MOVIE RECOMMENDER SYSTEM WITH SENTIMENT ANALYSIS

Review-4

TECHNICAL ANSWERS FOR REAL WORLD PROBLEMS (ECE-3999)

SLOT-TG1

SUBMITTED TO -

DR. HARISH KITTURMALLIKARJUN,

DEAN, SENSE

Famous 5

Reg. No	Student Name
Ayushman Mookherjee	18BEC0888
Shatakshi Singh	18BEC0879
Abhishek Harikrishnan	18BEC0909
Sabthesh Jk Srinivas	18BEC0877
Varnika V Singh	18BEC0784

ABSTRACT:

Many people use the Internet as a place for seeking opinions. With the increasing amount of user based content on the web, there has been an emergence of research fields that uses sentiment analysis to take advantage of, and process this data. This could result in more satisfied customers, as relevant information will be easier to find.

Movie recommendation system has become an interesting research topic due to the growth of users in a mobile environment. To recommend movies, a complete aggregation of user's preferences, feelings (emotions), and reviews required to assist users for find best movies in more convenient way. In this research, we present a new method, based on extracting and analysing adjectives from user-based reviews. We exploit the idea that adjectives often contain a sentiment, and present a system entirely based on adjectives as sentiment deciders.

However, to deal with the recommendation system, we must consider timeliness and accuracy. Hence, we will also adopt new user similarity metric,

similarity score and opinion mining. The primary objective of this project is to find the type of opinions (positive, negative, or neutral) for movies and suggest top-k recommendation list for users. We would extract aspect-based specific ratings from reviews and recommend reviews to users depends on user similarity and their rating patterns. Finally, validating the proposed movie recommendation system for various evaluation criteria, and the proposed system shows better result than conventional systems.

NOVELTY:

•Recommender systems are becoming more popular in today's era. As the name suggests, they are required for recommending products or services. Recommender systems minimize the transaction costs and improve the quality and decision-making process to users. It is applied in various neighbouring areas like information retrieval or human computer interaction (HCI). It gathers huge amount of information about user's preferences of several items like online shopping products, movies, TV, tourism, and food.

•In this project, we propose a novel scheme for extracting opinion-based movie similes from viewer-posted reviews. We look to design a recommendation system that provides a top-k list of items by user-movie similarity and review opinions.

MOTIVATION AND IDEOLOGY:

According to a survey from 2018 of 2400 adult Americans, by Pew Internet & American Life Project:

- 81% of Internet users have reached information on the Internet about a product they are thinking about buying.
- 20% are doing this on the typical day.
- 79% of Internet users are confident in making the correct decision, as they gather information online in advance of buying something.

With the increasing amount of data and comments, there is almost impossible to read it all. Most people are satisfied by reading the first few comments. A tool that can go through all the comments of a system, rank them positive or negative and summarize them can be extremely powerful in the recommendation area.

Netflix is an example of a global provider of a streaming service. Their main product is a subscription service that allows members to stream any movie or television show in their collection at any time. In December 2015 they had more than 65 million members, which streamed more than 100 million hours of movies and television shows per day. In October 2016, they opened a competition for the best collaborative filtering algorithm to predict user ratings for movies, based on previous ratings, called the Netflix Prize, with a grand prize of $1,000,000. As long as the competition was still active (nobody won the grand prize), it was also possible to win a progress prize of $50,000. It was given each year, to the best entry thus far, that also improved the previous progress prize winner by at least 1%. 3 years after the start of the competition, the team "BellKor's Pragmatic Chaos" won the grand prize for improving Netflix's own algorithm more than 10%. The fact that Netflix was willing to give away that amount of money, shows the importance of their recommender system, as they said themselves in the rules for the competition: "because, frankly, if there is a much better approach it could make a big difference to our customers and our business".

APPROACH:

•This section will briefly describe how we proceed to achieve our goal. We will start with a restudy to examine the recommender and sentiment analysis approaches that exist today, what their functionality, possibilities and limitations are.

•To further investigate how adjectives could be exploited and used to improve classification results, we will use LingPipeto extract adjectives from our dataset. For each adjective, we will find its synonyms and antonyms and their sentiment with the help from WordNet and SentiWordNet.

•The sentiment of all the adjectives will be calculated by the majority vote of the sentiments of the corresponding synonyms together with the opposite sentiment scores of the corresponding antonyms. The number of positive adjectives and negative adjectives will then be counted for each movie review and used as attributes for the classifier.

•Finally, the classification algorithms Random Forest and Support Vector Mac

•The details of the movies (title, genre, runtime, rating, poster, etc) are fetched using an API by TMDB.(Here)

•Using the IMDB id of the movie in the API, web scraping will be performed to get the reviews given by the user in the IMDB site using beautifulsoup4 and then sentiment analysis will be applied on those reviews.

METHODOLOGY:

- Step 1: Data Pre-processing
  Various preprocessing steps such as word segmentation, stopwords removal, POS tagging and representation of reviews are applied to the raw data to convert it to usable form.

- Step 2: Aspect Extraction
  The measurement of the polarity is done for the reviews. The adjectives are extracted and a polarity(Positive, negative or neutral) is assigned to the adjective.

- Step 3: Opinion Detection

The collected user reviews are sorted.

- Step 4: User Similarity Computation and Recommendation
  The similarity is calculated using a weight adjusted cosine score metric. The resulting matrix reviews are ranked in descending order to create the Top-K recommendation list which is used to generate a personalized list for each used.

WORKFLOW DISTRIBUTION:

•Database collection –Sabthesh and Varnika

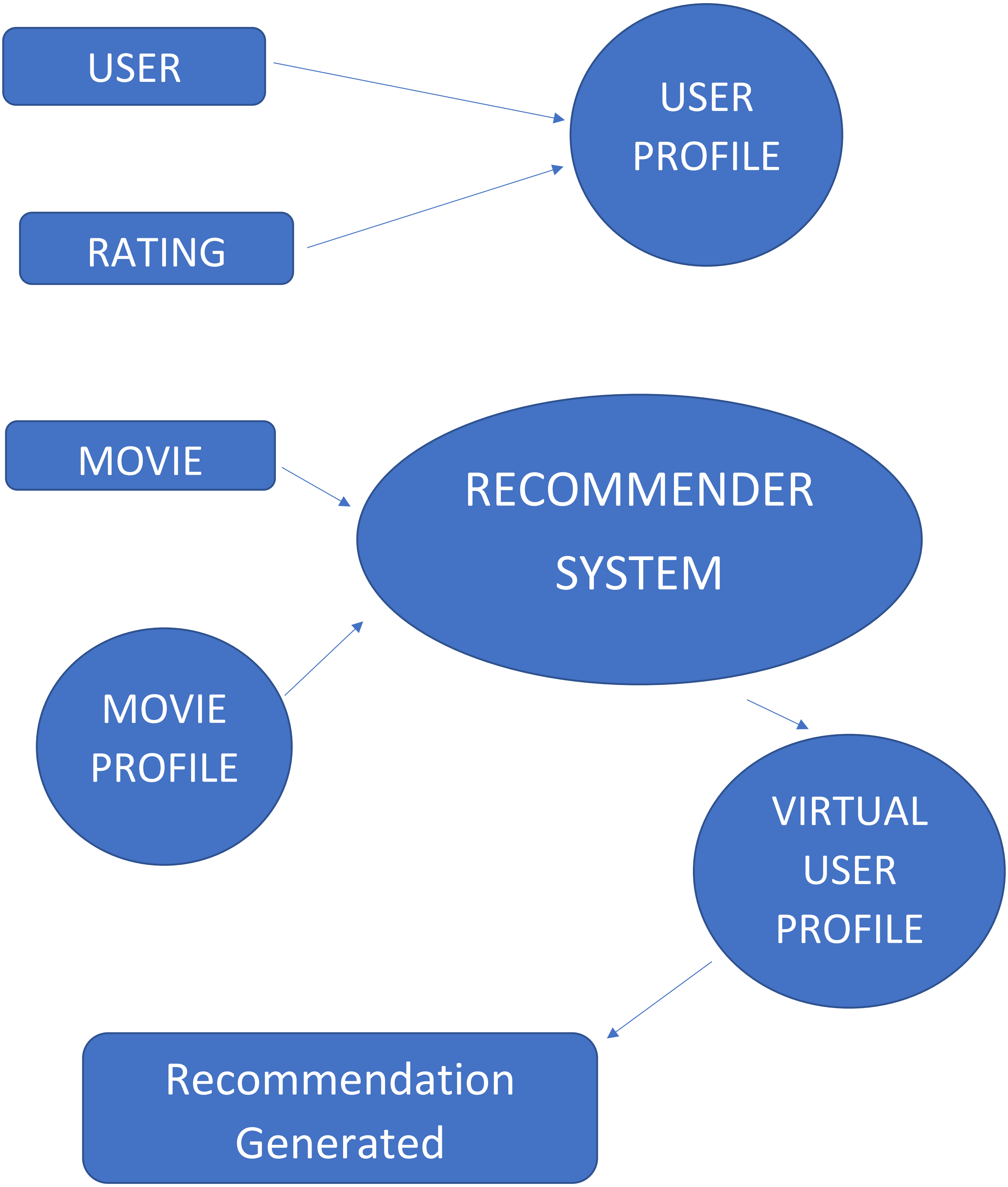
•Web development–Ayushman and Shatakshi

•API deployment and environment –Abhishek and Sabthesh

•Machine learning implementation and data analysis –Ayushman, Abhishek and Shatakshi

•Documentation and timeline–Sabthesh, Varnika and Abhishek

BLOCK DIAGRAM:



CONCEPTS AND LANGUAGES:

•Python 3.8

Used to write the algorithm for the sentiment analysis of the opinions.

•Web scraping

Used to get the reviews given by the user in the IMDB site using beautifulsoup4.

•NLP

Word Net (NLP Tool) is applied different applications [text mining, named entity recognition, information retrieval, etc.].

•Machine learning and sentiment analysis

Used for the calculation of the majority vote of the sentiments for each movie review with the help of positive and negative adjectives which will be used as attributes for the classifier.

•API's

Used to fetch details of the movies(title, genre, rating, runtime, etc) from TMDB.

•Jupyter Notebook

IDE used to write the Python algorithm

•HTML

Used for the design of the web application.

•JAVASCRIPT

Used for the implementation of complex features in the web pages.

•CSS

Used for describing the presentation of a document written in a markup language such as HTML.

•Flask

It is a micro web framework written in python and is used for the fetching of the API from TMDB.

LITERATURE SURVEY:

i. This paper presents an inductive learning way to deal with proposal that can utilize the ratings as well as different types of data about every ancient rarity in foreseeing client inclinations. This is required on the grounds that; most proposal frameworks make recommendations about ancient rarities to a client. For example, they may foresee whether a client would be keen on watching a specific film. Social recommendation strategies gather ratings from numerous people, and use nearest neighbour procedures to make proposals to a client concerning new ancient rarities. Nonetheless, these techniques don't utilize the critical measure of other data that is regularly accessible about the idea of every ancient rarity -, for example, projected records or film audits, for instance. The researchers attempt to show that their strategy beats a current social-sifting technique in the area of film proposals on a dataset of in excess of 45,000 film evaluations gathered from a network of more than 250 clients.

ii. The quick expansion of data technologies particularly Web 2.0 methods has changed the basic ways how things can be possible in numerous territories, including how scientists could impart and work together with one another. The existence of the sheer volume of specialists and exploration data on the Web has prompted the issue of data in surplus. There is a crucial need to create specialist suggestion operators to such an extent that clients can be furnished with customized proposals of the scientists they can possibly work together with for shared exploration benefits. In scholastic settings, prescribing reasonable exploration accomplices to specialists can encourage information revelation and trade, and at last improve the exploration profitability of scientists. Existing skill proposal research as a rule explores the master suggesting issue from two free measurements, to be specific, their social relations and ability data. The principle commitment of this paper is that the researchers have propose a system-based analyst suggestion approach which joins informal organization investigation and semantic idea examination in a brought together structure to improve the adequacy of customized specialist suggestion. The consequences of the test show that the proposed approach fundamentally outflanks the other pattern strategies. Besides, how the proposed structure can be applied to this present reality scholastic settings is clarified dependent on a contextual analysis.

iii. In this paper, a film recommendation structure dependent on hybrid suggestion and conclusion investigation is proposed to improve the precision of recommender frameworks. Moreover, Spark is utilized to improve the practicality of the framework. The proposed technique makes it advantageous and quick for clients to acquire valuable film suggestions, involves different sorts of clients and different sorts of motion pictures. Considering the valuable data covered up in surveys posted by clients and watcher history, community oriented filtering is viewed as the most well-known and broadly sent strategy in recommender system. Sentiment analysis will assist us with improving the exactness of proposal results and proposed structure can be improved in a few viewpoints. To begin with, this strategy can be verified in more information sets. Different information can be utilized by different assessment investigation, so the model can be tuned to oblige more situations. Secondly, in the examination cycle of the estimation examination, various types of abstract thoughts are included unavoidably, which actualizes unfavourable effects on the outcomes.

iv. In this article hybrid of k-means and cuckoo search is applied to the Movie lens point dataset to accomplish an improved film suggestion framework. The presentation of the methodology with respect to MAE, RMSE, SD, and t-value is estimated. The examination results on the Movielens dataset talked about demonstrated that the methodology that examined gives elite with respect to exactness and makes it fit for giving solid and customized film suggestion frameworks with the particular number of bunches. Assessment measurements (for a given number of bunches) began to be lesser than those of different techniques. A few drawbacks discovered are that on the initial profile if the underlying segment doesn't end up working admirably, at that point, proficiency may diminish. The best encouraging strategy is the one wherein the calculations utilized in grouping and enhancement give the best presentation with respect to precision and speed.

v. Recommendation systems have gotten pervasive lately as they manage the excess data issue by recommending clients the most applicable items from a monstrous measure of information. For media item, online collab film proposals make endeavours to help clients to get to their favoured motion pictures by catching decisively comparable neighbours among clients or films from their chronicled regular evaluations. In any case, because of the information meagrely, neighbour choosing is getting more troublesome with the quick expanding of motion pictures and clients. In this paper, a half breed model- based film suggestion framework which uses the improved K-implies grouping combined with hereditary calculations (GAs) to segment changed client space is proposed. The examination results on Movielens dataset show that the proposed approach can give elite as far as exactness, and create more solid and customized film proposals when contrasted and the current techniques.

vi. This article accomplished a very much arranged writing audit and ordered, integrate, and furthermore introduced the articles as indicated by the different impression of full of feeling recommender frameworks. Consequences of this paper are featured with important recommendations and future work. By distributed articles up until this point, emotional suggestion distributions are as yet developing and drifting exploration territory. There is a requirement for full of feeling recommender frameworks in which: utilization of feelings as the setting in the section stage and demonstrating emotional substance profiles ought to be maintained. Some calculations or models ought to be intended to predicate clients' present enthusiastic states or mind-set that can help in basic regions, for example, clinical application to every patient, fire stations, street security, and traffic examination. Further consideration is required on full of feeling figuring with the human dynamic cycle. Security and protection are as yet the significant issues in the emotional recommender systems. The principal spotlight ought to be on advanced methodologies: Managing full of feeling information quality, spatial publicly supporting, semantic, for proficient full of feeling recommender frameworks.

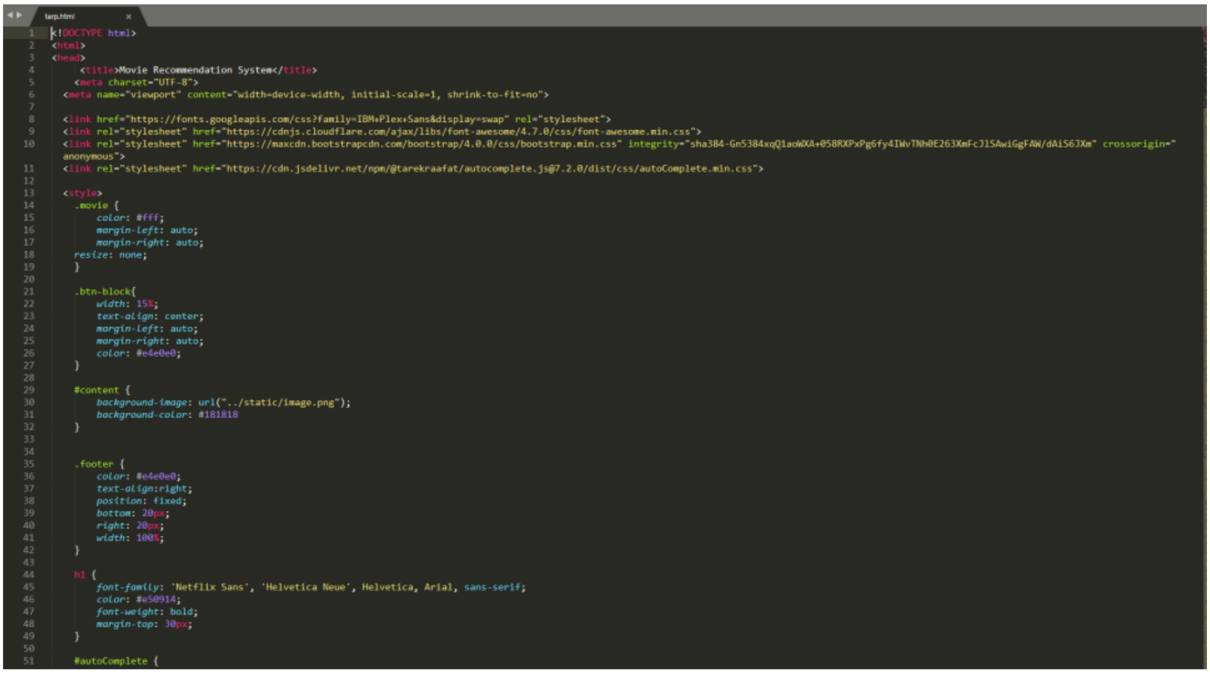
vii. This paper portrays, assesses and looks at a scope of strategies for individuals to-individuals suggestion in web-based dating, put together both with respect to Collaborative Filtering and Profile Matching. The researchers built up a two-phase fell recommender framework where competitors produced by Collaborative Filtering are then re-positioned utilizing a Decision Tree critic developed from preparing information. Assessment on recorded information demonstrated that the consolidated recommender advances-fewer famous competitors and improves client achievement rates. This research shows that "unadulterated" Collaborative Filtering is a promising way to deal with individuals to-individuals suggestion in web-based dating, however experiences the issue of low client inclusion, subsequently the following coherent advance is to assemble and assess cross breed recommenders that join Collaborative Filtering with Profile Matching or proportions of client similitude.

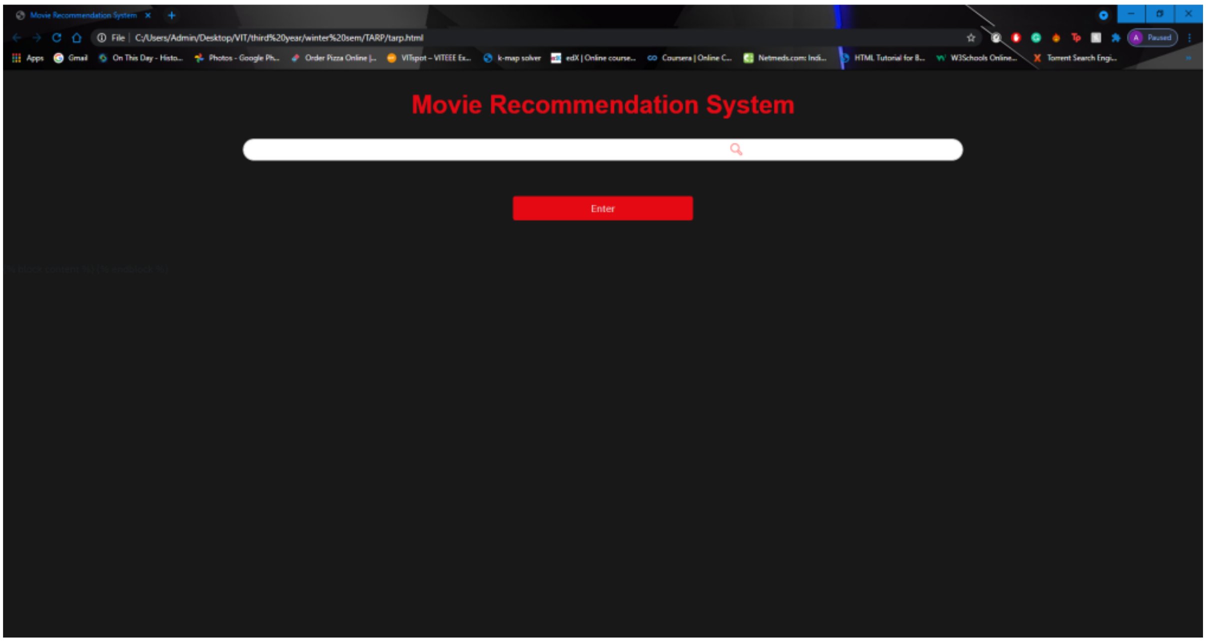
viii. This paper assembles two UGC-based factual models, which can use various sorts of UGC to make an association between a client's intrigued points and a thing's viewpoints. It gives unified approach to utilize various kinds of UGC in a recommender framework. Also, it checks that social labels and client surveys are important assets to derive a client's inclination and a thing's angles. Not just this, the exploratory outcomes confirm that the generative method of a rating and UGC text in our models is powerful. Likewise, the boundary assessment calculation is confirmed to be successful. These accomplishments are likewise important to other related exploration fields.

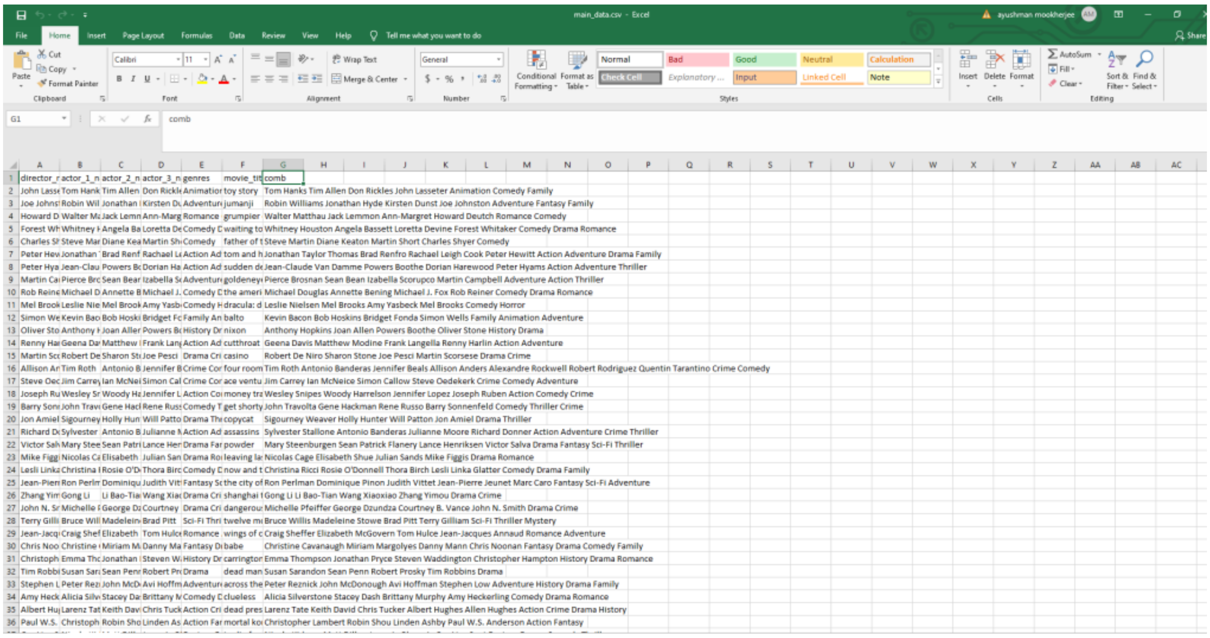
ix. In the spread of data, how to rapidly locate one's preferred film in countless motion pictures become a significant issue. Customized proposal framework can assume a significant job particularly when the client has no unmistakable target film. In this paper, the researcher’s structure and execute a film proposal framework model joined with the real needs of film suggestion through exploring of KNN calculation and collective sifting calculation. At this moment, a point by point standard and engineering of JAVAEE framework social information base model is given. At long last, the test outcomes indicated that the framework has a decent suggestion impact.

x. Customized suggestion framework essentially includes the client gauge alternatives inclinations, and to foresee the most fitting choices prescribed to the client. Shared separating calculation dependent on clients, despite the fact that the quick and exact to make recommendations, but the calculation exist such issues as information inadequacy and adaptability. While venture based synergistic sifting calculation to tackle the community separating calculation dependent on client information inadequacy issue, but since of the calculation depends on comparative things to suggest, not suggested across types, to be specific, absence of particular revelation. Improved blend albeit collective sifting calculation dependent on the task and clients can all the while explain the collaborative filtering algorithm in light of client and dependent on the task experienced issues, calculations of versatility issues despite everything exist, so coordination separating calculation stays to be further improved.

PROGRESS SO FAR:

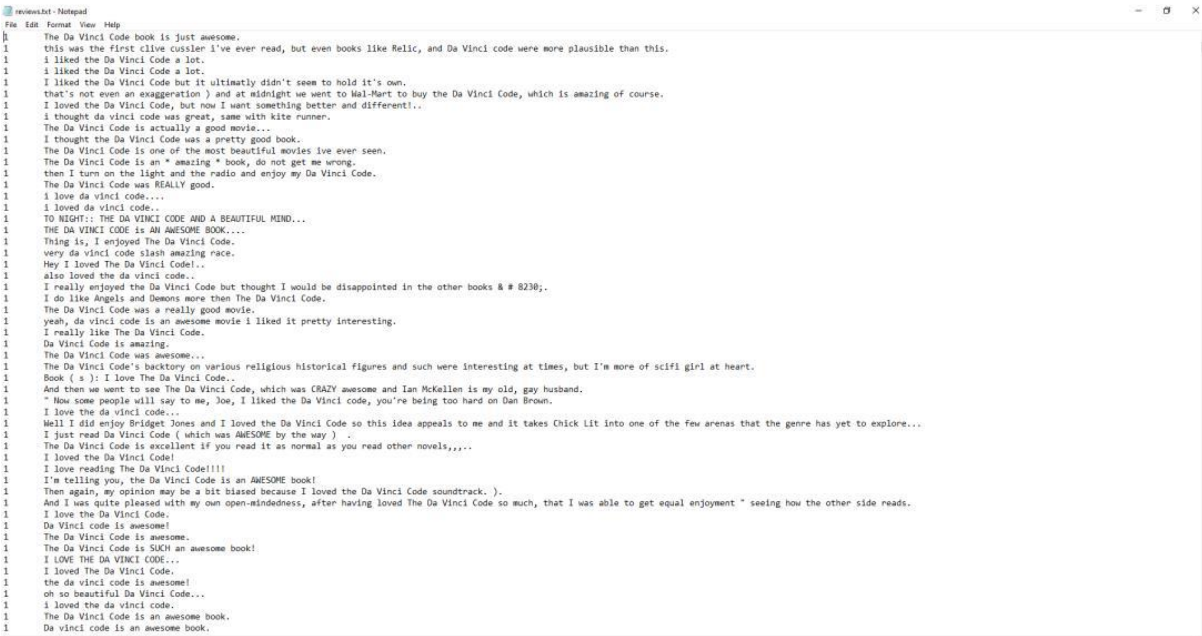






main_data.csv - Excel

	A	B	C	D	E	F	G
1	director_r	actor_1_n	actor_2_n	actor_3_n	genres	movie_tit	comb
2	John Lasse	Tom Hank	Tim Allen	Don Rickle	Animatior	toy story	Tom Hanks Tim Allen Don Rickles John Lasseter Animation Comedy Family
3	Joe Johns	Robin Wil	Jonathan	Kirsten Du	Adventure	jumanji	Robin Williams Jonathan Hyde Kirsten Dunst Joe Johnston Adventure Fantasy Family
4	Howard D	Walter Ma	Jack Lemn	Ann-Marg	Romance	grumpier	Walter Matthau Jack Lemmon Ann-Margret Howard Deutch Romance Comedy
5	Forest Wh	Whitney H	Angela Ba	Loretta De	Comedy D	waiting to	Whitney Houston Angela Bassett Loretta Devine Forest Whitaker Comedy Drama Romance
6	Charles Sh	Steve Mar	Diane Kea	Martin Sh	Comedy	father of t	Steve Martin Diane Keaton Martin Short Charles Shyer Comedy
7	Peter Hew	Jonathan	Brad Renf	Rachael Le	Action Ad	tom and h	Jonathan Taylor Thomas Brad Renfro Rachael Leigh Cook Peter Hewitt Action Adventure Drama Family
8	Peter Hya	Jean-Clau	Powers Bo	Dorian Ha	Action Ad	sudden de	Jean-Claude Van Damme Powers Boothe Dorian Harewood Peter Hyams Action Adventure Thriller
9	Martin Ca	Pierce Bro	Sean Bear	Izabella Sc	Adventure	goldeneye	Pierce Brosnan Sean Bean Izabella Scorupco Martin Campbell Adventure Action Thriller
10	Rob Reine	Michael D	Annette B	Michael J.	Comedy D	the ameri	Michael Douglas Annette Bening Michael J. Fox Rob Reiner Comedy Drama Romance
11	Mel Brook	Leslie Nie	Mel Brook	Amy Yasb	Comedy H	dracula: d	Leslie Nielsen Mel Brooks Amy Yasbeck Mel Brooks Comedy Horror
12	Simon We	Kevin Bac	Bob Hoski	Bridget Fo	Family An	balto	Kevin Bacon Bob Hoskins Bridget Fonda Simon Wells Family Animation Adventure
13	Oliver Sto	Anthony H	Joan Aller	Powers Bo	History Dr	nixon	Anthony Hopkins Joan Allen Powers Boothe Oliver Stone History Drama
14	Renny Har	Geena Da	Matthew	Frank Lang	Action Ad	cutthroat	Geena Davis Matthew Modine Frank Langella Renny Harlin Action Adventure
15	Martin Sco	Robert De	Sharon St	Joe Pesci	Drama Cri	casino	Robert De Niro Sharon Stone Joe Pesci Martin Scorsese Drama Crime
16	Allison An	Tim Roth	Antonio B	Jennifer B	Crime Cor	four room	Tim Roth Antonio Banderas Jennifer Beals Allison Anders Alexandre Rockwell Robert Rodriguez Quentin Tarantino Crime Comedy
17	Steve Oec	Jim Carrey	Ian McNei	Simon Cal	Crime Cor	ace ventu	Jim Carrey Ian McNeice Simon Callow Steve Oedekerk Crime Comedy Adventure
18	Joseph Ru	Wesley Sn	Woody Ha	Jennifer L	Action Co	money tra	Wesley Snipes Woody Harrelson Jennifer Lopez Joseph Ruben Action Comedy Crime
19	Barry Son	John Trav	Gene Hac	Rene Russ	Comedy T	get shorty	John Travolta Gene Hackman Rene Russo Barry Sonnenfeld Comedy Thriller Crime
20	Jon Amiel	Sigourney	Holly Hun	Will Patto	Drama Thr	copycat	Sigourney Weaver Holly Hunter Will Patton Jon Amiel Drama Thriller
21	Richard Do	Sylvester	Antonio B	Julianne M	Action Ad	assassins	Sylvester Stallone Antonio Banderas Julianne Moore Richard Donner Action Adventure Crime Thriller
22	Victor Sal	Mary Stee	Sean Patri	Lance Her	Drama Far	powder	Mary Steenburgen Sean Patrick Flanery Lance Henriksen Victor Salva Drama Fantasy Sci-Fi Thriller
23	Mike Figgi	Nicolas Ca	Elisabeth	Julian San	Drama Ro	leaving la	Nicolas Cage Elisabeth Shue Julian Sands Mike Figgis Drama Romance
24	Lesli Linka	Christina	Rosie O'D	Thora Birc	Comedy D	now and t	Christina Ricci Rosie O'Donnell Thora Birch Lesli Linka Glatter Comedy Drama Family
25	Jean-Pierr	Ron Perlm	Dominiqu	Judith Vit	Fantasy Sc	the city of	Ron Perlman Dominique Pinon Judith Vittet Jean-Pierre Jeunet Marc Caro Fantasy Sci-Fi Adventure
26	Zhang Yim	Gong Li	Li Bao-Tia	Wang Xiao	Drama Cri	shanghai t	Gong Li Li Bao-Tian Wang Xiaoxiao Zhang Yimou Drama Crime
27	John N. Sr	Michelle F	George Dz	Courtney	Drama Cri	dangerou	Michelle Pfeiffer George Dzundza Courtney B. Vance John N. Smith Drama Crime
28	Terry Gilli	Bruce Will	Madelein	Brad Pitt	Sci-Fi Thri	twelve mo	Bruce Willis Madeleine Stowe Brad Pitt Terry Gilliam Sci-Fi Thriller Mystery
29	Jean-Jacq	Craig Shef	Elizabeth	Tom Hulce	Romance	wings of c	Craig Sheffer Elizabeth McGovern Tom Hulce Jean-Jacques Annaud Romance Adventure
30	Chris Noo	Christine	Miriam M	Danny Ma	Fantasy Dr	babe	Christine Cavanaugh Miriam Margolyes Danny Mann Chris Noonan Fantasy Drama Comedy Family
31	Christoph	Emma Tho	Jonathan	Steven Wa	History Dr	carrington	Emma Thompson Jonathan Pryce Steven Waddington Christopher Hampton History Drama Romance
32	Tim Robbi	Susan Sar	Sean Penn	Robert Pro	Drama	dead man	Susan Sarandon Sean Penn Robert Prosky Tim Robbins Drama
33	Stephen L	Peter Rezi	John McD	Avi Hoffm	Adventure	across the	Peter Reznick John McDonough Avi Hoffman Stephen Low Adventure History Drama Family
34	Amy Heck	Alicia Silv	Stacey Da	Brittany M	Comedy D	clueless	Alicia Silverstone Stacey Dash Brittany Murphy Amy Heckerling Comedy Drama Romance
35	Albert Hu	Larenz Tat	Keith Dav	Chris Tuck	Action Cri	dead pres	Larenz Tate Keith David Chris Tucker Albert Hughes Allen Hughes Action Crime Drama History
36	Paul W.S.	Christoph	Robin Sho	Linden As	Action Far	mortal ko	Christopher Lambert Robin Shou Linden Ashby Paul W.S. Anderson Action Fantasy



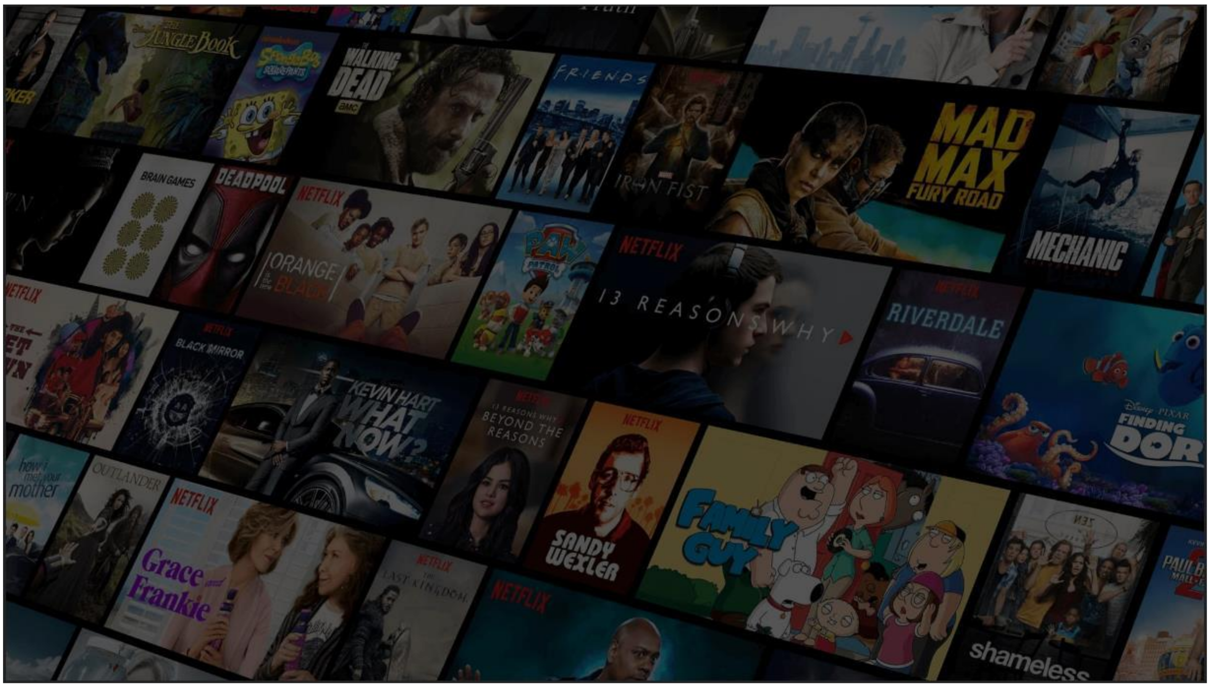
reviews.txt - Notepad

```
1	The Da Vinci Code book is just awesome.
1	this was the first clive cussler i've ever read, but even books like Relic, and Da Vinci code were more plausible than this.
1	i liked the Da Vinci Code a lot.
1	i liked the Da Vinci Code a lot.
1	I liked the Da Vinci Code but it ultimatly didn't seem to hold it's own.
1	that's not even an exaggeration ) and at midnight we went to Wal-Mart to buy the Da Vinci Code, which is amazing of course.
1	I loved the Da Vinci Code, but now I want something better and different!..
1	i thought da vinci code was great, same with kite runner.
1	The Da Vinci Code is actually a good movie...
1	I thought the Da Vinci Code was a pretty good book.
1	The Da Vinci Code is one of the most beautiful movies ive ever seen.
1	The Da Vinci Code is an * amazing * book, do not get me wrong.
1	then I turn on the light and the radio and enjoy my Da Vinci Code.
1	The Da Vinci Code was REALLY good.
1	i love da vinci code....
1	i loved da vinci code..
1	TO NIGHT:: THE DA VINCI CODE AND A BEAUTIFUL MIND...
1	THE DA VINCI CODE is AN AWESOME BOOK....
1	Thing is, I enjoyed The Da Vinci Code.
1	very da vinci code slash amazing race.
1	Hey I loved The Da Vinci Code!..
1	also loved the da vinci code..
1	I really enjoyed the Da Vinci Code but thought I would be disappointed in the other books & # 8230;.
1	I do like Angels and Demons more then The Da Vinci Code.
1	The Da Vinci Code was a really good movie.
1	yeah, da vinci code is an awesome movie i liked it pretty interesting.
1	I really like The Da Vinci Code.
1	Da Vinci Code is amazing.
1	The Da Vinci Code was awesome...
1	The Da Vinci Code's backtory on various religious historical figures and such were interesting at times, but I'm more of scifi girl at heart.
1	Book ( s ): I love The Da Vinci Code..
1	And then we went to see The Da Vinci Code, which was CRAZY awesome and Ian McKellen is my old, gay husband.
1	" Now some people will say to me, Joe, I liked the Da Vinci code, you're being too hard on Dan Brown.
1	I love the da vinci code...
1	Well I did enjoy Bridget Jones and I loved the Da Vinci Code so this idea appeals to me and it takes Chick Lit into one of the few arenas that the genre has yet to explore...
1	I just read Da Vinci Code ( which was AWESOME by the way )  .
1	The Da Vinci Code is excellent if you read it as normal as you read other novels,,,..
1	I loved the Da Vinci Code!
1	I love reading The Da Vinci Code!!!!
1	I'm telling you, the Da Vinci Code is an AWESOME book!
1	Then again, my opinion may be a bit biased because I loved the Da Vinci Code soundtrack. ).
1	And I was quite pleased with my own open-mindedness, after having loved The Da Vinci Code so much, that I was able to get equal enjoyment " seeing how the other side reads.
1	I love the Da Vinci Code.
1	Da Vinci code is awesome!
1	The Da Vinci Code is awesome.
1	The Da Vinci Code is SUCH an awesome book!
1	I LOVE THE DA VINCI CODE...
1	I loved The Da Vinci Code.
1	the da vinci code is awesome!
1	oh so beautiful Da Vinci Code...
1	i loved the da vinci code.
1	The Da Vinci Code is an awesome book.
1	Da vinci code is an awesome book.
```

main.py - C:\Users\Admin\Desktop\VIT\third year\winter sem\TARP\main.py (3.7.0)

File Edit Format Run Options Window Help

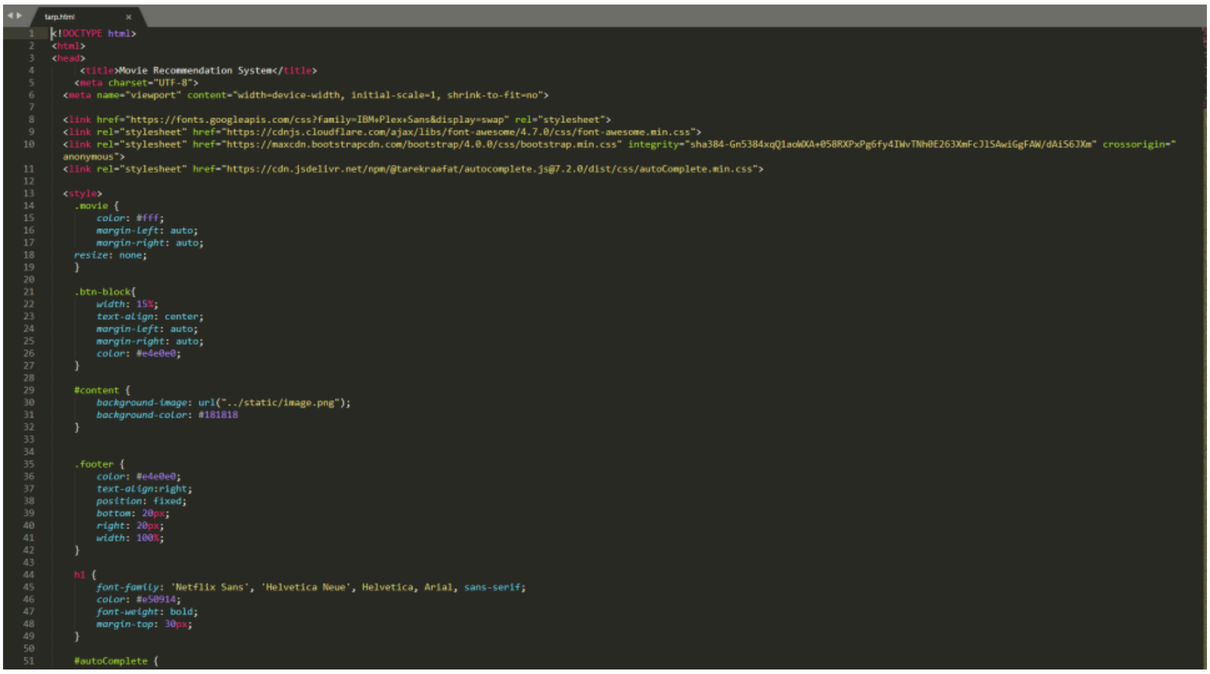
```python
import numpy as np
import pandas as pd
from flask import Flask, render_template, request
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics.pairwise import cosine_similarity
import json
import bs4 as bs
import urllib.request
import pickle
import requests
from datetime import date, datetime

# load the nlp model and tfidf vectorizer from disk
filename = 'nlp_model.pkl'
clf = pickle.load(open(filename, 'rb'))
vectorizer = pickle.load(open('tranform.pkl','rb'))

# converting list of string to list (eg. "["abc","def"]" to ["abc","def"])
def convert_to_list(my_list):
    my_list = my_list.split('","')
    my_list[0] = my_list[0].replace('["','')
    my_list[-1] = my_list[-1].replace('"]','')
    return my_list

# convert list of numbers to list (eg. "[1,2,3]" to [1,2,3])
def convert_to_list_num(my_list):
    my_list = my_list.split(',')
    my_list[0] = my_list[0].replace("[","")
    my_list[-1] = my_list[-1].replace("]","")
    return my_list

def get_suggestions():
    data = pd.read_csv('main_data.csv')
    return list(data['movie_title'].str.capitalize())

app = Flask(__name__)

@app.route("/")
@app.route("/home")
def home():
    suggestions = get_suggestions()
    return render_template('home.html',suggestions=suggestions)


@app.route("/recommend",methods=["POST"])
def recommend():
    # getting data from AJAX request
    title = request.form['title']
    cast_ids = request.form['cast_ids']
    cast_names = request.form['cast_names']
    cast_chars = request.form['cast_chars']
    cast_bdays = request.form['cast_bdays']
    cast_bios = request.form['cast_bios']
    cast_places = request.form['cast_places']
    cast_profiles = request.form['cast_profiles']
    imdb_id = request.form['imdb_id']
    poster = request.form['poster']
    genres = request.form['genres']
    overview = request.form['overview']
    vote_average = request.form['rating']
    vote_count = request.form['vote_count']
```



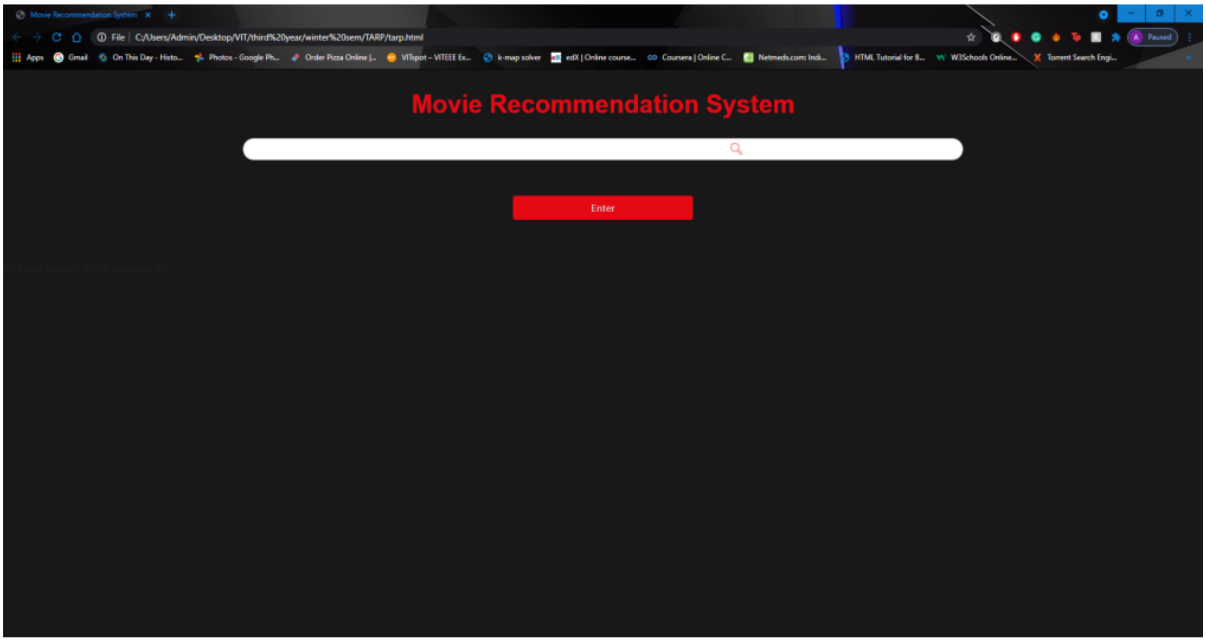
sentiment.ipynb - Notepad

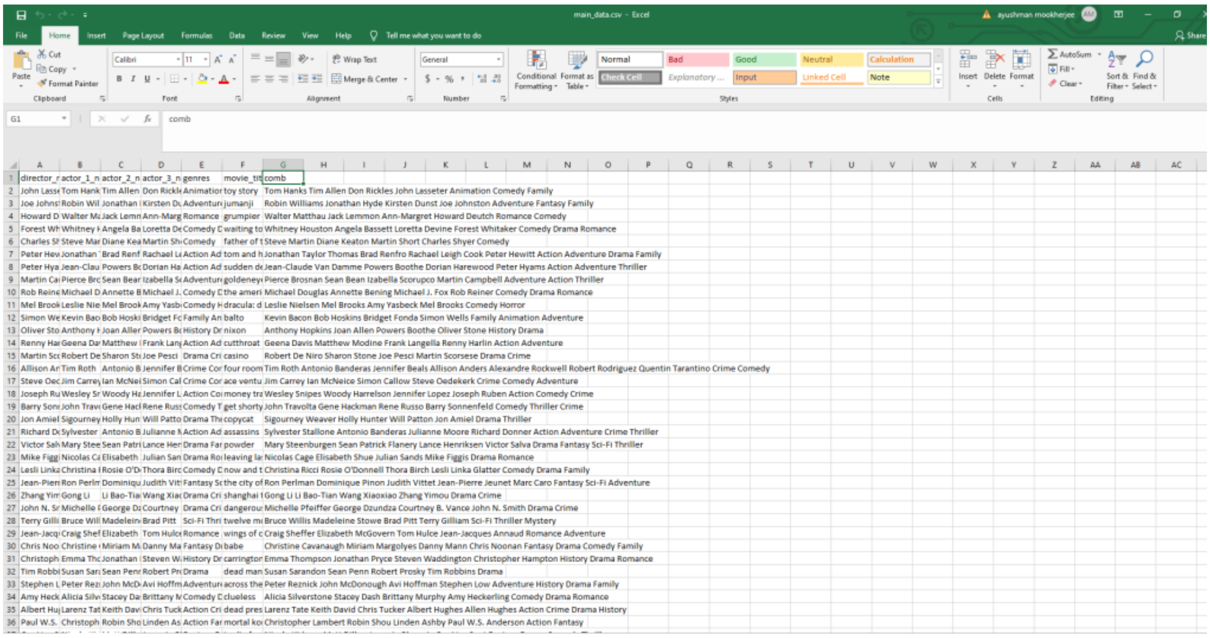
File Edit Format View Help

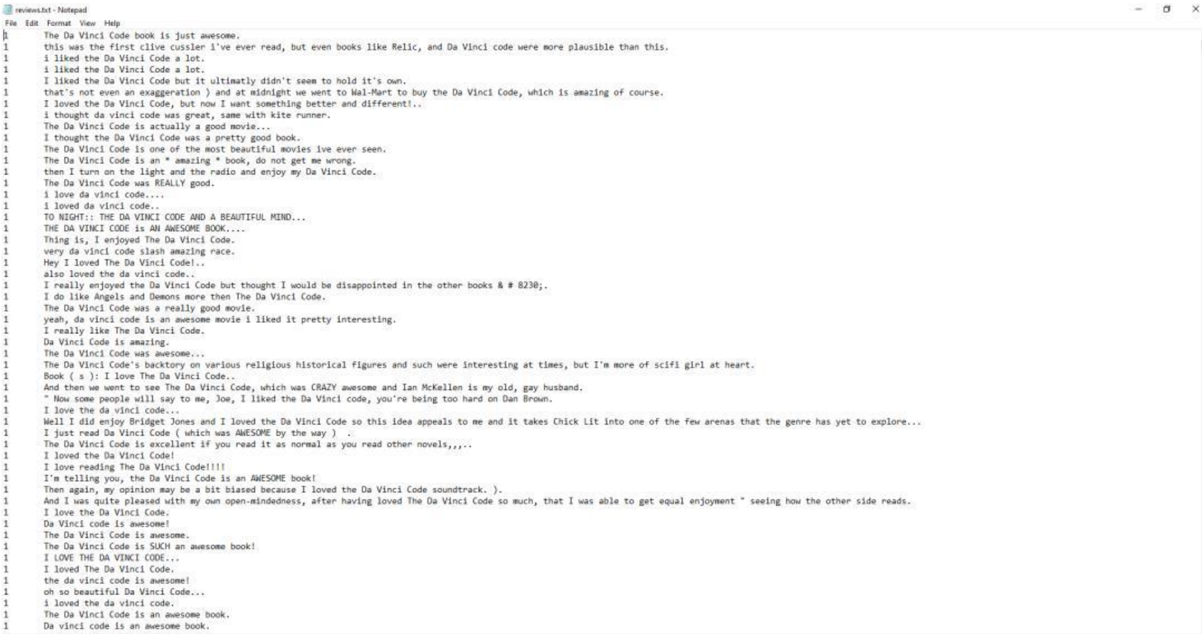
```
    {
     "data": {
      "text/plain": [
       "True"
      ]
     },
     "execution_count": 2,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "nltk.download(\"stopwords\")"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 3,
   "metadata": {},
   "outputs": [],
   "source": [
    "dataset = pd.read_csv('reviews.txt',sep = '\\t', names =['Reviews','Comments'])"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 4,
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/html": [
       "<div>\n",
       "<style scoped>\n",
       "    .dataframe tbody tr th:only-of-type {\n",
       "        vertical-align: middle;\n",
       "    }\n",
       "\n",
       "    .dataframe tbody tr th {\n",
       "        vertical-align: top;\n",
       "    }\n",
       "\n",
       "    .dataframe thead th {\n",
       "        text-align: right;\n",
       "    }\n",
       "</style>\n",
       "<table border=\"1\" class=\"dataframe\">\n",
       "  <thead>\n",
       "    <tr style=\"text-align: right;\">\n",
       "      <th></th>\n",
       "      <th>Reviews</th>\n",
       "      <th>Comments</th>\n",
       "    </tr>\n".
```



```
<!DOCTYPE html>
<html>
<head>
    <title>Movie Recommendation System</title>
   <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

  <link href="https://fonts.googleapis.com/css?family=IBM+Plex+Sans&display=swap" rel="stylesheet">
  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css" integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm" crossorigin="
  anonymous">
  <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/@tarekraafat/autocomplete.js@7.2.0/dist/css/autoComplete.min.css">

  <style>
    .movie {
        color: #fff;
        margin-left: auto;
        margin-right: auto;
    resize: none;
    }

    .btn-block{
        width: 15%;
        text-align: center;
        margin-left: auto;
        margin-right: auto;
        color: #e4e0e0;
    }

    #content {
        background-image: url("../static/image.png");
        background-color: #181818
    }


    .footer {
        color: #e4e0e0;
        text-align:right;
        position: fixed;
        bottom: 20px;
        right: 20px;
        width: 100%;
    }

    h1 {
        font-family: 'Netflix Sans', 'Helvetica Neue', Helvetica, Arial, sans-serif;
        color: #e50914;
        font-weight: bold;
        margin-top: 30px;
    }

    #autoComplete {
```



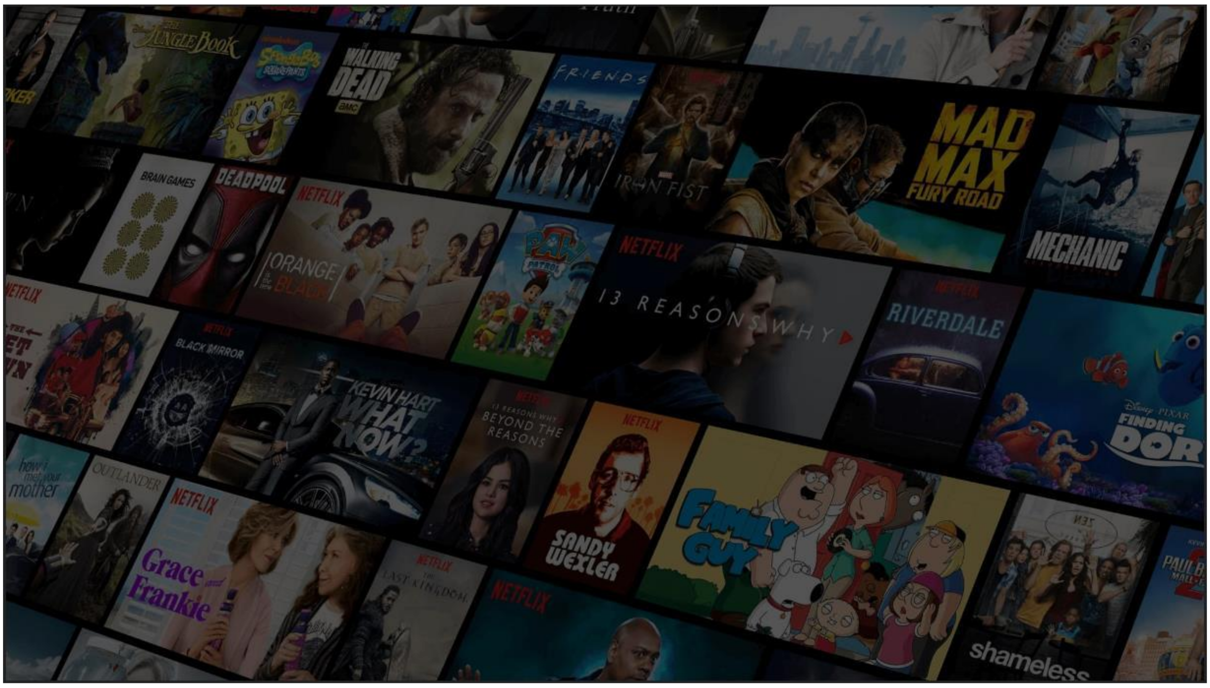

main_data.csv - Excel
ayushman mookherjee
File Home Insert Page Layout Formulas Data Review View Help Tell me what you want to do
G1 comb
director_r actor_1_n actor_2_n actor_3_n genres movie_tit comb
John Lasse Tom Hank Tim Allen Don Rickle Animatior toy story Tom Hanks Tim Allen Don Rickles John Lasseter Animation Comedy Family
Joe Johnst Robin Wil Jonathan Kirsten Du Adventure jumanji Robin Williams Jonathan Hyde Kirsten Dunst Joe Johnston Adventure Fantasy Family
Howard D Walter Ma Jack Lemn Ann-Marg Romance grumpier Walter Matthau Jack Lemmon Ann-Margret Howard Deutch Romance Comedy
Forest Wh Whitney H Angela Ba Loretta De Comedy D waiting to Whitney Houston Angela Bassett Loretta Devine Forest Whitaker Comedy Drama Romance
Charles Sh Steve Mar Diane Kea Martin Sh Comedy father of t Steve Martin Diane Keaton Martin Short Charles Shyer Comedy
Peter Hev Jonathan Brad Renf Rachael Le Action Ad tom and h Jonathan Taylor Thomas Brad Renfro Rachael Leigh Cook Peter Hewitt Action Adventure Drama Family
Peter Hya Jean-Clau Powers Bo Dorian Ha Action Ad sudden de Jean-Claude Van Damme Powers Boothe Dorian Harewood Peter Hyams Action Adventure Thriller
Martin Ca Pierce Bro Sean Bear Izabella Sc Adventure goldeneye Pierce Brosnan Sean Bean Izabella Scorupco Martin Campbell Adventure Action Thriller
Rob Reine Michael D Annette B Michael J. Comedy D the ameri Michael Douglas Annette Bening Michael J. Fox Rob Reiner Comedy Drama Romance
Mel Brook Leslie Nie Mel Brook Amy Yasb Comedy H dracula: d Leslie Nielsen Mel Brooks Amy Yasbeck Mel Brooks Comedy Horror
Simon We Kevin Bac Bob Hoski Bridget Fc Family An balto Kevin Bacon Bob Hoskins Bridget Fonda Simon Wells Family Animation Adventure
Oliver Sto Anthony H Joan Aller Powers Bo History Dr nixon Anthony Hopkins Joan Allen Powers Boothe Oliver Stone History Drama
Renny Har Geena Da Matthew Frank Lang Action Ad cutthroat Geena Davis Matthew Modine Frank Langella Renny Harlin Action Adventure
Martin Sco Robert De Sharon Sto Joe Pesci Drama Cri casino Robert De Niro Sharon Stone Joe Pesci Martin Scorsese Drama Crime
Allison An Tim Roth Antonio B Jennifer B Crime Cor four room Tim Roth Antonio Banderas Jennifer Beals Allison Anders Alexandre Rockwell Robert Rodriguez Quentin Tarantino Crime Comedy
Steve Oec Jim Carrey Ian McNei Simon Cal Crime Cor ace ventu Jim Carrey Ian McNeice Simon Callow Steve Oedekerk Crime Comedy Adventure
Joseph Ru Wesley Sr Woody Ha Jennifer L Action Co money tra Wesley Snipes Woody Harrelson Jennifer Lopez Joseph Ruben Action Comedy Crime
Barry Son John Trav Gene Hacl Rene Russ Comedy T get shorty John Travolta Gene Hackman Rene Russo Barry Sonnenfeld Comedy Thriller Crime
Jon Amiel Sigourney Holly Hun Will Patto Drama Thr copycat Sigourney Weaver Holly Hunter Will Patton Jon Amiel Drama Thriller
Richard Do Sylvester Antonio B Julianne M Action Ad assassins Sylvester Stallone Antonio Banderas Julianne Moore Richard Donner Action Adventure Crime Thriller
Victor Sah Mary Stee Sean Patri Lance Her Drama Far powder Mary Steenburgen Sean Patrick Flanery Lance Henriksen Victor Salva Drama Fantasy Sci-Fi Thriller
Mike Figgi Nicolas Ca Elisabeth Julian San Drama Ro leaving la Nicolas Cage Elisabeth Shue Julian Sands Mike Figgis Drama Romance
Lesli Linka Christina Rosie O'D Thora Birc Comedy D now and t Christina Ricci Rosie O'Donnell Thora Birch Lesli Linka Glatter Comedy Drama Family
Jean-Pierr Ron Perlm Dominiqu Judith Vit Fantasy Sc the city of Ron Perlman Dominique Pinon Judith Vittet Jean-Pierre Jeunet Marc Caro Fantasy Sci-Fi Adventure
Zhang Yim Gong Li Li Bao-Tia Wang Xiac Drama Cri shanghai t Gong Li Li Bao-Tian Wang Xiaoxiao Zhang Yimou Drama Crime
John N. Sr Michelle F George Dz Courtney Drama Cri dangerous Michelle Pfeiffer George Dzundza Courtney B. Vance John N. Smith Drama Crime
Terry Gilli Bruce Will Madelein Brad Pitt Sci-Fi Thri twelve mo Bruce Willis Madeleine Stowe Brad Pitt Terry Gilliam Sci-Fi Thriller Mystery
Jean-Jacq Craig Shef Elizabeth Tom Hulce Romance wings of c Craig Sheffer Elizabeth McGovern Tom Hulce Jean-Jacques Annaud Romance Adventure
Chris Noo Christine Miriam M Danny Ma Fantasy Dr babe Christine Cavanaugh Miriam Margolyes Danny Mann Chris Noonan Fantasy Drama Comedy Family
Christoph Emma Thc Jonathan Steven W History Dr carringtor Emma Thompson Jonathan Pryce Steven Waddington Christopher Hampton History Drama Romance
Tim Robbi Susan Sar Sean Penr Robert Pr Drama dead man Susan Sarandon Sean Penn Robert Prosky Tim Robbins Drama
Stephen L Peter Rez John McD Avi Hoffm Adventure across the Peter Reznick John McDonough Avi Hoffman Stephen Low Adventure History Drama Family
Amy Heck Alicia Silv Stacey Da Brittany M Comedy D clueless Alicia Silverstone Stacey Dash Brittany Murphy Amy Heckerling Comedy Drama Romance
Albert Hu Larenz Tat Keith Dav Chris Tuck Action Cri dead pres Larenz Tate Keith David Chris Tucker Albert Hughes Allen Hughes Action Crime Drama History
Paul W.S. Christoph Robin Sho Linden As Action Far mortal ko Christopher Lambert Robin Shou Linden Ashby Paul W.S. Anderson Action Fantasy


reviews.txt - Notepad
File Edit Format View Help
1 The Da Vinci Code book is just awesome.
1 this was the first clive cussler i've ever read, but even books like Relic, and Da Vinci code were more plausible than this.
1 i liked the Da Vinci Code a lot.
1 i liked the Da Vinci Code a lot.
1 I liked the Da Vinci Code but it ultimatly didn't seem to hold it's own.
1 that's not even an exaggeration) and at midnight we went to Wal-Mart to buy the Da Vinci Code, which is amazing of course.
1 I loved the Da Vinci Code, but now I want something better and different!..
1 i thought da vinci code was great, same with kite runner.
1 The Da Vinci Code is actually a good movie...
1 I thought the Da Vinci Code was a pretty good book.
1 The Da Vinci Code is one of the most beautiful movies ive ever seen.
1 The Da Vinci Code is an * amazing * book, do not get me wrong.
1 then I turn on the light and the radio and enjoy my Da Vinci Code.
1 The Da Vinci Code was REALLY good.
1 i love da vinci code....
1 i loved da vinci code..
1 TO NIGHT:: THE DA VINCI CODE AND A BEAUTIFUL MIND...
1 THE DA VINCI CODE is AN AWESOME BOOK....
1 Thing is, I enjoyed The Da Vinci Code.
1 very da vinci code slash amazing race.
1 Hey I loved The Da Vinci Code!..
1 also loved the da vinci code..
1 I really enjoyed the Da Vinci Code but thought I would be disappointed in the other books & # 8230;.
1 I do like Angels and Demons more then The Da Vinci Code.
1 The Da Vinci Code was a really good movie.
1 yeah, da vinci code is an awesome movie i liked it pretty interesting.
1 I really like The Da Vinci Code.
1 Da Vinci Code is amazing.
1 The Da Vinci Code was awesome...
1 The Da Vinci Code's backtory on various religious historical figures and such were interesting at times, but I'm more of scifi girl at heart.
1 Book (s): I love The Da Vinci Code..
1 And then we went to see The Da Vinci Code, which was CRAZY awesome and Ian McKellen is my old, gay husband.
1 " Now some people will say to me, Joe, I liked the Da Vinci code, you're being too hard on Dan Brown.
1 I love the da vinci code...
1 Well I did enjoy Bridget Jones and I loved the Da Vinci Code so this idea appeals to me and it takes Chick Lit into one of the few arenas that the genre has yet to explore...
1 I just read Da Vinci Code (which was AWESOME by the way) .
1 The Da Vinci Code is excellent if you read it as normal as you read other novels,,,..
1 I loved the Da Vinci Code!
1 I love reading The Da Vinci Code!!!!
1 I'm telling you, the Da Vinci Code is an AWESOME book!
1 Then again, my opinion may be a bit biased because I loved the Da Vinci Code soundtrack.).
1 And I was quite pleased with my own open-mindedness, after having loved The Da Vinci Code so much, that I was able to get equal enjoyment " seeing how the other side reads.
1 I love the Da Vinci Code.
1 Da Vinci code is awesome!
1 The Da Vinci Code is awesome.
1 The Da Vinci Code is SUCH an awesome book!
1 I LOVE THE DA VINCI CODE...
1 I loved The Da Vinci Code.
1 the da vinci code is awesome!
1 oh so beautiful Da Vinci Code...
1 i loved the da vinci code.
1 The Da Vinci Code is an awesome book.
1 Da vinci code is an awesome book.

main.py - C:\Users\Admin\Desktop\VIT\third year\winter sem\TARP\main.py (3.7.0)

File Edit Format Run Options Window Help

```python
import numpy as np
import pandas as pd
from flask import Flask, render_template, request
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics.pairwise import cosine_similarity
import json
import bs4 as bs
import urllib.request
import pickle
import requests
from datetime import date, datetime

# load the nlp model and tfidf vectorizer from disk
filename = 'nlp_model.pkl'
clf = pickle.load(open(filename, 'rb'))
vectorizer = pickle.load(open('tranform.pkl','rb'))

# converting list of string to list (eg. "["abc","def"]" to ["abc","def"])
def convert_to_list(my_list):
    my_list = my_list.split('","')
    my_list[0] = my_list[0].replace('["','')
    my_list[-1] = my_list[-1].replace('"]','')
    return my_list

# convert list of numbers to list (eg. "[1,2,3]" to [1,2,3])
def convert_to_list_num(my_list):
    my_list = my_list.split(',')
    my_list[0] = my_list[0].replace("[","")
    my_list[-1] = my_list[-1].replace("]","")
    return my_list

def get_suggestions():
    data = pd.read_csv('main_data.csv')
    return list(data['movie_title'].str.capitalize())

app = Flask(__name__)

@app.route("/")
@app.route("/home")
def home():
    suggestions = get_suggestions()
    return render_template('home.html',suggestions=suggestions)


@app.route("/recommend",methods=["POST"])
def recommend():
    # getting data from AJAX request
    title = request.form['title']
    cast_ids = request.form['cast_ids']
    cast_names = request.form['cast_names']
    cast_chars = request.form['cast_chars']
    cast_bdays = request.form['cast_bdays']
    cast_bios = request.form['cast_bios']
    cast_places = request.form['cast_places']
    cast_profiles = request.form['cast_profiles']
    imdb_id = request.form['imdb_id']
    poster = request.form['poster']
    genres = request.form['genres']
    overview = request.form['overview']
    vote_average = request.form['rating']
    vote_count = request.form['vote_count']
```



sentiment.ipynb - Notepad

File Edit Format View Help


```
    {
     "data": {
      "text/plain": [
       "True"
      ]
     },
     "execution_count": 2,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "nltk.download(\"stopwords\")"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 3,
   "metadata": {},
   "outputs": [],
   "source": [
    "dataset = pd.read_csv('reviews.txt',sep = '\\t', names =['Reviews','Comments'])"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 4,
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/html": [
       "<div>\n",
       "<style scoped>\n",
       "    .dataframe tbody tr th:only-of-type {\n",
       "        vertical-align: middle;\n",
       "    }\n",
       "\n",
       "    .dataframe tbody tr th {\n",
       "        vertical-align: top;\n",
       "    }\n",
       "\n",
       "    .dataframe thead th {\n",
       "        text-align: right;\n",
       "    }\n",
       "</style>\n",
       "<table border=\"1\" class=\"dataframe\">\n",
       "  <thead>\n",
       "    <tr style=\"text-align: right;\">\n",
       "      <th></th>\n",
       "      <th>Reviews</th>\n",
       "      <th>Comments</th>\n",
       "    </tr>\n".
```

Code:

1) For sentiment analysis

```
# In[15]:  import pandas as pd
           import numpy as np
           import nltk
           from nltk.corpus import stopwords
           from sklearn.feature_extraction.text import TfidfVectorizer
           from sklearn.model_selection import train_test_split
           from sklearn import naive_bayes
           from sklearn.metrics import roc_auc_score,accuracy_score
           import pickle
# In[2]:   nltk.download("stopwords")
# In[3]:   dataset = pd.read_csv('reviews.txt',sep = '\t', names =['Reviews','Comments'])
# In[4]:   dataset
# In[5]:   stopset = set(stopwords.words('english'))
# In[6]:   vectorizer = TfidfVectorizer(use_idf = True,lowercase = True, strip_accents='ascii',
           stop_words=stopset)
# In[16]: X = vectorizer.fit_transform(dataset.Comments)
           y = dataset.Reviews
           pickle.dump(vectorizer, open('tranform.pkl', 'wb'))
# In[17]:  X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20, random_state=42)
# In[18]:  clf = naive_bayes.MultinomialNB()
           clf.fit(X_train,y_train)
# In[19]:  accuracy_score(y_test,clf.predict(X_test))*100
# In[20]:  clf = naive_bayes.MultinomialNB()
           clf.fit(X,y)
# In[21]:  accuracy_score(y_test,clf.predict(X_test))*100
# In[22]:  filename = 'nlp_model.pkl'
           pickle.dump(clf, open(filename, 'wb'))
```

2) For data preprocessing

```
# In[1]:  import pandas as pd
          import numpy as np
# In[2]:  data = pd.read_csv('movie_metadata.csv')
# In[3]:  data.head(10)
```

```
# In[4]:  data.shape
# In[5]:  data.columns
# In[6]:  # we have movies only upto 2016
          import matplotlib.pyplot as plt
          data.title_year.value_counts(dropna=False).sort_index().plot(kind='barh',figsize=(15,16))
          plt.show()
# In[8]:  # recommendation will be based on these features only
          data =
data.loc[:,['director_name','actor_1_name','actor_2_name','actor_3_name','genres','movie_title']]
# In[9]:  data.head(10)
# In[11]: data['actor_1_name'] = data['actor_1_name'].replace(np.nan, 'unknown')
          data['actor_2_name'] = data['actor_2_name'].replace(np.nan, 'unknown')
          data['actor_3_name'] = data['actor_3_name'].replace(np.nan, 'unknown')
          data['director_name'] = data['director_name'].replace(np.nan, 'unknown')
# In[12]: data
# In[15]: data['genres'] = data['genres'].str.replace('|', ' ')
# In[16]: data
# In[17]: data['movie_title'] = data['movie_title'].str.lower()
# In[18]: # null terminating char at the end
          data['movie_title'][1]
# In[19]: # removing the null terminating char at the end
          data['movie_title'] = data['movie_title'].apply(lambda x : x[:-1])
# In[20]: data['movie_title'][1]
# In[21]: data.to_csv('data.csv',index=False)
```

3) Similarity Score of Movies

```
import numpy as np
import pandas as pd
from flask import Flask, render_template, request
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics.pairwise import cosine_similarity
import json
import bs4 as bs
import urllib.request
import pickle
import requests
```

```
# load the nlp model and tfidf vectorizer from disk
filename = 'nlp_model.pkl'
clf = pickle.load(open(filename, 'rb'))
vectorizer = pickle.load(open('tranform.pkl','rb'))
def create_similarity():
data = pd.read_csv('main_data.csv')
# creating a count matrix
cv = CountVectorizer()
count_matrix = cv.fit_transform(data['comb'])
# creating a similarity score matrix
similarity = cosine_similarity(count_matrix)
return data,similarity
def rcmd(m):
    m = m.lower()
    try:
        data.head()
        similarity.shape
    except:
        data, similarity = create_similarity()
        if m not in data['movie_title'].unique():
            return('Sorry! The movie you requested is not in our database. Please check the spelling or
        try with some other movies')
    else:
        i = data.loc[data['movie_title']==m].index[0]
        lst = list(enumerate(similarity[i]))
        lst = sorted(lst, key = lambda x:x[1] ,reverse=True)
        lst = lst[1:11] # excluding first item since it is the requested movie itself
        l = []
        for i in range(len(lst)):
            a = lst[i][0]
            l.append(data['movie_title'][a])
        return l
# converting list of string to list (eg. "["abc","def"]" to ["abc","def"])
def convert_to_list(my_list):
    my_list = my_list.split('","')
    my_list[0] = my_list[0].replace('["','')
```

```
    my_list[-1] = my_list[-1].replace('"]','')
    return my_list
def get_suggestions():
    data = pd.read_csv('main_data.csv')
    return list(data['movie_title'].str.capitalize())
app = Flask(__name__)
@app.route("/")
@app.route("/home")
def home():
    suggestions = get_suggestions()
    return render_template('home.html',suggestions=suggestions)
@app.route("/similarity",methods=["POST"])
def similarity():
    movie = request.form['name']
    rc = rcmd(movie)
    if type(rc)==type('string'):
        return rc
    else:
        m_str="---".join(rc)
        return m_str
@app.route("/recommend",methods=["POST"])
def recommend():
    # getting data from AJAX request
    title = request.form['title']
    cast_ids = request.form['cast_ids']
    cast_names = request.form['cast_names']
    cast_chars = request.form['cast_chars']
    cast_bdays = request.form['cast_bdays']
    cast_bios = request.form['cast_bios']
    cast_places = request.form['cast_places']
    cast_profiles = request.form['cast_profiles']
    imdb_id = request.form['imdb_id']
    poster = request.form['poster']
    genres = request.form['genres']
    overview = request.form['overview']
    vote_average = request.form['rating']
```

```
vote_count = request.form['vote_count']
release_date = request.form['release_date']
runtime = request.form['runtime']
status = request.form['status']
rec_movies = request.form['rec_movies']
rec_posters = request.form['rec_posters']
# get movie suggestions for auto complete
suggestions = get_suggestions()
# call the convert_to_list function for every string that needs to be converted to list
rec_movies = convert_to_list(rec_movies)
rec_posters = convert_to_list(rec_posters)
cast_names = convert_to_list(cast_names)
cast_chars = convert_to_list(cast_chars)
cast_profiles = convert_to_list(cast_profiles)
cast_bdays = convert_to_list(cast_bdays)
cast_bios = convert_to_list(cast_bios)
cast_places = convert_to_list(cast_places)
# convert string to list (eg. "[1,2,3]" to [1,2,3])
cast_ids = cast_ids.split(',')
cast_ids[0] = cast_ids[0].replace("[","")
cast_ids[-1] = cast_ids[-1].replace("]","")
# rendering the string to python string
for i in range(len(cast_bios)):
    cast_bios[i] = cast_bios[i].replace(r'\n', '\n').replace(r'\"','\"')
# combining multiple lists as a dictionary which can be passed to the html file so that it can be
processed easily and the order of information will be preserved
movie_cards = {rec_posters[i]: rec_movies[i] for i in range(len(rec_posters))}
casts = {cast_names[i]:[cast_ids[i], cast_chars[i], cast_profiles[i]] for i in
range(len(cast_profiles))}
cast_details = {cast_names[i]:[cast_ids[i], cast_profiles[i], cast_bdays[i],
cast_places[i], cast_bios[i]] for i in range(len(cast_places))}
# web scraping to get user reviews from IMDB site
sauce = urllib.request.urlopen('https://www.imdb.com/title/{}/reviews?ref_=tt_ov_rt'.form
at(imdb_id)).read()
soup = bs.BeautifulSoup(sauce,'lxml')
soup_result = soup.find_all("div",{"class":"text show-more__control"})
```

```
    reviews_list = [] # list of reviews
    reviews_status = [] # list of comments (good or bad)
    for reviews in soup_result:
        if reviews.string:
            reviews_list.append(reviews.string)
            # passing the review to our model
            movie_review_list = np.array([reviews.string])
            movie_vector = vectorizer.transform(movie_review_list)
            pred = clf.predict(movie_vector)
            reviews_status.append('Good' if pred else 'Bad')
# combining reviews and comments into a dictionary
movie_reviews = {reviews_list[i]: reviews_status[i] for i in
range(len(reviews_list))}
# passing all the data to the html file
return
render_template('recommend.html',title=title,poster=poster,overview=overview,
vote_average=vote_average,vote_count=vote_count,release_date=release_date,runtime=runtime,
status=status, genres=genres, movie_cards=movie_cards,reviews=movie_reviews,casts=casts,
cast_details=cast_details)
```

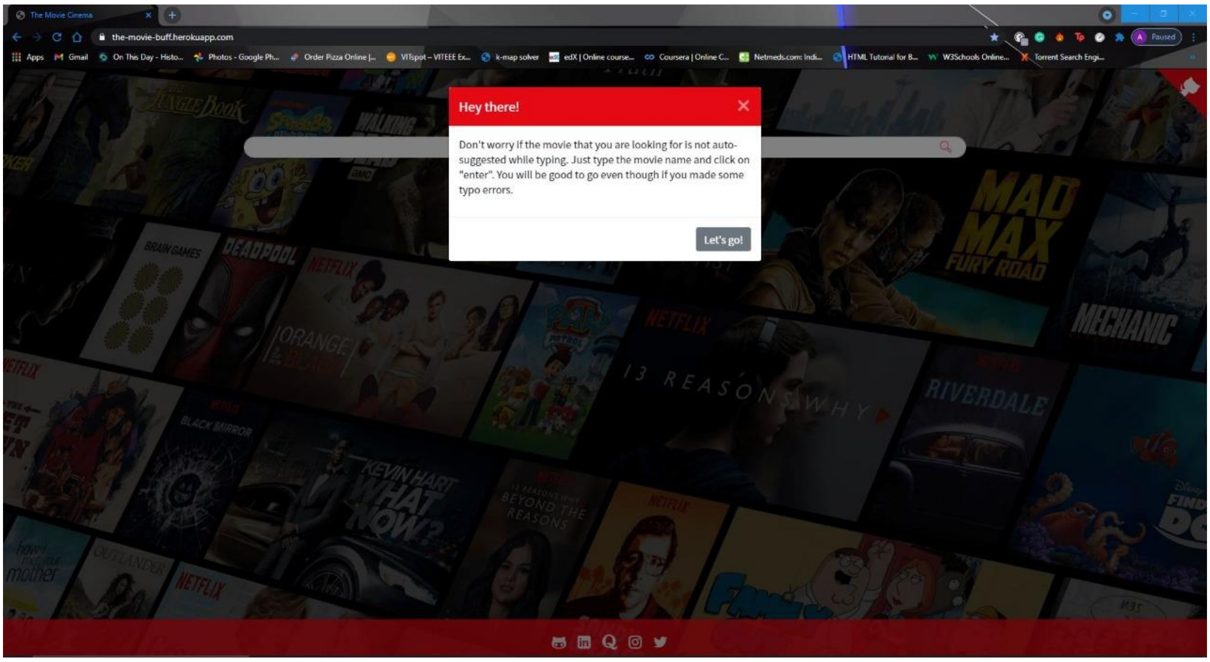


Figure 1: Loading page of our website which is named as "The movie buff" is shown above as hosted on herokuapp. This page contains the basic information required by the user to use the movie recommender system. The user now needs to click on the "lets go" action button and proceed further.

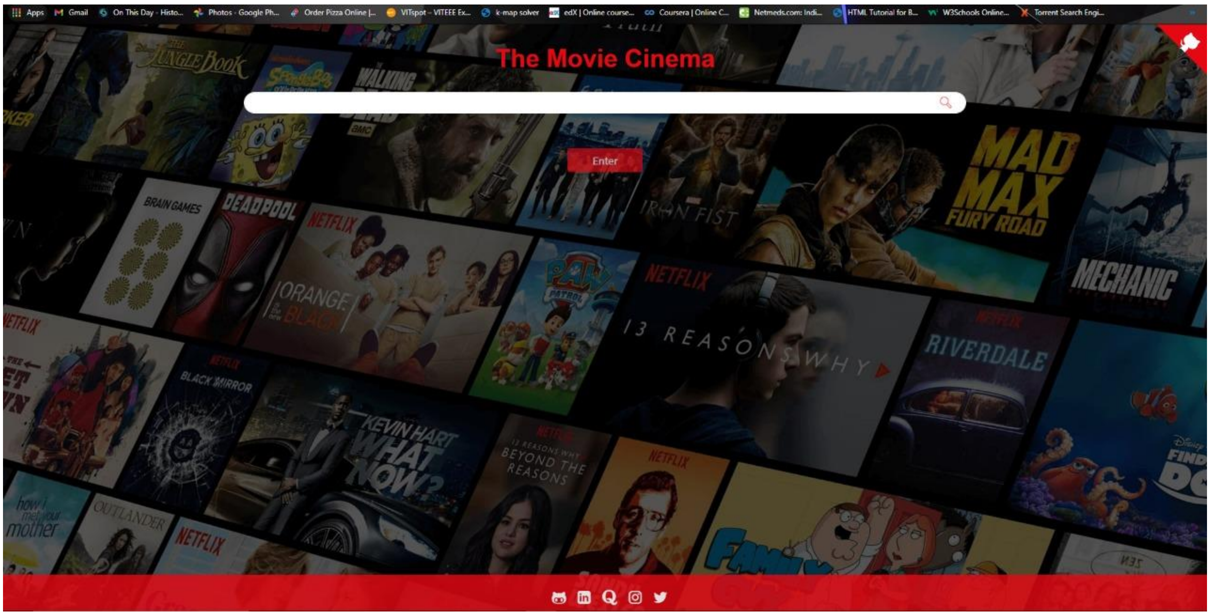


Figure 2: After the user clicks on the "lets go" action button from the previous home page, they are directed to the main search bar as shown above. Here the user enters the name of the movie which they have liked before or the name of the movie as per which they would like the recommendations.

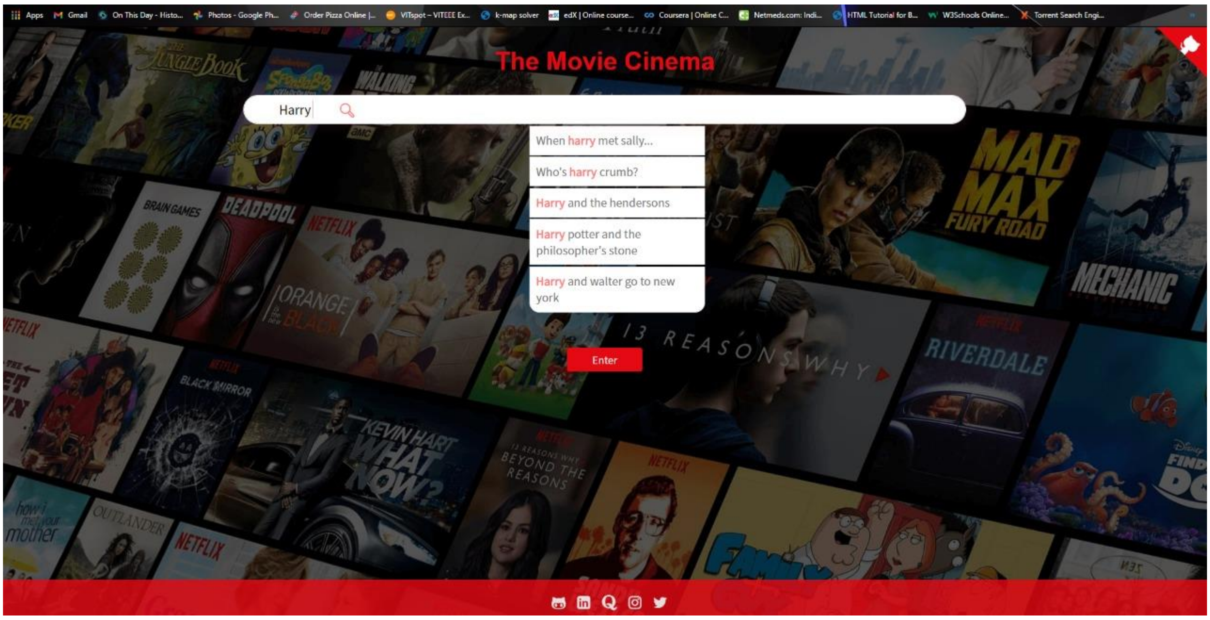


Figure 3: The predictive algorithm which was used when implementing ML now comes into picture as you can see the website tries to help the user by auto filling and giving suggestions. So, if the user wanted to type "harry potter and the philosopher's stone" the website has already predicted movies which has the keyword token as "harry" in them and suggests them. Here the suggestions from the website will keep on improving as the user keeps on typing and using the website further due to the implementation of sentiment analysis and regression models.

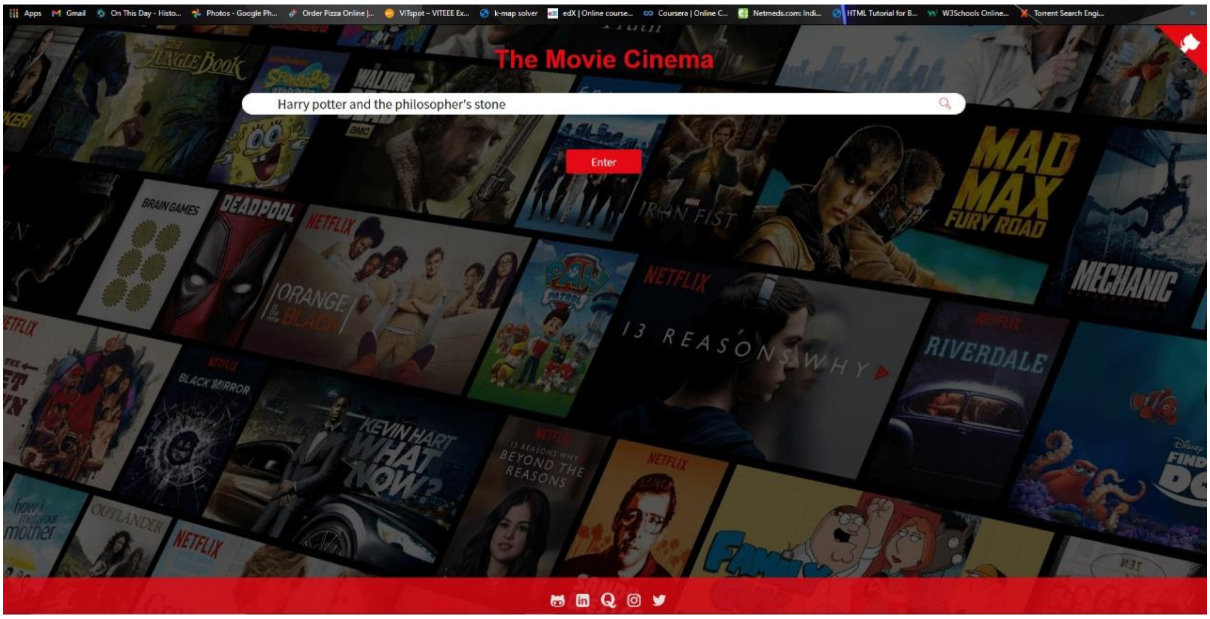


Figure 4: After typing the movie the user has to select the enter button and proceed further.

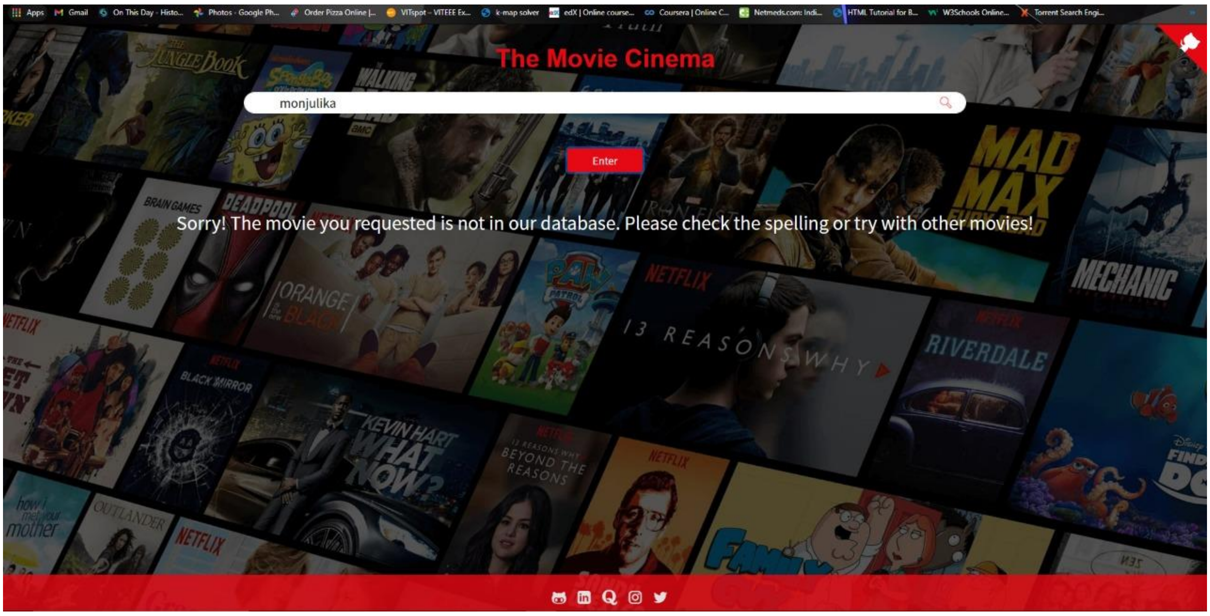


Figure 5: If by chance the movie requested by the user does not exist in the database, an error message is shown as above and in this case the user can check the spelling of the movie and try again. If it still does not work the user is requested to try with another movie of similar genre or another movie entirely.

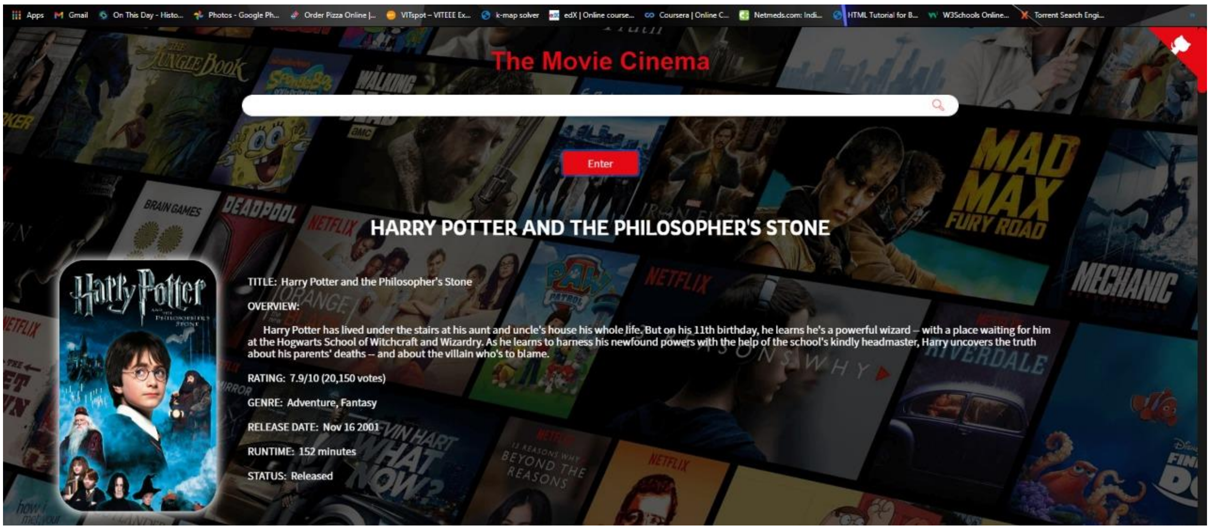


Figure 6: if the movie is found in our database, the details of the movie is showed above. An overview of the movie along with its IMDb rating, genre, status and run-time are shown.

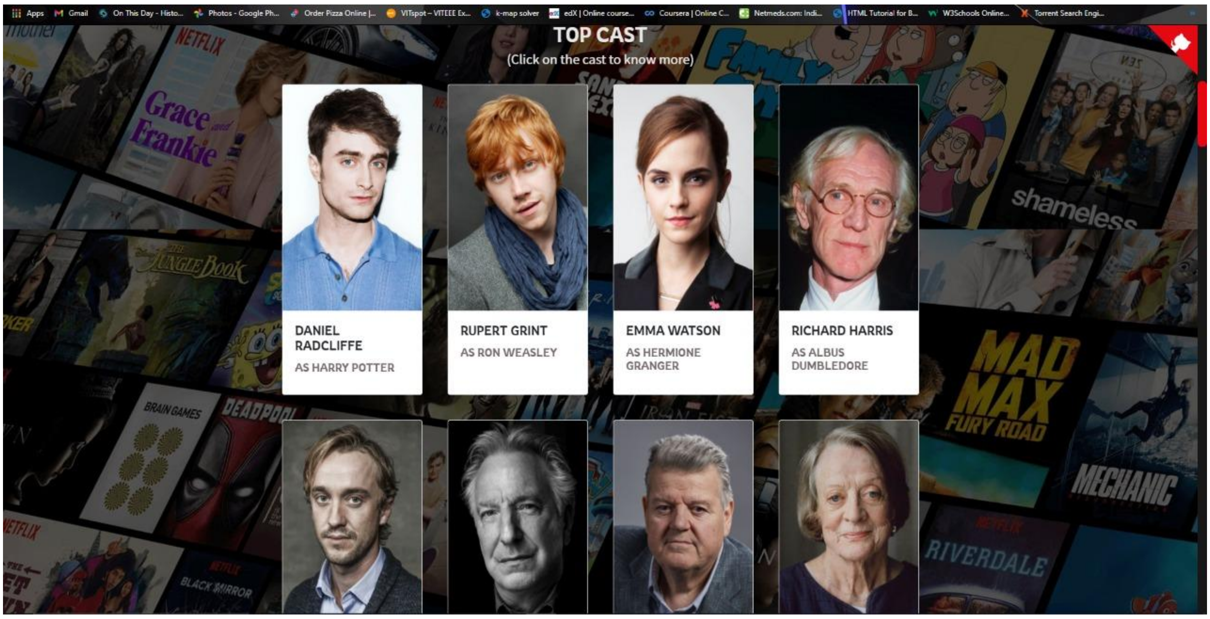


Figure 7: for more information, the details of the cast are also shown.

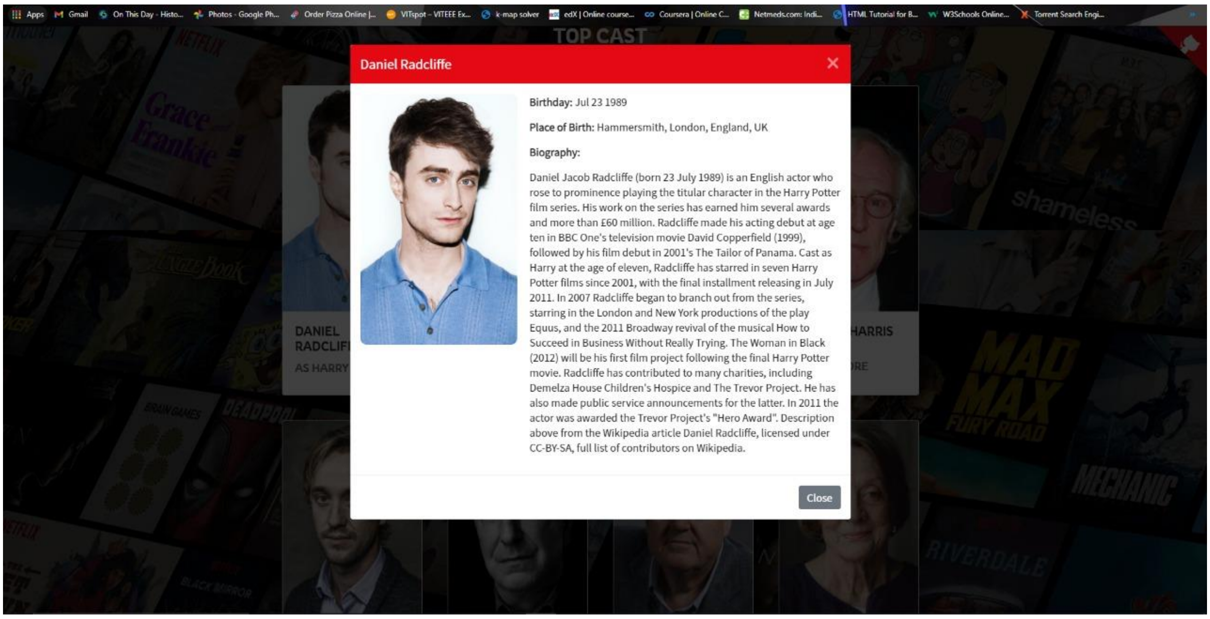


Figure 8: for more details about the actor/actress the user can click on their respective photos and get the information as shown above. all the information has been fetched by wikipedia with the help of web scraping and integrated in the website using the flask software.

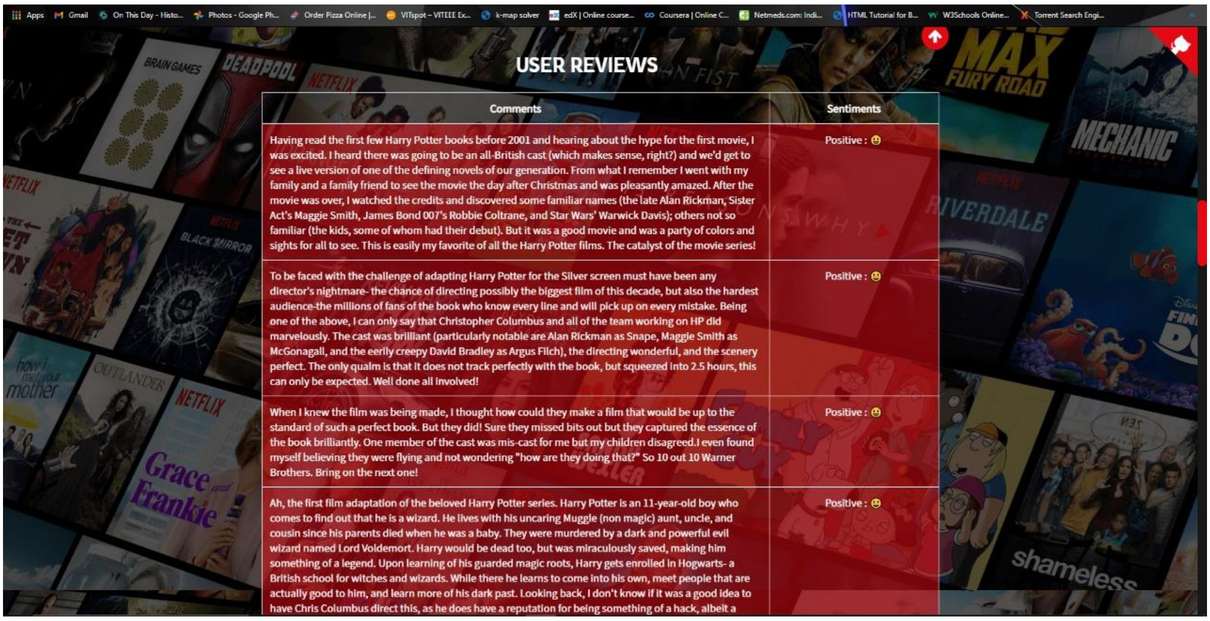


Figure 9: Now comes the main part of our project. The user reviews of that particular movie are fetched and shown above. Sentiment analysis and the adjectives used in the reviews are weighted and the algorithm comes up with a system to show whether the review is a positive review or a negative review with either a smiley face or a sad/angry face.

for example if the site shows 10 reviews, 7 of them being positive and 3 of them being negative. The algorithm then calculates a weighted average score of those and proceeds to make recommendations of similar movies.

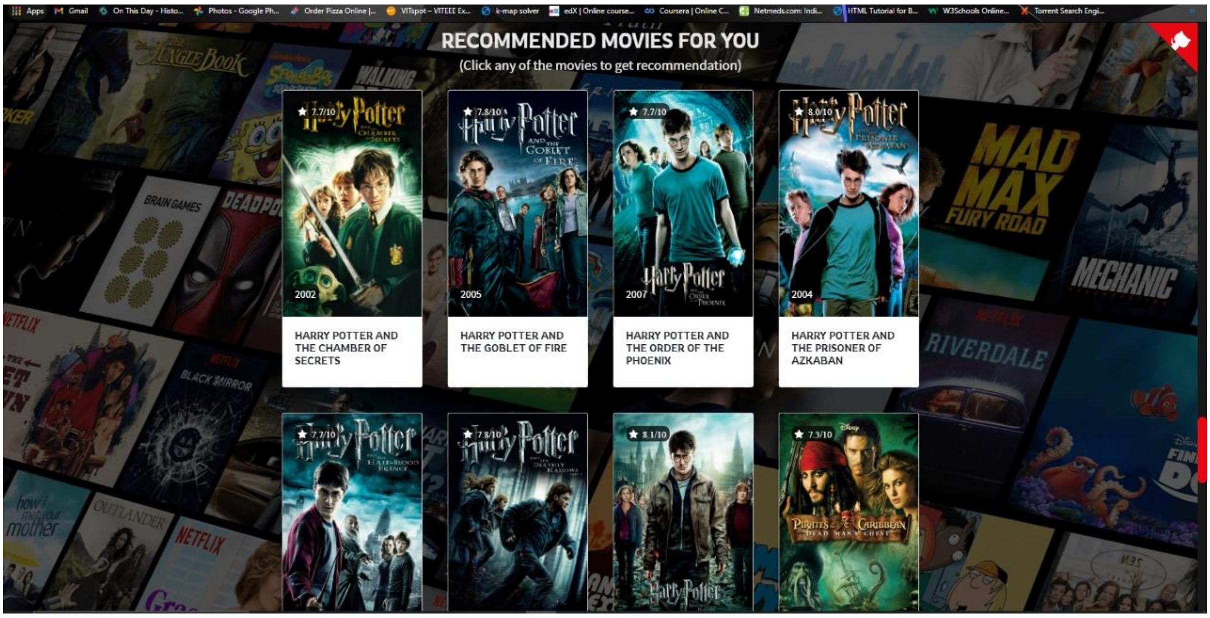


Figure 10

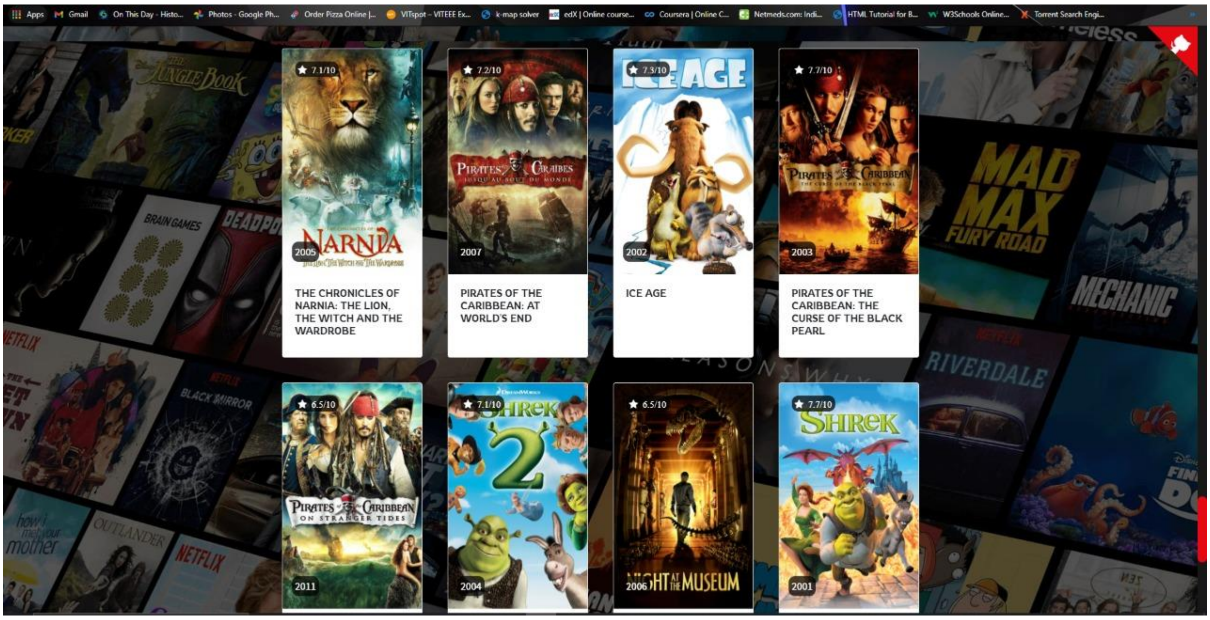


Figure 11

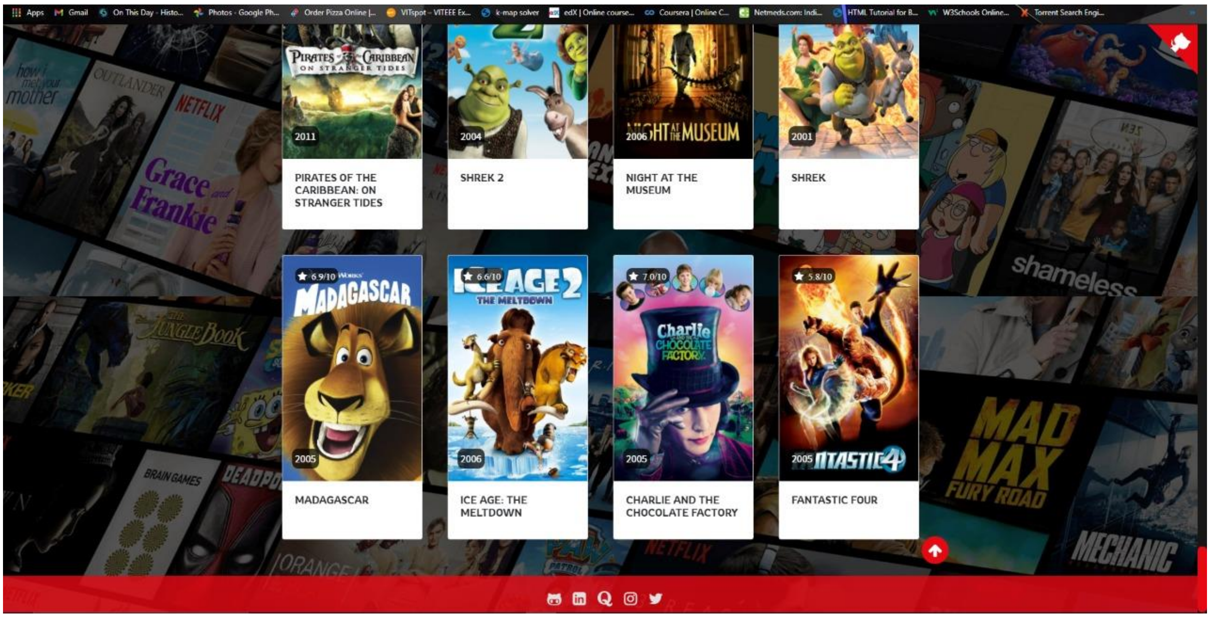


Figure 12

As you can see the algorithm now goes on ahead to give the user recommendations to various other movies factoring in their IMDb scores, their release date and of course their reviews as well.

The user can again select one of these recommendations and go through the whole process again. The algorithm improves/fine tunes itself the more number of times the user uses the website at one go. If the user is satisfied with the recommendation and chooses to close the website, the session is then terminated and the algorithm sort resets itself and gets ready for the next user.

Future Scope:

Recommendation systems have a huge scope and a market for use as well. In today's day and age people ask for recommendations not only for movies but also for places to eat, places to visit, investment purposes and many more. Our website can be modified and our algorithm can hence be tweaked as per the requirements and give recommendations as per the user's requirements.

The website can be made into a real time dynamic site with active databases and integrated with current services like Zomato, Swiggy for food services and orders. Netflix, Prime video or Hotstar for movies and TV shows. The website can be made into a stock trading tracker which would be used to determine the performances of the stocks available in the database taken form the BSE website and then with predictive algorithms the user could be recommended those stocks to invest and those stocks to avoid investing in.

The market for recommendation systems is huge and flexible as well. As the needs of the people increases the need of a recommendation system comes into the picture as well. The more there are product based companies out there and and the increase in the products they sell, more would be the use of any recommendation system.

REFERENCES:

[1] Chumki Basu, Haym Hirsh, William Cohen, 1998. Recommendation as Classification: Using Social and Content-Based Information in Recommendation.

[2] Yunhong Xu, Xitong Guo, Jinxing Hao, Jian Ma, Raymond Y.K. Lau, Wei Xu, 2012. Combining social network and semantic concept analysis for personalized academic researcher recommendation.

[3] Vincent W. Zheng, Bin Cao, Yu Zheng, Xing Xie and Qiang Yang, 2010. Collaborative Filtering Meets Mobile Recommendation: A User-centered Approach.

[4] Toon De Pessemier, Tom Deryckere, Luc Martens, 2009. Context Aware Recommendations for User-generated Content on a Social Network Site.

[5] Yibo Wang, Mingming Wang and Wei Xu, 2018. A Sentiment-Enhanced Hybrid Recommender System for Movie Recommendation: A Big Data Analytics Framework. In Hindawi Wireless Communications and Mobile Computing Volume 2018, Article ID 8263704.

[6] Sudhanshu Kumar, Kanjar De and Partha Pratim Roy, 2020. Movie Recommendation System Using Sentiment Analysis from Microblogging Data.

[7] Rahul Katarya, Om Prakash Verma, 2016. An effective collaborative movie recommender system with cuckoo search. In Egyptian Informatics Journal.

[8] Zan Wang, Xue Yu, Nan Feng, Zhenhua Wang, 2014. An improved collaborative movie recommendation system using computational intelligence. In Journal of Visual Languages and Computing 25 (2014) 667–675.

[9] Jie Lu, Dianshuang Wu, Mingsong Mao, Wei Wang, Guangquan Zhang, 2015. Recommender system application developments: A survey. In Decision Support Systems 74 (2015) 12-32.

[10] Rahul Katarya, Om Prakash Verma, 2016. Recent developments in affective recommender systems.