CS 481 Information Retrieval Project

Final Report

Group Members:

No.	Name	Roll Number
1.	Gulshan Singh	U101115FCS094
2.	Kashish Chaurasia	U101115FCS107

Movie Recommender System

1. Introduction

The Movie Recommender System(MRSYS), which is developed in our project, uses the Cosine Similarity algorithm, to provide recommendations based on users' ratings. More specifically, rather than using an average value of ratings as reference, Cosine Similarity try to find users who share similar interests by analysing the previous ratings of a user. Based on the ratings of the similar users, it has found, the system will predict the ratings of a target user on the movies he/she has not watched yet. After that, a list of recommended movies will be generated according to the predicted ratings and provide to users. MRSYS provides a search function for user to search for movies by using Java Stream based or using Lucene. All these functions provide user an easier and faster approach to search for a movie, get familiar with a movie, collect favourite movies, and receive recommendations according to their preferences.

2. Problem Statement

The biggest issue facing recommender systems is that they need a lot of data to effectively make recommendations. It's no coincidence that the companies most identified with having excellent recommendations are those with a lot of consumer user data: Google, Amazon, Netflix, Last.fm. The more item and user data a recommender system must work with, the stronger the chances of getting good recommendations. But to get good recommendations, you need a lot of users, so you can get a lot of data for the recommendations. Another problem with search engine would be the rapidly changing data. Clearly an algorithmic approach will find it difficult if not impossible to keep up with fashion trends. The main focus is on the algorithmic challenges in compactly representing a large data-set while supporting fast searches on it. Lastly the search engine is unpredictable because it cannot give a perfect match every time.

3. *Functionality*

When the login option is selected, the user's username and password are prompted. If authentication is successful, the main menu is displayed. Otherwise, the login page is displayed.

The main menu offers the following options;

- *Add a new movie* — The user is prompted for Movie(title, release year, IMDb url). Users are not allowed to add movies already found in the database. User is also required to rate every movie they add.
- *Rate a movie* — User is prompted for a movie ID. When a ID found in the movie database is entered, the user may rate the movie within the allowed range of -5 to 5 in increments of 1.
- *Rate random movies* — Users are shown random movies (limited by number of movies in database) which they have not rated and prompts them for a rating.
- *Search movies* — Users are allowed to search the movies either by using Java Stream based or using Lucene .
- *Top 10 movies of all time* — A list of the 10 highest rated movies is displayed. They are ranked based on average user ratings.

- *Get personalised movie suggestions* — Displays a list of recommended movies based on how the user rated movies. The displayed movies are ones the user has not rated before. It is calculated based on the similarity of the user against other users who have rated some or all of the movies the user has rated. The movie suggestions are generated based on the rated movies of the user with the highest similarity.

4. *Methodology*

MRSYS is a console-based movie recommender system based on a user's previous movie ratings. Users must create an account before using this service and rate 10 movies. Raw data from MovieLens was used and XML is used for persistence. Libraries used include Guava, Princeton's Stdlib and Xstream.

1. ***Data storage and retrieval:***

Our dataset(movie lens) is in csv format and we are using XML as our datastore. So we at first extracted data from CSV files and then stored them in XML file in appropriate format.

1. At first the csv files are parsed and then stored it a HashMap/ArrayList. HashMap and HashSet are mostly used for speed and to avoid overhead of duplicate entities. We have implemented the comparator interface so that we can get the entities in sorted order of ids, rating etc.
2. Serialized using XStream for eventual storage in XML datastore.
3. Any changes to be made are pushed/popped into/from a stack.
4. Data is read using ObjectInputStream and put into a stack.
5. Data in stack is stored using ObjectOutputStream.

CS 481 Information Retrieval Project

Final Report

2. ***Searching:***

Broadly two methodologies are being used:

1. Using Simple Java Stream Based.
2. Using Apache Lucene.
    - ***Fuzzy Query***: Uses similarity based on Levenshtein edit distance. query generates matching terms that are within the maximum edit distance specified in fuzziness and then checks the term dictionary to find out which of those generated terms actually exist in the index.
    - ***Wildcard Query:*** Matches documents that have fields matching a wildcard expression that is given in the search query.
    - ***Boolean Query:*** Combining two or more different types of queries with different boolean expressions. Internally we can use different type of search methods for individual term queries of the boolean expression.
    - ***Phrase Query:*** Used to search a sequence of texts in a document. We can give two words in our search query and get results according to the given slop(The distance in the number of words, between the terms to be matched).

Overall working of Lucene part:

- Take all documents, split them into words, build inverted index of each word.
    - Text file
    - RAM Directory
- Once an index is built, we can search that index using a *Query* and an *IndexSearcher.*
- According to the user choice appropriate searching technique is applied.
- Documents with top scores are retrieved through a ranked list. Using Vector space model and Boolean model.
- Boolean model is used first to narrow down the docs that need to be scored

CS 481 Information Retrieval Project

Final Report

3. ***Recommendation:***

a. ***Personal Recommendation***

1) Finds the potentially similar users set

- Retrieves all of user A's positive ratings (ratings that have rating point above 0)
- From each rating, retrieves the rated movie, records the genres, and retrieves all users that have rated that movie that same rating point

2) Calculates similarity between user A and every entity in the set resulted from step 1, an example: of finding similarity between user A and user B:

- Again using all the ratings of user A to create a vector, called vector U
- From each rating, retrieves the rated movie and checks whether user B has rated it. If yes, gets the rating point, otherwise the rating point is considered 0. This process ultimately creates another vector, called vector V (need to ensure that every rating of a movie in vector U corresponds to the rating of the same movie in vector V, and the size of these vector are equal)
- Similarity is then calculated in Matrix class.

3) Now that U and V are in the same dimension, works out the angle between U and V.

4) Doing step 3 results in a set of angles of user A vs other users in the system, picks out the minimum angle (should be less than 90 degree, which is orthogonal), resulting in vectors that more or less point to the same direction.

b. ***Top 10 Recommendation***

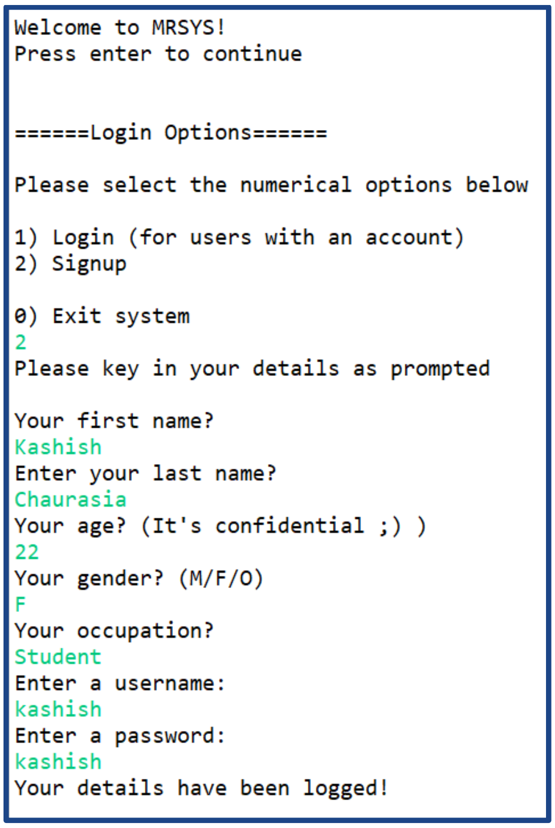
1) Doing step 4 will get to the most similar user to user A. Retrieves all positive ratings and get all movies from those ratings from this user and use the favorite genres of user A (collected in step 1) to filter movies to achieve a smaller set.

2) Returns the set, which contains the recommended movies.

5. Results & Conclusions

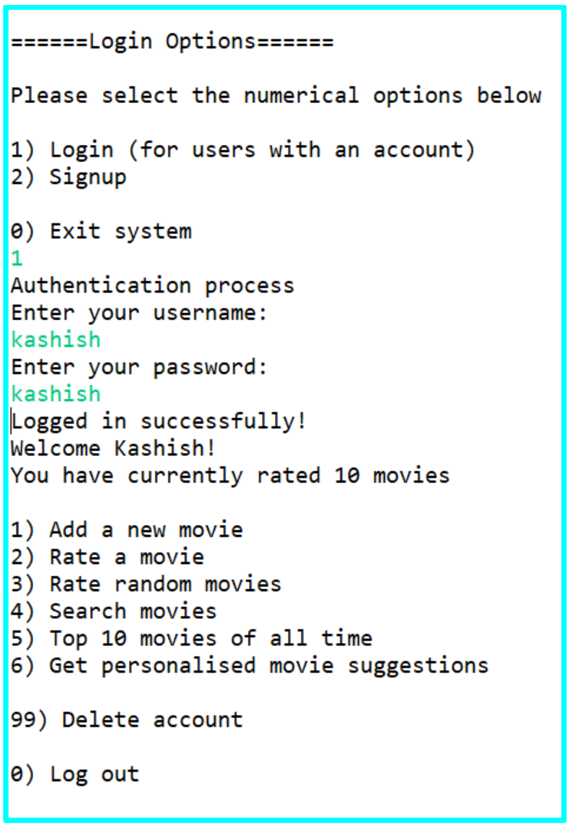
<u>*OUTPUTS:*</u>

Sign Up



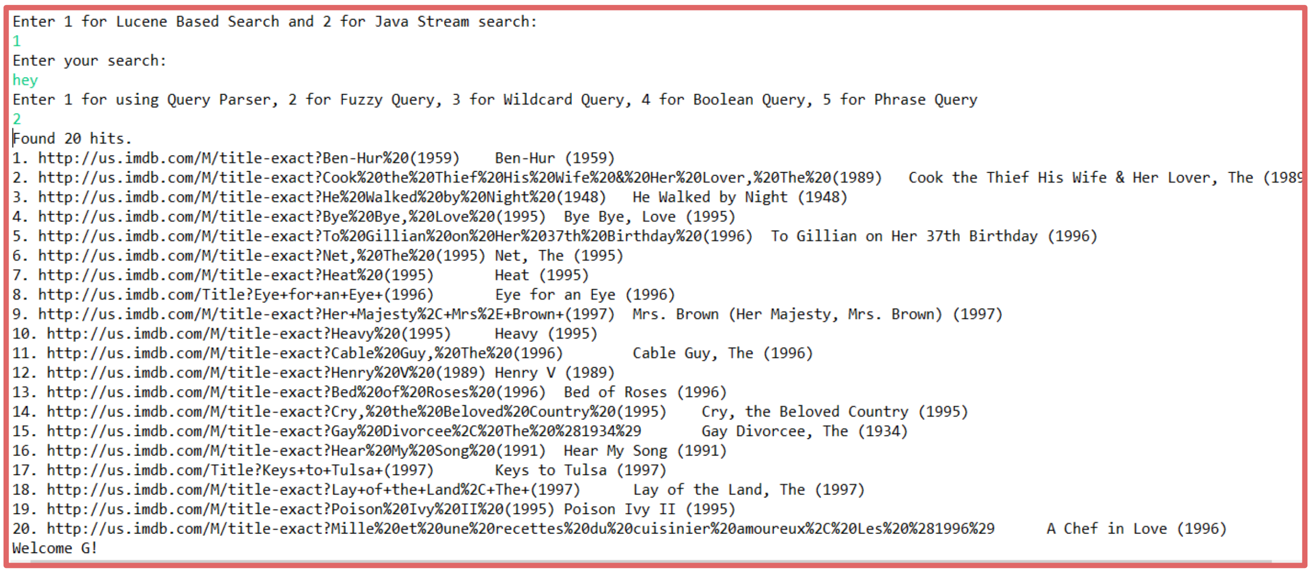
```
Welcome to MRSYS!
Press enter to continue


======Login Options======

Please select the numerical options below

1) Login (for users with an account)
2) Signup

0) Exit system
2
Please key in your details as prompted

Your first name?
Kashish
Enter your last name?
Chaurasia
Your age? (It's confidential ;) )
22
Your gender? (M/F/O)
F
Your occupation?
Student
Enter a username:
kashish
Enter a password:
kashish
Your details have been logged!
```

Menu



```
======Login Options======

Please select the numerical options below

1) Login (for users with an account)
2) Signup

0) Exit system
1
Authentication process
Enter your username:
kashish
Enter your password:
kashish
Logged in successfully!
Welcome Kashish!
You have currently rated 10 movies

1) Add a new movie
2) Rate a movie
3) Rate random movies
4) Search movies
5) Top 10 movies of all time
6) Get personalised movie suggestions

99) Delete account

0) Log out
```

Search

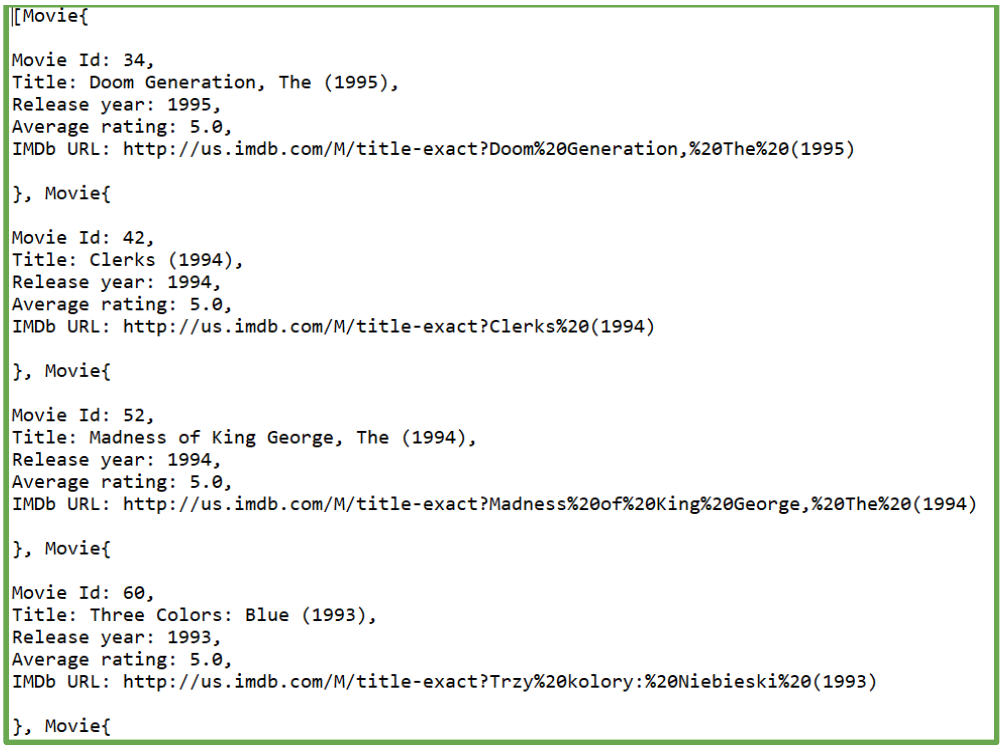


```
Enter 1 for Lucene Based Search and 2 for Java Stream search:
1
Enter your search:
hey
Enter 1 for using Query Parser, 2 for Fuzzy Query, 3 for Wildcard Query, 4 for Boolean Query, 5 for Phrase Query
2
Found 20 hits.
1. http://us.imdb.com/M/title-exact?Ben-Hur%20(1959)    Ben-Hur (1959)
2. http://us.imdb.com/M/title-exact?Cook%20the%20Thief%20His%20Wife%20&%20Her%20Lover,%20The%20(1989)   Cook the Thief His Wife & Her Lover, The (1989
3. http://us.imdb.com/M/title-exact?He%20Walked%20by%20Night%20(1948)   He Walked by Night (1948)
4. http://us.imdb.com/M/title-exact?Bye%20Bye,%20Love%20(1995)  Bye Bye, Love (1995)
5. http://us.imdb.com/M/title-exact?To%20Gillian%20on%20Her%2037th%20Birthday%20(1996)  To Gillian on Her 37th Birthday (1996)
6. http://us.imdb.com/M/title-exact?Net,%20The%20(1995) Net, The (1995)
7. http://us.imdb.com/M/title-exact?Heat%20(1995)       Heat (1995)
8. http://us.imdb.com/Title?Eye+for+an+Eye+(1996)       Eye for an Eye (1996)
9. http://us.imdb.com/M/title-exact?Her+Majesty%2C+Mrs%2E+Brown+(1997)  Mrs. Brown (Her Majesty, Mrs. Brown) (1997)
10. http://us.imdb.com/M/title-exact?Heavy%20(1995)     Heavy (1995)
11. http://us.imdb.com/M/title-exact?Cable%20Guy,%20The%20(1996)        Cable Guy, The (1996)
12. http://us.imdb.com/M/title-exact?Henry%20V%20(1989) Henry V (1989)
13. http://us.imdb.com/M/title-exact?Bed%20of%20Roses%20(1996)  Bed of Roses (1996)
14. http://us.imdb.com/M/title-exact?Cry,%20the%20Beloved%20Country%20(1995)    Cry, the Beloved Country (1995)
15. http://us.imdb.com/M/title-exact?Gay%20Divorcee%2C%20The%20%281934%29       Gay Divorcee, The (1934)
16. http://us.imdb.com/M/title-exact?Hear%20My%20Song%20(1991)  Hear My Song (1991)
17. http://us.imdb.com/Title?Keys+to+Tulsa+(1997)       Keys to Tulsa (1997)
18. http://us.imdb.com/M/title-exact?Lay+of+the+Land%2C+The+(1997)      Lay of the Land, The (1997)
19. http://us.imdb.com/M/title-exact?Poison%20Ivy%20II%20(1995) Poison Ivy II (1995)
20. http://us.imdb.com/M/title-exact?Mille%20et%20une%20recettes%20du%20cuisinier%20amoureux%2C%20Les%20%281996%29      A Chef in Love (1996)
Welcome G!
```

CS 481 Information Retrieval Project

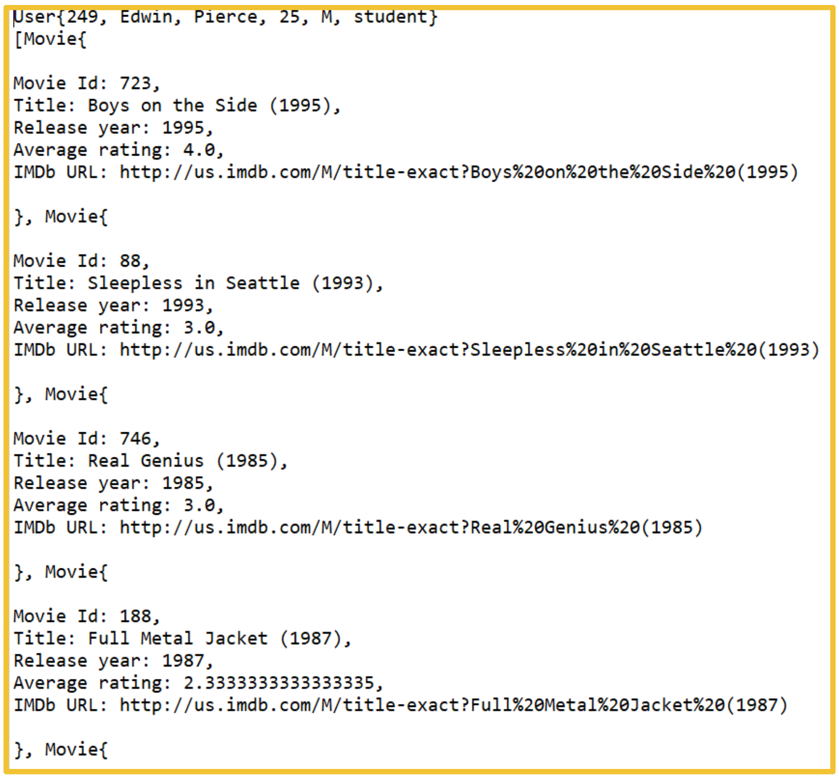
Final Report

Top 10 Movies Recommendation



```
[Movie{

Movie Id: 34,
Title: Doom Generation, The (1995),
Release year: 1995,
Average rating: 5.0,
IMDb URL: http://us.imdb.com/M/title-exact?Doom%20Generation,%20The%20(1995)

}, Movie{

Movie Id: 42,
Title: Clerks (1994),
Release year: 1994,
Average rating: 5.0,
IMDb URL: http://us.imdb.com/M/title-exact?Clerks%20(1994)

}, Movie{

Movie Id: 52,
Title: Madness of King George, The (1994),
Release year: 1994,
Average rating: 5.0,
IMDb URL: http://us.imdb.com/M/title-exact?Madness%20of%20King%20George,%20The%20(1994)

}, Movie{

Movie Id: 60,
Title: Three Colors: Blue (1993),
Release year: 1993,
Average rating: 5.0,
IMDb URL: http://us.imdb.com/M/title-exact?Trzy%20kolory:%20Niebieski%20(1993)

}, Movie{
```

Personal Movies Recommendation



```
User{249, Edwin, Pierce, 25, M, student}
[Movie{

Movie Id: 723,
Title: Boys on the Side (1995),
Release year: 1995,
Average rating: 4.0,
IMDb URL: http://us.imdb.com/M/title-exact?Boys%20on%20the%20Side%20(1995)

}, Movie{

Movie Id: 88,
Title: Sleepless in Seattle (1993),
Release year: 1993,
Average rating: 3.0,
IMDb URL: http://us.imdb.com/M/title-exact?Sleepless%20in%20Seattle%20(1993)

}, Movie{

Movie Id: 746,
Title: Real Genius (1985),
Release year: 1985,
Average rating: 3.0,
IMDb URL: http://us.imdb.com/M/title-exact?Real%20Genius%20(1985)

}, Movie{

Movie Id: 188,
Title: Full Metal Jacket (1987),
Release year: 1987,
Average rating: 2.3333333333333335,
IMDb URL: http://us.imdb.com/M/title-exact?Full%20Metal%20Jacket%20(1987)

}, Movie{
```

CS 481 Information Retrieval Project

Final Report

Conclusion:

Recommender systems are a powerful new technology for extracting additional value for a business from its user databases and in our project we've illustrated how to build a user-based collaborative filtering recommender system with different types IR search functionalities and usage of XML as database. We've followed a menu driven approach for our application and the main controllers is responsible for launching different functionalities of the system. Overall this model benefit users by enabling them to find movies they like. With the help of well-detailed metadata about our items, we can also add a content-based approach to recommendations.

6. Issues faced in the model solution:

1. The dataset contained few errors of missing field which resulted in exceptions being thrown. So we had to first find and fix all such errors.
2. Removal of stop words resulted in some important words being removed which resulted in bad prediction for some of the queries.
3. Top 10 results are purely based on average which is a bad measure for calculating top 10 movies of all time as averages are prone to outliers.

7. Issues faced in development

1. Conversion of CSV file to XML was a very tedious task because we have to store all three(users, movies, ratings) csv into 1 big xml file.
2. Parsing XML file and storing them into hashmap and hash sets of bean classes was a difficult and time taking task in comparison to any nosql DB like mongo as we have to code all the required methods.
3. Integration issues due to different coding approaches of different team members.