 Iraqi Journal of Science, 2018, Vol. 59, No.2A, pp:771-785
DOI:10.24996/ijs.2018.59.2A.15

ISSN: 0067-2904

Proposed Collaborative Filtering Recommender System Based on Implicit and Explicit User's Preferences

Alia Karim Abdul Hassan[1], Ahmed Bahaa Aldeen Abdulwahhab[2*]
[1]Department of Computer science, University of Technology, Baghdad, Iraq.
[2]Department of Informatics, Middle Technical University, Baghdad, Iraq.

Abstract
The expansion of web applications like e-commerce and other services yields an exponential increase in offers and choices in the web. From these needs, the recommender system applications have arisen. This research proposed a recommender system that uses user's reviews as implicit feedback to extract user preferences from their reviews to enhance personalization in addition to the explicit ratings. Diversity also improved by using k-furthest neighbor algorithm upon user's clusters. The system tested using Douban movie standard dataset from Kaggle, and show good performance.

Keywords: recommender system, collaborative filtering, sentiment analysis, implicit feedback.

تصميم نظام توصية افتراضي معتمد على رغبات المستخدم الضمنية و المباشرة

علياء كريم عبد الحسن[1]، احمد بهاء الدين عبد الوهاب[2*]
[1] قسم علوم الحاسبات ، الجامعه التكنلوجيه ، بغداد، العراق.
[2] قسم المعلوماتيه ، الجامعه التقنيه الوسطى ، بغداد، العراق.

الخلاصة
توسع تطبيقات الشبكة العنكبوتية في مجالات التجارة الالكترونية و الخدمات الاخرى، ادى لزيادة غيرمحدودة في العروض و الخيارات ، لذا كانت الحاجة لانظمة التوصية لتساعد المستخدم على تحديد الخيار الافضل له. هذا البحث استخدم تعليقات المستخدم السابقة لمعرفة الخصائص التي يفضلها في الافلام التي شاهدها وقيمها سابقا، من خلال تقنيات تحليل النصوص ومعالجتها لمعرفة الشعور من النص، بالاضافة الى التقييمات المباشرة، ومن ثم تجميع المستخدمين بشكل مجاميع حسب توافقهم في رغباتهم المذكورة في تعليقاتهم السابقة. تلى ذلك استخدام طريقة k–furthest لزيادة التنوع في الخيارات. اختبر هذا النظام ببيانات قياسية من شركة داوبيان الصينية ؛وقد اظهر هذ النظام كفاءة جيدة مع مراعاة التنوع في النتائج مع تخصيصها للمستخدم اعتمادا على تعليقاته.

1. Introduction
The growth of the web application like e-commerce created a critical challenge of information overload. As a consequence, the web users would be overwhelmed with choices and data [1]. Recommender systems are information filtering systems can suggest choices for the user according to his preferences [2]. There are two paradigms for building recommender systems, firstly, is the content

*Email: ahmed80.ab@gmail.com

based recommender systems which are domain dependent algorithms. In these algorithms, an item is recommended to the user according to its closeness to the attributes of an item has been chosen by the user previously. The second paradigm is collaborative filtering, which is domain independent, this approach exploits user consumption pattern of the items in the recommendation process. This approach is subdivided into two categories; the first category is memory based algorithms which calculate the recommendations from the nearest users, in spite of nearest neighbor accuracy, it suffers from scalability problem. The second category is the model-based approach, in this type, the recommender system depends on building a model using the previous user's rating dataset to predict recommendations. There many model-based techniques like Singular value decomposition which analyzes user-item matrix to identify the relations between items [3].

2. Problem statement and contributions

The recommender systems have issues like personalization, diversity, serendipity, and cold start problem [4]. Cold start problem is caused by insufficient information about the new user or new item which may cause inability in making recommendations. Personalization issue is meaning how making the system smart enough to understand the user preferences and interests [5]. Diversity issue is how to make the recommender systems provide a variety of choices that covers all users' interest [4]. Serendipity means the ability of the system to recommend a novel, relevant, items that different from items that the user has rated [3]. According to these issues the contributions of this research are, firstly exploit users' reviews in recommendation process by extracting latent features from these reviews like (actor he liked, the editor he liked, etc.) and consider these reviews as implicit feedback. These latent features would enhance personalization by understanding the user from his previous reviews. The second contribution is increasing the diversity and serendipity of results by applying K-furthest algorithm over clustered users; the k-Furthest algorithm would select related users but have different taste from the targeted user, then stabilize the result between targeted user preferences and movies in the recommendation list. The third contribution is managing the new user's cold start problem using registration process and new movie cold start problem by using an external source of information which is Wikipedia infobox.

3. Related works

Many researchers have worked to enhance user personalization, by understanding user preferences from implicit feedback. One of the researcher's directions is exploiting reviews sentiment analysis as an implicit feature in social network recommender systems. Dimah et al. used sentiment analysis in analyzing reviews of trusted users in the twitter. Dimah measured the degree of trust between friends to recommend positive and negative opinions among close friends [6]. Ashok et al. used more advanced sentiment analysis to extract point of interest to the user from the parsed comments in a social network [7]. Other researchers used another implicit factor to model user preferences. Yifan Lu et al. have depended on user consuming rate of a product to understand the user's preferences [2], while Gaweesh et al. use the number of listening to a specific track as implicit feedback in music recommender system, which is close to Yifan thought. Ayad et al. proposed a recommender system based on content description about products as implicit feedback, but this implicit feedback may be exploited by the advertisement companies [9]. Rafael et al.[8] built a dictionary of products by extracting products features from a user's reviews, then use this dictionary in recommender systems [10]. This research uses both explicit feedback (ratings) and implicit feedback (by analyzing user's reviews). This proposed architecture would improve user personalization by extracting the user interest from his previous review while the explicit rating to decrease data sparsity by filling all blanks in the user-item matrix. Implicit features used with k-furthest neighbors approach to achieve diversity.

4. Recommender system steps

The general framework of recommender system operations, as shown in Figure-1, consists of three steps which are, information collection step, that accomplished by explicit feedback and implicit feedback, then learning step, lastly prediction step [1].

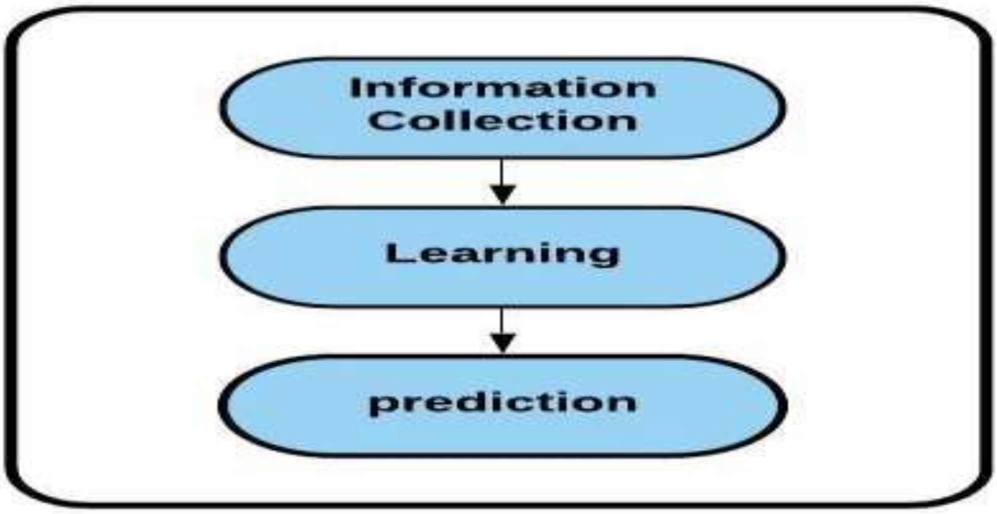


Figure 1-Recommender system process operations.

4.1 Information collection step

This process is building a user profile, like name, age, and habits. A recommender system agent cannot operate without user's profiles. The recommender system needs information about the users as much as possible [11]. The vital success factor in recommender system depends on its capability to model user interest. This process can be done using the following information operations:-

4.1. A Explicit feedback

The recommender systems guide the user through an interface to provide ratings for items to construct the user's model. Explicit feedback is a reliable and straightforward to gather information from users by the score, but it is unsuitable to know user feelings about an item [111].

4.1. B Implicit feedback

The system can catch the preferences of the user by monitoring his activities like the history of purchases, a stream of web pages clicks, his feelings in text using sentiment analysis. Despite that implicit feedback does not need any effort from a user like filling forms, it lacks features to be extracted to get user conditions and desires [111].

4.2 learning process

This process implements machine learning algorithm to build a model from elements collected from explicit and implicit feedback ratings [1].

4.3 Prediction process

There are many algorithms for prediction process; recommendations can be made either directly using dataset gathered in information collecting process through memory based or model based [1].

5. Sentiment analysis as an implicit feedback

Sentiment analysis or opinion mining is the study of emotions toward an item or entity12]. The sentiment analysis of users reviews can be used to measure the similarity between users, instead of ratings which are always have been used in collaborative filtering. Reviews sentiment analysis outperforms the explicit rating using stars, by specifying the reason behind user preferring the targeted item. For instance, in the case of a movie, did the reviewer prefer it, because of a specific actor? Or because of the director? So this paper debate that text reviews offer a deduce opinion of a user for an item, making them an ideal source of knowledge for enhancing recommendation process by extracting the latent feature from reviews [13].

The first step is by specifying the positive reviews for users who rate the item with three stars or higher, otherwise, the review considered to be negative. The second step dividing the data into training and testing sets, after that extracting the features of user interest from positive movie review as a latent feature [14]. A latent feature like (good movie) can be retrieved by converting the review into its part of speech, then from parsed reviews specify chunks using chunking process, the most critical chunk pattern are:-

1) Noun Verb Adjective
2) Verb ADJ ADP Verb
Note that ADJ. is for the adjective and ADP for adposition

These extracted chunks are latent features saved as a profile for each user to be used later in clustering process for gathering same users with same latent feature preferences retrieved from their reviews [15]. This latent features or chunks can be used as an excellent summary of user opinion about some product [16].

6. Proposed system structure

This section presents the proposed architecture of the movie recommendation that depends on the clustering the users using explicit rating and implicit latent features extracted from their reviews.

6.1 Proposed system modules description

The proposed system consists of the following modules which can be seen in Figure-2 which are:-

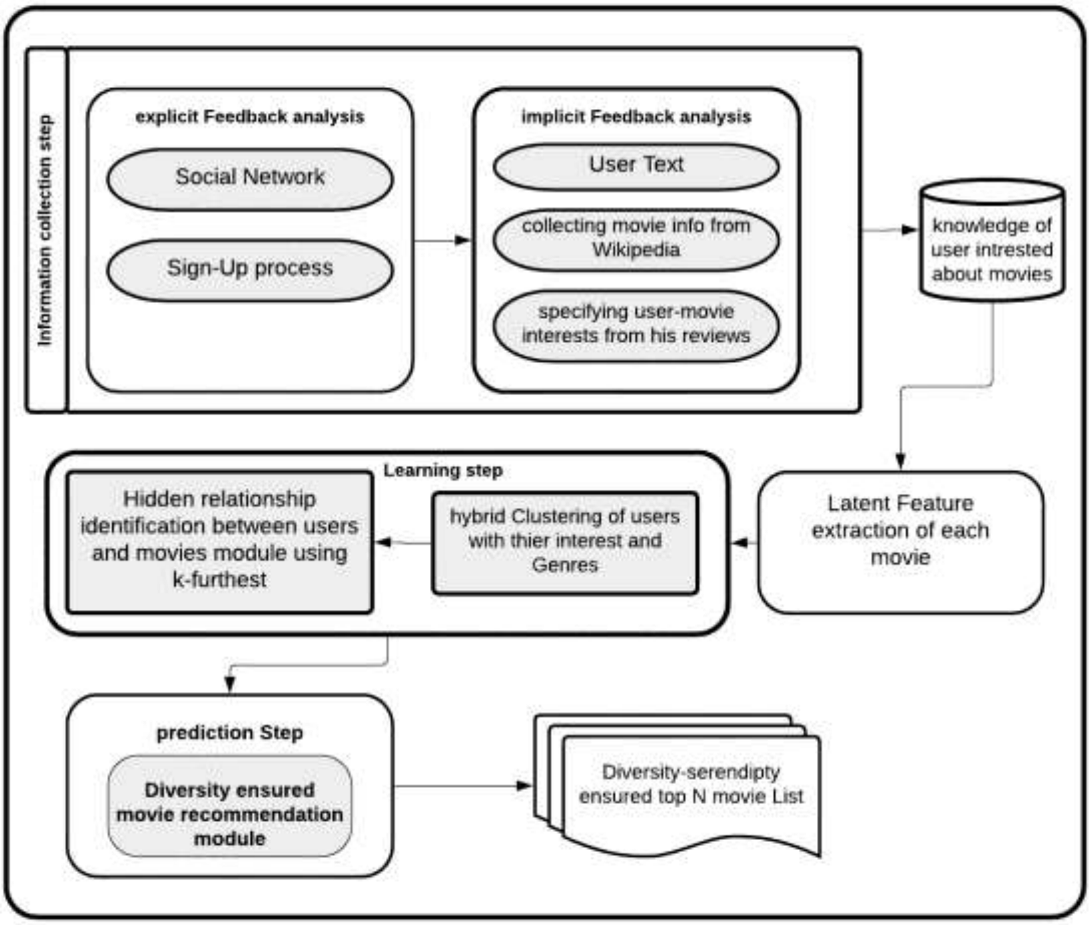


Figure 2-the proposed system modules.

A. User explicit feedback analysis module

Input: Explicit rating from Movie-review site, and Sign-Up process.

Output: - A set of preferences for every individual user and feature weight of the user on all the movies.

The Movie-review site comprises the explicit rating information on each movie by the users. Moreover, the recommender system enables the preference elicitation at the Sign-up stage to contextually identify the user preferences. By utilizing the information source of sign up process and input the user (name, age, and gender). Then the system computes the similarity between users using an algorithm (1), this algorithm has significance over cosine similarity as it takes into consideration the demographic similarity between users in gender, age, and movie rating similarity, see algorithm (1):-

Explicit rating algorithm

1. Start
2. Gendersimilarity =0 , agesimilarity=0, usersimilarity=0
3. If usergender1 = usergender2 then
   Gendersimilarity =1
4. Agediff=0
5. IF u1age > u2age then
   Agediff=u1age-u2age
   Else agediff=u2age-u1age
6. If agediff=0 then
   Agesimilarity =1
   Else if agediff>0 and agediff<=5 then
   Agesimilarity=0.98
   Else if agediff>5 and agediff>=15 then
   Agesimilarity =0.9
   Else if agediff>15 and agediff<=25 then
   Agesimilarity =0.85
7. # calculating the rating similarity between two users using Sik equation(1)
   $Sik = 1/\left[1 + \sqrt{\sum_{j=1}^{m} Rij - Rkj}\right]$ + Age similarity +gender similarity ---------- (1)
8. Totalcimilarity =(comma*gendersimilarity)+ (beta*agesimilarity)+(alpha*usersimilarity)
9. If totalsimilarity>0.47 then
   Add to list of similar users (user1, user2)
10. Use rate of movies of user1 and user2

Algorithm (1) explicit rating algorithm

After calculating the similarity between users using an algorithm (1). The proposed approach computes the rate for each movie from each user using the equation (1)

$$Explicit\ score(Uij) = Rij + [\sum_{k=1}^{n} Rkj * Sik \,/ \sum_{k=1}^{n} Sik\ ] \quad (1)$$

Where n is the number of the Similar user to the ith user.
S_{ik} refers the similarity between the ith user and kth user.
Rkj = rank of user k to movie j.
Rij=rank of user I to movie j

B. Implicit feedback analysis module.

Input: - User's textual reviews, Wikipedia Infobox information.
Output: - User’s interest score on every inherent feature of the movie

The proposed approach inherently explores the intention of the users by extracting the latent features behind the movies. With the help of unsupervised learning, the proposed approach recognizes the inherent features of the movies from the user-specific and movie-specific information source. User-specific information source refers the textual review submitted by the user, which is further analyzed by the sentiment analyzer to extract the feature-values this operation illustrated in implicit rating algorithm (2) below. Movie-specific information source refers the Wikipedia Infobox information that contains the feature-value pairs for each movie these latent features would be extracted from Wikipedia are (directed by, produced by, starring, music by, written by, and edited by).

```
Latent feature extractor algorithm
1. Start
2. Wikipedia feature_ list [director, starring, writes, Music, actor]
3. Collect reviews of user Ui
4. for each review Ri    # preprocessing
        A. remove commas and replace with space
        B. convert the review Ri into lower case
5.  for each review do:
        a. Tokenize the review
        b. Extract part of speech
        c. Extract noun phrases "NP: {<DT>? <JJ>*<NN>}"
8. IF noun = director then
        Save user_id, director_name in list
        Mj^F =1       # repeat 8 for starring, writer, Music, actor
features
9. End
```

Algorithm (2) sentiment analysis and latent feature extraction algorithm

After specifying the positive reviews of user Ui on movie j, the system computes the implicit score, which represents the score of the user to the movie from his review by using equation (2)

Implicit Score ($U_{i,j}$) =R_{ij} (M_j^F) / MaxRating (R_i) **(2)**

Where R_{ij} (M_j^F), refers the rating of the user I on the j^{th} item. M_j^F , if the user i has the positive review about the jth item features by the ith user, (MjF), is '1'; otherwise '0'. Then calculate the latent score, which is the average of the user explicit rate and his implicit rate from his review as in equation (3)

$$\boldsymbol{LS(Uij)} = \frac{\textbf{Implicit score } (\boldsymbol{Uij}) + \textbf{exiplit rating } (\boldsymbol{Uij})}{\mathbf{2}} \qquad \textbf{(3)}$$

LS ($U_{i,j}$) refers the Latent Score of the i^{th} user on the jth item

C. Latent feature based Hybrid Clustering module

Input: - User's preferences regarding latent features of movies

Output: - Feature-based hierarchical clusters of users

The proposed approach applies two-level clustering methods to precise group the similar users. Initially, the proposed approach applies K-means clustering on all the users with the input of latent feature scores on each movie. Then, it sub-cluster theses clusters using movies genres. Finally, the features such as 'director name,' 'actor name' along with movie category have the separate clusters of users based on the feature based similarity score. the steps of clustering are illustrated in Algorithm (3) below, the sample of clustering result can be seen in Figure-3 that is a summarized representation figure about the result of the hybrid clustering process.

Multiple k-means clustering algorithm

---

1. Start
2. Cluster users according to latent score to each movie
3. Extract the genres of each movie from Wikipedia information source
4. Generate sub cluster of users under movie genres
5. The resulted clusters is cluster of users according to latent features like actor and sub cluster the actor cluster according the movies genres acted by this actor

Algorithm (3) multiple k-means clustering

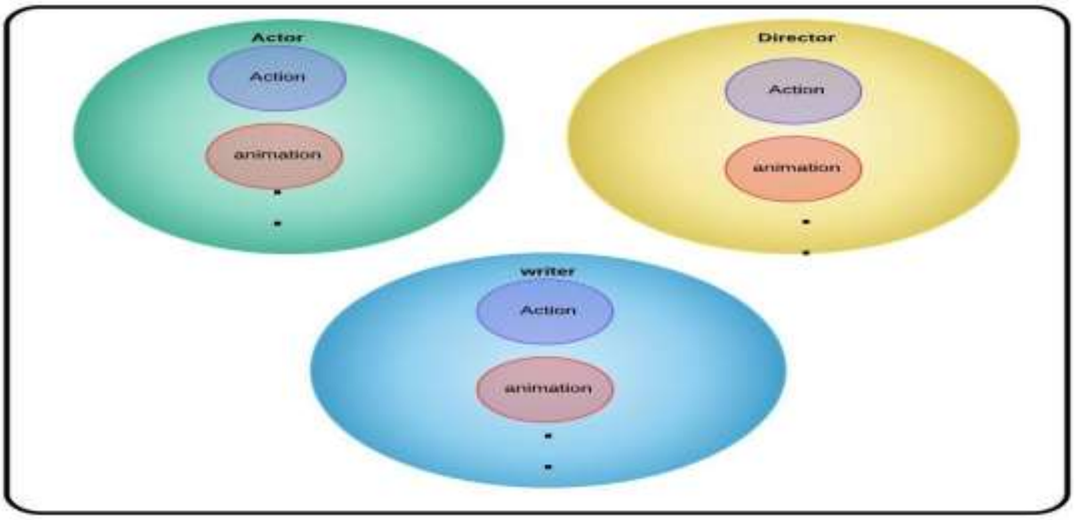


Figure 3- illustration of the resulted clusters

D. Hidden relationship identification between users and movies module.

Input: - clustered users

Output: - Hidden score for each user on a movie with a k-furthest neighbor of each user.

By exploiting the clustered users, the proposed approach identifies the hidden relationship between the user-movie pairs. It focuses on considering the movies have positive review submitted by the neighbors who are all residing in the same cluster and also considering the movies have negative review submitted by the dissimilar users who are all located in other clusters. From the neighbors of inter-clusters, the k-furthest neighbor is determined for each user according to their behavior and latent factors related to fulfilling the serendipity. The approach first identifies the most dissimilar users to each user, to create dissimilar neighborhoods. In every neighborhood, it then identifies the most disliked items and recommends these. This double negation creates a more diverse, yet personalized, set of recommendations. This approach determines the dissimilar user and their movies which have negative score using the k-furthest neighbor method. Those extracted movies are considered as the most preferred movie for the user X for example. To calculate for the k-furthest neighbor, any two users have to check if they are dissimilar, by checking if they have co-interacted with some items, this can be done by finding the users with the smallest similarity between them, and recommend the most disliked item by those users. See algorithm (4):

```
k-furthest neighbor procedure:
for each subcluster i do
    Select the target user with latent features
        for each subcluster j do
            if(i != j) then
                Users(Us)=Select the list of users who are not matched to the any one
            of latent features-value pair of target user
                Measure latent score difference between item of target user and Us.
                Sort the Latent score of Us descending order
                Select the K-Users as the dissimilar users of target user
            end If
        end for
end for
```

Algorithm (4) k-furthest approach

E. Diversity and Serendipity-ensured movie recommendation module.

Input: - Personalized movie recommendation list

Output: - Diversity and Serendipity-ensured Personalized movie recommendation list.

The proposed approach generates a list of movies for each user using k-furthest neighbor algorithm. Even though the recommended list comprises a list of user-preferred movies, there may be a duplication of movies regarding similar category. For instance, the recommendation list may contain movies belonging to the same group of the director or same genre as (horror). Hence, the proposed approach resolve this inconvenience by rearranging a list of ranked movies. The idea of this module is simple; it aims to fulfill the diversity by equalization the recommendation result list between user preferences and k-furthest neighbors users preferences. The system is doing this by extracting the preferences values (Pfi) and its genres (Pci) like the actors and movie genres list from user preferences. Then the system obtains the recommended features (Rfi) and their genres (Rci) from recommendation list that are resulted from k-furthest neighbor, then compromise between the desired preferences and the recommended one using equation (4):

$$Diversity\ (Ui) = \frac{\frac{Pf1 \cap Rf1}{Pf1 * N} + \frac{Pc1 \cap Rc1}{Pc1}}{2} \quad (4)$$

If the diversity below the threshold 0.5 then, this mean that the intersection between user preferences and actual recommendation is acceptable which increases the diversity. The diversification algorithm (5) below is detailed

For each user i do

Select the high preference movies (Pm_i) and high preference movie genres (Pc_i) is based on Latent feature score of user i.

Extract Preference Feature Values (Pf_i) of Pm_i

//Recommendation List

User i-Recommended items

$RecommItems_i$= {Rec-item1, Rec-item3, Rec-item5, Rec-$item_k$}

Extract Recommended Feature Values (Rf_i) of $RecommItems_i$

Extract Recommended Category (Rc_i) of $RecommItems_i$

$$\text{Diversity}(U_i) = \frac{\left(\frac{Pf_i \cap Rf_i}{Pf_i * N}\right) + \left(\frac{Pc_i \cap Rc_i}{Pc_i}\right)}{2}$$

Measure Diversity of U_i from above equation

If (diversity < threshold) then

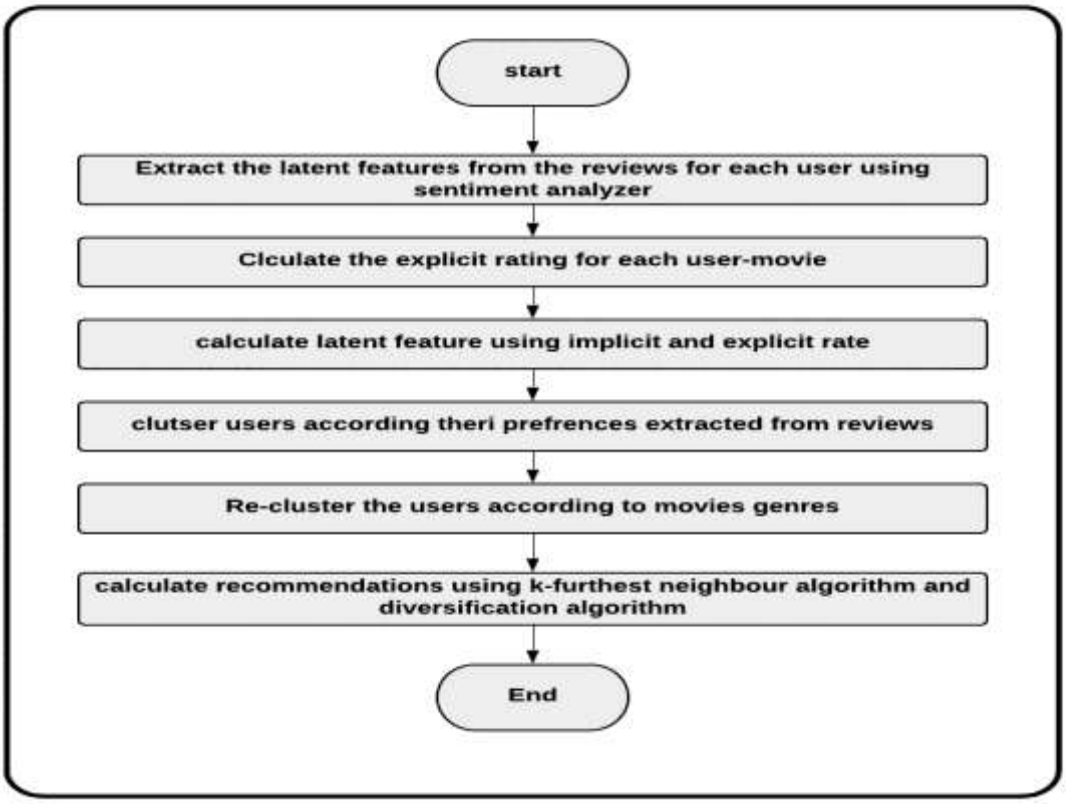
Re-Ranking the Recommendation List

End if

End for

Algorithm (5) The diversification algorithm

So the brief to the primary process of the proposed system model can be abstracted in the following flowchart (1)



Flowchart (1) The abstraction of the proposed system operation

7. Proposed system operating steps

Step1:- when a new user enters to the recommender system, the system starts the sign-up process to avoid user cold start problem; the user has to fill the fields of (user_ name age, gender, and hobbies), as shown in Figure-1 below:

Figure 1-new user registration process.

Step2:- proposed system select a set of favorite movies that are given to the user to provide the rating for those movies. According to the new user rating, the system computes the explicit feedback score for that user to all movies. Figure-2

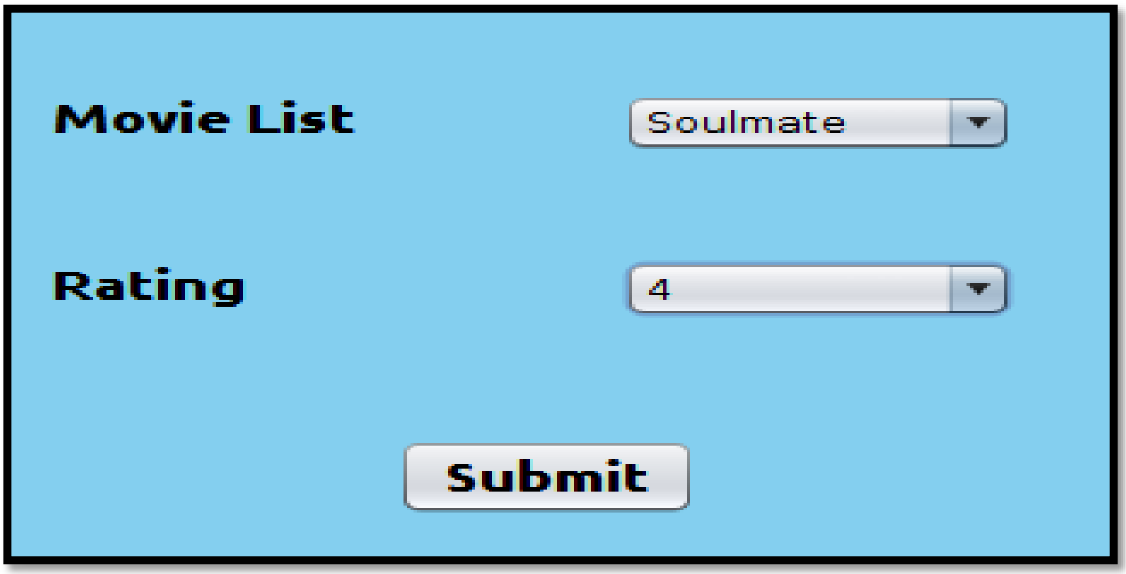


Figure 2- list of movies for rating by the user.

Step 3:- the system measures the explicit feedback for the user on each movie based on movies that are rated in step 2, user profile information, and sign-up process. The reader can notice the text box at the bottom of Figure-3

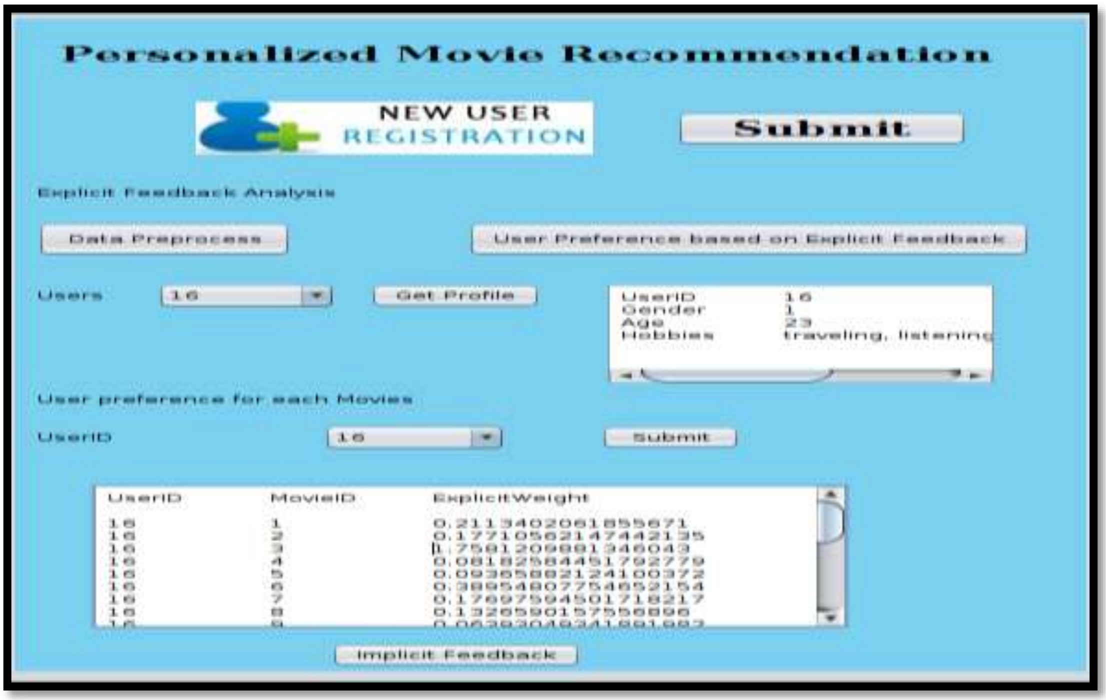


Figure 3- calculate the explicit feedback score for user id 16.

Step4:- before calculating the implicit feedback score for each user, the proposed system extracts movie feature from Wikipedia Infobox (director, writer, starring, etc.), as in Figure-4:

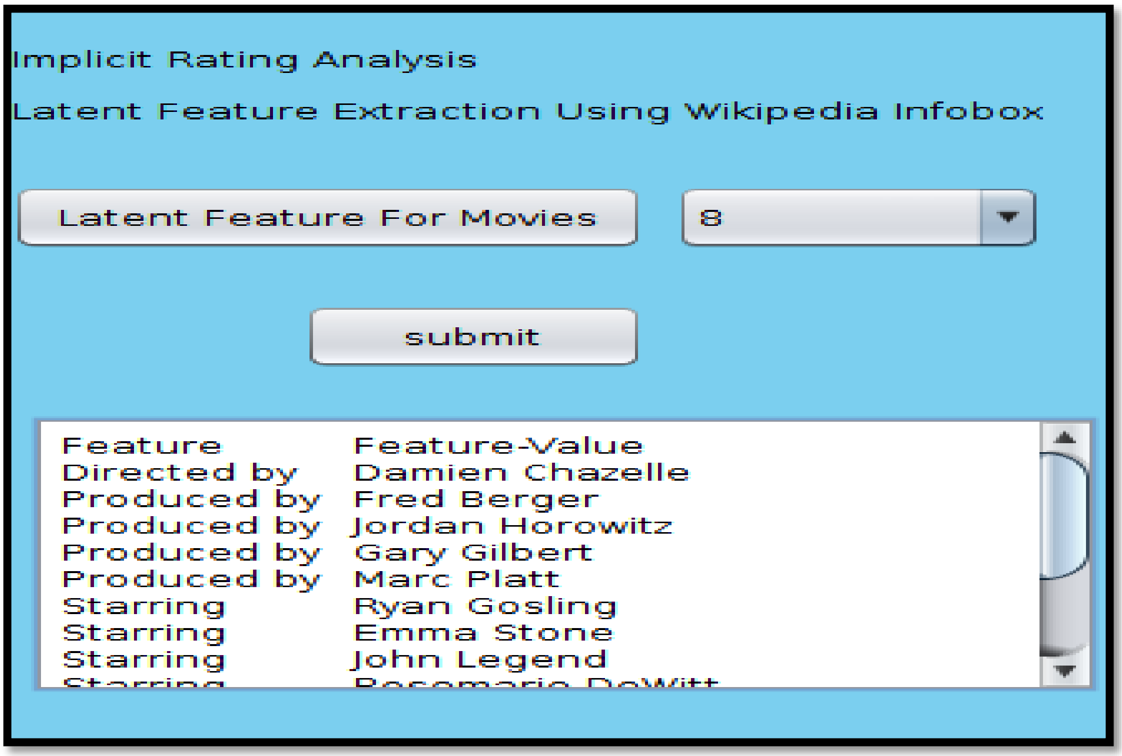


Figure 4- scrap movie features from Wikipedia

Step5:- by analyzing textual reviews of each user on the corresponding movie, the proposed system compute implicit score as in Figure-5. It uses sentiment analysis to contextually identify the preferences of each user on the movie which help to determine the intention behind the user ratings.

Figure 5- Compute an implicit score based on implicit feedback.

Step6:- calculate the latent score of user- movie pair by exploiting both explicit and implicit feedback, using the equation (3)
Step 7:- the proposed system applies k-means clustering on the latent score of user-movie pair.
Step8:- re-cluster the clusters on movie genres (Action, Animation, Comedy…etc.).
Step9:- find the dissimilar users using k-furthest neighbor, form the clustered result the system determines the different users and their movies which have negative scores using the k-furthest neighbor method, those movies are considered as a most preferred movie to user id =10, as shown in Figure-6 below

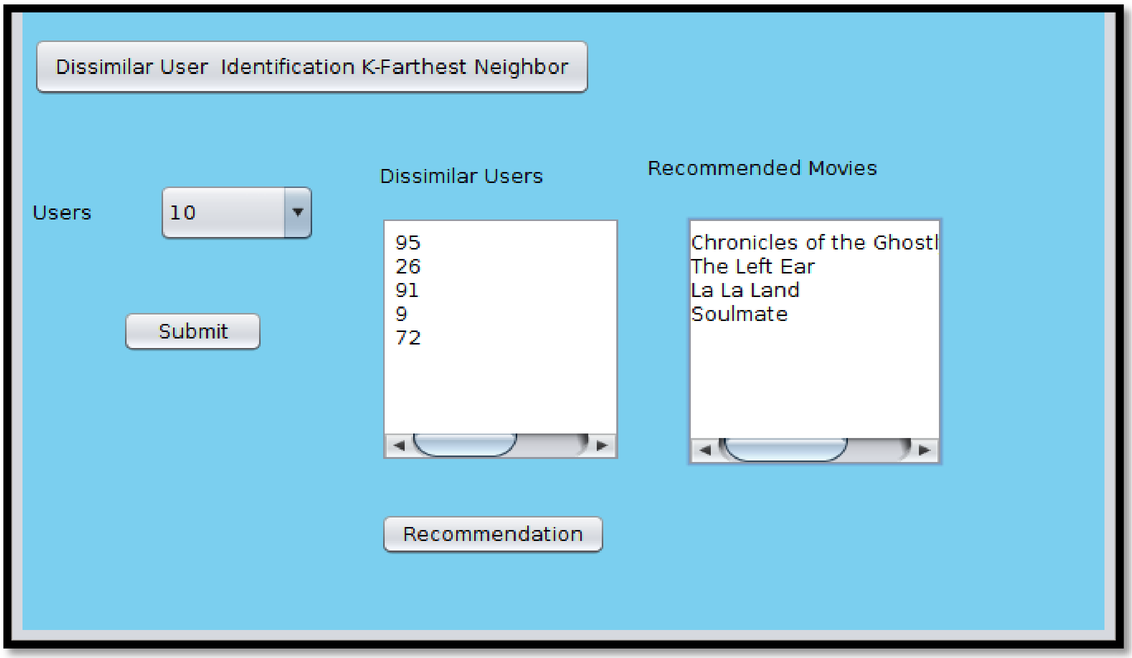


Figure 6- identify the dissimilar user to user number 10.

Step10:- in Figure-7 the system use ranking algorithm to recommend a list of movies using the latent score and cluster result. It calculates the diversity score by equalizing the targeted user preferences with k-furthest neighbor user's preferences. This step illustrated in the algorithm (5)

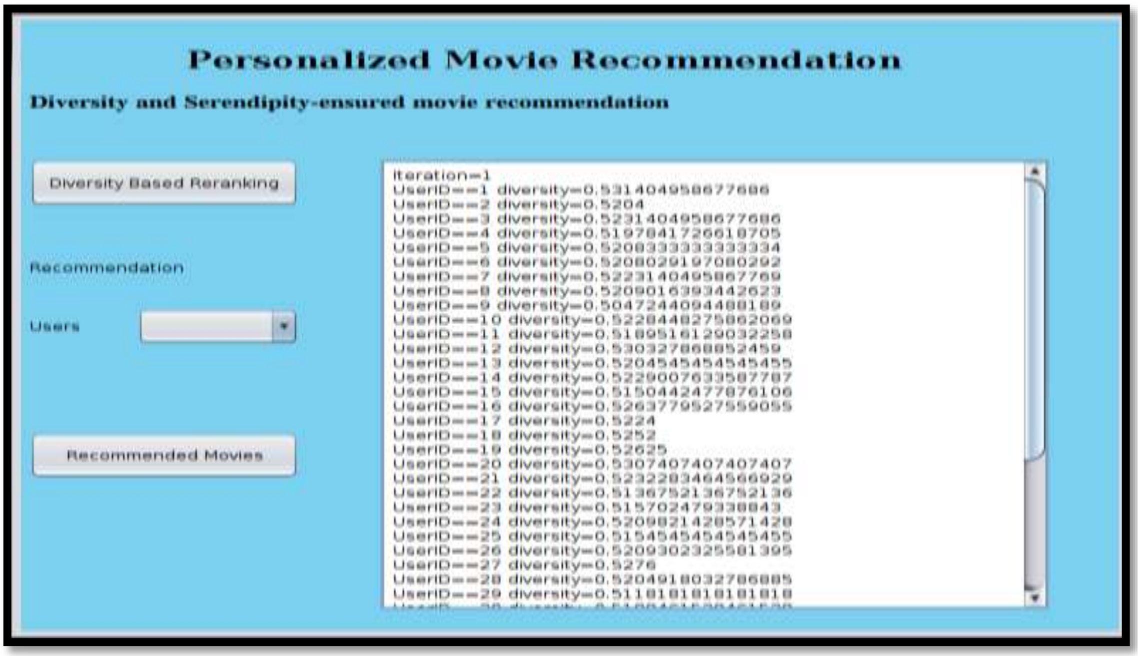


Figure 7- diversity score ranking for movies to the user.

Step11:- Finally the proposed approach suggests a list of movies to the user, which satisfies the diversity and serendipity ensured movies on the recommendation list. See Figure-8

Figure 8- the recommendation list

8. Results

The proposed system architecture design was tested using Douban movie short comments dataset from kaggle using 25 movies and 300 users. For this recommender system design the accuracy is measured using three measurements which are precision as in equation (5), recall in equation (6), and also we need to measure the diversity as follows

$$precision = \frac{true\ positive}{true\ positive + false\ positive} \tag{5}$$

$$recall = \frac{true\ positive}{true\ positive + false\ negative} \tag{6}$$

$$F - measure = 2 * \frac{precision * recall}{precision + recall} \quad (7)$$

Diversity: - measured by greater dissimilarity of all the movies pairs in the recommendation list of user i. the diversity measured by the threshold that is (0.5) and the result should be less than that threshold.

This experiment result showing that when increasing the number of users, the accuracy also increased, and this can be noticed undoubtedly from the precision Figure-6.

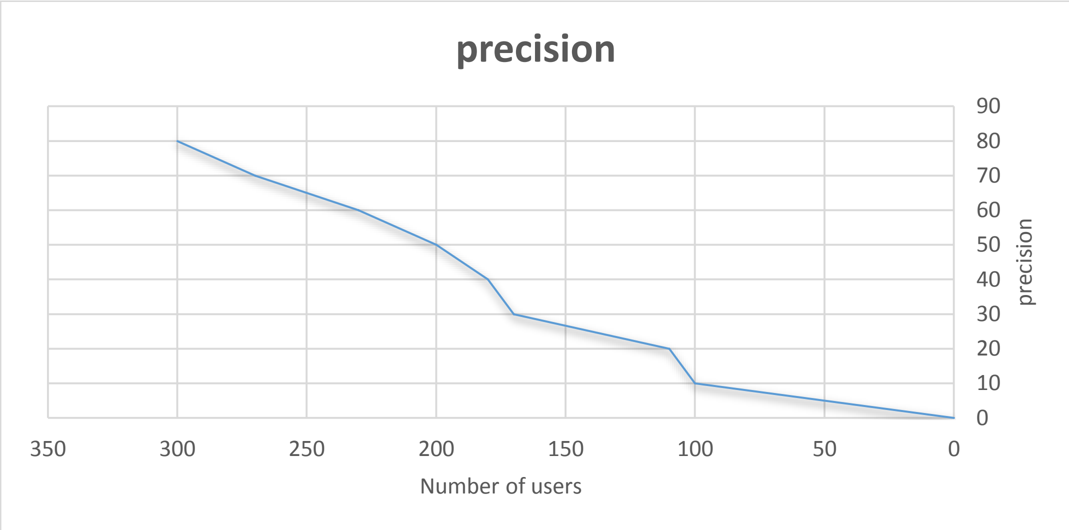


Figure 6-The precision graph.

By applying the k-means clustering, the approach clusters the most similar users based on their explicit rating and their intention on movies (latent features). Thus the recall increases as the number of users increased because identical users would be increased gradually that make the recommendation result more accurate. See Figure-7

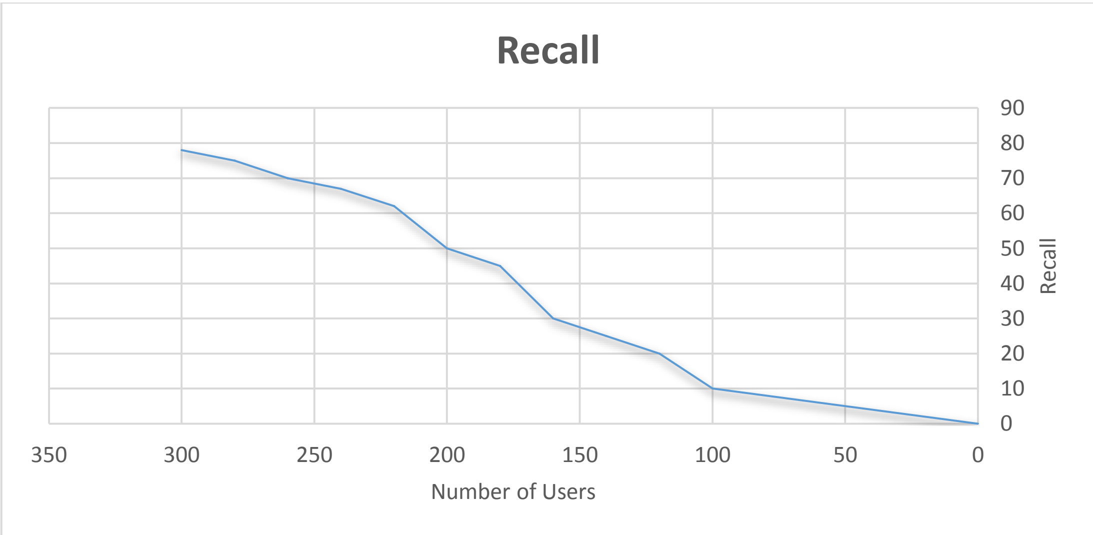


Figure 7-Recall graph.

So, the result table for the proposed recommender system is below. The reader noticed that the system reaches a good compromise between precision, recall, and diversity to avoid the overfitting, see Table- 1.

Table 1-The results of precision, recall, F-measure and diversity

Precision	Recall	F-measure	diversity
80.0	78.0	78.9	50.0

9. Conclusion

The proposed system shows as good performance according to precision and recall measurements. This proposed architecture design exhibit the ability of implicit feedback from user sentiment reviews to be exploited in understand user preferences, which lead to a better personalization. The system fulfills the diversity by clustering users according to their choices, which lead to diversification the result using k-furthest neighbor algorithm. Diverse and serendipitous recommendations ensured by taking into consideration items chosen from farther related users. Lastly, this proposed system tackle the new user cold start problem by registration; this issue also managed by approximate the rate of the movie for the user that he didn't rate using explicit rating method, that takes rate similarity between users with demographic user characteristics (age, and gender). This work handles the new item cold start issue by scrapping the movies information from Wikipedia. The use of Wikipedia to treat cold start problem open a chance to use crowdsourcing sites and services to manage cold start and leakage of information to recommender systems, and other information systems. The proposed system architecture can be tested in the future with content-based recommender system, and with context-aware approaches like location-based recommender system.

References

1. Isinkaye, F., Folajimi, Y. and Ojokoh, B. **2015.** Review recommendation systems: principles, methods, and evaluation. *Egyptian informatics journal*, **16**: 261-273.
2. Hu, Y., Koren, Y. and Volinsky, C. **2008.** Collaborative filtering for implicit feedback datasets. Proceeding ICDM 08 of eight IEEE international conference on data mining: 263-272.
3. Chamsi, R. **2015.** On enhancing recommender system by utilizing general social networks combined with users goals and contextual-awareness. Ph.D. Dissertation, computer science dept.University of Lyon.
4. Ricci, F. Rokach, L. and Shapira, B. **2011.** *Introduction to recommender system handbook*, Springer.
5. Hoan, L. **2014.** Dealing with new user cold start problem in recommender systems: a comparative review. *Information system journal*.
6. Alahmadi, D. and Zeng, X. **2015.** ITS: Implicit social trust and sentiment approach to recommender systems. *Expert systems for applications journal*.
7. Ashok, M., Rajanna, S. and Vineet, P. **2016.** A personalized recommender system using machine learning based sentiment analysis over social data. IEEE student's conference on Electrical, Electronics and computer science. DOI: 10.1109/SCEES.2016.7509354.
8. Jawaheer, G., Szonmszor, M. and Kostkova, P. **2010.** Comparison of implicit and explicit feedback from an online music recommendation service. Proceeding of the 1st international workshop on information heterogeneity and fusion in recommender systems. .Spain .pages:47-51.
9. Ayad, R. and Ashor, S. **2016.** Design recommender system in e-commerce site. *Iraqi journal of science*, **57**(4A): 2549-2556.
10. Martins, R. and Garcia, M. **2014.** A collaborative filtering approach based on the user's reviews. In the proceeding of Brazilian conference on intelligent systems. DOI: 10.1109ABRACIS.2014.45.
11. Douglas, W. and Kim, J. **1998.** Implicit feedback for recommender system. In proceeding of 5th DELOS workshop on filtering and collaborative filtering, pp: 31-37.
12. Krim, A., Bhaa, A. **2017.** Reviews sentiment analysis for collaborative recommender system. *Kurdistan journal of applied research*, **2**(3).
13. Terzi, M., Row, M., Ferrario, M. and While, j. **2014.** Text-based user K-NN: measuring user similarity based on text reviews. In proceeding of the 22nd international conference, Denmark, pp: 195-207.
14. Lak, P. and Turetken, O. **2014.** Star rating versus sentiment analysis – a comparison of explicit and implicit measures of opinions. In Proceedings of the 47th Hawaii international conference on system science. Waikoloa, DOI: 10.11096HICSS.2014.106.
15. Salehi, N., Ibrahim, R. and Ghoreashi, S. **2012.** Product feature extraction using NLP techniques. *Journal of computing*. **4**(7): 39-43.
16. Htay, S. and Thidar, K. **2013.** Extracting product features and opinion words using pattern knowledge in customer reviews. Doi: http://dx.doi.org/10.1155/2013/394758