Rage Against The Machine Learning: Predicting Song Popularity

Andreas "King-of-the-Keys" Garcia
Department of Computer Science
Email: andreas4@stanford.edu

Cody "C-Dawg" Kala
Department of Computer Science
Email: codykala@stanford.edu

Gabriel "Getaway Driver" Barajas
Department of Electrical Engineering
Email: gbarajas@stanford.edu

***Abstract*—What about a song affects its popularity? We sought answers to this question using Machine Learning (ML) techniques and the One Million Song Dataset.[4] We extracted hundreds of features from each song in the dataset, including metadata, audio analysis, string features, and common artist locations, and used various ML methods to determine which of these features were most important in determining song popularity. Linear regression with some polynomial features added to the data performed the best out of these methods, and we used the learned parameters to identify the most beneficial and harmful features to a song's popularity.**

I. Introduction

A. Motivation

Music is one of the most widely appreciated aspects of modern society, and popular musicians are some the most revered figures in the world. It is often very difficult to determine why a certain song is popular and other songs are not. We wish to analyze the popularity of music. In particular, we are interested in answering the question: what about a song affects its popularity? We feel that Machine Learning techniques including feature extraction and selection, regression, classification, and clustering are well suited to answering this question using data sets containing many songs. Determining a list of important features will then allow us to predict whether unobserved or new songs will be popular.

B. Related Work

Significant research has been done in this area, but many papers have focused on small, specific feature sets. Mussmann, Moore, and Coventry examined song lyrics and made modest improvements to their baseline song popularity predictors.[1] Koenigstein and Shavitt studied music piracy and found a strong correlation between popularity fluctuations of songs in the Billboard Magazine chart and fluctuations of illegal downloads of those songs.[2]

Pham, Kyauk, and Park focused on predicting song popularity in a previous CS229 project.[3] They used the One Million Song Dataset and extracted almost one thousand features many of which were string features associated with each song, but also included polynomial features and other metadata pulled directly from the dataset. They then narrowed down features using forward and backward selection, and evaluated their results using several classification and regression algorithms. They concluded that acoustic features (which pertained to the audio itself) were less important to song popularity than metadata such as genre labels or artist familiarity. Our project, though similar in end goal, ignores artist familiarity and focuses more on feature extraction, especially with regards to the acoustic features of each song.

II. Data Set

We used the Million Song Dataset (MSD).[4] Some of the many features included for each of the million songs were song key, tempo, average loudness, string terms associated with the song, and acoustic features related to the audio itself. In particular, we were interested in the feature, hotttnesss, which was a measure of song popularity determined by the music analysis website The Echonest. We would use this measure as a label to predict song popularity in our experiments. Due to the size and computational requirements of the entire dataset, we focused much of our analysis on a subset of 10,000 songs from the MSD. Many of the songs were missing information about features or hotttnesss, so after these songs were discarded, we were left with a workable subset of roughly 4,000 songs. Once we had experimented with different models and features, we used a larger workable subset of 14,000 songs to make final tests our models and features. The experiments and results in this paper used the larger dataset.

III. Feature Extraction

Much of our work centered on extracting suitable features from the dataset. Some of the out-of-the-box features could be used directly, but many needed to be processed before we could use them in our models (for example, string terms associated with each song could not be used directly as numeric values). We broke up our extracted features into four general categories:

A. Macro-level features (Count: 23)

These pertained to the song as a whole and included song key, time signature, loudness, and tempo. Some work was required to extract more meaningful numeric values than the out-of-the-box values. For example, song key was given from the MSD as a number from 1 to 12 (where 1 corresponded to the key of C, 2 to C#, etc.), and we felt that running regression on this number would not produce meaningful results, since the indexing was fairly arbitrary. Instead, we replaced this number with twelve indicator features, each taking on the value 1 if the given song was in the key

represented by that feature and zero otherwise. To make this measure more accurate, we also replaced the indicator value of the song key with the confidence value that the song key was correct (this confidence measure was included as a feature in the MSD and took on values between 0 and 1). We also used this procedure to extract time signature and mode information from each song. Each song had a measure of artist hotttnesss and artist familiarity, which obviously correlate with song hotttnesss. However, we did not use these features because they would obscure the influence of other features, not yield very meaningful results, and not be generalizable to music by unknown artists.

B. Micro-level features (Count: 177)

Each song in the MSD is split up into hundreds of "segments", which are typically under a second and relatively uniform in pitch and timbre. For each segment, timbre and pitch content are analyzed, yielding twelve numerical values for timbre and twelve for pitch. The pitch values correspond to the relative density of each of the twelve pitches and the timbre values correspond to the spectral density in each of the 12 spectral basis functions in Figure 1 [5]. We can interpret these values as general descriptions of how the segment sounds. For example, the first value corresponds to the average loudness, the second value corresponds to the flatness of the sound, and the third corresponds to brightness. Each segment also has values for max loudness, starting loudness, and offset of max loudness, which can be interpreted as attack time. These segment features are incredibly descriptive and rich in information about how a song sounds so we tried to extract as many useful features as we could from them using various statistics. For example, taking the average of the difference between subsequent timbre values gives a numerical measure of how quickly the timbre changes on average. From the pitch information, we extracted a rough approximation of the melody by looking at the argmax for each segment. From this, we computed statistics about melodic intervals and scale degrees.

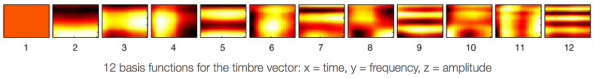

Fig. 1. Spectral basis functions for timbre features

C. Bag-of-words features (Count: 300)

Each song has an associated collection of around 20 "terms", which are words or short phrases commonly used to describe that song. We found the most common terms in a subset of the training set and made indicator features for each of the most common terms. Each term in each song also has an associated weight and frequency value, which are measures of how often that word is used to describe that song. We include the weight and frequency values as features as well as a simple indicator of whether or not a song has a certain term.

D. Location Features (Count: 40)

Location features: While the Bag-of-words features occasionally covered location information about a given songs artist, the MSD directly provided a more reliable location feature to determine where the artist was based. We searched through all the songs in our subset and found the forty most common locations. We then introduced forty features which indicated whether a song's artist was based in each location.

IV. Methods & Experiments

A. K-means

We ran the algorithm on the macro-level features as one of our preliminary experiments. Our goal was to determine whether the resulting centroids, if ordered by their hotttnesss, could give insight into which values of the macro-level features determined song popularity. The results are shown in Figure 2 for 5 centroids (we only show a handful of features here so that the table could fit in the report). The found centroids are listed in order of decreasing hotttnesss. Examining the features of these centroids, we see some correlations. For example, songs that are louder and whose song title length is shorter tend to be associated with higher hotttnesss. However, we found many of the other features did not show such correlations, and after observing the results for various numbers of centroids, we concluded K-means could not give us much more insight into song popularity.

Song Hotttness	Artist Name Length	Song Title Length	Song Duration	Loudness	Tempo
0.704	12.3536	17.2406	243.656	-7.4343	130.3511
0.5089	12.4088	18.4767	244.1543	-8.6097	103.5626
0.4726	13.6495	18.2828	229.7461	-8.556	170.8979
0.3556	14.0646	18.9211	227.5663	-19.6465	98.1065
0.2793	12.7951	19.5861	246.7856	-8.9479	122.2847

Fig. 2. Centroids from K-means on Macro-Level Features

B. PCA

To visualize our high dimensional micro-level features, we found the principal components of the data set with micro-level features and plotted the dataset projected onto various principal components. We colored the data by hotttnesss to look for trends. There seems to be some vague clustering of hotttnesss, but no simple correlation was observed, as we can see in Figure 3.

C. Regression

We tried several regression models to determine whether the features we extracted were useful in predicting song hotttnesss. To do this, we compared the mean squared error of each model to that of our baseline model. The baseline simply took the average hotttnesss value of the dataset and guessed that value for every song. We used regularization to tune the models. We focused on L2 regularization, but also experimented with L1 regularization. Both are discussed below.

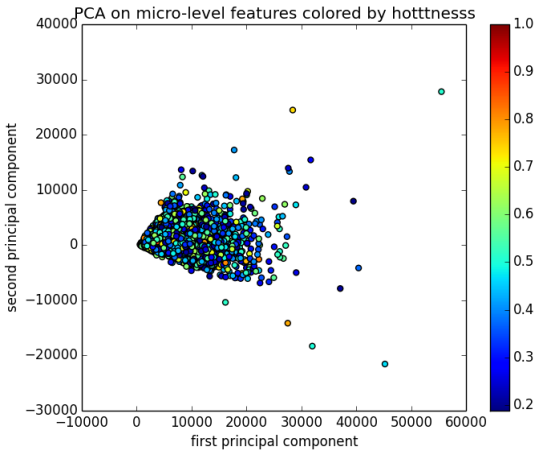

Fig. 3. Dataset projected onto first two principal components. The color scale refers to hotttnesss.

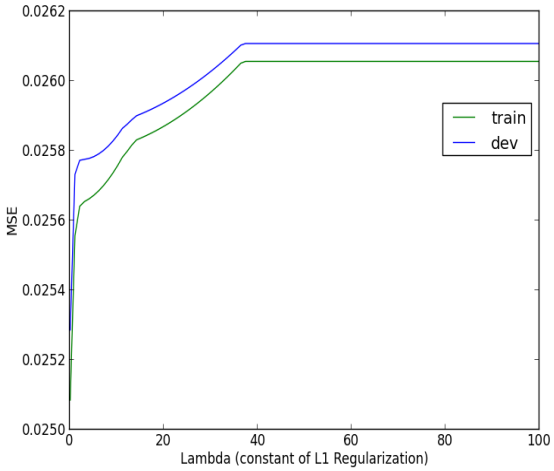

Fig. 4. MSE over different L1 Regularization constants

1) L2 Regularization: The normal mean squared error we would like to minimize is given by:

$$\frac{1}{m}\sum_{i=1}^{m}(y^{(i)} - h_\theta(x^{(i)}))^2$$

where $y^{(i)}$ and $h_\theta(x^{(i)})$ are, respectively, the true label of the ith training example and the hypothesis of this example, as parameterized by θ. We add a penalty $\lambda\|\theta\|_2$ to the above expression, where larger values of λ will increase the penalty for large parameters. In general, L2 regularization works against over-fitting. Most of the plots in this section use L2 regularization to improve predictions.

2) L1 Regularization: Like L2 regularization, L1 regularization adds a penalty to the mean squared error, but the penalty is instead $\lambda\|\theta\|_1$, the sum of the absolute values of the parameters θ. L1 regularization tends to produce sparse coefficients, so that in general many parameters disappear while only the most important ones persist.[6] Figure 4 shows the results of L1 regularization on all our features as λ increases. We see that the mean-squared error actually worsens for both training and development sets with stronger L1 regularization. As a result, we ruled out L1 regularization for further testing.

3) Linear Regression: With the goal of minimizing the mean squared error between true hotttnesss and our hypothesis, we implemented linear regression with L2 regularization. As opposed to some of our more complex models, linear regression did not have a problem with over-fitting, as we can see by the relatively small gap between training error and development error in Figure 5.

4) Polynomial Regression: In order to capture the non-linear structure of the data, we implemented polynomial regression. Simply taking all polynomial combinations of degree 2 of our 540 features would be impractical, as many of the resulting features would always be zero, such as $(key = A)(key = D)$, and more importantly, the number of features would surpass the size of the dataset, meaning that the parameters would fit exactly to the test set in an extreme case of over fitting. We tried quadratic regression, simply adding the square of each feature, but this caused slight over fitting. As shown in the plots for qr in Figure 5, the gap between train and development error is wider than that for linear regression. We also added to the feature set degree 2 polynomials involving the indicator features for the most common terms. With enough regularization, this resulted in the lowest development error out of all the regression models, as seen in the plots for pr in Figure 5.

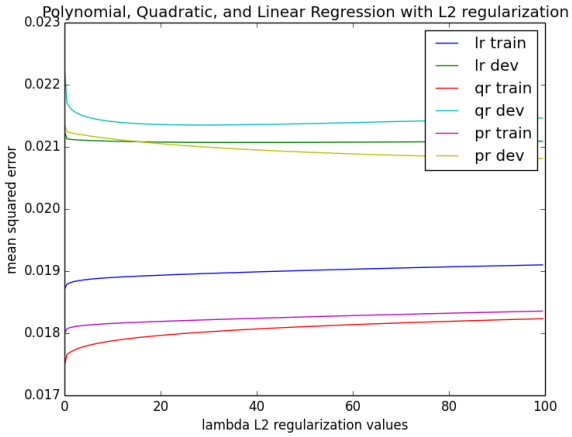

Fig. 5. Regression Models with L2 Regularization Error

5) Locally Weighted Linear Regression: We implemented locally weighted linear regression with the hopes that we could capture more local trends. The weights were given by $w^{(i)} = \exp\frac{-||x^{(i)}-x||_2^2}{2\tau^2}$. As we increased τ, the development error decreased monotonically due to less over-fitting. As τ increases, examples are weighted more evenly and we approach linear regression. Since our locally weighted linear regression mean-squared error was always higher than that of linear regression, we concluded the former model was inferior.

D. Classification

Another paradigm we considered for predicting song popularity using our extracted features was classification. We separated the data into ”hot” and ”not hot.” To do this,

we gave label 1 to songs whose hotttnesss was a standard deviation or more above the average hotttnesss across the dataset, and label 0 to the rest. Figure 6 shows the accuracy, precision, and recall results for training and development sets. Precision is the fraction of songs predicted to have label 1 that actually have true label 1, while recall is the fraction of songs with true label 1 that were predicted to have label 1. We see that accuracy for both training and development sets was good compared to a random guessing of the labels, but precision and recall were poor, meaning songs with true label 1 were not classified well.

Boosting: We tried using a Boosting algorithm to improve our classifiers. Essentially, this algorithm takes in a base classifier (e.g. Logistic Regression) and iteratively combines weighted copies of it, emphasizing data that was previously classified incorrectly. Not all the classifiers above were able to be boosted, but Figure 7 shows the results for algorithms that made the cut, after 50 iterations of boosting. We see that performance is slightly better than in the non-boosted case, but precision and recall are still poor. Ultimately, we concluded that the features we extracted do not lend themselves well to predicting hotttnesss classes.

		Logistic Regression	SVM	Random Forest	K Nearest Neighbors
Train	Accuracy	0.827	0.827	1	0.827
	Precision	0.333	0	1	0.7
	Recall	0.001	0	1	0.004
Development	Accuracy	0.825	0.825	0.834	0.825
	Precision	0	0	0.63	0
	Recall	0	0	0.124	0

Fig. 6. Performance of Several Classifiers

		Decision Tree	Logistic Regression	Random Forest
Train	Accuracy	0.847	0.827	1
	Precision	0.647	0.571	1
	Recall	0.256	0.005	1
Development	Accuracy	0.836	0.825	0.831
	Precision	0.573	0.25	0.615
	Recall	0.239	0.002	0.09

Fig. 7. Performance of Boosted Classifiers

V. Results & Analysis

Here we discuss the features that were most significant in predicting song popularity. We note from Figure 5 that linear regression with polynomial term features added performed slightly better than the other plotted regressions with enough L2 regularization. As a result, we used the parameters learned from this regression to determine which features (including polynomial terms) had highest significance. If the parameter associated with a feature had a high positive value, it meant the feature had a strong positive correlation with hotttnesss; whereas if the parameter had a high negative value, it meant the feature had a strong negative correlation with hotttnesss.

A. Regression Performance for Different Feature sets

Running regression on each of the subsets of features individually, we compare the significance of these subsets. As we can see in Figure 8, all of these subsets yielded significantly lower mean squared error than our baseline model of guessing the mean hotttnesss. For each subset, we used an L2 regularization constant of 50. The feature sets that were most successful were the terms features, and the polynomial features generated from the indicator variables for the top 50 terms.

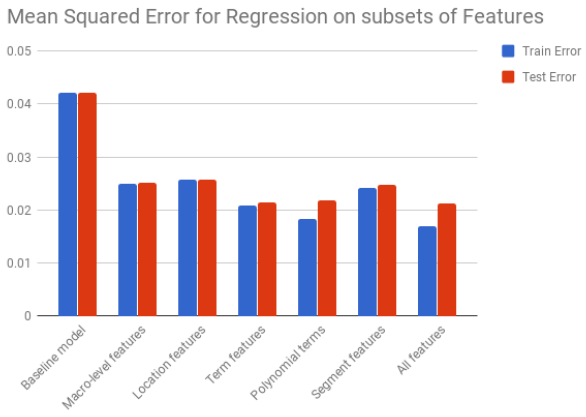

Fig. 8. Performance Comparison of Feature Subsets

B. Macro-level results

Positive	key is C	key is D	major mode	key is G	loudness
Negative	key is A	key is B	key is G#	key is D#	artist name length

Fig. 9. Most positive and negatively weighted macro-level features

As we see in figure 9, which lists the most positive and negative features in order of decreasing magnitude, some of the most significant macro-level features were the confidence features for key. We believe that the correlation between key and popularity is due to the fact that popular artists tend to write music in simple keys, rather than to any perceptual differences between the keys.

C. Location results

Positive	Atlanta	Los Angeles	Memphis	Boston
Negative	London	New York City	Philadelphia	Ontario

Fig. 10. Most positive and negatively weighted artist locations

Of the feature subsets we analyzed, artist location turned out to be the least indicative of song popularity, but there are certainly strong trends that we can see. According to the weights assigned to these features, the best and worst places to make popular music are listed in figure 10.

D. Bag-of-words (terms) results

Positive	guitar	acoustic	hardcore	instrumental	soundtrack
Negative	r&b	alternative rock	world	american	house

Fig. 11. Most influential terms

The term features ended up being the best indication of popularity. We expect this is because these are directly related to how the song is viewed in the public eye. We found some interesting and somewhat unexpected results from looking at the most positive and negatively weighted indicator features

for the top 100 terms as we can see in figure 11. We also wanted to see what combinations of terms might contribute to song popularity or unpopularity so we regressed on all degree 2 polynomials of indicator features for the top 50 terms and we show the results in figure 12. Note that by itself, "metal" is positively correlated with popularity, but when combined with "hip hop", it is negative.

Positive	metal	indie rock new wave	pop rock american	singer-songwriter female vocalist
Negative	american soundtrack	germany guitar	hip hop metal	acoustic hard rock

Fig. 12. Most influential term combinations

E. Micro-Level results

We will describe the four most positively correlated features in order from highest to lowest. Mean bar confidence is a measure of how distinguishable the bars of the song are. Frequency of melodic interval of zero is a measure of how often the approximated melody stays on one note. Mean of difference in timbre scale 11 (see figure 1) measures how quickly the "heaviness" of the song changes. Now we describe most negatively correlated features. The average difference of sums of pitch weights between subsequent segments can be interpreted as a measure of how quickly the harmonic complexity changes. The frequency of the minor second scale degree has negative correlation with popularity because this scale degree is rarely used and therefore sounds strange to most listeners.

F. The Entire Feature Set

We list the best and worst features across the feature set in figure 13. To intuitively understand what depth of information

Positive	indie rock new wave	mean bar confidence	acoustic guitar	singer-songwriter female vocalist
Negative	american soundtrack	indie reggae	New York City	acoustic hard rock

Fig. 13. Most influential features across whole feature set

is captured by the feature set, we implemented a program that, given a list of songs, uses distances between feature vectors to find which song is most similar to "Kick Out the Jams" by Rage Against The Machine. While we can not quantify the success of this function, we were satisfied with the results. It was particularly successful at capturing similarity in melodic and harmonic structure.

VI. Future Work

While we are proud of the results above, we feel we could make some improvements in future work. In particular, we may obtain better performance from our classifiers if we use more than two classes of hotttnesss. We would also like to expand our feature set to include song lyrics, which were unavailable in the data set that we used. There are other external features we would like to analyze such as radio airtime and amount of money invested in promotion.

VII. Conclusions

Our project revealed several unexpected results about what makes music popular. The fact that artist location played a strong role in determining popularity indicates that popularity is strongly tied to features unrelated to the sound of a song. Using features that mostly pertained to the content and descriptions of songs themselves, we were unable to decrease our test error below a certain threshold, indicating either that song hotttnesss is a noisy measurement, or that song popularity is based on many factors that are outside of our feature space. This can be explained by the fact that a song's popularity is largely driven by marketing, which is unrelated to song content. The trends that we revealed in our research are of great interest to record labels, radio stations, and music streaming websites who want to maximize the popularity of their products. While record labels and promoters will undoubtedly benefit from the trends revealed by machine learning methods, it is up to the future to decide how this will ultimately impact human creativity and musical exploration.

VIII. Contributions

We all aided each other in all parts of the project. Below we summarize the work each of us focused on, but we were not limited to these tasks.

A. Andreas Garcia

Extracted micro-level features and bag-of-words features. Implemented linear, polynomial, and locally weighted regression. Tested regularization parameters and implemented PCA.

B. Barajas, Gabriel

Extracted the macro-level features. Implemented some of the experiments, including K-means, Linear Regression with L1 regularization, and the Boosted classifiers.

C. Cody Kala

Helped process bag-of-words features. Implemented non-boosted classifiers and metrics for the classifiers. Cleaned data and combined extracted feature sets.

References

[1] Mussmann, Stephen, et al. Using Machine Learning Principles to Understand Song Popularity. Cs.purdue.edu, www.cs.purdue.edu/homes/moore269/docs/music.pdf.

[2] Koenigstein, Noam, and Yuval Shavitt. SONG RANKING BASED ON PIRACY IN PEER-TO-PEER NETWORKS. Http://Www.eng.tau.ac.il, www.eng.tau.ac.il/ shavitt/pub/ISMIR09.pdf.

[3] Pham, James, et al. Predicting Song Popularity. Cs.stanford.edu, cs229.stanford.edu/proj2015/140_report.pdf.

[4] 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.

[5] Tristan, Jehan. Analyzer Documentation . The Echo Nest, docs.echonest.com.s3-website-us-east-1.amazonaws.com/_static/AnalyzeDocumentation.pdf.

[6] Differences between L1 and L2 as Loss Function and Regularization. Http://Www.chioka.in, 30 Nov. 2014, www.chioka.in/differences-between-l1-and-l2-as-loss-function-and-regularization/.