Lecture 1: Introduction to Online Learning

ID 6002W: Online and Reinforcement Learning

Web-enabled M.Tech. program, IIT Madras

Suryanarayana Sankagiri

12th May, 2026

About Your Instructor

Call me Surya

- Joined IIT Madras in April 2026
- First time teaching in IIT!

- Helped design a new course on bandits + RL at EPFL in 2025
- This course is different: more tailored to working professionals

Find out more at sankagiri.github.io

What is Online Learning?

A data-driven paradigm for making and improving decisions over time

- **Data-driven:** like all machine learning, we estimate parameters or identify patterns from data
  - What to learn from data? user behaviour, environmental factors, performance of new product

- **What makes this different from other ML?**
  - data comes in one step at a time, AND
  - one must take actions (or make decisions) at each step

Two modes of learning in recommendation systems

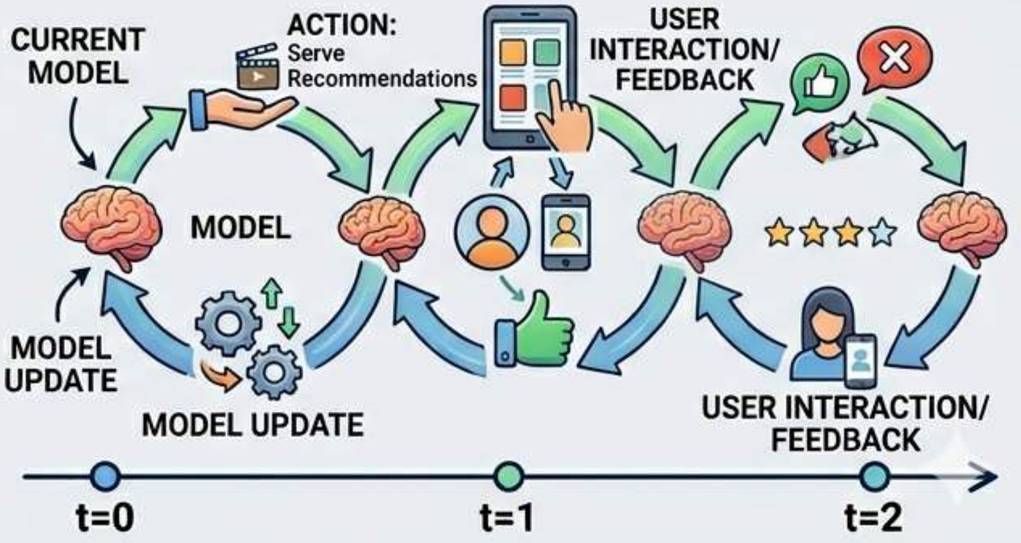
Offline (Batch) Learning

historical data
(clicks, purchases, skip)

batch training

deployment
(static model)

Online (Real-time) Learning



Broad Goals of Online Learning

Basic Principle of Learning:

More data $\Rightarrow$ Better estimation of unknowns $\Rightarrow$ Better actions

- Initially, actions will not be good (unavoidable)
- Eventually, actions should be as good as offline setting (all data given together)
- What about in-between?

<u>Need careful formulation</u>

Broad Goals of Online Learning

Bad intermediate decisions must be penalised

- *Forces* the algorithm to perform the best, given available data
- If no penalty for intermediate decisions, problem reduces to offline learning!

Two natural formalisms

- Find the best action, as quickly and reliably as possible (lecture 2)
- Perform well throughout the entire period (lectures 3 and 4)

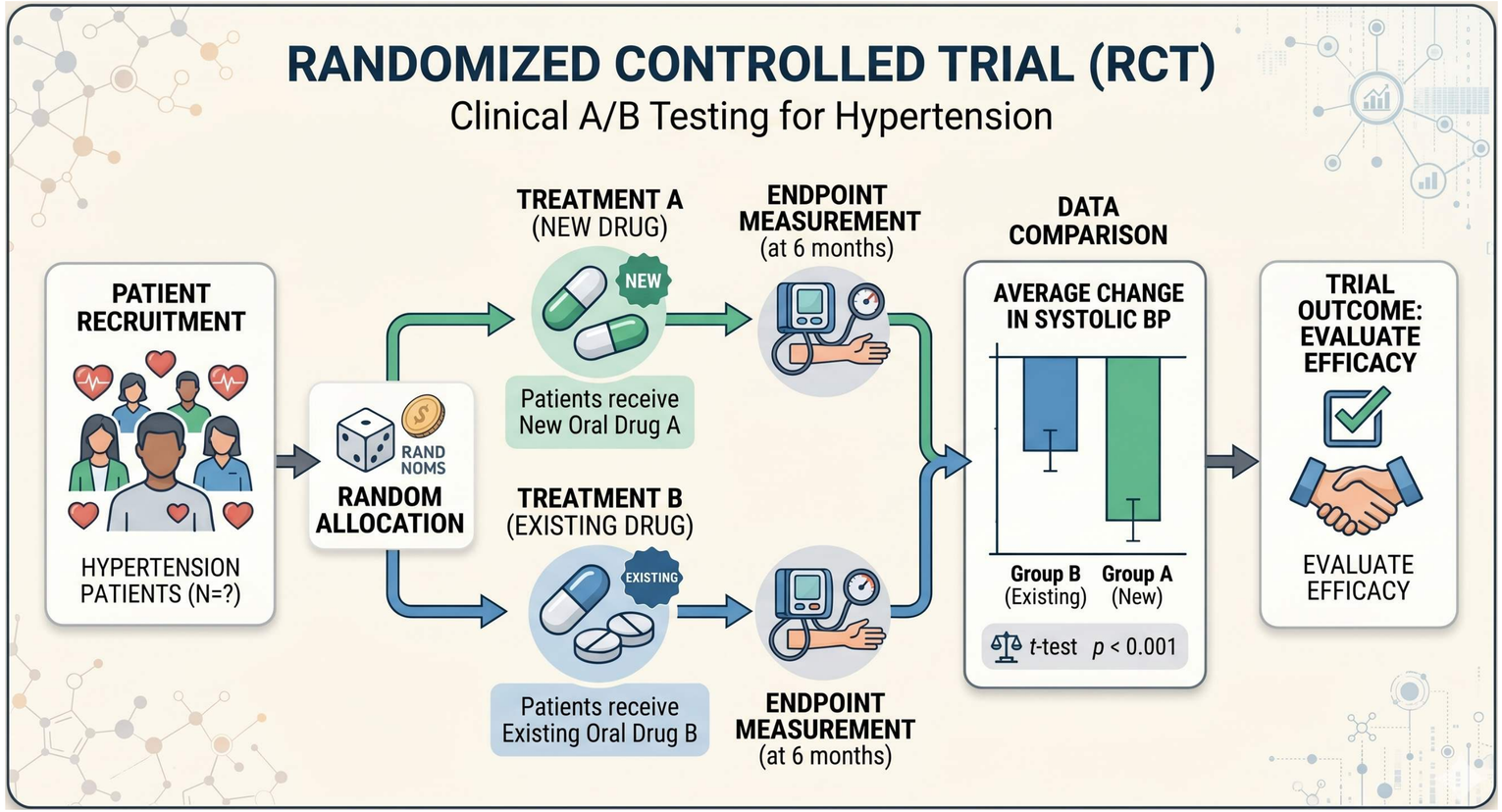
Examples of Online Learning

1. A/B testing in clinical trials
2. Dynamic pricing in ride-hailing systems
3. Recommender systems for news platforms
4. (Reinforcement learning) playing chess

Example 1: Adaptive A/B Testing

In Clinical Trials

Suppose we want to test out a new treatment for a disease.



Can we do better?

Example 1: Adaptive A/B Testing

In Clinical Trials

- Testing is expensive and risky
- We should stop testing as early as possible. But when?

Stop when we are confident!

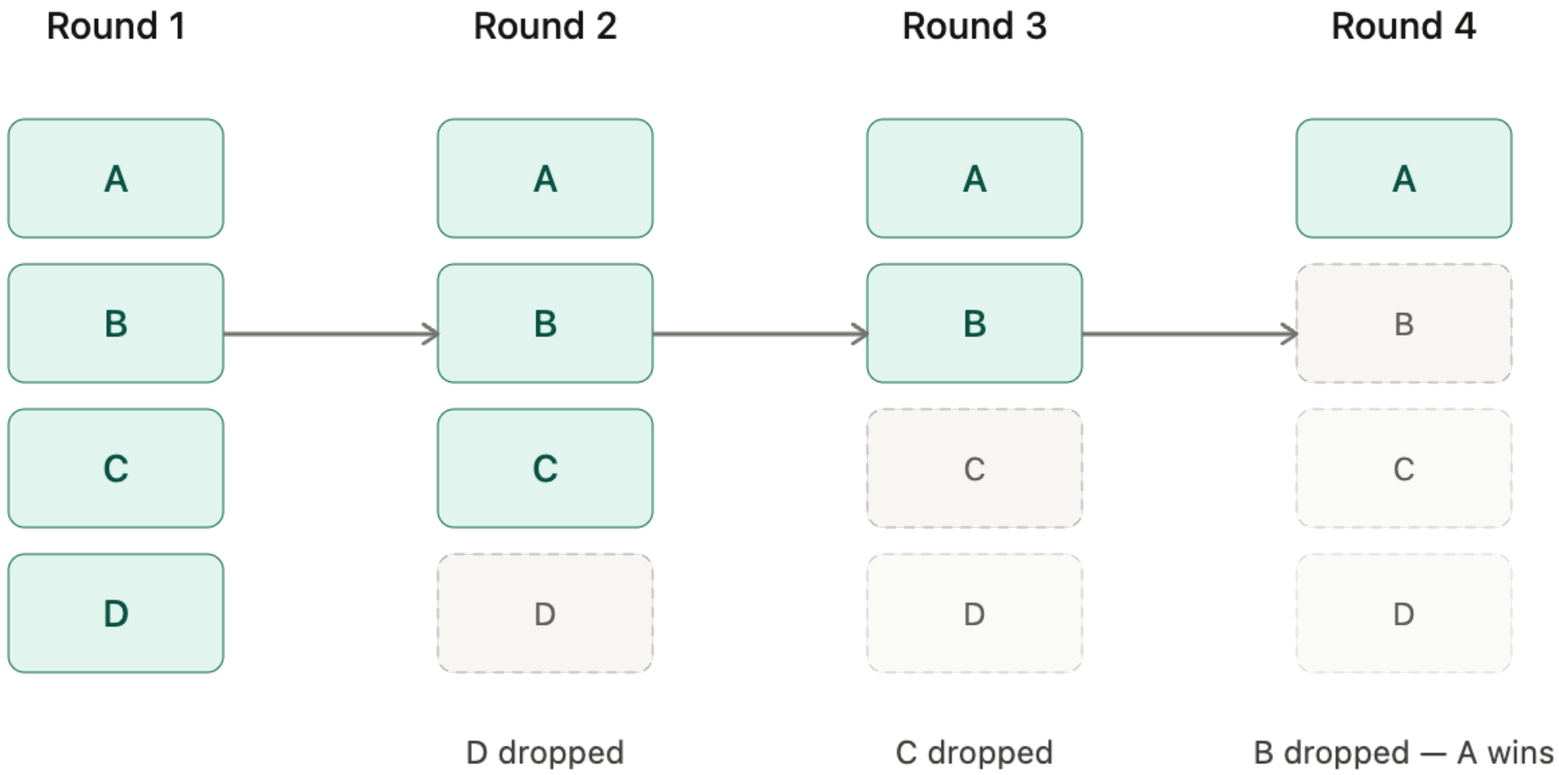
An online learning problem: *decide at each round* whether to stop or not

- **Stop too early**: chance of making a mistake in identifying best drug
- **Stop too late**: incur expense of testing, more patients suffer poor treatment

Example 1: Adaptive A/B Testing

With Multiple Options

A richer decision: which option to eliminate, when?



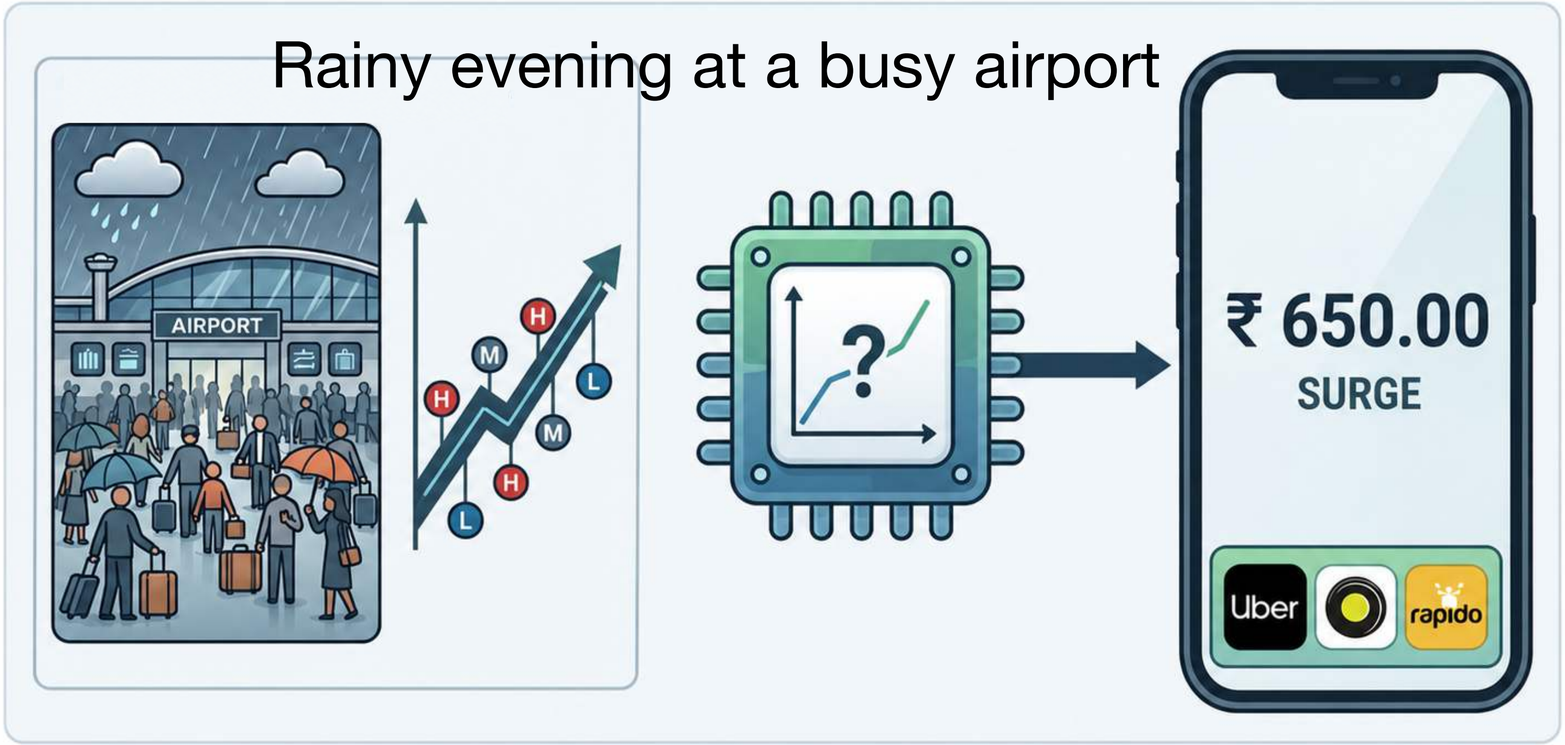
We will study best-arm identification in Lecture 2

Example 2: Dynamic Pricing

In Ride-Hailing

Price changes as a function of demand

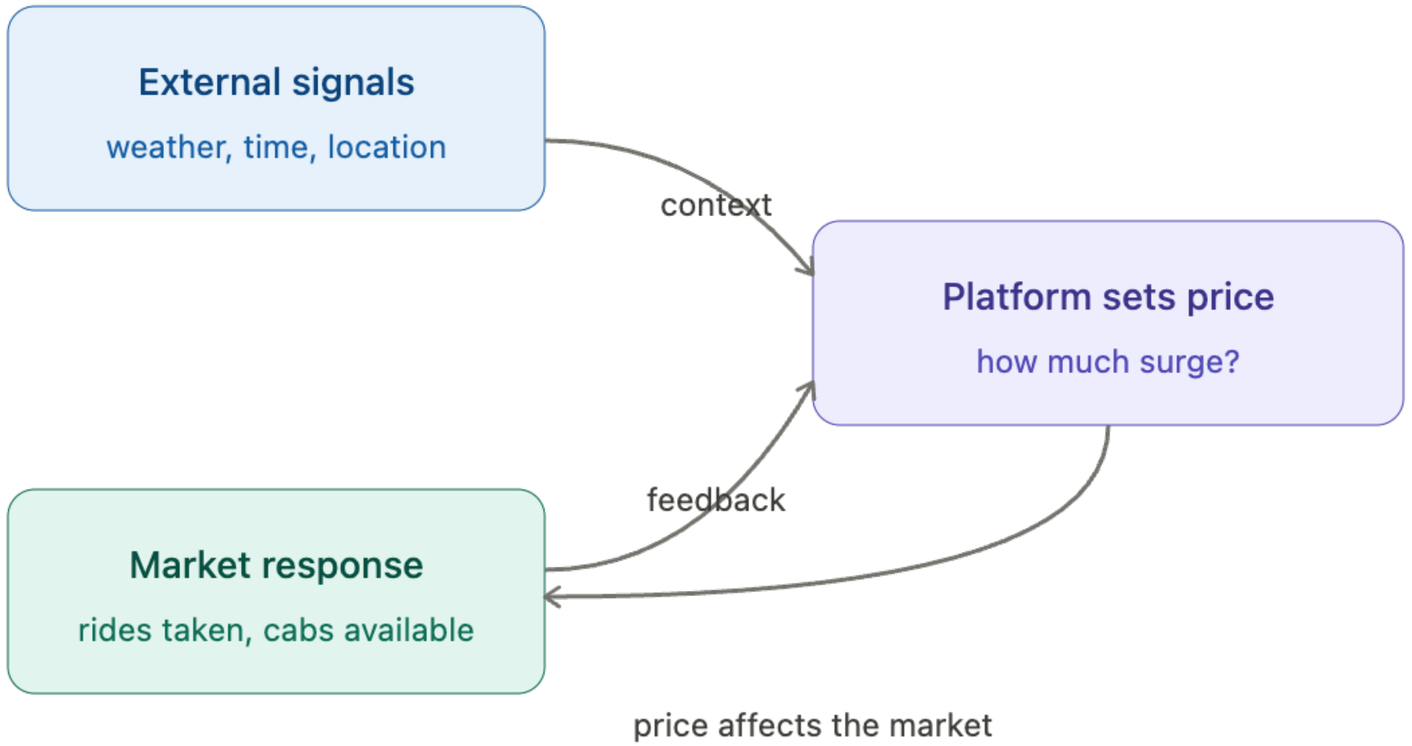
- Objective of the platform: maximise total revenue over time
- Unknown: how does the demand respond to price?



Example 2: Dynamic Pricing

In Ride-Hailing

- Sequential decision-making for platform: how to price?
  - Set price too low: missed revenue opportunity
    - Signal: all cabs taken
  - Set price too high: not many ride takers, low revenue
    - Signal: many empty cabs



Example 2: Dynamic Pricing

In Ride-Hailing

- The platform can learn from its past actions and corresponding data
- While learning, it must simultaneously make good decisions over time
- Maximise cumulative reward, or equivalently, minimise cumulative regret

- Basic regret minimising strategies will be covered in lectures 3 and 4
- Following lectures will explore models with additional complexities, e.g., context-aware pricing, delayed feedback, etc.

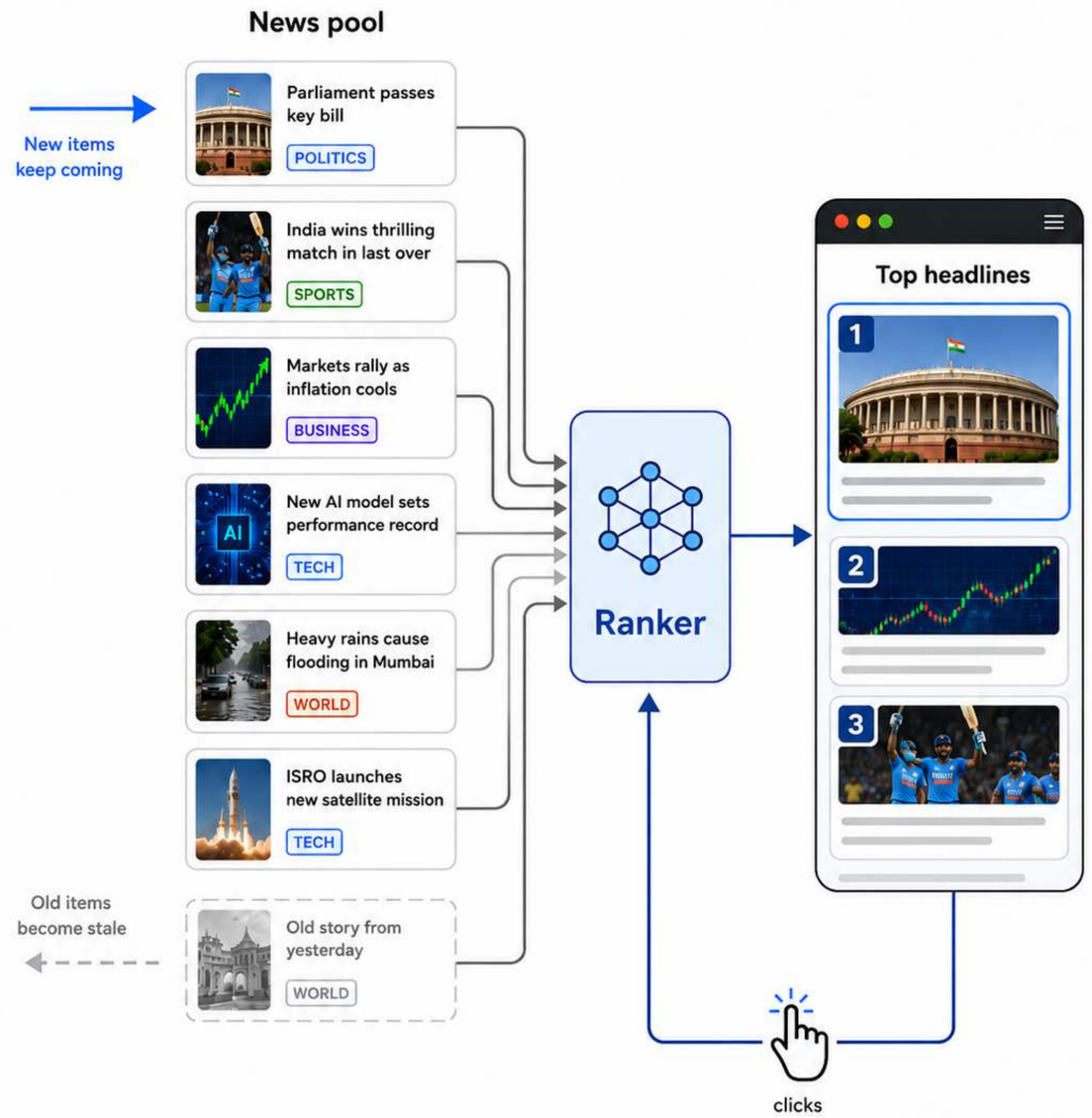
Example 3: Recommendation Systems

For News Platforms

Continuously decide what items to show as headlines

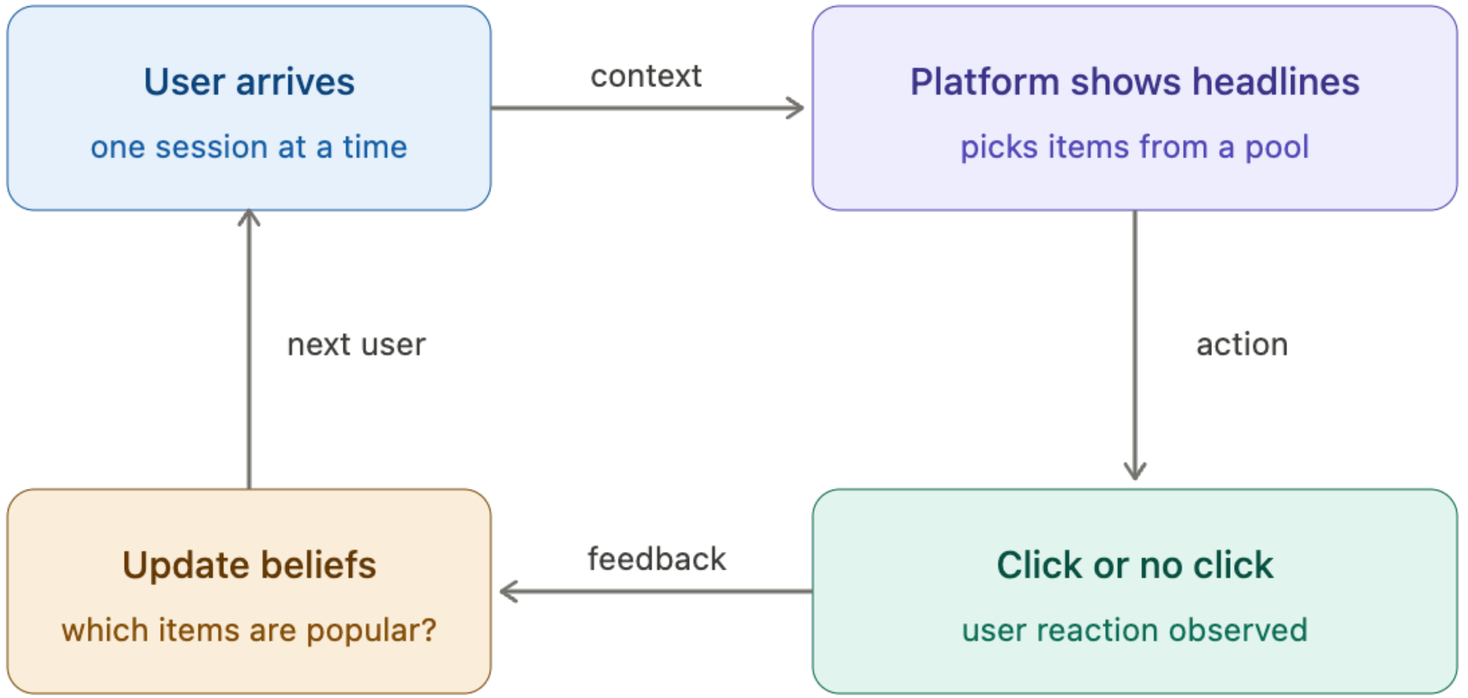
Challenges:

- Multiple news items, few slots.
- How do we know what is popular?
  - News items go stale — yesterday's popular item isn't today’s
  - New items keep arriving — must test them before exploiting



Example 3: Recommendation Systems

For News Platforms



- Initially, item popularity is unknown. Show at random
- Over time, popular items become clear —> show them

Example 3: Recommendation Systems

For News Platforms

The core dilemma

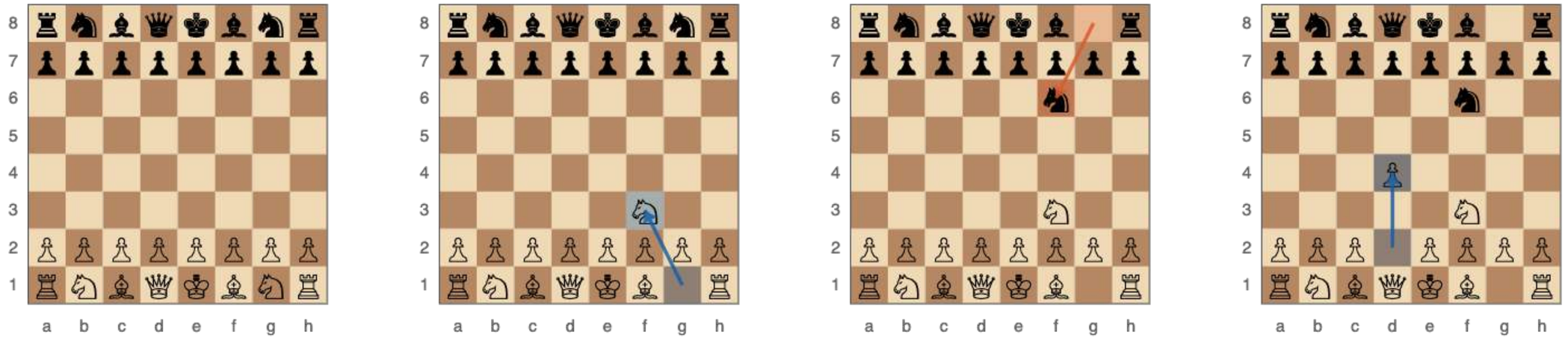
- Show the current best (exploit) → safe in the short run, but never discover better options
- Show something new (explore) → costs short-term clicks, but you learn

In online learning, the data you observe next depends on the actions you take now.

Exploration isn’t optional. *It is how you learn.*

Example 4: Playing Chess

With an AI Agent



- Many games involve a sequence of decisions
- After each move, the environment responds
- **Goal:** choose a trajectory of moves leading to a win as quickly as possible

Example 4: Playing Chess

With an AI Agent

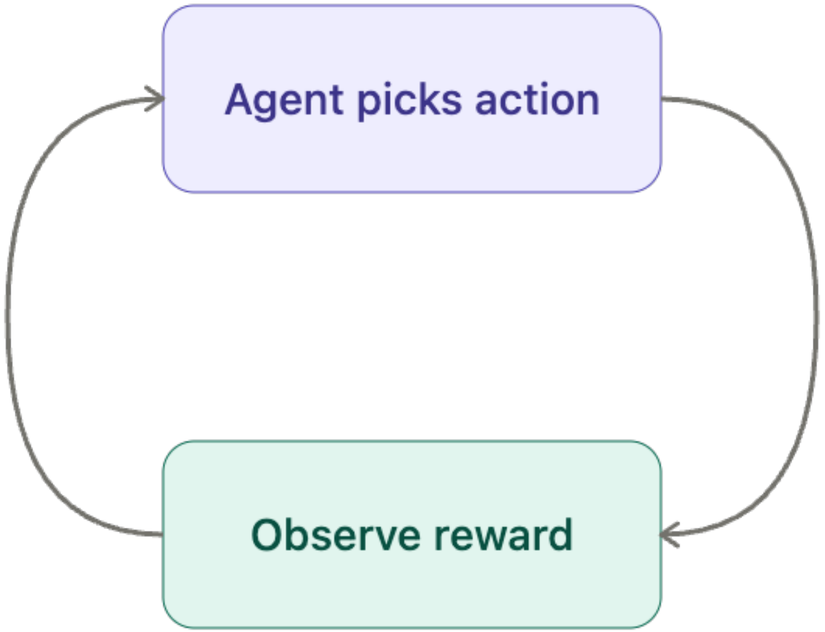
What's different here:

- Each move changes the **state of the board** — and the state restricts your next set of options
- Your action at time t depends on actions at t–1 (and earlier)
- Sometimes a short-term loss is the right long-term move (sacrificing a piece)

Example 4: Playing Chess

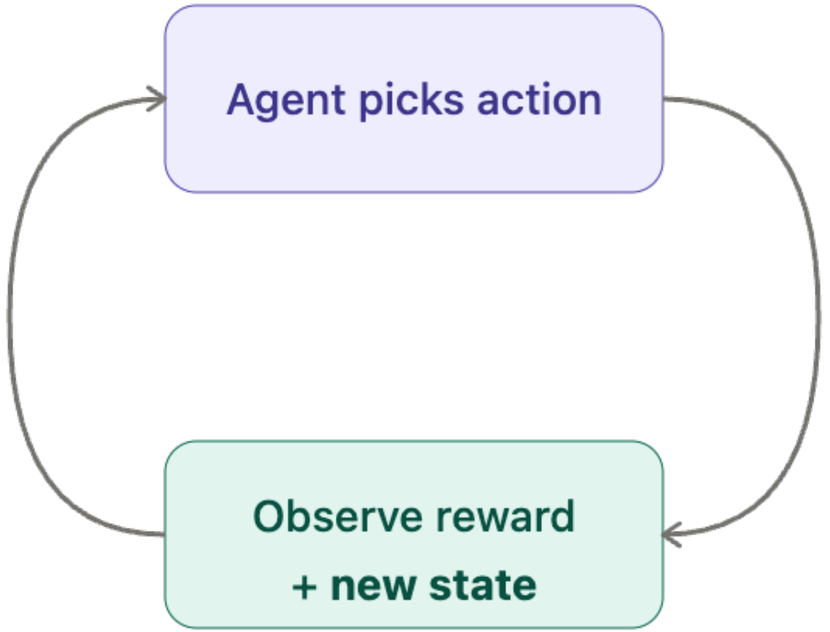
With an AI Agent

Online learning (bandits): examples 1–3



no state — each round starts fresh

Reinforcement learning: chess



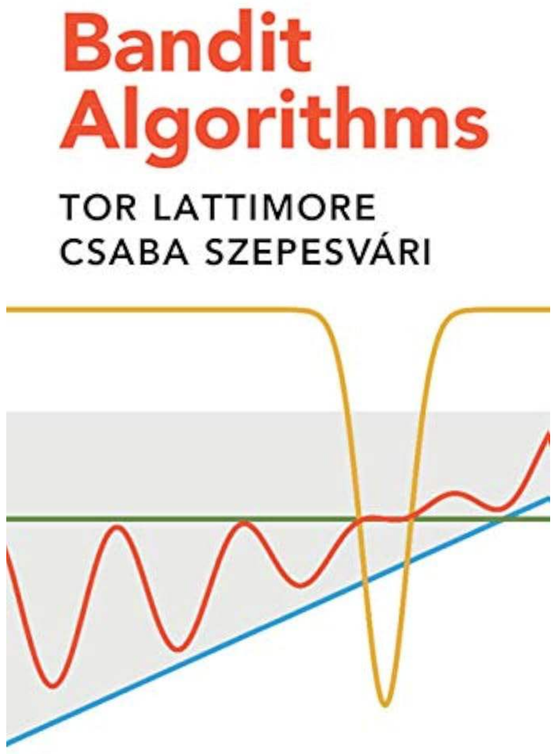
state carries over — actions shape the future

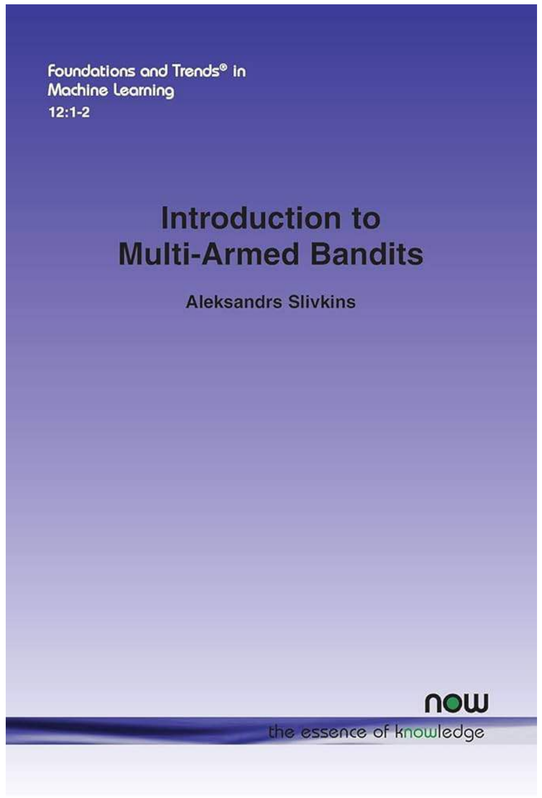
Example 4: Playing Chess

With an AI Agent

- Playing chess is an example of **reinforcement learning**
- Other examples of reinforcement learning are:
  - Learning to drive (self-driving cars)
  - Robots learning to walk
  - Dynamic pricing (actions now affect future state)
- Reinforcement learning will be covered in the second half of this course (30th June onwards) by Prof. P. S. Jayadev

The Multi-Arm Bandits Framework

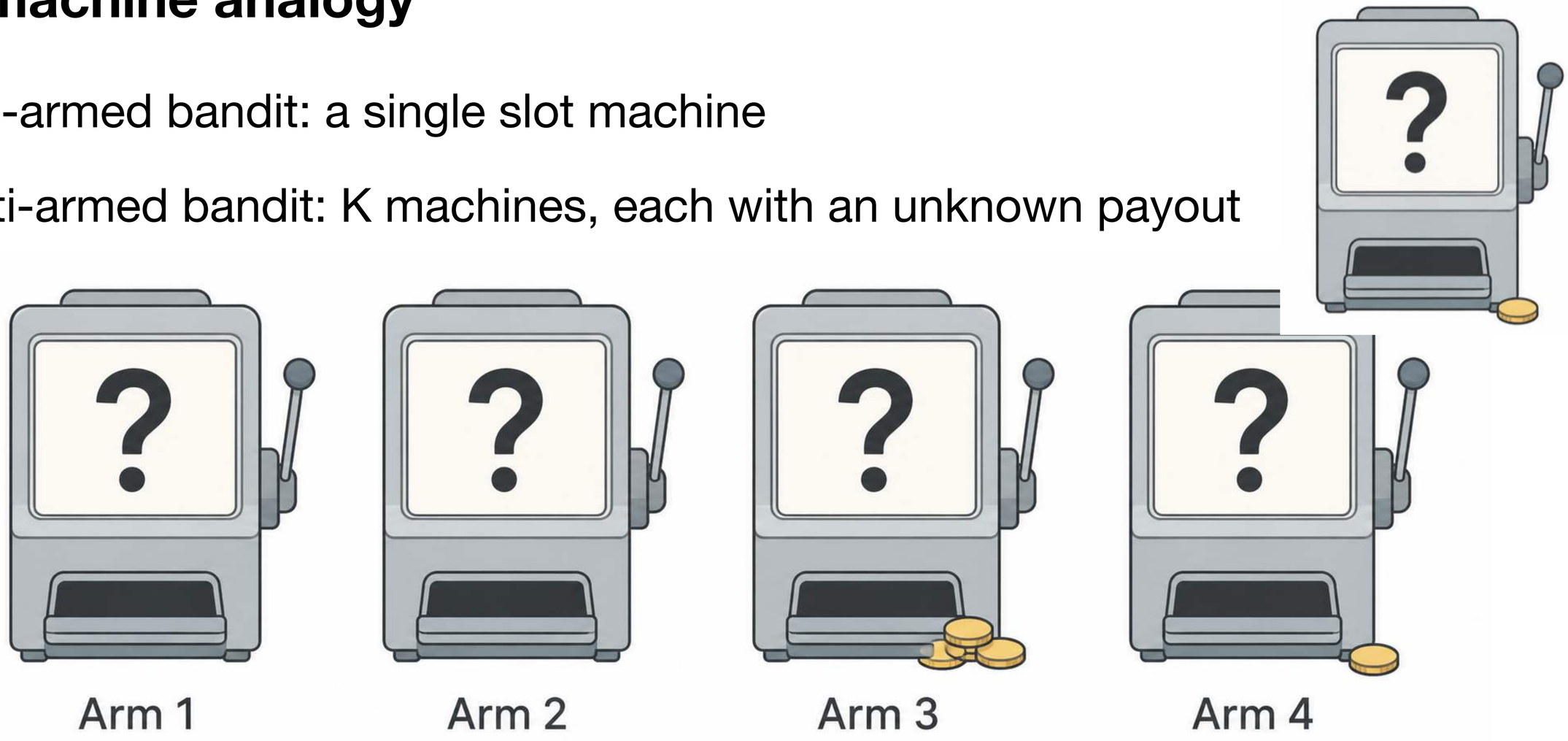




Multi-Armed Bandit

Slot machine analogy

- One-armed bandit: a single slot machine
- Multi-armed bandit: K machines, each with an unknown payout

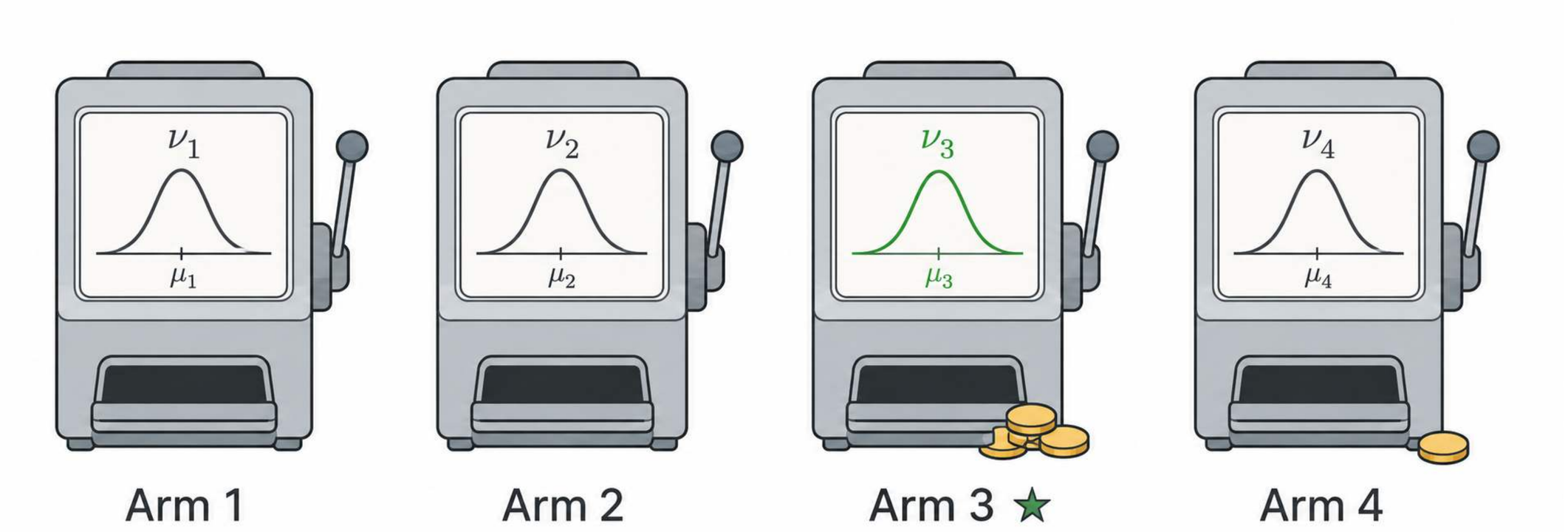


Each arm has an unknown reward distribution

Stochastic Multi-Armed Bandit

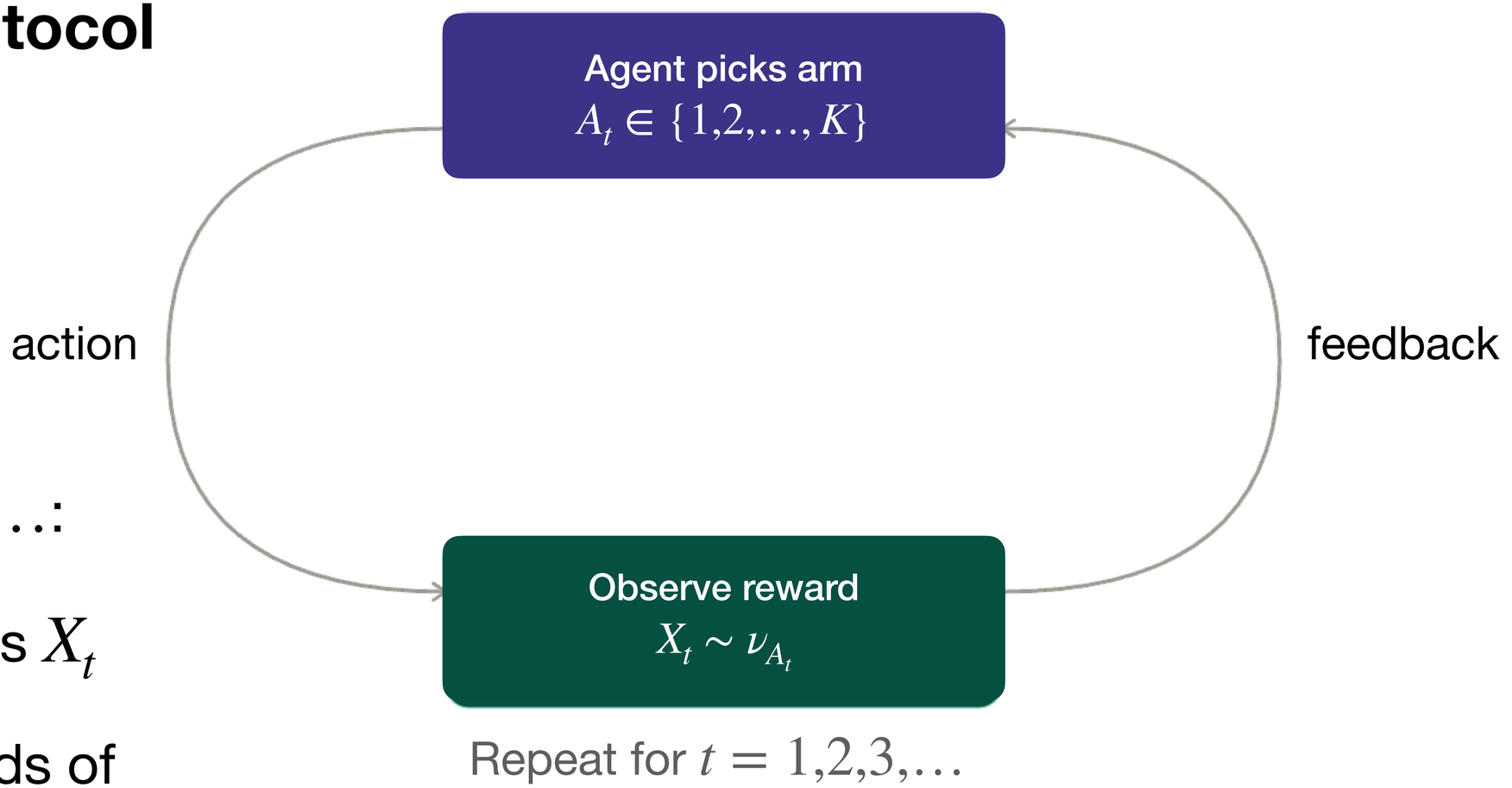
Formal setup

- K arms; arm i has reward distribution ν_i with mean μ_i
- Best arm: $i^* = \arg \max_{i \in \{1,2,\ldots,K\}} \mu_i$



Stochastic Multi-Armed Bandit

The interaction protocol



- At each round $t = 1,2,\ldots$:
- agent picks A_t, observes X_t
- Bandit feedback: rewards of un-pulled arms are not seen

Goal: use observed rewards to make better choices over time

Multi-Armed Bandit

Key Challenges

- Reward is random. Need to pull arm i a few times to estimate μ_i.
- Initially uncertain. Moreover, if μ_i is time-varying, then never fully certain
- Since we have bandit feedback (reward observed only from pulled arms), we must pull each arm a few times to learn the distribution
  - Essential to make good decisions confidently
    - Cost of exploration

Mapping Back

Clinical Trials → Best Arm Identification

- Arms = possible treatments
- Pulling arm i = giving treatment i to a patient
- Reward X_t = patient outcome (measure of how well patient recovered)
- μ_i = true efficacy of treatment i

Objective: best-arm identification

- Find i^* with high confidence, using as few patients as possible
- Bad pulls *during* the trial are tolerated —cost of doing the trial properly
- Will cover in lecture 2

Mapping Back

News Recommendation → Regret Minimisation

- Arms = candidate news items
- Pulling arm i = showing item i to a user
- Reward X_t = click (**1**) or no-click (**0**)
- μ_i = click-through rate of item i

Objective: regret minimisation

- $\text{Regret}(T) = T\mu^* - \sum_t X_t$
- Random selection: linear regret in T
- Goal: sublinear regret
  - every missed click costs us, so learning and earning must happen together

Common Themes

- Both objectives → exploration-exploitation tension
- Both objectives → measuring uncertainty via confidence intervals
- Both objectives → hardness depends on the suboptimality gap $\Delta_i = \mu^* - \mu_i$

Some Simple Algorithms

Two Naive Strategies

Pure Greedy

- Pull each arm a few times (n)
  - Calculate empirical mean of each arm $\hat{\mu}_i = (\Sigma_{t:A_t=i} X_t)/n$
  - Forever pull the empirically best arm: $\arg \max_{i \in \{1,2,\ldots,K\}} \hat{\mu}_i$
- *Failure mode*: some unlucky early pulls can lock you out of the true best arm forever

Pure Random

- Pull a uniformly random arm at every round
  - Equivalently, pull arms in round-robin fashion
- $\hat{\mu}_i \to \mu_i$, but regret grows linearly in T
- *Failure mode:* regret grows linearly in T

ε-Greedy Algorithm

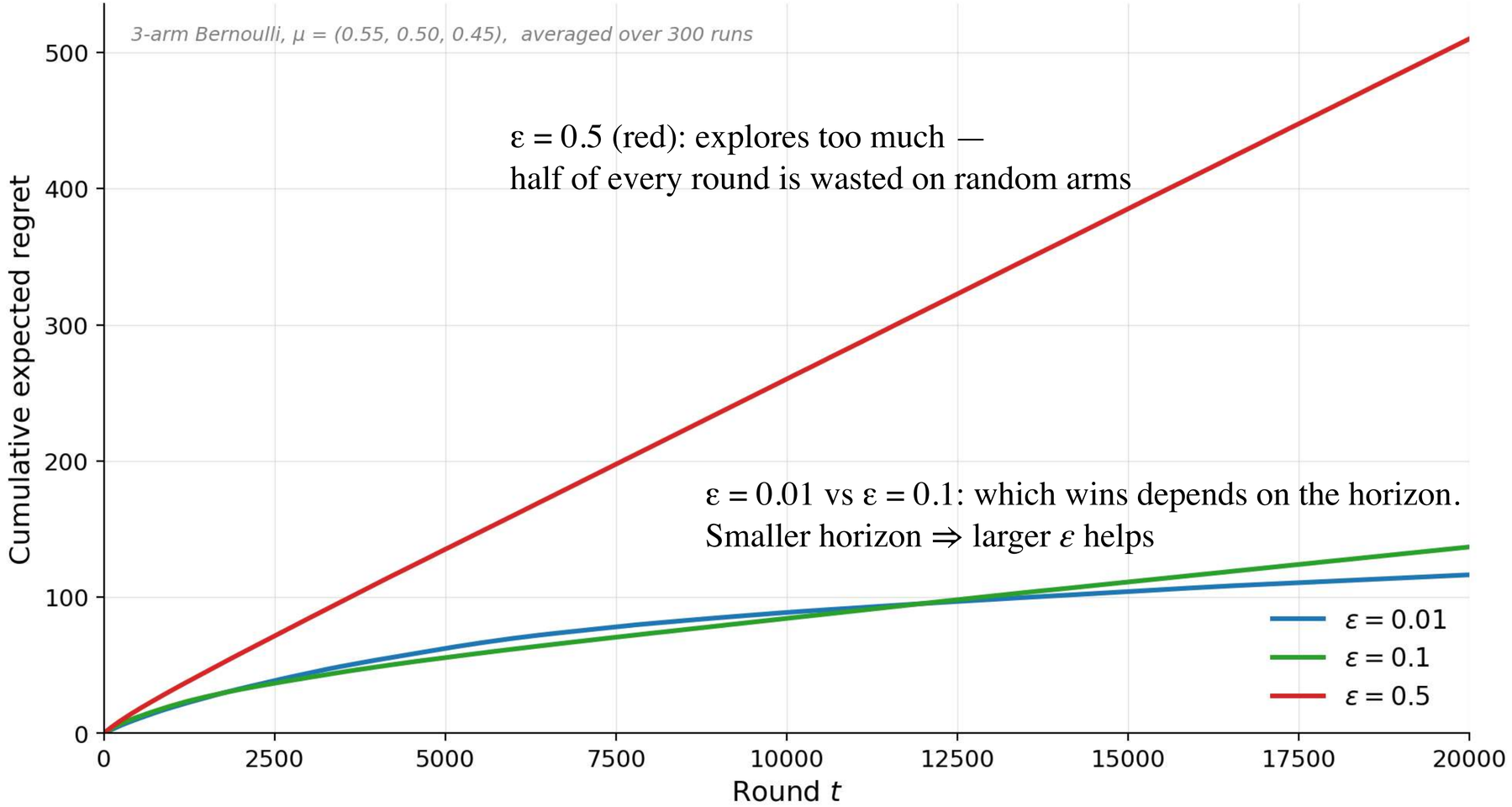
A better solution

- With probability ε: pull a uniformly random arm (explore)
- With probability $1 - \varepsilon$: pull the empirically best arm (exploit)
- $\varepsilon \in (0,1)$ is the exploration rate — a knob you tune

The simplest algorithm that does both exploration and exploitation by combining pure random and pure greedy strategies

ε-Greedy Algorithm

Behaviour and Limitations



All three curves keep growing — any fixed ε gives linear regret

No fixed ε is universally best

We want exploration that adapts: more when uncertain, less when confident. Wait for Lecture 3!

Coming Up Next

- Lecture 2: best arm identification
- Lectures 3&4: algorithms for regret minimisation: UCB, Thompson sampling
- Lecture 5: practical aspects of bandits (Spotify case-study)
- Lectures 6, 7, & 8: linear bandits (with contextual side-information)
- Lectures 9 & 10: practical aspects of bandits (Udemy case-study)
- Lectures 11, 12, &13: more advanced bandit models
- Lecture 14: recap and bridge to reinforcement learning

Course Logistics

Grading

- Two programming assignments
  - Assignment 1: covers lectures 1 - 5 (15%)
    - Will release end of first week, due by end of third week
  - Assignment 2: covers lectures 6 - 10 (15%)
    - Will release end of fourth week, due end of sixth week
- One comprehensive midterm (online, 28th June, Sunday) (20%)
- Reinforcement learning (50%)