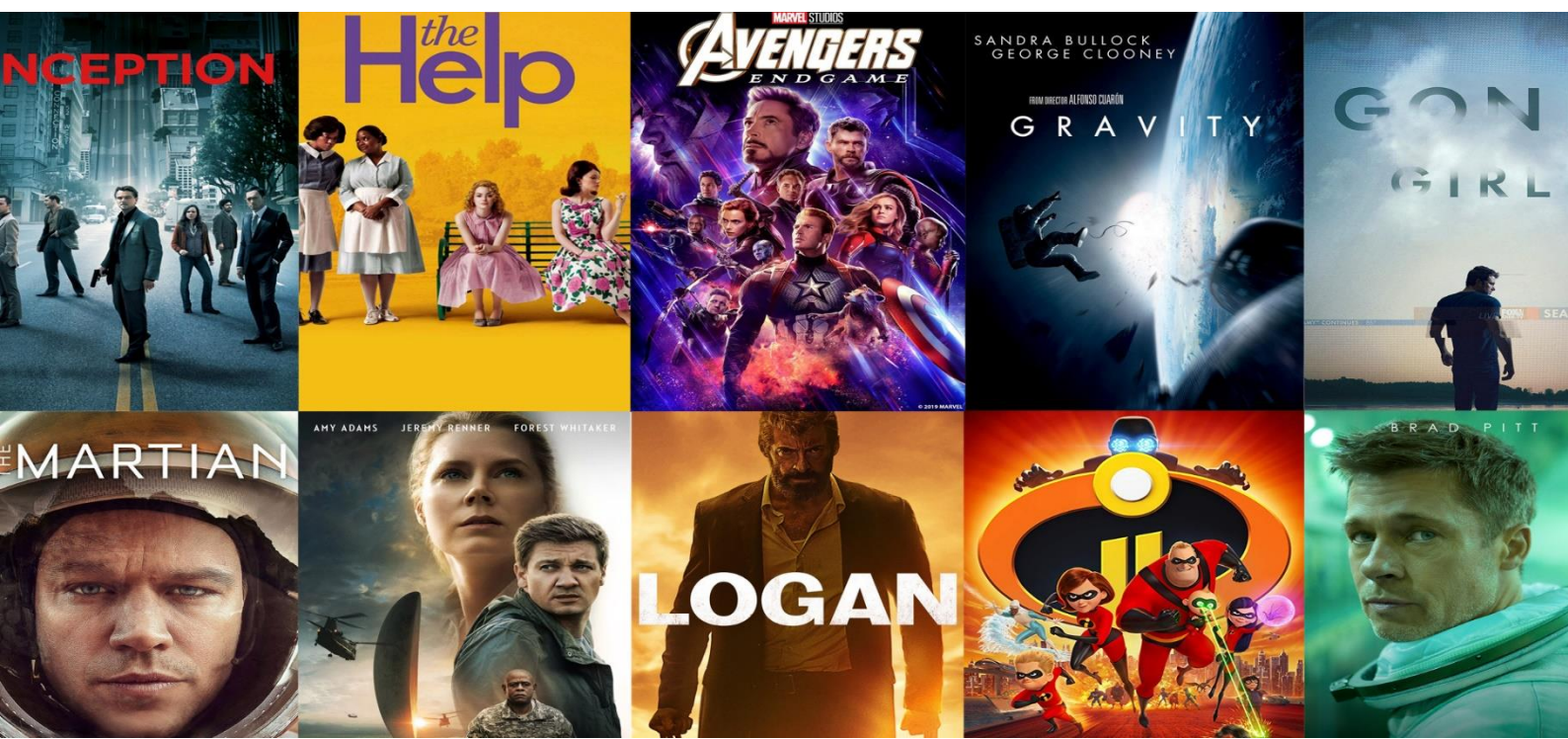




MOVIE RECOMMENDATION SYSTEM USING MACHINE LEARNING



PROJECT – MOVIE RECOMMENDATION SYSTEM USING MACHINE LEARNING

**SUBMITTED BY – JAYADEV P U
ENGINEERING WITH PYTHON
(10AM-1PM)
26_12_2021
BATCH – B17
PY26
DEC26th**

TEACHER IN CHARGE – Ms. SEEMA U

DATE – 28-02-2022

LANGUAGE – PYTHON

**LIBRARIES – NUMPY, PANDAS, DIFFLIB, PILLOW,
SKLEARN**

PYTHON TOOL USED – COSINE SIMILARITY

DEPLOYMENT TOOLS – GITHUB, STREAMLIT, HEROKU

INTRODUCTION

Recommendation System is a system that seeks to predict or filter preferences according to the user's choices.

It has a wide variety of applications, some of them are more popularised in the online platforms. The data/content in a recommendation system may differ on its use. The System identifies the user's choice and make suggestions on the basis of its selected features. Every user has their own preferences and aspects. It predicts interests of users and makes recommendation according to the interest model of users.

Here we use a MOVIE RECOMMENDATION SYSTEM, in which user can enter a favourite movie, so that the system can suggest some similar movies.

Movie Recommendation System can be classified as –

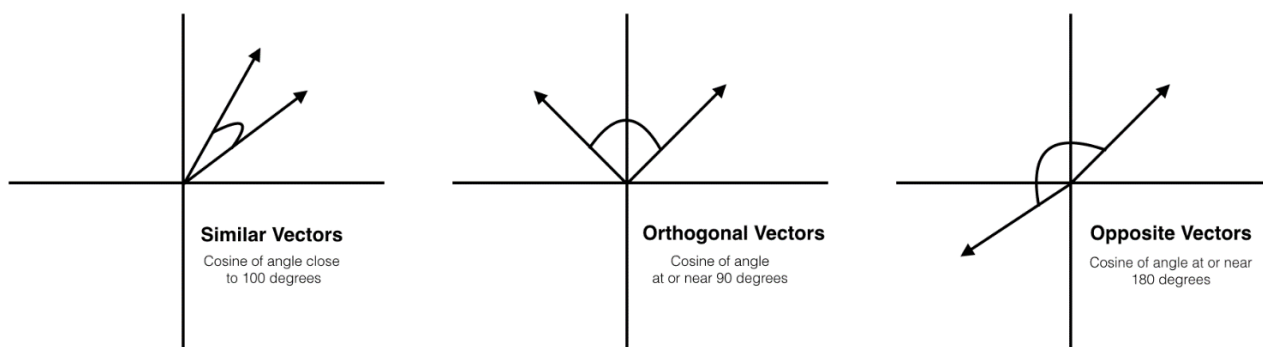
- Content Based Recommendation System
- Popularity Based Recommendation System
- Collaborative Recommendation System

This Recommendation System proposes a Content Based Recommendation System and uses Cosine Similarity measure to find the similarity of movies in the dataset.

Content-based recommendation is an important approach in recommendation systems. The basic idea is to recommend items that are similar with what user liked before. It extracts keywords of the data and calculate the weight by TF-IDF.

COSINE SIMILARITY

The cosine similarity measures the similarity between vector lists by calculating the cosine angle between the two vector lists. If you consider the cosine function, its value at 0 degrees is 1 and -1 at 180 degrees. This means for two overlapping vectors, the value of cosine will be maximum and minimum for two precisely opposite vectors.



The Cosine Similarity measurement begins by finding the cosine of the two non-zero vectors. This can be derived using the Euclidean dot product formula which is written as:

$$\mathbf{A} \cdot \mathbf{B} = \|\mathbf{A}\| \|\mathbf{B}\| \cos \theta$$

Then, given the two vectors and the dot product, the cosine similarity is defined as:

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|} = \frac{\sum_{i=1}^n A_i B_i}{\sqrt{\sum_{i=1}^n A_i^2} \sqrt{\sum_{i=1}^n B_i^2}},$$

The output will produce a value ranging from -1 to 1, indicating similarity where -1 is non-similar, 0 is orthogonal (perpendicular), and 1 represents total similarity.

DATASET

The dataset used is 'movies.csv'. Dataset contains over 4800 movies and 24 fields or columns.

Features or variables in the dataset:

- 'index',
- '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',
- 'cast',
- 'crew',
- 'director'

DATASET PROCESSING

The movies.csv dataset contains null values, which need to be processed before feature selection.

Shape of the Dataset

```
df.shape
```

```
(4803, 24)
```

Finding Null Values

```
df.isnull().any()
```

index	False
budget	False
genres	True
homepage	True
id	False
keywords	True
original_language	False
original_title	False
overview	True
popularity	False
production_companies	False
production_countries	False
release_date	True
revenue	False
runtime	True
spoken_languages	False
status	False
tagline	True
title	False
vote_average	False
vote_count	False
cast	True
crew	False
director	True

dtype: bool

Filling the Null values in the selected features

```
sf=['genres','keywords','tagline','cast','director']
```

```
for i in sf:  
    df[i]=df[i].fillna('')
```

SOURCE CODE

```
# MOVIE RECOMMENDATION SYSTEM USING MACHINE LEARNING

# Importing Libraries

# Python pillow library is used to image class within it to show the image
from PIL import Image
import numpy as np
import pandas as pd

# Streamlit is an open source app framework in Python language. It helps us create web
apps for data science and machine learning in a short time.
import streamlit as st

# DiffliB is a Python module that contains several easy-to-use functions and classes that
allow users to compare sets of data.
import difflib

# It is a simple technique to vectorize text documents - i.e. transform sentences into
arrays of numbers
from sklearn.feature_extraction.text import TfidfVectorizer

# In the sklearn module, there is an in-built function called cosine_similarity() to
calculate the cosine similarity.
from sklearn.metrics.pairwise import cosine_similarity

# Dataset

# Loading the dataset - "movies.csv" to "df" using pandas
df = pd.read_csv('movies.csv')

# Dataset preprocessing - Selecting required features or datas for the similarity
sf = ['genres', 'keywords', 'tagline', 'cast', 'director']
# The given dataset contains null values in some fields, so preprocessing the dataset is
required to fill null values.
for i in sf:
    df[i] = df[i].fillna('')

# Merging all the features to a single one for Vectorization
mf = df['genres']+' '+df['keywords']+' ' + \
    df['tagline']+' '+df['cast']+' '+df['director']
# TfidfVectorizer - Transforms text to feature vectors that can be used as input to
estimator.
vect = TfidfVectorizer()
fv = vect.fit_transform(mf)

# Cosine Similarity

# Cosine similarity measures the similarity between two vectors of an inner product space.
Cosine similarity is one of the metric to measure the text-similarity
similarity = cosine_similarity(fv)

# Streamlit
```

```

# Pillow is a Python Imaging Library (PIL), which adds support for opening, manipulating,
and saving images.
img = Image.open("mv.jpg")
# Streamlit - Display an image or list of images.
st.image(img, width=700)

# Streamlit - Display text in title formatting
st.title('MOVIE RECOMMENDATION SYSTEM USING MACHINE LEARNING')
# Streamlit - Write arguments to the app.
st.write("ENTER YOUR FAVOURITE MOVIE : ")
# Streamlit - Display a single-line text input widget.
movie_name = st.text_input('Enter the Movie name : ')
# Streamlit - Display a button widget
submit = st.button("SUBMIT")

# Output

# The try block lets you test a block of code for errors.
try:
    if submit:
        # mlist stores the list of all movie titles from the movies.csv dataset.
        mlist = df['title'].tolist()
        # get_close_matches() is a function that is available in the difflib Python
package.
        # The difflib module provides classes and functions for comparing sequences.
        # It finds the close matches of movies in mlist from the USER INPUT.
        match = difflib.get_close_matches(movie_name, mlist)
        # Finds the index of closely matched movie
        index = df[df.title == match[0]]['index'].values[0]
        # Finds the similarity of the user input with each movie in the dataset like a
loop.
        ss = list(enumerate(similarity[index]))
        # Sorting the similar values in descending or reverse order, so that the more
similar movies or values come first.
        sortedlist = sorted(ss, key=lambda x: x[1], reverse=True)
        # Writing or Printing the TOP 10, most similar movies in the sorted list using a
for loop.
        st.write("YOU MAY ALSO LIKE: ")
        k = 1
        for j in sortedlist:
            ind = j[0]
            x = df[df.index == ind]['title'].values[0]
            if(k <= 10):
                st.write(k, ".", x)
                k += 1
# The except block lets you handle the error.
except:
    st.write("MOVIE NOT FOUND")
    st.write("PLEASE ENTER THE CORRECT MOVIE NAME")

```


PROJECT



MOVIE RECOMMENDATION SYSTEM USING MACHINE LEARNING

ENTER YOUR FAVOURITE MOVIE :

Enter the Movie name :

SUBMIT

YOU MAY ALSO LIKE :

- 1 . Avatar
- 2 . Alien
- 3 . Guardians of the Galaxy
- 4 . Moonraker
- 5 . Space Dogs
- 6 . Alien³
- 7 . Aliens
- 8 . Clash of the Titans
- 9 . The Right Stuff
- 10 . Star Trek Beyond



MOVIE RECOMMENDATION SYSTEM USING MACHINE LEARNING

ENTER YOUR FAVOURITE MOVIE :

Enter the Movie name :

X-Men

SUBMIT

YOU MAY ALSO LIKE:

- 1 . X-Men
- 2 . X2
- 3 . X-Men: The Last Stand
- 4 . X-Men: Apocalypse
- 5 . X-Men: Days of Future Past
- 6 . Iron Man 2
- 7 . Avengers: Age of Ultron
- 8 . The Amazing Spider-Man 2
- 9 . The Avengers
- 10 . Ant-Man



MOVIE RECOMMENDATION SYSTEM USING MACHINE LEARNING

ENTER YOUR FAVOURITE MOVIE :

Enter the Movie name :

BIG B









SUBMIT

MOVIE NOT FOUND

PLEASE ENTER THE CORRECT MOVIE NAME

Made with Streamlit

FILES IN REPO

	Pipfile
	Procfile
	README.md
	movie_recommendation.py
	movies.csv
	mv.jpg
	requirements.txt
	setup.sh

PROJECT DEPLOYMENT

The project is deployed using **Streamlit** and **Heroku**

Using streamlit we can deploy any machine learning project and any python project with ease and without worrying about the frontend. Streamlit is very user-friendly.

Heroku is a Platform as a Service (PaaS). It is a cloud platform where one can build, operate and run his/her applications in the cloud itself. Heroku, other than being a very extensive and helpful platform, offers many free plans when you create a new account on the platform. It is great for beginners who are just starting out and trying to learn model deployment to take advantage of the free plans to deploy their model on cloud.

PROJECT DETAILS

Find the project on,

PROJECT LINK (HEROKU): <https://movierecommendationusingml.herokuapp.com/>

PROJECT REPO (GITHUB): <https://github.com/Jayadev-3/Movie-Recommendation-System-Using-Machine-Learning.git>

CONCLUSION

During the last years movie streaming platforms have become one of the most Internet services. Nowadays, everyone knows at least one of the biggest ones as for example Netflix. In fact, all these streaming platforms provides thousands of movies and series for all the users. But, have you ever wondered how do they recommend you the perfect movies?

The Content-based implementation achieves good results when searching for similar movies. There are plenty of way to expand on the work done in this project. Firstly, the content based method can be expanded to include more Criteria to help categorize the movies. Often, these systems are able to collect information about a user's choices, and can use this information to improve their suggestions in the future

REFERENCES

- <https://deepai.org/machine-learning-glossary-and-terms/cosine-similarity>
- <https://www.geeksforgeeks.org/python-measure-similarity-between-two-sentences-using-cosine-similarity/>
- <https://www.geeksforgeeks.org/>
- <https://stackoverflow.com/>
- PYTHON NOTES
- ICFOSS COURSE MATERIALS
- MACHINE LEARNING USING PYTHON | MANARANJAN PRADHAN | U
DINESH KUMAR Textbook