Multiobjective Optimization in Recommender Systems using Ensemble Methods

Tomáš Řehořek

Department of Theoretical Computer Science
Faculty of Information Technology
Czech Technical University in Prague

June 27, 2013
Outline I

1. Outline
2. Introduction
3. Main Contents
4. Experiments
Recommender Systems

Recommender Systems [F. Ricci et al., 2011]

Recommender Systems are software tools and techniques providing suggestions for items to be of use to a user.

Input data typically consist of:

1. **Users** database
   - set of unique identifiers of real people using the system,

2. **Items** catalog
   - set of products (e.g., movies, CDs, books, web pages, ...) available to the users,

3. **Transactions** made by users among items
   - product purchases, ratings (number of stars), page views...
Recommender Systems

Based on her past interaction with some items, the system generates **personalized recommendations** of other items that are likely to be relevant to the given user.

Three main approaches to personalized recommendation:

- **Knowledge-based** recommendation
  - exploits specific knowledge about the domain,
  - uses set of hard-coded rules,

- **Content-based** recommendation
  - exploits meta-data about the items,
  - builds predictive model for each user,

- **Collaborative Filtering**
  - exploits similarities between users,
  - does not require any knowledge about the domain
Knowledge-based Recommendation

if camera ∧ ¬memory-card then memory card;

if camera ∧ memory-card ∧ ¬tripod then tripod;

Several disadvantages:

• suitable for small catalogs only,
• requires human expertise,
• expensive to implement and maintain
Content-based Recommendation

Predictive modeling dataset for user $X$:

<table>
<thead>
<tr>
<th>ID</th>
<th>action</th>
<th>comedy</th>
<th>drama</th>
<th>...</th>
<th>horror</th>
<th>year</th>
<th>duration [min]</th>
<th>rating</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>...</td>
<td>0</td>
<td>2001</td>
<td>127</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>...</td>
<td>0</td>
<td>1998</td>
<td>60</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>...</td>
<td>0</td>
<td>1996</td>
<td>70</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>...</td>
<td>0</td>
<td>2007</td>
<td>92</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>...</td>
<td>1</td>
<td>2010</td>
<td>102</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>...</td>
<td>0</td>
<td>2006</td>
<td>55</td>
<td>?</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>...</td>
<td>0</td>
<td>2001</td>
<td>80</td>
<td>?</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>...</td>
<td>1</td>
<td>2007</td>
<td>108</td>
<td>?</td>
</tr>
</tbody>
</table>

Disadvantages:

- requires meta-data about items (may be costly),
- computationally expensive ($10^5$ different models for $10^5$ users)
Collaborative Filtering (CF)

<table>
<thead>
<tr>
<th>UserID</th>
<th>Item1</th>
<th>Item2</th>
<th>Item3</th>
<th>Item4</th>
<th>Item5</th>
<th>Item6</th>
<th>Item7</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>?</td>
<td>3</td>
<td>?</td>
<td>?</td>
<td>4</td>
<td>1</td>
<td>3</td>
<td>...</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>?</td>
<td>4</td>
<td>?</td>
<td>5</td>
<td>?</td>
<td>1</td>
<td>...</td>
</tr>
<tr>
<td>3</td>
<td>?</td>
<td>2</td>
<td>5</td>
<td>5</td>
<td>3</td>
<td>1</td>
<td>?</td>
<td>...</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>2</td>
<td>?</td>
<td>5</td>
<td>...</td>
</tr>
<tr>
<td>5</td>
<td>3</td>
<td>?</td>
<td>?</td>
<td>5</td>
<td>4</td>
<td>?</td>
<td>2</td>
<td>...</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>4</td>
<td>?</td>
<td>4</td>
<td>?</td>
<td>3</td>
<td>?</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Does not require meta-data nor domain-specific knowledge!

- dominant approach in large, real-time systems,
- recommendations are built by examining similar users,
- subject of our research
Netflix Prize

• Open competition held by Netflix, an American online movie retailer
• Grand prize of $1,000,000, awarded in 2009
• Goal was to improve the performance of an existing CF model used by Netflix by 10 %
• Encouraged researchers to put huge effort into research in the area of CF
• Contestants were to publish their algorithms during the competition
Netflix Prize: Consequences

- Put a lot of bias into the research

- Techniques and criterions used in Netflix Prize are now considered standard, most notably:
  - there are numerical, **explicit ratings** provided by users,
  - **predictive accuracy** of the model is the only criterion,
  - data matrices must be dense enough

- These conditions do not necessarily hold in all practical problems!
  - ratings may be only implicit and binary, generated from purchase history,
  - predictive accuracy might not fit business needs,
  - data matrices may be very sparse,
  - **all of these are subject to our research**
Formalization

We are given:

- Totally ordered set of items $\mathcal{I} = \{i_1, \ldots, i_n\}$,
- Totally ordered set of users $\mathcal{U} = \{U_1, \ldots, U_n\}$ such that $\forall U_i \in \mathcal{U}: U_i \subseteq \mathcal{I}$

Users are thought as sets of items

- Each user is expressed as a set of items she has purchased/viewed,
- Strong practical motivations: does not require explicit ratings to be collected

For users from $\mathcal{U}$, we are solving problem known as **Top-N recommendation**
Top-N Recommendation

- Problem of generating recommendations from purchase/rating history

- Searching for **predictive model** that would recommend $N$ items most likely to be relevant
  - recommendations must be personalized, user-specific

- Frequently utilized in e-commerce
  - fixed space for recommended items
Approaches to Collaborative Filtering

1. **Memory-based methods**
   - scan the whole database to find similar users (w.r.t. some distance measure),
   - generate recommendations by averaging these users,
   - user-based or item-based $k$-Nearest Neighbors algorithms,
   - fast learning ("lazy learning") phase, slow recommendation phase

2. **Model-based methods**
   - build predictive model from the data,
   - drop details, captures general principles,
   - slow learning phase, fast recommendation phase,
   - clustering, Bayesian networks, Association rules...

3. **Hybrid methods**
   - combination of the two preceding
Measuring Performance of CF Models

- General approach: measuring **predictive accuracy**
- Set of **test users** is used to evaluate the model
  - we split each test user’s history into **observation** and **testing** portion
  - models are to generate predictions based on observation
  - prediction generated are compared to known testing portion
- In Netflix Prize: Root Mean Square Error (RMSE)

\[
\text{RMSE}(model) = \sqrt{\sum_{i \in \text{rated}(U)} (model(U, i) - \text{rating}(U, i))^2}
\]

- In our case of Top-\(N\) recommendation: **Precision on** \(N\)

\[
\text{precision}(model) = \frac{\text{model}(U, N) \cap \text{testing}(U)}{N}
\]
Predictive Accuracy: Issues

- Predictive accuracy may not reflect actual business needs
- Models are pushed towards bestseller items
  - This is because the predictions are compared to existing ratings
  - Recommending bestsellers is generally a good strategy to maximize accuracy
- Long tail recommendation: We want to recommend surprising new items of high value for a specific user
- Other measures were designed to overcome this deficiency, namely the **Catalog coverage**:

\[
\text{coverage(model)} = \frac{|\bigcup_{U \in \mathcal{U}} \text{model}(U, N)|}{|\mathcal{I}|}
\]
Accuracy vs. Coverage

- Accuracy and Coverage are conflicting criterions
- High accuracy leads to low coverage and vice versa
- Accuracy may be viewed as a function of Coverage
- In our research, we consider maximizing both measures as a multi-objective optimization problem
Contributions of the Report

1. Defining selection and parametrization of proper CF algorithm for a given data as an **multi-objective optimization problem**
   - Simultaneous optimization of both the accuracy and the coverage of the model,

2. **Experimental analysis** of several CF algorithms considering Accuracy-Coverage tradeoff

3. Special emphasis on **Association Rules**
   - interesting model for binary-rated data
   - proposal of unifying framework for evaluation multiple variants of rule-based recommendation

4. Experiments with **model ensembles**
   - promising method for generating new Pareto-optimal states in Accuracy-Coverage optimization
We experiment with following algorithms:

- **$k$-Nearest Neighbors**
  - standard approach to CF

- **Association Rules**
  - both weighted and unweighted variants
  - using different rule-quality measures: confidence, lift, conviction

- (Sequential Patterns)
**k-Nearest Neighbors**

- Treats users as vectors (either from $\mathbb{R}^n$ of $\{0, 1\}^n$),
- For given user $U \in \mathcal{U}$, selects the $k$ most similar users w.r.t. some distance measure,
- Sums the user vectors up, and recommends the movies that correspond to positions of highest values in the resulting vector
- Cosine similarity is the typical distance measure for $a, b \in \mathbb{R}^n$:
  \[
  \text{sim}(a, b) = \frac{a \cdot b}{\|a\| \cdot \|b\|}
  \]
- In our case of binary ratings, we are using much faster formula:
  \[
  \text{sim}(A, B) = \frac{|A \cap B|}{\sqrt{|A| \cdot |B|}}
  \]
**Algorithm 1: k-NN-Based Recommendation**

input : Set of users $\mathcal{U}$, Target user $U \in \mathcal{U}$,  
Number of items to be recommended $N \in \mathbb{N}$,  
Number of neighbors to be examined $k \in \mathbb{N}$  
output: Top-$N$ recommendations $R(U) \in \mathcal{I}^N$  
dist $\leftarrow$ init_table()  
foreach $U' \in \mathcal{U}$ such that $U' \neq U$ do  
\[
\text{dist}[U'] \leftarrow \text{distance}(U, U')
\]

sorted_users $\leftarrow$ ascending_sort_by_value(dist)  
cand_items $\leftarrow$ init_table()  
for $i \leftarrow 1$ to $k$ do  
\[
\text{foreach item } \in \text{sorted_users}[k] \text{ do}
\]
\[
\text{if item } \notin \text{cand_items} \text{ then}
\]
\[
\text{cand_items}[\text{item}] \leftarrow 0
\]
\[
\text{cand_items}[\text{item}] \leftarrow \text{cand_items}[\text{item}] + 1
\]

sorted_items $\leftarrow$ descending_sort_by_value(cand_items)  
recomms $\leftarrow \emptyset$  
for $i \leftarrow 1$ to min ($N$, length (sorted_items)) do  
\[
\text{recomms } \leftarrow \text{recomms } \cup \{\text{sorted_items}[i]\}
\]

return recomms
Association Rules [Agrawal1994]

**Association rules** are simple statements about co-occurrences of events in data.

**Definition: Association Rule (AR)**

Let $\mathcal{I}$ be a finite set of **items**, and let $\mathcal{D} = \{T_1, \ldots, T_m\}$ be a finite set of **transactions** such that $\forall T_i \in \mathcal{D}: T_i \subseteq \mathcal{I}$. **Association Rule** is an implication

$$X \Rightarrow Y,$$

such that $X, Y \subseteq \mathcal{I}, X \neq \emptyset, Y \neq \emptyset, X \cap Y = \emptyset$.

We may use Association Rules to generate recommendation [Sarwar2000].
Let $\mathcal{I}$ be a set of **items**, and let $\mathcal{D}$ be a set of **transactions**. For a subset $A \subseteq \mathcal{I}$, we denote the **support** of $A$ as:

$$\text{supp}(A) = \frac{|\{T \in \mathcal{D} | A \subseteq T\}|}{|\mathcal{D}|}.$$ 

Given **minimal support** $s_{\text{min}} \in [0, 1]$, our task is to find set of ARs $X \Rightarrow Y$ such that

$$\text{supp}(X \cup Y) \geq s_{\text{min}}.$$ 

Standard approach to mining ARs: APRIORI algorithm

- we use this algorithm in our experiments
Rule-Based Recommendation

- ARs can be used for Top-$N$ recommendation
- We use two different variants
  - "Best-Rule" method as proposed in [Sarwar2000],
  - "Weighted-Rules" method following [Kononenko1992],
- We experiment with different rule-quality measures
  - confidence,
  - lift,
  - conviction
Algorithm 2: Best-Rule Recommendation

**input**: Set of users $\mathcal{U}$, Target user $U \in \mathcal{U}$, 
Set of association rules $\mathcal{R}$, 
Number of items to be recommended $N \in \mathbb{N}$, 

**output**: Top-$N$ recommendations $R(U) \in \mathcal{I}^N$

$cand\_rules \leftarrow \{(X \Rightarrow Y) \in \mathcal{R} \mid X \subseteq U\}$

$qualities \leftarrow init\_table()$

for $(X \Rightarrow Y) \in cand\_rules$ do
  $qualities[(X \Rightarrow Y)] \leftarrow measure((X \Rightarrow Y), U)$

$sorted\_rules \leftarrow descending\_sort\_by\_value(qualities)$

$recomms \leftarrow \emptyset$

for $i \leftarrow 1$ to length($sorted\_rules$) do
  $(X \Rightarrow Y) \leftarrow sorted\_rules[i]$
  for each item $\in Y$ do
    if item $\notin (U \cup recomms)$ then
      $recomms \leftarrow recomms \cup \{item\}$
      if $|recomms| = N$ then
        return recomms

return recomms
**Algorithm 3: Weighted-Rules Recommendation**

**input**: Set of users $\mathcal{U}$, Target user $U \in \mathcal{U}$, Set of association rules $\mathcal{R}$, Number of items to be recommended $N \in \mathbb{N}$

**output**: Top-$N$ recommendations $R(U) \in \mathcal{I}^N$

1. $\text{applicable\_rules} \leftarrow \{(X \Rightarrow Y) \in \mathcal{R} \mid X \subseteq U\}$
2. $\text{cand\_items} \leftarrow \text{init\_table}()$
3. for $(X \Rightarrow Y) \in \text{applicable\_rules}$ do
   1. foreach item $\in (Y \setminus U)$ do
      1. if item $\notin \text{cand\_items}$ then
         1. $\text{cand\_items}[\text{item}] \leftarrow 0$
      2. $\text{cand\_items}[\text{item}] \leftarrow \text{cand\_items}[\text{item}] + \text{measure}((X \Rightarrow Y), \mathcal{U})$
4. $\text{sorted\_items} \leftarrow \text{descending\_sort\_by\_value}(\text{cand\_items})$
5. $\text{recomms} \leftarrow \emptyset$
6. for $i \leftarrow 1$ to $N$ do
      1. $\text{recomms} \leftarrow \text{recomms} \cup \{\text{sorted\_items}[i]\}$
7. return $\text{recomms}$
Rule-Quality Measures

When sorting rules by quality, we may use several measures:

- **Confidence**

  \[
  \text{conf}(X \Rightarrow Y) = \frac{\text{supp}(X \cup Y)}{\text{supp}(X)}
  \]

- **Lift**

  \[
  \text{lift}(X \Rightarrow Y) = \frac{\text{conf}(X \Rightarrow Y)}{\text{supp}(Y)}
  \]

- **Conviction**

  \[
  \text{conv}(X \Rightarrow Y) = \frac{1 - \text{supp}(Y)}{1 - \text{conf}(X \Rightarrow Y)}
  \]
Experiments

• Several experiments were done with all the aforementioned algorithms

• We used real-world datasets from IPTV industry, namely the Video-on-Demand (VoD) service,
  • Different installations of nangu.TV platform, a comprehensive solution allowing ISPs to run IPTV services on their networks
  • Only purchase history available $\rightarrow$ binary ratings
Experimental datasets

Nangu-TV-1 Dataset

#Users: 9004   #Items: 17558   #Purchases: 136494

<table>
<thead>
<tr>
<th>Quantity</th>
<th>MIN</th>
<th>Q1</th>
<th>Q2</th>
<th>Q3</th>
<th>MAX</th>
</tr>
</thead>
<tbody>
<tr>
<td>Purchases per User</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>12</td>
<td>979</td>
</tr>
<tr>
<td>Purchases per Item</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>6</td>
<td>1697</td>
</tr>
</tbody>
</table>

Nangu-TV-2 Dataset

#Users: 15803   #Items: 738   #Purchases: 946807

<table>
<thead>
<tr>
<th>Quantity</th>
<th>MIN</th>
<th>Q1</th>
<th>Q2</th>
<th>Q3</th>
<th>MAX</th>
</tr>
</thead>
<tbody>
<tr>
<td>Purchases per User</td>
<td>1</td>
<td>12</td>
<td>37</td>
<td>85</td>
<td>552</td>
</tr>
<tr>
<td>Purchases per Item</td>
<td>1</td>
<td>497</td>
<td>1058</td>
<td>1888</td>
<td>6480</td>
</tr>
</tbody>
</table>
$k$-NN on Nangu-TV-1

![Graph showing Precision-Coverage tradeoff for different values of $k$ on the Nangu-TV-1 dataset](image)

**Figure**: Precision-Coverage tradeoff for different values of $k$ on the Nangu-TV-1 dataset
Association Rules on Nangu-TV-1

Figure: Precision-Coverage tradeoff for different configurations of AR-based recommender on the *Nangu-TV-1* dataset
Comparison of $k$-NN and ARs on Nangu-TV-1

**Figure**: Different variants of Association Rules compared with $k$-NN on the *Nangu-TV-1* dataset.
Ensemble of $k$-NN and ARs on Nangu-TV-1

Figure: Ensemble of $k$-NN, best-rule conviction ARs, and average-rules lift ARs on the Nangu-TV-1 dataset
Figure: Precision-Coverage tradeoff for different values of $k$ on the Nangu-TV-2 dataset
Figure: Precision-Coverage tradeoff for different configurations of AR-based recommender on the *Nangu-TV-2* dataset
Comparison of $k$-NN and ARs on Nangu-TV-2

Figure: Different variants of Association Rules compared with $k$-NN on the Nangu-TV-2 dataset
Ensemble of $k$-NN and ARs on Nangu-TV-1

Figure: Ensemble of $k$-NN, best-rule conviction ARs, and average-rules lift ARs on the Nangu-TV-2 dataset
Ensembling $k$-NN, ARs, and Sequential Patterns

On another dataset, Nangu-TV-3, we experimented with Sequential Patterns (SPs).

Figure: Different ensembles of $k$-NN, Association Rules, and Sequential Patterns on the Nangu-TV-3 dataset
Visualizing Association Rules for Nangu-TV-1

Figure: Association Rules Graph for the Nangu-TV-1 Dataset
Visualizing Association Rules for Nangu-TV-2

Figure: Association Rules Graph for the Nangu-TV-2 Dataset
Visualizing Association Rules for Nangu-TV-2

Figure: Filtered Association Rules Graph for the Nangu-TV-2 Dataset
Visualizing Association Rules for “Timeshift” Data

Figure: Association Rules for “Timeshift” Dataset
Confidence-driven AR-based Recommendation on MovieLens Dataset

Figure: Recommendations for a Specific User from the Database on the MovieLens Dataset, using Confidence as the Rule-Quality Metric
Lift-driven AR-based Recommendation on MovieLens Dataset

Figure: Recommendations for a Specific User from the Database on the MovieLens Dataset, using Lift as the Rule-Quality Metric
Title: **Meta-Learning Templates for Collaborative Filtering**

Topics:

- **Multi-Objective Optimization in Recommender Systems**
  - Extending results from this report
  - Making more experiment with more algorithms and datasets

- **Meta-Learning Templates in Collaborative Filtering**
  - Searching algorithm ensembles maximizing all the performance measures
    - universally applicable?
    - data-dependent?
Thank you for your attention!

Tomáš Řehořek
tomas.rehorek@fit.cvut.cz

Acknowledgements

Our research is partially supported by the following grants:

- Novel Model Ensembling Algorithms (SGS10/307/0HK3/3T/181) grant of the Czech Technical University in Prague,

- Research and development of the interactive and platform services for the interactive digital TV broadcast MPO/FR grant (FR-TI2/128).

We would like to thank the Alnair, a.s. company (developer of the nangu.TV platform) for supporting our research and providing us with the data.