EAI Endorsed Transactions
on Context-aware Systems and Applications
Research Article EAI.EU

Similarity Kernel for User-based Collaborative Filtering Recommendation System

Nghia Quoc Phan[1], Phuong Hoai Dang[2], Hiep Xuan Huynh[3]

[1] Travinh University, Nguyen Thien Thanh Street, Travinh City, Vietnam, nghiatvnt@gmail.com
[2] Danang University of Science and Technology, Nguyen Luong Bang St, Danang City, Vietnam, dhphuong@dut.udn.vn
[3] Cantho University, 3/2 Street, Ninh Kieu District, Cantho City, Vietnam, hxhiep@ctu.edu.vn

Abstract

This paper is to propose a new similarity kernel function for User-based collaborative filtering recommender system (UBCF). The similarity kernel function for two users is based on Chi-Square kernel. It is called Chi-Square similarity kernel (CSSK). From this similarity kernel function, we build the model for User-based collaborative filtering recommender system. Through experiments on two dataset MovieLense and MSWeb, it shows that the model using our similarity kernel function has accurate results compared with User-based collaborative filtering model using traditional similarity measures as Pearson correlation, Cosine similarity, and Jaccard.

Keywords: Similarity measures, User-based collaborative filtering recommender system, Chi-Square similarity kernel.

Received on 06 September 2016, accepted on 06 April 2017, published on 06 July 2017

Copyright © 2017 Nghia *et al.*, licensed to EAI. This is an open access article distributed under the terms of the Creative Commons Attribution licence (http://creativecommons.org/licenses/by/3.0/), which permits unlimited use, distribution and reproduction in any medium so long as the original work is properly cited.

doi: 10.4108/eai.6-7-2017.152759

1. Introduction

Recommender systems are software tools and techniques to provide recommendable information to users such as the information of useful products on commercial sites, interesting videos on YouTube, friends with the same interests on Facebook, specialized books on Amazon, and interesting news on online news sites [24][26]. The main purpose of the recommender system is to provide useful recommendations for online users in order to help them make better decisions based on multiple sources of data which are provided on web pages. According to the trend of development, a recommender system must propose recommendations that meet the personal interests of users based on online user profile and information about products such as the specifications of the products compared to other products of the same type, the feedback of the users and other information before making recommendations[10][12][22]. The recommender systems are primarily developed based on the effective exploitation of statistical methods [11] and knowledge discovery techniques on the transaction dataset of customer management online systems.

Currently, recommender system is considered a useful tool for solving information overload of the Internet [6][19]. Its development is always associated with the development of web technologies and machine learning algorithms [34]. Based on the method of collecting and processing data, the recommendation system is divided into three generations. The first generation of recommender systems uses traditional websites in order to gather information from three sources: (1) content-based data from the purchase or the use of products and services; (2) demographic data selected from the customer profile; (3) memory-based data collected from the user's preferences. In this generation, the quality of recommendation results is improved based on data classification algorithms and their integration [6][25][39]. The second generation of recommender systems is the increasing use of Web 2.0 by collecting information through social networks like Facebook, Zalo and other social networking sites. To satisfy information explosion from social networking sites, this generation continues to develop and improve the existing integrated methods and enhance solutions to exploit information from social networks more efficiently such as trust-aware algorithms [30][35], social adaptive approaches [7], social networks analysis [19][40], and other methods. The third generation of recommender systems is developed in parallel with the web 3.0 with information collected from integrated devices on the Internet such as cameras, sensors [35]. This generation uses

* Email: nghiatvnt@gmail.com

approaches of the integration of location information into the available recommendation algorithms in order to broaden its application in various fields such as health, weather, environment and universe [3].

From the first appearance with the name "The information Lense system" in 1987 [21] recommender system has been developed greatly in technology and its application in life. In particular, recommender systems are used by many managers as an effective tool in order to support the business in various fields such as Amazon, Netflix, and Pandora [5]. However, the present generation of recommender systems has not fully met the requirements of users yet [1]. Therefore, research on recommender systems has still been concerned such as research to improve methods and algorithms to increase accuracy of the existing recommender model [28], research to improve recommender systems to adapt to the information explosion and research to propose a new recommender model [14][17]. In addition, some new research directions are also set out as: research to proper combination of existing recommendation methods that use different types of available information; research to use the maximum capabilities of the sensors and devices on the Internet; research to collect and integrate information related to habits, consumption and individual tastes of users in the recommendation process; research to ensure the security conditions and privacy in the entire process of recommendation system; research to propose the measures for evaluating recommender systems and develop a standard for assessment measures and research to develop a framework for automated analysis on heterogeneous data.

In this paper, basing on the traditional model of UBCF using similarity measures: Pearson, Cosine, Jaccard, we propose a new approach for UBCF. The UBCF is based on Chi-Square kernel [4][32][33]. In this recommender system, we build a similarity kernel function based on Chi-Square kernel in order to determine the similarity between two users. It is named Chi-Square similarity kernel that substitutes the similarity measures in the system. The UBCF model using the CSSK is built and conducted through experiments on two datasets MovieLense [8] and MSWeb [18] and its results are also compared to UBCF model using the traditional similarity measures.

This paper is organized into six parts. Part one introduces general recommender systems, relevant studies and introduction to research. Part two presents UBCF. Part three shows how to build a similarity kernel between two users based on Chi-Square kernel. Part four describes the required steps to build UBCF model based on CSSK and presents the evaluation methods of recommender systems. Part five presents the experimental results of the model and compares the results with other models. The final part summarizes some important achieved results.

2. User-based collaborative filtering recommender system

UBCF is the first version of the recommender systems based on collaborative filtering. It was first introduced in the article "GroupLens: an open architecture for Collaborative filtering of netnews" in 1994 for GroupLens Usenet recommender system [31]. Subsequently, two other systems are also used this recommendation method: one for the users to listen to music Ringo [37][38] and the other for users to watch movies Bellcore [38]. UBCF is a simple algorithm to clarify the core premise of collaborative filtering methods. That is to find out users in the past who have similar behavior with current users. Then, we use the value of their rating for the items to predict the preferences of the current users. Thus, in order to obtain a list of items to introduce to new users, UBCF requires a function to compute the similarity of two users and a method to calculate the average deviation of rating values of the similar users based on a rating matrix of users for items [24][25][26].

2.1. Compute the similarity between two users

Selecting measures to calculate the similarity between two users is an important stage in the design of UBCF since it directly affects the outcome of the recommender system. Currently, research in the field of machine learning, many measures are proposed for this purpose. In which, Pearson correlation, Cosine, and Jaccard similarity are three measures used by many recommender systems.

Pearson correlation is the similarity measures between two users based on statistical correlation [24][25][34]. The similarity measures between user u and user v are determined by the following formula:

$$S(u,v) = \frac{\sum_{i\in I}(r_{v,i} - \bar{r_v})(r_{u,i} - \bar{r_u})}{\sqrt{\sum_{i\in I}(r_{v,i} - \bar{r_v})^2 \sum_{i\in I}(r_{u,i} - \bar{r_u})^2}}$$

Where $S(u,v)$ is the similarity value between user u and user v; I is a set of items rated by both users; $r_{v,i}$is the rating of user v for item i; $\bar{r_v}$ is the average rating value of user v; $r_{u,i}$ is the rating of user u for item i; $\bar{r_u}$ is the average rating value of user u.

Cosine similarity is a similarity measures between two users based on vector space linear algebra [24][25][34]. The rating values of each user on m items are represented by m-dimensional vectors. The similarity of two users u and v is determined by the distance Cosine between two vector $\vec{r_u}$ and vector $\vec{r_v}$ by the following formula:

$$S(u,v) = \cos(\vec{r_u}, \vec{r_v}) = \frac{\vec{r_u} \,.\, \vec{r_v}}{\|\vec{r_u}\|_2 X \|\vec{r_v}\|_2}$$

$$S(u,v) = \frac{\sum_{i=1}^{m} r_{u,i} r_{v,i}}{\sqrt{\sum_{i=1}^{m} r_{u,i}^2} \sqrt{\sum_{i=1}^{m} r_{v,i}^2}}$$

Where $S(u,v)$ is the similarity value between user u and user v; m is the dimensionality of the vector (number of items); $r_{u,i}$ is the rating of user u for item i; $r_{v,i}$ is the rating of user v for item i.

Jaccard is a measures which is used to determine the similarity between two users based on operations between two finite sets [26]. This measures is proposed to handle problems when the rating value of users for one item in the rating matrix is ignored and assigned a value equal 0. In this measures, the set of items that user u rating will be built into a corresponding set of user u profiles. The similarity of two users is calculated by the size of the intersection operation on two sets (∩) divided by the size of the union operation on two sets (∪). The similarity value between user u and user v is calculated by the following formula:

$$S(u,v) = \frac{|A \cap B|}{|A \cup B|}$$

Where $S(u,v)$ is the similarity value between user u and user v; A is a set representing for the rating values of user u for the list of items; B is a set representing the rating values of user v for the list of items.

2.2. Compute the recommendation results

In order to calculate the recommendation results for user u, in the first step, UBCF uses the similarity measures to find out N users who are similar with user u. As the list of N similar users is defined, the system will combine their rating values to generate predictions of user u's preference for an item i. Typically, the predictable results are based on the average weighted of rating values of N similar users, represented by the following formula [24][25]:

$$P(u,i) = \bar{r}_u + \frac{\sum_{u' \in N} S(u,u')(r_{u',i} - \bar{r}_{u'})}{\sum_{u' \in N} |S(u,u')|}$$

3. CSSK for two similar users

3.1. Kernel as similarity measures

Kernel theory views a kernel as implicitly mapping data points into a possibly very high dimensional space, and describes a kernel function as being good for a given learning problem if data is separable by a large margin in that implicit space. Furthermore, Kernel functions have become a popular useful tool in machine learning [2][23][29]. A kernel is a function that takes in two data objects such as images, DNA sequences, or vectors with m-dimensional and outputs a number, with the property that the function is symmetric and positive-semidefinite. That is, for any kernel K, there must exist an implicit mapping φ, such that for all inputs x_1, x_2 we have $K(x_1, x_2) = \varphi(x_1) \cdot \varphi(x_2)$ [36]. The kernel is then used inside a "kernelized" learning algorithm such as SVM or kernel-perceptron as the way in which the algorithm interacts with the data. Typical kernel functions for structured data include Linear Kernel, Polynomial Kernel, Gaussian Kernel, Chi-Square Kernel, and so on [2][4].

3.2. Similariy kernel for two similar users

Suppose that $U = \{u_1, u_2, \dots, u_n\}$ is a set of n users, $I = \{i_1, i_2, \dots, i_m\}$ is a set of m items, $R = \{r_{j,k}\}$ is a rating matrix of n users for m items with each row representing a user u_j $(1 \le j \le n)$, each column represents an item i_k $(1 \le k \le m)$, $r_{j,k}$ is the rating value of user u_j for item i_k.

	i_1	i_2	i_3	.	.	.	i_m
u_1	0	4	4	3	2	1	0
u_2	3	0	0	2	3	5	0
u_3	4	0	0	1	1	4	3
.	3	2	0	3	4	3	4
.	0	3	1	4	3	2	1
	2	1	4	0	3	0	5
u_n	2	0	0	3	1	4	1

Figure 1: Rating matrix of users for items

From the rating matrix, the rating values of each user on m items is represented by m-dimensional vector $u_j(r_{j,1}, r_{j,2}, \dots, r_{j,m})$. The similarity of two users u_i and u_j is determined by Chi-Square similarity kernel following formula [4]:

$$K(u_i, u_j) = \sum_{k=1}^{m} \frac{2\, r_{i,k}\, r_{jk}}{(r_{i,k} + r_{j,k})}$$

Where $K(u_i, u_j)$ is the similarity value between user u_i and user u_j; m is the dimensionality of the vector (number of items); $r_{i,k}$ is the rating of user u_i for item k; $r_{j,k}$ is the rating of user u_j for item k.

4. UBCF based on CSSK

From the steps to build UBCF [24][25][26] and the similarity kernel function to determine the similar between two users based on Chi-Square kernel presented in the above section, we propose the steps to build UBCF using CSSK as follows:

4.1. Build recommendation algorithm

In this algorithm, as a new user needs recommendation, the system will use the CSSK function to find out the list of similar users with the new user. Then, a list of high rated items to introduce a new user is calculated based on the rating value of users. Recommendation algorithm shall comply with the following steps:

Step 1: Update or build a new user profile.

Step 2: Find the list of similar users:

- Choose coefficient k for determining the list of k similar users to the new user.
- Identify the list of similar users based on CSSK.

Step 3: Find a list of items which is rated highly by k similar users:

- Calculate the average rating value for each item.
- Sort the list of items based on the average rating value.

Step 4: Select N items having the highest rating as the recommendation results.

In order to see more clearly the functions of the algorithm, we present the recommendation algorithm by example shown in Figure 2. In this figure, we assume that the system recommends 8 items (from i_1 to i_8) to users and the system has 10 users (from u_1 to u_{10}) rating for the items. The items are rated from 1 to 5 ("1" - the lowest rating; "5" - the highest rating; "?" - Users are not rated for the items). The system is required to recommend the items to new user ua, with coefficient k, which is given by 4 (4 similar users). From this requirement, the system finds the list of 4 similar users to user u_a: u_1, u_4, u_7, and u_8 to figure out the rating values that the user u_a ignores and to determine the list of items that user u_a will like.

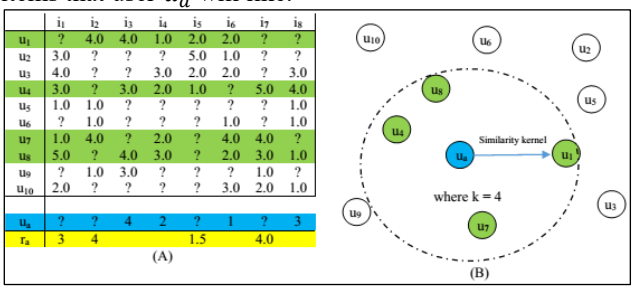

Figure 2: User-based collaborative filtering example with (A): Rating matrix and estimated ratings for user u_a; (B): Identifying similar users with new users.

4.2. Evaluating the recommender model

The evaluation of the accuracy of the recommender model is an important step in the recommender system design process [9][13][15]. It helps designers choose models, check the accuracy of the model before applying the model into practice. To evaluate UBCF model, the recommender system designers can be conducted through two steps:

4.2.1. Preparing data to evaluate the models

In order to evaluate recommender model, we need to build on some data and test it on some other data. Therefore, the first step is to prepare the data. In this step, the experimental dataset is divided into two subsets: training dataset and testing dataset [26]. Currently, many methods are being used to split datasets for evaluating recommender models such as:

Splitting: is the initial method to build a training set and test set by cutting experimental dataset into 2 parts [26]. For this method, the model designer should decide the percentage for the training set and test set. For example, the training set accounts for 80 percent and the test set does the remaining 20 percent.

Bootstrap sampling: is method to build a training set and test set by cutting experimental dataset into 2 parts. However, this approach is conducted randomly and repeatedly in order that a user may be a member of the training set in this cutting time but is a member of test set in the next cutting time. This can overcome the disadvantages about heterogeneity of experimental dataset and increase the optimization for small-sized dataset [26].

K-fold cross-validation: is a method to build a training set and test set by cutting experimental dataset into k subsets with the same size (called k-fold). After that, the model is evaluated k times. Every evaluation uses one subset for test set and the k-1 subsets is used as the training set. The evaluation results of this method are average value of k evaluations. This approach ensures that all users have appeared at least one time in test set [26]. Therefore, it is the most accurate of the three methods. However, it is costly for the calculation compared with the remaining two methods.

4.2.2. Evaluate recommender model

There are two methods for evaluating recommender model: evaluation based on the ratings and evaluation based on the recommendations. The first method evaluates the ratings generated by the model. The second method evaluates directly on the recommendations of the model.

Evaluation based on the ratings: this method evaluates the accuracy of the model by comparing the predicted rating value to the real value. More precisely, this method is to find out the average error value based on three indicators RMSE, MSE, and MAE. A model is evaluated good if these indicators show low value [13][15].

Root mean square error (RMSE): This is the standard deviation between the real and predicted ratings.

$$RMSE = \sqrt{\frac{\sum_{(i,j)\in\kappa}(r_{ij} - \hat{r}_{ij})^2}{|\kappa|}}$$

Mean squared error (MSE): This is the mean of the squared difference between the real and predicted ratings. It's the square of RMSE, so it contains the same information.

$$MSE = \frac{\sum_{(i,j)\in\kappa}(r_{ij} - \hat{r}_{ij})^2}{|\kappa|}$$

Mean absolute error (MAE): This is the mean of the absolute difference between the real and predicted ratings.

$$MAE = \frac{1}{|\kappa|}\sum_{(i,j)\in\kappa}|r_{ij} - \hat{r}_{ij}|$$

Where κ is the set of all user ratings for items; r_{ij} is the real rating value of user i for item j; $\hat{r}_{ij}$ is the predicted rating value of user i for item j.

Evaluation based on the recommendations: this method evaluates the accuracy of the model by comparing the model's recommendations with the choice of user's purchase. This approach uses confusion matrix to calculate the value of three indicators: Precision, Recall, and F-measure. The model is evaluated good if three indices gain high value [13][15].

Table 1: Confusion matrix

User Choices	Recommendations of the model	
	Recommend	Not recommend
Purchase	TP	FN
Not purchase	FP	TN

In which:

True Positives (TP): recommended items that have been purchased.
False Positives (FP): recommended items that have not been purchased.
False Negatives (FN): unrecommended items that have been purchased.
True Negatives (TN): unrecommended items that have not been purchased.
The formula of three indicators is used to evaluate:

$$\text{Precision} = \frac{\text{Correctly recommended items}}{\text{Total recommended items}} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

$$\text{Recall} = \frac{\text{Correctly recommended items}}{\text{Total useful recommendations}} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

$$\text{F} - \text{measure} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

5. Experiment

5.1. Data description

In this experiment, we use two different datasets to test the model on two different scenarios. In the first scenario, we use the MovieLense dataset [8] of GroupLens research project at the University of Minnesota in 1997. In the second scenario, we use the MSWeb dataset of Microsoft published in 1998 [18].
In the first scenario, we carried out experiments on MovieLense dataset. This dataset is collected from the rating results of 943 users for 1,664 movies (99,392 rating results from 0 to 5) through the MovieLense website (movielens.umn.edu) during 7 months (from 09/19/1997 to 22/04/1998). This dataset is organized in a matrix format consisting of 943 rows, 1,664 columns, and 1,569,152 cells containing rated values. However, each user just watches her/his favourite movies. Thus, the rating matrix has only 99,392 rating values of users for movie categories.
In the second scenario, we carried out experiments on MSWeb dataset. This dataset is of users visiting Microsoft sites during one week in February 1998. It is sampled and processed from the log file of the address www.microsoft.com. This dataset included 38,000 anonymous users getting access to 285 original web addresses, and is processed and organized into binary matrix with 32,710 rows, 285 columns and 98,653 rating values.

5.2. Implementation tools

In order to conduct experiment, we use ARQAT tool which is developed on language R by our team [20]. This is a tool package to be developed from engine platform ARQAT on language Java [16]. This tool includes the following functions: processing data, calculating similarity of two users based on CSSK [2], and designing and evaluating recommender models [27].

5.3. Scenario 1

5.3.1. Select and process data

The MovieLense dataset is stored under a real rating matrix. It consists of 943 rows, 1,664 columns and 1,569,152 cells containing rated value. In particular, more than 93 percent cells have rating values equal 0 and nearly 7 percent remaining cells have rating values from 1 to 5 (value 0 is 1,469,760; value 1 is 6,059; value 2 is 11,307; value 3 is 27,002; value 4 is 33,947; value 5 is 21,077). Therefore, the entire MovieLense dataset has only truly 99,392 rating value from users for movies. In particular, the majority of rating values range from 3 to 5 and 4 is rating value with the highest amount. In order to find out the number of users rating for each movie and the number of movies that each user rated, we do statistics by each movie, by each user and illustrate the statistical results in Figure 3.

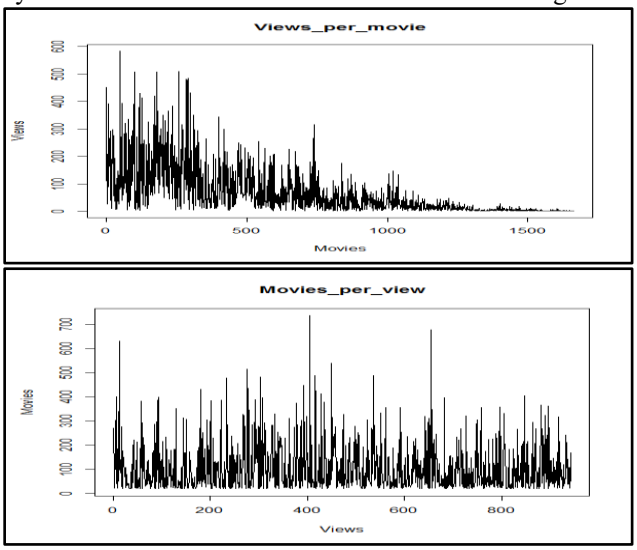

Figure 3: The chart presents statistical results for each movie and each user on MovieLense dataset.

Figure 3 reflects that some movies have only rated by a few users and some users have only rated for a few movies. If we use this case to training model, it is likely to lead to bias due to lack of data. Thus, we decided to select users with the least rating for 50 movies and movies rated by at least 100 users in order to build experimental datasets for model. From there, rating matrix has only 560 rows, 332 columns, and 55,298 rating values. In particular, we split the dataset into two subsets with training set and test set accounting for 80 percent and 20 percent respectively.

5.3.2. The result of the model

The objective of this scenario is to test the accuracy of the model on real number rating matrix. Therefore, we build the model on the training set with 449 users and test the model on test set with 111 users. The result of the model is

exported in matrix format with structure 10 x 111 (each column is a user; each cell is a selected movie to recommend for the user in the corresponding column). Figure 4 presents the results of recommender model to the first 4 users; each of them selects the 10 highest rated movies.

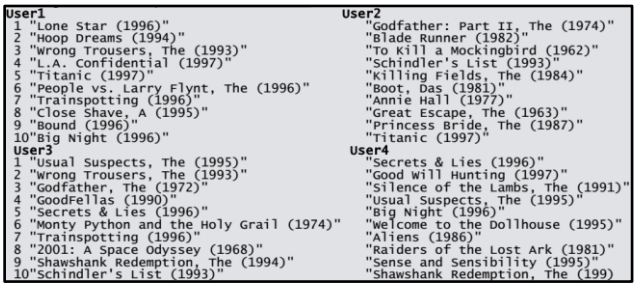

Figure 4: Recommendation results of the first 4 users

Based on the recommendation result matrix, we calculate the number of times that each movie is recommended and build a histogram for the distribution of movies in Figure 5. The chart shows that the number of movies is recommended from 5 times or less accounting for relatively large numbers. In particular, up to 38 movies are only recommended 1 time and 26 movies are recommended 2 times. In contrast, the number of movies is recommended from 5 or more times accounting for a relatively small amount. Most of them have the number from 1 to 2 movies. Of these, two movies are introduced up to 41 times.

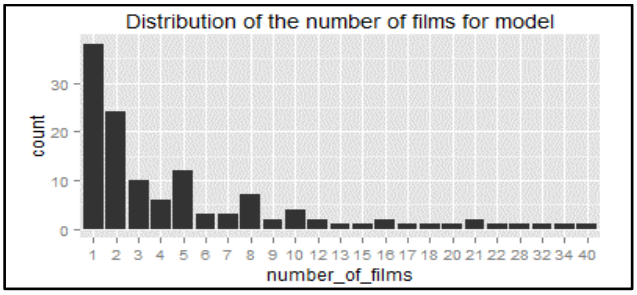

Figure 5: Distribution of the number of movies for the model.

5.3.3. Evaluation the model

Evaluation based on the ratings

In this section, we calculate the error parameters (RMSE, MSE, MAE) for each user and for the model based on the data which is built by k-fold method (with k=5). For the error parameters of each user, we perform the distribution of each error parameters by chart and compare those with the error parameters of the model using similarity Pearson measures (Figure 6). The chart shows that the number of users distributed on the error parameters of the model using CSSK has a higher value than that of the model using similarity Pearson measures. For the error parameters of the model, we continue to compare with the error parameters of the model using Pearson similarity measures in table 3. From the results of comparison, we found that the values of error parameters of our model are lower than the model using similarity Pearson measures on MovieLense dataset.

Table 2: Present comparation error parameters of two models

	RMSE	MSE	MAE
Model using similarity kernel	0.8961562	0.8030960	0.7077939
Model using similarity Pearson measures	0.9796664	0.9597462	0.7704055

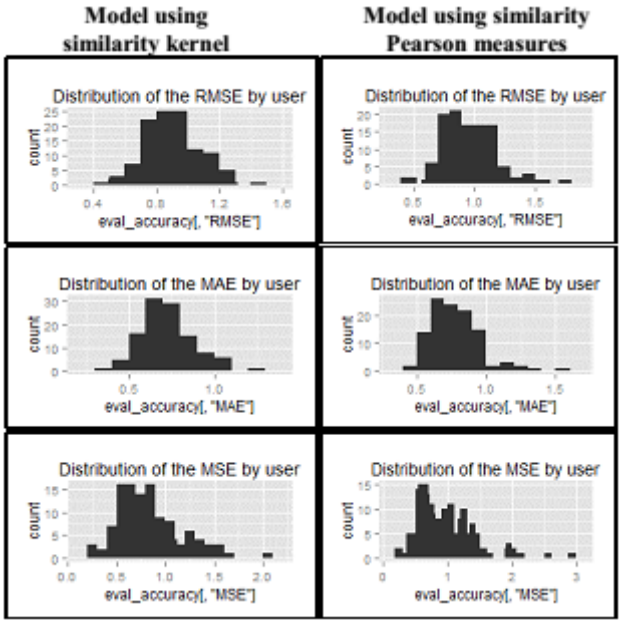

Figure 6: Comparison of error parameters of each user on two models.

Evaluation based on the recommendations

In this evaluation method, we calculate the index TP, FP, FN, TN, Precision, Recall, and F-measure corresponding to 5 different k-folds on both models: model using CSSK and model using similarity Pearson measures and compute the average value of each index on the 5 k-fold (Figure 7). The results show that the value of two indices Recall and F-measure of model using CSSK is higher than the corresponding indices in models using similarity Pearson measures. However, the average value of Precision index is in reverse tendency.

Similarity kernel

	TP	FP	FN	TN	precision	recall	f-measure
Fold1	5.714286	4.285714	78.6339	228.3661	0.5714286	0.0817825	0.1430865
Fold2	5.401786	4.598214	63.0625	243.9375	0.5401786	0.0930069	0.1586907
Fold3	5.294643	4.705357	60.8839	246.1161	0.5294643	0.0923489	0.1572673
Fold4	4.892857	5.107143	62.1160	244.8839	0.4892857	0.0823349	0.1409511
Fold5	5.607143	4.392857	73.6160	233.3839	0.5607143	0.0777199	0.1365173
Avg	**5.382143**	**4.617857**	**67.6625**	**239.3375**	**0.5382140**	**0.0854390**	**0.1473030**

Similarty Pearson measures

	TP	FP	FN	TN	precision	recall	f-measure
Fold1	4.991071	5.008929	66.51786	240.4821	0.4991071	0.0794999	0.1371534
Fold2	5.437500	4.562500	66.05357	240.9464	0.5437500	0.0879665	0.1514343
Fold3	5.223214	4.776786	63.08036	243.9196	0.5223214	0.0906580	0.1544999
Fold4	5.544643	4.455357	75.33036	231.6696	0.5544643	0.0774621	0.1359335
Fold5	5.732143	4.267857	67.47321	239.5268	0.5732143	0.0908563	0.1568512
Avg	**5.385714**	**4.614286**	**67.69107**	**239.3089**	**0.5385714**	**0.0852886**	**0.1472574**

Figure 7: Comparison of indicators based on the recommendations of two models.

5.4. Scenario 2

5.4.1. Select and process data

Binary MSWeb matrix has relatively big size with 32,710 rows, 285 columns, and 98,635 rating values. However, the study finds that plenty of users only visited a few sites and many sites are accessible only by a few users. In order to increase the reliability of the results of recommendations of the model, we build a set of data for models in a way that we only select the users to access at least 10 different web address and select the sites accessed at least 50 users. After the selections, we get data matrix for experiment with size 796 x 135. Similar to the first scenario, the experimental data matrix is divided into two subsets: training dataset with size 626 x 135 (80%), test dataset with size 170 x 135 (20%). In order to see the number of times that each user gets access to the websites, we perform binary rating matrix as diagram in Figure 8 (only for the first 50 users).

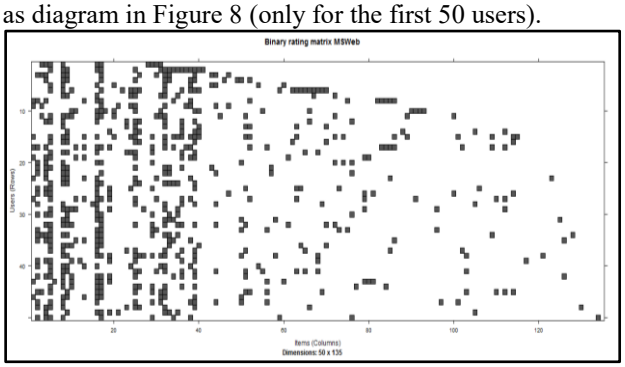

Figure 8: Rating MSWeb matrix for the first 50 users.

5.4.2. The result of the model

With the aim of checking the accuracy of the model on binary rating matrix, we build models according to the CSSK on dataset which is processed above. During this experiment, we choose each user that is introduced on 6 sites where the model predicts that they love those sites. The recommendation results of first 6 users are shown in Figure 9.

User1	**User2**	
1	"Free Downloads"	"Support Desktop"
2	"Support Desktop"	"Knowledge Base"
3	"Microsoft.com Search"	"Internet Site Construction for Developers"
4	"Windows Family of OSs"	"SiteBuilder Network Membership"
5	"Products"	"Windows NT Server"
6	"Developer Workshop"	"Developer Workshop"
User3	**User4**	
1	"Developer Workshop"	"isapi"
2	"Microsoft.com Search"	"Developer Network"
3	"Developer Network"	"MS Office Development"
4	"Windows NT Server"	"OutLook"
5	"Products"	"ActiveX Technology Development"
6	"ActiveX Technology Development"	"MS Office"
User5	**User6**	
1	"Microsoft.com Search"	"Microsoft.com Search"
2	"Free Downloads"	"Windows Family of OSs"
3	"Windows Family of OSs"	"Internet Explorer"
4	"MS Office"	"Windows95 Support"
5	"Internet Site Construction for Developers"	"IT Technical Information"
6	"SiteBuilder Network Membership"	"MS Word"

Figure 9: Presentation of recommendation results on MSWeb dataset for the first 6 users

From recommendation result matrix, we choose 10 websites where the model recommends to the most users (Figure 10a). Among them, the head of the list is the Microsoft search page (Microsoft.com Search) with 74 recommendations to the users. In contrast to the list of 10 top websites, we also extracted a list of 10 websites that have a little recommendation (Figure 10b). Most of these pages only get one recommendation. Except for Microsoft Excel page is chosen to recommend two times.

Names of website	Number of recommendation
Microsoft.com Search	74
Windows Family of OSs	65
Products	63
Support Desktop	63
Internet Explorer	58
isapi	52
Free Downloads	46
Knowledge Base	46
Windows95 Support	44
MS Office Info	39

(a)

Names of website	Number of recommendation
Access Development	1
MS Proxy Server	1
MS Publisher	1
Product Catalog	1
promo	1
sports	1
Training	1
Visual InterDev	1
Windows Hardware Testing	1
MS Excel	2

(b)

Figure 10: The list of 10 websites is recommended at most and the list of 10 websites is recommended at least.

5.4.3. Evaluate recommender model

Since MSWeb matrix is a binary data matrix, the model only is evaluated according to the recommendations. In this evaluation, we continue to use the k-fold approach to build dataset for evaluating the model with k = 5. In order to examine the accuracy of the model, we test the model with the number of recommendation websites for users that are ascending (from 1 to 15). The average rating Results of 5 k-fold on the model using the CSSK and the model using similarity Jaccard measures are shown in Figure 11. In this figure, we see that the Precision index always decreases as the number of recommendation websites increases on both models. In contrast to the Precision index, Recall index always increases on both models as the recommendation websites increase. Unlike the above two indicators, F-measure index reached the highest value as the recommendation websites reached 10 on the model using CSSK and reached 3 on models using similarity Jaccard measures. This suggests that the model using the CSSK gave the best results as each user is recommends to 10 websites and the model using similarity Jaccard measures has the best results as each user is recommends to 3 websites.

Similarity kernel

	TP	FP	FN	TN	precision	recall	F-measure
1	0.71625	0.28375	8.59625	122.4038	0.7162500	0.08123728	0.14592384
2	1.36125	0.63875	7.95125	122.0487	0.6806250	0.15312173	0.25000033
3	1.93000	1.07000	7.38250	121.6175	0.6433333	0.21703467	0.32457189
4	2.45500	1.54500	6.85750	121.1425	0.6137500	0.27605252	0.38081986
5	2.92250	2.07750	6.39000	120.6100	0.5845000	0.32708059	0.41944422
6	3.35875	2.64125	5.95375	120.0463	0.5597917	0.37523776	0.44930132
7	3.72500	3.27500	5.58750	119.4125	0.5321429	0.41540408	0.46658232
8	4.03250	3.96750	5.28000	118.7200	0.5040625	0.44925480	0.47508316
9	4.28250	4.71750	5.03000	117.9700	0.4758333	0.47620971	0.47602143
10	4.53125	5.46875	4.78125	117.2188	0.4531250	0.50272330	0.47663734
11	4.74625	6.25375	4.56625	116.4338	0.4314773	0.52617791	0.47414523
12	4.93500	7.06500	4.37750	115.6225	0.4112500	0.54679581	0.46943429
13	5.12500	7.87500	4.18750	114.8125	0.3942308	0.56698806	0.46508483
14	5.28375	8.71625	4.02875	113.9713	0.3774107	0.58330117	0.45829371
15	5.40500	9.59500	3.90750	113.0925	0.3603333	0.59592619	0.44910833

Similarty Jaccard measures

	TP	FP	FN	TN	precision	recall	F-measure
1	0.3178392	0.6821608	1.9912060	122.0088	0.31783920	0.1830669	0.23232233
2	0.5314070	1.4685930	1.7776382	121.2224	0.26570352	0.2974401	0.28067754
3	0.6846734	2.3153266	1.6243719	120.3756	0.22822446	0.3654355	0.28097337
4	0.8128141	3.1871859	1.4962312	119.5038	0.20320352	0.4284085	0.27565693
5	0.9309045	4.0690955	1.3781407	118.6219	0.18618090	0.4801262	0.26831570
6	1.0213568	4.9786432	1.2876884	117.7123	0.17022613	0.5207650	0.25658162
7	1.0992462	5.9007538	1.2097990	116.7902	0.15703518	0.5550132	0.24480527
8	1.1695980	6.8304020	1.1394472	115.8606	0.14619975	0.5822443	0.23371451
9	1.2324121	7.7675879	1.0766332	114.9234	0.13693467	0.6039365	0.22325027
10	1.2776382	8.7223618	1.0314070	113.9686	0.12776382	0.6231155	0.21204903
11	1.3228643	9.6771357	0.9861809	113.0138	0.12026039	0.6395016	0.20244949
12	1.3768844	10.6231156	0.9321608	112.0678	0.11474037	0.6574864	0.19538362
13	1.4208543	11.5791457	0.8881910	111.1118	0.10929648	0.6778671	0.18824166
14	1.4560302	12.5439698	0.8530151	110.1470	0.10400215	0.6908701	0.18078874
15	1.4874372	13.5125628	0.8216080	109.1784	0.09916248	0.7014221	0.17375992

Figure 11: Comparison of indicators based on the recommendations of two models.

In order to compare accuracy between model using CSSK and model using similar Jaccard measures on the two indices Precision and Recall, we use the chart ROC

(Receiver Operating Characteristic) to draw lines of the Precision and Recall ratios for the above two models. Figure 12 shows that on both models Precision index and Recall index tend to increase and decrease contradictorily. While the Recall index is increasing, in contrast Precision index is declining. The chart shows that the ratio of Precision and Recall on the model using the CSSK is higher than that on models using similarity Jaccard measures. It follows that the model using CSSK has the greater accuracy than that of similarity Jaccard measures on binary MSWeb dataset.

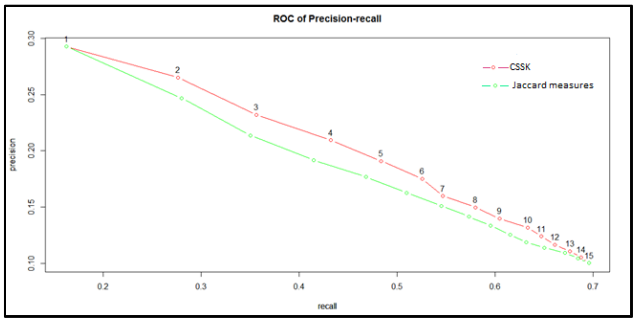

Figure 12: The chart reflects the ratio of Precision - Recall on two models

6. Conclusion

In this paper, we built UBCF model by suggesting a new similarity kernel based on Chi-Square kernel in order to determine the similarity of two users. Like other UBCF models, our model follows the main steps such as processing data, building the rating matrix, computing the similarity between two users, identify the item list that the similar users rated highly in order to showing the recommendation results and evaluate accuracy of the model. However, the new point of this model is to identify the list of similar users, instead of using the familiar measures such as Pearson correlation, Cosine similarity, Jaccard to determine the similar between two users. We use the CSSK. Through experiments, we found that our model results were relatively accurate on real number rating dataset and binary rating dataset. For the MovieLense dataset, the error parameters (RMSE, MSE, MAE) have a lower value than the model using the similarity Pearson measure. For the MSWeb dataset, the accuracy indicators as Precision, Recall, and F-measure have an outperformed value compared to the model using similarity Jaccard measures. Experimental results show that the UBCF model according to CSSK is capable to practice.

References

[1] Alan Said, Domonkos Tikk and Andreas Hotho. (2012) "The Challenge of Recommender Systems Challenges (tutorial)", *ACM RecSys'12 - Proceedings of the sixth ACM conference on Recommender systems*, pp.1-2.

[2] Alexandros Karatzoglou, Alex Smola and Kurt Hornik, (2016) "Kernel-Based Machine Learning Lab", *CRAN, Date/Publication 2016-03-29 14:14:49*.

[3] Ali Elkahky, Yang Song and Xiaodong He, (2015) "A Multi-View Deep Learning Approach for Cross Domain User Modeling in Recommendation Systems", *International World Wide Web Conference Committee (IW3C2), WWW 2015, May 18–22, 2015, Florence, Italy, ACM 978-1-4503-3469-3/15/05*.

[4] Andrea Vedaldi and Andrew Zisserman, (2011) "Efficient Additive Kernels via Explicit Feature Maps", *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, VOL. XX, NO. XX, JUNE 2011*, pp.1-14.

[5] Ben Schafer, Joseph Konstan, and John Ried, (1999) "Recommender Systems in E-Commerce", *EC '99 Proceedings of the 1st ACM conference on Electronic commerce, ISBN:1-58113-176-3*, pp.158-166.

[6] Bobadilla, Ortega, Hernando, and Gutiérrez, (2013) "Recommender systems survey", *Knowledge-Based Systems 46(2013)*, pp.109–132.

[7] F. Liu and H. J. Lee, (2010) "Use of social network information to enhance collaborative filtering performance", *Expert Systems with Applications 37(7)*, pp.4772-4778.

[8] F. Maxwell Harper, and Joseph A. Konstan, (2015) "The MovieLens Datasets: History and Context". *ACM Transactions on Interactive Intelligent Systems (TiiS) 5, 4, Article 19*, pp.1-19.

[9] Feng Zhang, TiGong, Victor E. Lee, and Gansen Zhao, (2016) "Chunming Rong and Guangzhi Qu, Fast algorithms to evaluate collaborative filtering recommender systems", *Knowledge-Based Systems 96 (2016),* pp.96–103.

[10] Ferdaous Hdioud, Bouchra Frikh, and Brahim Ouhbi, (2013) "Multi-Criteria Recommender Systems based on MultiAttribute Decision Making", *IIWAS '13 Proceedings of International Conference on Information Integration and Web-based Applications & Services*, pp. 203-208.

[11] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani, (2013) "An Introduction to Statistical Learning with Applications in R", *Springer New York Heidelberg Dordrecht London, ISBN 978-1-4614-7138-7 (eBook).*

[12] Gediminas Adomavicius, Bamshad Mobasher, Francesco Ricci, and Alex Tuzhilin, (2011) "Context-Aware Recommender Systems", *Association for the Advancement of Artificial Intelligence, ISSN 0738-4602*, pp.67-80.

[13] Gunawardana A and Shani G, (2009) "A Survey of Accuracy Evaluation Metrics of Recommendation Tasks", *Journal of Machine Learning Research, 10*, pp.2935–2962.

[14] Hao Wang, Naiyan Wang, and Dit-Yan Yeung, (2015) "Collaborative Deep Learning for Recommender Systems", *KDD'15, August 10-13, 2015, Sydney, NSW, Australia, 2015 ACM, ISBN 978-1-4503-3664-2*, pp.1235-1244.

[15] Herlocker JL, Konstan JA, Terveen LG, and Riedl JT, (2004) "Evaluating collaborative filtering recommender systems", *ACM Transactions on Information Systems, 22(1), ISSN 1046-8188*, pp.5–53.

[16] Hiep Xuan Huynh, Fabrice Guillet, and Henri Briand, (2005) "ARQAT: An Exploratory Analysis Tool For Interestingness Measures", pp.334-344.

[17] Huizhi Liang and Timothy Baldwin, (2015) "A Probabilistic Rating Auto-encoder for Personalized Recommender Systems", *CIKM'15, October 19–23, 2015, Melbourne, Australia, 2015 ACM, ISBN 978-1-4503-3794-6*, pp.1863-1866.

[18] Jack S. Breese, David Heckerman, and Carl M. Kadie, (1998) "Anonymous web data from www. microsoft. com",

Microsoft Research, Redmond WA, 98052-6399, USA, https://kdd. ics.uci. edu/databases/ msweb/ msweb.html.

[19] Jiliang Tang, Suhang Wang, Xia Hu, Dawei Yin, Yingzhou Bi, Yi Chang, and Huan Liu, (2016) "Recommendation with Social Dimensions", *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16)*, pp.251-257.

[20] Lan Phuong Phan, Nghia Quoc Phan, Ky Minh Nguyen, Hung Huu Huynh, Hiep Xuan Huynh, and Fabrice Guillet, (2016) "Interestingnesslab: A Framework for Developing and Using Objective Interestingness Measures*", ICTA 2016 : International Conference on Advances in Information and Communication Technology*, accepted.

[21] Malone TW, Grant KR, Turbak FA, Brobst SA, and Cohen MD, (1987) "Intelligent information sharing systems", *Communications of the ACM, 30(5), ISSN 0001-0782*, pp.390–402.

[22] Maria Augusta S. N. Nunes and Rong Hu, (2012) "Personality-based Recommender Systems: An Overview (tutorial)", *ACM RecSys'12- Proceedings of the sixth ACM conference on Recommender systems*, pp.1-2.

[23] Maria-Florina Balcan and Avrim Blum, (2006) "On a Theory of Learning with Similarity Functions", *Appearing in Proceedings of the 23rd International Conference on Machine Learning, Pittsburgh, PA*.

[24] Martin P. Robillard, Walid Maalej, Robert J. Walker, and Thomas Zimmermann, (2014) "Recommendation Systems in Software Engineering", *Springer Heidelberg New York Dordrecht London, ISBN 978-3-642-45135-5.*

[25] Michael D. Ekstrand, John T. Riedl, and Joseph A. Konstan, (2010) "Collaborative Filtering Recommender Systems", *Foundations and Trends in Human–Computer Interaction Vol. 4, No. 2 (2010)*, pp.81–173.

[26] Michael Hahsler, (2011) "recommenderlab: A Framework for Developing and Testing Recommendation Algorithms", *the Intelligent Data Analysis Lab at SMU, http://lyle. smu.edu /IDA/ recommenderlab/*.

[27] Michael Hahsler, (2015) "Lab for Developing and Testing Recommender Algorithms", *Copyright (C) Michael Hahsler, http://R-Forge .R-project. org/ projects/recommenderlab/*.

[28] Mingjie Qian, Liangjie Hong, Yue Shi, and Suju Rajan, (2015) "Structured Sparse Regression for Recommender Systems", *CIKM'15,October 19–23, 2015, Melbourne, VIC, Australia, 2015 ACM, ISBN 978-1-4503-3794-6*, pp.1895-1898.

[29] Nathan Srebro, (2007) "How Good is a Kernel When Used as a Similarity Measure?", *Volume 4539 of the series Lecture Notes in Computer Science*, pp.323-335.

[30] P. Bedi, H. Kaur, and S. Marwaha, (2007) "Trust based recommender system for semantic web", *IJCAI'07 - Proceedings of the 2007 International Joint Conferences on Artificial Intelligence*, pp.2677-2682.

[31] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, (1994) "GroupLens: an open architecture for collaborative filtering of netnews", *in ACM CSCW '94*, pp. 175–186, ACM.

[32] Phan Quốc Nghĩa and Huỳnh Xuân Hiệp, (2015) "Hệ tư vấn dựa trên khuynh hướng biến thiên hàm ý thống kê", *Hội thảo quốc gia lần thứ XVIII: Một số vấn đề chọn lọc của Công nghệ thông tin và truyền thông, Kỷ yếu hội thảo, Nhà xuất bản Khoa học và Kỹ thuật*, pp.93-99.

[33] Phan Quốc Nghĩa, Nguyễn Minh Kỳ, Nguyễn Tấn Hoàng, and Huỳnh Xuân Hiệp, (2015) "Hệ tư vấn dựa trên tiếp cận hàm ý thống kê", *Kỷ yếu hội nghị khoa học công nghệ quốc gia lần thứ VIII, Nhà xuất bản khoa học tự nhiên và công nghệ, ISBN: 978-604-913-397-8*, pp.297-408.

[34] Prem Meville and Vikas Sindhwani, (2010) "Recommender Systems", *Encyclopedia of Machine Learning, Springer-Verlag*, pp. 829-838.

[35] Quanjun Chen, Xuan Song, Harutoshi Yamada, and Ryosuke Shibasaki, (2016) "Learning Deep Representation from Big and Heterogeneous Data for Traffic Accident Inference", *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16)*, pp.338-344.

[36] Thomas Hofmann, Bernhard SchÖlkopf, and Alexander J. Smola, (2008) "KERNEL METHODS IN MACHINE LEARNING", *The Annals of Statistics 2008, Vol. 36, No. 3*, pp.1171–1220.

[37] U. Shardanand and P. Maes, (1995) "Social information filtering: Algorithms for automating "word of mouth"", *in ACM CHI '95, pp. 210–217, ACM Press/Addison-Wesley Publishing Co*.

[38] W. Hill, L. Stead, M. Rosenstein, and G. Furnas, (1995) "Recommending and evaluating choices in a virtual community of use", *inACM CHI '95, pp. 194–201, ACM Press/Addison-Wesley Publishing Co*.

[39] Xiaoyuan Su and Taghi M. Khoshgoftaar, (2009) "A Survey of Collaborative Filtering Techniques", *Hindawi Publishing Corporation, Advances in Artificial Intelligence, Volume 2009, Article ID 421425*, pp.1-9.

[40] Xiwang Yang, Harald Steck, Yang Guo, and Yong Liu, (2012) "On top-k recommendation using social networks", *ACM RecSys'12 - Proceedings of the sixth ACM conference on Recommender systems*, pp.67-74.