バンディットアルゴリズムのメモ

nakajmiya

January 5, 2026

1 問題設定

バンディット問題とは，選択肢の集合から 1 つを選択し，その選択による報酬を観測が，ほかの選択肢の報酬を観測することなく，報酬和を最大化する問題である．例えば，カジノに来て，スロットマシンで儲けたいと考えているとする．このとき，スロットマシンの選択肢は $K = 5$ 台あるが，それぞれのスロットマシンの報酬はわからない．合計 100 回スロットマシンのアームを引くことができるとして，各回に引くアームをどのように選択していくか考える．100 回のうち，各アームにおいて，最初の n 回は引いて，一番当たりの多かったとわかったアームを残りの $(100 - 5n)$ 回引く，といった方策を考えることができる．もし仮に n が小さく探索が不十分であれば，当たりやすいアームを選択することが難しくなる．一方で，n が大きすぎれば，当たりやすそうなアームの見当がつけられそうだが，最終的に得られる報酬和を最大化することが難しくなる．このように，どのスロットマシンを選択するかを決定する問題がバンディット問題である．特に，上の例のような複数ある選択肢から 1 つを選択する問題を多腕バンディット問題（multi-armed Bandit problem）と呼ぶ．

上ではスロットの例を考えたが，様々な現実問題をバンディット問題の枠組みで考えることができる．

1. インターネット広告配信: K 個の広告のうち，いくつかの広告を表示し，表示された広告のうちユーザーがクリックする広告の数やその利益を最大化する

2. 治験: 次々と訪問してくる患者に対して，K 個の治療法のうちどれを施すかを逐次的に決定し，治療に失敗する患者数の最小化を目指す

3. 推薦システム: 過去の購入履歴に基づいて各ユーザの訪問時に全商品の中から考えたいくつかの商品ペアを推薦し，それらのうち実際に購入される商品数をの最大化を目指す

さらに，バンディット問題は大きく 2 つに分類することができ，各アームからの報酬が何らかの確率分布に従って生成される確率的バンディットとプレイヤーの方策をしっている敵対者が報酬を決める場合を想定する敵対的バンディットの 2 つに分類できる．敵対的バンディットは，プレイヤーの方策を知っているという神のような能力を持つ敵対者が報酬を選ぶと仮定し，その最悪の場合でもうまくいく方策を考える．プレイヤーがランダム性を持たない方策を用いる場合には，プレイヤーの選択アームを敵対者は事前に知ることが可能であるため，プレイヤーは確率的な方策を用いる必要がある．一旦は導入として，確率的バンディット問題を考えることにする．

2 プレイヤー方策の評価法

バンディット問題において，アーム i の時刻 t における報酬 $X_i(t)$ は有界であるとする．時刻 t にプレイヤーが選ぶアームを $i(t)$ としたとき，目標としては，以下の 2 つの指標を最大化することが考えられる．

1. 有限時間区間における累積報酬:

$$\sum_{t=1}^{T} X_{i(t)}(t) \tag{1}$$

2. 無限時間区間における幾何割引された累積報酬:

$$\sum_{t=1}^{\infty} \gamma^{t-1} X_{i(t)}(t) \tag{2}$$

最近だと，有限時間区間における累積報酬を取り扱うことが多い．

さて，プレイヤーの目的はこれらの累積報酬を最大化することであったが，このような累積報酬の大小は，方策の良し悪しのみのみならず，報酬 $\{X_{i(t)}\}_{i,t}$ の大小にも依存する．そこで，純粋な方策の良さを評価するために，なんらかの意味で最適な方策をとった場合の累積報酬を目標値として，そえとの差を比較する．ここで到達しうる累積報酬の最大値は，最適なアームを選択し続けた場合の累積報酬，すなわち，

$$\sum_{t=1}^{T} \max_{\in[K]} X_i(t) \tag{3}$$

である．しかしこれは目標値としては高すぎるので，同じ選択肢を選び続けた場合の累積報酬の最大値，すなわち，

$$\max_{\in[K]} \sum_{t=1}^{T} X_i(t) \tag{4}$$

を目標とする．この目標値とプレイヤーの累積報酬

$$\sum_{t=1}^{T} X_{i(t)}(t) \tag{5}$$

の差を，プレイヤー方策の性能として評価する．これをリグレット（regret）と呼び，次のように定義する．

$$\mathrm{Regret}(T) \coloneqq \max_{\in[K]} \sum_{t=1}^{T} X_i(t) - \sum_{t=1}^{T} X_{i(t)}(t). \tag{6}$$

直感的には,「あのときあのアームを選んでいればよかった」というような後悔の度合いを表している.

各時刻 t におけるプレイヤー方策の選択 $i(t)$ も報酬 $X_i(t)$ も，確率的な場合を扱うことが多いため，リグレットよりも期待リグレット

$$\mathbb{E}\left[\mathrm{Regret}(T)\right] = \mathbb{E}\left[\max_{\in[K]} \sum_{t=1}^{T} X_i(t) - \sum_{t=1}^{T} X_{i(t)}(t)\right] \tag{7}$$

や，さらには擬リグレット（pseudo-Regret）

$$\overline{\mathrm{Regret}}(T) \coloneqq \sum_{t=1}^{T} \left(\max_{\in[K]} X_i(t) - X_{i(t)}(t)\right) \tag{8}$$

を用いた評価が良く用いられる．ここで，擬リグレットは期待リグレット以下の値をとる．すなわち

$$\overline{\mathrm{Regret}}(T) \leq \mathbb{E}\left[\mathrm{Regret}(T)\right] \tag{9}$$

が常に成り立つという性質がある．

3 確率的バンディット問題の方策

K 個のスロットマシンのアームを選んでお金を稼ごうとしているプレイヤーを考える．アーム i を引いたときに得られる報酬の期待値を μ_i とし，その報酬の確率分布を $P \in \mathcal{P}$ で表す．ここで，報酬の確率分布は期待値と一対一に対応付けられているとし，$P_i = P(\mu_i)$ で表されるとする（つまり，なんらかの方法で μ がわかれば，それを特徴づける分布も特定できる）．プレイヤーは，各時刻 $t = 1, 2, \ldots$ において，アーム $i = i(t)$ を選択し，報酬 $X_i(t) \sim P_i$ を得る．

ここで仮にプレイヤーがすべてのアームの真の期待値 μ_i を知っていたとすると，長期的に見て最適な方策は，その最大期待値

$$\mu^\star = \max_{\in[K]} \mu_i \tag{10}$$

を達成することができるアーム

$$i^\star = \arg\max_{\in[K]} \mu_i \tag{11}$$

を引き続けることである．その時刻 T までの累積報酬はの期待値は $\mu^\star T$ となる．一方，実際の進行で時刻 t にアーム $i(t)$ を選択した場合，累積報酬の期待値 $\mu^\star T$ との差は，$\Delta_i \coloneqq \mu^\star - \mu_i$ に対して，以下のように表される．

$$\begin{aligned}
\mathrm{regret}(T) &= \sum_{t=1}^{T} (\mu^\star - \mu_{i(t)}) \\
&= \sum_{i:\mu_i<\mu^\star}^{T} (\mu^\star - \mu_i) N_i(T+1) \\
&= \sum_{i:\mu_i<\mu^\star}^{T} \Delta_i N_i(T+1).
\end{aligned}$$

なお，$N_i(T)$ は，時刻 t の開始時点までにアーム i を引いた回数（つまり，最初の $(t-1)$ 回の選択のうちでアーム i を引いた回数）である．Equation 確率的バンディット問題の文脈においては，これをリグレットと呼ぶ．このリグレットを使った期待リグレット

$$\mathbb{E}\left[\mathrm{regret}(T)\right] = \sum_{i:\mu_i<\mu^\star}^{T} \Delta_i \mathbb{E}\left[N_i(T+1)\right] \tag{12}$$

の最小化を目指していく．

4 ε-貪欲法

このセクションから具体的に確率的バンディット問題で使われる小さなリグレットを達成するための方策の紹介をする．ヘフディングの不等式やチェルノフ・ヘフディングの不等式を適用するために，各アームからの報酬が $X_i(t) \in [0,1]$ になっているものを考える．

また以下では，アーム i からの報酬の標本平均を $\hat{\mu}_i$ で表し，特に時刻 t の開始時点での標本平均であることを明確にする場合は，$\hat{\mu}_i(t)$，すなわち

$$\hat{\mu}_i(t) = \frac{1}{N_i(t)} \sum_{s\in[t-1]:i(s)=i} X_{i(s)} \tag{13}$$

と表記する．また，アーム i を n 回引いた時点でのアーム i の標本平均を $\hat{\mu}_{i,n}$ と表記する（$\hat{\mu}_i(t) = \hat{\mu}_{i,N_i(t)}$．標本平均が最大のアームは，$\hat{i^\star} = \hat{i^\star}(t) = \arg\max_i \hat{\mu}_i(t)$ と表す.

この問題に対して，最も単純な方策が ε-貪欲法である．ε-貪欲法では，全体のアーム選択数 T のうち，εT 回を探索期間とし，残りの $(1-\varepsilon)T$ 回を活用期間とし，それまでに標本平均が高かったアーム $\hat{i^\star}$ を引き続けるものである．ε-貪欲法は，以下のアルゴリズムで表される．

Algorithm 1 ε-貪欲法

Require: 全体のアーム選択数 T, パラメータ $\varepsilon > 0$
Ensure: プレイヤーが選択した方策における累積報酬
1: 各アームの報酬の標本平均 $\hat{\mu}_i$ を初期化する.
2: K 個すべてのアーム i を $\varepsilon T/K$ 回ずつ引く.
3: アーム $\hat{i^\star} = \arg\max_i \hat{\mu}_i(\varepsilon T + 1)$ を $(1-\varepsilon)T$ 回引く.

次に紹介する UCB アルゴリズムに比べて性能は悪いものの，実装が容易でシステムに組み込みやすいというメリットがある．もちろん ε-貪欲法のリグレット上界も解析されているが，またの機会に …

5 Softmax 方策

ε-貪欲法のデメリットである「全ての選択肢を等しく評価して探索してしまう」という点を修正したのが Softmax 方策となる．Softmax 方策では，アーム i を探索する確率 p_i を温度パラメータ $\varepsilon > 0$ つきの Sotmax 関数を使って求め，それぞれの腕に対する既知の情報を踏まえた探索を目指す．具体的には，

$$\mathrm{p}_i = \frac{\exp(\hat{\mu}_i/\tau)}{\sum_{j=1}^{K} \exp(\hat{\mu}_j/\tau)} \tag{14}$$

に従ってアーム i を引く．この方法は，ε-貪欲法よりは改善されている一方で，どの程度のアームが選択されたかという情報を考慮していない．例えば，100 回選んで 50 回当たりが出た選択肢と 2 回選んで 1 回当たりが出た選択肢を同程度に評価している．

そこで，試行回数が多くなるにつれて温度パラメータである ε を小さくすることにより試行回数が少ないときはよりランダムに探索し，試行回数が多くなるにつれて確実に既知情報により良いとされる選択肢を選ぶように修正することも可能である．これをアニールと呼ぶ．

6 UCB 方策

報酬を最大化するためには，現在のところ報酬期待値が高そうに見えるアームを選択することが重要である．その一方で，選択数が少ないアームについては，まだ標本平均が真の期待値に収束していないという可能性がある．これらのバランスをとり理論限界を達成する方法として，古くから知られているのが UCB（Upper Confidence Bound ）方策である．UCB 方策のスコアの取り方は様々な種類があるが，ここではヘフディングの不等式に基づいたものを紹介する．この方策は，UCB スコアとして，

$$\mathrm{UCB}_i(t) = \hat{\mu}_i(t) + \sqrt{\frac{\log t}{2N_i(t)}} \tag{15}$$

を用い，これは標本平均 $\hat{\mu}_i(t)$ に補正項 $\sqrt{\frac{\log t}{2N_i(t)}}$ を加えたものになっている．補正項の役割は先に説明した通り，選択数 $N_i(t)$ が少ないアームほど大きな値をとり，標本数が少なく標本平均が小さいとしても，選択数が少ないアームを選択する確率を高めることができる．では，この式はどのように導かれるのかを説明する．まずはじめにヘフディングの不等式を思い出す．

Theorem 1 (ヘフディングの不等式)**.** $X_1, \ldots, X_n$ を独立な確率変数とし，$X_i \in [a, b]$ を満たすとする．このとき，任意の $\varepsilon > 0$ に対して，

$$\mathbb{P}\left(\mu - \bar{X}_n \geq \varepsilon\right) \leq \exp\left(-\frac{2n\varepsilon^2}{(b-a)^2}\right), \tag{16}$$

$$\mathbb{P}\left(\mu - \bar{X}_n \geq -\varepsilon\right) \leq \exp\left(-\frac{2n\varepsilon^2}{(b-a)^2}\right) \tag{17}$$

が成り立つ．上の 2 つをまとめて，以下も成り立つ．

$$\mathbb{P}\left(\left|\mu - \bar{X}_n\right| \geq \varepsilon\right) \leq 2\exp\left(-\frac{2n\varepsilon^2}{(b-a)^2}\right). \tag{18}$$

ヘフディングの不等式を利用すれば，アーム i の報酬の標本平均 $\hat{\mu}_i(t)$ が真の期待値 μ_i から ε 以上離れる確率は，以下のように評価できる．

$$\mathbb{P}\left(\mu_i - \hat{\mu}_i(t) \geq \varepsilon\right) \leq \exp\left(-2N_i(t)\varepsilon^2\right) =: \delta. \tag{19}$$

ここで，δ を使って ε について整理すると，

$$\varepsilon = \sqrt{\frac{-\log(\delta)}{2N_i(t)}} \tag{20}$$

となる．よって，

$$\begin{aligned} &\mathbb{P}\left(\mu_i - \hat{\mu}_i(t) \geq \sqrt{-\frac{\log(\delta)}{2N_i(t)}}\right) \leq \delta \\ &\Longrightarrow \mathbb{P}\left(\mu_i \leq \hat{\mu}_i(t) + \sqrt{-\frac{\log(\delta)}{2N_i(t)}}\right) \geq 1 - \delta \end{aligned} \tag{21}$$

とできる．このことから，UCB スコアはアーム i の報酬の期待値の上側 $1-\delta$ 信頼区間ということになる．ここで，$\delta = 1/t$ とおけば所与の式

$$\mathbb{P}\left(\mu_i \leq \hat{\mu}_i(t) + \sqrt{\frac{\log t}{2N_i(t)}}\right) \geq 1 - \frac{1}{t} \tag{22}$$

が得られる．つまり，UCB アルゴリズムの各時刻 t において，報酬の期待値の上側 $1 - 1/t$ 信頼区間が最大となるアーム i を選ぶようなアルゴリズムになっている．アルゴリズムは以下のようになる．

Algorithm 2 UCB アルゴリズム

Require: 全体のアーム選択数 T
Ensure: プレイヤーが選択した方策における累積報酬

1: すべてのアームを 1 回ずつ引く．
2: $K \leftarrow$ アームの数
3: **for** $t = 1, 2, \ldots, T$ **do**
4: 　各アーム $i \in [K]$ に対して，$\mathrm{UCB}_i(t) = \hat{\mu}_i(t) + \sqrt{\frac{\log t}{2N_i(t)}}$ を計算する．
5: 　アーム $i = \arg\max_{i \in [K]} \mathrm{UCB}_i(t)$ を引く（スコアが同じアームが複数ある場合は任意に選ぶ）．
6: **end for**

7 トンプソン抽出

トンプソン抽出では，期待値パラメータ μ_i が何らかの事前分布 $\pi_i(\mu_i)$ から生成されていると考える．この事前分布はプレイヤーが自由に設定することができるが，例えばベルヌーイ分布モデル $\{\mathrm{Ber}(\mu) : \mu \in [0, 1]\}$ では，共役事前分布がベータ分布 $\mathrm{Beta}(\alpha, \beta)$ であるため，これを用いるのが一般的である．ベルヌーイ分布に従う報酬の期待値の事後分布について計算してみる．アーム i を N_i 回選択し，そのうち N_i^+ 回当たりが出たとすると，報酬の真の期待値パラメータ $\mu_i \in [0, 1]$ の事後分布は以下のように導出される．

$$\begin{aligned} p(\mu_i | X_{N_i}) &= \frac{p(X_{N_i}|\mu_i)\pi(\mu_i)}{p(X_{N_i})}\pi_i(\mu_i) \\ &\propto p(X_{N_i}|\mu_i)\pi_i(\mu_i) \end{aligned} \tag{23}$$

これはベータ分布 $\mathrm{Beta}(N_i^+ + \alpha, N_i - N_i^+ + \beta)$ に比例する．

8　付録

8.1　区間推定

区間推定とは，真の母数 θ がある区間 (L, U) に入る確率を $1-\alpha$（α は θ が区間に入らない確率）以上になるように保証する方法であり，

$$\mathbb{P}(L \leq \theta \leq U) \geq 1-\alpha \tag{24}$$

となる確率変数 L, U を求めるものである．L, U はそれぞれ，下側信頼限界（lower confidence bound）と上側信頼限界（upper confidence bound）と呼ばれる．$1-\alpha$ は信頼係数と呼び，$100(1-\alpha)\%$ 信頼区間と呼ぶ．$1-\alpha$ は目的に応じた適当な量を選ぶが，通常は 0.95 や 0.99 が用いられる．