COMP30120

Recommender Systems Collaborative Filtering Part 1

Derek Greene

School of Computer Science and Informatics

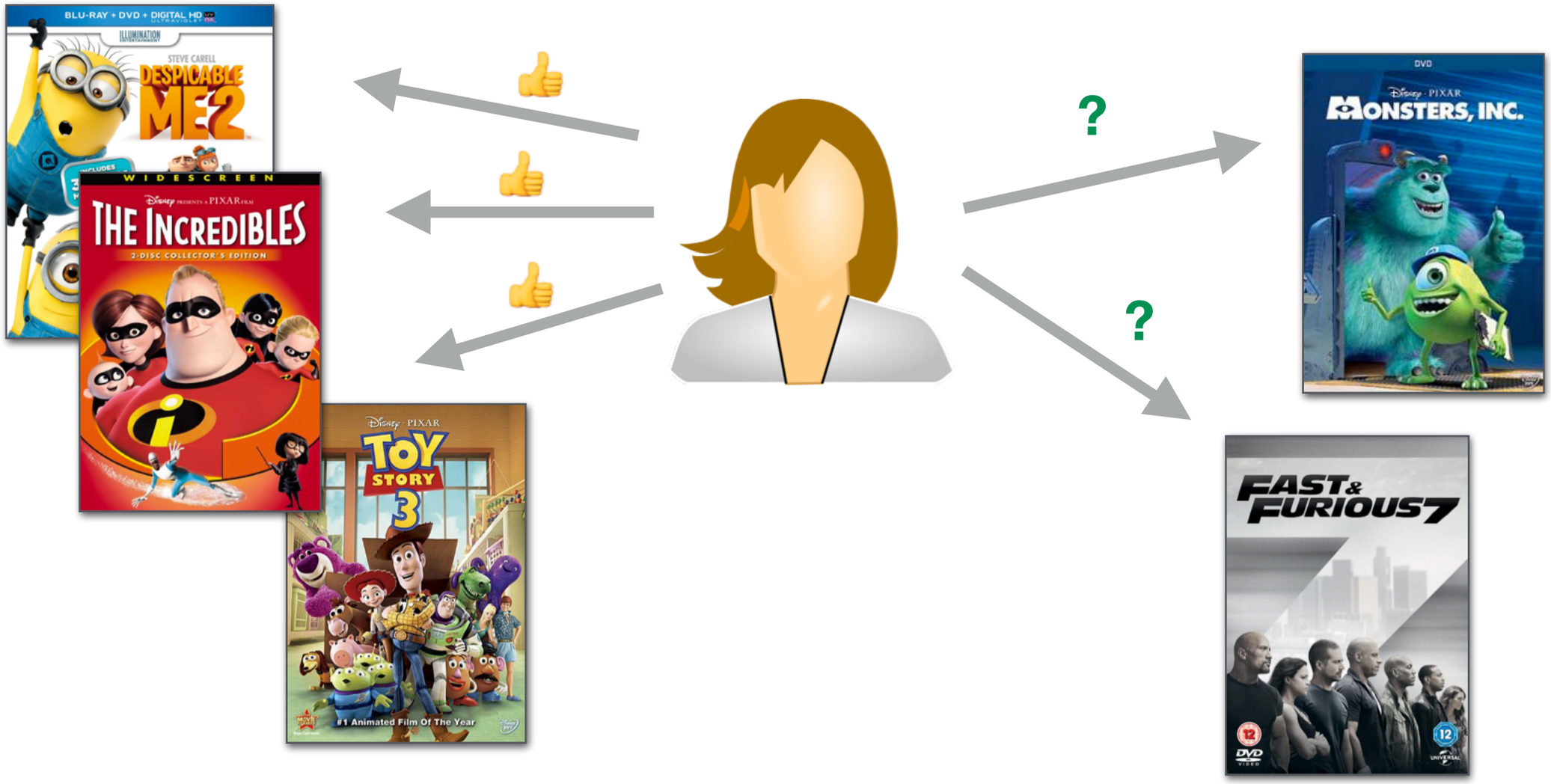
Autumn 2015

Overview

- Recommender Systems
  - Collaborative Filtering (CF)
  - “People who bought this also bought…”
  - Industry examples
- Content-based Recommendation
- User-based Collaborative Filtering
- Item-based Collaborative Filtering

Recommender Systems

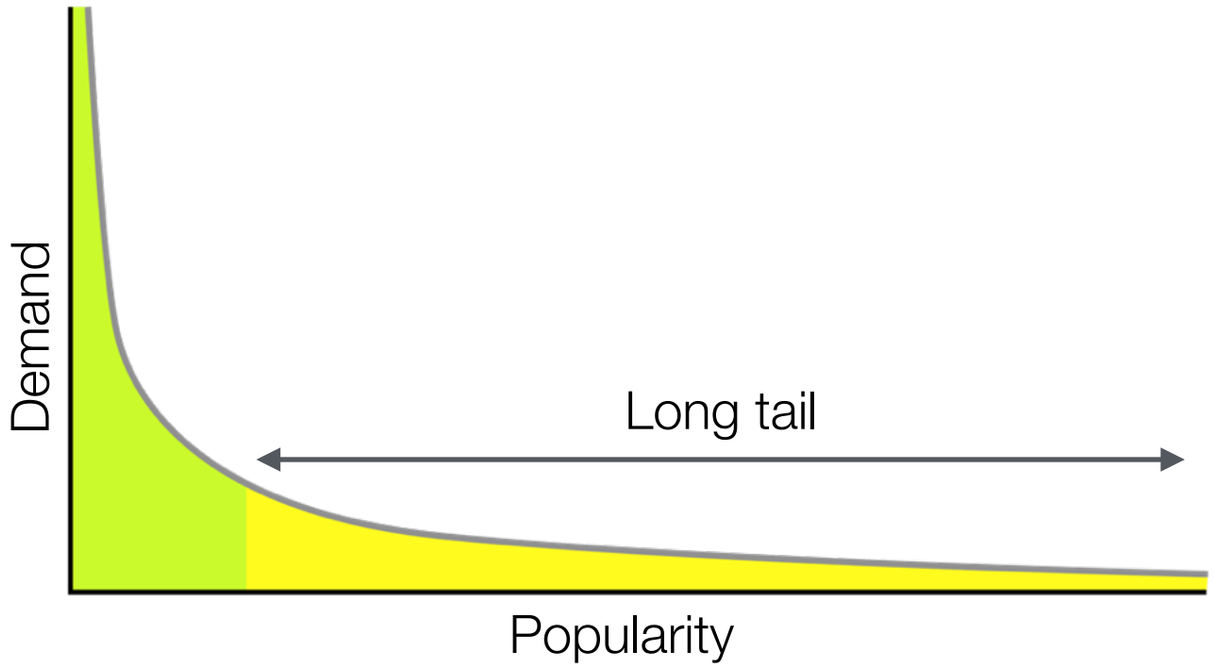
"Recommender Systems are software agents that elicit the interests and preferences of individual consumers and make recommendations accordingly… They have the potential to support and improve the quality of the decisions consumers make while searching for and selecting products online." - Xiao & Benbasat (2007)



The Long Tail

- Bricks-and-mortar stores can offer only a fixed number of items.

➡ Customers are only presented with a limited choice of popular items.



- Online stores can potentially offer millions of books, songs or movies, not just popular items.

http://en.wikipedia.org/wiki/Long_tail

➡ Users cannot be presented with all items, so recommenders that are personalised to individuals can guide their choices.

Collaborative Filtering

- **CF:** Approach for making predictions about the preferences of a user by collecting information from many other users.
- Discover patterns in observed preference behaviour across a large community of users.
  - Product purchase history
  - Item ratings for movies, books, songs etc.
  - Browsing history

➡ Predict new preferences based on those patterns alone.

- Does not rely on any “content” data, such as metadata about the items (e.g. product descriptions) or the information users (e.g. demographics).

Collaborative Filtering

The CF task typically has two forms…

1. Prediction

Ratings by Alice

★★★★★ ★★★★ ★★★

Ratings by Bob

★★★★★ ★★★★ ?????

2. Recommendation

Bob likes…

Alice likes…

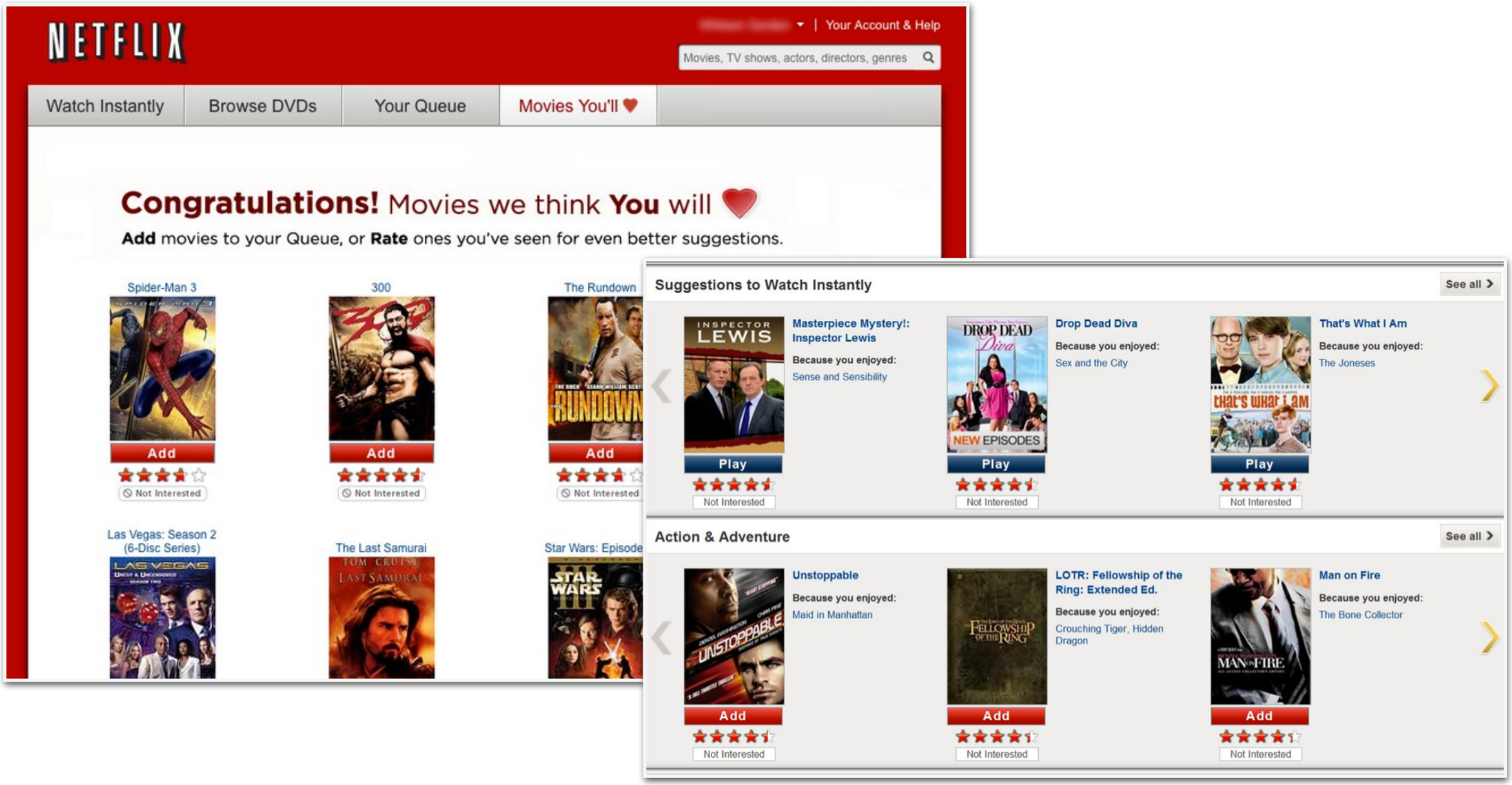
Bob may also like…

Industry Applications

Key technology for Amazon, where product recommendations are made based on purchase history and browsing history.

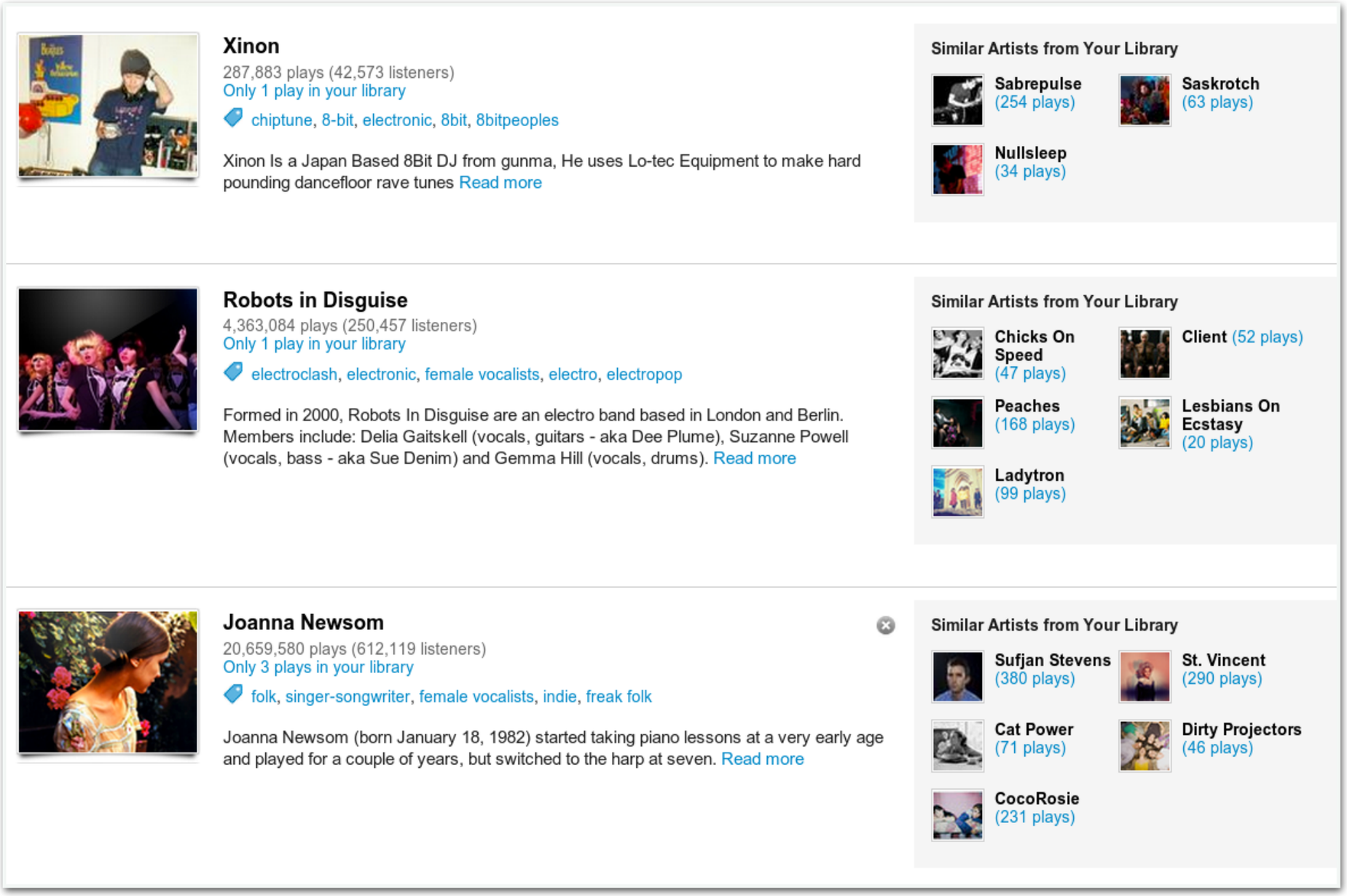
Industry Applications

Netflix provides personalised recommendations for movies you might like, based on previous ratings.



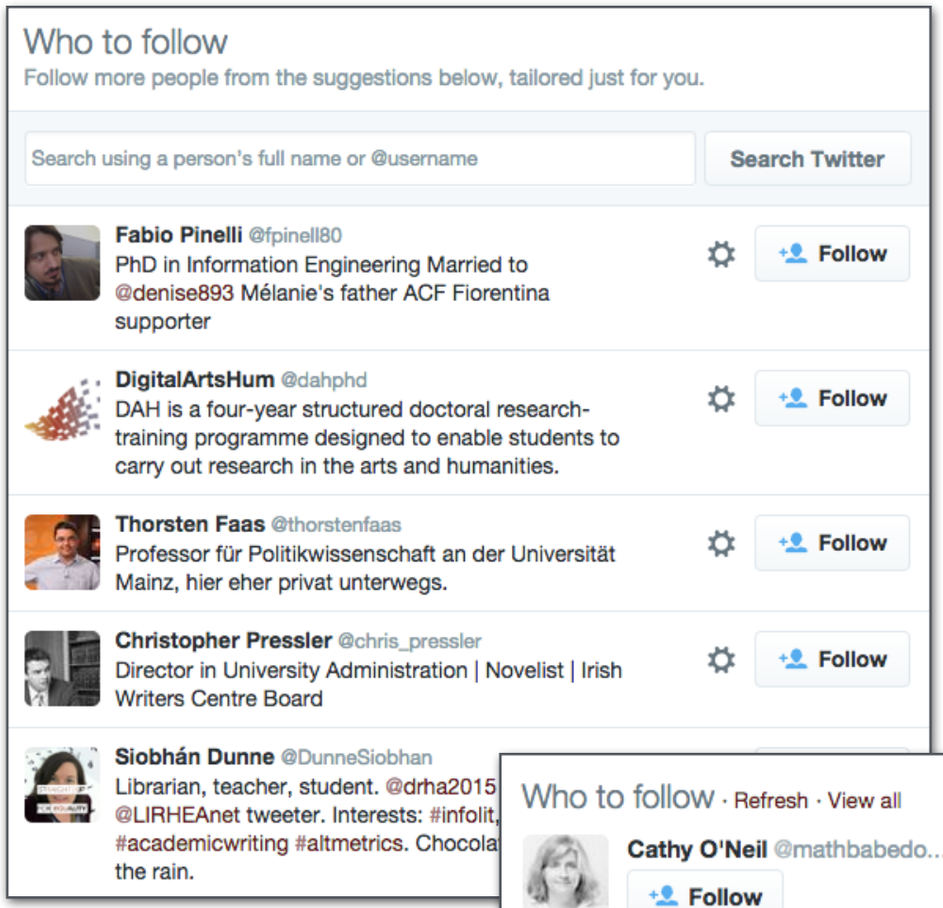
Industry Applications

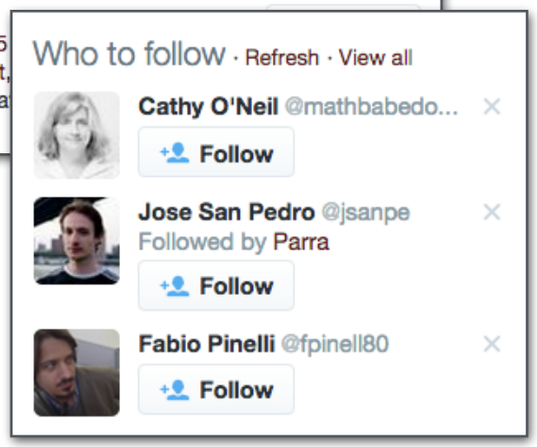
Last.fm and Spotify use play counts from a user's music history to recommend new artists and songs to play.

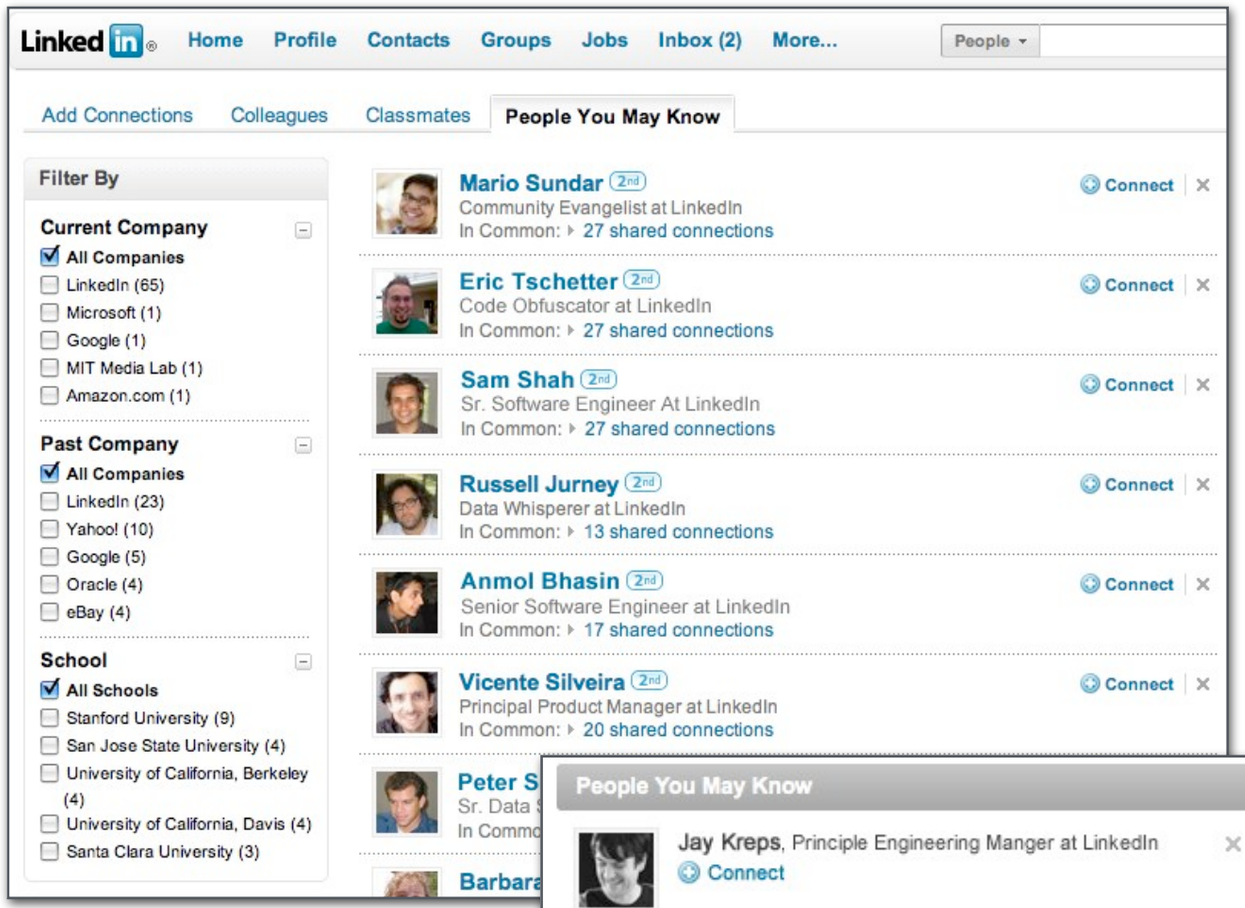


Industry Applications

Twitter, Facebook, and LinkedIn recommend other users to follow/ friend based on existing network connections and interests.







Challenges

Although recommender systems have been in use for some time, they still face challenges in real-world applications:

- Scalability: Large retailers will often have tens of millions of items in their product catalogue.
- Results must be returned in real-time.
- New customers will have very limited information. How do we recommend for them?
- A glut of information will exist for older customers. Is this information still relevant?
- Customer data is volatile and constantly changing.
- How do we deal with groups of users using a recommender (e.g. different members of a family)?
- How should users interact with the recommender?

Collaborative Filtering

- Collaborative filtering is a “representation-less” approach, since no descriptions are available of the items are involved.
- Data will often simply consist of ratings (e.g. 1-5 ★) that have been normalised in some way (e.g. range normalised to 0-1).
- Or in some cases we have simply binary values - e.g. 1 = user bought the item, 0 = user did not buy the item.

	Song 1	Song 2	Song 3	Song 4	Song 5	Song 6	Song 7	Song 8	Song 9	Song 10	Song 11	Song 12	Song 13	Song 14	Song 15
User 1	0.8	1.0						0.8							
User 2			0.6										0.2		
User 3		0.2													
User 4							1.0								
User 5			0.6							0.2					
User 6					0.6	0.6									
User 7				1.0											
User 8							1.0								
User 9						0.2			0.2						
User 10															

	Book 1	Book 2	Book 3	Book 4	Book 5	Book 6	Book 7	Book 8	Book 9	Book 10
User 1				1	1					
User 2	1		1					1		
User 3		1			1		1			
User 4					1				1	
User 5	1	1					1			
User 6			1			1				
User 7									1	
User 8										

Content-Based Recommendation

- Content-based recommenders analyse item descriptions and metadata to identify items that are of interest to the target user.
- Exact features depend on the nature of the data. Can be hard to choose appropriate features.
- Match users to items based on a similarity measure that is a weighted function of the similarity between these features.

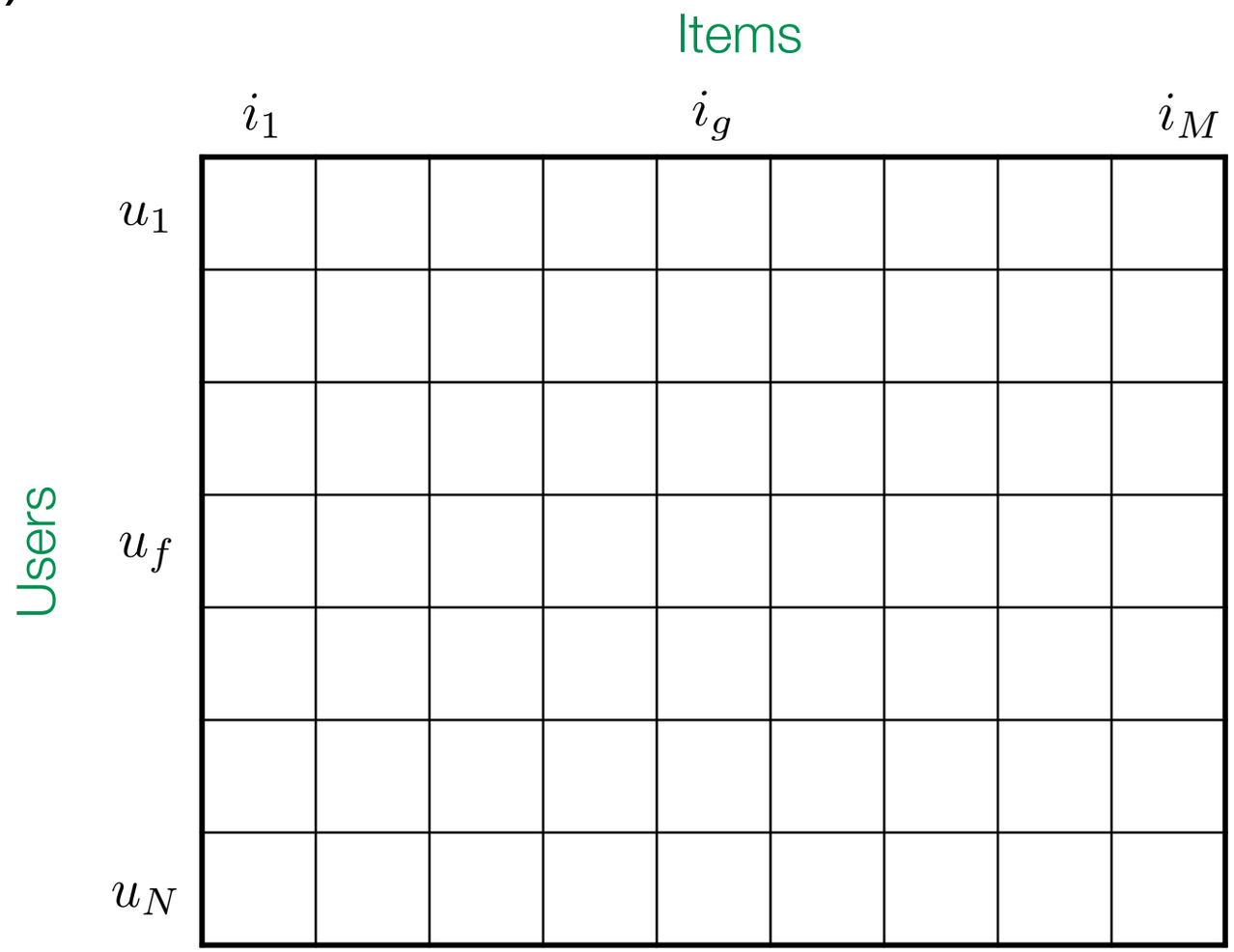
Similarity

➡ Data format for CF is extremely different. Representation-less approach makes it largely “domain agnostic”.

Collaborative Filtering

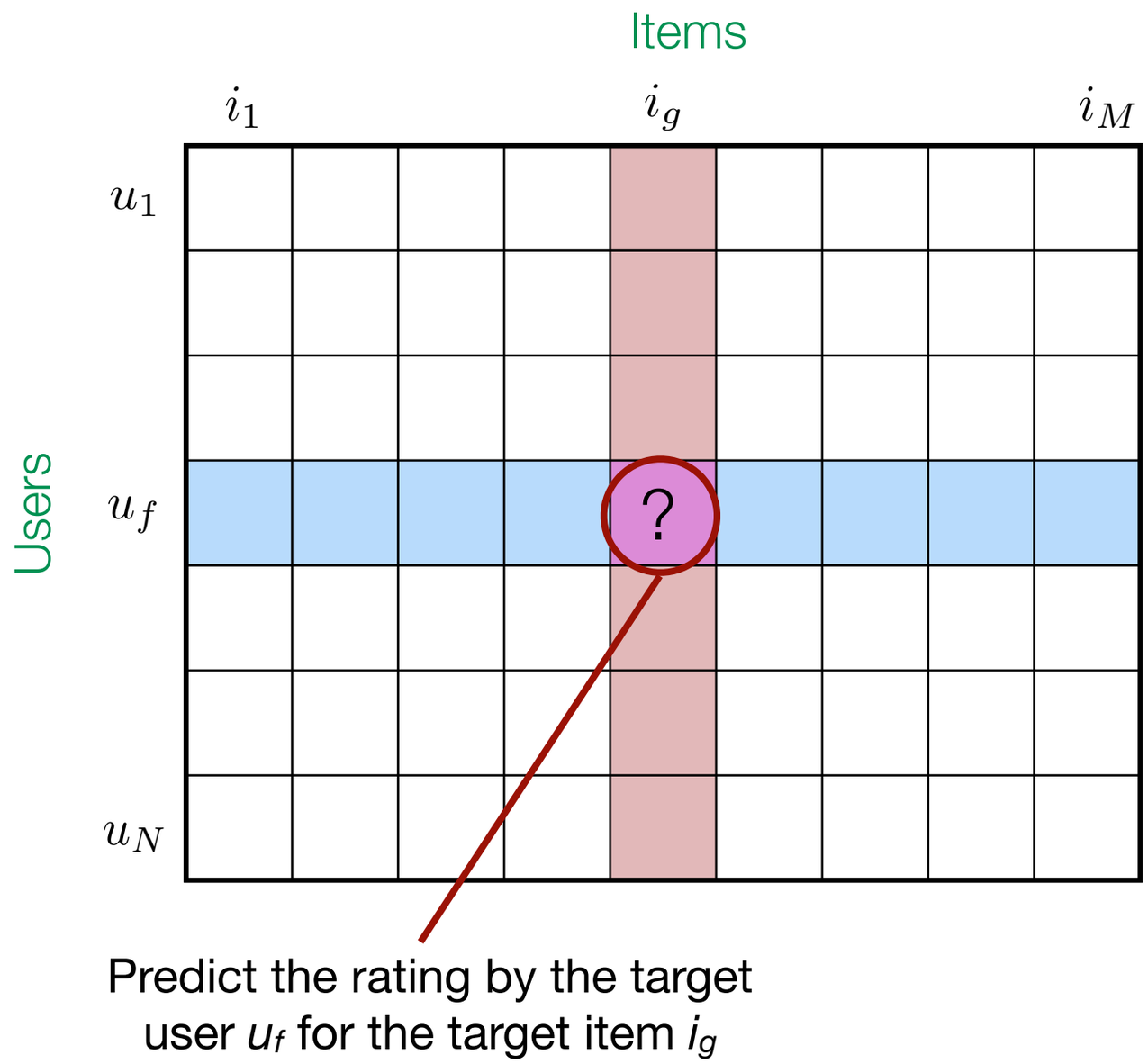
- **Data Format:** A set of *N* users and a catalogue of *M* items.
- We can represent each user as a *M*-dimensional vector.
- Non-zero values represent previous activity for a given item by that user (e.g. a rating for the item).

- Build a *NxM* user-item matrix (utility matrix).
- Rows are user profiles.
- Columns are item profiles.
- Matrix is highly sparse. Most users have rated very few items; most items have very few ratings.



CF Prediction Task

- **Goal:** Given the user-item matrix, make a prediction for a target user u_f against a target item i_g for which a rating does not exist.
- i.e. fill in the empty value in row *f* and column *g* in the matrix.
- The observed (non-empty) values provide us with our training data.



- Standard k-fold cross-validation evaluation approaches can be used to measure the accuracy of the predictions.

Data Normalisation

- Usually normalise or centre the user-item matrix before applying algorithms to remove bias.
- Where does the bias come from?
  - Some users give systematically higher ratings.
  - Some items receive systematically higher ratings.
- How is normalisation performed?
- Subtract a bias term: $\tilde{r}_{fg} = r_{fg} - b_{fg}$
- Different bias terms:
  - *Global mean rating*: average over all ratings in the data.
  - *Item mean rating*: average over all ratings for specific item.
  - *User mean rating*: average over all rating for specific user.

Collaborative Filtering Strategies

Two fundamental strategies in Collaborative Filtering:

1. User-based CF
   - Intuition: Personal tastes are correlated.
   - Find other users who share the same tastes as the target user, and use their information to make predictions.
2. Item-based CF
   - Match items previously purchased or rated by a target user to similar items.
   - Combine those similar items to make predictions.

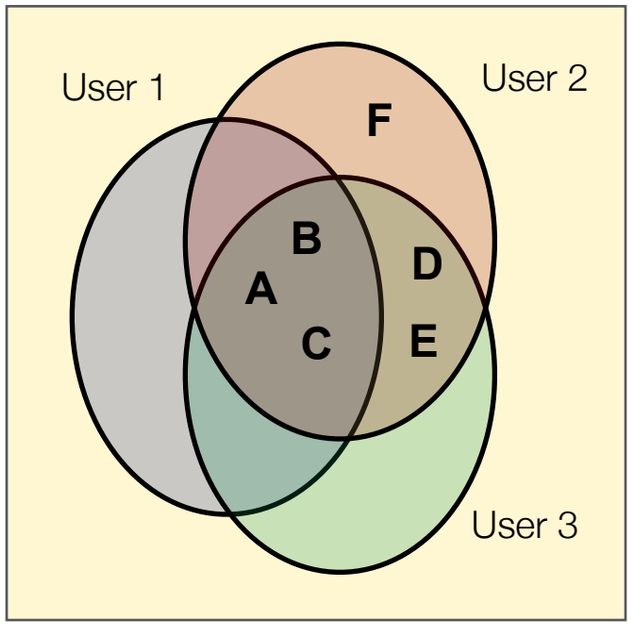
User-based Collaborative Filtering

- **Task:** Look for other users who share the same behaviour as the target user, and use their information to make predictions.

Bob likes… Alice likes… Bob may also like…

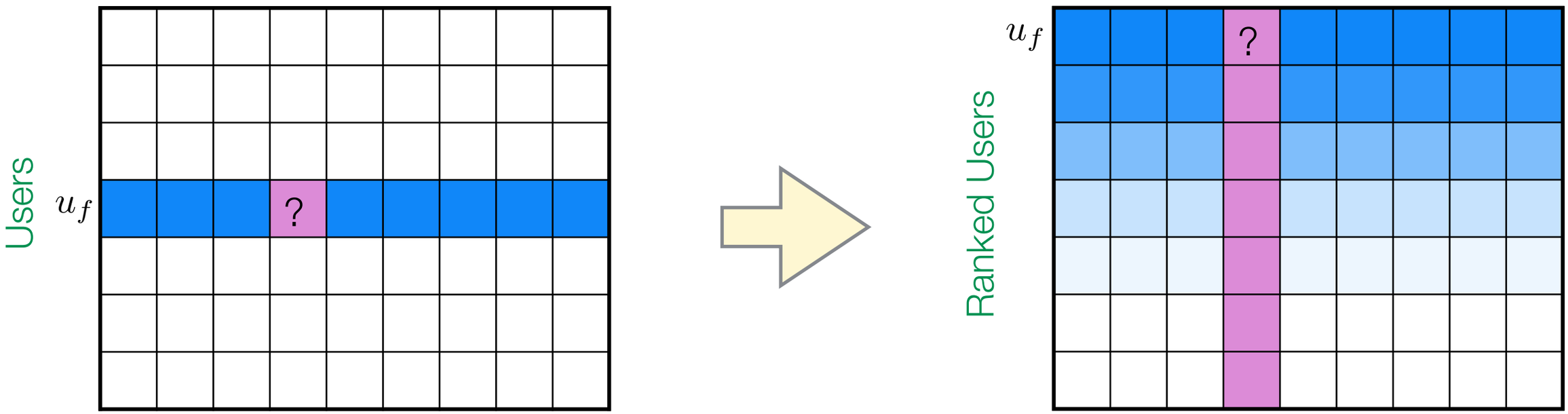
- **Example:** Group of users {1,2,3} who have have all purchased the items {A,B,C}. What should we recommend to target user 1?

➡ Items D and E can be recommended to User 1 based on their shared interests.



User-based Collaborative Filtering

- Predict a target user's interest in a target item based on ratings coming from other similar users.
- Each user profile (row) in the user-item matrix is sorted by its similarity with respect to the target user's profile.
- Ratings by more similar users should contribute more to predicting target item rating.



Original User-Item matrix

Predict based on similar users

Measuring Similarity

- When working with binary data (e.g. purchase history), can use set intersection measures to score the similarity between two users.

	Book 1	Book 2	Book 3	Book 4	Book 5	Book 6	Book 7	Book 8	Book 9	Book 10
User *p*				1	1			1		
User *q*					1				1	1

Jaccard Index

$$sim(p,q) = \frac{|B_p \cap B_q|}{|B_p \cup B_q|}$$

$$B_p = \text{Books purchased by } p$$
$$B_q = \text{Books purchased by } q$$

User *p* purchased {4,5,8}
User *q* purchased {5,9,10}

Intersection = {5}
Union = {4,5,8,9,10}

$$sim(p,q) = \frac{1}{5}$$

- Suitable for binary data. But when values consist of more detailed ratings (e.g. 1-5 ★), the Jaccard Index loses important information.

Measuring Similarity

- When working with real-valued ratings data, a common approach to scoring the similarity between two users is to measure the cosine similarity between their profile vectors:

	Book 1	Book 2	Book 3	Book 4	Book 5	Book 6	Book 7	Book 8	Book 9	Book 10
User *p*				1	1			1		
User *q*					1				1	1

$$\cos(\mathbf{p}, \mathbf{q}) = \frac{\mathbf{p} \cdot \mathbf{q}}{||\mathbf{p}|| \, ||\mathbf{q}||}$$

Binary matrix:
cos(p,q) = 0.333

	Movie 1	Movie 2	Movie 3	Movie 4	Movie 5	Movie 6	Movie 7	Movie 8	Movie 9	Movie 10
User *p*	0.4		0.2					1.0		0.6
User *q*	0.4				0.4			0.8		

Real-valued matrix:
cos(p,q) = 0.784

- Cosine treats empty values as 0s. Has effect of treating lack of a rating as more similar to disliking an item than liking it.
- Cosine similarity can be adjusted to remove this (i.e. Pearson correlation)

Predicting Ratings

Predictions in User-based CF:

1. Measure the similarity between the target user and all other users.
2. Rank the users and find the top *k* most similar users (nearest neighbours).
3. Aggregate the profiles of the neighbours in some way to get a predicting rating for the item of interest.
   - *Simple average*: average rating over the top *k* users.
   - *Weighted average*: average rating over the top *k* users, where more similar users contribute more.

	Movie 1	Movie 2	Movie 3	Movie 4	Movie 5
Target User		**???**	0.2		0.6
Neighbour *1*		0.6	0.2		0.6
Neighbour *2*	1.0		0.6		0.2
Neighbour *3*		1.0			1.0

	Movie 1	Movie 2	Movie 3	Movie 4	Movie 5
Aggregated	1.0	0.8	0.4		0.6

Aggregated prediction for Target User's rating for Movie 2 using simple average = 0.8
(Only counting non-empty)

Making Recommendations

Recommendation in User-based CF:

1. Measure the similarity between the target user and all other users.
2. Rank the users and find the top *k* most similar users (nearest neighbours).
3. Aggregate the profiles of the neighbours to get the top *n* recommended items.

	Book 1	Book 2	Book 3	Book 4	Book 5	Book 6	Book 7	Book 8	Book 9	Book 10
Target User			1	1	1			1	1	
Neighbour *1*				1	1		1	1	1	
Neighbour *2*	1			1			1	1		
Neighbour *3*		1		1		1		1		1

Profiles for *k=3* nearest neighbours of target user

	Book 1	Book 2	Book 3	Book 4	Book 5	Book 6	Book 7	Book 8	Book 9	Book 10
Aggregated	1	1	0	3	1	1	2	3	1	1

Aggregate these neighbour profiles by frequency

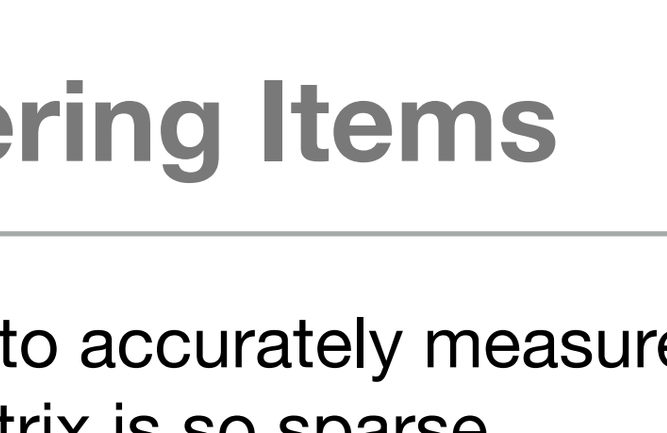
Top *n=3* ranked items:

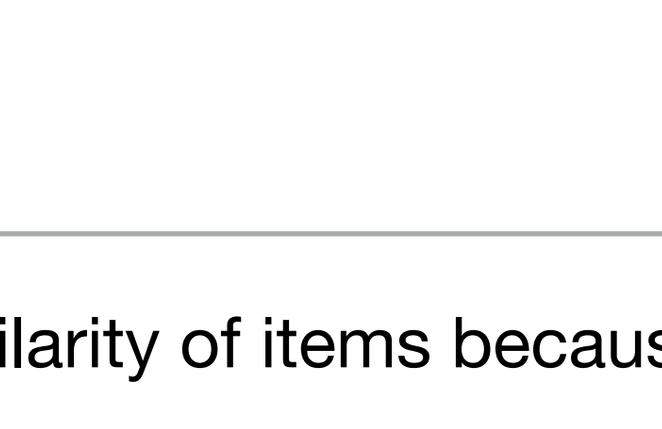
1. Book 8
2. Book 4
3. Book 7

Clustering Items

- Difficult to accurately measure the similarity of items because the user-item matrix is so sparse.
- **Solution:** Group similar items together, then aggregate their ratings to produce more dense data.
- **Approach:** Apply a standard algorithm to cluster the item vectors e.g. *k*-means, hierarchical agglomerative clustering.

Cluster 1





Cluster 2

Clustering Items

- After clustering, we create a new matrix.
- Each column represents a cluster of items.
- Values are the average of a user's rating for the items in each cluster.

	HP6	HP7	HP8	SW4	SW5	SW6
U 1	0.8	1.0			0.6	
U 2			0.6			
U 3		0.2		0.6		
U 4				1.0	1.0	1.0
U 5			0.6			0.4
U 6						0.6

User-Item matrix

	Cluster 1	Cluster 2
U 1	0.9	0.6
U 2	0.6	
U 3	0.2	0.6
U 4		1.0
U 5	0.6	0.4
U 6		0.6

User-Cluster matrix

Matrix sparsity reduced from 69% to 25%

Item-based Collaborative Filtering

- An alternative view: focus on item-item pairs, not users.
- **Predictions in Item-based CF:**
  1. Identify a short list of top alternative items which are similar to the target item (using same similarity measures as before).
  2. Predict rating as the weighted average of the ratings for the alternative items, where weights are based on similarity values.

$$\hat{r}_{fg} = \frac{\sum_{h \in N_g} sim(g,h) \times r_{fh}}{\sum_{h \in N_g} sim(g,h)}$$

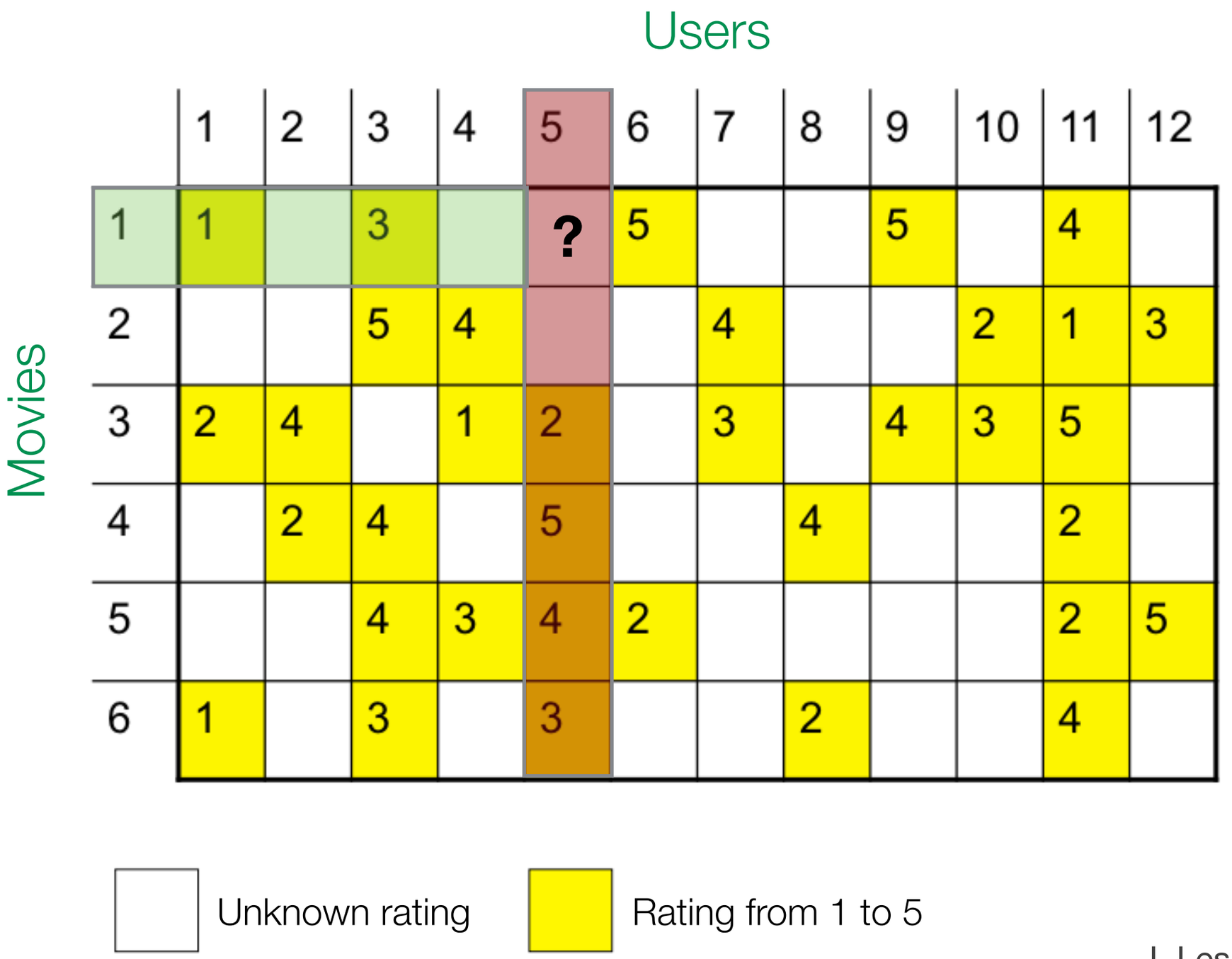
$\hat{r}_{fg}$ = predicted rating for item g by user f

r_{fh} = actual rating for item h by user f

$sim(g,h)$ = similarity between items g and h

$N(g)$ = set of items similar to item g

Example: Item-based CF



Movie-User Matrix

Task:
Predict rating of Movie 1 by User 5

J. Leskovec, A. Rajaraman, J. Ullman.
Mining of Massive Datasets

Example: Item-based CF

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	-0.18
3	2	4		1	2		3		4	3	5		0.41
4		2	4		5			4			2		-0.10
5			4	3	4	2					2	5	-0.31
6	1		3		3			2			4		0.59

Select *k=2* neighbours:
Movie 3 & Movie 6

J. Leskovec, A. Rajaraman, J. Ullman.
Mining of Massive Datasets

Example: Item-based CF

	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	

sim(1,m)
1.00
-0.18
0.41
-0.10
-0.31
0.59

Weights for neighbours:
Movie 3 = 0.41
Movie 6 = 0.59

Predict rating of Movie 1 by User 5
using weighted average:

(0.41*2 + 0.59*3) / (0.41+0.59) = 2.6

J. Leskovec, A. Rajaraman, J. Ullman.
Mining of Massive Datasets

Summary

- Recommender Systems
  - Collaborative Filtering (CF)
  - “People who bought this also bought…”
  - Industry examples
- Content-based Recommendation
- User-based Collaborative Filtering
- Clustering Items
- Item-based Collaborative Filtering

References

- Xiao, Bo, and Izak Benbasat. "E-commerce product recommendation agents: Use, characteristics, and impact." Mis Quarterly 31.1 (2007): 137-209.
- J. Leskovec, A. Rajaraman, J. Ullman “Mining of Massive Datasets”. Cambridge Press. http://www.mmds.org
- J Wang, AP De Vries, MJT Reinders. "Unifying user-based and item-based collaborative filtering approaches by similarity fusion". Proc. SIGIR 2006.
- B. Sarwar, G. Karypis, J. Konstan, J. Riedl. “Item-based Collaborative Filtering Recommendation Algorithms”, Proc. WWW 2010.

COMP30120 - Theory Exam

- **Tuesday 24th November 9am B004, arrive early.**
- 35% of overall module mark
- 45 minutes
- Choice: 8 questions, answer any 6
- All questions carry equal marks
- Written exam, use exam booklets
- Printed course notes and personal notes can be used
- **No use of laptop, phones, or tablets etc**
- **Scientific calculator required for exam, not phone**