DE GRUYTER OPEN

STUDIES IN LOGIC, GRAMMAR AND RHETORIC 37 (50) 2014

DOI: 10.2478/slgr-2014-0021

Clustering Algorithms in Hybrid Recommender System on MovieLens Data

Urszula Kużelewska[1]

[1] Department of Information Systems and Computer Networks, Faculty of Computer Science, Bialystok University of Technology, Poland, u.kuzelewska@pb.edu.pl

Abstract. Decisions are taken by humans very often during professional as well as leisure activities. It is particularly evident during surfing the Internet: selecting web sites to explore, choosing needed information in search engine results or deciding which product to buy in an on-line store. Recommender systems are electronic applications, the aim of which is to support humans in this decision making process. They are widely used in many applications: adaptive WWW servers, e-learning, music and video preferences, internet stores etc. In on-line solutions, such as e-shops or libraries, the aim of recommendations is to show customers the products which they are probably interested in. As input data the following are taken: shopping basket archives, ratings of the products or servers log files.

The article presents a solution of recommender system which helps users to select an interesting product. The system analyses data from other customers' ratings of the products. It uses clustering methods to find similarities among the users and proposed techniques to identify users' profiles. The system was implemented in Apache Mahout environment and tested on a movie database. Selected similarity measures are based on: Euclidean distance, cosine as well as correlation coefficient and loglikehood function.

Keywords: recommender system, support of decision making, clustering

1. Introduction

A recommender system predicts user's interests (items or their ratings) based on his/her past behaviour. The past behaviour data is an input for personalisation mechanism and it is also a result of human decision making process. Users interact with a recommender system by taking different decisions: e.g. selecting a product from a recommendation list, specifying interesting values of features, inputting search query. As a result the visitors see different, adapted to them, recommendation lists depending on their tastes (Jannach, Zanker, Felfernig, & Friedrich, 2010). In all these scenarios, users have to solve a decision task. Their decisions are beneficial to them:

ISBN 978–83–7431–413–8 ISSN 0860-150X

people find the product or information which they looked for. In case of internet shopping the personalisation increases satisfaction of a customer as well as profit to a seller. It avoids the so-called long tail phenomenon, in which about 80% of customers buy only 20% of the assortment.

Recommender systems (RS) began to emerge as a research domain at the end of XX century (Mahmood & Ricci, 2009), (Anand & Mobasher, 2003) and (Kantor, Ricci, Rokach, & Shapira, 2011). The recent years show tremendously increasing rate of business as well as researchers interest. There are web sites where RS are an essential part, e.g. Amazon, YouTube, Netflix, Tripadvisor, LastFM. There are dedicated conferences and workshops related to the field: ACM Recommender Systems (RecSys), ACM Special Interest Group on Information Retrieval (SIGIR), User Modeling, Adaptation and Personalization (UMAP), and ACMs Special Interest Group on Management Of Data (SIGMOD) (Kantor et al., 2011). There are also special RS issues in academic journals: AI Communications, IEEE Intelligent Systems, International Journal of Electronic Commerce, International Journal of Computer Science and Applications, and others (Kantor et al., 2011). Since 2011 a special workshop – Workshop on Human Decision Making in Recommender Systems (Decisions@RecSys) – has been organised focusing on the aspect of integrating different theories of human decision making into the construction of recommender systems (Cremonesi, Donatacci, Garzotto, & Turrin, 2012).

Recommender Systems are used in many and various areas. The examples are internet news servers: e.g. Google News (http://news.google.com/) as well as a scalable approach, which combines content analysis and collaborative filtering (described in (Li, Wang, Zhu, & Li, 2011)). Tourism is another example of a domain where personalisation is desirable. An instance is a system named Traveller (Schiaffino, 2009) which helps users to plan their journeys. In (Malak, Yan, & Jie, 2011) an interesting application Pe-Gov of a recommender system in government e-services is described.

This paper presents an approach to recommender systems based on clustering methods. The clustering part identifies similar users, who then are taken to create clusters profiles. The profiles represent the most common users' preferences in one cluster. An active user (a user to whom recommendations are generated) can be compared to the profiles instead of all data, which considerably reduces computation time. Evaluation of the system is measured as its effectiveness in human decision support process.

The rest of the paper is organised as follows: Section 2 describes general aspects in recommendation systems, as well as approaches and problems in

this domain. Section 3 presents implementations of clustering methods in the field of recommendations. The following section, Section 4 describes the proposed approach, whereas Section 5 contains results of performed experiments. The last section concludes the paper.

2. Recommender Systems – Classification and Problems

Considering the type of input data as well as used methods, recommendation systems are divided into content-based, collaborative filtering (CF) and knowledge-based (Jannach et al., 2010) (see Table 1).

Content-based recommendations (also called content-based filtering) base on attribute (characteristic) vectors of items created from text. The text is connected with the items, e.g. it is their description. In case of books, the item characteristics include its genre, topic or author.

Knowledge-based approach is better for one-time user stores, e.g. selling cameras (people do not buy cameras often). The approach bases on technical attributes of the items and user preferences. The attributes are often weighted.

Collaborative filtering techniques search similarities among users or items; however only archives of registered users behaviour are analysed. As an example, similar users have mostly the same products in their baskets and similar items are bought by the same customers. They can be classified into model-based and memory-based methods. The first approach builds a model on the ratings, which is then used in generating recommendations. The other approach calculates recommendations by searching similar users or items in the whole archived data.

Recommender systems face many challenges and problems. They particularly concern the most effective and precise approach – collaborative filtering. CF systems base on past behaviour of users, which requires gathering some information about the visitors' preferences. The stored information is called a user model or user profile (Jannach et al., 2010). The user profile can be created by explicit information from a user or implicitly, e.g. recording the observed pages, watched videos, listened music or analysis of customer's basket. In case of a new visitor, without any recorded profile, an issue called cold-start problem appears.

Another problem occurs when a new item is added to the offer. In case of CF methods, it has not been assigned yet to any user and cannot be recommended to anyone. Content-based approach solves this issue by calculating similarity between the new and already stored items.

Table 1

Division, methods and problems in recommender systems

Method	Used algorithms and ideas	Problems
Content-based	Description of items. Automatic methods of description extraction. Clustering algorithms.	Cold-start problem. Poor scalability.
Collaborative filtering	Other users' past behaviour. Neighborhood of items. Clustering algorithms.	Sparsity of archive matrix. Cold-start problem. Poor scalability. Large amount of data. New item problem.
Knowledge-based	Features of items. Explicit user preferences.	Selection of features. Acquisition of user preference. Ranking selected items.

In arbitrary recommender system application, a number of offered items is large, whereas a user during one session visits a few to tens of them. It results in sparsity of input data and lower reliability in terms of measuring the similarity between customers (Sarwar, Karypis, Konstan, & Riedl, 2001).

Finally, however, vitally important challenge in the field of on-line recommendations is scalability. RS deal with large amount of dynamic data, however the time of results generation should be reasonable to apply them in real-time applications. A user reading news expects to see next offer for him/her in seconds, whereas millions of archived news have to be analysed.

There are also hybrid approaches, which combine at least two different methods. Problems in each of them are solved by strengths of the other one (see Table 1).

3. Clustering Methods in Recommender Systems

Clustering is a domain of data mining which had been applied in a wide range of problems, among others, in pattern recognition, image processing, statistical data analysis and knowledge discovery (Kużelewska, 2013). The aim of cluster analysis is organising a collection of patterns (usually represented as a vector of measurements, or a point in a multi-dimensional space) into clusters based on their similarity (Jain, Murty, & Flynn, 1999).

The points within one cluster are more similar to one another than to any other points from the remaining clusters.

Clustering has been the subject of research in the area of recommender systems, although it has not been widely studied yet (Pitsilis, 2011). The most often used method in memory-based collaborative filtering to identify neighbours is kNN algorithm, which requires calculating distances between an active user and all the registered ones. In contrast, clustering (in model-based collaborative filtering) reduces computation time, due to introduction of clusters models.

There are two approaches, which apply clustering in RS domain: *Cluster-based* and *Cluster-only* (Rongfei, Maozhong, & Chao, 2010). In both computation efficiency of systems increases as the clustering phase is performed off-line. The first approach is the most common one and focuses only on time efficiency improvement, which is application of clustering to find neighbourhood of active users. Further recommendation generation for them is performed by memory-based CF methods on part of input data forming identified the most similar cluster. Final precision of recommendations can be lower in comparison with memory-based collaborative filtering.

The second approach uses clustering as a main module of a recommender system. The partitioning applied on input data builds its model and further calculations are performed only on this model. Final precision of recommendations can be also lower in comparison with memory-based CF methods.

One of the first *Cluster-based* approaches, where clustering was used to partition users' preferences in order to increase neighbour searching efficiency is described in (Sarwar, Karypis, Konstan, & Riedl, 2002). One of the recent examples is (Kim, 2005), where k-means clustering with genetic algorithms is used for this purpose as well as in (Moghaddam & Selamat, 2011), where initial clustering (DBSCAN) was applied on demographic attributes.

Hierarchical clustering was also used in recommender systems (Haruechaiyasak, Tipnoe, Kongyoung, Damrongrat, & Angkawattanawit, 2005). Input data was clustered using hierarchical agglomerative approach, then new items were joined to the most similar cluster in the dendrogram.

Due to its time efficiency, clustering is often applied in mobile phone RS. An example is recommendation system for tourists (Gavalas, 2011) where clusters are built on users sharing similar interests. Data are taken from registering forms and partitioned using k-means algorithm.

Interesting paper is (Bridge & Kelleher, 2002) which used k-means algorithm, however in a modified version. The method, RecTree, was used in order to reduce computation time of input data. The authors presented

two approaches: clustering users (cardinality reduction) as well as items (attributes extraction). The groups of elements were then replaced with cluster centroids. Experiments were performed on MovieLens data.

The methods mentioned above confirm high interest of researchers in combination clustering with recommendations. However, they mainly concentrate on traditional techniques, such as k-means or hierarchical. The problem has been realised and some articles concerning new clustering methods as well as the other approach – *Cluster-only* – appeared.

An example is (Agarwal, Haque, Liu & Parsons, 2005), in which the clustering method, SCuBA, is proposed. The approach is based on subspace clustering which addresses the following problems: sparsity and high-dimensionality of input data and real-time computation. The algorithm was used to recommend scientific articles basing on registered publications visit logs of users.

A new solution with a new similarity measure for clustering in RS area is proposed in (Rongfei et al., 2010). The measure is calculated on the basis of neighbour vector of data. The solution was tested on k-means and DBSCAN clustering algorithms in both: *Cluster-only* and *Cluster-based* approaches.

4. An Algorithm Used in the Experiments

The recommendation algorithm proposed in this article belongs to *Cluster-only* approach. It is composed of two phases: building a data model – off-line creation of users' profiles – and on-line generation of recommended items. Clustering methods are used in the first part and further recommendations are provided based only on the built model. The clustering is based on k-means method, however the step of centroid calculation was modified. In this article there are two versions of modification proposed.

The most popular technique was used as a clustering method: k-means. The best advantage of k-means is its low time complexity and quite good quality of results. The algorithm partitions data into clusters which are given numbers in advance. The clusters are represented by their centroids – vectors of average values for every attribute within one cluster. Finally, hyperspherical groups of close sizes are formed (Jain et al., 1999).

The general recommendations and testing procedure are shown in Algorithm 1. The input set contains data of n users, who rated a subset of items – $A = \{a_1, \ldots, a_k\}$. The set of possible ratings – V – contains values $v_1, \ldots, v_c$. The algorithm returns a set of cluster representatives C_r composed of nc vectors of size k and a $RMSE$ evaluation value. The evaluation

is performed on 100% size of test users data for 80% of every user's ratings. It means, that for each test user their 20% ratings is removed and then compared to generated recommendations. It is also possible to generate a list of recommendations R_{x_a} for an active user.

Algorithm 1: A general algorithm of a recommender system used in the experiments

Data:

- $U = (X, A, V)$ – matrix of learning data, where $X = \{x_1, \ldots, x_n\}$ is a set of users, $A = \{a_1, \ldots, a_k\}$ is a set of items and $V = \{v_1, \ldots, v_c\}$ is a set of ratings values,
- $\delta : v \in V$ – a similarity function,
- $nc \in [2, n]$ – a number of clusters

Result:

- $C = \{C_1, \ldots, C_{nc}\}$ – a set of clusters,
- $C_r = \{c_{r,1}, \ldots, c_{r,nc}\}$ – the set of representatives of the clusters,
- R_{x_a} – a list of recommended items for an active user,
- $RMSE$ – a root mean squared error of the system

begin

$C \longleftarrow$ `clusterData`(U, V, δ, nc);
$C_r \longleftarrow$ `calculateRepresentatives`(U, V, δ, C);
$R_{x_a} \longleftarrow$ `recommend`(x_a, C_r, δ);

In the experiments two modified versions (described by Equations (1) and (2)) of k-means method were used. The modification concerned the procedure of centroids calculation. Instead of average values of the cluster components, the values based on frequency were taken.

$$V_i(C_j) = \begin{cases} v_l, & \forall v_l \in V \quad v_l \overset{\max|x(a_i)|}{=} x(a_i); \\ 0, & \text{in all other cases.} \end{cases} \tag{1}$$

In the first approach (1) the most frequent ratings for every item were taken as the representatives of clusters. When the number of the most frequent ratings was greater than 1, the final representative value was equal to 0.

In the other approach (2) the frequency of p highest or lowest values were examined. If the number of p highest ratings is equal or greater than $\alpha \cdot n$, the final rating is an average of these p highest ratings. If predominant values are p lowest ones, finally it is taken the average of them. In the

following experiments the value of α was equal to 0.6. In all other cases the definitive representative k-th value is an average of all ratings for k-th item.

$$V_i(C_j) = \begin{cases} \overline{x(a_i)}, & \forall i \in \{c-p \ldots c\} \ |x(a_i)| \geq \alpha \cdot n \ ; \\ \overline{x(a_i)}, & \forall i \in \{1 \ldots p\} \ |x(a_i)| \geq \alpha \cdot n \ ; \\ 0, & \text{in all other cases.} \end{cases} \tag{2}$$

In other words, if the set of ratings for an item contains a highly predominant extreme value, it is taken as a representative for the item. This operation increases contrast in user's ratings and decreases influence of undecided users or noise.

Illustrative examples of cluster representative calculation in both approaches are presented respectively in Tables 2 and 3. The tables contain a matrix of users ratings. Possible values of the ratings are 1, 2, 3, 4, 5. The users are divided into 3 clusters: $C_1 \in \{x_1, x_2, x_3\}$, $C_2 \in \{x_4, x_5, x_6\}$ and $C_3 \in \{x_7, x_8, x_9, x_{10}\}$. The calculated cluster centers are different in every case.

The proposed method is effective with regard to time during the on-line recommendation phase. As the preferences of an active user are compared only to representatives and the estimated rating is calculated only on representative values, the final on-line complexity of the method depends only on the number of clusters nc and the number of items k ($O(k \cdot nc)$). Complexity of the off-line phase is relatively high, however it does not affect the recommendation time. In the off-line step the complexity is higher in comparison with k-means method ($O(n \cdot k)$, as there is the additional step – calculation of representatives which complexity is $O(n)$.

Table 2

Calculation of a representative vector of clusters in the approach described by Equation (1)

	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}
x_1	5		4				3	4		
x_2	4		4				5	5		
x_3	5		5				5	4		
x_4	3		2		4			5		
x_5	1		2		4			4		
x_6	4		1		4			4		
x_7		1		4		5	4		3	3
x_8		2		4		5	4		2	4
x_9		1		5		5	5		5	5
x_{10}		4		5		5	4		4	4
Cluster representatives										
C_1:	5	0	4	0	0	0	5	4	0	0
C_2:	0	0	2	0	4	0	0	4	0	0
C_3:	0	1	0	0	0	5	4	0	0	4

Table 3
Calculation of a representative vector of clusters in the approach described by Equation (2)

	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}
x_1	5		4				3	4		
x_2	4		4				5	5		
x_3	5		5				5	4		
x_4	3		2		4			5		
x_5	1		2		4			4		
x_6	4		1		4			4		
x_7		1		4		5	4		3	3
x_8		2		4		5	4		2	4
x_9		1		5		5	5		5	5
x_{10}		4		5		5	4		4	4
Cluster representatives										
C_1:	4.67	0		4.33	0	0	0	5	4.33	0	0
C_2:	0	0		1.67	0	4	0	0	4.33	0	0
C_3:	0	1.33		0	4.5	0	5	4.25	0	4.5	4.33

5. Experiments

According to the above description, the experiments involved model-based recommender systems in *Cluster-only* approach. The models are created by two modified versions of the clustering algorithm k-means. Additionally, the results are compared with two other approaches: a *Cluster-only* method based on traditional k-means and commonly used approach: a memory-based collaborative filtering method, in which neighbourhood is determined by k nearest neighbours. The first comparison allows to evaluate if the proposed modifications made the k-means algorithm useful in *Cluster-only* approach. The second comparison evaluates quality of recommendation generated by the proposed technique. The most precise approach is memory-based collaborative filtering method, thereby it is usually used in articles as a reference procedure.

The clustering algorithm as well as the recommendation system were created using Apache Mahout library (http://mahout.apache.org/). The methods were tested with various similarity measures implemented in Apache Mahout: Euclidean based, cosine coefficient, correlation measure and loglikehood similarity.

Recommendations were executed on benchmark MovieLens data (*Movie Lens 100k Data Set*, n.d.), which contained 943 users and 1682 items. Results were compared according to the representatives calculation approach (see tables 4, 5, 6), the similarity measure and a number of clusters. The first

column of the tables contains the set number of clusters in case of clustering based approaches, or size of neigbourhood in case of kNN collaborative filtering procedure.

Since the aim of recommender systems is to support users in making their decisions, evaluation of them is related to measuring user's satisfaction. This was performed comparing calculated by the system preferences with users' real ones.

In the experiments, the results are assessed using the evaluator class from Mahout, which calculates $RMSE$ error between the calculated and real preferences. The evaluation was performed on 20% of items and all users from the training set. The procedure of evaluation is presented in Algorithm 2.

Algorithm 2: The procedure of evaluation of the recommender system used in the experiments

Data:

- $U = (X, A, V)$ – recommendations data, where $X = \{x_1, \ldots, x_n\}$ is a set of users, $A = \{a_1, \ldots, a_k\}$ is a set of items and $V = \{v_1, \ldots, v_c\}$ is a set of rating values,
- $\delta : v \in V$ – a similarity function,
- $C_r = \{c_{r,1}, \ldots, c_{r,nc}\}$ – the set of representatives of the clusters
- $nTest$ – a number of training data to use for validation
- $nItems$ – a number of items to use for validation

Result:

- $RMSE$ – a root mean squared error of the system

begin

$X_{test} \longleftarrow$ `randomTestData`$(X, nTest)$;
for $x_i \in X_{test}$ **do**
  $RMSE \longleftarrow 0$;
  $items \longleftarrow$ `randomItems`(x_i, A);
  for $item_j \in nItems$ **do**
    $rating_{est} \longleftarrow$ `estimatePref`(i, j, X, C_r, δ);
    $RMSE \longleftarrow (x_i(item) - rating_{est})^2$;
  $RMSE \longleftarrow \sqrt{RMSE}$;
  $RMSE \longleftarrow RMSE/nTest$;

After the representatives set of the clusters is formed, the number $nTests$ of training data undergoes evaluation. The testing data are selected

randomly ($randomTestData$). An evaluation procedure is the same for every data object. The number $nItems$ of a user ratings is removed from user row of data. The ratings removed are selected randomly ($randomItems$), as well. Then for each of the cleared values, their estimates ($rating_{est}$) are calculated based on the representatives. The procedure $estimatePref$ is presented in Algorithm 3. Finally, the error is composed of differences between the cleared and estimated ratings.

Algorithm 3: The procedure of rating estimation in the recommender system used in the experiments

Data:

- $U = (X, A, V)$ – recommendations data, where $X = \{x_1, \ldots, x_n\}$ is a set of users, $A = \{a_1, \ldots, a_k\}$ is a set of items and $V = \{v_1, \ldots, v_c\}$ is a set of rating values,
- $\delta : v \in V$ – a similarity function,
- $C_r = \{c_{r,1}, \ldots, c_{r,nc}\}$ – the set of representatives of the clusters
- i – an index of an active user
- j – an index of the estimated item

Result:

- $rating_{est}$ – an estimated rating

begin

$C_{best} \longleftarrow$ `findTheMostSimilarCluster`(i, X, C_r, δ);
$rating_{est} \longleftarrow C_{best}(j)$;

As it can be seen in Tables, the solution based on unmodified k-means algorithm is not very good. Regardless of similarity coefficient used, every $RMSE$ value is very high. It means, that original k-means method cannot be used directly in *Cluster-only* approach.

However, the described modifications made in centroid calculation improved the results significantly. Particularly it concerns the second (see Equation 2) of the presented solutions. The best results were generated in case of combination of this solution and the similarity measures: correlation based and cosine. The $RMSE$ calculated for these cases is comparable to memory-based collaborative filtering algorithm (see Table 7). The most optimal numbers of groups are $2, \ldots, 10$.

Table 4

Evaluation (*RMSE*) of recommendation system based on k-means clustering algorithm

Number of clusters	Euclidean	Cosine	Correlation	Log likehood
2	2.61	3.07	3.15	3.48
5	2.50	3.01	3.17	3.58
10	3.09	3.14	3.19	3.57
20	3.40	3.13	3.32	3.61
50	3.60	3.66	3.58	3.59

Table 5

Evaluation (*RMSE*) of recommendation system based on k-means clustering algorithm and the most frequent representative values of clusters

Number of clusters	Euclidean	Cosine	Correlation	Log likehood
2	1.19	1.19	1.21	1.29
3	1.21	1.22	1.21	1.37
5	1.18	1.22	1.22	1.36
7	1.19	1.23	1.21	1.41
10	1.19	1.24	1.22	1.38
15	1.20	1.25	1.24	1.39
20	1.21	1.25	1.25	1.44
30	1.21	1.27	1.28	1.46
50	1.25	1.26	1.36	1.70

Table 6

Evaluation (*RMSE*) of recommendation system based on k-means clustering algorithm and the average ratings approach to representative values of clusters

Number of clusters	Euclidean	Cosine	Correlation	Log likehood
2	1.08	1.17	1.07	1.17
3	1.09	1.18	1.08	1.26
5	1.05	1.17	1.07	1.29
7	1.06	1.17	1.08	1.29
10	1.08	1.18	1.09	1.32
15	1.08	1.21	1.09	1.34
20	1.10	1.22	1.10	1.35
30	1.13	1.25	1.24	1.36
50	1.16	1.26	1.31	1.43

Table 7

Evaluation (*RMSE*) of memory-based collaborative filtering recommendation system

Number of clusters	Euclidean	Cosine	Correlation	Log likehood
2	1.21	1.24	1.37	1.23
3	1.21	1.21	1.35	1.20
5	1.18	1.18	1.31	1.17
7	1.18	1.15	1.30	1.18
10	1.17	1.14	1.29	1.11
15	1.17	1.14	1.26	1.11
30	1.09	1.09	1.25	1.08
50	1.05	1.08	1.22	1.08

6. Conclusion

Recommendations support humans in decision making process, thereby increasing users' satisfaction. Typical problem faced by internet users is overload of information, which decreases the quality of their choices. Decisions supported with recommender systems are more precise, due to more alternatives offered to users. In this article a clustering approach to recommendations is presented. Clustering algorithms are used in data mining tasks, where it is necessary to search similarities among data.

In the implemented *Cluster-only* recommender systems, partitioning methods were used in off-line phase of recommendations to identify similar users. The most similar objects are placed in one cluster which was described then by their representative values. The representatives of a cluster are calculated based on the frequency of ratings of users belonging to the cluster.

In on-line phase of recommendations, where it is necessary to create a list of recommended items, the current ratings of an active user are not compared to all archived data but only to their representatives. This approach is faster than *Cluster-based* methods, in which recommendations are generated based on an active user's similarity to objects from one cluster. Here, recommendations are generated based only on the selected representative, thereby the procedure of representatives calculation is very important.

This approach may be characterised by lower precision. However, in contrast to memory-based collaborative methods, it is faster, which is important in real-time applications.

In the article two methods of cluster representatives computation were presented and evaluated. Their effectiveness was compared to the centroid-based solution and memory-based collaborative filtering methods. The proposed procedure significantly increased accuracy of generated recommendations in comparison with centroid-based method. The results were characterised by lower values of $RMSE$, regardless of similarity coefficient.

The experiments presented in this article are the beginning of wider researches of application of clustering algorithms in RS domain. There are plans to develop the approach towards the following directions:

- to implement in Apache Mahout environment other clustering methods, such as DBSCAN or EM, and to perform experiments taking advantage of their ability to evaluate a number of clusters automatically,
- to apply a granulation and clustering method described in (Kużelewska, 2013), which allow to adjust resolution as well as accuracy of generated results,
- to experiment on other data, like binary (particularly on shopping baskets) to test efficiency of the proposed approach,
- to reduce items dimensionality by applying data mining attribute selection methods as well as techniques from text processing.

REFERENCES

Agarwal, N., Haque, E., Liu, H., & Parsons, L. (2005). Research paper recommender systems: A subspace clustering approach. In *Advances in Web-Age Information Management* (pp. 475–491). Springer.

Anand, S. S., & Mobasher, B. (2003). Intelligent techniques for web personalization. In *Proceedings of the 2003 International Conference on Intelligent Techniques for Web Personalization* (pp. 1–36). Springer-Verlag.

Bridge, D., & Kelleher, J. (2002). Experiments in sparsity reduction: Using clustering in collaborative recommenders. In *Artificial Intelligence and Cognitive Science* (pp. 144–149). Springer.

Cremonesi, P., Donatacci, A., Garzotto, F., & Turrin, R. (2012). Decision-Making in Recommender Systems: The Role of User's Goals and Bounded Resources. In *6th ACM Conference on Recommender Systems* (pp. 1–7).

Gavalas, K. M. D. (2011). A web-based pervasive recommendation system for mobile tourist guides. In *Personal and Ubiquitous Computing* (Vol. 15, pp. 759–770). Springer-Verlag.

Haruechaiyasak, C., Tipnoe, C., Kongyoung, S., Damrongrat, C., & Angkawattanawit, N. (2005). A dynamic framework for maintaining customer profiles in e-commerce recommender systems. In *IEEE International Conference on e-Technology, e-Commerce and e-Service*, (pp. 768–771).

Jain, A. K., Murty, M. N., & Flynn, P. J. (1999) Data clustering: a review. *ACM Computing Surveys*, *31*(3), 264–323.

Jannach, D., Zanker, M., Felfernig, A., & Friedrich, G. (2010). *Recommender systems: an introduction*, Cambridge University Press.

Kantor, P. B., Ricci, F., Rokach, L., & Shapira, B. (2011). *Recommender systems handbook*. Springer.

Kim, Y. S., T.H. (2005). An Effective Recommendation Algorithm for Clustering-Based Recommender Systems. In *Lecture Notes in Artificial Intelligence* (Vol. 3809, p. 1150–1153). Springer-Verlag.

Kużelewska, U. (2013). Advantages of Information Granulation in Clustering Algorithms. In *Agents and Artificial Intelligence* (pp. 131–145). Springer.

Li, L., Wang, D.-D., Zhu, S.-Z., & Li, T. (2011). Personalized news recommendation: a review and an experimental investigation. *Journal of Computer Science and Technology*, *26*(5), 754–766.

Mahmood, T., & Ricci, F. (2009). Improving recommender systems with adaptive conversational strategies. In *Proceedings of the 20th ACM Conference on Hypertext and Hypermedia* (pp. 73–82).

Malak, A.-H., Yan, L. H., & Jie, L. (2011). Personalized e-Government Services: Tourism Recommender System Framework. In *Lecture Notes in Business Information Processing*, (Vol. 75, p. 173–187). Springer.

Moghaddam, S. G., & Selamat, A. (2011). A scalable collaborative recommender algorithm based on user density-based clustering. In *3rd International Conference on Data Mining and Intelligent Information Technology Applications (ICMiA)* (pp. 246–249).

MovieLens 100k Data Set. (n.d) Retrieved 01.02.2013, from http://www.grouplens.org/node/73.

Pitsilis, G. e. a. (2011). Clustering Recommenders in Collaborative Filtering Using Explicit Trust Information. In *Trust Management V* (Vol. 358, p. 82–97). Springer.

Rongfei, J., Maozhong, J., & Chao, L. (2010). A new clustering method for collaborative filtering. In *International Conference on Networking and Information Technology* (pp. 488–492).

Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th International Conference on World Wide Web* (pp. 285–295).

Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2002). Recommender systems for large-scale e-commerce: Scalable neighborhood formation using clustering. In *Proceedings of the 5th International Conference on Computer and Information Technology* (Vol. 1).

Schiaffino, A. A., S. (2009). Building an expert travel agent as a software agent. In *Expert Systems with Applications* (Vol. 36, pp. 1291–1299). Elsevier.