A Study on Apriori based Movie Recommendation

Tapas Behera
Dept. of Computer Science & Information Technology
Siksha 'O' Anusandhan
Bhubaneswar, India
1941017035.a.tapas@gmail.com

*Sarita Mahapatra
Dept. of Computer Science & Information Technology
Siksha 'O' Anusandhan
saritamahapatra@soa.ac.in

Kritik Ranjan Mohanty

Dept. of Computer Science & Information Technology
Siksha 'O' Anusandhan
Bhubaneswar, India
1941017
001.a.kritikranjanmohanty@gmail.com

Mohammad Farhan
Dept. of Computer Science & Information Technology
Siksha 'O' Anusandhan
Bhubaneswar, India
194101
7135.a.mohammadfarhan@gmail.com

Sachidananda Tripathy
Dept. of Computer Science & Infromation Technology
Siksha 'O' Anusandhan
Bhubaneswar, India
sachidanandatripathy1234@gmail.com

*Sushree Bibhuprada B. Priyadarshini

Dept. of Computer Science & Information Technology
Siksha 'O' Anusandhan
Bhubaneswar, India
bimalabibhuprada@gmail.com

***Abstract*__**With so many movies available nowadays, it might be challenging for people to choose one that suit their preferences and tastes. We have created a system for suggesting movies to users based on their viewing interests and history. A dataset of user movie viewing history and preferences is first gathered and preprocessed by the movie recommendation system followed by employing an Apriori technique to find the common item sets from frequently watched movies. The recommendations are created using these item sets as a foundation. A real-world movie dataset is applied in extended trials for movie recommendation system while considering evaluation metrics like recall, precision, and accuracy. The experimental findings show that the suggested approach is capable of giving users precise and individualized movie suggestions by utilizing the Apriori algorithm.

Keywords. *Apriroi algorithm, dataset, Matplotlib*

I. INTRODUCTION

In modern era, recommendation project aims at developing a customized movie recommendation system using the Apriori algorithm. The project recognizes the rising need for helpful movie recommendations in the digital entertainment market, where clients are offered a bewildering variety of movie options. The first step of this research is the gathering of pertinent information for movie suggestion. This can come from a number of places, including user reviews, movie genres, movie information, and viewers watching histories, etc. The Apriori algorithm's implementation forms the basis of the movie recommendation system. An established approach for mining association rules that recognizes frequently occurring item sets and produces association rules is called the Apriori algorithm. The algorithm is modified in the context of the movie recommendation system to find trends and connections between movies based on user tastes and previous data.

A. *Motivation*

The movie recommendation system makes use of various rules after evaluating the association rules to produce recommendations that are specific to each user. To make reliable and pertinent movie choices, the system considers the user's tastes, viewing history, and may be other contextual data. Evaluation metrics are used to gauge how well the movie recommendation system is working. Accuracy, precision, recall, and user satisfaction metrics are a few examples of these metrics. By contrasting the suggested films with the consumers' actual preferences, the system's performance is assessed [1-5].

B. *Major Contribution*

This research contributes to the field of recommendation systems overall. Users enjoy watching movies more when the system takes associations between films into account and tailors recommendations based on their preferences, thus, making it easier for them to find new films that are relevant to their interests.

The rest part of the paper is arranged as follows: the next Section discusses the literature survey. Section III discusses the proposed method with extensive discussion on the tools and algorithms employed and the detailed working model followed by collaboration of detailed working model involved. Subsequently, Section IV details the results part. Finally, Section V concludes the paper with future research direction.

II. RELATED WORK

Various threads of work have been carried out for recommending movies in current scenario for specific set of users. The hybrid filtering method as used in [7] makes use of the ideas found in other methods. To get over the drawbacks of each approach, it blends context-based, content-based, and collaborative filtering [8]. It is better because it performs better when making recommendations and computes more quickly. The work done in [9] collaborates on the ecommerce based recommendation systems while that 0f [10] discusses the collaborative filtering strategy based recommendation system. Similarly, [11] incorporates the filtering algorithms employing k means neighbourhood voting. The work done in [12] discusses the movie recommendation system based on cuckoo search.

III. PROPOSED METHOD

A. Phases Involved in Proposed Framework

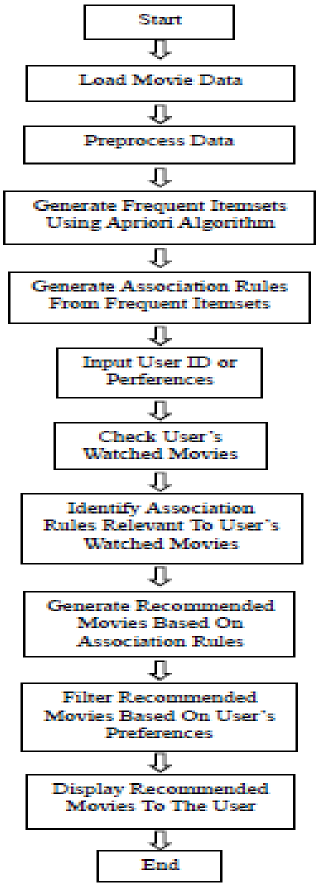

Start
Load Movie Data
Preprocess Data
Generate Frequent Itemsets Using Apriori Algorithm
Generate Association Rules From Frequent Itemsets
Input User ID or Perferences
Check User's Watched Movies
Identify Association Rules Relevant To User's Watched Movies
Generate Recommended Movies Based On Association Rules
Filter Recommended Movies Based On User's Preferences
Display Recommended Movies To The User
End

Fig. 1. Model Diagram showing the Phases of Recommendation System

(i) Loading movie data:

The movie dataset must be loaded into the system at this phase. The data set contains information about the viewing history and preferences of the project.

(ii) Preprocesing Data: The movie dataset is cleaned and transformed during the data preprocessing stage.

(iii) Generateing Frequent Itemsets using Apriori Algorithm:

To create frequent itemsets, the preprocessed movie dataset is subjected to the Apriori algorithm. Itemsets that are regularly viewed together by users are called frequent itemsets.

(iv) Generating Association Rules from Frequent Itemsets:

Association rules are constructed from the frequently occurring itemsets. Inferring associations between movies from their co-occurrence in user-movie interactions is done using association rules. An antecedent (premise) and a consequent (recommendation) make up each rule.

(v) Input User ID or Preferences:

The user is prompted by the system to provide their user ID or preferences, such as their current preferences or previously viewed movies.

(vi) Checking User's Watched Movies:

Using the user's ID or preferences, the system looks up the user's viewed films. In this stage, the dataset's movie history for the user is retrieved.

(vii) Identifying Association Rules Relevant to User's Watched Movies:

The algorithm creates suggested films based on the pertinent association rules and their consequences. The customer is given the opportunity to view one of these suggested films.

A. Filterring Recommended Movies based on User's Preferences:

Additional filtering is applied by the system depending on the user's choices or restrictions. This process helps the recommendations become even more tailored.

B. Displaying recommended movies to user:

The system presents the list of recommended movies to the user, either through a user interface or in the form of a personalized movie recommendation list.

B. TOOLS AND ALGORITHMS

(i) Abbreviations and Acronyms

GPU: T4

Processor: intel CORE i5

Storage: 256GB

Operating System: Windows 11

(ii) Google Colab:

It is primarily a web based device that the hunt engine large offers that enables customers to create, execute, and share Python code in a cloud-based system. It is especially helpful for activities involving data analysis, machine learning, and deep learning since it is meant to promote collaborative work and offers simple access to computing resources.

Users might also write code in cells and run it separately by the usage of Google Colab's Jupyter Notebook interface. It allows to get admission in to a extensive range of libraries and frameworks used in data science know-how and machine learning such as: NumPy, Pandas, TensorFlow, PyTorch, and scikit-learn, etc. It also supports the execution of Python 2 and Python 3 code.

Access to potent processing resources like CPUs, GPUs, and TPUs is made possible through Google Colab. Large dataset processing, sophisticated machine learning model training, and time-consuming calculations are frequently involved in creating movie recommendation systems. Utilizing these resources using Colab's hardware acceleration options lead to accelerate calculations and meet the computing needs of suggested recommendation system.

(iii) Libraries Used

(a) Apyori:

The Apriori algorithm for association rule mining is implemented using the Python module called apyori. It offers a quick and effective method for carrying out routine tasks like association rule extraction and itemset generation. You may quickly construct the Apriori algorithm and extract the association rules from your transaction dataset by utilising the apyori package. You may concentrate on analysing and using the created rules in your movie recommendation system or any other application because the library handles the labor-intensive tasks of frequent itemset creation and association rule extraction.

(b)Numpy:

A key Python module for scientific computing and numerical operations is the numpy library. It offers effective and practical methods for manipulating sizable multi-dimensional arrays and applying mathematical operations to them. Numerous mathematical operations and features are to be engaged in numpy that may be used in detail-by using-element on arrays. These comprise fundamental mathematical processes as well as trigonometric, logarithmic, statistical, and other functions. For building arrays from lists, ranges, or preexisting data, numpy provides functions. Using built-in functions, you can reshape, concatenate, split, and transpose arrays. Large datasets may be effectively stored and worked with and user-item matrices or sparse data representations are used in movie recommendation systems. These data structures, like matrices, can be efficiently computed with NumPy's array operations.

(c)Matplotlib:

Python users frequently utilize the matplotlib library for plotting and visualizing various data. It offers an extensive collection of tools for making several styles of plots, charts, and visualizations. You can plot data straight from numpy arrays. It is now simple to visualize numerical data and scientific calculation findings. It enables the creation of line plots, scatter plots, bar graphs, histograms, pie charts, and other plots using a range of plotting tools. With the use of these features, plot components including colors, markers, line styles, and labels can be changed effectively. For making different kinds of plots and visualisations, Matplotlib offers a large selection of tools and functions. It may be used to visualise data distributions, ratings, user preferences, movie popularity, and other pertinent information in a movie recommendation system. Visualisations aid in deciphering data, spotting trends, and communicating findings to users or stakeholders must be taken into account.

(d) Pandas:

Python has a robust and well-liked library for manipulating and analyzing data. It is a key tool for working with structured data since it offers simple-to-use data structures and data analysis tools. The Data Frame, a two-dimensional tabular data structure, is the fundamental data structure in pandas. Like a spreadsheet or SQL table, it enables you to store and modify structured data with labelled rows and columns. Data Frames support multiple operations like indexing, slicing, filtering, and aggregating and can handle a variety of data formats. For effective management and manipulation of structured data, such as information on user-item interactions, movie metadata, and ratings, Pandas offers a variety of data structures and methods. The DataFrame object, which enables sophisticated data manipulation operations including filtering, sorting, merging, grouping, and aggregating, etc.

(e) Programming language used:

Python offers a large ecosystem of tools and frameworks that are especially made for data analysis, machine learning, and recommendation systems. We decided to utilise Python for our project. It is simpler to construct and test recommendation models with libraries such as NumPy, Pandas, SciPy, Scikit-learn, and TensorFlow that offer strong tools for data manipulation, numerical computation, statistical analysis, and machine learning algorithms, etc. Python's syntax is clear and understandable, making it simple to read and create code. It increases productivity and code readability by enabling developers to describe complicated algorithms and concepts in a clear, succinct manner.

(f) *Association rule mining:*

A data mining method called association rule mining is used to discover exciting connections or styles in huge datasets. Based on the co-incidence of items or variables in transactions, it seeks to decide relationships or dependencies among them in a dataset. Identifying frequently occurring itemsets and producing association rules based on them are the two steps in the association rule mining process. A common technique used for frequent itemset creation and rule extraction is the Apriori algorithm. Association rule works as follows:

i) Frequent itemset generation:

Establish the minimal frequency or occurrence necessary for an itemset to be deemed frequent, also known as the Minimum Support Threshold(MST). Look for frequent item sets that satisfy the support threshold by scanning the dataset.

ii) Association rule generation:

Calculate the confidence for each rule, which is the measure of the conditional probability of the consequent given the antecedent. Create association rules based on the frequent itemsets that consist of an antecedent and a consequent . Establish the minimum confidence threshold, which denotes the bare minimum degree of certainty necessary for a rule to be considered "interesting."

iii) Rule evaluation:

Apply other metrics, such as lift, conviction, or support, to evaluate the standard and importance of the created rules. Using the assessment criteria, choose the rules that are the most intriguing and pertinent.

iv) Postprocessing and interpretation:

Use postprocessing methods, such as rule pruning or redundancy removal, to enhance the created rules. To understand the connections between things or variables, interpret the rules effectively. Use the rules for cross-selling, upselling, recommendation systems, and other pertinent application

C. APRIORI ALGORITHM

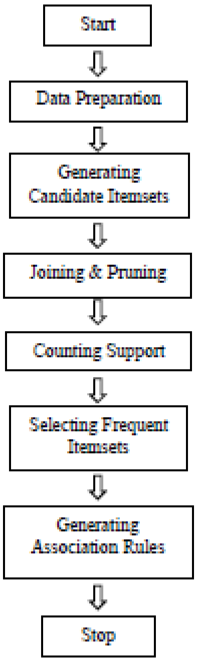

Start
Data Preparation
Generating Candidate Itemsets
Joining & Pruning
Counting Support
Selecting Frequent Itemsets
Generating Association Rules
Stop

Fig. 2. Steps involved in Apriori algorithm

The Apriori algorithm is very much popular for association rule mining. In order to create association rules, it seeks to identify frequent itemsets in a dataset. The Apriori method is described as follows:

a) Step 1: The Apriori algorithm starts with the data preparation step. It needs a dataset made up of films, each of which has a certain collection of elements.

b) Step 2: Generating Candidate Itemsets: The procedure begins by producing candidate itemsets with size k=1 and just one item per set. It counts the number of times where each item appears in the dataset by scanning it, then chooses the frequent things based on a minimal support threshold. Items that occur more frequently than the criterion are considered frequent.

c) Step 3: Joining and Pruning: By merging the frequent itemsets of size k, the algorithm creates candidate itemsets of size k+1 in the following phase. When two itemsets with the same initial k-1 items are combined, the programme looks to see if the result is a candidate itemset with possible support.

To reduce the search space, the algorithm performs pruning. It removes any candidate itemset of size k+1 that has a subset of size k which is not frequent. This is known as the "Apriori property", which states that if an itemset is infrequent, all its supersets will also be infrequent.

d) Step 4: Counting Support: To calculate the support for each potential itemset of size k+1, the programme searches the dataset once again. By counting the number of times each potential itemset appears in the dataset, it determines how frequently it happens.

e) Step 5: Selecting Frequent Itemsets: Based on a minimal support criterion, the algorithm chooses the often occurring itemsets of size k+1. The following iteration of the algorithm uses these frequently occurring itemsets as input to create bigger itemsets.

f) Step 6: Generating Association Rules: The algorithm produces association rules from the frequent item sets as a final step. An antecedent (on the left) and a consequent (on the right) make up an association rule. Based on the co-occurrence of the elements in the collection, the rules capture the relationships between them.

In a movie recommendation system, for instance, an association rule may be written like "Comedy, Drama" -> "Romance," suggesting that viewers of those genres are more likely to appreciate Romance movies.

D. Working of the Model

(i) Data Collection

The dataset used in this research consists of around 100 movies available on different OTT platforms. Each movie is associated with a set of genres. We have taken the preferences of more than 7000 users in this research. The Apriori algorithm relies on identifying the associations and patterns in the dataset. Data collection provides the necessary information about user preferences, movie attributes, and user-movie interactions. Without comprehensive and relevant data, it would be challenging to uncover meaningful associations that can drive accurate and personalized recommendations. The quality of movie recommendations heavily relies on the richness and diversity of the collected data.

A comprehensive dataset enables the system to identify associations and patterns across different movie attributes, user preferences, and behaviors. This, in turn, improves the accuracy and relevance of the hints supplied to users. Data collection enables the discovery of meaningful associations, generation of frequent itemsets, setting appropriate support thresholds, enhancing recommendation quality, personalizing recommendations, and facilitating continuous improvement of the system. The availability of comprehensive and Diverse information becomes vital for the achievement and effectiveness of the recommendation device.

(ii) Setting Minimum Support

A threshold number known as the minimum support establishes the minimal occurrence frequency of an itemset inside the dataset. It stands for the minimal proportion or number of transactions required for a group of things to seem to be frequent. The support threshold can be calculated in the following ways-

a. Estimate the whole volume of transactions by tallying up all the transactions in the dataset. The overall number of user-movie interactions is shown here.

b. Choose the distinct movie itemsets from the dataset to identify.

c. Identify movie itemsets: Find the distinct movie item sets in the data. A movie or a group of movies that appear together in a transaction are referred to as an itemset.

d. Calculate the support for ech itemset: count the number of transactions in which each itemset is present. To determine the support value, divide this count by the total number of transactions.

$$Support = \frac{User\ watch\ list\ which\ contains\ Movie\ M\ 1}{Total\ user\ watch\ list} ¿$$

)

e. Setting up minimum support threshold: we have taken minimum support of 0.003 so that an itemset must meet to be considered frequent. The criteria of the recommendation system and the desired trade-off between the quantity of frequently occurring itemsets and the algorithm's effectiveness determine the precise value for the support threshold. Support levels that are frequently employed ranging from 1% to 10%.

We can regulate the amount of frequently occurring itemsets taken into account while generating suggestions by modifying the support threshold. Lower thresholds result in more frequent itemsets, while higher support thresholds result in fewer frequent itemsets.

(iii) Setting Minimum Confidence

Confidence is the degree of the reliability that is described as the percentage of cases in which the association rule holds true.

The confidence is calculable as:

a. Calculation of confidence threshold: Find the confidence level for every affiliation rule. The proportion of transactions that contain each A and B to transactions that incorporate A is what's referred to as the confidence of a rule A -> B.

b. Setting up the minimum confidence: the minimum confidence we have taken is 0.2 so that an association rule must need to be considered significant. The criteria of the recommendation system and the desired trade-off between the quantity of recommended films and the intensity of the link determine the precise value for the confidence threshold. The range of frequently uses confidence criteria ranging from 50% to 80%.

$$Confidence = \frac{User\ watch\ list\ containing\ M\ 1 \wedge M\ 2}{User\ watch\ list\ containing\ movie\ m\ 1} (2$$

.

We can manage the degree of correlation needed for a proposal to be taken seriously by modifying the confidence threshold. Fewer association rules are deemed significant at higher confidence criteria, however, more rules are deemed significant at lower confidence thresholds.

(iv) Setting Minimum Lift

Lift measures the degree to which the antecedent (a group of movies) and the consequent (a suggested movie) in an association rule are related. It reveals the extent to which the occurrence of the antecedent affects a user's propensity to select the consequent. The minimum lift we have taken is 3 to get strong associations between the set of movies.

$$Lift = \frac{Confidence}{Support} (3)$$

We can find powerful and significant links between sets of movies by examining the lift values of association rules, and we can utilize these associations to inform the movie recommendation system's suggestions. To ensure the applicability of the advice, it is crucial to take into account additional variables like confidence and support in addition to lift.

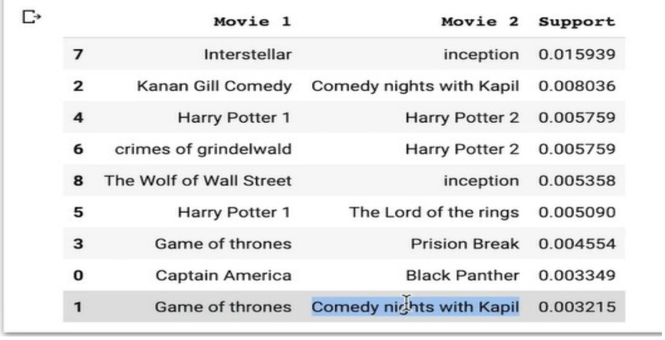
(v) *Frequent Itemsets Generation*

Using the apriori algorithm and user profiles, personalized frequent itemsets are generated. The system identifies movies that are associated with the user's preferred attributes or have strong associations with movies they have previously liked.

IV. RESULTS AND DISCUSSION

Fig. 3 shows the result the of the movie sets which have higher association between them. For example, our recommendation engine suggested inception to viewers of Interstellar. Since Christopher Nolan directed both of these movies, admirers of the director will undoubtedly watch Inception.

Our Apriori based recommendation system achieves an accuracy of 0.006 which was calculated by taking mean of the top nine support threshold. This shows that a significant number of the movies that users may be interested in are covered by our system's ability to reliably propose movies that users will enjoy.



	Movie 1	Movie 2	Support
7	Interstellar	inception	0.015939
2	Kanan Gill Comedy	Comedy nights with Kapil	0.008036
4	Harry Potter 1	Harry Potter 2	0.005759
6	crimes of grindelwald	Harry Potter 2	0.005759
8	The Wolf of Wall Street	inception	0.005358
5	Harry Potter 1	The Lord of the rings	0.005090
3	Game of thrones	Prision Break	0.004554
0	Captain America	Black Panther	0.003349
1	Game of thrones	Comedy nights with Kapil	0.003215

Fig 3. Result of top nine support threshold

V. CONCLUSION AND FUTURE DIRECTION

The research's goal was to create an Apriori-based movie recommendation system. Based on association rules created from user-movie interactions, the system was created to offer tailored movie suggestions. The system demonstrated the scalability of the Apriori technique, which allowed it to handle enormous movie collections well. The system efficaciously mined the commonplace itemsets and association guidelines, making it appropriate to be used in real-world programs wherein the scale of the dataset might be sizeable. The research also noted a few restrictions. The Apriori technique has certain drawbacks, including the potential for computational complexity, especially when dealing with huge datasets and high-dimensional item spaces. This may have an effect on the recommendation system's real-time performance and need optimization techniques.

Several important conclusions were attained as a result of the installation and assessment of the recommendation system. The Apriori algorithm demonstrated its ability to find common itemsets and provide useful association rules. With the use of these criteria, the system was able to provide users with suggestions that were pertinent to the relationships between the movies. In conclusion, by producing customized and pertinent movie recommendations, the Apriori algorithm-based movie recommendation system met the project's goals. Based on user-movie interactions, the system demonstrated its capacity to apply association rules to provide precise predictions. The project offers a strong platform for more study and advancements in the area of movie recommendation systems, despite several limits that were noted.

Following are some of the future directions:

a. *User Feedback:*

Encourage user comments on the suggested films to improve the recommendation algorithm. Users' satisfaction may be increased by collecting explicit feedback, reviews, and ratings from users. The association rules may be updated using this input, and the system can be modified over time.

b. *Hybrid approaches:*

Look into combining several recommendation systems, such as the Apriori algorithm, to take use of their advantages and get around their drawbacks. To increase recommendation accuracy and coverage, hybrid techniques can incorporate collaborative filtering, content-based filtering, and the apriori algorithm.

c. *Sequential patterns:*

Expand the Apriori algorithm to take successive user-movie interaction patterns into account. Sequential patterns record the temporal ordering of film tastes, thereby making suggestions more precise and individualized depending on consumers' watching histories.

d. *Contextual factors:*

Examine how to include contextual elements like location, time, and social context in the recommendation system. Context-aware suggestions may adjust to the tastes of users in various contexts, resulting in more pertinent and timely movie selections.

e. *Evaluation metrics:*

Investigate the creation of fresh assessment measures that account for the novelty, serendipity, fairness, and user pleasure elements of recommendation quality. Incorporate user research and feedback to evaluate the recommendation system's overall impact and efficacy.

f. *User privacy:*

Future research should address these issues in movie recommendation systems due to growing worries about algorithmic bias and data privacy. Research may concentrate on creating recommendation methods that maintain user privacy while making reliable suggestions. In order to prevent prejudices from being reinforced, efforts should also be made to guarantee fairness and openness in the recommendation process.

REFERENCES

[1]Pradana, H., etal, Product Recommendation using apriori in the selection of shoe based Android, In Proceedings of the International Conferences on Information System and Technology (CONRIST 2019), pp. 311-318, 2019.

[2] Zakaria, A., "A movie recommendation system design using association rules mining and classification techniques", pp. 189-199, 2022.

[3].Recommender system using affinity analysis by Jypter notebook community (https://notebook.community/Aniruddha-Tapas/Applied-Machine Learning/Machine%20Learning%20using %20GraphLab/)

[4] Hadoop-based movie recommendation engine: A comparison of the apriori algorithm vs the K-means method(https://www.altoros.com/) .

[5] Kurnia, W., Movie recommendation system using knowledge-based and k-means clustering, vol. 3, no. 4, pp. 460-465, 2022.

[6].Product recommendation using Machine learning(https://www.javatpoint.com/product-recommendation-machine-learning).

[7] Beniwal R., Debnath K., Jha D., Singh M., "Data Analytics and Management", *Springer; Berlin/Heidelberg, Germany*, Hybrid Recommender System Using Artificial Bee Colony Based on Graph Database; pp. 687–699, 2021.

[8] Çano E., Morisio M. Hybrid recommender systems: A systematic literature review. *Intell. Data Anal, vol.* **21**, pp. 1487–1524, 2017, doi: 10.3233/IDA-163209.

[9] Schafer J.B., Konstan J.A., Riedl J., "E-commerce recommendation applications", *Data Min. Knowl. Discov, vol.* **5**, pp. 115–153, 2001, doi: 10.1023/A:1009804230409

[10] Shen J., Zhou T., Chen L., "Collaborative filtering-based recommendation system for big data", *Int. J. Comput. Sci. Eng, vol.* **21**, pp. 219–225, 2020. doi: 10.1504/IJCSE.2020.105727

[11] Dakhel G.M., Mahdavi M., "A new collaborative filtering algorithm using K-means clustering and neighbors voting", *Proceedings of the 11th International Conference on Hybrid Intelligent Systems (HIS);* Malacca, Malaysia. 5–8 December, pp. 179–184., 2011

[12] Katarya R., Verma O.P. An effective collaborative movie recommender system with cuckoo search. *Egypt. Inform. J.* , vol. **18**, pp. 105–112, 2017, doi: 10.1016/j.eij.2016.10.002