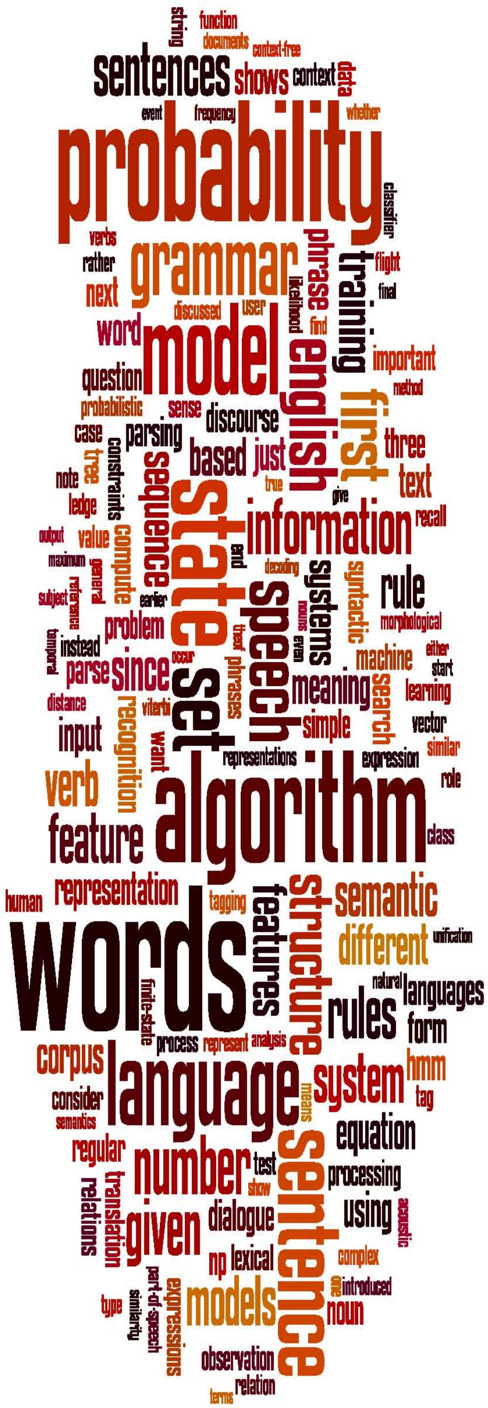


CS 124/LINGUIST 180 From Languages to Information

Dan Jurafsky

Stanford University

Recommender Systems & Collaborative Filtering

Slides adapted from Jure Leskovec

Recommender Systems

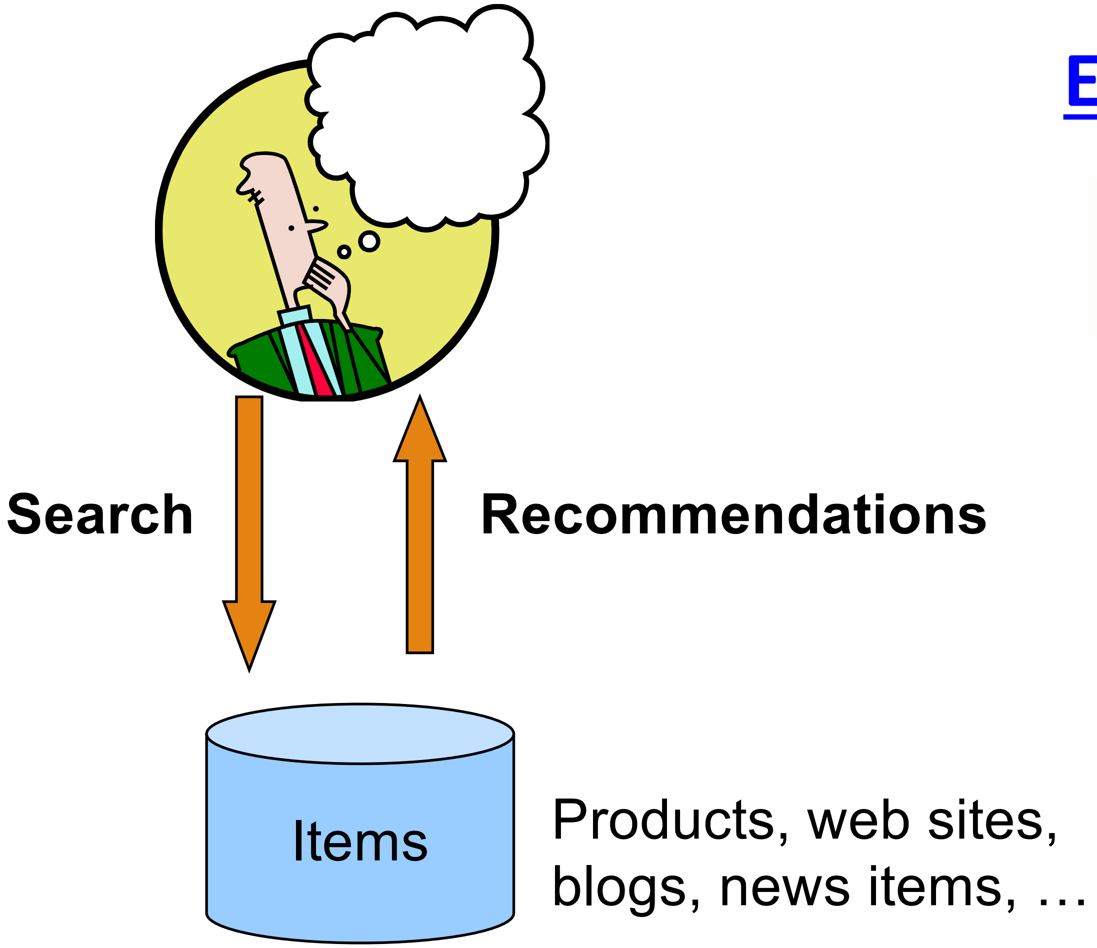
Customer X

- Buys CD of Mozart
- Buys CD of Haydn

Customer Y

- Does search on Mozart
- Recommender system suggests Haydn from data collected about customer **X**

Recommendations



Search

Recommendations

Items

Products, web sites, blogs, news items, …

Examples:

PANDORA

amazon.com

YouTube

NETFLIX

Google News

SLIDES ADAPTED FROM JURE LESKOVEC

From Scarcity to Abundance

Shelf space is a scarce commodity for traditional retailers

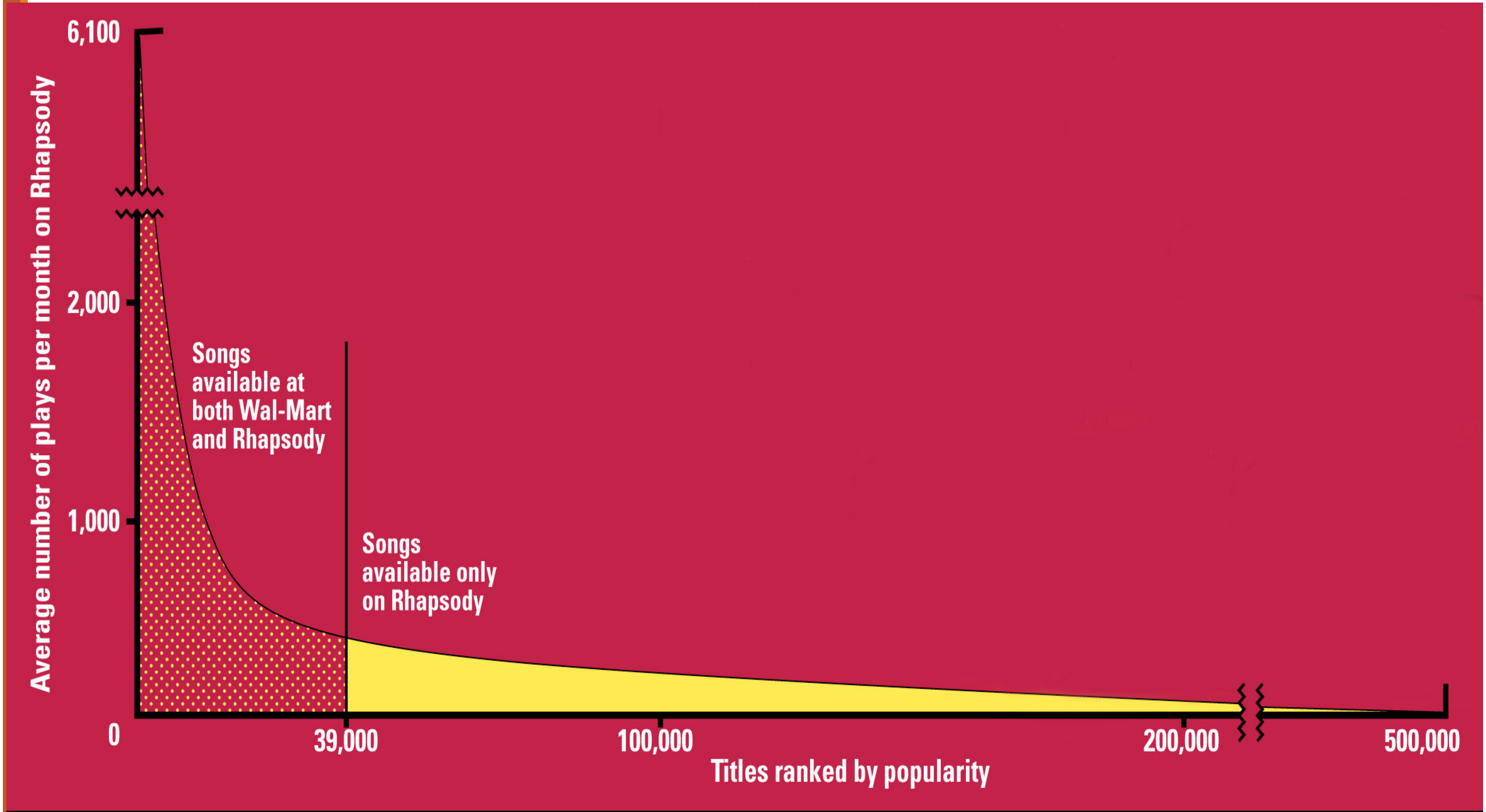
- Also: TV networks, movie theaters,…

Web enables near-zero-cost dissemination of information about products

- From scarcity to abundance

SLIDES ADAPTED FROM JURE LESKOVEC

The Long Tail



Sources: Erik Brynjolfsson and Jeffrey Hu, MIT, and Michael Smith, Carnegie Mellon; Barnes & Noble; Netflix; RealNetworks

Source: Chris Anderson (2004)

More choice requires:

Recommendation engines!

Knowing how these work

Isn't relevant only for building practical news or product recommenders….

We all need to understand how they work!

QAnon Supporters And Anti-Vaxxers Are Spreading A Hoax That Bill Gates Created The Coronavirus

It has no basis in reality, but that hasn't slowed its spread across Facebook and Twitter.

The Guardian

US edition

Las Vegas survivors furious as YouTube promotes clips calling shooting a hoax

https://www.theguardian.com/technology/2018/feb/02/how-youtubes-algorithm-distorts-trut

'Fiction is outperforming reality': how YouTube's algorithm distorts truth

THE WALL STREET JOURNAL.

TECH

How YouTube Drives People to the Internet's Darkest Corners

Google's video site often recommends divisive or misleading material, despite recent changes designed to fix the problem

Types of Recommendations

Editorial and hand curated

- List of favorites
- Lists of “essential” items

Simple aggregates

- Top 10, Most Popular, Recent Uploads

Tailored to individual users

Today class

- Amazon, Netflix, …

SLIDES ADAPTED FROM JURE LESKOVEC

Formal Model

$\boldsymbol{X}$ = set of **Customers**

$\boldsymbol{S}$ = set of **Items**

Utility function $\boldsymbol{u}: \boldsymbol{X} \times \boldsymbol{S} \rightarrow \boldsymbol{R}$

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

Utility Matrix

	Avatar	LOTR	Matrix	Pirates
Alice	1		0.2	
Bob		0.5		0.3
Carol	0.2		1	
David				0.4

SLIDES ADAPTED FROM JURE LESKOVEC

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 success/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 finish watching YouTube video) 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 based

} **This lecture!**

CS246!

Content-based Recommender Systems

Content-based Recommendations

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

Example:

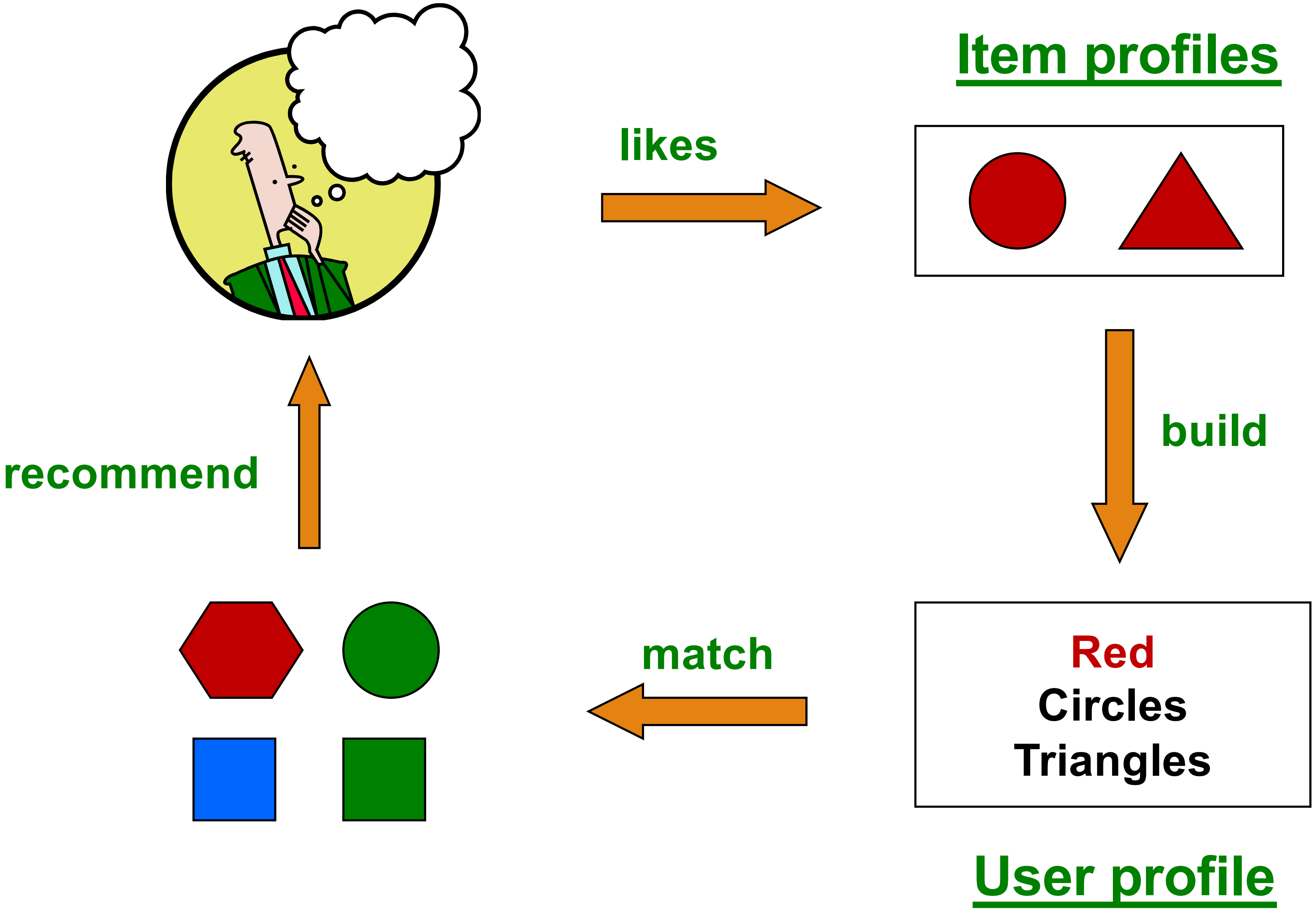
Movie recommendations

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

Websites, blogs, news

- Recommend other sites with "similar" content

Plan of Action



Item profiles

likes

build

Red
Circles
Triangles

User profile

match

recommend

SLIDES ADAPTED FROM JURE LESKOVEC

Item Profiles

For each item, create an **item profile**

Profile is a set (vector) of features

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

How to pick important features?

- **TF-IDF** (Term frequency * Inverse Doc Frequency)
  - **Term … Feature**
  - **Document … Item**

SLIDES ADAPTED FROM JURE LESKOVEC

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

- If everything is 1 or 0 (indicator features)
- But what if we want to have real or ordinal features too?

SLIDES ADAPTED FROM JURE LESKOVEC

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

- Maybe we want a scaling factor α between binary and numeric features

SLIDES ADAPTED FROM JURE LESKOVEC

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

- Maybe we want a scaling factor α between binary and numeric features
- Or maybe α=1

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

SLIDES ADAPTED FROM JURE LESKOVEC

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:

- Average(weighted?)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 cold-start or 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

SLIDES ADAPTED FROM JURE LESKOVEC

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**

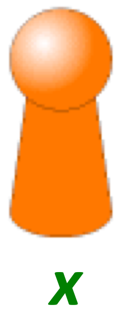
Collaborative Filtering

Harnessing quality judgments of other users

Collaborative Filtering

Version 1: "User-User" Collaborative Filtering

Consider user ***x***



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

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

SLIDES ADAPTED FROM JURE LESKOVEC

Finding Similar Users

Let r_x be the vector of user ***x***'s ratings

r_x = [*, _, _, *, ***]
r_y = [*, _, **, **, _]

r_x = {1, 0, 0, 1, 3}
r_y = {1, 0, 2, 2, 0}

Cosine similarity measure

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

Problem: Treats missing ratings as "negative"

- What do I mean?

SLIDES ADAPTED FROM JURE LESKOVEC

Utility Matrix

	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*)

Cosine similarity: Yes, 0.386 **>** 0.322

But only barely works…

Considers missing ratings as "negative"

HP = Harry Potter, TW = Twilight, SW = Star Wars

Utility Matrix

	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: 0 acts like a negative review
  - C really loves SW
  - A hates SW
  - B just hasn’t seen it
- Another problem: we’d like to normalize for 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

Terminological Note: subtracting the mean is **mean-centering**, not **normalizing**

(normalizing is dividing by a norm to turn something into a probability), but the textbook (and common usage) sometimes overloads the term “normalize”

Finding similar users with mean-centering

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

r_x = {1, 0, 0, 1, 3}

r_y = {1, 0, 2, 2, 0}

r_x = [*, _, _, *, ***]

r_y = [*, _, **, **, _]

Mean-centering:

- For each user x, let r_x be mean of r_x (ignoring missing values)
- $\overline{r}_x$ = (1 + 1 + 3)/3 = 5/3 ry = (1 + 2 + 2)/3 = 5/3
- Subtract this average from each of their ratings
  - (but do nothing to the "missing values"; they stay "null").
  - mean centered r_x = {-2/3, 0, 0, -2/3, 4/3}

Now: Keep only items they both rate (unlike 2 slides ago)

r_x = {-2/3, [redacted], [redacted], -2/3, [redacted]}

r_y = {-2/3, [redacted], [redacted], 1/3, [redacted]}

r_x = {-2/3, -2/3}

r_y = {-2/3, 1/3}

Take cosine:

- Now compute cosine between user vectors
  - cos([-2/3, -2/3], [-2/3, 1/3])

Finding similar users with mean centering: more formally

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

Cosine similarity measure

- sim(***x***, ***y***) = cos(***r***$\boldsymbol{r_x}$, $\boldsymbol{r_y}$) = $\frac{r_x \cdot r_y}{||r_x||\ ||r_y||}$
- **Problem:** Treats missing ratings as "negative"

Mean-centered overlapping-item cosine similarity

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

Let ***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 …

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

Shorthand:

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

- **Many other tricks possible…**

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 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"

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

s_{ij}… similarity of items i and j

r_{xj}…rating of user x on item i

$N(i;x)$…set of items rated by x similar to i

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

SLIDES ADAPTED FROM JURE LESKOVEC

Item-Item CF (|N|=2)

users

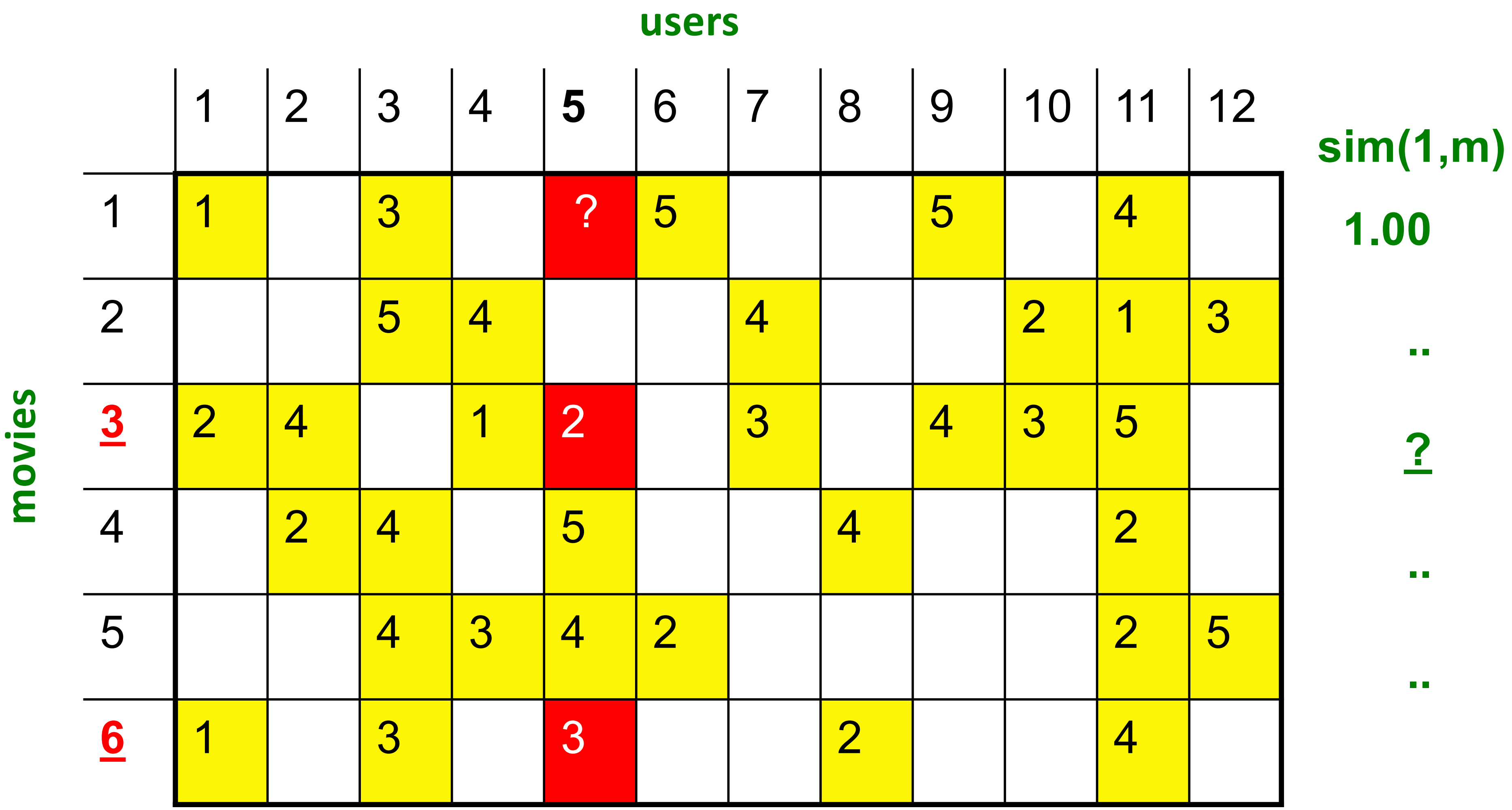
movies

	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**

SLIDES ADAPTED FROM JURE LESKOVEC

Item-Item CF (|N|=2)



users

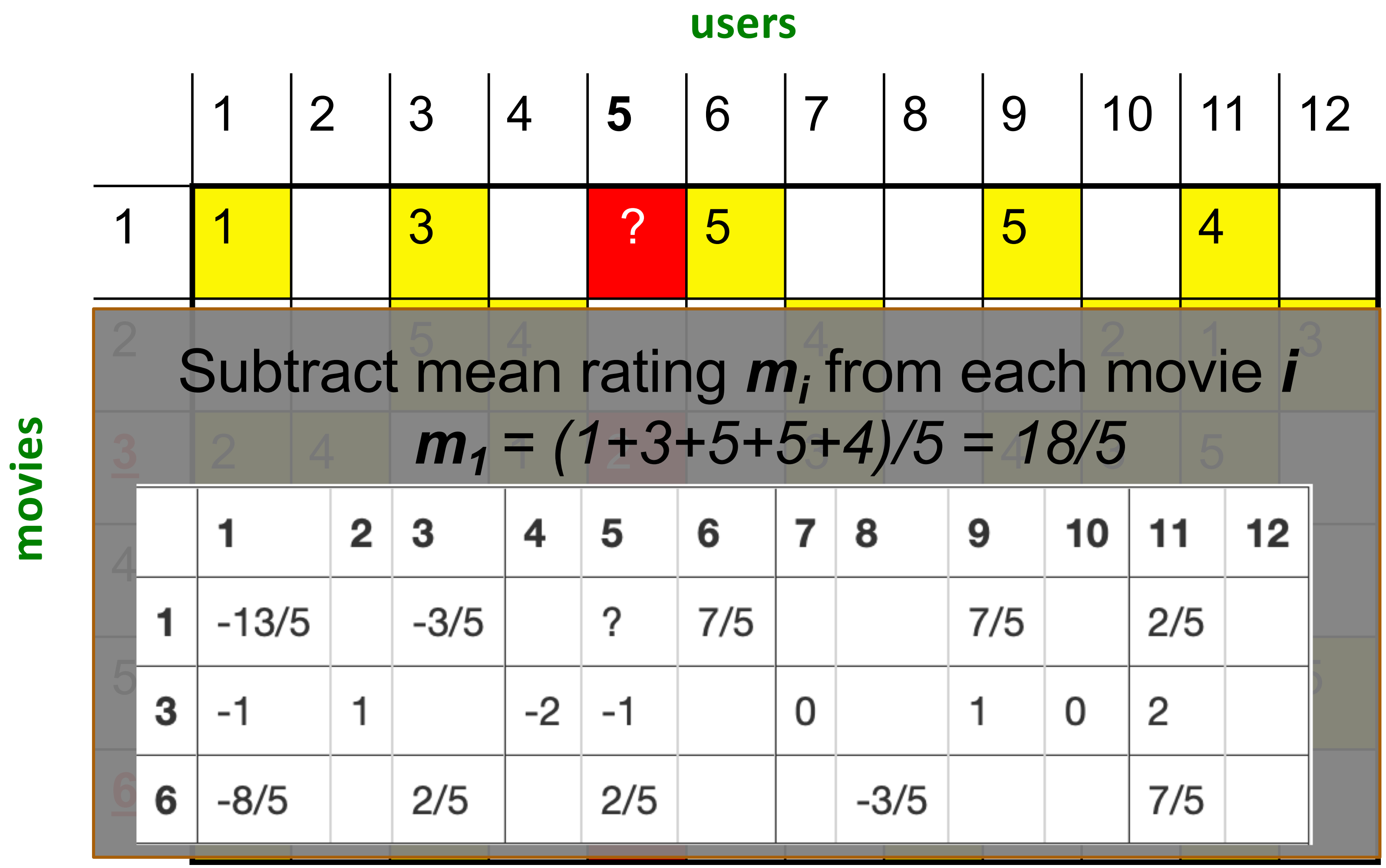
movies

	1	2	3	4	**5**	6	7	8	9	10	11	12	**sim(1,m)**
1	1		3		?	5			5		4		**1.00**
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		

Neighbor selection:
Identify movies similar to
movie **1**, **rated by user 5**

Here we use mean centered item-overlap cosine as similarity:
1) Subtract mean rating m_i from each movie i
2) Compute (item-overlapping) cosine similarities between rows

Item-Item CF (|N|=2)



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

Subtract mean rating m_i from each movie i

$m_1 = (1+3+5+5+4)/5 = 18/5$

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
1	-13/5		-3/5		?	7/5			7/5		2/5	
3	-1	1		-2	-1		0		1	0	2	
6	-8/5		2/5		2/5			-3/5			7/5	

Showing computation only for #3 and #6

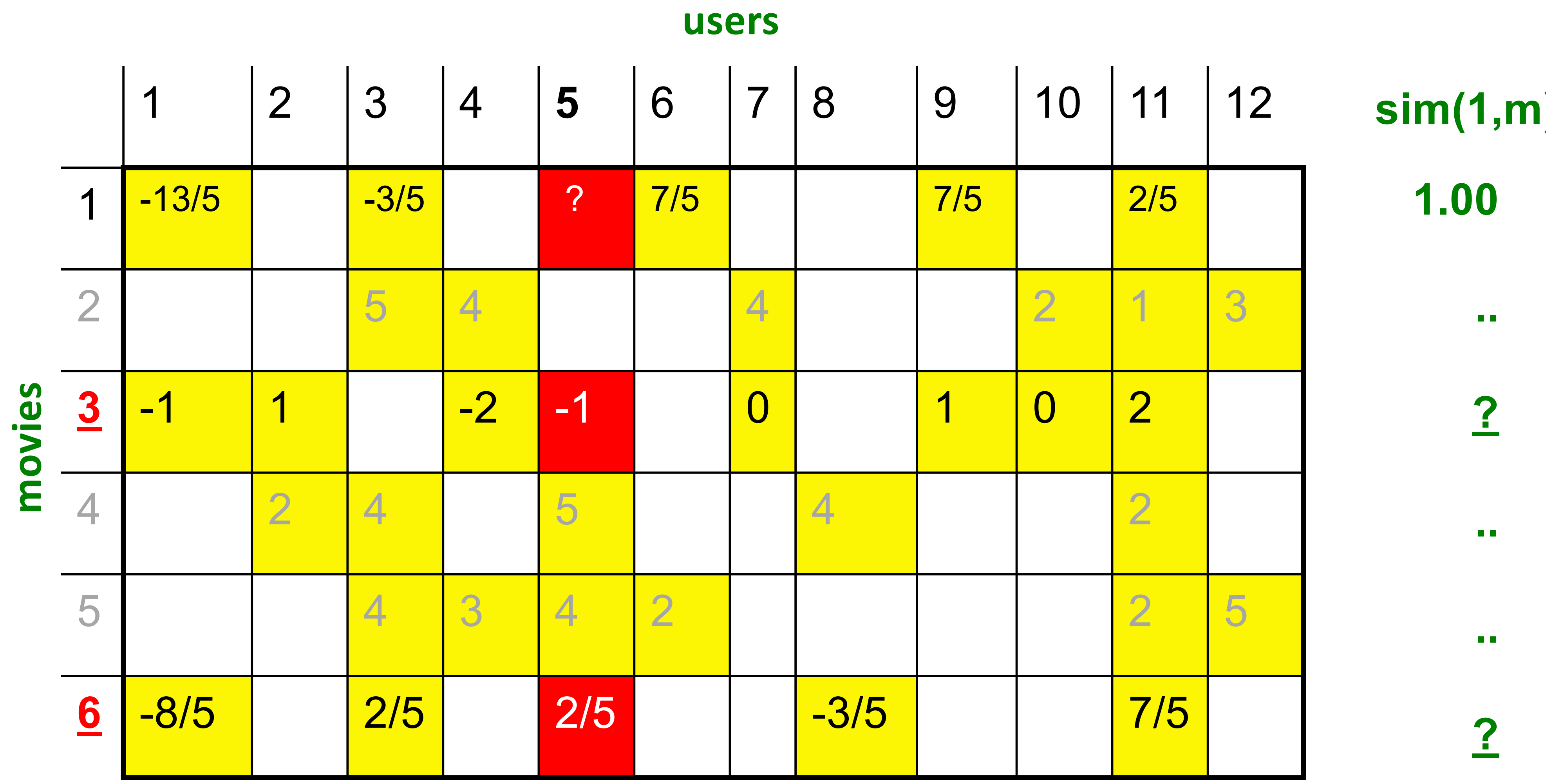
Neighbor selection:
Identify movies similar to movie **1**, **rated by user 5**

Here we use mean centered item-overlap cosine as similarity:

1) Subtract mean rating m_i from each movie i

2) Compute (item-overlapping) cosine similarities between rows

Item-Item CF (|N|=2)



users

movies

	1	2	3	4	**5**	6	7	8	9	10	11	12	**sim(1,m**
1	-13/5		-3/5		?	7/5			7/5		2/5		**1.00**
2			5	4			4			2	1	3	**..**
3	-1	1		-2	-1		0		1	0	2		**?**
4		2	4		5			4			2		**..**
5			4	3	4	2					2	5	**..**
6	-8/5		2/5		2/5			-3/5			7/5		**?**

Neighbor selection:
Identify movies similar to movie **1**, **rated by user 5**

Here we use mean centered item-overlap cosine as similarity:
1) Subtract mean rating m_i from each movie i
2) Compute (item-overlapping) cosine similarities between rows

Compute Cosine Similarity:

For rows 1 and 3, they both have values for users 1, 9 and 11.

$$\text{sim}(1, 3) = \frac{(-13/5)(-1)+(7/5)(1)+(2/5)(2)}{\sqrt{(-13/5)^2+(7/5)^2+(2/5)^2}\cdot\sqrt{(-1)^2+(1)^2+(2)^2}} \approx 0.658$$

For rows 1 and 6, they both have values for users 1, 3 and 11.

$$\text{sim}(1, 6) = \frac{(-13/5)(-8/5)+(-3/5)(2/5)+(2/5)(7/5)}{\sqrt{(-13/5)^2+(-3/5)^2+(2/5)^2}\cdot\sqrt{(-8/5)^2+(2/5)^2+(7/5)^2}} \approx 0.768$$

Item-Item CF (|N|=2)

users

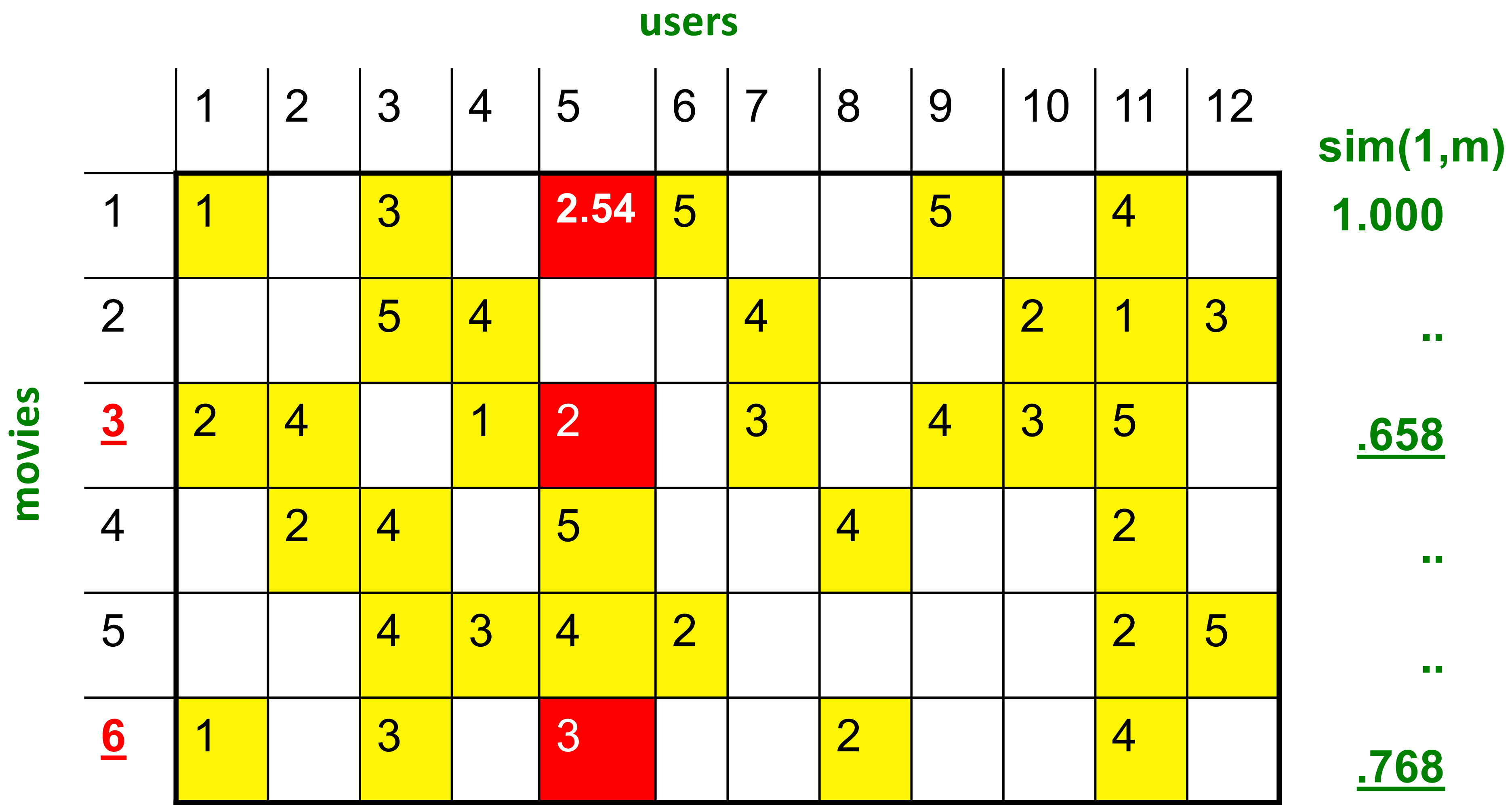
movies

	1	2	3	4	5	6	7	8	9	10	11	12	sim(1,m)
1	1		3		?	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

Compute similarity weights:

$s_{1,3}$=.658, $s_{1,6}$=.768 (we compute $s_{1,2}$, $s_{1,4}$, $s_{1,5}$ too; let's assume those are smaller)

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

SLIDES ADAPTED FROM JURE LESKOVEC

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)

Simplified item-item for our homework

First, assume you've converted all the values to

+1 (like),

0 (no rating)

−1 (dislike)

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	

Simplified item-item for our homework

First, assume you've converted all the values to

+1 (like),

0 (no rating)

−1 (dislike)

users

movies

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

Simplified item-item for our tiny PA6 dataset

Assume you've **binarized**, i.e. converted all the values to

- +1 (like), 0 (no rating) –1 (dislike)

For this binary case, some tricks that the TAs recommend:

- **Don't mean-center users, just keep the raw +1,0,-1**
- **Don't normalize (i.e. don't divide the product by the sum)**
- **i.e., instead of this:**

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

- **Just do this:**

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

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

- **Don't use mean-centered item-overlap cosine to compute s_{ij}**
  - **Just use cosine**

Simplified item-item for our tiny PA6 dataset

1. binarize, i.e. convert all values to
   - +1 (like), 0 (no rating) −1 (dislike)

2. The user x gives you (say) ratings for 2 movies m1 and m2

r_{xj}…rating of user ***x*** on item ***i***

3. For each movie *i* in the dataset
   - $r_{xi} = \sum_{j \in (m1,m2)} s_{ij}\, r_{xj}$
   - Where s_{ij}… cosine between vectors for movies ***i*** and ***j***

4. Recommend the movie *i* with max r_{xi}

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
- New items, Esoteric items

- Popularity bias:

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

SLIDES ADAPTED FROM JURE LESKOVEC

Hybrid Methods

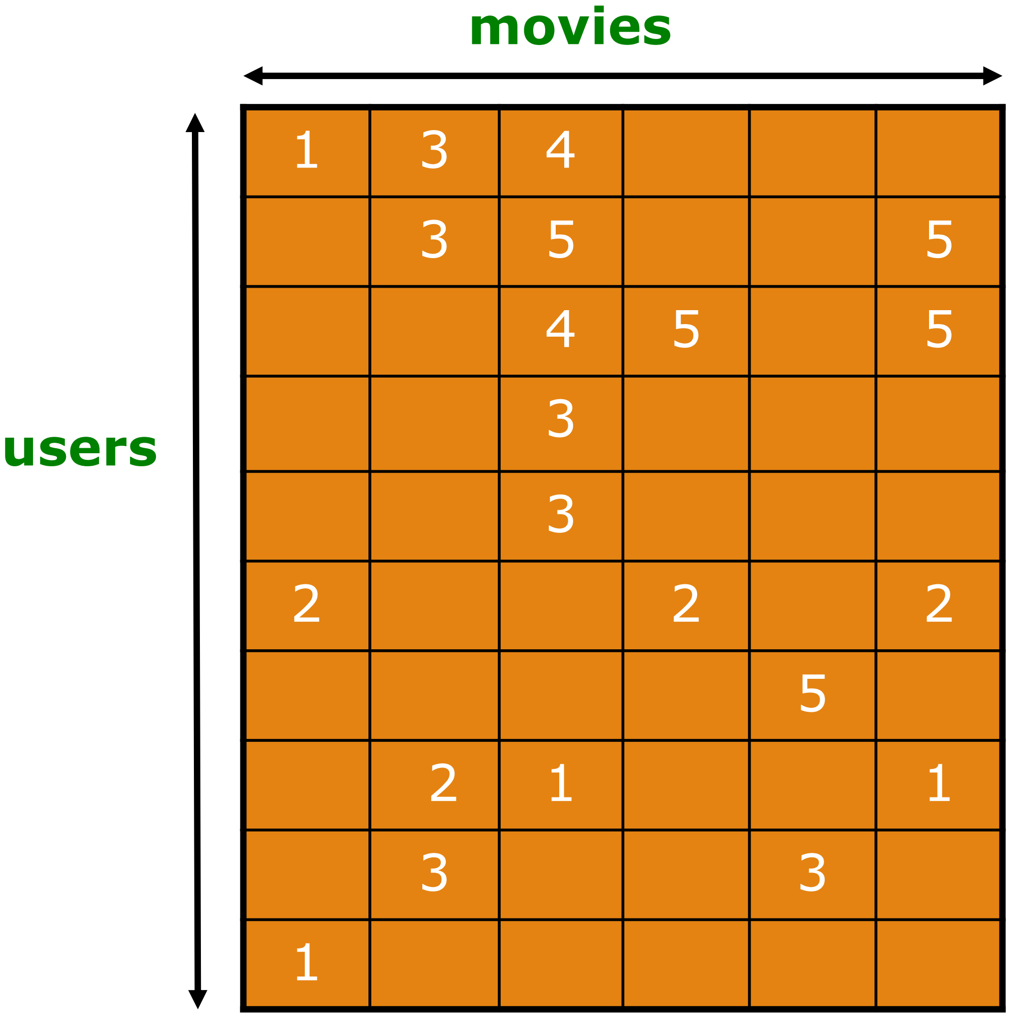
Implement two or more different recommenders and combine predictions

- Perhaps using a linear model

Add content-based methods to collaborative filtering

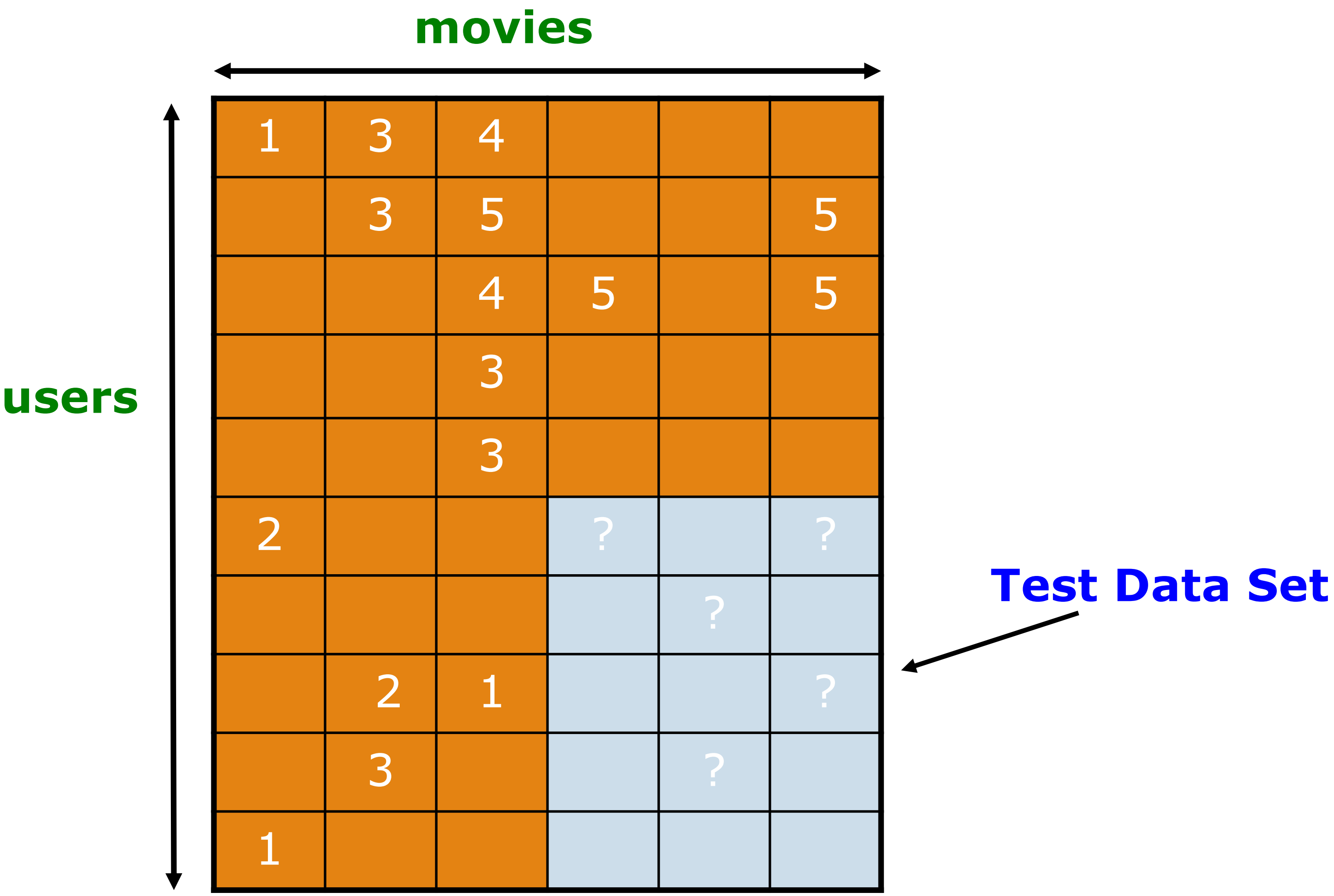
- Item profiles for new item problem
- Demographics to deal with new user problem

Evaluation



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

Evaluation



movies

users

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

Test Data Set

Evaluating Predictions

Compare predictions with known ratings

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

Problems with Error Measures

Narrow focus on accuracy sometimes misses the point

- Prediction Diversity
- Prediction Context

In practice, we care only to predict high ratings:

- RMSE might penalize a method that does well for high ratings and badly for others

There's No Data like More Data

Leverage all the data

- Simple methods on large data do best

Add more data

- e.g., add IMDB data on genres

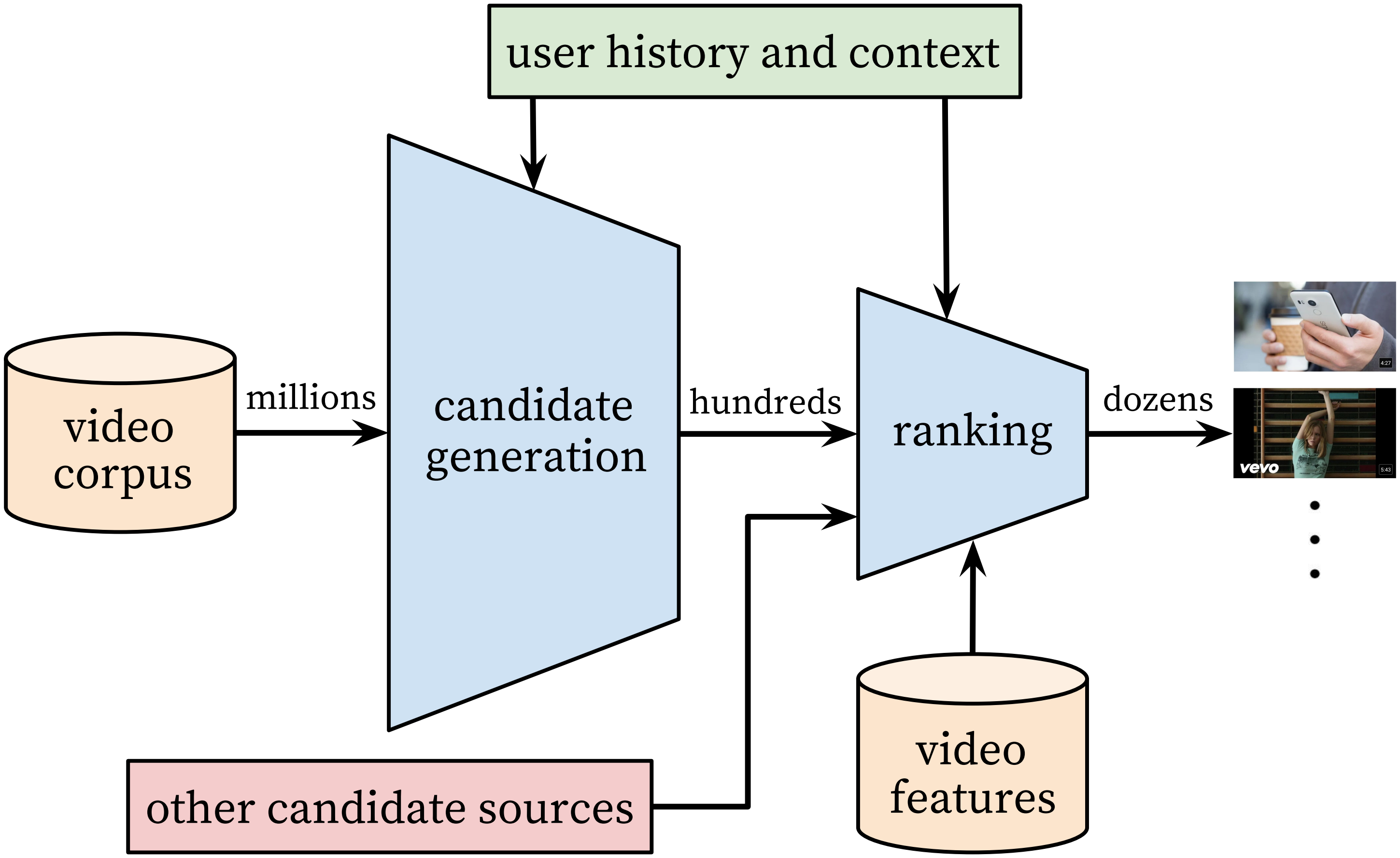
More data beats better algorithms

Summary on Recommendation Systems

- The Long Tail
- Content-based Systems
- Collaborative Filtering

State of the Art Example: YouTube

Covington, Adams, Sargin 2016



YouTube's Recommendation Algorithm

Covington, Adams, Sargin 2016. Deep Neural Networks for YouTube Recommendations

1. Represent each video as an embedding
2. Train neural net classifier (softmax over videos) to predict next video
   - Only include videos with many minutes watched
3. Input features:
   - User's watch history
     - Sequence of "video embeddings" for each video they watched
   - User's recent queries (word embeddings)
   - User's location
   - Date, popularity, virality of video

What could go wrong?: ethical and societal implications

Howard, Ganesh, Lioustiou. 2019. The IRA, Social Media, and Political Polarization in the United States, 2012-2018

Propaganda campaigns

- Russia's Internet Research Agency (IRA)
  - attack on the United States 2013-2018
  - computational propaganda on YouTube, Facebook, Instagram, to misinform/polarize US voters.
  - Goal: induce African American, Mexican American voters to boycott elections

Ethical and societal implications: Filter bubbles

THE WALL STREET JOURNAL.

TECH

How YouTube Drives People to the Internet's Darkest Corners

Google's video site often recommends divisive or misleading material, despite recent changes designed to fix the proble

"I realized really fast that YouTube's recommendation was putting people into filter bubbles," Chaslot said. "There was no way out. **If a person was into Flat Earth conspiracies, it was bad for watch-time to recommend anti-Flat Earth videos, so it won't even recommend them.**"

“The question before us is the ethics of leading people down hateful rabbit holes full of misinformation and lies at scale just because it works to increase the time people spend on the site – and it does work”

- – Zeynep Tufekci,

Open research questions

What would algorithms look like that could recommend but also include these social costs?