Web Mining

Web Usage Mining and Recommender Systems – Part 1 –

Prof. Dr. Christian Bizer

FSS 2024

Web Usage Mining

Definition

Discovery of patterns in click-streams and associated data collected as a result of user interactions with one or more web sites or applications.

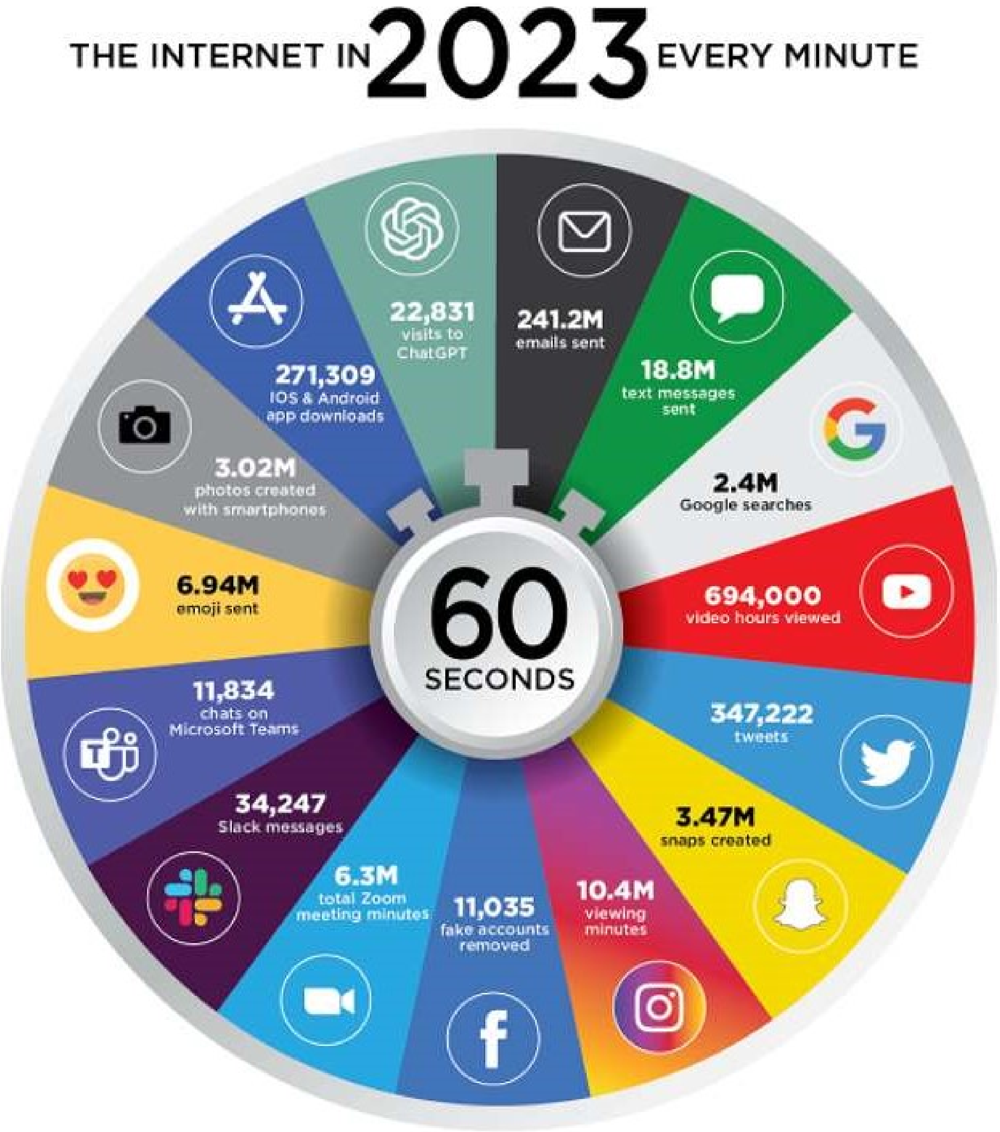
Typical Sources of Data

1. web server access logs
2. e-commerce and product-oriented user events (e.g., shopping cart changes, ad or product click-throughs, purchases)
3. user events on social network sites (e.g., likes, posts, comments)

Associated Data

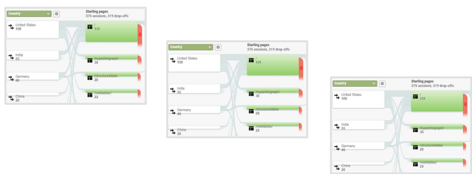
1. page attributes, page content, site structure
2. additional domain knowledge and demographic data
3. user profiles or user ratings

Web Usage Data: The Oil of the New Economy

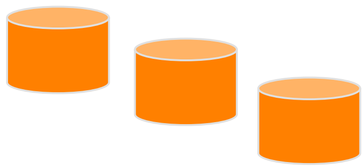


Google Analytics

Get Traffic Analysis



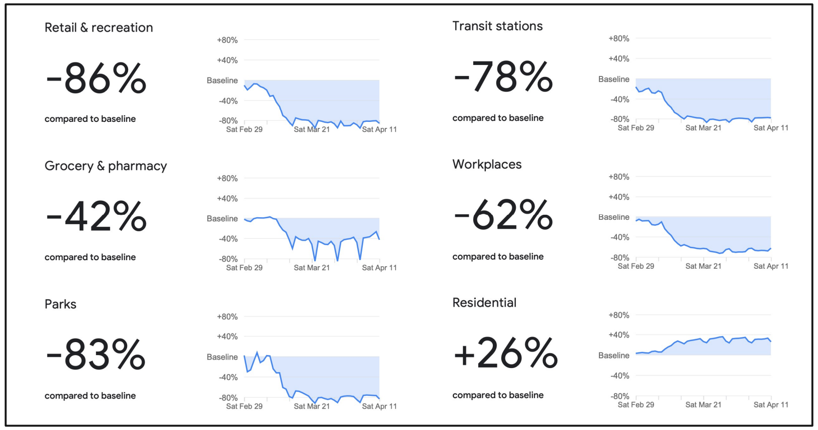
Provide Access to



Economic and Social Impact of Usage Data Collection

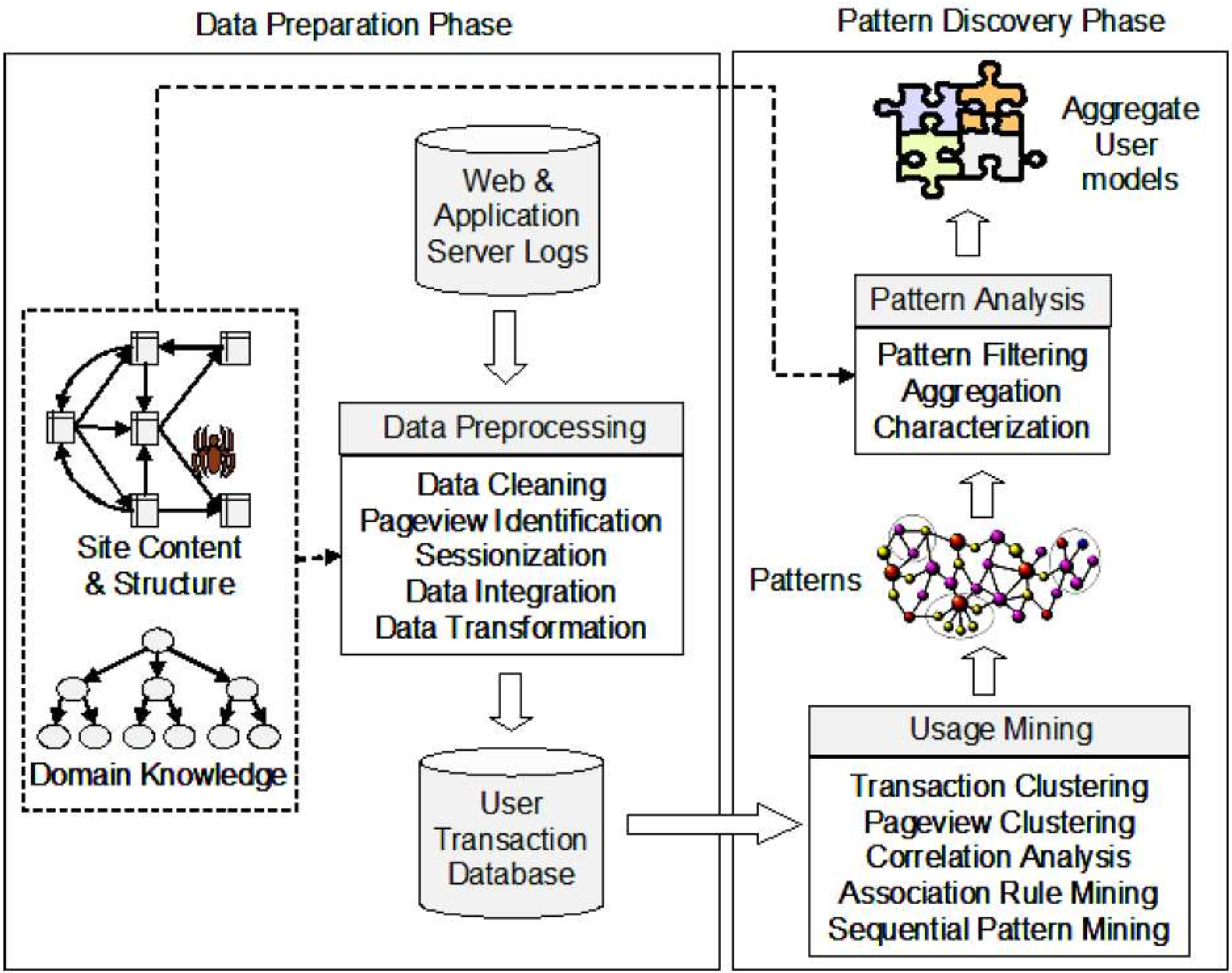
- **Who owns the usage data?**
  - the user? private companies? government?
- **Who is allowed to use it for what?**
  - Companies for targetting users?
  - Government for fighting COVID?
  - Government for law enforcement?
- **Privacy law, and agreement boxes**
- **Alternative: SOLID**
  - decentral data collection and decentral rights tracking
  - difficult to deploy
  - https://solidproject.org/

Google COVID Lockdown Movement Tracking



Social Scoring of „trusthworthiness“

The Web Usage Mining Process



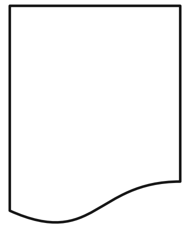
Chapter Outline

1. **Usage Data Collection**
2. **Usage Data Preparation**
   1. User and Session Identification
   2. Data Aggregation and Semantic Enrichment
3. **Usage Mining Tasks**
4. **Recommender Systems**
   1. Collaborative Filtering
   2. Content-based Recommendation
   3. Model-based Recommendation
   4. Hybrid Recommendation
   5. Evaluating Recommender Systems
   6. Attacks on Recommender Systems

1. Usage Data Collection

- **Server-Side Data Collection**
  - Traditional web server logs
    - Content: IP, timestamp, page URL, browser, …
    - Format: text files, database
  - Application Logs
    - Specific application events
      (e.g. change in shopping basket)
  - Restricted to single server

Logfile



- **Client-Side Data Collection**
  - via page tagging
    - often not restricted to single server
  - via providing the application
  - additional collectable data:
    - mouse movements
    - keyboard strokes
    - size of browser window

Page tagging

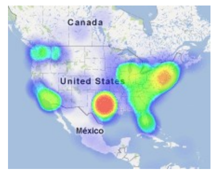
```
<script type="text/javascript">
  var _gaq = _gaq || [];
  _gaq.push(['_setAccount', 'UA-XXXXXX-X']);
  _gaq.push(['_trackPageview']);

  (function() {
    var ga = document.createElement('script'); ga.type
    ga.src = ('https:' == document.location.protocol ?
    var s = document.getElementsByTagName('script')[0]
  })();
</script>
```

Google Ads

Google Analytics

Google Photos



Providing the application

chrome

ANDROID APP ON Google play

Available on the App Store

Recording Users Entering and Leaving the Site

Web server logs may extend beyond visits to the site and show

- **where a visitor was before (via HTTP *Referer*)**

  203.30.5.145 - - [01/Jun/2021:03:09:21 -0600] "GET /Calls/OWOM.html HTTP/1.0" 200 3942 "http://www.lycos.com/cgi-bin/pursuit?query=advertising+psychology-&maxhits=20&cat=dir" "Mozilla[en] (Win10; I)"

- **and where she went next (via URL Rewriting):**

  often used be search engines to get user feedback about search results

2. Data Preparation

Content of a typical Apache web server log:

<ip_addr> - - <date><method><file><protocol><statuscode><bytes><referer><user_agent>

```
203.30.5.145 - - [01/Jun/2021:03:09:21 -0600] "GET /Calls/OWOM.html HTTP/1.0" 200
3942 "http://www.lycos.com/cgi-bin/pursuit?query=advertising+psychology-
&maxhits=20&cat=dir" "Mozilla/4.5 [en] (Win98; I)"

203.30.5.145 - - [01/Jun/2021:03:09:23 -0600] "GET /Calls/Images/earthani.gif
HTTP/1.0" 200 10689 "http://www.acr-news.org/Calls/OWOM.html" "Mozilla/4.5 [en]
(Win98; I)"

203.30.5.145 - - [01/Jun/2021:03:09:24 -0600] "GET /Calls/Images/line.gif
HTTP/1.0" 200 190 "http://www.acr-news.org/Calls/OWOM.html" "Mozilla/4.5 [en]
(Win98; I)"

203.252.234.33 - - [01/Jun/2021:03:12:31 -0600] "GET / HTTP/1.0" 200 4980 ""
"Mozilla/4.06 [en] (Win95; I)"

203.252.234.33 - - [01/Jun/2021:03:12:35 -0600] "GET /Images/line.gif HTTP/1.0"
200 190 "http://www.acr-news.org/" "Mozilla/4.06 [en] (Win95; I)"

203.252.234.33 - - [01/Jun/2021:03:12:35 -0600] "GET /Images/red.gif HTTP/1.0" 200
104 "http://www.acr-news.org/" "Mozilla/4.06 [en] (Win95; I)"

203.252.234.33 - - [01/Jun/2021:03:12:35 -0600] "GET /Images/earthani.gif
HTTP/1.0" 200 10689 "http://www.acr-news.org/" "Mozilla/4.06 [en] (Win95; I)"
```

Data Preparation

1. Data Cleansing

- remove irrelevant log entries and fields from server logs
  - usually: remove all log entries related to images or scripts
  - ignoring certain page-views / items
- remove log entries due to crawler navigation (>50% of all requests)

2. Data Integration

- synchronize data from multiple server logs (due to server farms)
- integrate semantics, e.g. meta-data (e.g., content labels), e-commerce and application server data, registration data

3. Data Transformation

- user identification
- session identification
- data aggregation / semantic enrichement

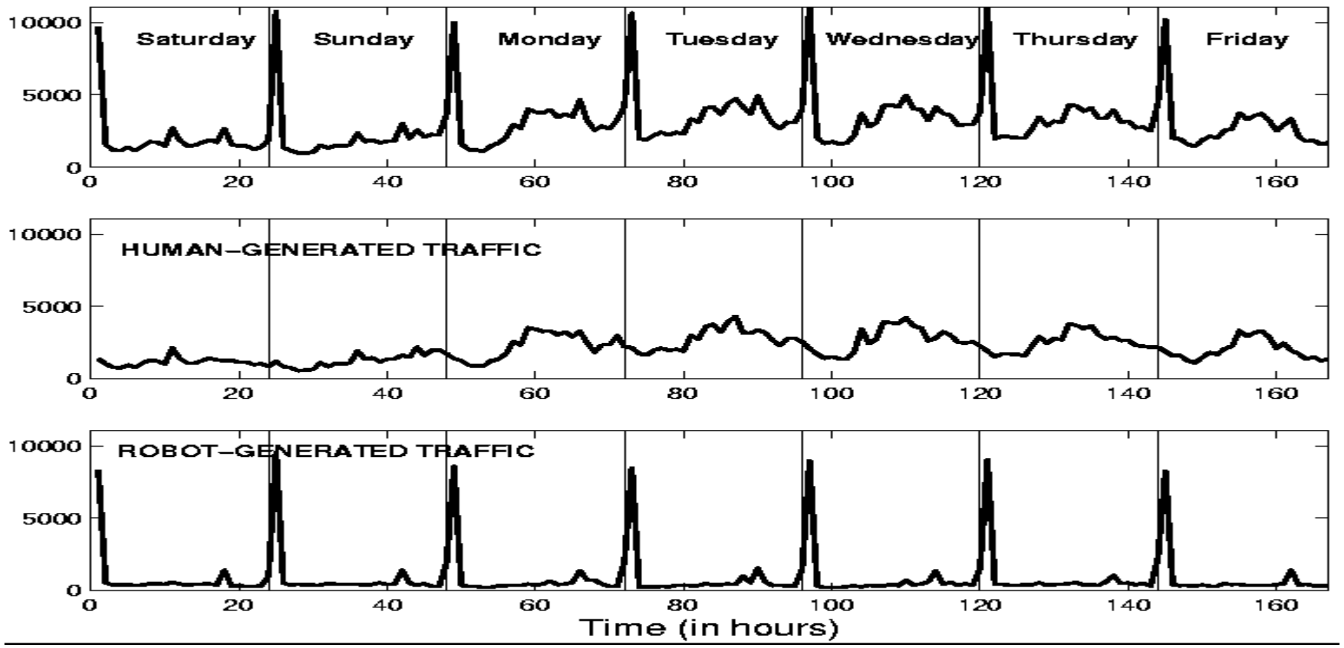
4. Data Reduction

- sampling

Robot Detection

1. **Identification via HTTP User-Agent Header**
   - using list of known robots, e.g. from http://useragentstring.com/
2. **Classification using Behavioural Features**
   - accesses robots.txt file
   - time on page
   - navigation patters
   - no download of images or scripts

Example of Web Crawler Traffic



Tan, Kumar: Discovery of Web Robot Sessions based on their Navigational Patterns. Data Mining and Knowledge Discovery 6(1),2002.

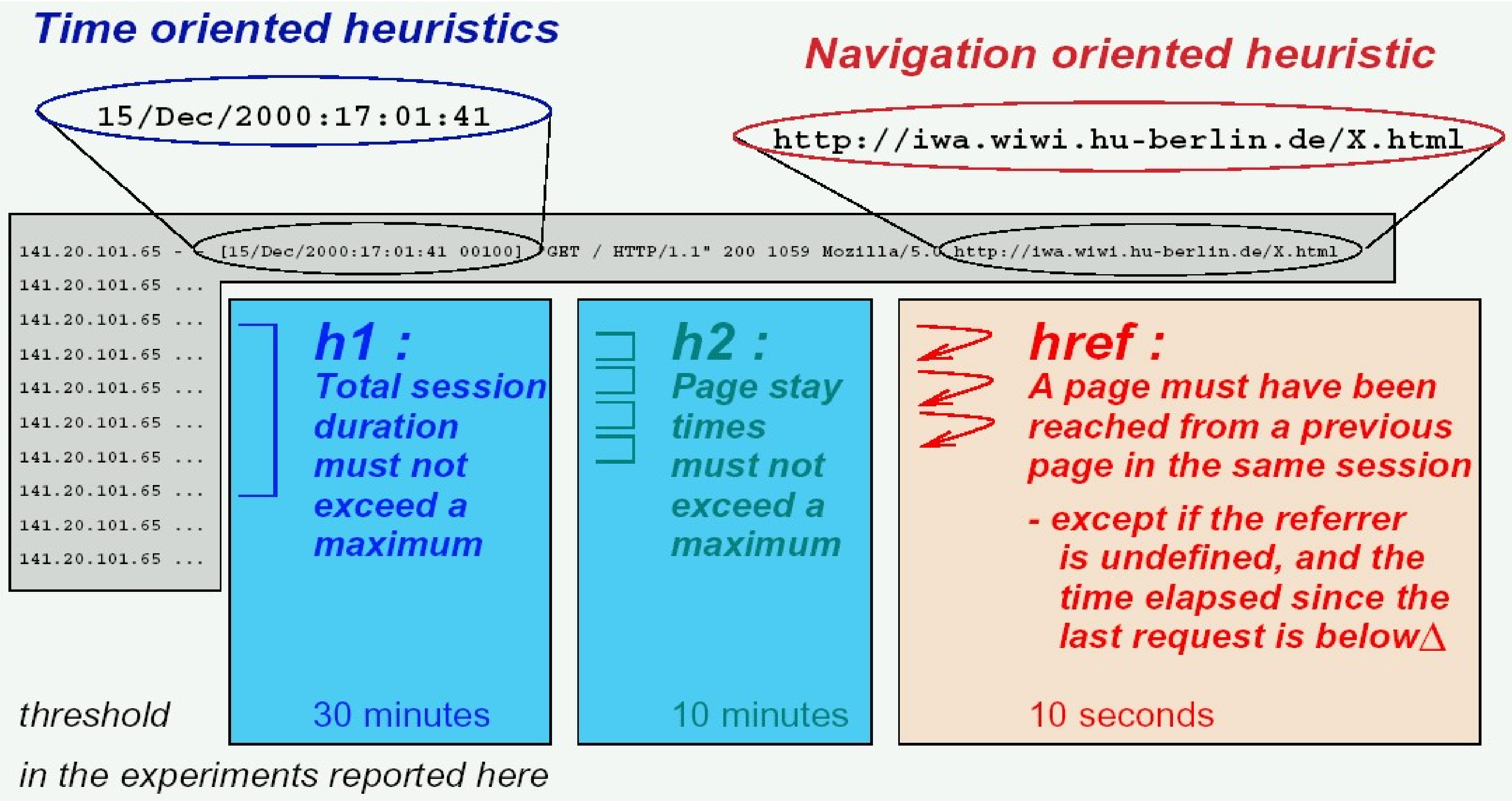
Mechanisms for User Identification

Method	Description	Privacy Concerns	Advantages	Disadvantages
IP Address + Agent	Assume each unique IP address/Agent pair is a unique user	Low	Always available. No additional technology required.	Not guaranteed to be unique. Defeated by rotating IPs.
Embedded Session Ids	Use dynamically generated pages to associate ID with every hyperlink	Low to medium	Always available. Independent of IP addresses.	Cannot capture repeat visitors. Additional overhead for dynamic pages.
Registration	User explicitly logs in to the site.	Medium	Can track individuals not just browsers	Many users won't register. Not available before registration.
Cookie	Save ID on the client machine.	Medium to high	Can track repeat visits from same browser.	Can be turned off by users.
Software Agents	Program loaded into browser and sends back usage data.	High	Accurate usage data for a single site.	Likely to be rejected by users.

Examples of agents: apps, browsers, page tags (use javascript)

Not anymore.

Mechanisms for Session Identification



Source: Spiliopoulou et al., 2003

Data Aggregation

- **aggregate log data in order to generate features that are suitable for the task at hand (identify robots, cluster users, …)**
- **Examples of possible Features**

Attribute Name	Description
totalPages	Total number of pages retrieved in a Web session
ImagePages	Total number of image pages retrieved in a Web session
TotalTime	Total amount of time spent by Web site visitor
RepeatedAccess	The same page requested more than once in a Web session
ErrorRequest	Errors in requesting for Web pages
GET	Percentage of requests made using GET method
POST	Percentage of requests made using POST method
HEAD	Percentage of requests made using HEAD method
Breadth	Breadth of Web traversal
Depth	Depth of Web traversal
MultiIP	Session with multiple IP addresses
MultiAgent	Session with multiple user agents

Data Aggregation

- **Example of a User Pageview Matrix**

Pageviews

Sessions / users

	A	B	C	D	E	F
user0	15	5	0	0	0	185
user1	0	0	32	4	0	0
user2	12	0	0	56	236	0
user3	9	47	0	0	0	134
user4	0	0	23	15	0	0
user5	17	0	0	157	69	0
user6	24	89	0	0	0	354
user7	0	0	78	27	0	0
user8	7	0	45	20	127	0
user9	0	38	57	0	0	15

- **Useful for discovering user groups (cluster analysis)**

Semantic Enrichment

- **Basic Idea**

> **Associate each requested page with one or more topics/ concepts to better understand user behavior.**

- **The request for a page signals interest in the concept(s).**
- **Aggregation Levels:**
  - Page level: 1 request ➔ 1 concept or n concepts
    for example: insurances, travel, …
  - Session level: set / sequence of pages ➔ 1 concept or n concepts
    for example: user compares insurance offers
- **Concepts can be part of a concept hierarchy or ontology:**
  - Useful for building/maintaining user profiles

Google Knowledge Graph

DBpedia Categories

Example: Semantic Enrichment

- **Input: User Pageview Matrix**

	A.html	B.html	C.html	D.html	E.html
user1	1	0	1	0	1
user2	1	1	0	0	1
user3	0	1	1	1	0
user4	1	0	1	1	1
user5	1	1	0	0	1
user6	1	0	1	1	1

- **Input: Page Topic Matrix**

	A.html	B.html	C.html	D.html	E.html
web	0	0	1	1	1
data	0	1	1	1	0
mining	0	1	1	1	0
business	1	1	0	0	0
intelligence	1	1	0	0	1
marketing	1	1	0	0	1
ecommerce	0	1	1	0	0
search	1	0	1	0	0
information	1	0	1	1	1
retrieval	1	0	1	1	1

- **Result : User Topic Matrix**

	web	data	mining	business	intelligence	marketing	ecommerce	search	information	retrieval
user1	2	1	1	1	2	2	1	2	3	3
user2	1	1	1	2	3	3	1	1	2	2
user3	2	3	3	1	1	1	2	1	2	2
user4	3	2	2	1	2	2	1	2	4	4
user5	1	1	1	2	3	3	1	1	2	2
user6	3	2	2	1	2	2	1	2	4	4

Interests that Google Stores about Me

Interests

Bicycles & Accessories, and 14 more Edit
From your previous activity on Google

Baden-Wuerttemberg, and 27 more
Based on the websites you've visited

We use interests from your activity on websites to tailor ads to you. How it works. Please note the listed categories do not include some of the ways ads may be tailored to you, including remarketing lists based on visits to advertiser websites.

Interest
Baden-Wuerttemberg
Business News
Canary Islands
Celebrities & Entertainment News
Colleges & Universities
Computers & Electronics
Coupons & Discount Offers
Dictionaries & Encyclopedias
Education
Email & Messaging
Finance
Financial Markets
General Reference

Greece
Health Insurance
Inheritance & Estate Planning
Insurance
Investing
Parenting
Performing Arts
Politics
Primary & Secondary Schooling (K-12)
Search Engines
Soccer
Social Networks
Travel
Water Sports
Weather

Google Ads

- **https://adssettings.google.com/**

3. Web Usage Mining Tasks

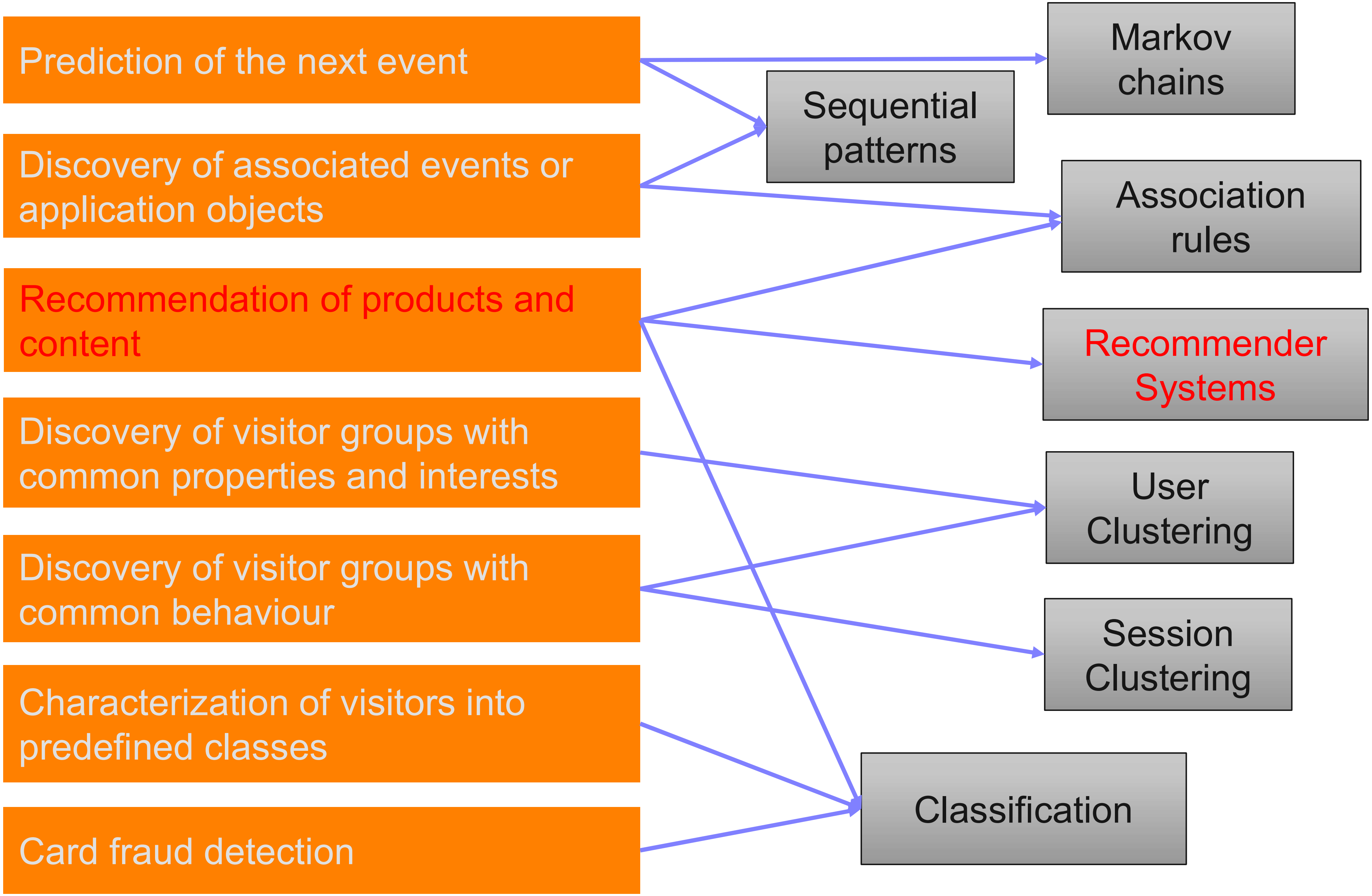
1. Content Personalization

- Personalized content and navigation elements
- Techniques: Classification, Re-Ranking, Sequential Pattern Mining, Recommender Systems

2. Marketing

- Discovery of associated products for cross-selling
  - Association rules, Sequential Pattern Mining
  - Placement of associated products on the same page
- Discovery of associated products in different price categories for up-selling
  - Association rules, Sequential Pattern Mining
- Identification of Customer Groups for Targeted Marketing
  - Clustering, Classification
- Personalized recommendations
  - Suggestions of similar items (e.g. pages or products)
  - Suggestions of items based on the preferences of similar users

Overview: Usage Mining Tasks and Techniques

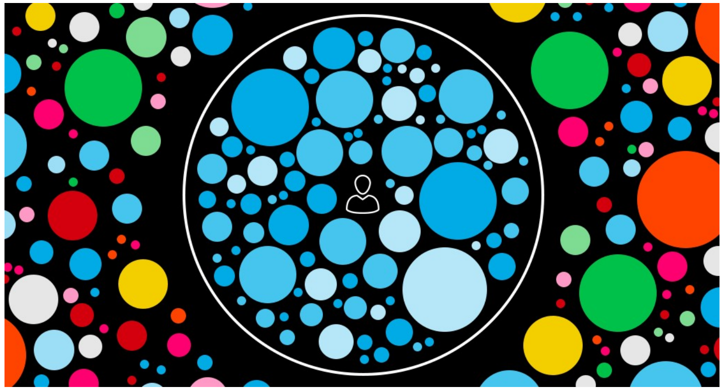


2. Recommender Systems

- **Recommender Systems (RS) help to match users with items**
  - ease information overload (songs on Spotify)
  - sales assistance (advisory versus persuasion)
- **Recommender Systems can be seen as a function**
  - Given:
    - User model (e.g. ratings, preferences, demographics, situational context)
    - Items (with or without description of item characteristics)
  - Predict:
    - Rating/Relevance score. Used for determining the top-k items
- **Concrete system design depends on**
  - the availability of exploitable data
  - domain characteristics

Application Domains of Recommender Systems

- Which music will I like?
- Which movie should I watch?
- Which news fit to my interests?
  my political position? (Filter bubbles)

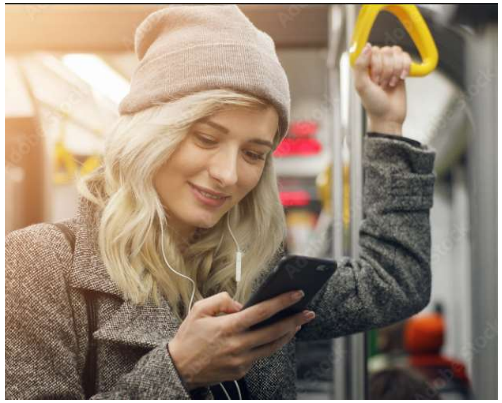


Telegram

When does a Recommender do a good Job?

1. User's Perspective

- Recommend me items that I like **and** did not know about
- **Serendipity:** Accident of finding something good while not specifically searching for it

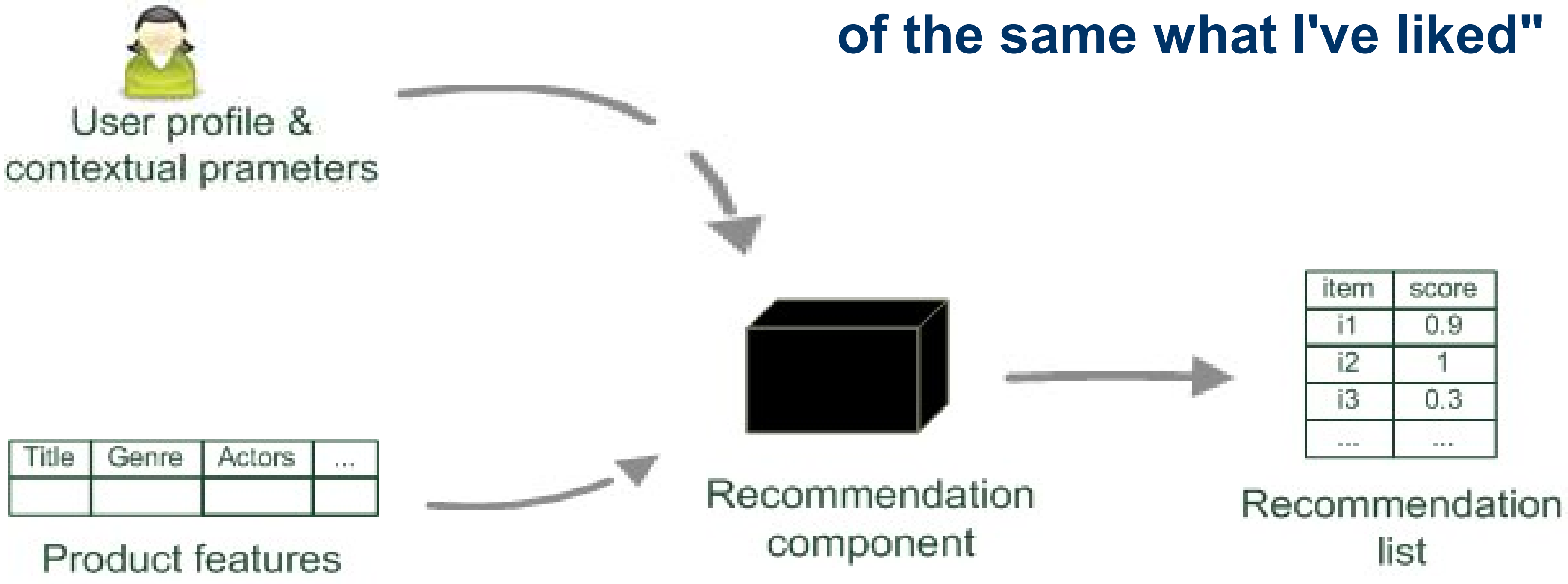


2. Merchant's Perspective

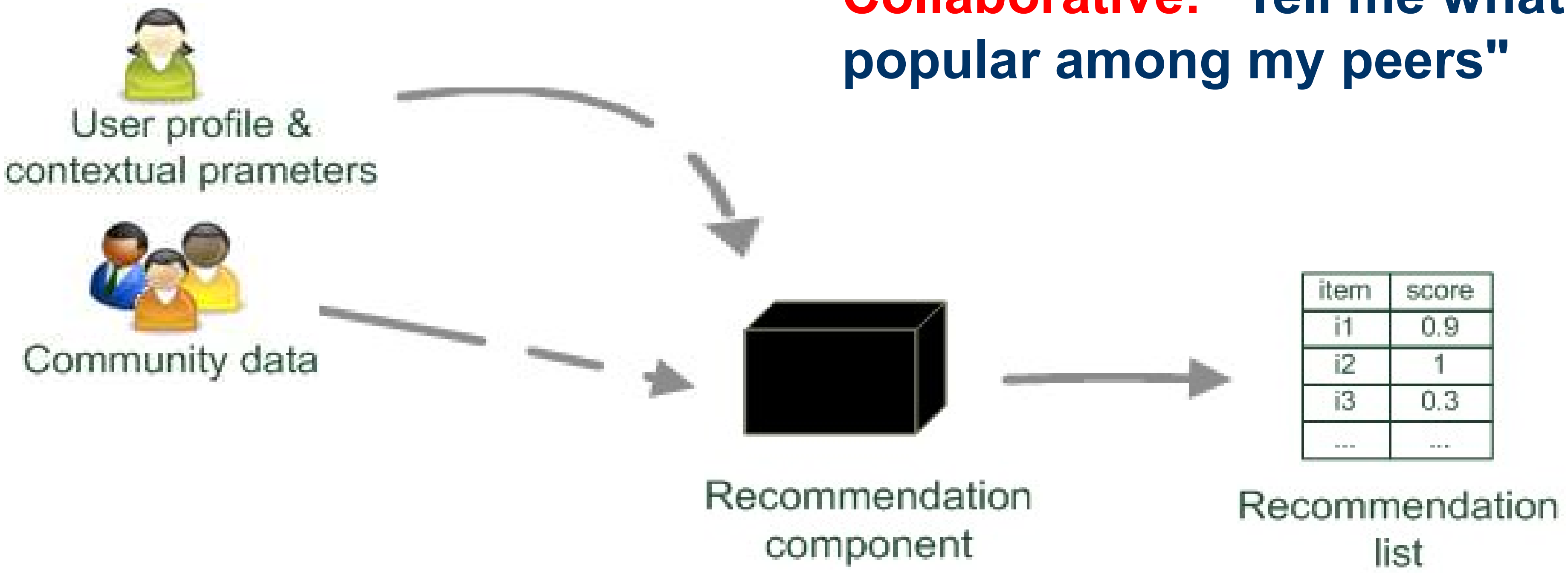
- increase the sale of high-revenue items
- thus, real-world recommender systems are not as neutral as the following slides suggest

Paradigms of Recommender Systems

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



Paradigms of Recommender Systems

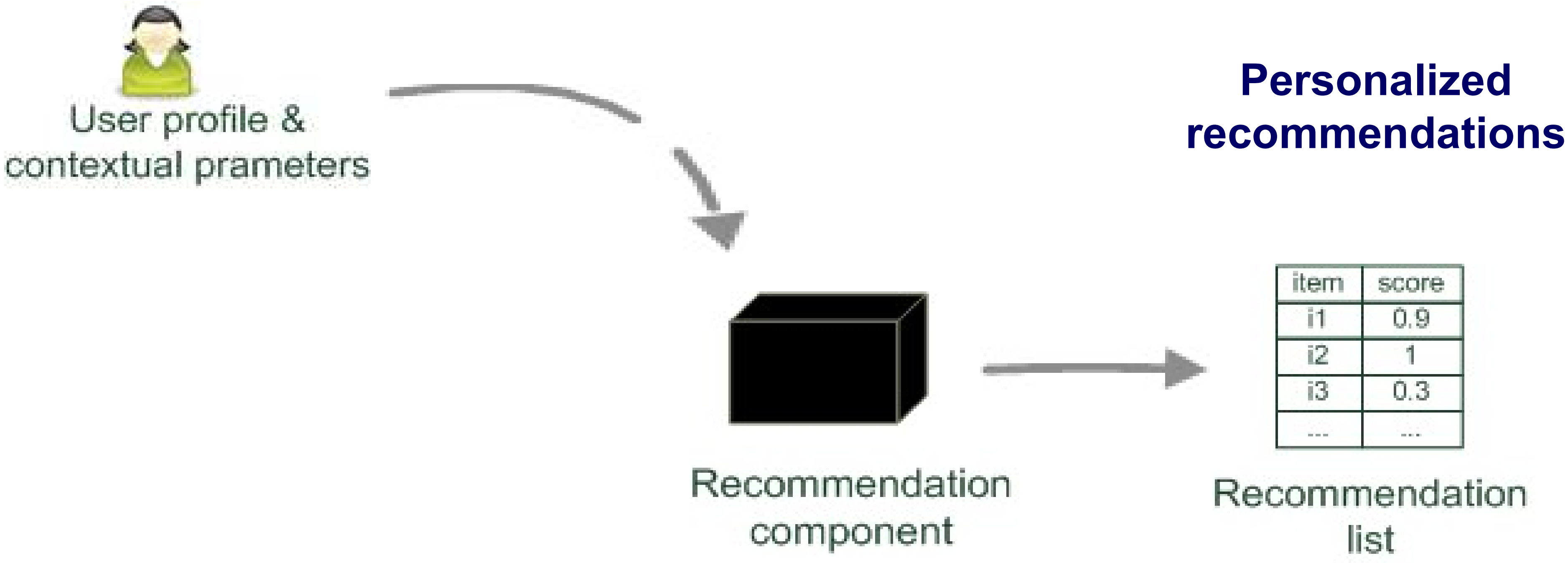


Collaborative: "Tell me what's popular among my peers"

User–Item Rating Matrix

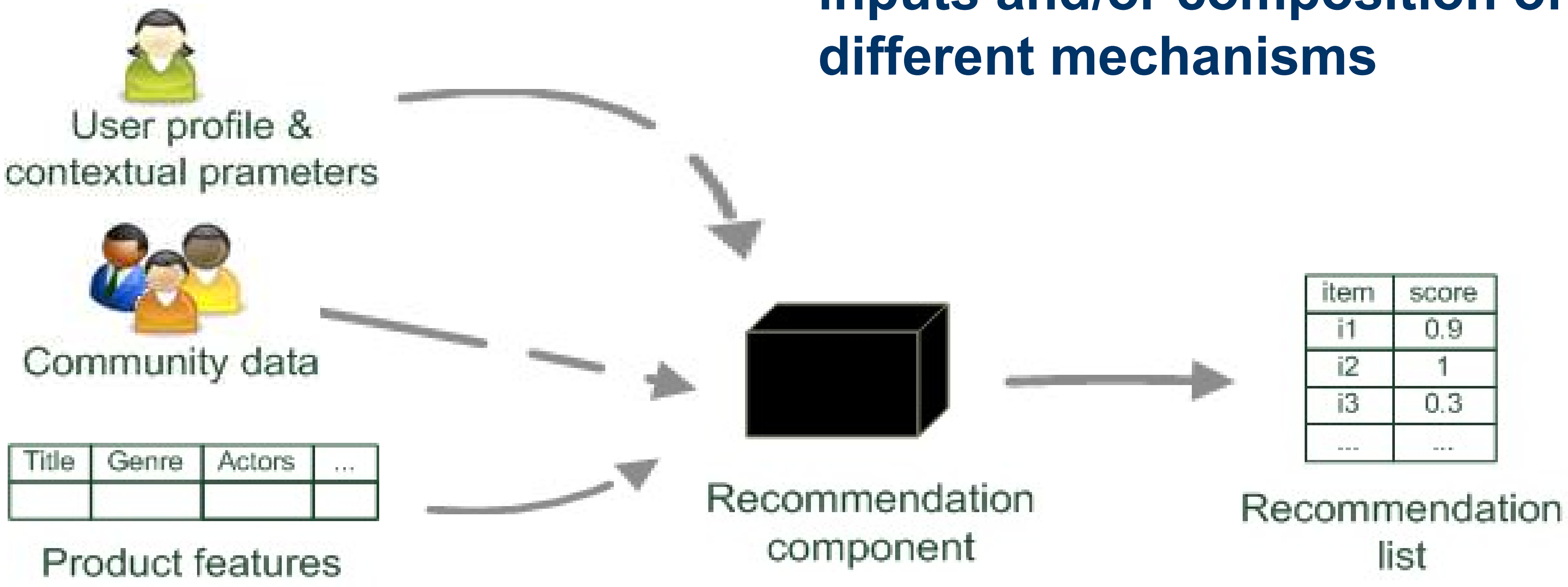
	Item1	Item2
Alice	5	?
User1	2	1
User2	4	3

Paradigms of Recommender Systems



- Demographic Recommendation
  - offer cameras with American electricity plug to people from US
  - offer Backstreet Boys albums to people under the age of 16
- Contextual Recommendation (Location / Time of Day/Year)
  - show holiday related advertisements based on user location
  - send coupon to mobile user who passes by a shop

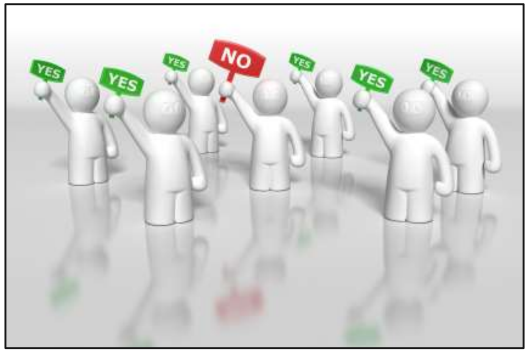
Paradigms of Recommender Systems



Hybrid: Combinations of various inputs and/or composition of different mechanisms

2.1 Collaborative Filtering

- **A standard approach to generate recommendations**
  - used by large e-commerce sites



- **Basic Assumptions**
  1. users give ratings to catalog items
     (implicitly or explicitly)
  2. customers who had similar tastes in the past,
     will have similar tastes in the future

- **Input: Matrix of given user–item ratings**

	Item1	Item2	Item3
Alice	5	3	4
User1	3	1	2
User2	4	3	4
User3	3	3	1

- **Output types**
  1. (Numerical) prediction indicating to what degree
     the current user will like a certain item
     (i.e., a rating itself)
  2. Ranking: Top-k list of recommended items

User-Based Nearest-Neighbor Collaborative Filtering

Given an "active user" (Alice) and an item i not yet rated by Alice

1. find a set of users (peers/nearest neighbors) who liked the same items as Alice in the past **and** who have rated item i
2. use their ratings of item i to predict, if Alice will like item i
3. do this for all items Alice has not seen and recommend the top-rated k items

Example: User–Item Rating Matrix

	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

See: Data Mining I: KNN Regression

User-Based Nearest-Neighbor Collaborative Filtering

Questions we need to answer

1. How do we measure user similarity?
   - given that real-world user/item matrices are very sparse (>90% missing values)
2. How many neighbors should we consider?
   - hyperparameter k in KNN regression
3. How do we generate a prediction from the neighbors' ratings?
   - given that different people use the rating scale differently

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

Measuring User Similarity

- A popular similarity measure in user-based CF is the **Pearson Correlation Coefficient**

  a, b : users

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

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

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

- Takes different usage of rating scale into account by comparing individual ratings to the user's average rating
- For Pearson we need paired data, that is, we take only the ratings for the set of items that are rated by both users (also to compute the average ratings)

Example: Measuring User Similarity

- A popular similarity measure in user-based CF is the **Pearson Correlation Coefficient**

  a, b : users

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

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

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

	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 = 0.70
User3	3	3	1	5	4	sim = 0.00
User4	1	5	5	2	1	sim = -0.79

Making Predictions

1. A simple prediction function:

$$pred(a,p) = \frac{\sum_{b \in N} sim(a,b) * r_{b,p}}{\sum_{b \in N} sim(a,b)}$$

- uses the similarity with a as a weight to combine ratings
- N is the number of similar users that should be considered (hyperparameter k)

2. A prediction function that takes rating behavior into account:

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

- calculates whether the neighbors' ratings for the unseen item i are higher or lower than their average
- uses the similarity with a as a weight to combine rating differences
- add/subtract the neighbors' bias from the active user's average and use this as a prediction

Example: Making Predictions

- To make a prediction for Item5, we first decide which of the neighbours' ratings we take into account and apply the second formula from the previous slide
- In our our example, an obvious choice would be to take User1 and User2 as peer users to predict Alice's rating
- Hence the prediction for Alice's rating for Item5 based on the ratings of nearest neighbours User1 and User2 will be

$$\mathrm{pred}(\mathrm{Alice}, \mathrm{Item5}) = 4 + (\,(\,0.85 * (3 - 2.4) + 0.70 * (5 - 3.8)\,)\,/\,(\,0.85 + 0.70\,)\,) = 4.87$$

	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 = 0.70

sim = 0.00

sim = -0.79

Improving the Similarity / Prediction Functions

1. **Neighborhood selection**
   - use similarity threshold instead of fixed number of neighbors
2. **Case amplification**
   - intuition: Give more weight to "very similar" neighbors, i.e., where the similarity value is close to 1.
   - implementation: $\boldsymbol{sim(a,b)^2}$
3. **Rating variance**
   - 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
4. **Number of co-rated Items**
   - Use "significance weighting", by e.g., linearly reducing the weight when the number of co-rated items is low

Memory-based and Model-based Approaches

- **User-based CF is said to be "memory-based"**
  - The rating matrix is directly used to find neighbors and make predictions
  - To predict we compute user similarity online and collect the ratings of the most similar ones. Such a KNN approach is called lazy learning.
  - This does not scale for large e-commerce sites, which have millions of customers

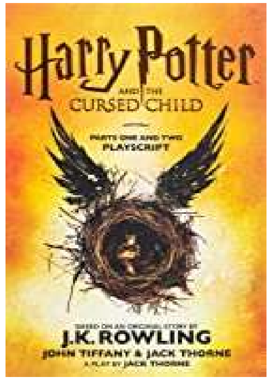
- **Model-based approaches**
  - We build a model offline
  - We use the model we computed offline to make predictions online
  - models are updated / re-trained periodically
  - Examples
    1. **Item-based collaborative filtering**
    2. Probabilistic methods
    3. Matrix factorization

now

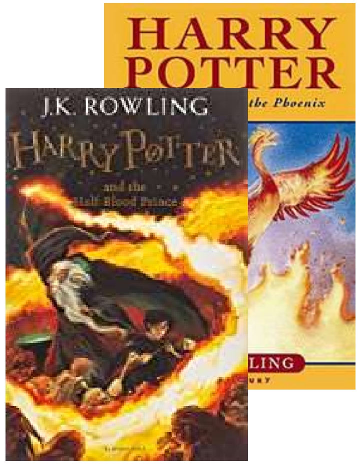
Next week

Item-based Collaborative Filtering

- **Basic idea:**
  - Use the similarity between items (and not users) to make predictions
- **Approach:**
  1. Look for items that have been rated similarly as Item5
  2. 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



Calculating Item-to-Item Similarity

- **Cosine Similarity**
  - similarity metric to find similar items which focuses on non-zero rating pairs

$$sim(\vec{a},\vec{b}) = \frac{\vec{a} \cdot \vec{b}}{|\vec{a}| * |\vec{b}|}$$

$$\vec{a} \cdot \vec{b} = \sum_{i=1}^{n} a_i b_i = a_1 b_1 + a_2 b_2 + \cdots + a_n b_n$$

$$|\vec{a}| = \sqrt{{a_1}^2 + {a_2}^2 + {a_3}^2}$$

  - cosine similarity does not take the differences in the average rating behaviour of different users into account

- **Adjusted Cosine Similarity**
  - adjusts ratings by taking the average rating behavior of a user into account
  - U: set of users who have rated both items a and b

$$sim(\vec{a},\vec{b}) = \frac{\sum_{u \in U}(r_{u,a} - \overline{r_u})(r_{u,b} - \overline{r_u})}{\sqrt{\sum_{u \in U}(r_{u,a} - \overline{r_u})^2}\sqrt{\sum_{u \in U}(r_{u,b} - \overline{r_u})^2}}$$

Making Predictions

- **A common prediction function for item-based CF: Weight ratings by item similarity**

$$pred(u,p) = \frac{\sum_{i \in ratedItem(u)} sim(i,p) * r_{u,i}}{\sum_{i \in ratedItem(u)} sim(i,p)}$$

	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

$ratedItem(u)$: Set of items rated by Alice

r_{ui} : Alice‘s rating for items i

$\mathbf{sim}(\mathbf{i},\mathbf{p})$: Similarity of item i with target item p

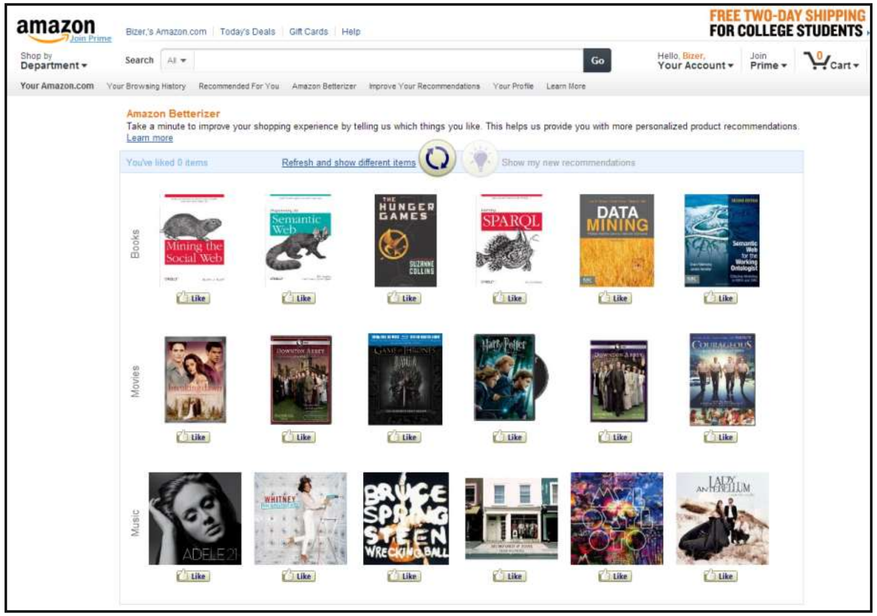
- **No need to adjust rating scale as we only use ratings by Alice**

Offline Pre-Calculations for Item-Based Filtering

- **Item-based filtering does not solve the scalability problem itself, but as there are usually less items than users, we can pre-calculate the item similarities and store them in memory.**
- **Neighborhood size is typically also limited to a specific size k**
  - An analysis of the MovieLens dataset indicates a k of 20 to 50 items is reasonable (Herlocker et al. 2002)
  - Not all neighbors are taken into account for the prediction, as Alice most likely only rated a small subset of the neighbors
- **Memory requirements**
  - Up to n^2 pair-wise similarities to be memorized (n = number of items) in theory
  - In practice, the memory requirements are significantly lower as
    - many items have no co-ratings (heavy metal and samba CDs)
    - neighborhood size often limited to k items above minimum similarity threshold

Explicit Ratings

- **Explicit ratings are probably the most precise ratings**
- **Commonly used response scales:**
  - 1 to 5 Likert scales
  - Like (sometimes also Dislike)
- **Main problems**
  - Users often not willing to rate items
    - number of ratings likely small
      → poor recommendation quality
  - How to stimulate users to rate more items?
    - Example: Amazon Betterizer (active learning)
- **Alternative**
  - Use implicit ratings
    (in addition to explicit ones)



Implicit Ratings

- **Events potentially interpretable as positive ratings**
  - items bought
  - clicks, page views
  - time spent on some page
  - time a movie was watched …
- **Advantage**
  - implicit ratings can be collected constantly by the web site or application in which the recommender system is embedded
  - collection of ratings does not require additional effort from the user
- **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
- **Most deployed collaborative filtering systems rely on implicit ratings**

Collaborative Filtering Discussion

- **Pros:**
  - well-understood, works well in some domains
  - requires no explicit item descriptions or demographic user profiles
- **Cons:**
  - requires user community to give enough ratings
    (most real-world systems thus employ implicit ratings)
  - no exploitation of other sources of recommendation knowledge
    (demographic data, item descriptions)
  - Cold Start Problem
    - how to recommend new items?
    - what to recommend to new users?
  - Approaches for dealing with the Cold Start Problem
    - ask/force users to rate a set of items (unrealistic)
    - use another method or combination of methods (e.g., content-based, demographic or simply non-personalized) until enough ratings are collected (see hybrid recommendation)

Chapter Outline

1. **Usage Data Collection**
2. **Usage Data Preparation**
   1. User and Session Identification
   2. Data Aggregation and Semantic Enrichment
3. **Usage Mining Tasks**
4. **Recommender Systems**
   1. Collaborative Filtering
   2. Content-based Recommendation
   3. Model-based Recommendation
   4. Hybrid Recommendation
   5. Evaluating Recommender Systems
   6. Attacks on Recommender Systems

Literature

- Bing Liu: Web Data Mining. Chapter 12: Web Usage Mining. 2011.
- Jannach, et al.: Recommender Systems: An Introduction. 2011.
- Charu Aggarwal: Recommender Systems: The Textbook. 2016.