MAB

Dane.cb

What is MAB?

- **A/B 테스트** 시, 각 버킷의 지표를 어떻게 비교하여 **가장 좋은 버킷을 선택**할 수 있을까?
  - 각 버킷을 Armed bandit으로 해석. 어느 bandit(bucket)의 arm을 내릴 것인가?
  - 다양한 KPI가 존재하겠지만, 오늘의 논의에서는 CTR, CVR과 같은 비율지표를 전재하겠음
  - 꼭 버킷에만 쓰이는 것은 아니다 …
    현재 카카오에서는, A/B 테스트 이외의 목적으로 MAB를 사용하고 있다(후술)
- MAB는 일종의 시스템으로, 해결할 방법론은 다양하다
  - 단순 비교 (임의 명칭)
  - Epsilon-greedy
  - Thompson sampling

Exploitation vs Exploation

- 그 전에, 알아두고 가자
- **Exploitation**: 기존에 좋은 결과를 선택해서 활용하자 -> 좋은 놈만 기회를 부여한다
- **Exploration**: 뭐가 좋은지 좀 더 실험해보자 -> 고루 기회를 부여한다

- MAB의 핵심: Exploration에 낭비를 최대한 줄이자!

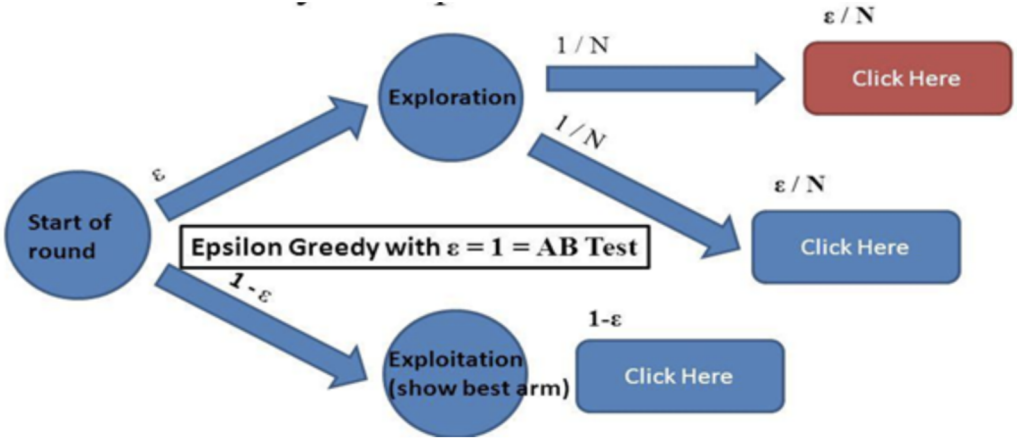
단순 비교

- 다른 특별한 알고리즘 없이, **누적된 데이터를 바탕으로 버킷을 평가**하는 것
- 현재 **추천팀에서 일반적으로 사용**하는 A/B테스트 방법론
- 가장 **단순**한 만큼 **아무런 전제 조건도 없다**
- 개인적으로도 가장 선호하는 방법이다

- 왜 단순한 만큼 강력한지 차차 비교를 통해 설명을 보강해가겠다

Epsilon-greedy

- 전체 결과 중 일부는 exploration에, 나머지는 exploitation에 사용한다
  - 예컨대 전체 중 80%는 지금까지 가장 성적이 좋았던 버킷의 결과로
  - 20%는 전체 버킷 중 랜덤으로
  - 일반적으로 exploitation에 큰 비중을 준다
- 특별히 어렵지 않은 알고리즘이지만, 빠르게 exploration을 줄여준다
  - 또한 지속적으로 exploration에 노출을 할당함으로써 일시적으로 튀는 버킷을 보정해줄 수 있다



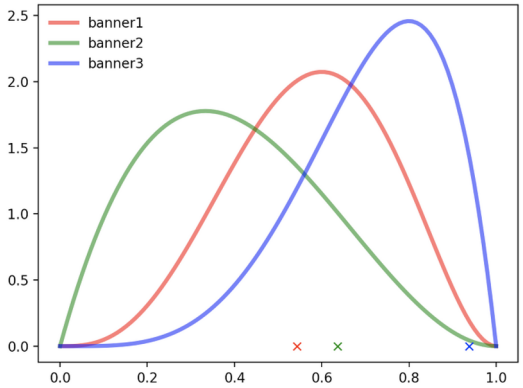
Epsilon-greedy

- 응용예시

- 주의사항
  - 단순히 구좌 위치로 한다면 위치의 영향을 받을 가능성이 있다 -> 매번 셔플
  - 예컨대 우측하단은 클릭률이 다른 곳보다 높을 수 있다
  - 엡실론의 값은 일반적으로 휴리스틱하게 정하는 것 같다
    - adaptive하게 정해나가는 방법론들도 많이 있따 -> 강화학습

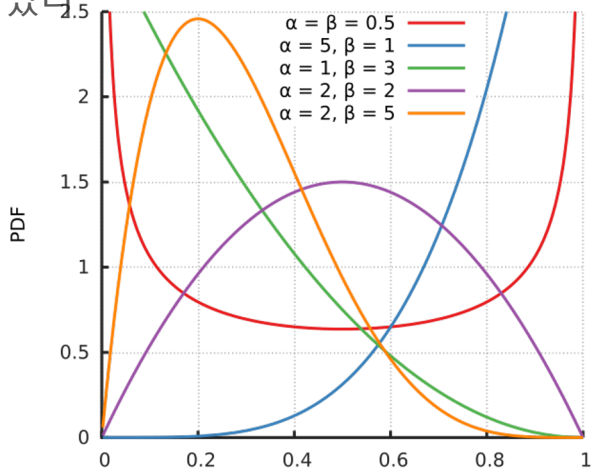
Thompson sampling

- Bayesian framework에 근거해 CTR과 같은 변수를 확률 변수로 바라봄 (이하 r.v.)
  - 위에서 전제했듯 여기서는 CTR이나 CVR 등의 비율 지표를 볼 것임
  - 따라서 우리가 궁금한 버킷별 CTR, 즉 p를 r.v로 바라본다: p ~ beta(a, b)
  - 이때 beta 분포를 사용하는 것은 conjugate같은 이론적 배경이 있지만 생략
  - Q: 비율이 아닌 지표를 보면 어떻게 해야하나? -> normal등으로 보는 것 같다

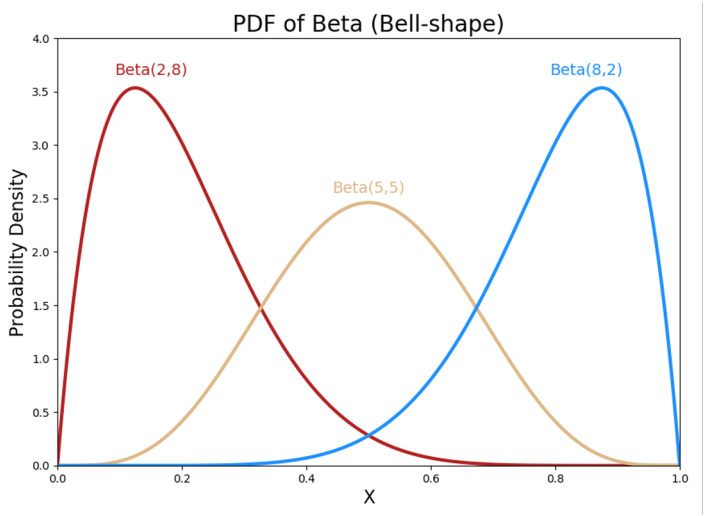


Thompson sampling

- Beta distribution
  - alpha와 beta라는 두 개의 모수를 가지고 있다 -> 모수: 분포의 형태를 결정짓는 수
  - prior: 실험이 이루어지기 전에 모수에 주는 값이라고 생각하자
  - non-informative prior: 실험에서 아무런 가정 없이 나중에 데이터만 따라가는 사전분포
    - beta의 경우에는 (alpha=1, beta=1)을 주로 사용함
    - prior를 조정함으로써 실험을 부분적으로 컨트롤할 수 있다.
    - 어떻게 컨트롤하는지 차차 살펴보자
  - alpha(click)가 커지면 p는 우측으로 이동한다
  - beta(unclick)가 커지면 p는 좌측으로 이동한다
  - mode(p) = alpha-1 / alpha+beta-2 =
    click / click+unclick = CTR(점추정)
  - alpha+beta = K = 노출수 = 신뢰도로 해석 가능



Thompson sampling



- 노출 분배는 어떻게 이루어질까?
- 각 버킷의 CTR에 대한 베타분포로부터 난수추출
  추출된 난수값이 가장 높은 arm을 노출시킨다
  - CTR ~ beta(alpha, beta)
  - alpha가 크면 난수추출된 값은 큰 경향이 있다
  - beta가 크면 난수추출된 값은 작은 경향이 있다
  - alpha와 beta의 값이 모두 커지면 난수추출은 CTR근처에서 멀리 가지 않는다 (low var)
    - 따라서 CTR이 큰 버킷이 거의 이긴다 (실험 후반)
  - alpha와 beta의 값이 모두 작으면 난수추출은 CTR에서 멀리까지도 나온다 (high variance)
    - 따라서 CTR이 작은 버킷도 자주 이긴다 (실험 초반)
  - 이러한 과정은 점진적으로 이루어지게 된다.

ED / TS의 단점 (뇌피셜)

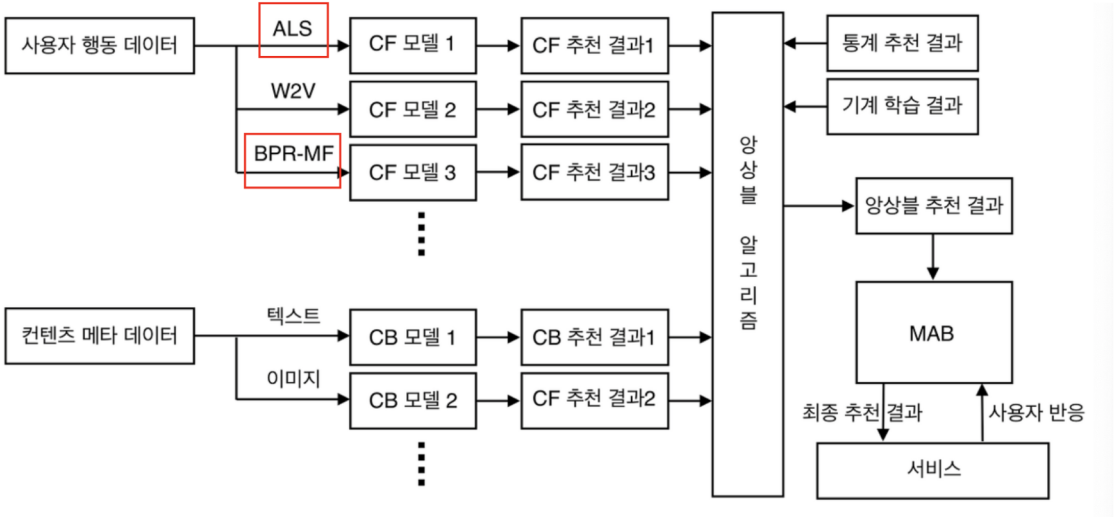
- 개인적으로 다양한 A/B 테스트들을 진행해오면서
  생각보다 95% CI는 부질없으며 많은 수의 샘플로 나온 0.01 p-value도 신뢰할 수 없다는 것을 관찰함
- Why???
  - 이전 실험의 영향을 받는다
  - 수요일에 배포하면 수요일의 패턴과 목요일의 패턴이 달라질 수 있다. 물론 주말도 밤낮도 다르다.
  - 실험을 진행함으로써 실험에 영향받은 유저들의 패턴이 달라질 수 있다 (그리고 버킷은 계속 셔플될 수 있다)
  - 동시에 두 개 이상의 실험을 진행하면 서로 상호작용할 수 있다
  - 실험의 미묘한 버그를 수정한다
  - 유저들마다 반응이 다르다(그리고 유저의 버킷은 셔플된다)
    - 하드 유저와 라이트 유저의 반응이 다를 수 있다
  - 아웃라이어가 존재한다
    - 크롤링이나 조작의 의도를 가진 봇이 특정 버킷에 돌고 있다면 끔찍하다

ED / TS의 단점 (뇌피셜)

- 즉 **CTR이라는 r.v.는 끊임없이 변동**할 수 있고
  - 이는 단기적인 대규모의 샘플로는 정확히 평가하기 어렵다고 생각한다
  - 심지어 장기적인 실험의 경우 유저나 서비스의 전반적인 성격이 달라질 수 있다 ....
- 그렇다면 ED/TS와 같이 **초기 exploration에 강하게 의존**하는 알고리즘은 …
  - 후반부에 쉽게 바뀌지 않을 수 있다
  - 특히 버킷별 차이가 엄청나게 뚜렷하지 않은 이상(그리고 보통 그럴 것 같다)
    초기의 결과가 지배적일 수 있다 (나중에 바뀌기 쉽지 않다)
  - decay도 뚜렷한 해결책은 아닐 수 있다
- 그리고 그 혜택 또한 뚜렷하지 않다
  - TS의 경우 adaptive하게 우월한 버킷에 수렴하는 것을 보장하려는 아이디어인데
  - 애초에 **버킷 간의 차이가 명백하면, 대부분의 경우 빠르게 실험을 종료**한다
  - 그리고 **버킷 간의 차이가 미묘하면 양쪽 버킷 모두에 많은 데이터**를 쌓아야 한다 ..

MAB: Ensemble

- 추천팀에서는 이런 MAB를 다소 다르게 사용하고 있다
- A/B테스트에서 버킷을 선택하는 용이 아니라
- 다양한 추천 결과를 앙상블하는 용으로 **MAB**를 사용하고 있다



Ensemble?

- 앙상블은 **여러 모델을 조합**하여 더 좋은 결과를 만들어내는 것
- ML에서는 다수의 week classifier를 조합해 strong classifier를 만들어내는 예시가 흔함 ex) Random forest, ~ boost
- 하지만 추천결과를 앙상블하는 것은 쉽지 않다 ……..
  - 추천결과의 weight가 다 다르기 때문에, 떨어진 결과는 rank로 이해할 수 있다.
  - GC에서 10등과 CF에서 10등이 가지는 의미가 다를 수 있다
  - 좀 더 정밀하게는 stacking이라는 방법론이 있지만, 이 경우 쓰이기가 힘들다
    - stacking은 각 모델에 대해 주어진 X에 따른 output(prob or number)들을 모은다
    - 이후 output1, output2, ...들을 가지고 새로운 output을 예측하는 형태인데
    - 추천 모델의 경우 모든 경우를 다 고려하기 어렵기 때문에
      결국 각 모델의 feature를 가져와야 한다
    - 근데 그러려면 모든 모델의 싱크가 맞아야하고 순차적으로 학습이 이루어져야 한다
  - 그렇지 않으면 rank를 기반으로 학습해야 하는데 이것도 실질적으로 상당히 어렵다
    - rank fusion을 사용하는 곳도 있긴 하다
  - 그래서 마지막에 여러 모델로부터 나온 **추천 결과 N개를, N개의 버킷**으로 보고 MAB를 돌린다

MAB Ensemble

- N개의 추천결과를 N개의 추천 결과로 보고 MAB를 돌린다는 것은
  - 각 아이템에 alpha와 beta를 학습하여 thompson sampling을 돌린다는 것
  - 이것은 사실 대량의 실시간 트래픽이 있어야만 가능한 모델
  - 트래픽이 모자라면 적절히 모든 아이템이 학습될 수 없다 …
- 근데 … 추천 풀은 매번 바뀌는데요?
  - 이 부분은 정확히 알지 못하고 있다 …
  - new arm의 경우 k가 낮아서 좀 더 잘 노출되는 것으로 파악 중
  - 근데 new arm이 많은 경우 늘 new arm만 노출되고 수렴하지 못하는 것이 아닌가 …
  - 뒤로 이어지겠습니다

MAB Ensemble

- Prior는 정말 1,1인가요?
    - prior beta이 낮아질수록 새로운 아이템이 잘 뽑히는데, 서비스마다 적절한 값이 다를 것이다
- 이에 대한 대안으로 moment method도 있다:
    - 기존의 노출/전환을 가지고 alpha와 beta를 업데이트 시켜놓음
    - prior dominent를 조심 -> alpha + beta를 제한함 (variance이용)
- 잘 돌아간 사례도 있긴 하지만 MM에 대한 개인적인 걱정으로는
    - 클러스터는 새로 학습할 때마다 완전히 바뀔 텐데
      이때 기존에 여러 클러스터에 나눠진 노출/전환을 어떻게 적용할 것인가?

MAB Ensemble

- Beta는 unclick인가요?
  - 대부분의 경우 MAB는 **문어발**이다
  - 따라서 10개 노출해서 A가 선택되었다고 한들 B~J가 실패했다고 확신하기는 어렵다
  - 어떤 서비스의 경우 A,B,C,D가 노출된 상태에서 A를 클릭하고, 뒤로 가기 후 B를 클릭하면
    - A,B,C,D 모두 2개씩의 imp를 받고 A,B는 각각 1개의 click을 받는다 …
  - 아무튼, 알파와 베타의 가치는 동등하지 않을 수 있다!
  - 위 링크에서는 이에 따라 unclick에 대한 beta를 작게 업데이트 하고 있음 (1/32)
  - 그리고 CTR이 올랐다 (!)