COMP9321:
Data services engineering

Week 10: Introduction to Recommender System

Term 1, 2022

By Mortada Al-Banna, CSE UNSW

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.

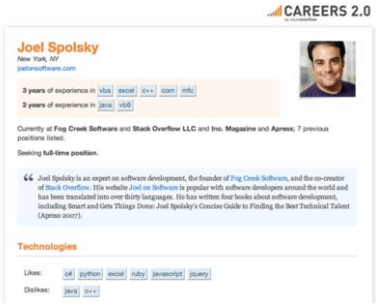
Relation to Information Retrieval:

- IR is finding material ... of an unstructured nature … that satisfies an information need from within large collections … .

(Manning et al. 2008)

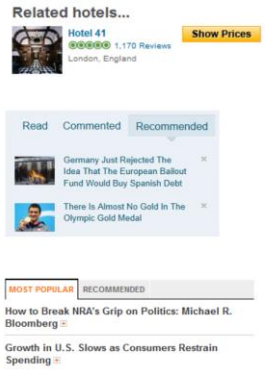
UNSW SYDNEY

People of Interest



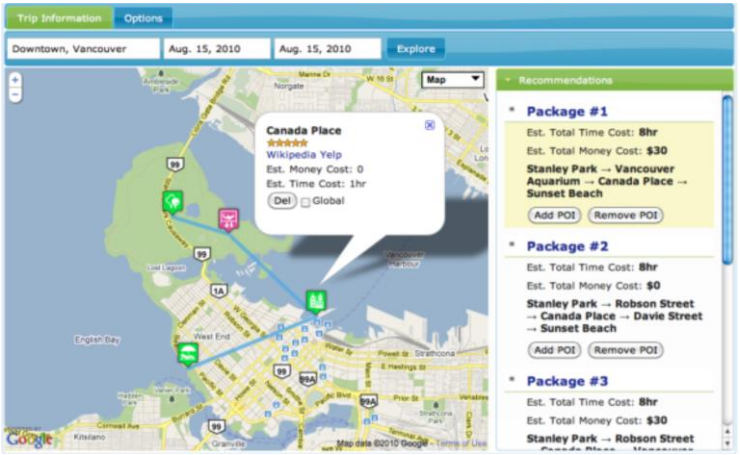
Expert

API/Software/App…



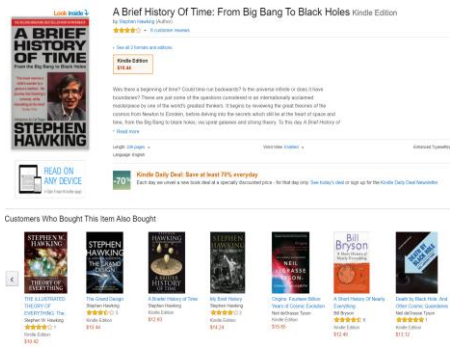
Hotel

Tweet



Trip

Music



Book

Location/Route

Medicine

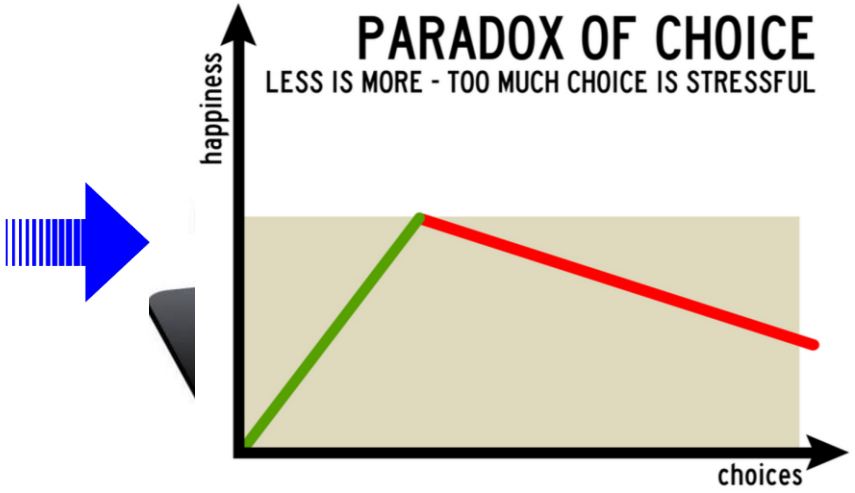
Video

UNSW SYDNEY

Why RS

- Netflix: 2/3 of the movies watched are recommended
- Google News: recommendations generate 38% more clickthrough
- Amazon: 35% sales from recommendations

Sources: Snapchat – S-1 filing. Instagram – Recode. Facebook – Q1 2016 earnings report.



"People read around 10 MB worth of material a day, hear 400 MB a day, and see 1 MB of information every second"

- The Economist, November 2006

"the average Australian is using about 258GB of data a month" – NBN Co, June 2019

JAN 2019
GLOBAL E-COMMERCE SPEND BY CATEGORY
THE TOTAL ANNUAL AMOUNT SPENT ON CONSUMER E-COMMERCE CATEGORIES AROUND THE WORLD, IN U.S. DOLLARS
FASHION & BEAUTY
$524.9 BILLION
ELECTRONICS & PHYSICAL MEDIA
$392.6 BILLION
FOOD & PERSONAL CARE
$209.5 BILLION
FURNITURE & APPLIANCES
$272.5 BILLION
TOYS, DIY & HOBBIES
$386.2 BILLION
TRAVEL (INCLUDING ACCOMMODATION)
$750.7 BILLION
DIGITAL MUSIC
$12.05 BILLION
VIDEO GAMES
$70.56 BILLION
we are social
statista
Hootsuite
we are social

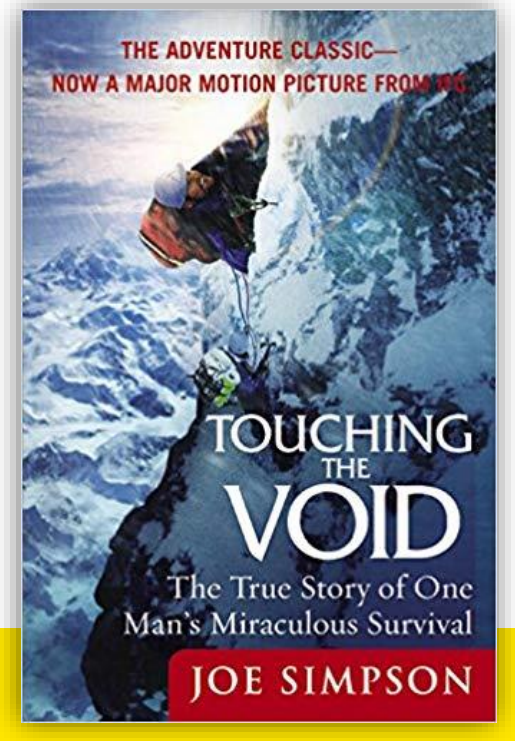
Why RS

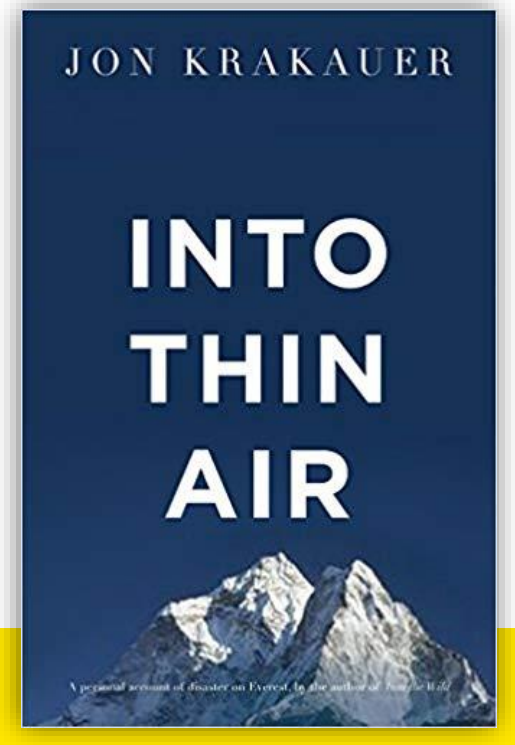
- *Recommend widely unknown items that users might actually like!"*
- *20% of items accumulate 74% of all positive ratings*

- **Long Tail Phenomenon: scarcity-driven abundance**

The Long Tail

- Shelf space is a scarce commodity for traditional retailers
  - Also: TV networks, movie theaters,…
- The web enables near-zero-cost dissemination of information about products
- More choice necessitates better filters
  - Recommendation engines (e.g., Amazon)
  - How Into Thin Air made Touching the Void a bestseller





Why RS?

- Value for the customers

– Find things that are interesting
– Narrow down the set of choices
– Explore the space of options
– Discover new things
…

- Value for the providers

– Boost profit ranging from10% - 50% caused by accurate “You Might Also Like” recommendations
– Improve retention, e.g., increase loyalty.
– Guide/Change consuming behaviors
…

Search vs. Recommender Systems

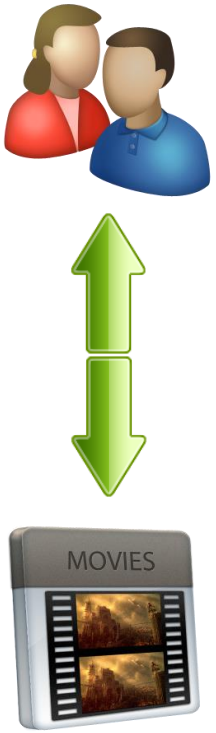
You don’t need to look for the products and services, the products/services find you

- Search allows you to search by entering the term, explicitly, while Recommender systems do not need a search term, it takes it implicitly
- Recommender system is proactive and a better user experience for finding things than traditional search

Problem domain

Recommendation systems (RS) help to match users with items

- Ease information overload
- Sales assistance (guidance, advisory, persuasion,…)



RS are software agents that elicit the interests and preferences of individual consumers […] and make recommendations accordingly. They have the potential to support and improve the quality of the decisions consumers make while searching for and selecting products online.

(Xiao & Benbasat 20071)

- Based on availability of exploitable data
- Implicit and explicit user feedback
- Domain characteristics

Purpose and success criteria (1)

Different perspectives/aspects

- Depends on domain and purpose
- No holistic evaluation scenario exists

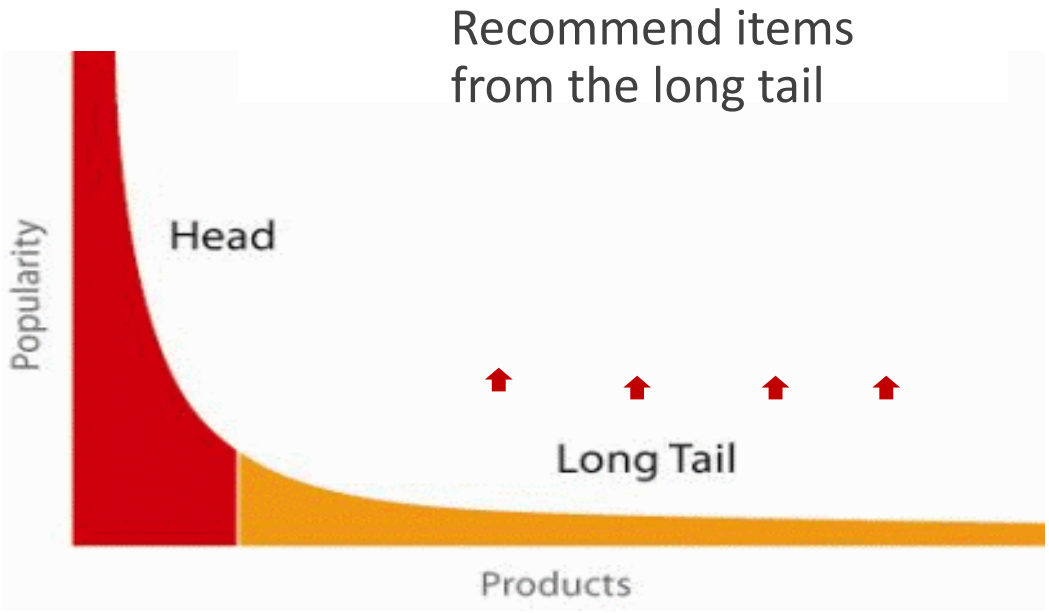
Retrieval perspective

- Reduce search costs
- Provide "correct" proposals
- Users know in advance what they want

Recommendation perspective

- Serendipity – identify items from the Long Tail
- Users did not know about existence

When does a RS do its job well?



- "Recommend widely unknown items that users might actually like!"
- 20% of items accumulate 74% of all positive ratings
- Items rated > 3 in MovieLens 100K dataset

Purpose and success criteria (2)

Prediction perspective

- Predict to what degree users like an item
- Most popular evaluation scenario in research

Interaction perspective

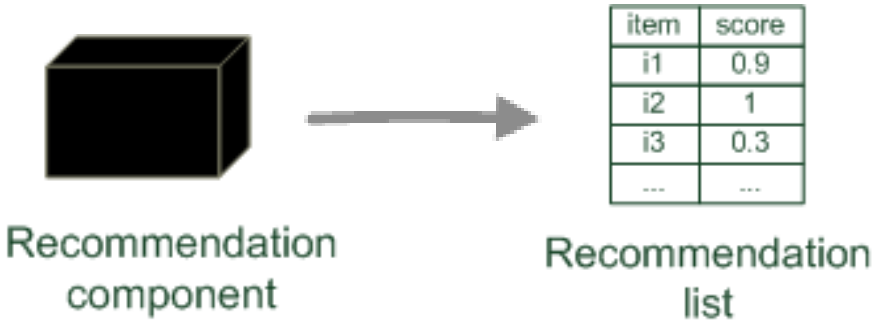
- Give users a "good feeling"
- Educate users about the product domain
- Convince/persuade users - explain

Finally, conversion perspective

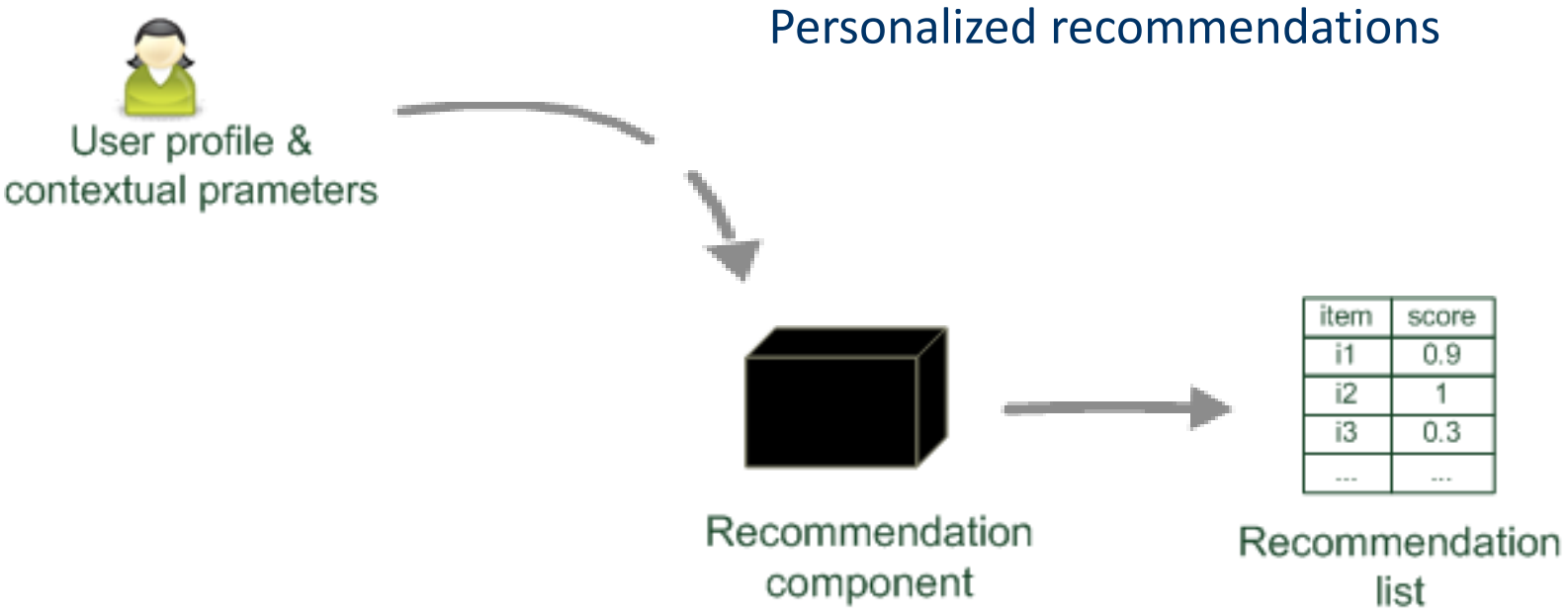
- Commercial situations
- Increase "hit", "clickthrough", "lookers to bookers" rates
- Optimize sales margins and profit

Paradigms of recommender systems

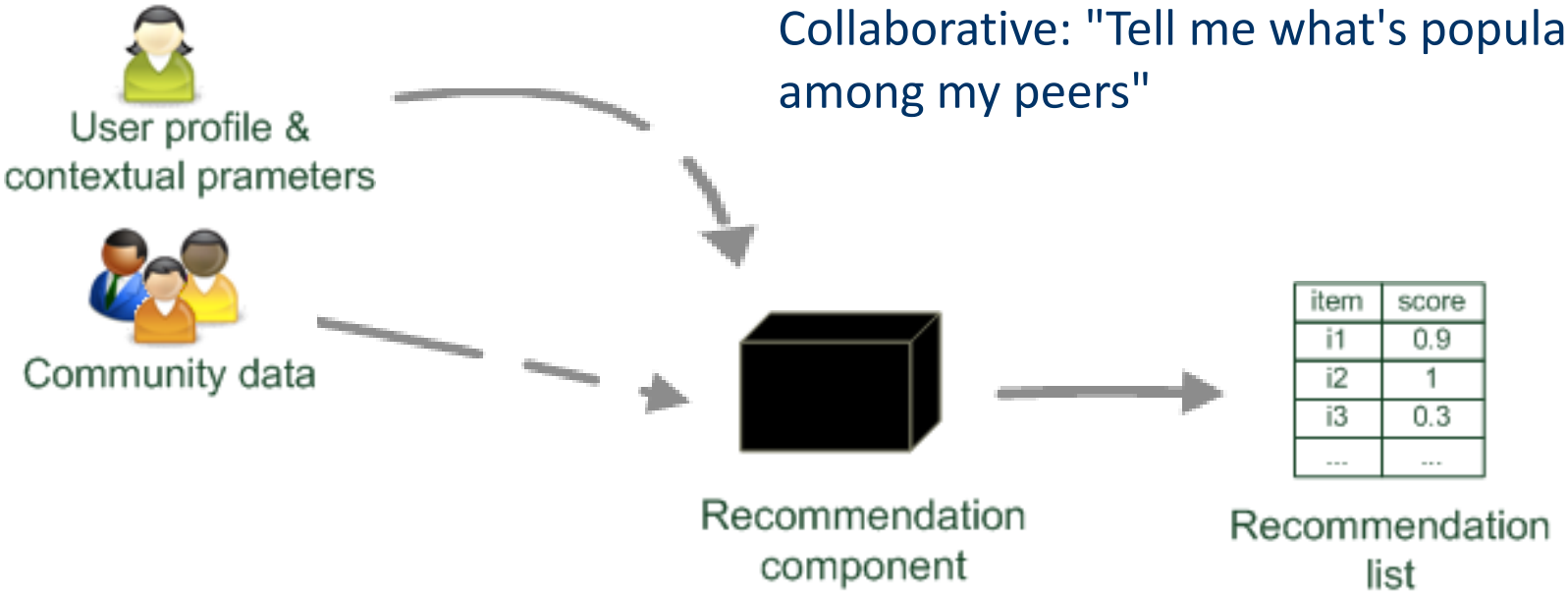
Recommender systems reduce information overload by estimating relevance



Paradigms of recommender systems

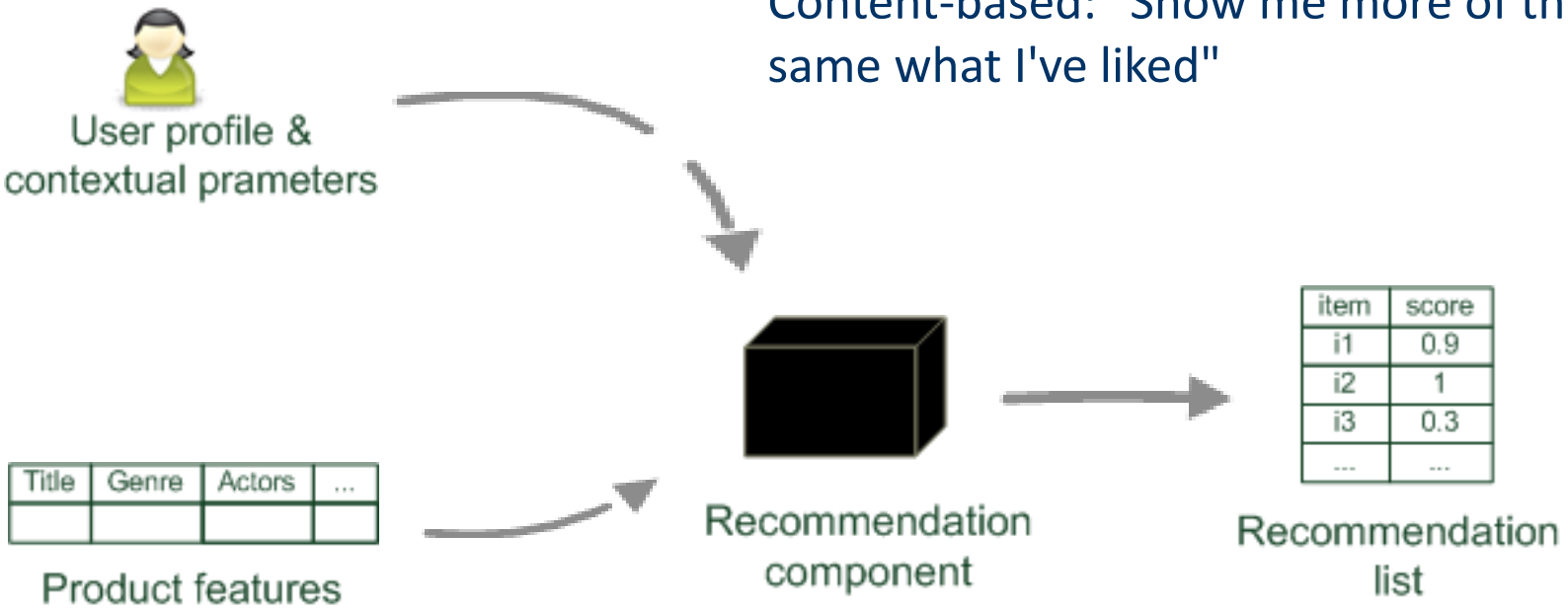


Paradigms of recommender systems

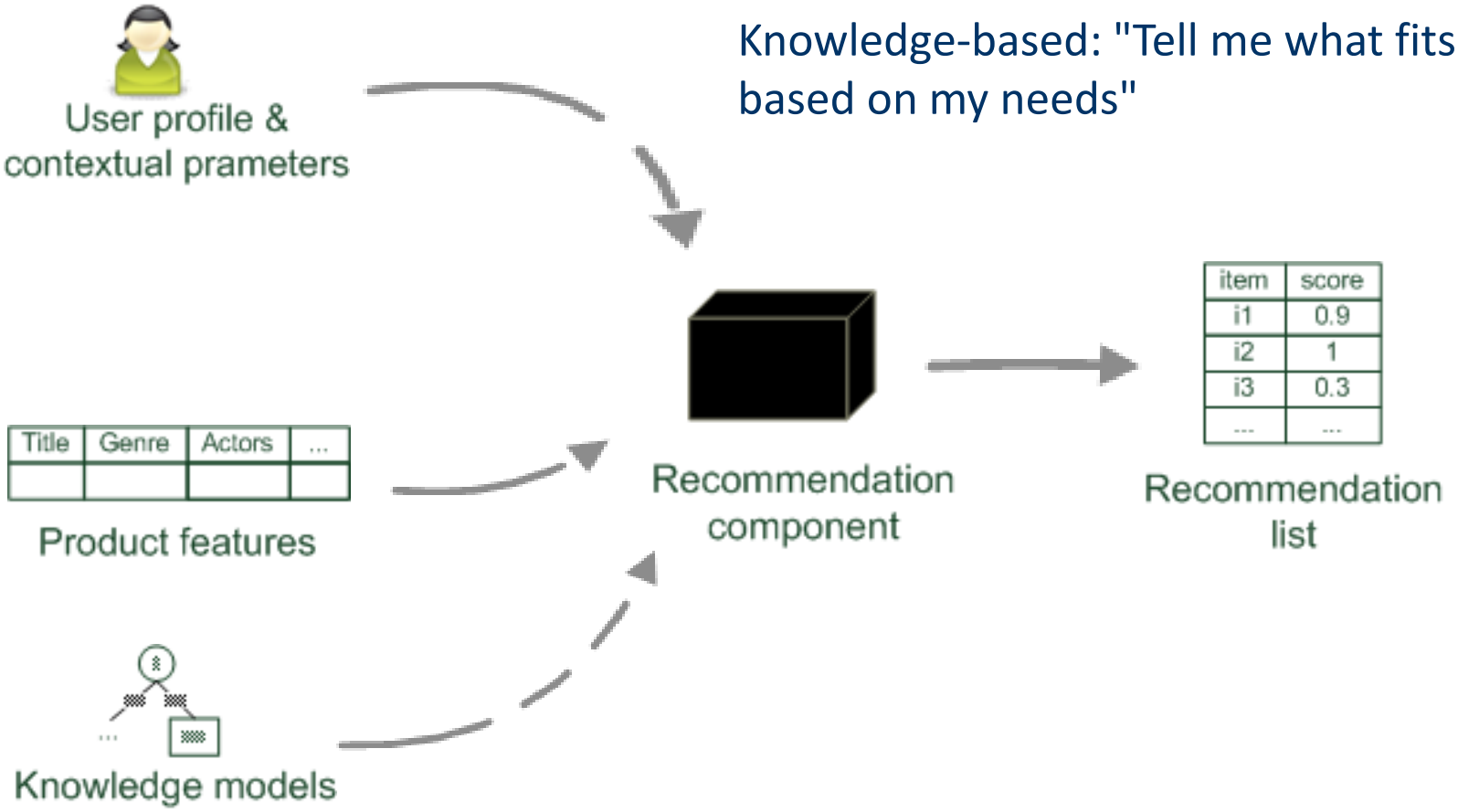


Paradigms of recommender systems

Content-based: "Show me more of the same what I've liked"

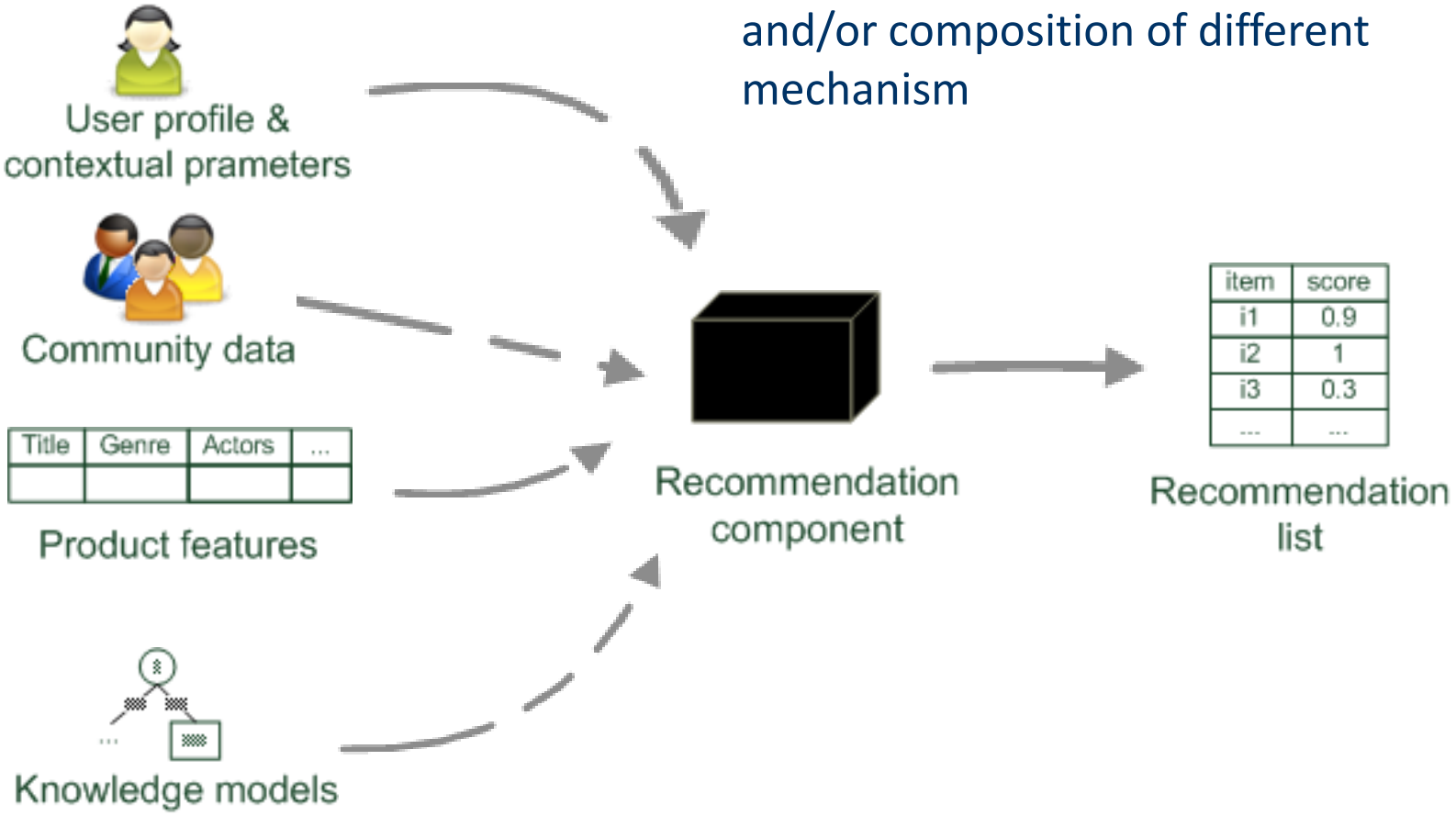


Paradigms of recommender systems



Paradigms of recommender systems

Hybrid: combinations of various inputs and/or composition of different mechanism



Collaborative Filtering

Introduction to Recommender Systems

UNSW SYDNEY

Collaborative Filtering (CF)

The most prominent approach to generate recommendations

- used by large, commercial e-commerce sites
- well-understood, various algorithms and variations exist
- applicable in many domains (book, movies, DVDs, ..)

Approach

- use the "wisdom of the crowd" to recommend items

Basic assumption and idea

- Users give ratings to catalog items (implicitly or explicitly)
- Customers who had similar tastes in the past, will have similar tastes in the future

Pure CF Approaches

Input

- Only a matrix of given user–item ratings

Output types

- A (numerical) prediction indicating to what degree the current user will like or dislike a certain item
- A top-N list of recommended items

User-based nearest-neighbor collaborative filtering (1)

The basic technique

- Given an "active user" (Alice) and an item i not yet seen by Alice
  - find a set of users (peers/nearest neighbors) who liked the same items as Alice in the past **and** who have rated item i
  - use, e.g. the average of their ratings to predict, if Alice will like item i
  - do this for all items Alice has not seen and recommend the best-rated

Basic assumption and idea

- If users had similar tastes in the past they will have similar tastes in the future
- User preferences remain stable and consistent over time

User-based nearest-neighbor collaborative filtering (2)

Example

- A database of ratings of the current user, Alice, and some other users is given:

	Item1	Item2	Item3	Item4	Item5
Alice	5	3	4	4	?
User1	3	1	2	3	3
User2	4	3	4	3	5
User3	3	3	1	5	4
User4	1	5	5	2	1

- Determine whether Alice will like or dislike *Item5*, which Alice has not yet rated or seen

User-based nearest-neighbor collaborative filtering (3)

Some first questions

- How do we measure similarity?
- How many neighbors should we consider?
- How do we generate a prediction from the neighbors' ratings?

	Item1	Item2	Item3	Item4	Item5
Alice	5	3	4	4	?
User1	3	1	2	3	3
User2	4	3	4	3	5
User3	3	3	1	5	4
User4	1	5	5	2	1

Measuring user similarity (1)

A popular similarity measure in user-based CF: **Pearson correlation**

a, b : users

$r_{a,p}$: rating of user a for item p

P : set of items, rated both by a and b

- Possible similarity values between -1 and 1

$$\boldsymbol{sim(a,b)} = \frac{\sum_{p \in P}(\boldsymbol{r_{a,p}} - \bar{\boldsymbol{r}}_{\boldsymbol{a}})(\boldsymbol{r_{b,p}} - \bar{\boldsymbol{r}}_{\boldsymbol{b}})}{\sqrt{\sum_{p \in P}(\boldsymbol{r_{a,p}} - \bar{\boldsymbol{r}}_{\boldsymbol{a}})^2}\sqrt{\sum_{p \in P}(\boldsymbol{r_{b,p}} - \bar{\boldsymbol{r}}_{\boldsymbol{b}})^2}}$$

Measuring user similarity (2)

A popular similarity measure in user-based CF: **Pearson correlation**

a, b : users

$r_{a,p}$: rating of user a for item p

P : set of items, rated both by a and b

- Possible similarity values between -1 and 1

	Item1	Item2	Item3	Item4	Item5
Alice	5	3	4	4	?
User1	3	1	2	3	3
User2	4	3	4	3	5
User3	3	3	1	5	4
User4	1	5	5	2	1

sim = 0.85

sim = 0.00

sim = 0.70

sim = -0.79

Making predictions

A common prediction function:

$$\boldsymbol{pred}(\boldsymbol{a},\boldsymbol{p}) = \overline{\boldsymbol{r_a}} + \frac{\sum_{\boldsymbol{b}\in\boldsymbol{N}} \boldsymbol{sim}(\boldsymbol{a},\boldsymbol{b}) * (\boldsymbol{r_{b,p}} - \overline{\boldsymbol{r_b}})}{\sum_{\boldsymbol{b}\in\boldsymbol{N}} \boldsymbol{sim}(\boldsymbol{a},\boldsymbol{b})}$$

Calculate, whether the neighbors' ratings for the unseen item i are higher or lower than their average

Combine the rating differences – use the similarity with a as a weight

Add/subtract the neighbors' bias from the active user's average and use this as a prediction

Improving the metrics / prediction function

Not all neighbor ratings might be equally "valuable"

- Agreement on commonly liked items is not so informative as agreement on controversial items
- **Possible solution**: Give more weight to items that have a higher variance

Value of number of co-rated items

- Use "significance weighting", by e.g., linearly reducing the weight when the number of co-rated items is low

Case amplification

- Intuition: Give more weight to "very similar" neighbors, i.e., where the similarity value is close to 1.

Neighborhood selection

- Use similarity threshold or fixed number of neighbors

Item-based collaborative filtering

Basic idea:

- Use the similarity between items (and not users) to make predictions

Example:

- Look for items that are similar to Item5
- Take Alice's ratings for these items to predict the rating for Item5

	Item1	Item2	Item3	Item4	Item5
Alice	5	3	4	4	?
User1	3	1	2	3	3
User2	4	3	4	3	5
User3	3	3	1	5	4
User4	1	5	5	2	1

Pre-processing for item-based filtering

Item-based filtering does not solve the scalability problem itself

Pre-processing approach by Amazon.com (in 2003)

- Calculate all pair-wise item similarities in advance
- The neighborhood to be used at run-time is typically rather small, because only items are taken into account which the user has rated
- Item similarities are supposed to be more stable than user similarities

Memory requirements

- Up to N^2 pair-wise similarities to be memorized (N = number of items) in theory
- In practice, this is significantly lower (items with no co-ratings)
- Further reductions possible
  - Minimum threshold for co-ratings
  - Limit the neighborhood size (might affect recommendation accuracy)

More on ratings – Implicit ratings

Typically collected by the web shop or application in which the recommender system is embedded

When a customer buys an item, for instance, many recommender systems interpret this behavior as a positive rating

Clicks, page views, time spent on some page, demo downloads …

Implicit ratings can be collected constantly and do not require additional efforts from the side of the user

Main problem

- One cannot be sure whether the user behavior is correctly interpreted
- For example, a user might not like all the books he or she has bought; the user also might have bought a book for someone else

Implicit ratings can be used in addition to explicit ones; question of correctness of interpretation

Data sparsity

For example, the Netflix Prize rating data in the user/movie rating matrix, you get 500,000 * 17000 = 8500M positions Out of which only 100M are not 0's

Cold start problem

- How to recommend new items? What to recommend to new users?

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
- Default voting: assign default values to items that only one of the two users to be compared has rated (Breese et al. 1998)

Alternatives

- Use better algorithms (beyond nearest-neighbor approaches)
- Example:
  - In nearest-neighbor approaches, the set of sufficiently similar neighbors might be too small to make good predictions
  - Assume "transitivity" of neighborhoods

Limitations of CF

- Cold Start: have enough other users already in the system to find a match. New items need to get enough ratings.
- Popularity Bias: hard to recommend items to someone with unique tastes
- Trends to recommend popular items (items from the tail don’t get so much data)

Content-based recommendation

Introduction to Recommender Systems

UNSW SYDNEY

Content-based recommendation

While CF – methods do not require any information about the items,

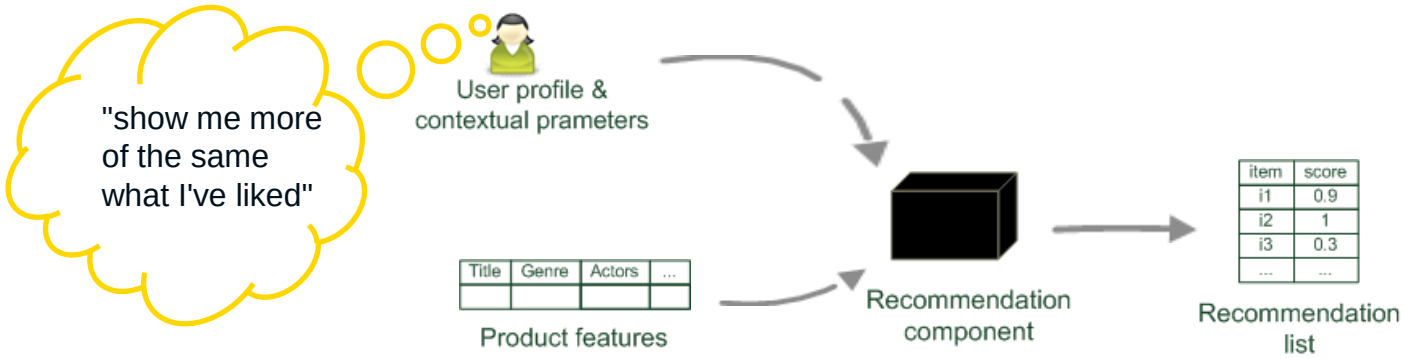
- it might be reasonable to exploit such information; and
- recommend fantasy novels to people who liked fantasy novels in the past

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



What is the "content"?

Most CB-recommendation techniques were applied to recommending text documents.

- Like web pages or newsgroup messages for example.

Content of items can also be represented as text documents.

- With textual descriptions of their basic characteristics.
- Structured: Each item is described by the same set of attributes

Title	Genre	Author	Type	Price	Keywords
The Night of the Gun	Memoir	David Carr	Paperback	29.90	Press and journalism, drug addiction, personal memoirs, New York
The Lace Reader	Fiction, Mystery	Brunonia Barry	Hardcover	49.90	American contemporary fiction, detective, historical
Into the Fire	Romance, Suspense	Suzanne Brockmann	Hardcover	45.90	American fiction, murder, neo-Nazism

- Unstructured: free-text description.

Content representation and item similarities

- **Item representation**

Title	Genre	Author	Type	Price	Keywords
The Night of the Gun	Memoir	David Carr	Paperback	29.90	Press and journalism, drug addiction, personal memoirs, New York
The Lace Reader	Fiction, Mystery	Brunonia Barry	Hardcover	49.90	American contemporary fiction, detective, historical
Into the Fire	Romance, Suspense	Suzanne Brockmann	Hardcover	45.90	American fiction, murder, neo-Nazism

- **User profile**

Title	Genre	Author	Type	Price	Keywords
…	Fiction	Brunonia, Barry, Ken Follett	Paperback	25.65	Detective, murder, New York

$keywords(b_j)$ describes Book b_j with a set of keywords

Simple approach

- Compute the similarity of an unseen item with the user profile based on the keyword overlap
  (e.g. using the Dice coefficient)

$$\frac{2 \times |keywords(b_i) \cap keywords(b_j)|}{|keywords(b_i)| + |keywords(b_j)|}$$

- Or use and combine multiple metrics

UNSW SYDNEY

Recommending items

Simple method: nearest neighbors

- Given a set of documents D already rated by the user (like/dislike)
  - Either explicitly via user interface
  - Or implicitly by monitoring user's behavior
- Find the n nearest neighbors of an not-yet-seen item i in D
  - Use similarity measures (like cosine similarity) to capture similarity of two documents
- Take these neighbors to predict a rating for i
  - e.g. $k = 5$ most similar items to i. ➡
    4 of k items were liked by current user item i will also be liked by this user

- Variations:
  - Varying neighborhood size k
  - lower/upper similarity thresholds to prevent system from recommending items the user already has seen
- Good to model short-term interests / follow-up stories
- Used in combination with method to model long-term preferences

Limitations of content-based recommendation methods

Keywords alone may not be sufficient to judge quality/relevance of a document or web page

- up-to-date-ness, usability, aesthetics, writing style
- content may also be limited / too short
- content may not be automatically extractable (multimedia)

Ramp-up phase required

- Some training data is still required
- Web 2.0: Use other sources to learn the user preferences

Overspecialization

- Algorithms tend to propose "more of the same"
- Or: too similar news items

Q&A

UNSW SYDNEY