Information Processing and Management 58 (2021) 102531

Contents lists available at ScienceDirect

Information Processing and Management

journal homepage: www.elsevier.com/locate/infoproman

A movie recommendation method based on users' positive and negative profiles

Yen-Liang Chen [*], Yi-Hsin Yeh, Man-Rong Ma

Department of Information Management, School of Management, National Central University, Chung-Li, Taiwan 32001, R.O.C

A R T I C L E I N F O

Keywords:
Recommendation
Collaborative filtering
User profile
Hybrid recommendation

A B S T R A C T

In the traditional content-based recommendation method, we usually use the movies users watched before or rated to represent their profile. However, there are many movies that users have never seen or rated. For an unrated movie, there are two possibilities: maybe the user likes it or does not like it. In this paper, we first focus on how to identify users' preferences for movies by using a collaborative filtering algorithm to predict the users' movie ratings. We can then create two movie lists for each user, where one is the movies the user likes (with higher predicting or true ratings), and the other is the movies the user does not like (with lower predicting or true ratings). Based on these two movie lists, we establish a user positive profile and a user negative profile. Therefore, our algorithm will recommend to users movies that are most similar to their positive profile and most different from their negative profile. Finally, our experiments show that our method can improve the MAE index of the traditional collaborative filtering method by 12.54%, the MAPE index by 17.68%, and the F1 index by 10.16%.

1. Introduction

Movies have become a daily leisure activity for many people. Since the Lumière Brothers released the first film in 1895, the vigorous development and creation of the film industry in various countries has made films an indispensable leisure activity for the general public. However, nowadays, after more than 100 years of film development, with the rapid development of technology, the traditional screen viewing that originally required audiences to enter a theater has gradually transformed into an online streaming platform that can be watched immediately at home. This online streaming media platform combines novel technologies, focuses on the trends of the big data era, and has improved and established various functions. These functions not only allow users to rate movies, but also allow friends to exchange movie experiences. In addition, the platform uses score data from many users to build a recommendation system to predict the preferred recommendations of individual users.

The recommendation system is widely used in online streaming platforms. The most basic and extensive recommendation methods can be classified as the following categories (He, Parra & Verbert, 2016; Lu, Wu, Mao, Wang & Zhang, 2015; Shah, Gaudani & Balani, 2016; Thorat et al., 2015): (1) Collaborative Filtering, (2) Content-based filtering, and (3) Hybrid Recommendation.

Collaborative filtering aims to use the ratings of similar users or items to predict the user's rating on the next item. On the other hand, content-based filtering (Bergamaschi & Po, 2015) is mainly based on information retrieval and information filtering. It uses the user profile and product content as data, and compares the information contained in the product with the user profile. Content-based

* Corresponding author.
E-mail address: ylchen@mgt.ncu.edu.tw (Y.-L. Chen).

https://doi.org/10.1016/j.ipm.2021.102531
Received 23 September 2020; Received in revised form 20 December 2020; Accepted 22 January 2021
Available online 15 February 2021
0306-4573/© 2021 Elsevier Ltd. All rights reserved.

methods usually use products that users have purchased in the past to build user profiles. The intuition underpinning this method is that what the users buy represents what they like. Therefore, we can use the features extracted from the purchased products (such as the product's theme, attributes or categories) to represent the user's interests and preferences. In addition, user profiles can also reflect users' hidden interests through the similarity and relevance of product topics. Content-based filtering will extract product characteristics to represent the target user's preference profile. It then compares the user profile with the characteristics of the product, and recommends the most relevant product set to the user.

When the content-based filtering method is applied to movie recommendation, the method creates a profile representing the user's preference based on the movies previously watched by the user, and then generates the top K movie recommendations from the movies that the user has not yet watched. The criteria for selecting these recommended movies are based on the highest similarity between the content of these movies and the user's interest profile. However, the disadvantage of this method is that it ignores information about movies that the user has never watched before. This missing information has two possible meanings: either the user likes them or the user does not like them. If it is the latter case, the negative information can help the recommendation system make more accurate recommendations by deleting those negative movies. Unfortunately, this negative information has not been used to filter out movies that users do not like. Therefore, in the past, movies recommended by CB to users may hide movies that users do not like. In this paper, we address this issue by first predicting the rating of each user for each item based on the CF algorithm. If the predicted rating value is high, the target user may like the item. On the other hand, if the predicted rating value is low, the target user may not like the item. A positive profile is established based on those items with a higher predicted rating, and a negative profile is established based on those items with a lower predicted rating. If an item has a high degree of similarity with the positive profile of the target user, but a low degree of similarity with the negative profile of the target user, then it may be ideal for recommendation to the target user.

When we are faced with an application situation where we have both product user rating information and product content information, the natural way is to use a hybrid method that combines collaborative filtering methods and content-based methods. This paper uses a hybrid approach to recommend movies that users like. However, there is a big difference between our method and the previous research, that is, we not only use the user's positive preference profile, but we also use the user's negative preference profile to keep favorite movies and delete disliked movies. As a result, our method can improve the accuracy of movie recommendation compared with the traditional CB method.

In order to construct our hybrid method, we first apply the CF method to obtain each user's prediction score for all items. These prediction scores are obtained based on the correlation of the users' scoring behaviors. However, the main disadvantage of obtaining scores through CF is that user characteristics and movie characteristics are not considered when generating predictions. In order to further improve the prediction results, we divide each user's movie set into two groups. Each user's first movie set contains movies for which the user has a higher predicted score. In other words, it represents the user's positive preference profile. Similarly, each user's second movie set contains movies for which the user has a lower predicted score, which represents the user's negative preference profile. Based on these two profiles, we will recommend movies that are similar to the user's positive profile but different from the user's negative profile.

The advantages of our method include the following: (1) we consider the relevance of the users' scoring behavior; (2) we consider user characteristics and movie characteristics; and (3) we use users' negative profiles to further enhance the recommendation results.

The main contributions of this research are:

- We propose a new hybrid method that combines CF and CB.
- Our method can build a positive profile and a negative profile for each user.
- This is the first paper that considers negative user profiles in CB, so we can delete movies negatively related to user preferences from the recommendation set.
- Our experimental results show that considering negative profiles can improve the traditional CB that only considers positive profiles. Based on this finding, all past studies in CB can try to further improve their recommendation results by using negative profiles.

The rest of this paper is organized as follows. In Section 2, we state our research objective. Section 3 is a review of related work. Section 4 introduces a hybrid movie recommendation method based on the user's positive and negative profiles. Section 5 contains a series of experiments to prove the effectiveness of our recommendation algorithm. Section 6 is the conclusion of this paper. Finally, in Section 7 we discuss the implications of this research and future work.

2. Research objective

Content-based filtering uses user profiles and product content as data, and compares the information contained in the product with the user profile. Content-based methods usually use products purchased in the past to build user profiles. The intuition underpinning the previous approach is that what users buy represents what they like. However, for a certain product, the user may like or dislike it. If we can discern the relationship between users and unrated products, then this negative information will further improve CB performance. Therefore, we pose the following research questions:

- How can an item be identified as a negative item?
- How can positive and negative information be used to build user profiles to more accurately describe user preferences?
- How can both positive and negative profiles be used to improve the traditional CB algorithm that only uses positive profiles?

3. Related work

The recommendation system is an information filtering mechanism, mainly used to reduce the extra cost involved in the process of searching for information. Based on the user's preferences, interests, behaviors, or needs, it recommends information, services, or products that the user potentially needs. The most popular recommendation algorithms can be classified as three main categories: content-based, collaborative filtering and hybrid approaches.

The process of content-based filtering is generally divided into the following three steps:

1 Product representation: Extract some features from each product to represent that product. If it is a movie, it can be distinguished according to the characteristics of the movie theme, genre, actors, director, movie distributor, star rating, keywords, user comments, and so on, and use the content characteristics to represent the movie.
2 Profile learning: Use the characteristics of those products that the user liked or purchased in the past to represent the user's preference profile. If the user is a movie viewer, we can create a profile representing the user by integrating or averaging the characteristics of the movies the user watched or liked in the past.
3 Recommendation generation: Compare the characteristics of user profiles and candidate products, and recommend a set of products most relevant to the user. For movie viewers, we can compare their profile with movie characteristics and find the most relevant movies as recommendations.

When processing the features of movie content, some methods are usually used to convert these features into vector form for easy comparison. In the following content-based movie recommendation systems, different vector models are used to represent the characteristics of movies and users. For example, some research (Walek & Fojtik, 2020) has used the VSM (Vector Space Model) model to transform movie content, some (Yin & Zhang, 2018) has used the TFIDF (item frequency-inverse document frequency) model, some (Bansal, Gupta & Arora, 2016; Bergamaschi & Po, 2015; Bougiatiotis & Giannakopoulos, 2018; Mishra, Kumar & Bhasker, 2015) has used the LSI (Latent Semantic Index) model, some (Bergamaschi & Po, 2015) has used the LDA (Latent Dirichlet distribution) model, some (Zhang, Ji, Li & Ye, 2016) has used the TWH (three wings and conference hall) model, and some research (Indira & Kavithadevi, 2019) has used the PCA (Principal Component Analysis) model. Each of these different models has its own advantages and disadvantages. For a systematic study and comparison of the pros and cons of each model, please refer to the work of Alghamdi and Alfalqi (2015), Jelodar et al. (2019) and Sharma, Kumar and Chand (2017).

The main task of collaborative filtering is to use the users' existing rating values on the product to predict the product that the target user may like. This method can be further divided into three approaches (Jiang, Duan, Jain, Liu & Liang, 2015; Ramos, Boratto & Caleiro, 2020; Xu, 2018; Ha and Lee, 2017; Najafabadi et al., 2019). The first method is user-based CF, which uses similar users' ratings of products to predict the target users' ratings of products. The second method is item-based CF, which uses users' ratings of similar items to predict users' ratings of target products. The third method is the matrix decomposition method, which decomposes the user product matrix into the multiplication of two matrices (Koren, Bell & Volinsky, 2009). The first matrix represents the relationship between users and hidden topics, and the second matrix represents the relationship between products and hidden topics. Since the original matrix is sparse, the composition matrix can be learned from the gradient descent method to minimize the difference between the actual rating and the predicted rating. After obtaining the two constituent matrices, we can multiply them to obtain any user's rating prediction for any item.

The public dataset, called MovieLens (Harper & Konstan, 2015), is most likely the most popular and widely used dataset in CF. It has long been the gold standard dataset in CF. Therefore, we can say that all pure CF algorithms can be regarded as an application of movie recommendation (Ha & Lee, 2017; Huang, Chen & Chen, 2016; Ponnam, Punyasamudram, Nallagulla & Yellamati, 2016; Sanz-Cruzado, Castells, Macdonald & Ounis, 2020; Subramaniyaswamy, Logesh, Chandrashekhar, Challa & Vijayakumar, 2017; Zhang, Kudo, Murai & Ren, 2020), even though the term "movie" does not appear in the titles of these studies.

Hybrid recommendation mainly combines collaborative filtering and content-based recommendation methods. This mixing method can be achieved in many ways. For example, we can obtain predictions based on content filtering and predictions based on collaborative filtering separately, and then combine them in a collaborative manner. Alternatively, we can unify these two methods into one model (Lu et al., 2015) to obtain a complete evaluation of the recommender system, and finally use it in a mixed manner. (Isinkaye et al., 2015) pointed out that there are seven main strategies for building a hybrid model, described as follows.

- Weighted: numerically combine the scores of different recommended components.
- Switching: The system will select among the recommended components and apply the selected one.
- Mixed: Recommendations from different recommenders are presented together.
- Feature Combination: combine features from different knowledge sources and give a single recommendation algorithm.
- Feature Augmentation: A recommendation technique is used to calculate a feature or a group of features, and then used as part of the next technology input.
- Cascade: In this technique, one recommendation technique is employed first to produce a coarse ranking of candidates and a second technique refines the recommendation from among the candidate set.
- Meta-level: apply a recommendation technique and generate a certain model, and then use it as input for the next technique.

In the past, many movie recommendation algorithms were developed based on hybrid technology. In other words, their model uses not only the ratings data in CF, but also the extracted features of users and movies in CB. Chikhaoui, Chiazzaro and Wang (2011) proposed an approach that combines collaborative filtering, content-based and demographic filtering approaches to develop a recommender system for predicting movie ratings in a dynamic way. Wei, Zheng, Chen and Chen (2016) proposed a hybrid movie recommendation method using tags and ratings. Their model aims to improve the fusion ability by applying the potential impact of the two aspects generated by users. One aspect is the personalized scoring system and singular value decomposition algorithm, and the other is the tag annotation system and topic model. In Soni, Goyal, Vadera and More (2017), the focus is on combining content-based algorithms, user-based collaborative filtering algorithms, and review-based text mining algorithms in the application of customized movie recommendation systems. Here movies are recommended based on ratings explicitly provided by the user, and according to the ratings and reviews of movies provided by other users as well. In Yang, Chen, Liu, Liu and Geng (2018), they consider social behaviors such as tags, comments, and likes as important features of movie content. Therefore, they combine these behavioral characteristics with traditional content characteristics. Based on this, they proposed a hybrid recommendation method that combines social behavior, the genres of movies and existing collaborative filtering algorithms to perform movie recommendation. Walek and Fojtik (2020) propose a holistic hybrid recommendation system called Predictory to recommend movies, which combines a recommendation module composed of a collaborative filtering system (using the SVD algorithm), a content-based system, and a fuzzy expert system.

Finally, in the present study we also used a hybrid approach to build the recommendation model. The combination strategy we adopted was Cascade. This is because we first use CF technology to predict user ratings. Then, we establish positive and negative user profiles and obtain ranking results through CB technology.

It can be seen from the above research that the combination of CF and CB is the main trend to improve the performance of movie recommendation. The disadvantage of CB is that its recommended content is usually limited to movies that are similar to previous movies that the audience has watched before. If there are different types of movies, even if the user likes them, they will not be able to generate such recommendations. On the other hand, the disadvantage of CF is that it ignores the characteristics of movies and users when making recommendations. In addition, the data sparsity and cold start of the CF model are also troublesome. For these reasons, the hybrid method is the most promising method in movie recommendation.

In this paper, we further point out the shortcomings of the previous CB method. That is, previous studies usually build user profiles based on movies or products that users have watched or purchased before. Our hypothesis for this research is that if we can create two types of profiles for each user, one for favorite movies and the other for disliked movies, using these two profiles can help us find a better movie collection to meet users' needs.

To divide the profile into two parts, we need to use user ratings for movies. Those with higher ratings can be used to build a positive profile, and those with lower ratings can be used to build a negative profile. However, since the rating matrix is usually sparse, there are few movies rated by each user. It is very unreliable to create positive and negative profiles for each user with very little scoring data, so this is not a good method. Therefore, our method is to use the CF model to generate predicted ratings for movies that have no known ratings. As a result, each user has a sufficient number of movie ratings to establish a positive and a negative profile.

Our approach can avoid the problem that CB recommendations are limited to products or movies that have been seen or purchased before. This is because, after applying CF to generate predicted ratings of other movies they have never watched, each user now has ratings values for all movies. Therefore, we have more positive movies to build a positive profile, and more negative movies to build a negative profile. In doing so, the recommendations of our model are not limited by the user's existing interests. On the contrary, our model can expand the existing interests of users.

Since the use of user negative profiles to improve recommendation performance is a brand new idea and has not appeared in previous CB studies, we believe that by embedding this new idea into the previous algorithm, we can completely change the traditional recommendation algorithm.

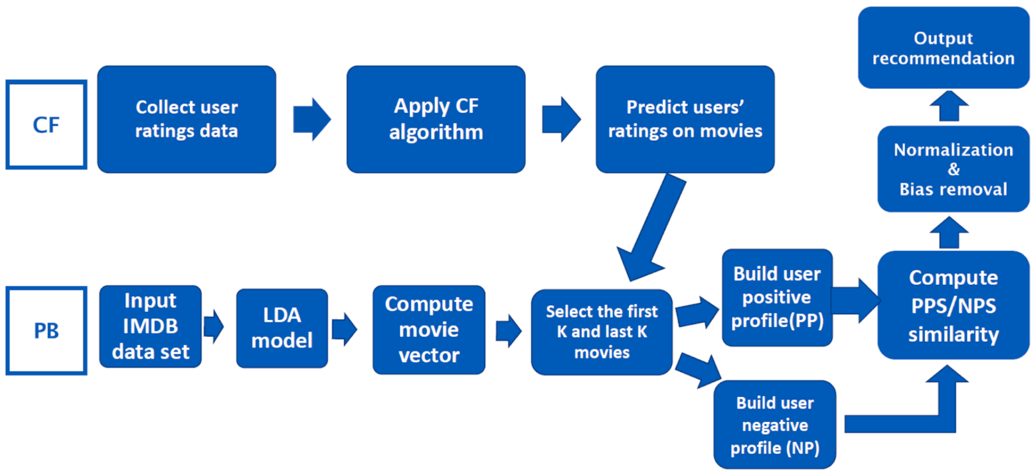
CF
Collect user ratings data
Apply CF algorithm
Predict users' ratings on movies
Output recommendation
Normalization & Bias removal
PB
Input IMDB data set
LDA model
Compute movie vector
Select the first K and last K movies
Build user positive profile(PP)
Build user negative profile (NP)
Compute PPS/NPS similarity

Fig. 1. The framework of recommendation.

4. Recommendation algorithm

In this section, we will outline our recommendation system. Fig. 1 shows the architecture of the entire system. In the system architecture, we have designed two main modules, namely the Collaborative-Filtering module (CF) and the Profile-Based module (PB). The CF module first collects actual rating data from the MovieLens dataset. Since the entire user-movie rating matrix is very sparse, we apply the user-based collaborative filtering algorithm (Herlocker, Konstan, Borchers & Riedl, 2015) to obtain the predicted rating values for all remaining movies which have no actual rating values. After the CF module is completed, each user has a rating value for each movie. In other words, the user-movie matrix is now completely filled. The value can be the real score given by the user or the predicted value calculated by the CF algorithm.

In the PB module, we first collect the characteristics of each movie from the hetrec2011-movielens-2k dataset (https://grouplens.org/datasets/hetrec-2011/) and the MovieLens 1 m dataset (http://www.grouplens.org). Then, by combining the features stored in these two datasets, each movie is converted into a three-segment movie vector using the LDA technology. These three segments respectively denote movie title, director and genre.

For each user, we must establish a positive profile and a negative profile. Since each user has a rating value for each movie after CF module processing, we can select the movies with the K largest ratings to build the positive profile, and select the movies with the K smallest ratings to build the negative profile. By averaging all the K movie vectors in the positive profile, the user's positive profile (PP) can be established. Similarly, by averaging all the K movie vectors in the negative profile, the user's negative profile (NP) can be established. Notice that now every user is represented by two vectors, where one is the PP vector and the other is the NP vector.

After obtaining the positive and negative profiles of each user, we must predict the rating that the user will give to each movie. Let movie_i denote the movie vector of movie i, let PP_u denote the positive profile vector of user u, and let NP_u denote the negative profile vector of user u. Then, the following steps are executed.

1 Compute the similarity between PP_u and movie_i, compute the similarity between NP_u and movie_i, and divide the PP similarity by the NP similarity. Let $\text{Sim}_{u,i}$ denote the result.
2 For user u, we normalize the values of all $\text{Sim}_{u,i}$ into the range of [1,5] by min-max normalization. Let $\text{rate}_{u,i}$ denote the normalized result.
3 Let $\text{Avg}_u = \text{Avg}_{\forall i}(\text{rate}_{u,i})$, and T_u be the average of all actual rating data of user u. Then, the bias of user u, b_u, can be expressed as $b_u = \text{Avg}_u - T_u$.
4 Finally, the final predicting rating of user u for movie i is defined as $\text{rate}_{u,i} - b_u$.

In step 1, the PP /NP similarity of the user and the movie is calculated by dividing the PP similarity by the NP similarity. This equation means that we want the PP similarity to be as large as possible, while at the same time we want the NP similarity to be as small as possible. In other words, if the movie is similar to the user's positive profile but different from the user's negative profile, the movie will have a higher score. In step 2, we use the min-max normalization conversion method to convert all PP/NP similarity values of the same user into values in the range of [1, 5]. Since the normalization process is performed for each individual user, it does not consider the different scoring behaviors of different users, so we must eliminate this potential bias. To this end, step 4 further removes the bias from each prediction rating.

In the following sections, we will introduce these functions in more detail. First, we will introduce the CF module. Then, we will introduce how to represent each movie as a three-segment vector. Next, we will introduce how to use CF predicted values and movie vectors to generate positive and negative profiles for each user. Finally, we will introduce how to calculate the PP/NP similarity between each user and the movie.

4.1. The CF module

In the CF stage, the user-based method is used as the main core technology. In the collaborative filtering stage, the following three steps are performed to compute the predicting rating of user u on item i:

1 Compute the cosine similarities of all users with the active user u.
2 Select a subset of users with highest similarity as a set of neighbors.
3 Use the deviation from mean formula to compute the predicting rating of user u for item i.

The first step is to compute the cosine similarity between each user v with the target user u. Let S denote the set of items that are commonly rated by users v and u, let u_i denote the rating user u gives to item i, let v_i denote the rating user v gives to item i, let $\bar{u}$ denote the average of all of user u's ratings, and let $\bar{v}$ denote the average of all of user v's ratings. The cosine similarity $r_{u,v}$ can be obtained by the formula $\frac{\sum_{i \in S} u_i \times v_i}{\sqrt{\sum_{i \in S} u_i^2} \times \sqrt{\sum_{i \in S} v_i^2}}$. In the second step, we will select the users with the highest coefficients as neighbors. In our paper, we select at most 30 similar users into the neighbor set. In the case of 30 similar users not being available, we select all of them. The third step is to compute the predicting rating of user u for item i based on the deviation from the mean formula. Let N denote the neighbor set. Then the formula to predict the rating of user u on item i can be written as $\bar{u} + \frac{\sum_{v \in N} (v_i - \bar{v}) r_{u,v}}{\sum_{v \in N} r_{u,v}}$.

4.2. Movie representation

In this step, we transform each movie according to its content into a three-segment vector. We use three attributes of movies to represent the movie content. They are movie title, director and genre. Thus, the problem we face now is that we must find a way to represent the three token streams so that we can determine the similarities between the movie content. Traditionally, a popular method is to use the TFIDF format to represent these three token streams. However, using TFIDF will result in sparse vectors leaving many zeros in many elements. In addition, the elements in the vector are interdependent, which may lead to bias in the comparison results. For these reasons, we chose to use the LDA (Latent Dirichlet Allocation) model to represent the data.

LDA is a generative statistical model that allows a set of observations to be explained by the unobserved group, which explains why certain data are similar. For example, if the observation is those words which appear in documents, it is assumed that each document is a mixture of a small number of topics, and the occurrence of each word can be attributed to one of the themes of the document (Jelodar et al., 2019). The advantages of using the LDA model to represent data streams include the fact that it is fast, intuitive, compact, easy to understand, and can be used to predict topics in documents that have not been read.

In this study, we set the number of topics to nt. By doing this, the movie is transformed into a three-segment vector, where each segment is an LDA distribution with *nt* topics.

4.3. Building the user profile

In this study, each user is represented by two profiles. The positive profile represents the movies that the user likes, while the negative profile represents the movies that the user does not like. In the first step of the algorithm, we obtain the predicting score of each movie for each user through CF. Let u denote a user, let $\mathrm{movie_i}$ denote the three-segment vector of movie i, and let score(u, i) denote the predicting score of user u for movie i. Assume that $\mathrm{LK_u}$ denotes the set of the largest K score(u, i) of user u, and $\mathrm{SK_u}$ denotes the set of the smallest K score(u, i) of user u. Then the positive profile $\mathrm{PP_u}$ of user u is obtained by averaging all three-segment vectors for the movies in $\mathrm{LK_u}$. That is, $\mathrm{PP_u}=\frac{\sum_{i\in \mathrm{LK_u}} \mathrm{movie_i}}{K}$. Likewise, the negative profile $\mathrm{NP_u}$ of user u is obtained by averaging all three-segment vectors for the movies in $\mathrm{SK_u}$. That is, $\mathrm{NP_u}=\frac{\sum_{i\in \mathrm{SK_u}} \mathrm{movie_i}}{K}$. Having obtained these two profiles, we can compute the similarities of user u with all movies.

Example 1. Suppose we have 10 movies, and we want to build the positive and negative profiles by setting $K = 3$. For the active user u, it is assumed that their CF ratings for movie 1 to movie 10 are 2.5, 3.4, 3.7, 4.3, 2.6, 4.7, 1.6, 3.8, 1.9, and 4.5, respectively. Then the three movies with the highest ratings are movie 6, movie 10 and movie 4, while the three movies with the lowest ratings are movie 7, movie 9 and movie 1. In other words, we have $\mathrm{LK_u}$={6, 10, 4} and $\mathrm{SK_u}$={7, 9, 1}. Therefore, the positive profile $\mathrm{PP_u}$ can be obtained by averaging the movie vectors 6, 10 and 4 in $\mathrm{LK_u}$, and the negative profile $\mathrm{NP_u}$ can be obtained by averaging the movie vectors 7, 9 and 1 in $\mathrm{SK_u}$.

4.4. Similarity computation

The purpose of this section is to calculate the similarity between the user and the movie so that we can recommend the movie the user might like. Similarity calculation includes three steps as follows.

1 Compute the similarity PPS(u, i) between $\mathrm{PP_u}$ and each movie vector $\mathrm{movie_i}$.
2 Compute the similarity NPS(u, i) between $\mathrm{NP_u}$ and each movie vector $\mathrm{movie_i}$.
3 Compute the similarity $\mathrm{Sim_{u,i}}$ by dividing PPS(u, i) by NPS(u, i).

Example 2. Assume that the similarity between $\mathrm{PP_u}$ and $\mathrm{movie_i}$ is 0.9, that is, PPS(u, i)=0.9. Suppose further that the similarity between $\mathrm{NP_u}$ and $\mathrm{movie_i}$ is 0.1, that is, NPS(u, i)=0.1. Then, we have $\mathrm{Sim}_{\mathrm{u},i} = 0.9/0.1 = 9$. This situation indicates that movie i is very similar to the user's positive profile but very dissimilar from the user's negative file. Therefore, we recommend item i to user u.

Here, we only show how to complete step 1. The remaining steps are similar or simple, so we will omit them. In step 1, vectors $\mathrm{PP_u}$ and $\mathrm{movie_i}$ contain three segments. In addition, each segment is an LDA probability distribution with *nt* topics. Therefore, the basic problem we must handle is how to compute the similarity between two LDA distributions. That is, how to compute the similarity of the k-th segment of $\mathrm{PP_u}$ and the k-th segment of $\mathrm{movie_i}$.

The following formula was used for measuring the similarity of the two LDA distributions in the work of Tong and Zhang (2016).

$$JSD(P \parallel Q) = \frac{1}{2}D_{KL}(P \parallel M) + \frac{1}{2}D_{KL}(Q \parallel M) \tag{1}$$

$$D_{KL}(P \parallel Q) = \sum_r P(r)\ln\frac{P(r)}{Q(r)} \tag{2}$$

where $M = \frac{1}{2}(P + Q)$. In this formula, P represents one LDA probability distribution, Q represents another LDA probability distribution, and r is the topic number. Let $\mathrm{PP_{u,k}}$ denote the LDA distribution of the k-th segment in $\mathrm{PP_u}$, and let $\mathrm{movie_{i,\ k}}$ denote the LDA distribution of the k-th segment in $\mathrm{movie_i}$. As a result, the similarity between these two segments is obtained as shown in Equation 3:

$$Sim_k(PP_u, movie_i) = 1 - JSD\left(PP_{u,k} \parallel movie_{i,k}\right) \tag{3}$$

The above formula indicates the similarity of the k-th segment of positive profile PP_u and the k-th segment of $movie_i$. Since we have three segments in total, we define the similarity between PP_u and $movie_i$ as follows:

$$\mathrm{PPS}(u,\ i) = \alpha\ \times \mathrm{Sim}_1(\mathrm{PP_u}, \mathrm{movie_i}) + \beta\ \times \mathrm{Sim}_2(\mathrm{PP_u}, \mathrm{movie_i}) + (1-\alpha-\beta) \times \mathrm{Sim}_3(\mathrm{PP_u}, \mathrm{movie_i}). \tag{4}$$

Example 3. Suppose the similarities between the three segments of PP_u and $movie_i$ are 0.3, 0.7, 0.4, respectively. Assume that α=0.3 and β=0.5. Then the similarity between PP_u and $movie_i$, $\mathrm{PPS}(u,\ i)$, is $(0.3\ \times\ 0.3) + (0.5\ \times\ 0.7) + (0.2\ \times\ 0.4)\ = 0.52$.

The above equation tells us the similarity between the positive profile of user u and movie i. By a similar process, we can obtain $\mathrm{NPS}(u,\ i)$, which indicates the similarity between the negative profile of user u and movie i. Finally, we divide $\mathrm{PPS}(u,\ i)$ by $\mathrm{NPS}(u,\ i)$ to obtain $Sim_{u,i}$. If this value is large, it means that movie i is very similar to user u's positive preference, but very different from user u's negative preference. Therefore, this movie should be recommended to user u.

5. Experiments

In this section, we first describe how we collected the dataset, and discuss the measurement metric. Then we conducted a series of experiments. Finally, we discuss the experimental results.

5.1. Dataset and measurement metric

The experimental dataset is mainly divided into the user rating dataset and the movie dataset. The user rating dataset mainly contains user rating data of movies, and the movie dataset contains more detailed movie information for each movie.

The user rating dataset uses the MovieLens dataset. This dataset contains ratings from many movie users. Four MovieLens datasets have been released, namely 100k, 1 m, 10 m and 20 m, which reflect the approximate number of ratings in each dataset. These datasets were first released in 1998, and major versions have been released every 5 to 6 years thereafter. Of these datasets, we chose the 1 m dataset to obtain the rating data as it is very popular (Beel, 2019). This dataset has a total of 1000,209 ratings on 3883 movies and 6040 users. The content includes the username (user ID) and score (rating) of each movie.

In the collection of movie data, in order to meet the needs of more detailed movie information in this study, we further used the movie content provided by the movie database IMDb (http://www.imdb.com), which currently has the largest amount of movie data. The data we need for each movie include three attributes: movie title, director and genre. Among them, title and genre are available in the MovieLens dataset (https://grouplens.org/datasets/movielens/). The only attribute we need to retrieve is the director of each movie. This data can be found from data set hetrec2011-movielens-2k, published by the GroupLens research group (https://grouplens.org/datasets/hetrec-2011/). Basically, hetrec2011-movielens-2k contains the movie information for MovieLens 10 m. Therefore, we can find the director information by retrieving this dataset thorough the IMDb movie id. Among 3883 movies in the MovieLens 1 m dataset, 3645 movies can be found in this dataset. There are only 283 movies that are not in the dataset hetrec2011-movielens-2k. For these remaining 283 movies, we added their directors into the dataset manually.

In this study, we use five indicators to measure the quality of the results. They are MAE (mean absolute error) and MAPE (mean absolute percentage error), precision, recall and F1-measure. Let T denote the test set with n data, let $\widehat{y_i}$ denote the predicted rating of item i in T, and let y_i denote the true rating of item i. Then MAE is defined as $\frac{\sum_{i=1}^{n}|y_i-\widehat{y_i}|}{n}$, while MAPE is defined as $\frac{\sum_{i=1}^{n}\left|\frac{y_i-\widehat{y_i}}{y_i}\right|}{n}$. The MAE indicator can show the absolute deviation between the true rating and the predicted rating, while the MAPE indicator can show the relative deviation.

On the other hand, the other three indicators can measure whether the recommended item is indeed a good item, and whether the good items are recommended by our algorithm. Let the threshold be 4. When the rating is greater than or equal to 4, it should be recommended. When the rating is less than 4, it should not be recommended. Let a denote the number of items with a predicted value of no less than 4 and a true value of no less than 4. Let b denote the number of items with a predicted value of no less than 4 but a true value of less than 4. Let c denote the number of items with a predicted value of less than 4 but a true value of no less than 4. Then precision is defined as $\frac{a}{a+b}$, recall is defined as $\frac{a}{a+c}$, and F1 is defined as $\frac{2\ \times \mathrm{precision} \times \mathrm{recall}}{\mathrm{precision}+\mathrm{recall}}$.

5.2. Parameters

In the experiment, several sets of parameters need to be set. First, they are the weights of the three segments of each movie vector or each user profile. Let these three parameters be α,β and γ respectively, where $\alpha+\beta+\gamma=1$. Weight α is used for titles, weight β is used for directors, and γ is used for genres. In our experiment, we set the value of each parameter from 0.1 to 0.8 (with a step size of 0.1). Besides, we must determine the appropriate number of topics for each segment of the profile. In the experiments, we test the number of topics from 10 to 50 with step 10.

In addition, the K value must be set. This value is used to select positive and negative movies to build a user profile. Since we do not know how big this value should be, we set the value of K from 500 to 1700 with a step size of 50. Furthermore, in the first phase of our algorithm, we use cosine similarity to determine how similar the two users are. Other similarity measures such as Pearson coefficient and Spearman rank coefficient can also be used. The Pearson's coefficient $r_{u,v}$ can be defined as $\frac{\sum_{i\in S}(v_i-\overline{v})(u_i-\overline{u})}{\sqrt{\sum_{i\in S}(v_i-\overline{v})^2}\sqrt{\sum_{i\in S}(u_i-\overline{u})^2}}$, where S denotes the set of items that are commonly rated by users v and u. The Spearman rank coefficient is similar to the Pearson's coefficient, but it is used to compute a measure between ranks instead of rating values. Therefore, we compare the cosine similarity with these two other similarity measures.

In the experiment, three models were compared. In the first model, we consider both the positive and negative profiles of users, and so call it the PP/NP model. In other words, we use the value of PPS/NPS to build a predictive model. In the second model, we only consider the positive profile, and so call it the PP model. In other words, we only use the value of PPS to build a predictive model. If the results of the first model can be better than the second model, it indicates that using a negative profile can improve the recommending performance of using only a positive profile. In the third model, we only consider negative profiles, and so call it the NP model. In other words, we only use the value of NPS to build a predictive model. If the results of the first model can be better than the third model, it means that including both positive and negative profiles can improve the recommending performance of using negative profiles alone.

In addition, we established three baseline algorithms to compare performance. These three comparison algorithms include a user-based collaborative filtering algorithm (Herlocker et al., 2015), an item-based collaborative filtering algorithm (Shah et al., 2016) and a Latent matrix factorization algorithm (Koren et al., 2009). These three algorithms represent the three most important approaches for implementing CF. We use 90% of the scoring data as the training set and the remaining 10% as the test set. In addition, the number of neighbors used to calculate the target rating is set to 30. The obtained values of the five indicators provide an objective benchmark to determine whether our model can improve upon traditional CF predictions.

5.3. *Experimental results*

In the first experiment, we studied the performance of the three baseline models. In the following table, we show the results. In Table 1, each value indicates the result we get by the specified method for the specific metric.

The above results show that the LMF model is slightly better than the user-based CF model and the item-based CF model due to having the best MAE, MAPE and accuracy. Interestingly, we found that the recall rate of all baseline algorithms is only about 40%. This means that these algorithms will not recommend 60% of the good items. This reflects that these algorithms underestimate the ratings of many good items, so these good items are classified as bad items. The reason for this phenomenon is that recommendation algorithms usually obtain product prediction scores based on scores of similar users or scores of similar products. However, the neighbors of a good item cannot all be good. A certain percentage of neighbors must be bad (value less than 4). Therefore, when the true rating is high, the weighted result is usually a lower estimate of the true rating.

In the second experiment, we wanted to determine the optimal number of topics for each segment. We set five possible options for the number of topics, namely 10, 20, 30, 40, and 50. Therefore, we executed the PP/NP model by changing all weight parameters and all K values. In Table 2, we show the best value of each indicator that can be obtained by setting a different number of topics.

In Table 2, it shows that the best number of topics is 20 due to it having the best MAE, MAPE and precision. Based on this result, all subsequent experiments used 20 as the number of topics for each segment. That is, we fixed $nt = 20$.

In the third experiment, we wanted to determine whether in the first step of the algorithm, cosine similarity is the best way to calculate user similarity. Therefore, we compared it with the other two well-known similarity methods (namely Pearson similarity coefficient and Spearman rank coefficient). We executed the PP/NP model using all weight combinations and K values. Through them, we report the best value of each indicator that can be obtained through these three different methods. The results are shown in Table 3.

The above results show that the performance of these three methods is roughly the same. However, we chose cosine similarity as our similarity measure because it has better MAE and MAPE. Traditionally, these two indicators are widely used to compare the performance of recommendation algorithms. Therefore, in subsequent experiments, we used the cosine metric to calculate the similarity between users.

In the fourth experiment, we studied the performance comparison of the NP model, the PP model and the PP/NP model. We executed the three models by varying all the weight parameters and all the K values. In Table 4, we show the best value of each indicator we can get from these three models and the baseline model.

Obviously, the PP/NP model works best among all five indicators. This means that it not only has the smallest deviation, but also has the best classification accuracy. Here, we notice that the PP and NP models have very poor results in terms of precision and recall. This may be for the following reasons. Low precision means that many bad items are classified as good items. In other words, it overestimates the ratings of many bad items. This makes the precision low. Low recall rate means that many good items are classified as bad items. In other words, it underestimates the ratings of many good items. This makes the recall rate low. These two phenomena indicate that these two models (NP and PP) classify many good items as bad items, and at the same time classify many bad items as

Table 1
The performance of the three baseline models.

Baseline	MAE	MAPE	F1	precision	recall
User-based CF	0.75568	0.29747	0.54223	0.82009	0.40501
Item-based CF	0.75131	0.29817	**0.54761**	0.82774	**0.40914**
LMF	**0.74269**	**0.29354**	0.53078	**0.85721**	0.38439
Best of baselines	0.74269	0.29354	0.54761	0.85721	0.40914

Table 2
The performance for different topic numbers.

Number	MAE	MAPE	F1	precision	Recall
10	0.67608	0.24560	**0.61677**	0.94929	**0.47108**
20	**0.66499**	**0.24268**	0.61187	**0.95252**	0.46869
30	0.75166	0.27787	0.53732	0.90308	0.40205
40	0.85874	0.31762	0.46498	0.79254	0.35656
50	0.90545	0.33718	0.41519	0.73984	0.32290

Table 3
The performance for different similarity metrics.

Sim Metric	MAE	MAPE	F1	precision	recall
Cosine	**0.66499**	**0.24268**	0.61187	0.95252	0.46869
Spearman	0.66644	0.24339	**0.61196**	**0.95259**	0.46714
Pearson	0.66548	0.24301	0.61189	0.95083	**0.47005**

Table 4
The comparison among PP, NP and PP/NP.

Models	MAE	MAPE	F1	precision	recall
NP	0.83184	0.32999	0.21809	0.22930	0.23152
PP	0.79205	0.31635	0.25819	0.32175	0.25615
PP/NP	**0.66499**	**0.24268**	**0.61187**	**0.95252**	**0.46869**
Best of baselines	0.74269	0.29354	0.54761	0.85721	0.40914

good items.

The PP/NP results in Table 4 the table above indicate that the synergy of the PP model and the NP model can correct these shortcomings. This improvement may be due to two reasons. First, the PP/NP model has smaller deviations than the PP and NP models. Small deviations essentially help improve the accuracy of classification. Secondly, the PP/NP model will reorder the similarity results obtained from the PP and NP models by dividing the similarity of PP by the similarity of NP. This re-ranking makes the similarity result ranking of the PP/NP model more consistent with the true similarity ranking. Therefore, it can help reduce the overestimation of bad items' ratings, and it can also reduce the underestimation of good items' ratings.

In the fifth experiment, we will show the best five MAE, MAPE and F1 results of the PP/NP model and the corresponding parameters. In Table 5, the first five rows show the best five MAE results and their corresponding parameters and other indicator values. The second five rows are for MAPE, and the last five rows are for F1.

The results in Table 5 indicate that when the goals are different, we must set different weight combinations and K values. If the goal is a small deviation, a weight combination (0.4, 0.2, 0.4) with a K value between 800 and 1050 is a good choice. However, if we want to get the best classification, a better combination of weights is (0.5, 0.1, 0.4), and the K value is between 1100 and 1450.

From the results in Table 5, we chose a certain combination to compare the baseline models to understand the improvements we can get from the PP/NP model. The selected combination is (0.4, 0.2, 0.4)−900. Then, we obtained the comparison results in Table 6. From Table 6, we can see that the improvements in all five indicators are significant.

Table 5
The top five MAE, MAPE and F1 results for the PP/NP model.

Criterion	Rank	Weight	K	MAE	MAPE	F1	Precision	Recall
MAE	1	0.4, 0.2, 0.4	900	**0.66499**	0.24316	0.59838	0.93841	0.45619
	2	0.4, 0.2, 0.4	850	**0.66513**	0.24314	0.59305	0.93307	0.45215
	3	0.4, 0.2, 0.4	1000	**0.66546**	0.24362	0.60079	0.93983	0.45844
	4	0.4, 0.2, 0.4	1050	**0.66669**	0.24436	0.60318	0.94306	0.46074
	5	0.4, 0.2, 0.4	800	**0.66674**	0.24268	0.59345	0.93555	0.45187
MAPE	1	0.4, 0.2, 0.4	800	0.66674	**0.24268**	0.59345	0.93555	0.45187
	2	0.5, 0.2, 0.3	800	0.67103	**0.24308**	0.59655	0.93972	0.44593
	3	0.4, 0.2, 0.4	850	0.66513	**0.24314**	0.59305	0.93307	0.45215
	4	0.4, 0.2, 0.4	900	0.66499	**0.24316**	0.59838	0.93841	0.45619
	5	0.5, 0.2, 0.3	900	0.66789	**0.24320**	0.60646	0.94388	0.44896
F1	1	0.5, 0.1, 0.4	1100	0.68514	0.24686	**0.61187**	0.94245	0.46437
	2	0.5, 0.1, 0.4	1450	0.69891	0.25191	**0.61091**	0.94479	0.46404
	3	0.5, 0.1, 0.4	1250	0.69077	0.24867	**0.61076**	0.94453	0.46267
	4	0.4, 0.2, 0.4	1450	0.67809	0.24870	**0.61053**	0.94404	0.46859
	5	0.4, 0.1, 0.5	1450	0.69502	0.25366	**0.60976**	0.93478	0.46811

Table 6
The comparison of the PP/NP and the baseline.

Models	MAE	MAPE	F1	precision	recall
(0.4,0.2,0.4)−900	**0.66499**	**0.24316**	**0.59838**	**0.93841**	**0.45619**
Best of baselines	0.74269	0.29354	0.54761	0.85721	0.40914
improvement	10.46%	17.16%	9.27%	9.47%	11.50%

Finally, we conducted the sixth experiment by setting different thresholds for PP and NP. Our speculation was that by setting different K values for PP and NP, we could further improve the indicator value of the algorithm. Suppose the K value of PP is called PK, and the K value of NP is called NK. Then, we execute the PP/NP model by changing all weight parameters and all PK values (from 500 to 1700 in step 50) and all NK values (from 500 to 1700 in step 50). In Table 7, we show the best value of each indicator that can be obtained from the two PP/NP models; one is PK = NK and the other is PK≠NK.

The results shown above indicate that by setting different K values for PP and NP, we can further improve the best value of each indicator. For example, the MAE value can be further reduced from 0.66499 to 0.64288, and the F1 value can be increased from 0.61187 to 0.62518. However, the improvement is not so significant; it is just a small improvement.

Next, we will display the best five MAE, MAPE, and F1 results of the PP/NP model (PK≠NK) and the corresponding parameters. In Table 8, the first five rows show the best five MAE results and the values of their corresponding parameters and other indicators. The last five rows are used for MAPE and the last five rows are used for F1.

According to the results in Table 8, we see that the weight combination (0.4, 0.3, 0.3) is suitable for the MAE indicator, the combination (0.4, 0.2, 0.4) is suitable for the MAPE indicator, and the combination (0.8, 0.1, 0.1) is suitable for the F1 indicator. Interestingly, we noticed that the parameters applicable to F1 have larger MAE (around 0.82) and MAPE (around 0.27). The possible explanation is as follows. There are many factors that may affect the classification accuracy of the model. Although small deviations help reduce classification errors, other factors are also important. They are the place where the deviation occurs and the direction of the deviation. If the deviation occurs near the threshold line (value = 4), even if it is small, it may seriously affect the classification accuracy. However, if the deviation occurs in a far area, for example, rating = 1 or 2, the impact may be small even if the deviation is large. Another factor is the direction of the deviation. For items with a true score of less than 4, if the deviation is negative, no matter how large it is, there will be no impact, and the score is still less than 4. On the other hand, for products with a true score of not less than 4, if the deviation is positive, no matter how large it is, the predicted score is still greater than 4.

From the results in Table 8, we chose a certain combination to compare the baseline models to understand the improvement we can get from the PP/NP model with PK≠NK. The selected combination is (0.4, 0.2, 0.4) −600–1050. Then, we obtained the comparison results in Table 9. From Table 9, we can see that the improvements in all five indicators are significant.

Finally, we compare two combinations selected from two PP/NP models. The first selected model is (0.4, 0.2, 0.4)−900–900, which is used in Table 6. The second selected model is (0.4, 0.2, 0.4)−600–1050, which is used in Table 9. The comparison results are shown in Table 10.

It can be seen that the improvement is not obvious. Undoubtedly, since the solution space of the PP/NP model with PK = NK is a subset of the PP/NP model with PK≠NK, the results of the former model cannot be better than the latter. If only a single indicator is considered, the latter model can be more improved than the former model. However, since we must consider five indicators at the same time, it is difficult to find a parameter combination in the latter model, which can greatly exceed the other parameter combinations in the former model in all five indicators.

6. Conclusion

This paper proposes a new approach that is different from traditional recommendations, that is, including positive/negative user profiles to obtain more accurate recommendations. In previous studies, recommendation algorithms usually generated recommendation results based on items that users like, items that similar users like, or items that the user previously purchased. In other words, previous research mainly focused on the positive profile to start the research on recommendation. No previous research has ever considered how to utilize negative information to build a recommendation model.

In this paper, we defined two profiles for each user. Positive profiles represent content the users like, while negative profiles represent content the users do not like. In order to obtain positive and negative profile information, we applied the CF algorithm to predict the user's ratings of all items. Then, we divided all these ratings of the same user into two parts. Items with a high rating value constitute a positive profile of the user, and items with a low rating value constitute a negative profile of the user. After that, we generate recommendations by selecting those items that are similar to the user's positive profile but different from the user's negative

Table 7
The comparison of the two PP/NP models and the baseline.

Models	MAE	MAPE	F1	precision	recall
PP/NP(PK≠ NK)	**0.64288**	**0.24153**	**0.62518**	**0.97730**	**0.49696**
PP/NP(PK = NK)	0.66499	0.24268	0.61187	0.95252	0.46869
Best of baselines	0.74269	0.29354	0.54761	0.85721	0.40914

Table 8
The top five MAE, MAPE and F1 results for the PP/NP model (PK≠NK).

Criterion	Rank	Weight	PK	NK	MAE	MAPE	F1	Precision	Recall
MAE	1	0.4, 0.3, 0.3	600	1500	**0.64288**	0.24622	0.583635	0.96616	0.44331
	2	0.4, 0.3, 0.3	650	1500	**0.64308**	0.24607	0.58220	0.963556	0.44262
	3	0.4, 0.3, 0.3	550	1500	**0.64335**	0.24664	0.58461	0.96501	0.44514
	4	0.4, 0.3, 0.3	500	1500	**0.64351**	0.24708	0.58556	0.96333	0.44695
	5	0.4, 0.3, 0.3	600	1450	**0.64356**	0.24614	0.58311	0.96571	0.44675
MAPE	1	0.4, 0.2, 0.4	600	800	0.65845	**0.24153**	0.59312	0.94067	0.45079
	2	0.4, 0.2, 0.4	600	1050	0.64956	**0.24163**	0.60324	0.95363	0.45971
	3	0.4, 0.2, 0.4	600	1000	0.65054	**0.24164**	0.59857	0.95243	0.45499
	4	0.4, 0.2, 0.4	600	900	0.65531	**0.24169**	0.59805	0.94637	0.45477
	5	0.4, 0.2, 0.4	550	800	0.65816	**0.24170**	0.59134	0.93755	0.45025
F1	1	0.8, 0.1, 0.1	1550	500	0.82254	0.27223	**0.62518**	0.86885	0.49536
	2	0.8, 0.1, 0.1	1500	500	0.82067	0.27173	**0.62466**	0.87089	0.49406
	3	0.8, 0.1, 0.1	1700	500	0.83067	0.27371	**0.62463**	0.85990	0.49696
	4	0.8, 0.1, 0.1	1600	500	0.82597	0.27277	**0.62428**	0.86421	0.495247
	5	0.8, 0.1, 0.1	1450	500	0.81745	0.27132	**0.62414**	0.87322	0.49286

Table 9
The comparison of PP/NP (PK≠NK) and the baseline.

Models	MAE	MAPE	F1	precision	recall
(0.4,0.2,0.4)−600–1050	0.64956	0.24163	0.60324	0.95363	0.45971
Best of baselines	0.74269	0.29354	0.54761	0.85721	0.40914
improvement	12.54%	17.68%	10.16%	11.25%	12.34%

Table 10
The comparison of the two PP/NP models.

Models	MAE	MAPE	F1	precision	recall
(0.4,0.2,0.4)−600–1050	0.64956	0.24163	0.60324	0.95363	0.45971
(0.4,0.2,0.4)−900–900	0.66499	0.24316	0.59838	0.93841	0.45619
improvement	2.32%	0.6%	0.8%	1.6%	0.8%

profile.

Our proposed method can benefit two traditional problems in recommender systems. They are the new item problem and the lack of serendipity problem. The CF algorithm usually has the first problem because there is no scoring data for new items. In our model, if there are few new items, the user profile will not be affected, because the user profile is generated by combining many item vectors, at least hundreds, as shown in the experiment. Excluding a few item vectors has a very limited impact on the generated user profile. In other words, a user profile generated with a few new items considered will be very similar to a user profile that does not consider a few new items. Therefore, if the vector of the new item x is similar to the user's positive profile but different from the user's negative profile, then we can recommend the new item x to the user.

The second problem usually occurs in the CB algorithm because it tends to recommend products that are similar to the products they previously purchased. This leads to a situation where recommended items are already familiar and not surprising to users. In our algorithm, we first apply the CF algorithm to estimate the prediction score of each user for each item. The spirit of the CF algorithm is that it will recommend items that similar users like without being limited by the user's own experience. Therefore, there will be more opportunities to recommend more diversified products. Since our user profile is established based on the results of CF, it will inherit this characteristic and generate more diverse recommendations.

In our experimental section, we verified the usefulness of including negative profiles to improve recommendations. The experimental results show that, compared with the traditional CF algorithm, the PP model and the NP model, the PP/NP model can obtain better results on all five indicators. In conclusion, this article proves that incorporating the user's negative preferences into the recommendation model design can greatly improve the traditional CB method that only uses positive profiles.

7. Implications of the study and future work

The results of this study indicate that negative profiles (users' negative preferences) should be included when designing recommendation algorithms. However, it is not easy to identify the user's negative preferences because almost all datasets only store the user's positive feedback, such as click, purchase or browsing. In this paper, we apply the CF algorithm to obtain the user's predicted score for all items, so we can divide the user's preferences into a positive profile and a negative profile. This demonstrates an example of how to create a user negative profile based on the dataset. Due to the popularity of scoring data, our method of inferring users' negative profiles can be applied to other similar recommendation applications. However, if there are no scoring data available, different methods should be designed to mine the user's negative profile from the data, and then the negative profile can be used to further eliminate bad suggestions and improve the overall recommendation.

Regarding future research, many previous recommendation applications can be reconsidered by including users' negative profiles. For example, in a video recommendation application on a platform such as YouTube, recommendations are usually based on previous videos watched by the user, or videos recently watched by the user, or channels subscribed to by the user. However, users have many unwatched movies or unsubscribed channels, so we have the opportunity to explore the negative profiles. We can then use the negative information to further refine the recommendations. There are many similar issues that can be reconsidered in this way. For example, music recommendation, tweet recommendation, news recommendation, blog recommendation, and book recommendation, just to name a few.

CRediT authorship contribution statement

Yen-Liang Chen: Conceptualization, Methodology, Supervision, Validation, Writing - original draft, Writing - review & editing. **Yi-Hsin Yeh:** Data curation, Software, Validation. **Man-Rong Ma:** Conceptualization, Methodology.

References

Alghamdi, R., & Alfalqi, K. (2015). A survey of topic modeling in text mining. *International Journal of Advanced Computer Science and Applications, 6*(1), 147–153.

Bansal, S., Gupta, C., & Arora, A. (2016). User tweets based genre prediction and movie recommendation using LSI and SVD. In *Proceedings of the Ninth International Conference on Contemporary Computing* (pp. 1–6).

Beel, J. (2019). And the winner is MovieLens - On the popularity of recommender system datasets. https://isg.beel.org/blog/2019/08/03/and-the-winner-is-movielens-on-the-popularity-of-recommender-system-datasets/.

Bergamaschi, S., & Po, L. (2015). Comparing LDA and LSA topic models for content-based movie recommendation systems. In *Proceedings of International Conference on Web Information Systems and Technologies* (pp. 247–263).

Bougiatiotis, K., & Giannakopoulos, T. (2018). Enhanced movie content similarity based on textual, auditory and visual information. *Expert Systems with Applications, 96*, 86–102.

Chikhaoui, B., Chiazzaro, M., & Wang, S. (2011). An improved hybrid recommender system by combining predictions. In *Proceedings of IEEE Workshops of International Conference on Advanced Information Networking and Applications* (pp. 644–649).

Ha, T., & Lee, S. (2017). Item-network-based collaborative filtering: A personalized recommendation method based on a user's item network. *Information Processing & Management, 53*(5), 1171–1184.

Harper, F. M., & Konstan, J. A. (2015). The movielens datasets: History and context. *ACM Transactions on Interactive Intelligent Systems, 5*(4). no. 19.

He, C., Parra, D., & Verbert, K. (2016). Interactive recommender systems: A survey of the state of the art and future research challenges and opportunities. *Expert Systems with Applications, 56*(1), 9–27.

Herlocker, J., Konstan, J., Borchers, A., & Riedl, J. (2015). An algorithmic framework for performing collaborative filtering. *ACM SIGIR Forum, 51*(2), 227–234.

Huang, T. C. K., Chen, Y. L., & Chen, M. C. (2016). A novel recommendation model with google similarity. *Decision Support Systems, 89*, 17–27.

Indira, K., & Kavithadevi, M. K. (2019). Efficient machine learning model for movie recommender systems using multi-cloud environment. *Mobile Networks and Applications, 24*, 1872–1882.

Isinkaye, F. O., Folajimi, Y. O., & Ojokoh, B. A. (2015). Recommendation systems: Principles methods and evaluation. *Egyptian Informatics Journal, 16*(3), 261–273.

Jelodar, H., Wang, Y., Yuan, C., Feng, X., Jiang, X., Li, Y., et al. (2019). Latent Dirichlet allocation (LDA) and topic modeling: Models, applications, a survey. *Multimedia Tools and Applications, 78*, 15169–15211.

Jiang, C. Q., Duan, R., Jain, H. K., Liu, S. X., & Liang, K. (2015). Hybrid collaborative filtering for high-involvement products: A solution to opinion sparsity and dynamics. *Decision Support Systems, 79*, 195–208.

Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix factorization techniques for recommender systems. *Computer, 8*, 30–37.

Lu, J., Wu, D. S., Mao, M. S., Wang, W., & Zhang, G. Q. (2015). Recommender system application developments: A survey. *Decision Support Systems, 74*, 12–32.

Mishra, R., Kumar, P., & Bhasker, B. (2015). A web recommendation system considering sequential information. *Decision Support Systems, 75*, 1–10.

Najafabadi, M. K., Mohame, A., & Onn, C. W. (2019). An impact of time and item influencer in collaborative filtering recommendations using graph-based model. *Information Processing & Management, 56*(3), 526–540.

Ponnam, L. T., Punyasamudram, S. D., Nallagulla, S. N., & Yellamati, S. (2016). Movie recommender system using item based collaborative filtering technique. In *Proceedings of the International Conference on Emerging Trends in Engineering Technology and Science (ICETETS)* (pp. 1–5).

Ramos, G., Boratto, L., & Caleiro, C. (2020). On the negative impact of social influence in recommender systems: A study of bribery in collaborative hybrid algorithms. *Information Processing & Management, 57*(2), Article 102058.

Sanz-Cruzado, J., Castells, P., Macdonald, C., & Ounis, I. (2020). Effective contact recommendation in social networks by adaptation of information retrieval models. *Information Processing & Management, 57*(5), Article 102285.

Shah, L., Gaudani, H., & Balani, P. (2016). Survey on recommendation system. *International Journal of Computer Applications, 137*(7), 43–49.

Sharma, D., Kumar, B., & Chand, S. (2017). A survey on journey of topic modeling techniques from SVD to deep learning. *International Journal of Modern Education and Computer Science, 9*(7), 50–62.

Soni, K., Goyal, R., Vadera, B., & More, S. (2017). A three way hybrid movie recommendation system. *International Journal of Computer Applications, 160*(9), 29–32.

Subramaniyaswamy, V., Logesh, R., Chandrashekhar, M., Challa, A., & Vijayakumar, A. (2017). A personalised movie recommendation system based on collaborative filtering. *International Journal of High Performance Computing and Networking, 10*(1/2), 54–63.

Thorat, B. P., Goudar, M. R., & Barve, S. (2015). Survey on collaborative filtering content-based filtering and hybrid recommendation system. *International Journal of Computer Applications, 110*(4), 31–36.

Tong, Z., & Zhang, H. (2016). A text mining research based on LDA topic modelling. In *Proceedings of the Sixth International Conference on Computer Science, Engineering and Information Technology (CCSEIT 2016)* (pp. 201–210).

Walek, B., & Fojtik, V. (2020). A hybrid recommender system for recommending relevant movies using an expert system. *Expert Systems with Applications, 158*, Article 113452.

Wei, S., Zheng, X., Chen, D., & Chen, C. (2016). A hybrid approach for movie recommendation via tags and ratings. *Electronic Commerce Research and Applications, 18*, 83–94.

Xu, C. H. (2018). A novel recommendation method based on social network using matrix factorization technique. *Information Processing & Management, 54*(3), 463–474.

Yang, C., Chen, X. H., Liu, L., Liu, T. T., & Geng, S. (2018). A hybrid movie recommendation method based on social similarity and item attributes. *Lecture Notes in Computer Science, 10942*, 275–285.

Yin, C., & Zhang, L. (2018). Recommendation algorithm for post-context filtering based on TF-IDF: Case study of catering O2O. *Data Analysis and Knowledge Discovery, 11*, 28–36.

Zhang, H., Ji, Y., Li, J., & Ye, Y. (2016). A triple wing harmonium model for movie recommendation. *IEEE Transactions on Industrial Informatics, 12*(1), 231–239.

Zhang, Z., Kudo, Y., Murai, T., & Ren, Y. (2020). Improved covering-based collaborative filtering for new users' personalized recommendations. *Knowledge and Information Systems, 62*, 3133–3154.