

9. 탐색과 이용(Exploration and Exploitation)

김호철

Contents

1 소개

2 다중 슬롯 머신(Multi-Armed Bandit)

- 2.1 Regret(후회)
- 2.2 Greedy와 ϵ -Greedy 알고리즘들
- 2.3 하한(Lower Bound)
- 2.4 신뢰 상한선(Upper Confidence Bound)
- 2.5 베이지안 밴딧(Bayesian Bandits)
- 2.6 정보 상태 공간(Information State Search)

3 결론(Conclusion)

1 소개

• 탐색과 이용 딜레마

- 온라인 의사 결정에는 기본적으로 2개의 선택이 있음
 - * 이용(Exploitation): 최신 정보를 바탕으로 최선의 결정
 - * 탐색(Exploration): 더 많은 정보를 수집
- 최선의 장기 전략에는 단기적인 희생이 포함될 수 있다.
- 전반적으로 최선의 결정을 내리기에 충분한 정보를 수집

• 탐색과 이용의 예

- 식당 선택
 - * 이용 : 자주가는 식당으로 가기
 - * 탐색 : 새로운 식당 시도
- 온라인 배너 광고
 - * 이용 : 가장 클릭수가 많았던 광고 보여주기
 - * 탐색 : 새로운 실험적 광고 보여주기
- 석유 시추
 - * 이용 : 잘 알려진 위치에서 시추

- * 탐색 : 새로운 위치에서 시추
- 게임 플레이
 - * 이용 : 가장 좋다고 알고 있는 동작 수행
 - * 탐색 : 새로운 실험적 동작 수행

• 원칙(Principles)

- 나이브 탐색(Naive Exploration) : ϵ -greedy같은 탐욕 정책에 노이즈를 추가
- 초기화 낙관(Optimistic Initialisation) : 다르게 입증 될 때까지 현재가 최고라고 가정
- 불확실 낙관(Optimism in the Face of Uncertainty) : 불확실한 value들을 가진 액션을 선호
- 확률 매칭(Probability Matching) : 더 좋은 확률에 따라 행동을 선택
- 정보 상태 서치(Information State Search) : 정보의 가치에 따라 예측적 검색

2 다중 슬롯 머신(Multi-Armed Bandit)

• 다중 슬롯 머신

- 다중 슬롯 머신은 $\langle \mathcal{A}, \mathcal{R} \rangle$ 로 된 튜플이다(매우 단순한 MDP, 상태가 없음)
- \mathcal{A} 는 m개의 알고 있는 액션(또는 "arm")들의 집합
- $\mathcal{R}^a(r) = \mathbb{P}[r|a]$ 는 보상에 대한 알 수 없는 확률 분포
- 각 단계 t 에서 에이전트는 액션 $a_t \in \mathcal{A}$ 을 선택한다.
- 환경은 보상 $r_t \sim \mathcal{R}^{a_t}$ 를 생성한다.
- 목표는 누적 보상 $\sum_{T=1}^t r_T$ 을 최대화 하는 것이다.

2.1 Regret(후회)

• Regret(후회)

- 액션-벨류(action-value)는 액션 a 에 대한 평균 보상이다.

$$Q(a) = \mathbb{E}[r|a] \quad (1)$$

- 최적 벨류 V^* 는(할 수 있는 액션중에 기대값이 제일 높은 값),

$$V^* = Q(a^*) = \max_{a \in \mathcal{A}} Q(a) \quad (2)$$

- regret은 한 단계의 기회 손실(최적 보상과 새로운 시도의 보상이 더 안좋았을 경우에 보상의 차)

$$l_t = \mathbb{E}[V^* - Q(a_t)] \quad (3)$$

- 전체 regret은 전체 기회 손실의 합이다.

$$L_t = \mathbb{E} \left[\sum_{T=1}^t V^* - Q(a_T) \right] \quad (4)$$

- 최대 누적 보상 = 최소 전체 regret

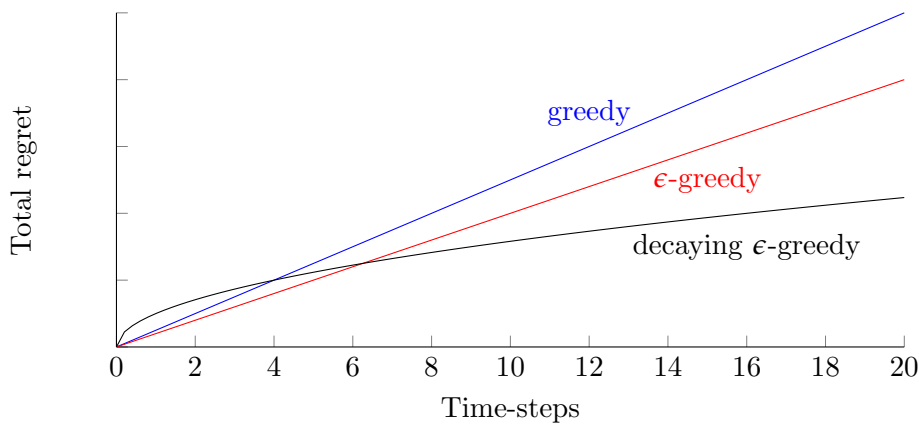
• Regret 카운팅

- 카운트 $N_t(a)$ 는 액션 a 의 선택이 기대되는 수이다.
- 간격(gap) Δa 는 동작 a 와 최적 동작 a^* 사이의 값 차이이다.

$$\Delta a = V^* - Q(a) \quad (5)$$

- Regret은 간격과 카운트에 대한 함수이다.
- 좋은 알고리즘은 큰 간격에서는 카운트가 작다.
- 문제는 간격을 모른다는 것이다.

• 선형이나 준선형 Regret



- 알고리즘이 영원히 탐색하면, 선형적인 regret이 될 것이다.
- 알고리즘이 전혀 탐색하지 않으면, 선형적인 regret이 될 것이다.
- 준선형(Sublinear)적인 regret이 가능할까?

2.2 Greedy와 ϵ -Greedy 알고리즘들

• Greedy 알고리즘

- $\hat{Q}_t(a) \approx Q(a)$ 를 추정하는 알고리즘을 고려

- 몬테카를로 평가로 각 액션들의 벨류(value)를 추정

$$\hat{Q}_t(a) = \frac{1}{N_t(a)} \sum_{i=1}^T r_i 1(a_i = a) \quad (6)$$

- 그리디 알고리즘은 가장 높은 값을 가진 액션을 선택한다.

$$a_t^* = \operatorname{argmax}_{a \in \mathcal{A}} \hat{Q}_t(a) \quad (7)$$

- 그리디 알고리즘은 차선최적(suboptimal) 액션에 영원히 빠질 수 있다.
- \Rightarrow 그리디는 선형적 regret을 가진다.

• ϵ -Greedy 알고리즘

- ϵ -greedy 알고리즘은 영원히 탐색한다.
 - * $1 - \epsilon$ 확률로 $a = \operatorname{argmax}_{a \in \mathcal{A}} \hat{Q}(a)$ 를 선택
 - * ϵ 확률로 무작위 액션 선택
- 상수 ϵ 은 최소 regret을 보장

$$l_t \geq \frac{\epsilon}{\mathcal{A}} \sum_{a \in \mathcal{A}} \Delta_a \quad (8)$$

- $\Rightarrow \epsilon$ -greedy는 선형적 regret을 가진다.

• 초기화 낙관(Optimistic Initialisation)

- 간단하고 실용적인 아이디어 : $Q(a)$ 를 높은 값으로 초기화
- 모든 value들을 최대값(maximum)으로 초기화
- 점진적 몬테카를로 평가를 통해 액션 벨류 업데이트
- $N(a) > 0$ 으로 시작

$$\hat{Q}_t(a_t) = \hat{Q}_{t-1} + \frac{1}{N_t(a_t)}(r_t - \hat{Q}_{t-1}) \quad (9)$$

- 초기 단계에 탐색을 많이 하게 하려는 의도
- 그러나 여전히 차선(suboptimal) 액션에 고정될 수 있음
- \Rightarrow greedy + 초기화 낙관은 선형적 regret을 가진다.
- $\Rightarrow \epsilon$ -greedy + 초기화 낙관은 선형적 regret을 가진다.
- 실무에서는 충분히 좋은 성능을 제공

• 감쇠(Decaying) ϵ_t -Greedy 알고리즘

- $\epsilon_1, \epsilon_2, \dots$ 들에 대한 감쇠 스케줄 선택
- 다음 스케줄을 보면,

$$\begin{aligned}
 c &> 0 \\
 d &= \min_{a|\Delta_a > 0} \Delta_i \\
 \epsilon_t &= \min \left\{ 1, \frac{c|\mathcal{A}|}{d^2 t} \right\}
 \end{aligned} \tag{10}$$

- 감쇠 ϵ_t -Greedy는 로그적 regret을 가진다.
- 불행히도 스케줄에는 간격들에 대한 사전 지식이 필요하다.
- 목표 : 어떤 다중 슬릿 머신에도 적용 가능한 준선형(sublinear-선형에 가까운) regret이 있는 알고리즘 찾기(보상에 대한 지식없이)

2.3 하한(Lower Bound)

• 하한(Lower Bound)

- 모든 알고리즘의 성능은 최적 머신과 다른 머신들 사이의 유사도에 의해 결정된다.
- 확률 분포가 유사한 머신들일수록 문제는 어려워진다.
- 이것은 간격 Δ_a 와 분포의 유사성 $KL(\mathcal{R}^a || \mathcal{R}^{a*})$ 에 의해 설명된다

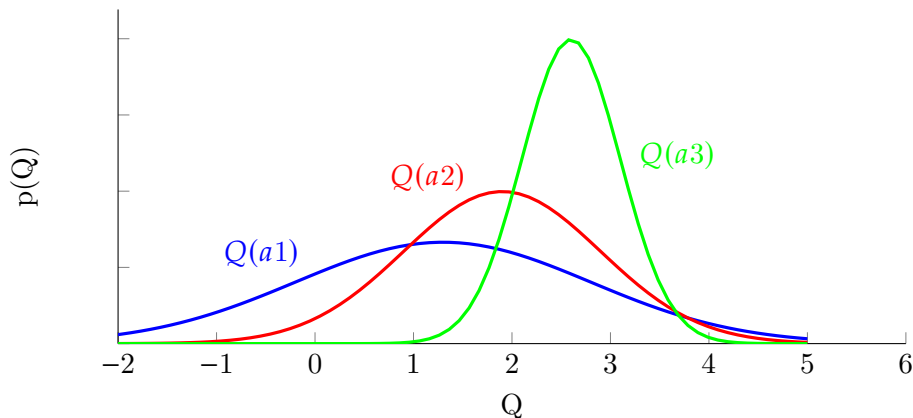
라이와 로빈스 정리

점근적(점점 가까워지는)인 총 regret은 무한 단계를 수행해도 로그보다 좋을 수 없다.

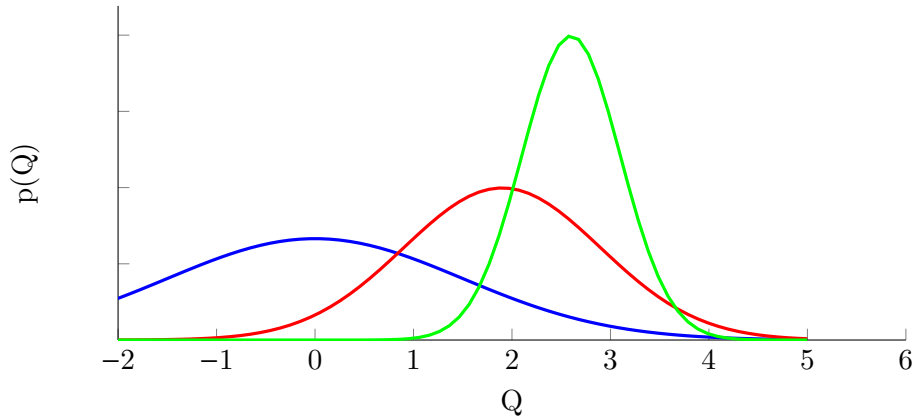
$$\lim_{t \rightarrow \infty} L_t \geq \log t \sum_{(a|\Delta_a > 0)} \frac{\Delta_a}{KL(\mathcal{R}^a || \mathcal{R}^{a*})}$$

2.4 신뢰 상한선(Upper Confidence Bound)

• 불확실 긍정(Optimism in the Face of Uncertainty)



- 어떤 액션을 선택해야 할까?
- 액션-벨류에 대해 불확실할수록, 그 액션을 탐색하는 것이 더 중요하다.
- 최적의 액션이 될 수 있기 때문이다.



- 파란 액션을 선택하면,
- 벨류에 대한 불확실성이 감소한다.
- 그리고 다른 액션을 선택할 가능성이 더 높음
- 최선의 액션을 취할 때까지

• 신뢰 상한선(Upper Confidence Bounds)

- 각 액션-벨류(action value)에 대하여 신뢰 상한 $\hat{U}_t(a)$ 을 추정 - 신뢰 구간의 상한(예 95% 상한)
- $Q(a) \leq \hat{Q}_t(a) + \hat{U}_t(a)$ 가 높은 확률로,
- 선택된 횟수(머신을 당긴 횟수) $N(a)$ 에 종속적이다.
 - * 작은 $N_t(a)$ (머신을 적게 당길수록) \Rightarrow 큰 $U_t(a)$ (추정된 value는 부정확하다) - 신뢰구간이 넓다
 - * 큰 $N_t(a)$ (머신을 많이 당길수록) \Rightarrow 작은 $U_t(a)$ (추정된 value는 정확하다) - 신뢰구간이 좁다
- 신뢰 상한선(UCB)을 최대로 하는 액션을 선택

$$a_t = \operatorname{argmax}_{a \in A} \hat{Q}_t(a) + \hat{U}_t(a) \quad (11)$$

• 호프딩 부등식(Hoeffding's Inequality)

- 모분포를 몰라도 UCB를 적용하기 위한 부등식

호프딩 부등식 정리

X_1, \dots, X_t 가 0과 1사이의 무작위 변수이고, $\bar{X}_t = \frac{1}{t} \sum_{i=1}^t X_i$ 이 샘플 평균이면,

$$\mathbb{P}[\mathbb{E}[X] > \bar{X}_t + u] \leq e^{-2tu^2}$$

($\mathbb{E}[X]$:실제 X의 모평균, u :틀린 정도)

- 슬롯 머신의 보상에 호프딩 부등식을 적용하면,
- 액션 a 를 선택하는 조건은($Q(a)$:기대값, $\hat{Q}_t(a)$:샘플 평균, $U_t(a)$:UCB),

$$\mathbb{P}[Q(a) > \hat{Q}_t(a) + U_t(a)] \leq e^{-2N_t(a)U_t(a)^2} \quad (12)$$

• UCB 계산하기

- 트루 벨류(true value)가 UCB를 넘길 확률 p 를 선택
- $Q(a) \leq \hat{Q}_t(a) + \hat{U}_t(a)$ 가 높은 확률로,
- $U_t(a)$ 를 풀면,

$$e^{-2N_t(a)U_t(a)^2} = p$$
$$U_t(a) = \sqrt{\frac{-\log p}{2N_t(a)}} \quad (13)$$

- 더 많은 보상을 위해 p 를 $p = t^{-4}$ 처럼 줄일 수 있다.
- 최적의 액션 선택을 위해 $t \rightarrow \infty$ 로 한다.

$$U_t(a) = \sqrt{\frac{2 \log t}{N_t(a)}} \quad (14)$$

• UCB1

- 다음이 UCB1 알고리즘이다.

$$a_t = \operatorname{argmax}_{a \in A} \hat{Q}_t(a) + \sqrt{\frac{2 \log t}{N_t(a)}} \quad (15)$$

정리

UCB 알고리즘은 로그 점근적 총 regret을 달성한다.

$$\lim_{t \rightarrow \infty} L_t \leq 8 \log t \sum_{a | \Delta_a > 0} \Delta_a$$

2.5 베이저안 밴딧(Bayesian Bandits)

• 베이저안 밴딧

- 지금까지 보상 분산 R 에 대해서는 가정하지 않았다.
 - * 보상의 제한(bound)은 예외
- **베이저안 밴딧**은 보상에 대한 사전 지식($p[R]$)을 이용한다.
- 실제 보상 $p[R|h_t]$ 의 사후 분포를 계산
 - * 여기서 $h_t = a_1, r_1, \dots, a_{t-1}, r_{t-1}$ 은 히스토리
- 탐색을 가이드하기 위해 이 사후를 이용
 - * 신뢰 상한선(베이저안 UCB)
 - * 확률 매칭(툼슨 샘플링)
- 사전 지식이 정확하면 성능이 향상

• 확률 매칭(Probability Matching)

- **확률 매칭**은 액션 a 가 최적의 액션일 확률에 따라 액션 a 를 선택

$$\pi(a|h_t) = \mathbf{P}[Q(a) > Q(a'), \forall a' \neq a|h_t] \quad (16)$$

- 확률 매칭은 불확실성에 대해 낙관적(optimistic)이다.
 - * 불확실한 액션은 최대가 될 확률이 더 높다
- 사후(posterior)로부터 분석적 계산은 어려워질 수 있다.

• 톼슨 샘플링(Thompson Sampling)

- **톤슨 샘플링**은 확률 매칭을 구현하였다.

$$\begin{aligned} \pi(a|h_t) &= \mathbf{P}[Q(a) > Q(a'), \forall a' \neq a|h_t] \\ &= \mathbf{E}_{R|h_t} \left[1(a = \operatorname{argmax}_{a \in A} Q(a)) \right] \end{aligned} \quad (17)$$

- 베이즈 법칙을 이용하여 사후 분포 $p[R|h_t]$ 를 계산한다.
- 사후에서 보상 분포 R 을 샘플링 한다.
- 액션-가치 함수(action-value function) $Q(a) = \mathbf{E}[R_a]$ 를 계산한다.
- 샘플에서 최대값의 액션을 선택 $a_t = \operatorname{argmax}_{a \in A} Q(a)$
- 톼슨 샘플링은 라이와 로빈스의 하한을 달성했다.

2.6 정보 상태 공간(Information State Search)

- 정보의 가치(Value)

- 탐색은 정보를 얻기 위해서는 유용하다.
- 정보의 가치(value)를 계량화(quantify) 할 수 있을까?
 - * 의사 결정을 내리기 전에 정보를 얻기 위해 의사 결정자가 지불 할 준비가 된 보상의 양
 - * 정보를 얻은 후에 장기 보상 - 즉시(immediate) 보상
- 불확실한 상황에서 정보 획득이 더 높다
- 따라서 불확실한 상황을 더 탐구하는 것이 합리적
- 정보의 가치(value)를 안다면 탐색(exploration)과 이용(exploitation)을 최적으로 절충할 수 있다.

- 정보 상태 공간

- 밴딧은 1-단계 의사 결정 문제이다. : A와 R만 있다.
- 또한 순서적 의사 결정 문제로 볼 수도 있다,
- 각 단계에는 정보 상태 \tilde{s} (틸다) 가 있다
 - * \tilde{s} 는 히스토리의 통계이다. $\tilde{s} = f(h_t)$
 - * 지금까지 누적된 모든 정보의 합
- 각 액션 a는 확률 $\tilde{P}_{\tilde{s}, \tilde{s}'}$ 와 함께 새로운 정보 상태 \tilde{s} (정보의 추가에 의한)로 전환 된다.
- 이것은 증강된(augmented) 정보 상태 공간에서 MDP \tilde{M} 를 정의 한다.

$$\tilde{M} = \langle \tilde{S}, A, \tilde{P}, R, \gamma \rangle \quad (18)$$

3 결론(Conclusion)

- 탐색/이용에 관한 몇가지 원칙이 발견되었다.
 - ϵ -그리디 같은 나이브(naive)한 방법
 - 초가화 낙관(Optimistic initialisation)
 - 신뢰 상한선(Upper confidence bounds)
 - 확률 매칭(Probability matching)
 - 정보 상태 서치(Information state search)
- 각 원칙은 밴딧 설정을 위해 개발되었지만,
- MDP설정에도 동일한 원칙이 적용된다.