CMPS 142
Professor Farzaneh Mirzazadeh
March 23, 2017

# Music Genre Classification

**Meet the Team Members:**

Ricardo Munoz: Undergraduate fourth year student. His role in this project was to analyze the dataset and extract the features that we found to be most prominent in determining a song's genre. Specifically, he took part in experimenting and extracting features with a visual equalizer.

Hershnoor Gill: Undergraduate fourth year student. His role in the project was to collect all the data found by Ricardo to construct the dataset. Specifically, he analyzed different patterns of classification with respect to the size of the dataset and with added noise.

**Problem:**

We decided to tackle the problem of classifying the music genre of four different genres of music: Rap, Classical, Electronic House, and Rock. We chose these specific genres because we believe they were distinguishable enough to get accurate results during our machine learning process using WEKA. Another problem we tackled was in finding which different classifier model had greater accuracy for different data sets. We created our own datasets of different sizes, and added noise to try to find the best performing model overall.

**Approach:**

At first, we were only going to address the single problem of identifying the genre of a song. However, after discussing with Professor Farzaneh, we found our problem to be too simple for the requirements of this project. So instead, we chose to broaden the scope of our project and address the following questions:

*Does increasing the dataset of song entries decrease our training/testing errors?*

*How is our dataset affected by unwanted noise? Is it sensitive to noise and does it increase our training/testing errors?*

In addition, we decided to experiment with multiple classifier models and compare which ones perform better on our different data sets (please refer to the directory named `WEKA_Model`).

**Constructing our Own Data Set:**

During the first couple of days working on this project, we spent a large amount of time constructing our dataset, rather than importing a data set from the web. The reason why we chose to construct our own dataset is because we wanted to choose the features we thought are most significant in determining a musical piece's genre. This involved selecting music out of our own libraries lying within the specific genres we chose to classify, importing each song to *Garageband* and using the Visual Equalizer to get the specific ranges of **decibels from four different frequency ranges** (bass $[20Hz - 110Hz]$, low middle $[110Hz - 640Hz]$, high middle $[640Hz - 3,600Hz]$, and treble $[3,600Hz - 20,000Hz]$ for 20 songs of each genre. For each song, we noted the decibel values that each frequency range reached and took the average of the 20 songs from each genre to develop the dB/frequency ranges that each genre lies within. In addition to the sound dB/frequencies, we also found the **beats per minute** (**BPM**) of each song through *songbpm.com*. The BPM of music is a prominent feature of music because different songs within the same genre tend to lie within the same BPM ranges (this is the theory behind DJing/music mixing!) This is why we extracted BPM as one of our features for classification. The reason we took averages of these ranges was so we could randomly generate more entries of music for each genre for our data sets to train/test on.


*The following image is a screenshot of the Visual Equalizer in Garageband for an electronic house song sample we used to determine the average dB/frequency range for the music genre.*

With these averages, we constructed three different data sets, all varying in size: one with 50 songs from each genre, another with 100 songs, and finally another with 200 songs from each genre (please refer to `50-50-50train/test.arff`, `100-100-100train/test.arff`, `200-200-200train/test.arff`).

**Generating Noisy Data:**

Next, we decided to tackle the problem of dealing with noise. In order to construct a dataset with noise, we downloaded a background-noise audio file, creating a loud cafe environment while our music was playing, and we analyzed the changes in our frequencies for each genre. Obviously, BPM did not change here since a song's BPM stays true to its rhythm through unwanted noise. We immediately observed changes in the dB/frequencies of each genre, and roughly adjusted each music genre's dB/frequency ranges accordingly to the unwanted noise effect on each genre to construct a noisy test data set.

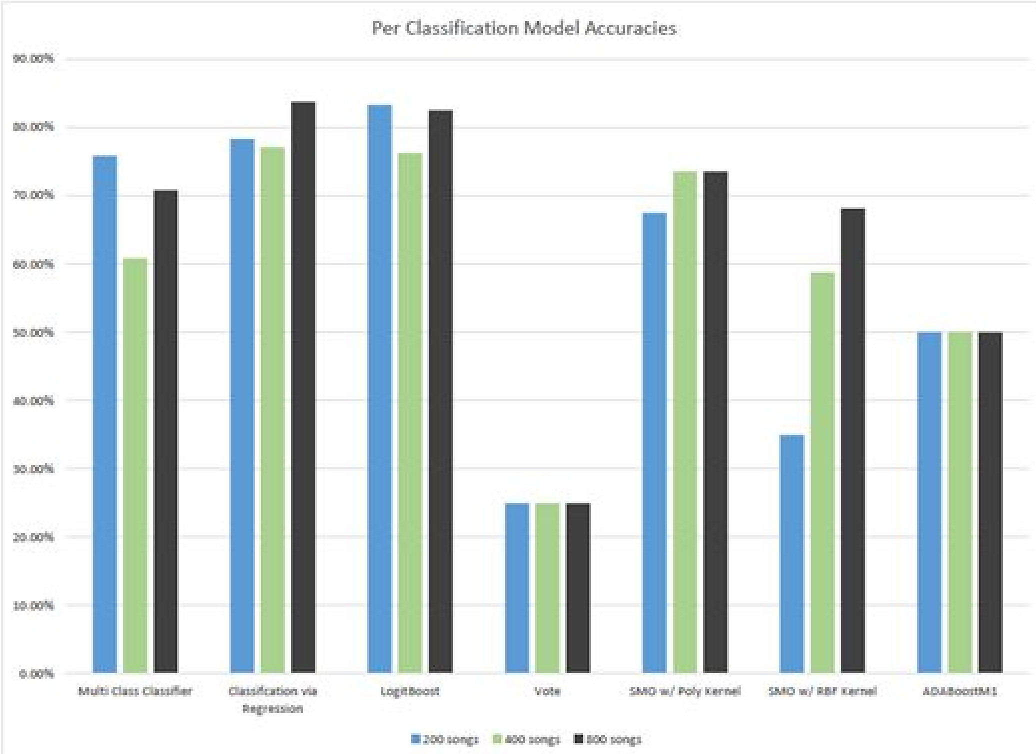

*The following image is a screenshot of the Visual Equalizer in Garageband for the* ***same*** *electronic house song sample however this time with added noise. As you can see, the high mid and treble dB have increased significantly.*

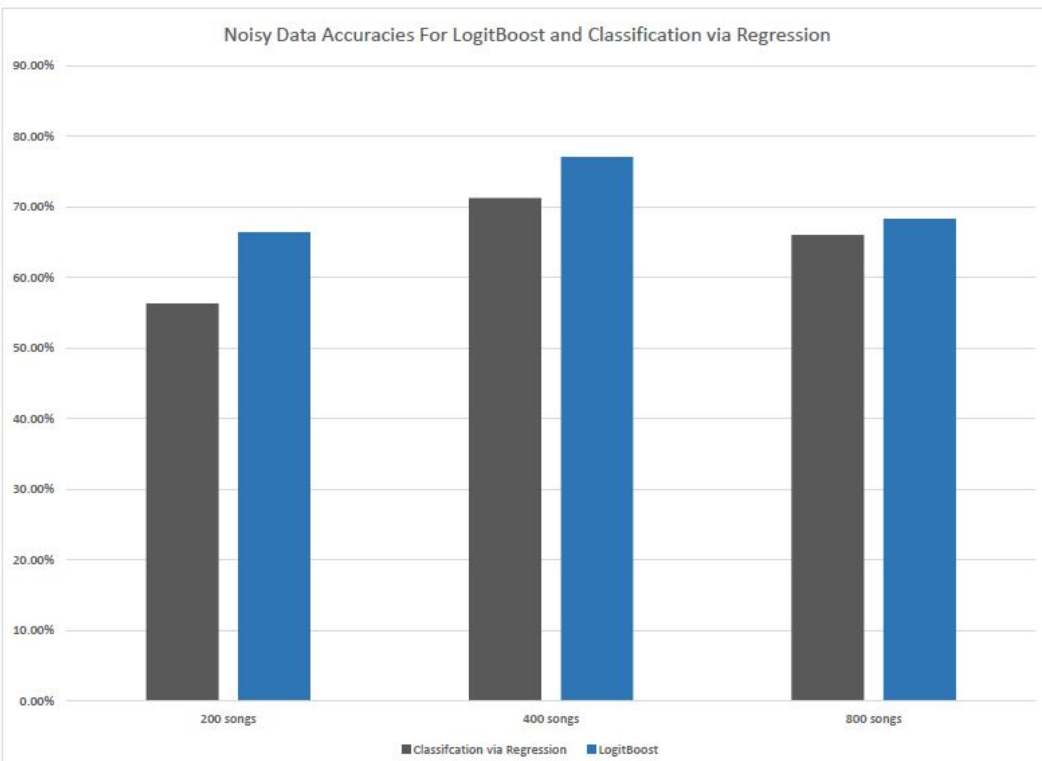
**Training/Testing our Data in WEKA:**

Once we fully constructed our different data sets, we decided to split each set into 62.5% training set and 37.5% testing set (not including our noisy test set). We decided to train and test our data with **7** different classifier models (please refer to the directory named `WEKA` to access the classifier outputs). Our goals were to find an optimal data set for genre classification, find out which models worked best with specific data set sizes, and to find the best working model for noisy data.

**Results and Critical Evaluation:**

Overall, we found that the *LogitBoost* and *ClassificationViaRegression* WEKA classifiers gave us the highest accuracy amongst each data set (200, 400, 800). We believe that the *ClassificationViaRegression* result provided a high accuracy due to each feature lying within a certain range (Regression proves to be useful for these types of data sets i.e. estimating the price of a house). Specifically On the other hand, we found that the *Vote* WEKA classifier gave us the lowest accuracy. We believe this is due to improper use of the classifier, since we received a 25% accuracy on every single data set. Overall, our experiment had a lower accuracy for all data sets and models compared to the those of in the article mentioned later. The biggest factor in receiving lower accuracy across the board can be attributed it having few features. The experiment did not answer our size question. One would expect the accuracy of the model to increase  as the size of the data set increased or decreased. Instead our experiment showed that the data set of 400 songs had the least accuracy and to give us conducive report it should have the middle accuracy. To test our noisy data set, we chose the two models that performed best in our test: *LogitBoost* and *ClassificationViaRegression*. We found that *LogitBoost* provided a higher accuracy for the noisy data, however we were unable to certainly determine which data set size is optimal for a model.


Per Classification Model Accuracies
90.00%
80.00%
70.00%
60.00%
50.00%
40.00%
30.00%
20.00%
10.00%
0.00%
Multi Class Classifier
Classifcation via Regression
LogitBoost
Vote
SMO w/ Poly Kernel
SMO w/ RBF Kernel
ADABoostM1
200 songs
400 songs
800 songs


Noisy Data Accuracies For LogitBoost and Classification via Regression
90.00%
80.00%
70.00%
60.00%
50.00%
40.00%
30.00%
20.00%
10.00%
0.00%
200 songs
400 songs
800 songs
Classifcation via Regression
LogitBoost

**Implementation Outcome:**

Our project was inspired by Charles Tripp's article "Waveform-Based Musical Genre Classification" that tried to implement an automatic musical genre classification system based on features extracted from the waveform of music. Tripp found that more successful designs used "gross spectral measurements" which is the shape of the "frequency-power distribution" and each genre has a unique shape.

Some difficulties we faced in the project was finding a dataset of music and extracting features and data from a song/music file. There are very few music datasets available to the general public due to copyright and other legal issues. We contemplated using *MillionSongDataset* but were unable to figure out how to extract relevant data from the files. Additionally we looked into using GTZAN dataset. The files were collected in 2000 - 2001. Having a dataset over a decade old can cause problems correctly identifying newer musical pieces as there may be different trends in music, a rock song in 2017 may be classified differently based on music in 2001. At first we tried to import the music files from the GTZAN dataset into MATLAB and had difficulty extracting and understanding the useful data. Due to time constraints and complexity we decided to use GarageBand to find ranges. Classification of musical genres is subjective, what one person will classify a song as rock another may classify it as pop.

**Conclusion:**

From doing this project we learned that classifying genres of music is an easy task for humans to accomplish but is rather difficult for machines. There are many features and variables that play into determining the genre that a machine can have a hard time. For example the beats per minute (BPM) of a rap song is constant throughout whereas a classical song has a wide range and nonconstant BPM. Humans can easily identify the song as classical but the machine may misclassify it based on where it got the BPM data.

Future works:

If someone were to continue this project, their first step would be to create a dataset by extracting features directly through software to provide more accurate and varying data. This was lacking in our implementation. Using MATLAB will make it easier and quicker to extract more features like averages of wavelengths, varying BPM and power. Another step to take it is to classify more genres, sub genres and music across different languages.