

Learning Analytics

Dr. Bowen Hui

Computer Science

University of British Columbia Okanagan

Recall: Two Major Types of Recommendation Systems

- **Content based** approach
 - Using content to understand user
- **Collaborative** approach
 - Using other users to understand user
 - Also called **collaborative filtering**



Example #1 from Amazon.ca

- Content based approach

Inspired by your browsing history

Page 1 of 13



The screenshot shows a horizontal scrollable list of four product recommendations. Each item includes a product image, a title, a star rating, and the number of offers. Navigation arrows are visible on the left and right sides of the scrollable area.

Product	Rating	Offers	Price
LEGO Harry Potter Advent Calendar 75964 Building Kit (305 Piece)	★★★★★	159	16 offers from CDN\$ 39.16
Hatchimals - Colleggtibles - Advent Calendar - Exclusive Hatchimals & Nests - Ages 5+	★★★★★	151	11 offers from CDN\$ 40.00
NINTENDO 400312 Super Mario Advent Calendar	★★★★★	60	CDN\$ 69.03 ✓prime
Paw Patrol - Advent Calendar - Includes 24 Collectible Figures - Ages 3+, 2018 Release	★★★★★	163	CDN\$ 32.98 ✓prime

I was looking at kids advent calendars...
will these recommendations be useful to me today?

Example #2 from Amazon.ca

- Collaborative approach

Customers who viewed this item also viewed

Page 1 of 10



←

→

Product Name	Rating	Price	Prime
Learning Resources Programmable Robot Mouse - LER2841	★★★★☆ 20	CDN\$ 29.82	✓prime
Learning Resources Botley The Coding Robot Activity Set, 77 Pieces	★★★★☆ 24	CDN\$ 61.46	✓prime
Learning Resources Botley The Coding Robot Action Challenge Accessory Set, Multicolor	★★★★☆ 2	CDN\$ 25.99	✓prime
Learning Resources Code & Go Robot Mouse Math, 16 Pieces	★★★★☆ 1	CDN\$ 25.99	✓prime
Learning Resources Botley The Coding Robot, Coding STEM Toy, 45 Piece Coding Set, Ages 5+	★★★★☆ 11	CDN\$ 61.93	✓prime

Collaborative Filtering (CF)

- Underlying assumption
 - Similar users have similar preferences

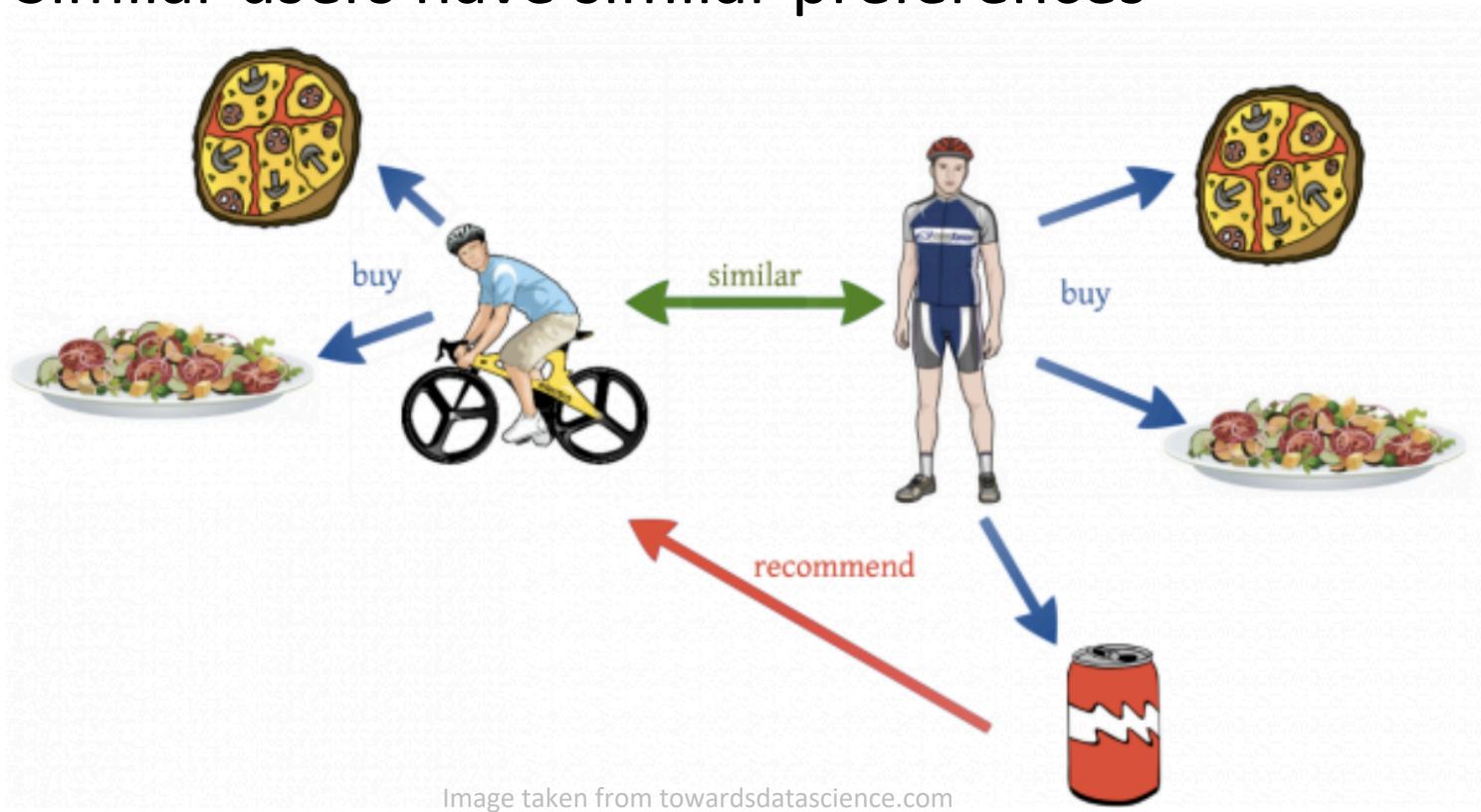


Image taken from towardsdatascience.com

CF Techniques

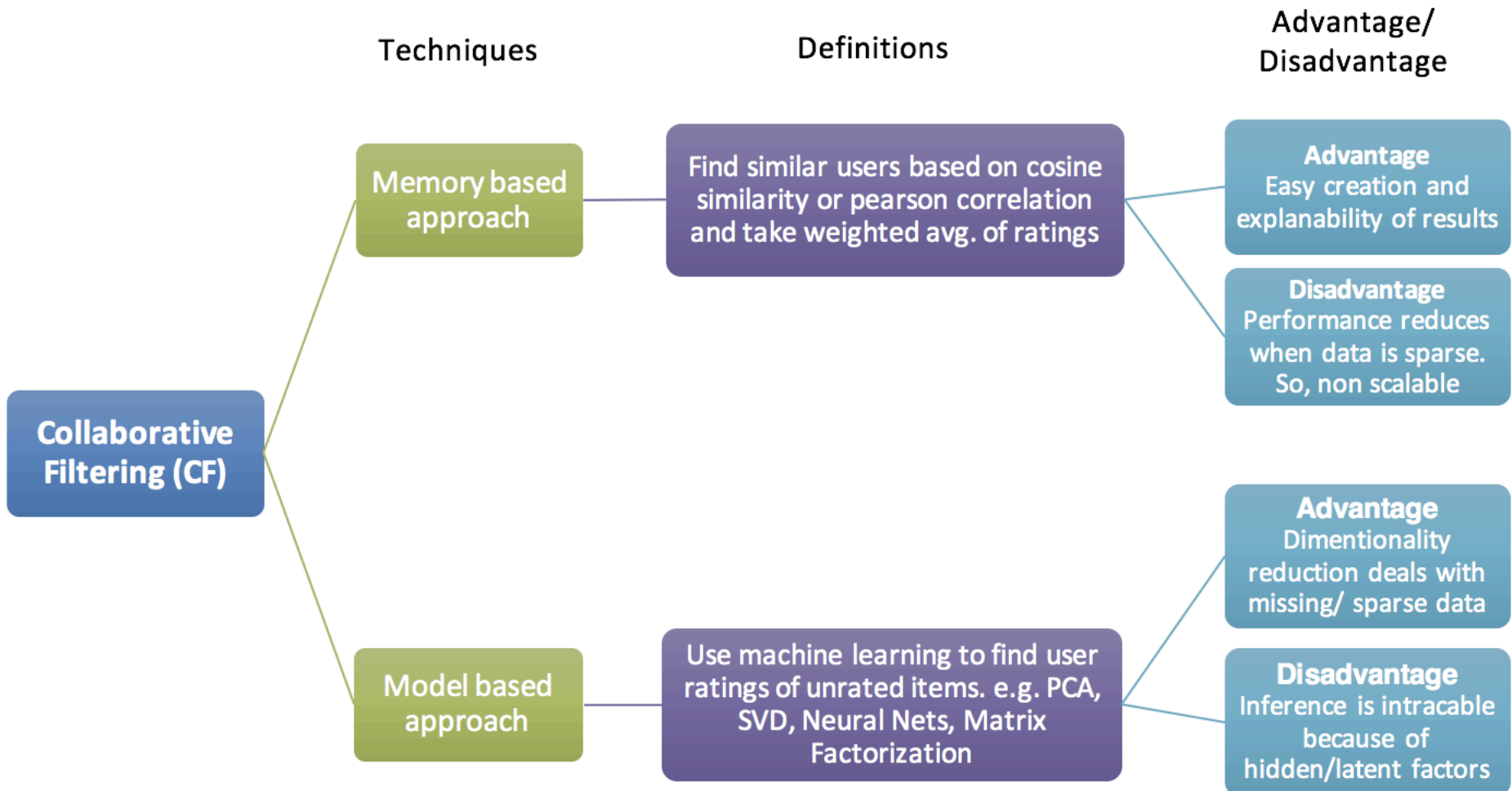


Image taken from towardsdatascience.com

Memory Based Approach

- User-item filtering

- Find other similar users based on similarity of ratings
- Recommend items those similar users like
- “Users who are similar to you also liked...”

Allows for
exploration!

- Item-item filtering

- Take an item, find users who liked that item, find other items that those users also like
- “Users who liked this item also liked...”

User-Item Filtering Process

- A database of user preferences
 - Based on explicit and/or implicit ratings
 - E.g. likes or dislikes, 5 stars
 - E.g. num views, add to wishlist, time spent on article
 - Continually updated
- Represent as **rating matrix**
 - Represented as a huge user-by-item matrix, $R=[r_{i,j}]$ where $r_{i,j}$ is user i 's rating of item j
 - Not every user will rate every item
 - A **sparse** matrix has many empty cells

	i_1	i_2	i_3	i_4	i_5
u_1	5		4	1	
u_2		3		3	
u_3		2	4	4	1
u_4	4	4	5		
u_5	2	4		5	2

Steps Involved

- Task: Recommend items based on preferences of similar users
- Steps:
 - How to determine similar users?
 - Given similar users, how to determine a user's missing rating of an item (based on ratings of similar users)?

Finding Similar Users

- How to identify similar users?
 - Not by demographic info like age, type of item, or other info about users or items
- Treat users as vectors of ratings
 - Two users are similar if they have similar ratings to the same items
(even if they are 20 years apart!)
 - Apply cosine similarity (previous lecture)
 - Define a k cutoff for number of similar users

Estimating Missing Rating

- What should be the value for $u_{3,1}$?
- Suppose nobody else rated i_1
 - Estimate using u_3 's average rating
 - No other information available
- Suppose similar users rated i_1
 - Should we still just use u_3 's average rating?

	i_1	i_2	i_3	i_4	i_5
u_1	5		4	1	
u_2		3		3	
u_3	?	2	4	4	1
u_4	4	4	5		
u_5	2	4		5	2

Estimating Missing Rating

- What should be the value for $u_{3,1}$?
- Suppose nobody else rated i_1
 - Estimate using u_3 's average rating
 - No other information available
- Suppose similar users rated i_1
 - Should we still just use u_3 's average rating?

No – need feedback from similar users

	i_1	i_2	i_3	i_4	i_5
u_1	5		4	1	
u_2		3		3	
u_3	?	2	4	4	1
u_4	4	4	5		
u_5	2	4		5	2

Estimating Missing Rating

- What should be the value for $u_{3,1}$?
- Suppose nobody else rated i_1
 - Estimate using u_3 's average rating
 - No other information available
- Suppose similar users rated i_1
 - Should we just use average of ratings made by k similar users?

	i_1	i_2	i_3	i_4	i_5
u_1	5		4	1	
u_2		3		3	
u_3	?	2	4	4	1
u_4	4	4	5		
u_5	2	4		5	2

Estimating Missing Rating

- What should be the value for $u_{3,1}$?
- Suppose nobody else rated i_1
 - Estimate using u_3 's average rating
 - No other information available
- Suppose similar users rated i_1
 - Should we just use average of ratings made by k similar users?

	i_1	i_2	i_3	i_4	i_5
u_1	5		4	1	
u_2		3		3	
u_3	?	2	4	4	1
u_4	4	4	5		
u_5	2	4		5	2

No – need to know how similar each of those users are to you

Estimating Missing Rating

- What should be the value for $u_{3,1}$?
- Suppose nobody else rated i_1
 - Estimate using u_3 's average rating
 - No other information available
- Suppose similar users rated i_1
 - What if some users are always tougher than others and always give lower ratings?

	i_1	i_2	i_3	i_4	i_5
u_1	5		4	1	
u_2		3		3	
u_3	?	2	4	4	1
u_4	4	4	5		
u_5	2	4		5	2

Estimating Missing Rating

- What should be the value for $u_{3,1}$?
- Suppose nobody else rated i_1
 - Estimate using u_3 's average rating
 - No other information available
- Suppose similar users rated i_1
 - What if some users are always tougher than others and always give lower ratings?

Need to modify each rating by a user's own average rating

	i_1	i_2	i_3	i_4	i_5
u_1	5		4	1	
u_2		3		3	
u_3	?	2	4	4	1
u_4	4	4	5		
u_5	2	4		5	2

Predicting the Missing Rating

- Predict rating $r_{u,i}$ as:

$$r_{u,i} = \bar{r}_u + \sum_{k \text{ users}} w_{u,k} (r_{k,i} - \bar{r}_k)$$

- where:

\bar{r}_u is the mean rating for user u

$w_{u,v}$ is the weight between users u and v

- If no rating of item i is available, prediction returns \bar{r}_u

Note: if weights are not in $[0,1]$ then need to **normalize** ratings

Rating Example

- Suppose we want to predict user 3's rating of item 1: $r_{3,1}$

	i_1	i_2	i_3	i_4	i_5
u_1	5		4	1	
u_2		3		3	
u_3	?	2	4	4	1
u_4	4	4	5		
u_5	2	4		5	2

Rating Example

- Suppose we want to predict user 3's rating of item 1: $r_{3,1}$
- Weighted average of ratings from k similar users: $r_{3,1} = \sum_{k \text{ users}} w_{3,k} r_{k,1}$
where:
 $w_{u,v}$ is weight b/w users u and v

	i_1	i_2	i_3	i_4	i_5
u_1	5		4	1	
u_2		3		3	
u_3	?	2	4	4	1
u_4	4	4	5		
u_5	2	4		5	2

Rating Example

- Suppose u_1, u_4, u_5 are equally similar
- We need: $r_{3,1} = \sum_{k \text{ users}} w_{3,k} r_{k,1}$
- Then: $r_{3,1} = w_{3,1}r_{1,1} + w_{3,4}r_{4,1} + w_{3,5}r_{5,1} = 3.67$

Higher than average

Lower than average

	i_1	i_2	i_3	i_4	i_5
u_1	5		4	1	
u_2		3		3	
u_3	?	2	4	4	1
u_4	4	4	5		
u_5	2	4		5	2

Rating Example

- Suppose we want to predict user 3's rating of item 1: $r_{3,1}$
- Weighted average of ratings from k similar users: $r_{3,1} = \sum_{k \text{ users}} w_{3,k} r_{k,1}$
where:
 $w_{u,v}$ is weight between users u and v
- Offset ratings by user's mean rating, \bar{r}_u :
 $r_{3,1} = \bar{r}_3 + \sum_{k \text{ users}} w_{3,k} (r_{k,1} - \bar{r}_k)$

Rating Example

- Suppose u_1, u_4, u_5 are equally similar
- We need: $r_{3,1} = \sum_{k \text{ users}} w_{3,k} r_{k,1}$
 $r_{3,1} = w_{3,1}r_{1,1} + w_{3,4}r_{4,1} + w_{3,5}r_{5,1} = 3.67$
- Offset by \bar{r}_u
- We need: $r_{3,1} = \bar{r}_3 + \sum_{k \text{ users}} w_{3,k} (r_{k,1} - \bar{r}_k)$
 $r_{3,1} = 2.75$
 $+w_{3,1}(1.67) + w_{3,4}(-0.33) + w_{3,5}(-1.25)$
 $= 2.75 + (0.02778) = 2.78$

	i_1	i_2	i_3	i_4	i_5
u_1	5		4	1	
u_2		3		3	
u_3	?	2	4	4	1
u_4	4	4	5		
u_5	2	4		5 ²	2

General Problems with CF

- Sparsity of data
 - Not every user will rate every item
 - Matrix R will have (many) missing values
 - Possible solutions: use additional sources, cluster users, cluster items, reduce matrix size
- Scalability
 - Too many users and too many items to maintain
 - Especially true with model based techniques
 - Possible solutions: cluster users/items, reduce matrix size

Practical Notes

- Many algorithms and implementations of CF
 - Some combine memory based and model based approaches
 - Doesn't consider metadata (e.g. author of book)
- Common recommendation systems use a hybrid approach
 - Combine content based and CF

Key Ideas

- Collaborative filtering (independent of content)
 - Similar users have similar preferences
- Representation:
 - Database of user-item preferences as a matrix
 - Finding similar users
- Algorithm:
 - Predict rating based on weighted average of similar users
- Known computational issues:
 - Sparsity
 - Scalability