CMU 10-715: Homework 8

Online Learning and Q Learning

DUE: Dec. 12, 2022, 11:59 PM.

Instructions:

- **Collaboration policy:** Collaboration on solving the homework is allowed, after you have thought about the problems on your own. It is also OK to get clarification (but not solutions) from books, again after you have thought about the problems on your own. Please don't search for answers on the web, previous years' homeworks, etc. (please ask the TAs if you are not sure if you can use a particular reference). There are two requirements: first, cite your collaborators fully and completely (e.g., "Alice explained to me what is asked in Question 4.3"). Second, write your solution *independently*: close the book and all of your notes, and send collaborators out of the room, so that the solution comes from you only.
- **Submitting your work:** Assignments should be submitted as PDFs using Gradescope unless explicitly stated otherwise. Each derivation/proof should be completed on a separate page. Submissions can be handwritten, but should be labeled and clearly legible. Else, submission can be written in LaTeX.
- **Late days:** For each homework you get three late days to be used only when anything urgent comes up. No points will be deducted for using these late days. We will consider an honor system where we will rely on you to use the late days appropriately.

1 [50 pts] Online Learning

We want to predict if the stock market will go up or down. On day t, we observe the actual outcome $y_t \in \{-1, 1\}$. Assume we have N experts who would vote positive $(+1)$ or negative (-1) on each day t. Consider the following online learning algorithm for this binary prediction problem:

Algorithm 1 Randomized weighted majority algorithm

```
Input: Pool of $d$ experts: $S = \{1, \cdots, d\}$, number of days: $T$, Hyperparameter $\epsilon$
Initialize $w = (1, 1, \ldots, 1) \in \mathbb{R}^d$
for $t \leftarrow 1$ to $T$ do
    (Randomly) Choose an expert $i \in [d]$ with probability $w_i/(\sum_j w_j)$
    Make a prediction based on chosen expert
    Update the weights for every wrong expert $j \in [d]$ as follows:
        $w_j \leftarrow (1 - \epsilon) w_j$ .
end for
```

- Implement the randomized weighted majority algorithm and run the algorithm on the dataset *online_data.csv*. For this dataset, we have $d = 10$ and $T = 10k$. The goal here is to predict as well as the best expert does in the hindsight. Append your code to your PDF submission.

- Compute your regret for the values of epsilon in the following set:

$$\mathcal{E} = \{0.05, 0.1, 0.3, 0.5, 0.7, 0.9, 0.95\}.$$

  For each epsilon $\epsilon \in \mathcal{E}$, run 3 different seeds (100, 200, 300), and plot the mean as well as mean $\pm$ standard deviation of the regret versus epsilon. Compare your results with a simple baseline where you choose the best expert so far. Add this as a baseline in the plot. Note that this baseline doesn't depend on epsilon and hence, we expect a horizontal line for the baseline. Summarize your observations in 2-3 sentences.

2 [50 pts] Q Learning

In this question you will implement the Q Learning algorithm.

Algorithm 2 Q Learning

```
Require: $\alpha, \epsilon, \gamma, ENV$
  while $episode \leq MAX\_ITERATIONS$ do
      Reset ENV and get initial state $s$
      while episode not done do
          Sample action $a$                                   ▷ according to $\epsilon$-greedy strategy
          Get new state $s'$, reward $r$
          Update $Q[s,a] = (1-\alpha)Q[s,a] + \alpha(r + \gamma \max_{a'} Q[s',a'])$
          update State $s$ to new state $s'$
      end while
  end while
  return Q
```

We have provided the starter code in *qlearning_starter.py*, where you will be using FrozenLake environment from gym. To install gym, simply run `pip install gym` or `pip3 install gym`

For more details about gym, please refer to https://github.com/openai/gym. For the following questions, the start code has outlined *TODO* blocks. Don't forget to append your code to your PDF submission.

(a) (20 points) Run the Q Learning algorithm using the default hyperparameters given ($\alpha = 0.8, \gamma = 0.9, \epsilon = 0.7$). You'll also need to write the $\epsilon-$greedy algorithm in order to sample the action at each time step.

(b) (30 points) Evaluate the environment using an $\epsilon-$greedy strategy.

- For each ϵ in $\{0, 0.25, 0.5\}$, report an episode of the game (see example below). You can use *env.render*() and plot the whole game trajectory.
- Describe the difference among the three episodes, and provide your interpretation.

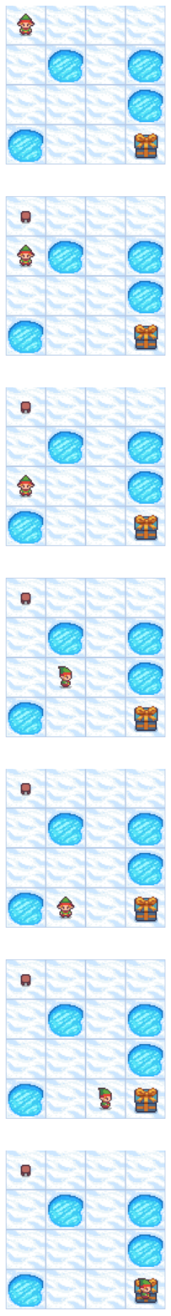

Figure 1: Example of an episode