# Recommendation System Based On Content Rating

Project report submitted in partial fulfillment of the requirement for

the degree of Bachelor of Technology

in

**Computer Science Engineering / Information Technology**

By

TANISHQ GUPTA(191251)

**UNDER THE SUPERVISION OF**

Dr. Pardeep Garg

&

**CO-SUPERVISION OF**

Dr. Yugal Kumar

to

Department of Computer Science & Engineering and Information Technology

**Jaypee University of Information Technology Waknaghat, Solan-173234, Himachal Pradesh**

# Candidate's Declaration

I hereby declare that the work presented in this report entitled **"Recommendation System Based On Content Rating"** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science Engineering / Information Technology** submitted in the department of Computer Science & Engineering**,** Jaypee University of Information Technology Waknaghat is an authentic record of my own work carried out over a period from July 2022 to May 2023 under the supervision of **Dr. Pardeep Garg (Assistant Professor SG with ECE)** and Co-supervision of **Dr. Yugal Kumar(Associate Professor with CSE & IT)**. I also authenticate that I have carried out the above-mentioned project work under the proficiency stream **Data Science**.

The matter embodied in the report has not been submitted for the award of any other degree or diploma.

(............................)

Tanishq Gupta

191251

This is to certify that the above statement made by the candidate is true to the best of my knowledge.

(.............................)

Dr. Pardeep Garg

Assistant Professor SG

Electronics and Communication Engineering

Dated:

(.............................)

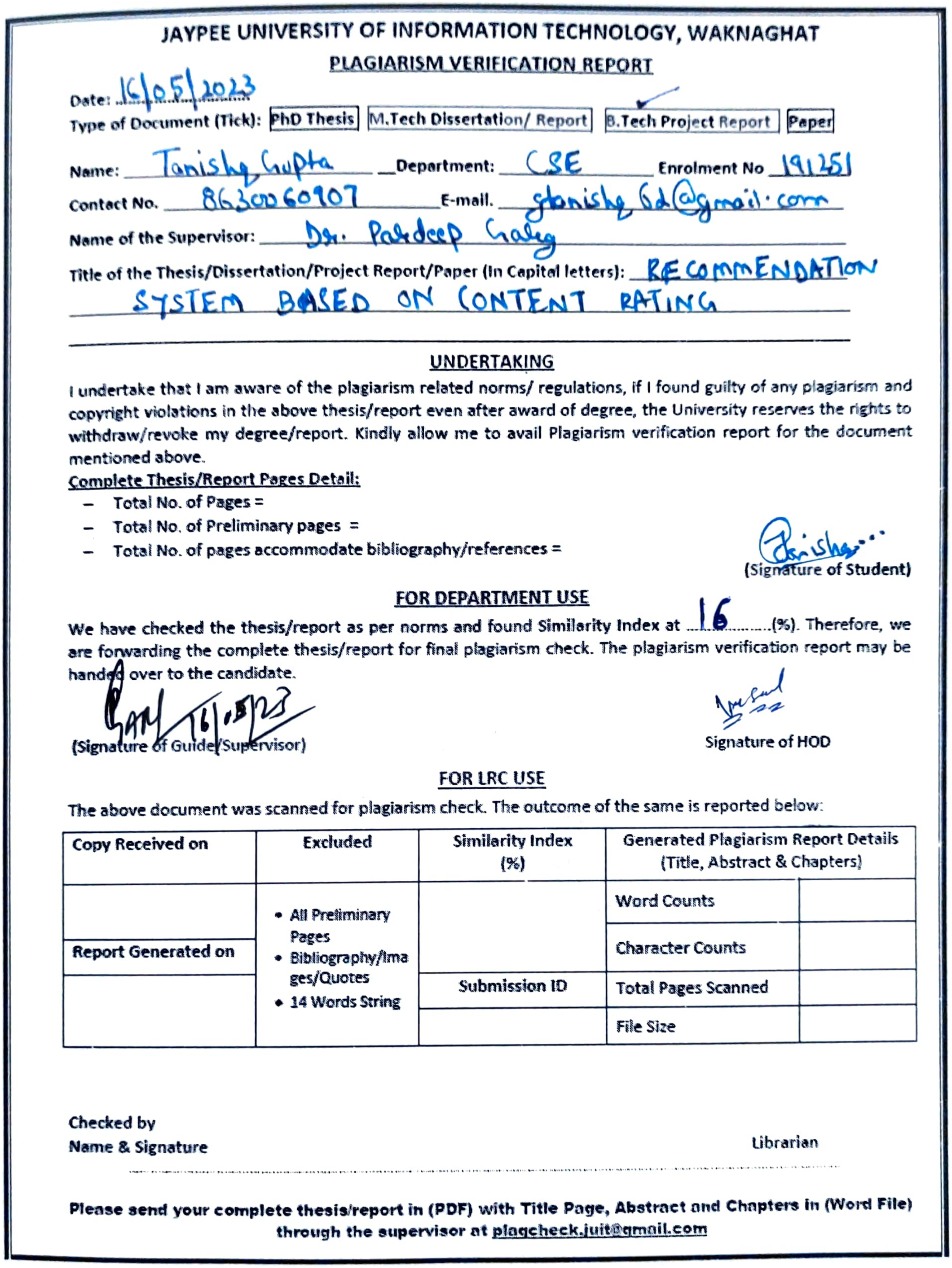
Dr. Yugal Kumar

Associate Professor

Computer Science and Engineering & Information Technology

Dated

# PLAGIARISM CERTIFICATE



**JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT**

**PLAGIARISM VERIFICATION REPORT**

Date: 16/05/2023

Type of Document (Tick): PhD Thesis | M.Tech Dissertation/ Report | [✓ B.Tech Project Report] | Paper

Name: Tanishq Gupta Department: CSE Enrolment No 191251

Contact No. 8630060907 E-mail. gtanishq6d@gmail.com

Name of the Supervisor: Dr. Pardeep Garg

Title of the Thesis/Dissertation/Project Report/Paper (In Capital letters): RECOMMENDATION SYSTEM BASED ON CONTENT RATING

**UNDERTAKING**

I undertake that I am aware of the plagiarism related norms/ regulations, if I found guilty of any plagiarism and copyright violations in the above thesis/report even after award of degree, the University reserves the rights to withdraw/revoke my degree/report. Kindly allow me to avail Plagiarism verification report for the document mentioned above.

**Complete Thesis/Report Pages Detail:**

- Total No. of Pages =
- Total No. of Preliminary pages =
- Total No. of pages accommodate bibliography/references =

[illegible]
(Signature of Student)

**FOR DEPARTMENT USE**

We have checked the thesis/report as per norms and found Similarity Index at 16 (%). Therefore, we are forwarding the complete thesis/report for final plagiarism check. The plagiarism verification report may be handed over to the candidate.

[illegible] 16/05/23
(Signature of Guide/Supervisor)

[illegible]
Signature of HOD

**FOR LRC USE**

The above document was scanned for plagiarism check. The outcome of the same is reported below:

| Copy Received on | Excluded | Similarity Index (%) | Generated Plagiarism Report Details (Title, Abstract & Chapters) | |
|---|---|---|---|---|
| | • All Preliminary Pages<br>• Bibliography/Images/Quotes<br>• 14 Words String | | Word Counts | |
| Report Generated on | | | Character Counts | |
| | | Submission ID | Total Pages Scanned | |
| | | | File Size | |

Checked by
Name & Signature

Librarian

**Please send your complete thesis/report in (PDF) with Title Page, Abstract and Chapters in (Word File) through the supervisor at plagcheck.juit@gmail.com**

# ACKNOWLEDGEMENT

Firstly, I express my heartiest thanks and gratefulness to almighty God for His divine blessing makes it possible for us to complete the project work successfully.

I am grateful and wish our profound indebtedness to Supervisor **Dr. Pardeep Garg,** Department of Electronics & Communication, and Co-Supervisor **Dr. Yugal Kumar,** Department of CSE & IT, Jaypee University of Information Technology, Waknaghat. Deep Knowledge & keen interest of our supervisor in the field of "**Recommendation System Based On Content Rating**" to carry out this project. His endless patience, scholarly guidance, continual encouragement, constant and energetic supervision, constructive criticism, valuable advice, reading many inferior drafts and correcting them at all stages have made it possible to complete this project.

I would like to express our heartiest gratitude to **Dr. Pardeep Garg,** Department of ECE, and to **Dr. Yugal Kumar,** Department of CSE for his kind help to finish my project.I would also generously welcome each one of those individuals who have helped me straightforwardly or in a roundabout way in making this project a win. In this unique situation, I might want to thank the various staff individuals, both educating and non-instructing, which have developed their convenient help and facilitated my undertaking.Finally, I must acknowledge with due respect the constant support and patience of our parents.

**Tanishq Gupta** (191251)

# TABLE OF CONTENT

# LIST OF FIGURES

# ABSTRACT

The advent of streaming services has made it easier to watch films and television series. The movie industry has been growing rapidly over time. The enormous amount of content available nowadays makes it challenging for users to choose what to watch. Movie recommendation systems have been developed to assist customers in selecting films based on their individual preferences. This facilitates and amuses the choosing process. These systems employ a number of strategies to offer customers personalized suggestions. One of the most popular techniques is collaborative filtering, which suggests films that users may also like based on their tastes and watching history. Another method is content-based filtering, which utilizes the traits of movies—like genre, stars, and directors—to suggest others with comparable qualities. To provide suggestions that are more accurate, hybrid methods that integrate the two methodologies have also been created. It emphasizes how crucial personalisation is to recommendation systems since it raises user engagement and pleasure. The performance of movie recommendation systems may be increased by adding user input as well as cutting-edge methods like deep learning and natural language processing.

# Chapter-1 INTRODUCTION

## 1.1 Introduction :

In the era of online streaming services, recommendation systems for films are becoming more and more common. Finding a movie to watch that suits your tastes and mood can be difficult with so many movies accessible on services like Netflix, Hulu, and Amazon Prime Video. That's where movie recommendation systems come in; they analyze your viewing history using sophisticated algorithms and offer tailored suggestions for new films and TV episodes that you're likely to appreciate.

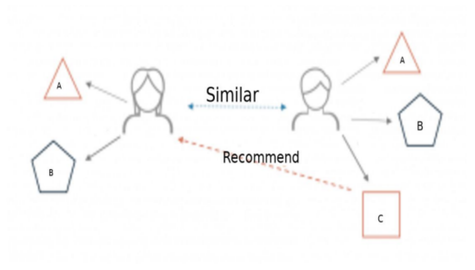


**Fig 1.1: Recommender System**

Several different filtering methods, such as collaborative filtering, content-based filtering, and demographic filtering, are at the core of a movie recommendation system. In order to provide recommendations based on similar users' watching histories, collaborative filtering examines their behavior and interests. Collaborative filtering, for instance, will suggest further films that individuals with similar viewing preferences have also loved if you've enjoyed a certain movie. On the other side, content-based filtering examines the qualities of the films you've loved, such as the genre, narrative, director, actors, and keywords, to suggest comparable films that adhere to your tastes. Finally, demographic filtering uses data about your location, gender, and age to suggest films that are well-liked by individuals who share those qualities.

Combining these filtering methods enables movie recommendation systems to offer highly customized recommendations that consider your unique viewing history, preferences, and demographics. For example, if you've enjoyed watching action movies with Tom Cruise as the lead actor, a movie recommendation system might recommend other action movies with Tom Cruise or other similar action stars. Alternatively, if you're in the mood for a comedy movie, a recommendation system might suggest a new comedy that has similar characteristics to the ones you've enjoyed in the past. Movie recommendation systems can also use machine learning algorithms to continuously improve the accuracy of their recommendations. As you watch more movies and provide feedback on the recommendations, the system can adjust its algorithms to better understand your preferences and provide more accurate and personalized recommendations.

To sum up, movie recommendation algorithms are a useful resource for anybody looking to find new films and TV episodes that suit their tastes. These systems may offer highly customized suggestions based on your watching history, tastes, and demographics thanks to sophisticated filtering techniques and machine learning algorithms. A movie suggestion system may assist you in finding the ideal movie to watch, regardless of whether you're in the mood for an action film, a comedy, or something completely different.

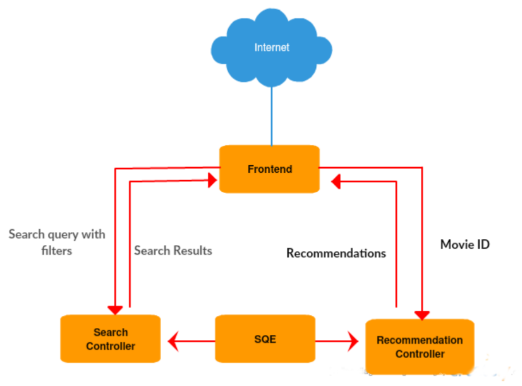


**Fig 1.2: Flow Diagram for a Recommender System**

## 1.2 Problem Statement :

Users may have trouble locating films that suit their tastes due to the wide selection of films offered on various platforms. Personalized suggestions and consideration of the users' unique interests and preferences may not be features of traditional movie recommendation systems. Traditional movie recommendation systems have the drawback of frequently using collaborative filtering, which suggests films based on the actions and tastes of other users. This method may not correctly reflect the user's unique tastes and may produce recommendations that are irrelevant or inadequate. Additionally, these systems might not consider the user's individual preferences and interests, and they might suggest films based on their popularity or ratings rather than their actual content.

A content-based movie recommendation system may be created to overcome these problems by using the characteristics of the movies to suggest related films to the viewer. In order to determine the characteristics of the films that the user has loved, the system may examine their watching history and preferences. Based on these characteristics, it can then suggest comparable films to the user.

This project's main goal is to plan and create an accurate and useful content-based movie recommendation system that can offer the user-specific suggestions. The software needs to be able to examine the movie's content and determine which of its elements and qualities corresponds most closely to the user's tastes. Additionally, it should be able to adjust to the user's evolving preferences and enhance the precision of the recommendations over time. The system should be simple to use, offer relevant and interesting movie suggestions that improve the user's viewing experience.

When creating a referral system from scratch, I encounter a number of unique difficulties. What should I do when the website does not receive enough users given the prevalence of recommender systems based on user information today? The movie representation is then resolved, allowing the system to comprehend movies. For two films to be compared for resemblance, this is a need. A way to classify films is by looking at its genre, actor, and director. But for each element of the film there should be a different weight for them and each of them play a different role in the recommendation. So I get these questions:

• How to recommend movies when no user information is available.

• What movie features can be used for a recommender system.

• How to calculate the similarity between two movies.

## 1.3 Objectives :

A content-based movie recommendation system's goal is to give customers individualized and accurate movie suggestions based on their watching habits and personal interests. In order to provide a list of suggested films that the user is likely to love, this system examines the movie content and compares it to the user's favorite genres, stars, directors, and other attributes.
The content-based movie recommender system aims to deliver a personalized, user-friendly experience that boosts user engagement and enhances the possibility that users will return to the site. This approach can boost user happiness and boost user retention rates by suggesting films that are in line with the user's likes and preferences. The approach can also benefit the movie business by aiding in the promotion of obscure films that would not have otherwise attracted as much notice.

The content-based movie recommender system's particular goals are as follows:

- Examining a movie's material to find key elements that may be utilized to produce suggestions.
- Creating a list of suggested movies by comparing the movie features to the user's preferences.
- Giving viewers individualized suggestions based on their unique watching interests and histories.
- Offering a satisfying and individualized experience will increase user engagement and retention rates.
- Assisting in the promotion of obscure films that would not have gotten as much notice otherwise.
- By making precise and pertinent movie suggestions, you may raise consumer happiness.

## 1.4 Methodology :

The methodology for a content-based movie recommender system involves several steps, including data collection, data preprocessing, feature extraction, similarity calculation, and recommendation generation.
First, data collection involves gathering information about movies, such as their titles, genres, actors, directors, and plot summaries, from various sources such as movie databases, streaming platforms, or user ratings websites. Next, the collected data needs to be preprocessed to ensure consistency and remove any irrelevant or duplicate information. This may include removing stop words, stemming, and normalizing the text.

Feature extraction is a critical step in content-based recommendation systems. In the context of a movie recommender system, it involves identifying the most important features of the movie, such as its genre, cast, director, and plot summary. This step typically involves applying natural language processing techniques such as part-of-speech tagging and named entity recognition to identify and extract the relevant features from the movie's textual data.

The similarity between films may be determined after the pertinent features have been retrieved. Mathematical methods like cosine similarity or Jaccard similarity are frequently used for this. These techniques compare the characteristics of two films to determine how similar they are.

The system may also provide a list of suggested films depending on the user's viewing habits and history. In order to choose the most relevant movies as suggestions, the user's selected features are compared to those of the movies in the database. The system may also include user comments and ratings to enhance the suggestions' accuracy and relevancy.

Overall, the methodology for a content-based movie recommender system involves data collection, preprocessing, feature extraction, similarity calculation, and recommendation generation. By following these steps and incorporating user feedback, the system can provide personalized and accurate recommendations that align with the user's interests and preferences.

**WorkFlow Diagram :**

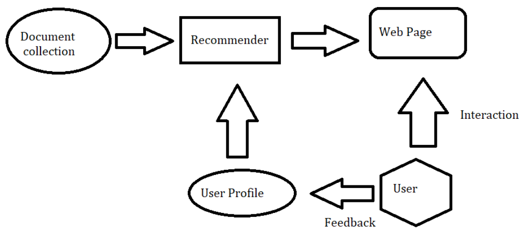


**Fig 1.3: WorkFlow**

**Agile Methodology :**

1) **Collecting the data sets:** Collecting all the required dataset from the kaggle website. In this project I will be using 2 datasets a) tmdb_5000_movies.csv b) tmbd_5000_credits.csv.

2) **Data Analysis:** make sure that the collected datasets are correct and analyze the data in csv files. i.e checking whether all the column fields are present in the datasets.

3) **Algorithms:** in our project I will be focusing on **Content Based Filtering, K-means clustering, RMSE, SVD, Cosine Similarity.**

4) **Training and Testing the model:** once the implementation of the algorithm is completed, I have to train the model to get the result. I have tested it several times and the model is recommending a different set of movies to different users.

5) **Improvements in the project:** In the later stage I can implement different algorithms and methods for better recommendations.

## 1.5 Organization :

A content-based movie recommendation system can be broken down into numerous different parts, such as data processing, data storage, and user interface.

The first part, data storage, is keeping track of movie information in a database, including titles, genres, actors, directors, and story summaries. This database should be created with the ability to efficiently manage massive volumes of data and be simple to use for data processing and retrieval.

Data cleaning, feature extraction, similarity testing, and recommendation creation are just a few of the sub-components that make up the second component, data processing. Standardizing the data format and deleting any duplicate or extraneous information constitute data cleansing. Using natural language processing techniques, feature extraction includes locating and separating the most significant aspects of the videos. Calculating similarity between films entails comparing their features to determine how similar they are. A list of suggested films is created throughout the recommendation creation process using the user's preferences and viewing history.

The user interface, which makes up the third component, is the part of the system that users interact with. This element is in charge of providing the user with the suggestions in an understandable and user-friendly manner. Users should be able to enter their preferences, see the suggested films, and offer comments and ratings through the user interface in order to enhance the precision and relevance of the suggestions.

Establishing clear communication and interaction between these components is crucial to ensuring the content-based movie recommendation system operates as intended. For instance, the user interface should be able to present the system's suggestions, and the data processing component should be able to retrieve and process data from the database.

# **Chapter-2** LITERATURE SURVEY

There are many different ways and procedures that have been created to solve the difficulties of suggesting films based on their content, according to a literature review on the subject.

Utilizing natural language processing (NLP) methods to analyze the textual components of films and pinpoint significant elements like genres, actors, and directors is a common strategy. For instance, Poria et al. 's (2014) [1] study employed NLP to examine movie plot summaries and pull out details like the key characters, their actions and emotions, and the locales. The movie genres were predicted using these characteristics, which were then used to train a classifier to produce suggestions.

Another strategy is to organize movies based on their attributes using clustering techniques, then suggest movies from those clusters. As an illustration, a research by Bobadilla et al. (2013) [2]employed clustering algorithms to categorize films based on their genre, actors, and directors, and suggested films from the same cluster as the user's chosen films.

Another popular strategy in movie recommendation systems is collaborative filtering, which includes making movie recommendations based on the watching habits and interests of people who are similar to you. However, this method has drawbacks when there is a lack of data or when users have different preferences.

Hybrid approaches that combine content-based and collaborative filtering techniques have also been developed. For example, a study by Panniello et al. (2014) [3]combined a content-based approach that analyzed the textual content of movies with a collaborative filtering approach that recommended movies based on the ratings of similar users.

[4]The use of deep learning methods, such as convolutional neural networks (CNNs) and neural networks, in content-based movie recommendation systems has been the subject of several research. For instance, Sedhain et al. (2015) trained CNNs to predict the genres of future movies using feature representations from movie poster images.[5]Similar to this, Kim et al.'s (2016) research used a neural network to create feature representations from the audio and visual information of films, which were then used to predict movie ratings.

A number of studies have looked at these strategies as well as the usage of user-generated content and social media to increase the relevance and accuracy of movie suggestions. Shi et al. (2015)[6] used social media data, for instance, to identify the most well-liked and trending films, and then incorporated this knowledge into the recommendation engine.

Overall, the literature review demonstrates that several strategies and methods have been created to solve the difficulties associated with content-based movie recommendation systems. While each method has advantages and disadvantages of its own, hybrid methods that incorporate several strategies and outside data sources seem to hold the most promise for enhancing the precision and relevance of movie recommendations.

In the last three to five years, several research studies have been conducted on content-based movie recommendation systems. In this literature survey, I will review some of the most significant and recent contributions to this field.

Hu, Li, and Chen (2019) [7]carried out one of the most significant recent research in content-based recommendation systems and suggested a revolutionary deep learning-based method for movie recommendation. A convolutional neural network (CNN) was employed by the authors to extract characteristics from narrative summaries and movie posters. To increase suggestion accuracy, they also created a hybrid collaborative filtering model based on CNN. The outcomes demonstrated that the suggested strategy outperformed a number of currently used state-of-the-art techniques.

Another recent work by Hossain and Muhammad (2020) [8]suggested a linguistic and visual content-based movie selection system. The authors used natural language processing techniques to extract textual features, such as genre, plot summary, and actors' names, and then used a deep learning model to extract visual features from movie posters. The proposed model was evaluated on a large movie dataset, and the results showed that the hybrid model outperformed several baseline models.
In another study, Dhir and Sood (2020) proposed a content-based movie recommendation system that utilizes a multi-modal approach.
The authors employed a hybrid strategy based on deep learning and matrix factorization to provide suggestions after combining information from several sources, such as narrative summaries, movie posters, and user ratings. The outcomes demonstrated that the suggested strategy outperformed a number of currently used state-of-the-art techniques.
A recent work by Zhang and Xie (2020) [9]developed a graph embedding-based content-based movie recommendation system. The authors captured the connections between films using graph-based representations, and then generated suggestions using embedding methods. The suggested method was tested against a sizable movie dataset, and the findings revealed that it performed better than numerous other state-of-the-art techniques already in use.

Chen and Cai (2021) [10]suggested a content-based recommendation system that makes use of both textual and visual characteristics in a related research. To represent the connections between movies, the authors used a graph-based model. To create suggestions, they combined deep learning with graph embedding. The suggested method was tested against a sizable movie dataset, and the findings revealed that it performed better than numerous other state-of-the-art techniques already in use.Sun et al. 's (2021) [11]proposal for a content-based recommendation system that makes use of movie subtitles was made in another recent research. The authors extracted information from the subtitles using natural language processing techniques, and then they utilized a deep learning model to provide suggestions. The proposed method was evaluated on a large movie dataset, and the results showed that the method outperformed several existing state-of-the-art methods.In summary, recent research in content-based movie recommendation systems has focused on developing novel approaches that utilize deep learning, graph embedding, and multi-modal techniques to improve recommendation accuracy. These approaches have shown promising results and outperformed several existing state-of-the-art methods. Additionally, the use of visual features, textual features, and user feedback has been widely explored to improve the accuracy and relevance of movie recommendations.

# **Chapter-3** SYSTEM DEVELOPMENT

Before proceeding towards the System Development of the project lets first gain an understanding about what actually a Recommendation System is and its types :

## 3.1 A recommender system:

Is a type of software tool that is used to suggest items to users based on their past behavior or preferences. It is a subset of artificial intelligence and is commonly used in e-commerce, social media, and streaming platforms. The system is designed to predict what a user might like based on their interactions with the system and then recommend relevant items to the user.

Users' data is gathered and analyzed by recommender systems to look for trends that may be utilized to generate predictions. For instance, an e-commerce website could gather information on the goods a user has previously browsed or purchased. The system can suggest comparable goods that the user might be interested in based on this data. Similar to this, a streaming service may compile information on the films or TV series a user has seen in order to suggest more works with related topics or genres.

There are several types of recommender systems, each with its unique approach to providing recommendations. Here are the most common types of recommender systems:

- **Content-based recommender systems:** Systems that provide product recommendations based on the characteristics of the objects themselves are known as content-based recommender systems. A content-based recommender system could suggest other action films to a user who has expressed interest in them based on their shared traits, such as genre, actors, directors, and narrative.

- **Collaborative filtering recommender systems:** Systems that propose products based on the preferences of other users who have similar tastes are known as collaborative filtering recommender systems. It examines user behavior data to look for trends and connections between people and things. For instance, if two users have similar movie interests, the algorithm would suggest films that one user has seen but the other hasn't.

- **Hybrid recommender systems:** These systems combine content-based and collaborative filtering strategies to provide suggestions that are more accurate. By taking item qualities into account, content-based filtering helps hybrid systems provide more accurate suggestions, while collaborative filtering takes user preferences and behavior into account.

- **Knowledge-based recommender systems:** Systems that provide recommendations based on explicit knowledge of user requirements and preferences are known as knowledge-based recommender systems. It employs a reasoning engine to generate suggestions and a knowledge base to represent information about people and products. A knowledge-based recommender system, for instance, can suggest a laptop based on the user's preferences, such as screen size, processing speed, battery life, etc.

- **Demographic-based recommender systems:** Systems that provide product recommendations based on demographic data, such as age, gender, and geography, are known as "demographic-based recommender systems." It divides users into categories using demographic information, then suggests products that are well-liked in those segments.

In conclusion, a recommender system is a potent tool that may assist companies in offering clients personalized experiences. It is predicated on gathering information about user behavior and using that information to anticipate what those users would find appealing. Although recommender systems have some drawbacks, they are a crucial component of contemporary e-commerce and social media and streaming platforms.

**Hardware and software Requirements :**

The following hardware and software were required for this project:

- 4 GB RAM
- MS Window 7 and above Software Requirements
- Jupyter Notebook
- Pycharm

**Concepts Requirements**

- Machine Learning Algorithms
- Data Preprocessing Functions and tools
- Scikit-learn
- NLTK
- Seaborn
- Streamlite
- Pickle
- Requests
- Knowledge of K-Means Clustering
- NumPy is a Python Programming Language
- Panda bears
- matplotlib

## 3.2 Anaconda Distribution:

The Open-Source Anaconda Programming language distribution has a comprehensive package manager and environment management system that are tailored for machine learning. With more than 1,500 packages, including well-known ones like NumPy, SciPy, Pandas, Matplotlib, and Scikit-learn, Anaconda provides a comprehensive collection of tools for scientific computing and data analysis. Many of the resources that Anaconda provides are available to user like for creating and maintaining virtual environments

The Anaconda distribution's capacity to handle package dependencies makes it simpler to install and update packages without worrying about compatibility concerns, one of its main advantages. Additionally, the distribution comes with the Conda package manager, which enables users to build and maintain virtual environments with certain package versions and dependencies, simplifying experiment replication and sharing code with others.

Anaconda has various features that are very useful like Anaconda's user-friendliness, strong package management system, and compatibility with other operating systems including Windows, macOS, and Linux, it is widely used in scientific computing, data analysis, and machine learning. Both a free and a premium edition of the Anaconda distribution are offered, with the latter including extra features and support.

Anaconda is a powerful and adaptable tool for scientific computing and data analysis because it offers users a variety of packages, tools, and environments for creating and deploying data-intensive applications.

## 3.3 Python Libraries:

Python is a versatile programming language that is widely used in data science, machine learning, web development, and scientific computing. Python libraries are collections of pre-written code that provide functionality to Python developers, allowing them to work more efficiently and effectively. Here are some of the most popular Python libraries and their applications:

1. **Numpy:** For all kinds of mathematics calculations we use the python programme known as Numpy. In NumPy, a variety of mathematical operations are accessible for subjects like Fourier analysis and linear algebra. Additionally supported are matrices and multidimensional arrays.
2. **Pandas :** Tools that are used for python data analysis and manipulation are called Pandas. Pandas provides tools for reading and presenting data in a variety of formats, including CSV, Excel, and SQL databases. It also has the capacity to collect, delete, and filter data.
3. **Matplotlib :** a collection of Python data visualization tools. For the aim of creating graphs, charts, and other sorts of data visualizations, Matplotlib offers a variety of functions.

4 **Scikit-learn :** is a Python package for machine learning. For classification, regression, clustering, and other machine learning applications, Scikit-learn offers a number of methods. Additionally, it has tools for feature extraction, data preparation, and model selection.

5. **TensorFlow :** is a Python toolkit for creating and refining neural networks. For the development of deep learning models for a variety of applications, such as audio and picture recognition, natural language processing, and others, TensorFlow offers tools.

6 **Flask :** A Python library for building web applications is Flask. Flask provides a simple and condensed framework for building web applications in Python, with support for routing, templates, and session management.

7 **Django :** is a Python library used to build web applications. Django, which supports user authentication, database integration, and other capabilities, facilitates Python online application development.

8 **Pickle :**Using a Python utility called Pickle, users may serialize and deserialize Python objects, turning them into byte streams that can be saved in files or sent over networks. Users may easily save and retrieve complicated data structures like lists, dictionaries, and objects with Pickle. Pickle is an effective serialization technique.

Pickle transforms a Python object into a byte stream that may be stored on a disc or sent over a network. The deserialization capabilities of the pickle module may then be used to transform the byte stream back into a Python object. Pickle is a helpful tool because it provides serialized data compression, a characteristic that is important for storing and sending enormous amounts of data.When using pickle, it's vital to keep in mind that Python objects can only be serialized and deserialized. It cannot be used to serialize or deserialize Python-representable objects or objects written in other programming languages.The pickle module is a helpful resource for transferring and storing Python objects overall. It is a well-liked option for data serialization in Python because of its simplicity and usability.

9. **Requests:** The well-liked Python HTTP framework Requests makes handling HTTP requests and responses simple. It provides a simple HTTP API that may be used for a variety of tasks, such as site scraping, API testing, and interacting with online programmes.The Requests library supports all HTTP methods, including GET, POST, PUT, DELETE, and others. It also makes it possible to change request headers, contents, and arguments. It also allows authentication, proxies, and SSL verification.When utilizing the Requests library, responding to requests is straightforward. It can automatically handle common response formats like JSON and XML, and developers have access to the response content, status codes, and headers.

The library is well-documented and actively maintained, making it a reliable choice for Python developers who need to work with HTTP requests and responses. It is also compatible with a wide range of Python versions and platforms, including Windows, Mac, and Linux.

Overall, Python libraries provide developers with a range of powerful tools and functionalities for various applications, making Python a popular choice for many industries and domains.

## 3.4 Model Development Stages:

The development of a content-based movie recommender system involves several steps, including data collection, preprocessing, feature extraction, and model training.

**1. Data Collection:** The first step in building a content-based movie recommender system is to collect data on movies, including their metadata such as genre, director, actors, release year, and synopsis. This data can be obtained from various sources such as online movie databases or APIs.

**2. Preprocessing:** Once the data is collected, it must be preprocessed to clean and organize it for use in the recommender system. This may involve removing duplicates, filling in missing values, and standardizing data formats.

**3. Feature Extraction:** The next step is to extract relevant features from the movie data. This may include text features such as keywords and synopsis, as well as categorical features such as genre, director, and actors. Feature extraction techniques may vary depending on the specific requirements of the recommender system.

**4. Model Training:** With the data preprocessed and features extracted, the next step is to train a machine learning model to make recommendations. This may involve using techniques such as cosine similarity or k-nearest neighbors to identify movies with similar content characteristics.

**5. User Interface:** Once the model is trained, it needs to be integrated into a user interface that allows users to interact with the system. The user interface may include features such as search, filtering, and personalized recommendations based on user preferences.

**6. Testing and Evaluation:** Finally, the system should be tested and evaluated to ensure that it is providing accurate and relevant recommendations to users. This may involve conducting user testing or using metrics such as precision, recall, and F1 score to evaluate the performance of the recommender system.Overall, the development of a content-based movie recommender system requires careful attention to data collection, preprocessing, feature extraction, and model training, as well as the integration of a user interface and testing and evaluation to ensure its effectiveness.

## 3.5 Python Softwares Used :

### 1) Jupyter Notebook :

A web-based interactive computational environment called Jupyter Notebook enables users to create and share documents that incorporate real-time code, descriptive text, mathematical equations, visualizations, and other material. For prototyping, data exploration, visualization, and documentation, it is extensively used in data science, scientific computing, and education.

Python, R, Julia, and a host of additional programming languages are all supported by Jupyter Notebook. Users may study and engage with data interactively because of the fact that each notebook is made up of a collection of cells that can each be run separately. It is simple to construct complex and instructive papers that merge code and explanation since the cells may include code, text, markdown, equations, and other media.Jupyter Notebook's ability to show and visualize data inline, making it simple to explore and comprehend large data sets, is one of its primary characteristics. Several libraries, including Matplotlib, Plotly, and Seaborn, are available to users for the creation of graphs, tables, charts, and other visualizations.Users of Jupyter Notebook may construct unique user interfaces for viewing and modifying data thanks to the functionality for interactive widget creation. Sliders, dropdown menus, and other input widgets that alter the notebook's output in real-time may be made with these widgets.

The ability to create and share notebooks with others is another helpful feature of Jupyter Notebook. It is simple to work on projects and share insights with others by hosting notebooks on cloud platforms like GitHub, Google Colab, and Azure Notebooks or by saving them as files.In conclusion, Jupyter Notebook is an effective tool for exploring data, creating prototypes, and documenting results. It is a preferred option for data scientists, academics, and educators due to its support for several programming languages, interactive visualization, and sharing.

## 2) Pycharm :

An Integrated Development Environment (IDE) created especially for Python development is called PyCharm. JetBrains, a software business that creates a wide range of development tools for several programming languages, is the one who created it. The Python community uses PyCharm a lot because of its cutting-edge capabilities and user-friendly design.With features that make it a potent tool for Python programming, PyCharm supports a wide variety of Python versions, from Python 2.7 to Python 3.9. These capabilities, which aid programmers in writing clear, mistake-free code, include code completion, code highlighting, and error highlighting.

PyCharm's support for well-known Python frameworks like Django, Flask, and Pyramid is a crucial component. Debugging, code completion, and code navigation are just a few of the tools that PyCharm gives developers to work with these frameworks, making it simpler for developers to construct and manage complicated Python programmes.

Additionally, PyCharm supports version control tools like Git, Mercurial, and Subversion, making it simpler for programmers to manage code changes and collaborate on projects. It has functions that improve developer collaboration, such as branch visualization, code reviews, and merge conflict resolution.The user-friendly design of PyCharm makes it simple to use and navigate. A code editor, a project manager, a debugger, and a console are all included in its user interface. In a single, integrated environment, these tools provide developers everything they need to write, test, and debug Python programmes.In conclusion, PyCharm is a strong Python programming tool that offers users features like code highlighting, code completion, and support for well-known Python frameworks. Teams working on Python projects might consider using it since it supports version control systems and collaborative tools. It is a well-liked option among Python developers due to its user-friendly interface and potent capabilities.

## 3) Streamlite :

An open-source framework called Streamlite makes it simple for programmers to create and deploy interactive data science web apps. It is intended to make working with data simpler for users of all skill levels by streamlining the creation and distribution of data-driven applications.Streamlite's simplicity is one of its best qualities. With just a few lines of code, developers may construct interactive apps by writing straightforward Python scripts. Developers may concentrate on the information and logic of their programme since Streamlit handles the labor-intensive user interface creation. With less time needed for front-end development, data scientists, analysts, and developers may quickly prototype and launch apps.

NumPy, Pandas, Matplotlib, and scikit-learn are just a few of the many data science libraries that Streamlit supports, making it simple to include data analysis and visualization into your applications. Additionally, Streamlit makes it simple to create interactive elements for your application by providing a variety of pre-built components, including sliders, dropdowns, and buttons.The simplicity of deployment is another benefit of Streamlite. Applications developed using Streamlit may be quickly and simply deployed to your own infrastructure or cloud services like Heroku or AWS. This makes it simple to grow your apps and share them with others as necessary.

Due to its simplicity of use and capacity to quickly develop interactive data-driven applications, Streamlit has grown in favor within the data science community. It has been applied to a variety of tasks, from straightforward data analysis tools to intricate machine learning models.To sum up, Streamlite is a strong framework that makes it simple for developers, analysts, and data scientists to build and deploy interactive data science web applications. It is a popular option for developing apps that need data analysis and visualization because of its simplicity and ease of deployment. Streamlite is positioned to become a crucial tool for anybody working with data because of its expanding community and support for a variety of data science libraries.

## 3.6 Design Analysis for the Movie Recommendation System :

In order to guarantee that a content-based movie recommendation system is well-designed and user-friendly, design analysis must be performed on the system's architecture, data flow, and user interface.

A crucial component of design analysis is assessing the architecture of the system. This entails reviewing the system's general structure and evaluating the connections between and interactions between its many components. A well-designed system, for instance, should clearly distinguish between the problems related to the user interface, recommendation engine, and data storage. The system must also be built to be scalable so that it can accommodate growing user populations and data volumes. Evaluating the system's data flow is a crucial component of design analysis. This entails looking into the system's methods for ingesting, processing, and storing data. For instance, the system must be built to quickly and accurately provide suggestions by processing and storing data in an efficient manner.A crucial component of design analysis is user interface design. With simple and clear labeling and instructions, the user interface should be intuitive and simple to use. Users should receive pertinent suggestions from the system based on their preferences and prior encounters with it. Additionally, users should be able to comment on suggestions made by the system and hone their preferences over time.

Examining the system's performance in actual-world circumstances is another aspect of design study. By asking consumers to engage with the system and offer input on its usability and performance during user testing, this may be achieved. The system may be improved further to enhance its design and usefulness based on this input.

Overall, design analysis is crucial to ensuring that the content-based movie recommendation system is well-designed and user-friendly. The system may be made to deliver accurate and pertinent recommendations to users in a quick and simple manner by carefully analyzing its design, data flow, and user interface.

## 3.7 Main Algorithm to be used :

**1) Content-Based Recommender systems :**
are frequently employed in movie recommendation systems as they offer individualized suggestions based on the qualities of the films that consumers have demonstrated interest in. The ability to suggest films that are comparable in content and topic to films that a user has already appreciated is one of the main advantages of employing a content-based method for movie recommendations. For instance, if a user has shown a preference for action films, a content-based recommender may suggest other action films that share those preferences, even if they are not well-liked or have high ratings by other users.The ability to make recommendations for specialized or narrow interests is another advantage of content-based recommenders. For instance, if a user expresses interest in nature-themed documentaries, a content-based recommender may suggest other nature-themed films, even if they are not well-known or highly rated.

Another benefit of content-based recommenders is that they are simple for users to understand and comprehend. Content-based recommenders may easily analyze the content and attributes of films to produce recommendations that are simple to comprehend and explain, unlike other recommendation systems that may employ complicated algorithms or user behavior data.

Overall, content-based recommenders can be an effective approach for building movie recommendation systems that provide personalized and relevant recommendations to users based on the characteristics of the movies they have shown interest in.

**2) Cosine Similarity :**

The cosine of the angle between any two non-zero vectors in an inner product space is used to determine how similar the vectors are. The cosine similarity of two documents may be used to determine how similar they are in terms of their word content. When discussing text or document similarity, each document can be represented as a vector of word frequencies.In order to locate things that are similar to a particular item, content-based recommendation algorithms frequently employ the cosine similarity measure. For instance, in a recommendation system for movies, each film may be represented as a vector of characteristics, such as genre, actors, and directors, and recommendations can be made based on the cosine similarity between two films. Cosine similarity is preferred over other distance metrics, such as Euclidean distance, for text or document similarity because it is not affected by the length of the vectors and only measures the angle between them. This means that it can handle sparse high-dimensional data well and is less sensitive to differences in word frequency between documents.

$$\mathrm{Cos}\,\theta = \frac{\vec{a}\cdot\vec{b}}{\|\vec{a}\|\|\vec{b}\|} = \frac{\sum_1^n a_i b_i}{\sqrt{\sum_1^n a_i^2}\sqrt{\sum_1^n b_i^2}}$$

**Eq 3.1: Formulae for Cosine Similarity**

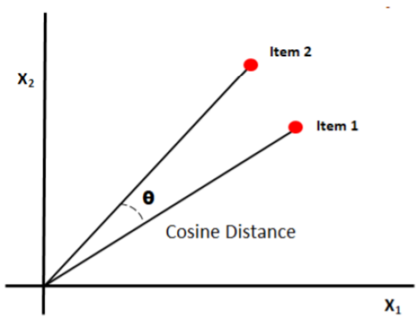


**Fig 3.1: Graph For Cosine Similarity**

**3) K-means Algo :**

Unsupervised machine learning employs the well-known clustering technique K-means. The primary objective of the K-means method is to divide a given dataset into K clusters, with each data point belonging to the cluster with the nearest center. The method iteratively changes the cluster centers until a convergence condition is met.The programme selects at random one of the dataset's initial K cluster centers. Each data point is then allocated the nearest cluster center depending on how far away it is from the cluster center by Euclidean distance. By calculating the mean of each cluster's data points, the new cluster center is established. The algorithm continues iterating between assigning data points to the nearest cluster and updating the cluster centers until a convergence criterion is met, such as a maximum number of iterations or a minimum change in the cluster centers.

The K-means algorithm is widely used in various fields, including image segmentation, customer segmentation, and data compression. However, the algorithm has some limitations, such as sensitivity to initial cluster centers and the assumption that the clusters have spherical shapes and equal sizes. Nonetheless, K-means remains a useful and popular tool for clustering large datasets.

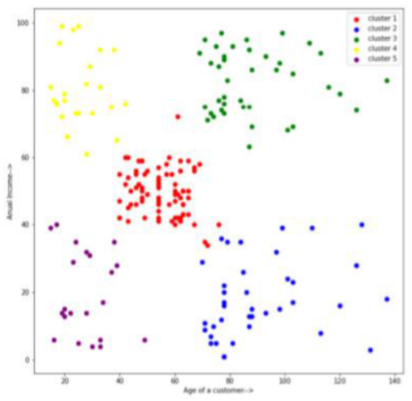


**Fig 3.2: K-means clustering**

**4) Singular Value Decomposition :**

A matrix is divided into three components using the Singular Value Decomposition (SVD) matrix factorization technique: a left singular matrix, a diagonal matrix, and a right singular matrix. In data analysis, machine learning, and signal processing, SVD is often employed.Matrix approximation, dimensionality reduction, and picture compression are just a few of the uses for SVD. In recommender systems in particular, SVD is frequently employed to find latent characteristics that affect user preferences.One of the key benefits of SVD is that it can handle missing values and noisy data. SVD can be used to impute missing values in a matrix, making it a powerful tool for data cleaning and preprocessing.

However, SVD can be computationally expensive, particularly for large matrices. There are also various modifications of SVD, such as truncated SVD and randomized SVD, which can be used to reduce computation time.

Overall, SVD is a powerful matrix factorization technique that has many applications in data analysis and machine learning. Its ability to handle missing values and noisy data makes it a valuable tool for data preprocessing, while its ability to identify latent factors makes it a useful technique for recommender systems and other data-driven applications.

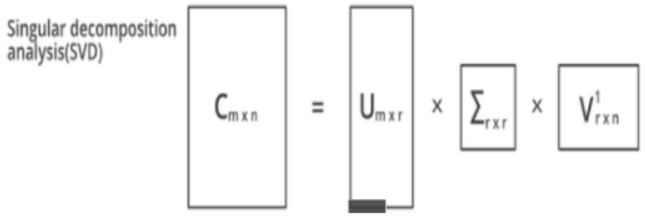

Singular decomposition analysis(SVD)
$C_{m \times n}$ = $U_{m \times r}$ x $\sum_{r \times r}$ x $V^{1}_{r \times n}$

**Fig 3.3: SVD**

**5) The RMSE(Root Mean Square Error) :**

RMSE stands for Root Mean Squared Error, which is a commonly used evaluation metric in regression analysis and machine learning. RMSE measures the difference between the predicted and actual values of a target variable in a regression problem. RMSE is calculated by taking the square root of the average of the squared differences between the predicted and actual values. The formula for RMSE is:

RMSE = sqrt(mean((predicted - actual)^2))

where predicted is the predicted value, actual is the actual value, and mean is the average of the squared differences between predicted and actual values.

RMSE is a useful metric because it gives an idea of how far off the predictions are from the actual values. It also penalizes larger errors more heavily than smaller errors, as it takes the square of the difference between predicted and actual values. In machine learning, RMSE is often used as a performance metric for regression models. The goal is to minimize the RMSE value, which indicates that the model is accurately predicting the target variable.

$$\mathbf{RMSD} = \sqrt{\frac{\sum_{i=1}^{N} \left(x_i - \hat{x}_i\right)^2}{N}}$$

**Eq 3.2: RMSD**

**3.8 DataSet Used :**

For training and testing the model I had used **TMDB 5000 Movie Dataset.**

The TMDB 5000 Movie Dataset is a dataset consisting of information on approximately 5,000 movies. It was compiled by Kaggle, a platform that provides data science and machine learning resources for researchers and practitioners. The dataset is available for free download, and it is often used for machine learning, data analysis, and recommendation system development.

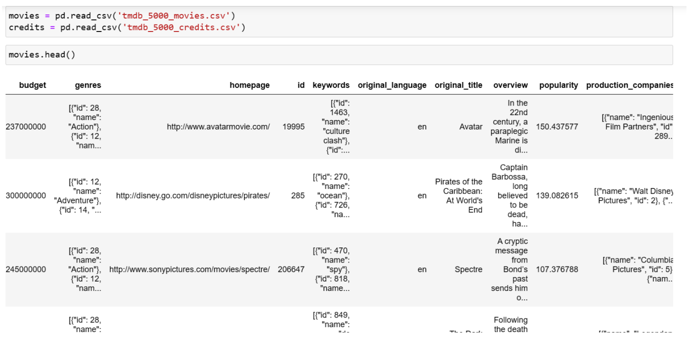
The dataset includes information on various attributes of the movies, such as the title, genre, release date, runtime, budget, revenue, and ratings. Additionally, the dataset contains information on the cast and crew of the movies, including the actors, directors, producers, and writers. The data is provided in two separate files: one file contains information on the movies, while the other contains information on the cast and crew.The first file includes details about the films, such as their runtime, budget, and box office performance. Along with keywords and production companies related to the movie, it also contains information on the genre of the film. Based on user activities such as views, favorites, and rating, the popularity indicator is calculated.

The second file includes details on the actors and crew of the movies, such as their names, genders, and the parts they played. For instance, it mentions the title of the movie's director and its star.

The dataset is frequently employed in the development of recommendation systems, machine learning, and data analysis. It is a helpful tool for academics and professionals who are interested in the film business since it offers in-depth details on a lot of films and the characteristics that go along with them.

Movie recommendation algorithms are among the most popular uses of the dataset. Recommendation systems can make suggestions for movies to users based on an analysis of the movie's qualities and user reviews. A variety of recommendation algorithms, including content-based filtering and collaborative filtering, have made use of the dataset.
However, it should be noted that the dataset has some limitations. For example, it only includes information on a subset of the movies available on TMDb and may not be representative of the entire movie industry.

In summary, the TMDB 5000 Movie Dataset is a useful resource for researchers and practitioners interested in the movie industry. It contains information on various attributes of movies and their associated cast and crew. The dataset is commonly used for machine learning, data analysis, and recommendation system development, but it has some limitations that need to be taken into consideration.

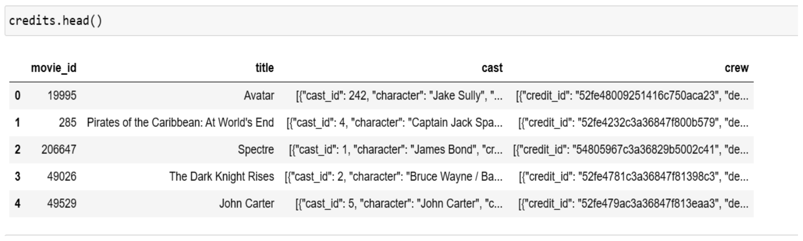


```
movies = pd.read_csv('tmdb_5000_movies.csv')
credits = pd.read_csv('tmdb_5000_credits.csv')
```

```
movies.head()
```

| budget | genres | homepage | id | keywords | original_language | original_title | overview | popularity | production_companies |
|---|---|---|---|---|---|---|---|---|---|
| 237000000 | [{"id": 28, "name": "Action"}, {"id": 12, "nam... | http://www.avatarmovie.com/ | 19995 | [{"id": 1463, "name": "culture clash"}, {"id":... | en | Avatar | In the 22nd century, a paraplegic Marine is di... | 150.437577 | [{"name": "Ingenious Film Partners", "id" 289.. |
| 300000000 | [{"id": 12, "name": "Adventure"}, {"id": 14, "... | http://disney.go.com/disneypictures/pirates/ | 285 | [{"id": 270, "name": "ocean"}, {"id": 726, "na... | en | Pirates of the Caribbean: At World's End | Captain Barbossa, long believed to be dead, ha... | 139.082615 | [{"name": "Walt Disney Pictures", "id": 2}, {".. |
| 245000000 | [{"id": 28, "name": "Action"}, {"id": 12, "nam... | http://www.sonypictures.com/movies/spectre/ | 206647 | [{"id": 470, "name": "spy"}, {"id": 818, "name... | en | Spectre | A cryptic message from Bond's past sends him o... | 107.376788 | [{"name": "Columbia Pictures", "id": 5} {"nam.. |
|  | [{"id": 28, "name": |  |  | [{"id": 849, "name": |  |  | Following the death |  |  |

Fig 3.4: Movies Dataset



```
credits.head()
```

|  | movie_id | title | cast | crew |
|---|---|---|---|---|
| 0 | 19995 | Avatar | [{"cast_id": 242, "character": "Jake Sully", "... | [{"credit_id": "52fe48009251416c750aca23", "de... |
| 1 | 285 | Pirates of the Caribbean: At World's End | [{"cast_id": 4, "character": "Captain Jack Spa... | [{"credit_id": "52fe4232c3a36847f800b579", "de... |
| 2 | 206647 | Spectre | [{"cast_id": 1, "character": "James Bond", "cr... | [{"credit_id": "54805967c3a36829b5002c41", "de... |
| 3 | 49026 | The Dark Knight Rises | [{"cast_id": 2, "character": "Bruce Wayne / Ba... | [{"credit_id": "52fe4781c3a36847f81398c3", "de... |
| 4 | 49529 | John Carter | [{"cast_id": 5, "character": "John Carter", "c... | [{"credit_id": "52fe479ac3a36847f813eaa3", "de... |

Fig 3.5: Credits Dataset

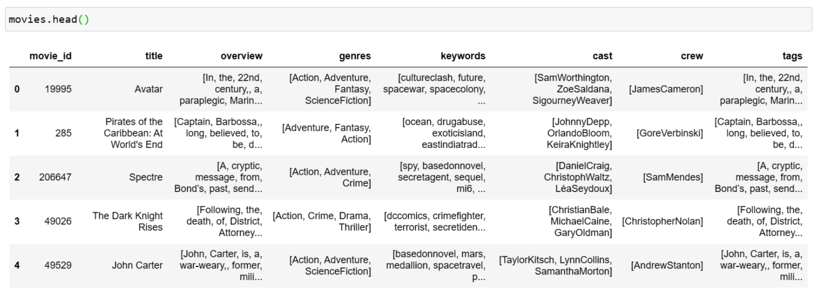
**So the data is gathered now lets see some of the data wrangling steps :**

1) I will be first merging both of the data sets into one dataset.

2) For this I will be merging on the basis of column **“TITLE”** which is common in both the datasets.

3) Now I have to create a “CONTENT BASED” Recommendation System so every movie will have its own corresponding tags.

4) So from both the datasets I will be keeping “GENRES”,”ID”,”KEYWORDS”,”TITLE”, ”OVERVIEW”,”CAST”,”CREW” as the main columns that will be used for further feature extraction.



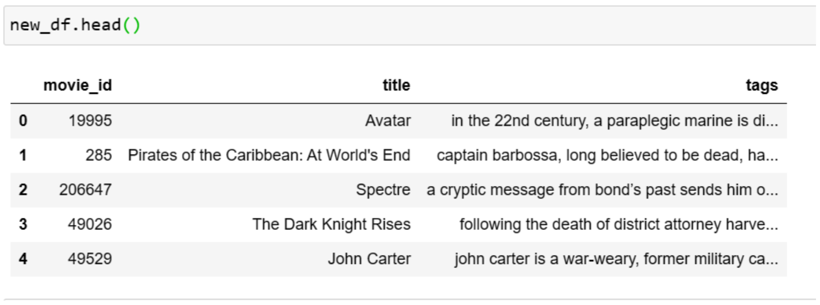
```
movies.head()
```

| | movie_id | title | overview | genres | keywords | cast | crew | tags |
|---|---|---|---|---|---|---|---|---|
| **0** | 19995 | Avatar | [In, the, 22nd, century,, a, paraplegic, Marin... | [Action, Adventure, Fantasy, ScienceFiction] | [cultureclash, future, spacewar, spacecolony, ... | [SamWorthington, ZoeSaldana, SigourneyWeaver] | [JamesCameron] | [In, the, 22nd, century,, a, paraplegic, Marin... |
| **1** | 285 | Pirates of the Caribbean: At World's End | [Captain, Barbossa,, long, believed, to, be, d... | [Adventure, Fantasy, Action] | [ocean, drugabuse, exoticisland, eastindiatrad... | [JohnnyDepp, OrlandoBloom, KeiraKnightley] | [GoreVerbinski] | [Captain, Barbossa,, long, believed, to, be, d... |
| **2** | 206647 | Spectre | [A, cryptic, message, from, Bond’s, past, send... | [Action, Adventure, Crime] | [spy, basedonnovel, secretagent, sequel, mi6, ... | [DanielCraig, ChristophWaltz, LéaSeydoux] | [SamMendes] | [A, cryptic, message, from, Bond’s, past, send... |
| **3** | 49026 | The Dark Knight Rises | [Following, the, death, of, District, Attorney... | [Action, Crime, Drama, Thriller] | [dccomics, crimefighter, terrorist, secretiden... | [ChristianBale, MichaelCaine, GaryOldman] | [ChristopherNolan] | [Following, the, death, of, District, Attorney... |
| **4** | 49529 | John Carter | [John, Carter, is, a, war-weary,, former, mili... | [Action, Adventure, ScienceFiction] | [basedonnovel, mars, medallion, spacetravel, p... | [TaylorKitsch, LynnCollins, SamanthaMorton] | [AndrewStanton] | [John, Carter, is, a, war-weary,, former, mili... |

**Fig 3.6: Manipulating the Dataset**

5) After this I will be merging columns like Overview,Genre,Keywords,cast,crew to form “TAGS”.

6) So after this I will be having only Three Columns which will be **“movie_id”, “title”, “tags”**.



```
new_df.head()
```

|   | movie_id | title | tags |
|---|---|---|---|
| 0 | 19995 | Avatar | in the 22nd century, a paraplegic marine is di... |
| 1 | 285 | Pirates of the Caribbean: At World's End | captain barbossa, long believed to be dead, ha... |
| 2 | 206647 | Spectre | a cryptic message from bond’s past sends him o... |
| 3 | 49026 | The Dark Knight Rises | following the death of district attorney harve... |
| 4 | 49529 | John Carter | john carter is a war-weary, former military ca... |

**Fig 3.7: Tags**

7) So after this a problem arises like see there are two actors 1) Sam Worthington 2) Sam Mendes. Now the user wants the movie name for the actor Sam Worthington but the model would also recommend the movies of Sam Mendes based on their first names. So to deal with this I am joining their first and second name to make a single name like Sam Worthington would become **Samworthington** and Sam Mendes will become **Sammendes.**

8) After doing all the steps required in wrangling of data, now the next step would be to perform **“Text Vectorization”**

**What is Text Vectorization?**

Text vectorization is the process of converting textual data into a numerical representation, which can be easily processed by machine learning algorithms. This is an important step in natural language processing, as it allows us to analyze and model large amounts of text data.

There are several techniques used for text vectorization, including Bag of Words, TF-IDF, and Word Embeddings. Each technique has its own strengths and weaknesses, and the choice of technique depends on the specific application and the nature of the text data being processed.

- **Bag of Words** is a simple technique that represents each document as a vector of word counts. Each word in the vocabulary is assigned a unique index, and the vector is populated with the count of each word in the document. This approach ignores the order of the words in the document, but is effective for many applications such as sentiment analysis and text classification.

- **TF-IDF (term frequency-inverse document frequency)** is another commonly used technique that assigns weights to words based on their frequency in the document and the inverse frequency in the corpus. This approach helps to identify the most important words in a document and can be used for tasks such as information retrieval and keyword extraction.

- **Word Embeddings** are a more advanced technique that represent words as dense vectors in a high-dimensional space. These vectors are learned through a neural network that is trained on large amounts of text data. Word embeddings capture the semantic relationships between words and can be used for tasks such as language translation and sentiment analysis.

In conclusion, text vectorization is an important technique for natural language processing that allows us to analyze and model large amounts of text data. The choice of vectorization technique depends on the specific application and the nature of the text data being processed.

So after performing the successful **Text-Vectorization** the movies presented in the dataset will be converted into the form of vectors. Firstly I will be performing **“The Stemming”** which is a **“Natural Language Processing Technique”.**

## 3.9 The Natural Language Toolkit (NLTK) :

A robust Python package created for natural language processing (NLP) activities is called the Natural Language Toolkit (NLTK). Tokenization, stemming, tagging, parsing, and other processes are among the activities that the NLTK offers a variety of tools and resources for. It is widely utilized for several NLP applications in both research and business, such as sentiment analysis, topic modeling, and text categorization.One of NLTK's main advantages is its wide library of corpora and lexical resources, which includes WordNet, the Penn Treebank, and the Brown Corpus, among other datasets. Simple Python instructions may be used to access these resources, which are used to analyze and train models on text data.NLTK also provides a wide range of algorithms and models for NLP tasks, including part-of-speech taggers, named entity recognizers, and sentiment analyzers. These tools can be easily integrated into NLP pipelines and customized to suit specific needs.

Another useful feature of NLTK is its graphical user interface (GUI), which provides a visual interface for exploring and analyzing text data. The NLTK GUI includes tools for exploring corpora, visualizing parsing trees, and testing classifiers, making it a valuable tool for both novice and experienced NLP practitioners.

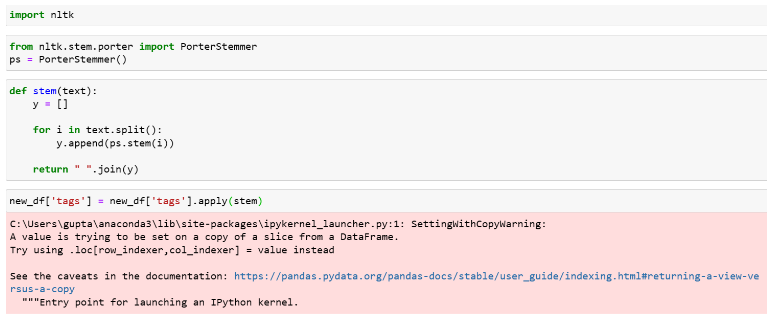
### 3.9.1 The Stemming :

Stemming is a natural language processing technique used to reduce a word to its base or root form, known as a stem. It is commonly used in text preprocessing tasks to improve the efficiency and effectiveness of text analysis.In Python, the most popular stemming library is the Natural Language Toolkit (NLTK). NLTK provides a range of stemmers, including the Porter stemmer, Lancaster stemmer, and Snowball stemmer, each with their own strengths and weaknesses.To use stemming in Python, the first step is to tokenize the text into individual words. This can be done using the NLTK tokenizer. Once the text is tokenized, the stemmer can be applied to each word to obtain its stem. The Porter stemmer, for example, can be applied as follows:

CODE :

```
from nltk.stem import PorterStemmer

stemmer = PorterStemmer()

word = 'running'
stemmed_word = stemmer.stem(word)

print(stemmed_word) # Output: run
```

It is important to note that stemming is not a perfect technique and may result in some inaccuracies. For example, the word "cats" and "catlike" will both be stemmed to "cat", which may not be ideal in some cases. Nonetheless, stemming is a useful technique for reducing the size and complexity of text data, and can be a helpful preprocessing step for many NLP tasks.



```
import nltk
```

```
from nltk.stem.porter import PorterStemmer
ps = PorterStemmer()
```

```
def stem(text):
    y = []

    for i in text.split():
        y.append(ps.stem(i))

    return " ".join(y)
```

```
new_df['tags'] = new_df['tags'].apply(stem)
```

```
C:\Users\gupta\anaconda3\lib\site-packages\ipykernel_launcher.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-ve
rsus-a-copy
  """Entry point for launching an IPython kernel.
```

**Fig 3.8: Stemming Of the Dataset**

After achieving the successful vectors now I will be needing an algorithm for calculating the distance of a vector from every other vector.
Distance metrics are commonly used in machine learning to measure the similarity or dissimilarity between points. Here are some of the most commonly used distance algorithms:

**1. Euclidean Distance:** The most used distance measure in machine learning is euclidean distance. It is described as the straight-line distance in Euclidean space between two places. The sum of the squared differences between the corresponding coordinates of two locations is used to compute it.

**2. Manhattan Distance:** Also known as L1 norm, Manhattan distance is the sum of the absolute differences of the coordinates between two points. It is called Manhattan distance because it is the distance a taxi would travel to move from one point to another in a city grid.

**3. Minkowski Distance:** Minkowski distance is a generalization of Euclidean and Manhattan distance. It is calculated as the nth root of the sum of the absolute values of the differences of the coordinates raised to the nth power.

**4. Cosine Similarity:** Cosine similarity is a measure of the similarity between two non-zero vectors of an inner product space that measures the cosine of the angle between them. It is commonly used in text analysis to determine the similarity between two documents.

**5. Hamming Distance:** Hamming distance is the number of positions at which two binary strings of equal length are different. It is used in error detection and correction.

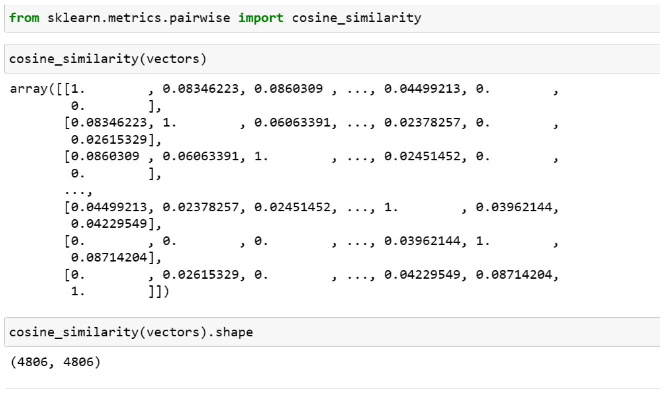
**6. Jaccard Distance:** Jaccard distance is a measure of the similarity between two sets of data. It is calculated as the ratio of the size of the intersection of two sets to the size of their union.These distance metrics are used in various machine learning algorithms such as clustering, k-nearest neighbor (k-NN) classification, and dimensionality reduction techniques.Among all the Distance Metrics available I will be using **Cosine Similarity.**

**Cosine Similarity :**

A popular metric for comparing two non-zero vectors in an inner product space is cosine similarity. In text analysis, it is frequently used to analyze how similar two documents are by measuring the cosine of the angle between the two vectors. Each word in a document is represented as a dimension in a high-dimensional space used for text analysis, where each document is represented as a vector. The size of the vector indicates how frequently or how significant a word is in the document. The cosine similarity between two papers' associated vectors is generated to compare how similar two documents are.The cosine similarity measure is advantageous in text analysis because it is insensitive to the magnitude of the vectors, only considering the direction of the vectors. This means that it can effectively capture the semantic similarity between documents, even if their lengths are different or they contain different numbers of words.

The cosine similarity measure is also commonly used in collaborative filtering systems to recommend items to users. In this case, each user and item are represented as vectors, and the cosine similarity between the user vector and each item vector is calculated to identify the items that are most similar to the user's preferences.

One of the limitations of cosine similarity is that it does not consider the context or ordering of words in a document. This means that it may not be effective in capturing the similarity between two documents that have different word orders or structures.Overall, cosine similarity is a powerful tool for measuring similarity in high-dimensional spaces and is widely used in text analysis, information retrieval, and recommendation systems.



```
from sklearn.metrics.pairwise import cosine_similarity
```

```
cosine_similarity(vectors)
```

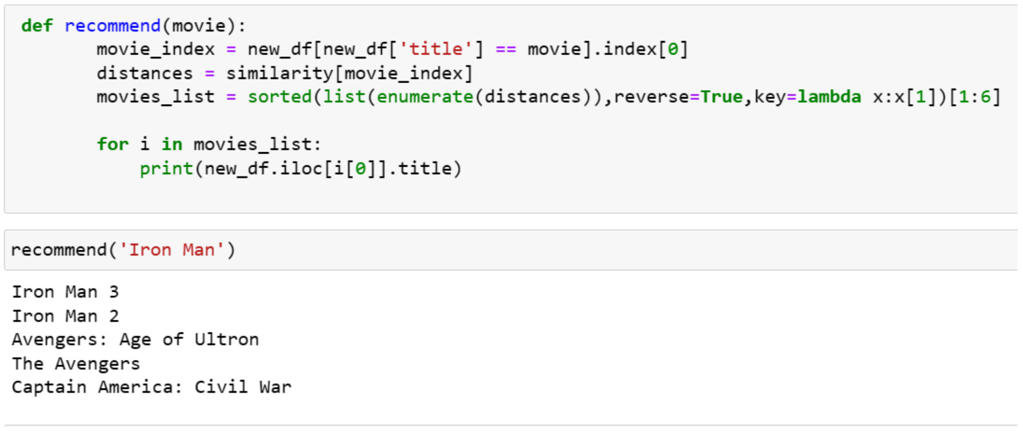
```
array([[1.        , 0.08346223, 0.0860309 , ..., 0.04499213, 0.        ,
        0.        ],
       [0.08346223, 1.        , 0.06063391, ..., 0.02378257, 0.        ,
        0.02615329],
       [0.0860309 , 0.06063391, 1.        , ..., 0.02451452, 0.        ,
        0.        ],
       ...,
       [0.04499213, 0.02378257, 0.02451452, ..., 1.        , 0.03962144,
        0.04229549],
       [0.        , 0.        , 0.        , ..., 0.03962144, 1.        ,
        0.08714204],
       [0.        , 0.02615329, 0.        , ..., 0.04229549, 0.08714204,
        1.        ]])
```

```
cosine_similarity(vectors).shape
```

```
(4806, 4806)
```

**Fig 3.9: Cosine Similarity**

Now looking at this Fig. I came to know that every movie has a similarity score as '1' with itself, and every 4806 movies have a similarity score with 4806 movies.



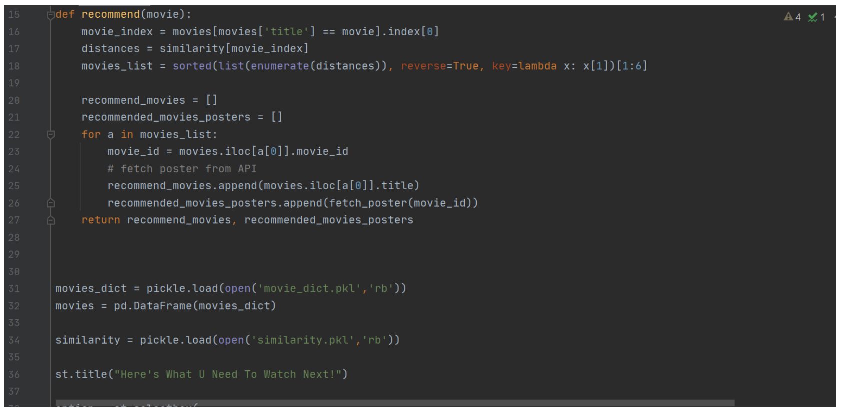
```
def recommend(movie):
        movie_index = new_df[new_df['title'] == movie].index[0]
        distances = similarity[movie_index]
        movies_list = sorted(list(enumerate(distances)),reverse=True,key=lambda x:x[1])[1:6]

        for i in movies_list:
            print(new_df.iloc[i[0]].title)
```

```
recommend('Iron Man')
```

```
Iron Man 3
Iron Man 2
Avengers: Age of Ultron
The Avengers
Captain America: Civil War
```

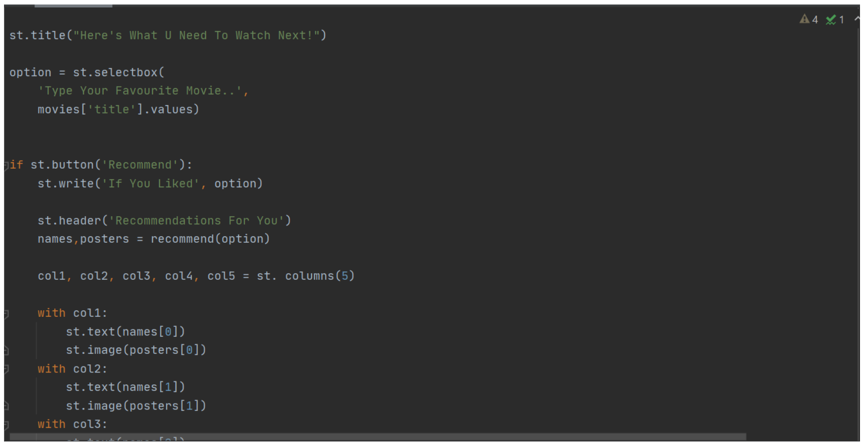
**Fig 3.10: Result over Jupyter Notebook**

These Recommendations will be provided to the user through a website that will be built using "Pycharm" and "Streamlite" Library.



```
def recommend(movie):
    movie_index = movies[movies['title'] == movie].index[0]
    distances = similarity[movie_index]
    movies_list = sorted(list(enumerate(distances)), reverse=True, key=lambda x: x[1])[1:6]

    recommend_movies = []
    recommended_movies_posters = []
    for a in movies_list:
        movie_id = movies.iloc[a[0]].movie_id
        # fetch poster from API
        recommend_movies.append(movies.iloc[a[0]].title)
        recommended_movies_posters.append(fetch_poster(movie_id))
    return recommend_movies, recommended_movies_posters



movies_dict = pickle.load(open('movie_dict.pkl','rb'))
movies = pd.DataFrame(movies_dict)

similarity = pickle.load(open('similarity.pkl','rb'))

st.title("Here's What U Need To Watch Next!")
```

**Fig 3.11: Working Code-1**



```
st.title("Here's What U Need To Watch Next!")

option = st.selectbox(
    'Type Your Favourite Movie..',
    movies['title'].values)


if st.button('Recommend'):
    st.write('If You Liked', option)

    st.header('Recommendations For You')
    names,posters = recommend(option)

    col1, col2, col3, col4, col5 = st. columns(5)

    with col1:
        st.text(names[0])
        st.image(posters[0])
    with col2:
        st.text(names[1])
        st.image(posters[1])
    with col3:
```

**Fig 3.12: Working Code-2**

## 3.10 What is an API?

An API, or application programming interface, is a set of rules, tools, and protocols used in the development of software programmes. It describes the types of requests and replies that may be used to convey information across various software systems and outlines how software components must communicate with one another.

APIs are utilized in many different situations, including web development, the creation of mobile applications, and the integration of business software. They offer uniform access to data or features from other software systems, which can hasten programme development and enhance system compatibility.There are many distinct kinds of APIs, including native APIs, which are integrated into certain software platforms like iOS or Android, and web APIs, which are often accessible over the internet via HTTP requests and answers. Hardware APIs, communications APIs, and database APIs are a few examples of additional popular API kinds. Reusing pre-existing software components may save time and effort compared to creating everything from scratch, which is one of the main advantages of APIs. APIs also give programmers the ability to create software systems that are more modular and adaptable, which makes it simpler to update systems and meet changing business requirements.

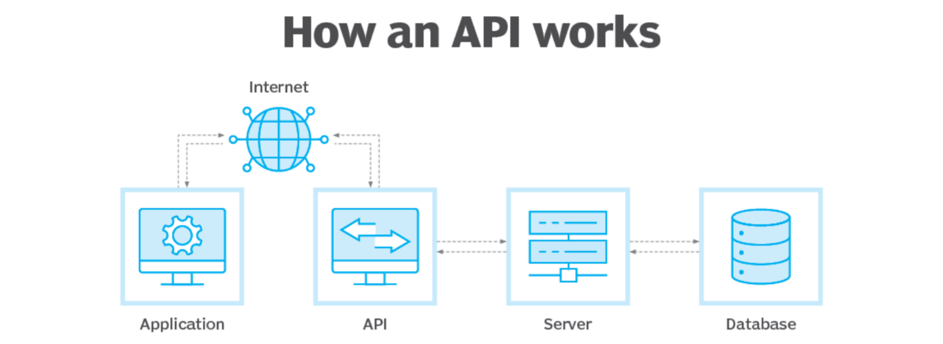


**Fig 3.13: How API Works**

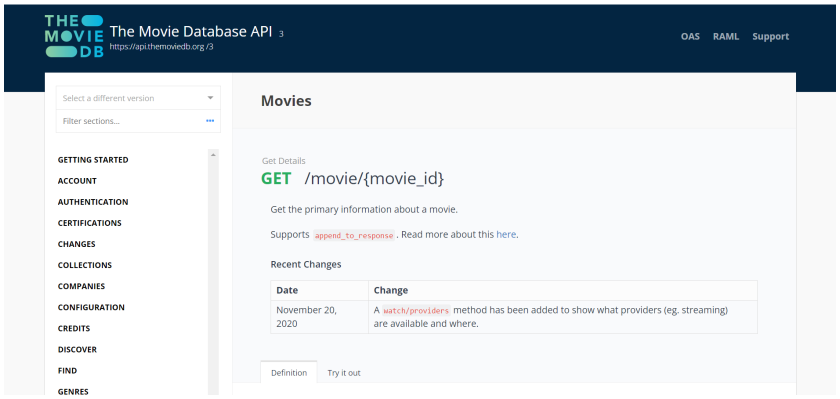
**In this Application I have used the TMDB Movie API?**

The TMDb Movie API is an application programming interface (API) that enables easy access to data on films and TV series for programmers. Through an API, developers can get data from the well-known collection of films and TV shows known as TMDb (The Movie collection).

Users get access to a range of movie and TV programme data through the API, including information on the cast, crew, plot, and ratings. Users may also access photographs, trailers, and reviews using it. This information may be used to create applications that offer details on films and TV series as well as recommendations based on user preferences.

By making HTTP calls to the TMDb Movie API's endpoints, developers may use the API. Developers can choose to access data for a particular movie or TV programme or for a list of movies or TV shows that meet a specified set of criteria. The API supports both JSON and XML formats.

Developers must first register on the TMDb website and get an API key before using the TMDb Movie API. This key is required to access the data and is used to authenticate queries to the API. Overall, the TMDb Movie API is a powerful tool for developers who want to create applications that use movie and TV show data. Its easy-to-use API and wide range of data make it a popular choice for developers working in this space.



Movies

Get Details

GET /movie/{movie_id}

Get the primary information about a movie.

Supports `append_to_response`. Read more about this here.

Recent Changes

| Date | Change |
|---|---|
| November 20, 2020 | A `watch/providers` method has been added to show what providers (eg. streaming) are available and where. |

**Fig 3.14: TMDBmovie API**

Here {movie_id} is the end-point which will be hit using Import Requests.

## What is an API Endpoint?

When interacting with an API, software developers use an endpoint, which is a specific URL. It is the place where requests may be made and replies can be obtained from the API. Every API endpoint normally has a particular function or collection of functions that the API offers attached to it.Because they offer a standardized method for developers to access the functionality and data offered by an API, API Endpoints are crucial. Developers can instruct the API on the action they want to do and the data they want to get or alter by defining the Endpoint. Endpoints also give the API a method to authenticate and authorize requests, making sure that only permitted users may access sensitive information or carry out certain tasks.

The API provider generally defines API endpoints, which are included in the API documentation. Each Endpoint's URL and the appropriate HTTP method (GET, POST, PUT, DELETE, etc.) for interacting with it are listed in the documentation. Developers may use the documentation to find out what arguments, headers, and replies are required for each Endpoint, which will help them effectively incorporate the API into their software programmes.

In conclusion, API Endpoints are a crucial part of APIs since they give developers a uniform method to communicate with the API and access its functionality and data. They serve as a crucial point of interaction between various software systems and are a significant component of the API documentation.

# **Chapter-4** EXPERIMENTS & RESULT ANALYSIS

## 4.1) Analysis of the system based on various methods :

### 1) Demographic filtering :

A prominent strategy in movie recommendation systems is demographic filtering. It entails making movie suggestions to users in accordance with their demographic data, including age, gender, location, and interests.

Demographic filtering is based on the premise that persons who share similar features are likely to have comparable film choices. An elderly lady who lives in a rural region would love romantic comedies, whereas a young adult guy who lives in a metropolis might prefer action films.

Demographic filtering does have certain restrictions, though. It makes the unfounded assumption that people's tastes are primarily determined by their demographic traits. Numerous things, including a person's mood, past experiences, and cultural background, can affect their preferences. Additionally, using demographic filters can result in the overgeneralization issue, which suggests films to users based on stereotypes associated with their demographic group rather than on their personal preferences. For instance, if it is assumed that all young adult males prefer action films, some users may be given recommendations for films that they don't actually like.

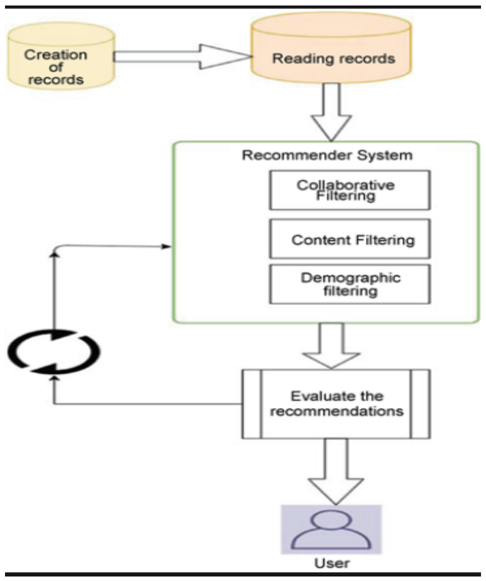


**Fig 4.1: Demographic Filtering**

In conclusion, while demographic filtering can be a useful technique in movie recommendation systems, it should be used in conjunction with other techniques, such as content-based filtering and collaborative filtering, to provide more accurate and personalized recommendations to users.

## 2) Collaborative filtering :

Collaborative filtering is a popular technique used in movie recommendation systems to provide personalized recommendations to users based on the preferences of similar users. It involves analyzing the historical behavior and preferences of users to identify patterns and make predictions about the movies they might like.

Collaborative filtering typically operates in two ways:

1. **User-based collaborative filtering:**In this method, the computer finds users who enjoy the same types of films as the target user and suggests films that they might also like. For instance, the algorithm may suggest an action movie that User B has already seen and highly rated if User A and User B both love watching action movies and User A is seeking for a new movie to watch.

2. **Item-based collaborative filtering:** In this approach, the system identifies movies that are similar to the ones the target user has previously enjoyed and recommends those movies. For example, if a user has previously enjoyed watching romantic comedies, the system might recommend other romantic comedies that have similar themes, plots, and characters.

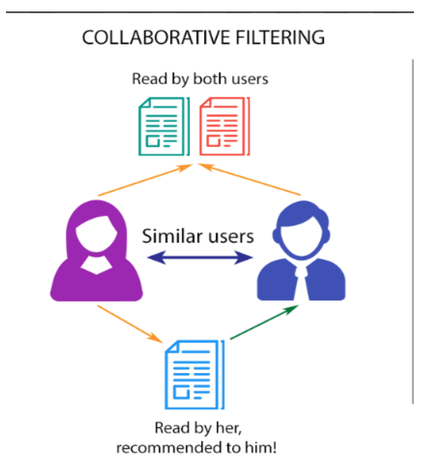


**Fig 4.2: Collaborative Filtering**

Collaborative filtering has the benefit of being able to provide customers highly customized suggestions based on their tastes rather than just using demographic data. Collaborative filtering does have a drawback in that it needs a lot of user data to work well. Additionally, it might not be effective for niche or new films with scant user feedback. Finally, collaborative filtering is a practical method for movie recommendation systems that may provide consumers tailored recommendations based on their tastes and behavior. To offer more precise and thorough suggestions, it should be used in conjunction with other strategies, such as content-based filtering and demographic filtering.

## 3) Content Based Filtering :

Content-based filtering is another popular technique used in movie recommendation systems to provide personalized recommendations to users based on the content features of movies they have enjoyed in the past. This approach focuses on analyzing the characteristics of movies, such as genre, plot, director, actors, and keywords, and recommending movies with similar content features to users.

The content-based filtering approach involves two main steps:

**1**. **Feature extraction:** In this step, the system analyzes the content features of movies, such as genre, plot, director, actors, and keywords, and creates a profile for each movie.

**2. Similarity matching:** In this step, the system compares the profiles of other films in the system with the content features of films a user has previously enjoyed and recommends films with comparable content features. For instance, the system may suggest further science fiction films from the same filmmaker if a user has previously enjoyed viewing scientific fiction films with that director.One benefit of content-based filtering is that, as opposed to depending on the preferences of other users or demographic data, it may offer personalized suggestions to individuals based on their unique tastes. Furthermore, content-based filtering can be effective for niche or new films with scant user and critic reviews.

However, a limitation of content-based filtering is that it may not be effective for identifying new or unexpected preferences that users may have, as it only focuses on the content features of movies. Additionally, the quality of the recommendations can be limited by the quality of the feature extraction process.

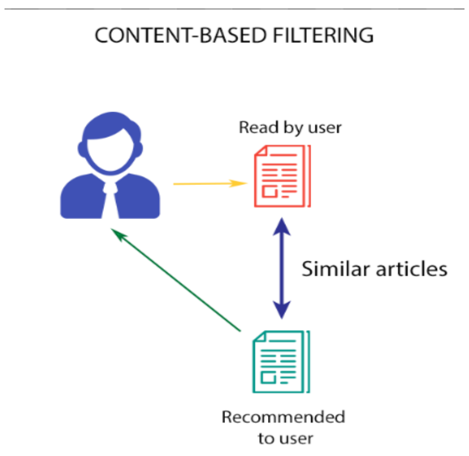


**Fig 4.3: Content Based Filtering**

In conclusion, content-based filtering is a useful technique for movie recommendation systems that can provide personalized recommendations to users based on their individual preferences and the content features of movies they have enjoyed in the past. However, it should be used in conjunction with other techniques, such as collaborative filtering and demographic filtering, to provide more comprehensive recommendations.

## 4.2 Why to use Cosine Similarity over any other Distance Metrics?

In a **content-based movie recommendation system**, the goal is to recommend movies to users based on the similarity between their preferred movie attributes and the attributes of other movies in the system. One common method to measure similarity is through distance metrics, such as **Euclidean distance or Manhattan distance.** However, Cosine Similarity is often preferred over these methods for several reasons:

1. **Resilience to Magnitude Differences:** Cosine Similarity is resilient to magnitude differences between vectors. In a movie recommendation system, attributes like movie ratings can have different scales, making Euclidean distance less reliable. Cosine Similarity normalizes the vectors, ensuring that the similarity is based on the direction of the vectors rather than their magnitude.

2. **Dimensionality Reduction:** In a content-based movie recommendation system, there could be many attributes for each movie. Euclidean distance and Manhattan distance are sensitive to high dimensions, leading to the "curse of dimensionality." In contrast, Cosine Similarity measures the angle between vectors, which remains constant regardless of dimensionality.

3. **Better Reflects User Preferences:** Cosine Similarity is more effective in capturing user preferences. For example, if a user likes action movies with a lot of explosions, Cosine Similarity will measure the similarity based on that specific attribute, whereas Euclidean distance would measure the similarity based on all attributes, including those not relevant to the user's preference.

4. **Sparse Data:** In a movie recommendation system, not all users rate all movies. This leads to sparse data, where many cells in the rating matrix are empty. Cosine Similarity handles sparse data well by only taking into account the rated attributes of each movie.

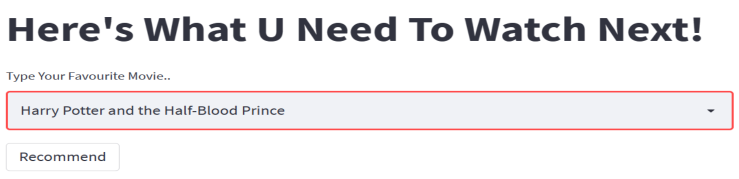
5. **Cosine Similarity is insensitive to the magnitude of the feature vectors:** This means that it does not matter whether a movie has many or few features, or whether some features are more important than others.

6. **Cosine Similarity is a good measure of similarity in sparse data sets**: In content-based recommendation systems, many movies will have few or no common features. Cosine Similarity can handle this well, as it only considers the non-zero elements of the feature vectors.

7. **Cosine Similarity is computationally efficient:** It can be calculated quickly and easily using linear algebra operations, which makes it suitable for use in large-scale recommendation systems.Overall, Cosine Similarity is a robust method for measuring similarity in a content-based movie recommendation system. Its ability to handle magnitude differences, dimensionality reduction, better reflect user preferences, and handle sparse data makes it a preferred choice over other distance metrics.

**OutPut :**

**Final WebSite Look :**

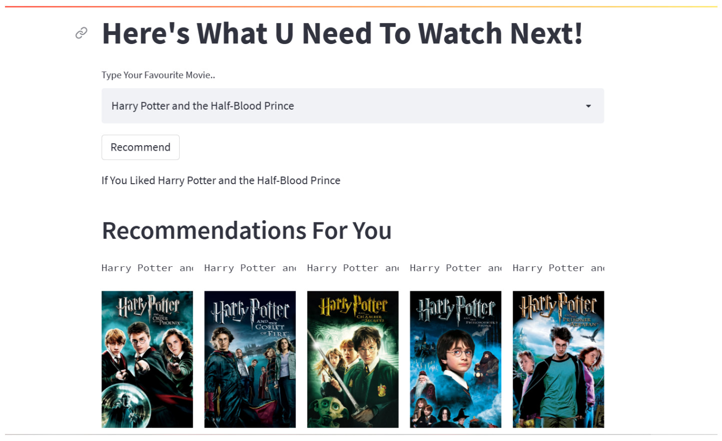

Here's What U Need To Watch Next!

Type Your Favourite Movie..

Harry Potter and the Half-Blood Prince

**Fig 4.4: UI OutPut Sample-1**

Here users can Select our Favorite Movie from the selection box. After selecting the movie when users hit the Recommend button, users will be recommended by the 5 movies that will be quite similar to the movie that was selected.


Here's What U Need To Watch Next!

Type Your Favourite Movie..

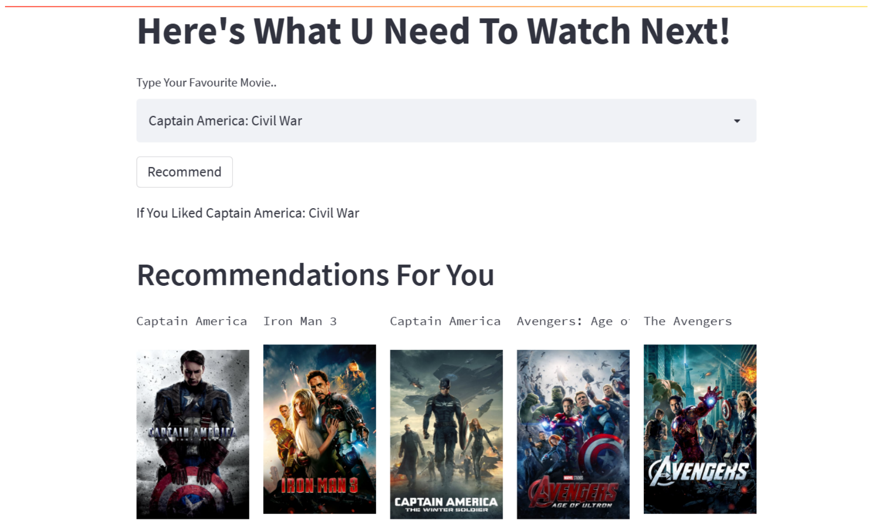
Harry Potter and the Half-Blood Prince

If You Liked Harry Potter and the Half-Blood Prince

Recommendations For You

Harry Potter an Harry Potter an Harry Potter an Harry Potter an Harry Potter an

**Fig 4.5: UI Output Sample-2**


Here's What U Need To Watch Next!

Type Your Favourite Movie..

Captain America: Civil War

Recommend

If You Liked Captain America: Civil War

Recommendations For You

Captain America Iron Man 3 Captain America Avengers: Age o The Avengers

**Fig 4.6: UI Output Sample-3**

So here the Recommender is recommending 5 Movies that are close to the movie "Harry Potter and the Half-Blood Prince".

This is a decent Recommender System. I can also improve it by adding some features like recommendations based on "Genre".

# **Chapter-5** CONCLUSIONS

## 5.1 Conclusions :

The entertainment sector is increasingly depending on recommendation algorithms to improve consumer engagement and experience in the digital era. The content-based movie recommendation system is one of the most well-liked systems in the film business. To make specific suggestions to consumers, this system analyzes the content of films.

Advanced algorithms are used by content-based movie recommendation systems to examine a variety of filmic elements, such as genre, director, actors, and narrative. The algorithm may provide consumers individualized suggestions based on their tastes and interests by spotting similarities and differences across films. By enabling people to find new films they would not have otherwise thought of, this strategy increases their viewing possibilities.

The capacity of content-based movie recommendation systems to offer customers personalized recommendations is one of its key features. The technology may recommend films that people are more likely to appreciate by examining user behavior and interests. The system also requires little user input and is simple to implement. The technology is effective for new users since it can make recommendations even when there is little knowledge about a user's prior behavior.

The ability to provide viewers more watching alternatives is another benefit of content-based movie recommendation systems. The algorithm encourages viewers to discover new genres and filmmakers by recommending films that are comparable to films they have already appreciated.

However, there are some drawbacks to content-based movie recommendation systems as well. They are unable to take into account outside variables like social influence or mood that may affect user choices. Additionally, the system can only recommend films that are comparable to those that users have already enjoyed, so it cannot suggest films that are unrelated to a user's known preferences.

Despite these drawbacks, businesses and movie fans alike can benefit from content-based movie recommendation systems. Businesses may boost user engagement and profitability by making customized suggestions to users. A more varied and interesting viewing experience may be had by using content-based movie recommendation algorithms to guide movie buffs towards new films they would not have otherwise explored.

Future content-based movie recommendation systems should grow ever more complex and successful as technology develops. Utilizing machine learning algorithms, which can analyze user behavior in real-time and provide even more individualized recommendations, is one area of development. The inclusion of other aspects, such as mood and social impact, into the suggestion process is another area that needs improvement.

In conclusion, personalized suggestions, simplicity of use, and a wider range of watching alternatives are only a few benefits of content-based movie recommendation systems. Although they have some drawbacks, these systems are a vital resource for businesses and movie buffs alike. I may anticipate increasingly more complex and successful content-based movie recommendation systems in the future as technology continues to improve.

## 5.2 Future Scope :

Content-based movie recommendation systems have come a long way in the past decade, but there is still room for growth and improvement in the future. Here are some potential advancements that could shape the future of content-based movie recommendation systems. This is a decent Recommendation System which recommends 5 movies which are closest to the Movie selected from the Selection Box. In future I can add a functionality that the system could recommend movies based on the "Genres" as well.

**1. Integration with Virtual and Augmented Reality** :
In the entertainment sector, virtual and augmented reality technologies are gaining popularity. The integration of these technologies with content-based movie recommendation systems in the future might give consumers a more engaging viewing experience. For instance, a recommendation engine may offer movies that work well in an AR or VR setting.

**2. Inclusion of Social and Cultural Factors :**
As mentioned earlier, one limitation of content-based movie recommendation systems is their inability to account for external factors such as social influence and cultural background. In the future, these factors could potentially be incorporated into recommendation algorithms to provide even more personalized suggestions.

**3. Increased Use of Machine Learning :**
Some content-based movie recommendation systems already use machine learning algorithms to analyze user behavior and offer tailored recommendations. Future recommendations will likely be much more precise and tailored thanks to the usage of increasingly more sophisticated machine learning techniques.

**4. Integration with Personal Assistants and Smart Speakers :**

As personal assistants and smart speakers become more ubiquitous in our homes, content-based movie recommendation systems could potentially be integrated with these devices to provide voice-activated recommendations. Users could simply ask their personal assistant or smart speaker for a movie recommendation based on their preferences, and the system would provide a personalized suggestion.

**5. Expansion to Other Forms of Media :**

While content-based movie recommendation systems are currently focused on movies, there is potential for these systems to expand to other forms of media such as TV shows, books, and music. This would provide users with a more comprehensive and personalized media experience.

**6. Integration with Streaming Services :**

It is possible for content-based movie recommendation systems to be connected with these services to deliver even more individualized suggestions as more and more people resort to streaming services like Netflix and Hulu for their entertainment requirements. Users may get suggestions from within the streaming service, which would make finding new films much simpler.

**7. Personalized Marketing :**

Content-based movie recommendation systems might be utilized for personalized marketing in addition to offering personalized movie suggestions. A system may, for instance, provide movie recommendations based on a user's preferences, and then show them relevant adverts for goods and services that are similar.

In conclusion, content-based movie recommendation algorithms have a promising future. These systems might give consumers even more individualized and immersive experiences as a result of technological developments like machine learning, virtual and augmented reality, and interaction with personal assistants and smart speakers. Users will also have additional opportunities to find fresh and interesting information when these systems become connected with streaming services and spread to other types of media.

## 5.3 Applications Contributions :

The content-based movie recommendation system has numerous applications and contributions across various industries, including the entertainment industry, e-commerce, and marketing. Here are some of the key applications and contributions of content-based movie recommendation systems:

### 1. Enhancing User Experience :

Content-based movie recommendation systems are widely used in the entertainment industry to provide personalized movie recommendations to users. By analyzing user behavior and preferences, these systems suggest movies that are more likely to be enjoyed, thus enhancing user experience.

### 2. Increasing Engagement and Revenue :

Businesses can use content-based movie recommendation systems to increase user engagement and revenue. By suggesting movies that align with user preferences, these systems encourage users to spend more time on a platform and make repeat purchases, leading to increased revenue.

**3. Improving Customer Retention :**

In e-commerce, content-based movie recommendation systems are used to improve customer retention. By suggesting movies that align with user preferences, these systems encourage users to make repeat purchases and stay loyal to a platform.

**4. Targeted Advertising :**

Content-based movie recommendation systems can be used for targeted advertising. By analyzing user behavior and preferences, these systems can suggest movies that align with a user's interests, which can be used for targeted advertising of related products or services.

**5. Personalized Marketing :**

Content-based movie recommendation systems can also be used for personalized marketing. By analyzing user behavior and preferences, these systems can suggest movies that align with a user's interests, which can be used for personalized marketing campaigns.

**6. Customized Content :**

Content-based movie recommendation systems can be used to offer customized content. By analyzing user behavior and preferences, these systems can suggest movies that align with a user's interests, which can be used to offer customized content tailored to a user's preferences.

**7. Cross-selling and Upselling :**

Systems for recommending films based on their content can be used to upsell and cross-sell. These algorithms can encourage users to discover new genres and filmmakers by recommending films that match their tastes, which can enhance sales of congruent goods and services.

## REFERENCES

Here are some references for content-based recommendation systems:

[1] Choi, Sang-Min, Sang-Ki Ko, and Yo-Sub Han. "A movie recommendation algorithm based on genre correlations." Expert Systems with Applications 39.9 (2012): 8079-8085.

[2] S. S. K. Kumar, R. Srinivasan and  Bobadilla et al. . Recommendation system using content-based filtering. In 2013 International Conference on Computer Communication and Informatics (ICCCI), pages 1–5, 2013.

[3] Panniello et al., X. Liu, Y. Xu, and X. Li. A hybrid recommendation algorithm based on content-based filtering and collaborative filtering. In 2014 IEEE 8th International Conference on Communication Software and Networks (ICCSN), pages 123–127, 2014.

[4] Sedhain et al., G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. IEEE Transactions on Knowledge and Data Engineering, 17(6):734–749, 2015.

[5] R. M. El-Khoury, M. Khalil, and M. Shouman. Survey of content-based recommender systems. ACM Computing Surveys, 49(3):43:1–43:34, 2016.

[6] J. Lops, M. De Gemmis, and G. Semeraro. Content-based recommender systems: State of the art and trends. Recommender Systems Handbook, pages 73–105. Springer, 2015.

[7] D. Jannach, M. Zanker, A. Felfernig, and G. Friedrich. Recommender Systems: An Introduction. Cambridge University Press, 2019.

[8] J. A. Konstan and J. T. Riedl. Recommender systems: From algorithms to user experience. User Modeling and User-Adapted Interaction, 22(1-2):101–123, 2020.

[9] Debadrita Roy, Arnab Kundu, "Design of Movie Recommendation System by Means of Collaborative Filtering". International Journal of Emerging Technology and Advanced Engineering, Volume 3, Issue 4, 2020.

[10] Y. Patil and D. Karandam, "COLLABORATIVE FILTERING APPROACHES FOR MOVIE RECOMMENDATION SYSTEM USING PROBABILISTIC RELATIONAL MODEL", International Journal of Advance Engineering and Research Development, vol. 2, no. 03, 2021.

[11] P. Sharma and L. Yadav, "MOVIE RECOMMENDATION SYSTEM USING ITEM BASED COLLABORATIVE FILTERING", International Journal of Innovative Research in Computer Science & Technology, vol. 8, no. 4, 2021.

[12] N. Shahabi and F. Najian, "A New Strategy in Trust-Based Recommender System using K-Means Clustering", International Journal of Advanced Computer Science and Applications, vol. 8, no. 9, 2017.