Prediction and Filtering for Recommender Systems

Definitions

Recommendation generation problem. Given a **set of users** and their (incomplete) **preferences** over **set of items**, find, for each user **new items for which they would have high preferences**.

Users. $C = \{c_1, \ldots, c_M\}$: a set of **users**.

Items. $S = \{s_1, \ldots, s_n\}$: a set of **items**.

Utility. $u : C \times S \longrightarrow \mathcal{R}$.

$u(c, s)$: *utility*, or *rating* or *preference* of user c for item s.

Typically, **utility function** is incomplete.

Utility function $u(c, s)$ can also be viewed as a **utility matrix** $u[.,.]$, where $u[i, j] = u(c_i, s_j)$. Matrix $u[]$ is **typically sparse**.

Problem. The main problem solved can be phrased in a number of ways:

- **User-based recommendations.** Given a user c find items $s'_1, \ldots s'_k$ (such that $u(c, s'_i)$ is undefined), for which c is predicted to have highest utility.
- **Individual ratings.** Given a user c and an item s, predict $u(c, s)$.
- **Item-based recommendations.** Given an item s, find users $c'_1, \ldots, c'_k$, (s.t., c'_i, s is undefined) for which $u(s, c'_i)$ is predicted to be the highest.

Recommender Systems

Recommender System: a system, which given C, S and a *partial* utility function u, solves one or more of the problems of recommendation generation.

Score Prediction in Recommender Systems

Collaborative Prediction. Given $c \in C$ and $s \in S$, estimate $u(c, s)$ based on the **known utilities** $u(c', s)$ for item s for users $C' = \{c'\} \subseteq C$.

Item-based Prediction. Given $c \in C$ and $s \in S$, estimate $u(c, s)$ based on the **known utilities** $u(c, s')$ for user c for items $S' = \{s'\} \subseteq C$.

Aggregation of the utilities. **score prediction methods aggregrate** the known utilities $u(c_i, s)$ for item s to predict the utility (rating) of s for user c (user-based aggregation):

$$u(c, s) = \mathsf{aggregate}_{c_i \in C} u(c_i, s).$$

Similar aggregation exists for items:

$$u(c, s) = \mathsf{aggregate}_{s_i \in S} u(c, s_i).$$

Notation. Let $s \in S$ be some item. As C^s we denote the set

$$C^s = \{c \in C | u(s, c) \ \ \textbf{is defined}\},$$

i.e., the set of all users which have an existing rating (utility) for item s.

Similarly, for a user $c \in C$,

$$S^c = \{s \in S | u(s, c) \ \ \textbf{is defined}\}.$$

Notation. Let $c \in C$ and let $S = \{s_1, \ldots, s_n\}$. As $u[c]$ we denote the (sparse) vector:

$$u[c] = (u(c, s_1), u(c, s_2), \ldots, u(c, s_n)).$$

Note. In the computations below, whenever we see a value of $u(c, s)$ that is not defined in the dataset, we assume that its value is 0 in all computations.

Mean utility. The most simple collaborative predictor is the mean utility of an item for all users who rated it.

$$u(c, s) = \frac{1}{|C^s|} \sum_{c_i \in C^s} u(c_i, s).$$

Similarly, we can have an item-based predictor: the mean (average) rating a user gave:

$$u(c,s) = \frac{1}{|S^c|} \sum_{s_i \in S^c} u(c, s_i).$$

Both of these are *very simplistic* predictions, as they ignores various information.

Collaborative Weighted sum. This predictor is one of the most commonly used. It assumes existence of a **similarity function** $sim(.,.)$ which reports the proximity between utility vectors for two users.

$$u(c,s) = k \cdot \sum_{c' \neq c} sim(u[c], u[c']) \cdot u(c', s),$$

where k, the *normalization factor* is typically set to

$$k = \frac{1}{\sum_{c' \neq c} |sim(u[c], u[c'])|}$$

$$u(c,s) = \frac{1}{\sum_{c' \neq c} |sim(u[c], u[c'])|} \cdot \sum_{c' \neq c} sim(u[c], u[c']) \cdot u(c', s),$$

Item-based Weighted sum. The item-based version of this predictor uses the **similarity function** $sim(.,.)$ which reports the proximity between two items.

$$u(c,s) = k \cdot \sum_{s' \neq s} sim(u[s], u[s']) \cdot u(c, s'),$$

where k, the *normalization factor* is typically set to

$$k = \frac{1}{\sum_{s' \neq s} |sim(u[s], u[s'])|}$$

$$u(c,s) = \frac{1}{\sum_{s' \neq s} |sim(u[s], u[s'])|} \cdot \sum_{s' \neq s} sim(u[s], u[s']) \cdot u(c, s'),$$

Weighted sum predictors have one weakness:

- **for collaborative prediction: insensitivity** to the fact that different users employ the rating/utility scale **differently** when reporting their preferences.
- **for item-based prediction: insensitivity** to the fact that different items may be rated by everyone differently.

Collaborative adjusted weighted sum. In predicting the utilities for a specific user, we take into account, the user's approach to rating the items. First, we compute the user's average rating $\bar{u_c}$:

$$\bar{u_c} = \frac{1}{|S^c|} \sum_{s' \in S^c} u(c, s').$$

We then predict $u(c,s)$ for some item $s \in S$ as follows:

$$u(c,s) = \bar{u_c} + k \cdot \sum_{c' \neq c} sim(u[c], u[c']) \cdot (u(c', s) - \bar{u_{c'}}).$$

Here, k is the same normalizing factor as above.

Item-based adjusted weighted sum. The item-based version of this predictor works as follows. First we compute the average rating for the item: $\bar{u_s}$:

$$\bar{u_s} = \frac{1}{|C^s|} \sum_{c' \in C^s} u(c', s).$$

We then predict $u(c,s)$ for some item $s \in S$ as follows:

$$u(c,s) = \bar{u_s} + k \cdot \sum_{s' \neq s} sim(u[s], u[s']) \cdot (u(c, s') - \bar{u_{s'}}).$$

Here, k is the normalizing factor from the item-based weighted sum computations.

N Nearest Neighbors predictors

All predictors discussed above can be updated **to include only N nearest neighbors of the user c in the comparison**.

Let $C'_c = \{c' \in C | rank(sim(u[c], u[c'])) \leq N\}$ be the set of N **nearest neighbors to user** c using similarity function $sim(.,.)$ for users.

Similarly, let $S'_s = \{s' \in S | rank(sim(u[s], u[s'])) \leq N\}$ be the set of N **nearest neighbors to item** s using similarity function $sim(.,.)$ for items.

Average Nnn ranking. Collaborative version:

$$u(s,c) = \frac{1}{N} \sum_{c' \in C'_c} u(c', s).$$

Item-based version:

$$u(s,c) = \frac{1}{N} \sum_{s' \in S'_s} u(c, s').$$

Weighted Nnn sum. Collaborative version:

$$u(c,s) = k \cdot \sum_{c' \in C'_c} sim(u[c], u[c']) \cdot u(c', s).$$

$$k = \frac{1}{\sum_{c' \in C'_c} |sim(c, c')|}.$$

Item-based version:

$$u(c,s) = k \cdot \sum_{s' \in S'_s} sim(u[s], u[s']) \cdot u(c, s').$$

$$k = \frac{1}{\sum_{s' \in S'_s} |sim(s, s')|}.$$

Adjusted weighted Nnn sum. Collaborative version:

$$u(c,s) = \bar{u_c} + k \cdot \sum_{c' \in C'_c} sim(u[c], u[c']) \cdot (u(c', s) - \bar{u_{c'}}).$$

Item-based version:

$$u(c,s) = \bar{u_s} + k \cdot \sum_{s' \in S'_s} sim(u[s], u[s']) \cdot (u(c, s') - \bar{u_{s'}}).$$

Similarity Measures

Two similarity measures are typically used in collaborative filtering.

Pearson Correlation. User similarity

$$sim(u[c], u[c']) = \frac{\sum_{i=1}^{n}(u(c, s_i) - \bar{u_c}) \cdot (u(c', s_i) - \bar{u_{c'}})}{\sqrt{\sum_{i=1}^{n}(u(c, s_i) - \bar{u_c})^2 \cdot \sum_{i=1}^{n}(u(c', s_i) - \bar{u_{c'}})^2}}.$$

Substituting s for c and c for s would yield the Pearson correlation for items.

This measure reflects *statistical correlation* between the two (sparse) vectors of data.

Cosine similarity. User similarity

$$sim(u[c], u[c']) = cos(u[c], u[c']) = \frac{u[c] \cdot u[c']}{||u[c]|| \cdot ||u[c']||} = \frac{\sum_{i=1}^{n} u(c, s_i) \cdot u(c', s_i)}{\sqrt{\sum_{i=1}^{n} u(c, s_i)^2 \cdot \sum_{i=1}^{n} u(c', s_i)^2}}.$$

Substituting s for c and c for s would yield the Pearson correlation for items.

Cosine similarity measures the *colinearity* of the two vectors (it is 1 if the vectors are colinear, and 0 if they are orthogonal).

Default Voting. Default voting extends Pearson Correlation similarity measure by substituting a default vote on all items not explicitly rated by the user.

$$sim(u[c], u[c']) =$$

$$\frac{(n+k)(\sum_j u(c, s_j)u(c', s_j) + kd^2) - (\sum_j u(c, s_i) + kd)(\sum_j u(c', s_j) + kd)}{\sqrt{((n+k)(\sum_j u^2(c, s_j) + kd^2) - (\sum_j u(c, s_j) + kd)^2)((n+k)(\sum_j u^2(c', s_j) + kd^2) - (\sum_j u(c', s_j) + kd)^2)}}$$

Here, in addition to all items rated by both users c and c', we assign a score of d to k items that neither user has rated. *Usually, d is a **neutral**, or **mildly negative** score*.

Inverse User Frequency. *Inverse user frequency* f_j of an item s_j is defined as

$$f_j = \log_2 \frac{|C^{s_j}|}{|C|}$$

The more frequently the item is rated, the lower its inverse user frequency is. The idea is that items that are rated infrequently should play more role in identifying similarity between users.

The *transformed vote* $v(c, s_j)$ is defined as:

$$v(c, s_j) = f_j u(c, s_j).$$

We can transform both the cosine similarity and the Pearson correlation measure, to use transformed votes instead of the original ratings. For *cosine similarity* the new formula will be:

$$sim(u[c], u[c']) = \frac{v[c] \cdot v[c']}{||v[c]|| \cdot ||v[c']||} = \frac{\sum_{i=1}^{n} v(c, s_i) \cdot v(c', s_i)}{\sqrt{\sum_{i=1}^{n} u(v, s_i)^2 \cdot \sum_{i=1}^{n} u(v', s_i)^2}}.$$

For Pearson correlation, the new formula is:

$$sim(u[c], u[c']) = \frac{\sum_j f_j \sum_j f_j u(c, s_j) u(c', s_j) - (\sum_j f_j u(c, s_j))(\sum_j f_j u(c', s_j))}{\sqrt{UV}}$$

where

$$U = \sum_j f_j \left(\sum_j f_j u^2(c, s_j) - \left(\sum_j f_j u(c, s_j) \right)^2 \right);$$

$$V = \sum_j f_j \left(\sum_j f_j u^2(c', s_j) - \left(\sum_j f_j u(c', s_j) \right)^2 \right).$$

Case Amplification

Case amplification is a technique that modifies the ratings $u(c, s)$ *before* they are used in collaborative filtering methods. The amplification usually is applied for $u(c, s) \in [-1, 1]$ and it rewards weights closest to 1 and punishes negative weights (ratings). A typical case amplification scheme is:

$u^{amp}(c, s) = u^{\rho}(c, s)$, if $u(c, s) \geq 0$;
$u^{amp}(c, s) = -(-u^{\rho}(c, s))$, if $u(c, s) < 0$;

The $u^{amp}(c, s)$ ratings are then used in computations.

References

[1] G. Adomavicius, A. Tuzhilin. Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions, *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, No. 6, June 2005.