Approaches to Genre Classification Using Machine Learning

Alpar Gür, *11134271, alpar.guer@smail.th-koeln.de;* David Henkel, *11156273, david.henkel@smail.th-koeln.de*

***Abstract*—This experiment demonstrates how music tracks' genres can be classified by machine learning models trained with the audio features provided by Spotify. Two classification approaches were pursued and compared against each other. The result shows that *one vs. all* approach and *random forest classifier* are a good fit for recognition of a genre, whereas *multiclass classification* struggles with tracks categorized under multiple genres.**

***Index Terms*—Machine Learning, Genre, One *vs.* All, Multi-class Classification, Ensemble Learning**

I. Introduction

Songs are usually categorized by genre. Unlike identifying shapes, music recognition is a delicate process due to the fact that there are no sharp boundaries between music genres. Furthermore, there are other characteristics that influence the way we perceive music. This paper documents the work that was done related to classification of songs' genres, in the context of Machine Learning (ML) course offered at Technische Hochschule Köln (TH Köln), spring term 2023.

We have trained multiple ML models using a variety of strategies on the tracks and their features we collected from Spotify's Application Programming Interface (API) [1]. As of today, there are more than 6000 registered genre in Spotify [2]. For this experiment, we narrowed the region of interest down to 10 genres. We consciously opted out for well-known distinct genres with the exception of a few of them showing similarities in certain ways. The purpose of this decision is to gain broad insight about models' behaviors. The dataset and Jupyter notebooks of this project are made publicly available on GitHub [5].

II. Data Collection & Preparation

Extract, transform, load (ETL) pipeline consists of three main steps - ***get track names, get track features, handle duplicates tracks***. This section explains each in detail.

A. Get Track Names

To have a balanced dataset, we retrieved 1000 tracks for every selected genre. The Spotify API provides a search to get maximum of 50 items belonging to a genre in a single API call [3]. Retrieved items hold various information about a track (*e.g. name, id, popularity, artists, available markets*). ***id*** is unique track identifier and we used this information to get features of each track.

B. Get Track Features

After curating the track pool, we collected extensive information about sonic characteristics of the tracks. These track features are numeric values prepared based on certain metrics. For instance, *danceability* of a track describes "how suitable a track is for dancing based on a combination of musical elements including tempo, rhythm stability, beat strength, and overall regularity". [4] For detailed description of each audio feature, readers are advised to refer to the official API specification.

C. Handle Duplicate Tracks

It is not uncommon that a track is registered under multiple genres in Spotify's catalogue and a track has multiple versions (radio edit or extended version). These two phenomenons result in duplicate items in dataset. We observed that keeping the first entry of duplicate items leads to an imbalanced dataset, hence removing the duplicates unevenly across each genre. To get over this obstacle, we decided to collect 1000 unique tracks for each genre and keep the duplicates. In total, we collected 19.149 tracks to be used in models' training and testing.

III. Model Training

We chose ***one vs. all*** and ***multiclass*** classification techniques to train variation of ML models offered by scikit-learn [6].

A. One vs. All Classification

This approach requires a binary classifier for each class. Binary classifiers are trained to predict if a track belongs to respective genre or not. Prior to training, we removed attributes irrelevant for this step (*e.g. id, track name*) and performed ***one-hot encoding*** followed by 80/20 train-test split of the dataset with regular shuffling. One-hot encoding transforms categorical data into numerical format. Most of the track feature values lie between 0 and 1. Though, some features like *loudness, tempo, and key* range in bigger intervals. We used ***standard scaler*** to have all track features in the same scale.

We assessed the performance of classifiers by their accuracy on the testing dataset. Majority of them have an accuracy score above 90%. The minimum accuracy is scored by the `DecisionTreeClassifier (DTC)` of the genre *rhythm and blues (R&B)*. Except for `DTC`, the models have jointly scored the best accuracy in the *classical* genre. Overall,

RandomForestClassifier (RFC) outperformed the rest (see **Figure 1**).

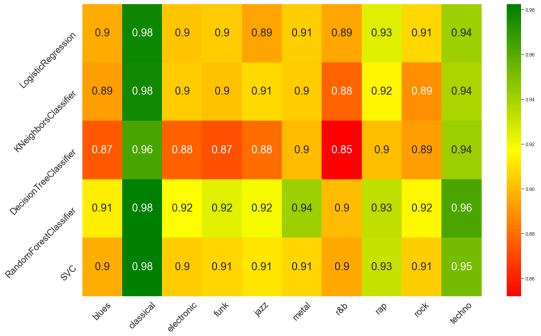

Fig. 1: Accuracy of each model and binary classifiers on the test dataset.

After the initial training, we optimized the hyperparameters of two outstanding models - Super Vector Classifier (SVC), RFC. Tuning was done by ***randomized search***. No improvement on the binary classifiers of RFC was observed. In fact, accuracy score of genres *jazz, electronic* showed extreme tiny decrease. On the other hand, genres *funk, electronic, metal* scored higher by 0.01. Although half of the genres performed worse than initial training of SVC.

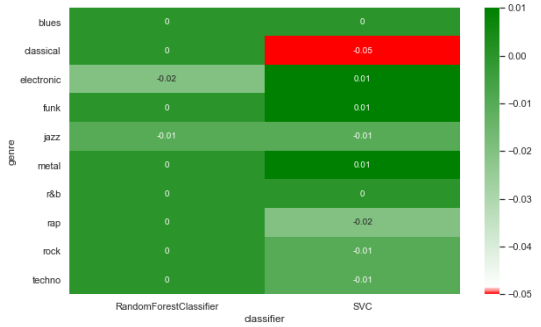

Fig. 2: Accuracy difference of models before and after hyperparameter tuning in percentage.

B. Multiclass Classification

To better evaluate the performance of the one vs. all approach, we trained three models - KNeighborClassifier (KNNC), RFC, DTC to make multiclass predictions. While binary classification deals with two classes, multiclass classfiers are capable of classifying instances into three or more mutually exclusive classes. To make sure that the multiclass classifiers have the same starting point and preprocessing, the dataset with the preprocessing pipeline of before is used.

Each of these classifiers were trained with the training dataset and then evaluated by their accuracy on the testing dataset. The results of the cross validation and accuracy are lower or equal to 0.5 for each classifier.

Classifier	Cross Validation Score	Test Score
KNNC	0.349	0.387
RFC	0.369	0.404
DTC	0.47	0.509

TABLE I: First multiclass classifier accuracy measurement

As shown in **table I**, DTC is the best performing one, followed by the RFC and then KNNC. The low accuracy shows that multiple classifiers do struggle to predict genres consistently. To get a better idea on how much the accuracy has improved, the accuracy will be calculated for each genre.

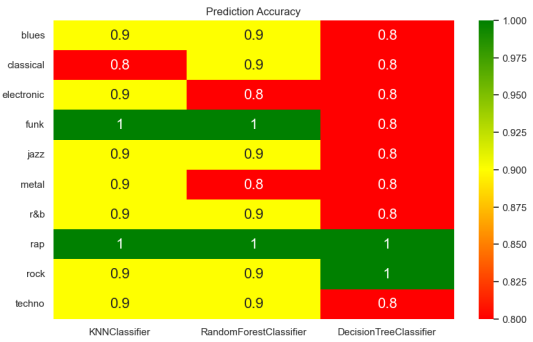

Fig. 3: Accuracy of each model on the test dataset.

Based on **figure 3**, it can be seen that the lowest prediction accuracy is equal to 0.8. The worst performing classifier is DTC. Two other classifiers seem to predict everything at least with an accuracy of 0.9 on average. There are a few exceptions that stand out. Genre *rap* is getting perfectly identified every time by all classifiers. KNNC and RFC are both also excelling in predicting genre *funk*, though in comparison they perform poorly on classifying genres *classical* and *electronic*. The predictions with classifiers seem to perform well genre-wise. Nevertheless, DTC holds potential for hyperparameter optimization.

To find the optimal set of hyperparameters for three classifiers, again *randomized search* was used. For each classifier, a reasonable hyperparameter range was fed into the search space for optimization. These individual combination of parameters were then used to configure the corresponding models' parameters. This whole process was repeated 100 times for each model with a variety of combinations.

Figure 4 illustrates the genre accuracy after the hyperparameter optimization. It is fairly easy to see which genres improved in comparison to the initial training run. **Figure 5** depicts the post-optimization difference in accuracy of models. Majority of the classifiers improved their recognition of correct genres with only one exception - RFC decreased in performance of detecting the genre *classical* by 10%. Another fact that stands out, is that DTC showed significant improvement by 20% increase in four out of ten genres.

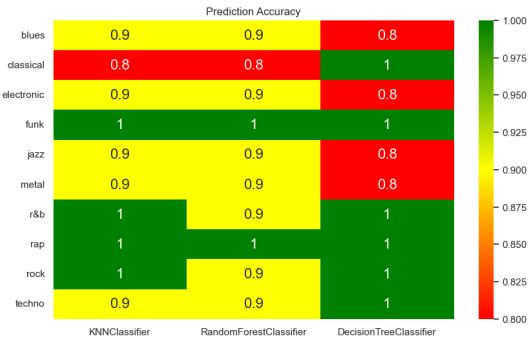

Fig. 4: Accuracy of each model on the test dataset, after hyperparameter optimization.

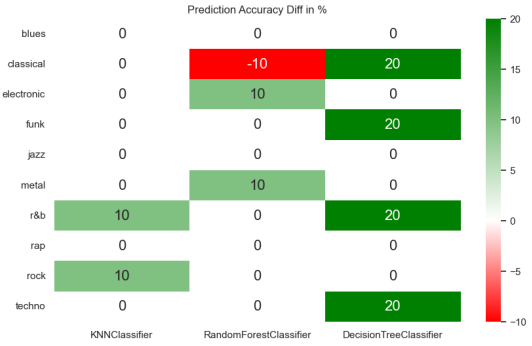

Fig. 5: Accuracy difference in percentages, after hyperparameter optimization.

To make a final comparison with the binary classification approach, a ***soft voting*** classifier was used to sum up the prediction output for each classifier. With soft voting the label is predicted based on the ***argmax*** of the sums of the predicted probabilities, where as *hard voting* uses for majority rule voting based on the class labels that were predicted. [7] The **table II** below lists the respective accuracies of the models.

Furthermore, it was in our interest to investigate how ***ensemble*** learning would perform for this problem. As an alternative multiclass classifier, we created an ensemble of binary classifiers, which were introduced in the section III-A for the same ML models of multiclass classification. The goal of ensemble model is to make binary classifiers collaborate. Likewise, we decided on soft voting strategy for model's prediction. The accuracy scores are almost the same as the optimized multiclass classifiers except `KNNC`, which scored an accuracy lower than its multiclass classifier version.

Classifier	Soft Voting	Optimized Accuracy	Accuracy
KNNC	0.5435	0.4255	0.387
RFC	0.63	0.394	0.404
DTC	0.4995	0.5085	0.509

TABLE II: Final Multiclass classifier accuracy

Classifier	Accuracy
KNNC	0.4715
RFC	0.6255
DTC	0.483

TABLE III: Ensemble learning accuracy

IV. Performance Evalution

Section III summarized the practical work conducted for the genre classification problem of music tracks. This section investigates the outcomes.

When comparing two approaches, one vs. all classifiers excel against multiclass classifiers. In addition to that, they showed no sign of overfitting and recorded a high accuracy score on the testing set with no special configuration of models' parameters. However though, hyperparameter optimization made no significant improvement on the models' performances. Observing from genre perspective, one vs. all classifiers performed very closely on genres *blues, funk, jazz* and *rock*.

Just as binary classifiers, multiclass classifiers did not suffer from overfitting, although they scored poorly in general. There are thousands of tracks in the complete dataset which are marked with multiple genres. Even though, we set a threshold of 1000 unique tracks for each genre, having tracks belonging to multiple genres possibly influenced the performance of models. Unlike one vs. all, many models improved their performances after hyperparameter optimization.

Ensemble of binary classifiers behaved very similar to multiclass classifier configured with soft voting. This strongly stems from the fact that the `VotingClassifier` class of scikit-learn, it internally trains binary classifiers for each class when class attribute `voting` is set to **soft**. Besides, the tracks with multiple genres cause same phenomenon in this case too.

To recap, one vs. all approach is very effective for this type of task. Usually, this approach is computationally more demanding, since a model for each class needs to be trained. In fact, we didn't observe any significant difference in the time required to train the models. Though, this may change if more genres and new tracks were to be introduced. Last but not least, letting the `VotingClassifier` create, train binary classifiers internally or programming it manually result in no noteworthy difference.

V. Conclusion

Art can be interpreted in many ways. When subjectivity is induced, objectivity decreases. When objectivity decreases, facts fade away. Just like any type of art, music embodies this fact and analysis of music is susceptible to this aspect. We based our analysis solely on the sonic characteristics of the tracks. Consequently, this approach yields promising results.

It proved to be successful to identify a track's genre based on the provided features.

The other side of the coin sheds light onto interesting topics. One remark to this approach is the dependency nature that it brings along. With the current set up, every single track available on Spotify can be analyzed. But if we wanted to analyze and predict the genre of a track which doesn't exist in the Spotify's database, it would require substantial adjustments to the data preparation pipeline in order to extract the needed track features from an audio file.

Second of all, we kept the number of classes extremely low. This enabled us to verify the chosen method quickly. Although, extending the palette of genres would eventually increase the complexity. In that case, new strategies shall be considered.

The word *genre* is defined as "a particular type or style of art, film, literature, or music which recognizable by certain features" by Cambridge Dictionary [8]. So far we only considered audible features. Controversially, tracks can be clustered based on the emotion they convey and this approach might bring a completely new breath the way we handle categorization of music.

REFERENCES

[1] Spotify API: https://developer.spotify.com/documentation/web-api. Accessed: 2023-06-07

[2] The Sound of Everything Playlist: https://open.spotify.com/playlist/69fEt9DN5r4JQATi52sRtq?si=380e42a1e6b149f8. Accessed: 2023-06-07

[3] Search for item: https://developer.spotify.com/documentation/web-api/reference/search. Accessed: 2023-06-08

[4] Audio Features: https://developer.spotify.com/documentation/web-api/reference/get-several-audio-features. Accessed: 2023-06-08

[5] GitHub Repository https://github.com/alpargur/machine-learning-101/tree/project/genre-classification. Accessed: 2023-6-08

[6] scikit-learn https://scikit-learn.org/stable/. Accessed: 2023-6-10

[7] scikit-learn Voting Classifier https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.VotingClassifier.html#sklearn.ensemble.VotingClassifier. Accessed: 2023-6-11

[8] Genre, word definition https://dictionary.cambridge.org/dictionary/english-german/genre. Accessed: 2023-6-12