[Fall 2025] Machine Learning Final Competition Guidelines

Audio Classification Task: Echoes of the City

Version: Nov 16, 2025

Figure. Times Square, filled with all kinds of sounds

Notice (Updating)

- Kaggle Link has been fixed.
- Phase One started!

Introduction

In modern cities, places like Times Square are filled with a chaotic mixture of sounds—honking cars, passing sirens, snippets of music, distant construction, and human voices all layered together. Automatically understanding these complex

acoustic scenes is a core challenge in urban computing and audio AI. In this Final Competition, you will build a classifier that predicts the sound category of short urban audio clips, such as car horns, sirens, dog barks, drilling, and street music. Given only the raw waveform, your goal is to design models that are robust to noise, overlapping events, and varying recording conditions. Through this task, you will gain hands-on experience with real-world audio data, explore modern techniques for sound classification, and see how machine learning can help make sense of the rich soundscape of the city.

Task Formulation

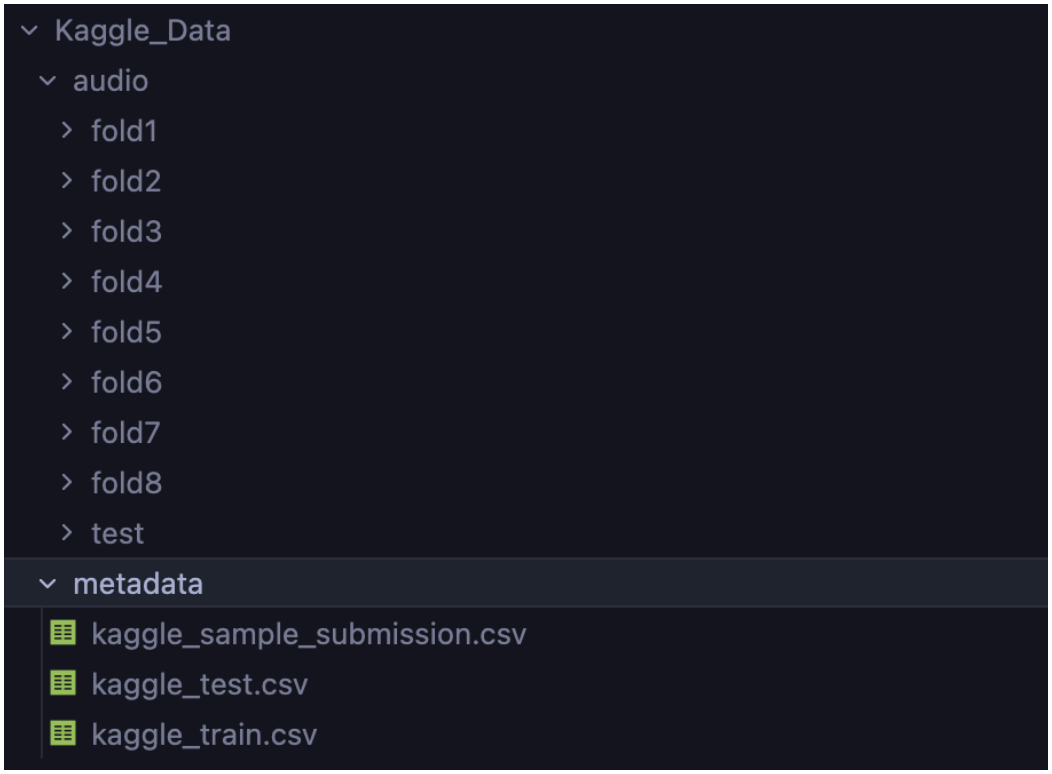
Primary task: Audio Classification

Data Intro: Our competition uses a curated collection of thousands of audio clips of real-world urban sounds from 10 categories, including air conditioners, car horns, children playing, dog barks, drilling, engine idling, gun shots, jackhammers, sirens, and street music. The clips come from real field recordings and are organized into 8 predefined folds to support standardized training and evaluation.

Data Format: You are also provided with a metadata file in CSV format, which contains one row per audio clip. The main fields are:

- slice_file_name
  The file name of the audio clip.
- start, end (No use)
  The start time and end time (in seconds) of the slice within the original recording (raw data).
- salience
  A subjective salience rating of the target sound in the clip:
  1 = foreground, 2 = background.
- fold
  The fold index (1–8) indicating which split this clip belongs to.

File Structure:



`audio/fold1` – **`audio/fold8:`** Contain the labeled training audio clips. Each WAV file is a short urban sound (≤ 4 s). You can decide how to use these folds (e.g., cross-validation *(Recommended)*).

`audio/test:` Contains the unlabeled test audio clips. You will make predictions for these files and submit them to Kaggle.

`metadata/kaggle_train.csv:` Metadata for the training set. Each row corresponds to one training clip.

`metadata/kaggle_test.csv:` Metadata for the test set. Each row corresponds to one test clip and provides the ID / file name that you must use in your submission. No labels are provided for these clips.

`metadata/kaggle_sample_submission.csv:` A sample submission file showing the expected Kaggle format. It contains the test IDs and a placeholder column for your predicted class id.

Evaluation Metrics

We use a weighted sum over two metrics to quantify the model performance:

- Overall Accuracy (80%): Correct rate over all classes and data points.
- Macro-F1 (20%): Measures how well your model performs across all classes.

Submission Guidelines

We host our competition on Kaggle (competition link). Your submission will be in the form of a CSV file, where you include the predicted class labels on the test dataset. The details can also be found in Kaggle->Overview->Evaluation. Please name your competition alias with your **NYU NetID (Your Custom Name, optional)**, for example ys1111(I Love ML) and the account with a wrong format will be removed. You can make at most 10 submissions per day.

Competition Milestones

- Phase One Leaderboard (10% of total points, **due Nov 30**)
    - You only need to submit your Kaggle username (i.e., the name you used on the Leaderboard) to Brightspace Assignment "Final Competition Phase 1". If you get a Kaggle score higher than 30%, you get all the points.
- Phase Two Leaderboard (90% of total points, **due Dec 14**)
    - This part is scored based on the weighted sum over the two metrics mentioned above. You need to submit your model weights, your code, and a PDF report (3-4 pages) to Brightspace. We will release a report template with suggested sections and format. Always make sure the results are reproducible. Your code should be runnable, and your logs should be kept if you submit an IPython notebook.

Other Rules (Very Important, Continuously Updated)

- Discussions are highly encouraged, but please always write your own code.
- Cases that are considered as cheating (which will lose all the points):
    - Labeling any split of the dataset manually in any form.

    - Using pre-trained model weights or distillation in any part of your code.
    - Using GenAI tools (e.g., ChatGPT) to introduce extra information for inference. (Using them for brainstorming or debugging is fine.)
    - The submitted code and model weights cannot reproduce the test set predictions on Kaggle. (Small deviations are not a problem.)
    - Extremely similar code submission from different students.
- We will randomly invite students to go through their solutions during the competition phase, so please make sure you know what your code is doing.
- For models that perform relatively well (or we find two models that have similar performance), we may meet with you during office hours or other times to discuss your code and implementation (and may require you to reproduce your results live or we run your code in person). However, if your actual performance differs from the leaderboard results, or if we discover other violations, we will also consider it cheating.
- We will use advanced tools to do the duplication check, these tools don't rely on text similarity but instead delve into your data flow and architecture.

If any cheating is detected, your Final Competition score will be recorded as 0 and reported to the Office of Student Affairs, and you will bear full responsibility for the consequences. At that point, we will not consider any explanations or appeals.

Tips from TAs

- Please feel free to start from the demo code
- Use cross-validation since the training data have been already formatted in 8 folders.
- Visualize the confusion matrix on the validation set to observe some potential problems of your current model.
- Do not reshuffle the data! If you reshuffle the data (for example, by merging all folds and creating a random train/test split), because many audio files in the same folder come from the same audio segment, related samples will end up in both sets, artificially boosting your scores and failing to reflect true

performance on unseen data—in short, your evaluation will be invalid. However, you can merge several folders to form a new folder, as long as the data originally in one folder remains in the same folder after merging.

- On this specific dataset, handling the noisy and imbalanced data properly might be more helpful than using a stronger model. Meanwhile, a model that is too simple may not possess enough learning capability for this task.