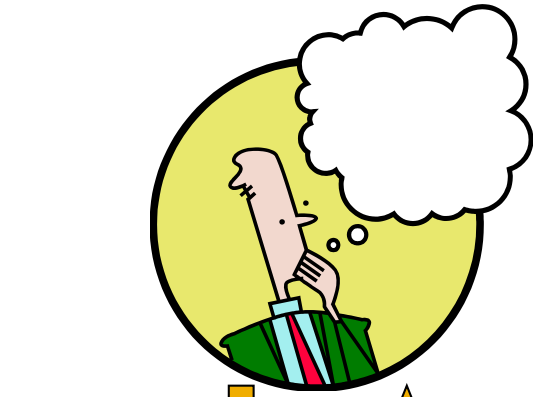
Recommender Systems: Content-based Systems & Collaborative Filtering

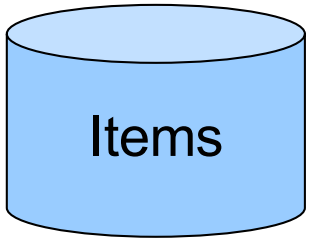
Slides borrowed (with modifications) from Mining of Massive Datasets
Jure Leskovec, Anand Rajaraman, Jeff Ullman
http://www.mmds.org

Recommendation Systems (RS)



Search

Recommendations



Products, web sites, blogs, news items, …

Examples:

amazon.com

del.icio.us

NETFLIX

movielens
helping you find the *right* movies

last·fm
the social music revolution

Google News

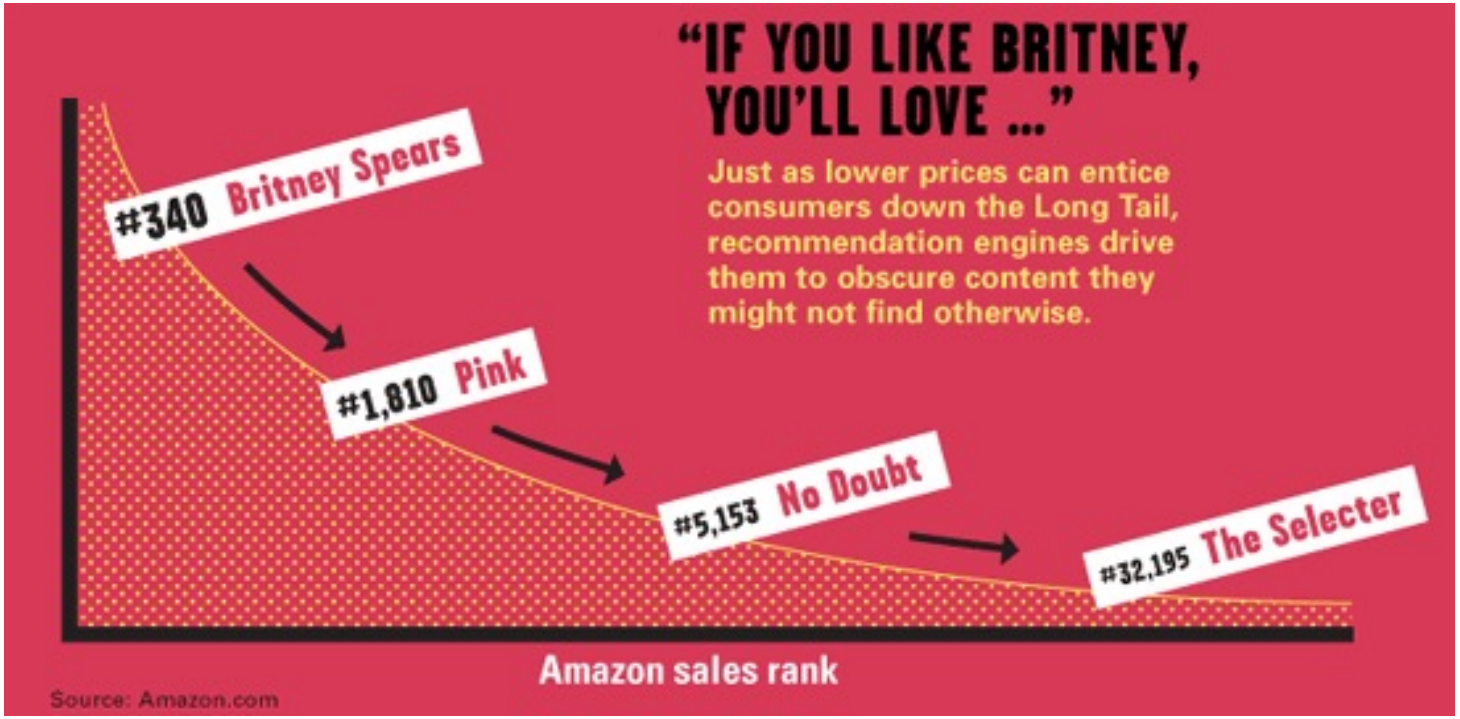
YouTube

XBOX LIVE

Why RS: From Scarcity to Abundance

- **Shelf space is a scarce commodity for traditional markets/retailers**
  - Only limited nos. of items can be displayed
  - Also, no scope of personalizing to individual users
  - Also for newspapers, movie theaters, …
- **Web enables near-zero-cost dissemination of information about products**
  - From scarcity to abundance
  - Scope of personalizing to individual users
- **More choice necessitates better filters**
  - Recommendation Systems

Physical vs. Online



A physical store can only show the most popular items to the left.
An online store can show the entire range, including the popular items and the long tail

Read http://www.wired.com/wired/archive/12.10/tail.html to learn more!

Types of Recommendations

- **Editorial and hand curated**
  - List of favorites
  - Lists of "essential" items
- **Simple aggregates**
  - Top 10, Most Popular, Recent Uploads
- **Tailored / personalized to individual users**
  - Amazon, Netflix, Youtube, …

Formal Model

- X = set of **Customers**
- S = set of **Items**

- **Utility function** $u: X \times S \rightarrow R$
  - R = set of ratings
  - R is a totally ordered set
  - e.g., **0-5** stars, real number in **[0,1]**
  - In practice, R can simply be 0 or 1 (whether a user has seen/purchased an item or not)

Utility Matrix

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

Utility Matrix

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

Goal of RS: predict the blanks in the Utility Matrix.

In practice, no need to predict each entry. Rather, for each row (user), predict some entries that are likely to have high values

Key Problems for RS

- **(1) Gathering "known" ratings for matrix**
  - How to collect the data in the utility matrix
- **(2) Extrapolate unknown ratings from the known ones**
  - Mainly interested in high unknown ratings
  - We are not interested in knowing what you don't like but what you would like
- **(3) Evaluating recommendation 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 usually not bothered to give ratings
- **Implicit**
  - Learn ratings from user actions
    - E.g., purchase implies high rating
  - What about low ratings?

(2) Extrapolating Utilities

- **Key problem:** Utility matrix ***U*** is **sparse**
  - Most people have not rated most items
  - **Cold start:**
    - New items have no ratings
    - New users have no history

- **Three approaches to recommender systems:**
  - **1)** Content-based
  - **2)** Collaborative Filtering
  - **3)** Latent factor based

We will study the first two approaches

Content-based Recommender Systems

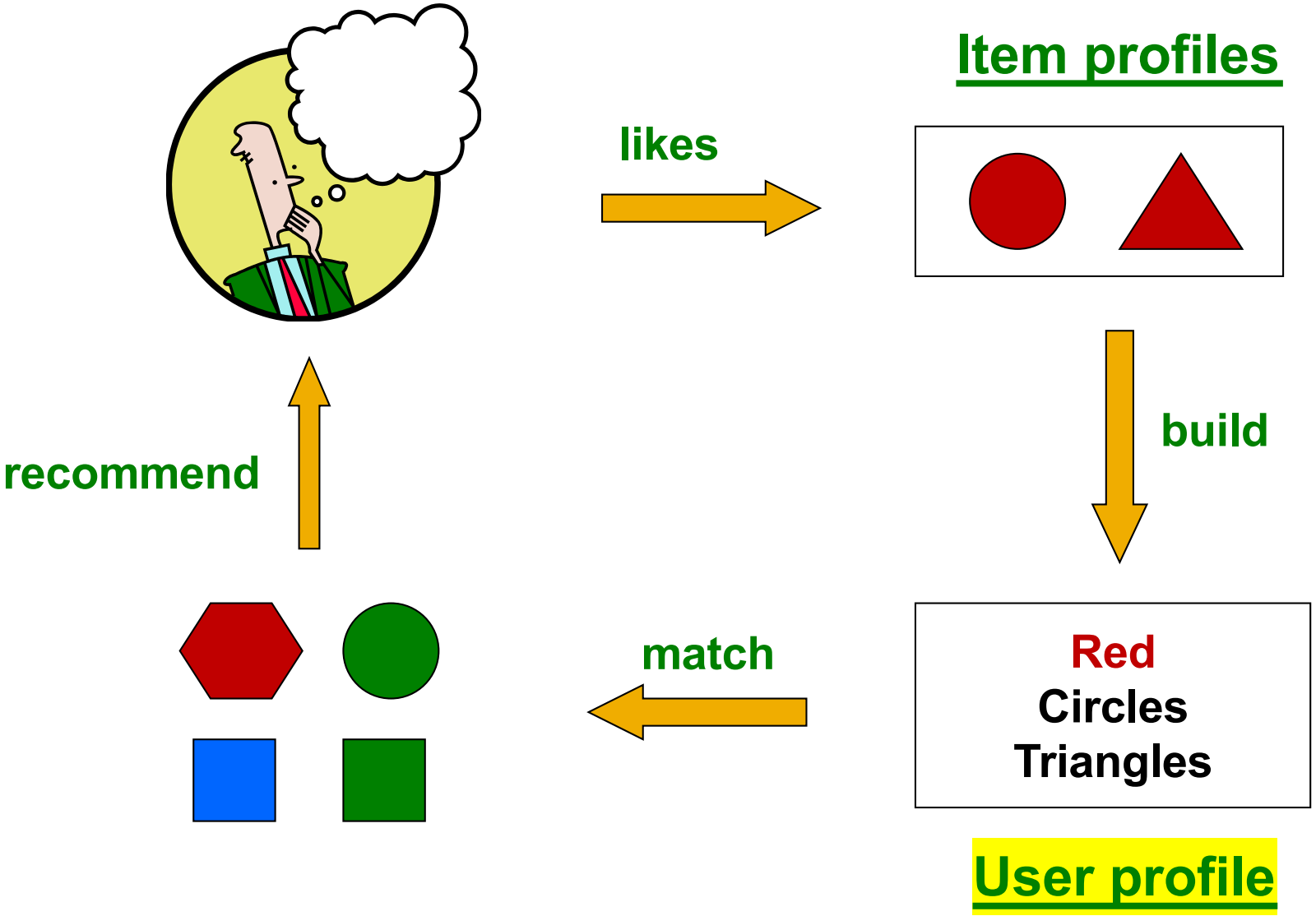
Content-based Recommendations

- **Main idea:** Recommend items to user ***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

- For each item, create an **item profile** representing important characteristics of item
- **Profile is a set (vector) of features**
  - **Movies:** author, title, actor, director,…
  - **Text:** Set of "important" words in document
- **How to pick important features?**
  - Usual heuristic from text mining is **TF-IDF** (Term frequency * Inverse Doc Frequency)
    - **Term … Feature**
    - **Document … Item**

Sidenote: TF-IDF

f_{ij} = frequency of term (feature) i in doc (item) j

$$TF_{ij} = \frac{f_{ij}}{\max_k f_{kj}}$$

Note: we normalize TF to discount for "longer" documents

n_i = number of docs that mention term i

N = total number of docs

$$IDF_i = \log \frac{N}{n_i}$$

TF-IDF score: $w_{ij} = TF_{ij} \times IDF_i$

Doc profile = set of words with highest **TF-IDF** scores, together with their scores

User Profiles and Prediction

- **User profile possibilities:**
  - Weighted average of rated item profiles
  - **Variation:** If utility matrix contains ratings, can normalize the ratings by subtracting the average value for a particular user (-ve weights for below-avg ratings)
  - …
- **Prediction heuristic:**
  - 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**
  - Can provide explanations of recommended items by listing content-features that caused an item to be recommended

Cons: Content-based Approach

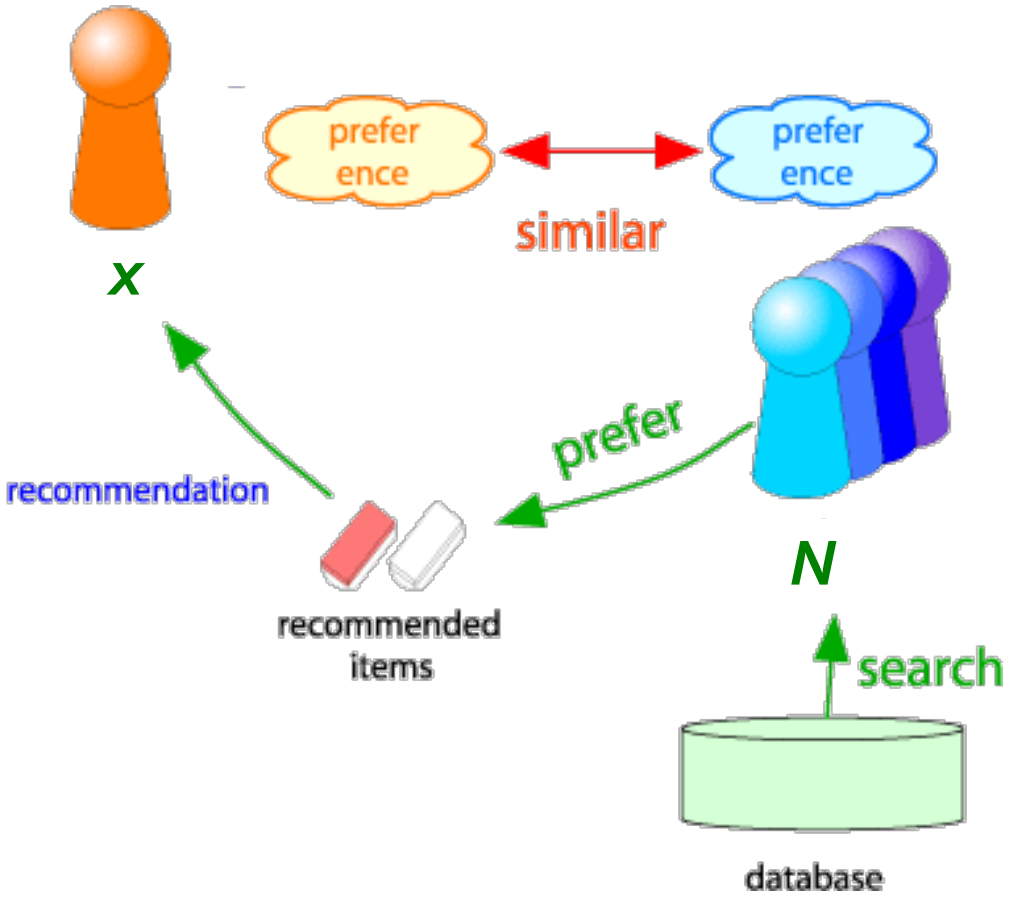
- **–: Finding the appropriate features is hard**
  - E.g., images, movies, music
- **–: Recommendations for new users**
  - How to build a user profile?
- **–: Overspecialization**
  - Never recommends items outside user's content profile
  - **Unable to exploit quality judgments of other users**

Collaborative Filtering

Harnessing quality judgments of other users

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***



Collaborative Filtering

- In place of using the features of items to determine their similarity, use the similarity between the user-ratings of two items
  - *In place of item-profile vector, use the column corresponding to that item in the utility matrix*
- Instead of having a user-profile vector, represent a user by the corresponding row of the utility matrix
  - *Two users are similar if their rows (ratings for different items) are similar*

Finding "Similar" Users

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

- Let r_x be the vector of user x's ratings
- **Jaccard similarity measure**
  - **Problem:** Ignores the value of the rating
- **Cosine similarity measure**
  - $\text{sim}(x, y) = \cos(r_x, r_y) = \frac{r_x \cdot r_y}{||r_x|| \cdot ||r_y||}$
  - **Problem:** Treats missing ratings as "negative"
- **Pearson correlation coefficient**
  - S_{xy} = items rated by both users x and y

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

r_x, r_y as sets:
r_x = {1, 4, 5}
r_y = {1, 3, 4}

r_x, r_y as points:
r_x = {1, 0, 0, 1, 3}
r_y = {1, 0, 2, 2, 0}

$\overline{r}_x$, $\overline{r}_y$ … avg. rating of **x**, **y**

Similarity Metric

Cosine sim:
$$sim(x,y) = \frac{\sum_i r_{xi} \cdot r_{yi}}{\sqrt{\sum_i r_{xi}^2} \cdot \sqrt{\sum_i r_{yi}^2}}$$

	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*)**
- **Jaccard similarity:** 1/5 < 2/4
- **Cosine similarity:** 0.386 > 0.322
  - Considers missing ratings as "negative"
  - **Solution: subtract the (row) mean**

	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

sim A,B vs. A,C:
0.092 > -0.559

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*:**
  - $r_{xi} = \frac{1}{k} \sum_{y \in N} r_{yi}$
  - $r_{xi} = \frac{\sum_{y \in N} s_{xy} \cdot r_{yi}}{\sum_{y \in N} s_{xy}}$

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

- **Many other tricks possible…**

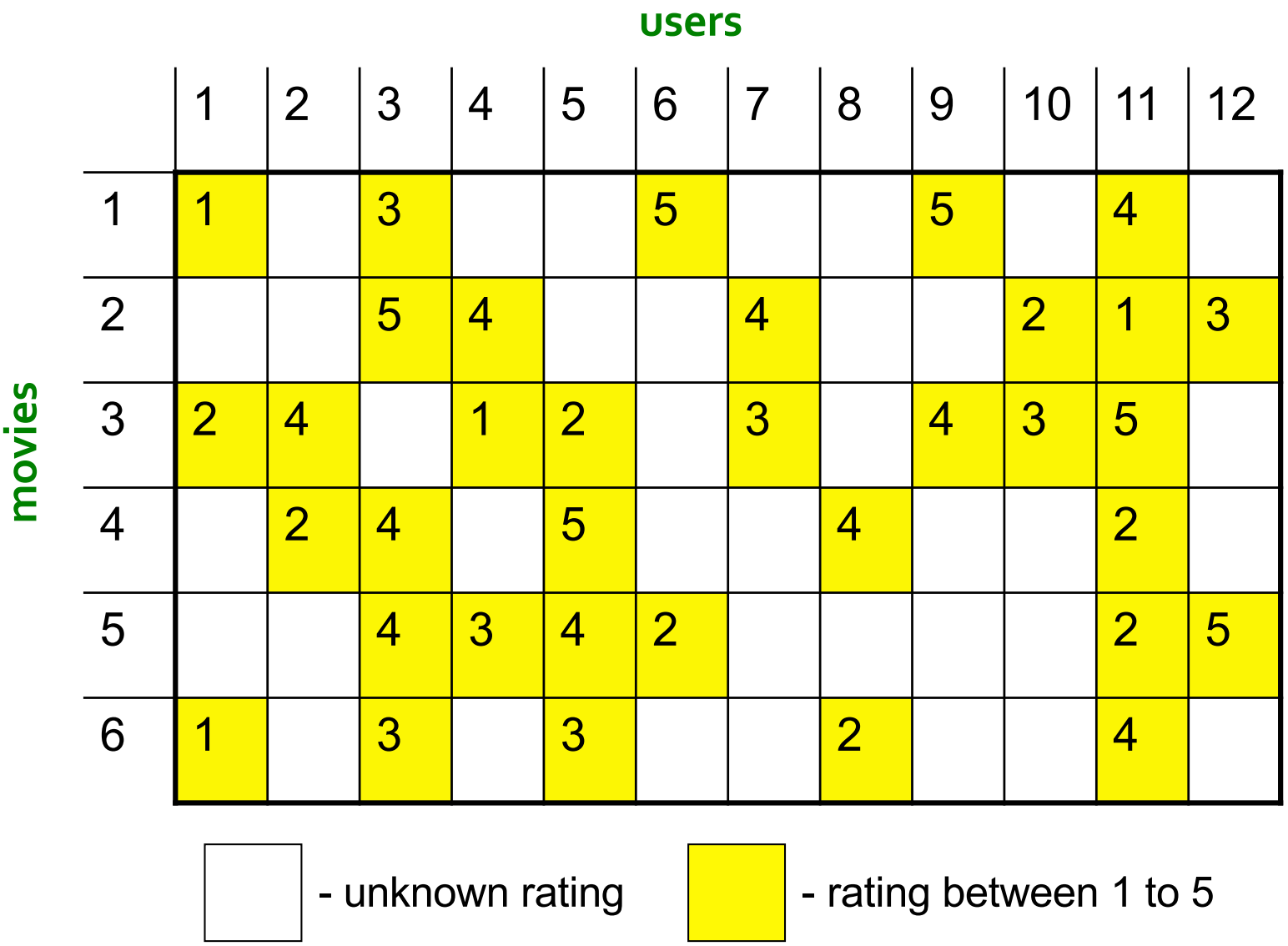
Item-Item Collaborative Filtering

- **So far: User-user collaborative filtering**
- **Another view: Item-item**
  - For item ***i***, find other similar items
  - Estimate rating for item ***i*** (by a user) based on ratings for similar items (by the same user)
  - Can use same similarity metrics and prediction functions as in user-user model

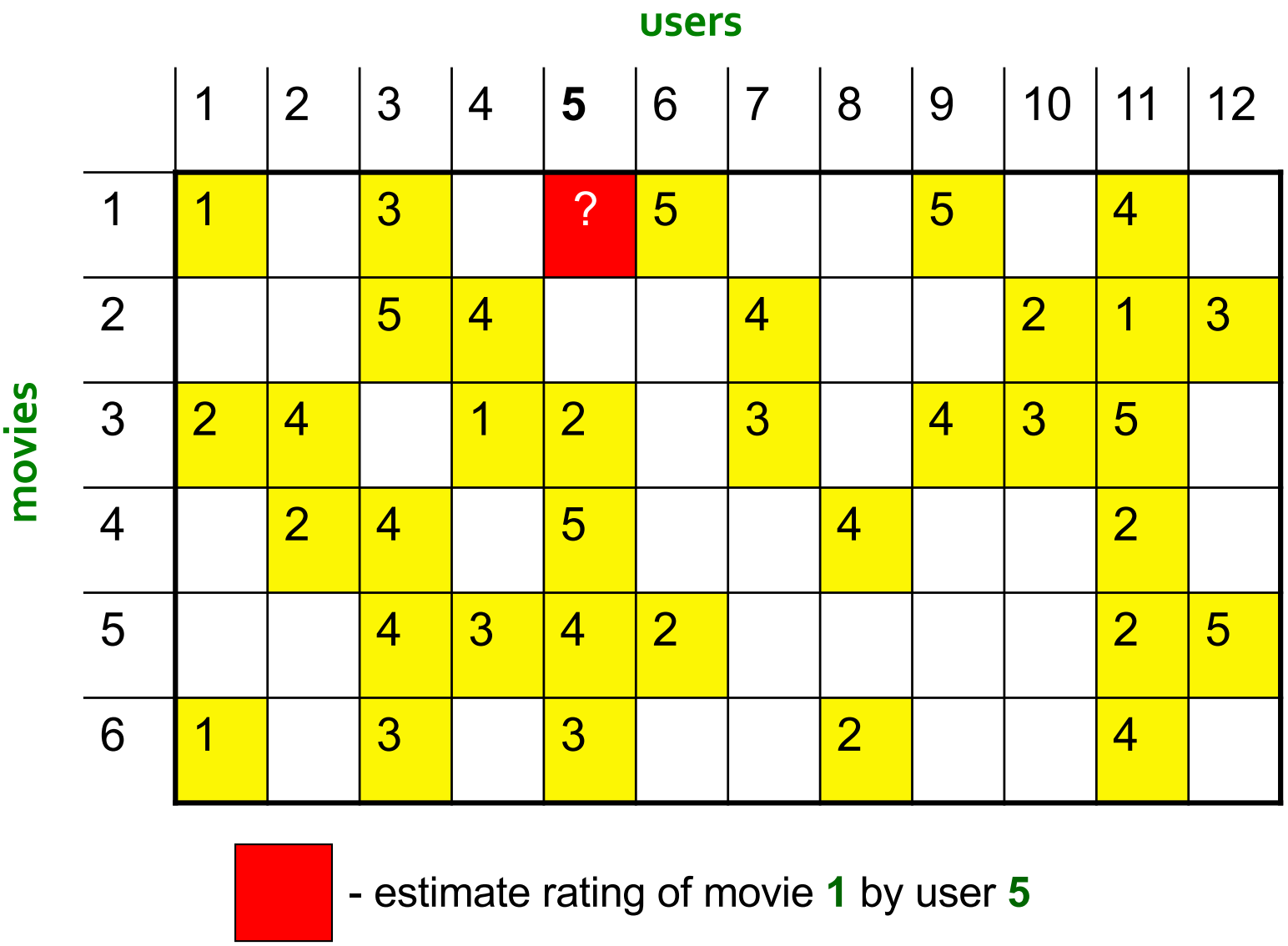
$$r_{xi} = \frac{\sum_{j \in N(i;x)} s_{ij} \cdot r_{xj}}{\sum_{j \in N(i;x)} s_{ij}}$$

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

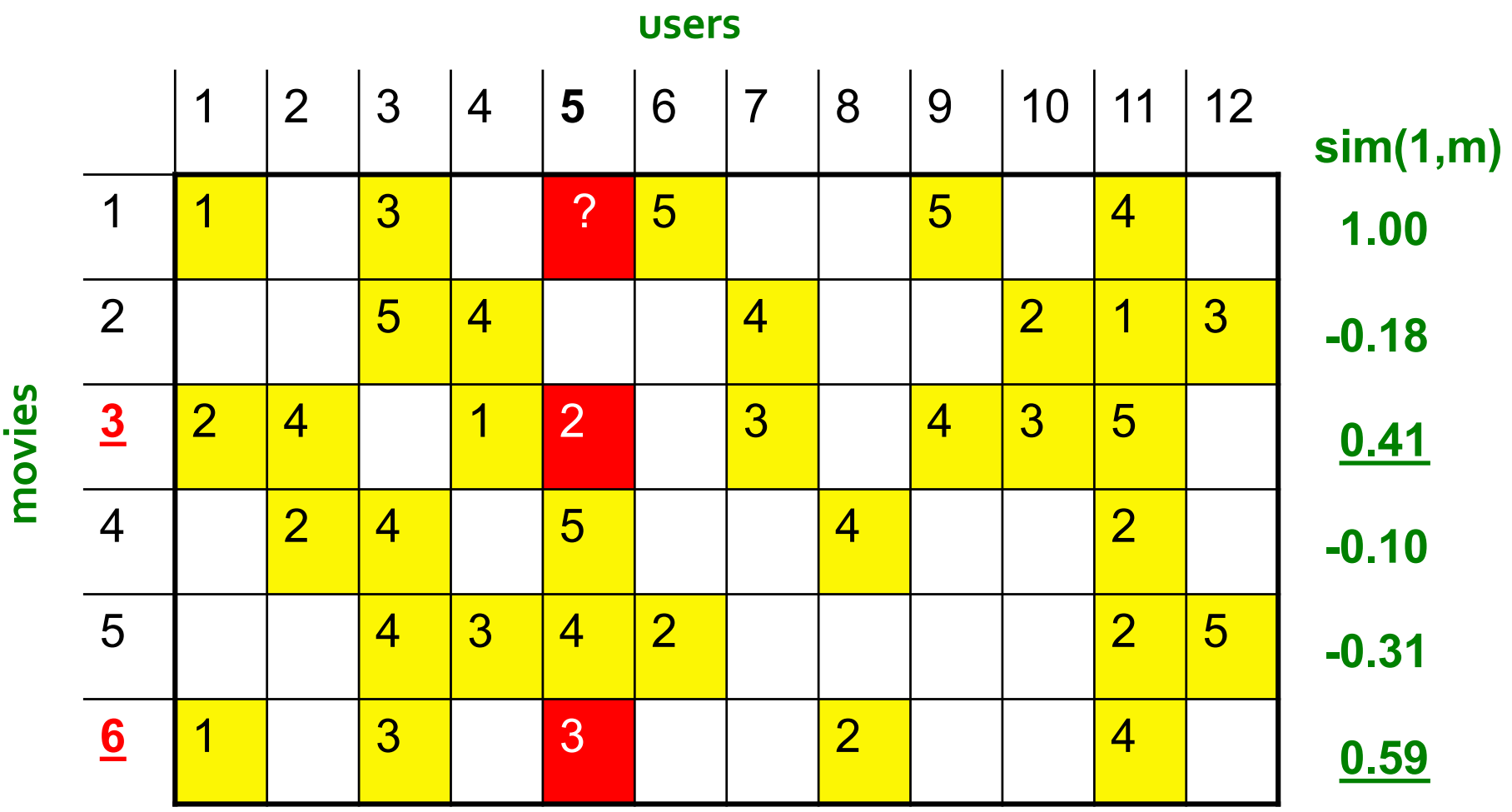
Item-Item CF (|N|=2)



Item-Item CF (|N|=2)



Item-Item CF (|N|=2)



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

Here we use Pearson correlation as similarity:

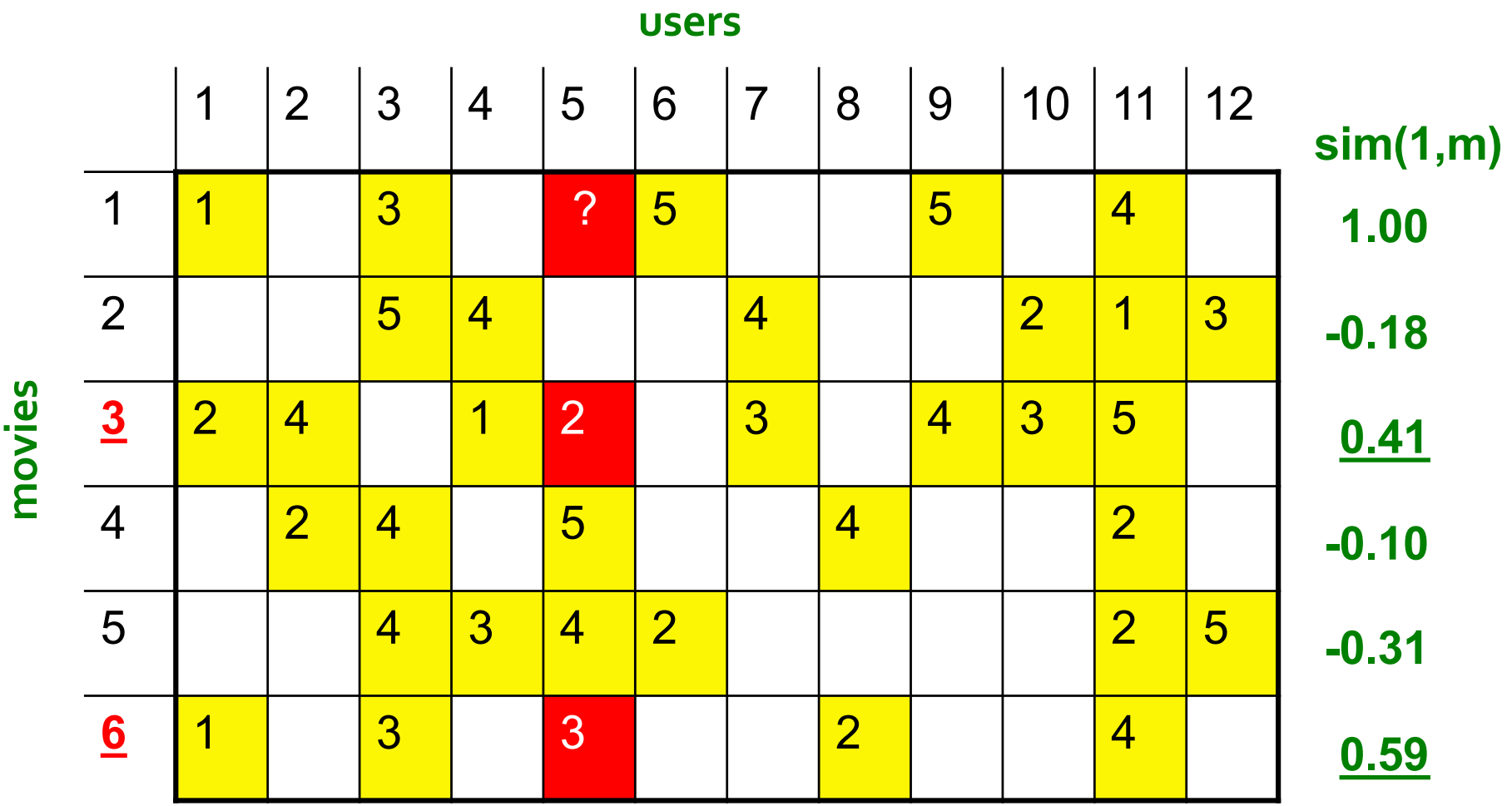
1) Subtract mean rating m_i from each movie ***i***

$m_1 = (1+3+5+5+4)/5 = 3.6$

row 1: *[-2.6, 0, -0.6, 0, 0, 1.4, 0, 0, 1.4, 0, 0.4, 0]*

2) Compute cosine similarities between rows

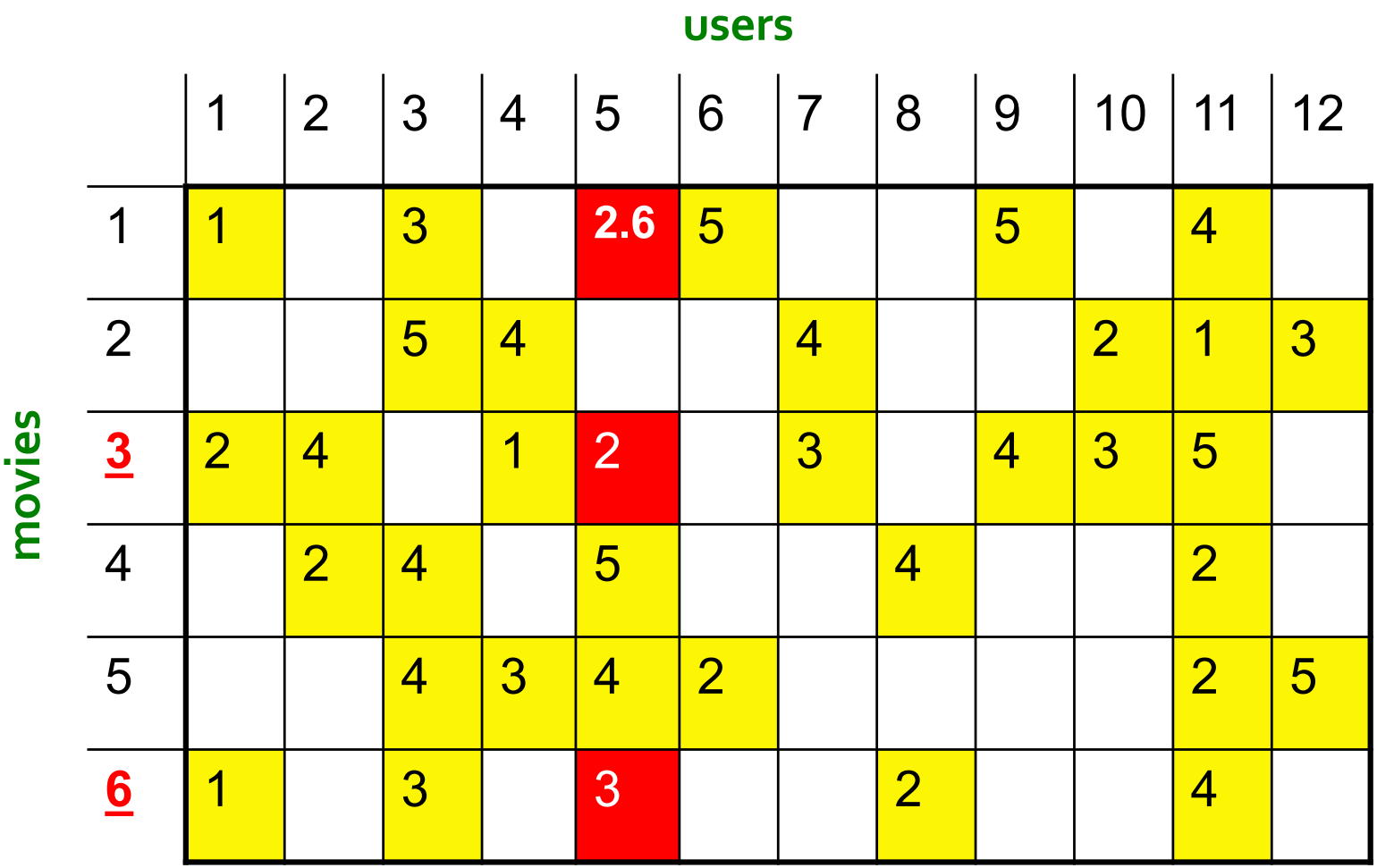
Item-Item CF (|N|=2)



Compute similarity weights:

$s_{1,3}$=0.41, $s_{1,6}$=0.59

Item-Item CF (|N|=2)



Predict by taking weighted average:

$r_{1.5}$ = (0.41*2 + 0.59*3) / (0.41+0.59) = 2.6

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

Item-Item vs. User-User

	Avatar	LOTR	Matrix	Pirates
Alice	1		0.8	
Bob		0.5		0.3
Carol	0.9		1	0.8
David			1	0.4

- **In practice, it has been observed that item-item often works better than user-user**
- **Why?** Items are simpler, users have multiple tastes

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

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

Social Recommendations

Some ideas

Social recommendations

- User-user collaborative filtering relies on identifying set of users 'similar to' target user *x*
- Use *x*'s social contacts (friends) to form this set, e.g., higher weight to users whose ratings are similar as well as are 'near' *x* in the social net
- Intuition – homophily, trust over friends
- Help to generate better explanations for recommendations

Remarks & Practical Tips

- Complexity
- Evaluation

Collaborative Filtering: Complexity

- Expensive step is finding ***k*** most similar users:
  O(|X|) X is the set of users
  - Too expensive to do at runtime
  - Could pre-compute
- Naïve pre-computation takes time **O(k ·|X|)**

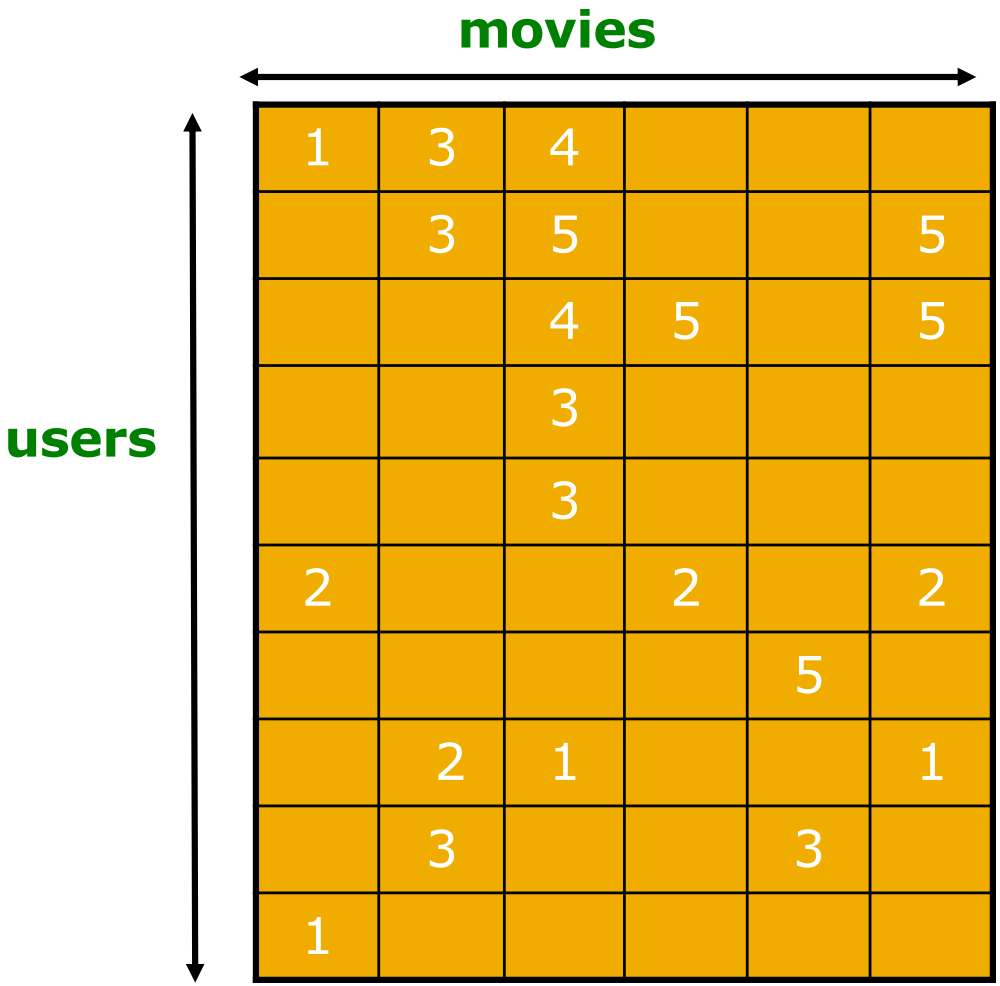
- **Several techniques to do this efficiently**
  - Near-neighbor search in high dimensions (**LSH**)
  - Clustering
  - Dimensionality reduction of matrix

Tip: leverage all data

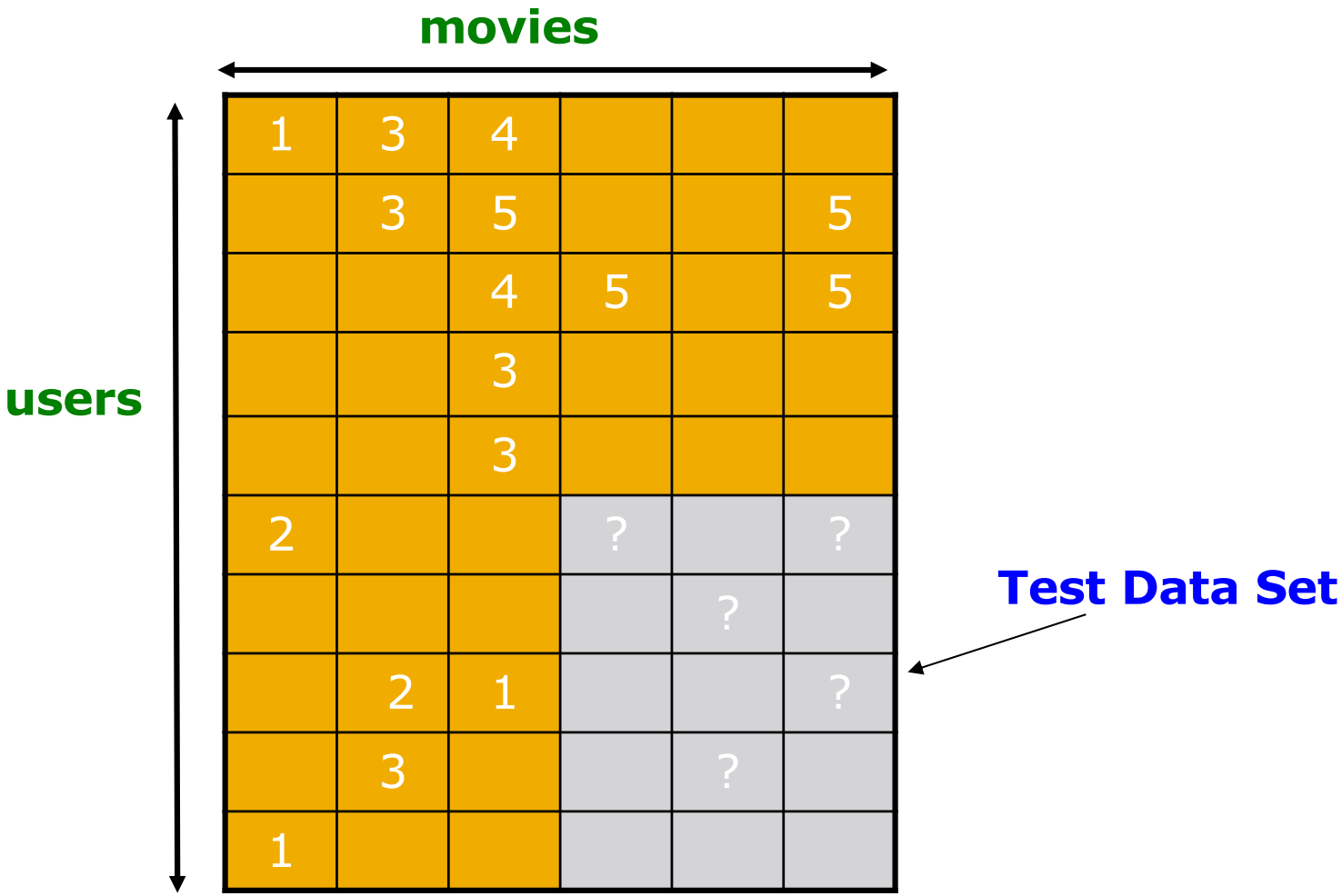
- **Leverage all the data**
  - Don't try to reduce data size in an effort to make fancy algorithms work
  - Simple methods on large data do best
- **Add more data if possible**
  - e.g., add IMDB data on genres for movie recs
- **More data beats better algorithms**

http://anand.typepad.com/datawocky/2008/03/more-data-usual.html

Evaluation of recommendations



Evaluation of recommendations



Evaluation measures

- **Compare predictions with known ratings**
  - **Root-mean-square error** (RMSE)
    - $\sqrt{\sum_{xi}(r_{xi} - r^*_{xi})^2}$ where r_{xi} is predicted, r^*_{xi} is the true rating of x on i
  - **Precision at top 10**:
    - % of top 10 recommendations that are actually liked
  - **Rank Correlation**:
    - Spearman's *correlation* between system's and user's complete rankings
- **Another approach: 0/1 model**
  - **Coverage:**
    - Number of items/users for which system can make predictions
  - **Precision:**
    - Accuracy of predictions

Problems with Eval Measures

- **In practice, we care only to predict high ratings:**
  - RMSE might penalize a method that does well for high ratings and badly for others

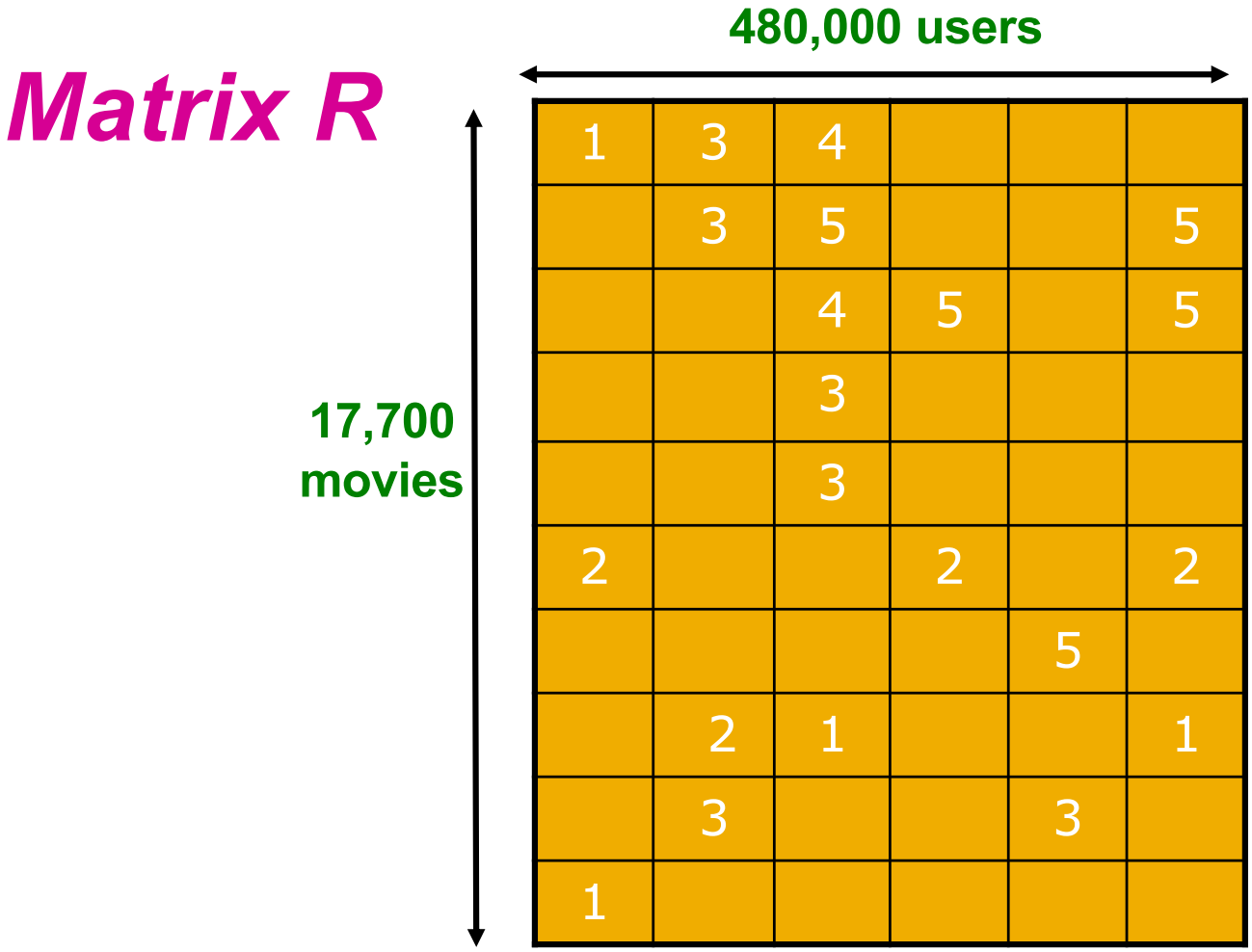
- **Narrow focus on accuracy sometimes misses other desirable properties of recs**
  - Diversity in recommendations
  - Recency / location of recommendations
  - Bias & fairness issues in recommendations

The Netflix Prize

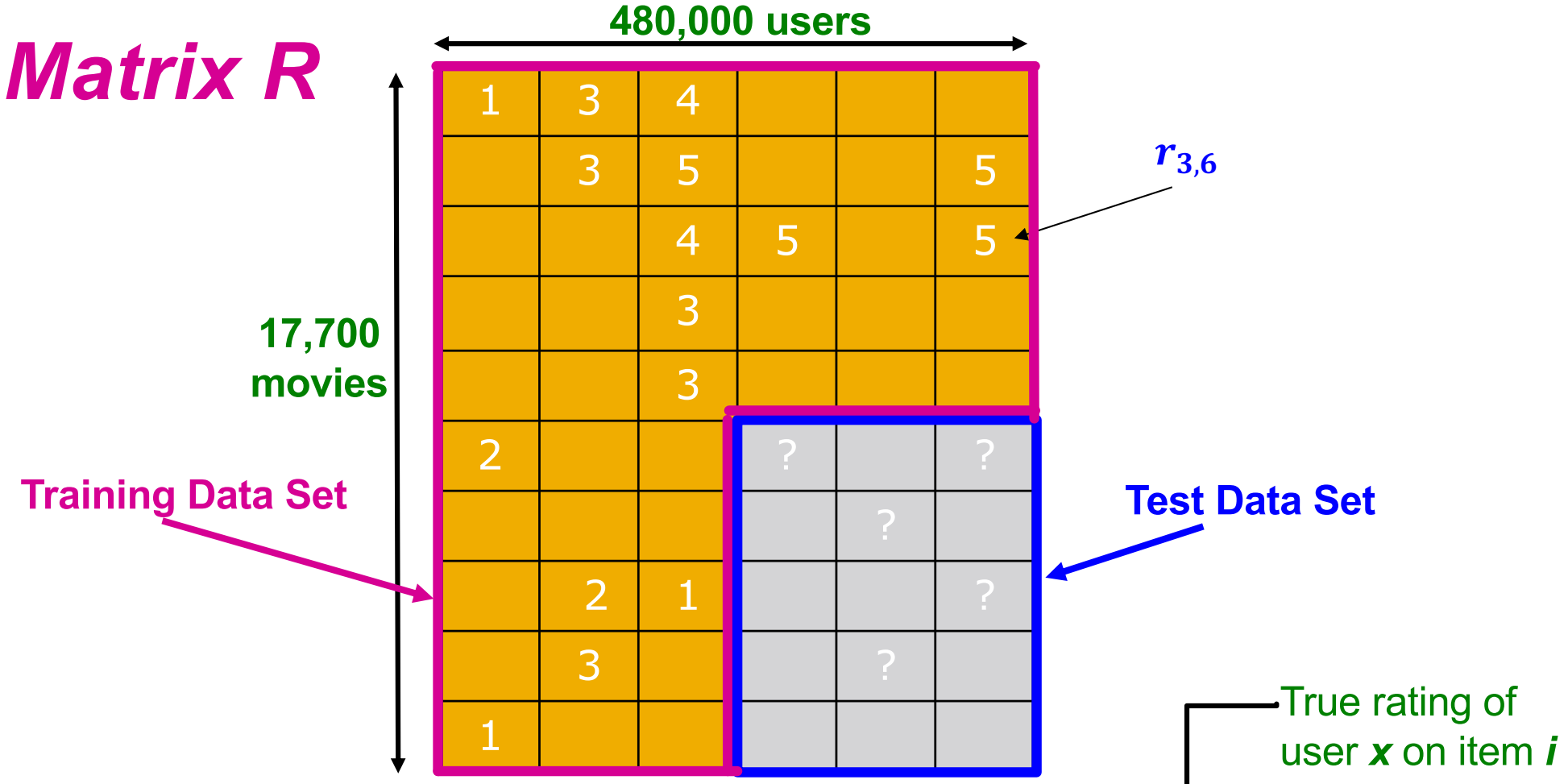
The Netflix Prize

- **Training data**
  - 100 million ratings, 480,000 users, 17,770 movies
  - 6 years of data: 2000-2005
- **Test data**
  - Last few ratings of some users (2.8 million)
  - **Evaluation criterion:** Root Mean Square Error (RMSE) $=$ $\frac{1}{|R|}\sqrt{\sum_{(i,x)\in R}(\hat{r}_{xi}-r_{xi})^2}$
  - **Netflix's system RMSE: 0.95**
- **Competition**
  - 2,700+ teams
  - **$1 million** prize for 10% improvement on Netflix system

The Netflix Utility Matrix *R*



Utility Matrix *R*: Evaluation

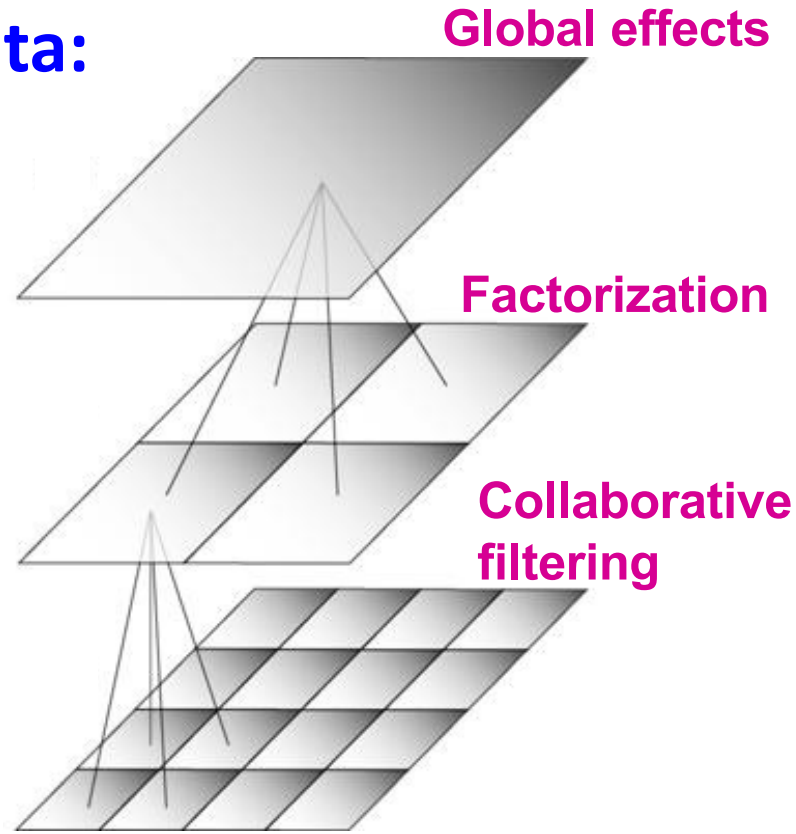


$$\text{RMSE} = \frac{1}{|R|}\sqrt{\sum_{(i,x)\in R}(\hat{r}_{xi} - r_{xi})^2}$$

BellKor Recommender System

- **The winner of the Netflix Challenge (after 3 years) – a combination of several algorithms**

- **Multi-scale modeling of the data:** Combine top level, "regional" modeling of the data, with a refined, local view:
  - **Global:**
    - Overall deviations of users/movies
  - **Factorization:**
    - Addressing "regional" effects
  - **Collaborative filtering:**
    - Extract local patterns



Modeling Local & Global Effects

- **Global:**
  - Mean movie rating: **3.7 stars**
  - *The Sixth Sense* is **0.5** stars above avg.
  - Joe rates **0.2** stars below avg.

    ⇒ Baseline estimation:

    ***Joe* will rate *The Sixth Sense* 4 stars**
- **Local neighborhood (CF/NN):**
  - *Joe* didn't like related movie *Signs*
  - **⇒ Final estimate:**

    ***Joe* will rate *The Sixth Sense* 3.8 stars**