ML II unsupervised learning, agents : 2024 - 2025 project

NICOLAS LE HIR
nicolaslehir@gmail.com

TABLE DES MATIÈRES

INTRODUCTION

You can push your work to the epitech repo dedicated to the project, inside the course team. Please do not send me a fork of the course repo, with some added files. Instead, make a repo containing only your project files.

The preferred format is using one notebook per exercise, with markdown comments explaining you approach. If you use python scripts, you may also write a pdf report. Short docstring at the top of files and functions will be appreciated, if relevant. There is no length constraint on the report, you do not need to write more than necessary. The goal of writing a report is to help me give you useful feedback.

General, important guidelines :
— never forget to label the axes of plots
— never forget to mention the dimensions (unités) of the quantities

The 5 parts of the project are independent.

1 PART 1 : DATA DISTRIBUTION AND THE LAW OF LARGE NUMBERS

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 should represent a quantities of your choice (e.g. the age of the individuals in a population,

the color of the eyes of these individuals, ...). 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.
- You may use simple laws. You could for instance start with a very simple joint distribution, make everything work, and then explore more complex distributions if you wish.

2 PART 2 : METEOROLOGICAL DATA : DIMENSIONALITY REDUCTION AND VISUALIZATION

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 **exercise_2** folder :

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

Perform a dimensionality reduction 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.

Which dimension, between 2 and 3, seems to allow to predict the label based on the projected components only ?

You may use libraries such as scikit-learn in order to implement your dimensionality reduction method, 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. You are encouraged to try at least one other dimensionality reduction method, and if the results is not as good as the previous method, to present them shorty in your report as well.

3 PART 3 : COMPANY CLUSTERING CUSTOMERS

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

exercise_3/data.npy. 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.

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

4 PART 4 : EXPLOITATION/EXPLORATION COMPROMISE

4.1 Setting

We consider a one dimensional world, with 8 possible positions, as defined in the folder **project/exercise_4**. An agent lives in this world, and can perform one of 3 actions at each time step : stay at its position, move right or move left.

In this folder, you can find 3 files :

— **simulation.py** is the main file that you can run to evaluate a policy.

— **agent.py** defines the Agent class. This simple agent only has two attributes.

  — **position** : its position

  — **known_rewards** : represents the knowledge of the agent about the rewards in the worlds (see below)

— **default_policy.py** implements a default policy that consists in always going left.

Some rewards are placed in this world randomly, and are randomly updated perdiodically, at a fixed frequency. This means that a good agent should update its policy periodically as well and adapt to the new rewards. The agent knows about a reward in the world if its position has been on the same position as the reward, but each time the rewards are updated, the agents forgets all this knowledge, as implemented line 46 in **simulation.py**.

simulation.py computes the statistical amount of reward obtained by the agent and plots the evolution of this quantity in **images/**. As you can see in the **images/** folder, the average accumulated reward with the default policy is around 16, with a little bit of variance.

4.2 Objective

Write a different, **stochastic** policy in a separate file named **<group_name>_policy.py** that achieves a better performance than the default policy. **<group_name>_** should

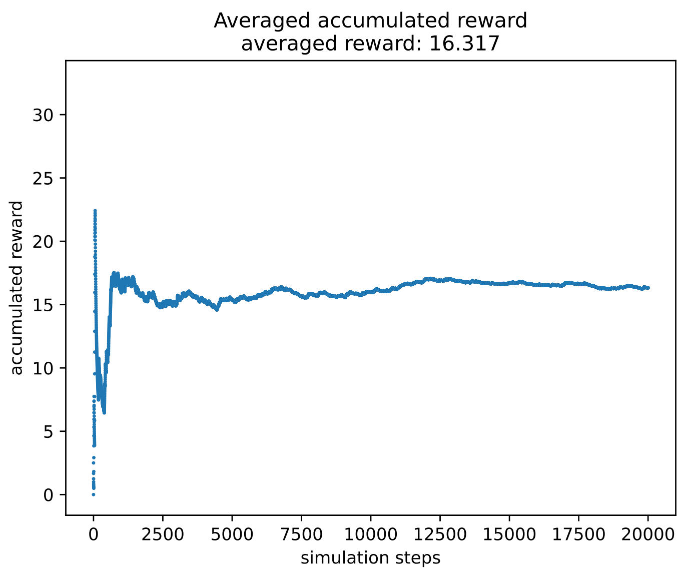

FIGURE 1 – Convergence of the average reward obtained by the agent with the default policy.

be the name of one of the students of your group, or any name that identifies your group.

You will need to

— import you policy in **simulation.py**

— replace line 51 by a line that calls your policy instead of the default policy.

Your objective is to obtain a final average reward of at least 20.

5 PART 5 : APPLICATION OF UNSUPERVISED LEARNING

Pick a dataset and perform an unsupervised learning on it. Your dataset has to be different from any dataset seen during the course.

Important : please try to justify your processing. Why is it interesting to do a clustering / dimensionality reduction / density estimation on this dataset ? What problem are we trying to solve ?

Suggestion of steps :

— 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.

— provide general analysis of the dataset, that studies its statistical properties, outliers, correlation matrices, or any other interesting analysis.

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

— discuss the relevant optimization details

— (mandatory) provide an **evaluation** or multiple evaluations of the results, thanks to scorings of your choice.

  — for a clustering, it can be an inertia, a normalized cut...

  — for a dimensionality reduction, the explained variance

  — for a density estimation, the kullbach leibler divergence between the dataset and a dataset sampled from the estimated distribution

  — but you are encouraged to use other evaluations if they are more relevant for your processing.

- Feel free to add useful visualizations for each step of your processing.
- discuss the results obtained. Have we solved a problem with this processing ?

Some resources to find datasets (but you probably know other good resources already) : Link 1, Link 2, Link 4. If necessary, you can tweak a dataset in order to artificially make it possible to apply techniques that you like, or downsample it. This is not a production project and you are encouraged to experiment.

6 THIRD-PARTY LIBRARIES

You may use libraries.

7 ORGANISATION

Number of students per group : 3.

Deadline for submitting the project :
- 1st session (January 2, 3rd 2025) : February 2nd 2025.
- 2nd session (january 30, 31st 2025) : March 2nd 2025.

The project should be shared through a github repo. **Please send me the repo url by email**.

Each exercise should be in its own folder.

If you used third-party libraries, please include a **requirements.txt** file in order to facilitate installations for my tests, but please specify whether or not you did use libraries that were not used during the course. **If you used only libraries that were used during the course, you do not need to add a requirements.txt file.**

`https://pip.pypa.io/en/stable/user_guide/#requirements-files`

You can reach me be email if you have questions.