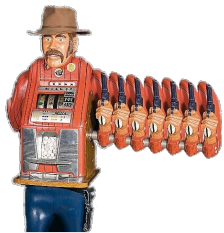
Machine Learning

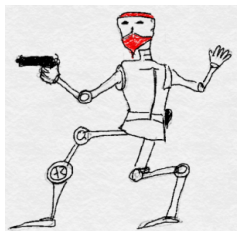
Multi-Armed Bandit



Alberto Maria Metelli

Credits to Francesco Trovò

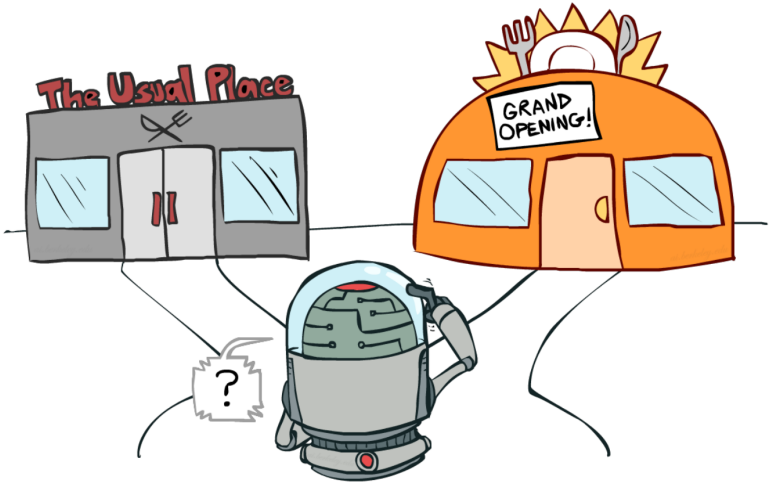
24 May, 2022



Outline

Traditional Motivating Example

Restaurant selection problem:



Motivating Example

Beer selection problem:

- We enter a newly opened brewery
- We are allowed to chose among a set of available beers (one at a time)
- After each beer you assign a mark from 1 to 10 according to how much you liked it
- It might happen that the value you assign to a beer varies (e.g., different bottles might have slightly different tastes)
- Your goal is twofold:
  - Find the beer you like the most
  - You do not want to get drunk in doing that (minimize the order of beers you do not like)

Exploration/Exploitation Dilemma

- The main point is that we are not sure about the value of each action $Q(a|s)$
- **Online** decision making make us face a fundamental choice:
  - **Exploration**: gather more information from unexplored/less explored options
  - **Exploitation**: select the option we consider to be the best one so far
- Depending on how much we are far-sighted we might make some sacrifice in the short-term to gain more in the future
  - Infinite time horizon: we want to gather enough information to find the best overall decision
  - Finite time horizon: we want to minimize the short-term loss due to uncertainty

Example of Real Applications

- Clinical Trial
  - Exploration: Try new treatments
  - Exploitation: Choose the treatment that provided the best results so far
- Slot machine (a.k.a. one-armed bandit) selection
  - Exploration: Try all the available slot machines
  - Exploitation: Pull the one which provided you the highest payoff so far
- Game Playing
  - Exploration: Play an unexpected move
  - Exploitation: Play the move you think is the best
- Oil Drilling
  - Exploration: Drill at an unexplored location
  - Exploitation: Drill at the best known location with current information

Different Kind of MAB

The only thing we need to have a full definition of the problem is the characterization of the **reward**:

- **Deterministic**: we have a single value for the reward for each arm (trivial solution)
- **Stochastic**: the reward of an arm is drawn from a distribution which is stationary over time
- **Adversarial**: an adversary chooses the reward we get from an arm at a specific round, knowing the algorithm we are using to solve the problem

From MDP to MAB

We can see the Multi-Armed Bandit setting as a specific case of an MDP

- $\mathcal{S}$ is a set of states $\rightarrow$ single state $\mathcal{S} = \{s\}$
- $\mathcal{A}$ is a set of actions $\rightarrow$ **arms** $\mathcal{A} = \{a_1, \ldots, a_N\}$
- P is a state transition probability matrix $\rightarrow P(s|a_i, s) = 1, \forall a_i$
- R is a reward function $\rightarrow R(s, a_i) = R(a_i)$
- γ is a discount factor $\rightarrow$ finite time horizon $\gamma = 1$
- μ^0 is a set of initial probabilities $\rightarrow \mu^0(s) = 1$

Common Approaches in RL

- ε-greedy:

$$\pi(a_i|s) = \begin{cases} 1-\varepsilon & \text{if } \hat{Q}(a_i|s) = \max_{a \in \mathcal{A}} \hat{Q}(a|s) \\ \dfrac{\varepsilon}{|\mathcal{A}|-1} & \text{otherwise} \end{cases}$$

  - Perform the greedy action except for a small amount of times
  - Does not achieve the optimal policy
- Softmax (Boltzmann distribution):

$$\pi(a_i|s) = \frac{e^{\frac{\hat{Q}(a_i|s)}{\tau}}}{\sum_{a \in \mathcal{A}} e^{\frac{\hat{Q}(a|s)}{\tau}}}$$

  - Weights the actions according to its estimated value $\hat{Q}(a|s)$
  - τ is a temperature parameter which decreases over time

Even if these algorithms converge to the optimal choice, we do not know how much we lose during the learning process

Multi-Armed Bandit Setting

A Multi-Armed Bandit problem is a tuple $\langle \mathcal{A}, \underline{\mathcal{R}} \rangle$

- $\mathcal{A} = \{a_1, \ldots, a_N\}$ is a set of N possible arms (choices)
- $\mathcal{R}$ is an set of **unknown** distributions $\mathcal{R}(a_i)$

$$\text{Reward: } r \sim \mathcal{R}(a_i) \qquad \text{Expected reward: } R(a_i) = \mathbb{E}_{r \sim \mathcal{R}(a_i)}[r]$$

- We assume (the random variable) $r \in [0, 1]$

The process we consider is the following:

- At each round t the agent selects a single arms a_{i_t}
- The environment generates a stochastic reward $r_{a_{i_t},t}$ drawn from $\mathcal{R}(a_{i_t})$
- The agent updates her information by means of a history h_t (pulled arm and received reward)

Objective of a MAB Algorithm

The final objective of the agent is to maximize the cumulative reward over a given time horizon T:

$$\sum_{t=1}^{T} r_{a_{i_t},t}$$

where $r_{a_{i_t},t}$ is the realization of the reward for the arm a_{i_t} we choose for the turn

Possibly we also want to converge to the option with largest expected reward if one considers $T \to \infty$

Alternative Goal: Minimize the Regret

The objective function can be reformulated in the following way:

- Define the expected reward of the **optimal arm** a^* as:

$$R^* = R(a^*) = \max_{a \in \mathcal{A}} R(a) = \max_{a \in \mathcal{A}} \mathbb{E}_{r \sim \mathcal{R}(a)}[r]$$

- At a given time step t, we select the action a_{i_t}, we observe the reward $r_{a_{i_t},t} \sim \mathcal{R}(a_{i_t})$ and we incur in an **expected loss** of:

$$R^* - R(a_{i_t})$$

Expected Pseudo Regret Definition

We want to minimize the expected regret suffered over a finite time horizon of T rounds

Expected Pseudo Regret

$$L_T = TR^* - \mathbb{E}\left[\sum_{t=1}^{T} R(a_{i_t})\right]$$

The expected value is taken w.r.t. the stochasticity of the reward function and the randomness of the used algorithm

Note that the maximization of the cumulative reward is equivalent to the minimization of the cumulative regret

Another Regret Definition

Another way of reformulating the cumulate regret is:

- Define the average difference in reward between a generic arm a_i and the optimal one a^* as $\Delta_i := R^* - R(a_i)$ (**suboptimality gap**)
- Define the number of times an arm a_i has been pulled after a total of $t-1$ time steps as $N_t(a_i)$

$$
\begin{aligned}
L_T &= TR^* - \mathbb{E}\left[\sum_{t=1}^{T} R(a_{i_t})\right] \\
&= \mathbb{E}\left[\sum_{t=1}^{T} R^* - R(a_{i_t})\right] \\
&= \sum_{a_i \in \mathcal{A}:\Delta_i > 0} \mathbb{E}[N_T(a_i)]\,(R^* - R(a_i)) = \sum_{a_i \in \mathcal{A}:\Delta_i > 0} \mathbb{E}[N_T(a_i)]\Delta_i
\end{aligned}
$$

i.e., we want to minimize the number of times we select a suboptimal arm

Lower Bound for Stochastic MAB

- The definition of regret in terms of Δ_i implies that any algorithm performance is determined by the **similarity** among arms
- The more the arms are similar, the more the problem is difficult

It is possible to show the following:

Theorem (MAB lower bound, Lai & Robbins 1985)

Given a MAB stochastic problem ***any algorithm*** *satisfies:*

$$\lim_{T\to\infty} L_T \geq \log T \sum_{a_i \in \mathcal{A}:\Delta_i>0} \frac{\Delta_i}{KL(\mathcal{R}(a_i), \mathcal{R}(a^*))}$$

where $KL(\mathcal{R}(a_i), \mathcal{R}(a^*))$ *is the Kullback-Leibler divergence between the two Bernoulli distributions* $\mathcal{R}(a_i)$ *and* $\mathcal{R}(a^*)$

Pure Exploitation Algorithm

- Always select the action s.t. $a_{i_t} = \arg\max_{a \in \mathcal{A}} \hat{R}_t(a)$ where the expected reward for an arm is:

$$\hat{R}_t(a_l) = \frac{1}{N_t(a_l)} \sum_{j=1}^{t-1} r_{a_{i_j},j} \, \mathbb{1}\left\{a_l = a_{i_j}\right\}$$

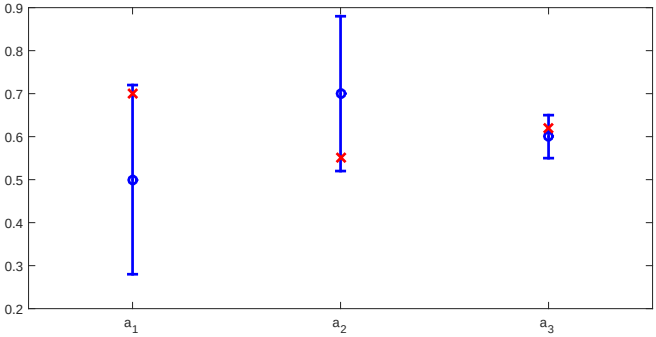
- This strategy is the one which is trying to minimize the cumulated regret in a straightforward way
- Might not converge to the optimal action
- We are not considering the uncertainty corresponding to the $\hat{R}_t(a)$ estimate

Hint: We need to provide an explicit bonus for **exploration**

Two Formulations

- **Frequentist** formulation
  - $R(a_1), \dots R(a_N)$ are **unknown parameters**
  - A policy selects at each time step an arm based on the observation history
- **Bayesian** formulation
  - $R(a_1), \dots R(a_N)$ are **random variables** with prior distributions $f_1, \dots, f_N$
  - A policy selects at each time step an arm based on the observation history and on the provided priors

Optimism in face of Uncertainty



- The more we are uncertain on a specific choice
- The more we want the algorithm to explore that option
- We might lose some value in the current round, but it might turn out that the explored action is the best one

Upper Confidence Bound Approach

- Instead of using the empiric estimate we consider an upper bound $U(a_i)$ over the expected value $R(a_i)$
- More formally, we need to compute an upper bound

$$U(a_i) := \hat{R}_t(a_i) + B_t(a_i) \geq R(a_i)$$

  with **high probability** (e.g., more than $1 - \delta$, $\delta \in (0, 1)$)
- The bound length $B_t(a_i)$ depends on how much information we have on an arm, i.e., the number of times we pulled that arm so far $N_t(a_i)$:
  - Small $N_t(a_i) \rightarrow$ large $U(a_i)$ (the estimated value $\hat{R}_t(a_i)$ is **uncertain**)
  - Large $N_t(a_i) \rightarrow$ small $U(a_i)$ (the estimated value $\hat{R}_t(a_i)$ is **accurate**)

Höeffding Inequality Bound

In order to set the upper bound we resort to a classical **concentration inequality**:

Höeffding Bound

Let $X_1, \ldots, X_t$ be i.i.d. random variables with support in $[0, 1]$ and identical mean $\mathbb{E}[X_i] =: X$ and let $\bar{X}_t = \dfrac{\sum_{i=1}^{t} X_i}{t}$ be the sample mean. Then:

$$\mathbb{P}\left(X > \bar{X}_t + u\right) \leq e^{-2tu^2}$$

We will apply this inequality to the upper bounds corresponding to each arm:

$$\mathbb{P}\left(R(a_i) > \hat{R}_t(a_i) + B_t(a_i)\right) \leq e^{-2N_t(a_i)B_t(a_i)^2}$$

Computing the Upper Bound

- Pick a probability p that the real value exceeds the bound:

$$e^{-2N_t(a_i)B_t(a_i)^2} = p$$

- Solve to find $B_t(a_i)$:

$$B_t(a_i) = \sqrt{\frac{-\log p}{2N_t(a_i)}}$$

- Reduce the value of p over time, e.g., $p = t^{-4}$

$$B_t(a_i) = \sqrt{\frac{2\log t}{N_t(a_i)}}$$

- Ensure to select the optimal action as the number of samples increases:

$$\lim_{t\to\infty} B_t(a_i) = 0 \Rightarrow \lim_{t\to\infty} U_t(a_i) = R(a_i)$$

UCB1

- For each time step t
- Compute $\hat{R}_t(a_l) = \dfrac{\sum_{j=1}^{t-1} r_{a_{i_j},j} \, \mathbb{1}\left\{a_l = a_{i_j}\right\}}{N_t(a_l)} \;\; \forall a_i \in \mathcal{A}$
- Compute $B_t(a_l) = \sqrt{\dfrac{2 \log t}{N_t(a_l)}} \;\; \forall a_l \in \mathcal{A}$
- Play arm $a_{i_t} = \arg\max_{a_l \in \mathcal{A}} \left\{ \hat{R}_t(a_l) + B_t(a_l) \right\}$

Theorem (UCB1 Upper Bound, Auer & Cesa-Bianchi 2002)

At finite time T, the expected total regret of the UCB1 algorithm applied to a stochastic MAB problem is:

$$L_T \leq 8 \log T \sum_{a_i \in \mathcal{A}: \Delta_i > 0} \frac{1}{\Delta_i} + \left(1 + \frac{\pi^2}{3}\right) \sum_{a_i \in \mathcal{A}: \Delta_i > 0} \Delta_i$$

Other UCBs

- UCBV $\rightarrow$ Bernstein inequality
- KLUCB $\rightarrow$ Chernoff upper bound
  - Matches the lower bound for Bernoulli rewards!
- BayesUCB $\rightarrow$ Upper bounds based on beta distribution
- ...

Thompson Sampling

Designed by William R. Thompson in 1933
General Bayesian methodology for online learning

- Consider a **Bayesian prior** for each arm $f_1, \ldots, f_N$ as a starting point
- At each round t sample from each one of the distributions $\hat{r}_1, \ldots, \hat{r}_N$
- **Pull** the arm a_{i_t} with the highest sampled value $i_t = \arg\max_i \hat{r}_i$
- **Update** the prior incorporating the new information

We will assume Bernoulli rewards

Thompson Sampling for Bernoulli Rewards

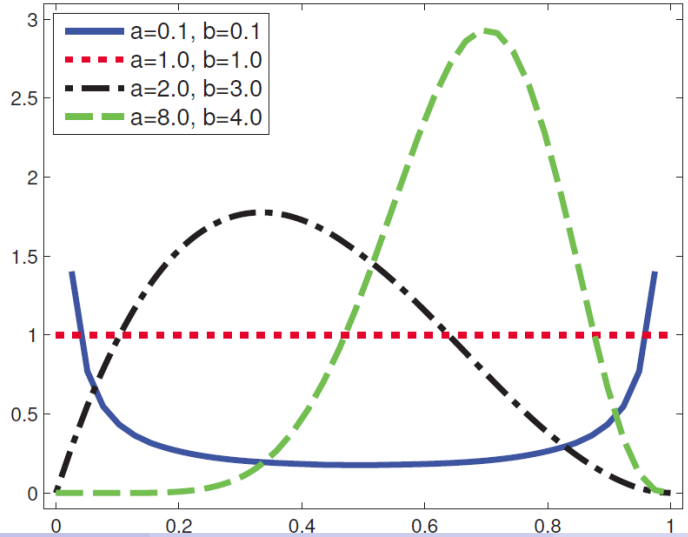
- The conjugate prior distributions are $Beta(\alpha, \beta)$ and the Bernoulli
- Start from a prior $f_i(0) = Beta(1, 1)$ (uniform prior) for each arm $a_i \in \mathcal{A}$
- Keep a distribution $f_i(t) = Beta(\alpha_t, \beta_t)$ incorporating information gathered from each arm $a_i \in \mathcal{A}$
- Incremental update formula for the pulled arm a_i:
  - In the case of a success occurs $f_i(t+1) = Beta(\alpha_t + 1, \beta_t)$
  - In the case of a failure occurs $f_i(t+1) = Beta(\alpha_t, \beta_t + 1)$

Theorem (Thompson Sampling Upper Bound, Kaufmann & Munos 2012)

At time T, the expected total regret of Thompson Sampling algorithm applied to a stochastic MAB problem is:

$$L_T \leq O\left(\sum_{a_i \in \mathcal{A}: \Delta_i > 0} \frac{\Delta_i}{KL(\mathcal{R}(a_i), \mathcal{R}(a^*))}(\log T + \log\log T)\right)$$

Examples of Beta Distributions



Adversarial MAB Setting

A *Multi-Armed Bandit Adversary setting* is a tuple $\langle \mathcal{A}, \underline{\mathcal{R}} \rangle$

- $\mathcal{A} = \{a_1, \ldots, a_N\}$ is a set of N possible arms (choices)
- $\mathcal{R}$ is a reward vector for which the realization $r_{a_i,t}$ is decided by an **adversary** player at each turn

The process we consider is the following:

- At each time step t the agent selects a single arms a_{i_t}
- At the same time the adversary chooses rewards $r_{a_i,t},\ \forall a_i \in \mathcal{A}$
- The agent gets reward $r_{a_{i_t},t}$
- The final objective of the agent is to maximize the cumulative reward over a time horizon T:

$$\sum_{t=1}^{T} r_{a_{i_t},t}$$

We assume that the adversary is **oblivious** (or non-adaptive), i.e, she selects the rewards in advance knowing the agent's algorithm but not the actual action realization.

Adversarial Regret Definition

- We cannot compare the cumulated regret we gained with the optimal one
- Moreover, the fact that there is an adversary choosing the regret does not allow to use deterministic algorithms (e.g., UCB)

Pseudo Regret

$$L_T = \max_{a_i \in \mathcal{A}} \mathbb{E}\left[\sum_{t=1}^{T} r_{a_i,t}\right] - \mathbb{E}\left[\sum_{t=1}^{T} r_{a_{i_t},t}\right]$$

where the expectation is w.r.t. the randomization used by the algorithm and the adversary

We are comparing the policy with the best constant action

Lower Bound

Theorem (Minimax Lower Bound)

Let $\sup$ *be the supremum over all distribution of rewards such that, for* $i \in \{1, \dots, N\}$ *the rewards* $r_{i,1}, \dots, r_{i,T}$ *and* $r_{i,j} \in \{0, 1\}$ *are i.i.d., and let* $\inf$ *be the infimum over all forecasters. Then:*

$$\inf \sup L_T \geq \frac{1}{20}\sqrt{TN}$$

EXP3

- Variation of the Softmax algorithm
- Probability of choosing an arm:

$$\pi_t(a_i) = (1-\beta)\frac{w_t(a_i)}{\sum_j w_t(a_j)} + \frac{\beta}{N}$$

where:

$$w_{t+1}(a_i) = \begin{cases} w_t(a_i)e^{\eta \frac{r_{a_j,t}}{\pi_t(a_i)}} & \text{if } a_i \text{ has been pulled, i.e., } a_{i_t} = a_i \\ w_t(a_i) & \text{otherwise} \end{cases}$$

EXP3 Upper Bound

Theorem (EXP3 Upper Bound)

At time T, the pseudo regret of EXP3 algorithm applied to an adversarial MAB problem with $\beta = \eta = \sqrt{\frac{N \log N}{(e-1)T}}$ *is:*

$$L_T \leq O(\sqrt{TN \log N}),$$

where the expectation is taken with respect to both the random generation rewards and the internal randomization of the forecaster

An Example

- In the beer selection problem you now have a set of breweries
- Each night your friend decides which one to pick
- Once you are in a specific brewery you are free to pick a beer of your choice

Is this a Bandit problem?

- If we fix the brewery, it is a stochastic MAB
- Since we do not control the transition from one brewery to another, we can use MAB techniques over each one of the breweries
- Called **Contextual MAB**

Beyond Classical Bandit Problems

Other Bandits:

- Budget-MAB: we are allowed to pull arms until a fixed budget elapses, where the pulling action incurs in a reward and a cost
- Continuous Armed Bandit: we have a set of arms $\mathcal{A}$ which is not finite

Other sequential decision settings:

- Best-arm identification problem: we just want to identify the optimal arm with a given confidence, without caring about the regret
- Expert setting: we are allowed also to see the reward which would have given the not pulled arm each turn (online learning problem)

References

An exhaustive reviews of most of the existing bandit techniques and results are:

Regret Analysis of Stochastic and Nonstochastic Multi-armed Bandit Problems

By Sebastien Bubeck and Nicolò Cesa-Bianchi

`http://sbubeck.com/SurveyBCB12.pdf`

Bandit Algorithms

By Tor Lattimore and Csaba Szepesvári

`https://tor-lattimore.com/downloads/book/book.pdf`