Homework 1: Common Guidelines

General Pipeline

- **Collect data** from the folders of the original dataset.
- **Retrieve the annotations:** sometimes annotations are implicit in the organization of the folders of the dataset, sometimes are expressed in the name of the file, sometimes are in a separated CSV file.
- If necessary, **apply some preprocessing** to audio files: for example, are they mono or stereo? Are all of them within the same range of values?
- Split your dataset in **training set** and **testing set:** what proportion would you use in this split? Can you add some constraints such that you are sure that training set and testing set are as different as possible?
  For classification, check also if the dataset is balanced, i.e. it has a similar number of samples for each class. If it is not, you can decide either to balance it by sub-sampling or, in the evaluation part, to use metrics suitable for un-balanced datasets. You can also decide to apply cross-validation (k-fold for example) in your project.
- **Extract the features:** think which set of features can be useful for your task, test different parameters for computing them and think how they should be aggregated over time. Depending on the task it might be better to aggregate all the features in time (e.g. taking statistical moments) and estimate a single label/value per audio track **or** predict a label/value for each time window and then aggregate all of them in a single result per audio track.
- If you want, **apply feature selection** methods
- Choose one **classification/regression method:** fit the model on your training set and use the testing set for the evaluation of your method. If you want, repeat the fit with different parameters for your classifier/regressor for fine tuning your model. In the evaluation part, choose which metrics are meaningful for your task (maybe more than one!).

FAQs:

- **How can we deal with large data?**
  If you want to have more “structured“ data you can use **Pandas** library. It allows to use fast, flexible, and expressive data structures designed to make working with “relational” or “labeled” data both easy and intuitive. Here you can find a quick starting guide:
  https://pandas.pydata.org/pandas-docs/stable/getting_started/10min.html

- **How do we extract features**?
  A first option can be to implement from scratch the features you need. Alternatively, the libraries available are
  - **Librosa** https://librosa.github.io
  - **Essentia** https://essentia.upf.edu/documentation.html using the Python bindings
  - **VAMP Plugins** using the Python wrapper https://vamp-plugins.org/vampy.html. VAMP plugins are a set of plugins that extract meaningful information from audio using state-of-the-art algorithms. You can find a list of available plugin at the following link https://vamp-plugins.org/download.html

- **How do we define and train our models?**
  You can find several classifier, regressors, feature selection algorithms (and more) implemented in the **scikit-learn** library https://scikit-learn.org/stable/. In the documentation you can find also a number of tutorials useful for your homework. Here a list of user guides that can be probably of help for the homework:
  - Supervised Learning in general https://scikit-learn.org/stable/supervised_learning.html#supervised-learning
  - Multiclass and multilabel: https://scikit-learn.org/stable/modules/multiclass.html
    "All scikit-learn classifiers are capable of multiclass classification, but the meta-estimators offered by ***`sklearn.multiclass`*** *permit changing the way they handle more than two classes because this may have an effect on classifier performance"*
  - Feature selection https://scikit-learn.org/stable/modules/feature_selection.html#feature-selection
  - Cross validation https://scikit-learn.org/stable/modules/cross_validation.html#cross-validation
  - Fine tuning https://scikit-learn.org/stable/modules/grid_search.html
  - Model evaluation https://scikit-learn.org/stable/modules/model_evaluation.html
  - Pipeline (optional): https://scikit-learn.org/stable/modules/compose.html#pipeline ….and many others!

- **Additional hints:**
  - check **tqdm** to make your loops show a progress meter https://github.com/tqdm/tqdm . This allows you to have an idea, in case of for loops, of how long your computation can take.
  - check **multiprocessing** library https://docs.python.org/2/library/multiprocessing.html if you think that parallelization can speed up your code.
  - if you want to check some literature about your homework, remember that using the proxy service provided by Politecnico di Milano you can access to

a large number of scientific publications https://www.asict.polimi.it/en/network-services/proxy.html

- don’t be scared of listening/plotting/look at the distribution of your data. This can help you in figuring out which are the best design choices for your homeowork

These are just suggestions!

Feel free to use any other Python library or resource for your implementation.