Upper Confidence Bound (UCB)

Sunday, December 15, 2024 11:25 AM

Reinforcement learning
- Can be used to make robot dogs walk
- Train the dog to walk through trial and error

The Multi-Armed Bandit Problem
- A one armed bandit is a slot machine
    - Bandit because it is a very quick way to make money
- Multi-Armed Bandit
    - 5 or 10 slot machines
        - How to play them to maximize your return
    - Assumption
        - Each machine has a distribution of numbers that represent results
    - Our goal is to figure out which has the best distribution
        - The longer you take to figure out the more money you lose
    - Regret (Quantifiable)
        - Suffer when you don't use the optimal method

- Given multiple ads, how to conclude which ad is the best
    - Find out the best ad, in the process of exploiting the best one

The Multi-Armed Bandit Problem

- We have d arms. For example, arms are ads that we display to users each time they connect to a web page.
- Each time a user connects to this web page, that makes a round.
- At each round n, we choose one ad to display to the user.
- At each round n, ad i gives reward $r_i(n) \in \{0, 1\}$: $r_i(n) = 1$ if the user clicked on the ad i, 0 if the user didn't.
- Our goal is to maximize the total reward we get over many rounds.

Upper Confidence Bound
- STEP 1: At each round n, we consider two number for each ad i:
    - Ni (n) - the number of times the ad I was selected up to round n
    - Ri (n) - The sum of rewards of the ad I up to round n
- STEP 2: From these two numbers we compute:
    - The average reward of ad i up to round n
        - $\bar{r}_i(n) = \frac{R_i(n)}{N_i(n)}$

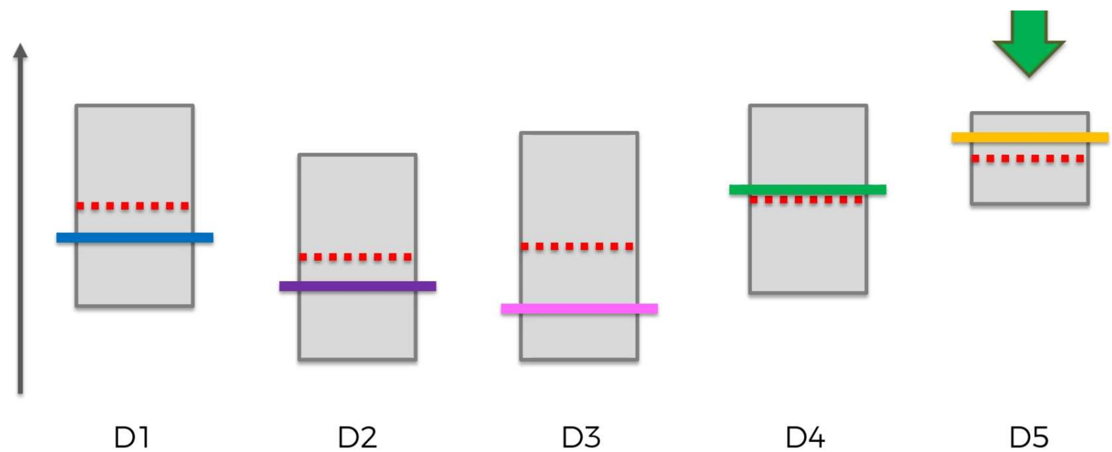
- the confidence interval $[\bar{r}_i(n) - \Delta_i(n), \bar{r}_i(n) + \Delta_i(n)]$ at round n with
  - $$\Delta_i(n) = \sqrt{\frac{3}{2}\frac{\log(n)}{N_i(n)}}$$
- STEP 3: We select the ad i that has the maximum

$$\text{UCB } \bar{r}_i(n) + \Delta_i(n).$$

How does it work?

- We want to find the best distribution
- We first start by assuming that each distribution has a starting point
  - This also includes a confidence band, that the expected value of distribution lies in the confidence band
- If the user did not click on the add, the point moves down and the confidence band shrinks a little
  - Generally. Based on probability
- If the user did click on the add, the point moves up and the confidence band shrinks a little
  - Generally. Based on probability
- Keep doing this for all arms

- Repeat this process, while always taking the arm with the highest confidence band



This is a deterministic algorithm

Thompson Sampling Algorithm

Sunday, December 15, 2024 1:24 PM

Bayesian Inference

- Ad i gets rewards $\boldsymbol{y}$ from Bernoulli distribution $p(\boldsymbol{y}|\theta_i) \sim \mathcal{B}(\theta_i)$.
- θ_i is unknown but we set its uncertainty by assuming it has a uniform distribution $p(\theta_i) \sim \mathcal{U}([0,1])$, which is the prior distribution.
- Bayes Rule: we approach θ_i by the posterior distribution

- $$\underbrace{p(\theta_i|\mathbf{y})}_{\text{posterior distribution}} = \frac{p(\mathbf{y}|\theta_i)p(\theta_i)}{\int p(\mathbf{y}|\theta_i)p(\theta_i)d\theta_i} \propto \underbrace{p(\mathbf{y}|\theta_i)}_{\text{likelihood function}} \times \underbrace{p(\theta_i)}_{\text{prior distribution}}$$

- We get $p(\theta_i|\mathbf{y}) \sim \beta(\text{number of successes} + 1, \text{number of failures} + 1)$
- At each round n we take a random draw $\theta_i(n)$ from this posterior distribution $p(\theta_i|\mathbf{y})$, for each ad i.
- At each round n we select the ad i that has the highest $\theta_i(n)$.

STEP 1: At each round n, we consider two numbers for each ad i:

- Ni1 (n) - The number of times the ad I got reward 1 up to round n
- Ni0 (n) - the number of times the ad I got reward 0 up to round n

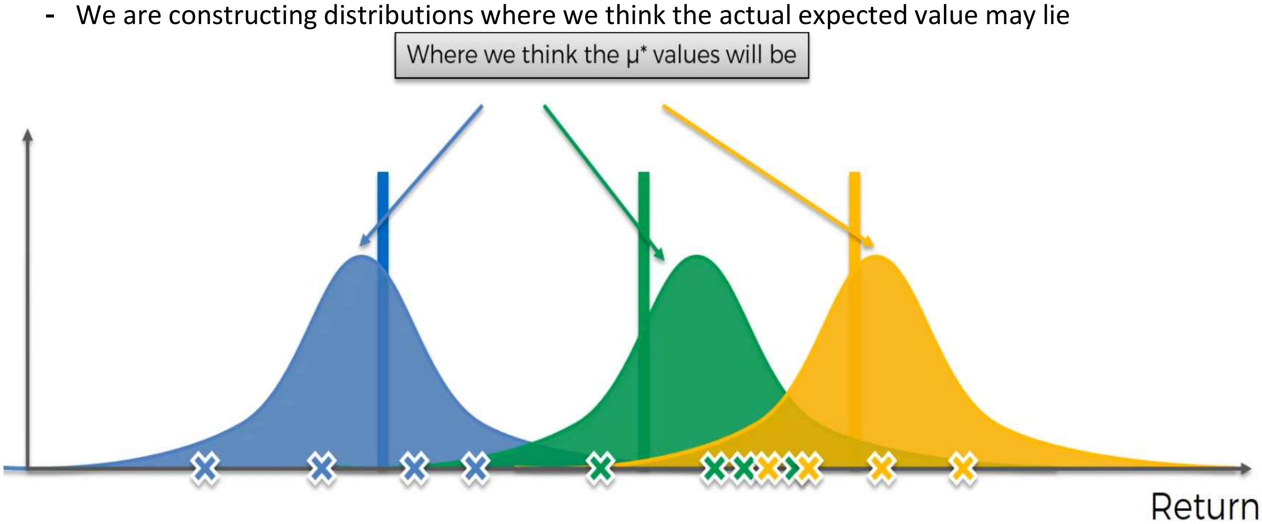
STEP 2: For each ad I, we take a random draw from the distribution:

$$\theta_i(n) = \beta(N_i^1(n) + 1, N_i^0(n) + 1)$$

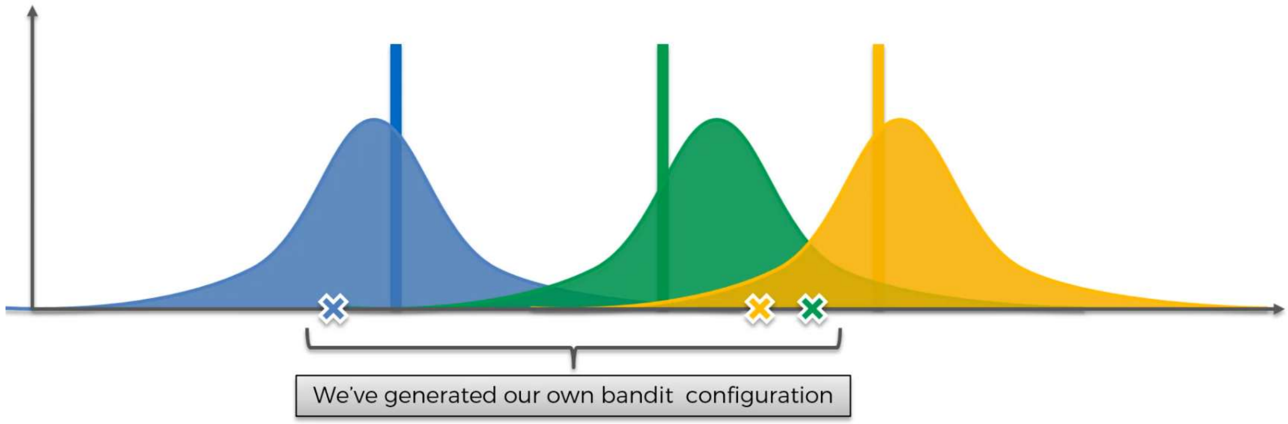
STEP 3: We select the ad that had the highest theta i (n)

We aren't trying to guess the actual distribution of each arm

- We are constructing distributions where we think the actual expected value may lie



By selecting random points of each arm, we've generated our own bandit configuration



- The green arm had the highest probability in this case
- We will select this arm to pull, but the probability will be less, to the green distribution will be less

This is a probabilistic algorithm

UCB vs Thompson Sampling

- UCB is deterministic
- Thompson is probabilistic
- UCB requires an update at every round
- Thompson can accommodate delayed feedback
    - Runs in a batch matter
- Thompson has better empirical evidence