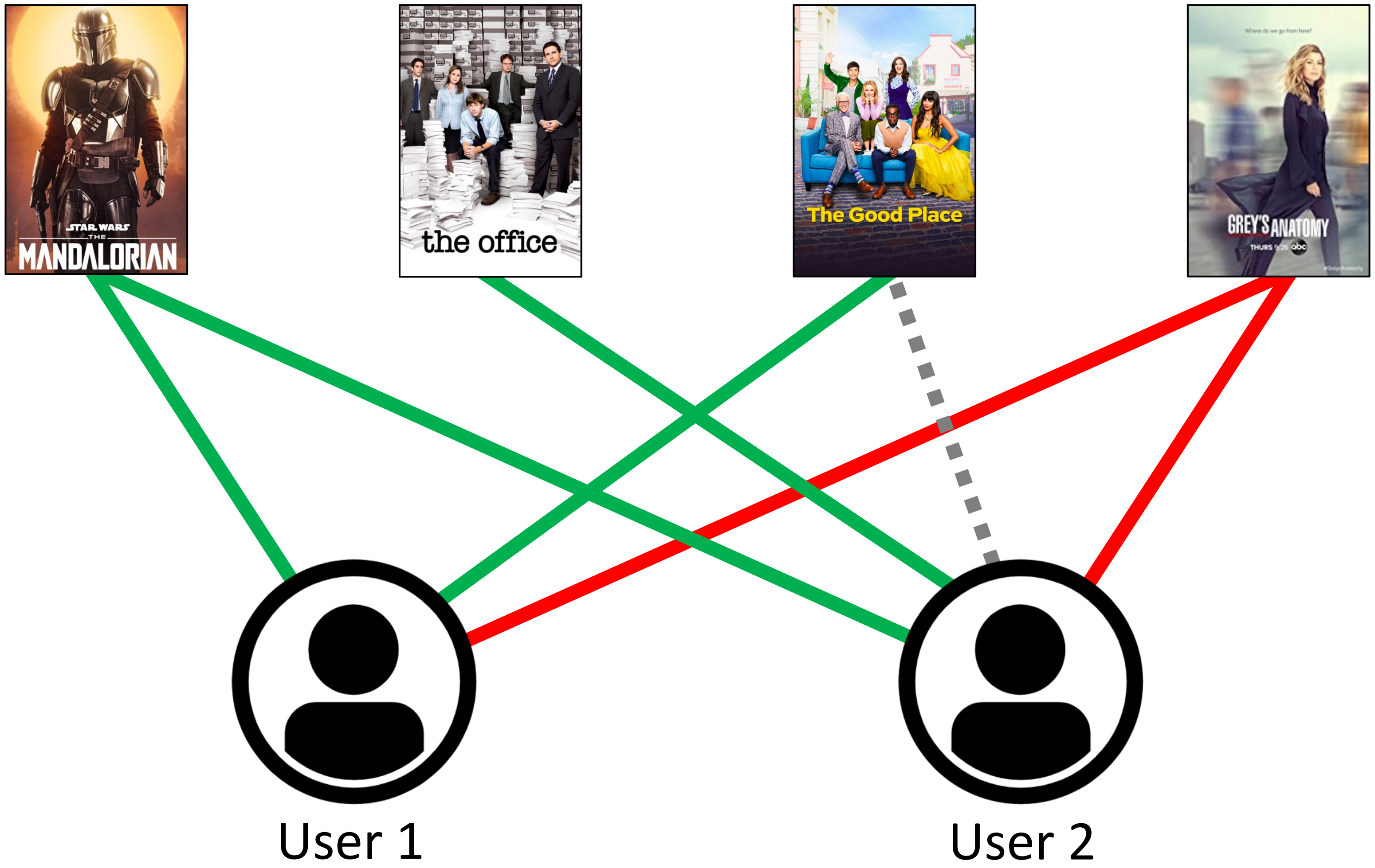
Announcements

- Quiz 11 is due **Thursday, December 1 at 8pm**
- HW 6 due **Friday, December 2 at 8pm**
  - 2 day extension

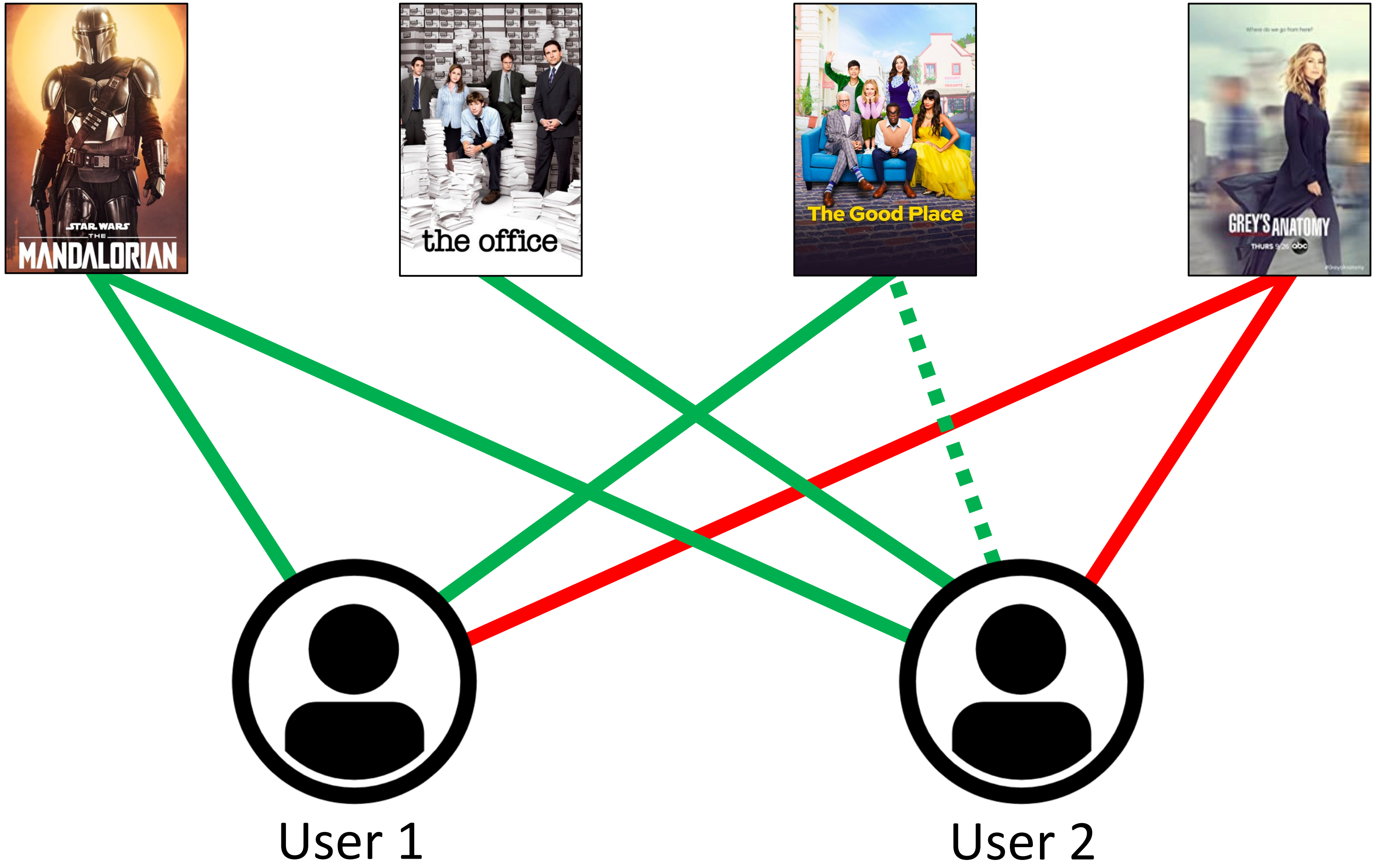
Recommender Systems

- **Media recommendations:** Netflix, Youtube, etc.
- **News feed:** Google News, Facebook, Twitter, Reddit, etc.
- **Search ads:** Google, Bing, etc.
- **Products:** Amazon, ebay, Walmart, etc.
- **Dating:** okcupid, eharmony, coffee-meets-bagel, etc.

Collaborative Filtering



Collaborative Filtering



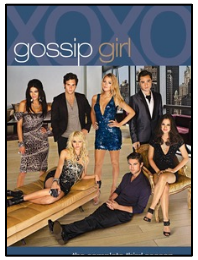
Collaborative Filtering

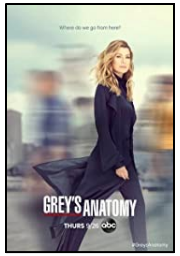
- **Given:**
  - Matrix $X_{i,k} = \begin{cases} \text{rating}_{i,k} & \text{if user}_i \text{ rated product}_k \\ \text{N/A} & \text{otherwise} \end{cases}$
  - Assume fixed set of n users and m products
  - **Not given any information about the products!**

- **Problem:** Predict what $X_{i,k}$ would be if it is observed
  - Not quite supervised or unsupervised learning!

Collaborative Filtering

Missing entries!

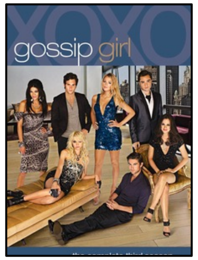


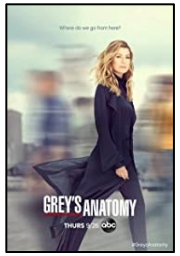


	Gossip Girl	The Office	The Mandalorian	Criminal Minds	The Good Place	Grey's Anatomy	...
Grace	4	5	4	1	5	3	...
Eric	1	4	5	1	5	3	...
Haren	5	5	5	1	3	4	...
Sai	1	2	5	4	3	5	...
Siyan	3	1	1	3	4	5	...
Nikhil	2	3	4	2	2	2	...
Felix	1	1	1	5	2	2	...

Collaborative Filtering

Missing entries!





	Gossip Girl	The Office	The Mandalorian	Criminal Minds	The Good Place	Grey's Anatomy	...
Grace		5		1	5		...
Eric		4	5		5	3	...
Haren	5		5		3	4	...
Sai		2					...
Siyan	3	1		3		5	...
Nikhil				2	2		...
Felix	1		1		2		...

General Strategy

- **Step 1:** Construct user-item ratings
- **Step 2:** Identify similar users
- **Step 3:** Predict unknown ratings

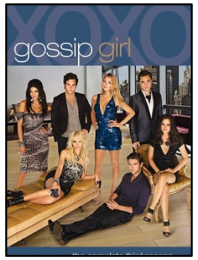
Step 1: Constructing User-Item Ratings

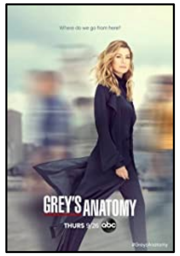
- Can use explicit ratings (e.g., Netflix)

- Can be implicitly inferred from user activity
  - User stops watching after 15 minutes
  - User repeatedly clicks on a video

- Feedback can vary in strength
  - **Weak:** User views a video
  - **Strong:** User writes a positive comment

Step 2: Identifying Similar Users





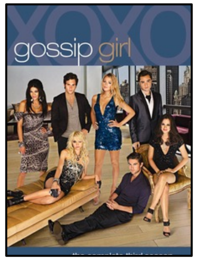
	Gossip Girl	The Office	The Mandalorian	Criminal Minds	The Good Place	Grey's Anatomy	...
Grace		5		1	5		...
Eric		4	5		5	3	...
Haren	5		5		3	4	...
Sai		2					...
Siyan	3	1		3		5	...
Nikhil				2	2		...
Felix	1		1		2		...

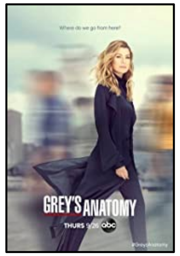
Step 2: Identifying Similar Users

	Gossip Girl	The Office	The Mandalorian	Criminal Minds	The Good Place	Grey's Anatomy	...
Grace		5		1	5		...
Eric		4	5		5	3	...
Haren	5		5		3	4	...
Sai		2					...
Siyan	3	1		3		5	...
Nikhil				2	2		...
Felix	1		1		2		...

similar

Step 2: Identifying Similar Users





	Gossip Girl	The Office	The Mandalorian	Criminal Minds	The Good Place	Grey's Anatomy	...
Grace		5		1	5		...
Eric		4	5		5	3	...
Haren	5		5		3	4	...
Sai		2					...
Siyan	3	1		3		5	...
Nikhil				2	2		...
Felix	1		1		2		...

not similar

Step 2: Identifying Similar Users

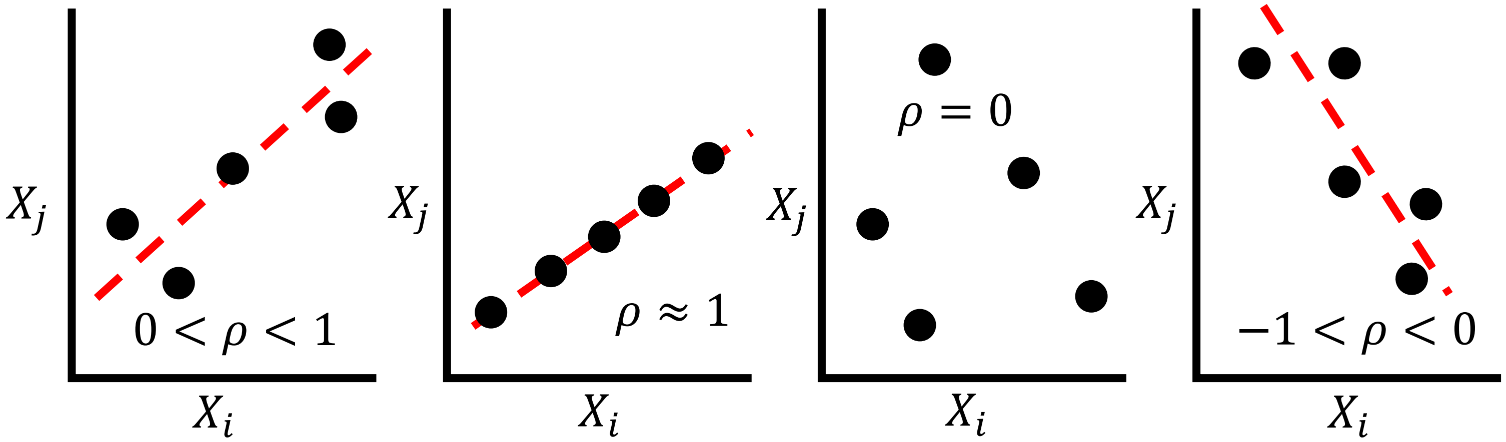
- **How to measure similarity?**
  - Distance $d(X_i, X_j)$, where X_i is vector of ratings for user i

- **Strategy 1:** Euclidean distance $d(X_i, X_j) = \|X_i - X_j\|_2$
  - Ignore entries where either X_i or X_j is N/A
  - **Shortcoming:** Some users might give higher ratings everywhere!

- Similar issues with other distance metrics such as cosine similarity

Step 2: Identifying Similar Users

- **Strategy 2:** Pearson correlation: $\rho = \dfrac{\sum_{k=1}^{m}(X_{i,k}-\bar{X}_i)(X_{j,k}-\bar{X}_j)}{\sqrt{\sum_{k=1}^{m}(X_{i,k}-\bar{X}_i)^2 \sum_{k=1}^{m}(X_{j,k}-\bar{X}_j)^2}}$
  - Here, $\bar{X}_i = \frac{1}{m}\sum_{k=1}^{m} X_{i,k}$
  - Normalization by variance deals with differences in individual rating scales



Step 3: Predict Unknown Ratings

- **Weighted averaging strategy**
  - Compute weights $w_{i,j} = g\left(d\left(X_i, X_j\right)\right)$ based on the distances
  - Normalize the weights to obtain $\bar{w}_{i,j} = \frac{w_{i,j}}{\sum_{j=1}^{n} w_{i,j}}$
  - For user i rating item k, predict

$$X_{i,k} = \bar{X}_i + \sum_{j=1}^{n} \bar{w}_{i,j} \cdot \left(X_{j,k} - \bar{X}_j\right)$$

Step 3: Predict Unknown Ratings

- **Variations**
  - Instead of weights, choose a neighborhood (e.g., threshold based on similarity, top-k based on similarity, or use k-means clustering)
  - Instead of subtracting the mean, normalize by standard deviation

Matrix Factorization

- **Model family:** Consider parameterization

$$X_{i,k} \approx U_i^\top V_k$$

  - Both $U_i \in \mathbb{R}^d$ and $V_k \in \mathbb{R}^d$ are parameters
  - U_i represents "features" for user i
  - V_k represents "features" for product k

Matrix Factorization

- **Loss function:**

$$L(U, V; X) = \sum_{i=1}^{n} \sum_{k=1}^{m} 1(X_{i,k} \neq \mathrm{N/A}) \cdot \left(X_{i,k} - U_i^\top V_k\right)^2$$

- **Optimizer:**
  - Can be minimized using gradient descent
  - **“Alternating” least squares:** Hold U fixed, then optimizing V is linear regression (and vice versa), so alternate between the two

Koren, et al. (2009) Matrix factorization techniques for recommender systems. *Computer* 42 (8), ACM.
https://datajobs.com/data-science-repo/Recommender-Systems-%5BNetflix%5D.pdf

Collaborative Filtering

- **Pros**
  - No domain knowledge needed, only user behavior
  - Captures that users may have diverse preferences
- **Cons**
  - Suffers when data is sparse
  - Does not consider item content, so cannot generalize to new items
  - Does not consider user features, so cannot generalize to new users

Content-Based Approaches

- **Step 1:** Manually construct feature vector U_i for item
- **Step 2:** Manually construct feature vector V_k for user
- **Step 3:** Train a model using supervised learning to predict the user's rating for the given item:

$$X_{i,j} \approx f_\beta(U_i, V_k)$$

Content-Based Approaches

- **Pros**
  - Incorporates external sources of knowledge on items/users to generalize
  - More explainable since recommendations are based on handcrafted features

- **Cons**
  - Requires domain knowledge and feature engineering
  - Narrow recommendations

Hybrid Approaches

- **Combine collaborative filtering with content-based approaches**
  - Ensemble different predictions
  - Concatenate collaborative filtering features with handcrafted features

- **Deep-learning based approaches**
  - Can be used with both approaches (or a combination)
  - Active area of research

Other Considerations

- **Challenges measuring utility**
  - Ratings can be misleading
  - Fake reviews/ratings are commonplace
- **Time-varying preferences**
  - User preferences change, item popularities change
  - Can upweight recent data (e.g., exponentially weighted moving average)
- **Evaluation**
  - **Offline:** Split users into train/test, and evaluate model on test users
  - **Online:** Split users into train/test, and run separate algorithms for each

What About New Users?

- Called the “cold start” problem

- **Feature-based approach**
  - Just featurize the user!

- **Collaborative filtering**
  - Need to collect ratings from the user!
  - A special kind of reinforcement learning problem called a **multi-armed bandit**

Lecture 23: Multi-Armed Bandits Part 1

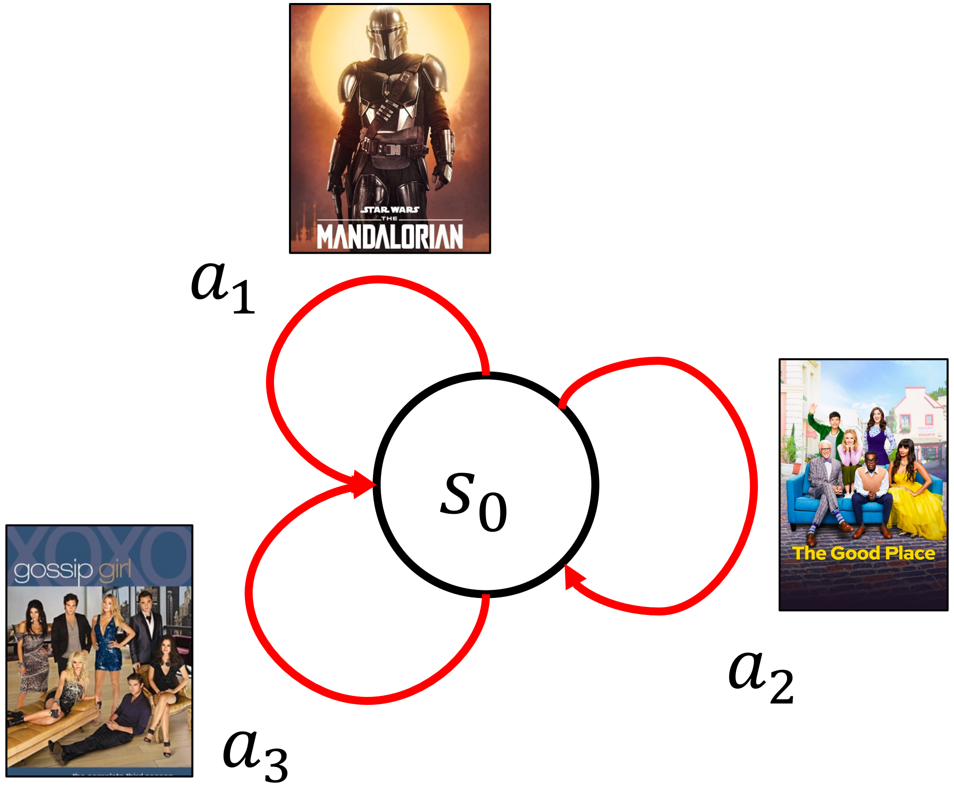
CIS 4190/5190

Fall 2022

Multi-Armed Bandits

- **State:** None! (To be precise, a single state $S = \{s_0\}$)
- **Action:** Item to recommend (often called **arms**)
- **Transitions:** Just stay in the same state
- **Rewards:** Rating assigned by the user
  - Goal is to recommend items that the user likes
  - Denote $R(a) = R(s_0, a)$, where a is the chosen action

Multi-Armed Bandits



Note: In practice, we don't want to repeatedly recommend the same item, but we will ignore this point

Multi-Armed Bandits

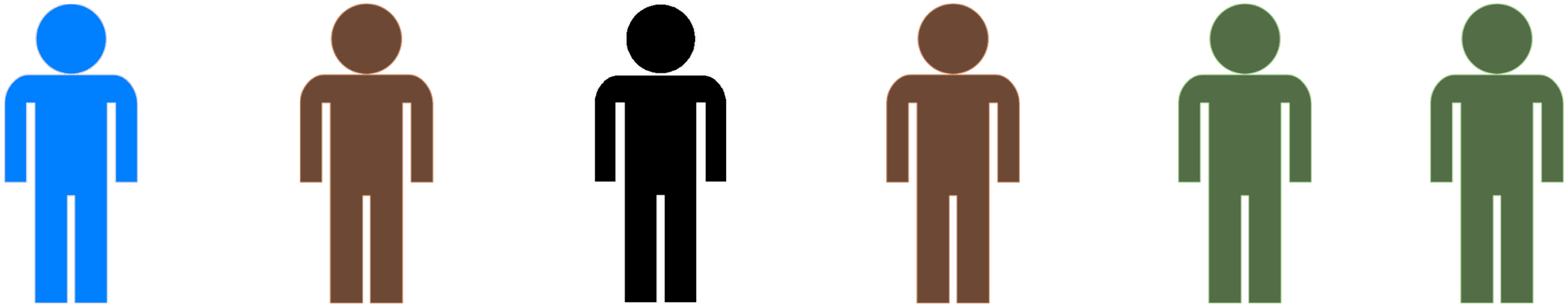
- **Many applications**
  - Cold-start for news/ad/movie recommendations
  - A/B testing
  - Flagging potentially harmful content on a social media platform
  - Prioritizing medical tests

- Learning dynamically

- Many practical RL problems are multi-armed bandits

Multi-Armed Bandits

- **Simpler setting**
  - Google wants to show the most popular ad for a search term (e.g., “lawyer”)
  - There are a fixed number of ads to choose from



Ad 3	Ad 1	Ad 2	Ad 3	Ad 2	Ad 3
Click	No Click	Click	No Click	Click	??

Multi-Armed Bandits

- For $t \in \{1,2,\dots,T\}$
  - Compute reward estimates $r_{t,a} = \frac{\sum_{i=1}^{t-1} r_i \cdot 1(a_i = a)}{\sum_{i=1}^{t-1} 1(a_i = a)}$
  - Choose action a_t based on reward estimates
  - Add (a_t, r_t) to replay buffer

- **Question:** How to choose actions?
  - **Exploration:** Try actions to better estimate their rewards
  - **Exploitation:** Use action with the best estimated reward to maximize payoff

Multi-Armed Bandit Algorithms

- **Naïve strategy:** ϵ-Greedy
  - Choose action $a_t \sim \mathrm{Uniform}(A)$ with probability ϵ
  - Choose action $a_t = \arg\max_{a \in A} r_{t,a}$ with probability $1 - \epsilon$

- Can we do better?

Multi-Armed Bandit Algorithms

- **Upper confidence bound (UCB)**
  - Choose action $a_t = \arg\max_{a \in A} \left\{ r_{t,a} + \frac{\text{const}}{\sqrt{N_t(a)}} \right\}$
  - $N_t(a) = \sum_{i=1}^{t-1} 1(a_i = a)$ is the number of times action a has been played

- **Thompson sampling**
  - Choose action $a_t = \arg\max_{a \in A} \{ r_{t,a} + \epsilon_{t,a} \}$, where $\epsilon_{t,a} \sim N\left(0, \frac{\text{const}}{\sqrt{N_t(a)}}\right)$

Two Real-World Applications

- **Application 1:** Testing travelers for COVID-19 at the Greek border

Efficient and targeted COVID-19 border testing via reinforcement learning

Hamsa Bastani, Kimon Drakopoulos, Vishal Gupta, Ioannis Vlachogiannis, Christos Hadjichristodoulou, Pagona Lagiou, Gkikas Magiorkinis, Dimitrios Paraskevis & Sotirios Tsiodras

Nature **599**, 108–113 (2021) | Cite this article

- **Application 2:** Prioritize content for review on the Meta platform

Bandits for Online Calibration: An Application to Content Moderation on Social Media Platforms

Vashist Avadhanula[◇,1], Omar Abdul Baki[◇], Hamsa Bastani[◇,†,2], Osbert Bastani[◇,†,3], Caner Gocmen[◇], Daniel Haimovich[◇], Darren Hwang[◇], Dima Karamshuk[◇], Thomas Leeper[◇], Jiayuan Ma[◇], Gregory Macnamara[◇], Jake Mullett[◇], Christopher Palow[◇], Sung Park[◇], Varun S Rajagopal[◇], Kevin Schaeffer[◇], Parikshit Shah[◇], Deeksha Sinha[◇], Nicolas Stier-Moses[◇], Peng Xu[◇]

[◇]Meta, [†]University of Pennsylvania