A Novel Deterministic Framework for Non-probabilistic Recommender Systems

Avinash Bhat, Divya Madhav Kamath and C. Anitha

Abstract Recommendation is a technique which helps and suggests a user, any relevant item from a large information space. Current techniques for this purpose include non-probabilistic methods like content-based filtering and collaborative filtering (CF) and probabilistic methods like Bayesian inference and Case-based reasoning methods. CF algorithms use similarity measures for calculating similarity between users. In this paper, we propose a novel framework which deterministically switches between the CF algorithms based on sparsity to improve accuracy of recommendation.

Keywords Collaborative filtering · Non-probabilistic approach · Switching recommender systems · Sparsity · Similarity metrics · Genre independence

1 Introduction

Recommender systems are used for prediction of the behavior of a user to provide personalized content and services. A recommendation is a suggestion made to a user, keeping in mind his previous choices to improve his decision-making process. A recommender system carries this operation and predicts the user's rating to a product or a service without overwhelming them with sizeable data [1–3]. To provide relevant recommendations, a system must understand user's preferences, which can be done using Collaborative Filtering or Content-based Filtering or Hybrid methods [1, 3–5].

A. Bhat (✉) · C. Anitha
The National Institute of Engineering, Mananthavadi Road, Vidayaranya Puram, Mysuru, Karnataka, India
e-mail: avinashbhatneelavar@gmail.com

C. Anitha
e-mail: anithac.cse@nie.ac.in

D. M. Kamath
SJCE, JSS STU Campus, Manasagangotri, Mysuru, Karnataka, India
e-mail: divyamadhavkamath@gmail.com

A. Abraham et al. (eds.), *Emerging Technologies in Data Mining and Information Security*, Advances in Intelligent Systems and Computing 813,
https://doi.org/10.1007/978-981-13-1498-8_8

This paper aims to provide a framework for recommender systems which gives better accuracy, based on sparsity of the dataset. We focus on scope of switching recommender system, a type of hybrid recommender, which can switch between techniques based on a certain criterion [5].

Collaborative Filtering includes methods which make recommendations depending on behavioral analysis of other users in the system. These algorithms predict the preferences of the users and then rank the items based on the preferences to generate the recommendation list [1, 3, 6]. CF algorithms are of two types, User-based CF and Item-based CF. User based CF is also called user–user CF in which a group of users whose ratings similar to that of one user is listed and their behavior is analyzed collectively to predict preferences of the target user [1, 3, 7]. Item-based CF is also called item-item CF in which the recommendations are generated based on similarity between the rated items to other items which are not rated [1, 3, 7].

The concept of similarity metrics is used in data mining for determining the similarity between items or objects [1, 8]. Similarity is a value in the range [0, 1] that gives the measure of the degree to which the two items are alike [7]. Here, 0 corresponds to no similarity and 1 corresponds to total similarity. There are a number of measures used to calculate the similarity, few of which have been discussed in this paper.

- Euclidean Distance—Distance between two points in a Euclidean space is called Euclidean Distance. This notion is employed in recommender systems for similarity identification, more close two points are, more the similarity [8, 9]. The Euclidean distance E(x, y) between two points x = (x1, y1) and y = (x2, y2) is given by

$$E(x, y) = \sqrt{(x1 - x2)^2 + (y1 - y2)^2} \tag{1}$$

- Manhattan Distance—Collective summation of the extent from initial point to end point when traveled strictly vertical and/or horizontal. It returns the absolute distance between two data points [10]. Mathematical expression is,

$$M(x, y) = \sum_{k=1}^{n} |x_k - y_k| \tag{2}$$

- Pearson Correlation—The statistical correlation between common ratings of two users to calculate the similarity is called Pearson correlation [8, 9]. A threshold, which is the minimum number of correlated items that is necessary for performing the computation, is set for computation of similarities. Equation is as follows:

$$P(x, y) = \frac{N\sum xy - \left(\sum x\right)\left(\sum y\right)}{\sqrt{\left[N\sum x^2 - \left(\sum x\right)^2\right]\left[N\sum y^2 - \left(\sum y\right)^2\right]}} \tag{3}$$

- Spearman Rank Correlation—Spearman Correlation uses the same computation as the Pearson method. The difference lies in the fact that the items are represented with respect to their ranking as opposed to their ratings. The rating ranges from a rank of one, given to higher ratings and for lower ratings the rank reduces.
- Cosine Similarity—It uses a vector-space approach, where each user is represented as a vector. The similarity is obtained by calculating the cosine distance between the rating vectors making use of their dot product [8].

$$sim = \cos\theta = \frac{AB}{|A||B|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}} \tag{4}$$

- Tanimoto Distance—It is a variation of the Jaccard Distance and is defined as ratio of intersecting values to the union of values [8]. This can be represented by the equation,

$$T_s(X, Y) = \frac{\sum_i X_i \bigwedge Y_i}{\sum_i X_i \bigvee Y_i} \tag{5}$$

The reason these measures were selected for this paper is because they represent different classes of calculating similarity, in essence, based on distance, based on correlation and based on vector. They are compared to identify the suitable measures for each kind of CF algorithms [11, 12].

The paper is organized as follows. Section 2 focuses on related work, followed by a description of proposed framework in Sect. 3. The recommender system built for the purpose of analysis is discussed in Sect. 4. Results and discussion of the analysis and conclusion in given in Sects. 5 and 6 respectively.

2 Related Work

Recommendation systems started off predominantly as offline systems rather than online real-time systems we use today. The capacity of computers to provide recommendations was recognized early in history of computing [7]. Elaine Rich, in his paper on User Modeling via Stereotypes [13], introduced a computer based librarian which grouped the books based on similarity of user input to hard coded information. Tapestry [14], by David Goldberg et al. proposed a model which would allow users to read any mail (by which it meant any online article), and rate them which

would further call for recommendations, close to how recommenders function today. Further, John T. Riedl et al. proposed Grouplens architecture [6, 15] for collaborative filtering. During this period, collaborative filtering and recommenders became a trend leading to a number of recommenders for various domains, like Ringo [16] for music, Amazon for e-commerce systems, Netflix for online streaming. The Netflix Prize awarded 1 million who could beat Netflix recommender by ten percent, and was won by BellKor's Pragmatic Chaos [1, 17], by implementing matrix factorization methods to improve the density of the dataset enabling the recommender to perform better. Recommender techniques include probabilistic models based on information retrieval, Bayesian inference [3, 7, 18], and case-based reasoning methods [19]. A survey on these models was done by Bobadilla et al. [20]. A switching hybrid recommender system can switch between recommended techniques by applying certain criterion. One such approach combined Naive Bayes classification approach with the collaborative filtering [21]. An experimental study was done by Michael D. Ekstrand et al. which let the users choose the algorithm for recommendation [22]. Advantages of improving the similarity measures in recommenders was shown by Sarwar et al. and Massa et al. who implemented Cosine similarity, adjusted cosine similarity, Pearson and Spearman Correlation for recommenders [23, 24]. Best comparison is done by Chandelier et al. who implemented various similarity as well as error measures to validate the prediction [25]. This paper aims at improving the accuracy of non-probabilistic collaborative filtering using a switching recommender system which uses sparsity of the dataset to switch between the algorithms. This framework follows up on Ekstrand et al. model [22], but by automating the choice of algorithm, we aim to make the system faster.

3 Proposed System

3.1 Overview

Current recommendation systems make use of a single type of similarity measure, or in case of collaborative filtering, either user based or item-based. However, this can be optimized for better accuracy. We see that the sparsity of the dataset plays a vital role in the accuracy of these methods. The proposed system provides a way to automatically select the various stages of recommendation based on the dataset.

In this section, we present a framework which is able to determine the type of CF and the similarity measure which can provide the best possible prediction for a particular dataset. The framework is designed to diagnose the dataset based on its sparsity, and is able to pick better algorithms for the given data.

The framework can be thought of as a black box, to which the input is the dataset, and the output is a list of recommendations (Fig. 1b).

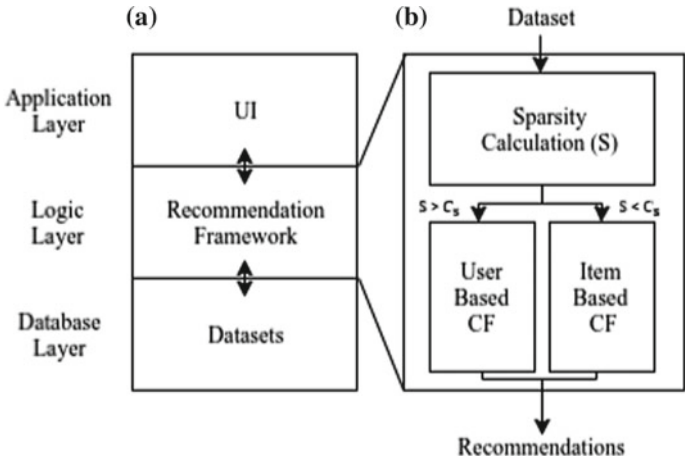

Fig. 1 **a** Three-tier architecture of the framework. The datasets are fed to the framework which outputs the results to the user. **b** Expansion of the framework

3.2 Layout

A dataset can be viewed in two different ways, a user-oriented view which is used by User-based CF and an item oriented view used by Item-based CF. The user oriented view looks like {*user*1: {*item*1: *rating*, *item*2: *rating*…}, *user*2: {*item*1: *rating*…}…}.

The item oriented view is {*item*1: {*user*1: *rating*…}, *item*2: {*user*1: *rating*… }… }.

The input to the framework is a user oriented view. In order to transform the dataset to item oriented format used by Item-based CF, a *transform Matrix* function is used.

Algorithm 1. The *transform Matrix* function.

```
begin
for user in user_ortn
      for item in user_ortn[user]
            item_ortn[item][user]= user_ortn[user][item]
return item_ortn
```

The backbone of the framework lies in the sparsity calculation. Sparsity is nothing but the population of the dataset. The algorithm for sparsity calculation is as follows.

Algorithm 2. The *calculate Sparsity* function.

```
begin
num_user = max_number_of_users
num_item = max_number_of_items
slots = num_user * num_item
for user in user_ortn:
      for item in user_ortn[user]:
            density = density + 1
      end for
end for
sparsity = density/slots
return sparsity
```

This function returns the sparsity S of the dataset. After a comparison of S with the sparsity coefficient C_S, the framework redirects the code to a User-based CF module or an Item-based CF Module. Sparsity Coefficient C_S is a specific percentage value, which if exceeded, the dataset can be said to be dense. Once the code is branched, the framework chooses the similarity measure which is suitable for the type of collaborative filtering being performed. The output of the framework is the list of possible recommendations.

4 Evaluation

The proposed framework is based upon the analysis done on two datasets: MovieLens and Jester. Two recommendation engines, one User-based and one Item-based, were built for the purpose.

4.1 Datasets

MovieLens was developed by the GroupLens project at the University of Minnesota [6, 26]. This data contains ratings on scale 1–5. Datasets used were as follows:

- *MovieLens Older Dataset Small* was a dataset of size 5 MB. It has 100,000 ratings from 1000 users on 1700 movies.
- *MovieLens* 1 *M Dataset* consists of 6 MB data holding about 1 million ratings from 6000 users on 4000 movies.
- *MovieLens Latest Dataset Small* of size 1 MB with 100,000 ratings applied to 9,000 movies by 700 users was used.

The Jester dataset is the dataset released from the Jester Online Joke Recommender System [27]. This data contains ratings on scale −10 to 10. Datasets used were,

- *Jester Dataset 1* is a dataset of size 3.9 MB which contains data from about 24,983 users.
- *Jester Dataset 2* is a dataset of size 3.6 MB which a collection of data about ratings of 23,500 users.
- *Jester Dataset 2+* is a dataset of size 5.1 MB. This dataset stores information of 500,000 ratings from 79,681 users.

The data from these datasets were loaded to the recommender system in the user oriented view. Based on the sparsity calculated by the sparsity algorithm (Algorithm 2), the datasets were classified as sparse or dense. Since the scale of rating of these datasets were different, they were normalized to the scale [−1, 1] during the evaluation.

4.2 System Specification

Analysis was performed on a 64-bit Ubuntu 16.04 LTS platform with a 4 GB memory, and a quad core Intel i5-5200U CPU with clock speed of 2.2 GHz.

4.3 Recommender Systems

Two recommender systems, one user-based and another item-based were written in Python language to compare the efficiency and accuracy of the similarity of efficiency measures [28]. The user-based recommender calculates the similarity between a person and other people and predicts the possible data of the person based on the most similar people. The item-based recommender calculates the similarity between two items based on the ratings received. Once the dataset was loaded to the recommender, the data was randomly split into training and test data. Prediction of possible rating of test data was made using the available training data which was further compared to actual data available. Lower error between the two data is a measure of better prediction accuracy. The random division of data for testing and training purpose was done again and the process is repeated, leading to calculation of average error value.

5 Results and Discussion

The calculated average error values are based on two accuracy metrics below.

5.1 Accuracy Metrics

The recommendations made have to be tested for accuracy as well as speed to understand which performs better in this particular dataset. The data was divided as 85% for training, and 15% for testing. The analysis is based on the validity or the accuracy of the results. Two measures of evaluating the accuracy of predictions are used to check the validity of the predictions [1, 3, 7].

- Root Mean Squared Error (RMSE)—The difference between predicted values and observed values (prediction error) when taken individually are called residuals. RMSE is the standard deviation of these residuals. It is a measure of how spread out these residuals are. It is a good error metric for numerical predictions [29]. Mathematically,

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(y_i - \hat{y}_i\right)^2} \tag{5}$$

- Mean Absolute Error (MAE)—The sum of the absolute value of the residuals is MAE. It indicates the margin of an error which can be expected from the forecast on an average [29]. It is formulated as,

$$MAE = \frac{1}{n}\sum_{i=1}^{n}\left|y_i - \hat{y}_i\right|^2 \tag{6}$$

5.2 Results

The sparsity of the dataset was calculated and the outcome is as represented in Table 1. The datasets were tested using the similarity measures mentioned above considering the User-based CF and Item-based CF (Figs. 2 and 3). The graphical representation of Figs. 2 and 3 is depicted in Figs. 4 and 5.

5.3 Discussion

In order to develop the proposed framework, two different genre of datasets were evaluated. Based on the sparsity of the datasets, following observations were made.

- Denser the dataset, higher the variation in the ratings and hence more possibility for error. This leads us to our first argument that efficiency of similarity measures is independent of the genre of the dataset and rather it only depends on the sparsity of the dataset.

Table 1 Classification of the datasets based on sparsity

Genre	Dataset	Volume	Sparsity	Dense/sparse
Movie	MovieLens 1 M	22384240	0.044684	Sparse
	MovieLens 100 k older	6023136	0.016574	Sparse
	MovieLens 100 k latest	1569152	0.063533	Sparse
Jokes	Jester dataset 1	6500	0.991385	Dense
	Jester dataset 2	6500	0.986769	Dense
	Jester dataset 2+	17640	0.955272	Dense

ITEM-BASED COLLABORATIVE FILTERING						
	SPARSE DATASETS					
SIMILARITY MEASURES	1M Dataset		100k Dataset Latest		100k Dataset Older	
	MAE	RMSE	MAE	RMSE	MAE	RMSE
Euclid Distance	0.2977	0.3436	0.3904	0.5064	0.5937	0.6894
Manhattan Distance	0.2978	0.3435	0.3905	0.5063	0.5938	0.6893
Pearson Correlation	0.4482	0.5146	0.6500	0.7986	0.8011	0.9431
Spearman Correlation	0.3185	0.3829	0.4061	0.4869	0.4579	0.5515
Cosine Similarity	0.3631	0.4190	0.4776	0.6122	0.5637	0.6840
Tanimoto Distance	0.7548	1.0070	0.6879	0.9147	0.7992	1.0549
	DENSE DATASETS					
SIMILARITY MEASURES	Jester Dataset 2+		Jester Dataset 1		Jester Dataset 2	
	MAE	RMSE	MAE	RMSE	MAE	RMSE
Euclid Distance	3.5855	4.7018	4.5583	5.6827	4.3113	5.4467
Manhattan Distance	3.8214	4.7928	4.8498	5.9823	4.3377	5.3231
Pearson Correlation	3.7232	4.8707	4.5674	5.7247	4.4863	5.5783
Spearman Correlation	4.7306	5.7467	4.9540	5.9838	4.8581	5.8347
Cosine Similarity	3.6502	4.8486	4.2797	5.4621	4.3655	5.4575
Tanimoto Distance	4.1897	5.3187	5.2042	5.8388	4.4991	5.4918

Fig. 2 MAE and RMSE values of different similarity measures for item-based CF

- Item-based algorithm calculates the similarity of the items based on the user's ratings. In a sparse matrix, it is feasible to calculate the relationship between the items rather than that between the users. Two users might rate two different sets of items making it impossible to calculate the similarity between them. Hence, Item-based CF has better accuracy for dense datasets.
- In a sparse dataset, since there are less ratings, the accuracy of calculation of the similarity between the users is less. Distance measures can be a better approach towards similarity calculation. Let's consider the MovieLens datasets, where the ratings are more mutually exclusive, which makes the distance calculation a viable approach. Hence, Euclidean Distance metric performs distinctively well (Fig. 3), making it a good choice for an Item-based recommender. In a dense matrix, since

USER-BASED COLLABORATIVE FILTERING						
	SPARSE DATASETS					
SIMILARITY MEASURES	1M Dataset		100k Dataset Latest		100k Dataset Older	
	MAE	RMSE	MAE	RMSE	MAE	RMSE
Euclid Distance	0.6671	0.8260	0.6064	0.7679	0.6996	0.8665
Manhattan Distance	0.7134	0.8833	0.6654	0.8400	0.7473	0.9240
Pearson Correlation	0.6748	0.8236	0.6970	0.8594	0.7019	0.8538
Spearman Correlation	0.7549	0.9232	0.7435	0.9139	0.7889	0.9626
Cosine Similarity	0.7861	0.9599	0.7783	0.9548	0.8297	1.0116
Tanimoto Distance	0.7844	0.9589	0.7851	0.9671	0.8356	1.0188
	DENSE DATASETS					
SIMILARITY MEASURES	Jester Dataset 2+		Jester Dataset 1		Jester Dataset 2	
	MAE	RMSE	MAE	RMSE	MAE	RMSE
Euclid Distance	4.0290	4.9961	4.6158	5.3930	4.1081	4.9523
Manhattan Distance	4.0640	5.0421	4.6837	5.4644	4.1350	4.9839
Pearson Correlation	3.8612	4.7787	4.2533	5.0416	3.8065	4.6107
Spearman Correlation	4.0111	4.9713	4.5724	5.3570	3.9642	4.7892
Cosine Similarity	3.8737	4.8127	4.0810	4.8777	3.7926	4.6388
Tanimoto Distance	4.1450	5.1252	4.8286	5.6070	4.1837	5.0346

Fig. 3 MAE and RMSE values of different similarity measures for user-based CF

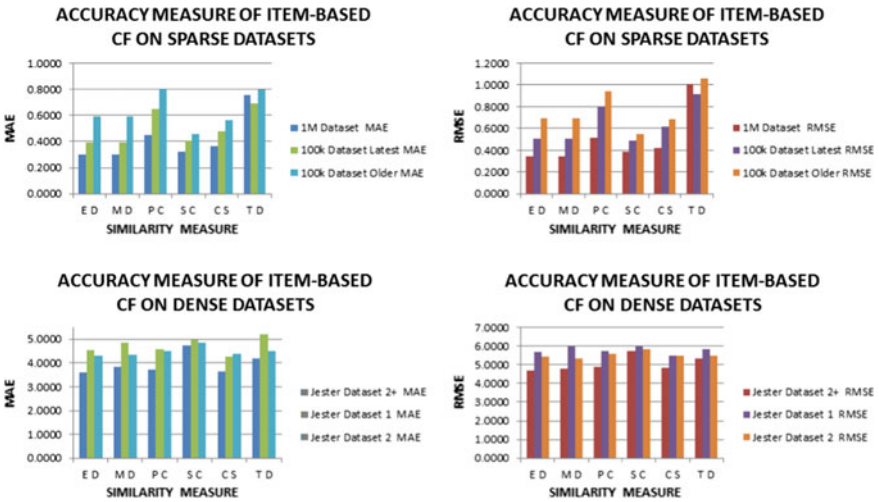

Fig. 4 Comparison of performance of similarity measures on the datasets in item-based CF

there are copious ratings, correlation and vector based approaches have a better accuracy. In Jester dataset, Pearson Correlation gives a low error (Fig. 4), making it appropriate for User-based recommender.

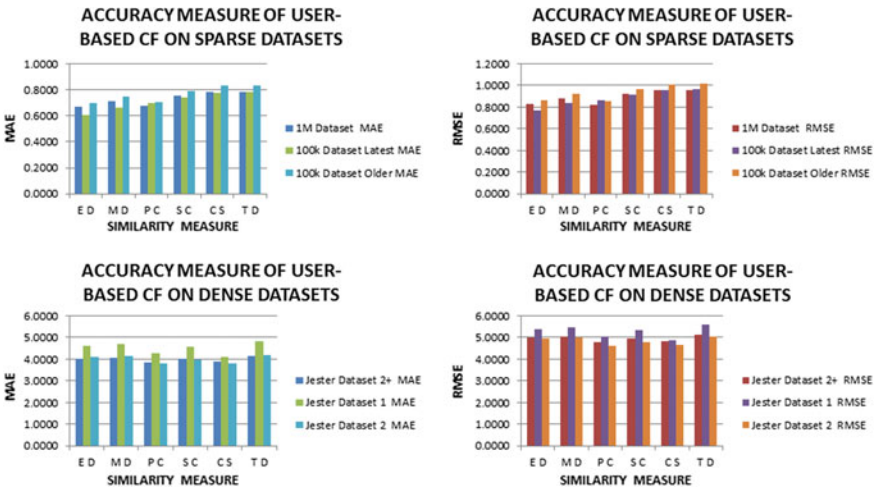

Fig. 5 Comparison of performance of similarity measures on the datasets in user-based CF

6 Conclusion

The proposed framework can be used as a recommendation system for any genre of dataset. The user can simply use the proposed system as a black box which takes any dataset as input and gives the best recommendation as output, making the user free from the complex logistics involved. The following conclusions formed the basis for the development of the framework. First, similarity measures depend only on the sparsity of the dataset. Second, User-based CF gives better accuracy for dense datasets, while Item-based CF gives better accuracy for sparse datasets. Finally, most suitable similarity measure for User-based CF was found to be Pearson correlation with an accuracy in the range of 91–94% for sparse datasets and in the range of 94–96% for dense datasets and Euclidean distance was found appropriate for Item-based CF with accuracy range of 99.4–99.7% for sparse datasets and 94–96% for dense datasets.

Acknowledgements The authors express their thanks to GroupLens and Jester team who compiled excellent datasets for research on recommender systems. We are grateful to the Principal of our college The National Institute of Engineering, Mysuru for encouraging us and providing the required facility.

References

1. Lü, L., Medo, M., Yeung, C.H., Zhang, Y.C., Zhang, Z.K., Zhou, T.: Recommender systems. Phys. Rep. **519**(1), 1–49 (2012)
2. Ullman, J.D.: Mining of Massive Datasets. Cambridge University Press, pp. 92–97 (2011)
3. Ricci, F., Rokach, L., Shapira, B.: Introduction to recommender systems handbook. Recommender Systems Handbook, pp. 1–35 (2011)
4. Adomavicius, G., Tuzhilin, A.: Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. IEEE Trans. Knowl. Data Eng. **17**(6), 734–749 (2005)
5. Burke, R.: Hybrid recommender systems: survey and experiments. User Model. User Adap. Interact. **12**(4), 331–370 (2002)
6. Herlocker, J.L., Konstan, J.A., Terveen, L.G., Riedl, J.T.: Evaluating collaborative filtering recommender systems. ACM Trans. Inf. Syst. (TOIS) **22**(1), 5–53 (2004)
7. Ekstrand, M.D., Riedl, J.T., Konstan, J.A.: Collaborative filtering recommender systems. Found. Trends® Human Comput. Interact. **4**(2), 81–173 (2011)
8. Amatriain, X., Pujol, J.M.: Data mining methods for recommender systems. Recommender Systems Handbook, pp. 227–262 (2015)
9. Shimodaira, H.: Similarity and recommender systems. School of Informatics, The University of Eidenburgh, 21 (2014)
10. How to build a recommendation system, Mickael Le Gal (2014)
11. Sondur, M.S.D., Chigadani, M.A.P., Nayak, S.: Similarity measures for recommender systems: a comparative study. J. Res. **2**(03) (2016)
12. Bagchi, S.: Performance and quality assessment of similarity measures in collaborative filtering using mahout. Procedia Comput. Sci. **50**, 229–234 (2015)
13. Rich, E.: User modeling via stereotypes. Cogn. Sci. **3**(4), 329–354 (1979)
14. Goldberg, D., Nichols, D., Oki, B.M., Terry, D.: Using collaborative filtering to weave an information tapestry. Commun. ACM **35**(12), 61–70 (1992)
15. Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., Riedl, J.: GroupLens: an open architecture for collaborative filtering of netnews. In: Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work, pp. 175–186 (1994)
16. Shardanand, U., Maes, P.: Social information filtering: algorithms for automating "word of mouth". In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 210–217 (1995)
17. Koren, Y.: The bellkor solution to the netflix grand prize. Netflix Prize Doc. **81**, 1–10 (2009)
18. Condli, M.K., Lewis, D.D., Madigan, D., Posse, C.: Bayesian mixed-E ects models for recommender systems. In: Proceedings of ACM SIGIR, vol. 99 (1999)
19. Smyth, B.: Case-based recommendation. The Adaptive Web, pp. 342–376 (2007)
20. Bobadilla, J., Ortega, F., Hernando, A., Gutiérrez, A.: Recommender systems survey. Knowl. Based Syst. **46**, 109–132 (2013)
21. Ghazanfar, M., Prugel-Bennett, A.: An improved switching hybrid recommender system using naive Bayes classifier and collaborative filtering (2010)
22. Ekstrand, M.D., Kluver, D., Harper, F.M., Konstan, J.A.: Letting users choose recommender algorithms: an experimental study. In: Proceedings of the 9th ACM Conference on Recommender Systems, pp. 11–18 (2015)
23. Sarwar, B., Karypis, G., Konstan, J., Riedl, J.: Item-based collaborative filtering recommendation algorithms. In: Proceedings of the 10th International Conference on World Wide Web, pp. 285–295 (2001)
24. Massa, P., Avesani, P.: Trust-aware collaborative filtering for recommender systems. CoopIS/DOA/ODBASE **1**(3290), 492–508 (2004)
25. Candillier, L., Meyer, F., Boullé, M.: Comparing state-of-the-art collaborative filtering systems. In: International Workshop on Machine Learning and Data Mining in Pattern Recognition, pp. 548–562 (2007)

26. MovieLens Datasets. https://grouplens.org/datasets/movielens/
27. Jester Datasets. http://eigentaste.berkeley.edu/dataset/
28. Toby, S.: Programming Collective Intelligence. O'Reilly Media, pp. 7–27 (2007)
29. Wit, J.: Evaluating recommender systems: an evaluation framework to predict user satisfaction for recommender systems in an electronic programme guide context, Master's thesis, University of Twente (2008)