Recommendation System

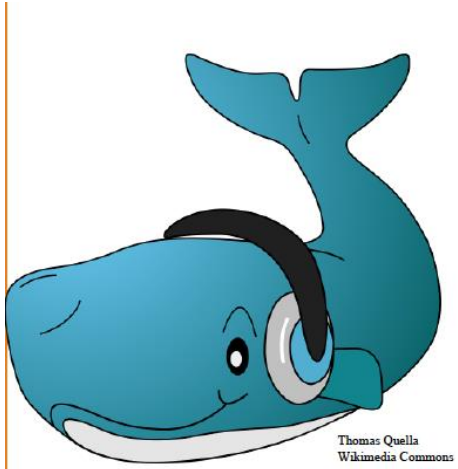
[Slides from Fardina Fathmiul Alam]

OUTLINE

- What are Recommender Systems ?
- Content Based Filtering
- Collaborative Filtering

RECOMMENDER SYSTEMS: THE TASK

"Recommendation Engines" and "Recommendation Systems" are terms that are often used interchangeably,



Plays an Justin Bieber song

What should we recommend next?

RECOMMENDATION ENGINES

- Another specific, common form of supervised learning.
- Recommend the most relevant items to a particular user
  - They are used to, given a user's existing interests, predict other things that user would like.

WHAT PROBLEM DO RECOMMENDATION ENGINES SOLVE?

- Often answers the question of "will person X be interested in product Y"

RECOMMENDER SYSTEMS

RS seen as a function. Given:

- User model (e.g. ratings, preferences, demographics, situational context)
- Items (with or without description of item characteristics)

Find:

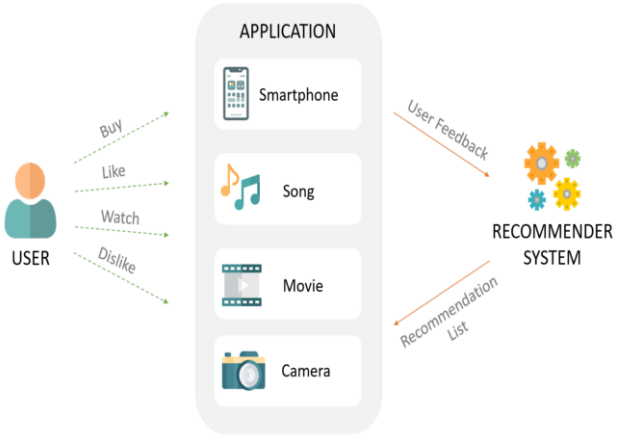
- Relevance score. Used for ranking.

Finally:

- Recommend items that are assumed to be relevant

But:

- Remember that relevance might be context-dependent.
- Characteristics/Features of the list/item itself might be important (diversity).

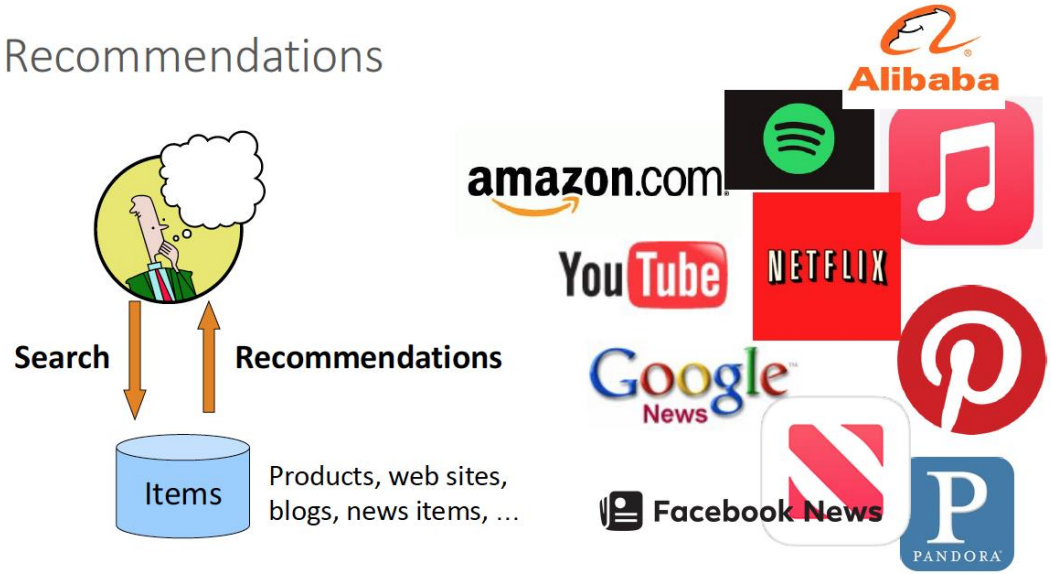


WHY ARE WE TALKING ABOUT THEM?

- Most major tech companies use them
  - Recommendation algorithms back almost all of the most profitable tech enterprises.

Examples

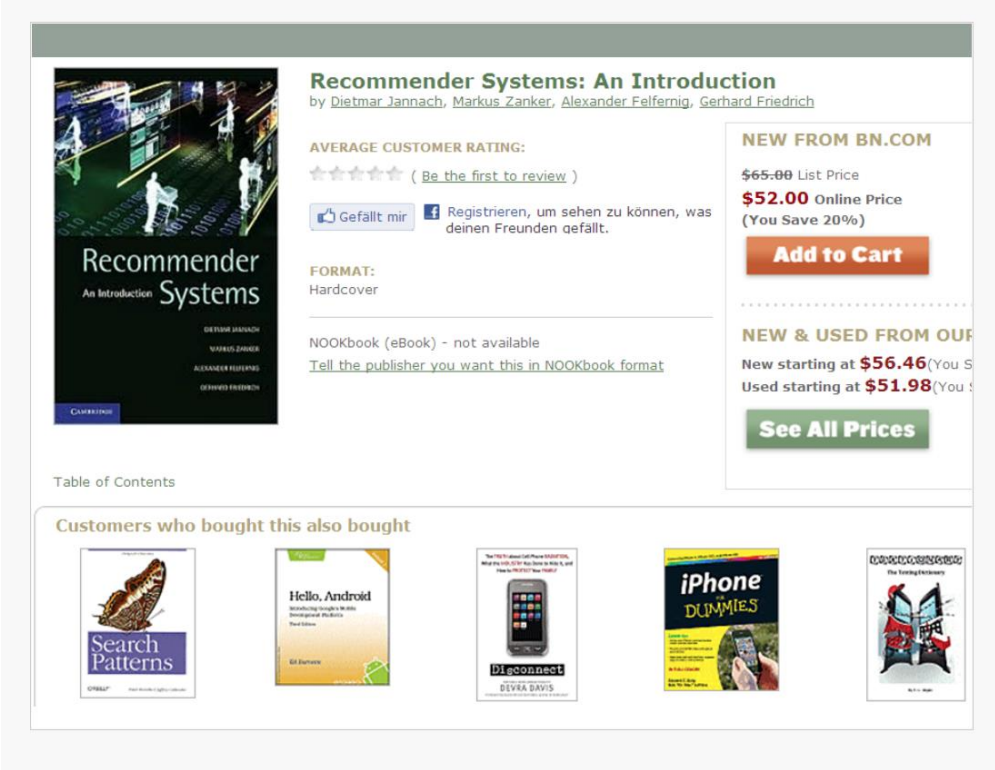
- Netflix
- AMAZON
- Youtube
- All ads
- Steam
- Tinder
- Social Media



Slides adapted from Jure Leskovec, CS246 and J. Leskovec, A. Rajaraman, J. Ullman: *Mining of Massive Datasets*

Examples

- Amazon recommendations
- Google ads
- Netflix
- Pandora / Spotify
- ...pretty much everything



https://www.kaggle.com/datasets/netflix-inc/netflix-prize-data

MEASURING SUCCESS

How do I know if someone liked something (products, movies etc.)?

MEASURING SUCCESS

How do I know if someone liked something?

- If they clicked (Click through rate), (Conversion)
- Slowed down on scroll
- How long they watched it
- Thumbs up / Thumbs down
- Put it on a watch list
- Shared the link
- Liked it
- Comments
- Spent money
- Saved it under collection

AN EXAMPLE: MOVIE RECOMMENDATION

How do we build one of these Recommendation System?

Let's use what we know to write a movie recommendation engine.

DATA

- What will our data be?
- What will our labels be?

DATA

- What will our data be?
  - Movies, vectorized in some way
- What will our labels be?
  - Ratings from the users

HOW MIGHT WE FEATURIZE A MOVIE?

What are some features for a movie that might be salient for people to like or dislike it?

HOW MIGHT WE FEATURIZE A MOVIE?

- Genre
- Director
- Runtime
- Budget
- Actors, etc.

For a movie, we can compute all of these metrics and turn that or transform that movie into a vector.

HOW CAN WE CREATE LABELS?

HOW CAN WE CREATE LABELS?

- You ask them
  - Star rating
  - %
  - Thumbs up / Thumbs down
- How much of the movie they watched
  - Or how many times they watched
  - Attention
- Amazon
  - Bought it (and didn't return)
  - Looking at it
  - Reviews

HOW TO BUILD ONE

Class label: Liked / didn't like

Vectorize the movies: You can pull features like genre, length, cast members, director, rating, title keywords etc.

OUR ALGORITHM

< Runtime, genre, budget, actors, director, thumbs_up> —> train supervised learning model.

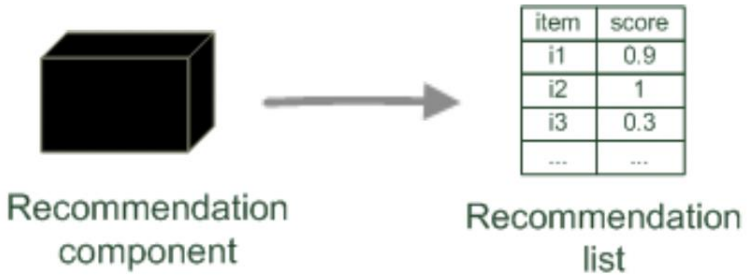
When we get a new < Runtime, genre, budget, actors, director>, we predict whether or not it will get a thumbs up

This is characteristic of a content-based recommendation system (NEXT TOPIC) using a supervised learning approach

PARADIGMS OF RECOMMENDER SYSTEMS

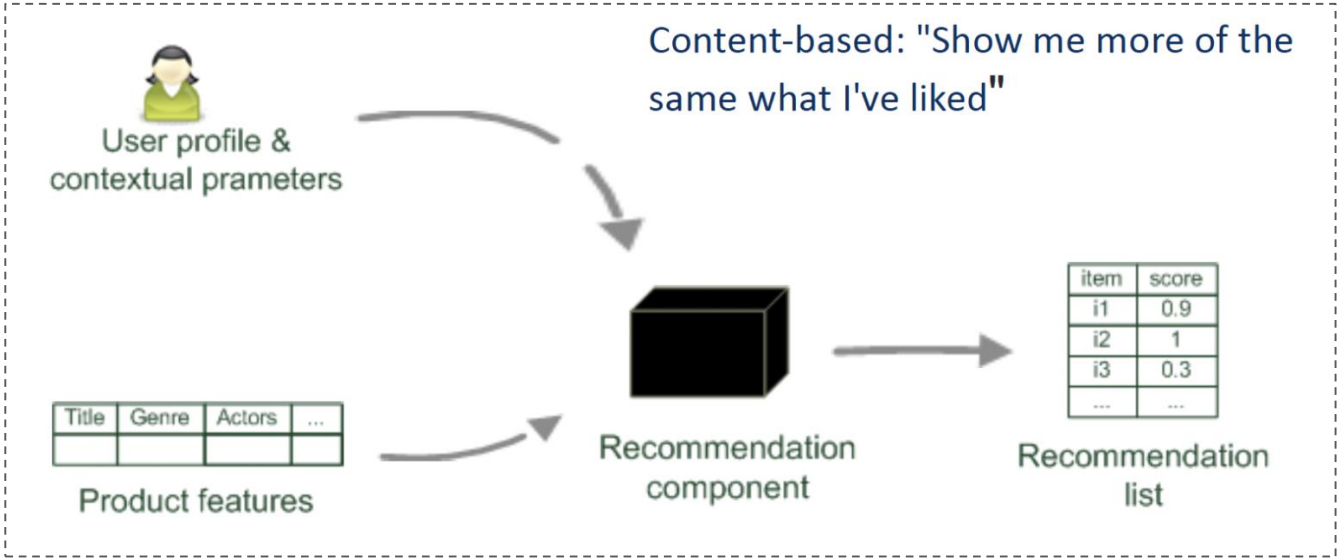
- **Content based approaches**
- **Collaborative approaches**
- **Hybrid systems**, and more.

Recommender systems reduce information overload by estimating relevance



PARADIGMS OF RECOMMENDER SYSTEMS

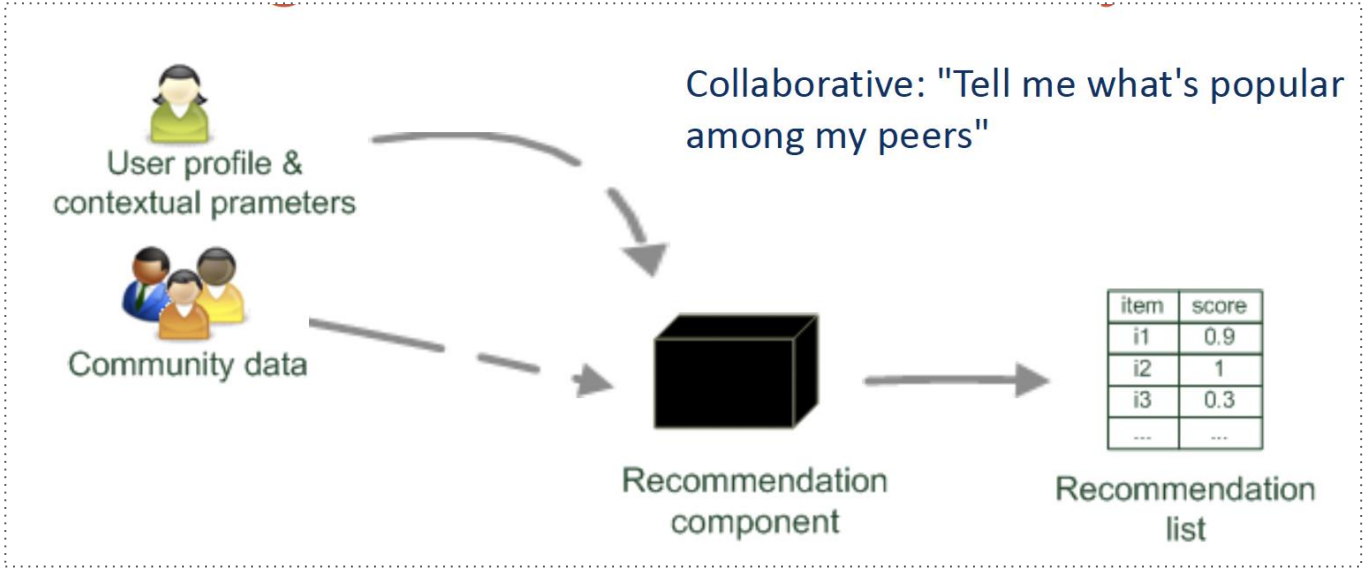
Content based approaches (Use prior information about users and/or items)



- Techniques include regression or classification models.
- Recommendations are based on the attributes or characteristics of items and the preferences of users.

PARADIGMS OF RECOMMENDER SYSTEMS

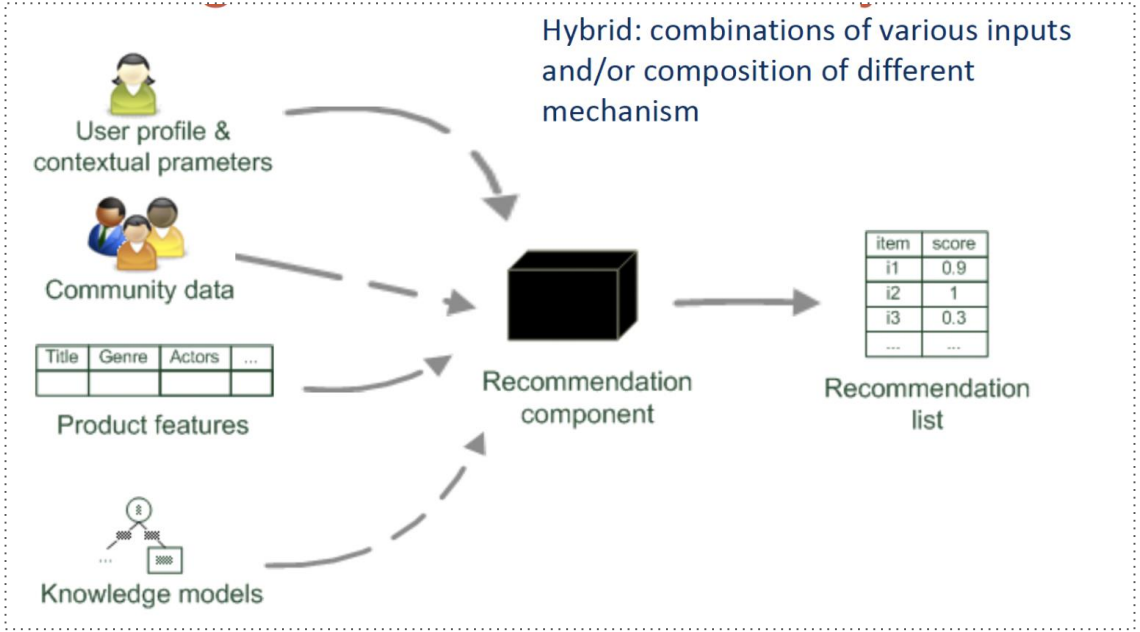
Collaborative approaches (Solely rely on the user-item interaction matrix.)



- Techniques include user-user, item-item and matrix factorisation.
- User-User: Recommendations based on similar users' preferences.
- Item-Item: Recommendations based on similar items.
- Matrix Factorization: Decompose the user-item interaction matrix into lower-dimensional matrices to capture latent factors.

PARADIGMS OF RECOMMENDER SYSTEMS

Hybrid approaches (Combine multiple recommendation approaches.)

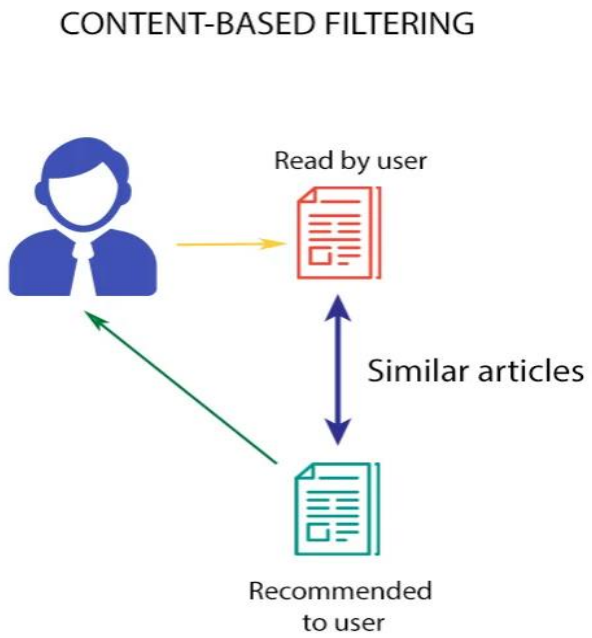


- Leverage the strengths of both content-based and collaborative methods.matrices to capture latent factors.

CONTENT - BASED RECOMMENDATION SYSTEM

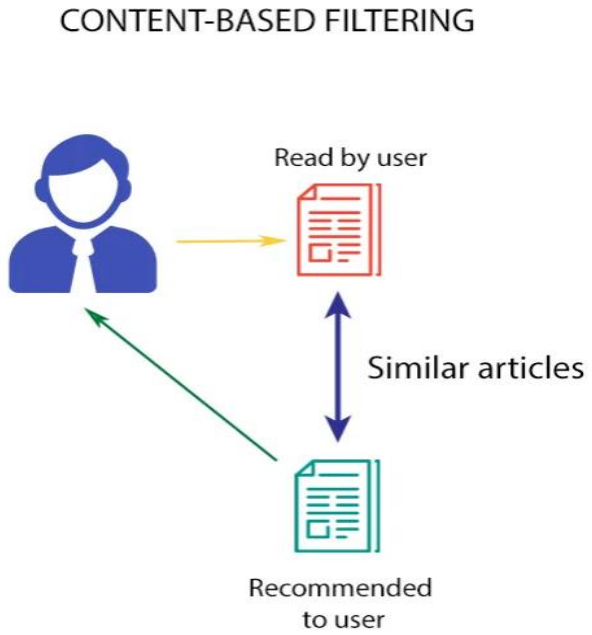
CONTENT-BASED RECOMMENDATION

Content-based recommendation systems are a popular and widely used approach to provide **personalized recommendations to users**.



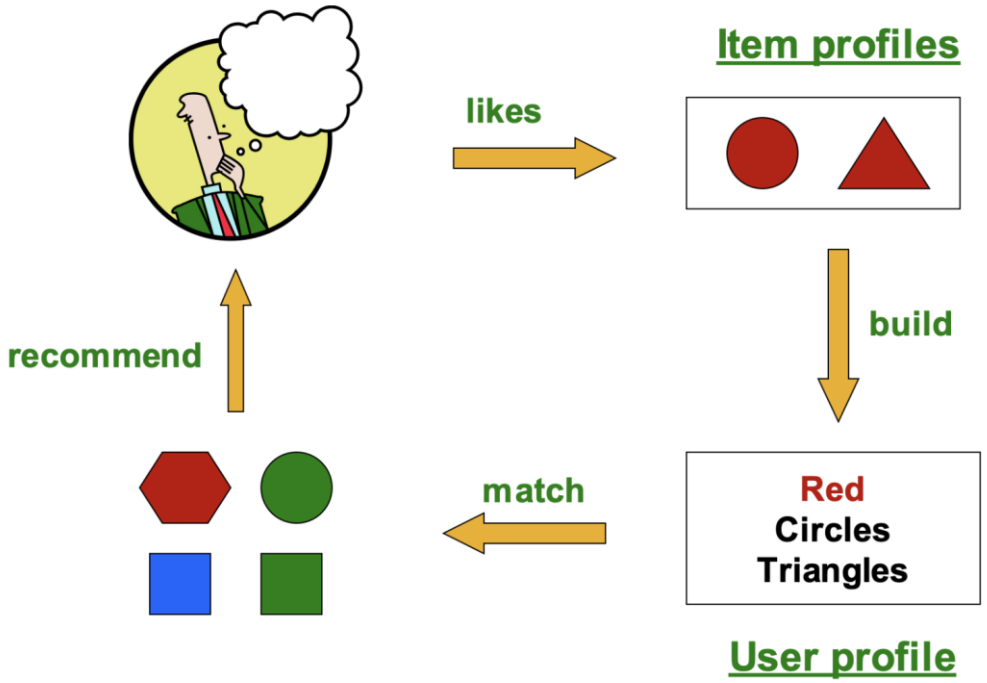
CONTENT-BASED RECOMMENDATION

MAIN IDEA: A user's preferences can be predicted based on their previous interactions with items, such as their viewing and purchasing history.



CONTENT-BASED RECOMMENDATION

GOAL: Recommend items to a user that are similar to items that they have previously interacted with.



CONTENT-BASED RECOMMENDATION

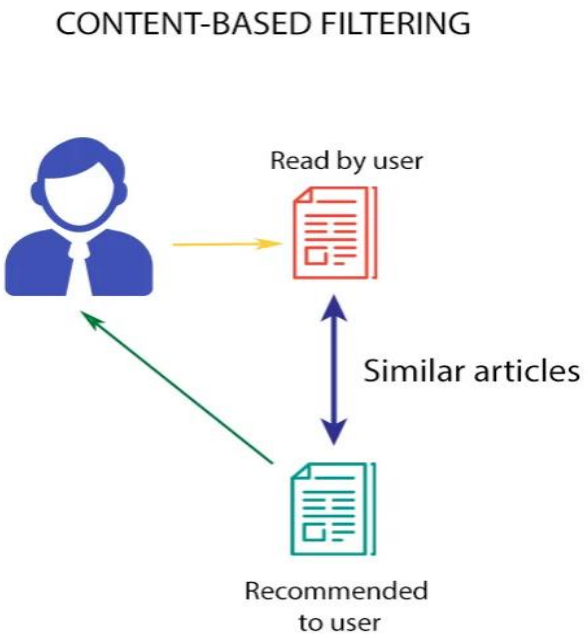
Content based Recommendation **REQUIRE** information about the items.

What do we need:

- ❏ Some information about the available items such as the genre ("**content**")
- ❏ Some sort of user profile describing what the **user likes (the preferences)**

The Task:

- ❏ Learn user preferences
- ❏ Locate/recommend items that are **"similar" to the user preferences.**



PROCESS OF CONTENT-BASED RECOMMENDATION

As content-based recommendation works on similarities between products, that's why we need to:

1. **Featurize each item**: we must create a vector representing all the product features.

- Each item is represented by a set of content features. These features could include attributes like genre, keywords, actors, directors, etc.

2. **Calculate Similarity between vectors**: then, we calculate the similarity between those item vectors using methods like:

  - Euclidean Distance
  - Manhattan Distance
  - Jaccard Distance
  - Cosine Distance (or cosine similarity – we will be using this metric)

This measures how closely items match **in terms of their content features**.

PROCESS OF CONTENT-BASED RECOMMENDATION

3. **Build a Model of User Preferences:** Construct a model that learns what kinds of items the user likes based on the content features of items they have interacted with positively in the past. This model captures the user's taste profile

4. **Identify Potential Recommendations:** Based on the user's taste profile and the similarity between items' content features, select items that the user hasn't interacted with or bought yet but the model predicts they may like align with their preferences.

PROCESS OF CONTENT-BASED RECOMMENDATION

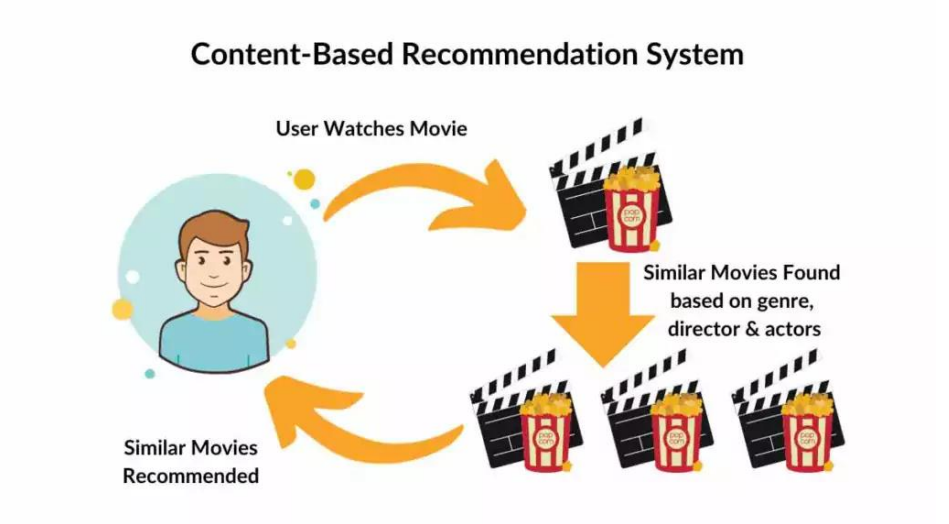
5. **Present Curated Recommendations:** Show the user a curated list of items they haven't bought but are predicted to match their preferences.

This process ensures that recommendations are tailored to the user's tastes by analyzing item features, user behavior, and similarity metrics.

EXAMPLE: CONTENT-BASED RECOMMENDATION

if a set of users likes action movies by Jackie Chan, this algorithm may recommend the movies having the below characteristics.

- Having a genre of action
- Having Jackie Chan in the cast.



Pros: Content Based Systems

- No need for other users
- Personalized Recommendations: recommend items based on unique user tastes.
- Include New and Unpopular Items: Suggestions are not limited by popularity; they're tailored to item features.
- Provide Explanations: Offer clear reasons for recommendations, driven by user preferences and item features.

Cons: Content Based Systems

- Challenges with Feature Selection: Identifying suitable features can be challenging.
- Difficulty in Building User Profiles: Building user profiles is tough, especially if the user has limited interaction history
- Content Profile Constraint: Recommendations stay within the user's content profile, potentially limiting exposure to diverse content.

Cons: Content Based Systems cont.

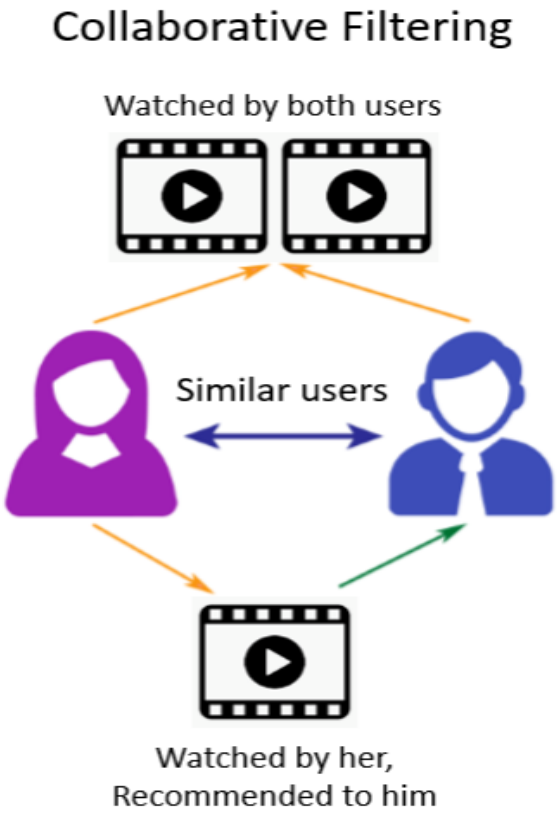
- Opportunity Missed: relies solely on existing user preferences, it may miss opportunities to recommend items the user might enjoy but hasn't encountered.
- Need for Retraining: Requires periodic retraining to adapt to changes in user preferences or to include new items.

COLLABORATIVE FILTERING - BASED RECOMMENDATION SYSTEM

COLLABORATIVE FILTERING

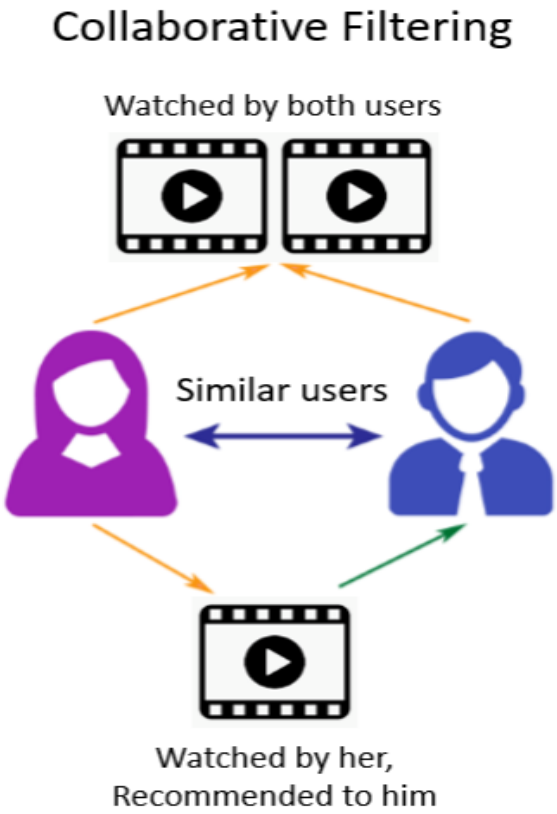
Insight: People who are similar may enjoy similar things.

COLLABORATIVE FILTERING



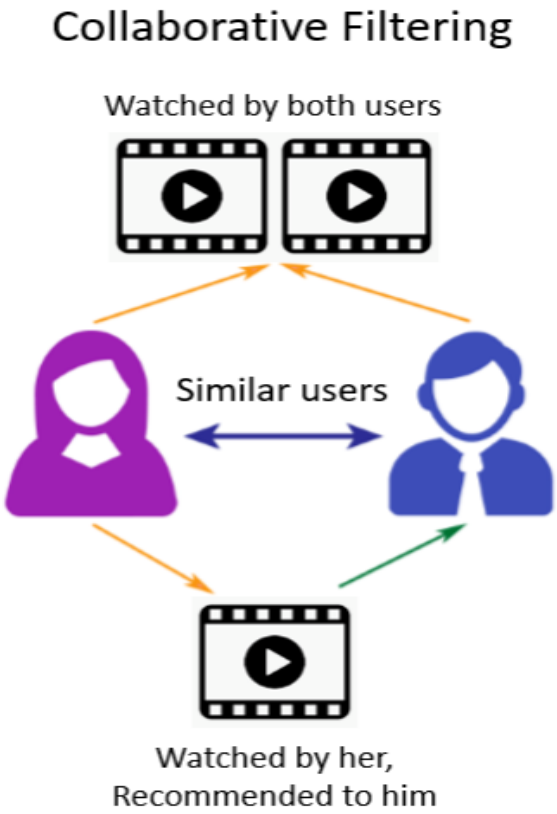
Remember, unlike content-based recommendation (use additional information or features about users and/or items) Collaborative filtering does NOT require any information/features about the items.

COLLABORATIVE FILTERING



In collaborative filtering, a recommendation system recommends a user the products on the basis of the **preferences of the other users with similar tastes**.

EXAMPLE: COLLABORATIVE FILTERING



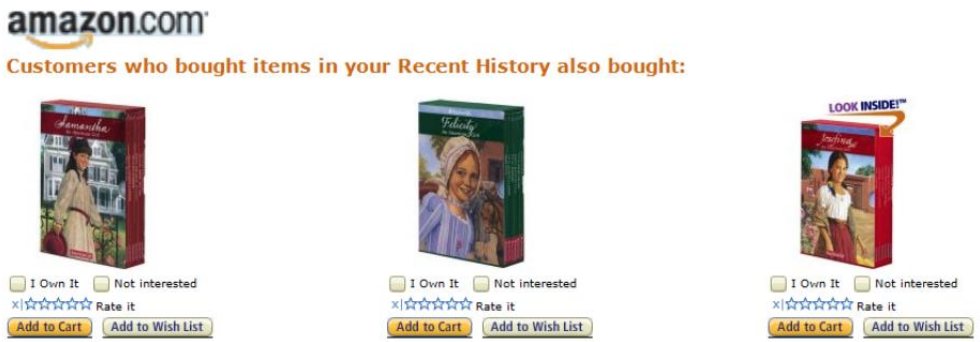
if you are listening to music on Spotify, then it is likely that the **music liked by the other users with the similar taste** will be suggested to you.

COLLABORATIVE FILTERING

What to recommend?

“Find similar users and recommend items that they like”!

- Use other people’s ratings to help rank and recommend things to other people



COLLABORATIVE FILTERING: TYPES

Two types:

1. User-based nearest-neighbor collaborative filtering
2. Item-based collaborative filtering

COLLABORATIVE FILTERING: TYPES

User-based nearest-neighbor collaborative filtering:

Recommendations are made based on the preferences and behaviors of **similar users** (users ‘N’ with similar preferences to a target user ‘I’).

COLLABORATIVE FILTERING: TYPES

User-based nearest-neighbor collaborative filtering:

Recommendations are made based on the preferences and behaviors of **similar users** (users ‘N’ with similar preferences to a target user ‘I’).

Idea: “users who have similar tastes in the past will *continue to have* similar tastes in the future”.

COLLABORATIVE FILTERING: TYPES

User-based nearest-neighbor collaborative filtering:

Example: If User A and User B have similar preferences,

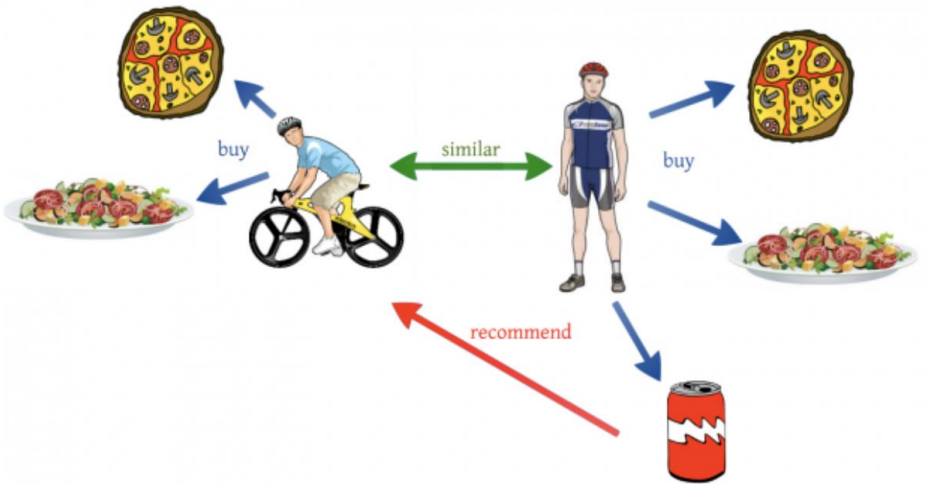
- Items liked or recommended by User B BUT NOT YET SEEN by User A might be suggested to User A.

“John (User A) receives recommendations for movies that Harry (User B) has enjoyed, as they share similar movie preferences.”

COLLABORATIVE FILTERING: TYPES

User-based nearest-neighbor collaborative filtering:

Example 2: John (User A) receives recommendations for energy drinks, inspired by Harry (User B) choice, as they both share a fondness for fitness-related items like pie and protein salad



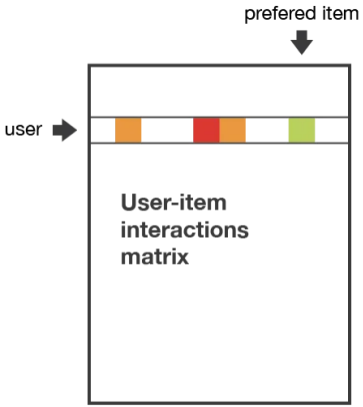
USER-BASED COLLABORATIVE FILTERING

Insight: People who are similar may enjoy similar things.

Question: How do we find people who are similar?

OVERVIEW OF THE STEPS: USER-BASED COLLABORATIVE FILTERING

- **User Data Collection:** Collect data on user preferences and interactions (e.g. ratings, likes, views, and other relevant user activities).
- **Create a User-Item Matrix:** representing user-item relationships (movies, products, etc.)



OVERVIEW OF THE STEPS: USER-BASED COLLABORATIVE FILTERING

- **Similarity Calculation:** Measure the similarity between users based on their preferences.
- **Neighborhood Selection:** Identify a subset of users (“neighborhood”) who are most similar to the target user.

OVERVIEW OF THE STEPS: USER-BASED COLLABORATIVE FILTERING

- **Predict the target user's rating** for unseen items based on similar users' ratings.(use weighted averages or other prediction models).
- **Recommend the top-N items** with the highest predicted ratings to the target user.

Apply post-processing steps, refinements, and continuously update recommendations based on user feedback to ensure relevance over time.

LET'S SEE AN EXAMPLE WITH STEPS

Problem Statement: I need to know what movie to recommend to my users?

Key idea: Past similar preferences can inform future preferences

- User rated movies between 1 (aweful) to 5 (really loved the movie).
- Does not mean they *will rate every single movie*!

Q: What should I recommend today to the user Alice?

	HP1	HP2	HP3	TW	SW1	SW2	SW3
Alice	4			5	1		
Ben	5	5	4				
Clif				2	4	5	

Figure: User - Item Interaction Matrix where Movie Ratings is between [1-5]

VECTORIZE USERS: USER-ITEM MATRIX

- The matrix entries contain the user's preferences or interactions with items.
- Find similar users and recommend items that they like:
  - Represent users by their rows in the utility matrix
  - **IDEA: Two users are similar if their vectors are similar!**

	HP1	HP2	HP3	TW	SW1	SW2	SW3
Alice	4			5	1		
Ben	5	5	4				
Clif				2	4	5	

Figure: User - Item Interaction Matrix where Movie Ratings is between [1-5]

VECTORIZE USERS: USER-ITEM MATRIX

Rating vector for Alice

	HP1	HP2	HP3	TW	SW1	SW2	SW3
Alice	4			5	1		
Ben	5	5	4				
Clif				2	4	5	

Figure: User - Item Interaction Matrix where Movie Ratings is between [1-5]

Next Question is: Which user (Ben, or Clif) is more similar to Alice?

VECTORIZE USERS: USER-ITEM MATRIX

	HP 1	HP2	HP3	TW	SW1	SW2	SW3
Alice(A)	4			5	1		
Ben (B)	5	5	4				
Clif (C)				2	4	5	

- Consider user x and y with rating vector r_x and r_y
- We need a similarity matrix sim(x,y)

Ques: What is the intuition from sim(A,B) vs sim(A,C) from above?

VECTORIZE USERS: USER-ITEM MATRIX

	HP1	HP2	HP3	TW	SW1	SW2	SW3
Alice (A)	4			5	1		
Ben (B)	5	5	4				
Clif (C)				2	4	5	

- Consider user x and y with rating vector r_x and r_y
- We need a similarity matrix sim(x,y)

Intuition from above: sim(A,B) > sim(A,C)

HOW DO WE FIND SIMILARITY?

HOW DO WE FIND SIMILARITY?

- Jaccard Coefficient/ Similarity
- Cosine Similarity

HOW DO WE FIND SIMILARITY?: SOME WAYS

Finding "Similar" Users

r_x = [1, _, _, 1, 3]
r_y = [1, _, 2, 2, _]

- Let r_x be the vector of user x's ratings
- **Jaccard similarity measure**
  - **Problem:** Ignores the value of the rating

r_x, r_y as sets:
r_x = {1, 4, 5}
r_y = {1, 3, 4}

- **Cosine similarity measure**
  - $\text{sim}(x, y) = \cos(r_x, r_y) = \frac{r_x \cdot r_y}{||r_x|| \cdot ||r_y||}$
  - **Problem:** Treats some missing ratings as "negative"

r_x, r_y as points:
r_x = {1, 0, 0, 1, 3}
r_y = {1, 0, 2, 2, 0}

- **Pearson correlation coefficient**
  - S_{xy} = items rated by both users x and y

$$sim(x, y) = \frac{\sum_{s \in S_{xy}} (r_{xs} - \overline{r_x})(r_{ys} - \overline{r_y})}{\sqrt{\sum_{s \in S_{xy}} (r_{xs} - \overline{r_x})^2} \sqrt{\sum_{s \in S_{xy}} (r_{ys} - \overline{r_y})^2}}$$

$\overline{r_x}$, $\overline{r_y}$ … avg. rating of **x**, **y**

OPTION 1: JACCARD SIMILARITY

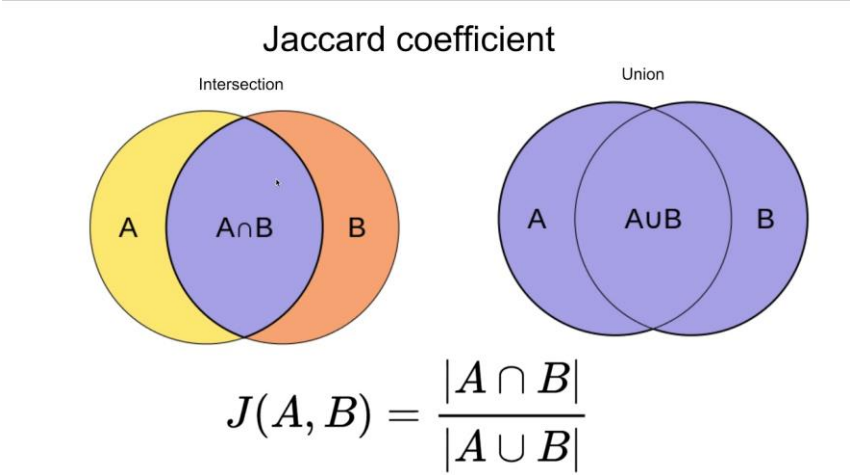
$sim(A,B) = |r_a \cap r_b| / |r_a \cup r_b|$

$= 1 / 5$

$sim(A,C) = 2/4$

	HP 1	HP2	HP3	TW	SW1	SW2	SW3
Alice(A)	4			5	1		
Ben (B)	5	5	4				
Clif (C)				2	4	5	

- **Result:** sim(A,B) < sim(A,C)
- **Problem:** Ignores rating values



OPTION 2: COSINE SIMILARITY

Cosine similarity measure

- $\text{sim}(\boldsymbol{x}, \boldsymbol{y}) = \cos(\boldsymbol{r}_x, \boldsymbol{r}_y) = \frac{r_x \cdot r_y}{||r_x||\,||r_y||}$

	HP 1	HP2	HP3	TW	SW1	SW2	SW3
Alice(A)	4			5	1		
Ben (B)	5	5	4				
Clif (C)				2	4	5	

Intuitively we want: sim(*A*, *B*) > sim(*A*, *C*)

$$\text{sim(A,B)} = \frac{4 \times 5}{\sqrt{4^2 + 5^2 + 1^2}\sqrt{5^2 + 5^2 + 4^2}} = 0.380$$

$$\text{sim(A,C)} = \frac{5 \times 2 + 1 \times 4}{\sqrt{4^2 + 5^2 + 1^2}\sqrt{2^2 + 4^2 + 5^2}} = 0.322$$

OPTION 2: COSINE SIMILARITY

Sim (A,B)=0.38, Sim(A,C) =0.32

- Sim(A,B) > Sim(A,C) but not so much
- **Problem:** treats missing ratings as negative
  - if a user hasn't provided a rating for an item, the system assumes the user dislikes that item.

	HP 1	HP2	HP3	TW	SW1	SW2	SW3
Alice(A)	4			5	1		
Ben (B)	5	5	4				
Clif (C)				2	4	5	

OPTION 2: COSINE SIMILARITY

Sim (A,B)=0.38, Sim(A,C) =0.32

- Sim(A,B) > Sim(A,C) but not so much
- **Problem:** treats missing ratings as negative
  - if a user hasn't provided a rating for an item, the system assumes the user dislikes that item.

	HP 1	HP2	HP3	TW	SW1	SW2	SW3
Alice(A)	4			5	1		
Ben (B)	5	5	4				
Clif (C)				2	4	5	

Try:

- ❏ Centered Cosine Similarity
- ❏ Pearson Correlation Coefficient works well too

OPTION 3: CENTERED COSINE

Normalize rating by subtracting row means

	HP 1	HP2	HP3	TW	SW1	SW2	SW3
Alice(A)	4			5	1		
Ben (B)	5	5	4				
Clif (C)				2	4	5	

→ **row mean 10/3**

→ **row mean 14/3**

OPTION 3: CENTERED COSINE

Normalize rating by subtracting row means

	HP 1	HP2	HP3	TW	SW1	SW2	SW3
Alice(A)	4			5	1		
Ben (B)	5	5	4				
Clif (C)				2	4	5	

→ **row mean 10/3**

→ **row mean 14/3**

	HP1	HP2	HP3	TW	SW1	SW2	SW3
Alice(A)	2/3			5/3	-7/3		
Ben (B)	1/3	1/3	-2/3				
Clif (C)				-5/3	1/3	4/3	

OPTION 3: CENTERED COSINE

Sim (A,B)=cos(r_a, r_b)= 0.09,
Sim(A,C) =-0.56

	HP1	HP2	HP3	TW	SW1	SW2	SW3
Alice(A)	2/3			5/3	-7/3		
Ben (B)	1/3	1/3	-2/3				
Clif (C)				-5/3	1/3	4/3	

- Sim(A,B) > Sim(A,C)
- Sim(A,C) =-0.56 actually captures the fact that A and C are quite dissimilar users.
- Captures Intuition Better
  - Missing rating treated as “average”
  - Handle “tough raters” and “easy raters”
  - This is similar to pearson correlation!

OPTION 3: CENTERED COSINE

Sim (A,B)=cos(r_a, r_b)= 0.09,
Sim(A,C) =-0.56

	HP1	HP2	HP3	TW	SW1	SW2	SW3
Alice(A)	2/3			5/3	-7/3		
Ben (B)	1/3	1/3	-2/3				
Clif (C)				-5/3	1/3	4/3	

- Sim(A,B) > Sim(A,C)
- Sim(A,C) =-0.56 actually captures the fact that A and C are quite dissimilar users.
- Captures Intuition Better
  - Now a "0" means no information
  - And negative ratings mean viewers with opposite rating will have vectors in opposite directions.

Now A and C are (correctly) way further apart than A and B

RETURN OF KNN

- Given a user, find their k nearest neighbors
- Predict the user's rating based on the **weighted average of those users.**
  - **Rate “i” as the mean of what k-people-like-me rated i**

$$r_{xi} = \frac{\sum_{y \in N} s_{xy} \, r_{yi}}{\sum_{y \in N} s_{xy}}$$

S_{xy} = similar neighbour

PREDICT AND RECOMMEND

Predict: Which movie to Recommend to User A next? HP2 or HP3?

	HP1	HP2	HP3	TW	SW1	SW2	SW3
Alice(A)	4			5	1		
Ben (B)	5	5	4				
Clif (C)		2	1	2	4	5	

Given,

- Sim (A,B)=cos(r_a, r_b)= 0.09,
- Sim(A,C) =-0.56

*** Let consider, Clif has already rated HP2 and HP3

PREDICT AND RECOMMEND

- Predict the user's rating based on the weighted average of those users

Predict: Which movie to Recommend to User A next? HP2 or HP3?

	HP1	HP2	HP3	TW	SW1	SW2	SW3
Alice(A)	4			5	1		
Ben (B)	5	5	4				
Clif (C)		2	1	2	4	5	

Given,
Sim (A,B)=cos(r_a, r_b)= 0.09,
Sim(A,C) =-0.56

Prediction for HP2: $P_{A,HP2}$= [S(A,B)*5 + S(A,C)*2] / [S(A,B) + S(A,C)] = 1.42

Same: $P_{A,HP3}$= [S(A,B)*4 + S(A,C)*1] / [S(A,B) + S(A,C)] = 0.42

So, we will recommend HP2 to Alice next

COLLABORATIVE FILTERING

TYPE 2: Item-based collaborative filtering

COLLABORATIVE FILTERING: ANOTHER VERSION

Alternative view that often works better: **Item-Item** **(NEXT)**

COLLABORATIVE FILTERING: TYPES

Item-based collaborative filtering:

IDEA: A user is likely to have the same opinion for similar items.

Example: if I like Canon cameras, I might also like Canon video cameras

COLLABORATIVE FILTERING: TYPES

Item-based collaborative filtering:

In contrast to user-based collaborative filtering, item-based collaborative filtering focuses on the **similarities between items**. (not on user or item features).

Recommendations are made by identifying items that are similar to the ones a user has liked or interacted with.

similarity is based **solely on user-item interactions and does not consider the content features of items**.

COLLABORATIVE FILTERING: TYPES

Item-based collaborative filtering:

Assumption: If a user has shown interest in a particular item, they are likely to be interested in items that are similar to it.

similarity is based **solely on user-item interactions and does not consider the content features of items**.

COLLABORATIVE FILTERING: TYPES

STEPS OF Item-based collaborative filtering:

- For a **given item i, find** other **similar items.**
- **Estimate rating for item i** based on ratings for those similar items.
- Can use same similarity metrics and prediction functions as in user-user model

Courtesy: Jure Leskovec & Mina Ghashami, Stanford

COLLABORATIVE FILTERING: TYPES

STEPS OF Item-based collaborative filtering:

- For a **given item i, find** other **similar items.**
- **Estimate rating for item i** based on ratings for those similar items.
- Can use same similarity metrics and prediction functions as in user-user model
- **Algorithm:** "Rate an **unseen item i as the mean of my ratings for other items,** weighted by their **similarity to i."**

$$r_{xi} = \frac{\sum_{j \in N(i;x)} s_{ij} \cdot r_{xj}}{\sum_{j \in N(i;x)} s_{ij}}$$

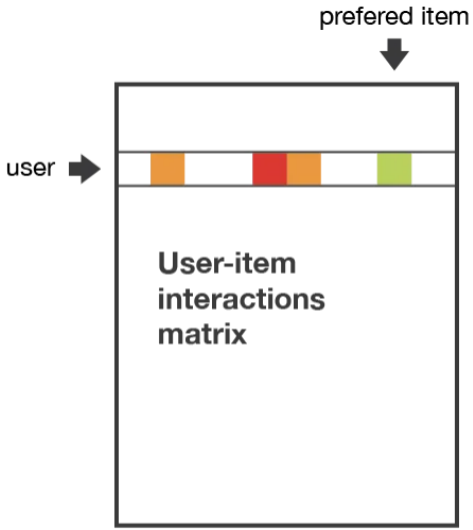
s_{ij}… similarity of items ***i*** and ***j***
r_{xj}…rating of user ***x*** on item ***j***
$N(i;x)$… set of items which were rated by ***x*** and similar to ***i***

Courtesy: Jure Leskovec & Mina Ghashami, Stanford

COLLABORATIVE FILTERING: TYPES

Process of Item-based collaborative filtering:

1. **User-Item Interaction Matrix:** The system first constructs a matrix where rows represent users and columns represent items. Each cell in the matrix contains the user's interaction (e.g., rating, purchase, view) with the corresponding item.



Note: The matrix is sparse, but the empty cells are not (necessarily) zero!

COLLABORATIVE FILTERING: TYPES

Process of Item-based collaborative filtering:

2. Item Similarity Calculation: To find the similarity between two items, the system calculates the cosine similarity between their interaction vectors.

These vectors represent the ratings or interactions of all users with each item.

One common approach: Cosine similarity. In the context of item-based collaborative filtering, it quantifies the similarity in user interaction patterns between two items.

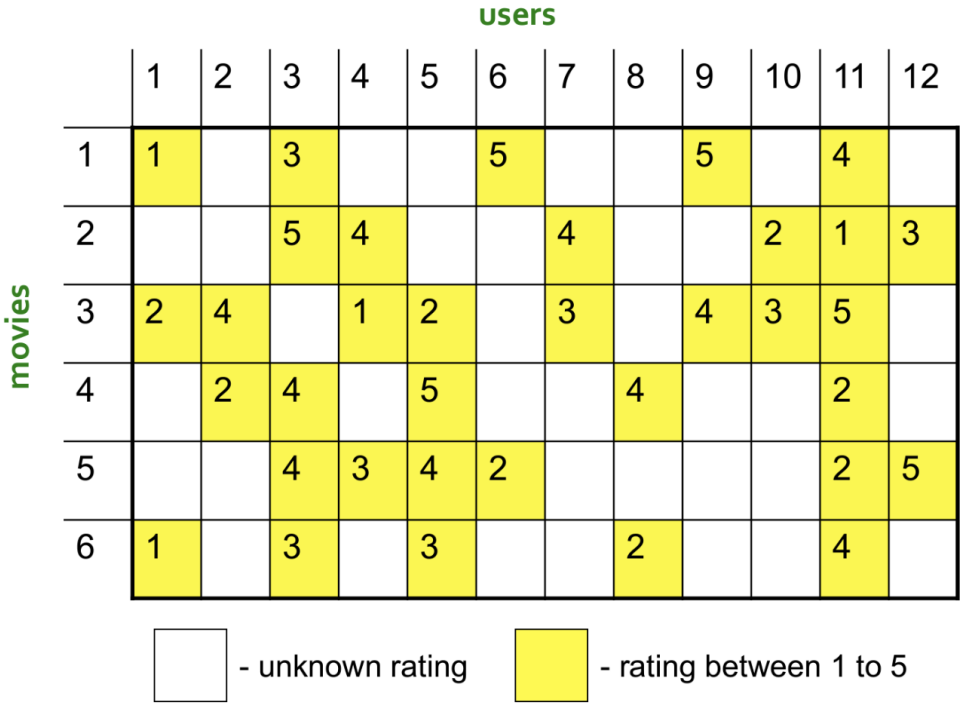
COLLABORATIVE FILTERING: TYPES

Process of Item-based collaborative filtering:

3. Recommendation Generation: Once the similarity between items is calculated, the system identifies items similar to those a user has interacted with and recommends them to the user.

EXAMPLE (ITEM-ITEM CF)

Item-based collaborative filtering: Let's say, |N| =2 (the size of the neighborhood, or the number of similar items)

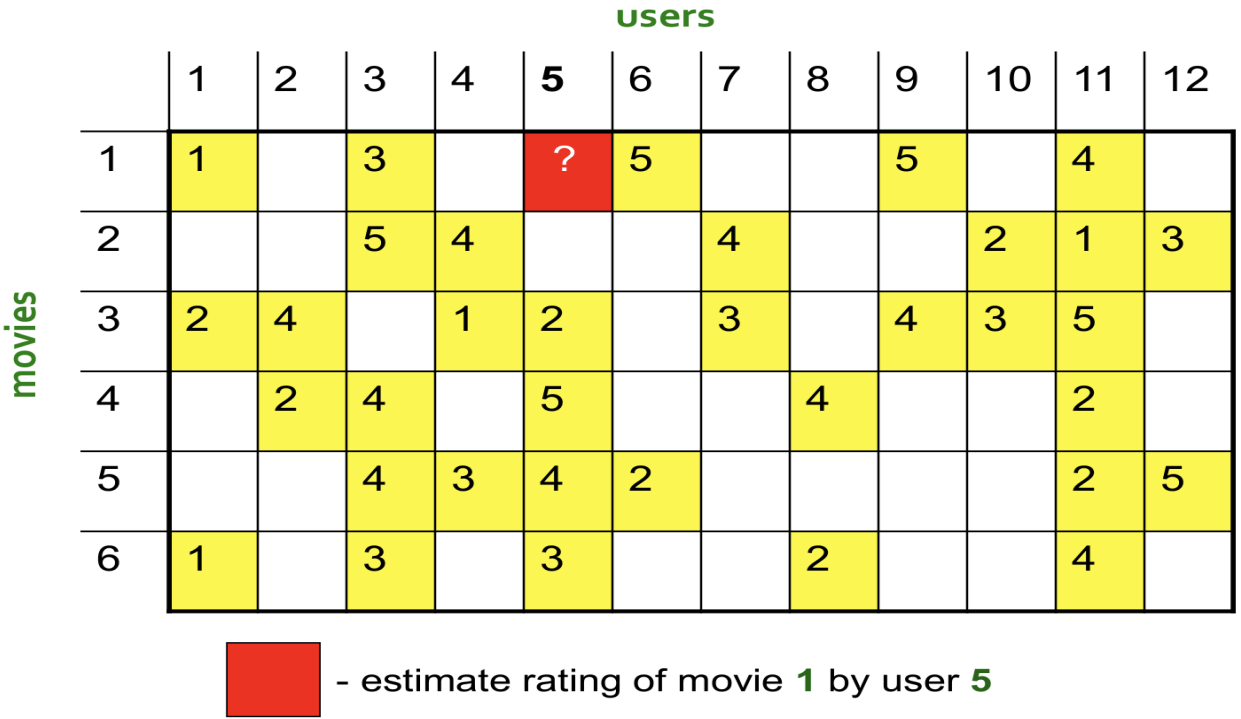


movies \ users	1	2	3	4	5	6	7	8	9	10	11	12
1	1		3			5			5		4	
2			5	4			4			2	1	3
3	2	4		1	2		3		4	3	5	
4		2	4		5			4			2	
5			4	3	4	2					2	5
6	1		3		3			2			4	

Figure: User - Item Interaction Matrix where Movie Ratings is between [1-5].

EXAMPLE (ITEM-ITEM CF; |N|=2)

Let's say, we want to know what USER 5 think about Movie 1?



	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	

EXAMPLE (ITEM-ITEM CF; |N|=2): NEIGHBOUR SELECTION

Let's say, we want to know what USER 5 think about Movie 1?

Let's say, we already know that, Movie 3 and Movie 6 are most similar to Movie 1.

users

movies	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	

Neighbor selection:
Identify movies similar to movie **1**, **rated by user 5**

We now need to calculate similarity between these rows!

EXAMPLE (ITEM-ITEM CF; |N|=2): SIMILARITY CALCULATION

Let's say, we want to know what USER 5 think about Movie 1?

Let's say, we already know that, Movie 3 and Movie 6 are most similar to Movie 1.

users

	1	2	3	4	**5**	6	7	8	9	10	11	12
1	1		3		?	5			5		4	

movies

Subtract mean rating m_i from each movie i

$m_1 = (1+3+5+5+4)/5 = 18/5$

	1	2	3	4	5	6	7	8	9	10	11	12
1	-13/5		-3/5		?	7/5			7/5		2/5	
3	-1	1		-2	-1		0		1	0	2	
6	-8/5		2/5		2/5			-3/5			7/5	

Showing computation only for #3 and #6

Here we use "mean centered item-overlap cosine as similarity:

1) Subtract mean rating mi from each movie i
2) Compute (item-overlapping) cosine similarities between rows

EXAMPLE (ITEM-ITEM CF; |N|=2): SIMILARITY CALCULATION

Let's say, we want to know what USER 5 think about Movie 1?

Compute Cosine Similarity S(1,m):
2. Compute (item-overlapping) cosine similarities between rows

users

movies	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	

For rows 1 and 3, they both have values for users 1, 9 and 11.

$$\text{sim}(1,3) = \frac{(-13/5)(-1)+(7/5)(1)+(2/5)(2)}{\sqrt{(-13/5)^2+(7/5)^2+(2/5)^2}\sqrt{(-1)^2+(1)^2+(2)^2}} \approx 0.658$$

For rows 1 and 6, they both have values for users 1, 3 and 11.

$$\text{sim}(1,6) = \frac{(-13/5)(-8/5)+(-3/5)(2/5)+(2/5)(7/5)}{\sqrt{(-13/5)^2+(-3/5)^2+(2/5)^2}\sqrt{(-8/5)^2+(2/5)^2+(7/5)^2}} \approx 0.768$$

Subtract mean rating m_i from each movie i
$m_1 = (1+3+5+5+4)/5 = 18/5$

	1	2	3	4	5	6	7	8	9	10	11	12
1	-13/5		-3/5		?	7/5			7/5		2/5	
3	-1	1		-2	-1		0		1	0	2	
6	-8/5		2/5		2/5			-3/5			7/5	

EXAMPLE (ITEM-ITEM CF; |N|=2): SIMILARITY CALCULATION

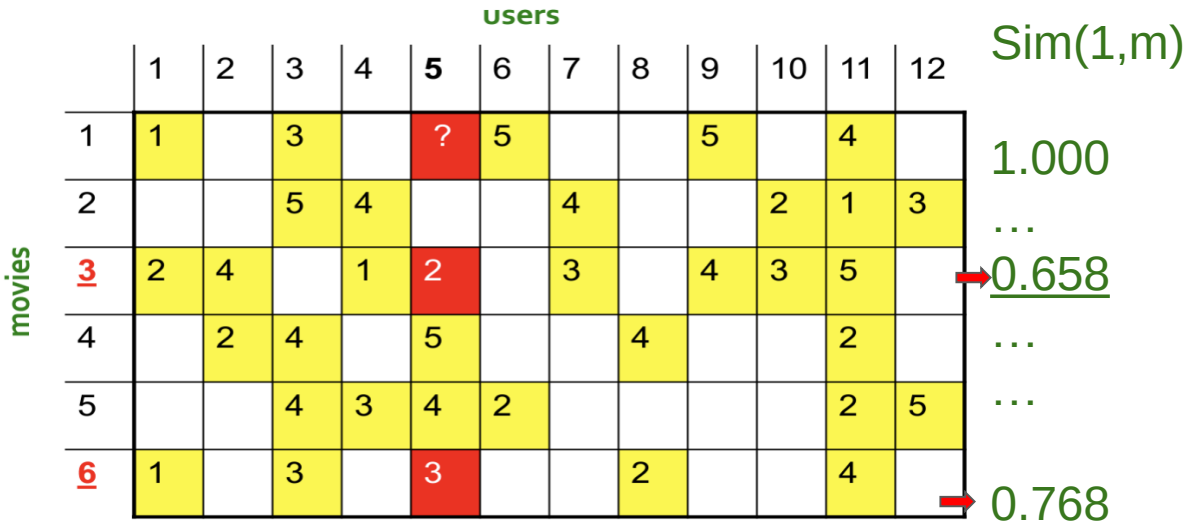
Let's say, we want to know what USER 5 think about Movie 1?

Compute Similarity Weight:

$S_{1,3}$ = 0.658

$S_{1,6}$= 0.768

We computed $S_{1,2}$, $S_{1,4}$,$S_{1,6}$ too! Let's assume those are smaller!



EXAMPLE (ITEM-ITEM CF; |N|=2): APPROXIMATE RATING WITH WEIGHTED MEAN

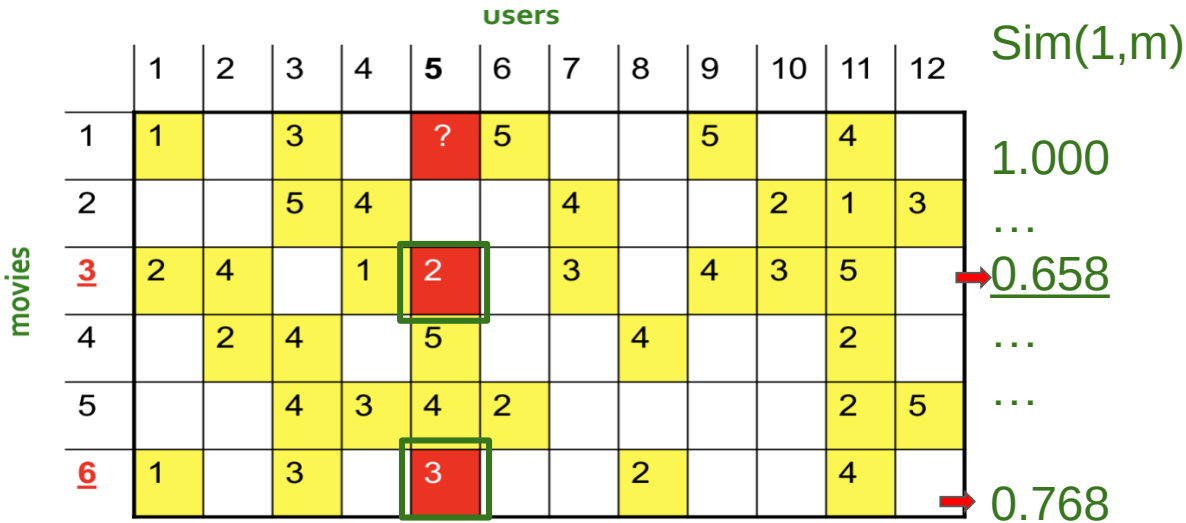
Let's say, we want to know what USER 5 think about Movie 1?

Predict By Taking Weighted Average:

$R_{user1,movies5} = R_{1,5} =$

(0.658*2 +0.768*3) / (0.658+0.768)

=2.54
~ 2.6



$$r_{xi} = \frac{\sum_{j \in N(i;x)} s_{ij} \cdot r_{xj}}{\sum_{j \in N(i;x)} s_{ij}}$$

s_{ij}… similarity of items ***i*** and ***j***
r_{xj}…rating of user ***x*** on item ***j***
N(i;x)… set of items which were rated by ***x*** and similar to ***i***

EXAMPLE (ITEM-ITEM CF; |N|=2): APPROXIMATE RATING WITH WEIGHTED MEAN

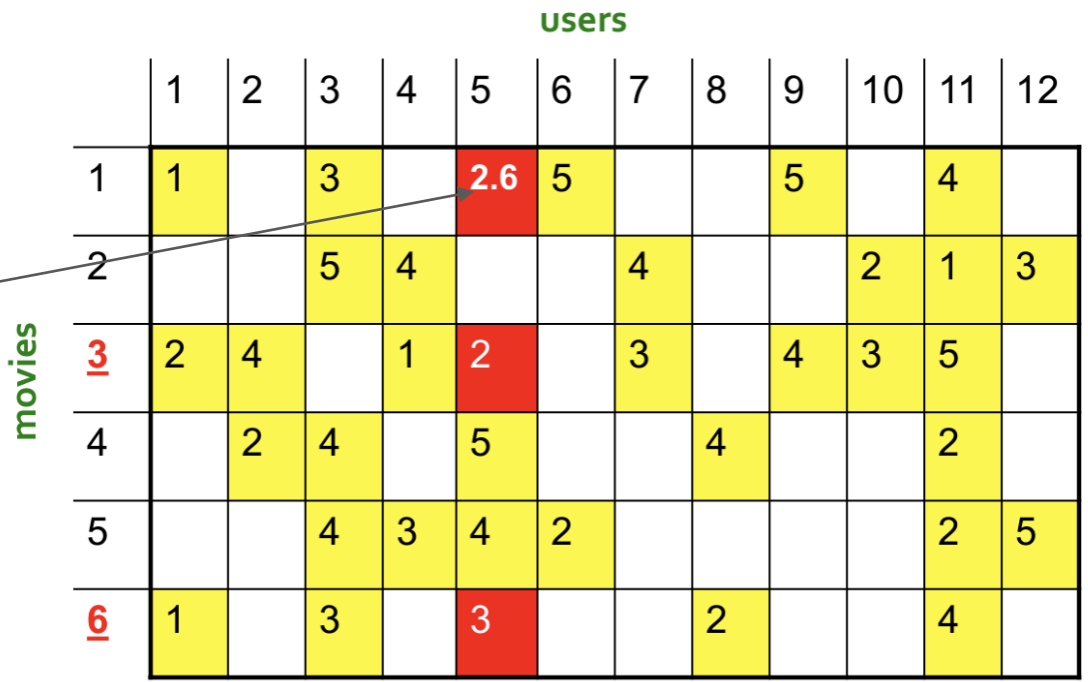
Let's say, we want to know what USER 5 think about Movie 1?

Predict By Taking Weighted Average:

$R_{user1,movies5} = R_{1,5} =$

(0.658*2 +0.768*3) / (0.658+0.768)

=2.54
~ 2.6



Collaborative Filtering

Pros:

- **Diverse Recommendations:** recommend any item in the system, regardless of its attributes or characteristics.
- **Uses more information:** using user-item interaction data

Cons:

- Data Sparsity
- Cold start problem
- Popularity bias

Collaborative Filtering: CONS

Data Sparsity: might not have enough ratings in the user item dataset which makes it impossible to provide proper recommendations.

Cold start problem: the challenge of making accurate recommendations for new users or items with limited or no historical data.

- How to recommend new items? What to recommend to new users?
- Cold start can also happen when we have a new item which has not received a rating.
  - Straightforward approaches
    - Ask/force users to rate a set of items
    - Use another method (e.g., content-based, demographic or simply non-personalized) in the initial phase

Collaborative Filtering: CONS

Popularity bias: Since CF methods rely on historical user-item interactions to make recommendations, they tend to recommend popular items more frequently because they have more data available about them.

This can result in a bias towards popular items, potentially neglecting less popular but potentially relevant recommendations for users.

Next: EVALUATION OF RECOMMENDATION SYSTEM

EVALUATION OF RECOMMENDATION SYSTEM

As always, before we use this algorithm, we'd like to know how well it performs!

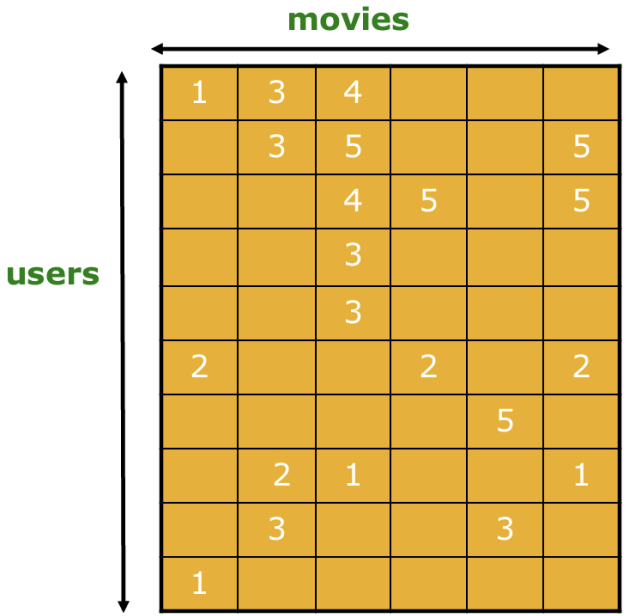
What unique challenges are present here?

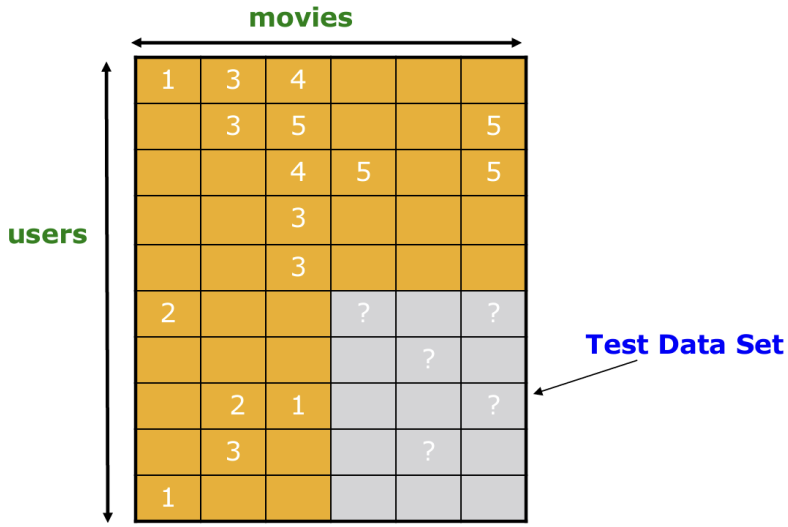
A sample user:

		Ant Man		Endgame	Thor 1	Return of Thor		Iron Dude
Justin	1			0		1		0
			1					

The challenge: We don't know if no rating means the person doesn't know about the movie, or if they do know about it and knew not to watch it because they wouldn't like it.

EVALUATION OF RECOMMENDATION SYSTEM





- **Compare predictions with known ratings**
  - **Root-mean-square error** (RMSE)
    - $\sqrt{\frac{1}{N}\sum_{xi}(r_{xi} - r^*_{xi})^2}$ where r_{xi} is predicted, r^*_{xi} is the true rating of x on i
      - *N is the number of points we are making comparisons on*
  - **Precision at top 10**:
    - % of relevant items in top 10
  - **Rank Correlation**:
    - Spearman's *correlation* between system's and user's complete rankings

EVALUATION OF RECOMMENDATION SYSTEM
cont.

RMSE (Root Mean Square Error):

- Measures the average squared difference between predicted and actual ratings. **Lower RMSE indicates better accuracy**. Useful for assessing the overall performance of recommendation algorithms.

Precision at Top 10:

- Measures the proportion of relevant items among the top 10 recommendations. **Higher precision indicates better quality recommendations.** Useful for evaluating the relevance of top recommendations.

EVALUATION OF RECOMMENDATION SYSTEM
cont.

Precision at Top 10:

For content based filtering, we can look at the ten highest ranked things, and compute the precision of that list.

(how many of those top ten things the user actually liked or bought)

However, the things that the user didn't buy might have just been things they haven't heard of!

EVALUATION OF RECOMMENDATION SYSTEM cont.

Rank Correlation: Pearson Correlation:

- Measures the similarity between predicted and actual rankings of items. **Higher correlation indicates better alignment between predicted and actual rankings.**Useful for assessing the consistency of recommendation rankings.

PER USER LEAVE-ONE-OUT-CROSS-VALIDATION

For each user, leave out one thing they've rated, and then predict the rating.

This tells us about how well we do at the things the user has seen (already rated), but we have no idea how they're doing with the things they haven't (new, not rated yet).

Collaborative vs Content Based Differences

No.	Collaborative Recommendation Engine	Content Based Recommendation Engine
1.	A collaborative recommendation engine emphasizes on the user preference.	A content based recommendation engine emphasizes on the content features.
2.	In collaborative filtering, a recommendation engine requires the user profile to suggest relevant content.	In content based filtering, a recommendation system uses the content profile too which includes the content features.
3.	The collaborative recommendation systems feed on the user ratings, reviews, thumbs ups & downs, and other feedback on various products or services. So, the products with no ratings or feedback can't be recommended to any user. Neither a new user who hasn't given any reviews or ratings can get any recommendation by the collaborative recommendation engine. This is called the cold **start problem.**	The content-based recommendation systems are product features oriented and hence don't have such problems.
4.	A collaborative recommendation engine doesn't always ensure precise recommendations. Because the users with similar tastes may not like the same products always.	A content based recommendation engine can provide more accurate recommendations as it focuses on the features of the content a user likes.

(SOME MORE) RECOMMENDER SYSTEMS (ISH)

ASSOCIATION RULES

In today's data-driven world, businesses strive to *understand the relationships between different products or items purchased together by customers*.

This understanding allows them to optimize marketing strategies, enhance product recommendations, and improve overall customer experience

ASSOCIATION RULES

Collaborative Filtering systems give predictions based on other similar users' rating of the same item

Complementary idea: Find rules that associate/connect the presence of one set of items with that of another set of items

Association rules are about finding connections between different sets of items. For example, if people often buy chips when they buy salsa, that's an association rule. It helps us understand how items are related to each other in a shopping basket.

ASSOCIATION RULES and RECOMMENDATION SYSTEM

- **Identifies** patterns and relationships between items in a dataset.
- **Enhances recommendation systems** by revealing item associations and user preferences.
- **Leads to more diverse and accurate recommendations** tailored to individual users.

We will now learn

- What is Association Rules
- What is Frequent Itemset
- Mining Association Rules: Two Steps
  - Frequent Itemset Generation
    - Apriori principle
  - Rule Generation

ASSOCIATION RULES

- Given a set of transactions, **find rules** that will predict the occurrence of an item based on the occurrences of other items in the transaction.

Market-Basket transactions

TID	Items
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Example of Association Rules

(when customers buy diapers, they are likely to also buy beer.)

$$\{Diaper\} \rightarrow \{Beer\},$$
$$\{Milk, Bread\} \rightarrow \{Eggs, Coke\},$$
$$\{Beer, Bread\} \rightarrow \{Milk\},$$

Implication means co-occurrence, not causality!

Buying one set of items doesn't cause the purchase of another set. Instead, they tend to occur together frequently in transactions.

DEFINITION: FREQUENT ITEMSET

- **Itemset**
  - A collection of one or more items
    - Example: {Milk, Bread, Diaper}
  - k-itemset
    - An itemset that contains k items
- **Support count (σ)** (or Frequency)
  - Frequency of occurrence of an itemset
  - E.g. σ({Milk, Bread, Diaper}) = 2
- **Support**
  - Fraction of transactions that contain an itemset
  - E.g. s({Milk, Bread, Diaper})
    = σ({Milk, Bread, Diaper}) / |T| = 2/5 = 0/4 0r 40%
- **Frequent Itemset**
  - An itemset whose support is greater than or equal to a *minsup* threshold

TID	Items
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

$$s(X) = \frac{\sigma(X)}{|T|}$$

DEFINITION: Association Rule

- **Association Rule**
  - An implication expression of the form $X \rightarrow Y$, where X and Y are itemsets
  - Example: $\{Milk, Bread\} \rightarrow \{Diaper\}$

TID	Items
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

- **Rule Evaluation Metrics**
  - Support (s)
    - Fraction of transactions that contain both X and Y
  - Confidence (c)
    - Measures how often items in Y appear in transactions that contain X

S: This measure gives an idea of how frequent an itemset is in all the transactions.

C: measures how likely it is for an item (consequent) to be bought when another set of items (antecedents) is already in the cart.

Example:

$\{Milk, Bread\} \rightarrow \{Diaper\}$

$$s = \frac{\sigma(\{Milk, Bread, Diaper\})}{|T|} = \frac{2}{5} = 0.4$$

$$c = \frac{\sigma(\{Milk, Bread, Diaper\})}{\sigma(\{Milk, Diaper\})} = \frac{2}{3} = 0.67$$

$$c(X \rightarrow Y) = \frac{\sigma(X \cup Y)}{\sigma(X)} = \frac{s(X \cup Y)}{s(X)}$$

ASSOCIATION RULE MINING TASK

Given a set of transactions T, the goal of association rule mining is to find all rules having

- support ≥ minsup threshold
- confidence ≥ minconf threshold *(we can determine these thresholds by domain knowledge, trial and error, business requirements etc.)*

Brute-force approach:-

- List all possible association rules
- Compute the support and confidence for each rule
- Prune rules that fail the minsup and minconf thresholds

Computationally prohibitive!

MINING ASSOCIATION RULES

TID	Items
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Example of Rules:

{Milk,Diaper} → {Beer} (s=0.4, c=0.67)
{Milk,Beer} → {Diaper} (s=0.4, c=1.0)
{Diaper,Beer} → {Milk} (s=0.4, c=0.67)
{Beer} → {Milk,Diaper} (s=0.4, c=0.67)
{Diaper} → {Milk,Beer} (s=0.4, c=0.5)
{Milk} → {Diaper,Beer} (s=0.4, c=0.5)

Observations:

- All the above rules are binary partitions of the same itemset: {Milk, Diaper, Beer}
- Rules from the same set have identical support (s=0.4) but varying confidence (c).
- Thus, we may can adjust confidence independently of support, allowing flexibility in rule generation.

MINING ASSOCIATION RULES: TWO STEPS

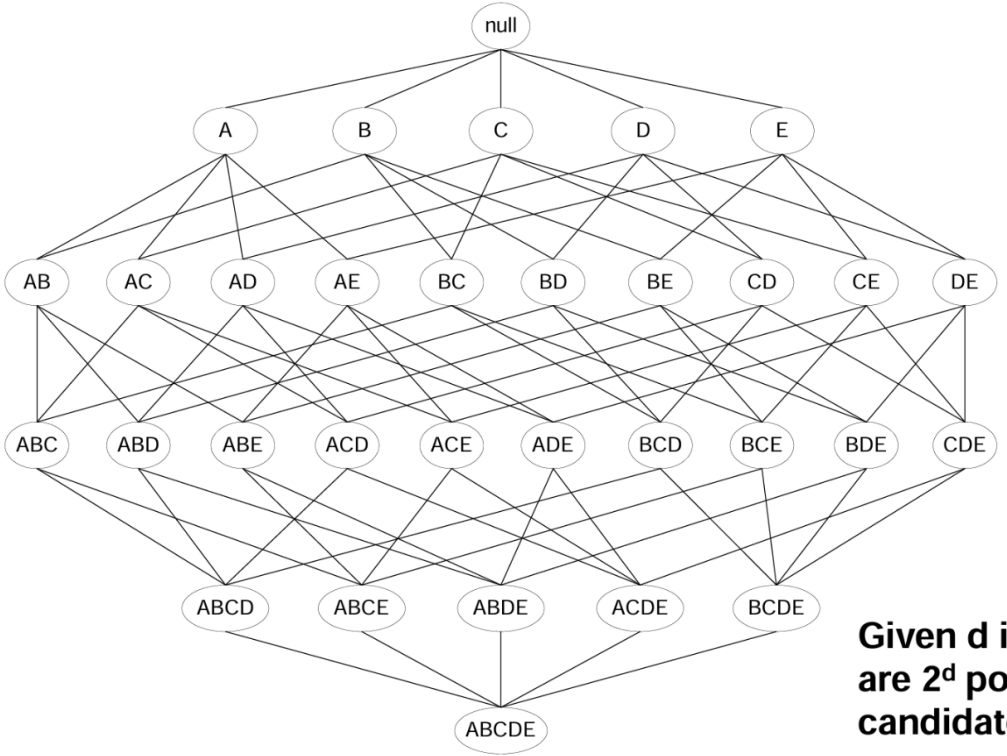
Two-step approach:

1. Frequent Itemset Generation
   - Generate all itemsets whose support minsup.

2. Rule Generation

   - Generate high confidence rules from each frequent itemset, where each rule is a binary partitioning of a frequent itemset

FREQUENT ITEMSET GENERATION IS STILL COMPUTATIONALLY EXPENSIVE



Given d items, there are 2^d possible candidate itemsets

ONE SOLUTION: REDUCING NUMBER OF CANDIDATES

Apriori principle:- If an itemset is frequent, then all of its subsets must also be frequent

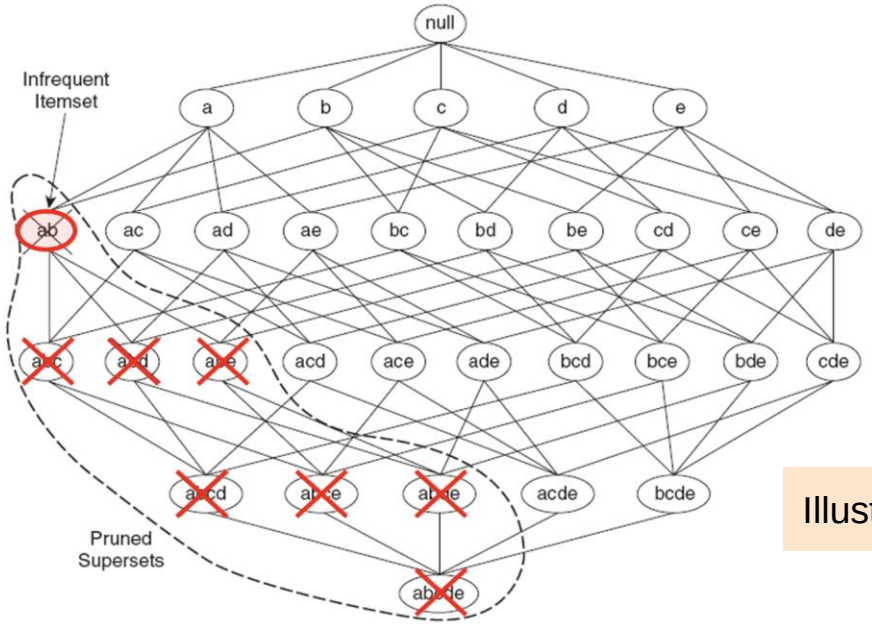


Figure 6.4. An illustration of support-based pruning. If $\{a, b\}$ is infrequent, then all supersets of $\{a, b\}$ are infrequent.

Illustrating Apriori Principle

Next:
Illustrating Apriori Principle for “Frequent Itemset Generation”

ILLUSTRATING APRIORI PRINCIPLE

1. Frequent Itemset Generation

Given:

Market-Basket transactions

TID	Items
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Minimum Support Threshold = 3

STEP 01: "1-ITEMSET GENERATION

Calculate Support Count or Frequency

Item	Count
Bread	4
Coke	2
Milk	4
Beer	3
Diaper	4
Eggs	1

ILLUSTRATING APRIORI PRINCIPLE

Frequent Itemset Generation

Given:

Market-Basket transactions

TID	Items
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Minimum Support Threshold = 3

STEP 02: "2-ITEMSET GENERATION" FROM "1-ITEMSETS

Calculate Support Count or Frequency

Items (1-itemsets)

Item	Count
Bread	4
~~Coke~~	2
Milk	4
Beer	3
Diaper	4
~~Eggs~~	1

Pairs (2-itemsets)

Itemset	Count
{Bread,Milk}	3
~~{Bread,Beer}~~	2
{Bread,Diaper}	3
~~{Milk,Beer}~~	2
{Milk,Diaper}	3
{Beer,Diaper}	3

(No need to generate candidates involving Coke or Eggs)

ILLUSTRATING APRIORI PRINCIPLE

Frequent Itemset Generation

Given:

Market-Basket transactions

TID	Items
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

STEP 03: "3-ITEMSET GENERATION" FROM "2-ITEMSETS

Items (1-itemsets)

Item	Count
Bread	4
~~Coke~~	2
Milk	4
Beer	3
Diaper	4
~~Eggs~~	1

Pairs (2-itemsets)

Itemset	Count
{Bread,Milk}	3
~~{Bread,Beer}~~	2
{Bread,Diaper}	3
~~{Milk,Beer}~~	2
{Milk,Diaper}	3
{Beer,Diaper}	3

(No need to generate candidates involving Coke or Eggs)

Minimum Support Threshold = 3

Triplets (3-itemsets)

Itemset	Count
{Bread,Milk,Diaper}	2

If every subset is considered,
${}^6C_1 + {}^6C_2 + {}^6C_3 = 41$
With support-based pruning,
6 + 6 + 1 = 13

ILLUSTRATING APRIORI PRINCIPLE

Market-Basket transactions

TID	Items
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Frequent Itemset Generation

Items (1-itemsets)

Item	Count
Bread	4
Coke	2
Milk	4
Beer	3
Diaper	4
Eggs	1

Pairs (2-itemsets)

Itemset	Count
{Bread,Milk}	3
{Bread,Beer}	2
{Bread,Diaper}	3
{Milk,Beer}	2
{Milk,Diaper}	3
{Beer,Diaper}	3

(No need to generate candidates involving Coke or Eggs)

Triplets (3-itemsets)

Itemset	Count
{Bread,Milk,Diaper}	2

Given:

Minimum Support Threshold = 3

If every subset is considered,
${}^6C_1 + {}^6C_2 + {}^6C_3 = 41$
With support-based pruning,
6 + 6 + 1 = 13

ILLUSTRATING APRIORI PRINCIPLE

Market-Basket transactions

TID	Items
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Frequent Itemset Generation

Support

Items (1-itemsets)

Item	Count
Bread	4
Coke	2
Milk	4
Beer	3
Diaper	4
Eggs	1

Pairs (2-itemsets)

Itemset	Count
{Bread,Milk}	3
{Bread,Beer}	2
{Bread,Diaper}	3
{Milk,Beer}	2
{Milk,Diaper}	3
{Beer,Diaper}	3

(No need to generate candidates involving Coke or Eggs)

Triplets (3-itemsets)

Itemset	Count
{Bread,Milk,Diaper}	2

Given:

Minimum Support Threshold = 3

If every subset is considered,
${}^6C_1 + {}^6C_2 + {}^6C_3 = 41$
With support-based pruning,
6 + 6 + 1 = 13

(No need to generate candidates involving {Bread, Bear} or {Milk Bear}

APRIORI ALGORITHM/ PRINCIPLE

- Method:
  - Let k=1
  - Generate frequent itemsets of length 1
  - Repeat until no new frequent itemsets are identified
    - Generate length (k+1) candidate itemsets from length k frequent itemsets
    - Prune candidate itemsets containing subsets of length k that are infrequent
    - Count the support of each candidate by scanning the DB
    - Eliminate candidates that are infrequent, leaving only those that are frequent

1 - FREQUENT ITEM	SUPPORT COUNT
BREAD	4
MILK	4
BEER	3
DIAPER	4

2 - FREQUENT ITEM	SUPPORT COUNT
{BREAD,MILK}	3
{BREAD,DIAPER}	3
{MILK,DIAPER}	3
{BEER, ,DIAPER}	3

Rule Generation: How to efficiently generate rules from frequent itemsets?

Given, min confidence = 70%

Confidence ($X \rightarrow Y$) : $P(Y \mid X) = P(X \cup Y) / P(X)$

We have 4 frequent itemsets: {BREAD,MILK}, {BREAD,DIAPER}, {MILK,DIAPER}, {BEER, ,DIAPER}

Therefore, candidate rules are:

For {BREAD,MILK}, there are two possibilities:

- Bread > Milk: 3/4: 0.75
- Milk > Bread: 3/4: 0.75

For {BEER, ,DIAPER}, there are two possibilities:

- Beer > Diaper: 3/3: 1 (strong!)
- Diaper > Bear: 3/4: 0.75

Continue the proceed for other all frequent datasets.

1 - FREQUENT ITEM	SUPPORT COUNT
BREAD	4
MILK	4
BEER	3
DIAPER	4

2 - FREQUENT ITEM	SUPPORT COUNT
{BREAD,MILK}	3
{BREAD,DIAPER}	3
{MILK,DIAPER}	3
{BEER, ,DIAPER}	3

3 - FREQUENT ITEM	SUPPORT COUNT
{BREAD,MILK, DIAPER}	3 (WE ARE CONSIDERING)

ANOTHER EXAMPLE: Rule Generation: LETS SAY {BREAD,MILK, DIAPER} is FREQUENT

Given, min confidence = 70%

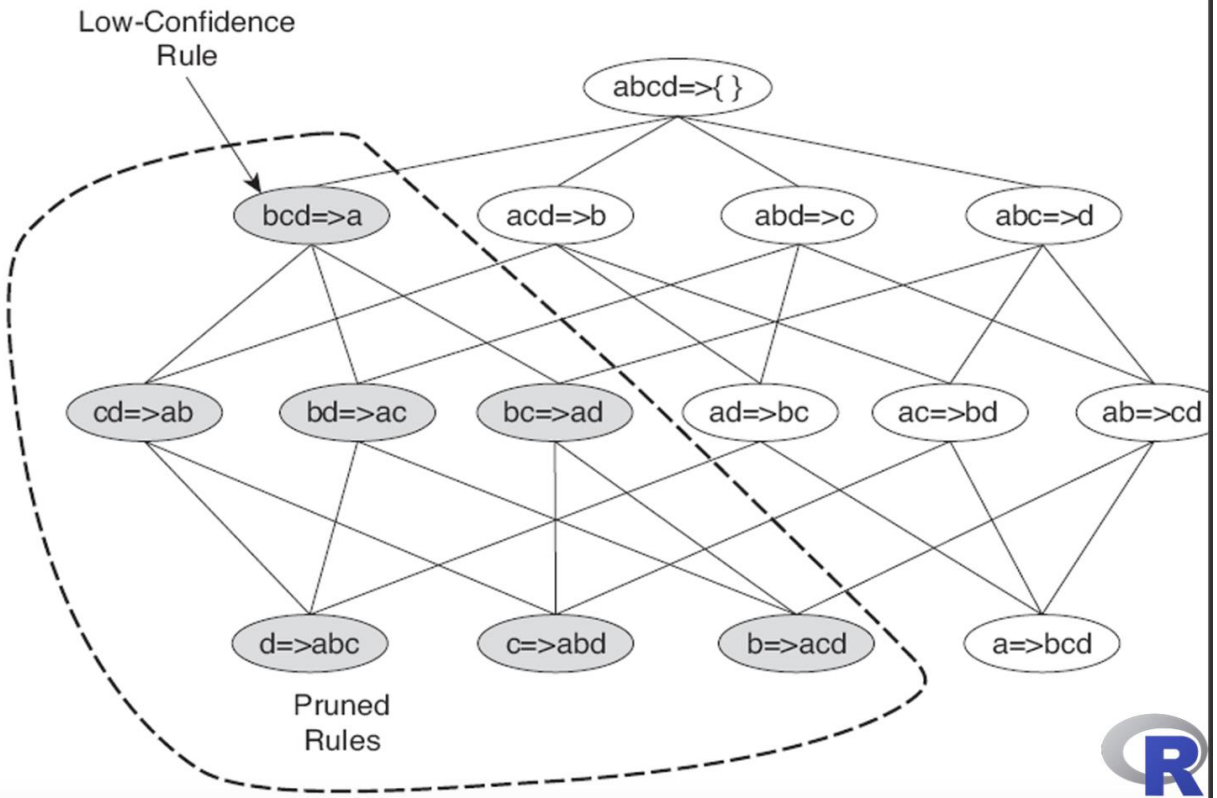
Confidence ($X \rightarrow Y$) : $P(Y \mid X) = P(X \cup Y) / P(X)$

We have 5 frequent itemsets: {BREAD,MILK}, {BREAD,DIAPER}, {MILK,DIAPER}, {BEER, ,DIAPER}, {BREAD,MILK, DIAPER}

For {BREAD,MILK ,DIAPER}, some possibilities:

- BREAD,MILK > Diaper: 3/4: 0.75
- MILK, DIAPER > BREAD: 3/4: 0.75
- BREAD, MILK > Diaper: 3/4 : 0.75

Rule Generation for Apriori Algorithm



Try if you are interested in collaborative filtering:

https://gist.github.com/marianamaroto/633da2272fea4aaf101cdff9e0393aa6#file-netflix_recommendation_system-ipynb

(NOT FOR THE EXAM)

ITEM-ITEM vs USER-USER

In practice, it has been observed that often works better than user-user item-item!

- Why? Items are simpler, users have multiple tastes (people are dynamic, their taste change while on the other hand items are constant)
- Item-based filtering finds similar items to what a person likes, no matter what others like.

Example:

Alice likes action but not romance.

Bob likes comedy and romance but not action.

Item-based filtering suggests movies based on what Alice or Bob liked, fitting their unique tastes.

Example: User Based Neighboorhood Collaborative Filtering

Scenario: Suppose User A and User B have similar movie preferences. User A has watched and liked movies X, Y, and Z.

Process: The system identifies User B as a similar user based on their past movie preferences. *If User B has also liked movies X, Y, and Z, the system recommends other movies that User B has liked but User A hasn't seen yet.*

Example: User A likes action movies, and the system identifies User B as having similar tastes. User B has watched and enjoyed movies W, X, and Y. The system recommends movie W to User A because it's likely to align with their preferences.

Example: Item Based Filtering

Scenario: Consider a scenario where a user has watched and liked movie X.

Process: The system identifies other movies that are similar to movie X based on certain criteria such as genre, actors, or director. It then recommends these similar movies to the user.

Example: User A likes the movie X, which is a science fiction film featuring actor A and directed by Director B. The system identifies other science fiction movies with actor A or directed by Director B, such as movies Y and Z, and recommends them to User A because of their similarity to movie X.

But, How do we find similarity?

- Cosine Similarity
  - Problem: Treats lack of ratings as a negative!
- Jaccard Coefficient
  - Problem: Doesn't take ratings into account
- Correlation Coefficient
  - Works pretty well!

SIMILARITY MEASURE

There are several similarity measures that can be used in a recommendation system, including:

Pearson Correlation Coefficient: It measures the linear correlation between two variables and is commonly used in recommendation systems.

Cosine Similarity: It measures the cosine of the angle between two vectors and is widely used in recommendation systems due to its ability to handle sparse data.

Jaccard Similarity: It measures the similarity between two sets and is often used in recommendation systems for binary data.

SIMILARITY MEASURE cont.

There are several similarity measures that can be used in a recommendation system, including:

Euclidean Distance: It measures the straight-line distance between two points in a multi-dimensional space and is often used in recommendation systems.

Manhattan Distance: It measures the absolute differences between two points in a multi-dimensional space and is often used in recommendation systems.