Recommender Systems: Content-based, Knowledge-based, Hybrid

Radek Pelánek
Today

- lecture, basic principles:
  - content-based
  - knowledge-based
  - hybrid, choice of approach, . . .
  - critiquing, explanations, . . .

- illustrative examples from various domains: videos, recipes, products, finance, restaurants, ...

- highlighting wider context / connections:
  - machine learning, natural language processing, constraint satisfaction problems, automata, ...
at the end:

- brief presentation of your projects
- application of covered notions to projects
  ⇒ *make notes during lecture*
collaborative filtering: “recommend items that similar users liked”

content based: “recommend items that are similar to those the user liked in the past”
Content-based Recommendations

we need explicit (cf latent factors in CF):
- information about items (e.g., genre, author)
- user profile (preferences)
Typical Example: Recipes

- title
- brief description
- list of ingredients
- cooking instructions
- tags (cuisine type, dietary restrictions)
- numerical attributes (cooking time, estimated price)
How to recommend similar items? How to measure similarity of two recipes?

What could be “user profile”?

How would we recommend recipes to a given user?
Architecture of a Content-Based Recommender

Handbook of Recommender Systems
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

<table>
<thead>
<tr>
<th>Title</th>
<th>Genre</th>
<th>Author</th>
<th>Type</th>
<th>Price</th>
<th>Keywords</th>
</tr>
</thead>
<tbody>
<tr>
<td>The Lace Reader</td>
<td>Fiction, Mystery</td>
<td>Brunonia Barry</td>
<td>Hardcover</td>
<td>49.90</td>
<td>American contemporary fiction, detective, historical</td>
</tr>
<tr>
<td>Into the Fire</td>
<td>Romance, Suspense</td>
<td>Suzanne Brockmann</td>
<td>Hardcover</td>
<td>45.90</td>
<td>American fiction, murder, neo-Nazism</td>
</tr>
</tbody>
</table>

- Unstructured: free-text description.
Content: Multimedia

- manual annotation
  - songs, hundreds of features
  - Pandora, Music Genome Project
  - experts, 20-30 minutes per song
- automatic techniques – signal processing
User Profile

- explicitly specified by user
- automatically learned
  - easier than in CF – features of items are now available
Similarity: Keywords

- general similarity approach based on keywords
- two sets of keywords $A, B$ (description of two items or description of item and user)
- how to measure similarity of $A$ and $B$?
user preferences: sport, funny, comedy, learning, tricks, skateboard

- video 1: machine learning, education, visualization, math
- video 2: late night, comedy, politics
- video 3: football, goal, funny, Messi, trick, fail
sets of keywords $A, B$

- **Jaccard coefficient**: \[ \frac{|A \cap B|}{|A \cup B|} \]
- **Dice coefficient**: \[ \frac{2 \cdot |A \cap B|}{|A| + |B|} \]

many other coefficients available, see e.g. “A Survey of Binary Similarity and Distance Metrics”

Jaccard coefficient/index/score is very simple, but worth knowing under this name; it is used in many different settings
Recommendations by Nearest Neighbors

- $k$-nearest neighbors (kNN)
- predicting rating for not-yet-seen item $i$:
  - find $k$ most similar items, already rated
  - predict rating based on these
- good for modeling short-term interest, “follow-up” stories
Melt the butter and heat the oil in a skillet over medium-high heat. Season chicken with salt and pepper, and place in the skillet. Brown on both sides. Reduce heat to medium, cover, and continue cooking 15 minutes, or until chicken juices run clear. Set aside and keep warm. Stir cream into the pan, scraping up brown bits. Mix in mustard and tarragon. Cook and stir 5 minutes, or until thickened. Return chicken to skillet to coat with sauce. Drizzle chicken with remaining sauce to serve.
Similarity: Text Descriptions

Examples: product description, recipe instructions, movie plot

basic approach: bag-of-words representation (words + counts of occurrences)

limitations?
7 and
4 the
4 chicken
4 to
3 heat
3 in
3 skillet
3 with
2 brown
2 minutes
2 or
2 until
2 stir
2 sauce
1 melt
1 butter
disadvantages of simple counts:
- importance of words ("skillet" vs "with")
- length of documents

TF-IDF – standard technique in information retrieval
- Term Frequency – how often term appears in a particular document (normalized by document length)
- Inverse Document Frequency – how often term appears in all documents
Term Frequency – Inverse Document Frequency

keyword (term) \( t \), document \( d \)

- \( TF(t, d) = \frac{\text{frequency of } t \text{ in } d}{\text{maximal frequency of a term in } d} \)
- \( IDF(t) = \log(\frac{N}{n_t}) \)
  - \( N \) – number of all documents
  - \( n_t \) – number of documents containing \( t \)
- \( TFIDF(t, d) = TF(t, d) \cdot IDF(t) \)

note: there are multiple specific definitions of TF-IDF; they all express the same basic idea, but in different manners

Similarity

similarity between user and item profiles (or two item profiles):

- vector of keywords and their TF-IDF values
- cosine similarity – angle between vectors
  \[ \text{sim}(\mathbf{a}, \mathbf{b}) = \frac{\mathbf{a} \cdot \mathbf{b}}{||\mathbf{a}|| ||\mathbf{b}||} \]
- (adjusted) cosine similarity
  - normalization by subtracting average values
  - closely related to Pearson correlation coefficient
Improvements

using all words $\rightarrow$ long, sparse vectors

typical processing steps:

- common words, stop words (e.g., “a”, “the”, “on”)
- lemmatization, stemming (e.g., “went” $\rightarrow$ “go”, “university” $\rightarrow$ “univers”)
- cut-offs (e.g., $n$ most informative words)
- phrases (e.g., “United Nations”, “New York”)

wider context: natural language processing techniques
Limitations of Bag-of-words

- semantic meaning unknown
- example – use of words in negative context

steakhouse description: “there is nothing on the menu that a vegetarian would like...” ⇒ keyword “vegetarian” ⇒ recommended to vegetarians
Incorporating Domain Knowledge

user preferences: sport, funny, comedy, learning, tricks, skateboard

- video 1: machine learning, education, visualization, math
- video 2: late night, comedy, politics
- video 3: footbal, goal, funny, Messi, trick, fail
Ontologies, Taxonomies, Folksomies

- ontology – formal definition of entities and their relations
- taxonomy – tree, hierarchy (example: news, sport, football, football world cup)
- folksonomy (folk + taxonomy) – collaborative tagging, tag clouds
Large Language Models

- methods based on neural networks, large language models (LLMs), BERT, GPT, word/sentence embeddings, ...
- potentially powerful, can capture semantics
- however: “black-box” methods, hard interpretability, pre-trained method may not be suitable for a specific task
- field with intensive developments
Recommendation as Classification

- classification problem: features $\rightarrow$ like/dislike (rating)
- use of general machine learning techniques
  - probabilistic methods – Naive Bayes
  - linear classifiers
  - decision trees
  - neural networks
  - ...

wider context: machine learning techniques
Content-Based Recommendations: Advantages

- **user independence** – does not depend on other users
- **new items** can be easily incorporated (no cold start)
- **transparency** – understandable, provides explanations (at least with basic methods)
Content-Based Recommendations: Limitations

- limited content analysis
  - content may not be automatically extractable (multimedia)
  - missing domain knowledge
  - keywords may not be sufficient
- overspecialization – “more of the same”, too similar items
- new user – ratings or information about user has to be collected
Content-Based vs Collaborative Filtering

- paper “Recommending new movies: even a few ratings are more valuable than metadata” (context: Netflix)
- our experience in educational domain – difficulty rating (Sokoban, countries)
Knowledge-based Recommendations

application domains:
- expensive items, not frequently purchased, few ratings (car, house)
- time span important (technological products)
- explicit requirements of user (vacation)

- collaborative filtering unusable – not enough data
- content based – “similarity” not sufficient
Knowledge-based Recommendations

- constraint-based
  - explicitly defined conditions
- case-based
  - similarity to specified requirements

“conversational” recommendations
Constraint-Based Recommendations – Example

<table>
<thead>
<tr>
<th>id</th>
<th>price(€)</th>
<th>mpx</th>
<th>opt-zoom</th>
<th>LCD-size</th>
<th>movies</th>
<th>sound</th>
<th>waterproof</th>
</tr>
</thead>
<tbody>
<tr>
<td>P₁</td>
<td>148</td>
<td>8.0</td>
<td>4×</td>
<td>2.5</td>
<td>no</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>P₂</td>
<td>182</td>
<td>8.0</td>
<td>5×</td>
<td>2.7</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>P₃</td>
<td>189</td>
<td>8.0</td>
<td>10×</td>
<td>2.5</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>P₄</td>
<td>196</td>
<td>10.0</td>
<td>12×</td>
<td>2.7</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>P₅</td>
<td>151</td>
<td>7.1</td>
<td>3×</td>
<td>3.0</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>P₆</td>
<td>199</td>
<td>9.0</td>
<td>3×</td>
<td>3.0</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>P₇</td>
<td>259</td>
<td>10.0</td>
<td>3×</td>
<td>3.0</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>P₈</td>
<td>278</td>
<td>9.1</td>
<td>10×</td>
<td>3.0</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
</tbody>
</table>

Recommender Systems: An Introduction (slides)
Constraint Satisfaction Problem

- $V$ is a set of variables
- $D$ is a set of finite domains of these variables
- $C$ is a set of constraints

Typical problems: logic puzzles (Sudoku, N-queen), scheduling
CSP: $N$-queens

Problem: place $N$ queens on an $N \times N$ chess-board, no two queens threaten each other

- $V$ – $N$ variables (locations of queens)
- $D$ – each domain is $\{1, \ldots, N\}$
- $C$ – threatening
CSP Algorithms

- basic algorithm – backtracking
- heuristics
  - preference for some branches
  - pruning
  - ... many others
CSP Example: N-queens Problem
Recommender Knowledge Base

- customer properties \( V_C \)
- product properties \( V_{PROD} \)
- constraints \( C_R \) (on customer properties)
- filter conditions \( C_F \) – relationship between customer and product
- products \( C_{PROD} \) – possible instantiations
$V_C = \{ k_{cl}: [\text{expert, average, beginner}] \} \quad \text{/* level of expertise */}$
$\text{wr}_c: [\text{low, medium, high}] \quad \text{/* willingness to take risks */}$
$\text{id}_c: [\text{shortterm, mediumterm, longterm}] \quad \text{/* duration of investment */}$
$\text{aw}_c: [\text{yes, no}] \quad \text{/* advisory wanted? */}$
$\text{ds}_c: [\text{savings, bonds, stockfunds, singlshares}] \quad \text{/* direct product search */}$
$\text{sl}_c: [\text{savings, bonds}] \quad \text{/* type of low-risk investment */}$
$\text{av}_c: [\text{yes, no}] \quad \text{/* availability of funds */}$
$\text{sh}_c: [\text{stockfunds, singlshares}] \quad \text{/* type of high-risk investment */}$

$V_{PROD} = \{ \text{name}_p: [\text{text}] \} \quad \text{/* name of the product */}$
$\text{er}_p: [1..40] \quad \text{/* expected return rate */}$
$\text{ri}_p: [\text{low, medium, high}] \quad \text{/* risk level */}$
$\text{mniv}_p: [1..14] \quad \text{/* minimum investment period of product in years */}$
$\text{inst}_p: [\text{text}] \quad \text{/* financial institute */}$
\[ C_R = \{ CR_1: wr_c = \text{high} \rightarrow id_c \neq \text{shortterm}, \\
CR_2: kl_c = \text{beginner} \rightarrow wr_c \neq \text{high} \} \]

\[ C_F = \{ CF_1: id_c = \text{shortterm} \rightarrow mniv_p < 3, \\
CF_2: id_c = \text{mediumterm} \rightarrow mniv_p \geq 3 \land mniv_p < 6, \\
CF_3: id_c = \text{longterm} \rightarrow mniv_p \geq 6, \\
CF_4: wr_c = \text{low} \rightarrow ri_p = \text{low}, \\
CF_5: wr_c = \text{medium} \rightarrow ri_p = \text{low} \lor ri_p = \text{medium}, \\
CF_6: wr_c = \text{high} \rightarrow ri_p = \text{low} \lor ri_p = \text{medium} \lor ri_p = \text{high}, \\
CF_7: kl_c = \text{beginner} \rightarrow ri_p \neq \text{high}, \\
CF_8: sl_c = \text{savings} \rightarrow name_p = \text{savings}, \\
CF_9: sl_c = \text{bonds} \rightarrow name_p = \text{bonds} \} \]

\[ C_{PROD} = \{ C_{PROD_1}: name_p = \text{savings} \land er_p = 3 \land ri_p = \text{low} \land mniv_p = 1 \land inst_p = A; \\
C_{PROD_2}: name_p = \text{bonds} \land er_p = 5 \land ri_p = \text{medium} \land mniv_p = 5 \land inst_p = B; \\
C_{PROD_3}: name_p = \text{equity} \land er_p = 9 \land ri_p = \text{high} \land mniv_p = 10 \land inst_p = B \} \]
Development of Knowledge Bases

- difficult, expensive
- specialized graphical tools
- methodology (rapid prototyping, detection of faulty constraints, ...)
Unsatisfied Requirements

no solution to provided constraints
  • we want to provide user at least something
  • constraint relaxation
  • proposing “repairs”
  • minimal set of requirements to be changed
User Guidance

requirements elicitation process
- session independent user profile
- static fill-out forms
- conversational dialogs
User Guidance

Recommender Systems Handbook; Developing Constraint-based Recommenders
Fig. 6.4: Interactive and personalized preference elicitation example. Customers specify their preferences by answering questions.
Critiquing

Find your Favourite restaurant

In Vienna you chose:
Biergasthof
+43 1 123 123 123
Mariahilferstrasse 123,
1010 Wien
local food, central in the city, weekend brunch, room with a view,
parents for beer, seasonal dishes, group bookings, open all day

For Graz we recommend:
Brauhof
+43 316 45 45 45
Brauhofstrasse 45,
8023 Graz
local food, own beer, weekend lunch, open all day, private function room,
famous for beer, seasonal dishes, group bookings, good transport connection

Recommender Systems: An Introduction (slides)
Critiquing

Critique on price

threshold: items with a lower price than the entry item are considered further

new most similar item
Critiquing: Example

A Visual Interface for Critiquing-based Recommender Systems
Fig. 3 Critiquing support to guide users to critique the current example product for comparing it with the other tradeoff alternatives

Critiquing-based recommenders: survey and emerging trends
Critiquing: Example

Fig. 5 The Dynamic Critiquing interface with system suggested compound critiques for users to select (McCarthy et al. 2005c)
Knowledge-based Recommendations: Limitations

- cost of knowledge acquisition (consider your project proposals)
- accuracy of models
- independence assumption for preferences
Hybrid Methods

collaborative filtering: "what is popular among my peers"
content-based: "more of the same"
knowledge-based: "what fits my needs"

- each has advantages and disadvantages
- hybridization – combine more techniques, avoid some shortcomings
- simple example: CF with content-based (or simple “popularity recommendation”) to overcome “cold start problem”
Hybridization Designs

- monolithic design, combining different features
- parallel use of several systems, weighting/voting
- pipelined invocation of different systems
Types of Recommender Systems

- non-personalized
- demographic
- collaborative filtering
- content-based
- knowledge-based
- hybrid

what to apply when?
Taxonomy of Knowledge Sources

Knowledge source

- Social
  - Context
  - Opinions
  - Behavior
  - Demographics
  - Ratings
  - Tags
  - Reviews

- Individual
  - Opinions
  - Behavior
  - Demographics
  - Requirements
  - Query
  - Constraints
  - Preferences
  - Context

- Content
  - Item Features
  - Domain Knowledge
  - Contextual Knowledge
  - Means-ends
  - Feature
  - Ontology
  - Domain
  - Constraints

Matching Recommendation Technologies and Domains
Knowledge Sources and Recommendation Types

Matching Recommendation Technologies and Domains
### Sample Domains for Recommendation

<table>
<thead>
<tr>
<th>Domain</th>
<th>Risk</th>
<th>Churn</th>
<th>Heterogeneity</th>
<th>Preferences</th>
<th>Interaction Style</th>
<th>Scrutability</th>
<th>Examples</th>
<th>Technology</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>ACR news[45] and [38] Google news[16]</td>
<td></td>
</tr>
<tr>
<td>E-commerce</td>
<td>Low</td>
<td>High</td>
<td>High</td>
<td>Stable</td>
<td>Implicit</td>
<td>Not required</td>
<td>Amazon.com eBay</td>
<td>Collaborative-Filtering</td>
</tr>
<tr>
<td>Web Page Recommender</td>
<td>Low</td>
<td>High</td>
<td>High</td>
<td>Unstable</td>
<td>Implicit</td>
<td>Not required</td>
<td>[9, 36, 4]</td>
<td>Collaborative-Filtering Hybrid</td>
</tr>
<tr>
<td>Movie</td>
<td>Low</td>
<td>Low</td>
<td>Low</td>
<td>Stable</td>
<td>Implicit</td>
<td>Not required</td>
<td>Netflix[50], MovieLens[21]</td>
<td>Collaborative-Filtering</td>
</tr>
<tr>
<td>Music</td>
<td>Low</td>
<td>Low</td>
<td>Low</td>
<td>Stable?</td>
<td>Implicit</td>
<td>Not required</td>
<td>Pandora and [24, 28, 14]</td>
<td>Content-based Hybrid</td>
</tr>
<tr>
<td>Financial-services</td>
<td>High</td>
<td>Low</td>
<td>Low</td>
<td>Stable</td>
<td>Explicit</td>
<td>Required</td>
<td>Koba4MS[17], FSAervisor[19, 65]</td>
<td>Knowledge-Based</td>
</tr>
<tr>
<td>Life-insurance</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Software Engineering</td>
<td>Low</td>
<td>Low</td>
<td>Low</td>
<td>Stable</td>
<td>Explicit /Implicit</td>
<td>Required</td>
<td>[13] and [29]</td>
<td>Hybrid and Content-based</td>
</tr>
<tr>
<td>Tourism</td>
<td>High</td>
<td>Low</td>
<td>Low</td>
<td>Unstable</td>
<td>Explicit</td>
<td>Required</td>
<td>Travel Recommender [55] [37]</td>
<td>Content-based Knowledge-based</td>
</tr>
<tr>
<td>Job search Recruiting</td>
<td>High</td>
<td>Low</td>
<td>Low</td>
<td>Stable</td>
<td>Explicit</td>
<td>Required</td>
<td>CASPER [35] and [39]</td>
<td>Content-based</td>
</tr>
<tr>
<td>Real Estate</td>
<td>High</td>
<td>Low</td>
<td>Low</td>
<td>Stable</td>
<td>Explicit</td>
<td>Required</td>
<td>RentMe, FlatFinder[67] and [73]</td>
<td>Knowledge-based</td>
</tr>
</tbody>
</table>

Matching Recommendation Technologies and Domains
Explanations of Recommendations

- recommendations: selection (ranked list) of items
- explanations: (some) reasons for the choice
Goals of Providing Explanations

Why explanations?
Goals of Providing Explanations

Why explanations?

- transparency, trustworthiness, validity, satisfaction (users are more likely to use the system)
- persuasiveness (users are more likely to follow recommendations)
- effectiveness, efficiency (users can make better/faster decisions)
- education (users understand better the behaviour of the system, may use it in better ways)
Examples of Explanations

- knowledge-based recommenders
  - “Because you, as a customer, told us that simple handling of car is important to you, we included a special sensor system in our offer that will help you park your car easily.”
- algorithms based on CSP representation
Examples of Explanations

- knowledge-based recommenders
  - “Because you, as a customer, told us that simple handling of car is important to you, we included a special sensor system in our offer that will help you park your car easily.”
- algorithms based on CSP representation
- recommendations based on item-similarity
  - “Because you watched X we recommend Y”
Explanations – Collaborative Filtering

Your Neighbors' Ratings for this Movie

<table>
<thead>
<tr>
<th>Rating</th>
<th>Number of Neighbors</th>
</tr>
</thead>
<tbody>
<tr>
<td>★</td>
<td>1</td>
</tr>
<tr>
<td>★★</td>
<td>2</td>
</tr>
<tr>
<td>★★★</td>
<td>7</td>
</tr>
<tr>
<td>★★★★</td>
<td>14</td>
</tr>
<tr>
<td>★★★★★</td>
<td>9</td>
</tr>
</tbody>
</table>

Your Neighbors' Ratings for this Movie

Explaining Collaborative Filtering Recommendations, Herlocker, Konstan, Riedl
### Ratings for *Sixth Sense, The* (1999) by your Neighbors

<table>
<thead>
<tr>
<th>Rating</th>
<th>Your Neighbors' Ratings</th>
</tr>
</thead>
<tbody>
<tr>
<td>Must See</td>
<td></td>
</tr>
<tr>
<td>Will</td>
<td></td>
</tr>
<tr>
<td>Enjoy It</td>
<td></td>
</tr>
<tr>
<td>It's OK</td>
<td></td>
</tr>
<tr>
<td>Fairly</td>
<td></td>
</tr>
<tr>
<td>Bad</td>
<td></td>
</tr>
<tr>
<td>Awful</td>
<td></td>
</tr>
</tbody>
</table>

- **Strong Neighbors** (very similar)
- **Weak Neighbors**

Click on a bar to see that neighbor's profile!

**Figure 4.** A screen explaining the recommendation for the movie “The Sixth Sense.” Each bar represents a rating of a neighbor. Upwardly trending bars are positive ratings, while downward trending ones are negative. The x-axis represents similarity to the user.
# Explanations – Comparison

<table>
<thead>
<tr>
<th>#</th>
<th>Explanation</th>
<th>N</th>
<th>Mean Response</th>
<th>Std Dev</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Histogram with grouping</td>
<td>76</td>
<td>5.25</td>
<td>1.29</td>
</tr>
<tr>
<td>2</td>
<td>Past performance</td>
<td>77</td>
<td>5.19</td>
<td>1.16</td>
</tr>
<tr>
<td>3</td>
<td>Neighbor ratings histogram</td>
<td>78</td>
<td>5.09</td>
<td>1.22</td>
</tr>
<tr>
<td>4</td>
<td>Table of neighbors ratings</td>
<td>78</td>
<td>4.97</td>
<td>1.29</td>
</tr>
<tr>
<td>5</td>
<td>Similarity to other movies rated</td>
<td>77</td>
<td>4.97</td>
<td>1.50</td>
</tr>
<tr>
<td>6</td>
<td>Favorite actor or actress</td>
<td>76</td>
<td>4.92</td>
<td>1.73</td>
</tr>
<tr>
<td>7</td>
<td>MovieLens percent confidence in prediction</td>
<td>77</td>
<td>4.71</td>
<td>1.02</td>
</tr>
<tr>
<td>8</td>
<td>Won awards</td>
<td>76</td>
<td>4.67</td>
<td>1.49</td>
</tr>
<tr>
<td>9</td>
<td>Detailed process description</td>
<td>77</td>
<td>4.64</td>
<td>1.40</td>
</tr>
<tr>
<td>10</td>
<td># neighbors</td>
<td>75</td>
<td>4.60</td>
<td>1.29</td>
</tr>
<tr>
<td>11</td>
<td>No extra data – focus on system</td>
<td>75</td>
<td>4.53</td>
<td>1.20</td>
</tr>
<tr>
<td>12</td>
<td>No extra data – focus on users</td>
<td>78</td>
<td>4.51</td>
<td>1.35</td>
</tr>
<tr>
<td>13</td>
<td>MovieLens confidence in prediction</td>
<td>77</td>
<td>4.51</td>
<td>1.20</td>
</tr>
<tr>
<td>14</td>
<td>Good profile</td>
<td>77</td>
<td>4.45</td>
<td>1.53</td>
</tr>
<tr>
<td>15</td>
<td>Overall percent rated 4+</td>
<td>75</td>
<td>4.37</td>
<td>1.26</td>
</tr>
<tr>
<td>16</td>
<td>Complex graph: count, ratings, similarity</td>
<td>74</td>
<td>4.36</td>
<td>1.47</td>
</tr>
<tr>
<td>17</td>
<td>Recommended by movie critics</td>
<td>76</td>
<td>4.21</td>
<td>1.47</td>
</tr>
<tr>
<td>18</td>
<td>Rating and %agreement of closest neighbor</td>
<td>77</td>
<td>4.21</td>
<td>1.20</td>
</tr>
<tr>
<td>19</td>
<td># neighbors with std. deviation</td>
<td>78</td>
<td>4.19</td>
<td>1.45</td>
</tr>
<tr>
<td>20</td>
<td># neighbors with avg correlation</td>
<td>76</td>
<td>4.08</td>
<td>1.46</td>
</tr>
<tr>
<td>21</td>
<td>Overall average rating</td>
<td>77</td>
<td>3.94</td>
<td>1.22</td>
</tr>
</tbody>
</table>

Table 1. Mean response of users to each explanation interface, based on a scale of one to seven. Explanations 11 and 12 represent the base case of no additional information. Shaded rows indicate explanations with a mean response significantly different from the base cases (two-tailed $\alpha = 0.05$).
Moment of Recommendation

- front page, dashboard
- follow-up
- sidebar
- on demand

recommendation placements may differ in their requirements
Your Projects: Questions

- What is the purpose / use case? What is the “business model”?
- What will you recommend? In what situation?
- A new system or extention of an existing one?
- What data you have?
  - items
  - user preferences; explicit/implicit ratings?
- Which techniques are relevant/suitable for you project? Collaborative filtering? Content-based? Knowledge-based? Combination?
- Are the following notions relevant: taxonomy, critiquing, explanations?
Projects

- board games
- travel
- movies I
- movies II: Movielens predictions
- books (?)
- volunteering
- MOOC