Journal of Advances in Mathematics and Computer Science

35(2): 121-137, 2020; Article no.JAMCS.56828

ISSN: 2456-9968

(Past name: British Journal of Mathematics & Computer Science, Past ISSN: 2231-0851)

Recommender Systems: Algorithms, Evaluation and Limitations

Mubaraka Sani Ibrahim[1*] and Charles Isah Saidu[1]

[1]Department of Computer Science, Baze University, Abuja, Nigeria.

Authors' contributions

This work was carried out in collaboration between both authors. Authors MSI and CIS designed the study, managed the literature searches, wrote the protocol and performed the statistical analysis. Both authors read and approved the final manuscript.

Article Information

DOI: 10.9734/JAMCS/2020/v35i230254
Editor(s):
(1) Dr. Doina Bein, California State University, USA.
Reviewers:
(1) Yulia, Petra Christian University, Indonesia.
(2) Pasupuleti Venkata Siva Kumar, Vallurupalli Nageswara Rao Vignana Jyothi Institute of Engineering &Technology, India.
Complete Peer review History: http://www.sdiarticle4.com/review-history/56828

Review Article

Received: 28 February 2020
Accepted: 04 May 2020
Published: 14 May 2020

Abstract

Aims/ objectives: This paper presents the different types of recommender filtering techniques. The main objective of the study is to provide a review of classical methods used in recommender systems such as collaborative filtering, content-based filtering and hybrid filtering, highlighting the main advantages and limitations. This paper also discusses the state-of-art machine learning based recommendation models including Clustering models and Bayesian Classifiers. Further, we discuss the widespread application of recommender systems to a variety of areas such as e-learning and e-news. Finally, the paper evaluates the performance of matrix factorization-based models, nearest neighbours algorithms and co-clustering algorithms in terms of different metrics.

Keywords: Recommender system; collaborative filtering; recommendation; content filtering; evaluation.

1 Introduction

The advent of internet allows people from all around the world access to vast amount of information, there is a need to filter and deliver relevant information in order to alleviate the problem of information overload. This problem can be solved by information filtering systems such as recommender system.

**Corresponding author: E-mail: mubaraka.sani@bazeuniversity.edu.ng;*

Recommender systems are computer programs that predict preferences and recommend the best items to users in wide variety of areas. Examples include product recommendations to users on an e-commerce site such as Amazon [1]; content recommendation to users on social media platforms like Facebook [2] and Twitter [3]; learning content recommendation to improve learning outcome in e-learning systems such as AHA! [4]; content recommendations to users visiting a news site like GroupLens [5]and NewsWeeder [6]; music recommendation on email-based systems such as Ringo [7]; or provides movie recommendations on platforms like Netflix [8] to enhance user engagement and increase subscriptions.

Several studies discuss personalization through recommender systems. Imran [9] personalized learning management system LMS by integrating a recommender system in already developed learning management system. Ibrahim [10] proposed a framework that is designed with the goal of providing personalization by automatically suggesting learning objects to learners with the aim of improving learning experience.

Recommender systems research has led to increased theoretical contribution as well as evolvement of enhanced algorithms. Due to their importance in practice in the current data, recommender systems have been gaining remarkable attention to industry, academic community and research-community. In this paper, we review various approaches to building recommender system namely the collaborative filtering, content-based filtering, hybrid and knowledge-based recommender systems. By outlining their similarities and differences in terms of their computational and space complexities we show their limitations as well as suitable application areas.

Therefore, this paper is structured as follows, section 2 explains recommender filtering techniques, section 3 highlights the applications of recommender systems, section 4 discussed our evaluation process and results. Conclusions and future work are given in section 5.

2 Recommendation Filtering Techniques

Recommender filtering techniques are classified according to approach used to make prediction and the provision of relevant information to individual users'. There are different recommendation filtering techniques in literature namely collaborative filtering [11], content-based filtering [12], knowledge-based filtering and hybrid filtering algorithm. Fig. 1 illustrates different filtering techniques.

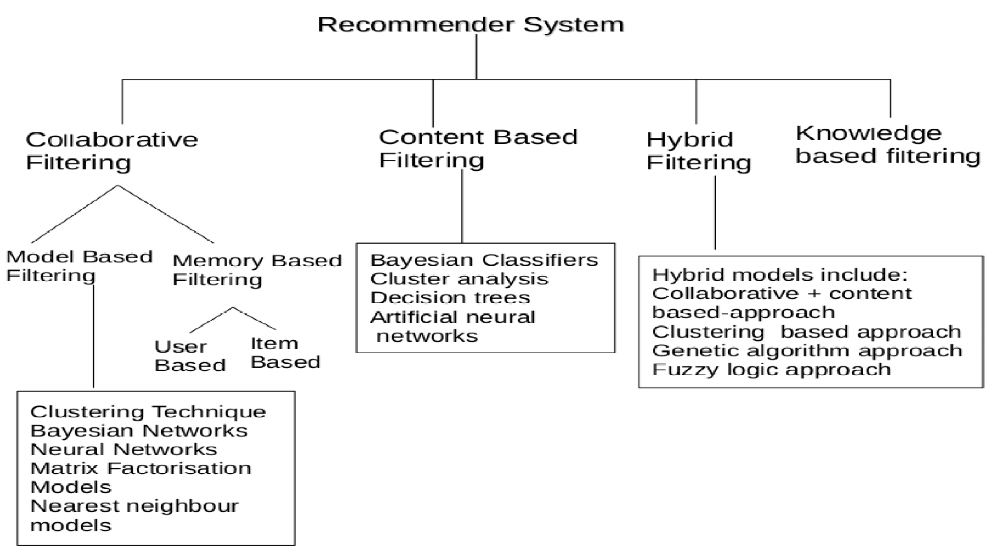

Fig. 1. Recommender filtering technique

2.1 Notation

Going forward, we define in Table 1 a set of notations used to describe algorithms and similarity measures.

Table 1. Table of notation

Symbol	Description
m	Maximum number of items
n	Maximum number of users
r_{ui}	Rating of user u for item i
r_{vi}	Rating of user v for item i
$\overline{r_u}$	Average rating for user u
$\overline{r_v}$	Average rating for user v
$\hat{r}_{ui}$	Predicted rating of user u for item i
I_u	Set of items rated by user u
I_v	Set of items rated by user v
I_{uv}	Set of items rated by user u and user v

2.2 Content-based filtering

Content-based filtering CB recommends items that are similar to the ones that the user liked in the past [13]. Lu et al. [14] outlined the basic principles of content-based technique. According to Lu, this approach analyzes user preference to determine attributes associated with items. Then user profile is matched against the attributes of the items and similar items are recommended to the user.

The users profile is based on a weighted vector of item attributes. The weights denote the importance of each attribute to the user and can be computed using a variety of techniques such as Bayesian classifiers, cluster analysis, decision trees, and artificial neural networks [15]. Limitations of CB include serendipity problems, new user problem and limited content analysis problem [13]. This technique is mostly applicable to areas where detailed attribute information is available like movies and music recommendation.

2.3 Collaborative filtering

Collaborative filtering *CF* deals with a set of users U_u where $u \in \{1, 2, ..., n\}$, n is the maximum number of users and $n \in Z$. Each user, U_u has associated with it a profile of ratings for items I_i where $i \in \{1, 2, ..., m\}$, m is the maximum number of items and $m \in Z$. Users rate item in a discrete range of possible values $r_{ui}= \{1, 2, ..., k\}$ where $k \in Z$. represents users' level of satisfaction with an item. For example, the range of possible values could be $\{1, ..., 5\}$ where 1 represents user's dissatisfaction and 5 represents user's satisfaction with an item.

Collaborative filtering relies on the preference of users with similar interest to make future recommendations using similarity metrics (see section. 2.3.1). User preference is inferred by matching user's data with $n \times m$ rating matrix which stores user preference/rating for each item as shown in Table 2.

This method relies on two types of ratings: explicit rating and implicit rating.

i. Explicit rating depends on feedback from users to make recommendation to the target user.
ii. Implicit rating is a rating that depend on user activities such as number of clicks and time spent on content.

Table 2. Rating matrix

	Item 1	Item 2	Item 3	…	Item m
User 1	r_{11}	r_{12}	?	…	r_{1m}
User 2	r_{21}	r_{22}	r_{23}	…	r_{2m}
User 3	r_{31}	?	r_{33}	…	r_{3m}
…	…	…	…	…	…
User n	r_{n1}	r_{n2}	r_{n3}	…	r_{nm}

The main advantage of this approach is that it is capable of recommending items without relying on domain knowledge about content. This algorithm requires large amount of information to make accurate prediction, utilizing small amount of information leads to the cold start problem. Common problems of this method includes cold start, scalability and data sparsity. Collaborative filtering can be categorized based on the way user rating is processed and contents recommended to the target user [16]. The sub-categories of collaborative filtering are:

i. Memory-based Filtering Algorithms
ii. Model-based Filtering Algorithms

2.3.1 Memory-based collaborative filtering

There are generally two types of memory-based filtering algorithm, user-based and item-based collaborative filtering.

i. User-based collaborative filtering identifies users that share similar interest with the target user and calculate predicted preference for the target user [17].
ii. Item-based collaborative filtering [18] finds a correlation between all pairs of items.

Different similarity measures are used to predict item/users similar to the target user. These measures process user ratings of items to generate recommendation to the target user. Examples are Pearson correlation similarity, Cosine similarity, Euclidean distance, Manhattan distance, Minkowski distance and Jaccard similarity.

i. Pearson correlation similarity [19]: Is a measure of the linear correlation between two users' *u* and *v* by considering commonly rated items. This method evaluates to a value between -1 and 1[20,21]. The value 1 represents a total positive linear correlation, *0* is no linear correlation, and *-1* is total negative linear correlation. Pearson similarity between the two users is defined in equation. 2.1

$$P_{SIM}(u,v) = \frac{\sum_{i \in I_{uv}} (r_{u,i} - \overline{r_u})(r_{v,i} - \overline{r_v})}{\sqrt{\sum_{i \in I_{uv}} (r_{u,i} - \overline{r_u})^2} \sqrt{\sum_{i \in I_{uv}} (r_{v,i} - \overline{r_v})^2}} \tag{2.1}$$

where $\overline{r_u}$ and $\overline{r_v}$ represents average rating of user *u* and user *v* .

ii. Cosine similarity: This method measures the similarities between two users' *u* and *v* by considering users as vectors of item ratings and taking the angle between each rating vectors, hence measuring the cosine between the vectors of two users [16]. Cosine similarity is an efficient metric especially for sparse vectors. The cosine similarity between two users is defined in equation. 2.2

$$Co\sin e_{SIM}(u,v) = \frac{\sum_{i \in I_{uv}} (r_{ui})(r_{vi})}{\sqrt{\sum_{i \in I_{uv}} (r_{ui})^2} \sqrt{\sum_{i \in I_{uv}} (r_{vi})^2}} \tag{2.2}$$

iii. Jaccard Similarity: Measures the similarity/overlap between two finite set (in this recommender systems context, similarity/overlap of two users' ratings). This measure is defined as the size of the intersection between two finite set divided by the size of union between the sets as in equation. 2.3. The range of Jaccard similarity value is between *0* and *1* with *0* indicating no overlap and *1* indicating complete overlap.

$$Jaccard_{SIM}(u,v) = \frac{|I_u \cap I_v|}{|I_u \cup I_v|} \tag{2.3}$$

iv. Mean squared difference: this metric is used to compute the mean squared difference similarity between all pairs of users or items.

Mean squared difference *MSD* for each user *u* is calculated as follows as in equation. 2.4

$$MSD_u = \frac{1}{m}\sum_{i=1}^{m}(r_{ui} - r_{vi})^2 \tag{2.4}$$

2.3.2 Model-based collaborative filtering

Model-based collaborative filtering develops a model using machine learning algorithms to predict user ratings. Model-based algorithms include clustering models, Bayesian networks, latent semantic models, deep learning and Markov decision process MDP [22]. Commonly used model-based algorithms are discussed below.

i. Clustering collaborative-filtering CF group users with similar interest into clusters. Users in the same group share similar interest or ratings while users in different groups have dissimilar interests. Similarity is determined using a distance measure, such as ones reviewed in section 2.3.1. Clustering is employed to improve efficiency because the number of operations is reduced [13].

There are many clustering algorithms such as k-mean, density-based clustering and hierarchical clustering.

A k-means clustering algorithm [13] works as follows:

(a) The algorithm works by randomly selecting *k* centroids.
(b) It partitions *N* items into number of clusters whose centroid is close to them.
(c) Then the given distance measure is used to measure distance between items. The closer the distance is between items the more similar they are.
(d) The cluster centroid is adjusted to account for the items whose cluster has changed.
(e) The iterative process continues until there are no further items that change their cluster membership. *k-means* is a simple and efficient algorithm however it assumes problems when clusters are of differing sizes and densities [13].

Co-clustering approach is similar to the correlation and clustering-based techniques for the reason that neighbourhoods' are employed for prediction [23]. Co-clustering algorithm [24] is a technique which allows simultaneous clustering of the rows and columns of a matrix. This algorithm generates prediction based on the average ratings of the user-item neighbourhood.

The prediction $\hat{r}_{ui}$ is given as:

$$\hat{r}_{ui} = \overline{C_{ui}} + (\mu_u - \overline{C_u}) + (\mu_i - \overline{C_i}) \tag{2.5}$$

where $\overline{C_{ui}}$, $\overline{C_u}$, $\overline{C_i}$, μ_u and μ_i denoted as average rating of co-cluster C_{ui}, average rating of users cluster C_u, the average rating of items cluster C_i, mean of all ratings given by user *u,* mean of all ratings given by item *i* respectively.

ii. Matrix Factorization models factorize a user-item rating matrix into product of two lower dimensionality matrices [25]. The row or column associated to a specific user or item is referred to as latent factors. This method relies on latent factors that serve a major purpose in recommendation. Matrix factorization has the advantage of utilizing implicit feedback by analyzing user behavior such as purchase history or search patterns to make predictions. Algorithms such as Singular value decomposition SVD, SVDpp and Non-Negative Matrix Factorization NMF derived from SVD are used for recommendation purposes. For matrix factorization expected rating $\hat{r}_{ui}$ can be expressed as: $\hat{r}_{ui} = q_i^T p_u$

(a) Singular value decomposition *SVD* [13,25,26] is matrix factorization technique that is generally used to identify latent semantic factors in information retrieval [25]. SVD in collaborative filtering requires factoring user-item ratings matrix. In SVD each item *i* is associated with an item-factor vector q_i and user *u* is associated with a user-factor vector p_u. These factors q_i and p_u can be found in such a way that the square error difference between their dot product and the known rating in the user-item matrix is minimum using equation. 2.6 [25,26].

$$\min_{q^*p^*} \sum_{(u,i)\in k} (r_{ui} - q_i^T p_u)^2 \tag{2.6}$$

To learn the factors q_i and p_u, a regularized term is added to avoid overfitting the factored matrix to the original rating matrix as in equation. 2.7 [25], where *k* is the set of the user-item pairs *(u, i)* for which r_{ui} is known.

$$\min_{q^*p^*} \sum_{(u,i)\in k} (r_{ui} - q_i^T p_u)^2 + \lambda(\|q_i\|^2 + \|p_u\|^2) \tag{2.7}$$

(b) SVDpp algorithm [13] is a derivative of SVD that considers implicit ratings. Although SVD is known for its accuracy and scalability, SVDpp is known to offer better accuracy than SVD [26]. The predicted rating $\hat{r}_{ui}$ is given in equation. 2.8:

$$\hat{r}_{ui} = \mu + b_u + b_i + q_i^T + (p_u + |I_u|^{\frac{-1}{2}} \sum_{j\in I_u} y_j) \tag{2.8}$$

where p_u, q_i, y_j, μ, b_u and b_i represents user factors, item factors, implicit item factors, mean of all known ratings, user biases and item biases respectively.

(c) Non-negative Matrix Factorization NMF: is a collaborative filtering algorithm based on NMF [27]. The prediction $\hat{r}_{ui}$ is expressed as $\hat{r}_{ui} = q_i^T p_u$ where user and item factors are kept positive. The optimization procedure is a regularized stochastic gradient descent SGD employed with a specific choice of step size that ensures non-negativity of factors. At each step of these SGD procedure, the factors *f*or user *u* and item *i* are updated as follows:

$$p_{uf} \leftarrow p_{uf} \cdot \frac{\sum_{i \in I_u} q_{if} \cdot r_{ui}}{\sum_{i \in I_u} q_{if} \cdot \hat{r}_{ui} + \lambda_u |I_u| P_{uf}} \tag{2.9}$$

$$q_{if} \leftarrow q_{if} \cdot \frac{\sum_{u \in U_i} p_{uf} \cdot r_{ui}}{\sum_{u \in U_i} p_{uf} \cdot \hat{r}_{ui} + \lambda_i |U_i| q_{if}} \tag{2.10}$$

where λ_i and λ_u are regularization parameters.

iii. Deep Artificial Neural Network DeepNets: Deep artificial neural networks consist of stacked inter-connected nodes and weighted links modeled after the human brain and designed to recognize patterns. Within a short span of years, increased interest in deep learning techniques have started to dominate algorithmic research in recommender systems. Different applications of deep learning techniques are explained in section 3.

iv. Nearest Neighbour models: Most common collaborative filtering methods are based on neighbourhood models. Koren [26] discussed two approaches, user-oriented and item-oriented methods. In user-oriented approach, ratings are predicted based on recorded ratings of similar minded users. Item-oriented approach, a rating is estimated using known ratings made by the same user on similar items.

Common similarity measure such as Pearson correlation is used to compute similarity between items. Algorithms such as KNNWithMeans and KNNWithZScore are derived from a basic nearest neighbour approach.

(a) KNNBasic is a basic collaborative filtering algorithm. The prediction $\hat{r}_{ui}$ is given as:

$$\hat{r}_{ui} = \frac{\sum_{v \in N_i^k(u)} sim(u,v) . r_{vi}}{\sum_{v \in N_i^k(u)} sim(u,v)} \tag{2.11}$$

or

$$\hat{r}_{ui} = \frac{\sum_{j \in N_u^k(i)} sim(i,j) . r_{uj}}{\sum_{j \in N_u^k(i)} sim(i,j)} \tag{2.12}$$

where $N_i^k(u)$ and $N_u^k(i)$ represents the *k* nearest neighbour of user *u* that have rated item *i* and the *k* nearest neighbors of item *i* that are rated by user *u* respectively.

(b) KNNWithMeans is basic collaborative filtering algorithm that considers the mean ratings of each user. The prediction $\hat{r}_{ui}$ is given as:

$$\hat{r}_{ui} = \mu_u + \frac{\sum_{v \in N_i^k(u)} sim(u,v).(r_{vi} - \mu_v)}{\sum_{v \in N_i^k(u)} sim(u,v)} \tag{2.13}$$

or

$$\hat{r}_{ui} = \mu_i + \frac{\sum_{j \in N_u^k(i)} sim(i,j).(r_{uj} - \mu_j)}{\sum_{j \in N_u^k(i)} sim(i,j)} \tag{2.14}$$

where μ_u and μ_i represents the mean of all ratings given by user *u* and the mean of all ratings given to item *i*.

(c) KNNWith Z Score is a collaborative filtering algorithm that takes into account the z-score normalization of each user. The prediction $\hat{r}_{ui}$ is given as:

$$\hat{r}_{ui} = \mu_u + \sigma_u \frac{\sum_{v \in N_i^k(u)} sim(u,v).(r_{vi} - \mu_v) / \sigma_v}{\sum_{v \in N_i^k(u)} sim(u,v)} \tag{2.15}$$

or

$$\hat{r}_{ui} = \mu_i + \sigma_i \frac{\sum_{j \in N_u^k(i)} sim(i,j).(r_{uj} - \mu_j) / \sigma_j}{\sum_{j \in N_u^k(i)} sim(i,j)} \tag{2.16}$$

where σ_u, σ_v represents the standard deviation of all ratings given by user *u* and the standard deviation of all ratings given to item *i* respectively.

(d) KNN Baseline is a basic collaborative filtering algorithm taking into account a baseline rating. The prediction $\hat{r}_{ui}$ is given as:

$$\hat{r}_{ui} = b_{ui} + \frac{\sum_{v \in N_i^k(u)} sim(u,v).(r_{vi} - b_{vi})}{\sum_{v \in N_i^k(u)} sim(u,v)} \tag{2.17}$$

or

$$\hat{r}_{ui} = b_{ui} + \frac{\sum_{j \in N_u^k(i)} sim(i,j).(r_{uj} - b_{uj})}{\sum_{j \in N_u^k(i)} sim(i,j)} \tag{2.18}$$

where b_{ui} represents the baseline rating of user *u* for item *i*.

v. Bayesian Classifier: Most model-based recommender systems are based on Bayesian classifiers. Bayesian classifier is a probabilistic technique for solving classification problems. The Naive Bayes is a type of Bayesian classifier that is based on the Bayes theorem with the naïve assumption among variables [28]. Mooney [29] proposed a naive Bayesian text classifier for book recommendation with independent assumption among variables, for instance the occurrence words *w* in a document *D* is dependent on the document class *c* but independent of other words *w*. There are *T* different types of information about a books such as title, author, genre and synopsis. The posterior probability of each class c_j given a document *D* is computed using Bayes rule:

$$P(c_j \mid D) = \frac{P(c_j)}{P(D)} \prod_{i=1}^{|D|} P(w_i \mid c_j) \tag{2.19}$$

where $P(c_j)$ and $P(w_i | c_j)$ must be estimated from the training data.

The posterior category probabilities for a book, b_y can be calculated as follows:

$$P(c_j \mid b_y) = \frac{P(c_j)}{P(b_y)} \prod_{t=1}^{T} \prod_{i=1}^{|d_m|} P(w_{ti} \mid c_j, T_t) \tag{2.20}$$

where, $P(w_{ti} | c_j, T_t)$ is the probability of a word w_{ti} given a class c_j and type T_t. Hence, the parameters of the model are also estimated from the training data.

vi. Fuzzy Logic is suitable for modeling inexact data and incomplete knowledge. Fuzzy logic has been widely used in the design of a recommender system to handle the uncertainty, impreciseness and vagueness in item features as well as users behavior and preference [30]. Applications of fuzzy logic to recommender systems are highlighted in section 3.

2.4 Knowledge-based filtering

Knowledge-based filtering KBF [31,32] relies on explicit knowledge acquisition to provide recommendation. In this approach item domain knowledge is matched against user preference. The cold-start problem of collaborative filtering algorithm is eliminated because this type of recommendation does not rely on user's interaction with items. The major limitation faced by the KBF is the construction of the knowledge base [33], which usually requires considerable domain knowledge, and expertise in knowledge representation.

2.5 Hybrid filtering

Hybrid filtering combines the features of two or more traditional recommender algorithms to improve efficiency and enhance performance [34,35]. Hybrid approach can also be used to overcome limitations of recommender system such as cold start and sparsity problems. Burke [34] compares seven basic hybridization strategies: weighted, mixed, switching, feature combination, feature augmentation, cascade and meta-level.

2.6 Problems

Collaborative filtering and content-based filtering have several shortcomings.

i. Data Sparsity: The availability of rating data affects the performance of recommender system. There are situations where there is immense history of rating data, such situations are called warm-start scenarios. Collaborative filtering generally depends on other users to predict user ratings thereby perform better than content-based filtering in such scenarios. Content-based filtering requires extensive attribute information to predict user ratings therefore outperform collaborative filtering in cold-start situations where there is sparse past rating data.

ii. Scalability: Collaborative filtering performs better with large number of rating data however a massive increase in the number of users and items will require more computational resources.

iii. Grey sheep: This collaborative filtering problem refers to users with inconsistent preferences [37]. Such users can hardly benefit from collaborative filtering because their interest rarely match other users. Another challenge is a black sheep problem which refers to users with unique preferences thereby recommendation for such users is usually a difficult task.

iv. Serendipity: Content-based systems typically recommend items that match similar ones that a user preferred in the past. Therefore, content-based systems are faced with over-specialization or serendipity problem [13] since they recommend items that are too similar thereby hindering the chance of discovering unexpected new interests.

v. Limited content problem: Content-based systems depend on attribute information to provide recommendations. These systems have a natural limit in the number and type of features that they can capture [13], sometimes additional information is needed to provide suitable recommendation.

3 Applications of Recommender System

This section highlights the widespread application of recommender systems.

i. Clustering Algorithms: Clustering is widely applied in recommender to improve efficiency and solve problems. Wasid [36] proposed a technique that incorporates multi-criteria clustering approach into traditional recommender systems to improve efficiency. A clustering algorithm was introduced to identify and resolve the grey-sheep users problem [37] and deal with cold start problem of recommender system [38].

ii. Bayesian Classifier: Bayesian classifiers have been employed in recommender systems to improve performance. For example Ghazanfar [28] proposed a framework that combines a Naive Bayes classifier with collaborative filtering to provide better performance in terms of coverage and accuracy.

Lee et al. [39] proposed an ontology-based product recommender system that combines Bayesian belief network and ontology to generate recommendations to individual customers.

Zimmerman et al. [40] provided a TV program recommender system, which combines Bayesian classifier, artificial neural networks and decision tree to generate recommendations.

Google News [41] is a personalized news recommender system that predict news preference of individual user by developing a Bayesian model from genuine news interest of users and the current news trend. NewsDude [42] is a news recommender system that uses a machine learning approach to model user's short term and long term interests. NewsDude employs nearest neighbor algorithm to model short-term interests and naive Bayesian classifier to model long-term interest.

iii. Artificial Neural Networks: Recent advances in deep learning based recommender systems have gained increasing attention by overcoming the limitations of conventional models and improving recommendation quality. For example, Lin [43] compared two recent neural approaches to baseline neural methods in the field of information retrieval.

Zhang et al. [44] al highlights the state-of-art categories of deep learning based recommendation models and classified the models into recommendation with neural building blocks and recommendation with deep hybrid model based on the deep learning technique employed. Sheikh et al. [45] introduced deep learning based content-collaborative methodology for personalized size and fit recommendation to alleviate the sparsity problem of collaborative filtering. This method incorporates a split-input neural network architecture with global and entity-level embedding parameter. The global parameters enable the model to capture information relevant for predicting customer's size/fit and the entity-level embedding parameters allows the model to acquire implicit properties of individual customers and articles for personalized recommendations.

He [46] introduced neural network-based collaborative filtering that expresses matrix factorization by replacing the inner product with a neural architecture to learn an arbitrary function from data.

iv. Fuzzy Logic: Porcel [47] introduced a recommender system for research resources based on fuzzy linguistic modeling. The recommender system employs a multi-granular linguistic modeling to improve users filtering activity and generates useful recommendations to researchers in accordance with their research areas.

Chen et al. [48] discussed the applications of fuzzy item response theory in e-learning system. Chen developed a personalized intelligent tutoring system capable of recommending courseware with suitable difficulty levels for learners according to learners uncertain/fuzzy feedback responses.

v. Genetic Algorithm: Genetic algorithm GA is heuristic search algorithm that is inspired by Darwin's theory of natural evolution. Genetic algorithm has been widely applied to many applications in different RS. For example, Bobadila [49] presents a metric applicable to collaborative filtering recommendation to measure similarity between users using genetic algorithm. This metric was calculated via a simple linear combination of values and weights. Values are calculated for each pair of users between which the similarity is obtained, whilst weights are only calculated once, making use of a prior stage in which a genetic algorithm extracts weightings from the recommender system. The obtained result improves prediction quality, recommendation quality and overall performance.

Kim and Ahn [50] applied GA-based k-means clustering to online shopping market to improve segmentation for personalized recommender systems in comparison to other typical clustering algorithms.

Marung [51] proposed Top-N recommender system using visual-clustering methods based on genetic algorithm to address cold-start and sparsity problems. The methods presented are the hybrid between the visual-clustering recommendation and user-based methods, and the hybrid between the visual-clustering recommendation and item-based methods.

4 Evaluation Process

We used the MovieLens data set [52] for our evaluation. This is a popular data set used by researchers and developers in the field of recommender systems. This data set downloaded from the internet has been widely used in collaborative filtering research. This data set contains 100836 ratings created by 610 users between March 29, 1996 and September 24, 2018 across 9742 movies. We only considered users that have rated 10 or more movies. The rating distribution of the data set is shown in Fig. 2. We ran all our experiments on a Linux based PC with 4 Intel Core i3-5005U processor having a speed of 2.00 GHz and 7.7 GB of RAM.

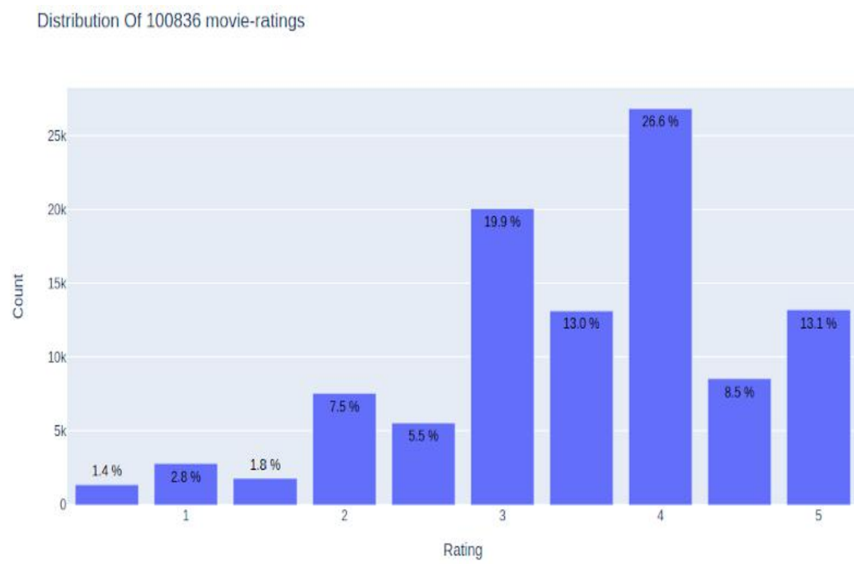

Fig. 2. Percentage of rating for MovieLens dataset

Algorithm	fit_time	test_mae	test_rmse	test_time
SVDpp	272.005514	0.643926	0.842530	10.474269
KNNBaseline	0.324139	0.647128	0.846371	4.486481
KNNWithZScore	0.198570	0.650938	0.856072	3.956285
SVD	4.343011	0.656671	0.856194	0.336155
KNNWithMeans	0.141537	0.657155	0.858871	3.572907
NMF	4.742258	0.680345	0.887622	0.278471
CoClustering	1.556771	0.705544	0.905774	0.268092
KNNBasic	0.127423	0.701924	0.912592	3.299703

Fig. 3. Comparative performance of different algorithms based on accuracy

Algorithm	fit_time	test_mae	test_rmse	test_time
KNNBaseline	0.268406	0.646759	0.845898	4.361257
KNNWithMeans	0.135328	0.656871	0.858200	3.405153
KNNWithZScore	0.204893	0.652606	0.858560	3.820586
KNNBasic	0.111292	0.701760	0.912736	3.039191

Fig. 4. Comparative performance of nearest neighbour models based on accuracy

Algorithm	fit_time	test_mae	test_rmse	test_time
SVDpp	272.295559	0.642863	0.840413	10.456716
SVD	4.518770	0.654791	0.852877	0.308611
NMF	4.639264	0.681363	0.887861	0.279209

Fig. 5. Comparative performance of matrix factorization models based on accuracy

4.1 Evaluation Method

Recommender systems research has used several types of measures for evaluating the quality of a recommender system. The most commonly used accuracy metrics for evaluating accuracy of recommender systems are mean absolute error (MAE) and root mean squared error RMSE. In this study, MAE and RMSE accuracy metrics are used for evaluation.

Mean absolute error measures the average of the absolute difference between the predicted and actual rating.

Mean absolute error *MAE* for each user is calculated as:

$$MAE = \frac{1}{m}\sum_{u=1}^{n} \sum_{i=1}^{m} \left| r_{ui} - \hat{r}_{ui} \right| \tag{4.1}$$

Root mean square error *RMSE* is the square root of the average of the difference between predicted and actual ratings.

$$\sqrt{\frac{1}{m}\sum_{u=1}^{n} \sum_{i=1}^{m} \left(r_{ui} - \hat{r}_{ui} \right)^2} \tag{4.2}$$

where r_{ui} and $\hat{r}_{ui}$ are the actual and predicted rating respectively.

However, accuracy metrics are not suitable for measuring ranking performance of recommender systems. Therefore, the study also conducted an evaluation based on ranking metrics Precision@k, Recall@k and F_1score.

	Algorithms	F1Score	Precision_at_k	Recall_at_k
0	KNNBaseline	0.716955	0.913683	0.589934
1	KNNBasic	0.722575	0.911444	0.598544
2	KNNWithMeans	0.705978	0.919640	0.572879
3	KNNWithZScore	0.705299	0.920494	0.571657
4	SVD	0.716650	0.917486	0.587949
5	SVDpp	0.714059	0.924333	0.581725
6	NMF	0.696143	0.919668	0.560029
7	CoClustering	0.695086	0.922929	0.557465

Fig. 6. Comparative performance based on ranking metrics

Precision@k is the fraction of recommended items that are relevant. Precision@k for each user is calculated as shown in equation. 4.3

$$\mathrm{Pr}\,ecision@k = \frac{\left|\{\mathrm{Re}\,commended\ items\ that\ are\ relevant\}\right|}{\left|\{\mathrm{Re}\,commended\ items\}\right|} \tag{4.3}$$

Recall@k is the fraction of relevant items that are recommended. The Recall@k score for each user is calculated as shown in equation 4.4.

$$\mathrm{Re}\,call@k = \frac{\left|\{\mathrm{Re}\,commended\ items\ that\ are\ relevant\}\right|}{\left|\{\mathrm{Re}\,levant\ items\}\right|} \tag{4.4}$$

F_1 score is the harmonic mean score of precision and recall. F1 score is calculated as shown in equation. 4.5 as follows:

$$F_1\ score = \frac{2(precision * recall)}{(precision + recall)} \tag{4.5}$$

4.2 Results

In this section we present the results of our evaluation. A set of algorithms representative of the different techniques found in the literature were considered for evaluation. In this work, we run a cross validation procedure for 8 algorithms reporting accuracy measures and computation times shown in Fig. 3. Algorithms such as KNN Baseline, KNN with Z Score, KNN with Means, KNN Basic, SVD, SVDpp, NMF and Co-clustering are all described in section 2. Regarding accuracy metrics differences among nearest neighbours models such as KNN Baseline, KNN with Z Score, KNN with Means, KNN Basic are shown in Fig. 4. In this group KNN Baseline presents the best result. In the matrix factorization models, SVD, SVDpp and NMF algorithms present different results. SVDpp present the best result in this group as shown in Fig. 5. SVDpp obtains the best overall result slightly better results KNN Baseline.

To perform evaluation based on ranking metrics the dataset is split into training data and test data, the algorithm uses 75% of the dataset as training data and 25% as test data to generate predicted ratings. The Precision@k and Recall@k is computed for each user and the average is computed over all users. The results of the evaluation in terms of Precision@k, Recall@k and F1 score for different algorithms are displayed in Fig. 6. In our evaluation KNNBasic presents the best result in F1 score and Recall@k whereas SVDpp model records the best Precision score.

5 Conclusions

In this paper, we review the different types of recommendation filtering methods focusing on model based algorithms namely: matrix factorization model, nearest neighbours model and clustering models. Widespread applications of recommender systems are highlighted such as the application to fuzzy logic and genetic algorithms. We evaluate 8 types of algorithms thereby showing the performance of different types of algorithm in terms of different metrics such as MAE, RMSE, Recall, Precision and F1 score. As a future work, we would like to apply models such as tensor factorization models to generate context-dependent recommendation. Furthermore, we would like to evaluate our algorithms on dataset of domains other than movies, such as Book-Crossing data set.

Competing Interests

Authors have declared that no competing interests exist.

References

[1] Linden JYG, Smith B. Amazon.com recommendations: Item-to-item collaborative. IEEE, Internet Computing. 2003;7:7680.

[2] Baatarjav E, Phithakkitnukoon S, Dantu R. Group recommendation system for Facebook. 2008;211-219.

[3] Goel A, Lin J, Sharma A, Wang D, Zadeh R. WTF: The who-to-follow system at Twitter. Proceedings of the 22nd International Conference on World Wide Web; 2013.

[4] De Bra P, Calvi L. AHA! An open adaptive hypermedia architecture. New Review of Hypermedia and Multimedia. 1998;4(1):115-139.

[5] Resnick P, Lacovou N, Suchak M, Bergstrom P, Riedl J. GroupLens: An open architecture for collaborative filtering of netnews. Proceedings of the 1994 ACM Conference. 1994;175-186.

[6] Lang K. Newsweeder: Learning to filter netnews. In Proceedings of the Twelfth International Conference on Machine Learning; 1995.

[7] Shardanand U, Maes P. Social information filtering: Algorithms for automating. In Proceedings of CHI 95. Denver, CO.; 1995.

[8] Netflix. [Online]
Available:http://www.netflix.com/

[9] Imran H, Hoang Q, Chang T, Graf S, Kinshuk D. A framework to provide personalization in learning management systems through a recommender system approach; 2014.

[10] Ibrahim MS, Hamada M. Adaptive learning framework. In IEEE 15th International Conference on Information Technology Based Higher Education and Training ITHET; 2016.

[11] Schafer J, Frankowski D, Herlocker J, Sen S. Collaborative filtering. In the Adaptive Web., Springer, Berlin, Heidelberg. 2007;4321.

[12] Pazzani M, Billsus D. Content-based recommendation systems. In the Adaptive Web. Lecture Notes in Computer Science. Springer, Berlin, Heidelberg. 2007;4321.

[13] Ricci F, Rokach L, Bracha Shapira B, Paul BK. Recommender systems handbook, 1st Ed. Springer. Berlin, Heidelberg. 2010;151-153.

[14] Lu J, Wu D, Mao M, Wang W, Zhang G. Recommender system application developments: A survey. Decision Support Systems. 2015;74.

[15] Blanda S. Online recommender systems-how does a website know what i want? 2015. [Online].
Available:https://blogs.ams.org/mathgradblog/2015/05/25/online-recommender-systems-website-want/
[Accessed 8 February 2020]

[16] Cacheda F, Carneiro V, Fernandez D, Formoso V. Comparison of collaborative filtering algorithms: Limitations of current techniques and proposals for scalable, high performance recommender systems. ACM Transactions, Web 5. 2011;1:Article 2.

[17] Ge F. User-based collaborative filtering recommendation algorithm based on Folksonomy smoothing. In Advances in Computer Science and Education Applications. Communications in Computer and Information Science. Springer, Berlin, Heidelberg. 2011;202.

[18] Sarwar B, Karypis G, Konstan J, Riedl J. Item-based collaborative filtering. In Recommendation Algorithms. Proceedings of ACM World Wide Web Conference; 2001.

[19] Allen R. User models: Theory, method, practice. International Journal Man-Machine Studies; 1990.

[20] Raghuwanshi S, Pateriya R. Recommendation systems: Techniques, challenges, application and evaluation. Soft Computing for Problem Solving SocProS. 2017;2.

[21] Desrosiers C, Karypis G. A comprehensive survey of neighborhood-based recommendation methods. In Recommender Systems Handbook, Springer, Boston, MA; 2011.

[22] Do MT, Nguyen VD, Nguyen L. Model-based approach for collaborative filtering. In 6th International Conference on Information Technology for Education; 2010.

[23] George T, Merugu S. A scalable collaborative filtering framework based on coclustering. In Fifth IEEE International Conference on Data Mining (ICDM'05); 2005.

[24] Banerjee A, Dhillon I, Ghosh J, Merugu S, Modha D. A generalized maximum entropy approach to bregman co-clustering and matrix approximation. In Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining; 2004.

[25] Koren Y, Bell R, Volinsky C. Matrix factorization techniques for recommender systems. Computer. 2009;42(8):30-37.

[26] Koren Y. Factorization meets the neighborhood: A multifaceted collaborative filtering model. In 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'08); 2008.

[27] Luo X, Zhou M, Xia Y, Zhu Q. An efficient non-negative matrix-factorization based approach to collaborative filtering for recommender systems. In IEEE Transactions on Industrial Informatics; 2014.

[28] Ghazanfar MA, Prugel-Bennett A. An improved switching hybrid recommender system using Naive Bayes classifier and collaborative filtering. In International MultiConference of Engineers and Computer Scientists IMECS. 2010;1.

[29] Mooney RJ, Roy L. Content-based book recommending using learning for text categorization. In DL-00, 5th ACM Conference on Digital Libraries, San Antonio, US.

[30] Jain A, Gupta C. Fuzzy logic in recommender systems. In Fuzzy Logic Augmentation of Neural and Optimization Algorithms, Springer. 2018;255-273.

[31] Aggarwal C. Recommender systems: The textbook, 1st Ed. Springer, Switzerland; 2016.

[32] Burke R. Knowledge-based recommender systems. Encyclopedia of Library and Information Science. 2000;69(32):180-200.

[33] Bouraga S, Jureta I, Faulkner S, Herssens C. Knowledge-based recommendation systems: A survey. International Journal of Intelligent Information Technologies. 2014;10:1-19.

[34] Burke R. Hybrid web recommender systems. In the Adaptive Web, Springer, Berlin Heidelberg. 2007;377-408.

[35] Cano E. Hybrid recommender systems: A systematic literature review. Intelligent Data Analysis. 2017;21:1487-1524.

[36] Wasid M, Ali R. An improved recommender system based on multi-criteria clustering approach. In 8th International Congress of Information and Communication Technology (ICICT- 2018); 2018.

[37] Ghazanfar M, Prugel-Bennett A. Leveraging clustering approaches to solve the gray-sheep users problem in recommender systems. Expert Systems with Applications. 2014;41(7):3261-3275.

[38] Shinde SKU. Hybrid personalized recommender system using centering-bunching based clustering algorithm. Expert Systems with Applications. International Journal. 2012;39:1381-1387.

[39] Lee T, Jonghoon C, Shim J, Lee S. An ontology-based product recommender system for B2B market places. International Journal of Electronic Commerce. 2006;11:125-155.

[40] Zimmerman J, Kauapati K, Buczak AL, Schaffer D, Gutta S, Martino J. TV personalization system. In Personalized Digital Television. Human-Computer Interaction Series, Springer, Dordrecht. 2004;6.

[41] Liu J, Dolan P, Pedersen E. Personalized news recommendation based on click behavior. International Conference on Intelligent User Interfaces Proceedings IUI, in Intelligent User Interfaces, Proceedings IUI; 2010.

[42] Billsus D, Pazzani M. Hybrid user model for news story classification. In Seventh International Conference on User Modeling, Banff, Canada; 1999.

[43] Lin J. The neural hype and comparisons against weak baselines. ACM SIGIR Forum. 2019;52:40-51.

[44] Zhang S, et al. Deep learning based recommender system: A survey and new perspectives. ACM Computing Surveys. 2018;52(1).

[45] Sheikh AGR, Koriagin E, Ho Y, Shirvany R, Vollgraf R, et al. A deep learning system for predicting size and fit in fashion e-commerce. In 13th ACM Conference on Recommender Systems; 2019.

[46] He X, Liao L, Zhang H, Nie L, Hu X, Chua T. Neural collaborative filtering. In 26th International Conference on World Wide Web; 2017.

[47] Porcel C, Lopez-Herrera A, Herrera-Viedma E. Recommender system for research resources based on fuzzy linguistic modeling expert systems with applications. Expert Systems with Applications. 2019;36:5173-5183.

[48] Chen C, Duh L. Personalized web-based tutoring system based on fuzzy item response theory. Expert Systems with Applications. 2009;34:2298-2315.

[49] Bobadilla J, Ortega FH, Javier A. Improving collaborative filtering recommender system results and performance using genetic algorithms. Knowledge-Based Systems. 2011;24(8):1310-1316.

[50] Kim K, Ahn H. Using a clustering genetic algorithm to support customer segmentation for personalized recommender systems. In Artificial Intelligence and Simulation. AIS 2004. Lecture Notes in Computer Science, Springer, Berlin Heidelberg. 2005;3397.

[51] Marung U, Theera-Umpon N, Auephanwiriyakul S. Top-N recommender systems using genetic algorithm-based visual-clustering methods. Symmetry. 2016;8:54.

[52] Maxwell Harper F, Konstan JA. The MovieLens datasets: History and context. 2015;5(4).

Peer-review history:
The peer review history for this paper can be accessed here (Please copy paste the total link in your browser address bar)
http://www.sdiarticle4.com/review-history/56828