Formal Model

X = set of **Users**

S = set of **Items**

Utility function ***u***: ***X*** × ***S*** → ***R***

- ***R*** = set of ratings
- ***R*** is a totally ordered set
- e.g., **1-5** stars, real number in **[0,1]**

Utility Matrix

		Harry Potter			Twilight	Star Wars		
		HP1	HP2	HP3	TW	SW1	SW2	SW3
Anita	A	4			5	1		
Beyonce	B	5	5	4				
Calvin	C				2	4	5	
David	D		3					3

Key Problems

1. **Gathering "known" ratings for matrix**
   - How to collect the data in the utility matrix

2. **Extrapolate unknown ratings from known ones**
   - Mainly interested in high unknown ratings
   - We are not interested in knowing what you don't like but what you like

3. **Evaluating extrapolation methods**
   - How to measure performance of recommendation methods

(1) Gathering Ratings

Explicit

- Ask people to rate items
- Doesn’t work well in practice – people can't be bothered
- Crowdsourcing: Pay people to label items

Implicit

- Learn ratings from user actions
  - E.g., purchase (or watch video, or read article) implies high rating

(2) Extrapolating Utilities

Key problem: Utility matrix ***U*** is **sparse**

- Most people have not rated most items

- **The "Cold Start" Problem:**
  - New items have no ratings
  - New users have no history

(2) Extrapolating Utilities

Three approaches to recommender systems:

1. Content-based
2. Collaborative Filtering
3. Latent factor (Neural embedding) based

Content-based vs. Collaborative Filtering

Thomas Quella
Wikimedia Commons

Database

- Ella Fitzgerald: Jazz, Mid-20th century, vocal legend, famous duets, …
- Louis Armstrong: Jazz, Mid-20th century, vocal legend, famous duets, …

Customer W

- Plays Ella Fitzgerald
- What should we recommend next?

Photographer: Paul Stafford for www.travelmag.com
https://www.flickr.com/photos/113306963@N05/33886542421

Customer D

- Plays Ella Fitzgerald
- Plays Louis Armstrong

Content-based

Collaborative filtering

Suggest Louis Armstrong

Recommender Systems and Collaborative Filtering

Content-based Recommender Systems

Content-based Recommendations

Main idea: Recommend items to customer ***x*** similar to previous items rated highly by ***x***

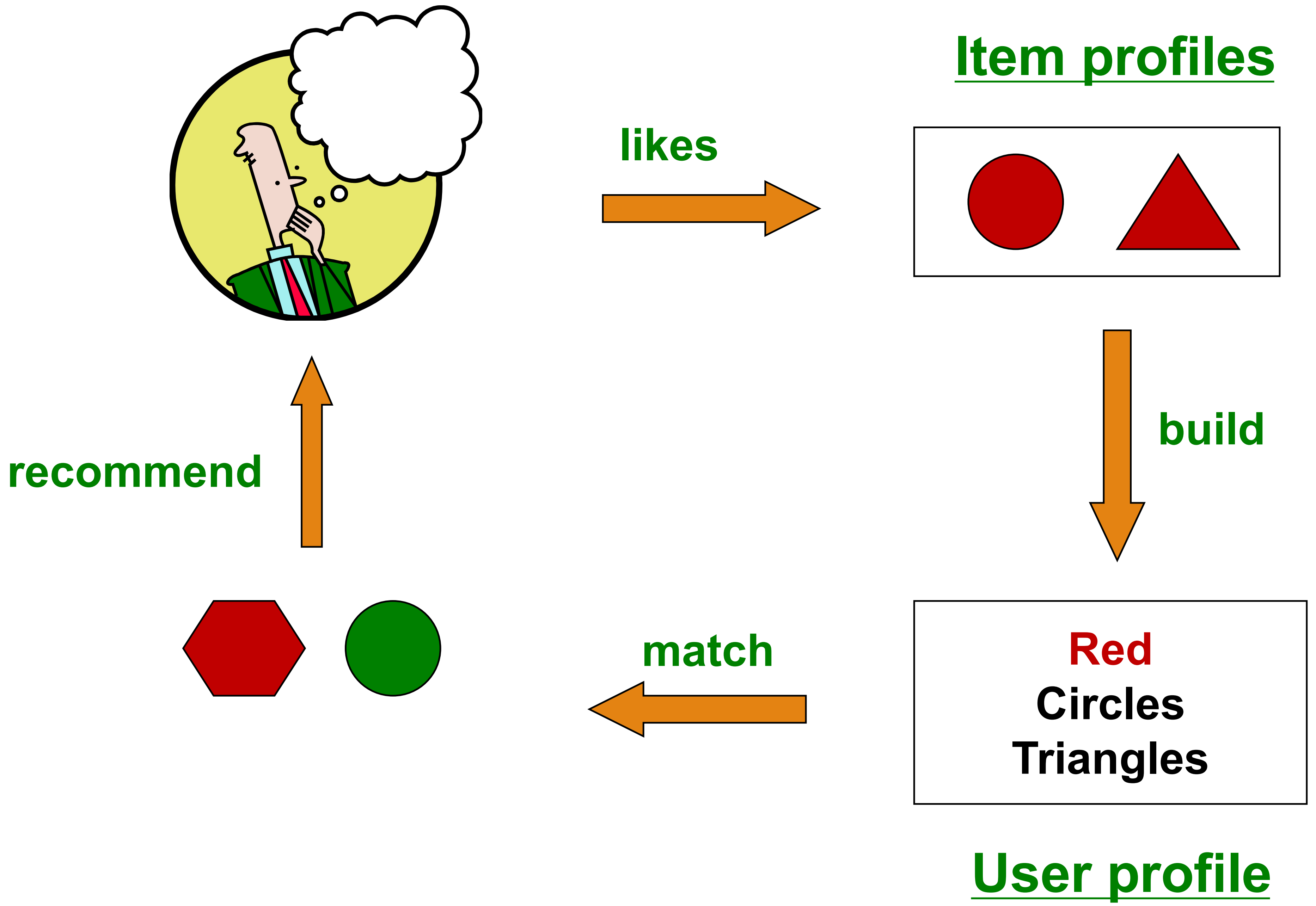
Movie recommendations

- Recommend movies with same actor(s), director, genre, …

Websites, blogs, news

- Recommend other sites with similar types or words

Plan of Action



Item Profiles

For each item, create an item profile

Profile is a set (vector) of features

- **Movies:** genre, director, actors, year…
- **Text:** Set of "important" words in document

How to pick important features?

- **TF-IDF** (Term frequency * Inverse Doc Frequency)
- For example use all words whose tf-idf > threshold, normalized for document length

Content-based Item Profiles

	Melissa McCarthy	Actor A	Actor B	…	Johnny Depp	Comic Genre	Spy Genre	Pirate Genre
Movie X	0	1	1	0	1	1	0	1
Movie Y	1	1	0	1	0	1	1	0

But what if we want to have real or ordinal features too?

Content-based Item Profiles

	Melissa McCarthy	Actor A	Actor B	…	Johnny Depp	Comic Genre	Spy Genre	Pirate Genre	Avg Rating
Movie X	0	1	1	0	1	1	0	1	3
Movie Y	1	1	0	1	0	1	1	0	4

For example "average rating"
Maybe we want a scaling factor α between binary and numeric features

Content-based Item Profiles

	Melissa McCarthy	Actor A	Actor B	…	Johnny Depp	Comic Genre	Spy Genre	Pirate Genre	Avg Rating
Movie X	0	1	1	0	1	1	0	1	3α
Movie Y	1	1	0	1	0	1	1	0	4α

Scaling factor α between binary and numeric features

$$\text{Cosine(Movie X, Movie Y)} = \frac{2+12\alpha^2}{\sqrt{5+9\alpha^2}\sqrt{5+16\alpha^2}}$$

$\alpha = 1$: 0.82 $\alpha = 2$: 0.94 $\alpha = 0.5$: 0.69

User Profiles

Want a vector with the same components/dimensions as items

- Could be 1s representing user purchases
- Or arbitrary numbers from a rating

User profile is aggregate of items:

- Weighted average of rated item profiles

Sample user profile

- Items are movies
- Utility matrix has 1 if user has seen movie
- 20% of the movies user U has seen have Melissa McCarthy
- U["Melissa McCarthy"] = 0.2

	Melissa McCarthy	Actor A	Actor B	…	
User U	0.2	.005	0	0	…

Prediction

- Users and items have the same dimensions!

	Melissa McCarthy	Actor A	Actor B	…	
Movie i	0	1	1	0	…
User x	0.2	.005	0	0	0

- So just recommend the items whose vectors are most similar to the user vector!

- Given user profile ***x*** and item profile ***i***,
- estimate $u(\boldsymbol{x},\boldsymbol{i}) = \cos(\boldsymbol{x},\boldsymbol{i}) = \frac{\boldsymbol{x}\cdot\boldsymbol{i}}{||\boldsymbol{x}||\cdot||\boldsymbol{i}||}$

Pros: Content-based Approach

+: No need for data on other users

- No user sparsity problems

+: Able to recommend to users with unique tastes

+: Able to recommend new & unpopular items

- No first-rater problem

+: Able to provide explanations

- Just list the content-features that caused an item to be recommended

Cons: Content-based Approach

– **Finding the appropriate features is hard**
- E.g., images, movies, music

– **Recommendations for new users**
- **How to build a user profile?**

– **Overspecialization**
- Never recommends items outside user's content profile
- People might have multiple interests
- **Unable to exploit quality judgments of other users**

Recommender Systems and Collaborative Filtering

Collaborative Filtering: User-User

Collaborative filtering

Instead of using content features of items to determine what to recommend

Find similar users and recommend items that they like!

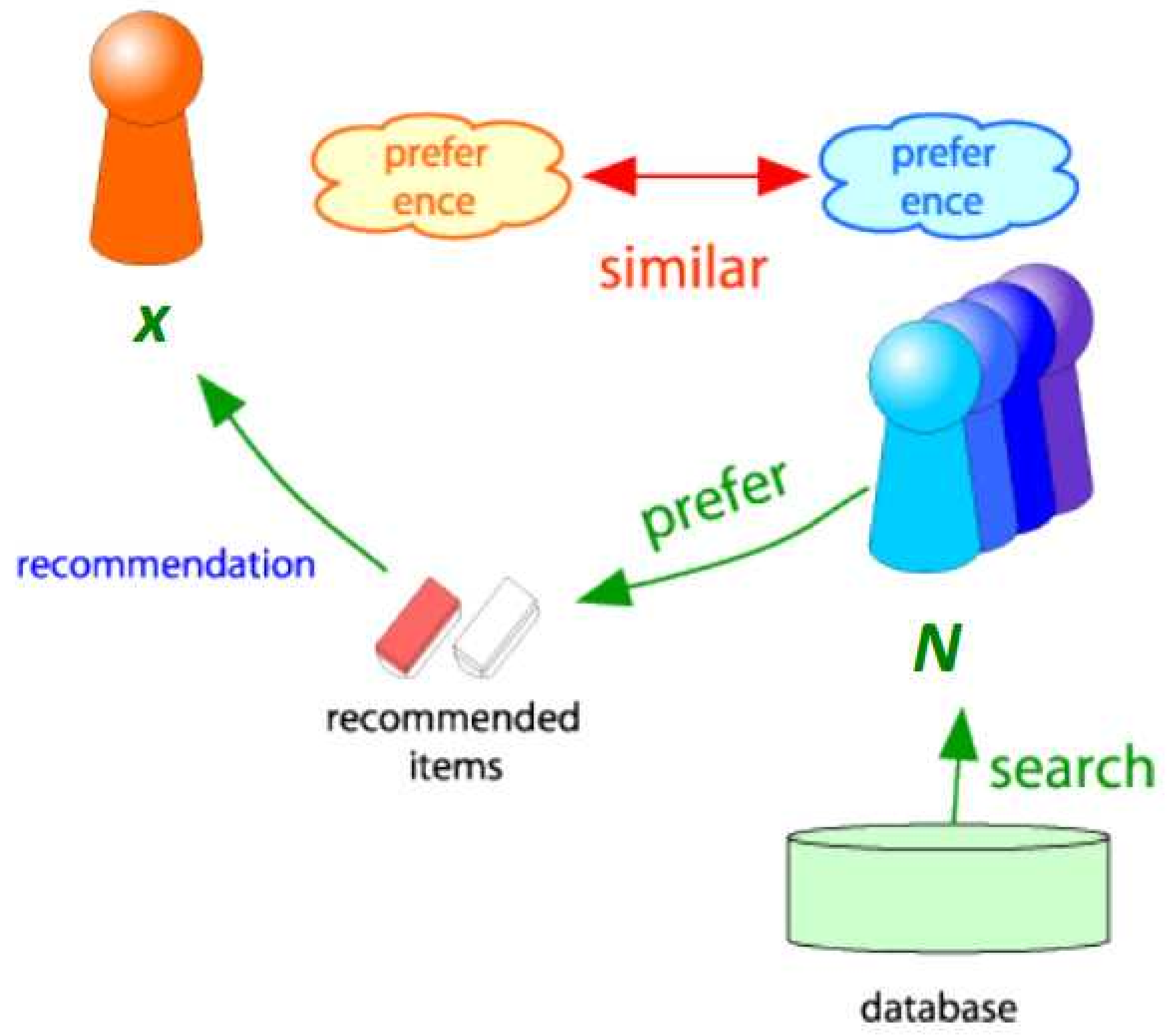
Collaborative Filtering
Version 1: "User-User" Collaborative Filtering

Consider user ***x***

and unrated item ***i***

Find set ***N*** of other users whose ratings are "**similar**" to ***x***'s ratings

Estimate ***x***'s ratings for ***i*** based on ratings for ***i*** of users in ***N***



Collaborative filtering

Find similar users and recommend items that they like:

- Represent users by their rows in the **utility matrix**
- Two users are similar if their vectors are similar!

	Harry Potter			Twilight	Star Wars		
	HP1	HP2	HP3	TW	SW1	SW2	SW3
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					3

Problems with raw utility matrix cosine

	Harry Potter			Twilight	Star Wars		
	HP1	HP2	HP3	TW	SW1	SW2	SW3
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					3

Intuitively we want: sim(*A*, *B*) > sim(*A*, *C*)

$$\text{sim(A,B)} = \frac{4 \times 5}{\sqrt{4^2+5^2+1^2}\sqrt{5^2+5^2+4^2}} = 0.380$$

$$\text{sim(A,C)} = \frac{5 \times 2 + 1 \times 4}{\sqrt{4^2+5^2+1^2}\sqrt{2^2+4^2+5^2}} = 0.322$$

Yes, 0.380 > 0.322
But only barely works…

Problem with raw cosine

	HP1	HP2	HP3	TW	SW1	SW2	SW3
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					3

- Problem with cosine:
  - C really loves SW
  - A hates SW
  - B just hasn’t seen it
- Another problem: we’d like to normalize the raters
  - D rated everything the same; not very useful

Mean-Centered Utility Matrix: subtract the means of each row

	HP1	HP2	HP3	TW	SW1	SW2	SW3
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					3

	HP1	HP2	HP3	TW	SW1	SW2	SW3
A	$2/3$			$5/3$	$-7/3$		
B	$1/3$	$1/3$	$-2/3$				
C				$-5/3$	$1/3$	$4/3$	
D		0					0

- Now a 0 means no information
- And negative ratings means viewers with opposite ratings will have vectors in opposite directions!

Modified Utility Matrix: subtract the means of each row

	HP1	HP2	HP3	TW	SW1	SW2	SW3
A	$2/3$			$5/3$	$-7/3$		
B	$1/3$	$1/3$	$-2/3$				
C				$-5/3$	$1/3$	$4/3$	
D		0					0

$$\text{Cos(A,B)} = \frac{(2/3)\times(1/3)}{\sqrt{(2/3)^2+(5/3)^2+(-7/3)^2}\sqrt{(1/3)^2+(1/3)^2+(-2/3)^2}} = 0.092$$

$$\text{Cos(A,C)} = \frac{(5/3)\times(-5/3)+(-7/3)\times(1/3)}{\sqrt{(2/3)^2+(5/3)^2+(-7/3)^2}\sqrt{(-5/3)^2+(1/3)^2+(4/3)^2}} = -0.559$$

Now A and C are (correctly) way further apart than A,B

Mean-centered overlapping-item cosine similarity

Let $\boldsymbol{r_x}$ be the vector of user $\boldsymbol{x}$'s ratings,and $\overline{r_x}$ be its mean (ignoring missing values)

Instead of basic cosine similarity measure

- sim($\boldsymbol{x}$, $\boldsymbol{y}$) = cos($\boldsymbol{r_x}$, $\boldsymbol{r_y}$) = $\frac{r_x \cdot r_y}{||r_x||\ ||r_y||}$

Mean-centered <u>overlapping-item</u> cosine similarity

(Variant of Pearson correlation)

- $\boldsymbol{S_{xy}}$ = items rated by both users $\boldsymbol{x}$ and $\boldsymbol{y}$

$$\boldsymbol{sim(x, y)} = \frac{\sum_{s \in S_{xy}} (\boldsymbol{r_{xs}} - \overline{\boldsymbol{r_x}})(\boldsymbol{r_{ys}} - \overline{\boldsymbol{r_y}})}{\sqrt{\sum_{s \in S_{xy}} (\boldsymbol{r_{xs}} - \overline{\boldsymbol{r_x}})^2} \sqrt{\sum_{s \in S_{xy}} (\boldsymbol{r_{ys}} - \overline{\boldsymbol{r_y}})^2}}$$

Rating Predictions

From similarity metric to recommendations for an unrated item *i*:

Let $\boldsymbol{r_x}$ be the vector of user ***x***'s ratings

Let ***N*** be the set of ***k*** users most similar to ***x*** who have rated item ***i***

Prediction for item *i* of user *x*:

- Rate i as the mean of what k-people-like-me rated i

$$r_{xi} = \frac{1}{k} \sum_{y \in N} r_{yi}$$

- Even better: Rate i as the mean weighted by their similarity to me …

$$r_{xi} = \frac{\sum_{y \in N} s_{xy}\, r_{yi}}{\sum_{y \in N} s_{xy}}$$

- **Many other tricks possible…**

Shorthand:
$$\boldsymbol{s_{xy} = sim(x, y)}$$

Recommender Systems and Collaborative Filtering

Collaborative Filtering: User-User

Recommender Systems and Collaborative Filtering

Collaborative Filtering: Item-Item

Collaborative Filtering Version 2: Item-Item Collaborative Filtering

So far: User-user collaborative filtering

Alternate view that often works better: Item-item

- For item ***i***, find other similar items
- Estimate rating for item ***i*** based on ratings for those similar items
- Can use same similarity metrics and prediction functions as in user-user model
- "Rate i as the mean of my ratings for other items, weighted by their similarity to i"

$$\boxed{r_{xi} = \frac{\sum_{j \in N(i;x)} s_{ij} \; r_{xj}}{\sum_{j \in N(i;x)} s_{ij}}}$$

N(i;x)…set of items rated by ***x*** *and* similar to ***i***
s_{ij}… similarity of items ***i*** and ***j***
r_{xj}…rating of user ***x*** on item ***j***

Item-Item CF (|N|=2)

users

movies

	1	2	3	4	5	6	7	8	9	10	11	12
1	1		3			5			5		4	
2			5	4			4			2	1	3
3	2	4		1	2		3		4	3	5	
4		2	4		5			4			2	
5			4	3	4	2					2	5
6	1		3		3			2			4	

☐ - unknown rating　🟨 - rating between 1 to 5

Item-Item CF (|N|=2)

users

movies

	1	2	3	4	**5**	6	7	8	9	10	11	12
1	1		3		?	5			5		4	
2			5	4			4			2	1	3
3	2	4		1	2		3		4	3	5	
4		2	4		5			4			2	
5			4	3	4	2					2	5
6	1		3		3			2			4	

🟥 - estimate rating of movie **1** by user **5**

Item-Item CF (|N|=2)
Approximate rating with weighted mean

users

movies	1	2	3	4	5	6	7	8	9	10	11	12	sim(1,m)
1	1		3		**2.54**	5			5		4		**1.000**
2			5	4			4			2	1	3	**..**
3	2	4		1	2		3		4	3	5		**.658**
4		2	4		5			4			2		**..**
5			4	3	4	2					2	5	**..**
6	1		3		3			2			4		**.768**

Predict by taking weighted average:

$$r_{ix} = \frac{\sum_{j \in N(i;x)} s_{ij} \, r_{jx}}{\sum s_{ij}}$$

$r_{1,5}$ = **(0.658*2 + 0.768*3) / (0.658+0.768) = 2.54**

Item-Item vs. User-User

- **In practice, item-item often works better than user-user**
- **Why?** Items are simpler, users have multiple tastes
  - (People are more complex than objects)

Pros/Cons of Collaborative Filtering

+ Works for any kind of item

- No feature selection needed

- Cold Start:

- Need enough users in the system to find a match

- Sparsity:

- The user/ratings matrix is sparse
- Hard to find users that have rated the same items

- First rater:

- Cannot recommend an item that has not been previously rated

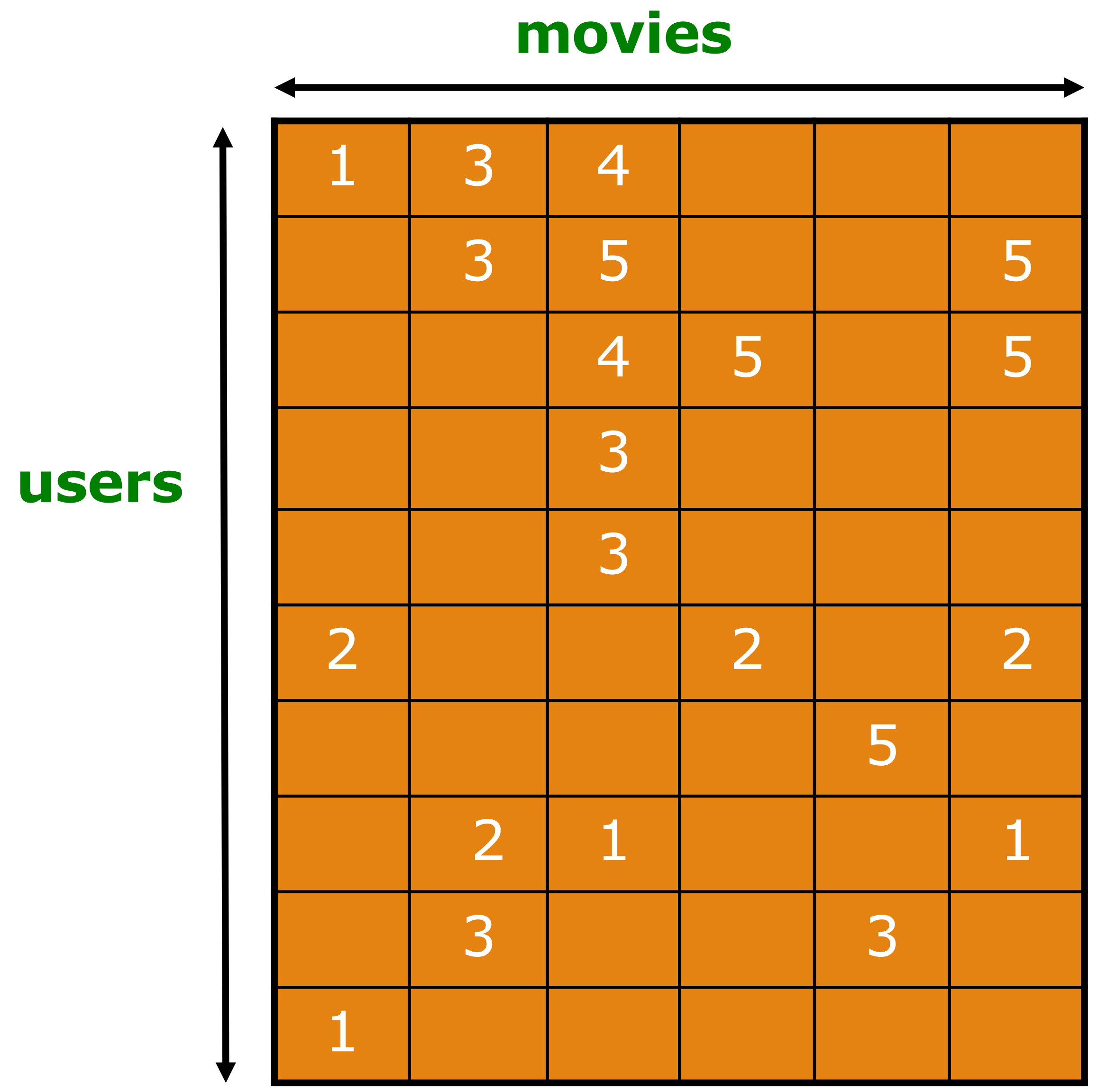
- Popularity bias:

- Cannot recommend items to someone with unique taste
- Tends to recommend popular items

- Ethical and social issues:

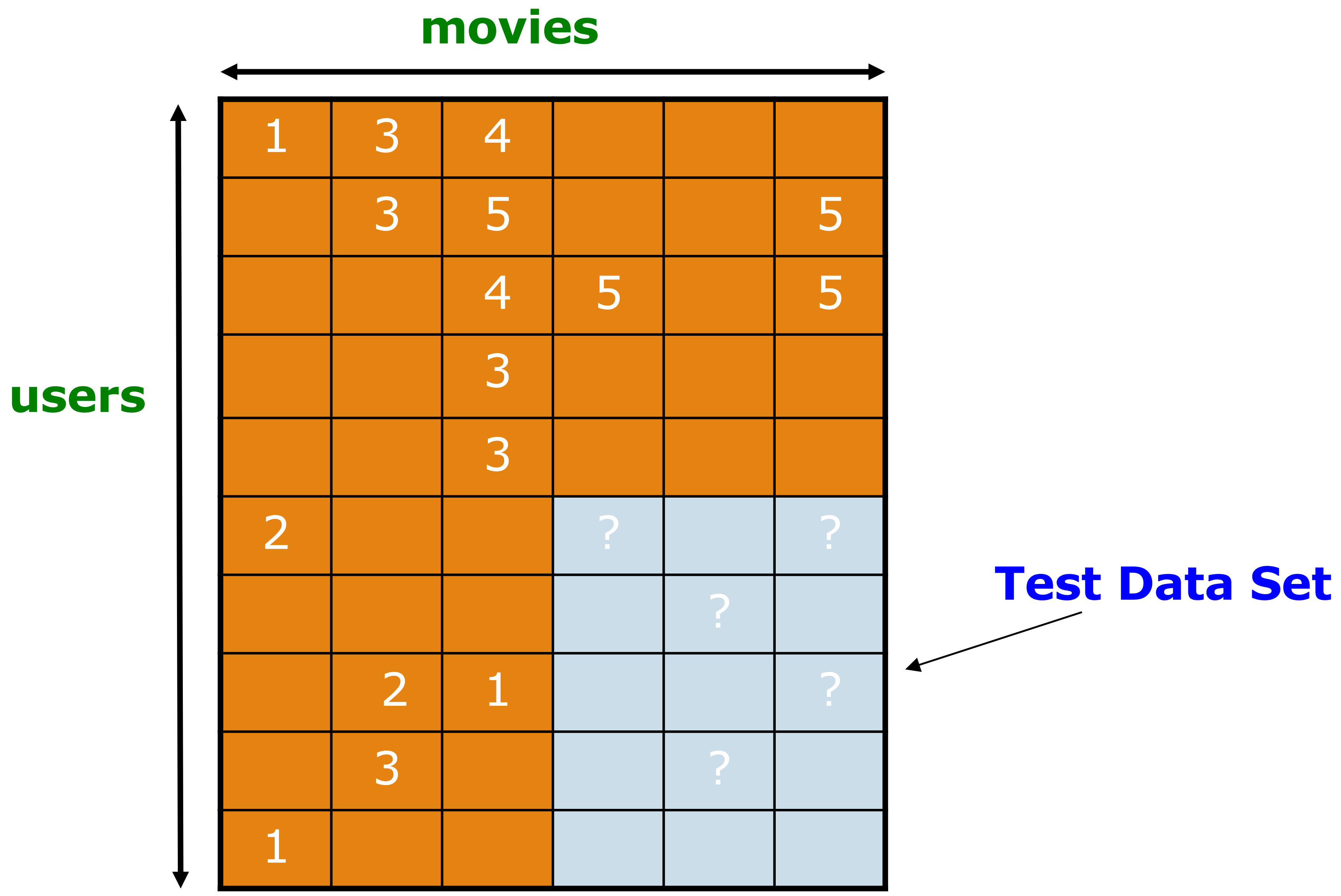
- Can lead to filter bubbles and radicalization spirals

Evaluation



1	3	4			
	3	5			5
		4	5		5
		3			
		3			
2			2		2
				5	
	2	1			1
	3			3	
1					

Evaluation



1	3	4			
	3	5			5
		4	5		5
		3			
		3			
2			?		?
				?	
	2	1			?
	3			?	
1					

Evaluating Predictions

Compare predictions with known ratings

- **Root-mean-square error** (RMSE)
  - $\sqrt{\frac{\sum_{xi}(r_{xi}-r_{xi}^{*})^{2}}{N}}$
  - where $\boldsymbol{r_{xi}}$ is predicted, $\boldsymbol{r_{xi}^{*}}$ is the true rating of $\boldsymbol{x}$ on $\boldsymbol{i}$
- **Rank Correlation**:
  - Spearman's *correlation* between system's and user's complete rankings