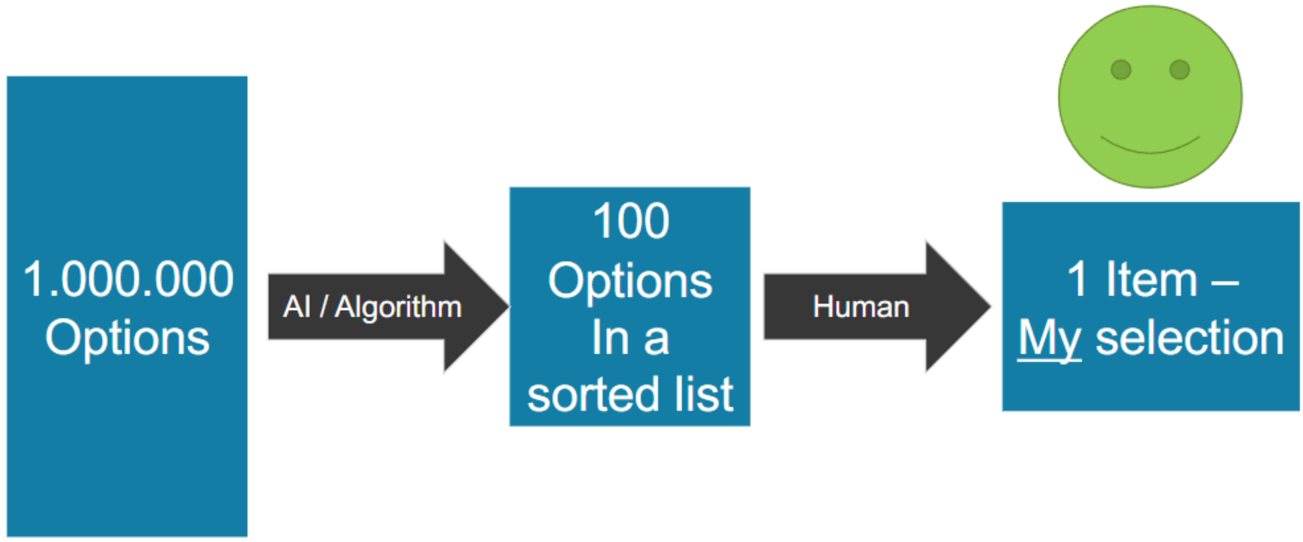
Lecture 7 - Recommender Systems

Recommender Systems (RSs) help users discover relevant content by providing personalized suggestions.

> **Definition of Recommender Systems - Ricci, Rokach & Shapira (2011)**
>
> "Recommender Systems (RSs) are software tools and techniques providing suggestions for items to be of use to a user. […] The suggestions relate to various decision-making processes, such as what items to buy, what music to listen to, or what online news to read."

Why Are Recommender Systems Needed?

- The explosion of digital content makes **manual selection impractical**.
- AI-generated content further increases the **amount of available information**.
- Without RSs, users struggle to **find relevant content efficiently**.



> **Why Random Selection Fails**
>
> Most digital content is **irrelevant** to a particular user. Without intelligent filtering, random selection leads to a **frustrating user experience**.

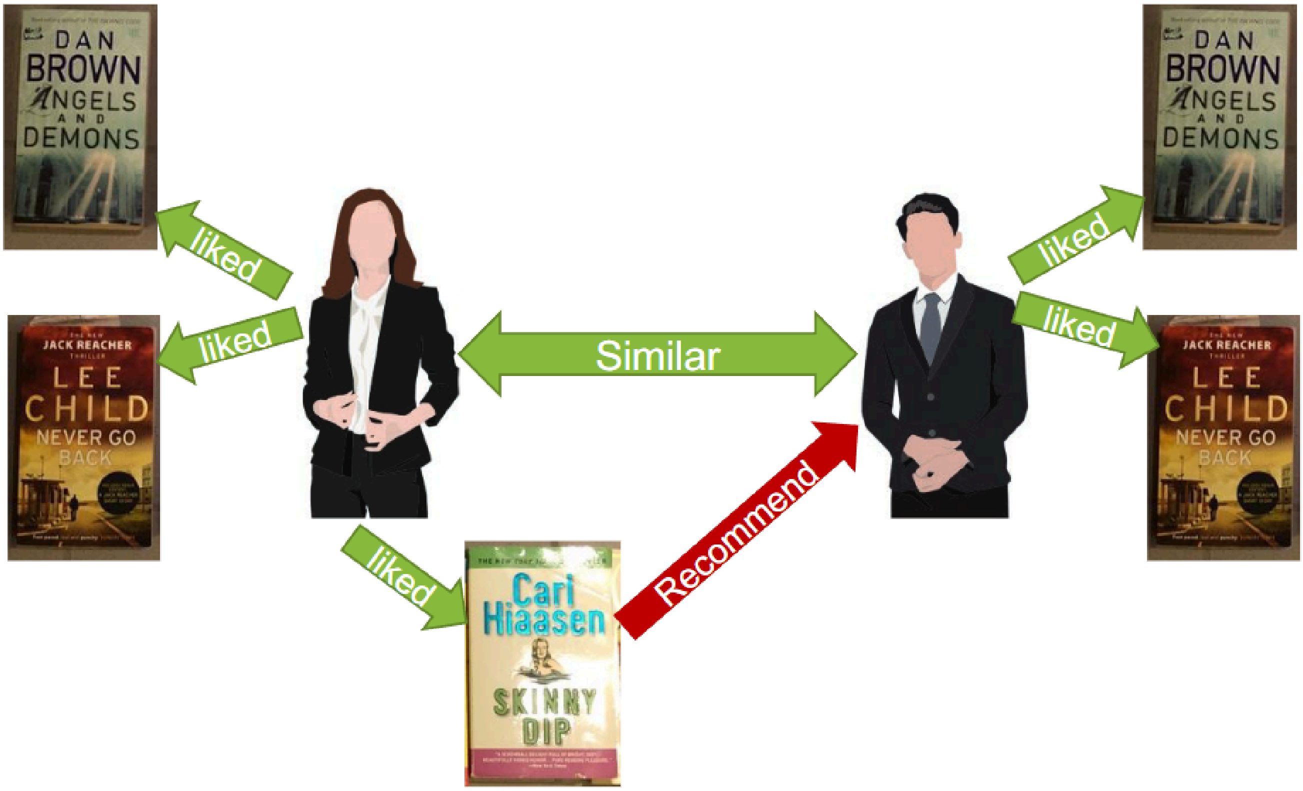
Types of Recommender Systems

Content-Based Filtering

- Items are recommended **based on features** they share with previously liked items.
- Uses metadata like **genre, author, keywords, length, or language**.
- Requires a **taxonomy or vector-based representation** of item features.

Collaborative Filtering

- Recommends items **based on user behavior and preferences**.
- Two main types:
  - **User-based:** Finds users with similar preferences.
  - **Item-based:** Finds items that receive similar ratings.
- Requires a **large dataset of user interactions**.



> ♨ **Difference Between Content-Based and Collaborative Filtering**
>
> Content-based filtering analyzes **item features**, while collaborative filtering relies on **user interactions and similarities**.

Hybrid Approaches

- Combines **content-based and collaborative** filtering techniques.
- Helps address **cold-start problems**, where user data is initially unavailable.
- Example: **Netflix uses hybrid models** to improve recommendations.

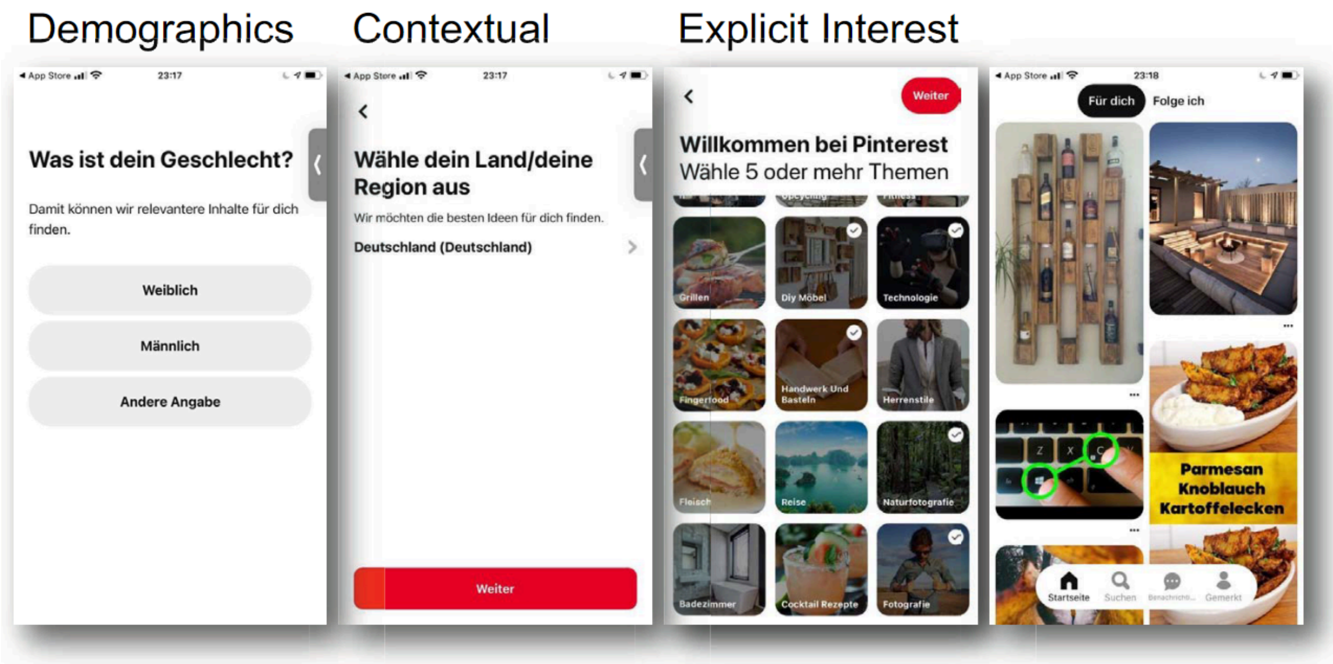
Type	Basis for Recommendation	Pros	Cons
Content-Based	Item characteristics	Works well with known items	Needs rich item metadata
Collaborative	User behavior	Learns from others' actions	Requires large user data
Hybrid	Both	Balances both approaches	Computationally expensive

---

Challenges in Recommender Systems

Cold-Start Problem

- Occurs when **new users or items** have no prior data.
- Solutions:
  - Asking users for initial preferences.
  - Using **demographic data** or pre-trained models.
  - Hybrid filtering to start with content-based and later transition to collaborative filtering.



Sparsity of Ratings

- Many users **rate only a small number of items**, making data sparse.
- Solution: Matrix factorization techniques (e.g., **SVD, ALS**) to infer missing ratings.

Bias and Filter Bubbles

- RSs may **reinforce existing preferences** and limit diversity.
- Solution: Introduce **serendipity and exploration factors** into recommendations.

> **Importance of Serendipity**
>
> Users often **enjoy discovering unexpected content** that aligns with their interests but isn't an obvious choice.

User Interface (UI) and User Experience (UX) Considerations

Transparency and Control

- Users prefer to **understand why** an item is recommended.
- RSs should allow users to **adjust preferences and provide feedback**.

Example:

- **Version 1:** "Take the train at 12:17."
- **Version 2:** "Do you prefer the **12:17 train (45 min)** or the **12:15 bus (50 min, unreliable)?"

Reducing Cognitive Load

- The **number of choices** should match the UI context.
- Example:
  - **Voice UI:** Few choices (1-3 options).
  - **Mobile app:** Moderate number of choices (5-10 items).
  - **Desktop UI:** Large selection possible (20+ items).

> **Why UI Matters**
>
> A well-designed UI makes RSs feel **helpful rather than intrusive**. If users feel **forced**, they might reject recommendations altogether.

Machine Learning in Recommender Systems

Similarity Metrics

- Used in collaborative filtering to measure user/item relationships:
  - **Cosine similarity**
  - **Pearson correlation**
  - **Euclidean distance**

Matrix Factorization Techniques

- **Singular Value Decomposition (SVD)** – Breaks down user-item interactions into latent factors.
- **Alternating Least Squares (ALS)** – Optimizes for missing data in sparse matrices.
- **Neural Networks & Deep Learning** – Used for large-scale recommendation tasks.

> **How Machine Learning Helps RSs**
>
> ML allows RSs to **predict user preferences**, optimize for **personalization**, and improve over time **based on feedback**.
