Parker Johnson
COEN140
December 13th, 2019

Music Genre Classification: Machine Learning Final Report

Introduction/Motivation

Analysis of the structure of music is helpful for audio-based companies (like Spotify, SoundCloud, or Shazaam) that are looking to make music recommendations for their customers. The first step towards this sort of application is genre classification.

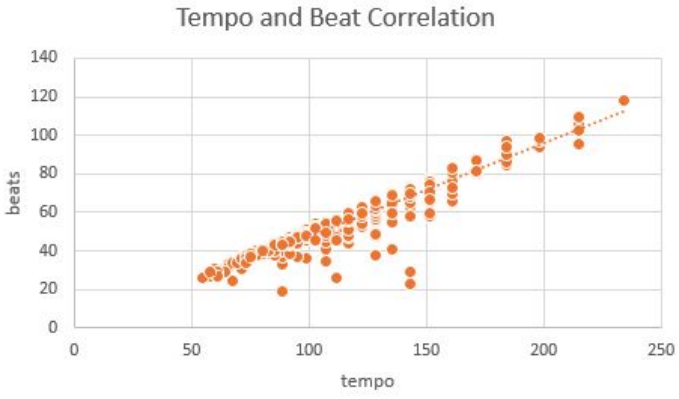
My motivation for pursuing this topic was my love or music, which stems from years of piano-lessons, an array of musical theater performances, and from my own fascination with music as a whole.

Methodology

The data I worked with was the 'GITZAN Genre Collection', which is a Dataset consisting of 1,000 songs across 10 genres (with each genre representing 100 tracks). The tracks are 16-bit audio files that can be expressed as a function of time (30-seconds) and frequency (22kHz). With these two parameters, a variety of features can be extracted from the audio files using an audio-processing library from Python called Librosa.

There are two sub-groups of features that I used for comparison in my models. The first subgroup includes beat-synchronous features like tempo, beat, and zero-cross rating. I grouped these features together because of how much the features correlated throughout the dataset. All of the features derived from the spectrum of the signal are correlated (Spectral Centroid, Roll-off, and Bandwidth). Another feature that was helpful

Figure 1: Graph of Tempo (Speed) and Beat (beats per min)



for genre distinction was zero-cross rating, which measures the number of times that the signal energy changes negative and positive. Music genres with higher percussion sounds like metal, disco, and pop will have a higher zero-cross rating.

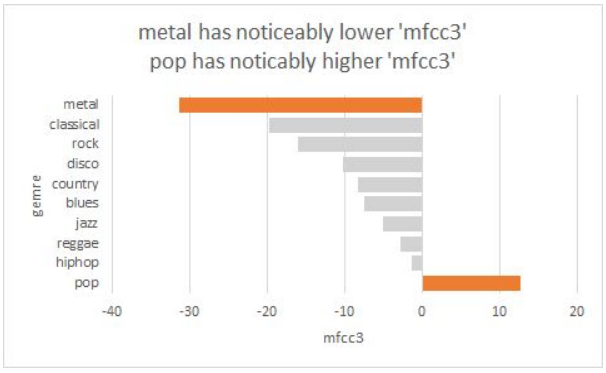
Figure 2: Average zero-cross rating by genre

Row Labels	Average of zero_crossing_rate
metal	0.147506281
pop	0.133610349
disco	0.129404142
rock	0.109246864
hiphop	0.108005414
reggae	0.091593845
country	0.08340713
blues	0.078982852
classical	0.077635974
jazz	0.076979727

The second group consisted only of MFCCs or Mel Frequency Cepstral Coefficients. These Coefficients are widely used in speech/speaker recognition because of their relationship to characteristics of an audio file vocal track. Each MFCC feature vector describes only the spectral envelope of one frame.

I chose to apply three models to the features I extracted, Logistic Regression, OneVsRest classifier, and Random Forest classifier. In order to implement a Logistic Regression model with more than one class, we can no longer map the probability of an instance being in one class vs. another with 0 or 1 with the Sigmoid function. Instead, the model maps to a Softmax function that maps each instance to the probability that lies within the range of 0 to 1.

Figure 3: Mfcc3; Metal vs. Pop



The OneVsRest classifier requires training a single classifier per class, which treats samples within its own class as positives and all other classes as negative. Then, each of these base classifiers produces a set of confidence scores for its decisions. New predictions are made by applying all the classifiers and returning the label with the highest confidence score.

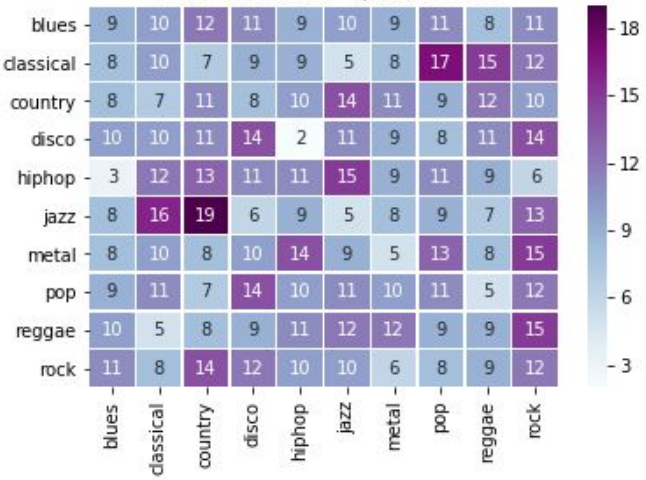
The Random Forest classifier is a Decision Tree classifier that selects a random set of features when considering splits for each node in a decision tree. It is also called random because

when the model is building each decision tree for a class, it takes a random sampling of the raining points.

Experimental Results and Analysis

For my baseline model, I tested how well a random number generator could predict all 1,000 instances. On average, this generated string of predicted labels between 0-9 (There are 10 classes, one for each genre). The accuracy of this model hovered anywhere from 0.09 to 0.11. This makes sense, as it should have roughly a one in ten chance of randomly guessing correctly for each class.

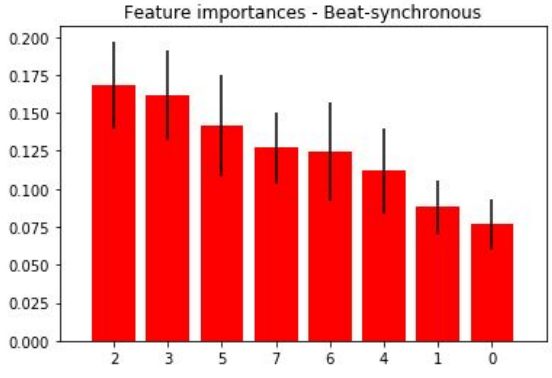
Figure 4: Confusion matrix with predicted labels as the x-axis and true labels on the y-axis



Beat-Synchronous - 8 features

After this, I tested my models with the first sub-group of features, the beat-synchronous features. With only eight features, each model was able to correctly categorize 40% of the songs into its correct genre. With so few data points per genre, I might have over-fit some of the decision trees within the Random Forest. This could explain its 100& accuracy.

Figure 5: Most important features in descending order: chroma_stft(2), rmse(3), specral_bandwith(5)



8 features	LR w/ Softmax	OVR	Random Forest
Train Accuracy	0.5208955223880597	0.4940298507462686	1.0
Test Accuracy	0.4272727272727272	0.4727272727272727	0.49

For the classifiers that were trained with beat-synchronous features, the genres that were the easiest to classify were classical, metal, and pop. All of the identifiable genres had the highest zero-cross rating, I am not trying to infer anything about the importance of zero-cross rating, but instead, I am referencing Figure 1. In both OVR and LR, rock is the most ambiguous genre. In both cases, the prediction for a rock song is split relatively evenly among four other genres.

Figure 6 - Confusion matrix - LR

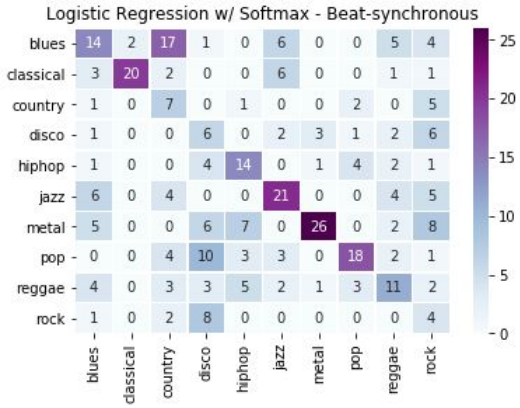


Figure 7 - Confusion matrix - OVR

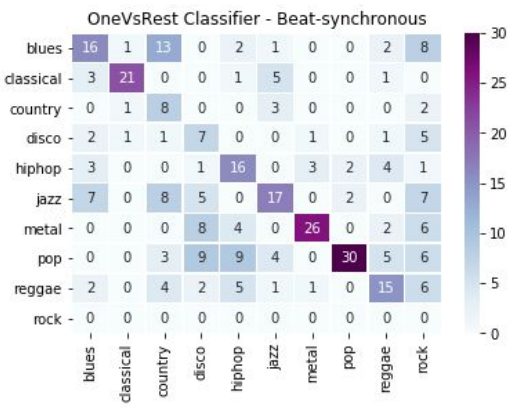
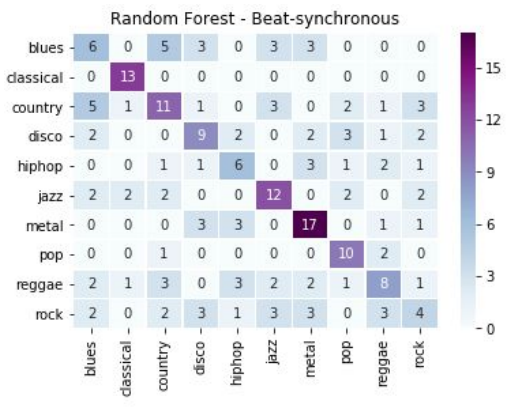


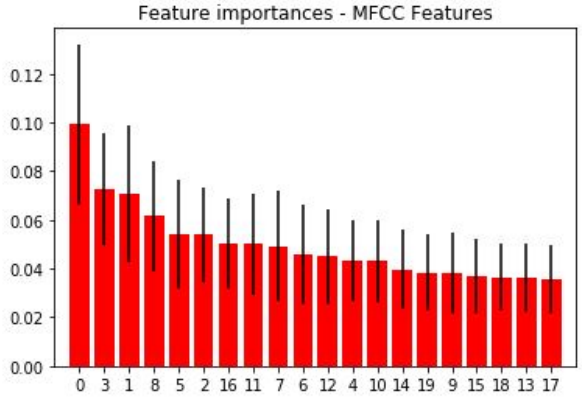
Figure 8 - Confusion matrix - RF



Mfccs - 20 features

The models were able to predict the genre labels slightly better than the first sub-group of features. On average, each model improved its training accuracy by ~6%. There was over twice the number of features for this set of predictions. OVR had the lowest improvement with 3%. Train Accuracy (not counting RF) improved by an average of ~12%.

Figure 9: Most important features in descending order: Mfcc1(0 on x-axis), Mfcc4(3 on x-axis), Mfcc2(1 on x-axis)



20 features	LR w/ Softmax	OVR	Random Forest
Train Accuracy	0.6388059701492538	0.6343283582089553	1.0
Test Accuracy	0.5363636363636364	0.503030303030303	0.565

Figure 11 - Confusion matrix - LR

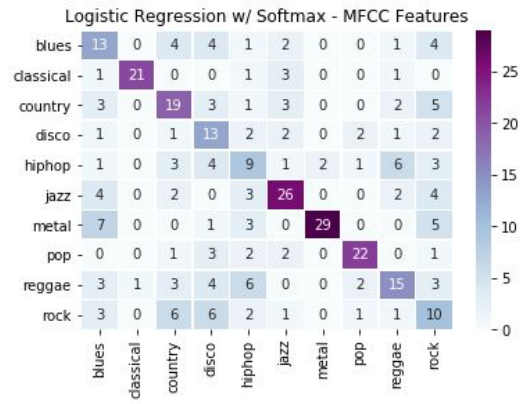


Figure 12 - Confusion matrix - OVR

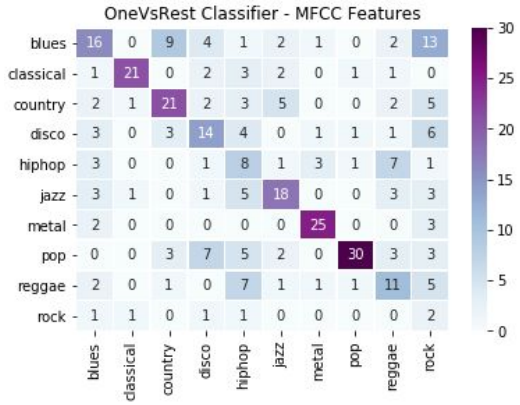
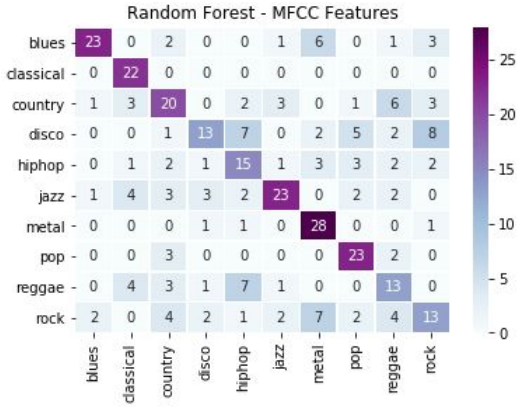


Figure 13 - Confusion matrix - RF



Above is a representation of the test accuracy returned from the models when trained on the second sub-group of features. I would have hypothesized that genres that traditionally have a distinct vocal structure aspect would have been easier to identify by these models. However, this hypothesis would require me to define what a 'distinct vocal structure' means. Jazz, metal, and pop remained the most distinguishable. Hip-hop, disco, and rock went through the same amount of trouble as the Beat-synchronous feature models

All - 28 features

Here are the results for the models when trained on the combination of both sub-groups of features. In the future, I would like to focus on one model trained on data from fewer genres.

20 features	LR w/ Softmax	OVR	Random Forest
Train Accuracy	0.7283582089552239	0.7164179104477612	1.0
Test Accuracy	0.6060606060606061	0.5787878787878787	0.612121212121212

Figure 14 - Confusion matrix - LR

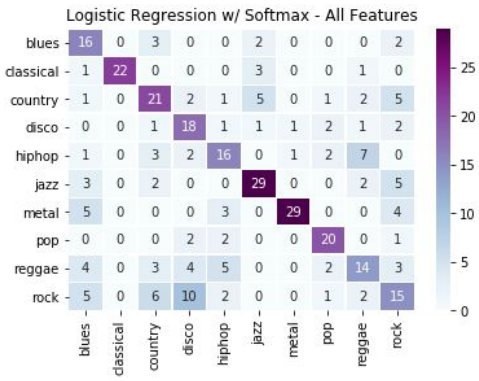


Figure 15 - Confusion matrix - OVR

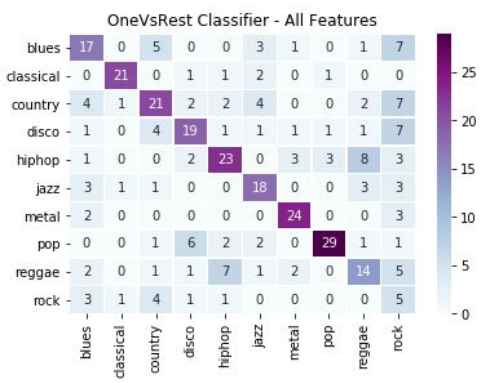
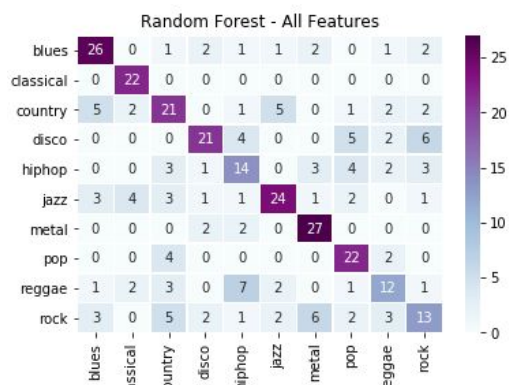


Figure 16 - Confusion matrix - RF



References

Music Structure Analysis by Ridge Regression of Beat-Synchronous Audio features. Yannis Panagakis and Constantine Kotropoulos (ISMIR 2012)

Libraries

from sklearn.preprocessing import StandardScaler	- Standard Deviation
from sklearn.svm import LinearSVC	- Component of OVR
from sklearn.multiclass import OneVsRestClassifier	- Model
from sklearn.metrics import accuracy_score	- Metrics
from sklearn.metrics import precision_score	- Metrics
from sklearn.model_selection import train_test_split	-Test/Train
from sklearn.metrics import confusion_matrix	-Visualization
import seaborn as sn	-Visualization
import numpy as np	-Data Warehousing
import pandas as pd	-Data Warehousing
import os	-File OS

Beat-synchronous features

Index	Feature Name
0	tempo
1	beat
2	chroma_stft
3	rmse
4	spectral_centroid
5	spectral_bandwith
6	rolloff
7	zero_cross_rating