Data Analysis and Knowledge Discovery

Lecture 8

Faculty of Mathematics and Computer Science
Babeș-Bolyai University

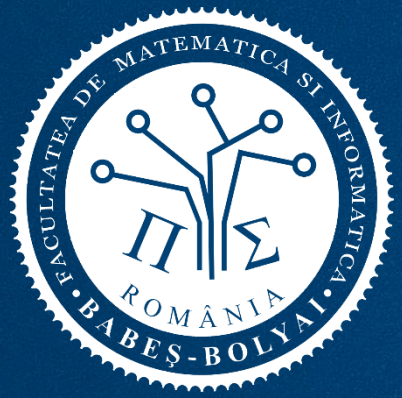
Sergiu Limboi, PhD Teaching Assistant

Motto: “Data tells you what exists. Recommender systems decide what comes next.”

Recommendation Systems: From Patterns to Personalized Decisions

AGENDA

- Warm-Up
- What is a Recommendation System?
- The Recommendation Process
- Recommendation Algorithms
- Content-based Filtering
- Collaborative Filtering
- Evaluation measures for Recommendation Systems
- Challenges in Recommendation Systems
- Teamwork Time
- Key Takeaways



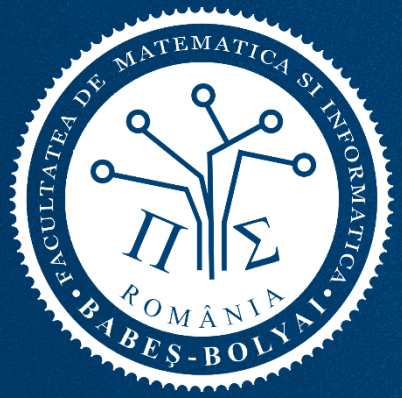
Warm-Up

Faculty of Mathematics and Computer Science

Warm-Up

Go to www.menti.com and enter the code **5566 2269**

or use the QR code



What is a Recommendation System?

Motivation

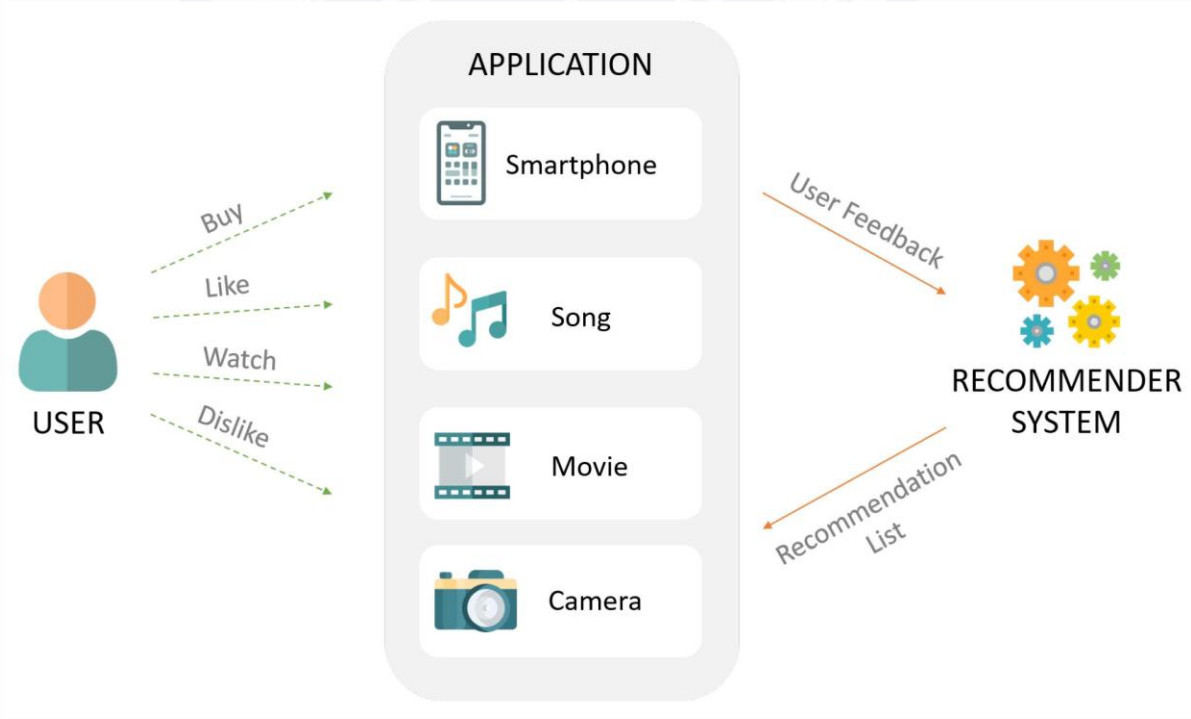
We grouped observations (clustering)

We detected anomalies

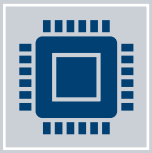
NOW: We use patterns to influence decisions

What is a Recommendation System?

- Recommendation Systems (RS) serve as vital tools for managing information overload and have significant influence over what users see, buy, or consume.



What is a Recommendation System?



Recommendation Systems (RS) encompass software tools and methodologies designed to offer guidance to users across various decision-making scenarios.

RS have proved in the latest years to be an effective solution for the **Information Overload** problem.

Basically, a RS leads the user towards new, not yet discovered items that are most likely to be of interest for the user's current need.

The Recommendation Problem

- Let's consider:
  - $U = \{u_1, u_2, \ldots, u_m\}$ denotes the group of users
  - $I = \{i_1, i_2, \ldots, i_n\}$ denotes the collection of all potential items that can be recommended.
  - $r : U \times I \rightarrow S$ quantifies the utility r(u, i) of item **i** to user **u**, S is a totally ordered set.
  - The utility is the numerical rating (ranging from 1 to 5), that measures how much **u** prefers **i** .
  - The recommendation problem is basically choosing an item $i_u \in I$ for each user $u \in U$ that maximizes the utility function r .

$$\forall u \in U, i_u = argmax(r(u, i))$$

  - To build the list of recommended items for the user, items maximizing the utility function r will be selected until the number of selected items reaches an established threshold, for example: top 10 recommendations.

The Recommendation Process

The Recommendation Process

Information Collection Phase

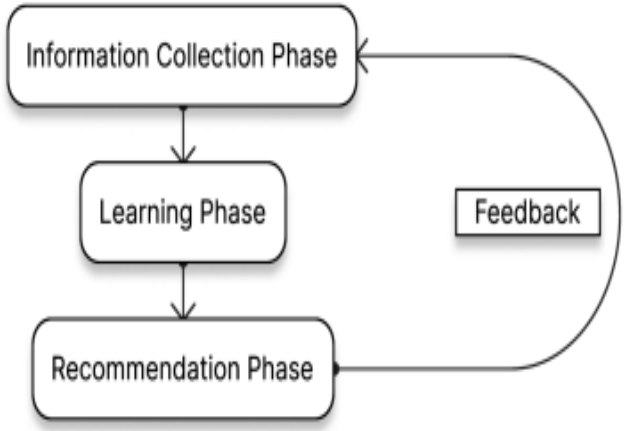
Gather user data (attributes, behaviour, interactions) to build user profiles

Learning Phase

Train models to understand user preferences and patterns

Recommendation Phase

Generate personalized suggestions based on learned preferences



User Profiling in Recommendation Systems

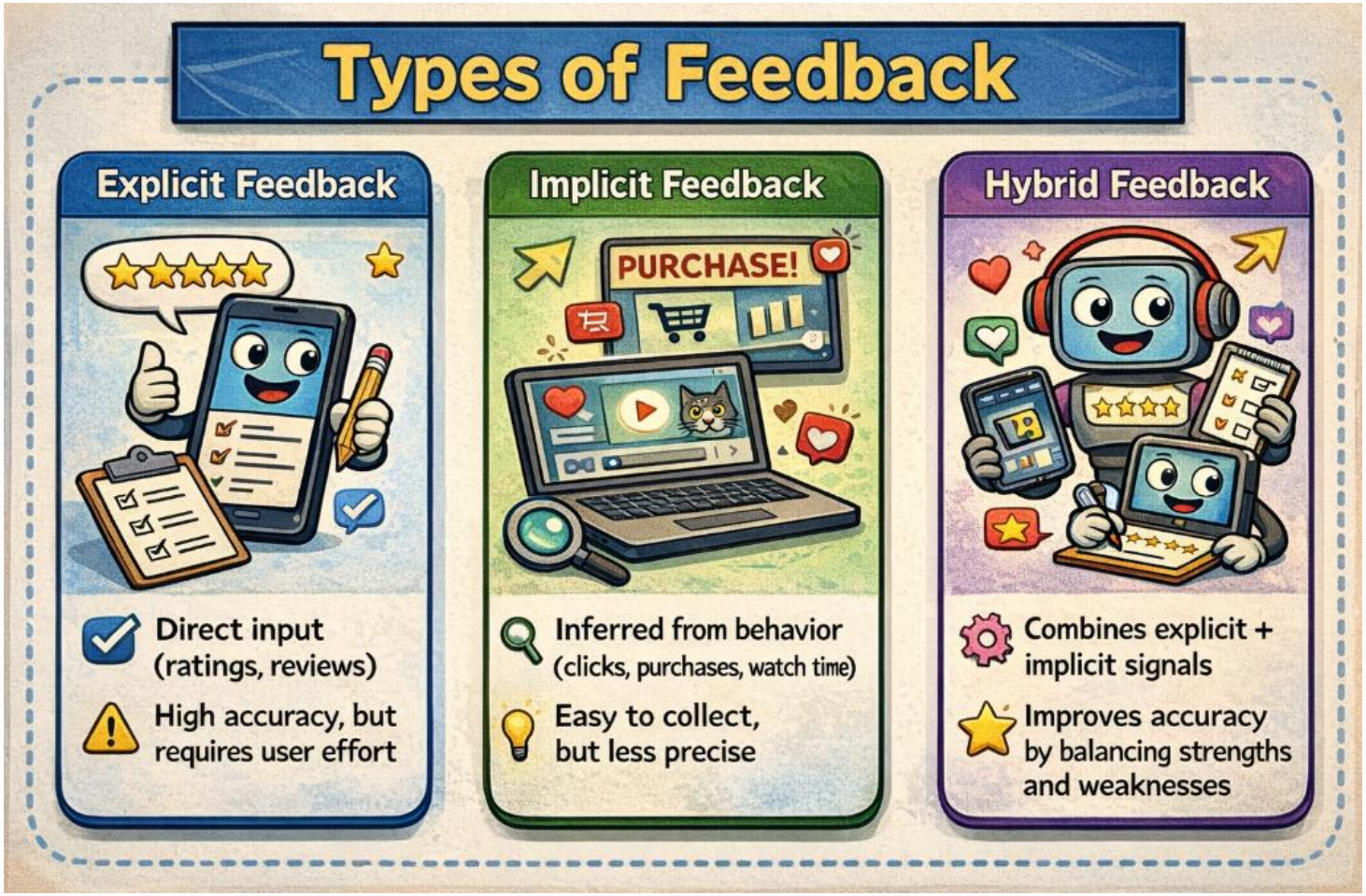
Data includes **attributes, behaviour, and accessed content**

Recommendation quality depends on **how well user profiles are built**

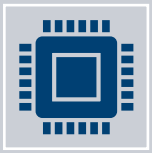
Accurate profiling enables **better understanding of user preferences**

Better understanding → **more relevant and personalized recommendations**

Types of Feedback in Recommendation Systems



Recommendation Systems' Fundamentals



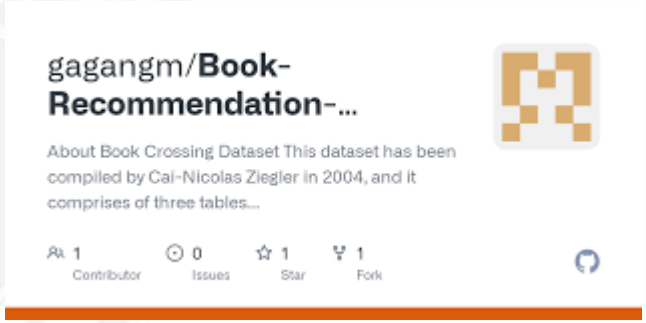
In general, the information used by RS consists: the items, the users, and the relationships between items and users.

The user's preferences and requirements are stored within their model.

A recommendation system acts as a tool that formulates suggestions by creating and utilizing these models.

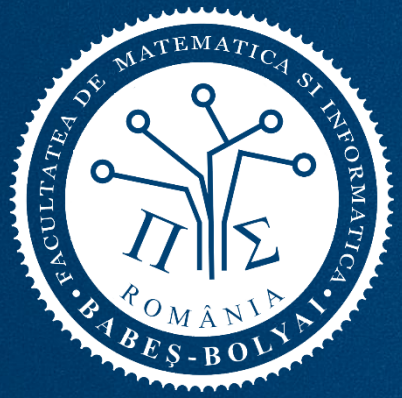
From Data to Recommendations: Popular Benchmark Datasets

- MovieLens
- Netflix Prize
- Jester
- EachMovie
- Book-crossing
- Last.FM
- Delicious



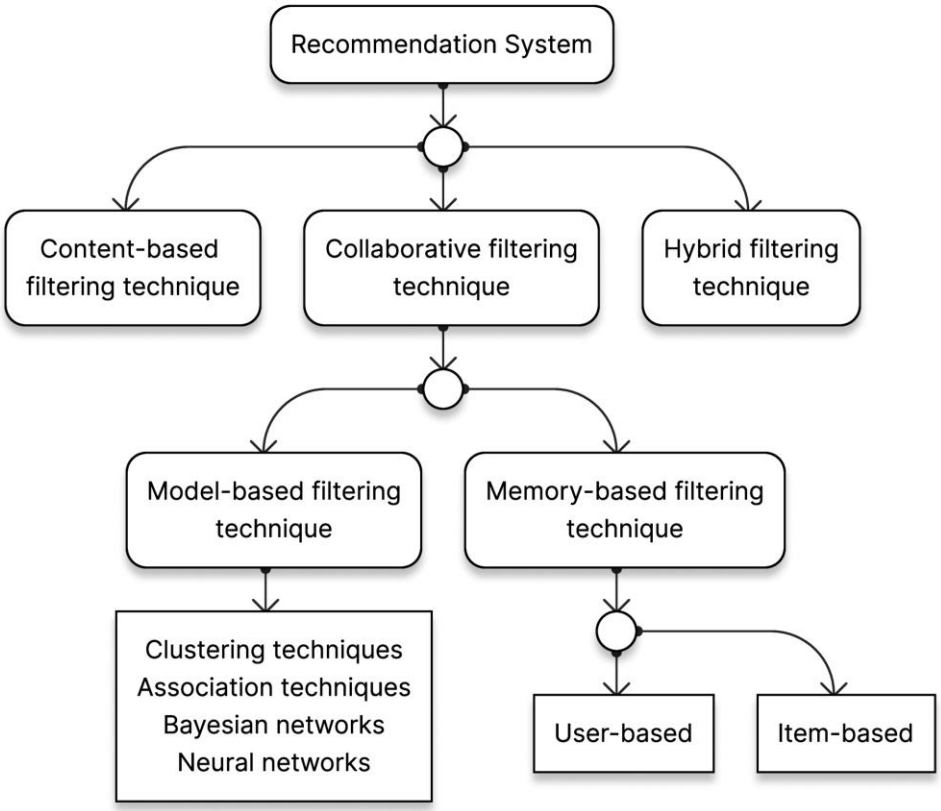
last.fm

movielens
Non-commercial, personalized movie recommendations.



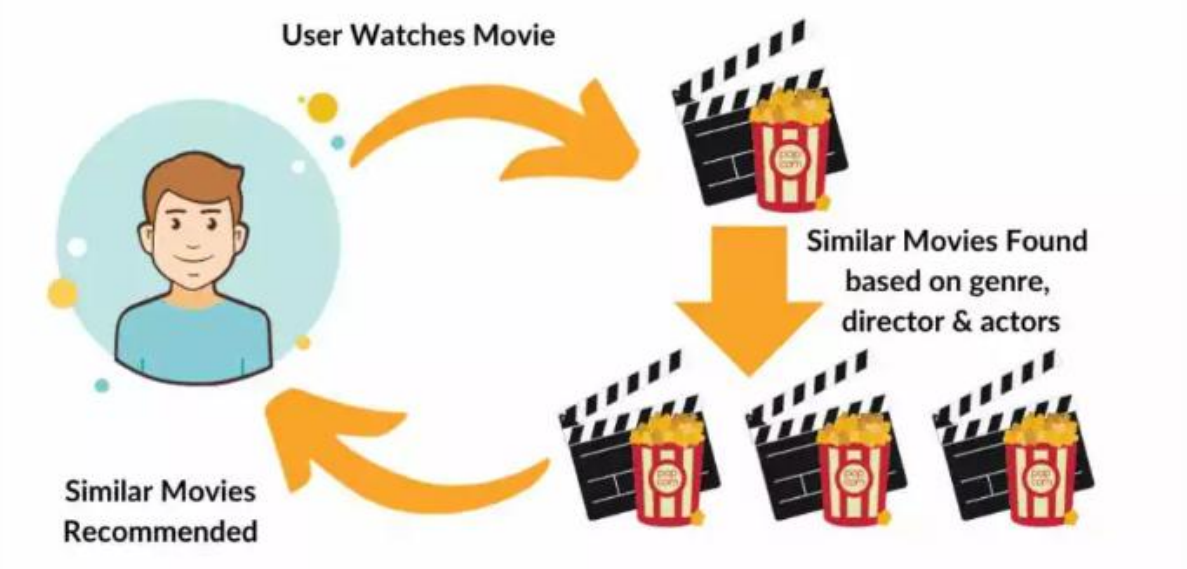
Recommendation Algorithms

Recommendation Algorithms



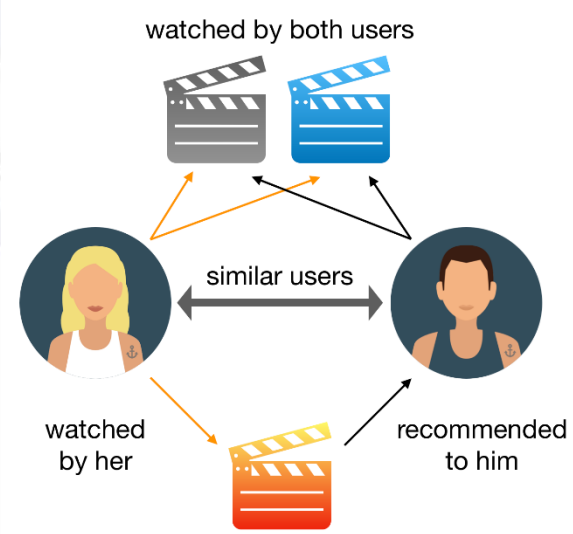
Content-based Filtering (CBF)

- Uses **item features (attributes)** to make recommendations
- Builds a **user profile** from past interactions and preferences
-
- Recommends items **similar to those the user liked before**
- Focuses on **matching user interests with item characteristics**



Collaborative Filtering (CF)

- Uses **user-item interactions** (ratings, behavior)
- Builds a **user–item matrix** of preferences
- Finds **similar users (neighborhoods)** based on behavior
- Recommends items liked by **similar users** but not yet seen

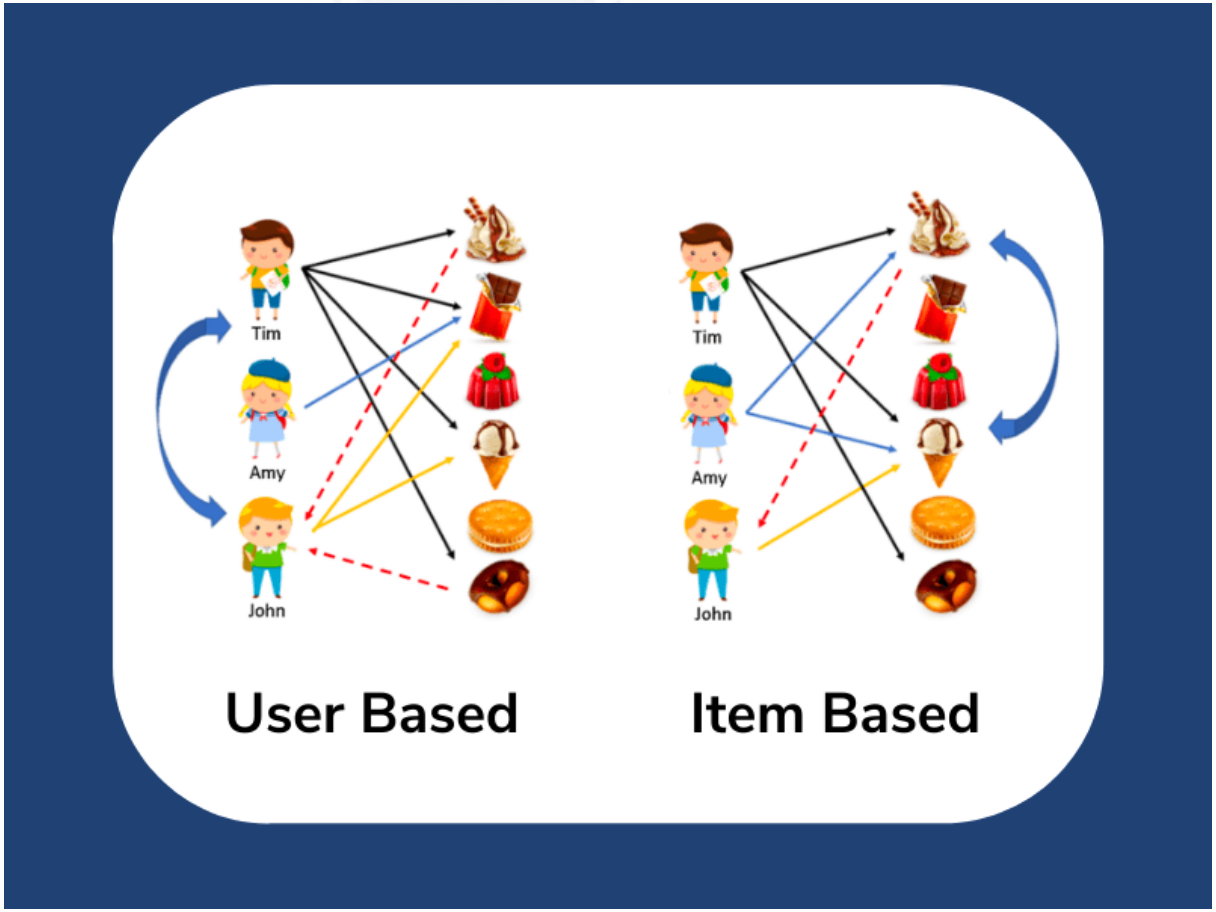


Memory-Based vs Model-Based CF

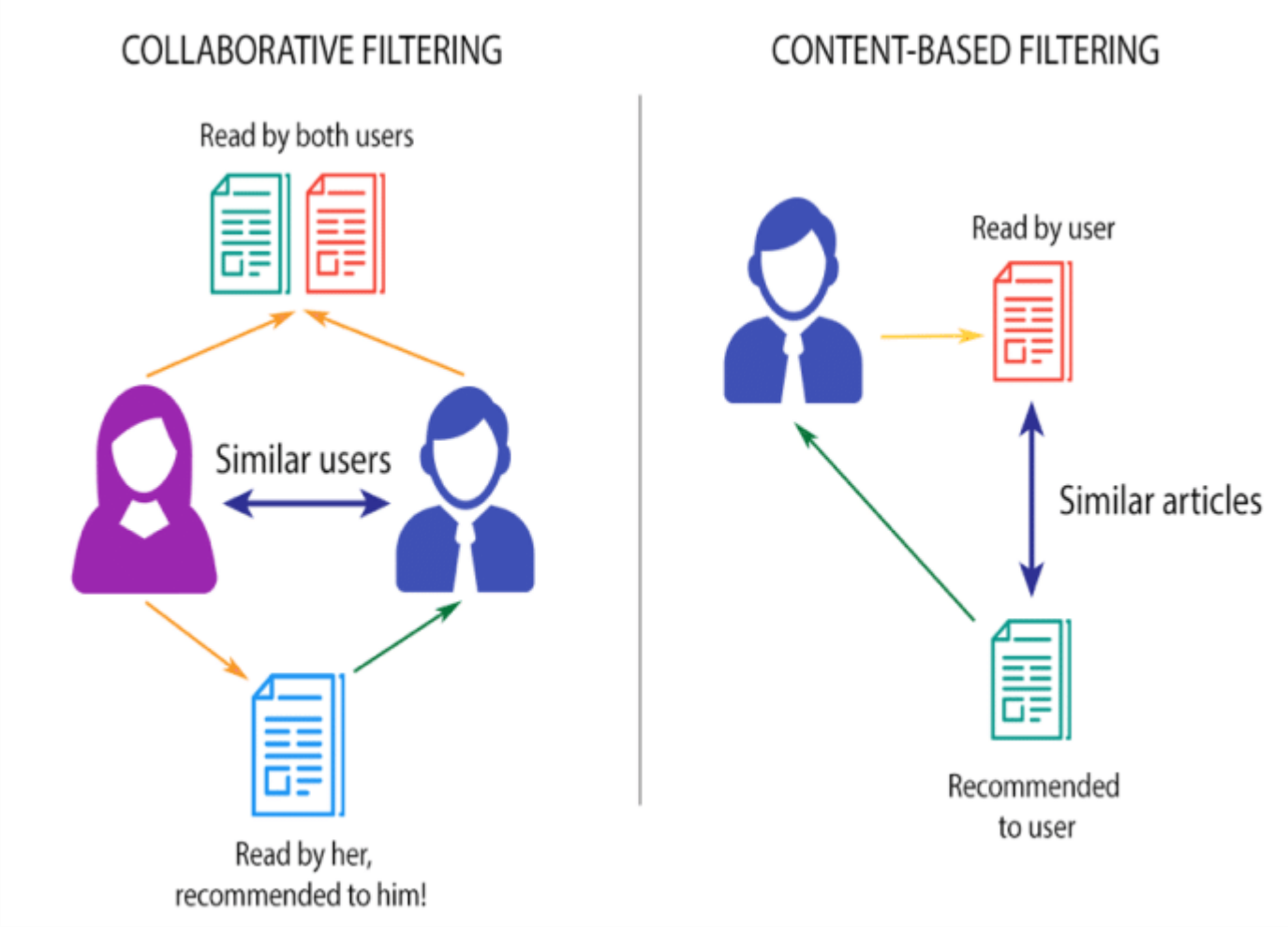
- **Memory-Based CF**
  - Uses **existing user-item ratings directly**
  - Finds **neighbors (similar users/items)**
  - Two types:
    - **User-Based (UBCF)** → similar users
    - **Item-Based (IBCF)** → similar items
  - Predictions = **weighted average of neighbors' ratings**

- **Model-Based CF**
  - Learns a **model from historical data**
  - Uses **ML / data mining techniques**
  - Faster predictions after training
  - Captures **hidden relationships**

User based vs. Item based Collaborative Filtering



Collaborative vs. Content-Based Filtering



Hybrid Filtering (HF)

- Combines **Content-Based (CBF)** and **Collaborative Filtering (CF)**

- **Hybrid Approaches**
  - **Parallel Combination**
    - Run CBF and CF separately → combine results
  - **Feature Augmentation (CBF → CF)**
    - Add content features into CF
  - **Feature Augmentation (CF → CBF)**
    - Add collaborative signals into CBF
  - **Unified Model**
    - Integrate both into a **single learning model**

Content-based Filtering

Content-based Filtering

- Recommend items similar to what you already liked.
  - You watched:
    - Matrix (Sci-Fi, Action)
    - Interstellar (Sci-Fi, Space)
  - System learns: This user likes **Sci-Fi**
  - System searches for movies with **similar features**

Content-based Filtering

- Each item = vector of features

Movie	Action	Romance	Sci-Fi
Matrix	1	0	1
Titanic	0	1	0
Interstellar	0	0	1

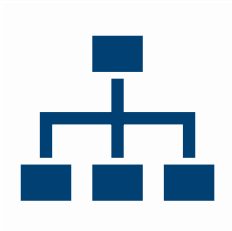
- Step 1: Build user profile
  - Combine what user liked
  - You liked: Matrix → [1,0,1] Interstellar → [0,0,1]
  - Take all liked items
  - Compute average : User = **[0.5, 0, 1]**

$$p_u = \frac{1}{|I_u|} \sum_{i \in I_u} x_i$$

Content-based Filtering

- Step 2: compute the items' score
  - System asks: "Which movie looks similar to this profile?"
  - It checks: "How close are they?"
    - Star Wars → similar ✅
    - Titanic → different ❌
  - **For each candidate item**
    - Compute: $score(i) = sim(p_u, x_i)$
  - Example of similarity measures:
    - Cosine: $\text{sim}(p_u, x_i) = \frac{p_u \cdot x_i}{\|p_u\| \|x_i\|}$
- Step 3: **Rank items**
  - Sort item by score $Top\text{-}K = \arg\max_i \ score(i)$
- **Step 4: Recommend**
  - Return top K items

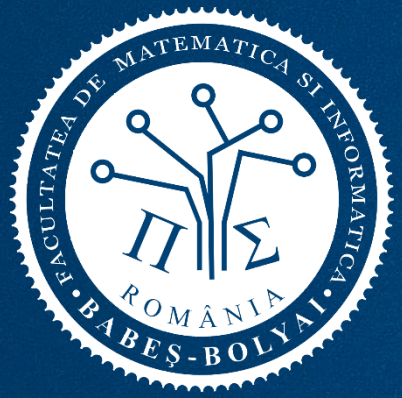
Limitations of Content-based Filtering



Over-specialization→ Only similar items

Needs good features: Bad features → bad recommendations

No discovery: Cannot suggest new types of content



Collaborative Filtering

Collaborative Filtering

- User based: “Users similar to you will like similar items.”
- Item based: “If you liked an item, you will like similar items.”

Example of user based CBF

Inception

Matrix
Interstellar

Matrix
Interstellar
Inception

Collaborative Filtering

- Popular algorithm: k-Nearest Neighbor (Knn)
- **User-based kNN approach:**
  - Identify the k **users most similar** to the target user.
  - The calculation of the prediction rating based on the ratings from the neighborhood for items not rated by the target user.
  - The selection of the top-n items recommendation list for target user.
- **Item-based kNN approach:**
  - Identify the k **items most similar** to the target item.
  - Predict the target user's rating for an unseen item using a **weighted aggregation** of the user's ratings on similar items.
  - Rank candidate items and recommend the **top-n** items with the highest predicted relevance.

Collaborative Filtering

Compute user-to-user similarity

- Cosine similarity:

$$sim(u,v) = \frac{\sum_{i \in I_{uv}} r_{ui} \cdot r_{vi}}{\sqrt{\sum r_{ui}^2} \cdot \sqrt{\sum r_{vi}^2}}$$

- Pearson correlation:

$$sim(u,v) = \frac{\sum_{i \in I_{uv}} (r_{ui} - \bar{r}_u)(r_{vi} - \bar{r}_v)}{\sqrt{\sum (r_{ui} - \bar{r}_u)^2} \cdot \sqrt{\sum (r_{vi} - \bar{r}_v)^2}}$$

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

- Jaccard similarity:

$$sim(u,v) = \frac{|I_u \cap I_v|}{|I_u \cup I_v|}$$

Collaborative Filtering

Compute item-to-item similarity

Example: Pearson correlation

$$sim(i,j) = \frac{\sum_{u \in U_{ij}} (r_{ui} - \bar{r}_u)(r_{uj} - \bar{r}_u)}{\sqrt{\sum (r_{ui} - \bar{r}_u)^2} \cdot \sqrt{\sum (r_{uj} - \bar{r}_u)^2}}$$

Collaborative- Filtering-Rating prediction formula

- User-based CF
  - N(u) = neighbors of user u
  - r_{vi} = rating given by neighbor v to item i
  - $sim(u, v)$ similarity between user u and user v

$$\hat{r}_{ui} = \frac{\sum_{v \in N(u)} sim(u,v) \cdot r_{vi}}{\sum_{v \in N(u)} |sim(u,v)|}$$

- Item-based CF
  - N(i) = neighbors of item i
  - r_{uj} =rating given by neighbor u to item j
  - $sim(i, j)$ similarity between item i and item j

$$\hat{r}_{ui} = \frac{\sum_{j \in N(i)} sim(i,j) \cdot r_{uj}}{\sum_{j \in N(i)} |sim(i,j)|}$$

Limitations of Collaborative Filtering

- Cold Start Problem
- Popularity Bias
  - Popular items get recommended more
  - Less popular items are ignored
- Limited Interpretability
  - Hard to explain why something is recommended
  - “Users like you liked this” is vague
  - Not transparent for users
- Sensitivity to Noise
  - Fake ratings / spam / bots
  - Users giving random ratings
    - Can distort similarity
    - Degrades recommendation quality
- Scalability Issues
  - Millions of users and items
  - Similarity computation is expensive
    - High memory and computation cost

Evaluation measures for Recommendation Systems

Accuracy measures

- Accuracy measures-evaluating the correctness of the recommendations provided by the system.
  - For a user u:
    - Recommended items:
      - $Rec(u) = Top{-}n\ recommended\ items$
    - Relevant items (ground truth):
      - $Rel(u) = items\ the\ user\ actually\ liked$
- Example:
  - Recommended (Top 5): A, B, C, D, E
  - Relevant: B, D, F
  - Correct recommendations = {B, D}

Accuracy measures

- Precision $$precision(u) = \frac{|recommended \cap relevant|}{|recommended|}$$
  - how frequently the RS makes correct decisions when evaluating if an item is good for the target user.
  - Example:
    - Recommended: 5 items
    - Relevant inside: 2
    - $Precision(\mathrm{u}) = \frac{2}{5} = 0.4$
  - High precision → few bad recommendations
  - Low precision → many irrelevant items

Evaluation measures for Recommendation Systems
Accuracy measures

- Recall

$$recall(u) = \frac{|recommended \cap relevant|}{|relevant|}$$

  - the proportion of correctly recommended items to the total number of relevant items

- F1-score

$$F1(u) = \frac{2(precision \cdot recall)}{precision + recall}$$

  - It is particularly valuable when there is a disparity between precision and recall.

Diversity and Novelty Measures

- Diversity and novelty measures are designed to assess the variety and newness of recommendations provided to users.
- **Intra-list Diversity**
  - Diversity **within a single recommendation list**
  - Ensures variety (genres, topics, attributes)
- **Inter-list Diversity**
  - Diversity **across different users' recommendation lists**
  - Avoids recommending the same items to everyone
- **Novelty**
  - Measures how **new or unknown** items are to the user
  - Promotes **discovery of less popular items**

Coverage and Serendipity Measures

- Coverage can be defined as how well recommendations cover the set of items (prediction coverage) and how well they can be generated for all the potential users of the system (catalogue coverage).

- Prediction coverage

$$\mathrm{PC} = \frac{|Ip|}{|I|}$$

  - where I is the set of items and Ip is the set of items for which a prediction can be made.

Coverage and Serendipity Measures

- **Weighted precision coverage (WPC)**

$$\mathrm{WPC} = \frac{\sum_{i \in I_p} r(i)}{\sum_{j \in I} r(j)}$$

- The **serendipity measures** are designed to evaluate the system's ability to provide recommendations that are surprising, unexpected, or go beyond users' typical preferences.

- **The unexpected set of recommendations**

$$\mathrm{UNEXP} = \frac{REC}{PREC}$$

  - PREC represents the set of recommendations produced by a basic prediction model,
  - REC refers to the recommendations generated by the system. If a recommendation from REC is not included in PREC, it is classified as a surprising recommendation.

Utility-based Measures

- Mean Absolute Error (MAE)

$$MAE(u) = \frac{1}{N}\sum_{i=1}^{N}|r_{u,i} - \hat{r}_{u,i}|$$

- Mean Squared Error (MSE)

$$MSE(u) = \frac{1}{N}\sum_{i=1}^{N}(r_{u,i} - \hat{r}_{u,i})^2$$

- Root Mean Squared Error (RMSE)

$$RMSE(u) = \sqrt{MSE(u)}$$

Online Measures

- Metrics that are particularly relevant in online, real-time settings where user interactions with recommendations can be directly observed.

The Click-Through Rate measures the ratio of clicks to impressions.

- In the context of recommendation systems, an impression occurs when a recommendation is presented to the user, and a click happens when the user interacts with the recommended item by, for example, clicking on it.

$$\text{CTR} = \frac{\text{Number of Clicks}}{\text{Number of Impressions}} \times 100\%$$

- A higher CTR indicates that a larger proportion of users are interacting with the recommended items

Online Measures

- **Conversion Rate** evaluates the proportion of users who take a specific action (e.g., make a purchase) after clicking on a recommended item. It measures the effectiveness of recommendations in leading users to desired outcomes

$$\text{CR} = \frac{\text{Number of Conversions}}{\text{Number of Clicks}} \times 100\%$$

- A higher CR indicates that a significant portion of users who clicked on recommendations proceeded to take the desired action.

Challenges in Recommendation Systems

Challenges in Recommendation Systems

❄️ Cold Start

- New users → no history
- New items → no ratings
- Hard to personalize at the beginning

📉 Data Sparsity

- Users interact with very few items
- Most of the user–item matrix is empty
- Difficult to find similarities

⚡ Scalability

- Millions of users & items
- Need fast, efficient algorithms
- Real-time recommendations required

Challenges in Recommendation Systems

🛡️ Robustness

- Vulnerable to fake users / spam attacks
- Need detection of malicious profiles

🔄 Pro-Activeness (Context-Aware)

- Recommendations should adapt to:
  - location
  - time
  - context (e.g., weather, behavior)
  - especially important in dynamic systems (tourism, mobile apps)

🌍 Diversity & Novelty

- Avoid recommending only similar items
- Balance: accuracy vs diversity
  - too much diversity → irrelevant
  - too little diversity→ boring recommendations

Teamwork Time

Teamwork Time-Recommendation System Design

- Each team will choose or receive **one platform** and discuss:
  - what is being recommended
  - what data the platform may use
  - what type of recommendation approach is suitable
  - what the main challenge is

- Platforms

Teamwork Time-Recommendation System Design

- What is the system recommending?
  - Examples: hotels / restaurants / attractions
- What data could be used?
  - Past clicks
  - Ratings
  - Purchases
  - Views/ watch time
- What recommendation type fits best?
  - Example:
    - content-based filtering
    - collaborative filtering
    - hybrid recommendation
    - popularity-based recommendation
- What is the biggest challenge?
  - Examples:
    - cold start
    - sparse data
    - changing preferences
    - short attention span

Teamwork Time-Recommendation System Design

- What about?

Google Maps

Business Purpose of Recommendation Systems

Why do companies use them?

- To influence user behaviour and achieve business goals

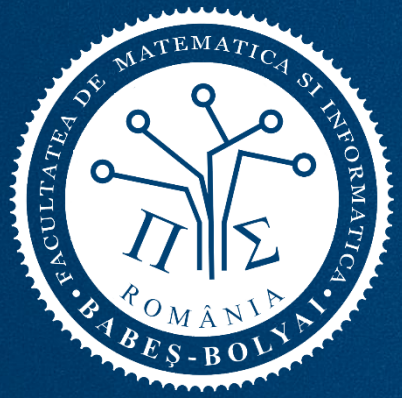
Main Objectives

- **Increase engagement**
  → keep users longer (TikTok, YouTube)
- **Increase revenue**
  → drive purchases (Amazon, e-commerce)
- **Improve conversion**
  → turn views into actions (TripAdvisor bookings)
- **Enhance user experience**
  → reduce search effort

Business Purpose of Recommendation Systems

🔄 Business Impact

- More user activity
- More time on platform
- More transactions
- Competitive advantage



Key Takeaways

Key Takeaways

Recommender systems do not just reflect preferences.

They influence and shape them.

Recommendations are everywhere.

Find patterns between users and items.

Recommendation = **decision layer of data science**

Data → patterns → predictions → **actions**

Industry impact is HUGE

Thank you for your attention — questions, thoughts, or challenges?

FACULTY OF MATHEMATICS AND COMPUTER SCIENCE
BABEȘ-BOLYAI UNIVERSITY

1 Mihail Kogălniceanu Street,
Cluj-Napoca, Cluj, România

www.cs.ubbcluj.ro