Knight problem

Kiwifarm Srl

February 2024

1 Introduction

Skills related to the entire lifecycle of a machine learning project and, in general, to software engineering are important for both Data Scientists and Machine Learning Engineers.

However, both roles, in their daily work, share the need to model problems and respond quickly with various possible solutions: this is the reason why the exercise, in the first part, does not involve machine learning.

2 Main topics and their sub topics

As previously stated, some degree of competence in the following topics is required:

- Python
- Machine Learning / Deep Learning (at least **one** of the following):
  - object detection;
  - object tracking;
  - time series anomaly detection;
  - tabular data classification;
  - reinforcement learning;
  - transformers.
- Systems (plus):

  - AWS;
  - Docker / Docker Compose;
  - distributed architectures.

3 Part 1

A knight is in a known cell on an empty chessboard and needs to reach another known cell.

Write a Python program that can take as input the two positions on the chessboard and returns as output:

- the set of all minimum-length sequences to move the piece from the initial cell to the final cell;
- a graphviz/dot file of the previous "all shortest path" graph.

Your code can use external libraries or you can avoid them as you wish. The case do not require machine learning specific algorithms or approaches.

Please provide also:

- a `requirements.txt`, if needed;
- a `Dockerfile` for the full environment, if you know how to create it;
- a short `README.md`.

In any case, please write a plain Python script **avoiding** Jupyter Notebooks.

3.1 Bonus

During the previous round, top candidates provided a working, well documented, easily testable, `Dockerfile` or even `docker-compose` ready solution (with mounted volumes to quickly read the produced files) with one or more of the following not requested extra features:

- algebraic (chess) notation;
- classes;
- code documentation;
- command line parameters;

- config file (preferred: `JSONC` format);
- extra files (such as `pdf` or `jpg`), not just the graphviz `dot` file, of the oriented graph over a properly oriented chessboard, or with extra dot property for a better reading of the graph itself;
- `docker compose`;
- error/exception handling;
- linting;
- logging.

Feel free to pick items from the previous list or to add your own. 🙂

4 Part 2

Write a short markdown file named `NOTES.md` which should contain a description in English on how you would extend you previous script, showing some knowledge of the previously listed Section 2.

You don't have to write something about every subject: you could choose a subset of them or, in general, try to be creative.

The more topics you can link together or the more creative you are in describing possible lines of research and development, the better. 🙂

When you feel you have achieved a good result in the first and second parts of the exercise, I would ask you to please send all response files via email in a zip format, preferably within **five** working days.

Thank you for your time.