Multi-Armed Bandits

Jon Courtney
Adam Richards
Frank Burkholder

galvanize

Multi-Armed Bandits

- Introduction and use cases of Multi-Armed Bandits
- Zen and the Art of Minimizing Regret
- Overview of Common Strategies
  - Epsilon-Greedy
  - Softmax
  - UCB1
  - Bayesian Bandit
- Other forms of Multi-Arm Bandits

galvanize

The Bandits Problem

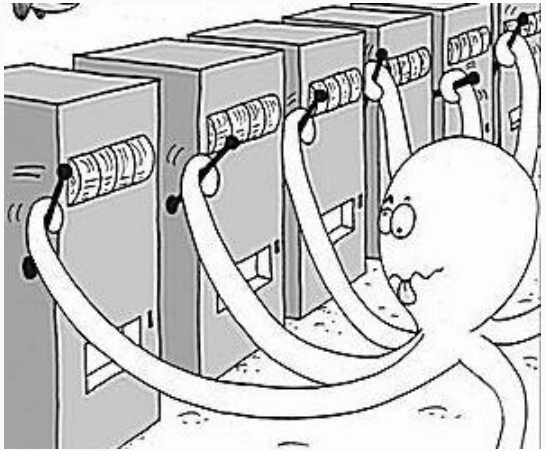
Suppose you are faced with N slot machines — so called “one-armed bandits”. Each “bandit” has an unknown probability of distributing a prize. (Assume for now the prizes are the same for each bandit; only the probabilities differ.) Some bandits are very generous, others not so much. Of course, you don't know what these probabilities are. By only choosing one bandit per round, your task is devise a strategy to maximize your winnings.

Origins

The multi-armed bandits problem was first tackled by Allied scientists during WWII...

“Efforts to solve it so sapped the energies and minds of Allied analysts that the suggestion was made that the problem be dropped over Germany, as the ultimate instrument of intellectual sabotage.”

— P. Whittle, Cambridge University



Applications of Multi-armed Bandits

galvanize

Multi-Armed Bandits go beyond traditional A/B testing...

- **Internet display advertising**: What ad strategy will maximize sales? Naturally minimizing strategies that do not work (generalizes to A/B/C/D strategies).
- **Biology**: How do animals maximize fitness w.r.t energy?
- **Finance**: Which stock portfolio gives the highest return, under time-varying return profiles?
- **Psychology**: How does punishment and reward affect our behavior? How do humans learn?
- **Dating** : Play the field or settle down? ;-)

The Bandits Dilemma

galvanize

- If we knew the bandit with the largest probability, then always picking that bandit would yield the maximum winnings. So our task could be phrased as *Find the best bandit, and find it as quickly as possible*.
- The task is complicated by the stochastic nature of the bandits. A suboptimal bandit can return many winnings, purely by chance, which would make us believe that it is a very profitable bandit. Similarly, the best bandit can return many duds. Should we keep trying losers then, or give up?
- A more troublesome problem is, if we have found a bandit that returns "pretty good" results, do we keep drawing from it to maintain our "pretty good" score, or do we try other bandits in hopes of finding an even-better bandit? This is the *exploration vs. exploitation* dilemma.

Exploration vs Exploitation

galvanize

- **Exploration**: Trying out different options to try and determine the reward associated with the given approach (i.e., acquiring more knowledge).

- **Exploitation**: Going with the approach that you currently believe to have the highest expected payoff (i.e., optimizing decisions based on existing knowledge).

Traditional A/B Testing: Process

galvanize

Consider the task of identifying the best of two websites based on click-thru rate (CTR):

- Decide your significance level and how many iterations to run, then...
- Start with pure exploration in which groups A and B are assigned equal number of users.
- Once you think you have determined the better site, switch to pure exploitation in which you stop the experiment and send all users to the better performer.

Traditional A/B Testing: Potential Issues

galvanize

- Equal number of observations are routed to A and B for a preset amount of time or iterations
- Only after that preset amount of time or iterations do we stop and use the better performer
- Waste time (and money!) showing users the site that is not performing as well
- No estimate of how likely it is that A is better (or worse) than B, just that you have evidence that A is better (or worse) at a certain confidence level.

Note: Bayesian A/B testing is an improvement over frequentist testing, but — in its basic form — still directs equal numbers of users to each web site until the experiment concludes.

Multi-Armed Bandit: Benefits

galvanize

- Shows a user the site that you think is best most of the time, based on what you know at the moment.
    - (How this is done is dictated by the strategy chosen.)
- As the experiment runs, we update the belief about the true CTR.
- Run for however long until we are satisfied the experiment has determined the better site.
- Balances exploration and exploitation rather than doing only one or the other.

Multi-Armed Bandit: Formalization

galvanize

- Model is given by a set of real distributions $B = R_1, \ldots, R_k$
- ...where each distribution is associated with a reward delivered by one of the K levers.
- We will let $\mu_1, \ldots, \mu_k$ be the mean values associated with these reward distributions.
- The gambler (i.e., the “agent”) plays one lever per round and observes the associated reward.
- The goal is to maximize the sum of the collective rewards, or (alternatively) to minimize the gambler/agent’s ***regret***.

Regret

The *regret p* that an agent experiences after *T* rounds is the difference between the reward sum associated with an optimal strategy and the sum of collected rewards.

$$p = T\mu^* - \sum_{t=1}^{T} \hat{r}_t$$

- μ^* is the maximal reward mean, $\mu^* = \max_k \{\mu_k\}$
- $\hat{r}_t$ is the reward at time t

Regret is simply a measure of how often you choose a suboptimal bandit. We can think of this as the cost function we are trying to minimize.

A Zero-Regret Strategy

galvanize

- A zero-regret strategy is a strategy whose average regret per round ***p/T*** tends to zero when the number of rounds played tends toward infinity.
- A zero-regret strategy does not guarantee you will never choose a sub-optimal outcome, but rather guarantees that, over time, you will tend to choose the optimal outcome.

Bandit strategies

- Epsilon-greedy
- UCB1 (upper confidence bound)
- Softmax
- Bayesian bandit

and others.

“from one thing, know ten thousand things”
— Miyamoto Musashi, The Book of Five Rings: Miyamoto Musashi

Epsilon-Greedy Strategy

- Explore with some probability ε (often 10%).
- Exploit at all other times; i.e., choose the bandit with the best performance so far.
- After we choose a given bandit we update its performance based on the result.
- Exhibits linear regret for constant ε

UCB1: Upper Confidence Bound

galvanize

For the UCB1 algorithm we will choose whichever bandit that has the largest value, where the value of bandit A is given as

$$\hat{\mu}_A + \sqrt{\frac{2 log N}{n_A}}$$

- $\hat{\mu}_A$: the observed payout rate of bandit A
- n_A : The number of times bandit A has been played
- N : the total number of times any bandit has been played

Exhibits “optimism in the face of uncertainty” : UCB1 gives weight to bandits that are relatively under-explored.

Softmax

- For the softmax algorithm we will choose the bandit randomly, in proportion to its estimated value relative to the other bandits

$$\frac{e^{\hat{\mu}_i/\tau}}{\sum_{j=1}^{k} e^{\hat{\mu}_j/\tau}}$$

- $\hat{\mu}_i$: the observed payout rate of bandit i
- τ : is a “temperature” parameter controlling the randomness of the choice (usually in range [0.01 - 0.1])
  - $\tau \to \infty$: Random exploration
  - $\tau \to 0$: Constant exploitation

Bayesian Bandits

galvanize

The Bayesian bandit algorithm involves modeling each of our bandits with a *beta distribution* using the following shape parameters:

α : 1 + number of times bandit has won

β : 1 + number of times bandit has lost

We then take a random sample from each bandit's distribution and choose the bandit with the highest value.

Bayesian bandits provide an approximately-optimal solution that scales and performs quite well.

Everyone loves a Bayesian bandit

Online Algorithms

galvanize

These strategies are an example of *online / reinforcement learning algorithms* because they are continuously updated with new information. The algorithm starts in a state of ignorance, and begins to acquire data by testing the system. As it acquires data and results, it learns what the best and worst behaviors are. In this case, it learns which bandit is the best.

Contextual Bandits

galvanize

- Like in a normal multi-arm problem, an agent must choose between arms during each iteration.
    - E.g., which online advertisement to serve
- In addition to bandit history, the agent also sees a *context vector* associated with the current iterations state.
    - E.g., the current user's profile information
- The agent uses the context vector and the history of past rewards to choose the arm to play in the current iteration.
- Over time, the aim is for the agent to learn how the context vectors relate to the associated rewards so as to pick the optimal arm.

Summary

- The problem is framed as a gambler/agent confronted with a row of slot machines (*aka one-armed bandits*)
- The agent has to decide which machines to play, how many times to play each machine, and it which order.
- Each bandit will provide a reward from an unknown distribution
- Objective is to maximize the sum of rewards earned through a series of lever pulls
- There are several classes of fairly optimal solutions

Review Questions

galvanize

- What is the difference between *exploration* and *exploitation*, and how are they related?
- Explain (in plain English) what *regret* means in a bandits context?
- Explain the following strategies: *epsilon-greedy, softmax,* and *UCB1*.
- What is the *Bayesian Bandit* strategy and how is it an example of *online learning*?


galvanize