OpenML in Python

OpenML is an online collaboration platform for machine learning:

- Find or share interesting, well-documented datasets
- Define research / modelling goals (tasks)
- Explore large amounts of machine learning algorithms, with APIs in Java, R, Python
- Log and share reproducible experiments, models, results
- Works seamlessly with scikit-learn and other libraries
- Large scale benchmarking, compare to state of the art

Installation

- `pip install openml`

```
In [0]: !pip install openml
```

Exercise

- Find datasets with more than 10000 examples
- Find a dataset called 'eeg_eye_state'
- Find all datasets with more than 50 classes

Download datasets

Download the `eeg_eye_state` dataset. This is done based on the dataset ID ('did').

Get the actual data.
Returned as numpy array, with meta-info (e.g. target feature, feature names,...)

Exercise

- Explore the data visually

Task

The function `openml.evaluation.list_evaluations(...)` returns a dictionary of evaluation records. It has several filtering functions, to keep the resulting set small (keep in mind that OpenML has almost 10 million runs, and more than a billion evaluation records). The function is documented in the API docs (https://openml.github.io/openml-python/master/generated/openml.evaluations.list_evaluations.html#openml.evaluations.list_evaluations). It returns a dict mapping from `run_id` to OpenMLEvaluation (https://openml.github.io/openml-python/master/generated/openml.OpenMLEvaluation.html#openml.OpenMLEvaluation). Examples of filters are `task`, `flow` and `function`. Note that one of these is mandatory.

- Obtain a subset of 100 predictive accuracy (`predictive_accuracy`) results on the letter dataset (task id = 6).
- Obtain a subset of 100 predictive accuracy (`predictive_accuracy`) results per task in the OpenML 100 and plot these

Dataset Upload

There are various ways to upload a dataset. The most convenient ways are documented in this example (https://github.com/openml/openml-python/blob/master/examples/create_upload_tutorial.py). Most conveniently, this can be done using a pandas dataframe (https://github.com/openml/openml-python/blob/a0ef724fec6ab31f6381d3ac2a84827ab535170d/examples/create_upload_tutorial.py#L206). Additionally, we need to create a OpenMLDataset (https://openml.github.io/openml-python/master/generated/openml.OpenMLDataset.html#openml.OpenMLDataset) object, containing information about the dataset. Most notably, the arguments `name`, `default_target_attribute`, `attributes` and `data` need to be set.

- Find your favorite dataset (on your laptop), load it as pandas dataframe and upload it to OpenML.
- Common problem: Server returns error 131. This means that the description file was not complete. The XSD (https://github.com/openml/OpenML/blob/master/openml_OS/views/pages/api_new/v1/xsd/openml.data.upload.xsd) for uploading the dataset hints what fields are mandatory.

- If you did not bring you own dataset, find an interesting public dataset on:
  - Kaggle open datasets: https://www.kaggle.com/datasets (https://www.kaggle.com/datasets)
  - Data.world: https://data.world/ (https://data.world/)
  - Wolfram Alpha: https://datarepository.wolframcloud.com/ (https://datarepository.wolframcloud.com/)
  - More: https://git.io/vdTXm (https://git.io/vdTXm)
- Note that:
  - The dataset should not already be on OpenML.
  - Tabular (e.g. CSV) data, representing a classification or regression problem
  - No text/image data, unless already featurized