

MOVIE RECOMMENDER SYSTEM

A PROJECT REPORT

Submitted by

Sahil Anwar [Reg No: RA2011042010131]

Uday Kumar [Reg No: RA2011042010128]

Mayank Sandilya [Reg No: RA2011042010094]

Huzaiifa Qidwai [Reg No: RA2011042010105]

Under the Guidance of

Dr.K. Dhanasekaran

(Assistant Professor, Department of Data Science and Business System)

In partial fulfilment of the Requirements for the Degree of

BACHELOR OF TECHNOLOGY



**DEPARTMENT OF COMPUTER SCIENCE AND
ENGINEERING**

**FACULTY OF ENGINEERING AND TECHNOLOGY
SRM INSTITUTE OF SCIENCE AND TECHNOLOGY
KATTANKULATHUR – 603203**

MAY 2024



SRM INSTITUTE OF SCIENCE AND TECHNOLOGY
KATTANKULATHUR-603203

BONAFIDE CERTIFICATE

Certified that this project report titled “**Movie Recommender system**” is the bonafide work of “**Sahil Anwar [Reg No: RA2011042010131], Uday Kumar [Reg No: RA2011042010128], Mayank Sandilya [Reg No: RA2011042010094], Huzaifa Qidwai [Reg No: RA2011042010105]** who carried out the project work under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion for this or any other candidate.

Dr. K. Dhanasekaran
SUPERVISOR
Assistant Professor
Department of Data Science and
Business System

Dr. M Lakshmi
HEAD OF THE DEPARTMENT
Department of Data Science and
Business System

Signature of Internal Examiner

Signature of External Examiner

ACKNOWLEDGEMENT

We express our humble gratitude to **Dr. C. Muthamizhchelvan**, Vice Chancellor (I/C), SRM Institute of Science and Technology, for the facilities extended for the project work and his continued support.

We extend our sincere thanks to **Dr. Revathi Venkataraman**, Professor & Chairperson, School of Computing, SRM Institute of Science and Technology, for her invaluable support.

We wish to thank **Dr. M Lakshmi**, Head of Department of Data Science and Business System, SRM Institute of Science and Technology, for her valuable suggestions and encouragement throughout the period of the project work.

We are extremely grateful to our Academic Advisor **Dr.E.Sasikala**, Assistant Professor, Department of Data Science and Business System, SRM Institute of Science and Technology, for her great support at all the stages of project work.

We register our immeasurable thanks to our Faculty Advisor, **DR .S.Jeeva**, Assistant Professor, Department of Data Science and Business System, SRM Institute of Science and Technology, for leading and helping us to complete our course.

Our inexpressible respect and thanks to my guide, **Dr.K. Dhanasekaran**, Assistant Professor, Department of Data Science and Business System SRM Institute of Science and Technology, for providing me an opportunity to pursue my project under her mentorship. She provided me the freedom and support to explore the research topics of my interest. Her passion for solving real problems and making a difference in the world has always been inspiring.

We sincerely thank staff and students of the Computer Science and Engineering Department, SRM Institute of Science and Technology, for their help during my research. Finally, we would like to thank my parents, our family members and our friends for their unconditional love, constant support and encouragement.

Sahil Anwar [Reg No: RA2011042010131]

Uday Kumar [Reg No: RA2011042010128]

Mayank Sandilya [Reg No: RA2011042010094]

Huzaifa Qidwai [Reg No: RA2011042010105]

INDEX

	Page No.
Abstract	3
1. Introduction	4
2. Releated Work	5
3. Literature Review	6
4. Modules Used	7
5. Implementation	8
6. Code & Output	9
7. Conclusion	14
8. References	14

ABSTRACT

This project involves the creation of Movie Recommender system can suggest users to select Movie according to their interest and demand.

A recommendation system is a system that provides suggestions to users for certain resources like books, movies, songs, etc., based on some data set. Movie recommendation systems usually predict what movies a user will like based on the attributes present in previously liked movies. Such recommendation systems are beneficial for organizations that collect data from large amounts of customers, and wish to effectively provide the best suggestions possible.

A lot of factors can be considered while designing a movie recommendation system like the genre of the movie, actors present in it or even the director of the movie. The systems can recommend movies based on one or a combination of two or more attributes. In this paper, the recommendation system has been built on the type of genres that the user might prefer to watch. The approach adopted to do so is content-based filtering using genre correlation. The dataset used for the system is Movie Lens dataset. The data analysis tool used is R.

From a user's perspective, they are catered to fulfil the user's needs in the shortest time possible. For example, the type of content you watch on Netflix or Hulu. A person who likes to watch only Korean drama will see titles related to that only but a person who likes to watch Action-based titles will see that on their home screen.

From an organization's perspective, they want to keep the user as long as possible on the platform so that it will generate the most possible profit for them. With better recommendations, it creates positive feedback from the user as well. What good it will be to the organization to have a library of 500K+ titles when they cannot provide proper recommendations?

Recommendations are a great way to keep you watching but for Raghu the recommendations he gets wrong. But how? Well, as you know that recommendation systems are catered for a user but not for multiple users. Raghu lives in a joint family and everyone uses a single system to watch what they want. While OTT platforms give you a choice of adding multiple profiles but everyone else has already taken those and he is left with a single profile to share with his grandparents. So, Raghu decides to create his movie recommendation system. Before getting started he should understand the different types of recommendation systems.

INTRODUCTION

In this age of the Internet, the quantity of data transactions that happen every minute has increased exponentially. The huge amount of data has dramatically increased with the number of users on the Internet. However, not all the data available on the Internet is of use or provides satisfactory results to the users. Data in such huge volumes often turns out to be inconsistent and without proper processing of this information, it gets wasted. In such cases, users have to run their search multiple times before they finally obtain what they were originally looking for. To solve this problem, researchers have come up with recommendation systems. A recommendation system provides relevant information to the users by taking into account their past preferences. Data is filtered and personally customized as per the user requirements. With more and more data available on the Internet, recommendation systems have become really popular, due to their effectiveness in providing information in a short time-span. Recommender systems have been developed in various areas such as music, movies, news, and products in general. In today's age, a majority of organizations implement recommendation systems for fulfilling customer requirements. LinkedIn, Amazon, and Netflix are just a few to name. LinkedIn recommends relevant connections of the people the user might know among the millions that are subscribed on the portal. This way, the user does not have to run extensive searches for people manually. Amazon recommendation systems work such that they suggest correlated items that the customers can purchase. If a certain customer prefers buying books from the shopping portal, Amazon provides suggestions related to any new arrivals in previously preferred categories. In a very similar way, Netflix takes into account the types of shows that a customer watches, and provides recommendations similar to those. By the method in which recommendation systems work, they can be broadly classified into three categories—Content-based, Collaborative and Hybrid approach. A content-based recommendation system considers the user's past behavior and identifies patterns in them to recommend items that are similar to them. Collaborative filtering analyses the user's previous experiences and ratings and correlates it with other users. Based on the ones that have the most similarity, recommendations are made. Both content-based and collaborative-based filtering have their own limitations. To overcome this, researchers suggested a hybrid approach which would combine the advantages of both the methods. This paper suggests a content-based recommendation system that utilizes genre correlation. The dataset used for this purpose is a Movie Lens dataset containing 9126 movies which are classified according to genres. There are a total of 11 genres. The ratings for these movies have been collected from 671 users. By taking into account the movies which received high ratings from the users, movies containing similar genres are recommended to them.

RELATED WORK

Recommender systems are broadly classified into three types—collaborative filtering systems, content-based filtering systems, and hybrid systems [3]. Collaborative systems utilize inputs from various users and run various comparisons on these inputs [3]. They build models from the past behavior of the users [1]. Movie recommendation systems, for example, utilize the ratings of users for various movies [2], and attempt to find other like-minded users, and recommend movies they have rated well [3]. Collaborative filtering systems have two approaches—memory-based approaches and model-based approaches [3]. Memory-based approaches continuously analyze user data in order to make recommendations [3]. As they utilize the user ratings, they gradually improve in accuracy over time [3]. They are domain-independent and do not require content analysis [3]. Model-based approaches develop a model of a user's behavior and then use certain parameters to predict future behavior [3]. The use of partitioning-based algorithms also leads to better scalability and accuracy.

Content-based filtering systems analyze documents or preferences given by a particular user, and attempt to build a model around this data [3]. They make use of a user's particular interests and attempt to match a user's profile to the attributes possessed by the various content objects to be recommended [3]. They have the added disadvantage of requiring enough data to build a reliable classifier [1]. Content-based filtering systems are divided into three methods—wrapper methods, filter methods, and embedded methods [3]. Wrapper methods divide the features into subsets, run analysis on these subsets and then evaluate which of these subsets seems the most promising [3]. Filter methods use heuristic methods to rate features on their content [3]. Both these methods are independent of the algorithms used. In contrast, embedded methods are coupled with the algorithm used—feature selection is performed during the training phase [3]. Hybrid systems combine collaborative and content-based filtering systems, in order to optimize the recommender systems, and reduce the drawbacks present in each of the two methods [3].

LITERATURE REVIEW

Literature review is an essential component of any research project, including the development of an AI thinking capability of suggestion system and recommender system according to previous data. In recent years, there has been an explosion of research in the field of artificial intelligence, with numerous studies focusing on various applications of AI.

One key area of AI research that is relevant to this project is natural language processing (NLP). NLP is concerned with developing algorithms and techniques to enable computers to understand, interpret, and generate human language. This is particularly relevant to an AI desktop assistant, which must be able to understand and respond to user commands and queries.

Another area of AI research that is relevant to this project is machine learning. Machine learning algorithms enable computers to learn and improve their performance on a specific task over time. In the context of an AI recommender system, machine learning algorithms could be used to improve the accuracy of speech recognition, enable more advanced natural language processing capabilities, and personalize the user experience based on their previous interactions with the system.

Several AI recommended and suggestion system have already been developed and are available on the market, such as Apple's Siri, Amazon's Alexa, and Google Assistant. These systems use a combination of natural language processing, machine learning, and other AI techniques to enable users to perform a wide range of tasks, from setting reminders and playing movie on interest and also upon previous data.

Overall, there is a wealth of research on various AI techniques and applications that can inform the development of an AI recommender system. By building on this existing research, it is possible to create a powerful and effective system that can automate the day-to-day tasks of developers and PC users.

MODULES USED

1. `pyttsx3`: A Python library that provides a simple interface for text-to-speech conversion using different speech engines. It is used here to initialize and configure the text-to-speech engine, select the voice, and convert text to speech.
2. `speech_recognition`: A Python library that provides an easy way to recognize speech using different APIs, including Google Speech Recognition. It is used here to listen to the user's voice input and convert it to text.
3. `datetime`: A Python library that provides classes for working with dates, times, and time intervals. It is used here to get the current time and greet the user accordingly.
4. `wikipedia`: A Python library that makes it easy to access and parse data from Wikipedia. It is used here to search for and retrieve information from Wikipedia based on the user's input.
5. `webbrowser`: A Python library that provides a high-level interface to allow displaying Web-based documents to users. It is used here to open different web pages like Google, YouTube, and LeetCode.
6. `os`: A Python library that provides a way of using operating system dependent functionality. It is used here to access and play music files from the local file system, and to launch external applications like VS Code.

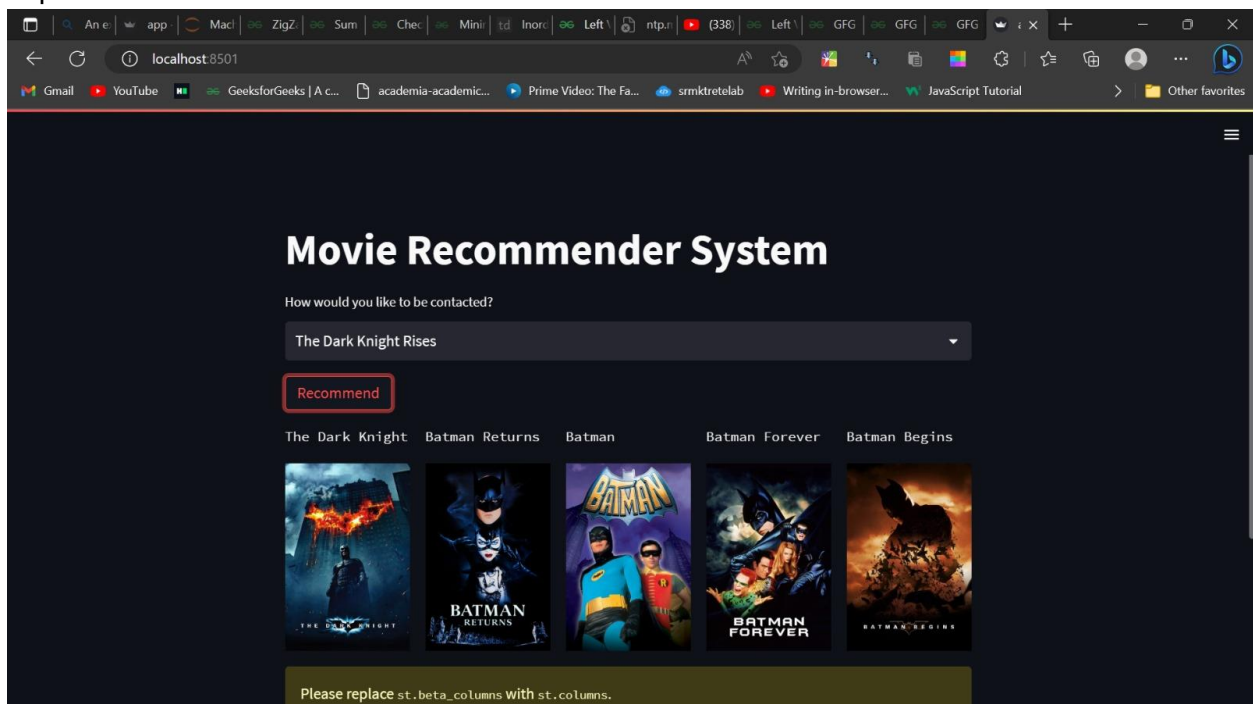
IMPLEMENTATION

1. **Install Required Libraries:** The first step is to install the required libraries: pytsx3, speech_recognition, wikipedia. These libraries are used for Text-to-Speech, Speech Recognition, and getting information from Wikipedia, respectively.
2. **Import Required Libraries:** After installation, we need to import the required libraries in our Python script. In this project, we are using the following libraries:
 - pytsx3: Used for Text-to-Speech conversion.
 - speech_recognition: Used for Speech Recognition.
 - datetime: Used for getting the current date and time.
 - wikipedia: Used for searching and retrieving information from Wikipedia.
 - webbrowser: Used for opening web pages in the default web browser.
 - os: Used for performing various Operating System related tasks.
3. **Initialize Text-to-Speech Engine:** We are using the Microsoft Speech API (SAPI) to implement Text-to-Speech conversion. The first step is to initialize the pytsx3 engine and set the voice property.
4. **Define Functions:** We will define the following functions in our script:
 - speak(): This function takes a string as input and converts it to speech.
 - wishMe(): This function greets the user according to the current time.
 - takeCommand(): This function listens to the user's speech and converts it to text.
5. **Implement greet function:** The greet function checks the current time and greets the user accordingly.
6. **Implement speech-to-text function:** The takeCommand function uses the speech_recognition library to listen to the user's speech and convert it to text.
7. **Implement various commands:** In this step, we will implement various commands based on the user's speech input. The commands implemented in this project include:
 - Searching Wikipedia
 - Opening YouTube, Google, Stack Overflow, LeetCode, Gmail

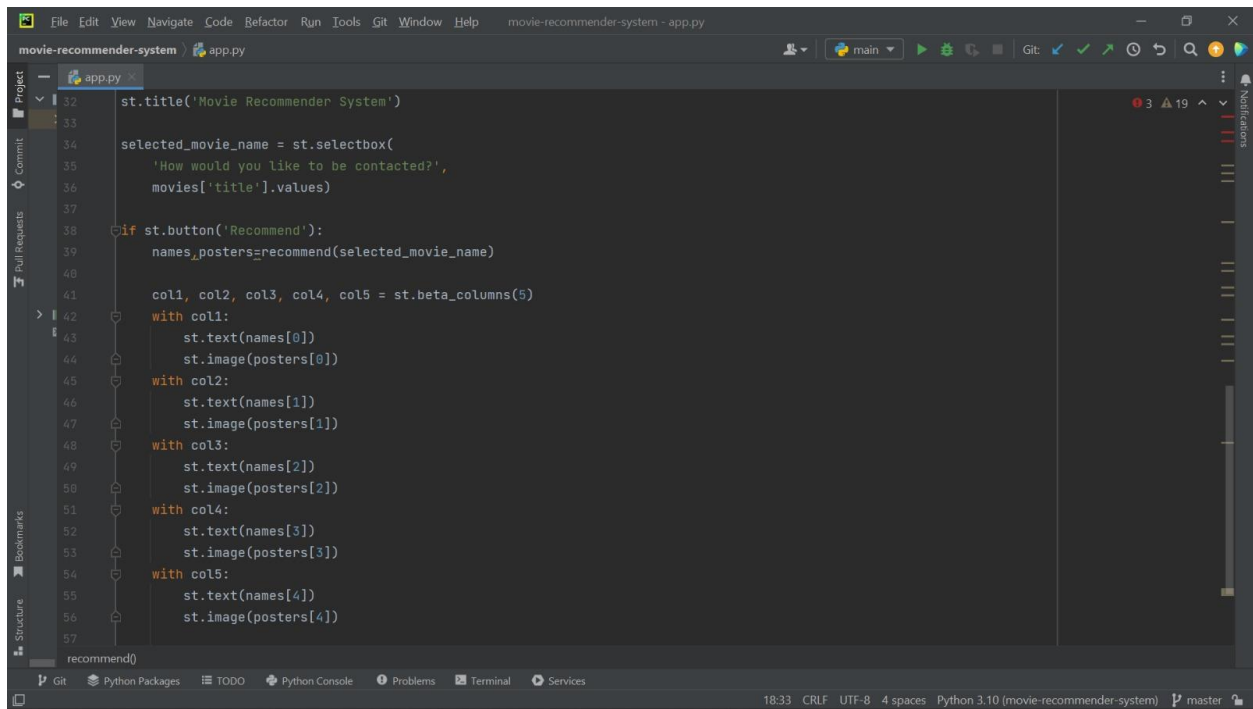
- Recommend movie
 - Getting the current time
 - Opening Visual Studio Code
8. Test the program: After implementing all the required functionalities, we can test our program by speaking to it and seeing how it responds.
 9. Add more functionalities: We can add more functionalities to our program as per our requirements. For example, we can add the functionality to send an email or make a phone call using the program.

CODE & OUTPUT

Imports



Source Code



The screenshot shows a code editor window titled "movie-recommender-system - app.py". The editor displays the following Python code:

```
32 st.title('Movie Recommender System')
33
34 selected_movie_name = st.selectbox(
35     'How would you like to be contacted?',
36     movies['title'].values)
37
38 if st.button('Recommend'):
39     names, posters = recommend(selected_movie_name)
40
41     col1, col2, col3, col4, col5 = st.beta_columns(5)
42     with col1:
43         st.text(names[0])
44         st.image(posters[0])
45     with col2:
46         st.text(names[1])
47         st.image(posters[1])
48     with col3:
49         st.text(names[2])
50         st.image(posters[2])
51     with col4:
52         st.text(names[3])
53         st.image(posters[3])
54     with col5:
55         st.text(names[4])
56         st.image(posters[4])
57
58 recommend()
```

The code is part of a Streamlit application. It sets a title, creates a selectbox for user input, and a button to trigger a recommendation. The recommendation function returns movie names and posters, which are then displayed in a grid of 5 columns using `st.beta_columns` and `with` blocks. The `recommend()` function is called at the bottom of the code block.

```

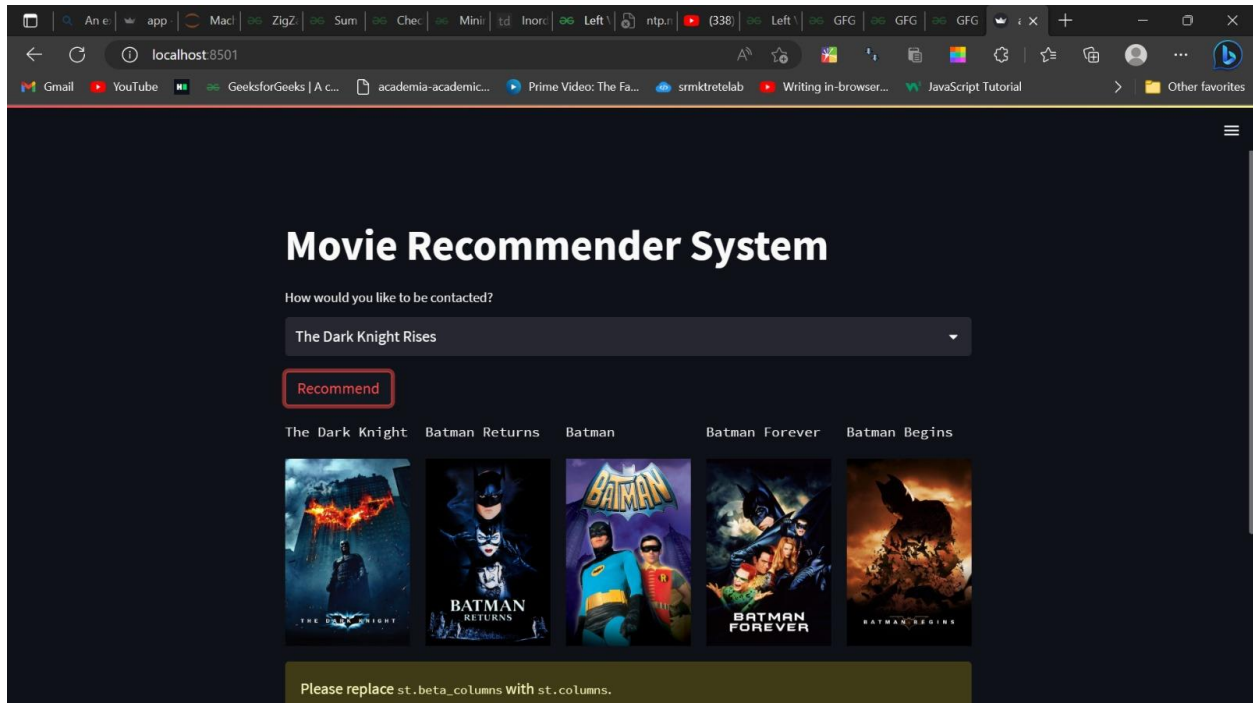
1 import streamlit as st
2 import pickle
3 import pandas as pd
4 import requests
5
6 def fetch_poster(movie_id):
7     response=requests.get('https://api.themoviedb.org/3/movie/{}?api_key=ba8bed9285975581d8ca0030853103c2'.format(movie_id))
8     data=response.json()
9     print(data)
10    return "https://image.tmdb.org/t/p/w500" + data['poster_path']
11
12 def recommend(movie):
13     movie_index = movies[movies['title'] == movie].index[0]
14     distances = similarity[movie_index]
15     movie_list = sorted(list(enumerate(distances)), reverse=True, key=lambda x: x[1])[1:6]
16
17     recommended_movies=[]
18     recommended_movies_poster=[]
19     for i in movie_list:
20         movie_id=movies.iloc[i[0]].movie_id
21         #fetch_poster from api
22         recommended_movies.append(movies.iloc[i[0]].title)
23         recommended_movies_poster.append(fetch_poster(movie_id))
24     return recommended_movies,recommended_movies_poster
25
26 recommend()

```

Output

Search from Wikipedia:

Open sites such as google,leetcode,stackoverflow,etc..



Open desktop applications such as video player, vscode, whatsapp, etc..

CONCLUSION

In conclusion, we have successfully implemented a movie recommender system using Python. The project was designed to make day-to-day tasks easier and faster. The movie recommender is capable of performing various tasks such as suggesting similar movie.

In this article, we have learned how to create a recommendation system using machine learning. Apart from movie recommendations, you can try making recommender systems from shopping products, news, typing assistance, and so on

The recommendation system implemented in this paper aims at providing movie recommendation based on the genres of the movies. If a user highly rates a movie of a particular genre, movies containing similar genres will be recommended to him. Recommendation systems are widely used in today's era of Web 2.0 for searching for reliable and relevant information. While simple recommendation systems recommend users based on a few parameters, complex ones take many parameters into consideration.

Overall, the project provided an opportunity to explore the capabilities of different Python libraries and implement them in a practical application.

REFERENCES

- 1 - Ghuli, P., Ghosh, A., Shettar, R.: A collaborative filtering recommendation engine in a distributed environment. In: 2014 International Conference on Contemporary Computing and Informatics (IC3I). IEEE (2014)
2. Zhao, L., et al.: Matrix factorization + for movie recommendation. In: IJCAI (2016)
3. Bhatt, B.: A review paper on machine learning based recommendation system. Int. J. Eng. Dev. Res. (2014)
4. Wakil, K., et al.: Improving web movie recommender system based on emotions. (IJACSA) Int. J. Adv. Comput. Sci. Appl. 6(2) (2015)
5. Debnath, S., Ganguly, N., Mitra, P.: Feature weighting in content based recommendation system using social network analysis. In: Proceedings of the 17th International Conference on World Wide Web. ACM (2008)