AI Team – Audio Data Challenge

Description:

Access data in folder named "Data" containing audio files (.wav) from various sources and four target classes. Please note that the sample rate of each .wav file can be varied. Please use Python for the task. Please feel free to use the internet and relevant libraries. All the following code should be in a GitHub repository. Please read this carefully before starting, particularly with the minimal success criterion before improving on it.

I. Build a multi-class classifier to predict common sounds in urban environment

1. Load the audio files, plot, and analyze waveforms and spectrograms for each class.
2. Create training and testing sets (random split or cross-validation techniques).
3. You may use audio signal processing techniques to pre-process the signals.
4. Apply feature engineering techniques to get some form of feature extraction as a model input.
5. Use a linear SVM (Support Vector Machine) and another ML (Machine Learning) technique to build and compare models.
6. Plot a confusion matrix and other validation techniques to analyze model performance.
7. Explain the cause of confusions and errors.

II. Create a CI/CD framework to maintain your repository

1. Commit your GitHub repository.
2. Write at least one unit test for your repo.
3. Create a GitHub action that runs that unit test.

III. Create a pip installable version of your code

Create a pip installable version of your code that accepts X,y does the feature extraction and runs steps 6,7 automatically. As an example, let us say your package is called my_package and you have a function called run_job(X,y). Then the following code should work:

```
# pip install <link your package on github>

From my_package import run_job

# Assume you have X,y in memory, create some random X and y, that is fine too

Run_job(X,y) # should display/ or save to a directory all graphs and results etc
```

Evaluation criteria:

For Part I, we are only interested in a "decent" model, do not invest time in making the model perfect or increasing the metrics to 99%. It is much more important for you to be able to do the following:

1. Explain at least **3 signal processing features** you chose and why. Explain why the features work, at a conceptual level.
2. Explain why you evaluated the model using the validation techniques and metrics you chose.
3. Explain how you would go about improving the model.

For Part II, we are looking for familiarity with GitHub and the ability to write a unit test and run that unit test on GitHub using a GitHub action. The focus here should be in completing this task with two things in mind:

1. If someone contributes a new feature, new model etc. to your pipeline how would you know that your entire pipeline works automatically.
2. You do not need to write more than one unit test, but you need to be able to explain potential next steps to ensure pipeline quality.

For part III, we are looking for the ability to write a pip installable package. Again, the focus is not to create something overly complicated but to ensure the following.

1. Can we pip install your code
2. Can we use a function from within your code? The above example is the most usual use case we could think of. But if you have a function in your code to add two numbers together and we can pip install your repo and use that function that would be a fine result too. Any simple function will do, it would be great if that function did something ML or Signal Processing related but it is not necessary to meet this criterion.

Overall, our focus is on your ability to complete the above tasks before assessing the quality of them.

We will focus on -

1. Code and functions organization.
2. Your thought process about approaching signal processing/machine learning problems

Deliverables:

1. GitHub repo (with GitHub actions etc)
2. Pip installable package
3. 1–2-page report/ presentation for Part I explanations/ analysis/ confusion matrices etc.

Minimal Success Criteria:

There is every chance that you will not have enough time to complete the above in the time frame. This is expected; however, we would advise you to try to have the following as a minimally viable solution, in decreasing order of importance,

1. A simple model that works and has an evaluation metric that makes sense.

2. A report that explains the features you chose (following guidance above), along with the next steps.
3. A GitHub repo, with a unit test in it
4. A GitHub Action that should work (even if it does not work due to implementation errors).
5. A pip installable package that imports a function, this function can be as simple as adding two numbers together, but it should come from your repo.