Composer Predictor

Sean Yeh (seanyeh2014 [at] u.northwestern.edu)

Paul Juhn (pauljuhn2014 [at] u.northwestern.edu)

Our task is to predict the composer of melodic fragments of musical pieces based on previous data from those specific composers.This task is important to see if similarities exist between composers and if they do, what makes them similar. On the other hand, we can also see what features makes a composer unique, which allows us to categorize that melody. There have also been times in history where a piece of classical music surfaces and the composer is unknown. With enough data, our composer predictor could classify such unknown works and give credit where it's due.

We first took many midi files of pieces and wrote a script that separates them into short melodic fragments. This way we have multiple data examples from a smaller number of larger works. There are a few downsides with this approach, such as repetitions in the music or generic fragments that don't carry enough information about the composer. However, for this class, we decided to keep the scope small and manageable. We then analyzed the melodic fragments for various features, such as presence of low notes, chromaticism, ratio of melodic leaps, number of notes per fragment, melodic range, repeated notes, rhythmic variance, and rhythmic_variety. We separated the data into a training and testing set (70/30 split) and measured our success on the percentage of melodic fragments predicted correctly.

Our results were fairly successful, but that was not too surprising given the small scope of our project. We only looked at two composers, Bach and Chopin, who are quite different. However, our program is written without hardcoding the number of composers, and it is quite trivial to add more pieces. All that is needed is to create a new folder of the composer's name in

the raw data directory and add MIDI files. Using the scikit-learn machine learning library for Python, we used the ExtraTreesClassifier algorithm as our default algorithm, which according to their website, is a class that “implements a meta estimator that fits a number of randomized decision trees (a.k.a. extra-trees) on various sub-samples of the dataset and use averaging to improve the predictive accuracy and control over-fitting.” An another algorithm we tried was SVC, a C-Support Vector Classification algorithm, which performed slightly worse. This can be enabled by running “./model.py --algorithm=SVC”.

We could have not completed our project without the help of many 3rd party libraries. The most important libraries we used were MIT’s music21 library for music analysis and manipulation and scikit-learn, a library for machine learning algorithms. We also used other libraries such as pandas, numpy, and matplotlib.

Below is an example result:

```
(composer-predictor)/m/s/d/p/composer-predictor git:master ››› ./model.py -t -f -c

Cross-validation scores: [ 0.9122807   0.89473684  0.94736842  0.90909091  0.85454545]
Cross-validation avg: 0.90360446571

Prediction accuracy on test data: 0.959016393443

Feature Importances:
0.222399: rhythmic_variance
0.173407: num_notes
0.157857: rhythmic_variety
0.115456: Low_notes
0.114009: leaps_ratio
0.112826: repeated_notes
0.081684: range_melody
0.022362: chromaticism
```