Monte Carlo Methods

Rupam Mahmood

February 10, 2020

Estimating expectation by sample averaging

$$v_\pi(0) \doteq E_\pi\left[G_0 \,|\, S_0 = 0\right] \approx \sum_{e=1}^{N} \frac{G_{0,e}}{N}$$

Return of eth episode

- ✓ Sample average is an unbiased estimator of expectation

  Lemma 1 (Mahmood 2017)

- ✓ Sample average is a consistent estimator of expectation

  Lemma 3 (Mahmood 2017)

Unbiased and consistent estimation

Say X_i is an iid random variable

The sample average $Z_n = \frac{\sum_{i=}^{n} X_i}{n}$ is an estimate of $E[X_i]$

Then we have $E[Z_n] = E[X_i]$; unbiasedness of Z_n

And we have $P\left(\lim_{i\to\infty} Z_n = E[X_i]\right) = 1$; consistency of Z_n

Monte Carlo prediction

First-visit MC prediction, for estimating $V \approx v_\pi$

Input: a policy π to be evaluated

Initialize:

$V(s) \in \mathbb{R}$, arbitrarily, for all $s \in \mathcal{S}$

$Returns(s) \leftarrow$ an empty list, for all $s \in \mathcal{S}$

Loop forever (for each episode):

Generate an episode following π: $S_0, A_0, R_1, S_1, A_1, R_2, \ldots, S_{T-1}, A_{T-1}, R_T$

$G \leftarrow 0$

Loop for each step of episode, $t = T-1, T-2, \ldots, 0$:

$G \leftarrow \gamma G + R_{t+1}$

Unless S_t appears in $S_0, S_1, \ldots, S_{t-1}$:

Append G to $Returns(S_t)$

$V(S_t) \leftarrow \text{average}(Returns(S_t))$

Friday's Monte Carlo code: Is it every-visit or first-visit?

```python
import numpy as np

neps = 1000; ns = 1; g = 0.9; probl = 0.1

pol = np.zeros((ns, 2)); pol[0, 0] = probl; pol[0, 1] = 1 - probl

rewards = np.zeros((ns, 2)); rewards[0, 0] = -1; rewards[0, 1] = +1

rets = []
for ep in range(neps):
    s = 0; ret = 0; t = 0
    while s != 1:
        rnd = np.random.random()
        if rnd < pol[s, 0]: a = 0
        else: a = 1

        sp = s + a
        r = rewards[s, a]
        ret += r *(g**t)
        s = sp; t = t + 1
    rets.append(ret)
print("Average return:", np.mean(rets))
```

Friday's Monte Carlo code: what's the difference between this and MC prediction algorithm from the book?

First-visit MC prediction, for estimating $V \approx v_\pi$

Input: a policy π to be evaluated

Initialize:

$V(s) \in \mathbb{R}$, arbitrarily, for all $s \in \mathcal{S}$

$Returns(s) \leftarrow$ an empty list, for all $s \in \mathcal{S}$

Loop forever (for each episode):

Generate an episode following π: $S_0, A_0, R_1, S_1, A_1, R_2, \ldots, S_{T-1}, A_{T-1}, R_T$

$G \leftarrow 0$

Loop for each step of episode, $t = T-1, T-2, \ldots, 0$:

$G \leftarrow \gamma G + R_{t+1}$

Unless S_t appears in $S_0, S_1, \ldots, S_{t-1}$:

Append G to $Returns(S_t)$

$V(S_t) \leftarrow \text{average}(Returns(S_t))$

```
import numpy as np

neps = 1000; ns = 1; g = 0.9; probl = 0.1

pol = np.zeros((ns, 2)); pol[0, 0] = probl; pol[0, 1] = 1 - probl

rewards = np.zeros((ns, 2)); rewards[0, 0] = -1; rewards[0, 1] = +1

rets = []
for ep in range(neps):
    s = 0; ret = 0; t = 0
    while s != 1:
        rnd = np.random.random()
        if rnd < pol[s, 0]: a = 0
        else: a = 1

        sp = s + a
        r = rewards[s, a]
        ret += r *(g**t)
        s = sp; t = t + 1
    rets.append(ret)
print("Average return:", np.mean(rets))
```

Monte Carlo version of classical policy iteration (with construction of greedy policies)

$$\pi_0 \xrightarrow{\text{E}} q_{\pi_0} \xrightarrow{\text{I}} \pi_1 \xrightarrow{\text{E}} q_{\pi_1} \xrightarrow{\text{I}} \pi_2 \xrightarrow{\text{E}} \cdots \xrightarrow{\text{I}} \pi_* \xrightarrow{\text{E}} q_*$$

Could we use state value estimates?

Why is exploring starts necessary?

Why is exploring starts impractical?

What is the other impractical assumption here?

Monte Carlo control with generalized policy iteration

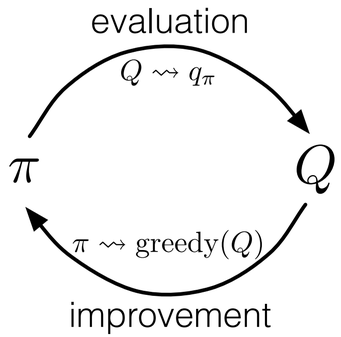
Monte Carlo ES (Exploring Starts), for estimating $\pi \approx \pi_*$

Initialize:

$\pi(s) \in \mathcal{A}(s)$ (arbitrarily), for all $s \in \mathcal{S}$
$Q(s,a) \in \mathbb{R}$ (arbitrarily), for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$
$Returns(s,a) \leftarrow$ empty list, for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$

Loop forever (for each episode):

Choose $S_0 \in \mathcal{S}$, $A_0 \in \mathcal{A}(S_0)$ randomly such that all pairs have probability > 0
Generate an episode from S_0, A_0, following π: $S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T$
$G \leftarrow 0$
Loop for each step of episode, $t = T-1, T-2, \ldots, 0$:
$G \leftarrow \gamma G + R_{t+1}$
Unless the pair S_t, A_t appears in $S_0, A_0, S_1, A_1 \ldots, S_{t-1}, A_{t-1}$:
Append G to $Returns(S_t, A_t)$
$Q(S_t, A_t) \leftarrow \text{average}(Returns(S_t, A_t))$
$\pi(S_t) \leftarrow \arg\max_a Q(S_t, a)$



Monte Carlo control without exploring start

On-policy first-visit MC control (for ε-soft policies), estimates $\pi \approx \pi_*$

Algorithm parameter: small $\varepsilon > 0$

Initialize:

$\pi \leftarrow$ an arbitrary ε-soft policy

$Q(s,a) \in \mathbb{R}$ (arbitrarily), for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$

$Returns(s,a) \leftarrow$ empty list, for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$

Repeat forever (for each episode):

Generate an episode following π: $S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T$

$G \leftarrow 0$

Loop for each step of episode, $t = T-1, T-2, \ldots, 0$:

$G \leftarrow \gamma G + R_{t+1}$

Unless the pair S_t, A_t appears in $S_0, A_0, S_1, A_1 \ldots, S_{t-1}, A_{t-1}$:

Append G to $Returns(S_t, A_t)$

$Q(S_t, A_t) \leftarrow \text{average}(Returns(S_t, A_t))$

$A^* \leftarrow \arg\max_a Q(S_t, a)$ (with ties broken arbitrarily)

For all $a \in \mathcal{A}(S_t)$:

$$\pi(a|S_t) \leftarrow \begin{cases} 1 - \varepsilon + \varepsilon/|\mathcal{A}(S_t)| & \text{if } a = A^* \\ \varepsilon/|\mathcal{A}(S_t)| & \text{if } a \neq A^* \end{cases}$$