Stochastic Search and Optimisation
Multi-armed Bandits

And Jesus went into the temple of God, and cast out all them that sold and bought in the temple, and overthrew the tables of the money changers, and the seats of them that sold doves,

Matthew 21:12

Introduction

The Multi-armed Bandit (MAB) problem is something of a test-problem, and serves as an introduction to more general Markov Decision Processes.

We can think of a MAB as a sequential gambling game (the name refers to slot machines, or “one-armed bandits”). Each turn we have to choose one of n options/bandits, where each choice comes with a different random reward. The goal is to maximise the overall/long-term reward, which requires a balance of effort between *exploration* and *exploitation*.

There are direct applications of MABs to clinical trials; ad placement on web pages; packet routing; Nash equilibria; etc.

Definition

Let the reward from bandit i be independent each time, and independent of the other bandits, with a Bernoulli distribution mean θ_i. Let $\mathbf{i} = (i(1), i(2), \ldots, i(N))$ be the sequence of decisions made, and let $X(t)$ be the return from the t-th decision, so

$$X(t) \sim \text{Bernoulli}(\theta_{i(t)}).$$

Our goal is to choose $\mathbf{i}$ to maximise the expected total return

$$R = R(N) = \sum_{t=1}^{N} a^t \theta_{i(t)}.$$

Here $a \in (0, 1]$ is a discounting factor, and N may be infinite (in which case we require $a < 1$ to avoid $R = \infty$).

$i(t)$ may depend on past choices $i(1), \ldots, i(t-1)$ and outcomes $X_{i(1)}, \ldots, X_{i(t-1)}$, but can not depend on future outcomes.

Exercise

mab_sim.r

```
# multi-armed bandit simulator
# 3 machines, each return the same amount,
# but with different chances of winning
# you get 100 plays: maximise the number of wins

set.seed(3)
p <- runif(3) # do not look at p until finished!

wins <- 0
tries <- 0
while (tries < 100) {
  cat("enter choice of machine and number of tries")
  input <- scan(n = 2, quiet = TRUE)
  k <- input[1]
  n <- min(input[2], 100 - tries)
  wins <- wins + sum(runif(n) < p[k])
  tries <- tries + n
  cat(wins, "wins from", tries, "tries\n")
}
```

Bayesian framework

Bayesian statistics provide a way of modeling how much we know about each bandit.

Lemma If $\theta \sim \text{beta}(\alpha, \beta)$ and $X \sim \text{Bernoulli}(\theta)$ then

$$\theta \,|\, X = x \sim \text{beta}(\alpha + x, \beta + 1 - x).$$

Proof

$$\begin{aligned} f_\theta(\theta) &\propto \theta^{\alpha-1}(1-\theta)^{\beta-1} \\ f_{X|\theta}(x|\theta) &\propto \theta^{x}(1-\theta)^{1-x} \\ f_{\theta|X}(\theta|x) &\propto f_{X|\theta}(x|\theta) f_\theta(\theta) \\ &= \theta^{(\alpha+x)-1}(1-\theta)^{(\beta+1-x)-1} \end{aligned}$$

Bayesian framework

Let $\Theta_i(t) \sim \text{beta}(\alpha_i(t), \beta_i(t))$ describe what we know about θ_i after t trials.

Before the first trial we know nothing about θ_i, so for all i

$$\Theta_i(0) \sim \text{beta}(1,1) \sim U(0,1).$$

If we choose arm j on the t-th trial then, given $X(t) = x$,

$$\Theta_j(t) \sim \text{beta}(\alpha_j(t-1) + x, \beta_j(t-1) + 1 - x).$$

That is

$$\begin{aligned} \alpha_j(t) &= 1 + \sum_{s=1}^{t} I_{\{i(s)=j\}} X(s) \\ \beta_j(t) &= 1 + \sum_{s=1}^{t} I_{\{i(s)=j\}} (1 - X(s)) \end{aligned}$$

Bayesian framework

The mean and variance of a beta(α, β) are $\alpha/(\alpha+\beta)$ and $\alpha\beta/((\alpha+\beta)^2(\alpha+\beta+1))$, so as we observe the j-th arm more often, the variance of $\Theta_j(t)$ goes to zero so it converges to a point mass.

Dynamic programming solution

$N < \infty$

For finite N we can use dynamic programming. We illustrate this in the case $n = 2$ and $a = 1$.

Let $R_k(\alpha_1, \beta_1; \alpha_2, \beta_2)$ be the optimal expected return when we know that $\Theta_1(t) \sim \text{beta}(\alpha_1, \beta_1)$, $\Theta_2(t) \sim \text{beta}(\alpha_2, \beta_2)$, and we have k decisions left to make.[1]

If $k = 1$ then we just choose the arm with the largest expected return. That is

$$R_1(\alpha_1, \beta_1; \alpha_2, \beta_2) = \max\left\{\frac{\alpha_1}{\alpha_1 + \beta_1}, \frac{\alpha_2}{\alpha_2 + \beta_2}\right\}.$$

For $k > 1$ we can calculate $R_k(\alpha_1, \beta_1; \alpha_2, \beta_2)$ recursively.

[1] Note that t doesn't matter once we know α_i and β_i.

$$R_k(\alpha_1, \beta_1; \alpha_2, \beta_2) =$$
$$\max \left\{ \frac{\alpha_1}{\alpha_1 + \beta_1} [1 + R_{k-1}(\alpha_1 + 1, \beta_1; \alpha_2, \beta_2)] \right.$$
$$+ \frac{\beta_1}{\alpha_1 + \beta_1} R_{k-1}(\alpha_1, \beta_1 + 1; \alpha_2, \beta_2),$$
$$\frac{\alpha_2}{\alpha_2 + \beta_2} [1 + R_{k-1}(\alpha_1, \beta_1; \alpha_2 + 1, \beta_2)]$$
$$\left. + \frac{\beta_2}{\alpha_2 + \beta_2} R_{k-1}(\alpha_1, \beta_1; \alpha_2, \beta_2 + 1) \right\}$$

Using this, if we calculate $R_1(\alpha_1, \beta_1; \alpha_2, \beta_2)$ for all $\alpha_1 + \beta_1 + \alpha_2 + \beta_2 = (N - 1) + 4$ then we can calculate $R_2(\alpha_1, \beta_1; \alpha_2, \beta_2)$ for all $\alpha_1 + \beta_1 + \alpha_2 + \beta_2 = (N - 2) + 4$, and so on, until we get $R_N(1, 1, 1, 1)$.

Dynamic programming solution

Numerical problems

Unfortunately the dynamic programming approach is impractical for all reasonably sized problems. In the case $n = 2$ there are $O(N^3)$ combinations of $\alpha_1, \beta_1, \alpha_2, \beta_2$ satisfying $\alpha_1 + \beta_1 + \alpha_2 + \beta_2 = (N - 1) + 4$, so to find $R_N(1, 1, 1, 1)$ we need $O(N^4)$ calculations.

This is not too bad perhaps, but for larger n one can show that the number of calculations required is $O(N^{2n})$, which very quickly becomes unmanageable for $n > 2$.

Thus we need alternative approaches.

Gittins Index Theorem

Gittins (1979), Bandit processes and dynamic allocation indices. JRSS B.

Suppose that $N = \infty$ and we have a discount factor $a < 1$. The state of bandit i at time t is defined as the distribution of $\Theta_i(t)$, which is given by $\alpha_i(t)$ and $\beta_i(t)$.

Gittins Index Theorem says that there is some index function ν such that the optimal choice of bandit at time t is the i that maximises $\nu(\Theta_i(t)) = \nu(\alpha_i(t), \beta_i(t))$.

The theorem actually applies to a larger class of problems than the MAB we have considered. See for example Gittins, Glazebrook & Weber (2011), Multi-armed Bandit Allocation Indices, 2nd Edition. Wiley.

Dynamic programming for the index

The index ν does not depend on how much we know about the other bandits (their states). In particular, the index applied to bandit 1 would be the same even if we had perfect information about all the other bandits. That is, even if we knew $\theta_2, \ldots, \theta_n$ exactly.

If we know $\theta_2, \ldots, \theta_n$ exactly then it is clear that the best of these is the largest. Thus the index applied to bandit 1 must behave in the same way as in a two-bandit problem where we know θ_2 exactly.

$n = 2, N = \infty, a < 1$

Let $R(\alpha, \beta; \theta)$ be the maximal return when $\Theta_1 \sim \text{beta}(\alpha, \beta)$ and $\Theta_2 = \theta$, then

$$R(\alpha, \beta; \theta) = \max\left\{\frac{\theta}{1-a}, \frac{\alpha}{\alpha+\beta}[1 + aR(\alpha+1, \beta; \theta)] + \frac{\beta}{\alpha+\beta} aR(\alpha, \beta+1; \theta)\right\}$$

Here $\theta/(1-a)$ is the total (discounted) return from a policy that always chooses bandit 2. If it is optimal to choose bandit 2 this time, then nothing has changed next time because we have no new information about bandit 1, so it will be optimal to choose bandit 2 next time as well.

As $N = \alpha + \beta \to \infty$ we have

$$R(\alpha, \beta; \theta) \to \max\left\{\frac{\alpha}{\alpha+\beta}, \theta\right\}.$$

Thus for any θ we can get arbitrarily good approximations to $R(\alpha, \beta; \theta)$ for all $\alpha, \beta \in \mathbb{N}$, by starting with $\alpha + \beta$ large and working backwards. This requires $O(N^2)$ calculations.

If we do this for a fine grid of θ values, then for any α, β pair we can find the θ for which

$$\frac{\theta}{1-a} = \frac{\alpha}{\alpha+\beta}[1 + aR(\alpha+1, \beta; \theta)] + \frac{\beta}{\alpha+\beta} aR(\alpha, \beta+1; \theta).$$

This tells us that $\Theta_1 \sim \text{beta}(\alpha, \beta)$ is equivalent to $\Theta_2 = \theta$, so we can take

$$\nu(\alpha, \beta) = \theta.$$

Heuristc policies

- Max expected return
- Thompson
- ϵ-Greedy
- Hoeffding upper confidence bound
- Bayes upper credible bound