Multi-armed bandit problem

Anežka Lhotáková

August 29, 2022

Abstract

In probability theory and machine learning, multi-armed bandit problem refers to a model which can be seen as a set of real distributions $\{F_1, \ldots, F_K\}$, each distribution being associated with the rewards delivered by one of the $K \in \mathbb{N}^+$ levers. Let $\mu_1, \ldots, \mu_K$ be the mean values associated with these reward distributions. The gambler iteratively plays one lever per round and observes the associated reward. The objective is to maximize the sum of the collected rewards. Several strategies or algorithms have been proposed as a solution to this problem in the last two decades. In this project, we will focus on the most popular strategies and theirs parameter optimization through heuristicall approach.

Introduction

In this paper, model with i.i.d. (independent and identically distributed) rewards is presented. We define

- horizon $\mathtt{H}$ as the number of rounds to be played,
- maximum mean reward as $\mu^* = \mathtt{max}\{\mu_1, \mu_2, \ldots, \mu_K\}$,
- regret ρ after $\mathtt{T}$ rounds played as

  $$\rho(\mathtt{T}) = \mathtt{T}\mu^* - \sum_{t=1}^{\mathtt{T}} r_t,$$

  where r_t is the reward in round t (ρ is also a random variable),
- expected regret as $\mathbb{E}[\rho(\mathtt{T})]$,
- symbol $\ell_t^i, (i = 1, \ldots, K$ and $t = 1, \ldots, \mathtt{H})$ for the levers.

The protocol to the model is following:

Given $\mathtt{H}$ rounds to play, considering K levers, in each round $t = 1, \ldots, \mathtt{H}$:

1. Based on the strategy, the algorithm picks a lever ℓ_t^i, $i = 1, \ldots, K$.
2. Algorithm observes reward r_t for the chosen lever.
3. If $t < \mathtt{H}$, the algorithm will play again (point 1., $t \to t + 1$), otherwise the game is over.

The goal is to maximize the total reward over the T rounds. Above that, we make two important assumptions:

- The algorithm observes only the reward for the selected action, and nothing else.
- The reward for each action is i.i.d. For each lever ℓ^i, there is a distribution F_i over reals, called the reward distribution. Every time the lever is chosen, the reward is sampled independently from its distribution. The reward distributions are initially unknown to the algorithm.

For better understanding of the model, let us begin with an example. Imagine player walking into a casino with two slot-machines. Both slot-machines ℓ_1, ℓ_2 have their already given distributions F_1, F_2. Let the reward r of each slot-machine be given as $r_1 \sim F_1 = \mathcal{N}(10, 3)$ and $r_2 \sim F_2 = \mathcal{N}(8, 4)$. Unfortunately, this information is hidden from the player. What possible strategies can the player use in order to maximize his win over $\mathtt{H} = 100$ rounds? The mean optimal win is

$$r_O = \mathtt{H} \cdot \mu^* = \mathtt{H} \cdot \max\{\mu_1, \mu_2\} = 100 \cdot 10 = 1\ 000.$$

a) *Explore only*: Firstly, he could choose to pull the levers randomly for the whole 100 rounds. In that case, his mean reward after 100 games would be

$$r = 50 \cdot 10 + 50 \cdot 8 = 900,$$

meaning expected regret is equal to $\mathbb{E}[\rho] = 1\ 000 - 900 = 100$.

b) *Exploit only*: Second strategy could be based on first experience. One round on each slot-machine could be played and explored. In first round, player pulls lever ℓ_A and gets a reward of 9. Then, pulling the second lever ℓ_B and winning 11. Naturally, for the remaining 98 rounds reasonable player would decide to keep playing (exploit) with the "better" lever ℓ_B. This strategy might have two possible outcomes

- positive outcome: the lever ℓ_B is the machine ℓ_1 with $r_1 \sim \mathcal{N}(10, 3)$ and thus the mean reward would be $r = \frac{9+11}{2} + 10 \cdot 98 = 990\$$,
- negative outcome: the lever ℓ_B is the less beneficial lever ℓ_2 with $r_2 \sim \mathcal{N}(8, 4)$ giving mean reward $r = \frac{9+11}{2} + 8 \cdot 98 = 794\$$.

Since the player does not know whether the decision of choosing ℓ_B was correct or not, expected regret of this strategy would be $\mathbb{E}[\rho] = 1\ 000 - [\frac{9+11}{2} + 0.5 \cdot (98 \cdot 10 + 98 \cdot 8)] = 1\ 000 - 892 = 108$.

Explore-only strategy gave the player expected regret of 100, which is lower than exploit-only strategy with expected regret 108. However, we can see how sensitive the problem is. What if the player got lucky and in exploitation part chose the more winning machine? This dilemma is called *Exploration-Exploitation dilemma* and even thought it is not the subject of this project, it is closely related to multi-armed bandit problem. It will come as no surprise that finding the balance between explore-exploit approaches is the key part of the most popular strategies for solving multi-armed bandit problem.

Strategies

In the example above were mentioned two basic strategies - **Exploration** and **Exploitation strategy**. In this section, some of the other strategies or solutions to multi-armed bandit problem are presented.

ε-first strategy

ε-first strategy extends the exploring part in exploitation strategy, which in general increases chances for choosing the more benefitial lever. Let $\varepsilon \in (0, 1)$.

The idea of ε-first strategy is to

1. explore (randomly choose levers) over the first $\varepsilon\mathtt{H}$ rounds,
2. observe mean reward for each lever after first $\varepsilon\mathtt{H}$ rounds played,
3. choose lever with higher mean reward,
4. exploit the chosen lever for the remaining $(1 - \varepsilon)\mathtt{H}$ rounds.

The advantage of this method is without a doubt extended exploration part, which provides more clues about the hidden distributions F_i of each lever. On the other hand, since we know so little about the distributions of each lever, we can not with certainty determine the optimal range of exploration part. Setting the value of ε so we do not perform unnecessary exploration rounds on the expenses of exploitation rounds and the other way around is a difficult task to do and strongly depends on the distributions.

ε-greedy strategy

Another way to upgrade exploration/exploitation strategy is to use $\varepsilon \in (0, 1)$ as a threshold for re-decision. In ε-greedy strategy we continuously monitor our decision during the whole game. Firstly, lever is randomly chosen to be played for the first $\varepsilon\mathtt{H}$ rounds. In $(\varepsilon\mathtt{H} + 1)$ round the other lever is played. At this point, re-decision is made. If the reward obtained from other lever is lower than mean reward obtained during $\varepsilon\mathtt{H}$ rounds, player continues to play with the same lever as in the $\varepsilon\mathtt{H}$ rounds. In the opposite case, player randomly choose which lever is to be played in the following $\varepsilon\mathtt{H}$ rounds.

In ε-greedy strategy is

1. randomly selected one of the levers,
2. exploited for $\varepsilon\mathtt{H}$ rounds and calculated mean reward $\tilde{r}$ after $\varepsilon\mathtt{H}$ rounds,
3. in $(\varepsilon\mathtt{H} + 1)$ round explored the second lever, obtaining levers reward r,
   - if $\tilde{r} \geq r \rightarrow$ keep same lever as in exploitation $\rightarrow$ 2.,
   - if $\tilde{r} < r \rightarrow$ randomly select new lever $\rightarrow$ 2. with new chosen lever,
4. the game ends after $\mathtt{H}$ rounds are played.

The contribution of this method to the improvement of previous startegies lies in periodic update of our decision after $\varepsilon\mathtt{H}$ rounds. But again, in disadvantage, the series of playing the more benefitial lever could be accidentaly interrupted by sporadic good win of the other lever in exploring round.

ε-decay strategy

ε-decay (also known as ε-decreasing) strategy is a strategy, where the $\varepsilon \in (0, 1)$ is not fixed, but it is a decreasing series of $(\varepsilon_i)_{i=1}^m$, where $\varepsilon_1 > \varepsilon_2 > \cdots > \varepsilon_m$. In the ε-decay strategy, the random lever is pulled with a probability of $\frac{1}{1+\beta t}$, where t is the number of rounds played, otherwise lever with highest mean reward is pulled.

ε-decay strategy

1. randomly selected one of the levers,
2. exploit for $\varepsilon_i\mathtt{H}$ rounds and calculated mean reward $\tilde{r}$ after $\varepsilon_i\mathtt{H}$ rounds,
3. in $(\varepsilon_i\mathtt{H} + 1)$ round is selected random lever with probability of $\frac{1}{1+\beta t}$, otherwise lever with the highest mean reward is pulled,
4. set ε_i as new ε_{i+1} $(\varepsilon_i > \varepsilon_{i+1})$ and return to point 2.,
5. the game ends after $\mathtt{H}$ rounds are played.

This method could be again upgraded by using more specific probability distribution for pulling the lever. Exaple of such method is SoftMax strategy:

SoftMax strategy

The SoftMax strategy consists of a random choice according to a Gibbs distribution. The lever k is chosen with probability

$$p_k = \frac{\mathsf{e}^{\frac{\hat{\mu}_k}{\tau}}}{\sum_{i=1}^n \mathsf{e}^{\frac{\hat{\mu}_i}{\tau}}},$$

where $\hat{\mu}_i$ is the estimated mean of the rewards brought by the lever i and $\tau \in \mathbb{R}^+$ is a parameter called the temperature. The choice of τ's value is left to the user. More generally, all methods that choose levers according to a probability distribution reflecting how likely the levers are to be optimal, are called probability matching methods.

SoftMax strategy using FSA

As an upgrade for SoftMax strategy could help Fast Simulated Annealing, where the temperature τ is not set as a constant, but as a function defined as

$$\tau(t) = \frac{T_0}{1 + (\frac{t}{n_0})^\alpha},$$

where T_0 is an initial temperature and α, n_0 are cooling parameters.

Results based on heuristics

Before the results, lets focus on the theory (based on the lectures [5]).

Grid Search

The Grid Search has been the traditional way of hyperparameter optimization. It is a very simple exhaustive searching through a manually specified subset of the hyperparameter space of a learning algorithm. A grid search algorithm must be guided by some performance metric. In this research, our metric will be the mean reward after 1 000 games.

Best bandits based on Grid Search

In this text, the goal is to determine the best multi armed bandit strategy among mentioned strategies. In the game, following rules are defined:

- the number of rounds to be played `H` $= 1\ 000$,
- the number of bandits: 10,
- the distributions of rewards for each bandit:

$$r_0 \sim N(0,1),$$
$$r_1 \sim N(1,1),$$
$$\dots,$$
$$r_9 \sim N(9,1).$$

This game will be repeated 1 000 times and the average final reward over these thousand games will be observed. As the distributions are now revealed, it is clear we except the algorithms will eventually discover the last lever with r_9 is the most generous one.

The best ε-first strategy

Based on the definition, we are exploring the possible values of $\varepsilon = 0.00, 0.01, 0.02, \dots, 1.00$. In the following graph the final mean reward (after 1 000 games is presented). Based on the results, we declare $\varepsilon =$ the best option with the respect to the setting of the game and experiment.

The best ε-greedy strategy

Similarly as in previous method, we are exploring the possible values of $\varepsilon = 0.00, 0.01, 0.02, \dots, 1.00$. However the role of ε slightly changes and now it defines a points in the game, where the decision might be switched. In the following graph the final mean reward (after 1 000 games is presented). Based on the results, we declare $\varepsilon = 0.09$ the best option with the respect to the setting of the game and experiment.

The best ε-decay strategy

Similarly as in previous method, we are exploring the possible values of $\varepsilon = 0.00, 0.01, 0.02, \dots, 1.00$. However the role of ε slightly changes and now it defines a points in the game, where the decision might be switched. In the following graph the final mean reward (after 1 000 games is presented). Based on the results, we declare $\varepsilon = 0.09$ the best option with the respect to the setting of the game and experiment.

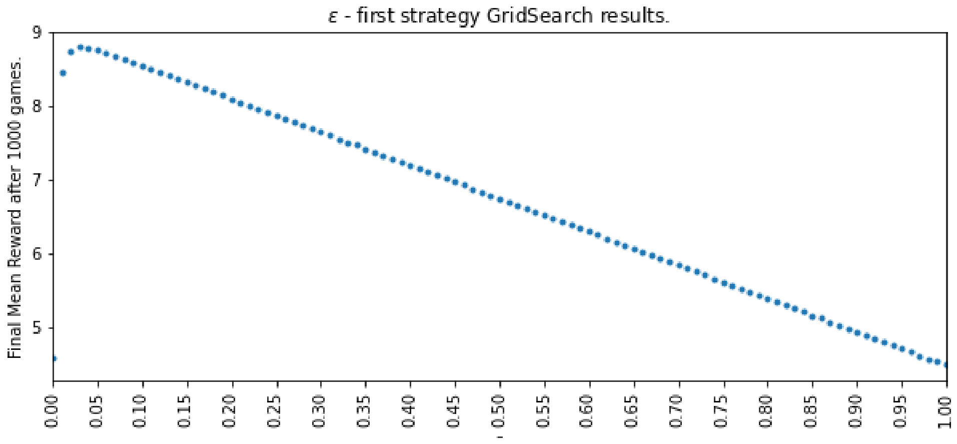

Figure 1: The genetic optimization.

The best SoftMax strategy

Similarly as in previous method, we are exploring the possible values of $\varepsilon = 0.00, 0.01, 0.02, \ldots, 1.00$. However the role of ε slightly changes and now it defines a points in the game, where the decision might be switched. In the following graph the final mean reward (after 1 000 games is presented). Based on the results, we declare $\varepsilon = 0.09$ the best option with the respect to the setting of the game and experiment.

GA

Genetic algorithms are commonly used to generate high-quality solutions to optimization and search problems by relying on biologically inspired operators such as mutation, crossover and selection. In the following picture, which I stole from our lectures [7.], is described the idea behind GA:

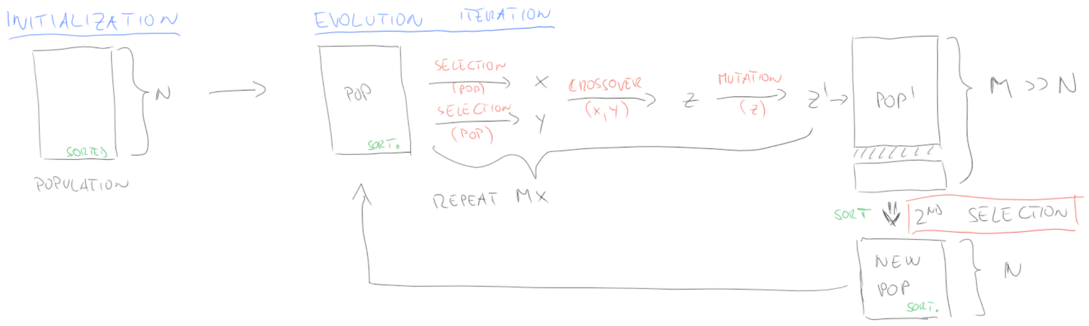

Figure 2: The genetic optimization.

As Grid Search was just fine for the ε-based strategies, the SoftMax strategy offers an opportunity to test Genetic Algorithms. The geneticalgorithm library offers an implementation of GA, the documentation can be found here: https://github.com/rmsolgi/geneticalgorithm .

References

1. **Ritvik Kharkar**, A.A. [ritvikmath], Multi-Armed Bandit: Data Science Concepts (2020/09/23), YouTube, https://www.youtube.com/watch?v=e3L4VocZnnQ&ab_channel=ritvikmath
2. **Robert C. Gray**, A.A. [Academic Gamer], Multi-Armed Bandits: A Cartoon Introduction - DCBA #1 (2020/08/08), YouTube, https://www.youtube.com/watch?v=bkw6hWvh_3k&ab_channel=AcademicGamer
3. **Aleksandrs Slivkins**, Introduction to Multi-Armed Bandits, Microsoft Research NYC, Version September 2019, 2017-2019
4. **João Gama, Rui Camacho, Pavel B. Brazdil, Alípio Mário Jorge, Luís Torgo**, Machine Learning: ECML 2005, 16th European Conference on Machine Learning, Proceedings, Porto, Portugal, October 3-7, 2005, 0302-9743
5. HEUR Lectures, 2022
6. https://en.wikipedia.org/wiki/Hyperparameter_optimization