# Towards Efficient Recommendation for Films

**Qiong Jia[1,a,*], Jing Zhou[1,b]**

[1]School of Computer Science, Communication University of China, Beijing, China

[a]jqiong@cuc.edu.cn , [b]zhoujing@cuc.edu.cn

**Keyword:** Classification Accuracy, Collaborative Filtering, Expectation Maximization Algorithm, K-nearest Neighbour, Personalized Recommendation

**Abstract:** We first examine the techniques, development, and application future of the current recommender systems in the film industry. Various recommendation techniques in current applications and the K-nearest neighbor (aka. KNN) algorithm, in particular, is then introduced in detail. This is followed by an introduction to the Expectation Maximization (aka. EM) algorithm based on the Bayesian classifier, which has been applied to the classification and similarity calculations of films. Finally, the *movie_reviews* data in the NLTK (Natural Language Toolkit) library is used to facilitate experiments. We evaluate the classification accuracy of the KNN algorithm and the EM algorithm based on the Bayesian classifier. The experimental results demonstrate that, the classification accuracy of the EM algorithm for films is higher than that of the KNN algorithm and it is feasible and useful to apply the EM algorithm to films classification.

## 1. Introduction

With the rapid development of information technology, more and more people watch video via the Internet. According to the 39th China Internet Network Development Statistics Report published by China Internet Network Information Center (as shown in Figure 1). Until December 2016, China's online film users reached 545 million, an increase of 40.64 million compared with the corresponding period of 2015 with a growth rate of 8.1%.

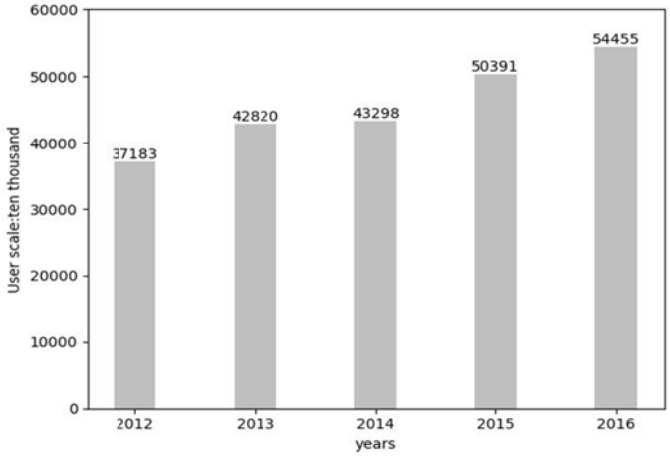


Figure 1. Growth of online video subscribers between 2012 and 2016

Meanwhile, the online film resources are massive and sometimes it is not easy for users to express their needs explicitly by using a few keywords. Therefore, searching for the video of interest can be time-consuming. Which in the worst case, may lead to a negative impact on user experience. As the watching time of users becomes more fragmented, many people are reluctant to spend too much time on searching for desirable video online. Therefore, investigating related recommendation techniques and developing a recommender system for videos/films have become an inevitable trend.

To improve the quality of recommendation, which was intended to give users a better experience

researchers designed a variety of recommender systems [1, 2]. The well-known recommender systems include Amazon, Grouplens, and Ringo [8]. Being user-centered, these recommender systems are mainly based on user evaluations for resources (or ratings) to obtain user data, analyze user interest, and finally deduce user interest in new resources.

At present, collaborative filtering (CF) [4] and content-based recommendations are most widely used. Furthermore, hybrid algorithms are also applied in recommendation technologies. The designers of most recommenders, however, still face data sparsity and the cold start problems. The data sparseness problem refers to the fact that the amount of items (films in our case) to which users have given ratings is too small. Moreover, the data sparseness leads to the increase of complexity of similarity calculations. The cold start problem is the situation in which the new items appear in the system without any user's ratings attached or new users who have yet to get a chance to give ratings to any items. Therefore, it is difficult to deliver efficient recommendations or even launch the recommendation process in certain cases. In a nutshell, personalized recommendation technique is worth of further investigation. We plan to improve the classification accuracy of films by applying the EM algorithm to videos/films recommendation.

We describe and compare both recommendation algorithms, K-nearest neighbour and EM, in Section 2. Related experiments and result analysis are presented in Section 3. We conclude our paper in Section 4.

## 2. Recommendation Algorithms

In order to offer personalized recommendations and solve the problem of information overload, recommender systems are widely used to recommend products and services to users. For example, the recommender system can select and recommend several films that users are most likely keen on from thousands of films. Due to the widespread use of recommender systems, researchers have done a lot of investigation into recommendation algorithms. Let us take the film recommendation for example.

### 2.1. K-Nearest Neighbor Algorithm

KNN [5], one of the memory-based methods, uses the entire user-project database to make predictions directly. We use the KNN algorithm to predict user ratings and hobbies. Initially, we calculate the target user information and other user information to find the users who have similar characteristics or hobbies with the target user. When $K$ users (neighbors) with similar interest are found, the prediction results can be obtained by integrating the information from neighbors' history record. The KNN algorithm includes both user-based [6, 7] and project-based [8, 9] algorithms. According to this characteristic of the KNN algorithm, we can use the user-project interaction data [3] to ignore the attributes of the user and the project itself.

A typical user-based KNN cooperative filtering algorithm consists of two phases: neighbor formation and recommendation. The algorithm compares the activity record of the target user with other users' history record $T$ at the neighbor formation stage. Then we find $k$ users whose styles are similar to the target user. The record (or data) of the target user is denoted by $u$ (represented by a vector), another user's record is denoted by $v$ ($v \in$ T), and the top $k$ most similar records to $u$ are the neighbors of $u$. The similarity between the target user $u$ and its neighbor $v$ can be calculated using the Pearson correlation coefficient.

$$\text{sim}(u,v) = \frac{\sum_{i \in C} (r_{u,i} - \overline{r_u})(r_{v,i} - \overline{r_v})}{\sqrt{\sum_{i \in C} (r_{u,i} - \overline{r_u})^2} \sqrt{\sum_{i \in C} (r_{v,i} - \overline{r_v})^2}} \tag{1}$$

where $C$ represents a collection of films that are marked by user $u$ and user $v$ at the same time, $r_{u,i}$ and $r_{v,i}$ are the scores (or weight) that the target user $u$ and the neighbor $v$ give to the film $i$, respectively. Besides, $r_u$ and $r_v$ are the average scores (or weight) given by $u$ and $v$. The most similar users are selected according to the calculated similarity.

When the nearest neighbor is determined, the target user $u$ use the following formula to derive

the predicted score for the film at the recommended phase.

$$p(u,i) = \overline{r_u} + \frac{\sum_{v \in V} \text{sim(u, v)}(\text{r}_{\text{v,i}} - \overline{r_v})}{\sum_{v \in V} | sim(u,v) |} \tag{2}$$

where $V$ is a collection of $k$ similar users, $r_{v,i}$ is the rating on film $i$ from user $v$. $r_u$ and $r_v$ are the average score given by $u$ and $v$ respectively, sim($u$, $v$) is the Pearson correlation coefficient described above. After the score is predicted, we select the highest rated film recommendation to the user.

The user-based CF lacks scalability and project-based CF overcomes this problem since the latter can pre-calculate the similarity between all the films. We can compare the films according to the film's scoring model from the user. We use KNN method to find films with a similar score given by different users, and use the following formula to adjust the cosine similarity. The greater the similarity, the shorter the distance between the target user' review and one of its neighbors' review.

$$\text{sim (i, j)} = \frac{\sum_{u \in U} (r_{u,i} - \overline{r_u})(r_{u,j} - \overline{r_u})}{\sqrt{\sum_{u \in U} (r_{u,i} - \overline{r_u})^2} \sqrt{\sum_{u \in U} (r_{u,j} - \overline{r_u})^2}} \tag{3}$$

where $U$ is the set of all the users, $i$ and $j$ are films, $r_{u,i}$ is the score of the film $i$ from user $u$ ($u \in$ U), and $r_u$ is the average score of the user $u$. We calculate the similarity between pairs of films based on the user score of the film. According to the calculation results, $k$ films that are most similar with the target film is selected and the target user rating on the target film is generated using the following formula.

$$p(u,i) = \frac{\sum_{j \in J} r_{u,j} sim\ (i,\ j)}{\sum_{j \in J} sim\ (i,\ j)} \tag{4}$$

where $J$ is a collection of similar $k$ films, $r_{u,j}$ is the score of user $u$ on the film $j$, and sim($i$, $j$) is the similarity between the film $i$ and $j$ defined above. The idea is to use the user scores of similar films to speculate their scores of a target film and the films with the highest scores will be chosen to form part of the recommendation.

Being one of the mainstream algorithms for collaborative filtering, the KNN's advantages are obvious. KNN can be used in combination with clustering algorithms to reduce the amount of computation [11]. The improved KNN algorithm has higher accuracy and flexibility. For example, one of the many KNN variations is based on weighted distance to improve accouracy [12]. Because our work presented here mainly deals with the comparison of the KNN algorithm and the EM algorithm in terms of classification accuracy. For simplicity, we only stick to the basic verision of the KNN and EM algorithms.

### 2.2. EM Algorithm Based on Bayesian Classifier

The EM (Expectation Maximization) algorithm, an iterative algorithm for maximum likelihood estimation, is widely used with incomplete data. The algorithm consists of two steps: the Expectation Step (E-step) and Maximization Step (M-step).

In the process of E-step, existed parameters (the films that the target user label) are usually used to estimate and fill the incomplete parts of the data. Each object $X$ is assigned to cluster $C_k$ with probability $P$ ($X_i \in C_k$), and the formula is as shown in formula (5). Where $P$ ($X_i|C_k$) represents the cluster membership probability of $X_i$ in cluster $C_k$.

$$\begin{aligned} \text{P}(\text{X}_\text{i} \in \text{C}_\text{k}) &= \text{P}(\text{C}_\text{k} \mid \text{X}_\text{i}) \\ &= \text{P}(\text{C}_\text{k}) * \text{P}(\text{X}_\text{i} \mid \text{C}_\text{k}) / \text{P}(\text{X}_\text{i}) \\ &= \text{P}(\text{C}_\text{k}) * \text{P}\ （\text{X}_\text{i} \mid C_k) / \sum_{i=1}^{k} P(C_k) * P(X_i \mid C_k) \end{aligned} \tag{5}$$

In the M-step of maximizing the likelihood estimation, each parameter is re-estimated. The

model parameters are recalculated using the probability estimation values that obtained previously. Furthermore, the formula is as follows.

$$m_k = 1/n * \sum_{i=1}^{m} X_i * P(X_i \in C_k) / \sum_{i=1}^{j} P(X_i \in C_j) \quad (6)$$

And then repeat M-step, the EM algorithm converges to a local optimal solution when the parameters in the model no longer change.

The EM algorithm is able to function well under incomplete data, to a certain extent, which alleviate both cold start and data sparseness problems facing current CF techniques.

In the case of film classification, the film reviews in the marked data set (indicated by *L*) have their own category labels (type, actors, users' rating, etc.). Those film reviews missing the category label can be seen as a non-marked data set (indicated by *U*). On the basis of the existing model, we use the EM algorithm to estimate these category labels to fill missing data. That is, assigning a probability category label *Pr* ($c_j$ | $d_i$) to each film identify reviews $d_i$ in set *U*. In this way, all the probabilities that films belong to the category will converge after a certain number of repeat.

The pseudo code for the EM algorithm show in Figure 2.

The EM algorithm is not a specific algorithm, but a framework or strategy, which simply runs a basic algorithm multiple times in a circular manner. We use the Bayesian classifier as the basic algorithm and this kind of EM algorithm is proposed by Nigam et al. [10].

```
Algorithm of EM( L,U )
1: Learn an initial naive Bayesian classifier f from only the labeled
     set L
2: repeat
     //E-step
3:   for each example di in U do
4:     Using the Current classifier f to compute Pr (cj|di)
5:   end
     //end of E-step
     //M-step
6:   Learn a new naive Bayesian classifier f from L∪ U by computing Pr(cj)
     and Pr(wt|cj)
     //end of M-step
7: until the classifier parameters stabilize
8: Return the classifier f from the last iteration
```

Figure 2. Pseudo code for the EM algorithm

## 3. Experiments

We carried out a series experiments on both the KNN and the EM algorithms. The reviews of *movie_reviews* in NLTK (the Natural Language Toolkit, one of the most commonly used Python libraries in the NLP field)[1] text library were used as the experimental data. In the first place, we obtained all of the film review documents, which were broken into two categories: positive and negative, and consider them as a word list consisting of all the words from the original review document. Then we computed the frequency of words in the word list and had the words sorted in descending order, thus forming a frequency list of the original film review document.

We combined all the frequency lists into a global frequency list (GFL) and, in a random manner, selected five groups of words from the GFL as characteristic words: they are the top 1,000, 2,000, 3,000, 4,000, and 5,000 words, respectively. These characteristic words were used as the training set[2] to training the classifier to find out the criteria of classification for reviews. The reason that we

[1] https://baike.baidu.com/item/NLTK/20403245?fr=aladdin

[2] The training set consists of part of the characteristic words from the original film review documents and is used to facilitate

varied the quantity of characteristic words from the training set is to compare the classification accuracy of KNN and EM in response to the change. The remainder of the characteristic words from the GFL are used as the test set[3] to predict the classifier's accuracy.

### 3.1. Experimental Methodology

Initially we set the *k*_value in the KNN algorithm, and built the test set and training set matrices that contain the probability of the constituent characteristic words. We worked out the product of a test set matrix and its counterpart training set matrix. The process was repeated for another four times for each of the rest of the test set and its corresponding training set matrices. Then, we took the largest *k* sample points (matrix product used as a measurement of similarity: the greater its value, the greater the similarity). Next, we added weights to the *k* points: the greater the similarity, the greater the weight. Finally, we selected the final category of the film based on the result of the multiplication of the weight and the number of categories. The weighted KNN algorithm was demonstrated to significantly improve the accuracy of classification as shown in Figure 3.

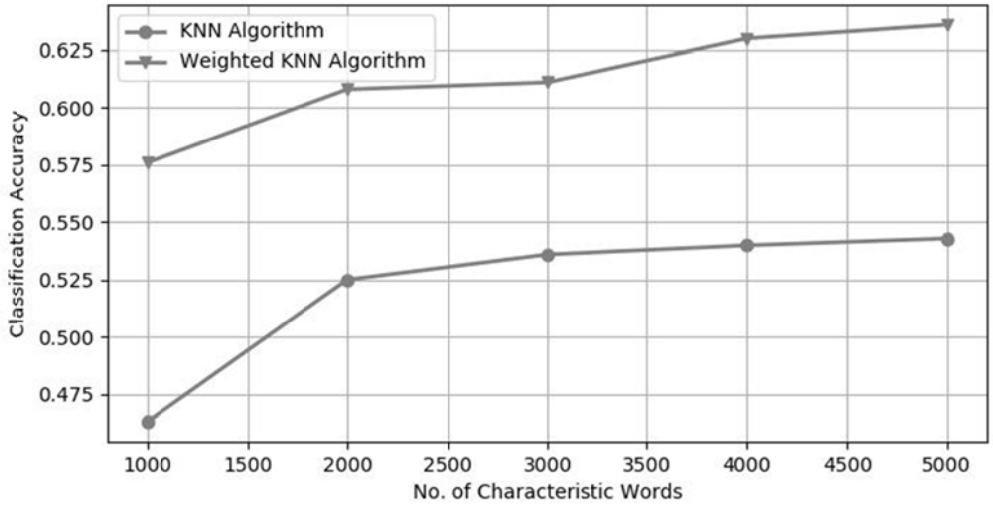


Figure3. Comparison Between Weighted KNN Algorithm and KNN algorithm

We use the KNN classifier to classify multiple test sets samples, and compared the classification results with the classification values of the original data. If the classification result is the same as the original data, the result is considered to be correct. The number of the correct results is defined as the correct value. We also define the accuracy of the classifier as the ratio of the correct value to the total length of the sample as. As for the selection of *k*_value, the classification results are susceptible to noise when *k* is very small and the neighbors may contain too many other categories of points (the weighting of the distance[4] can reduce the effect of the *k* value setting) when *k* is very large. Furthermore, *k* is generally less than the square root of the training sample number *N*.

The EM algorithm is a popular iterative refinement algorithm, which is based on the Bayesian classifier. The Bayesian classifier is used to estimate the initial classification of missing samples, and calculating the value of the model parameter. Then the E-step and M-step are executed to update the missing data values iteratively until the convergence is reached.

### 3.2. Experimental Settings

In the experiment, we used the Dell Vostro 5459-1528, with Windows 10 operating system. The CPU was Intel Core i5-6200U, and Python 2.7 was used.

We chose the k-value to be 5 to compare the result. The Bayesian classifier is trained by using the characteristic words selected from the frequency list as the eigenvalues. After obtaining the

building the classification model.

[3] The test set is a set of samples from the GFL that are used to test the classification ability of a trained model.

[4] This phrase mean that the higher probability is, the closer of possible category, and the higher weight of the category.

preliminary classification results, the iterative steps are executed to obtain the result of refinement. Finally, we compared the classification result with categorization information attached to the original reviews by the provider of *movie_reviews*, and derived the classification accuracy.

### 3.3. Result Analysis

In the experiment, the *movie_reviews* corpus in NLTK was selected as the training set. Then the KNN algorithm and EM algorithm were used to classify and calculate the classification accuracy. We used the Weighted KNN algorithm and EM algorithm based on the Bayesian classifier for a comparison of classification accuracy respectively. The experimental results are presented in Figure 4.

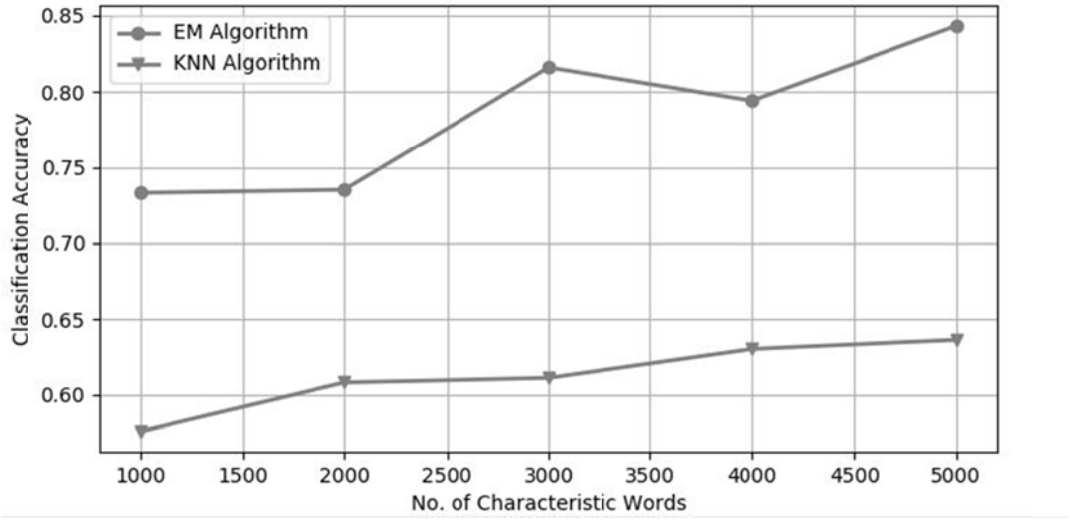


Figure 4. Comparison between EM algorithm and Weighted KNN algorithm

It can be seen that with the increase of the number of the characteristic word in the test set, the classification accuracy of the weighted KNN classifier can be slightly improved, but the increase is not obvious. In contrast, the classification accuracy of the EM algorithm is higher than that of the weighted KNN algorithm significantly.

Unlike the EM algorithm that merely uses the training set to estimate the classification result of the test set, KNN also needs to compute the distance between pairs of reviews, see Equation (3). In order to obtain the $k$ nearest neighbors in KNN, the amount of computation involved is extensive. Although the primary idea of the EM algorithm is complicated, its application is rather simple: we use the Bayesian classifier as the basic algorithm to carry out the two EM steps in an iterative fashion.

## 4. Conclusions

Keen on the recommendation techniques in the film domain, we investigated into the classification algorithm in the recommendation process. In particular, the EM algorithm based on the Bayesian classifier was applied to film classification. It can complement the original user ratings data, which, to a certain extent, alleviates the data sparseness and cold start problems typically found in recommender systems and improves the recommendation accuracy. In contrast with the KNN algorithm, the EM algorithm has obvious advantages.

The EM algorithm also has its shortcomings. When it encounters massive data, the number of iterations in the algorithm will increase significantly. Furthermore, it is difficult to achieve convergence, which also calls for further improvement. The experiment only used the data in the NLTK text library and in the future, we anticipate to employ real-world data to further enhance the classification accuracy of the EM algorithm.

**Acknowledgements**

This work was funded by the Engineering Disciplines Planning Project of the Communication University of China (No. 3132015XNG1515) and the Open Project Program of Jiangsu Engineering Center of Network Monitoring and Nanjing University of Information Science and Technology Project (PAPD and CICAEET). The authors would also like to acknowledge the input of the National Key Technology R & D Program (No. 2015BAK25B03).

**References**

[1] D. Goldberg, D. Nichols, BM. Oki and D. Terry. Using Collaborative Filtering to Weave an Information Tapestry, Communications of the ACM, 1992, 35 (12): 61-70.

[2] T. Hofmann. Latent Semantic Models for Collaborative Filtering. ACM Transactions on Information Systems, 2004, 22(1): 89-115.

[3] OH. Embarak. A Method for Solving the Cold Start Problem in Recommender Systems. Innovations in Information Technology. Abu Dhabi: IEEE, 2011: 238-243.

[4] M. Balabanovi and Y. Shoham. Fab: Content-based, Collaborative Recommendation Communications of the ACM, 1997, 40(3): 66-72.

[5] J. Herlocker, J. Konstan, L.Terveeb, and J. Riedl. Evaluating Collaborative Filtering Recommender Systems. ACM Transactions on Information Systems (TOIS), 2004, 22(1): 5-23.

[6] JS. Breese, D. Heckerman, and C. Kadie. Empirical Analysis of Predictive Algorithms for Collaborative Filtering. The Fourteenth Conference on Uncertainty in Artificial Intelligence Morgan Kaufmann Publishers Inc. 1998, 7(7): 43-52.

[7] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom and J. Riedl. GroupLens: An Open Architecture for Collaborative Filtering of Netnews. The ACM Conference on Computer Supported Cooperative Work. 1994:175-186.

[8] G. Linden, B. Smith, J. York. Amazon.com Recommendations: Item-to-Item Collaborative Filtering. IEEE Internet Computing, 2003, 7(1): 76-80.

[9] R. Jin, S. Luo and CX. Zhai. A Study of Mixture Models for Collaborative Filtering. Information Retrieval, 2006, 9(3): 357-382.

[10] K. Nigam, AK. Mccallum, S. Thrun and T. Mitchell. Text Classification from Labeled and Unlabeled Documents using EM. Machine Learning, 2000, 39(2-3):103-134.

[11] J. Shen Dynamic Collaborative Filtering Recommender Model Based on Rolling Time Windows and its Algorithm. Computer Science (2013).

[12] X. Hao, X. Tao, C. Zhang and Y. Hu. An Effective Method to Improve KNN Text Classifier. The Eighth ACIS International Conference on Software Engineering, 2007, 1: 379-384.

[13] J. Guo, W. Li, C. Li and S. Gao. Standardization of Interval Symbolic Data Based on the Empirical Descriptive Statistics. Computational Statistics & Data Analysis, 2012, 56(3): 602-610.