Online Optimization & Learning in Games

Chen-Yu Wei

Platforms

- Course webpage: https://bahh723.github.io/oolg2026fa/
  - Syllabus, announcement, slides, lecture recordings, references
  - Can be accessed from my personal website
- Gradescope (haven't created)
  - Homework submission
- Piazza (haven't created)
  - Questions and discussions
- Canvas
  - We will **NOT** use canvas

Topics in This Course

- Online Learning in Non-Stationary / Adversarial Environments
- Games in Learning
- Learning in Games

Prerequisites

- Probability
- Linear algebra
- Calculus
- Machine learning
- Convex analysis
  - Definitions and Properties of **convex functions** and **convex sets**
- Mathematical maturity. For example,
  - Understand definitions precisely; knows what is assumed and what needs a proof
  - Able to produce mathematical proofs unambiguously and logically
  - Able to to tell the correctness of a proof / generate counterexamples

  (this course might be a chance to develop some of it)

Course Components: Problem Sets (50%)

- **4** problem sets that requires proving theorems mathematically
- The problems are not merely to reinforce the course materials
  - It contain things not covered in the class which will be used in later lectures
  - The assignments should be treated as part of the course materials
- Free to discuss with classmates or AI
  - Still, it is encouraged to tackle the problems first by yourself
- Late policy
  - Each late day results in 10% deduction in the grade
  - The homework cannot be submitted 3 days after the deadline (resulting in 0 points)
- Submission
  - Latex or hand-writing are both fine
  - Convert to PDF and submit to Gradescope

Course Components: Two Quizzes (10%)

- The first 40 minutes in two of the lectures
  - The quiz date will be announced at least 1 week before the quiz in the lecture, through piazza announcement, and on the website
- Each quiz will cover
  - ~75%: understanding of the homework problems whose solution has been released before the quiz dates
  - ~25%: understanding of the central concepts in the lectures
- If you have to miss the exam due to extenuating circumstances (e.g., illness, family emergency), we will arrange one (and only one) make-up session in the same week

Course Components: Project (30%)

- Grouping constraints will be announced roughly at the third week (after the enrollment list is settled)
- We will assign each group papers to read.
- Each group is required to give a 25 minutes (including QA) presentation on the paper, and write a report.
- More specification will be announced later.

Course Components: Participation (8%), Evaluation (2%)

- Participating in each student presentation session earns 1 point
- The remaining 5 points are based on the following criteria:
  - ≥ 1 if attendance rate ≥ 30%
  - ≥ 2 if attendance rate ≥ 50%
  - ≥ 3 if attendance rate ≥ 80% or if you have occasional interaction in the class (not including final presentation) or piazza
  - = 5 if you have very active interaction in the class or piazza

- Completing the course evaluation at the semester end earns 2 points.

Resources

Course materials of

- Introduction to Online Optimization/Learning at USC by Haipeng Luo
  - Overlapping with the first 1/3 of our course
- Learning in Games (And Games in Learning) at Upenn by Aaron Roth
  - Overlapping with the first 2/3 of our course

Topics in This Course

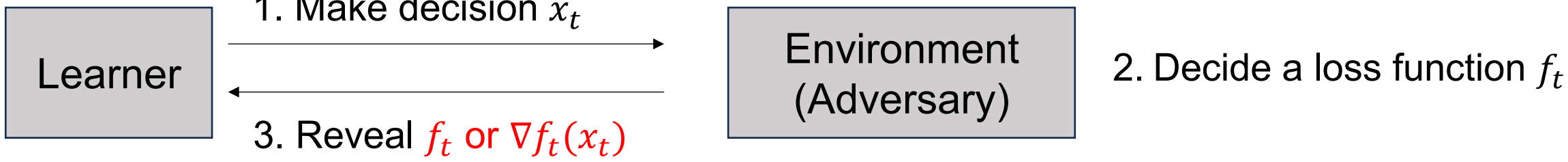
- Learning in Non-Stationary / Adversarial Environments
- Games in Learning
- Learning in Games

Online Learning in Non-Stationary / Adversarial Environments

In each round $t = 1,2, \ldots$

Online Adversarial Learning formulation:

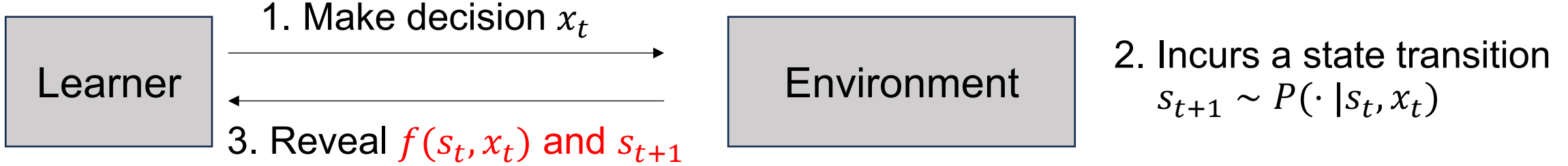
Challenges: adversary



Learner suffers loss $f_t(x_t)$ in round t

c.f. **Reinforcement Learning** formulation:

Challenges: partial feedback, long-term dependence



Learner suffers loss $f(s_t, x_t)$ in round t

What Does “Online” Mean?

- Online learning mostly means **interactive learning:** the learner repeatedly makes a decision, observes feedback, and updates its strategy.
- Online learning is also commonly associated with—but not necessarily—the following:
  - **Computation:** Each new feedback only requires incremental updates to the model
  - **Evaluation:** We care about the learner’s cumulative performance during the learning process (not just the final output model)
  - **Environment:** The environment may be non-stationary or adversarial
- In this course, we adopt **cumulative evaluation** and **adversarial environment.**

Formalization: Online Convex Optimization

Given: convex set $\mathcal{X} \subset \mathbb{R}^d$

For round $t = 1, 2, \ldots, T$:

Learner makes a decision $x_t \in \mathcal{X}$

Adversary decides a convex function $f_t: \mathcal{X} \to \mathbb{R}$ (may depend on x_t)

Adversary reveals f_t or $\nabla f_t(x_t)$ to the learner

The learner tries to minimize $\sum_{t=1}^{T} f_t(x_t)$ (more on this later)

Why care about the “adversarial” case?

One direct application is for truly non-stationary or adversarial environment:

- Recommendation systems face changing preference of the users
- Cybersecurity attackers adapt their behavior while the systems update over time.

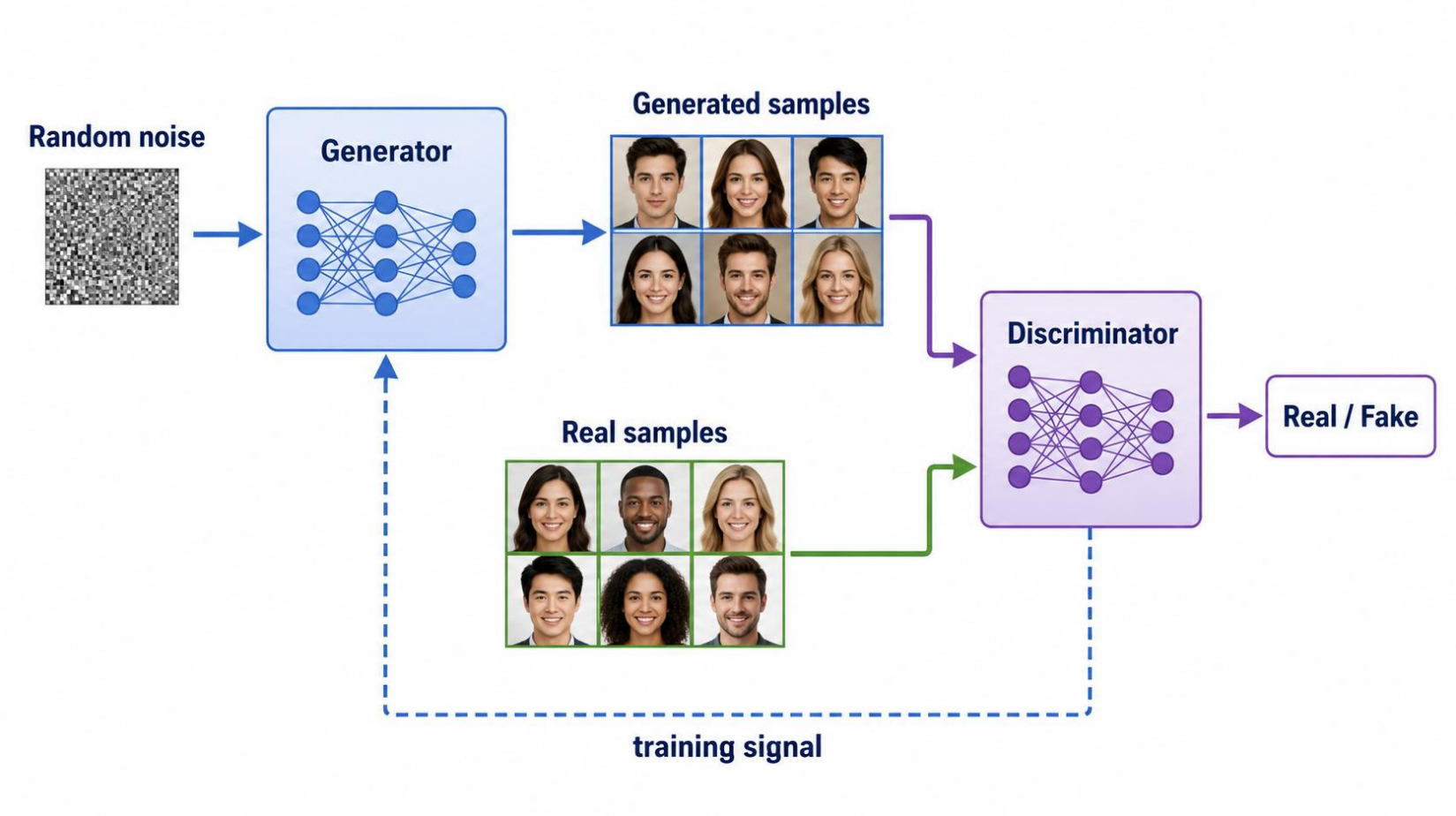
But the adversarial online learning framework has much more implications than it might seem.

Games in Learning

(Creating an adversary)

Learning distribution matching

- Generative Adversarial Network (GAN)

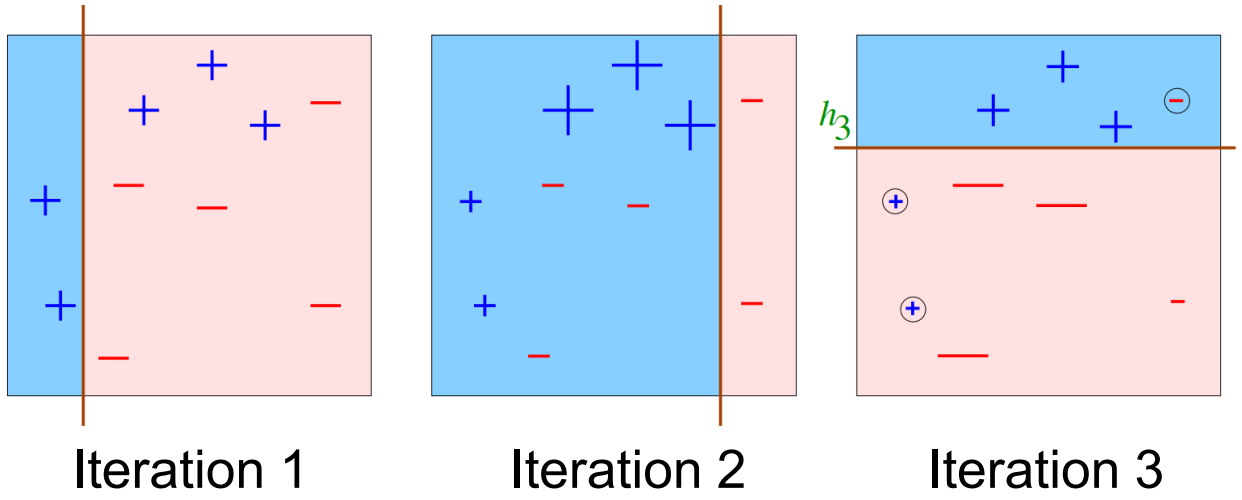


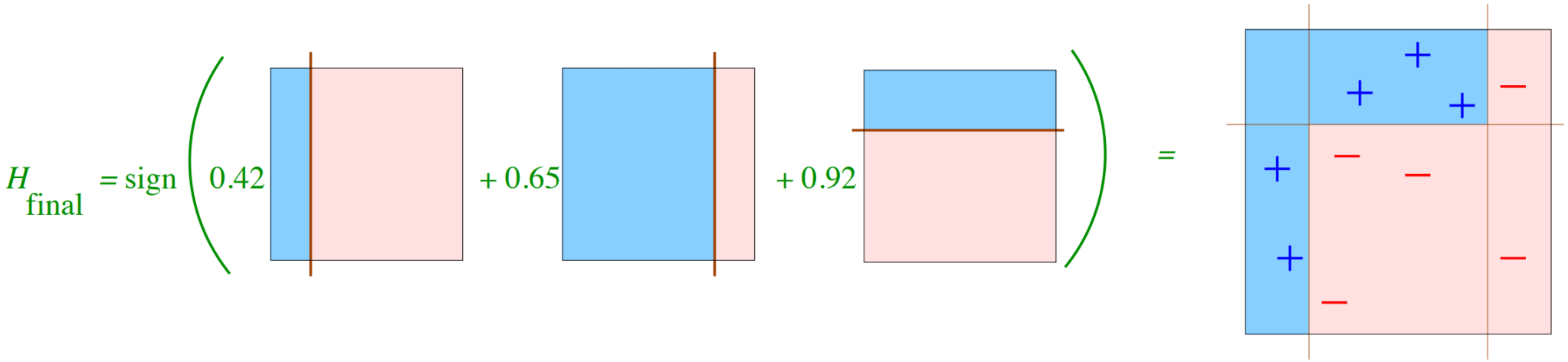
Boosting weak learners

Schapire's slides: toy example of AdaBoost

- Boosting

  Player 1: classifier
  Player 2: weight update

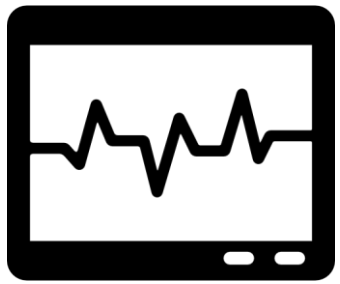




Learning with constraints

Patients

Feature

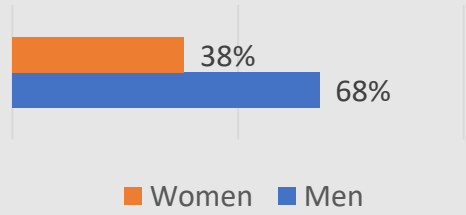


Diagnostic model

Prediction

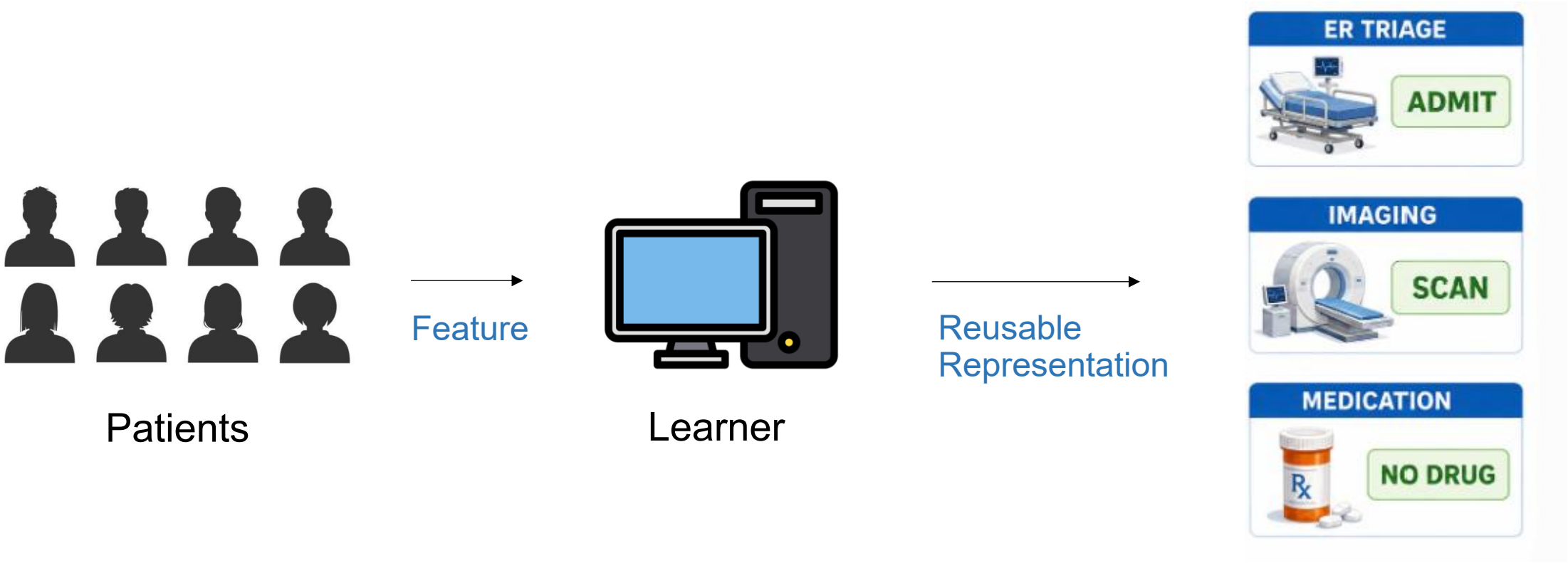
Further test / No further test

Fairness auditor



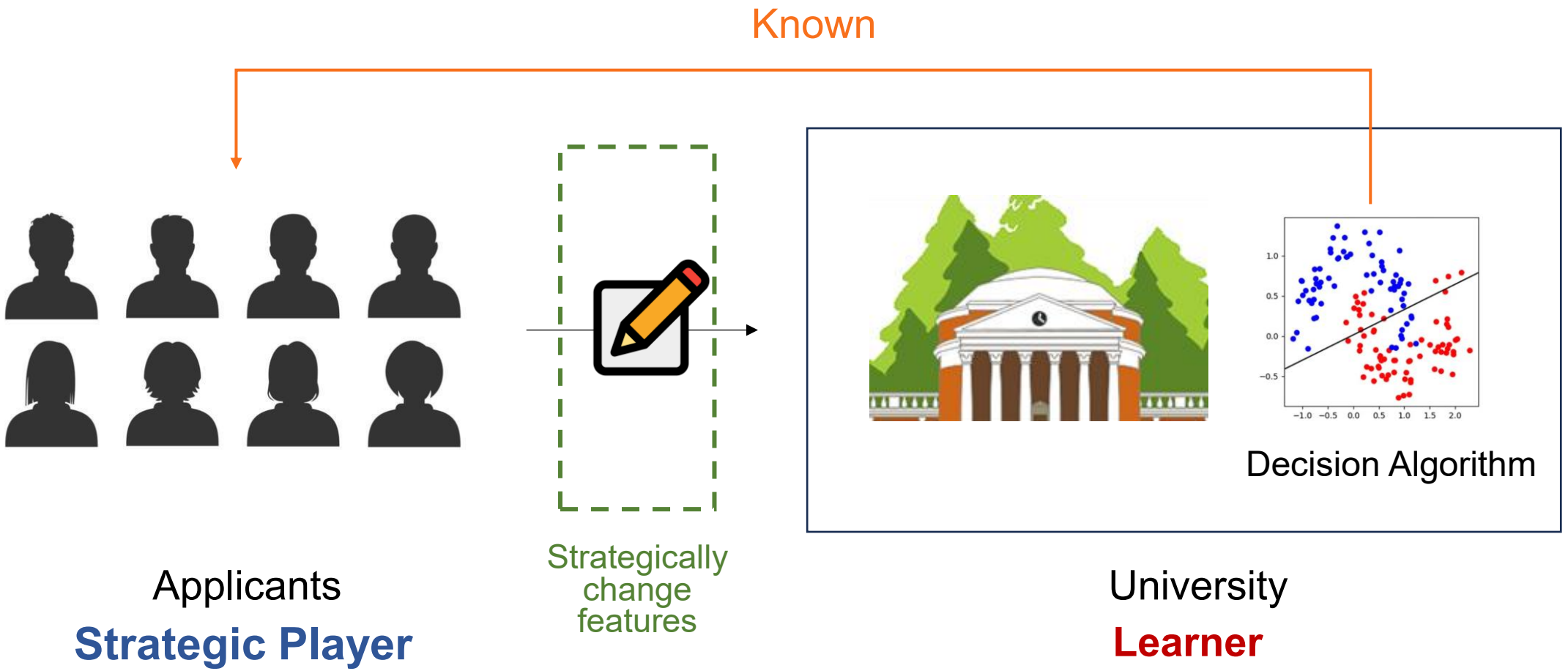
gender / age / race …

Learning for multiple downstream tasks / objectives



Learning in Games

Learning against Strategic Players



Algorithmic Collusion

- **Duopoly / Oligopoly:** Two or few companies dominate the market, selling the same or highly substitutable products
- Collusion: when the price is significantly higher than the price at equilibrium
- How can an auditor (government) detect such scenario?
  - Of course, it's difficult for the auditor to calculate the equilibrium value

Applications of adversarial online learning

- Learning in non-stationary / adversarial environments
- Learning to match distributions
- Ensemble (boosting)
- Learning with fairness constraints
- Learning for multiple downstream tasks
- Robustness to strategic manipulations

…

Surprisingly, they can all be reduced to the clean formulation:

No-regret learning for online convex optimization

No-regret learning for online convex optimization

Online Convex Optimization

Given: convex set $\mathcal{X} \subset \mathbb{R}^d$

For round $t = 1, 2, \ldots, T$:

Learner makes a decision $x_t \in \mathcal{X}$

Adversary decides a convex function $f_t: \mathcal{X} \to \mathbb{R}$ (may depend on x_t)

Adversary reveals f_t or $\nabla f_t(x_t)$ to the learner

$$\textbf{Regret} := \sum_{t=1}^{T} f_t(x_t) - \min_{x \in \mathcal{X}} \sum_{t=1}^{T} f_t(x)$$

A **no-regret algorithm** is an algorithm that ensures $\textbf{Regret} = \boldsymbol{o(T)}$ against any adversarial choice of $f_1, f_2, \ldots, f_T$.

What does $\text{Regret} = o(T)$ mean?

$$\text{Regret} = o(T) \quad \Rightarrow \quad \underbrace{\frac{1}{T}\sum_{t=1}^{T} f_t(x_t)}_{\text{Total loss of the learner}} \leq \underbrace{\min_{x \in \mathcal{X}} \frac{1}{T}\sum_{t=1}^{T} f_t(x)}_{\text{Total loss of the hindsight player}} + o(1)$$

Total loss of the learner: Can choose different decisions in different rounds, but doesn't know f_t before taking x_t

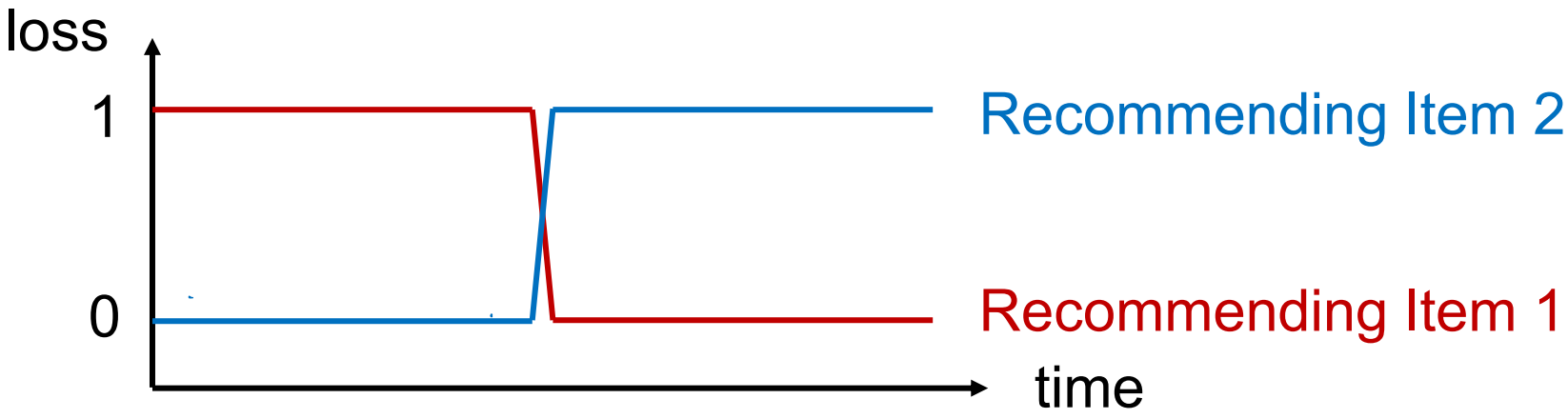
Total loss of the hindsight player: Knows $f_1, \ldots, f_T$, but needs to make a fixed decision x in all rounds

The learner is (asymptotically) no worse than the hindsight player

Does This Regret Definition Make Sense?

- No-regret algorithm is a cornerstone of this course, so, of course, it will be useful.
- But why do we compare the learner's performance with a benchmark that *chooses a fixed decision*?
- Recall the topics of this course
  - Learning in non-stationary environments
  - Games in learning (constraints, multi-objective, boosting, …)
  - Learning in games (other strategic players)

$o(T)$ Regret ⇒ Learning in Non-Stationary Environments?



Ideal behavior: Tracking the best decision over time **(Get 0 total loss)**

But to achieve $o(T)$ regret, the learner only needs to, e.g.,

- Always choose the same item
- Always uniformly randomly choose one item

} **(Get $\frac{T}{2}$ total loss)**

$$\textbf{Regret} := \sum_{t=1}^{T} f_t(x_t) - \min_{x \in \mathcal{X}} \sum_{t=1}^{T} f_t(x)$$

$o(T)$ Regret ⇒ Learning in Non-Stationary Environments?

In these non-stationary environments, maybe a more reasonable regret definition is

$$\text{(Dynamic) Regret} := \sum_{t=1}^{T} f_t(x_t) - \sum_{t=1}^{T} \min_{x \in \mathcal{X}} f_t(x) \quad ??$$

But this cannot be $o(T)$ in general…

But At least, Regret makes sense in stationary environments

When $f_t = f$ for all time t:

$$\text{Regret} = \sum_{t=1}^{T} f(x_t) - \min_{x \in \mathcal{X}} \sum_{t=1}^{T} f(x)$$

$$= \sum_{t=1}^{T} (f(x_t) - f(x^\star)) \qquad \text{where } x^\star \text{ is the minimizer of } f$$

$\text{Regret} = o(T) \quad \Rightarrow$ Learner's average loss approaches $\min_x f(x)$

Short Conclusion

- Regret may not (yet) make sense for non-stationary environments
- But it makes perfect sense for stationary environments

Roadmap:

- We are going to ignore its meaning for now, and focusing on introducing and analyzing algorithms achieving $\mathrm{Regret} = o(T)$ anyway.
- The implication of Regret will emerge in ~2 weeks
- That said, all algorithms should be intuitive in the special case $f_t = f \ \ \forall t$