Project Report
On

MOVIE RECOMMENDATION SYSTEM

As a Field work for Course
ARTIFICIAL INTELLIGENCE
(INT-404)

Submitted By

Paramjit Singh
11807995
K18KKB49

Submitted To

Mr. Sagar Pande

Lovely Professional University
Jalandhar, Punjab, India.

Transforming Education Transforming India

1. ABSTRACT

The Project is titled as “Movie Recommendation System” and the basic aim of this system is to suggest some movies to the user based on some sort of similarity between the previous entered/selected movies by him/her. In our day to day life, we knowingly or unknowingly use various recommendation systems, be it on YouTube, Netflix, Amazon Prime or some other means. So, this project is the basic implementation of the wide area of recommendation networks and engines. In addition to this, basic GUI model has been implemented to make the surfing for the users easy and quick because nowadays user wants both intelligent and attractive systems. So, in the upcoming pages, the underlying principles, various methodologies, implementation and working would be explained.

2. METHODOLOGIES

The basic methodology behind this system is use of Natural Language Processing (NLP). There is use of NLP Algorithms like Bag Of Words. The name of the movie, actors, directors, genres, types have been clubbed into a single column or sentence and on that instances of ‘**CountVectorizer**’ class have been implemented. For example, the frequency array of all the words is developed for a particular movie and that array is then compared with the frequency array of all the other movies using ‘**cosine_similarity()**’ function from ‘**sklearn**’ library. Thus, by finding similarity based on content movies are recommended to the user. That is why this approach of filtering is also called Content Based Filtering.

And in the GUI model, out of say 4000 movies of dataset; randomly 10 movies are selected using the ‘**random.choice()**’ function from ‘**random**’ module and displayed on the GUI window. Then the user selects one movie and that name is passed to **movieCallBack()** function which gives us list of top five similar movies and then the final movie list is displayed before the user. And when the user again runs the program, new randomly generated list is displayed.

3. PREVIOUS WORK and LIMITATIONS

Prior to above scenario, the system was just able to display list of similar movies, there was no scope for user interface and interaction, so now that has been removed using the GUI model which is explained as above.

The main limitation of this system is that, at present situation it doesn’t store the user inputs in the databases which could be used for future references, the present model is based on basic implementation of NLP using just input and output streams. A good improvement in the project could be use of Collaborative Filtering approach which most of the recommendation engines use.

4. IMPLEMENTATION

The basic code of the project is shown as below …….

```
### importing python libraries #####################

import random

import pandas as pd

from sklearn.feature_extraction.text import CountVectorizer

from sklearn.metrics.pairwise import cosine_similarity




###########  user defined functions ###############

def get_title_from_index(index):

    return df[df.index == index]["title"].values[0]


def get_index_from_title(title):

    return df[df.title == title]["index"].values[0]

##################################################





from tkinter import *

top=Tk()



###read csv for title column
```

```
df=pd.read_csv("movie_dataset.csv")        # csv file having movie dataset
title_col=df["title"]



###code to make GUI for Selection or Main window
top.title("movie recommender")
top.geometry("821x650")



### movieCallBack() function will finds out movies similar to a movie selected by user
def movieCallBack(film):

    ### Select Features
    features=['keywords','cast','genres','director']

    for feature in features:
        df[feature]=df[feature].fillna('')



    ### we have to create a new column in DataFrame which would combine all the above selected features
    def combine_features(row):
        try:
            return row['keywords']+" "+row['cast']+" "+row['genres']+" "+row['director']
        except:
            print("Error",row)
```

```
df["combined_features"]=df.apply(combine_features,axis=1)


### Creating count matrix from this new combined column
cv=CountVectorizer()
count_matrix=cv.fit_transform(df["combined_features"])



### Compute the Cosine Similarity based on the count_matrix
cosine_sim=cosine_similarity(count_matrix)
movie_user_likes = film       # name of selected movie is given here


### Get index of this movie from its title
movie_index=get_index_from_title(movie_user_likes)
similar_movies=list(enumerate(cosine_sim[movie_index]))



### Get a list of similar movies in descending order of similarity score
sorted_similar_movies=sorted(similar_movies,key=lambda x:x[1],reverse=True)



### Storing most similar first five movies in a list
```

```
i=0

film_list2=[]

for movie in sorted_similar_movies:

    film_list2.append(get_title_from_index(movie[0]))

    i=i+1

    if i>5:

        break
```

### code to make GUI for recommendation window

```
tp=Tk()

tp.title("recommendations")

tp.geometry("815x380")




l1=Label(tp, text = "You previously selected",pady=10)

l1.grid(row = 1, column = 0, sticky = W)




box4=Button(tp,text=film,width=15,height=1,border=7,bg='lime',relief='flat')

box4.grid(row = 2, column = 0, sticky = W, padx = 4)


box5=Button(tp,text=" Recommended
Movies",width=20,height=1,border=7,bg='mediumturquoise',relief='flat')

box5.grid(row = 3, column = 2, sticky = W, padx = 4)
```

```
b1=Button(tp,text=film_list2[1],width=20,height=15,border=3,bg='turquoise',relief='flat',command=lambda:movieCallBack(film_list2[1]))

b2=Button(tp,text=film_list2[2],width=20,height=15,border=5,bg='mediumturquoise',relief='flat',command=lambda:movieCallBack(film_list2[2]))

b3=Button(tp,text=film_list2[3],width=20,height=15,border=7,bg='lightseagreen',relief='flat',command=lambda:movieCallBack(film_list2[3]))

b4=Button(tp,text=film_list2[4],width=20,height=15,border=5,bg='mediumturquoise',relief='flat',command=lambda:movieCallBack(film_list2[4]))

b5=Button(tp,text=film_list2[5],width=20,height=15,border=3,bg='turquoise',relief='flat',command=lambda:movieCallBack(film_list2[5]))



    b1.grid(row = 4, column = 0, sticky = W, padx = 4,pady=10)

    b2.grid(row = 4, column = 1, sticky = W, padx = 4)

    b3.grid(row = 4, column = 2, sticky = W, padx = 4)

    b4.grid(row = 4, column = 3, sticky = W, padx = 4)

    b5.grid(row = 4, column = 4, sticky = W, padx = 4)


    tp.mainloop()




l1=Label(top, text = " ",pady=10)
```

```
l1.grid(row = 0, column = 0, sticky = W)



box1=Button(top,text=" Click on any ",width=20,height=2,bg='lime',relief='flat')

box1.grid(row = 1, column = 1, sticky = W, padx = 4)

box2=Button(top,text=" Movie Button to
",width=20,height=2,border=5,bg='turquoise',relief='flat')

box2.grid(row = 1, column = 2, sticky = W, padx = 4)

box3=Button(top,text=" get similar movies ",width=20,height=2,bg='lime',relief='flat')

box3.grid(row = 1, column = 3, sticky = W, padx = 8)


l2=Label(top, text = " ",pady=10)

l2.grid(row = 2, column = 0, sticky = W)



# by using random.choice() function a random movie would be displayed on Movie
Selection window

# everytime the program is run

film_list1=[]

for i in range(10):

    film_list1.append(random.choice(title_col))



# creating buttons for the selection of a particular movie by user

b1=Button(top,text=film_list1[0],command=lambda:movieCallBack(film_list1[0]),width
=20,height=15,border=5,bg='lime',relief='flat')

b2=Button(top,text=film_list1[1],command=lambda:movieCallBack(film_list1[1]),width
=20,height=15,border=3,bg='turquoise',relief='flat')
```

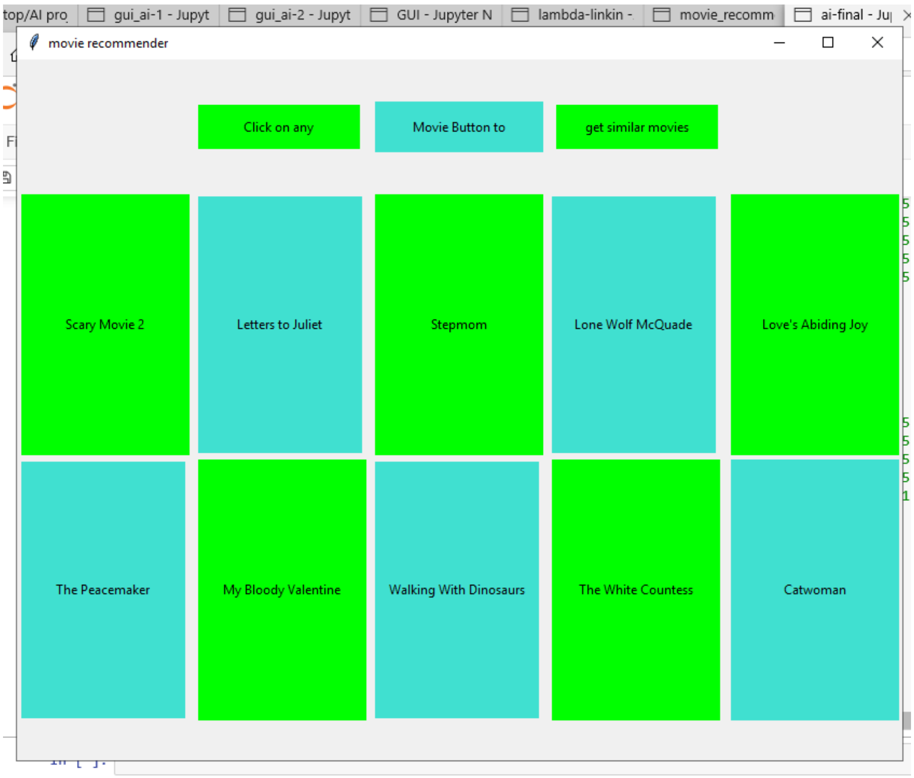
```
b3=Button(top,text=film_list1[2],command=lambda:movieCallBack(film_list1[2]),width=20,height=15,border=5,bg='lime',relief='flat')

b4=Button(top,text=film_list1[3],command=lambda:movieCallBack(film_list1[3]),width=20,height=15,border=3,bg='turquoise',relief='flat')

b5=Button(top,text=film_list1[4],command=lambda:movieCallBack(film_list1[4]),width=20,height=15,border=5,bg='lime',relief='flat')


b1.grid(row = 3, column = 0, sticky = W, padx = 4)

b2.grid(row = 3, column = 1, sticky = W, padx = 4)

b3.grid(row = 3, column = 2, sticky = W, padx = 4)

b4.grid(row = 3, column = 3, sticky = W, padx = 4)

b5.grid(row = 3, column = 4, sticky = W, padx = 4)


b6=Button(top,text=film_list1[6],command=lambda:movieCallBack(film_list1[6]),width=20,height=15,border=3,bg='turquoise',relief='flat')

b7=Button(top,text=film_list1[5],command=lambda:movieCallBack(film_list1[5]),width=20,height=15,border=5,bg='lime',relief='flat')

b8=Button(top,text=film_list1[8],command=lambda:movieCallBack(film_list1[8]),width=20,height=15,border=3,bg='turquoise',relief='flat')

b9=Button(top,text=film_list1[7],command=lambda:movieCallBack(film_list1[7]),width=20,height=15,border=5,bg='lime',relief='flat')

b10=Button(top,text=film_list1[9],command=lambda:movieCallBack(film_list1[9]),width=20,height=15,border=5,bg='turquoise',relief='flat')


b6.grid(row = 4, column = 0, sticky = W, padx = 4,pady=4)

b7.grid(row = 4, column = 1, sticky = W, padx = 4,pady=4)

b8.grid(row = 4, column = 2, sticky = W, padx = 4,pady=4)

b9.grid(row = 4, column = 3, sticky = W, padx = 4,pady=4)

b10.grid(row = 4, column = 4, sticky = W, padx = 4,pady=4)
```

top.mainloop()

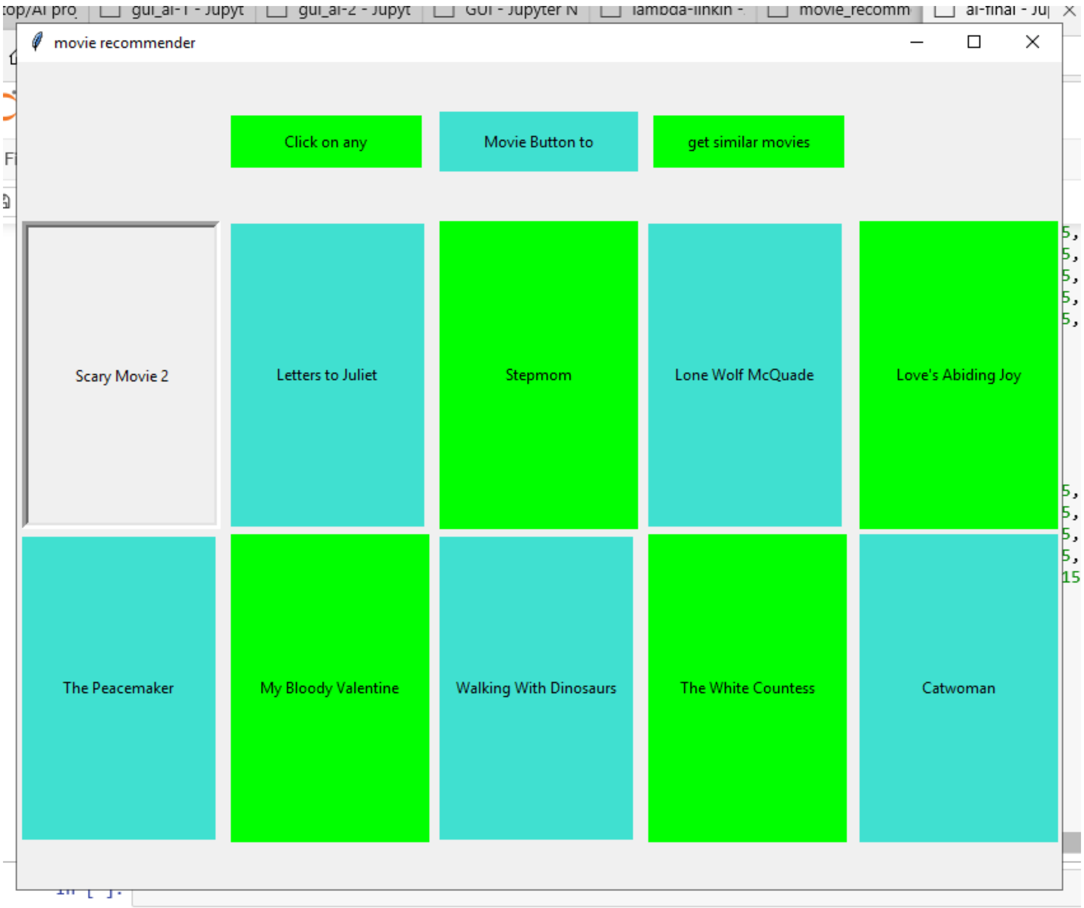
### end of program

The code which is highlighted in bold is basically the fresh and **original part of the code** which has been developed by me only. I have mainly focused on developing an attractive GUI for the user mainly due to personal interests in it. But all together, importance has been given in bringing out a good system as a whole.

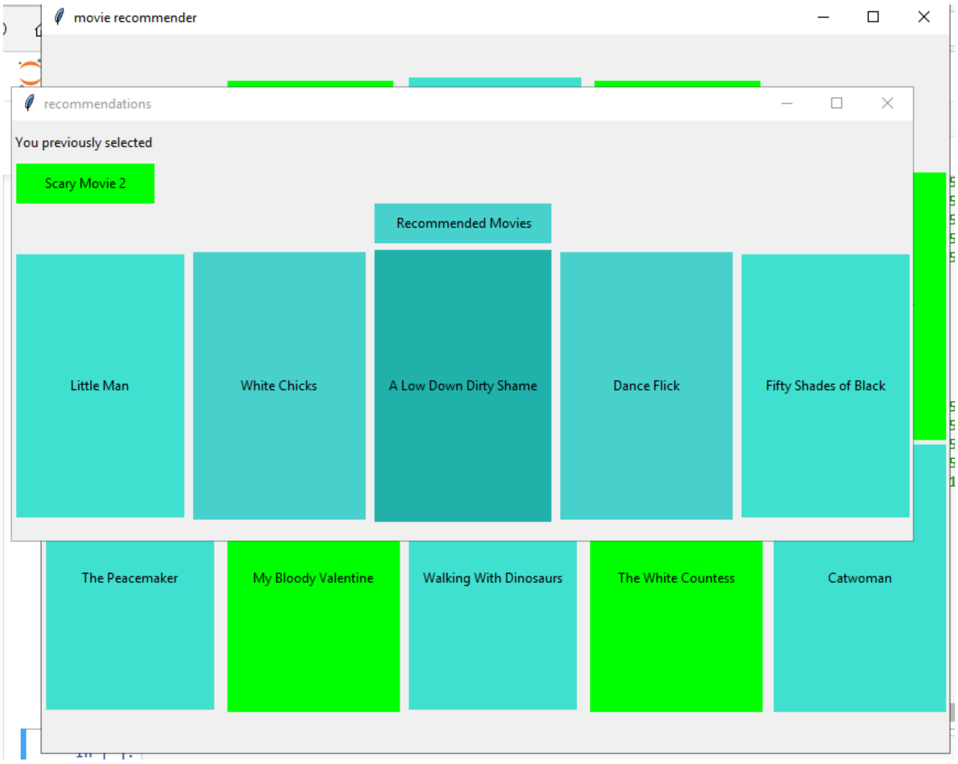
5. OUTPUT



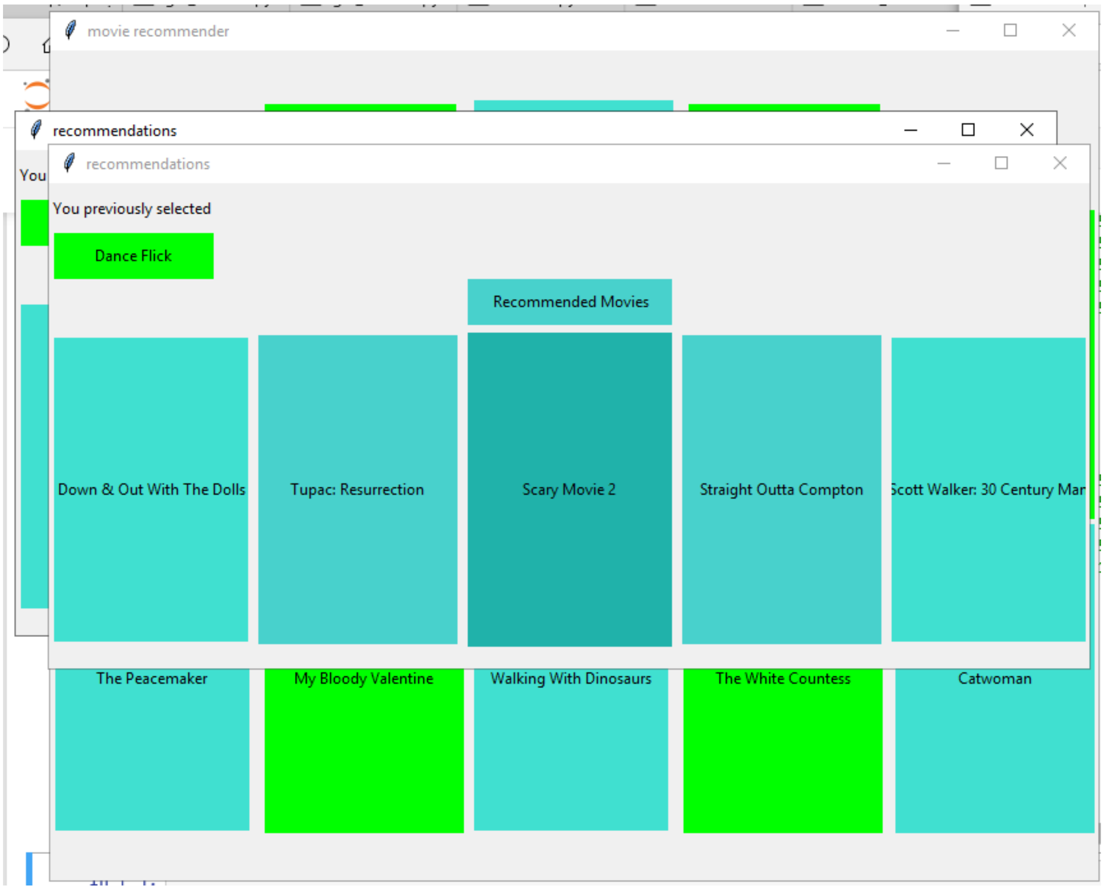
1a


movie recommender
Click on any
Movie Button to
get similar movies
Scary Movie 2
Letters to Juliet
Stepmom
Lone Wolf McQuade
Love's Abiding Joy
The Peacemaker
My Bloody Valentine
Walking With Dinosaurs
The White Countess
Catwoman

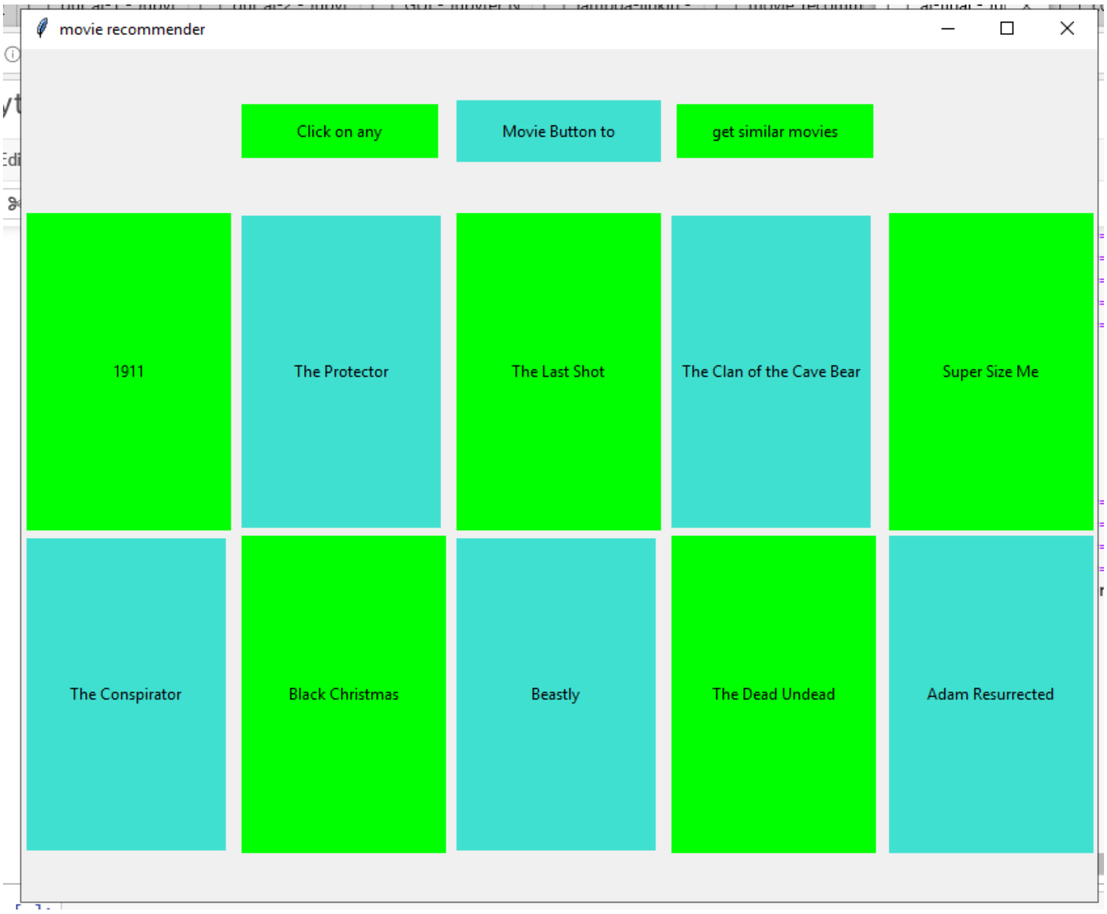
1b


movie recommender
recommendations
You previously selected
Scary Movie 2
Recommended Movies
Little Man
White Chicks
A Low Down Dirty Shame
Dance Flick
Fifty Shades of Black
The Peacemaker
My Bloody Valentine
Walking With Dinosaurs
The White Countess
Catwoman

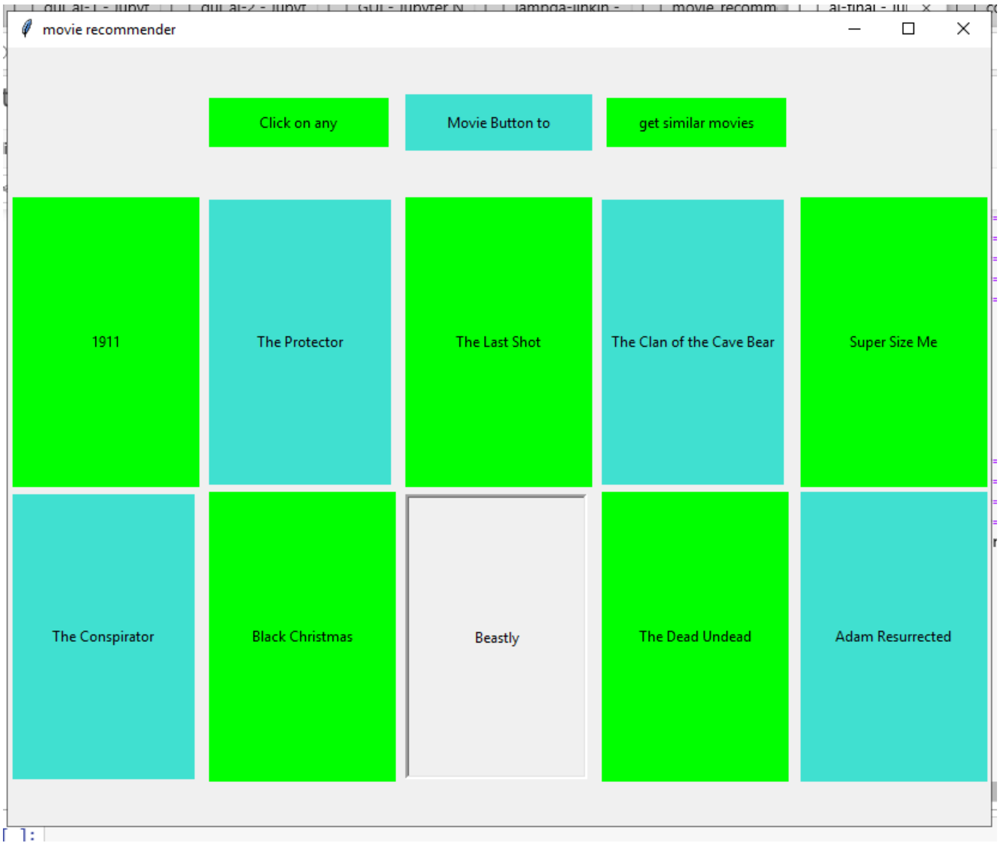
1c

movie recommender
recommendations
You
recommendations
You previously selected
Dance Flick
Recommended Movies
Down & Out With The Dolls
Tupac: Resurrection
Scary Movie 2
Straight Outta Compton
Scott Walker: 30 Century Man
The Peacemaker
My Bloody Valentine
Walking With Dinosaurs
The White Countess
Catwoman

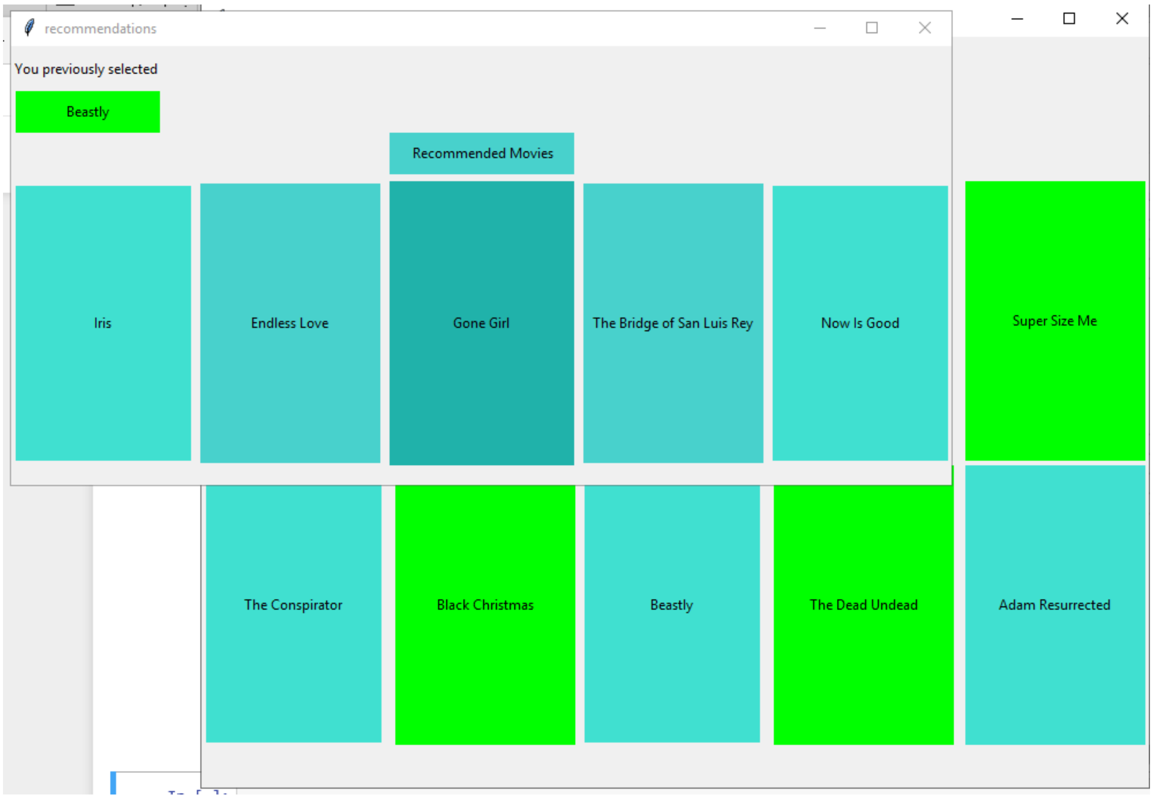
1d


movie recommender
Click on any
Movie Button to
get similar movies
1911
The Protector
The Last Shot
The Clan of the Cave Bear
Super Size Me
The Conspirator
Black Christmas
Beastly
The Dead Undead
Adam Resurrected

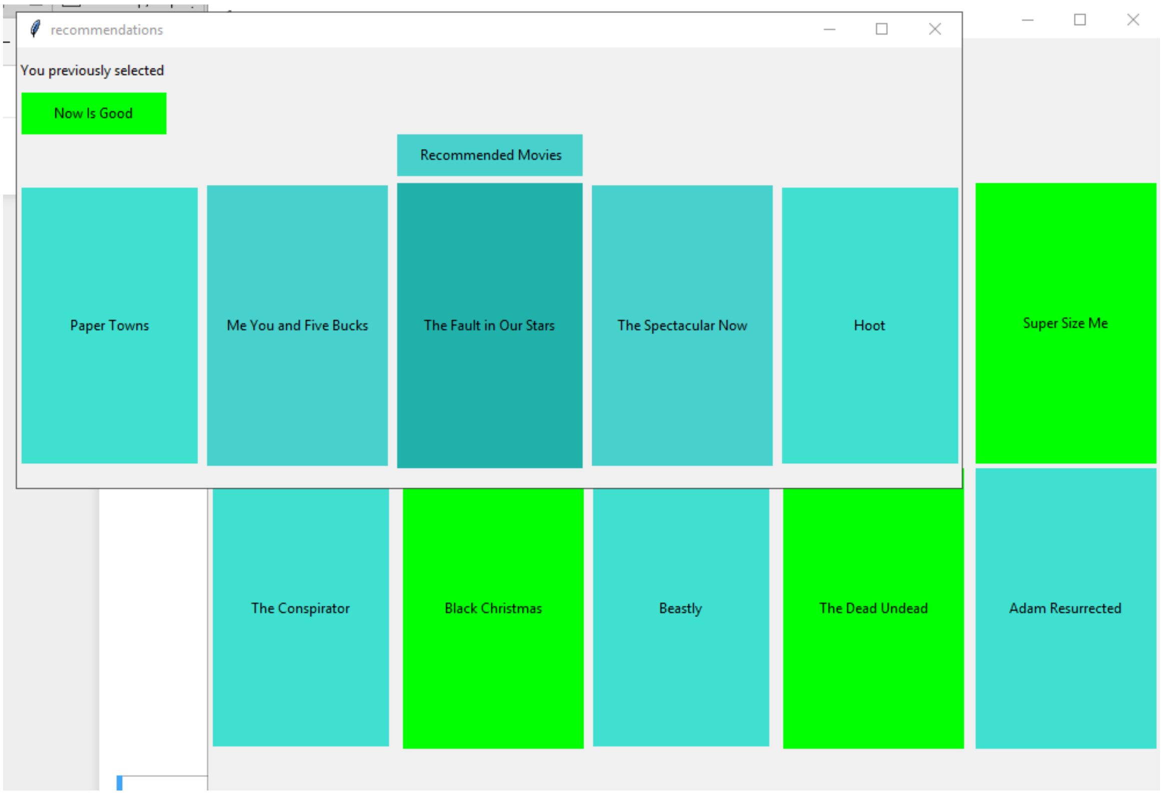
2a


movie recommender
Click on any
Movie Button to
get similar movies
1911
The Protector
The Last Shot
The Clan of the Cave Bear
Super Size Me
The Conspirator
Black Christmas
Beastly
The Dead Undead
Adam Resurrected

2b


recommendations
You previously selected
Beastly
Recommended Movies
Iris
Endless Love
Gone Girl
The Bridge of San Luis Rey
Now Is Good
Super Size Me
The Conspirator
Black Christmas
Beastly
The Dead Undead
Adam Resurrected

2c


recommendations
You previously selected
Now Is Good
Recommended Movies
Paper Towns
Me You and Five Bucks
The Fault in Our Stars
The Spectacular Now
Hoot
Super Size Me
The Conspirator
Black Christmas
Beastly
The Dead Undead
Adam Resurrected

2d

6. IMPORTANT LIBRARIES USED

The project is coded in Python programming language so the following libraries are used in the making of the project

a) Pandas

import pandas as pd

Pandas is an opensource library that allows to you perform data manipulation in Python. Pandas provide an easy way to create, manipulate and wrangle the data. So, the pandas are basically used to import dataset csv file into program carry out various manipulations over it.

b) Scikit-learn

from sklearn.feature_extraction.text import CountVectorizer

from sklearn.metrics.pairwise import cosine_similarity

Scikit-learn is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support vector machines, random forests, etc. Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python.

In this project, classes like CountVectorizer and functions like cosine_similarity are used from scikit-learn library.

c) Tkinter

from tkinter import *

Tkinter is the standard GUI library for Python. Python when combined with Tkinter provides a fast and easy way to create GUI applications. It provides a powerful object-oriented interface to the Tk GUI toolkit.

****random module**

import random

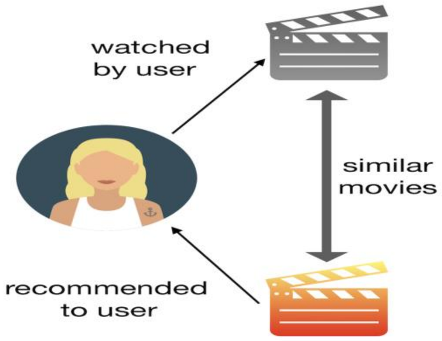
'**random.choice()**' function from '**random**' module is used to select a random movie when the user runs the program for each time.

6. TEAM RESPONSIBILITIES

This is a single-person project so as such no work is distributed. But if we look at the phases of project development, it can be said that firstly a basic movie recommender is built and then that has been incorporated into a GUI system.

7. REFERENCES

- Documentation from original libraries
- Youtube tutorials like CodeHeroku
- Websites like geekforgeeks, tutorialpoints, stackoverflow etc.



Everything is a Recommendation

Over 80% of what people watch comes from our recommendations

Recommendations are driven by Machine Learning