Multi Armed Bandits and Applications

Dr. M.C.Kaptein
Assistant Professor AI

4 December 2015

Section 1

Background

Background

- MSc. Economic Psychology, Tilburg University
- PdEng. User System Interaction, University of Eindhoven
- Ph.D. Industrial Design, University of Eindhoven & Stanford University
- Post Doc. Marketing, Aalto School of Economics, Helsinki
- Assistant Professor Statistics, Tilburg University, Tilburg
- Founder PersuasionAPI.

Section 2

The multi armed bandit problem

Slot machines

Formal presentation

- For $t = 1, \ldots, t = T$
- Select and action a_t out of $\mathcal{A}_t$. Often actions $k = 1, \ldots, k = K$.
- Observe reward r_t (generated by some unknown distribution $F_k(r|\theta_k)$)
- Play according to some policy $\Pi : \{a_1, \ldots, a_{t-1}, r_1, \ldots, r_{t-1}\} \mapsto a_t$

Aim of a “good” policy

- Well, get as much reward as possible!
- Thus, maximize $\sum_{t=1}^{T} r_t$
- Or, minimize regret: $\sum_{t=1}^{T}(\Pi^*(t) - \Pi(t))$

Exploration-Exploitation tradeoff

- Suppose observations $X_k \sim \text{Bern}(p_k)$
- Explore: $p_1 > p_2$? Play alternating arms to learn.
- Exploit: Play arm 1.

Very general trade-off: Exploring the outcomes of uncertain actions, versus choosing actions that one beliefs to be good.

Omnipresence of the tradeoff

Exploration vs. exploitation found in many places:

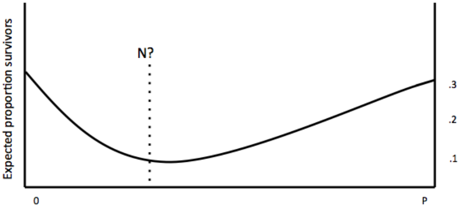
- Clinical trial: which medicine to subscribe?
- Online content selection: Which ad, news article, or product to show?
- Job choices: Try something new, vs. stick to what you have?
- Food choices: Try a new dish, stick to one you like
- Etc. etc.

Also known as *Earning vs. Learning*.

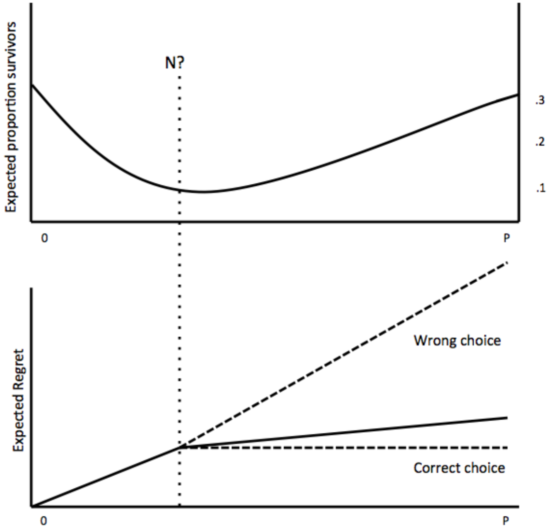
The randomized clinical trial

What is the regret of a simple RCT choosing between two medications?

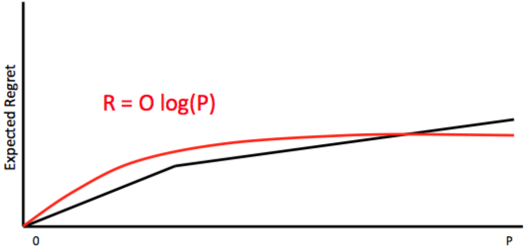
- Simple choices: $X_1 \sim \text{Bern}(p_1 = .1)$, $X_2 \sim \text{Bern}(p_2 = .5)$
- Obviously: p_1, p_2 unkown at start
- N patients in trial, P total patients in population
- Always $Pr(Wrong) > 0$



The randomized clinical trial: Regret



Optimal policies



Section 3

Thompson sampling & optimal design

Thompson sampling

Compute or sample from $\Pr(\theta|\mathcal{D})$. We can then select an action according to its probability of being optimal:

$$\int 1\left[\mathbb{E}(r|a,\theta) = \max_{a'} \mathbb{E}(r|a',\theta)\right] \Pr(\theta|\mathcal{D}) d\theta \tag{1}$$

where 1 is the indicator function.

Thompson sampling Bernoulli Bandit

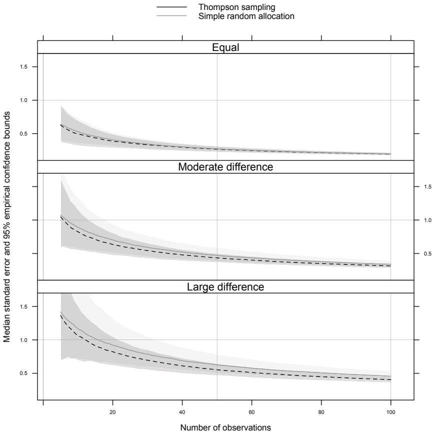
Thompson sampling in practice for the k-armed Bernoulli Bandit

- Suppose again $X_k \sim \text{Bern}(p_k)$
- Use (indepedent) $\text{Beta}(\alpha_k = 1, \beta_k = 1)$ priors
- Generate random draw d_k from each $k = 1, \ldots, k = K$ Beta() distributions
- Select arm $k' = \text{max}_k d_k$
- Update posterior $\text{Beta}(\alpha_{k'} + r_t, \beta_{k'} + 1 - r_t)$

Thompson sampling is an asymptotically optimal strategy.

Experimental design as exploration vs. exploitation [1]

- Thompson sampling for optimal design
- Sample for most "informative" datapoints
- Assume heterogeneity of variances
- Select treatments based on posterior variance estimates



[1] Kaptein, M.C. (2014).
The Use of Thompson Sampling to Increase Estimation Precision.
Behavior Research Methods

Section 4

The contextual Multi-armed Bandit problem

Extension: contexts

- For $t = 1, \ldots, t = T$
- **Observe the world,** $x_t \in \mathcal{X}_t$
- Select and action a_t out of $\mathcal{A}_t$. Often actions $k = 1, \ldots, k = K$.
- Observe reward r_t (generated by some unknown distribution $F_k(r|\theta_k)$)
- Play according to some policy $\Pi : \{x_1, \ldots, x_{t-1}, a_1, \ldots, a_{t-1}, r_1, \ldots, r_{t-1}\} \mapsto a_t$

Examples of contextual bandits

- Clinical trial: which medicine to subscribe to a **specific patient**?
- Online content selection: Which ad, news article, or product to show to a **user**?
- Job choices: You have information regarding the jobs
- Food choices: You know the ingredients of the dish
- Etc. etc.

Interesting model for (e.g.,) **treatment personalization**.

Section 5

Bootstrap Thompson Sampling

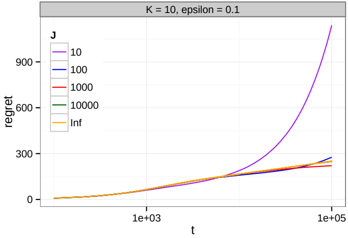
Thompson sampling for contextual bandits

- Setup some model $r = f(a, x; \theta)$
- Choose prior $Pr(\theta)$
- Observe $\mathcal{D} = (x_t, a_t, r_t)$
- Use Bayes rule and sample $\theta_{t'}$ from $Pr(\theta|\mathcal{D})$
- Select action a that maximizes $f(a, x; \theta)$ given $x_{t'}$ and $\theta_{t'}$

This can be hard if $Pr(\theta|\mathcal{D})$ is hard to sample from.

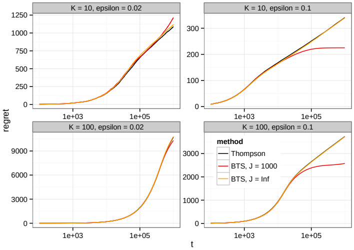
Bootstrapped Bandit[2]

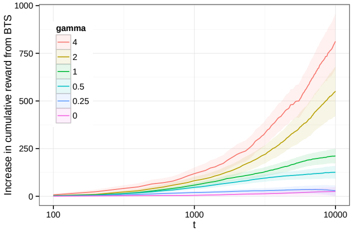
- Thompson sampling works well if posterior is known
- Not the case for complex models
- What about the (double or nothing) bootstrap distribution?
- For $1, \ldots, J$ online bootstrapped replicates



[2] Kaptein, & Eckles (2014).
Scalable Thompson Sampling with the Online Bootstrap.
arXiv

Bootstrap bandit continued . . .



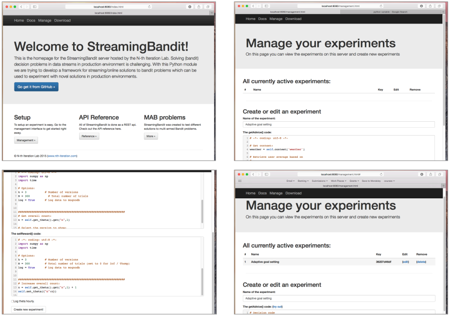


Section 6

Streaming Bandit: software

Streaming Bandit [3]

- Back end solution for streaming bandits
- Setup a REST server to handle action selection
- Recently released first stable version



[3] Kaptein, M.C. & Kruijswijk, J. (2015).
Available on Github. `https://github.com/MKaptein/streamingbandit`

Design choice: learning vs. choosing

We identify two steps:

1. The *summary* step: In each summary step $\theta_{t'-1}$ is updated by the new information $\{x_{t'}, a_{t'}, r_{t'}\}$. Thus, $\theta_{t'} = g(\theta_{t'-1}, x_{t'}, a_{t'}, r_{t'})$ where $g()$ is some update function.
2. The *decision* step: In the decision step, the model $r = f(a, x_{t'}; \theta_{t'})$ is evaluated for the current context and the possible actions. Then, the recommended action at time t' is selected.

Implemented in `getAction()` and `setReward()` calls.

Online learning

Forces an online learning approach.

- Summation over datapoints:
- Version 1: $S_T = \sum_{t=1}^{T} x_t$
- Version 2: $S_T = S_{T-1} + x_t$

Linear vs. Quadratic function of T to compute.[4]

[4] Ippel, L., Vermunt, J., & Kaptein, M.C. (2015) Streaming EM approximations. *Under submission.*

Section 7

Applications of the software:

Simple experiment using StreamingBandit

Summarize:

```
import libs.base as base
prop = base.Proportion(self.get_theta(key="version",
   value=self.action["version"]))
prop.update(self.reward["click"])
self.set_theta(prop, key="version", value=self.action["version"])
```

Decide:

```
import libs.base as base
propl = base.List(self.get_theta(key="version"),
  base.Proportion, ["A", "B"])
if propl.count() > 1000:
  self.action["version"] = propl.max()
else
  self.action["version"] = propl.random()
```

Simple experiment using StreamingBandit 2

Decide:

```
import libs.thompson as thmp
propl = thmp.BBThompsonList(self.get_theta(key="version"),
   Proportion, ["A", "B"])
self.action["version"] = propl.thompson()
```

Streaming Bandit in practice: Lock in Feedback

Possible policy for the continuum bandit problem:

- $a \in \mathbb{R}$
- Project together with Prof. Dr. Davide Iannuzzi
- Oscillate a with a know frequency
- Amplify r, and integrate to obtain first derivative.[5]

[5] Kaptein, M.C. & Iannuzzi, D. (2015)
Lock in Feedback in Sequential Experiments. *Under submission.*

Streaming Bandit in practice: Santander

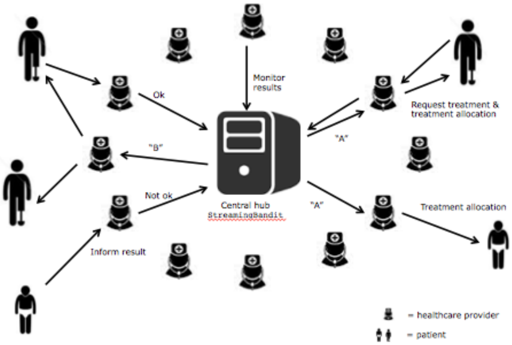
Software currently used for contextual bandit trials

- Observe features of a customer requesting a loan
- Select an interest rate
- Observe acceptance of loan ($r = f(IR, y)$)
- Objective: Choose interest such as to maximize profit

Section 8

Future

Personalized feedback and treatment selection



Questions?

Maurits Kaptein
Archipelstraat 13
6524LK, Nijmegen
0621262211
maurits@mauritskaptein.com