Reinforcement Learning: An Introduction-Multi-armed Bandits

Lastly modified on 2023-01-08

차 례

Preface

내가 보려고 만든 sutton cheetsheet

I. Multi-armed Bandits

강화학습을 다른 종류의 학습방법과 구분 짓는 가장 중요한 특징은 올바른 행동을 알려주는 지침(instruct)가 아닌 행동의 좋고 나쁨을 평가(evaluate)하는 훈련정보를 사용한다는 점이다. 즉, 올바른 행동(exploit)을 알기 위해서는 특정 행동이 얼마나 좋은지 피드백을 통해서 평가 해야만 하며, 더 좋은 행동을 찾기 위해 탐험(expolration)이 필요하다. 본 장에서는, 강화학습에서의 중요한 요소 중 하나인 'expolration-expolitation' 측면에서 가장 단순한 환경인 Multi-armed Bandits에 대해 알아본다.

A k-armed Bandit Problem

k개의 슬롯머신을 생각해보자. 각 슬롯머신은 각기 다른 잭팟 확률을 가지고 있다. 이때, 10분당 하나의 슬롯머신의 레버를 당길 수 있을때 10시간 동안 최대의 상금을 가져가기 위해선 어떻게 해야할까?

시간 t에서 k-개의 레버 중 하나를 선택하는 행동을 A_t로 표현하고, 그에 따른 보상을 R_t라고 하자. 임의의 행동 a의 가치는 행동 a가 선택되었을 때, 기대값으로 표현한다.

$$q_*(a) = \mathbb{E}[R_t|A_t = a]$$

그러나, 행동의 가치는 알려져 있지 않기 때문에 우리는 이 가치를 추정해야만 한다. 시간 t에서 추정한 행동의 가치는 $Q_t(a)$로 표현한다. 이때, 시간 t에서 평가한 행동의 가치 중 최대의 가치를 가지는 행동을 수행할 경우 exploiting한다고 하며, 정확한 평가를 위해(장기적으로 더 좋은 결과를 위해) 다른 슬롯 머신을 당겨보는 행동을 exploraing 한다고 말한다.

Action-value Methods

행동의 가치를 추정하고, 추정값으로 부터 행동을 선택하는 방법을 총칭하여 action-value method라고 한다. 행동 가치($q_*(a) = \mathbb{E}[R_t|A_t = a]$)를 추정 방법 중 가장 직관적이며 쉬운 방법은 실제 받은 보상을 산술평균하는 것이다.

$$Q_t(a) = \frac{\sum_i^{t-1} R_i \cdot 1_{A_t=a}}{\sum_i^{t-1} 1_{A_t=a}}$$

또한 가장 쉬운 행동 선택 방법은 추정 가치가 최대인 행동 중 하나를 greedy하게 선택하는 것이다.

$$A_t = argmax_a Q_t(a)$$

또 하나의 대안은 $\epsilon - greedy$ 방법으로, ϵ의 확률로 random하게 모든 슬롯 머신을 당기며, 그 이외의 확률은 greedy 행동을 하는 방법이다. 이 경우 레버를 당기는 횟수가 무한정 하다면, $q*(a) = Q_t(a)$로 converge 한다.

A simple bandit algorithm

Initialize, for $a = 1$ to k:
$Q(a) \leftarrow 0$
$N(a) \leftarrow 0$

Loop forever:
$A \leftarrow \begin{cases} \text{argmax}_a Q(a) & \text{with probability } 1-\varepsilon \quad \text{(breaking ties randomly)} \\ \text{a random action} & \text{with probability } \varepsilon \end{cases}$
$R \leftarrow bandit(A)$
$N(A) \leftarrow N(A) + 1$
$Q(A) \leftarrow Q(A) + \frac{1}{N(A)}[R - Q(A)]$

The 10-armed Testbed

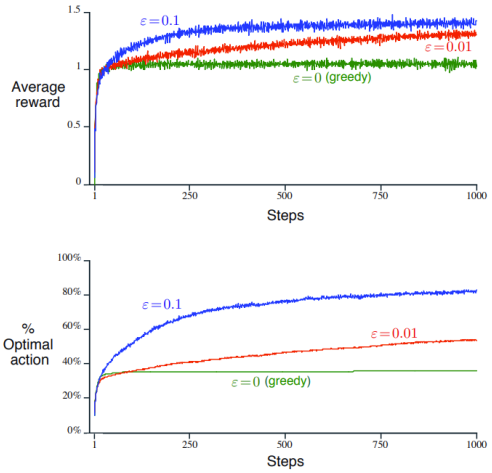


Figure 2.2: Average performance of ε-greedy action-value methods on the 10-armed testbed. These data are averages over 2000 runs with different bandit problems. All methods used sample averages as their action-value estimates.

10-aramed bandit 문제를 생각해보자. 이때 각 bandit의 가치, $q_*(a)$는 $N(0,1)$에서 무작위로 선택된 값으로 설정되며 보상은 $R_t \sim N(q_*(a), 1)$를 따른다고 하자.

그림 2-2은 $\epsilon - greedy$와 $greedy$ 방법을 비교한 그림이다.

어느 방법이 좋은지는 문제에 따라 다르다.

- 예를 들어, 분산이 0인 경우는 greedy 방법이 좋을 수 있다.
- 반대로 분산이 매우 큰 경우 e-greedy 방법이 더 좋은 대안이 될 수도 있다.
- 그러나 이를 deterministic한 문제의 경우 greedy 방법이 항상 좋다고 오해하면 안된다. 일반적으로 대부분의 강화학습의 경우인 nonstationary 문제의 경우, 즉 행동가치의 참값이 시간에 따라 변하는 경우 탐험하는 것이 필요하다.

Incremental Implementation

지금까지 가치 추정 방법은 표본평균으로 해왔다. 하지만 n이 커질 수록 모든 값들을 기록하는 것은 비효율 적이다. 따라서 아래와 같은 방법을 Incremental Implementation을 주로 이용한다.

$$\begin{aligned} Q_{n+1} &= \frac{1}{n} \cdot \sum_{i}^{n} R_i \\ &= \frac{1}{n}[R_n + \sum_{i}^{n-1} R_i] \\ &= \frac{1}{n}[R_n + (n-1)Q_n] \\ &= \frac{1}{n}[R_n + nQ_n - Q_n] \\ &= Q_n + \frac{1}{n}[R_n - Q_n] \end{aligned} \tag{1}$$

Tracking a Nonstationary Problem

앞서 언급한 averaging method는 stationary bandit[1] 문제에는 적합하다. 그러나 우리가 후에 다루게 될 강화학습 문제에서는 대부분의 경우 nonstationary[2]이다.

따라서, 이 경우 과거의 리워드 보다 현재의 리워드에 좀더 많은 가중치를 주는 것이 적절할 수 있다. 가장 기본적인 방법은 constant-step size parameter, $\alpha \in (0, 1]$를 도입하는 것이다. 예를 들어 아래와 같이 우리는 Incremental update rule를 정할 수 있다.

$$Q_{n+1} = Q_n + \alpha[R_n - Q_n]$$

[1] 시간에 따라 리워드의 확률 분포가 변하지 않음.

[2] 시간에 따라 리워드의 확률 분포가 변함. 따라서 행동가치의 참값이 시간에 따라 변함.

이것은 Q_{n+1}를 weight average하는 것과 같은 효과를 지니게 되며, 아래와 같이 과거의 reward 텀들과 Q_1으로 표현 할 수 있다.[3]

[3] 수식 전개는 매우 간략함으로 생략하며 책을 참고하길 바란다.

$$Q_{n+1} = (1-\alpha)^n Q_1 + \sum_i^n \alpha \cdot (1-\alpha)^{n-i} R_i$$

위의 수식을 살펴 보면, 현재와 가까운 보상들에 대해서는 많은 가중치를 주는 것을 확인할 수 있으며, 종종 exponential recency-weighted average라고 불리기도 한다.

앞서 살펴 본 $\alpha = \frac{1}{n}$(표본 평균과 같음) 경우와 같이, 시간에 따라 다르게 설정할 수도 있다. 표본 평균의 방법의 경우, 큰수의 법칙에 따라 참값으로 수렴됨이 보장된다. 그러나 이것은 특수한 케이스이며 모든 경우 그러하지는 않다. 잘 알려진 확률이론에 따르면, 아래의 조건을 만족할 경우 수렴성이 보장된다.

$$\sum_i^\infty \alpha_n(a) = \infty, \sum_i^\infty \alpha_n^2(a) < \infty$$

이 조건에 따르면, 고정 step-size α는 두번째 조건을 만족하지 않는다. 즉, 추정 값이 완전히 수렴하지 않고 최근 값에 연속적으로 변한다는 것이다. 하지만 non-stationary 환경에서는 이와 같은 사실은 바람직한 것이며, 일반적으로 practical 하게 접근할때 위의 이론은 잘 사용하지 않는다(∞를 다 계산할수 없다.).

Optimistic Initial Values

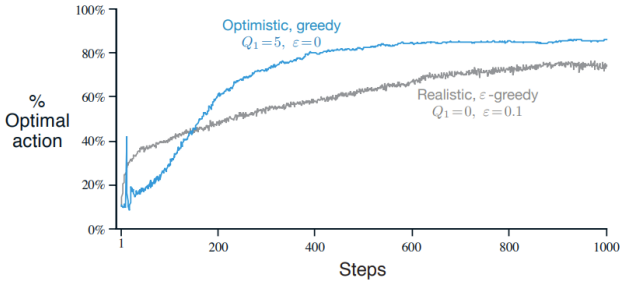


Figure 2.3: The effect of optimistic initial action-value estimates on the 10-armed testbed. Both methods used a constant step-size parameter, $\alpha = 0.1$.

지금까지 살펴본, 행동가치의 초기 추정값 $Q_1(a)$는 0으로 설정되었다. 이경우 고정 step-size α의 경우 bias가 n이 커짐에 따라 줄어들긴 하지만 항상 지속된다. 그러나 이러한 bias는 practical 하게는 문제가 되진 않는다. 오히려 반대로 생각해보면, 사전

지식을 알고 있다면 bias를 통해서 expolration 하게 만들 수 있다.

예를 들어, 앞서 언급한 The 10-armed Testbed의 경우를 생각해보자. 이 문제에서는 $q_*(a)$는 평균이 0이고 분산이 1인 표준 정규분포에서 선택되었다. 이때 행동 가치의 초기 추정값을 5로 설정한다면 어떻게 될까?[4] 어떤 행동이 초기에 선택되더라 하더라도 이 것은 초기 추정치 보다 낮은 값을 가지게 된다. 따라서 leaner는 이 보상 값에 대해서 실망하게 될것이며, 다른 행동을 선택 하게 될것이다.[5]

하지만 위 방법은 stationary 문제의 경우 적합한 방법이지만, non-stationary의 경우 적합하지 않다. 탐험에 대한 원동력이 본디 일시적이기 때문이다. 만약 초기값 설정 후 많은 시간이 지나, 문제가 변경되어 탐험할 필요가 생긴다면 이 방법은 뒤의 문제에 대해서는 영향을 주지 못한다.

[4] 5라는 초기 추정치는 매우 긍정적인 설정 값이다 = Optimistic Initial values 이다.

[5] 예를 들어, $Q_1(a_1) = 5, Q_1(a_2) = 5$의 초기값으로 설정하였다고 생각해보자. 이후 a_1를 선택하여 0의 보상값을 받으면, $Q_2(a_1) = 2.5$의 추정치로 수정되며 따라서, $Q_2(a_2) = 5$가 높은 값이기 때문에 행동 a_2를 선택할 것이다.

Upper-Confidence-Bound Action Selection

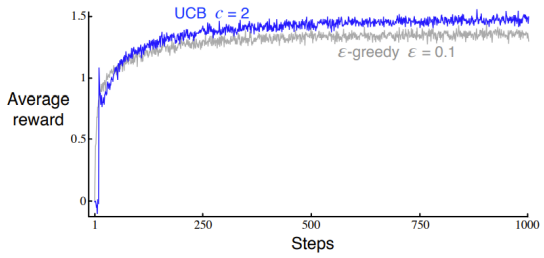


Figure 2.4: Average performance of UCB action selection on the 10-armed testbed. As shown, UCB generally performs better than ε-greedy action selection, except in the first k steps, when it selects randomly among the as-yet-untried actions.

앞서 언급한 방법들을 살펴보자. $greedy$방법의 경우 현재로서는 최선의 선택을 하지만, 실제로는 다른 행동이 좋을 가능성이 존재한다. $e-greedy$방법의 경우 ϵ 확률만큼 다른 행동을 탐험하지만, 무작위로 탐험을 진행한다. 어떤 다른 방법이 있을까?

UCB방법은 행동가치의 추정의 불확실성을 고려해서, 잠재가치가 높은 행동을 선택하는 방법이다. 현재 추정값의 confidence interval을 생각해보자. 어떤 행동의 경우, 다른 행동과 비교해 평균 추정 값이 비록 낮지만 upper confidece bound는 높을 수 있다. 즉, 이 행동이 잠재 가치가 높을 수 있다는 것이며, UCB는 이러한 아이디어에서 출발하며 아래와 같은 rule을 가진다.

$$A_t = argmax_a[Q_t(a) + c\sqrt{\frac{lnt}{N_t(a)}}]$$

여기서 $N_t(a)$는 시간 t이전에 행동 a가 선택된 횟수를 나타내며, c는 expolration

의 세기를 결정하는 parameter 이다. 위의 식에 따르면, UCB 전략은 신뢰 상한이 가장 큰 행동을 선택하며, 행동이 선택된 횟수가 많아 짐에 따라 신뢰 상한은 줄어 든다 .

위의 그림에서 보이듯이, UCB는 꽤 reasonalble 한 전략이며 종종 $e-greedy$ 방법보다 좋은 결과를 보여준다. 하지만 이를 강화학습문제로 확장할 경우, larage-state-space와 non-stationary, approximation error 등을 고려해야 함으로 일반적으로 practical 하지 않다.

Gradient Bandit Algorithms

이전까지 언급했던 방법들은, 행동 가치를 추정하고 그것을 바탕으로 다양한 행동 선택 전략을 구사했다. 이번에는 행동 가치를 직접적으로 추정하지 않는 방법을 살펴 볼것이다.

Gradient Bandit 방법은 강화학습에서의 Policy gradient 방법의 원형으로, 행동가치를 추정하는 것이 아니라 어떤 행동의 prefrence;$H_t(a)$를 학습하는 방법이다. 이때 행동의 preference가 높을 수록 해당 행동은 더 많이 선택된다. 흔히 deeplearning에서 classification 문제와 같이 soft-max 분포를 통해서 행동을 선택하게 한다.

$$Pr[A_t = a] = \frac{e^{H_t(a)}}{\sum_b^k e^{H_t(b)}} = \pi_t(a_t)$$

update rule은 아래와 같이 진행한다.

$$H_{t+1}(A_t) = H_t(A_t) + \alpha(R_t - \bar{R}_t)(1 - \pi(A_t))$$

$$H_{t+1}(a) = H_t(A_t) - \alpha(R_t - \bar{R}_t)\pi(A_t), \forall a \neq A_t$$

이때, $\bar{R}_t$는 시간 t까지 Incrementally 계산된 평균 값이며, 초기 값, $H_1(a)$는 0으로 설정한다. 위 식에 따르면, 선택 된 행동의 보상이 평균값보다 크게 되면 이 행동의 prefrence를 높히게 되며, 반대의 경우 줄이게 된다. 또한 해당 행동과 다른 행동들은 이와 반대로 update훟 진행한다. 즉, 행동의 선택에 있어서 다른 행동들의 상대적 선호도의 크기가 중요 한것이며, 선호도는 어떠한 보상값, 또는 행동 가치값의 해석의 기준이 될 수 없다. 위와 같은 방식은 일반적으로 강화학습에서 많이 사용되는 policy gradient ascent, Advantage와 비슷한 맥락으로 이해할 수 있다.

Excercise 2.1

In e-greedy action selection, for the case of two actions and ϵ = 0.5, what is the probability that the greedy action is selected?

Answer :

suppose, a_1 is greedy action , for example $Q_t(a_1) = 10, Q_t(a_2) = 1$ case 1 (exploit) : Action a_1 selected, (50%) case 2 (exploration) : a_1 is selected randomly in Action set $\{a_1, a_2\}$

$$P[A_t = a_1] = 0.5 + 0.5 \cdot 0.5 = 0.75$$

Exercise 2.2

Bandit example Consider a k-armed bandit problem with k = 4 actions, denoted 1, 2, 3, and 4. Consider applying to this problem a bandit algorithm using e-greedy action selection, sample-average action-value estimates, and initial estimates of $Q_1(a)$ = 0, for all a. Suppose the initial sequence of actions and rewards is A_1 = 1, R_1 = -1, A_2 = 2, R_2 =1, A_3 = 2, R_3 =-2, A_4 = 2, R_4 =2, A_5 = 3, R_5 = 0. On some of these time steps the " case may have occurred, causing an action to be selected at random. On which time steps did this definitely occur? On which time steps could this possibly have occurred?

Answer :

The table below calculates the Q-estimat on each time step , where row represent time step and column represent estimate value, $Q(a_1), Q(a_2)...$

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ -1 & -1/2 & 0 & 0 \\ -1 & 1/3 & 0 & 0 \\ -1 & 1/3 & 0 & 0 \end{bmatrix} \tag{2}$$

at time step 4,5 optimal action was a_3, a_2, but diffrent action was chosen (ϵ case have occured)

Excerice 2.3

In the comparison shown in Figure 2.2, which method will perform best in the long run in terms of cumulative reward and probability of selecting the best action? How much better will it be? Express your answer quantitatively.

Answer :

to do.

Excercise 2.4

If the step-size parameters, α_n, are not constant, then the estimate Q_n is a weighted average of previously received rewards with a weighting different from that given by (2.6). What is the weighting on each prior reward for the general case, analogous to (2.6), in terms of the sequence of step-size parameters?

Ansewr :

$$\begin{aligned} Q_{n+1} &= Q_n + \alpha_n[R_n - Q_n] \\ &= \alpha_n R_n + (1-\alpha_n)Q_n \\ &= \alpha_n R_n + (1-\alpha_n)[\alpha_{n-1}R_{n-1} + (1-\alpha_{n-1})Q_{n-1}] \\ &= \alpha_n R_n + (1-\alpha_n)\cdot\alpha_{n-1}R_{n-1} + (1-\alpha_n)(1-\alpha_{n-1})Q_{n-1} \\ &= \dots \end{aligned} \tag{3}$$

$$Q_{n+1} = Q_1 \cdot \prod_1^n (1-\alpha_i) + \sum_i^n \alpha_i \prod_{j=i+1}^n (1-\alpha_j) R_i$$

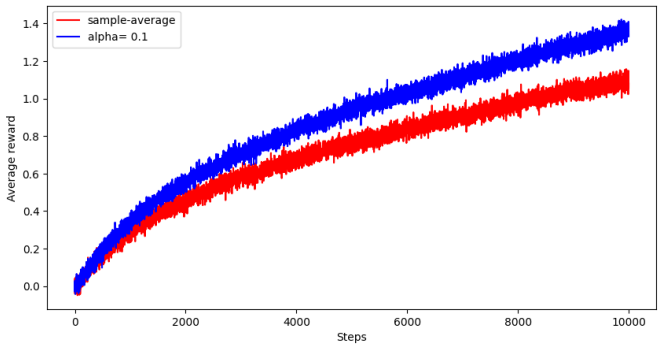
Exercise 2.5 (*programming*)

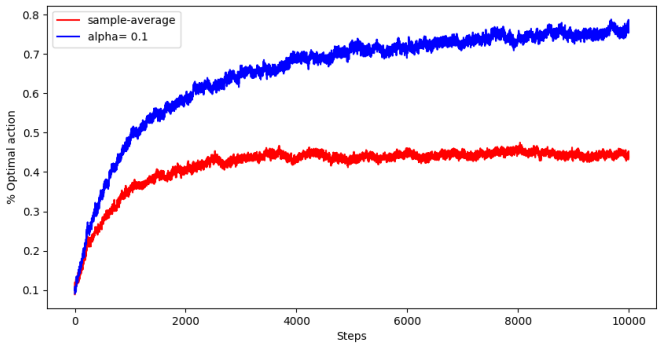
Design and conduct an experiment to demonstrate the difficulties that sample-average method have for nonstationary problems. Use a modified version of the 10-armed testbed in which all the $q_*(a)$ start out equal and then take independent random walks(say by adding normally distributed increment with mean 0 and standard deviation 0.01 to all the $q_*(a)$ on each step). Prepare plot like Figure 2.2 for an action-value method using sample averages, incrementally computed, and another action-value method using a constant step-szie parametr $\alpha = 0.1$, use

$\epsilon = 0.1$ and longer runs, say of 10,000 steps

Answer : see chapter_2.ipynb.

(non-stationary)





Excercise 2.6

Mysterious Spikes The results shown in Figure 2.3 should be quite reliable because they are averages over 2000 individual, randomly chosen 10-armed bandit tasks.Why, then, are there oscillations and spikes in the early part of the curve for the optimistic method? In other words, what might make this method perform particularly better or worse, on average, on particular early steps?

Anwser :

At the beginning of learning, all arms will be pulled because of the optimstic initialization. This will continue until the q value is corrected and becomes similar to the actual value, which can receive more rewards on average by doing a lot of exploration initially compared to the epsilon greedy method.However, after

all values have been modified to some extent, it will be biased toward one action because it is fundamentally a greedy method, which can cause spikes and ociliation

This method can be effective for bandit problems with low variance. Conversely, bandit problems with high variance can be a problem, and in this case, using it in combination with the e-greedy method can be a solution.

Excercise 2.7

Unbiased Constant-Step-Size Trick In most of this chapter we have used sample averages to estimate action values because sample averages do not produce the initial bias that constant step sizes do (see the analysis leading to (2.6)). However, sample averages are not a completely satisfactory solution because they may perform poorly on nonstationary problems. Is it possible to avoid the bias of constant step sizes while retaining their advantages on nonstationary problems? One way is to use a step size of $\beta_n = \alpha o_{n-1}^{-} + \alpha(1 - o_{n-1}^{-})$, for $n > 0$ with $\bar{o_0} = 0$ Carry out an analysis like that in (2.6) to show that Qn is an exponential recency-weighted average without initial bias.

Anwser : todo.

Excercise 2.8

UCB Spikes In Figure 2.4 the UCB algorithm shows a distinct spike in performance on the 11th step. Why is this? Note that for your answer to be fully satisfactory it must explain both why the reward increases on the 11th step and why it decreases on the subsequent steps. Hint: If c = 1, then the spike is less prominent

Anwser :

Initially, all arms will be selected. Therefore, the average reward will be similar to the average result of doing random action. However, from the 11th step after all arms are selected, policy will be biased to a specific arm rather than randomly pulling all arms, and there is a high probability that they are not optimal. After many steps are performed, the upper bound will gradually decrease and become similar to the actual value, and it is possible to witness an increase in the learning graph.

Excercise 2.9

Excercise 2.10

Excercise 2.11

```
"Hello"
```

```
## [1] "Hello"
```