By: Marwan Harajli
Last Edit: 11/10/2018
Subject: Data Wrangling in Springboard Capstone Project 1.

1-Introduction:

This report discusses certain data wrangling procedures used on the Urban Noises dataset. The urban noises dataset consists of 8732 .wav files representing noises from one of these 10 settings: jackhammer, engine idling, siren, air conditioner, children playing, street music, dog barking, drilling, car horn, or gun shot. Each .wav file has a unique integer id and is used to name the file (23.wav as an example). The dataset also includes a .csv file that defines the class (noise setting) for 5435 of these .wav files. The .csv file consists of 5435 rows and 2 columns where one of the columns lists the id's of the sounds, and the second their class.

The link provided outlines a proposal to use the 5435 .wav files to train a classification model and test the model with the rest of the files.

2-Reading the Data Set:

To make use of these .wav files, the library *soundfile* is used to load each .wav file as a numpy array (represented as a mono sounds as opposed to stereo) accompanied by a single integer representing the sampling rate. The sampling rate defines the number of samples a given array provides per second. The reason for using soundfile boiled down to its ability to read 24 bit depth audio files (which was the case for some of the .wav files and scipy and librosa were not capable of loading them).

With this, the dataset has been transformed to 8732 numpy arrays. Each array has the following associated characteristics:

1. ID
2. Sampling Rate.
3. Duration (calculated as length of the array divided by the sampling rate)
4. Class (if available).

A pandas dataframe called PropsDF was compiled that summarizes these characteristics for all loaded numpy arrays. The head of the dataframe is shown below. The code used to compile this dataframe along with all code used for purposes related to this report can be found in this github repository in the data_wrangling ipython notebook.

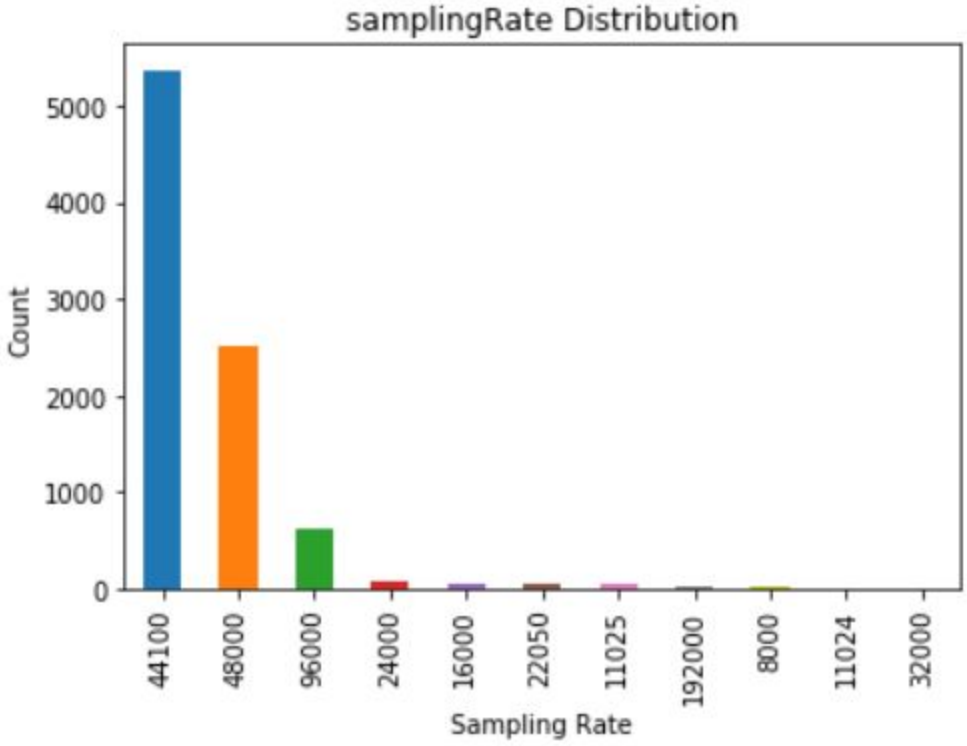
	ID_x	SR	length	Max	Min	Average	Duration	RealID	ID_y	Class
0	train\0.wav	44100	176400	0.477905	-0.522919	-0.000102	4.0	0	0.0	siren
1	train\1.wav	48000	192000	0.391418	-0.390747	-0.000525	4.0	1	1.0	street_music
2	train\10.wav	44100	176400	0.537064	-0.554016	-0.001036	4.0	10	10.0	street_music

In what follows the distribution of these characteristics over the data is presented.

3-Exploring the Data:

3.1-Sampling Rate:

We begin by exploring the sampling rate (SR) of the .wav files and present its distribution in the following bar graph (the generation of this graph can be found in the github repo mentioned above) .



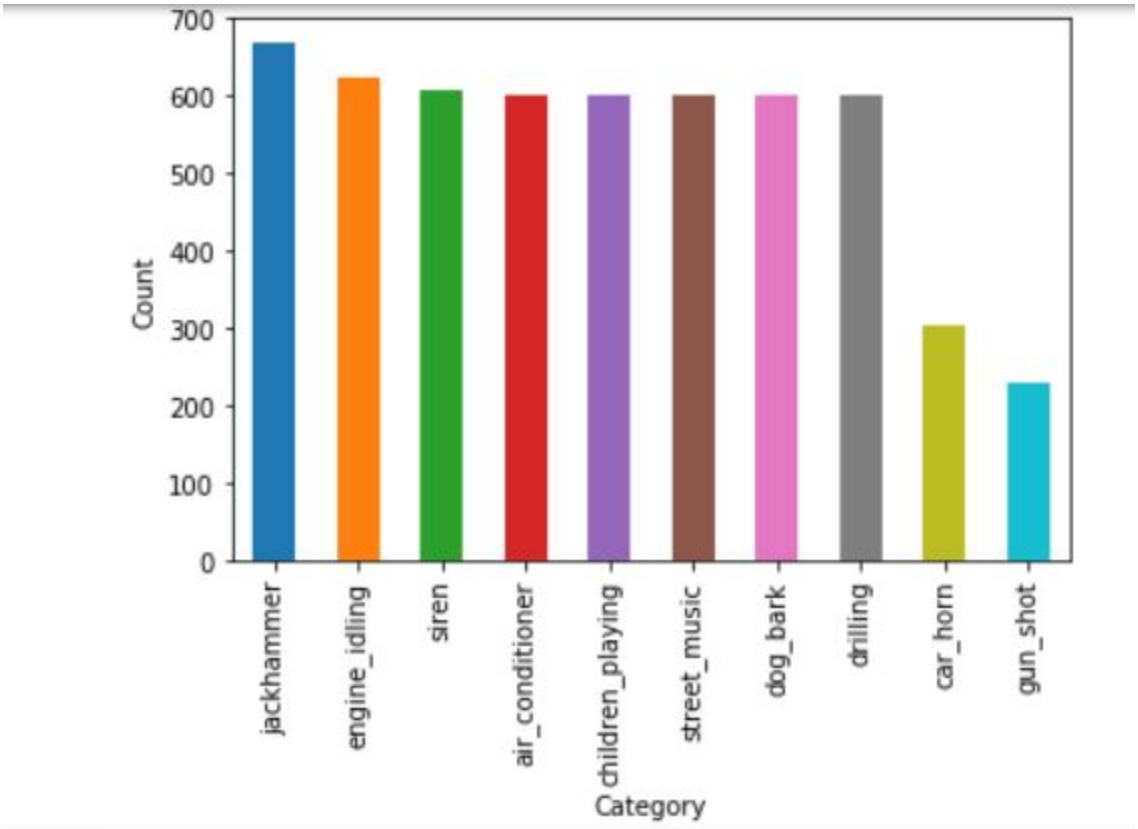
As can be seen, most (about 60%) of the .wav files have a SR of 44100 while the rest have SRs varying from 8000 to 192000.

It should be mentioned that an array representation of a sound with a higher sampling rate involves using longer arrays. It can thus be beneficial to use smaller sampling rates to reduce memory and processing demands. Using very small sampling rates however is also unfavorable as the arrays become too coarse to well represent the sounds. Finally, making all the arrays have the same sampling rate adds a level of homogeneity that can be favorable.

For the purposes mentioned above, the librosa library is used to resample every loaded array into a 22050 SR representation using the resample method.

3.2 Classes:

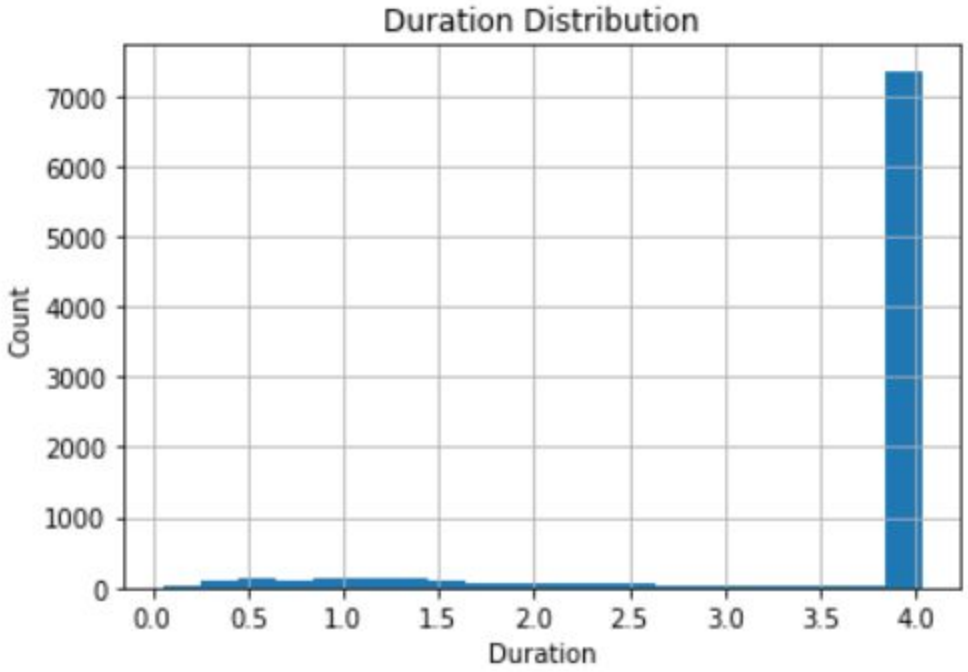
The distribution of classes over the training data is shown in the bar graph below.



As can be seen, the data is mostly balanced with the exception of car horn and gun shot sounds being under represented.

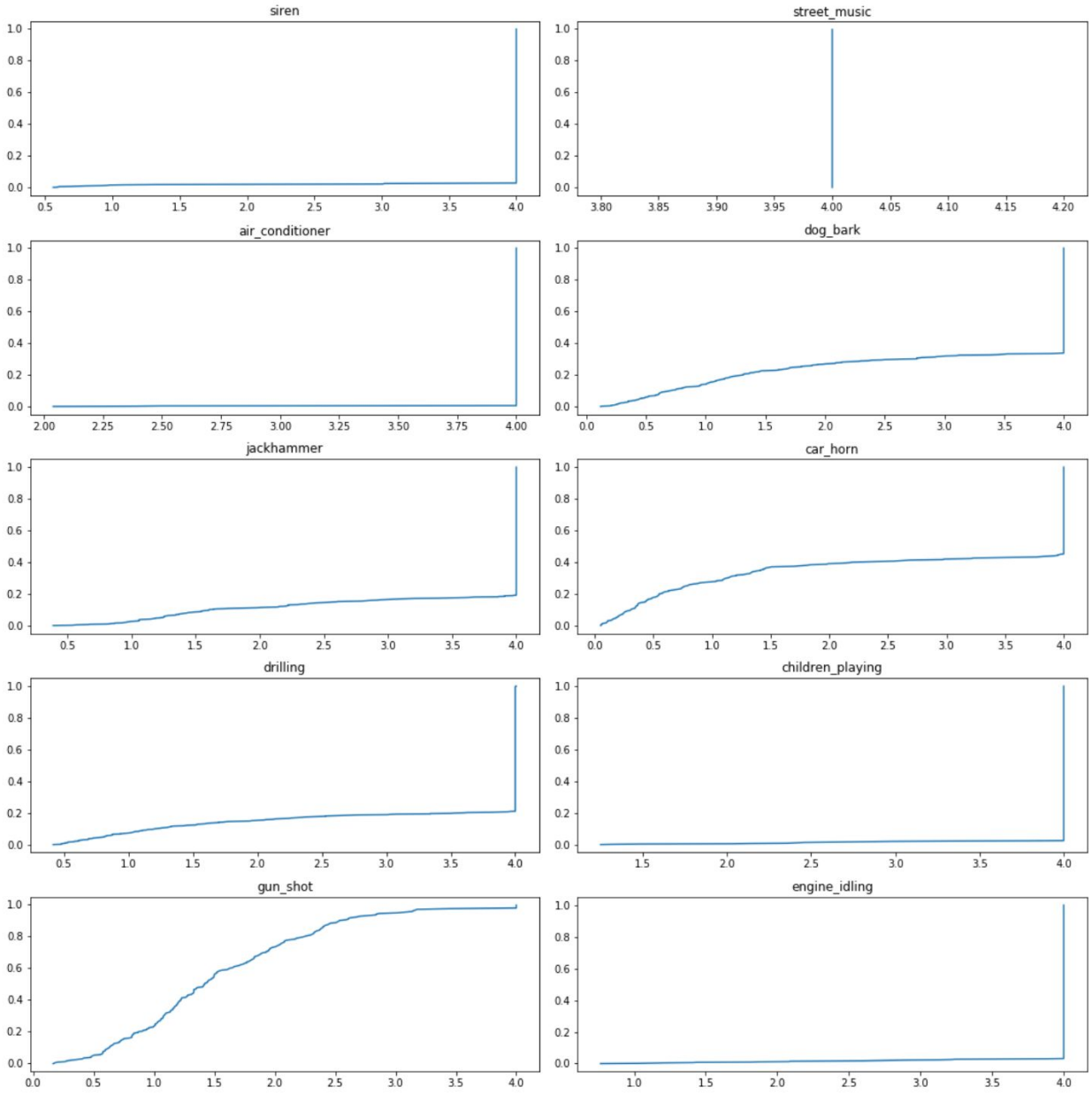
3.3-Duration:

In this section the duration of each audio file is discussed. The distribution of durations is shown in the graph below:



We see that the majority of the audio files have a duration of 4 seconds with the rest having smaller durations.

As mentioned in the capstone project 1 proposal, features are to be extracted from these arrays and used in training a classification algorithm. Some of these features (spectrograms as an example) can be sensitive to the duration of the sound file. A longer sound produces a longer spectrogram and vice versa. This can have different sound files producing features that have different dimensions which can cause problems in making these features useful for training a classification algorithm. It would thus be useful to consider only looking at the sounds that have a duration of 4 seconds. However before doing so, we must check that no one class is majorly affected by doing so. To check that, cumulative distribution functions are drawn for the Durations of each individual class and presented below:



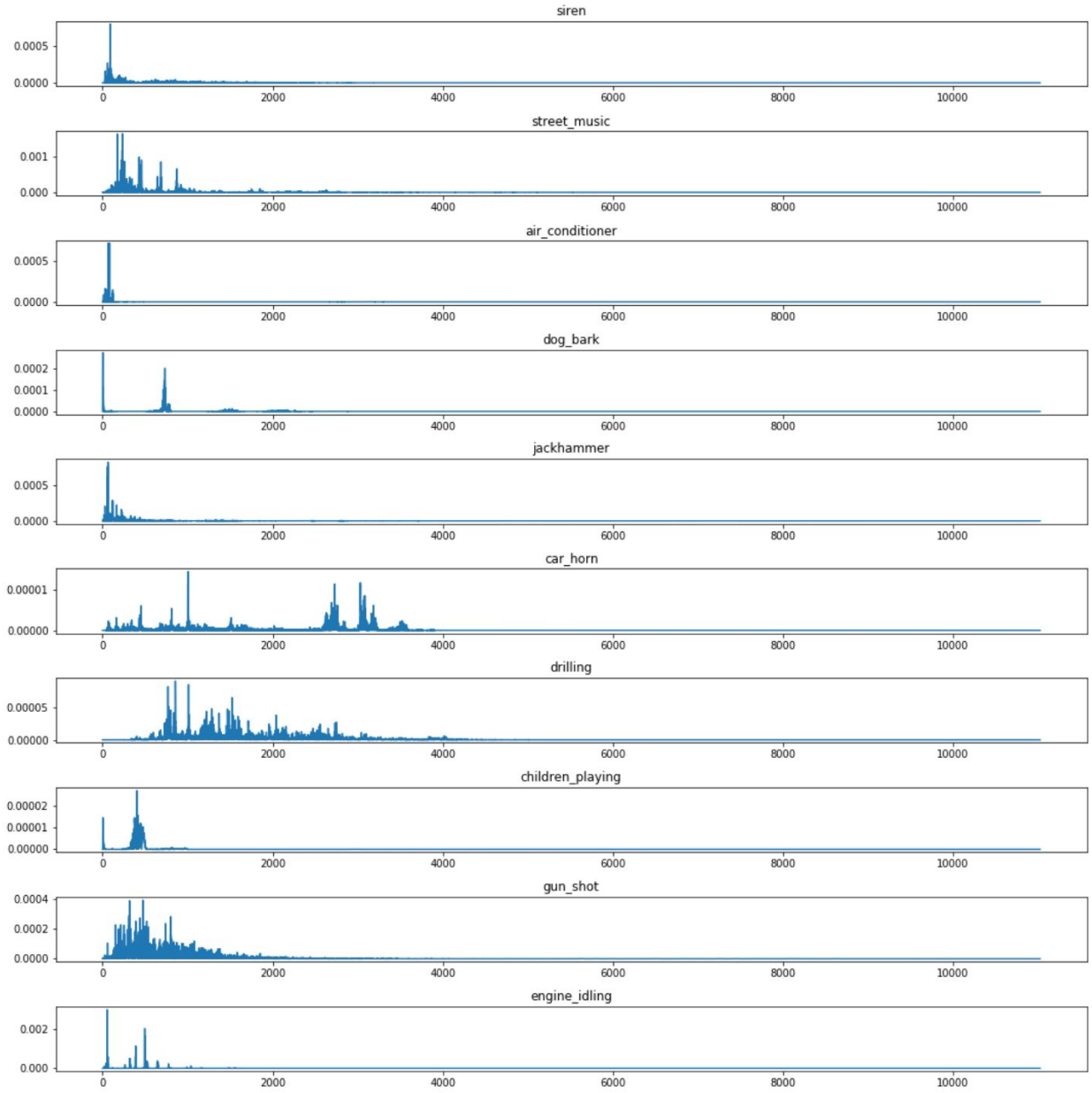
As can be seen from the above CDFs, gun shot audios have the majority of their durations less than 4 seconds. Car horn audios also have 40% of their durations less than 4.0 s. With that, deleting all audios with durations different than 4.0 seconds becomes problematic.

4- Features:

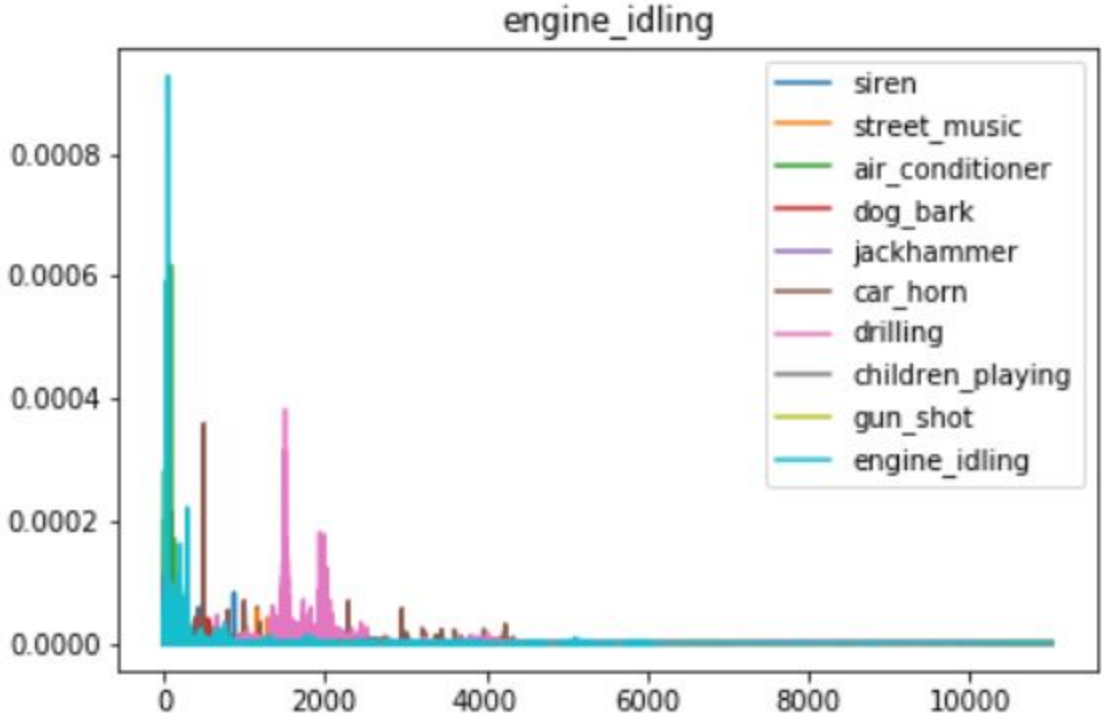
In this section we look at features that can be obtained for the arrays using scipy and librosa. We show these features for all 10 classes to visually inspect if these features help differentiate between different sounds.

4.1-Scipy Periodogram:

The periodogram of an instance of each class is obtained using the scipy.signal.periodogram method. A periodogram is an estimate of the spectral density of a signal. Below, separate subplots containing the periodogram of each is shown.

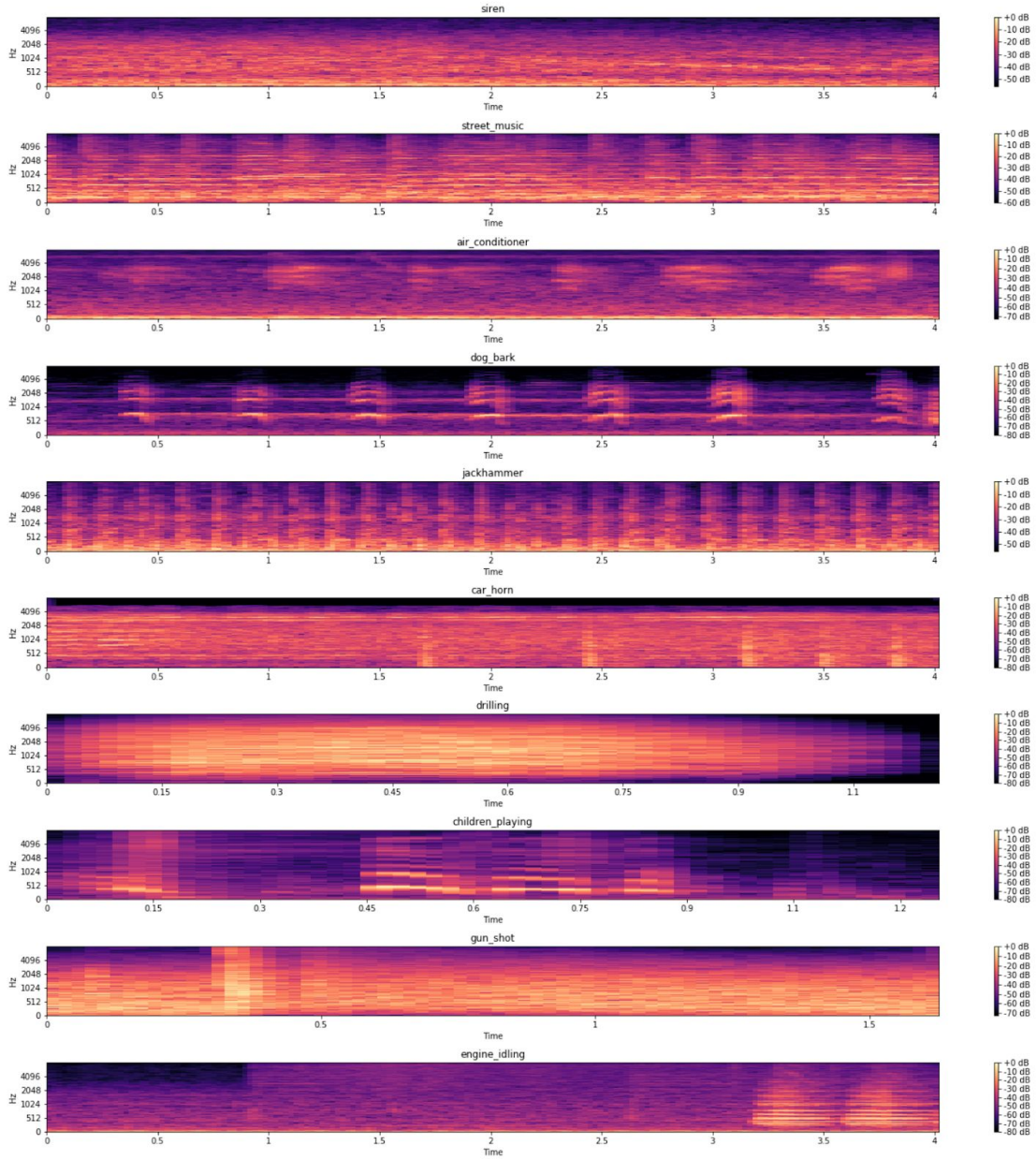


In this plot, the above shown periodograms are joined into one plot to show y-ordinate differences.



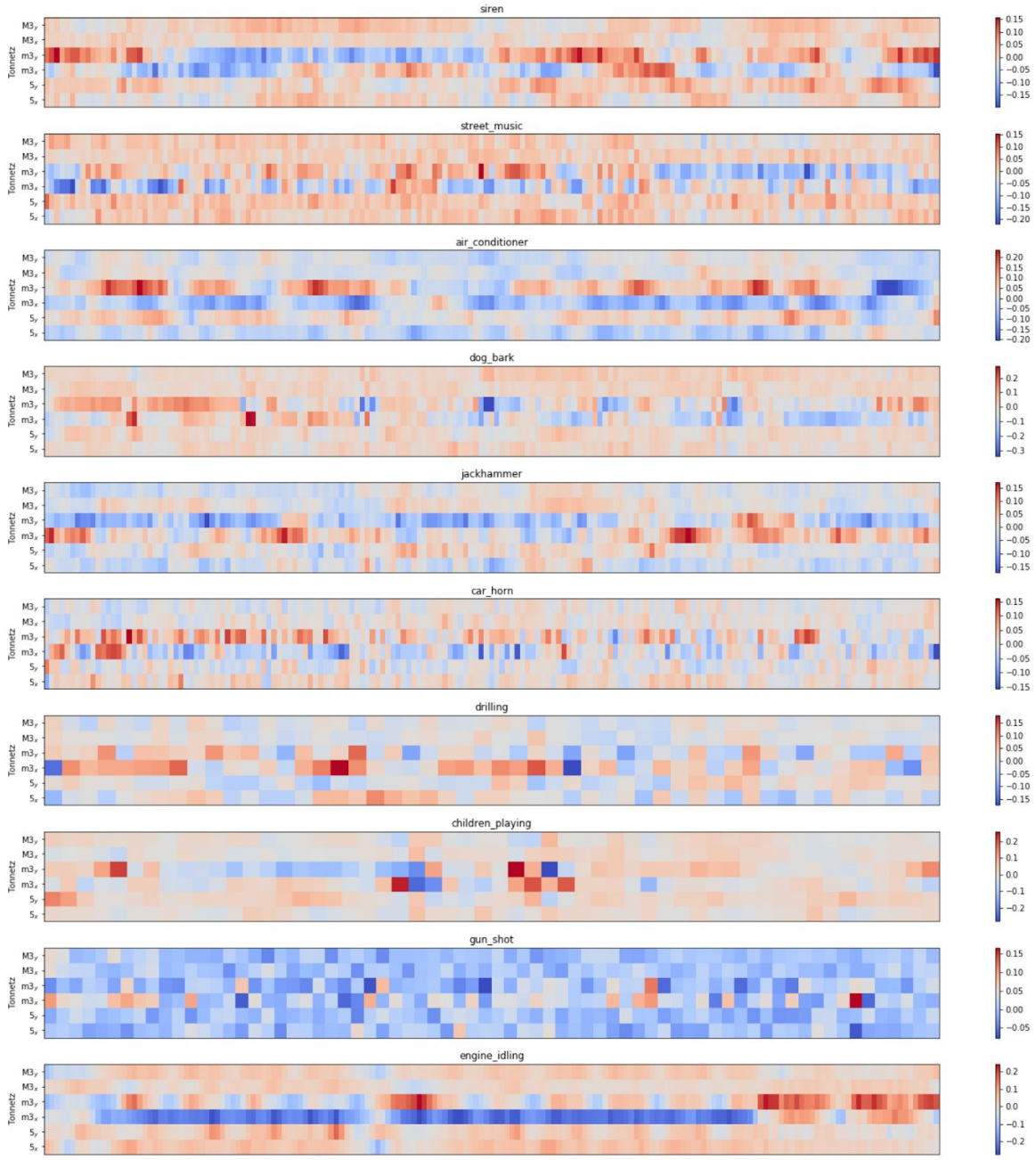
4.2-Librosa mel-spectrograms

The mel-scaled spectrogram of an instance of each class is obtained using the librosa.feature.melspectrogram method. A mel-spectrogram is representation of the power spectral density (at each time), with the frequency axis transformed to the mel scale. Below, separate subplots containing the spectrogram of each is shown. A clearer image can be found here.



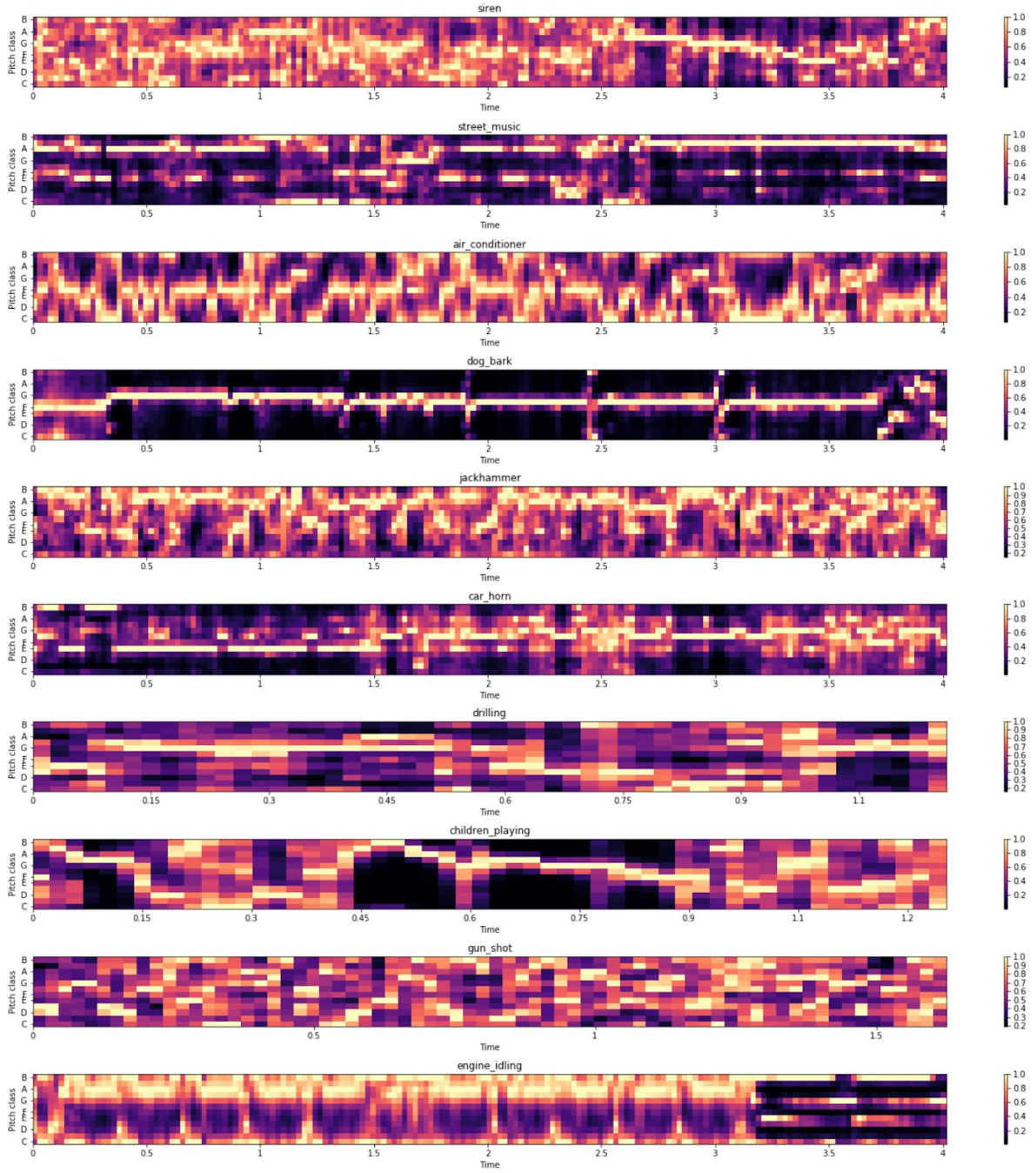
4.3-Librosa Tonnetz:

The tonal centroid features (tonnetz) of an instance of each class is obtained using the librosa.feature.tonnetz method. Tonnetz centroid features estimate tonal centroids as coordinates in a six-dimensional interval space. Below, separate subplots containing the tonnetz features of each is shown. A clearer image can be found here.



4.4-Librosa Chroma:

The chromogram of an instance of each class is obtained using the librosa.feature.chroma_stft method. Chromogram of a short-time fourier transform projects into bins representing the 12 distinct semitones (or chroma) of the musical octave. Below, separate subplots containing the chroma features of each is shown. A clearer image can be found here.



5- Conclusions:

This report outlines the obstacles met in creating useful data to train a classification algorithm.

The data, in the form of .wav files, is transformed into numpy arrays of different lengths (depending on the sound duration) but identical sampling rates of 22050 samples/second. Identical sampling rates over all the data is deemed favorable to decrease memory and processing demands.

The deletion of sound files with durations less than 4.0 seconds (the most common sound duration) is investigated and found not to be feasible mainly due to almost all gun shot sounds being less than 4.0 s in length.

A few sample features are extracted from the arrays using methods from the two libraries librosa and scipy. These features are visually shown and presented in this report. With that, the sound files are ready to be used to train a classification algorithm.