Recommender Systems - introduction

Ladislav Peška,

Department of Software Engineering,
Charles University in Prague,
Czech Republic

Background, disclaimer

Content:

- What are recommender systems?
- Why we should love them?
- How they work?
- Challenges, problems, risks, practical deployment
- Discussion

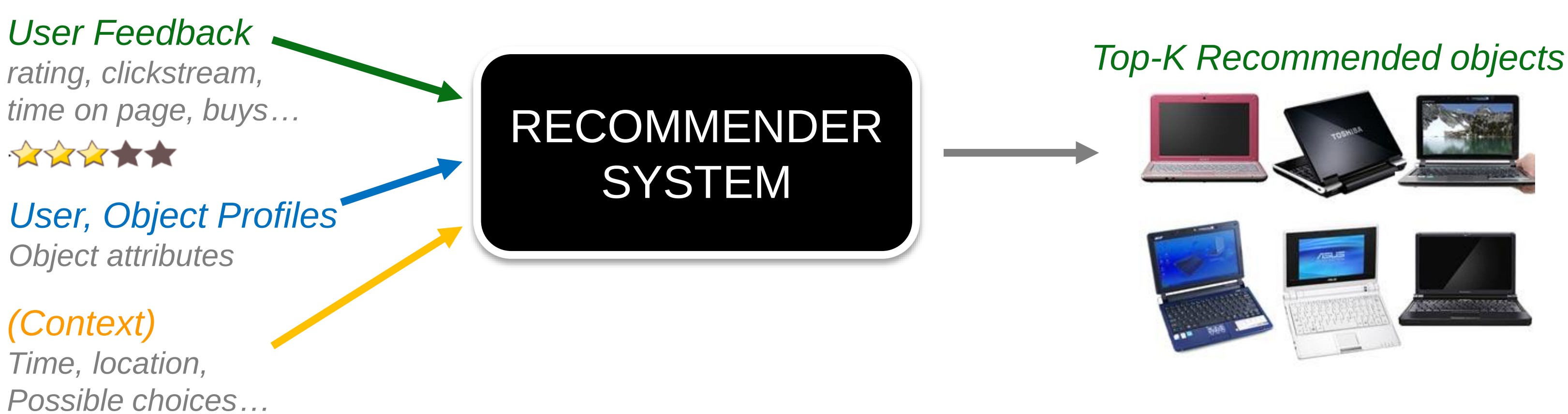
http://www.recommenderbook.net/teaching-material/slides
http://ksi.mff.cuni.cz/~peska/ndbi037/recsysIntro.pdf

You all know them, maybe you just didn’t know that so far…;-)

RECOMMENDER SYSTEMS

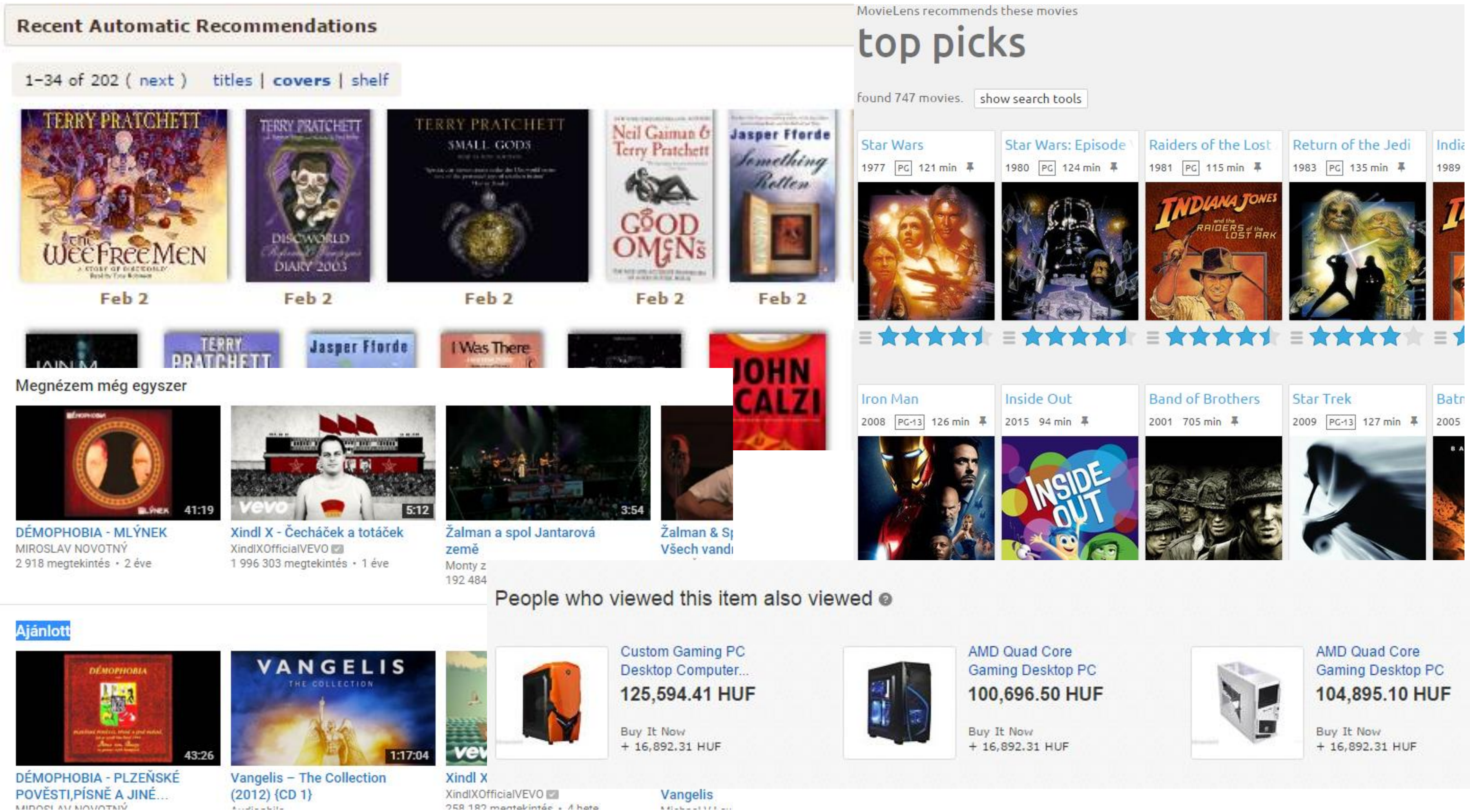
Recommender Systems

- Propose relevant items to the right persons at the right time
- Machine learning application
- Expose otherwise hard to find, uknown items
- Complementary to the catalogues, search engines etc.
- **„Win-win strategy“**



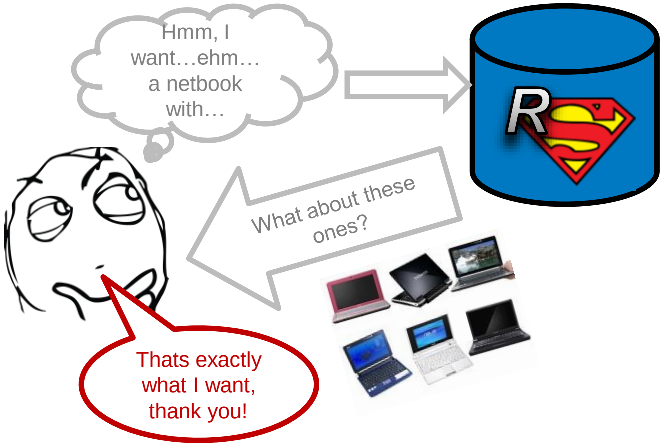
Recommender Systems are everywhere ☺

- Movies, news, books, e-commerce, web/site-wide search, social networks…



Motivation

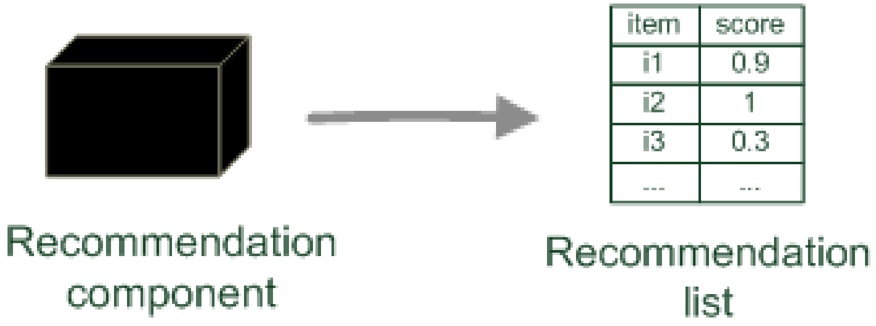
- Recommender systems
  - Are complementary to classical GUI elements *(cats. hierarchy, search queries,…)*
  - Most effective when users dont not know exactly what they want
    - Or their intent is hard to express
    - Serendipity (i.e. a lucky hit)
  - Should both increase user sattisfaction and provider’s success metrics



How do they work?

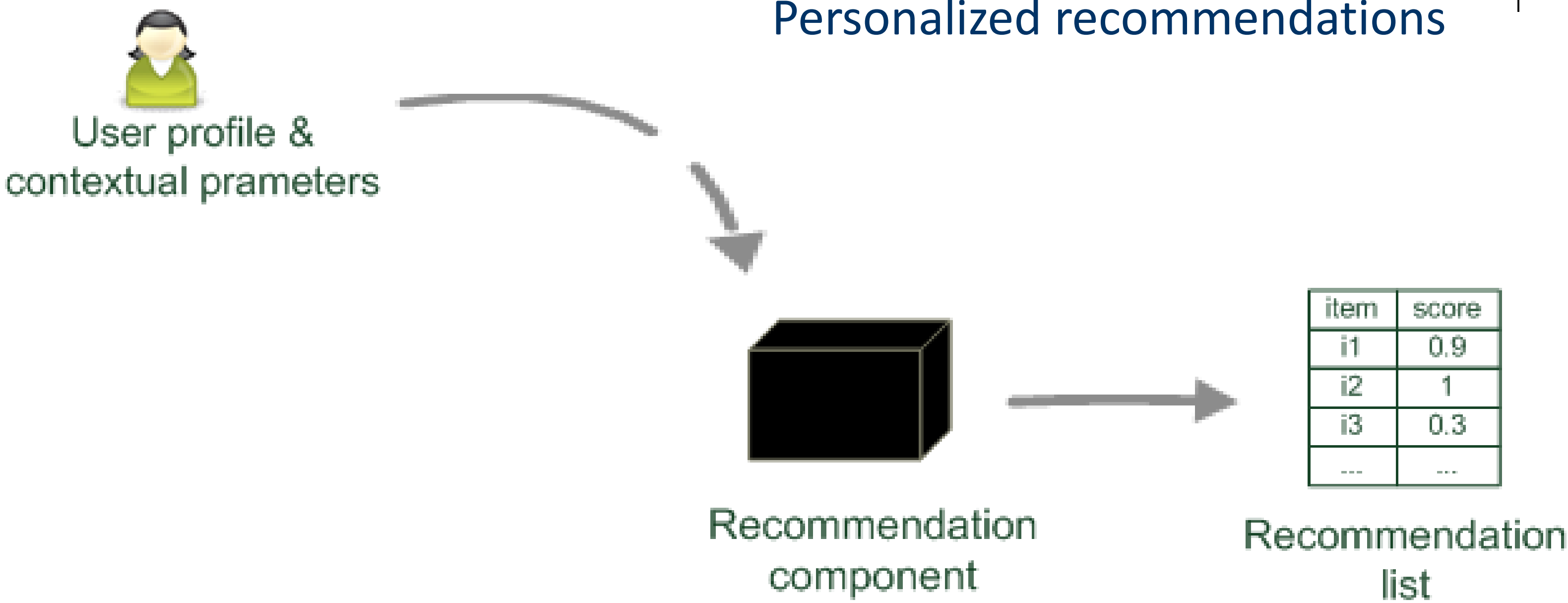
Paradigms of recommender systems

Recommender systems reduce information overload by estimating relevance



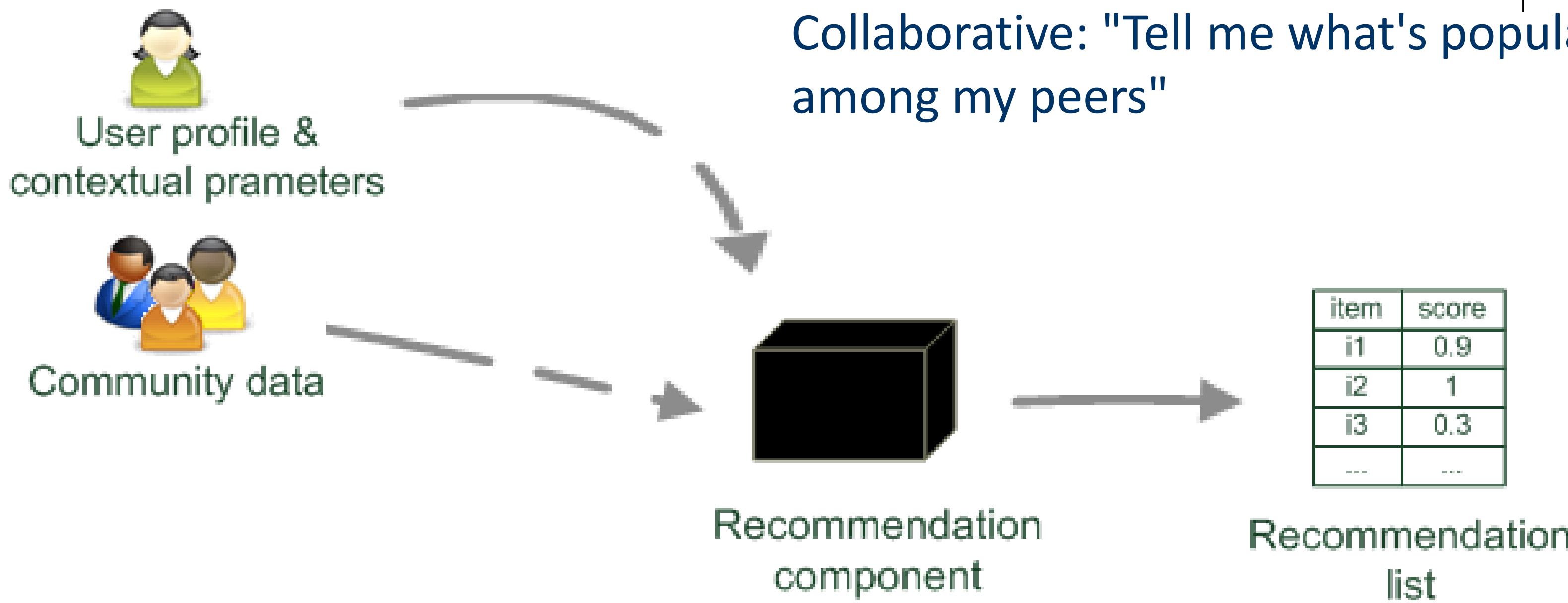
Paradigms of recommender systems

Personalized recommendations



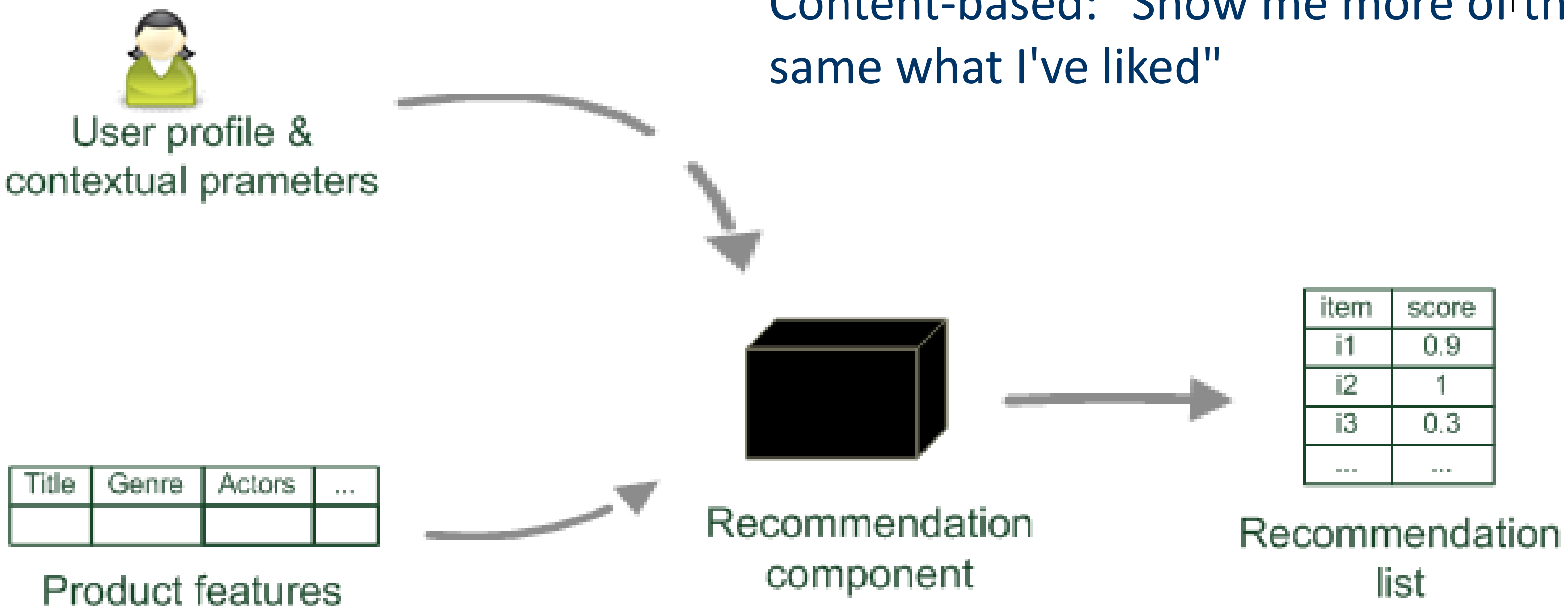
Paradigms of recommender systems

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



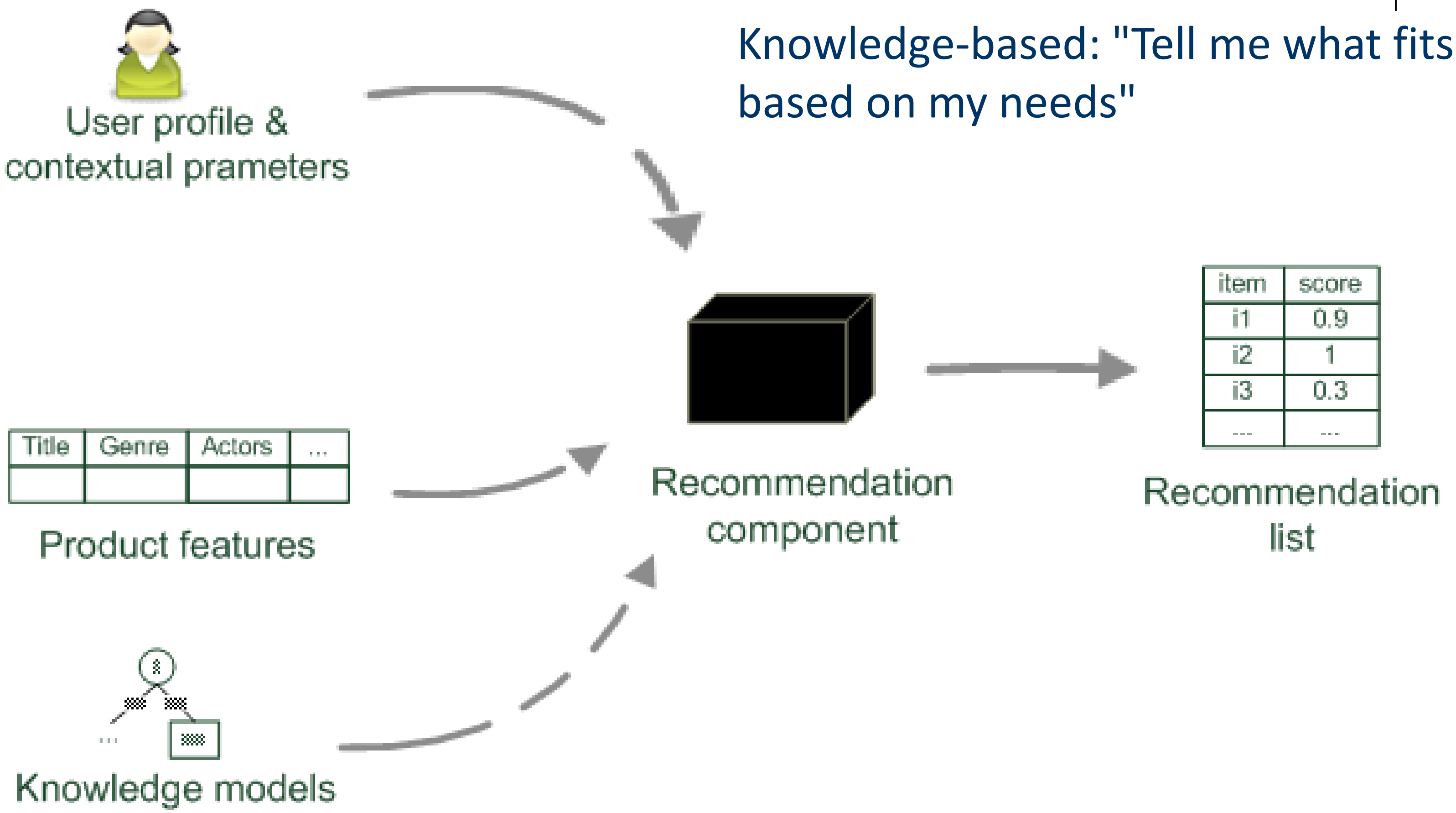
Paradigms of recommender systems

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



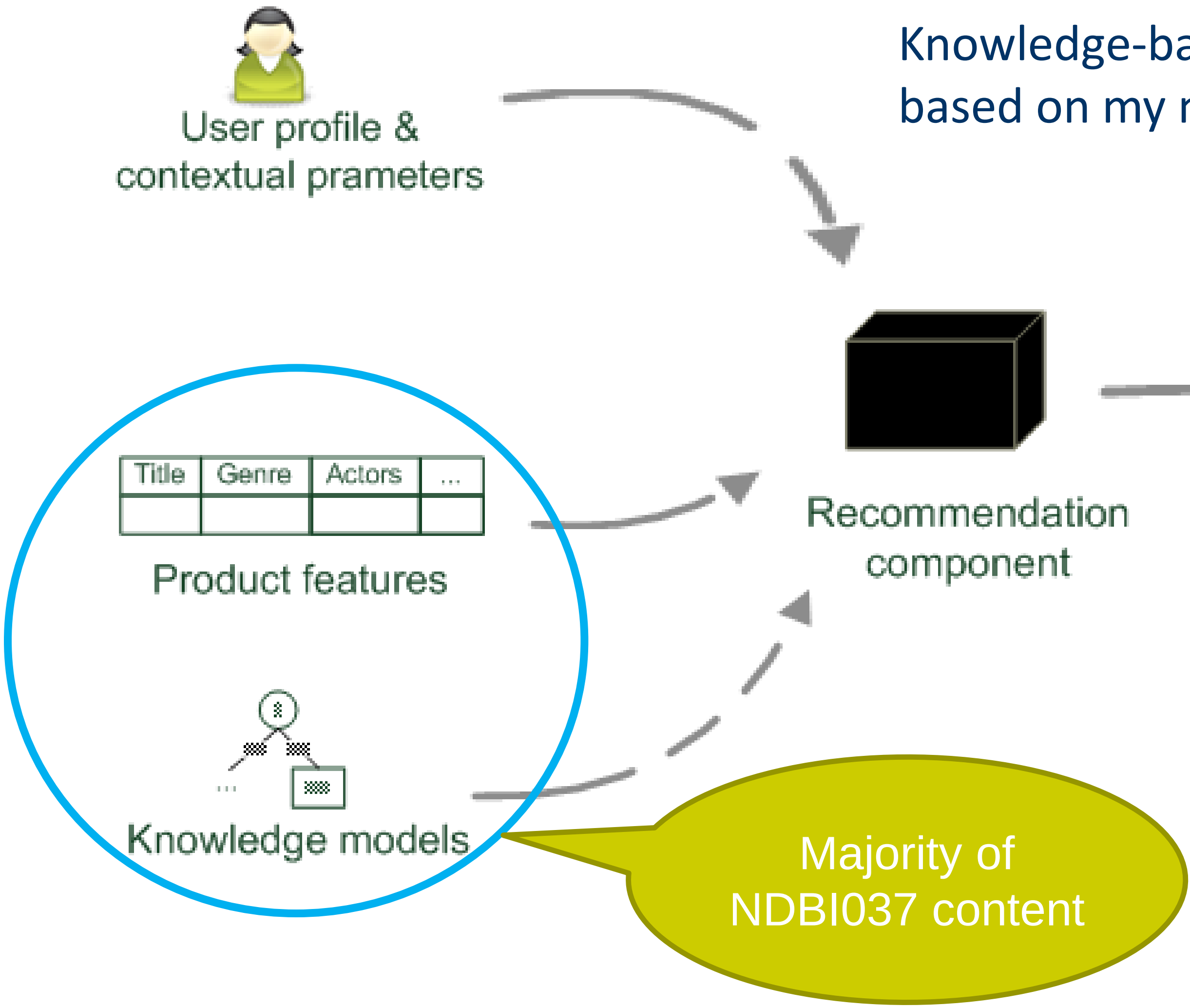
Paradigms of recommender systems

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



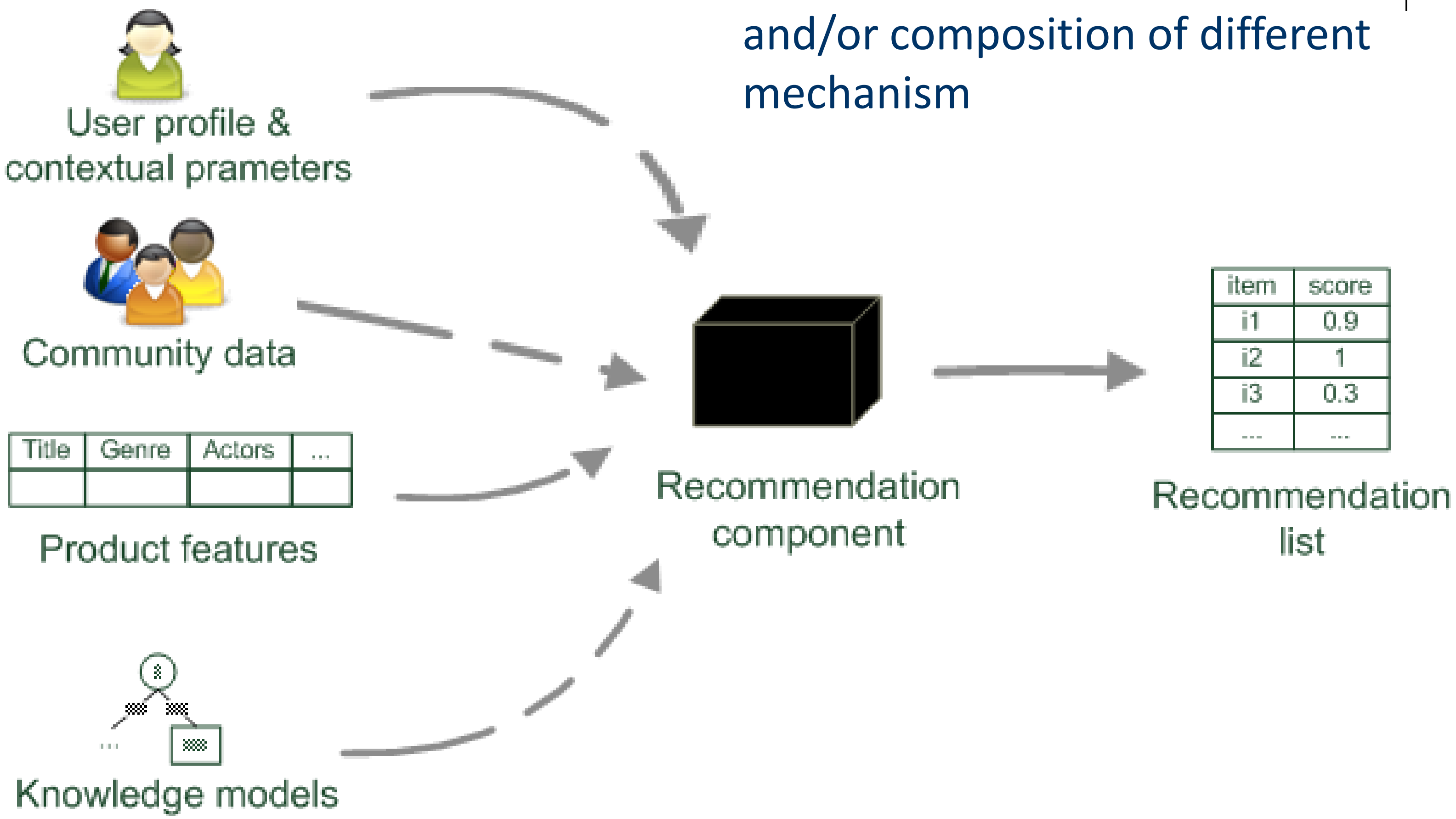
Paradigms of recommender systems

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



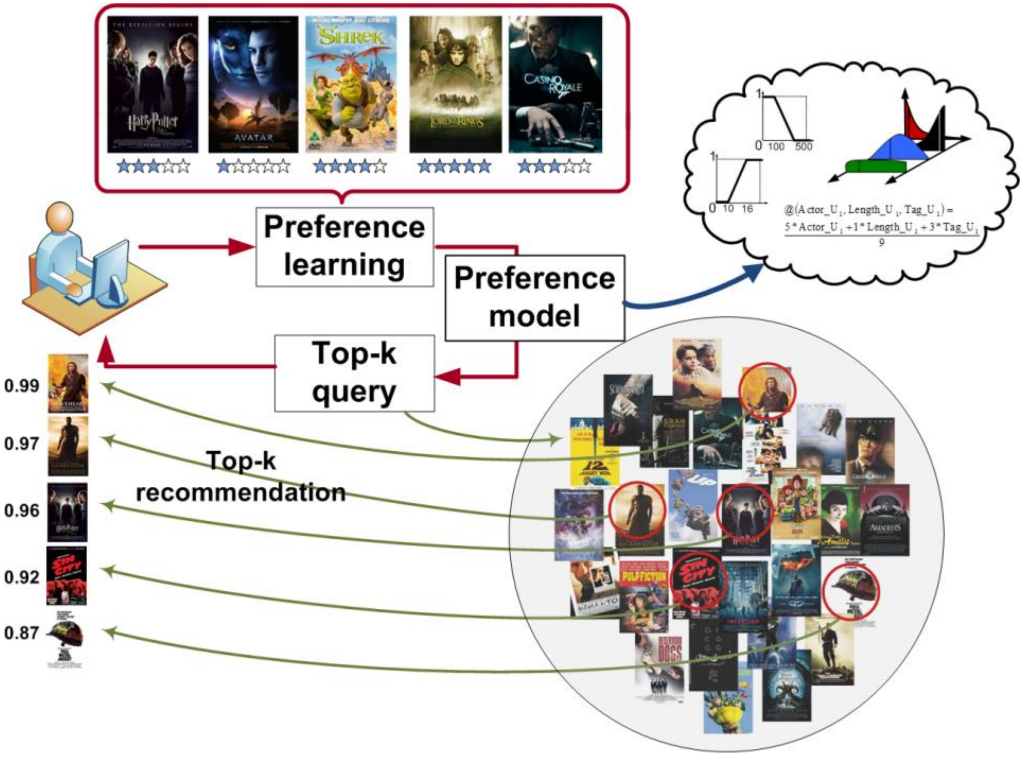
Paradigms of recommender systems

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

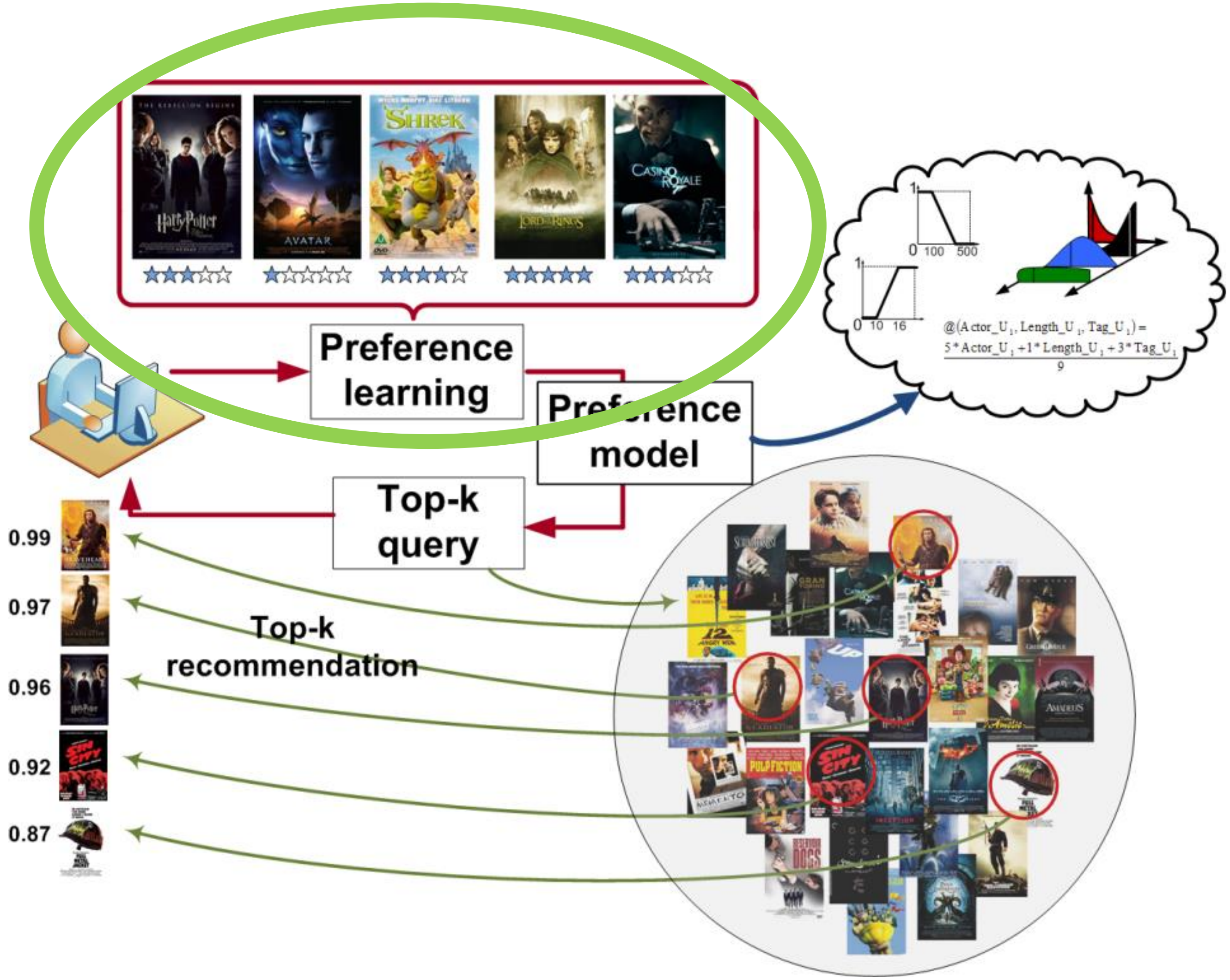


How Does Recommender Systems Work?

1. Get feedback from users (+ additional data)
2. Learn model of user's preferences
3. On demand provide recommendations



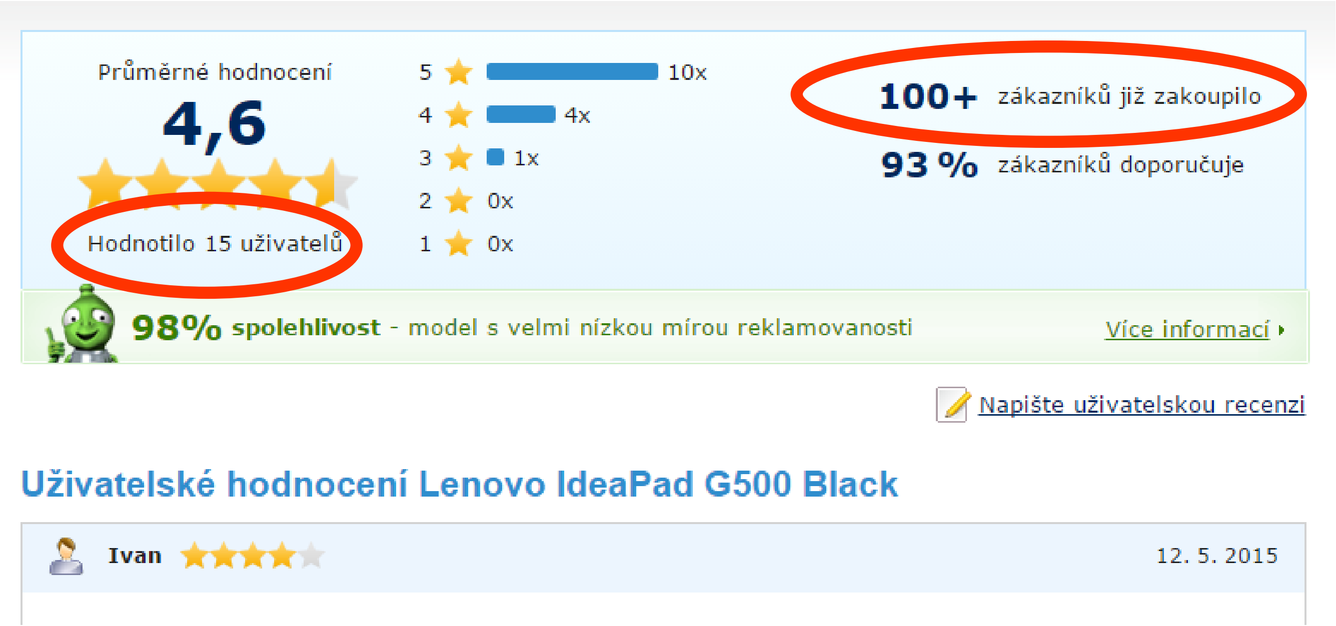
User Feedback



Zpětná vazba

- Explicit feedback *(uživatelské hodnocení)* 
  - − Ne od všech uživatelů, ne ke každému objektu, vyžaduje úsilí uživatele
  - − Problém v e-commerce (chybí motivace a nutnost nejprve objekt vyzkoušet)
  - \+ Poměrně přesně popisuje preferenci uživatele
- Implicit feedback *(sledování chování uživatele)*
  - − Šum, těžko se interpretuje
  - − Není jasné co všechno sledovat
  - \+ **Můžeme mít feedbacku „kolik chceme“**

Zpětná vazba



Implicit Feedback v E-commerce

Kategorie

- Seznam / výběr objektů
- **Parametry kategorie, vyhledávané slovní spojení,…**
- Evaluace jednotlivých objektů (visible time, mouseover,…)
- Evaluace kategorie jako celku

Detail objektu

- **Počet zobrazení**
- Akce myši, scrolování
- **Čas na stránce**
- **Nákup**
- (bookmark, tisk, eyetracking, kopírování textu …)

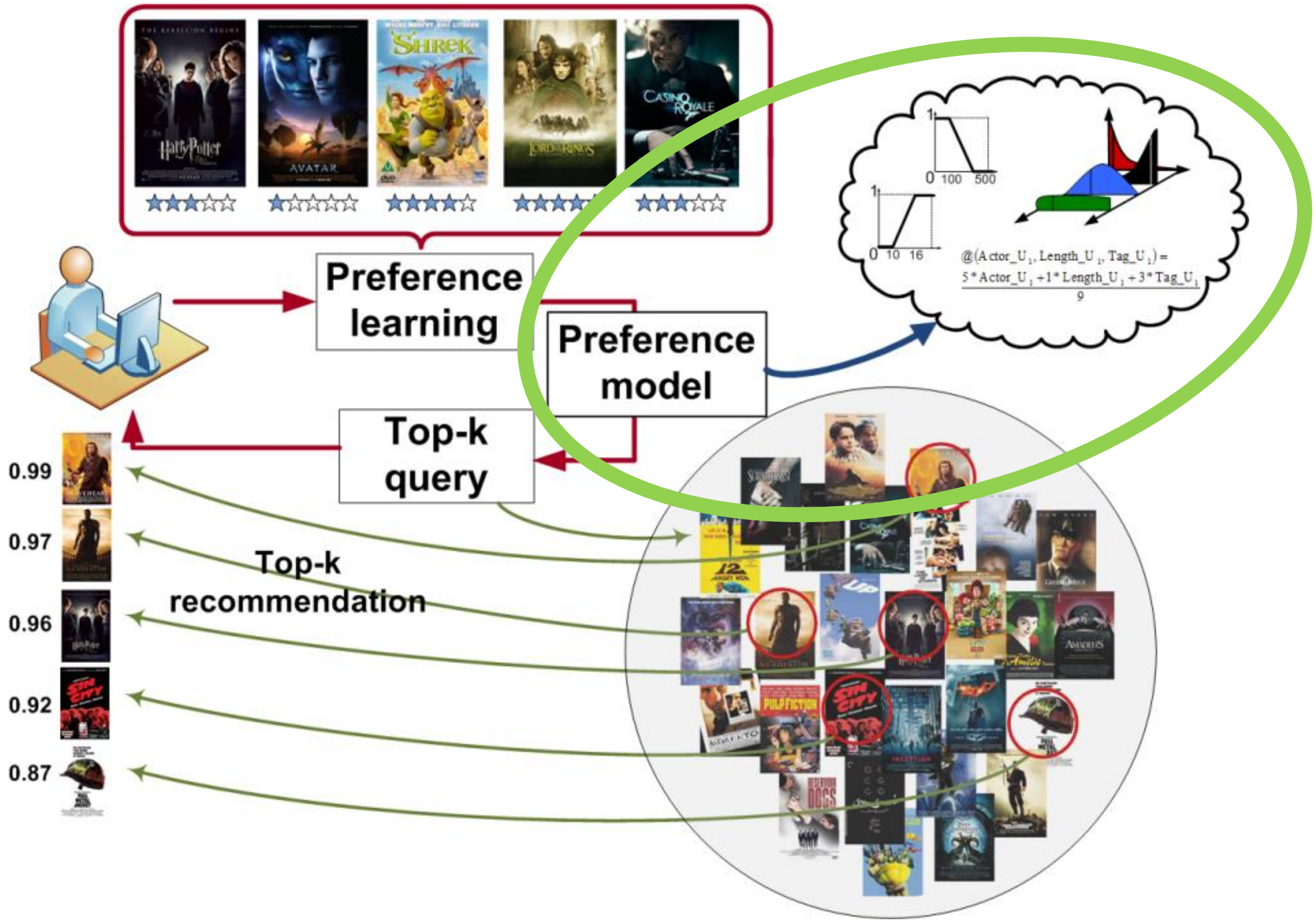
Jak ze zpětné vazby získám preferenci?

- Explicitní feedback je relativně přímočarý
  - Normalizace hodnocení mezi uživateli

- Implicitní může být větší problém
  - Time on page: 120 sec
  - Number of visits: 2
  - Scrolling distance: 500px
  - Mouse distance: 150px
  - …

Recommending Algorithms

Doporučovací algoritmy



DISCLAIMER

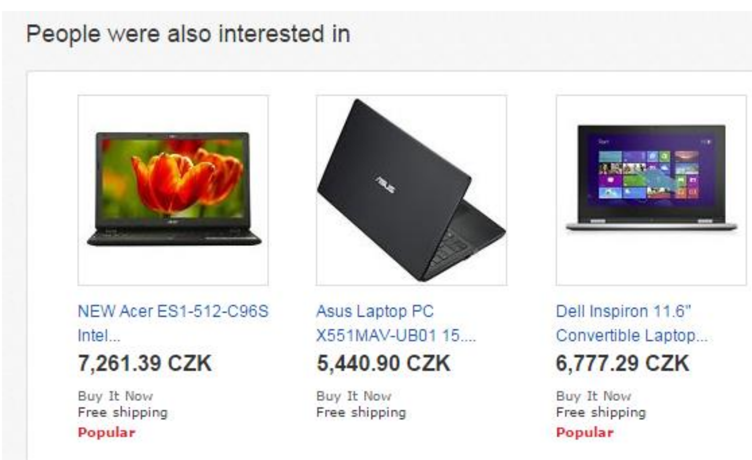
- Whenever there is a ***human factor*** (client, user) in evaluation

  …and this THE case of RecSys
  - You cannot say, what is right or wrong
    - You may just evaluate and iterpret usage statistics or ask users for their opinion
  - *Algorithms materialize some hypothesis about user's behavior*
    - *Sometimes work, sometimes not*
    - *Never cover all aspects of each user*

Recommending Algorithms

No personalization

- *„Most popular“*
- *„People who bought this also bought that“*
- *„Similar objects to this one“*
- *…*



Personalized

- Similarity of objects *„Content-based“*
- Similarity of behavior *„Collaborative filtering“*
- Hybrid, context-based
- Session based…

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**

Collaborative-filtering algorithms

- K-Nearest Neighbors
  - Simplest solution, 90' way of thinking, performance problems, easy implementation
- Matrix factorization (latent factor models)
  - SOTA before deep learning emerged
- Graph based algorithms
  - Social networks
- Market Basket Analysis
- …

User-based nearest-neighbor collaborative filtering

- You can write it after reading 3-sentence description

- Get most similar users to the current one.
- Aggregate their feedback on not-yet-visited items.
- Recommend items with best aggregated feedack.

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 / **matrix factorizations** are examples for model-based approaches

Matrix Factorization

- Matrix of ratings (implicit feedback) of users on items
- Decompose this matrix into latent factors of users and items
  - Decomposition should provide similar results on known data

$$\mathbf{R} \approx \mathbf{U}\mathbf{O}^T = \underbrace{\begin{bmatrix} \mu_1^T \\ \mu_2^T \\ \vdots \end{bmatrix}}_{n \times f} \times \underbrace{\begin{bmatrix} \sigma_1 & \sigma_2 & \ldots \end{bmatrix}}_{f \times m}$$

  - Define error of this representation
  - Learn to minimize this error

- http://www2.research.att.com/~volinsky/papers/ieeecomputer.pdf

Faktorizace Matic

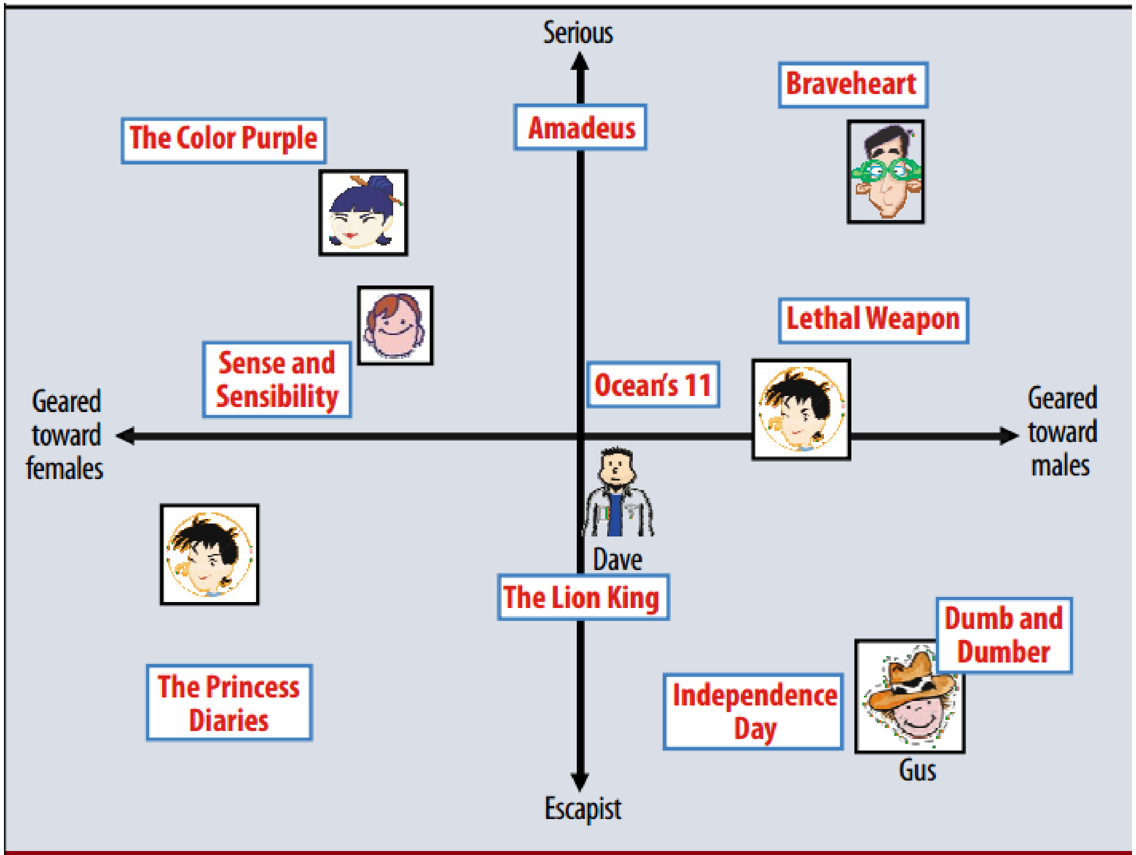


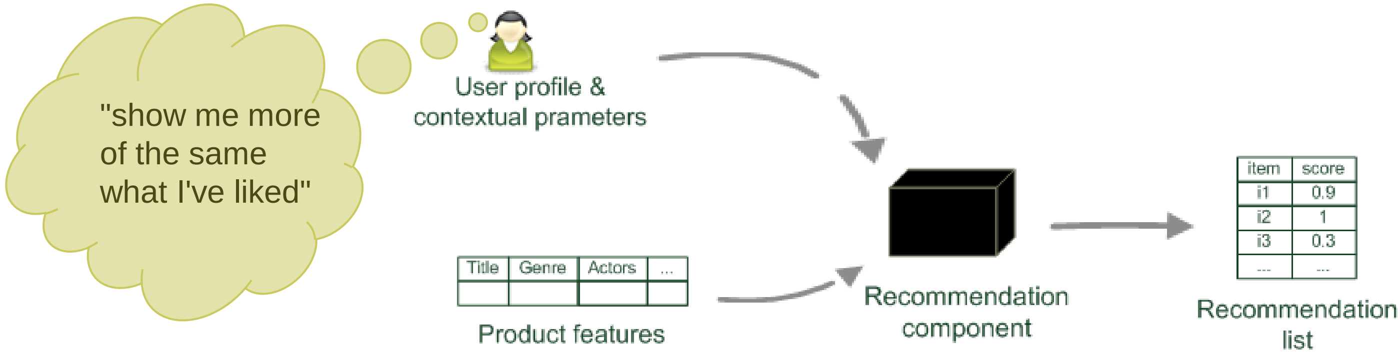
Figure 2. A simplified illustration of the latent factor approach, which characterizes both users and movies using two axes—male versus female and serious versus escapist.

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, hybrid 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)

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



Content-based Algorithms

- Vector Space Model
  - Shared space of users (a.k.a. query) and objects, define similarity w.r.t. content-based attributes
  - Often TF-IDF weighting
- Common machine learning
  - Decision trees / forests / GBT
  - Graph-based algorithms (Linked Data)
  - SVM…

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
    Unstructured: free-text description.

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

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)
  - Or use and combine multiple metrics

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

Term-Frequency - Inverse Document Frequency ($TF - IDF$)

- Simple keyword representation has its problems
  - in particular when automatically extracted as
    - 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 in multi-dimensional Euclidian space
    - 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 II

- Given a keyword i and a document j
- $TF(i,j)$
  - term frequency of keyword i in document j
- $IDF(i)$
  - inverse document frequency calculated as $\boldsymbol{IDF(i) = log\frac{N}{n(i)}}$
    - N : number of all recommendable documents
    - $n(i)$: number of documents from N in which keyword i appears
- $TF - IDF$
  - is calculated as: $\boldsymbol{TF\text{-}IDF(i,j) = TF(i,j) * IDF(i)}$

Cosine similarity

- Usual similarity metric to compare vectors: Cosine similarity (angle)
  - Cosine similarity is calculated based on the angle between the vectors
    - $sim(\vec{a}, \vec{b}) = \frac{\vec{a} \cdot \vec{b}}{|\vec{a}| * |\vec{b}|}$
- Adjusted cosine similarity
  - take average user ratings into account ($\bar{r}_u$), transform the original ratings
  - U: set of users who have rated both items a and b
  - $sim(\vec{a}, \vec{b}) = \frac{\sum_{u \in U}(r_{u,a} - \bar{r}_u)(r_{u,b} - \bar{r}_u)}{\sqrt{\sum_{u \in U}(r_{u,a} - \bar{r}_u)^2}\sqrt{\sum_{u \in U}(r_{u,b} - \bar{r}_u)^2}}$

How to know what to use?

- Experiment as much as you can
  - *If You want to double your success rate, you should double your failure rate.*

Experiments

Online

- Production servers
- Hard to repeat
- Expensive (time + money)
- Selected methods
- Real metrics
- GUI changes etc. can be also evaluated

Offline

- Data-based simulation
- Easily repeatable
- Fast and cheap
- Artificial metrics (RMSE, MAE, diversity…)
- Causality problems; Only algorithms can be compared

Success in offline do not imply success in online…

…however the lack of success usually holds.

Conclusions

- Recommender systems slowly became standard in web applications.
- There is always problem with insufficient data
  - Tradeoff between complexity and train-ability
  - Multiple pathways to explore
  - Domain dependent best practices
- Basic algorithms are easy to handle (devil in details)
- Important research topic (Bc, Mgr, PhD thesis etc.)

Co jsem vynechal:

- Deep learning
- Jak zobrazovat doporučení
- Jak vysvětlovat doporučení (user trust)
- Vývoj preferencí uživatele v čase, dlouhodobé / krátkodobé preference
- Kontext (místo, čas, nabídka,…)

Dotazy?