Bandits and Reinforcement Learning

COMS E6998.001 Fall 2017
Columbia University

Alekh Agarwal Alex Slivkins
Microsoft Research NYC

What the course is about?

- Algorithms for sequential decisions and “interactive” ML under uncertainty
  - algorithm interacts with environment, learns over time.
    loop: observe “state” – make a decision – observe reward/feedback
  - machine learning, theoretical CS, AI, operations research, economics
  - since 1933, very active in the past decade
- Focus on “bandits” (no state) and “contextual bandits”
  (state does not depend on past actions)
- Focus on theory (design & analysis of algorithms)
  - … using tools from Probability
  - … with lots of examples & discussions for motivations & applications

This lecture

- course organization
- intro to the problem space
- short break
- review of concentration inequalities (necessary basics)

Prerequisites

- **Algorithm design & mathematical proofs**:
  Exposure to algorithms and proofs at the level of an undergraduate algorithms course (CSOR 4231). Graduate ML course (COMS 4771) or current enrollment therein. If you do not meet these, please email the instructors.
- **Probability & statistics**:
  - I will review concentration inequalities later today
  - Review of basic probability will be posted on course webpage
  - deeper familiarity would help, but not required
- **Programming**: familiarity with programming is not required;
  however, your “project” may involve simulations/experiments if you choose so.

Logistics

- Instructors: Alekh Agarwal and Alex Slivkins (Microsoft Research NYC).
- Schedule: Wednesdays 4:10-6:40pm, 404 Intl Affairs Bldg
- Office hours: after each class, and online TBD
- Course webpage: http://alekhagarwal.net/bandits_and_rl/index.html
- Q&A and announcements: we will use Piazza, please sign up! https://piazza.com/columbia/fall2017/comse6998001/home
- Contact: **:** bandits-fa17-instr@microsoft.com (but please use Piazza if appropriate)
- Waitlist: let's see how it goes … sign up for Piazza!

Coursework and assessment

- **Project**: reading, coding, and/or original research on a course-related topic
  - written report: a short academic-style paper
- **Grading:** letter grade based on the project
- **Homeworks:** 2-3 problem sets throughout the course, **not graded**
  - to check/solidify your understanding of the material
  - we'll distribute hints/solutions, and we'll be available to discuss

Projects

- **default**: reading several papers, making sense of a given topic
  - simulations and/or research if you feel brave and inspired
- **specific topic suggestions** – will be posted soon
- **topic proposals** – due Oct 20 (tentatively)
- **feedback / discussion**: we'll aim to be available before and after the proposal
- **output:** written report, short presentation in the last class
- **we can only handle 10-12 projects**
  - Students will need to bunch up, esp. on reading projects

Related courses at Columbia

- Daniel Russo @Business School, Fall'17
  Dynamic Programming and Reinforcement Learning (B9140-001)
- Shipra Agrawal @IEOR department, Spring'18
  "Reinforcement learning"

Our course focuses more heavily on contextual bandits and off-policy evaluation than either of these, and is complimentary to these other offerings

Intro to the problem space

(Informal & very stylized) running examples

- **News site.** When a new user arrives, the site picks a news header to show, observes whether the user clicks. Goal: maximize #clicks.
- **Dynamic pricing.** A store is selling a digital good (e.g., an app or a song). When a new customer arrives, the store picks a price. Customer buys (or not) and leaves forever. Goal: maximize total profit.
- **Personalized health advice.** A health app gives you health/lifestyle recommendations, and tracks how well you follow. Goal: maximize #adopted recommendations (weighted by their usefulness).
- **Chatbot for task completion.** You arrive with a specific task in mind (e.g.: tech support issue, buying a ticket), and a chatbot assists you. Goal: maximize #completed tasks.

Basic model

- A fixed set of K actions ("arms")
- In each round $t = 1 \ldots T$ algorithm observes a context/state x_t, chooses an arm a_t, and observes the reward r_t for the chosen arm
- **"Bandit feedback"**: no other rewards are observed!
- **IID rewards:** reward for each arm is drawn independently from a fixed distribution that depends on the arm and the context, but not on the round t.

No contexts $\Rightarrow$ multi-armed bandits
contexts do not depend on past actions $\Rightarrow$ contextual bandits
contexts/state depends on past actions $\Rightarrow$ reinforcement learning

Examples

Example	Context	Action	Reward	
News site	User location	an article to display	1 if clicked, 0 otherwise	makes sense even w/o context ⇒ bandits or contextual bandits
Dynamic pricing	Buyer's profile	a price p	p if sale, 0 otherwise	
Health advice	User health profile	what to recommend	1 if adopted, 0 otherwise	Context is essential ⇒ contextual bandits
Chatbot	Stage of conversation	what to say	1 if task completed, 0 otherwise	Context is essential, depends on the past actions ⇒ reinforcement learning

Exploration-exploitation tradeoff

- Bandit feedback => need to try different arms to acquire new info
  - if algorithm always chooses arm 1, how would it know if arm 2 is better?
- fundamental tradeoff between acquiring info about rewards (***exploration***) and making optimal decisions based on available info (***exploitation***)
  - multi-armed bandits is a simple model to study this tradeoff

Rich problem space

- Bandits vs contextual bandits vs reinforcement learning
- … many other distinctions, even for bandits

Distinction #1: which problem to solve?

- Explore-exploit problem:
  we control the choice of actions and want to maximize cumulative reward
- Offline evaluation:
  some algorithm collects data, and we use this data to answer *counterfactuals*:
  what if we ran this policy (mapping from contexts to actions) instead?
  - Off-policy: we do not have control over data collection
  - On-policy: we design data collection ("exploration policy")

Distinction #2: where rewards come from?

- **IID rewards:** the reward for each arm is drawn independently from a fixed distribution that depends on the arm but not on the round t.
- **Adversarial rewards:** rewards are chosen by an adversary.
- **Constrained adversary**:
  rewards are chosen by an adversary with known constraints, e.g.:
  - reward of each arm can change by at most ϵ from one round to another
  - reward of each arm can change at most once
- **Stochastic rewards** (beyond IID):
  reward of each arm evolves over time as a random process
  - e.g. random walk: changes by $\pm\epsilon$ in each round

Distinction #3: extra feedback

- Bandit feedback (most of this course): reward for chosen arm and no other info
  News site: a click on a news article
- Partial feedback
  News site: time spent reading an article?
  Health advice: how diligently was each recommendation followed?
  Dynamic pricing: sale @p => sale at any smaller price
  Still, no full "counterfactual" answer (*what could have happened*)
- Full feedback: rewards for all arms are revealed
  Dynamic pricing $\rightarrow$ choosing min acceptable price at an auction (*reserve price*)
  Given the bids tells you what would have happened at any other reserve price

Other distinctions

- **Bayesian prior**? problem instance comes from known distribution ("prior"), optimize in expectation over this distribution
- **Global constraints:** e.g.: limited #items to sell
- **Complex decisions**
  A news site picks a *slate* of articles
  Health advice consists of multiple specific recommendations.
- **Structured rewards:** rewards may have a known structure
  e.g.: arms are points in $[0,1]^d$ and in each round the reward
  is a linear / concave / Lipschitz function of the chosen arm
- **Policy sets**: compare to a restricted set of *policies*: mappings from contexts to arms.
  e.g.: linear policies or decision trees of bounded width and depth

Course outline

- Multi-armed bandits (4 lectures: Alex)
  - Bandits with IID rewards
  - Adversarial rewards, full feedback
  - Adversarial bandits
  - Impossibility results (any algorithm cannot do better than …)
- Contextual bandits (4 lectures: Alekh)
- Reinforcement learning (2 lectures: Alekh)
- Back to bandits, topic TBD (1 lecture: Alex)
- Final class: project presentations

Some philosophy

- Reality can be complicated … we often study simpler models.
- a good model captures some essential issues
  … present in multiple applications
  … and allows for clean solutions with good performance
  and/or clean/strong *impossibility results*
  … and provides intuition/suggestions for more realistic models
- even a good model typically does not fully capture any one application
- very rich problem space => why work on problems with shaky motivation?

More examples

Example	Action	Rewards / costs
medical trials	drug to give	health outcomes
internet ads	which ad to display	bid value if clicked, 0 otherwise
content optimization	e.g.: font color or page layout	#clicks
sales optimization	which products to sell at which prices	$$$
recommender systems	suggest a movie, restaurants, etc.	#users that followed suggestions
computer systems	which server(s) to route the job to	job completion time
crowdsourcing systems	which tasks to give to which workers	quality of completed work
	which price to offer?	#completed tasks
wireless networking	which frequency to use?	#successful transmissions
robot control	a “strategy” for a given state & task	#tasks successfully completed
game playing	an action for a given game state	#games won