STAT 451: Predicting Spotify Song Popularity with Multiple Machine Learning Models

Anantha Rao anrao3@wisc.edu
Desmond Fung dfung2@wisc.edu
Xingrong Chen xchen942@wisc.edu

Abstract

With the increasing popularity of streaming platforms for music, understanding the music industry has become more about understanding the listening behavior on these platforms. Here, we sought to fit machine learning models which could accurately predict if a song is popular. We used a data set from songs on Spotify with 13 Spotify-defined features for each of 19,000 songs and their relative popularity on Spotify. We also tested for feature importance in predicting popularity to get a sense of what kinds of acoustic features of music make it popular. The project then goes to fit several models with different hyper parameters and ranked their performance.

1. Introduction

Would we know a popular song when we hear one? While we may possess our own opinions about which songs are popular or catchy, how might we statistically predict which songs will be popular? One of the many facets of human life where there is an abundance of data is in music. In this project, we use a Spotify data set with 19,000 songs to uncover the relationship between the features of a song and the song's popularity. From this analysis, we've trained and adapted many machine learning models to predict a song's popularity.

Additionally, as a side-effect of this study, we learn something more generally about human listening behavior in the realm of music consumption. We'd learn what kind of attributes of a song appeal to people in aggregate, and gain some insight into the sounds in music that naturally attract positive attention.

Musicians may even gain insight into what the consumer listens to most intently and this could inform music production as well as distribution. If our model is interpretable, we could tell the precise effects of each feature on determining the popularity of a song. This would greatly help in forming general theories or 'rules of thumb' about music when it comes to writing music.

Thanks to the growing popularity of streaming platform such as Spotify and Apple Music, data can now be easily reached and user behaviors are accessible for research. Using a variety of machine learning techniques including Logistic regression, Decision Trees, Random Forest, and the kNN algorithm, we can now address the questions of what makes a song popular and examine the largest component of a song's success.

Our research will examine Spotify-defined features related to the musicality and sound qualities of songs. Features related to the sound such as song duration, acousticness, loudness, tempo, etc. are the things we are interested in here; as well as the connection they have (if any) to song popularity. If there is a connection between these features and popularity, we hope to identify a model or an ensemble of models which will both predict the next big hit, but also bring light to the attributes of a song that humans find most attractive.

2. Related Work

The first piece of related work comes from Mohamed Nasreldin, Stephen Ma, Eric Dailey, and Phuc's Dang[6] study on whether certain characteristics of a song will influence a song's popularity. Their study focuses on the influence of a song's audio features, artist features, and song related features. In their research, songs that appeared on the Top 100 Billboard at least once were labeled as 'popular'. F1-scores was then used for each feature to identify the importance of features in their data set. They found technical features such as tempo, mode, share the same importance as artist features such as familiarity. By training many machine learning algorithms like Random forest, kNN, XGboost, decision tree, support vector machine, and tuning their hyper-parameter settings with grid search, they found that the algorithm with the best performance only achieves 63% test set accuracy (shown in Figure 1). It is not uncertain if it is possible to predict a better result than random guess a song will be popular or not. However, predicting an unpopular song is much easier because the chosen data set consists of mostly unpopular songs.

Other studies have found some features and attributes about songs that do and do not predict popularity. One such

	Model
Score	
0.632412	XGB
0.617116	Logistic Regression
0.611623	Random Forest
0.542687	KNN
0.520234	Decision Tree
0.512634	Support Vector Machines

Figure 1. Results from Mohamed Nasreldin, Stephen Ma, Eric Dailey, and Phuc's study [6]

study examined the impact of lyrical content on popularity. They found, after some digging that lyrics have little to no power in predicting popularity[5]. This same study seemed to suggest though, that the song's genre is an important predictor in how popular it will be.

Additional studies have found significant predictive effects at different times of year. For example, a study at Stanford[3] found seasonal effects; discovering that while summer months(June- August) don't tend to produce more popular music than holiday months(November - January) it appears the features of popular music are slightly different between the two seasons[3]. They also discovered that popular songs in roughly the same time period are more similar than popular songs before and after that time period; suggesting that the features of popular songs tend to evolve together over time.

3. Proposed Method

In this project, our group proposed to build different machine learning models in order to make predictions on a song's popularity, given a data set provided by Edwin Ramirez on Kaggle [1]. It contains 19,000 songs that were released before 2018 as well as 15 unique Spotify-defined features like loudness, dance-ability, and acousticness etc that we will be analysing.

Since we had a large data set, we decided to use a hold-out method of tuning hyper-parameters. Accordingly, we split our data into training and testing sets using the "train test split" method from the sci-kit learn package. The original range of values for "song popularity" in our initial data set ranged from 0 to 100. However, to convert this response variable to a binary classification of 'popular' and 'unpopular' we decided to re-label it such that any song in the upper 20th percentile of popularity was labelled as popular(1) and the rest as unpopular(0).

A successful project here would be to find a model or an ensemble of models which can reasonably and accurately predict the popularity of a song. We hope that we can even supersede the accuracy of the models and methods used in the standard kaggle notebook shown above. Here, we've randomized our training and test sets and achieved good training accuracy without over fitting.

We also sought to have some amount of of interpretability in our model so as to gain a more human understanding of the relevant features that make a popular song. Then, rather than having a black box predictor which only tells us "popular" or "unpopular", we have some level of understanding as to why a song isn't popular on a feature level. In this way, the creation of an interpretable model would be useful to consumer scientists/behavioral analysts because the models could be leveraged to develop theories of listening behavior. With more data and expertise in those areas, a model produced here could inspire prediction of overall music trends over the years. To this end, we will analyze feature importance with several different models: XGboost and RandomForest. This will be discussed in more detail later.

Before we formally define model success and loss, we should first re-emphasize that we'll be converting the interval variable 'song popularity' to a class label with 1 and 0 as defined as songs with ratings $\geq$ 72 and $<$ 72 respectively. Then to quantify success in the aforementioned areas, we would first examine training accuracy between our predictions and class labels. We plan on using grid search to evaluate the model at a variety of different hyper parameter settings. We could then run a test set through our model to evaluate its prediction error.

Various model accuracy and feature importance (using XGBoost and Random Forest) were computed, where the accuracy is the number of accurate predictions we make compared to the true label of the track. The F score computed from XGBoost allows us to determine which feature is promising to the model accuracy. We may consider our project to be successful if we achieve more than 80% test accuracy on any of the machine learning models and the F score is at least 80. Feature importance was also estimated using the "feature importance permutation" function from mlxtend to check if we obtained consistent results from random forest.

In summary, we will be using Logistic regression, KNN, Decision Trees, Bagging, and Random Forest all with and without a formal Grid Search cross validation process for model training. We split the data such that 85% was allocated to the training set and the remaining was the test set. We then evaluated feature importance using XGboost and Random Forests to see which of the acoustic properties of songs tends to characterize popular music on Spotify.

4. Experiments

The Logistic regression, KNN, Decision Trees, Bagging, and Random Forest models were all tested in a similar man-

ner to evaluate their abilities on making accurate predictions of a song's popularity. Using sci-kit learn and mlxtend, which are discussed later, we trained each model using their respectively hyper-parameter settings.

For this project, our group is particular interested in Random Forest, given its ability to produce predictive model for both classification and regression problems, and the fact that little pre-processing needs to be done to obtain an accurate accuracy score.

4.1. Dataset

Our data set consisted of 19,000 songs on Spotify with information about song title, composer, the playlist it's on, as well as Spotify defined features for auditory qualities of the song. An example of what our data looks like is shown below:

	song_name	song_popularity	song_duration_ms	acousticness	danceability	energy	instrumentalness	key	liveness	loudness	audio_mode	speechiness
0	Boulevard of Broken Dreams	73	262333	0.005520	0.496	0.682	0.000029	8	0.0589	-4.095	1	0.0294
1	In The End	66	216933	0.010300	0.542	0.853	0.000000	3	0.1080	-6.407	0	0.0498
2	Seven Nation Army	76	231733	0.008170	0.737	0.463	0.447000	0	0.2550	-7.828	1	0.0792
3	By The Way	74	216933	0.026400	0.451	0.970	0.003550	0	0.1020	-4.938	1	0.1070
4	How You Remind Me	56	223826	0.000954	0.447	0.766	0.000000	10	0.1130	-5.065	1	0.0313

Figure 2. An overview of our data set

Our manipulation of the original data set consisted only of scaling the features to be binary class labels where 1 marks a song_popularity score $\geq$ 72 (80th percentile and up), whereas 0 marks a song_popularity score $<$ 72 (below 80th percentile).

The original data can be found by clicking here. To get a better sense of what our data looks like, a correlation matrix, shown in figure 3, is constructed to evaluate which feature needs further analysis. Negative correlation suggests features have an inverse relationship. So if one feature increase, the other feature decreases. A strong positive correlation between energy and loudness and a moderate negative correlation between acousticness and loudness were observed. Other than that, no feature indicates a clear relationship with song popularity.

4.2. Software and Hardware

The models implemented in this project were carried out using Jupyter Notebooks running Python 3.8 or later on each of our own laptop. The primary python libraries we used include numpy and pandas to clean and tidy data, matplotlib for general plotting, and scikit-learn and mlxtend for building a wide range of machine learning models.

5. Results

We first use the Logistic regression with 12 penalty and this model achieve 81.7% accuracy on the test set. Due to the skewed data set,however, this result is not quite useful.

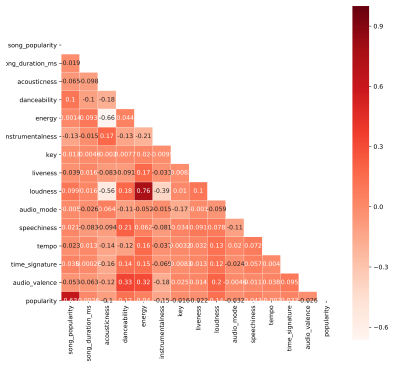

Figure 3. Features' of Correlation Plot

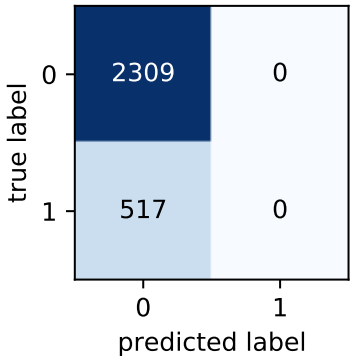

Figure 4. Confusion Matrix of Logistic Regression

This is especially apparent when we look at the confusion matrix for this model:

Because our model predicted all the song as unpopular, Logistic regression will always achieve approximately 80% accuracy due to the fact that 80% of the songs are classified as unpopular. However, Logistic regression is still useful because it tells us that some features, shown in table 1, are significant in predicting the popularity of songs.

features	p-value
acousticness	2.66e-07 ***
danceability	$<$ 2e-16 ***
energy	$<$ 2e-16 ***
instrumentalness	$<$ 2e-16 ***

Table 1. Part of significant features

Second, we fitted a KNN model to the training set with K=5. This model obtained an accuracy of 81.63% on the test set.

A plot of accuracy against k values is provided in figure 5, where we attempt several "k" values. While only achieving 81.6% accuracy with k = 5, when using grid search we found the number of neighbors which produces the best ac-

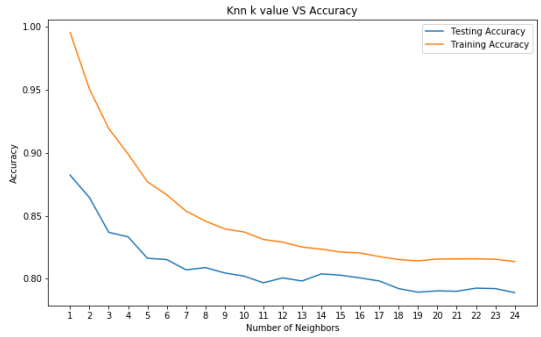

Figure 5. kNN performance accuracy on the test set with different number of neighbors

curacy is 1 with a test set accuracy of 88%.

We then fit an unpruned Decision Tree and found that the classification accuracy was 86.54%. It became clear however, that performance hit a ceiling after reaching a certain depth; as shown in graph 6 below:

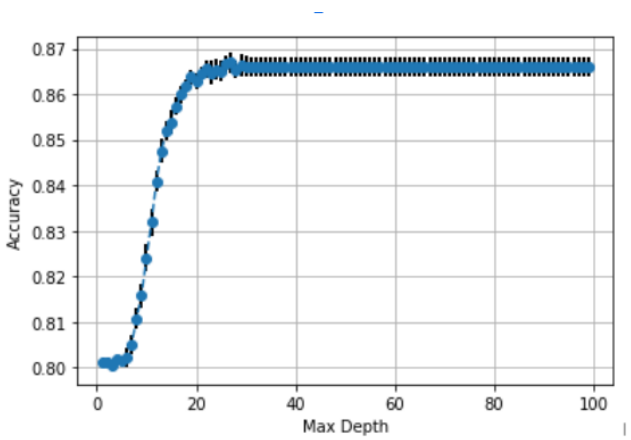

Figure 6. Decision Tree performance accuracy on the test set with different max_depth

So after using Grid Search and enumerating the combinations of hyper parameters and performance accuracies, we deemed it plausible to reduce maximum depth to 30 with a split criterion of entropy since there was minuscule difference in accuracy.

After trying these standard machine learning models, we then turned to ensemble models in order to improve testing accuracy above 90%.

The first ensemble method that we tried using involved a bagging method with decision trees of depth 15 for each estimator and 500 total estimators. With this model, we achieved 92.5% accuracy on the test set. Increasing the number of estimators could surely increase the accuracy rate. However, we discuss later our reasoning for capping our bagging model at 500 base estimators.

The second ensemble method employed was Random Forest Model. For this model, we decided to use 250 base estimators. Evaluating this model on the test set reveals an even better performance accuracy of 94%. With such close performance of the two ensemble models, we decided to further investigate to see if there is detectable difference between these two generalization accuracies.

Our Random Forest and Bagging with Decision Tree methods showed similar performance on the test set. We decided to do further analysis with a McNemar's hypothesis test to evaluate the difference. McNemar's Test is a statistical test to compare the predictive accuracy [4] between two models. Given that random forest and bagging with decision tree have an accuracy of 93.98% and 92.5% respectively, a 2x2 contingency table (shown in figure 5) can provide further insights on whether the difference in performance of the two machine learning classifiers is significant.

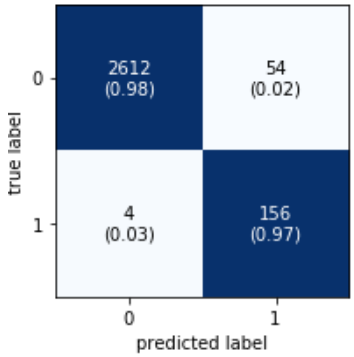

Figure 7. Confusion Matrix between Bagging with Decision Tree and Random Forest

To test the null hypothesis that the predictive performance of two models are equal, using significant level of $\alpha = 0.05$, we proceed with this test for computing the chi-squared and p-value. We obtained a p-value of 1.24e-10. Since the p-value is a lot smaller than our assumed significant threshold ($\alpha = 0.05$), this offers strong support for rejecting the null hypothesis and conclude that the performances of the models are not equal. In particular, random forest outperform bagging with decision tree and is the best model for predicting song popularity.

6. Discussion

These results are interesting, as they suggest that song popularity can indeed be reliably predicted from just auditory features. This plays to the narrative that popular music does tend to sound the same along certain dimensions. These results give credence to the idea that there is in fact some kind of "pop music formula."

For each of the models we selected an arbitrary hyper

parameter to start. Usually, we stuck with what was the default value for the respective classifier model in sci-kit learn. Due to our large data set there was sufficient testing data on which we could reliably estimate each model's true generalization performance.

Logistic Regression doesn't provide meaningful insight in this task. This is because our data was skewed towards unpopular songs. Since the relationship between the probability of a song being popular and song's auditory features are not linear, the assumptions of this model may not be valid. Despite this, we could still use this analysis to find the significant impact of certain features using the hypothesis test result from Logistic regression.

KNN is one of the simplest classification algorithms that can be used to solve both classification and regression problem. It can give highly competitive results even with such simplicity and we expect it to perform better than our logistic regression model. One of the most important tasks in kNN is choosing the correct number of neighbors. A major drawback of KNN is that it is particularly susceptible to the curse of dimensionality. With 13 feature values such as energy, liveness, etc. in our dataset, kNN will take a large portion of the hyper volume into consideration to find k nearest neighbor and hence this affects the trust-ability and potentially the accuracy of this model.

Regarding our Decision Tree Model, we settled on a maximum depth of 30. In this situation, a depth of 30 yielded exactly the same accuracy as an unpruned tree. This led us to conclude that the tree may have hit the bottom of the data set early on or that performance maxed out.

Concerning out bagging ensemble model, we selected the hyper parameters with a different criteria than the other models. To select the number of estimators for this ensemble we each ran this bagging classifier on our computers and tried a different number higher and higher until we reach a point where we had the highest number of estimators that could be completed in under 2 minutes. So this hyper parameter decision was based on a time consideration.

We do realize that the choice of 2 minutes is rather arbitrary. Also, each of our computer set ups has a different computational power and will have slightly different timings for the same computation. However, we wanted to consider time as a factor in determining the hyper parameters, and while we could aim for a shorter time period, we figured 2 minutes is an arbitrary but reasonable amount of time for the accuracy that the model afforded.

Finally, random forest is by far the best machine learning model we were able to fit, followed by bagging with decision tree, which align with what we have learned in class. The reason why random forest worked better than a regular bagging model can be explained by the introduction of splitting on a random subset of features that further diversifies the individual tress. Essentially, random forest are bagged decision trees that split on a random subset of features on each node.

Below we have displayed a chart of the performances for each of our models. In all, we were able to increase the performance of each next model from the last one.

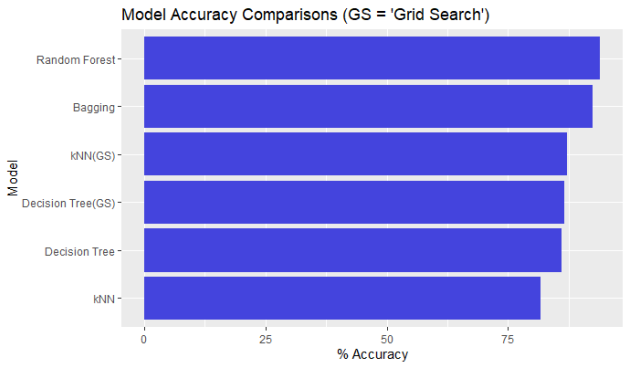

Figure 8. Each model's performance accuracy on the test set

We suspect there could have been even more improvement with the ensemble models had Grid Search been employed. However, the ensemble models already took a lot of time to train, and for the bagging with decision trees model, there were already considerations of time that we'd implemented. Also, we excluded logistic regression from this chart, because it predicted a song to be unpopular every time. Therefore this model's accuracy would merely be the natural distribution of popular and unpopular songs and would be exactly 80% accuracy.

6.1. Feature importance

In addition to tuning a good predictive model, we also wanted to have some amount of interpretability. Knowing which auditory features correspond to popularity helps give a rule of thumb for what kinds of sounds attract masses of people. In order to get deeper insight into the types or classes of songs which are popular. We conducted feature importance analysis with both XGBoost and Random Forest to answer this question.

Here we have the feature importance as determined by the XGBoost algorithm:

According to this figure, the features most important in predicting song popularity are tempo, speechiness, acousticness, loudness, and audio_valence. We were somewhat surprised to see 'tempo' so high up in importance since tempo is merely the rate of a song (beats per minute). We noted in our further analysis of feature importance that this effect did not persist. However, the other features seemed reasonable. One thing to note however, is that this importance metric does not necessarily indicate in which direction the relationship goes. This is to say that, we do not know just from this whether or not a more acoustic song means more popular. We further inquired into feature importance

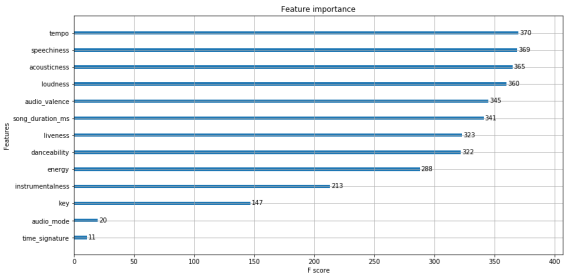

Figure 9. Each model's performance accuracy on the test set

by evaluating it again, but this time using the Random Forest algorithm.

We computed the feature importance bar plot via the "mean impurity decrease" strategy implemented in mlxtend. All features important values were normalized so that they sum up to one. We found that our newly obtained bar plot shared some common important features with the XGBoost algorithm. As we can see in figure 10, acousticness (feature 1), loudness (feature 7), and audio valence (feature 12) are estimated to be the most informative ones in both random forest classifier and XGBoost algorithm. Subsequently, we implemented feature importance via the "permutation importance approach". As we saw in figure 11, acousticness (feature 1), loudness (feature 7), and audio valence (feature 12) are also predicted to be important here, which is consistent with the feature importance values computed earlier. [2]

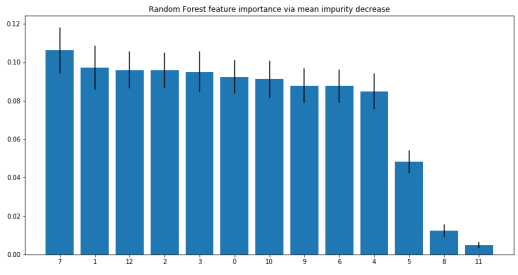

Figure 10. Random Forest feature importance via mean impurity decrease

7. Conclusions

In this paper, we sought to fit a machine learning model which could accurately predict if a song would be popular or not based on the auditory features of the song. We examined the ability of several models in doing this as well as applied logistic regression to get a preemptive sense of which of the features significantly affected popularity. Due to our large data set, we did not encounter many issues with estimating the a model's accuracy in predicting popular songs.

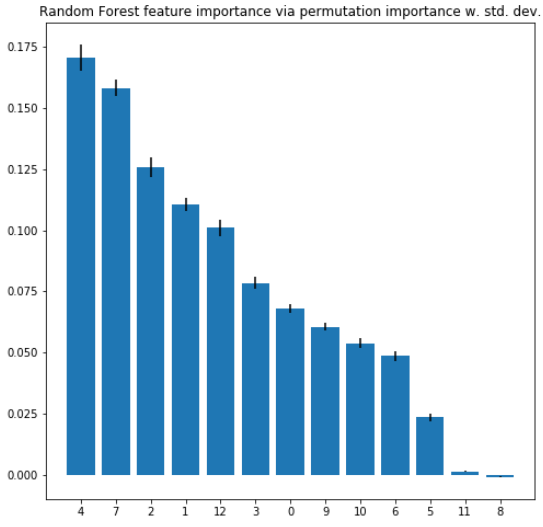

Figure 11. Random Forest feature importance via permutation importance w. std. dev

When conducting a McNemar test between our two best performing models, we found that, indeed, the Random Forest model was the best predictor of song popularity.

We were also interested in the auditory features which were the most important in predicting song popularity. We examined which features were most important with both XGboost as well as Random Forest and found that the features which are the most characteristic of popular music are acousticness, danceability, loudness, audio_valence. So a good rule of thumb for predicting popularity may be, if you can dance to it, it is more likely to be popular.

Our results are promising and interesting. They validate the idea that song popularity is indeed some kind of function of the auditory dimensions of a song and that there is some consistency to this relationship. This gives insight into the genre of 'pop' music as well. Although other work has found that pop music can change over time, this work shows that if pop music does indeed change, it changes together. Pop music coheres to the other popular features of the time.

Future directions that could see even more success with predicting popularity might incorporate non auditory features about a song. For example, the presence of a song on certain social media platforms as well as information about how many monthly listeners the composer of a song has on Spotify. We suspect these would be excellent predictors, because both of these metrics are directly related to the amount of reach a song has.

8. Acknowledgements

Considering the fact that none of us in the group has a machine learning background at the beginning of the semester, it was essential that we acknowledge Professor Raschka's teaching and the opportunity to explore such an interesting topic. Additionally, we would like to recognize Edwin Ramirez for providing the large, reliable data set [1], which allowed us to learn about acoustic characteristics' role in the song popularity.

9. Contributions

Each group member contributed to the modelling and writing process.

Anantha was responsible for fitting and tuning the decision tree and bagging models as well as helping with the McNemar test. He also wrote for each of the writing subsections.

Xingrong was responsible for doing much of the work in logistic regression, contributed to KNN, helped coding up some of the graphs and helped with writing in each portion of the report.

Desmond was responsible for modelling KNN and Random Forest,and did a lot of graph creation. He also helped with McNemar testing, and helped write parts for every section of the report.

10. Code

The code and associated notebooks implemented for this project can be accessed at the following GitHub repository: https://github.com/VanR20/Song_Popularity

References

[1] "19,000 Spotify Songs Dataset". In: (). URL: https://www.kaggle.com/edalrami/19000-spotify-songs?select=song_data.csv.

[2] "Feature Importance Permutation Documentation". In: (). URL: http://rasbt.github.io/mlxtend/user_guide/evaluate/feature_importance_permutation/.

[3] "HITPREDICT: PREDICTING HIT SONGS USING SPOTIFY DATA". In: (). URL: http://cs229.stanford.edu/proj2018/report/16.pdf.

[4] "McNemar's Test Documentation". In: (). URL: http://rasbt.github.io/mlxtend/user_guide/evaluate/mcnemar/.

[5] "Predicting A Song's Commercial Success Based on Lyrics and Other Metrics". In: (). URL: http://cs229.stanford.edu/proj2014/Angela%20Xue,%20Nick%20Dupoux,%20Predicting%20the%20Commercial%20Success%20of%20Songs%20Based%20on%20Lyrics%20and%20Other%20Metrics.pdf.

[6] "Song Popularity Predictor AUC scores". In: (). URL: https://towardsdatascience.com/song-popularity-predictor-1ef69735e380.