Recommender System

IIM2016003 Neeraj Mishra
ITM2016004 Avinash Yadav
IRM2016002 Ankit Kumar
ISM2016006 Chirag Meel

Table of Contents

1. Introduction

1.1 Purpose
1.2 Scope
1.3 Document Conventions
1.4 Intended Audience and Reading Suggestions
1.5 Technology Used
1.6 Overview
1.7 References

2. Overall Description

2.1 Use case Diagram
2.2 Product Perspective
2.3 Assumptions and Dependencies

3. Specific Requirements

3.1 Functionality
3.2 Supplementary Requirements

4. External Interface Requirements

4.1 User Interfaces
4.2 Hardware Interfaces
4.3 Software Interfaces
4.4 Communications Interfaces

5. Non-functional Requirements

5.1 Safety Requirements
5.2 Security Requirements
5.3 Software Quality Attributes

1. Introduction

1.1 Purpose

The main purpose of the system is to provide for a recommender system incorporating collaborative and content based filtering and implementing the same on the Movie Lens dataset to provide for a movie recommender system to the users of the system

1.2 Scope

- ✓ Provide for personalized watch list dashboard to the user.
- ✓ Provide for login facility to old users and signup facility to new users.
- ✓ Provide new user with a basic cold-start setup after considering his/her choice of movies.
- ✓ Equip users with the capability of providing rating to movies.
- ✓ Provide users with recommendation after item based filtering depending on the selected movie.
- ✓ Provide for a mechanism to provide the user with the calculated rating of any selected movie of user's choice.
- ✓ Allowing users to view recommendations based on ratings and watch list of other users with similar tastes.
- ✓ Allowing users to modify old ratings as per new acquired taste.

1.3 Document Conventions

RDD – Resilient Distributed Dataset

JSON- JavaScript Object Notation

HTML-Hyper Text Markup Language

CSS – Cascading Style Sheets

1.4 Intended Audience and Reading Suggestions

The intended audiences of the document are project managers, developers, testers and end users. The Software Requirements Specification (SRS) document is intended to provide the requirements of the Recommender System project and the expectations of the stakeholders. The document includes the project perspective, data model and constraints of the overall system.

1.5 Technologies used

Software Frontend	-	HTML, CSS, JavaScript, Flask
Software Backend	-	Spark
Development tool	-	PyCharm
Database	-	MYSQL
Designing tool	-	Star UML

1.6 Overview

It is a basic system to recommend top movies based on the calculated scores using a weighted rating formula. A weighted rating system that takes into account both the number of voters as well as the respective ratings of a movie provides for an ideal recommender system. We could have simply used movie rating as our recommendation metric. However this has a few caveats.

1) It doesn't take the popularity of a movie into consideration. For example, a recently released local movie with a relatively small size (e.g. 30) of audience and voters (who voted this movie 9.5) would be considered better than a movie with a rating of 8.5 with voters over 100,000.

2) It will in general always prefer movies with extremely high ratings and will not wait for the ratings to regularize after a sufficient viewership (votes) have been achieved.

The system provides the user with the ability to rate new movies, change the rating of any previously viewed movie and the facility of viewing a list of recommendations of movies according to various filtering algorithms as per user's choice. On selecting a particular movie its calculated rating is computed and displayed along with a list of movies recommended for the user showing close resemblance to the particular movie.

The user can also view a recommended list of movies based on preferences and ratings of other users with similar taste in movies.

1.7 References

[1] Greg Linden, Brent Smith and Jeremy York: Amazon.com Recommendations: Item-to-Item Collaborative Filtering

[2] Jianming He and Wesley W. Chu: A Social Network Based Recommender System.

[3] Gediminas Adomavicius and Alexander Tuzhilin: Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions

[4] Thanisa Numnonda: A real-time recommendation engine using lambda Architecture

[5] Nikos Manouselis, Hendrik Drachsler, R iina Vuorikari, Hans Hummel, Rob Koper: Recommender Systems in Technology Enhanced Learning

[6] Emanuel Lacic,Dominik Kowald and Elisabeth Lex Neighborhood Troubles: On the Value of User Pre-Filtering To Speed Up and Enhance Recommendations

[7] Yanxiang Huang, Bin Cui,Wenyu Zhang, Jie Jiang, Ying Xu TencentRec: Real-time Stream Recommendation in Practice

[8] Raymond J. Mooney, Loriene Roy: Content-Based Book Recommending Using Learning for Text Categorization

[9] Marko Balabanovic and Yoav Shoham: Content-Based, Collaborative Recommendation

[10] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl: Analysis of Recommendation Algorithms for E-Commerce

[11] Chhavi Rana and Sanjay Kumar Jain: A study of the dynamic features of Recommender systems

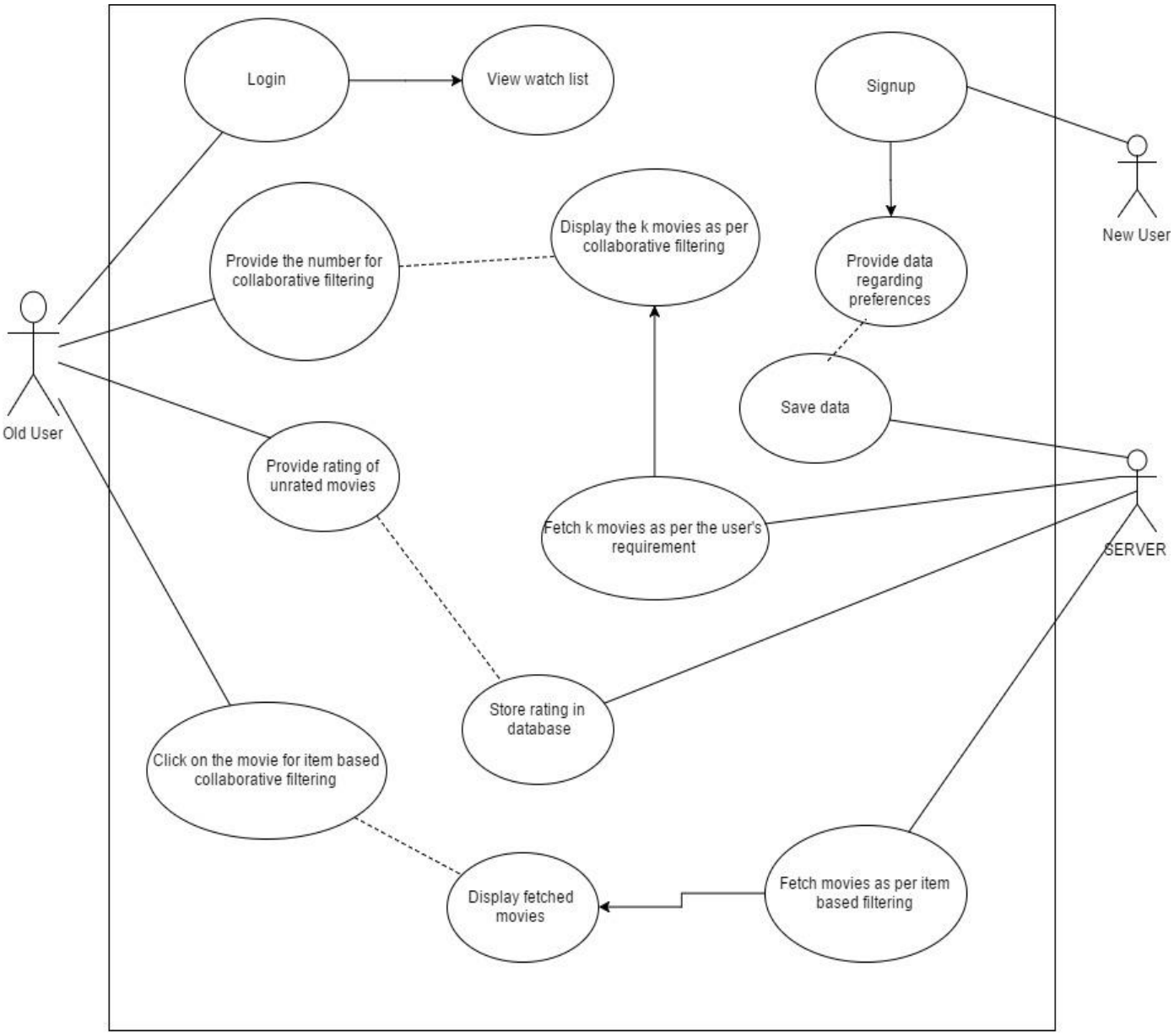
[12] Ago Luberg, Tanel Tammet and Priit JrvSmart City: A Rule-based Tourist Recommendation System

[13] Francesco Ricci, Lior Rokach, and Bracha Shapira Recommender Systems: Introduction and Challenges

[14] Zi-Ke Zhang, Chuang Liu, Yi-Cheng Zhang and Tao Zhou: Solving the Cold-start problem in recommender systems with social tags.

2. Overall Description

2.1 Use-Case Model Survey



2.2 Product Perspective

1. New User: - utilize the platform in following ways:-

    a. **Create Account: -**.Enter details for future login purposes.

    b. **Provide preference data: -** Enter data to overcome cold-start problem.

2. Old User: - utilize the platform in following ways:-

    a. **Login:** User enter into account through password thus able to see his watch list

    b. **Utilize item based collaborative filtering: -** User selects a movie which results in calculation and simultaneous display of a calculated rating for the movie as well as a list of movies recommended for him/her as per current choice.

    c. **Provide rating: -** User can provide rating to new movies as well as modify the ratings of previously rated movies.

    d. **Provide k for collaborative filtering: -** User provides a number k which shows k recommended movies as per other users with similar preferences.

    e. **Logout: -** Logout will remove user session.

3. Server: - provide support to users

    a. **Save data**: -To save data provided by users.

    b. **Show data**: -To show data after calculation and filtering.

2.3 Assumptions and Dependencies

- ✓ The user does not provide haphazard ratings.
- ✓ The new user provides information in a non-erratic manner.
- ✓ The user ratings have resemblance to his/her taste.

3. Specific Requirements

3.1 Functionality

There are number of functions that the application is supposed to perform which are as follows: -

- Firstly user have to login for that he/she have to register him/her self, so for that there will be a Sign Up page where user have to fill the entries like Username and Password. The new users will be redirected to a page asking for their preferences and likings in terms of movies.
- There will be login page / Sign-in page where user have to login with their username and password.
- There will be movie searching option.
- Then there will be a user based watch-list section page will be open as soon as the user logs in.
- The user can then select a movie which will result in the display of the movie description, a calculated rating as per item based filtering and a list of movies recommended.
- The user can also provide a value k and get a list of recommended movies after considering the ratings and tastes of similar users.
- In last user can log-out, just by clicking on the logout button.

3.2 Supplementary Requirements

- ✓ Server should be efficient and able to process data at fast rates.

- ✓ The users using this and importing the reports from this tool should have supporting software to run them.

- ✓ To use the application in the best possible way please read the tips displayed while using the tool.

1. External Interface Requirements

Requirements for the external interface define the tools, software or database elements with which the system or the component should interact.
Information from this section will give confidence that the system will properly interact with external components.
If different parts of the product have different external interfaces, then there should be a section with their description in the detailed requirements for each part.

1.1 User Interfaces

- Front-end: HTML/CSS, JAVASCRIPT: We aim to provide a comfortable yet pleasing user experience.
- Back-end: Flask- being a popular framework of Python, we plan to use its flexibility and simplicity for our server side development.

1.2 Hardware Interfaces

- Ubuntu, Windows.

1.3 Software Interfaces

- Operating System - Ubuntu.
- Database - MySQL
- Python- As python has numerous libraries which could be used for graph plotting and machine learning, it proves out to be apt for this project
- Libraries – sklearn, pandas, numpy, Pyspark.

1.4 **Communication Interfaces**

This project supports all types of web browsers.

2. Non-functional Requirements

2.1 Performance Requirements

2.1 **Safety Requirements**

If there is extensive damage to a wide portion of the database due to catastrophic failure, our database will truncate all the records and will start from a new state. This will cause no harm to user as we were storing only results of previous queries which was run by different clients across the globe.

2.2 **Security Requirements**

We are providing the results and data sets provided by a particular client to be accessed by all users, but if a user chooses to hide his/her transaction he/she can do so. Our server will assure that this data can't be accessed by anyone.

2.3 **Software Quality Attributes**

- *Availability - The server should be up and running at all times.*
- *Correctness - The difference between actual and predicted accuracies should be less.*
- *Maintainability - The data set provided by the user should not be distorted.*