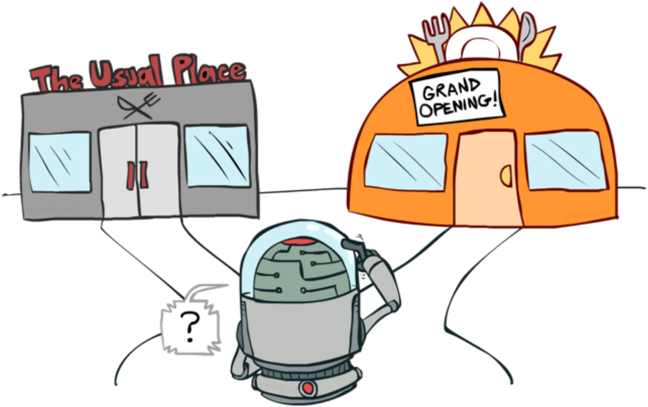
Thompson sampling

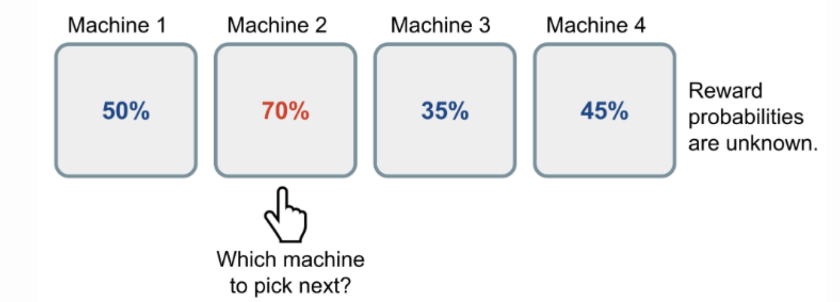
Salim Amoukou

April 19, 2019

Motivation : Selection or exploration ?



Bandit problem



- Constraint of exploring or exploiting.

The simplest strategy

- Greedy strategy : Select the arm with the highest average so far.

The simplest strategy

- Greedy strategy : Select the arm with the highest average so far.

- But it will definitely fail : By bad-luck :(

The simplest strategy

- Greedy strategy : Select the arm with the highest average so far.

- But it will definitely fail : By bad-luck :(
  - We need some exploration.

`Let be little smart...`

ϵ-greedy

ϵ-greedy

ϵ-greedy:

- Select an arm at random with probability ϵ, otherwise do greedy.

ϵ-greedy

ϵ-greedy:
- Select an arm at random with probability ϵ, otherwise do greedy.

Theoritical guarentees :

- If ϵ is constant:
  - For large enough t : $\mathbb{P}(a_t \neq a) \approx \epsilon$
    - we have **linear regret**
- If $\epsilon \propto \frac{1}{t}$:
  - For large enough t :$\mathbb{P}(a_t \neq a) \approx \epsilon_t$
    - we have **logarithme regret**

Empirical mean

Question : How far is the empirical mean $\hat{R}(a)$ from the true mean R(a) ?

If we known: $|R(a) - \hat{R}(a)| \leq bound$ then :

- $R(a) \leq \hat{R}(a) + bound$
- we could select the arm with the highest bound

Empirical mean

Question : How far is the empirical mean $\hat{R}(a)$ from the true mean R(a) ?

If we known: $|R(a) - \hat{R}(a)| \leq bound$ then :

- $R(a) \leq \hat{R}(a) + bound$
- we could select the arm with the highest bound

Idea : Overtime, $\hat{R}(a)$ will be more precise and bound tighter.

The best case :

Optimistic strategy :

- Assume that we have an oracle $UB_n(a)$ that returns an upper bound on R(a) for each arm a
- And $\lim_{n \to \infty} UB_n(a) = R(a)$, then :

The best case :

Optimistic strategy :

- Assume that we have an oracle $UB_n(a)$ that returns an upper bound on R(a) for each arm a
- And $\lim\limits_{n \to \infty} UB_n(a) = R(a)$, then :

Theorem

Optimistic strategy that selects $argmax_a UB_n(a)$ will converge to a^*

The best case :

Optimistic strategy :

- Assume that we have an oracle $UB_n(a)$ that returns an upper bound on R(a) for each arm a
- And $\lim_{n\to\infty} UB_n(a) = R(a)$, then :

Theorem

Optimistic strategy that selects $argmax_a UB_n(a)$ will converge to a^*

Proof.

Suppose that it converges to sub optimal a, then:
By contradiction, $R(a) = UB_\infty(a) \geq UB_\infty(a') = R(a') \quad \forall a'$ $\square$

How to find an optimistic upper bound ?

How to find an optimistic upper bound ?

- Unless we assume prior known ledge over reward distribution
  - Ex : Gaussian distribution

How to find an optimistic upper bound ?

- Unless we assume prior known ledge over reward distribution
  - Ex : Gaussian distribution
- Solution: Estimate a bound $\hat{U}_t(a)$ for each action such that : value
  - $R(a) \leq \hat{R}(a) + \hat{U}_t(a)$ with high probability
  - Depend on $N_t(a)$:
    - Small $N_t(a) \Rightarrow$ large $\hat{U}_t(a)$ (estimate value is uncertain)
    - Large $N_t(a) \Rightarrow$ Small $\hat{U}_t(a)$ (estimate value is certain)

How to find an optimistic upper bound ?

- Unless we assume prior known ledge over reward distribution
  - Ex : Gaussian distribution
- Solution: Estimate a bound $\hat{U}_t(a)$ for each action such that : value
  - $R(a) \leq \hat{R}(a) + \hat{U}_t(a)$ with high probability
  - Depend on $N_t(a)$:
    - Small $N_t(a) \Rightarrow$ large $\hat{U}_t(a)$ (estimate value is uncertain)
    - Large $N_t(a) \Rightarrow$ Small $\hat{U}_t(a)$ (estimate value is certain)

Use Hoeffding inequality:

$$\mathbb{P}[\mathbb{E}(\mathbb{X}) > \hat{X}_n + \mu] \leq e^{2t\mu^2}$$

We can choose $U_t(a) = \sqrt{\frac{2log\epsilon}{N_t(a)}}$

UCB algorithm

UCB(h)

$V \leftarrow 0,\ n \leftarrow 0,\ n_a \leftarrow 0 \quad \forall a$

Repeat until $n = h$

Execute $\mathrm{argmax_a}\ \tilde{R}(a) + \sqrt{\frac{2 \log\ n}{n_a}}$

Receive r

$V \leftarrow V + r$

$\tilde{R}(a) \leftarrow \frac{n_a \tilde{R}(a) + r}{n_a + 1}$

$n \leftarrow n + 1, \quad n_a \leftarrow n_a + 1$

Return V

Put prior knownledge

In the bandit problem, we just have to put a prior on the mean.

Put prior knownledge

In the bandit problem, we just have to put a prior on the mean.

- In the Bernoulli bandit :
  - $\forall$ arm a, $\mathbb{P}(R(a) = 1) = \theta_a$

Put prior knownledge

In the bandit problem, we just have to put a prior on the mean.

- In the Bernoulli bandit :
  - $\forall$ arm a, $\mathbb{P}(R(a) = 1) = \theta_a$
  - Let put Beta prior : $\theta_a \sim \; Beta(\alpha, \beta)$

Put prior knownledge

In the bandit problem, we just have to put a prior on the mean.

- In the Bernoulli bandit :
  - $\forall$ arm a, $\mathbb{P}(R(a) = 1) = \theta_a$
  - Let put Beta prior : $\theta_a \sim \ Beta(\alpha, \beta)$
  - Update prior with the outcome reward:
    - $\mathbb{P}(\theta_a | R_1(a) = 1) \propto \mathbb{P}(\theta_a) \times \mathbb{P}(R_1(a) = 1 | \theta_a) = \theta_a^{\alpha}(1 - \theta_a)^{\beta - 1}\theta_a$
    - $\theta_a^{pos} \sim Beta(\alpha + 1, \beta)$

Put prior knownledge

In the bandit problem, we just have to put a prior on the mean.

- In the Bernoulli bandit :
  - $\forall$ arm a, $\mathbb{P}(R(a) = 1) = \theta_a$
  - Let put Beta prior : $\theta_a \sim \ Beta(\alpha, \beta)$
  - Update prior with the outcome reward:
    - $\mathbb{P}(\theta_a | R_1(a) = 1) \propto \mathbb{P}(\theta_a) \times \mathbb{P}(R_1(a) = 1 | \theta_a) = \theta_a^{\alpha}(1 - \theta_a)^{\beta - 1}\theta_a$
    - $\theta_a^{pos} \sim Beta(\alpha + 1, \beta)$

Idea : the posterior becomes more and more peak to the real parameter

Put prior knownledge

In the bandit problem, we just have to put a prior on the mean.

- In the Bernoulli bandit :
  - $\forall$ arm a, $\mathbb{P}(R(a) = 1) = \theta_a$
  - Let put Beta prior : $\theta_a \sim \ Beta(\alpha, \beta)$
  - Update prior with the outcome reward:
    - $\mathbb{P}(\theta_a | R_1(a) = 1) \propto \mathbb{P}(\theta_a) \times \mathbb{P}(R_1(a) = 1 | \theta_a) = \theta_a^{\alpha}(1 - \theta_a)^{\beta - 1}\theta_a$
    - $\theta_a^{pos} \sim Beta(\alpha + 1, \beta)$

Idea : the posterior becomes more and more peak to the real parameter

Remark : Posterior distribution is not always easy to compute.

Thompson sampling

ThompsonSampling(h)
$V \leftarrow 0$
For $n = 1$ to h
Sample $R_1(a), \ldots, R_k(a) \sim \Pr(R(a)) \;\; \forall a$
$\hat{R}(a) \leftarrow \frac{1}{k}\sum_{i=1}^{k} R_i(a) \quad \forall a$
$a^* \leftarrow \mathrm{argmax}_a \, \hat{R}(a)$
Execute a^* and receive r
$V \leftarrow V + r$
Update $\Pr(R(a^*))$ based on r
Return V

SIMULATION TIME !

Application of bandit

In many learning applications, true labels are not fully available.

- Ex: System recommendation

Application of bandit

In many learning applications, true labels are not fully available.

- Ex: System recommendation

This leads to an online multiclass setting with limited feedback.

Application of bandit

In many learning applications, true labels are not fully available.

- Ex: System recommendation

This leads to an online multiclass setting with limited feedback.

- Is there an efficient learner (with guarantees) in this case ?

The simplest online algorithm: Perceptron

Online algorithm for binary classification :

Algorithm Perceptron

---

set $w^1 := 0$

for $t = 1, 2, \ldots$

receive example x_t

predict label $\hat{y}_t := \text{sign}(w^t \cdot x_t)$

nature reveals the label y_t

update weight $w^{t+1} := w^t + u^t$,

where $u^t := x_t(\mathbb{1}[y_t = 1] - \mathbb{1}[\hat{y}_t = 1])$

The simplest online algorithm: Perceptron

Online algorithm for binary classification :

Algorithm Perceptron

---

set $w^1 := 0$

for $t = 1, 2, \ldots$

  receive example x_t

  predict label $\hat{y}_t := \mathrm{sign}(w^t \cdot x_t)$

  nature reveals the label y_t

  update weight $w^{t+1} := w^t + u^t$,

    where $u^t := x_t(\mathbb{1}[y_t = 1] - \mathbb{1}[\hat{y}_t = 1])$

One can show that the number of mistakes is finite if the data is linearly separable.

Perceptron for multiclass

Algorithm k-class Perceptron

set $W^1 := 0$ ($W^t = [w_1^t, \ldots, w_k^t]^\top$)
for $t = 1, 2, \ldots$
 receive example x_t
 predict label $\hat{y}_t := \arg\max_j (W^t x_t)_j$
 nature reveals the label y_t
 update weight $W^{t+1} := W^t + U^t$,
  where $U^t_{r,j} := x_{t,j}(\mathbb{1}[y_t = r] - \mathbb{1}[\hat{y}_t = r])$

Let go back to our problem

In partial information case : nature reveal $1_{\{y_t=\hat{y}_t\}}$ not y_t
Challenging :

- Cannot use perceptron update
- Cannot use bandit in online convexe optimization

Let go back to our problem

In partial information case : nature reveal $1_{\{y_t=\hat{y}_t\}}$ not y_t
Challenging :

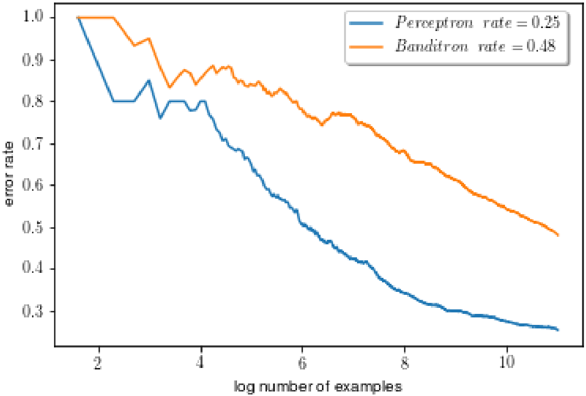
- Cannot use perceptron update
- Cannot use bandit in online convexe optimization

Banditron to the rescue

Algorithm The Banditron

Parameters: $\gamma \in (0, 0.5)$
Initialize $W^1 = \mathbf{0} \in \mathbb{R}^{k \times d}$
for $t = 1, 2, \ldots, T$ **do**
  Receive $\mathbf{x}_t \in \mathbb{R}^d$
  Set $\hat{y}_t = \arg\max_{r \in [k]} (W^t \mathbf{x}_t)_r$
  $\forall r \in [k]$ define $P(r) = (1 - \gamma)\mathbf{1}[r = \hat{y}_t] + \frac{\gamma}{k}$
  Randomly sample $\tilde{y}_t$ according to P
  Predict $\tilde{y}_t$ and receive feedback $\mathbf{1}[\tilde{y}_t = y_t]$
  Define $\tilde{U}^t \in \mathbb{R}^{k \times d}$ such that:
    $\tilde{U}^t_{r,j} = x_{t,j} \left(\frac{\mathbf{1}[y_t = \tilde{y}_t]\mathbf{1}[\tilde{y}_t = r]}{P(r)} - \mathbf{1}[\hat{y}_t = r] \right)$
  Update: $W^{t+1} = W^t + \tilde{U}^t$
end for

Banditron vs Perceptron on MNIST



Conclusion :

- Limit of Banditron
- Limit of Thompson sampling

Remerciements : Merci beaucoup à **Raphael Cousin !!**

Sources :

- Tutorial on thompson sampling [Daniel J. Russo1 , Benjamin Van Roy , Abbas Kazerouni , Ian Osband and Zheng We4]
- An Empirical Evaluation of Thompson Sampling [Olivier Chapelle, Lihong Li]