[추천시스템] Multi-Armed Bandit

Info.

- 본 문서는 Multi-Armed Bandit에 대한 내용을 조사/정리한 내용임
- 목차
  - 참고자료
  - Multi-Armed Bandit (MAB)
  - 문제 정의
  - 알고리즘
  - 논의

작성완료

1. 참고자료
   a. https://imaddabbura.github.io/post/epsilon_greedy_algorithm/
   b. http://sanghyukchun.github.io/96/
   c. https://brunch.co.kr/@chris-song/62
2. Multi-Armed Bandit (MAB)
   a. Introduction
      i. 팔이 한개인 강도(one-armed bandit)가 카지노에서 슬롯머신을 당긴다고 가정해보자.
      ii. 횟수(시간)는 한정되어 있고 1번 시도시 1번의 결과만 확인할 수 있는데 reward를 극대화 할수 있는 전략은 무엇일까?

      iii.
      iv. source: towardsdatascience.com
      v. 마음에 드는 하나의 슬롯머신을 찍어서 그 슬롯머신에서만 H 횟수만큼 시도한다 → *하필 그 슬롯머신의 reward가 별로라면?*
      vi. 모든 슬롯머신을 동일한 횟수만큼 시도한다 → *가장 reward가 좋은 슬롯머신만 시도하는 게 효율적이지 않을까?*
      vii. 랜덤하게 몇 개의 슬롯머신만 시도한후 그중 가장 좋은 rewards를 보인 슬롯머신만 계속 당긴다?
   b. Exploration and Exploitation Trade-off or Dilemma
      i. Exploration → 탐색하고 결과를 확인하는 과정 (너무 부족하거나 과하면 좋은 전략이라 할수 없다)
      ii. Exploitation → reward를 취하는 과정
      iii. 한 곳에 치우치지 않도록 적당한 밸런스를 잡아가는 것이 중요 → MAB algorithm 역할

c. Advantage of MAB

i. A/B 테스트

1. 한 유저에게 하나의 시안을 보여줄 수 있음 → 안 좋은 시안을 본 유저(한정된 자원)는 단순히 소모됨
2. 테스트 기간에 할수 있는 것이 별로 없음. 결과를 기다려야 하니 비용/시간 소모가 발생됨

ii. 임상 실험

1. 환자에게 신약 처치 (k개의 약물을 n번시도)
2. 시간적 비용 및 리스크 높음

iii. 대안

1. *"Smoothly decrease exploration over time instead of sudden jumps"*
2. *"Focus resources on better options and not keep evaluating inferior options during the life of the experiment"*
3. Note
    a. 탐색 시작 전에는 확률 등 사전정보가 없으며, 직접 시도&에러를 통해 정보(feedback)를 얻는 과정을 반복함
    b. 피드백을 통해 the balance between exploration and exploitation을 찾아 최적의 profit을 얻는 것이 목적

3. 문제 정의

a. 슬롯머신의 Reward가 어떤 probabilistic distribution(e.g. binary distribution)을 따른다고 가정하고, H시간후의 reward를 maximize 하는 strategy(혹은 policy)를 구하는 것

i. 혹은 regret을 minimize하는 것으로 표현 가능

b. 무수히 많은 variant 가 있지만, Finite-armed stochastic bandit problem 으로 우선 정의

i. why stochastic? Assumption → Rewards are dependent on i.i.d (independent and identically distributed random variables)
ii. arm의 개수(k), payoff function(x), play time(H) → Finite

c. 수식(Regret Function)

$$R = \left(\max_{i=1,\dots,K} \mathbb{E} \sum_{t=1}^{H} x_{i,t} \right) - \mathbb{E} \sum_{t=1}^{H} x_{S_t,t}$$

i.
ii. optimal policy 대비 연구자의 policy로 인한 결과의 차이
iii. optimal policy 를 알수 없으므로 사전분포 (e.g. binary distribution)를 따른다고 가정하고 최적화 작업을 진행함

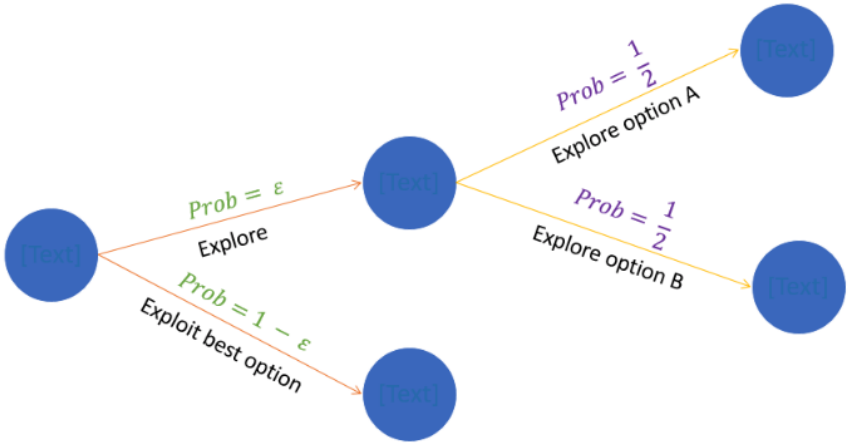
4. 알고리즘

a. Epsilon-greedy (ε-greedy)

i. 다른 알고리즘 대비 이론적으로 또는 실험적으로 우수하지 않으나 직관적임

- Assume we have a coin that has a probability of coming heads = ϵ and a probability of coming tails = 1 - ϵ. Therefore,
    - If it comes heads, explore randomly the available options (exploration).
    - The probability of selecting any option is $^1/_2$.
    - If it comes tails, select the best option (exploitation).

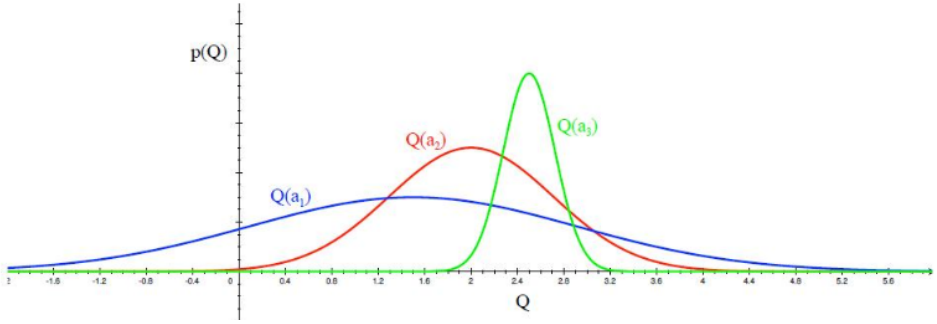
ii.



iii.
iv. Source: https://imaddabbura.github.io/post/epsilon_greedy_algorithm/
v. 한계점

1. 확률적으로 sub-optimal arm을 뽑고 uniform randomly하게 선택되므로 관측(테스트) 못한 arm 존재
2. global optimal에 도달했어도 계속 ε 확률로 탐색을 진행해야함.
3. 대안: ε 를 constant로 사용하는 대신 adaptive하게 update 혹은 일정 비율로 감소시킴

b. Upper Confidence Bound(UCB)
i. 시간(t) 마다 과거의 관측결과(empirical mean)과 관측횟수(N)를 고려해 구한 upper confidence bound(UCB)를 이용하는 알고리즘
ii. The more uncertain we are about an arm, the more important it becomes to explore that arm.



1.
iii. UCB1
1. 수식

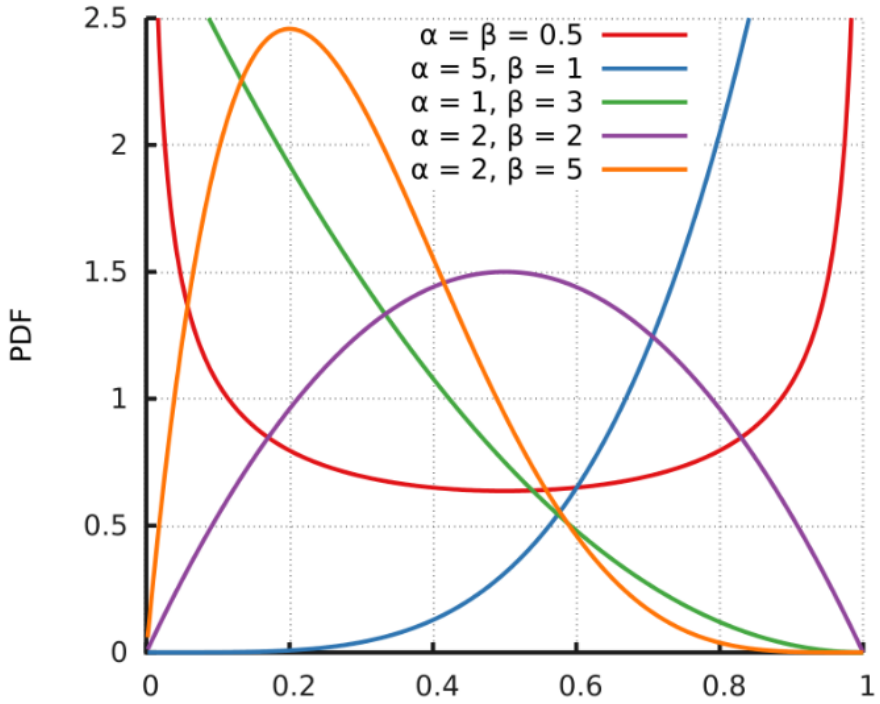
$$i = \arg\max_i \bar{x}_i + \sqrt{\frac{2\ln t}{n_i}}.$$

a.
2. Notation
a. X bar: i 번째 arm의 지금까지 관측한 평균치
b. t: 현재시간 (전체 round 횟수)

        c. n: 현재시간 중 arm i 가 재생된 횟수
    3. arm i 가 많이 시도될수록 불확실성이 감소함 (the uncertainty term decreases)
    4. 반대로 t 가 높아질수록 (with Ni constant) 불확실성 증가
    5. 단, t는 log scale, n은 linear
c. Thompson Sampling
    i. Source: https://brunch.co.kr/@chris-song/66
    ii. 베타분포

$$Beta(x|a,b) = \frac{1}{B(a,b)} x^{a-1}(1-x)^{b-1}$$

1. 베타 분포



알파와 베타 값에 따른 베타 분포 출처: 위키피디아

2.
3. Source: https://en.wikipedia.org/wiki/Beta_distribution

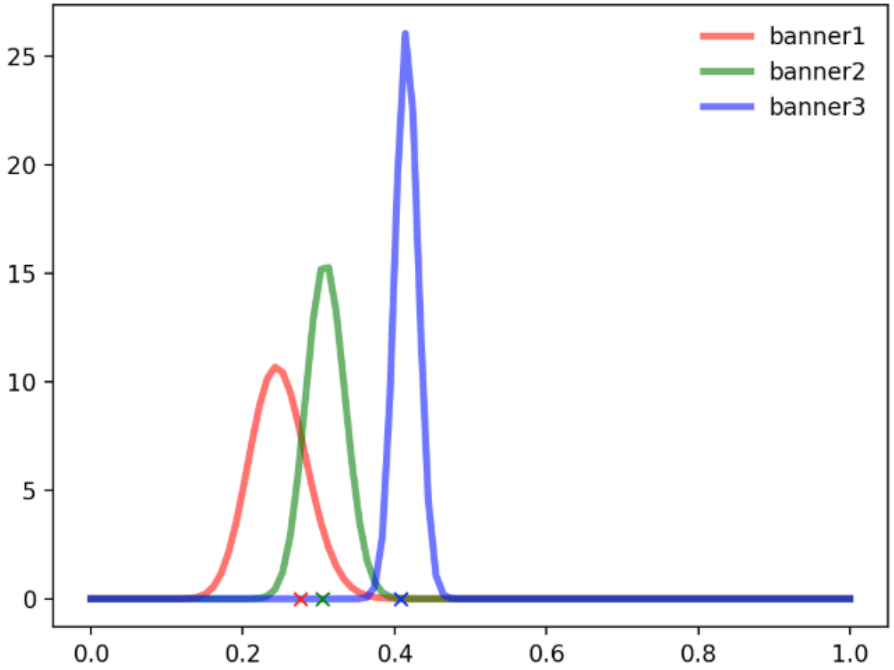
    iii. 베타함수

$$B(a,b) = \int_0^1 x^{a-1}(1-x)^{b-1}dx$$

베타 함수의 정의

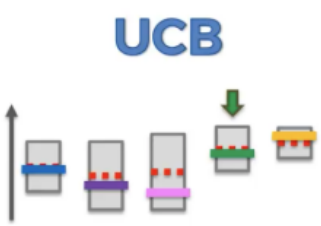
1.

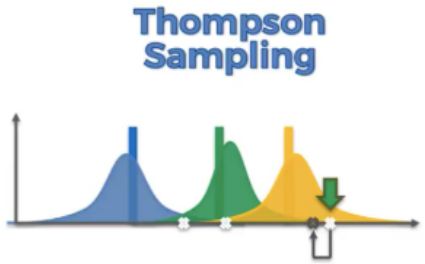
iv. 배너 클릭률 예측 예시
    1. 배너 후보 → 3개 (A,B,C)
    2. Parameter for Beta function:
        a. Beta(배너 클릭 횟수 +1, 배너클릭하지 않은 횟수 +1)
        b. https://brunch.co.kr/@chris-song/66



c.

v. UCV VS Thompson Sampling

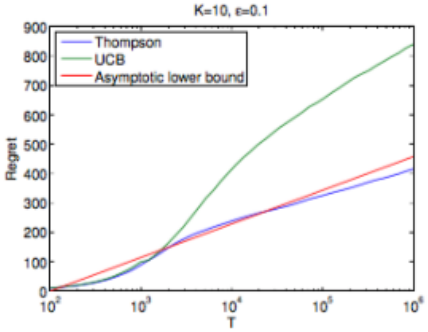


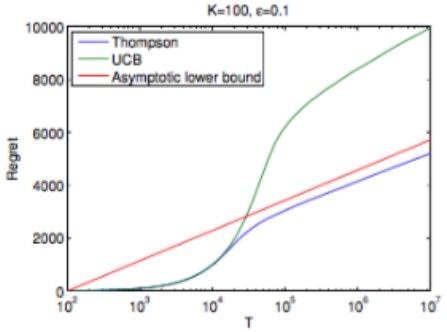


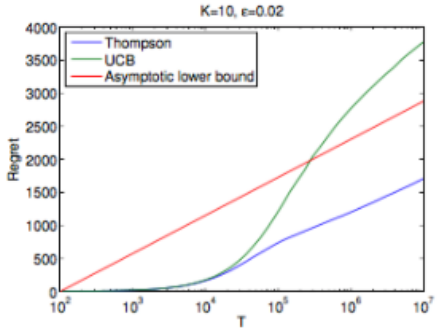
- Deterministic
- Requires update at every round

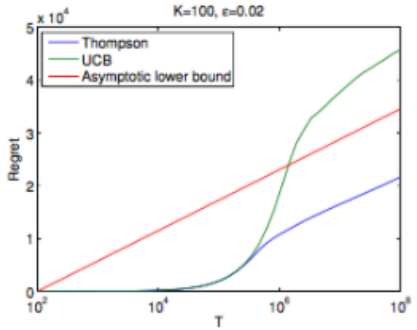
- Probabilistic
- Can accommodate delayed feedback
- Better empirical evidence

1.



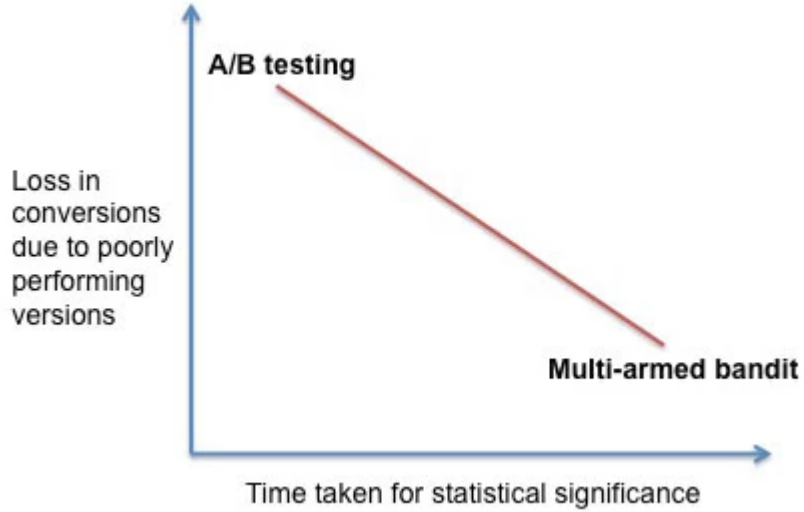






2.
3. Source: http://papers.nips.cc/paper/4321-an-empirical-evaluation-of-thompson-sampling.pdf

5. 논의
   a. Q) Is MAB really a Panacea?
   b. A) A/B testing is meant for strict experiments where focus is on statistical significance,
      whereas MAB algorithms are meant for continuous optimization where focus is on maintaining higher average conversion rate.



c.
d. Source: https://vwo.com/blog/multi-armed-bandit-algorithm/