

“Movie Recommendation System”

OBJECTIVE:

This movie Recommendation system is created by the help of Machine Learning and Python. The features of the movie Recommendation system are Predict the similar movie .

TECH STACK USED:

Python

Python is a high-level, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation. Python is dynamically-typed and garbage-collected. It supports multiple programming paradigms, including structured, object-oriented and functional programming

Machine Learning

Machine Learning is the field of study that gives computers the capability to learn without being explicitly programmed. ML is one of the most exciting technologies that one would have ever come across. As it is evident from the name, it gives the computer that makes it more similar to humans: The ability to learn. Machine learning is actively being used today, perhaps in many more places than one would expect.

PROJECT DESCRIPTION:

The movie Recommendation system is built using Machine Learning Support. It takes the input a movie name and predicts the similar movie using movie category.

PROJECT GITHUB LINK:

https://github.com/Dhananjay17728/Movie_recommndation_sysstem

Screenshot

Movie Recommender System

Serch the movie

Avatar

Recommend

Aliens vs Preda Aliens Falcon Rising Independence Da Titan A.E.



```
File Edit Selection View Go Run Terminal Help app.py - Movie_recommndation_sysstem - Visual Studio Code
EXPLORER
MOVIE_RECOMMENDATIO...
  .ipynb_checkpoints
  Movie Recommend...
  app.py
  archive.zip
  gitattributes
  Movie Recommendation...
  movie_dict.pkl
  movies.pkl
  similarity.pkl
  tmdb_5000_credits.csv
  tmdb_5000_movies.csv
app.py
1 import streamlit as st
2 import pickle
3 import pandas as pd
4 import requests
5
6 def fetch_poster(movie_id):
7     response=requests.get('https://api.themoviedb.org/3/movie/{}?api_key=3d0c52cde153f7335519fce778b238e1&language=en-US'.f
8     data=response.json()
9     print("https://image.tmdb.org/t/p/w500"+data['poster_path'])
10    return "https://image.tmdb.org/t/p/w500"+data['poster_path']
11
12 def recommend(movie):
13     movie_index=movies[movies['title']==movie].index[0]
14     distances=similarity[movie_index]
15     movies_list=sorted(list(enumerate(distances)),reverse=True,key=lambda x:x[1])[1:6]
16     recommended_movies=[]
17     recommended_movies_poster=[]
18     for i in movies_list:
19         movie_id=movies.iloc[i[0]].movie_id
20         recommended_movies.append(movies.iloc[i[0]].title)
21         recommended_movies_poster.append(fetch_poster(movie_id))
22     return recommended_movies,recommended_movies_poster
23
24 movies_dict=pickle.load(open('movie_dict.pkl','rb'))
25 movies=pd.DataFrame(movies_dict)
26 similarity=pickle.load(open('similarity.pkl','rb'))
27 st.title('Movie Recommender System')
28 selected_movie_name=st.selectbox(
29     'Serch the movie ',
30     movies['title'].values
31 )
32
33 if st.button('Recommend'):
34     names,poster=recommend(selected_movie_name)
35     col1,col2,col3,col4,col5=st.columns(5)
```

