Date: Tuesday, April 16, 2019

Multi-armed Bandits

* We want to optimize our choice much faster without making too many mistakes.

- K = number of arms (conditions)
- t = number of rounds (sample size)
- $r_{k,t}$ = estimated reward for arm k at round t

Aim: Identify $j^* \in \{1, \ldots, K\}$ that has optimal reward.
$\equiv$ identify j^* that minimizes regret.

↳ Because the optimal arm is unknown, and the reward at each round is stochastic (random), we accomplish our aim by minimizing expected regret.

Example: The click-through rate of an individual is unknown. When we test that person we get a 1 if clicked or 0 if not according to a Bernoulli distribution.

When thinking about multi-armed bandit algorithms, we need to decide how much exploration vs explotation at each round.

Note: Not all units are placed at once like in A/B testing; rather allocation is sequential

Greedy approach

Example: (continuation from slides)

- Step 4 → Play machine 3
  - ↳ If win → reward ($r_{34} = 1.00$) 100%
  - ↳ If lose → reward ($r_{34} = 2/3 \approx 0.6667$) 66.667%

we only played machine 3 three times

Note: There are several strategies for initialization:

1) Try each one and calculate rewards (maybe not enough ?)
2) Try each k times and calculate rewards (Better than 1 for opt, but creates more risks)
3) Set rewards according to past info or prior.

* Fast and easy to implement but, ..., we will most often get stuck in a local optimal because no exploration (not testing losing machines).

↳ can fix by adding a little exploration!

A note on analysis: A helpful plot is to look at win probabilities or metric over iteration. This can reveal trends and whether two arms have seemingly optimal metrics.

ε-greedy will converge to the optimal but is not the most efficient.

Softmax approach

- At round t, we have estimated probabilities (or metrics)
- Convert these probabilities to softmax rewards
  - ↳ balance the original probabilities according to their value relative to all other arms.

Note: These softmax rewards add to 1 and are probabilities. So, in the algorithm, we pull arm k with its probability r_{kt} at round t.

- In softmax, exploration and explotation is fully guided by the calculated softmax probs.

Example: Ranking Problems

- Identify the best p arms in order of their rewards.
  ↳ Practice: Best p recommendations on Netflix for "binge watchers".

Overall, there are $k > p$ recommendations that we could make.

So far, three strategies:

1) A/B testing overall k strategies

- Will work. Pairwise tests can determine ordering of metric.
  ↳ Ex. time watching the recommendation, Stars raiting after watching (issue: no-response).

- t-test on:

  $H_0: \mu_1 = \mu_2$ vs. $H_A: \mu_1 > \mu_2$
  where μ_j = mean time spend watching recommendation for rec j.

- Disadvantages: - Computationally "exhausting"
  - Time required to obtain desired sample size.
  - (*) - Higher risk for bad recommendations to valuable bing watchers.
    ↳ With A/B testing we do not avoid bad recommendations due to its "fully exploration" strategy.

2) ϵ - greedy approach

- Will it work? Yes, but it will take a long time because we only explore 2nd - pth best with probability ϵ/k.

- Point: Don't use for this problem.

3) Softmax approach

- Will it work? Yes!
  (This is the best choice for avoiding risk of bad recommendations).
- Exploring 2nd - pth best recommendations happen much faster than ϵ-greedy.
- The rate is proportional to the true reward of each.