International Journal of Scientific Research & Engineering Trends
Volume 8, Issue 2, Mar-Apr-2022, ISSN (Online): 2395-566X

Predicting the Song Popularity Using Machine Learning Algorithm

Yasmin Essa, Adnan Usman, Tejasvi Garg, Murari Kumar Singh
School of Engineering
Sharda University,
Greater Noida, India
2018001544.yasmin@ug.sharda.ac.in, 2018009696.adnan@ug.sharda.ac.in, 2018010884.tejaswi@ug.sharda.ac.in,
murari.kumar@sharda.ac.in

Abstract-Being ready to predict popularity of a song supported metadata and attributes are often of great industrial importance. We aim to attain this using machine learning techniques. We use data obtained from Spotify Web API which contains information of over 160,000 songs from 1930 to 2021. We perform the desired pre-processing to check several regressions and classification algorithms supported obtained results; we build ensemble learning models for classification. Models are tuned to present optimal test results. Due to the imbalanced classification, the models are able to predict non-popular songs more easily than popular ones, where there are a high number of false negatives.

Keywords- Machine learning, music, popularity, prediction, songs, regression, classification, ensemble learning, random forests, boosting.

I. INTRODUCTION

Hundreds of songs are released per annum, but only a few of them make it to the highest charts. Music analysis has made it possible to get metadata of a song. Similarly, such features are available for artists and genres.

We aim to answer the question - "Is it possible to predict the popularity of a song using these attributes in Machine Learning algorithms?" Together with that, we learn more about the features involved, how important (or unimportant) they're and what are the possible limitations.

Our key contributions are the subsequent -

- We explore an acceptable dataset which contains the requisite features and tune it to our needs.
- We perform Exploratory Data Analysis (EDA) on the information.
- We apply several machine learning algorithms to predict the popularity of a song. We use regression, classification and ensemble learning algorithms. Inferences are gained at every step and further work is determined to support that.
- The model hyperparameters are tuned to fetch the simplest possible results. Different models arecompared.

Songs released by popular singers tend to bechart-toppers, however what about the occasional tuneputout via a lesser regarded artist that turns into a break hitapparently at random?

My goal is to create a system of getting to know a model thatmaypredict a tune's reputation.

To do this, I'll examine features like danceability, strength, or even speechiness.music has always been an essential part of my life. I'm able tonevertheless remember the primary time I heard the bandPhish inhigh faculty, and from there something 'clicked', and my musical tastes branched out broadly from just alt-rock to jazz, funk, progressive rock, bluegrass, and plenty of differentgenres.

But, my non-public musical tastes regularly differ pretty drastically from what's famous within the mainstream. As such, I've constantly been inquisitive about why positive songs are popular. I.e., what is it about certain songs the reason for them to have billions of listens?

II. LITERATURE REVIEWS

The ability to predict whether a song is going to be successful or not is of economic importance, because of which there has been a keen interest in using Machine Learning techniques in predicting the popularity of a song. This can be often called Hit Song Science, a term coined by Mike McCready. There are several papers that quantify certain lyrical and acoustical characteristics of a song and use those to predict popularity.

An early attempt inusing such features is by B. Logan et. al [1]. Another paper by R. Nijkamp [2] uses the identical Spotify dataset that this project aims to use and uses the attributes provided by Spotify as features to make a regression model.

There have also been attempts to predict song popularity from just lyrical properties. One such attempt is by A. Shinghi et. al [3] where they use rhyme, syllable and meter

features to predict popularity. They found that lyrical features worked better than audio features in popularity predictions. Another approach to hit song prediction is to ignore intrinsic characteristics of a song altogether (lyrical or acoustical) and use social media metrics of the artist or buzz created by users. Such attempts were made as early as 2008 by Bischoff et. al[4].

In a very similar vein E. Zangerle et. al investigates correlations between twitter hashtag #nowplaying and billboard top 100 charts [5]. While Machine Learning algorithms are used in many fields for prediction problems, using it to predict the popularity of a song is a problem that has been addressedin many previous attempts. Several amounts of research have been conducted onsong popularity prediction, which is resulting in different conclusions.

Most of these researches are based on the Million Song Dataset (MSD), which is an open dataset provided by the Echo Nest API (Currently acquired by Spotify).The MSD is an attempt to help researchers by providing a large-scale dataset [4].

In 2008, an endeavor to validate the hypothesis that the popularity of songscan be predicted from acoustic or human features which were conducted by Pachetand Roy. Their experiment was based on a dataset of 32000 titles and 632 features, which was a considerably huge dataset at that time. However, they were not ableto develop a good classification model and concluded that the popularity of a songcould not be predicted by using state-of-the-art machine learning techniques [6].

In a 2011 study made by Ni et al., provided with a positive result on the problemof predicting music popularity. Their goal was to distinguish the top 5 hits from the top 30-40 hit list. Their dataset was based on UK charts during a time period of 50years. 5947 unique songs were collected from the Official Charts Company (OCC),and the audio features were extracted from The Echo Nest. They indicated that itis possible to identify music hits [7].

Another study conducted in the same year of 2011, by Borg and Hokkaneninvestigated if they could predict the popularity of a song based on its audio featuresand Youtube view counts. The features for audio tracks were obtained from TheEcho Nest. For this task, several Support Vector Machines (SVM) were built and achieved a very modest result. The SVMs, regardless of feature choice andparameters, never achieved more than 53% accuracy. They concluded that audiofeatures alone do not seem to be good predictors of what makes a song popular.They suggest that popularity is likely driven by social forces [8].

Fan and Casey at Dartmouth College compared the prediction of UK hitsongs against Chinese hit songs. Their research, which was published in the year of2013, also used a time-weighted Linear Regression model and SVMs on the audiofeatures obtained using the Echo Nest API. The set of song tracks were collectedfrom OCC for UK hits and ZhongGuoGeQuPaiHangBang for Chinese hits. Theyalso defined a “Hit song” as the songs which were ranked 1-20 of the chart and a“non-hit song” as the songs which ranked from 21-40. Their research concluded thatChinese hit song prediction was more accurate than the UK hit song prediction.The error rate for predicting Chinese songs was 41%, while the prediction of UKhits generated a 52% error rate. The Chinese hits appeared to have significantlydifferent characteristics than UK hits [9].

Herremans et al. focused on classifying dance hit songs in a 2014 study. Theyalso extracted features using The Echo Nest API. The datasets used in the modelswere based on the OCC listings. The peak chart position of the songs which rankedfrom 1-40 was used in order to determine if a song was a dance hit or not. And theywere able to create a dataset of dance hit songs from 2009 to 2013. Many MachineLearning algorithms such as Decision trees, Naive Bayes, Logistic Regression, andSVMs were used in their research. They concluded that the Logistic Regression wasthe best algorithm, which could be used to predict dance hit songs, by analyzingaudio features with an accuracy of 0.65 [10].

Another study by Pham et al., at Stanford University in 2016, used differentmachine learning algorithms such as SVMs, Artificial Neural Networks, and LogisticRegression in order to test their ability to predict the popularity of music tracks.They classified using both audio features as well as metadata, which was obtainedby the MSD. A subset of 2,717 tracks was used after removing records which consistof incomplete data that lacked some features of the initially selected 10,000 musictracks. They considered a hit song as a song with a high popularity value providedby The Echo Nest API. They finally concluded that all the models were performingwith nearly similar accuracies, around 75% [5].

While most of these researches are preliminary based on western music tracks, there seems to be a lack of research conducted in the context of Sinhala songs. A recent study by Paranagama et al., in 2017 came up with a solution to automate the process of determining the user ratings of songs by using a multilayer neuralnetwork. They’ve finally been able to achieve an accuracy of 50% in performanceindices, with optimizing the code and solution in various ways such as usingclustering to determine the labels and using pre-stored data for feeding the input [11].

Also, when considering music recommendation systems, Bo Shao et al.discusses the “Collaborative filtering” and “Content-based” approaches and theirdisadvantages. That is the initial startup problem of not having enough dataonuserpreferences over music tracks and how itwould

negatively affect the recommender system's accuracy. They propose a novel approach to overcome these issues by using user access patterns along with content-based features [12].

Music recommendation is vastly used in industry-wide applications. According to Covington et alYouTube uses the watch time, and the click-through rate in order to determine the trending videos. That is the number of users who clicked on a video thumbnail out of the total number of impressions, and the proportion of the total length of the video that user watched has a significant impact rather than the view count of a video [13]. According to Zannettou et al., YouTube content creators tend to use more click baits with false information because the YouTubes algorithm does not consider such click baits in their recommendation algorithm [14].

HY Chang et al. implies how crucial the selection of appropriate music genre, when using music recommendations for stress-related therapies. They proposed a personalized stress-relieving music recommendation system based on EEG feedback [15].

Several previously conducted researches suggest that the popularity of a song can be predicted using machine learning techniques; some also disagree with it. Also, by considering the need of having studies on predicting the popularity of Sinhala songs, this research would expect to continue to investigate this problem.

III. IMPLEMENTATION

1. Introduction to Dataset:
Spotify includes a public API which serves informationabout tracks, albums, artists, genres etc. [6] We obtained our dataset from Kaggle which could be a collection of knowledge of quite 198,000 songs collected from Spotify Web API. [7] Features like acousticness, danceability, energy etc. are available for tracks, artists, genres and years.

2. Problem Statement:
Song Popularity Prediction our aim is to create and test models which might predict the song popularity score (regression) and whether or not the song is popular (classification).

3. Preprocessing Steps:
We apply the subsequent pre-processing steps to our data, to suit our analysis better -

- We remove the features not useful for our analysis like 'id', 'duration' etc.
- We convert the specific variables 'key' and 'mode' using One-Hot Encoding to form them moresuitable for machine learning algorithms.
- Track and artist tables are (left-outer) joined over artist, whose aggregation operations are performed as summarized in Table I. Similarly, track and year tables are joined.

As a result, we obtain 72 columns which might be used as features.

Table 1. Aggregate Operations for Joining Track Artist.

Feature	Operation	Feature	Operation
Acousticness	Mean	Danceability	Mean
Energy	Max	Instrumentalness	Max
Liveness	Max	Loudness	Mean
Speechiness	Mean	Tempo	Mean
Valence	Mean	Popularity	Max
Count	Max	Mode	Max

4. Exploratory Data Analysis:
We analyzed the bottom data to get a matrix between columns, box-plots, and histograms. While some strong correlations were observed between features - as for example, Energy and Danceability - we found just one strong correlation between a feature and popularity.

The subsequent sections cater to algorithms used. We used scikit-learn a preferred machine learning library in Python [8], for the desired tools.

5. Regression algorithms:

5.1 Linear Regression and Polynomial Regression: Perform ing iterative feature selection supported correlation with popularity, we noticed a plateau at around 27 features after which the coefficient of determination failed to increase appreciably and sacrificed computational time.For polynomial regression, we went up to degree 2, beyond which we were computationally limited, due to the dimensions of the dataset.

5.2 Lasso Regression: Lasso Regression has the property of acting and includes a feature selector. [9] It reduced all our feature weights to zero apart from two - artist popularity and year popularity, while giving an affordable accuracy.

5.3 Decision Tree Regression: We consider only binary trees for Decision Tree algorithms. The algorithm results in severe overfitting (99% accuracy on cross validation vs 75% accuracy on test set), and thus we tune the model hyperparameters, whose optimum values are summarized in Table II.

Table 2. Hyperparameters of Decision Tree Algorithms.

Hyperparameter	Regression	Classification
Maximum depth	10	10
Minimum samples to Split	6	12
Minimum samples required at leaf	7	1

6. Classification Algorithms:

The popularity values between 0-100 are converted to labels supporting a decided threshold value. The brink was fixed by looking at the popularity distribution of the songs which appeared within the Billboard Hot 100 a minimum of once using their API. The positive class is in minority as compared to the negative class i.e. the non-popular class. Therefore, we used stratified K Folds technique rather than normal Folds during splitting our dataset and also during cross validation.

In this case, classification accuracy alone can't be used as a reliable performance metric [9]. Therefore, we use other metrics like precision, recall and F1-score. More on this may be explained within the results section.

6.1 Support Vector Machines (SVM): During our experimentation, we observed that a non-linear SVM classifier gives better results. We applied the so-called kernel trick which is simply exploiting math to map the info to a better dimensional space. We use the defaultlinear kernel during training.It'sa regularization parameter called C, which was tuned using grid search technique to provide us the most effective possiblerecall and F1 score

6.2 Decision Tree Classification: We used "Entropy" impurity (versus the conventional "Gini") as our loss function because it yields slightly better results. The model hyperparameters and their optimum values are summarized in Table II.

6.3 Perceptron: We tuned the learning rate and the number of training epochs. After a point, increasing the number of training epochs was not improving results. We chose a feasible value which converged. The values of these hyperparameters are given in Table III.

Table 3.Hyperparameters of Perceptron Classifier.

Hyperparameter	Value
Learning rate	1
Number of training epochs	10000

7. Ensemble Learning:

Ensemble learning involves the use of multiple learning algorithms to improve performance. We limit our scope to classification problems using tree-based algorithms, except for voting classifiers where multiple algorithms need to be used.

7.1 Voting Classifiers: Voting classifiers take a few classifiers' decisions to aggregate them and predict a final decision. Often a classifiers' individual classification is much weaker than a voting classifier's that is the aggregate of all of them. We use both hard and soft voting classifiers and use all the classifiers with thehyperparameters that we have tuned before. We use pipelines to assemble several steps that can be cross validated together while setting different parameters for each of the different classifiers.

7.1.1 Hard Voting Classifier: We use SVM, Logistic Regression, Decision Tree and Perceptron as the voters of this classifier. The class with majority votes from all these classifiers are predicted by the voting classifier.

7.1.2 Soft Voting Classifier: We use all the classifiers except Perceptron as the voters in this, as Perceptron does not give class probabilities. These voters' decisions are pipelined into the voting classifier weighted by the predicted class probabilities. This classifier is expected to give better results as the more confident votes are given higher weightage.

7.2 Random Forests: Random forests are an ensemble of Decision Trees trained via the bagging method. They use the same hyperparameters as DecisionTrees can be used for classification as well as regression. We used Random Forest Classification to train our model. There was marginal change in performance when we used default hyperparameters. We used Randomized Grid Search, which optimizes values for hyperparameters by iterating through random combinations from the parameter grid. We set the Grid Search to go through 100 iterations with 3-fold cross-validation and arrived at a marginal improvement in performance with the optimal hyperparameters. The optimal hyperparameters are shown below.

Table 4. Hyperparameters of Random Forest Classifier.

Hyperparameter	Value
Max depth	20
Minimum samples required at leaf	4
Minimum samples required to split	2
Number of Trees	400
Max features	sqrt

7.3 Boosting: Boosting is a general term for an Ensemble method which combines several weak learners to a strong learner. [9] We consider the two most popular methods - AdaBoost (Adaptive Boosting) and Gradient Boosting. A common disadvantage in boosting algorithms is that they are sequential, unlike bagging algorithms like Random Forests.

7.3.1 AdaBoost: AdaBoost involves the use of consecutive predictors, Decision Stumps (Decision Trees of unit depth) in our case. After every prediction, the weights of training instances are

updated such that the instances which the model underfitting get a larger relative weight. The predicted class is the one which receives the majority of the votes from all the predictors, considering the prediction probabilities as well. The tuned hyperparameters are summarized in Table V.

Table 5.Hyperparameters of AdaBoost Classifier.

Hyperparameter	Value
Number of estimators	250
Learning rate	0.68

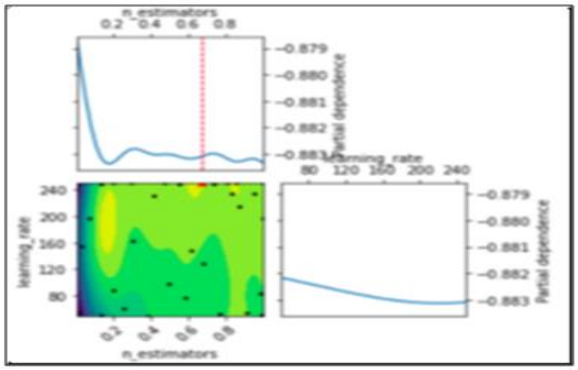
7.3.2Gradient Boosting: Gradient Boosting tries to correct its predecessor by fitting the new predictor to the residual errors made. We used Scikit-learn's new experimental Histogram-based Gradient Boosting Classification (HistGradient Boosting Classifier) which is orders of magnitude faster than the conventional classifier [8]. It discretizes the features to ordinal bins which makes decision trees run faster and the cost on performance is minor. The hyperparameters considered are summarized in Table VI. In addition to the number of trees, we also regulate the size by controlling the maximum number of leaf nodes in a tree.

Table 6.Hyperparameters of Gradient Boosting Classifier.

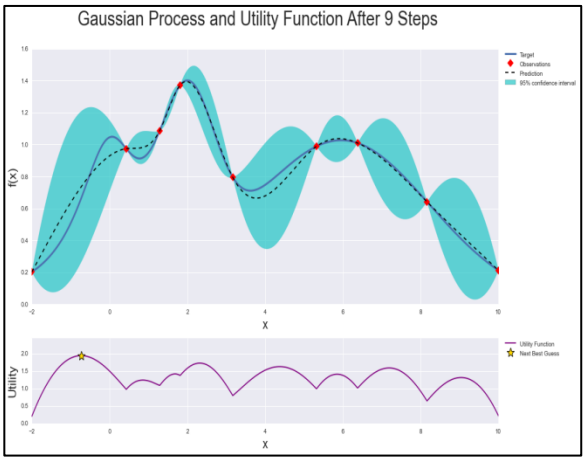
Hyperparameter	Regression
Maximum number of iterations (estimators)	200
Learning rate	0.06
Maximum number of leaf nodes	100
Minimum samples required at leaf	78

A general observation is that on increasing the size of the tree, a decreased learning rate is obtained while tuning the parameters.

7.3.3 Bayesian Optimization in hyperparameter tuning: Boosting algorithms being relatively slower, naivehyperparameter iteration (scikit-learn'sGridSearchCV) is not a feasible option. We use the Bayesian optimization technique for iterating through the hyperparameter space. The technique is helpful when we have a computationally intensive and cost function with noisy evaluations whose closed form solution is not known. This is exactly the case for these algorithms. It does not iterate through the entire space and uses a"surrogate" function to model the same. An acquisition function subsequently chooses the next sample to test the function. [10] We use the implementation from scikit-optimize library [11]. The figure below is Bayesian Optimization which builds a chance model of thegoal function and makes use of it to select hyperparameters tocompare in the true objective feature. The genuine goalfeature is a fixed feature.The image was taken from online pictures as areference.



(a)



(b)

Fig 1. Bayesian Optimization for AdaBoost Classifier.

IV. RESULTS

1. Regression Algorithms:
Decision Tree yields very competitive results compared to the simpler algorithms. In Linear Regression, the results don't improve significantly with anincreased number of features.

2. Classification Algorithms:
As discussed before, it can be seen that accuracy is not a suitable metric for this problem. Our recall scores may have

Table 7. Linear Models.

Regression	Metrics			
	$R2_{Train}$	$R2_{Test}$	$RMSE_{Train}$	$RMSE_{Test}$
Linear Regression	0.82907	0.82920	9.03576	9.05644
Polynomial Regression	0.84282	0.84208	8.66458	8.70819
LASSO Regression	0.79677	0.83086	8.98626	9.01212
Decision Tree	0.86986	0.85649	7.88407	8.30137

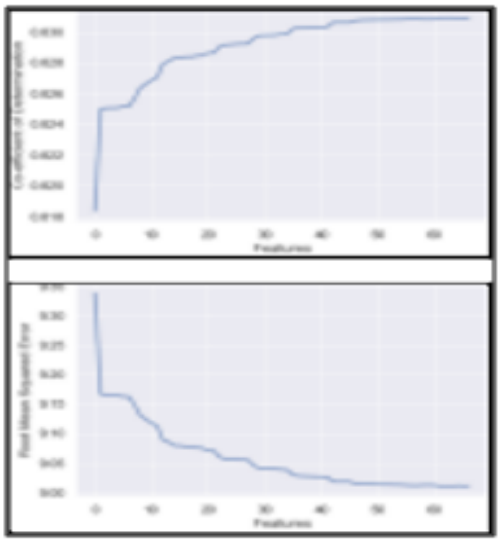


Fig 2. Plots for Linear Regression. (a) R2 score vs. Number of features (b) RMSE vs. Number of features.

Table 8. Performance metrics of classifiers.

Classifier	Metrics				
	Train accuracy	Test accuracy	Precision	Recall	F1 score
Soft voting	0.9198	0.9211	0.84	0.67	0.75
Hard voting	0.9144	0.9142	0.89	0.60	0.72

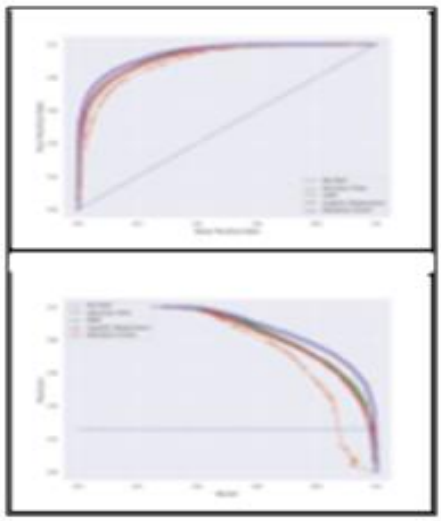


Fig 3. Curves for classification algorithms. (a) ROC curve (b) Precision-Recall curve.

We plot the standard ROC curve and the Precision-Recall curve, the latter being more suited given our imbalanced class distribution. Here, the Decision Tree Classifier's curve is closest to the perfect curve which agrees with the metrics shown in Table VIII.

3. Ensemble Learning:

Ensemble learning techniques show quite some improvement, specifically in the F1 scores. Recall scores are still around the best performing non-ensemble classifier. Something to note is that these techniques are computationally expensive. This is the same reason as to why we couldn't generate the ROC and Precision-Recall curves on time for the submission.

Table 9. Performance metrics of tree-based methods.

	Metrics				
	Train accuracy	Test accuracy	Precision	Recall	F1 score
Random Forest	0.9545	0.8954	0.86	0.69	0.76
Adaptive Boosting	0.8837	0.8820	0.83	0.70	0.76
Gradient Boosting	0.8918	0.8911	0.86	0.70	0.77

Table 10. Performance metrics of voting classifiers.

Classifier	Metrics				
	Train accuracy	Test accuracy	Precision	Recall	F1 score
Logistic Regression	0.9105	0.9110	0.79	0.64	0.70
SVM	0.9144	0.9142	0.81	0.69	0.74
Decision tree	0.8924	0.8868	0.84	0.71	0.77
Perceptron	0.8116	0.8133	0.69	0.50	0.58

Some scope for improvement number of false negatives is considered high.

A classifier can be by the confusion matrix generated for a given sample.True Positive (TP), False Positive (FP), True Negative (TN) and False

Negative (FN) were calculated for each test case. Then accuracy and FP rate was calculatedfor each test case. Reducing FP rate is significant as same as increasing the accuracygiven that this method mainly focuses on identifying music on radio broadcasts.accuracy and FP rate can be calculated from the formulas given below.

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

And the Precision can be numerically calculated as:

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

The Extended Gradient Boosting ClassificationAlgorithm produced thefollowing results.

Shape of training data: (9452, 69)

Shape of testing data: (4052, 69)

Accuracy score on train dataset : 0.7329665679221329

Accuracy score on test dataset : 0.7129812438302073

4. A Measure for the recognition:
The popularity score for every song is calculated per the subsequent formula.

First the download count for one song is normalized to a price between 0and 1 (inclusive) consistent with the subsequent formula.

Norm(download count) = download count of the song−min(downloads)

max(downloads)−min(downloads)

Next the view count for one song is normalized to a price between 0 and 1 (inclusive) in keeping with the subsequent formula.

Norm(view count) = view count of the song−min(views)

max(views)−min(views)

The Click through Rate (Downloads per View) is obtained by the subsequentformula.

click through rate =Norm(download count) — Norm(view count)

Finally the recognition score comes by this formula.

The popularity score = Norm(download count)∗ click through rate

The click through rate is an optimal method so as to see the recognitionwhere the ranking supports the reach to a selected audience [19, 20].A study by Zhou et al. presents their findings stating that the clicking through rate isan important thing about YouTube video recommendation which also concludes thatthe popularity of the video relies on theright recommendation mechanism [21].

Therefore, here the press through rates is getting used because of the measure of thepopularity score.Experimentally by observing the distributions, the songs which have a downloadcount but or up to 41,250 and a viewcount but or adequate 93,874are only considered. Other song statistics are considered as noise. The popularity scores are worth between 0 and 1. The songs are divided into 3 classes supporting these popularity values.The boundary and therefore the lowerbound for every of those classes are decided by observing the elbow curve turningpoints when the ordered distribution of therecognition score values are plotted.The KneeLocator Python library is getting used for this.

V. CONCLUSION

We conclude that popularity can be predicted reasonably well using machine learning techniques. In both regression and classification, Decision Treesworked reasonably well. Ensemble algorithms give a slight improvement over traditional classification algorithms, very similar results with a common shortcoming of mis-classifying popular songs as unpopular. These algorithms show similar performance, with voting classifiers and Random Forest giving marginally better accuracy and boosting algorithms giving better recall and F1 score. The differences are negligible and parallelizable algorithms can be preferred.The purpose of using machine learning algorithms during this project was to work out.If such patterns exist in genre and therefore the experiments would suggest thatthe music audio signal consists of patterns which makes it a likable song for several.

This research considered only the audio signal related data so as to predictwhether successful or not. The test results produced 71% accuracy in predicting thecorrect popularity class for unseen data and therefore the oftenness related features, duration and tempo would be key elements where song popularity depends on.As depicted within the SHAP values, the 'loa 2' (The second coefficient of the LinearPredictive Coding) has the foremost significance and contribution towards the output. Linear predictive coding may be a method used mostly in audio signal processing and speech processing for representing the spectral envelope of a digital signal of speech in compressed form, using the data of a linear predictive model.

Inaddition thereto, it alsowants to identify the resonance characteristics of audiothis means that the repetition property which lies within the audio signal is a key element that correlates with music popularity.Additionally most of the similar research during this area is using theMillion Song Dataset where the features are already extracted using proprietaryalgorithms. Since this research describes how the features were extracted and also thedataset is re-obtainable by anyone, it might be contributing to further research onpredicting the recognition of songs.

VI. IMPLICATIONS FOR FURTHER RESEARCH

This research only depends on the information collected at a particular point of time. Sincethetrendsofmusicchangeovertime,further research is also done by collecting a dataset over a period of some

time and analyzing the results. Also, this research uses the XGBoost algorithm for processing. Other machine learning algorithms are going to be used to compare ends up in order to look out the foremost effective model.

The purpose of using machine learning algorithms during this project was to see if such patterns exist in music genre and also the experiments would suggest that the music genre and also the experiments would suggest that the music audio signal consists of patterns which make it a likable song for several. With the results produced by this research, it could also be considered that the forms of repetition within a song impact the recognition, since it makes the song memorable.

REFERENCES

[1] R. Dhanaraj and B. Logan, "Automatic prediction of pp. 355–360, 2008. hit songs," pp. 488–491, 2005.
[2] R. Nijkamp, "Prediction of product success:Y. Ni, R. Santos-Rodriguez, M. Mcvicar, and T. De explaining song popularity by audio features fromBie, "Hit song science spotify data," in 11th IBA
[3] Bachelor Thesis once again a science," in 4th International Workshop on Conference, 2018. Machine Learning.
[4] A. Singhi and D. Brown, "Can song lyrics predict and Music: Learning from Musical Structure, Sierra hits?" 2015.
[5] K. Bischoff, C. S. Firan, M. Georgescu, Nevada, Spain, Citeseer, W. Nejdl, and R. Paiu, "Social knowledge-driven music 2011.hit prediction," in Proceedings of the 5th International.
[6] N. Borg and G. Hokkanen, "What makes for a hit Conference on Advanced Data Mining and Applications, pop song? What makes forser. ADMA '09? Berlin, Heidelberg: Springer-Verlag, a pop song," Unpublished thesis, Stanford University, 2009, p. 43–54.
[7] E. Zangerle, M. Pichl, B. Hupfauf, California, USA, 2011.and G. Specht, "Can microblogs predict music charts?
[8] J. Fan and M. Casey, "Study of chinese and uk hit an analysis of the relationship between now playing songs prediction," in tweets and music charts," in Proceedings of the 17th Proceedings of International Symposium on Computer International Society for Music Information Retrieval Music Multidisciplinary Conference 2016 (ISMIR 2016). ISMIR, 2016. Research, pp. 640–652, 2013.
[9] "Web api reference." [Online]. Available:algorithms," 2012.
[10] T. Head, M. Kumar, H. Nahrstaedt, G. Louppe, and I. Shcherbatyuk, "scikit-optimize/scikit-optimize," 2020.[Online]. Available:https://zenodo.org/record/4014775
[11] B. Shao, D. Wang, T. Li, and M. Ogihara, "Music recommendation based on acoustic features and user access patterns," IEEE Transactions on Audio, Speech, and Language Processing, vol. 17, no. 8, pp.1602–1611, 2009.
[12] M. B. Holbrook and R. M. Schindler, "Some exploratory findings on the development of musical tastes," vol. 16, no. 1, p. 119.
[13] M. Nasreldin, "Song popularity predictor," May 2018.
[14] R. Dhanaraj and B. Logan, "Automatic prediction of hit songs." in ISMIR, pp. 488–491, 2005. D. P. Ellis, B. Whitman, and P.T. Bertin-Mahieux, Lamere, "The million song dataset," 2011.
[15] J. Q. Pham, "Predicting song popularity," 2015.
[16] F. Pachet and P. Roy, "Hit song science is not yet a science.," in ISMIR,
[17] D. Herremans, D. Martens, and K. S¨orensen,https://developer.spotify.com/documentation/web-api/reference/ [7] Y. E. Ay, "Spotify dataset 1921-2020, 160k
[18] Journal of New Music Research, vol. 43, no. 3, pp. tracks (version 10n," Jan 2021. [Online]. Available: 291–302, 2014. "Dance hit song prediction," https://www.kaggle.com/yamaerenay/spotify
[19] T. Paranagama and A. Ariyaratne, "A machinedataset-19212020-160k-tracks/version/10 learning approach to classify.
[20] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, sinhala songs based on user ratings," 08 2017.
[21] B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R.B. Shao, D. Wang, T. Li, and M. Ogihara, "Music Weiss, V. Dubourg, J. Vander plas, A. Passos, D.Cournapeau, M. Brucher, M. Perrot, and E. Duch recommendation based essay, "Scikit-learn: Machine learning in Python," on acoustic features and user access patterns," IEEE Journal of Machine Learning Research, vol. 12, pp. Transactions on Audio, 2825–2830, 2011. Speech, and Language Processing, vol. 17, no. 8, pp.
[22] A. Geron, ´ Hands-on machine learning with 1602–1611, 2009. Scikit-Learn, Keras, and TensorFlow: Concepts,
[23] M. Airoldi, D. Beraldo, and A. Gandini, "Follow the tools, and techniques to build intelligent systems.algorithm: An O'Reilly Media, 2019. exploratory investigation of music on youtube," Poetics,
[24] J. Snoek, H. Larochelle, and R. P. Adams, "Practical vol. 57, pp. 1–13, Bayesian optimization of machine learning 2016.
[25] S. Zannettou, M. Sirivianos, J. Blackburn, and N. Kourtellis, "The web of false information: Rumors, fake news, hoaxes, clickbait,and various other shenanigans," Journal of Data and Information Quality (JDIQ), vol. 11, no. 3, pp. 1–37, 2019. H.-Y. Chang, S.-C.
[26] Huang, and J.-H. Wu, "A personalized music Recommendation system based onelectroencephalography feedback,"Multimedia

Tools and Applications, vol. 76, no. 19, pp. 19523–19542, 2017.

[27] H. MacGillivray, J. M. Utts, and R. F. Heckard, Mind on statistics. Cengage B. McFee, C. Raffel, D. Liang, D. P. Ellis, M. McVicar, E. Battenberg, and O. Nieto, "librosa: Audio and music signal analysis in python," in Proceedings of the 14th python in science conference, vol. 8, 2015.

[28] C. McKay, I. Fujinaga, and P. Depalle, "jaudio: A feature extraction library,"in Proceedings of the International Conference on Music Information Retrieval,pp. 600–3, 2005.

[29] P. Chandar and B. Carterette, "Estimating clickthrough bias in the cascade model," in Proceedings of the 27th ACM International Conference on Information and Knowledge Management, pp. 1587–1590, 2018.

[30] J. Davidson, B. Liebald, J. Liu, P. Nandy, T. Van Vleet, U. Gargi, S. Gupta,Y. He, M. Lambert, B. Livingston, et al., "The youtube video recommendationsystem," in Proceedings of the fourth ACM conference on Recommendersystems, pp. 293–296, 2010.

[31] R. Zhou, S. Khemmarat, and L. Gao, "The impact of youtube recommendation system on video views," in Proceedings of the 10th ACM SIGCOMM conferenceon Internet measurement, pp. 404–410, 2010.

[32] S. Rathi, "Generating counterfactual and contrastive explanations usingshap," arXiv preprint arXiv: 1906.09293, 2019.

[33] N. Dave, "Feature extraction methods lpc, plp and mfcc in speech recognition,"International journal for advanced research in engineering and technology, vol. 1, no. 6, pp. 1–4, 2013.