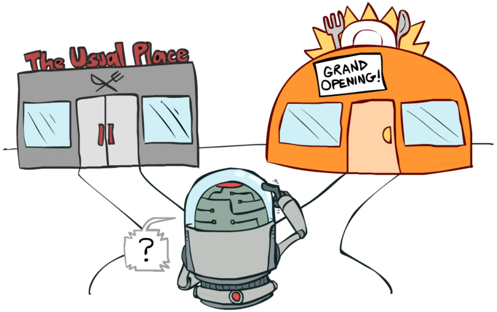
Machine Learning

Multi-armed Bandit

Alberto Maria Metelli and Francesco Trovò

Restaurant selection problem



- 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

Example of Real Applications

- Clinical Trial
  - Exploration: Try new treatments
  - Exploitation: Choose the treatment that provides the best results
- 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

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** $\mathcal{A} = \{a_1, \dots, a_N\}$
- 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 $\rightarrow$ $\mu^0(s) = 1$

Different Kinds 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

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
- 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 $\hat{Q}(a|s)$

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 $\langle \mathcal{A}, \mathcal{R} \rangle$

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

Reward: $r \sim \mathcal{R}(a_i)$ 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 history h_t (pulled arms and received rewards)

Goal: Minimize the Regret

The objective function can be reformulated in the following way:

- Idea: Quantfy the loss due to the fact we are learning
- 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
- A good algorithm should have a regret s.t. $\frac{L_T}{T} \rightarrow 0$ for $T \rightarrow \infty$

Lower Bound for Stochastic MAB

It is possible to show the following:

MAB lower bound, Lai & Robbins 1985

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

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

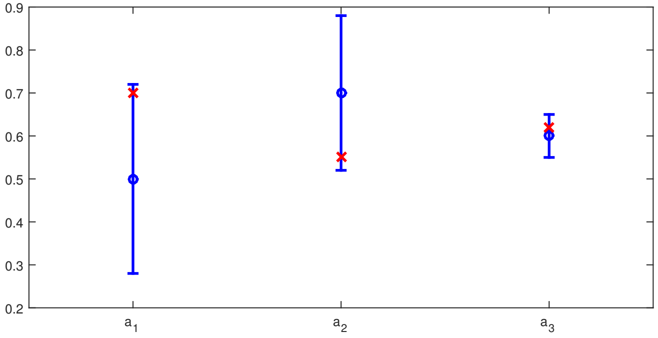
where $KL(\mathcal{R}(a_i), \mathcal{R}(a^*))$ is the Kullback-Leibler divergence between the two Bernoulli distributions $\mathcal{R}(a_i)$ and $\mathcal{R}(a^*)$

- Since the regret in terms of $\Delta_i := R^* - R(a_i)$ implies that any algorithm performance is determined by the **similarity** among arms
- The more the arms are similar, the more the problem is difficult

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**)

UCB1

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

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:

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

Thompson Sampling

- Designed by William R. Thompson in 1933
- Initially designed for clinical trials
- General Bayesian methodology for online learning

The procedure consists in:

- Consider a **Bayesian prior** for each arm $f_1, \dots, f_N$ as a starting point
- At each round t sample from each one of the distributions $\hat{r}_1, \dots, \hat{r}_N$
- **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

Remark: we will assume Bernoulli rewards (1: success, 0: failure)

Thompson Sampling for Bernoulli Rewards

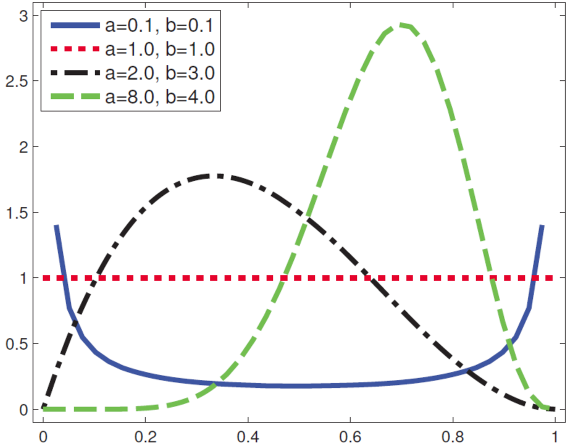
- The conjugate prior distributions for Bernoulli are $Beta(\alpha, \beta)$
- Start from a prior $f_i(0) = Beta(1, 1)$ (uniform prior) for each arm $a_i \in \mathcal{A}$
- Keep a distribution $f_i(t) = Beta(\alpha_{i,t}, \beta_{i,t})$ incorporating information gathered from each arm $a_i \in \mathcal{A}$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

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



References

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`