Collaborative Filtering: A Machine Learning Perspective (Formulations)

Sara Javanmardi
sjavanma@uci.edu
Outline

• Definition
• Origin: The Tapestry System (1992)
• A Space of Formulations
  – Preference Indicators
  – Additional Features
  – Preference Dynamics
• Collaborative Filtering Algorithms
  – Memory-based Algorithms
  – Model-based Methods
• Conclusion
Collaborative filtering or recommender systems use a database about user preferences to predict additional topics or products a new user might like.
The Tapestry mail system

- Tapestry is intended to handle any incoming stream of electronic documents and serves both as a mail filter and repository.
- Prevent users from being inundated by a huge stream of incoming documents.
- Information filtering can be more effective when humans are involved in the filtering process.
- To support both content-based filtering and collaborative filtering, it entails people collaborating to help each other perform filtering by recording their reactions (annotations) to documents they read.
- Annotations can be accessed by other people’s filters.
How did Tapestry work?

- To support classification of documents, Tapestry provides appraiser functions.
- The Tapestry architecture performs filtering in two steps:
  - The first level of filtering is performed by filter queries.
  - The second level of filtering is done by appraiser functions that run only over the contents of accepted documents in the first level.
- Tapestry Query Language (TQL): Like SQL but it supports sets.

Example 1:
m.sender = ‘Smith’ OR
m.date < ’April 15, 1991’) AND
m.subject LIKE ’%Tapestry%’.

Example 2:
m.to = {’Joe’, LIKE ’%Bill%’}
An annotation object, which always has a field `msg` that links it to an email.

- TQL helps to handle annotations.
- Annotations are not stored as fields of the document they annotate, TQL treating them as additional document fields.

Annotation:

```plaintext
a.type = 'vote'
AND a.owner = 'weiser'
AND a.msg = m
```

Now it’s possible to have queries like:

```plaintext
EXISTS (a: a.type = 'vote'
AND a.owner = 'weiser'
AND a.msg = m)
```
A Space of Formulations

- Structuring of the space of formulations based on three independent characteristics:
  - the type of preference indicators used
  - the inclusion of additional features
  - the treatment of preference dynamics

- **Goal**: To propose a structure that covers all formulations of collaborative filtering currently under study.
Preference Indicators

• The main types of preference indicators:
  – numerical
  – ratings triplets
  – numerical rating vectors
  – co-occurrence pairs
  – count vectors
Preference Indicators

• Rating triplet \((u, y, r)\) where \(u\) is a user index, \(y\) is an item index, and \(r\) is a rating value.

• The rating values may be ordinal or continuous.

• numerical rating vector has the form:

\[
{r}^{u} = (r_{1}^{u}, \cdots, r_{M}^{u})
\]

• where \(r_{y}^{u}\) is the rating assigned by user \(u\) to item \(y\).

• The components of the vector \(r_{y}^{u}\) are either all ordinal or all continuous values. Any component of the vector may be assigned the value \(\bot\), indicating the rating for the corresponding item is Unknown.
In a pure approach, users are described by their preferences for items, and items are described by the preferences users have for them.

When additional content-based features are included the formulation is sometimes called hybrid collaborative Filtering.

Additional features can include information about users such as age and gender, and information about items such as an author and title for books, an artist and genre for music, ...
Preference Dynamics

• Over long periods of time, or in domains where user preferences are highly dynamic, older preference indicators may become inaccurate.

• Some methods are proposed for dealing with dynamic user profiles.
  – Advantage:
    • They can deal naturally with user preferences changing over time.
  – Disadvantage:
    • more complex models and prediction methods are required.
Pure, Non-Sequential, Rating-Based Formulation

- **Formal Definition**
  - $M$ items 1,⋯, $M$
  - $N$ users 1,⋯, $N$
  - A user $u$ can provide an opinion about an item $y$ by assigning it a numerical rating $r^u_y$ from the ordinal scale $1,\cdots, V$
  - Each user can supply at most one rating for each item.
  - A user rating vector (user rating profile) is assigned with each user:

$$r^u \in \{1, \cdots, V, \bot\}^M$$
Pure, Non-Sequential, Rating-Based Formulation

• The task of rating prediction
  – We are given the rating vectors of the N users, and the rating vector of a particular active user $a$.
  – We wish to predict rating values for all items that have not yet been assigned ratings by the active user $a$. 
Additional Features (cont.)

• Reduces the effect of two well-known problems:
  – The cold start problem
  – The new user problem
The Cold Start Problem

- It occurs when there are few entries recorded in the rating database.

- In this case more accurate recommendations can be made by recommending items according to similarities in their content-based features.
The New User Problem

• It occurs in an established collaborative filtering system when recommendations must be made for a user on the basis of few recorded ratings.

• In this case better recommendations may be achieved by considering similarities between users based on additional user features.
• The task of producing recommendations
  – computing predictions for all unrated items,
  – sorting the predicted ratings
  – recommending the top T items.
Collaborative Filtering Techniques

- Memory-Based Algorithms
- Model-Based Methods
Memory-Based Algorithms

- $v_{i,j} = \text{vote of user } i \text{ on item } j$
- $I_i = \text{items for which user } i \text{ has voted}$
- Mean vote for $i$ is

$$\overline{v}_i = \frac{1}{|I_i|} \sum_{j \in I_i} v_{i,j}$$

- Predicted vote for “active user” $a$ is weighted sum:

$$p_{a,j} = \overline{v}_a + \kappa \sum_{i=1}^{n} w(a,i)(v_{i,j} - \overline{v}_i)$$

normalizer weights of $n$ similar users
Memory-Based Algorithms (cont)

- K-nearest neighbor

\[ w(a, i) = \begin{cases} 
1 & \text{if } i \in \text{neighbors}(a) \\
0 & \text{else}
\end{cases} \]

- Pearson correlation coefficient:

\[ w(a, i) = \frac{\sum_j (v_{a,j} - \bar{v}_a)(v_{i,j} - \bar{v}_i)}{\sqrt{\sum_j (v_{a,j} - \bar{v}_a)^2 \sum_j (v_{i,j} - \bar{v}_i)^2}} \]

- Vector Similarity (Cosine distance)

\[ w(a, i) = \sum_j \frac{v_{a,j}}{\sqrt{\sum_{k \in I_a} v_{a,k}^2}} \frac{v_{i,j}}{\sqrt{\sum_{k \in I_i} v_{i,k}^2}} \]
Model-Based Methods (From Probability Perspective)

- Cluster Models

\[ \Pr (C = c, v_1, \ldots, v_n) = \Pr (C = c) \prod_{i=1}^{n} \Pr (v_i | C = c) \]

- Bayesian Network Model
Conclusion

- Bayesian networks with decision trees at each node and correlation methods outperform Bayesian-clustering and vector-similarity methods.

- Between correlation and Bayesian networks, the preferred method depends on:
  - The nature of the dataset
  - The nature of the application (ranked versus one-by-one presentation)
  - The availability of votes with which to make predictions
  - Size of database
  - Speed of predictions
  - Learning time
Questions?