Mini Project Report

On

Movie Recommendation System Using Machine Learning

2022-2023

Submitted to:

Dr Divesh Kumar

Assistant Professor

(CC-CSE-I-IV-Sem)

Submitted by:

Mayank Chuphal

University Roll-2118778

CSE-I-IV-Sem

Session :- 2022-23

Department Of Computer Science And Engineering

GRAPHIC ERA HILL UNIVERSITY, DEHRADUN

Certificate

Certified that Mayank Chuphal(Class Roll No : 45) has Developed a Mini Project on "Movie Recommendation System Using Machine Learning" for the CS 4th Semester Mini Project in Graphic Era Hill University , Dehradun. The Project Carried out by students is their own work as best of my knowledge.

Class Co-Ordinator

Dr. Divesh Kumar

CSE-I-4th Sem

(CSE-Dept.)

GEHU, Dehradun

ACKNOWLEDGEMENT

I wish to Thank my Parents for their Continuing Support and encouragement. I also wish to Thank Them for Providing the opportunity to reach this far in my studies.

I would like to thank particularly to our Class Co-Ordinator
Dr. Divesh Kumar and our Project Guide for her Patience, support and encouragement throughout the completion of this project and having faith in me.
At Last, but not the Least I greatly indebted to all other Persons who helped during this work.

Mayank Chuphal
Roll No :45
CSE–I–4th–Sem
Session: 2022 – 2023

Table of Content

Introduction

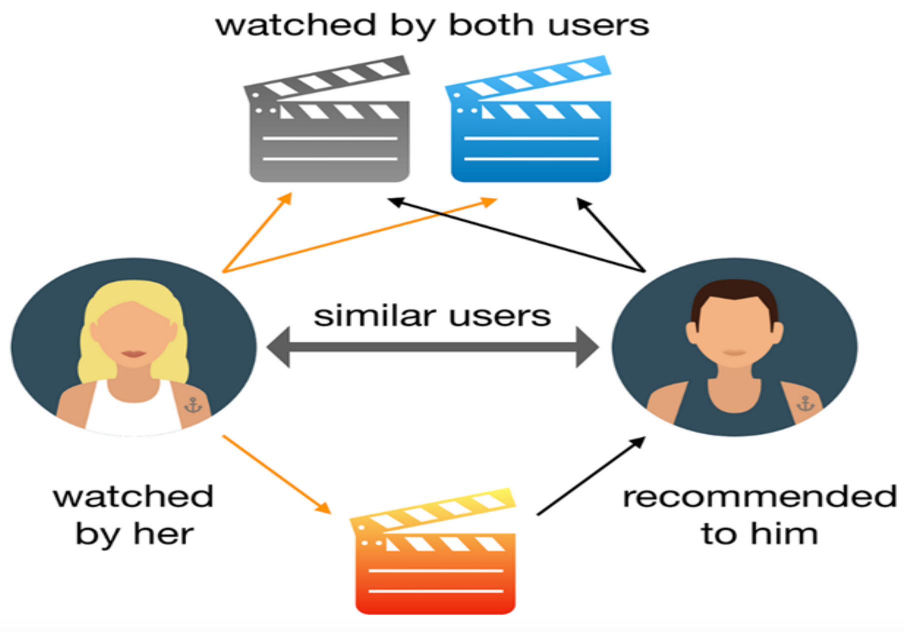
1.1-What is a Recommender System?

A recommender system is a simple algorithm whose aim is to provide the most relevant information to a user by discovering patterns in a dataset. The algorithm rates the items and shows the user the items that they would rate highly. An example of recommendation in action is when you visit Amazon and you notice that some items are being recommended to you or when Netflix recommends certain movies to you. They are also used by Music streaming applications such as Spotify and Deezer to recommend music that you might like.

1.2-Types of recommender systems

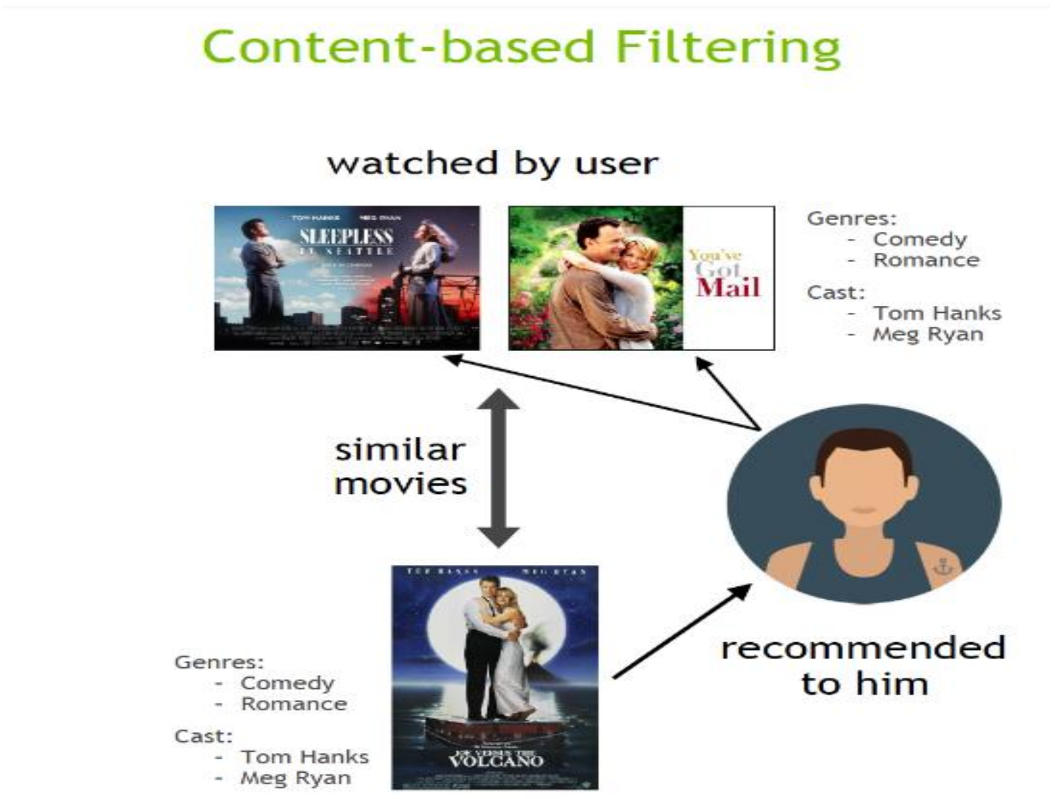
While there are a vast number of recommender algorithms and techniques, most fall into these broad categories: collaborative filtering, content filtering and context filtering.

(i) **Collaborative filtering** algorithms recommend items (this is the filtering part) based on preference information from many users (this is the collaborative part). This approach uses similarity of user preference behavior, given previous interactions between users and items, recommender algorithms learn to predict future interaction.



These recommender systems build a model from a user's past behavior, such as items purchased previously or ratings given to those items and similar decisions by other users. The idea is that if some people have made similar decisions and purchases in the past, like a movie choice, then there is a high probability they will agree on additional future selections. For example, if a collaborative filtering recommender knows you and another user share similar tastes in movies, it might recommend a movie to you that it knows this other user already likes.

(ii) Content filtering, by contrast, uses the attributes or features of an item (this is the content part) to recommend other items similar to the user's preferences. This approach is based on similarity of item and user features, given information about a user and items they have interacted with (e.g. a user's age, the category of a restaurant's cuisine, the average review for a movie), model the likelihood of a new interaction. For example, if a content filtering recommender sees you liked the movies You've Got Mail and Sleepless in Seattle, it might recommend another movie to you with the same genres and/or cast such as Joe Versus the Volcano.



(iii) **Hybrid recommender systems** combine the advantages of the types above to create a more comprehensive recommending system.

(iv) Context filtering includes users' contextual information in the recommendation process. This approach uses a sequence of contextual user actions, plus the current context, to predict the probability of the next action. In the Netflix example, given one sequence for each user—the country, device, date, and time when they watched a movie—they trained a model to predict what to watch next.

2-About project

2.1 What is Movie Recommendation System?

A movie recommendation system in machine learning (ML) is a system that suggests movies to users based on their preferences and behavior. It analyzes historical data such as user ratings, movie genres, actors, directors, and other relevant information to make personalized recommendations. The goal is to provide users with movie suggestions that are likely to align with their tastes and interests, enhancing their overall movie-watching experience.

Here's a simplified overview of how a movie recommendation system typically works:

1. Data Collection: The system gathers data from various sources, such as user ratings, movie metadata (genres, actors, directors, release dates), and possibly additional information like user demographics or reviews.

2. Preprocessing and Feature Extraction: The collected data is preprocessed to remove noise, handle missing values, and transform it into a suitable format for ML algorithms. Features like movie genres, actors, and directors may be extracted or encoded to represent relevant characteristics.

3. Building User Profiles: User profiles are created by analyzing the historical data of each user, including their movie preferences, ratings, and possibly demographic information. This helps in understanding user preferences and forming the basis for personalized recommendations.

4. Training ML Models: Different ML algorithms can be used to train models on the collected data. Popular techniques include collaborative filtering, content-based

filtering, and hybrid approaches that combine multiple methods. These models learn patterns and relationships between users and movies to make predictions about user preferences.

5. Evaluation and Validation: The trained models are evaluated using metrics such as precision, recall, or mean average precision to assess their performance and accuracy in predicting user preferences. Cross-validation techniques may be employed to validate the models.

6. Generating Recommendations: Once the models are trained and validated, they can generate personalized movie recommendations for users. These recommendations can be based on a user's browsing history, explicit ratings, or a combination of both.

7. Continuous Learning and Feedback: Recommendation systems often incorporate feedback loops, where user interactions and new data are continually gathered to improve the accuracy of future recommendations. This feedback helps the system adapt to changing user preferences over time.

Overall, a movie recommendation system in ML leverages historical data, user preferences, and various algorithms to provide personalized movie suggestions, enhancing user engagement and satisfaction.

2.2 Project Flow

In this project we are using content based filtering to make our recommendation system.So our main aim is to create tags from the dataset which we are using here. So the various steps involved in making this project are as follows :

1.Getting the data: First of all we need a dataset to work on.We can get that dataset from Kaggle and we will convert it into a dataframe.

2.Preprocessing and feature selection: It means preparing the data to give it to algorithm.Since there are many rows and columns in our dataframe and some of these are irrevalant, so in this step we will only extract the relevant data by reducing the number of rows and columns.

3.Model Training: In this step we will create our machine model and train it with the required data.

4.Creating Website: So we created the model in above step now in this step we will convert this model into a website using streamlit,flask etc.

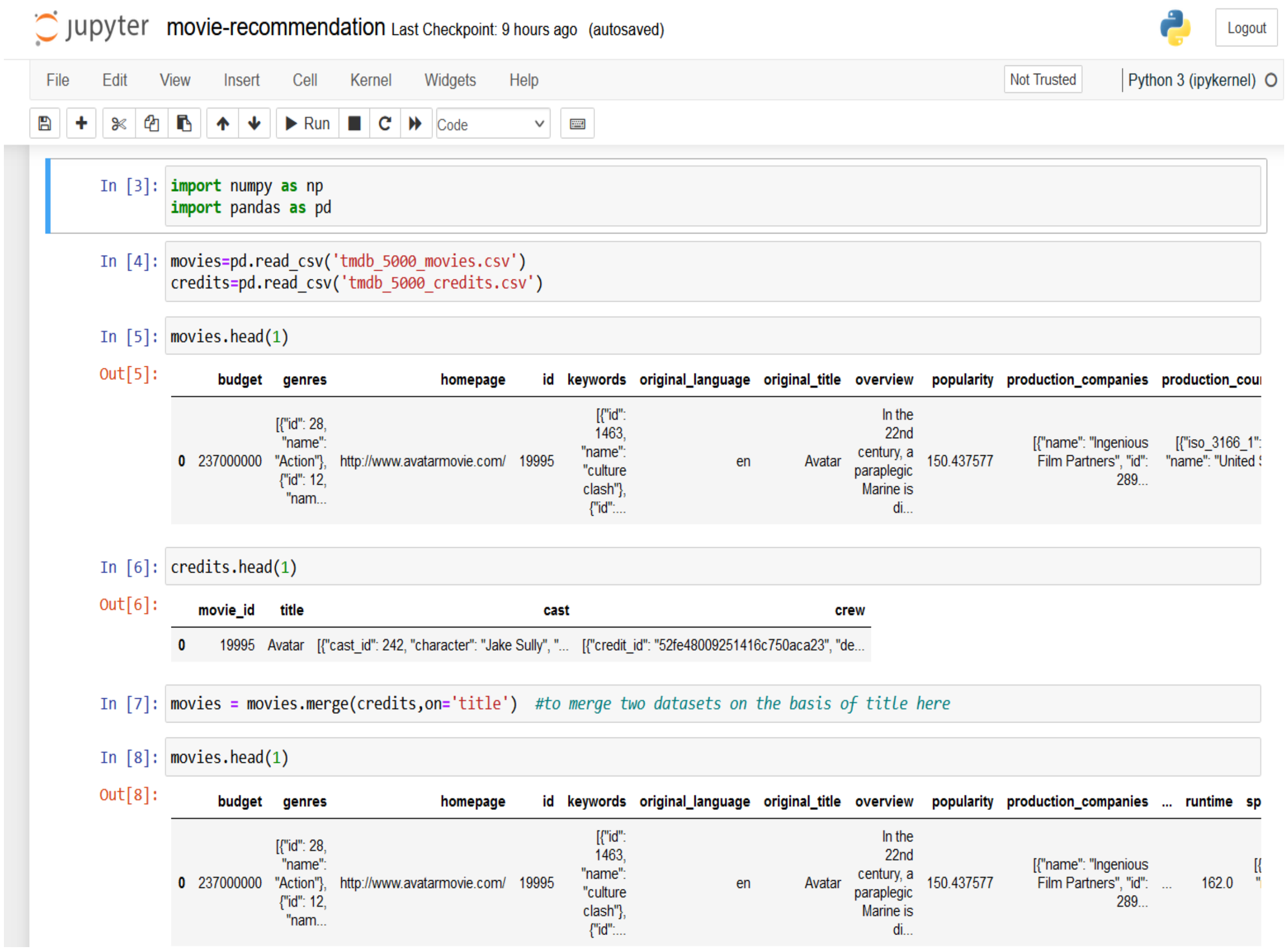
5.Deploy the model: After integrating it on the website we will deploy the website on server.

3. Tools used in this project

3.1 Requirements and Libraries

(i) Jupyter Notebook

The Jupyter Notebook is the original web application for creating and sharing computational documents. It offers a simple, streamlined, document-centric experience.We have written our code for extracting data,preprocessing and model training in jupyter notebook.



```
In [3]: import numpy as np
        import pandas as pd

In [4]: movies=pd.read_csv('tmdb_5000_movies.csv')
        credits=pd.read_csv('tmdb_5000_credits.csv')

In [5]: movies.head(1)

Out[5]:
```

	budget	genres	homepage	id	keywords	original_language	original_title	overview	popularity	production_companies	production_cou
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": "name": "United

```
In [6]: credits.head(1)

Out[6]:
```

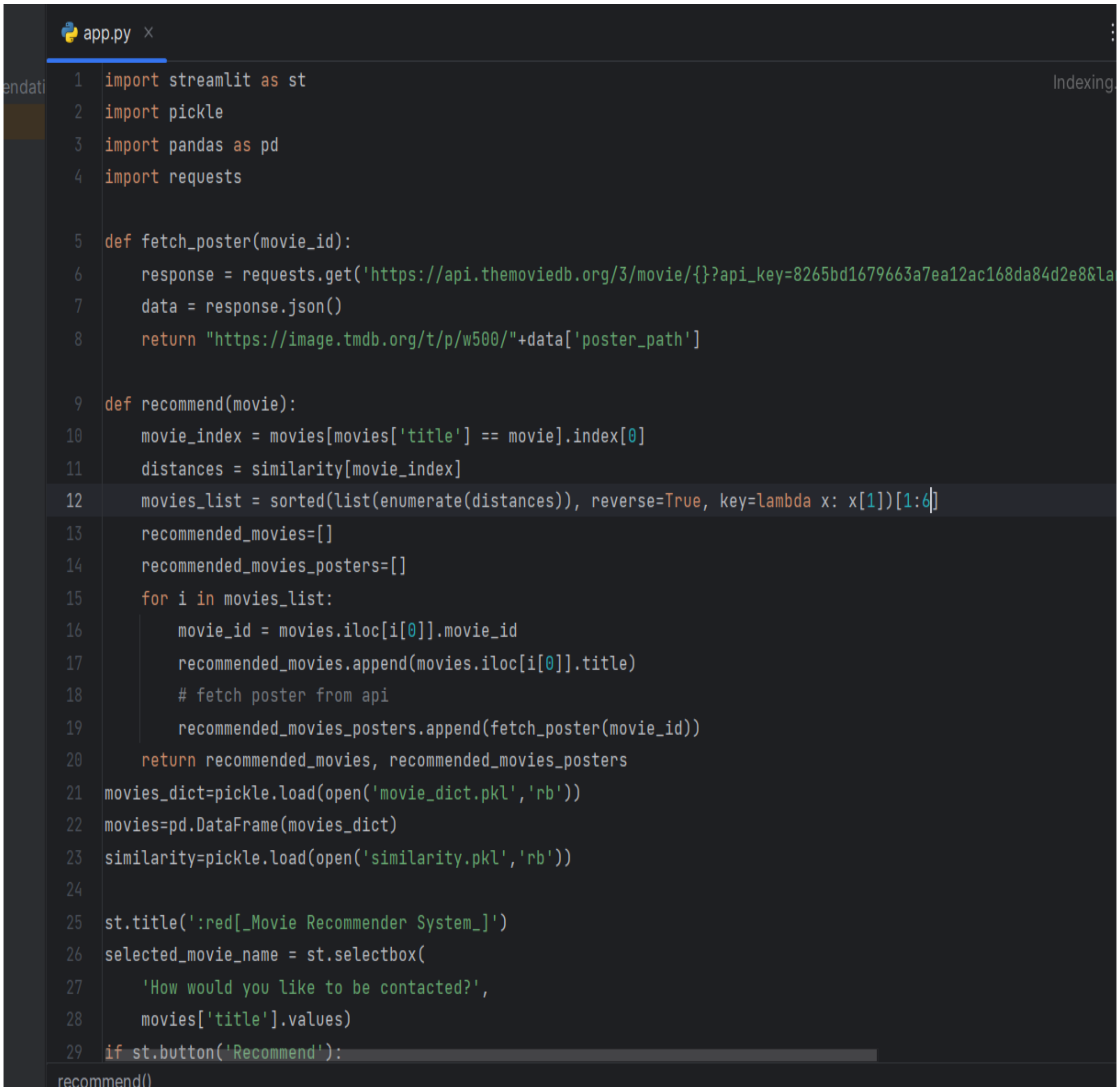
	movie_id	title	cast	crew
0	19995	Avatar	[{"cast_id": 242, "character": "Jake Sully", "...	[{"credit_id": "52fe48009251416c750aca23", "de...

```
In [7]: movies = movies.merge(credits,on='title')  #to merge two datasets on the basis of title here

In [8]: movies.head(1)

Out[8]:
```

	budget	genres	homepage	id	keywords	original_language	original_title	overview	popularity	production_companies	...	runtime	sp
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...	...	162.0	[{ "

(ii) PyCharm

PyCharm is an integrated development environment (IDE) used for programming in python.It provides code analysis , graphical debugger,an integrated unit tester and supports web development with Django.
We used pycharm to create a project and used our model in it by using pickle library and create our website by using streamlit library.



```
import streamlit as st
import pickle
import pandas as pd
import requests

def fetch_poster(movie_id):
    response = requests.get('https://api.themoviedb.org/3/movie/{}?api_key=8265bd1679663a7ea12ac168da84d2e8&la
    data = response.json()
    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_movies=[]
    recommended_movies_posters=[]
    for i in movies_list:
        movie_id = movies.iloc[i[0]].movie_id
        recommended_movies.append(movies.iloc[i[0]].title)
        # fetch poster from api
        recommended_movies_posters.append(fetch_poster(movie_id))
    return recommended_movies, recommended_movies_posters
movies_dict=pickle.load(open('movie_dict.pkl','rb'))
movies=pd.DataFrame(movies_dict)
similarity=pickle.load(open('similarity.pkl','rb'))

st.title(':red[_Movie Recommender System_]')
selected_movie_name = st.selectbox(
    'How would you like to be contacted?',
    movies['title'].values)
if st.button('Recommend'):
```

(iii) Libraries Used

Libraries includes Pandas for reading csv files and working with them, Numpy for creating ,merging dataframes and extracting data from it.
nltk for stemming purpose.
ast for converting string of list to lists.
Sklearn for feature extraction and vectorization.
Pickle is used for serialization of object,saving machine learning algorithm.
Streamlit, Pickle libraries are used in making the GUI(Graphical User Interface).

3. Other Tools used for project

Requests module is used to send HTTP requests using Python.We used requests module to get response data. We used it to get the poster path and image path of a movie.
Kaggle is used for getting data set.
TMDB platform is used to get an api that provides movie details by giving movie id.
We have also create some lambda functions with functions like replace(),lower(),join().

4.   Output

4.1 Main Idea(How does this project works?)

In this project we used a data set and done some preprocessing on it and make a dataframe of only relevant columns like movie_id,crew,title etc.

From this data we created another column of tags which is a combination of keywords,cast,crew,genres,overview. As there were many duplicate words in tags we removed them by doing stemming.

Now to recommend movies we have movie_id, name and tags.But we cannot directly use tags to recommend a movie because machine only understands numeric values.So we converted these tags into vectors by doing text vectorization.
We have 4806 movies here so we created 4806 vectors and based on these vectors we can now recommend other vectors. Since each movie is vector now suppose a person likes a movie(vector) then we will recommend that person all the other vectors which are closest to the liked vector.

For text vectorization we have use bag of words technique here in this technique we will combine all the tags of movies to form a large text.
From this text we will find 5000 most common words.
Now we will check how many times these 5000 words are occurring in each movie.

For example:suppose in movie 1 word 1 occurs 5 times, word 2 occurs 3 times and so on.

This will create a various rows of count of each words in each movie.These rows are vectors.

Since we have vectors now.

We have 4806 movies this means we have 4806 vectors also.

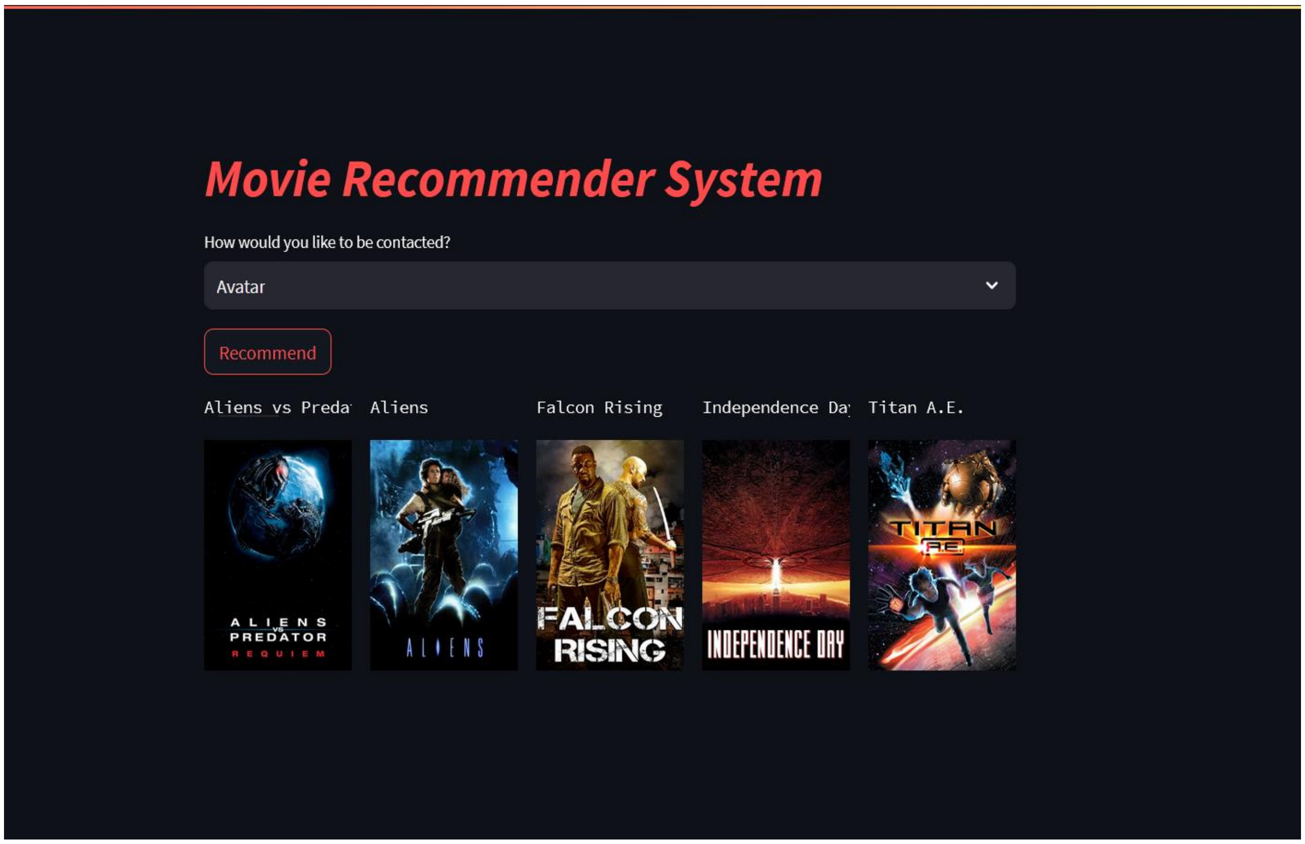
Each vector have 5000 numbers.

Each movie is a vector and we have to calculate distance for each movie with every other movie.

More distance means less similarity and less distance means more similarity we will not calculate euclidian distance (underroot(x2-x1)^2+(y2-y1)^2) instead we will calculate the angle between them i.e cosine distance.

This is the main idea behind this project.

GUI:



5. Conclusion

We have successfully developed our project on Movie Recommendation System.

In this, we have taken use of Jupyter Notebook and Python, other platforms like PyCharms and streamlit for the effective use of the model.

Goals-

- To make it more effective use of the model.

REFERENCE

1. **StackOverflow**– where Developers learn, share and create careers.
2. **GeeksForGeeks**– A computer Science Portal for geeks.
3. **StreamLitDocumentation**– for building website.
4. **Youtube(Campusx)–** for learning and knowledge.
5. **W3schools** – for machine learning.