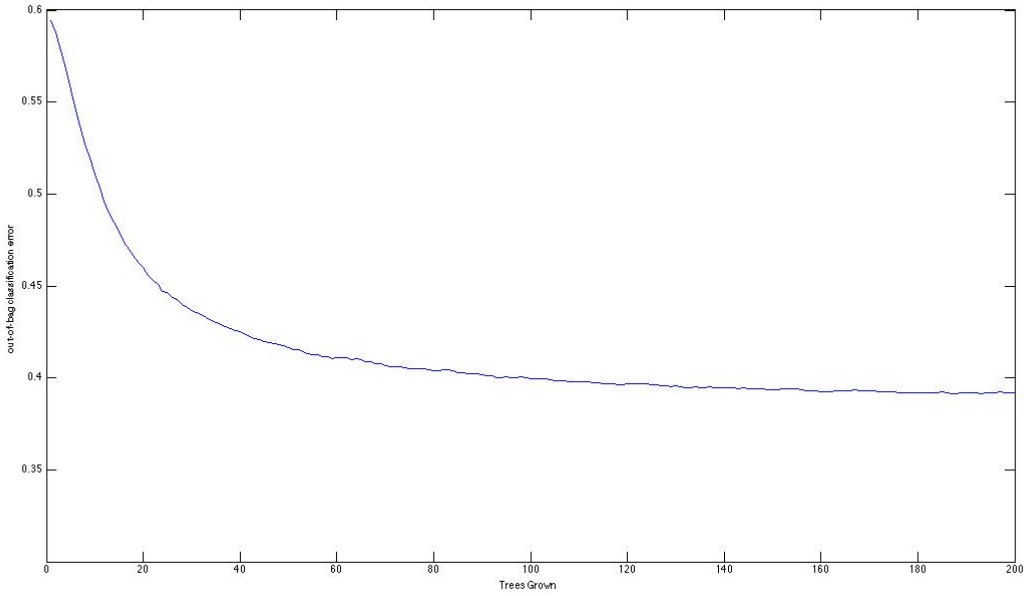
6. Genre Classification Results

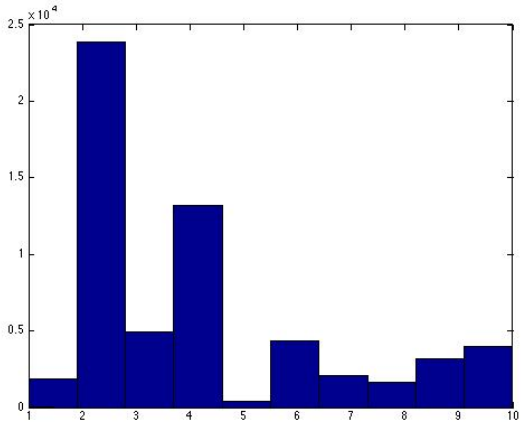
Results for various methods on different subsets are reported in the table below. Because the dataset was not uniformly populated, we also used the uniform dataset. The original distribution of songs in a given genre is shown below in the histogram. Average results over all four clusters are reported for cross-validation on the entries labeled K-Means + Support Vector Machine. In addition, classification results for pairs of genre are reported in the second table below.

Results for four genres are comparable to results seen elsewhere, but recent work on music classification suggests that it is possible to gain more accuracy on up to ten genres. [4].

Notably, in all three classification scenarios, Random Forests perform approximately as well as K-means in combination with Support Vector Machines. It is interesting to note that, upon adding more trees to our random

forest, the results seem to suggest that our Support Vector Machines after K-Means are both apprach the same accuracy. This can be seen in the following graph, representing the out of bag error of our Random Forest as more trees are added. This result is from the using the entire genre dataset.





Dataset	Method	Result
All 59600 Songs	SVM	49.09
All 59600 Songs	K-Means + SVM	54.15
All 59600 Songs	Random Forest	56.80
All 59600 Songs	10-Nearest Neighbors	43.84
Uniform Genre	SVM	47.60
Uniform Genre	K-Means + SVM	52.93
Uniform Genre	Random Forest	51.18
Uniform Genre	10-Nearest Neighbors	19.72
4 Genre	SVM	76.00
4 Genre	K-Means + SVM	83.35
4 Genre	Random Forest	83.19
4 Genre	10-Nearest Neighbors	74.58

Classical	Classic Rock	Electronica	Folk	Hip-hop	Jazz/Blues	Metal	Pop	Punk	Soul
Classical	83.05	87.07	83.05	92.04	83.99	93.47	91.86	91.30	91.59
Classic Rock	0	74.56	73.50	80.35	76.43	87.75	72.23	77.00	77.48
Electronica	0	0	81.03	77.16	79.25	89.25	79.79	84.45	79.54
Folk	0	0	0	87.61	76.75	92.17	78.64	86.68	82.27
Hip-hop	0	0	0	0	83.18	87.07	79.60	80.70	70.44
Jazz/Blues	0	0	0	0	0	91.62	83.02	87.74	82.74
Metal	0	0	0	0	0	0	91.01	81.09	94.36
Punk	0	0	0	0	0	0	0	82.07	77.60
Soul	0	0	0	0	0	0	0	0	84.40

7. Moving Forward

First we make a note on feature selection. In Exploring Billions of Audio Features [5], Pachet notes a distinction between the low- and high-level features used for many music information retrieval and/or classification tasks. He describes that many of the features so commonly used, in our case spectral coefficients and mfccs, are not adequate for many tasks and gives a general framework for exploring the enormous possibilities in exploring the feature space for musical signals. Relating this to our failure to provide results with respect to estimating popularity, it would seem that the task is not necessarily entirely hopeless and that our inconclusive findings (as well as Pachet's) do not demonstrate the inability to predict popularity at least somewhat from musical features (though the many studies of cultural dynamics would seem to indicate that there are other parameters at work, though possibly in conjunction with the musical qualities as well).

8. References

[1]C.-C. Chang, C.-J. Lin. Liblinear: a library for large linear classification Software available at http://www.csie.ntu.edu.tw/ cjlin/liblinear/ 2001.

[2] Mandel, M., & Ellis, D. (2005b). Song-level features and Support Vector Machines for music classification. Extended Abstract. MIREX 2005 genre classification contest (www.music-ir.org/evaluation/mirex-results).

[3] Pachet, F. & Roy, P. (2007). Exploring billions of audio features. In International Workshop on Content-Based Multimedia Indexing (CBMI), pp. 227235.

[4] Pachet, F., Roy, P.: Hit song science is not yet a science. In: 9th International Conference on Music Information Retrieval (ISMIR 2008) (2008)

[5] Thierry Bertin-Mahieux, Daniel P.W. Ellis, Brian Whitman, and Paul Lamere. The Million Song Dataset. In Proceedings of the 12th International Society for Music Information Retrieval Conference (ISMIR 2011), 2011.

[6] Mirex 2009 Audio Competition (`"http://www.music-ir.org/mirex/wiki/2009:Audio_Genre_Classification_%28Mixed_Set%29_Results"`)