Computers and Industrial Engineering, Volume 130, April 2019, pp. 187-197
DOI:10.1016/j.cie.2019.02.028

An integrated recommender system for improved accuracy and aggregate diversity

Sujoy Bag[a], Abhijeet Ghadge[b]*, Manoj Kumar Tiwari[a]

[a]*Department of Industrial and Systems Engineering, Indian Institute of Technology, Kharagpur, India*
[b]*Centre for Logistics and Supply Chain Management, Cranfield School of Management, Cranfield University, UK*

Abstract

Information explosion creates dilemma in finding preferred products from the digital marketplaces. Thus, it is challenging for online companies to develop an efficient recommender system for large portfolio of products. The aim of this research is to develop an integrated recommender system model for online companies, with the ability of providing personalized services to their customers. The K-nearest neighbors (KNN) algorithm uses similarity matrices for performing the recommendation system; however, multiple drawbacks associated with the conventional KNN algorithm have been identified. Thus, an algorithm considering weight metric is used to select only significant nearest neighbors (SNN). Using secondary dataset on MovieLens and combining four types of prediction models, the study develops an integrated recommender system model to identify SNN and predict accurate personalized recommendations at lower computation cost. A timestamp used in the integrated model improves the performance of the personalized recommender system. The research contributes to behavioral analytics and recommender system literature by providing an integrated decision-making model for improved accuracy and aggregate diversity. The proposed prediction model helps to improve the profitability of online companies by selling diverse and preferred portfolio of products to their customers.

Keywords: Recommender system, Behavioral analytics, Extreme learning, Aggregate diversity, E-business; Decision support system

1. Introduction

As of January 2018, Amazon, a widely recognized e-commerce company has over 562.4 million products on online market (Scrapehero, 2018). Recently, Walmart, another US-based e-retailing company acquired Flipkart for $16 billion (Economictimes, 2018), to compete with Amazon; creating a continuous growth in offered products and available platforms in the recent times. Along with quantity, such online companies are trying to improve the quality of products and services provided. The increase in the portfolios of products and the availability of multiple online platforms have complicated the decision-making process for customers. Thus, in the digital era, it is very rewarding to predict customer's purchasing intention and provide relevant product recommendations (Bag et al., 2019). Recently, Smith and Linden published a report on how to enhance productivity in terms of sales, product search and visibility by using a recommendation system (Smith and Linden, 2017). According to this report, YouTube has introduced a recommendation system in 2010 to increase users' search throughput by suggesting preferred videos based on past search patterns. Since then the recommender system has been widely used in the online business (Andjelkovic et al., 2019; Nilashi et al., 2017; Park, Oh, & Yu, 2017). Amazon and Netflix significantly benefit from the recommendation system. It is estimated that customers view 30% of Amazon's pages and watch 80% of Netflix movies following recommendations provided by Amazon and Netflix respectively. Three groups of stakeholders namely; consumers, retailers, and product companies get 'win-win' profit from the recommendation system.

There are two types of recommender systems prominently used in the literature; content-based recommendation (Besbes et al., 2015; Son and Kim, 2017) and collaborative filtering (Karabadji et al., 2018). Content-based recommendation stores the history of the product that a particular user has liked in the past and then, builds a user model to recommend a similar type of product that the user is most likely to prefer in the future. For example, YouTube or Netflix provide video/movie recommendations to their users based on their past viewing activity. Similarly, online shopping websites capture behavioral data such as the list of products that the user is browsing through and recommend them with the product(s) that they are most likely to buy. On the other hand, collaborative filtering recommend a customer from the history of the targeted customer and customers who have similar tastes with that of selected customer (Burke et al., 2015).

Rapid growth of product variety and user multiplicity in the present digital world has become a challenge for developing efficient algorithms for recommender systems (Scrapehero, 2018). Accordingly, the user-item rating metric has become sparse, as customers are unwilling to provide feedback on various products; thus, hindering the ability to capture behavioral data. Estimating users' interest from the limited information is a challenging task. To overcome this problem, several techniques such as mixed similarity learning and clustering are applied on the implicit data (Liu et al., 2017; Najafabadi et al., 2017). Moreover, a formal probabilistic framework (Markov random fields) and a heuristic similarity measure (PIP) exists in the literature to improve accuracy of collaborative filtering from explicit sparse data (Ahn, 2008; Patra et al., 2015; Tran et al., 2016). The computational complexity and the implementation costs of these techniques are high compared to traditional similarity approaches. Also, aggregate diversity has been ignored in the past studies. Owing to data sparsity, the traditional similarity model uses a number of nearest neighbors to accurately predict the rating, which leads to high computational time and produces biases in the system proportionally (Chae et al., 2018). Thus, in this paper, the proposed algorithm attempts to determine the maximum co-rated items of every user, using weights generated from a Relative Similarity Index (RSI).

The efficiency of a recommender system is largely assessed based on accuracy metrics. Several past studies have attempted to improve accuracy (e.g., Bobadilla et al., 2011; Patra et al., 2015). However, multiple times, the accuracy has proved inadequate in evaluating the performance of a recommender system. Hence, recommendation diversity is utilized to estimate the number of unique products recommended to users regardless of popular items (Adomavicius and Kwon, 2012, 2014; Kaminskas and Bridge, 2016; Kunaver and Požrl, 2017). A trade-off between accuracy and diversity exists in collaborative filtering. Different optimization and matrix factorization based models have been proposed in the literature to balance accuracy and diversity (Gogna & Majumdar, 2017; Liu et al., 2017). One level of diversity, known as aggregate diversity, indicates the diversity of items in the recommendation lists of entire user sets. Aggregate diversity increases user awareness of niche products, leading to enhanced sales of long-tail items (Anderson, 2009). As unpopular items cost less to produce and have a higher profit margin (e.g., lower license fees of unpopular movies), they help to improve the aggregate diversity for the recommender system. Muter and Aytekin (2017) have introduced a scalable optimization approach for improving aggregate diversity. However, in most of the studies, accuracy has been decreasing with increasing aggregate diversity. Therefore, in this research, first significant nearest neighbors of every user are identified and later, four types of prediction models are applied for calculating ratings of the unrated items. i) traditional user-based recommender system prediction model, ii) multiple linear regression analysis, iii) neural network with linear activation function and, iv) extreme learning machine with different activation functions are used in this study. In the recommender system-based prediction model, instead of K-nearest neighbors, significant nearest neighbors concept is used to filter biased nearest neighbors and improve the accuracy as well aggregate diversity for developing a healthier and robust recommender system. Timestamp (a digital record of the time of occurrence of a particular event) is applied in the recommender system for capturing aging factor of the rating in the process of finding nearest neighbors (Bagher et al., 2017; Shi, 2014). In this study, the timestamp of the targeted rating is added to the input parameter for improving the accuracy of the prediction model. The objective of this research is to develop an integrated model for enhancing the performance of recommender systems. The research contributes to the behavioral analytics and recommender system literature by providing an integrated recommender system for improved accuracy and personalization with less operation cost.

The remaining sections of the paper are discussed as follows. Past literature on prediction techniques and similarity measures are discussed in section 2. Diagnosis of the existing similarity measures is recognized and discussed in section 3. Detailed formulation is proposed in section 4. The results of the experimental analysis are presented in section 5. Finally, discussion on findings, contribution, limitations and possible future research directions are discussed in the concluding section.

2. Literature review

In this section, an extant literature review on collaborative filtering recommender systems is provided. Existing neighborhood-based prediction techniques and similarity measures are also discussed.

2.1. Conceptual background

Recommending personalized products and/or services to customers is one of the key requirements for today's e-businesses (Balakrishnan et al., 2018; Liu et al., 2015; Wang, 2015). Commercial websites have used this system for recommending customers the right products at the right time (Adomavicius et al., 2017; Alexandrescu, Butincu, & Craus, 2017; Lu, Xiao, & Ding, 2016); thereby benefiting both for themselves and their customers. Collaborative filtering is one of the most popular techniques in recommender systems due to its simplicity in concept and user-friendliness in implementation (Wu et al., 2018; Yoon et al., 2017). However, it suffers from cold start problems, scalability, over-fitting and data sparsity (Ahn, 2008; Guo et al., 2017; Patra et al., 2015). These problems extremely moderate the performance of a recommender system, while applying traditional similarity metrics in collaborative filtering techniques. There are two types of collaborative filtering namely, memory-based and model-based (Yang et al., 2017). Memory-based techniques predict items' ratings based on the nearest neighbors ratings (Ghazarian & Nematbakhsh, 2015; Guo et al., 2014). In model-based filtering, first, a model is constructed to make the prediction of item rates, along with the available information about the products (Jiang et al., 2015). Memory-based collaborative filtering is categorized into two types as user-based and item-based (Patra et al., 2015). User-based collaborative filtering (UBCF) exploits the shared structure of like-minded users. It predicts the ratings of unrated items of an individual user by means of k-nearest neighbors of user and similarity metrics (Ahn, 2008). Nearest neighborhood-based collaborative filtering approach is one of the mainstream methods for an ideal recommender system. Collaborative filtering (CF) can easily determine nearest neighbors by applying several traditional similarity metrics (Ahn, 2008; Patra et al., 2015). Various heuristic approaches have played a significant role in recognizing the k-nearest neighbors of a particular user. Typical heuristic approaches include entropy-based neighbor selection methods (Kaleli, 2014), Bhattacharyya coefficient (Patra et al., 2015), graph-based contextual modeling and post filtering (Wu et al., 2015) and proximity-impact-popularity (PIP) measure (Ahn, 2008). The PIP similarity measure has been performed in several recommender system algorithms. However, all of these heuristic approaches are time-consuming and have a certain implementation complexity. It is important to design a feasible recommendation strategy that provides desired product(s) suggestions for users. More recently, Najafabadi et al. (2017) implemented association rule mining to improve the accuracy of collaborative filtering recommendations. They employed a clustering technique on implicit data to reduce the size of the dataset and dimensionality of the item space for improving accuracy. Similarly, weighted clustering and k-means clustering methods were employed for collaborative filtering to improve the quality of recommendations (Kant el al., 2018; Salah et al., 2016; Yin et al., 2016). Adomavicius and Kwon (2012, 2014) proposed a rank-based method and an optimization based approach to improve aggregate recommendation diversity. Furthermore, context-aware recommender systems (Panniello et al., 2014), Gaussian cloud transformation (Chen et al., 2015), and a probabilistic model (Javari and Jalili, 2014) have been proposed to resolve issues associated with the accuracy of recommendation systems. Although performing collaborative filtering in sparse data remains a challenge in the recommender system field, this paper concentrates on improving the accuracy and aggregate diversity of a recommender system for sparse data.

2.2. Neighborhood-based prediction technique

The k-nearest neighbors of a targeted user (using various similarity metrics) are found in the neighborhood-based prediction model. Subsequently, ratings of that user on unrated items are predicted based on the rating patterns of the k-nearest neighbors on the corresponding items. A certain number of terminologies and symbolizations are presented in this section for simplifying the mathematical model of the rating prediction. Assume U and I as a set of users and items of a recommender system, respectively. $R(u,i)$ and $R^*(u,i)$ represent the actual rating and predicted rating of a user u on the item i. Here, u is the targeted user, whose average rating $\overline{R(u)}$ and predicted rating $R^*(u,i)$ are computed from Equation 1.1.

$$R^*(u,i) = \overline{R(u)} + \frac{\sum_{v\in S(u)} Sim(u,v)\cdot\left(R(v,i) - \overline{R(v)}\right)}{\sum_{v\in S(u)} \left|Sim(u,v)\right|} \qquad (1.1)$$

Where, $R(v,i)$ is rating of the nearest neighbor v for item i. $\overline{R(v)}$ is the mean rating of the nearest neighbor v of user u. $S(u)$ is the number of other similar (top) users who also rated item i. $Sim(u,v)$ is the similarity value between user u and its nearest neighbor v. In another neighborhood-based prediction method, prediction of the unknown rating is generated based on item-item similarity. However, in this study, only neighborhood-based prediction technique of user-user similarity is applied for predicting the unknown ratings.

2.3. Existing similarity measures

Two types of similarity measures exist in the literature, traditional and heuristic, which are consequently discussed.

The traditional similarity measures

There are several traditional similarity metrics including Cosine (COS), Adjusted Cosine (ACOS), Pearsons Correlation (COR), Constrained Pearsons Correlation (CPC), Spearman's Rank Correlation (SRC), Mean Squared Difference (MSD), Jaccard and JMSD (Bag et al., 2019; Patra et al., 2015). In this study, only three metrics COS, COR, and CPC are considered and discussed; as below:

Cosine similarity is the most popular similarity measure used in many e-commerce websites including Amazon and YouTube. Cosine similarity between two users is calculated with the cosine of the angle between rating vectors of each user as in Equation 1.2.

$$Sim(u,v)^{COS} = \frac{\sum_{i\in I(u,v)} R(u,i)\cdot R(v,i)}{\sqrt{\sum_{i\in I(u,v)} R(u,i)^2}\cdot\sqrt{\sum_{i\in I(u,v)} R(v,i)^2}} \tag{1.2}$$

Where, $R(u,i)$is the rating of user u for item i and $I(u,v)$ is the number of co-rated items of user u and v.

Similarly, **Pearson correlation coefficient** is calculated by dividing the ratio of cross-product of over rating or under rating about mean by product of sum of squares of mean rating difference (Bag et al., 2019).

$$Sim(u,v)^{COR} = \frac{\sum_{i\in I(u,v)} \left(R(u,i)-\overline{R(u)}\right)\cdot\left(R(v,i)-\overline{R(v)}\right)}{\sqrt{\sum_{i\in I(u,v)} \left(R(u,i)-\overline{R(u)}\right)^2}\cdot\sqrt{\sum_{i\in I(u,v)} \left(R(v,i)-\overline{R(v)}\right)^2}} \tag{1.3}$$

Where, $R(u,i)$ is the rating of item i by user u and $I(u,v)$is the number of co-rated items of users u and v. $\overline{R(u)}$ is the average rating given by user u. A value of -1 in pearson similarity indicates items are negatively correlated, 0 indicates items are uncorrelated and +1 indicates items are positively correlated.

Constrained Pearson’s coefficient is an extension of Pearson’s coefficient and calculated by Equation 1.4. The only difference is that median is used in the latter while mean is used in the former.

$$Sim(u,v)^{CPC} = \frac{\sum_{i\in I(u,v)} \left(R(u,i)-R_m\right)\cdot\left(R(v,i)-R_m\right)}{\sqrt{\sum_{i\in I(u,v)} \left(R(u,i)-R_m\right)^2}\cdot\sqrt{\sum_{i\in I(u,v)} \left(R(v,i)-R_m\right)^2}} \tag{1.4}$$

where, $R(u,i)$is the rating of user u for item i and $I(u,v)$ is the number of co-rated items of user u and v. R_m is the median value in the rating scale.

Heuristic similarity measure

In the heuristic similarity measure, only the PIP similarity model has been considered for comparing the results with the proposed similarity model.

Proximity impact popularity:

Most recommender systems face difficulty with the sparse dataset. This issue leads to a cold-start problem in collaborative filtering systems. To address this difficulty, Ahn (2008) introduced a heuristic similarity measure, proximity impact popularity (PIP). This heuristic similarity is calculated by the multiplication of three similarity factors: proximity, impact, and popularity. The PIP similarity measure utilizes domain-specific meanings of rating, unlike traditional similarity or distance measures (Ahn, 2008).

The PIP similarity between user $u$and v is calculated with the following equation:

$$Sim(u,v)^{PIP} = \sum_{i\in I(u,v)} PIP\left(R(u,i),R(v,i)\right) \tag{1.5}$$

Further, PIP similarity can be calculated by using Equation 1.6.

$$\begin{aligned} PIP\left(R(u,i),R(v,i)\right) = &\Pr oximity\left(R(u,i),R(v,i)\right) * \operatorname{Im} pact\left(R(u,i),R(v,i)\right) \\ &* Popularity\left(R(u,i),R(v,i)\right) \end{aligned} \tag{1.6}$$

where, $PIP\left(R(u,i),R(v,i)\right)$is the PIP value for the two ratings, $R(u,i)$ and $R(v,i)$of the co-rated item $i$by users $u$and v, respectively.

Algorithm* 1*: *Calculation of p*roximity impact popularity (PIP) *similarity measure*

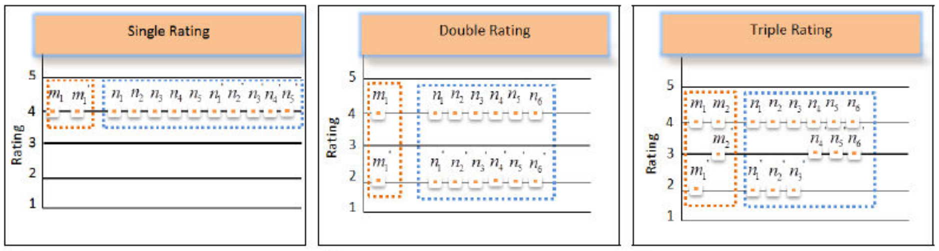
```
## Initialization
1 : $Ratings = (r_1, r_2)$
2 : $R_{\max}$ = maximum rating in the rating scale
3 : $R_{\min}$ = minimum rating in the rating scale
4 : $R_{med} = \frac{R_{\max} + R_{\min}}{2}$
5 : if $(r_1 \rangle R_{med}$ and $r_2 \langle R_{med}) or (r_1 \langle R_{med}$ and $r_2 \rangle R_{med})$ then
6 :     $Agreement(r_1, r_2) = 1$
7 : else
8 :     $Agreement(r_1, r_2) = 0$
9 : end if
## Proximity
10 : if $Agreement(r_1, r_2) == 1$ then
11 :     $D(r_1, r_2) = |r_1 - r_2|$
12 : else
13 :     $D(r_1, r_2) = 2 \cdot |r_1 - r_2|$
14 : end if
15 : $Proximity(r_1, r_2) = \{\{2 \cdot (R_{\max} - R_{\min}) + 1\} - D(r_1, r_2)\}^2$
## Impact
16 : if $Agreement(r_1, r_2) == 1$ then
17 :     $Impact(r_1, r_2) = (|r_1 - R_{med}| + 1)(|r_2 - R_{med}| + 1)$
18 : else
19 :     $Impact(r_1, r_2) = \frac{1}{(|r_1 - R_{med}| + 1)(|r_2 - R_{med}| + 1)}$
20 : end if
## Popularity
21 : $\mu_k$ = mean rating of item k by all users
22 : if $(r_1 \rangle \mu_k$ and $r_2 \rangle \mu_k) or (r_1 \langle \mu_k$ and $r_2 \langle \mu_k)$ then
23 :     $Popularity(r_1, r_2) = 1 + \left( \frac{r_1 + r_2}{2} - \mu_k \right)^2$
24 : else
25 :     $Popularity(r_1, r_2) = 1$
26 : end if
```

Proximity factor is determined by the difference between the two ratings. Further, it is checked that whether two ratings are in acceptance or denial. The pair is called in acceptance when both ratings are on the same side of the median on the rating scale. The impact factor depends on how users have preferred an item. It provides a higher credibility by being '*strongly liked*' or *'strongly disliked'*. The Popularity factor is accepted by the average rating of a co-rated item. Calculation of proximity impact popularity (PIP) similarity measure is presented in Algorithm 1 (Ahn, 2008).

3. Diagnosis of existing similarity measures

Traditionally, the measures such as COS, COR, and CPC are used for computing the similarity between a pair of users (and a pair of items) based on co-rated items (or co-rated users). Here, the number of co-rated items signifies the similarity strength between two users. However, the number of co-rated items has been overlooked in the literature while computing the similarity between two users. Thus, traditional similarities compute similarity metrics that are biased, which mislead on the performance of the recommender systems.

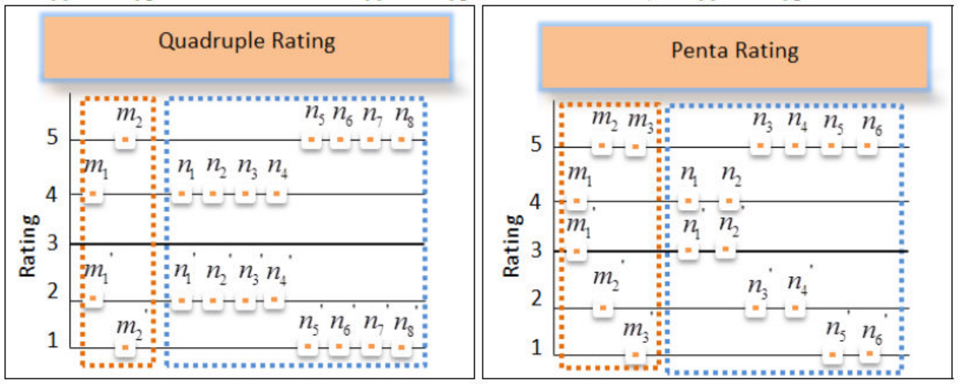
In Figure 1, five different rating values (1 to 5) have been considered with various numbers of co-rated items. The left side of the dotted line represents the first user pair (P1) and the right side represents the second user pair (P2). The number of co-rated items in P2 varies from 2 to 6 times higher than the number of co-rated items in P1. Furthermore, various types of rating distances have been considered to identify the limitations of traditional similarity in the context of recommender system. Details of the limitations are presented in Figure 1 and Figure 2.



(a) Second pair P2 have 5 times more co-rated items than first pair P1, but the similarity values are same for both pairs and these are: $COS_{P1} = COS_{P2} = 1$, $COR_{P1} = COR_{P2} = 0$, $CPC_{P1} = CPC_{P2} = 1$

(b) Second pair P2 have 6 times more co-rated items than first pair P1, but the similarity values are same for both pairs and these are: $COS_{P1} = COS_{P2} = 1$, $COR_{P1} = COR_{P2} = 0$, $CPC_{P1} = CPC_{P2} = -1$

(c) Second pair P2 have 3 times more co-rated items than first pair P1, but the similarity values are same for both pairs and these are: $COS_{P1} = COS_{P2} = 0.981$, $COR_{P1} = COR_{P2} = 0$, $CPC_{P1} = CPC_{P2} = -0.707$

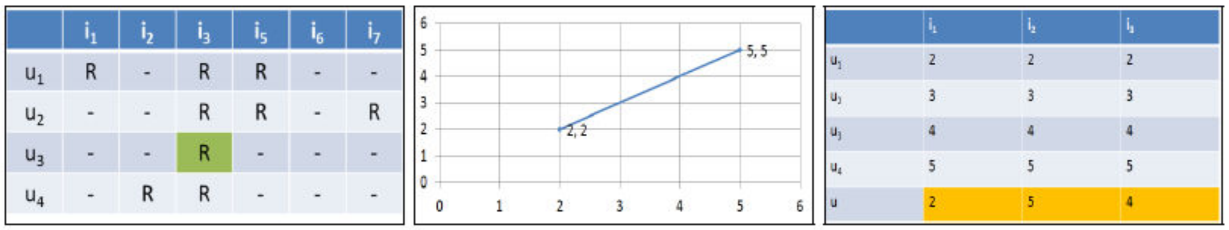


(d) Second pair P2 have 4 times more co-rated items than first pair P1, but the similarity values are same for both pairs and these are: $COS_{P1} = COS_{P2} = 0.908$, $COR_{P1} = COR_{P2} = -1$, $CPC_{P1} = CPC_{P2} = -1$

(e) Second pair P2 have 2 times more co-rated items than first pair P1, but the similarity values are same for both pairs and these are: $COS_{P1} = COS_{P2} = 0.888$, $COR_{P1} = COR_{P2} = -0.866$, $CPC_{P1} = CPC_{P2} = -0.894$

Figure 1. Diagnosis of existing traditional similarity measures

Figure 2 (a) shows that the number of co-rated items between user u_3 and any other users, and it is always one. In this scenario, COS always generates similarity score 1, COR cannot be calculated, CPC generates a similarity score of 0, even though rating values of both users are 3. Furthermore, CPC always generates a similarity score of 1, while rating values of both users are greater than 3. Figure 2(b) shows that two vectors are on the same line. In this case; COS, COR and CPC generate the same values with the scenario in Figure 2(a). Figure 2(c) shows the flat user ratings which also lead to the same kind of problem. COS generates the same similarity score 0.9467 between user u and any other users regardless of the rating value. Because of the high probability of occurring by chance in the above case, the correlation-based similarity measure performs inaccurately. This issue becomes serious when rating datasets are sparse.



	i_1	i_2	i_3	i_5	i_6	i_7
u_1	R	-	R	R	-	-
u_2	-	-	R	R	-	R
u_3	-	-	R	-	-	-
u_4	-	R	R	-	-	-

	i_1	i_2	i_3
u_1	2	2	2
u_2	3	3	3
u_3	4	4	4
u_4	5	5	5
u	2	5	4

(a) Single Co-rated Item (b) Two vectors are on the same line (c) Flat user ratings

Figure 2. Diagnosis of traditional similarity measures

4. Proposed formulation and algorithm design

The similarity measures of COS, COR, and CPC are implemented widely in the collaborative filtering domain (Ahn, 2008). However, sometimes, these similarity measures may mislead about the performance of collaborative filtering; where dissimilar users may accept being homogeneous and similar users may lead to being heterogeneous. The paper uses three similarity measures, weight metrics, and four machine learning approaches for predicting the ratings.

4.1. Motivation

In a previous study, Singh et al. (2015) introduced a new metric, the relative similarity index (RSI), to improve accuracy along with diversity of recommender systems. Basically, cosine similarity has only been performed in the existing RSI algorithm, where RSI is generated from the multiplication of cosine similarities and the calculated weights. Weights are estimated from the ratio of a number of co-rated items of targeted users’ pairs and the maximum number of co-rated items present within any two users’ pairs in the entire dataset. This RSI algorithm performs well for cosine similarity; however, several significant settings have been overlooked in the existing RSI algorithm, which motivates us to propose a new SNN identification approach. It is known that weight is a sensitive entity, where insignificant value leads to an inappropriate similarity index. In addition to the existing algorithm, RSI similarity is directly used to predict the unrated ratings, the value of which is smaller than the traditional similarity. The smaller value of similarity decreases the accuracy of the prediction model. Moreover, the existing algorithm has only been performed on a single traditional (i.e., cosine) similarity.

4.2. New SNN identification approach for collaborative filtering

To obtain a better result, SNN identifying approach has been incorporated into a collaborative filtering algorithm. Relative similarity weights are set to find the top k-nearest neighbors. This measure is an easy plugin to existing CF systems by exchanging only the significant nearest neighbors with the nearest neighbor; thus, not requiring vast re-implementation or additional data collection.

To prevent the bias of the similarity model, the denominator of the weighted values is computed from a maximum number of co-rated items present between the targeted user and any other user, instead of maximum number of co-rated items present within any two users' pairs in the entire dataset.

$$RSI_{user,next\_user} = \left(\frac{co-rated}{\max co-rated_{user}}\right)\cdot\left(Sim_{user,next\_user}\right) \tag{1.7}$$

The users with top RSI values with respect to the target user are called the Significant Nearest Neighbors (SNN). In the proposed algorithm, RSI is used to find the most SNN and the traditional similarity is applied to predict the unrated item's score.

$$SNN_k = Top_k\left(RSI_{user}\right) \tag{1.8}$$

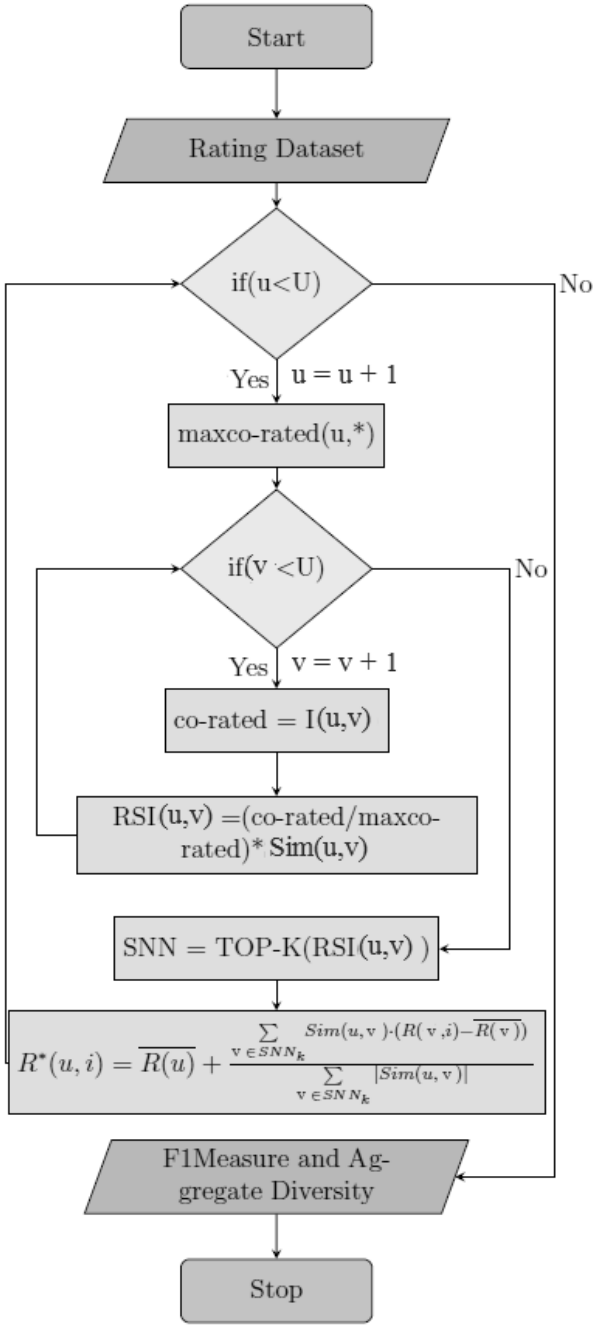

Start
Rating Dataset
if(u<U)
No
Yes
u = u + 1
maxco-rated(u,*)
if(v <U)
No
Yes
v = v + 1
co-rated = I(u,v)
RSI(u,v) =(co-rated/maxco-rated)* Sim(u,v)
SNN = TOP-K(RSI(u,v))
$R^*(u,i) = \overline{R(u)} + \frac{\sum_{v \in SNN_k} Sim(u,v)\cdot(R(v,i)-\overline{R(v)})}{\sum_{v \in SNN_k} |Sim(u,v)|}$
F1Measure and Aggregate Diversity
Stop

Figure 3: Flowchart of significant nearest neighbor algorithm

Algorithm* 2 *: Identify Significant Nearest Neighbors (SNN)

```
Input : User - item rating metric
Output : Value of predicted ratings
1 : $\max - co - rated_{[1-U]} = [0,...,0], RSI(u,v) = [\ ]$
## Find the max - co - rated items of every users
2 : for the user $u = 1$ to $(|U| - 1)$ do
3 :   for the user $v = u + 1$ to $|U|$ do
4 :     if $|I_u \cap I_v| > \max - co - rated_u$ then
5 :       $\max - co - rated_u = |I_u \cap I_v|$
6 :     end if
7 :     if $|I_u \cap I_v| > \max - co - rated_v$ then
8 :       $\max - co - rated_v = |I_u \cap I_v|$
9 :     end if
10 :  end for
11 : end for
## RSI metrics for every users
12 : for the user $u = 1$ to $(|U| - 1)$ do
13 :   for the user $v = u + 1$ to $|U|$ do
14 :       $Sim(u,v) = Sim(v,u)^{COS}$ or $Sim(v,u)^{COR}$ or $Sim(v,u)^{CPC}$
15 :       $RSI(u,v) = Sim(u,v) \times \left( \frac{|I_u \cap I_v|}{max\text{-}co\text{-}rated_u} \right)$
16 :       $RSI(v,u) = Sim(v,u) \times \left( \frac{|I_u \cap I_v|}{max\text{-}co\text{-}rated_v} \right)$
17 :   end for
18 : end for
## Top significant nearest neighbors
19 : for the user $u = 1$ to $|U|$ do
20 :   $SNN_K = Top_K(RSI(u,*))$
21 : end for
```

In this model, misleading nearest neighbors are filtered by prioritizing additional numbers of co-rated items to improve the performance of the prediction model. The flowchart of the proposed SNN algorithm is shown in Figure 3 to represent the functionality of the proposed method. Additionally, identification of SNN, prediction model using SNN and three similarity metrics are shown in algorithms 2 and 3 respectively to illustrate the functionality of the models.

In algorithm 2, the $\max-co-rated_u$ and the $\max-co-rated_v$ cannot always be the same, it depends on previously stored value in $\max-co-rated_u$ and the $\max-co-rated_v$. Suppose previously stored $\max-co-rated_u$ and $\max-co-rated_v$ are 3 and 5 respectively. Now, let's consider $|I_u \cap I_v|$ is 4. The updated values of $\max-co-rated_u$ and the $\max-co-rated_v$ will be 4 and 5, because $|I_u \cap I_v| < 5$ and hence there is no change in the $\max-co-rated_v$. This strategy has been performed to reduce the time complexity. Thus, it can be observed that the second loop starts from user v = u+1 instead for v = 1.

4.3. Prediction Model

After finding the SNN, four types of prediction models are used for rating prediction.

Prediction through recommendation algorithm

In algorithm 3, the pseudo code of rating prediction using SNN approach is shown, where Sim function represents the similarity value of user u with the significant nearest neighbor v.

```
Algorithm 3: Rating prediction using traditional similarity with SNN
Input: User - item rating metric
Output: Value of predicted ratings
 1: $R^*(u,i) = [\ ]$
## Prediction of un - rated items
 2: for the user $u = 1$ to $|U|$ do
 3:   for the item $i = 1$ to $|I|$ do
 4:     if $R(u,i) == 0$ then
 5:       $R^*(u,i) = \overline{R(u)} + \frac{\sum_{v \in SNN_K} Sim(u,v) \cdot \left(R(v,i) - \overline{R(v)}\right)}{\sum_{v \in SNN_K} |Sim(u,v)|}$
 6:     end if
 7:   end for
 8: end for
```

Prediction through other machine learning algorithms

In this prediction model, three types of machine learning approaches are applied namely, multiple linear regression analysis, neural network with linear activation function and extreme learning machine with different activation functions.

The SNNs of all users are identified and then ratings of top 10 SNN items are taken as an input parameter by multiplying the corresponding similarity value. The rating of the targeted user on a particular item is considered as an output parameter. Furthermore, the timestamp of the targeted rating is also added to the input parameter to identify the significance of the timestamp in the prediction model for accurate prediction. To fit with other parameters, the value of the timestamp is normalized in the rating scale of 1 to 5 through the following equation.

$$\begin{aligned}\mathrm{NormTimestamp} = (((&Old\mathrm{Timestamp}Value - Old\mathrm{Timestamp}Min) * (NewMax - NewMin)) \\ &/ (Old\mathrm{Timestamp}Max - Old\mathrm{Timestamp}Min)) + NewMin\end{aligned} \tag{1.9}$$

where, $Old\mathrm{Timestamp}Value$ is the actual value of timestamp and $\mathrm{NormTimestamp}$ is the normalized value of the timestamp in the rating scale of 1 to 5. $Old\mathrm{Timestamp}Min$ and $Old\mathrm{Timestamp}Max$ are the minimum and maximum value of timestamp respectively. $NewMax$ and $NewMin$ are maximum and minimum normalized values of timestamp (here, 5 and 1), respectively.

Later, multiple linear regression analysis has been carried out to predict the targeted user's rating on a specific item. In the regression model, output parameter is taken as a dependent variable and other input parameters are considered as independent variables. The details of all parameters have been shown in Table 1. Moreover, regression analysis with and without timestamp (i.e. with and without SNN11_user_rating variable) have been performed to identify the significance of the timestamp in the proposed prediction model.

To select the number of hidden nodes in the hidden layer, various approaches such as (1) try and error, (2) rule of thumb, (3) simple and (4) two-phase methods are presented in the literature. In this study, the rule of thumb method was applied to select the number of hidden nodes in the hidden layer (Heaton, 2017; Panchal & Panchal, 2014).

Table 1: List of variable names, types and corresponding values of multiple linear regressions

Variable Name	Variable Type	Values
Targated_user_rating (Output)	Dependent	Rating (targeted user)
SNN1_user_rating (Input)	Independent	Sim (SNN1) * rating (SNN1)
SNN2_user_rating (Input)	Independent	Sim (SNN2) * rating (SNN2)
SNN3_user_rating (Input)	Independent	Sim (SNN3) * rating (SNN3)
SNN4_user_rating (Input)	Independent	Sim (SNN4) * rating (SNN4)
SNN5_user_rating (Input)	Independent	Sim (SNN5) * rating (SNN5)
SNN6_user_rating (Input)	Independent	Sim (SNN6) * rating (SNN6)
SNN7_user_rating (Input)	Independent	Sim (SNN7) * rating (SNN7)
SNN8_user_rating (Input)	Independent	Sim (SNN8) * rating (SNN8)
SNN9_user_rating (Input)	Independent	Sim (SNN9) * rating (SNN9)
SNN10_user_rating (Input)	Independent	Sim (SNN10) * rating (SNN10)
SNN11_user_rating (Input)	Independent	Normalized (Timestamp)

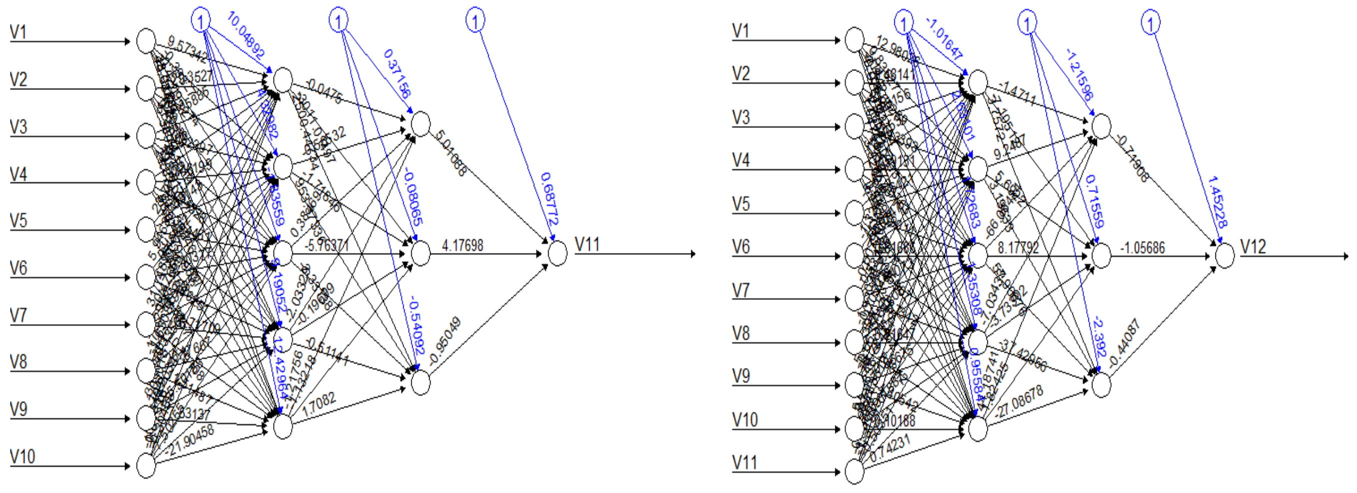
1. The number of hidden neurons should be in the range between the numbers of the input nodes and the numbers of the output nodes.

2. The number of hidden neurons should be the sum of 2/3 of numbers of the input nodes and the numbers of the output nodes.

10 and 11 numbers of input nodes are used, whereas the number of output node is 1 in this study. Therefore, based on the above rule of thumb, the number of hidden nodes is (10*(2/3)) + 1 = approx. 8 for the first neural network, and (11*(2/3)) + 1 = approx. 8 for the second neural network. Thus, two neural networks such as 10-5-3-1 and 11-5-3-1 have been taken in this analysis to predict the rating where, a timestamp is added as one input parameter in the second neural network. A list of parameter names and corresponding values of the 11-5-3-1 neural network is presented in Table 2. Moreover, a sample structure of the neural networks 10-5-3-1 and 11-5-3-1 with trained weights are shown in Figures 4 (a) and 4 (b), respectively.

Table 2: List of variable names and corresponding values of 11-5-3-1 neural network

Variable Name	Values
V12 (Output)	Rating (targeted user)
V1 (Input)	Sim (SNN1) * rating (SNN1)
V2 (Input)	Sim (SNN2) * rating (SNN2)
V3 (Input)	Sim (SNN3) * rating (SNN3)
V4 (Input)	Sim (SNN4) * rating (SNN4)
V5 (Input)	Sim (SNN5) * rating (SNN5)
V6 (Input)	Sim (SNN6) * rating (SNN6)
V7 (Input)	Sim (SNN7) * rating (SNN7)
V8 (Input)	Sim (SNN8) * rating (SNN8)
V9 (Input)	Sim (SNN9) * rating (SNN9)
V10 (Input)	Sim (SNN10) * rating (SNN10)
V11 (Input)	Normalized (Timestamp)



(a) 10-5-3-1 neural network (b) 11-5-3-1 neural network

Figure 4: Structure of the neural networks with trained weights

Extreme learning machine

An extreme learning machine (ELM) is a single or multiple hidden layer feed-forward neural network (FNN)-type learning system, whose input weights and hidden layer biases are randomly assigned, while output weights need tuning (Liu & Xu, 2018). In this study, an online sequential extreme learning machine with random weights is used with four different activation functions (Huang et al., 2012; Huang et al., 2006; Tang et al., 2016) such as radial basis function with Gaussian kernels (rbf), sigmoidal function (sig), sine function (sin) and hard limit function (hardlim).

5. Analysis and results

In this section, the dataset and experiment setting of the research is described. Later, the performance of the proposed model is compared with the existing recommender system based on several traditional and heuristic similarity metrics.

5.1. Dataset and experiment setting

A key input for the recommender system is the feedback given by the users for their purchased product and/or service. This system predicts the best alternatives for customers and assists them in choosing suitable products. The feedback data is based on past behavioral data of similar customers. In this study, MovieLens, an established dataset was used in the recommender system domain to validate the performance of the proposed model. In the dataset, there are 943 users and 1682 items with the rating scale ranging from 1 to 5.

To obtain the results, first step is to model the raw data in a user-item metric and then, use neighborhood-based CF along with the modified similarity methods. Once the similarities are obtained, RSI rank metrics determine k-nearest neighbors. Ratings are predicted using the nearest neighbors, and accordingly the recommendations are made to the users of the system.

5.2. Evaluation metrics

In this analysis, accuracy metrics and aggregate diversity are considered to assess the performance of the proposed recommender system algorithm. There are mainly two types of accuracy metrics to evaluate the efficiency of a recommender system; various statistical accuracy metrics and decision support metrics. Well-known statistical accuracy metrics include mean absolute error (MAE) and root means square error (RMSE). The statistical accuracy metrics show how well a recommender system can predict the rating of all user-item pairs. The decision support metrics consist of precision, recall, and F1-measures. If N is the top n items, and relevance threshold denotes T_h, then the recommendation accuracy can be computed (Singh et al., 2015; Wu et al., 2015) as:

$$\mathrm{Pr}\,ecision = \frac{\left|\{relevant - items\} \cap \{top - N - items\}\right|}{N} \tag{1.10}$$

$$\mathrm{Re}\,call = \frac{\left|\{relevant - items\} \cap \{top - N - items\}\right|}{\left|\{relevant - items\}\right|} \tag{1.11}$$

$$F1\,measure = \frac{precision + recall}{2 \times precision \times recall} \tag{1.12}$$

Here, precision is the percentage of truly relevant ratings of the items, among those which are predicted by a particular recommender system. The recall is the percentage of correctly predicted ratings of correlated items among all the ratings known to be relevant; whereas, the F1-measure is the harmonic mean of precision and recall. In this study, the accuracy of the proposed algorithm is measured by using the evaluation metric of the F1-measure.

Accuracy is insufficient to measure the efficiency of a recommender system. As a result, aggregate diversity is also an essential feature in the recommender system to assess the performance of the algorithm. There are mainly two types of diversity: individual and aggregate (Bobadilla et al., 2013). Individual diversity is the dissimilarity among the items recommended to a user. Aggregate diversity can be found by calculating the number of distinct elements recommended to the entire user segment. Aggregate diversity can be computed from the following equation (Adomavicius and Kwon, 2012, 2014):

$$diversity = \left|\bigcup_{u \in U} L_n(u)\right| \tag{1.13}$$

where u is any particular user, U is the total user in the dataset and $L_n(u)$is the list of relevant items recommended to the user u.

5.3. Results and discussion

All traditional similarity measures have recognized that, if the threshold value decreases, the number of recommended items increases for both approaches (SSN and KNN). In the same way, increasing the recommended items raises the F1 measure and reduces aggregate diversity. The threshold value is the acceptable predicted rating, which can be a range from 1 to 5. The variation of F1 measure and aggregate diversity are observed in the threshold values of 3.5 to 4.5 as this threshold value is significant for recommending items. In this study, four instances of nearest neighbors such as 5, 20, 50, and 100 are considered to obtain results. Further, three traditional similarities, COS, COR, and CPC, are utilized for comparing the generated outcomes from KNN and SNN approaches. The results of the proposed SNN method perform remarkably better when five nearest neighbors are considered.

Figure 5 displays the F1 measure of traditional similarity using KNN, traditional similarity using SNN, and PIP similarity. PIP similarity, an effective algorithm in recommendation systems for sparse data, is used to compare the performance of traditional similarity on behalf of the KNN and SNN approaches. Traditional similarity with the support of the SNN approach always performs better compared to the KNN approach. Furthermore, the F1 measure of traditional similarity using SNN performs better than the PIP similarity, while the model uses 5 nearest neighbors.

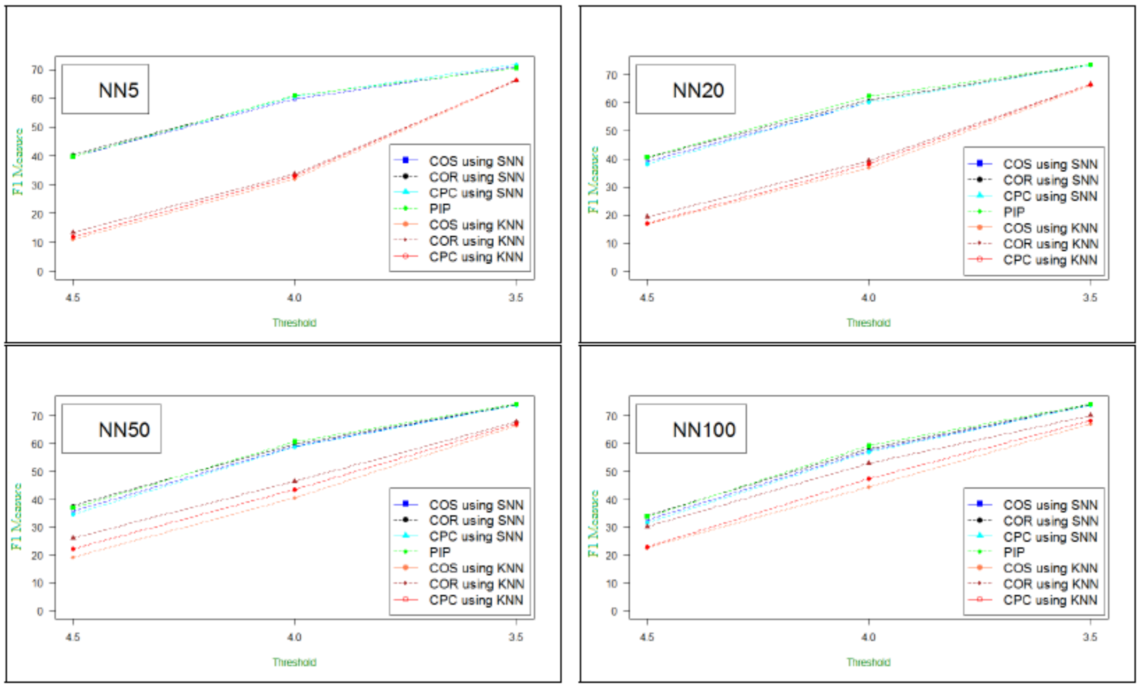


Figure 5: F1 measure of traditional similarity using KNN, traditional similarity using SNN, and PIP similarity

Table 3 shows the variation of F1 measure and aggregate diversity while using KNN and SNN methods. Here, Th., F1 and AD represent the threshold value, F1 measure (harmonic mean of precision and recall) and aggregate diversity (AD), respectively. The threshold value is the acceptable predicted rating for recommendation to the customers, which can be in any range of rating vector (here, between 1 and 5). Higher threshold value indicates that the system recommends most likely items to the customers. In our experiment, we checked the performance of three traditional similarities such as Cosine, Pearson, and constrained Pearson similarity. The accuracy increases with the loss of aggregate diversity and vice versa. There is a noticeable trade-off that exists between F1 measure and diversity. However, the advantage of the model is that the F1 measure and aggregate diversity are simultaneously improved when five nearest neighbors and SNN approach (instead of KNN approach) are used with the consideration of a (assumed) threshold value of 4 or 4.5. A slight increment in the accuracy and decrement in aggregate diversity have been noticed while using SNN approach instead of KNN with 5 nearest neighbors and 3.5 threshold value. However, if the system wishes to recommend most likely items to the customers, then it is better to use 5 nearest neighbors and SNN approach instead of the KNN approach.

Table 3: F1 measure and aggregate diversity of using KNN and SNN methods

Th.	COS				COR				CPC				NN
	F1		AD		F1		AD		F1		AD		
	KNN	SNN	KNN	SNN	KNN	SNN	KNN	SNN	KNN	SNN	KNN	SNN	
3.5	66	71	1301	1216	66	72	1301	1260	66	72	1301	1250	5
4	32	60	880	880	34	61	883	940	33	61	884	922	
4.5	11	40	321	578	13	40	352	638	12	40	347	591	
3.5	66	73	1302	1084	67	73	1302	1183	66	73	1302	1148	20
4	37	60	888	786	39	61	891	899	38	60	897	837	
4.5	17	39	420	477	19	40	439	582	17	38	441	507	
3.5	66	74	1301	1040	68	74	1300	1106	67	74	1302	1079	50
4	40	59	910	724	46	60	904	825	43	59	918	777	
4.5	19	36	468	402	26	38	524	483	22	35	511	448	
3.5	67	74	1294	1011	70	74	1261	1051	68	74	1292	1045	100
4	44	57	908	696	53	58	910	778	47	57	930	746	
4.5	23	33	519	372	30	34	617	451	23	32	573	408	

Figure 6 represents the MAE values while using a standard prediction model of recommender system and various machine learning algorithms. It is observed that the timestamp significantly supports in predicting ratings accurately. Further, it is noticed that the generated SNN values from CPC perform better than other similarity approaches.

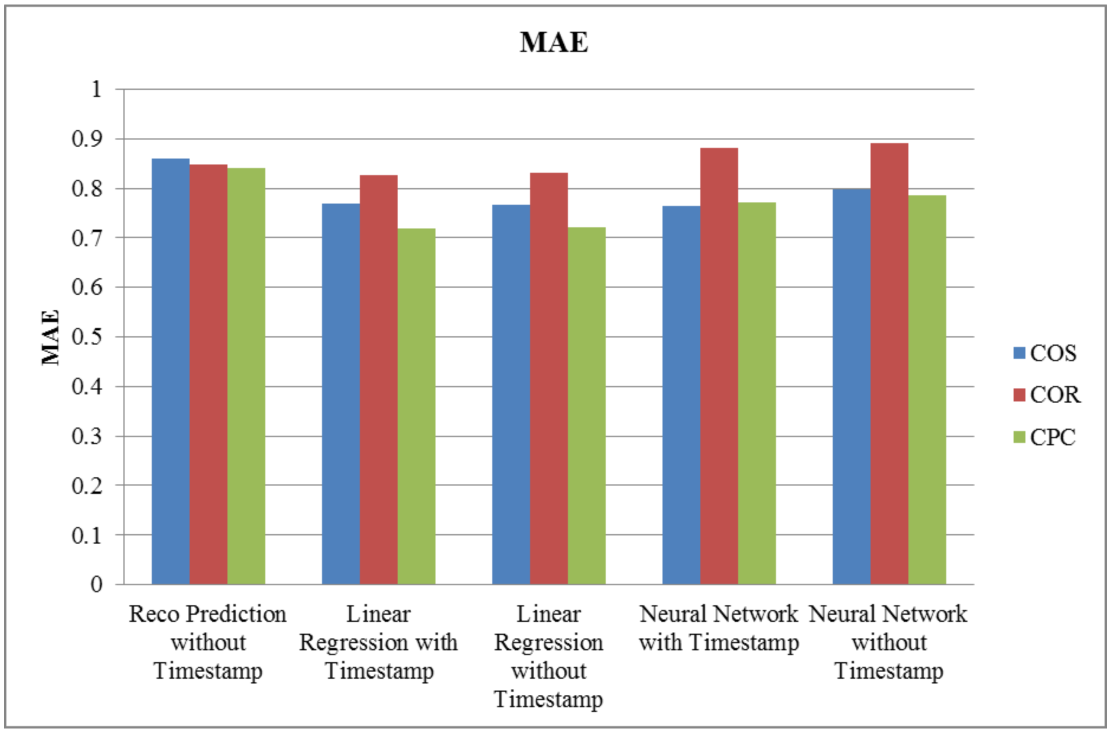


Figure 6: MAE values of using various machine learning algorithms

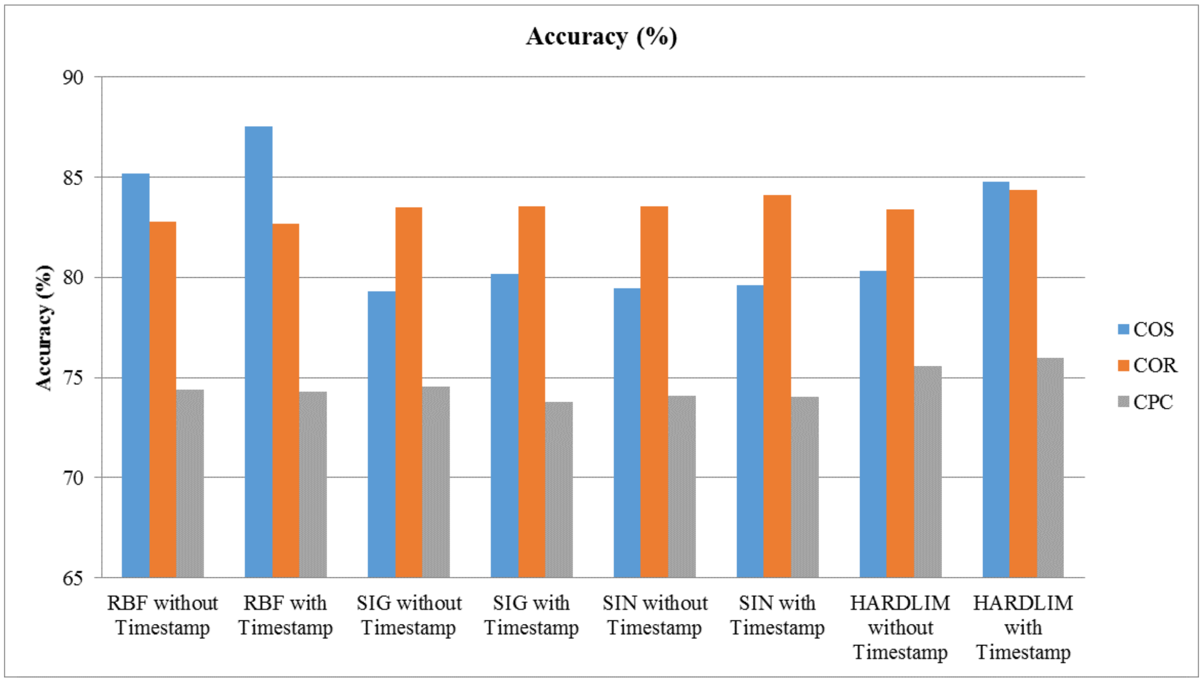


Figure 7: Prediction accuracy of using ELM with different activation functions

Moreover, Figure 7 shows the prediction accuracy of unrated items using ELM with different activation functions. COS, COR, and CPC are three traditional similarity measures which are used to find the SNNs. In this analysis, SNN generated from COS and COR perform better than CPC. Moreover, the combination of COS, utilization of timestamp parameter and ELM with RBF activation function provides better results.

6. Conclusion

The aim of this research was to develop a prediction model for online companies, with the ability of providing personalization and tailored services to their online customers. In order to achieve the research aim, first the drawbacks of current similarity metrics were identified and discussed. Past similarity matrices mislead about the similarity value and cause the selection of biased nearest neighbors. This problem increases when limited number of co-rated items are present in the dataset (i.e., sparse dataset). In this study, weight metrics were adopted to filter biased nearest neighbors. After removing the biased nearest neighbors, the model identified the significant nearest neighbors and applied four types of prediction models. By removing the biased nearest neighbors, the computation cost decreased as it used only significant nearest neighbors. Furthermore, the accuracy and aggregate diversity have both been improved with the help of the unbiased similarity metrics. As the aggregate diversity increases the user awareness of niche products, it leads to enhanced sales of long-tail items for online companies. Clearly, the proposed prediction model can increase productivity in terms of profit by selling diverse products, while meeting customer interests. It is found that the use of a timestamp also helps to improve the personalization in the recommender system. The study developed a robust computational decision-making model to achieve accuracy and aggregate diversity in a recommender system based on past interactive, behavioral data. The proposed prediction model contributes by enhancing different aspects of productivity by selling diverse products, while meeting consumer interests.

Besides developing a SNN based algorithm, this study contributes to practice in the following ways. Firstly, it directs managers to make use of appropriate algorithms based on their needs. Multiple linear regression analysis and neural network are found to be the most appropriate for decreasing MAE value in the recommender system. However, it should be noted that it increases the computational cost. Extreme learning machine is found to be suitable for accurate prediction with low computation cost. As the aggregate diversity increases, developed integrated model helps to enhance the sale of long-tail items. The study also directs e-businesses in choosing appropriate learning algorithm based on their requirements in recommender systems.

The proposed model only investigated the importance of a single type of side information (i.e., timestamp). However, in real-life applications, users may have various types of behavioral reactions towards items of interest. For instance, users often click and view product details, search and compare several alternative products, zoom in and out of product images, and so on, before purchasing a product. Thus, in the future, such additional information (clickstream, transaction, and user online behavior data) can be incorporated into the proposed model to perform a behavioral recommendation system. Proposed prediction model for recommendation system could perform poorly, if the available rating vectors are in the form of binary rating scale (i.e. only like dislike data are available). Moreover, users may have different types of communities in their social networks, including those based on friendship ties, and others based on common interests or behaviors. Interestingly, even the context of user-item interactions is often multi-dimensional (e.g. temporal, geographical, social). The multiple relations among users, items and related data create new challenges for the proposed model. In the future, an advanced model can be developed to exploit multi-dimensional (homogeneous and heterogeneous) information to provide users with personalized recommendations. The proposed recommendation system could be meaningless for the highly dense dataset; as traditional similarity methods are adequate to identify appropriate nearest neighbors in the highly dense dataset. In this study, only a single dataset has been used to perform the proposed model. Thus, this analysis can be extended by using developed algorithms with different traditional similarity approaches and other datasets like Jester, Book-Crossings, Ciao, Douban, Epinions, FilmTrust, etc. Furthermore in this research, exact amount of productivity increments for the proposed model have not been explicitly captured. However, an evident improvement in multiple factors such as accuracy, visibility, throughput and sales is indicative of improved productivity.

In the recent digital era, the large volume of rating data, velocity of incremental updates in the internet and variety of side information create challenges for the scalable prediction of user preferences. Big data analytics is required to overcome such challenges, which are not considered in this model. In the future instead of using neural network and regression analysis to handle big data problems, a high-performance extreme learning machine can be applied. The integrated model helps to predict interests in products based on their and similar other customer's behavior; thus, aiding online companies to provide personalized recommendations to enhance various aspects of productivity.

References

Adomavicius, G., Bockstedt, J. C., Curley, S. P., & Zhang, J. (2017). Effects of online recommendations on consumers' willingness to pay. *Information Systems Research*, *29*(1), 84–102.

Adomavicius, G., & Kwon, Y. (2012). Improving aggregate recommendation diversity using ranking-based techniques. *Knowledge and Data Engineering, IEEE Transactions on*, *24*(5), 896–911.

Adomavicius, G., & Kwon, Y. (2014). Optimization-based approaches for maximizing aggregate recommendation diversity. *INFORMS Journal on Computing*, *26*(2), 351–369.

Ahn, H. J. (2008). A new similarity measure for collaborative filtering to alleviate the new user cold-starting problem. *Information Sciences*, *178*(1), 37–51.

Alexandrescu, A., Butincu, C. N., & Craus, M. (2017). Recommending Products and Services Belonging to Online Businesses Using Intelligent Agents. *Service Science*, *9*(4), 338–348.

Anderson, C. (2009). *The Longer Long Tail: How Endless Choice is Creating Unlimited Demand*. Random House Business. Retrieved from https://books.google.co.in/books?id=3bN-PwAACAAJ

Andjelkovic, I., Parra, D., & O'Donovan, J. (2019). Moodplay: Interactive music recommendation based on Artists' mood similarity. *International Journal of Human-Computer Studies*, *121*, 142–159.

Bag, S., Kumar, S., Awasthi, A., & Tiwari, M. K. (2019). A noise correction-based approach to support a recommender system in a highly sparse rating environment. Decision Support Systems, 118, 46–57. https://doi.org/10.1016/j.dss.2019.01.001

Bag, S., Kumar, S. K., & Tiwari, M. K. (2019). An efficient recommendation generation using relevant Jaccard similarity. Information Sciences, 483, 53–64. https://doi.org/10.1016/j.ins.2019.01.023

Bag, S., Tiwari, M. K., & Chan, F. T. S. (2019). Predicting the consumer's purchase intention of durable goods: An attribute-level analysis. Journal of Business Research, 94, 408–419. https://doi.org/10.1016/j.jbusres.2017.11.031

Bagher, R. C., Hassanpour, H., & Mashayekhi, H. (2017). User trends modeling for a content-based recommender system. *Expert Systems with Applications*, *87*, 209–219.

Balakrishnan, J., Cheng, C.-H., Wong, K.-F., & Woo, K.-H. (2018). Product recommendation algorithms in the age of omnichannel retailing--An intuitive clustering approach. *Computers & Industrial Engineering*, *115*, 459–470.

Besbes, O., Gur, Y., & Zeevi, A. (2015). Optimization in online content recommendation services: Beyond click-through rates. *Manufacturing & Service Operations Management*, *18*(1), 15–33.

Bobadilla, J., Ortega, F., Hernando, A., & Alcalá, J. (2011). Improving collaborative filtering recommender system results and performance using genetic algorithms. *Knowledge-Based Systems*, *24*(8), 1310–1316.

Bobadilla, J., Ortega, F., Hernando, A., & Gutiérrez, A. (2013). Recommender systems survey. *Knowledge-Based Systems*, *46*, 109–132.

Burke, R., O'Mahony, M. P., & Hurley, N. J. (2015). Robust collaborative recommendation. In *Recommender systems handbook* (pp. 961–995). Springer.

Chae, D.-K., Lee, S.-C., Lee, S.-Y., & Kim, S.-W. (2018). On identifying k-nearest neighbors in neighborhood models for efficient and effective collaborative filtering. *Neurocomputing*, *278*, 134–143.

Chen, J., Liu, Y., & Li, D. (2015). Enhancing Recommender Diversity Using Gaussian Cloud Transformation. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, *23*(4), 521–544.

Cpcstrategy. (2017). The 2016 guide to Amazon sponsored products. Retrieved from http://cdn-website.cpcstrategy.com/wp-content/uploads/amazon-sponsored-products-3p-sellers.pdf

Economictimes. (2018). What Walmart's Flipkart acquisition means for India, consumers and its arch-rival Amazon. Retrieved from https://economictimes.indiatimes.com/industry/services/retail/softbank-ceo-confirms-walmart-flipkart-deal/articleshow/64093437.cms,

Ghazarian, S., & Nematbakhsh, M. A. (2015). Enhancing memory-based collaborative filtering for group recommender systems. *Expert Systems with Applications*, *42*(7), 3801–3812.

Gogna, A., & Majumdar, A. (2017). DiABlO: Optimization based design for improving diversity in recommender system. *Information Sciences*, *378*, 59–74.

Guo, G., Qiu, H., Tan, Z., Liu, Y., Ma, J., & Wang, X. (2017). Resolving data sparsity by multi-type auxiliary implicit feedback for recommender systems. *Knowledge-Based Systems*, *138*, 202–207. https://doi.org/10.1016/j.knosys.2017.10.005

Guo, G., Zhang, J., Thalmann, D., & Yorke-Smith, N. (2014). Leveraging prior ratings for recommender systems in e-commerce. *Electronic Commerce Research and Applications*, *13*(6), 440–455.

Heaton, J. (2017). Heaton Research: The Number of Hidden Layers. Retrieved from https://www.heatonresearch.com/2017/06/01/hidden-layers.html,

Huang, G.-B., Zhou, H., Ding, X., & Zhang, R. (2012). Extreme learning machine for regression and multiclass classification. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, *42*(2), 513–529.

Huang, G.-B., Zhu, Q.-Y., & Siew, C.-K. (2006). Extreme learning machine: theory and applications. *Neurocomputing*, *70*(1–3), 489–501.

Javari, A., & Jalili, M. (2014). A probabilistic model to resolve diversity--accuracy challenge of recommendation systems. *Knowledge and Information Systems*, 1–19.

Jiang, S., Qian, X., Shen, J., Fu, Y., & Mei, T. (2015). Author topic model-based collaborative filtering for personalized POI recommendations. *IEEE Transactions on Multimedia*, *17*(6), 907–918.

Kaleli, C. (2014). An entropy-based neighbor selection approach for collaborative filtering. *Knowledge-Based Systems*, *56*, 273–280.

Kaminskas, M., & Bridge, D. (2016). Diversity, Serendipity, Novelty, and Coverage: A Survey and Empirical Analysis of Beyond-Accuracy Objectives in Recommender Systems. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, *7*(1), 2.

Kant, S., Mahara, T., Jain, V. K., Jain, D. K., & Sangaiah, A. K. (2018). LeaderRank based k-means clustering initialization method for collaborative filtering. *Computers & Electrical Engineering*, *69*, 598–609.

Karabadji, N. E. I., Beldjoudi, S., Seridi, H., Aridhi, S., & Dhifli, W. (2018). Improving Memory-Based User Collaborative Filtering with Evolutionary Multi-Objective Optimization. *Expert Systems with Applications*.

Koren, Y. (2010). Factor in the neighbors: Scalable and accurate collaborative filtering. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, *4*(1), 1.

Kunaver, M., & Požrl, T. (2017). Diversity in recommender systems--A survey. *Knowledge-Based Systems*, *123*, 154–162.

Li, Y., Wang, D., He, H., Jiao, L., & Xue, Y. (2017). Mining intrinsic information by matrix factorization-based approaches for collaborative filtering in recommender systems. *Neurocomputing*, *249*, 48–63.

Liu, J., Zhang, J., Li, Y., He, S., & Zheng, C. (2015). The Mobile Personalized Recommendation Model Containing Implicit Intention. In *National Conference on Big Data Technology and Applications* (pp. 26–33).

Liu, M., Pan, W., Liu, M., Chen, Y., Peng, X., & Ming, Z. (2017). Mixed similarity learning for recommendation with implicit feedback. *Knowledge-Based Systems*, *119*, 178–185.

Liu, X., & Xu, L. (2018). The Universal Consistency of Extreme Learning Machine. *Neurocomputing*.

Lu, S., Xiao, L., & Ding, M. (2016). A video-based automated recommender (VAR) system for garments. *Marketing Science*, *35*(3), 484–510.

Muter, I., & Aytekin, T. (2017). Incorporating Aggregate Diversity in Recommender Systems Using Scalable Optimization Approaches. *INFORMS Journal on Computing*, *29*(3), 405–421.

Najafabadi, M. K., Mahrin, M. N., Chuprat, S., & Sarkan, H. M. (2017). Improving the accuracy of collaborative filtering recommendations using clustering and association rules mining on implicit data. *Computers in Human Behavior*, *67*, 113–128.

Nilashi, M., Bagherifard, K., Rahmani, M., & Rafe, V. (2017). A recommender system for tourism industry using cluster ensemble and prediction machine learning techniques. *Computers &*

Industrial Engineering, *109*, 357–368.

Panchal, G., & Panchal, M. (2014). Review on methods of selecting number of hidden nodes in artificial neural network. *International Journal of Computer Science and Mobile Computing*, *3*(11), 455–464.

Panniello, U., Tuzhilin, A., & Gorgoglione, M. (2014). Comparing context-aware recommender systems in terms of accuracy and diversity. *User Modeling and User-Adapted Interaction*, *24*(1–2), 35–65.

Park, Y., Oh, J., & Yu, H. (2017). RecTime: Real-Time recommender system for online broadcasting. *Information Sciences*, *409*, 1–16.

Patra, B. K., Launonen, R., Ollikainen, V., & Nandi, S. (2015). A new similarity measure using Bhattacharyya coefficient for collaborative filtering in sparse data. *Knowledge-Based Systems*, *82*, 163–177.

Salah, A., Rogovschi, N., & Nadif, M. (2016). A dynamic collaborative filtering system via a weighted clustering approach. *Neurocomputing*, *175*, 206–215.

Scrapehero. (2018). How Many Products Does Amazon Sell? – January 2018. Retrieved from https://www.scrapehero.com/many-products-amazon-sell-january-2018/

Shi, Y. (2014). An improved collaborative filtering recommendation method based on timestamp. In *Advanced Communication Technology (ICACT), 2014 16th International Conference on* (pp. 784–788).

Singh, S., Bag, S., & Jenamani, M. (2015). Relative Similarity Based Approach for Improving Aggregate Recommendation Diversity. In *In Proc. 12th Ieee Indicon Conf.*

Smith, B., & Linden, G. (2017). Two Decades of Recommender Systems at Amazon. com. *IEEE Internet Computing*, *21*(3), 12–18.

Son, J., & Kim, S. B. (2017). Content-based filtering for recommendation systems using multiattribute networks. *Expert Systems with Applications*, *89*, 404–412.

Tang, J., Deng, C., & Huang, G.-B. (2016). Extreme learning machine for multilayer perceptron. *IEEE Transactions on Neural Networks and Learning Systems*, *27*(4), 809–821.

Tran, T., Phung, D., & Venkatesh, S. (2016). Collaborative filtering via sparse Markov random fields. *Information Sciences*, *369*, 221–237.

Wang, C.-H. (2015). Using quality function deployment to conduct vendor assessment and supplier recommendation for business-intelligence systems. *Computers & Industrial Engineering*, *84*, 24–31.

Wu, H., Yue, K., Liu, X., Pei, Y., & Li, B. (2015). Context-Aware Recommendation via Graph-Based Contextual Modeling and Postfiltering. *International Journal of Distributed Sensor Networks*, *501*, 613612.

Wu, J., Chang, J., Cao, Q., & Liang, C. (2018). A trust propagation and collaborative filtering based method for incomplete information in social network group decision making with type-

2 linguistic trust. *Computers & Industrial Engineering*.

Yang, B., Lei, Y., Liu, J., & Li, W. (2017). Social collaborative filtering by trust. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *39*(8), 1633–1647.

Yin, X. F., Fu, X., Ponnambalam, L., & Goh, R. S. M. (2016). A k-means clustering for supply chain risk management with embedded network connectivity. *International Journal of Automation and Logistics*, *2*(1–2), 108–121.

Yoon, J., Seo, W., Coh, B.-Y., Song, I., & Lee, J.-M. (2017). Identifying product opportunities using collaborative filtering-based patent analysis. *Computers & Industrial Engineering*, *107*, 376–387.