Machine Learning Tek5 2025-2026 project

NICOLAS LE HIR
nicolas.lehir@epitech.eu

TABLE DES MATIÈRES

INTRODUCTION

0.1 Organization

0.1.1 Repository

The project consists in several independent exercises. **Before the defense, please send me the information about your group members and the repo url.** You can push your work to the epitech repo dedicated to the project, inside the course team. Please do not send a fork of the course repo, with some added files. Instead, make a repo containing only your project files.

0.1.2 File format

The format of the files is mandatory. One notebook per exercise, named like the exercise name :

- 1_1_1.ipynb
- 1_1_2.ipynb
- 1_1_3.ipynb
- ...
- 2_3.ipynb
- 2_4.ipynb

Your notebook may import functions from other modules (python files) if that makes the code easier to read. To explain your work, you can write markdown cells inside the notebooks.

Please push your notebook with all the cell outputs and all the generated images. You do not have to push the datasets.

For exercices 1.2.1, 1.2.2, 1.3.1, 1.3.2, the path to the data in you notebook must be relative and **exactly match** the path in the repo. For instance **data/classification/X_test.npy**.

0.1.3 Group registration

For questions related to group inscriptions to the project, please contact the administration directly via a ticket. I have no possibility to act on the group registrations.

0.1.4 Guidelines

Note that during the course, we will discuss your projects together and I will be available for questions during the complete duration of the course. As a result, I should be already familiar with your work before the defense.

All processing should be made with python3.

During the defense, what is expected is that you explain your process to arrive to the results, and **conclusions** are expected, instead of low level explanations on elementary functions. Please write some sumary of your conclusions at the **beginning** of the notebook of each exercise (this actually facilitates reading a lot). For instance :

- (For exercice 1.2.1) : "we managed to obtain a test accuracry of 0.88 with the (input model A name) model, whereas with (input model B name) we could not obtain more than 0.75."

You may use the notebooks as your slides (to confirm).

— Example of explanations and discussions that are expected in th defense : discussion over the choice of models (model selection), hyperparameters, details on the optimization (or model training) process, and conclusions on which model(s) worked best, or over the choice of preprocessing methods on the datasets.

— Example of explanations **not** to include in the defense : presentation of elementary python function, or library functions from numpy, matplotlib, or the libraries themselves. You should **not** present these.

You can however discuss specific functions or methods that are important for the optimization process, for instance in pytorch or tensorflow algorithms and optimizers (but again, you do not need to explain what pytorch is).

Important guidelines :

— when you mention the score of an estimator, **always** explicitely mention whether it is a train, test, validation, or cross-validation score (see the "scoring" chapter in the class)

— never forget to label the axes of plots

— never forget to mention the dimensions (unités) of the quantities

0.2 Defense

There will be a final 40 minutes time slot to to present your work. However, only certain exercices are discussed during the defense.

— 1.2.3 (5 min)

— 1.3.1 (1 min)

— 1.3.2 (1 min)

— 2.1 (5 min)

— 2.2 (5 min)

— 2.3 (5 min)

— 2.4 (5 min)

For these exercises, the **maximum** duration during the defense is indicated in its title, in order to have the time to interact and give you some feedback. Please practice with a timer and use a timer during the defense. You can also go faster if it is sufficient to explain your work, no problem with that.

For the other ones, I read your notebooks only.

0.3 Datasets for part 2

In 2, you might choose datasets that you find interesting. The objective is that we validate your datasets together during day 2 and / or day 3. During the couse, we have time slots that are dedicated to discussing your advancement on the project and think together. Some resources to find datasets : Link 1, Link 2, Link 4 (but feel free to use different ones).

1 PART I

1.1 Elementary tools for machine learning

1.1.1 Artificial dataset generation (1 min)

The goal of this exercise is to work with statistical notions such as mean, standard deviation, and correlation.

Generate a numerical dataset with 300 datapoints (i.e. lines) and at least 6 columns and save it to a csv file named **artificial_dataset.csv**. This dataset must represent physical quantities of your choice, with units. Ths statistical relationships between the columns must make sense.

The columns must satisfy the following requirements :

— they must all have a different mean

— they must all have a different standard deviation (English for "écart type")

— at least one column should contain integers.

— at least one column should contain floats.

— some columns must be positively correlated (a pair of column must have a correlation > 0.2).

— some columns must be negatively correlated (a pair of column must have a correlation < -0.4).

— some columns must have a correlation close to 0.

1.1.2 Definition of metrics (1 min)

Choose a dataset, from a source you like and define two different **metrics** on this dataset, which means define two methods to compute **dissimilarities** between the samples, by taking into account all their features (columns of the dataset). The objective is that :

— the two samples that are the closest in the dataset are different according to metric 1 and to metric 2.

— the two samples that are the most far appart in the dataset are different according to metric 1 and to metric 2.

The units of measurement (unités, like Kg, cm) should be taken into account while computing the metrics. Compute explicitely the most similar and most dissimilar samples for each metric and discuss the result by commenting on the balance of the features in each metric.

1.1.3 Law of large numbers (1 min)

The goal of this exercise is to manipulate a data distribution and to get familiar with the law of large numbers in an informal way.

1) Propose a 2-dimensional random variable $Z = (X, Y)$, with X and Y being two real ($\in \mathbb{R}$), discrete or continuous random variables. These two variables must represent meaningful quantities of your choice, and they must have units. Compute the expected value of Z, that must be finite.

2) Sample a number n (of your choice) of points from the law of Z and plot them in a 2 dimensional figure.

3) For increasing values n, compute the empirical average of the first n samples and verify that it converges to the expected value, by plotting the euclidean distance between the empirical average and the expected value as a function of n.

Remarks :

— Please pay attention to the fact that the expected value and the empirical average are **different objects**.

— Note that both the empirical average and the expected value are vectors with 2 entries, as our variable is 2-dimensional.

1.2 Supervised learning

1.2.1 Classification (1 min)

We would like to predict the winner of a Basketball game, as a function of the data gathered at half-time.

The dataset is stored in **project/data/classification/** :

— The train and test inputs representing the features are stored in **X_train.npy** and **X_test.npy** respectively.

— If the home team wins, the label is 1, −1 otherwise. The train and test labels are stored in **y_train.npy** and **y_test.npy** respectively.

Your objective is to obtain a mean accuracy superior than 0.84 on the test set. https://scikit-learn.org/stable/modules/model_evaluation.html#accuracy-score

Remark : Pay attention to the fact that **the test must not be used for training. The test set should be used only once, even for scoring**. If you compute the score several times, with different models on the test set, it means that you use it more than once, even if you do not call a scikit model.train() method on the test set (see the class for more explanations)! Note that this strict unique usage of the test set is not always common practice in companies, but try to apply it for this exercise.

You are free to choose the classification methods, but you must compare at least 2 models. You can do more than 2 but this is not mandatory for this exercise. Discuss the choice of the optimization procedures, solvers, hyperparameters, cross-validation, etc. It is sufficient that 1 of your models reaches the objective score.

Several methods might work, including some methods that we have not explicitely studied in the class, do not hesitate to try them.

Indication : a solution, with the correct hyperparameters, exists in scikit among the following scikit classes :

— linear_model.LogisticRegression

— svm.SVC

— neighbors.KNeighborsClassifier

— neural_network.MLPClassifier

Please note that there is no length constrain on your solution notebook, it may be short or long.

1.2.2 Regression (1 min)

We would like to predict the amount of electricity produced by a windfarm, as a function of the information gathered in a number of physical sensors (e.g. speed of the wind, temperature, ...).

The dataset is stored in **project/data/regression/**, similarly to 1.2.1 and the instructions are the same (compare at least two models and discuss the hyperparameter

optimization procedure). Your objective is to obtain a R2 score superior to 0.85 on the test set, for at least 1 of your models.

The same remark about the test set, presented in exercice 1.2.1 also applies here.

https://scikit-learn.org/stable/modules/generated/sklearn.metrics.r2_score.html

Several methods might work, including some methods that we have not explicitely studied in the class. Do not hesitate to try such methods.

Indication : a solution, with the correct hyperparameters, exists in scikit among the following scikit classes :

— linear_model.Ridge

— linear_model.Lasso

— neural_network.MLPRegressor.

— svm.SVR, ensemble.AdaBoostRegressor.

1.2.3 *Deep learning on MNIST (5 min)*

As we have seen during the course, we can reach more than 0.99 of test accuracy in a couple of minutes on a laptop on the MNIST dataset (https://en.wikipedia.org/wiki/MNIST_database). However, in many machine learning applications, especially in embedded systems, there is a need to optimize the learning time and also the inference time.

Look for the **fastest possible way** to reach a 0.97 test accuracy, on a given hardware of your choice. The hardware that you use may be the laptop of one of your group members (you can also compare two laptops). You need to briefly present the hardware used (type of processor, frequency, number of cores, etc)

In that respect, your objective is to work on the **acceleration** of **learning (model training)** and / or of the **inference time**, using any method that seems relevant. Please note that your objective is not to simply reach the target test accuracy.

It is required that you explore at least one method taken from a book or from a scientific article. You can find the book or the article yourself, or use one of the references presented during the course. For instance, part II of the Deep learning book (https://www.deeplearningbook.org/) contains many interesting discussions, but other sources are accepted as well. Any research is welcome, including research on libraries that are not as famous as pytorch and tensforflow, like jax.

Here are some suggestions :

— you can start from the architecture that we use in this example : https://github.com/nlehir/ML_tek5/tree/main/code/day_3/2_neural_networks/mnist and then try other methods to accelerate learning or inference time.

— try to **isolate** paramaters by monitoring the change in speed after changing only one aspect of the pipeline, leaving the rest similar.

You are encouraged to present and analyse results of code profiling in order to find speed bottlenecks. Pay attention to the fact that profiling GPUs or pytorch code might have specificities.

1.3 Unsupervised learning

1.3.1 *Clustering (1 min)*

A company has gathered data about its customers and would like to identify similar clients, in order to propose relevant products to new clients, based on their features. This can be represented as a clustering problem. The data are stored in **project/data/clustering/**. They are 4 dimensional.

Pick :

— **two** clustering methods

— **two** heuristics to choose a relevant number of clusters,

and perform different clusterings of this dataset (overall, you have $2 \times 2 = 4$ methods). You must use a different metric for each clustering method. You could for instance use the standard euclidean metric for one method, and a different metric for the other method, for instance based on a rescaling of the dimensions of the data (hence, you could transform the data first, and apply a known metric on the transformed data.)

Compare and discuss the difference between the results of the different methods you tried. Discuss whether one mehod (combination of the clustering method and of heuristic) seems to give more interesting or clearer results than the others.

Super important : write your final recommendation for the number of clusters to use at the **beggining** of the notebook.

You may use libraries such as scikit-learn in order to implement the methods.

1.3.2 Dimensionality reduction and application to a prediction (1 min)

A meteorological station has gathered 800 data samples in dimension 6, thanks to 6 sensors. The operators of the station would like to predict the risk of a tempest the next day, but first, they need to reduce the dimensionality of the data, in order to apply a supervised learning algorithm on the reduced data.

The data are stored in the **project/data/dimensionality_reduction/** folder :

— **data.npy** contains the raw data

— **labels.npy** contains the results for each sample : 1 if there is a tempest, 0 otherwise.

Perform dimensionality reductions of the data, to a dimension of 2 and 3 and plot these reductions onto scatter plots in dimension 2 and 3 as well, coloring the projected samples according to the label of the original sample. You must try at least 2 dimensionality reduction methods. Objective : 1 of the methods, in dimension 2 or 3 must separate the two classes and thus allow to predict the label based on the projected components only.

Super important : write your final conclusion on the method that works best at the **beginning** of the notebook.

You may use libraries such as scikit-learn in order to implement your dimensionality reduction methods, that you are free to choose (linear or non linear). One of the methods that we have seen during the class works well, with a well chosen output dimension.

2 PART II

In this part, you have more freedom to choose the dataset and the problem to solve. However, the following datasets are **not** accepted :

— red wine quality :
https://www.kaggle.com/datasets/uciml/red-wine-quality-cortez-et-al-2009

— wine dataset for clustering :
https://www.kaggle.com/datasets/harrywang/wine-dataset-for-clustering/data

— mall customers :
https://www.kaggle.com/datasets/vjchoudhary7/customer-segmentation-tutorial-in-python

— country data :
https://www.kaggle.com/datasets/rohan0301/unsupervised-learning-on-country-data

— iris dataset.

— titanic dataset.

2.1 Classification (5 min)

Solve a classification problem on a dataset of your choice.

Mandatory (very important) : before any processing, you must fulfil these two steps :

— **mandatory :** present the basic information about the problem, in the following format :
  — Number of features
  — Number of samples
  — Name of the target variable and physical meaning, if not obvious

— present the dataset shortly in your own words (please do not copy a description from another resource) and link to the url where you downloaded it from. For instance, it is very important to **present the features of this dataset that are not obvious** for someone (me) that is not necessary familiar with the dataset.

— **explain very explicitely what problem you are trying to solve, and in particular what quantity you are trying to predict, as a function of which features**. If the quantity to predict is one of the columns of the dataset, name it explicitely. If relevant, discuss why solving this problem would be interesting or have a value for an industry.

You are encouraged to compare several estimators / optimization procedures, from different points of view (scoring, training time, etc). **General guideline :** as this course is dedicated to discovering and exploring some of the many principles of machine learning, rather than being a production-oriented course, you are encouraged to explore original and personal approaches. It is not a huge deal is the final scores are not outstanding, as long as you took the chance to explore a custom approach and learned new methods.

Suggestion of steps :

— provide general analysis of the dataset, that studies its statistical properties, outliers, correlation matrices, or any other interesting analysis. You may produce visualizations. If you present statistical analysis, discuss whether it is consistent with intuition or not (instead of for instance just showing a correlation matrix).

— if relevant or necessary, preprocess the data, and to justify this preprocessing. You could compare the estimator(s) obtained with and without preprocessing.

— discuss the relevant optimization details : cross validation, hyperparameters, etc

Again, explicit **conclusions** are very important :

— provide an **evaluation** or multiple evaluations of the obtained estimator(s), thanks to scorings of your choice.

— discuss the results obtained. Have we solved the problem with this model ? Could the model be actually used ? What precision can be expected with the model on unseen data ?

2.2 Regression (5 min)

The instructions are identical to 2.1, but this time a regression must be made.

2.3 Clustering (5 min)

The instructions are identical to 2.1, but this time a clustering must be made. Discuss the important algorithmic details and research : for instance, the **metric** and the algorithm(s) that were used. Again, it is very important to discuss the results obtained, and to at least interpret the clusters obtained. A numerical **evaluation** of the model is also expected. Note that several scores exist for clustering (`https://scikit-learn.org/stable/api/sklearn.metrics.html`). Use a scoring, or several scoring, that are meaningful with respect to the considered problem.

2.4 Density estimation OR dimensionality reduction OR outlier detection (5 min)

The instructions are identical to 2.3, but this time one of the following processings must be made :

— density estimation

— dimensionality reduction

— outlier detection

Students are sometimes confused about the justification of a dimensionality reduction, which is totally understandable. In this project, what you could do is to study the impact on the performance of a supervised learning problem, when the features are compressed with a dimensionality reduction. The supervised learning score can be monitored as a function of the extent of the reduction. Please note that all other usual applications of dimensionality reduction are also allowed.