Final Project Report

Qingcheng Wei, Cole Whaley, Sammy Gagou, Emily Yan

December 15, 2023

1 Problem Overview

When searching for new music on Spotify, it can sometimes be hard to pinpoint genres based on what one has already seen. This issue motivates the use of machine learning models to more accurately identify and find music of interest. What exactly a genre is can be ambiguous, so we decided to narrow our selection down to R&B, Pop, Metal, Jazz, House, Hip-Hop, Electronic, Disco, Dance, Country, Classical, and Rock music. We also chose to evaluate what genre a piece of music might fall under using these features: popularity, duration (in milliseconds), danceability, energy, key, loudness, mode, speechiness, acousticness, instrumentalness, liveness, valence, tempo, and time signature (a measure of how the beat in a song is formatted). In particular, we chose to use both random forest model and CNN model to determine the genre a particular song might fall under.

2 Data Visualization

In order to more accurately determine what genre a particular song falls under, we used a dataset from kaggle.com of roughly 114,000 songs, with the values of their aforementioned features. Although the size of this dataset might not seem appropriate for a rather simple problem, we see a dataset of this size to be necessary in order to more accurately distinguish subtleties between songs that might otherwise fall under the same genre.

3 Data Pre-processing

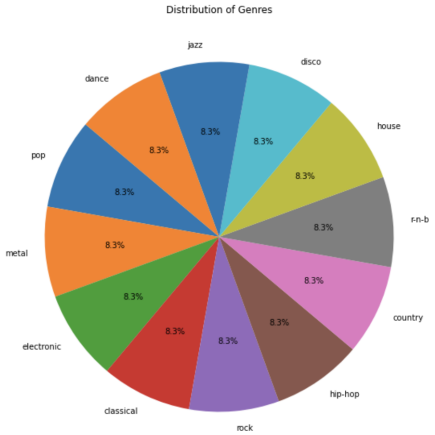

Figure 1: Distribution of Music Genres

In our preprocessing pipeline, we emphasize the normalization of input data as an essential step before feeding it into the model. This normalization helps ensure that features with different scales contribute equally to the model's training process.

Additionally, we optimize the dataset by removing music genres that are infrequently represented, allowing us to focus on those that are more mainstream. This selective approach enables the model to specialize in predicting the more prevalent genres in our dataset.

To enhance the efficiency of our model and eliminate irrelevant features, we exclude certain categorical data fields, such as album name and track name, that exhibit no discernible correlation with the classification labels.

Recognizing the potential impact of different artists on musical preferences and genre classifications, we employ label encoding specifically on the artist name variable. This allows us to capture and represent the diverse tastes associated with individual artists, which we believe can influence the ultimate music genre prediction.

4 Model Overview

For this particular problem, we chose to evaluate genres of a particular song by using a random forest classifier. In particular, when determining a song, we want to use our outlined features to create more branches in our decision tree and be able to take multiple predictions from different decision trees in order to combine them for genre determination. Finally, we plan to determine genres in each decision tree by selecting particular leaf nodes and recursively iterate through a each decision tree until that leaf provides us with an answer.

4.1 Random Forest (Decision Tree) Model

Because our problem required a multi-class determination, we sought first to use a model which did not require categorizing numerical features nor normalizing these features. Our initial attempt for determining genre utilized a random forest classifier. This model was chosen first based upon several factors to be expanded on below such as: prediction combination from multiple trees and reducing risk of over-fitting.

4.1.1 Walking through the model

As aforementioned, our first model was chosen based on key characteristics that seemed to fit our problem well.

These key components include:

1. **Combination of predictions from multiple trees:** Our primary thought was simply to use a decision tree classifier; however, we ultimately decided that combining the efforts of multiple trees may increase our model's overall accuracy. Additionally, it provides protection from over-fitting (as will be discussed in item 3)

2. **Classify genres based on leaf node selection:** Because of the tree structure, the model was able to make clear decisions for a multi-class problem. Upon reaching a leaf node, the tree's prediction was trivial to present and compare with other trees' predictions.

3. **Mitigate Over-fitting:** Over-fitting can greatly affect a model's accuracy and skew results. In order to mitigate this, we chose a model that would average decisions and seek to avoid a plunge in accuracy as a result of a large data quantity.

4.1.2 Model Framework

We experimented with several parameters to the model function. Ultimately, we landed at using 200 n estimators. We did not see a significant increase in accuracy when adjusting the minimum samples split, max depth, minimum samples leaf parameters. Additionally, we kept the random state the same constant, yet arbitrary, value (37) throughout the model. With this blend of values, our random forest classifier displayed relatively high accuracy and a very high ROC-AUC score. This ROC-AUC score shows the model's ability to distinguish between the classes. With a score of 94, our model did this very well.

4.2 CNN Model

To expand our model repertoire beyond decision tree models and try to enhance its accuracy beyond the current threshold, we additionally incorporate a Convolutional Neural Network (CNN) model into the training data. This inclusion aims to assess the CNN model's performance in terms of accuracy and contribute to the overall improvement of our predictive capabilities.

4.2.1 Walking through the model

CNN Model is based on a Convolutional Neural Network, which is a type of deep learning algorithm designed for processing structured grid data. It is particularly powerful for tasks such as image and video recognition, image classification, and computer vision. For this task, we employ this model in numerical data.

Some key components of a CNN structure include:

1. **Convolutional Layer:** Convolutional Layer is the core block of the CNN as it applied convolutional operations to the input data using filters or kernels, allowing the network to learn hierarchical structures.

2. **Pooling Layer:** Pooling layers such as max pooling are used to reduce the spatial dimensions of the input data and decrease the computational load.

3. **Flattening Layer:** This layer is used before the fully connected layers to flatten the previous layers into a one-dimensional vector.

4. **Fully Connected Layer:** In the final layers of the network, this layer is used for classification tasks. These layers connect every neuron in one layer to every neuron in the next layer, enabling the network to make predictions.

4.2.2 Model Framework

In crafting our model, extensive experimentation led us to a structure carefully calibrated to achieve a harmonious equilibrium between optimizing accuracy, managing computational complexity, and mitigating the risk of overfitting. The model architecture involves four convolutional layers, succeeded by one max-pooling layer, one flattening layer, and four fully connected layers, ultimately culminating in predictions across 12 categories. For training, we configured the optimizer as Adam and employed sparse categorical cross-entropy to evaluate accuracy. We chose the epoch of 50 and batch size of 64 to train the model. The outlined architecture encapsulates the essence of our framework, embodying a thoughtful balance between performance and complexity.

Layer (type)	Output Shape	Param #
conv1d_12 (Conv1D)	(None, 14, 32)	128
conv1d_13 (Conv1D)	(None, 12, 64)	6208
conv1d_14 (Conv1D)	(None, 10, 128)	24704
conv1d_15 (Conv1D)	(None, 8, 256)	98560
max_pooling1d_3 (MaxPooling 1D)	(None, 4, 256)	0
flatten_3 (Flatten)	(None, 1024)	0
dense_12 (Dense)	(None, 128)	131200
dense_13 (Dense)	(None, 64)	8256
dense_14 (Dense)	(None, 32)	2080
dense_15 (Dense)	(None, 12)	396

Figure 2: CNN Architecture

4.3 Model Performance

In order to evalute the performance of our random forest classifier model, we used a confusion matrix on our validation set, which resulted in an accuracy of roughly 66.7%. As seen in the following figure,

correct identification of what genre a song falls under was generally achieved, with errors in determining subtleties between the following pairs: (jazz, hip-hop), (hip-hop, classical), (metal, classical), (pop, house), (rnb, house), (electronic, rock), and others. Generally, we can notice that it was rather difficult for our model to recognize the differences between hip-hop, metal, jazz, classical and other genres.

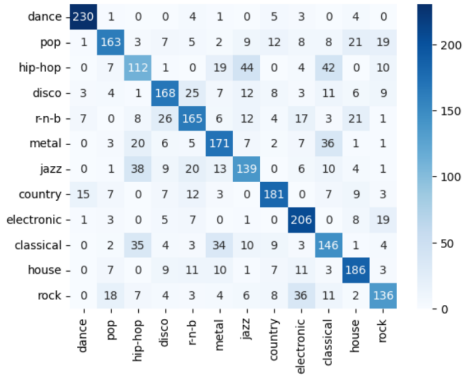

Figure 3: Confusion Matrix for Decision Tree Model

As for the CNN model, the culmination of our efforts resulted in a final accuracy of approximately 57%, which, notably, falls below the accuracy achieved by the decision tree model.

5 Model Comparison

As previously mentioned, the accuracy of the Convolutional Neural Network was around 57% and the accuracy of the Random Forest Classifier was 66.7%. We expected the accuracy to be lower for the Convolutional Neural Network since we probably did not have enough data to feed into this model in particular such that it would produce similar results as the Random Forest Classifier. Decision trees can be effective for relatively simple and small datasets just like our dataset with relatively clear decision boundaries.

References

[1] Spotify Tracks Dataset (2022). https://www.kaggle.com/datasets/maharshipandya/-spotify-tracks-dataset
data.

[2] Random forests Leo Breiman and Adele Cutler. Random forests - classification description.
(n.d.). https://www.stat.berkeley.edu/~breiman/RandomForests/cc_home.htm