

# Recommender System

---

- What is it?
- How to build it?
- Practical challenges & state-of-the-art
- R package: **recommenderlab**

# Recommender System

- **What is it?**
- How to build it?
- Practical challenges & state-of-the-art
- R package: **recommenderlab**

A **recommender system** or a **recommendation system** (sometimes replacing "system" with a synonym such as platform or engine) is a subclass of information filtering **system** that seeks to predict the "rating" or "preference" a user would give to an item.

**Recommender system - Wikipedia**

[https://en.wikipedia.org/wiki/Recommender\\_system](https://en.wikipedia.org/wiki/Recommender_system)

- RS is everywhere: Amazon, Wayfair, Netflix, Google News, Pinterest, Spotify, Facebook, Linkedin, OkCupid .....
- A system that can **automatically** recommend items to users, which are likely to be of interest to the users, by utilizing **historical information**.

# Recommender System

- What is it?
- **How to build it?**
- Practical challenges & state-of-the-art
- R package: `recommenderlab`

## Non-personalized RS

Best Selling books

Top Cyber Monday Deals

Most Popular in Electronics

Best Liked

Top 5 Essential Winter Boots

# Recommender System

- What is it?
- **How to build it?**
- Practical challenges & state-of-the-art
- R package: **recommenderlab**

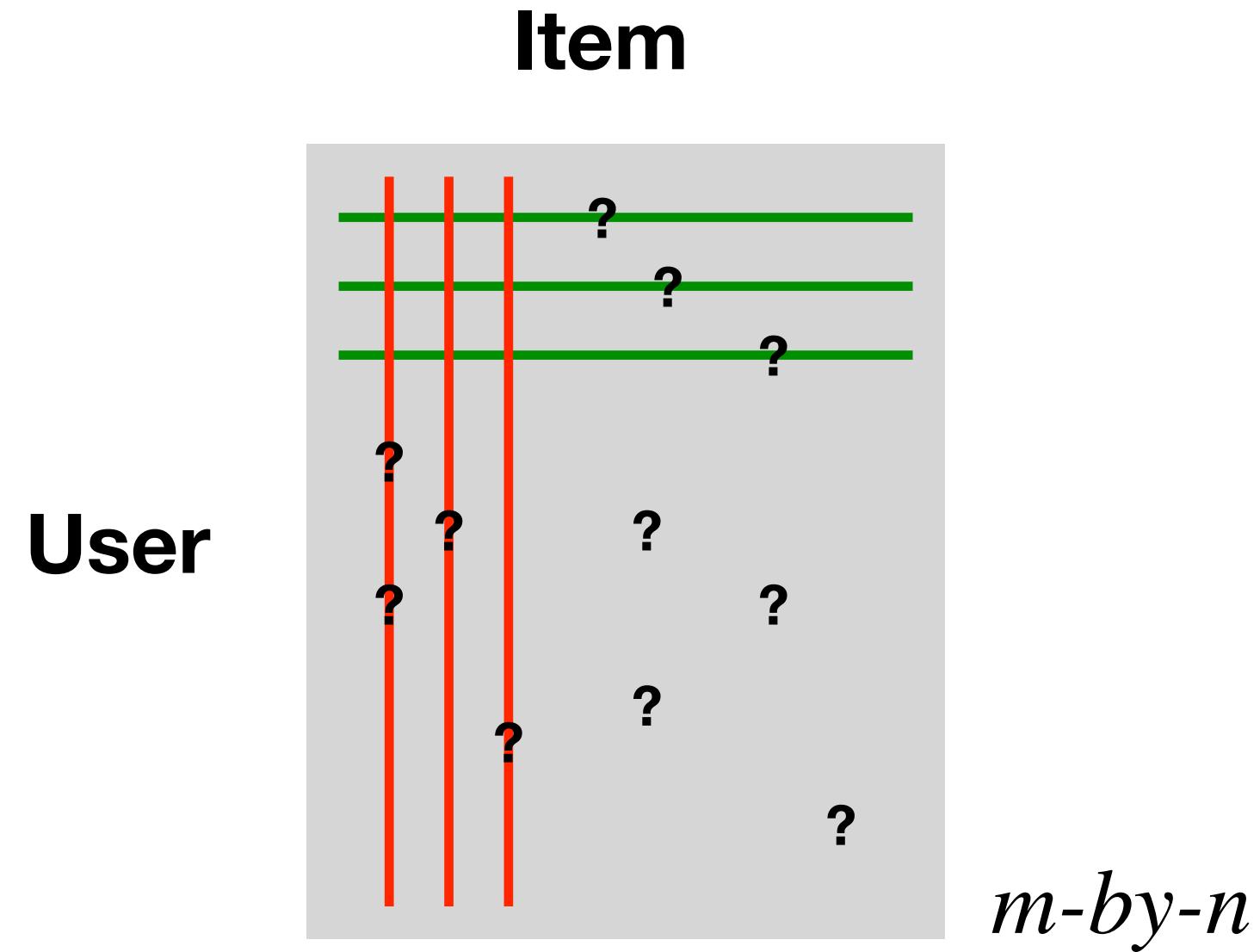
Non-personalized RS

Personalized RS

- **Collaborative Filtering** method
  - Item-based CF
  - User-based CF
- **Latent Factor** method

# Data

## User-Item Rating Matrix: R

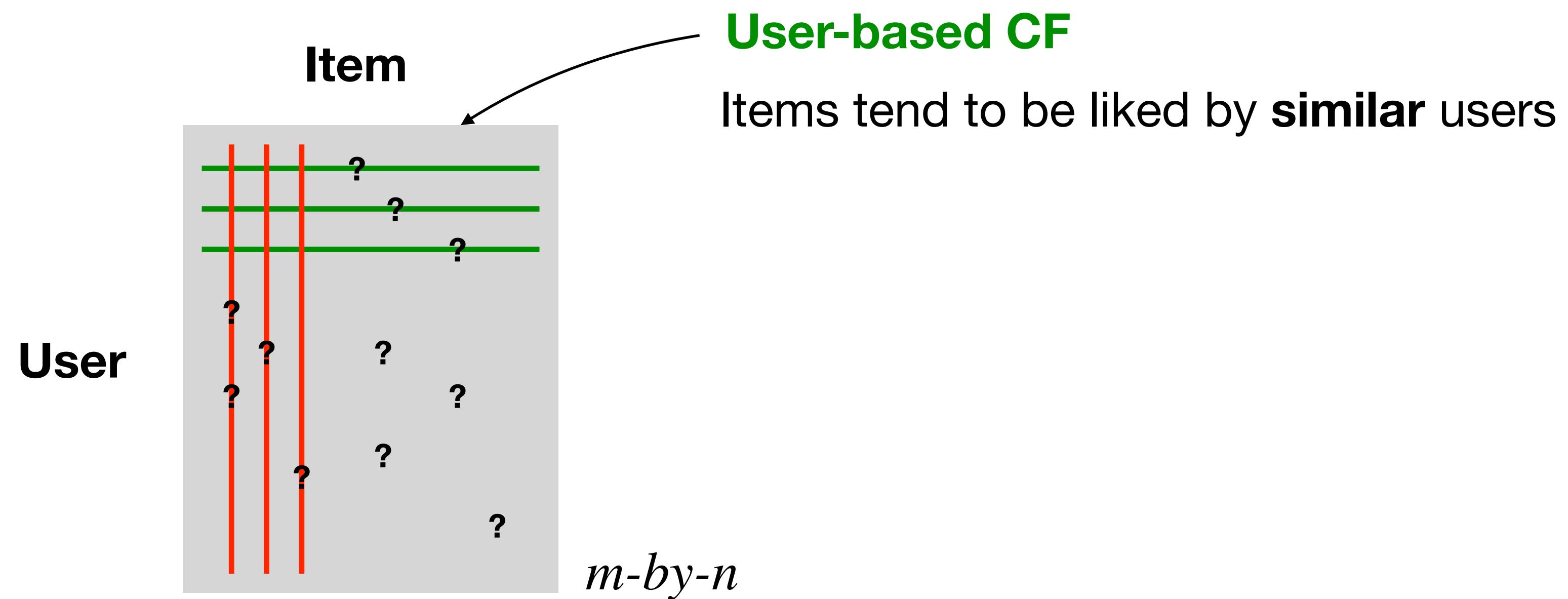


### Two Types of Information

1. Characteristic information about the items/Users
2. User-item interactions

# Collaborative Filtering (CF) Method

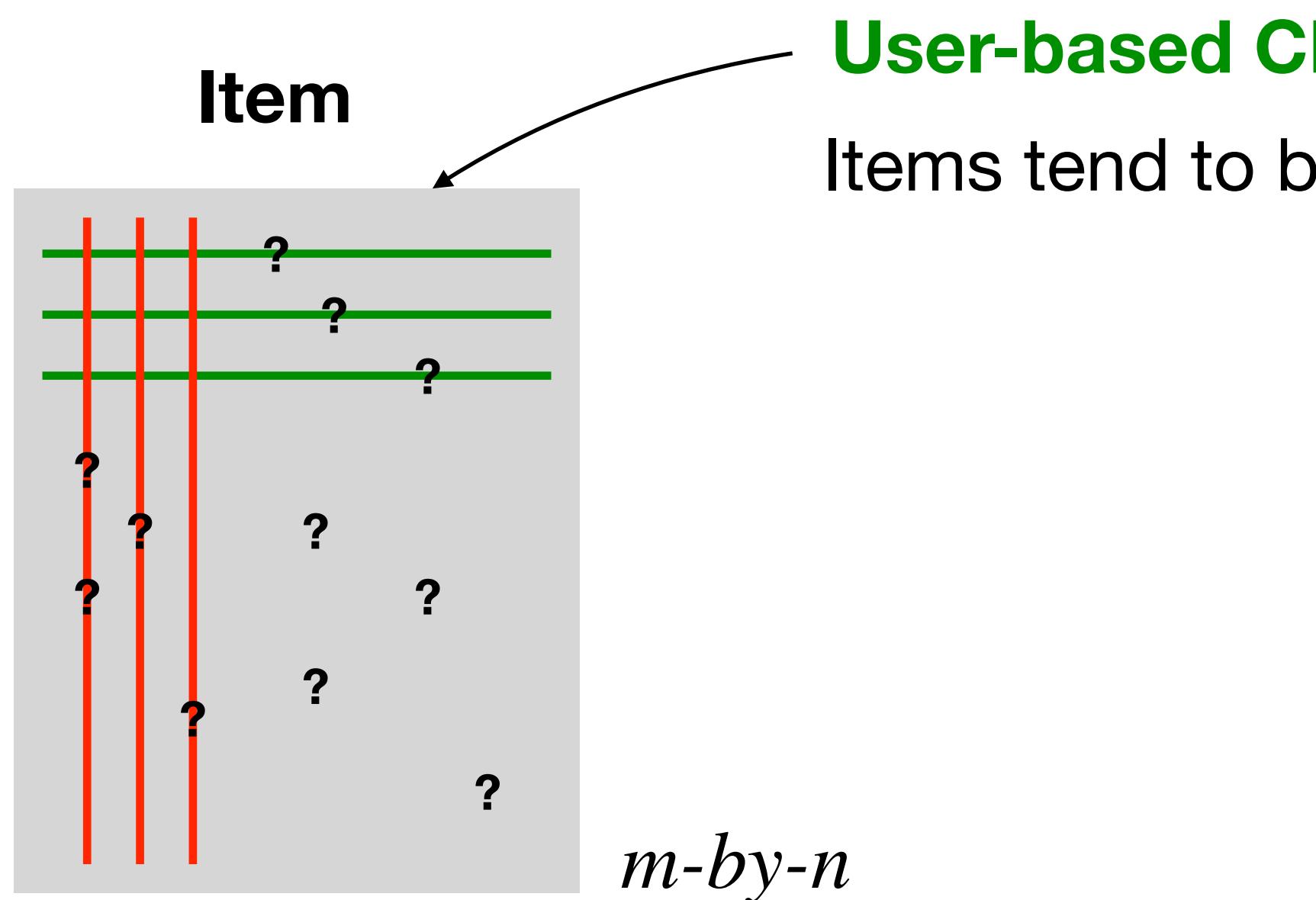
## User-Item Rating Matrix: R



# Collaborative Filtering (CF) Method

## User-Item Rating Matrix: R

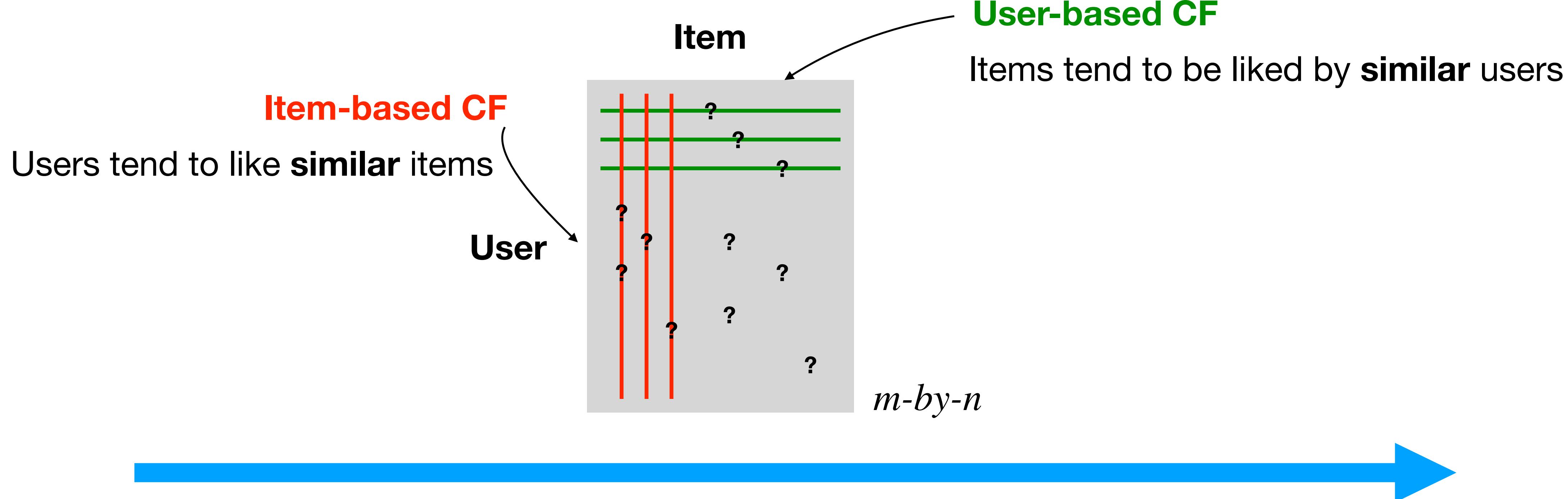
**Item-based CF**  
Users tend to like **similar** items



**User-based CF**  
Items tend to be liked by **similar** users

# Collaborative Filtering (CF) Method

## User-Item Rating Matrix: R



Item Based Recommendation: **Show me more of the same I've liked**

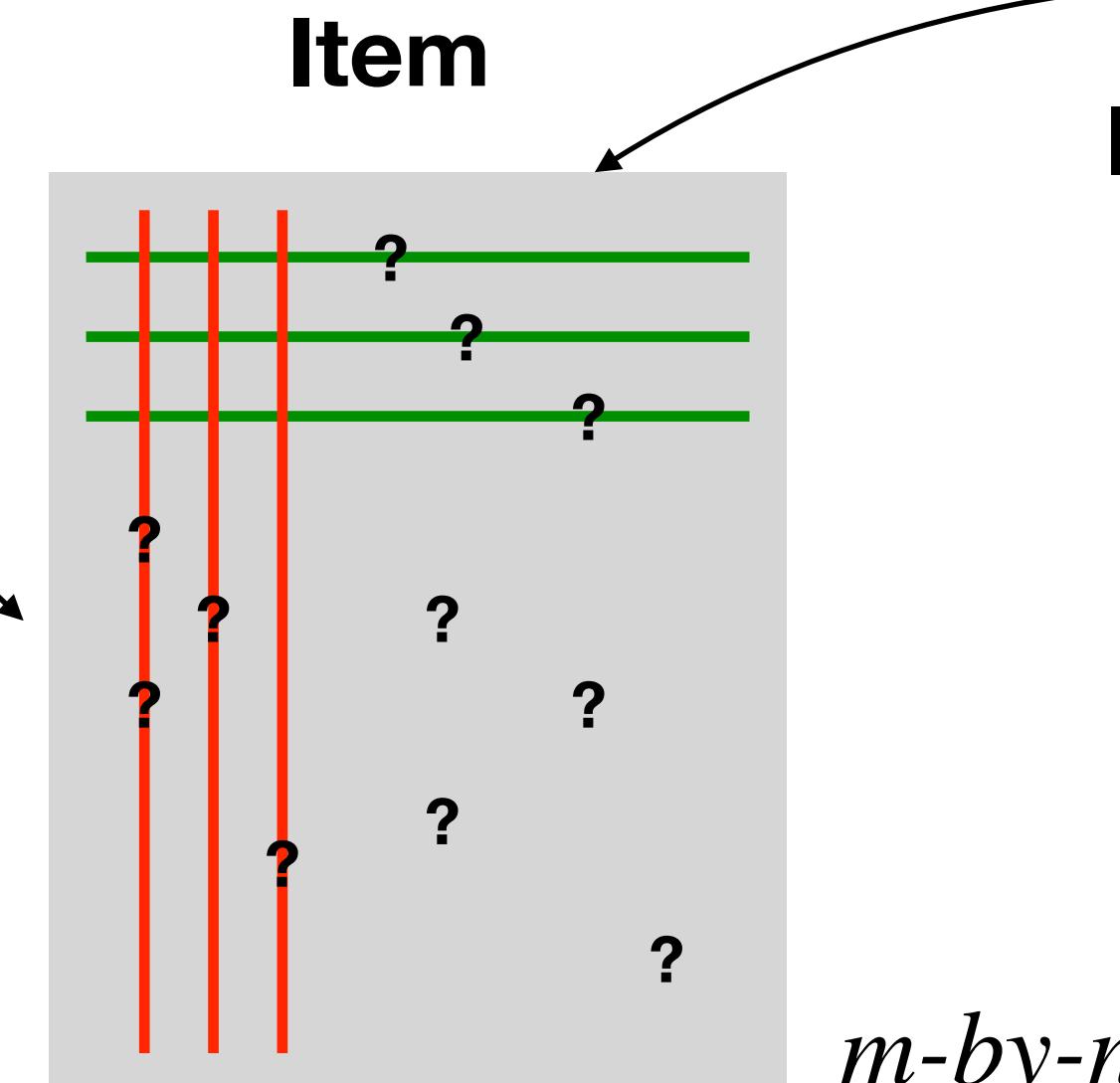
User Based Recommendation: **Tell me what's popular among my peers**

# Collaborative Filtering (CF) Method

## User-Item Rating Matrix: R

**Item-based CF**  
Users tend to like **similar** items

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



## User-based CF

Items tend to be liked by **similar** users

## Similarity Measure

- Jaccard similarity: useful for binary ratings

$$\frac{|A \cap B|}{|A \cup B|}, \quad \text{where } A, B \text{ are two sets.}$$

- Cosine similarity: useful for numerical ratings

$$\frac{u^t v}{\|u\| \cdot \|v\|}, \quad \text{where } u, v \text{ are two vectors}$$

- Centered cosine similarity (Pearson correlation):

$$\frac{(u - \bar{u})^t (v - \bar{v})}{\|u - \bar{u}\| \cdot \|v - \bar{v}\|}, \quad \text{where } u, v \text{ are two vectors}$$

# Collaborative Filtering (CF) Method

## User-Item Rating Matrix: R

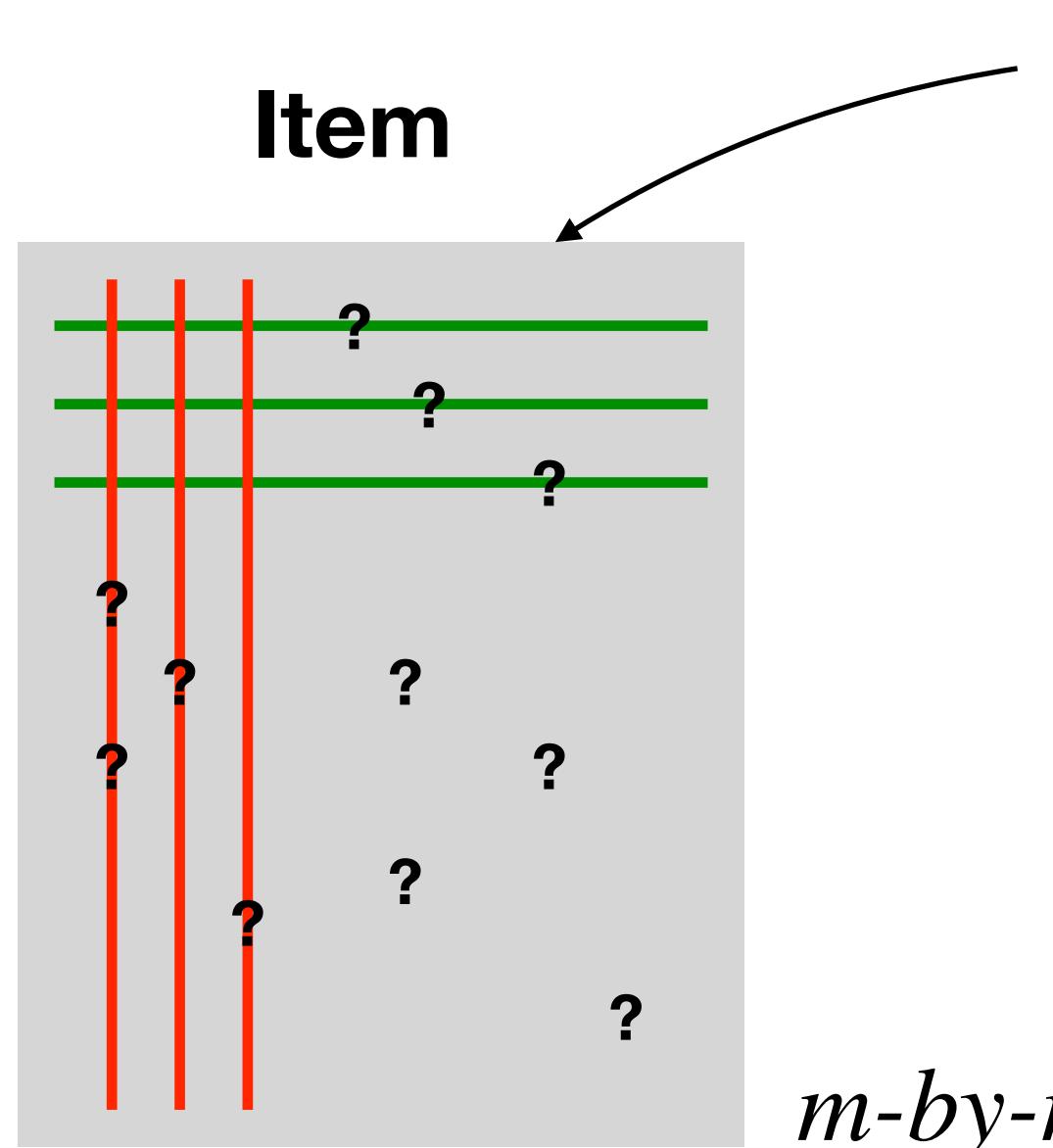
**Item-based CF**  
Users tend to like **similar** items

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

### Advantage of Centering:

1. Missing = Average instead of zero
2. Handle tough/easy raters

	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



### User-based CF

Items tend to be liked by **similar** users

### Similarity Measure

- Jaccard similarity: useful for binary ratings

$$\frac{|A \cap B|}{|A \cup B|}, \quad \text{where } A, B \text{ are two sets.}$$

- Cosine similarity: useful for numerical ratings

$$\frac{u^t v}{\|u\| \cdot \|v\|}, \quad \text{where } u, v \text{ are two vectors}$$

- Centered cosine similarity (Pearson correlation):

$$\frac{(u - \bar{u})^t (v - \bar{v})}{\|u - \bar{u}\| \cdot \|v - \bar{v}\|}, \quad \text{where } u, v \text{ are two vectors}$$

	$i_1$	$i_2$	$i_3$	$i_4$	$i_5$	$i_6$	$i_7$	$i_8$
$u_1$	?	4.0	4.0	2.0	1.0	2.0	?	?
$u_2$	3.0	?	?	?	5.0	1.0	?	?
$u_3$	3.0	?	?	3.0	2.0	2.0	?	3.0
$u_4$	4.0	?	?	2.0	1.0	1.0	2.0	4.0
$u_5$	1.0	1.0	?	?	?	?	?	1.0
$u_6$	?	1.0	?	?	1.0	1.0	?	1.0
$u_a$	?	?	4.0	3.0	?	1.0	?	5.0
$\hat{r}_a$	3.5	4.0		1.3		2.0		

(a)

$$= (3.0 + 4.0)/2$$

$$=(1.0 + 2.0 + 1.0)/3$$

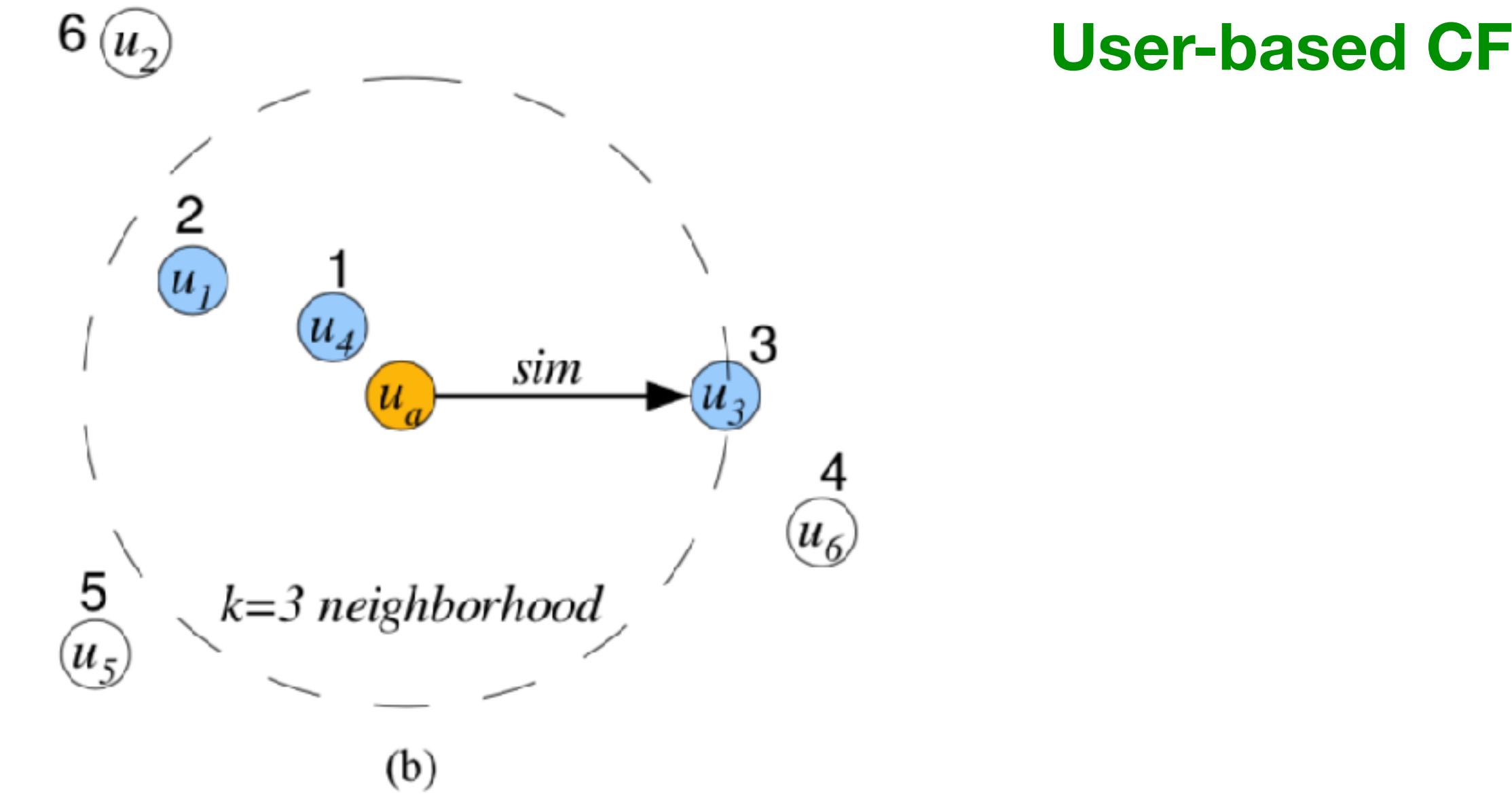


Figure 1: User-based collaborative filtering example with (a) rating matrix and estimated ratings for the active user, and (b) user neighborhood formation.

Source: *recommenderlab: A Framework .... By Michael Hahsler*

Note: We could use different neighborhood for different items, e.g., when computing the rate for item  $i$ , choose kNN among users who have rated item  $i$ , instead of using the same kNN neighborhood for all items.

## Item-based CF

$$\hat{r}_{ai} = \frac{1}{\sum_{j \in S(i) \cap \{l; r_{al} \neq ?\}} s_{ij}} \sum_{j \in S(i) \cap \{l; r_{al} \neq ?\}} s_{ij} r_{aj}$$

**Formula used by Item-based CF (sec 2.2, eq 5)**, where we compute a weighted average of items that are within kNN and also have been rated by this user.

S	$i_1$	$i_2$	$i_3$	$i_4$	$i_5$	$i_6$	$i_7$	$i_8$	$\hat{r}_a$
$i_1$	-	0.1	0	<b>0.3</b>	<b>0.2</b>	<b>0.4</b>	0	0.1	-
$i_2$	0.1	-	<b>0.8</b>	<b>0.9</b>	0	<b>0.2</b>	0.1	0	0.0
$i_3$	0	<b>0.8</b>	-	0	<b>0.4</b>	0.1	0.3	<b>0.5</b>	4.6
$i_4$	<b>0.3</b>	<b>0.9</b>	0	-	0	0.1	0	<b>0.2</b>	3.2
$i_5$	<b>0.2</b>	0	<b>0.4</b>	0	-	0.1	<b>0.2</b>	0.1	-
$i_6$	<b>0.4</b>	<b>0.2</b>	0.1	<b>0.3</b>	0.1	-	0	0.1	2.0
$i_7$	0	<b>0.1</b>	<b>0.3</b>	0	<b>0.2</b>	0	-	0	4.0
$i_8$	<b>0.1</b>	0	<b>0.5</b>	<b>0.2</b>	0.1	0.1	0	-	-

$u_a$	2	?	?	?	4	?	?	5
-------	---	---	---	---	---	---	---	---

Figure 2: Item-based collaborative filtering

Source: *recommenderlab: A Framework .... By Michael Hahsler*

For the similarity matrix shown on the upper-right,

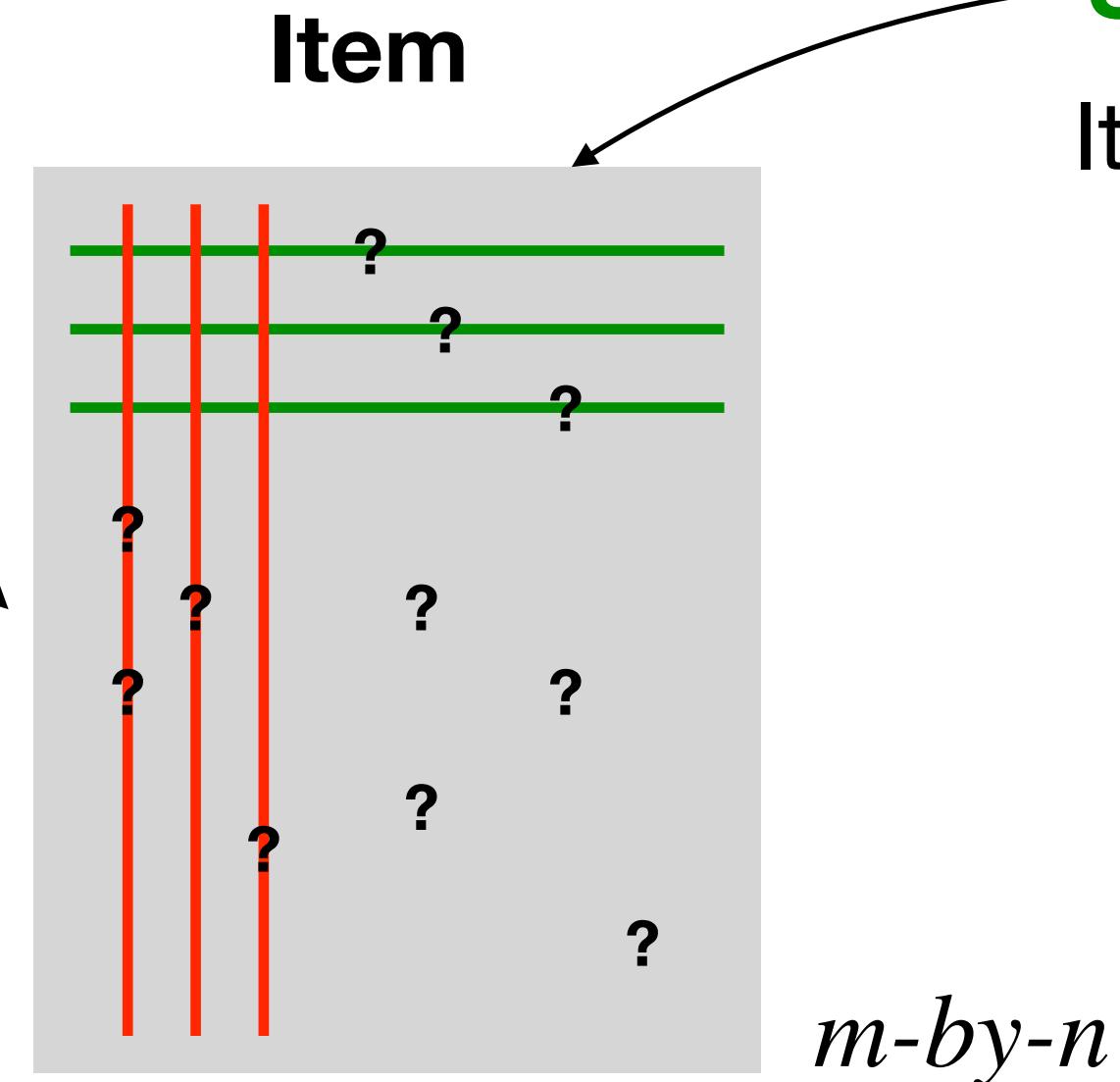
- the largest three entries in each row are **highlighted in bold** since we only consider 3NN
- columns 1, 5, 8 are **highlighted in blue** since the test user (whose ratings we aim to predict) has rated only items 1, 5, 8.

When we compute the weighted average, we only need to consider entries **highlighted both in blue and bold**. This is why the prediction for item 2 is missing (i.e., 0).

**0.0 = 3NN are missing**  
**4.6 = (0.4/0.9)(4) + (0.5/0.9)(5)**  
**3.2 = (0.3/0.5)(2) + (0.2/0.5)(5)**

# Collaborative Filtering (CF) Method

**Item-based CF**  
Users tend to like **similar** items



**User-based CF**

Items tend to be liked by **similar** users

**Pros**

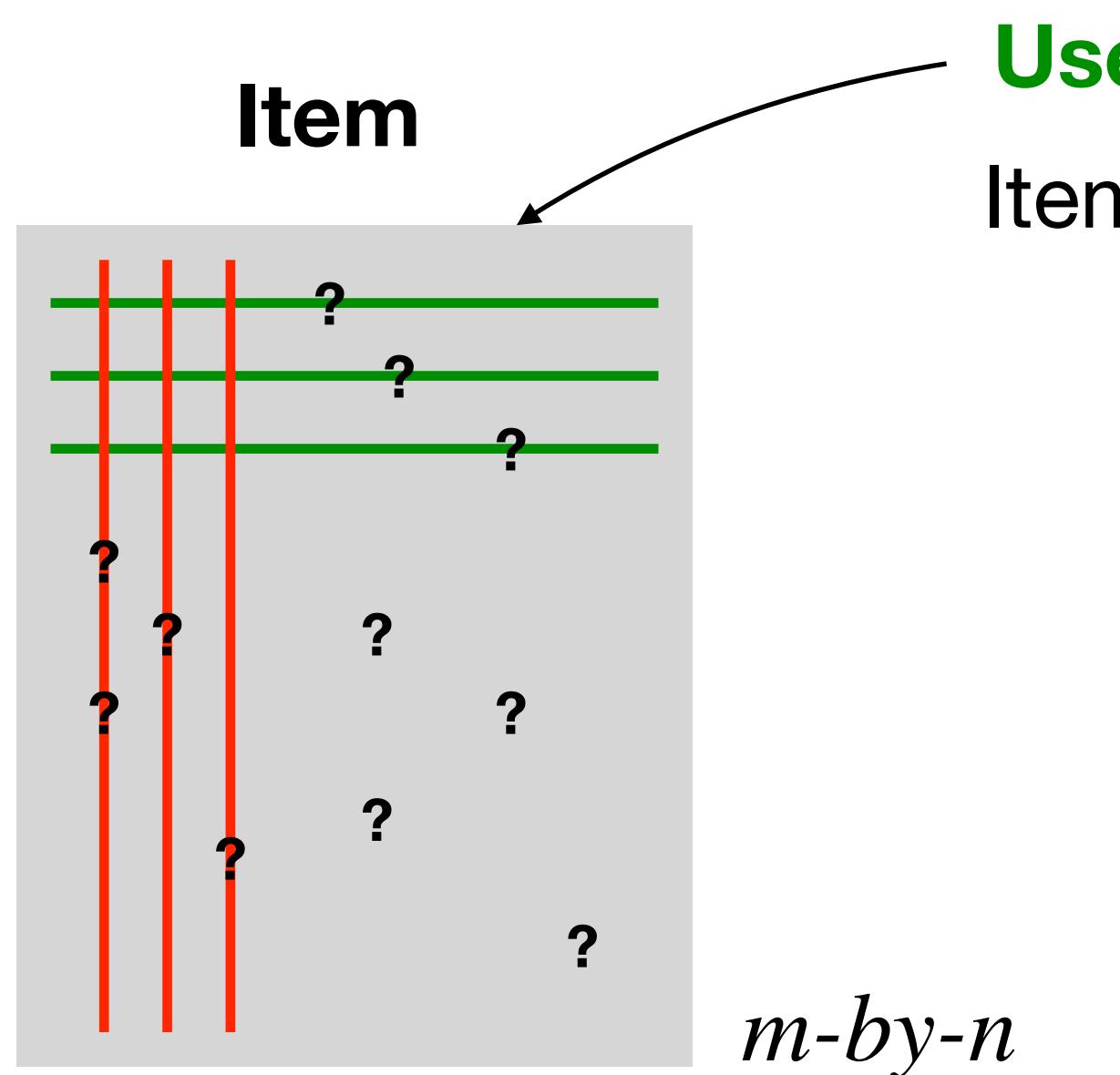
- No need to define features
- Can recommend outside the user's profile

**Cons**

- Need enough user data to start recommendation; cannot operate on day 1
- Cannot recommend new, unrated items
- **Cold start** problem for new users/items

## IBCF vs UBCF

**Item-based CF**  
Users tend to like **similar** items



**User-based CF**

Items tend to be liked by **similar** users

## Computation Cost

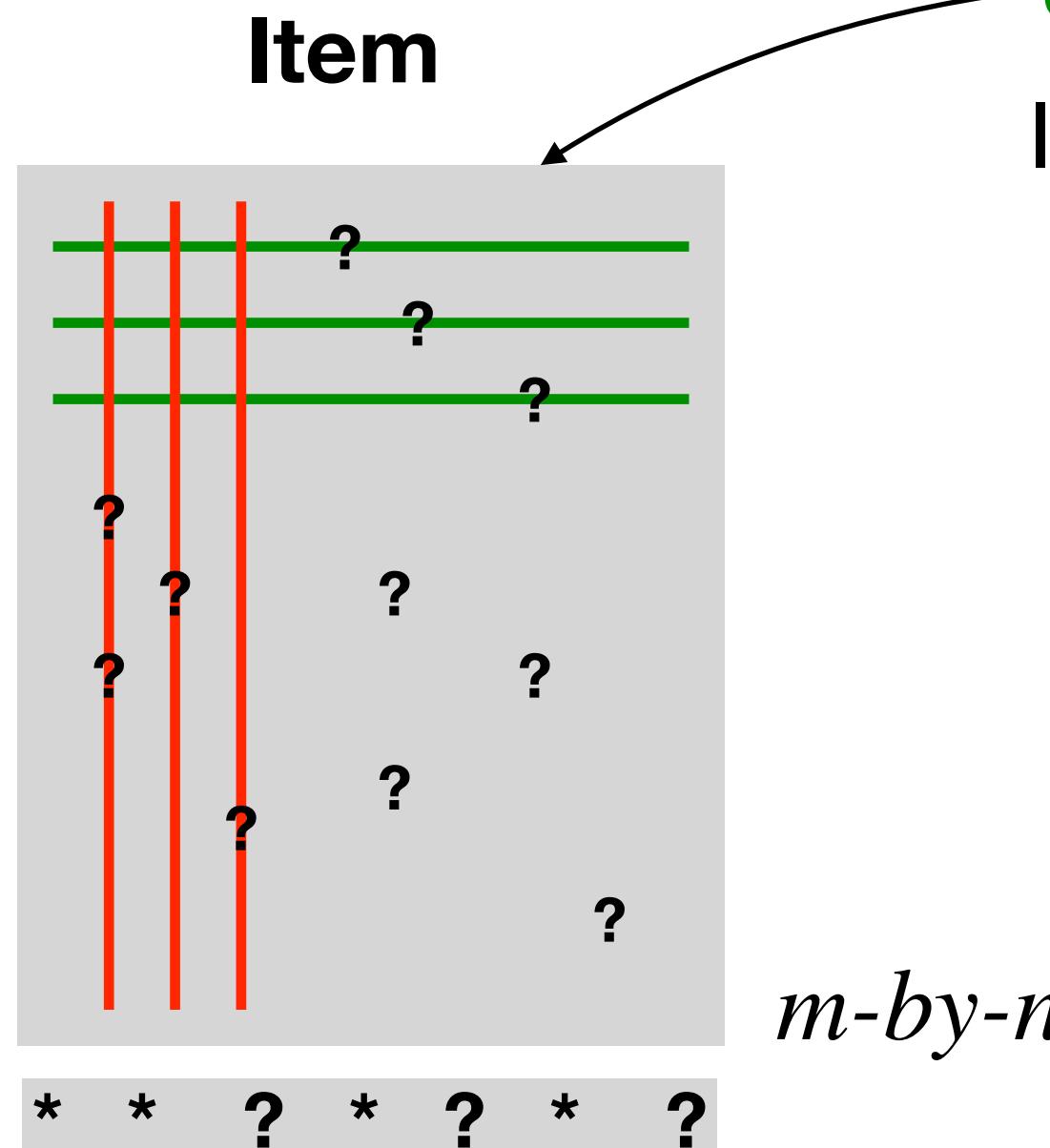
- when filling all ? entries
- when predicting one new user

## IBCF vs UBCF

**Item-based CF**

Users tend to like **similar** items

User



**User-based CF**

Items tend to be liked by **similar** users

## Computation Cost

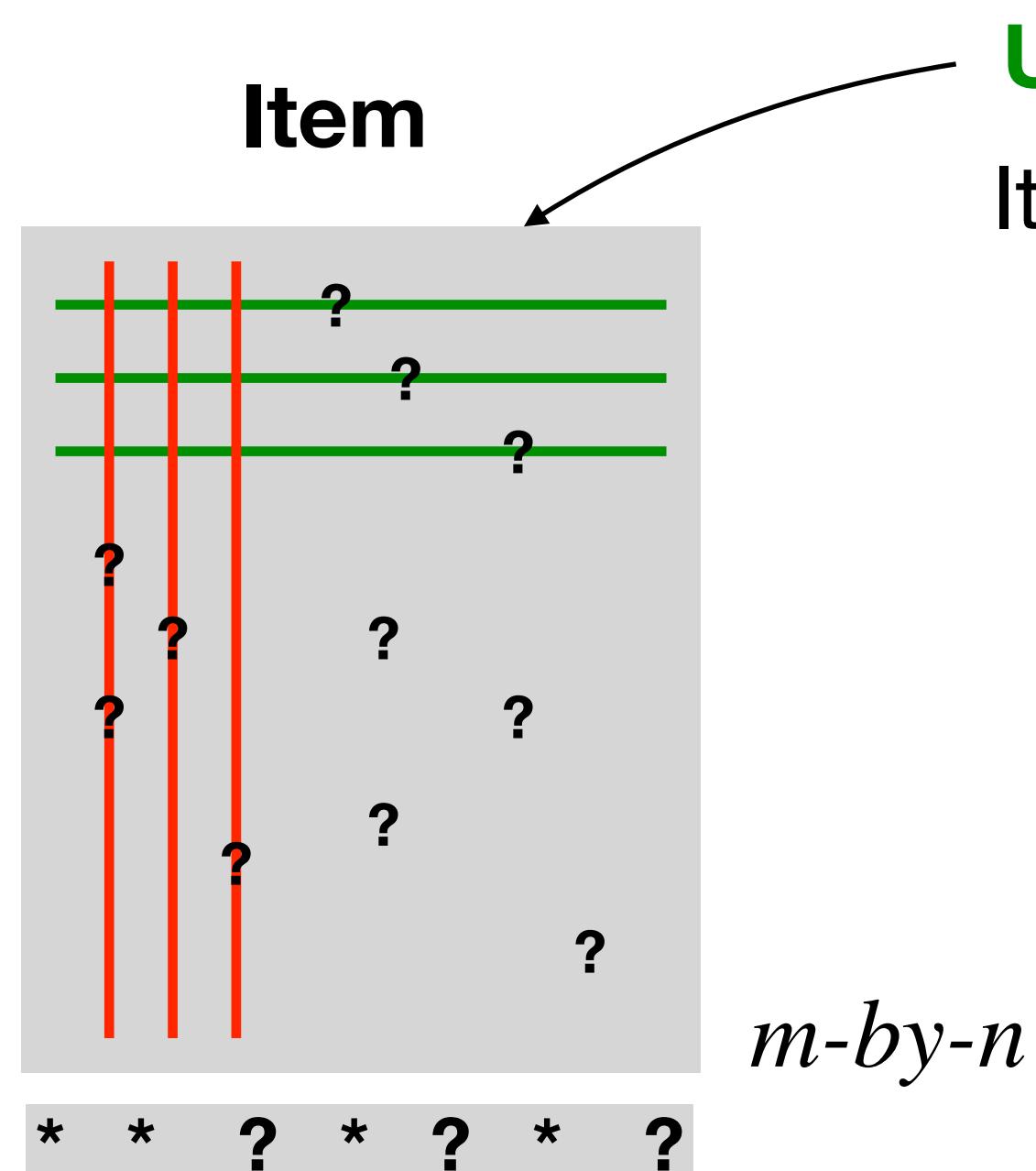
- when filling all ? entries
- when predicting one new user

- **UBCF**: no training; computation is needed at the prediction stage
- **IBCF**: main computation occurs in the training (compute the item-to-item M-by-M similarity matrix as well as the related sorting); memory challenge at the prediction stage

## IBCF vs UBCF

**Item-based CF**

Users tend to like **similar** items



**User-based CF**

Items tend to be liked by **similar** users

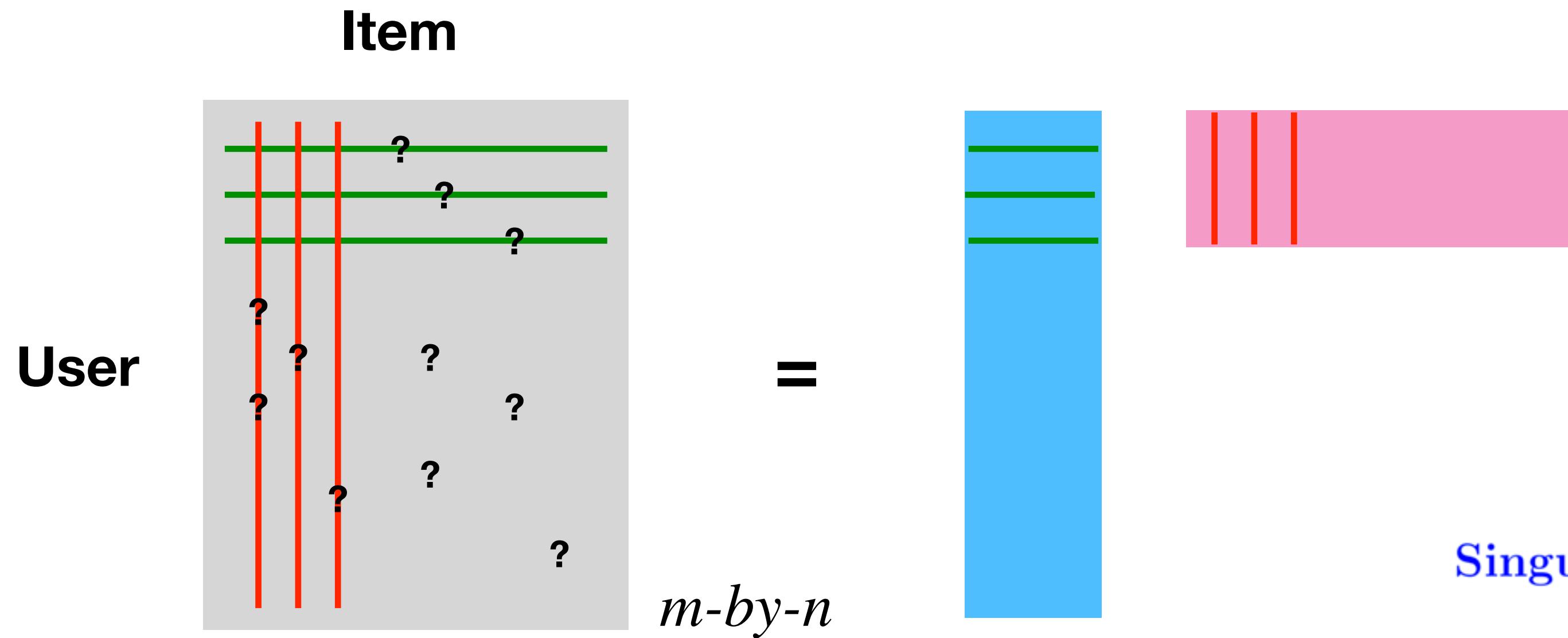
## Computation Cost

- when filling all ? entries
- when predicting one new user

*Which one would Amazon choose?*

# Latent Factor Model

## User-Item Rating Matrix: $R$



## Singular Value Decomposition

Approximate  $R_{m \times n} \approx U_{m \times d} V_{d \times n}^t$  by minimizing

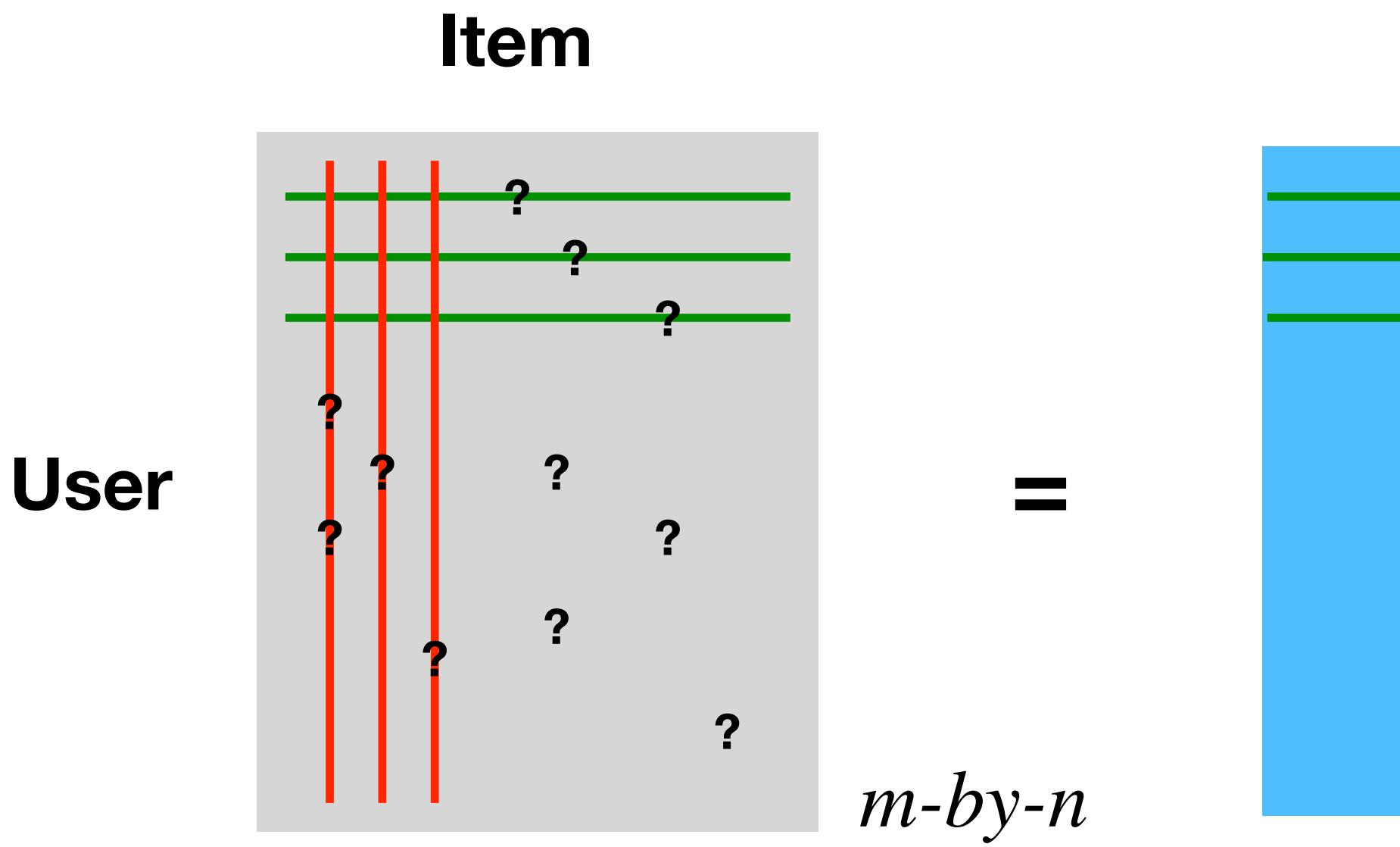
$$\sum_{R_{ij} \neq \text{NA}} (R_{ij} - u_i^t v_j)^2 + \lambda_1 \text{Pen}(U) + \lambda_2 \text{Pen}(V),$$

where  $u_i$  is the  $i$ -th row of matrix  $U$  and  $v_j$  is the  $j$ -th row of matrix  $V$ . Then we can predict any missing entries in  $R$  by the corresponding inner product of  $u_i$  and  $v_j$ .

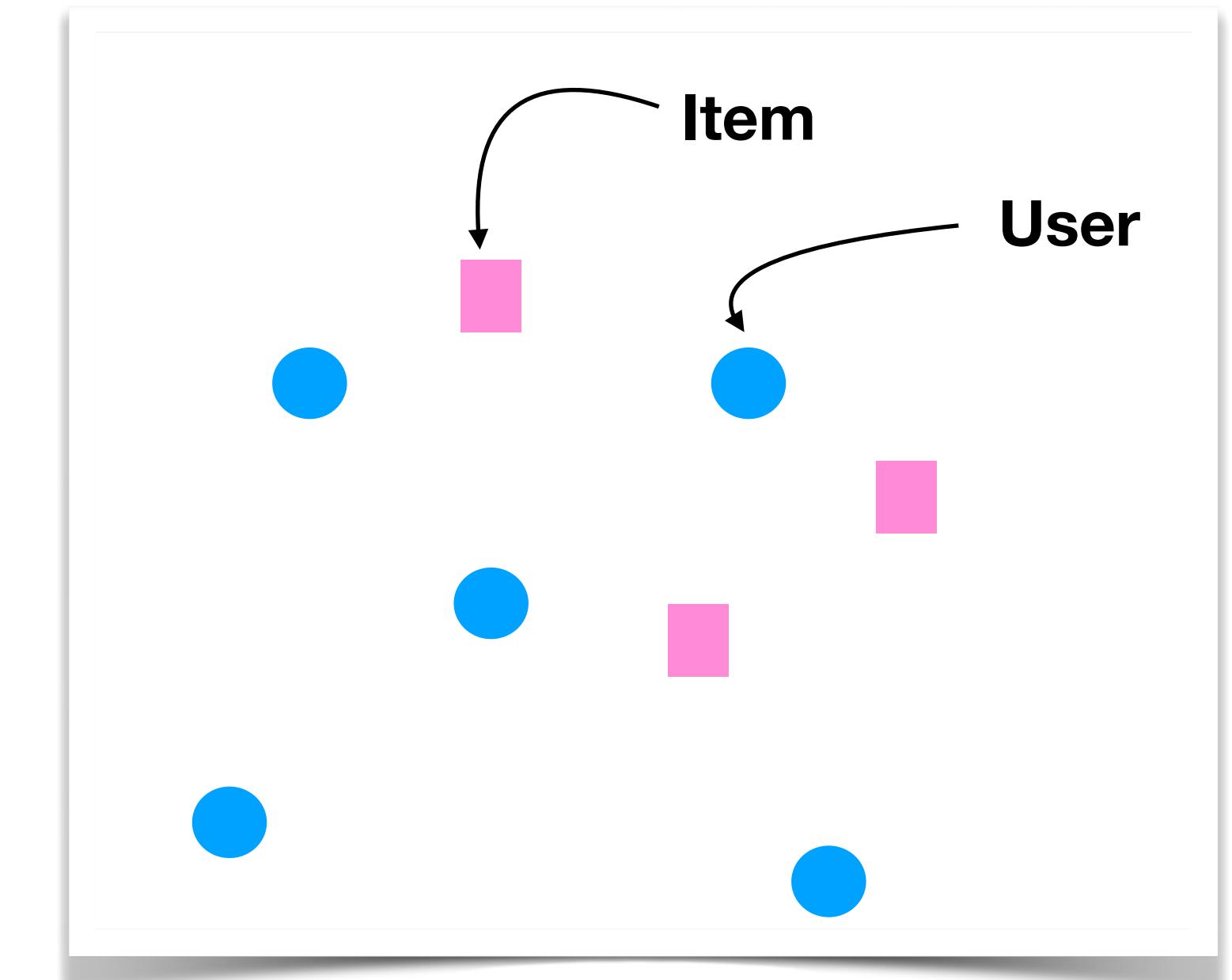
The classical **SVD** algorithm isn't applicable here due to missing entries, instead algorithms based on **Stochastic Gradient Descent** are employed in practice.

# Latent Factor Model

## User-Item Rating Matrix: R



We **embed** the  $n$  users and  $m$  items in a Euclidean space. Then for each user, we can recommend items that are closer to that user.



# Recommender System

---

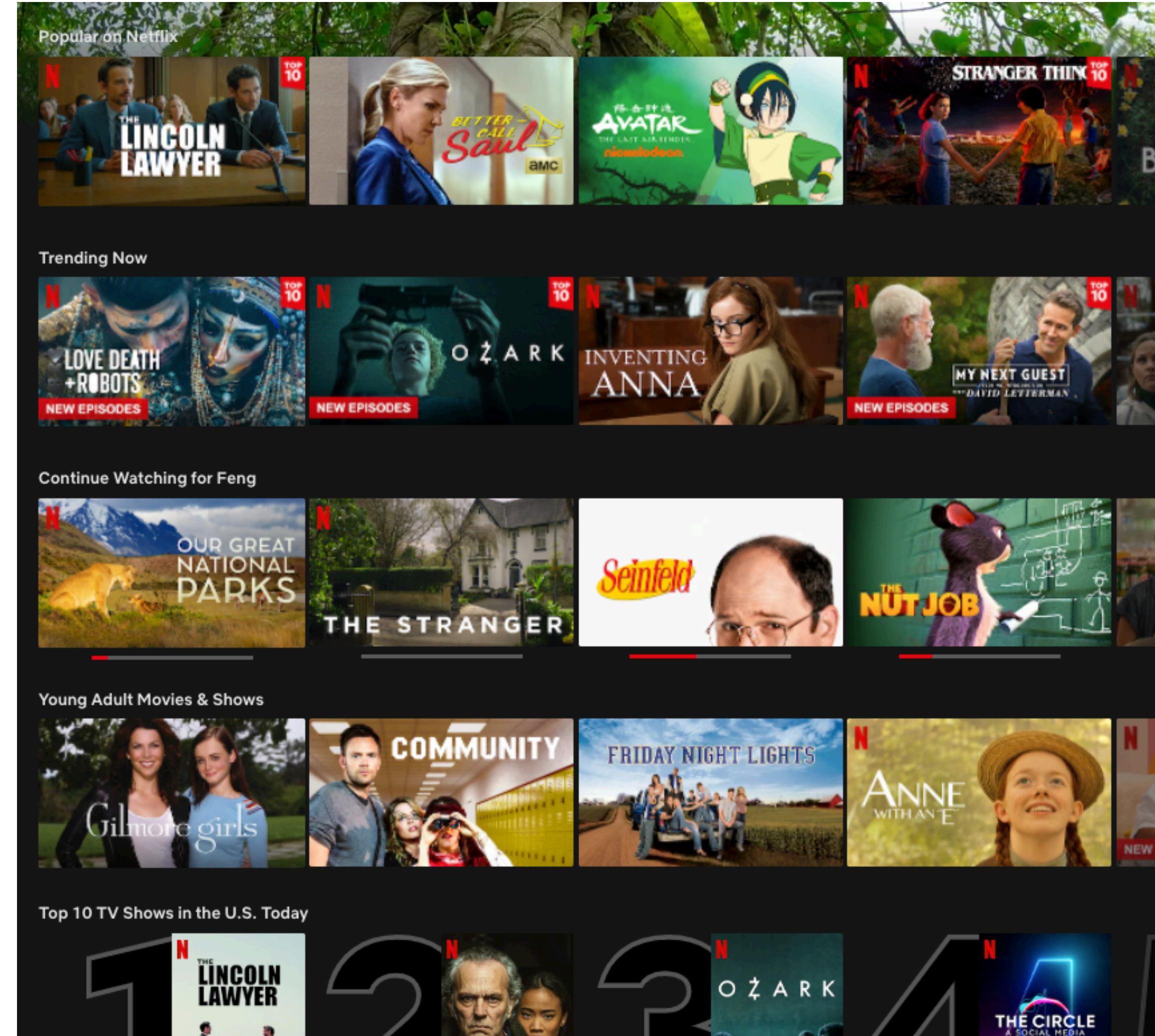
- What is it?
- How to build it?
- **Challenges, solutions & state-of-the-art**
- R package: **recommenderlab**

# Practical Challenges & Issues

---

- **Cluster** users and items to reduce computation
- **Hybrid**: combine multiple recommender systems
- Different **contexts** (location, time, device) and **interface** (computer, mobile) need different recommendation systems.
- How to evaluate a recommender system?
  - **RMSE** vs **Top-k**
  - **Serendipity/Diversity** versus **Accuracy**
- How to incorporate user **feedback**
- **Scalability**: large amount of users and items
- **Sparsity** of the data
- **Utility matrix**: how to construct it based the problem at hand
- **Cold-start**: how to recommend a new item or make recommendation to a new user

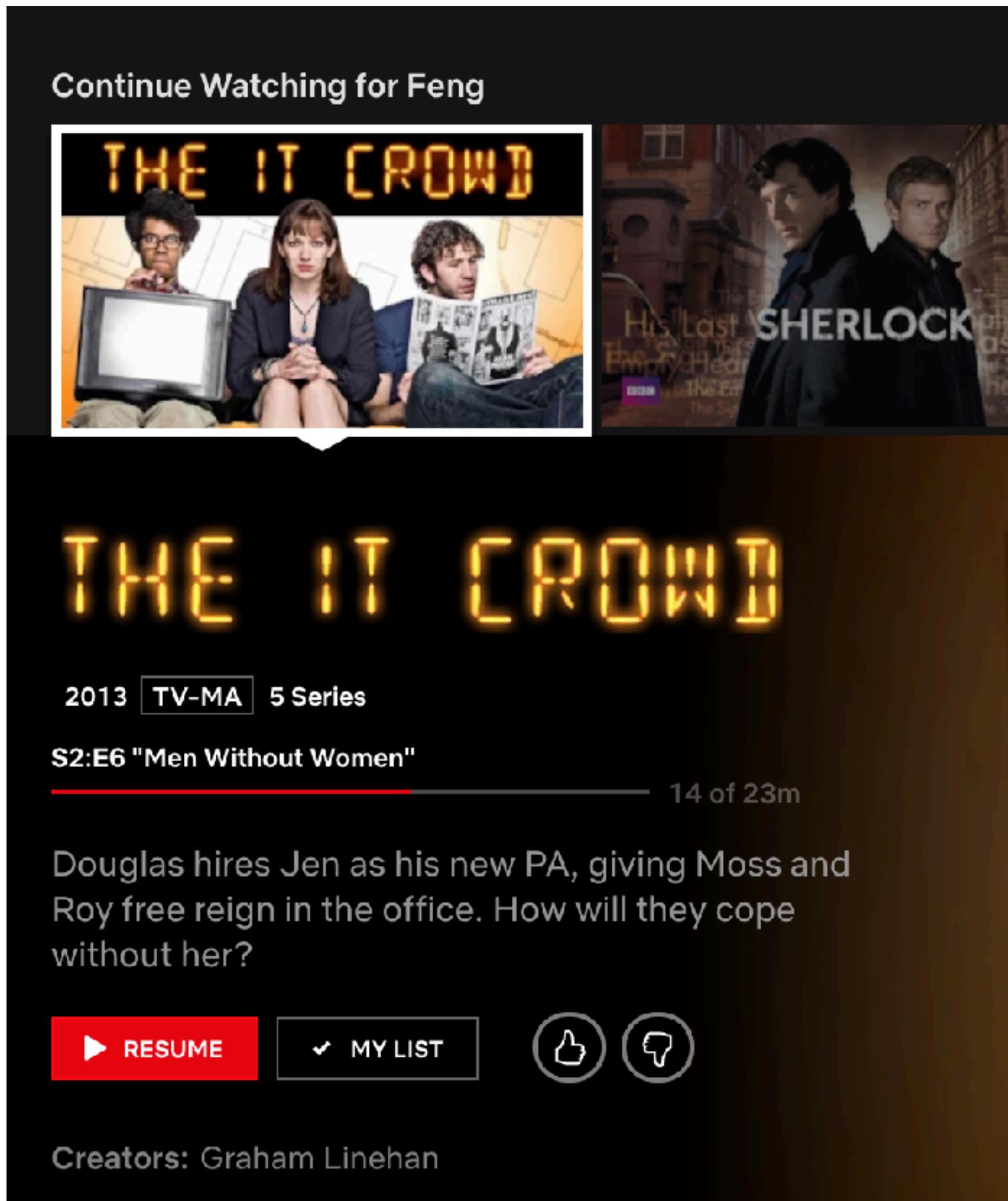
# Practical Issues: Hybrid of Recommender Systems



# Practical Issues: Beyond the Five Star Ratings

HOME > DIGITAL > NEWS

JULY 6, 2018 7:26AM PT



## Netflix Is Shutting Down User Reviews This Summer

"According to Netflix, thumbs-based ratings deliver more accurate recommendations and they're easier for people to understand than the five-star scale. The company said that in testing, it saw a **200%** increase in ratings by users with the thumbs-up/thumbs-down system."

### Netflix will kill its 5-star rating system in favor of 'thumbs-up, thumbs-down' — here's why

Nathan McAlone Mar. 16, 2017, 6:16 PM



"Netflix CPO Neil Hunt told Business Insider that Netflix wanted to do away with the stars, since it considered them a **poor method** of understanding what shows and movies people liked.

The problem, Hunt said at the time, is that people subconsciously try to be critics. When they rate a movie or show from one to five stars, they fall into trying to objectively assess the "quality," instead of basing the stars on how much "enjoyment" they got out of it.

# Deep Recommender Systems

- Use Deep Learning to construct latent factors for items/users
- Train a Deep Learning model to learn the preference between users and items

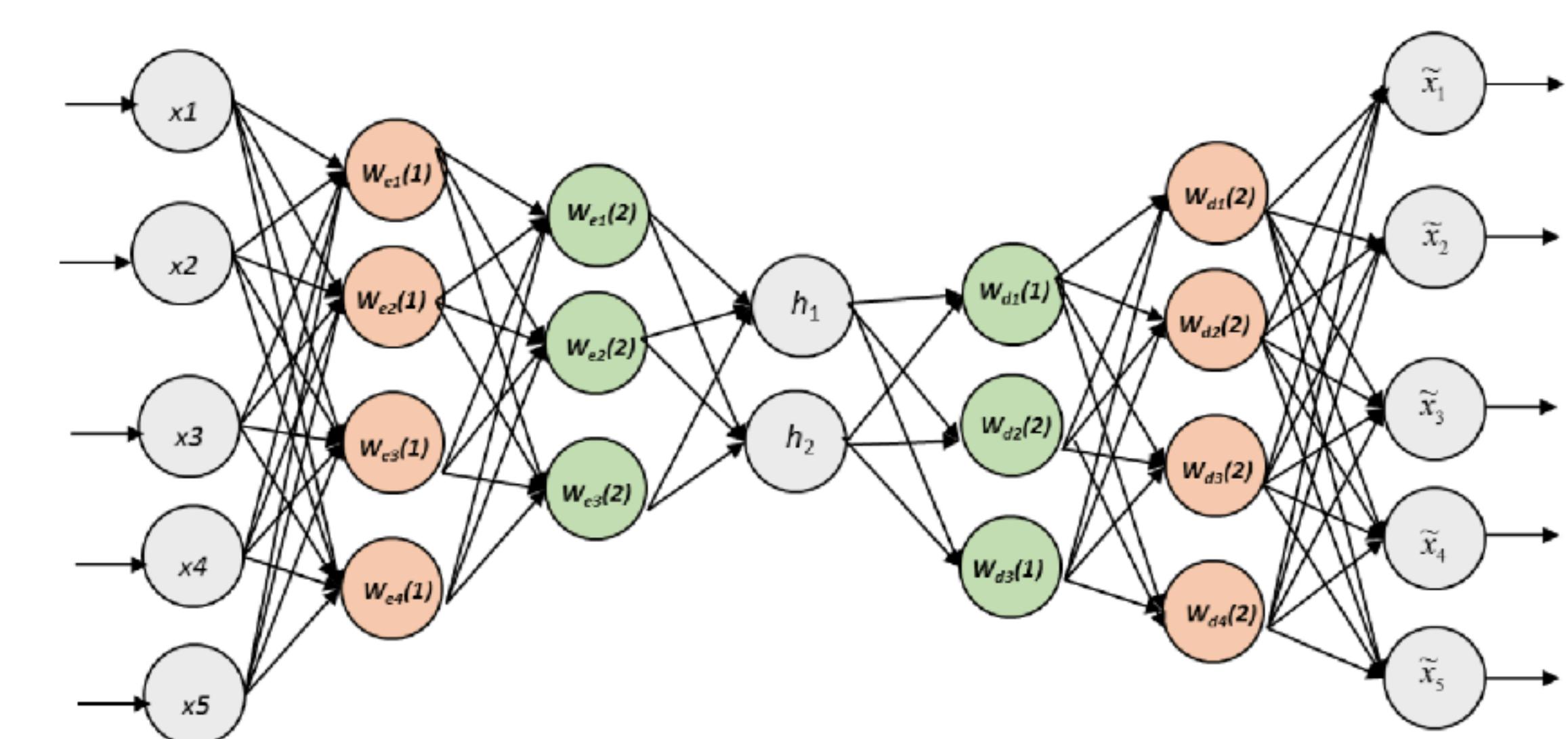
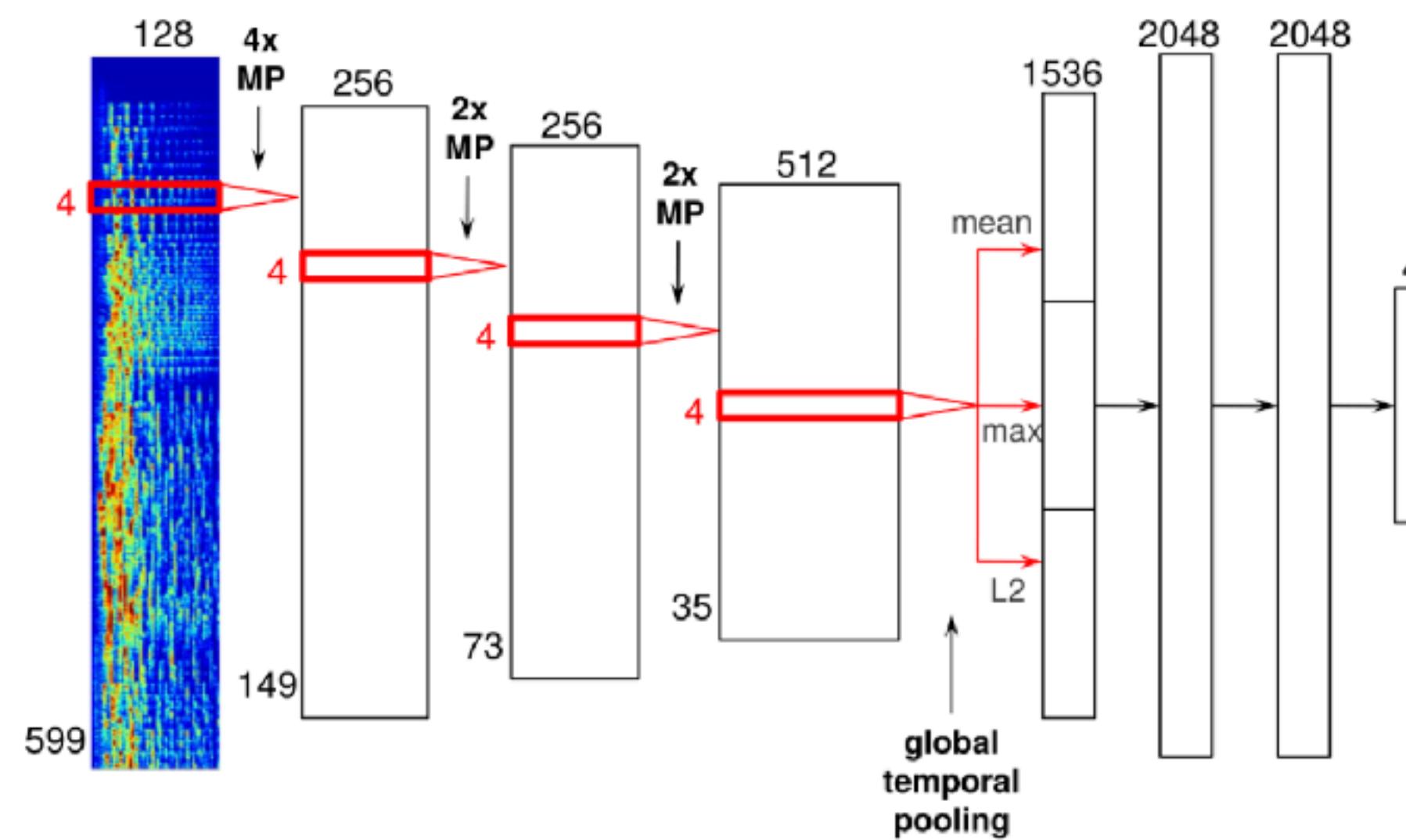


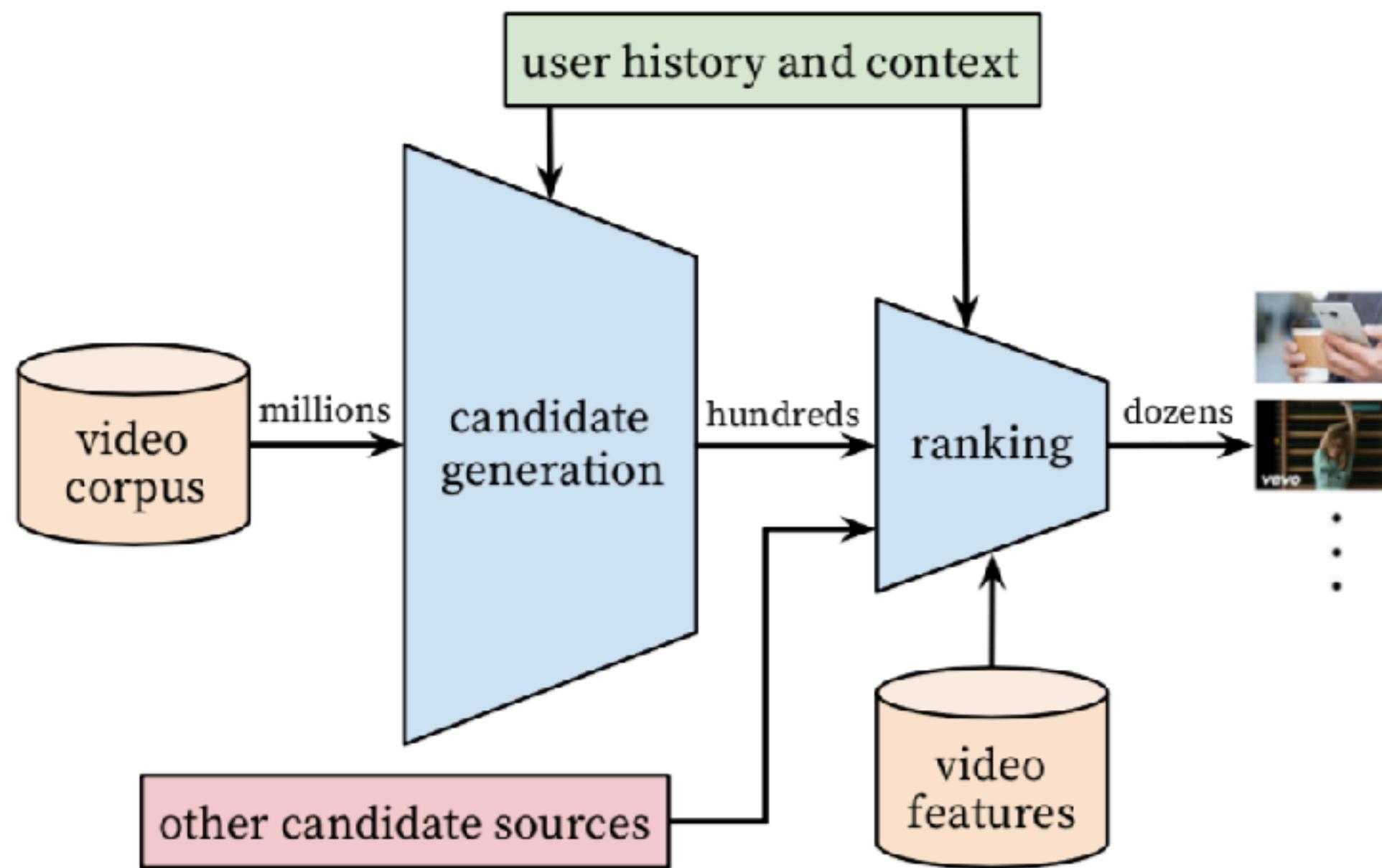
Fig. 2. Deep Autoencoder architecture.

<http://benanne.github.io/2014/08/05/spotify-cnns.html>

<https://towardsdatascience.com/deep-autoencoders-for-collaborative-filtering-6cf8d25bbf1d>

# Deep Recommender Systems

- Use Deep Learning to construct latent factors for items/users
- Train a Deep Learning model to learn the preference between users and items



**Figure 2:** Recommendation system architecture demonstrating the “funnel” where candidate videos are retrieved and ranked before presenting only a few to the user.

Covington et al. (2016)

# Recommender System

---

- What is it?
- How to build it?
- Challenges, new directions and state-of-the-art
- **R package: recommenderlab**

Along with Shiny App, you can build your own recommender system:

<https://fengliang.shinyapps.io/MovieRecommend/>