Practical 3: Classifying Sounds

Camelot submission closes at 11:59pm on Thursday, April 5th, 2018.
Writeup due 11:59 on Friday, April 6th, 2018

You will do this assignment in groups of three. You can seek partners via Piazza. Course staff can also help you find partners. Please see `practical-logistics.pdf` for a description of competing on Camelot.ai, what to submit to Canvas, and more. Make sure to use the provided LaTeX template for your writeup.

Audio classification is key to developing machine-listening systems. In this practical, you will classify sounds recorded using microphones around a city into 10 classes. Your predictions on some of the unidentified sounds will appear on the public leaderboard, and your predictions on the remaining unidentified sounds will appear in the private leaderboard. In making your predictions, you will primarily have at your disposal a series of amplitudes sampled for each sound. The classes of sounds under consideration in this practical are: `air_conditioner`, `car_horn`, `children_playing`, `dog_bark`, `drilling`, `engine_idling`, `gun_shot`, `jackhammer`, `siren`, and `street_music`.

Data Files

There are 3 files of interest, which can be downloaded from the practicals repository:

- `train.csv.gz` – This file contains information about the 6325 sounds in the training set. It has 88,201 columns. The first 88,200 columns are the sampled amplitudes. The last column is the sound class.
- `test.csv.gz` – This file contains information about the 1000 sounds in the test set. It has 88,201 columns. The first column is the sample ID. The other 88,200 columns are the sampled amplitudes. You will be making predictions on these sounds.
- `sample_predictions.csv` – A sample submission file. You will produce a similar file. The format is comma-delimited, with the first column being the sample ID, and the second column being your class prediction, an integer between 0 and 9 (inclusive).

  ```
  Id,Prediction
  0,0
  1,1
  2,2
  ...
  998,8
  999,9
  ```

  The class numbers correspond to the predicted classes, in alphabetical order; see table below.

Class Distribution

The distribution of sound classes in the training data is approximately as follows. It may be worthwhile to keep in mind that some classes are very infrequent.

0	`air_conditioner`	27.84%
1	`car_horn`	2.36%
2	`children_playing`	11.29%
3	`dog_bark`	7.78%
4	`drilling`	9.42%
5	`engine_idling`	10.99%
6	`gun_shot`	0.21%
7	`jackhammer`	9.03%
8	`siren`	10.02%
9	`street_music`	11.07%

Evaluation

The evaluation metric for this practical is categorization accuracy. That is, you will be scored on the percentage of the test executables that are correctly classified. In math:

$$\text{Categorization Accuracy} = \frac{\text{Number Correctly Classified Examples}}{\text{Total Number of Examples}}.$$

Sample Code

In the practical repository, you will also find the file `generate_spectrograms.ipynb`. This file is by no means required, but it provides one potentially useful transformation of the data. Specifically, it converts amplitudes sampled for each sound into spectrograms, which are 2D visual representations of sound that plot time windows along the x-axis and frequencies along the y-axis. If you plot the spectrogram of a sine wave at a constant frequency of 440Hz, it will appear as an (almost) horizontal line in the spectrogram as shown in the iPython notebook. Make sure to `pip install librosa` to run this code. We hope this will get you started as you think about feature representation!

Baselines

You will find two baselines on the leaderboard, the Logistic Regression baseline and the Random Forest baseline. Though we have not provided the code for these, they are simple approaches that you are probably well acquainted with by now.

Solution Ideas

As in the previous practicals, you have a lot of flexibility in your feature engineering and modeling choices. The extra challenge here is that you are dealing with time series data. What kinds of models or transformations might appropriately take time into account? Here is a list of classification techniques to get you thinking:

- **Logistic regression on basic features:** As always, a good place to start is to turn the data into some sort of vectorial feature representation and use a logistic regression technique.
- **Use a generative classifier:** You could build a model for the class-conditional distribution associated with each type of sound and compute the posterior probability for prediction.
- **Use a neural network:** If you think there isn't enough flexibility, you could implement a multi-layer perceptron (or some variation thereof) and train it with backpropagation.
- **Use a support vector machine:** If you prefer your objectives convex.
- **Go totally Bayesian:** Worried that you're not accounting for uncertainty? You could take a fully Bayesian approach to classification and marginalize out your uncertainty in a generative or discriminative model.
- **Use a decision tree:** If you think a linear classifier is too simple but don't want to train a neural network, you could try a decision tree.
- **Use KNN:** Have a great way to think about similarities between the sounds? You could try K nearest neighbors and see how that works.