Machine Learning Project
WS24/25

Machine Learning Group
Fakulty IV, Technische Universität Berlin
Prof. Dr. Klaus-Robert Müller
Email: klaus-robert.mueller@tu-berlin.de

Machine Learning Project

Aim and scope

This module is designed with the purpose of equipping students with a comprehensive grasp of the practical application of Machine Learning techniques in both academic and industrial scenarios. Unlike other modules that predominantly delve into methodologies, this module offers a holistic perspective on the complete life-cycle of a Machine Learning project.

Students are not required to invent or re-implement algorithms from scratch. Instead, they should utilize the `PyTorch` library and the `scikit-learn` toolkit. Nonetheless, programming will play a significant role, encompassing tasks such as data logistics, feature extraction, model analysis and assessment. Please note that all code must be written in Python.

Project Machine Learning vs. Lab Course Machine Learning

There is a clear difference between the Lab Course and the Project:

- In the Lab Course, the tasks are very well-defined, providing precise instructions on programming, plotting, and discussion topics.
- The Project represents the more advanced class, where instructions are intentionally more generalized. Here, you have the autonomy to determine your approach to tasks, identify the key aspects of importance, and craft meaningful figures in your own creative way.

Prerequisites

There are no formal entry requirements. However, we strongly recommend

- the module Machine Learning 1,
- Python Programming for Machine Learning and
- ideally, the Lab Course Machine Learning.

Students who have not attended these lectures should ensure they have the following skills:

Basic theory: Students should be fluent in probability theory, linear algebra, and understand how and why mainstream learning algorithms work.

Some practical ML experience: Having prior experience with the practical application of ML algorithms is essential. Students must have the ability to effectively choose hyper-parameters and evaluate the performance of a trained predictor.

Python programming: All code submissions are required to be written in Python. Students are expected to demonstrate proficiency in Python programming, utilizing essential libraries such as `NumPy` and `SciPy`.

Dates and structure

The project is divided into the following three milestones:

1. Data processing and prototyping.
2. Calibration and assessment of a learning method (reproducing existing results).
3. Calibration of final model and in–depth evaluation.

For each milestone, each group must hand in

1. a report (not longer than ten pages) and

2. code in a standardized format.

We will schedule a meeting approximately two weeks before each deadline to discuss any challenges and track your progress. Following the completion of each milestone, we will organize a seminar where students will have the opportunity to present their solutions in brief, informal talks lasting less than 10 minutes.

Registration and examinations

The course enrollment is limited to 36 participants to ensure an optimal learning experience. Placement in the course is primarily determined on a first-come, first-served basis. Attending the initial meeting is mandatory. In case you are unable to attend, please notify the course organizers via email. Failure to do so may result in the assumption of your withdrawal from the course, prompting us to offer your spot to another interested individual.

The grades are entirely determined by the coursework submitted for each milestone[1]. Each milestone will be assessed individually on a scale of zero to ten points, with the final grade determined by the cumulative sum of these points.

Programming hints and guidelines

You will need the packages `NumPy` and `SciPy` for numerical linear algebra. If you are new to Python, but experienced with Matlab, have a look at

`http://www.scipy.org/NumPy_for_Matlab_Users`.

Additional information and resources can be found under

`https://wiki.ml.tu-berlin.de/wiki/Main/PythonKurs`.

We highly recommend you to install the package `matplotlib`, which is a plotting library for `NumPy`. Additionally, it will be very helpful to install `IPython`, which is an improved Python shell, and especially useful for interactive data visualization. Once installed, the command

```
ipython --pylab
```

will start a Python environment, in which all numerical and plotting libraries are already imported.

In the Python Programming for Machine Learning course, `Jupyter` notebooks are used as a comfortable development environment:

```
jupyter notebook
```

In the code you submit, it is essential to document each and every function with a descriptive doc-string. Make sure to follow the specified file names, function signatures, and expected behaviors *precisely*.

Contact

Mina Jamshidi-Idaji
`mina.jamshidi.idaji@tu-berlin.de`

[1] *Prüfungsäquivalente Studienleistungen* in TU-parlance.