

# Recommendation system based on collaborative filtering in RapidMiner

Zhihang Tang, Zhonghua Wen

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

---

## Abstract

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. The application of recommender systems outlined was just a small introduction to the possibilities of the extension. 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 recommender systems will assist you in reaching quality, informed decisions.

*Keywords:* recommender systems, collaborative-based systems, nearest neighbour

---

## 1 Introduction

Making choices is an integral part of everyday life, especially today when users are overwhelmed with information, from the Internet and television, to shelves in local stores and bookshops. We cope with this information overload by relying on daily recommendations from family, friends, authoritative users, or users who are simply willing to offer such recommendations. This is especially important when we lack information to make a rational decision, for example, choosing a hotel for vacations in a new city, selecting a new movie to watch, or choosing which new restaurant to visit.

Recommender systems [1] 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, such as item taxonomies, descriptions and other, and/or data on user experience with items of interest, for example, user choices, rankings, scores, tags, etc. 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 represent a broad and very active [2-4] field of research and were, from their origins in the 1990s [5], somewhat detached from the data mining field. This was mostly due to the specific form of data that recommender systems used. There are three basic types of recommender systems: collaborative filtering, content-based, and hybrid systems. Collaborative filtering recommender systems use social information, preferences, and experiences of other users in order to find users with similar taste. The assumption of these systems is that users of

similar tastes might enjoy and consume similar items. Content filtering recommender systems use the available structured and unstructured information on users or items to recommend further items of interest. They assume that the user might be interested in items similar to the ones in which an interest has already been displayed. Both of these systems have their advantages, which are additionally reinforced, or disadvantages, which are diminished with the usage of hybrid systems—systems which combine both the collaborative and the content-based recommendation approach.

Recommender systems are ubiquitous, and an average Internet user has almost certainly had experiences with them, intentionally or not. For example, the well-known Internet commerce, Amazon.com employs a recommender system that recommends products its users might be interested in, based on the shopping habits of other users. Social networking sites like Facebook or LinkedIn use recommender systems for recommending new friends to users based on their social network structure. The music website Last.fm uses a recommender system to recommend new music to a user, based on the listening habits of users with similar music taste. The Internet Movie Database (IMDb) recommends similar movies, based on the content and style of the movies user previously browsed. Streaming provider Netflix tries to predict new movies a user might be interested in based on his watching habits and movie ratings, compared to other users. These, and numerous other examples like Stumble Upon, Google AdSense, YouTube, etc., which differ in services provided, like audio, video, general item, social network, other Internet content, books, etc., demonstrate the importance of these systems.

Recommender systems facilitate making choices, improve user experience, and increase revenue, therefore should be easily accessible for deployment to interested

parties. This led us to write a paper on recommender systems in a clearly understood and easily applied way through RapidMiner. We believe that RapidMiner’s workflow approach entices systematic research and facilitates its implementation in combination with Rapid Analytics. The combination of research and production environment renders itself as an excellent environment for understanding recommender systems through practice. Throughout this paper, you will learn the basics of theory related to recommender systems, with a strong emphasis on practical implementation. This practical work will introduce you to all the necessary knowledge for rapid prototyping of recommender systems, thus enabling you to master them through application of your data. The implementation of

recommender systems in RapidMiner has been additionally simplified through the Recommender Extension.

## 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. The overview of operators supported in the Recommender Extension is given in the Table 1.

TABLE 1 Recommendation operators supported by the Recommender Extension.

Item Recommendation (IR)	Collaborative Filtering-based Item Recommendation	Item k-NN, Most Popular, Bayesian Personalized Ranking Matrix Factorization, User k-NN, Weighted Regularized Matrix Factorization, Random
	Attribute-based Item Recommendation	User Attribute k-NN, Item At-tribute k-NN
Item Rating Prediction (RP)	Collaborative Filtering-based Rating Prediction	Random, Global Average, User Item Baseline, User k-NN, Item k-NN, Slope One, Bi-Polar Slope One, Matrix Factorization, Biased Matrix Factorization, Factor Wise Matrix Factorization
	Attribute-based Rating Prediction	User Attribute k-NN, Item At-tribute k-NN

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 recommendation operators to input datasets, proper roles have to be set for these attributes, as seen in Table 1. Any additional attributes will not be considered. An example of an AML, and a related DAT file for item recommendation operators is given in Figure 1.

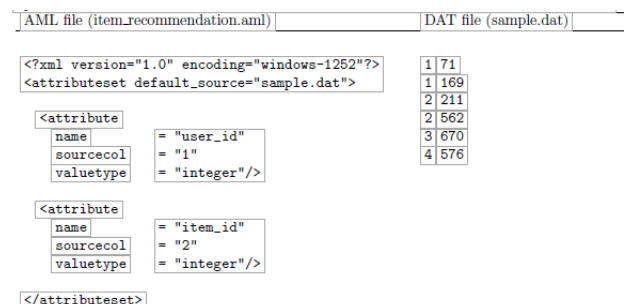


FIGURE 1 An example of an AML and a related DAT file for item recommendation operators

The recommender system datasets used throughout this paper consists of content and collaborative data. Content data was taken from the ALIDATA DISCOVERY (<http://102.alibaba.com/competition/addDiscovery/index.htm>)

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 Figure 2.

user_id	brand_id	type	visit_datetime
10944750	13451	0	6月4日
10944750	13451	2	6月4日
10944750	13451	2	6月4日
10944750	13451	0	6月4日
10944750	13451	0	6月4日
10944750	13451	0	6月4日
10944750	13451	0	6月4日
10944750	13451	0	6月4日
10944750	13451	0	6月4日
10944750	21110	0	6月7日
10944750	1131	0	7月23日
10944750	1131	0	7月23日
10944750	8689	0	5月2日
10944750	8689	2	5月2日
10944750	8689	2	5月2日
10944750	8689	0	5月2日

FIGURE 2 Initial data from Alibaba

### 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 preferences 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 Pearson 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.

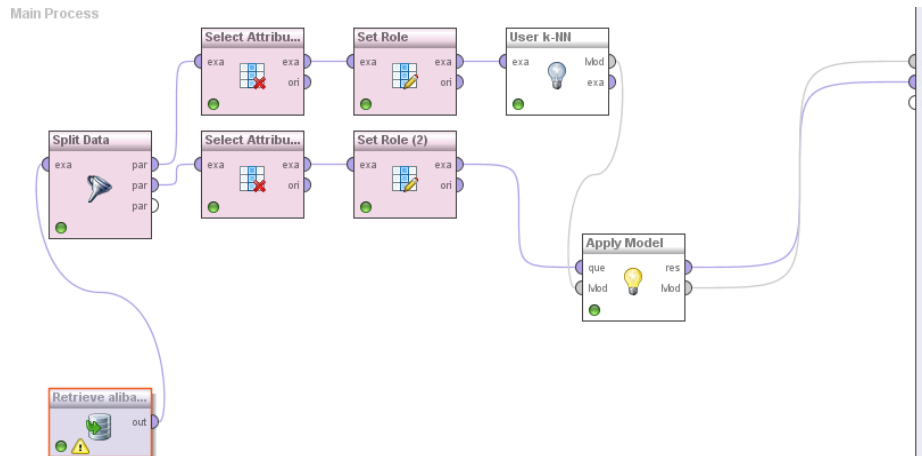


FIGURE 3 An example of an item recommendation workflow

The recommended results shown in Figure 4.

Row No.	user_id	item_id	rank
1	10944750	7868	1
2	10944750	14261	2
3	10944750	21110	3
4	10944750	11176	4
5	10944750	29099	5
6	10944750	14020	6
7	10944750	18805	7
8	10944750	4949	8
9	10944750	12977	9
10	10944750	4571	10
11	12028500	7868	1
12	12028500	14261	2
13	12028500	7105	3
14	12028500	21110	4
15	12028500	18730	5
16	12028500	14020	6
17	12028500	11176	7
18	12028500	23662	8
19	12028500	18805	9
20	12028500	29099	10
21	12154500	23662	1
22	12154500	18730	2
23	12154500	18805	3

FIGURE 4 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.

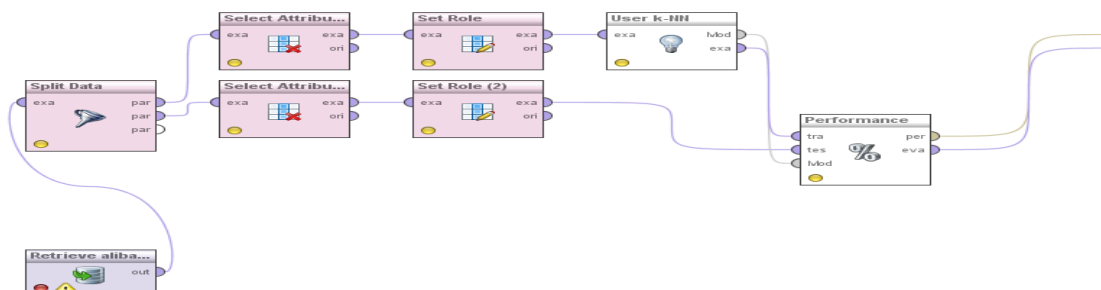


FIGURE 5 Measuring performance of a recommendation model.

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.

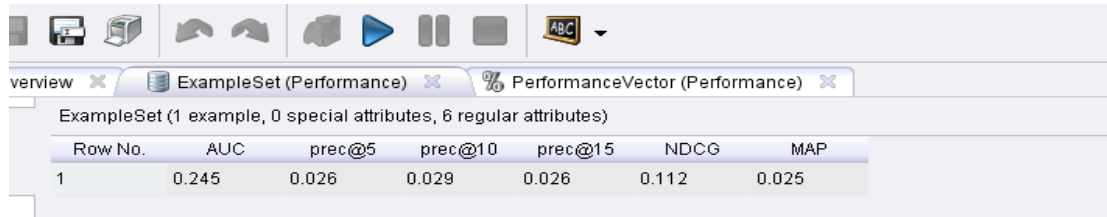


FIGURE 6 the performance of Recommender Systems

**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.2014GK3157).

**References**


[1] Resnick P, Varian H R 1997 Recommender systems *Communications of the ACM* **40**(3) 56-8  
 [2] Adomavicius G, Tuzhilin A 2005 *IEEE Transactions on Knowledge and Data Engineering* **17**(6) 734-49  
 [3] Lu L, Medo M, Zhang Y C, Yeung C H, Zhang Z K, Zhou T 2012 Recommender systems *Physics Reports* **519**(1) 1-49  
 [4] Rendle S, Tso-Sutter K, Huijsen W, Freudenthaler C, Gantner Z, Warmen C, Brussee R, Wibbels M 2011 Report on state of the art recommender algorithms *MyMedia public deliverable D4.1.2*. Technical report  
 [5] Goldberg D, Nichols D, Oki B M, Terry D 1992 Using collaborative filtering to weave an information tapestry *Communications of the ACM* **35**(12) 61-70  
 [6] 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 1966-5, 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.