BIG DATA
Analytics & Management

Lecture 8 (04/03, 04/05): Recommender Systems

Decisions, Operations & Information Technologies
Robert H. Smith School of Business
Spring, 2017

Outline

UNIVERSITY OF MARYLAND
ROBERT H. SMITH SCHOOL OF BUSINESS

- Introduction to recommender systems
- How do they work?
  - ❑Collaborative filtering (CF)
  - ❑MapReduce-based CF
  - ❑Content-based filtering
  - ❑Knowledge-Based recommendations
  - ❑Hybridization strategies
- How to measure their success?
  - ❑Evaluation techniques

Why using Recommender Systems?

- Value for the customer
  - ❑ Find things that are interesting
  - ❑ Narrow down the set of choices
  - ❑ Help me explore the space of options
  - ❑ Discover new things
  - ❑ …

- Value for the provider
  - ❑ Additional and probably unique personalized service for the customer
  - ❑ Increase trust and customer loyalty
  - ❑ Increase sales, click trough rates, conversion etc.
  - ❑ Opportunities for promotion, persuasion
  - ❑ …

Real-world check

UNIVERSITY OF MARYLAND
ROBERT H. SMITH SCHOOL OF BUSINESS

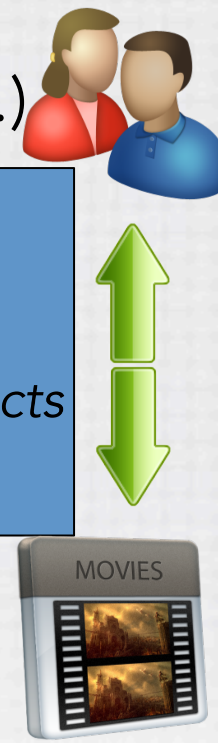
- Myths from industry
  - ❑Amazon.com generates X percent of their sales through the recommendation lists (30 < X < 70)
  - ❑Netflix generates X percent of their sales through the recommendation lists (30 < X < 70)
- There must be some value in it
  - ❑Recommendation of groups, jobs or people on LinkedIn
  - ❑Friend recommendation and ad personalization on Facebook
  - ❑Song recommendation at last.fm
  - ❑News recommendation at Forbes.com (plus 37% CTR)

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, MISQ, 2007]



- Different system designs / paradigms
  - ❑ Based on availability of exploitable data
  - ❑ Implicit and explicit user feedback
  - ❑ Domain characteristics

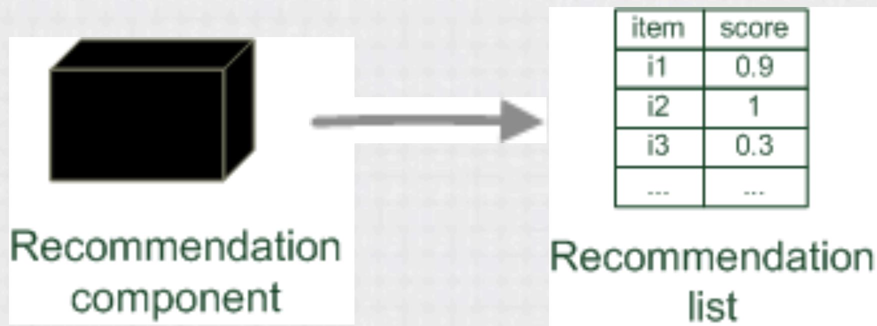
Recommender systems

University of Maryland Robert H. Smith School of Business

- 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 of the list itself might be important (diversity)

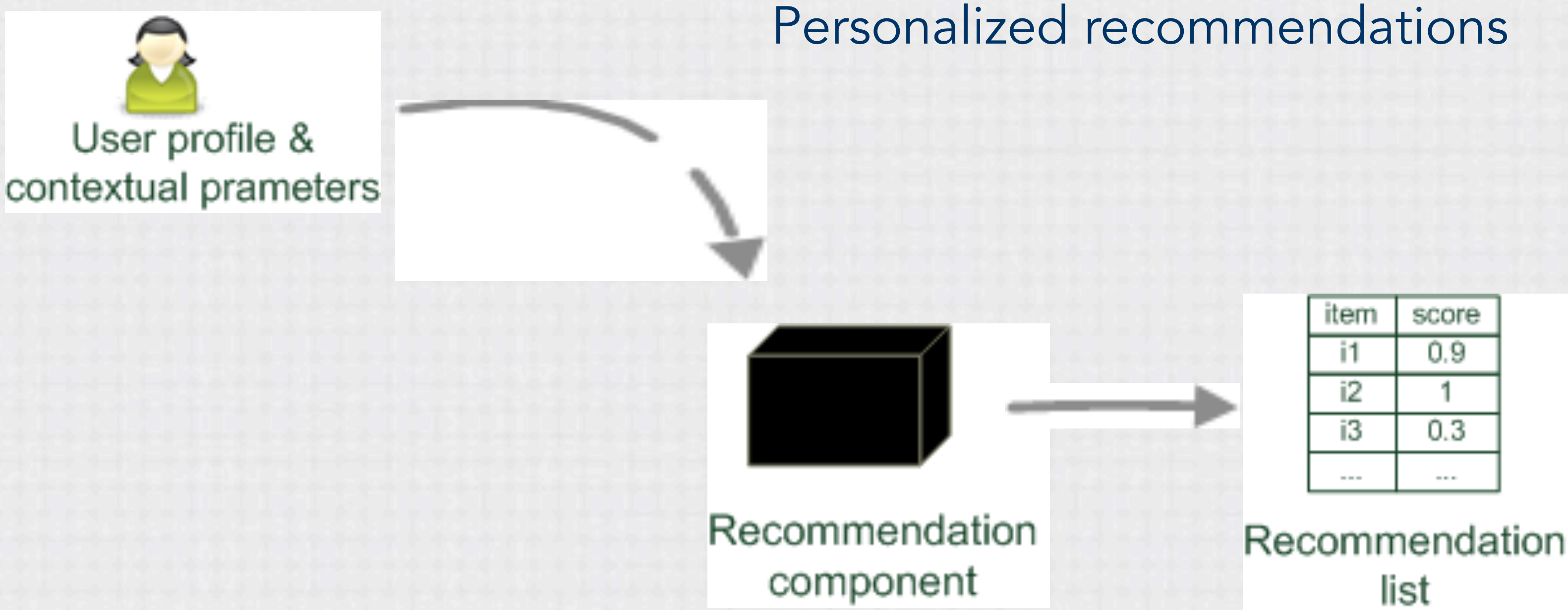
Paradigms of RS

Recommender systems reduce information overload by estimating relevance



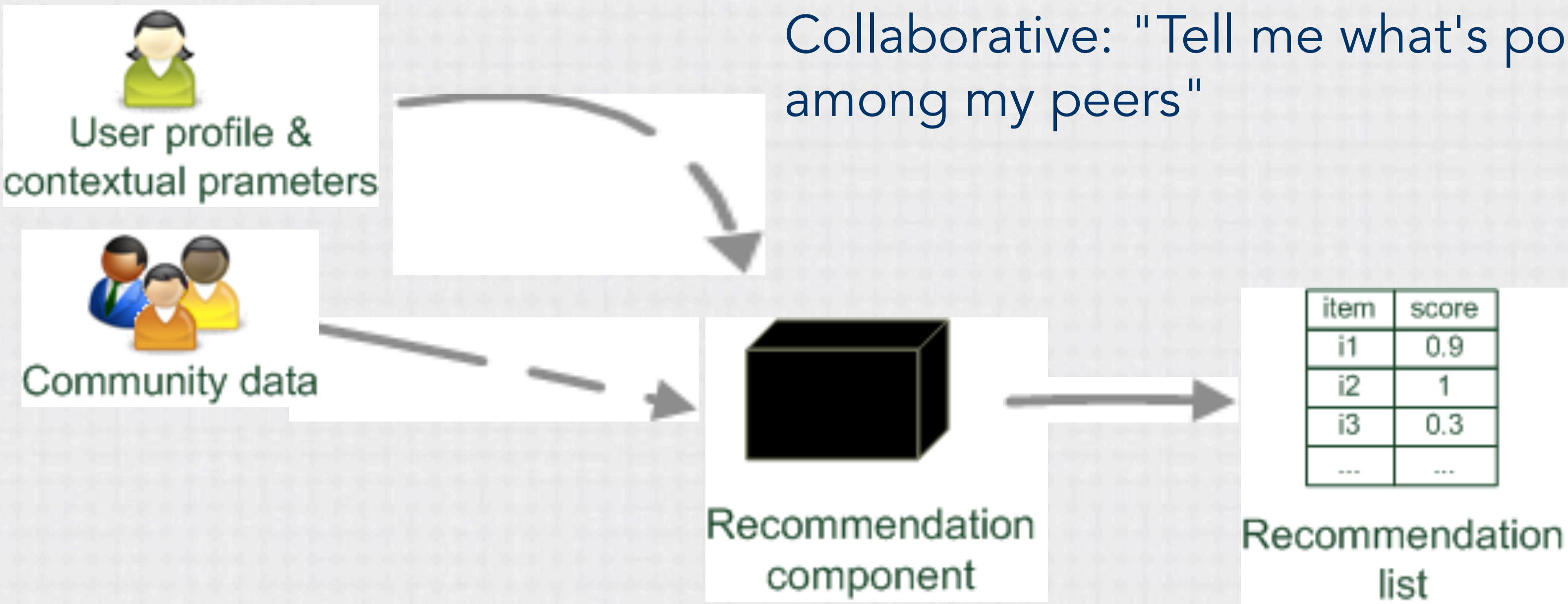
Paradigms of RS

Personalized recommendations



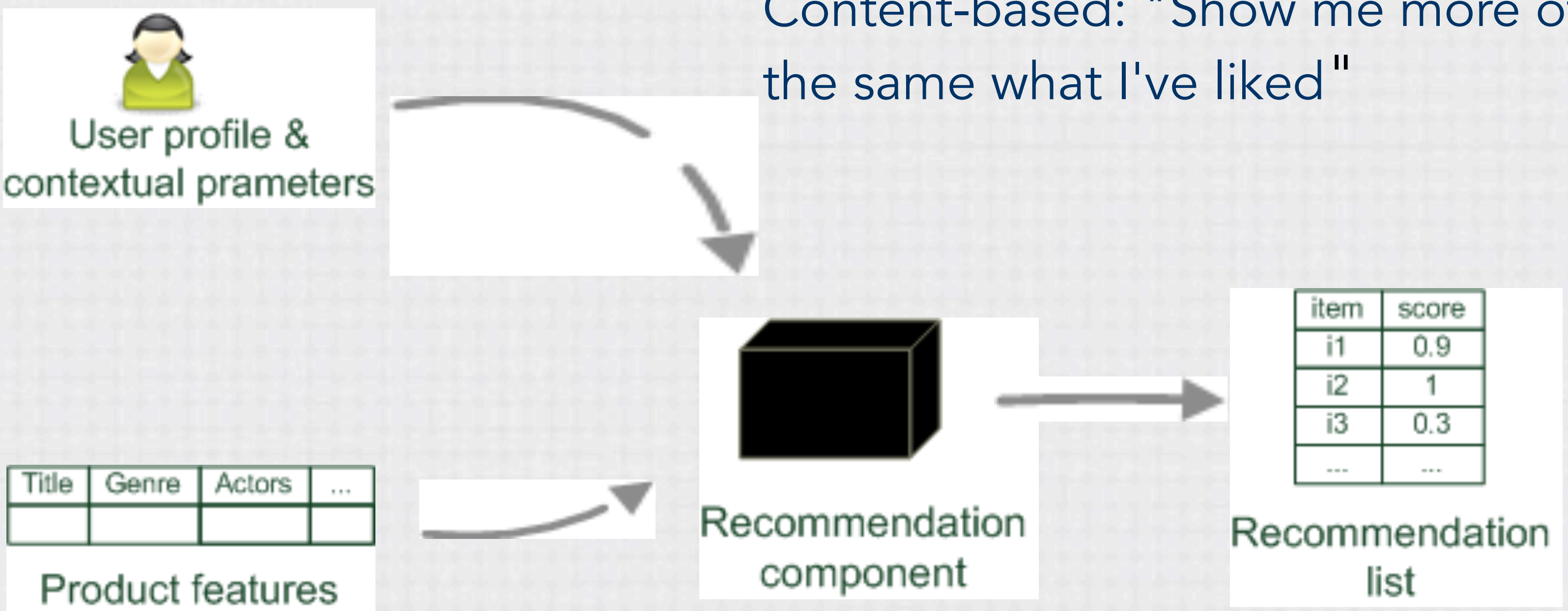
Paradigms of RS

Collaborative: "Tell me what's popular among my peers"



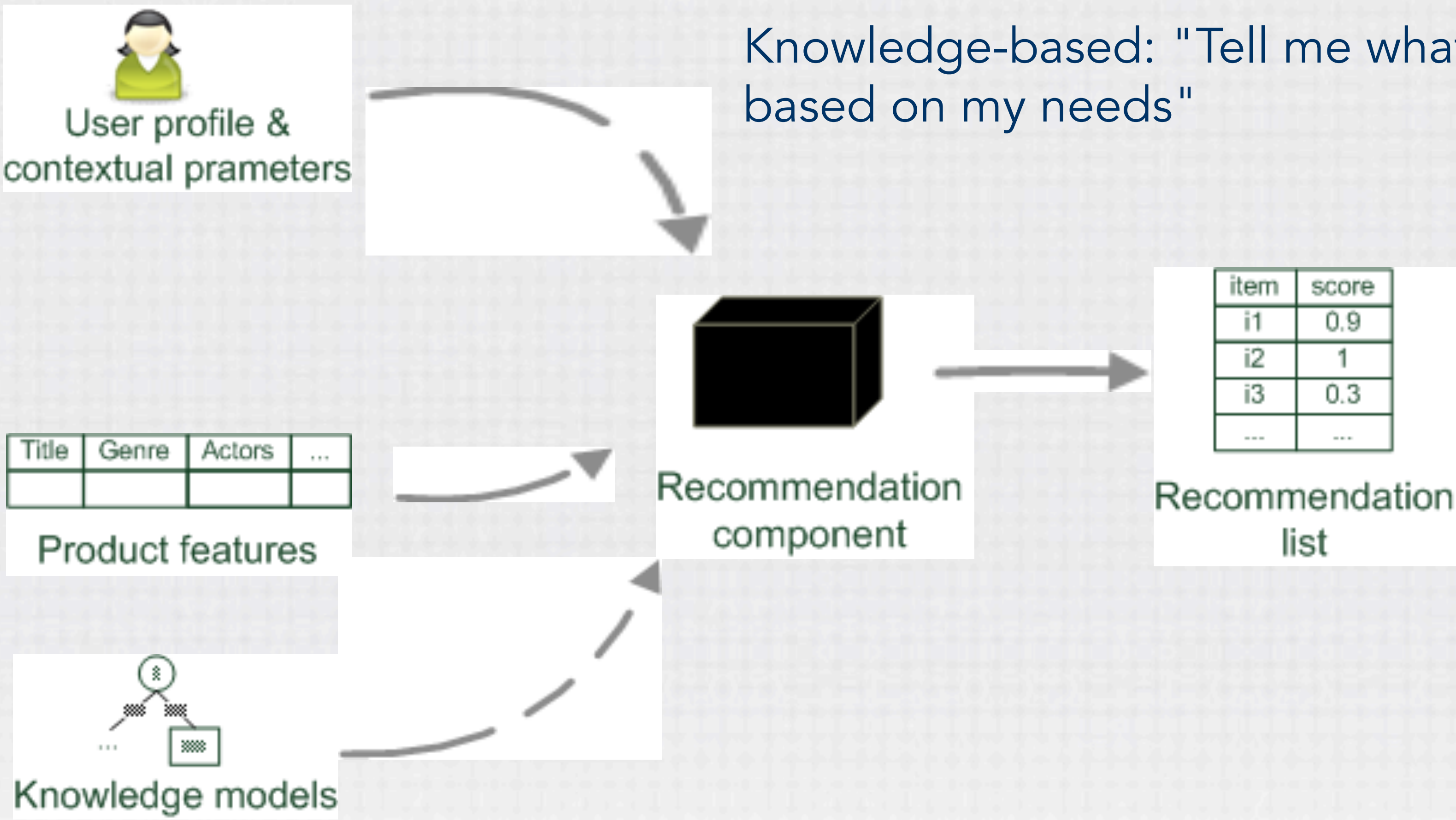
Paradigms of RS

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

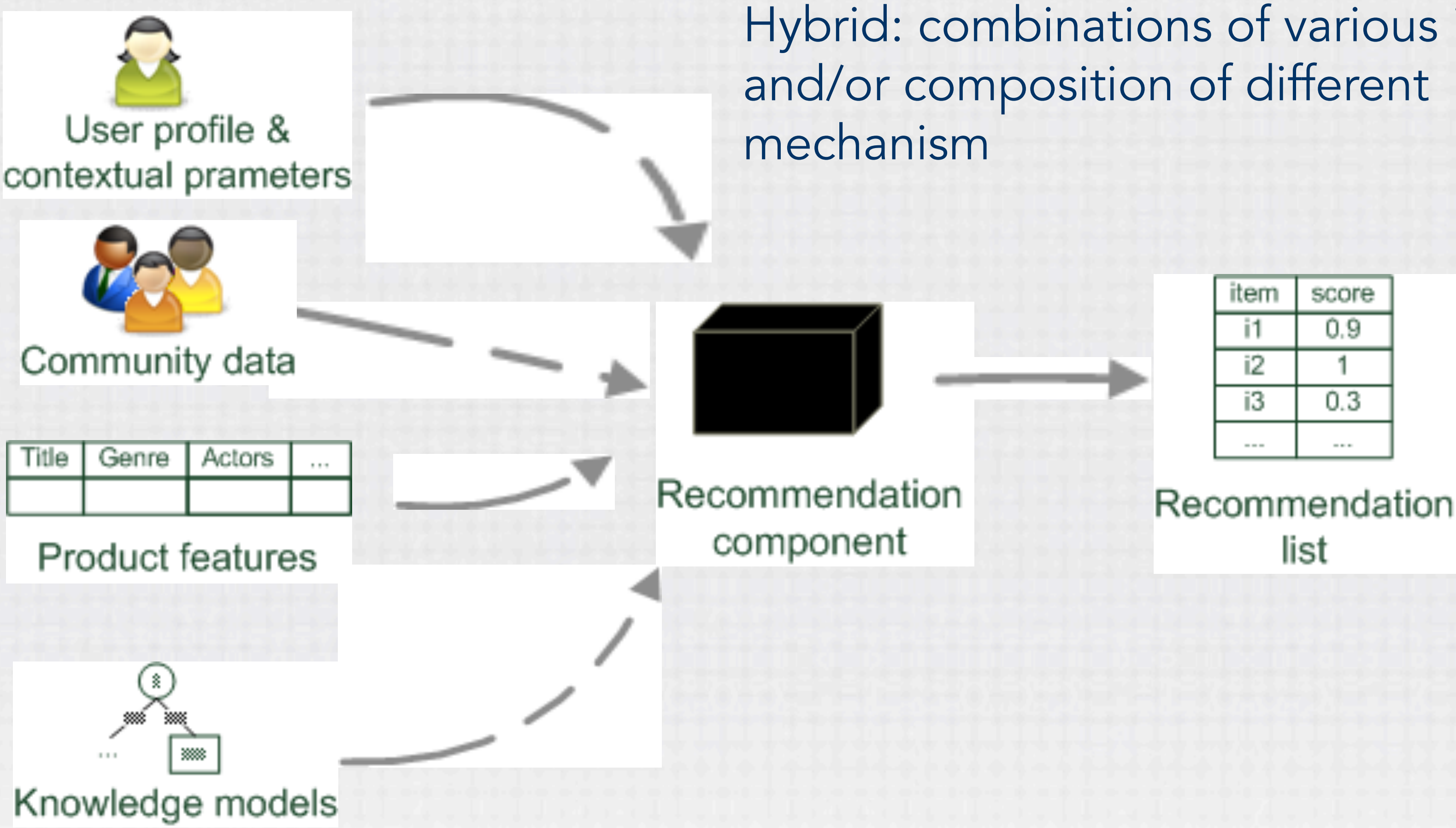


Paradigms of RS

Knowledge-based: "Tell me what fits based on my needs"



Paradigms of RS



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

RS: basic techniques

	Pros	Cons
Collaborative	No knowledge-engineering effort, serendipity of results, learns market segments	Requires some form of rating feedback, cold start for new users and new items
Content-based	No community required, comparison between items possible	Content descriptions necessary, cold start for new users, no surprises
Knowledge-based	Deterministic recommendations, assured quality, no cold-start, can resemble sales dialogue	Knowledge engineering effort to bootstrap, basically static, does not react to short-term trends

Collaborative Filtering (CF)

Collaborative Filtering

- 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

Introduction

- Recommender Systems – Apply knowledge discovery techniques to the problem of making personalized recommendations for information, products or services, usually during a live interaction

- Collaborative Filtering – Builds a database of users' preference for items. Thus, the recommendation can be made based on the neighbors who have similar tastes

Motivation of CF

- Need to develop multiple products that meet the multiple needs of multiple consumers
- Recommender systems used by e-commerce
- Multimedia recommendation

- **Personal tastes** matters

Users, items, preferences

UNIVERSITY OF MARYLAND
ROBERT H. SMITH SCHOOL OF BUSINESS

- Terminology
  - **Users** interact with **items** (books, videos, news, other users,…)
  - **Preferences** of each user towards a small subset of the items known (numeric or boolean)

Basic strategies

UNIVERSITY OF MARYLAND
ROBERT H. SMITH
SCHOOL OF BUSINESS

- Predict and Recommend

- **Predict** the opinion: how likely that the user will have on the this item

- **Recommend** the 'best' items based on
  - the user's previous likings, and
  - the opinions of like-minded users whose ratings are similar

Explicit and implicit ratings

- Where do the preference come from?
- Explicit ratings
  - Users explicitly express their preferences (e.g. ratings with stars)
  - Willingness of the users required
- Implicit ratings
  - Interactions with items are interpreted as expressions of preference (e.g. purchasing a book, reading a news article)
  - Interactions must be detectable

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 CF (1)

- The basic technique:
  - ❑Given an "active user" (Alice) and an item I not yet seen by Alice
  - ❑The *goal is to estimate Alice's rating for this item*, e.g., by
    - find a set of users (peers) who liked the same items
      a
    - u … t, if
      A
    - c
      r

	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

User-based nearest-neighbor CF (2)

UNIVERSITY OF MARYLAND
ROBERT H. SMITH SCHOOL OF BUSINESS

- 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

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

Measuring user similarity (2)

UNIVERSITY OF MARYLAND
ROBERT H. SMITH SCHOOL OF BUSINESS

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

  a, b : users

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

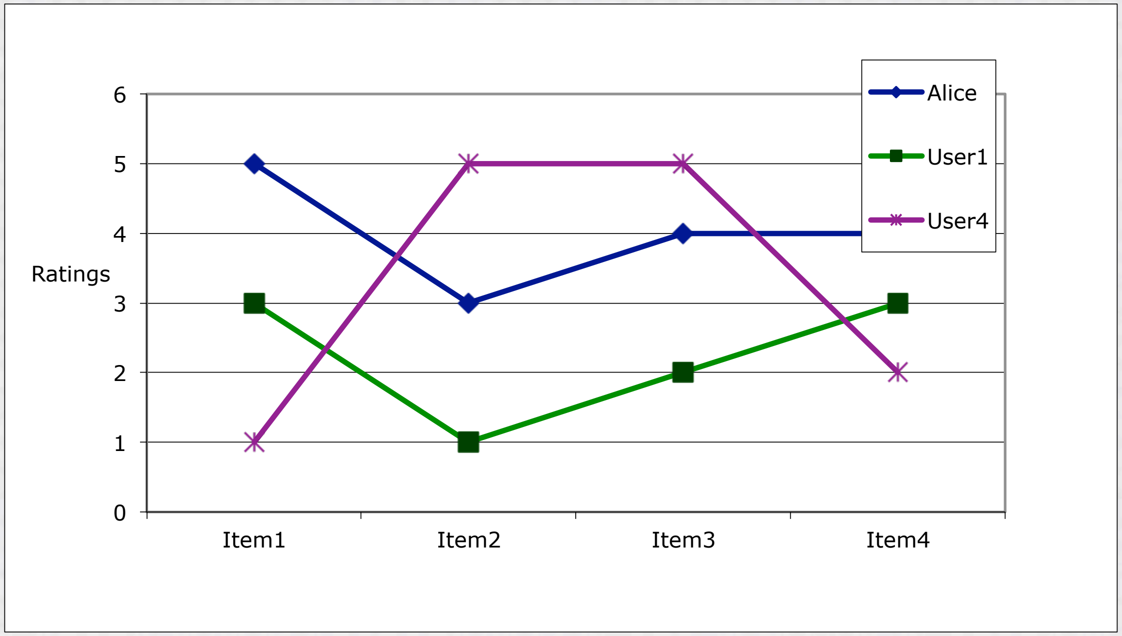
  P : set of items, rated both by a and b

  – Possible similarity values between -1 and 1

	Item1	Item2	Item3	Item4	Item5	
Alice	5	3	4	4	?	
User1	3	1	2	3	3	sim = 0.85
User2	4	3	4	3	5	sim = 0.00
User3	3	3	1	5	4	sim = 0.70
User4	1	5	5	2	1	sim = -0.79

Pearson correlation

- Takes differences in rating behavior into account



- Works well in usual domains, compared with alternative measures
  - such as cosine similarity

Making predictions

- A common prediction function:

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

- 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 metrics prediction function

UNIVERSITY OF MARYLAND
ROBERT H. SMITH SCHOOL OF BUSINESS

- 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

Memory-based and model-based approaches

- User-based CF is said to be "memory-based"
  - the rating matrix is directly used to find neighbors / make predictions
  - does not scale for most real-world scenarios
  - large e-commerce sites have tens of millions of customers and millions of items
- Model-based approaches
  - based on an offline pre-processing or "model-learning" phase
  - at run-time, only the learned model is used to make predictions
  - models are updated / re-trained periodically
  - large variety of techniques used
  - model-building and updating can be computationally expensive
  - *item*-based CF is an example for model-based approaches

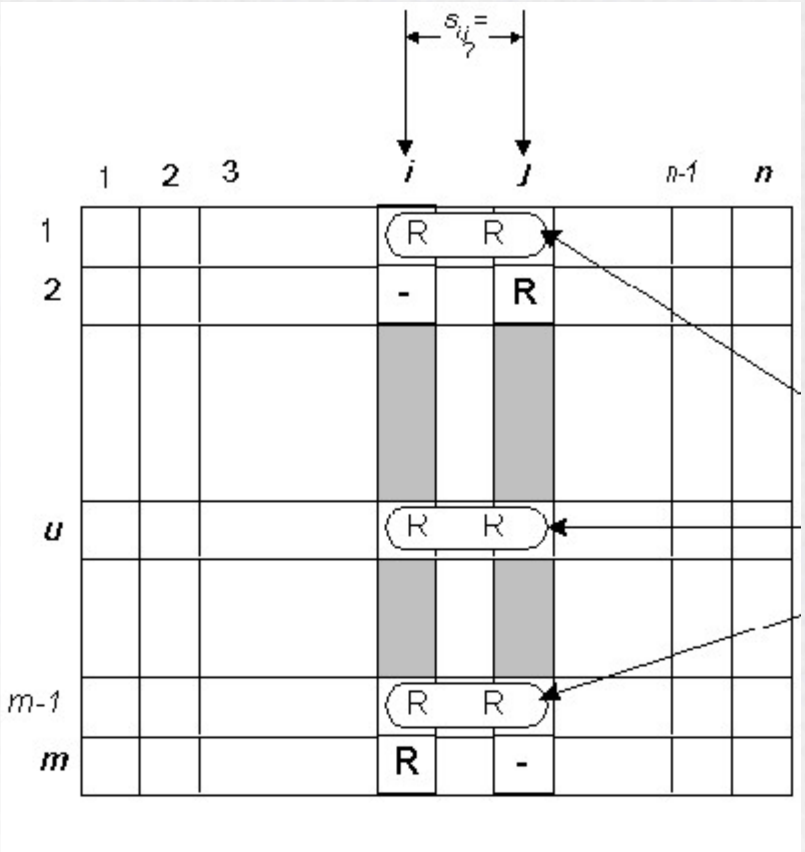
Item-based CF

UNIVERSITY OF MARYLAND
ROBERT H. SMITH SCHOOL OF BUSINESS

- 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

Item-to-Item CF algorithm similarity calculation



Computed by looking into co-rated items only. These co-rated pairs are obtained from different users.

Item-to-Item CF algorithm similarity calculation

UNIVERSITY OF MARYLAND
ROBERT H. SMITH SCHOOL OF BUSINESS

- For similarity between two items i and j,

$$sim(i,j) = \frac{\sum_{u \in U}(R_{u,i} - \bar{R_u})(R_{u,j} - \bar{R_u})}{\sqrt{\sum_{u \in U}(R_{u,i} - \bar{R_u})^2}\sqrt{\sum_{u \in U}(R_{u,j} - \bar{R_u})^2}}.$$

Similarity of „The Matrix" and „Inception"

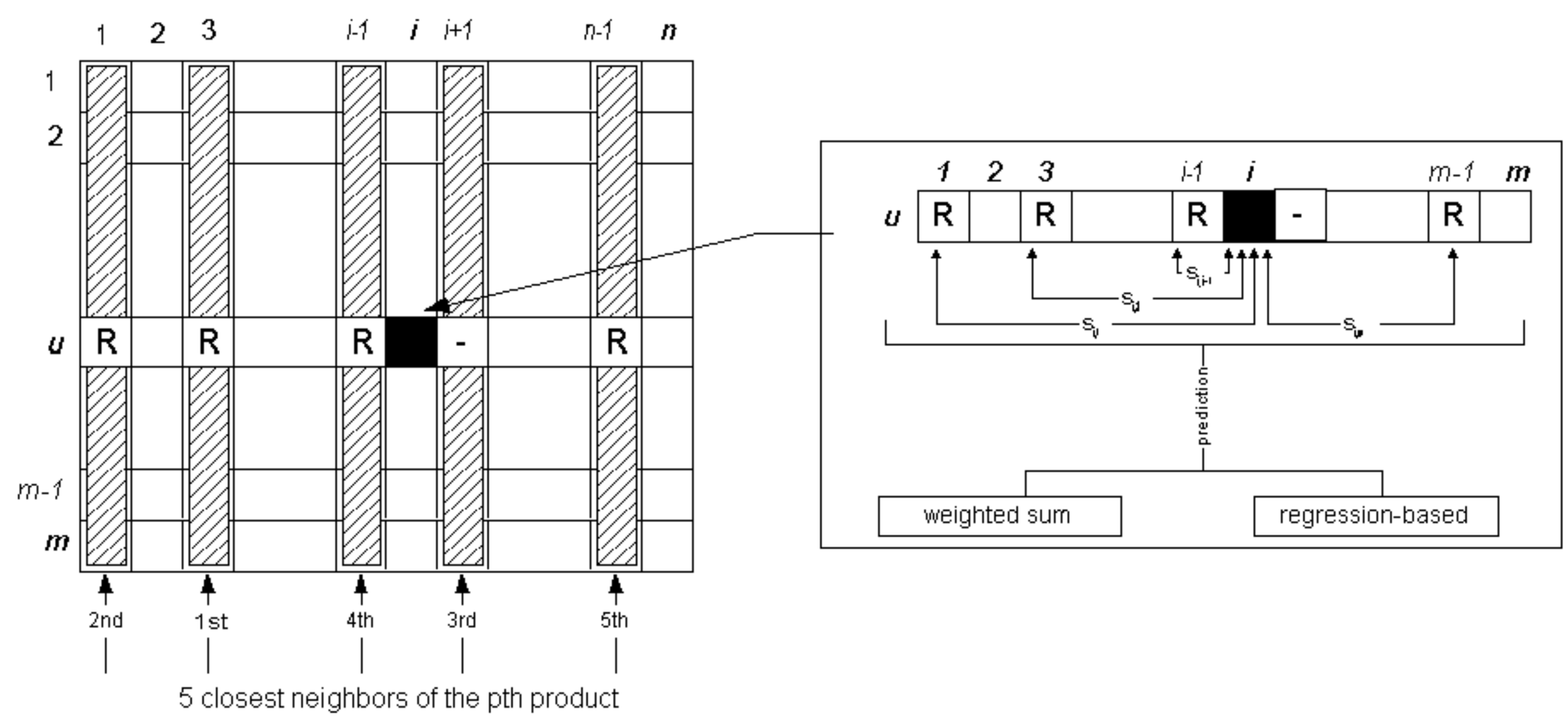
- rating vector of „The Matrix": (5,-,4)
- rating vector of „Inception": (4,5,2)

5	4
-	5
4	2

- isolate all **cooccurred ratings** (all cases where a user rated both items)
- pick a **similarity measure** to compute a similarity value between -1 and 1
  e.g. Pearson-Correlation

$$r = r_{xy} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

Item-to-Item CF algorithm prediction computation



- Recommend items with high-ranking based on similarity

Item-to-Item CF algorithm prediction computation

UNIVERSITY OF MARYLAND
ROBERT H. SMITH SCHOOL OF BUSINESS

- Weighted Sum to capture how the active user rates the similar items

$$P_{u,i} = \frac{\sum_{\text{all similar items, N}} (s_{i,N} * R_{u,N})}{\sum_{\text{all similar items, N}} (|s_{i,N}|)}$$

Rating score of user *u* on item *N*

- The item-item scheme provides better quality of predictions than the user-user scheme

- The **item neighborhood is fairly static**, which can be pre-computed
  - Improve the online performance

Prediction: Estimate Bob's preference towards „The Matrix"

- look at all items that
  - a) are **similar** to „The Matrix"
  - b) have been **rated** by Bob

  => „Alien", „Inception"

- estimate the unknown preference with a weighted sum

$$P_{Bob,\,\mathrm{Matrix}} = \frac{s_{\mathrm{Matrix},\,Alien} * r_{Bob,\,Alien} + s_{\mathrm{Matrix},\,Inception} * r_{Bob,\,Inception}}{\left|s_{\mathrm{Matrix},\,Alien}\right| + \left|s_{\mathrm{Matrix},\,Inception}\right|} = 1.5$$

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 (items, which are rated at least by *n* users)
    - Limit the size of the neighborhood (might affect recommendation accuracy)

More on ratings

- Pure CF-based systems only rely on the rating matrix
- Explicit ratings
  - ❑ Most commonly used (1 to 5)
  - ❑ Research topics
    - "Optimal" granularity of scale; indication that 10-point scale is better accepted in movie domain
    - Multidimensional ratings (multiple ratings per movie)
  - ❑ Challenge
    - Users not always willing to rate many items; sparse rating matrices
    - How to stimulate users to rate more items?
- Implicit ratings
  - ❑ clicks, page views, time spent on some page, demo downloads …
  - ❑ Can be used in addition to explicit ones; question of correctness of interpretation

Data sparsity problems

- 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
- Alternatives
  - ❑Use better algorithms (beyond nearest-neighbor approaches)

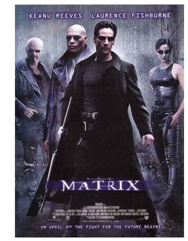
Collaborative filtering issues

- Pros:
  - ❑ well-understood, works well in some domains, no knowledge engineering required
- Cons:
  - ❑ requires user community, sparsity problems, no integration of other knowledge sources, no explanation of results
- What is the best CF method?
  - ❑ In which situation and which domain?
- How to evaluate the prediction quality?
  - ❑ MAE / RMSE: What does an MAE of 0.7 actually mean?
  - ❑ Serendipity: Not yet fully understood
- What about multi-dimensional ratings?

MapReduce-based Item-based CF

Algorithm in Map/Reduce

- How can we compute the similarities efficiently with Map/Reduce?
- Key ideas
  - ❑ We can ignore pairs of items without a co-occurring rating
  - ❑ We need to see all co-occurring ratings for each pair of items in the end
- Inspired by an algorithm designed to compute the pairwise similarity of text documents



5	4
-	5
4	2

Algorithm in Map/Reduce - Pass 1

Map - make user the key

(Alice,Matrix,5)	→	Alice (Matrix,5)
(Alice,Alien,1)	→	Alice (Alien,1)
(Alice,Inception,4)	→	Alice (Inception,4)
(Bob,Alien,2)	→	Bob (Alien,2)
(Bob,Inception,5)	→	Bob (Inception,2)
(Peter,Matrix,4)	→	Peter (Matrix,4)
(Peter,Alien,3)	→	Peter (Alien,3)
(Peter,Inception,2)	→	Peter (Inception,2)

Reduce - create inverted index

Alice (Matrix,5) Alice (Alien,1) Alice (Inception,4)	→	Alice (Matrix,5)(Alien,1)(Inception,4)
Bob (Alien,2) Bob (Inception,5)	→	Bob (Alien,2)(Inception,5)
Peter (Matrix,4) Peter (Alien,3) Peter (Inception,2)	→	Peter (Matrix,4)(Alien,3)(Inception,2)

Algorithm in Map/Reduce - Pass 2

Map - emit all cooccurred ratings

`Alice (Matrix,5)(Alien,1)`
`(Inception,4)`
	→	`Matrix,Alien (5,1)`
		`Matrix,Inception (5,4)`
		`Alien,Inception (1,4)`

`Bob (Alien,2)(Inception,5)`	→	`Alien,Inception (2,5)`

`Peter (Matrix,4)(Alien,3)`
`(Inception,2)`
	→	`Matrix,Alien (4,3)`
		`Matrix,Inception (4,2)`
		`Alien,Inception(3,2)`

Reduce - compute similarities

`Matrix,Alien (5,1)` `Matrix,Alien (4,3)`	→	`Matrix,Alien (-0.47)`
`Matrix,Inception (5,4)` `Matrix,Inception (4,2)`	→	`Matrix,Inception (0.47)`

`Alien,Inception (1,4)`
`Alien,Inception (2,5)`
`Alien,Inception (3,2)`
	→	`Alien,Inception (-0.63)`

Content-based recommendation

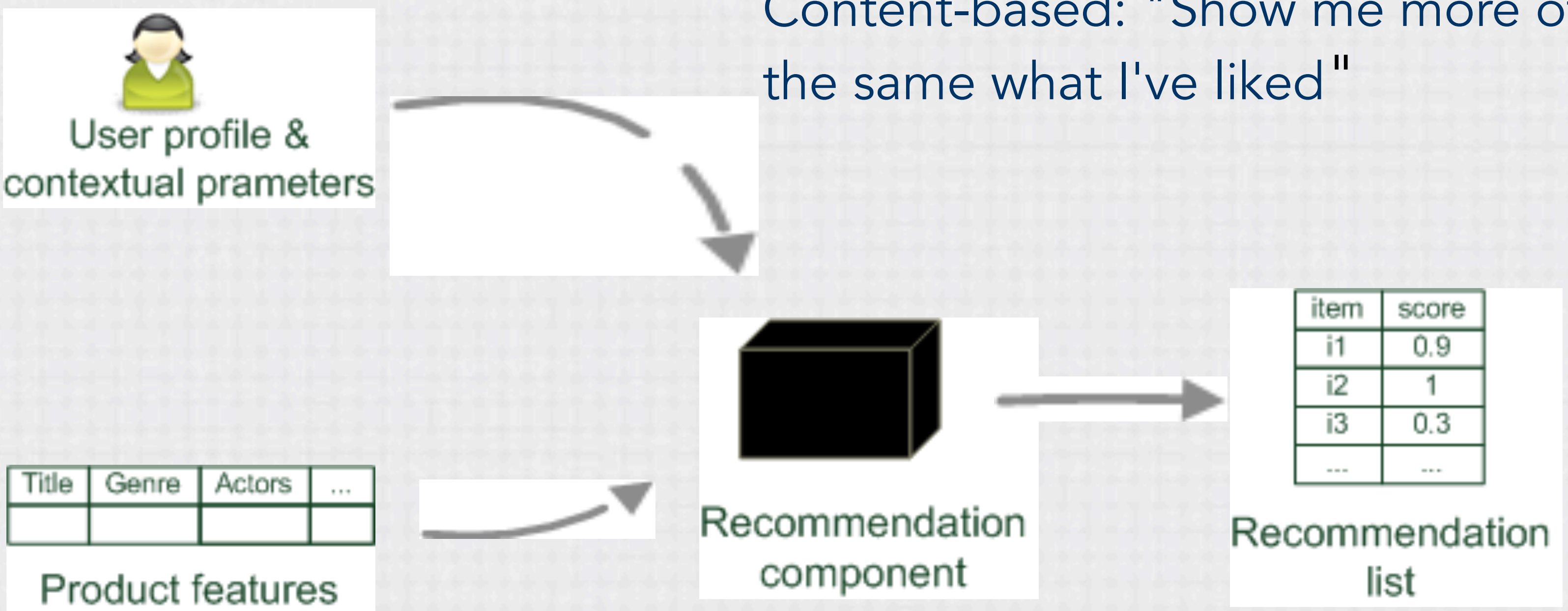
Content-based recommendation

UNIVERSITY OF MARYLAND
ROBERT H. SMITH SCHOOL OF BUSINESS

- Collaborative filtering does NOT require any information about the items,
  - However, it might be reasonable to exploit such information
  - E.g. 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

Paradigms of RS

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



What is the "content"?

- The genre is actually not part of the content of a book
- Most CB-recommendation methods originate from Information Retrieval (IR) field:
  - The item descriptions are usually automatically extracted (important words)
  - Goal is to find and rank interesting text documents (news articles, web pages)

- Here:
  - Classical IR-based methods based on keywords
  - No expert recommendation knowledge involved
  - User profile (preferences) are rather learned than explicitly elicited

Content representation and item similarities

- Simple approach
  - ❑Compute the similarity of an unseen item with the user profile based on the keyword overlap

TF-IDF

- Simple keyword representation has its problems
  - ❑In particular when automatically extracted because
    - Not every word has similar importance
    - Longer documents have a higher chance to have an overlap with the user profile
- Standard measure: TF-IDF
  - ❑Encodes text documents as weighted term vector
  - ❑TF: Measures, how often a term appears (density in a document)
    - Assuming that important terms appear more often
    - Normalization has to be done in order to take document length into account
  - ❑IDF: Aims to reduce the weight of terms that appear in all documents

TF-IDF

UNIVERSITY OF MARYLAND
ROBERT H. SMITH SCHOOL OF BUSINESS

- Compute the overall importance of keywords
  - Given a keyword i and a document j

    *TF-IDF (i,j) = TF(i,j) * IDF(i)*
- Term frequency (TF)
  - Let *freq(i,j)* number of occurrences of keyword *i* in document *j*
  - Let *maxOthers(j)* denote the highest number of occurrences of another keyword in document *j*
    - TF(i, j) = freq(i,j)/maxOthers(j)
- Inverse Document Frequency (IDF)
  - N: number of all recommendable documents
  - n(i): number of documents in which keyword *i* appears
    - IDF = log(N/n(i))

More on the vector space model

- Vectors are usually long and sparse
- Improvements
  - ❑ Remove stop words ("a", "the", ..)
  - ❑ Use stemming
  - ❑ Size cut-offs (only use top n most representative words, e.g. around 100)
  - ❑ Use additional knowledge, use more elaborate methods for feature selection
  - ❑ Detection of phrases as terms (such as United Nations)
- Limitations
  - ❑ Semantic meaning remains unknown
    - ❑ Example: usage of a word in a negative context, e.g., "there is nothing on the menu that a vegetarian would like.."
- Usual similarity metric to compare vectors: cosine similarity

Probabilistic methods

- Recommendation as classical text classification problem
- Simple approach:
  - 2 classes: like/dislike
  - Simple Boolean document representation
  - Calculate probability that document is liked/disliked based on Bayes theorem

Doc-ID	recommender	intelligent	learning	school	**Label**
1	1	1	1	0	**1**
2	0	0	1	1	**0**
3	1	1	0	0	**1**
4	1	0	1	1	**1**
5	0	0	0	1	**0**
6	1	1	0	0	**?**

$$\begin{aligned} P(X|Label{=}1) &= P(recommender{=}1|Label{=}1) \times \\ &\quad P(intelligent{=}1|Label{=}1) \times \\ &\quad P(learning{=}0|Label{=}1) \times P(school{=}0|Label{=}1) \\ &= 3/3 \times 2/3 \times 1/3 \times 2/3 \\ &\approx 0.149 \end{aligned}$$

Remember:

P(Label=1|X) = k*P(X|Label=1) * P(Label=1)

Improvements

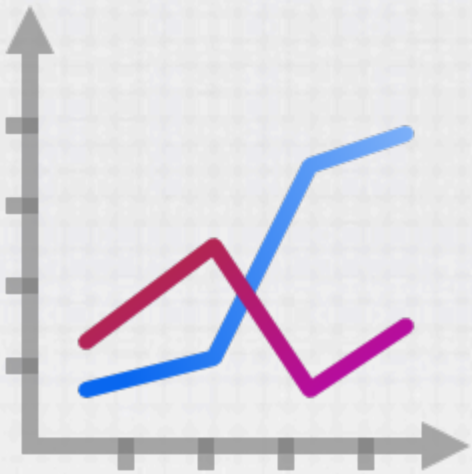
- Side note: Conditional independence of events does in fact not hold
  - “New”/ “York“ and “Hong” / “Kong”
  - Still, good accuracy can be achieved
- Boolean representation simplistic
  - Keyword counts lost
- Other linear classification algorithms (machine learning) can be used
  - Support Vector Machines, ..

Limitations of content-based recommendation methods

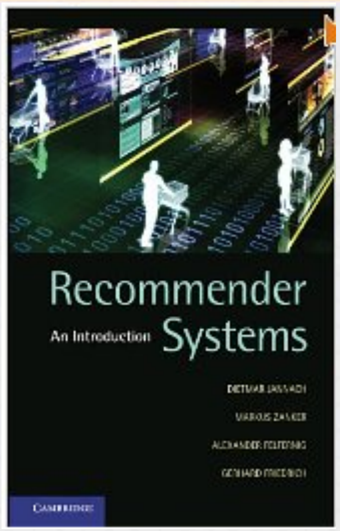
UNIVERSITY OF MARYLAND
ROBERT H. SMITH SCHOOL OF BUSINESS

- Keywords alone may not be sufficient to judge quality/relevance of a document
  - ❑Up-to-dateness, 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

Evaluation of Recommender Systems

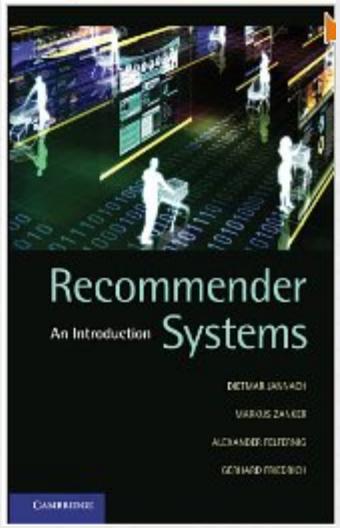


RS in e-commerce



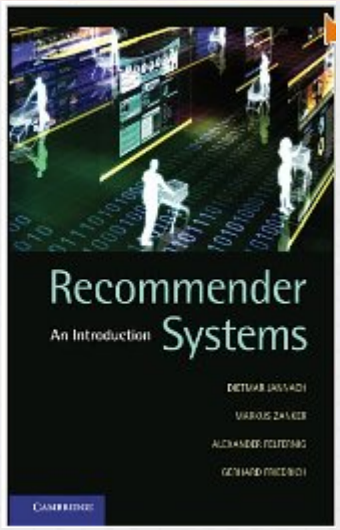
- One RS research question
  - ❑What should be in that list?

RS in e-commerce



- Another question both in research and practice
  - ❑How do we know that these are good recommendations?

RS in e-commerce



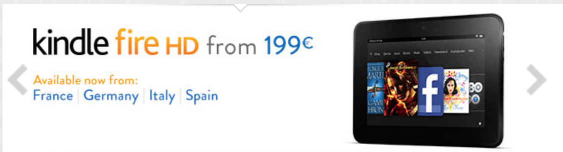
- This might lead to …
  - ❑ What is a good recommendation?
  - ❑ What is a good recommendation strategy?
  - ❑ What is a good recommendation strategy for my business?

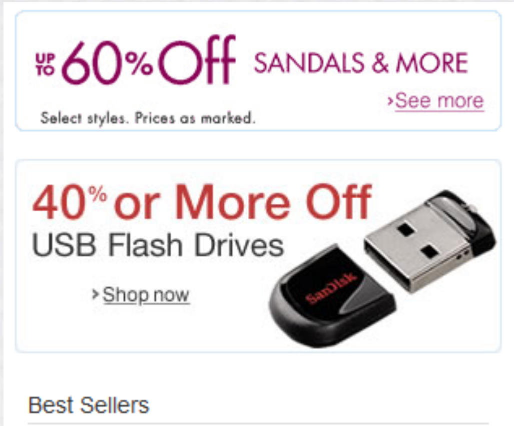
These have been in stock for quite a while now …

What is a good recommendation?

What are the measures in practice?

- Total sales numbers
- Promotion of certain items
- …
- Click-through-rates
- Interactivity on platform
- …
- Customer return rates
- Customer satisfaction and loyalty





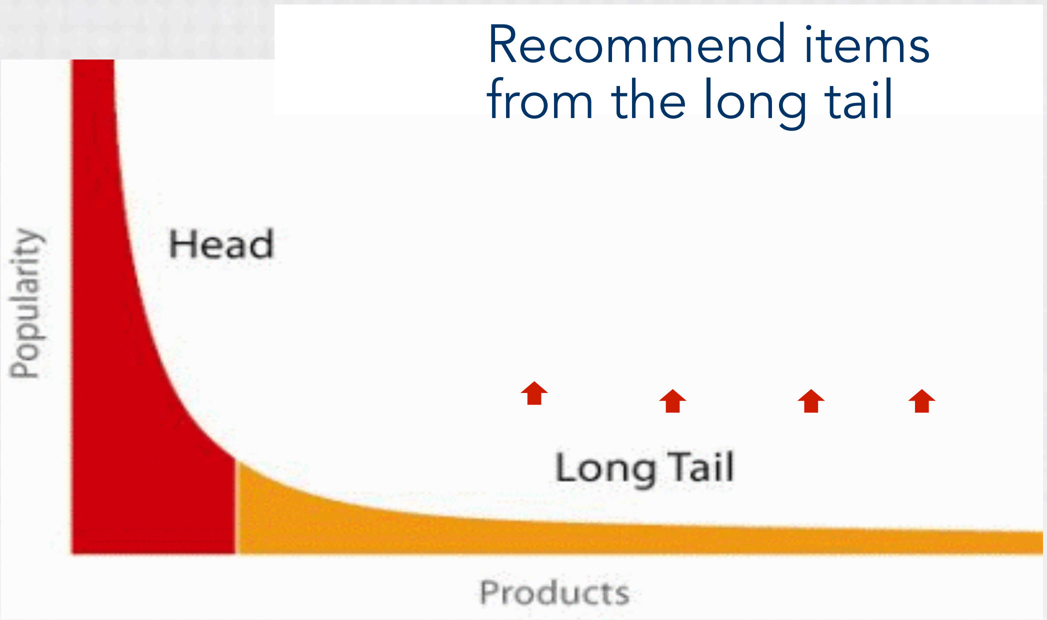
Purpose and success criteria (1)

Different perspectives/aspects

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

- Retrieval perspective
  - ❑ Reduce search costs
  - ❑ Provide "correct" proposals
  - ❑ Assumption: 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

Purpose and success criteria (2)

- Prediction perspective
  - ❑ Predict to what degree users like an item

- 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

How do we know?

- Test with real users
  - ❑A/B tests
  - ❑Example measures: sales increase, click through rates
- Laboratory studies
  - ❑Controlled experiments
  - ❑Example measures: satisfaction with the system (questionnaires)
- Offline experiments
  - ❑Based on historical data
  - ❑Example measures: prediction accuracy, coverage

Evaluation in information retrieval (IR)

- Recommendation is viewed as information retrieval task:
  - ❑ Retrieve (recommend) all items which are predicted to be "good" or "relevant".
- Common protocol :
  - ❑ Hide some items with known ground truth
  - ❑ Rank items or predict ratings -> Count -> Cross-validate
- Ground truth established by human domain experts

		Reality	
		Actually Good	Actually Bad
Prediction	Rated Good	True Positive (tp)	False Positive (fp)
	Rated Bad	False Negative (fn)	True Negative (tn)

Metrics: precision and recall

- Precision: a measure of exactness, determines the fraction of relevant items retrieved out of all items retrieved
  - E.g. the proportion of recommended movies that are actually good

$$Precision = \frac{tp}{tp + fp} = \frac{|good\ movies\ recommended|}{|\text{all recommendations}|}$$

$$Recall = \frac{tp}{tp + fn} = \frac{|good\ movies\ recommended|}{|all\ good\ movies|}$$

- Recall: a measure of completeness, determines the fraction of relevant items retrieved out of all relevant items
  - E.g. the proportion of all good movies recommended

Metrics: Rank score position matters

For a user:

Actually good
Item 237
Item 899

hit

Recommended (predicted as good)
Item 345
Item 237
Item 187

- Rank score extends recall and precision to take the positions of correct items in a ranked list into account
  - ❑ Particularly important in recommender systems as lower ranked items may be overlooked by users

Accuracy measures

UNIVERSITY OF MARYLAND
ROBERT H. SMITH SCHOOL OF BUSINESS

- Datasets with items rated by users
  - ❑ MovieLens datasets 100K-10M ratings
  - ❑ Netflix 100M ratings
- Historic user ratings constitute ground truth
- Metrics measure error rate
  - ❑ Mean Absolute Error (*MAE*) computes the deviation between predicted ratings and actual ratings

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|p_i - r_i|$$

  - ❑ Root Mean Square Error (*RMSE*) is similar to *MAE*, but places more emphasis on larger deviation

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(p_i - r_i)^2}$$