Online Learning Applications

Part 1: Introduction to Online Learning

Online learning

A learner interacts repeatedly with an **unknown** environment. During each interaction:

- The learner chooses **actions**
- The environment returns a **reward**
- The environment returns some **feedbacks**

The goal of the learner is to maximize its happiness (**reward**).

Learner

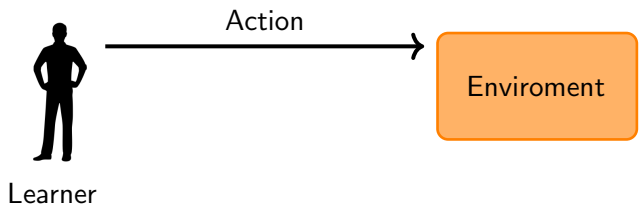
Enviroment

Online learning

A learner interacts repeatedly with an **unknown** environment. During each interaction:

- The learner chooses **actions**
- The environment returns a **reward**
- The environment returns some **feedbacks**

The goal of the learner is to maximize its happiness (**reward**).

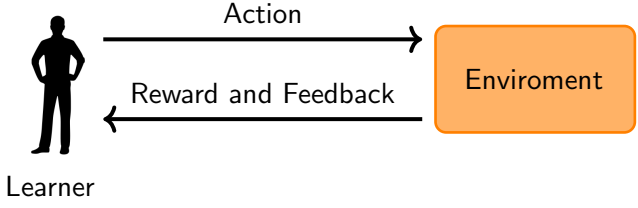


Online learning

A learner interacts repeatedly with an **unknown** environment. During each interaction:

- The learner chooses **actions**
- The environment returns a **reward**
- The environment returns some **feedbacks**

The goal of the learner is to maximize its happiness (**reward**).



Online learning

A learner interacts repeatedly with an **unknown** environment. During each interaction:

- The learner chooses **actions**
- The environment returns a **reward**
- The environment returns some **feedbacks**

The goal of the learner is to maximize its happiness (**reward**).

Learner

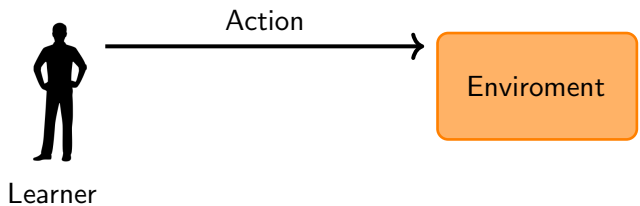
Enviroment

Online learning

A learner interacts repeatedly with an **unknown** environment. During each interaction:

- The learner chooses **actions**
- The environment returns a **reward**
- The environment returns some **feedbacks**

The goal of the learner is to maximize its happiness (**reward**).

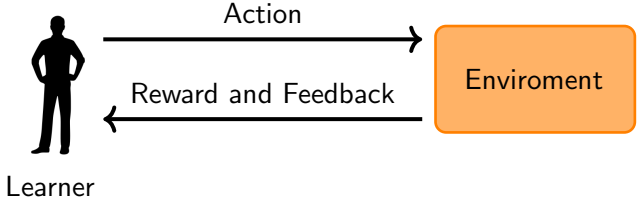


Online learning

A learner interacts repeatedly with an **unknown** environment. During each interaction:

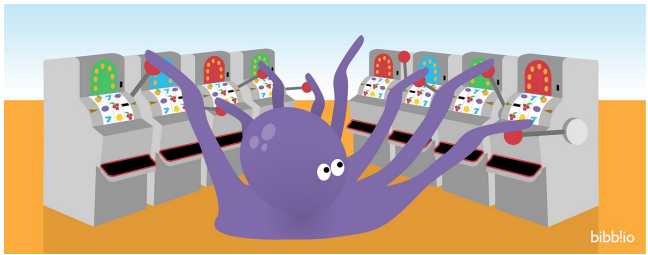
- The learner chooses **actions**
- The environment returns a **reward**
- The environment returns some **feedbacks**

The goal of the learner is to maximize its happiness (**reward**).



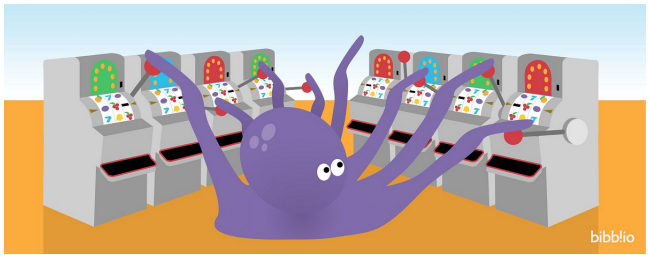
A toy multi-armed bandit problem

- k slot machines
- Each slot machine is different but we don't know its property (e.g., winning probability or the amount of the win)
- At each round of the interaction:
  - We choose a slot machine (**action**)
  - We might win some money (**reward**)
  - The only **feedback** we get is the amount of the win



A toy multi-armed bandit problem

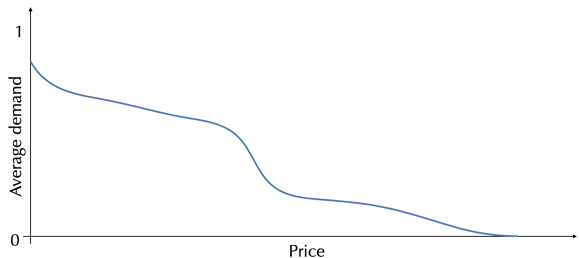
- k slot machines
- Each slot machine is different but we don't know its property (e.g., winning probability or the amount of the win)
- At each round of the interaction:
  - We choose a slot machine (**action**)
  - We might win some money (**reward**)
  - The only **feedback** we get is the amount of the win



- **Finite set of possible actions**

Pricing

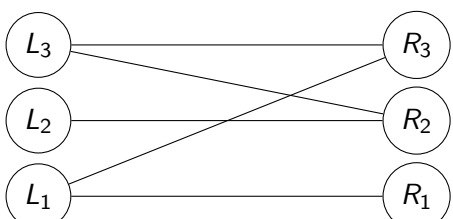
- A seller has (an unlimited number of units of) a product to sell
- When a potential buyer arrives:
  - The seller chooses a price (**action**)
  - The seller receives as **reward** the price if the buyer buys
  - The only **feedback** the seller gets is if the buyer buys or not



- **Continuous set of actions**

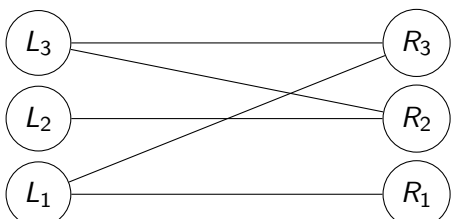
Matching

- Two sets of items to match
- At each round two sets of items $\mathcal{S}_1$ and $\mathcal{S}_2$ arrives:
  - Choose a matching, i.e., match each item from $\mathcal{S}_1$ with at most an item from $\mathcal{S}_2$ (**action**)
  - The **reward** depends on the quality of the matching
  - I can receive as **feedback** only the cumulative reward, or the reward of each matched couple, or ...



Matching

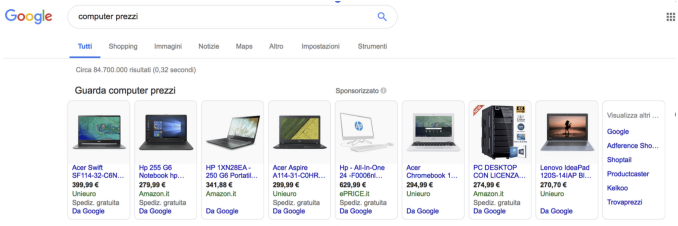
- Two sets of items to match
- At each round two sets of items $\mathcal{S}_1$ and $\mathcal{S}_2$ arrives:
  - Choose a matching, i.e., match each item from $\mathcal{S}_1$ with at most an item from $\mathcal{S}_2$ (**action**)
  - The **reward** depends on the quality of the matching
  - I can receive as **feedback** only the cumulative reward, or the reward of each matched couple, or ...



- **Combinatorial set of actions**

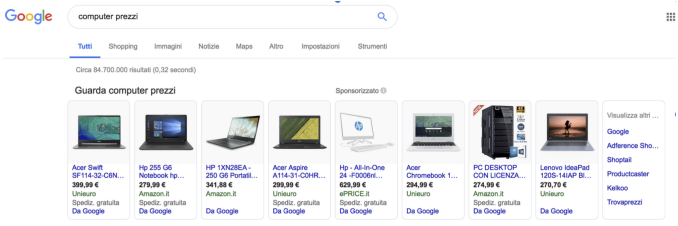
Online advertising

- An advertiser runs an advertising campaign on the web
- The advertiser has a **budget** B
- Every time a new user arrives on the web pages:
  - The advertiser choose a **bid**
  - If the advertiser's bid is the highest among the advertisers, their ad is shown
  - If the advertiser's ad is shown, the advertiser gets some **reward** and **pays** their bid
- The advertiser stops when the budget is depleted



Online advertising

- An advertiser runs an advertising campaign on the web
- The advertiser has a **budget** B
- Every time a new user arrives on the web pages:
  - The advertiser choose a **bid**
  - If the advertiser's bid is the highest among the advertisers, their ad is shown
  - If the advertiser's ad is shown, the advertiser gets some **reward** and **pays** their bid
- The advertiser stops when the budget is depleted



- **Budget constraints**

Online learning framework

At each round $t \in \{1, \dots, T\}$:

- The learner chooses an action $a_t \in A$
- The environment chooses a loss function $\ell_t : A \to [0, 1]$

Online learning framework

At each round $t \in \{1, \dots, T\}$:

- The learner chooses an action $a_t \in A$
- The environment chooses a loss function $\ell_t : A \to [0, 1]$
- **We are not assuming that this loss function is stochastic. We will consider both the stochastic and adversarial setting**

Online learning framework

At each round $t \in \{1, \dots, T\}$:

- The learner chooses an action $a_t \in A$
- The environment chooses a loss function $\ell_t : A \to [0, 1]$
- **We are not assuming that this loss function is stochastic. We will consider both the stochastic and adversarial setting**
- The learner suffers a loss $\ell_t(a_t)$
- The learner might receive an additional feedback

We will see two types of feedback:

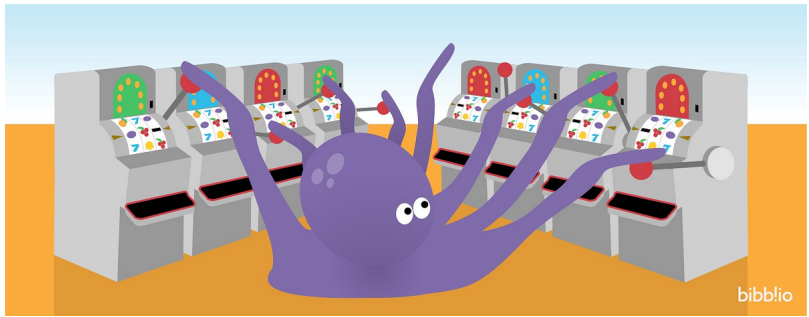
- **Full** feedback
- **Bandit** feedback

Goal

The learner goal is to minimize the cumulative loss. Equivalently, if we set the reward $r(a) = 1 - \ell(a)$, the goal is to maximize the reward.

Toy problem (formal)

- We can choose among k slot machines $a_1, \ldots, a_K$
- At each round $t = 1, \ldots, T$:
  - We choose a slot machine a_t
  - We receive a reward (money) $r_t(a_t) = 1 - \ell_t(a_t)$
  - The only feedback that we get is the amount of the win

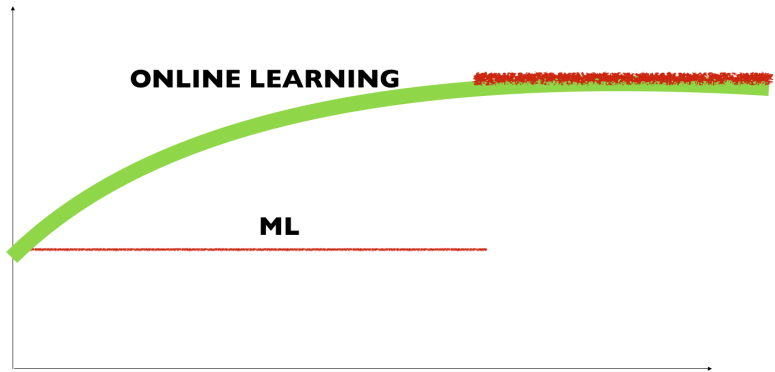


Online learning vs machine learning

Differently from classical machine learning problem:

- Data arrives sequentially
- We must take decision during learning
- We cannot divide the learning in a training and learning phase
- We do not (always) assume that the environment is stochastic

Online learning vs machine learning



Performance measure

- We compare the performance of the learner with a baseline
- A natural baseline is the best **fixed** action

Performance measure

- We compare the performance of the learner with a baseline
- A natural baseline is the best **fixed** action $\rightarrow$ **this is not the optimal strategy in non-stochastic settings**

Performance measure

- We compare the performance of the learner with a baseline
- A natural baseline is the best **fixed** action $\rightarrow$ **this is not the optimal strategy in non-stochastic settings**
- We measure the performance of the algorithm with the regret
- The **regret** measures how much I've lost using the learner instead of always playing the optimal fixed action

Regret

We define the **expected regret** R_T of an algorithm as:

$$R_T = \mathbb{E}\left[\sum_{t\in[T]} \ell_t(a_t)\right] - \min_{a\in A} \sum_{t\in[T]} \ell_t(a),$$

where the expectation is on the randomness of the algorithm.

No-regret algorithms

Our goal is to design learners that guarantee that the regret grows **sublinearly** in T.

No-regret Algorithm

An algorithm is **no-regret** if:

$$\lim_{T \to \infty} \frac{R_T}{T} = 0.$$

A no-regret algorithm guarantees that:

- On average the regret at each round goes to 0
- We "converge" to the optimal solution

No-regret algorithms

Our goal is to design learners that guarantee that the regret grows **sublinearly** in T.

No-regret Algorithm

An algorithm is **no-regret** if:

$$\lim_{T \to \infty} \frac{R_T}{T} = 0.$$

A no-regret algorithm guarantees that:

- On average the regret at each round goes to 0
- We "converge" to the optimal solution $\rightarrow$ **this is meaningful only in stochastic settings**

Online learning with a finite set of actions

- In the first part of the course, we will focus on the simplest online leaning problem in which the agent must choose among a **finite set** of actions $\{a_1, \ldots, a_K\}$
- Dealing with this simple problem is sufficient to highlight the main challenges and techniques of online learning problems
- This problem is called **expert problem** or **Multi-Armed bandit** (MAB) depending on the feedback
- In the following parts of the course, we will consider more general problems that include:
  - Continuous action sets (**pricing**)
  - Budget constraints (**advertising**)
  - Combinatorial action sets (**matching**)

Stochastic environment

In the stochastic setting:

- At each time $t \in [T]$, the loss $\ell_t(a)$ of an action a is sampled from a distribution $\mathcal{D}_a$
- We denote with $\hat{\ell}(a)$ the expected loss of an arm a
- We can use a simpler (and almost equivalent) notion of regret called **pseudo-regret**

Pseudo-regret

The pseudo-regret of an algorithm is:

$$\mathcal{R}_T = \mathbb{E}\left[\sum_{t\in[T]} \hat{\ell}(a_t)\right] - T \min_a \hat{\ell}(a),$$

where the expectation is on the randomness of the algorithm.

Adversarial environment

In the adversarial setting:

- At each time $t \in [T]$, we don't make any assumption on the loss $\ell_t(a)$ of an action a

Adversarial environment

In the adversarial setting:

- At each time $t \in [T]$, we don't make any assumption on the loss $\ell_t(a)$ of an action a

Some clarifications:

- Adversarial loss means that we do not assume that the loss are stochastic
- It does not mean that an adversary is trying to maximize the learner loss (this can be done setting $\ell_t(a) = 1$ for each a and t)

Full feedback (expert problem)

Under full feedback, after the learner plays an action, it observes the losses of all the actions:

- At each time $t \in [T]$, after the learner played an action a, it observes the loss function $\ell_t(\cdot)$
- This problem is called **expert problem**

Full feedback is strong since:

- The learner knows the losses of all the actions even if it doesn't play them
- The learner doesn't have to "explore" since the feedback is independent from the played action

Bandit feedback (MAB)

Under bandit feedback, the learner observes only the loss of the action that it played:

- At each time $t \in [T]$, after the learner played an action a, it only observes the loss $\ell_t(a)$
- This problem is called **Multi-Armed Bandit** (MAB)

Bandit feedback is weaker than full feedback since:

- The learner knows the only the loss of the action that it played
- The learner doesn't know the loss of the other actions
- The learner has to "explore", playing (possibly suboptimal) actions to learn their losses

Classes of problems

In the following, we will study the following three problems:

- Stochastic MAB
- Adversarial expert problem
- Adversarial MAB (hardest)

Classes of problems

In the following, we will study the following three problems:

- Stochastic MAB
- Adversarial expert problem
- Adversarial MAB (hardest)

We will not study the Stochastic expert problem since can be addressed optimally using the techniques for the other problems (e.g., the stochastic MAB or the adversarial expert problem)