Comparative study of personalizing recommender systems based on shopping system

Zhihang Tang, Zhonghua Wen

School of Computer and Communication, Hunan Institute of Engineering Xiangtan 411104, China

Abstract

Making choices is an integral part of everyday life; Recommender systems facilitate decision-making processes through informed assistance and enhanced user experience. To aid in the decision-making process, recommender systems use the available data on the items themselves, Personalized recommender systems subsequently use this input data, and convert it to an output in the form of ordered lists or scores of items in which a user might be interested. These lists or scores are the final result the user will be presented with, and their goal is to assist the user in the decision-making process. Recommender systems facilitate making choices, improve user experience, and increase revenue, therefore should be easily accessible for deployment to interested parties. The implementation of recommender systems in RapidMiner has been additionally simplified through the Recommender Extension.

Keywords: shopping system, personalized recommender systems, RapidMiner, decision-making

1 Introduction

Recommender systems have become extremely common in recent years, and are applied in a variety of applications. The most popular ones are probably movies, music, news, books, research articles, search queries, social tags, and products in general. However, there are also recommender systems for experts, jokes, restaurants, financial services, life insurance, persons (online dating), and twitter followers [1].

Recommender systems typically produce a list of recommendations in one of two ways – through collaborative or content-based filtering [2]. Collaborative filtering approaches build a model from a user's past behavior (items previously purchased or selected and/or numerical ratings given to those items) as well as similar decisions made by other users; then use that model to predict items (or ratings for items) that the user may have an interest in [3].

Content-based filtering approaches utilize a series of discrete characteristics of an item in order to recommend additional items with similar properties. These approaches are often combined (see Hybrid Recommender Systems).

Each type of system has its own strengths and weaknesses. In the above example, Last.fm requires a large amount of information on a user in order to make accurate recommendations. This is an example of the cold start problem, and is common in collaborative filtering systems. While Pandora needs very little information to get started, it is far more limited in scope (for example, it can only make recommendations that are similar to the original seed). Recommender systems are a useful alternative to search algorithms since they help users discover items they might not have found by themselves. Interestingly enough, recommender systems are often implemented using search engines indexing non-traditional data. Montaner provides the first overview of recommender systems, from an intelligent agents perspective [4]. Adomavicius provides a new overview of recommender systems [5]. Herlocker provides an additional overview of evaluation techniques for recommender systems [6]. They also provide a literature survey on research paper recommender systems [7].

Recommender system is an active research area in the data mining and machine learning areas. Some conferences such as RecSys, SIGIR, KDD have it as a topic.

2 Basic conception

2.1 RECOMMENDATION OPERATORS

The Recommender Extension has a total 26 recommendation operators. These operators are grouped in the following categories: Item Recommendation, Item Rating Prediction, and Recommender Performance.

Item recommendation operators operate over large matrices that contain information about which user consumed which item. These input matrices often contain large numbers of empty entries, and can thus be used in a more space-efficient way. We describe the appropriate format used for efficient data loading and storage in the next subsection.

2.2 DATA

Typical recommender systems, operating on user usage data, are built on top of large matrices called utility matrices. These matrices usually contain elements from a limited set of numbers, for example from 0, 1, 2, 3, 4, 5, where 0 typically denotes that the user had no interaction with the item, and the rest describes the level of that

interaction in a form of rating. Due to a large number of both users and items, these matrices are typically very large. In addition, since users mostly consume a very small portion of items out of the total number of items, these matrices tend to contain a lot of zeros-they tend to be very sparsely populated. This is why special data structures for handling sparse data need to be implemented. In RapidMiner, we can use AML reader operators to read such datasets. Input datasets used to learn a recommender system model must be formatted in two columns; for example, the first column can contain user IDs, while the second can contain item IDs. Attributes names, and their positioning can be arbitrary. Prior to applying recommenddation operators to input datasets, proper roles have to be set for these attributes. Any additional attributes will not be considered. An example of an AML, and a related DAT file for item recommendation operators is given in Fig. 1.

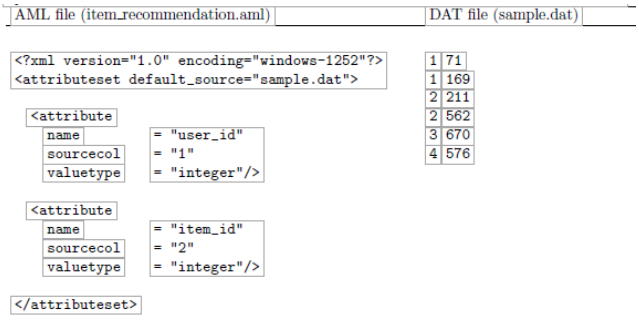


FIGURE 1 An example of an AML and a related DAT file for item recommendation operators

The recommender system datasets used throughout this paper consists of content and collaborative data. Content data was taken from the shopping system.

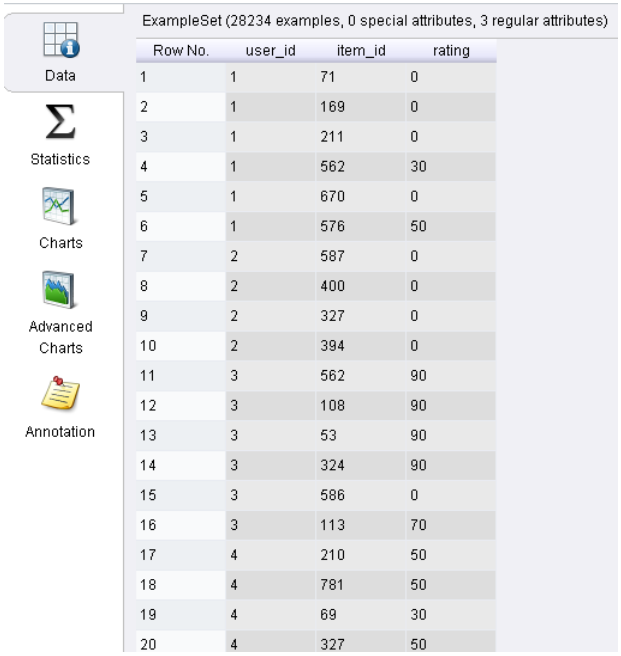
The content dataset described contains the following content attributes for each item:

brand ID: a unique integer that represents a lecture;

brand name: a text string containing a name of a particular lecture;

brand description: a text string denoting a description of a particular lecture.

A particular item identifier is denoted by the small letter i and the set of all items is denoted by the capital letter I. Collaborative data contains synthetic click streams of users, where each click stream is a sequence of items viewed by a particular user in some time interval. In the following text, we refer to the synthetic users as users. A particular user identifier is denoted by the small letter u and the set of all users is denoted by the capital letter U. Click streams are transformed into the sparse matrix A, which is called the usage matrix. The non-zero elements of the usage matrix (A (i, u)) tell us that the item i was consumed by the user u. Using this dataset, we construct collaborative and content recommender systems in the following sections. The collaborative recommender systems rely on the usage matrix A while the content recommender systems rely on items textual descriptions. We can get Fig. 2.



Row No.	user_id	item_id	rating
1	1	71	0
2	1	169	0
3	1	211	0
4	1	562	30
5	1	670	0
6	1	576	50
7	2	587	0
8	2	400	0
9	2	327	0
10	2	394	0
11	3	562	90
12	3	108	90
13	3	53	90
14	3	324	90
15	3	586	0
16	3	113	70
17	4	210	50
18	4	781	50
19	4	69	30
20	4	327	50

FIGURE 2 Initial data from shopping system

2.3 COLLABORATIVE-BASED SYSTEMS

The main idea of collaborative recommendation approaches is to use information about the past behavior of existing users of the system for predicting which item the current user will most probably like and thus might consume. Collaborative approaches take a matrix of given user-item ratings or viewings as an input and produce a numerical prediction indicating to what degree the current user will like or dislike a certain item, or a list of n recommended items. The created list should not contain items the current user has already consumed.

Neighborhood-based recommender systems work by counting common items two users have viewed for every pair of users in the system, or the number of common users that viewed the same pair of items. Using this count, similarity between two users or items is calculated. Neighborhood systems use intuition that two users who have viewed a large number of common items have similar tastes. That information can be used to recommend items that one user consumed and the other one did not. We are interested in finding pairs of users having the most similar taste, or pairs of items having the most users that viewed both items. Those pairs of users/items are called "the closest neighbors". We describe two main approaches of the neighborhood-based recommender systems: user and item-based nearest neighbor recommendation.

2.3.1 User-based nearest neighbor recommendation

Given a user-item viewing matrix and the ID of the current user as input, identify other users having similar past prefe-

rences to those of the active user. Subsequently, for every product the active user has not yet consumed, compute prediction based on the product usage of the selected user subset. These methods assume that users, who have previously shared similar tastes, will share similar tastes in the future, and that user preferences remain stable and constant over time.

To calculate similarity between users, two typical similarity measures are used: the Pear- son correlation and the Cosine correlation [6]. In our item recommendation problem we used cosine correlation as a similarity measure. Typically, we do not consider all users in the database when calculating user similarity, rather the k most similar ones.

2.3.2 Item-based nearest neighbor recommendation

When dealing with large problems, consisting of millions of users, user-based collaborative filtering approaches lead to increased memory usage and execution time. Since the system is required to calculate a large volume of potential neighbors, it becomes impossible to compute predictions in real time. In some cases, the number of users dominates the number of items in the system so it would be natural to try to use items for making recommendations. That is the reason for creating a second neighborhood-based recommender system based on items instead of users.

As opposed to the user-based approach, the item-based recommendation approach computes prediction using the similarity between items. We use a cosine similarity measure, as we did in the user-based approach. Likewise, as in the user-based approach, we use k-nearest neighbors, i.e. the k most similar items for prediction.

3 Personalizing Recommender Systems

Collaborative recommender operators use the user-item matrix to build a recommendation model. This user-item matrix is presented as an example set of user-item pairs describing user consumption history. The recommendation model built with this matrix is used to recommend items to users from a query set. The query set is an example set containing identification numbers of users for which we want to make recommendations. For each user in the query set we recommend only the items not consumed by this user. Figure 3 depicts a basic collaborative recommender operator workflow.

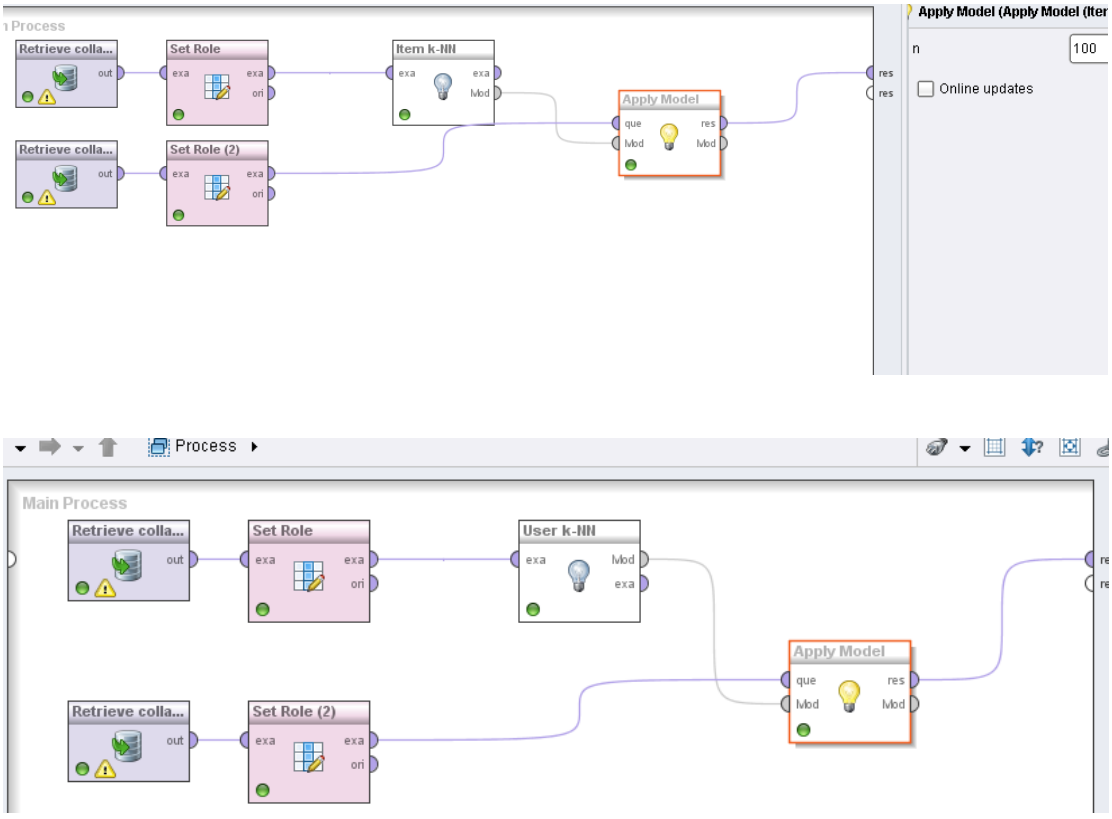


FIGURE 3 Two examples of an item recommendation workflow

The Recommended results shown in Figure 4.

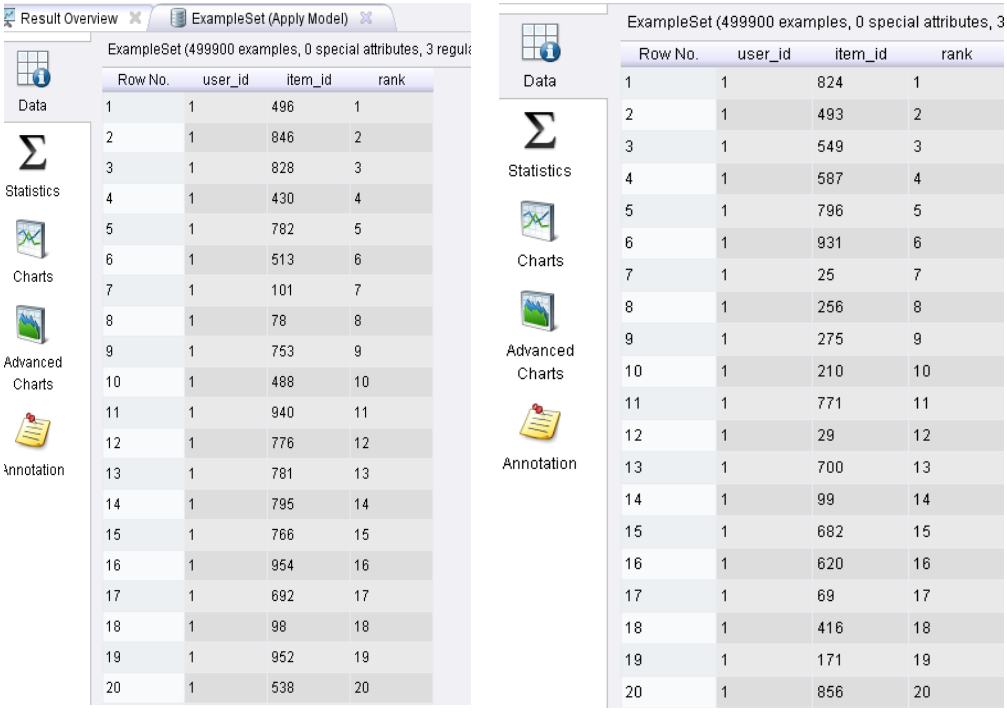


FIGURE 4 Comparative study of the recommended results

In the item recommendation workflow, the first two operators read the train and the query example sets using the Read AML operators (1,4). Following, the appropriate roles are set to attributes using the Set Role operator (2). The user identification role was set to user id attribute and item identification role to item id attribute. Data attributes can have arbitrary names but roles for those attributes must be set. Next, we use the train data with the appropriately set roles to train an Item k-NN model (3). At this point we can use our trained model to recommend new items to users in the query set using the Apply Model operator (6). Prior to model application, the user identification role was set for the query set (5). The Apply Model operator (6) returns an example set containing the first n ranked recommendations for every user in a query set. In Figure 3 we have seen how to make recommendations for particular users. In the following figure, Figure 4, we show how to measure performance of a recommendation model.

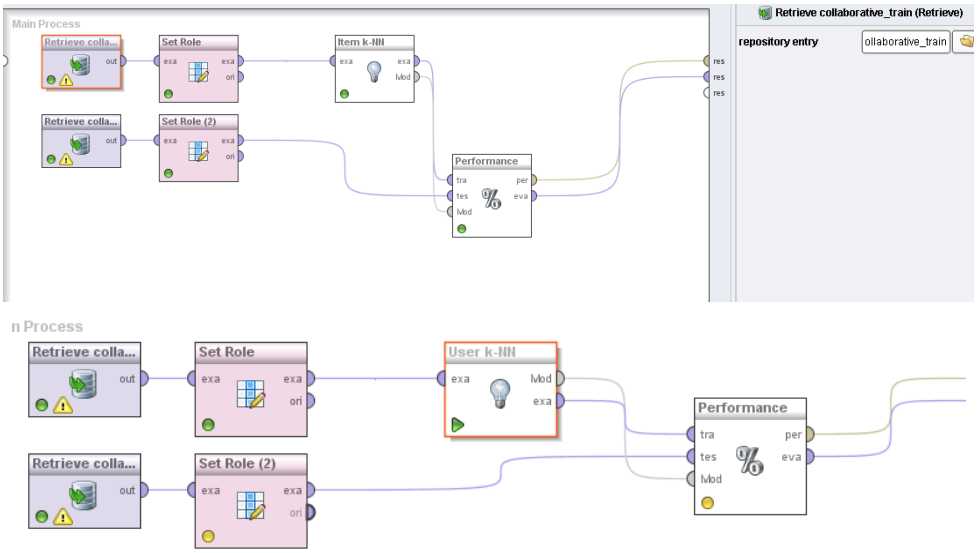


FIGURE 5 Comparative study of measuring performance of a recommendation model

The data management part of the workflow for measuring recommender model performance in Figure 5 is the same as in Figure 3. We use the Read AML operators (1,4) to load the data input, and the Set Role operators (2,5) to set the appropriate roles. In this workflow we use the test data (4) containing two attributes, the user id and the item id attribute and we set user identification and item identification roles to those at-tributes, respectively. The difference from the previous workflow is the need to calculate the performance of our built recommendation model (3). We use the Performance operator (6) to measure standard recommendation error measures we previously defined: AUC, Prec@k, NDCG, and MAP. The Performance operator (6) returns a performance vector and an example set containing performance measures. This enables a user to choose which format suits his or her needs. We can get Figure 6.

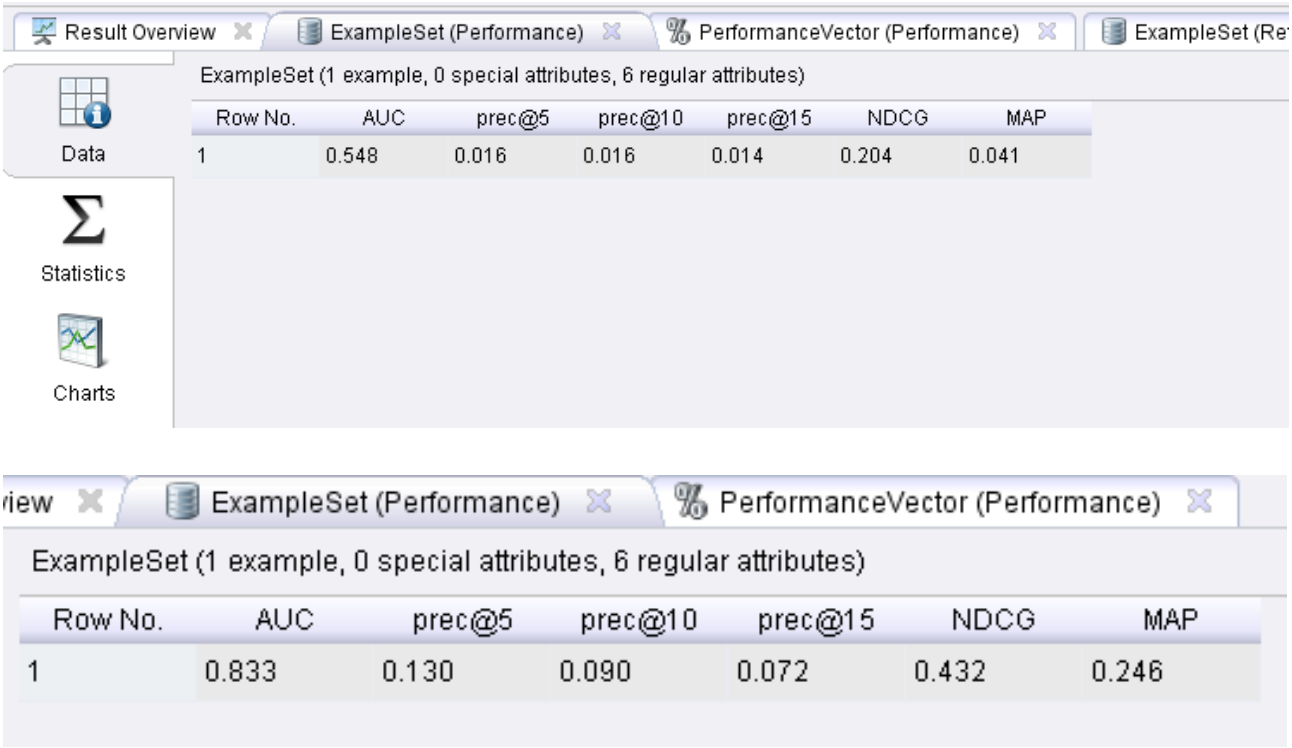


FIGURE 6 Comparative study of the performance of recommender systems

4 Conclusions

Recommender systems became essential in an information- and decision-overloaded world. They changed the way users make decisions, and helped their creators to increase revenue at the same time. Bringing recommender systems to a broader audience is essential in order to popularize them beyond the limits of scientific research and high technology entrepreneurship. The goal of the Recommender Extension for RapidMiner and this paper was to bring recommenders to a broad audience, in a theoretical, practical, and above all, application way.

In this paper we presented recommender systems and their different techniques: collaborative filtering, content-based recommender systems, and hybrid systems. We presented the advantages and disadvantages of each of those systems and demonstrated how they could be implemented easily in RapidMiner. The application of recommender systems outlined was just a small introduction to the possibilities of the extension. We hope you will use the knowledge obtained through this paper in your own applications, problems, and businesses, and that recommender systems will assist you in reaching quality, informed decisions.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (No. 61272295) and 2014 science and technology plan of Hunan province (No. 2014GK 3157).

References

[18] Gupta P, Goel A, Lin J, Sharma A, Wang D, Zadeh R 2013 WTF: The Who to Follow Service at Twitter *Proceedings of the 22th International World Wide Web Conference (WWW 2013)* 505-14
[19] Jafarkarimi H, Sim A T H, Saadatdoost R 2012 A Naïve Recommendation Model for Large Databases *International Journal of Information and Education Technology* **2**(3) 216-9
[20] Melville P, Sindhwani V 2010 Recommender Systems *Encyclopedia of Machine Learning*
[21] Montaner M, Lopez B, de la Rosa J L 2003 A Taxonomy of Recommender Agents on the Internet *Artificial Intelligence Review* **19**(4) 285-330
[22] Adomavicius G, TuzhilinA 2005 Toward the Next Generation of Rec-ommender Systems: *IEEE Transactions on Knowledge and Data Engineering* **17**(6) 734-49
[23] Beel J, Langer S, Genzmehr M, Gipp B 2013 A Comparative Analysis of Offline and Online Evaluations and Discussion of Research Paper Recommender System Evaluation *Proceedings of the Workshop on Reproducibility and Replication in Recommender Systems Evaluation (RepSys)* 7-14
[24] Gunawardana A, Shani G 2009 A survey of accuracy evaluation metrics of recommendation tasks *Journal of Machine Learning Research* **10** 2935-62

Authors

Zhihang Tang, born in August 1974, Hunan, China.

Current position, grades: teacher at the department of computer and communication, Hunan Institute of Engineering (Xiangtan, China) since 2003.
University studies: PhD degree at Donghua University China in 2009.
Scientific interests: intelligent decision and knowledge management
Publications: 30 papers

Zhonghua Wen, born in May 1966, Hunan, China.

Current position, grades: professor at the Hunan Institute of Engineering.
University studies: PhD degree in computer engineering at Zhongshan University, China.
Scientific interests: intelligent decision and knowledge management.
Publications: 50 papers.