RESEARCH & INNOVATION DIVISION
IJIST

A Demographic Fuzzy Similarity Computation Method for Recommender Systems

Madiha Khan*, Mubbashir Ayub Minhas, Arta Iftikhar
Department of Software Engineering (UET Taxila, Punjab, Pakistan).
***Correspondence.** Madiha Khan madiha.khan@students.uettaxila.edu.pk

Citation | Khan. M. Minhas. M. A, Iftikhar. A, "A Demographic Fuzzy Similarity Computation Method for Recommender Systems", IJIST, Vol 6, Issue 3, pp. 1168-1193, Aug 2024

Received | July 21, 2024 **Revised |** Aug 21, 2024 **Accepted |** Aug 23, 2024 **Published |** Aug 24, 2024.

In this allegedly never-ending stream of e-commerce, it is crucial to offer high-quality suggestions so that consumers can choose wisely from a wide range of picks. Recommender Systems (RS) has proven to be an essential instrument for improving sales of online vendors and enabling consumers with personalized product recommendations. Collaborative Filtering (CF), an extensively preferred approach for Recommender Systems, provides suggestions based on the ratings of users with similar interests. The primary operating component in CF is to measure similarity among items or users. Recommender Systems use a user-item matrix, which is often highly sparse and suffers from a cold start, ultimately leading to imprecise recommendations. Instead of relying merely on ratings that are uncertain and can be fake, we integrated the demographic information of users with CF to attain more precise predictions and recommendations in our work. To cope with the uncertainty factor in the recommendation process and to depict the physical world more realistically, we applied the Fuzzy set theory to users' ratings and demographic features. ML-100K and ML-latest-small datasets are used to evaluate the accuracy of the proposed similarity measure. Compared to the most advanced methods, our proposed demographic fuzzy similarity computation method exhibits considerable achievements in terms of MAE, RMSE, and coverage metrics on standard recommendation datasets, which we used for experimentation.

Keywords. Recommender Systems; Collaborative Filtering; Similarity Measures; Fuzzy Sets.

Introduction.

With every moment in this fast-paced world, massive amounts of information are added to the pile [1]. Accessing the relevant information appears to be difficult, tedious, and time-consuming amid this digital explosion [2] [3] [1]. Because of globalization, individuals all over the world connect and share information, services, and products [1]. The digital world has replaced previous standards by altering people's methods of living. Traditional businesses are increasingly shifting to E-commerce, and customers are anxious to explore what new products they offer. People now have a wide range of choices and get confused when bombarded with apparently immense information available [4]. Recommender Systems have emerged to address the problem of information overload, also known as the big data problem [5] [6]. These systems give automatic product recommendations that assist e-businesses in reaching their relevant consumers while consumers find the best fit conveniently and quickly [7] [8]. These systems are utilized across many sectors, including e-commerce, online articles and books, online broadcasting, movie sites, and so forth [9] [7] [10].

In a few decades, immense progress has been made in this domain [7]. Three filtering techniques are being used to process such massive data. These are Content-Based Filtering (CBF), Collaborating Filtering (CF), and Hybrid Filtering [7]. The CBF approach uses profile information about items and users to make predictions [11]. In this method, the system keeps track of the user's activities and content of items liked by users in the past to suggest similar items [7]. When a system recommends similar items based on the user's history, the user loses the opportunity to try something new [6] [1]. In reality, a user needs various options rather than merely homogenous possibilities [1]. In addition, extracting information from multimedia data in a user profile is complicated [12]. In CF, the system makes recommendations based on similar users or similar items liked by the users in the past [4] [13]. The Hybrid Filtering technique is a combination of Content-based Filtering and Collaborative filtering to utilize the key benefits of both [7].

Collaborative Filtering is the most popular technique that RS uses [4] [10] [14] because it is domain-independent and performs better in terms of accuracy [11]. The main idea behind CF is that people who have liked similar items before may like similar items in the future [7] [14]. CF is further sub-categorized into Model-based and Memory-based Collaborative Filtering [14]. A model is learned on a training dataset once, and there is no need to consult the entire dataset every time in model-based CF [15] [13]. These methods require training with many learning parameters [15]. The predictions are made based on the learned model [15]. Model-based approaches are quick as they respond in real time, and training can be done offline [13]. However, their accuracy is compromised by their inability to respond immediately to new users and ratings [10] [15]. In contrast, memory-based CF uses a user-item matrix and considers new users and ratings to find similar users [15] [13]. Model-based CF lags behind memory-based CF methods in terms of accuracy, due to which the latter is preferred by most of the Recommender Systems [14] [10] [4] like Amazon, YouTube, and Netflix [16].

Although rich literature is available on the accomplishments of Collaborative Filtering for Recommender Systems, some problems, such as cold start and data sparsity, still need to be solved [10] [5]. Recommending items to new users who have not rated any item is challenging because no rating history is available [3]. Similarly, in the case of a user who has rated very few items, it is not easy to make accurate predictions of items that match the user's interests [5]. Some users have similar interests but have no co-rated items due to cold start and sparsity problems [3]. So, there must be a way to calculate similarity even for items users have not commonly rated. During the registration process, users may be prompted to rate or specify their interests. Collecting user preferences early on may help address the cold start problem. However, this might potentially irritate users, particularly if they are asked to rate numerous items or respond to various questions during registration, ultimately resulting in user discontent, biassed

data, and less dynamic recommendations. Users provide demographic information during registration. The proposed approach utilizes user age and user locations in the form of zip codes, to calculate similarities. After calculating the demographic-based similarities, the system can initially recommend popular items to users in specific areas or age groups. The system undergoes constant changes as new users participate and provide ratings. These preferences are then incorporated into the system, enabling a gradual transition from recommendations based on demographics to personalized suggestions based on real user interaction or ratings. The seamless integration of this technology allows it to efficiently solve the cold start problem and adjust to user-specific preferences over time.

A user's demographic information can play a significant role in revealing his current concerns and interests. The user's age, gender, educational domain, work, and geographic region reveal much about the items the user could be interested in [17] [7]. For example, people of the same age group follow similar trends like similar tastes in music. Similarly, people in the same region experience similar cultures, traditions, and weather conditions. Some generous raters usually give high ratings even to the items they dislike; some users rate more items, whereas some do not like to rate them. Some users give honest ratings, whereas some give deceitful ratings [10]. This shows that different rating habits exist among users [10], and there is a probability that ratings are uncertain [2] and may not necessarily represent users' actual interests [11]. Since distinct rating habits exist among the users [10], it clearly shows that relying merely on user-based ratings for items is insufficient [18]. Human thinking and reasoning involve fuzziness [19], which led us to devise a method that depicts the physical world more realistically. We want our system to cope with unreliable and incomplete information. Realizing the uncertain nature of user ratings, we used fuzzy set theory, where a membership function provides a degree of similarity between a user and a fuzzy set. The proposed novel similarity method uses the Fuzzy set theory by applying the Gaussian Membership Function to ratings and demographics to compute similarity value. The rest of the paper is organized into further sections explaining current methods and related work in the Literature Review. The objectives and novelty statement are provided in the Objectives section. Materials and Methods presents our research methodology and tools used in detail. Results and Discussions is dedicated to experimental results and performance comparison.

Literature Review

A lot of advancements have been made in recommendation methods over time [7] [14]. Many similarity measures have evolved to achieve better recommendations but failed to achieve ultimate accuracy [10] [5] [7]. Initially, the similarity was computed by simply calculating distance in their rating values. Euclidean distance [20] measures the length between line segments, whereas Manhattan distance [21] is one norm of distance between two vectors. Cosine Similarity [22] can be computed by taking the Cosine of angles between rating vectors in n-dimensional space for co-rated items with the default rating set to average or zero. Setting the same default rating in a highly sparse user-item matrix seems unreasonable. The drawbacks of Cosine are eliminated by subtracting the corresponding user average from each co-rated pair in ACOS [22].

PCC [1] was introduced to calculate the similarity between co-rated items and return a value between -1 and 1. Extending PCC, the Constrained PCC (CPCC) [1], the weighted Pearson Correlation Coefficient (WPCC) [23], and the sigmoid function-based Pearson Correlation Coefficient (SPCC) [23] were proposed later. Jaccard Measure [24] considered non co-rated items but ignored the absolute values of item ratings given by users. Mean Squared Differences (MSD) [1], a variation of Mean Absolute Differences considered absolute ratings but failed to provide good coverage. Jaccard lessens the deficiencies in the coverage of MSD when joined as JMSD [25]. PIP [16] comprising of three factors, Proximity, Impact, and Popularity, focuses mainly on finding the difference more than common behavior and repeatedly penalizing the computed value when two ratings are not in agreement is unreasonable. Besides, absolute ratings were not considered, and the effects of not commonly rated items were ignored. NHSM [4] is

an improvement of PIP, which uses multiplication again and again, which deteriorates the similarity value, resulting in a small similarity value. The Modified Jaccard measure used in NHSM must be corrected, as the numerator and denominator are unequal. For example, user u_1 has rated all 5 items the same as rated by user u_2. The similarity should be 1, but here, the similarity calculated by Jaccard in NHSM is 5/(5×5) which gives 5/25, which cannot get equal to 1.

Bhattacharyya similarity (BCF) [26] computed divergence between two probability distributions for non co-rated items and used both local and global information to calculate similarity but failed to distinguish two items with the same similarity value rated by different numbers of users. Hellinger's Distance [5] used all user ratings to classify items into different classes, stating that the rating probability distribution of each item is more similar in the same class. SMD [9] used all ratings and calculated the similarity between two rating vectors based on discovering the latent differences between both vectors. HSMD [9] is a variation of SMD, but HSMD deals with the absolute ratings directly without binary conversion. The user Rating Preference Behavior (RPB) was used with an improved model of standard PCC to form IPWR [10]. It focuses mainly on co-rated items and users' rating preferences as a function of user average rating value and variance or standard deviation. The Triangle similarity [27] used the angle and lengths of the rating vectors. The Jaccard measure was combined to improve Triangle, making a new hybrid measure known as TMJ [27]. The obtained similarity is further complemented with the user rating preference (URP) to get Improved Triangle Similarity (ITR) [28]. Fuzzy Similarity Measure (FSR) [17] uses fuzzy logic theory and represents linguistic expressions mathematically, commonly in the form of a triangular or trapezoidal fuzzy number. Fuzzy Similarity Measure (FSR) [17] is based on MSD [1], Significance and Popularity. Trapezoidal fuzzy sets were used in FSR to cope with the uncertainty and relativity of user ratings to items and to improve accuracy. MFSR [17] is a further step to perform multi-level calculations. Researchers [29] introduced Fuzzy set theory in 1965; since then, different researchers have proposed various techniques with time. Yager [30] presented a fuzzy method using the users' preference information. A Fuzzy method for context-aware RS was proposed in 2006 [31], whereas the HU-FCF [18] method used demographics instead of user ratings. Considering the drawbacks, we aim to overcome the limitations of current similarity measures by proposing a demographic-based similarity measure that uses a fuzzy approach to enhance accuracy with less complexity.

Objectives.

The primary objectives of the proposed research are comprehensively outlined below, which encompass utilizing novel concepts and approaches to augment the accuracy, personalization, and overall effectiveness of recommendation systems.

- We developed a methodology that substantially enhances the accuracy and precision of recommendations in comparison to current methodologies.
- We have created a robust system that provides recommendations to new users who have no prior history of ratings.
- The system evaluates user behavior to enhance the relevancy and personalization of recommendations.
- We have developed a technique that efficiently handles uncertainty in user ratings and demographic information.
- We have implemented measures to ensure that the system gives precise and reliable recommendations for both co-rated and non co-rated situations.

This study presents the Demographic-based Fuzzy Similarity (DBFS) measure, a novel approach that combines demographic information with fuzzy logic to solve the cold start issue in recommender systems. DBFS, unlike traditional techniques, incorporates demographic

similarities in addition to user-item ratings to boost initial recommendations. This approach leads to a substantial improvement in accuracy and relevance. Additionally, by integrating Gaussian membership functions, the approach adeptly manages uncertainty in user ratings and demographics. The empirical results demonstrate that DBFS surpasses current measures, providing a resilient solution for producing top-notch recommendations in situations with sparse data.

Material and Methods.

The proposed research is based on memory-based CF and uses users' item rating history. Users' absolute ratings are user responses to items in a user-item matrix of $m \times n$, where 'm' represents the total number of users and 'n' is the total number of Items [26] [6]. Inherently, the matrix is sparse, with a small fraction of user ratings on items [13] [11]. In memory-based collaborative filtering (CF), the existing user-item ratings stored in the system are used directly to make predictions for new items [15]. This can be achieved by two methods, namely, user-based and item-based recommendation [15]. The choice between a user-based and an item-based recommendation approach has the most significant influence on the accuracy and efficiency of the recommender system [6]. In conventional recommender systems, where the number of users significantly exceeds the number of available items, item-based approaches are generally preferred due to their computational efficiency and the lower frequency of updates, but user-based approaches typically offer unique recommendations, potentially resulting in more satisfactory user experience [6]. User-based filtering is highly effective in situations when personalized recommendations are of utmost importance. We chose the User-based approach because it is most suitable when utilizing demographic information. This technique relies on user preferences rather than item features to make recommendations. On the other hand, the item-based approach may not provide the same degree of personalization as user-based filtering and can encounter difficulties when dealing with new items that lack sufficient rating data. In the literature review section, we highlighted the challenges that prevent accurate recommendations by identifying the significant problems in the Recommendation systems. To address the identified obstacles, let us examine the key aspects our research primarily focuses on.

- **Cold Start & Sparsity.** When a new user or new item is added, we do not have rating history or co-rated items, but we can use the demographic information of users, like age, gender, or location, to predict user interests.
- **User Behaviour.** We have observed that distinct rating habits exist among the users. Some users give generous ratings, some rate almost every item, some do not like to rate items, some give honest ratings, and some give fake ratings.
- **Co-rated Items.** Users with more commonly rated items are more similar, and their interests seem more common. To calculate similarity, we focused on what is similar among users instead of considering dissimilarities, which led us to consider co-rated items instead of not commonly rated items.
- **Demographics.** People of the same area, profession, or age group share common interests, leading us to use demographic information for recommendations. Moreover, when users have no co-rated items, they may have their demographics in common, which can help make predictions.

Analyzing the complex nature of recommendation systems, we divide the proposed method into two parts. Rating-based similarity and Demographic-based similarity.

A. Rating-Based Similarity.

Rating-based similarity calculates the similarity between users by using ratings. As we have discussed previously, user behavior is completely subjective. For example, user u_1 has rated 3 items only, and the user u_2 has rated 7 items. Another user, let us say u_3, has rated 100 items.

The co-rated items between user u_1 and user u_2 is 3 and between user $u_1$and user u_3 is also 3. The similarity between user u_1 and user u_2 is more credible than the similarity between user u_1 and user u_3. Jaccard Similarity calculates credibility[24], which takes user behavior into account by calculating the number of commonly rated items of the users from the total no of items rated by either. For user u and user v, the Jaccard Similarity is calculated as given in (1).

$$\text{Jacc}(u, v) = \frac{|I_u \cap I_v|}{|I_u \cup I_v|} \quad (1)$$

Here, I_u and I_v represent the items rated by user u and user v respectively. Instead of simply calculating the difference between absolute ratings and then summing it up to calculate a similarity value, we looked for how many items the user u_1 and user u_2 agreed with each other on commonly rated items. Let us suppose, on a rating scale of 1 to 5, a user u_1 has rated 4 and the user u_2 has rated 5 to an Item I_1, which shows that both have rated above the average and tend to like Item I_1 and their interests are in agreement. Similarly, if both users rate below the average, it indicates that they do not appear to have a preference for Item I_1, but they still have a mutual agreement. The value of the agreement is 1 only if;

$$(r_u = r_v) \text{ or } (r_u, r_v < r_{med}) \text{ or } (r_u, r_v < r_{med}) \text{ or}$$
$$(r_u = r_{med}, r_v = 2) \text{or } (r_u = 2, r_v = r_{med}) \text{or}$$
$$(r_u = r_{med}, r_v = 4) \text{or } (r_u = 4, r_v = r_{med});$$
$$\text{Otherwise} = 0.$$

Here, $r_{med} = 3$, the median of the rating whereas, r_u and r_v represent the ratings of user u and user v respectively. **Figure 1** depicts the rationale behind the conditions that determine agreement. The agreement for co-rated items is calculated by the total number of agreements divided by the number of co-rated items as given in (2).

$$\text{Agree}(u, v) = \frac{\#\text{ agreements}}{|I|} \quad (2)$$

Where $I = I_u \cap I_v$. For some co-rated items, users u and v may rate items with the same value. The value for the same ratings is 1 if $(r_u = r_v)$; $\text{Otherwise} = 0$. We calculated the exact ratings by dividing the total number of the exact same values of ratings by the number of co-rated items as given in (3).

$$\text{Exact}(u, v) = \frac{\#\text{ exactly same}}{|I|} \quad (3)$$

We calculated the number of remaining items where the rating values of user u and user v are not equal in (4).

$$\text{Rem}(u, v) = \frac{\#\text{ agreements} - \#\text{exactly same}}{|I|} \quad (4)$$

If a user gives an average rating value of 3 to an item on a rating scale of 1 to 5, it is unclear if the user is inclined to like the item or not. The total number of ratings nearest to r_{med} is represented as $\#Near$, which is the sum of the number of times any of the user rates with an average rating value r_{med} and the other user rates the nearest value to the average value. The number of items that is nearest to r_{med}, is 1 if;

$$(ru, rv < rmed) \text{ or } (ru, rv > rmed) \text{ or}$$
$$(ru = r_{med}, rv = 2) \text{ or } (ru = 2, rv = r_{med}) \text{ or}$$
$$(ru = r_{med}, rv = 4) \text{ or } (ru = 4, rv = r_{med}); \text{ Otherwise} = 0$$

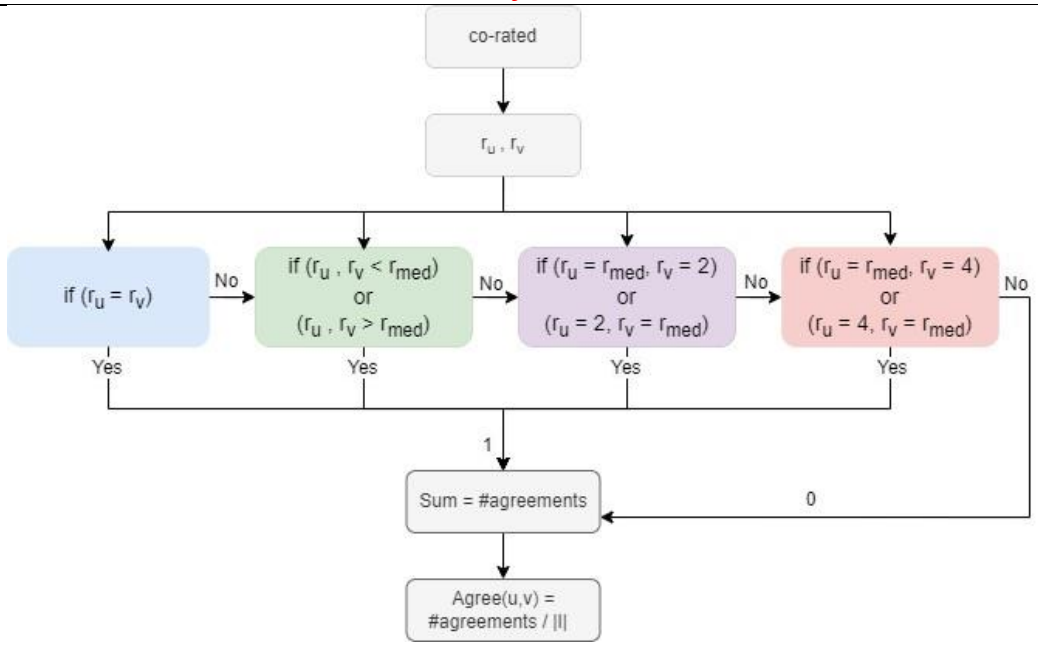

Figure 1. Flowchart of Agreements.

Fuzzy set theory is capable of handling inherently rough or inaccurate concepts [17]. The fuzziness in the fuzzy set is determined by its Membership function [14]. There are various membership functions, such as triangular, trapezoidal, and Gaussian [32]. The triangular membership function is used if there is a single peak value. The Trapezoidal Membership function is used if data remains constant after a certain value. Meanwhile, the Gaussian Membership function is preferred for normal distribution. To calculate the difference between the absolute ratings of users, we used the Gaussian Membership Function. It maps every element of the universe of discourse X to the interval [0,1] [2] [14]. The Gaussian Membership Function (GMF) is given as (5).

$$GMF(x) = e^{-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2}} \quad (5)$$

A Gaussian membership function is a mathematical function employed in fuzzy set theory to quantify the extent to which a specific input is a member of a fuzzy set. The degree of membership is a numerical number ranging from 0 to 1. Values that are close to the mean μ have high membership values, approaching 1. Conversely, values that are distant from the mean have low membership values, approaching 0. The Gaussian membership function can be customized to accurately represent various data distributions and membership criteria by modifying the mean (μ) and standard deviation (σ). The smoothness of the Gaussian function ensures that even minor changes in input values result in small changes in membership degrees. This characteristic is particularly beneficial in situations that need high sensitivity and precision. Triangular and trapezoidal functions are less complex and require less computational power, but they may not accurately represent the natural distribution of data as well as Gaussian functions. The Gaussian function is a continuous and differentiable mathematical function that smoothly spans membership degrees, allowing for an accurate representation of the inherent uncertainty and ambiguity present in real-world data. We conducted further fine-grained analysis to enhance the accuracy of our similarity calculations for absolute rating values. We divided them into the following four classes.

Class 1.

$$(r_u, r_v < r_{med}) \; or \; (r_u, r_v > r_{med})$$

Class 2.

$$(r_u = r_{med}, r_v = 2) \; or \; (r_u, = 2, r_v = r_{med})$$

Class 3.

$$(r_u = r_{med}, r_v = 4) \; or \; (r_u = 4, r_v = r_{med})$$

Class 4.

$$(r_u, r_v < r_{med}) or \; (r_u, r_v > r_{med}) \; or \; (r_u = r_{med}, r_v = 2) or$$
$$(r_u = 2, r_v = r_{med}) \; or \; (r_u = r_{med}, r_v = 4) \; or \; (r_u = 4, r_v = r_{med})$$

The difference between absolute ratings in each class is calculated (6).

$$Sim_{gaussian}(u,v) = \frac{\sum_{(i \epsilon I, j \epsilon P)} e^{-\frac{1}{2}\frac{(r_{ui}-r_{vi})^2}{\sigma_{uj}^2}}}{\#\, Near} \times |R| \quad (6)$$

Where $i \in R$ and $j \in P$, 'R' is the set of items in respective class, and 'P' is the set of all items rated by the user, r_{ui} and r_{vi} are the rating values of user u and user v for the item i respectively, μ_u is the mean of ratings of user u and σ_u is the standard deviation of ratings of user u. The similarity of ratings is calculated as given in (7).

$$Sim_{rating} = \frac{\sum_{c=1}^{4} Sim_{gaussian}(u,v)}{|I|} \quad (7)$$

Where 'c' represents the number of the class. The complete rating-based similarity is calculated by using (1), (2), (3), (4) and (7) as given below in (8).

$$Sim_{RB}(u,v) = Sim_{rating}(u,v) \times Rem(u,v) \times Exact(u,v) \times Agree(u,v) \times Jacc(u,v) \quad (8)$$

Demographic-Based Similarity.

For calculating demographic-based similarity, we have the locations and ages of users. Let's suppose $X = \{age_{u1}, age_{u2}, age_{u3}, \ldots, age_{um}\}$ is the set of ages of different users, where 'm' represents the number of users. The similarity between users' ages is determined by utilizing the Gaussian membership function (GMF) described in equation (5) to calculate the similarity, as stated in equation (9). The key benefit of using the Gaussian membership function is that it gives a smooth bell-shaped curve with non-zero values at all points.

$$Sim_{age}(u,v) = e^{-\frac{1}{2}\frac{(age_u - age_v)^2}{\sigma^2}} \quad (9)$$

$$\sigma = \sqrt{\frac{\sum_{i=1}^{m}(age_u - \mu)^2}{|m|}} \quad (10)$$

Where 'μ' is the average age of users, and 'σ' is the standard deviation of set X which can be calculated using (10). For example, $X = \{40,25,28,66,41,30,35,55\}$.

$$\mu = \frac{40+25+28+66+41+30+35+55}{8} = 40$$

$$\sigma = \sqrt{\frac{(0)^2+(-15)^2+(-12)^2+(-26)^2+(-1)^2+(-10)^2+(-5)^2+(15)^2}{8}} = 13.20$$

$$Sim_{age}(u_1,u_2) = e^{-\frac{1}{2}\frac{(40-25)^2}{(13.20)^2}} = 0.5243$$

Similarly,

$$Sim_{age}(u_1,u_1) = 1.0000 \quad ; \quad Sim_{age}(u_1,u_2) = 0.5243;$$
$$Sim_{age}(u_1,u_3) = 0.6615 \quad ; \quad Sim_{age}(u_1,u_4) = 0.1437;$$
$$Sim_{age}(u_1,u_5) = 0.9971 \quad ; \quad Sim_{age}(u_1,u_6) = 0.7505;$$
$$Sim_{age}(u_1,u_7) = 0.9307 \quad ; \quad Sim_{age}(u_1,u_8) = 0.5243$$

The similarity value of u_5 is the maximum, and the closest age to u_1 from the set X is 41. The next closest is u_7 with age 35. The closest the age to the user u, the highest the similarity value. It can be observed that the similarity value of u_1 and u_2 is 0.5243, which is the same as

the similarity between u_1 and u_8 because the difference between age 40 with age 25 is the same as the difference between age 40 and age 55. Similarly, we utilized the zip codes of users' locations to calculate the similarity of their geographical positions. The demographic-based similarity is calculated using (11).

$$Sim_{DB}(u,v) = \frac{Sim_{age}(u,v) + Sim_{loc}(u,v)}{2} \tag{11}$$

Rating Prediction.

Finally, the Demographic-based Fuzzy Similarity (DBFS) is calculated, which uses Sim_{RB} for ratings, and in case of cold start and new user, it uses ⟦Sim⟧ _DB for predictions. For the prediction of rating value, we used Resnick's Formula [28] as given in (12).

$$\hat{r}_{u,i} = \bar{r} + \frac{\sum_{v \epsilon NN} Sim_{RB} \times (r_{vi} - \bar{r}_v)}{\sum_{v \epsilon NN} |Sim_{RB}|} \tag{12}$$

Where $\hat{r}_{u,i}$ is the predicted value of the missing rating of item i of user u, NN denotes the number of nearest neighbors with similarity computed above the threshold K. The proposed methodology is illustrated in **Figure 2.**

Evaluation Metrics.

Evaluation metrics play a vital role in recommender systems by offering a quantitative method to evaluate and compare various algorithms, guaranteeing their efficacy in providing useful recommendations. Researchers utilize evaluation metrics to enhance scientific rigor. These metrics enable repeated testing of results and facilitate comparisons between studies. Some metrics, like as log-loss or AUC (Area Under the Curve), can be more challenging to grasp. Evaluation metrics such as ROC-AUC or PR-AUC (Precision-Recall Area Under the Curve) are useful when dealing with imbalanced datasets, but they may not offer distinct insights in situations when dealing with regression tasks. Therefore, the performance of the proposed similarity measure is evaluated using the following metrics. Mean Absolute Error (MAE), Root Mean Square Error (RMSE), Precision, Recall, F1 Score, and Coverage.

The Mean Absolute Error (MAE) [33] measures the average magnitude of errors in a set of predictions. The MAE is calculated as given in (13). MAE is commonly employed in regression tasks to quantify the proximity between predictions and actual data. MAE is less susceptible to the influence of outliers compared to RMSE because it does not calculate the squared errors.

$$MAE = \frac{1}{N}\sum_{i=1}^{N} |r_{u,i} - \hat{r}_{u,i}| \tag{13}$$

Where $r_{u,i}$ is the actual rating, $\hat{r}_{u,i}$ is the predicted rating for user u on item i and N represents the total number of predictions. The MAE gives an average error from 0 to the maximum value of the rating scale, with lower values indicating better accuracy. The Root Mean Square Error (RMSE) [33] gives the squared value of errors, giving more weight to more significant errors and making them more sensitive to outliers. The RMSE is the average of squared differences between a prediction and an actual observation and is calculated using (14). RMSE is valuable when there is a strong aversion to huge errors since it assigns greater importance to them.

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N} |r_{u,i} - \hat{r}_{u,i}|} \tag{14}$$

The Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) are utilized to assess the accuracy of the recommendation method. Precision [33] and Recall [33], in contrast to MAE and RMSE, approach the recommendation problem as a binary classification task

(relevant vs. non-relevant), which is more closely aligned with user satisfaction. An item is considered relevant if its rating exceeds the average rating. Precision is the model's capacity to identify only relevant data points, as determined by the formula in (15). Precision is essential when the potential consequences of false positive results are significant.

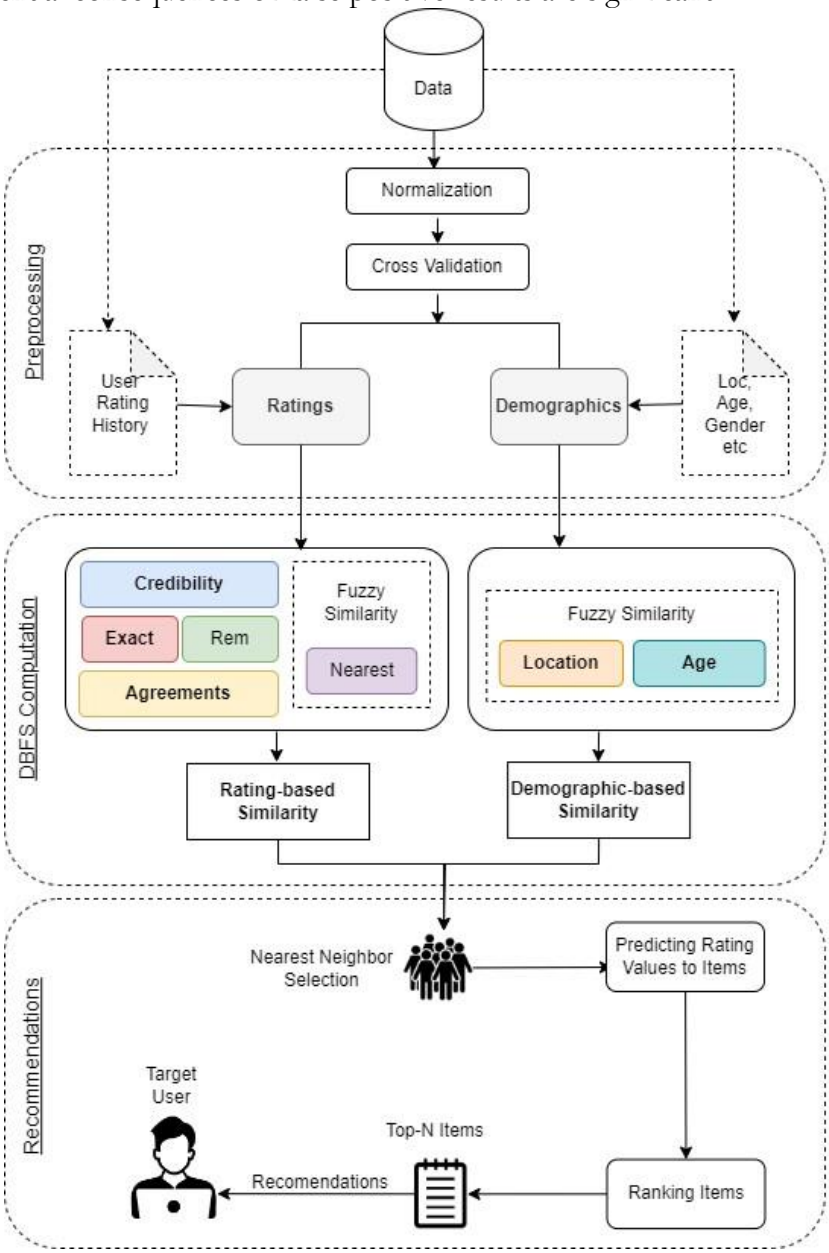

Figure 2. Flowchart of Methodology.

$$Precision = \frac{N_{rs}}{N_s} \tag{15}$$

Where N_{rs} is the number of relevant items recommended to the user, and N_s is the number of recommended items. Recall is a model's ability to identify all relevant cases within a data set, calculated as given in (16). Recall plays a crucial role when the consequences of missing essential information are costly.

$$Recall = \frac{N_{rs}}{N_r} \tag{16}$$

Where $\boldsymbol{N_r}$ is the number of relevant items. High Recall indicates that the model successfully identifies most of the relevant items. The F1-Measure [33] evaluates the model's

accuracy and predictive ability on a dataset by combining Precision and Recall to provide a balanced metric. The Precision, Recall, and F1-Measure computation is performed based on the top-N list for the target user [13]. F1-Measure is calculated using (17).

$$F1 - Measure = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (17)$$

Table 1. Rating Distribution of Users in ML-latest-small dataset.

Rating Values	ML-latest-small
0.5	1370
1.0	2811
1.5	1791
2.0	7551
2.5	5550
3.0	20047
3.5	13136
4.0	26818
4.5	8551
5.0	13211
Total	**100,836**

Table 2. Rating Distribution of Users in ML-100K dataset.

Rating Values	ML-100K
1.0	6110
2.0	11370
3.0	27145
4.0	34174
5.0	21201
Total	**100,000**

Coverage is the percentage of possible recommendations that the recommender system can provide [33][34]. In certain scenarios, it may be feasible to compromise on the extent of coverage to improve accuracy. This can be achieved by selectively delivering only very confident predictions to the user. However, it is crucial to prioritize achieving the highest possible coverage [34]. The coverage is calculated as given in (18).

$$Coverage = \frac{I_{ts}}{I_p} \quad (18)$$

Where I_{ts} represents the total number of items in test set whereas I_p represents a number of items with predictions. A model's increased coverage indicates its ability to predict a wider range of data points, enhancing its practicality in real-world scenarios.

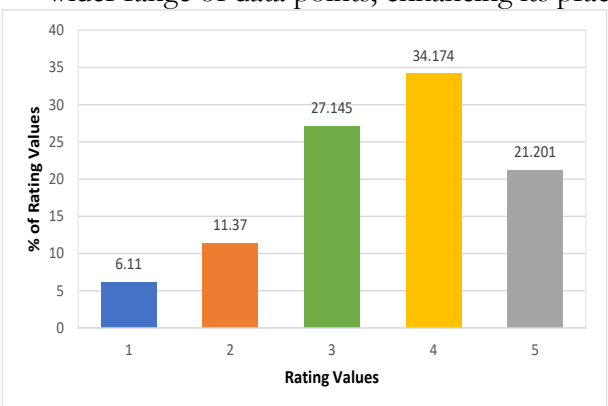

Figure 3. Percentage distribution of rating values in ML-100K dataset.

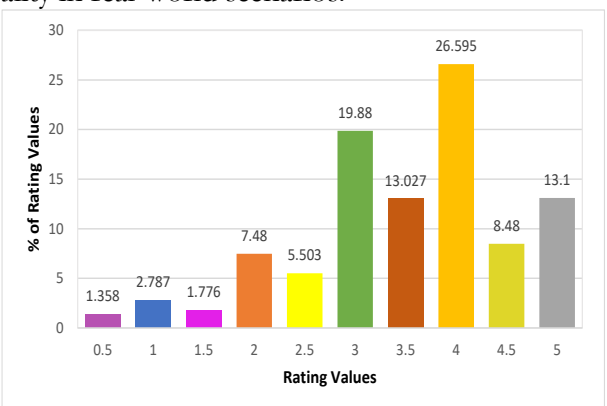

Figure 4. Percentage distribution of rating values in ML-latest-small dataset.

Result and Discussions.

Datasets Used.

The datasets we used for our experiments are MovieLens-100K (ML-100K) [34] and MovieLens-latest-small (ML-latest) [34]. ML-100K is comprised of 943 users and 10, 000 ratings over 1682 movies, with each user rating at least 20 movies on a rating scale of 1 to 5. The ML-100K dataset contains simple demographic information like age, gender, occupation, and user location in the form of zip codes. One of the reasons for choosing ML-100K is the availability of users' demographic information in this dataset. The sparseness in datasets refers to the proportion of missing values in the user-item matrix. The sparseness of the dataset is approximately 0.936953 which means that about 93.7% of the user-item matrix is empty or has missing values. This indicates that only about 6.3% of the possible ratings are provided in the dataset [9]. The dataset ML-latest-small consists of 610 users and 100836 ratings over 9742 movies where each user has rated a minimum of 20 movies on a rating scale of 0.5 to 5.0. The ML-latest-small dataset has a sparsity of approximately 0.98303, indicating that around 98.3% of the user-item matrix is either empty or contains missing values. It indicates that a mere 1.7% of the possible ratings are included in the dataset. The majority of users have given a rating of 4 in both datasets. The distribution of user rating values in ML-latest-small and ML-100K datasets is shown in **Table 1** and **Table 2.** The proportion of movies with a rating value of 4 in the ML-100K dataset is 34%, whereas in the ML-latest datasets, it is 26%. **Figure 3.** and **Figure 4** shows the percentage distribution of rating values in datasets.

K-Nearest Neighbors (KNN).

K-Nearest Neighbors is a widely used algorithm employed to generate personalized recommendations in recommender systems [6]. It is a collaborative filtering technique with several significant advantages. Firstly, it can provide a recommendation by listing the neighbors with the same tastes and who are more likely to have similar preferences. Secondly, it is computationally and space efficient, enabling it to handle large recommender systems. Lastly, it demonstrates remarkable stability in an online setting, even when new users and items are continuously added. Another advantage of this system is its ability to generate serendipitous recommendations, which can help users find unexpected yet very interesting items. KNN identifies the K most similar users to a target user by analyzing their rating patterns. Items that have been highly rated by similar users but have not been rated by the target user are recommended to the target user. The choice of K is crucial in neighborhood-based recommendation methods. The prediction accuracy follows a concave function as shown in Table 3, with low accuracy when using a small number of neighbors.

As K increases, more neighbors contribute and variance is averaged out, improving prediction accuracy. However, accuracy drops when too many neighbors are used, as strong local relations are diluted. The optimal value of K should be determined through cross-validation [35]. A neighborhood with 20 to 50 neighbors is suitable in most real-world scenarios, as it provides enough neighbors to balance out extremes [36][37]. The experiments are performed on seven similarity measures to compare user-based similarities for K Nearest Neighbors, where K = {5,10,15,20,25,30,35,40,45,50,55,60}. Changing the value of K results in different recommendations. The weighted average of the ratings provided by the neighbors can be used to calculate the predicted rating for an item. A detailed comparative analysis is performed where state-of-the-art similarity measures (Cosine, PCC, Jaccard, IPWR with variance, IPWR with SD, HSMD, and Hellinger's Distance) are evaluated for ML-100K and ML-latest datasets. To ensure uniformity across datasets and provide a comprehensive evaluation, we select a value of *K* for comparison [38]. The uniformity of the data facilitates the comparison of different analyses, making it possible to attribute differences in performance to the inherent qualities of the dataset rather than variances in algorithm design.

Table 3. Statistical Results of Mean Absolute Error (MAE) for ML-100K and ML-latest-small datasets.

MAE (ML-100K)

Similarity Measures	5	10	15	20	25	30	35	40	45	50	55	60
Cos	0.881	0.848	0.836	0.831	0.827	0.825	0.823	0.822	0.822	0.821	0.821	0.82
PCC	0.886	0.854	0.845	0.84	0.838	0.836	0.835	0.834	0.834	0.834	0.833	0.833
Jaccard	0.847	0.82	0.813	0.81	0.808	0.808	0.807	0.806	0.807	0.807	0.808	0.808
IPWR with SD	0.836	0.807	0.797	0.791	0.788	0.789	0.788	0.788	0.788	0.788	0.788	0.789
IPWR with Variance	0.858	0.833	0.824	0.82	0.818	0.817	0.816	0.816	0.815	0.815	0.814	0.814
Hellinger's Distance	0.82	0.795	0.789	0.787	0.787	0.788	0.789	0.79	0.791	0.792	0.793	0.794
HSMD	0.857	0.827	0.818	0.813	0.81	0.809	0.808	0.808	0.808	0.808	0.808	0.808
DBFS	0.777	0.759	0.754	0.753	0.753	0.753	0.753	0.754	0.754	0.754	0.755	0.756

MAE (ML-latest)

Similarity Measures	5	10	15	20	25	30	35	40	45	50	55	60
Cos	0.795	0.775	0.77	0.767	0.765	0.765	0.764	0.764	0.764	0.764	0.764	0.764
PCC	0.799	0.791	0.789	0.788	0.788	0.788	0.787	0.787	0.787	0.787	0.787	0.787
Jaccard	0.744	0.753	0.748	0.745	0.745	0.744	0.744	0.744	0.744	0.744	0.744	0.744
IPWR with SD	0.76	0.747	0.744	0.744	0.743	0.743	0.743	0.743	0.743	0.743	0.743	0.743
IPWR with Variance	0.786	0.775	0.772	0.771	0.771	0.771	0.771	0.771	0.771	0.77	0.77	0.77
Hellinger's Distance	0.759	0.75	0.75	0.75	0.752	0.753	0.753	0.754	0.754	0.755	0.755	0.755
HSMD	0.781	0.758	0.751	0.748	0.747	0.747	0.746	0.746	0.746	0.746	0.746	0.746
DBFS	0.72	0.715	0.714	0.713	0.713	0.713	0.713	0.713	0.713	0.713	0.713	0.714

Table 4. Statistical Results of Root Mean Square Error (RMSE) for ML-100K and ML-latest-small datasets.

RMSE (ML-100K)

Similarity Measures	5	10	15	20	25	30	35	40	45	50	55	60
Cos	1.109	1.064	1.048	1.041	1.037	1.034	1.032	1.031	1.03	1.029	1.028	1.028
PCC	1.116	1.074	1.062	1.056	1.053	1.051	1.05	1.049	1.048	1.048	1.048	1.047
Jaccard	1.068	1.033	1.023	1.018	1.016	1.016	1.015	1.015	1.015	1.015	1.015	1.015
IPWR with SD	1.057	1.016	1.003	0.997	0.995	0.993	0.993	0.991	0.991	0.991	0.992	0.992

IPWR with Variance	1.081	1.046	1.034	1.029	1.026	1.025	1.023	1.023	1.022	1.022	1.021	1.021
Hellinger's Distance	1.045	1.008	0.999	0.995	0.995	0.995	0.995	0.996	0.997	0.998	0.999	1
HSMD	1.079	1.037	1.025	1.019	1.016	1.015	1.014	1.013	1.013	1.014	1.014	1.014
DBFS	0.999	0.975	0.967	0.965	0.964	0.964	0.964	0.964	0.964	0.964	0.964	0.965

RMSE (ML-latest)

Similarity Measures	**5**	**10**	**15**	**20**	**25**	**30**	**35**	**40**	**45**	**50**	**55**	**60**
Cos	1.025	1.002	0.955	0.922	0.991	0.99	0.99	0.99	0.989	0.989	0.989	0.989
PCC	1.034	1.024	1.021	1.021	1.02	1.02	1.02	1.02	1.02	1.02	1.02	1.02
Jaccard	1	0.975	0.969	0.966	0.965	0.965	0.964	0.964	0.964	0.964	0.964	0.964
IPWR with SD	0.987	0.972	0.97	0.969	0.968	0.968	0.968	0.968	0.968	0.968	0.968	0.968
IPWR with Variance	1.016	1	1	0.999	0.999	0.999	0.999	0.999	0.999	0.999	0.999	0.999
Hellinger's Distance	0.987	0.975	0.974	0.975	0.976	0.977	0.978	0.978	0.979	0.979	0.98	0.98
HSMD	1.004	0.978	0.97	0.967	0.966	0.965	0.965	0.965	0.965	0.965	0.965	0.965
DBFS	0.951	0.944	0.943	0.942	0.942	0.942	0.942	0.942	0.942	0.942	0.942	0.942

Table 5. Statistical Results of Precision for ML-100K and ML-latest-small datasets.

Precision (ML-100K)

Similarity Measures	**5**	**10**	**15**	**20**	**25**	**30**	**35**	**40**	**45**	**50**	**55**	**60**
Cos	0.575	0.566	0.564	0.565	0.564	0.563	0.562	0.562	0.561	0.563	0.561	0.562
PCC	0.571	0.568	0.566	0.566	0.564	0.564	0.565	0.563	0.563	0.564	0.563	0.563
Jaccard	0.58	0.574	0.57	0.57	0.57	0.572	0.57	0.57	0.569	0.569	0.567	0.567
IPWR with SD	0.589	0.581	0.582	0.582	0.581	0.578	0.581	0.579	0.577	0.578	0.577	0.575
IPWR with Variance	0.575	0.569	0.564	0.564	0.564	0.562	0.561	0.559	0.561	0.561	0.562	0.563
Hellinger's Distance	0.595	0.594	0.589	0.59	0.589	0.583	0.583	0.584	0.58	0.578	0.577	0.578
HSMD	0.589	0.586	0.584	0.577	0.575	0.575	0.572	0.573	0.572	0.573	0.57	0.569
DBFS	0.589	0.587	0.587	0.586	0.586	0.586	0.586	0.587	0.587	0.586	0.586	0.588

Precision (ML-latest)

Similarity Measures	**5**	**10**	**15**	**20**	**25**	**30**	**35**	**40**	**45**	**50**	**55**	**60**

Cos	0.577	0.577	0.579	0.58	0.579	0.578	0.579	0.58	0.58	0.579	0.58	0.579
PCC	0.569	0.567	0.569	0.568	0.568	0.568	0.569	0.568	0.568	0.568	0.568	0.568
Jaccard	0.581	0.579	0.58	0.58	0.582	0.579	0.578	0.579	0.579	0.579	0.579	0.578
IPWR with SD	0.584	0.584	0.587	0.59	0.588	0.588	0.587	0.587	0.587	0.587	0.587	0.587
IPWR with Variance	0.581	0.58	0.581	0.583	0.582	0.579	0.58	0.58	0.579	0.58	0.58	0.579
Hellinger's Distance	0.585	0.587	0.59	0.589	0.589	0.587	0.587	0.585	0.584	0.584	0.583	0.583
HSMD	0.597	0.592	0.59	0.588	0.589	0.587	0.586	0.587	0.586	0.584	0.584	0.585
DBFS	0.578	0.575	0.576	0.575	0.575	0.575	0.575	0.575	0.575	0.575	0.575	0.575

Table 6. Statistical Results of Recall for ML-100K and ML-latest-small datasets.

Recall (ML-100K)

Similarity Measures	**5**	**10**	**15**	**20**	**25**	**30**	**35**	**40**	**45**	**50**	**55**	**60**
Cos	0.38	0.395	0.403	0.404	0.403	0.403	0.402	0.403	0.402	0.403	0.401	0.403
PCC	0.381	0.394	0.395	0.396	0.396	0.397	0.399	0.398	0.398	0.4	0.399	0.399
Jaccard	0.418	0.425	0.429	0.431	0.431	0.431	0.431	0.43	0.428	0.427	0.426	0.426
IPWR with SD	0.398	0.41	0.416	0.421	0.425	0.427	0.429	0.429	0.427	0.429	0.429	0.429
IPWR with Variance	0.393	0.405	0.406	0.409	0.408	0.408	0.406	0.409	0.409	0.409	0.409	0.412
Hellinger's Distance	0.402	0.424	0.43	0.434	0.434	0.434	0.436	0.435	0.433	0.431	0.433	0.432
HSMD	0.366	0.386	0.394	0.396	0.4	0.402	0.403	0.405	0.406	0.408	0.408	0.406
DBFS	0.491	0.497	0.498	0.499	0.498	0.497	0.496	0.497	0.495	0.494	0.493	0.494

Recall (ML-latest)

Similarity Measures	**5**	**10**	**15**	**20**	**25**	**30**	**35**	**40**	**45**	**50**	**55**	**60**
Cos	0.367	0.372	0.375	0.375	0.375	0.375	0.377	0.378	0.379	0.378	0.378	0.378
PCC	0.382	0.383	0.387	0.386	0.387	0.388	0.388	0.388	0.388	0.388	0.388	0.388
Jaccard	0.384	0.382	0.383	0.383	0.382	0.378	0.377	0.377	0.376	0.377	0.376	0.375
IPWR with SD	0.393	0.402	0.409	0.412	0.411	0.413	0.412	0.411	0.41	0.41	0.41	0.409
IPWR with Variance	0.373	0.379	0.38	0.382	0.382	0.381	0.381	0.381	0.381	0.382	0.381	0.381
Hellinger's Distance	0.4	0.401	0.4	0.396	0.395	0.392	0.391	0.39	0.388	0.387	0.386	0.385

HSMD	0.346	0.35	0.351	0.351	0.353	0.353	0.353	0.353	0.351	0.351	0.351	0.351
DBFS	0.454	0.458	0.458	0.456	0.455	0.455	0.454	0.454	0.454	0.454	0.454	0.454

Table 7. Statistical Results of F1-Measure for ML-100K and ML-latest-small datasets.

F1-Measure (ML-100K)

Similarity Measures	5	10	15	20	25	30	35	40	45	50	55	60
Cos	0.456	0.464	0.469	0.47	0.469	0.468	0.468	0.468	0.467	0.468	0.466	0.468
PCC	0.455	0.464	0.464	0.465	0.464	0.465	0.467	0.465	0.465	0.467	0.466	0.465
Jaccard	0.485	0.488	0.488	0.49	0.489	0.491	0.49	0.489	0.488	0.487	0.486	0.485
IPWR with SD	0.473	0.479	0.483	0.487	0.489	0.489	0.492	0.491	0.489	0.49	0.49	0.49
IPWR with Variance	0.465	0.472	0.47	0.472	0.472	0.472	0.469	0.471	0.472	0.472	0.472	0.474
Hellinger's Distance	0.478	0.493	0.495	0.498	0.498	0.496	0.497	0.497	0.493	0.492	0.493	0.493
HSMD	0.45	0.464	0.47	0.468	0.471	0.472	0.471	0.473	0.474	0.475	0.475	0.473
DBFS	0.534	0.536	0.537	0.537	0.536	0.536	0.535	0.537	0.536	0.534	0.534	0.535

F1-Measure (ML-latest)

Similarity Measures	5	10	15	20	25	30	35	40	45	50	55	60
Cos	0.448	0.452	0.455	0.455	0.455	0.455	0.457	0.458	0.458	0.457	0.458	0.457
PCC	0.456	0.456	0.46	0.459	0.459	0.46	0.46	0.46	0.46	0.46	0.46	0.461
Jaccard	0.46	0.458	0.46	0.46	0.46	0.456	0.455	0.455	0.455	0.455	0.455	0.454
IPWR with SD	0.469	0.475	0.482	0.485	0.484	0.485	0.484	0.483	0.482	0.483	0.482	0.482
IPWR with Variance	0.453	0.458	0.459	0.461	0.461	0.46	0.46	0.459	0.459	0.46	0.46	0.459
Hellinger's Distance	0.475	0.476	0.476	0.473	0.473	0.47	0.469	0.467	0.466	0.465	0.464	0.463
HSMD	0.436	0.438	0.438	0.438	0.44	0.439	0.439	0.439	0.438	0.437	0.437	0.438
DBFS	0.507	0.508	0.508	0.507	0.506	0.506	0.506	0.506	0.506	0.506	0.506	0.506

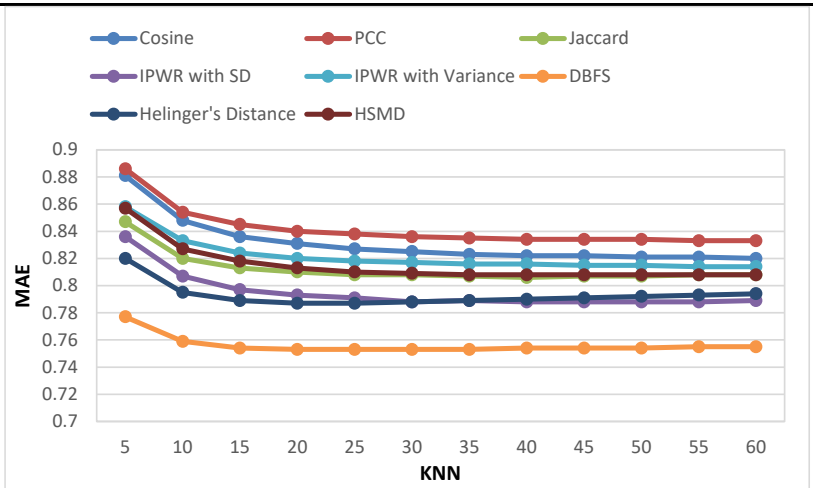

Figure 5 (a). Performance Comparison of Mean Absolute Error (MAE) for ML-100K dataset

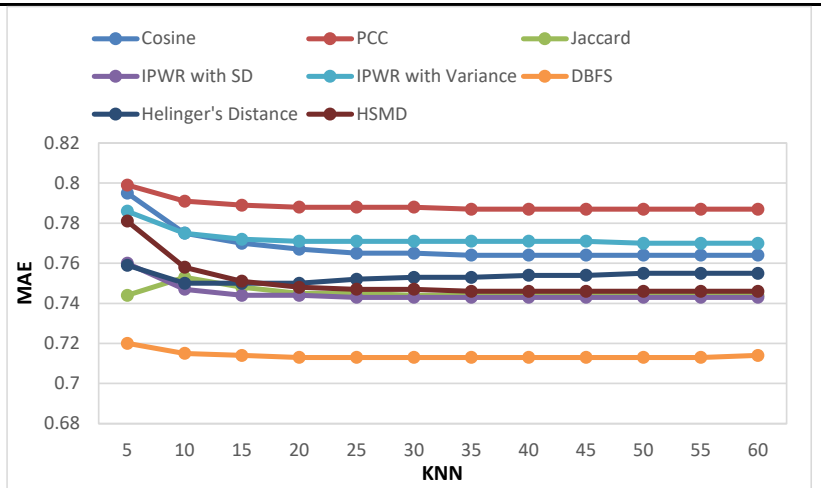

Figure 5 (b). Performance Comparison of Mean Absolute Error (MAE) for ML-latest-small dataset.

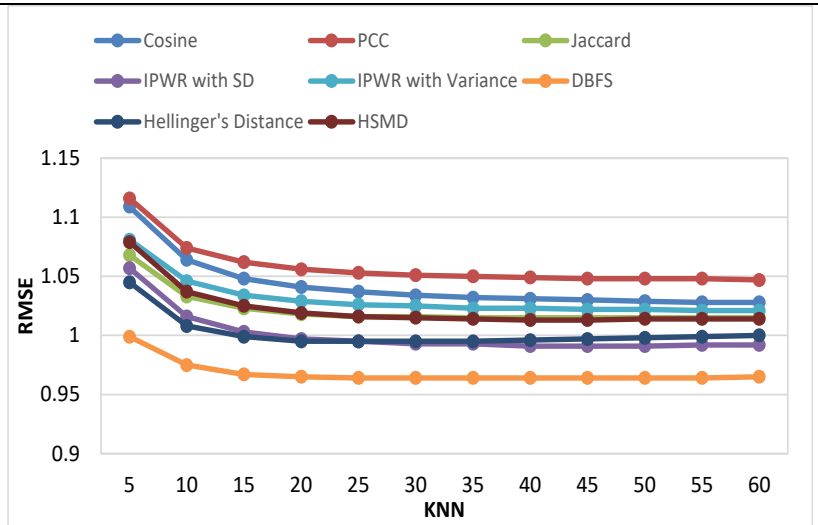

Figure 6 (a). Performance Comparison of Root Mean Square Error (RMSE) of ML-100K dataset.

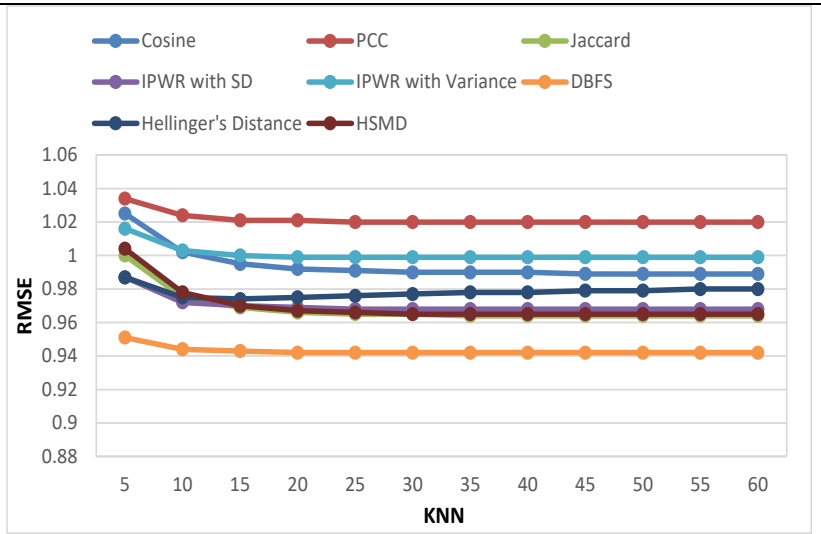

Figure 6 (b). Performance Comparison of Root Mean Square Error (RMSE) of ML-latest-small dataset.

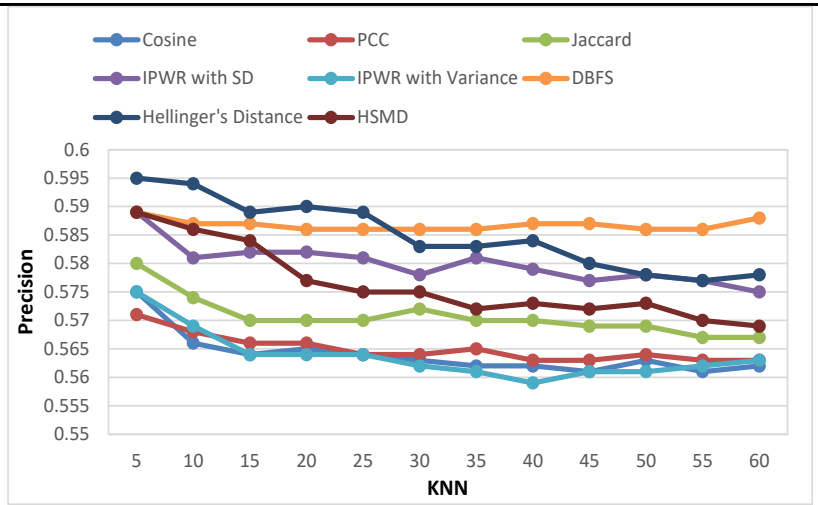

Figure 7 (a). Performance Comparison of Precision for ML-100K dataset.

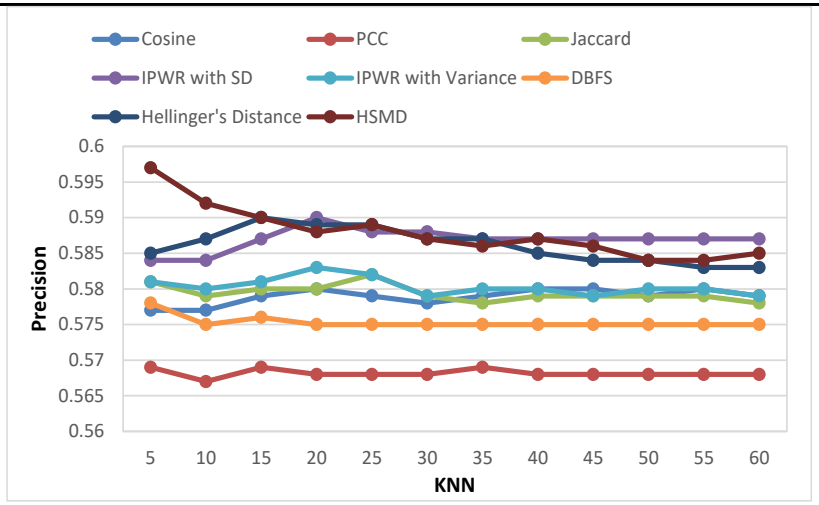

Figure 7 (b). Performance Comparison of Precision for ML-latest-small dataset

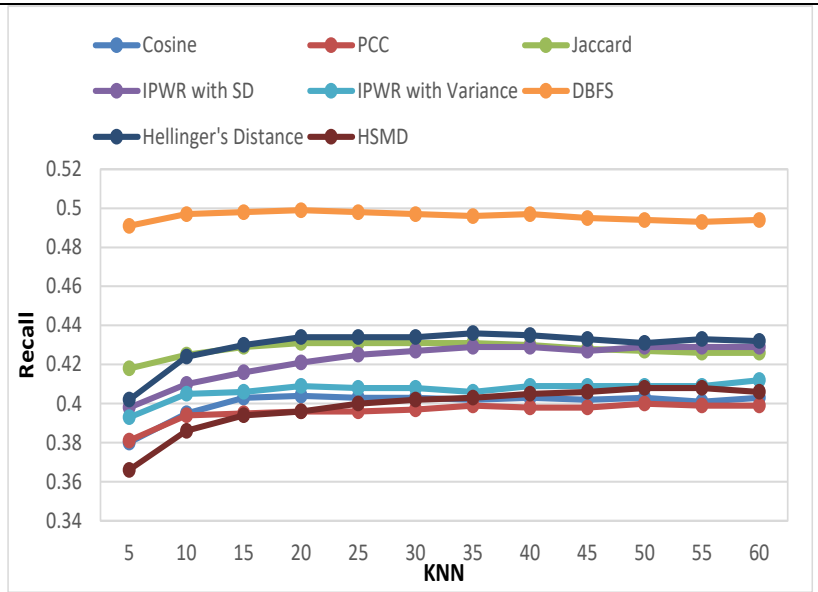

Figure 8 (a). Performance Comparison of Recall for ML-100K dataset.

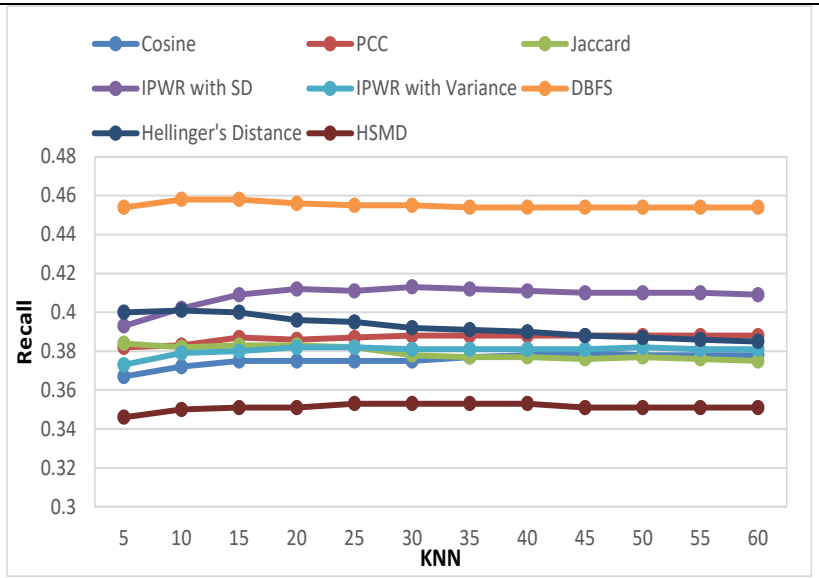

Figure 8 (b). Performance Comparison of Recall for ML-latest-small dataset.

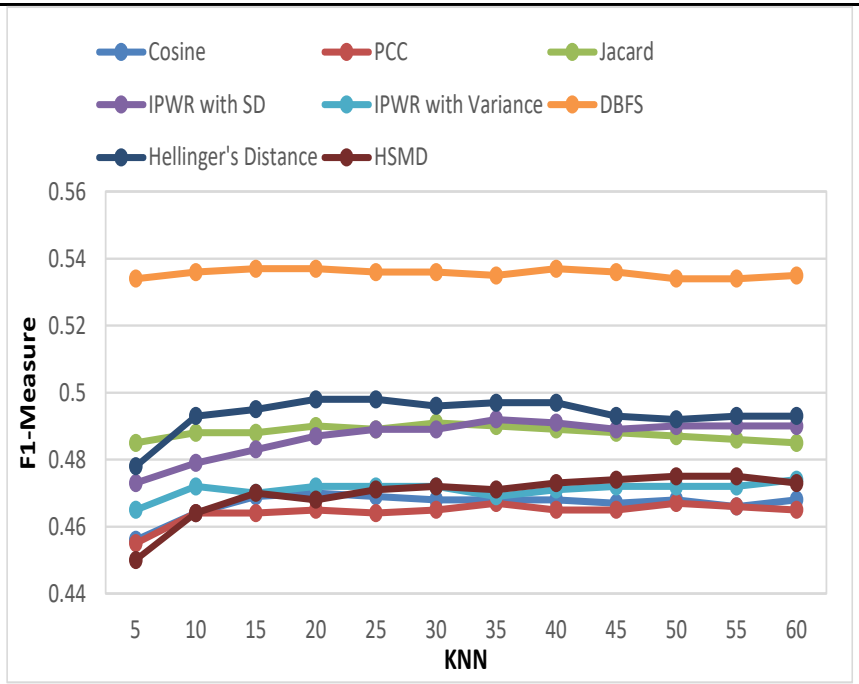

Figure 9 (a). Performance comparison of F1-Measure for ML-100K dataset.

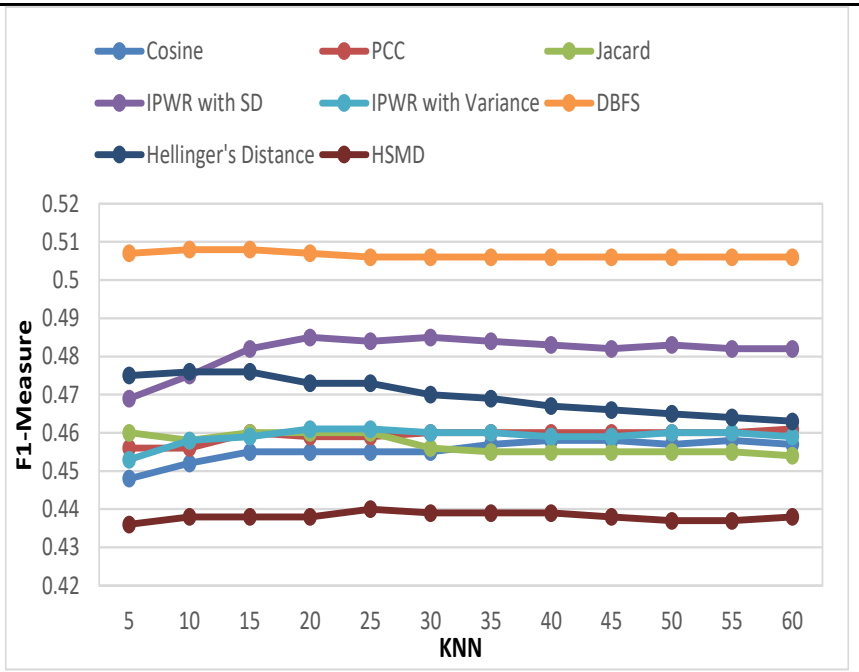

Figure 9 (b). Performance comparison of F1-Measure for ML-latest-small dataset.

Top-N Recommendations.

Top-N Recommendations involve selecting and presenting the highest N items from the entire collection, based on their expected relevance to the user. Users are more concerned with getting a list of valuable recommendations than with the accurate predicted rating of each item. Usually, the output is a list of the top-N recommendations rather than a numeric value. The optimal value for N in top-N recommender systems might differ based on several aspects, such as the unique application, user preferences, and the characteristics of the recommended items. A top-N list is created by first estimating ratings for all unrated items, sorting the results, and keeping the top N items [13]. N should be carefully selected to avoid issues with efficiency or accuracy. If N is too large, storing the neighborhood listings and forecast ratings will require a lot of memory. However, if you choose a small value for N, it might affect coverage which means some of the items will never be recommended [6]. Top-N suggestions are more desirable than predicted rating values due to their better alignment with user behavior and their ability to simplify decision-making. Furthermore, evaluation metrics such as Precision, Recall, and F1-Measure provide clear insights into the effectiveness of recommendations, making top-N lists a more practical and user-friendly choice for recommender systems.

Five-Fold Cross Validation.

k-fold cross-validation is a method for assessing a model's efficacy. It involves partitioning the data into numerous subsets to train and evaluate the model on these subsets [28]. The objective is to guarantee that the model generalizes effectively to data that has not been observed and does not exhibit overfitting or underfitting. In k-fold cross-validation, the data is partitioned into k-folds of equal size, with one set aside for testing and the remaining set for training [28]. It determines the performance metric (e.g., precision, recall, RMSE) for each k fold. The average of the k fold metrics is the ultimate performance metric. Five-fold cross-validation is implemented to compare the efficacy of the models and predict ratings where 20% of users are selected for testing while the remaining users are used for training. This is a frequently employed default option that achieves an appropriate equilibrium between variance and bias. This approach eliminates overfitting by ensuring that the model is tested on many subsets, resulting in a more reliable and generalizable evaluation. Additionally, it offers a comprehensive performance statistic by calculating the average of the results from all five iterations. This provides a more thorough insight into how the model performs across various data splits.

Performance Comparison.

Performance Comparison of Mean Absolute Error (MAE).

In order to assess the performance of DBFS compared to its competitors in terms of Mean Absolute Error (MAE), experimental data were acquired for various values of K-Nearest Neighbors, as depicted in **Table 3.** For the MovieLens-100K dataset with $K = 5$, the Mean Absolute Error (MAE) of the DBFS is 0.777. In comparison, the MAE of the IPWR with SD is 0.836, and the Hellinger's distance is 0.820. When K is increased to 30, the MAE reduces to 0.753 for DBFS. On the other hand, the MAE of IPWR with SD is 0.789, and the Hellinger's distance is 0.788. Similarly, when K is set to 60, the DBFS produces the lowest Mean Absolute Error (MAE) compared to other methods, as illustrated in **Figure 5 (a)** In the ML-latest dataset, when $K = 5$, the DBFS yields an MAE of 0.72. In contrast, the Jaccard has a mean absolute error (MAE) of 0.744, while the IPWR with SD has an MAE of 0.76. DBFS has optimal performance for values of K equal to 30 and 60. The DBFS approach demonstrates the lowest MAE compared to other methods in both the ML-100K and ML-latest datasets, as demonstrated in **Figure 5 (a)** and **Figure 5 (b).** When K is equal to 1, the prediction is based only on the rating given by the user who is most similar to the current user. If the rating of this person deviates significantly from the norm or does not accurately reflect the overall pattern, the prediction error can be substantial. When the value of K is set to 5, the impact of outliers is diminished, resulting in an increased probability of accurate predictions compared to when K is set to 1. With $K = 10$, the forecast now takes into account a wider spectrum of user opinions. The process of averaging helps to reduce the impact of noise and outliers, resulting in a potentially more precise prediction and a decrease in MAE. Although increasing the value of K can reduce the MAE to a certain extent, there is a maximum limit beyond which further increases in K will not result in any additional reduction in MAE. When the value of K grows very large,

the included neighbors may exhibit reduced similarity and contribute unwanted noise, which could potentially lead to an increase in MAE.

Performance Comparison of Root Mean Square Error (RMSE).

The performance of different similarity measures is compared with DBFS, particularly regarding RMSE, using the experimental data as shown in **Table 4.** The root mean square error (RMSE) for DBFS is 0.999 for $K = 5$ in the ML-100K dataset. In contrast, the RMSE for IPWR with SD is 1.057. The root mean square error (RMSE) of the DBFS reduces to 0.964 when $K = 30$. In contrast, the RMSE for the IPWR with SD is 0.993, and for the Pearson correlation coefficient (PCC), it is 1.051. At $K = 60$, the root mean square error (RMSE) for DBFS is 0.965, the lowest among all the other methods listed in **Table 4.** The DBFS performs remarkably, as illustrated in **Figure 6 (a).** Within the ML-latest-small dataset, DBFS performs better than other similarity measures across various nearest-neighbor values. When $K = 5$, the DBFS produces an RMSE value of 0.951. As the amount of K increases, the root mean square error (RMSE) drops to 0.942 when $K = 60$. DBFS has notable performance in RMSE, as illustrated in **Figure 6 (a)** and **Figure 6 (b).**

Performance Comparison of Precision.

The experimental results of precision, obtained using several nearest neighbors for similarity metrics, are presented in **Table 5.** for comparison. In the ML-100K dataset, the precision of DBFS is 0.589, which is equivalent to the precision of HSMD and IPWR. However, the precision is slightly higher for Hellinger's distance, which is 0.595. At $K = 30$, the precision of DBFS reaches its greatest value of 0.586, surpassing other similarity measures. In contrast, the precision of the competitors diminishes at $K = 30$. Among other approaches, the precision of DBFS reaches its maximum value at $K = 60$. The Precision performance of DBFS is illustrated in **Figure 7 (a).** The ML-latest dataset shows that the Precision of DBFS is 0.578 at $K = 5$ and is consistent at 0.575 from $K = 20$ to $K = 60$. The accuracy of HSMD is 0.597 at $K = 5$ and decreases to 0.590, which is equivalent to the precision of Hellinger's distance at $K = 20$. The performance of DBFS in Precision is shown in **Figure 7 (b).** While high precision is necessary, it is insufficient to comprehend the similarity measure's performance thoroughly. It should be combined with recall as in F1-Measure for a complete evaluation. The precision graph's fluctuations in Hellinger's distance, HSMD, and IPWR with SD and other measures in **Figure 7 (a)** and **Figure 7 (b)** indicate that the system's capacity to provide pertinent recommendations is dependent on the selection of K. A stable precision graph of DBFS, in which the precision stays mostly constant while K changes, indicates that the recommender system is reliable and not unduly affected by K selection. This is usually a good indication that the model is probably well-tuned and successfully balances the effects of neighbors, offering reliable performance for a range of K values.

Performance Comparison of Recall.

The performance of DBFS is evaluated against other advanced similarity measures, and the experimental findings are presented in **Table 6.** In the ML-100K dataset, the Recall value for DBFS is 0.491, which is the greatest compared to other methods. At $K = 20$, the Recall value for DBFS increases to 0.499. On the other hand, the Recall value for Hellinger's distance is 0.430. At $K = 60$, the Recall of DBFS is consistently high at 0.494. **Figure 8 (a)** illustrates that DBFS exhibits outstanding performance compared to other similarity metrics in terms of Recall. In the ML-latest dataset, where $K = 5$, the Recall for DBFS is 0.454, while the Recall for Hellinger's distance and IPWR with SD is 0.40 and 0.393, respectively. The DBFS provides the highest recall among all the nearest neighbors we have chosen for comparison. The notable achievement of DBFS in Recall is shown in **Figure 8 (b).**

Performance Comparison of F1-Measure.

The experimental results of the F1-Measure for compared similarity measures are presented in **Table 7.** In the ML-100K dataset, the F1-Measure value for DBFS at $K = 5$ is 0.534, which is much higher than the other values. The DBFS achieves the highest F1-Measure value at $K = 15$ and $K = 20$, reaching a value of 0.537. The F1-Measure has consistently high results compared to the other similarity measures, as depicted in **Figure 9 (a).** In the ML-latest dataset, the F1-Measure of DBFS with a value of $K = 30$ is 0.507. In contrast, the F1-Measure for Hellinger's distance is 0.478, and for Jaccard it is 0.485. The F1-Measure exhibits an increase to 0.508 when $K = 19$ and thereafter decreases to 0.506 when $K = 25$. The DBFS

yields a maximum value of 0.508 for the 10 and 15 nearest neighbors. The ML-100K dataset has an average F1-Measure of 0.535, whereas the ML-latest dataset has an average F1-Measure of 0.506. **Figure 9 (b)** displays the F1-Measure performance for the ML-latest dataset.

Performance Comparison of Coverage.

The proposed Demographic-based Fuzzy similarity measure outperforms the other similarity measures in terms of Mean Absolute Error (MAE), Root Mean Square Error (RMSE), F1 score, and other evaluation metrics. In addition, it attains 100% coverage. Consequently, the novel similarity measure not only enhances the accuracy of predictions but also guarantees the inclusion of all items in the dataset in the recommendations.

Table 8. Comparison of the proposed DBFS with other Similarity Measures in ML-100K and ML-latest-small datasets.

	ML-100K					ML-latest-small				
Similarity Measures	**MAE**	**RMSE**	**Precision**	**Recall**	**F1**	**MAE**	**RMSE**	**Precision**	**Recall**	**F1**
Cos	0.831	1.042	0.564	0.400	0.466	0.768	0.994	0.578	0.375	0.455
PCC	0.788	1.058	0.565	0.396	0.464	0.788	1.021	0.568	0.386	0.459
Jaccard	0.812	1.022	0.570	0.427	0.488	0.745	0.968	0.579	0.379	0.456
IPWR with SD	0.794	1.000	0.580	0.422	0.486	0.744	0.970	0.586	0.408	0.481
IPWR with Variance	0.821	1.031	0.563	0.406	0.471	0.772	1.000	0.580	0.380	0.459
Hellinger's Distance	0.792	1.000	0.585	0.429	0.493	0.753	0.978	0.586	0.392	0.469
HSMD	0.815	1.022	0.576	0.398	0.469	0.750	0.97	0.587	0.351	0.438
DBFS	0.756	0.968	0.568	0.495	0.535	0.713	0.943	0.575	0.455	0.506

Comparison of DBFS with Other Similarity Measures.

Table 8 presents the average values of MAE, RMSE, Precision, Recall, and F1-Measure for the ML-100K and ML-latest datasets. It compares the performance of the proposed DBFS measure with other similarity measures. The averages were calculated for twelve distinct values of the number of nearest neighbors, with K set to $K = \{5, 10, 15, \ldots, 60\}$. The MAE and RMSE specifically emphasize the prediction accuracy of each measure. **Table 8** demonstrates that the DBFS measure attains the lowest MAE and RMSE values, signifying a substantial decrease in prediction error in comparison to the other measures in both ML-100K and ML-latest datasets. This enhancement in accuracy is additionally demonstrated in **Figure 10 (a)** and **Figure 10 (b).** The average precision of DBFS is a bit low as illustrated in **Figure 10 (c)** but Recall is high compared to other methods, it means the system is very good at retrieving relevant items but also includes irrelevant items in its recommendations. Although precision evaluates the correctness of positive predictions, it does not consider the omission of relevant items. Consequently, it is frequently employed alongside with Recall to offer a more thorough assessment of overall performance. DBFS exhibits a high average Recall in both datasets compared to other similarity measures, as depicted in **Figure 10 (d).**

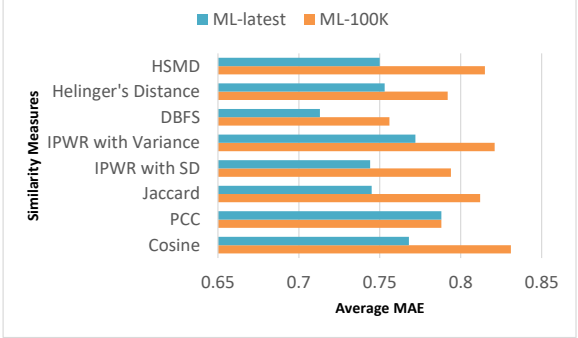

Figure 10 (a). Comparison of Average MAE of DBFS with other Similarity Measures.

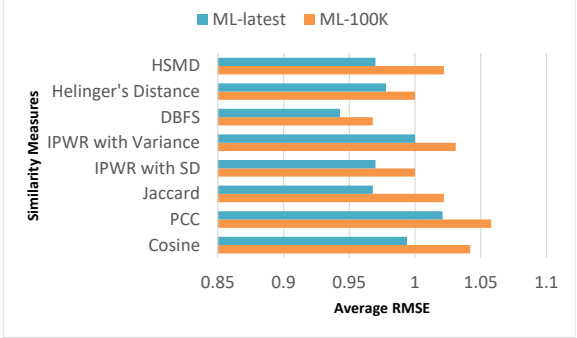

Figure 10 (b). Comparison of Average RMSE of DBFS with other Similarity Measures.

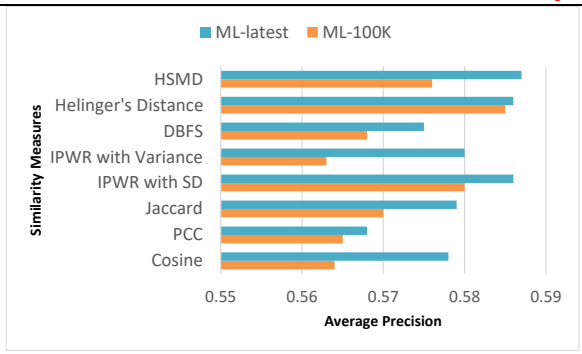

Figure 10 (c). Comparison of Average Precision of DBFS with other Similarity Measures.

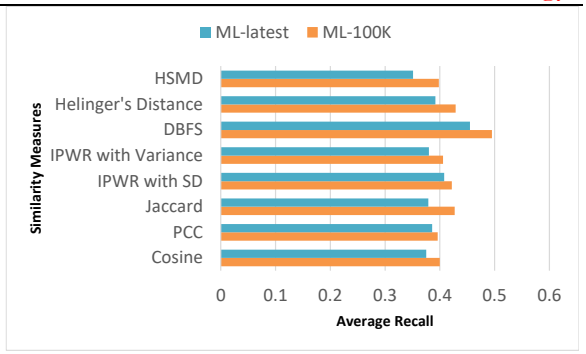

Figure 10 (d). Comparison of Average Recall of DBFS with other Similarity Measures.

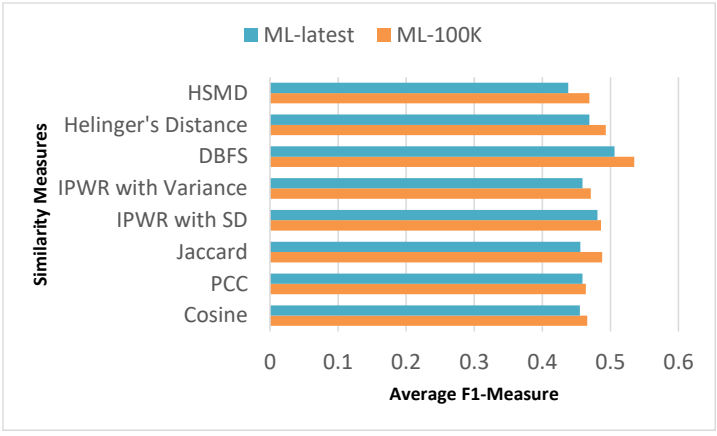

Figure 10 (e). Comparison of Average F1-Measure of DBFS with other Similarity Measures.

Personalization seeks to customize recommendations based on the distinct tastes and preferences of each user. As a result of personalization, the system may suggest a diverse range of objects that it deems to be potentially relevant to the user. While attempting to encompass a user's diverse interests and nuances in their profile, the system may suggest items that are only marginally related or items that the user may find intriguing but are not exactly what they actually want. This can increase the number of false positives, thereby lowering precision. For example, the system recommends songs similar to what the user has listened to before, ensuring that most genres and artists of interest are covered (high recall). However, it might also recommend slightly less relevant songs, reducing precision. The system throws in some completely new artists or genres that the user has never explored, aiming to pleasantly surprise them. While this can lead to discovering new favorites, it also increases the likelihood of irrelevant recommendations (low precision). Despite the irrelevant items, the number of relevant articles ensures the user stays informed about a wide range of topics, and the overall effectiveness (F1-Measure) of DBFS is high compared to other methods as illustrated in **Figure 10 (e).** Although precision is low, it is not so low as to drag down the F1-Measure of DBFS significantly. This implies that while there are false positives, they are not overwhelming compared to the true positives. The F1-Measure is higher in both ML-100K and ML-latest-small datasets, compared to other methods which means that despite the lower precision, the overall trade-off between precision and recall is more favorable in the proposed DBFS method. If the system is designed to promote diversity in recommendations, it might include a wider variety of items, some of which are less relevant. This can lead to lower precision but might improve overall user satisfaction by introducing users to a broader range of content.

Key Findings Discussion.

Compared to other metrics, such as HSMD, IPWR with SD, and Hellinger's Distance, the proposed DBFS significantly outperforms all. In the ML-100K dataset, the average MAE for the DBFS is 0.756, whereas HSMD, Hellinger's Distance, and IPWR with SD have average MAEs of 0.815, 0.792, and 0.794, respectively. The DBFS demonstrates a 4.9% improvement in MAE and a 2.7% improvement in RMSE over HSMD, a 4.5% improvement in MAE and a 3.29% improvement in RMSE over Hellinger's Distance, and a 4.7% improvement in MAE and a 3.2% improvement in RMSE over IPWR with SD. Additionally, DBFS improves Precision by 1.7% over HSMD, 1.03% over IPWR with SD, and 0.1% over Hellinger's Distance.

In the ML-latest dataset, the average MAE for the novel DBFS is 0.713, compared to 0.750 for HSMD, 0.753 for Hellinger's Distance, and 0.744 for IPWR with SD. The DBFS shows a 4.9% improvement in MAE over HSMD, a 5.3% improvement over Hellinger's Distance, and a 4.1% improvement over IPWR with SD. In terms of RMSE, DBFS improves by 2.7% over both HSMD and IPWR with SD, and by 3.5% over Hellinger's Distance. Precision improvements with DBFS are 1.8% over both Hellinger's Distance and IPWR with SD, and 1.54% over HSMD. Empirical results demonstrate that DBFS consistently achieves lower MAE and RMSE values across multiple datasets, indicating its superior accuracy and reliability. Specifically, DBFS provides significant improvements in precision, ensuring more accurate recommendations.

Conclusion.

The development of recommendation systems that are more efficient and user-centric is becoming increasingly important in a wide variety of applications, ranging from e-commerce to content streaming. The objective of this study is to enhance the accuracy of memory-based Collaborative Filtering by considering user rating patterns and including demographic data. The suggested method surpasses previous methods in terms of MAE, RMSE, Precision, Recall, and F1-Measure, demonstrating its efficacy in predicting user preferences and delivering useful recommendations. Nevertheless, the study is constrained by limitations such as the dependence on unavailable demographic data and the possibility of computational complexity. The research has the potential to be extended to provide accurate suggestions in diverse systems, and additional fuzzy membership functions might be investigated to achieve even higher precision. The suggested strategy improves the accuracy of predictions and achieves complete coverage, making it a scalable and effective approach ideal for a wide range of applications.

lAcknowledgment.. We want to express our sincere gratitude to the GroupLens Research Project for their invaluable work in accumulating the MovieLens datasets. Their dedicated efforts have provided a solid foundation for our research, allowing us to build upon their work without starting from scratch.

Author's Contribution. All the authors contributed equally.

Conflict of Interest. None of the authors of this research have reported any potential conflicts of interest that could be seen as conflicting with this work.

References.

[1] U. Shardanand and P. Maes, "Social information filtering. algorithms for automating "word of mouth"," pp. 210–217, 1995, doi. 10.1145/223904.223931.

[2] T. Kaya and C. Kaleli, "A novel top-n recommendation method for multi-criteria collaborative filtering", [Online]. Available. https.//www.sciencedirect.com/science/article/abs/pii/S0957417422001750?via%3Dihub

[3] "A trust-aware recommendation method based on Pareto dominance and confidence concepts", [Online]. Available. https.//www.sciencedirect.com/science/article/abs/pii/S0950705116304208?via%3Dihub

[4] H. Liu, Z. Hu, A. Mian, H. Tian, and X. Zhu, "A new user similarity model to improve the accuracy of collaborative filtering," Knowledge-Based Syst., vol. 56, pp. 156–166, Jan. 2014, doi. 10.1016/J.KNOSYS.2013.11.006.

[5] Junpeng Guo, J. Deng, X. Ran, Y. Wang, and H. Jin, "An efficient and accurate recommendation strategy using degree classification criteria for item-based collaborative filtering", [Online]. Available. https.//www.sciencedirect.com/science/article/abs/pii/S0957417420305807

[6] "A Comprehensive Survey of Neighborhood-based Recommendation Methods," SpringerLink, [Online]. Available. https.//link.springer.com/chapter/10.1007/978-0-387-85820-3_4

[7] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems. A survey of the state-of-the-art and possible extensions," IEEE Trans. Knowl. Data Eng., vol. 17, no. 6, pp. 734–749, Jun. 2005, doi. 10.1109/TKDE.2005.99.

[8] "Correcting noisy ratings in collaborative recommender systems", [Online]. Available. https.//www.sciencedirect.com/science/article/abs/pii/S0950705114004493?via%3Dihub

[9] A. A. Amer, Hassan I. Abdalla, and L. Nguyen, "Enhancing recommendation systems

performance using highly-effective similarity measures", [Online]. Available. https.//www.sciencedirect.com/science/article/abs/pii/S0950705121001052?via%3Dihub

[10] M. Ayub et al., "Modeling user rating preference behavior to improve the performance of the collaborative filtering based recommender systems," PLoS One, vol. 14, no. 8, 2019, doi. 10.1371/journal.pone.0220129.

[11] J. F. G. da S. N. N. de M. Junior, "Effects of Data Sparsity on Recommender Systems based on Collaborative Filtering", [Online]. Available. https.//ieeexplore.ieee.org/document/8489095

[12] M. J. Pazzani and D. Billsus, "Content-Based Recommendation Systems," pp. 325–326, 2007.

[13] A. M. Rashid, S. K. Lam, G. Karypis, and J. Riedl, "ClustKNN . A Highly Scalable Hybrid Model- & Memory-Based CF Algorithm Categories and Subject Descriptors," Webkdd '06, no. September, 2006.

[14] V. K. Mohammad Wasid, "A Particle Swarm Approach to Collaborative Filtering based Recommender Systems through Fuzzy Features", [Online]. Available. https.//www.sciencedirect.com/science/article/pii/S1877050915013757?via%3Dihub

[15] D. Valcarce, A. Landin, J. Parapar, and Á. Barreiro, "Collaborative filtering embeddings for memory-based recommender systems", [Online]. Available. https.//www.sciencedirect.com/science/article/abs/pii/S0952197619301599?via%3Dihub

[16] H. J. Ahn, "A new similarity measure for collaborative filtering to alleviate the new user cold-starting problem", [Online]. Available. https.//www.sciencedirect.com/science/article/abs/pii/S0020025507003751?via%3Dihub

[17] "MFSR. A novel multi-level fuzzy similarity measure for recommender systems", [Online]. Available. https.//www.sciencedirect.com/science/article/abs/pii/S0957417421004103?via%3Dihub

[18] L. H. Son, "HU-FCF. A hybrid user-based fuzzy collaborative filtering method in Recommender Systems", [Online]. Available. https.//www.sciencedirect.com/science/article/abs/pii/S0957417414002723?via%3Dihub

[19] I. Y. S. Michael Gr. Voskoglou, "Dealing with the Fuzziness of Human Reasoning," 2013, [Online]. Available. https.//arxiv.org/abs/1311.5355

[20] B. O'Neill, "Preface to the Revised Second Edition", [Online]. Available. https.//www.sciencedirect.com/science/article/pii/B9780120887354500031?via%3Dihub

[21] F. E. Szabo, "The linear algebra survival guide. illustrated with Mathematica," 2015, [Online]. Available. https.//www.sciencedirect.com/book/9780124095205/the-linear-algebra-survival-guide

[22] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation," Proc. 10th Int. Conf. World Wide Web, vol. 285–295, 2001.

[23] J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl, "An algorithmic framework for performing collaborative filtering," Proc. 22nd Annu. Int. ACM SIGIR Conf. Res. Dev. Inf. Retrieval, SIGIR 1999, pp. 230–237, 1999, doi. 10.1145/312624.312682.

[24] H. G.-M. Georgia Koutrika, Benjamin Bercovitz, "FlexRecs. expressing and combining flexible recommendations," pp. 745–758, [Online]. Available. https.//dl.acm.org/doi/10.1145/1559845.1559923

[25] "A new collaborative filtering metric that improves the behavior of recommender systems", [Online]. Available. https.//www.sciencedirect.com/science/article/abs/pii/S0950705110000444?via%3Dihub

[26] Bidyut Kr. Patra a, R. Launonen, V. Ollikainen, and S. Nandi, "A new similarity measure using Bhattacharyya coefficient for collaborative filtering in sparse data", [Online]. Available. https.//www.sciencedirect.com/science/article/abs/pii/S0950705115000830?via%3Dihub

[27] S. B. Sun et al., "Integrating Triangle and Jaccard similarities for recommendation," PLoS One, vol. 12, no. 8, p. e0183570, Aug. 2017, doi. 10.1371/JOURNAL.PONE.0183570.

[28] A. Iftikhar, M. A. Ghazanfar, M. Ayub, Z. Mehmood, and M. Maqsood, "An Improved Product Recommendation Method for Collaborative Filtering," IEEE Access, vol. 8, pp. 123841–123857, 2020, doi. 10.1109/ACCESS.2020.3005953.

[29] E. S. Awodey, "L. A. Zadeh. Similarity relations and fuzzy orderings. Information sciences," Cambridge Univ. Press, vol. 38, pp. 338–353, 1965.

[30] R. R. Yager, "Fuzzy logic methods in recommender systems", [Online]. Available. https.//www.sciencedirect.com/science/article/abs/pii/S0165011402002233?via%3Dihub

[31] H. Park, J. Yoo, and S. Cho, "A Context-Aware Music Recommendation System," pp. 970–971, 2006.

[32] Azene Zenebe and A. F. Norcio, "Representation, similarity measures and aggregation methods using fuzzy sets for content-based recommender systems", [Online]. Available. https.//www.sciencedirect.com/science/article/abs/pii/S0165011408001462?via%3Dihub

[33] J. T. R. I. & C. Jonathan L. Herlocker, Joseph A. Konstan, Loren G. Terveen, "Evaluating collaborative filtering recommender systems," vol. 22, no. 1, pp. 5–53, [Online]. Available. https.//dl.acm.org/doi/10.1145/963770.963772

[34] J. A. K. F. Maxwell Harper, "The MovieLens Datasets. History and Context," pp. 1–19, 2015, [Online]. Available. https.//dl.acm.org/doi/10.1145/2827872

[35] C. C. Aggarwal, Neighborhood-Based Collaborative Filtering. doi. 10.1007/978-3-319-29659-3.

[36] J. Herlocker, J. A. Konstan, and J. Riedl, "An empirical analysis of design choices in neighborhood-based collaborative filtering algorithms," Inf. Retr. Boston., vol. 5, no. 4, pp. 287–310, 2002, doi. 10.1023/A.1020443909834.

[37] S. D. Sondur, S. Nayak, and A. P. Chigadani, "Similarity Measures for Recommender Systems. A Comparative Study," Int. J. Sci. Res. Dev., vol. 2, no. 3, pp. 76–80, 2016, [Online]. Available. http.//www.journal4research.org/Article.php?manuscript=J4RV2I3036

[38] "An improved similarity measure for generalized fuzzy numbers and its application to fuzzy risk analysis", [Online]. Available. https.//www.sciencedirect.com/science/article/abs/pii/S1568494616305336?via%3Dihub