PREDICTING THE POPULARITY OF A SONG

CS 334: MACHINE LEARNING

Jonathan Kim Anna Mola Mohammed Alsalamah

ABSTRACT

For our project, we tried to predict whether a song would be popular based on its objective features that were calculated by Spotify, like the key or duration. We used the "Spotify Hit Predictor Dataset" from Kaggle [1] that contains over 40,000 songs. We made predictions using different subsets of the dataset. We predicted whether a song was a hit with about an 83% accuracy, using four different machine learning models. The algorithms we used were: Decision Tree Classifier, Random Forest Classifier, Ridge Classifier, and Logistic Regression. The most accurate model was the Random Forest Classifier with an 83% accuracy.

1. INTRODUCTION

In the music industry, there is often an understood recipe for a perfect "pop" hit. Yet sometimes a song that seems to be the next big thing will flop. Could there still be something that was missing from the song that decreased its chances of succeeding? That's where machine learning algorithms come in. We used the "Spotify Hit Predictor Dataset" [1] to predict whether a song would be a hit on the Billboard Hot 100, based on objectively calculated audio features of the song. The machine learning models output a binary classification for a song that determines whether the song will be a hit or not.

The problem that we are trying to address is the ambiguity that arises from interpretation. Art is highly subjective, and it can often be hard to determine whether a given song could be a hit. As such, it may be useful for music producers and labels to algorithmically determine the potential sales of a track. The goal of this project would be to provide a more objective metric that can be used to predict whether a given song will be a hit or not. Using abstractions of certain song qualities, we hope to predict whether a song will place in Billboard's Hot 100.

2. BACKGROUND

The problem of trying to predict a song's popularity based on its audio features has been explored by others. There are some companies that merely exist to help record labels in the industry churn out guaranteed hits. In New Yorker article "The Formula" [2] from 2006, the author mentions a company called Platinum Blue that has a "proprietary computer program that uses "spectral deconvolution software" to measure the mathematical relationships among all of a song's structural components" and can predict a song's popularity with an 80% rate of accuracy. It's clear that in the last 15 years there is a lot of money and reputation resting on whether an algorithm can correctly identify hits.

There are also more recent academic works that explore and discuss this problem. Adeagbo [3] predicted the success of certain songs in the genre of Afrobeats. Adeagbo was able to achieve an 86% accuracy with the Random Forest and Gradient Boosting algorithms. This

study was successful, but was limited to the Afrobeats genre. This means the songs in the dataset most likely had many audio features in common with each other.

Additionally, Araujo [4] attempted to find an accurate model for predicting music popularity on streaming platforms. They built an SVM classifier with an RBF kernel that obtained accuracy above 80%. What is most notable about this academic study is that they used audio features in combination with social network information, and concert and festival data. They justified using these extra data sources by citing work in the field that had used those data sources separately and had arrived at relatively accurate predictions.

3. METHODS

In order to predict the success of a song, we used four different machine learning models: Decision Tree Classifier, Random Forest Classifier, Ridge Classifier, and Logistic Regression.

Decision Tree Classifier

As described in class, the decision tree is a machine learning model that uses gini/entropy as a measure of information gain. The decision tree greedily selects the feature that results in the largest information gain and splits the incoming data on said feature. Any inputted data whose values match the data on the left-hand side of the split will go to the left-hand side, and vice-versa for the right-hand side. This decision process is repeated nlog(n) times, with n representing the size of the original dataset. The result of this splitting is a tree used to assign a label to a given input.

Random Forest Classifier

A Random Forest Classifier is similar to a decision tree, but rather than just using one tree, it uses an ensemble of many different decision trees (making a forest). The Random Forest Classifier trains each decision tree on a sublist of the original data, which is formed by picking data points at random (with repetition) from said dataset. Inputted data are then assigned predictions using the majority label found in decision trees that do not contain the given data point.

Ridge Classifier

The Ridge Classifier is based on Ridge regression. This type of regression is commonly used when there are many predictors or the predictors have a high multicollinearity. The classifier converts the target values into either -1 or 1 based on the class in which it belongs to. Ridge regression is also very similar to least squares. Ridge regression performs L2 regularization. The cost function for ridge regression is:

$$(y - X\beta)^T(y - X\beta) + \lambda\beta^T\beta$$

Logistic Regression

The Logistic Regression model is used to analyze the relationship between multiple independent features and one binary target variable. Logistic regression is used to find the connection between the given features and how likely the outcome is given those features. It is a model that is best suited for linearly separable data. Unlike perceptrons which assign a binary

label absolutely, Logistic Regressions model the *probability* of a given label being the correct one.

4. RESULTS

Data Description

The dataset is titled “The Spotify Hit Predictor Dataset (1960-2019)” [1]. It is a database of over 40,000 hit songs from the last 6 decades, each uniquely identified by 19 features. The features for each song were fetched using the Spotify Web API. These features include: danceability, energy, mode, speechiness, acousticness, instrumentalness, liveness, valence, key, loudness, tempo, duration is milliseconds, time signature, time stamp of chorus hit, and sections. The first are 8 features represented by a value from 0 to 1. Tempo, duration, time signature, key, chorus hit, and sectionsA song is determined as a “hit” if it landed within the Billboard’s weekly top 100 during its lifespan. The data is split up into 6 .csv files, with each .csv representing the hit songs of a given decade. The distributions for the numeric features are shown in Figure 2.

Preprocessing

We used sklearn’s StandardScaler to normalize our data for use in our Logistic Regression model. The StandardScaler remaps the original dataset so that the average falls at 0. Additionally, we noticed that some of the dataset included duplicates. This is due to the fact that some songs remained popular over many years, and therefore qualified as a hit over multiple decades (and the dataset is separated by decade).

Feature selection

We used a Pearson correlation matrix (Figure 1) to determine which features to filter out when inputting new data. A visualization was created using Matplotlib and the optimal features were selected using it. The features “sections” and “duration_ms” had high multicollinearity because their correlation score was greater than 0.85. The feature “sections” was removed from the data.

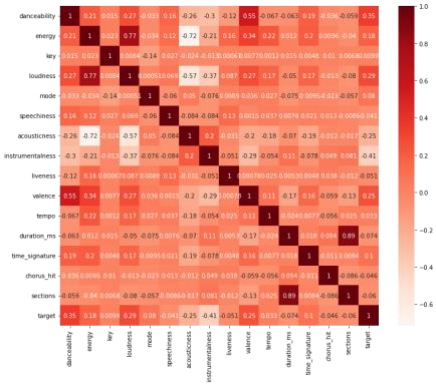

Figure 1: Pearson correlation matrix for the full dataset

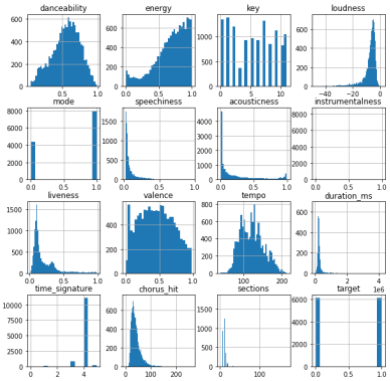

Figure 2: Distributions of numeric features

Modeling Choices

We trained our models on a training set that was 70% of the data and a test set that was 30% of the data.

- Decision Tree Classifier- We selected the decision tree classifier as, although it wasn't included in most of the research we ran into, we felt confident in our knowledge of the model itself due to what we learned in class. As such, we believe training the model and seeing the results would give us a good baseline of how to interpret and optimize other models we may not be as familiar with.
- Random Forest Classifier- On top of the benefit offered by prior knowledge(given that we needed to make a naive version of a Random Forest in class), Random Forest also performed the best in many of the prior papers we looked at. These two advantages made picking this model an easy choice.
- Ridge Classifier- This model is ideally suited for binary labels, which is what we need for our dataset. We picked this model for many of the same reasons as we did Logistic Regression.
- Logistic Regression- Similar to the decision tree, this model was described in detail during class, allowing us to know how to tune the model in an ideal way. This model was also used on some of the papers we looked at, and much like Random Forest, it yielded favorable results.

Initially, we found the baseline accuracy for each of the models with the standard parameters. We used these metrics to compare the improvement our optimization attempted to make on the baseline models. We did this by running a Grid Search Cross Validation on every model with 5 k-folds. Parameters are automatically determined by GridSearchCV, which takes in an sklearn model along with an array of parameters to run through. The optimal hyperparameters are selected from the array entries that produce the best/most accurate results.

Empirical Results and Comparisons

Baseline accuracies for different decades:

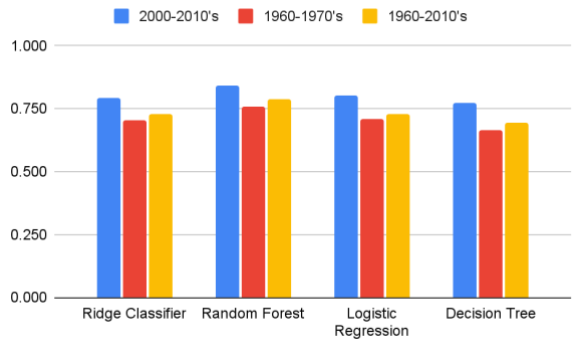

Figure 1: Bar chart of baseline accuracy scores on test data

	2000-2010's	1960-1970's	1960-2010's
Ridge Classifier	0.792	0.704	0.726
Random Forest	0.842	0.760	0.786
Logistic Regression	0.802	0.707	0.729
Decision Tree	0.771	0.666	0.695

Figure 2: Bar chart of baseline accuracy scores on test data

Optimized accuracies for different decades:

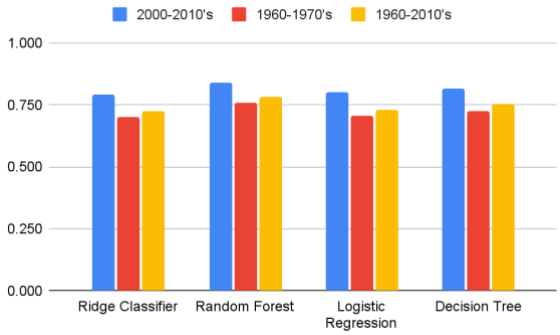

Figure 3: Bar chart of optimized accuracy scores on test data

	2000-2010's	1960-1970's	1960-2010's
Ridge Classifier	0.793	0.700	0.726
Random Forest	0.839	0.757	0.781
Logistic Regression	0.802	0.707	0.729
Decision Tree	0.816	0.727	0.756

Figure 4: Table of optimized accuracy scores on test data

The figures below are all based on the accuracy of the subset of songs from the 2000s to the 2010s. This subset of songs proved to have better accuracy and is more relevant to current trends.

The Ridge Classifier gave a 79.3% accuracy on the test set after being optimized.

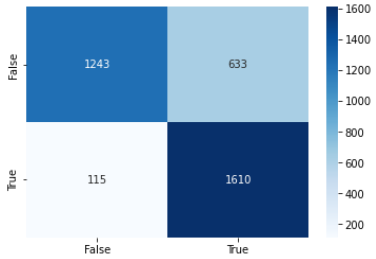

Figure 5: Ridge Classifier Confusion Matrix

The Random Forest Classifier gave an 83.9% accuracy on the test set after being optimized.

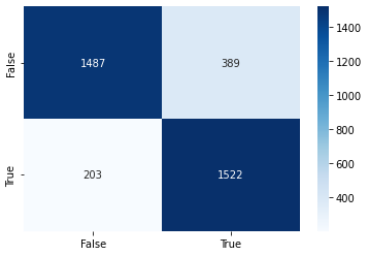

Figure 6: Ridge Classifier Confusion Matrix

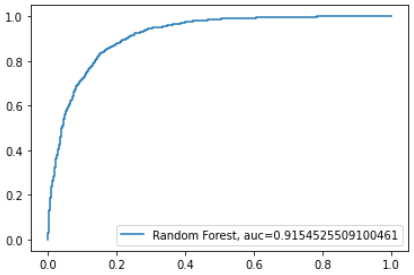

Figure 7: Ridge Classifier ROC Curve

Logistic Regression gave an 80.2% accuracy on the test set after being optimized.

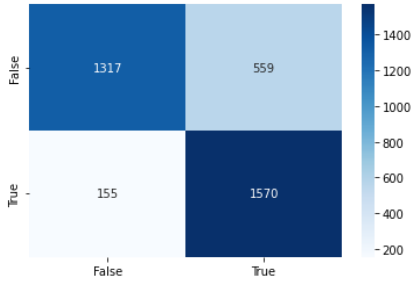

Figure 8: Logistic Regression Confusion Matrix

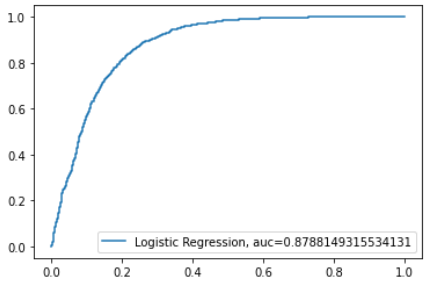

Figure 9: Logistic Regression ROC Curve

The Decision Tree Classifier gave 81.6% accuracy on the test set after being optimized.

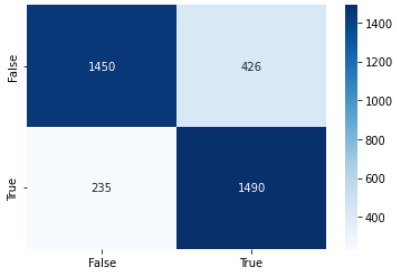

Figure 10: Decision Tree Classifier Confusion Matrix

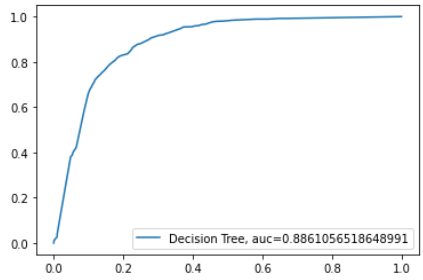

Figure 11: Decision Tree Classifier ROC Curve

DISCUSSION

For our preliminary results, we tested on a smaller subset of our data (data from exclusively the 2000s) in order to speed up the process of training. Our results for that subset were favorable, having achieved accuracies in the mid 80s. When we combined our datasets to include all decades, however, our accuracy dropped a marked amount to 75 percent. We expected this shift, and were still happy with our results.

We initially hypothesized the reason behind this change was because of varying tastes throughout decades--features that would make a hit in the 60s would not necessarily translate to a hit today, and vice-versa. However, this idea was partially debunked after we began testing our data more granularly. Testing a subset of the songs from the 60s-70s and separately the 00s-10s, we expected to find similar accuracies. Instead, we observed significant differences in the results, with the accuracy for the more recent decades falling around 83 percent while the older decades achieved an accuracy closer to 73 percent. These numbers went against our hypothesis that musical tastes are temporal in nature, as data clustered by time period should retain their accuracy. We do not know with absolute certainty why songs from the 1960s and 70s yielded worse results than more recent decades, but some of our hypotheses include the data being unreliable from that era, Spotify's algorithm not being as accurate for older songs, and hit music being less predictable in the past.

Overall, we found that modern songs can be reliably labeled as a hit or a flop, with about an 83% accuracy. Through the process of iteration and observation, we learned the importance of

testing all hypotheses thoroughly, even when your reasons behind observed data seem sound. For example, when we observed that our accuracy dropped once we included older years, we hypothesized that this discrepancy was caused by variation between time periods. This proved to not be the case as, even after correcting for this idea by separating the dataset into two decade chunks, a decrease in accuracy was still observed when testing 2000-2010s vs 1960-1970s. We would have never realized this if we hadn't segmented the original data in this way. This lesson of conducting more exploratory data analysis/visualization and robust testing of hypotheses is one that we will be sure to take into our future endeavors.

REFERENCES

[1] Ansari, F. The Spotify Hit Predictor Dataset (1960-2019). Kaggle (2020). Retrieved from: https://www.kaggle.com/theoverman/the-spotify-hit-predictor-dataset.

[2] Gladwell, M. The Formula. *The New Yorker* (2006). Retrieved from: https://www.newyorker.com/magazine/2006/10/16/the-formula.

[3] Adeagbo, A. Predicting Afrobeats Hit Songs Using Spotify Data. arXiv.org (2020). Retrieved from: https://arxiv.org/abs/2007.03137.

[4] Araujo, C.V., Cristo, M., & Giusti, R. (2020). A Model for Predicting Music Popularity on Streaming Platforms. RITA, 27, 108-117.