Expert Systems with Applications 41 (2014) 3261–3275

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Leveraging clustering approaches to solve the gray-sheep users problem in recommender systems

Mustansar Ali Ghazanfar [a,*], Adam Prügel-Bennett [b,1]

[a] Department of Software Engineering, Faculty of Telecommunication and Information Engineering, University of Engineering and Technology, Taxila, Pakistan
[b] School of Electronics and Computer Science, University of Southampton, Highfield Campus, Southampton SO17 1BJ, United Kingdom

ARTICLE INFO

Keywords:
Gray-sheep users
Recommender systems
Collaborative filtering
K-Means clustering

ABSTRACT

Recommender systems apply data mining and machine learning techniques for filtering unseen information and can predict whether a user would like a given item. This paper focuses on gray-sheep users problem responsible for the increased error rate in collaborative filtering based recommender systems. This paper makes the following contributions: we show that (1) the presence of gray-sheep users can affect the performance – accuracy and coverage – of the collaborative filtering based algorithms, depending on the data sparsity and distribution; (2) gray-sheep users can be identified using clustering algorithms in offline fashion, where the similarity threshold to isolate these users from the rest of community can be found empirically. We propose various improved centroid selection approaches and distance measures for the K-means clustering algorithm; (3) content-based profile of gray-sheep users can be used for making accurate recommendations. We offer a hybrid recommendation algorithm to make reliable recommendations for gray-sheep users. To the best of our knowledge, this is the first attempt to propose a formal solution for gray-sheep users problem. By extensive experimental results on two different datasets (MovieLens and community of movie fans in the FilmTrust website), we showed that the proposed approach reduces the recommendation error rate for the gray-sheep users while maintaining reasonable computational performance.

1. Introduction

1.1. Recommender systems

There has been an exponential increase in the volume of available digital information (e.g., videos in Youtube (youtube.com) and Netflix (netflix.com), music in LastFm (last.fm)), electronic resources (e.g., research papers in CiteULike (citeulike.org)), and online services (e.g., Flicker (flickr.com), Delicious (delicious.com), Amazon (amazon.com)) in recent years. This information overload has created a potential problem, which is how to filter and efficiently deliver relevant information to a user. Furthermore, information needs to be prioritised for a user rather than just filtering the right information; otherwise, it could become overwhelming. Search engines help Internet users by filtering pages to match explicit queries, but it is very difficult to specify what a user wants by using simple keywords. The Semantic Web also provides some help to find useful information by allowing intelligent search queries; however, it depends on the extent to which the web pages are annotated. These problems highlight a need for information filtering systems that can filter unseen information and can predict whether a user would like a given resource. Such systems are called *recommender systems*, and they mitigate the aforementioned problems to a great extent. Example of the recommender system are the *Amazon* recommender engine (Linden, Smith, & York, 2003) *Youtube* (www.youtube.com) video recommender service and *MovieLens* (www.movielens.com) movie recommender system, which recommend videos and movies based on the person's opinions.

A recommender system consists of two basic entities: users and items, where users provide their opinions (ratings) about items. We denote these users by $\mathcal{U} = \{u_1, u_2, \ldots, u_M\}$, where the number of people using the system is $|\mathcal{U}| = M$, and denote the set of items being recommended by $\mathcal{I} = \{i_1, i_2, \ldots, i_N\}$, with $|\mathcal{I}| = N$. The users will have rated some, but not all, of the items. We denote these ratings by $(r_{iu}|(i,u) \in \mathcal{D})$, where $\mathcal{D} \subset \mathcal{I} \times \mathcal{U}$ is the set of user-item pairs that have been rated. We denote the total number of ratings made by $|\mathcal{D}| = T$. Typically each user rates only a small number of the possible items, so that $|\mathcal{D}| = T \ll |\mathcal{I} \times \mathcal{U}| = N \times M$. It is not unusual in practical systems to have $T/(N \times M) \approx 0.01$. The set of possible ratings made by the users can be thought of as elements of an $M \times N$ rating matrix R. We denote the items for which there are ratings by user u as $\mathcal{D}_u$, and the users who have rated an item i by $\mathcal{D}_i$. The task is to create a recommendation algorithm that predicts an unseen rating r_{iu}, i.e., for $(i,u) \notin \mathcal{D}$.

* Corresponding author. Tel.: +92 051 9047 566; fax: +92 051 9047 420.
E-mail addresses: eng.musi@gmail.com (M.A. Ghazanfar), apb@ecs.soton.ac.uk (A. Prügel-Bennett).
[1] Tel.: +44 023 80594473; fax: +44 023 80594498.

0957-4174/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.eswa.2013.11.010

1.2. Main types of recommender system

There are two main types of recommender systems: collaborative filtering (CF) and content-based filtering recommender systems, as discussed below:

- *Collaborative filtering (CF)*: Collaborative filtering recommender systems (Konstan et al., 1997; Pennock, Horvitz, Lawrence, & Giles, 2000; Shardanand & Maes, 1995) recommend items by taking into account the taste (in terms of preferences of items) of users, under the assumption that users will be interested in items that users similar to them have rated highly. Examples of these systems include the GroupLens system (Konstan et al., 1997), and Ringo (www.ringo.com). Collaborative filtering can be classified into two sub-categories as follows:
  - *Memory-based approaches*: Memory-based approaches (Shardanand & Maes, 1995) make a prediction by taking into account the entire collection of previous rated items by a user. Examples of these systems include GroupLens recommender systems (Konstan et al., 1997; Resnick, Iacovou, Suchak, Bergstrom, & Riedl, 1994).
  - *Model-based approaches*: Model-based approaches use rating patterns of users in the training set, group users into different classes, and use ratings of predefined classes to generate recommendation for an *active user* on a *target item* (refer to Appendix D). Examples of these systems include item-based CF (Sarwar, Karypis, Konstan, & Reidl, 2001), Singular Value Decomposition (SVD) based models (Ghazanfar & Prügel-Bennett, 2013; Kurucz, Benczúr, & Csalogány, 2007; Sarwar, Karypis, Konstan, & Riedl, 2000b; Vozalis & Margaritis, 2007), Bayesian networks (Breese, Heckerman, & Kadie, 1998), clustering models (Park & Tuzhilin, 2008; Rashid, Lam, Karypis, & Riedl, 2006; Sarwar, Karypis, Konstan, & Riedl, 2002; Xue et al., 2005), and Kernel-mapping recommender (Ghazanfar, Szedmák, & Prügel-Bennett, 2011; Ghazanfar, Prügel-Bennett, & Szedmák, 2012).
- *Content-based filtering (CBF)*: Content-based filtering recommender systems (Lang, 1995; van Meteren & van Someren, 2000) recommend items based on the content information of an item, under the assumption that users will like similar items to the ones they liked before. In these systems, an item of interest is defined by its associated features, for instance, NewsWeeder (Lang, 1995), a newsgroup filtering system uses the words of text as features. The textual description of items is used to build item profiles. User profiles can be constructed by building a model of the user's preferences using the descriptions and types of the items that a user is interested in, or a history of user's interactions with the system is stored (e.g., user purchase history, types of items they purchased together, etc.).

  Furthermore, hybrid recommender systems have been proposed (Burke, 2002; Ghazanfar & Prügel-Bennett, 2010c; Lucas et al., 2013), which combine individual recommender systems to avoid certain limitations of individual recommender systems.

Recommendations can be presented to an active user in the followings two different ways: by predicting ratings of items, a user has not seen before and by constructing a list of items ordered by their preferences also called top-N recommendations (Sarwar et al., 2000b). In this paper, we focus on both of these approaches.

1.3. Problem statement

Two of the important design objectives of a recommender system are accuracy and scalability. In the Collaborative Filtering (CF) domain, they are in conflict, since the less time an algorithm spends searching for neighbours, the more scalable it will be, but produces worse quality recommendations. The CF approaches based on K-means clustering algorithms have been proposed to increase the scalability of recommender systems. We investigate how to improve the quality of clusters and recommendations focusing on the following key issues[2]:

1. How do different centroid selection approaches affect the quality of clusters/recommendations?
2. How does the choice of distance metric affect the quality of clusters/recommendations?

Humans typically do not have predictable simple taste—they rate items differently and the reasons for rating an item are likely to be complex. In the CF domain, the correlation coefficient (in the case of Pearson correlation) between two users varies between 1, indicting absolute agreement, to −1, indicating absolute disagreement between two users. Based on the correlation coefficient, we can categorise users into two main classes[3]: (1) white sheep—the users who have high correlation value with many other users; and (2) gray-sheep—the users who partially agree/disagree with other users and have low correlation coefficient with almost all users.

In this paper, we systematically explore the gray-sheep users problem. Specifically, we look at four key questions:

1. How can the gray-sheep users be effectively detected in a recommender system?
2. Does the presence of the gray-sheep users affect the recommendation quality of the community?
3. How do the CF algorithms perform over these users?
4. How do the text categorisation algorithms trained on the content profiles perform over these users?

We proposed a clustering solution to detect the gray-sheep users in off-line fashion. We offered a *switching hybrid recommender system* (Burke, 2002) and showed that the proposed approach reduces the recommendation error rate for the gray-sheep users while maintaining reasonable computational performance. To the best of our knowledge, this is the first attempt to propose a formal solution to satisfy the needs of gray-sheep users. We evaluate our algorithm over the MovieLens (www.movielens.org) and FilmTrust (www.trust.mindswap.org/FilmTrust) datasets.

The rest of the paper has been organised as follows. In Section 2, we present the related work by giving an overview of different clustering algorithms and shed light on the gray-sheep users problem. In Section 3, we present various centroid selection algorithms. In Section 4, we discuss various distance measures that we have used in this work. We outline our algorithm to detect the gray-sheep users in Section 5. We briefly describe the experimental setup in Section 6. In Section 7, we present the results in detail. Section 8 gives a brief discussion followed by the conclusion in Section 9.

2. Related work

In this section, we give a brief overview of clustering algorithms that have been used in recommender systems. We then discuss the gray-sheep users problem and describe how the recommender systems community has overlooked this problem.[4]

[2] Refer to Appendix D for definition of different terms (such as centroid, text categorisation, distance, etc.) used in this work.

[3] Some authors have used another class "black-sheep" for the users having no (or very few) other users with whom they correlate. The CF-based algorithms cannot make predictions for these users (Su & Khoshgoftaar, 2009).

[4] This is an extended version of our previous work (Ghazanfar & Prügel-Bennett, 2011).