

Multi-Arm Bandits

Easwar Subramanian

TCS Innovation Labs, Hyderabad

Email : cs5500.2020@iith.ac.in

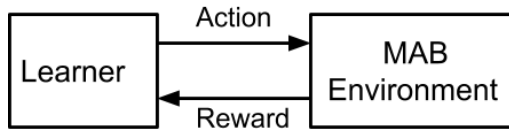
October 14, 2023

- 1 Introduction to Bandit Problem
- 2 Naive Approaches
- 3 Optimism in the Face of Uncertainty
- 4 Thompson Sampling

Introduction to Bandit Problem



- ▶ **Learning Problem** : Which arm is the best ?
- ▶ **Decision Problem** : Which arm to pull next ?

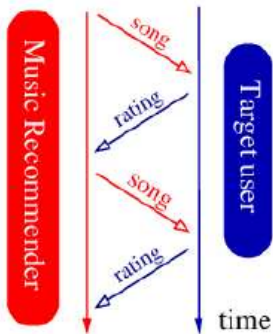


- ▶ There are K arms to pull and there are N rounds
- ▶ The agent can pull any of the K arms in each round $t \in \{0, 1, \dots, N\}$
- ▶ On pulling arm a , the agent gets a random reward r_a sampled from a distribution (independent of previous choices and rewards)
- ▶ Goal
 - ★ **Regret minimization** : : Maximize the sum of rewards (in expectation)
 - ★ **Best Arm Identification** : Discover the best arm (given some budget)

- ▶ Managing exploration-exploitation trade-off
- ▶ Baby reinforcement learning
- ▶ Lots of applications in online learning

Example : Music Recommendation

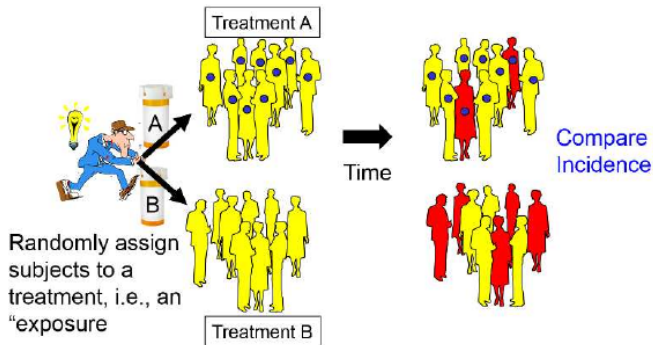
Music Recommendation



Objective maximize the sum of ratings

- Decision Problem : Which song to play next ?

Example : Clinical Trial



- Decision Problem : Which treatment to be given for next patient ? Or Which line of treatment is best ?

Example : Dynamic Pricing

- ▶ Seller has a commodity to sell in market
- ▶ N possible discrete prices
- ▶ Observation : Sale or no sale for offered price
- ▶ Explore different prices or pick the best performing prices so far

- ▶ Ad Placement
- ▶ A/B Testing
- ▶ Network Routing
- ▶ Game Tree Search

- ▶ A multi-arm bandit is defined as a tuple $\langle \mathcal{A}, \mathcal{R} \rangle$
- ▶ \mathcal{A} is the set of arms available
- ▶ $\mathcal{R}^a(r) = \mathbb{P}(r|a)$ is the unknown of distribution of rewards of arm a
- ▶ At each step t the agent selects an action $a_t \in \mathcal{A}$ and gets a reward $r_t \sim \mathcal{R}^{a_t}$
- ▶ The goal is to maximise cumulative reward $\sum_{t=1}^N r_t$

- ▶ The goal is to maximize cumulative reward $\sum_{t=1}^N r_t$
- ▶ Define the action value function $Q(a)$ to be the mean reward for action a i.e. $Q(a) = \mathbb{E}(r|a)$

- ▶ The optimal value V^* is

$$V^* = Q(a^*) = \max_{a \in \mathcal{A}} Q(a)$$

- ▶ The regret is the lost opportunity at one step

$$l_t = \mathbb{E}[V^* - r_t] = V^* - \mathbb{E}[r_t]$$

- ▶ Total regret is the total opportunity loss

$$L_N = \mathbb{E} \left[\sum_{t=1}^N (V^* - r_t) \right] = NV^* - \mathbb{E} \left[\sum_{t=1}^N r_t \right]$$

- ▶ Maximize cumulative reward \equiv Minimize total regret

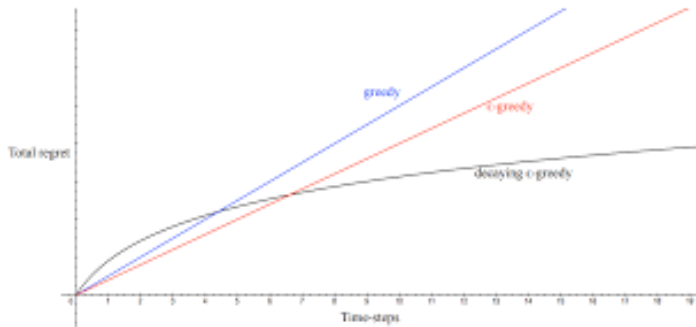
- ▶ Let the **count** $N_t(a)$ be the number of times arm a is pulled upto time t
($N_t(a) \equiv \sum_{\tau=1}^t \mathbb{1}_{A_\tau=a}$)
- ▶ Let Δ_a be the **gap** between optimal reward (from optimal action a^*) and reward of arm a

$$\Delta_a = V^* - Q(a)$$

- ▶ Regret is a function of gaps and counts given as

$$\begin{aligned} L_N &= \mathbb{E} \left[\sum_{t=1}^N (V^* - r_t) \right] = \mathbb{E} \left[\sum_{t=1}^N \Delta_{A_t} \right] \\ &= \mathbb{E} \left[\sum_{t=1}^N \sum_{a \in \mathcal{A}} \mathbb{1}_{A_t=a} \Delta_a \right] = \sum_{a \in \mathcal{A}} \Delta_a \mathbb{E} (N_N(a)) \end{aligned}$$

- ▶ A good algorithm ensures small counts for large gaps



- ▶ Algorithms that **explore forever** have total linear regret
- ▶ Algorithms that **never explore** have total linear regret
- ▶ **Question** : Is it possible for develop algorithms have sub-linear regret ?

- ▶ We consider algorithms that estimate $\hat{Q}_t(a) \approx Q(a)$
- ▶ The sample estimate $\hat{Q}(a)$ is estimated via Monte-Carlo simulations

$$\hat{Q}_t(a) = \frac{1}{N_t(a)} \sum_{\tau=1}^t r_{\tau} \mathbb{1}(A_{\tau} = a)$$

Naive Approaches

- ▶ At any time t , a greedy algorithm selects the action with highest $\hat{Q}_t(a)$, i.e.

$$a_t^* = \arg \max_{a \in \mathcal{A}} \hat{Q}_t(a)$$

- ▶ Greedy algorithm can lock into a sub-optimal arm forever
- ▶ \implies Greedy has linear total regret

Algorithm Explore then Commit

```
1: Let  $K$  be the number of arms;  $N$  be the total rounds; Initialize  $M$ 
2: for  $m = 1, 2, \dots, M$  do
3:   for  $a = 1, 2, \dots, K$  do
4:     Pull arm  $a$ ; Observe reward  $r_a$ ; Compute mean reward  $\hat{Q}(a)$  for arm  $a$ ;
5:   end for
6: end for
7: for  $i = MK + 1, \dots, N$  do
8:   Pull the arm with the best mean reward [i.e.  $a^* = \arg \max_a \hat{Q}(a)$ ]
9: end for
```

Question : Which parts of the algorithm explores and which part exploits ?

Algorithm Explore then Commit

1: Let K be the number of arms; N be the total rounds; Initialize M

Exploration Phase

2: **for** $m = 1, 2, \dots, M$ **do**

3: **for** $a = 1, 2, \dots, K$ **do**

4: Pull arm a ; Observe reward r_a ; Compute mean reward $\hat{Q}(a)$ for arm a ;

5: **end for**

6: **end for**

Exploitation Phase

7: **for** $t = MK + 1, \dots, N$ **do**

8: Pull the arm with the best mean reward [i.e. $a^* = \arg \max_a \hat{Q}(a)$]

9: **end for**

Question : Why do we expect this algorithm to work ?

- ▶ Suppose X_1, X_2, \dots are independent samples of a random variable X having mean μ
- ▶ Denote empirical mean of m samples by $\hat{\mu}_m$ defined as

$$\hat{\mu}_m = \frac{1}{m} \sum_{i=1}^m X_i$$

- ▶ Weak law of large numbers states that $\hat{\mu}_m \rightarrow \mu$ in probability as $m \rightarrow \infty$
- ▶ Strong law of large numbers states that $\hat{\mu}_m \rightarrow \mu$ almost surely as $m \rightarrow \infty$

- ▶ At round m , upon pulling arm a , the agent gets a random reward $r_m^a \sim \mathcal{R}^a$
- ▶ After M rounds, we have $\hat{Q}(a)$ as the empirical mean reward for pulling arm a

$$\hat{Q}(a) = \frac{1}{m} \sum_{i=1}^m r_m^a$$

$$\hat{Q}(a) \rightarrow Q(a)$$

as the number of rounds gets large

Question : Is there a shortcoming to ETC ?

ETC does not use the experience generated after the initial explore phase

Algorithm Greedy Algorithm

```
1: Let  $K$  be the number of arms;  $N$  be the total rounds; Initialize  $M$ 
2: for  $m = 1, 2, \dots, M$  do
3:   for  $a = 1, 2, \dots, K$  do
4:     Pull arm  $a$ ; Observe reward  $r_a$ ; Compute mean reward  $\hat{Q}(a)$  for arm  $a$ ;
5:   end for
6: end for
7: for  $t = MK + 1, \dots, N$  do
8:   Pull the arm with the current best mean reward [i.e.  $a^* = \arg \max_a \hat{Q}(a)$ ]
9:   Update the mean observed rewards with the latest observation
10: end for
```

Question : Will this work well ? Can we improve exploration ?

The greedy algorithm is unlikely to explore during the exploitation phase

Algorithm ϵ - Greedy Algorithm

- 1: Let K be the number of arms; N be the total rounds; Initialize M and choose $\epsilon \in (0, 1)$ small
 - 2: **for** $m = 1, 2, \dots, M$ **do**
 - 3: **for** $a = 1, 2, \dots, K$ **do**
 - 4: Pull arm k ; Observe reward r_a ; Compute mean reward $\hat{Q}(a)$ for arm a ;
 - 5: **end for**
 - 6: **end for**
 - 7: **for** $t = MK + 1, \dots, N$ **do**
 - 8: With probability $1 - \epsilon$, pull the arm with the **current** best mean reward [i.e. $a^* = \arg \max_a \hat{Q}(a)$], else play another arm uniformly at random
 - 9: Update the mean observed rewards with the latest observation
 - 10: **end for**
-

Quesiton : Do you see possible drawback ?

The ϵ -greedy algorithm explores forever. Also, has total linear regret.

- ▶ **Idea** : Initialise $Q(a)$ for all actions to high value
- ▶ Update action value by incremental Monte-Carlo evaluation; Let $a \in \mathcal{A}$ be the arm pulled at round t , Then,

$$\hat{Q}_t(a) = \hat{Q}_{t-1}(a) + \frac{1}{N_t(a)} \left(r_t - \hat{Q}_{t-1}(a) \right)$$

where $r_t \sim \mathcal{R}^a$ is the reward obtained at round t

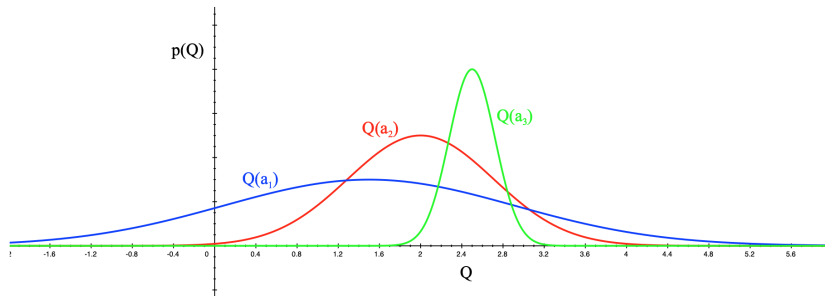
- ▶ Encourages systematic exploration early on
- ▶ Locking onto sub-optimal arm is a possibility
- ▶ Greedy + optimistic initialization has linear total regret
- ▶ ϵ - Greedy + optimistic initialization has linear total regret

Algorithm ϵ - Greedy with Decay Algorithm

- 1: Let K be the number of arms; N be the total rounds; Initialize M and choose $\epsilon \in (0, 1)$ small and choose a small decay rate $r \in (0, 1)$
 - 2: **for** $m = 1, 2, \dots, M$ **do**
 - 3: **for** $a = 1, 2, \dots, K$ **do**
 - 4: Pull arm a ; Observe reward r_a ; Compute mean reward $\hat{Q}(a)$ for arm a ;
 - 5: **end for**
 - 6: **end for**
 - 7: **for** $t = MK + 1, \dots, N$ **do**
 - 8: With probability $1 - \epsilon$, pull the arm with the **current** best mean reward [i.e. $a^* = \arg \max_a \hat{Q}(a)$], else play another arm uniformly at random
 - 9: Update the mean observed rewards with the latest observation
 - 10: Reduce ϵ by fraction r
 - 11: **end for**
-

Certain choices of decay schedule can achieve logarithmic asymptotic total regret

Optimism in the Face of Uncertainty



- ▶ Which arm (among the three) should we choose at next round ?
- ▶ **Optimism in the Face of Uncertainty** \implies pick the arm that we are most uncertain about
- ▶ The more uncertain we are about the action-value of an arm, the more we should explore that action; as it could turn out to be the best action

- ▶ Estimate an upper confidence $\hat{U}_t(a)$ for action a at time t such that

$$Q(a) \leq \hat{Q}_t(a) + \hat{U}_t(a)$$

- ▶ The upper confidence bound depends on the number of times an arm a has been pulled so far

- ★ Small $N_t(a) \implies$ Large $\hat{U}_t(a)$

- ★ Large $N_t(a) \implies$ Small $\hat{U}_t(a)$

- ▶ Select action a , at time t , that maximizes

$$a_t = \arg \max_a \left[\hat{Q}_{t-1}(a) + \hat{U}_{t-1}(a) \right]$$

- ▶ Hoeffding's inequality provides a way to arrive at the formulation for $\hat{U}_t(a)$

Theorem

Let X_1, \dots, X_t be i.i.d. (independent and identically distributed) random variables and they are all bounded by the interval $[0, 1]$. The sample mean is $\bar{X}_t = \frac{1}{t} \sum_{\tau=1}^t X_\tau$. Then for $u > 0$, we have,

$$\mathbb{P}[\mathbb{E}[X] > \bar{X}_t + u] \leq e^{-2tu^2}$$

- We will apply Hoeffding's inequality to the rewards of the bandit

$$\mathbb{P}[Q(a) > \hat{Q}_t(a) + \hat{U}_t(a)] \leq e^{-2N_t(a)\hat{U}_t(a)^2}$$

Calculating Upper Confidence Bound

- ▶ Pick a probability p that true value exceeds UCB
- ▶ Now solve for $\hat{U}_t(a)$ by setting

$$p = e^{-2N_t(a)\hat{U}_t(a)^2}$$

then,

$$\hat{U}_t(a) = \sqrt{\frac{-\log p}{2N_t(a)}}$$

- ▶ Reduce p as t^{-4} as we observe more rewards
- ▶ Ensures optimal action selection asymptotically (as $t \rightarrow \infty$)

$$\hat{U}_t(a) = \sqrt{\frac{2 \log t}{N_t(a)}}$$

Algorithm UCB1 Algorithm

- 1: Let K be the number of arms;
- 2: **for** $a = 1, 2, \dots, K$ **do**
- 3: Pull arm a ; Observe reward r_a ; Compute mean reward $\hat{Q}(a)$ for arm a ;
- 4: **end for**
- 5: **for** $t = K + 1, \dots, N$ **do**
- 6: Pull arm a such that

$$a_t = \arg \max_a \left[\underbrace{\hat{Q}_t(a)}_{\text{Exploitation}} + \underbrace{\sqrt{\frac{2 \log t}{N_t(a)}}}_{\text{Exploration}} \right]$$

- 7: Update the mean observed rewards and UCB coefficient of the arm chosen
 - 8: **end for**
-

- ▶ So far we have made no assumptions about the reward distribution \mathcal{R} (except bound on rewards)
- ▶ Necessary to make assumptions; Strong assumptions, when made the right way, lead to better algorithms
- ▶ Examples :
 - ★ Bernoulli
 - ★ Gaussian with unknown mean and unit variance
 - ★ Many more ...

- ▶ So far we have made no assumptions about the reward distribution \mathcal{R} (except bound on rewards)
- ▶ Bayesian bandits exploit prior knowledge of rewards, $p[\mathcal{R}]$
- ▶ They compute posterior distribution of rewards $p[\mathcal{R}|h_t]$ where $h_t = \{a_1, r_1, \dots, a_{t-1}, r_{t-1}\}$
- ▶ Use posterior to guide exploration (Bayesian UCB, probability matching)
- ▶ Better performance if prior knowledge is accurate

- ▶ Assume reward distribution is Gaussian
 - ★ Reward of every arm is given by $\mathcal{N}(\mu_a, \sigma_a)$
- ▶ Upon pulling arm a , observe reward r_a ; Compute posterior using Baye's law
- ▶ Pick arm a that maximizes standard deviation of $\hat{Q}_t(a)$

$$a_t = \arg \max_a \left[\underbrace{\mu_{t,a}}_{\text{Exploitation}} + \underbrace{\sqrt{\frac{c\sigma_{t,a}}{N_t(a)}}}_{\text{Exploration}} \right]$$

Thompson Sampling

- Consider a Bernoulli bandit

- ★ Each one of the K machines has a probability θ_k of providing a reward to the player

Let us consider a single Bernoulli bandit with probability θ of obtaining a reward

- Suppose R be the random variable that denotes the outcome of pulling the arm of a bandit
- ★ $\mathbb{P}(R = 1) = \theta$ and $\mathbb{P}(R = 0) = 1 - \theta$
- ★ The probability mass function can be written as

$$\mathbb{P}(R = r) = \theta^r (1 - \theta)^{1-r}$$

- ★ The expected reward after one round is given by $\mathbb{E}(R) = \theta$

Let R_1, R_2, \dots, R_n be outcomes of n rounds of pulling the bandit arm

- ▶ **Frequentist approach** : Estimate the fixed but unknown parameter θ using the average of R_1, \dots, R_n for large n
- ▶ **Bayesian approach** : Treat θ as an uncertain parameter, and estimate its distribution from the data $D_n = \{R_1, \dots, R_n\}$ by computing the posterior distribution using Baye's formula

$$\mathbb{P}(\theta|D_n) = \frac{\mathbb{P}(D_n|\theta) \cdot \eta(\theta)}{\mathbb{P}(D)}$$

where $\eta(\theta)$ is a suitable prior distribution on θ

A suitable prior distribution for a Bernoulli bandit is uniform prior

- Suppose we take a uniform prior, then,

$$\mathbb{P}(\theta|D_n) = \underbrace{c\theta^{S_n}(1-\theta)^{n-S_n}}_{\text{Beta Distribution}}$$

with $S_n = R_1 + R_2 + \dots + R_n$

- The posterior $c\theta^{S_n}(1-\theta)^{n-S_n}$ is of the form that resembles Beta distribution with parameters α and γ given by

$$\beta_{\alpha,\gamma}(\theta) = \frac{\Gamma(\alpha + \gamma)}{\Gamma(\alpha)\Gamma(\gamma)} \theta^{\alpha-1} \cdot (1-\theta)^{\gamma-1}$$

- Note that $\beta_{1,1}$ is a uniform distribution
- Initialize the Beta parameters α and β such that prior is uniform
 - ★ $\alpha = 1$ and $\gamma = 1$; we expect the reward probability to be 50% (uniform prior)
 - ★ $\alpha = 9000$ and $\gamma = 1000$; we strongly believe that the reward probability is 90% (not a recommended choice for prior)

- ▶ Assuming uniform prior, after n rounds, we have, $\theta|D_n \sim \beta_{\alpha_{n+1}, \gamma_{n+1}}$
- ▶ Recursive posterior updates :
 - ★ If $\theta|D_n \sim \beta_{\alpha_n, \gamma_n}$ then $\theta|D_{n+1} \sim \beta_{\alpha_{n+1}, \gamma_{n+1}}$ with

$$\begin{aligned}\alpha_{n+1} &= \alpha_n + R_{n+1} \\ \gamma_{n+1} &= \gamma_n + (1 - R_{n+1})\end{aligned}$$

Algorithm Thompson Sampling Algorithm

- 1: Let K be the number of arms;
- 2: **for** $t = 1, \dots, N$ **do**
- 3: **for** $a = 1, 2, \dots, K$ **do**
- 4: Sample θ_t^a from its posterior; $\theta_t^a \sim \beta_{\alpha_t^a, \gamma_t^a}$
- 5: **end for**
- 6: Play the arm $a^* = \arg \max_a \theta_t^a$ and observe the reward R_t
- 7: Update the posterior of the chosen arm by updating the parameters of the corresponding Beta distribution

$$\begin{aligned}\alpha_{t+1}^{a^*} &= \alpha_t^{a^*} + R_t \\ \gamma_{t+1}^{a^*} &= \gamma_t^{a^*} + (1 - R_t)\end{aligned}$$

8: **end for**

- ▶ The exploration techniques mentioned here can easily be extended to full reinforcement learning setting
- ▶ There are other variants of bandit problems that include **Best arm identification**, **PAC** and **Contextual Bandits**
 - ★ PAC : find an arm within ϵ of the best arm with probability at least $1 - \delta$
- ▶ Information state space approach involves modelling the arm selection problem as an MDP with state comprising of history (h_t) of past decisions and rewards. Subsequently, use model free RL or Bayesian RL to solve the MDP