

2. Recommender Systems

Recommenders Everywhere

Recommenders Everywhere

The image is a collage of various recommendation systems. It includes:

- Amazon:** A screenshot of the Amazon website showing a book recommendation for "Java 8 in Action: Lambdas, Streams, and Functional-style Programming" by Raoul-Gabriel Urma and Mario Fusco. The price is \$39.86.
- Google:** A screenshot of the Google News page showing a technology article titled "Google Nexus 9 review: A powerful tablet m...".
- iTunes:** A screenshot of the iTunes "Recommendations for You" section, showing music recommendations based on purchases. It lists albums like "Primordial Audio Chronicle - Single" by Dorje and "Beyond Magnetic - EP" by Th1rt3en.
- Facebook:** A screenshot of a Facebook news feed showing various posts and suggestions, including "N24", "The LAD Bible", "POLITIK", and "Metal Hammer".
- ScienceDirect:** A screenshot of a ScienceDirect article titled "Computer Speech & Language", Volume 29, Issue 1, January 2015, Pages 20-31. The article is "Unsupervised language model adaptation using LDA-based mixture models and latent semantic marginals" by Md. Akmal Haidar and Douglas O'Shaughnessy.
- Other:** A screenshot of a music player interface showing recommendations for Philip Sayce, Bluebird, Paulo Mendonca, and Linkin Park.

Outline

2.1. What are Recommender Systems?

2.2. Collaborative Filtering

2.3. Content-Based Recommendation

2.4. Hybridization & Evaluation

1. What are Recommender Systems?

- Recommender systems are about **matching users and items**
- Recommender systems are **about discovery not search**
 - no explicit information need; no explicit query
 - rather: *“entertain me”, “show me something interesting”*
- Recommender systems have **big business impact** [5]
 - 66% of movies watched on Netflix have been recommended
 - 35% of sales of Amazon.com are based on recommendations

Goals

- User: A good recommender brings up items that are
 - **relevant** (i.e., the user likes them once he uses them)
 - **novel** (i.e., the user does not yet know about the items)
 - **surprising** (i.e., the items are different from what the user already knows)
- Company: A good recommender brings up items that
 - users are likely to purchase (i.e., buy, rent, watch)
 - have high margins (e.g., to drive earnings)

Netflix Prize

- **Competition** by Netflix video rental company
- **driver** for research in recommender systems
- ran over **three years** (2007 – 2009)
- goal was to **beat CineMatch** (Netflix's recommendation algorithm) **by more than 10%** in terms of **root mean squared error (RMSE)**
- award: **\$1,000,000**
- included a **data release** (100M ratings from 480K users for 17K movies); now **retracted** due to legal issues
- winning approach **BellKor's Pragmatic Chaos** [2]
was a combination of several independently proposed approaches



Approaches

- **Different research communities** (e.g., DM, IR, ML) have worked on recommender systems and come up with **very different ideas**
- **Collaborative filtering** only assumes (partial) knowledge about how useful specific items are to specific users (e.g., ratings)
- **Content-based recommendation**, in addition, knows about properties of the items (e.g., cast of movie, content of book)
- **Hybridization strategies** aim to provide better recommendations by systematically combining multiple baseline recommenders










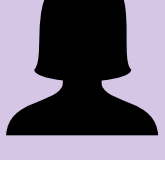
2. Collaborative Filtering

- **Collaborative filtering** only assumes (partial) knowledge about **how useful specific items are to specific users** (e.g., ratings)
- **No background knowledge** about items (e.g., cast or content) or users (e.g., age, gender, location)
- Challenges:
 - recommend **few** items from a **large** pool
 - **data sparsity** (large number of users and items)
 - **scalability**

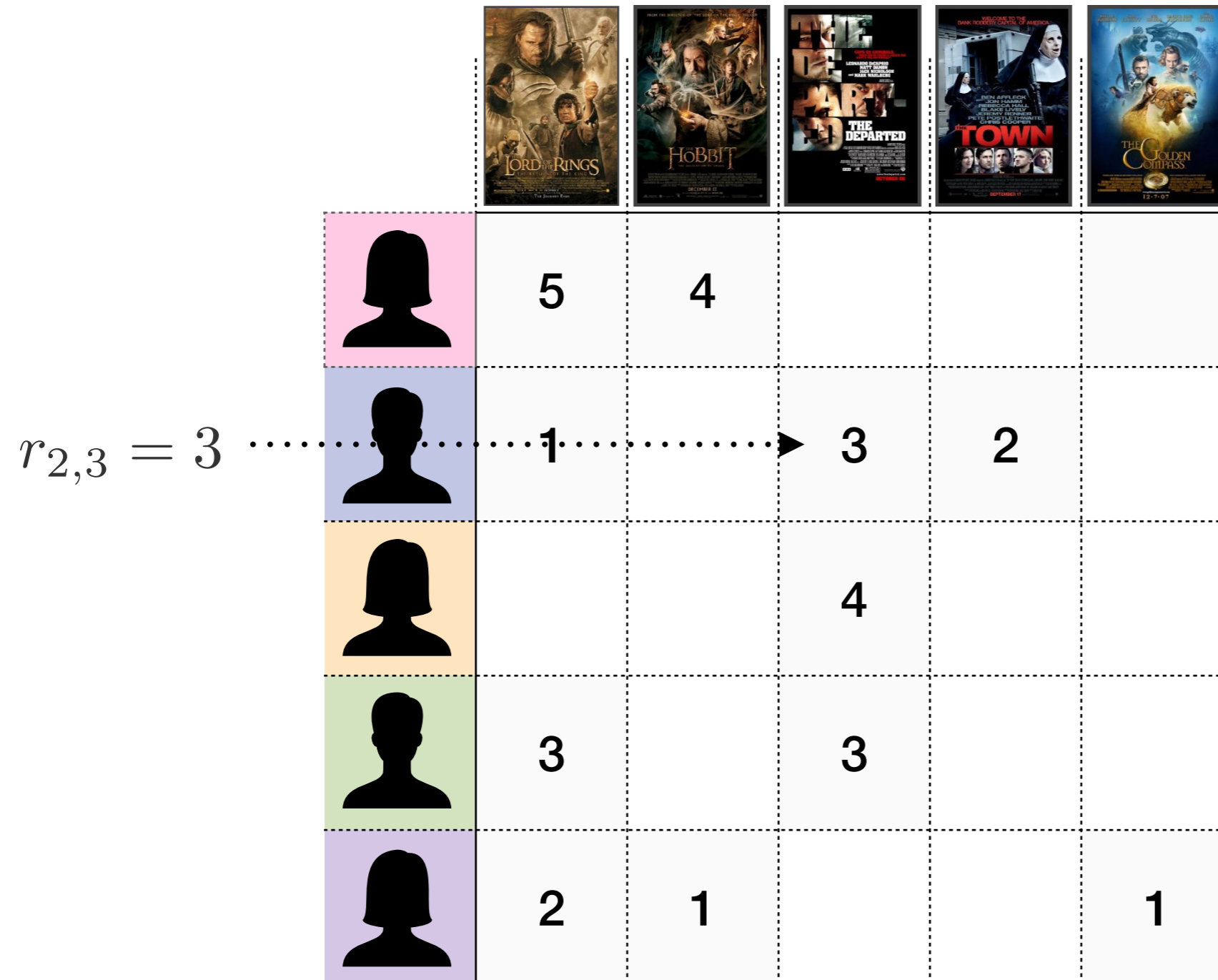
Explicit vs. Implicit Utility

- **Explicit utility values** are directly provided by users (e.g., ratings)
 - **none available for new users** (cold start problem)
 - users are **typically reluctant** to provide ratings
 - **not necessarily comparable** (pessimists vs. optimists)
- **Implicit utility values** can be obtained by observing users
 - based on **transactions** (e.g., purchases or clicks)
 - by **measuring engagement** (e.g., time spend watching video)

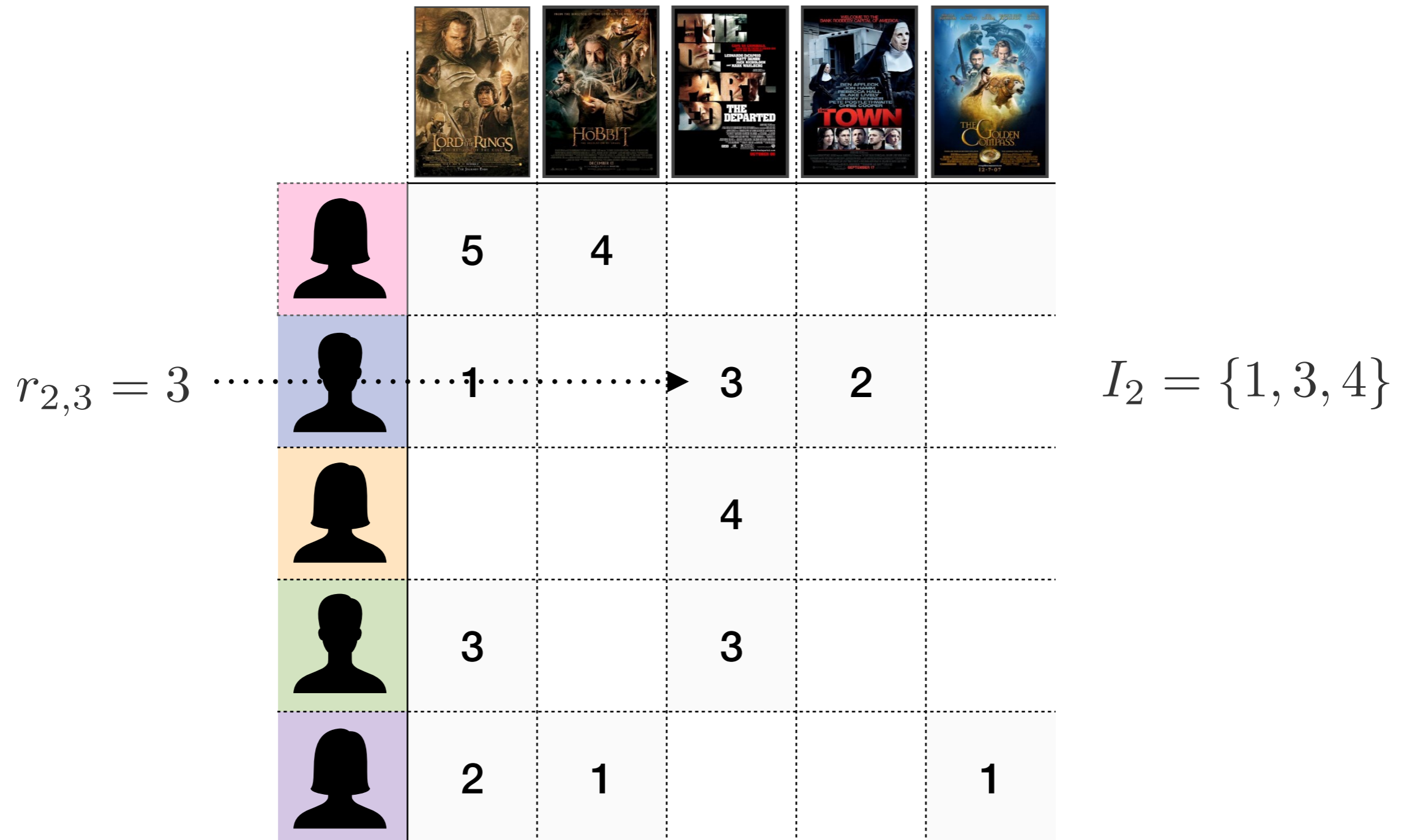
Utility Matrix

					
	5	4			
	1		3	2	
			4		
	3		3		
	2	1			1

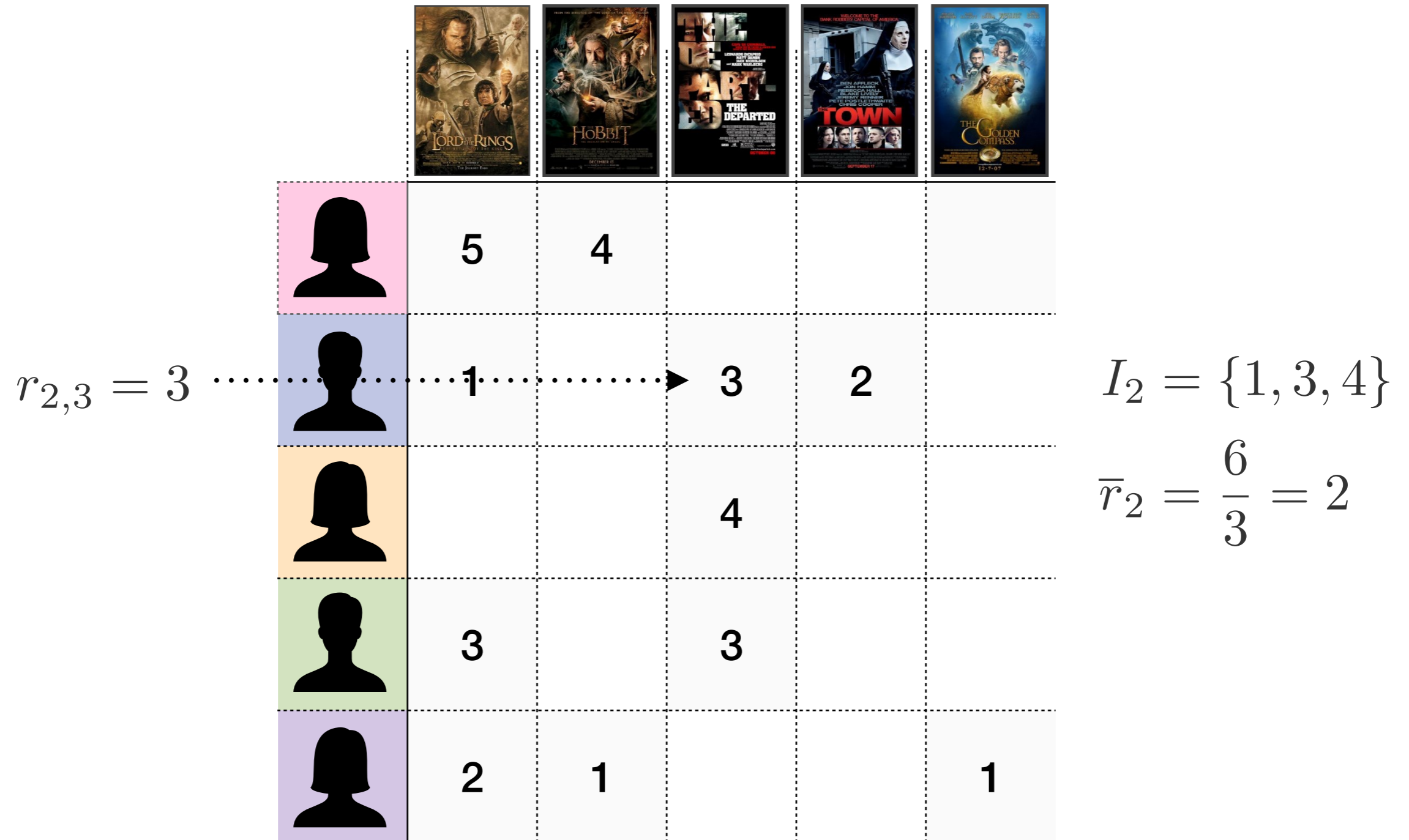
Utility Matrix



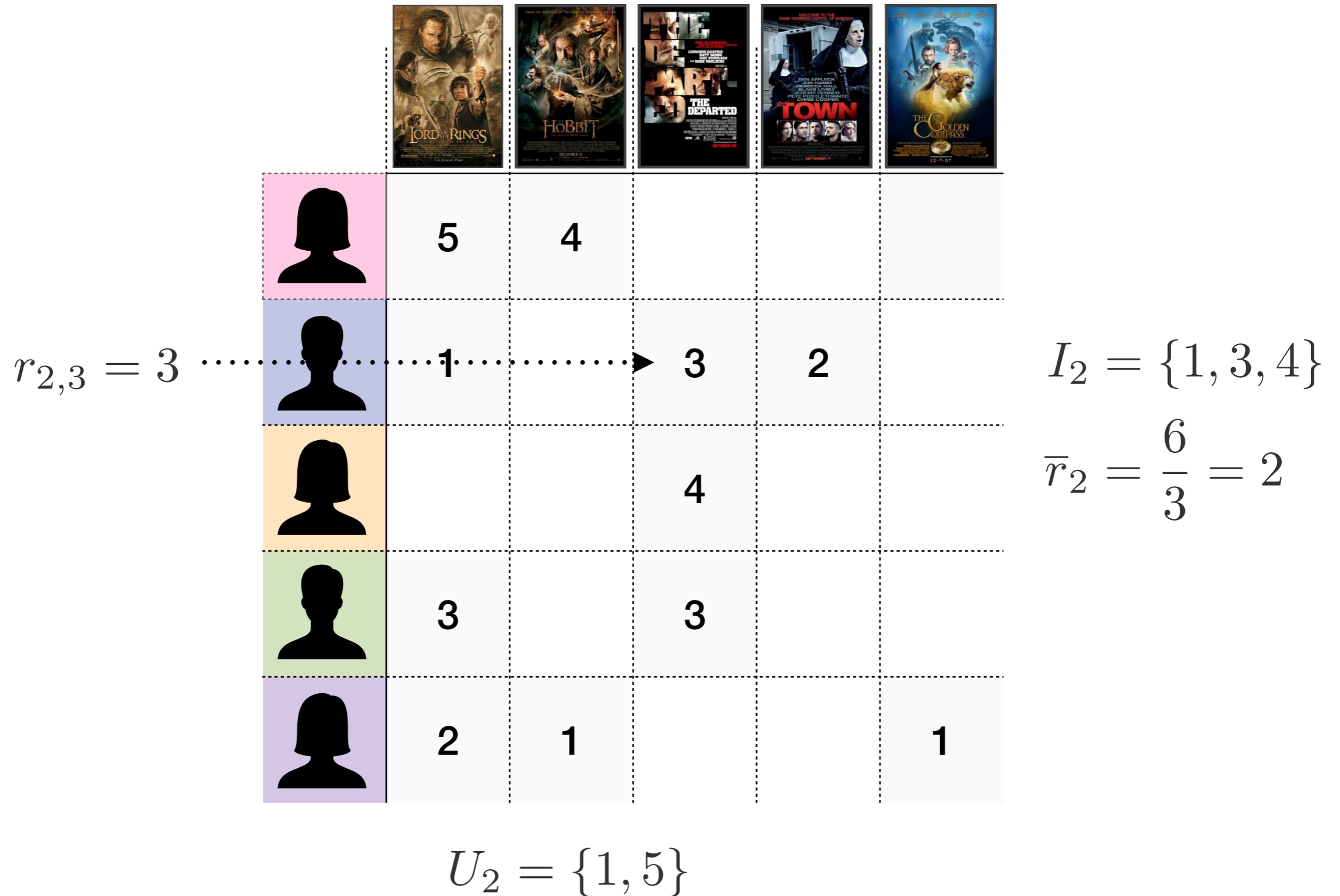
Utility Matrix



Utility Matrix



Utility Matrix



Characteristics

- **Most values** of the utility matrix **are missing**, i.e., the **data is very sparse** (e.g., in Netflix dataset only 1% of values is known)
- **Missing values** are **different from zeros** and do not indicate that the user dislikes the item
- **Magnitude of utility values** (e.g., ratings) differs from user to user (optimists vs. pessimists)

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

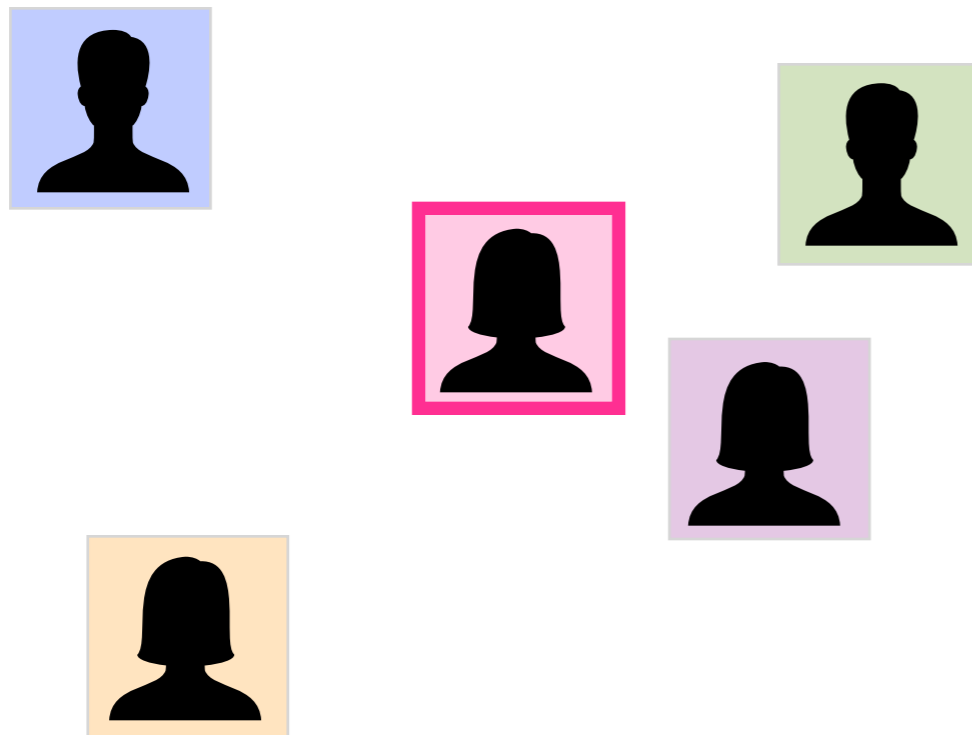
2.1. User-User Collaborative Filtering











- User-user collaborative filtering aka. **k-NN collaborative filtering** as first generation of recommenders (proposed in early 1990's)
- Idea: Recommend items that are of high utility to **similar users**

	5	4			
	1		3	2	
			4		
	3		3		
	2	1			1

2.1. User-User Collaborative Filtering

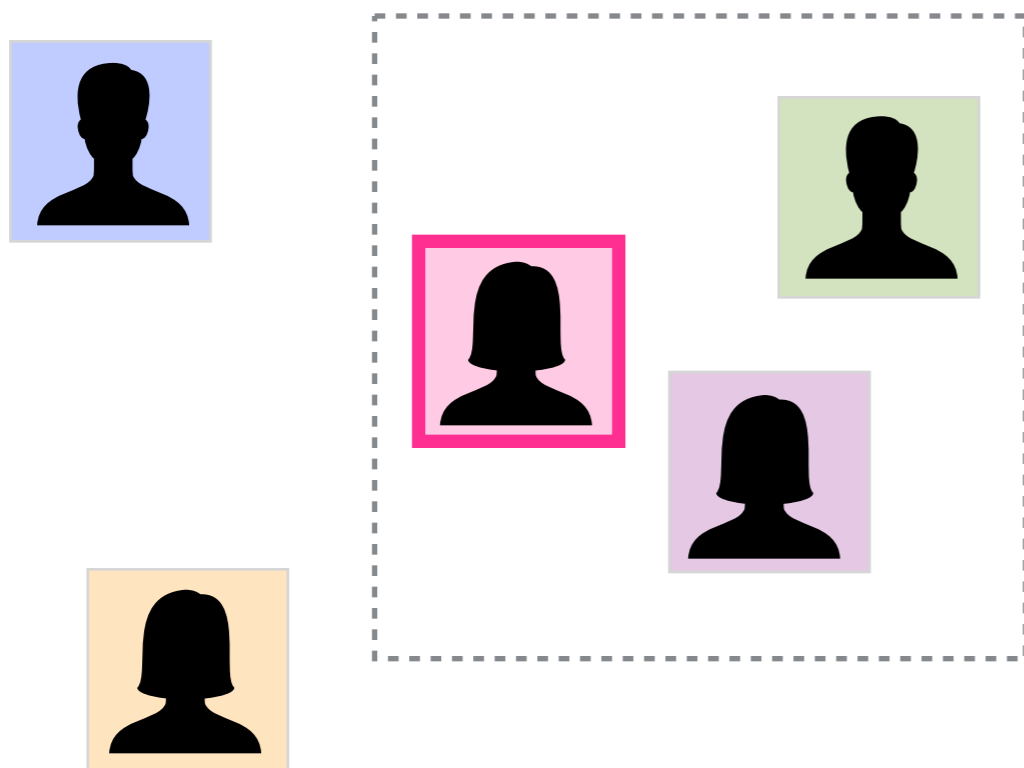
- User-user collaborative filtering aka. **k-NN collaborative filtering** as first generation of recommenders (proposed in early 1990's)
- Idea: Recommend items that are of high utility to **similar users**













					
	5	4			
	1		3	2	
			4		
	3		3		
	2	1			1

2.1. User-User Collaborative Filtering

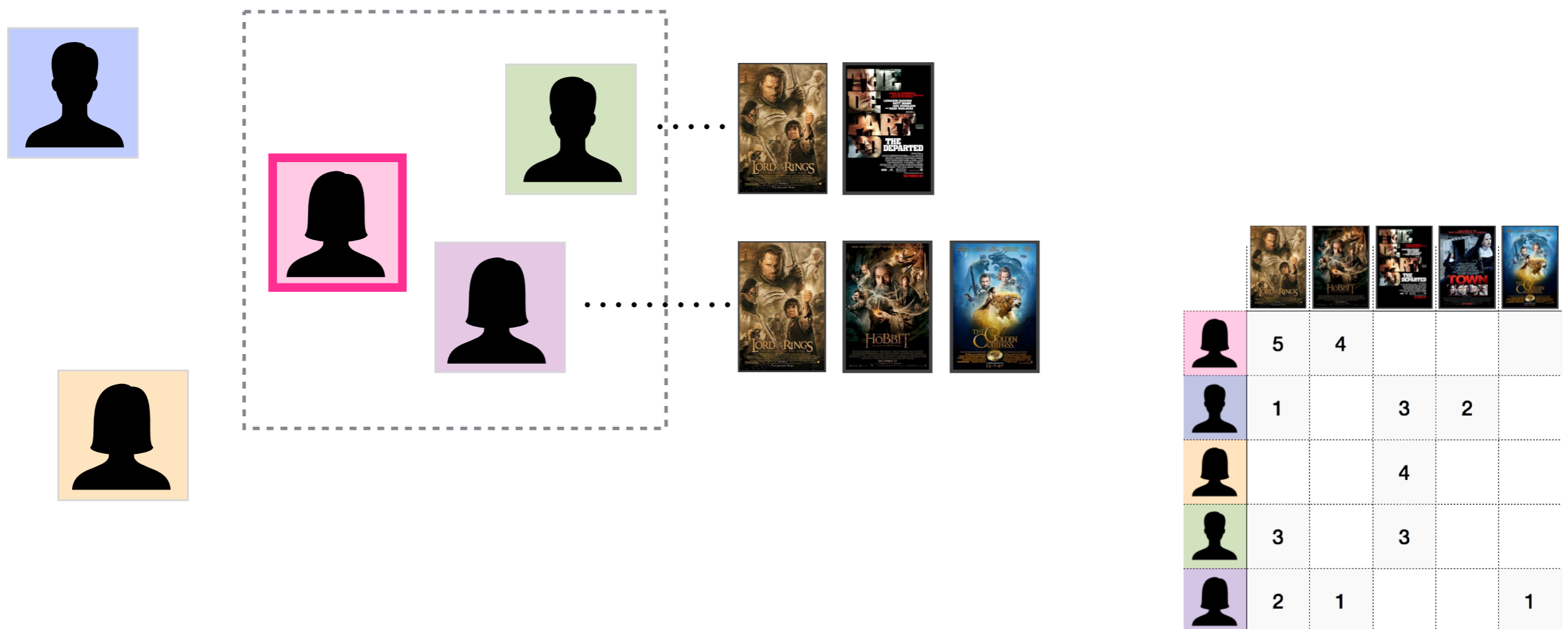
- User-user collaborative filtering aka. **k-NN collaborative filtering** as first generation of recommenders (proposed in early 1990's)
- Idea: Recommend items that are of high utility to **similar users**



					
	5	4			
	1		3	2	
			4		
	3		3		
	2	1			1

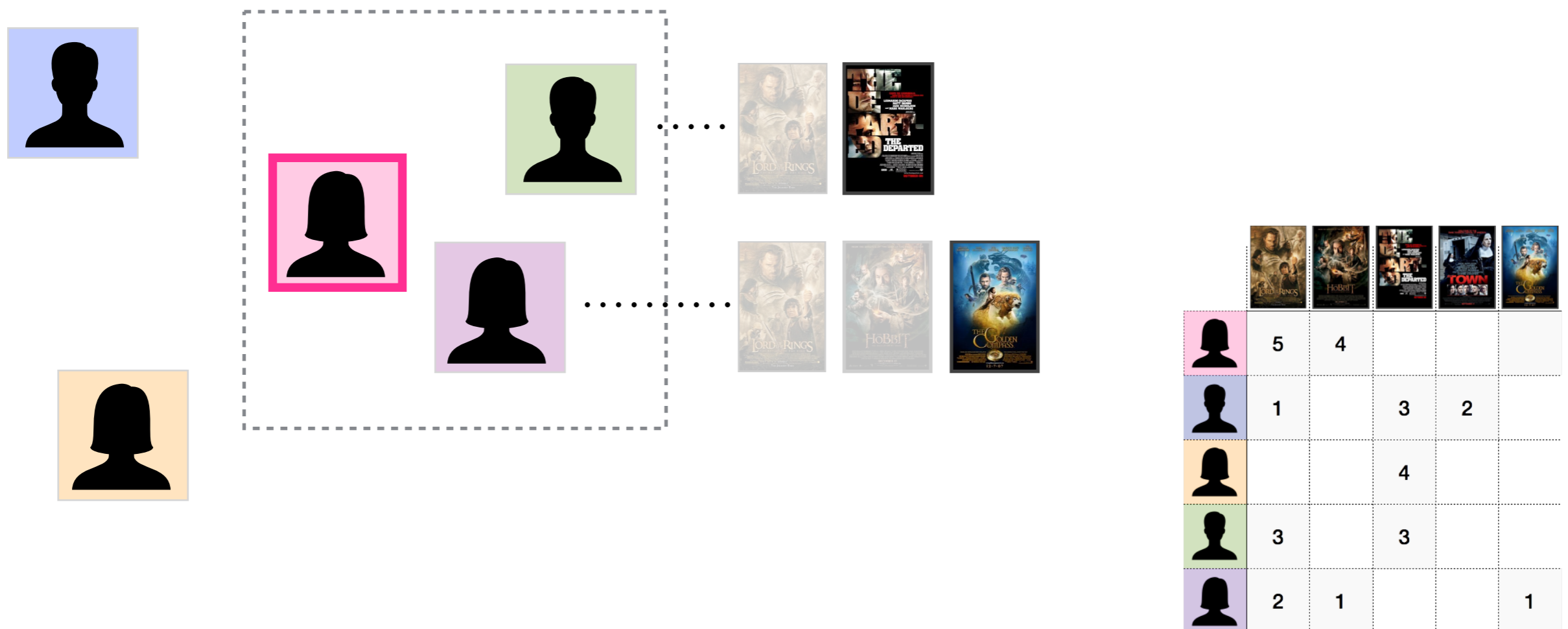
2.1. User-User Collaborative Filtering

- User-user collaborative filtering aka. **k-NN collaborative filtering** as first generation of recommenders (proposed in early 1990's)
- Idea: Recommend items that are of high utility to **similar users**



2.1. User-User Collaborative Filtering

- User-user collaborative filtering aka. **k-NN collaborative filtering** as first generation of recommenders (proposed in early 1990's)
- Idea: Recommend items that are of high utility to **similar users**



Measures of User Similarity











- How can we measure the similarity between two users u and v ?

- Pearson correlation** (on items with known utility for both users)

$$s(u, v) = \frac{\sum_{i \in I_u \cap I_v} (r_{u,i} - \bar{r}_u) \cdot (r_{v,i} - \bar{r}_v)}{\sqrt{\sum_{i \in I_u \cap I_v} (r_{u,i} - \bar{r}_u)^2} \cdot \sqrt{\sum_{i \in I_u \cap I_v} (r_{v,i} - \bar{r}_v)^2}}$$

- Cosine similarity** (missing utility values as zeros)

$$s(u, v) = \frac{\sum_i (r_{u,i} \cdot r_{v,i})}{\sqrt{\sum_i r_{u,i}^2} \cdot \sqrt{\sum_i r_{v,i}^2}}$$

					
	5	4			
	1		3	2	
			4		
	3		3		
	2	1			1

Generating Recommendations

- Identify **neighborhood** $N(u,k)$ of k **users most similar** to u

- Predict utility** of item i as

$$\hat{r}_{u,i} = \underbrace{\bar{r}_u}_{\text{Baseline prediction}} + \frac{\sum_{v \in N(u,k)} s(u,v) \cdot \underbrace{(r_{v,i} - \bar{r}_v)}_{\text{Deviation of similar user } v}}{\sum_{v \in N(u,k)} s(u,v)}$$

- Recommend n items having **highest predicted utility**











Discussion

- **Pearson correlation** and **cosine similarity** only work if users u and v have **known utility values for common item** (e.g., have rated at least one common movie)
- User similarity is **sensitive to updates** (e.g., additional ratings) so that precomputing user similarities is not attractive
- **Neighborhood computation** is computationally expensive

2.2. Item-Item Collaborative Filtering

- **Item-item collaborative filtering** addresses the shortcomings of user-user collaborative filtering (proposed in early 2000's)
- Idea: Recommend **items that are similar** to items of high utility













					
	5	4			
	1		3	2	
			4		
	3		3		
	2	1			1

2.2. Item-Item Collaborative Filtering

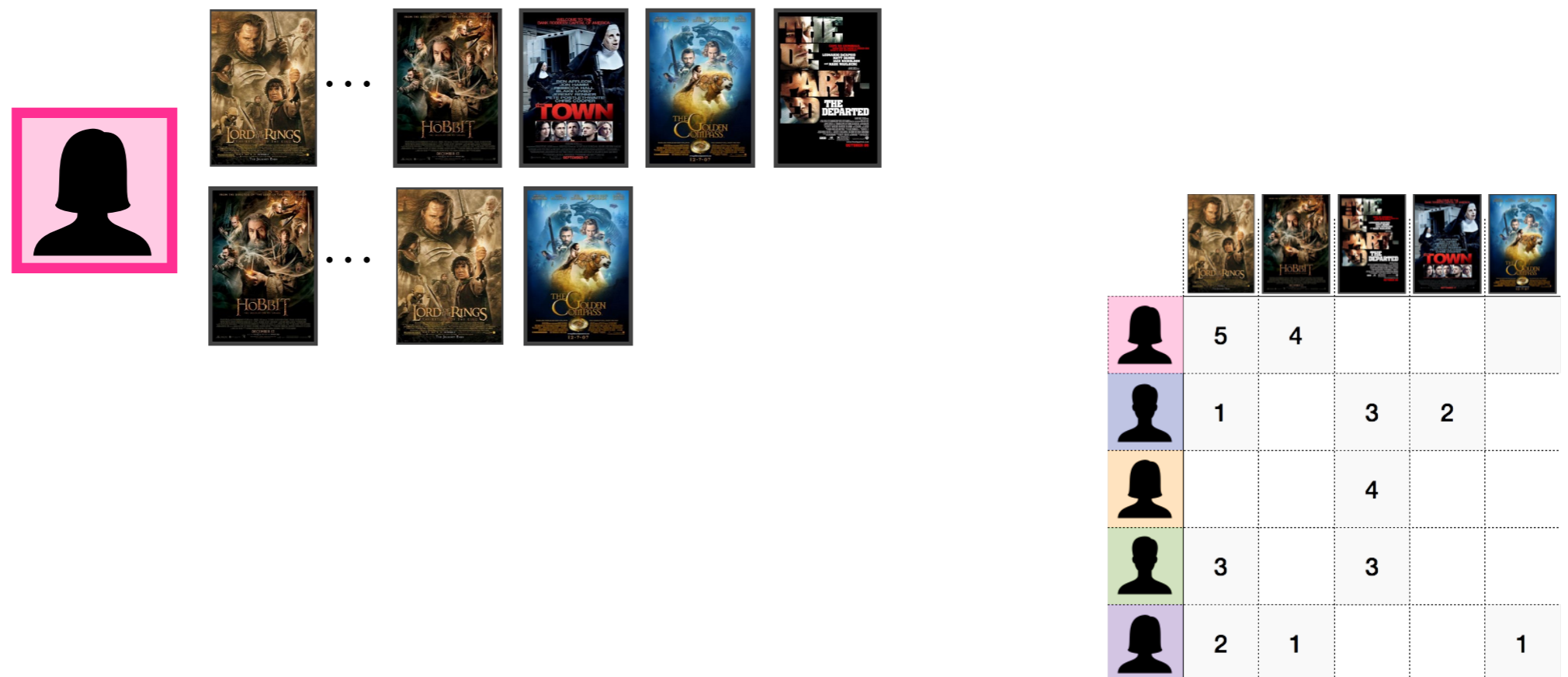
- **Item-item collaborative filtering** addresses the shortcomings of user-user collaborative filtering (proposed in early 2000's)
- Idea: Recommend **items that are similar** to items of high utility



					
	5	4			
	1		3	2	
			4		
	3		3		
	2	1			1

2.2. Item-Item Collaborative Filtering

- Item-item collaborative filtering addresses the shortcomings of user-user collaborative filtering (proposed in early 2000's)
- Idea: Recommend **items that are similar** to items of high utility













Measures of Item Similarity

- How can we measure the similarity between two items i and j ?
- Pearson correlation** (on users with known utility for both items)

$$s(i, j) = \frac{\sum_{u \in U_i \cap U_j} (r_{u,i} - \bar{r}_u) \cdot (r_{u,j} - \bar{r}_u)}{\sqrt{\sum_{u \in U_i \cap U_j} (r_{u,i} - \bar{r}_u)^2} \cdot \sqrt{\sum_{u \in U_i \cap U_j} (r_{u,j} - \bar{r}_u)^2}}$$

- Cosine similarity** (missing utility values as zeros)

$$s(i, j) = \frac{\sum_u (r_{u,i} \cdot r_{u,j})}{\sqrt{\sum_u r_{u,i}^2} \cdot \sqrt{\sum_u r_{u,j}^2}}$$

					
	5	4			
	1		3	2	
			4		
	3		3		
	2	1			1

Generating Recommendations

- **Predict utility** of item i as

$$\hat{r}_{u,i} = \underbrace{\bar{r}_u}_{\text{Baseline prediction}} + \frac{\sum_{j \in S(u,i,k)} s(i,j) \cdot \underbrace{(r_{u,j} - \bar{r}_u)}_{\text{Deviation for similar item } j}}{\sum_{j \in S(u,i,k)} s(i,j)}$$

with $S(u,i,k)$ as the set of k items with **known utility** for user u that are **most similar** to item i

- Recommend n items having **highest predicted utility**

Discussion

- **Pearson correlation** and **cosine similarity** only work if items i and j have **known utility values for common user** (e.g., have been rated by the same user)
- Item similarity is **less sensitive to updates** (e.g., additional ratings), assuming that there are **many more users than items**
- In practice, **item similarities are typically precomputed**, and **truncated** (keeping top- k most similar items per item)

2.3. Association Rules

- **Association rule mining** developed for **market basket analysis** to learn rules (patterns) from **customer transactions** (e.g., buys **soda** and **beer** \Rightarrow buys **snacks**)
- Association rules can be used to **generate recommendations** by considering **items with known utility** per user a transaction
- Let **A** and **B** be set of items, we are interested in identifying association rules **A** \Rightarrow **B** with sufficient **support** and **confidence**

Support and Confidence

- For a set of items (itemset) A its **support** $s(A)$ is the **fraction of transactions that contains A**

$$s(A) = \frac{\# \text{ transactions containing } A}{\# \text{ transactions}}$$

- For an association rule $A \Rightarrow B$ its **confidence** $c(A \Rightarrow B)$ is the **fraction of transactions containing A that also contain B**

$$c(A \Rightarrow B) = \frac{\# \text{ transactions containing } A \cup B}{\# \text{ transactions containing } A}$$

Identifying Frequent Itemsets

- **Apriori algorithm** [1] can be used to **identify frequent itemsets** having a support above a minimum support threshold

- **Iterative algorithm** exploiting **anti-monotonicity of supports**

$$A \subset B \Rightarrow s(A) \geq s(B)$$

- Sketch:

- **identify** frequent 1-itemsets (i.e., containing a single item)
- **repeat** (until no frequent k-itemsets are found)
 - **generate candidates** by joining frequent (k-1)-itemsets
 - **prune** infrequent candidates and emit frequent k-itemsets

Generating Association Rules

- Generate **association rules** from frequent itemset X
 - consider every **non-empty subset** $A \subset X$ and let $B = X \setminus A$
 - output **association rule** $A \Rightarrow B$ if $c(A \Rightarrow B)$ above threshold

Generating Recommendations

- Consider all items I_u with **known utility** for user u
 - identify all association rules $A \Rightarrow B$ so that $A \subseteq I_u$
 - items from $B \setminus I_u$ are **candidates** for recommendation; for each candidate keep track of **highest confidence** of any association rule suggesting it
 - recommend n items having highest confidence

2.4. Dimensionality Reduction

- Idea: Identify a **small number** (in comparison to m and n) **of common interests** (topics) to represent users and items; recommend items to users that belong to the same topics
- Utility matrix **R** can be seen as user vectors (in a m -dimensional vector space) or item vectors (in a n -dimensional vector space)
- **Dimensionality reduction** methods reveal the **latent structure** of a matrix by representing it as a **product of multiple smaller matrices** (e.g., UV decomposition, singular value decomposition, principal component analysis)

Singular Value Decomposition

- Determine **k-SVD** of utility matrix R ($m \times n$)

$$\begin{array}{c} n \\ \boxed{R} \\ m \end{array} \approx \begin{array}{c} k \\ \boxed{U} \\ m \end{array} \times \begin{array}{c} k \\ \boxed{\Sigma} \\ k \end{array} \times \begin{array}{c} k \\ \boxed{T^T} \\ n \end{array}$$

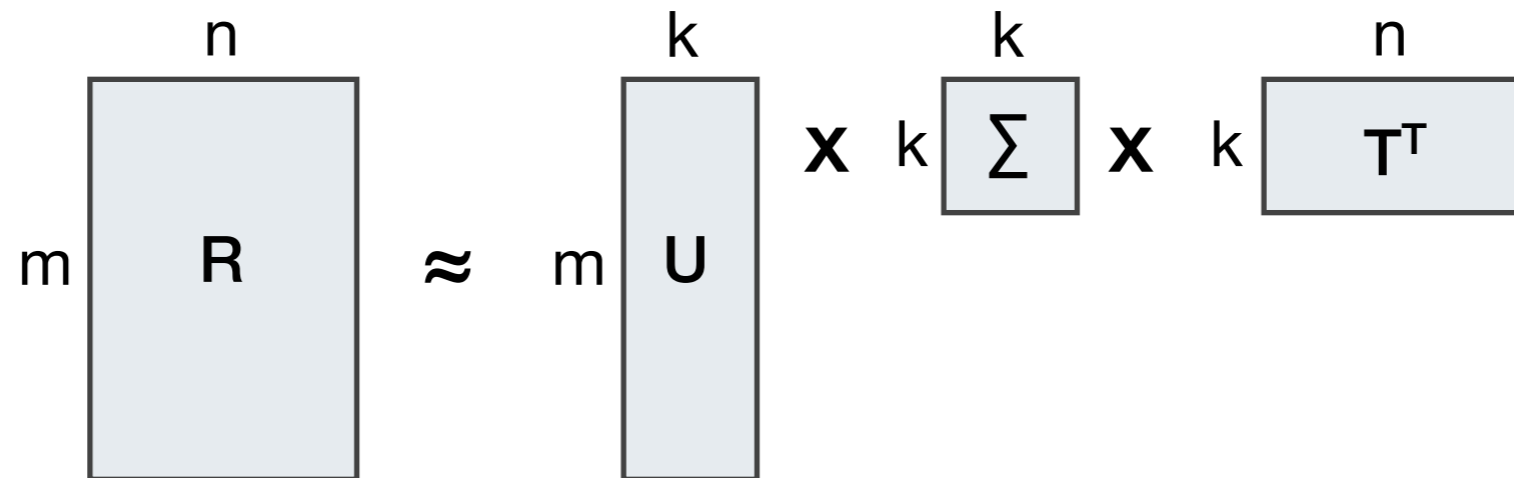
as best possible rank- k approximation under Frobenius norm

- \mathbf{U} captures **user-topic associations**
- Σ captures **topic importance**
- \mathbf{T} captures **item-topic associations**

Imputation

- **SVD** requires a **complete matrix** but R **misses a lot of values**
- **Imputation** is the process of filling missing values with defaults
 - average utility assigned to item by different users
 - average utility assigned to other items by same user
 - other baseline predictors

Generating Recommendations



- **Predict utility** of item i for user u as

$$\hat{r}_{u,i} = \sum_k U_{u,k} \cdot \Sigma_{k,k} \cdot T_{k,i}^T$$

- **Predict utilities** of all items for user u as

$$U_u \times \Sigma \times T^T$$

3. Content-Based Recommendation

- **Content-based recommendation** assumes (partial) knowledge about **how useful specific items are to specific users** and background knowledge about **properties of the items**
- Idea: Recommend **items that are similar** to items of high utility



3. Content-Based Recommendation

- **Content-based recommendation** assumes (partial) knowledge about **how useful specific items are to specific users** and background knowledge about **properties of the items**
- Idea: Recommend **items that are similar** to items of high utility



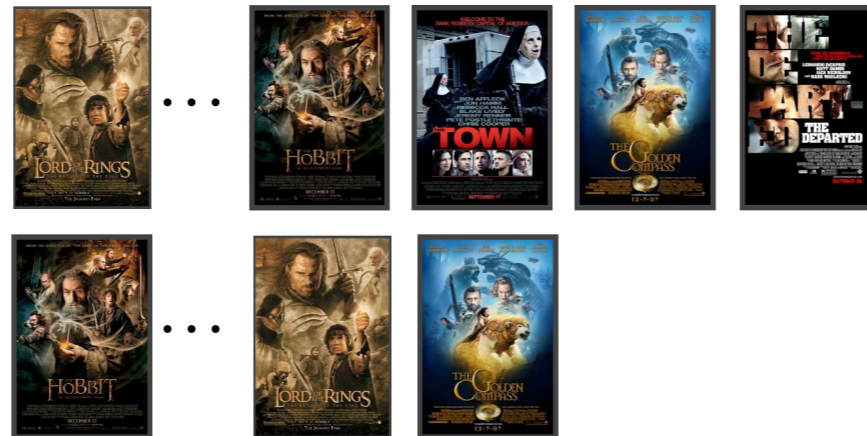
3. Content-Based Recommendation

- **Content-based recommendation** assumes (partial) knowledge about **how useful specific items are to specific users** and background knowledge about **properties of the items**
- Idea: Recommend **items that are similar** to items of high utility



3. Content-Based Recommendation

- Content-based recommendation assumes (partial) knowledge about how useful specific items are to specific users and background knowledge about properties of the items
- Idea: Recommend items that are similar to items of high utility



Actors:
VM, LT, IMK

Year:
2003

Content:
Third part of fantasy trilogy. Involves dwarfs and hobbits.



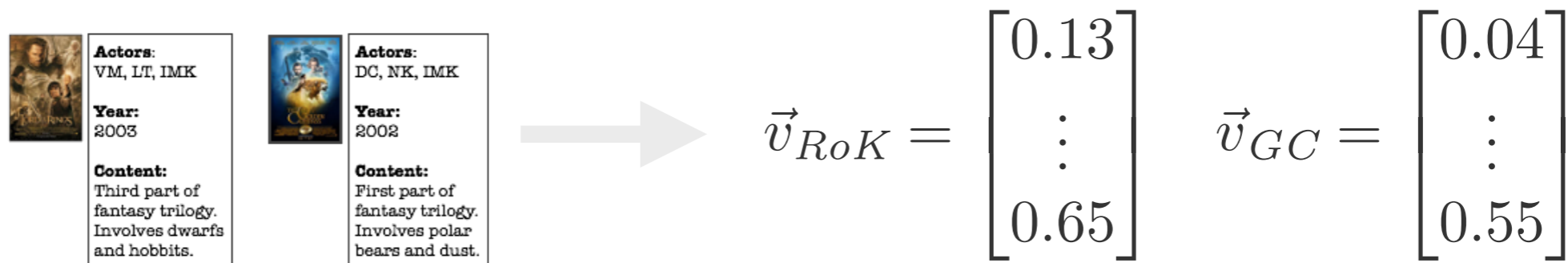
Actors:
DC, NK, IMK

Year:
2002

Content:
First part of fantasy trilogy. Involves polar bears and dust.

Items and Users as Vectors

- Represent **items as vectors** in a high-dimensional vector space (works well, for instance, for text documents with tf.idf weighting)



- Represent **user as vector** obtained as **weighted combination of item vectors** of items with known utility values

$$\vec{u} = \sum_{i \in I_u} \frac{r_{u,i}}{\sum_{j \in I_u} r_{u,j}} \cdot \vec{v}_i$$

- Recommend items with **high cosine similarity** to user vector

Domain-Specific Item Similarity

- **Not all item properties** are suitable for representation in vector and we may lose their semantics when doing so
 - **Category** (e.g., /Travel/U.S.A., /Travel/Canada, /Cooking/Italian)
 - **Year** (e.g., 1980 should be less similar to 2002 than 1981)
- Define **domain-specific item similarity** based on their properties, for instance, as weighted sum of property-specific similarities



Actors:
VM, LT, IMK
Year:
2003
Content:
Third part of
fantasy trilogy.
Involves dwarfs
and hobbits.



Actors:
DC, NK, IMK
Year:
2002
Content:
First part of
fantasy trilogy.
Involves polar
bears and dust.

$$\begin{aligned} s(RoK, GC) &= \alpha \cdot s_a(RoK, GC) \\ &+ \beta \cdot s_y(RoK, GC) \\ &+ \gamma \cdot s_c(RoK, GC) \end{aligned}$$

Domain-Specific Item Similarity

- ◉ Recommend **items that are similar** to items of high utility

$$score(u, j) = \sum_{i \in I_u} r_{u,i} \cdot s(i, j)$$

4. Hybridization & Evaluation

- **Combining different recommenders** can be attractive
 - improved recommendations (cf. winner of Netflix competition)
 - overcoming cold start problems
 - improved performance
- **Hybridization strategies** systematically combine recommenders
 - **Ensemble** (combine outputs of different recommenders)
 - **Switch** (choose recommender to use)

Ensemble

- Obtain (top-k) recommendations from **multiple recommenders**
- **Combine recommendations** by aggregating per item
 - **predicted utility** by different recommenders
 - **reciprocal rank** in output of different recommenders
 - **votes** (item in output) from different recommenders

R1									
	0.6	0.2	0.1	utility	0.6	0.6	0.4	0.2	0.2
R2				1/rank	1/1	4/3	1/2	1/3	1/2
	0.5	0.4	0.2	vote	1	2	1	1	1

Switch

- **Decide** (or learn to decide) when to use **which recommender**
- Example: Collaborative filtering suffers from **cold start problem**
 - use content-based recommender, if user has too few known utility values (e.g., has rated too few items)
 - otherwise, use item-item collaborative filtering

Evaluation

- Recommender systems can be evaluated like other IR systems
 - user judges **whether recommended items are relevant**
 - determine **precision, recall, F1**
 - captures only whether relevant items are returned
- More commonly, the **focus is on prediction accuracy**
 - **split utility values** from dataset (e.g., movie ratings) into **training and test data** (repeat multiple times)
 - measure **mean absolute error** on test data

$$\frac{1}{n} \sum_{(u,i)} |\hat{r}_{u,i} - r_{u,i}|$$

Summary

- **Recommender systems** help users to discover relevant and surprising items and drive many of today's businesses
- **Collaborative filtering** uses only knowledge about how useful items are to users; variety of approaches have been proposed
- **Content-based recommendation** also uses knowledge about properties of the items (e.g., content); IR-style approaches
- **Hybridization strategies** combine multiple recommenders, for instance, to obtain better recommendations or performance
- **Evaluation of recommender systems** usually focuses on prediction accuracy and uses training/test splitting of data

When Recommender Systems Fail

The screenshot shows an Amazon product page for an American Weigh 100g x 0.01g Digital Scale. The page layout includes a top navigation bar with the Amazon logo, search bar, and links for Sign In, Try Prime, and Wish List. Below the navigation bar are departmental links and a 'Holiday Toy List' banner. The main product area features a large image of the scale, a title, a star rating, and a price of \$10.12. A 'Customers Who Bought This Item Also Bought' section is visible at the bottom, displaying a carousel of related products such as rolling papers, storage containers, and measuring spoons.

amazon Try Prime Search Home & Kitchen : Go Sign In Try Prime Wish List

Departments Fire & Kindle Recommended for You Today's Deals Gift Cards Help Sell

Kitchen & Dining Best Sellers Small Appliances Kitchen Tools Cookware Bakeware Cutlery Dining & Entertaining Storage & Organization Wedding Registry

Home & Kitchen > Kitchen & Dining > Kitchen Utensils & Gadgets > Measuring Tools & Scales

American Weigh 100g x 0.01g Digital Scale
by American Weigh
★★★★★ 1,570 customer reviews | 68 answered questions

List Price: \$22.95
Price: **\$10.12** & FREE Shipping on orders over \$35. Details
You Save: **\$12.83 (56%)**

In Stock.
Sold by The Houseware Shoppe and Fulfilled by Amazon.

Want it tomorrow, Nov. 4? Order within **5 hrs 58 mins** and choose **One-Day Shipping** at checkout. Details

Size: **3 x 5 x 0.8 inches** | 5.5 x 3.5 x 1 inches | 1000 x 0.1 g

- Weighs up to 100 grams in 0.01 gram increments
- The backlit LCD display helps make the numbers viewable and easy to read
- Flip-open lid protects the delicate weighing surface
- Powered by 2 AAA batteries (included)
- Backed by a powerful 10 year warranty

60 new from \$7.01 2 used from \$14.50

Share [social icons]

Qty: 1

Yes, I want **FREE Two-Day Shipping** with Amazon Prime

Add to Cart

Turn on 1-Click ordering

Add to Wish List

Other Sellers on Amazon

\$10.07 + Free Shipping
Sold by: A-SZCXTOP(Shipping from HongKong) Add to Cart

\$10.37 & FREE Shipping on orders over \$35.00. Details
Sold by: Amazon.com Add to Cart

62 used & new from \$7.01

Have one to sell? Sell on Amazon

Customers Who Bought This Item Also Bought

Page 2 of 11 | Start over

- Raw King Size Organic Cigarette Rolling Papers, 4 Packs
★★★★★ 259
\$4.72
- 500 CLEAR Reclosable Zipper Bag, 2" x 2" - 2 mil thick
★★★★★ 319
\$5.51 Prime
- SPACE CASE Grinder Sifter Magnetic 4 Pc. Medium Titanium
★★★★★ 103
\$66.99 Prime
- 500 RAW Rolling Papers Filter Tips (10 Booklets of 50) Standard Size Vegan Pearl Tinted
★★★★★ 242
\$4.14
- Tightvac Minivac 1-Ounce Vacuum Sealed Dry Goods Storage Container, Black Pearl Tinted
★★★★★ 22
\$14.89 Prime
- Raw Rolling Tray + Raw 110mm Roller + Raw King Size Rolling Papers Bundle
★★★★★ 149
\$12.49 Prime
- Norpro 3080 Mini Stainless Steel Measuring Spoons, Set includes (tad, dash, pinch,...
★★★★★ 402
\$6.69 Prime
- Newer® 100 Gram Stainless Steel Calibration Weight for Digital Jewellery...
★★★★★ 27
\$4.25
- Chromium Crusher 3.0" Heavy Duty Durable Zinc Tobacco Spice Herb Grinder w/ Lifetime...
★★★★★ 59
\$19.96 Prime

Source: Alexis C. Madrigal: *The (Unintentional) Amazon Guide to Dealing Drugs*, The Atlantic, April 15 2014
<http://www.theatlantic.com/technology/archive/2014/04/the-unintentional-amazon-guide-to-dealing-drugs/360636/>

References

- [1] **R. Agrawal and R. Srikant:** *Fast Algorithms for Mining Association Rules*
VLDB 1994
- [2] **M. D. Ekstrand, J. T. Riedl, J. A. Konstan:**
Collaborative Filtering Recommender Systems,
FTIR 4(2):81–173, 2010
- [3] **Y. Kohen:** *The BellKor Solution to the Netflix Grand Prize*
http://www.netflixprize.com/assets/GrandPrize2009_BPC_BellKor.pdf
- [4] **J. Leskovec, A. Rajaraman, J. D. Ullman:** *Mining of Massive Datasets (Chapter 9: Recommendation Systems)*, 2014
Available at: <http://www.mmids.org>
- [5] **A. Karatzoglou:** *Recommender Systems*,
Tutorial at European Summer School for Information Retrieval, 2013
- [6] **G. Linden, B. Smith, and J. York:** *Amazon.com recommendations Item-to-item collaborative filtering*, IEEE Internet Computing 7(1):76–80, 2003