MOVIE RECOMMENDATION SYSTEM

A Project Report submitted in the partial fulfillment of the

requirements for the award of the degree

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

Submitted by

R. Sivarjun (19471A05B5)

K. Siva Rama Krishna (19471A0589)

Under the esteemed guidance of

Dr. M. Sireesha M.Tech.,Ph.D.

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

NARASARAOPETA ENGINEERING COLLEGE

AUTONOMOUS

Accredited by NAAC with A+ Grade and NBA under Cycle -1

NIRF rank in the band of 251-320 and an ISO 9001:2015 Certified

Approved by AICTE, New Delhi, Permanently Affiliated to JNTUK, Kakinada

KOTAPPAKONDA ROAD, YALAMANDA VILLAGE, NARASARAOPET- 522601

2022-2023

NARASARAOPETA ENGINEERING COLLEGE: NARASARAOPETA (AUTONOMOUS)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the main project entitled **"MOVIE RECOMMENDATION SYSTEM''** is a bonafide work done by R. Sivarjun (19471A05B5), K. Siva Rama Krishna (19471A0589) in partial fulfillment of the requirements for the award of the degree of **BACHELOR OF TECHNOLOGY** in the department of **COMPUTER SCIENCE AND ENGINEERING during 2022-2023.**

PROJECT GUIDE
Dr. M. Sireesha M.Tech.,Ph.D.

PROJECT CO-ORDINATOR
Dr. M. Sireesha M.Tech.,Ph.D.

HEAD OF THE DEPARTMENT
Dr. S. N. TirumalaRao M.Tech.,Ph.D.

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

We wish to express our thanks to carious personalities who are responsible for the completion of the project. We are extremely thankful to our beloved chairman sir **M. V. Koteswara Rao** **B.sc** who took keen interest in us in every effort throughout this course. We owe out gratitude to our principal **Dr. M. Sreenivasa Kumar** **M.Tech.,Ph.D(UK),MISTE,FIE(1)** for his kind attention and valuable guidance throughout the course.

We express our deep felt gratitude to **Dr. S. N. Tirumala Rao** **M.Tech.,Ph.D.** H.O.D. CSE department and our guide **Dr. M. Sireesha** **M.Tech.,Ph.D.** of CSE department whose valuable guidance and unstinting encouragement enable us to accomplish our project successfully in time.

We extend our sincere thanks to **Dr. M. Sireesha** **M.Tech.,Ph.D.** Associate Professor coordinator of the project for extending her encouragement. Their profound knowledge and willingness have been a constant source of inspiration for us throughout this project work.

We extend our sincere thanks to all other teaching and non-teaching staff to department for their cooperation and encouragement during our B.Tech degree. We have no words to acknowledge the warm affection, constant inspiration and encouragement that we receive from our parents. We affectionately acknowledge the encouragement received from our friends those who involved in giving valuable suggestions had clarifying out all doubts which had really helped us in successfully completing our project.

By

R. Sivarjun (19471A05B5)

K. Siva Rama Krishna (19471A0589)

ABSTRACT

A recommendation system is a system that provides suggestions to users for certain resources like books, movies, songs, etc., based on some data set. Movie recommendation systems usually predict what movies a user will like based on the attributes present in previously liked movies. Such recommendation systems are beneficial for organizations that collect data from large amounts of customers, and wish to effectively provide the best suggestions possible. A lot of factors can be considered while designing a movie recommendation system like the genre of the movie, actors present in it or even the director of the movie. The systems can recommend movies based on one or a combination of two or more attributes. The dataset used for the system is Movie Lens dataset. The data analysis tool used is Python.

NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)

INSTITUTE VISION AND MISSION

INSTITUTION VISION

To emerge as a Centre of excellence in technical education with a blend of effective student centric teaching learning practices as well as research for the transformation of lives and community.

INSTITUTION MISSION

M1: Provide the best class infra-structure to explore the field of engineering and research

M2: Build a passionate and a determined team of faculty with student centric teaching,imbibing experiential, innovative skills

M3: Imbibe lifelong learning skills, entrepreneurial skills and ethical values in students for addressing societal problems

NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

VISION OF THE DEPARTMENT

To become a centre of excellence in nurturing the quality Computer Science & Engineering professionals embedded with software knowledge, aptitude for research and ethical values to cater to the needs of industry and society.

MISSION OF THE DEPARTMENT

The department of Computer Science and Engineering is committed to

M1: Mould the students to become Software Professionals, Researchers and Entrepreneurs by providing advanced laboratories.

M2: Impart high quality professional training to get expertize in modern software tools and technologies to cater to the real time requirements of the Industry.

M3: Inculcate team work and lifelong learning among students with a sense of societal and ethical responsibilities.

Program Specific Outcomes (PSO's)

PSO1: Apply mathematical and scientific skills in numerous areas of Computer Science and Engineering to design and develop software-based systems.

PSO2: Acquaint module knowledge on emerging trends of the modern era in Computer Science and Engineering

PSO3: Promote novel applications that meet the needs of entrepreneur, environmental and social issues.

NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)

Program Educational Objectives (PEO's)

The graduates of the programme are able to:

PEO1: Apply the knowledge of Mathematics, Science and Engineering fundamentals to identify and solve Computer Science and Engineering problems.

PEO2: Use various software tools and technologies to solve problems related to academia, industry and society.

PEO3: Work with ethical and moral values in the multi-disciplinary teams and can communicate effectively among team members with continuous learning.

PEO4: Pursue higher studies and develop their career in software industry.

NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)

Program Outcomes

1. Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

2. Problem analysis: Identify, formulate, research literature, and analyzecomplex engineering problems reaching substantiated conclusions using firstprinciples of mathematics, natural sciences, and engineering sciences.

3. Design/development of solutions: Design solutions for complexengineering problems and design system components or processes that meetthe specified needs with appropriate consideration for the public health andsafety, and the cultural, societal, and environmental considerations.

4. Conduct investigations of complex problems: Use research-basedknowledge and research methods including design of experiments, analysis andinterpretation of data, and synthesis of the information to provide validconclusions.

5. Modern tool usage: Create, select, and apply appropriate techniques,resources, and modern engineering and IT tools including prediction andmodeling to complex engineering activities with an understanding of thelimitations.

6. The engineer and society: Apply reasoning informed by the contextualknowledge to assess societal, health, safety, legal and cultural issues and theconsequent responsibilities relevant to the professional engineering practice.

7. Environment and sustainability: Understand the impact of theprofessional engineering solutions in societal and environmental contexts,and demonstrate the knowledge of, and need for sustainable development.

8. Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

9. Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

10. Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

11. Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

12. Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

Project Course Outcomes (CO'S):

CO425.1: Analyse the System of Examinations and identify the problem.

CO425.2: Identify and classify the requirements.

CO425.3: Review the Related Literature

CO425.4: Design and Modularize the project

CO425.5: Construct, Integrate, Test and Implement the Project.

CO425.6: Prepare the project Documentation and present the Report using appropriate method.

Course Outcomes – Program Outcomes mapping

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3
C425.1		✓											✓		
C425.2	✓		✓		✓								✓		
C425.3				✓		✓	✓	✓					✓		
C425.4			✓			✓	✓	✓					✓	✓	
C425.5					✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
C425.6									✓	✓	✓		✓	✓	

Course Outcomes – Program Outcome correlation

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3
C425.1	2	3											2		
C425.2			2		3								2		
C425.3				2		2	3	3					2		
C425.4			2			1	1	2					3	2	
C425.5					3	3	3	2	3	2	2	1	3	2	1
C425.6									3	2	1		2	3	

Note: The values in the above table represent the level of correlation between CO's and PO's:

1. **Low level**
2. **Medium level**
3. **High level**

Project mapping with various courses of Curriculum with Attained PO's:

Name of the course from which principles are applied in this project	Description of the device	Attained PO
C3.2.4, C3.2.5	Gathering the requirements and defining the problem, plan to develop a Movie Recommendation System using cosine similarity	PO1, PO3
CC4.2.5	Each and every requirement is critically analyzed, the process model is identified and divided into five modules	PO2, PO3
CC4.2.5	Logical design is done by using the unified modelling language which involves individual team work	PO3, PO5, PO9
CC4.2.5	Each and every module is tested, integrated, and evaluated in our project	PO1, PO5
CC4.2.5	Documentation is done by all our four members in the form of a group	PO10
CC4.2.5	Each and every phase of the work in group is presented periodically	PO10, PO11
CC4.2.5	Implementation is done and the project will be handled by the users and in future updates in our project can be done based on preferences of user while liking a movie	PO4, PO7
CC4.2.8 CC4.2.	The physical design includes components like computer, dataset, and software	PO5, PO6

INDEX

List of Figures

1.INTRODUCTION

1.1 Introduction

In our day-to-day life, we are going to use many applications on our mobile. For example, consider the application flip kart, When are searched for an item in that application it will show similar items to your search. Even if you close that particular application it will send you a notification about similar items on your search again and again. Like this, there are many applications we are going to use in our daily life which recommends our favorite products. That means the recommendations are possible only when there is data about a user. Therefore, large organizations collect data from different users in multiple actions. Here, the data is stored in various formats. From that data, the organizations will recommend or predict what the user like most. How to recommend something? Mainly there are three ways to recommend something [5, 12].

Those are:

1. Content-Based Recommendations
2. Collaborative Recommendations
3. Hybrid Recommendations

Here, let us go with one by one.

A. Content-Based Recommendations:

Content-Based Recommendations will be made recommendations based on the user search. For example, consider the content-based movie recommendation system, If a user searches for a particular movie, the system will show the results which are most similar to the search, or if a user liked a movie, Here also it will show similar movies to the liked movie. The data used to recommend a movie is dependent on the particular user. Data is nothing but information or raw data about a user or something. Recommendations will change from person to person [6, 9]. For better understanding about Content-Based Recommendation System consider Fig 1.1.

B. Collaborative Recommendations:

Collaborative Recommendations will be made recommendations based on the similarity between the users either positively or negatively. Here, The data of multiple users are used to make recommendations. For example, consider the collaborative movie recommendation system, Let us consider User-A, User-B, and User-C. User-A liked some movies, User-B liked some movies and User-C liked some movies. If User-A and User-B like almost all similar movies, Then the movies

which is not visited by User-A and which are liked by User-B will recommend to User-A and wise versa. If User-A-liked movies are almost all negative to User-C, Therefore here it shows that the movies which are disliked by User-A are liked by User-C. Taking this into consideration the collaborative movie recommendation system will recommend the movies to the movies to User-A which are disliked by User-B. This is how the collaborative recommendations are done [7, 10]. For better understanding about Collaborative Recommendation System consider Fig 1.2.

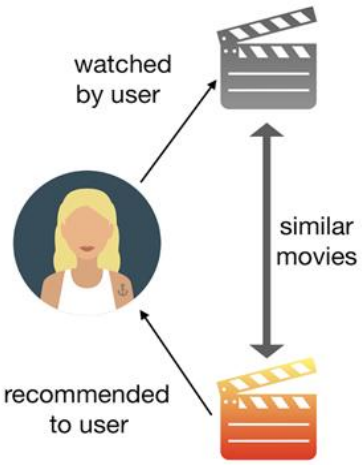

Fig 1.1: Example for Content-Based Recommendations

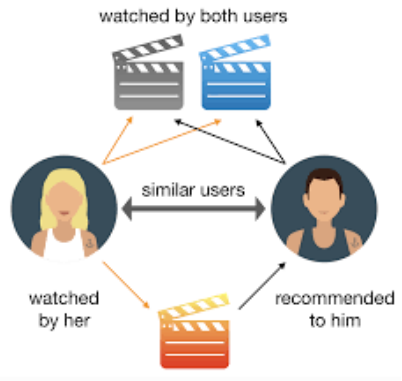

Fig 1.2: Example for Collaborative Recommendation

C. Hybrid Recommendations:

Both content-based recommendation system and collaborative recommendation systems have individual advantages and disadvantages to overcome the Hybrid Recommendation System introduced. Hybrid Recommendation is nothing but the combination of both advantages of a content-based recommendation system and a collaborative recommendation system [8, 11]. Fig 1.3 gives you the better explanation about how the Hybrid Recommendation System works.

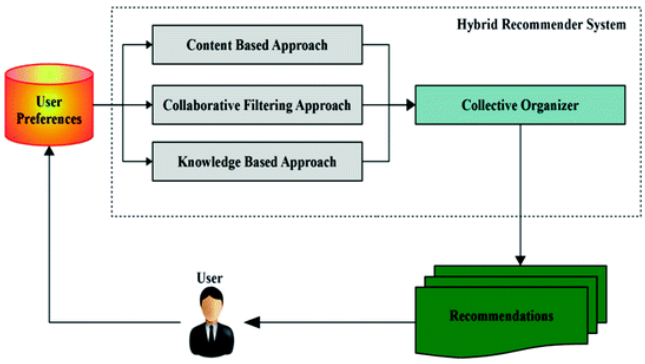

Fig 1.3: Example for Hybrid Recommendations

1.2 Existing System

Movie recommendation systems are designed to suggest movies to users based on their preferences and past movie choices. There are different types of recommendation systems, and here are some of the existing ones:

A. **Collaborative Filtering:** This type of recommendation system analyzes user behavior, such as movie ratings, and suggests similar movies to the user based on the ratings of other users who have similar tastes.

B. **Content-Based Filtering:** This type of recommendation system analyzes the attributes of the movies, such as genre, actors, and directors, and suggests movies to the user based on their preferences.

C. **Hybrid Filtering:** This type of recommendation system combines collaborative and content-based filtering to provide personalized recommendations to the user.

D. **Knowledge-Based Systems:** This type of recommendation system asks the user a series of questions to determine their preferences and suggests movies based on their answers.

E. **Demographic-Based Systems:** This type of recommendation system suggests movies to users based on their age, gender, location, and other demographic factors.

F. **Popularity-Based Systems:** This type of recommendation system suggests popular movies to users based on their ratings and box office success.

G. **Association Rules:** This type of recommendation system suggests movies to users based on the association between movies. For example, if a user likes "The Lord of the Rings," the system may suggest "The Hobbit" or "Game of Thrones."

These are some of the existing movie recommendation systems. Each system has its advantages and disadvantages.

Disadvantages:

1. Doesn’t generate accurate and efficient results.
2. It produces cost effective results

1.3 Proposed System

The proposed system is a content-based movie recommendations using cosine similarity. Content-based filtering is a type of recommendation system that analyzes the attributes of movies and suggests similar movies to the user based on their preferences. The system uses features such as genre, actors, directors, plot summary, and keywords to create a profile of the user's interests and recommend movies that are similar to the ones they have enjoyed in the past. This type of recommendation system is particularly useful for users who have specific tastes or preferences when it comes to movies. For example, if a user likes action movies with Tom Cruise, the system will suggest other action movies with Tom Cruise or movies with similar genres or actors. We can improve the performance by using similarity between the selected movie and movies present in database.

Advantages:

1. **Scalability:** Cosine similarity-based recommendation systems can handle large datasets and recommend movies in real-time. This is because cosine similarity only compares the similarity of movie attributes and does not require complex algorithms or user behavior analysis.
2. **Personalization:** Content-based movie recommendations using cosine similarity provide personalized suggestions to users based on their past viewing history and preferences. The system analyzes the attributes of movies that the user has enjoyed in the past and recommends similar movies based on cosine similarity scores.
3. **Transparency:** Cosine similarity-based recommendation systems are more transparent than other types of recommendation systems, as the system's recommendation is based on the similarity between the movie attributes and not on complex algorithms or user behavior.

1.4. System Requirements

1.4.1 Hardware Requirements:

- System type : intel®core™i5-7500UCPU@2.70gh
- Cache memory : 4 MB
- RAM : 12 GB
- Hard Disc : 8 GB

1.4.2 Software Requirements:

- Operating system : windows 10, 64 bit OS
- Coding language : Python
- Python distribution : Anaconda, Spyder, Flask

2. LITERATURE SURVEY

2.1 LITERATURE SURVEY

There are three ways to made recommendations. Making genres as one matrix, Ratings which are above “3” will noted as “1” and which are below the “3” will rated as “-1” then finding the dot product of those two matric, finally finding the Euclidian distance between the users the recommendations are made [1].

The article "Matrix Factorization Techniques for Recommender Systems" by Koren, Bell, and Volinsky discusses the use of matrix factorization techniques to improve the accuracy of recommender systems. The authors begin by explaining the challenges of building a recommender system, namely the problem of sparsity, where there are many missing values in the data matrix that represents user-item interactions [2]. Matrix factorization techniques aim to overcome this problem by decomposing the data matrix into two lower-rank matrices, which can then be used to predict missing values. The authors then describe two popular matrix factorization techniques: Singular Value Decomposition (SVD) and Alternating Least Squares (ALS). They explain the advantages and disadvantages of each technique and compare their performance on real-world datasets .

The article also discusses several extensions and modifications of matrix factorization techniques, such as incorporating side information and using regularization to prevent overfitting [2]. Finally, the authors provide some insights into the practical implementation of matrix factorization techniques, including the choice of hyperparameters, handling missing values, and dealing with large datasets. Overall, the article provides a comprehensive overview of matrix factorization techniques for recommender systems, their advantages and limitations, and practical considerations for their implementation. The authors demonstrate the effectiveness these techniques on real-world datasets and highlight the potential for further improvements and extensions [2].

Wang et al. (2015) proposed a novel deep learning approach for collaborative filtering-based recommendation systems, called Collaborative Deep Learning (CDL). The CDL model consists of two parts: a deep neural network for learning feature representations of user and item data, and a collaborative filtering component that combines these feature representations to make

recommendations. The feature representation learning component uses a stacked auto encoder to encode user and item data into low-dimensional feature vectors.

The collaborative filtering component then combines these feature vectors to estimate the rating that a user would give to an item. The authors evaluated the CDL model on two large-scale datasets: the MovieLens dataset and the Netflix dataset [3]. They compared the performance of CDL with several state-of-the-art collaborative filtering algorithms, including SVD, SVD++, and NMF. The experimental results showed that CDL outperformed these baseline algorithms on both datasets, achieving significant improvements in recommendation accuracy.

Overall, the CDL model demonstrates the potential of deep learning techniques for improving the performance of collaborative filtering-based recommendation systems, and it provides a promising direction for future research in this area. The feature representation learning component uses a stacked autoencoder to encode user and item data into low-dimensional feature vectors. The collaborative filtering component then combines these feature vectors to estimate the rating that a user would give to an item [3].

The paper "Collaborative Deep Learning for Recommender Systems" by Wang et al. proposes a novel approach for building recommender systems using deep learning techniques [4]. The authors begin by highlighting the limitations of traditional collaborative filtering techniques, such as sparsity and scalability issues. They propose a solution based on deep learning, which is a type of neural network that can learn complex representations of the input data.

The proposed approach, called Collaborative Deep Learning (CDL), consists of a three-layer neural network that takes as input user-item interaction data and outputs a predicted rating for each user-item pair. The first layer of the network represents the users, the second layer represents the items, and the third layer combines the user and item representations to generate the predicted ratings. The authors also propose an extension to the basic CDL model that incorporates auxiliary information, such as user and item attributes, to improve the accuracy of the recommendations. The paper provides experimental results on two large-scale datasets that demonstrate the superiority of the CDL approach over traditional collaborative filtering techniques and other state-of-the-art deep learning-based recommender systems [4]. The authors also perform ablation studies to evaluate the contribution of each component of the CDL model.

Overall, the paper presents a novel and effective approach for building recommender systems using deep learning techniques [4]. The proposed CDL model addresses some of the limitations of traditional collaborative filtering techniques and provides state-of-the-art performance on real-world datasets. The authors also highlight the potential for future research to extend the CDL approach to incorporate more complex user-item interactions and to handle dynamic environments.

2.2 NATURAL LANGUAGE PROCESING (NLP)

Natural language processing (NLP) refers to the branch of computer science—and more specifically, the branch of artificial intelligence or AI—concerned with giving computers the ability to understand text and spoken words in much the same way human beings can. NLP combines computational linguistics—rule-based modelling of human language—with statistical, machine learning, and deep learning models. Together, these technologies enable computers to process human language in the form of text or voice data and to 'understand' its full meaning, complete with the speaker or writer's intent and sentiment.

NLP drives computer programs that translate text from one language to another, respond to spoken commands, and summarize large volumes of text rapidly—even in real time. There's a good chance you've interacted with NLP in the form of voice-operated GPS systems, digital assistants, speech-to-text dictation software, customer service chatbots, and other consumer conveniences. But NLP also plays a growing role in enterprise solutions that help streamline business operations, increase employee productivity, and simplify mission-critical business processes.

2.3 NLP TASKS:

2.3.1 Syntactic analysis, also known as parsing or syntax analysis, identifies the syntactic structure of a text and the dependency relationships between words, represented on a diagram called a parse tree.

2.3.2 Semantic analysis focuses on identifying the meaning of language. However, since language is polysemic and ambiguous, semantics is considered one of the most challenging areas in NLP. Semantic tasks analyse the structure of sentences, word interactions, and related concepts, in an attempt to discover the meaning of words, as well as understand the topic of a text.

Below, we've listed some of the main sub-tasks of both semantic and syntactic analysis:

2.3.3 Tokenization is an essential task in natural language processing used to break up a string of words into semantically useful units called tokens. Sentence tokenization splits sentences within a text, and word tokenization splits words within a sentence. Generally, word tokens are separated by blank spaces, and sentence tokens by stops. However, you can perform high-level tokenization for more complex structures, like words that often go together, otherwise known as collocations (e.g., New York). Here's an example of how word tokenization simplifies text: Customer service couldn't be better! = "customer service" "could" "not" "be" "better".

2.3.4 Part-of-speech tagging:

Part-of-speech tagging (abbreviated as PoS tagging) involves adding a part of speech category to each token within a text. Some common PoS tags are verb, adjective, noun, pronoun, conjunction, preposition, intersection, among others. In this case, the example above would look like this: "Customer service": NOUN, "could": VERB, "not": ADVERB, be": VERB, "better": ADJECTIVE, "!": PUNCTUATION.

PoS tagging is useful for identifying relationships between words and, therefore, understand the meaning of sentences.

2.3.5 Dependency Parsing:

Dependency grammar refers to the way the words in a sentence are connected Fig 2.1 givies the clear view of dependency parsing. A dependency parser, therefore, analyzes how 'head words' are related and modified by other words to understand the syntactic structure of a sentence.

2.3.6 Constituency Parsing:

Constituency Parsing aims to visualize the entire syntactic structure of a sentence by identifying phrase structure grammar. It consists of using abstract terminal and non-terminal nodes associated to words, as shown in below Fig 2.2.

You can try different parsing algorithms and strategies depending on the nature of the text you intend to analyze, and the level of complexity you'd like to achieve.

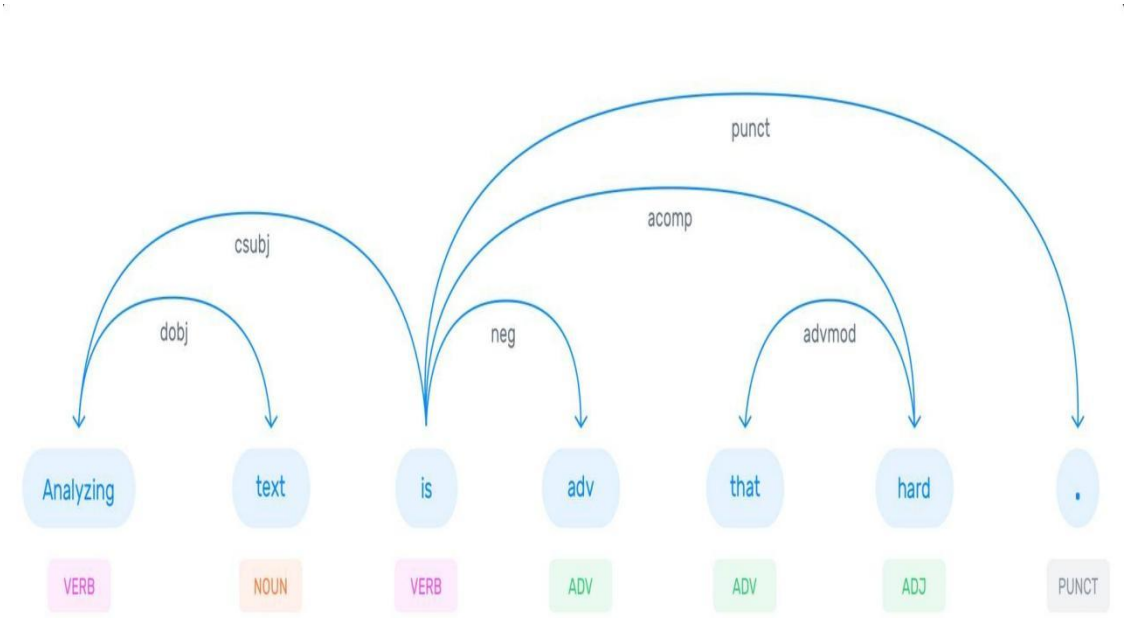

Fig 2.1: Dependency Parsing

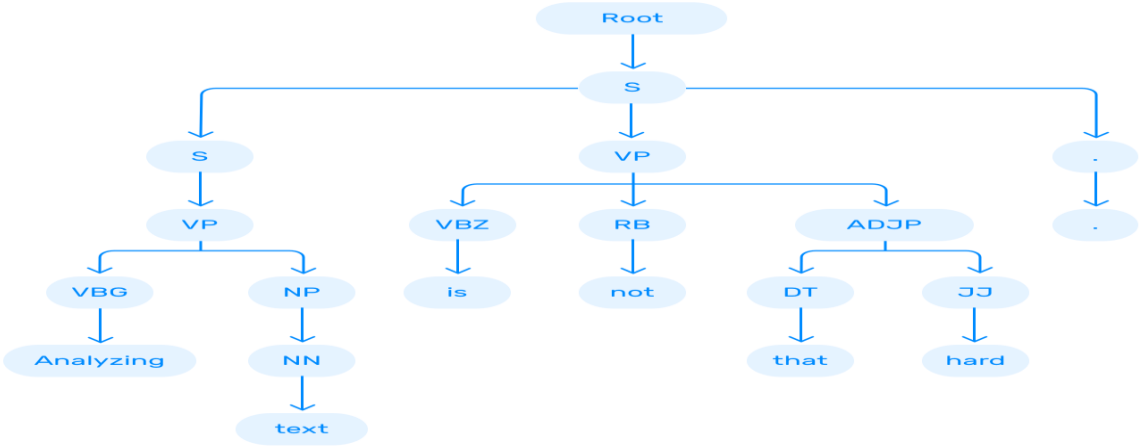

Fig 2.2: Constituency Parsing

2.3.7 Lemmatization & Stemming:

When we speak or write, we tend to use inflected forms of a word (words in their different grammatical forms). To make these words easier for computers to understand, NLP uses lemmatization and stemming to transform them back to their root form. The word as it appears in the dictionary – its root form – is called a lemma. For example, the terms "is, are, am, were, and been," are grouped under the lemma 'be.' So, if we apply this lemmatization to "African elephants

have four nails on their front feet," the result will look something like this: African elephants have four nails on their front feet = "African," "elephant," "have," "4", "nail," "on," "their," "foot"]. This example is useful to see how the lemmatization changes the sentence using its base form (e.g., the word "feet"" was changed to "foot").

When we refer to stemming, the root form of a word is called a stem. Stemming "trims" words, so word stems may not always be semantically correct. For example, stemming the words "consult," "consultant," "consulting," and "consultants" would result in the root form "consult''.While lemmatization is dictionary-based and chooses the appropriate lemma based on context, stemming operates on single words without considering the context. For example, in the sentence: "This is better"

The word "better" is transformed into the word "good" by a lemmatizer but is unchanged by stemming. Even though stemmers can lead to less-accurate results, they are easier to build and perform faster than lemmatizers. But lemmatizers are recommended if you're seeking more precise linguistic rules.

2.3.8 Stopword Removal:

Removing stop words is an essential step in NLP text processing. It involves filtering out high-frequency words that add little or no semantic value to a sentence, for example, which, to, at, for, is, etc. You can even customize lists of stopwords to include words that you want to ignore.

Let's say you want to classify customer service tickets based on their topics. In this example: "Hello, I'm having trouble logging in with my new password", it may be useful to remove stop words like "hello", "I", "am", "with", "my", so you're left with the words that help you understand the topic of the ticket: "trouble", "logging in", "new", "password".

2.3.9 Word Sense Disambiguation:

Depending on their context, words can have different meanings. Take the word "book", for example:

- You should read this book; it's a great novel!
- You should book the flights as soon as possible.
- You should close the books by the end of the year.
- You should do everything by the book to avoid potential complications.

There are two main techniques that can be used for word sense disambiguation (WSD): knowledge-based (or dictionary approach) or supervised approach. The first one tries to infer meaning by observing the dictionary definitions of ambiguous terms within a text, while the latter is based on natural language processing algorithms that learn from training data.

2.3.10 Named Entity Recognition (NER):

Named entity recognition is one of the most popular tasks in semantic analysis and involves extracting entities from within a text. Entities can be names, places, organizations, email addresses, and more. Relationship extraction, another sub-task of NLP, goes one step further and finds relationships between two nouns. For example, in the phrase "Susan lives in Los Angeles," a person (Susan) is related to a place (Los Angeles) by the semantic category "lives in."

2.3.11 Text Classification:

Text classification is the process of understanding the meaning of unstructured text and organizing it into predefined categories (tags). One of the most popular text classification tasks is sentiment analysis, which aims to categorize unstructured data by sentiment.

2.4 Common Examples of NLP

1. Email filters
2. Virtual assistants, voice assistants, or smart speakers
3. Online search engines
4. Predictive text and autocorrect
5. Monitor brand sentiment on social media
6. Sorting customer feedback
7. Automating processes in customer support
8. Chatbots
9. Automatic summarization
10. Machine translation
11. Natural language generation

3. ARCHITECTURE DIAGRAM

Content-based movie recommendation system is a type of recommendation system that suggests movies to users based on the attributes of movies that the user has enjoyed in the past. The system analyzes the content of movies, such as genre, actors, directors, plot summary, and keywords, and creates a profile of the user's interests. The system then recommends movies to the user that are similar to the ones they have enjoyed in the past.

Content-based movie recommendation systems use different techniques to match the user's preferences with the attributes of movies. One popular technique is cosine similarity, which measures the similarity between two movies based on their attributes. The system calculates the cosine similarity score between the movies and recommends the movies with the highest similarity score. Fig 3.1 gives the clear view of content-based filtering of movies.

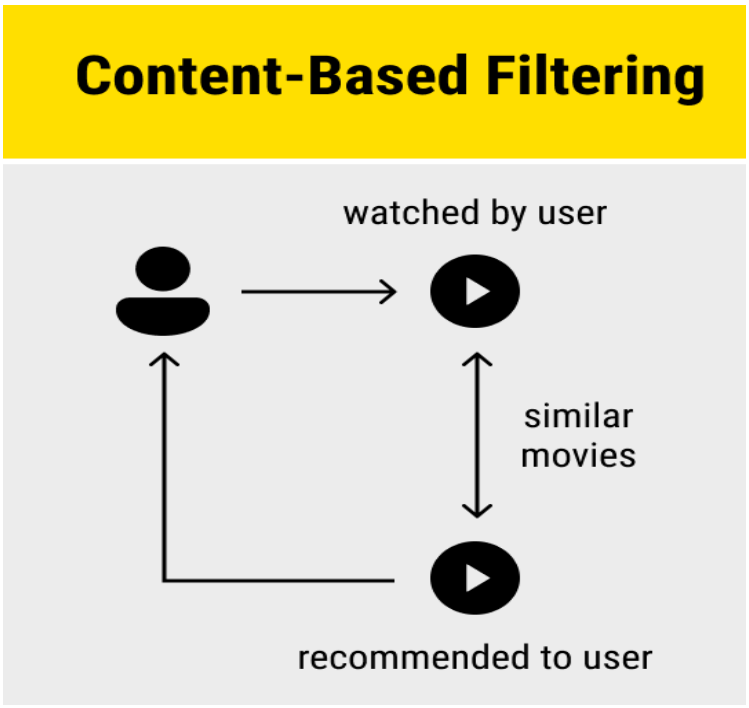

Fig 3.1: Architecture diagram

Content-based movie recommendation systems have several advantages over other types of recommendation systems. They provide personalized recommendations to users, are transparent and interpretable, and can handle new users or items without any prior data.

3.1. Prevalence of Movie Recommendation

The movie recommendation model proposed in this work enables users to expect the most similar movie predictions with respective to their interest and organizations gather the interests of different users. The data gathered is stored in the form of CSV format along with its features. This will allow us to use multiple features as input parameters according to the user preferences. The Design mainly consists of 2 stages i.e. the initial and final stage. The initial stage mainly includes the data analysis and pre-processing. The final stage includes the recommendations of the model.

3.2. Importance of ML in Movie Recommendation

Machine Learning allows an organization on forecasting the interests of different users. Machine Learning provides multiple number of models to generalize it to the unseen data from the same population and measuring the performance. By implementing such models using machine learning in an application, The users get attracted to the particular application. Along with meeting the organization objective, the model should take care of accuracy, execution time, complexity and scalability as well to be considered as best model.

3.3. Implementation of machine learning using Python

Python is a popular programming language. It was created in 1991 by Guido van Rossum.
It is used for:

1. web development (server-side),
2. software development,
3. mathematics,
4. system scripting.

The most recent major version of Python is Python 3. However, Python 2, although not being updated with anything other than security updates, is still quite popular. It is possible to write Python in an Integrated Development Environment, such as Thonny, Pycharm, Netbeans or Eclipse, Anaconda which are particularly useful when managing larger collections of Python files.

Python was designed for its readability. Python uses new lines to complete a command, as opposed to other programming languages which often use semicolons or parentheses. Python relies on indentation, using whitespace, to define scope; such as the scope of loops, functions and classes. Other programming languages often use curly-brackets for this purpose.

In the older days, people used to perform Machine Learning tasks manually by coding all the algorithms and mathematical and statistical formula. This made the process time consuming, tedious and inefficient. But in the modern days, it is become very much easy and efficient compared to the olden days by various python libraries, frameworks, and modules. Today, Python is one of the most popular programming languages for this task and it has replaced many languages in the industry, one of the reason is its vast collection of libraries. Python libraries that used in Machine Learning are:

1. Numpy
2. Scipy
3. Scikit-learn
4. Pandas
5. Matplotlib

NumPy is a very popular python library for large multi-dimensional array and matrix processing, with the help of a large collection of high-level mathematical functions. It is very useful for fundamental scientific computations in Machine Learning. It is particularly useful for linear algebra, Fourier transform, and random number capabilities. High-end libraries like TensorFlow uses NumPy internally for manipulation of Tensors.

SciPy is a very popular library among Machine Learning enthusiasts as it contains different modules for optimization, linear algebra, integration and statistics. There is a difference between the SciPy library and the SciPy stack. The SciPy is one of the core packages that make up the SciPy stack. SciPy is also very useful for image manipulation.

Skikit-learn is one of the most popular Machine Learning libraries for classical Machine Learning algorithms. It is built on top of two basic Python libraries, NumPy and SciPy. Scikit-learn supports most of the supervised and unsupervised learning algorithms. Scikit learn can also be used for data-mining and data-analysis, which makes it a great tool who is starting out with Machine Learning.

Pandas is a popular Python library for data analysis. It is not directly related to Machine Learning. As we know that the dataset must be prepared before training. In this case, Pandas comes handy as it was developed specifically for data extraction and preparation. It provides high-level data

structures and wide variety tools for data analysis. It provides many inbuilt methods for groping, combining and filtering data.

Matpoltlib is a very popular Python library for data visualization. Like Pandas, it is not directly related to Machine Learning. It particularly comes in handy when a programmer wants to visualize the patterns in the data. It is a 2D plotting library used for creating 2D graphs and plots. A module named pyplot makes it easy for programmers for plotting as it provides features to control line styles, font properties, formatting axes, etc. It provides various kinds of graphs and plots for data visualization, histogram, error charts, bar chats, etc.

3.4. Scope of the project

Movie Recommendation System can help to determine the movies which will like to user. There are various factors that influence the interests of the user which include genre of the movie, keywords used in the movie, cast and crew of the movie as well as overview of the movie. This project involves the collecting the user data, analysing it and developing a recommendation system algorithm.

3.5 Analysis

The Fig 3.2 is the dataset contains 7 attributes which are used to predict the movie recommendation system such as

Id: Which is unique to each and every movie.

Title: Which denotes the title of the movie.

Overview: Overview describes the brief summery about the movie.Fig 3.11 gives you the type of information and structure of information present in genres attribute before pre-processing. Fig 3.12 gives you the type of information and structure of information present in genres attribute after pre-processing.

Genres: Genres attribute describes the genre of the movie. Fig 3.9 gives you the type of information and structure of information present in genres attribute before pre-processing. Fig 3.10 gives you the type of information and structure of informationpresent in genres attribute after pre-processing.

Cast: Cast attribute consists the actors and actresses who appear in the movie. Fig 3.5 gives you the type of information and structure of information present in cast attribute before

pre-processing and Fig 3.6 gives you the type of information and structure of information present in cast attribute after pre-processing.

Crew: The Crew attribute includes the directors, producers, writers and people behind the movie. Fig 3.7 gives you the type of information and structure of information present in crew attribute before pre-processing and Fig 3.8 gives you the type of information and structure of information present in crew attribute after pre-processing.

Keywords: Keywords such as theme, setting or notable feature are included in the keywords attribute.

DataSet:

	id	title	overview	genres	cast	crew	keywords
0	862	Toy Story	Led by Woody, Andy	[{'id': 16, 'nam	[{'cast_id': 14, 'ch	[{'credit_id': '52fe4284	[{'id': 931, 'name': 'jealousy'}, {'i
1	8844	Jumanji	When siblings Judy a	[{'id': 12, 'nam	[{'cast_id': 1, 'cha	[{'credit_id': '52fe44bf	[{'id': 10090, 'name': 'board gam
2	15602	Grumpier Old Men	A family wedding re	[{'id': 10749, 'r	[{'cast_id': 2, 'cha	[{'credit_id': '52fe466a	[{'id': 1495, 'name': 'fishing'}, {'ic
3	31357	Waiting to Exhale	Cheated on, mistrea	[{'id': 35, 'nam	[{'cast_id': 1, 'cha	[{'credit_id': '52fe4477	[{'id': 818, 'name': 'based on nov
4	11862	Father of the Bride Part II	Just when George B	[{'id': 35, 'nam	[{'cast_id': 1, 'cha	[{'credit_id': '52fe4495	[{'id': 1009, 'name': 'baby'}, {'id':
5	949	Heat	Obsessive master th	[{'id': 28, 'nam	[{'cast_id': 25, 'ch	[{'credit_id': '52fe4292	[{'id': 642, 'name': 'robbery'}, {'ic
6	11860	Sabrina	An ugly duckling hav	[{'id': 35, 'nam	[{'cast_id': 1, 'cha	[{'credit_id': '52fe4495	[{'id': 90, 'name': 'paris'}, {'id': 38
7	45325	Tom and Huck	A mischievous youn	[{'id': 28, 'nam	[{'cast_id': 2, 'cha	[{'credit_id': '52fe46bd	[]
8	9091	Sudden Death	International action	[{'id': 28, 'nam	[{'cast_id': 1, 'cha	[{'credit_id': '52fe44db	[{'id': 949, 'name': 'terrorist'}, {'i
9	710	GoldenEye	James Bond must ur	[{'id': 12, 'nam	[{'cast_id': 1, 'cha	[{'credit_id': '52fe426e	[{'id': 701, 'name': 'cuba'}, {'id': 7
10	9087	The American President	Widowed U.S. presi	[{'id': 35, 'nam	[{'cast_id': 1, 'cha	[{'credit_id': '52fe44da	[{'id': 833, 'name': 'white house'
11	12110	Dracula: Dead and Loving It	When a lawyer shov	[{'id': 35, 'nam	[{'cast_id': 9, 'cha	[{'credit_id': '52fe44b7	[{'id': 3633, 'name': 'dracula'}, {'i
12	21032	Balto	An outcast half-wol	[{'id': 10751, 'r	[{'cast_id': 1, 'cha	[{'credit_id': '593f24b9	[{'id': 1994, 'name': 'wolf'}, {'id':
13	10858	Nixon	An all-star cast pow	[{'id': 36, 'nam	[{'cast_id': 1, 'cha	[{'credit_id': '52fe43c5	[{'id': 840, 'name': 'usa president
14	1408	Cutthroat Island	Morgan Adams and	[{'id': 28, 'nam	[{'cast_id': 1, 'cha	[{'credit_id': '52fe42f4	[{'id': 911, 'name': 'exotic island'
15	524	Casino	The life of the gamb	[{'id': 18, 'nam	[{'cast_id': 4, 'cha	[{'credit_id': '52fe424d	[{'id': 383, 'name': 'poker'}, {'id':
16	4584	Sense and Sensibility	Rich Mr. Dashwood	[{'id': 18, 'nam	[{'cast_id': 6, 'cha	[{'credit_id': '52fe43ce	[{'id': 420, 'name': 'bowling'}, {'ic
17	5	Four Rooms	It's Ted the Bellhop'	[{'id': 80, 'nam	[{'cast_id': 42, 'ch	[{'credit_id': '52fe420d	[{'id': 612, 'name': 'hotel'}, {'id':
18	9273	Ace Ventura: When Nature Calls	Summoned from an	[{'id': 80, 'nam	[{'cast_id': 1, 'cha	[{'credit_id': '52fe44df	[{'id': 409, 'name': 'africa'}, {'id':
19	11517	Money Train	A vengeful New Yor	[{'id': 28, 'nam	[{'cast_id': 1, 'cha	[{'credit_id': '52fe4450	[{'id': 380, 'name': 'brother brotl
20	8012	Get Shorty	Chili Palmer is a Mia	[{'id': 35, 'nam	[{'cast_id': 1, 'cha	[{'credit_id': '52fe448d	[{'id': 395, 'name': 'gambling'}, {'

Fig 3.2 Dataset before pre-processing

3.6 Data Pre-processing

Before feeding data to an algorithm we have to apply transformations to our data which is referred as pre-processing. By performing pre-processing the raw data which is not feasible for analysis is converted into clean data. In-order to achieve better results using a model in Machine Learning, data format has to be in a proper manner. The data should be in a particular format for

different algorithms. For example, if we consider Random Forest algorithm it does not support null values. So that those null values have to be managed using raw data.

Pre-processing refers to the transformations applied to our data before feeding it to the algorithm. Data Pre-processing is a technique that is used to convert the raw data into a clean data set. In other words, whenever the data is gathered from different sources it is collected in raw format which is not feasible for the analysis. Fig 3.3 shows the flow of data pre-processing.

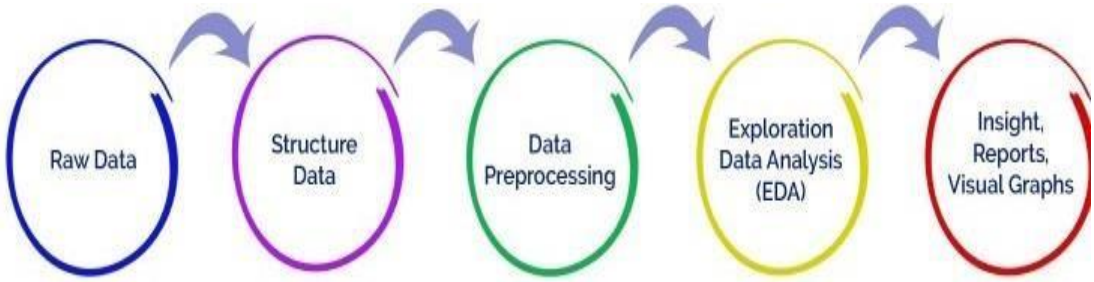

Fig:3.3 Data Pre-processing

Need of Data Preprocessing: For achieving better results from the applied model in Machine Learning projects the format of the data has to be in a proper manner. Some specified Machine Learning model needs information in a specified format. For example, Random Forest algorithm does not support null values, therefore to execute random forest algorithm null values have to be managed from the original raw data set. Another aspect is that data set should be formatted in such a way that more than one Machine Learning and Deep Learning algorithms are executed in one data set, and best out of them is chosen.

3.6.1 Missing values

Filling missing values is one of the pre-processing techniques. The missing values in the dataset is represented as '?' but it a non-standard missing value and it has to be converted into a standard missing value NaN. So that pandas can detect the missing values. We are dropping that missing values. The Fig 3.4 shows the dropping of null values in the taken daataset.

```
In [14]: movies.isnull().sum()

Out[14]: id            0
         title         6
         overview    954
         genres        0
         dtype: int64

In [15]: movies.dropna(inplace = True )

In [16]: movies.isnull().sum()

Out[16]: id          0
         title       0
         overview    0
         genres      0
         dtype: int64
```

Fig 3.4: Screenshot of removing null values

```
In [39]: movies['cast'][0]

Out[39]: "[{'cast_id': 14, 'character': 'Woody (voice)', 'credit_id': '52fe4284c3a36847f
         8024f95', 'gender': 2, 'id': 31, 'name': 'Tom Hanks', 'order': 0, 'profile_pat
         h': '/pQFoyx7rp09CJTAb932F2g8Nlho.jpg'}, {'cast_id': 15, 'character': 'Buzz Lig
         htyear (voice)', 'credit_id': '52fe4284c3a36847f8024f99', 'gender': 2, 'id': 12
         898, 'name': 'Tim Allen', 'order': 1, 'profile_path': '/uX2xVf6pMmPepxnvFWyBtje
         xzgY.jpg'}, {'cast_id': 16, 'character': 'Mr. Potato Head (voice)', 'credit_i
         d': '52fe4284c3a36847f8024f9d', 'gender': 2, 'id': 7167, 'name': 'Don Rickles',
         'order': 2, 'profile_path': '/h5BcaDMPRVLHLDzbQavec4xfSdt.jpg'}, {'cast_id': 1
         7, 'character': 'Slinky Dog (voice)', 'credit_id': '52fe4284c3a36847f8024fa1',
         'gender': 2, 'id': 12899, 'name': 'Jim Varney', 'order': 3, 'profile_path': '/e
         Io2jVVXYgjDtaHoF19Ll9vtW7h.jpg'}, {'cast_id': 18, 'character': 'Rex (voice)',
         'credit_id': '52fe4284c3a36847f8024fa5', 'gender': 2, 'id': 12900, 'name': 'Wal
         lace Shawn', 'order': 4, 'profile_path': '/oGE6JqPP2xH4tNORKNqxbNPYi7u.jpg'},
         {'cast_id': 19, 'character': 'Hamm (voice)', 'credit_id': '52fe4284c3a36847f802
         4fa9', 'gender': 2, 'id': 7907, 'name': 'John Ratzenberger', 'order': 5, 'profi
         le_path': '/yGechiKWL6TJDfVE2KPSJYqdMsY.jpg'}, {'cast_id': 20, 'character': 'Bo
         Peep (voice)', 'credit_id': '52fe4284c3a36847f8024fad', 'gender': 1, 'id': 887
```

Fig 3.5: Screenshot of cast attribute before pre-processing

```
In [41]: movies['cast'].apply(convertcast)

Out[41]: 0                     [Tom Hanks, Tim Allen, Don Rickles]
         1           [Robin Williams, Jonathan Hyde, Kirsten Dunst]
         2               [Walter Matthau, Jack Lemmon, Ann-Margret]
         3         [Whitney Houston, Angela Bassett, Loretta Devine]
         4                [Steve Martin, Diane Keaton, Martin Short]
                                       ...
         45624          [Leila Hatami, Kourosh Tahami, Elham Korda]
         45625           [Angel Aquino, Perry Dizon, Hazel Orencio]
         45626         [Erika Eleniak, Adam Baldwin, Julie du Page]
         45627    [Iwan Mosschuchin, Nathalie Lissenko, Pavel Pa...
         45628                                                   []
         Name: cast, Length: 44482, dtype: object
```

Fig 3.6: Screenshot of cast attribute after pre-processing

```
In [42]: movies['crew'][0]

Out[42]: '[{\'credit_id\': \'52fe4284c3a36847f8024f49\', \'department\': \'Directin
         g\', \'gender\': 2, \'id\': 7879, \'job\': \'Director\', \'name\': \'John
         Lasseter\', \'profile_path\': \'/7EdqiNbr4FRjIhKHyPPdFfEEEFG.jpg\'}, {\'cr
         edit_id\': \'52fe4284c3a36847f8024f4f\', \'department\': \'Writing\', \'ge
         nder\': 2, \'id\': 12891, \'job\': \'Screenplay\', \'name\': \'Joss Whedon
         \', \'profile_path\': \'/dTiVsuaTVTeGmvkhcyJvKp2A5kr.jpg\'}, {\'credit_id
         \': \'52fe4284c3a36847f8024f55\', \'department\': \'Writing\', \'gender\':
         2, \'id\': 7, \'job\': \'Screenplay\', \'name\': \'Andrew Stanton\', \'pro
         file_path\': \'/pvQWsu0qc8JFQhMVJkTHuexUAa1.jpg\'}, {\'credit_id\': \'52fe
         4284c3a36847f8024f5b\', \'department\': \'Writing\', \'gender\': 2, \'id
         \': 12892, \'job\': \'Screenplay\', \'name\': \'Joel Cohen\', \'profile_pa
         th\': \'/dAubAiZcvKFbboWlj7oXOkZnTSu.jpg\'}, {\'credit_id\': \'52fe4284c3a
         36847f8024f61\', \'department\': \'Writing\', \'gender\': 0, \'id\': 1289
         3, \'job\': \'Screenplay\', \'name\': \'Alec Sokolow\', \'profile_path\':
         \'/v79vlRYi94BZUQnkkyznbGUZLjT.jpg\'}, {\'credit_id\': \'52fe4284c3a36847f
         8024f67\', \'department\': \'Production\', \'gender\': 1, \'id\': 12894,
```

Fig 3.7: Screenshot of crew attribute before pre-processing

```
In [46]: movies['crew'].apply(convertcrew)

Out[46]: 0           [John Lasseter]
         1            [Joe Johnston]
         2           [Howard Deutch]
         3         [Forest Whitaker]
         4           [Charles Shyer]
                          ...
         45624    [Hamid Nematollah]
         45625            [Lav Diaz]
         45626      [Mark L. Lester]
         45627    [Yakov Protazanov]
         45628       [Daisy Asquith]
         Name: crew, Length: 44482, dtype: object
```

Fig 3.8: Screenshot of crew attribute after pre-processing

```
In [29]: movies.iloc[0].genres

Out[29]: "[{'id': 16, 'name': 'Animation'}, {'id': 35, 'name': 'Comedy'}, {'id': 107
         51, 'name': 'Family'}]"
```

Fig 3.9: Screenshot of genres attribute before pre-processing

```
In [33]: movies['genres'].apply(convert)

Out[33]: 0         [Animation, Comedy, Family]
         1       [Adventure, Fantasy, Family]
         2                 [Romance, Comedy]
         3          [Comedy, Drama, Romance]
         4                        [Comedy]
                           ...
         45624               [Drama, Family]
         45625                       [Drama]
         45626      [Action, Drama, Thriller]
         45627                            []
         45628                            []
         Name: genres, Length: 44482, dtype: object
```

Fig 3.10: Screenshot of genres attribute after pre-processing

```
In [36]: movies.iloc[0].keywords

Out[36]: "[{'id': 931, 'name': 'jealousy'}, {'id': 4290, 'name': 'toy'}, {'id': 5202, 'n
         ame': 'boy'}, {'id': 6054, 'name': 'friendship'}, {'id': 9713, 'name': 'friend
         s'}, {'id': 9823, 'name': 'rivalry'}, {'id': 165503, 'name': 'boy next door'},
         {'id': 170722, 'name': 'new toy'}, {'id': 187065, 'name': 'toy comes to lif
         e'}]"
```

Fig 3.11: Screenshot of keywords attribute before pre-processing

```
In [34]: movies['keywords'].apply(convert)

Out[34]: 0        [jealousy, toy, boy, friendship, friends, riva...
         1        [board game, disappearance, based on children'...
         2        [fishing, best friend, duringcreditsstinger, o...
         3        [based on novel, interracial relationship, sin...
         4        [baby, midlife crisis, confidence, aging, daug...
                                      ...
         45624                                        [tragic love]
         45625                                [artist, play, pinoy]
         45626                                                   []
         45627                                                   []
         45628                                                   []
         Name: keywords, Length: 44482, dtype: object
```

Fig 3.12: Screenshot of keywords attribute after pre-processing

After completion of data pre-processing the data set looks like of Fig 3.13. Consider Fig 3.13 for better understanding.

id	title	overview	genres	cast	crew	keywords
862	Toy Story	['Led', 'by', 'Woody,', "Andy'	['Animation', 'Comedy', 'Fam	['TomHanks', 'TimAlle	['JohnLasseter']	['jealousy', 'toy', 'boy', 'friends
8844	Jumanji	['When', 'siblings', 'Judy', 'a	['Adventure', 'Fantasy', 'Fami	['RobinWilliams', 'Jona	['JoeJohnston']	['boardgame', 'disappearance',
15602	Grumpier C	['A', 'family', 'wedding', 'rei	['Romance', 'Comedy']	['WalterMatthau', 'Jack	['HowardDeutch']	['fishing', 'bestfriend', 'duringc
31357	Waiting to	['Cheated', 'on,', 'mistreate	['Comedy', 'Drama', 'Romanc	['WhitneyHouston', 'A	['ForestWhitaker']	['basedonnovel', 'interracialrel
11862	Father of th	['Just', 'when', 'George', 'Ba	['Comedy']	['SteveMartin', 'Diane	['CharlesShyer']	['baby', 'midlifecrisis', 'confide
949	Heat	['Obsessive', 'master', 'thief	['Action', 'Crime', 'Drama', 'Th	['AlPacino', 'RobertDe	['MichaelMann']	['robbery', 'detective', 'bank', '
11860	Sabrina	['An', 'ugly', 'duckling', 'havi	['Comedy', 'Romance']	['HarrisonFord', 'JuliaC	['SydneyPollack']	['paris', 'brotherbrotherrelatio
45325	Tom and Hu	['A', 'mischievous', 'young',	['Action', 'Adventure', 'Drama	['JonathanTaylorThom	['PeterHewitt']	[]
9091	Sudden De	['International', 'action', 'su	['Action', 'Adventure', 'Thrille	['Jean-ClaudeVanDam	['PeterHyams']	['terrorist', 'hostage', 'explosiv
710	GoldenEye	['James', 'Bond', 'must', 'unr	['Adventure', 'Action', 'Thrille	['PierceBrosnan', 'Sear	['MartinCampbell']	['cuba', 'falselyaccused', 'secre
9087	The Americ	['Widowed', 'U.S.', 'presider	['Comedy', 'Drama', 'Romanc	['MichaelDouglas', 'An	['RobReiner']	['whitehouse', 'usapresident',
12110	Dracula: De	['When', 'a', 'lawyer', 'shows	['Comedy', 'Horror']	['LeslieNielsen', 'MelB	['MelBrooks']	['dracula', 'spoof']
21032	Balto	['An', 'outcast', 'half-wolf', '	['Family', 'Animation', 'Adver	['KevinBacon', 'BobHo	['SimonWells']	['wolf', 'dog-sleddingrace', 'ala
10858	Nixon	['An', 'all-star', 'cast', 'power	['History', 'Drama']	['AnthonyHopkins', 'Jo	['OliverStone']	['usapresident', 'presidentialel
1408	Cutthroat Is	['Morgan', 'Adams', 'and', 'h	['Action', 'Adventure']	['GeenaDavis', 'Matthe	['RennyHarlin']	['exoticisland', 'treasure', 'map
524	Casino	['The', 'life', 'of', 'the', 'gamb	['Drama', 'Crime']	['RobertDeNiro', 'Shar	['MartinScorsese']	['poker', 'drugabuse', '1970s', '
4584	Sense and S	['Rich', 'Mr.', 'Dashwood', 'di	['Drama', 'Romance']	['KateWinslet', 'Emma	['AngLee']	['bowling', 'basedonnovel', 'se
5	Four Room	["It's", 'Ted', 'the', "Bellhop'	['Crime', 'Comedy']	['TimRoth', 'AntonioBa	['AllisonAnders']	['hotel', "newyear'seve", 'witc
9273	Ace Ventur	['Summoned', 'from', 'an', 'a	['Crime', 'Comedy', 'Adventu	['JimCarrey', 'IanMcNe	['SteveOedekerk']	['africa', 'indigenous', 'humana
11517	Money Trai	['A', 'vengeful', 'New', 'York'	['Action', 'Comedy', 'Crime']	['WesleySnipes', 'Woo	['JosephRuben']	['brotherbrotherrelationship',
8012	Get Shorty	['Chili', 'Palmer', 'is', 'a', 'Mia	['Comedy', 'Thriller', 'Crime']	['JohnTravolta', 'Genel	['BarrySonnenfeld'	['gambling', 'miami', 'basedonr
1710	Copycat	['An', 'agoraphobic', 'psycho	['Drama', 'Thriller']	['SigourneyWeaver', 'H	['JonAmiel']	['policebrutality', 'psychology',
9691	Assassins	['Assassin', 'Robert', 'Rath', '	['Action', 'Adventure', 'Crime	['SylvesterStallone', 'A	['RichardDonner']	['competition', 'assassination',
12665	Powder	['Harassed', 'by', 'classmates	['Drama', 'Fantasy', 'ScienceF	['MarySteenburgen', 'S	['VictorSalva']	['deadanimal', 'shottodeath', 't
451	Leaving Las	['Ben', 'Sanderson,', 'an', 'al	['Drama', 'Romance']	['NicolasCage', 'Elisabe	['MikeFiggis']	['individual', 'prostitute', 'alcol

Fig:3.13 Dataset after pre-processing

3.7 Cross Validation:

Cross-validation is a technique in which we train our model using the subset of the dataset and then evaluate using the complementary subset of the data-set. The two steps involved in cross-validation are as follows :

- Recommending movies from the model.
- Displaying the similarity to each and every movie present in dataset.
- Comparing the recommended movies with similarity of the movies.

Fig 3.14 is the screenshot of recommendations of a selected movie or given movie and it is cross checked by displaying the similarity. The Fig 3.15 shows the similarity between the movies.

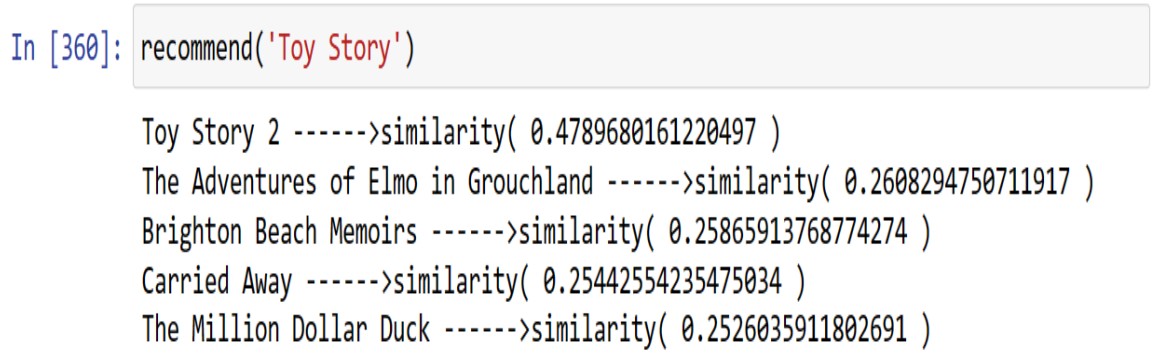

Fig:3.14: Movies recommended by implemented model

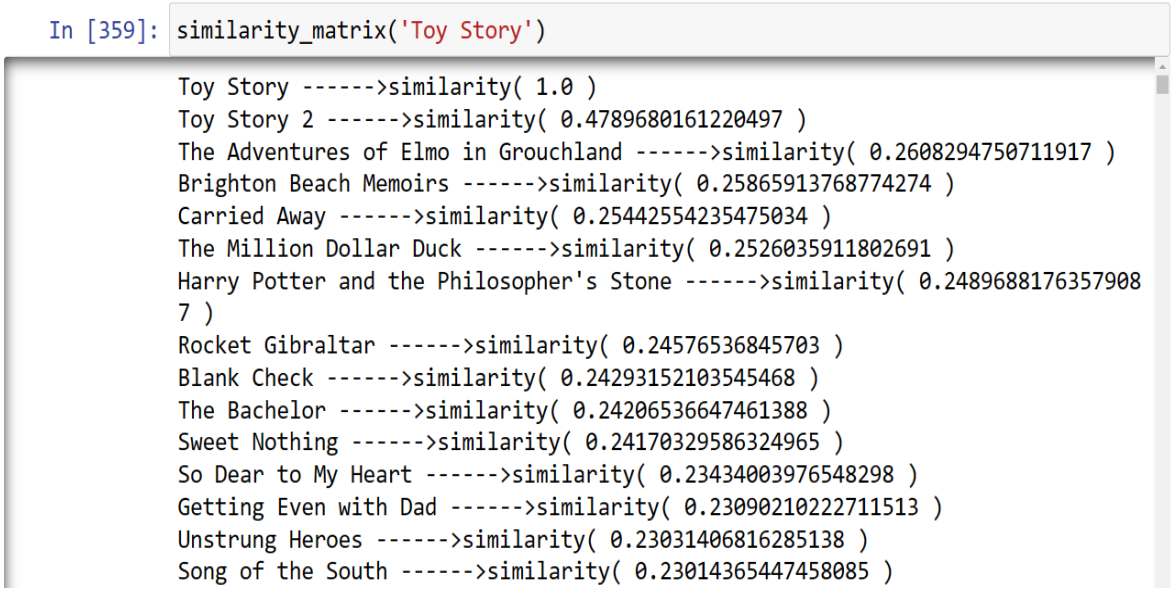

Fig:3.15: Printing the similarity for every movie in the dataset

4. IMPLEMENTATION

```
import numpy as np
import pandas as pd
movies = pd.read_csv('movies_metadata.csv')
credits = pd.read_csv('credits.csv')
keywords = pd.read_csv('keywords.csv')
links = pd.read_csv('links.csv')
ratings_small = pd.read_csv('ratings_small.csv')
links_small = pd.read_csv('links_small.csv')
movies.info()
credits.info()
keywords.info()
links.info()
links_small.info()
ratings_small.info()
ratings_small.head()
movies = movies[['id','title','overview','genres']]
movies.head()
movies.info()
movies['id'].values
movies.isnull().sum()
movies.dropna(inplace = True )
movies.isnull().sum()
movies['id'] = movies['id'].astype(int)
movies = movies.merge(credits,on='id')
movies.info()
movies = movies.merge(keywords,on='id')
movies.info()
movies.isnull().sum()
movies.duplicated().sum()
```

```
movies.drop_duplicates(inplace=True)
movies.duplicated().sum()
movies.info()
movies.head()
movies.to_csv("dataset.csv")
movies.iloc[0].genres
import ast
def convert(obj):
    L = []
    for i in ast.literal_eval(obj):
        L.append(i['name'])
    return L
movies['genres'].apply(convert)
movies['genres'] = movies['genres'].apply(convert)
movies.head(21)
movies.iloc[0].keywords
movies['keywords'] = movies['keywords'].apply(convert)
movies.head()
movies['cast'][0]
def convertcast(obj):
    L = []
    counter = 0
    for i in ast.literal_eval(obj):
        if counter != 3:
            L.append(i['name'])
            counter+=1
        else:
            break
    return L
movies['cast'].apply(convertcast)
movies['cast'] = movies['cast'].apply(convertcast)
```

```
movies.head()
movies['crew'][0]
def convertcrew(obj):
    L = []
    for i in ast.literal_eval(obj):
        if i['job'] =='Director':
            L.append(i['name'])
            break
    return L
movies['crew'].apply(convertcrew)
movies['crew'] = movies['crew'].apply(convertcrew)
movies.head()
movies['overview'][0]
movies['overview'].apply(lambda x:x.split())
movies['overview'] = movies['overview'].apply(lambda x:x.split())
movies.head()
movies['genres'].apply(lambda x:[i.replace(" ","") for i in x])
movies['genres'] = movies['genres'].apply(lambda x:[i.replace(" ","") for i in x])
movies.head()
movies['keywords'] = movies['keywords'].apply(lambda x:[i.replace(" ","") for i in x])
movies['cast'] = movies['cast'].apply(lambda x:[i.replace(" ","") for i in x])
movies['crew'] = movies['crew'].apply(lambda x:[i.replace(" ","") for i in x])
movies.head()
movies['details'] = movies['overview'] + movies['genres'] + movies['keywords'] + movies['cast'] +
movies['crew']
movies.head()
movies.to_csv("finaldataset.csv")
new_df = movies[['id','title','details']]
new_df
new_df['details'].apply(lambda x:" ".join(x))
new_df['details'] = new_df['details'].apply(lambda x:" ".join(x))
```

```
new_df.head()
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['details'] = new_df['details'].apply(stem)
new_df['details'][0]
DataFrame = new_df[:5000]
DataFrame
from sklearn.feature_extraction.text import CountVectorizer
cv = CountVectorizer(max_features=5000,stop_words='english')
cv.fit_transform(DataFrame['details']).toarray()
cv.fit_transform(DataFrame['details']).toarray().shape
vectors = cv.fit_transform(DataFrame['details']).toarray()
vectors[0]
cv.get_feature_names()
from sklearn.metrics.pairwise import cosine_similarity
cosine_similarity(vectors)
similarity = cosine_similarity(vectors)
similarity[0]
k=sorted(list(enumerate(similarity[0])),reverse = True,key=lambda x:x[1])[1:6]
k
for i in range(5):
    print(DataFrame.iloc[k[i][0]].title, k[i][1])
def similarity_matrix(movie):
    lis = []
    movie_index = DataFrame[DataFrame['title'] == movie].index[0]
```

```
    distances = similarity[movie_index]
    movies_list = sorted(list(enumerate(distances)),reverse = True,key=lambda x:x[1])
    for i in range(len(movies_list)):
        lis.append(DataFrame.iloc[movies_list[i][0]].title)
        print(DataFrame.iloc[movies_list[i][0]].title,'------>similarity(',movies_list[i][1],')')
def recommend(movie):
    movie_index = DataFrame[DataFrame['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 range(len(movies_list)):
        print(DataFrame.iloc[movies_list[i][0]].title,'------>similarity(',movies_list[i][1],')')
similarity_matrix('Toy Story')
recommend('Toy Story')
import pickle
pickle.dump(DataFrame,open('movies.pkl','wb'))
DataFrame.to_dict()
pickle.dump(DataFrame.to_dict(),open('movie_dict.pk1','wb'))
pickle.dump(similarity,open('similarity.pk1','wb'))
```

5. RESULT ANALYSIS

In the resultant dataset consists of seven fields those are id, title, overview, genres, cast, crew, and keywords. The id field consists of id of the movie. The title field consists of title of the movie. The genres field consists of genres present in an movie. The overview field consists of overview of the movie. The cast attributes of top three casts present in an movie. The crew attribute consists of the director who worked in an movie. And finally, the keywords attribute consists of the keywords which are used in an movie.

Here, we check weather the recommended movies are the most similar movies present in the dataset or not by displaying the simialrity to each and every movie present in the dataset. Comparing the both weather the results are true or not. After comparing, our team conclude that the recommended movies are the most similar movies in the dataset. The Fig 5.1 shows the recommendation of movies to the given movie named 'Toy Story'.

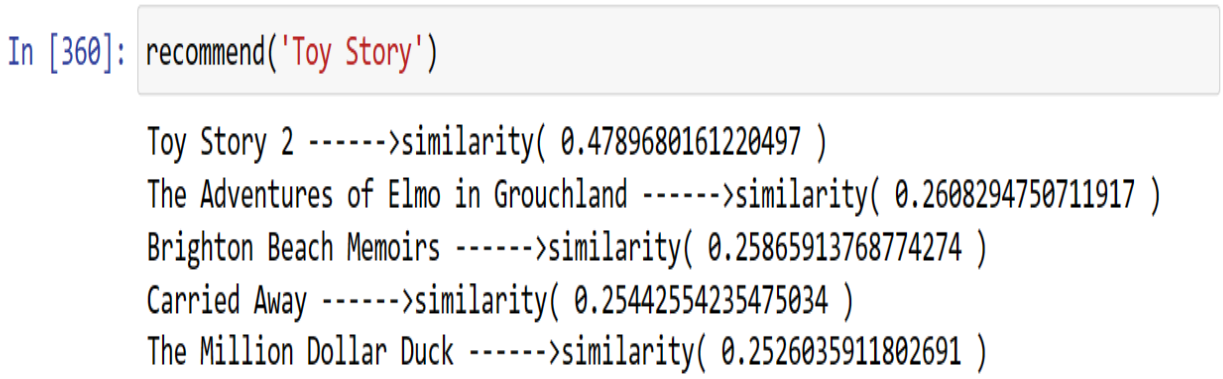

Fig:5.1: Displaying Recommendations

6. OUTPUT SCREENSHORTS

The home screen looks like the Fig 6.1, which contains the title of the project. Below the title of the project the statement ‘SEARCH AND SELECT THE MOVIE YOU LIKED MOST’ which acts as an hint for the user to search. And bellow it contains one rectangular box. The right side of the box contains one inverted triangle, when ever the user click it. It will displays the drop down of the movies list. Drop down of movies list is shown in Fig 6.2. Once the drop down is displayed the user can able to search and select the movie. After selecting a particular movie which is liked by the user. The selected movie title is displayed in inside the box. And finally, when ever the user click on the ‘Recommend’ button. The model will displays the top five movie tiltle’s which highest similarity to the selected movie title. The recommendations of movie title’s for the selected movie title is shown in the Fig 6.3.

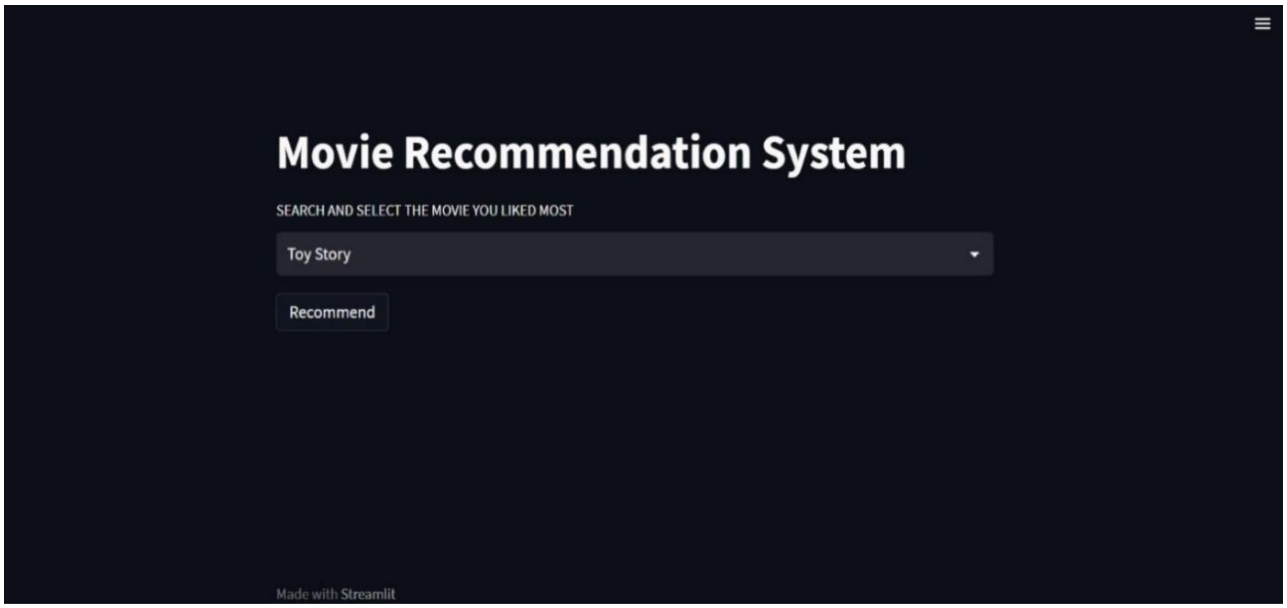

Fig:6.1 Home screen

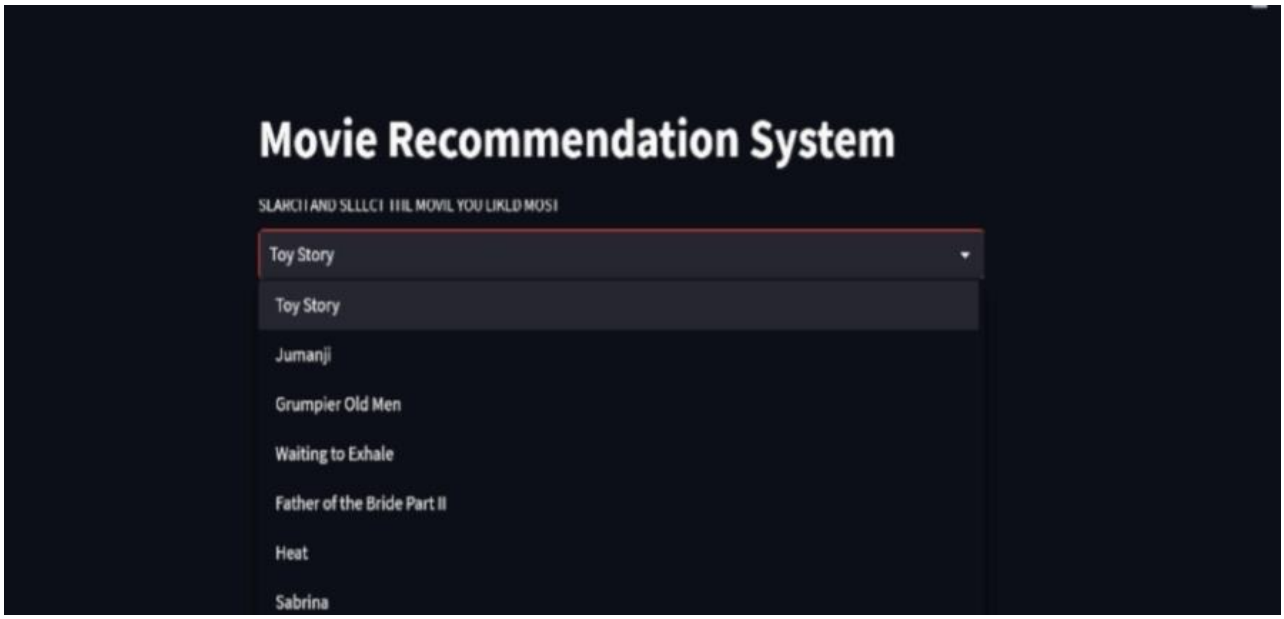

Fig:6.2 Dropdown of the movie list present in dataset

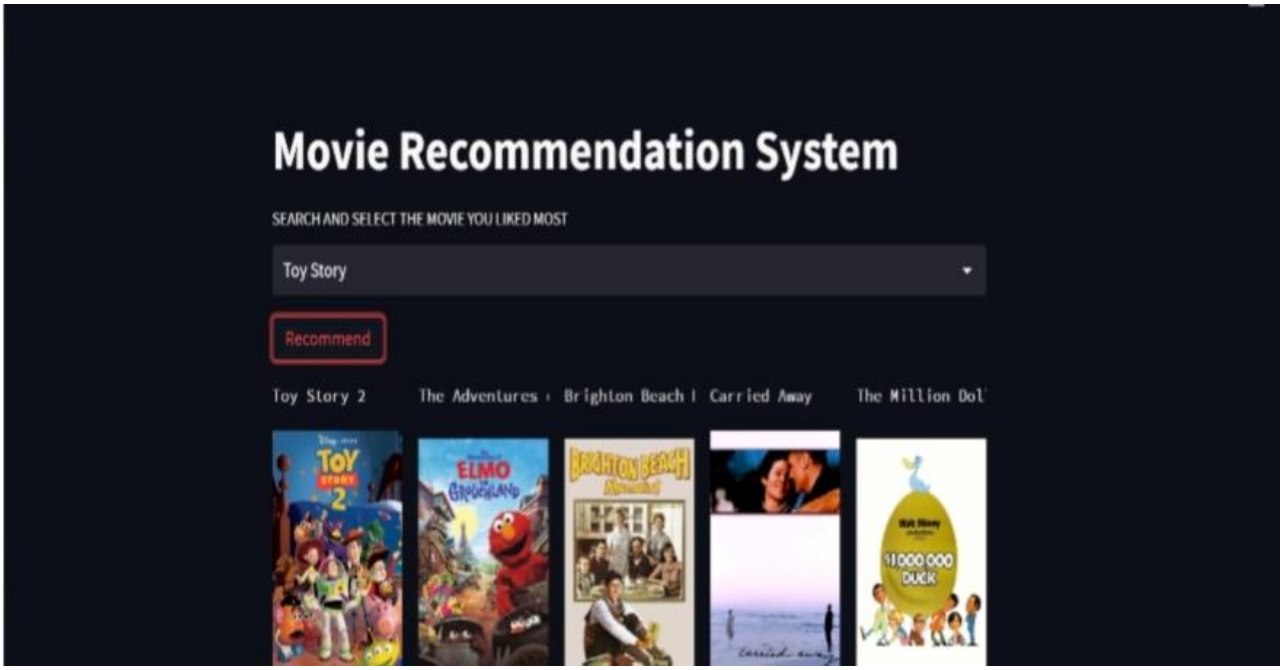

Fig:6.3 Recommended movies for a selected movie

7. CONCLUSION & FUTURE SCOPE

Now a days recommendation systems become more popular in various fields such as movie recommendation, music recommendation, product recommendations and so on. The applications which are using such recommendations are Netflix, Amazon prime, Aha, Meesho, Flipkart, Ajio and so on. Most of the fields using recommendation system, so we need to increase the accuracy of the results using different machine learning algorithms. Movie Recommendation System will help to users discover new movies based on their preference and interests. Benefits of using this system is saving users time for searching the movies, Discovers new content movies.

Looking for the future, there is much potential for further development and improvement of the Movie Recommendation System. Advances in machine learning and natural language processing additional data sources such as social media can provide even more personalized recommendations. Additionally, the rise of streaming services and increased availability of the data can provide new opportunities for recommendation systems to provide value to users. Overall, the future of the Movie Recommendation System is promising and we can expect continued innovation in this field.

8. BIBLIOGRAPHY

1. SRS Reddy, Sravani Nalluri, Subramanyam Kunisetti, S. Ashok and Venkatesh ----- Content-Based Movie Recommendation System Using Genre Correlation.
2. Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix factorization techniques for recommender systems. Computer, 42(8), 30-37.
3. He, X., Liao, L., Zhang, H., Nie, L., Hu, X., & Chua, T. S. (2017). Neural collaborative filtering. In Proceedings of the 26th international conference on world wide web (pp. 173-182).
4. Wang, H., Wang, N., Yeung, D. Y., & Shi, W. (2015). Collaborative deep learning for recommender systems. In Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (pp. 1235-1244).
5. X. Su and T. M. Khoshgoftaar, "A survey of collaborative filtering techniques," Advances in Artificial Intelligence, vol. 2009, Article ID 421425, 19 pages, 2009.
6. R. Burke, "Hybrid recommender systems: overview, current research, and future directions," in Recommender Systems Handbook, F. Ricci, L. Rokach, and B. Shapira, Eds. Springer, 2011, pp. 377-408.
7. A Content-Based Movie Recommender System Using Natural Language Processing by A. Biradar and P. Mahalle, International Journal of Computer Applications, 2016.
8. K. Burke, A. Mobasher, and R. Burke, "Hybrid recommender systems: survey and experiments," User Modeling and User-Adapted Interaction, vol. 12, no. 4, pp. 331-370, 2002.
9. A Content-Based Movie Recommender System Using Deep Learning by S. Tariq and S. Z. Qasim, International Journal of Innovative Technology and Exploring Engineering, 2021.
10. J. Lops, M. de Gemmis, and G. Semeraro, "Content-based recommender systems: state of the art and trends," in Recommender Systems Handbook, F. Ricci, L. Rokach, and B. Shapira, Eds. Springer, 2011, pp. 73-105.
11. A Hybrid Approach for Movie Recommendation Using Content-Based and Collaborative Filtering Techniques by N. U. Ekpunobi and C. U. Ekpunobi, International Journal of Advanced Research in Computer Science and Software Engineering, 2021.

12. A Content-Based Movie Recommendation System Using Semantic Analysis and Collaborative Filtering by M. Alam and M. A. Hossain, International Journal of Computer Science and Information Security, 2020.