# Collaborative Filtering Recommendation Algorithm Based on User Interest Clustering Postprint

**Authors:** Huang Xianying, Long Shuyan, Xie Jin

**Date:** 2018-05-24T00:00:00+00:00

## Abstract

To address the problems of traditional collaborative filtering algorithms ignoring that user interests stem from keywords and suffering from data sparsity, a collaborative filtering recommendation algorithm incorporating user interest degree clustering is proposed. Utilizing user ratings on items and extracting keywords from item attributes, a novel RF-IIF (rating frequency-inverse item frequency) algorithm is proposed. This algorithm derives user preferences for keywords based on both the target user' s rating frequency for a specific keyword and the rating frequency of that keyword across all users, thereby forming a user-keyword preference matrix upon which clustering is conducted. Subsequently, a logistic function is employed to obtain user interest degrees in items, thereby clarifying user preferences. Within the clusters, similar users to the target user are identified, and the top-N items favored by these neighbors are extracted for recommendation. Experimental results demonstrate that the algorithm consistently outperforms traditional algorithms in accuracy, exhibits relatively accurate judgment of user preferences, alleviates the data sparsity problem, and effectively improves both the accuracy and efficiency of recommendations.

## Full Text

### Collaborative Filtering Recommendation Algorithm Combined with User Interest Degree Clustering

**Huang Xianying, Long Shuyan†, Xie Jin**
(School of Computer Science & Engineering, Chongqing University of Technology, Chongqing 400054, China)

**Abstract**

Traditional collaborative filtering algorithms ignore the fact that user interests originate from keywords and suffer from data sparsity problems. To address these issues, this paper proposes a collaborative filtering recommendation algorithm combined with user interest degree clustering. By leveraging user ratings on items and extracting keywords from item attributes, we introduce a novel RF-IIF (Rating Frequency-Inverse Item Frequency) algorithm that captures user preferences for keywords based on both the target user' s rating frequency for a specific keyword and the keyword' s rating frequency across all users, forming a user-keyword preference matrix for clustering. We then employ a logistic function to derive user interest degrees in items, clarifying user preferences. Similar users are identified within clusters, and the top-N items preferred by neighbors are extracted for recommendation. Experimental results demonstrate that the proposed algorithm consistently outperforms traditional methods in accuracy, provides more precise judgment of user interests, alleviates data sparsity, and effectively improves both recommendation accuracy and efficiency.

**Keywords:** collaborative filtering; recommendation algorithm; user interest; K-means clustering

## 1 Introduction

Recommender systems (RS) are software tools or technical methods that suggest useful items to users [1]. Early recommendation systems provided only popular and generic content that failed to meet individual needs, leading to the development of personalized recommendation systems. The simplest approach to personalization involves analyzing user historical behavior data (including ratings and browsing history) to derive personalized preferences and predict the most suitable items.

Based on different user requirements, various recommendation methods have emerged, commonly including content-based methods, association rule-based methods, and collaborative filtering methods [2]. Among these, collaborative filtering is one of the most extensively studied and effective recommendation algorithms, widely adopted by major e-commerce platforms such as Taobao, JD.com, Amazon, and Dangdang.

Collaborative filtering algorithms are categorized into model-based and memory-based approaches [3-5]. Memory-based collaborative filtering is further divided into user-based and item-based methods. User-based collaborative filtering calculates similarity between the target user and other users, identifies those with similar interests, and generates recommendations based on their preferences. However, as the number of users and items grows, data sparsity [6], inaccurate similarity computation, and poor real-time performance become critical challenges affecting system performance.

To address rating prediction problems caused by data sparsity, researchers have introduced clustering algorithms. For instance, [7] proposed a recommendation algorithm combining item clustering with the Slope one scheme, aggregating items into clusters and applying Slope one to each cluster to predict ratings for unknown items. [8] introduced a collaborative filtering algorithm based on matrix clustering, applying the algorithm to submatrices after clustering user rating data, thereby improving recommendation accuracy. [9] presented an item clustering-based collaborative filtering algorithm that clusters items based on user ratings and searches for nearest neighbors within similar clusters, effectively enhancing real-time response speed. Although these improved algorithms mitigate traditional collaborative filtering problems to some extent, they still overlook the fundamental insight that user interest in items originates from keywords—users rate items because they are interested in specific keywords associated with those items.

To address these limitations, this paper proposes a user-keyword relationship-centric collaborative filtering algorithm. By combining the user-item matrix with the item-keyword matrix to construct a user-keyword matrix, we introduce the RF-IIF algorithm to compute user preferences for keywords, forming user preference vectors for clustering. We then utilize a logistic function to calculate user interest degrees in items, enabling efficient identification of similar neighbors within clusters. This approach enhances efficiency by leveraging user interest degree clustering to fully understand users and items, discover hidden relationships between users, enable recommendations for niche items, and improve algorithm real-time performance.

## 2 Collaborative Filtering Recommendation Algorithm

### 2.1 Basic Collaborative Filtering Process

The fundamental concept of collaborative filtering is to recommend items that interest users with similar tastes [10]. The algorithm consists of four steps: (1) collect user rating data and construct a user-item rating matrix; (2) compute user similarities using the rating matrix; (3) select top-N nearest neighbors based on similarity results; and (4) calculate predicted ratings for the target user based on neighbor ratings and generate recommendations. Collaborative filtering does not rely on explicit features or attributes of users and items but instead analyzes historical data that better reflects user preferences to uncover hidden relationships, yielding more personalized recommendations.

**Definition 1.** *User-Item Matrix.* All user rating records can be represented as a user-item rating matrix $R \in \mathbb{R}^{m \times n}$, where $U = \{u_1, u_2, \dots, u_m\}$ denotes the set of $m$ users and $I = \{I_1, I_2, \dots, I_n\}$ denotes the set of $n$ items, as shown in Equation (1). We use $r_{ij}$ to represent the rating of user $u_i$ on item $I_j$, with values ranging from 1 to 5, where higher scores indicate stronger preference.

$$R = \begin{bmatrix} r_{11} & r_{12} & \cdots & r_{1n} \\ r_{21} & r_{22} & \cdots & r_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ r_{m1} & r_{m2} & \cdots & r_{mn} \end{bmatrix}$$

**Similarity Computation.** To predict ratings for unrated items, collaborative filtering first identifies a set of similar neighbors, making user similarity calculation a critical component. Common similarity metrics include cosine similarity [11], Pearson correlation coefficient [12], and adjusted cosine similarity. Since cosine similarity ignores different rating scales across users, adjusted cosine similarity subtracts each user' s average rating before computing similarity:

$$sim(i,j) = \frac{\sum_{u \in U_{ij}} (r_{ui} - \bar{r}_u)(r_{uj} - \bar{r}_u)}{\sqrt{\sum_{u \in U_{ij}} (r_{ui} - \bar{r}_u)^2} \sqrt{\sum_{u \in U_{ij}} (r_{uj} - \bar{r}_u)^2}}$$

where $U_{ij}$ represents the set of users who rated both items $i$ and $j$, and $\bar{r}_u$ denotes user $u$' s average rating across all rated items.

After obtaining the nearest neighbor set $N_u$ for target user $u$, the predicted rating for item $i$ is computed using Equation (3):

$$P_{u,i} = \bar{r}_u + \frac{\sum_{v \in N_u} sim(u,v) \cdot (r_{v,i} - \bar{r}_v)}{\sum_{v \in N_u} |sim(u,v)|}$$

where $\bar{r}_u$ and $\bar{r}_v$ represent the average ratings of users $u$ and $v$, respectively. Based on these predictions, the top-N items with highest predicted ratings are recommended to the target user.

## 3 Proposed Algorithm: Collaborative Filtering with User Interest Degree Clustering

The proposed algorithm derives user preferences for keywords from rating records and item keywords, employs the RF-IIF algorithm to obtain keyword preference degrees, performs clustering on the resulting preference matrix, and uses a logistic function to compute user interest degrees for items. Finally, it identifies nearest neighbors within clusters to generate recommendations. The algorithm flow is illustrated in Figure 1 [Figure 1: see original paper].

### 3.1 User-Keyword Preference Matrix Computation

Users typically select items due to interest in specific keyword attributes, with varying interest degrees across different keywords. Therefore, we map user-item ratings onto corresponding keyword attributes, enabling similarity measurement

between users who may not have rated the same items but share interest in certain keywords. User ratings on keyword attributes reflect preferences, allowing us to derive a user-keyword matrix from the user-item rating matrix and item attribute matrix.

**Definition 2.** *Item-Keyword Matrix.* In recommender systems, items are described by various attributes. We represent the set of item keywords as $T = \{t_1, t_2, \ldots, t_q\}$, with the item-keyword matrix $W \in \mathbb{R}^{q \times n}$ defined as in Equation (4), where "1" indicates that an item possesses a keyword attribute and "0" indicates it does not. Each column vector represents an item' s keyword profile.

$$W = \begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1n} \\ w_{21} & w_{22} & \cdots & w_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ w_{q1} & w_{q2} & \cdots & w_{qn} \end{bmatrix}$$

Item attribute lists typically include properties such as genre, release date, and type, from which we extract keyword values. By mapping user-item ratings onto these keyword attributes, we generate a user-keyword rating matrix.

**Definition 3.** *User-Keyword Rating Matrix.* The user-keyword rating matrix $K \in \mathbb{R}^{m \times q}$ is derived from the user-item rating matrix $R$ and item-keyword matrix $W$ through matrix multiplication, as shown in Equation (5):

$$K = R \times W^T$$

In matrix $K$, each user has different ratings for different keywords, reflecting varying preferences. Specifically, user $u_i$' s preference for keyword $t_q$ increases with their rating frequency for that keyword and decreases with the keyword' s popularity. Based on this characteristic, we propose the RF-IIF algorithm to predict user preferences for keywords from rating frequencies and cluster users accordingly.

**Rating Frequency (RF).** RF represents the frequency with which a target user rates a particular keyword, calculated as in Equation (6):

$$RF_{iq} = \frac{w_{iq}}{\sum_{t \in T} w_{it}}$$

where $w_{iq}$ denotes the number of times user $u_i$ rated keyword $t_q$ in matrix $K$, $T$ is the set of all keywords, and $\sum_{t \in T} w_{it}$ represents the total rating count of user $u_i$.

**Inverse Item Frequency (IIF).** IIF represents the inverse of the number of users who rated a keyword, calculated as in Equation (7):

$$IIF_q = \log \frac{N}{N_q}$$

where $N$ is the total number of users in matrix $K$, and $N_q$ is the number of users who rated keyword $t_q$.

The preference degree of user $u_i$ for keyword $t_q$ is then given by Equation (8):

$$Pre_{iq} = RF_{iq} \times IIF_q$$

The preference degrees of user $u_i$ across all keywords form the preference vector $Pre_i = (Pre_{i1}, Pre_{i2}, \ldots, Pre_{ir})$, and the collection of all user preference vectors constitutes the user-keyword preference matrix $Pre$, as shown in Equation (9):

$$Pre = \begin{bmatrix} Pre_{11} & Pre_{12} & \cdots & Pre_{1r} \\ Pre_{21} & Pre_{22} & \cdots & Pre_{2r} \\ \vdots & \vdots & \ddots & \vdots \\ Pre_{m1} & Pre_{m2} & \cdots & Pre_{mr} \end{bmatrix}$$

The RF-IIF algorithm effectively distinguishes user preferences: popular keywords with high rating frequencies yield lower preference values, while niche keywords rated by fewer users receive higher preference values, indicating greater relative importance to those users.

### 3.2 User Clustering for Nearest Neighbor Identification

The user-keyword preference matrix, while clarifying preferences, remains high-dimensional and sparse. Therefore, we employ K-means clustering to partition users into smaller, similar clusters, enabling neighbor search within clusters to mitigate data sparsity.

K-means clustering measures similarity through distance calculation, grouping objects with distances below a threshold into clusters. Given its suitability for sparse data, we use cosine similarity for clustering:

$$sim(Pre_a, Pre_b) = \frac{\sum_{q=1}^{r} Pre_{aq} \cdot Pre_{bq}}{\sqrt{\sum_{q=1}^{r} Pre_{aq}^2} \sqrt{\sum_{q=1}^{r} Pre_{bq}^2}}$$

where $Pre_a$ and $Pre_b$ represent the preference vectors of users $u_a$ and $u_b$, respectively. The K-means algorithm groups users with similar keyword preferences into clusters.

### 3.3 Prediction and Recommendation

Keyword rating frequency indicates user preference—higher frequency suggests stronger interest. Compared to using raw ratings, rating frequency provides more accurate interest assessment. However, since rating frequencies vary significantly across keywords, we employ the logistic function (proposed by Pierre François Verhulst in 1844) to nonlinearly map rating frequencies to interest degrees. The logistic function, commonly used to model population growth and learning processes, is a smooth, continuous, strictly monotonic S-shaped function defined as:

$$f(x) = \frac{1}{1 + e^{-x}}$$

User interest in items grows nonlinearly with rating frequency. Under non-exclusive item conditions, higher rating frequency indicates greater interest. We thus use the logistic function to model the relationship between users and rating frequencies, deriving user interest degrees in items as in Equation (12):

$$H_{ij} = \frac{1}{1 + e^{-(R_{ij} - \bar{R})}}$$

where $H_{ij}$ represents user $u_i$' s interest degree in item $I_j$ (value in (0,1)), which increases monotonically and nonlinearly with rating count $R_{ij}$, and $\bar{R}$ is the average rating count across all items.

The interest degree values of user $u_i$ for all items in $I$ form the interest vector $H_i = (H_{i1}, H_{i2}, \ldots, H_{im})$. After obtaining user-item interest degrees, we identify Top-N neighbors similar to the target user within clusters using the adjusted cosine similarity formula (Equation 12):

$$S(u_a, u_b) = \frac{\sum_{i \in I_{ab}} (H_{ai} - \bar{H}_a)(H_{bi} - \bar{H}_b)}{\sqrt{\sum_{i \in I_{ab}} (H_{ai} - \bar{H}_a)^2} \sqrt{\sum_{i \in I_{ab}} (H_{bi} - \bar{H}_b)^2}}$$

where $I_{ab}$ is the set of items of common interest to users $u_a$ and $u_b$, and $\bar{H}_a$ and $\bar{H}_b$ represent their average interest degrees.

After identifying the Top-N users most similar to the target user to form the neighbor set $N_u$, we predict the target user' s preference for unrated items using Equation (13):

$$P_{u,i} = \bar{H}_u + \frac{\sum_{v \in N_u} S(u, v) \cdot (H_{v,i} - \bar{H}_v)}{\sum_{v \in N_u} |S(u, v)|}$$

where $\bar{H}_u$ is user $u$' s average interest degree across all rated keywords. Finally, the top-N items with highest predicted ratings are recommended to the target user.

## 4 Experimental Results and Analysis

### 4.1 Data Processing

We evaluate the algorithm using the MovieLens dataset [17] provided by GroupLens. The dataset contains 943 users, 1,682 items, and 100,000 rating records, with each user having at least 20 ratings on a scale of 1 (very poor) to 5 (very good). We randomly select five disjoint subsets for cross-validation, partitioning each into 80% training set and 20% test set. The user-item rating data format is shown in Figure 2 [Figure 2: see original paper], with columns representing: User ID | Item ID | Rating | Timestamp. Item attributes are shown in Figure 3 [Figure 3: see original paper], where 0 indicates absence and 1 indicates presence of a keyword attribute. The keywords include: Action, Adventure, Animation, Children' s, Comedy, Crime, Documentary, Drama, Fantasy, Film-Noir, Horror, Musical, Mystery, Romance, Sci-Fi, Thriller, War, Western.

### 4.2 Evaluation Metrics

We employ Mean Absolute Error (MAE) and Precision to measure recommendation quality. MAE is a standard metric for statistical accuracy and comparison, measuring the error between predicted and actual ratings. Smaller MAE values indicate higher accuracy. Given a set of predicted user interests $P = (p_1, p_2, \ldots, p_n)$ and actual interests $Q = (q_1, q_2, \ldots, q_n)$, MAE is calculated as:

$$MAE = \frac{1}{n} \sum_{i=1}^{n} |p_i - q_i|$$

Precision measures the probability that recommended items are truly interesting to the target user:

$$Precision = \frac{N_c}{N}$$

where $N$ is the total number of recommendations and $N_c$ is the number of correct recommendations.

**Experiment 1: Parameter Analysis.** We analyze the impact of the key parameter—the number of keywords—on recommendation effectiveness. Since the algorithm recommends based on user interest degrees across different keywords, we investigate how keyword count affects results. As shown in Figure 4 [Figure 4: see original paper], the algorithm performs best with 6 keywords. With too

few keywords, user similarity becomes difficult to compute, degrading performance; with too many, matrix sparsity increases, adversely affecting similarity calculation and raising error rates.

**Experiment 2: Algorithm Comparison.** Using the optimal keyword count of 6, we compare our algorithm with existing methods in terms of MAE and Precision. Figure 5 [Figure 5: see original paper] shows that as neighbor count increases, MAE decreases and stabilizes, with our algorithm achieving approximately 0.643 at 40 neighbors, outperforming the logistic clustering-based algorithm at 0.676. Figure 6 [Figure 6: see original paper] demonstrates that Precision increases and stabilizes with neighbor count, reaching approximately 0.2487 at 20 neighbors, surpassing the algorithm in [15] at 0.2442. Our algorithm consistently outperforms both traditional and recent methods.

**Efficiency Analysis.** Figure 7 [Figure 7: see original paper] compares recommendation generation time with and without clustering. While both approaches exhibit increasing time with neighbor count, clustering significantly reduces runtime, demonstrating substantially improved efficiency.

## 5 Conclusion

This paper introduces a logistic function to model the nonlinear relationship between user rating frequency and interest degree, computing user interest in item keywords to form a user-keyword interest matrix and subsequently a user-item interest degree matrix. By performing user clustering on this new matrix and searching for nearest neighbors within the target user' s cluster, we narrow the search scope and enhance algorithm efficiency. Experiments demonstrate that the proposed algorithm effectively leverages user interest degrees in item keywords to identify nearest neighbors and improve recommendation accuracy. However, the algorithm does not fully consider temporal dynamics in user interests. Future work will incorporate timestamp information and forgetting mechanisms to further improve recommendation performance.

## References

[1] 项亮. 推荐系统实践 [M]. 北京: 人民邮电出版社, 2012. (Xiang Liang. Recommended system practice [M]. Beijing: People' s Posts and Telecommunications Press, 2012.)

[2] Herlocker L, Konstan A, BorchersS A, et al. An algorithmic framework for performing collabora-tive filtering [C]// Proc of International ACM SIGIR Conference on Research and Development in Information Retrieval. 1999.

[3] Dehghani Z, Reza S, Salwah S, et al. A systematic review of scholar context-aware recommender systems [J]. Expert Syst. Appl. 2015 (42): 1743.

[4] Adomavicius G, Tuzhilin A. Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions [J]. IEEE Trans

on Knowledge & Data Engineering, 2005, 17 (6): 734-749.

[5] Mnih A, Salakhutdinov R. Probabilistic matrix factorization [C]// Advances in Neural Information Processing Systems. 2007: 1257.

[6] Paterek A. Improving regularized singular value decomposition for collaborative filtering [C]// Proc of KDD Cup Workshop at SIGKDD&the 13th ACM Int Conf on Knowledge Discovery and Data Mining. 2007: 39.

[7] You Haipeng, Li Hui, Wang Yunmin, et al. An improved collaborative filtering recommendation algorithm combining item clustering and slope one scheme [C]. Lecture Notes in Engineering & Computer Science, vol 2215. 2015: 313-316.

[8] 高凤荣, 邢春晓, 杜小勇, 等. 基于矩阵聚类的协作过滤算法 [J]. 华中科技大学学报: 自然科学版, 2005, 33 (S1): 257-260. (Gao Fengrong, Xing Chunxiao, Du Xiaoyong, Wang Shan. A collaborative filtering algorithm based on matrix clustering [J]. Journal of Huazhong University of Science and Technology (Natural Science Edition), 2005, 33 (S1): 257-260.)

[9] 兰艳, 曹芳芳. 面向电影推荐的时间加权协同过滤算法的研究 [J]. 计算机科学, 2017, 44 (4): 295-301. (Lan Yan, Cao Fangfang. A temporal weighted collaborative filtering algorithm for movie recommendation [J]. Computer Science, 2017, 44 (4): 295-301.)

[10] 范波, 程久军. 用户间多相似度协同过滤推荐算法 [J]. 计算机科学, 2012, 39 (1): 23-26. (Fan Bo, Cheng Jiujun. Among multiple users similarity collaborative filtering algorithm [J]. Computer Science, 2012, 01: 23-26.)

[11] Zhao Z D, Shang M S. User-based collaborative filtering recommendation algorithms on Hadoop [C]// Proc of the 3rd International Conference on Knowledge Discovery and Data Mining. 2010: 478-481.

[12] Herlocker J L. Evaluating collaborative filtering recommender systems [J]. Acm Trans on Information Systems, 2004, 22 (1): 5-53.

[13] 黄震华, 张佳雯, 田春岐, 等. 基于排序学习的推荐算法研究综述 [J]. 软件学报, 2016, 27 (3): 691-713. (Huang Zhenhua, Zhang Jiawen, Tian Chunqi, et al. A survey of recommendation algorithms based on ranking learning. [J]. Software Journal, 2016, 27 (3): 691-713.)

[14] 张松, 张琳, 王汝传. 基于用户限制聚类的协同过滤推荐算法 [J]. 南京邮电大学学报: 自然科学版, 2017. 37 (3): 93-99. (Zhang Song, Zhang Lin, Wang Ruchuan. Collaborative filtering recommendation algorithm based on user restricted clustering [J]. Nanjing University of Posts and Telecommunications: Natural Science Edition, 2017, 37 (3): 93-99.)

[15] 朱东郡, 李敬兆, 谭大禹, 等. 基于标签聚类和兴趣划分的协同过滤推荐算法 [J]. 计算机工程, 2017, 43 (11): 146-151. (Zhu Dongjun, Li Jingzhao, Tan Dayu, et al. Collaborative filtering recommendation algorithm based on tag clustering and interest partition [J]. Computer Engineering, 2017. 43 (14): 146.)

[16] Forsati R, Barjasteh I, Masrour F, et al. PushTrust: an efficient recommendation algorithm by leveraging trust and distrust relations [C]// Proc of Conference on Recommender Systems. 2015: 51-58.

[17] MovieLens_100K [DB/OL]. https://grouplens.org/datasets/movielens/.

*Note: Figure translations are in progress. See original paper for figures.*