# Identifying Grey Sheep Users By The Distribution of User Similarities In Collaborative Filtering

Yong Zheng
School of Applied Technology
Illinois Institute of Technology
Chicago, Illinois, USA 60616
yzheng66@iit.edu

Mayur Agnani
School of Applied Technology
Illinois Institute of Technology
Chicago, Illinois, USA 60616
magnani@hawk.iit.edu

Mili Singh
School of Applied Technology
Illinois Institute of Technology
Chicago, Illinois, USA 60616
msingh32@hawk.iit.edu

## ABSTRACT

Recommender Systems have been successfully applied to alleviate the information overload problem and assist the process of decision making. Collaborative filtering, as one of the most popular recommendation algorithms, has been fully explored and developed in the past two decades. However, one of the challenges in collaborative filtering, the problem of "Grey Sheep" user, is still under investigation. "Grey Sheep" users is a group of the users who have special tastes and they may neither agree nor disagree with the majority of the users. The identification of them becomes a challenge in collaborative filtering, since they may introduce difficulties to produce accurate collaborative recommendations. In this paper, we propose a novel approach which can identify the Grey Sheep users by reusing the outlier detection techniques based on the distribution of user-user similarities. Our experimental results based on the MovieLens 10M rating data demonstrate the ease and effectiveness of our proposed approach.

## CCS CONCEPTS

• **Information systems → Recommender systems**;

## KEYWORDS

recommender system; collaborative filtering; grey sheep

## 1 INTRODUCTION

Recommender system is one of the information systems which assist user's decision making by recommending a list of appropriate items to the end users tailored to their preferences. It has been successfully applied to a number of applications, such as e-commerce (e.g., Amazon, eBay), online streaming (e.g., Netflix, Pandora), social networks (e.g., Facebook, Twitter), tourism (e.g., Tripadvisor) and restaurant (e.g., Yelp), etc.

Several recommender systems have been developed to provide accurate item recommendations. There are three types of these algorithms: collaborative filtering approaches, content-based recommendation algorithms and the hybrid recommendation models [3]. Collaborative filtering (CF) is one of the most popular algorithms since it is effective and it does not rely on any content information. CF is also cost-inexpensive and easy to be interpreted. It has been well developed and applied in real practice, such as the recommendation applications on Amazon.com.

Most of the efforts by the research communities were made to improve the effectiveness of the CF algorithms, while far too little attention has been paid to the problem of "Grey Sheep" users which is one of the challenges in collaborative filtering. J. McCrae, et al. categorize the users into three classes [10]: "the majority of the users fall into the class of *White Sheep* users, where these users have high rating correlations with several other users. The *Black Sheep* users usually have very few or even no correlating users, and the case of black sheep users is an acceptable failure [1]. The bigger problem exists in the group of *Grey Sheep* users, where these users have different opinions or unusual tastes which result in low correlations with many users; and they also cause odd recommendations for their correlated users". Therefore, Grey Sheep (GS) user usually refers to "a small number of individuals who would not benefit from pure collaborative filtering systems because their opinions do not consistently agree or disagree with any group of people [5]".

Related research point out that GS users must be identified from the data and treated individually for these reasons:

- They may leave negative impact on the quality of recommendations for the White Sheep users [5–8, 10, 12, 13], especially when it comes to the collaborative filtering algorithms
- Collaborative filtering approaches do not work well for GS users [5–8, 12]. GS users should be treated separately with another type of the recommendation models, such as content-based approaches, even if there are limited number of GS users in the data.
- Due to the presence of GS users, the poor recommendations may result in critical consequences [5, 8, 10]: unsatisfied users, user defection, failure among learners, inaccurate marketing or advertising strategies, etc

There are two significant characteristics of GS users indicated by the related research: On one hand, *GS users do not agree or disagree with other users* [6, 10]. Researchers believe GS users may fall on the boundary of the user groups. Ghazanfar, et al. [6, 7] introduces a clustering technique to identify the GS users, while Gras, et al. [8] reuses the outlier detection based on the user's rating distributions. On the other hand, *GS users may have low correlations with many other users, and they have very few highly correlated*

[1] The problem of black sheep users is caused by the situation that we do not have rich or even no rating profiles for these users. It is acceptable failure since the problem can be alleviated or solved if these users will continue to leave more ratings on the items.


*RIIT'17, October 4–7, 2017, Rochester, NY, USA.*
 ISBN 978-1-4503-5120-1/17/10…$15.00
DOI: https://doi.org/10.1145/3125649.3125651

*neighbors* [5]. Unfortunately, no previous study has investigated how to take advantage of this characteristics to identify GS users in the recommender systems.

In this paper, we make the first attempt to identify GS users by exploring the distribution of user similarities or correlations. More specifically, we statistically analyze a user's correlations with all of the other users, figure out bad and good examples, and reuse the outlier detections to identify potential GS users. Statistical test is applied to examine whether the recommendation performance by the collaborative filtering approach is significantly worse for these identified GS users and other common users. The proposed approach is evaluated based on the popular MovieLens rating data.

## 2 RELATED WORK

We introduce collaborative filtering first, and discuss corresponding progress of GS user identification in this section.

### 2.1 Preliminary: Collaborative Filtering

Recommender systems have been demonstrated as useful tools to assist user's decision making. For example, Netflix may recommend a list of movies that you may be interested in, and Amazon may suggest appropriate books that you may want to purchase, while users do not need to issue a query to look for something they want. The idea behind these recommender systems is that the recommendation model can infer which items a user may like based on user's preference history, such as how a user rates a movie, which items they browse online, or the purchase/order history in user profiles.

**Table 1: Example of a Movie Rating Data**

| | Pirates of the Caribbean 4 | Kung Fu Panda 2 | Harry Potter 6 | Harry Potter 7 |
|---|---|---|---|---|
| $U_1$ | 4 | 4 | 1 | 2 |
| $U_2$ | 3 | 4 | 2 | 1 |
| $U_3$ | 2 | 2 | 4 | 4 |
| $U_4$ | 4 | 4 | 1 | ? |

Rating prediction is a common task in the recommender systems. Take the movie rating data shown in Table 1 for example, there are four users and four movies. The values in the data matrix represent users' rating on corresponding movies. We have the knowledge about how the four users rate these movies. And we'd like to learn from the knowledge and predict how the user $U_4$ will rate the movie "Harry Potter 7".

One of the most popular recommendation algorithms is collaborative filtering [10, 13]. There are memory-based collaborative filtering, such as the user-based collaborative filtering (UBCF) [11], and model-based collaborative filtering, such as matrix factorization. In this paper, we focus on the UBCF since it may suffer from the problem of GS users seriously.

The assumption in UBCF is that a user's rating on one movie is similar to the preferences on the same movie by a group of $K$ users. This group of the users is well known as $K$ nearest neighbors (KNN). Namely, they are the top-$K$ users who have similar tastes with a given user. Take Table 1 for example, to find the KNN for user $U_4$, we observe the ratings given by the four users on the given movies except "Harry Potter 7". We can see that $U_1$ and $U_2$ actually give similar ratings as $U_4$ – high ratings (3 or 4-star) on the first two movies and low rating on the movie "Harry Potter 6". Therefore, we infer that $U_4$ may rate the movie "Harry Potter 7" similarly as how the $U_1$ and $U_2$ rate the same movie.

To identify the KNN, we can use similarity measures to calculate user-user similarities or correlations, such as the cosine similarity shown by Equation 1.

$$sim(U_i, U_j) = \frac{\overrightarrow{R_{U_i}} \bullet \overrightarrow{R_{U_j}}}{\|\overrightarrow{R_{U_i}}\|_2 \times \|\overrightarrow{R_{U_j}}\|_2} \tag{1}$$

We use a rating matrix similar to Table 1 to represent our data. $\overrightarrow{R_{U_i}}$ and $\overrightarrow{R_{U_j}}$ are the row vectors for user $U_i$ and $U_j$ respectively, where the rating is set as zero if a user did not rate the item. The size of these rating vectors is the same as the number of movies. In Equation 1, the numerator represents the dot product of the two user vectors, while the denominator is the multiplication of two Euclidean norms (i.e, L2 norms). The value of $K$ in KNN refers to the number of the top similar neighbors we need in the rating prediction functions. We need to tune up the performance by varying different numbers for $K$.

Once the KNN are identified, we can predict how a user rates one item by the rating function described by Equation 2.

$$P_{a,t} = \bar{r}_a + \frac{\sum_{u \in N} (r_{u,t} - \bar{r}_u) \times sim(a, u)}{\sum_{u \in N} sim(a, u)} \tag{2}$$

where $P_{a,t}$ represents the predicted rating for user $a$ on the item $t$. $N$ is the top-$K$ nearest neighborhood of users $a$, and $u$ is one of the users in this neighborhood. The *sim* function is a similarity measure to calculate user-user similarities or correlations, while we use cosine similarity in our experiments. Accordingly, $r_{u,t}$ is neighbor $u$'s rating on item $t$, $\bar{r}_a$ is user $a$'s average rating over all items, and $\bar{r}_u$ is $u$'s average rating.

This prediction function tries to aggregate KNN's ratings on the item $t$ to estimate how user $a$ rates $t$. However, the predicted ratings may be not accurate if user $a$ is a GS user, since the user similarities or correlations between $a$ and his or her neighbors may be very low. From another perspective, if a GS user is selected as one of the neighbors for a common user, it may result in odd recommendations or predictions since GS users may have unusual tastes on the items.

### 2.2 Grey Sheep User Identifications

There are several research [5, 10, 12, 13] that point out the problem of GS user, define or summarize the characteristics of GS users, but very few of the existing work were made to figure out the solutions to identify GS users.

As mentioned previously, there are two mainstream statements as the characteristics for the GS users: *GS users do not agree or disagree with other users* [6, 10]. Researchers believe GS users may fall on the boundary of the user groups. Ghazanfar, et al. [6, 7] proposes a clustering technique to identify the GS users, while they define improved centroid selection methods and isolates the GS users from the user community by setting different user similarity thresholds. The main drawback in their approach is the difficulty to find the optimal number of clusters, as well as the high computation cost to

end up convergence in the clustering process, not to mention the unpredictable varieties by initial settings and other parameters in the technique. In their experiments, they demonstrate that content-based recommendation algorithms can be applied to improve the recommendation performance for the GS users. By contrast, Gras, et al. [8] reuses the outlier detection based on the user's rating distributions. They additionally take the imprecision of ratings (i.e., prediction errors) into account. However, the rating prediction error can only be used to evaluate whether a user is a GS user, it may not be appropriate to utilize it to identify GS users. It is because GS user is not the only reason that leads to large prediction errors. From another perspective, a user associated with large prediction errors is not necessary to be a GS user.

Another characteristics is that *GS users may have low correlations with many other users, and they have very few highly correlated neighbors* [5]. Unfortunately, no previous study has investigated how to take advantage of this characteristics to identify GS users in the recommender systems. This characteristics is highly related to the distribution of user-user similarities or correlations. In this paper, we make the first attempt to identify GS users by exploring the distribution of their user correlations. Note that our work is different from the Gras, et al. [8]'s work, since they stay to work on the distribution of user ratings, while we exploit the distribution of user similarities.

## 3 METHODOLOGIES

As mentioned in [5], White Sheep users are the common users that have high correlations with other users. Namely, we can find a set of good KNN for White Sheep users. By contrast, GS users have correlations with other users but most of the correlations are relatively low.

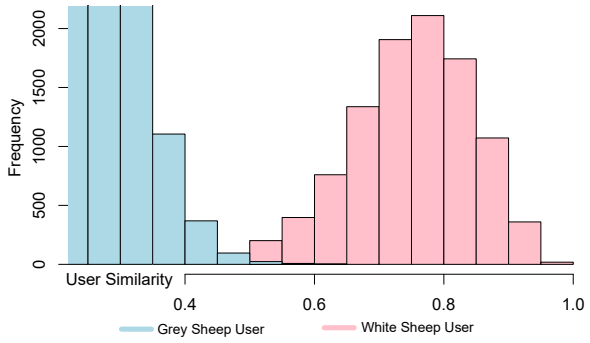


**Figure 1: Distribution of User Similarities by Example of White Sheep and Grey Sheep Users**

For each user, we are able to obtain his or her similarities with other users based on the cosine similarity described by Equation 1. The distribution of user similarities can be depicted by a histogram as shown in Figure 1. The x-axis is the value of user similarities, and the y-axis tells how many similarities values (i.e., the frequency) fall in each bin of the user similarities. A White Sheep user usually has higher correlations with other users, therefore its distribution of user similarities is expected to be left-skewed and the frequency at higher similarities should be significantly larger. In terms of the GS users, we do not have many high correlations with other users, and most of the user similarities are low. It may result in the blue histogram in Figure 1 which presents a heavily right-skewed distribution.

Imagine that there could be thousands of or even more users in a data set. The process of identifying the GS users becomes the procedure of distinguishing GS users from the White Sheep users, where the distribution of user similarities for GS users may meet the following requirements:

- It is usually a right-skewed distribution.
- The descriptive statistics of the user similarities, such as the first, second and third quartiles (q1, q2, q3), as well as the mean of the correlations, may be relatively smaller, since GS users have low correlations with other users.

However, it is difficult to define how small the user similarity is we can say the user is a GS user. The same thing happens to the degree of the skewness. It may not be practical to simply define a threshold for the user similarities or the skewness of the distributions to identify the GS users.

Outlier detection [4, 9], as a result, becomes one of the potential solutions to help us distinguish GS users from the group of common users. It refers to the process of the identification of observations which do not conform to an expected pattern or other items in a dataset [4]. The idea behind is that we can detect abnormal observations by defining the common or good examples in the data. The outlier detection could be a supervised, unsupervised or semi-supervised learning process. Gras, et al. [8]'s work also points out that the identification of GS users is closely related to the outlier detection problem in data mining.

Our proposed methodologies can be summarized by the following four steps: distribution representations, example selection, outlier detection and examination of GS users.

### 3.1 Distribution Representations

The first step is to obtain user-user similarities and represent the distribution of user similarities for each user in the data set. We use the cosine similarity described by Equation 1 to calculate the user-user similarity between every pair of the users. Note that the similarity of two users may be zero if there are no co-rated items by them. We remove the zero similarities from the distribution, since we only focus on the known user-user similarities in our data.

**Table 2: Example of Distribution Representations**

| User | q1 | q2 | q3 | Mean | STD | Skewness |
|---|---|---|---|---|---|---|
| 40459 | 0.051 | 0.089 | 0.133 | 0.098 | 0.060 | 0.964 |
| 7266 | 0.028 | 0.056 | 0.091 | 0.064 | 0.045 | 1.245 |
| 34975 | 0.128 | 0.181 | 0.243 | 0.193 | 0.093 | 0.671 |
| 34974 | 0.093 | 0.149 | 0.209 | 0.156 | 0.084 | 0.568 |
| 34977 | 0.047 | 0.077 | 0.121 | 0.112 | 0.115 | 2.516 |
| ... | ... | ... | ... | ... | ... | ... |

As a result, we are able to obtain a list of non-zero user-user similarities for each user. We further represent each user by the descriptive statistics of his or her distribution of the user similarities, including, q1, q2, q3, mean, standard deviation (STD) and skewness, as shown by Table 2.

### 3.2 Example Selection

To apply the outlier detection, we need to select *good* (i.e., White Sheep users) and *bad* (i.e., potential GS users) examples in order to construct a user matrix similar to Table 2. This step is necessary especially when there are large scale of the users in the matrix. Note that we are not going to label these examples, since we are not exactly sure which users are White Sheep or GS users, and the following outlier detection technique we use is not a supervised learning process.

According to the definition of White Sheep and GS users in [5], we can find many highly correlated neighbors with White Sheep users, while GS users usually have relatively low correlations with others. We believe the good examples may have higher values in similarity statistics, including the q1, q2, q3 and mean similarity in Table 2. By contrast, these statistics for bad examples may be much lower. In addition, the distribution of user similarities associated with the good examples may be left-skewed, while it is more likely to be right-skewed for the bad examples, as shown in Figure 1.

We suggest to filter the users by the descriptive statistics of their similarity distributions, such as the first quartile (q1), the second quartile (q2), the third quartile (q3), as well as mean of the similarity values, etc. More specifically, the bad examples could be selected by the following constraints:

- **Low similarity statistics**: In this case, q1, q2, q3 and mean may be much smaller than other users. We can select a lower-bound as the threshold. For example, if a user's mean similarity is smaller than *the first quartile* of mean similarities (i.e., the list of mean values over all of the users), this user is selected as one of the bad examples. The constraints could be flexible. They can be applied to the mean similarity only, or they could be applied to any subsets of {q1, q2, q3, mean} at the same time.
- **The degree of skewness**: This time, we apply a constraint on the skewness. For example, if a user's skewness value in his or her similarity distribution is larger than *the third quartile* of skewness values over all of the users, this user may be selected as one of the bad examples. It is because GS users may have very few highly correlated neighbors, and most of their user correlations are pretty low, which results in a heavily right-skewed similarity distribution.

Again, the constraints could be very flexible. On one hand, they can be applied on any selected descriptive statistics, such as q1, mean or skewness. On another hand, the threshold could be any reasonable ones. For example, we can set the threshold as *the first quartile* of mean similarities or the *average value* of mean similarities. Flexible constraints may select large proportion of the bad examples, while the strict constraints may result in limited number of candidates. The best selection may vary from data to data. We use similar strategy to select the good examples, but the good examples must have high similarity statistics and a relatively left-skewed distribution of the user similarities.

### 3.3 Outlier Detection

There are several outlier detection [4, 9] techniques, such as the probabilistic likelihood approach, the clustering based or the density based methods, etc. In this paper, we adopt a density based method which relies on the local outlier factor (LOF) [2]. LOF is based on the notion of local density, where locality is given by the $\mathbb{k}$ nearest neighbors[2] whose distance is used to estimate the density. The nearest neighbor, in our case, can be produced by using distance metrics on the feature matrix, while the feature matrix is the distribution representation matrix as shown in Table 2. By comparing the local density of a user to the local densities of his or her neighbors, one can identify regions of similar density, and the users that have a substantially lower density than their neighbors can be viewed as the outliers (i.e., the GS users) finally. Due to that the distances among the users are required to be calculated, we apply a normalization to the matrix in Table 2 in order to make sure all of the columns are in the same scale.

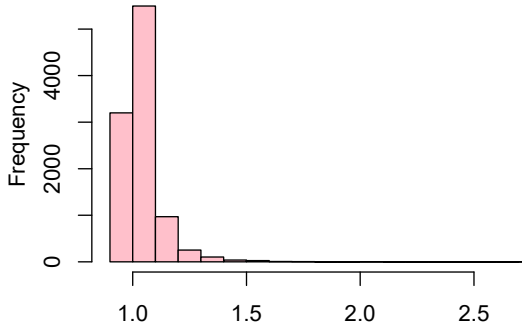


**Figure 2: Histogram of LOF Scores**

Figure 2 presents an example of the histogram of LOF scores, while x-axis represents the LOF score and each score is associated with a single user. A user will be viewed as a common user if his or her LOF score is close to the value of 1.0. By contrast, it can be an outlier (i.e., GS user) if the LOF score is significantly larger or smaller than 1.0. We set a threshold for the LOF score, and tune up the results by varying the values of $\mathbb{k}$ and the LOF threshold in our experiments in order to find qualified GS users as many as possible.

### 3.4 Examinations

With different values of $\mathbb{k}$ and the LOF threshold, we are able to collect different sets of the users as the outliers or GS users. We use the following approaches to examine the quality of the GS users:

- The recommendation performance for the group of GS users by collaborative filtering must be significantly worse than the performance for the White Sheep users. More specifically, the average rating prediction errors (see Section 4.1) based on the rating profiles associated with these GS users must be significantly higher than the errors that are associated with non-GS users. If the prediction errors for GS users and the remaining group of the users are close, we will perform two-independent sample statistical test to examine the degree of significance.
- We additionally visualize the distribution of similarities for GS users, in comparison with the one by non-GS users. The ideal visualization is expected to be similar to the one shown by Figure 1, where the distributions for GS and White sheep users are right and left-skewed respectively.

[2]We use $\mathbb{k}$ to distinguish it from the $K$ in $K$NN based UBCF algorithm.

By meeting the basic requirements above, we continue to tune up the values of $\mathbb{k}$ and the LOF threshold, in order to find GS users as many as possible. But note that GS users are always a small proportion of the users in the data.

# 4 EXPERIMENTS AND RESULTS

## 4.1 Experimental Settings

We use the MovieLens 10M rating data set[3] which is a large-scale movie rating data available for research. In this data, we have around 10 million ratings given by 72,000 users on 10,000 movies. Since this data is big enough, we simply split the data into training and testing set, where the training set is 80% of the whole data. Each user has rated at least 20 movies. We believe these users have rich rating profiles, and black sheep users are not included in this data.

We apply our proposed methodologies on the training set to identify GS users, and examine them by the recommendation performance over the test set. To obtain the prediction errors, we apply UBCF described by Equation 2 as the collaborative filtering recommendation algorithm. In UBCF, we adopt the cosine similarity to measure the user-user similarities, and vary different value of $K$ ($K$ = 50 is the besting setting in our experiments) in order to find the best KNN. The recommendation performance is measured by mean absolute error (MAE) which can be depicted by Equation 4.1. $R$ represents the test set, where $|R|$ denotes the total number of ratings in the test set. $R_{a,t}$ is the actual rating given by user $a$ on item $t$. $(a,t)$ is the <user, item> tuple in the test set. $P_{a,t}$ is the predicted rating by the function in Equation 2. The "abs" function is able to return the absolute value of the prediction error.

$$MAE = \frac{1}{|R|} \sum_{(a,t)\epsilon R} abs(P_{a,t} - R_{a,t}) \tag{3}$$

## 4.2 Results and Findings

We follow the four steps in Section 3 to identify the GS users from the training set. As mentioned in the Section 3.2, it is flexible to set different constraints to select good and bad examples. In our experiments, we tried both strict and loose constraints, and we find that we are able to identify more qualified GS users when we use the strict constraints. The strict constraints can be described as follows: we go through the distribution representation matrix, and select the bad examples (i.e., potential GS users) if his or her q1, q2 and mean similarity value is smaller than the first quartile of the q1, q2 and mean distribution of all the users. This group of bad examples is further filtered by the skewness – skewness value smaller than the third quartile of the skewness distribution over all the users will be removed. Finally, we obtain 1,010 bad examples. By applying the loose constraints, we do find more bad examples, but we cannot identify more qualified GS users finally. Therefore, we only present the results based on the strict constraints in the following paragraphs. In terms of the good examples, we select the users whose mean similarity is larger than the third quartile of mean values over all the users. Due to that there are too many users in the pool, we finally perform a random sampling to use just 20% of these users as the good examples.

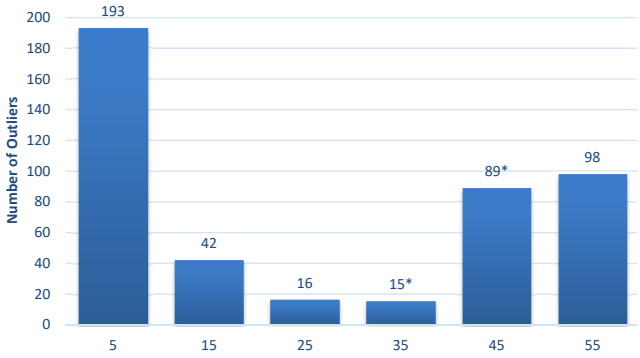


**Figure 3: The number of outliers identified by different $\mathbb{k}$ values. Note that (*) tells that the two-independent sample statistical test was failed in that setting.**

Afterwards, we blend the good and bad examples, and apply the LOF technique to identify the GS users. We tried different values of $\mathbb{k}$ and LOF thresholds in our experiments. The number of outliers identified can be shown by Figure 3. For each value of $\mathbb{k}$, we vary the LOF threshold in order to find the largest number of outliers. By using $\mathbb{k}$ = 5 and the LOF threshold as 1.34, we are able to identify 193 outliers as the GS users. It is not surprising, since GS users are always a small proportion of the users in the data. In addition, the group of outliers can only be considered as GS users if the MAE of the rating profiles associated with these users is significantly larger than the MAE based on the non-GS users. We use 95% as confidence level, and apply the two-independent sample statistical test to examine whether they meet this requirement. The test failed only when $\mathbb{k}$ equals to 35 and 45. As a result, the best setting in our experiment is that $\mathbb{k}$ equals 5 and LOF threshold as 1.34, since they help us identify the largest number of qualified GS users.

**Table 3: MAE Results**

| All Users | Good Examples | Bad Examples | Remaining Users | GreySheep Users |
|---|---|---|---|---|
| 0.869 | 0.868 | 0.868 | 0.869 | 0.908 |

Table 3 describes the MAE evaluated based on the rating profiles in the test set associated with different groups of the users. The "remaining users" refer to users excluding the identified GS users. There are no statistically differences on MAE values for these user groups if we do not take the group of GS uses into account. In terms of the 193 GS users we find, the MAE value is 0.908. The increase in MAE compared with other group of the users is 4.6%. We further perform a two-independent statistical test on the MAE values by the group of GS users and other group of users, in order to examine whether the result is statistically significant. We use 95% as confidence level, and the statistic test produces p-values smaller than 0.05, which confirms that the MAE by the GS users is

[3]https://grouplens.org/datasets/movielens/10m/

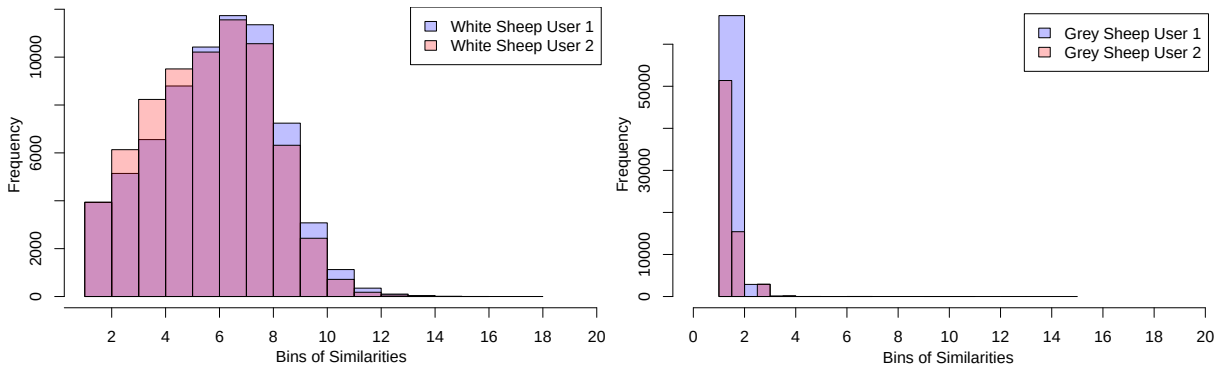


**Figure 4: Visualization of the Similarity Distributions**

significantly higher. By this way, we confirm the 193 outliers we find can be viewed as GS users.

We look into the characteristics of identified GS and White Sheep users. We select two GS users and two White Sheep users as the representatives, visualize the distribution of user similarities, as shown in Figure 4. The bars in slate blue and coral present the histograms for two users, while the bars in plum capture the overlaps between two histograms. The x-axis is the bins of the similarities, while we put the similarity values (in range [0, 1]) into 20 bins with each bin size as 0.05. The y-axis can tell how many similarity or correlation values that fall in corresponding bins. Note that the distribution based on user similarities for White Sheep users is not the same as the ideal situation described in Figure 1. The reason why is that there are no left-skewed similarity distributions in this data. Most of the user's similarity distributions are shown as normal, slightly right-skewed or heavily right-skewed. From Figure 4, we can clearly observe that most of the correlations between GS users and other users are pretty low, which presents a heavily right-skewed distribution of the similarities. The situation is much better for the White Sheep users, since they usually have highly correlated neighbors. According to the observations at the bins from 15 and 20, we can discover that we have at least 100 high correlations for the White Sheep users, but almost zero for the GS users. This pattern is consistent with the definition of GS and White Sheep users in [10]. According to previous research [6, 10], we need to apply other recommendation algorithms (such as content-based approaches) to reduce the prediction errors for these GS users, where we do not explore further in this paper.

## 5 CONCLUSIONS

In this paper, we propose a novel approach to identify Grey Sheep users based on the distribution of user similarities or correlations. The proposed approach in this paper is much easier than the previous methods [6, 8, 10] in terms of the complexity. In our future work, we will compare the proposed approach with existing ones to explore whether our approach is significantly better. Furthermore, the same approach can also be used to identify *Grey Sheep items* in addition to the Grey Sheep users.

The problem of Grey Sheep users may not only happen in the traditional recommender systems, but also exist in other types of the recommender systems. For example, in the context-aware recommender systems [1, 14, 15], the definition of Grey Sheep users could be the users who have unusual tastes in specific contextual situations. The proposed approach in this paper can be easily extended to these special recommender systems, and we may explore it in our future work.

## REFERENCES


[1] Gediminas Adomavicius, Bamshad Mobasher, Francesco Ricci, and Alexander Tuzhilin. 2011. Context-Aware Recommender Systems. *AI Magazine* 32, 3 (2011), 67–80.

[2] Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. 2000. LOF: identifying density-based local outliers. In *ACM sigmod record*, Vol. 29. ACM, 93–104.

[3] Robin Burke. 2002. Hybrid recommender systems: Survey and experiments. *User modeling and user-adapted interaction* 12, 4 (2002), 331–370.

[4] Varun Chandola, Arindam Banerjee, and Vipin Kumar. 2009. Anomaly detection: A survey. *ACM computing surveys (CSUR)* 41, 3 (2009), 15.

[5] Mark Claypool, Anuja Gokhale, Tim Miranda, Pavel Murnikov, Dmitry Netes, and Matthew Sartin. 1999. Combining content-based and collaborative filters in an online newspaper. In *Proceedings of ACM SIGIR workshop on recommender systems*, Vol. 60.

[6] Mustansar Ghazanfar and Adam Prugel-Bennett. 2011. Fulfilling the Needs of Gray-Sheep Users in Recommender Systems, A Clustering Solution. In *Proceedings of the 2011 International Conference on Information Systems and Computational Intelligence*. 18–20.

[7] Mustansar Ali Ghazanfar and Adam Prügel-Bennett. 2014. Leveraging clustering approaches to solve the gray-sheep users problem in recommender systems. *Expert Systems with Applications* 41, 7 (2014), 3261–3275.

[8] Benjamin Gras, Armelle Brun, and Anne Boyer. 2016. Identifying Grey Sheep Users in Collaborative Filtering: a Distribution-Based Technique. In *Proceedings of the 2016 Conference on User Modeling Adaptation and Personalization*. ACM, 17–26.

[9] Victoria Hodge and Jim Austin. 2004. A survey of outlier detection methodologies. *Artificial intelligence review* 22, 2 (2004), 85–126.

[10] John McCrae, Anton Piatek, and Adam Langley. 2004. Collaborative filtering. *http:// www. imperialviolet. org* (2004).

[11] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. 1994. GroupLens: an open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 ACM conference on Computer supported cooperative work*. ACM, 175–186.

[12] Manuela Ruiz-Montiel and José Aldana-Montes. 2009. Semantically enhanced recommender systems. In *On the move to meaningful internet systems: OTM 2009 workshops*. Springer, 604–609.

[13] Xiaoyuan Su and Taghi M Khoshgoftaar. 2009. A survey of collaborative filtering techniques. *Advances in artificial intelligence* 2009 (2009), 4.

[14] Y. Zheng, B. Mobasher, and R. Burke. 2014. CSLIM: Contextual SLIM Recommendation Algorithms. In *Proceedings of the 8th ACM Conference on Recommender Systems*. ACM, 301–304.

[15] Yong Zheng, Bamshad Mobasher, and Robin Burke. 2015. Similarity-Based Context-aware Recommendation. In *Proceedings of the 2015 Conference on Web Information Systems Engineering*. Springer Berlin Heidelberg, 431–447.