Departamento de Informática
Universidad Carlos III de Madrid

Bachelor in Data Science and Engineering

Machine Learning I
Course 2020-2021

Tutorial 4: Introduction to Reinforcement Learning

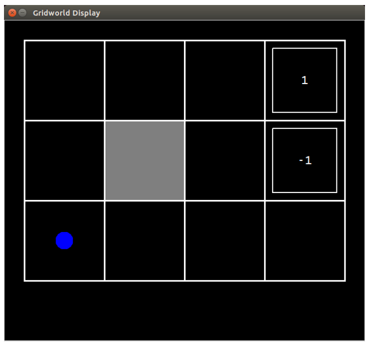
March 27, 2023

- The aim of this tutorial is to get used to the software where you will implement *Q-learning* in the *GridWorld* domain.
- The tutorial can be done in Linux, Windows or Mac.
- It is important to solve the exercises in order.

1 Exercise 1

1. Download the code from Aula Global.
2. Unzip the previous file
3. Open a terminal and enter to the reinforcement tutorial directory.
4. To start we will run GridWorld in manual control mode, which uses the arrow keys.

   python gridworld.py -m -n 0
5. The aim is to reach the cell labeled with 1 as soon as possible, avoiding the -1 cell.



Questions

1. How many cells/states appear on the dashboard? How many actions can the agent execute? If you wanted to solve the game through reinforcement learning, how would you do it?

2. Open the *qlearningAgents.py* file and look for the *QLearningAgent* class. Describe its methods.
3. Now execute the agent with:

   python gridworld.py -a q -k 100 -n 0
4. What information is displayed in the maze? What appears by terminal when movements are made in the maze?
5. What kind of movement does the previous agent make?
6. Draw the MDP considering a deterministic environment.
7. Can several optimal policies be drawn from it? Describe all the optimal policies for this problem.
8. Code the *update* method from *QLearningAgent* using the update functions of *Q-Learning*.
9. Set the value of *epsilon* to 0.05 in the init method of *QLearningAgent*. Run the code again with:

   python gridworld.py -a q -k 100 -n 0

   What happens?
10. After the execution, open the file *qtable.txt*. What does it contain?

2 Exercise 2

Now we are generating an stochastic MDP:

1. Run and play a couple of games with the manual agent:

   python gridworld.py -m -n 0.3

   What is going on? Do you think the *QLearningAgent* will be able to learn in this new scenario?
2. Reset the values of table Q of the file *qtable.txt*. To do this, execute from the terminal:

   cp qtable.ini.txt qtable.txt
3. Run the *QLearningAgent*:

   python gridworld.py -a q -k 100 -n 0.3
4. After a few episodes, is the optimal policy generated? And if it is, does it take longer or shorter than in the deterministic case?

3 Files to submit

All the lab assignments **must** be done in groups of 2 people. You must submit a .zip file containing the required material through Aula Global before the following deadline: **Thursday, April 10th at 8:00**. The name of the zip file must contain the last 6 digits of both student's NIA, `i.e., tutorial4-123456-234567.zip`

The zip file must contain the following files:

1. A **PDF** document with:
   - Cover page with the names and NIAs of both students.
   - Answers to all the questions.
   - Description of the implemented *update* function.
   - Conclusions.
2. The files generated/modified (Q tables) during this tutorial.

Please, **be very careful and respect the submission rules**.

APPENDIX: Code parameters

In order to see all the available options, the following command must be entered:

```
python gridworld.py --help
```

You can change the following parameters:

- **-d discount** Discount parameter. 0.9 by default.
- **-n noise** To make the actions non-deterministic. 0.2 by default.
- **-k episodes** Number of learning episodes. 1 by default.
- **-g maze** Maze used. *BookGrid* by default. You can choose from *BookGrid, BridgeGrid, CliffGrid, MazeGrid* and *getAAGrid*.
- **-a agent** Type of agent. *BookGrid* by default. You can choose from *random*, *value* y *q*.
- **-m** Manual mode.