**Optimization of the Hybrid Movie Recommendation System Based on Weighted Classification and User Collaborative Filtering Algorithm**

**1. INTRODUCTION:**

With the rapid development of information technology and social networks, the data generated by the Internet has risen exponentially in recent years, and the era of big data is coming. With the increase of data, it is more and more difficult for people to find the information they really want from the massive data. At this time, the recommender system can play the maximum application value. According to the user information and user historical behavior data, the recommendation system can accurately predict the user’s preferences, personalize the things that users may be interested in, and greatly reduce the cost of finding target information. Content-based collaborative filtering algorithm (CBF) and traditional collaborative filtering algorithm (CF) have their own shortcomings. When a new project is added to the system, but its project characteristics cannot be obtained or described, CBF cannot be used at this time. Recommender system makes up for the deficiency of search engine. It does not need users to put forward clear requirements. Instead, it recommends information that meets users’ personalized needs to users by analyzing users’ historical behavior, applying recommendation algorithm, or establishing users’ interest model. In the research of recommender system, people focus on the improvement of recommender algorithm. At present, the mainstream recommendation algorithms include the following categories: collaborative filtering-based recommendation, content-based recommendation, and hybrid recommendation. Although collaborative filtering algorithm has been widely used in movie recommendation, music recommendation, and other fields, it still has some problems such as data sparsity, user interest change, cold start, and scalability.

Different user groups often form their own unique behavior patterns. For example, the users who love military war tend to score more in military movies, but their scoring behavior in literary movies is relatively sparse. On the contrary, the users who love literature and art have more scoring behavior in literature and art movies, but less in military movies. For some popular movies, no matter what the theme is, users in each user group are usually very active in their scoring behavior. The neighborhood-based collaborative filtering is to calculate the similarity between items by analyzing the user’s behavior. It thinks that item A and item B have great similarity because most users who like item A also like item B. This means that the similarity between popular movies and war movies is different in the above two user groups, similar in the military enthusiast group, and not similar in the literature and art enthusiast group. There are also many people who have studied the hybrid recommendation method. Froelich and Hajek combined the domain model-based algorithm with the matrix decomposition based algorithm in collaborative filtering algorithm used photos uploaded by users and preference photos as mixed preferences to study the characteristics of users. proposed the multidimensional matrix factorization model, combined with collaborative filtering (CF) algorithm, and used user and project attributes to make score prediction, which improved the accuracy of prediction score. The first mock exam algorithm usually constructs a single global model for all users. It is considered that the similarity of two identical items in any group is the same. Obviously, this single model cannot find the difference of similarity among items in different user groups and thus cannot accurately capture user preferences and recommend the general effect.

The recommendation algorithm proposed in this paper aims to solve the above problems by constructing a local model. In order to solve the problem that a single model cannot accurately capture user preferences, this paper proposes a local model weighted fusion recommendation algorithm based on user clustering. The local model is trained by user subgroup partition, and finally the global model weighted fusion is used to improve the quality of recommendation. Because the local model only uses the local training data in the training process, it will lose the same important global information, so the fusion algorithm proposed in this paper is the linear weighted fusion between the local model and the global model. The local model plays an auxiliary and correction role in the prediction of the global model. The importance of different models to the fusion model is controlled by adjusting the weight parameters. In addition, in order to achieve the division of user subgroups, this paper uses the text content information of the movie, proposes to use LDA topic model to calculate the user feature vector, and uses spectral clustering algorithm to achieve user clustering based on the feature vector. The recommendation algorithm proposed in this paper is an effective combination of content-based recommendation, neighborhood-based collaborative filtering, and model-based collaborative filtering, so it has the advantages of interpretable recommendation results, fast recommendation speed, and high recommendation quality. The effectiveness of the proposed recommendation algorithm is proved by several experiments using the movie dataset.

**1.1 Objective of the project:**

Aiming at the problem that the single model of the traditional recommendation system cannot accurately capture user preferences, this paper proposes a hybrid movie recommendation system and optimization method based on weighted classification and user collaborative filtering algorithm. The sparse linear model is used as the basic recommendation model, and the local recommendation model is trained based on user clustering, and the top-*N* personalized recommendation of movies is realized by fusion with the weighted classification model. According to the item category preference, the scoring matrix is converted into a low-dimensional, dense item category preference matrix, multiple cluster centers are obtained, the distance between the target user and each cluster center is calculated, and the target user is classified into the closest cluster. Finally, the collaborative filtering algorithm is used to predict the scores for the unrated items of the target user to form a recommendation list. The items are clustered through the item category preference, and the high-dimensional rating matrix is converted into a low-dimensional item category preference matrix, which further reduces the sparsity of the data. Experiments based on the Douban movie dataset verify that the recommendation algorithm proposed in this article solves the shortcomings of a single algorithm model to a certain extent and improves the recommendation effect.

**2. LITERATURESURVEY:**

**A survey of collaborative filtering-based recommender systems: from traditional methods to hybrid methods based on social networks**

In the era of big data, recommender system (RS) has become an effective information filtering tool that alleviates information overload for Web users. Collaborative filtering (CF), as one of the most successful recommendation techniques, has been widely studied by various research institutions and industries and has been applied in practice. CF makes recommendations for the current active user using lots of users’ historical rating information without analyzing the content of the information resource. However, in recent years, data sparsity and high dimensionality brought by big data have negatively affected the efficiency of the traditional CF-based recommendation approaches. In CF, the context information, such as time information and trust relationships among the friends, is introduced into RS to construct a training model to further improve the recommendation accuracy and user’s satisfaction, and therefore, a variety of hybrid CF-based recommendation algorithms have emerged. In this paper, we mainly review and summarize the traditional CF-based approaches and techniques used in RS and study some recent hybrid CF-based recommendation approaches and techniques, including the latest hybrid memory-based and model-based CF recommendation algorithms. Finally, we discuss the potential impact that may improve the RS and future direction. In this paper, we aim at introducing the recent hybrid CF-based recommendation techniques fusing social networks to solve data sparsity and high dimensionality and provide a novel point of view to improve the performance of RS, thereby presenting a useful resource in the state-of-the-art research result for future researchers.

**Improve performance of association rule-based collaborative filtering recommendation systems using genetic algorithm**

Recommender systems that possess adequate information about users and analyze their information, are capable of offering appropriate items to customers. Collaborative filtering method is one of the popular recommender system approaches that produces the best suggestions by identifying similar users or items based on their previous transactions. The low accuracy of suggestions is one of the major concerns in the collaborative filtering method. Several methods have been introduced to enhance the accuracy of this method through the discovering association rules and using evolutionary algorithms such as particle swarm optimization. However, their runtime performance does not satisfy this need, thus this article proposes an efficient method of producing cred associations rules with higher performances based on a genetic algorithm. Evaluations were performed on the data set of MovieLens. The parameters of the assessment are: run time, the average of quality rules, recall, precision, accuracy and F1-measurement. The experimental evaluation of a system based on our algorithm outperforms show than the performance of the multi-objective particle swarm optimization association rule mining algorithm, finally runtime has dropped by around 10%.

**Movie recommendation system content-based and collaborative filtering**

Movies are one of the sources of entertainment, but the problem is in finding the desired content from the ever-increasing millions of content every year. However, recommendation systems come much handier in these situations. The aim of this paper is to improve the accuracy and performance of a regular filtering technique. Although varieties of methods are used to implement a recommendation system, Content-based filtering is the simplest method. Which takes input from the users, rechecks his/her history/past behaviour, and recommends a list of similar movies. In this paper, to prove the effectiveness, K-NN algorithms and collaborative filtering are used to mainly focus on enhancing the accuracy of results as compared to content-based filtering. This approach is based on cosine similarity using k-nearest neighbour with the help of a collaborative filtering technique, at the same time removing the drawbacks of the content-based filtering. Although using Euclidean distance is preferred, cosine similarity is used as the accuracy of cosine angle and the equidistance of movies remain almost the same.

**Personalized rough-set-based recommendation by integrating multiple contents and collaborative information**

In recent years, explosively-growing information makes the users confused in making decisions among various kinds of products such as music, movies, books, etc. As a result, it is a challenging issue to help the user identify what she/he prefers. To this end, so called recommender systems are proposed to discover the implicit interests in user’s mind based on the usage logs. However, the existing recommender systems suffer from the problems of cold-start, first-rater, sparsity and scalability. To alleviate such problems, we propose a novel recommender, namely FRSA (Fusion of Rough-Set and Average-category-rating) that integrates multiple contents and collaborative information to predict user’s preferences based on the fusion of Rough-Set and Average-category-rating. Through the integrated mining of multiple contents and collaborative information, our proposed recommendation method can successfully reduce the gap between the user’s preferences and the automated recommendations. The empirical evaluations reveal that the proposed method, FRSA, can associate the recommended items with user’s interests more effectively than other existing well-known ones in terms of accuracy.

**Combining rough set-based relevance and redundancy for the ranking and selection of nominal features**

In this paper, we propose a new method for features ranking and selection. Our approach is based on ranking nominal features in terms of their relevance to the assigned class and mutual redundancy with the other features. To calculate the relevance and redundancy, we propose to use a rough-set based approach. After performing the ranking, features filtering is carried out in a supervised way enabling the user to decide on the number of the retained features. The experiments revealed that thanks to our method, it is possible to filter out numerous features describing data while still maintaining satisfactory classification accuracy achieved by the classifier trained using the reduced dataset. The comparative experiments performed with the use of publicly available datasets proved the high efficiency and competitiveness of our approach.

**Movie recommender system using two way filtering and agglomerative hierarchical clustering**

Now a days the online services acts as a major technology for sharing the computing resource, data through web. Since in the internet, there are number of choices which is overwhelming we need to track the efficient product especially if the people are working with e-commerce applications. Hence the recommender system is introduced to get the best product based on the ratings given by the existing customers. In existing approach focus is based collaborative filtering and content based recommendation which fails to meet the applications in web service. Hence the method of Agglomerative Hierarchical Clustering is introduced which considers both rating data (QoS) and semantic content data (functionalities) using a probabilistic generative model. This paper explores the usage of recommendation system to assist users in choosing a movie based on similar interest. It focuses on the movie recommendation system which makes use of clustering, then the collaborative filtering algorithm is applied on the clustered dataset.

**Developing a hybrid collaborative filtering recommendation system with opinion mining on purchase review**

The most commonly used algorithm in recommendation systems is collaborative filtering. However, despite its wide use, the prediction accuracy of this algorithm is unexceptional. Furthermore, whether quantitative data such as product rating or purchase history reflect users’ actual taste is questionable. In this article, we propose a method to utilise user review data extracted with opinion mining for product recommendation systems. To evaluate the proposed method, we perform product recommendation test on Amazon product data, with and without the additional opinion mining result on Amazon purchase review data. The performances of these two variants are compared by means of precision, recall, true positive recommendation (TPR) and false positive recommendation (FPR). In this comparison, a large improvement in prediction accuracy was observed when the opinion mining data were taken into account. Based on these results, we answer two main questions: ‘Why is collaborative filtering algorithm not effective?’ and ‘Do quantitative data such as product rating or purchase history reflect users’ actual tastes?’

**3. SYSTEM ANALYSIS**

**3.1 Existing System**

Now-a-days due to increasing demand and access of online information technology leads to major data accumulation which is called as BigData and its become mandatory to process such BigData to provide recommendation to known or unknown users on which items to purchase or which movie to watch. In past many recommendation algorithms was introduced such as Content Based Recommendation (CBR (which matches similarity between user items and then perform recommendation and Collaborative Filtering (this will matches similarity between similar behaviour user and then perform recommendation). All existing algorithms will generate high dimensional array from dataset by taking matrix with users and their ratings and if dataset contains huge record then matrix contains more ratings columns which lead to high dimensional array and this will consume more time and recommendation will be inaccurate.

**Disadvantages of Existing System:**

1. Accuracy is less.

2. Existing algorithms taking more time.

**3.2 Proposed System**

To overcome from above problem author of this paper introduced Hybrid Recommendation with weighted classification. Propose algorithm consists of following modules

Sparse Linear Model: using this module we will extract required features from dataset such as User\_ID, Movie\_ID and ratings to build basic recommendation model. Extracted features will be split into TRAIN and TEST data where application will use 80% dataset size for training and 20% dataset size for testing.

Local Recommendation Model Training: extracted sparse features will get trained with clustering algorithm called KMEANS. KMEANS will group similar behaviour users in to same cluster. Similar behaviour user’s similarity will be calculated using Pearson Spearman function. KMEANS trained model will be applied on 20% test data to predict LOCAL HIT RATE (refers to correct number of ratings prediction) and Average hit rate.

Top-N personalized recommendation using Weighted Classification: In this module we will take USER ID & MOVIE ID as input and then extract Cluster Centres from KMEANS and then NEAREST NEIGHBORS will be calculated between INPUT and CLUSTER CENTRES by using PEARSON formula and this formula will return MAX Threshold similarity users and their movies as RECOMMENDATION LIST. Calculating correct rating prediction using NEAREST NEIGHBORS is known as Global HIT RATE and its average is called as Average Hit Rate.

**Advantages of Proposed System:**

1. Accuracy is more.
2. Proposed algorithms taking less time.

**Modules**

1. Upload Movielens Dataset
2. Calculate Sparse Linear Model
3. Run Local Clustering Algorithm
4. Comparison Graph
5. Run Weighted Classification

**Modules Description:**

**Upload Movielens Dataset:** using this module we will upload dataset to application

**Calculate Sparse Linear Model:** using this module we will read dataset and then build SPARSE LINEAR features model

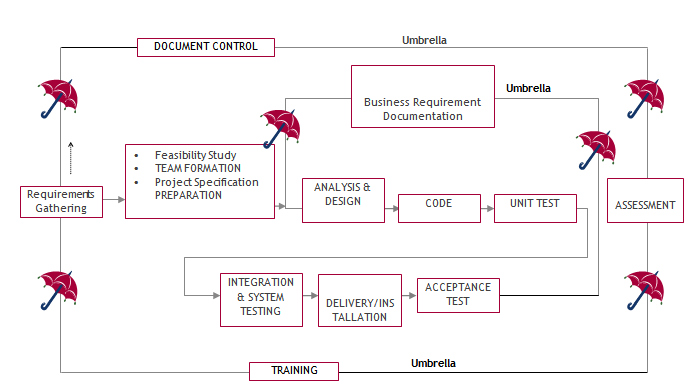
**Run Local Clustering Algorithm:** sparse features will be input to KMEANS to group similar user into same cluster by using PEARSON function

**Comparison Graph:** using this module we will plot local and global HR, ARHR graph

**Run Weighted Classification:** using this module we will accept User and Movie ID and then using weighted classification will predict movie recommendation

**3.3. PROCESS MODEL USED WITH JUSTIFICATION**

**SDLC (Umbrella Model):**

****

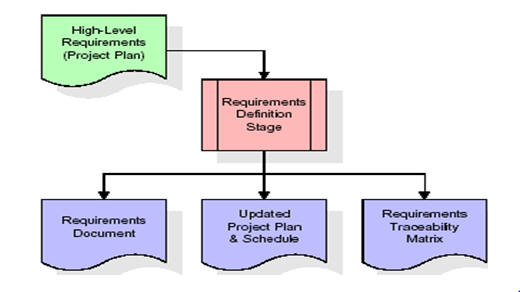
The requirements gathering process takes as its input SDLC is nothing but Software Development Life Cycle. It is a standard which is used by software industry to develop good software.

**Stages in SDLC:**

* Requirement Gathering
* Analysis
* Designing
* Coding
* Testing
* Maintenance

**Requirements Gathering stage:**

The goals identified in the high-level requirements section of the project plan. Each goal will be refined into a set of one or more requirements. These requirements define the major functions of the intended application, define operational data areas and reference data areas, and define the initial data entities. Major functions include critical processes to be managed, as well as mission critical inputs, outputs and reports. A user class hierarchy is developed and associated with these major functions, data areas, and data entities. Each of these definitions is termed a Requirement. Requirements are identified by unique requirement identifiers and, at minimum, contain a requirement title and textual description.



These requirements are fully described in the primary deliverables for this stage: the Requirements Document and the Requirements Traceability Matrix (RTM). The requirements document contains complete descriptions of each requirement, including diagrams and references to external documents as necessary. Note that detailed listings of database tables and fields are *not* included in the requirements document.

The title of each requirement is also placed into the first version of the RTM, along with the title of each goal from the project plan. The purpose of the RTM is to show that the product components developed during each stage of the software development lifecycle are formally connected to the components developed in prior stages.

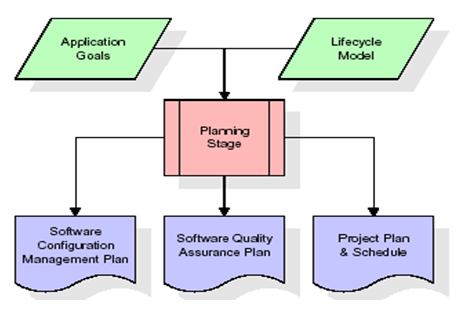
In the requirements stage, the RTM consists of a list of high-level requirements, or goals, by title, with a listing of associated requirements for each goal, listed by requirement title. In this hierarchical listing, the RTM shows that each requirement developed during this stage is formally linked to a specific product goal. In this format, each requirement can be traced to a specific product goal, hence the term requirements traceability.

The outputs of the requirements definition stage include the requirements document, the RTM, and an updated project plan.

* Feasibility study is all about identification of problems in a project.
* No. of staff required to handle a project is represented as Team Formation, in this case only modules are individual tasks will be assigned to employees who are working for that project.
* Project Specifications are all about representing of various possible inputs submitting to the server and corresponding outputs along with reports maintained by administrator.

**Analysis Stage:**

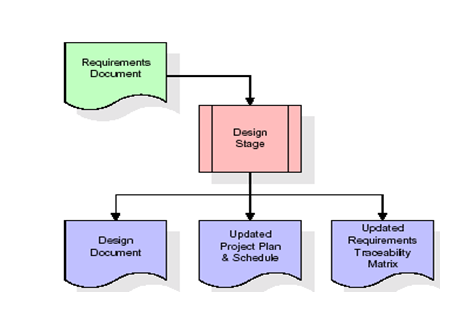
The planning stage establishes a bird's eye view of the intended software product, and uses this to establish the basic project structure, evaluate feasibility and risks associated with the project, and describe appropriate management and technical approaches.



The most critical section of the project plan is a listing of high-level product requirements, also referred to as goals. All of the software product requirements to be developed during the requirements definition stage flow from one or more of these goals. The minimum information for each goal consists of a title and textual description, although additional information and references to external documents may be included. The outputs of the project planning stage are the configuration management plan, the quality assurance plan, and the project plan and schedule, with a detailed listing of scheduled activities for the upcoming Requirements stage, and high level estimates of effort for the out stages.

**Designing Stage:**

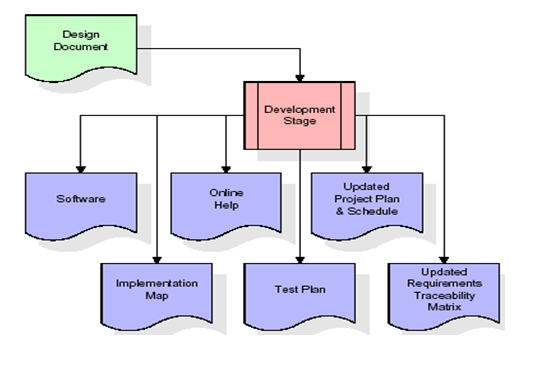
The design stage takes as its initial input the requirements identified in the approved requirements document. For each requirement, a set of one or more design elements will be produced as a result of interviews, workshops, and/or prototype efforts. Design elements describe the desired software features in detail, and generally include functional hierarchy diagrams, screen layout diagrams, tables of business rules, business process diagrams, pseudo code, and a complete entity-relationship diagram with a full data dictionary. These design elements are intended to describe the software in sufficient detail that skilled programmers may develop the software with minimal additional input.



When the design document is finalized and accepted, the RTM is updated to show that each design element is formally associated with a specific requirement. The outputs of the design stage are the design document, an updated RTM, and an updated project plan.

**Development (Coding) Stage:**

The development stage takes as its primary input the design elements described in the approved design document. For each design element, a set of one or more software artifacts will be produced. Software artifacts include but are not limited to menus, dialogs, and data management forms, data reporting formats, and specialized procedures and functions. Appropriate test cases will be developed for each set of functionally related software artifacts, and an online help system will be developed to guide users in their interactions with the software.



The RTM will be updated to show that each developed artefact is linked to a specific design element, and that each developed artefact has one or more corresponding test case items. At this point, the RTM is in its final configuration. The outputs of the development stage include a fully functional set of software that satisfies the requirements and design elements previously documented, an online help system that describes the operation of the software, an implementation map that identifies the primary code entry points for all major system functions, a test plan that describes the test cases to be used to validate the correctness and completeness of the software, an updated RTM, and an updated project plan.

**Integration & Test Stage:**

During the integration and test stage, the software artefacts, online help, and test data are migrated from the development environment to a separate test environment. At this point, all test cases are run to verify the correctness and completeness of the software. Successful execution of the test suite confirms a robust and complete migration capability. During this stage, reference data is finalized for production use and production users are identified and linked to their appropriate roles. The final reference data (or links to reference data source files) and production user list are compiled into the Production Initiation Plan.



The outputs of the integration and test stage include an integrated set of software, an online help system, an implementation map, a production initiation plan that describes reference data and production users, an acceptance plan which contains the final suite of test cases, and an updated project plan.

* **Installation & Acceptance Test:**

During the installation and acceptance stage, the software artifacts, online help, and initial production data are loaded onto the production server. At this point, all test cases are run to verify the correctness and completeness of the software. Successful execution of the test suite is a prerequisite to acceptance of the software by the customer.

After customer personnel have verified that the initial production data load is correct and the test suite has been executed with satisfactory results, the customer formally accepts the delivery of the software.



The primary outputs of the installation and acceptance stage include a production application, a completed acceptance test suite, and a memorandum of customer acceptance of the software. Finally, the PDR enters the last of the actual labor data into the project schedule and locks the project as a permanent project record. At this point the PDR "locks" the project by archiving all software items, the implementation map, the source code, and the documentation for future reference.

**Maintenance:**

Outer rectangle represents maintenance of a project, Maintenance team will start with requirement study, understanding of documentation later employees will be assigned work and they will undergo training on that particular assigned category. For this life cycle there is no end, it will be continued so on like an umbrella (no ending point to umbrella sticks).

**3.4. Software Requirement Specification**

**3.4.1. Overall Description**

A Software Requirements Specification (SRS) – a [requirements specification](http://en.wikipedia.org/wiki/Requirements_specification) for a [software system](http://en.wikipedia.org/wiki/Software_system) is a complete description of the behaviour of a system to be developed. It includes a set of [use cases](http://en.wikipedia.org/wiki/Use_case) that describe all the interactions the users will have with the software. In addition to use cases, the SRS also contains non-functional requirements. [Non-functional requirements](http://en.wikipedia.org/wiki/Non-functional_requirements) are requirements which impose constraints on the design or implementation (such as [performance engineering](http://en.wikipedia.org/wiki/Performance_engineering) requirements, [quality](http://en.wikipedia.org/wiki/Quality_%28business%29) standards, or design constraints).

System requirements specification: A structured collection of information that embodies the requirements of a system. A [business analyst](http://en.wikipedia.org/wiki/Business_analyst), sometimes titled [system analyst](http://en.wikipedia.org/wiki/System_analyst), is responsible for analyzing the business needs of their clients and stakeholders to help identify business problems and propose solutions. Within the [systems development lifecycle](http://en.wikipedia.org/wiki/Systems_development_life_cycle) domain, the BA typically performs a liaison function between the business side of an enterprise and the information technology department or external service providers. Projects are subject to three sorts of requirements:

* [Business requirements](http://en.wikipedia.org/wiki/Business_requirements) describe in business terms what must be delivered or accomplished to provide value.
* Product requirements describe properties of a system or product (which could be one of several ways to accomplish a set of business requirements.)
* Process requirements describe activities performed by the developing organization. For instance, process requirements could specify .Preliminary investigation examine project feasibility, the likelihood the system will be useful to the organization. The main objective of the feasibility study is to test the Technical, Operational and Economical feasibility for adding new modules and debugging old running system. All system is feasible if they are unlimited resources and infinite time. There are aspects in the feasibility study portion of the preliminary investigation:
* **ECONOMIC FEASIBILITY**

A system can be developed technically and that will be used if installed must still be a good investment for the organization. In the economical feasibility, the development cost in creating the system is evaluated against the ultimate benefit derived from the new systems. Financial benefits must equal or exceed the costs. The system is economically feasible. It does not require any addition hardware or software. Since the interface for this system is developed using the existing resources and technologies available at NIC, There is nominal expenditure and economical feasibility for certain.

* **Operational Feasibility**

Proposed projects are beneficial only if they can be turned out into information system. That will meet the organization’s operating requirements. Operational feasibility aspects of the project are to be taken as an important part of the project implementation. This system is targeted to be in accordance with the above-mentioned issues. Beforehand, the management issues and user requirements have been taken into consideration. So there is no question of resistance from the users that can undermine the possible application benefits. The well-planned design would

Ensure the optimal utilization of the computer resources and would help in the improvement of performance status.

* **TECHNICAL FEASIBILITY**

Earlier no system existed to cater to the needs of ‘Secure Infrastructure Implementation System’. The current system developed is technically feasible. It is a web based user interface for audit workflow at NIC-CSD. Thus it provides an easy access to .the users. The database’s purpose is to create, establish and maintain a workflow among various entities in order to facilitate all concerned users in their various capacities or roles. Permission to the users would be granted based on the roles specified. Therefore, it provides the technical guarantee of accuracy, reliability and security.

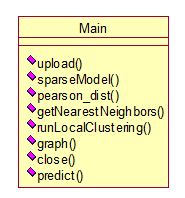
**4. SYSTEM DESIGN**

**UML Diagram:**

**Class Diagram:**

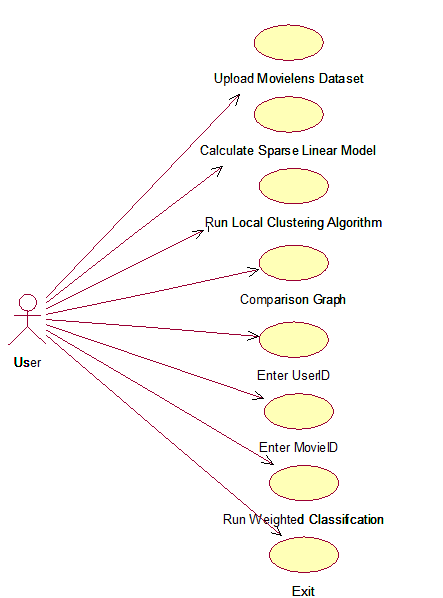
The class diagram is the main building block of object oriented modeling. It is used both for general conceptual modeling of the systematic of the application, and for detailed modeling translating the models into programming code. Class diagrams can also be used for data modeling. The classes in a class diagram represent both the main objects, interactions in the application and the classes to be programmed. In the diagram, classes are represented with boxes which contain three parts:

* The upper part holds the name of the class
* The middle part contains the attributes of the class
* The bottom part gives the methods or operations the class can take or undertake.



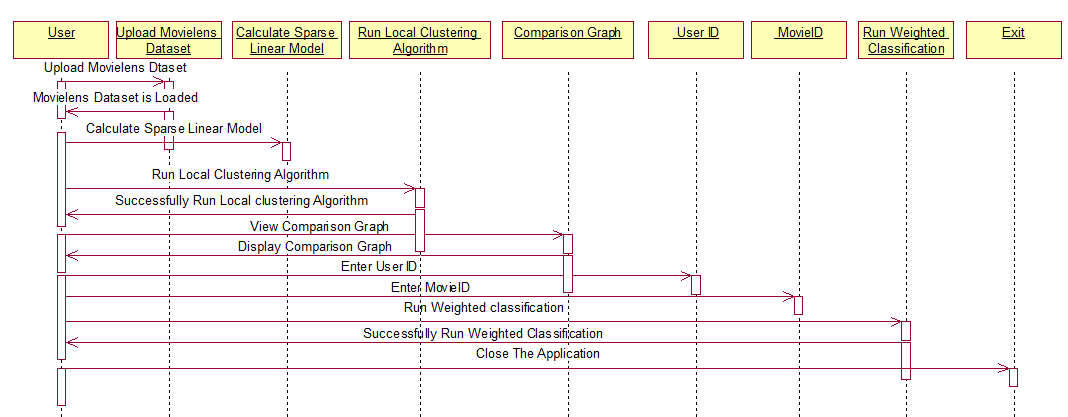
**Use case Diagram:**

A use case diagram at its simplest is a representation of a user's interaction with the system and depicting the specifications of a use case. A use case diagram can portray the different types of users of a system and the various ways that they interact with the system. This type of diagram is typically used in conjunction with the textual use case and will often be accompanied by other types of diagrams as well.



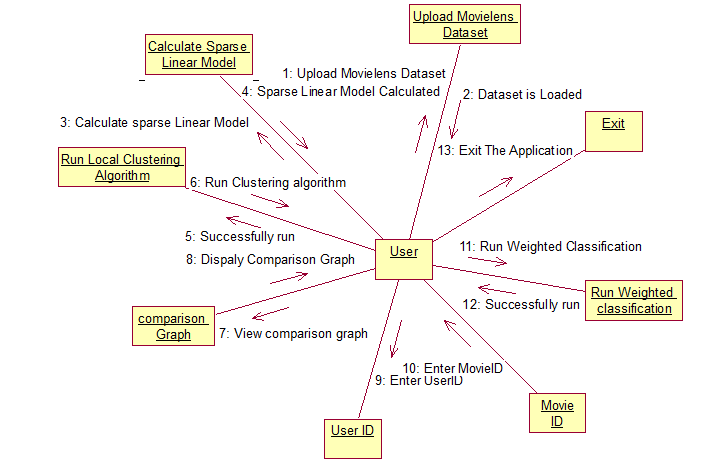
**Sequence Diagram:**

A sequence diagram is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.



**Collaboration diagram:**

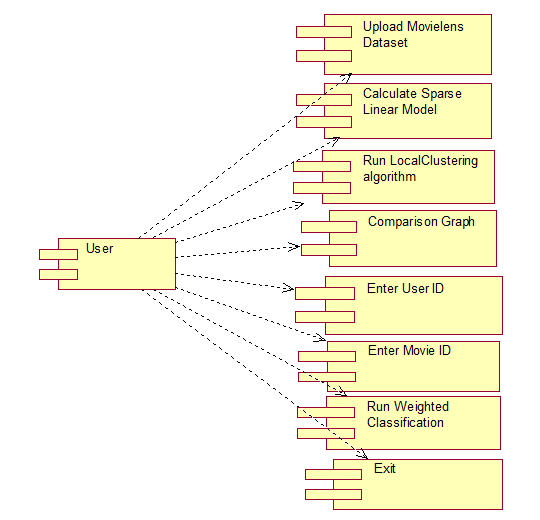
A collaboration diagram describes interactions among objects in terms of sequenced messages. Collaboration diagrams represent a combination of information taken from class, sequence, and use case diagrams describing both the static structure and dynamic behaviour of a system.



**Component Diagram:**

In the Unified Modelling Language, a component diagram depicts how components are wired together to form larger components and or software systems. They are used to illustrate the structure of arbitrarily complex systems.

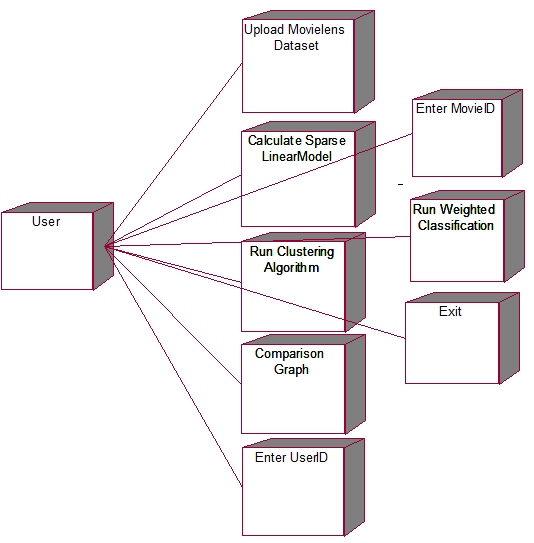
Components are wired together by using an assembly connector to connect the required interface of one component with the provided interface of another component. This illustrates the service consumer - service provider relationship between the two components.



**Deployment Diagram:**

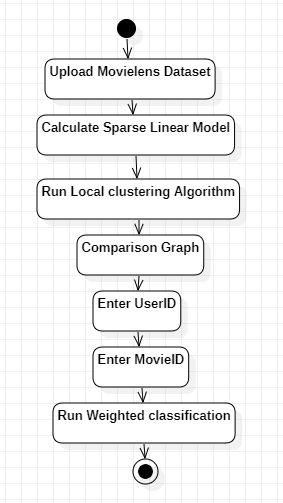
A deployment diagram in the Unified Modeling Language models the physical deployment of artifacts on nodes. To describe a web site, for example, a deployment diagram would show what hardware components ("nodes") exist (e.g., a web server, an application server, and a database server), what software components ("artifacts") run on each node (e.g., web application, database), and how the different pieces are connected (e.g. JDBC, REST, RMI).

The nodes appear as boxes, and the artifacts allocated to each node appear as rectangles within the boxes. Nodes may have sub nodes, which appear as nested boxes. A single node in a deployment diagram may conceptually represent multiple physical nodes, such as a cluster of database servers.



**Activity Diagram:**

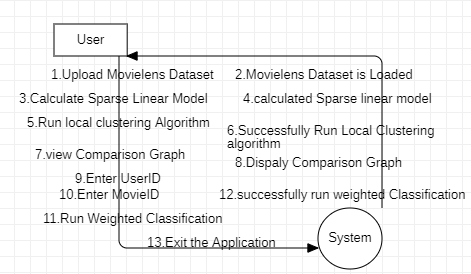
Activity diagram is another important diagram in UML to describe dynamic aspects of the system. It is basically a flow chart to represent the flow form one activity to another activity. The activity can be described as an operation of the system. So the control flow is drawn from one operation to another. This flow can be sequential, branched or concurrent.



**Data Flow Diagram:**

Data flow diagrams illustrate how data is processed by a system in terms of inputs and outputs. Data flow diagrams can be used to provide a clear representation of any business function. The technique starts with an overall picture of the business and continues by analyzing each of the functional areas of interest. This analysis can be carried out in precisely the level of detail required. The technique exploits a method called top-down expansion to conduct the analysis in a targeted way.

As the name suggests, Data Flow Diagram (DFD) is an illustration that explicates the passage of information in a process. A DFD can be easily drawn using simple symbols. Additionally, complicated processes can be easily automated by creating DFDs using easy-to-use, free downloadable diagramming tools. A DFD is a model for constructing and analyzing information processes. DFD illustrates the flow of information in a process depending upon the inputs and outputs. A DFD can also be referred to as a Process Model. A DFD demonstrates business or technical process with the support of the outside data saved, plus the data flowing from the process to another and the end results.



**5. IMPLEMETATION**

**5.1 Python**

Python is a general-purpose language. It has wide range of applications from Web development (like: Django and Bottle), scientific and mathematical computing (Orange, SymPy, NumPy) to desktop graphical user Interfaces (Pygame, Panda3D). The syntax of the language is clean and length of the code is relatively short. It's fun to work in Python because it allows you to think about the problem rather than focusing on the syntax.

**History of Python:**

Python is a fairly old language created by Guido Van Rossum. The design began in the late 1980s and was first released in February 1991.

**Why Python was created?**

In late 1980s, Guido Van Rossum was working on the Amoeba distributed operating system group. He wanted to use an interpreted language like ABC (ABC has simple easy-to-understand syntax) that could access the Amoeba system calls. So, he decided to create a language that was extensible. This led to design of a new language which was later named Python.

**Why the name Python?**

No. It wasn't named after a dangerous snake. Rossum was fan of a comedy series from late seventies. The name "Python" was adopted from the same series "Monty Python's Flying Circus".

**Features of Python:**

**A simple language which is easier to learn**

Python has a very simple and elegant syntax. It's much easier to read and write Python programs compared to other languages like: C++, Java, C#. Python makes programming fun and allows you to focus on the solution rather than syntax.

If you are a newbie, it's a great choice to start your journey with Python.

**Free and open-source**

You can freely use and distribute Python, even for commercial use. Not only can you use and distribute software’s written in it, you can even make changes to the Python's source code.

Python has a large community constantly improving it in each iteration.

**Portability**

You can move Python programs from one platform to another, and run it without any changes.

It runs seamlessly on almost all platforms including Windows, Mac OS X and Linux.

**Extensible and Embeddable**

Suppose an application requires high performance. You can easily combine pieces of C/C++ or other languages with Python code.

This will give your application high performance as well as scripting capabilities which other languages may not provide out of the box.

**A high-level, interpreted language**

Unlike C/C++, you don't have to worry about daunting tasks like memory management, garbage collection and so on.

Likewise, when you run Python code, it automatically converts your code to the language your computer understands. You don't need to worry about any lower-level operations.

**Large standard libraries to solve common tasks**

Python has a number of standard libraries which makes life of a programmer much easier since you don't have to write all the code yourself. For example: Need to connect MySQL database on a Web server? You can use MySQLdb library using import MySQLdb .

Standard libraries in Python are well tested and used by hundreds of people. So you can be sure that it won't break your application.

**Object-oriented**

Everything in Python is an object. Object oriented programming (OOP) helps you solve a complex problem intuitively.

With OOP, you are able to divide these complex problems into smaller sets by creating objects.

**Applications of Python:**

**1. Simple Elegant Syntax**

Programming in Python is fun. It's easier to understand and write Python code. Why? The syntax feels natural. Take this source code for an example:

a = 2

b = 3

sum = a + b

print(sum)

**2. Not overly strict**

You don't need to define the type of a variable in Python. Also, it's not necessary to add semicolon at the end of the statement.

Python enforces you to follow good practices (like proper indentation). These small things can make learning much easier for beginners.

**3. Expressiveness of the language**

Python allows you to write programs having greater functionality with fewer lines of code. Here's a link to the source code of Tic-tac-toe game with a graphical interface and a smart computer opponent in less than 500 lines of code. This is just an example. You will be amazed how much you can do with Python once you learn the basics.

**4. Great Community and Support**

Python has a large supporting community. There are numerous active forums online which can be handy if you are stuck.

**5.2 Sample Code:**

**HybridMovieRecommendation.py**

from tkinter import messagebox

from tkinter import \*

from tkinter import simpledialog

import tkinter

from tkinter import filedialog

import matplotlib.pyplot as plt

from tkinter.filedialog import askopenfilename

import numpy as np

import os

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from pyclustering.cluster.kmeans import kmeans

from pyclustering.utils.metric import distance\_metric, type\_metric

from sklearn.preprocessing import normalize

from scipy import stats

main = tkinter.Tk()

main.title("Optimization of the Hybrid Movie Recommendation System Based on Weighted Classification and User Collaborative Filtering Algorithm")

main.geometry("1300x1200")

global filename, cluster, X, Y

global hr, arhr, movies, users, ratings

global X\_train, X\_test, y\_train, y\_test, cs, kmeans\_instance, metric

def upload():

global filename, movies, users, ratings

filename = filedialog.askdirectory(initialdir=".")

text.delete('1.0', END)

text.insert(END,filename+" loaded\n\n")

#reading movies, ratings and user dataset

ratings = pd.read\_csv('Dataset/ratings.csv', nrows=10000,sep='\t', encoding='latin-1', usecols=['user\_id', 'movie\_id', 'rating'])

movies = pd.read\_csv('Dataset/movies.csv',encoding='latin-1',sep='\t', usecols=['movie\_id', 'title', 'genres'])

users = pd.read\_csv('Dataset/users.csv', encoding='latin-1',sep='\t')

#replacing missing values with 0

ratings.fillna(0, inplace = True)

text.insert(END, "User Details \n\n"+str(users.head())+"\n\n")

text.insert(END, "Movie Details \n\n"+str(movies.head())+"\n\n")

text.insert(END, "Rating Details \n\n"+str(ratings.head())+"\n\n")

def sparseModel():

global ratings, X, Y

global X\_train, X\_test, y\_train, y\_test

text.delete('1.0', END)

ratings.fillna(0, inplace = True)

dataset = ratings.values

#converting ratings in to sparse matrix

matrix = ratings.values

#extracting user and movie id as X features and RATINGS as Y and this features are called as sparse matrix

X = matrix[:,0:2]

Y = matrix[:,2]

text.insert(END,"Sparse Matrix Extracted from Dataset\n\n")

text.insert(END,str(X)+"\n\n")

text.insert(END,"Total records found in dataset : "+str(X.shape[0])+"\n")

text.insert(END,"Total features found in dataset: "+str(X.shape[1])+"\n\n")

X = normalize(X)

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, Y, test\_size=0.2)

text.insert(END,"Dataset Train and Test Split\n\n")

text.insert(END,"80% dataset records used to train ML algorithms : "+str(X\_train.shape[0])+"\n")

text.insert(END,"20% dataset records used to train ML algorithms : "+str(X\_test.shape[0])+"\n")

def pearson\_dist(x, y): #function to calculate similarity between X user and Y user

x = np.asarray(x)

y = np.asarray(y)

a\_norm = np.linalg.norm(x)

b\_norm = np.linalg.norm(y)

stats.pearsonr(x, y)[0]

similiarity = np.dot(x, y.T)/(a\_norm \* b\_norm)

similiarity = 1. - similiarity

return similiarity #return pearson similarity

def getNearestNeighbors(test): #function to calculate nearest neighbour using weighted threshold

list\_of\_test\_users = np.zeros((len(test), 5))

for index\_point in range(len(test)): #loop all test users and then find users similarity

list\_of\_test\_users[index\_point] = [metric(test[index\_point], c) for c in cs]

label = np.argmax(list\_of\_test\_users, axis=1) #extract maximum similarity user as label

return label #return maximum similarity user value

def runLocalClustering():

global ratings, X, Y, cs, hr, arhr, metric, kmeans\_instance

global X\_train, X\_test, y\_train, y\_test

hr = []

arhr = []

text.delete('1.0', END)

metric = distance\_metric(type\_metric.USER\_DEFINED, func=pearson\_dist) #here we are creating distance similarity measure with pearson function

#defining initial center points

initial\_centers = [[0.023917746175045114, 0.9995049723305172], [0.8574672718847574, 0.49772130656070934],

[0.5351297755937158, 0.8354536941067743], [0.988826874252877, 0.12248444970535248], [0.18061215808615247, 0.9812169115929825]]

kmeans\_instance = kmeans(X\_train, initial\_centers, metric=metric, tolerance = 0.01, ccore = False)#creating KMEANS with given X train data, initial center and metric

kmeans\_instance.process() #start grouping similar behaviour user into same cluster

clusters = kmeans\_instance.get\_clusters() #return number of clusters

cs = initial\_centers

label = kmeans\_instance.predict(X\_train)#local hit rate will be calculated by using local KMEANS clustering object

for i in range(len(label)):

label[i] = label[i] + 1

hitrate = 0

for i in range(len(label)):

if label[i] == y\_train[i]:

hitrate += 1

total\_test\_users = len(X\_test) / 2

hitrate = (hitrate / total\_test\_users)

hr.append(hitrate)

avg = hitrate / 100

arhr.append(avg)

text.insert(END,"Local KMEANS Clustering Computation Completed\n\n")

text.insert(END,"Propose TopPop Algorithm Local Hit Rate: "+str(hitrate)+"\n")

text.insert(END,"Propose TopPop Algorithm Local Avg Hit Rate: "+str(avg)+"\n\n")

label = getNearestNeighbors(X\_test) #global hit rate calculation using maximum similarity by applying nearest neighbour formula with max weight

for i in range(len(label)):

label[i] = label[i] + 1

hitrate = 0

for i in range(len(label)):

if label[i] == y\_test[i]:

hitrate += 1

hitrate = hitrate / len(X\_test)

hr.append(hitrate)

avg = hitrate / 100

arhr.append(avg)

text.insert(END,"Propose TopPop Algorithm Global Hit Rate: "+str(hitrate)+"\n")

text.insert(END,"Propose TopPop Algorithm Global Avg Hit Rate: "+str(avg)+"\n")

def graph():

global hr, arhr

height = [hr[0], arhr[0], hr[1], arhr[1]]

bars = ('Local HR','Local ARHR', 'Global HR', 'Global ARHR')

y\_pos = np.arange(len(bars))

plt.bar(y\_pos, height)

plt.xticks(y\_pos, bars)

plt.title("HR & ARHR Comparison Graph")

plt.show()

def close():

main.destroy()

def predict():

global kmeans\_instance, X, Y, movies, ratings

text.delete('1.0', END)

user\_id = int(text1.get("1.0",END))

movie\_id = int(text2.get("1.0",END))

test = []

test.append([user\_id, movie\_id])

test = np.asarray(test)

test = normalize(test)

predict = kmeans\_instance.predict(test)[0]

predict = predict + 1

print(predict)

recommendationList = []

movieList = movies.values

ratingList = ratings.values

for i in range(len(Y)):

if Y[i] == predict:

movieID = ratingList[i,1]

for k in range(len(movieList)):

if movieList[k,0] == movieID:

recommendationList.append(movieList[k,1])

if len(recommendationList) >= 10:

break

if len(recommendationList) > 0:

text.insert(END,"Predicted Rating Score is : "+str(predict)+"\n\n")

text.insert(END,"Below are the Recommended Movie List\n\n")

for k in range(len(recommendationList)):

text.insert(END,recommendationList[k]+"\n")

else:

text.insert(END,"Unable to find recommendation for given user")

font = ('times', 16, 'bold')

title = Label(main, text='Optimization of the Hybrid Movie Recommendation System Based on Weighted Classification and User Collaborative Filtering Algorithm')

title.config(bg='firebrick4', fg='dodger blue')

title.config(font=font)

title.config(height=3, width=120)

title.place(x=0,y=5)

font1 = ('times', 12, 'bold')

text=Text(main,height=20,width=150)

scroll=Scrollbar(text)

text.configure(yscrollcommand=scroll.set)

text.place(x=50,y=120)

text.config(font=font1)

font1 = ('times', 13, 'bold')

uploadButton = Button(main, text="Upload Movielens Dataset", command=upload, bg='#ffb3fe')

uploadButton.place(x=50,y=550)

uploadButton.config(font=font1)

processButton = Button(main, text="Calculate Sparse Linear Model", command=sparseModel, bg='#ffb3fe')

processButton.place(x=310,y=550)

processButton.config(font=font1)

knnButton = Button(main, text="Run Local Clustering Algorithm", command=runLocalClustering, bg='#ffb3fe')

knnButton.place(x=600,y=550)

knnButton.config(font=font1)

graphButton = Button(main, text="Comparison Graph", command=graph, bg='#ffb3fe')

graphButton.place(x=900,y=550)

graphButton.config(font=font1)

l1 = Label(main, text='User ID:')

l1.config(font=font1)

l1.place(x=50,y=600)

text1=Text(main,height=2,width=20)

scroll1=Scrollbar(text1)

text1.configure(yscrollcommand=scroll1.set)

text1.place(x=160,y=600)

text1.config(font=font1)

l2 = Label(main, text='Movie ID:')

l2.config(font=font1)

l2.place(x=370,y=600)

text2=Text(main,height=2,width=20)

scroll2=Scrollbar(text2)

text2.configure(yscrollcommand=scroll2.set)

text2.place(x=500,y=600)

text2.config(font=font1)

predictButton = Button(main, text="Run Weighted Classification", command=predict, bg='#ffb3fe')

predictButton.place(x=750,y=600)

predictButton.config(font=font1)

exitButton = Button(main, text="Exit", command=close, bg='#ffb3fe')

exitButton.place(x=50,y=650)

exitButton.config(font=font1)

main.config(bg='LightSalmon3')

main.mainloop()

**6. TESTING**

**Implementation and Testing:**

Implementation is one of the most important tasks in project is the phase in which one has to be cautions because all the efforts undertaken during the project will be very interactive. Implementation is the most crucial stage in achieving successful system and giving the users confidence that the new system is workable and effective. Each program is tested individually at the time of development using the sample data and has verified that these programs link together in the way specified in the program specification. The computer system and its environment are tested to the satisfaction of the user.

**Implementation**

The implementation phase is less creative than system design. It is primarily concerned with user training, and file conversion. The system may be requiring extensive user training. The initial parameters of the system should be modifies as a result of a programming. A simple operating procedure is provided so that the user can understand the different functions clearly and quickly. The different reports can be obtained either on the inkjet or dot matrix printer, which is available at the disposal of the user. The proposed system is very easy to implement. In general implementation is used to mean the process of converting a new or revised system design into an operational one.

## Testing

Testing is the process where the test data is prepared and is used for testing the modules individually and later the validation given for the fields. Then the system testing takes place which makes sure that all components of the system property functions as a unit. The test data should be chosen such that it passed through all possible condition. Actually testing is the state of implementation which aimed at ensuring that the system works accurately and efficiently before the actual operation commence. The following is the description of the testing strategies, which were carried out during the testing period.

**System Testing**

Testing has become an System integral part of any system or project especially in the field of information technology. The importance of testing is a method of justifying, if one is ready to move further, be it to be check if one is capable to with stand the rigors of a particular situation cannot be underplayed and that is why testing before development is so critical. When the software is developed before it is given to user to user the software must be tested whether it is solving the purpose for which it is developed. This testing involves various types through which one can ensure the software is reliable. The program was tested logically and pattern of execution of the program for a set of data are repeated. Thus the code was exhaustively checked for all possible correct data and the outcomes were also checked.

**Module Testing**

To locate errors, each module is tested individually. This enables us to detect error and correct it without affecting any other modules. Whenever the program is not satisfying the required function, it must be corrected to get the required result. Thus all the modules are individually tested from bottom up starting with the smallest and lowest modules and proceeding to the next level. Each module in the system is tested separately. For example the job classification module is tested separately. This module is tested with different job and its approximate execution time and the result of the test is compared with the results that are prepared manually. The comparison shows that the results proposed system works efficiently than the existing system. Each module in the system is tested separately. In this system the resource classification and job scheduling modules are tested separately and their corresponding results are obtained which reduces the process waiting time.

**Integration Testing**

After the module testing, the integration testing is applied. When linking the modules there may be chance for errors to occur, these errors are corrected by using this testing. In this system all modules are connected and tested. The testing results are very correct. Thus the mapping of jobs with resources is done correctly by the system.

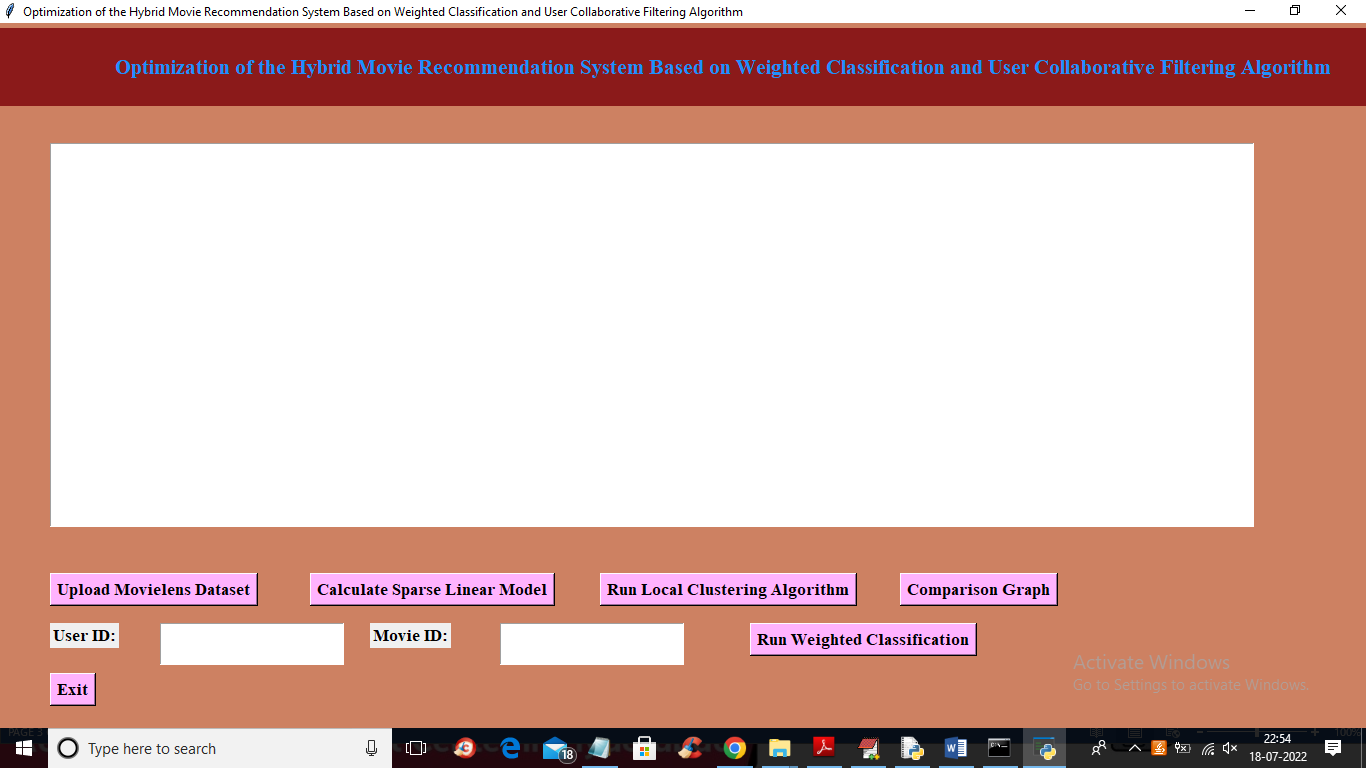
**Acceptance Testing**

When that user fined no major problems with its accuracy the system passers through a final acceptance test. This test confirms that the system needs the original goals, objectives and requirements established during analysis without actual execution which elimination wastage of time and money acceptance tests on the shoulders of users and management, it is finally acceptable and ready for the operation.

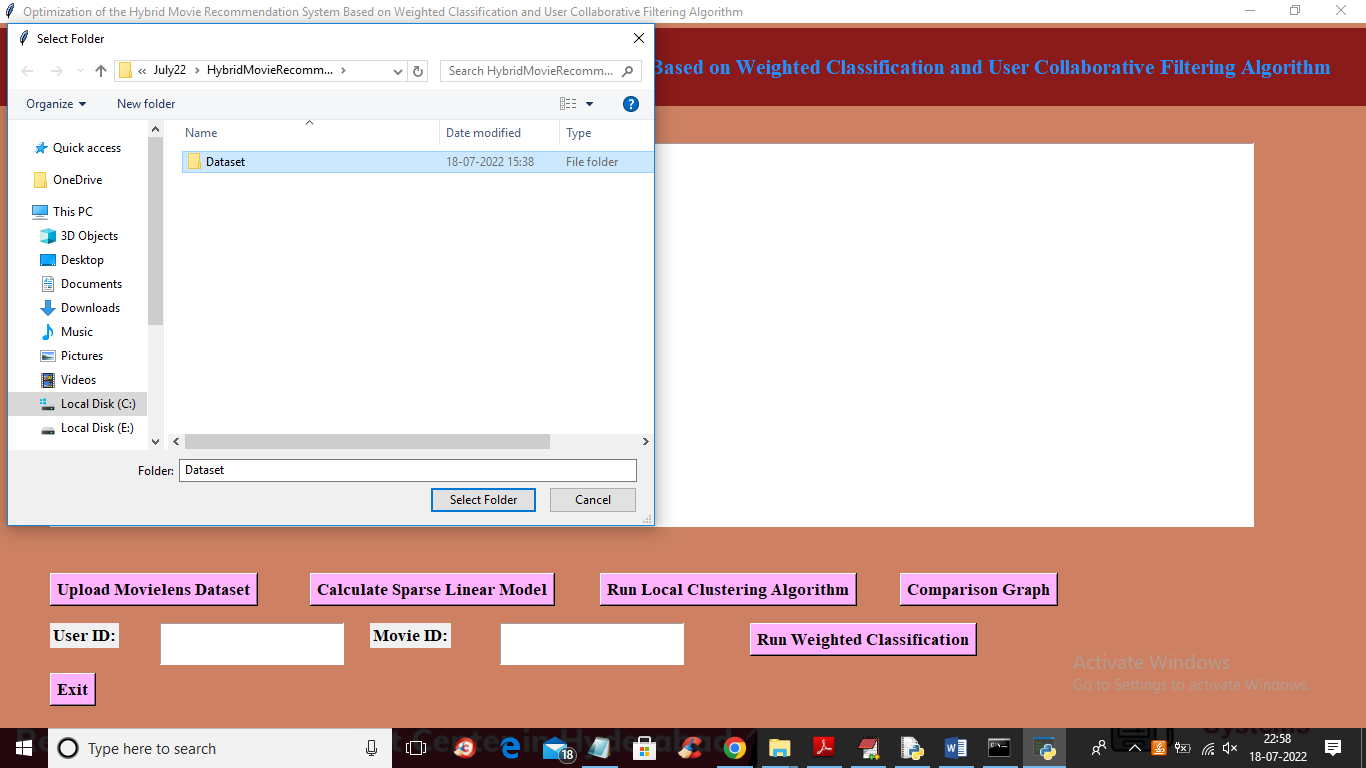
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case Id** | **Test Case Name** | **Test Case Desc.** | **Test Steps** | | | **Test Case Status** | **Test Priority** |
| **Step** | **Expected** | **Actual** |
| 01 | Upload Movie-lens Dataset | Verify the  Either  Movie-lens Dataset is uploaded or not | If  Dataset  May Not be uploaded | We cannot do the further operations | Dataset is Uploaded | High | High |
| 02 | Calculate sparse Linear Model | Verify sparse linear model calculated or not | If Dataset May not be uploaded | We cannot calculated sparse linear model | We can calculated sparse linear model | High | High |
| 03 | Run Local Clustering Algorithm | Verify The Local Clustering algorithm run or not | If the Local Clustering algorithm May Not be trained | We cannot run Local Clustering algorithm | We can run Local Clustering algorithm | High | High |
| 03 | Comparison Graph | Verify comparison graph displayed or Not | Without saving algorithm values | We cannot displayed comparison graph | We can displayed comparison graph | High | High |

## 7. SCREEN SHOTS:

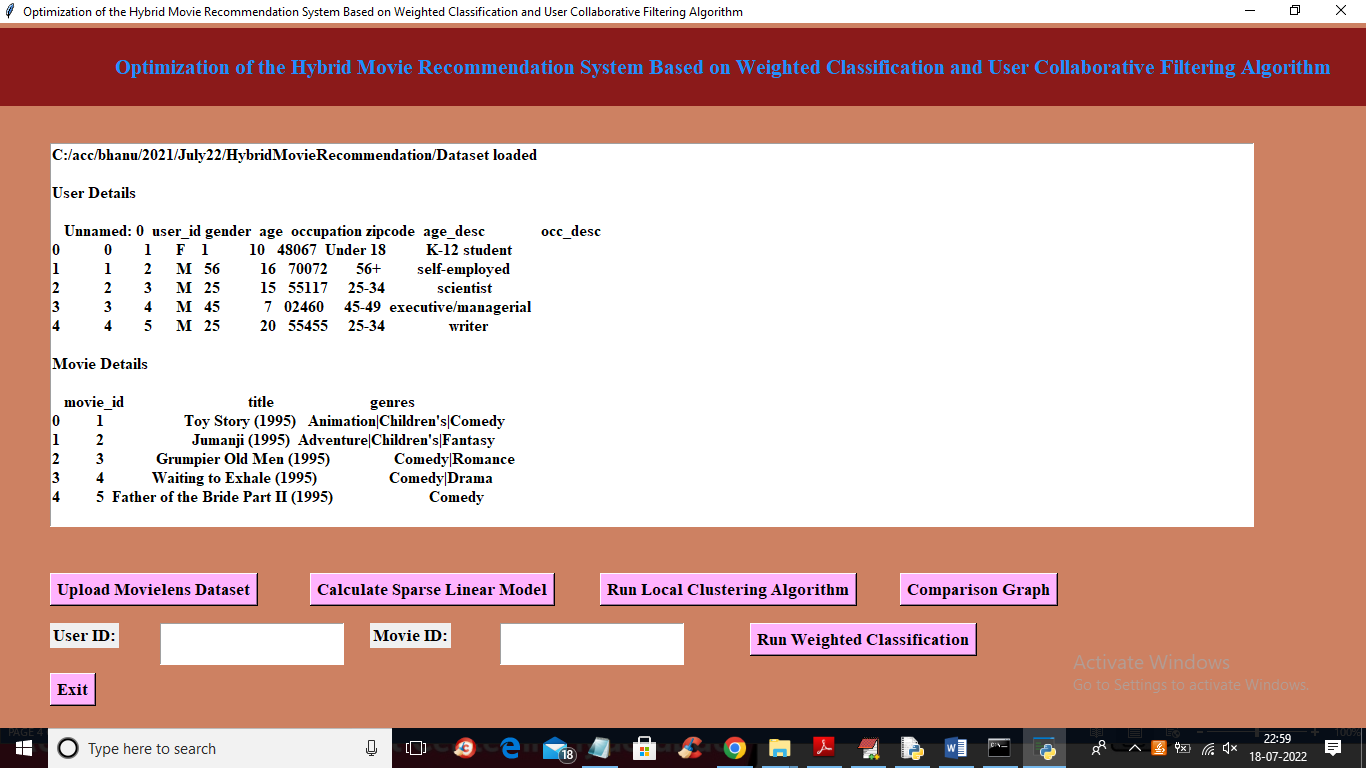
To run project double click on ‘run.bat’ file to get below screen



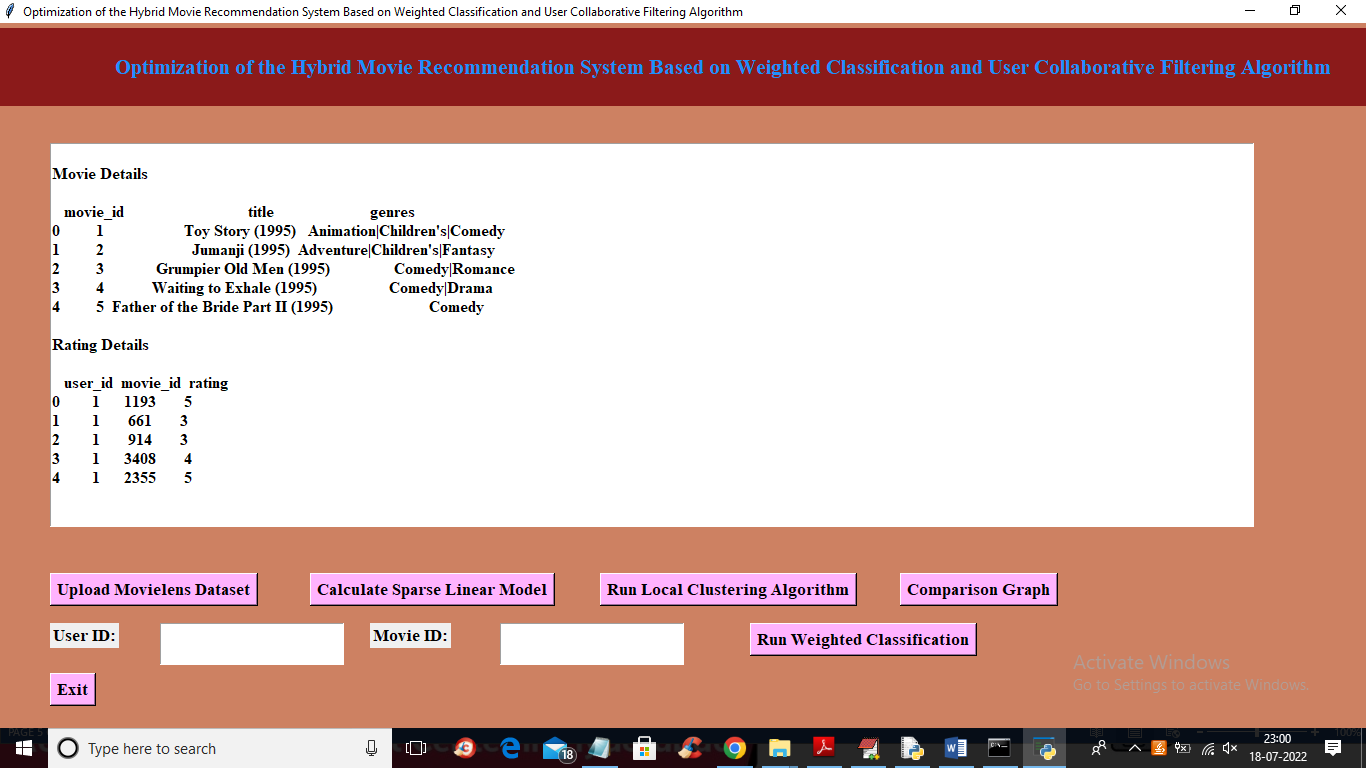
In above screen click on ‘Upload Movie Lens Dataset’ button to upload dataset and get below output



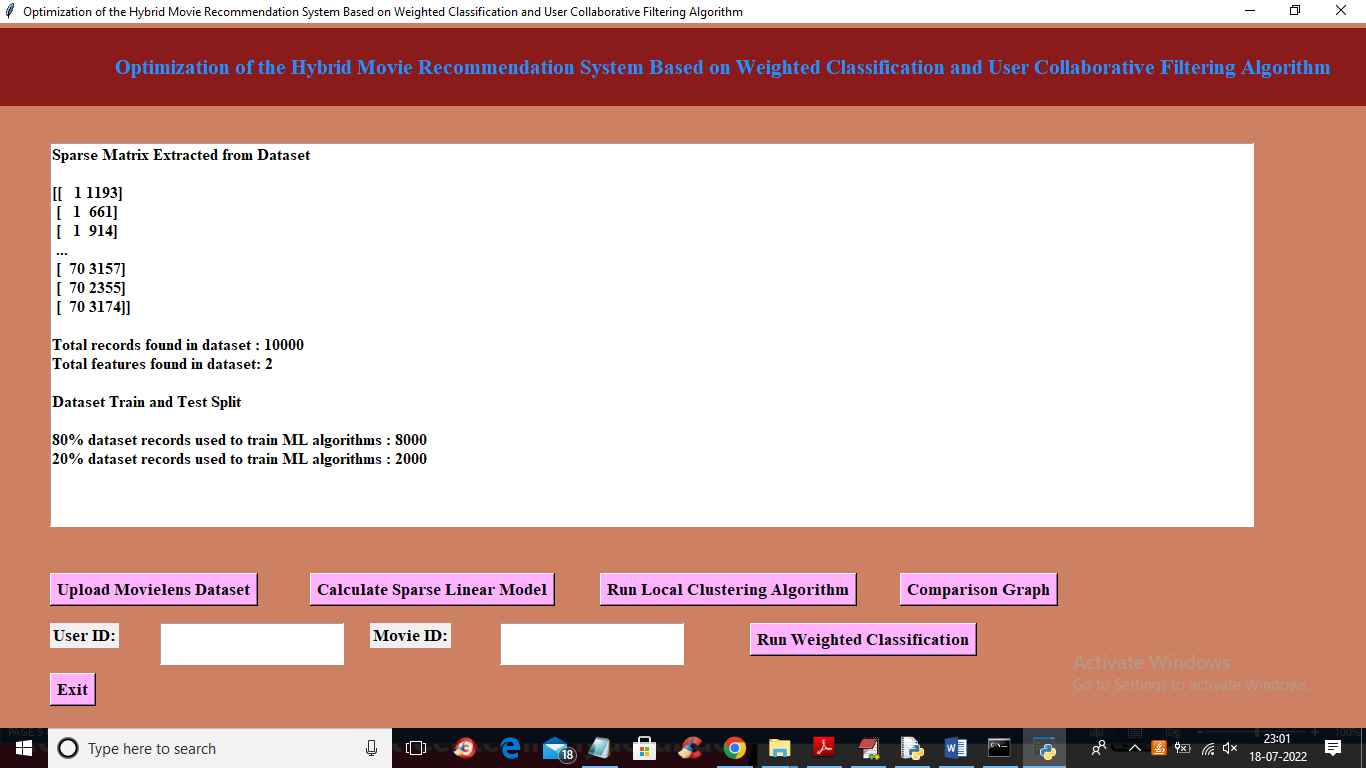
In above screen selecting and uploading ‘Dataset’ folder and then click on ‘Select Folder’ button to upload dataset and get below output



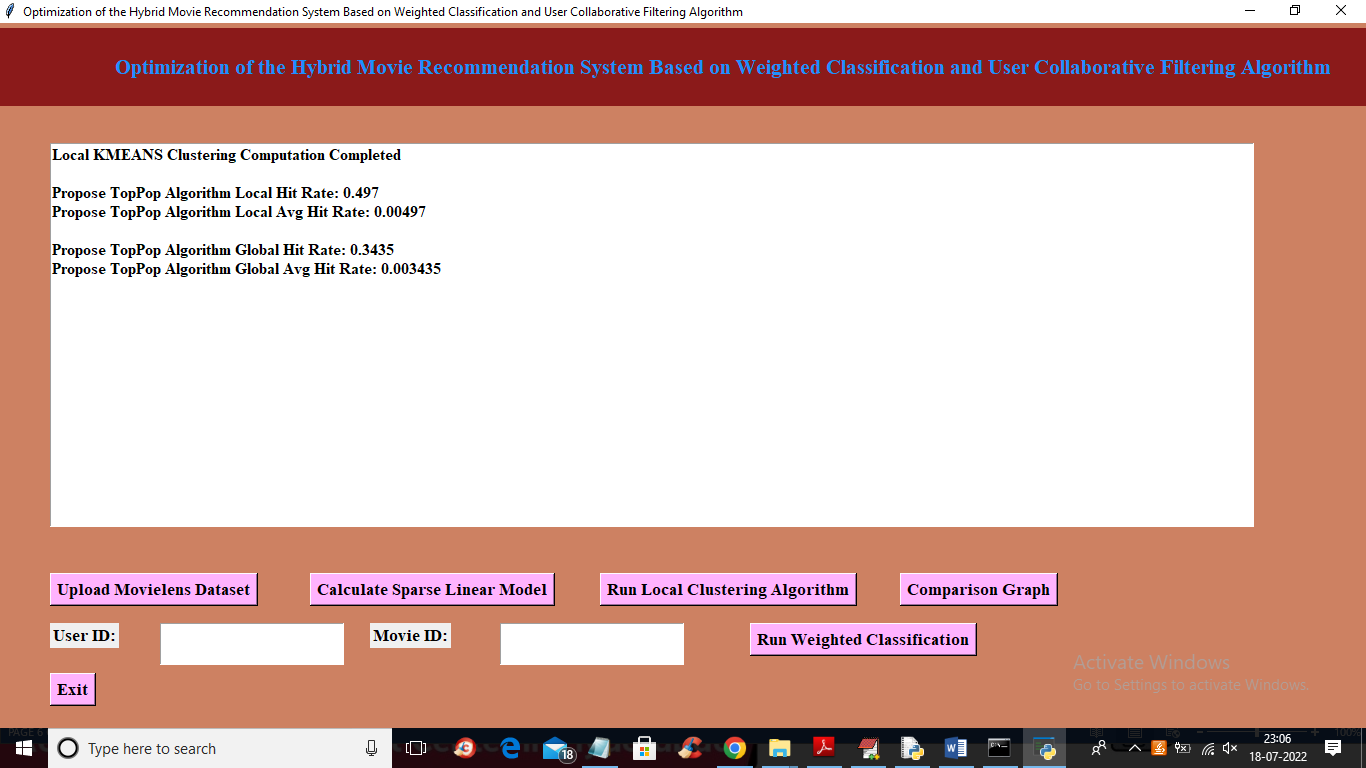
In above screen user details and movie details are loaded and scroll down above screen to view RATING details



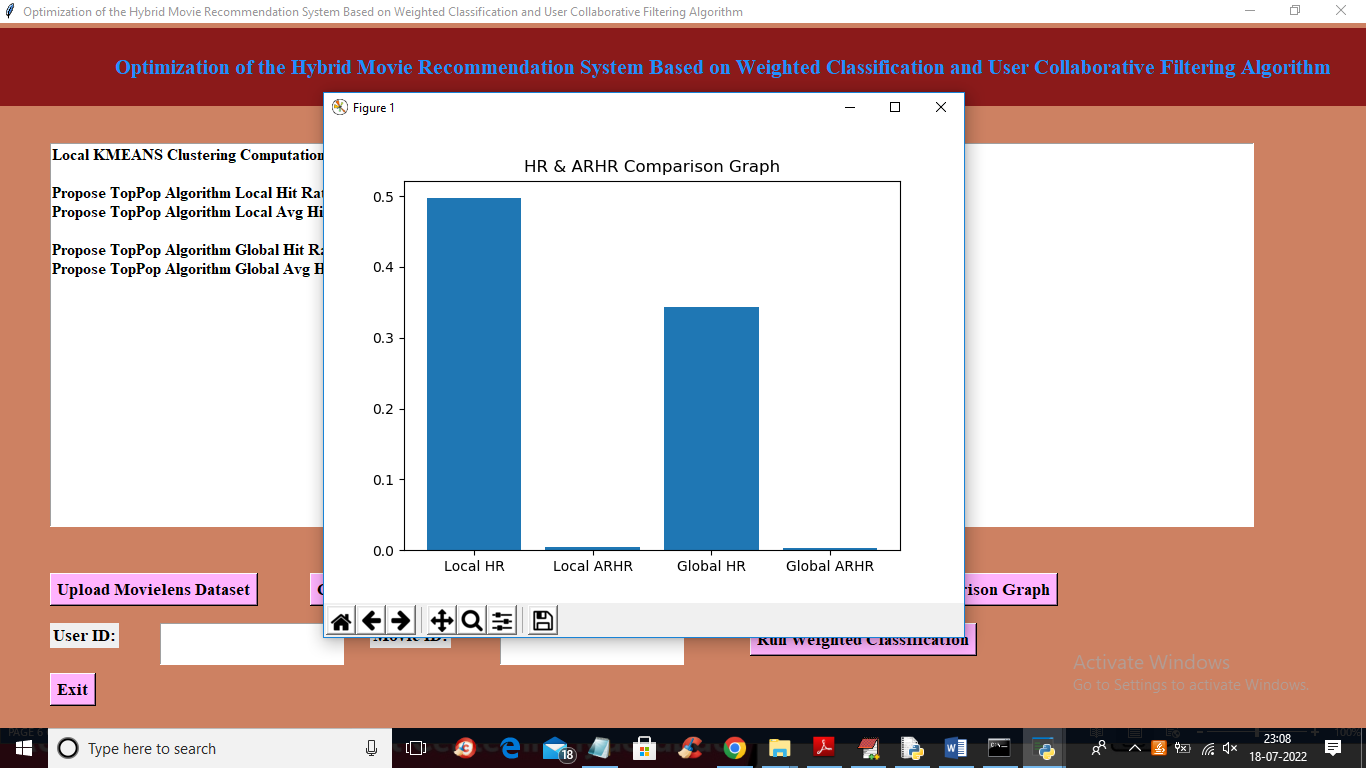
In above screen Rating details also loaded and now click on ‘Calculate Sparse Linear Model’ button to extract sparse features and get below output



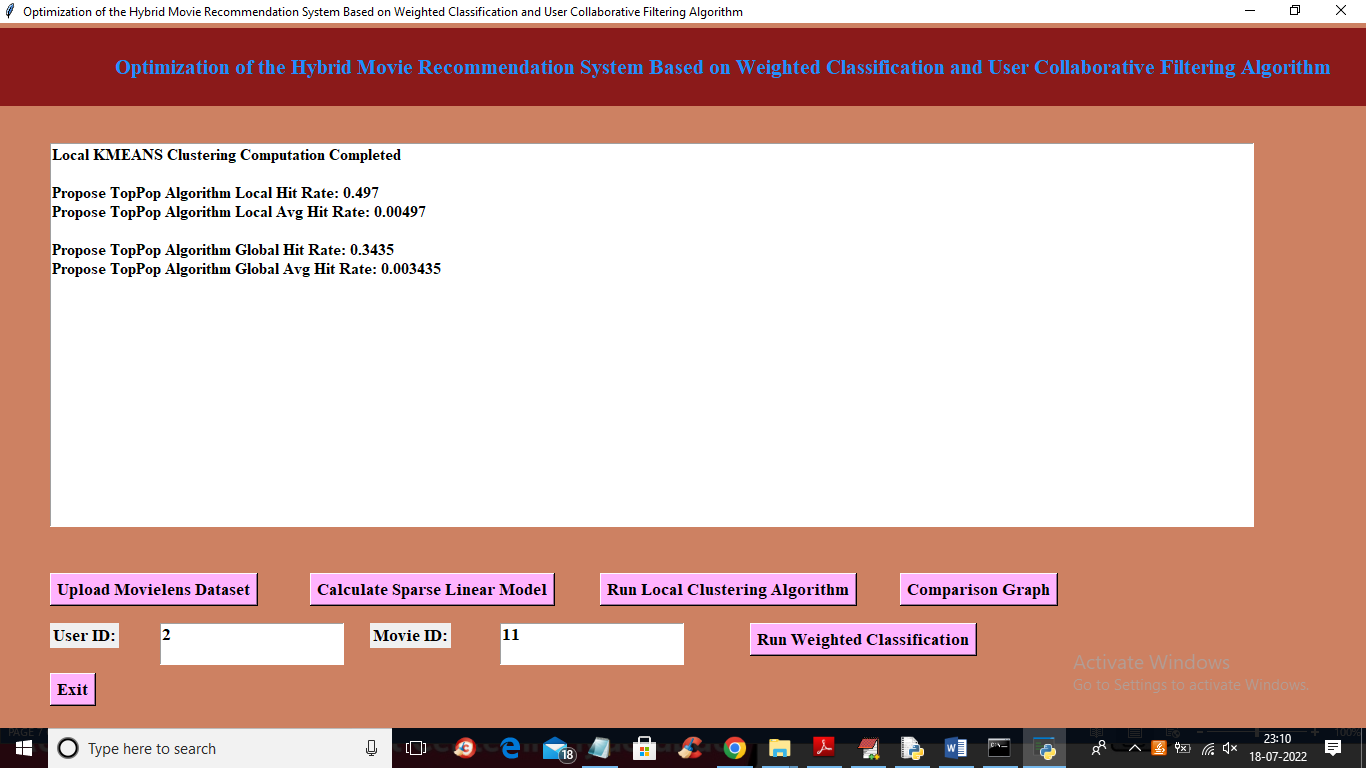
In above screen sparse in square bracket we can see sparse matrix features and then we can see dataset size and then train and test data split details and now dataset is ready and now click on ‘Run Local Clustering Algorithm’ to build clusters and get below output



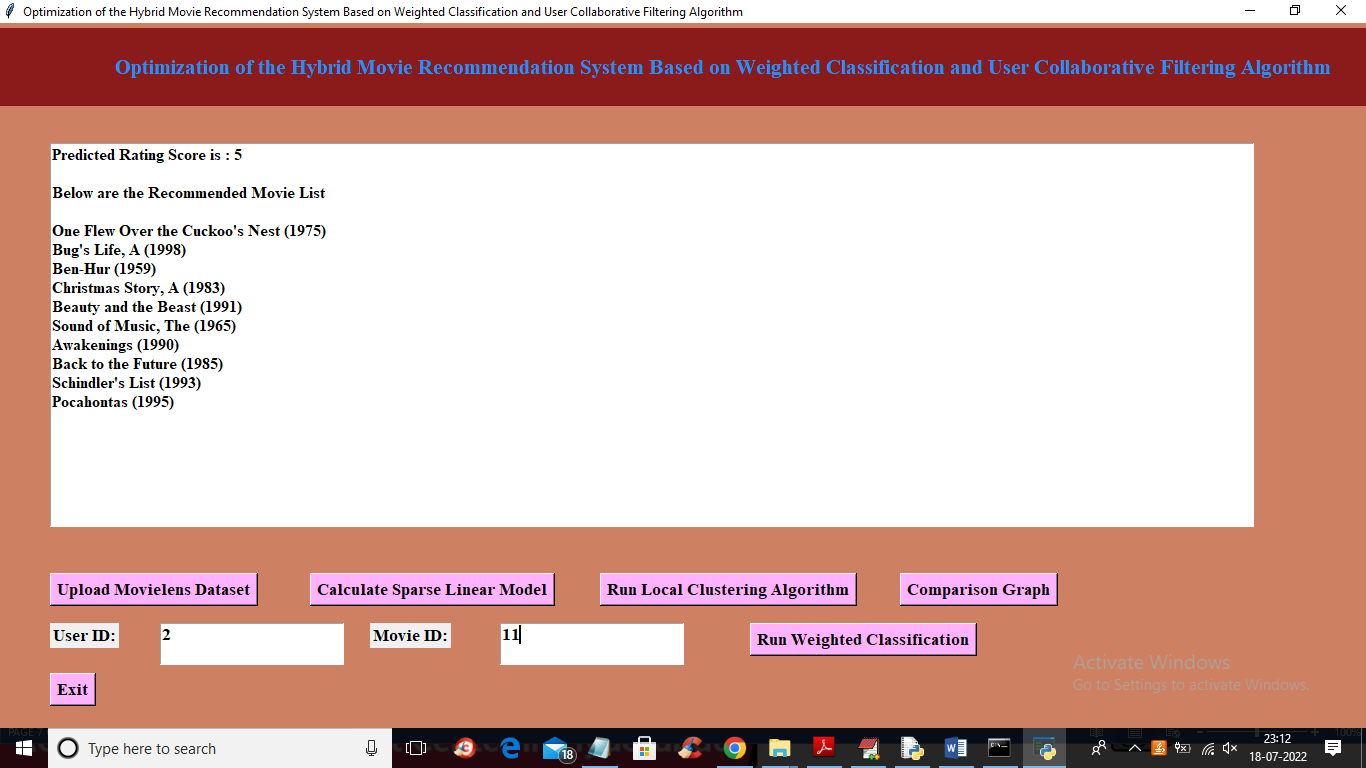
In above screen clustering task completed and we can see local and global HIT Rate and now click on ‘Comparison Graph’ button to plot local and global comparison graph



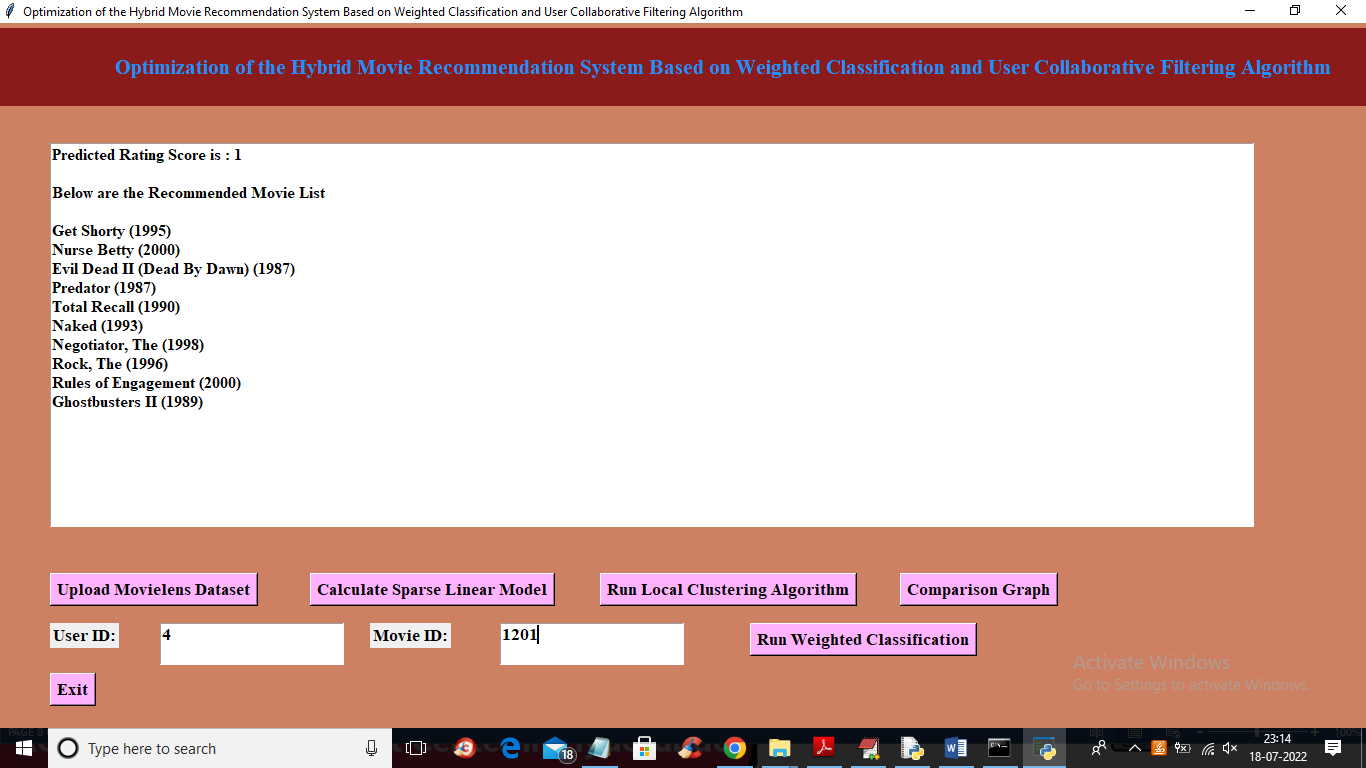
In above screen x-axis contains local and global Hit Rate and y-axis contains values and now enter user id and movie and then click on ‘Run Weighted Classification’ button to predict move recommendation



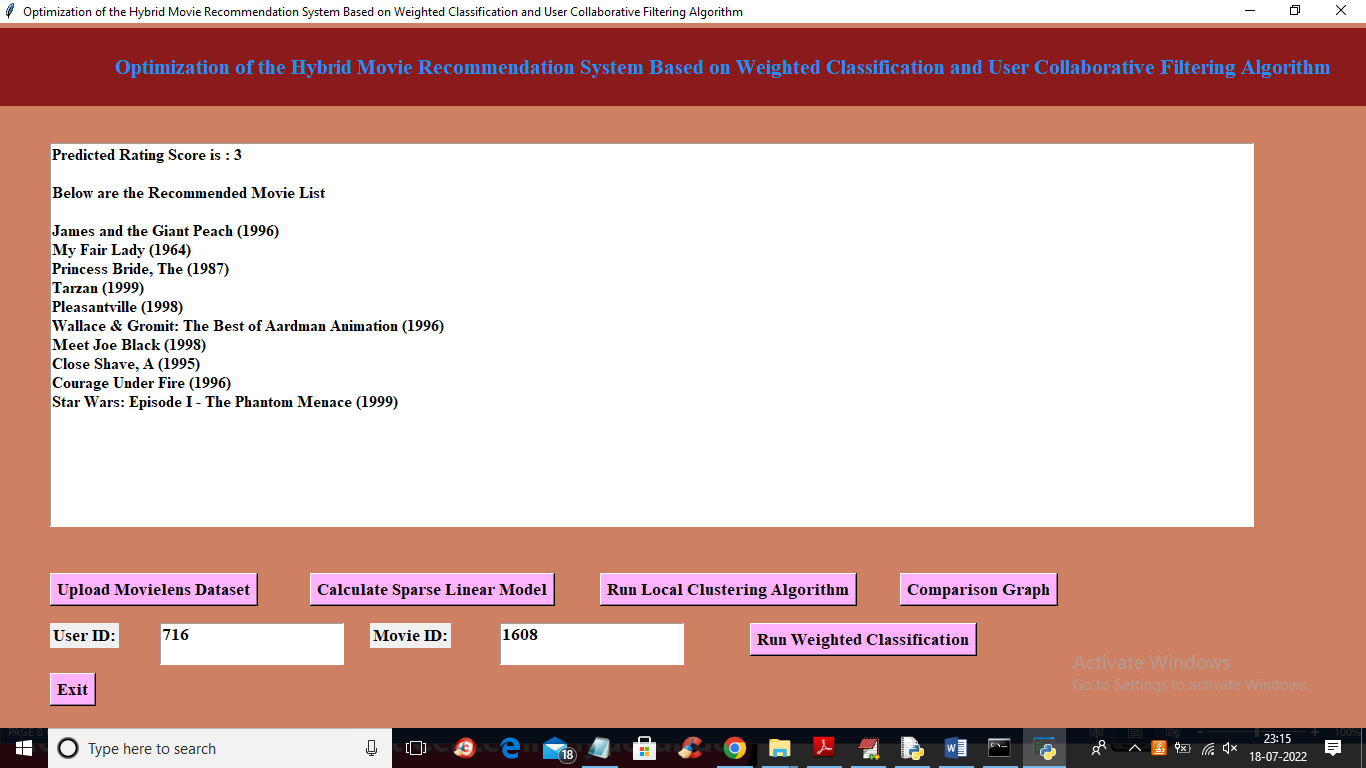
In above screen I entered user ID as 2 and Movie ID as 11 and then click on ‘Run Weighted Classification’ button to get below recommendation list

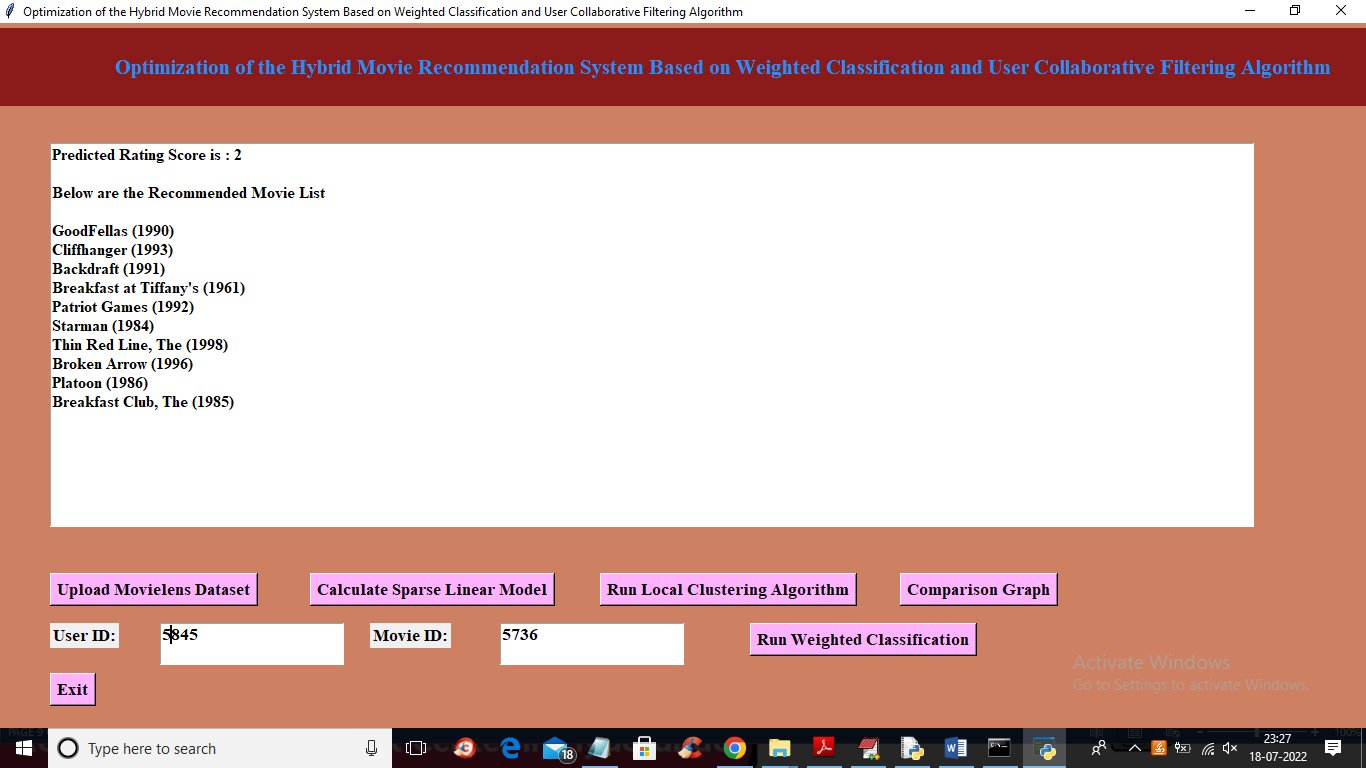


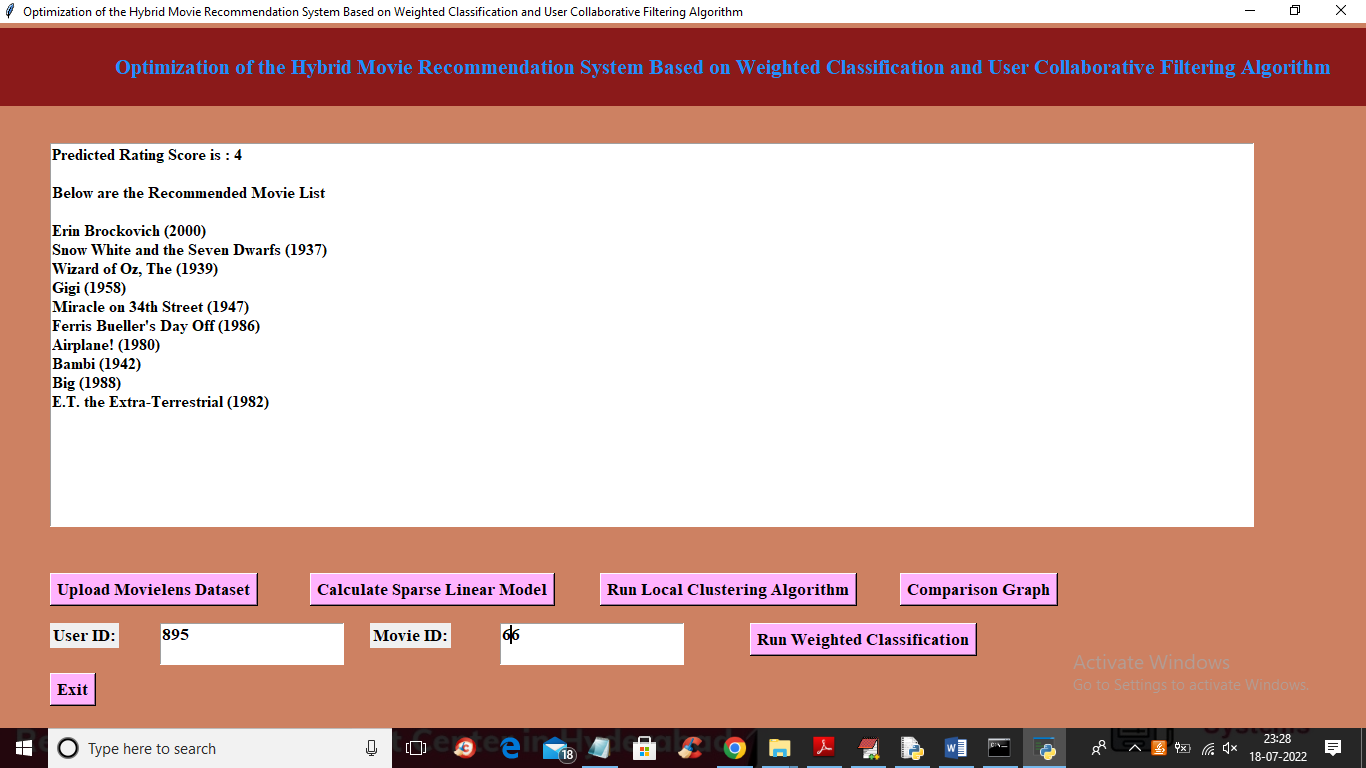
In above screen for given user id and movie ID we got predicted rating as 5 and then based on predicted rating score and user behaviour displaying top 10 recommended movies. Similarly enter any user id and movie id to get recommendation list



In above screen for 4 and 1201 we got above output







Similarly enter user and movie id and get recommendation list

**8. CONCLUSIONS:**

With the rapid development of Internet technology, the amount of information is growing at an explosive speed. Users are usually helpless in the face of how to obtain effective information more efficiently. It is difficult for them to find the information they are interested in simply and quickly. The birth of the personalized recommendation system provides users with a passive way to obtain information and makes up for the lack of search engine which can only provide the same information, which can provide personalized information for users. At present, personalized recommendation system has been widely used in video websites, music websites, e-commerce, news reading websites, and other fields and has attracted more and more attention from scholars and industry. This paper proposes a hybrid movie recommendation system optimization based on weighted classification and user collaborative filtering algorithm. The research focus of the algorithm is to consider the user’s behavior information and item category preference information at the same time. Firstly, the user’s web log is obtained. At the same time, according to the access time of the item obtained in the web log, the user’s recent behaviour information is obtained. The recent behaviour information reflects the user’s current interest. The behaviour information is transformed into user’s score of the item, and the score matrix is filled with the transformed score. The sparsity of the filled score matrix is reduced to a certain extent compared with the previous one. Secondly, according to the item category preference, the scoring matrix is transformed into a low-dimensional and dense item category preference matrix to obtain multiple clustering centers. The distance between the target user and each clustering center is calculated, and the target user is classified into the nearest clustering. Finally, the collaborative filtering algorithm is used to predict the score of the target user’s unsatisfied items and form a recommendation list. The innovation of the first mock exam is that the traditional recommendation system cannot capture user preferences accurately. A hybrid movie recommendation system and optimization method based on weighted classification and user collaborative filtering algorithm are proposed. The sparse linear model is used as the basic recommendation model, and the local recommendation model is trained based on user clustering. Finally, the top-N personalized recommendation of movies is realized by fusing with the weighted classification model. In this paper, user behaviour information is used to fill the scoring matrix, which alleviates the data sparsity to a certain extent. By clustering items by item category preference, the high-dimensional rating matrix is transformed into a low-dimensional item category preference matrix, which further reduces the sparsity of data, and finally improves the recommendation accuracy of the recommendation algorithm. Based on the analysis of functional requirements, the overall framework of the system and the design of each functional module are completed, and the basic functions such as personalized recommendation, popular movie recommendation, evaluation movie, and input movie are realized, and the above functions are displayed through the page.

**9. REFERENCES:**

[1] R. Shukla, S. Silakari, and P. K Chande, “An improved recommendation approach based on user’s standard and interests,” International Journal of Computer Applications, vol. 119, no. 1, pp. 6–14, 2015.

[2] R. Bhuvanya, K. Vijayalakshmi, and S. Uma, “Movie recommender system using two way filtering and agglomerative hierarchical clustering,” Journal of Computational and Theoretical Nanoscience, vol. 15, no. 6, pp. 2269–2272, 2018.

[3] Y. Yun, D. Hooshyar, J. Jo, and H. Lim, “Developing a hybrid collaborative filtering recommendation system with opinion mining on purchase review,” Journal of Information Science, vol. 44, no. 3, pp. 331–344, 2018.

[4] H. Alhoori and R. Furuta, “Recommendation of scholarly venues based on dynamic user interests,” Journal of Informetrics, vol. 11, no. 2, pp. 553–563, 2017.

[5] R. Chen, Q. Hua, Y.-S. Chang, B. Wang, L. Zhang, and X. Kong, “A survey of collaborative filtering-based recommender systems: from traditional methods to hybrid methods based on social networks,” IEEE Access, vol. 6, pp. 64301–64320, 2018.

[6] F. Pemisindo and N. Legowo, “Hybrid recommendation system with clustering and classification method based on case of clinic XYZ,” International Journal of Advanced Trends in Computer Science and Engineering, vol. 9, no. 2, pp. 1352–1357, 2020.

[7] M. Li, C. Wang, and Y. Sun, “A dual-clustering based collaborative filtering recommendation algorithm with confidence estimation of user,” Journal of Computational Information Systems, vol. 11, no. 12, pp. 4245–4254, 2015.

[8] N. Tohidi and C. Dadkhah, “Improving the performance of video collaborative filtering recommender systems using optimization algorithm,” The International Journal of Nonlinear Analysis and Applications (IJNAA), vol. 11, no. 1, pp. 283–295, 2020.

[9] J. Shi, Y. Tang, and P. Xu, “A collaborative filtering recommendation method based on clustering and user trust,” Journal of Computational Information Systems, vol. 11, no. 18, pp. 6845–6852, 2015.

[10] W. Froelich and P. Hajek, “Combining rough set-based relevance and redundancy for the ranking and selection of nominal features,” Procedia Computer Science, vol. 176, pp. 1459–1468, 2020.

[11] J.-H. Su, B.-W. Wang, C.-Y. Hsiao, and V. S. Tseng, “Personalized rough-set-based recommendation by integrating multiple contents and collaborative information,” Information Sciences, vol. 180, no. 1, pp. 113–131, 2010.

[12] J. Bobadilla, F. Ortega, A. Hernando, and J. Bernal, “A collaborative filtering approach to mitigate the new user cold start problem,” Knowledge-Based Systems, vol. 26, pp. 225–238, 2012.

[13] B. S. Neysiani, N. Soltani, N. Soltani, R. Mofidi, and M. H. Nadimi-Shahraki, “Improve performance of association rule-based collaborative filtering recommendation systems using genetic algorithm,” International Journal of Information Technology and Computer Science, vol. 11, no. 2, pp. 48–55, 2019.

[14] I. Kim and V. Gupta, “Using social tags and user rating patterns for collaborative filtering,” International Journal of Information Systems and Social Change, vol. 8, no. 2, pp. 25–49, 2017.

[15] N. Yin, “A big data analysis method based on modified collaborative filtering recommendation algorithms,” Open Physics, vol. 17, no. 1, pp. 966–974, 2019.