Spotify Machine Learning Project

Dallas Strandell

December 27, 2021

1 Introduction

This document describes the method to pull and model data using Spotify's API. The goal of which is to model a song's audio features and predict the popularity. Two sources were used as inspiration and as a source of code for this project:

https://towardsdatascience.com/what-makes-a-song-great-part-2-e82a44be659c

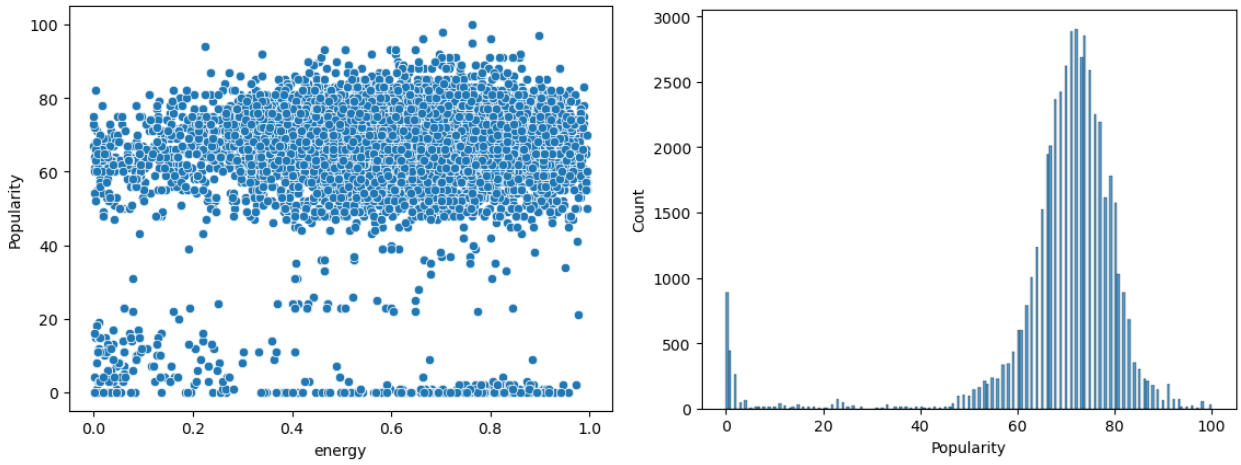
https://github.com/ZipBomb/spotify-song-suggestion

The data was first created using the notebook create_data.ipynb. This involved pulling pseudo-random songs from Spotify using the Spotify API. Audio features were also pulled for each song. This includes the popularity that Spotify calculates for each song. However, many of these audio features are somewhat arbitrary in that they are on a 0-1 or 0-100 scale. Popularity for example is scaled 0-100, and Spotify gives little information on how this is calculated. Initial analysis and decision tree modeling was then done inside data_analysis_3.ipynb. Further modeling was done with XGB_fitting.ipynb. The notebooks have more in depth information on the models.

2 Initial Data Analysis

The first step was to remove any string features such as song and artists titles from the dataframe. plot each of the audio features vs the song popularities. None of the plots showed and obviously trend for predicting popularity. An example plot is seen in Figure 1a. A histogram of the popularity was also plotted (Fig. 1b). These plots combined to show 3 rough groups of popularity (50-100, 5-50 and those grouped near 0). It is possible that removing one or both of the lower popularity groups could increase model accuracy, but

this was not done here. Correlations for each audio feature with the songs' population were also calculated (Table 1). Instrumentalness had the highest total and negative correlation while loudness had the highest positive correlation.



(a) Plot of "popularity" vs "energy" audio features

(b) Histogram of popularity

Figure 1: Initial data analysis plots

Feature	Correlation
year	-0.0908
danceability	0.0186
energy	0.0369
key	-0.0001
loudness	0.1509
mode	0.0281
speechiness	-0.0195
acousticness	-0.0977
instrumentalness	-0.2136
liveness	-0.0184
valence	-0.0767
tempo	-0.0157
duration (ms)	0.0543

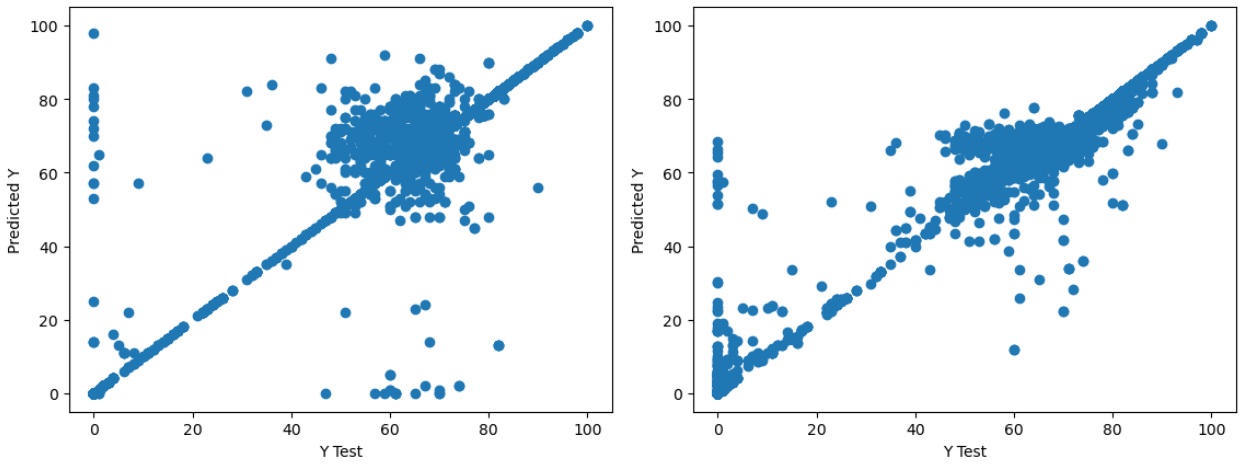
Table 1: Population correlation with audio features.

3 Model Results

Metrics	Decision Tree	Random Forest	Random Forest Optimized	XGB Optimized
R^2 test	0.930	0.958	0.957	0.934
RMSE test	4.130	3.213	3.266	4.021
RMSE train	0.104	1.172	1.291	2.689
test/train ratio	39.7x	2.7	2.5	1.7

Table 2: Selected modeling results

Metrics for selected models can be seen in Table 2. While other modeling methods were used it was decided to use only tree based methods. Decision tree regression was the first method used, and the modeling resulted in a root mean squared error (RMSE) of 4.13 and an R^2 of 0.93. However, the train RMSE is much lower at 0.104. This is a possible sign of overfitting. Random forest regression was also used, and it resulted in a lower test RMSE of 3.213. The training RMSE was also much closer to the test value.

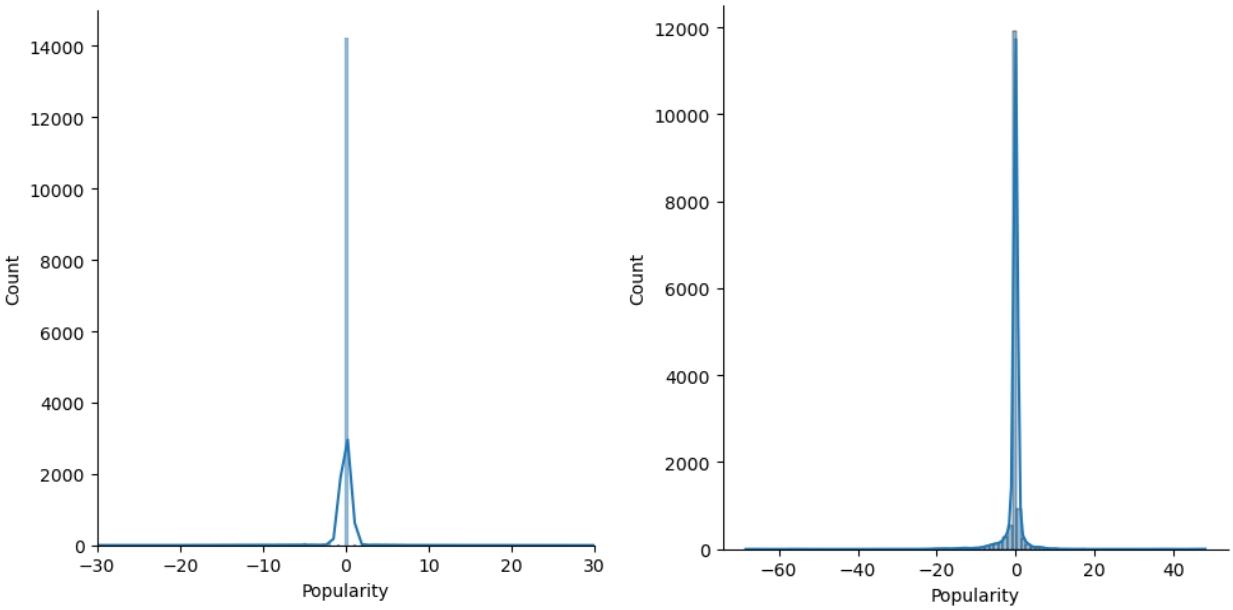


(a) Decision tree

(b) Random forest

Figure 2: Scatter plots of predicted popularity vs test values

Plots of predicted vs test values can be seen in Figure 2. The most striking feature of the decision tree plot in 2a is a thin line along the x=y diagonal line. This is where the predicted and test values are equal. The random forest model has a similar feature but has more near diagonal values. This feature can be more clearly seen in histogram plots of test minus predicted values in Figure 3. These histograms show a large grouping around 0. However, the decision tree only has one visible bin at 0. Yet the RMSE of the decision tree model is lower than the random forest. This is likely due to a higher number of outliers in the DT model.

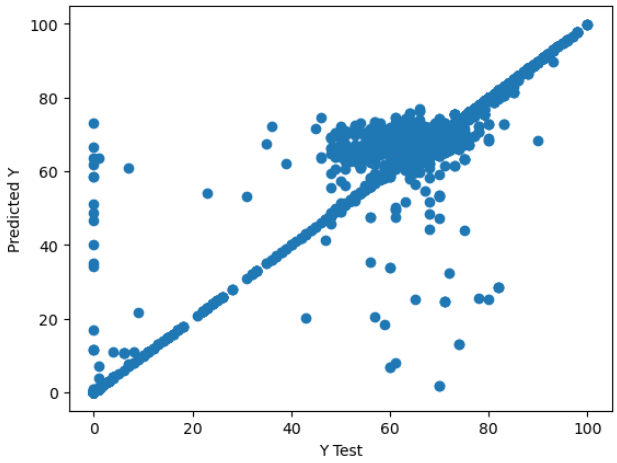


(a) Decision tree

(b) Random forest

Figure 3: Histogram plots of y test - predicted values

An XGBoost regression model was also used. The focus for this model was to lower overfitting. This raised the final model RMSE by approximately 25%. However, this also resulted in the lowest difference between the test and train RMSE. The y test histrogram is between the decision tree and random forest models in terms of the popularity distribution (Figure 5).



(a) Decision tree

Figure 4: Histogram plots of y test - predicted values

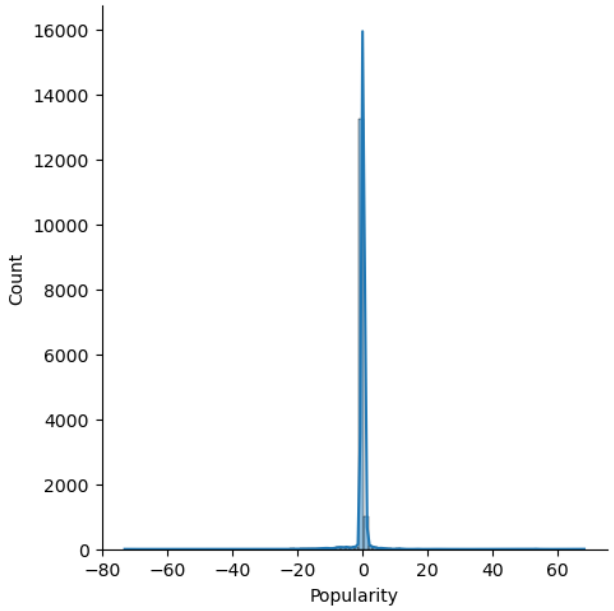


Figure 5: Histogram plots of y test - predicted values

4 Conclusion

The models described in this report can be used to predict the popularity of a song on Spotify. The decision tree model is likely overfit while the XGBoost model is the least overfit. However, the random forest model was the most accurate in terms of test data (cross validation).

5 Appendix

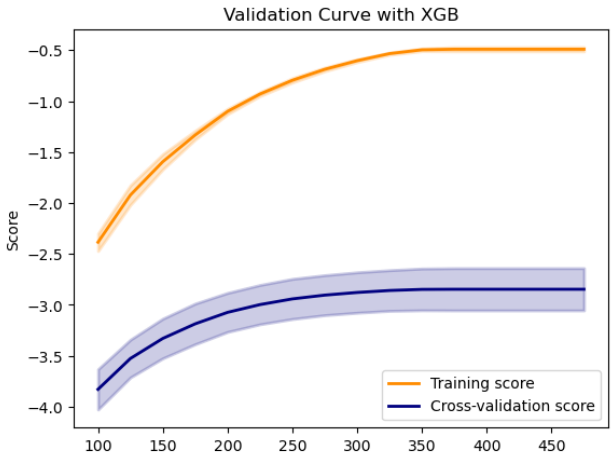


Figure 6: Example cross validation curve of XGBoost regressor. The X axis is the n_estimators hyperparameter while the y axis is the (negative) RMSE model score.

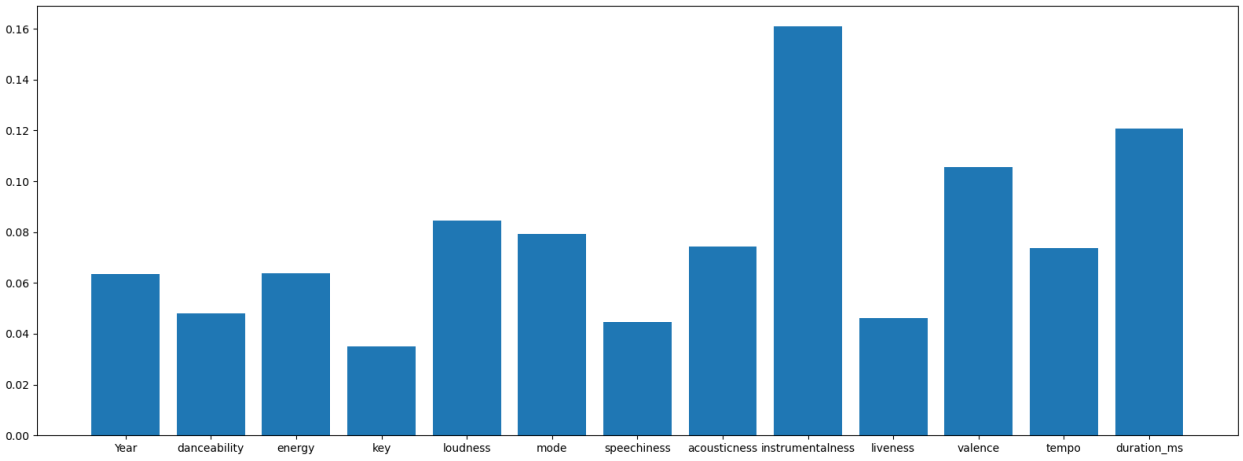


Figure 7: Example feature importance from the decision tree model.