By: Marwan Harajli
Subject: Springboard Capstone project 1 Milestone Report.
Last Updated: 12/16/2018

1- Problem Statement:

In this report we discuss the progress achieved in the problem of classifying urban noises. More precisely, given .wav files each representing one of ten sounds/classes (Air conditioner, car horn, children playing, dog barking, drilling, engine idling, gun shot, jackhammer, siren, and street music), we investigate writing algorithms that can identify the respective classes of these files.

To achieve the above, we proposed using a supervised learning setting where training data would be used to train a classification algorithm.

In what follows we discuss the obstacles met in handling the data (the sounds), cleaning the data, and extracting features from the data. Finally, we discuss the usability of the extracted features.

2- Project Importance:

This project presents itself as an educational opportunity in tackling unstructured data. Since the data is unstructured, useful features need to be found and extracted. Moreover, due to the nature of the unstructured data (being audio data), the project also presents an opportunity to delve into audio processing. The latter has a wide array of applications that ranges from speech processing to recommending music to radio stations. With that, the project yields the potential to be a good data science exercise that also opens the door to the exciting field of audio processing.

3- The Data:

The data consists of 8732 .wav files with 5435 of them labeled (with one of 10 labels) . The files can be found in this link. 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 sound 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.

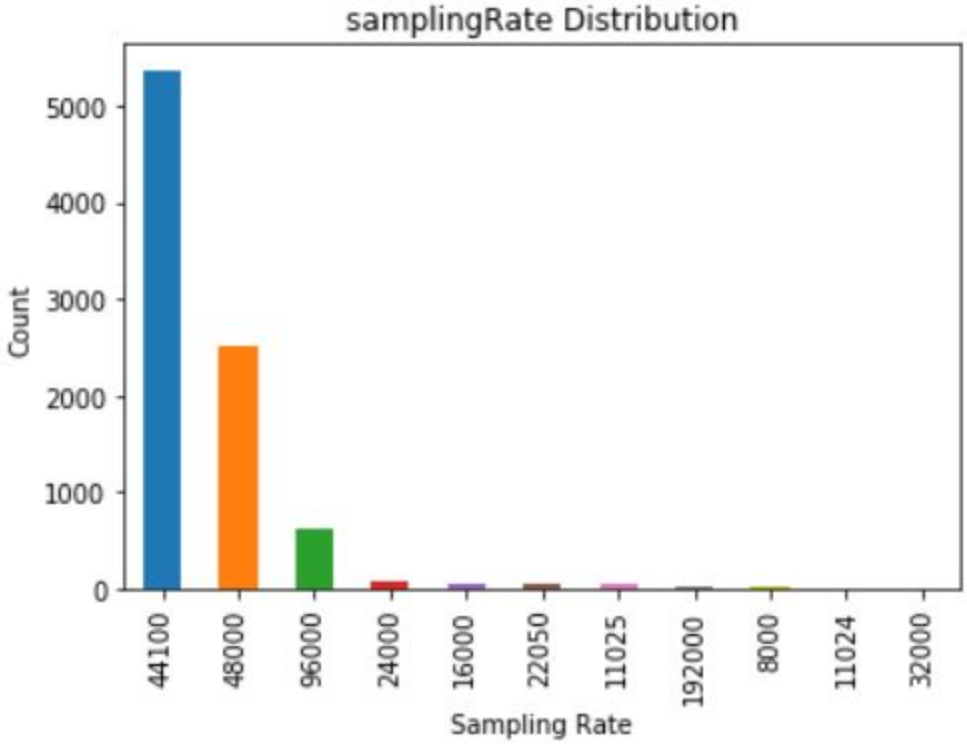
The reason for using soundfile boiled down to its ability to read 24 bit depth audio files (which was the case for some of the .wav files and consequently scipy and librosa were not capable of loading them).

4- Data Wrangling and Cleaning:

In the process of understanding the data and setting it up for classification, we investigated two properties of relevance to each sound file: sampling rate and duration.

As seen in Figure 1, sampling rates vary over the data. Note the sampling rates on the horizontal axis are listed in order of appearance in the data rather than numerical value.

Figure 1: Sampling Rate Distribution



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.

Figure 2 shows the variation in duration over the sound files. The majority of the files have 4 s duration with a small number having smaller durations. This instigates the question regarding the possibility of deleting these files with smaller duration to avoid arrays of different lengths. To answer that question we check that no one class is majorly affected by doing so. Consequently, cumulative distribution functions are drawn for the Durations of each individual class and presented in Figure 3.

Taking a closer look at Figure 3, we see that 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.

Figure 2: Duration Distribution

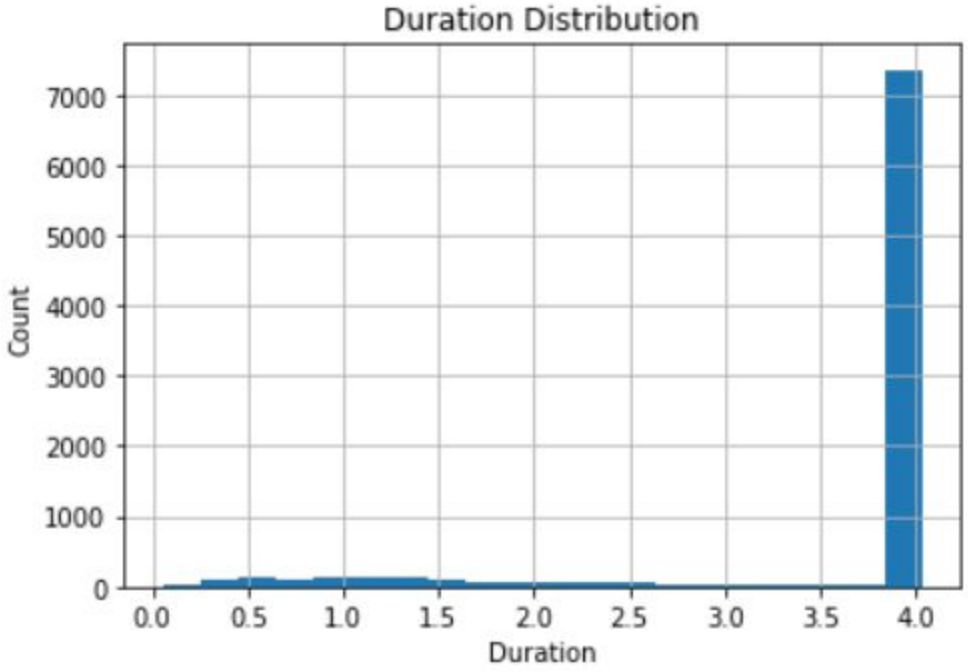
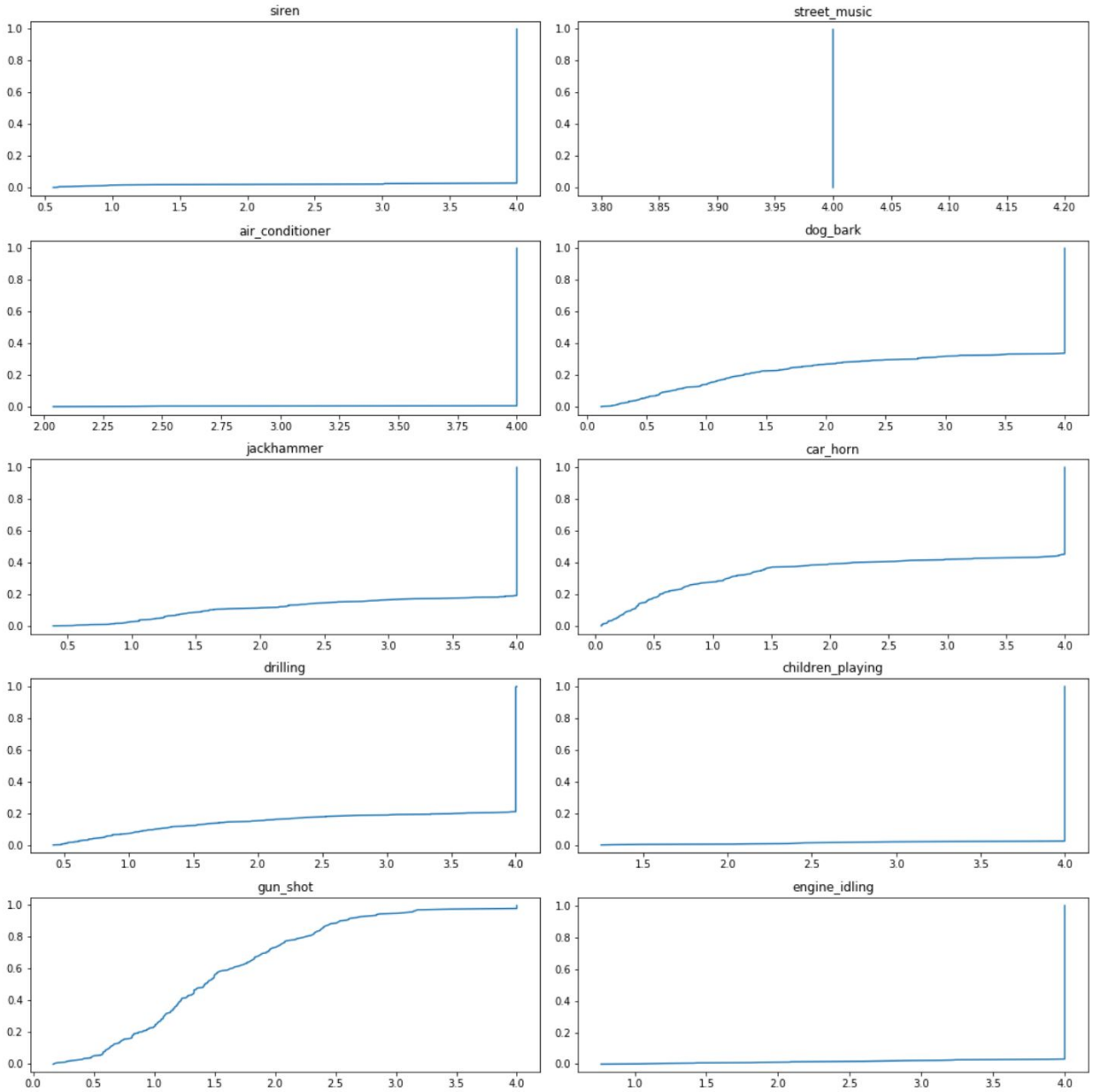


Figure 3: Duration CDFs for Different Classes



5- Feature Extraction:

The following features are extracted from each array:

A- Mel-frequency cepstral coefficients.
B- Mel-scaled spectrogram.
C- Chroma of a short-time Fourier transform.
D- Octave based contrast.
E-Tonnetz coefficients.

A description of each of these features can be found in the data wrangling report.

Each of these features fi (i in [1,2,3,4,5]) is considered as a transformation that take an array s (the sound) of size n, and returns a matrix Mi of size mi*ki (n,mi,ki are non-zero positive integers).

$$f_i : R^n \rightarrow R^{m_i \times k_i}$$
$$s \rightarrow M_i$$

ki is dependent on n (i.e. the duration of the sound), while mi is dependent on the transformation itself but independent of the duration (for example how many mel bins are chosen for the mel-scaled power spectrogram). In this report, we explore the possibility of temporally averaging each Mi to obtain a mi*1 vector.

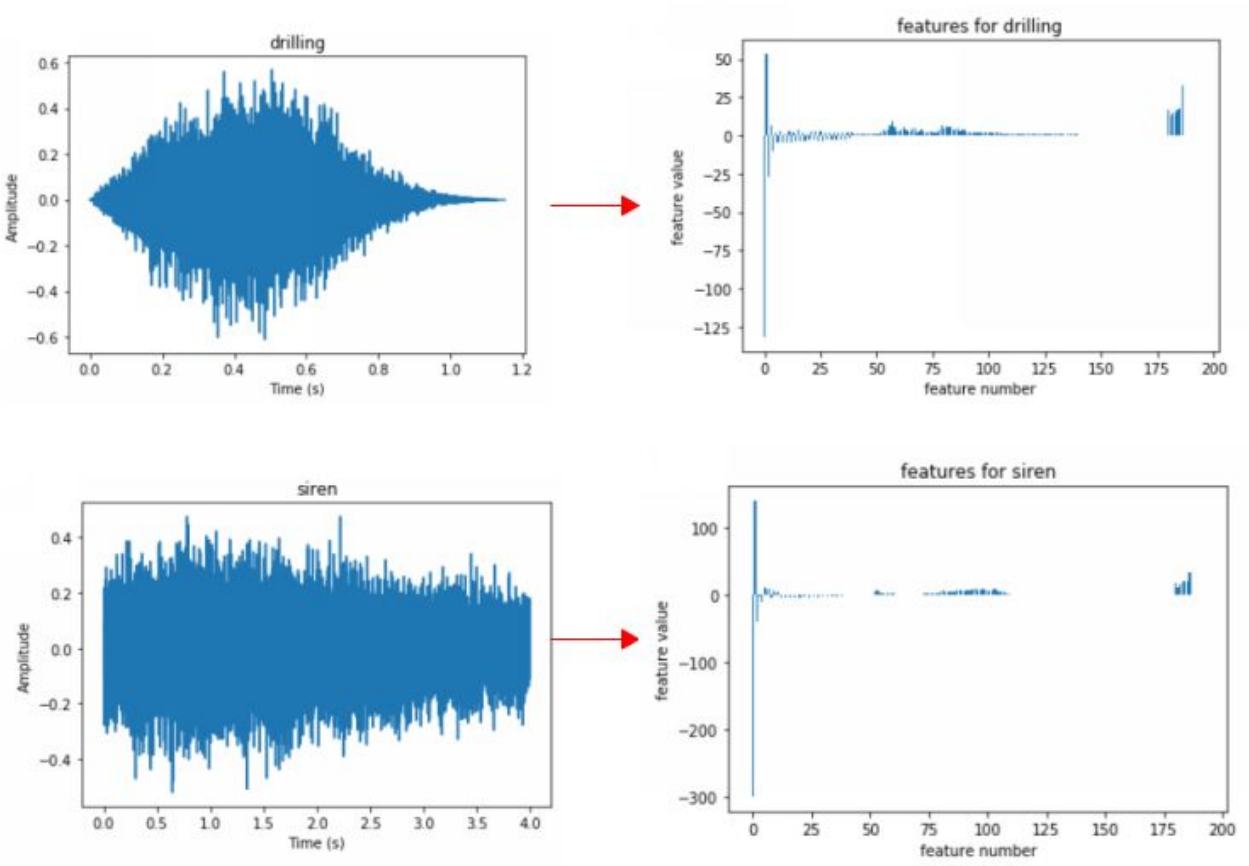
$$f_i^{'} : R^n \rightarrow R^{m_i}$$
$$s \rightarrow M_i^{'}$$

Finally, stacking these 5 $M_i^{'}$ vectors above each other, each sound obtains a feature (not to be confused with the features of the data wrangling report) of length:

$$length = \sum_{i=1}^{5} m_i$$

With this strategy, sounds with different durations produce features of identical lengths. Figure 4 shows the features extracted from two different sounds.

Figure 4: Feature Extraction for a Drilling sound and a Siren sound



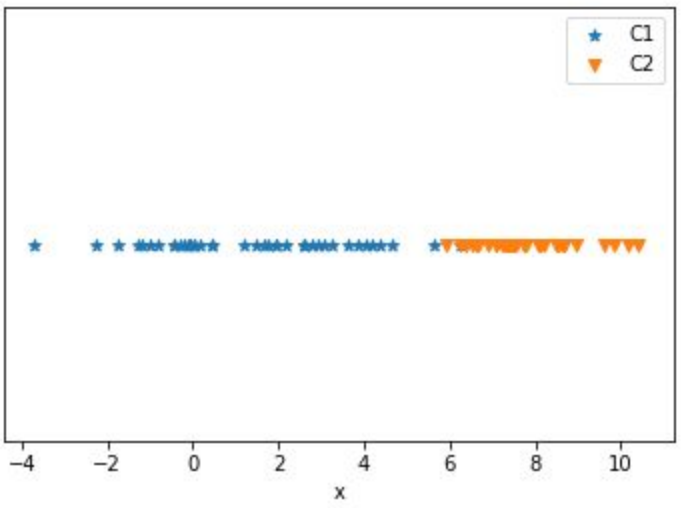
6- Assessment of Features:

In this section we assess the usability of the extracted features. To perform this assessment we propose a probabilistic framework for predicting classes and see how well it performs with the features we have extracted.

6.1 Probabilistic Framework:

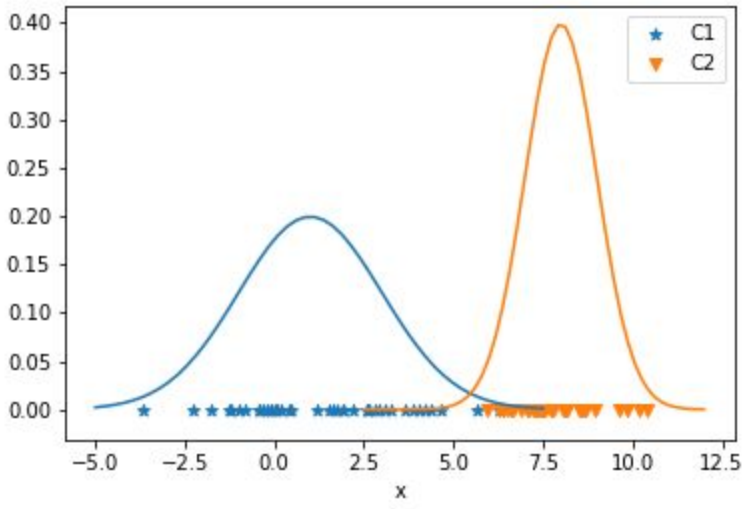
Assume we have a scenario where two classes C1 and C2 exist. These classes are to be predicted using a single feature x. A set of x values and their corresponding classes are given as shown in Figure 5.

Figure 5: Data Obtained



These points are in fact generated assuming each class has its feature distributed according to a distinct normal distribution as shown in Figure 6. C1 has x normally distributed with mean 1 and std 2 while C2 has x normally distributed with mean 8 and std=1.

Figure 6: Distribution of feature for each class

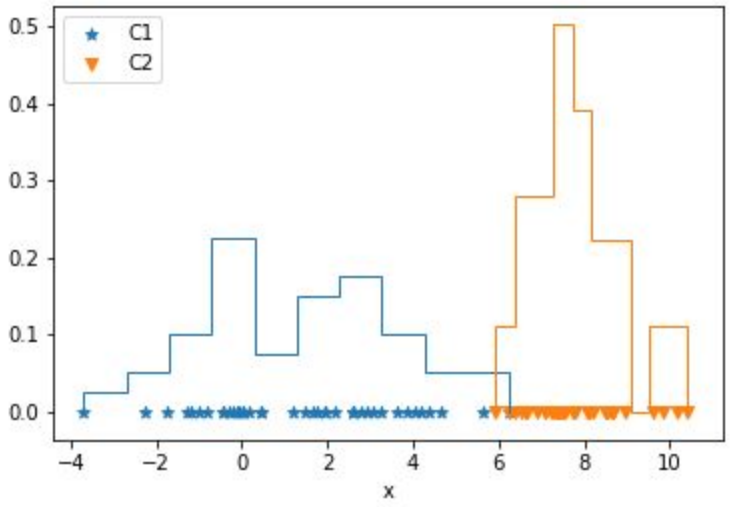


Now given a new observation, say x= 7 we wish to predict what class this observation belongs to. A likelihood estimator for (Ci|Observation) is pdf(observation|Ci) where pdf is short for probability density function. Thus evaluating pdf(observation|C1) and pdf(observation|C2) and comparing them we can conclude the most likely class this observation belongs to; in this case C2 (since pdf(x=7|C1)=0.0022 and pdf(x=7|C2)=0.242).

Up to this point however we have assumed we know the underlying distribution for the feature for each class. This is not true in our data set. Rather we need to use the data at hand to produce estimates for pdf(x|Ci).

This can be done using a normed histogram to get an estimate of the pdf as shown in Figure 7. With this estimate, we see that once again x=7 is classified as C2 since the estimate for pdf(x=7|C2) is greater than the estimate for pdf(x=7|C1).

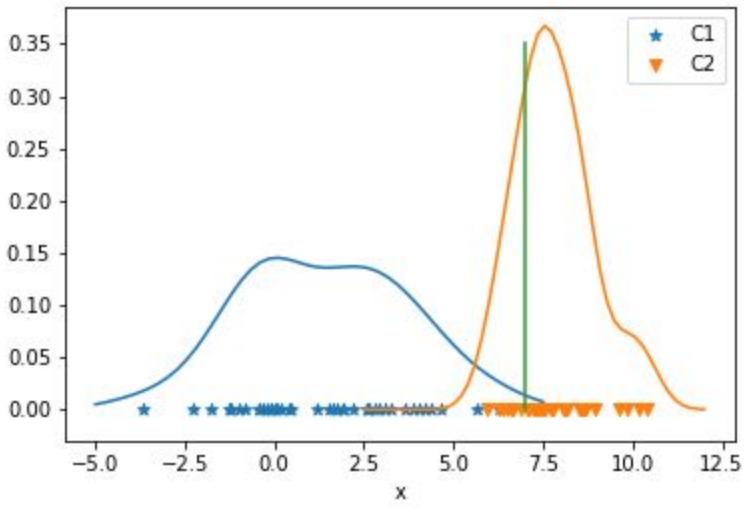
Figure 7: Estimate of Distribution of feature for each class using histograms



Histograms can be computationally tough to generate for high dimensional features however. Consequently we try estimating pdfs using kernel density estimators (using the scipy library with the stats.gauusian_kde method).

Figure 8 shows the estimates for the pdfs using kernel density estimators. The figure also shows the line x=7 to show that the estimate for pdf(x=7|C2) is higher than the estimate for pdf(x=7|C1).

Figure 8: Estimate of Distribution of feature for each class using KDE



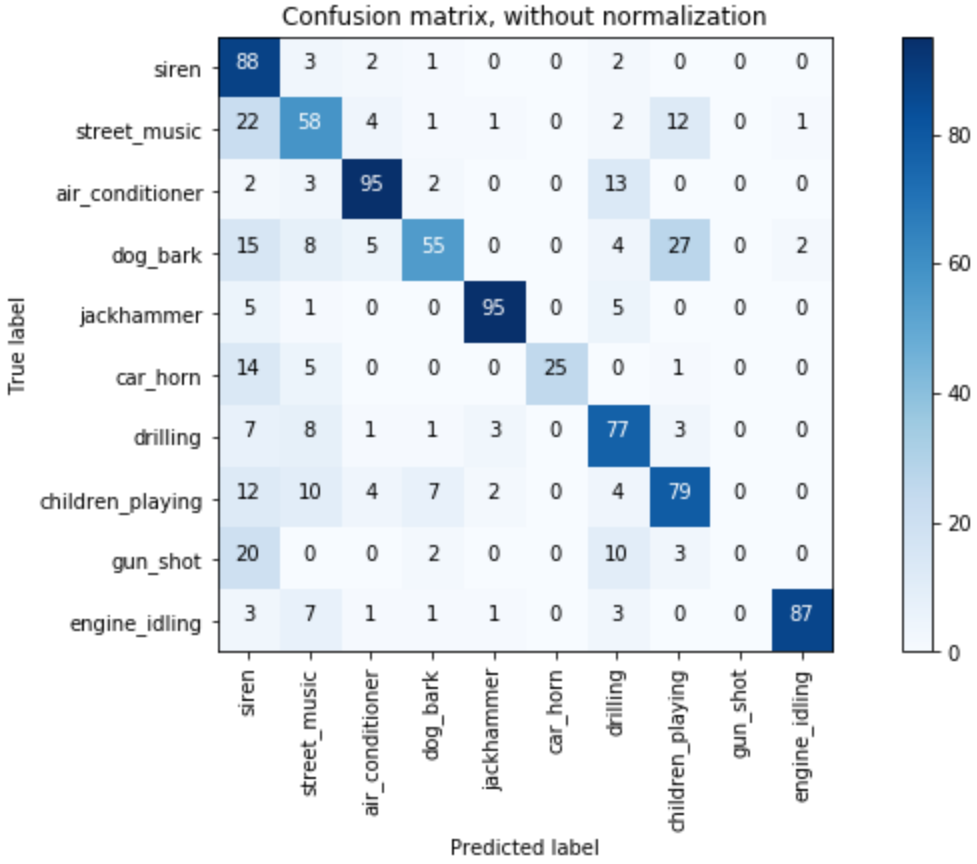
6.2 Application to Sound Extracted Features:

In our dataset, each labeled sound is transformed into a feature vector of 193 dimensions. We split the data into a training set (approximately 4500 sounds) and a testing set (approximately 1000 sounds). Using kernel density estimators, we obtain pdf estimates for the features over each class using the training data. With that we can estimate pdf(feature|Ci) for every i and for every feature in the test set. Finally, for each sound in the testing set, the class Ci that maximizes the estimate for pdf(features of sound|Ci) is assigned as the predicted class of the sound.

The above procedure yielded a 70% accuracy rate in predicting the right class for the sounds in the testing set. Figure 9 summarizes the results obtained with a confusion matrix.

Since this method yielded acceptable results, we can conclude that the features extracted do provide information that help in a classification scheme. We should note that this does not mean that every feature (of the 193) provides information that helps classify the sound, rather all 193 together do.

Figure 9: Confusion Matrix for Probabilistic Classification



7- Conclusions and Work to be Completed:

In this report we discussed the progress so far in identifying 10 different noises. In summary:

1. The data was obtained as .wav files.
2. The .wav files were transformed into numpy arrays.
3. The numpy arrays were all altered to represent the sounds with a 22050 sampling rate.
4. A 193 dimensional feature vector was extracted from each numpy array.
5. The usability of these features was assessed by testing the performance of a probabilistic classification setting.

With the above, we can now use these features to train more involved classification algorithms.