General guidelines:

- This assignment is tentatively due on **Tuesday, 5pm, week 8 (May 20);** as always, the final deadline is **as posted on gradescope**
- Note the slightly unusual deadline: I want to avoid a clash with the HW4 deadline, and I want the leaderboard to close while people are awake
- Also note that there is **no late period** for this assignment: please submit on time!
- **This assignment is worth 20% of the final grade** (noting grade scaling as posted on the course webpage)
- Assignment 1 must be completed individually

Data and baselines

Assignment 1 data and baselines:
https://drive.google.com/drive/folders/1U_Wb7Q1JhrljRiG9dQ6oyCSPvPCbynm9?usp=sharing
(data is also posted here in case drive gets rate-limited:
https://cseweb.ucsd.edu/classes/sp25/cse253-a/data/)

There are two files in the above folder:

student_files.tar.gz: A compressed folder containing:
- Training datasets for each of the three tasks below, *with* labels
- Testing datasets for each of the three tasks, *without* labels

baselines.ipynb: Working baselines for each of the three tasks

Note that testing labels are not provided: rather you have to make your own predictions on the test set and upload them to gradescope.

Also note that there is no stub provided: it's easiest to just edit **baselines.ipynb** directly to form the basis of your solution. You are not required to follow any of the functions outlined in that file if you don't want to.

Tasks

You are required to build classifiers for each of the following three tasks:

Task 1: Composer classification (symbolic, multiclass). For this task you are required to process a midi file and predict which composer wrote the piece of music. This task is evaluated based on **accuracy** (percentage of correct predictions).

Task 2: Next sequence prediction (symbolic, binary). For this task you are given two bars of music (really, two midi files) and you are required to guess whether the second bar would follow the first in a real piece of music (or whether they are unrelated). I.e., predict “True” if the two bars are neighbors and “False” otherwise. This task is evaluated based on **accuracy.**

Task 3: Music tagging (continuous, multilabel, multiclass). Predict a set of tags (e.g. “electronic”, “chill”) associated with an audio file. The input audio files are outputs of a generative model that synthesized music based on the given tags. Note that for this task an example can have multiple tags. This task is evaluated based on the **mean average precision** (mAP).

Baselines for the three tasks are fairly straightforward, mostly just to show you how to process the data and generate valid outputs that can be submitted to gradescope:

Task 1: Logistic regression; features are computed based on the average pitch and duration.

Task 2: No ML is used: this baseline just measures whether the two bars have a similar average pitch.

Task 3: CNN based on MelSpectrograms; this baseline also shows you how to build your own validation set using a sample of the training data.

Submission

Running the baseline code will produce a valid submission that you can submit to gradescope. It will generate three outputs:

predictions1.json, predictions2.json, predictions3.json

These contain the predictions for the three tasks on the test set. Although you are not given the test labels, the evaluation code in the baselines file shows you how your predictions will be compared to the test labels by the autograder.

The autograder will evaluate your test performance and print your results on a leaderboard. Note that there is both a “public” and “private” leaderboard, each of which contains a random half of the test set. The latter will only be visible after the deadline to prevent people from overfitting to the public half.

You are also required to submit two additional files:

assignment1.py: the code you used to train your model. The autograder does not currently run your code, though this should be runnable in the event that there is any doubt about your solution.
writeup.txt: 1-2 sentences describing your solution to each task. I just want this so that I can describe solutions after the assignment finishes.

Grading

The assignment is worth **20 marks.** The grade breakdown is as follows:

- Submit valid writeup.txt and assignment1.py files **(2 marks).** The autograder doesn't currently check these (and will just give out the marks) but could be modified e.g. if people are submitting empty files.
- Each of the three tasks is worth **6 marks** (3 for the public leaderboard and 3 for the private)

The **breakdown of the 3 marks per leaderboard & task is TBD** but planned guide is as follows (same for all three tasks):

- **0.5 marks:** Submit a valid solution (just run the baseline code and submit it!)
- **1 mark:** Submit a solution that is at least 5% better than the baseline (i.e., >= baseline performance * 1.05)
- **1.5-2.5:** Submit a solution that is "significantly" better than the baseline. Tentative thresholds are below (may adjust down or add smaller increments, won't adjust up)
    - Task 1: 0.4 (1.5); 0.5 (2.0); 0.6 (2.25); 0.7 (2.5)
    - Task 2: 0.7 (1.5); 0.8 (2.0); 0.85 (2.25); 0.9 (2.5)
    - Task 3: 0.3 (1.5); 0.35 (2.0); 0.4 (2.5)
- Submit a solution that is among the top 10% of submissions: **3 marks** (possibly computed "smoothly" rather than given to the top 10% in a binary fashion)

(the above three marks are computed for both the public and private leaderboards, and for each of the three tasks, for a total of 18 marks)

Note: baseline performance, task 1/2/3 (public): 0.251256 / 0.623775 / 0.270429

Getting started

We extracted features from midi files in **Homework 1** and built CNN classifiers for audio in **Homework 2:** you can use those (or the solution files, which I'll post) as the basis of your solutions.

Also please watch the **Thursday week 4 lecture** for discussion of this assignment and some tips about getting started!