Wrap-up

모두의연구소

Bandits for Recommender System

BRS LAB

MAB 문제 정의 (1)

Week 3

2021. 06. 12.

Introduction

❑ Learning method in RL

- **RL** : Use training **information that evaluates the actions taken**
  - Evaluative feedback : 해당 액션이 얼마나 좋은지를 의미
  - 해당 액션이 Worst인지 Best인지는 알 수 없음
- **SL** : Use training **instructive information by giving correct actions**
  - Instructive feedback : 정답인 액션을 의미
  - 실제 수행한 액션과는 독립적임

→ The **need for active exploration**, for an explicit search for good behavior

❑ Bandit problem

- The evaluative aspect of **reinforcement learning in a simplified setting**, one that does not involve learning to act in more than one situation
- This non-associative setting is the one in which most prior work involving evaluative feedback has been done, and **it avoids much of the complexity of the full reinforcement learning problem**

2.1. K-armed Bandit Problem

❑ K-armed bandit problem

- Slot machine K개의 레버를 당기는 문제
  - 반복되는 행동 선택을 통해 최고의 레버에 행동을 집중하여 상금을 극대화

- **Selecting a action** : k개의 서로 다른 옵션(또는 행동) 중 하나를 선택해야 함
- **Taking a reward** : 각 선택 후에는 선택한 행동에 따라 고정 확률 분포(Stationary)에서 선택한 보상(Numerical value)을 받음
- **Objective** : 일정 기간(Time step)에 걸쳐 예상되는 총 보상을 극대화하는 것

❑ Action value

- k개의 액션 각각의 그 액션이 선택되었을 때 예상되는 또는 평균 보상을 가짐
- $q_*(a) \doteq \mathbb{E}[R_t \mid A_t = a]$
  - t : time step
  - A_t : time step t에서 선택한 행동
  - R_t : 선택된 행동(A_t)에 대한 보상

2.1. K-armed Bandit Problem

❑ Action value (Cont.)

- 주어진 모든 행동의 가치($Q_*(a)$, Optimal action value)를 알고 있다면, k-armed bandit 문제를 해결하는 것은 간단함
  → 항상 가장 가치가 높은 작업을 선택

- 행동 값을 확실하게 알지 못한다고 가정
  - $Q_t(a)$: The estimated value of action *a* at time step *t*

- Greedy action (Exploitation)
  - 현재 Time step t에서 가지는 행동 가치 중 가장 큰 가치를 가지는 행동을 취하는 것

- Non-greedy action (Exploration)
  - Greedy action이 아닌 다른 행동을 취하는 것

- Exploitation/Exploration Trade-off
  - Exploitation : the **right thing to do to maximize the expected reward on the one step**
  - Exploration : the thing **may produce the greater total reward in the long run**

2.2. Action value methods

❑ Issue 1 : Estimating the values of actions

- Sample-average method

$$Q_t(a) \doteq \frac{\text{sum of rewards when } a \text{ taken prior to } t}{\text{number of times } a \text{ taken prior to } t} = \frac{\sum_{i=1}^{t-1} R_i \cdot \mathbb{1}_{A_i=a}}{\sum_{i=1}^{t-1} \mathbb{1}_{A_i=a}}$$

  - 실제로 받은 보상에 대한 평균
  - $\mathbb{1}_{\text{predicate}}$: 참이면 1, 거짓이면 0 인 랜덤 변수
  - 분모가 0인 경우 : 0으로 정의
  - 분모가 무한대로 갈수록(샘플이 많을 수록) Qt (a)는 q * (a)로 수렴

❑ Issue 2 : Action selection

- Greedy action selection
  - $A_t \doteq \underset{a}{\arg\max}\, Q_t(a)$

- Epsilon-greedy action selection
  - $A_t \doteq \underset{a}{\arg\max}\, Q_t(a) \qquad (1-\varepsilon)$
  - Select an action randomly (ε)

2.3. The 10-armed Testbed

❑ Greedy vs. Epsilon-greedy

- 10-armed testbed
  - 10-armed bandit
  - 2000번 무작위 추출
  - $q_*(a) \sim N(0,1)$ $(a = 1, . . . , 10)$
  - $R_t(a) \sim N(q_*(a), 1)$

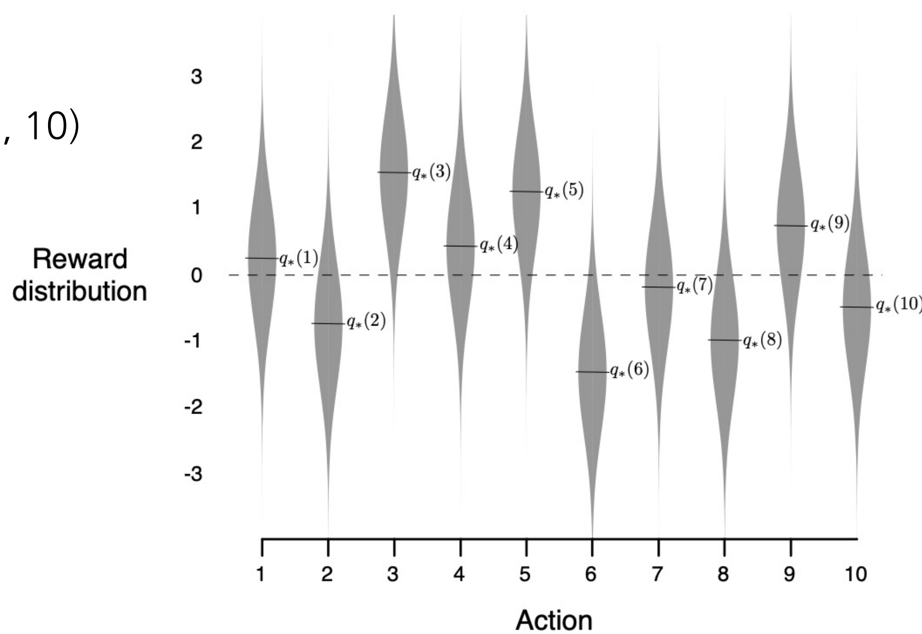


Figure 2.1: An example bandit problem from the 10-armed testbed. The true value $q_*(a)$ of each of the ten actions was selected according to a normal distribution with mean zero and unit variance, and then the actual rewards were selected according to a mean $q_*(a)$ unit variance normal distribution, as suggested by these gray distributions.

2.3. The 10-armed Testbed

❑ Experimental result (Greedy vs. Epsilon-greedy)

- Greedy action selection, Epsilon-greedy action selection (ε = 0.01, ε = 0.1)
  - Estimating action-values using Sample-average method

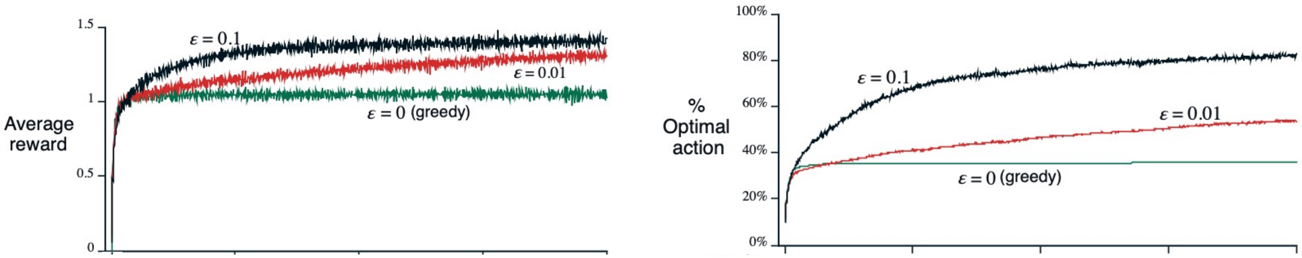
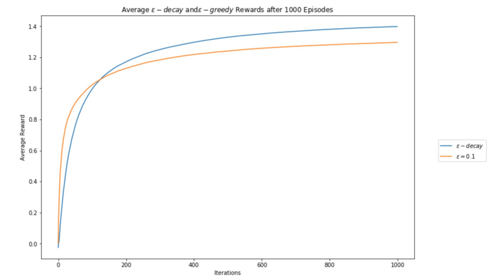


Figure 2.2: Average performance of ε-greedy action-value methods on the 10-armed testbed. These data are averages over 2000 runs with different bandit problems. All methods used sample averages as their action-value estimates.

- Greedy method는 처음에는 다른 방법보다 약간 빠르게 향상되었지만 그 다음에는 낮은 수준에서 수렴
  - 차선책을 수행하는 데 멈춰 있기 때문에 장기적으로 더 좋지 않음

- ε- Greedy method는 계속해서 탐색하고 새로운 인식 가능성을 높이기 때문에 결국 더 나은 성과를 보임
  - ε = 0.1 방법은 더 많이 탐색했고 일반적으로 최적의 동작을 일찍 찾았지만, 91 % 이상 해당 동작을 선택하지 않음
  - ε = 0.01 방법은 더 느리게 개선되었지만 결국 그림에 표시된 두 성능 측정 모두에서 ε = 0.1 방법보다 더 잘 수행됨
  - 시간이 지남에 따라 ε을 줄여서 높은 값과 낮은 값을 모두 얻을 수 있음



2.3. The 10-armed Testbed

❑ Advantages of ε-Greedy over greedy method

- ε-greedy가 가지는 장점은 task에 따라 다름
- Large reward variance (Noiser rewards)
  - 최적의 행동을 찾기 위해 더 많은 탐색이 필요하며, 이 경우 ε-greedy 방법은 greedy 방법에 비해 훨씬 더 잘 동작함
- Zero reward variance
  - Greedy mothod 또한 한 번 시도 후 각 각 행동 가치를 알 수 있음
    → Greedy가 Best 일수도
- In case of non-stationary task

2.4. Incremental Implementation

❑ Sample-average

- 보상의 샘플 평균으로 행동 가치를 추정

$$Q_n \doteq \frac{R_1 + R_2 + \cdots + R_{n-1}}{n-1}.$$

❑ Incremental mean (in computationally efficient manner)

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

A simple bandit algorithm

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

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

$$NewEstimate \leftarrow OldEstimate + StepSize\ \Big[Target - OldEstimate\Big].$$

with constant memory and constant per-time-step computation

2.5. Tracking a Nonstationary Problem

❑ Nonstationary problem

- 시간에 따라 보상의 확률 분포가 변화하는 문제

❑ More weight to recent rewards

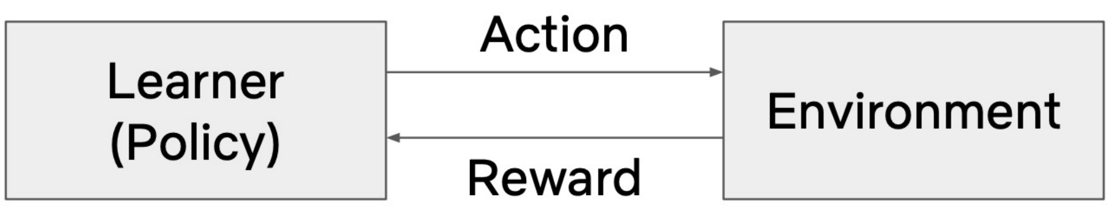
- 오래 전의 보상보다 최근 보상에 더 많은 비중을 두는 것이 합리적임
  → Non-staionary인 경우, 변경된 latest Reward distribution에 대한 weight를 높이는 것이 더 합리적
- n-1 과거 보상의 평균 Q_n을 일정한(constant) 단계 크기 매개 변수 α ∈ (0, 1]로 수정되어 표현

$$
\begin{aligned}
Q_{n+1} &= Q_n + \alpha\Big[R_n - Q_n\Big] \\
&= \alpha R_n + (1-\alpha)Q_n \\
&= \alpha R_n + (1-\alpha)\left[\alpha R_{n-1} + (1-\alpha)Q_{n-1}\right] \\
&= \alpha R_n + (1-\alpha)\alpha R_{n-1} + (1-\alpha)^2 Q_{n-1} \\
&= \alpha R_n + (1-\alpha)\alpha R_{n-1} + (1-\alpha)^2 \alpha R_{n-2} + \\
&\qquad\qquad \cdots + (1-\alpha)^{n-1}\alpha R_1 + (1-\alpha)^n Q_1 \\
&= (1-\alpha)^n Q_1 + \sum_{i=1}^{n} \alpha(1-\alpha)^{n-i} R_i .
\end{aligned}
$$

- $\sum_{n=1}^{\infty} \alpha_n(a) = \infty$: α가 초기 조건이나 변동을 극복할 수 있을만큼 충분히 큰지 확인
- $\sum_{n=1}^{\infty} \alpha_n^2(a) < \infty.$: 궁극적으로 수렴을 보장 할만큼 충분히 작아지는 것을 보장

Define rec-sys problem with MAB setting

Bandit Algorithms Setting



Each round:

- Learner chooses an **action**
- Environment provides a real-valued **reward** for action
- Learner updates to **maximize the cumulative reward**

Define rec-sys problem with MAB setting