Predicting Hit Songs Using Repeated Chorus

by Atom Singh

Submission date: 22-May-2022 09:29PM (UTC-0500)
Submission ID: 1842100424
File name: Predicting_Hit_Songs_G14.pdf (543.06K)
Word count: 2254
Character count: 11939

Predicting Hit Songs Using Repeated Chorus

Authors:
Ansh Jain
Naresh Kumawat
Neeraj Singhal

Created: May 20, 2022

Project mentors:
Dr. Bharavi Mishra

Courses:
Machine Learning

Project category:
Audio & Music

ABSTRACT

Songwriters and producers have long felt that the ability to compose a catchy hook is what makes a song popular. We investigated this myth by compiling a database of successful artists' choruses and used eight supervised Machine Learning approaches to predict popularity only based on audio characteristics extracted from a 15-second chorus. We were able to demonstrate superior model performance than random using a K-nearest neighbour (KNN). As a result, we find that the story is plausible.

1. INTRODUCTION

A musical hook is normally a section of a song that captures the attention of the listener. It might be a riff or a single sound in a song, but it's usually the opening few lines of a song's recurring chorus. Making a hook is a common songwriting method. The most memorable hooks will stay with you for days. Songwriters and producers often assume that a catchy hook is that makes a song famous.

We debunked this myth by compiling a dataset of prominent singers' choruses and using supervised machine learning (ML) algorithms to predict song popularity solely based on the chorus's acoustic qualities.

2. RELATED WORK

We discovered several efforts to forecast song popularity using machine learning approaches throughout our literature study. The quantity of metadata you need to make predictions provides you an approximate idea of how much literature there is. This is where song data, such as audio and lyric qualities and metadata, as well as non-data aspects like album and artist information, are defined.

In this sense, Dhanaraj R. et al. are song data purists. Using probabilistic latent semantic analysis to extract Mel frequency cepstral coefficients (MFCC) with audio data and embedded lyric information in early 2005 , followed by boost and linear support vector machine (SVM) To make predictions, I utilised a tree. No music is allowed to chart. In a similar line, LiChia and et al. 2017 paper [6] published a new investigation. Convolutional Neural Networks (CNNs) are used to rate the popularity of Chinese and Western music, however MFCC characteristics are also employed.

On the other hand, a few earlier efforts have only relied on information to forecast popularity. A famous example is Eva and et al2016 .'s paper[10], in which they attempted to predict a popular song using information obtained from Twitter hashtags.

Many others used a hybrid method that incorporated data and metadata information. Pachet and Roy, for example, gathered 49 audio parameters, such as spectral characteristics, temporal characteristics, and harmonic characteristics, as well as a number of metadata aspects. This study employed Radial Basis Function SVM for modelling and concluded that forecasting song popularity is "still not a science." Alternatively, Kai et al. published a paper in 2019 as part of this approach. It employed five machine learning algorithms and 27 Spotify variables to forecast if a song will chart on Billboard.

This work expands the limitations of this spectrum, where just the sound qualities of an audio piece (chorus) are employed for prediction, to completely investigate the myth.

3. DATASET AND FEATURES

There are no publicly available music bundle data sets that we are aware of. To prepare your data and features, create a data pipeline that looks something like this:
1. Using billboard.py, get the titles of popular and unpopular songs by the same artists from Billboard.com.
[11]
2. Use youtube-dl to download the entire tracks from Youtube. [12]
3. Using pychorus [2], extract the repeating chorus.
[13]
4. Using librosa, extract the audio characteristics of the hooks.
[14]

3.1 Definition of popular and unpopular Songs

We started by compiling a list of popular songs and singers from the website named Billboard Hot 100 chart. [15] Billboard Magazine publishes a weekly list of music industry records in the United States. Billboard also provides monthly and yearly charts based on the weekly charts' weighted performance [16].

We classify the tracks on the chart as popular and assign them a 1 in our study. It also looks at an artist's full career and assigns disliked or zero ratings to other works.

3.2 chorus and Audio feature Engineering

Using youtubedl, get complete audio files from Youtube for prominent artists and popular and unpopular songs. Choose the first released version of the studio album for tracks that have various versions, such as remakes or remixes.

Then I extract 15 seconds using pychorus.
A song's chorus that is repeated on each track. Pychorus' main goal is to extract comparable structures from the frequency spectrum. This is effectively unsupervised learning, and the efficiency of the extraction may be influenced by how it is interpreted. However, we assume that these are the original truth in this work. We experimented with a few well-known songs and found that the chorus we retrieved closely matched our impression.

The chorus' acoustic qualities were extracted using librosa. The investigation was limited to eleven major spectral characteristics. Table 1 lists the measures by name, description, and number.

The size of each track's feature set is a function of t, where t is a function of the chorus duration (15 seconds in this example) and the soundtrack sampling rate i. We turned our raw observations into seven statistics: minimum, mean, median, maximum, standard deviation, skewness, and kurtosis, because we didn't want time or sample rate to alter our forecasts. After conversion, each song track generates a total of 518 sound characteristics.

Table 1 Audio features details

Major features	Descriptions	Raw Dims	Transformed Dims
chroma_stft	Compute a chromagram from a waveform or power spectrogram.	$12 \times t_i$	84
chroma_cqt	Constant-Q chromagram	$12 \times t_i$	84
chroma_cens	Computes the chroma variant "Chroma Energy Normalized" (CENS)	$12 \times t_i$	84
mfcc	Mel-frequency cepstral coefficients (MFCCs)	$20 \times t_i$	140
rms	Compute root-mean-square (RMS) value for each frame, either from the audio samples y or from a spectrogram S.	$1 \times t_i$	7
spectral_centroid	Compute the spectral centroid.	$1 \times t_i$	7
spectral_bandwidth	Compute p th-order spectral bandwidth.	$1 \times t_i$	7
spectral_contrast	Compute spectral contrast	$7 \times t_i$	49
spectral_rolloff	Compute roll-off frequency	$1 \times t_i$	7
tonnetz	Computes the tonal centroid features (tonnetz)	$6 \times t_i$	42
zero_crossing_rate	Compute the zero-crossing rate of an audio time series.	$1 \times t_i$	7
Total		$74 \times t_i$	**518**

3. The final project dataset

As we learn that accessing data features takes a lot Of time

So, to save time and work while collecting data, we build a tiny dataset which represent a proof-of-concept for our executive project report.

We chose the top 7 artists with the most hits on the HOT100 Quarterly Chart from 2006 through 2020 for this data set. Other songs from the Unpopular Songs collection

Same album, same artist. As a result, this final report's dataset contains a 554 518 audio feature matrix and a 554 1 response vector.

4. METHODS

4.1 Data exploration and dimension reduction

We utilised a pair scatter plot and Principal Component Analysis to investigate the ultra-high dimensional data in order to develop intuition and lower the dimensions (PCA)

Scatter Plot in Pairs Figure 1 presents a pair plot of all significant audio characteristics' median values. The signal to noise ratio is relatively high, as the graph shows, since there is no apparent demarcation between any pair of audio characteristics and response classes. Furthermore, some pairings show linear correlation, indicating that PCA might be used to lower the dimensions in the following part.

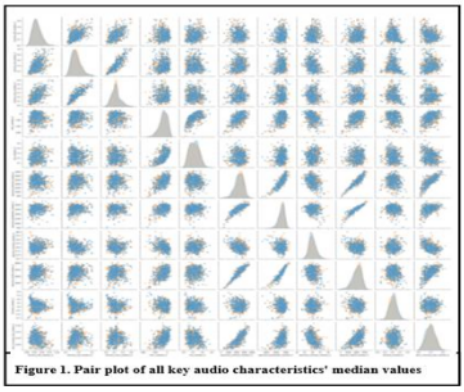


Figure 1. Pair plot of all key audio characteristics' median values

PCA Figure 2 depicts the cumulative explained variance of PCA to help understand its intrinsic dimensionality. To explain 95% of the variation, 170 Principal Components (PC) are required. Given the vast number of features and restricted data size, we choose to use the PCA projection to minimise dimensionality first..

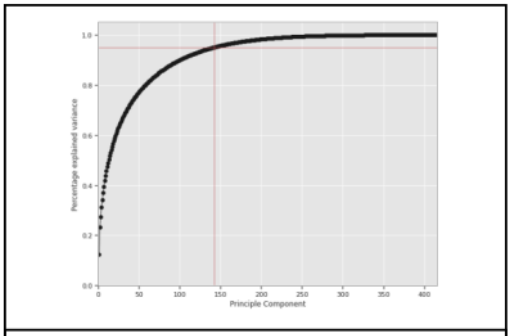
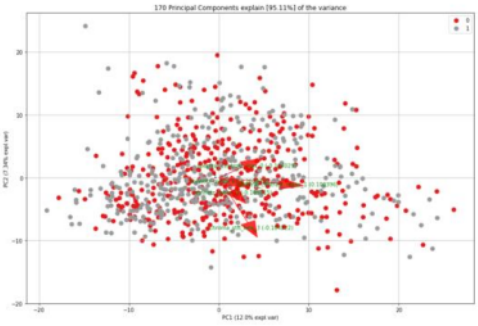


Figure 2. cumulative explained variance of PCA.

The graph of biplot of the two PCs is shown in Figure 3. Within the first two PCs, there is no evident grouping, similar to the preceding pair plot. We also looked at additional PCs and found that none of them were visibly clustered, confirming the difficulty of the situation.



Decision trees: The branches of a decision tree are created in a hierarchical manner, with each branch acting as an ifelse expression. Subdividing a data collection based on its most relevant qualities creates branching. The decision tree's leaves are where the final categorization happens.

4.2 Algorithms and models

This project exemplifies the trade-off between bias and

variation. On the one hand, in contrast to the number of characteristics, the number of observations is quite small. However, on the other side, the

Without a defined categorization boundary, as we stated in the preceding section, the situation becomes extremely hard. To minimise overfitting, the main goal is to apply regularisation in combination with ML models. For all candidate learning algorithms, we employed 5-fold Cross Validation (CV) to fine tune the tuning parameters from the regularisation condition of the patterns.

We employed six distinct machine learning algorithms in our supervised classification setup: Random Forest (RF), Linear Discriminant Analysis (LDA), SVM, Gradient Boosting Machine (GBM), Logistics Regression (LR), and NN.

Elastic Net with LR LR is a linear model that models a binary dependent variable using a logistic function. Elastic Net is a regularisation method that combines the L1 and L2 penalties linearly to produce the hyperparameter, which indicates the L1 penalty ratio.

$$\sum_{i=1}^{p}((1-\alpha)\beta_j^2 + \alpha|\beta_j|)$$

With shrinking, LDA **LDA** is a linear model as well, but it makes more random assumptions about the underlying densities. The shrinkage approach reduces the LDA's distinct covariance to scalar covariance, introducing the hyperparameter, which represents the depletion factor.

$$\hat{\Sigma}(\alpha) = \hat{\Sigma}(\alpha) + (1-\alpha)\hat{\Sigma}\hat{\sigma}^2 I$$

Kernel tricks with SVM SVM is a generalised classification approach for determining the best hyperplane separation. By generating a linear boundary in a big converted level of the feature space, the kernel trick aids in the creation of nonlinear boundaries. We employed three distinct types of kernels in this study: linear, semilinear, and nonlinear.ear

kernel, Radial Basis Function (RBF) kernel, and polynomial kernel. We also used elastic net

The pattern's involvement is controlled through regularisation.

RF In addition to regularisation terms, we looked at a few ensemble learning algorithms to overcome the model's efficiency and remove overfitting. Ensemble learning brings together a group of struggling pupils, who are then combined to form a composite predictor. RF uses de-correlated decision trees and projections based on the average of the committees as a weak learner.

GBM Boosting can also be used for aggregate learning technique. Boosting, unlike Random Forest, involves forming a group of weak learners who cast weighted votes to make predictions.

With regularisation, NN The NN model is perhaps the most widely used in current machine learning. Its main principle is to get linear combinations of the inputs as new features, which are then sent to the following layer as a nonlinear function of these features. A few regularisation strategies for NN have been developed throughout time. Weight decay is the most explicit way, which involves adding a penalty to the error function.The shrinkage factor is represented by the hyperparameter.

$$R(\theta) + \lambda\Big(\sum_{k,m}\beta_{km}^2 + \sum_{k,l}\alpha_{ml}^2\Big)$$

5. EVALUATIONS AND DISCUSSIONS

5.1 Results

We used a 72/25 split to divide the data into a training set and a test set for assessment. Because no unique application exists, four metrics (f1, precision, accuracy, recall) were examined for CV during hyperparameter adjustment during training, as well as for model comparison period testing and model comparison in the next section. Typically, any evaluation function is specified as

$$accuracy = \frac{TP+TN}{TP+TN+FP+FN}$$

$$precision = \frac{TP}{TP+FP}$$

$$recall = \frac{TP}{TP+FN}$$

$$Fscore = \frac{2}{recall^{-1} + precision^{-1}}$$

where TP =True positive; FP =False positive;

FN = False negative; TN= True negative

Despite our best efforts, all models underperform our expectations. In a close call, the best model for this job is NN. On the test set, the model's accuracy, precision, and recall are all greater than 50%, indicating that it outperforms a random guess.

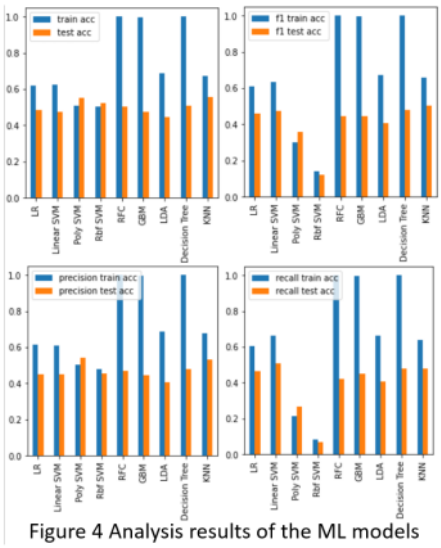


Figure 4 Analysis results of the ML models

All other models produce similar test set scores as the 5-fold CV, demonstrating that there is no overfitting. Despite the scant data, our focus regularisation technique succeeded.

	name	train acc	test acc	f1 train acc	f1 test acc	precision train acc	precision test acc	recall train acc	recall test acc
0	LR	0.618333	0.483444	0.609881	0.458333	0.613014	0.452055	0.606780	0.464789
1	Linear SVM	0.626667	0.476821	0.636364	0.476821	0.610592	0.450000	0.664407	0.507042
2	Poly SVM	0.510000	0.549669	0.300000	0.358491	0.504000	0.542857	0.213559	0.267606
3	Rbf SVM	0.505000	0.523179	0.139130	0.121951	0.480000	0.454545	0.081356	0.070423
4	RFC	1.000000	0.503311	1.000000	0.444444	1.000000	0.468750	1.000000	0.422535
5	GBM	0.996667	0.476821	0.996610	0.447552	0.996610	0.444444	0.996610	0.450704
6	LDA	0.685000	0.443709	0.674699	0.408451	0.685315	0.408451	0.664407	0.408451
7	Decision Tree	1.000000	0.509934	1.000000	0.478873	1.000000	0.478873	1.000000	0.478873
8	KNN	0.673333	0.556291	0.657343	0.503704	0.678700	0.531250	0.637288	0.478873

1

To overcome the problem of overfitting we can use ***K- Fold Cross-Validation.***

In this K-Fold cross-validation method, the entire data set is split into K pieces of equal size. Each section is called a "fold". Since there are K parts, it is called K-Fold. 1 fold is used for the validation set and the remaining K-1 folds are used for the training set.

This cycle is continued K times until every fold serves as the test set and the remaining folds serve as the data set.

5.2 K nearest neighbor deep dive and error analysis

We tuned all of the models extensively, but owing to the page constraint, we'll just go through the most successful approach, kNN, as an example.
The kNN was implemented using Sklearn. We used Sklearn since it outperforms TensorFlow and PyTorch in terms of performance and simplicity of use.
We used permutation tests to analyse the feature importance test and develop intuition for error analysis. The weights of the top one most important are relatively low, which corresponds to our earlier observations.

Importance	feature	weight
1	spectral_centroid_max_0	0.132 +/- 0.019
2	spectral_centroid_median_0	0.120 +/- 0.018
3	spectral_bandwidth_max_0	0.048 +/- 0.014

6. CONCLUSION

Because the model gives better results than rhyme, we may say that the belief that a good refrain makes a song famous is realistic. However, given the poor performance ratings in all models, we think that the song's success is related to far more external influences.

Dataset size and model complexity are still constrained by today's standards due to time and resource restrictions. As a result, we want to produce additional data in the future and investigate ML models.

7. REFERENCE

[1] Goto, M. (2006). 1783-1794 in IEEE Transactions on Audio, Speech, and Language Processing. The technique of a hook section detection method for musical audio signals to a music listening station.

[2] K. Middlebrook and K. Sheik (2019). Song Hit Prediction: Using Spotify Data to Predict Billboard Hits abs/1908.08609 on arXiv.

[3] S. Dieleman and B. Schrauwen (2013). Music Audio Feature Learning Using Multiscale Approaches

[4] R. Dhanaraj and B. Logan, ISMIR (2005). Automatic Hit Song Prediction ISMIR.

[5] L. Yang, S. Chou, J. Liu, Y. Yang, and Y. Chen (2017). Using convolutional neural networks to revisit the challenge of audio-based popular song prediction. 621-625 at the 2017 IEEE International Conference on Acoustics, Speech, and Signal Processing.

[6] X. Bresson, M. Defferrard, K. Benzi, P. Vandergheynst, and M. Defferrard (2017). FMA: A Music Analysis Dataset abs/1612.01840 on arXiv.

Predicting Hit Songs Using Repeated Chorus

ORIGINALITY REPORT

4%	4%	3%	1%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	www.analyticsvidhya.com Internet Source	3%
2	Chin-Chuan Chang, Chien-Hua Chen, Jer-Guang Hsieh, Jyh-Horng Jeng. "Iterated Cross Validation Method for Prediction of Survival in Diffuse Large B-cell Lymphoma for Small Size Dataset", Research Square Platform LLC, 2022 Publication	1%
3	jwcn-eurasipjournals.springeropen.com Internet Source	1%
4	Learn.G2.Com Internet Source	<1%

Exclude quotes	On	Exclude matches	Off
Exclude bibliography	On		