Group 7 - Final Report
CAP5610 Machine Learning | Spring 2022

Machine Learning: Would Your Favorite Song Make it to Spotify's "Top 50 - Global" Playlist?

Group 7 Members:
Bailey LaRea
Nadine Rose
Suma Marri

Problem Statement

The goal of this project is to identify a Machine Learning model to successfully predict if a song will appear on Spotify's "Top 50 - Global" playlist using track attribute data such as track's name, artist name, genre, beats per minute, energy, danceability, loudness (dB), liveness, balance, length, acousticness, speechiness, and popularity.

While developing this project, some of the questions that come to mind are can track attribute data be used to successfully determine popularity? What metrics might Spotify use to generate this playlist? If a track is successfully identified on the Top 50 playlist, the model will be deemed successful.

For the scope of this study, this project aims to look at the 2019 version of this specific Spotify playlist as realistically, the playlist updates in real-time on a weekly basis.

Motivation

The motivation that drives this project is the interest in what classifies a song to be recognized as popular on the Spotify platform. Evidently, their true methods to calculate the popularity of a track are not shared with the public domain. However, by using the track attribute data, this project aims to see if it is possible to use Machine Learning techniques to predict if a song will be placed in a specific playlist.

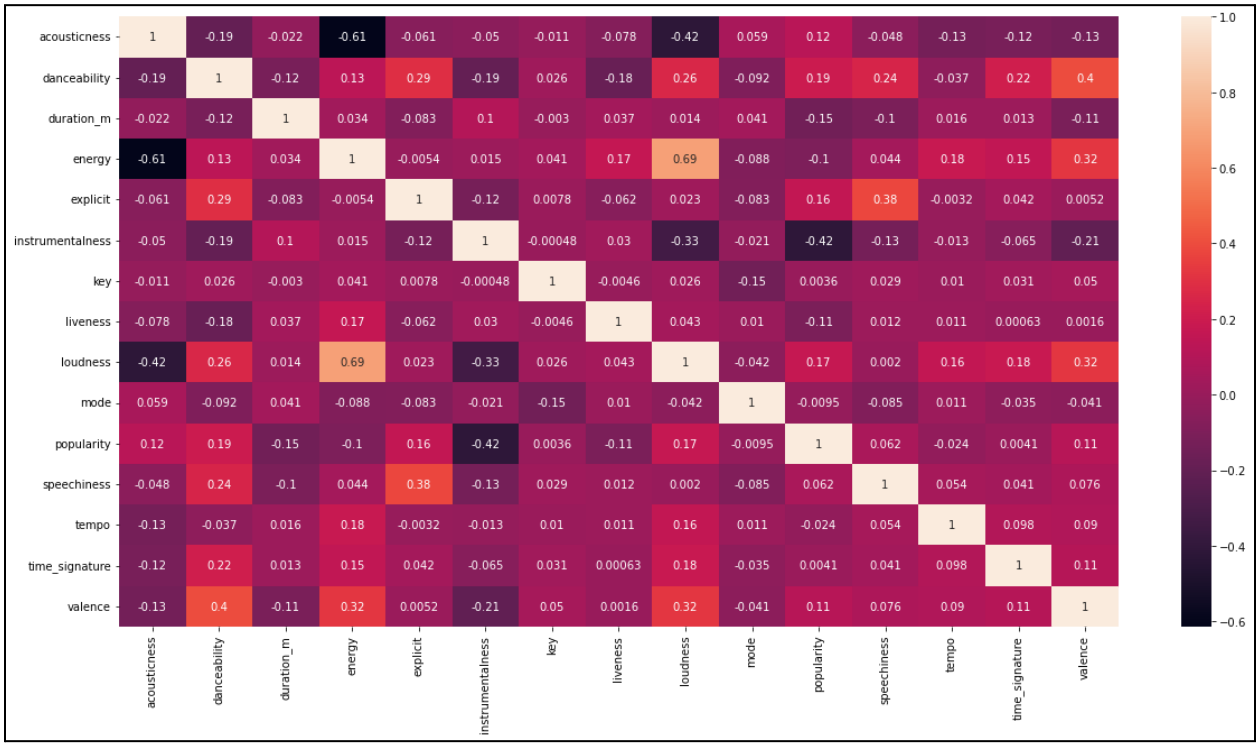
Description of Data

This project works with two different datasets. One dataset contains the 50 tracks in the 2019 "Top 50 - Global" playlist. It contains 13 features and 50 rows, with one row to represent each of the 50 tracks. This dataset was retrieved from Kaggle. The larger dataset contains songs that were released in 2019. This dataset was obtained using the API from Spotify for Developers. It contains 20 features, such as track name and audio analysis, and 10,950 rows.

Each dataset originally has 13 main features: TrackName, ArtistName, Genre, BeatsPerMinute, Energy, Danceability, LoudnessdB, Liveness, Valence, Length, Acousticness, Speechiness, and Popularity.

When evaluating the correlation between each feature, there were no strong positive or negative correlations to the popularity variable. However, there were many features with correlations close to zero and as a result, these features were dropped. These irrelevant features were:

- Time Signature
- Key
- Mode



Modeling Approach

Initially, when starting the project it was unclear whether to frame the project statement in order to use regression or to frame it as a classification type of issue. This is why the work was done with two baselines, for regression and classification models respectively.

Regression Models

A linear regression model was implemented for the baseline of the regression models. The 2019 dataset of songs were split into train and test sets. The size of the test was 40% and the random state of 8. This train test split was used for all regression models.

Linear Regression

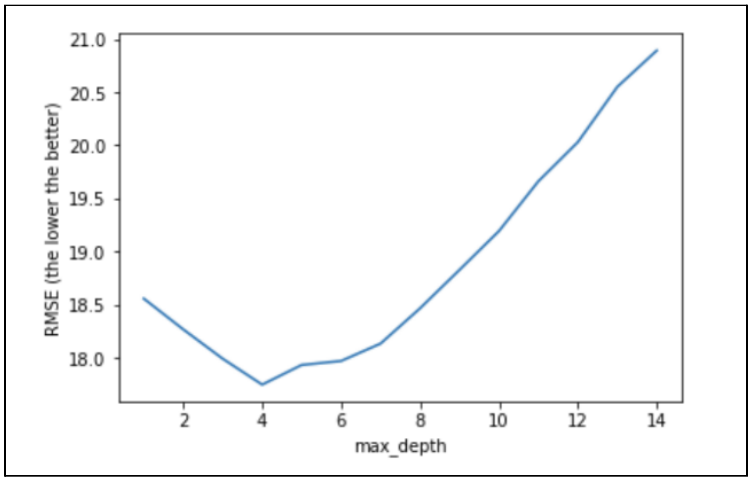
When defining the Linear Regression model, the default parameters were used and the result produced an RMSE score of 18.518. This lower score was particularly concerning and it was believed that the structure of the question might align more with a classification model. As this project aims to successfully classify whether a track will be placed in the Top 50 Global playlist. However, to complete the testing on the effectiveness of regression models, two more models are tested.

Random Forest Regression

For the Random Forest learning model. There were 150 estimators, 5 features, and a random state of 1 applied as the parameters. The result produced an RMSE of approximately 18.2575. The Random Forest Regressor was tested and returned a lower RMSE than the Decision Tree Regressor.

Decision Tree Regression

This model used parameters set as a max depth of 10 and a random state of 1. The Decision Tree Regressor gave good results with a low RMSE of approximately 17.7453 and at a branch/tree depth of 4.



Classification Models

A Decision Tree Classification represented the baseline for the classification models used in this project. The remaining classification models used to compare to the baseline were Random Forest and K-Nearest Neighbor (KNN).

Decision Tree Classification

For a baseline, the decision tree classifier was employed using the default parameters. Through using a Repeated Stratified KFold, the mean accuracy achieved was around 69%. This is a decent score but could be potentially fine-tuned to see improvement.

K-Nearest Neighbor

In order to turn the numerical value into a categorical value, the target variable of popularity was divided into three bins: low, medium, and high. Naturally, the bins are imbalanced. So resampling was applied using an over-sampling method. The KNN model produced an accuracy score of 99%. Although it is a very high score, since it is very close to one it is very likely overfitting occurred with this model.

Random Forest Classification

In the end, the model selected was the Random Forest Classifier as it had the highest accuracy score at around 71%, and the project is based on classification, so it would be unwise to use a regressor. The parameters used for the Random Forest Classifier were:

- n_estimators = 1,000
- max_depth = 5
- random_state = None

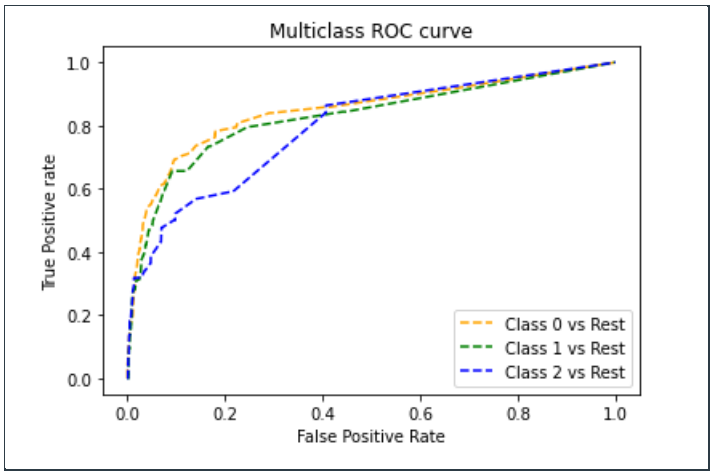
There was a test size of .3 and RandomOverSampler() was used on the training data after it was split.

Project Trajectory

The original approach was that this is a time series problem since it involves prediction, but in reality, it is actually a classification problem since there is no real timeline in which the songs were released. Additionally, the way the songs make it to the Top 50 playlist is not dependent on time. Regression models were wrongly performed to then have three classification ones following it.

Results/Interpretation

At first, it was important to get a high true positive rate so that the model would be able to collect the majority of the top 50 songs. Later, it was realized that it is better to have a lower false-positive rate and a high true positive rate, but not a perfect high positive rate. So, it was decided to create a ROC curve that predicted an effective threshold for the model and have a high true positive rate and a lower false-positive rate. After looking at the Multiclass ROC curve graph, it was easier to decide which threshold to use for the prediction.



The impact this project will have will allow anyone to be able to use their songs and figure out if they can make it to the Top 50 Playlist. It would be interesting to see which of the friends is listening to the most popular music. Predicting the top 50 songs is more difficult than expected. Doing it in a Machine Learning Model was the hardest part as this task was easily completed in "for" loops. This is what was originally done, then it was realized that the model was never used. This meant going back to the drawing board. The final model completed led to the accuracy seen below. One surprise found was how accurate the model was when compared to the "for" loops that were over 90% accurate at predicting which songs would make it onto the Top 50 Playlist.

Conclusion

The lower the threshold, the higher chance the model was able to get the top 50 songs in the predictions, however, there is a high false-positive rate and a low false-negative rate. When the threshold was increased, there were fewer songs in the results, however, the false-positive rate was lower and the false-negative rate was higher meaning the model was more accurate. It was decided that the model was better if there was a higher false-positive than a higher false negative, so the model would recommend a lower threshold when it comes to predicting the top 50 songs. The model successfully predicted the top 50 songs with high accuracy.

Future Work

Originally, the models built were for a time series project, meaning Regression was used. It was quickly determined that this was incorrect and classification models were used going forward. For future work, it would be best to have a more consistent way to find the threshold to where it would return a similar value every time.

Additionally, one of the biggest bottlenecks in this project was trying to work with special characters within the track names. Different characters from other languages influenced the dataset and were not properly read by Python. For the scope of this project, it might have been beneficial to specifically work with one of the US Top 50 playlists to allow for characters that

are easily recognizable by Python. Also, there were issues with the top 50 dataset, for example, one track was released in 2018, and 2 tracks were removed from Spotify. So, when the data was collected on the songs released in 2019 from the API, it couldn't get all the songs in 2019, which increased the false-negative rate. When collecting the main dataset, all the songs were collected that were released in 2019 and at the end of 2018, including deleted songs.

Another idea that could have been implemented is feature importance. It was never applied to the model, although it could provide more insight as to what makes up a popular song when making a prediction. It would also be nice to drop more features so that could hopefully make our model more accurate.

	feature	importance
4	instrumentalness	0.401280
2	duration_ms	0.140301
0	acousticness	0.115545
6	loudness	0.071760
1	danceability	0.063537
8	valence	0.058355
3	energy	0.056643
7	tempo	0.054504
5	liveness	0.038076

Lastly, due to time constraints, the KNN model could not be further investigated. The model showed great performance with a very high accuracy score of 0.99. Evidently, the model seemed to be overfitting due to the very high performance. The investigation of the Decision Tree Classifier model was stopped, although the resulting mean accuracy was slightly lower than the Random Forest model.