Using Collaborative Filtering to Overcome the Curse of Dimensionality when Clustering Users in a Group Recommender System

Ludovico Boratto and Salvatore Carta
Dipartimento di Matematica e Informatica, Università di Cagliari, Via Ospedale 72, Cagliari, Italy

Keywords: Group Recommendation, Clustering, Curse of Dimensionality, Collaborative Filtering, Prediction Accuracy.

Abstract: A characteristic of most datasets is that the number of data points is much lower than the number of dimensions (e.g., the number of movies rated by a user is much lower than the number of movies in a dataset). Dealing with high-dimensional and sparse data leads to problems in the classification process, known as *curse of dimensionality*. Previous researches presented approaches that produce group recommendations by clustering users in contexts where groups are not available. In the literature it is widely-known that clustering is one of the classification forms affected by the curse of dimensionality. In this paper we propose an approach to remove sparsity from a dataset before clustering users in group recommendation. This is done by using a Collaborative Filtering approach that predicts the missing data points. In such a way, it is possible to overcome the curse of dimensionality and produce better clusterings. Experimental results show that, by removing sparsity, the accuracy of the group recommendations strongly increases with respect to a system that works on sparse data.

1 INTRODUCTION

A widely-known problem in the recommender systems literature is that the number of items available in a dataset is usually much higher than the number of items rated by a user (Amatriain et al., 2011). Referring to a real-world scenario like Amazon.com, a user considers a few tens of items, while the system contains millions of them.

The problem that arises when the number of dimensions in the data increases and data becomes sparse, is known as *curse of dimensionality* (Bellman, 1961). Curse of dimensionality prevents a proper classification of data, since the phenomena that occur on high-dimensional and sparse data do not allow to give statistical significance to the classification.

As (Radovanovic et al., 2010) highlights, clustering is one of the forms of classification affected by the curse of dimensionality.

Group recommendation (Boratto and Carta, 2011) is designed for contexts in which more than a person is involved in the recommendation process (Jameson and Smyth, 2007). Producing recommendations for a group is also useful in contexts in which the amount of recommendations that can be built is limited.

A company decides to print recommendation flyers that present suggested products. Even if the data to produce a flyer with individual recommendations for each customer is available, printing a different flyer for everyone would be technically too hard to accomplish and costs would be too high. A possible solution would be to set a number of different flyers to print, such that the printing process could be affordable in terms of costs and the recipients of the same flyer would be interested by its content.

With respect to classic group recommendation, these systems have to define groups, in order to respect the constraint on the number of recommendations that can be produced. In a context like this, overcoming the curse of dimensionality is crucial, since these approaches work in a recommendation domain and they deal with sparse data. Moreover, in order to detect groups, users are clustered, so the clustering process is affected by data sparsity.

In (Boratto and Carta, 2013), it was highlighted that the best way to predict ratings in a group recommender system that detects groups is to predict individual ratings for each user.

This paper deals with the clustering task of a group recommender system that detects groups, by presenting an approach to remove sparsity from data. This is done by using a Collaborative Filtering algorithm to predict the missing data points, in order to overcome the curse of dimensionality that occurs in the clustering task of the system. By improving the

DOI: 10.5220/0004865005640572
In *Proceedings of the 16th International Conference on Enterprise Information Systems* (ICEIS-2014), pages 564-572
ISBN: 978-989-758-028-4

accuracy of the task that detects groups, the overall quality of the system should improve, i.e., more accurate group recommendations should be produced.

The proposed approach, by predicting the ratings before clustering users, offers a simple but effective solution to the data sparsity problem in this context. In fact, the architecture of the group recommender system and the flow of the computation change but, at the same time, the proposed approach does not add any computational complexity to the group recommender system. So, the proposed solution is able to effectively solve the sparsity problem in the clustering task of a group recommender system, without negatively affecting its performances in any way.

The scientific contributions coming from this paper are the following:

- we present a novel approach to deal with the curse of dimensionality in a group recommendation context. No approach in the literature deals with sparse data in a group recommender system;
- we show how we remove sparsity from data, while keeping the same computational complexity;
- we analyze the impact of sparsity on groups of different sizes and show that if a system handles small groups, it is more affected by sparsity.

Experimental results, validated through statistical hypothesis tests, confirm that removing sparsity before clustering users allows to significantly improve the performances of a group recommender system that automatically detects groups.

The paper is structured as follows: Section 2 presents related work; Section 3 describes a group recommender system that works on sparse data; Section 4 proposes a solution to remove sparsity before clustering the users; Section 5 presents the experiments performed to evaluate our proposal; Section 6 contains conclusions and future work.

2 RELATED WORK

This section presents the existing group recommendation approaches and the works that deal with the curse of dimensionality problem when clustering users.

2.1 Group Recommendation

Here we present the most important and recent works developed in the group recommendation research area. Since our proposal uses individual predictions to avoid sparsity, this section is divided into approaches that build predictions for a group and approaches that build predictions for each user.

2.1.1 Approaches that Build Predictions for Each Group

MusicFX (McCarthy and Anagnost, 1998) is a system that recommends music to the members of a fitness center. Since people in the room change continuously, the system gives the users that are working out in the fitness center the possibility to login. The music to play is selected considering the preferences of each user in a summation formula.

In-Vehicle Multimedia Recommender (Zhiwen et al., 2005) is a system that aims at selecting multimedia items for a group of people traveling together. The system aggregates the profiles of the passengers and merges them, by using a notion of distance between the profiles. A content-based system is used to compare multimedia items and group preferences.

FIT (Family Interactive TV System) (Goren-Bar and Glinansky, 2004) is a TV program recommender system. The only input required by the system is a stereotype user representation (i.e., a class of viewers that would suit the user, like *women*, *businessmen*, *students*, etc.), along with the user preferred watching time. When someone starts watching TV, the system looks at the probability of each family member to watch TV in that time slot and predicts who there might be watching TV. Programs are recommended through an algorithm that combines such probabilities and the user preferences.

In (McCarthy et al., 2006) a group recommender system called *CATS (Collaborative Advisory Travel System)* is presented. Its aim is to help a group of friends plan and arrange ski holidays. To achieve the objective, users are positioned around a device called "DiamondTouch table-top" and the interactions between them (since they physically share the device) help the development of the recommendations.

2.1.2 Approaches that Build Predictions for Each User

PolyLens (O'Connor et al., 2001) is a system built to produce recommendations for groups of users who want to see a movie. A Collaborative Filtering approach is used to produce recommendations for each user of the group. The movies with the highest recommended ratings are considered and a "least misery" strategy is used, i.e., the recommended rating for a group is the lowest predicted rating for a movie, to ensure that every member is satisfied.

Pocket RestaurantFinder (McCarthy, 2002) is a system that suggests restaurants to groups of people who want to dine together. Each user fills a profile with preferences about restaurants, like the price range or the type of cuisine they like (or do not like).

Once the group composition is known, the system estimates individual preference for each restaurant and averages those values to build a group preference and produce a list of recommendations.

Travel Decision Forum (Jameson, 2004) is a system that helps groups of people plan a vacation. Since the system aims at finding an agreement between the members of a group, asynchronous communication is possible and, through a web interface, a member can view (and also copy) other members preferences. Recommendations are made by using the median of the individual preferences.

In (Chen and Pu, 2013), Chen and Pu present *CoFeel*, an interface that allows to express through colors the emotions given by a song chosen by the *GroupFun* music group recommender system. The interface allows users to give a feedback about how much they liked the song and the system considers the preferences expressed through the emotions, in order to generate a playlist for a group.

In (Jung, 2012), Jung develops an approach to identify long tail users, i.e., users who can be considered as experts on a certain attribute. So, the ratings given by the long tail user groups are used, in order to provide a relevant recommendation to the non-expert user groups, whch are called short head groups.

2.2 The Curse of Dimensionality in Clustering

In (Jing et al., 2007), authors highlight that two main types of approaches are embraced in order to overcome the curse of dimensionality in clustering.

The first category of approaches operates a *hard subspace clustering*, which aims at finding all the clusters in every subspace, therefore allowing a data point to be in more than a cluster in a different subspace. These approaches are based on the fact that in a d-dimensional space there are 2^d axis-parallel subspaces and that if a cluster appears in a lower-dimensional space, it also appears in a higher-dimensional one. This approach is followed by CLIQUE (Agrawal et al., 1998), MAFIA (Goil et al., 1999) and OptiGrid (Hinneburg and Keim, 1999).

The second type of approach, i.e., the *soft subspace clustering*, weights each dimension, in order to derive the contribution of a dimension in a clustering. By using this approach, the subspace is identified by the importance (expressed by the weight), that each dimension has in the clustering. This approach is embraced by (DeSarbo et al., 1984; Soete, 1988; Makarenkov and Legendre, 2001; Huang et al., 2005).

2.3 Neighbors Selection in Collaborative Filtering

Approaches like (Boumaza and Brun, 2012), use a global set of neighbors, to predict the ratings for all the users. Therefore the space occupied by the model is reduced. This approach is less affected by sparsity, since the same neighbors are used for each user.

2.4 Discussion

None of the works presented in the group recommendation area uses the predictions in order to avoid the sparsity problem. Moreover, in our approach we could not adopt an algorithm that builds predictions for each group, since the clustering algorithm builds groups using individual preferences.

The approaches that deal with the curse of dimensionality in clustering operate in a completely different way from our approach. In fact, our solution keeps the same number of dimensions of the original data, but tries to predict the missing values with a Collaborative Filtering approach, while subspace clustering reduces the number of dimensions (a detailed motivation of our choice is presented along with the algorithm in Section 4).

The approach that uses a global set of neighbors is more suitable to improve the scalability of a Collaborative Filtering algorithm, while our approach is focused on efficiently removing sparsity when clustering users. So, we decided to adopt a classic User-Based Collaborative Filtering approach, which selects different neighbors for each user.

3 GROUP RECOMMENDATION FOR AUTOMATICALLY DETECTED GROUPS

Recently, a group recommender system that automatically detects groups by clustering users was presented (Boratto and Carta, 2013).

The algorithm, that from here on will be named *Cluster&Predict*, detects groups of similar users, predicts individual preferences and aggregates these preferences into a group model. The tasks performed by the system are the following:

1. *Detection of the groups.* Considering the individual preferences expressed by each user, groups of similar users are detected with the k-means clustering algorithm.
2. *Predictions of the missing ratings for individual users.* Individual predictions are calculated for

Table 1: Example of *user model*.

	i_1	i_2	i_3	i_4	i_5	i_6	i_7	i_8
u	r_{u1}		r_{u3}				r_{u7}	r_{u8}

Table 2: Example of *rating matrix*.

	i_1	i_2	i_3	i_4	i_5	i_6	i_7	i_8
u_1	r_{11}		r_{13}	r_{14}			r_{17}	r_{18}
u_2	r_{21}	r_{22}		r_{24}	r_{25}		r_{27}	
u_3		r_{22}	r_{33}	r_{34}		r_{36}		
				...				
u_n	r_{n1}	r_{n2}	r_{n3}		r_{n5}	r_{n6}		r_{n8}

each user with a User-Based Collaborative Filtering Approach presented in (Schafer et al., 2007).

3. *Aggregation of the predictions (Group modeling).* Once groups have been detected, a group model is built by aggregating all the predictions of a group.

All the tasks are now be described in detail.

Detection of the Groups. The first task that the system performs is the partitioning of the set of users into a number of groups equal to the number of recommendations that can be produced. Since in this application scenario groups do not exist, unsupervised classification (*clustering*) is necessary.

In order to create groups with similar preferences, a *user model*, like the one in Table 1, is considered. A model contains, for each *item* i_n that a user u evaluated, a *rating* r_{un}, which expresses with a numerical value how much the user likes the item. All the user models like the one previously described take the form of a matrix, usually called *ratings matrix*, like the one in Table 2.

It was recently highlighted (Amatriain et al., 2011) that the k-means clustering algorithm is by far the most used clustering algorithm in recommender systems and alternatives are rarely used, mostly because of the simplicity and the efficiency that the algorithm can offer.

This task detects groups by clustering users with the k-means clustering algorithm, that takes as input a rating matrix, like the one presented in Table 2. So, users are grouped based on the ratings available in the individual user models (each user model is represented as a line in the rating matrix and contains her/his preferences).

The output is a partitioning of the users into groups (clusters), such that users with similar models (i.e., similar ratings for the same items) are in the same group and receive the same recommendations.

Predictions of the Missing Ratings for Individual Users. The missing ratings are predicted for each user in a group with a classic User-Based Nearest Neighbor Collaborative Filtering algorithm, presented in (Schafer et al., 2007). The algorithm predicts a rating p_{ui} for each item i that was not evaluated by a user u, considering the rating r_{ni} for the item i of each similar user n. A user n similar to u is called a *neighbor* of u[1]. Equation (1) gives the formula used to predict the ratings:

$$p_{ui} = \overline{r}_u + \frac{\sum_{n \subset neighbors(u)} userSim(u,n) \cdot (r_{ni} - \overline{r}_n)}{\sum_{n \subset neighbors(u)} userSim(u,n)} \qquad (1)$$

Values $\overline{r}_u$ and $\overline{r}_n$ represent, respectively, the mean of the ratings expressed by user u and user n. Similarity *userSim*() between two users is calculated using the Pearson's correlation, a coefficient that compares the ratings of all the items rated by both the target user and the neighbor. Pearson's correlation between a user u and a neighbor n is given in Equation (2) (I_{un} is the set of items rated by both u and n).

$$userSim(u,n) = \frac{\sum_{i \subset I_{un}} (r_{ui} - \overline{r}_u)(r_{ni} - \overline{r}_n)}{\sqrt{\sum_{i \subset I_{un}} (r_{ui} - \overline{r}_u)^2} \sqrt{\sum_{i \subset I_{un}} (r_{ni} - \overline{r}_n)^2}} \qquad (2)$$

The metric range is between 1.0 (complete similarity) and -1.0 (complete dissimilarity). Negative values do not increase the prediction accuracy, so they are discarded by the task.

Aggregation of the Predictions (Group Modeling). In order to create a model that represents the preferences of a group, the *Additive Utilitarian* group modeling strategy (Masthoff, 2011) is adopted. The strategy sums the individual ratings for each item and produces a list of group ratings (the higher the sum is, the earlier the item appears in the list). The ranked list of items is exactly the same that would be produced when averaging the individual ratings, so this strategy is also called 'Average strategy'. An example of how the strategy works is given in Table 3.

In order to have the same scale of ratings both in the group models and in the individual user models, the produced group models contain the average of the individual predictions, instead of the sum.

[1]The neighbors of a user are selected considering the whole user set and not just the users that belong to her/his group. In fact, the ratings are predicted for each item, so the neighbors (i.e., the most similar users) who rated that specific item might be outside the cluster.

Table 3: Example of the *Additive Utilitarian* strategy.

	i_1	i_2	i_3	i_4	i_5	i_6	i_7	i_8
u_1	8	10	7	10	9	8	10	6
u_2	7	10	6	9	8	10	9	4
u_3	5	1	8	6	9	10	3	5
Group	20	21	21	25	26	28	22	15

4 IMPROVING ACCURACY BY OVERCOMING THE CURSE OF DIMENSIONALITY

The system presented in the previous section (*Cluster&Predict*) works on sparse data, because in (Boratto and Carta, 2013) three approaches to predict the ratings were evaluated and the rest of the tasks were implemented in the same way in all the systems. Therefore, the clustering task was not studied in that paper and, as previously described, groups are detected based on the preferences explicitly expressed by each user.

When designing the approach to overcome the curse of dimensionality, we avoided classic subspace clustering approaches for these main reasons:

- we wanted to overcome these phenomena without increasing the complexity of the system;
- since a user model is built by considering the preferences for a set of items (which represent the data dimensions), it would be hard to introduce a weight that represents the "importance" of an item for the whole dataset.

In order to remove sparsity from data, we propose an approach that builds individual predictions *before* clustering users. The system, named *Predict&Cluster*, includes all the predicted ratings in the user models and clusters users based both on the explicitly expressed preferences and on the predicted ratings. So the clustering task now works on a full matrix and the phenomena that occur due to sparsity are avoided.

In other words, the two systems (i.e., *Predict&Cluster* and *Cluster&Predict*) perform the same tasks, but the first two are switched, in order to predict the ratings before the clustering task.

The tasks performed by *Predict&Cluster* are the following:

1. *Predictions of the missing ratings for individual users.* Individual predictions are calculated for each user with a User-Based Collaborative Filtering Approach.
2. *Detection of the groups.* Considering both the individual preferences expressed by each user and the predicted ratings, groups of similar users are detected with the k-means clustering algorithm.
3. *Aggregation of the predictions (Group modeling).* Once groups have been detected, a group model is built by aggregating all the predictions of a group.

The algorithms performed by the tasks are the same ones performed by the *Cluster&Predict* system, in order to evaluate the effects that removing sparsity has on the accuracy of a group recommender system that automatically detects groups.

4.1 Discussion

This apparently simple approach causes important changes in the architecture of the group recommender systems. In fact, in the *Predict&Cluster* system, the input given to the clustering algorithm is the output produced by the rating prediction task. This allows to use a much larger amount of information in the clustering, with respect to the previous architecture.

Given the study conducted in this paper, the choice to select the neighbors on the whole user set is strengthened by the fact that if predictions were calculated considering a clustering built using a sparse rating matrix, the phenomena related to the curse of dimensionality would affect the quality of the clustering (i.e., the groups), which might not be significant.

The effect of sparsity is independent from the clustering algorithm used by the group recommender system. In fact, by using an algorithm different from k-means, that still works in a metric space, the same phenomena that occur due to sparsity would still occur (i.e., it would be hard to build significant similarities among users). If a different class of clustering algorithms was used, like the ones that cluster a graph, sparsity would still affect the outcome of the algorithm, since the links among two users would be built considering sparse data (for example, using a similarity/distance metric based on the ratings) and would still not be significant.

5 EXPERIMENTAL FRAMEWORK

This section presents the framework built for the experiments.

5.1 Experimental Setup

To conduct the experiments, we adopted MovieLens-1M, which is a dataset widely used in the literature.

The clusterings with k-means were created with a testbed program called KMlocal (Kanungo et al., 2002), that contained a variant of k-means, called *EZ Hybrid*. The k-means algorithm minimizes the *average distortion*, i.e., the mean squared distance from each point to its nearest center. *EZ Hybrid* is the algorithm that returned the lowest distortion with this dataset, so it is the one used to cluster the users.

The number of neighbors used to predict the ratings is 100; see (Boratto and Carta, 2013) for the details of the experiments that allowed to set the value.

An analysis has been performed, by comparing the RMSE values obtained by each system, considering different numbers of groups to detect. The choice to measure the performances for different numbers of groups has been made to show how the quality of the systems change as the constraint changes. In each experiment, four different clusterings with 20, 50, 200 and 500 groups were created. Moreover, we compared the results obtained with the four clusterings with the results obtained considering a single group with all the users (i.e., predictions are calculated considering the preferences of all the users), and the results obtained by the system that calculates predictions for each user.

RMSE was chosen as a metric to compare the algorithms because, as the organizers of the Netflix prize highlight[2], it is widely used, allows to evaluate a system through a single number, and emphasizes the presence of large errors.

In order to evaluate if the RMSE values returned by two experiments are significantly different, independent-samples two-tailed Student's t-tests have been conducted. In order to make the tests, a 5-fold cross-validation was preformed.

The results obtained by each system with this experimental setup are compared, in order to evaluate the effects of sparsity in this context.

5.2 Dataset and Data Preprocessing

The dataset used, i.e., MovieLens-1M[3], is composed of 1 million ratings, expressed by 6040 users for 3900 movies. This framework uses only the file `ratings.dat`, which contains the ratings given by the users. The file contains four features: *UserID*, that contains user IDs in a range between 1 and 6040, *MovieID*, that contains movie IDs in a range between 0 and 3952, *Rating*, that contains values in a scale between 1 and 5 and *Timestamp*, that contains a timestamp of when a user rated an item. The file was preprocessed, by mapping the feature *UserID* into a new set of IDs between 0 and 6039, to facilitate the computation using data structures. In order to conduct the cross-validation, the dataset was split into five subsets with a random sampling technique (each subset contains 20% of the ratings).

[2]http://www.netflixprize.com/faq

[3]http://www.grouplens.org/

5.3 Metrics

The quality of the predicted ratings was measured through the Root Mean Squared Error (RMSE). The metric compares each rating r_{ui}, expressed by a user u for an item i in the test set, with the rating p_{gi}, predicted for the item i for the group g in which user u is. The formula is shown below:

$$RMSE = \sqrt{\frac{\sum_{i=0}^{n}(r_{ui} - p_{gi})^2}{n}}$$

where n is the number of ratings in the test set.

In order to compare if two RMSE values returned by two experiments are significantly different, independent-samples two-tailed Student's t-tests have been conducted. These tests allow to reject the null hypothesis that two values are statistically the same. So, a two-tailed test will test if an RMSE value is significantly greater or significantly smaller than another RMSE value. Since each experiment was conducted five times, the means M_i and M_j of the RMSE values obtained by two systems i and j are used to compare the systems and calculate a value t:

$$t = \frac{M_i - M_j}{s_{M_i - M_j}}$$

where

$$s_{M_i - M_j} = \sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}$$

s^2 indicates the variance of the two samples, n_1 and n_2 indicate the number of values considered to build M_1 and M_2 (in our case both are equal to 5, since experiments were repeated five times). In order to determine the $t - value$ that indicates the result of the test, the degrees of freedom for the test have to be determined:

$$\text{d.f.} = \frac{(s_1^2/n_1 + s_2^2/n_2)^2}{(s_1^2/n_1)^2/(n_1 - 1) + (s_2^2/n_2)^2/(n_2 - 1)}$$

Given t and $d.f.$, the $t - value$ (i.e., the result of the test), can be obtained in a standard table of significance as

$$t(d.f.) = t - value$$

The $t - value$ derives the probability p that there is no difference between the two means. Along with the result of a t-test, the standard deviation SD of the mean is presented.

5.4 Experimental Results

Figure 1 and Table 4 report the performances of the two systems.

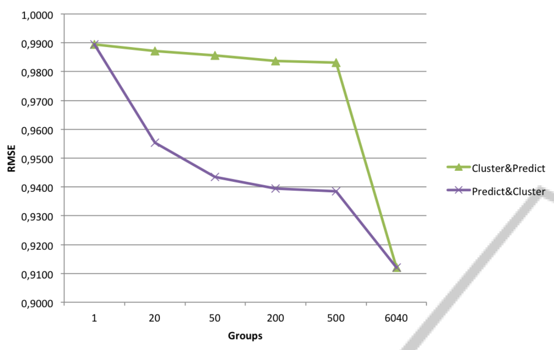

Figure 1: RMSE values of the two systems.

Table 4: RMSE values of the two systems.

	1 group	20 groups	50 groups	200 groups	500 groups	6040 groups
Cluster&Predict	0.9895	0.9872	0.9857	0.9837	0.9832	0.9120
Predict&Cluster	0.9895	0.9554	0.9435	0.9395	0.9385	0.9120

The first trivial aspect that can be noticed is that, when users are not clustered (i.e., the *1 group* and *6040 groups* settings), the results obtained by the systems are the same.

The interesting part to analyze is when users are clustered, in order to evaluate the effects of sparsity on the accuracy of the systems. A number of interesting aspects can be noticed when analyzing the results:

- by removing sparsity, performances strongly improve. In fact, *Predict&Cluster* always outperforms *Cluster&Predict*. This is the sign that sparsity strongly affects clustering also in a group recommendation context and that the performances strongly improve by removing sparsity;
- the improvement in the results obtained by *Predict&Cluster* gets better as the number of groups increases. This means that the more groups are created, the more sparsity affects the results of a system. Therefore, the removal of sparsity is even more significant for a high number of groups;
- as the number of groups grows, performances improve (i.e., RMSE values lower) in both the systems. This means that if a system can work with more groups, its performances improve with respect to a lower number of groups (no matter if it works with sparse data or not).

Student's t-tests allow to validate the results obtained by the two systems on each clustering.

With 20 groups, the test returned a complete statistical difference comparing the RMSE values for *Predict&Cluster* ($M = 0.9554$, $SD = 0.00$) and *Cluster&Predict* ($M = 0.9872$, $SD = 0.00$); $t(4) = -26.18$, $p = 0.0$.

For 50 groups, the test returned a complete statistical difference between the values obtained for *Predict&Cluster* ($M = 0.9435$, $SD = 0.00$) and *Cluster&Predict* ($M = 0.9857$, $SD = 0.00$); $t(4) = -55.27$, $p = 0.0$.

For 200 groups, the same happens when comparing the RMSE obtained for *Predict&Cluster* ($M = 0.9395$, $SD = 0.00$) and *Cluster&Predict* ($M = 0.9837$, $SD = 0.00$); $t(8) = -77.21$, $p = 0.0$.

Finally, even with 500 groups there is a significant difference in the RMSE values for *Predict&Cluster* ($M = 0.9385$, $SD = 0.00$) and *Cluster&Predict* ($M = 0.9832$, $SD = 0.00$); $t(7.68) = -72.62$, $p = 0.0$.

Results suggest that removing sparsity allows to significantly improve the accuracy of a system.

6 CONCLUSIONS AND FUTURE WORK

This paper presented an approach to remove sparsity from data, in order to overcome the curse of dimensionality and improve the clustering task of a group recommender system that detects groups. The system proposed in this work adopts a User-Based Collaborative Filtering approach to predict the missing data points and cluster a full rating matrix.

Experimental results confirm that overcoming the curse of dimensionality leads to great improvements in the accuracy of the group recommendations produced by a system.

Even though the presented approach solves the sparsity problem from the clustering part of the system, sparsity still affects the prediction task of the system and this is a widely-known problem in the recommender systems literature. So, improvements may be done on the system by using an hybrid approach to avoid the limitations of Collaborative Filtering (for example, we might combine a Content-Based approach, that does not rely on the ratings).

Future work will analyze how the preferences of the individual users in a group should be aggregated, in order to derive a group preference (*group modeling*). Different group modeling strategies will be studied in the scenario presented in this paper, in order to analyze the effects of the different approaches and find the best one to model the groups.

ACKNOWLEDGEMENTS

This work is partially funded by Regione Sardegna under project CGM (Coarse Grain Recommendation), through Pacchetto Integrato di Agevolazione (PIA) 2008 "Industria Artigianato e Servizi".

REFERENCES

Agrawal, R., Gehrke, J., Gunopulos, D., and Raghavan, P. (1998). Automatic subspace clustering of high dimensional data for data mining applications. In *SIGMOD 1998, Proceedings ACM SIGMOD International Conference on Management of Data*, pages 94–105. ACM Press.

Amatriain, X., Jaimes, A., Oliver, N., and Pujol, J. M. (2011). Data mining methods for recommender systems. In *Recommender Systems Handbook*, pages 39–71. Springer.

Bellman, R. (1961). *Adaptive control processes: a guided tour*. Princeton University Press Princeton, N.J.

Boratto, L. and Carta, S. (2011). State-of-the-art in group recommendation and new approaches for automatic identification of groups. In *Information Retrieval and Mining in Distributed Environments*, volume 324 of *Studies in Computational Intelligence*, pages 1–20. Springer Berlin Heidelberg.

Boratto, L. and Carta, S. (2013). Exploring the ratings prediction task in a group recommender system that automatically detects groups. In *IMMM 2013, The Third International Conference on Advances in Information Mining and Management*, pages 36–43.

Boumaza, A. M. and Brun, A. (2012). From neighbors to global neighbors in collaborative filtering: an evolutionary optimization approach. In *Genetic and Evolutionary Computation Conference, GECCO '12*, pages 345–352. ACM.

Chen, Y. and Pu, P. (2013). Cofeel: Using emotions to enhance social interaction in group recommender systems. In *Alpine Rendez-Vous (ARV) 2013 Workshop on Tools and Technology for Emotion-Awareness in Computer Mediated Collaboration and Learning*.

DeSarbo, W. S., Carroll, J. D., Clark, L. A., and Green, P. E. (1984). Synthesized clustering: A method for amalgamating alternative clustering bases with differential weighting of variables. *Psychometrika*, 49(1):57–78.

Goil, S., Nagesh, H., and Choudhary, A. (1999). Mafia: Efficient and scalable subspace clustering for very large data sets. Technical report, Northwestern University.

Goren-Bar, D. and Glinansky, O. (2004). Fit-recommend ing tv programs to family members. *Computers & Graphics*, 28(2):149–156.

Hinneburg, A. and Keim, D. A. (1999). Optimal grid-clustering: Towards breaking the curse of dimensionality in high-dimensional clustering. In *Proceedings of the 25th International Conference on Very Large Data Bases*, VLDB '99, pages 506–517. Morgan Kaufmann Publishers Inc.

Huang, J. Z., Ng, M. K., Rong, H., and Li, Z. (2005). Automated variable weighting in k-means type clustering. *IEEE Trans. Pattern Anal. Mach. Intell.*, pages 657–668.

Jameson, A. (2004). More than the sum of its members: challenges for group recommender systems. In *Proceedings of the working conference on Advanced visual interfaces, AVI 2004*, pages 48–54. ACM Press.

Jameson, A. and Smyth, B. (2007). Recommendation to groups. In *The adaptive web*, pages 596–627. Springer-Verlag, Berlin, Heidelberg.

Jing, L., Ng, M., and Huang, J. (2007). An entropy weighting k-means algorithm for subspace clustering of high-dimensional sparse data. *Knowledge and Data Engineering, IEEE Transactions on*, 19(8):1026–1041.

Jung, J. J. (2012). Attribute selection-based recommendation framework for short-head user group: An empirical study by movielens and imdb. *Expert Systems with Applications*, 39(4):4049–4054.

Kanungo, T., Mount, D. M., Netanyahu, N. S., Piatko, C. D., Silverman, R., and Wu, A. Y. (2002). An efficient k-means clustering algorithm: Analysis and implementation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24:881–892.

Makarenkov, V. and Legendre, P. (2001). Optimal variable weighting for ultrametric and additive trees and *k*-means partitioning: Methods and software. *J. Classification*, 18(2):245–271.

Masthoff, J. (2011). Group recommender systems: Combining individual models. In *Recommender Systems Handbook*, pages 677–702. Springer.

McCarthy, J. (2002). Pocket RestaurantFinder: A situated recommender system for groups. In *Workshop on Mobile Ad-Hoc Communication at the 2002 ACM Conference on Human Factors in Computer Systems*.

McCarthy, J. F. and Anagnost, T. D. (1998). Musicfx: An arbiter of group preferences for computer supported collaborative workouts. In *CSCW '98, Proceedings of the ACM 1998 Conference on Computer Supported Cooperative Work*, pages 363–372. ACM.

McCarthy, K., Salamó, M., Coyle, L., McGinty, L., Smyth, B., and Nixon, P. (2006). Cats: A synchronous approach to collaborative group recommendation. In *Proceedings of the Nineteenth International Florida Artificial Intelligence Research Society Conference*, pages 86–91. AAAI Press.

O'Connor, M., Cosley, D., Konstan, J. A., and Riedl, J. (2001). Polylens: A recommender system for groups of users. In *Proceedings of the Seventh European Conference on Computer Supported Cooperative Work*, pages 199–218. Kluwer.

Radovanovic, M., Nanopoulos, A., and Ivanovic, M. (2010). Hubs in space: Popular nearest neighbors in high-dimensional data. *Journal of Machine Learning Research*, 11:2487–2531.

Schafer, J. B., Frankowski, D., Herlocker, J. L., and Sen, S. (2007). Collaborative filtering recommender systems. In *The Adaptive Web, Methods and Strategies of Web Personalization*, pages 291–324. Springer.

Soete, G. (1988). Ovwtre: A program for optimal variable weighting for ultrametric and additive tree fitting. *Journal of Classification*, 5(1):101–104.

Zhiwen, Y., Xingshe, Z., and Daqing, Z. (2005). An adaptive in-vehicle multimedia recommender for group users. In *Proceedings of the 61st Semiannual Vehicular Technology Conference*, volume 5, pages 2800–2804.