Predicting Music Popularity Using Music Charts

Carlos Vicente S. Araujo
Institute of Computing
Federal University of Amazonas
Manaus, Brazil
vicente@icomp.ufam.edu.br

Marco A. P. Cristo
Institute of Computing
Federal University of Amazonas
Manaus, Brazil
marco.cristo@icomp.ufam.edu.br

Rafael Giusti
Institute of Computing
Federal University of Amazonas
Manaus, Brazil
rgiusti@icomp.ufam.edu.br

***Abstract*—Online streaming platforms have become the most important method of music consumption. Most streaming platforms provide tools to assess the popularity of a song by means of scores and rankings. In this paper, we present a methodology for predicting if a song will appear on Spotify's Top 50 Global ranking after a certain amount of time. Spotify is one of the biggest streaming services and if a song is popular on this platform, it is likely to ensure good financial return to the artists and their label. We approach the problem as a classification task and employ classifiers built on past information from the platform's Top 50 Global ranking, in addition to acoustic features from the songs. The Support Vector Machine classifier with RBF kernel reached the best results in our experiments with an AUC higher than 80% when predicting the popularity of songs two months in advance.**

***Index Terms*—machine learning, popularity prediction, music prediction, Spotify**

I. Introduction

Music is serious business. According to the International Federation of the Phonographic Industry—IFPI—, the worldwide revenue from streaming, performance rights, and sales of CDs and digital music amounted to US$ 19.1 billion in 2018. The majority of this figure is due to digital media: 47% of that share comes from streaming and 12% from digital sales[1]. Currently, one of the largest streaming services is Spotify, which by the end of 2018 had over 191 million active users[2].

In such a high-volume market, it is important to understand what makes a product or artist valuable. That knowledge could be used to promote marketing strategies, to decide the best moment to release a new album, to align the artists' effort with public interest, etc. One way to measure the success of a song or album is by inspecting rankings and charts. Spotify, for instance, publishes several charts. In particular, its "Top 50" is a daily chart that features the 50 most popular songs of the previous day, ranked by number of streams. Given that Spotify reaches such a large audience, we assume it would be of great interest to artists to know if, and when, their work will become popular in the platform. That could be used, for example, to boost marketing efforts on a piece that is not yet expected to become popular, or perhaps to steer effort from unpromising cases into more promising ones.

Supported by Coordination for the Improvement of Higher Education Personnel (CAPES).

[1]www.ifpi.org/downloads/GMR2019.pdf

[2]https://www.fool.com/investing/2018/12/17/whats-most-popular-music-streaming-service-2018.aspx

The scientific literature is rich in works that deal with predictions in the music market, including predicting the popularity of songs and albums, with machine learning methods achieving the best results in general. In the past, several sources have been used to provide data that reflects the popularity of musics, such as social networks [1] [2], acoustic features extracted from the songs themselves [3] [4], attendance statistics in live performance and festivals [5] [6] and from the collaboration networks among artists [7] [8].

In this work, we consider a song to be successful if it is featured in Spotify's "Top 50" charts. This is similar to approaches that consider monthly or yearly rankings as a criterion of popularity [9] [10]. We employ a model that is trained with data from that chart, and we can predict with 89.09% accuracy whether a music will be popular two months later. We focus on Spotify because it publishes daily rankings and because we are interested in the global streaming market. However we note that our methodology could be adapted to different sources.

The remainder of this article is as follows: in Section II we show related works. In Section III we present the methodology used in this study. The results we obtained are presented in Section IV. Finally, in Section V we make a brief conclusion and point out possible future work.

II. Related Work

We have observed few popular approaches to the problem of music success prediction, despite it having received growing attention for years. We remark three general strategies: the first uses social network data to assess current public perception and extrapolate how successful a song or album will be. The second relies on acoustic information of previously successful songs to predict the success of other songs. The third, which our work is a part of, uses past data from charts to predict whether a song will be featured in that same chart in the future.

An example of the first strategy is the work of Dhar and Chang [11]. The authors gathered comments related to 108 albums, before and after they were released, from sources that included social networks and blogs. Their objective was to verify if the sales were reflected in the comments. They also took into consideration data from the songs and artists: the recording label, the time between announcement and release of the album and the reception of the artists' previous work—the number and positiveness of the reviews

978-1-7281-4550-1/19/$31.00 ©2019 IEEE
DOI 10.1109/ICMLA.2019.00149

published in specialized venues, such as the Rolling Stone and Entertainment Weekly magazines, and the scores given by users in online platforms. The authors concluded that the factors that best correlated with sales were the number of posts in social networks and blogs—without checking if those comments were positive or negative—, and the average rating of the artist's previous works by users. At the same time, they also remark that traditional factors, such as traditional media coverage, are still important, and that albums released by larger musical labels attain results 12 times greater.

Shulman, Sarma and Cosley [12] also take the same approach. The authors compiled a data set with data from 437 thousand users and the music users interacted with in the platform Last.fm (a social network about music). The data were separated into two kinds: temporal and non-temporal, but no specific information is given about which constitutes each kind. The objective was to create a model to predict if a specific song would get higher than average interaction on Last.fm. A logistic regression model using both temporal and non-temporal data was reported to achieve 81% accuracy, while the temporal data alone led to a classifier with 80.6%.

Among works in the second group, the ones based on acoustic features, Lee and Lee [13] collected data on $16,686$ songs that appeared for more than two weeks on Billboard's Hot 100 ranking between 1970 and 2014. For each song, the authors extracted features like chroma, rhythm, timbre, and MFCC. They also employed non-acoustic information, including the number of weeks each song was featured in the ranking and the average of their weekly ranks. They trained a Support Vector Machine (SVM) model with RBF kernel and achieved 70% accuracy when predicting which would be the best rank a song would achieve.

Interiano et al. [14] also employed acoustic information. They collected data from $500,000$ songs featured in the Top 100 Singles Chart by the Official Charts Company (OCC)[3] in the UK and from MusicBrainz[4]. Their objective was to predict whether a song would be a success, which they defined as being featured in the Top 100. To accomplish this, they extracted high-level acoustic features which identify music by different properties, including tonality, "danceability", and mood—*e.g.*, whether a song sounds happy, sad, or aggressive. A Random Forest classifier achieved 70% accuracy. However, they achieved an accuracy of 85% by adding a variable indicating whether the piece was performed by a "superstar", which they defined as an artist that featured as first in the ranking at least once in the past five years.

Finally, the third approach consists of making the prediction mostly with data drawn from the rankings or charts themselves. Herremans, Martens and Sörensen [9] is an example of this. The authors extracted data from the OCC Top 40 Dance Music between 2009 and 2013 and considered a song to be successful if it was featured up to a certain position in the ranking. The authors obtained the better results when the Top 10 tracks were considered as success, the ones in positions #11 to #30 were not used in the experiment and the last 10 were considered as flop. Using a SVM classifier with polynomial kernel they reached an accuracy of 85% in average.

Reiman and Örnell [10] collected data for 287 songs that were featured in the Billboard Hot 100 between 2016 and 2018. Because Billboard only provides the rankings, the authors extracted data about the songs from Spotify. They also collected data for other 322 songs chosen randomly from 13 distinct genres that never appeared on the Hot 100. A Gaussian Naive Bayes classifier was able to predict with 60.2% accuracy whether a particular song would be featured in the ranking.

The work we propose in this paper also lies in this third approach. We have collected data from Spotify's "Top 50" lists, and we consider a song to be successful it is featured in those lists. Hence, our objective is to predict whether a song will be successful in the future. Unlike the previously described works, we make predictions in long term. Thus, instead of simply splitting the data set into training and test sets, we perform series of classification rounds where the results of a round are used as input of the following ones, enabling long term predictions.

III. Methodology

We framed our objective as a classification problem. Specifically, given a song that may or may not currently be popular, can we predict whether it will be popular after a specified number of days into the future?

Our methodology is divided into the following steps. First we collect data from past editions of the "Top 50" (Section III-A). We next transform ranking entries into instances (Sec. III-B). Then, to increase the prediction horizon, we perform several prediction rounds where the results of one round are used as estimates to the features of the next ones (Sec. III-C). Finally, we begin our experimental methodology, which consists in defining a set of classifiers and validating them against a test data set (Sec. III-D).

A. Data Collection

Data from the Top 50 Global lists—a.k.a. rankings—were collected using Spotify's Web API[5]. The data were collected on a daily basis between November 2018 and April 2019.

For each daily list, we collected information from nine fields made available by the API. Namely, the entry's rank, the ranking date, the artists names, the song title, the song release date, its duration in milliseconds, and a URL for the song's 30-second sample. Additionally, each entry has an "explicit" flag, which indicates whether the song contains profanity, and a popularity score, which is a value in the $[0, 100]$ interval that reflects how popular the song was on that day.

For each song featured in at least one edition of the ranking, we downloaded its 30-second sample in order to extract acoustic features. We note that approximately 3% of the songs were missing a URL and were discarded. For the remaining songs,

[3]https://www.officialcharts.com/

[4]https://musicbrainz.org/

[5]https://developer.spotify.com/documentation/web-api/

we used the Python package LibROSA [15] to extract five acoustic features, namely Mel-Frequency Cepstral Coefficients (MFCC), spectral centroid, spectral flatness, zero crossings, and tempo. We chose this specific set of features because they appeared frequently during our literature overview.

B. Instance extraction

Having collected data from 180 editions of Spotify's "Top 50" ranking, the simplest way we could construct a data set would be by taking every entry from each ranking and transform it into an instance. However, this would mean that we could only have instances for a song while they are popular. Because our methodology requires information about non-popular songs, we augmented the data set as follows.

Let U be the set of all songs that were featured at least once, during the whole data collection period, in Spotify's Top 50 lists. Let P_i be the set of popular songs for a particular i-th edition of the list—*i.e.*, those that feature in the ranking of that particular day. Then we now define the set of unpopular songs as the set $\overline{P_i} := U \setminus P_i$.

Given these definitions, for each daily edition of the list, we can produce a maximum of $|U|$ instances. Each song $s_{i,j} \in P_i$ translates directly into an instance $x_{i,j}$—*i.e.*, it is a feature vector that contains the same information provided in the list, plus the acoustic features previously extracted. However, the songs in $\overline{P_i}$ do not contain a rank value, neither a score value. We considered all unpopular songs to be tied after the 50^{th} rank and assign to all of them the average rank $\frac{(51+|U|)}{2}$. We also estimate their score to be the lowest observed value during the whole period.

Next, because we are attempting to handle an inherently temporal problem without explicitly making use of temporal models, we additionally augment the data set by employing lagged features. We chose this approach because we only have information about the songs while they are popular, and attempting to construct a truly temporal data set would lead to very short time series with many gaps. While lagged features introduce some correlation among instances, our results suggest that this compromise is acceptable.

Our instances are lagged with information concerning a song's popularity a few days *after* its popularity status was verified. Let $X_i = \{x_{i,1}, x_{i,2}, \ldots, x_{i,|U|}\}$ be the set of instances drawn from the popular and unpopular songs of a particular day. Then each instance $x_{i,j}$ is augmented by adding lagged features from $x_{i+\delta,j}$, $x_{i+2\delta,j}$, ..., $x_{i+k\delta,j}$. The lagged features are binary variables that indicate whether a song is popular in the future.

The last feature added to each instance is the target value, which is a binary variable that indicates whether a song is popular $(k+1)\delta$ days after the date of the first instance. For example, if we define δ=30 and k=4, then for each instance the target will be its popularity status five months into the future. The only imposition made by our methodology is that the values of δ and k must be the same when training and applying a classifier. We also note that the greater we set those values, the further we can predict into the future, but this also means we need to collect data for longer periods.

At this point, we want to remark that this process is repeated a few times in a series of classification rounds. A consequence of this is that lagged and target features may be real or synthetic. We will revisit this particularity in depth in Section III-C.

The last step to construct our data set consists of transforming non-numeric features and removing obsolete ones. We replace dates with epochs by using the "to_numeric" function from the Pandas library, and we drop the names of the tracks and artists, as well the URLs. The names were only required to create instances for unpopular songs, and the URL was necessary to retrieve the 30-second samples. Finally, we standardize all features by replacing them with z-scores.

C. Training and Prediction

For our experiments, we defined the lag period to be δ=20 days and the number of lagged features to be k=3. We chose those values in order to maximize the number of lagged features, given the number of rankings we collected.

We used data from November 1^{st} through December 10^{th} of 2018 as the base dates for our training data set. During the entire period (180 days), we observed 274 different songs, therefore the training data set contains $40 \times 274 = 10{,}960$ instances. Each instance is lagged with popularity information from the next 60 days and is labeled according to the popularity of each song $(k+1)\delta = 80$ days later. Therefore the labels run from January 20^{th} through February 28^{th}. This organization is represented by the boxes at the top of Figure 1.

Each test instance must be derived by the same procedure. That is, it must have lagged features for 60 days after its base date, and the target is 80 days later. For example, if the base date is January 1st, 2019, then lagged features are required for Jan/21, Feb/10, and Mar/02. This instance could be used to predict the popularity of a song on March 22^{nd}. In other words, the model is only able to make a prediction 20 days after the date of the last lagged feature.

We overcome this limitation by performing several induction and prediction rounds. At the start of each round, we build a training set from 40 editions of the ranking, adding lagged features and labels as previously explained. Then we induce a classifier, which we use to predict the popularity of every song for 20 new days, as illustrated in the topmost segment of Figure 1. These predictions provide estimates for new lagged features and targets, which we may now use as the target features of the second round, as shown in the middle segment of Figure 1. Each round allows us to predict 20 days into the future. However, due to current limitations of real data, our methodology is limited to a number of rounds. We currently do not provide means to estimate rank values or scores of songs that are popular at a given day, therefore we can no longer induce classifiers when that data becomes unavailable.

At each round in Figure 1, the lighter boxes represent the days used to extract data to induce a classifier, while the darker boxes represent the days used to extract data to predict

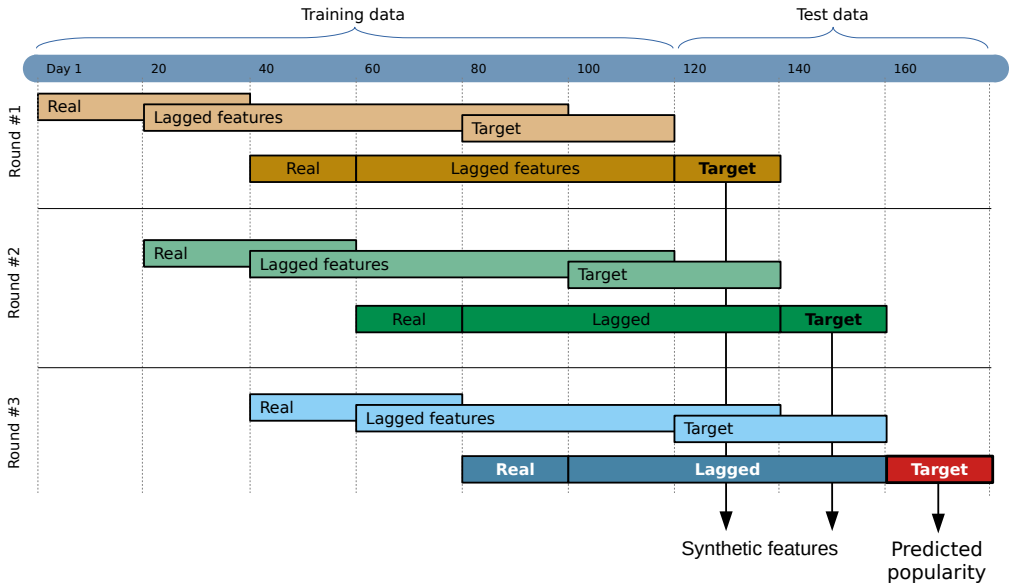

Fig. 1. Depiction of the classification rounds. At each round, we train a classifier and predict popularity information more 20 days into the future. The lighter boxes of each round represent the data used to train the classifier, and the darker boxes represent the data used to predict the new popularity information, which is used to train subsequent classifiers. This allows us to improve our prediction horizon threefold. After the last round, the predicted popularity is both the set of test labels and the long-term prediction one would be interested in obtaining for any particular song. The boldface boxes represent test features, which are completely separated from the training at each previous stage.

popularity for the next set of 20 days. We remark that, although there is some overlap in the data, this is due to the fact that each classifier is trained with data that was "shifted" over time. This is similar to prediction of time series and do not represent a mixture of training and test data. Most importantly, the target features used to test the methodology, represented by the boldface boxes in the figure, are disjoint from the data used to induce each previous classifier, being used instead by subsequent classifiers in the form of synthetic data as estimates of unobserved popularity information.

D. Data Prediction

We evaluated our methodology with different classification models. We tested ensemble techniques with Ada Boost and Random Forests, probabilistic models with Bernoulli and Gaussian Naive Bayes, and multiple kernels for the Support Vector Machine classifier.

AdaBoost is a meta-classifier that boosts weak-learners by assigning weights to the data set. The first base classifier is generated with all instances sharing the same weight. At each iteration, the weight of the instances incorrectly classified by the previous model is increased, so that each base classifier is adjusted to focus on more difficult cases [16]. The predictive power of the ensemble may vary significantly depending on which base classifier is employed. In our experiments, we used the AdaBoost-SAMME [17] implementation from Scikit-Learn, which employs small decision trees as weak learners.

Naive Bayes (NB) methods are a set of supervised learning algorithms based on applying Bayes' theorem with the "naive" assumption of conditional independence between every pair of features given the value of the class variable. The different naive Bayes classifiers differ mainly by the assumptions they make regarding the distribution of the attributes. In the Gaussian Naive Bayes algorithm the likelihood of the features is assumed to be Gaussian [18], while Bernoulli Naive Bayes assumes that the data follow the multivariate Bernoulli distribution [19].

A random forest is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting [20]. Random forests are ensembles that take advantage of feature selection methods and improve prediction power by combining several weak-learning classifiers.

SVM is an instance-based classifier that projects the training samples into a space of higher dimensionality, where it assumes to have a representative layout of the original space. In this projection, SVM attempts to find the hyperplane that best separates the classes, effectively dividing the decision space into two subspaces. When classifying a new sample, SVM projects its features into the same high-dimensional space and verifies on which subspace the projected instance "falls", and then assigns it the class label associated with that subspace [21] [22]. The kernel function defines the inner product in the transformed space, so that different kernels imply on different ways to calculate inner products [23].

Besides testing different models, we also verified whether using acoustic information in addition to data collected from the rankings would translate into higher classification accuracy. Therefore, we repeat the entire process discussed in the previous section twice: once where every instance contained

acoustic features, and a second time where acoustic features had been removed from the data.

IV. Results

Our test data set was extracted from the last round of the process presented in Section III-C. Specifically, we are testing our methodology's ability to predict, two months in advance, the popularity of 274 for songs for every day after day 160, between April 10^{th} and 29^{th}, 2019. Therefore, our test set has 4,517 instances. Because some entries were excluded from the data set during the collection phase, the number of popular instances for each ranking is not exactly 50, so the total number of of negative (unpopular) instances is in the test is $4,517$, while the number of positive (popular) instances is 963.

We show the metrics attained by each model without acoustic information in Table I. The metrics for the models using acoustic information in addition to ranking data are shown in Table II.

We evaluate our proposal from the perspective of multiple metrics, rather than a single one. Because our data set is heavily unbalanced, with roughly 4.6 times more negative (unpopular) instances than positive (popular), metrics more robust to imbalance are more appropriate in this work. In particular, we focus more on recall, miss rate values, F1 Score, and AUC than in precision and accuracy.

Considering these factors, the best model for our problem was the one that uses all available data and was trained using the SVM classifier with RBF kernel. The polynomial kernel-trained model reaches better results in precision, specificity and fall-out because it correctly predicted eight more negative instances than the RBF kernel, even though it has correctly predicted 12 fewer positive instances.

Using the acoustic features the RBF kernel model predicts correctly 29 negative instances and 4 positive instances more than the previous data based model. These 33 instances represent an improvement of only 5.23% over the equivalent classifier without acoustic features. While we do not categorically draw any conclusion, we remark that such a difference may not be statistically relevant. This indicates that in our experiments the acoustic features did not present a relevant impact on the performance of our model. One possible explanation for this may be the fact that the extraction of these features depended on short samples of the tracks. These samples may not be representative of the songs in their entirety, which may have skewed such data.

On the other hand, the fact that the best model in our research does not show a significant gain in performance when using acoustic features can be encouraging. The extraction of such features can be computationally expensive, and it is a good signal that they may not be essential to achieve satisfactory results.

We have verified that the main point to develop in our model is to find a way to decrease the number of false positives, *i.e.* instances that the model predicts would be present in the ranking, but that in fact are not. Our best model incorrectly predicts 31.57% of positive instances.

In summary, we created a model that predicts whether a song will or not appear in Spotify's Top 50 Global ranking after two months of the entries date. The model uses information of the songs previously observed in that list. In this task, the values obtained in accuracy, Negative Predictive Value (NPV), specificity and AUC were all above 80%. In addition, the fall-out in the best case was only 6.51%.

V. Conclusion and Future Work

In this paper we present a methodology to predict whether a song will be popular using data collected from popularity charts. We collected data from the streaming platform Spotify and trained classification models that are able to make predictions two months into the future. Our experimental methodology shows that our model attains AUC metric ranging from 0.60 to 0.80, depending on the classifier used to make the predictions.

Specifically, the best results were obtained by the SVM classifier with RBF kernel. The accuracy, NPV, specificity and AUC were above 80% in this experiment. We also noticed that when using the acoustic features the performance was only 5.23% higher, which may not be statistically relevant. This might indicate that acoustic information may be completely neglected, but further studies are required to confirm that hypothesis.

We remark that, while our work is limited to the extent of the data provided by Spotify, the proposed model may be extended to other platforms. In particular, if a platform provides lists of popular artists, albums, or songs, those could be directly used in our model. For instance, we suspect we might employ the same methodology to predict the popularity of YouTube videos, which are frequently ranked in the "Popular on YouTube" playlist.

We do identify a few limitations in our methodology. First, although we are able to extend our prediction horizon by promoting repeated steps of induction and prediction, we are still required to collect twice as much data from the period we want to make our final predictions. This is because we require some real data to be used in the "induction" step. We believe this necessity could be alleviated if we employed a regressor instead of a classifier at the end of each round. This would allow us to extend even further our prediction horizon.

Furthermore, while we do aim at exploring models that do not require social network information, we note that our model might benefit from them. We intend to use that type of data to garner more information as means of more accurately establishing the popularity of a song, album or artist. This could be used to improve accuracy and, most importantly, reduce our false positive rate.

References

[1] Y. Kim, B. Suh, and K. Lee, "#nowplaying the future billboard: Mining music listening behaviors of twitter users for hit song prediction," in *Proceedings of the First International Workshop on Social Media Retrieval and Analysis*, ser. SoMeRA '14. New York, NY, USA: ACM, 2014, pp. 51–56. [Online]. Available: http://doi.acm.org/10.1145/2632188.2632206

TABLE I
PERFORMANCE OF THE MODELS WITHOUT ACOUSTIC FEATURES.

Previous Data	Ada Boost	Bernoulli NB	Gaussian NB	Random Forest	SVM Linear	SVM Poly	SVM RBF	SVM Sigmoid
Accuracy	0.8828	0.8849	0.8359	0.8730	0.8849	0.8849	0.8849	0.7856
Precision	0.6762	0.6697	0.5255	0.6599	0.6697	0.6697	0.6697	0.3750
NPV	0.9241	0.9316	0.9281	0.9113	0.9316	0.9316	0.9316	0.8608
Recall	0.6397	0.6802	0.6843	0.5722	0.6802	0.6802	0.6802	0.3302
Specificity	0.9347	0.9285	0.8683	0.9371	0.9285	0.9285	0.9285	0.8827
F1 Score	0.6574	0.6749	0.5945	0.6129	0.6749	0.6749	0.6749	0.3512
AUC	0.7872	0.8043	0.7763	0.7546	0.8043	0.8043	0.8043	0.6064
Fall-out	0.0653	0.0715	0.1317	0.0629	0.0715	0.0715	0.0715	0.1173
Miss Rate	0.3603	0.3198	0.3157	0.3401	0.3198	0.3198	0.3198	0.6698

TABLE II
PERFORMANCE OF THE MODELS USING ALL AVAILABLE DATA.

All Data	Ada Boost	Bernoulli NB	Gaussian NB	Random Forest	SVM Linear	SVM Poly	SVM RBF	SVM Sigmoid
Accuracy	0.8867	0.8869	0.8286	0.8847	0.8849	0.8901	**0.8909**	0.7887
Precision	0.6804	0.6774	0.5092	0.6918	0.6697	**0.6935**	0.6915	0.3818
NPV	0.9298	0.9318	0.9284	0.9207	0.9316	0.9305	**0.9328**	0.8608
Recall	0.6698	0.6802	0.6895	0.6199	0.6802	0.6719	**0.6843**	0.3271
Specificity	0.9329	0.9309	0.8583	0.9411	0.9285	**0.9367**	0.9349	0.8871
F1 Score	0.6750	0.6788	0.5858	0.6539	0.6749	0.6825	**0.6879**	0.3523
AUC	0.8013	0.8055	0.7739	0.7805	0.8043	0.8043	**0.8096**	0.6071
Fall-out	0.0671	0.0691	0.1417	0.0589	0.0715	**0.0633**	0.0651	0.1129
Miss Rate	0.3302	0.3198	0.3105	0.3801	0.3198	0.3281	**0.3157**	0.6729

[2] C. V. Araujo, R. M. Neto, F. G. Nakamura, and E. F. Nakamura, "Predicting music success based on users' comments on online social networks," in *Proceedings of the 23rd Brazillian Symposium on Multimedia and the Web*, ser. WebMedia '17. New York, NY, USA: ACM, 2017, pp. 149–156. [Online]. Available: http://doi.acm.org/10.1145/3126858.3126885

[3] J. Lee and J.-S. Lee, "Predicting music popularity patterns based on musical complexity and early stage popularity," in *Proceedings of the Third Edition Workshop on Speech, Language & Audio in Multimedia*, ser. SLAM '15. New York, NY, USA: ACM, 2015, pp. 3–6. [Online]. Available: http://doi.acm.org/10.1145/2802558.2814645

[4] I. Karydis, A. Gkiokas, V. Katsouros, and L. Iliadis, "Musical track popularity mining dataset: Extension & experimentation," *Neurocomputing*, vol. 280, pp. 76 – 85, 2018, applications of Neural Modeling in the new era for data and IT. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0925231217317666

[5] S. Arakelyan, F. Morstatter, M. Martin, E. Ferrara, and A. Galstyan, "Mining and forecasting career trajectories of music artists," in *Proceedings of the 29th on Hypertext and Social Media*, ser. HT '18. New York, NY, USA: ACM, 2018, pp. 11–19. [Online]. Available: http://doi.acm.org/10.1145/3209542.3209554

[6] D. M. Steininger and S. Gatzemeier, "Using the wisdom of the crowd to predict popular music chart success," in *Proceedings of the 21st European Conference on Information Systems*, 2013, p. 215.

[7] M. O. Silva, L. M. Rocha, and M. M. Moro, "Collaboration profiles and their impact on musical success," in *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*, ser. SAC '19. New York, NY, USA: ACM, 2019, pp. 2070–2077. [Online]. Available: http://doi.acm.org/10.1145/3297280.3297483

[8] C. V. Araújo, R. M. Neto, F. G. Nakamura, and E. F. Nakamura, "Using complex networks to assess collaboration in rap music: A study case of dj khaled," in *Proceedings of the 23rd Brazillian Symposium on Multimedia and the Web*, ser. WebMedia '17. New York, NY, USA: ACM, 2017, pp. 425–428. [Online]. Available: http://doi.acm.org/10.1145/3126858.3131605

[9] D. Herremans, D. Martens, and K. Sörensen, "Dance hit song prediction," *Journal of New Music Research*, vol. 43, no. 3, pp. 291–302, 2014. [Online]. Available: https://doi.org/10.1080/09298215.2014.881888

[10] M. Reiman and P. Örnell, "Predicting hit songs with machine learning," 2018.

[11] V. Dhar and E. A. Chang, "Does chatter matter? the impact of user-generated content on music sales," *Journal of Interactive Marketing*, vol. 23, no. 4, pp. 300 – 307, 2009. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1094996809000723

[12] B. Shulman, A. Sharma, and D. Cosley, "Predictability of popularity: Gaps between prediction and understanding," in *Tenth International AAAI Conference on Web and Social Media*, 2016, pp. 348–357.

[13] J. Lee and J. Lee, "Music popularity: Metrics, characteristics, and audio-based prediction," *IEEE Transactions on Multimedia*, vol. 20, no. 11, pp. 3173–3182, Nov 2018.

[14] M. Interiano, K. Kazemi, L. Wang, J. Yang, Z. Yu, and N. L. Komarova, "Musical trends and predictability of success in contemporary songs in and out of the top charts," *Royal Society Open Science*, vol. 5, no. 5, p. 171274, 2018. [Online]. Available: https://royalsocietypublishing.org/doi/abs/10.1098/rsos.171274

[15] 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*, 2015, pp. 18–25.

[16] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of Computer and System Sciences*, vol. 55, no. 1, pp. 119 – 139, 1997. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S002200009791504X

[17] T. Hastie, S. Rosset, J. Zhu, and H. Zou, "Multi-class adaboost," *Statistics and its Interface*, vol. 2, no. 3, pp. 349–360, 2009.

[18] H. Zhang, "The optimality of naive bayes," *AA*, vol. 1, no. 2, p. 3, 2004.

[19] C. Manning, P. Raghavan, and H. Schütze, "Introduction to information retrieval," *Natural Language Engineering*, vol. 16, no. 1, pp. 100–103, 2010.

[20] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, Oct 2001. [Online]. Available: https://doi.org/10.1023/A:1010933404324

[21] R. Giusti, D. F. Silva, and G. E. A. P. A. Batista, "Improved time series classification with representation diversity and svm," in *2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA)*, Dec 2016, pp. 1–6.

[22] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the support of a high-dimensional distribution," *Neural Computation*, vol. 13, no. 7, pp. 1443–1471, 2001. [Online]. Available: https://doi.org/10.1162/089976601750264965

[23] R. Herbrich, *Learning kernel classifiers: theory and algorithms*. MIT press, 2001.