

**TOPIC:**  
**RECOMMENDER SYSTEMS**



**CT102**  
**Information**  
**Systems**

# DEFINITION:

## RECOMMENDER SYSTEM

A recommender system provides suggestions for **items** to a **user** to support *decision-making processes*, such as:

- what items to buy: computers, phones, cameras, appliances, cars, etc.
- what music to listen to or what videos to watch
- what books or news articles to read
- what films to watch
- where to stay; where to eat
- who to connect to or link to

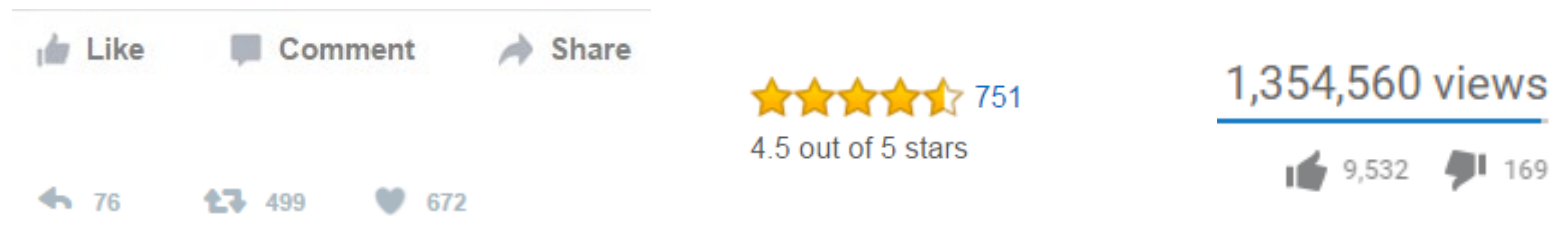
Generally the systems use the idea of **preferences** and/or **ratings** to make recommendations or use the existing **social links** between people.

# PREFERENCES AND RATINGS

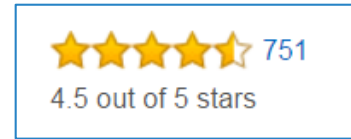


**Preference:** a greater liking for one alternative (or thing) over another or other thing (Oxford English Dictionary), e.g. choosing to watch one particular video over others in a list

**Rating:** a measurement of how good or liked something is, e.g. likes, stars, actual rating value.



# Characteristics of preferences and ratings



**Preference** data and **rating** data is viewed as an *indicator of user's likes and tastes*

Can be gathered **implicitly** or **explicitly**

Explicitly gathered using some meaningful icon (heart, thumbs up, stars, number, etc.)

Implicitly gathered based on a person's actions (viewing, listening, etc.)

# EXAMPLE:

Preference question:

Do you like the series  
“Succession”?

Explicit Preference: ?

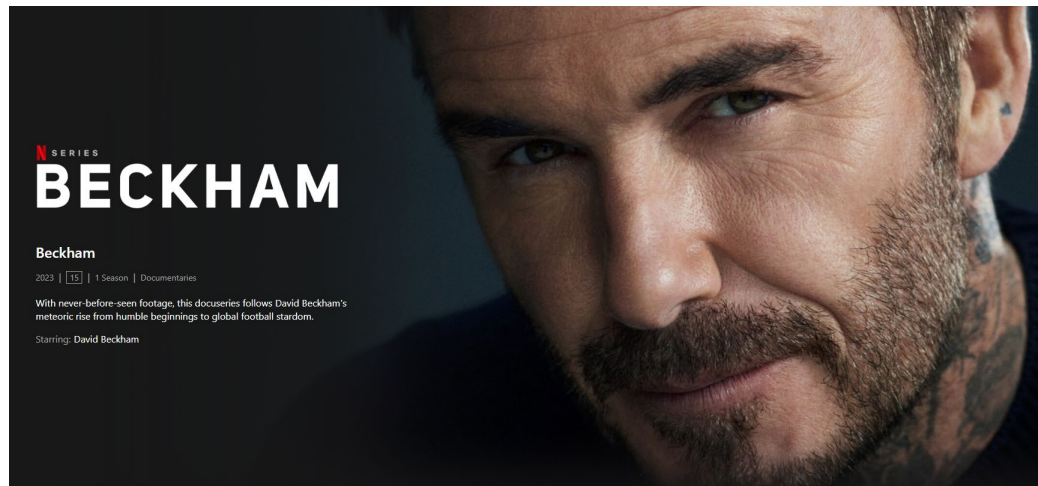
Implicit Preference: ?



# GATHERING EXPLICIT RATINGS

System will ask user to explicitly indicate “like” or “rate” or to give a value for one or more items.

Usually a numeric representation of the rating is stored if this is not directly provided by a person (i.e. a heart or star is stored as a number)



# IMPLICIT RATINGS AND PREFERENCES

Try infer user's rating (opinion) based on actions taken by the user.

e.g.,

- Watching video and sharing videos
- Number of times listening to a mp3 and/or adding it to play list
- Posting a comment or liking something
- Purchasing something
- Time spent reading an item
- Following a new account or user

Usually **numeric representations** of preferences are stored and these may be **weighted** to indicate importance and/or time

# SCORES

Usually in a range:

- Binary: 0, 1 (0 dislike; 1 like)
- -1, 0, 1 (-1 dislike; 0 neutral; 1 like)
- 1, 2, 3, 4, 5 (1 strong dislike; 5 strong like)
- 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 (1 strong dislike; 10 strong like)

## Notes:

- Larger scales don't necessarily give better indicator of a person's likes and tastes
- People tend to use different “portions” of the scale



# Numeric rating and preference data can be viewed as a matrix ...

[illegible]

# PREFERENCES VS RATINGS VS REVIEWS?

Often a combination or all types of feedback are used

- **Reviews** have content (**text**) and may be more nuanced and require different techniques to process (sentiment analysis, stance analysis for example) and are not generally used in recommendation systems.
- Nowadays the idea of preferences and ratings are often combined
- **Implicit ratings** and preferences are generally more “**noisy**” than explicit ratings and preferences but are easier to obtain, i.e. are a by-product of using a particular app and require no extra effort by a person.
- We often don’t bother giving explicit ratings and reviews when asked (e.g., after purchasing a product)
- The concept of “**preferences**” is a complex one – and depends on many factors – not least the list you are given to choose from.
- Most systems are not very nuanced in terms of the concept of “preferences” – for example, the concept of “**meta-preferences**” – the choice of a dessert might be influenced by the meta-preference to be healthy or avoid sugar

# MAIN DATA USED IN RECOMMENDATION SYSTEMS

- Unique ID for each users
- Unique ID for each item (whatever that might be, song, album, book, video, etc.)
- Ratings/Preference numbers (aggregated) for users and for items
- Content of items (generally represented by text even if the item is non-text, e.g. video, music, etc.)
- Content associated with user (textual list of preferences, mood, location, age – provided by user or gathered by system based on implicit actions)

# USING THE PREFERENCE DATA?

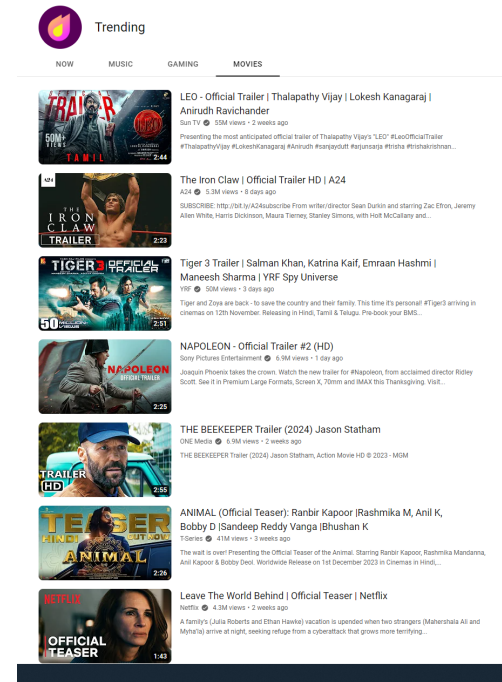
Can be used for personalised and non-personalised **recommendations**

## WHAT APPROACHES ARE USED?

- Database
- Statistical (correlation)
- Matrix factorization
- Machine Learning , particularly Reinforcement learning and Clustering Approaches

# NON-PERSONALISED RECOMMENDATIONS

- Lists of items which can be generated for everyone.
- Often based on popularity, newness and velocity.
- **Popularity:** number of views, shares, likes, hashtag use, etc.
- **Newness:** recent content generally ranked higher.
- **Velocity:** The rate of the popularity growth (e.g., over minutes, hours, days).
- People often want to see/hear/know about what everyone else is talking about.
- Also useful for new users who have not given any indication of preferences yet and to offer diversity to users (something different to their personalised recommendations)



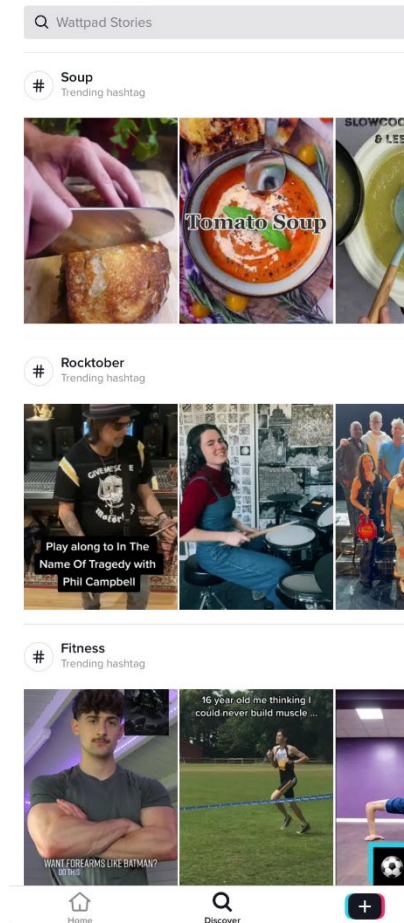
# IMPLEMENTATION

Very easy to implement and update.

Keep an ordered list of the content IDs based on the weighting of popularity + newness + velocity

The popularity measure should be a **strong indicator** such as views, plays, likes, hashtags, etc.

These lists might be kept for different genres and displayed per genre, as well as the overall most popular across all genres being displayed for everyone.



# PERSONALISED RECOMMENDATIONS

Based on **User profile** and/or **Item profile** information which is used as an *indicator* of a person's likes/tastes/preferences.

**Recommendation task can be stated as:** for some *active* person **A** recommend personalised content not yet liked/seen by person **A** based on:

- I. the preferences of person **A** (Content-based recommendation).
- II. the preferences of other people with whom person **A** shares some preferences in common (Collaborative-based recommendation).

## i) The preferences of a user: Content-based recommendation

Content-based recommendation recommends items based on items users have indicated a preference for in the past where the similarity between items is based on the **content** of those items.



## More on Attributes

- The attributes used are dependent on the domain and ideally are **well-defined** (structured/using tags) so that meaningful matching can be performed using a query language.  
i.e., so that a preference indicator can be associated with a genre, year, style, playlist, age category, creator name, etc.
- In the absence of the attributes being well defined (e.g., description) the main **descriptive terms** from the content are associated with the user profile and a more generic matching approach is used.
- In both cases, a higher weighting would be given to more recent preference indicators.
- Larger apps, like YouTube, Netflix, Amazon, etc., would use a combination of approaches.

# CONTENT BASED RECOMMENDATION APPROACHES

1. Database approach (Relational or Semantic Web).
2. Can represent attributes as weighted terms in vectors and use **Euclidean dot product** to get recommendations.
3. Machine learning approaches

*Going from simple and quick matching to more complex and time intensive matching*


# EXAMPLE: USING THE VECTOR APPROACH

(Taken from: [buildingrecommenders.wordpress.com](http://buildingrecommenders.wordpress.com))

**Given:** the titles of six books and binary user preference data (based on ratings or purchases)

**Assume:** the active user **A** has already purchased the book “Introduction to Recommender Systems” and this is stored as positive indicator of the user’s preferences for the book

**Task:** use a content approach, with only titles available, to find which of the other 5 books user **A** should be recommended



Introduction to Recommender Systems
Machine learning Paradigms
Social Network-based Recommender Systems
Learning Spark
Recommender Systems Handbook
Recommender Systems and the Social Web

# CONTENT BASED APPROACH



Introduction to Recommender Systems
Machine learning Paradigms
Social Network-based Recommender Systems
Learning Spark
Recommender Systems Handbook
Recommender Systems and the Social Web

- Assume we only have book titles available to use (attribute = title)

Using:

- Query is title of book user **A** has indicated a preference for (“Introduction to Recommender Systems”)
- This will be compared to all other books to find those most similar.

# STEPS:



Introduction to Recommender Systems
Machine learning Paradigms
Social Network-based Recommender Systems
Learning Spark
Recommender Systems Handbook
Recommender Systems and the Social Web

- Remove stop words from each title (to, and, the)
- As typically book titles (or other titles) will not have repeated words the weighting used will be Boolean (1 = present, 0 = absent).
- Represent each word in a vector with respect to the weighting.
- Start calculating similarity.

# VECTOR REPRESENTATION OF BOOKS



introduction	1					
recommender	1		1		1	1
systems	1		1		1	1
machine		1				
learning		1		1		
paradigms		1				
social			1			1
network-based			1			
spark				1		
handbook					1	
web						1



Introduction to Recommender Systems
Machine learning Paradigms
Social Network-based Recommender Systems
Learning Spark
Recommender Systems Handbook
Recommender Systems and the Social Web

# SIMILARITY:



Introduction to Recommender Systems



Social Network-based Recommender Systems

$\langle 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0 \rangle$

$\langle 0, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0 \rangle$

$$\frac{2}{(\sqrt{3} \times \sqrt{4})} = 0.5773502691896258$$

= 0.58



introduction	1					
recommender	1		1		1	1
systems	1		1		1	1
machine		1				
learning		1		1		
paradigms		1				
social			1			1
network-based			1			
spark				1		
handbook					1	
web						1

# SIMILARITY:



Introduction to Recommender Systems



Recommender Systems Handbook

$\langle 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0 \rangle$

$\langle 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0 \rangle$

$$\frac{2}{(\sqrt{3} \times \sqrt{3})} = \frac{2}{3} = 0.6666666666666667$$

= 0.67



introduction	1					
recommender	1		1		1	1
systems	1		1		1	1
machine		1				
learning		1		1		
paradigms		1				
social			1			1
network-based			1			
spark				1		
handbook					1	
web						1



# USING COSINE SIMILARITY TO FIND SIMILARITY OF BOOKS TO ACTIVE USER'S BOOK:

Similarity of all books to “Introduction to Recommender Systems”:



# CAN PRE-CALCULATE THE SIMILARITY OF ALL BOOKS TO EACH OTHER:



# HOW WOULD THIS WORK MORE GENERALLY?

This would be a good **general approach** for free-text (**unstructured**) attributes such as “description”.

In reality, for books other than academic books, attributes such as genre and author would be more useful for matching and would be much easier to implement with a **database approach**.

It is important to note though that for either approach the similarity of items to each other can be **pre-calculated** thus when/if a user indicates a preference for an item those items which are similar can be retrieved very, very quickly.

# ADVANTAGES AND DISADVANTAGES OF CONTENT BASED APPROACH

- No cold start problem - don't need many past user or item preferences.
- No popularity bias – can recommend rare items as long as they match the content of an item the user has indicated a preference for.
- Attributes can be weighted to keep user profiles current and fresh.
- Can provide an explanation of the recommendation (“because you bought this ... )
- Item content (attributes) need to be machine readable and meaningful (e.g., genres, authors, text title, abstract text, etc.)
- “Stereotyping” possible – only get items similar to what you have already liked.
- Serendipity difficult – small chance of getting something unexpected or outside of your “filter bubble” (but un-personalised recommendations can help here)

## SUMMARY OF THESE STRENGTHS AND WEAKNESSES FROM A RECOMMENDATION PERSPECTIVE ...

Content based approaches are good at **Exploitation** - content similar to that for which a person has already expressed a preference are recommended ... but the approaches are not as good at **Exploration** - content which doesn't match with content of what the person has previously indicated a preference for will not be recommended.

Approach (ii) .... the preferences of other people with whom person *A* shares some preferences in common

## Collaborative-based recommendation

Does not use any content data.

Based on the intuition that “people’s preferences are correlated”.


Tries to mimic “word of mouth” recommendation that often happens between people.

Uses **preference** data to form similarity between **people** and between **content items**.

# EXAMPLE

Adapted from one of the earliest papers on collaborative filtering by Resnick et al.,

**Given:** Ratings by 4 users for 6 movies (where movies are represented by IDs). Ratings are in the range 1-5 (1="strong dislike"; 5 = "strong like")

Movie #	Ken	Lee	Meg	Nan
1	1	4	2	2
2	5	2	4	4
3			3	
4	2	5		5
5	4	1		1
6		2	5	


**Sample Goal:**

Which users have similar tastes?

# EXAMPLE

Adapted from one of the earliest papers on collaborative filtering by Resnick et al.

**Given:** Ratings by 4 users for 6 movies (where movies are represented by IDs). Ratings are in the range 1-5 (1="strong dislike"; 5 = "strong like")

<i>Movie #</i>	<i>Ken Lee</i>	<i>Meg</i>	<i>Nan</i>
1	1	2	2
2	5	4	4
3		3	
4	2		5
5	4		1
6		5	

**Sample Goals:**


Which users have similar tastes?



# EXAMPLE

Adapted from one of the earliest papers on collaborative filtering by Resnick et al.

**Given:** Ratings by 4 users for 6 movies (where movies are represented by IDs). Ratings are in the range 1-5 (1="strong dislike"; 5 = "strong like")

Movie #	Ken Lee	Meg	Nan
1	1	4	2
2	5	2	4
3		3	
4	2	5	5
5	4	1	1
6		2	5

## Sample Goals:

Which users have similar tastes?

Find whether Ken is interested in (likes) movie 6

Will Nan like movie 6?

# ADVANTAGES OVER CONTENT FILTERING:

- Support for recommendation of items where content cannot be analysed easily in an automated manner
- Ability to take issues of taste into account that may be difficult to represent with the content-based approach
- Serendipitous recommendations possible – can recommend items without the user needing to have seen items similar to the item previously or without the user needing to state an explicit preference for that type of item (e.g. genre of music, etc.)

# APPLICATIONS

- The approach has been successful in a number of domains ....
- Especially when a person can easily consume content (e.g. Netflix, Amazon Prime, Spotify etc.) but may not want to invest time and effort in explicitly specifying their preferences.
- Most well-known examples involve recommending music, videos, books, streaming, e-commerce and social media domains.
- Often combined with other approaches (content and popularity) as they compliment each other in terms of what they each offer.

# MAIN DATA USED IN COLLABORATIVE FILTERING APPROACH:

- Users (represented by an ID)
- Items (represented by an ID)
- Ratings/Preference scores of users for items

# MATRIX DATA REPRESENTATION

In this example,

- Values are in the range [1-5]
- 0 indicates no value
- Rows represent users
- Columns represent items
- Generally, unless storing a very small amount of data, would not store in this format (would only store non-zero data)

```
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 3 3
0 0 4 3 5 0 3 4 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0
0 5 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 4 0 5 0 0 0 0 0 0 0 0 0 2 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0
0 0 4 3 4 0 0 0 0 0 0 4 4 0 0 0 0 0
4 0 0 2 5 0 0 0 0 4 3 0 0 0 0 0 0 0
0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

# FEATURES OF THE DATA

Typically, as can be seen from the matrix representation, there are many 0 entries.

This is referred to as matrix *sparsity* .... i.e., there are many items for which we do not have a value (no preference or rating from the user).

For many real sets of data on average only 1% of items will have rating/preference scores.

This makes sense given the number of items which exist but it can make it difficult to find similar items and users if there is no “overlap” between user/item preferences.

# STORAGE OF SAMPLE DATA

Usually stored as a triple:

- userID, itemID, preferenceValue

- e.g.,

196, 242, 3

186, 302, 3

22, 377, 1

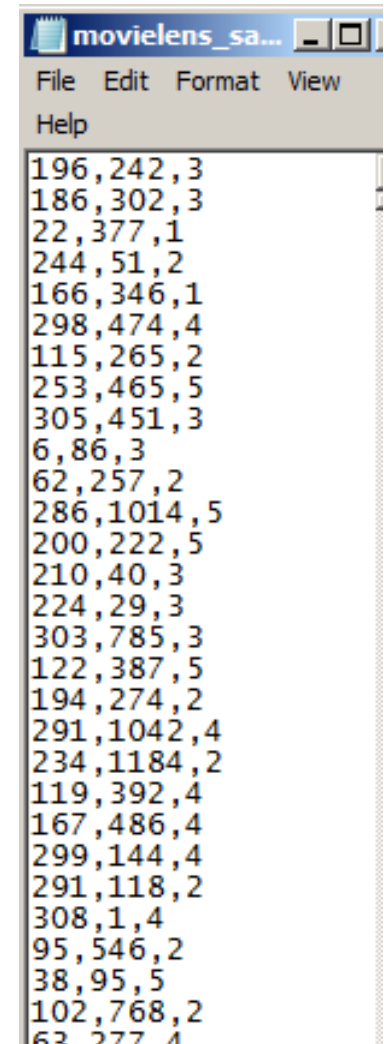
244, 51, 2

166, 346, 1

298, 474, 4

115, 265, 2

253, 465, 5



# COLLABORATIVE FILTERING APPROACHES

- Numerous approaches have been researched and tested.
- Modern approaches often use some form of **machine learning** technique.
- A common technique which is easy to understand is a statistical **user-user** or **item-item** correlation, neighbour-based approach which finds groups of similar users or similar items.
- For practical applications, it is more useful to find and store similar items.



# STEPS IN A PEARSON CORRELATION NEIGHBOUR-BASED APPROACH

1. **Calculate user-user or item-item similarity:** Find how similar each user/item is to every other user/item. Many approaches possible – mostly using maths techniques, IR techniques (vector similarity) or machine learning techniques. We will look at a popular and simple approach using the **Pearson correlation** formula.
2. **Select Neighbourhood:** Form groups or neighbourhoods of users/items who are similar based on the calculations in part 1.
3. **Generate Recommendation:** In each group, can make recommendations based on other similar users/items

**PEARSON CORRELATION:** Weighted average of deviations from the neighbours' mean is calculated

$$w_{a,u} = \frac{\sum_{i=1}^m (r_{a,i} - \bar{r}_a) \times (r_{u,i} - \bar{r}_u)}{\sqrt{\sum_{i=1}^m (r_{u,i} - \bar{r}_u)^2} \times \sqrt{\sum_{i=1}^m (r_{a,i} - \bar{r}_a)^2}}$$

where for m items:

$r_{a,i}$  is rating of user  $a$  for item  $i$

$\bar{r}_a$  is the average rating given by user  $a$

$r_{u,i}$  is rating of user  $u$  for item  $i$

$\bar{r}_u$  is the average rating given by user  $u$

Notes: The result is in the range  $[-1, 1]$ .

The correlation can be computed only if there are common items rated by both the users

## 2. SELECT NEIGHBOURHOOD:

*Some approaches:*

**Correlation thresholding:** where all users with similarity above a certain threshold are selected. Threshold usually:

- 0 if only want to look at positive correlations
- Usually slightly over 0, e.g. 0.1

**Best-N correlations:** where the N neighbours who have the highest similarity are chosen

# SIZE OF NEIGHBOURHOOD?

Is there a best number of neighbours to pick?

If a large number of neighbours are picked:

- Potentially get *low precision* predictions

If a small number of neighbours are picked?

- Potentially get few or no predictions

### 3. GENERATE RECOMMENDATION:

For some user (the active user):

- make recommendations based on what the neighbours have rated but the active user has not rated
- often the *weighted average* of neighbor's ratings are used

**3. GENERATE RECOMMENDATION:** To give a prediction,  $P$ , for an item  $i$  for an active user  $a$  use weighted average approach:

$$P_{a,i} = \bar{r}_a + \frac{\sum_{u=1}^n (r_{u,i} - \bar{r}_u) \times w_{a,u}}{\sum_{u=1}^n w_{a,u}}$$

$\bar{r}_a$  is the average rating given by active user  $a$

$r_{u,i}$  is rating of user  $u$  for item  $i$

$\bar{r}_u$  is the average rating given by neighbour  $u$

$w_{a,u}$  is the similarity between user  $u$  and  $a$

## EXAMPLE 1 AGAIN

Movie#	Ken	Lee	Meg	Nan
1	1	4	2	2
2	5	2	4	4
3			3	
4	2	5		5
5	4	1		1
6	?	2	5	

*Task:*

Find recommendation for movie 6 for Ken

$$w_{a,u} = \frac{\sum_{i=1}^m (r_{a,i} - \bar{r}_a) \times (r_{u,i} - \bar{r}_u)}{\sqrt{\sum_{i=1}^m (r_{u,i} - \bar{r}_u)^2} \times \sqrt{\sum_{i=1}^m (r_{a,i} - \bar{r}_a)^2}}$$

## STEPS:

1. Use a Pearson Correlation approach to find similar users
2. Include all users with positive correlations as neighbours (similarity  $> 0$ )
3. Use a weighted average approach for recommendation



# 1. FINDING SIMILAR USERS

$$w_{a,u} = \frac{\sum_{i=1}^m (r_{a,i} - \bar{r}_a) \times (r_{u,i} - \bar{r}_u)}{\sqrt{\sum_{i=1}^m (r_{u,i} - \bar{r}_u)^2} \times \sqrt{\sum_{i=1}^m (r_{a,i} - \bar{r}_a)^2}}$$

First calculate averages for each person

Movie#	Ken	Lee	Meg	Nan
1	1	4	2	2
2	5	2	4	4
3			3	
4	2	5		5
5	4	1		1
6	?	2	5	

$$\text{Avg(Ken)} = (1+5+2+4)/4 = 3$$

$$\text{Avg(Lee)} = (4+2+5+1+2)/5 = 2.8$$

$$\text{Avg(Meg)} = (2+4+3+5)/4 = 3.5$$

$$\text{Avg(Nan)} = (2+4+5+1)/4 = 3$$

# 1. FINDING SIMILAR USERS

$$w_{a,u} = \frac{\sum_{i=1}^m (r_{a,i} - \bar{r}_a) \times (r_{u,i} - \bar{r}_u)}{\sqrt{\sum_{i=1}^m (r_{u,i} - \bar{r}_u)^2} \times \sqrt{\sum_{i=1}^m (r_{a,i} - \bar{r}_a)^2}}$$

Now find correlations

Movie#	Ken	Lee	Meg	Nan
1	1	4	2	2
2	5	2	4	4
3			3	
4	2	5		5
5	4	1		1
6	?	2	5	

Avg(Ken) = 3

Avg(Lee) = 2.8

Avg(Meg) = 3.5

Avg(Nan) = 3

w(Ken, Lee) :

Top Line:

$$(1-3) \times (4-2.8) + (5-3) \times (2-2.8) + (2-3) \times (5-2.8) + (4-3) \times (1-2.8) =$$

Ken denominator:

$$\sqrt{(1-3)^2 + (5-3)^2 + (2-3)^2 + (4-3)^2}$$

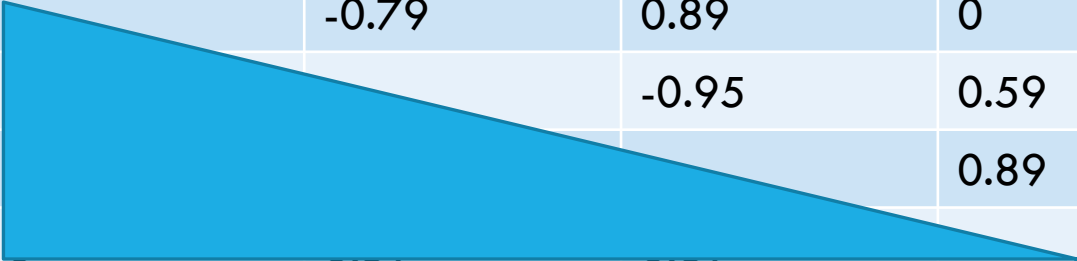
Lee denominator:

$$\sqrt{(4-2.8)^2 + (2-2.8)^2 + (5-2.8)^2 + (1-2.8)^2}$$

corr(Ken, Lee) = -0.79

# PEARSON CORRELATION SIMILARITIES

	Ken	Lee	Meg	Nan
Ken		-0.79	0.89	0
Lee			-0.95	0.59
Meg				0.89
Nan				



### 3. GENERATE RECOMMENDATION:

$$P_{a,i} = \bar{r}_a + \frac{\sum_{u=1}^n (r_{u,i} - \bar{r}_u) \times w_{a,u}}{\sum_{u=1}^n w_{a,u}}$$

- What we want to know is if this value is  $\leq 3$  or  $> 3$
- It will only be recommended if it is  $> 3$

- $P_{\text{ken, movie6}} =$

$$3 + \frac{(5 - 3.5) \times 0.89}{0.89}$$

- $P_{\text{ken, movie6}} = 4.5$

## CONSIDER USING AN ITEM-ITEM NEIGHBOUR BASED APPROACH

Movie#	Ken	Lee	Meg	Nan
1	1	4	2	2
2	5	2	4	4
3			3	
4	2	5		5
5	4	1		1
6	?	2	5	

e.g. Calculate the similarity of movies 2 and 6 using the Pearson correlation formula

avg of movie 2 = 3.75; avg of movie 6 = 3.5

Movie 2 ratings: 5, 2, 4, 4

Movie 6 ratings: 0, 2, 5, 0

$$\frac{((2 - 3.75) \times (2 - 3.5) + (4 - 3.75) \times (5 - 3.5))}{\left(\sqrt{(2 - 3.75)^2 + (4 - 3.75)^2} \times \sqrt{(2 - 3.5)^2 + (5 - 3.5)^2}\right)} = \frac{4}{5} = 0.8$$

## EXAMPLE 2

We are given the following information about 4 users and we wish to make a recommendation for the movie “Black Panther” for Sam who has an average rating of 4:

Joe has rated “Black Panther” 4 and has an average rating of 3.5

Ana has rated “Black Panther” 5 and has an average rating of 3

Ali has rated “Black Panther” 4 and has an average rating of 4.5

The correlations between the users have been calculated as follows:

$$\text{corr}(\text{Sam}, \text{Joe}) = .79$$

$$\text{corr}(\text{Sam}, \text{Ana}) = .57$$

$$\text{corr}(\text{Sam}, \text{Ali}) = 0$$

$$P_{a,i} = \bar{r}_a + \frac{\sum_{u=1}^n (r_{u,i} - \bar{r}_u) \times w_{a,u}}{\sum_{u=1}^n w_{a,u}}$$

Two neighbours: Joe and Ana

$\text{corr}(\text{Sam}, \text{Joe}) = 0.79$

$\text{corr}(\text{Sam}, \text{Ana}) = 0.57$

Joe has rated “Black Panther” 4 and has an average rating of 3.5

Ana has rated “Black Panther” 5 and has an average rating of 3

Sam’s average is 4: Joe’s correlation with Sam      Ana’s correlation with Sam

$$\begin{array}{c} \text{Joe's average and rating} \quad \text{Ana's average and rating} \\ \underbrace{4}_{\text{Sam's avg}} + \frac{(\underbrace{4 - 3.35}_{\text{Joe's dev}}) \times \underbrace{0.79}_{\text{Joe's corr}} + (\underbrace{5 - 3}_{\text{Ana's dev}}) \times \underbrace{0.57}_{\text{Ana's corr}}}{\underbrace{0.79 + 0.57}_{\text{Sum of corrs}}} = 5.21 \end{array}$$

$\Rightarrow$  ‘Black Panther’ recommended to Sam

# HYBRID RECOMMENDER APPROACHES

- Combines more than one approach to make recommendations.
- Approaches combined depend on the data that is available.
- Usually combine popularity, collaborative and content approaches – where each approach can outweigh the disadvantages of the other.
- Recent approaches also integrate data from LOD such as DBpedia and MusicBrainz.org etc.



# WHO IS USING THEM AND WHY?



NETFLIX incorporated one of the earliest successful recommendation systems.

For apps like TikTok, Instagram, YouTube, etc., the better the recommendation algorithm is, the longer people will stay on the app, and the more likely they are to return sooner to see new content.

Typically, to begin these apps might ask for some explicit indicator or preference and/or use popular items.

The more a person uses an app the more information the recommendation system has about their preferences.

# PROBLEMS AND CONCERNS

**The Observer**  
Social media

Molly Russell was trapped by the cruel algorithms of Pinterest and Instagram  
*John Naughton*



Sat 1 Oct 2022 19:00 BST

f t e

The coroner's verdict was a world first in citing social media as a causal factor in a death



📷 Molly Russell was found by a coroner to have 'died from an act of self-harm whilst suffering from depression and the negative effects of online content'. Photograph: Family handout/PA

“It all comes back, in the end, to the business model of companies like Instagram. They thrive on, and monetise, **user engagement** – how much attention is garnered by each piece of content, how widely it is shared, how long it is viewed for and so on. The recommendation engine is programmed to monitor what each user likes and to suggest other content that they might like.”

# SUMMARY

Filtering and Recommendation are a large area of research and a huge application area on the web in particular with most systems offering some form of recommendation.

Many approaches used using different data - content, rating - statistical and machine learning – we concentrate on one approach – a nearest neighbour approach.

In general, amounts of data collected and used are huge.

# WHAT'S IMPORTANT TO KNOW

- Difference between implicit and explicit ratings
- Difference between non-personalised and personalised recommendations
- Difference between content-based and collaborative-based recommendations - what data they use and how they differ in what they can recommend
- How a content-based approach works (using vectors and Cosine similarity)
- How a collaborative-based approach works (using Pearson correlation and weighted average)