ARTIFICIAL INTELLIGENCE
(INT404)

Project Report

On

MOVIE RECOMMENDATION SYSTEM

Submitted To:
Mrs. Pooja Rana

Submitted By: Abhinav Mittal
Registration Number: 11802460
Section: K18AP
Roll Number: B49

Lovely Professional University
Jalandhar, Punjab, India.

1. ABSTRACT

The aim of Movie Recommendation System is to suggest movies to users based on the name of movies searched by the user previously. Various websites and applications like Netflix, YouTube, Amazon Prime, etc. use movie recommendation systems to suggest similar content to its users based on their search and watch history. In this project, basic GUI has been implemented to make the browsing easy, simple and fast to make the user experience better.

2. INTRODUCTION

A recommendation system is a type of information filtering system which challenges to assume the priorities of a user, and make recommendations on the basis of user's priorities. In a very general way, recommender systems are algorithms aimed at suggesting relevant items to users (items being movies to watch, text to read, products to buy or anything else depending on industries). Recommender systems are really critical in some industries as they can generate a huge amount of income when they are efficient or also be a way to stand out significantly from competitors. As a proof of the importance of recommender systems, we can mention that, a few years ago, Netflix organised a challenge (the "Netflix prize") where the goal was to produce a recommender system that performs better than its own algorithm with a prize of 1 million dollars to win.

3. LITERATURE REVIEW

Cosine similarity is a metric used to determine how similar the documents are irrespective of their size. It measures the cosine of the angle between two vectors projected in a multi-dimensional space. In this context, the two vectors we are talking about are arrays containing the word counts of two documents.

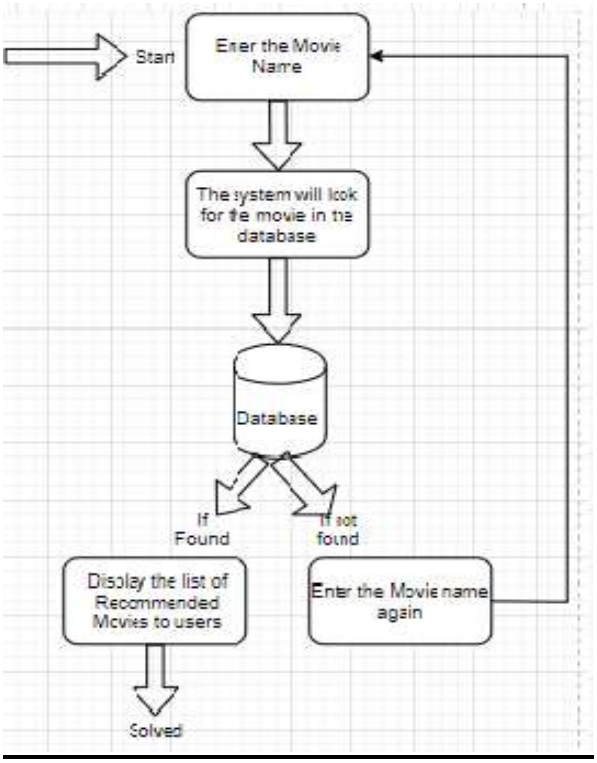
When plotted on a multi-dimensional space, where each dimension corresponds to a word in the document, the cosine similarity captures the orientation (the angle) of the documents and not the magnitude. If you want the magnitude, compute the Euclidean distance instead.

The cosine similarity is advantageous because even if the two similar documents are far apart by the Euclidean distance because of the size (like, the word 'cricket' appeared 50 times in one document and 10 times in another) they could still have a smaller angle between them. Smaller the angle, higher the similarity.

This article helped me in understanding the concept of cosine similarity. Article link is given below :

https://www.machinelearningplus.com/nlp/cosine-similarity/

4. PROPOSED METHODOLOGY



1) The methodology used in this system is Natural Language Processing (NLP).

2) The algorithms like Bag of Words are used. CountVectorizer class have been implemented in which name of movies, genres, actors, directors have been grouped into a single column. 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.

3) In the GUI model, out of say 4000 movies of dataset; the user enters one movie and that name is passed to **func1()** function which gives us list of similar movies and then the final movie list is displayed before the user.

4) On executing the program, a prompt window appears.

5) Enter a movie name and click on submit button.

6) System will look into the database for the movie.

7) If the movie is present in database then another window displaying movie recommendations will appear else if movie is not present in database then nothing will appear.

5. RESULT AND DISCUSSION

In most of the search engines, systems are just able to display list of similar movies based on their search and watch history as there was no user interface and user interaction but, in this project, GUI has been implemented to enhance user experience. But still this system has some flaws which are that it cannot store the inputs of user in database for further usage as this system is based on basic implementation of Natural Language Processing (NLP) using input and output streams. Another way is the use of Collaborative Filtering Approach which is used by the industry.

CODE

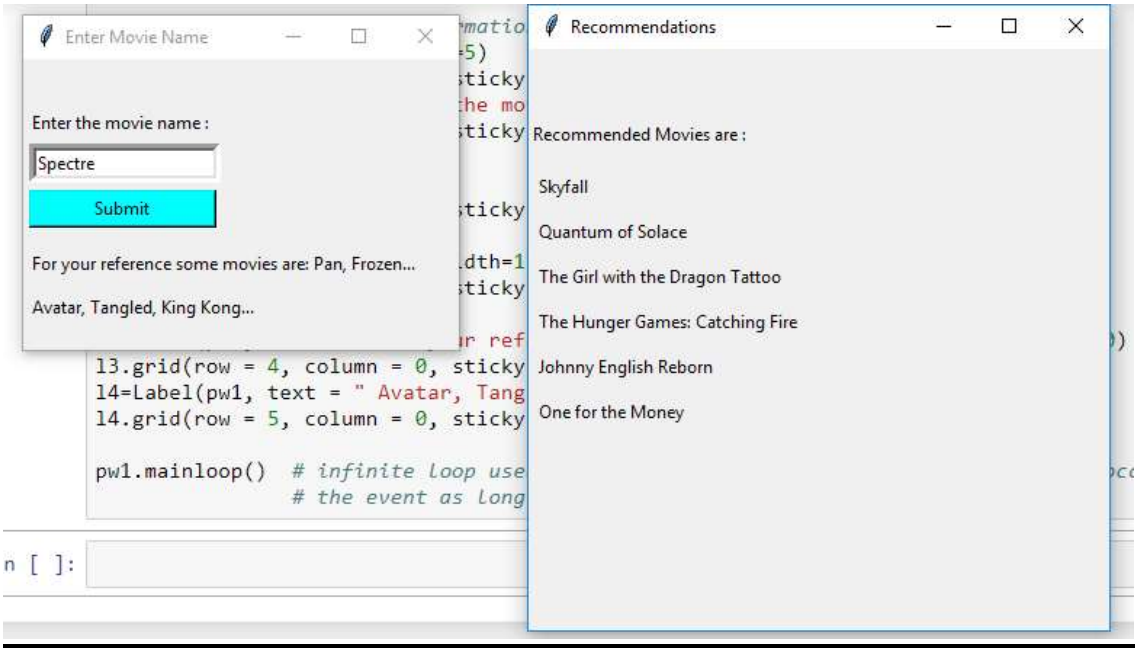
```
import pandas as pd # for data manipulation and analysis

from sklearn.feature_extraction.text import CountVectorizer # Scikit-learn library
# The CountVectorizer provides a simple way to both tokenize a collection of text documents and build a vocabulary of known words

from sklearn.metrics.pairwise import cosine_similarity #metric used to measure how similar the documents are irrespective of
                                                       #their size

from tkinter import *

pw1=Tk()

df=pd.read_csv("movie_dataset.csv") # to read dataset
title_col=df["title"]

pw1.title("Enter Movie Name")  # title of prompt window 1
pw1.geometry("300x200")        # dimension of prompt window 1

def func1():
    pic=e1.get()
    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]

    attributes=['keywords','cast','genres','director']

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

```
    def combine_attributes(row):
        try:
            return row['keywords']+" "+row['cast']+" "+row['genres']+" "+row['director']
        except:
            print("Error",row)

    df["combined_attributes"]=df.apply(combine_attributes,axis=1)

    cv=CountVectorizer()
    matrix=cv.fit_transform(df["combined_attributes"])

    cosine_sim=cosine_similarity(matrix)
    movie_user_likes = pic

    index=get_index_from_title(movie_user_likes)

    similar_movies=list(enumerate(cosine_sim[index])) #Enumerate() method adds a counter to an iterable and returns it in a form
    #of enumerate object.This enumerate object can then be used directly in for loops or be converted into a list of tuples
    #using list() method.

    sorted_similar_movies=sorted(similar_movies,key=lambda x:x[1],reverse=True)

    i=0
    list1=[]
    for movie in sorted_similar_movies:
        list1.append(get_title_from_index(movie[0]))
        i=i+1
        if i>10:
            break
```

```
    pw2=Tk()
    pw2.title("Recommendations")  # title of prompt window 2
    pw2.geometry("400x400")       # dimensions of prompt window 2

# for GUI and displaying information in prompt window 2
    lb1=Label(pw2, text = " ",pady=10)
    lb1.grid(row = 0, column = 0, sticky = W)#sticky is used to indicate the sides and corners of the cell to which widget sticks
    lb2=Label(pw2, text = "Recommended Movies are : ",pady=10)
    lb2.grid(row = 1, column = 0, sticky = W)
    lb3=Label(pw2, text = list1[1],padx=5,pady=7)
    lb3.grid(row = 2, column = 0, sticky = W)
    lb4=Label(pw2, text = list1[2],pady=5,padx=5)
    lb4.grid(row = 3, column = 0, sticky = W)
    lb5=Label(pw2, text = list1[3],padx=5,pady=7)
    lb5.grid(row = 4, column = 0, sticky = W)
    lb6=Label(pw2, text = list1[4],pady=5,padx=5)
    lb6.grid(row = 5, column = 0, sticky = W)
    lb7=Label(pw2, text = list1[5],padx=5,pady=7)
    lb7.grid(row = 6, column = 0, sticky = W)
    lb8=Label(pw2, text = list1[6],pady=5,padx=5)
    lb8.grid(row = 7, column = 0, sticky = W)

    pw2.mainloop() # infinite loop used to run the application, wait for an event to occur and process
                   # the event as long as the window is not closed.

# for GUI and displaying information in prompt window 1
l1=Label(pw1, text = " ",pady=5)
l1.grid(row = 0, column = 0, sticky = W)
l2=Label(pw1, text = " Enter the movie name : ",pady=5)
```

```
l2=Label(pw1, text = " Enter the movie name : ",pady=5)
l2.grid(row = 1, column = 0, sticky = W)

e1=Entry(pw1, bd =5)
e1.grid(row = 2, column = 0, sticky = W,padx=4)

b1=Button(pw1,text="Submit",width=17,height=1,bg='cyan',command=lambda:func1())
b1.grid(row = 3, column = 0, sticky = W, padx = 4,pady=5)

l3=Label(pw1, text = " For your reference some movies are: Pan, Frozen... ",pady=10)
l3.grid(row = 4, column = 0, sticky = W)
l4=Label(pw1, text = " Avatar, Tangled, King Kong... ",pady=1)
l4.grid(row = 5, column = 0, sticky = W)

pw1.mainloop()  # infinite loop used to run the application, wait for an event to occur and process
                # the event as long as the window is not closed.
```

OUTPUT



LIBRARIES USED

The project is coded in Python programming language and the following libraries are used in this 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.

CountVectorizer class and cosine_similarity function is 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.

6. CONCLUSION

This project of Movie Recommendation System is used for recommending movies to users based on their search history and is built using basic Natural Language Processing. It is coded in Python language using Python Libraries and Cosine Similarity technique. GUI is also implemented for better user experience and interaction. A CSV file containing information about movies has been linked to the code for recommending movies.

7. REFERENCES

- YouTube tutorials like CodeHeroku.
- Websites like GeeksForGeeks, Tutorialspoint, StackOverflow, Machinelearningplus, etc.

8. GitHub Link : https://github.com/abhinavm31/Movie-Recommendation-System