


Branch: master

reinforcement-learning / MC /

Create new file

Find file

History



pieromacaluso

Updated links to new version of Sutton's book

Latest commit a35df15 on Mar 13

..

<div> <div></div> <div>Blackjack Playground.ipynb</div> </div>	Update playground output	last year
<div> <div></div> <div>MC Control with Epsilon-Greedy Policies Solutio...</div> </div>	Fix typo in MC Control	last year
<div> <div></div> <div>MC Control with Epsilon-Greedy Policies.ipynb</div> </div>	Change kernel to python3	2 years ago
<div> <div></div> <div>MC Prediction Solution.ipynb</div> </div>	Change kernel to python3	2 years ago
<div> <div></div> <div>MC Prediction.ipynb</div> </div>	Minor fix: sync sample policy with the solution	last year
<div> <div></div> <div>Off-Policy MC Control with Weighted Importan...</div> </div>	Change kernel to python3	2 years ago
<div> <div></div> <div>Off-Policy MC Control with Weighted Importan...</div> </div>	Change kernel to python3	2 years ago
<div> <div></div> <div>README.md</div> </div>	Updated links to new version of Sutton's book	3 months ago

 README.md

Model-Free Prediction & Control with Monte Carlo (MC)

Learning Goals

- Understand the difference between Prediction and Control
- Know how to use the MC method for predicting state values and state-action values
- Understand the on-policy first-visit MC control algorithm
- Understand off-policy MC control algorithms
- Understand Weighted Importance Sampling
- Understand the benefits of MC algorithms over the Dynamic Programming approach

Summary

- Dynamic Programming approaches assume complete knowledge of the environment (the MDP). In practice, we often don't have full knowledge of how the world works.
- Monte Carlo (MC) methods can learn directly from experience collected by interacting with the environment. An episode of experience is a series of (State, Action, Reward, Next State) tuples.
- MC methods work based on episodes. We sample episodes of experience and make updates to our estimates at the end of each episode. MC methods have high variance (due to lots of random decisions within an episode) but are unbiased.
- MC Policy Evaluation: Given a policy, we want to estimate the state-value function $V(s)$. Sample episodes of experience and estimate $V(s)$ to be the reward received from that state onwards averaged across all of your experience. The same technique works for the action-value function $Q(s, a)$. Given enough samples, this is proven to converge.
- MC Control: Idea is the same as for Dynamic Programming. Use MC Policy Evaluation to evaluate the current policy then improve the policy greedily. The Problem: How do we ensure that we explore all states if we don't know the full environment?
- Solution to exploration problem: Use epsilon-greedy policies instead of full greedy policies. When making a decision act randomly with probability epsilon. This will learn the optimal epsilon-greedy policy.
- Off-Policy Learning: How can we learn about the actual optimal (greedy) policy while following an exploratory (epsilon-greedy) policy? We can use importance sampling, which weighs returns by their probability of occurring under the policy we want to learn about.

Lectures & Readings

Required:

- [Reinforcement Learning: An Introduction](#) - Chapter 5: Monte Carlo Methods

Optional:

- David Silver's RL Course Lecture 4 - Model-Free Prediction ([video](#), [slides](#))
- David Silver's RL Course Lecture 5 - Model-Free Control ([video](#), [slides](#))

Exercises

- Get familiar with the [Blackjack environment \(Blackjack-v0\)](#)
- Implement the Monte Carlo Prediction to estimate state-action values
 - [Exercise](#)
 - [Solution](#)
- Implement the on-policy first-visit Monte Carlo Control algorithm
 - [Exercise](#)
 - [Solution](#)
- Implement the off-policy every-visit Monte Carlo Control using Weighted Important Sampling algorithm
 - [Exercise](#)
 - [Solution](#)