CONCORDIA UNIVERSITY

Department of Mathematics & Statistics

STAT 497 (MACF 491, MAST 679, MAST 881) Section H

Reinforcement Learning

Project

Frédéric Godin, Ph.D., FSA, ACIA

January 2019

1 Instructions

- The project is done in teams of **three to four** people. In some specific cases, special permissions can be granted by the instructor to form smaller teams. If this is your case, please write to the instructor (**frederic.godin@concordia.ca**) to get his approval. You can have mixed teams containing both undergraduate and graduate students.

- Plagiarism is formally prohibited. Please refer to Concordia's policy related to plagiarism if you are not sure you understand clearly what plagiarism is.

- The deadline to hand in your project is **Sunday April 14, 2019 at 11h59 pm**.

- The topic you choose to study and the composition of your team should be approved by the instructor (through email) before **Sunday February 17, 2019**. See Section 2.1 for more details.

- A zip file **".zip"** must be handed in by email to the instructor before the deadline. The instructor's email is **frederic.godin@concordia.ca**. This file should contain

    - A **pdf** report summarizing the work that was performed and the content of the analyses that were done. See Section 2.3 for more details. This file can be prepared among others with Microsoft Word or LaTeX. This pdf file does not need to contain the R code developed during the project.

    - The integral *R* code needed to run all experiments which lead to results of your analyses. All numerical experiments must be done with the *R* software. If the R code comprises many files, you should have a file called **"main.R"** which calls all of the other scripts. In this case, you should also present a small document called **codestructure** (could be for instance a Word, pdf, Visio, or other file type) documenting the structure of your code (i.e. stating how all files are linked together).

- Although this is not necessary, additional documents can be attached to the zip file (e.g. some data files). In this case, include a folder **AdditionalDocuments** in your zip file which contains the additional documents. If other documents are attached, make sure the size of the file is reasonable (e.g. don't attach files of size 1GB).

- **Ensure the name of all teammates and their Concordia ID number are written on the first page of your report**. A single zip file should be sent to the instructor for each team.

2 The project

The objective of the project is to use reinforcement learning and/or dynamic programming methods seen in class to solve a sequential decision problem of your choice.

2.1 Problem selection and data

The first thing you need to decide is what problem are you going to solve. The problem should be a sequential decision problem. It should be sufficiently complex so as not to lead to a trivial solution. Nevertheless, the problem selected should be reasonable for a term project in the sense that it should not be too hard to solve by a small team of students before the deadline with commonly available technology.[1]

Potential examples of interesting problems include (but are not limited to):

- Games,
  - Black jack, poker, 3D tic-tac-toe, video games RPG battles, exiting a maze, etc.
- Financial optimization,
  - Option pricing, investment portfolio management, hedging.
- k-armed bandit problems,
  - Generalizations involving dependence between rewards, selection of multiple arms, risk-averse bandits.

Once you've decided upon the problem you want to tackle, you have contact your instructor; your project problem must be pre-approved. The deadline for pre-approval of your problem is **Sunday February 17, 2019**. Write a short email to the instructor listing the members of your team and briefly explaining what is the problem you will study. The instructor will then reply to approve your project provided it is appropriate.

2.2 Analyses

The next step is tp decide on which method(s) you will use to solve the problem, and then implement them in R. The implementation should be your own; **you should not rely on external reinforcement learning**

[1]For instance. if you need a large cluster with several thousands computers running in parallel for a month to solve your problem, this would be excessive.

packages to apply your reinforcement learning method. It can be acceptable to use a package for a sub-component of your method; i.e. if you use a neural network to approximate the policy or the value function, you are not expected to re-code yourself the calibration of a neural network. However, the portion of the code related to dynamic programming or reinforcement learning should be your own.

You should then perform a performance assessment of the methods you've implemented which could answer for instance the following questions:

- If multiple methods were implemented, how do their solutions compare? Why are some methods more efficient than others in the context of your problem?
- How efficient are the methods you implemented; has the solution stabilized (i.e. converged)? After how much time (how many episodes or iterations)? Do you achieve the desired goal with a suitable performance?
- Do the methods used learn quickly or slowly (how faster do the methods improve at the beginning)?
- Have you used some approximation methods? Do you think such approximations are appropriate? Why ?

2.3 Report

Once you performed all of your analyses, these should be summarized in a pdf report.

The following sections should be included in the report:

- Introduction
  - This should describe the reinforcement learning problem you considered. A very brief description of the types of analyses that were performed should be given at this point.
  - Although it is not required, if ever you want to impress your instructor, a short literature review could be performed and documented. The literature involves listing (and very briefly describing) the work of other researchers who worked on similar problems. Scientific papers can be consulted for instance on Google Scholar or on the Concordia library website.
- Problem description
  - This should describe the sequential decision problem considered: what are the states, actions in each state and rewards? What is the dynamics of the system (what are the transition probabilities, of how do the sequence of states is generated)? What is the objective?
- Analyses (can be split into multiple subsections)
  - Here you should describe the methods you considered to attempt solving sequential decision problem. You should do a performance assessment of the methods; compare their respective performance (as a function of computational effort/time required).
  - You might also want to comment on your implementation. For instance, how did you label the various states so as to map them

into a sequence $1, \ldots, |\mathcal{S}|$? Same thing for the actions. You can comment on the structure of your code if you want (i.e. to make your code easier to understand and check)

- Conclusion
  - Summarize the problem you tackled, how you did, and provide the most salient results from your analysis.
  - Potential future avenues of research (additional analysis which could be performed) could be stated here.
- References (if applicable)

Note: if you write your report in the format of a scientific paper typically found in peer reviewed journals, your instructor will be positively impressed.

3 Important advices

- Ensure your document is clean and clearly written. Points can be deducted if the clarity is deemed inadequate. This includes putting comments within your R code to ensure readability.
- **IMPORTANT**: Some students might be tempted to include in their report every single result of every experiment they performed during their analyses to show how hard they worked. This might lead to a very voluminous report which would be hardly readable; this would clearly be inappropriate. For instance, reports should **never exceed thirty pages** (maybe excluding figures/tables), and should often be much shorter. Students should attempt mastering the art of parsimony while

writing their report by including only meaningful details that will help understanding their approach. It is difficult to read a report that is too long and that goes in all directions. To ensure your report is easily readable might require refining the text several times; writing good texts that are readable is not a trivial endeavor and takes time.

- When you provide tables with numerical results, ensure the number of digits is reasonable. For instance, it is sufficient to know that $\beta = 0.92$ instead of writing $\beta = 0.921095973$. Ensure your tables and figures are easily readable.

Good luck, and remember to try to have fun doing this project. Life is too short not to!