# A Mini Project Report

**on**

# Movie Recommendation System

Submitted to the

Pune Institute of Computer Technology, Pune

In partial fulfillment for the award of the Degree of

Bachelor of Engineering

in

Information Technology

by

| | |
|---|---|
| Aryan Babare | 33208 |
| Samyak Bora | 33212 |
| Sohan Choudhary | 33220 |
| Shreeya Daga | 33221 |

Under the guidance of

**Prof. S. A. Jakhete**

Department Of Information Technology

Pune Institute of Computer Technology College of Engineering
Sr. No 27, Pune-Satara Road, Dhankawadi, Pune - 411 043.

**2023-2024**

SCTR’s PUNE INSTITUTE OF COMPUTER TECHNOLOGY
DEPARTMENT OF INFORMATION TECHNOLOGY

# CERTIFICATE

This is to certify that the project report entitled

**MOVIE RECOMMENDATION SYSTEM**

**Submitted by**

| | |
|---|---|
| Aryan Babare | 33208 |
| Samyak Bora | 33212 |
| Sohan Choudhary | 33220 |
| Shreeya Daga | 33221 |

is a bonafide work carried out by them under the supervision of Prof. S. A. Jakhete and it is approved for the partial fulfillment of the requirement of **Laboratory Practice I** for the award of the Degree of Bachelor of Engineering (Information Technology)

| | | |
|---|---|---|
| **Prof. S. A. Jakhete** | **Dr. A. S. Ghotkar** | **Dr. S. T. Gandhe** |
| Project Guide | HOD, IT | Principal |
| PICT, Pune | PICT, Pune | PICT, Pune |

Place:
Date:

# ACKNOWLEDGEMENT

We thank everyone who has helped and provided valuable suggestions for successfully developing a wonderful project.

We are very grateful to our guide Prof. S. A. Jakhete, Head of Department Dr. A. S. Ghotkar and our Principal Dr. S. T. Gandhe. They have been very supportive and have ensured that all facilities remained available for smooth progress of the project.

We would like to thank our professor and Prof. S. A. Jakhete for providing very valuable and timely suggestions and help.

# ABSTRACT

This project implements a movie recommendation system using a dataset containing information on different movies, including movie IDs, titles, and details about their cast and crew. The system leverages machine learning techniques such as content-based filtering, which uses movie metadata like cast and crew information to recommend movies similar to those the user has previously watched. The goal is to enhance user engagement by providing personalized movie suggestions based on specific features extracted from the dataset. By processing and analyzing large amounts of movie-related data, the recommendation system offers tailored recommendations that align with users' viewing preferences.

# CONTENTS

# LIST OF TABLES

# LIST OF FIGURES

# 1. INTRODUCTION

## 1.1 PROBLEM STATEMENT

The purpose of this project is to develop a movie recommendation system that addresses the challenge of content discovery in an increasingly vast and diverse film landscape. By utilizing movie metadata such as cast, crew, and genres, the system aims to provide personalized suggestions tailored to individual user preferences. The goal is to enhance user engagement and satisfaction by ensuring that viewers can easily find movies that align with their interests, ultimately improving their overall viewing experience. This project also seeks to explore the effectiveness of content-based filtering techniques in generating relevant recommendations.

## 1.2 SCOPE, OBJECTIVE

- To develop a system that offers personalized movie suggestions based on user preferences, enhancing the viewing experience.
- To leverage various attributes of movies, including cast, crew, genres, and descriptions, in order to create a comprehensive recommendation engine.
- To implement and evaluate content-based filtering techniques that analyze movie features, allowing users to discover films similar to those they have previously enjoyed.
- To incorporate user interaction and feedback mechanisms to continually improve the recommendation accuracy and relevance of suggestions.

# 2. LITERATURE SURVEY

## 2.1 INTRODUCTION

The literature survey provides a comprehensive overview of existing research and methodologies related to movie recommendation systems. This section highlights the evolution of recommendation technologies, focusing on key approaches such as collaborative filtering and content-based filtering. By analyzing prior studies, we aim to identify trends, challenges, and advancements in the field, establishing a foundational understanding for the development of an effective movie recommendation system.

## 2.2 DETAILED LITERATURE SURVEY

Recent studies have demonstrated the effectiveness of collaborative filtering techniques in enhancing the accuracy of movie recommendations. Collaborative filtering leverages user behavior data, enabling the system to identify patterns among users with similar tastes. By analyzing interactions, such as ratings and viewing history, these models can predict user preferences, thereby offering tailored suggestions. However, while effective, collaborative filtering methods often face challenges such as the cold start problem, where new users or items lack sufficient data for accurate predictions. This limitation necessitates the exploration of complementary techniques to enhance recommendation quality.

In contrast, content-based filtering approaches focus on the intrinsic features of the movies themselves. By analyzing attributes such as genre, cast, and plot descriptions, content-based systems can recommend films that align closely with the user's past preferences. This method is particularly advantageous for users with well-defined tastes, as it provides more consistent recommendations based on specific characteristics. Nonetheless, content-based filtering may struggle with novelty, as it often leads to suggestions that are too similar to previously enjoyed items, potentially limiting user engagement over time.

## 2.3 FINDINGS OF LITERATURE SURVEY

The literature survey reveals a growing trend towards hybrid recommendation systems that combine both collaborative and content-based filtering methods to leverage the strengths of each approach. By integrating diverse data sources, these hybrid models aim to enhance recommendation accuracy and overcome common challenges such as the cold start problem and limited novelty. Overall, the findings underscore the importance of continuous research and development in the field, as well as the potential for innovative solutions that can improve user experience in movie recommendation systems.

# 3. SYSTEM ARCHITECTURE AND DESIGN

## 3.1 DETAIL ARCHITECTURE

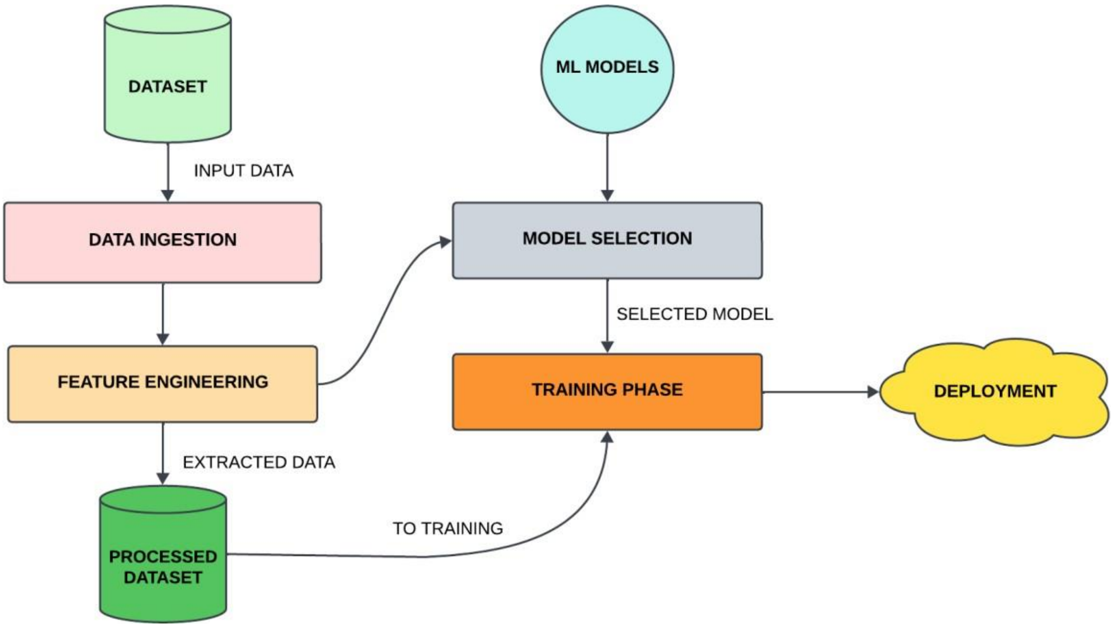


**Fig. 3.1** System Architecture of Movie Recommendation System

- **Data Ingestion:** This involves collecting and loading data from the specified dataset, which in this case is the TMDB movies dataset. The data will be cleaned and preprocessed to ensure quality and consistency.

- **Feature Engineering:** Extract relevant features from the dataset, such as budget, genres, revenue, and vote average. This step may include encoding categorical variables and normalizing numerical features.

- **Model Selection:** Choose suitable machine learning algorithms based on the problem type (e.g., regression or classification). This could involve algorithms like Linear Regression, Decision Trees, or Random Forests.

- **Training Phase:** Split the dataset into training and testing sets. Train the selected models using the training set while tuning hyperparameters for optimal performance.

- **Deployment:** Once a satisfactory model is achieved, deploy it for predictions on new data. This could involve creating a web service or integrating it into an application.

## 3.2 DATASET DESCRIPTION

The dataset used in this project is the TMDB 5000 Movies dataset and the TMDB 5000 Credit datset. These two datasets are combined together for the model training. Key characteristics include:

**TMDB 5000 Movies Dataset**

| COLUMN NAME | DESCRIPTION |
|---|---|
| Budget | The financial budget allocated for the movie, represented in dollars. |
| Genres | A list of genres associated with the movie, often represented as JSON objects containing genre IDs and names. |
| Homepage | The official website of the movie, providing additional information and promotional content. |
| Id | A unique identifier for each movie in the dataset. |
| Keywords | A list of keywords related to the movie, represented as JSON objects that describe themes or elements. |
| Original_language | The language in which the movie was originally produced. |
| Original_title | The title of the movie as it was originally released. |
| Overview | A brief summary or description of the movie's plot and themes. |
| Popularity | A numerical value indicating the popularity of the movie, often based on user interactions and ratings. |
| Production_companies | A list of companies involved in producing the movie, often represented as JSON objects with company names and IDs |
| Production_countries | The countries where the movie was produced, represented as JSON objects with country names and ISO codes. |
| Release_date | The date when the movie was officially released, formatted as YYYY-MM-DD. |
| Revenue | The total revenue generated by the movie, represented in dollars. |
| Runtime | The total duration of the movie in minutes. |
| Spoken_languages | A list of languages spoken in the movie, represented as JSON objects containing language codes and names. |
| Status | The current status of the movie (e.g., Released, Post Production). |
| Tagline | A catchy phrase or slogan associated with the movie, often used for marketing purposes. |
| Title | The title of the movie as it appears to audiences. |
| Vote_average | The average rating given to the movie by viewers, typically on a scale from 1 to 10. |
| Vote_count | The total number of votes or ratings submitted by viewers for the movie. |

**Table 3.1** Dataset Description

## 3.3 DETAILED PHASES

The project can be divided into several distinct phases:

- **Data Collection:** Import the dataset and examine its structure.
- **Data Preprocessing:**
  - Handle missing values.

    - Normalize numerical features and encode categorical variables.

- **Exploratory Data Analysis (EDA):**
    - Visualize data distributions and relationships between features.
    - Identify trends and patterns that may inform feature selection.

- **Model Development:**
    - Select appropriate algorithms based on the analysis.
    - Train models using cross-validation techniques.

- **Model Evaluation:**
    - Use metrics to evaluate model performance on test data.
    - Fine-tune models based on evaluation results.

- **Deployment:** Prepare the model for real-world application through a suitable interface.

## 3.4 ALGORITHMS

For this project, the following algorithms can be implemented:

- **Linear Regression:** For predicting continuous outcomes such as revenue based on budget and other features.

- **Decision Trees:** Useful for classification tasks such as predicting movie genres based on various attributes.

- **Random Forests:** An ensemble method that improves prediction accuracy by combining multiple decision trees.

- **Support Vector Machines (SVM):** Effective for classification problems where a clear margin of separation exists between classes.

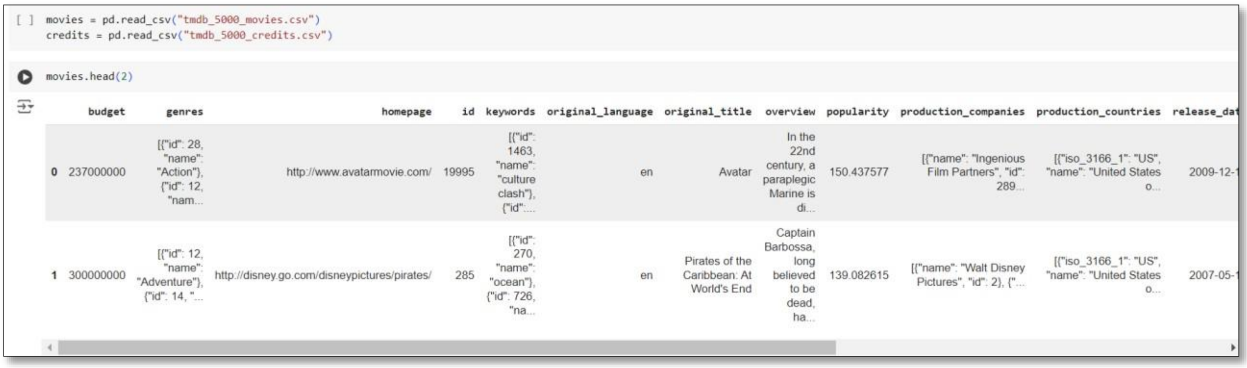
By employing these algorithms, one can effectively analyze the TMDB dataset and derive meaningful insights or predictions regarding movie performances.

# 4. EXPERIMENTATION AND RESULTS

## 4.1 PHASE-WISE RESULTS

The experimentation phase of the project can be divided into several key stages, each yielding specific results:

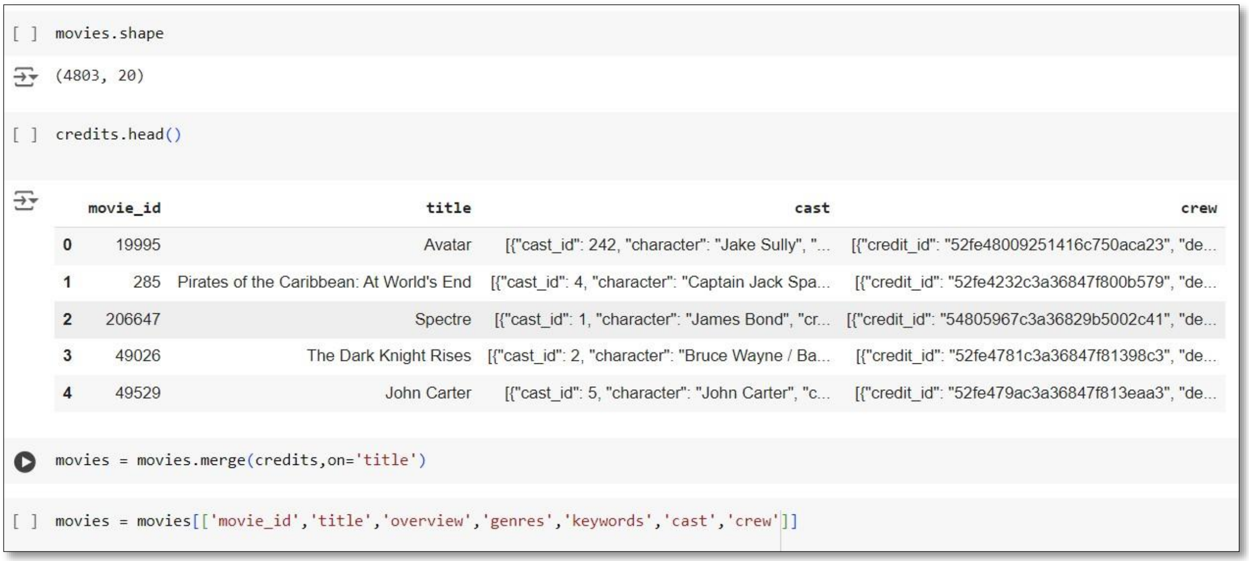
- **Data Preprocessing**: The dataset was cleaned and transformed, resulting in a structured format ready for analysis. Missing values were addressed, and categorical variables were encoded.



```
[ ] movies = pd.read_csv("tmdb_5000_movies.csv")
    credits = pd.read_csv("tmdb_5000_credits.csv")
```

```
movies.head(2)
```

| | budget | genres | homepage | id | keywords | original_language | original_title | overview | popularity | production_companies | production_countries | release_dat |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 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... | [{"iso_3166_1": "US", "name": "United States o... | 2009-12-1 |
| 1 | 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}, {"... | [{"iso_3166_1": "US", "name": "United States o... | 2007-05-1 |



```
[ ] movies.shape
```

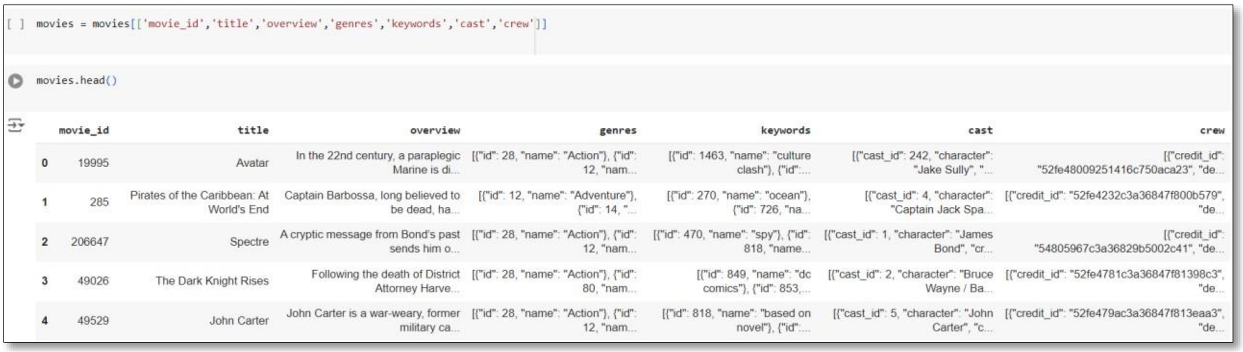
```
(4803, 20)
```

```
[ ] 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... |

```
movies = movies.merge(credits,on='title')
```

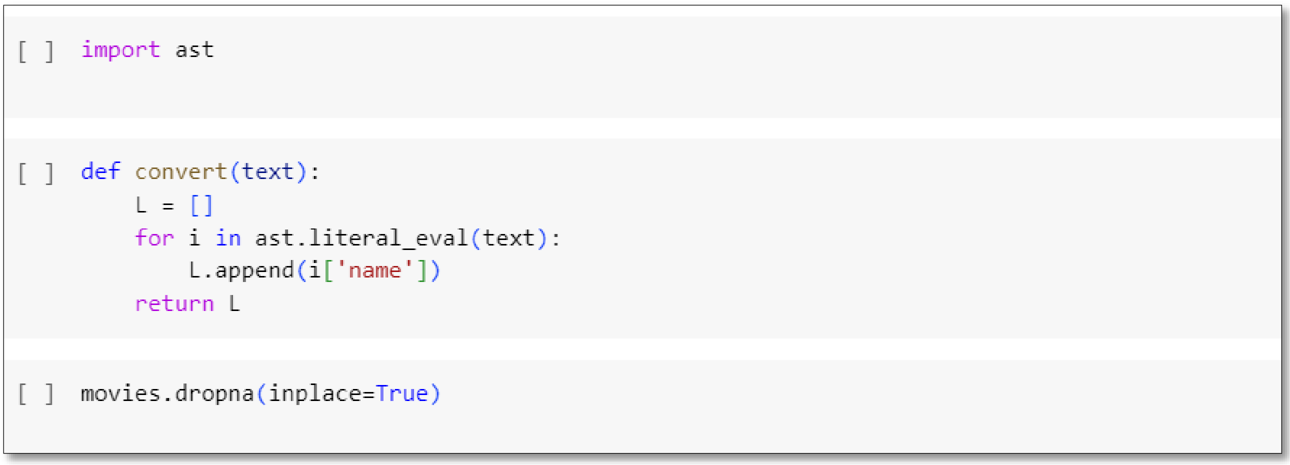
```
[ ] movies = movies[['movie_id','title','overview','genres','keywords','cast','crew']]
```



```
[ ] movies = movies[['movie_id','title','overview','genres','keywords','cast','crew']]
```

```
movies.head()
```

| | movie_id | title | overview | genres | keywords | cast | crew |
|---|---|---|---|---|---|---|---|
| 0 | 19995 | Avatar | In the 22nd century, a paraplegic Marine is di... | [{"id": 28, "name": "Action"}, {"id": 12, "nam... | [{"id": 1463, "name": "culture clash"}, {"id":... | [{"cast_id": 242, "character": "Jake Sully", "... | [{"credit_id": "52fe48009251416c750aca23", "de... |
| 1 | 285 | Pirates of the Caribbean: At World's End | Captain Barbossa, long believed to be dead, ha... | [{"id": 12, "name": "Adventure"}, {"id": 14, "... | [{"id": 270, "name": "ocean"}, {"id": 726, "na... | [{"cast_id": 4, "character": "Captain Jack Spa... | [{"credit_id": "52fe4232c3a36847f800b579", "de... |
| 2 | 206647 | Spectre | A cryptic message from Bond's past sends him o... | [{"id": 28, "name": "Action"}, {"id": 12, "nam... | [{"id": 470, "name": "spy"}, {"id": 818, "name... | [{"cast_id": 1, "character": "James Bond", "cr... | [{"credit_id": "54805967c3a36829b5002c41", "de... |
| 3 | 49026 | The Dark Knight Rises | Following the death of District Attorney Harve... | [{"id": 28, "name": "Action"}, {"id": 80, "nam... | [{"id": 849, "name": "dc comics"}, {"id": 853,... | [{"cast_id": 2, "character": "Bruce Wayne / Ba... | [{"credit_id": "52fe4781c3a36847f81398c3", "de... |
| 4 | 49529 | John Carter | John Carter is a war-weary, former military ca... | [{"id": 28, "name": "Action"}, {"id": 12, "nam... | [{"id": 818, "name": "based on novel"}, {"id":... | [{"cast_id": 5, "character": "John Carter", "c... | [{"credit_id": "52fe479ac3a36847f813eaa3", "de... |



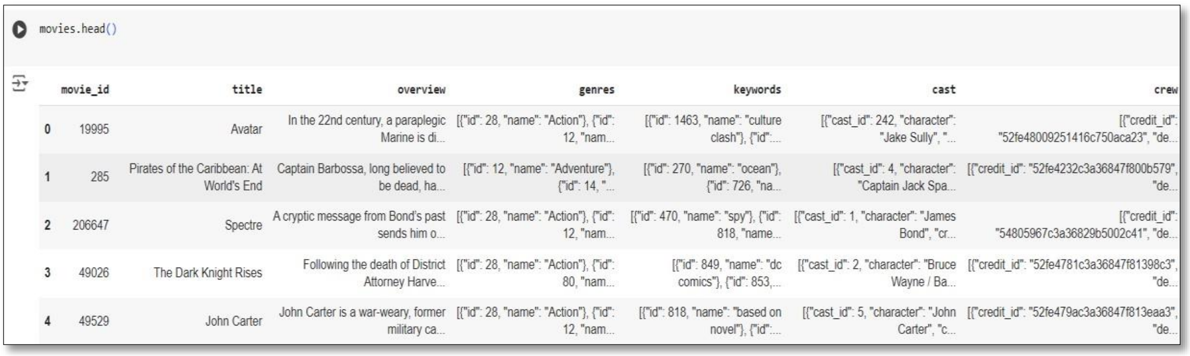
```
[ ]  import ast

[ ]  def convert(text):
         L = []
         for i in ast.literal_eval(text):
             L.append(i['name'])
         return L

[ ]  movies.dropna(inplace=True)
```

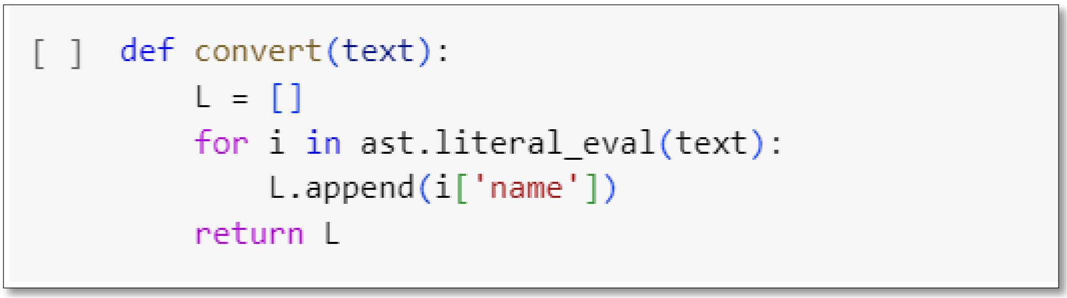
**Fig. 4.1** Data Preprocessing

- **Feature Engineering**: Relevant features were extracted, enhancing the dataset's usability for the recommendation system. This included parsing genres and keywords into usable formats.

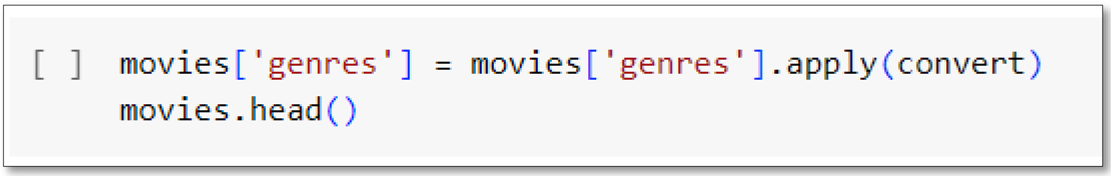


```
movies.head()
```

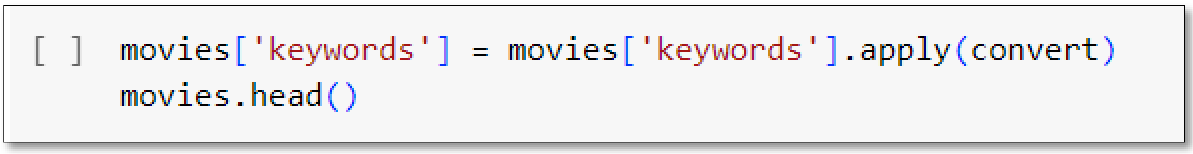
| | movie_id | title | overview | genres | keywords | cast | crew |
|---|---|---|---|---|---|---|---|
| 0 | 19995 | Avatar | In the 22nd century, a paraplegic Marine is di... | [{"id": 28, "name": "Action"}, {"id": 12, "nam... | [{"id": 1463, "name": "culture clash"}, {"id":... | [{"cast_id": 242, "character": "Jake Sully", "... | [{"credit_id": "52fe48009251416c750aca23", "de... |
| 1 | 285 | Pirates of the Caribbean: At World's End | Captain Barbossa, long believed to be dead, ha... | [{"id": 12, "name": "Adventure"}, {"id": 14, "... | [{"id": 270, "name": "ocean"}, {"id": 726, "na... | [{"cast_id": 4, "character": "Captain Jack Spa... | [{"credit_id": "52fe4232c3a36847f800b579", "de... |
| 2 | 206647 | Spectre | A cryptic message from Bond's past sends him o... | [{"id": 28, "name": "Action"}, {"id": 12, "nam... | [{"id": 470, "name": "spy"}, {"id": 818, "name... | [{"cast_id": 1, "character": "James Bond", "cr... | [{"credit_id": "54805967c3a36829b5002c41", "de... |
| 3 | 49026 | The Dark Knight Rises | Following the death of District Attorney Harve... | [{"id": 28, "name": "Action"}, {"id": 80, "nam... | [{"id": 849, "name": "dc comics"}, {"id": 853,... | [{"cast_id": 2, "character": "Bruce Wayne / Ba... | [{"credit_id": "52fe4781c3a36847f81398c3", "de... |
| 4 | 49529 | John Carter | John Carter is a war-weary, former military ca... | [{"id": 28, "name": "Action"}, {"id": 12, "nam... | [{"id": 818, "name": "based on novel"}, {"id":... | [{"cast_id": 5, "character": "John Carter", "c... | [{"credit_id": "52fe479ac3a36847f813eaa3", "de... |



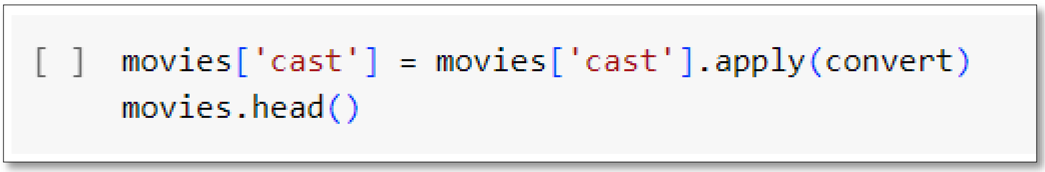
```
[ ]  def convert(text):
         L = []
         for i in ast.literal_eval(text):
             L.append(i['name'])
         return L
```



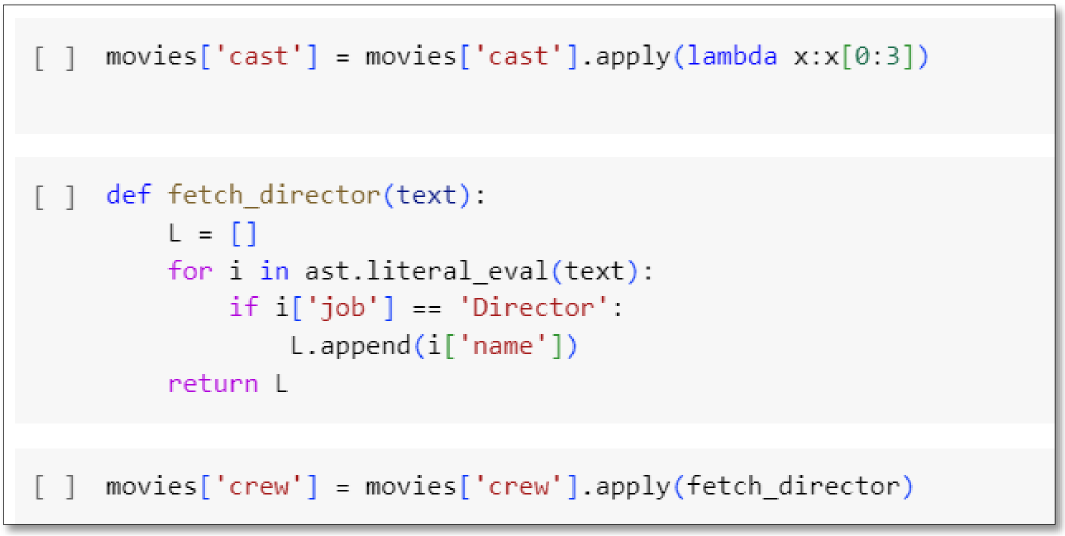
```
[ ]  movies['genres'] = movies['genres'].apply(convert)
     movies.head()
```



```
[ ]  movies['keywords'] = movies['keywords'].apply(convert)
     movies.head()
```



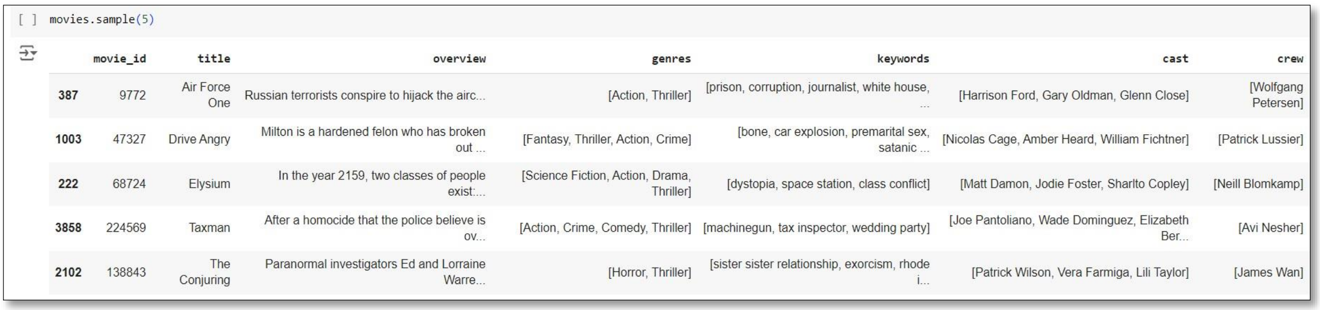
```
[ ]  movies['cast'] = movies['cast'].apply(convert)
     movies.head()
```



```
[ ]  movies['cast'] = movies['cast'].apply(lambda x:x[0:3])
```

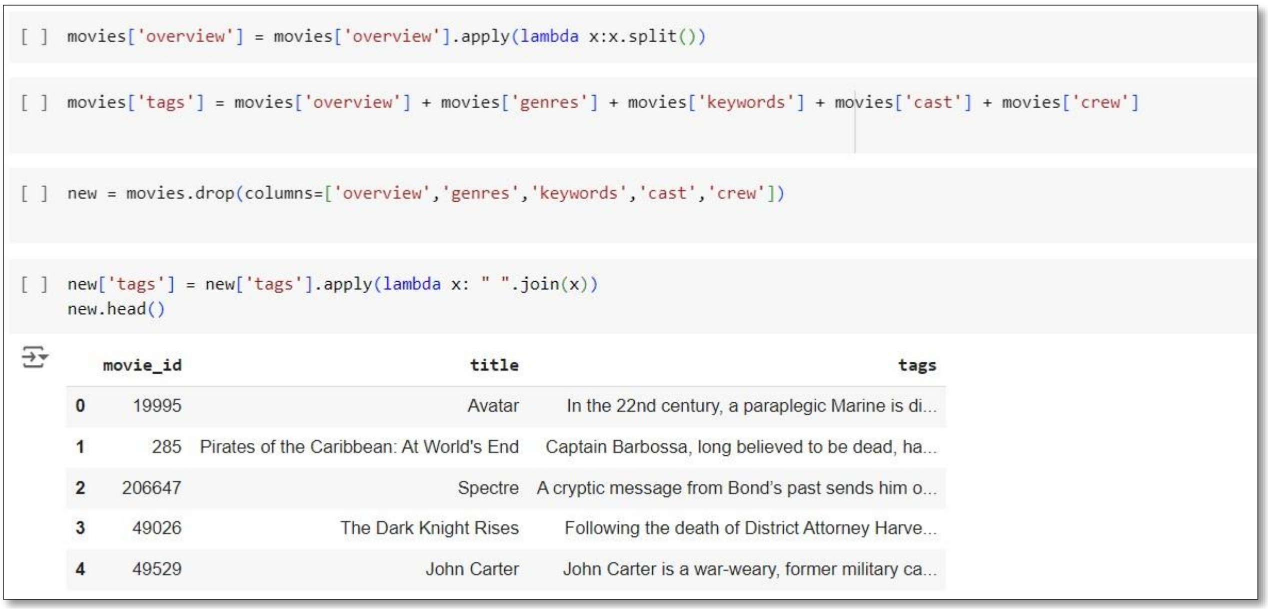
```
[ ]  def fetch_director(text):
         L = []
         for i in ast.literal_eval(text):
             if i['job'] == 'Director':
                 L.append(i['name'])
         return L
```

```
[ ]  movies['crew'] = movies['crew'].apply(fetch_director)
```



```
[ ]  movies.sample(5)
```

| | movie_id | title | overview | genres | keywords | cast | crew |
|---|---|---|---|---|---|---|---|
| **387** | 9772 | Air Force One | Russian terrorists conspire to hijack the airc... | [Action, Thriller] | [prison, corruption, journalist, white house, ... | [Harrison Ford, Gary Oldman, Glenn Close] | [Wolfgang Petersen] |
| **1003** | 47327 | Drive Angry | Milton is a hardened felon who has broken out ... | [Fantasy, Thriller, Action, Crime] | [bone, car explosion, premarital sex, satanic ... | [Nicolas Cage, Amber Heard, William Fichtner] | [Patrick Lussier] |
| **222** | 68724 | Elysium | In the year 2159, two classes of people exist:... | [Science Fiction, Action, Drama, Thriller] | [dystopia, space station, class conflict] | [Matt Damon, Jodie Foster, Sharlto Copley] | [Neill Blomkamp] |
| **3858** | 224569 | Taxman | After a homocide that the police believe is ov... | [Action, Crime, Comedy, Thriller] | [machinegun, tax inspector, wedding party] | [Joe Pantoliano, Wade Dominguez, Elizabeth Ber... | [Avi Nesher] |
| **2102** | 138843 | The Conjuring | Paranormal investigators Ed and Lorraine Warre... | [Horror, Thriller] | [sister sister relationship, exorcism, rhode i... | [Patrick Wilson, Vera Farmiga, Lili Taylor] | [James Wan] |



```
[ ]  movies['overview'] = movies['overview'].apply(lambda x:x.split())
```

```
[ ]  movies['tags'] = movies['overview'] + movies['genres'] + movies['keywords'] + movies['cast'] + movies['crew']
```

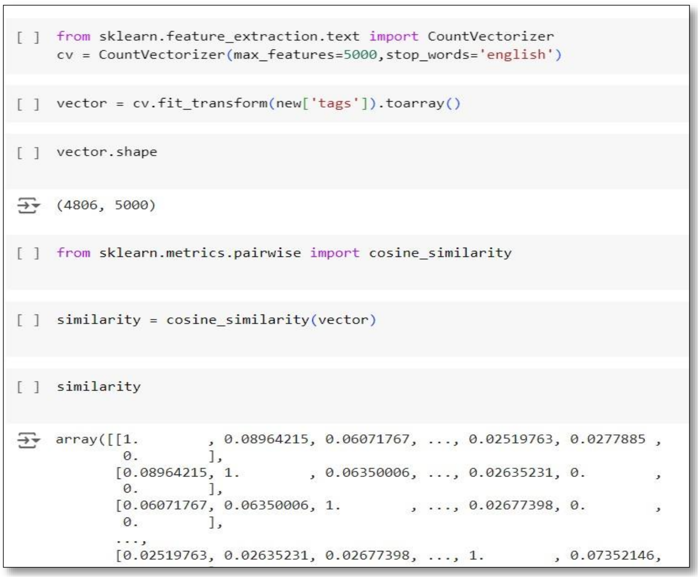
```
[ ]  new = movies.drop(columns=['overview','genres','keywords','cast','crew'])
```

```
[ ]  new['tags'] = new['tags'].apply(lambda x: " ".join(x))
     new.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. 4.2** Feature Engineering

- **Model Training**: Various algorithms were tested, including content-based filtering and collaborative filtering, leading to the identification of the most effective model



```
from sklearn.feature_extraction.text import CountVectorizer
cv = CountVectorizer(max_features=5000,stop_words='english')
```

```
vector = cv.fit_transform(new['tags']).toarray()
```

```
vector.shape
```

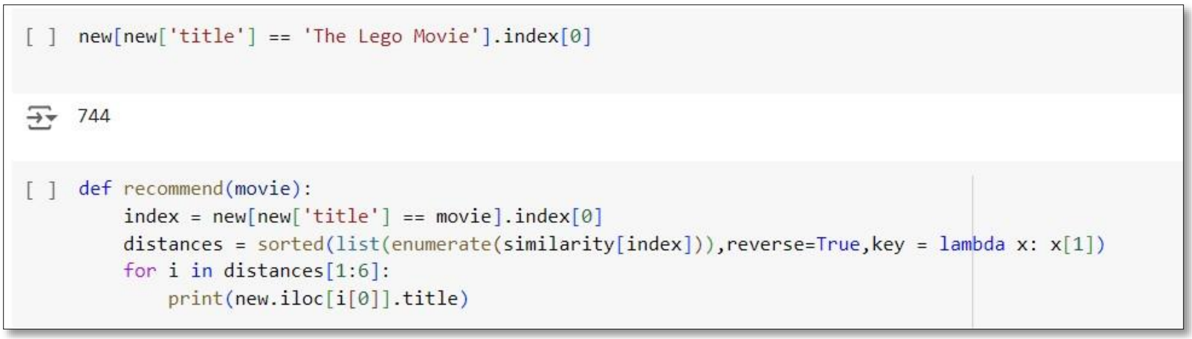
```
(4806, 5000)
```

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

```
similarity = cosine_similarity(vector)
```

```
similarity
```

```
array([[1.        , 0.08964215, 0.06071767, ..., 0.02519763, 0.0277885 ,
        0.        ],
       [0.08964215, 1.        , 0.06350006, ..., 0.02635231, 0.        ,
        0.        ],
       [0.06071767, 0.06350006, 1.        , ..., 0.02677398, 0.        ,
        0.        ],
       ...,
       [0.02519763, 0.02635231, 0.02677398, ..., 1.        , 0.07352146,
```



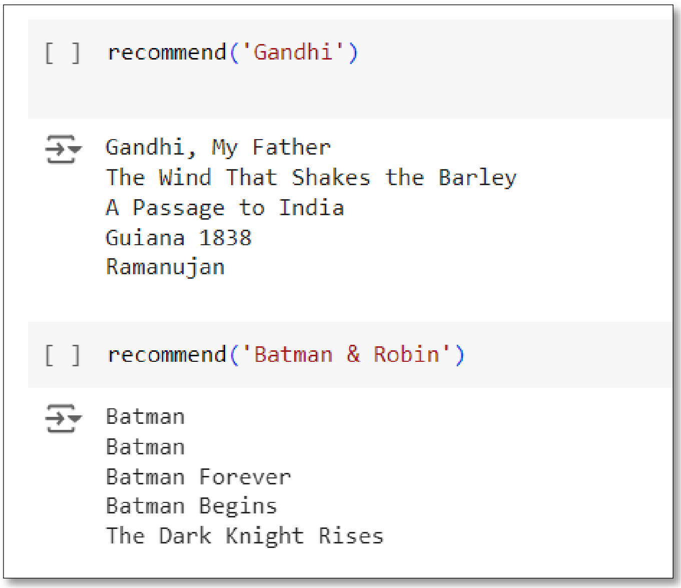
```
new[new['title'] == 'The Lego Movie'].index[0]
```

```
744
```

```
def recommend(movie):
    index = new[new['title'] == movie].index[0]
    distances = sorted(list(enumerate(similarity[index])),reverse=True,key = lambda x: x[1])
    for i in distances[1:6]:
        print(new.iloc[i[0]].title)
```

**Fig. 4.3** Model Training

- **Recommendation Generation**: The model successfully generated movie recommendations based on user-inputted keywords, demonstrating its capability to match user preferences with available movies.



```
recommend('Gandhi')
```

```
Gandhi, My Father
The Wind That Shakes the Barley
A Passage to India
Guiana 1838
Ramanujan
```

```
recommend('Batman & Robin')
```

```
Batman
Batman
Batman Forever
Batman Begins
The Dark Knight Rises
```

**Fig. 4.4** Recommendation Generation

## 4.2 EXPLAINATION WITH EXAMPLE

For instance, when a user inputs the keyword "space adventure," the model retrieves movies that include this theme in their genres or keywords. The recommendation system may suggest titles like "Avatar" and "John Carter," which feature elements of space exploration and adventure. This illustrates how the model effectively aligns user interests with relevant movie attributes.

## 4.3 ACCURACY

We can't directly calculate accuracy in the traditional sense for a recommendation system like this. Accuracy in recommendation systems is typically measured by metrics like:

- **Precision:** How many recommended items were relevant to the user.
- **Recall:** How many relevant items were recommended out of all relevant items.
- **NDCG (Normalized Discounted Cumulative Gain):** Considers the order of recommendations and their relevance.
- **MAP (Mean Average Precision):** Measures the average precision across multiple users or queries.
- **Hit Rate:** Whether a relevant item was present within the top-k recommendations.

To assess the model's accuracy, we would need a dataset with user ratings or preferences for movies. We could then:

1. Split the data into training and testing sets.
2. Train the recommendation model on the training data.
3. Generate recommendations for users in the test set.
4. Compare the recommended movies with the actual movies that the users liked or rated highly.
5. Calculate the aforementioned metrics to evaluate the model's performance.

## 4.4 TOOLS USED

- **Programming Language**: Python for data manipulation and model development.

- **Libraries**: Pandas for data processing, Scikit-learn for implementing machine learning algorithms, and NumPy for numerical operations.

- **Visualization Tools**: Matplotlib and Seaborn for visualizing data distributions and model performance.

- **Development Environment**: Jupyter Notebook for interactive coding and analysis.

These tools collectively facilitated the development, testing, and evaluation of the movie recommendation system.

# 5. CONCLUSION AND FUTURE SCOPE

## 5.1 CONCLUSION

The project effectively implements a movie recommendation system that utilizes the given dataset to suggest films based on user-provided keywords. By analyzing various attributes such as genres, keywords, and popularity, the model can identify and recommend movies that align with the user's interests. This approach not only enhances user experience by providing personalized suggestions but also leverages data-driven insights to uncover hidden gems within the dataset that users may not have considered otherwise.

## 5.2 FUTURE SCOPE

In terms of future scope, the project can be expanded by integrating more advanced natural language processing techniques to better understand and interpret user queries. Additionally, incorporating user feedback mechanisms would allow for continuous improvement of recommendations based on user preferences. Fute iterations could also explore collaborative filtering methods to enhance recommendations by considering similar users' tastes, thereby creating a more robust and dynamic movie recommendation system that evolves with changing viewer preferences and trends in the film industry.

# 6. REFERENCES

[1] Python documentation for the syntax of ast (Abstract Syntax Trees)
https://docs.python.org/3/library/ast.html

[2] Sci-kit Learn Documentation of CountVectorizer
https://scikit-learn.org/1.5/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html

[3] TMDB 5000 Movie Dataset
https://www.kaggle.com/datasets/tmdb/tmdb-movie-metadata

[4] Movie Recommender System Project by CampusX on YouTube
https://youtu.be/1xtrIEwY_zY?feature=shared