MONASH University

MONASH
INFORMATION
TECHNOLOGY

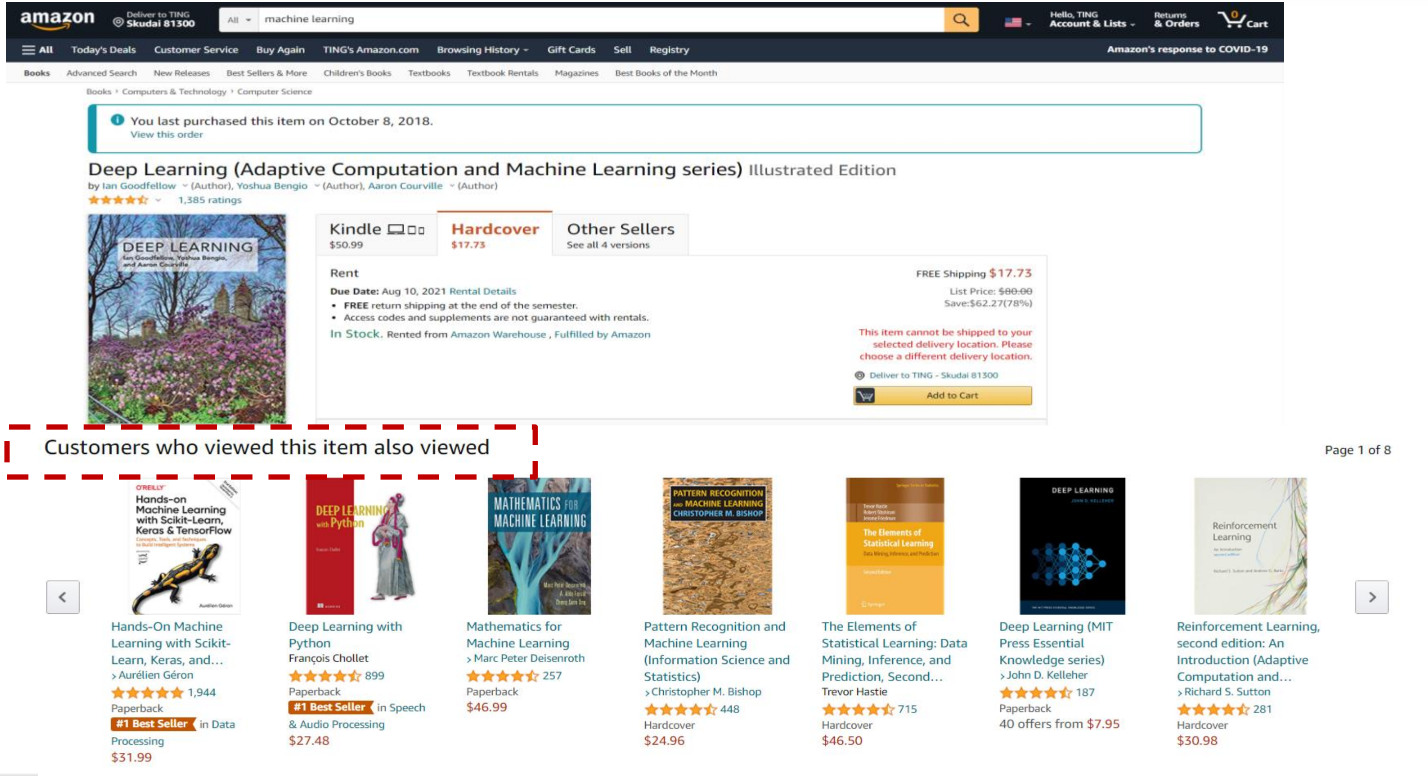
Machine Learning: Collaborative Filtering

GROUP OF EIGHT AUSTRALIA

Recommender System

- Goal: To identify items that we like
- It predicts user preference for a set of items based on users' past behaviour and feedback.
- This system is personalizing user's web experience.
  - E.g. telling what to buy (Amazon), which movies to watch (Netflix,TikTok), whom to be friends with (Facebook), which songs to listen (Spotify) etc.

Everyday Examples of Collaborative Filtering



MONASH University

User feedback

- **Explicit feedback:** Direct preferences given by the user to the item (e.g., user rating) 

- **Implicit feedback :** Indirect feedback, gathered from user behaviour (e.g. number of views, clicks, shares likes, visited/brought objects etc.).

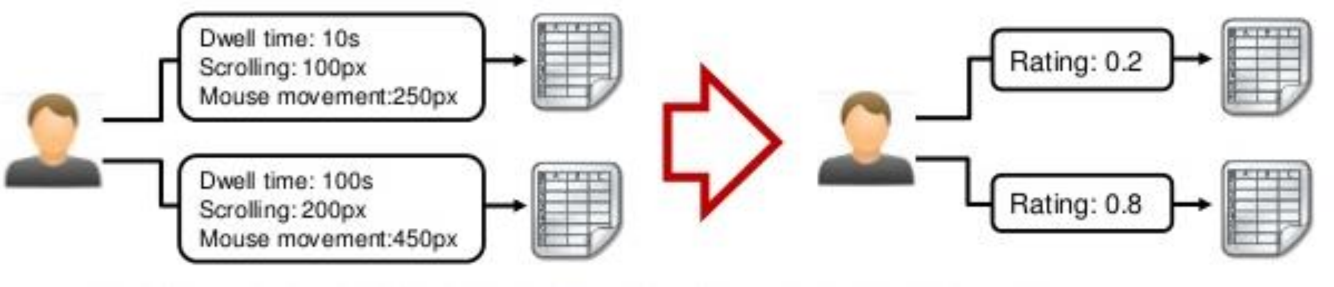
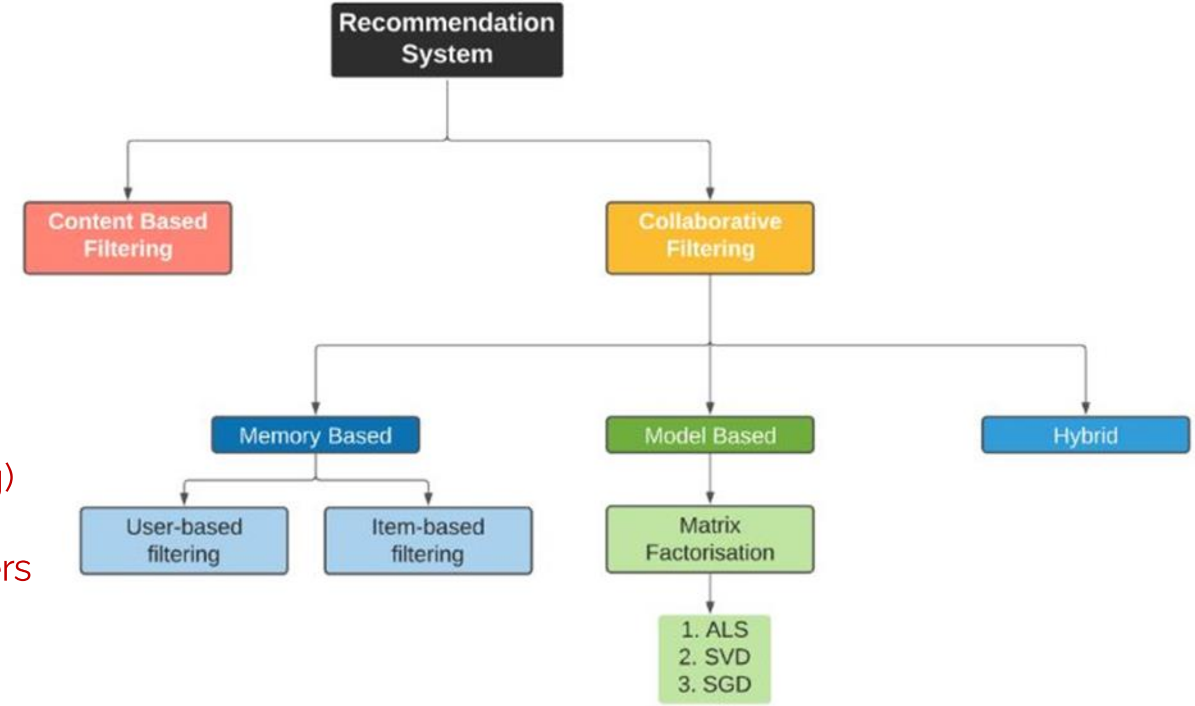


Image from: https://www.slideshare.net/LadislavPeska/using-the-context-of-user-feedback-in-recommender-systems

Recommender System

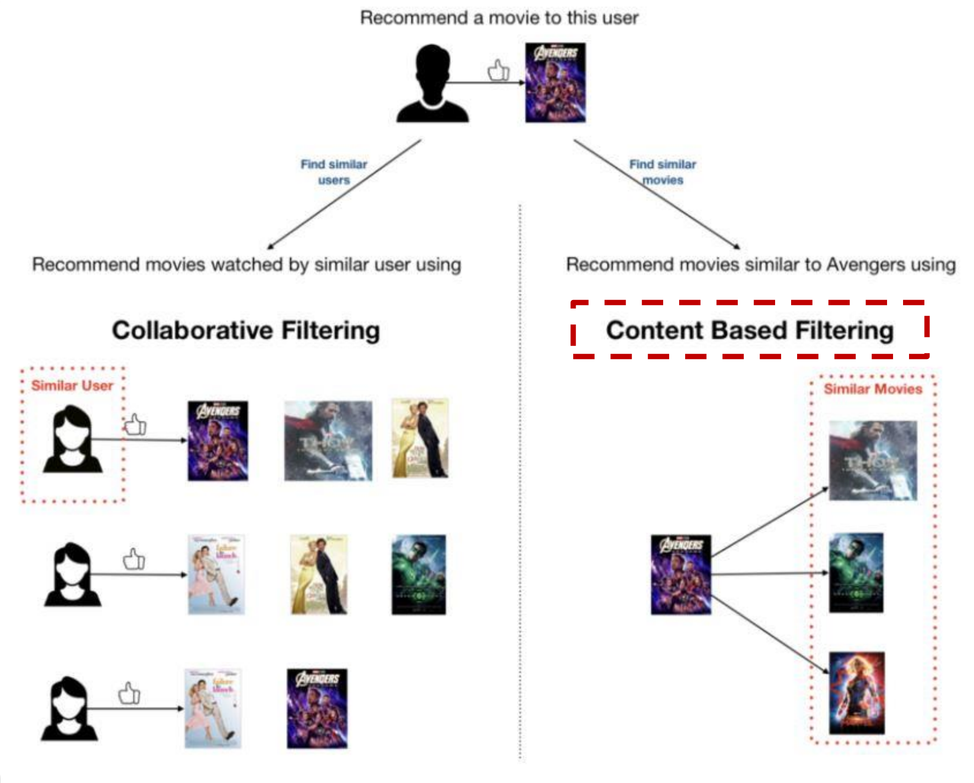
- Two common approaches:
  - Content based
  - **Collaborative Filtering**



Collaborative filtering is a method of making automatic predictions (filtering) about the interests of a user by collecting preferences from many users (collaborating)

https://towardsdatascience.com/build-recommendation-system-with-pyspark-using-alternating-least-squares-als-matrix-factorisation-ebe1ad2e7679

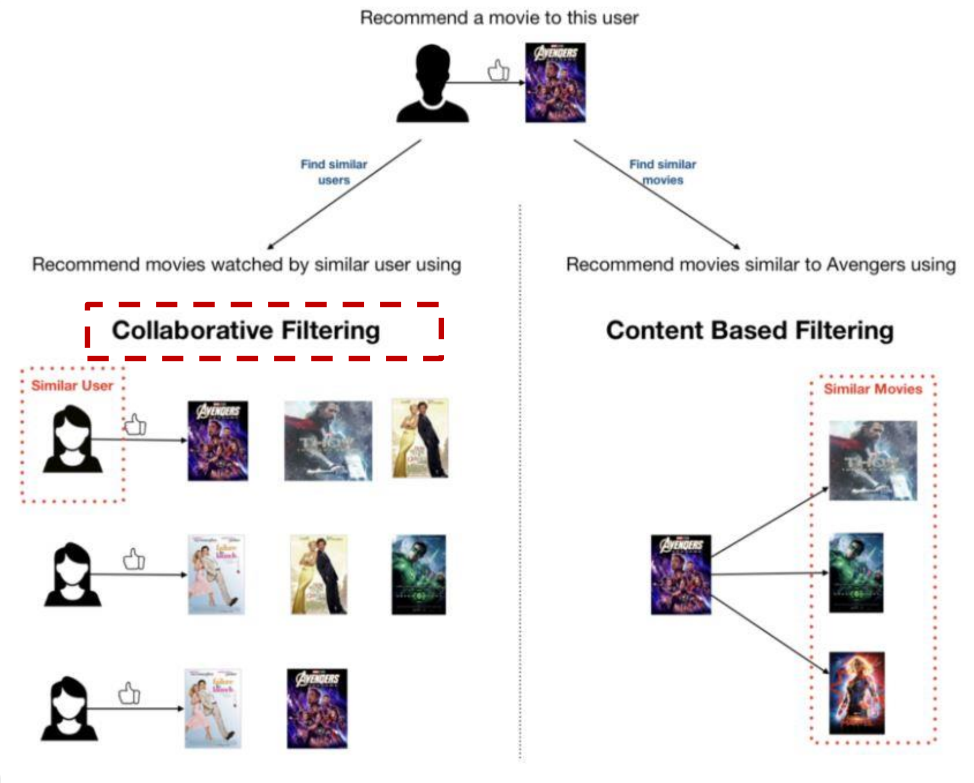
Content based Filtering



- ❑ Main Idea: Recommend items similar to the items previously liked by the user
- ❑ Example:
  - ➢ **Movie recommendations:** Recommend movies with the same actor(s), director, and genre.
  - ➢ **Websites, blogs, news:** Recommend other sites with "similar" content
- ❑ It requires a sufficient amount of information about the content of the items

https://towardsdatascience.com/build-recommendation-system-with-pyspark-using-alternating-least-squares-als-matrix-factorisation-ebe1ad2e7679

Collaborative Filtering (CF)

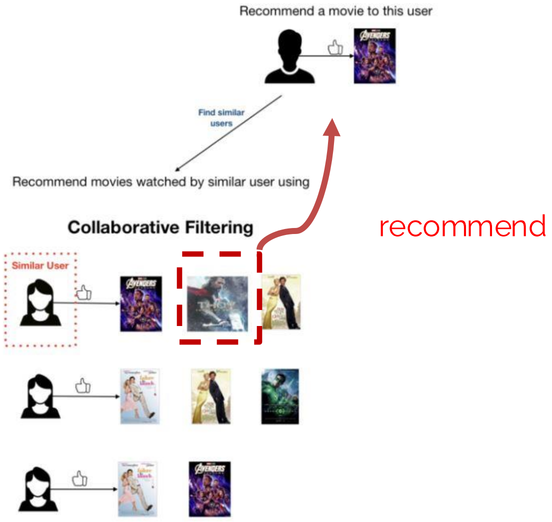


- ❑ Main Idea: Use input/behavior of all previous users to make future recommendation
- ❑ Recommend items to a user based on the items liked by another set of users whose rating pattern (like & dislike) are similar to the user
- ❑ Example:
  - ➢ **Movie recommendations:** Recommend movies watched by similar user
- ❑ It's domain-free - It does not look at the details of content, only looks at who is rating the content & what is the rating
- ❑ Make use of **similarity between users** past feedback/preferences (***user*-based CF**)

Collaborative Filtering

- **Collaborative filtering** is a method to predict a user's rating on particular item by comparing one user to all other users.

- *For example*:
  To predict *PersonA* rating on a particular item,
  - Compute the similarity between *PersonA* with all users.
  - Find the top users who are most similar to the *PersonA*
  - Predict ratings of *PersonA* on the item based on the rating of similar users.



Why Collaborative Filtering?

- It benefits from large user bases.
- It's flexible across different domains.
- It produces the level of recommendations required.
- It can capture more nuance around items.

Collaborative Filtering Process

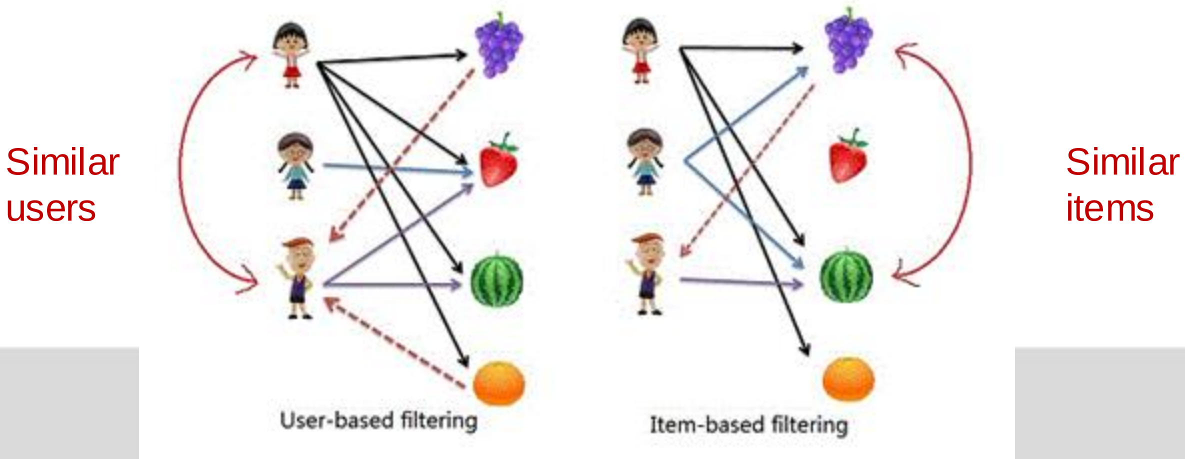
Data Collection -> Data Processing -> Calculate Referrals -> Derive Results

- **Data collection**: Collecting user behaviour and associated data items
- **Data processing**: Processing the collected data
- **Recommendation Calculation**: The recommended calculation method used to calculate referrals
- **Derive the result**: Extract the similarity, sort it, and extract the top N to complete

Memory-based Collaborative Filtering

Memory-based (neighborhood approach) CF recommends items by finding similarities between users or items

- **User-based CF**: To recommend items to a user based on another set of users with a similar rating pattern to the user
- **Item-based CF**: To recommend items with the most similarity with the user's other favourite items.

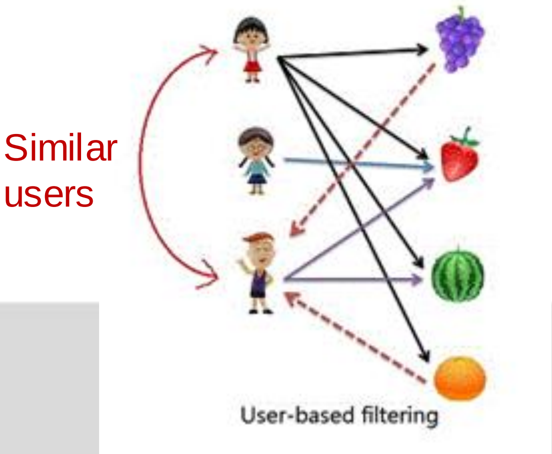


Collaborative Filtering – User based

It calculates the **similarity between users** to make implicit recommendation.

Steps:

1. Calculate the similarity between *PersonA* and all other users.
2. Predict the ratings of items for *PersonA* based on similar users.
3. Select top-N rated items for *PersonA*.



MONASH University

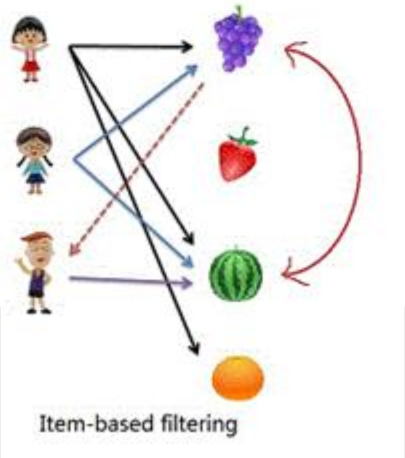
Collaborative Filtering – Item based

It calculates the **similarity between items** to make implicit recommendation

.

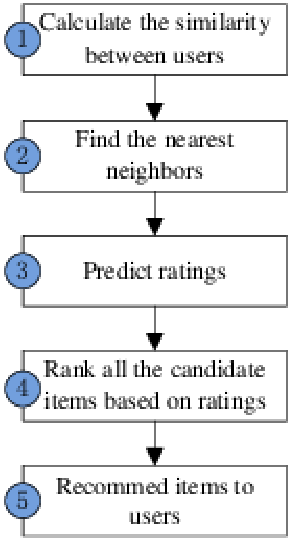
Steps:

1. Calculate the similarity between any two items to get item-item similarity matrix.
2. Predict the ratings of items for *PersonA* based on similar items.
3. Select top-N rated items for *PersonA.*

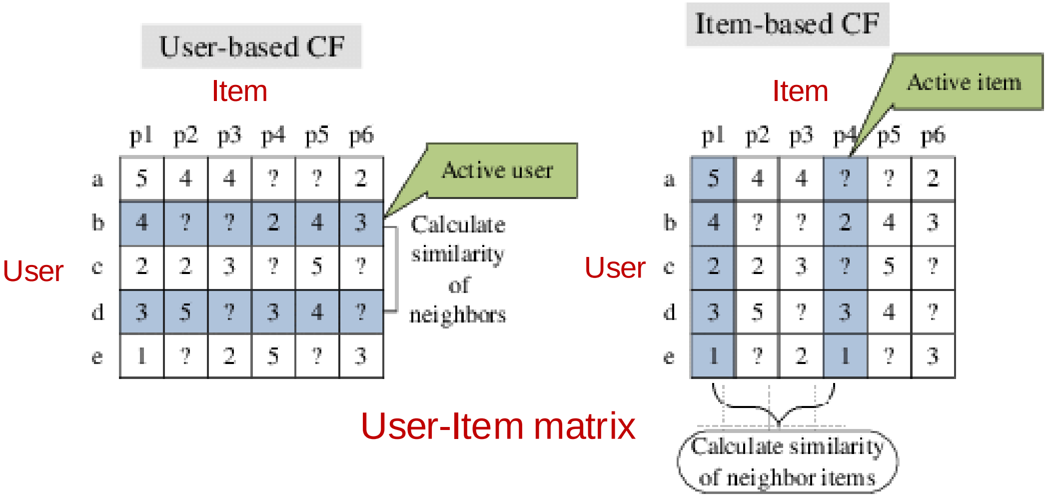


Similar items

User-Based Vs Item-Based CF



The procedure of memory-based collaborative filtering RS



	p1	p2	p3	p4	p5	p6
a	5	4	4	?	?	2
b	4	?	?	2	4	3
c	2	2	3	?	5	?
d	3	5	?	3	4	?
e	1	?	2	5	?	3

For both user-based or item-based CF, the computation of similarity is based on the preference for the item.

The features used to calculate similarity can be user's purchase frequency, user's preference rating, number of product clicks, or a combination of all of these.

Adopted from paper 'A Survey of Collaborative Filtering-Based Recommender Systems: From Traditional Methods to Hybrid Methods Based on Social Networks'

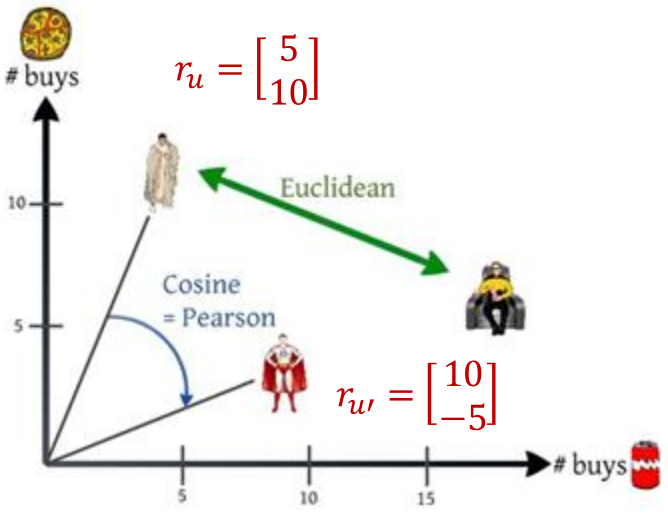
How to get similarity?

$$sim(u,u') = cos(\theta) = \frac{\mathbf{r}_u \cdot \mathbf{r}_{u'}}{\|\mathbf{r}_u\| \|\mathbf{r}_{u'}\|}$$

$\|.\|$ = Norm (magnitude) of vector

- **Cosine similarity:**
  - Measures the cosine of the vector angle between ratings of two users
  - If cosine value is 1, two users are completely similar in their preference,
  - if cosine value is -1, they are completely dissimilar
  - Similar to Pearson correlation, which measures correlation between two users

- **Euclidean distance:**
  - Measures distance in rating/preference between two users
  - If the distance is small, two users have a similar preference (i.e., the similarity is high).



Collaboration Filtering (User-based): Walkthrough Example

User-Item matrix

Name	Star Trek	Star wars	Superman	Batman	Hulk
Harry	4	2	?	5	4
John	5	3	4	?	3
Rob	3	?	4	4	3

Aim: Recommend top-2 movies to Harry

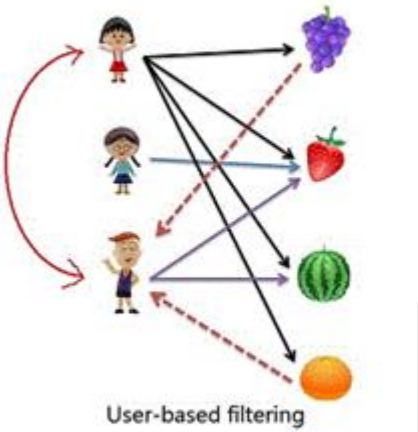
Collaborative Filtering – User based(same as slide 17)

It calculates the **similarity between users** to make implicit recommendation.

Steps:

1. Calculate the similarity between *PersonA* and all other users.
2. Predict the ratings of items for *PersonA* based on similar users.
3. Select top-N rated items for *PersonA*.

Similar users



Collaboration Filtering: Walkthrough Example (user-based)

Step 1: Calculate the similarity between Harry and all other users

Name	Star Trek	Star wars	Superman	Batman	Hulk
Harry	4	2	?	5	4
John	5	3	4	?	3
Rob	3	?	4	4	3

Cosine similarity

$$sim(u,u') = cos(\theta) = \frac{\mathbf{r}_u \cdot \mathbf{r}_{u'}}{\|\mathbf{r}_u\| \|\mathbf{r}_{u'}\|} = \sum_i \frac{r_{ui} r_{u'i}}{\sqrt{\sum_i r_{ui}^2} \sqrt{\sum_i r_{u'i}^2}}$$

r_{ui} - value of ratings user *u* gives to item *i*

? – missing values

Collaboration Filtering: Walkthrough Example (user-based)

Step 1: Calculate the similarity between Harry and all other users

Name	Star Trek	Star wars	Superman	Batman	Hulk
Harry	4	2	?	5	4
John	5	3	4	?	3
Rob	3	?	4	4	3

$$sim(u,u') = \sum_i \frac{r_{ui} r_{u'i}}{\sqrt{\sum_i r_{ui}^2}\sqrt{\sum_i r_{u'i}^2}}$$

r_{ui} - value of ratings user *u* gives to item *i*

Cosine similarity

$$\text{Sim(Harry, John)} = \frac{(4*5)+(2*3)+(4*3)}{sqrt(4^2+2^2+4^2)*sqrt(5^2+3^2+3^2)}$$

$$= 0.97$$

$$\text{Sim(Harry, Rob)} = \frac{(4*3)+(5*4)+(4*3)}{sqrt(4^2+5^2+4^2)*sqrt(3^2+4^2+3^2)}$$

$$= 1.00$$

MONASH University

$$r_{Harry} = \begin{bmatrix}4\\2\\4\end{bmatrix} \quad r_{John} = \begin{bmatrix}5\\3\\3\end{bmatrix}$$

$$r_{Harry} = \begin{bmatrix}4\\5\\4\end{bmatrix} \quad r_{Rob} = \begin{bmatrix}3\\4\\3\end{bmatrix}$$

Collaboration Filtering: Walkthrough Example (user-based)

Step 2: Predict the ratings of movies for Harry

Name	Star Trek	Star wars	Superman	Batman	Hulk
Harry	4	2	?	5	4
John	5	3	4	?	3
Rob	3	?	4	4	3

Predicted rating is calculated based on aggregation of some similar users' rating of the item

$$r_{u,i} = k \sum_{u' \in U} \mathrm{simil}(u, u') r_{u',i}$$

U – set of similar users {John, Rob}

with normalising factor

$$k = 1 / \sum_{u' \in U} |\mathrm{simil}(u, u')|,$$

Calculate *k* as a normalising factor $k = \frac{1}{(0.97+1)} = 0.51$

R(Harry, Superman) = k*$((sim(Harry,\ John) * R(John, Superman)) + (sim(Harry, Rob) * R(Rob, Superman)))$

$$= 0.51((0.97 * 4) + (1 * 4)) = 4.02$$

Collaboration Filtering: Walkthrough Example (user-based)

Step 3: Select top-2 rated movies for Harry

Name	Star Trek	Star wars	Superman	Batman	Hulk
Harry	4	2	***4.02***	5	4
John	5	3	4	?	3
Rob	3	?	4	4	3

Top-2(Harry, movies)= Batman, Superman

MONASH University

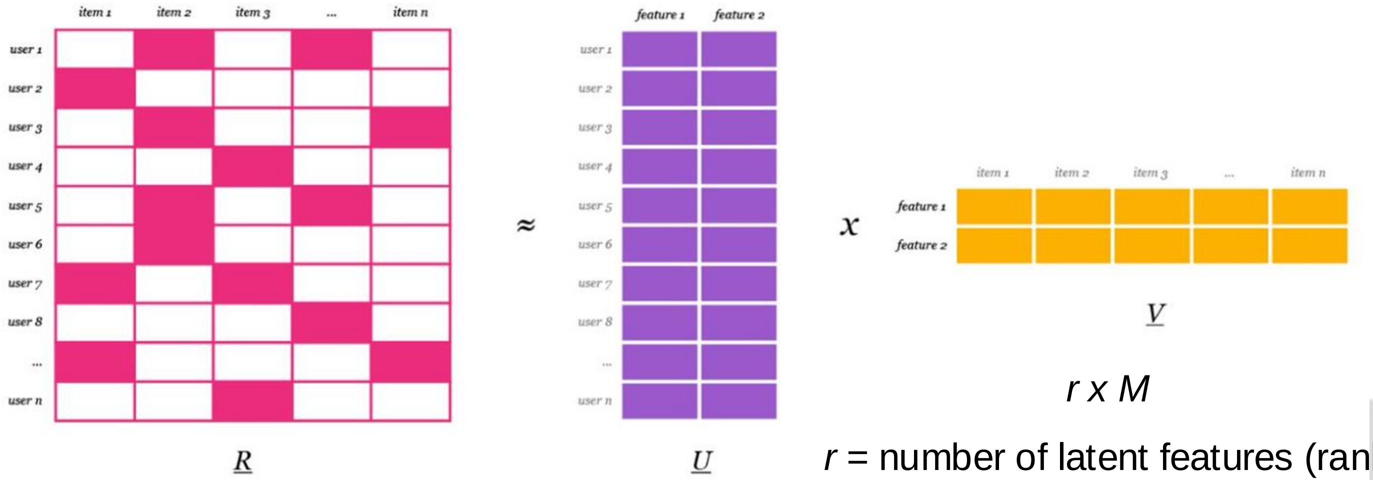
Model-based Collaborative Filtering

- ❑ Latent factor model-based CF learns the (latent) user and item profiles through **matrix factorisation**
- ❑ **Matrix factorisation:** Factor a large matrix into some smaller representation of the original matrix through alternating least squares. The product of lower dimensional matrices equals the original one
- ❑ Example: Factor the rating matrix **R** into user matrix **U** and item matrix **V**

User-Item matrix

Latent features (or factors)

N = # items
M = # users
r = # latent features



r x M

r = number of latent features (rank)

N x M

N x r

Alternating least squares

- ALS method aims to estimate the user and item factor matrices (**U & V**) such that their product will approximate the original rating matrix **R**.

- This is achieved by minimizing root mean square error (RMSE) between the original ratings **R** and the predicted values $\hat{\mathbf{R}}$

ALS Procedure:

Optimizing alternately to find U, V

- Randomly initialize U and V
- Iterating the following steps:
  - Fixing U → Optimizing V
  - Fixing V → Optimizing U
- Each iteration will minimize the square errors

≈

●	◆
1	.1
-1	0
.2	-1
.1	1

✖

■	.9	-1	1	1	-.9
▲	-.2	-.8	-1	.9	1

≈

.88	-1.08	0.9	1.09	-0.8
-0.9	1.0	-1.0	-1.0	0.9
0.38	0.6	1.2	-0.7	-1.18
-0.11	-0.9	-0.9	1.0	0.91

Rating matrix, **R** User factor matrix, **U** Item factor matrix, **V** Predicted rating matrix $\hat{\boldsymbol{R}}$

■ arthouse <-> blockbuster
▲ children's <-> adult's
● preference for arthouse <-> blockbuster
◆ preference for children's <-> adult's

https://developers.google.com/machine-learning/recommendation/collaborative/matrix

Collaborative filtering in Spark

- `spark.ml` currently supports model-based collaborative filtering, in which users and products are described by a small set of latent factors that can be used to predict missing entries.
- `spark.ml` uses the Alternating Least Squares (ALS) algorithm to learn these latent factors.

Challenges of Collaborative Filtering

- Cold Start Problem
- Data sparsity can affect the quality of user-based recommenders
- Scaling can be challenging for growing datasets as the complexity can become too large. Item-based recommenders are faster than user-based when the dataset is large.
- You might observe that the recommendations tend to be already popular, and the items from the long tail section might get ignored.