Progress Report

This week, we met to evaluate the work we had to do and assign tasks to each person. We determined that Erik and Mandela would work together on assembling the data set and extracting features and Joshua and Elana would work on implementing the algorithm. Each group's progress write-up is given below. In summary, Erik and Mandela successfully extracted features from audio, and Joshua and Elana successfully prepared a machine learning framework. In week 2 of development, we plan to integrate these components, resulting in the final music recognition software.

Erik & Mandela

We searched online for samples from which to build our training and test sets from, and we found a database from the University of Iowa that offered high quality recordings of instruments (http://theremin.music.uiowa.edu/MIS.html). We downloaded roughly thirty audio clips for each of the instruments we are planning on testing. Each sample features one of the instruments playing a note for roughly three seconds.

The next task was to convert the sounds files to the correct format. The progam we used to process the audio only accepted .wav files, so we used ffmpeg (http://ffmpeg.org/download.html) to convert each of the audio files, which were originally in .aif format, into .wav files.

Next we researched the best way to extract features, and we found a property known as Mel-Frequency Cepstrum Coefficients (http://en.wikipedia.org/wiki/Mel-frequency_cepstrum). They are frequently used in speech recognition and audio processing, and they provide an indicator of the timbre of the sound. Conveniently, we also found a python library that reads in a .wav file and outputs a matrix corresponding to the MFCCs of the sample (http://python-speech-features.readthedocs.org/en/latest/). The matrix contains 13 columns and rows proportional to the length of the audio, typically one per 10ms of audio.

We downloaded and installed this library, and then we loaded each of the sound files and used the program to extract the MFCCs for each files. We are still determining how to best utilize the coefficients in our machine learning algorithm. We think we can either average the values in each column or feed in the entire matrix. For now, our program outputs the MFCCs for each file into a json file, which the machine learning portion of the program can then use.

Joshua & Elana

We researched specific machine learning techniques to determine which would be most appropriate for this project. First, we determined the appropriate libraries, deciding that Scikit-learn would be best suited for the type of work we were conducting. In particular, the thorough Skikit-learn documentation was very helpful for our continued work.

Next, we decided upon the appropriate algorithms. Since we planned to write code that would distinguish between instruments, and since the data sets we would be using were already labeled (see Erik & Mandela's part above), we decided to use a supervised learning algorithm known as the SVM. Using feature vectors extracted from the audio data, an SVM represents the data in multi-dimensional space and determines a hyperplane which decides between the data. This gives binary results, but that works well for our project — the algorithm works separately for each instrument. We get a binary reading for each instrument — we return to the user the instrument for which the SVM returned "yes".

There is, of course, the possibility that the SVM will return multiple values. We will investigate a solution to this case in week 2 of the project.

Before integrating the feature vectors extracted from Erik & Mandela's component of the project, we designed and built our SVM using dummy data. In particular, we gave the machine test data in the format (x,x) -> x. For example, (3,3) -> 3 and (1,1) -> 1. Essentially, we are extracting the x-component = y-component of a point on the line y=x. We wanted our machine to find the closest point on the line x=y.

Scikit-learn examples involving SVMs were particularly important for this development. In particular, we began our project inspired by facial recognition algorithms, since the algorithm problem is very similar to music recognition.

In week 2, we will work toward advancing our SVM to identify music.