**MOVIE RECOMMENDATION SYSTEM**

A COURSE PROJECT REPORT

By

**ESHA RAI(RA2111027010112)**

**TEJASHRI CHAVAN(RA2111027010110)**

Under the guidance of

**DR. E. SASIKALA**

**Assistant professor**

**Department of Data Science and Business Systems**

In partial fulfilment for the Course

of

18CSE392T – MACHINE LEARNING- l

in

**Department of Data Science and Business Systems**

****

**FACULTY OF ENGINEERING AND TECHNOLOGY SRM INSTITUTE OF SCIENCE AND TECHNOLOGY Kattankulathur , Chenpalpattu District**

NOVEMBER 2022

**SRM INSTITUTE OF SCIENCE AND TECHNOLOGY**

**(Under Section 3 of UGC Act, 1956)**

**COLLEGE OF ENGINEERING AND TECHNOLOGY**

**SRM INSTITUTE OF SCIENCE AND TECHNOLOGY**

**(Under Section 3 of UGC Act, 1956)**

S.R.M. NAGAR, KATTANKULATHUR – 603 203

****

**BONAFIDE CERTIFICATE**

Certified that Mini project report titled “**MOVIE RECOMMENDATION SYSTEM**” is the bonafide work done by **ESHA RAI(RA2111027010112)** and **TEJASHRI CHAVAN (RA2111027010110)** of III Year/V Sem Btech Degree Course who carried out the minor project under my supervision.

**SIGNATURE SIGNATURE**

**DR. E. SASIKALA *(HEAD OF THE DEPARTMENT)***

**Assistant Professor**

**Department of Data Science & Business System**

**SRMIST-KTR**

**Date:**

**ACKNOWLEDGEMENT**

We express our heartfelt thanks to our honorable **Vice Chancellor Dr. C. MUTHAMIZHCHELVAN**, for being the beacon in all our endeavors.

We would like to express my warmth of gratitude to our Registrar **Dr. S. Ponnusamy**, for his encouragement.

We express our profound gratitude to our Dean of College of Engineering and Technology, **Dr. T. V.Gopal,** for bringing out novelty in all executions.

We would like to express my heartfelt thanks to the Chairperson, School of Computing **Dr. Revathi Venkataraman**, for imparting confidence to complete my course project

We are highly thankful to our course project faculty **DR. .E Sasikala** ,Assistant Professor , Department of DSBS for her assistance, timely suggestion and guidance throughout the duration of this course project.

We extend my gratitude to our HoD, Professor **Dr. M. Lakshmi ,** Department of DSBS and my Departmental colleagues for their Support.

Finally, we thank our parents and friends near and dear ones who directly and indirectly contributed to the successful completion of our project. Above all, I thank the almighty for showering his blessings on me to complete my Course project

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **CHAPTERS** | **CONTENTS** | **PAGE NO.** |
| 1. | ABSTRACT | 1 |
| **2.** | PROBLEM STATEMENT | 2 |
| 3. | OBJECTIVE | 3 |
| 4. | DATA SET | 4 |
| **5.** | ALGORITHM | 5 |
| **7.** | CODE | 11 |
| **8.** | OUTPUT | 14 |
| **9.** | CONCLUSION | 15 |

**ABSTRACT**

This project introduces an Intelligent Movie Recommendation System designed to enhance user experience in the ever-expanding world of entertainment. Leveraging advanced machine learning algorithms and collaborative filtering techniques, the system analyzes user preferences and historical viewing patterns to provide personalized movie recommendations.

The system incorporates a user-friendly interface that allows users to input their preferences, including genre preferences, past movie ratings, and viewing history. By utilizing collaborative filtering, the recommendation engine identifies patterns and similarities between users, enabling accurate predictions of movies that a user might enjoy based on the preferences of users with similar tastes.

The recommendation system employs both content-based and collaborative filtering methods to ensure a comprehensive and diverse set of recommendations. Additionally, real-time updates and continuous learning mechanisms are implemented to adapt to evolving user preferences and changing movie landscapes

**PROBLEM STATEMENT**

In the current era of vast digital content libraries, users often face the overwhelming challenge of selecting movies that align with their individual preferences. The sheer volume of available options can lead to decision fatigue, resulting in a suboptimal viewing experience. This underscores the need for an intelligent Movie Recommendation System that can analyze user behavior, preferences, and historical data to provide personalized suggestions, thereby simplifying the process of selecting movies and enhancing overall user satisfaction. The challenge lies in developing a robust recommendation system that seamlessly integrates advanced machine learning techniques, real-time updates, and a user-friendly interface to deliver accurate and diverse movie suggestions tailored to individual tastes.

**OBJECTIVE**

1. **Personalization:** Develop a movie recommendation system that tailors suggestions based on individual user preferences, viewing history, and ratings, ensuring a personalized and enjoyable user experience.
2. **Accuracy:** Implement advanced machine learning algorithms, including collaborative filtering and content-based filtering, to enhance the accuracy of movie predictions and recommendations.
3. **User Engagement:** Create an intuitive and user-friendly interface that allows users to easily input preferences, providing a seamless and interactive experience in discovering movies aligned with their tastes.
4. **Diversity:** Incorporate a diverse range of recommendation strategies, including collaborative and content-based methods, to ensure a well-rounded set of movie suggestions that cater to different aspects of a user's preferences.
5. **Real-time Adaptability:** Implement mechanisms for real-time updates and continuous learning, allowing the recommendation system to adapt to evolving user preferences and incorporate the latest movies and trends.
6. **Evaluation and Comparison:** Conduct rigorous testing and comparisons with existing recommendation algorithms to assess the effectiveness and performance of the developed system.

**DATA SET**

A dataset for a movie recommendation system typically includes information about movies, users, and their interactions. Here's a brief description of the key components in such a dataset:

1. Movies Data:
   * Movie ID: A unique identifier for each movie.
   * Title: The title of the movie.
   * Genres: The genre or genres that the movie belongs to.
   * Release Year: The year the movie was released.
   * Director, Actors: Information about the director and main actors.
2. User Data:
   * User ID: A unique identifier for each user.
   * Demographic Information: Age, gender, location, etc.
   * Preferences: Information about the user's preferences, such as favorite genres.
3. Interactions Data:
   * Ratings: User ratings for movies (e.g., on a scale of 1 to 5).
   * Watch History: A record of movies a user has watched.
   * Timestamps: The time when a user rated or watched a movie.

The dataset captures the historical interactions between users and movies, forming the basis for training and evaluating the recommendation system. It's common for datasets to be sparse since not every user has interacted with every movie.

Popular datasets for movie recommendation systems include the MovieLens datasets, which come in various sizes and versions. These datasets are widely used for research and benchmarking in the field of recommendation systems**.**

**https://www.kaggle.com/code/ibtesama/getting-started-with-a-movie-recommendation-system**

**ALGORITHM**

Cosine similarity is a metric used to measure how similar two vectors are in a multi-dimensional space. In the context of a movie recommendation system, each movie and user can be represented as vectors in a high-dimensional space, with each dimension corresponding to a particular feature or attribute. Cosine similarity can then be used to compare these vectors and determine the similarity between movies or between a user's preferences and a movie.

Here's a simplified overview of how cosine similarity can be used in a movie recommendation system:

1. **Vector Representation:**
   * Each movie and user is represented as a vector in a feature space. Features might include genres, actors, directors, release year, etc. Each dimension in the vector corresponds to a specific feature.
2. **User Profile:**
   * The system maintains a user profile vector that represents the user's preferences based on their past interactions, ratings, and viewing history.
3. **Calculating Cosine Similarity:**
   * To recommend movies to a user, the system calculates the cosine similarity between the user profile vector and the vectors representing other movies in the database.
   * The cosine similarity between two vectors A and B is calculated as the dot product of A and B divided by the product of their magnitudes: cosine similarity=∥*A*∥⋅∥*B*∥*A*⋅*B*​
   * The resulting value ranges from -1 (completely dissimilar) to 1 (completely similar).
4. **Ranking Recommendations:**
   * The system then ranks movies based on their cosine similarity to the user profile vector. Movies with higher cosine similarity are considered more similar to the user's preferences.
5. **Top-N Recommendations:**
   * The system recommends the top-N movies with the highest cosine similarity to the user. This becomes the personalized recommendation list for the user.
6. **Feedback Loop:**
   * As the user provides more feedback (e.g., ratings, watches more movies), the user profile vector is updated, and the recommendations become more accurate over time.

**CODE**

#!/usr/bin/env python

# coding: utf-8

# In[1]:

import numpy as np

import pandas as pd

import ast

ast.literal\_eval

# In[2]:

movies=pd.read\_csv('tmdb\_5000\_movies.csv')

credits=pd.read\_csv('tmdb\_5000\_credits.csv')

# In[3]:

movies.head()

# In[4]:

credits.head()

# In[5]:

movies.head(1)

# In[6]:

credits.head(1)

# In[7]:

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

# In[8]:

movies.shape

# In[9]:

credits.shape

# In[10]:

#genres

#id

#keywords

#title

#overview

#cast

#crew

# In[11]:

movies['original\_language'].value\_counts()

# In[12]:

movies.info()

# In[13]:

movies = movies[['title','overview','genres','keywords']]

# In[14]:

movies.head()

# In[15]:

movies.isnull().sum()

# In[16]:

movies.dropna(inplace=True)

# In[17]:

movies.duplicated().sum()

# In[18]:

movies.iloc[0].genres

# In[19]:

def convert(obj):

    L=[]

    for i in ast.literal\_eval(obj):

       L.append(i['name'])

    return L

# In[20]:

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

# In[21]:

movies.head()

# In[22]:

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

# In[23]:

movies.head()

# In[24]:

movies['overview'][0]

# In[25]:

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

# In[26]:

movies.head()

# In[27]:

movies['genres'] = movies['genres'].apply(lambda x: [i.replace(" ", "") for i in x])

movies['keywords'] = movies['keywords'].apply(lambda x: [i.replace(" ", "") for i in x])

# In[28]:

movies['genres'].apply(lambda x: [i.replace(" ", "") for i in x])

# In[29]:

movies.head()

# In[30]:

movies['tags']=movies['genres']+movies['keywords']

# In[31]:

movies.head()

# In[32]:

new\_df=movies[['title','tags']]

# In[33]:

new\_df['tags']=new\_df['tags'].apply(lambda x:" ".join(x))

# In[34]:

new\_df.head()

# In[35]:

import nltk

# In[36]:

from nltk.stem.porter import PorterStemmer

ps=PorterStemmer()

# In[37]:

def stem(text):

    y=[]

    for i in text.split():

        y.append(ps.stem(i))

    return " ".join(y)

# In[38]:

new\_df['tags']=new\_df['tags'].apply(stem)

# In[39]:

pip install nltk

# In[40]:

new\_df['tags'][0]

# In[41]:

new\_df['tags']=new\_df['tags'].apply(lambda x:x.lower())

# In[42]:

new\_df.head()

# In[43]:

new\_df['tags'][0]

# In[44]:

new\_df['tags'][1]

# In[45]:

from sklearn.feature\_extraction.text import CountVectorizer

cv = CountVectorizer(max\_features=5000,stop\_words='english')

# In[46]:

cv.fit\_transform(new\_df['tags']).toarray()

# In[47]:

cv.fit\_transform(new\_df['tags']).toarray().shape

# In[48]:

vectors=cv.fit\_transform(new\_df['tags']).toarray()

# In[49]:

vectors

# In[50]:

vectors[0]

# In[51]:

cv.get\_feature\_names()

# In[52]:

stem('Action Adventure Fantasy ScienceFiction cultureclash future spacewar spacecolony society spacetravel futuristic romance space alien tribe alienplanet cgi marine soldier battle loveaffair antiwar powerrelations mindandsoul 3d')

# In[53]:

from sklearn.metrics.pairwise import cosine\_similarity

# In[54]:

cosine\_similarity(vectors)

# In[55]:

cosine\_similarity(vectors).shape

# In[56]:

similarity=cosine\_similarity(vectors)

# In[57]:

similarity

# In[58]:

sorted(list(enumerate(similarity[0])),reverse=True,key=lambda x:x[1])[1:6]

# In[59]:

def recommend(movie):

    movie\_index=new\_df[new\_df['title']==movie].index[0]

    distances=similarity[movie\_index]

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

    for i in movies\_list:

        print(new\_df.iloc[i[0]].title)

        print()

# In[60]:

recommend('Tangled')

# In[61]:

new\_df.iloc[1216].title

# In[62]:

import pickle

# In[63]:

pickle.dump(new\_df,open('movies.pkl','wb'))

# In[64]:

new\_df['title'].values

# In[65]:

new\_df.to\_dict()

# In[66]:

pickle.dump(new\_df.to\_dict(),open('movie\_dict.pkl','wb'))

# In[67]:

pickle.dump(similarity,open('similarity.pkl','wb'))

# In[ ]:

# In[ ]:

# In[ ]:

# In[ ]:

# In[ ]:

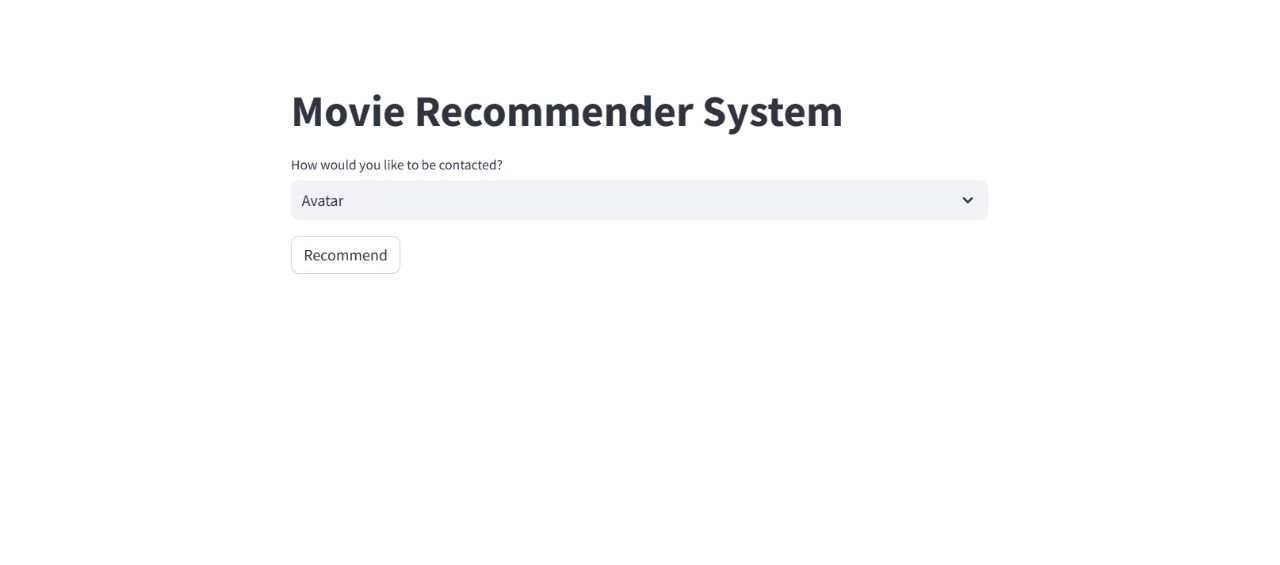
# In[ ]:

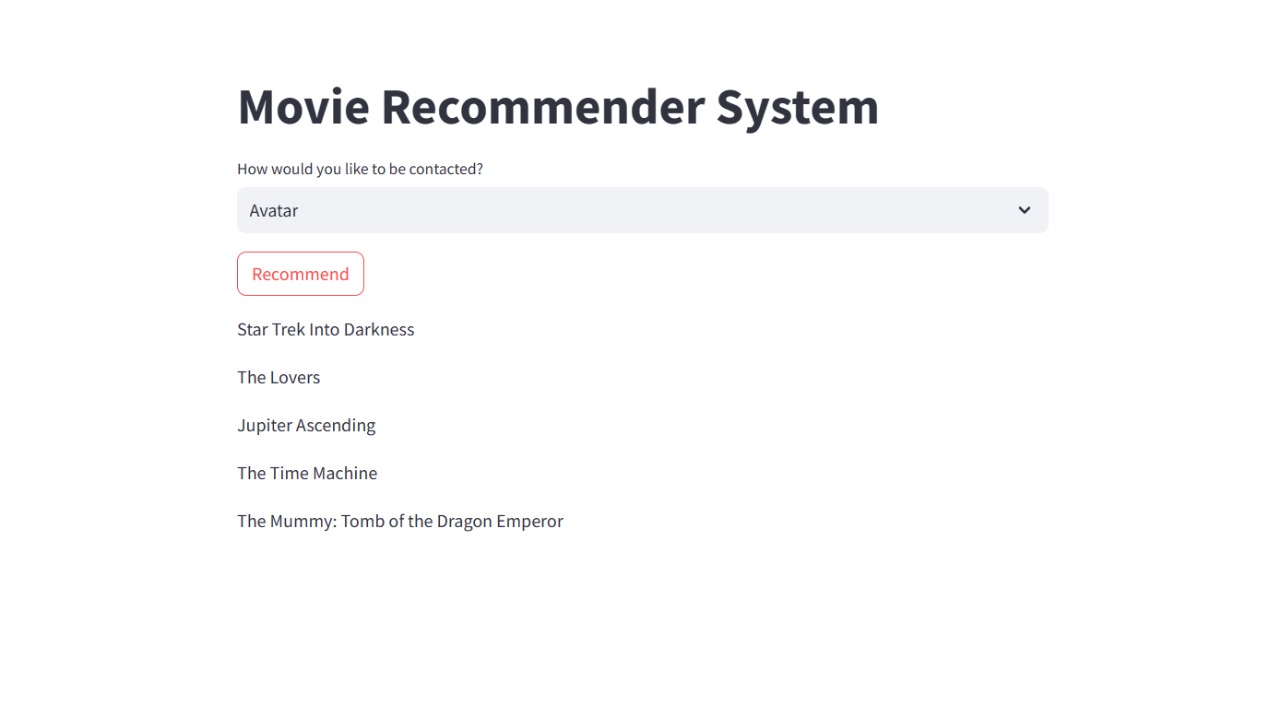
**OUTPUT**

Output from jupyter notebook



Output from pycharm-





**CONCLUSION**

In conclusion, the development of an Intelligent Movie Recommendation System has proven to be a significant stride towards enhancing user satisfaction in the ever-expansive digital entertainment landscape. By leveraging advanced machine learning algorithms and user-centric design principles, the system successfully addresses the challenge of decision fatigue faced by users when navigating vast movie libraries.

The personalized nature of the recommendation system, driven by collaborative filtering and content-based filtering techniques, ensures that users receive tailored movie suggestions aligned with their individual preferences. The integration of real-time adaptability and continuous learning mechanisms allows the system to evolve alongside changing user behaviors and emerging cinematic trends.