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

Radek Pelánek

2014
Today

- lecture, basic principles:
  - content-based
  - knowledge-based
- discussion – projects
  - brief presentation of your projects
  - application of notions to projects
Content-based vs Collaborative Filtering

- 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)
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, http://www.pandora.com
  - Music Genome Project
  - experts, 20-30 minutes per song

- automatic techniques – signal processing
User Profile

- explicitly specified by user
- automatically learned
Similarity: Keywords

sets of keywords $A$, $B$

- Dice coefficient: $\frac{2 \cdot |A \cap B|}{|A| + |B|}$
- Jaccard coefficient: $\frac{|A \cap B|}{|A \cup B|}$
keywords (particularly automatically extracted) – disadvantages:
  - importance of words ("course" vs "recommender")
  - length of documents

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

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

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

- all words – long, sparse vectors
  - common words, stop words (e.g., “a”, “the”, “on”)
  - stemming (e.g., “went” → “go”, “university” → “univers”)
  - cut-offs (e.g., $n$ most informative words)
  - phrases (e.g., “United Nations”)

Limitations

- 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
Similarity

- cosine similarity – angle between vectors
  \[ \text{sim}(\vec{a}, \vec{b}) = \frac{\vec{a} \cdot \vec{b}}{|\vec{a}| |\vec{b}|} \]

- (adjusted) cosine similarity
  - normalization by subtracting average values
  - closely related to Pearson correlation coefficient
Recommendations

- nearest neighbors
- Rocchio’s relevance feedback method (interactivity)
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
Other Methods

- probabilistic methods – Naive Bayes
- linear classifiers
Limitations of Content-Based Recommendations

- 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
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
- time span important (e.g., technological products)
- explicit requirements of user

- 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>mpix</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 Example: N-queens Problem
CSP Algorithms

- basic algorithm – backtracking
- heuristics
  - preference for some branches
  - pruning
  - ... many others
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
$V_C = \{ kl_c: [\text{expert, average, beginner}] \} \quad /* \text{level of expertise} */$

$wr_c: [\text{low, medium, high}] \quad /* \text{willingness to take risks} */$

$id_c: [\text{shortterm, mediumterm, longterm}] \quad /* \text{duration of investment} */$

$aw_c: [\text{yes, no}] \quad /* \text{advisory wanted?} */$

$ds_c: [\text{savings, bonds, stockfunds, singlshares}] \quad /* \text{direct product search} */$

$sl_c: [\text{savings, bonds}] \quad /* \text{type of low-risk investment} */$

$av_c: [\text{yes, no}] \quad /* \text{availability of funds} */$

$sh_c: [\text{stockfunds, singlshares}] \quad /* \text{type of high-risk investment} */ \}$

$V_{PROD} = \{ name_p: [\text{text}] \} \quad /* \text{name of the product} */$

$er_p: [1..40] \quad /* \text{expected return rate} */$

$ri_p: [\text{low, medium, high}] \quad /* \text{risk level} */$

$mnip_p: [1..14] \quad /* \text{minimum investment period of product in years} */$

$inst_p: [\text{text}] \quad /* \text{financial institute} */ \}$
$C_R = \{ CR_1: wr_c = \text{high} \rightarrow id_c \neq \text{shortterm}, \\
CR_2: kl_c = \text{beginner} \rightarrow wr_c \neq \text{high} \}$

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

$C_{PROD} = \{ CPROD_1: name_p = \text{savings} \land er_p = 3 \land ri_p = \text{low} \land mniv_p = 1 \land inst_p = A; \\
CPROD_2: name_p = \text{bonds} \land er_p = 5 \land ri_p = \text{medium} \land mniv_p = 5 \land inst_p = B; \\
CPROD_3: name_p = \text{equity} \land er_p = 9 \land ri_p = \text{high} \land mniv_p = 10 \land inst_p = B \}$
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
requirements elicitation process

- session independent user profile (e.g., social networking sites)
- static fill-out forms
- conversational dialogs
Recommender Systems Handbook; Developing Constraint-based Recommenders
Fig. 6.4: Interactive and personalized preference elicitation example. Customers specify their preferences by answering questions.
Critiquing Recommender Systems: An Introduction (slides)
Critiquing

Critique on price

threshold: items with a higher mpix than the entry item are considered further

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

new most similar item
Limitations

- cost of knowledge acquisition (consider 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
- pipelined invocation of different systems
Your Projects: Useful Questions

- How will the user interact with the system?
- Where/how will you obtain (meta)data about items?
- Do you already have some data about user preferences (ratings)?
- How will you collect ratings? (explicit/implicit)
- Which techniques are relevant/suitable for you project?
Project Topics – Short Text

- blog posts
- funny quotes
- recipes
Project Topics – Short Text

- blog posts
- funny quotes
- recipes

- content-based aspects: manual labels, TF-IDF
- ratings: implicit?, explicit?
- recipes – critiquing?, knowledge-based aspects ("quick preparation", "cheap ingredients", ...)

Project Topics – Products

- (board) games
- wine
- PC components
Project Topics – Products

- (board) games
- wine
- PC components

- content-based similarity
- knowledge-based aspects?, critiquing?
Project Topics – Educational

- vocabulary
vocabulary

frequencies of words
tag?: verbs, animals, travel, ...
rating ∼ testing?