Multi-Armed Bandit Problem

Sunith Suresh, Lin Xiao, Ilan Man and Sanjay Hariharan

April 25, 2016

1 Introduction

2 Multi-armed bandit review

Intro the MAB problem.
Typically the objective is to maximize the expected total rewards. As such, the goal of the multi-armed bandit problem is to devise a scheme by which one can select the optimal machine to play at each time step.

A related and important concept to reward maximization is the concept of minimizing regret. Define R_N^* and $\hat{R_N}$ as the actual and expected maximum rewards, respectively, after N iterations. Therefore the regret of a given multi-armed bandit scheme is $R_N^* - \hat{R_N}$.

The following sections outline various approaches to solving the MAB problem.

2.1 ϵ-greedy

- Sunith

2.2 upper confidence bounds

- Ilan

- explain concept of bounding regret
- application of Hoeffding's Inequality
- calculate UCB

3 Bayesian approach

3.1 Bernoulli Bandit

We formulate a simple bandit scheme involving N Machines and a Bernoulli Reward (0 or 1). We play one of the N machines at every iteration of the game for T iterations. Each Machine has a latent parameter governing its reward. At every step we draw a reward from the selected machine.

Specifically, for each machine, we generate a probability of getting a reward of 1, θ_i. We assume that θ_i is distributed as a Beta. A natural choice for the likelihood of a reward, r_i is a Bernoulli(θ_i). To be clear:

$$\theta_i \sim \text{Beta}(\alpha_i, \beta_i)$$

$$r_i = \text{Reward from Machine i} \sim \text{Bernoulli}(\theta_i)$$

Exploiting the conjugacy of the Beta-Bernoulli model, the posterior distribution of getting a reward from machine$_i$, after playing for T iterations is:

$$\text{Beta}(\alpha_i + R_T, \beta_i + T - R_T)$$

where $R_T = \sum_{t=1}^{T} r_t$

Before beginning to play, we have no prior knowledge about either Machine's propensity to yield a positive reward. Therefore, we initially set $\alpha_i, \beta_i = 1$, which is a common objective prior for the Beta distribution.

3.2 General algorithm

The generic algorithm is as follows:

1. For t = 1, ..., T
2. U(α_i, β_i) = i_t // select machine i at time t using the policy/acquisition function
3. $r_t \sim$ Bernoulli(θ_i) i // We play Machine i, and get Reward r_t
4. If $r_t = 1 : \alpha_i = \alpha_i + 1$ // if the reward was successful, increase our positive belief in machine i
5. else if $r_t = 0 : \beta_i = \beta_i + 1$ // if we didn't get a reward, increase our negative belief (i.e. failure) in machine i
6. $R_t = R_t + r_t$ // add reward at time t to running total

This algorithm provides a framework for how to think about this problem in a general way. To maximize expected rewards, we must optimally choose the acquisition function U. The remaning sections discuss the acquisition function.

3.2.1 Acquisition function

In order to maximize expected rewards, we define the acquisition function $U(\alpha_i, \beta_i)$ which is the function that determines how we choose the next machine to play. This is also known as the *policy*. This function balances the trade off between selecting the best machine based on previous plays, and the possibility of a better machine that hasn't been selected yet. This trade off is the exploration/exploitation dilemma mentioned ealier.

3.3 Backwards algorithm

- Sanjay/Sunith
- (IM - this seems like a another way to say "make the acquisition function/policy just be greedy at every step")
- if so, then I can write this out - if there is more to it, then go ahead and fill this section out

3.4 Thompson Sampling: Hueristic approach

- Lin
- basic idea of Thompson sampling (i.e. sample in proportion to both machines)
- talk about how it balances exploration v exploitation because it samples and doesn't just use expected maximum reward at each step (like algorithm above)

3.5 Analytical theory

- Sanjay/Lin
- no code, just formulas and explanation

3.6 Dynamic programming and Gittins

- Sunith

- write out algorithm (code optional)
- explain gittins index and why its optimal

3.7 Gaussian Process and Reinforcement Learning

- Sanjay

- assume knowledge of GPs, so no need to explain what a GP is
- quick overview of how MAB relates to reinforcement learning
- quick overview of how GPs relate to reinforcement learning
- using the mechanics of GP, what the connection between GP and MAB is
- show a chart or two like in the presentation
- this will be multiple sections

4 Empirical Comparisons

4.1 Data set

4.2 Charts

- using the same data set, compare UCB vs. epsilon vs. Thompson vs. gittins vs. GP

5 Conclusion

6 References