Sean Simon

Predicting Music Genre

Springboard Capstone 2

Intro

Building a music playlist is no easy task. Having a fresh playlist for each occasion is even harder: dinner parties will lose their luster if the same bebop jazz album is on shuffle for each gathering. Two major challenges in constructing a vibrant playlist are staying on top of new music releases, and keeping the soundscape robust with a smattering of tracks from adjacent genres but with the same desired feel. Spotify offers data for music hosted on their platform with features such as tempo, key, accousticness, and importantly, genre. By fitting a machine learning model to this data for the purpose of classifying genre, either by identifying genre and selecting new music from that genre or by expanding the borders of the genre by selecting music from "adjacent" genres that the classifier learns are similar, robust playlists can be created. The availability of such a model puts power into the hands of Spotify users to create the ultimate playlist.

Data

The raw data set for this work was obtained from Kaggle.com, where itself was obtained from the Spotify API. The data is an 8 mb CSV file of 50,005 rows, where each row is a single song with 18 columns of features, including genre. The data set contains 10 genres: Alternative, Anime, Blues, Classical, Country, Electronic, Hip-Hop, Jazz, Rap, and Rock. The 18 features are Instance ID, Artist Name, Track Name, Popularity, Accousticness, Danceability, Duration, Energy, Instrumentalness, Key, Liveness, Loudness, Mode, Speechiness, Tempo, Obtained Date, Valence, and Genre.

Rows of missing data, totalling 5 rows, were removed from the data set completely. Other missing values persisted in the Duration and Tempo columns, totalling about 5,000 missing values each. These missing values were imputed using the median and mean of each column, respectively. In regards to imputing with the median for the Duration feature, there was a heavy right hand skew due to certain songs that were full DJ sets totalling over an hour of playing time. Tempo was much more normally distributed and therefore imputed with the mean.

High entropy features Instance ID, Artist Name, and Track Name were also removed from the data: for example Instance ID was wholly unique for each row and therefore contained no predictive power aside from the possibility of confounding analysis depending on how the data was entered originally. Obtained Date was also removed because it only recorded how the data was accessed rather than any feature about the music. In the end, the cleaned data set represented 50,000 unique songs with 13 unique features and 1 classification label.

Exploration

An important aspect of the data is how it's distributed, especially for classification problems because an uneven representation of labels can present some analysis problems. In this case, each of the 10 genres are evenly represented, with about 5,000 entries each.

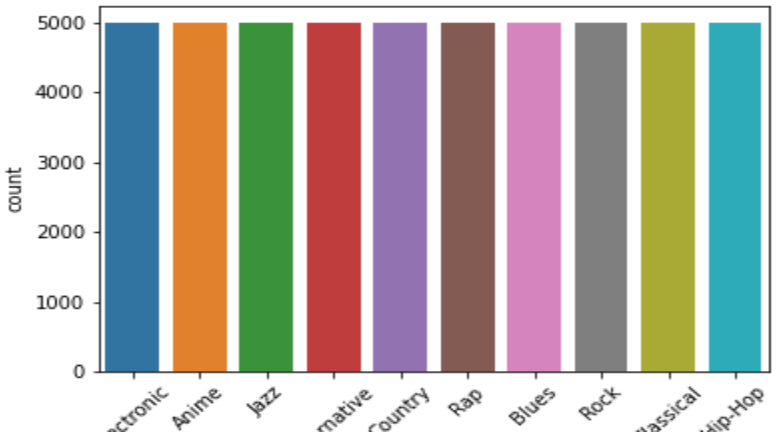

Fig 1.: Distribution of Genre labels in data set

Almost as important is the distribution of the other features, because most machine learning tools assume a normal distribution. In this data set Danceability, Tempo, and Valence are normally distributed, but Duration, Liveness, and Speechiness are not and will have to be transformed prior to modelling.

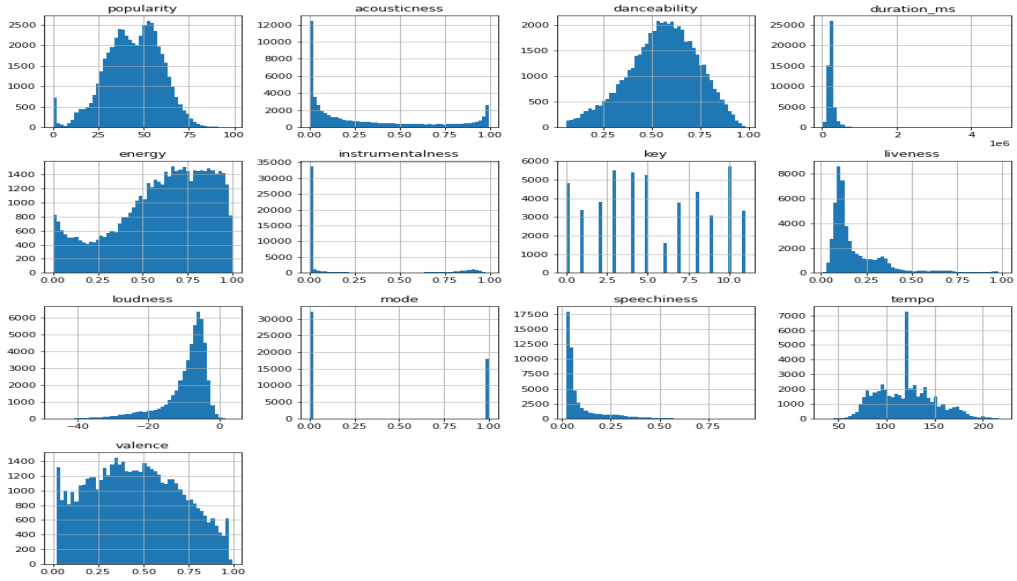

Fig 2.: Distribution of observation features

Insight can also be gained on how the features interact with each other and how the genres separate by using a pairplot. Songs with high energy also tend to have high loudness, whereas high acousticness tends to have low loudness. Hip-Hop has a much higher danceability than Classical, and Anime is visible in low-popularity regions. Notably, Hip-Hop appears to completely cover Rap, so these two genres appear hard to separate.

Fig 3.: Pairplot of each feature

Seen more easily with a heatmap is the fact that beyond energy and loudness there aren't any two features with enough collinearity to support removing any further columns from the data set and it can remain as is.

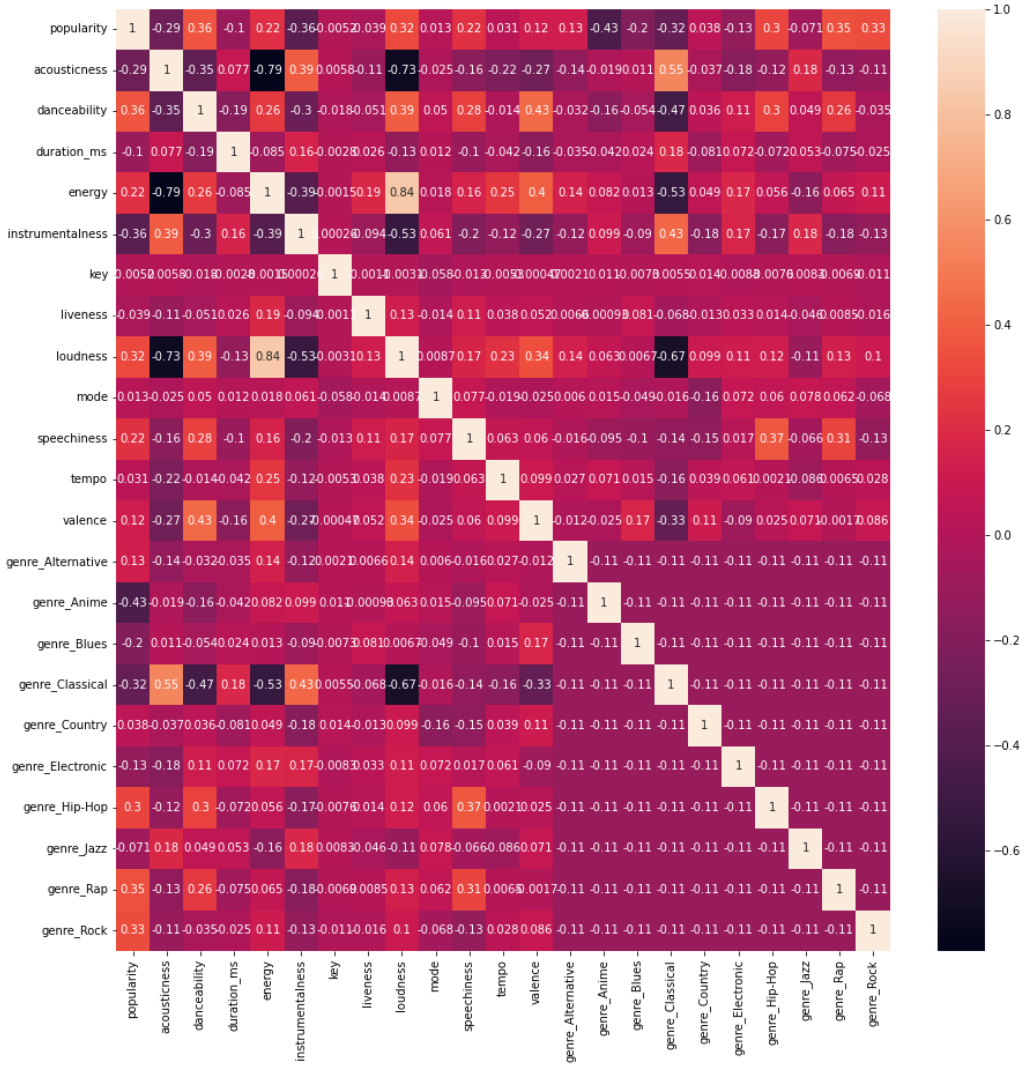

Fig 4.: Heatmap of the correlation between features of the data set

Feature Engineering

With the data cleaned, visualized for distributions, and demonstrating clear delineations between several labels, it must be prepared for model fitting. Importantly, these preparation steps must be defined based on training data and then later applied to testing data for verification of the model. Therefore, the data was randomly split into a training set with 75% of the data and a testing set with 25% of the data. Further, each set was split into a matrix of features with 13 columns and a vector of labels.

Secondly, within the 13 unique features are two categorical features: Key and Mode with 12 and 2 categories respectively. Notably, categorical features need to be numeric for most machine learning models to work. This can be done using One Hot Encoding which creates a new feature column for each categorical label. Taking the Key as an example, in each of the new columns a value of 1 is recorded if the observation features that key or a 0 otherwise. In each case, Key and Mode, an encoded column is dropped because it features redundant information: if all newly encoded columns have a 0, a 1 can be assumed in the column that is not present. After the One Hot Encoding, the training data matrix features 23 columns of features.

Next, the skewed data in Acousticness, Duration, Instrumentalness, Liveness, Loudness, and Speechiness was transformed with a Power Transformer to normalize their distributions better. And lastly, some of the data existed on different scales, such as Popularity (0 - 100) and Acousticness (0 - 1). Popularity, Acousticness, Duration, Instrumentalness, Liveness, Loudness, Speechiness, and Tempo were therefore all scaled with a Min-Max scaler to all be between 0 and 1 with the purpose of allowing machine learning models to understand the differences between features more easily.

Modeling

With the data cleaned and prepared for modeling, the right type of model must be chosen. Typical rule of thumb is that tens of percentage points of accuracy can be gained by selecting the right model in the first place, and then a further few single digit percentage points can be gained by hyperparameter tuning. To this end, 8 models from Scikit-Learn that specialize in multilabel classification problems were instantialized in their default form and fit to the data with 10-fold cross-validation. For each model, a box plot was drawn for the cross-validation scores, with a Gradient Boost Classifier performing above the competition by a wide margin.

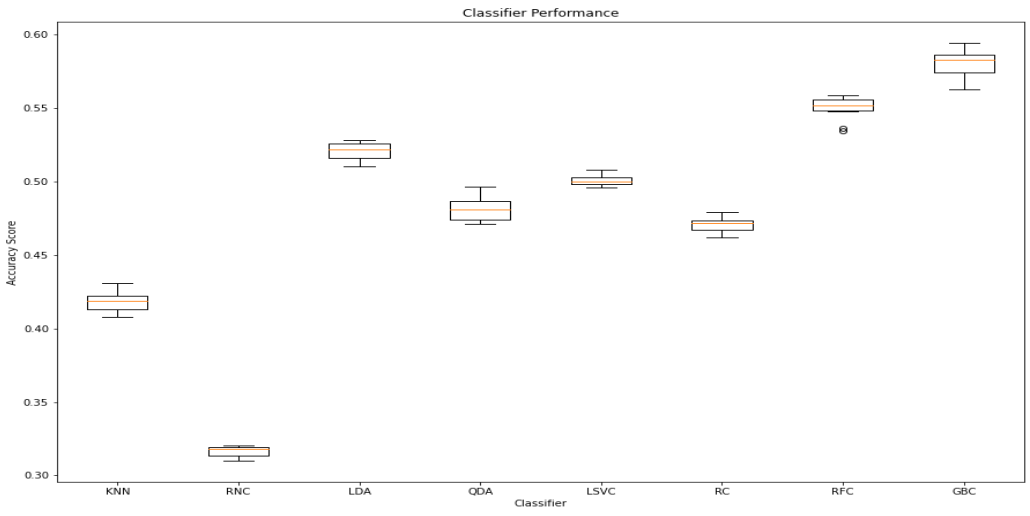

Fig 5.: Accuracy of each classification model type

The Gradient Boost Classifier was taken forward to hyperparameter tuning with the Grid Search CV method. Of interest were the parameters Learning Rate, N Estimators, Criterion, Max Depth, and Max Features. Ultimately, grid searching identified a Friedman MSE criterion, 0.1 learning rate, max depth of 3, feature selection limited to the square root of all features, and 400 estimators. With these settings, a maximum accuracy of 58.4% was obtained on the training data during cross validation.

Refitting the model to the entire training set and then predicting the test set led to an accuracy of 49.2%. Randomly assigning a label would have produced an accuracy of 10% due to the even distribution of 10 classes. The confusion matrix displays an even more telling story, with very high accuracy on the Classical genre, but overlap between Hip-Hop and Rap - something that was predicted when looking at the pairplot data.

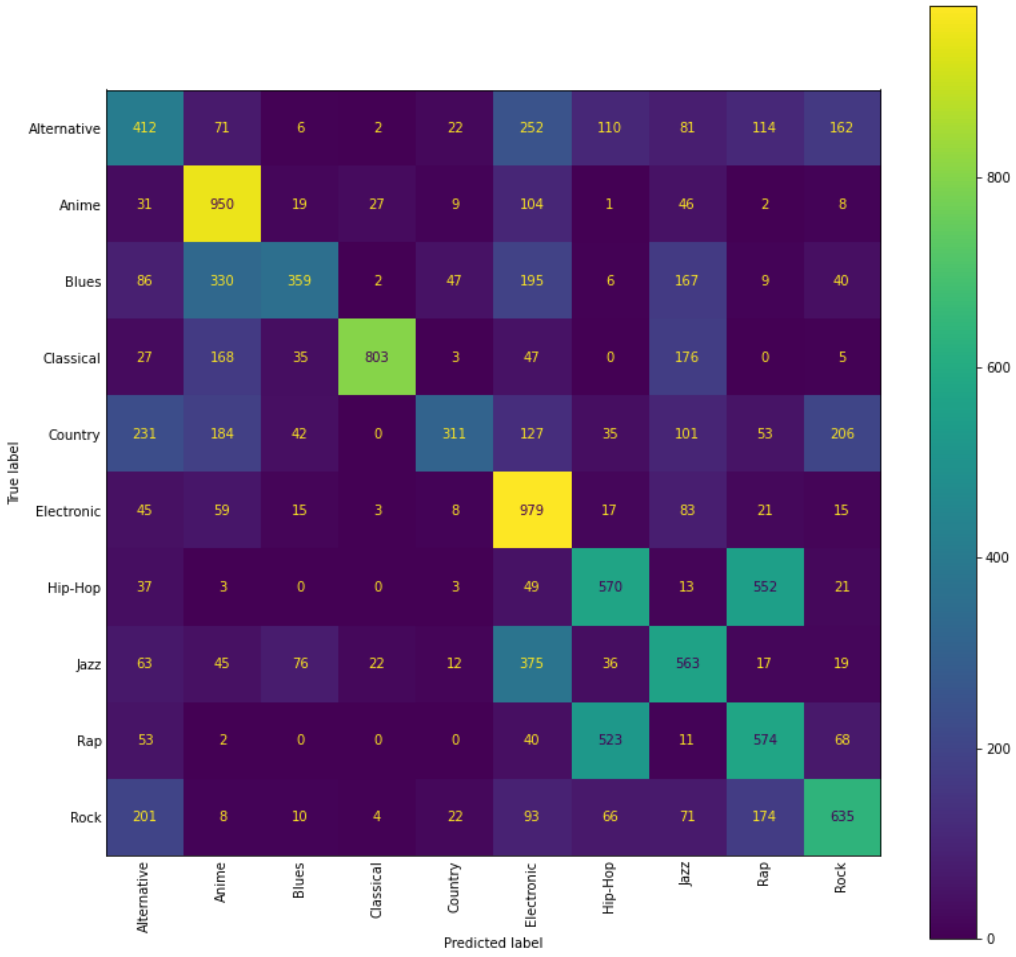

Fig 6.: Confusion matrix of model

Conclusion

Ultimately, a model was fit that can predict music genre based on song data with 50% accuracy, a notable improvement over the 10% accuracy expected when randomly assigning the set of genres to the data. Looking at the confusion matrix, it can be seen that certain musical genre labels can be predicted pretty well, such as Anime, Classical, and Electronic. The Anime label has a low precision, however: Blues, Classical, and Country are often wrongly predicted as Anime. This makes sense, because as a genre Anime is a soundtrack, meaning it can borrow from many other genres. At the other end of the spectrum, Country is predicted poorly. However, True Country gets misinterpreted as Alternative, Anime, and Rock quite often, and one would probably agree that in reality Country songs can be Alternative and Rock sounding. Lastly, there is poor differentiation between Rap and Hip-Hop, but these are two very similar genres with a lot of speechiness.

In the context of the objective as outlined in the intro, the model does a very good job, even with it's confusions. When constructing a playlist, the model can ascertain the primary genre, and suggest songs with similarity to add next. It can be argued that a playlist with mostly Rock could be made more robust with additions from the similar genre Alternative, or both Electronic and Jazz combined might make a robust playlist in the background for studying.

All of that said, there's a wide margin between 50% accuracy and 100% accuracy. Future iterations of this work should strive to close this gap. One possibility could come from using PCA to separate out the features better, since it was noticed there is distinct overlap between certain genres. Another option would be to bring in additional data, like perhaps advanced signal processing on the time series frequency data to harvest even more data from the songs. This also opens up the possibility of the model to generalize further beyond Spotify data only. Lastly, Gradient Boost Classifiers are slow, and further feature engineering should be explored to see if a faster model can be fitted with similar performance.