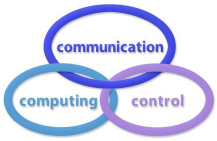
INTERNATIONAL JOURNAL OF COMPUTERS COMMUNICATIONS & CONTROL
Online ISSN 1841-9844, ISSN-L 1841-9836, Volume: 19, Issue: 4, Month: August, Year: 2024
Article Number: 5840, https://doi.org/10.15837/ijccc.2024.4.5840



CCC Publications

Design and Development of an Efficient Demographic-based Movie Recommender System using Hybrid Machine Learning Techniques

Vishal Paranjape, Neelu Nihalani, Nishchol Mishra

Vishal Paranjape*
Research Scholar, Department of Computer Applications
University Institute of Technology, Rajiv Gandhi Proudyogiki Vishwavidyalaya
Bhopal, Madhya Pradesh, India 462033
*Corresponding author: researchscholar482002@gmail.com

Neelu Nihalani
Professor, Department of Computer Applications
University Institute of Technology, Rajiv Gandhi Proudyogiki Vishwavidyalaya
Bhopal, Madhya Pradesh, India 462033
nihalani.neelu@gmail.com

Nishchol Mishra
Associate Professor, School of Information Technology
Rajiv Gandhi Proudyogiki Vishwavidyalaya
Bhopal, Madhya Pradesh, India 462033
nishchol@rgpv.ac.in

Abstract

Movie Recommender systems are frequently used in academics and industry to give users with relevant, engaging material based on their rating history. However, most traditional methods suffer from the cold-start problem, which is the initial lack of item ratings and data sparsity. The Hybrid Machine Learning (ML) technique is proposed for a movie recommendation system. Demographic data is collected from the Movie Lens dataset, and attributes are evaluated using the Attribute Analysis module. The Aquila Optimization Algorithm is used to select the best attributes, while Random Forest classifier is used for classification. Data is clustered using Fuzzy Probabilistic C-means Clustering Algorithm (FPCCA), and the Correspondence Index Assessment Phase (CIAP) uses Bhattacharyya Coefficient in Collaborative Filtering (BCCF) for similarity index calculation. The Outcomes gives the proposed method obtained low error, such as MAE has 0.44, RMSE has 0.63 compared with the baseline methods.

Keywords: Hybrid Machine Learning (ML) Technique, Recommendation system, Similarity Index, Attribute analysis, Demographic data, Fuzzy Probabilistic C-means Clustering Algorithm (FPCCA), Random Forest (RF)

1 Introduction

Recommender Systems (RS) play an essential role in assisting the advancement of e-commerce in numerous applications on the WWW. Recommender systems have grown significantly in recent years across a wide range of internet-based industries, including e-commerce, intelligent marketing, electronic book recommendation, music/movie recommendation, and the travel and tourism sector [1]. The most popular use of a movie recommender is to assist users in choosing movies from a sizable film library. Based on user similarity, this system can rank objects, display consumers' high-level items, and recommend movies [2]. The user's actions while watching the movies are considered implicitly in constructing a movie recommendation system. On the other hand, the user's past ratings or history are used explicitly to create movie recommendation systems [3].

Sentiment analysis can be particularly effective in enhancing the quality of recommendations when only low ratings data are available. It is because recommendation algorithms choose the things to recommend primarily based on user ratings. These ratings are typically quite low and insufficient [4-6]. The technology recommender systems use can be broadly divided into collaborative filtering and content [7]. The content-based model (CBR) analyses and investigates many aspects of the items to produce predictions. The collaborative model (CF) uses user or item similarity to anticipate what a user might like. CF methods use crowds' wisdom to suggest products based on users' similar tastes and preferences [8].

The most important phase in collaborative filtering is identifying the user's preferences comparable to those of other users. However, it has drawbacks, including sparsity, cold start, and scalability [9]. The user-item rating matrix, which may be further divided into user-based and item-based algorithms, is a prerequisite for the memory-based CF method [10]. In the user-based algorithm, the users with the highest similarity between the active user and the other users are chosen as the active user's neighbours. The suggested method then calculates the average of the neighbours' ratings for a particular item [11]. Finally, weights are assigned based on the relative similarity. There are various restrictions on memory-based CF. The rating matrix is extremely sparse because users and commodities are growing in quantity. However, users can only generate ratings for a limited number of commodities. The similarity values are inaccurate if the data are sparse [12, 13]. Collaborative filtering mixed with other methods is the most typical [14].

These strategies, known as cold-start problems caused by a new user's lack of ratings at the beginning of their system usage, have been solved. However, due to the system's failure to generate insightful suggestions, this issue might harm the recommender's effectiveness because both systems make assumptions based on user rating history [15-18]. Therefore, a different type of consumer feedback must be specifically acquired to recommend suggestions rather than ratings [19]. In this work, a demographic-based movie recommendation system is proposed to assess user demographic data that will be utilized to provide the best movie recommendations.

The key contribution of this article is enumerated as

- Data scarcity is one of the biggest problems in movie recommendation algorithms. The current strategy for resolving the issue of cold users and cold items does not consider demographic information when generating recommendations. In already published works, the prediction accuracy is influenced by cold start user, neighbor quantity, and neighbor quality.
- By incorporating demographic data such as user id, age, gender, and occupation, the system leverages an Attribute analysis module to assess attribute value distributions and identify invalid ranges, ensuring robust data integrity.
- The utilization of the Aquila Optimized Algorithm (AOA) for attribute selection and the Random Forest (RF) Classifier for optimized attribute selection further enhances the precision of movie recommendations.
- Additionally, the integration of the Fuzzy Probabilistic C-means Clustering Algorithm (FPCCA) contributes to minimizing sparsity and cold start problems by efficiently clustering unlabeled data.

- The innovative use of the Correspondence Index Assessment phase, employing the Bhattacharyya coefficient and a hybrid method for similarity index calculation, culminates in a final recommendation system that achieves optimal movie suggestions with high accuracy.

The rest of the article is constituted as Part I describes the introduction, Part II reveals the related works based on existing techniques, Part III depicts the proposed methodology of the framework, Part IV discusses the outcome of the Simulation process and Part V Conclude this research with future work.

2 Related Works

In this section, the recent papers related to the movie recommender system are listed below;

Choudhury et.al [20] proposed a trust matrix metric that combines user similarity and weighted trust propagation. However, in the cold-start situation, when new users or goods have little to no trust history, it could encounter challenges. This may lead to insufficient trust spreading, which in turn may result in less-than-ideal suggestions for these users or products. To generate the user or items recommendations, Deep matrix factorization (DMF), a deep learning-based collaborative filtering framework that successfully incorporates side information, is proposed by Yi et.al [21]. Implicit feedback embedding (IFE) is introduced for recommendation algorithms that frequently employ implicit feedback input. A DL-based model's tuning procedure is laborious and time-consuming. Therefore, automated hyper-parameter modification is a crucial problem. Bhalse et.al [22] described applying a singular value decomposition collaborative filtering method to a web-based movie recommendation system; this work addresses the sparsity problem and seeks to resolve it. Investigation of the evaluation parameter is not expanded in this approach

. Yassine et.al [23] described that recommendation systems aim to identify users' interests and suggest the products most likely to spark those interests. This study proposes a novel intelligent recommender system using the well-known unsupervised machine learning technique K-means clustering and collaborative filtering (CF). Advanced machine learning algorithms, deep learning models, and other approaches utilized in recommendation systems are not explored and improved in this approach.

Awan et.al [24] described a hybrid technique of Apache Spark and machine learning libraries to create a recommender engine for collaborative filtering, utilizing the alternating least squares model to identify the highest-rated films in a movie recommendation system. However, the ALS-related selection parameters might impact the performance of the RS. So that Sieves in RS cannot be extended for higher results when providing recommendations. Kiran et.al [25] presented an innovative hybrid recommender system that uses deep learning to replenish these gaps. Instead of focusing on the ranking or streaming cases, it covers the rating prediction issue of the recommender systems problem. Behera et.al [26] examined the weighted hybrid CF system by combining the restricted Boltzmann machine (RBM) with content Knearest neighbours (KNN). However Random recommendation is worse. To show the efficiency of the constructed hybrid sparrow clustered (HSC) recommender system by Sharma et.al [27], it is deployed to the MovieLens dataset. Due to the minimal interaction history in the MovieLens dataset, the HSC recommender system may experience cold start issues with new users or products.

Previous research has focused on solving the "cold start" problem, which is a typical obstacle in recommender systems. Many strategies, particularly for novel users or items, have been investigated to lessen the constraints brought about by inadequate user-item interaction data. Utilizing metadata or content attributes to provide suggestions without regard to user history, content-based approaches have become a popular alternative. These techniques, which rely on the intrinsic properties of objects and users, successfully handle the "cold start" issue. Hybrid recommender systems are a different tactic that leverages the advantages of both content-based and collaborative filtering methods. Although collaborative filtering is susceptible to "cold start" issues, content-based techniques that provide tailored recommendations based on item features complement it. In order to improve the recommendation process for users with little contact history, some research has also looked at the use of implicit feedback or demographic data. When combined, these tactics provide a more thorough method of addressing the "cold start" issue by providing a variety of solutions that may be tailored to suit various datasets

and scenarios. In spite of these developments, research is still being done to improve and develop new approaches for solving the "cold start" problem in recommender systems in a more sophisticated and efficient manner.

As a result, the movie recommendation is not suited, covers the prediction rating issue, the performance was impacted by selecting the parameters, consumes more and scalability issue occurs, and evaluation parameters were not enhanced. Hence there is a need to propose a novel recommendation system to overcome the issues in movie recommendation.

3 Efficient Demographic-Based Movie Recommender System Using Hybrid Machine Learning Techniques

A movie recommendation system, or a movie recommender system, uses machine learning to filter or estimate consumers' preferences for films based on their prior decisions and behaviour. One of movie recommendation systems' most significant challenges is data scarcity. It compromises the quality of recommendations. It occurs due to popular movies that most people do not rate to increase the quality of recommendations. The user or rating matrix is very sparse to find sufficient reliable similar users, and the system's accuracy is critical. The current approach to dealing with the problem of chilly users and cold objects does not consider demographic data when making recommendations. In existing works, cold start user, number of neighbours, and neighbour quality) affect the prediction accuracy.

In this work, user demographic data is used as input for the movie recommender system, thus known as the demographic-based movie recommender system. Users' demographic data are initially collected, like user id, gender, age, and occupation, for generating movie recommendations based on rating. Then a ternate attribute analysis module is employed, in which the types of demographic data, the distribution of attribute values across the dataset, and the validity of combining these attributes for recommendations are determined. Then the attribute is analyzed in terms of the Invalid Value Range. Then the data is optimized via Aquila Optimization Algorithm (AOA), which initializes the location for selecting the best attributes. By collecting the initial positions with the help of optimization, AOA categorizes the user items and movie items based on the fitness function. RF Classifier classifies the optimized attribute, and a Fuzzy Probabilistic C-means Clustering Algorithm (FPCCA) is employed to cluster the data. It utilizes improved probabilistic fuzzy C-means clustering and uses the membership and typicality matrix with probability values to cluster the unlabeled data for clustering the user profiles. Then a correspondence Index Assessment phase is utilized to calculate the similarity index of the movies by the Bhattacharyya coefficient in Collaborative Filtering (BCCF), which uses all ratings of each pair of users instead of using co-rated items. The BCCF uses the numerical values of all ratings a pair of users give, not only those given on similar products. The BCCF calculates the ultimate similarity value by combining local and global similarity, which minimizes sparsity & cold start problems, and the hybrid method recommends the movies with maximum accuracy in the recommendation system. The process flow of the proposed method is manifested in Figure 1 is given below

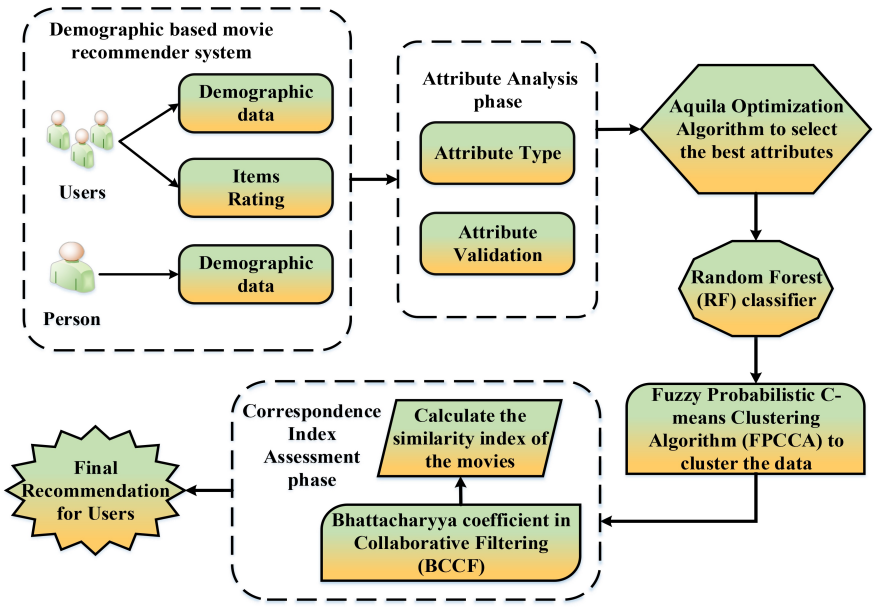


Figure 1: Process Flow of Demographic-based Movie Recommender System using Hybrid Machine Learning Techniques

3.1 Data Pre-processing

In this phase, the demographic-based movie recommender system and attribute analysis module are utilized to collect the user demographic data as input and validate the attribute recommendations.

3.1.1 Demographic-based movie recommender system

Utilizing user demographic information from their profiles (such as age, gender, location, etc.), the demographic-based recommender assumes that users who share the same demographic attribute(s) will rate goods similarly. This recommender gathers a neighborhood of users with comparable demographic characteristics to create newly recommended goods. A group of researchers applied a hybrid model-based approach to the movie domain to improve the recommendation and suggestion process. It categorized the movie genres based on user demographic attributes, including user age (kid, teenager, or adult), student status (yes or no), whether the user has children (yes or no), and gender (female or male). In the data input stage, ratings and demographic information about the other users are stored, along with information about the new target user (the user who needs recommendations). Users' demographic information is used in the similarity calculation stage to identify other users in the neighborhood who share the target user's demographic characteristics. After that, data was collected from the user demographic data, then fed into the analysis module to validate the attribute given in the below section

3.1.2 Attribute analysis module

All user information (demographics and item evaluations) is saved in the data source. The attribute analysis module examines the types of demographic attributes, how the values of the attributes are distributed, and whether it is legitimate to use these attributes as a basis for recommendations. With the help of the attribute analysis module, the type and value ranges of the demographic attributes used to recommend movies are identified. The frequency of each attribute value is then calculated, revealing the number of users that share that value. The module then verifies the attributes that can be applied to recommendations by examining the following conditions.

Invalid Value Range: It happens when certain demographic attribute ranges have low frequency, like the age attribute's "less than nine years old" range with only one user within it. As a result, the system cannot provide recommendations for new users who fall within this range using the age

attribute. Then the validated attributes created the data in pre-processing, and to select of the attributes will be discussed in the below section

3.2 Aquila Optimization Algorithm (AOA) [28]

This section utilizes the Aquila Optimization Algorithm(AOA) to select the best attributes in the below demographic data. The pre-processed data is given to the Aquila Optimizer (AO), a population-based optimization algorithm that draws inspiration from how Aquila naturally catches prey. In the Northern Hemisphere, it is one of the most well-liked birds. It belongs to the family "Accipitridae", which virtually all birds do. The Aquila is known to employ four distinct hunting tactics, each of which has its own set of traits, and the majority of them can swiftly and shrewdly switch between them.

The first technique, high soar with a vertical stoop, in which the Aquila climbs high above the ground, is employed for hunting birds in flight. Once it has located prey, the Aquila begins a long, low-angled glide, increasing its speed as the wings close farther. This strategy requires that Aquila have a height advantage above the target to succeed. Immediately before the encounter, the wings and tail are spread out to resemble a thunderclap, and the feet are pushed forward to capture the prey. The second technique, the contour flight with a brief glide attack, is Aquila's most popular strategy. In this technique, the flight rises slowly from the ground at a low altitude. A meticulous pursuit is made of the prey, whether flying or running. Ground squirrel, grouse, or seabird hunting are all excellent candidates for this tactic. The third technique is a low flight with a gradual downward attack. In this instance, the Aquila dives to the ground before attacking the prey one at a time. The Aquila picks its prey and attempts to enter by landing on its neck and back. Hunting slow prey, such as rattlesnakes, hedgehogs, foxes, tortoises, and any species lacking an escape mechanism, uses this strategy.

The fourth strategy involves the Aquila walking and catching its prey as it flies across the land. It is applied to eliminate the young of big prey animals (like deer or sheep) from the covered region. Finally, Aquila is one of the smartest and most skilled hunters—second only to humans. The approaches mentioned above mostly inspired the recommended AO algorithm. The following sections go on how the AO performs these procedures.

3.2.1 Configuration of the solution

The first step in the optimization method in AO is the creation of a population of candidate solutions (C), which is done stochastically between the upper (UB) and lower (LB) bounds of the challenge. The best-obtained data in each cycle is broadly identified as the ideal answer.

$$C = \begin{bmatrix} c_{1,1} & \cdots & c_{1,j} & c_{1,n-1} & c_{1,n} \\ c_{1,2} & \cdots & c_{2,j} & \cdots & c_{2,n} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ c_{M-1,1} & \cdots & c_{M-1,j} & \cdots & c_{M-1,n} \\ c_{M,1} & \cdots & c_{M,j} & c_{M,n-1} & c_{M,n} \end{bmatrix} \tag{1}$$

where C is for a set of current candidate solutions generated at random by Equation (2), C_i stands for the decision values (positions) of the ith solution, M stands for the overall population of candidate solutions, and n stands for the magnitude of the problem's dimension.

$$C_{i,j} = rand \times (UB_j - LB_j) + LB_j, i = 1, 2, ..., Mj = 1, 2, ..., n \tag{2}$$

LB_j stands for the j^{th} lower bound, UB_j stands for the j^{th} upper limit of the given problem, and the rand is the random integer.

3.2.2 Mathematical form of AO

The proposed AO approach displays each stage of the hunt while replicating Aquila's hunting behaviour. The four optimization stages for the proposed AO algorithm are thus classified as follows:

high soar with a vertical stoop, contour flying with a short glide attack, exploitation inside a converging search space by low flight with slow descent assault, and swooping by walk and grab prey. The AO algorithm can go from exploration phases to exploitation steps using its varied features. Exploration stages are carried out if $z \leq \left(\frac{2}{3}\right) * Z$; else, exploitation steps are finished. The following describes the AO mathematical model:

Expanded Exploration (A_1): The Aquila uses the first strategy (A1), which entails a high soar with a vertical stoop, to identify the prey area and select the optimal hunting area. To locate the prey, the AO soars over the search area and scans the area. Equation (3) is a mathematical representation of this phenomenon is given by

$$A_1(z+1) = A_{best}(z) \times \left(1 - \frac{z}{Z}\right) + (A_N(z) - A_{best}(z) * rand) \tag{3}$$

where represents the answer discovered by the initial search method A1 for the subsequent iteration of z. $A_{best}(z)$, which indicates the approximate location of the prey, is the best result up until the zth iteration. The extended search's iteration count is controlled by the expression $(1 - \frac{z}{Z})$ (exploration). Equation (4) can be used to get $A_N(z)$, representing the location mean value of the present solutions connected at the z^{th} iteration. The symbols "z" and "Z" represent the current iteration and the maximum number of iterations. 'rand' is a value selected randomly from 0 to 1.

$$A_N(z) = \frac{1}{M} \sum_{i=1}^{M} A_i(z), \forall j = 1, 2, ..., n \tag{4}$$

where M is the population size and n is the problem's dimension size.

Narrowed Exploration (A_2): The Aquila attacks using a second strategy (A2) after circling above the intended prey and preparing the terrain. This method is known as contour flying with a brief glide attack. To focus his attack, AO carefully researches the selected area of the target prey. Equation (5) provides the mathematical expression for this behaviour.

$$A_2(z+1) = A_{best}(z) \times Levy(d_s) + A_R(z) + (v - u) * rand \tag{5}$$

where $A_2(z+1)$ is the outcome of the subsequent n iteration of the second search technique. The levy flight distribution function, $Levy(d_s)$, is obtained using Equation (6), and the dimension space is ds. $A_R(z)$ is a solution selected at random in the range [1, M] at the ith iteration.

$$Levy(d_s) = x \times \frac{y \times \sigma}{|d|^{\frac{1}{\alpha}}} \tag{6}$$

where y and d are random values between 0 and 1, x is a constant with a value of 0.01, and σ is derived using Equation (7).

$$\sigma = \left(\frac{\Gamma(1+\alpha) \times sin\left(\frac{\pi\alpha}{2}\right)}{\Gamma\left(\frac{1+\alpha}{2}\right) \times \alpha \times 2^{\frac{\alpha-1}{2}}} \right) \tag{7}$$

Where α is a fixed-value constant having the value 1.5. Equation (5) uses the variables v and u to indicate the spiral form of the search.

$$v = t \times cos(\theta) \tag{8}$$

$$u = t \times sin(\theta) \tag{9}$$

Where

$$t = t_1 + G \times h_1 \tag{10}$$

$$\theta_1 = -\beta \times h_1 + \theta_1 \tag{11}$$

$$\theta_1 = \frac{3\pi}{2} \tag{12}$$

For a given number of search cycles, t_1 is between 1 and 20, while G is constant with a tiny value of 0.00565. Between 1 and the search space's length (n), "h_1" is an integer value, and "β" is a tiny constant term with a value of 0.005.

Expanded Exploitation (A_3): The third approach (A3) is employed when the precise position of the prey has been determined, and the Aquila is prepared to land and attack. With an initial attack, the Aquila descends vertically to gauge the prey's response. This tactic is called low flying with progressive descending assault. The AO approaches the prey and launches an assault using the target's designated region. This behaviour can be described mathematically using Equation (13), given by

$$A_3(z+1) = (A_{best}(z) - A_N(z)) \times \gamma - rand + ((UB - LB) \times rand + LB) \times \delta \tag{13}$$

Where $A_3(z+1)$ is the solution for z in the subsequent iteration of the third search technique. $A_{best}(z)$ represents the approximate location of the prey up until the ith iteration (the best-obtained solution) and is the mean value of the current solution at the zth iteration, which is evaluated by Equation (4). In this instance, the exploitation adjustment parameters are set to a low value (0.1). The given problem's upper bound is marked by UB, and LB denotes its lower bound.

Narrowed Exploitation (A_4): The fourth strategy (A4) has the Aquila approach the target and then attack it over land using stochastic motions. This method is called "walk and grab prey." Finally, AO attacks the prey in the last location. This behaviour can be mathematically described using Equation (14), given by

$$A_4(z+1) = Q_f \times A_{best}(z) - (L_1 \times A(z) \times rand) - L_2 Levy(d_s) + rand \times L_1 \tag{14}$$

where $A_4(z+1)$ denotes the fourth search method's solution for the subsequent iteration of z. Equation (15) determines the quality function Qf, which balances the search methods. Equation (16) is used to produce L1, which depicts several AO motions that are used to track the prey while in flight. L2 displays decreasing values from 2 to 0, which correspond to the AO's flight slope as it pursues the prey along the journey from the starting (1) to the ending (z) locations which is evaluated by Equation (17). At the third iteration, the current answer is $A(z)$

$$Q_f(z) = z^{2\times\frac{rand-1}{(1-Z)^2}} \tag{15}$$

$$L_1 = 2 \times rand - 1 \tag{16}$$

$$L_2 = 2 \times \left(1 - \frac{z}{Z}\right) \tag{17}$$

rand is a random number between 0 and 1, and $Qf(z)$ is the quality function value at the zth iteration. The current iteration and the maximum number of iterations are shown by the symbols z and Z, respectively. The levy flight distribution function, $Levy(d_s)$, is calculated using Equation (6). The Algorithm 1 for Aquila Optimization is given below

Algorithm 1: Aquila Optimization Algorithm

Step 1: Configuration Phase:
Step 2: Configure the Population A
Step 3: Configure the parameters of the AO(i.e.,β,δ, etc..)
Step 4: While(the end condition is not met)do
Step 5: Evaluate the fitness function values
Step 6: $A_{best(z)}$=Regulate the best attained solution according to the fitness values
Step 7: for(i=1,2,....,M)do
Step 8: Upgrade the mean value of the current solution $A_{N(z)}$
Step 9: Upgrade the u, v, L_1, L_2, Levy(d_s), etc.
Step 10: if $z \leq \left(\frac{2}{3}\right) * Z$ then
Step 11: if rand=0.5 then
Step 12: { **Phase 1:Expanded Exploration (A_1)** }
Step 13: Upgrade the current solution using Equation (3)
Step 14: if Fitness($A_1(z+1)$)<Fitness($A(z)$) then
Step 15: $A(z)=(A_1(z+1))$
Step 16: if Fitness ($A_1(z+1)$)< fitness ($A_{best(z)}$) then
Step 17: $A_{best(z)}=A_1(z+1)$
Step 18: end if
Step 19: end if
Step 20: else
Step 21: { **Phase 2: Narrowed Exploration (A_2)** }
Step 22: Upgrade the current solution using Equation (5)
Step 23: if Fitness($A_2(z+1)$)<Fitness($A(z)$) then
Step 24: $A(z)=(A_2(z+1))$
Step 25: if Fitness ($A_1(z+1)$)< fitness ($A_{best(z)}$) then
Step 26: $A_{best(z)}=A_1(z+1)$
Step 27: end if
Step 28: end if
Step 29: end if
Step 30: else
Step 31: if rand=0.5 then
Step 32: { **Phase 3:Expanded Exploitation (A_3)** }
Step 33: Upgrade the current solution using Equation (13)
Step 34: if Fitness($A_3(z+1)$)<Fitness($A(z)$) then
Step 35: $A(z)=(A_3(z+1))$
Step 36: if Fitness ($A_3(z+1)$)< fitness ($A_{best(z)}$) then
Step 37: $A_{best(z)}=A_3(z+1)$
Step 38: end if
Step 39: end if
Step 40: else
Step 41: { **Phase 4: Narrowed Exploration (A_4)** }
Step 42: Upgrade the current solution using Equation (14)
Step 43: if Fitness($A_4(z+1)$)<Fitness($A(z)$) then
Step 44: $A(z)=(A_4(z+1))$
Step 45: if Fitness ($A_4(z+1)$)< fitness ($A_{best(z)}$) then
Step 46: $A_{best(z)}=A_1(z+1)$
Step 47: end if
Step 48: end if
Step 49: end if
Step 50: end if
Step 51: end for
Step 52: end while
Step 53: return the best solution (A_{best})

3.3 Random Forest (RF) Classifier

The RF classifier is of the bagging variety and amalgamates different Decision Trees (DT). To improve predicted accuracy, it combines a variety of decision tree classifiers, trains them using different dataset subsamples, and averages the outcomes. The basic idea behind how a random forest classifier works is that it outperforms all other models when it combines independent and unrelated decision trees into a voted ensemble. There is randomness among these models as a consequence of this low correlation. The training set is obtained using the bootstrap approach. Not every DT is trimmed. Each DT offers a single vote when recognizing an unseen example, and the class with the most votes is the final identification outcome. The RF algorithm 2 works as follows is given below

Algorithm 2: Random Forest (RF) Classifier Algorithm

Step 1: Input data :Training dataset U, Attributes Set (F) and number of Attributes (S)
Step 2: Set Index: the number of DT X, the number of attributes 's' chosen for each DT training, and the feature coefficient of DTs (Gini, C4.5 or Entropy). Normally, $s = \log_2 S$
Step 3: Using the bootstrap approach, get the training subset U_x for the xth DT from U.
Step 4: Choose randomly a s-size attribute subset F_x for x th DT from F.
Step 5: Depending on the attribute value, expand each DT with the corresponding data and attributes subsets. To build the complete RF model, combine all of the DTs.
Step 6: Every DT offers a distinct conclusion, and the ultimate outcome can be produced by using the voting procedure.

The RF algorithm is resilient to noise and has a low computing cost. It is known as "the approach on behalf of ensemble learning technology" since it has performed well in numerous practical classification tasks. Contrary to many conventional machine learning algorithms, the RF algorithm achieves identification while considering the ranks of attributes. Therefore, the RF Classifier classified the optimized attributes for accurate selection of attributes and then clustered the data in the movies will be discussed in the below section

3.4 Fuzzy Probabilistic C-means Clustering Algorithm (FPCCA)

This section deals with clustering the data by the clustering algorithm, which utilizes the best-selected attributes given to the input of the FPCM. The proposed FPCM's parent model is the fuzzy C-means clustering. The FCM works with larger data sets and provides quantified cluster information. But the suggested FPCM does the same task more effectively. The user is first given the total number of clusters (U); if there are two clusters, for example, two clusters will be created. After that, the FCM chooses U random clients and treats them as cluster centres. Each cluster centre's and each customer's fuzzy membership Mki distance can be determined. The consumer will be associated with the cluster with the highest attribute values, depending on the membership function, which may be forty percent or sixty percent. The sum of the membership values for all the clusters equals 1. The member function is evaluated by using Equation (18)

$$M_{ki} = \frac{1}{\sum_{j=1}^{N} \left[\frac{\|a_k - u_i(b)\|_x}{\|a_k - u_i(b)\|_x} \right]^{\frac{2}{c-1}}} \tag{18}$$

With the aid of Eq. (19), new cluster centres are created utilizing this membership function.

$$u_i(b) = \frac{\sum_{k=1}^{P} (M_{Ki})^c a_k}{\sum_{k=1}^{P} (M_{Ki})^c} \tag{19}$$

The procedure will end if the objective function determined by Eq. (20) reaches its minimum value; otherwise, the membership function and cluster centre will keep changing. Due to the similar nature of the items within the cluster, the cluster quality increases when the goal function is lowered.

$$G_M = \sum_{k=1}^{P} \sum_{i=1}^{N} (M_{ki})^c \left\| a_k - u_i(b) \right\|_x^2 \tag{20}$$

where P represents the amount of data points, N represents the number of clusters, Mki represents the fuzzy membership performance of a_k in class i, and c represents the algorithm's degree of fuzziness $(1 \leq c < \infty)$. $\|a_k - u_i(b)\|_x$ is the Euclidean distance in terms of the number of attributes X between the kth data point and the ith cluster centre. The primary goal of FPCM is to enhance objective function Equation (20). The proposed FPCM algorithm's incremental approaches are thoroughly discussed in the steps below;

Stage 1: Measurement of the distance matrix: The user defines the number of clusters U in the clustering process, after which the centroid is randomly selected to be equal to the number of clusters U. The distance between data points and cluster centres is calculated using the Euclidean distance measure method. Equation (21) is used to compute the Euclidean distance given by

$$d(a_k, u_i(b)) = \left\| a_k - u_i(b) \right\|_x, 1 \leq i \leq N, 1 \leq k \leq N$$

$$\left\| a_k - u_i(b) \right\|_x = \sqrt{\sum_{X=1}^{F_a} \left[(a_k)_X - (u_i(b))_X \right]^2} \tag{21}$$

$\|a_k - u_i(b)\|_x$ is the Euclidean distance concerning the number of attributes X between the kth data point and the ith cluster centre. Here, Fa is the number of features or attributes.

Stage 2: Typicality matrix estimation: In this stage, Eq. (22) is used to compute the typicality matrix TM_{ki}. It was created by modifying the probabilistic C-means clustering method.

$$TM_{ki} = \frac{1}{1 + \left[\frac{\|a_k - u_i(b)\|_x}{\lambda_i}\right]^{\frac{1}{c-1}}}, 1 \le i \le N, 1 \le k \le P \tag{22}$$

Stage 3: Probability matrix computation: The probability matrix is computed once the distance matrix and typicality matrix have been determined. Let a be the data point, the cluster centres be $u_i(b) = (u_1, u_2, ..., u_n)$, and $\{d_i(a_k) : i = 1, ..., N\}$ its distance from the specified centres. PMki in Eq. (23) stands for the membership probability of a. A given cluster centre ui(b) and a specified data point a_k are connected by a Euclidean segment called $d_i(a_k)$.

$$PM_{Ki} = \frac{\prod_{j \ne i} d_j(a_k)}{\sum_{h=1}^{i} \prod_{j \ne h} d_j(a_k)}, i = 1, ..., N, k = 1, ..., P \tag{23}$$

Stage 4: Membership Matrix Evaluation: The probability matrix PM_{Ki} and the FCM membership matrix M_{ki}, used in Eq (18), are combined to create the proposed membership matrix. Equations (24) and (25) are used to calculate the proposed membership matrix MM_{Ki}. Based on the obtained maximum new membership value MM_{Ki}, clusters are created.

$$MM_{Ki} = M_{ki} \times PM_{Ki} \tag{24}$$

$$MM_{ki} = \frac{1}{\sum_{j=1}^{N} \left[\frac{\|a_k - u_i(b)\|_x}{\|a_k - u_i(b)\|_x}\right]^{\frac{2}{c-1}}} \times \frac{\prod_{j \ne i} d_j(a_k)}{\sum_{h=1}^{i} \prod_{j \ne h} d_j(a_k)} \tag{25}$$

Stage 5: Upgrade the centroid: The centroid is upgraded using Eq (26) when creating clusters.

$$u_i^l(b) = \frac{\sum_{k=1}^{P} (MM_{Ki} + TM_{Ki})\, a_k}{\sum_{K=1}^{p} (MM_{ki} + TM_{ki})}, 1 \le j \ge N \tag{26}$$

The following stages are repeated using the new centroid after each cluster's centroid has been upgraded, and the process continues through the calculation of the new centroid's update. The modified centroids of each cluster are updated repeatedly until they resemble one another. The proposed FPCM clustering's step-by-step processes are described in Algorithm 3 is given below

Algorithm 3: Fuzzy Probabilistic C-means Clustering Algorithm(FPCCA)

Input: best attributes
Output: Clustered data
Parameters:
N=Total number of clusters
$u_i(b)$=Set of Centroids
$d(a_k, u_i(b))$ = Distance of kth data point with respect to the ith centroid
PM_{ki}=Probability of $(a_k, u_i(b))$
TM_{ki}=Typicality Value of $(a_k, u_i(b))$
MM_{ki}=Membership function
start
Step 1: Construct the number of clusters N
Step 2:Select $u_i(b)$
Step 3:Evaluate the distance matrix $d(a_k, u_i(b))$ according to Eq. (21)
Step 4:Evaluate the typicality matrix TM_{ki} according to Eq. (22)
Step 5:Evaluate the probability matrix PM_{ki} according to Eq. (23)
Step 6: Create cluster based Membership matrix MM_{ki} according to Eq. (25)
Step 7:Upgrade $u_i(b)$ according to Eq. (26)
if all $u_i^l(b) = u_i^{l+1}(b)$
end
else
Back to step 3
end if
end

Finally, the data is clustered by Fuzzy Probabilistic C-means Clustering Algorithm(FPCCA), and it generates the matrix to calculate the similarity index will be discussed in the below section.

3.5 Correspondence Index Assessment Phase (CIAP)

This section discusses the Correspondence Index Assessment Phase(CIAP), which utilizes the Bhattacharyya Coefficient in Collaborative Filtering(BCCF) [29] to calculate the similarity index of the movie that has clustered data of the movie given as the input. Then the Bhattacharyya Coefficient in Collaborative Filtering(BCCF) is described by finding neighbours of an active user using the right similarity measure is a crucial step in the neighbourhood-based CF strategy. The number of ratings a single user makes is typically low in sparse data, and co-rated things are uncommon. The proposed method is effective when there are few or no co-rated products between two users. The similarity metric uses both local and worldwide rating data. Local information was calculated by utilizing the correlation of user evaluations. The global information is extracted regarding how closely two items are comparable. For this, the Bhattacharyya similarity measure is used. To calculate the distance (divergence) between two probability distributions, the Bhattacharyya measure, which is described, is used in the proposed metrics.

The Bhattacharyya measure is frequently used in signal processing, image processing, and pattern recognition research communities. The degree of similarity between two probability distributions is measured. Two density distributions over a continuous domain, q1(y) and q2(y), shall be used. Next, the Bhattacharyya Coefficient (BC) (similarity) between these densities is established by Equation (27) is given by

$$BC(q_1, q_2) = \int \sqrt{q_1(y)q_2(y)dy} \tag{27}$$

Over a discrete domain Y, the BC is stated as follows: Eq. (28) is given by

$$BC(q_1, q_2) = \sum_{y\epsilon Y} \sqrt{q_1(y)q_2(y)} \tag{28}$$

The provided rating data estimate the densities of q1(y) and q2(y). These densities can be estimated using histogram formulation. Let $\hat{q_i}$ and $\hat{q_j}$ be the two items' respective estimated discrete densities for i and j determined from rating information. Then, item i and item j's BC similarity is calculated as,

$$BC(i, j) = BC(\hat{q_i}, \hat{q_j}) = \sum_{r=1}^{n} \sqrt{(\hat{q_{ir}})(\hat{q_{jr}})} \tag{29}$$

Where n depicts the number of bins and $\hat{q_{ir}} = \frac{\#r}{\#i}$, where $\#i$ represents the number of users who rated item i, $\#r$ depicts the number of users who rated item i with rating value 'r' $\sum_{r=1}^{n} \hat{P_(ir)} = \sum_{r=1}^{n} \hat{P_{ir}} = 1$.

The BCCF provides numerical values for all of a pair of users' ratings, not just those on shared products. The BCCF calculates the ultimate similarity value by combining local and global similarity. Take IA and IB represent the items consumers scored A and B, respectively. There might not be a co-rated item ($I_A \bigcap I_B = \phi$) between A and B. The function of the BC coefficient between two rated items and the local similarity between the ratings on the pair of items determines the similarity between users A and B in the BCF measure of Equation (30).

$$S(A, B) = \sum_{i\epsilon I_A} \sum_{j\epsilon I_B} BC(i, j)loc(R_{Ai}, R_{Bj}) \tag{30}$$

The BC (i, j) offers global rating data for items i, j and loc(.) and determines whether the ratings are locally similar. As noted in the preceding subsection, BC can calculate the similarity between i and j, although no single user assesses both items. When two items are similar from a global standpoint, the BC (i, j) function increases the local similarity between user ratings on the individual items i and

j Eq. (30). The impact of local similarity between evaluations on the pair of items lessens, however, if items i and j are distinct to one another.

The local similarity is crucial and offers information about local users. The local similarity must provide a positive and a negative correlation between user ratings. Two functions can be used to determine how comparable a pair of ratings $loc(R_{Ai}, R_{Bj})$ are local. The first function uses Eq. (31) to determine the correlation between these two evaluations is given by

$$loc_{cor}(R_{Ai}, R_{Bj}) = \frac{(R_{Ai} - \hat{R_A})(R_{Bj} - \hat{RB})}{\sigma_A \sigma_B} \tag{31}$$

where $\hat{R_A}$ represents user A's average rating; R_{Ai} represents user A's rating of item i and σ_A represents user A's standard deviation of ratings. The average user rating is used as the reference scale by the function $loc_{cor}(.)$. For this objective, the ratings scale's median can also be considered for determining the local similarity between a pair of ratings R_{Ai} and R_{Bi} requires using another function $loc_{med}(R_{Ai}, R_{Bj})$. The Equation (32) is given as follows;

$$\mathrm{loc}_{\mathrm{med}}(R_{Ai}, R_{Bj}) = \frac{(R_{Ai} - R_{\mathrm{med}}) - (R_{Bj} - R_{\mathrm{med}})}{\sqrt{\sum_{k \in I_A} (R_{Ak} - R_{\mathrm{med}})^2} \sqrt{\sum_{k \in I_B} (R_{Bk} - R_{\mathrm{med}})^2}} \tag{32}$$

r_{med} is the rating scale's median, I_A is the set of items the user A rated, and R_{Ak} is the rating the user A gave item k. If ratings are given to the same item and the item similarity is 1 (BC (i, i) 1), the BCF gives the most weight to local similarity. The local similarity is not considered if two users rated completely different items (BC (i, j) =0). With the addition of S (A, B) of the Jaccard similarity measure Jacc (A, B) between users A and B, the number of shared items is given greater weight. As a result, Eq. (30) is altered as

$$S(A, B) = Jacc(A, B) + \sum_{i \epsilon I_A} \sum_{j \epsilon I_B} BC(i, j) loc(R_{Ai}, R_{Bj}) \tag{33}$$

The BCCF similarity metric has some key characteristics, which are listed as follows.

Co-rated items are few or absent: If an active user does not rate a certain minimal amount of items on which a sizable number of users have rated, then none of the available similarity metrics can compute the neighbourhood of that active user. Moreover, as BCCF does not depend on the quantity of co-rated items, it can compute the neighbourhood of the active user in this scenario.

Local and global data: The BCCF pulls global data from the sparse rating data that helps calculate the similarity between two users. The likelihood of discovering users comparable to an active user among the scant data is increased as a result. In Equation (29), the BC(.) determines whether two objects are globally similar. Using the functions loccor(.) or locmed(.), the proposed BCCF also calculates local information (similarity).

Utilization of all absolute ratings: JMSD, a version of Jaccard, uses all rating information but only in a limited sense (it is impossible to use the ratings' numerical values). In contrast, when computing each pair of ratings, BCCF uses the ratings' numerical values of Equation (33) to fully account for local user similarity and similarity between the rated objects. As a result, the movie's similarity index was calculated and minimized sparsity & cold start problems.

3.6 Hybrid Machine Learning Techniques

Aquila optimization is used in the Hybrid ML approach to optimize the Random Forest classifier's parameters, including the number of trees, the maximum depth of the trees, and the minimum amount of samples needed to divide a node. After that, the dataset is utilized to create a collection of cluster centres using the Fuzzy Probabilistic C-means Clustering method. Then, this procedure is used to determine the degree of membership of each data point to each cluster. The degree of membership values that arise are utilized to determine the similarity index between each pair of clusters using the Bhattacharyya Coefficient. The effectiveness of the classification can be enhanced by using this similarity index to evaluate the degree of similarity between the clusters. The similarity matrix for the clusters is then determined using the Bhattacharyya Coefficient similarity index values. The

similarity matrix allows the Random Forest classifier to capture the complicated data structure more accurately and efficiently. The hybrid method may considerably increase the accuracy and efficiency of classification tasks on complicated datasets by combining these various approaches, which accurately recommend movies. Algorithm 4 for Hybrid ML techniques is given below. The outcome of the proposed method will be discussed in the below section.

Algorithm 4: Hybrid Machine Learning Techniques

Step 1: Configuration Phase:
Step 2: Configure the Population A
Step 3: Configure the parameters of the AO(i.e.,β,δ, etc..)
Step 4: While(the end condition is not met)do
Step 5: Evaluate the fitness function values
Step 6: $A_{best(z)}$=Regulate the best attained solution according to the fitness values
Step 7: for(i=1,2,....,M)do
Step 8: Upgrade the mean value of the current solution $A_{N(z)}$
Step 9: Upgrade the u, v, L_1, L_2, Levy(d_8), etc.
Step 10: if $z \leq \left(\frac{2}{3}\right) * Z$ then
Step 11: if rand=0.5 then
Step 12: { **Phase 1:Expanded Exploration (A_1)** }
Step 13: Upgrade the current solution using Equation (3)
Step 14: if Fitness($A_1(z+1)$)<Fitness($A(z)$) then
Step 15: $A(z)=(A_1(z+1))$
Step 16: if Fitness $(A_1(z+1))$< fitness $(A_{best(z)})$ then
Step 17: $A_{best(z)}=A_1(z+1)$
Step 18: end if
Step 19: end if
Step 20: else
Step 21: { **Phase 2: Narrowed Exploration (A_2)** }
Step 22: Upgrade the current solution using Equation (5)
Step 23: if Fitness($A_2(z+1)$)<Fitness($A(z)$) then
Step 24: $A(z)=(A_2(z+1))$
Step 25: if Fitness $(A_1(z+1))$< fitness $(A_{best(z)})$ then
Step 26: $A_{best(z)}=A_1(z+1)$
Step 27: end if
Step 28: end if
Step 29: end if
Step 30: else
Step 31: if rand=0.5 then
Step 32: { **Phase 3:Expanded Exploitation (A_3)** }
Step 33: Upgrade the current solution using Equation (13)
Step 34: if Fitness($A_3(z+1)$)<Fitness($A(z)$) then
Step 35: $A(z)=(A_3(z+1))$
Step 36: if Fitness $(A_3(z+1))$< fitness $(A_{best(z)})$ then
Step 37: $A_{best(z)}=A_3(z+1)$
Step 38: end if
Step 39: end if
Step 40: else
Step 41: { **Phase 4: Narrowed Exploration (A_4)** }
Step 42: Upgrade the current solution using Equation (14)
Step 43: if Fitness($A_4(z+1)$)<Fitness($A(z)$) then
Step 44: $A(z)=(A_4(z+1))$
Step 45: if Fitness $(A_4(z+1))$< fitness $(A_{best(z)})$ then
Step 46: $A_{best(z)}=A_1(z+1)$
Step 47: end if
Step 48: end if
Step 49: end if
Step 50: end if
Step 51: end for
Step 52: end while
Step 53: return the best solution (A_{best})

4 Simulation Results And Discussion

The proposed model outcomes, performance evaluation, and comparative analysis are presented in this part. The analyses investigated in this study were carried out using data. Python language implementation used the Keras package to implement the machine learning techniques using Tensor Flow.

4.1 Dataset Illustration

Most frequently, recommender systems that attempt to forecast user movie ratings based on those of other users use the Movie Lens dataset. In other words, anticipate that people with similar tastes rank films in a highly correlated manner. The collection includes 1,000,209 anonymous ratings of roughly 3,900 films submitted by 6,040 Movie Lens users. The ratings are given on a numerical scale of 1 (poor), 2 (average), 3 (good), 4 (very good), and 5 (excellent). The user-product background data, such as age, occupation, genre, etc., are also included in this dataset. The file contains all ratings, including those with user IDs between 1 and 6040, Movie IDs between 1 and 3952, and at least 20 ratings per user. Users voluntarily supply all demographic data, which is not verified for correctness. This data collection only contains users who have contributed some demographic data.

There are movie details in the file. Titles are the same as those listed in the IMDB, including the year of release. The genres are pipe-separated and include ones like "action," "adventure," "animation," "children's," "comedy," "crime," etc. Due to inadvertent duplicate submissions or test entries, some Movie IDs do not match a movie. The dataset is divided into training and testing to recommend the movie. The proposed technique utilizes 80% for training and 20% for testing, and data for testing model parameters. The training and testing in the Hybrid Machine Learning Technique learn the data in the dataset to recommend the movies. Many metrics were used to test the Movie Lens dataset, such as accuracy, Root Mean Square, and Mean Absolute Error.

4.2 Performance evaluation Parameters

This section demonstrates that performance parameters such as Accuracy, Mean Absolute Error and Root Mean Square Error have been performed in the proposed method. The performance parameters will be described in the forthcoming section

4.2.1 Accuracy

The accuracy metric of the model is commonly used to summarise its performance across all classes. It works best when each class is given equal weight. It is calculated by dividing the total number of predictions by the number of accurate ones. The ratio of accurate forecasts to all input samples is measured. The accuracy can be expressed as

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \times 100$$

4.2.2 Mean Absolute Error

Model evaluation for regression models uses the Mean Absolute Error statistic (MAE). The mean absolute Error of a model concerning a given test set is the average of each prediction error's absolute value overall test set occurrences. Each prediction error differs between the value seen and the value anticipated for the instance. As shown by the MAE Equation is

$$MAE = \sum_{i=1}^{n} \frac{|y_i - x_i|}{n}$$

Where y_i is the prediction, xi is the actual value, and n is the number of instances. In time series analysis, the mean absolute Error is a typical indicator of predicted errors.

4.2.3 Root Mean Square Error

Root mean square error, also known as root mean square deviation, is one of the methods most frequently employed to assess the accuracy of forecasts. It exemplifies the Euclidean distance between forecasts and measured true values. Compute the root-mean-square Error for each data point by calculating the residual (difference between prediction and reality) together with its norm, mean, and square root (RMSE). RMSE is widely applied in supervised learning because it demands and uses real measurements at each projected data point.

The RMSE is the difference between the expected and actual values divided by the square root of the second sample moment. Errors (or prediction errors) are used instead of residuals when calculations are performed outside the data sample used for estimation. The errors in forecasts for various data points are combined using the RMSE to create a single indication of predictive power. The RMSE equation is written as

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} (\hat{y_i} - y_i)^2}{N}}$$

Where $\hat{y_i}$ is the predicted values of the regression dependent variable yi with variable observed over N times is calculated as the square root of the mean of the deviations' squares for T distinct forecasts.

Table 1: Performance parameters of the proposed method

Parameters	**Performance**
Accuracy	99.9%
Root Mean Square Error (RMSE)	0.63
Mean Absolute Error (MAE)	0.44

4.3 Performance Evaluation Results

This section describes the proposed framework results for Distance evaluation and recommendation of the movie.

4.3.1 Attribute Analysis Results

The attribute analysis module segregates the demographic attributes in Movie Len's dataset by type and value range. The attribute analysis module then examines the attributes that might be used to provide recommendations based on incorrect value ranges. To add users to the testing dataset from the training dataset, a certain number of users must have their ratings. Figure 2 shows the attribute analysis for users based on age, and ratings are given below

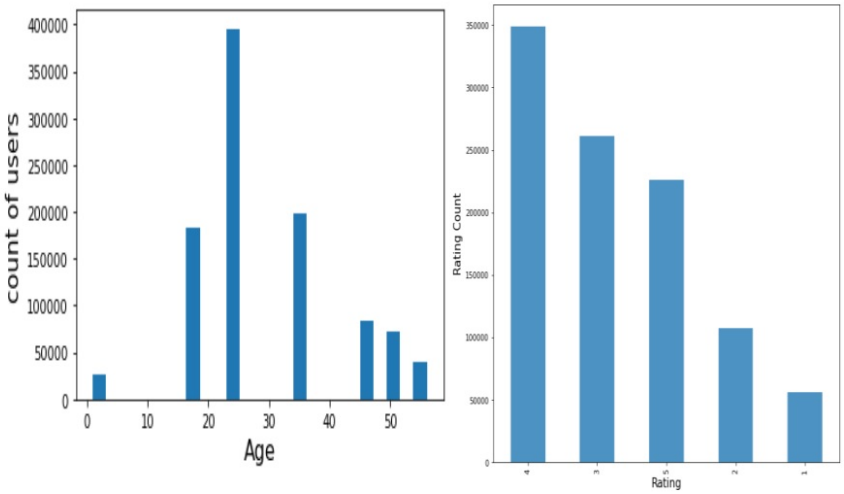


Figure 2: Attribute Analysis of users and ratings

4.3.2 Fuzzy Probabilistic C-means clustering

From Figure 3, the Fuzzy Probabilistic C-means are clustering clusters of the data based on rating and movies. FPCCA ought to cluster movies based on their attributes and reviews. Each movie has a probability of 1 if the probabilities add up to 1, indicating some degree of participation in each cluster. The average feature and rating values of every movie in a cluster serve as the centroids, representing the clusters. Using these centroids, users may receive movie recommendations depending on their preferences.

4.3.3 Distance Calculation Results

If the movie name entered is present in the database, use the recommendation system to locate comparable films, rank them according to how closely they resemble the input film, and output only the top 5 films along with their distances from it. Figure 4 reveals the distance evaluation of the proposed method.

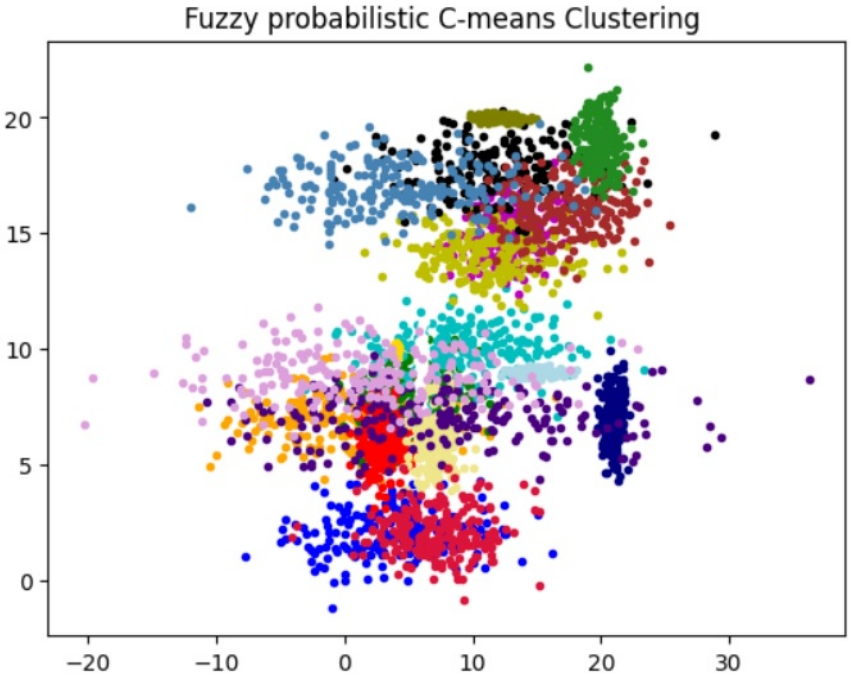


Figure 3: Output for Fuzzy Probabilistic C-means clustering

Searching for recommendations.....

	Title	Distance
1	Brother Minister: The Assassination of Malcolm X	0.428748
2	Batman Forever	0.426903
3	Red Rock West	0.419258
4	Muriel's Wedding	0.418472

Figure 4: Distance Evaluation of the Proposed Technique

4.3.4 Recommendation of the Movie Result

Numerous users may be rating and evaluating specific films. The ratings for the other films could be as low as one user. In these circumstances, certain less well-liked films may make the recommended list while other, more well-liked films may not. One can incorporate a rule to assess a movie's popularity to eliminate this bias accurately. Moreover, despite what the average ratings might imply, recent films may be more well-liked than older ones. In these situations, additional weight could be assigned to the ratings of recently released films to move them up the list of suggestions. Figure 5 depicts the recommendation of the movies based on the ratings given below

title	rating	total number of ratings
!Women Art Revolution	4.5	1
#1 Cheerleader Camp	5.0	1
#Horror	2.5	1
#chicagoGirl	1.0	1
$1,000 on the Black	4.5	1

Figure 5: Recommendation of the Movie

4.4 Comparison

This section focuses on the comparative efficiency of the proposed Hybrid ML Technique using a few prior methodologies. On comparing the proposed hybrid ML technique against the baseline of Item-based CF (IBCF) [30], User-based CF (UBCF) [31], Singular Value Decomposition (SVD) [32], Neural Collaborative Filtering (NCF) [33], Neural Matrix Factorization (NMF) [34].

The proposed method of MAE enhances by using Hybrid Machine Learning(ML) Technique. The proposed technique achieved less error based on FPCCA and CIAP minimize the data sparsity issue. The existing techniques has certain issues including cold start problem, scalability and Prediction rating issue, the proposed method was compared to the existing techniques such as IBCF, UBCF, SVD, NCF and NMF attained the MAE as 0.779,0.745,0.721,0.633 and 0.726 respectively. As a result, the proposed method achieved the Error of 0.44 which is less than existing approach also reduced the cold start problem.

By utilizing a Hybrid Machine Learning (ML) Technique, the RMSE enhancement method is proposed to overcome the prediction rating issue and scalability. When compared to the prior techniques such as IBCF, UBCF, SVD, NCF and NMF achieving values of 1.05, 1.02,0.92,0.853 and 1.03 respectively. As a result, the proposed technology, which outperformed from the existing method, attained an error of 0.63 and minimize the scalability and maximize prediction rating.

Table 2: Performance Comparison on Movielens 1M Dataset

Prior Methods	**RMSE**	**MAE**
IBCF [30]	1.05	0.779
UBCF [31]	1.05	0.745
SVD [32]	0.92	0.721
NCF [33]	0.853	0.633
NMF [34]	1.03	0.726
Proposed Hybrid ML	0.63	0.44

The table 3 describes the dataset comparative analysis of the proposed Hybrid ML Technique with prior models such as IBCF, UBCF, SVD, NCF and NMF for recommend the movies which attained

the RMSE of 0.63 and MAE of 0.44 respectively for Movielens 1M dataset which performs better than the previously used approaches and there are less errors as well as higher accuracy. Finally, the optimization, Clustering, Similarity index and recommendation performances of the movies in the proposed Hybrid ML techniques are compared with prior techniques can be efficiently the recommended the movies for the users.

5 Conclusion And Future Work

The article introduces a novel Hybrid Machine Learning Technique designed for the complex issue of movie suggestions in the rapidly growing field of digital entertainment. This technique, which makes use of the pre-processing as Attribute analysis module which was evaluated the categories of demographic data and examined attributes based on the invalid value range. Aquila Optimized Algorithm (AOA) is employed to select the best attributes and RF classifier provides an optimized attribute with comprehensive and precise solution for improving the user experience in the cinematic landscape. Fuzzy Probabilistic C-means Clustering Algorithm (FPCCA) is evoked, which groups the data into many clusters and clusters the unlabelled data, which minimizes sparsity & cold start problems. Then Correspondence Index Assessment phase, which utilized the Bhattacharyya coefficient that calculated the similarity index of the movies and the hybrid method produced the final recommendation for optimal movie recommendation with high accuracy. The MovieLens dataset was utilized for the research analysis, and the proposed Hybrid ML technique gives the low error of RMSE as 0.63 and MAE as 0.44 compared to conventional methods. and the movies were accurately recommended for the users. The efficiency of the Hybrid Machine Learning Technique depends on the completeness and quality of the demographic data, which might present difficulties when the information is erroneous or inadequate. The method's performance could differ between different datasets and situations, and real-world applications should carefully assess the computing requirements of this approach. In the future, the proposed method will be applied to context-based movie recommendations efficiently.

References

[1] Anbazhagu, U. V., Niveditha, V. R., Bhat, C. R., Mahesh, T. R., and Swapna, B. (2024). High-Performance Technique for Item Recommendation in Social Networks using Multiview Clustering. International Journal of Computers Communications & Control, 19(1).

[2] Cintia Ganesha Putri, D., Leu, J. S., and Seda, P. (2020). Design of an unsupervised machine learning-based movie recommender system. Symmetry, 12(2), 185.

[3] Ahuja, R., Solanki, A., and Nayyar, A. (2019, January). Movie recommender system using K-Means clustering and K-Nearest Neighbor. In 2019 9th International Conference on Cloud Computing, Data Science and Engineering (Confluence) (pp. 263-268). IEEE.

[4] Alshammari, M., and Alshammari, A. (2023). Friend recommendation engine for Facebook users via collaborative filtering. International Journal of Computers Communications & Control, 18(2).

[5] Lee, C., Han, D., Han, K., and Yi, M. (2022). Improving graph-based movie recommender system using cinematic experience. Applied Sciences, 12(3), 1493.

[6] Walek, B., and Fojtik, V. (2020). A hybrid recommender system for recommending relevant movies using an expert system. Expert Systems with Applications, 158, 113452.

[7] Saraswat, M., Chakraverty, S., and Kala, A. (2020). Analyzing emotion based movie recommender system using fuzzy emotion features. International Journal of Information Technology, 12(2), 467-472.

[8] Afoudi, Y., Lazaar, M., and Al Achhab, M. (2021). Hybrid recommendation system combined content-based filtering and collaborative prediction using artificial neural network. Simulation Modelling Practice and Theory, 113, 102375.

[9] Sujithra Alias Kanmani, R., Surendiran, B., and Ibrahim, S. P. (2021). Recency augmented hybrid collaborative movie recommendation system. International Journal of Information Technology, 13(5), 1829-1836.

[10] Vyas, P., Bhardwaj, A., and Vishwakarma, R. K. (2022). Decision-making Recommender System using Machine Learning Collaborative Filtering. Mathematical Statistician and Engineering Applications, 71(4), 3813-3820.

[11] Behera, G., and Nain, N. (2022). DeepNNMF: deep nonlinear non-negative matrix factorization to address sparsity problem of collaborative recommender system. International Journal of Information Technology, 1-9.

[12] Lang, F., Liang, L., Huang, K., Chen, T., and Zhu, S. (2021). Movie recommendation system for educational purposes based on field-aware factorization machine. Mobile Networks and Applications, 26(5), 2199-2205.

[13] Chen, H., Sun, H., Cheng, M., and Yan, W. (2021). A recommendation approach for rating prediction based on user interest and trust value. Computational Intelligence and Neuroscience, 2021.

[14] Liu, Y., Wang, S., Khan, M. S., and He, J. (2018). A novel deep hybrid recommender system based on auto-encoder with neural collaborative filtering. Big Data Mining and Analytics, 1(3), 211-221.

[15] Li, T., Su, X., Liu, W., Liang, W., Hsieh, M. Y., Chen, Z., and Zhang, H. (2022). Memory-augmented meta-learning on meta-path for fast adaptation cold-start recommendation. Connection Science, 34(1), 301-318.

[16] Viktoratos, I., and Tsadiras, A. (2022). A Machine Learning Approach for Solving the Frozen User Cold-Start Problem in Personalized Mobile Advertising Systems. Algorithms, 15(3), 72.

[17] Rama, K., Kumar, P., and Bhasker, B. (2021). Deep autoencoders for feature learning with embeddings for recommendations: a novel recommender system solution. Neural Computing and Applications, 33(21), 14167-14177.

[18] Sharma, S., Rana, V., and Kumar, V. (2021). Deep learning based semantic personalized recommendation system. International Journal of Information Management Data Insights, 1(2), 100028. Prabu, P., Sivakumar, R., and Ramamurthy, B. (2021). Corpus based sentimenal movie review analysis using auto encoder convolutional neural network. Journal of Discrete Mathematical Sciences and Cryptography, 24(8), 2323-2339.

[19] Choudhury, S. S., Mohanty, S. N., and Jagadev, A. K. (2021). Multimodal trust based recommender system with machine learning approaches for movie recommendation. International Journal of Information Technology, 13(2), 475-482.

[20] Yi, B., Shen, X., Liu, H., Zhang, Z., Zhang, W., Liu, S., and Xiong, N. (2019). Deep matrix factorization with implicit feedback embedding for recommendation system. IEEE Transactions on Industrial Informatics, 15(8), 4591-4601.

[21] Bhalse, N., and Thakur, R. (2021). Algorithm for movie recommendation system using collaborative filtering. Materials Today: Proceedings.

[22] Yassine, A. F. O. U. D. I., Mohamed, L. A. Z. A. A. R., and Al Achhab, M. (2021). Intelligent recommender system based on unsupervised machine learning and demographic attributes. Simulation Modelling Practice and Theory, 107, 102198.

[23] Awan, M. J., Khan, R. A., Nobanee, H., Yasin, A., Anwar, S. M., Naseem, U., and Singh, V. P. (2021). A recommendation engine for predicting movie ratings using a big data approach. Electronics, 10(10), 1215.

[24] Kiran, R., Kumar, P., and Bhasker, B. (2020). DNNRec: A novel deep learning based hybrid recommender system. Expert Systems with Applications, 144, 113054.

[25] Behera, D. K., Das, M., Swetanisha, S., and Sethy, P. K. (2021). Hybrid model for movie recommendation system using content K-nearest neighbors and restricted Boltzmann machine. Indonesian Journal of Electrical Engineering and Computer Science, 23(1), 445-452.

[26] Sharma, B., Hashmi, A., Gupta, C., Khalaf, O. I., Abdulsahib, G. M., and Itani, M. M. (2022). Hybrid sparrow clustered (HSC) algorithm for top-N recommendation system. Symmetry, 14(4), 793.

[27] Abualigah, L., Yousri, D., Abd Elaziz, M., Ewees, A. A., Al-Qaness, M. A., and Gandomi, A. H. (2021). Aquila optimizer: a novel meta-heuristic optimization algorithm. Computers and Industrial Engineering, 157, 107250.

[28] Wang, Y., Wang, L., Zhao, L., Ran, X., and Deng, S. (2021). Privacy recommendation based on Bhattacharyya coefficient. Procedia Computer Science, 188, 61-68.

[29] Wang, Y., Deng, J., Gao, J., and Zhang, P. (2017). A hybrid user similarity model for collaborative filtering. Information Sciences, 418, 102-118.

[30] Manochandar, S., and Punniyamoorthy, M. (2021). A new user similarity measure in a new prediction model for collaborative filtering. Applied Intelligence, 51(1), 586-615.

[31] Sallam, R. M., Hussein, M., and Mousa, H. M. (2020). An enhanced collaborative filtering-based approach for recommender systems. Int. J. Comput. Appl, 176(41), 9-15.

[32] Jena, K. K., Bhoi, S. K., Mallick, C., Jena, S. R., Kumar, R., Long, H. V., and Son, N. T. K. (2022). Neural model based collaborative filtering for movie recommendation system. International Journal of Information Technology, 14(4), 2067-2077.

[33] Kapetanakis, S., Polatidis, N., Alshammari, G., and Petridis, M. (2020). A novel recommendation method based on general matrix factorization and artificial neural networks. Neural Computing and Applications, 32, 12327-12334.

Journal's webpage: http://univagora.ro/jour/index.php/ijccc/

This journal is a member of, and subscribes to the principles of,
the Committee on Publication Ethics (COPE).
https://publicationethics.org/members/international-journal-computers-communications-and-control

Cite this paper as:

Paranjape, V.; Nihalani, N.; Mishra, N. (2024). Design and Development of an Efficient Demographic-based Movie Recommender System using Hybrid Machine Learning Techniques, *International Journal of Computers Communications & Control*, 19(4), 5840, 2024.
https://doi.org/10.15837/ijccc.2024.4.5840