Ενότητα 8

Παροχή Συστάσεων

Νίκος Καρακαπιλίδης

Industrial Management & Information Systems Lab, MEAD
University of Patras, Greece
nikos@mech.upatras.gr

Μαθησιακοί στόχοι

- Εξοικείωση με τους βασικούς αλγόριθμους παροχής συστάσεων

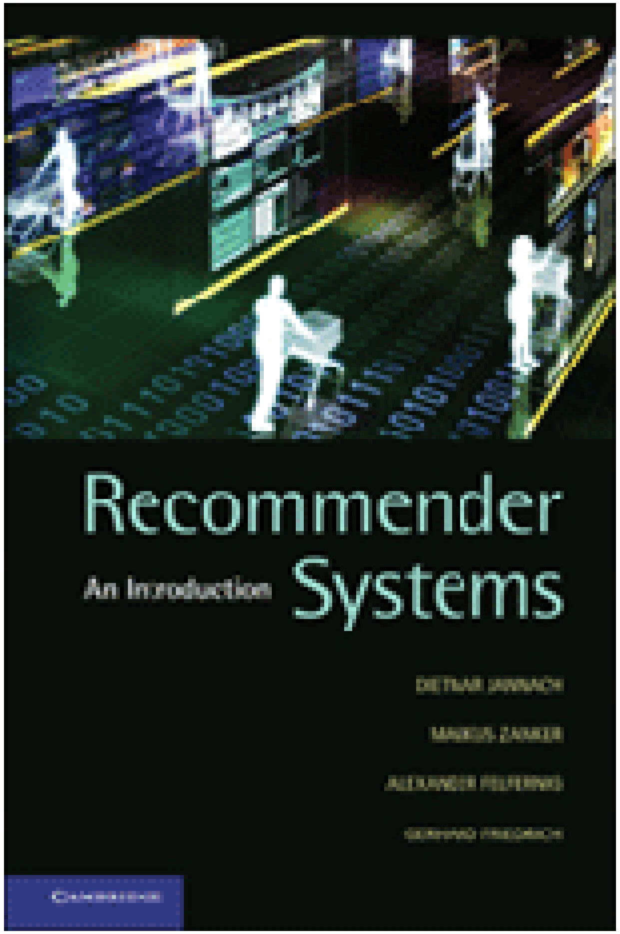
- Εισαγωγή σε τεχνικές συσσωμάτωσης ατομικών βαθμολογιών για παροχή συστάσεων σε ομάδες

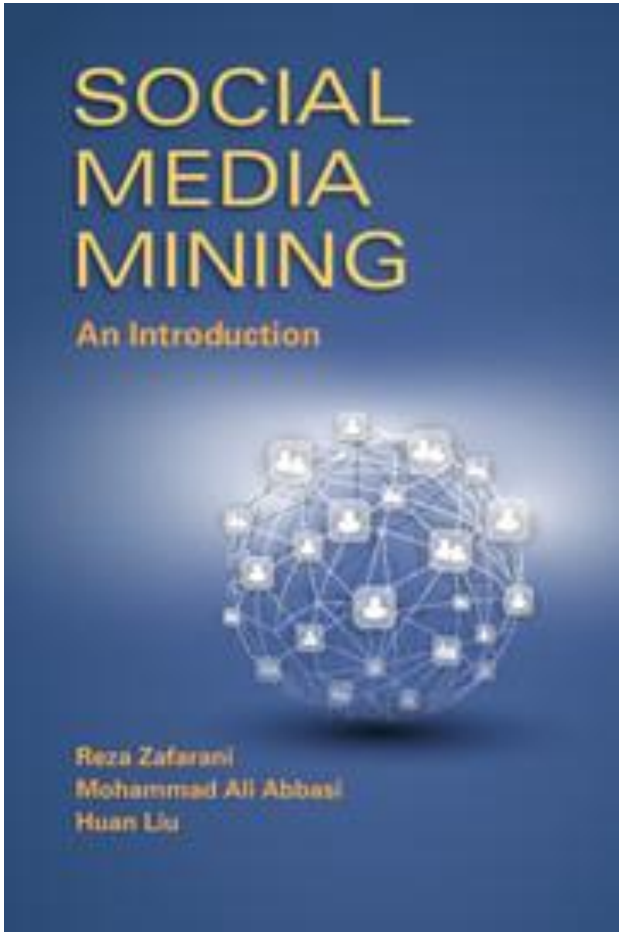
- Κατανόηση των στόχων της παροχής εξηγήσεων για τις συστάσεις ενός συστήματος και εισαγωγή σε σχετικές προσεγγίσεις

Περιεχόμενα ενότητας

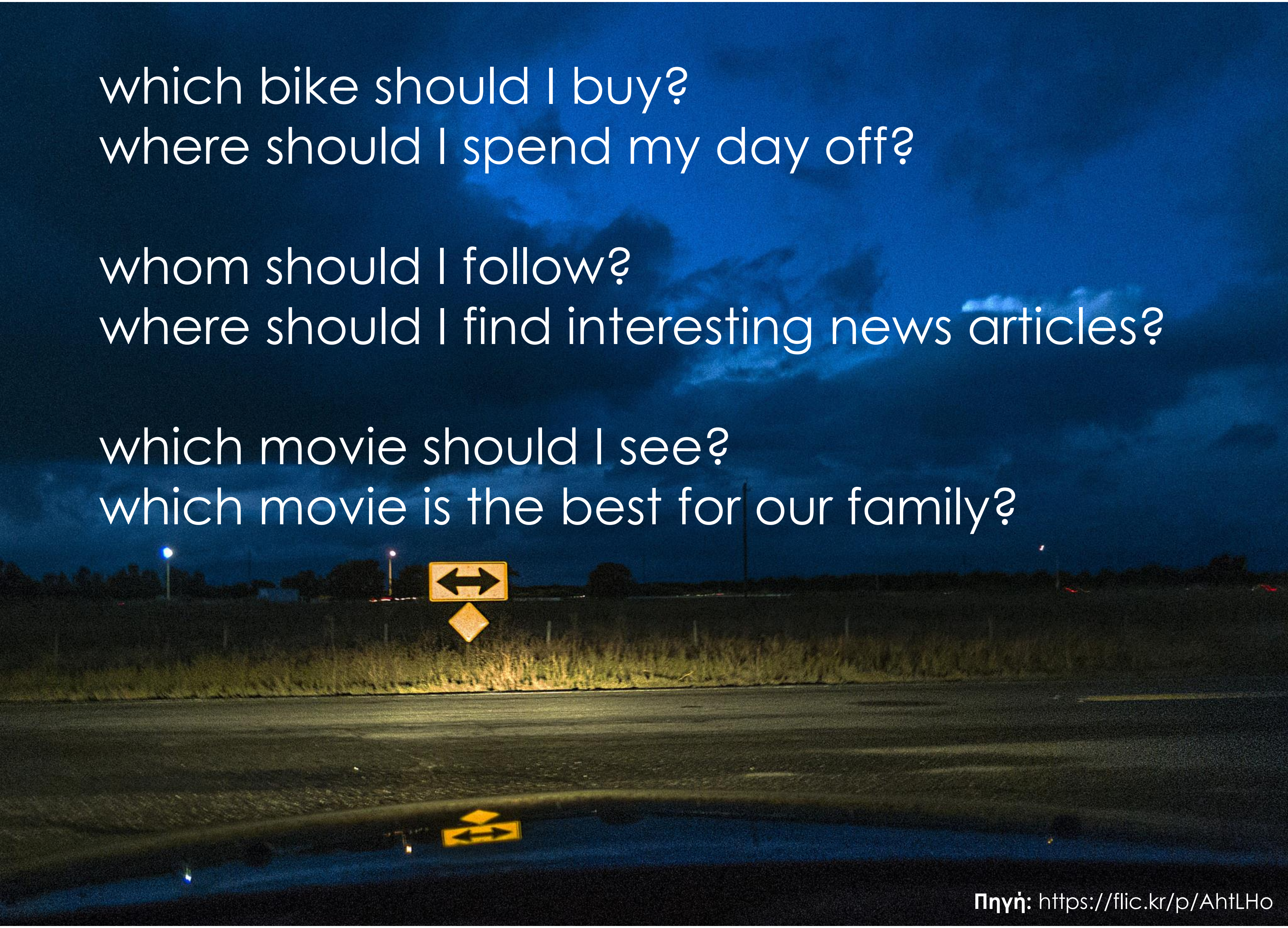
- Αλγόριθμοι παροχής συστάσεων
  - Collaborative Filtering
  - Content-Based Recommendation
  - Knowledge-Based Recommendation
  - Hybrid recommendation approaches
- Παρέχοντας συστάσεις σε ομάδες
- Εξηγώντας τις παρεχόμενες συστάσεις

Acknowledgements





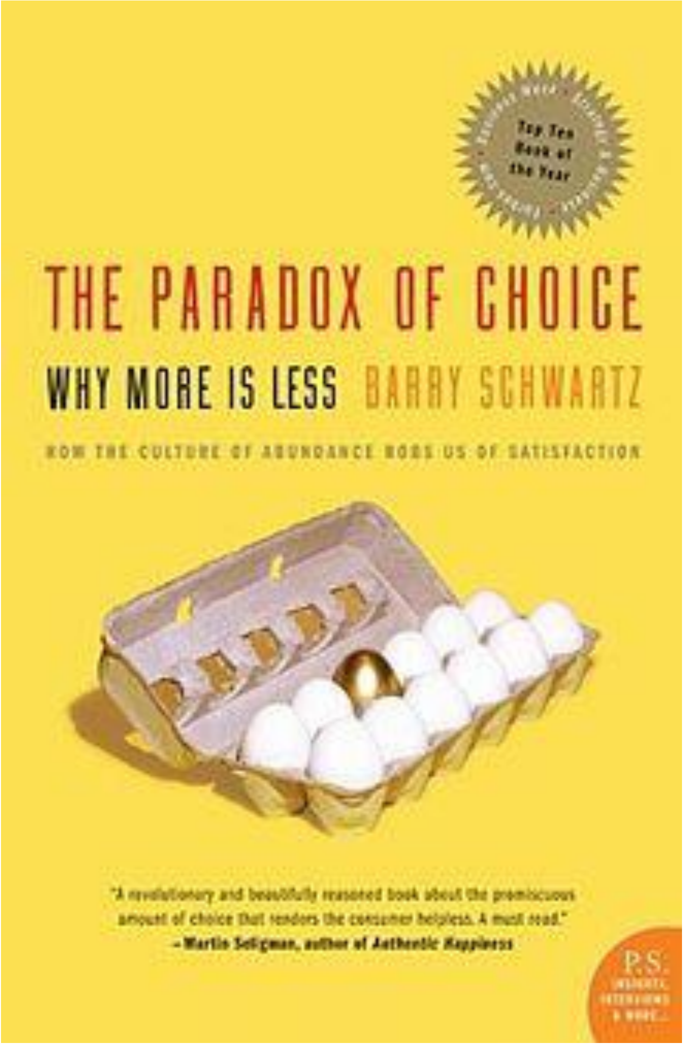
- Building on material existing at: http://www.recommenderbook.net/teaching-material/slides & http://dmml.asu.edu/smm/slides/



Πηγή: https://flic.kr/p/AhtLHo

The paradox of choice

- Too many choices …
  - … often with no obvious advantage among them
  - "choice overload can make you question your decisions before you even make them, it can set you up for unrealistically high expectations, and it can make you blame yourself for failures … this can lead to decision-making paralysis, anxiety, and stress"
- Not enough resources to check all options
  - Information overload
  - Limited knowledge or experience
  - Limited time



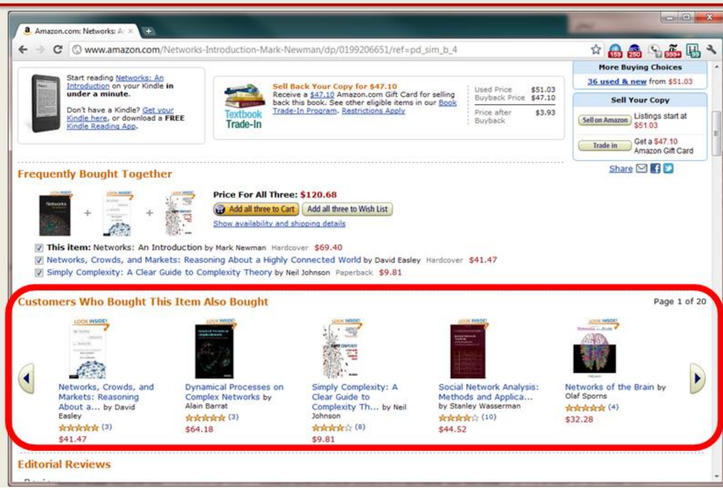
Goal of Recommendation

Come up with a short list of items that fits user's interests

Πηγή: https://flic.kr/p/modui

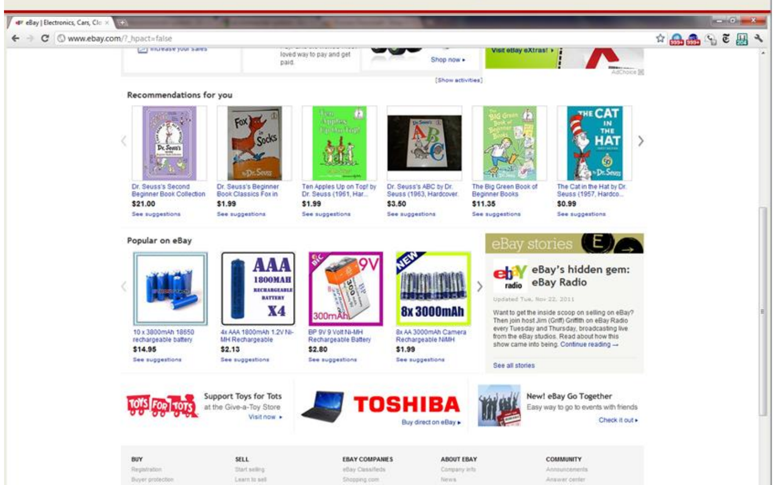
Recommender Systems - Examples

Book recommendation in Amazon

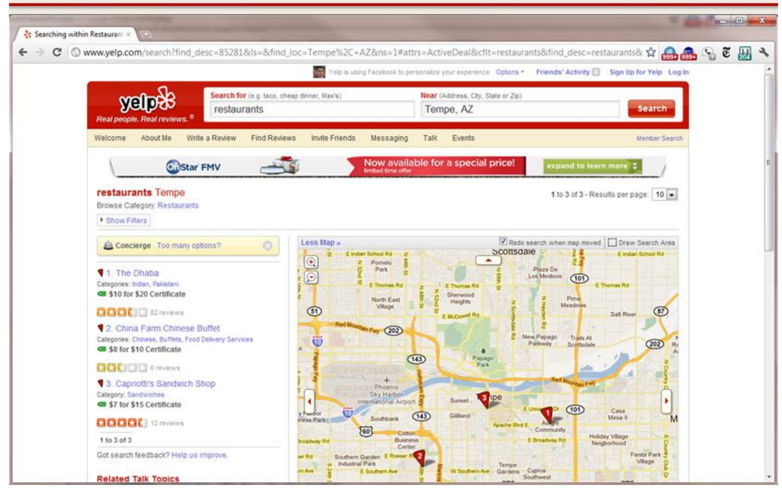


Video clip recommendation in YouTube

Product Recommendation in ebay



Restaurant Recommendation in Yelp



Main idea

Use historical data such as the user's past preferences or similar users' past preferences to predict future likes

- Users' preferences are likely to remain stable, and change smoothly over time
  - By watching the past users' or groups' preferences, we try to predict their future likes
  - Then we can recommend items of interest to them
- Formally, a recommender system takes a set of users U and a set of items I and learns a function f such that:

$$f : U \times I \to \mathbb{R}$$

Recommendation vs. Search

- Search engines find results that match the query provided by the user
- The results are generally provided as a list ordered with respect to the relevance of the item to the given query
- Consider the query "best 2015 movie to watch"
  - The same results for an 8 year old and an adult

Search engines' results are not customized

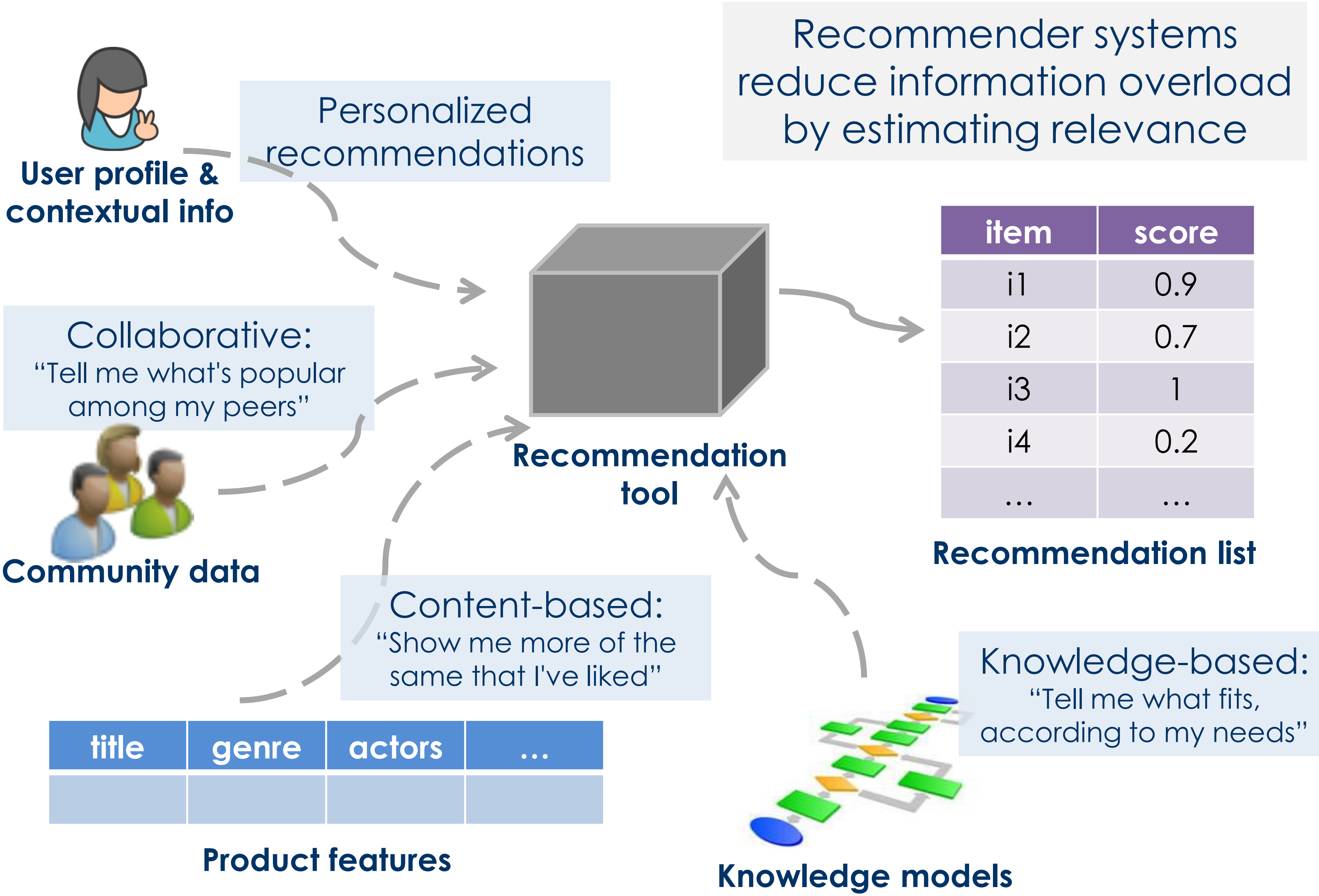
Challenges

- The Cold Start Problem
  - Recommendation systems often use historical data provided by the user to recommend items. However, when individuals join sites, they have no history (e.g. they haven't bought any product). This makes it hard to infer what they are going to like.
- Data Sparsity
  - Similar to the cold-start problem, data sparsity is when not enough historical or prior information is available. Unlike the cold start problem, this is in the system as a whole and is not specific to an individual.
- Attacks
  - e.g. Push Attack (pushing the ratings up by creating fake users)
- Privacy
  - Employing user's private information to recommend to others
- Explanation
  - Recommendation systems often recommend items without any explanation of why recommending them

Basic Recommendation Algorithms

Πηγή: https://flic.kr/p/fsM11Y

Paradigms of recommender systems



item	score
i1	0.9
i2	0.7
i3	1
i4	0.2
...	...

title	genre	actors	...

Paradigms of recommender systems

Collaborative Filtering (CF)

- The most prominent approach to generate recommendations
  - Used by large e-commerce sites
  - Various algorithms and variations exist
  - Applicable in many domains (book, movies, DVDs, ..)
- Approach
  - Use the "wisdom of the crowd" to recommend items

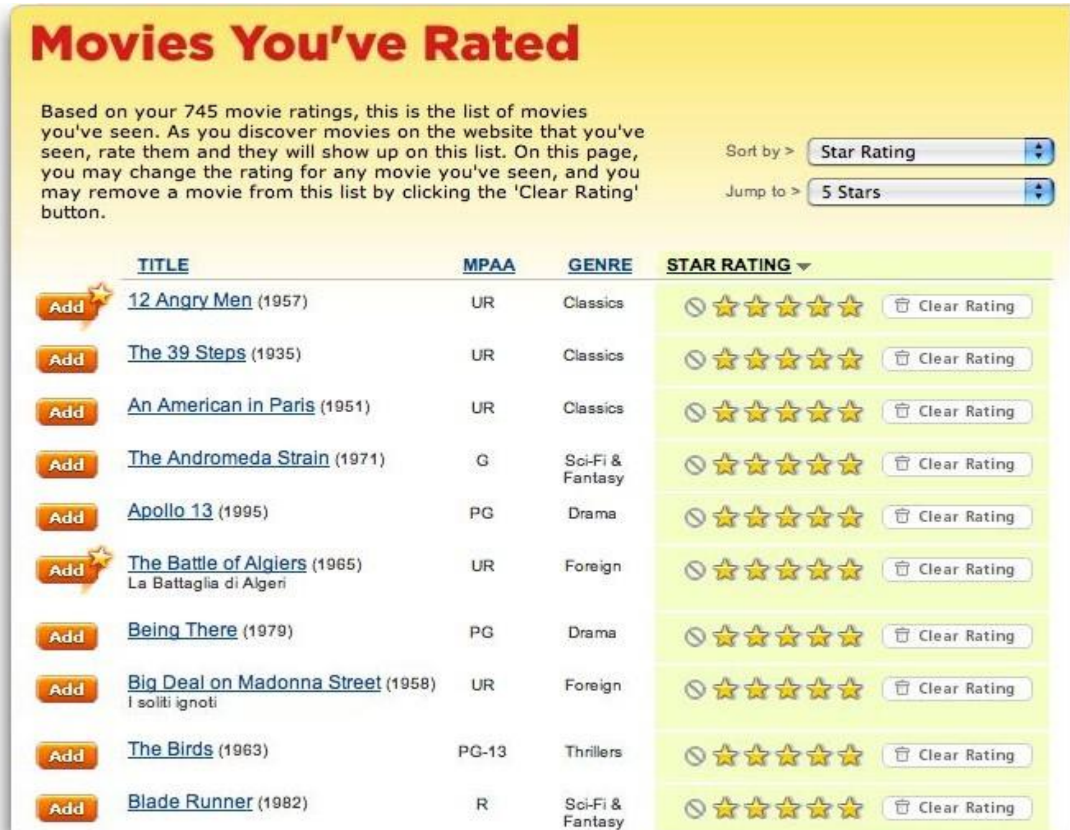
- Basic idea & assumption
  - Implicitly or explicitly, users give ratings to catalog items
  - Customers who had similar tastes in the past, will have similar tastes in the future

Pure CF Approaches

- **Input**
  - Just a matrix of given user-item ratings **(no additional information about the users or content of the items)**
- **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

Πηγή: https://flic.kr/p/5g77wy

Rating Items



Πηγή: http://www.askdavetaylor.com/view_your_favorite_netflix_movies/

- Explicit ratings: directly entered by a user
- Implicit ratings: inferred from user behavior (e.g. play lists or music listened to, for a music Rec system – or even the amount of time users spent on a webpage)

Types of CF Algorithms

- Memory-based: Recommendation is directly based on previous ratings in a stored matrix that describes user-item relations

	Dogville	Léon	Taxi Driver	Pulp Fiction	The Godfather
Nikos	5	4	4	4	5
Alex	3	1	2	3	3
Helen	4	3	4	3	5
Jim	3	3	1	5	4
Zoe	1	5	5	2	1

- Model-based: We assume that an underlying model (hypothesis) governs the way users rate items
  - The model is then used to recommend ratings
  - An example model: users rate low budget movies poorly

Memory-Based Collaborative Filtering

- User-based CF

  Users with similar previous ratings for items are likely to rate future items similarly

	I1	I2	I3	I4
U1	1	2	4	4
U2	1	2	4	?
U3	2	5	2	2
U4	5	2	3	3

- Item-based CF

  Items that have received similar ratings previously from users are likely to receive similar ratings from future users

	I1	I2	I3	I4
U1	1	2	4	4
U2	1	2	4	?
U3	2	5	2	2
U4	5	2	3	3

CF Algorithm

```
1.  Weigh all users/items with respect to their
    similarity with the current user/item
2.  Select a subset of the users/items (neighbors) as
    recommenders
3.  Predict the rating of the user for specific items
    using neighbors' ratings for the same (or similar)
    items
4.  Recommend items with the highest predicted rank
```

User-based nearest-neighbor CF (1/4)

- 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** have rated item i
    - use a function (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 ones

- 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 CF (2/4)

	Lion King	Aladdin	Mulan	Anastasia
John	3	0	3	3
Joe	5	4	0	2
Jill	1	2	4	2
Jane	3	?	1	0
Jorge	2	2	0	1

Predict Jane's rating for Aladdin
(which Jane has not yet rated or seen)

Basic concerns

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

Similarity between Users (or Items)

- Cosine Similarity

$$sim(U_i, U_j) = cos(U_i, U_j) = \frac{U_i \cdot U_j}{||U_i||\ ||U_j||} = \frac{\sum_k r_{i,k} r_{j,k}}{\sqrt{\sum_k {r_{i,k}}^2}\sqrt{\sum_k {r_{j,k}}^2}}.$$

- Pearson Correlation Coefficient

$$sim(U_i, U_j) = \frac{\sum_k (r_{i,k} - \bar{r}_i)(r_{j,k} - \bar{r}_j)}{\sqrt{\sum_k (r_{i,k} - \bar{r}_i)^2}\sqrt{\sum (r_{j,k} - \bar{r}_j)^2}}.$$

User-based nearest-neighbor CF (3/4)

Step 1: Calculate average ratings

$$\bar{r}_{John} = \frac{3+3+0+3}{4} = 2.25$$

$$\bar{r}_{Joe} = \frac{5+4+0+2}{4} = 2.75$$

$$\bar{r}_{Jill} = \frac{1+2+4+2}{4} = 2.25$$

$$\bar{r}_{Jane} = \frac{3+1+0}{3} = 1.33$$

$$\bar{r}_{Jorge} = \frac{2+2+0+1}{4} = 1.25$$

Step 2: Calculate user-user similarity

$$sim(Jane, John) = \frac{3\times 3+1\times 3+0\times 3}{\sqrt{10}\,\sqrt{27}} = 0.73$$

$$sim(Jane, Joe) = \frac{3\times 5+1\times 0+0\times 2}{\sqrt{10}\,\sqrt{29}} = 0.88$$

$$sim(Jane, Jill) = \frac{3\times 1+1\times 4+0\times 2}{\sqrt{10}\,\sqrt{21}} = 0.48$$

$$sim(Jane, Jorge) = \frac{3\times 2+1\times 0+0\times 1}{\sqrt{10}\,\sqrt{5}} = 0.84$$

User-based nearest-neighbor CF (4/4)

Step 3: Update ratings (assume that neighborhood size = 2)

$$
\begin{aligned}
r_{Jane,Aladdin} &= \bar{r}_{Jane} + \frac{sim(Jane, Joe)(r_{Joe,Aladdin} - \bar{r}_{Joe})}{sim(Jane, Joe) + sim(Jane, Jorge)} \\
&\quad + \frac{sim(Jane, Jorge)(r_{Jorge,Aladdin} - \bar{r}_{Jorge})}{sim(Jane, Joe) + sim(Jane, Jorge)} \\
&= 1.33 + \frac{0.88(4 - 2.75) + 0.84(2 - 1.25)}{0.88 + 0.84} = 2.33
\end{aligned}
$$

Item-based CF

Step 1: Calculate average ratings

$$\bar{r}_{Lion\ King} = \frac{3+5+1+3+2}{5} = 2.8$$

$$\bar{r}_{Aladdin} = \frac{0+4+2+2}{4} = 2.$$

$$\bar{r}_{Mulan} = \frac{3+0+4+1+0}{5} = 1.6$$

$$\bar{r}_{Anastasia} = \frac{3+2+2+0+1}{5} = 1.6$$

Step 2: Calculate item-item similarity

$$sim(Aladdin, Lion\ King) = \frac{0\times 3+4\times 5+2\times 1+2\times 2}{\sqrt{24}\sqrt{39}} = 0.84$$

$$sim(Aladdin, Mulan) = \frac{0\times 3+4\times 0+2\times 4+2\times 0}{\sqrt{24}\sqrt{25}} = 0.32$$

$$sim(Aladdin, Anastasia) = \frac{0\times 3+4\times 2+2\times 2+2\times 1}{\sqrt{24}\sqrt{18}} = 0.67$$

Step 3: Update ratings (assume that neighborhood size = 2)

$$\begin{aligned} r_{Jane,Aladdin} &= \bar{r}_{Aladdin} + \frac{sim(Aladdin, Lion\ King)(r_{Jane,Lion\ King} - \bar{r}_{Lion\ King})}{sim(Aladdin, Lion\ King) + sim(Aladdin, Anastasia)} \\ &+ \frac{sim(Aladdin, Anastasia)(r_{Jane,Anastasia} - \bar{r}_{Anastasia})}{sim(Aladdin, Lion\ King) + sim(Aladdin, Anastasia)} \\ &= 2 + \frac{0.84(3 - 2.8) + 0.67(0 - 1.6)}{0.84 + 0.67} = 1.40 \end{aligned}$$

CF pros & cons

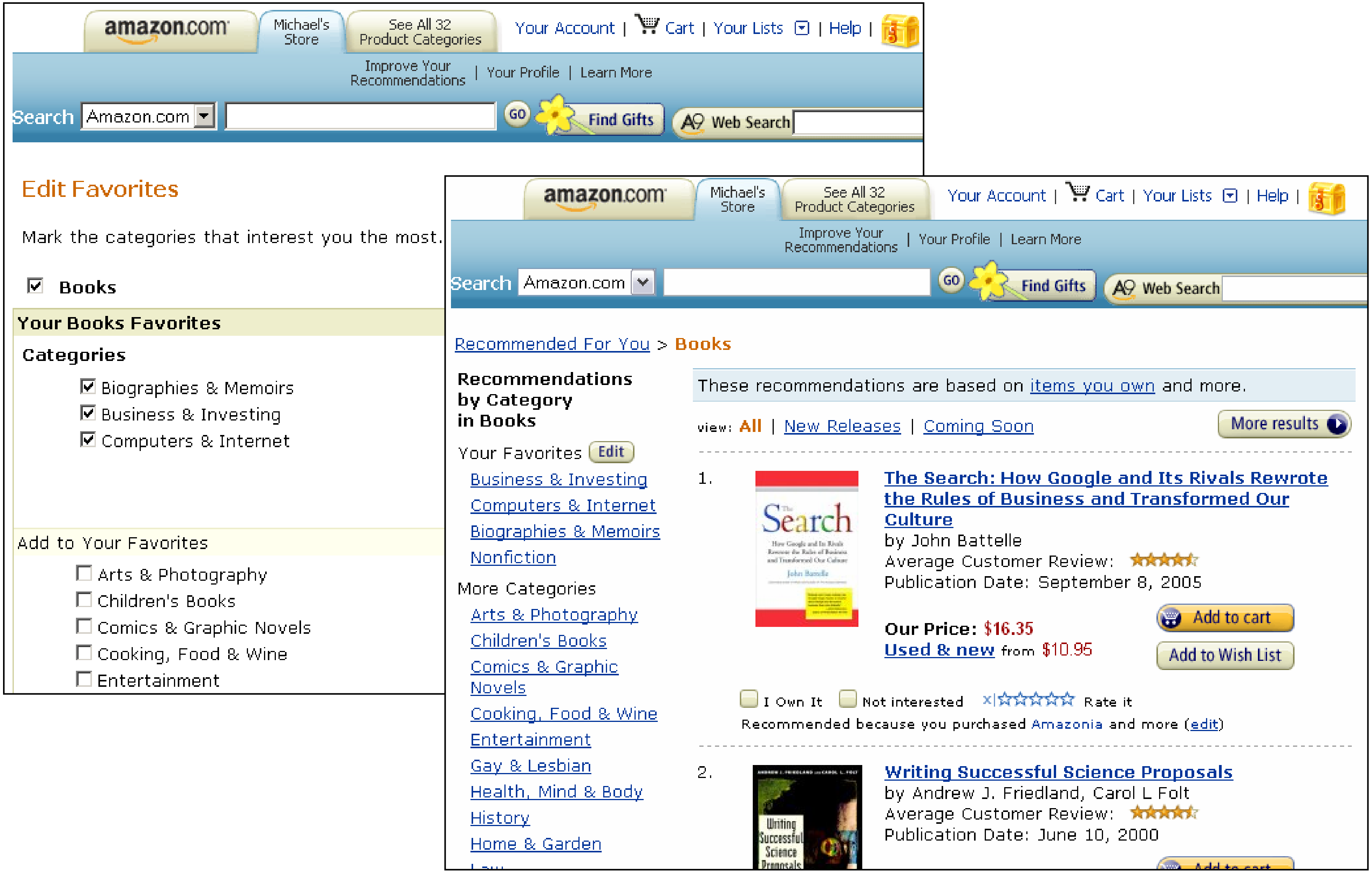
- Pros
  - well-understood
  - works well in some domains
  - no knowledge engineering required
- Cons
  - requires user community
  - sparsity problems
  - no integration of other knowledge sources
  - no explanation of results

Πηγή: https://flic.kr/p/pZXTtL

Content-Based Recommendation

- Content-based recommendation systems are based on the fact that a user’s interest should match the description of the items that she should be recommended by the system
  - Remember that CF methods do not require any information about the items
- 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
  - To learn user preferences
  - To find and recommend items that are “similar” to the user preferences

Content-based Recommendation - Example



Content representation

- Most CB-recommendation techniques were applied to recommending text documents (e.g. web pages, messages)
- 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) vs. Unstructured (free-text description)

items

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

Content-based Recommendation Algorithm

1. Describe the items to be recommended
2. Create a profile of the user that describes the types of items the user likes
3. Compare items with the user profile to determine what to recommend

Require: User i's Profile Information, Item descriptions for items $j \in \{1, 2, \ldots, n\}$, k keywords, r number of recommendations.

1: **return** r recommended items.
2: $U_i = (u_1, u_2, \ldots, u_k)$ = user i's profile vector;
3: $\{I_j\}_{j=1}^{n} = (i_{j,1}, i_{j,2}, \ldots, i_{j,k})$= item j's description vector;
4: $s_{i,j} = sim(U_i, I_j)$;
5: Return top r items with maximum similarity $s_{i,j}$.

The user profile is often updated automatically in response to feedback on the desirability of items that are presented to the user

Content-based Recommendation Methods

- We first represent both user profiles and item descriptions by vectorizing them using a set of ***k*** keywords
- We can vectorize (e.g., using TF-IDF) both users and items and compute their similarity

$$I_j = (i_{j,1}, i_{j,2}, \ldots, i_{j,k}) \qquad U_i = (u_{i,1}, u_{i,2}, \ldots, u_{i,k}).$$

$$sim(U_i, I_j) = cos(U_i, I_j) = \frac{\sum_{l=1}^{k} u_{i,l} i_{j,l}}{\sqrt{\sum_{l=1}^{k} u_{i,l}^2} \sqrt{\sum_{l=1}^{k} i_{j,l}^2}}$$

- We then recommend the top most similar items to the user

Term Frequency - Inverse Document Frequency (TF-IDF)

- Simple keyword representation is problematic
  - Particularly in automatic extraction, since not every word has similar importance while longer documents have a higher chance to have an overlap with the user profile

- Standard measure: TF-IDF
  - 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

For a keyword $\boldsymbol{i}$ and a document $\boldsymbol{j}$

- $\boldsymbol{TF(i,j)}$ is the term frequency of keyword $\boldsymbol{i}$ in document $\boldsymbol{j}$
- $\boldsymbol{IDF(i) = log\frac{N}{n(i)}}$

  N: number of all recommendable documents

  $n(i)$: number of documents from N in which keyword i appears
- $\boldsymbol{TF\text{-}IDF(i,j) = TF(i,j) * IDF(i)}$

CB approaches - Critique

- In contrast to CF approaches, content-based techniques do not require user community in order to work
- Presented approaches aim to learn a model of user's interest preferences based on explicit or implicit feedback
  - Deriving implicit feedback from user behavior can be problematic
- Evaluations show that a good recommendation accuracy can be achieved with help of machine learning techniques
  - These techniques do not require a user community
- Danger exists that recommendation lists contain too many similar items
  - All learning techniques require a certain amount of training data
- Pure content-based systems are rarely found in commercial settings

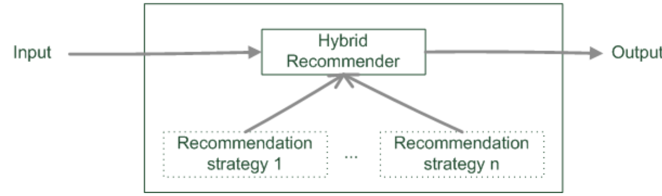
Knowledge-Based Recommendation

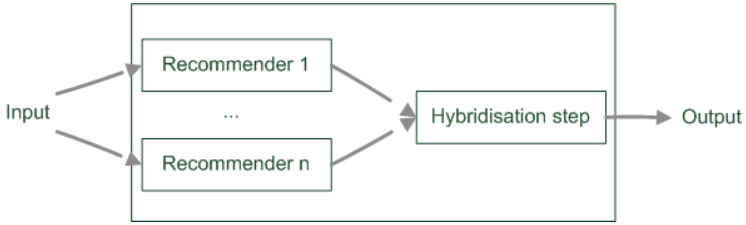
- Constraint-based
  - Based on explicitly defined set of recommendation rules
    - "IF user requires A THEN proposed item should possess feature B"
  - Fulfill recommendation rules
- Case-based
  - Based on different types of similarity measures
  - Retrieve items that are similar to specified requirements

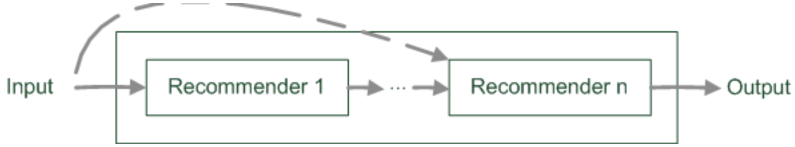
- Both approaches are similar in their conversational recommendation process
  - Users specify the requirements
  - System tries to identify solutions
  - If no solution can be found, users change requirements

Hybrid recommendation approaches

- All three base techniques are naturally incorporated by a good sales assistant (at different stages of the sales act) but have their shortcomings
  - For instance, cold start problems
- Building on two (or more) techniques
  - Avoid some of the shortcomings
  - Reach desirable properties not present in individual techniques
- Different hybridization designs
  - Monolithic exploiting different features
  - Parallel use of several techniques
  - Pipelined invocation of different systems







Recommendation to a Group

Πηγή: https://flic.kr/p/bw7bUz

Recommendation to Groups

- Find content of interest to all members of a group of socially acquainted individuals
- Examples
  - A movie for friends to watch together
  - A travel destination for a class to spend the holiday break
  - A good restaurant for colleagues to have a working lunch
  - A music to be played in a public area
- Tasks of a Group Recommender System
  - Acquiring preferences
  - Generating recommendations
  - Explaining recommendations
  - Helping group members to achieve consensus

Aggregation Strategies

- Average Satisfaction
  - Average everyone's ratings and choose the max

$$R_i = \frac{1}{n} \sum_{u \in G} r_{u,i}$$

- Least Misery
  - Minimize the dissatisfaction among group's members (max of the mins of all)

$$R_i = \min_{u \in G} r_{u,i}$$

- Most Pleasure
  - The maximum of individuals' maximum ratings is taken as group's rating

$$R_i = \max_{u \in G} r_{u,i}$$

Aggregation Strategies – An example

	soda	water	tea	coffee
Nikos	1	3	1	1
Alex	2	2	4	2
Helen	3	3	4	5

Average Satisfaction

$$\begin{aligned} R_{Soda} &= \frac{1+2+3}{3} = 2. \\ R_{Water} &= \frac{3+2+3}{3} = 2.66 \\ R_{Tea} &= \frac{1+4+4}{3} = 3. \\ R_{Coffee} &= \frac{1+2+5}{3} = 2.66 \end{aligned}$$

Least Misery

$$\begin{aligned} R_{Soda} &= \min\{1,2,3\} = 1 \\ R_{Water} &= \min\{3,2,3\} = 2 \\ R_{Tea} &= \min\{1,4,4\} = 1 \\ R_{Coffee} &= \min\{1,2,5\} = 1 \end{aligned}$$

Most Pleasure

$$\begin{aligned} R_{Soda} &= \max\{1,2,3\} = 3 \\ R_{Water} &= \max\{3,2,3\} = 3 \\ R_{Tea} &= \max\{1,4,4\} = 4 \\ R_{Coffee} &= \max\{1,2,5\} = 5 \end{aligned}$$

Providing explanations

Πηγή: https://flic.kr/p/4oU4d1

Explanations in recommender systems

- Additional information to explain the system's output following some objectives
- Many recommender systems act like "black boxes", providing no transparency into the rationale of the recommendation process
  - In order to increase their quality, recommender systems must be able to explain what they do and justify their actions in terms that are understandable to the user
- Why do we need explanations?
  - A selling agent may be interested in promoting particular products with some reason
  - A buying agent is concerned about making the right buying decision

Objectives of explanations (1/2)

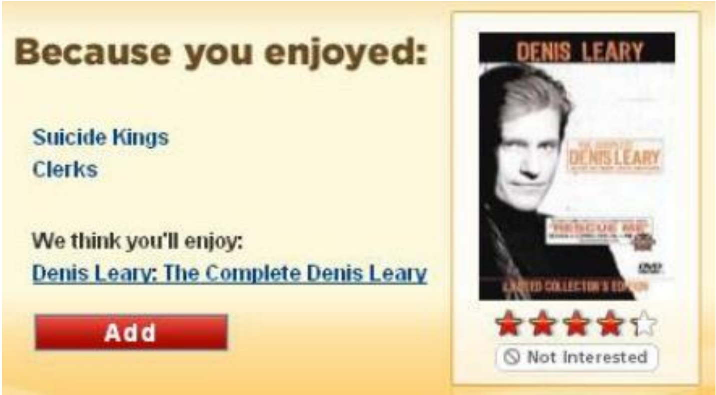
- Transparency
  - Provide information so the user can comprehend the reasoning used to generate a specific recommendation (e.g. why an item was preferred over another)
- Validity
  - Allow a user to check the validity of a recommendation
- Trustworthiness
  - Trust building can be viewed as a mechanism for reducing the complexity of human decision making in uncertain situations and the uncertainty about the quality of a recommendation
- Persuasiveness
  - Persuasive explanations for recommendations aim to change the user’s buying behavior
- Effectiveness
  - The support a user receives for making high-quality decisions
  - Help customers discover their preferences
  - Help users make better decisions

Objectives of explanations (2/2)

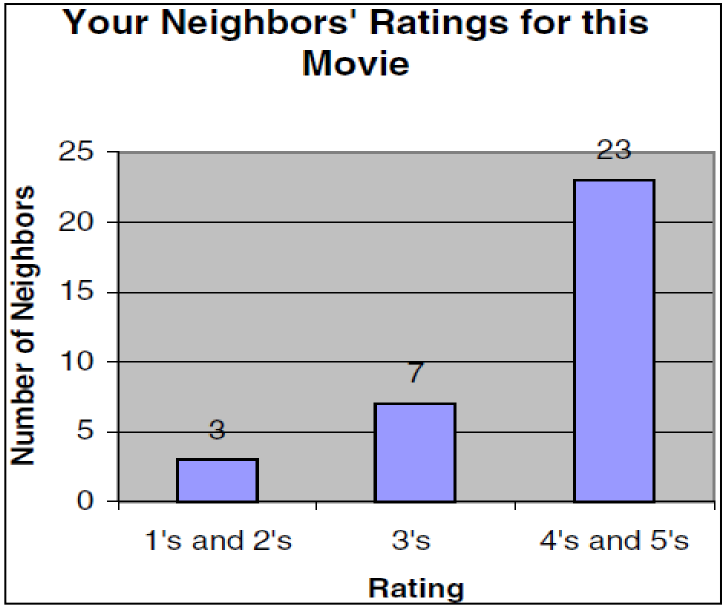
- Efficiency
  - Reduce the decision-making effort (e.g. time needed for decision making)
- Satisfaction
  - Improve the overall satisfaction stemming from the use of a recommender system
- Relevance
  - Additional information may be required in conversational recommenders
  - Explanations can be provided to justify why additional information is needed from the user
- Comprehensibility
  - Support the user by relating the user's known concepts to the concepts employed by the recommender
- Education
  - Educate users to help them better understand the product domain

Examples of explanations

- Similarity between items



- Similarity between users



Explanation types (1/2)

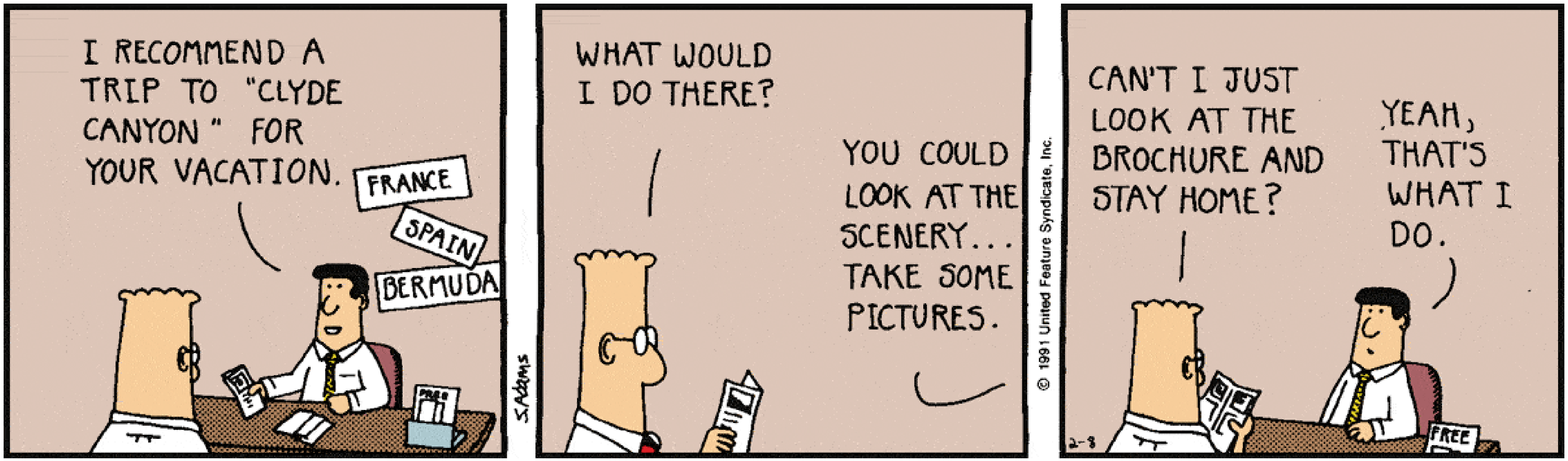
- Nearest neighbor explanation
  - Customers who bought item X also bought items Y and Z
  - Item Y is recommended because you have highly rated the related item X
- Content based explanation
  - This story deals with topics X and Y, which belong to your topic of interest
- Social-network based explanation
  - People leverage their social network to reach information and make use of trust relationships to filter information
    - Your friend X wrote that blog
    - 50% of your friends liked this item (while only 5% disliked it)

Explanation types (2/2)

- A hybrid framework for explanations building that combines:
  - multi-attribute based ranking
  - collaborative filtering
  - sentiment analysis

Hotel A is recommended to you because it has been highly rated for its **front desk** from **7 out of 10** users with **highly similar** tastes to yours, who have already chosen it. In addition, these users have expressed **positive** reviews for **Hotel A**.

Αντί επιλόγου



Βιβλιογραφία

- R. Burke, “Hybrid Web Recommender Systems”. In: P. Brusilovsky, A. Kobsa and W. Nejdl (Eds), *The Adaptive Web*, Lecture Notes in Computer Science, Vol. 4321, Springer-Verlag, Berlin, 2007, pp. 377-408 → http://link.springer.com/chapter/10.1007%2F978-3-540-72079-9_12

- L. Herlocker, J. Konstan and J. Riedl, “Explaining collaborative filtering recommendations”. In: Proceedings of the 2000 ACM Conference on Computer Supported Cooperative Work (CSCW’00), Philadelphia, Pennsylvania, 2000, pp. 241–250 → http://grouplens.org/site-content/uploads/explain-CSCW-20001.pdf

- F. Ricci, L. Rokach, B. Shapira and P.B. Kantor (Eds), “Recommender Systems Handbook”, Springer, 2011 → www.cs.bme.hu/nagyadat/Recommender_systems_handbook.pdf

- R. Zafarani, M.A. Abbasi and H. Liu, “Social Media Mining: An Introduction”, Cambridge University Press, 2014 (free online book) → http://dmml.asu.edu/smm/

Σημείωμα αδειοδότησης