PERSONALISED MOVIE RECOMMENDATION BASED ON MULTI MODEL DATA INTEGRATION

J G I Madushanki

199477V

Degree of Master of Science

Department of Computational Mathematics

University of Moratuwa
Sri Lanka

March 2022

PERSONALISED MOVIE RECOMMENDATION BASED ON MULTI MODEL DATA INTEGRATION

J G I Madushanki

199477V

Thesis/Dissertation submitted in partial fulfillment of the requirements for the degree Master of Science

Department of Computational Mathematics

University of Moratuwa

Sri Lanka

March 2022

DECLARATION

I declare that this is my own work and this thesis/dissertation2 does not incorporate without acknowledgement any material previously submitted for a Degree or Diploma in any other University or institute of higher learning and to the best of my knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text. Also, I hereby grant to University of Moratuwa the non-exclusive right to reproduce and distribute my thesis/dissertation, in whole or in part in print, electronic or other medium. I retain the right to use this content in whole or part in future works (such as articles or books).

Signature: Date:

J. G. I. Madushanki 04th of March 2022

The above candidate has carried out research for the Masters/MPhil/PhD thesis/ dissertation under my supervision.

Signature of the supervisor: Date

04th of March 2022

Abstract

Recommendation systems plays an essential role in the modern era, and it is a part of routine life where it guides the users in a personalised manner towards interesting and useful objects in a large collection of possible options. The aim of the movie recommendation system is to help movie lovers by generating suggestions on what movie to watch. If movie recommender systems are not in place, movie lovers need to spend time on choosing a movie by going through long lists of movies, which is a time consuming task. Therefore, a lot of research has been conducted to generate movie recommendations using different approaches including pure recommendation techniques and hybrid techniques. However, the recommendations generated through these approaches lack personalisation and accuracy.

This thesis presents our approach to generate personalised movie recommendations using multi model data integration to improve the personalisation and accuracy. Different data sources are integrated as inputs when designing this research. A content-based filtering technique collaborated with genetic algorithm-based optimization was utilized for implementation of this research. A precision value of 0.65 was obtained while evaluating the multi-model data integration-based movie recommender system with genetic algorithm-based optimization.

DEDICATION

I dedicate my dissertation work to my family, friends and my supervisor Dr A.T.P. Silva who encouraged me to achieve the final year research project. A special feeling of gratitude to my loving parents whose thoughts and words of encouragement to achieve this. I also dedicate this dissertation to my office mates who have supported me throughout this process. I will always appreciate the kind words and encouragement of them to achieve my goal. I dedicate this work and give many thanks to my supervisor and staff at University of Moratuwa for their help and specially for academic staff for providing guidance throughout this research.

ACKNOWLEDGEMENTS

I would like to thank my supervisor Dr A.T.P. Silva for providing me guidance and insights throughout this project. This project would not have been possible without the support of my supervisor. Also, I would like to extend my thanks to the University of Moratuwa for providing me the opportunity to carry out this research successfully and for the continuous support provided throughout the research. I would like to pay my sincere gratitude to all the academic and non-academic staff members of University of Moratuwa and the batchmates for their generous support, comments and encouragement throughout the project. Last, but not least, my warm and heartfelt thanks go to my family for their tremendous support and hope they had given to me. Without that hope, this research would not have been possible. Thank you all for the strength you gave me.

TABLE OF CONTENTS

DECLARATION ii
ABSTRACT iii
DEDICATION iv
ACKNOWLEDGEMENTS v
TABLE OF CONTENTS vi
LIST OF FIGURES ix
LIST OF TABLES xi
LIST OF ABBREVIATIONS xii
LIST OF APPENDICES xiii
INTRODUCTION 1
1.1 Prolegomena 1
1.2 Aim and Objectives 1
1.3 Background and Motivation 2
1.4 Problem Definition 4
1.5 Novel Approach to Personalized Movie Recommender System 4
1.6 Resource Requirements 5
1.7 Structure of the Thesis 6
1.8 Summary 6
DEVELOPMENTS AND ISSUES OF PERSONALISED MOVIE RECOMMENDATION SYSTEMS 7
2.1 Introduction 7
2.2 Early Developments of Movie Recommender Systems 7
2.3 Breakthrough in Movie Recommender System 8
2.4 Challenges in Movie Recommender Systems 13
2.5 Problem Definition 18
2.6 Summary 18
TECHNOLOGIES USED FOR MOVIE RECOMMENDER SYSTEMS 19
3.1 Introduction 19
3.2 Recommender Systems 19
3.2.1 Collaborative filtering 19
3.2.2 Content based filtering 20
3.2.3 Hybrid filtering 20

3.2.4 Knowledge based recommender systems 21
3.2.5 Ontology based recommender systems 21
3.2.6 Intelligent recommender system 22
3.2.7 Demographics based recommender system 22
3.2.8 Context aware recommender system 22
3.3 Genetic Algorithm 23
3.3.1 Initial Population 23
3.3.2 Fitness Function 23
3.3.3 Genetic Algorithm Operators 23
3.3.4 Selection 24
3.4 Personalised Movie Recommendation Using Content Based Filtering 24
3.5 Summary 24
APPROACH 26
4.1 Introduction 26
4.2 Hypothesis 26
4.3 Input 26
4.4 Output 26
4.5 Process 27
4.6 Users 28
4.7 Features 28
4.8 Summary 28
DESIGN 29
5.1 Introduction 29
5.2 Pre-Processing Module 29
5.3 Review Analysis Module 29
5.4 Profiling Module 30
5.5 Content Based Recommendation Module 30
5.6 Evolutionary Computing Module 30
5.7 Summary 31
IMPLEMENTATION 32
6.1 Introduction 32
6.2. Implementation 32
6.2.1 preprocessing module 32
6.2.2 Review analysis module 38

6.2.3 The Profiling module 40
6.2.4 Content based recommendation module 43
6.2.5 Evolutionary computing module 44
6.3 Summary 47
EVALUATION 48
7.1 Introduction 48
7.2 Results 48
7.3 Summary 62
CONCLUSION AND FUTURE WORK 63
8.1 Introduction 63
8.9 Summary 63
REFERENCES 64
APPENDIX 68
Appendix A 68

LIST OF FIGURES

Figure 5.1. 1 Top level architecture of personalised movie recommender system 29

Figure 5.6. 1 The main steps involve in genetic algorithm 31

Figure 6.2.1. 1 Mappings in Movie Lens dataset for TMDB Id and IMDB Id 33
Figure 6.2.1. 2 Mappings in Movie Lens dataset for YouTube Id and Movie Id 34
Figure 6.2.1. 3 Preprocessing user data 36
Figure 6.2.1. 4 Java method to get YouTube video details 36
Figure 6.2.1. 5 Python script to get IMDB review data 37
Figure 6.2.1. 6 Python script to create TMDB and IMDB mapping dataset 38

Figure 6.2.2. 1 Naive Bayes classifier to get review sentiment 39
Figure 6.2.2. 2 Integration of review sentiment with movie profiles 40

Figure 6.2.3. 1 Integration of review sentiment with movie profiles 40
Figure 6.2.3. 2 Sample movie profile 41
Figure 6.2.3. 3 Sample review data obtained from IMDB API 41
Figure 6.2.3. 4 Creation of user profiles 42
Figure 6.2.3. 5 Creation of movie profiles 43

Figure 6.2.4. 1 Multi model data integration-based content-based filtering implementation 44

Figure 6.2.5. 1 Architecture of evolutionary computing-based approach 45
Figure 6.2.5. 2 Creation of individuals and chromosomes 46
Figure 6.2.5. 3 Single point crossover implementation 47
Figure 6.2.5. 4 Swap mutation implementation 47

Figure 7.2. 1 Results of recommendation approaches 49
Figure 7.2. 2 Precision obtained for recommendation approaches 50
Figure 7.2. 3 Calculated precision for experiments 50
Figure 7.2. 4 Calculated Degree of novelty per movie 51
Figure 7.2. 5 Calculated degree of novelty for experiments 51
Figure 7.2. 6 Google recommendations for movie Garfield: A Tail of Two Kitties 52
Figure 7.2. 7 Recommended movies to movie Garfield: A Tail of Two Kitties 52
Figure 7.2. 8 Google recommendations for movie Thor 53
Figure 7.2. 9 Recommended movies to movie Thor 53

Figure 7.2. 10 Google recommendations for movie Interstellar 54
Figure 7.2. 11 Recommended movies to movie Interstellar 54
Figure 7.2. 12 Google recommendations for movie Inception..... 55
Figure 7.2. 13 Recommended movies to movie Inception 55
Figure 7.2. 14 Google recommendations for movie The Amazing Spider-Man 2 56
Figure 7.2. 15 Recommended movies to movie The Amazing Spider-Man 2..... 56
Figure 7.2. 16 Google recommendations for movie Toy Story..... 57
Figure 7.2. 17 Recommended movies to Toy story 57
Figure 7.2. 18 Google recommendations for movie Apollo 13..... 58
Figure 7.2. 19 Recommended movies to movie Apollo 13..... 58

LIST OF TABLES

Table 2. 1 Summary of benefits and challenges identified in literature review..... 14

LIST OF ABBREVIATIONS

Abbreviation	Description
CARS	Context aware recommender systems
CSV	Comma Separated Values
DBRS	Demographics based recommender systems
GA	Genetic Algorithm
IRS	Intelligent recommender systems
KBRS	Knowledge based recommender systems
KNN	K Nearest Neighbors
MAE	Mean Absolute Error
OBRS	Ontology based recommender systems
RMSE	Root Mean Square Error
SVD++	Singular value decomposition plus-plus

LIST OF APPENDICES

Appendix - A Questionnaire to identify user interests and behaviors towards movies 69

CHAPTER 1

INTRODUCTION

1.1 Prolegomena

The movie recommendation generation is an exhaustive task which consumes various tastes of users and various genres of movies. It is a key task in social life of humans due to its ability in providing enhanced entertainment.[1] In order to provide reliable recommendations, the recommender systems need to exactly capture the user needs and preferences into the user profile. However, for a subjective and complex domain like movies, user behaviour plays a critical role in the decision-making process. Even though there are many approaches developed in the past, personalisation of movie recommendations has still been a research challenge in terms of quality of the recommendation. [2], [3] To recommend movies, a complete aggregation of user's preferences, emotions and reviews can be used to assist the users to find the best movies in a more convenient way. [4] Despite numerous interests shown in research related to movie recommender systems, there has been limited research related to multi model data integration on generating personalised movie recommendations. Main focus is on improving the quality and the personalisation of the movie recommendations by using content-based recommendation techniques and the genetic algorithm approach integrated with the multi-model data integration as the solution to this problem.

Rest of the chapters are organized with sections, namely, aims and objectives, background and motivation, problem definition, novel approach to personalised movie recommender system, the structure of the thesis with a summary for the chapter.

1.2 Aim and Objectives

The key aim of this research project is to design and develop a personalised movie recommender system based on multi-model data integration.

In order to reach the aim, the following objectives are defined.

- The critical review of literature related to multi-modal data integration and personalised recommendations.
- An in-depth study of the technologies which are used in personalised recommender systems.
- Design and develop the personalised movie recommendation system based on the proposed technologies.
- Evaluate the proposed system for its effectiveness.
- Documenting the research in the thesis.

1.3 Background and Motivation

Recommender systems have emerged as an important recommendation technique on the web and are widely used to generate recommendations on different items. These models have become extremely popular, and they are used in movies, food, books, television, restaurants, etc. Many companies get benefits from recommender systems in enhancing the user experience and satisfaction by providing quick and coherent suggestions. [5] Search engines solve the problem to some extent, but it does not solve the personalisation problem. [6] As a huge volume of data is available online, the need for personalisation and filtering systems are growing day by day. Recommender systems have become an answer to the need of personalisation and improving future suggestions. In order to provide reliable recommendations, the recommender system requires to capture the user's needs and preferences exactly. However, traditional recommender system models are based on user profiles and influence of user emotions are not taken into consideration.

In order to help people to choose the most suitable products from a massive collection of available options recommender systems were launched around mid-1990s. The nature of human beings, relying on opinions of associates before starting something new led to the idea of implementing recommender systems. [7] For instance, when buying a new laptop, mobile phone, before going to a movie, trying a new meal, visiting a doctor we tend to get the opinions from peers. Thus predicting the “preference” of an item is the main purpose of the recommender systems and it could

be classified to information filtering systems. [6] Until today recommender systems have been evolved and developed around various fields using different recommendation techniques.

Recommender systems have evolved to be an essential element in websites such as YouTube, The Amazon, Google, Netflix. [8] Among the areas, recommender systems have been used, it is commonly used as playlist generators in entertainment-based services such as Amazon Prime Video, YouTube, Netflix, Spotify, Zee5, and product recommender platforms such as Amazon and also content recommenders for social media platforms such as Twitter and Facebook. [9] Netflix is a hybrid movie recommender system, and it uses content-based filtering and collaborative filtering to make recommendations. The similarities of similar users are compared in collaborative filtering while content-based filtering offers movies that share characteristics with movies that user has highly rated. [10]

Movies falling into different genres like horror, action, animated, adventure etc. are launched every year. Due to abundant stocks of movie content, people are facing difficulties in finding their favorite movies. [11] Movie recommender systems help users to discover the movies according to their preference based on the other user's movie experience. Movie recommender systems are efficient and effective as time is not wasted in useless browsing. [10] Simple recommendation systems consumes few attributes, while the more complex recommender systems make use of more attributes to generate the results and increase the user friendliness. [6]

The modern community drives the community point of view by sharing their opinions on social media platforms. They spread information fast and make others want to experience the things they are interested in. [12] Sentiment analysis and opinion mining has become an interesting area that finds the opinions of users all over the world through analysis of reviews from online social networks like Facebook, Yahoo, Twitter, etc. [13]

Therefore, movie data captured through different data sources along with sentiment of the movie reviews are integrated to generate personalised movie recommendations as accuracy and the quality of the movie recommendations can be improved.

1.4 Problem Definition

Many people would like to watch movies as a source of entertainment. But finding the appropriate movies has been a challenging task as a lot of movies are released every year. In order to cater the requirement of recommending movies according to the taste of the user, movie recommender systems were developed. Even Though, many approaches are taken to generate movie recommendations, the quality of the recommendations are low as multi model data integration is not taken into consideration when generating personalised movie recommendations. Therefore, personalised movie recommendation has been a research challenge.

1.5 Novel Approach to Personalized Movie Recommender System

This research focused on generating personalised movie recommendations using multi model data integration. It is a five-step process, including pre-processing of data, review analysis, profiling, content-based filtering and optimizing the results with the use of evolutionary computing.

In the first step data extracted through Movie Lens dataset, TMDB API, IMDB API, YouTube API and user interest data captured through a questionnaire are subjected to data pre-processing and obtain the pre-processed data. Pre-processed data are passed as inputs to the review analysis module and profiling module.

Pre-processed text data are passed as the input for the Review analysis module. Under the review analysis module sentiment analysis will be performed on movie reviews obtained from IMDB API. After performing sentiment analysis sentiment value will be calculated as the output. Generated sentiment value will be integrated with the movie profiles to use in the recommendation process.

The output of the pre-processing module and review analysis modules are passed as the inputs for the profiling module. Under the profiling module two types of profiles will be created by integrating the different types of data from different sources, namely, user profiles and movie profiles.

Movie profiles are passed as the input to the content-based filtering module and a similarity score will be calculated after applying recommendation technique. Predefined N number of top recommendations will be generated based on the calculated similarity score.

After getting the top N recommendations, the recommended movie list is passed to the evolutionary computing module to optimize the recommended movie list and N recommendations act as the parent population. A single recommended movie is used as a gene and chromosomes which comprises movies are generated. Genetic operations, namely crossover and mutation are performed, and fitness function is evaluated to identify the highest fitted individuals. A score will be calculated as the output of the fitness function by evaluating the genres of the recommended movies with user given movies. Offspring are generated for a predefined number of generations and best-fitted individuals are generated as the output of the genetic algorithm module.

Using this novel approach to generate personalised movie recommendations, aim is to improve the quality and the accuracy of the movie recommendations.

1.6 Resource Requirements

Data set

- Movie data collected from Movie Lens dataset.
- Movie data collected from the TMDB API.
- Movie reviews data collected from the IMDB API.
- Video statistic data collected from the YouTube API.
- User data and user interest data collected through a questionnaire.

Hardware requirements

- A computer with 8GB RAM and Windows Operating System.

Software requirements

- PyCharm IDE
- IntelliJ IDE
- Anaconda Navigator
- Python libraries

1.7 Structure of the Thesis

Thesis is structured with 8 chapters, including the introduction and chapter 2 presents the literature review, chapter 3 describes the technology adopted, chapter 4 presents the approach including hypothesis, input, the process, output users and features, chapter 5 described the design of the solution, chapter 6 describe the implementation of the solution, chapter 7 presents the evaluation, and finally chapter 8 presents the conclusion and future work.

1.8 Summary

This chapter presented an introduction to personalised movie recommender systems and identified achievements in the field and research challenges. Here the research has been defined as improving the quality of personalised movie recommendations with multi model data integration. In the next chapter, a detail discussion on technology adopted for implementing personalised movie recommender system is presented.

CHAPTER 2

DEVELOPMENTS AND ISSUES OF PERSONALISED MOVIE RECOMMENDATION SYSTEMS

2.1 Introduction

Chapter 1 presented a small introduction to personalised movie recommender systems and discussed aims and objectives, background and motivation, problem definition, solution, resource requirements, the structure of the thesis with a small summary of the chapter. We have come up with multi model data integration based content-based recommendation approach as the solution to address the above problem. The evidence of the applications of content based filtering based recommender system approach for similar projects provides the inspiration the new approach. Rest of the chapter has been structured with the subheadings of early developments of recommender systems, recommender systems and the latest development, challenges in recommender systems, the problem definition with a summary for the chapter.

2.2 Early Developments of Movie Recommender Systems

Around mid-1970s recommender systems developed as an independent research area in Duke University. Since then, a lot of research work has been conducted in this area until today with the introduction of various techniques. Tapestry is the first recommender system which came into existence that was developed at the Xerox Palo Alto Research Center. The increasing number of receiving emails which are mostly irrelevant ones that which is troublesome and difficult to manage was the motive that led to its development. In order to get rid of this problem users had to setup their mailing list to allow only the contacts in the contact list could send emails to them and other emails were sent to the spam folder. This is the same approach currently practised in email accounts. For this implementation, the collaborative filtering technique was used. [7]

The Group Lens research team from the University of Minnesota initiated research on movie recommender systems. The movie recommender system known as Movie Lens was their research object. [1] The early research was mainly focused on the content of the recommender system. However, this recommendation method was limited to study on the available content, which makes researchers invest in designing a new recommender system.

2.3 Breakthrough in Movie Recommender System

Many recommendation systems have been made over the past years. Researchers have implemented these systems using various machine learning algorithms and big data analytics techniques.

Agarwal et al [14] presented a movie recommendation system using the Apache Mahout library and collaborative filtering approach. In their work they have used the Yahoo Research Web scope database. For user-based filtering, the user similarity is calculated as an addition to the Pearson Correlation Similarity which uses the Pearson Correlation Coefficient to decide the similarity between user ratings. To calculate the user neighbourhood, a distance based clustering algorithm called Nearest N User Neighbourhood is used. Item similarity is computed using Log Likelihood Similarity algorithm.

Kashyap et al. [10] came up with a K-means algorithm-based movie recommender system which allows the user to make a choice from a given list of attributes and recommend a list of movies based on cumulative weight of different attributes. They used the Movie Lens small dataset for this purpose.

Wang, Wang and Xu [1] presented a big data analytics based framework integrated with hybrid recommendation technique and sentiment analysis. They have used Spark to enhance the timeliness of the system and sentiment analysis is introduced to improve the accuracy of the recommender system.

Nagamanjula et al [13] presented a movie recommendation system based on opinion mining and user similarity analysis which provides top-k movie recommendations by using Naive Bayes based Support Vector Machine algorithm which comprises two classifier algorithms, Support vector machines and Naive Bayes classifier which can give better performance. In this work reviews are collected from the users for movies and data is pre-processed with major preprocessing steps. Then preprocessed data is subjected to explicit and implicit aspect extraction, and finally, top-k movies are recommended to target users. Similarities among the users are computed using weight adjusted cosine score metric. Movie Lens data set is used for the testing of the system.

Ho et al [15] proposed E-MRS, which has the capability to incorporate user emotions into the user profile to provide well-recommended products based on their emotional state using hybrid filtering approach. An Emotion Detection Algorithm is used to determine the emotions based on the colour choice of the user. During the registration, users are prompted to answer a questionnaire to express their opinion about which categories of films or which films correspond to each emotion. Each film selected by the user is considered as being rated implicitly. User profile vectors are created using information about the movie ratings and the emotion related to each movie.

Babanne et al. [16] proposed a real time, content-based approach to generate personalised video recommendations which works as a personal assistant to the user. It extracts facial features through video inputs from users. Face detection is carried out using Haar cascade classifiers, LBPH and OpenCV. Haar cascade is a classifier that detects faces and body parts in an image. An age detection algorithm and gender detection algorithm is used to detect the age and the gender of the person. Besides, Convolutional Neural Network is used for gender detection. FER 2013 dataset is used for emotion detection and for age and gender detection Adience dataset is used.

Furtado and Singh [6] proposed a content-based filtering movie recommendation system based on Cosine similarity and KNN which will improve the accuracy of the recommendations. The Cosine similarity between two objects measured by the cosine angle between the two objects. Here the angle between the two movies will decide the

similarity between the two movies. If the angle between two vectors is small, they are almost similar to each other, and if the angle between the two vectors is large then the vectors are very different from each other. After calculating the cosine similarity, a normalized popular score will be passed through a distance computing function. Then by using the KNN algorithm, the nearest neighbors will be identified, and recommendations are generated for the user.

Vibhandik et al. [11] presented a hybrid filtering based recommender system which uses KNN, collaborative filtering and Cosine similarity to minimize the error of finding the good recommendations. They have used the TMDB API to get movie data and user data for the recommendation system.

Surendran et al. [17] proposed a hybrid filtering based movie recommender system which recommends movies to a user by evaluating IMDB ratings. When a new user sign up, his/her details such as which movies they have seen and its corresponding ratings are saved with other users in the dataset. After finding out similar users from the dataset using collaborative filtering and content-based filtering, recommendations are generated for a particular user.

Kanoje et al. [18] presented a profile based recommender system approach to recommend universities to a user. This approach extracts information and creates user and university profiles. User profile extraction is done when the user registers into the system. To allow implicit profiling, users' details are extracted from different social networks and profiles of the user. To gain the knowledge from the data, the dataset is processed using the Weka tool. The university database provided by UCI Machine Learning Repository was used to evaluate the proposed approach.

Damanhuri et al. [12] presented an approach to profile users and model reputation computation based on sentiment analysis. Names of the popular characters are taken from the DBpedia and passed to the Twitter API through the Tweepy library and it will collect all the tweets of the given people. Then the retrieved tweets are passed to sentiment analysis process, and it analyses people's views and discussion on someone

on Twitter and generate a score. It rates each person based on tweet sentiments and orders the list of these people from the most liked or the most disliked. In the implementation Textblob python module which enables natural language processing and help with tokenization and sentiment analysis is used.

Choudhury et al. [19] proposed a trustworthy recommendation model which recommends the movie to the trusted user based on the implicit trust value and a different machine learning approach. Trust-based filtering approach is used to improve the accuracy of the recommendations. The accuracy of the proposed approach is evaluated against back propagation neural network, Singular value decomposition and deep neural network.

Ju and Xu [20] presented a novel approach based on K-means clustering algorithm and collaborative filtering with artificial bee colony algorithm. Artificial Bee Colony algorithm was used to overcome the local optimum problem associated with the K-means clustering algorithm. The modified cosine similarity approach is used to calculate the similarity between users within the same clusters and finally recommendations are generated against the target users.[21]

Ji *et al.* [22] proposed a hybrid recommendation system called BRScS that can combine multi-source heterogeneous data such as reviews, scores, and information on social media network. With the introduction of the user trust model to combine social media data together with scores and reviews can solve the cold start problem and data-sparsity problem productively because friends' data can be used to make top N recommendations. The model is scalable and can integrate more heterogeneous data types easily. However, providing explanations to generated recommendations are difficult as the proposed approach is based on neural network.

Anwar and Uma [23] presented a movie recommendation system using collaborative filtering and Single Value Decomposition plus-plus algorithm which can overcome the cold start and data sparsity problems up to a certain extent. The proposed approach is compared with the well-known machine learning approaches, namely KNN, SVD and

Co-clustering. The recommender system error is evaluated considering MAE and RMSE.

Wroblewska *et al.* [24] developed a machine learning based custom platform and algorithm which can be applied to almost any domain which consumes multimodal and multi-view data. The developed recommender system utilizes a multimodal fusion of multiple interaction types such as purchases, clicks, adding a product to cart and multiple attribute modalities such as text, images, audio, video, and other behavioural data through time. Proposed algorithm was tested against the DeepFashion and Street2Shop datasets and achieved good results in various e-commerce stores.

Stitini et al. [25] presented a genetic algorithm and content-based filtering based novel recommendation system which focus to find new items that the user will like more. The proposed approach solves the over-specialization problem associated with content-based filtering by integrating with genetic algorithm. The genetic algorithm approach is used to identify the best recommendation list for a single individual based on their preferences. In the proposed approach a set of movie genres are consumed as inputs and a set of recommended movies are given as the output. A genre score will be calculated and finally, scores for the entire population will be calculated and the most fitted individuals are chosen.

Valero et al. [26] presented a soft computing approach based on genetic algorithms which can be used to set the optimum combination of the product attributes. The purpose of this work is to learn the user's preferences about products or services defined by a set of attributes, in order to determine which attributes are more or less valuable, essential or dispensable for users. Besides this work presents that obtained weights could be employed to predict user ratings of movies with a highly good rate of correct predictions. Genetic algorithm is used to capture the preferences of the users of the system that are similar to the current user and apply the learned weights to provide suitable recommendations to the current user.

Due to accuracy obtained through early recommender systems, researchers started to

turn more into user centric content descriptors in recent years. [27] Personalised movie recommendation systems are developed as an assistant to a user for entertainment purposes. Many approaches were developed in the past but these recommender systems are subjected to the limitation of lack of accuracy. Traditional models capture the user emotions through user profiles using simple techniques like questionnaires, reviews and ratings etc. But accuracy of the recommendations are not guaranteed because the user may not tell the truth or have difficulties in expressing how he feels. As a solution, multi model data sources such as text, video and image data from different sources like comments, posts, reviews from social media platforms, ratings and video recordings of viewers when watching a movie can be integrated to capture the constantly changing preferences of the user.

2.4 Challenges in Movie Recommender Systems

Over the past, many recommendation systems were developed using either content based, collaborative or hybrid filtering methods. These systems utilized different big data techniques and machine learning algorithms for the implementation.[5] Even though a lot of work has been conducted on recommender systems, still there are some limitations that need more attention in research. The most common restrictions are the cold start problem, the data sparsity problem, scalability, privacy protection and the gray sheep problem. [7], [10]

The cold start problem occur when it is difficult to make any correct recommendations to the user as either the user is new to the system or a given item is added to the system for the first time. [9] There are two types of cold start problems namely, new user cold start problem and new item cold start problem. [7] The new user cold start problem occurs because there is only less information about the user as the user is new to the system. As a result, generating recommendations becomes difficult. Similarly, when a fresh item or a product is added to the system new item cold start problems which makes generating the recommendations difficult.

The sparsity problem is also a major problem experienced by recommender systems. The sparsity problem occur with the lack of user ratings and the available ratings are

also disperse or sparse. As the recommendations are based on user ratings, it is a major issue in collaborative filtering technique.

Recommender systems work well with small size datasets, but when dealing with real world datasets which grows fast, is quite an unmanageable task. Therefore, scalability issues arise as the system can't perform well when the capacity of data increases.

As recommender systems consume a lot of user data to provide better recommendations it has become a privacy concern. In addition to that user data can be easily accessible by a malicious user.

Gray sheep problem occurs as a result of inconsistent behaviour of the user. These users don’t have well-defined tastes and they can wish one thing at one moment and a completely different item in the other. As a result, irrelevant recommendations are generated from the recommender system resulting in a decrease of efficiency.

A summary of the previously conducted research covered in the literature review is summarized in the below table 2.1 with contributions and limitations.

Table 2. 1 Summary of benefits and challenges identified in literature review

Research	**Contribution**	**Limitations**
[1]	Introduced Spark to improve the timeliness. Introduced sentiment analysis to improve the accuracy of the recommender system.	Not verified with more data sets. The sentiment analysis model can be tuned to accommodate more situations.

[2]	Experimented the improvisation of the recommender system using fuzzy logic.	Movie details and movie ratings were only used for generating recommendations.
[4]	User similarity analysis and opinion mining performed to provide top-k movie recommendations using Naive Bayes based Support Vector Machine algorithm.	Opinion mining performed only through text movie reviews.
[5]	Consider user ratings when recommend movies.	Attributes like actors, director and the genre of the movie are not considered for generating recommendations.
[7]	Proposed a content-based filtering movie recommendation system using Cosine similarity and KNN which improves the accuracy of the recommendations.	Hybrid approach enhance the accuracy and the quality of the recommendations. More parameters can be considered to improve the quality of the recommendations.
[10]	Uses K means algorithm-based approach which allows a user to select his choices from given list of	Research was conducted using a small dataset.

	attributes and then recommend movies based on the cumulative weight of different attributes.	Results are not compared with a different machine learning algorithm. Formal evaluation was not conducted.
[11]	Presented a hybrid filtering based recommender system which uses KNN, collaborative filtering and Cosine similarity to minimize the error of finding the good recommendations.	Including more features and algorithms to improve the accuracy.
[12]	User profiling and reputation computation based on crawled tweets.	System only supports crawled tweets that are saved to the local directory and doesn't support tweets posted real time.
[15]	Incorporated user emotions into the user profile to provide well-recommended products based on their emotional state using hybrid filtering approach. An Emotion Detection Algorithm is used to determine the emotions	User emotions were captured explicitly through a questionnaire and implicit feedback was not considered

	based on the colour choice of the user.	
[16]	Proposed a real time, content-based approach to generate personalised video recommendations which works as a personal assistant to the user which can extract facial features through video inputs from users.	Hybrid approach can enhance the accuracy and the quality of the recommendations.
[17]	Explicit feedback from the user taken during the registration to generate recommendations	Implicit feedback was not considered when generating movie recommendations
[18]	Presented implicit and explicit profile-based recommender system approach	Profile information from social networking websites are not considered
[24]	The proposed approach consumes multimodal and multi-view data.	Interpretability not considered.
[25]	Experimented a genetic algorithm-based approach to overcome the overspecialization problem.	Tested with small and medium sized dataset. Limited content issue.
[26]	Can use to give good recommendations even when there is no enough	Initial population was generated randomly.

	information of the current user.	
[33]	Evaluated against models such as XGBoost, Surprise Baselineonly, Surprise KNNBaseline, Matrix Factorization SVD to calculate the RMSE and MAPE to find the best model.	Hyper parameters tuning not performed to improve the RMSE.

2.5 Problem Definition

Generating movie recommendations is a comprehensive task which needs to deal with the emotional state of users. However, the quality and the accuracy of the generated movie recommendations are not guaranteed as the implicit user behaviours are not taken into consideration. Therefore, generating personalised movie recommendations has been a research challenge.

2.6 Summary

This chapter presented a critical review of literature in movie recommender systems and identified achievements in the field and research challenges. Here the research has been defined as inadequate research has been done to achieve quality and the accuracy of personalised movie recommendations. In the next chapter we give a detailed discussion on technology adopted for implementing personalised movie recommender system.

CHAPTER 3

TECHNOLOGIES USED FOR MOVIE RECOMMENDER SYSTEMS

3.1 Introduction

Chapter 2 presented a comprehensive critical review of literature on the application of the recommender system approach to generate movie recommendations and defined our research problem as inadequate research has been done to achieve quality and the accuracy of personalised movie recommendations. To solve this problem, we have recognized the need for research into improving the quality and accuracy of personalised movie recommendations. This chapter describes the theory behind recommender systems and their further developments.

3.2 Recommender Systems

Recommender systems are broadly classified into several categories based on the information they consumed to recommend items. There are several recommendation techniques mentioned in literature such as content-based filtering, collaborative filtering, and hybrid filtering, knowledge-based recommender systems and some other recommendation approaches like community-based systems and demographic-based systems. However, three recommendation approaches are mostly followed in the literature, namely collaborative filtering, content-based filtering and hybrid filtering. [28]

3.2.1 Collaborative filtering

In collaborative filtering, approach recommendations are given to the user based on the other users who had similar preferences in the past. The collaborative filtering based systems require users to express their views on products or items. After collecting the views of the users, items will be recommended based on the user's opinion similarity. Users who share the same opinion acts as the contributors.

Traditional collaborative recommendation techniques include item-based, user-based and model-based methods. [28] The collaborative filtering approach provides improved results when a plentiful number of users and the number of items/products available and knowledge engineering is not required. Although a collaborative filtering algorithm is simple and efficient, it suffers from numerous problems such as the cold start problem with fresh users and fresh items/products, prediction accuracy and a shortage of capturing complex interactions between the user and the item.[8] In collaborative filtering clusters of similar users are used to make recommendations. [6] In addition to that in the Collaborative method, the large amount of information has been collected and analysed on the user's behaviours, preference. [11]

3.2.2 Content based filtering

Content based recommender systems work with the user provided explicit or implicit data. [27] Based on that data, user profiles are created which are used to make recommendations to that user. Content based filtering approach does not rely on the other user's preferences and differentiation between items are possible. It is most suitable for situations where there is data on a product/item, but not on the user. [17] As users perform thousands of transactions according to their tastes, it makes it difficult to make an accurate recommendation. The content based filtering technique has the ability to explain how recommendations are produced to users. [29] In content-based filtering also cold start problems can be noticed for new users as recommendations cannot be generated for fresh users, as there are no ratings on which to base a forecast. In addition to that gray sheep problem and sparsity problem can also be noticed. It is easy to explain the work of a content-based recommender system by cataloguing the content features of an item/product. [30] But in order to generate recommendations they need a rich description of items and well-organized user profiles.

3.2.3 Hybrid filtering

Hybrid filtering approach is a mixture of content based filtering and collaborative filtering. [5] It is introduced to avoid major issues associated with recommender

systems like the sparsity problem and the cold start problem. Weaknesses of an individual technique can be avoided in a combined model.[28] In these types of recommender systems, collaborative filtering and content-based filtering are performed separately and finally the results of both techniques are combined to provide recommendations. Therefore, it is the most common and popular technique today. [6] The data sparsity problem and the cold start problem can be addressed by hybrid approach.[29]

3.2.4 Knowledge based recommender systems

Knowledge based recommender systems operate based on the domain knowledge on users and items. In this approach three main types of information are required, namely, knowledge about the items, users, and the similarity between the users and items. KBRS is based on explicit knowledge on user desires and recommendation criteria, such as which item should be recommended in which context. [25] KBRS approach support to form hybrid recommendations by integrating with other recommendation approaches. This recommendation approach is able to get rid of the cold start problem. Knowledge and engineering skills are required for KMRS which is the main limitation of KBRS.

3.2.5 Ontology based recommender systems

Ontology based recommender systems require domain knowledge and ideal knowledge about the users and items. In OBRS an ontology is mainly utilized for knowledge representation. OBRS and KBRS are related to each other. OBRS also suffers from limitations of traditional recommendation systems such as the data sparsity problem, Ramp Up and overspecialization problem. OBRS are more useful and appropriate for e-learning, e-commerce, and tourism related fields. On the other hand, development of ontologies is costly and complex task. [23]

3.2.6 Intelligent recommender system

Intelligent recommender systems are an extension to the KBRS. IRS exploit knowledge, learn and explore new knowledge from reviews and recognize preferences. IRS is composed of learning methods, knowledge representation models and reasoning mechanisms. In addition to that, the IRS has five Knowledge models such as users, items, context, criticisms and domain that can be considered during the recommendation. It can get rid of the cold-start problem as it does not depend on user ratings and due to that knowledge engineering processes is not required. IRS improves performance by utilizing the information about users through learning mechanisms. [23]

3.2.7 Demographics based recommender system

Demographics based recommender systems utilizes user demographic data and classify the user on the basis of the user's demographic data stored in user profiles for suggesting the item. The demographic approach doesn't require previous user ratings like in CRS and CBRS. The DBRS is composed of three stages namely, input data collection, computation of similarity and recommendation generation. The input data collection is the first step which requires a new target user's demographic information and also demographic information and the rating of other users. The Similarity Computation step uses user demographic information to retrieve users having related demographic interest by creating a neighbourhood. Finally, the recommendation generation stage generates the neighbouring user positively rated items as recommendations. [23]

3.2.8 Context aware recommender system

Context aware recommender system generates by using particular contextual conditions of the user. CARS collect the circumstance of the user at the time of making recommendations. There are three main types of architecture for building CARS, namely, Contextual Modeling, Contextual Pre-filtering, and Contextual Post-filtering. [23]

3.3 Genetic Algorithm

Genetic Algorithms (GAs) are adaptive heuristic search techniques that are used to solve optimization search problems, based on genetic processes of biological organisms which simulate adaptive processes of natural systems that retain features of natural systems. Holland and scholars at the University of Michigan developed Gas around the 1970s. Their objective was to mimic adaptive processes of natural systems that retain features of natural systems. GA offers a valid approach to problems requiring efficient and effective search techniques. It is inspired with biological ideas such as inheritance, mutation, selection and crossover. It follows the idea of survival of the fittest. GAs has gained popularity and performance over other search techniques through years as it searches only a limited number of solutions and it's easy to implement and use. GA based approaches are widely used to solve optimization problems in many applications such as traveling salesman problem, airport traffic control, information retrieval (IR), reactive power optimization, job shop scheduling, and hydraulics systems such as water pipeline systems. [25],[31] [32]

3.3.1 Initial Population

The first offspring generation represented by chromosomes which is used to mate and generate child offspring.

3.3.2 Fitness Function

The fitness function is a mathematical function which calculates a score against a given individual which decide whether a solution is successful or not.

3.3.3 Genetic Algorithm Operators

Candidates for the next generation are designed using GA operators, namely mutation and crossover.

- Crossover

Crossover operator is based on confidence intervals, and it is associated with the capacity of interpolations, which is related to the fact of an individual belonging to a

confidence interval built out of the best individuals in the population. There are many methods to perform crossover operation including uniform crossover operator, single-point crossover operator, and multi-point crossover operator.

- Mutation

Mutation is an exploration based operator which inspects for new solutions and prevents the system from converging on a local maximum, which can lose genetic diversity. The mutation operator alters gene values with a new value in an unplanned way within the allowed gene domain.

3.3.4 Selection

Selection process decides which individuals from the old population and offspring populations will be chosen to form the succeeding offspring generation. The main pillar of genetic algorithm is the selection operator. There are different selection approaches used for the selection process such as random selection, elitist selection, and tournament selection.

3.4 Personalised Movie Recommendation Using Content Based Filtering

Content based filtering technique is utilized in the movie recommendation process to improve the quality and the accuracy of the recommendations. In the content-based filtering approach a score is calculated combining the scores obtained from pure recommendation techniques. The initial recommendation list is based on a hybrid recommendation method containing movies ranked according to their scores.[9] In addition to that, personalisation can be improved by introducing the multi-model data integration with genetic algorithm approach.

3.5 Summary

This chapter presented the technologies adopted for personalised movie recommender systems including a small description of recommender systems, content-based filtering, collaborative filtering and hybrid filtering approach. In the next chapter, we

give a detailed discussion on the approach for implementing personalised movie recommender system, including the hypothesis, input, output, the process, users, features with a summary of the chapter.

CHAPTER 4

APPROACH

4.1 Introduction

Chapter 3 presented the technologies that can be used to solve our research problem of inadequate research to achieve quality and the accuracy of personalised movie recommendations. We have come up with multi model data integration-based content-based filtering recommendation approach as the solution to address the above problem. The new approach is inspired with numerous evidences of the application of content based recommender system approach for similar projects. Rest of the chapter has been structured with the subheadings, hypothesis, input, output, process, users and features.

4.2 Hypothesis

The hypothesis of research is that movie recommendations can be done by multi model data integration.

4.3 Input

The multi-model data integration based solution for the movie recommendations which can introduce inputs as movie data, movie review data obtained from multiple data sources. For this purpose, TMDB API, IMDB API, YouTube API, Movie Lens dataset is selected to get movie data and movie review data.

4.4 Output

This research produces a list of recommended movies as the output. Here the list of recommended movies are presented with input parameters, movies that the user preferred.

4.5 Process

Recommendation module is the core of the proposed solution. This involves multiple tasks pertaining from pre-processing of data to visualization of the output results. These tasks include preprocessing of data, performing sentiment analysis on movie reviews collected from IMDB API, creating movie profiles, and getting recommendations through content-based filtering recommendation module and optimizing the recommendations using evolutionary computing algorithm.

To get the data from IMDB API, TMDB API and YouTube API registration is required to get the token keys which acts as the consent to stream the data. Keyword filtering can be used to crawl the most relevant information from the APIs. The collected data is subjected to a pre-processing stage to capture the relevant information. The preprocessed data is passed into the review analysis module and profiling module.

In the profile extraction process, useful information will be extracted from different sources. Here we have to perform two basic profile creation steps as user profiling and movie profiling. In the movie profiling and user profiling step following sub tasks are performed.

Movie profiling:

- Mining features against different data sources to discover the category of the movie, user groups who watched the movie most, level of interest (i.e. boring or enjoying).
- To determine the level of acceptance by the audience, review analysis is performed which helps to further classify movies.
- Rate the movie based on YouTube views, comments received etc.

User profiling:

- Mining user interest from the existing data sources.
- To classify users into one more user categories.
- Identify movies a given user has watched in the past and mine the interest from that.

After generating the profiles, that information is passed into the content-based filtering recommendation module to generate recommendations. The recommended movie list is utilized as the population for the evolutionary computing module. In the evolutionary computing module movies act as the genes and chromosomes will be generated by using gene values. Besides, crossover and mutation operations will be performed, and fitness of individuals will be calculated. Offspring will be generated through generations, until the termination criteria is met. The best-fitted movie combination will be chosen by evaluating the fitness of the offspring

4.6 Users

Proposed system can be used by movie lovers belonging to any age category who loves to watch movies.

4.7 Features

Output of the recommender system will be displayed on a user interface.

4.8 Summary

This chapter presented the approach on the personalised movie recommender system, including the hypothesis, input, the process, output users and features. Here the hypothesis of research is defined as movie recommendations can be done using multi model data integration. In the next chapter, we give a detailed discussion on the design of the personalised movie recommender system.

CHAPTER 5

DESIGN

5.1 Introduction

Chapter 4 presented our approach, personalised movie recommendation using multi model data integration. This chapter presents the top-level architecture of the personalised movie recommender system which comprises five modules namely, pre-processing module, review analysis module, profiling module, content-based filtering module and evolutionary computing module. The top-level architecture of personalised movie recommendation using multi model data integration is shown in Figure 5.1.1.

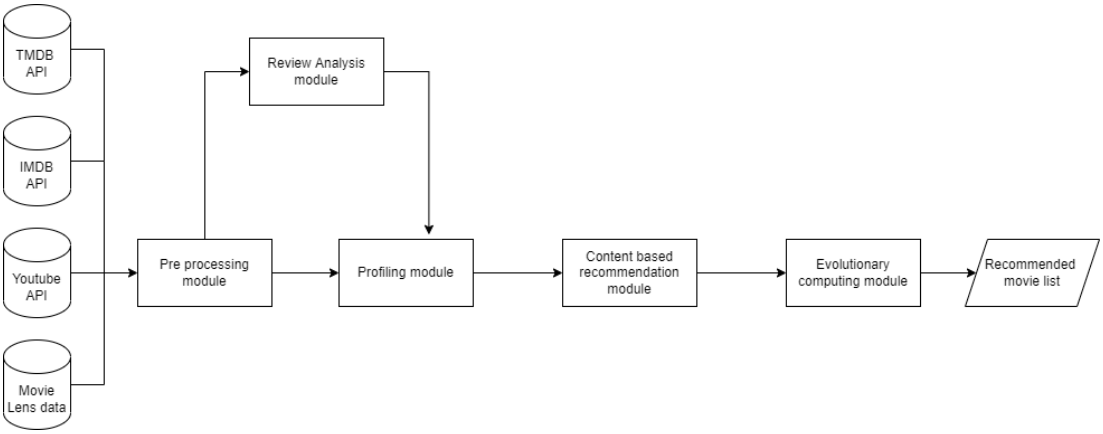


Figure 5.1. 1 Top level architecture of the personalised movie recommender system

5.2 Pre-Processing Module

Pre-processing module pre-processes the data extracted from TMDB API, IMDB API, YouTube API and data obtained from Movie Lens dataset. Major pre-processing techniques like removal of missing values, stemming, stop word removal, punctuation removal are conducted. Pre-processed data are then mapped into CSV files and passed as inputs to the review analysis module and profiling module.

5.3 Review Analysis Module

Pre-processed text data are passed as the input for the Review analysis module. Under the review analysis module sentiment analysis will be performed on movie reviews

obtained from IMDB API. After performing sentiment analysis, a sentiment value will be calculated as the output to identify the user sentiment towards the movie.

5.4 Profiling Module

The output from the pre-processing module and review analysis modules are passed as the inputs to the profiling module. Under the profiling module, user profiles and movie profiles are created with the integration of different types of data from different data sources.

5.5 Content Based Recommendation Module

Aggregated movie data is passed as the input to the content-based recommendation module. Under the content-based recommendation module a score will be calculated after applying recommendation techniques. The Cosine similarity matrics based score is calculated analysing the similarity between items. The predefined number of top N recommendations will be generated based on the calculated similarity score.

5.6 Evolutionary Computing Module

The recommended movie list is passed into the evolutionary computing module as input and optimized movie list is obtained as the output. A fitness function is evaluated and fitness value is generated and best-fitted individuals are chosen for the next generation. Single point crossover and swap mutation operations are performed on the chosen population until the termination condition is met. Once the termination criteria are met, the best-fitted individual is selected and the gene values of the individual is obtained as the result.

The main steps of genetic algorithm-based approach is shown in Figure 5.6.1.

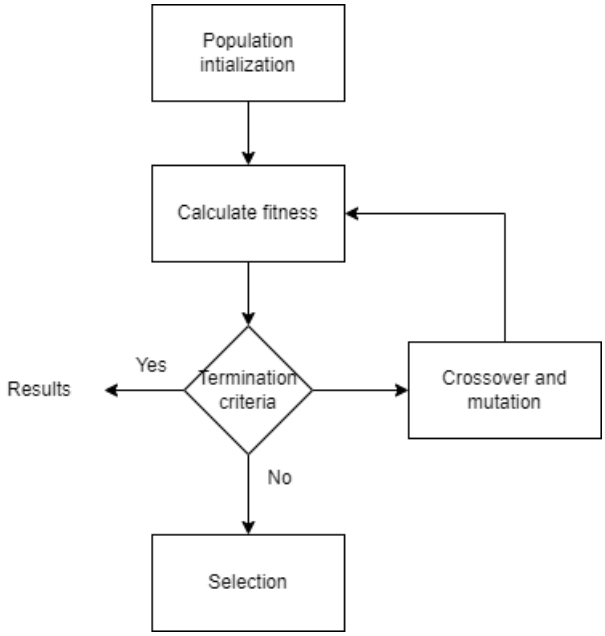


Figure 5.6. 1 The main steps involve in genetic algorithm

Using this novel approach to generate personalised movie recommendations, we improve the quality and the accuracy of the movie recommendations.

5.7 Summary

This chapter presented the design of the personalised movie recommender system, including details on the pre-processing module, review analysis module, profiling module and content-based filtering recommendation module. In the next chapter, we give a detailed discussion on the implementation of the personalised movie recommender system.

CHAPTER 6

IMPLEMENTATION

6.1 Introduction

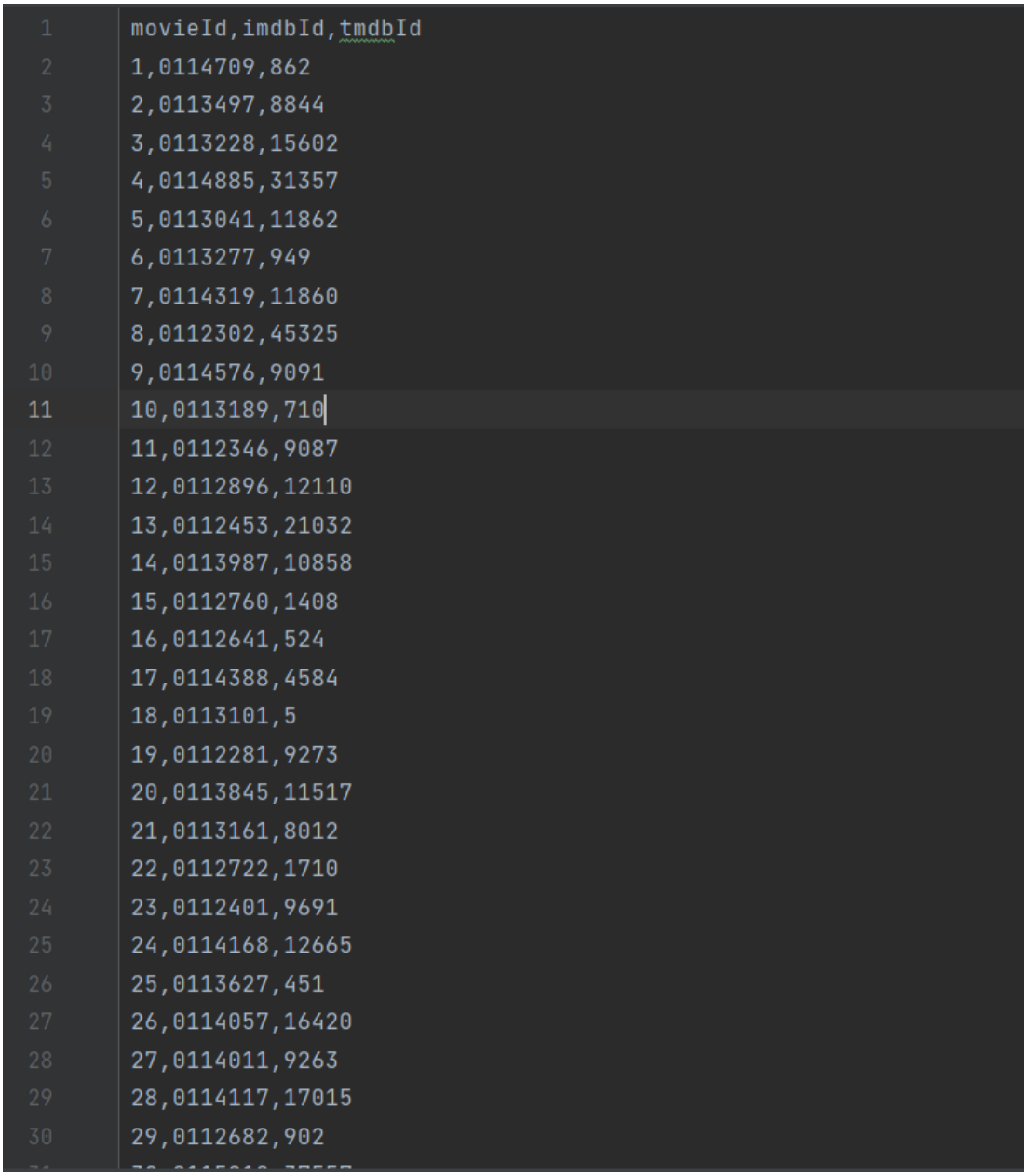
In chapter 4, we presented the design of the proposed multi model data integration-based movie recommender system that can be used to solve our research problem of inadequate research to achieve quality and the accuracy of personalised movie recommendations. In addition to that, module wise design details are discussed in the previous chapter. This chapter has been structured to discuss the implementation details behind the proposed multi model data integration-based movie recommender system.

6.2. Implementation

Module wise implementation details are discussed as follows.

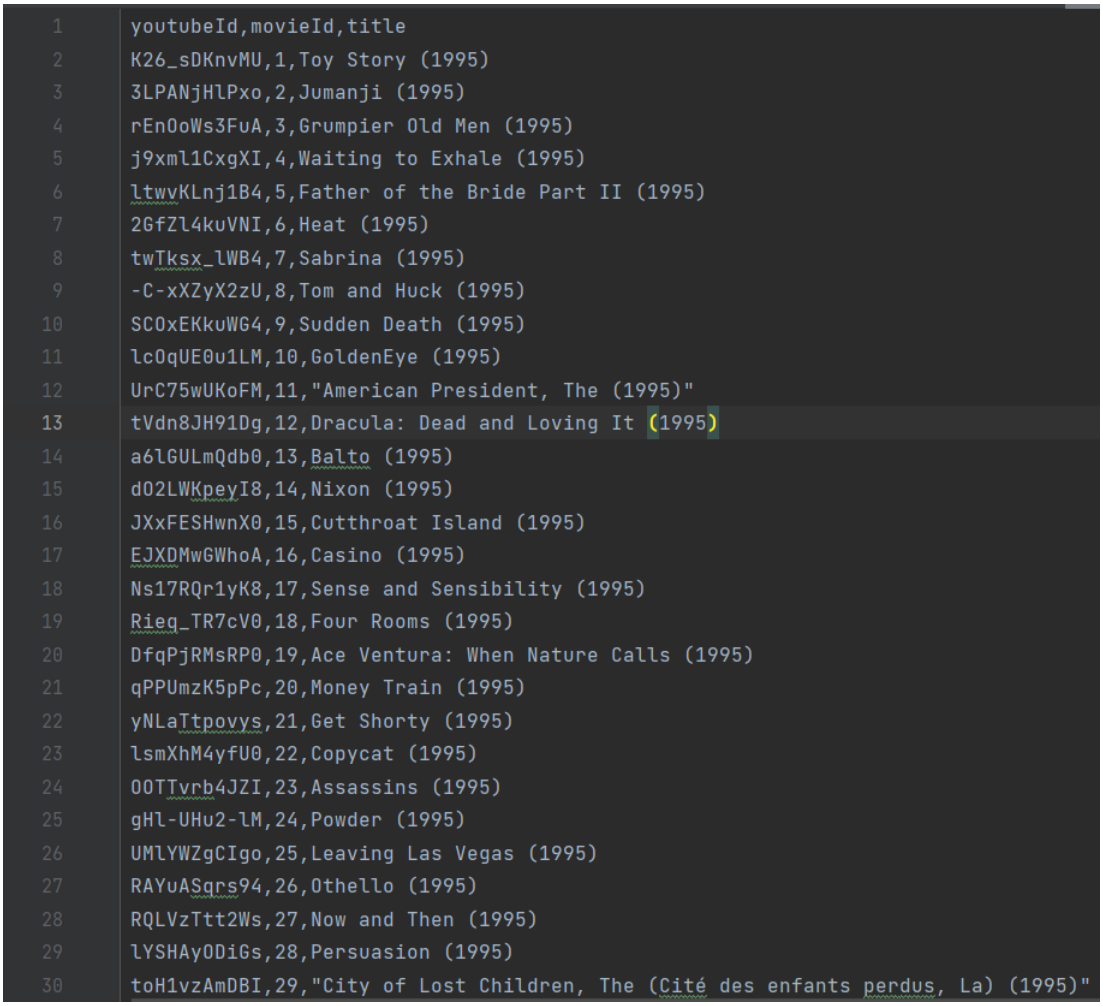
6.2.1 preprocessing module

Under the data preprocessing module data is extracted from TMDB API, IMDB API and YouTube API. Mappings available in Movie Lens dataset are used to extract the data from the APIs. In order to get the data from IMDB API, TMDB API and YouTube API registration is required to get the token keys which acts as the consent to stream the data. Data extraction scripts are written using Python and Java programming languages. Extracted data is saved into a Mongo DB database. Movie data is saved to movie collection and review data is saved to reviews collection.



```
movieId,imdbId,tmdbId
1,0114709,862
2,0113497,8844
3,0113228,15602
4,0114885,31357
5,0113041,11862
6,0113277,949
7,0114319,11860
8,0112302,45325
9,0114576,9091
10,0113189,710
11,0112346,9087
12,0112896,12110
13,0112453,21032
14,0113987,10858
15,0112760,1408
16,0112641,524
17,0114388,4584
18,0113101,5
19,0112281,9273
20,0113845,11517
21,0113161,8012
22,0112722,1710
23,0112401,9691
24,0114168,12665
25,0113627,451
26,0114057,16420
27,0114011,9263
28,0114117,17015
29,0112682,902
```

Figure 6.2.1. 1 Mappings in Movie Lens dataset for TMDB Id and IMDB Id



```
youtubeId,movieId,title
K26_sDKnvMU,1,Toy Story (1995)
3LPANjHlPxo,2,Jumanji (1995)
rEnOoWs3FuA,3,Grumpier Old Men (1995)
j9xml1CxgXI,4,Waiting to Exhale (1995)
ltwvKLnj1B4,5,Father of the Bride Part II (1995)
2GfZl4kuVNI,6,Heat (1995)
twTksx_lWB4,7,Sabrina (1995)
-C-xXZyX2zU,8,Tom and Huck (1995)
SCOxEKkuWG4,9,Sudden Death (1995)
lcOqUE0u1LM,10,GoldenEye (1995)
UrC75wUKoFM,11,"American President, The (1995)"
tVdn8JH91Dg,12,Dracula: Dead and Loving It (1995)
a6lGULmQdb0,13,Balto (1995)
dO2LWKpeyI8,14,Nixon (1995)
JXxFESHwnX0,15,Cutthroat Island (1995)
EJXDMwGWhoA,16,Casino (1995)
Ns17RQr1yK8,17,Sense and Sensibility (1995)
Rieq_TR7cV0,18,Four Rooms (1995)
DfqPjRMsRP0,19,Ace Ventura: When Nature Calls (1995)
qPPUmzK5pPc,20,Money Train (1995)
yNLaTtpovys,21,Get Shorty (1995)
lsmXhM4yfU0,22,Copycat (1995)
OOTTvrb4JZI,23,Assassins (1995)
gHl-UHu2-lM,24,Powder (1995)
UMlYWZgCIgo,25,Leaving Las Vegas (1995)
RAYuASqrs94,26,Othello (1995)
RQLVzTtt2Ws,27,Now and Then (1995)
lYSHAyODiGs,28,Persuasion (1995)
toH1vzAmDBI,29,"City of Lost Children, The (Cité des enfants perdus, La) (1995)"
```

Figure 6.2.1. 2 Mappings in Movie Lens dataset for YouTube Id and Movie Id

In [8]:
```
import pandas as pd

# read the data set into a pandas dataframe
df = pd.read_csv('Questionnaire to identify user interests and behaviors towards movies (Responses) - Form Responses.csv')
```

In [9]:
```
# drops the 'Timestamp' column
df = df.drop('Timestamp', 1)
```

In [10]:
```
# list  column names
list(df.columns)
```

Out[10]:
```
['What is your name?',
 'What is your gender?',
 'What is your age category?',
 'How much do you like watching movies?',
 'Which way do you prefer to watch movies?',
 'At what time do you prefer to watch movies?',
 'With whom do you watch movies with?',
 'What criteria you focus on when choosing a movie?',
 'How often do you watch movies? [In theaters]',
 'How often do you watch movies? [Rent (pay per individual movie)]',
 'How often do you watch movies? [Paid movie subscription (Netflix, etc.)]',
 'How often do you watch movies? [Personal library (movies you own)]',
 'How often do you watch movies? [Free online streaming (Hulu, etc.)]',
 'How much do you like watching movies from the following genres? [Action]',
 'How much do you like watching movies from the following genres? [Adventure]',
 'How much do you like watching movies from the following genres? [Animation]',
 'How much do you like watching movies from the following genres? [Children]',
 'How much do you like watching movies from the following genres? [Comedy]',
 'How much do you like watching movies from the following genres? [Crime]',
 'How much do you like watching movies from the following genres? [Documentary]',
 'How much do you like watching movies from the following genres? [Drama]',
 'How much do you like watching movies from the following genres? [Fantasy]',
 'How much do you like watching movies from the following genres? [Horror]',
 'How much do you like watching movies from the following genres? [Musical]',
 'How much do you like watching movies from the following genres? [Mystery]',
 'How much do you like watching movies from the following genres? [Sci-Fi]',
 'How much do you like watching movies from the following genres? [Thriller]',
 'How much do you like watching movies from the following genres? [War]',
 'How much do you like watching movies from the following genres? [Western]',
 "In general, how much do you like movies that were released in the following decades? [1920's]",
 "In general, how much do you like movies that were released in the following decades? [1930's]",
 "In general, how much do you like movies that were released in the following decades? [1940's]",
 "In general, how much do you like movies that were released in the following decades? [1950's]",
 "In general, how much do you like movies that were released in the following decades? [1960's]",
 "In general, how much do you like movies that were released in the following decades? [1970's]",
 "In general, how much do you like movies that were released in the following decades? [1980's]",
 "In general, how much do you like movies that were released in the following decades? [1990's]",
 "In general, how much do you like movies that were released in the following decades? [2000's]",
 "In general, how much do you like movies that were released in the following decades? [2010's]",
 "In general, how much do you like movies that were released in the following decades? [2020's]",
 'Watching movies can make you, ',
 'Do you consider the running time when choosing a movie?',
```

```
In [11]: # Rename columns
data = df.rename(columns =
                 {
                  'What is your name?':'name',
                  'What is your gender?':'gender',
                  'What is your age category?':'ageCategory',
                  'How much do you like watching movies?':'preferenceForWatchingMovies',
                  'Which way do you prefer to watch movies?':'methodsOfWatchingMovies',
                  'At what time do you prefer to watch movies?':'preferredTimeSlotForWatchingMovies',
                  'With whom do you watch movies with?':'withWhomYouWatchMovies',
                  'What criteria you focus on when choosing a movie?':'focussedCriteriaOnChoosingMovie',

                  'How often do you watch movies? [In theaters]':'frequencyOfWatchingMovies-[In theaters]',
                  'How often do you watch movies? [Rent (pay per individual movie)]':'frequencyOfWatchingMovies-[Rent (pay per ir
                  'How often do you watch movies? [Paid movie subscription (Netflix, etc.)]':'frequencyOfWatchingMovies-[Paid mov
                  'How often do you watch movies? [Personal library (movies you own)]':'frequencyOfWatchingMovies-[Personal libra
                  'How often do you watch movies? [Free online streaming (Hulu, etc.)]':'frequencyOfWatchingMovies-[Free online s

                  'How much do you like watching movies from the following genres? [Action]':'genrePreference-[Action]',
                  'How much do you like watching movies from the following genres? [Adventure]':'genrePreference-[Adventure]',
                  'How much do you like watching movies from the following genres? [Animation]':'genrePreference-[Animation]',
                  'How much do you like watching movies from the following genres? [Children]':'genrePreference-[Children]',
                  'How much do you like watching movies from the following genres? [Comedy]':'genrePreference-[Comedy]',
                  'How much do you like watching movies from the following genres? [Crime]':'genrePreference-[Crime]',
                  'How much do you like watching movies from the following genres? [Documentary]':'genrePreference-[Documentary]'
                  'How much do you like watching movies from the following genres? [Drama]':'genrePreference-[Drama]',
                  'How much do you like watching movies from the following genres? [Fantasy]':'genrePreference-[Fantasy]',
                  'How much do you like watching movies from the following genres? [Horror]':'genrePreference-[Horror]',
                  'How much do you like watching movies from the following genres? [Musical]':'genrePreference-[Musical]',
                  'How much do you like watching movies from the following genres? [Mystery]':'genrePreference-[Mystery]',
                  'How much do you like watching movies from the following genres? [Sci-Fi]':'genrePreference-[Sci-Fi]',
                  'How much do you like watching movies from the following genres? [Thriller]':'genrePreference-[Thriller]',
                  'How much do you like watching movies from the following genres? [War]':'genrePreference-[War]',
                  'How much do you like watching movies from the following genres? [Western]':'genrePreference-[Western]',

                  'In general, how much do you like movies that were released in the following decades? [1920\'s]':'preferenceOnM
                  'In general, how much do you like movies that were released in the following decades? [1930\'s]':'preferenceOnM
                  'In general, how much do you like movies that were released in the following decades? [1940\'s]':'preferenceOnM
                  'In general, how much do you like movies that were released in the following decades? [1950\'s]':'preferenceOnM
                  'In general, how much do you like movies that were released in the following decades? [1960\'s]':'preferenceOnM
                  'In general, how much do you like movies that were released in the following decades? [1970\'s]':'preferenceOnM
                  'In general, how much do you like movies that were released in the following decades? [1980\'s]':'preferenceOnM
                  'In general, how much do you like movies that were released in the following decades? [1990\'s]':'preferenceOnM
                  'In general, how much do you like movies that were released in the following decades? [2000\'s]':'preferenceOnM
                  'In general, how much do you like movies that were released in the following decades? [2010\'s]':'preferenceOnM
                  'In general, how much do you like movies that were released in the following decades? [2020\'s]':'preferenceOnM

                  'Watching movies can make you, ':'reasonForWatchingMovies',
                  'Do you consider the running time when choosing a movie?':'considersRunningTime',
                  'How do you get recommendations when choosing a movie?':'methodsOfGettingRecommendations',
                  'When choosing a movie, do you like to watch a movie from the current trending list?':'preferenceToTrendingList
                  'Name a few of your favorite movies':'favoriteMovies',
                  'Which do you prefer? ':'preferredMovieType',
                  'Do you like genre hybrids (two or more genres combined)?':'preferenceForGenreHybrids'
                     })
```

```
In [14]: # Save results to a new csv file
data.to_csv('user-data-cleaned.csv')
```

Figure 6.2.1. 3 Preprocessing user data

```
@PostConstruct
public void getYoutubeVideoDetails() {
    try {
        //TODO: 3 separate csv files used as daily limit is 10000
        BufferedReader br = new BufferedReader(new FileReader( fileName: "src/main/resources/data/ml-youtube-4.csv"));
        br.readLine();
        while ((line = br.readLine()) != null)
        {
            try {
                String[] video = line.split(splitBy);    // use comma as separator
                System.out.println("Movie Id " + video[1]);

                final YoutubeVideoResponseDTO youtubeVideoResponseDTO = restTemplate.getForEntity(UriComponentsBuilder.fromUriString(youtubeVideoApi).queryParam( name: "id", vide
                        .queryParam( name: "key", youtubeApiKey)
                        .queryParam( name: "part", ...values: "snippet,contentDetails,statistics,status")
                        .toUriString(), YoutubeVideoResponseDTO.class).getBody();

            LOGGER.info("Retrieving movie details");
                final Movie movie = movieRepository.findById(video[1]).get();
            LOGGER.info("Retrieved movie details");

                movie.setYoutubeStatistics(new StatisticsDTO(youtubeVideoResponseDTO.getItems().get(0).getStatistics().getViewCount(),
                        youtubeVideoResponseDTO.getItems().get(0).getStatistics().getLikeCount(), youtubeVideoResponseDTO.getItems().get(0).getStatistics().getDislikeCount(),
                        youtubeVideoResponseDTO.getItems().get(0).getStatistics().getFavoriteCount(), youtubeVideoResponseDTO.getItems().get(0).getStatistics().getCommentCount()
            movieRepository.save(movie);

            } catch (Exception e){
                LOGGER.error("An error occurred while retrieving youtube data");
            }
        }
    } catch (IOException e) {
        LOGGER.error("An error occurred while retrieving youtube data");
```

Figure 6.2.1. 4 Java method to get YouTube video details

```python
import requests
import json
import pymongo
import pandas as pd
from pymongo import MongoClient
from pymongo import ReturnDocument

api_key = "k_mmic14oo"
base_url = "http://localhost:8080/api/reviews/"

def get_review_data(tmdbId):
    api = base_url + tmdbId
    response = requests.request("GET", api)
    json_data = json.loads(response.text)
    content_list = []

    if not json_data['items']:
        content_list = []
    else:
        for i in range(len(json_data['items'])):
            content = json_data['items'][i]['content']
            content_list.append(str(content))
            data = str(content)
        create_json_object(tmdbId, content_list)
    return content_list

def get_id_from_file(filename):
    with open(filename) as f:
        next(f)
        for line in f:
            tmdbId,imdbId = line.strip().split(',')
            get_imdb_data(tmdbId, imdbId)
            if 'str' in line:
                break

def create_json_object(tmdbId, content_list):
    db_connect = pymongo.MongoClient("localhost", 27017)
    database_name = 'movie-review'
    database = db_connect[database_name]
    try:
        conn = MongoClient()
    except:
        print("Could not connect to MongoDB")

    user_rec = {
        "tmdbId":tmdbId,
        "review":content_list
        }

    # database
    db = conn.database
    print(tmdbId)
    rec_id1 = database.reviews.insert_one(user_rec)
```

Figure 6.2.1. 5 Python script to get IMDB review data

TMDB-IMDB Data Set Creation

In [65]:
```
import pandas as pd
data = pd.read_csv('links.csv')
cleaned_data = data.dropna()
cleaned_data.to_csv('links_cleaned.csv', index=False)
```

In [1]:
```
import requests
import json

# base request URL
base_url = "https://api.themoviedb.org/3/movie/"

# query string for the request URL
querystring = {"api_key":"e24efe85215222c3359fc508bc686a9f", "language":"en-US"}

text_file1 = open("tmdb-imdb-data.csv", "a")
header = "tmdb_id,imdb_id"+"\n"
text_file1.write(header)
text_file1.close()


def get_tmdb_data(tmdbId):
    # send a GET request
    tmdb_api = base_url + tmdbId
    response = requests.request("GET", tmdb_api, params=querystring)

    #open a new text file inorder to write the content
    text_file = open("tmdb-imdb-data.csv", "a")

    json_data = json.loads(response.text)
    if (json_data['id'] is None):
        return
    else:
        tmdb_id = json_data['id']
        imdb_id = json_data['imdb_id']
        data = str(tmdb_id) +"," + str(imdb_id) + "\n"
        print(data)
        text_file.write(data)
        text_file.close()

def write_data_to_csv():
    with open("links_cleaned.csv") as f:
        next(f)
        for line in f:
            movieId,imdbId,tmdbId = line.strip().split(',')
            get_tmdb_data(tmdbId)
            if 'str' in line:
                break
```

In [64]:
```
write_data_to_csv()
```

Figure 6.2.1. 6 Python script to create TMDB and IMDB mapping dataset

6.2.2 Review analysis module

Reviews extracted from IMDB API are passed into the review analysis module and a sentiment score is obtained on the sentiment of the review. Naive Bayes classifier is used to perform the sentiment analysis of the review data. Before passing the data into the Naive Bayes algorithm, processing steps like stemming, stop-word removal, punctuation removal and converting the reviews into lower case letters are performed. The sentiment score will be calculated per review and sentiment score along with the movie Id is written to a CSV file.

```
import pandas as pd
from sklearn.model_selection import train_test_split
import joblib
from sklearn.feature_extraction.text import CountVectorizer
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer
from sklearn.naive_bayes import MultinomialNB
import re
import nltk
import pymongo
from pymongo import MongoClient
from pymongo import ReturnDocument

data = pd.read_csv('google_play_store_apps_reviews_training.csv')
data.head()
nltk.download('stopwords')

def preprocess_data(data):
    corpus = []

    # loop through the no of rows to clean
    for i in range(0, len(data)):
        review = re.sub('[^a-zA-Z]', ' ', data['review'][i])
        review = review.lower()
        review = review.split()
        ps = PorterStemmer()

        # loop for stemming each word in string array at ith row
        review = [ps.stem(word) for word in review
                    if not word in set(stopwords.words('english'))]

        # rejoin all string array elements o create back into a string
        review = ' '.join(review)

        # append each string to create array of clean text
        corpus.append(review)
        return data

data = preprocess_data(data)

# Split into training and testing data
x = data['review']
y = data['polarity']
x, x_test, y, y_test = train_test_split(x,y, stratify=y, test_size=0.25, random_state=42)

# Vectorize text reviews to numbers
vec = CountVectorizer(stop_words='english')
x = vec.fit_transform(x).toarray()
x_test = vec.transform(x_test).toarray()

model = MultinomialNB()
model.fit(x, y)
model.score(x_test, y_test)
prediction = model.predict(vec.transform(['best one ever']))

prediction[0]
```

Figure 6.2.2. 1 Naive Bayes classifier to get review sentiment

```
import requests
import json
import re

base_url = "http://localhost:8080/api/reviews/"

db_connect = pymongo.MongoClient("localhost", 27017)
database_name = 'movie-review'
database = db_connect[database_name]

def get_review_data(tmdbId):
    api = base_url + tmdbId
    response = requests.request("GET", api)
    json_data = json.loads(response.text)
    content_list = []
    sentiment =[]
    final_df = pd.DataFrame()

    if not json_data['reviews']:
        content_list = []
    else:
        for i in range(len(json_data['reviews'])):

            review=json_data['reviews'][i]
            review = re.sub('[^a-zA-Z]', ' ', review)
            review = review.lower()
            prediction = model.predict(vec.transform([review]))
            sentiment.append(prediction[0])
        df_ = pd.DataFrame({"tmdbId":tmdbId,"sentiment":sentiment})
        df_ = df_.groupby('tmdbId')['sentiment'].apply(lambda x: x.value_counts().index[0]).reset_index()
        cursor = database.movie.update_one({'tmdbId' : tmdbId}, {'$set' : {'review_sentiment' : str(df_['sentiment'][0])}})
```

```
data = pd.read_csv('reviews-tmdbId.csv')

for i in range(0, len(data)):
    get_review_data(str(data['tmdbId'][i]))
```

Figure 6.2.2. 2 Integration of review sentiment with movie profiles

6.2.3 The Profiling module

Under the profiling module 2 types of profiles are created, namely user profiles and movie profiles. The movie data obtained through TMDB API, YouTube API, IMDB API and Movie Lens data are integrated together to form movie profiles. User profiles are generated using the user interest data collected from the questionnaire. After performing the review analysis sentiment value towards the movie is also integrated to the movie profiles. Pre-processed user data obtained from a questionnaire is also integrated to form user profiles. All the profiles are saved into a Mongo database.

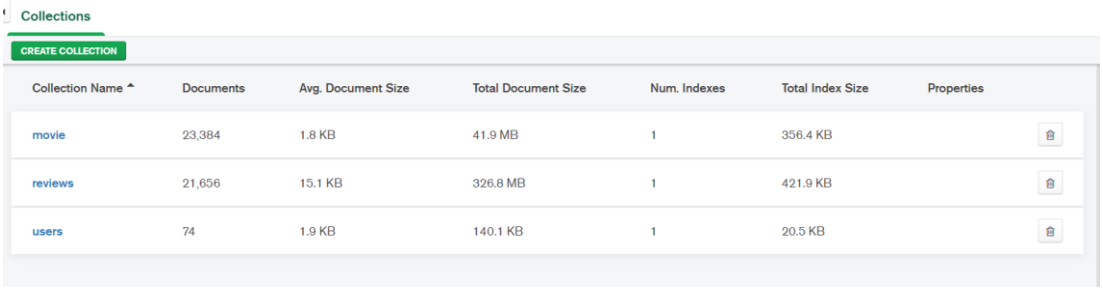


Figure 6.2.3. 1 Integration of review sentiment with movie profiles

```
_id: "1"
tmdbId: "862"
imdbId: "tt0114709"
title: "Toy Story"
genres: Array
    0: "Animation"
    1: "Adventure"
    2: "Family"
    3: "Comedy"
popularity: 165.112
release_date: "1995-10-30"
revenue: 165.112
runtime: 81
vote_average: 8
vote_count: 14850
tagline: ""
overview: "Led by Woody, Andy's toys live happily in his room until Andy's birthd..."
budget: 165.112
cast: Array
crew: Array
youtubeStatistics: Object
    viewCount: 104851
    likeCount: 104
    dislikeCount: 0
    favoriteCount: 0
    commentCount: 0
_class: "com.research.dataCollector.domain.Movie"
review_sentiment: "1"
```


Figure 6.2.3. 2 Sample movie profile


```
_id: ObjectId("61fdfe2c0ced6c859738fc5f")
tmdbId: "862"
review: Array
    0: ""
    1: ""
    2: "I can see why this was a big hit at the time. Well made and with enoug..."
    3: "Andy's toys come to life. Woody (Tom Hanks) is his favorite until a ne..."
    4: "What a wonderful integration of classic toys and a superb plot. It's a..."
    5: ""
    6: "The original meaning of a playmate that is of course. Pixar is inventi..."
    7: "With films like "Toy Story", I feel like I'm going out of my movie com..."
    8: "Pixar became a household name in 1995 with "Toy Story", about cowboy W..."
    9: "This was the very first computer animated film from Walt Disney Pictur..."
    10: ""
    11: "While I don't think 3D computer animation has the charm of well made h..."
    12: "A kid's bedroom full of toys comes to life when humans aren't around; ..."
    13: "Toy Story is a sheer delight to view on the screen. The characters are..."
    14: ""
    15: "This film will continue to delight children for many years to come and..."
    16: "Just who knew how well this Disney-Pixar collaboration would be accept..."
    17: "Toy Story is a computer animated comedy produced by Pixar Animation St..."
    18: ""
    19: ""
    20: "I like the bit where he said trouble where down there"
    21: ""
    22: "Rollicking, entertaining adventure. Good fun, with some great humour, ..."
    23: "The first computer animated film from Pixar that dazzled audiences and..."
    24: "It was a classic on the day it was released, and it's still a classic ..."
```


Figure 6.2.3. 3 Sample review data obtained from IMDB API

```
# Inserting data in MongoDB
import pymongo
import pandas as pd
from pymongo import MongoClient

db_connect = pymongo.MongoClient("localhost", 27017)
database_name = 'movie-review'
database = db_connect[database_name]

try:
    conn = MongoClient()
    print("Connected successfully!!!")
except:
    print("Could not connect to MongoDB")

db = conn.database
collection = database.users
df = pd.read_csv('user-data-cleaned.csv')

print(len(df.index))

# Function to count words in a string.
def word_count(string):
    tokens = string.split()
    n_tokens = len(tokens)
    return n_tokens

def get_name():
    if word_count(df['name'][i]) == 1:
        return ""
    else:
        return df['name'][i].split(" ")[1]

def create_json_object(i):
    lastname = get_name()
    fav_movies = df['favoriteMovies'][i].replace(' \n',', ').replace('\n',', ')[:-2]

    user_rec = {
            "_id":i,
            "firstName":df['name'][i].split(" ")[0],
            "lastName": lastname,
            "gender":df['gender'][i],
            "ageCategory":df['ageCategory'][i],
            "preferenceForWachingMovies":df['preferenceForWatchingMovies'][i],
            "methodsOfWatchingMovies":[df['methodsOfWatchingMovies'][i]],
            "preferredTimeSlotForWatchingMovies":df['preferredTimeSlotForWatchingMovies'][i],
            "withWhomYouWatchMovies":[df['withWhomYouWatchMovies'][i]],
            "focussedCriteriaOnChoosingMovie":[df['focussedCriteriaOnChoosingMovie'][i]],
            "frequencyOfWatchingMovies": {
                    "theaters": df['frequencyOfWatchingMovies-[In theaters]'][i],
                    "rent": df['frequencyOfWatchingMovies-[Rent (pay per individual movie)]'][i],
                    "paid subscription" : df['frequencyOfWatchingMovies-[Paid movie subscription (Netflix, etc.)]'][i],
                    "personal library": df['frequencyOfWatchingMovies-[Personal library (movies you own)]'][i],
                    "online streaming":df['frequencyOfWatchingMovies-[Free online streaming (Hulu, etc.)]'][i]
            },

            "genrePreference":{
                "action":[df['genrePreference-[Action]'][i]],
                "adventure": [df['genrePreference-[Adventure]'][i]],
                "animation": [df['genrePreference-[Animation]'][i]],
                "children": [df['genrePreference-[Children]'][0]],
                "comedy": [df['genrePreference-[Comedy]'][i]],
                "crime": [df['genrePreference-[Crime]'][i]],
                "documentary": [df['genrePreference-[Documentary]'][i]],
                "drama": [df['genrePreference-[Drama]'][i]],
                "fantasy": [df['genrePreference-[Fantasy]'][i]],
                "horror": [df['genrePreference-[Horror]'][i]],
                "musical": [df['genrePreference-[Musical]'][i]],
                "mystery": [df['genrePreference-[Mystery]'][i]],
                "scifi": [df['genrePreference-[Sci-Fi]'][i]],
                "thriller": [df['genrePreference-[Thriller]'][i]],
                "war": [df['genrePreference-[War]'][i]],
                "western": [df['genrePreference-[Western]'][i]]
            },
            "preferenceOnMoviesOfDifferentDecades":{
                "1920": [df['preferenceOnMoviesOfDifferentDecades-1920'][i]],
                "1930": [df['preferenceOnMoviesOfDifferentDecades-1930'][i]],
                "1940": [df['preferenceOnMoviesOfDifferentDecades-1940'][i]],
                "1990": [df['preferenceOnMoviesOfDifferentDecades-1990'][i]],
                "2000": [df['preferenceOnMoviesOfDifferentDecades-2000'][i]],
                "2010": [df['preferenceOnMoviesOfDifferentDecades-2010'][i]],
                "2020": [df['preferenceOnMoviesOfDifferentDecades-2020'][i]]
            },
            "reasonForWatchingMovies":[df['reasonForWatchingMovies'][i]],
            "considersRunningTime":df['considersRunningTime'][i],
            "methodsOfGettingRecommendations":[df['methodsOfGettingRecommendations'][i]],
            "favoriteMovies": fav_movies.split(','),
            "preferenceToTrendingList":df['preferenceToTrendingList'][i],
            "preferredMovieType":df['preferredMovieType'][i],
            "preferenceForGenreHybrids":df['preferenceForGenreHybrids'][i]
            }


    # Insert Data
    rec_id1 = collection.insert_one(user_rec)

    print("Data inserted successfully")


for i in range(len(df.index)):
    create_json_object(i)
```

Figure 6.2.3. 4 Creation of user profiles

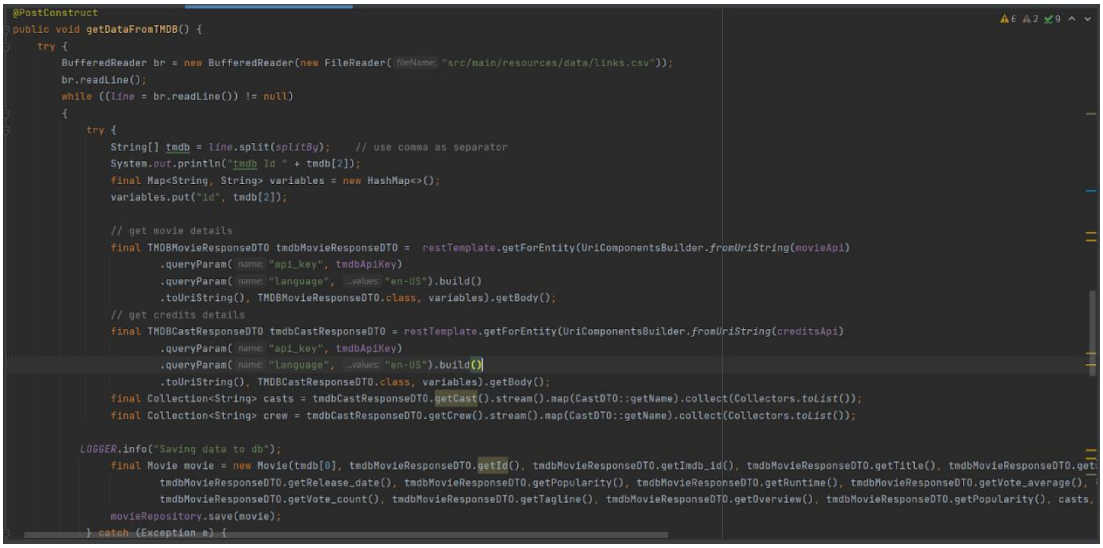


Figure 6.2.3. 5 Creation of movie profiles

6.2.4 Content based recommendation module

Movie profiles are passed as the input for the content-based recommendation module. Cosine similarity metric is calculated to identify the similarity between movies. The pre-defined N number of recommendations are obtained as the output of the content-based recommendation module. The results of the content-based filtering are saved into a CSV file and evolutionary computing module consumes the results.

```
movies_df = pd.read_csv('movie-index.csv')
features = ['genres', 'title', 'cast', 'crew', 'vote_average','popularity','commentCount',
            'likeCount','viewCount','review_sentiment']
movies_df['cast'].isnull().values.any()

def combine_features(row):
    return row['title']+' '+row['genres']+' '+row['crew']+' '+row['cast']+' '+ str(row['vote_average'])+' '+str(row['popu

def get_title_from_index(index):
    return movies_df[movies_df.index == index]["title"].values[0]

def get_index_from_title(title):
    movie_index = movies_df[movies_df.title == title]["index"]
    if movie_index.empty:
        return None
    else:
        return movie_index.values[0]

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

movies_df['combined_features'] = movies_df.apply(combine_features, axis = 1)
cv = CountVectorizer()
count_matrix = cv.fit_transform(movies_df['combined_features'])
cosine_sim = cosine_similarity(count_matrix)

# this movie name should be in the data set
movie_index = get_index_from_title(movie_user_likes)
similar_movies = list(enumerate(cosine_sim[movie_index])) #accessing the row corresponding to given movie to find all the
sorted_similar_movies = sorted(similar_movies,key=lambda x:x[1],reverse=True)[1:]

recommended_movie_list=[]
i=0
print("Top 4 similar movies to " + movie_user_likes + " are:\n")
for element in sorted_similar_movies:
    title = get_title_from_index(element[0])
    print(title)
    recommended_movie_list.append(title)
    i=i+1
    if i>=4:
        break
return {'movies': recommended_movie_list}
```

Figure 6.2.4. 1 Multi model data integration-based content-based filtering implementation

6.2.5 Evolutionary computing module

Genetic algorithm approach is used as an optimization technique to improve the recommended movie list. The results obtained from the content-based filtering module are passed as the input to the evolutionary computing module. The recommendation list acts as the initial population for the genetic algorithm. A single movie recommendation acts as the gene for the chromosomes. A set of individuals are created with the movies obtained from the recommendation module. A score will be calculated against individual movies matching with the genres of the user preferred movie. A fitness function is evaluated, and fitness value will be calculated by matching the genres of the user provided movie against the recommended movies. A score will be generated as the output of the fitness function. The best-fitted individuals are selected from the population and single point crossover and swap mutation operations are performed on the chosen population until the termination condition is met. The elitist selection method is used as the selection method as it has the ability to select the individuals from the parent and the offspring population to generate the next

population. Once the termination criteria are met, the best-fitted individual is selected and the gene values of the individual is obtained as the result. As the gene consists of the movie Ids movie titles are obtained from the movie profiles and displayed as the output.

The architecture of evolutionary computing-based approach is shown in Figure 6.1

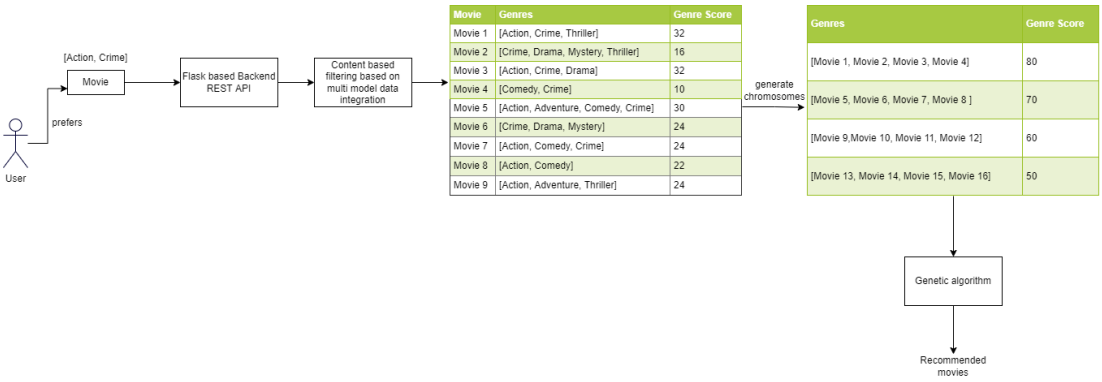


Figure 6.2.5. 1 Architecture of evolutionary computing-based approach

```
def create_individuals(data,text_file,x,y):
    list2=[]
    total=0
    val = data.iloc[x:y]['tmdb_id'].values
    score = data.iloc[x:y]['genre_score'].values
    list2.append(val[0])
    list2.append(val[1])
    list2.append(val[2])
    list2.append(val[3])
    total = score[0]+score[1]+score[2]+score[3]
    data1 = '"' + str(list2) + '"' +"," + str(total)+ "\n"
    text_file.write(data1)
    print(list2,total)

def create_chromosomes():
    data = pd.read_csv("genes.csv")
    range_val=len(data)//4
    x=0
    y=4

    text_file = open("chromosomes.csv", "w")
    header = "chromosome,score"+"\n"
    text_file.write(header)

    for i in range(range_val):
        list2 = []
        create_individuals(data, text_file, x,y)
        x+=4
        y+=4
```

Figure 6.2.5. 2 Creation of individuals and chromosomes

```
def single_point_crossover(l, q):
    k = random.randint(0, 4)
    print("Crossover point :", k)

# interchanging the genes
    for i in range(k, 4):
        l[i], q[i] = q[i], l[i]
    print(l)
    print(q,"\n\n")
    return l, q
```

Figure 6.2.5. 3 Single point crossover implementation

```
def swap_mutation(p, i1, i2):
    p[i1], p[i2] = p[i2], p[i1]
    return p
```

Figure 6.2.5. 4 Swap mutation implementation

A back-end REST API endpoint is implemented using Flask to get the recommendations and integrate with the front-end module.

6.3 Summary

This chapter presented the module wise implementation details of the multi-model data integration-based movie recommender system which comprised of modules, namely, preprocessing module, review analysis module, profiling module, the content-based recommendation module and the evolutionary computing module. In the next chapter evaluation process of personalised movie recommender system is discussed in detail.

CHAPTER 7

EVALUATION

7.1 Introduction

Chapter 6 presented the implementation details of the proposed multi model data integration-based movie recommender system that can be used to solve our research problem of inadequate research to achieve quality and the accuracy of personalised movie recommendations. In addition to that, module wise implementation details are discussed in the previous chapter. This chapter has been structured to discuss the results and evaluation details behind the proposed multi model data integration-based movie recommender system.

7.2 Results

The evaluation process of the proposed approach consists of three different experiments. The first trial was conducted to assess the results using content-based filtering with multi-model data integration. The second trial was conducted to assess the results using content-based filtering with multi model data integration and genetic algorithm-based optimization. The third experiment was conducted to test the results obtained from content-based filtering with multi model data integration and genetic algorithm-based optimization with Google recommendations. Ten experiments were conducted to test each approach.

In order to evaluate the proposed approach precision and novelty metrics were calculated. Precision is the relationship between the interested recommendations to the total number of generated recommendations. TP or true positive in the equation 1 is the count of generated interesting recommendations. FP or false positive in the equation 1 is the count of not interesting recommendations.

$$\text{Precision} = \frac{TP}{TP + FP} \quad (1)$$

Novelty determines how unknown recommendations are generated as recommendations to a user.

$$\text{Novelty} = \frac{\text{Number of unknown recommended items}}{\text{The total number of recommendations obtained}} \quad (2)$$

The term “novel recommendations” refers to generating new recommendations that the user is not familiar with. Novelty offers the capability to generate a unique and surprising recommendations that a user is unlikely to be aware of.

Experiment	Movie Name	Genres	Recommendations With GA						Recommendations Without GA					
			Recommended movies	Recommended movies genres	TP	FP	Precision	Novelty	Recommended movies	Recommended movies genres	TP	FP	Precision	Novelty
1	Garfield: A Tail of Two Kitties	Comedy	Love Actually	Comedy, Romance, Drama	3	1	3/4	3/4	Garfield	Animation, Comedy, Family	4	0	1	1/4
		Family	Harry Potter and the Prisoner of Azka	Adventure, Fantasy					Harry Potter and the Chamber of Sec	Adventure, Fantasy				
		Adventure	Armageddon	Action, Thriller, Science Fiction, Adventure					Harry Potter and the Philosopher's S	Adventure, Fantasy				
		Animation	The Truman Show	Comedy, Drama					Twisted	Action, Adventure, Crime, Drama, Thriller				
2	Thor	Adventure	War of the Worlds	Adventure, Thriller, Science Fiction	4	0	1	3/4	Captain America: The Winter Soldier	Action, Adventure, Science Fiction	4	0	1	1/2
		Fantasy	Surrogates	Science Fiction, Action, Thriller					Surrogates	Science Fiction, Action, Thriller				
		Action	Inception	Action, Science Fiction, Adventure					Inception	Action, Science Fiction, Adventure				
			The Island	Action, Thriller, Science Fiction, Adventure					The Island	Action, Thriller, Science Fiction, Adventure				
3	Interstellar	Adventure	Inception	Action, Science Fiction, Adventure	3	1	3/4	3/4	Inception	Action, Science Fiction, Adventure	4	0	1	3/4
		Drama	Starship Troopers	Adventure, Action, Thriller, Science Fiction					Surrogates	Science Fiction, Action, Thriller				
		Science Fiction	Daredevil	Fantasy, Action, Crime, Science Fiction, Thriller					The Island	Action, Thriller, Science Fiction, Adventure				
			Minority Report	Action, Thriller, Science Fiction, Mystery					Minority Report	Action, Thriller, Science Fiction, Mystery				
4	Inception	Action	Daredevil	Fantasy, Action, Crime, Science Fiction, Thriller	2	2	1/2	3/4	Interstellar	Adventure, Drama, Science Fiction	3	1	3/4	3/4
		Science Fiction	Minority Report	Action, Thriller, Science Fiction, Mystery					Surrogates	Science Fiction, Action, Thriller				
		Adventure	Star Trek	Science Fiction, Action, Adventure					The Island	Action, Thriller, Science Fiction, Adventure				
			Captain America: The Winter Soldier	Action, Adventure, Science Fiction					Men in Black 3	Action, Comedy, Science Fiction				
5	The Amazing Spider-Man 2	Action	Surrogates	Science Fiction, Action, Thriller	3	4	3/4	3/4	Men in Black 3	Action, Comedy, Science Fiction	1	3	3/4	3/4
		Adventure	Men in Black 3	Action, Comedy, Science Fiction					Surrogates	Science Fiction, Action, Thriller				
		Fantasy	Inception	Action, Science Fiction, Adventure					The Amazing Spider-Man	Action, Adventure, Fantasy				
			The Island	Action, Thriller, Science Fiction, Adventure					The Island	Action, Thriller, Science Fiction, Adventure				
6	Toy Story	Animation	A Bug's Life	Adventure, Animation, Comedy, Family	4	0	1	1/4	Toy Story 2	Animation, Comedy, Family	3	1	1	1/4
		Adventure	Toy Story 2	Animation, Comedy, Family					A Bug's Life	Adventure, Animation, Comedy, Family				
		Family	The Incredibles	Action, Adventure, Animation, Family					Monsters, Inc.	Animation, Comedy, Family				
		Comedy	Interstellar	Adventure, Drama, Science Fiction					Cars	Animation, Adventure, Comedy, Family				
7	Apollo 13	Drama	Ace Ventura: When Nature Calls	Crime, Comedy, Adventure	2	2	1/2	1	Ace Ventura: When Nature Calls	Crime, Comedy, Adventure	2	2	1/2	1
		History	Men in Black II	Action, Adventure, Comedy, Science Fiction					Men in Black II	Action, Adventure, Comedy, Science Fiction				
		Adventure	War of the Worlds	Adventure, Thriller, Science Fiction					Minority Report	Action, Thriller, Science Fiction, Mystery				
			The Patriot	Drama, History, War, Action					The Island	Action, Thriller, Science Fiction, Adventure				
8	Jurassic Park	Adventure	The Lost World: Jurassic Park	Adventure, Action, Science Fiction	3	1	3/4	3/4	The Lost World: Jurassic Park	Adventure, Action, Science Fiction	1	3	1/2	1/2
		Science Fiction	Apollo 13	Drama, History, Adventure					Apollo 13	Drama, History, Adventure				
			Minority Report	Action, Thriller, Science Fiction, Mystery					Minority Report	Action, Thriller, Science Fiction, Mystery				
			A.I. Artificial Intelligence	Drama, Science Fiction, Adventure					A.I. Artificial Intelligence	Drama, Science Fiction, Adventure				
9	Wonder Woman	Action	Green Lantern: Emerald Knights	Action, Adventure, Animation, Science-Fiction, F	2	2	1/2	1/2	Batman: Under the Red Hood	Science Fiction,Crime, Action, Animation, Myste	1	3	1/2	1/2
		Adventure	Superman: Doomsday	Action, Animation, Science Fiction, Family					Superman/Batman: Public Enemies	Science Fiction, Animation, Action, Adventure, A				
		Animation	Justice League: Doom	Action, Animation, Science Fiction					Justice League: Doom	Action, Animation, Science Fiction				
		Fantasy	Green Lantern: First Flight	Action, Adventure, Animation, Fantasy, Science-					Green Lantern: First Flight	Action, Adventure, Animation, Fantasy, Science				
		Science Fiction												
10	The Avengers	Thriller	The Day The Earth Stood Still	Drama, Science Fiction, Thriller	2	2	1/2	3/4	Fight Club	Drama	2	2	1/2	1
		Science Fiction	Fight Club	Drama					The Day the Earth Stood Still	Drama, Science Fiction, Thriller				
			Harry Potter and the Order of the Phe	Adventure, Fantasy, Mystery					Minority Report	Action, Thriller, Science Fiction, Mystery				
			Santa Claus: The Movie	Family, Fantasy, Adventure					Surrogates	Science Fiction, Action, Thriller				

Figure 7.2. 1 Results of recommendation approaches

The results obtained for two approaches are shown in the above figure 7.1. The evaluation matrics precision and novelty are calculated based on the results.

Figure 7.2.1 shows the bar graph comparing the precision of the two tested approaches, namely, multi model data integration-based movie recommendation with genetic algorithm-based optimization and multi model data integration based movie recommendation with genetic algorithm based optimization.

	Multi model data integartion based recommendation with genetic algorithm based optimization	Multi model data integartion based recommendation without genetic algorithm based optimization
Precision	0.65	0.625

Figure 7.2. 2 Precision obtained for recommendation approaches

As shown in the figure 7.2.2 Precision values obtained for the first approach and second approach are 0.65 and 0.625. Therefore, genetic algorithm optimization and multi model data integration-based movie recommender system has the high precision value compared to the other approach. Figure 7.2.3 illustrates the precision values obtained in two approaches in a bar chart.

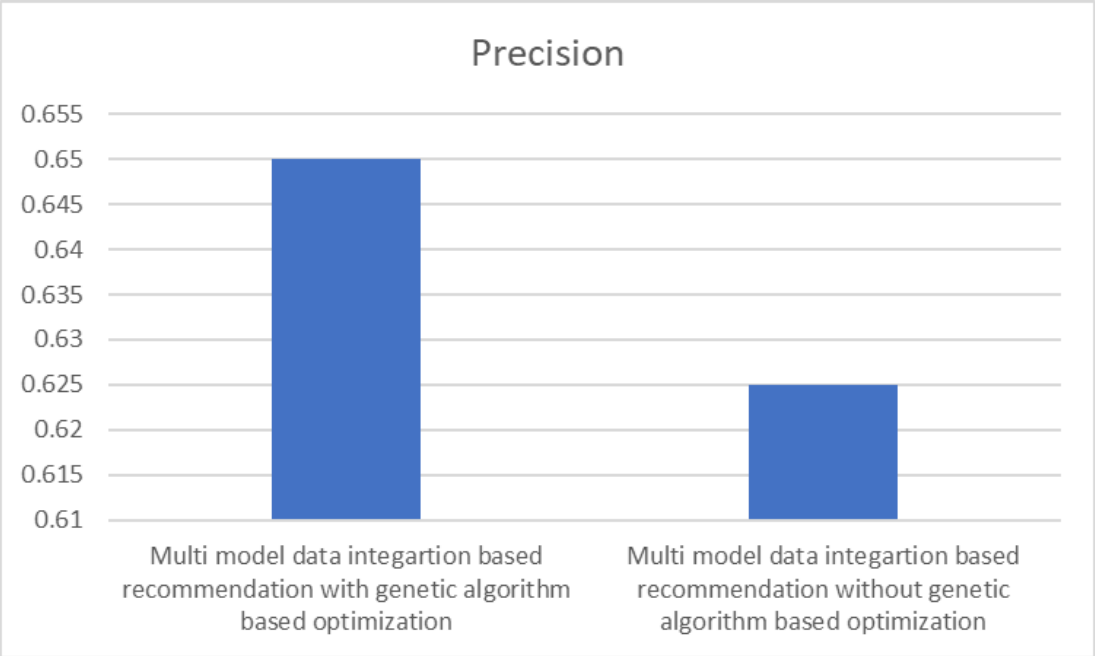


Figure 7.2. 3 Calculated precision for experiments

Figure 7.2.4 shows the calculated results of the metric degree of novelty per movie for two experiments conducted. Figure 7.2.5 shows the bar graph comparing the degree of novelty obtained for tested movies for the two tested approaches, namely, multi model data integration-based movie recommendation with genetic algorithm based

optimization and multi model data integration based movie recommendation with genetic algorithm based optimization.

Move Name	Degree of novelty	
	Multi model data integartion based recommendation with genetic algorithm based optimization	Multi model data integartion based recommendation without genetic algorithm based optimization
Garfield: A Tail of Two Kitties	0.75	0.25
Thor	0.75	0.5
Interstellar	0.75	0.75
Inception	0.75	0.75
The Amazing Spider-Man 2	0.75	0.75
Toy Story	0.25	0.25
Apollo 13	1	1
Jurassic Park	0.25	0.5
Wonder Woman	0.5	0.5
The Avengers	0.75	1

Figure 7.2. 4 Calculated Degree of novelty per movie

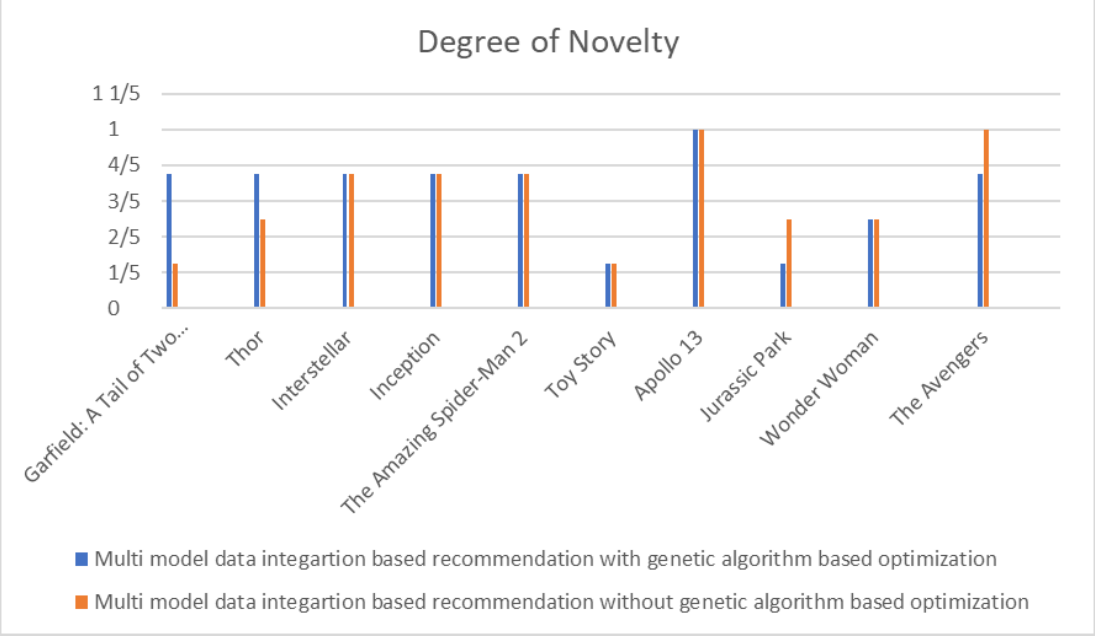


Figure 7.2. 5 Calculated degree of novelty for experiments

Following results show the results obtained from the third experiment which was conducted to compare the results obtained from the Google recommender system with the results obtained from the implemented movie recommender system.

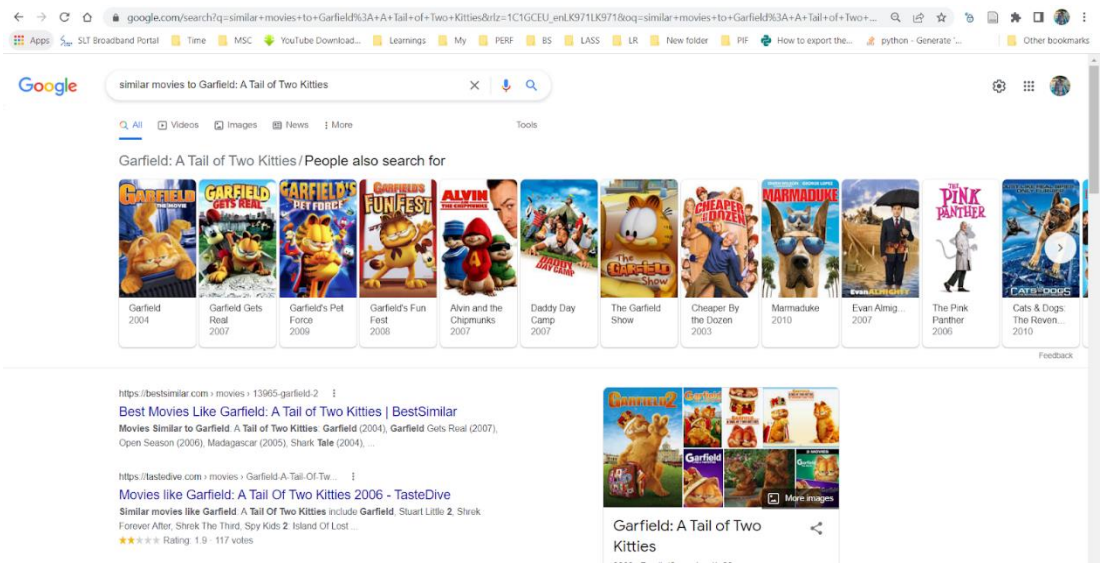


Figure 7.2. 6 Google recommendations for movie Garfield: A Tail of Two Kitties

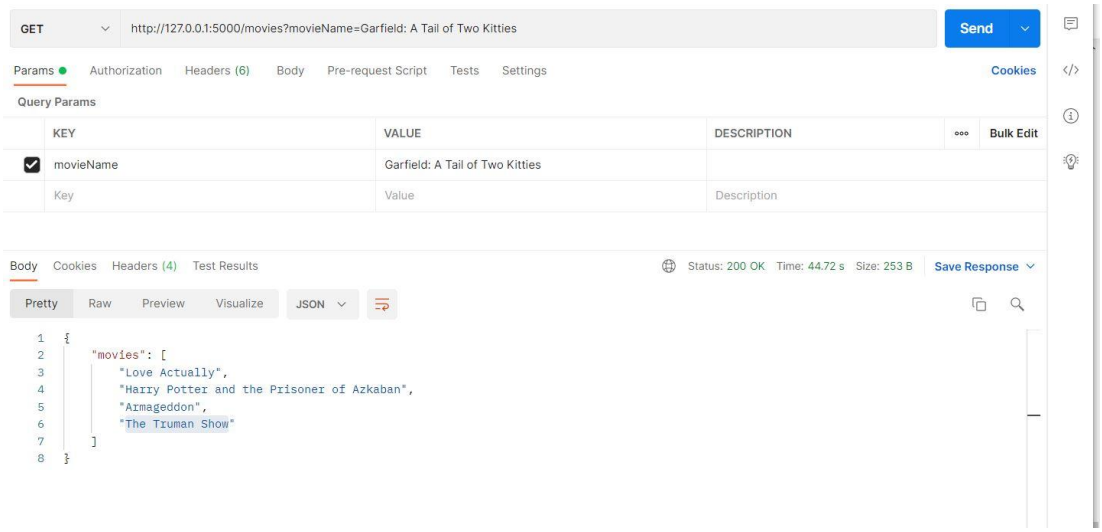


Figure 7.2. 7 Recommended movies to movie Garfield: A Tail of Two Kitties

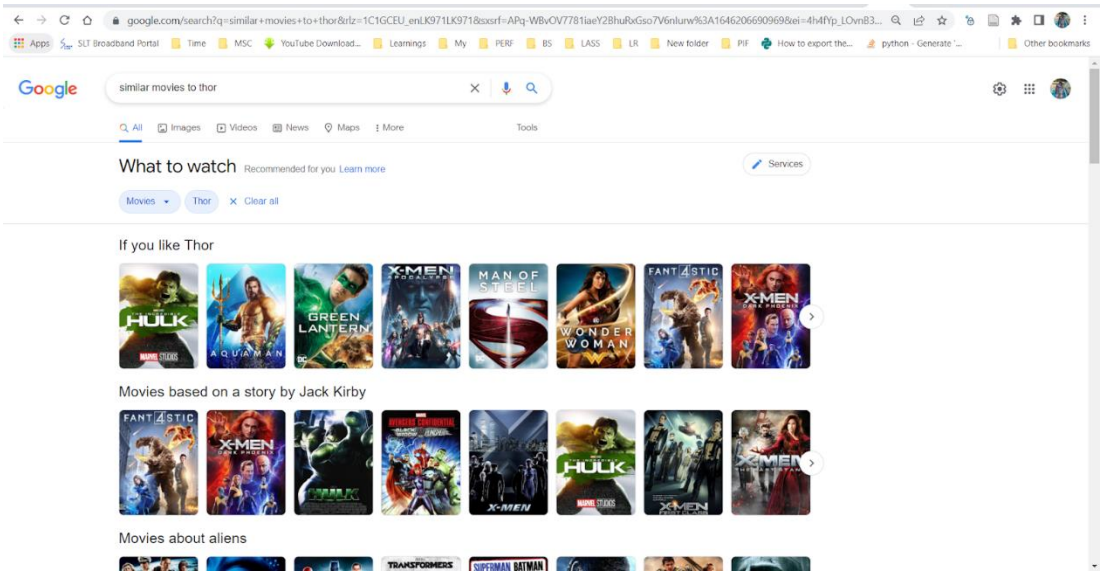


Figure 7.2. 8 Google recommendations for movie Thor

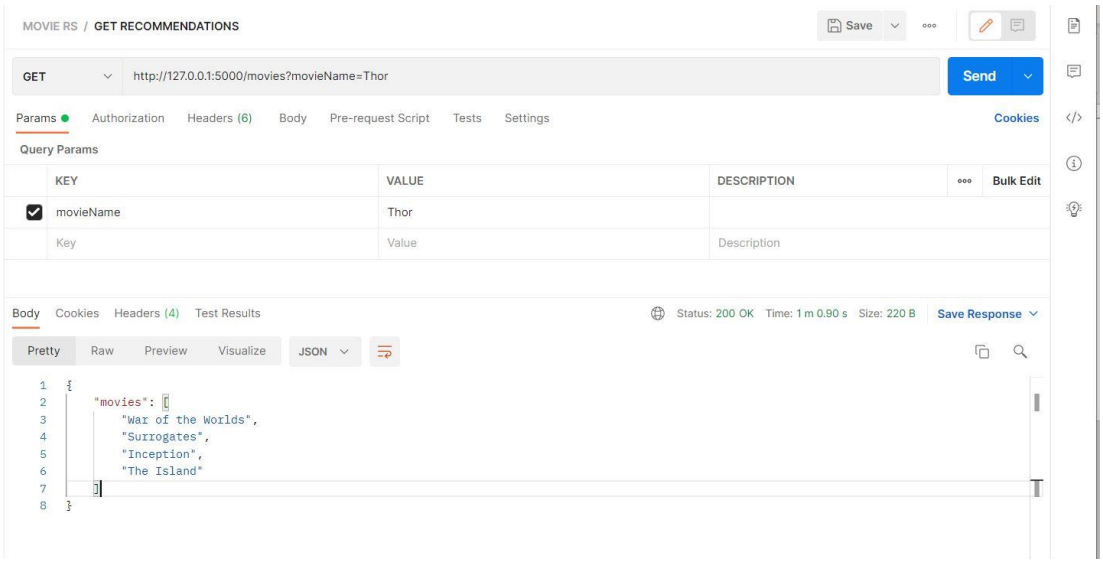


Figure 7.2. 9 Recommended movies to movie Thor

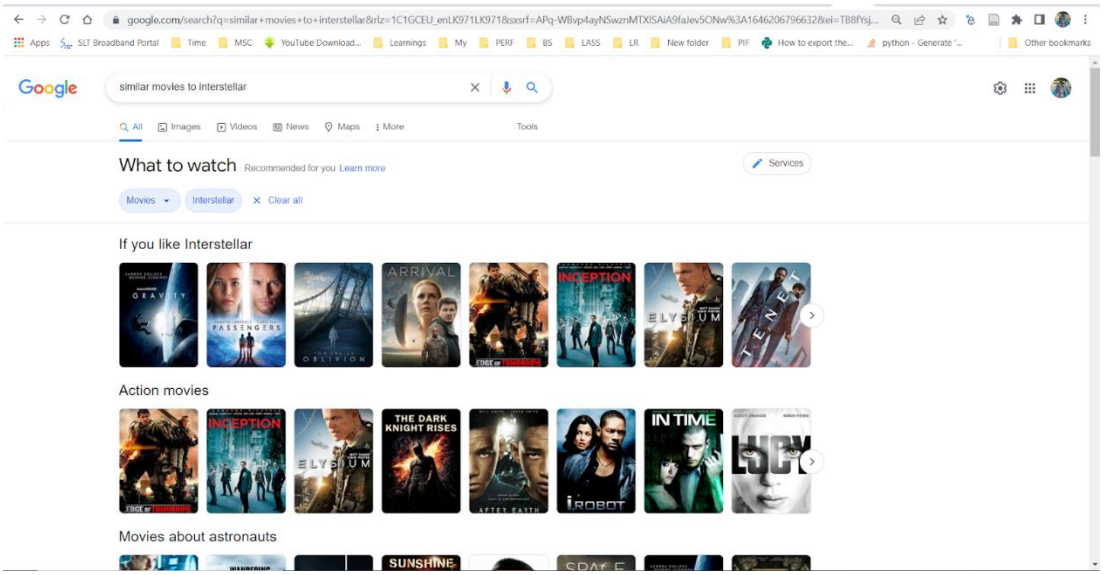


Figure 7.2. 10 Google recommendations for movie Interstellar

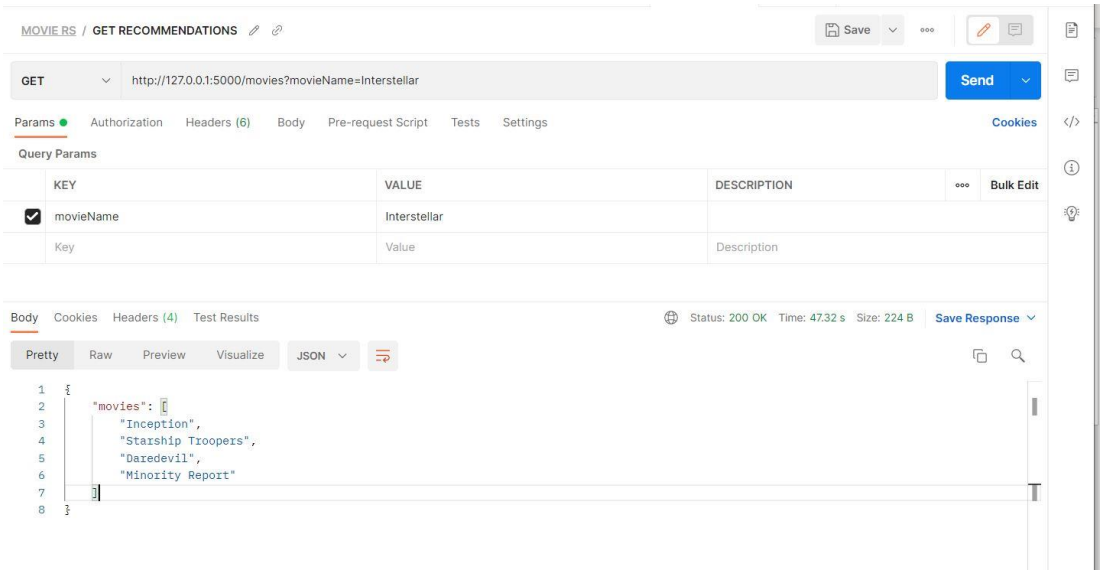


Figure 7.2. 11 Recommended movies to movie Interstellar

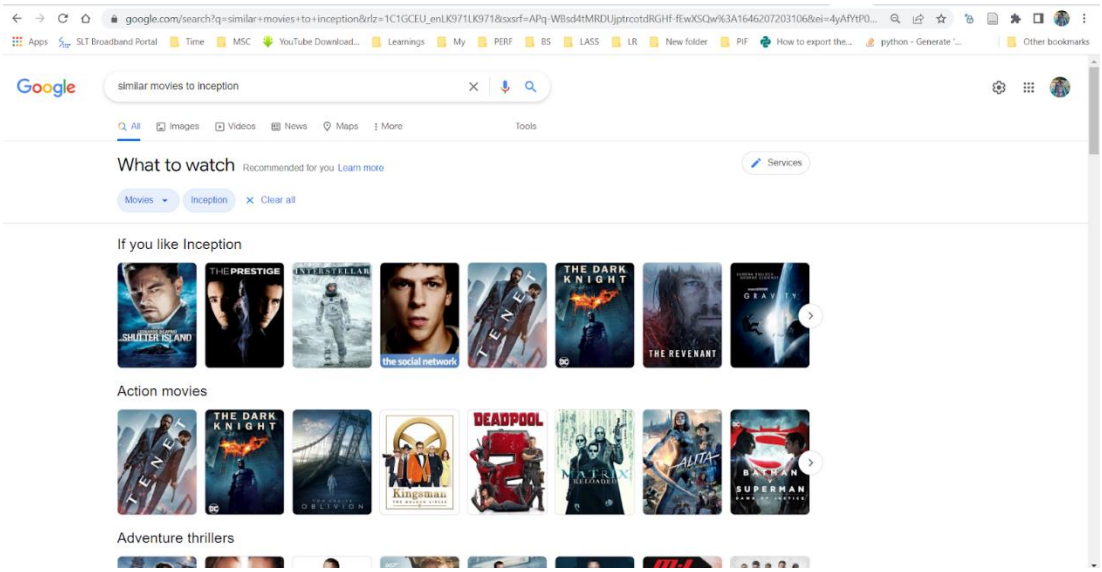


Figure 7.2. 12 Google recommendations for movie Inception

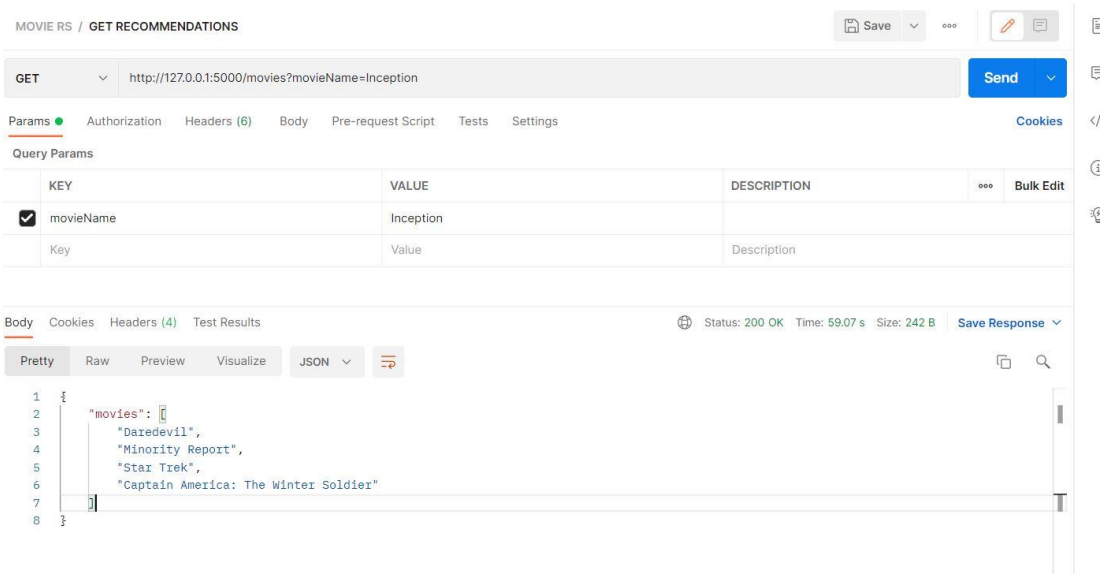


Figure 7.2. 13 Recommended movies to movie Inception

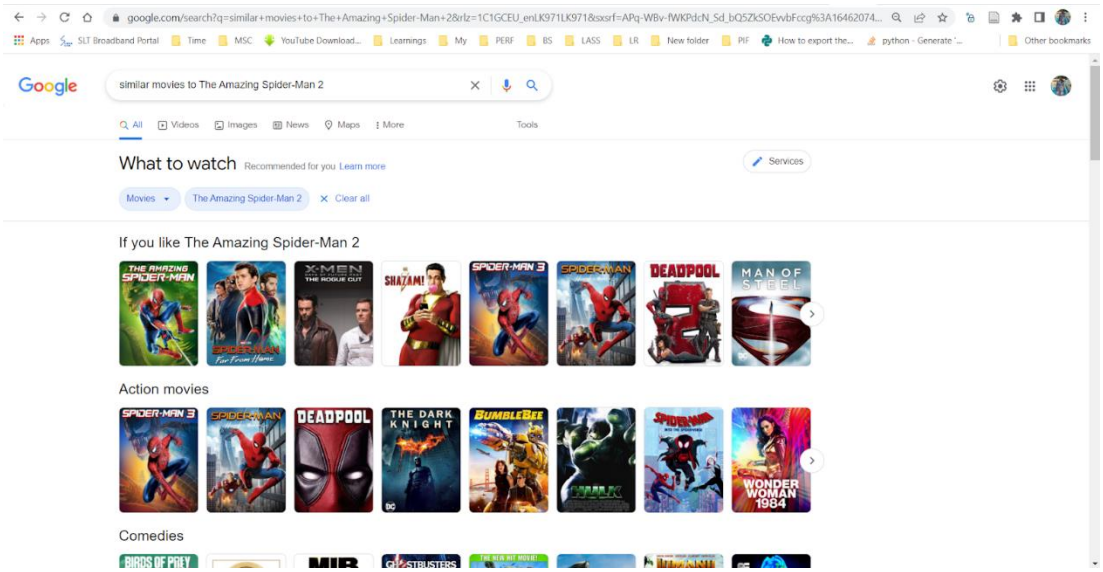


Figure 7.2. 14 Google recommendations for movie The Amazing Spider-Man 2

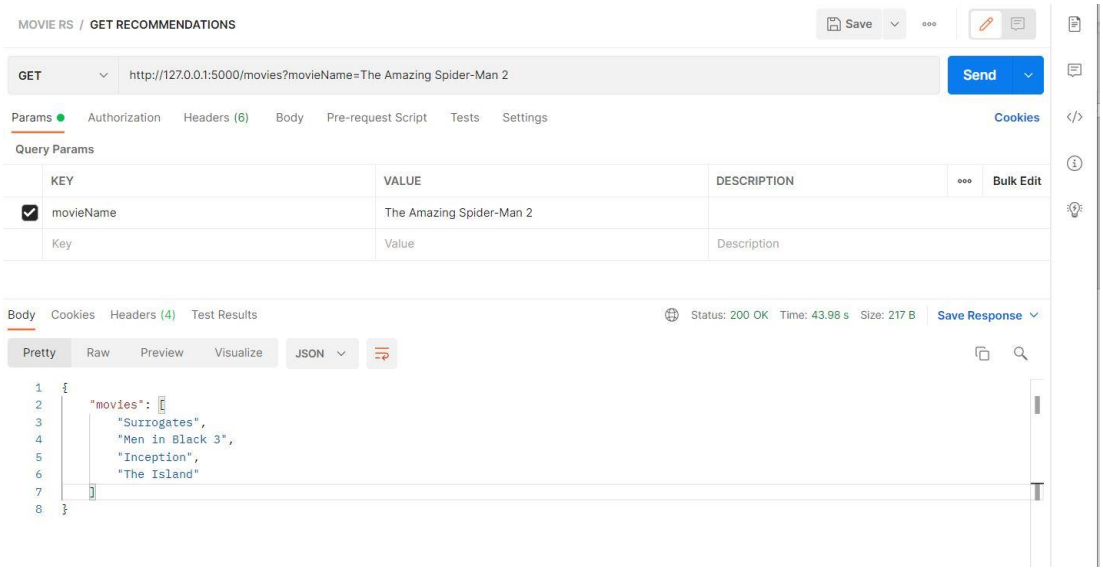


Figure 7.2. 15 Recommended movies to movie The Amazing Spider-Man 2

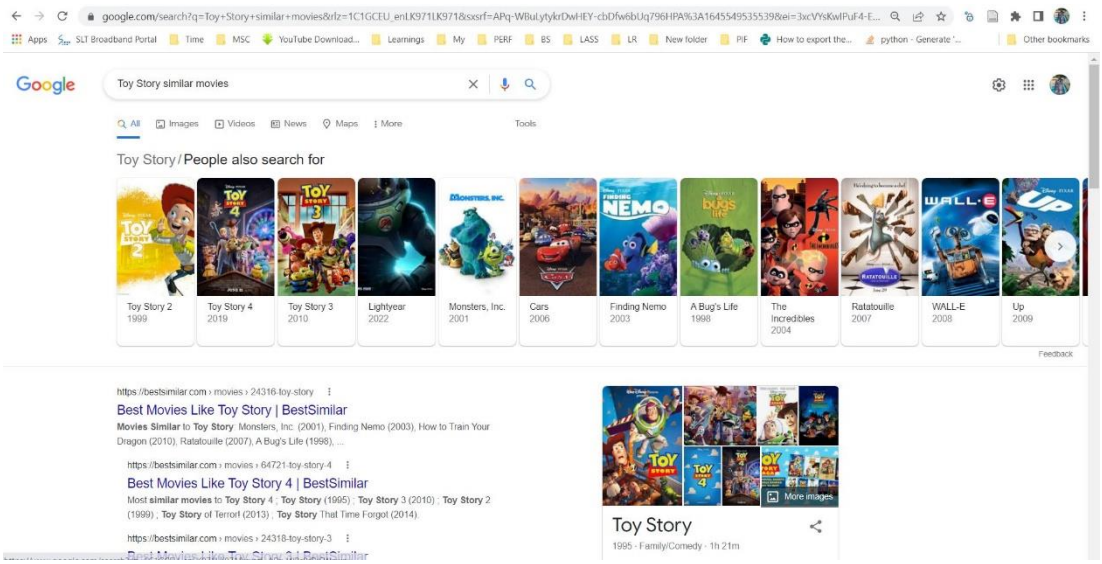


Figure 7.2. 16 Google recommendations for movie Toy Story

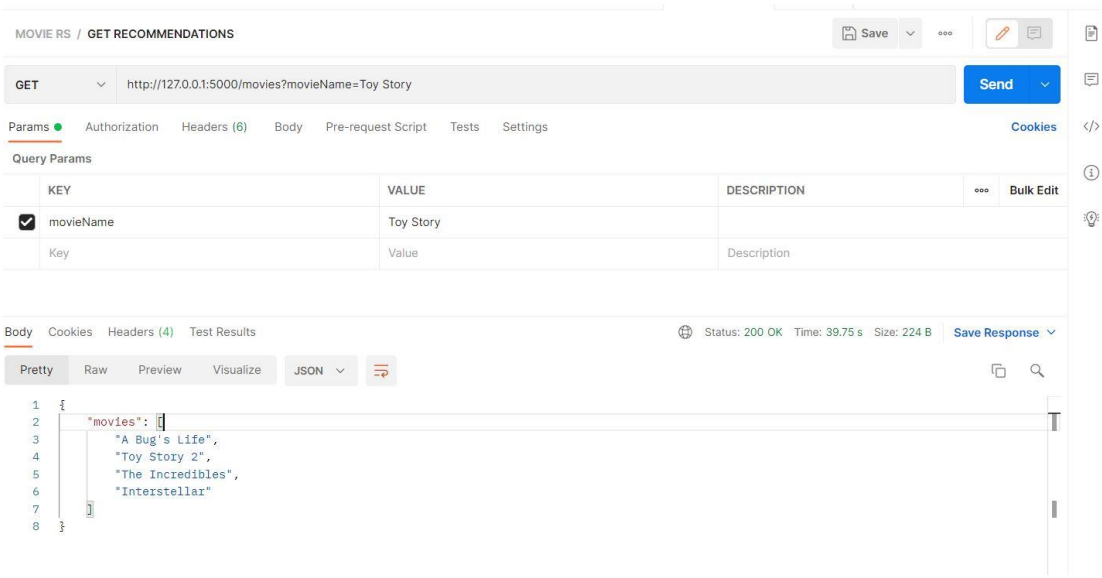


Figure 7.2. 17 Recommended movies to Toy story

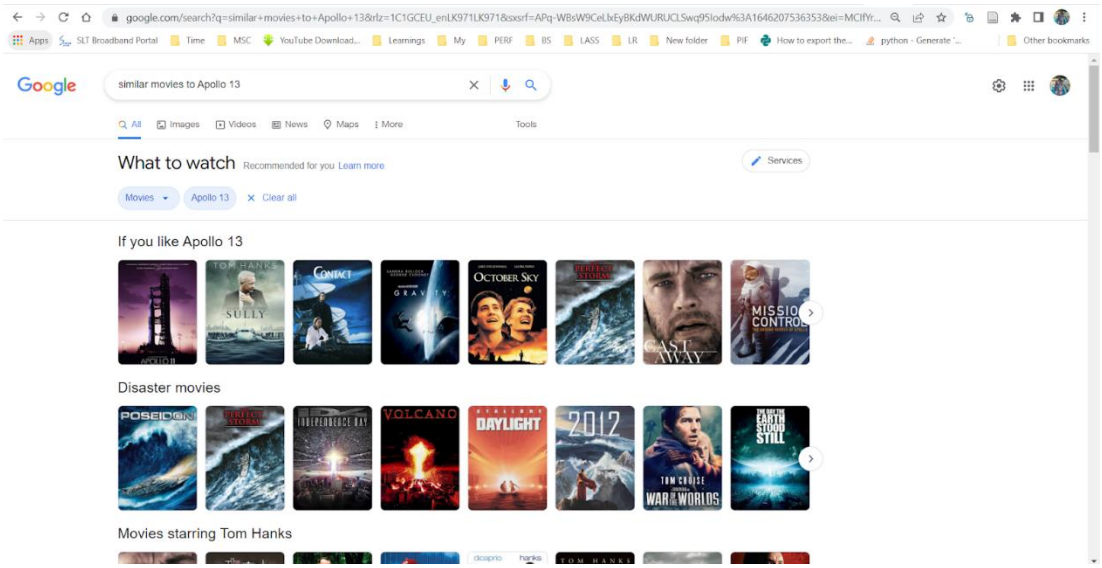


Figure 7.2. 18 Google recommendations for movie Apollo 13

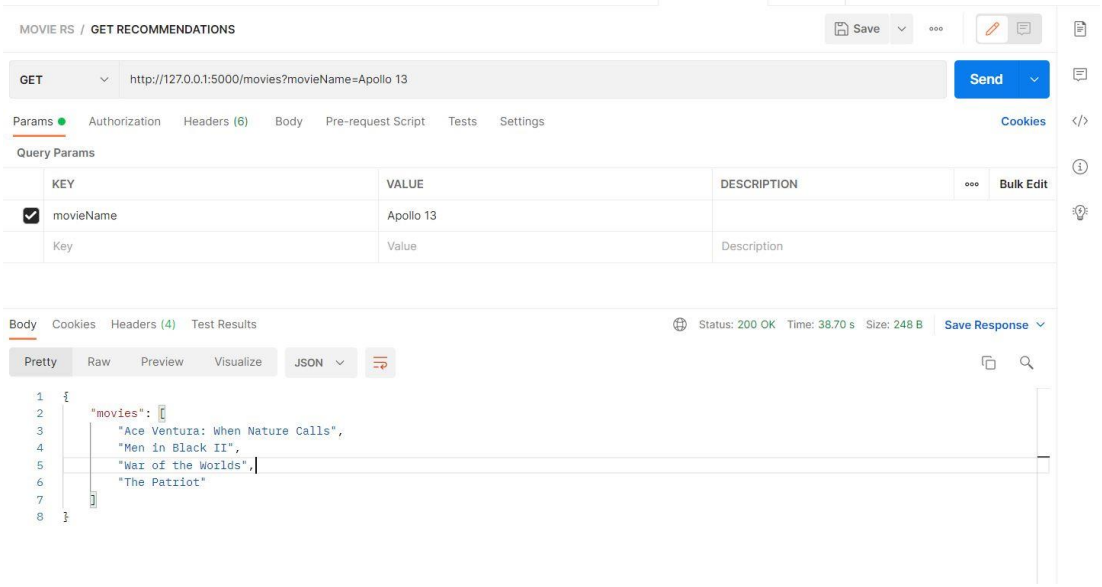


Figure 7.2. 19 Recommended movies to movie Apollo 13

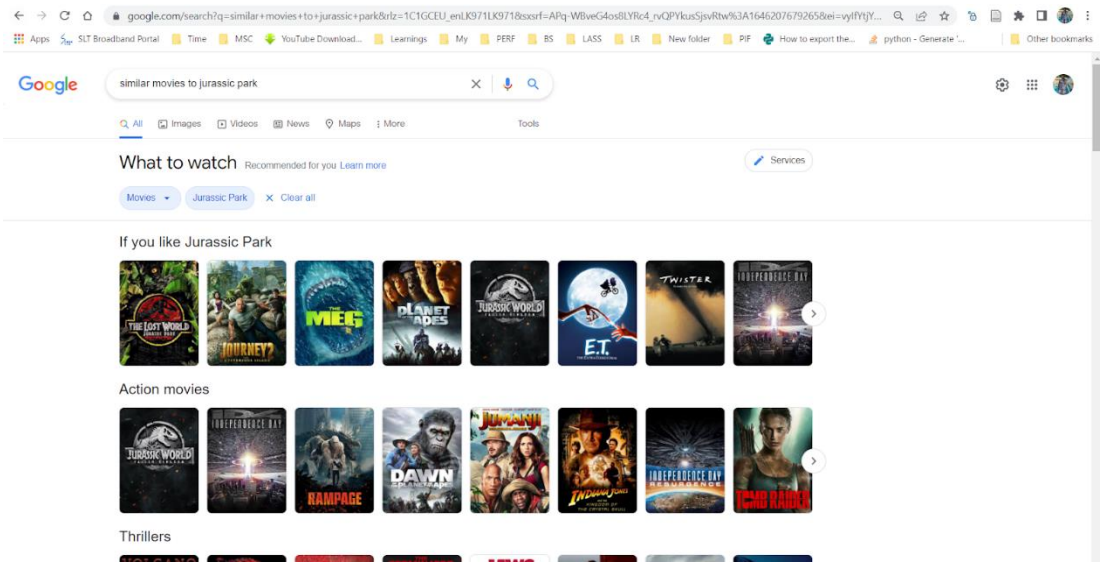


Figure 7.2.20: Google recommendations for movie Jurassic Park

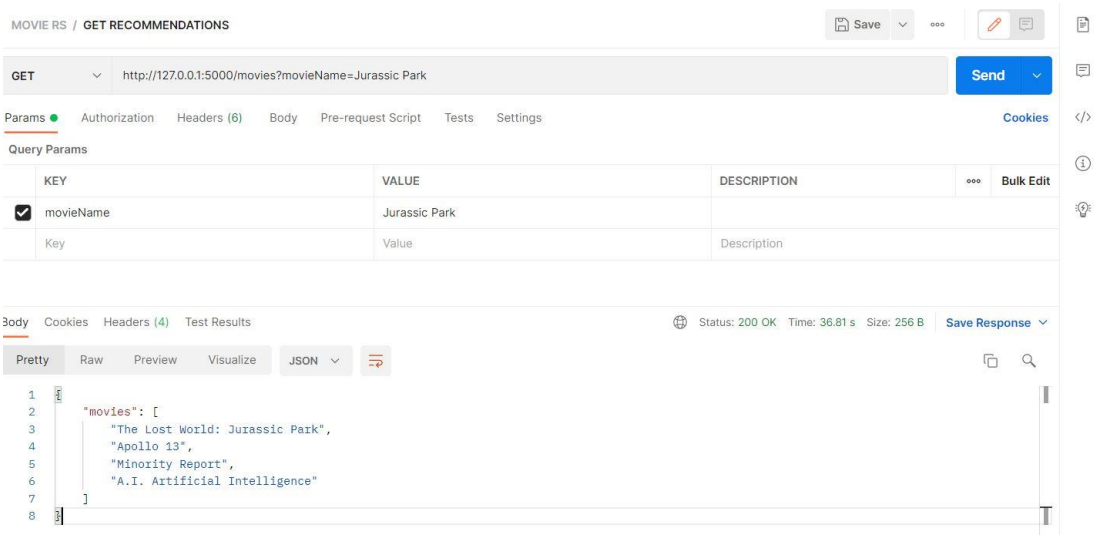


Figure 7.2.21: Recommended movies to movie Jurassic Park

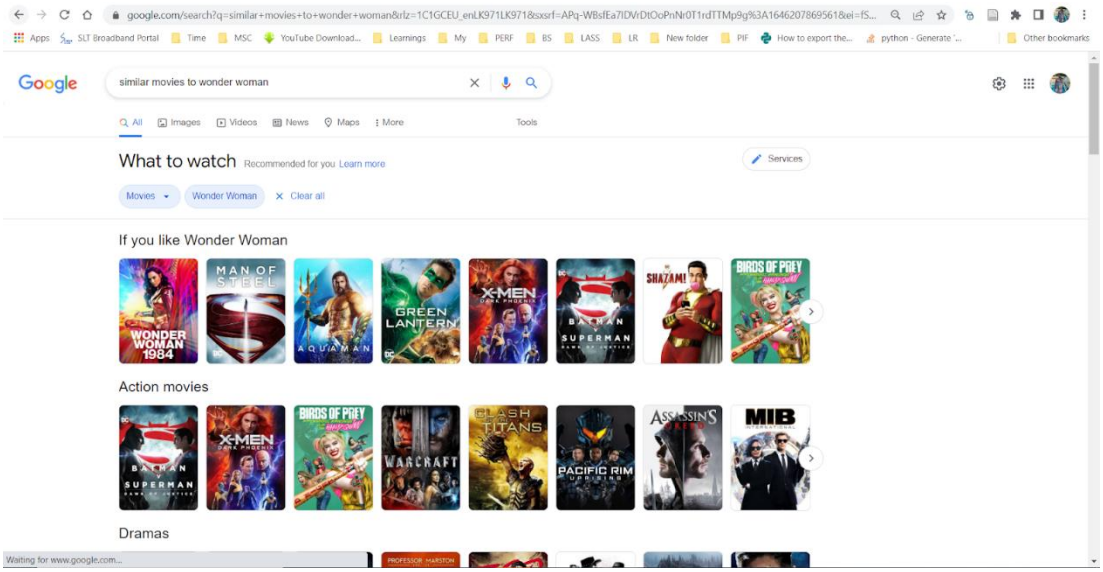


Figure 7.2.22: Google recommendations for movie Wonder Woman

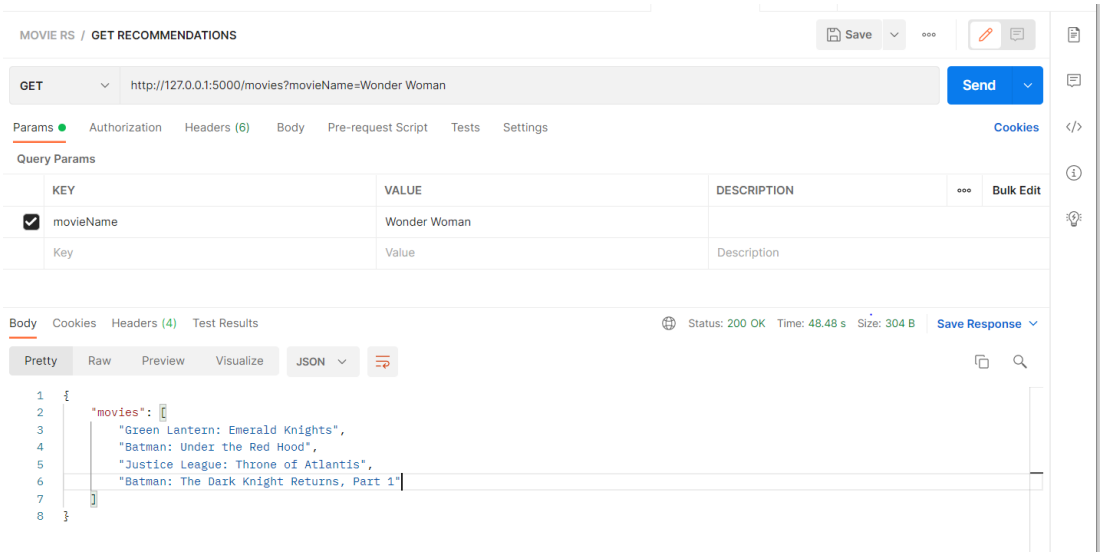


Figure 7.2.23: Recommended movies to movie Wonder Woman

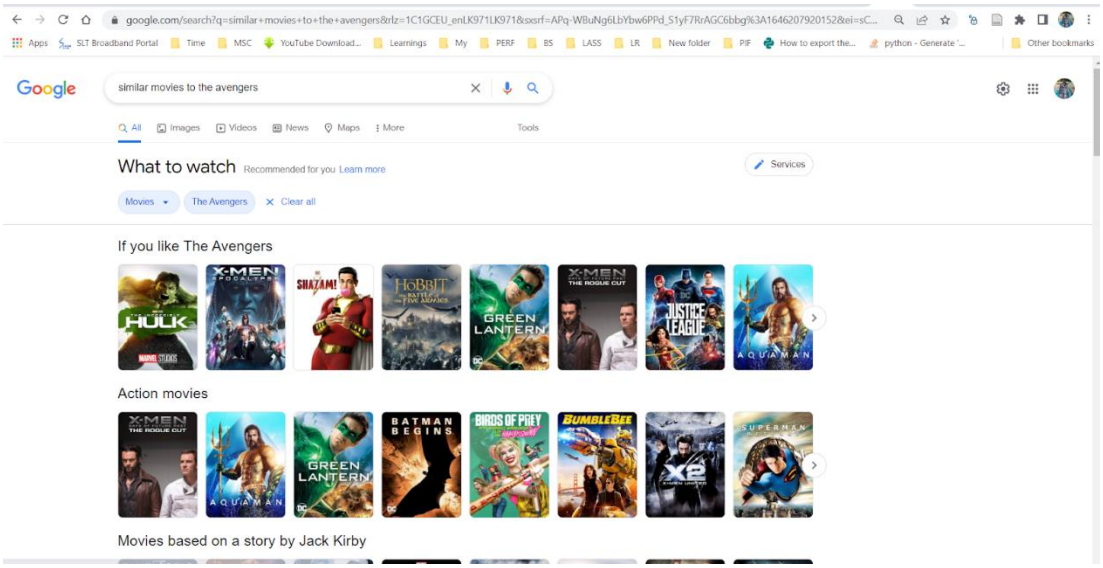


Figure 7.2.24: Google recommendations for movie The Avengers

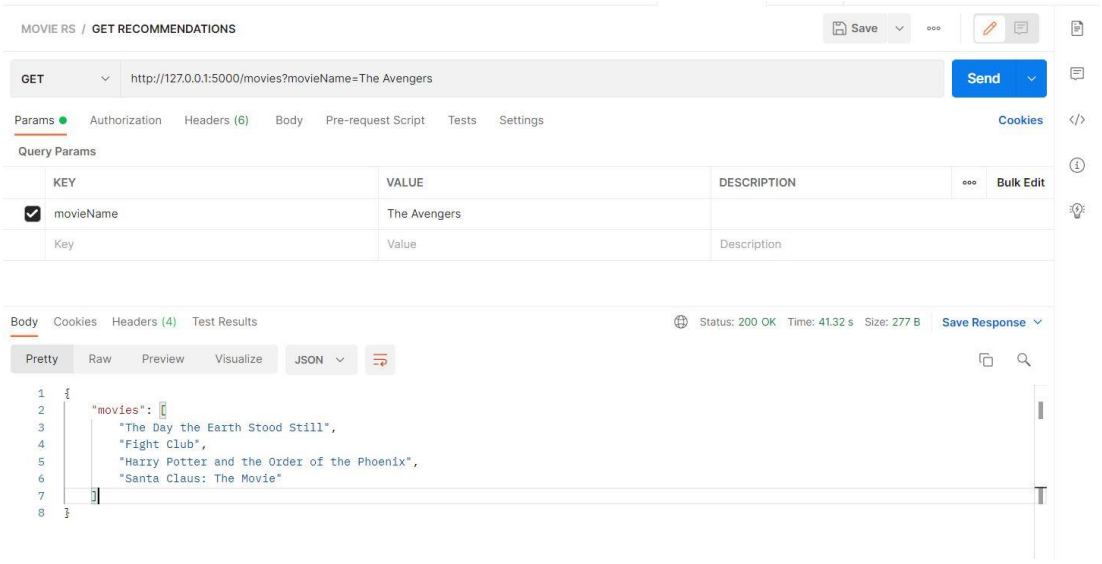


Figure 7.2.25: Recommended movies to movie The Avengers

It's evident that for 40% of experiments conducted, at least one recommended movie in the multi-model data integration based recommendations is in the top ten movie recommendations obtained from the google recommendations.

7.3 Summary

Based on the design and implementation details, the quality of the movie recommendations can be improved by using multi model data integration. This chapter presents the results and evaluation details on compared approaches which used to generate personalised movie recommendations.

CHAPTER 8

CONCLUSION AND FUTURE WORK

8.1 Introduction

According to the previous chapters, multi model data integration and evolutionary computing-based movie recommender system was implemented. At the beginning a literature review was conducted, focusing on the data pre-processing, review analysis, profiling, content-based recommendation, and evolutionary computing which are the main five modules in the project. In addition to that, different approaches that can be used for the implementation are discussed in the previous chapters. This chapter describes the conclusion and further work which can be conducted to improve the quality of the movie recommendations.

Due to the information overload problem, recommender systems have become increasingly useful when generating recommendations. In this research attempt is to improve the quality of the recommendations through multi-model data integration and introduction of genetic algorithm approach to optimize the recommendation list and introduce new items to the user. The movie Lens dataset was used to implement and generate mappings required for the multi-model data integration-based movie recommender system. This study was conducted on a medium-sized dataset, where fast growing datasets are involved in a realistic situation. The other limitation is usage of only the movie's genre function on optimizing the recommendation list. In future more features should be identified and tested with the genetic algorithm approach to optimize the recommendations.

8.9 Summary

This chapter presented the conclusion and future work on multi model data integration and evolutionary computing based personalised movie recommender system.

REFERENCES

[1] Y. Wang, M. Wang, and W. Xu, “A Sentiment-Enhanced Hybrid Recommender System for Movie Recommendation: A Big Data Analytics Framework,” *Wirel. Commun. Mob. Comput.*, vol. 2018, pp. 1–9, 2018, doi: 10.1155/2018/8263704.

[2] M. M. R. Siddiquee, N. Haider, and R. M. Rahman, “A fuzzy based recommendation system with collaborative filtering,” in *The 8th International Conference on Software, Knowledge, Information Management and Applications (SKIMA 2014)*, Dhaka, Bangladesh, Dec. 2014, pp. 1–8. doi: 10.1109/SKIMA.2014.7083524.

[3] N. Raval and V. Khedkar, “A Review Paper On Collaborative Filtering Based Moive Recommedation System,” vol. 8, no. 12, p. 6, 2019.

[4] Nagamanjula, R. A Novel Scheme for Movie Recommendation System using User Similarity and Opinion Mining International Journal of Innovative Technology and Exploring Engineering (IJITEE) ISSN: 2278-3075, Volume-8 Issue-4S2 March, 2019

[5] Agarwal, A. Srinivasan, S. (2020). International Journal of Engineering and Advanced Research Technology (IJEART). International Journal of Engineering and Advanced Research Technology (IJEART), 7(7).

[6] F. Furtado and A. Singh, “Movie Recommendation System using Machine Learning,” *Int. J. Res. Ind. Eng.*, no. Online First, Apr. 2020, doi: 10.22105/riej.2020.226178.1128.

[7] R. Sharma and R. Singh, “Evolution of Recommender Systems from Ancient Times to Modern Era: A Survey,” *Indian J. Sci. Technol.*, vol. 9, no. 20, May 2016, doi: 10.17485/ijst/2016/v9i20/88005.

[8] M. F. Aljunid and M. Doddaghatsta Huchaiah, “Multi-model deep learning approach for collaborative filtering recommendation system,” *CAAI Trans. Intell. Technol.*, vol. 5, no. 4, pp. 268–275, Dec. 2020, doi: 10.1049/trit.2020.0031.

[9] R. Kumar, G. Basava, and F. Furtado, “An Efficient Content, Collaborative – Based and Hybrid Approach for Movie Recommendation Engine,” p. 11, 2020.

[10] A. Kashyap, “A Movie Recommender System: MOVREC using Machine Learning Techniques,” p. 6, 2020.

[11] G. Vibhandik *et al.*, “Movie Recommendation System Using Machine Learning,” vol. 8, no. 11, p. 3, 2020.

[12] A. M. Damanhuri and Z. Huaping, “DESIGN OF PEOPLE PROFILING AND MODELING REPUTATION COMPUTATION BASED ON SENTIMENT ANALYSIS,” *SINERGI*, vol. 23, no. 1, p. 1, Feb. 2019, doi: 10.22441/sinergi.2019.1.001.

[13] Nagamanjula, R. A Novel Scheme for Movie Recommendation System using User Similarity and Opinion Mining International Journal of Innovative Technology and Exploring Engineering (IJITEE) ISSN: 2278-3075, Volume-8 Issue-4S2 March, 2019

[14] Agarwal, A. Srinivasan, S. (2020). International Journal of Engineering and Advanced Research Technology (IJEART). International Journal of Engineering and Advanced Research Technology (IJEART), 7(7).

[15] A. T. Ho, I. L. L. Menezes, and Y. Tagmouti, “E-MRS: Emotion-based Movie Recommender System,” p. 8.

[16] V. Babanne, M. Borgaonkar, M. Katta, P. Kudale, and V. Deshpande, “EMOTION based PERSONALIZED RECOMMENDATION SYSTEM,” vol. 07, no. 08, p. 5, 2020.

[17] A. Surendran, A. K. Yadav, and A. Kumar, “Movie Recommendation System using Machine Learning Algorithms,” vol. 07, no. 04, p. 3, 2020.

[18] S. Kanoje, S. Girase, and D. Mukhopadhyay, “User Profiling for Recommendation System,” p. 6.

[19] S. S. Choudhury, S. N. Mohanty, and A. K. Jagadev, “Multimodal trust based recommender system with machine learning approaches for movie recommendation,” *Int. J. Inf. Technol.*, vol. 13, no. 2, pp. 475–482, Apr. 2021, doi: 10.1007/s41870-020-00553-2.

[20] C. Ju and C. Xu, “A New Collaborative Recommendation Approach Based on Users Clustering Using Artificial Bee Colony Algorithm,” *Sci. World J.*, vol. 2013, pp. 1–9, 2013, doi: 10.1155/2013/869658.

[21] Sharma, M. and Mann, S. (no date) ‘A Survey of Recommender Systems: Approaches and Limitations’, International Journal of Innovations in Engineering and Technology, p. 9.

[22] Z. Ji, C. Yang, H. Wang, and J. E. A.-I. Igo, “BRScS: A Hybrid Recommendation Model Fusing Multi-source Heterogeneous Data,” p. 9.

[23] T. Anwar and V. Uma, “Comparative study of recommender system approaches and movie recommendation using collaborative filtering,” *Int. J. Syst. Assur. Eng. Manag.*, vol. 12, no. 3, pp. 426–436, Jun. 2021, doi: 10.1007/s13198-021-01087-x.

[24] A. Wroblewska *et al.*, “Multi-modal Embedding Fusion-based Recommender,” *ArXiv200506331 Cs*, May 2020, Accessed: Nov. 15, 2021. [Online]. Available: http://arxiv.org/abs/2005.06331

[25] O. Stitini, S. Kaloun, and O. Bencharef, “An Improved Recommender System Solution to Mitigate the Over-Specialization Problem Using Genetic Algorithms,” *Electronics*, vol. 11, no. 2, p. 242, Jan. 2022, doi: 10.3390/electronics11020242.

[26] S. Valero and V. Botti, “Capturing User’s Preferences using a Genetic Algorithm - Determining Essential and Dispensable Item Attributes,” in ICAART 2010 - Proceedings of the International Conference on Agents and Artificial Intelligence, Volume 1 - Artificial Intelligence, Valencia, Spain, January 22-24, Jan. 2010.

[27] M. Tkalcˇicˇ, A. Ko, and J. Tasicˇ, “Affective recommender systems: the role of emotions in recommender systems,” p. 6.

[28] F. Mansur, V. Patel, and M. Patel, “A review on recommender systems,” in *2017 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS)*, Coimbatore, Mar. 2017, pp. 1–6. doi: 10.1109/ICIIECS.2017.8276182.

[29] F. O. Isinkaye, Y. O. Folajimi, and B. A. Ojokoh, “Recommendation systems: Principles, methods and evaluation,” *Egypt. Inform. J.*, vol. 16, no. 3, pp. 261–273, Nov. 2015, doi: 10.1016/j.eij.2015.06.005.

[30] S. Reddy, S. Nalluri, S. Kunisetti, S. Ashok, and B. Venkatesh, “Content-Based Movie Recommendation System Using Genre Correlation,” in *Smart Intelligent Computing and Applications*, vol. 105, S. C. Satapathy, V. Bhateja, and S. Das, Eds. Singapore: Springer Singapore, 2019, pp. 391–397. doi: 10.1007/978-981-13-1927-3_42.

[31] A. Dahiya and S. Sangwan, “Literature Review on Genetic Algorithm,” vol. 05, no. 16, p. 6, 2018.

[32] S. Sharma and V. K. Dwivedi, “Genetic algorithm: review and applications,” p. 7.

[33] M. Chenna Keshava, P. Narendra Reddy, S. Srinivasulu, B. Dinesh Naik, and JNTUA College of Engineering, Pulivendula, “Machine Learning Model for Movie Recommendation System,” IJERT, vol. V9, no. 04, p. IJERTV9IS040741, May 2020, doi: 10.17577/IJERTV9IS040741.

APPENDIX

Appendix A

Questionnaire to identify user interests and behaviors towards movies

Hi, I am a postgraduate of University of Moratuwa following a masters in Artificial Intelligence at the Department of Computational Mathematics. This research is carried out to generate personalized movie recommendations by multi model data integration which is my final year research project. "Questionnaire to identify user interests and behaviors towards movies" has been designed to collect data to identify the user interests required for generating movie recommendations.

Please note that I will protect the anonymity and confidentiality of your personal data and the data will only be used for the analysis of in this study and future studies arising from this topic.

Take few minutes of your time and fill out the form below with your responses.

* Required

1. What is your name? *

2. What is your gender? *

Mark only one oval.

- ◯ Male
- ◯ Female

3. What is your age category? *

Mark only one oval.

- 10-20
- 20-30
- 30-40
- 40-50
- 50-60
- above 60

4. How much do you like watching movies? *

Mark only one oval.

- I really like watching movies.
- I like watching movies.
- I neither like nor dislike watching movies.
- I dislike watching movies.
- I really dislike watching movies.

5. Which do you prefer? *

Mark only one oval.

- Short films
- Feature Films

6. Do you like genre hybrids (two or more genres combined)? *

Mark only one oval.

- ◯ Yes
- ◯ No

7. Which way do you prefer to watch movies? *

Check all that apply.

- ☐ On TV
- ☐ Online
- ☐ Go to cinema
- ☐ On DVD
- Other: ☐ _____

8. At what time do you prefer to watch movies? *

Mark only one oval.

- ◯ 11.00 AM - 3.00 PM
- ◯ 3.00 PM - 7.00 PM
- ◯ 7.00 PM - 11.00 PM
- ◯ 11.00 PM - 3.00 AM

9. With whom do you watch movies with? *

Check all that apply.

- ☐ Friends
- ☐ Relatives
- ☐ Lover
- ☐ Alone
- ☐ Pet

10. What criteria you focus on when choosing a movie? *

11. How often do you watch movies? *

Mark only one oval per row.

	Never	Once a Year or Less	Several Times a Year	Once a Month	2-3 Times a Month	Once a Week	2-3 Times a Week	Daily
In theaters								
Rent (pay per individual movie)								
Paid movie subscription (Netflix, etc.)								
Personal library (movies you own)								
Free online streaming (Hulu, etc.)								

12. How much do you like watching movies from the following genres? *

Mark only one oval per row.

	Really dislike	Dislike	Neither like nor dislike	Like	Really like	Not sure of genre definition
Action	◯	◯	◯	◯	◯	◯
Adventure	◯	◯	◯	◯	◯	◯
Animation	◯	◯	◯	◯	◯	◯
Children	◯	◯	◯	◯	◯	◯
Comedy	◯	◯	◯	◯	◯	◯
Crime	◯	◯	◯	◯	◯	◯
Documentary	◯	◯	◯	◯	◯	◯
Drama	◯	◯	◯	◯	◯	◯
Fantasy	◯	◯	◯	◯	◯	◯
Horror	◯	◯	◯	◯	◯	◯
Musical	◯	◯	◯	◯	◯	◯
Mystery	◯	◯	◯	◯	◯	◯
Sci-Fi	◯	◯	◯	◯	◯	◯
Thriller	◯	◯	◯	◯	◯	◯
War	◯	◯	◯	◯	◯	◯
Western	◯	◯	◯	◯	◯	◯

15. Do you consider the running time when choosing a movie? *

Mark only one oval.

- ◯ Yes
- ◯ No

16. How do you get recommendations when choosing a movie? *

Mark only one oval.

- ◯ From friends
- ◯ From recommendation websites
- ◯ Other: _____

17. Name a few of your favorite movies *

_____

_____

_____

_____

_____

18. When choosing a movie, do you like to watch a movie from the current trending list? *

Mark only one oval.

- ◯ Yes
- ◯ No

Figure 46: Questionnaire to identify user interests