Bandits

Spyros Samothrakis
Research Fellow, IADS
University of Essex

February 7, 2017

Bandits

- We will discuss bandits
- We are in effect revisiting some ideas from lecture two
  - Hypothesis testing
- I think the this is a much easier to understand framework vs hypothesis testing
- Requires a bit more heuristic thinking…

Examples

- You send a user an e-mail
  - User clicks on the link you get $r = 1$
  - User fails to click on the link after 3 days $r = 0$
- Playing games
  - What is the next best action to take in Chess?
    - Chess has a sequential element - hence “Reinforcement Learning”
    - But close enough…
- Online adverts
  - User clicks on an advert $(r = 1)$
  - User clicks fails to click on an advert $(r = 0)$

The bandit problem

- Bandits are a tuple $< A, R >$
- Where $a \in A$ is a set of actions
- $r \in R$ is a set of rewards
- $R(a, r) = P(r|a)$
  - The probability of getting a reward r given that I have done action a
- "You do an actions, you get some feedback"

The goal

- Find an optimal policy $\pi(a) = P(a)$ that maximises the long term sum of rewards
  - Long term sum is $\sum_{\tau=0}^{T} = r_\tau$
- The "action-value" function $Q(a)$ is the expected reward for taking action a
  - Q $= E[r|a]$
- The "value" function is $V = E_\pi[r]$
  - The average Q values, given that a policy that I follow

Example problem

```
Dear Sir/Madam,

Best quality flasks and vials for your experiments

Click the link below to buy - discounted prices

(Link)
```

```
Dear <Name>,

This is Nick from www.MegaFlasksAndVials.com - super discounts below

(Link)
```

Let's simulate

- First e-mail is a_0
- Second e-mail is a_1
- Policy is $\pi(a_0) = 0.5, \pi(a_1) = 0.5$
- Let's manually calculate some Q's and V's on e-mail sending problem

Goals (1)

- So our goal is to find the best action
- Optimal $V^* = \max_{a \in A} Q(a)$
- But these values can only be found through averages
  - $\hat{Q}(a)$, $\hat{V}$
- We could have done confidence intervals...
  - But this would entail a random policy
  - Maybe we can do better

Goals (2)

- We would like to find the best action using the minimum amount of samples possible
- Keep focusing on the best arm/action
  - While also checking making sure that other actions are sufficiently explored
- This is known as the "exploration/exploitation" dilemma

Regret (1)

- Regret is $I_t = E\left[\sum_{\tau=0}^{T} (V^* - Q(a_\tau))\right]$
  - Or, equivalently $E\left[\sum_{\tau=0}^{T}\left(\max_{a \in A} Q(a) - Q(a_\tau)\right)\right]$
- The count is $N_t(a)$, the number of times we took action a until time t
- The Gap $\Delta_a = V^*(a) - Q(a)$, the difference between the optimal action and the action taken

Regret (2)

- It turns out that
  - $\sum_{a \in A} (E[N_t(a)\Delta_a])$
- We would like to minimise the times we have large gaps
- But we have no clue what the gaps are...

Another example

- Three actions to choose from
- Link in internal promo e-mail
  - Thus users more likely to click

```
n_actions = 3

def action_0():
    return np.random.choice([1,0], p=[0.5, 0.5])

def action_1():
    return np.random.choice([1,0], p=[0.6, 0.4])

def action_2():
    return np.random.choice([1,0], p=[0.2, 0.8])

rewards = [action_0, action_1, action_2]
```

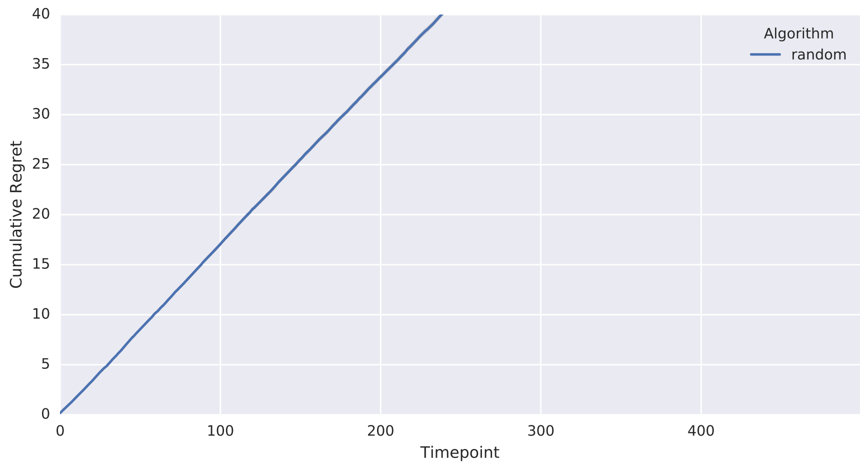

About Bandits Adapting to changing rewards regimes The Adversarial case Contextual Bandits Conclusion

Pure exploration

- Somewhat similiar to the A/B case
  - But in A/B you should have set a cut-off point
- You send more or less the equal number of e-mails
- Very simple setup

About Bandits Adapting to changing rewards regimes The Adversarial case Contextual Bandits Conclusion

Regret of pure exploration

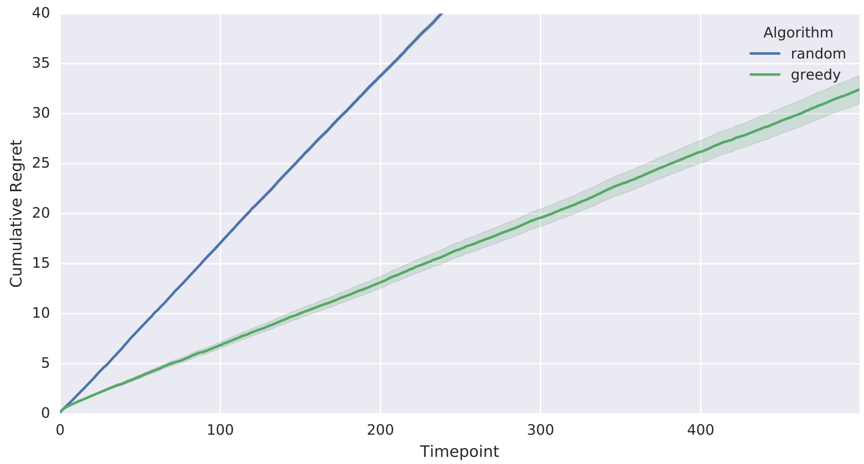


About Bandits Adapting to changing rewards regimes The Adversarial case Contextual Bandits Conclusion

Greedy

- You choose the action that does that is currently doing better
- Can you see a problem with this? * It might get stuck in suboptimal actions
- Let's try to do this on the board

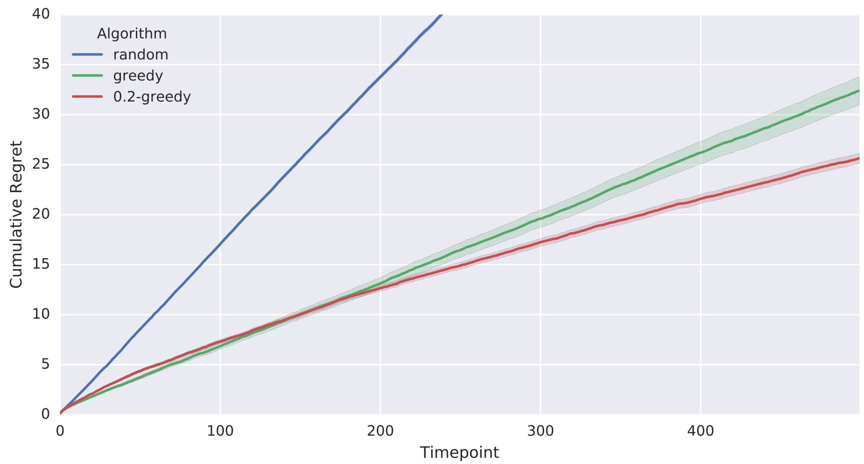
Regret of greedy



ϵ-greedy

- You set a small probability ϵ with which you act randomly
- The rest of the time you add greedily * i.e. you choose the best action
- This is a very common (but inefficient setup)
- What is the optimal ϵ?

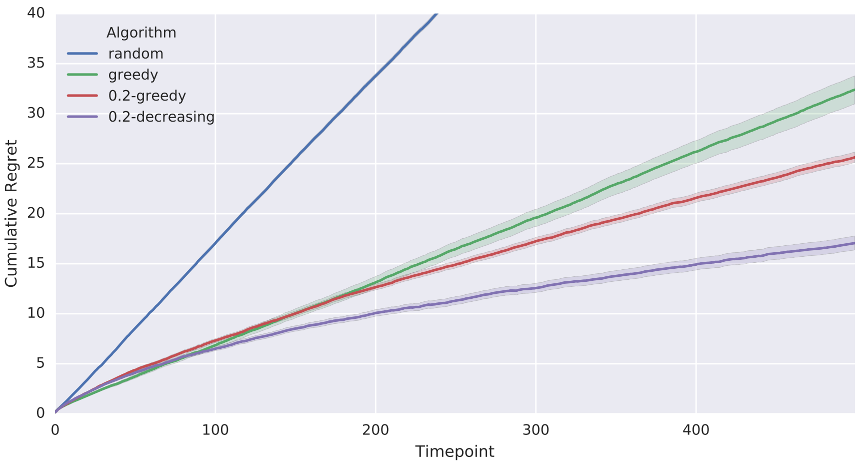
Regret of ϵ-greedy



ϵ-decreasing

- Same as epsilon greedy, but now you decrease epsilon as you pull the arms
- We do ~python e *= 0.99 ~python

Regret of ϵ-decreasing



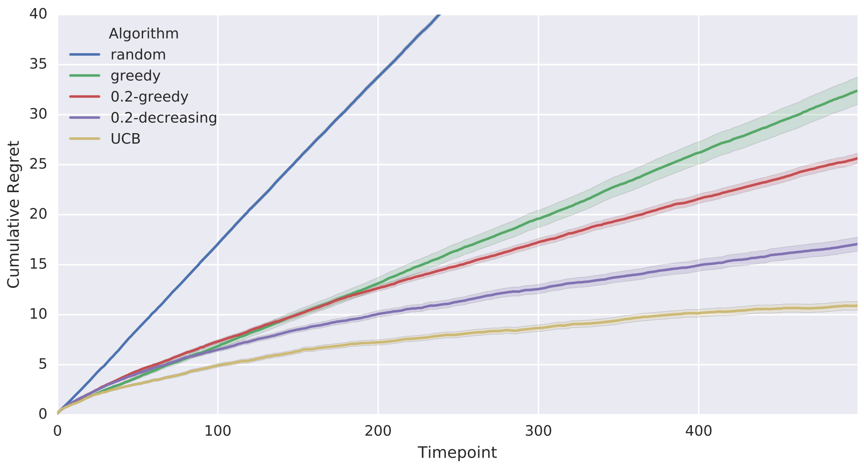
Optimism in the face of uncertainty

- There is a principle termed "optimism in face of uncertainty"
- In practical terms this means that you should try actions with highly uncertain outcomes
  - You believe the best action is the one you haven't explored enough
- Works well in practice

Upper Confidence Bounds

- A very popular algorithm
- Fairly robust
- $UCB(a) = \hat{Q}(a) + U(a)$
- $UCB1(a) = \hat{Q}(a) + C\sqrt{\frac{log(\tau)}{N_\tau(a)}}$
- $N_\tau(a)$ is the times action a was executed
- τ is the current timepoint/time
- $C \in [0, \inf]$ is a constant - I set it to 0.5 for the plots below
  - Can you guess what the effect of C is?

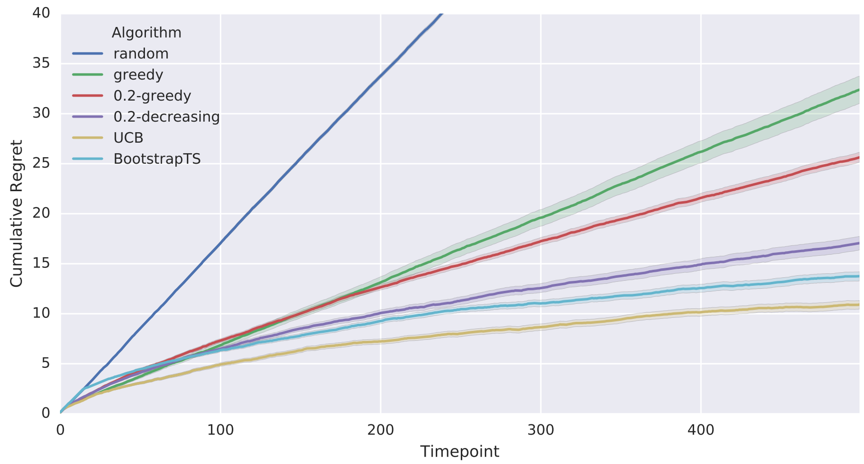
Regret of Upper confidence bounds



Bootstrap Thompson Sampling

- What if we could take bootstrap samples of action rewards that we have collected?
- You would have incorporated the uncertainty within your bootstraps
- If you have a number of bootstraps you have a distribution over possible $\hat{Q}(s)$
- Sample from this distribution
- A version of probability matching
  - $\pi(a) = P[\hat{Q}(a) > \hat{Q}(a') \forall a' in A]$

Regret of Bootstrap Thomson Sampling



Change of rewards

- What if rewards just change
- Because people are bored of your e-mails
  - They talk to each other
  - Out of fashion
- You might want to have continuous adaptation
- Keeping all values and finding $Q(\hat{s})$ is expensive
  - What happens in round 1000?
  - How many additions / divisions?

The sequential case

- What if you are to take a series of action?
- Surely your current action depends on your future actions
- Hence there is going to be a change in the distribution of rewards * Induced by the experimenter
- For example, you do an e-mail campaign

Example e-mail campaign

- You send your first e-mail
  - "Number of clicks on a specific product"
- Send second e-mail
  - "Will you buy the add-on?"
- Send third e-mail
  - "Let us service your product"
- You want to get all rewards
- Creates a hierarchy of rewards

Introducing state

- $s \in S$ can be used to differentiate between different "states", conditioning π, V and Q values on states
- $\pi(s, a)$, $V(s)$, $Q(s, a)$
- e.g. in the example above, you have $Q(``firstemail'', ``emailtypeA'')$
- Let's write the rest of the states, the policies, V and Q-Values

Incremental calculation of a mean

$$\hat{Q}_\tau(s, a) = \hat{Q}_{\tau-1}(s, a) + \frac{\overbrace{\mathrm{v}_\tau - \hat{Q}_{\tau-1}(s, a)}^{\textbf{Error}}}{\tau}$$

V can be the reward or the sum of rewards you got from different bandits

$$\hat{Q}_\tau(s, a) = \hat{Q}_{\tau-1}(s, a) + \frac{1}{\tau}\overbrace{\mathrm{v}_\tau - \hat{Q}_{\tau-1}(s, a)}^{\textbf{Error}}$$

$$\hat{Q}_\tau(s, a) = \hat{Q}_{\tau-1}(s, a) + \alpha\left[\mathrm{v}_\tau - \hat{Q}_{\tau-1}(s, a)\right]$$

Incremental bootstrap

- A bit harder to do

Oza, Nikunj C., and Stuart Russell "Online bagging and boosting." Systems, man and cybernetics, 2005 IEEE international conference on. Vol. 3. IEEE, 2005.

When you are being despised

- Imagine an adversarial scenario
  - You put adverts on your website
  - Somebody creates an ad-blocker
- Create a spammy e-mails
  - People use filters to evade them
  - Based on certain words

Equilibria

Finding Equilibria

Rethinking states

Random

E-decreasing

LinUCB

Bootstrap Thomson Sampling

Conclusion

- First hit on bandits
- Super-exciting research area