15-889e Real Life Reinforcement Learning

Homework 2

Due 11/23 at 1:30pm

Rules:

1. Homework is due Monday 11/23 at 1:30pm. Please see course website for policy on late submission.
2. We recommend that you typeset your homework using appropriate software such as LaTeX. If you are writing please make sure your homework is cleanly written and legible. The TAs will not invest undue effort to decrypt bad handwriting.
3. Please submit the written portion of the homework and the programming portion separately.
4. You may do the programming part of the homework in pairs. If you choose to do this, you should submit a single homework for that part, with both of your names and andrew ids.
5. You are allowed to discuss ideas about the written portion of the homework with others, but you must write up your own solution. If you do discuss the written portion with others, please indicate your collaborators in your submission. Also, if you use any outside sources (wikipedia, a technical paper or chapter, etc) to answer the questions, please list them.

Problem 1:

Recall that *importance sampling* can be used to generate an estimate of the performance of one policy, called the *evaluation policy*, given a trajectory that was generated by a different policy, called the *behavior policy*. The importance sampling estimate for a trajectory τ is:

$$IS(\tau) = \prod_{t=1}^{H} \frac{\pi_e(A_t|S_t)}{\pi_b(A_t|S_t)} \sum_{t=1}^{H} \gamma^{t-1} R_t,$$

where π_e is the evaluation policy, π_b is the behavior policy, γ is the reward discount factor, H is the trajectory length, and S_t, A_t, and R_t are the state, action, and reward at time t. The product in this equation is called the *importance weight*:

$$IW(\tau) = \prod_{t=1}^{H} \frac{\pi_e(A_t|S_t)}{\pi_b(A_t|S_t)}$$

and the sum is called the *return*. If there are multiple trajectories, $\mathcal{D} = \{\tau_i\}_{i=1}^{n}$, then the mean IS estimator is:

$$IS(\mathcal{D}) = \frac{1}{n} \sum_{i=1}^{n} IS(\tau).$$

A) Show that the expected value of $IS(\tau)$ if τ is produced by π_b is the expected return of π_e.

B) Show that

$$E[IW(\tau)|\tau \sim \pi_b] = 1,$$

and therefore that

$$E\left[\sum_{i=1}^{n} IW(\tau_i)\right] = n.$$

C) Due to this result, a researcher proposes using $\frac{1}{\sum_{i=1}^{n} IW(\tau_i)}$ rather than $\frac{1}{n}$ when averaging the importance sampling estimates from many trajectories. The researcher calls this new estimator *approximate importance sampling* and is defined as:

$$AIS(\mathcal{D}) = \frac{1}{\sum_{i=1}^{n} IW(\tau_i)} \sum_{i=1}^{n} IS(\tau).$$

Show that $AIS(\mathcal{D}) \in [0, HR_{max}]$ if the rewards are bounded by $R_t \in [0, R_{max}]$.

D) Why is the result in C) important? Why does it suggest that approximate importance sampling might give better estimates than ordinary importance sampling?

E) Show that $AIS(\mathcal{D})$ is not always an unbiased estimator of the expected return of π_e.

F) What is $AIS(\mathcal{D})$ an unbiased estimator of if $\mathcal{D}$ contains only a single trajectory? Show this result mathematically.

G) Given what we know about approximate importance sampling, could we use it with the Chernoff-Hoeffding inequality to produce a confidence interval on the performance of the evaluation policy?

H) If you are given an evaluation policy, π_e, but can select the behavior policy, π_b, you might do so with the goal of minimizing the variance of *IS*. Show (e.g., by counter-example) that using $\pi_b = \pi_e$ is not necessarily optimal.

Expected Regret bounds

Assume a reinforcement learning algorithm A for discounted infinite-horizon MDPs has expected regret

$$\mathbb{E}_* \left[\sum_{t=1}^{T} r_t \right] - \mathbb{E}_A \left[\sum_{t=1}^{T} r_t \right] = f(T) \tag{1}$$

for all $T > 0$ where $\mathbb{E}_*$ is over probability distribution with respect to the optimal policy π_* and $\mathbb{E}_A$ is the expectation w.r.t. the algorithm's behavior. We further assume that rewards $r_t \in [0, 1]$ are normalized.
a) Show that for any $\epsilon' > 0$ and $T' \geq H \log \frac{H}{\epsilon'}$ where $H = 1/(1-\gamma)$ with discount factor γ it holds

$$\left| V(s) - \sum_{t=1}^{T'} \gamma^{t-1} \mathbb{E}[r_t | s_1 = s] \right| \leq \epsilon' \tag{2}$$

b) Show that for any $\epsilon' > 0$ and $T' \geq H \log \frac{H}{\epsilon'}$

$$\mathbb{E}_*[V_*(s_{T+1})] - \mathbb{E}_A[V_A(s_{T+1})] \leq f(T' + T) - f(T) + 2\epsilon' \tag{3}$$

where V_A is the value function of the (possibly nonstationary) policy that algorithm A follows.
c) Assume now $f(T) = \sqrt{T}$. Show that for any $\epsilon > 0$ and $t \geq 1 + \frac{H^2}{\epsilon^2} \log^2 \frac{4H}{\epsilon}$

$$\mathbb{E}_*[V_*(s_t)] - \mathbb{E}_A[V_A(s_t)] \leq \epsilon \tag{4}$$

This shows that for all time steps after t (as defined above), the algorithm will make near optimal decisions in expectation, similar to the PAC setting. Hint: To use the results you have already derived (in parts a or b), it will be helpful to set ϵ' to a function of ϵ.

Sample Efficency of RL Algorithms for HIV Treatment

For the programming you will get to investigate how different online RL algorithms perform experimentally. The domain is to create structured treatment interruption strategies for HIV patients. Posing this as a reinforcement learning problem is described in "Clinical Data Based Optimal STI Strategies for HIV: a Reinforcement Learning Approach": http://orbi.ulg.ac.be/bitstream/2268/13368/1/ernst-benelearn-2006.pdf

A simulator for this domain has already been previously implemented in RLPy. RLPy is a software tool to make it easier to run RL and sequential decision making experiments. We highly encourage you to use RLPy for this assignment. If you would like to not use it, and implement a simulator yourself, please contact us first to check if your alternate plan is acceptable.

RLPy is freely available on the web, and documentation about using it and pointers to the code base are available at: http://rlpy.readthedocs.org/en/latest/

For this homework you will be implementing online sample efficient RL algorithms for this HIV domain. Please implement and compare the results for at least 2 of the following:

- A PAC algorithm
- A regret minimizing algorithm
- A Bayes optimal algorithm
- An algorithm designed to be empirically data efficient

One of the two approaches you use should involve at least one of the following: Gaussian Processes, random forests, neural nets or Monte Carlo tree search. You are welcome to use packages to implement GPs, NN, or random forests. There is existing code for MCTS for POMDPs that you are welcome to build on, but it will require modification for the RL case.

You should implement your algorithms, and perform at least 10 runs of each algorithm, and present graphs showing

- the average reward per time step vs the time step
- the average cumulative reward per time step vs the time step

Please do this for two very different horizon lengths, such as $H = 10$ and $H = 200$. Plot these results separately, as some algorithm's behavior depends on the input horizon.

Algorithms 1-3 have formal guarantees under certain circumstances, but empirically it is very likely to be more effective to tune the parameters. You should justify your choice of the parameters you use, and/or explore more than one option.

In addition to the required figures, you should include a short writeup of what you did, how you ran the experiments, and whether and why one or the other algorithm is more effective at leveraging data.

Please submit your code (that can be run using RLPy).