MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE GWALIOR

(A Govt. Aided UGC Autonomous Institute Affiliated to RGPV, Bhopal)

NAAC Accredited with A++ Grade

Project Report

on

MOVIE RECOMMENDER SYSTEM

(270506)

Submitted By:

Harsh Vardhan Choudhary (0901AD211019)

Shruti Garg (0901AD211059)

Faculty Mentor:

Prof. Deepti Gupta

CENTRE FOR ARTIFICIAL INTELLIGENCE

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE

GWALIOR - 474005 (MP) est. 1957

JULY-DEC. 2023

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE GWALIOR

(A Govt. Aided UGC Autonomous Institute Affiliated to RGPV, Bhopal)

NAAC Accredited with A++ Grade

CERTIFICATE

This is certified that **Harsh Vardhan Choudhary (0901AD211019) and Shruti Garg (0901AD211059)** has submitted the project report titled "**Movie Recommendation System**" under the mentorship of **Prof. Deepti Gupta**, in partial fulfilment of the requirement for the award of degree of Bachelor of Technology in **Artificial Intelligence And Data Science** from Madhav Institute of Technology and Science, Gwalior.

23/11/23

Prof. Deepti Gupta
Faculty Mentor
Assistant Professor
Centre for Artificial Intelligence

23/11/23

Dr. R. R. Singh
Coordinator
Centre for Artificial Intelligence

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE GWALIOR

(A Govt. Aided UGC Autonomous Institute Affiliated to RGPV, Bhopal)

NAAC Accredited with A++ Grade

DECLARATION

I hereby declare that the work being presented in this project report, for the partial fulfilment of requirement for the award of the degree of Bachelor of Technology in **Artificial Intelligence And Data Science** at Madhav Institute of Technology & Science, Gwalior is an authenticated and original record of my work under the mentorship of **Prof. Deepti Gupta**, Assistant Professor, Centre for Artificial Intelligence.

I declare that I have not submitted the matter embodied in this report for the award of any degree or diploma anywhere else.

Harsh Vardhan Choudhary (0901AD211019)

Shruti Garg (0901AD211059)

3rd Year

Centre for Artificial Intelligence

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE GWALIOR

(A Govt. Aided UGC Autonomous Institute Affiliated to RGPV, Bhopal)

NAAC Accredited with A++ Grade

ACKNOWLEDGEMENT

The full semester project has proved to be pivotal to my career. I am thankful to my institute, **Madhav Institute of Technology and Science** to allow me to continue my disciplinary/interdisciplinary project as a curriculum requirement, under the provisions of the Flexible Curriculum Scheme (based on the AICTE Model Curriculum 2018), approved by the Academic Council of the institute. I extend my gratitude to the Director of the institute, **Dr. R. K. Pandit** and Dean Academics, **Dr. Manjaree Pandit** for this.

I would sincerely like to thank my department, **Centre for Artificial Intelligence,** for allowing me to explore this project. I humbly thank **Dr. R. R. Singh**, Coordinator, Centre for Artificial Intelligence, for his continued support during the course of this engagement, which eased the process and formalities involved.

I am sincerely thankful to my faculty mentors. I am grateful to the guidance of **Prof. Deepti Gupta**, Assistant Professor, Centre for Artificial Intelligence, for his continued support and guidance throughout the project. I am also very thankful to the faculty and staff of the department.

Harsh Vardhan Choudhary (0901AD211019)

Shruti Garg (0901AD211059)

3rd Year

Centre for Artificial Intelligence

ABSTRACT

This project introduces a movie recommendation system built upon a content-based filtering approach, vectorisation techniques, and cosine similarity to facilitate personalized movie recommendations. Utilizing a dataset encompassing 5000 movies and featuring 20 attributes such as budget, genres, keywords, release date, and more, this system employs a content-based filtering strategy to generate movie suggestions based on inherent movie characteristics. Leveraging vectorisation methods to transform textual data into numerical representations and employing cosine similarity as a metric, the system calculates similarities between movies. The deployment of this recommendation system is achieved through Streamlit, providing a user-friendly interface for users to explore and discover tailored movie recommendations. By utilizing movie attributes and cosine similarity, the system aids users in discovering movies aligned with their preferences, ultimately enhancing their movie-watching experience.

सार

यह प्रोजेक्ट एक मूवी सिफारिश प्रणाली पर आधारित है, जो एक सामग्री-आधारित फ़िल्टरिंग दृष्टिकोण, वेक्टरीकरण तकनीकों, और कोसाइन समानता का उपयोग करके व्यक्तिगत मूवी सिफारिशों को सुविधाजनक बनाता है। 5000 मूवीज़ समाविष्ट करने वाले एक डेटासेट का उपयोग करते हुए और जिसमें बजट, शैली, कीवर्ड्स, रिलीज़ डेट, आदि जैसी 20 विशेषताएं होती हैं, यह सिस्टम मूवी की विशेषताओं पर आधारित फ़िल्टरिंग करने की रणनीति का उपयोग करता है ताकि मूवी सुझाव बनाया जा सके। पाठ्यक डेटा को संख्यात्मक प्रतिनिधित्वों में परिवर्तित करने के लिए वेक्टरीकरण विधियों का उपयोग करते हुए और कोसाइन समानता को माप के रूप में लेकर, सिस्टम मूवीज़ के बीच समानताएँ की गणना करता है। सुझाव प्रणाली का डिप्लॉयमेंट स्ट्रीमलाइट के माध्यम से किया गया है, जो उपयोगकर्ताओं को एक उपयोगकर्ता-मित्र अन्वेषण के लिए साझा किया गया है। मूवी विशेषताओं और कोसाइन समानता का उपयोग करके, सिस्टम उपयोगकर्ताओं को उनकी पसंदों के साथ संगत मूवीज़ खोजने में मदद करता है, अंत में उनके मूवी देखने के अनुभव को बेहतर बनाता है।

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1 : PROJECT OVERVIEW

1.1 INTRODUCTION

The movie recommendation system stands as a pivotal advancement in the entertainment industry, revolutionizing the way audiences discover and engage with films. Through the amalgamation of sophisticated algorithms, data processing techniques, and user interaction patterns, this system provides personalized movie suggestions, enhancing user experience and satisfaction. Leveraging content-based filtering, collaborative filtering, or hybrid approaches, these systems analyze vast datasets encompassing diverse movie attributes such as genres, keywords, release dates, and more. By comprehensively understanding user preferences and movie characteristics, these systems generate tailored recommendations, guiding users toward discovering films aligned with their tastes and interests.

1.2 PROJECT AIM

The primary aim of this project is to create an effective movie recommendation system using content-based filtering techniques. The system will analyze the features of movies, such as genre, actors, directors, keywords, or plot summaries, and represent them in a numerical format using vectorisation. Then, it will utilize cosine similarity to compute the similarity between these numerical representations of movies and recommend similar movies to users based on their preferences.

1.3 OBJECTIVE

Data Collection and Preprocessing: Gather movie-related data from sources like IMDb, TMDB, or other movie databases. Preprocess the data by cleaning, organizing, and extracting relevant features such as genre, cast, crew, synopsis, and ratings.

Feature Representation using Vectorization: Convert the textual or categorical movie features into a numerical format suitable for machine learning algorithms. Techniques like TF-IDF (Term Frequency-Inverse Document Frequency) or one-hot encoding can be used for this purpose.

Cosine Similarity Calculation: Implement cosine similarity to measure the similarity between movies based on their numerical representations. This involves calculating the cosine of the angle between the feature vectors of different movies.

User Profile Creation: Capture user preferences by analyzing the movies they have interacted with, liked, or rated. Create a user profile based on these preferences to understand their taste in movies.

Recommendation Generation: Utilize the user profile and the similarity scores between movies to generate a list of recommended movies. The system will suggest movies similar to those that the user has previously shown interest in.

Evaluation and Improvement: Assess the performance of the recommendation system using metrics like precision, recall, or accuracy. Continuously refine the system by incorporating user feedback and enhancing the recommendation algorithm.

CHAPTER 2 : LITERATURE REVIEW

Movie recommendation systems are a popular research area in recent years, as they provide personalized suggestions to users based on their past preferences and ratings. There are two main types of movie recommendation systems: content-based and collaborative filtering.

Content-based systems recommend items that are similar to those provided by the user. For example, if a user has rated the movie "The Shawshank Redemption" highly, the system might recommend other movies that are similar to it, such as "The Godfather" or "The Dark Knight".

Collaborative filtering systems identify users whose tastes are similar to those of the given user and recommend items they have liked. For example, if a user has rated the movie "The Shawshank Redemption" highly, the system might recommend other movies that have been rated highly by other users who have also rated "The Shawshank Redemption" highly.

Movie recommendation systems can be used in a variety of ways. They can be used to help users find new movies to watch, to discover new genres of movies, and to get personalized recommendations for movies that they are likely to enjoy.

CHAPTER 3 : SYSTEM SPECIFICATION

3.1 HARDWARE SPECIFICATION

Since the hardware is an important part while developing a project, it's necessary to find hardware requirements.

Processor– The minimum level of the required processor for this platform is Pentium IV with 800 MHz processing speed. Since the time taken for processing the instruction depends on the processor's power, it is very important to choose the required processor.

RAM– For a higher speed of processing, it also depends on the memory. Therefore, for better performance minimum RAM should be 2 GB.

Hard disk– In modern days, a vast amount of data is generated daily from internet platforms. So, a good size hard disk is required for the storage of the processed data.

Cache Memory– The access time of the operation tasks mainly depends on the cache memory. Therefore, the recommended cache memory is 1000 KB.

3.2 SOFTWARE SPECIFICATION

Our system should meet the following minimum specifications OS — Ubuntu, Windows, Mac OS

We will be using ANACONDA - NAVIGATOR and Python and we also need an application Visual Studio Code and PyCharm as a code editor.

CHAPTER 4: METHODOLOGY

4.1 DATA COLLECTION

In machine learning, data collection is a critical step that involves gathering the necessary data required for training and testing a model. The choice of data collection method depends on the type of data required and the nature of the problem being addressed. In this model, data is taken from Kaggle which is a dataset repository. Our data has the following attributes are budget, genres, homepage, id, keywords, original_language, original_title, overview popularity, production_companies, production_countries, release_date, revenue, runtime, spoken_languages, status, tagline, title, vote_average, vote_count , movie_id, title, cast, crew.

```
In [3]: movies = pd.read_csv('movies.csv')
        credits = pd.read_csv('credits.csv')

In [5]: movies.head()
```

Out[5]:

	budget	genres	homepage	id	keywords	original_language	original_title	overview	popularity	production_co
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": "I Film Partn
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": "Wa Pictures", "i
2	245000000	[{"id": 28, "name": "Action"}, {"id": 12, "nam...	http://www.sonypictures.com/movies/spectre/	206647	[{"id": 470, "name": "spy"}, {"id": 818, "name...	en	Spectre	A cryptic message from Bond's past sends him o...	107.376788	[{"name": " Pictures
3	250000000	[{"id": 28, "name": "Action"}, {"id": 80, "nam...	http://www.thedarkknightrises.com/	49026	[{"id": 849, "name": "dc comics"}, {"id": 853,...	en	The Dark Knight Rises	Following the death of District Attorney Harve...	112.312950	[{"name": "L Pictures", "id":

FIGURE 4.1

4.2 DATA PRE-PROCESSING

Data pre-processing is an essential step in machine learning that involves transforming raw data into a format that is suitable for training a model.

4.2.1 DATA CLEANING

Data cleaning involves handling missing data, dealing with duplicates, and removing outliers. Missing data can be imputed with appropriate values such as the mean or median of the feature, or it can be removed entirely if the amount of missing data is significant.

```
In [8]: movies.isnull().sum()
Out[8]: budget                     0
        genres                     0
        homepage                3091
        id                         0
        keywords                   0
        original_language          0
        original_title             0
        overview                   3
        popularity                 0
        production_companies       0
        production_countries       0
        release_date               1
        revenue                    0
        runtime                    2
        spoken_languages           0
        status                     0
        tagline                  844
        title                      0
        vote_average               0
        vote_count                 0
        dtype: int64

In [12]: movies.dropna(inplace=True)
```

FIGURE 4.2.1

4.2.2 DIMENSIONALITY REDUCTION

Dimensionality reduction is a data pre-processing technique used in machine learning to reduce the number of features or variables in a dataset. It is often used when a dataset contains a large number of features, making it difficult to train a model efficiently or accurately.

```
In [8]: movies = movies[['movie_id','title','overview','genres','keywords','cast','crew']]
In [9]: movies.head()
```

Out[9]:

	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...

FIGURE 4.2.2

4.2.3 FEATURE ENGINEERING

This involves creating new features from existing features. For example, categorical features can be literal eval, or new features can be created by combining existing features. Feature engineering can improve the performance of a model by providing it with more relevant and informative features.

```
In [11]: def convert(text):
             L = []
             for i in ast.literal_eval(text):
                 L.append(i['name'])
             return L
```

```
In [13]: movies['genres'] = movies['genres'].apply(convert)
         movies.head()
```

Out[13]:

	movie_id	title	overview	genres	keywords	cast	crew
0	19995	Avatar	In the 22nd century, a paraplegic Marine is di...	[Action, Adventure, Fantasy, Science Fiction]	[{"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...	[Adventure, Fantasy, Action]	[{"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...	[Action, Adventure, Crime]	[{"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...	[Action, Crime, Drama, Thriller]	[{"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...	[Action, Adventure, Science Fiction]	[{"id": 818, "name": "based on novel"}, {"id":...	[{"cast_id": 5, "character": "John Carter", "c...	[{"credit_id": "52fe479ac3a36847f813eaa3", "de...

FIGURE 4.2.3

4.2.4 OUTLIER REMOVAL

Outlier removal is a common data pre-processing technique used in machine learning to handle extreme values that can skew the distribution of the data and affect the performance of the model. One way to remove outliers is by using the standard deviation and mean of the data.The provided data is pre-proccessed already and hence it didn't acquire any outlier.

4.2.5 REMOVE STOP WORDS

In Natural Language Processing (NLP) or text processing tasks, removing stopwords from text data is a common preprocessing step. Stopwords are generally common words in a language that do not contribute significantly to the meaning of the text. In Python, you can remove stopwords using libraries like NLTK

```
In [29]: new['tags'] = new['tags'].apply(lambda x: " ".join(x))
         new.head()
```

Out[29]:

	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...

```
In [30]: from sklearn.feature_extraction.text import CountVectorizer
         cv = CountVectorizer(max_features=5000,stop_words='english')
```

FIGURE 4.2.5

4.2.6 VECTORISATION

Vectorization in machine learning refers to the process of converting non-numeric data into numerical vectors or arrays that can be processed by machine learning models. It is a crucial step in preparing raw data for analysis or modeling.

In natural language processing (NLP) or text-based tasks, vectorization involves converting textual data (like words, sentences, or documents) into numerical representations that machine learning algorithms can understand. There are various techniques for vectorization, such as:

Bag-of-Words (BoW): This technique represents text as a sparse matrix where each row corresponds to a document, and each column corresponds to a unique word in the entire corpus. The values in the matrix indicate the frequency of each word in the document.

Term Frequency-Inverse Document Frequency (TF-IDF): TF-IDF is another method for vectorizing text, which considers not only the frequency of words but also their importance in the context of a document and the entire corpus.

Word Embeddings: Word embeddings, such as Word2Vec, GloVe, or FastText, convert words into dense, low-dimensional vectors by capturing semantic relationships between words. These representations often preserve semantic meaning and relationships between words.

```
In [30]: from sklearn.feature_extraction.text import CountVectorizer
         cv = CountVectorizer(max_features=5000,stop_words='english')

In [32]: vector = cv.fit_transform(new['tags']).toarray()

In [33]: vector.shape
Out[33]: (4806, 5000)
```

FIGURE 4.2.6

4.3 COSINE SIMILARITY

Cosine similarity is a measure used to determine how similar two vectors (arrays of numbers) are in a multi-dimensional space. It calculates the cosine of the angle between these vectors and is commonly used in various fields, including information retrieval, natural language processing, and recommendation systems.

In the context of machine learning and text processing, cosine similarity is often employed to assess the similarity between two documents or sets of features represented as vectors. Here's how cosine similarity is calculated:

Let's assume we have two vectors A and B representing the features of two documents, where each component of the vectors corresponds to a specific feature:

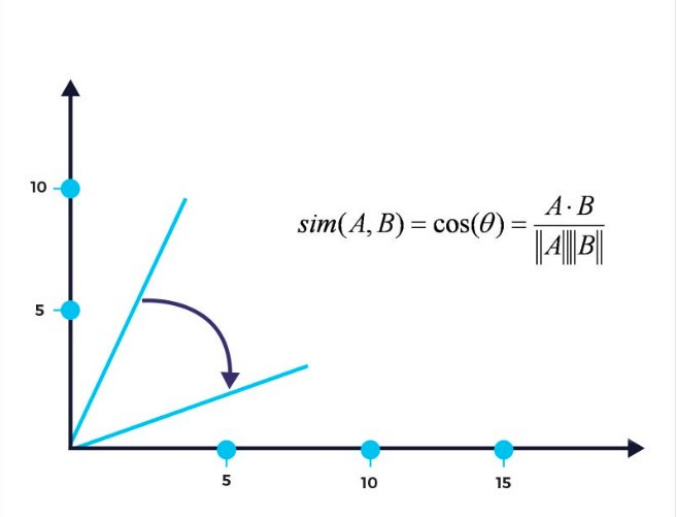

FIGURE 4.3.1

Where:

A·B represents the dot product of vectors A and B.

‖A‖ and ‖B‖ denote the Euclidean norms (or magnitudes) of vectors A and B, respectively.

The cosine similarity ranges between -1 and 1:

1 indicates that the vectors are in the same direction, i.e., perfectly similar.

0 means the vectors are orthogonal (perpendicular), indicating no similarity.

-1 signifies that the vectors are in opposite directions, i.e., perfectly dissimilar.

```
In [34]: from sklearn.metrics.pairwise import cosine_similarity

In [35]: similarity = cosine_similarity(vector)

In [36]: similarity

Out[36]: 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,
                 0.04774099],
                [0.0277885 , 0.        , 0.        , ..., 0.07352146, 1.        ,
                 0.05264981],
                [0.        , 0.        , 0.        , ..., 0.04774099, 0.05264981,
                 1.        ]])
```

FIGURE 4.3.2

4.4 DEPLOYMENT

Streamlit is a Python framework for building and deploying web applications effortlessly. With its simple API, developers can create interactive web apps using familiar Python scripting in just a few lines of code. It enables quick prototyping and deployment of data-driven apps, allowing easy integration of data visualizations, machine learning models, and interactive components, making the development process efficient and straightforward.

For the poster and movie name we have used TMDb API. The TMDb API (The Movie Database API) is a RESTful API that provides access to a vast amount of movie and TV show data. It allows developers to retrieve information about movies, TV shows, actors, genres, ratings, reviews, and more.

```
import pickle

import streamlit as st

import requests

import pandas as pd



def fetch_poster(movie_id):

url = "https://api.themoviedb.org/3/movie/{}?api_key=8265bd1679663a7ea12ac168da84d2e8&language=en-US".format(movie_id)

response = requests.get(url)

data = response.json()

#st.text("https://api.themoviedb.org/3/movie/{}?api_key=8265bd1679663a7ea12ac168da84d2e8&language=en-US".format(movie_id))

return "https://image.tmdb.org/t/p/w500/" + data['poster_path']
```

```
def recommend(movie):

movie_index = movies[movies['title'] == movie].index[0]

distances = similarity[movie_index]

movies_list = sorted(list(enumerate(distances)), reverse=True, key=lambda x: x[1])[1:6]

recommended_movie_names = []

recommended_movie_posters = []


for i in movies_list:

# fetch the movie poster

movie_id = movies.iloc[i[0]].movie_id

recommended_movie_posters.append(fetch_poster(movie_id))

recommended_movie_names.append(movies.iloc[i[0]].title)

return recommended_movie_names,recommended_movie_posters


st.header('Movie Recommender System')

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


movies = pd.DataFrame(movies_dict)


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


#movie_list = movies['title'].values


selected_movie = st.selectbox(

"Type or select a movie from the dropdown",

movies['title'].values
```

```
)

if st.button('Show Recommendation'):
recommended_movie_names, recommended_movie_posters = recommend(selected_movie)
columns = st.columns(5)

for i in range(5):
with columns[i]:
st.image(recommended_movie_posters[i])
columns[i].subheader(recommended_movie_names[i])
```

4.5 USER INTERFERENCE

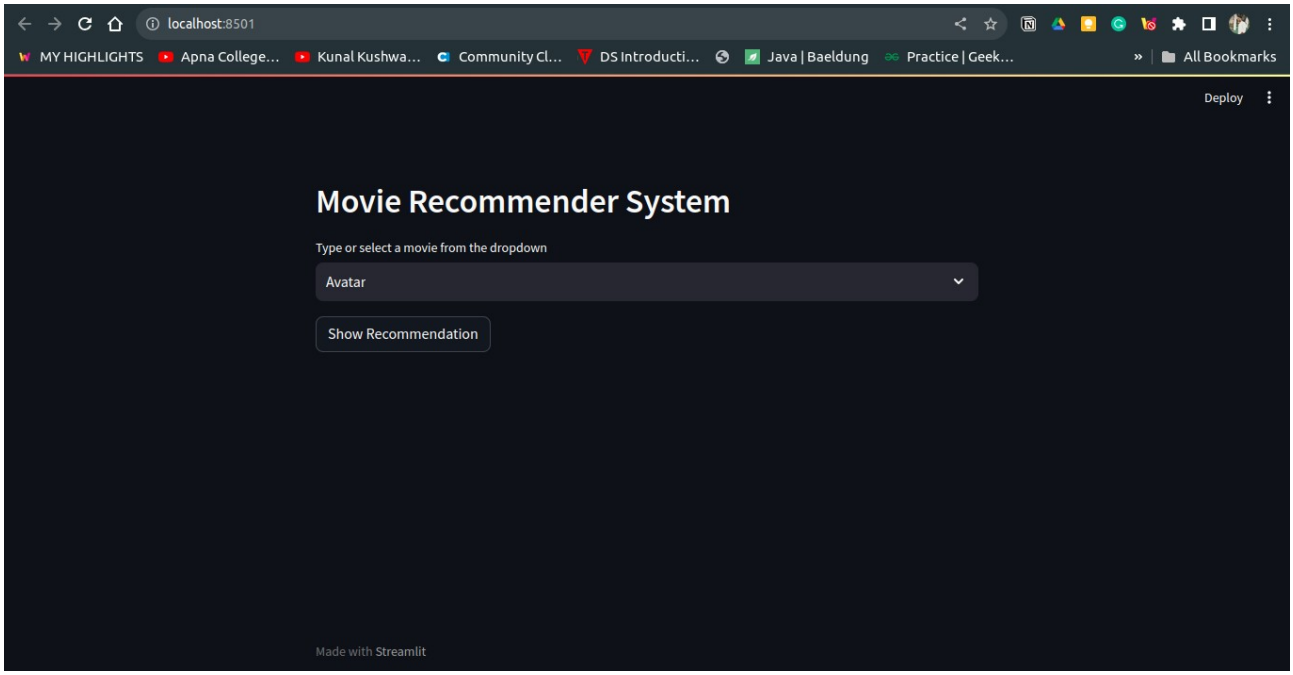

FIGURE 4.5.1

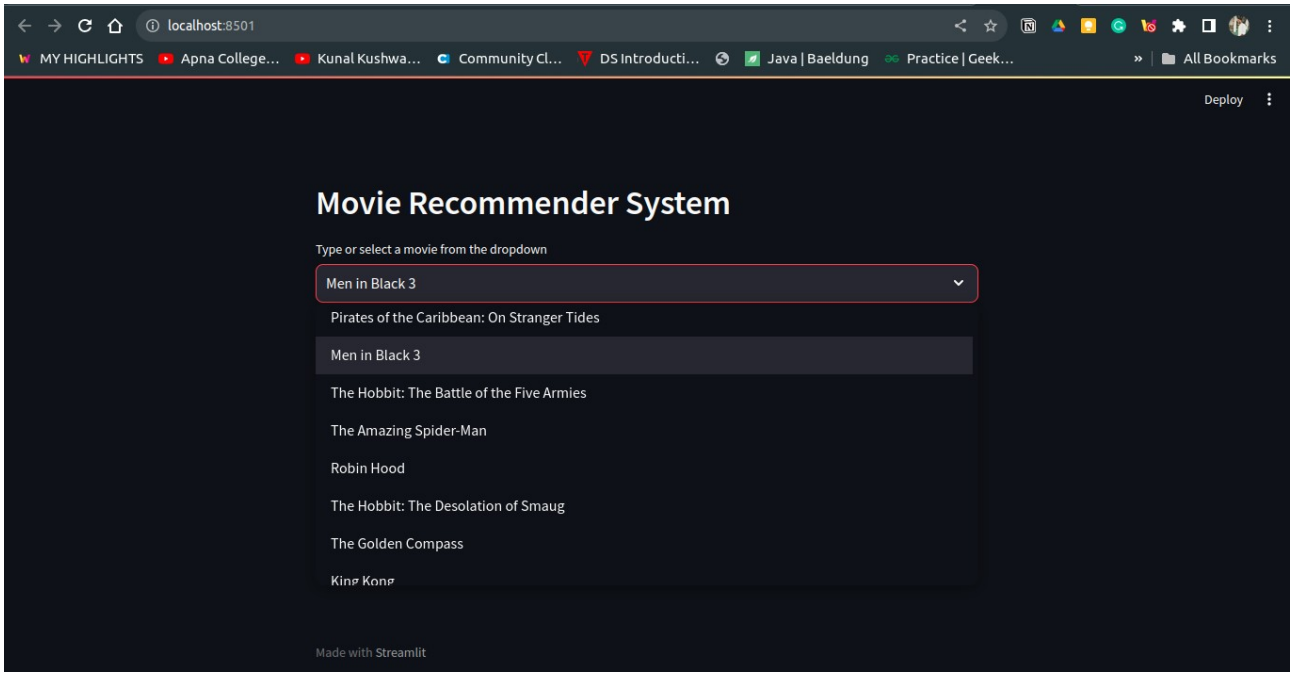

FIGURE 4.5.2

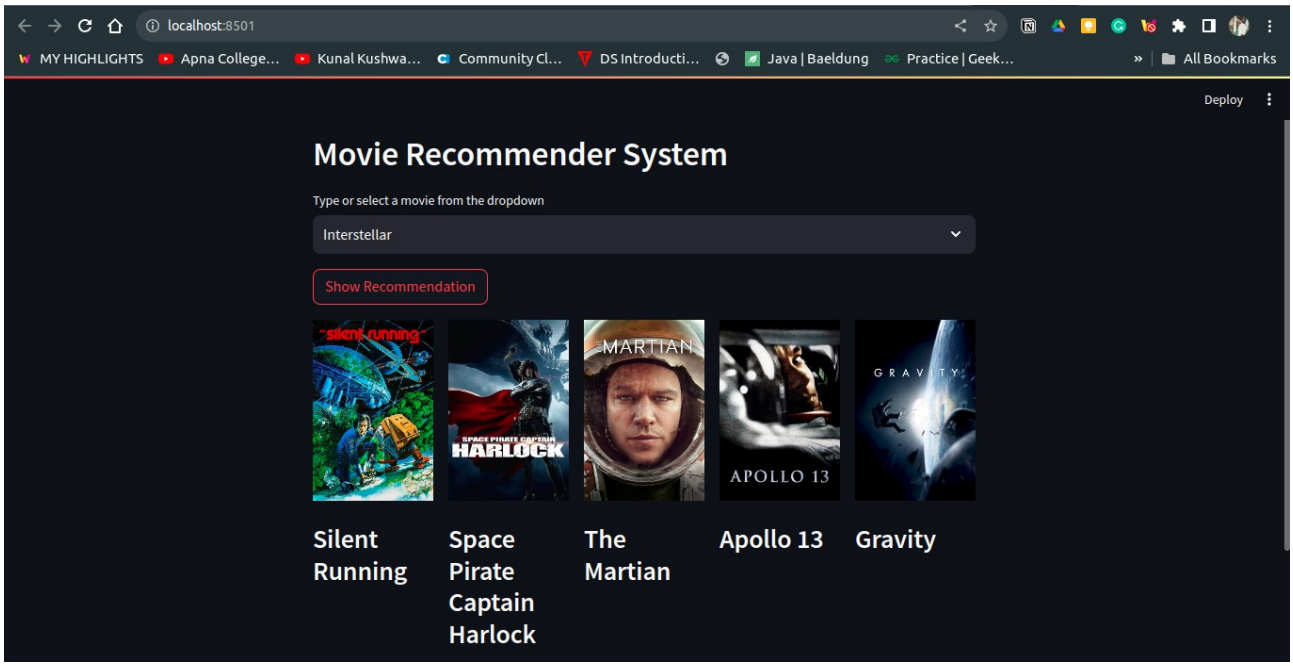

FIGURE 4.5.3

CHAPTER 5 : TECHNIQUES AND TOOLS

Technologies and tools refer to the various software, hardware, and other resources that are used to accomplish a specific task or achieve a particular goal. They are often used interchangeably, but generally, technologies refer to the broader range of resources and methods used to achieve a specific outcome, while tools refer to specific software applications, hardware devices, or other resources used to carry out specific tasks.

5.1 TECHNOLOGIES USED

- **Python**

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built-in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy-to- learn syntax emphasizes readability and therefore reduces the cost of program maintenance.

- **Pandas**

Panda is an open-source library designed primarily for working quickly and logically with relational or labeled data. It offers a range of data structures and procedures for working with time series and numerical data. The NumPy library serves as the foundation for this library. Pandas is quick and offers its users exceptional performance & productivity.

- **NumPy**

NumPy is the fundamental package for scientific computing in Python. It is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more.

- **Scikit-learn**

Scikit-learn, also known as sklearn, is a popular Python library for machine learning (ML) that provides a range of tools for data pre-processing, model selection, and evaluation. Scikit-learn is built on top of other popular scientific computing libraries, including NumPy, SciPy, and Matplotlib, and is designed to be easy to

use and flexible. Scikit-learn provides a wide range of ML algorithms, including linear regression, logistic regression, support vector machines (SVMs), decision trees, and random forests.

5.2 TOOLS USED

- **Jupyter Notebook**

Jupyter Notebook is a widely used open-source tool in data science and machine learning for interactive computing and data exploration. It offers an environment where users can create and share documents containing live code, equations, visualizations, and narrative text.

- **PyCharm:**

PyCharm is an integrated development environment (IDE) used for Python programming. PyCharm provides a range of features designed to help developers write and debug Python code more efficiently. Some of its key features include code completion, code analysis, debugging, and version control integration. PyCharm also provides a range of tools for web development, including support for popular web frameworks such as Django and Flask

- API

An API (Application Programming Interface) is a tool that lets different software applications communicate with each other. APIs are made up of sets of rules and protocols. They are often made up of different parts that act as tools or services for programmers. Here we have used TMDb API.

CHAPTER 6 : CONCLUSION

In conclusion, the movie recommendation system we have developed, rooted in content-based filtering and powered by vectorization techniques and cosine similarity, represents a significant advancement in the realm of personalized movie suggestions. By harnessing a dataset encompassing 5000 movies with 20 diverse attributes, this system adeptly employs content-based strategies to offer tailored recommendations based on inherent movie characteristics. The utilization of advanced vectorization methods and cosine similarity calculations facilitates the precise measurement of similarity between movies, enabling the system to provide users with movie suggestions aligned with their preferences.

The deployment of this recommendation system via Streamlit signifies a pivotal milestone in enhancing user accessibility and interaction. Streamlit's intuitive interface enables seamless navigation through the curated movie suggestions, offering users an engaging platform to explore and discover movies tailored to their tastes. The combination of content-based filtering, vectorization techniques, and Streamlit's deployment streamlines the movie discovery process, transforming how users engage with and discover movies.

As technology progresses and datasets expand, this recommendation system is poised to evolve further, offering even more refined and accurate movie suggestions. Through its reliance on content-based filtering, vectorization, and cosine similarity, this system not only simplifies the overwhelming choice of movies but also enriches the overall movie-watching experience for users, making it more personalized and enjoyable.

.

CHAPTER 7 : FUTURE WORK

Recommender system has developed for many years, which ever entered a low point. In the past few years, the development of machine learning, large-scale network and high performance computing is promoting new development in this field. We will consider the following aspects in future work.

• Use collaborative filtering recommendation.

After getting enough user data, collaborative filtering recommendation will be introduced. Collaborative filtering is based on the social information of users, which will be analyzed in the future research.

• Introduce more precise and proper features of movie.

Typical collaborative filtering recommendation use the rating instead of object features. In the future we should extract features such as color and subtitle from movie which can provide a more accurate description for movie.

• Introduce user dislike movie list.

The user data is always useful in recommender systems. In the future we will collect more user data and add user dislike movie list. We will input dislike movie list into the recommender system as well and generate scores that will be added to previous result. By this way we can improve the result of recommender system.

• Introduce machine learning.

For future study, dynamic parameters will be introduced into recommender system, we will use machine learning to adjust the weight of each feature automatically and find the most suitable weights.

• Make the recommender system as an internal service.

In the future, the recommender system is no longer a external website that will be just for testing. We will make it as an internal APIs for developers to invoke. Some movie lists in the website will be sorted by recommendation.

REFERENCES

https://www.kaggle.com/datasets/tmdb/tmdb-movie-metadata/

https://developers.google.com/machine-learning/recommendation/content-based/basics

https://www.javatpoint.com/

https://docs.streamlit.io/

https://developer.themoviedb.org/docs