Institute of Systems Science, National University of Singapore

MOVIE RECOMMENDATION PROJECT REPORT

Movie Recommendation system

TEAMMEMBERS

SUN CHENGYUAN
XU MINGJIE

MASTEROFTECHNOLOGY

Content

1.0 EXECUTIVE SUMMARY

Many users on movie websites will post corresponding comments and ratings after watching a movie. When applied to a recommendation system, movie reviews and scoring analysis are very valuable. These comment information more effectively reflect the user's opinion of a certain movie. The degree of preference for a movie. When users browse the website to find movies, the website can use this information to recommend some movies that the target user may be interested in, which improves the user experience and helps users find their favorite movies to enjoy in the shortest time. Of course, we can also use some other features like page movie click rate, stay time, etc. to infer the user's preference and attention to the movie.

1.1 MARKET RESEARCH

In recent years, with the rapid development of the Internet, a large amount of network information is presented to us at the same time. For example, there are tens of thousands of movies on Netflix. In the face of such a large amount of information, how do users find the parts they are interested in? The recommendation system should be implemented. And the recommendation system is to recommend information that may be of interest to users according to their interests and hobbies to solve the problem of "information overload" and improve the efficiency of information processing.

Recommendation algorithms are also widely used in the Internet industry. Toutiao and Meituan, Dianping have personalized recommendations. Abstractly speaking, the recommendation algorithm is a fitting function for content satisfaction, involving user characteristics and content characteristics. The two major sources of the dimensions required for model training, and click-through rate, page dwell time, comments or orders, etc. can all be used as a quantified Y value, so that feature engineering can be carried out to construct a data set.

Then select a suitable supervised learning algorithm for training, and after getting the model, recommend the preferred content for the customer, such as Movies, TV shows, dramas.

2.0 PROBLEM DESCRIPTION

In the field of Movie website, Recommendation technology also shows great potential. For example, the recommendation algorithm that YouTube is famous for, they will base on your browser history and categorize videos, then recommend videos in your YouTube home page even you did not subscribe the Youtuber.

Since most movie websites divide the tags of movies, users may only choose the tags to browse movies. However, the user may have potential favorite movies but is restricted by the inherent movie classification and it is difficult to find the favorite movies. With a large number of movies, the user usually does not have enough time to select and browse the favorite movies one by one.

Then we believe that our recommendation system in this way can not only help users save time, recommend movies that interest users, but also improve the praise of the website or software.

2.1 PROJECT OBJECTIVE

In our project, we will create a movie watching website, including movie website, recommendation system, Python crawl code.

Users can browse movie information and query movies on the website, and the website will make real-time movie recommendations to the user based on the user's browsing history.

We use crawl code to crawl movie related information as our database, it obtains movie basic info, movie ratings, reviews, and user watching history, these data are very important for us to build recommendation system.

By burying points in the movie website system, the user's click events (such as which movie the user likes or the rating of a certain movie) are obtained and the information is transmitted to the recommendation system, the recommendation system makes corresponding processing based on the information, and recommends the result is stored in the MySQL database, and the web front end displays the recommended movie to the user by querying the database.

2.2 BUSINESS CASE

After we done our system, we do a survey around our friends and parents, which have 20 people include 3 young, 10 middle-aged and 7 old-aged, thus the test population samples are equally distributed. The survey conclusion is around two third of our test samples think this product is useful and can bring help to select their Interested movie.

3.0 RECOMMEND SYSTEM

Figure1 shows the reason system schematic for our movie recommendation. The historical data is downloaded from MovieLens Latest Datasets and contains 100,000 ratings and 3,600 tag applications applied to 9,000 movies by 600 users. Last updated 9/2018. The data set is generated by the ALS algorithm and the co-occurrence similarity algorithm respectively to generate the ALS recommendation table(alstab) and the similarity table(similartab). And also we interviewed some users who frequently browse movie websites and summed up some rules (rule engine) to optimize the system(Figure 2).

In the movie website system, the user's events (such as which movie the user likes or the rating of a certain movie are obtained) and the information is transmitted to the recommendation system, the recommendation system makes corresponding processing based on the information, and recommends the results are stored in the MySQL database, and the web front-end displays the recommended movies to the user by querying the database.

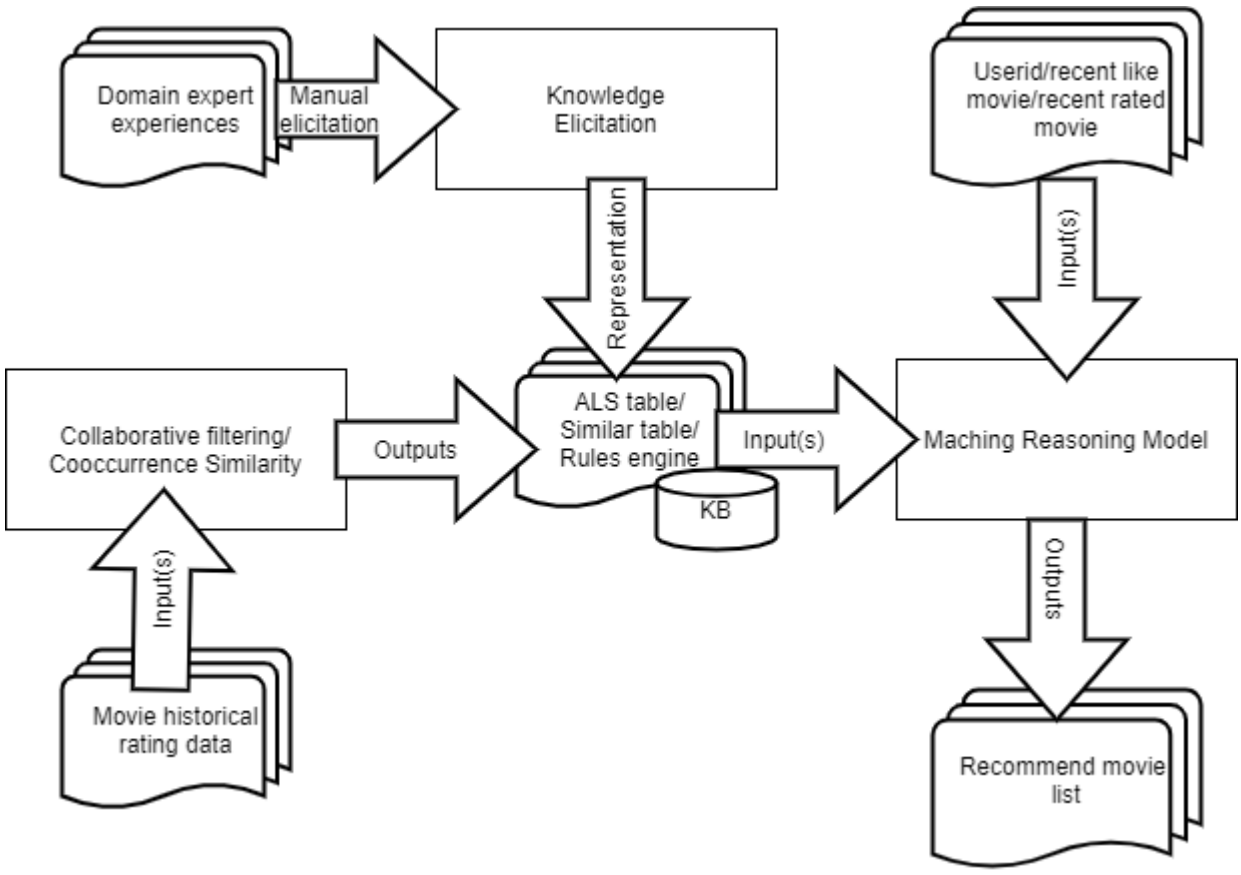


Figure 1: Schematic diagram of the reasoning system

3.1 SYSTEM STRUCTURE

In the actual business environment, massive amounts of data are often needed to support movie recommendations. A system that can process large amounts of data stably and quickly is essential. Therefore, we chose Hadoop and Spark clusters as the core architecture of the recommendation system. Considering the high price and complexity of physical machine clusters in this system, we use multiple docker contains as the basis to build Spark/Hadoop clusters.

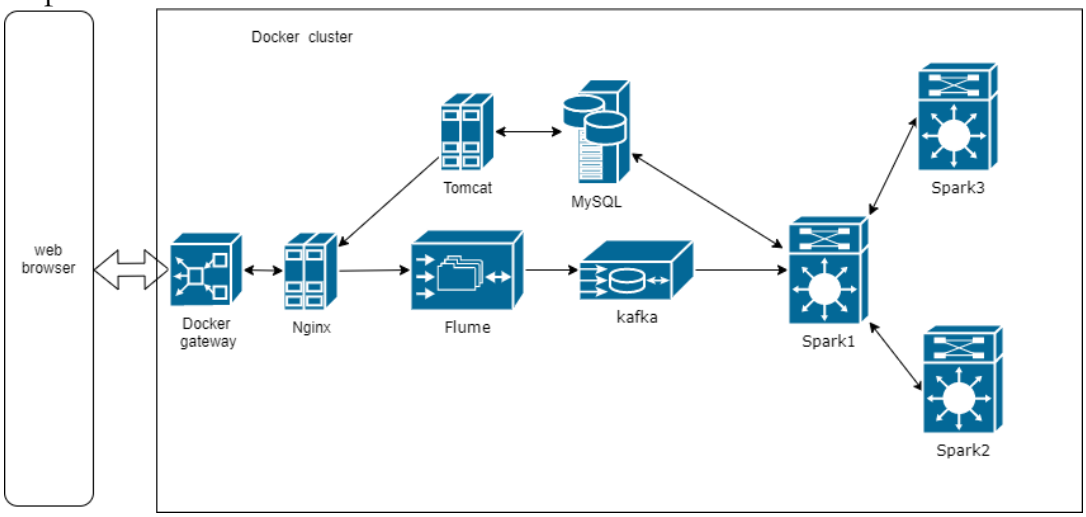


Figure 2 System topology diagram

3.2 Recommendation workflow

Model training and rules extracting

First, we need to extract the rating data from the csv file and do the data cleaning so that the data can be used to train the collaborative filtering model. Dataset can represent user, movie, rating, and timestamp respectively.

Secondly, we split the dataset into training set and testing set (6:4). ALS is a least squares regression model provided by MLlib. The principle is to control one matrix each time, optimize the other, and iterate repeatedly and finally converge. The three numerical parameters received by ALS are the characteristic length, that is, the size of K, the number of Iterations, and the lambda parameter. In this case, we user MLlib.ALS to train the model with training. And for improving the accuracy of the model, we user grid search, two variables are set as the two vectors, lambda = List (0.001, 0.005, 0.01, 0.015), iteration = List (10, 12, 15, 20). For each loop we calculate RMSE with testing dataset, and then output the best model for production.

The third step, we use the ALS model to predict for all users and select the top5 movies as recommend movie to output to asltab in MySQL.

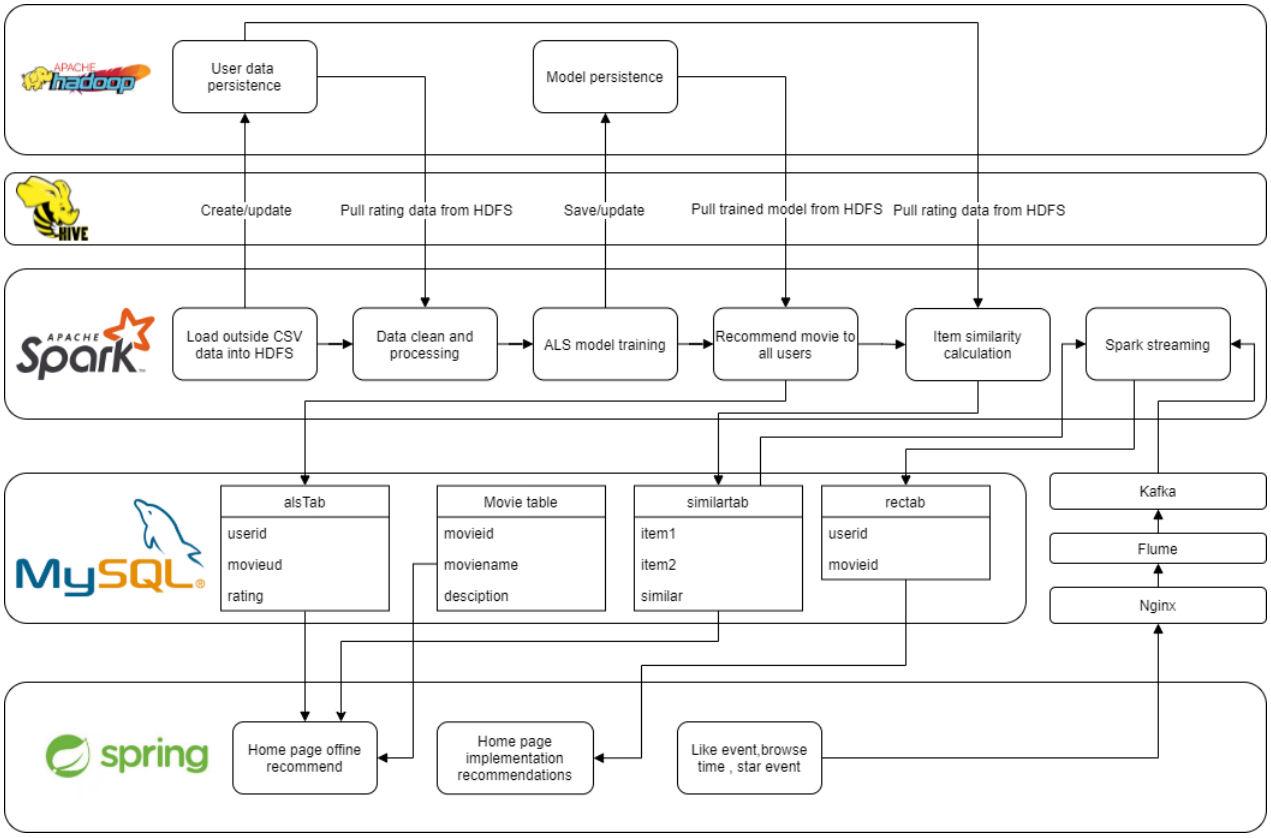


Figure 3 Recommendation System Structure

Additional step, we calculate the co-occurrence similarity foreach item based on the rating table and output into similar table(similartab).

Recommend movies to users (Off Line)

Suppose we want to recommend 5 movies that may be of interest to users. The built-in function (Figure 4) of the recommendation model helps us accomplish this. What we get are five Rating instances.

We further print out the names of the movies that the user likes and the movies recommended to him. Then we query the database to read the movie name

Recommend movies to users (Real time)

The offline recommendation is based on the system built with Spark streaming, Flume, Kafka and Nginx. There are 3 conditions will trigger the real time recommend:

1. Login user click like on a movie page
2. Login user rate a movie and the mark more than 4
3. Login user stay in a movie page for more than 10s

Once one of above triggered, Nginx will send the record into Flume/Kafka and received by Spark streaming. Then Spark will find the most similar movies based on the co-occurrence similarity and output into recommendation table(rectab)

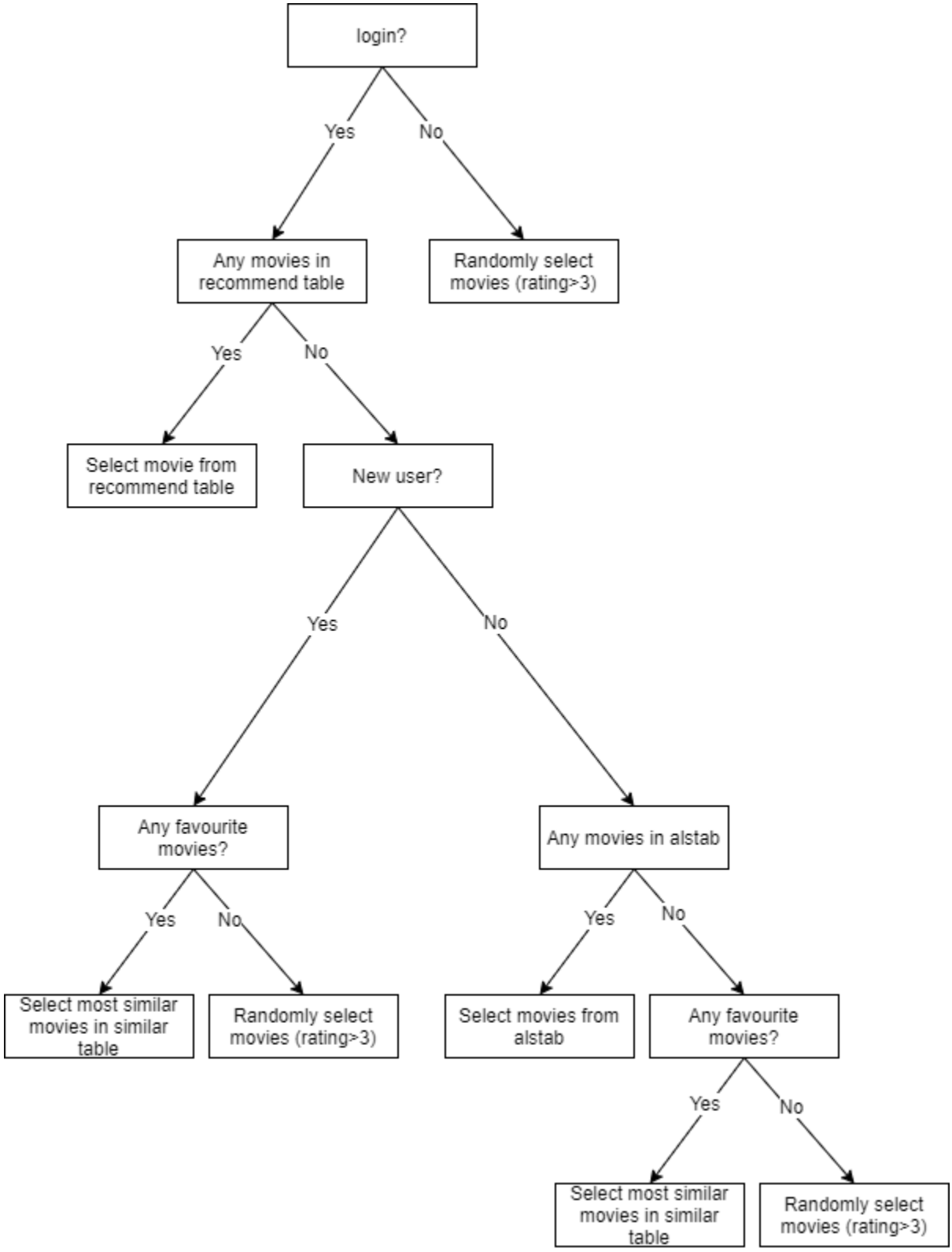


Figure 4 Rule engine

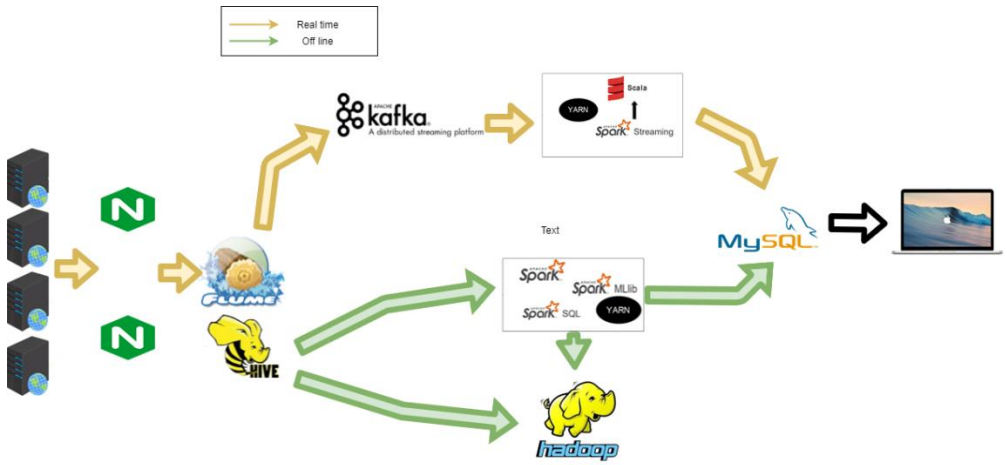


Figure 5: Recommend workflow

3.3 Collaborative filtering and ALS algorithm

Collaborative filtering

We assume that there are 100 users, 100 movies, and user ratings for movies. However, no website will have users who have watched all movies and rated them. In an ideal state, we would like to have 10,000 movie ratings, but this is obviously impossible. We may have 1,000 ratings, maybe 500.

What collaborative filtering has to do is to estimate the remaining 9,000 unscored scores based on the only 1,000 scores. What's the point of this? In this way, we can predict the movies that users may like and recommend them. At the same time, we can learn the characteristics of users, the characteristics of movies and find similar likes for them.

this id problem from the perspective of a matrix. Suppose we have U users and I movies. Some users have rated some movies. The ratings can be expressed as a U*I matrix. Most of the values in this matrix are missing. , Therefore a sparse matrix: Rm×n

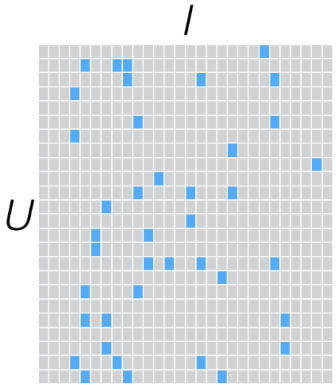


Figure 6

Now we want to represent the user as a K-dimensional feature, and the movie as a K-dimensional feature, and the dot product of the feature is the corresponding score.

Collaborative filtering is to use the existing scores to learn these K-dimensional features, so as to fill in the unrated items Rm×n≈Xm×kYTn×k

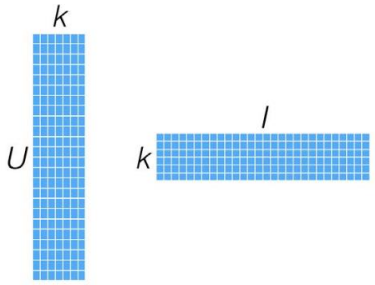


Figure 7

Mathematically speaking, it is to decompose the U*I matrix into the matrix product of U*K and K*I to fill in the unrated items while satisfying the existing scores.

The specific method to find the decomposition matrix is to use the least squares method (Alternating Least Squares, ALS).

ALS algorithm

Fortunately, we don't need to explicitly define these associated dimensions, but just assume that they exist, so the associated dimensions here are also called Latent factors. The typical value of k is generally 20 to 200.

This method is called probabilistic matrix factorization (PMF). The ALS algorithm is the application of PMF in numerical calculations.

In order to make the low-rank matrices X and Y approximate to R as much as possible, the following square error loss function needs to be minimized:

minx*,y*∑u,i is known(rui−xTuyi)2

minx*,y*∑u,i is known(rui−xuTyi)2

Taking into account the stability of the matrix, using Tikhonov regularization, the above formula becomes:

minx*,y*L(X,Y)=minx*,y*∑u,i is known(rui−xTuyi)2+λ(|xu|2+|yi|2)(2)

(2)minx*,y*L(X,Y)=minx*,y*∑u,i is known(rui−xuTyi)2+λ(|xu|2+|yi|2)

Optimize the above formula to obtain the training result matrix Xm×k, Yn×kXm×k, Yn×k. When forecasting, substitute User and Item into rui=xTuyirui=xuTyi to get the corresponding score prediction value.

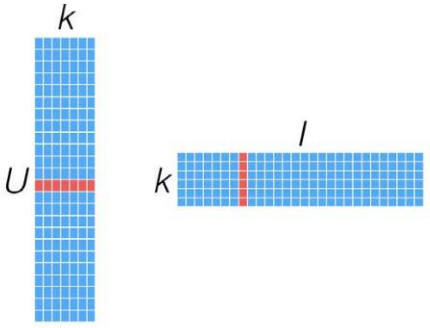


Figure 8

4.0 Web Solution

4.1 Movie Web

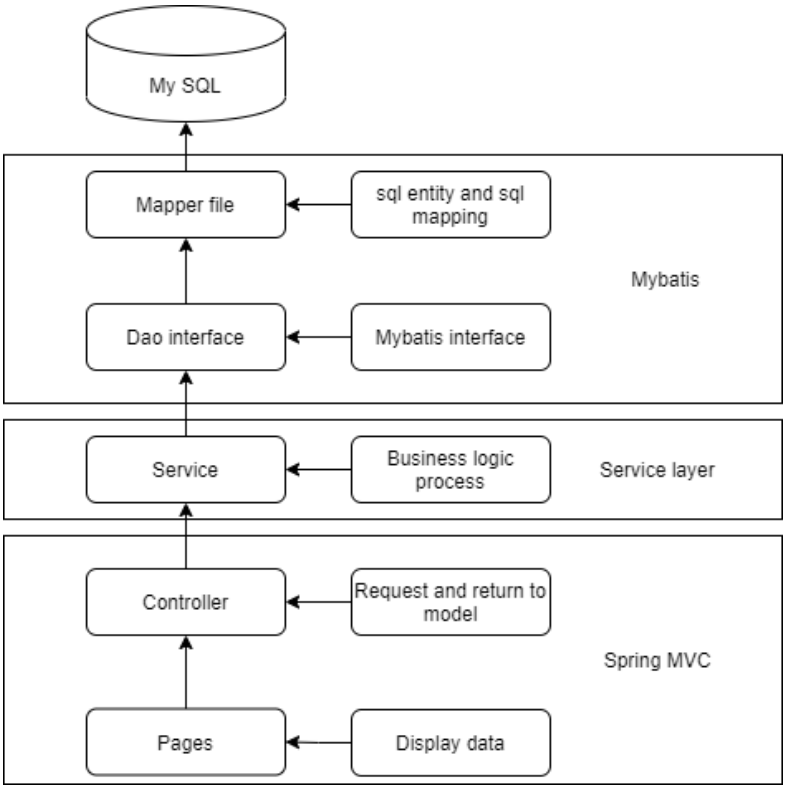


Figure 9 Web Structure

Spring MVC is applied to build Movie web.

A login has a user Profile page, user can view their like movie and rating movie. Everytime they rating a movie or like a movie will show the recording here.

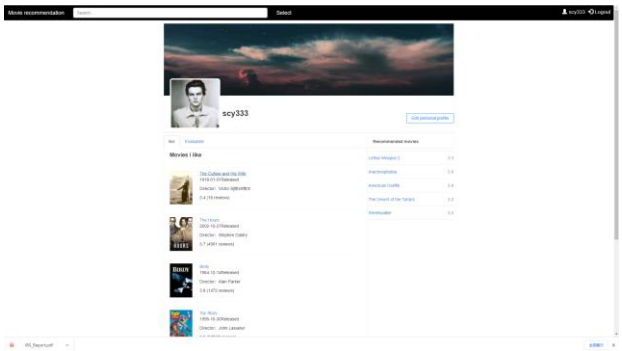


Figure 10

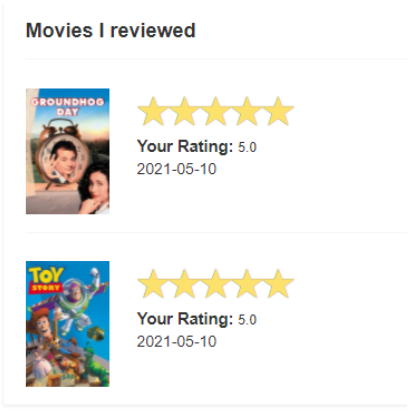


Figure 11

And right side will show our recommendation movie from our recommendation table.

The recommendation movie will also display to user in Movie home page. If user is not login, it will show random 5 movies in home page.

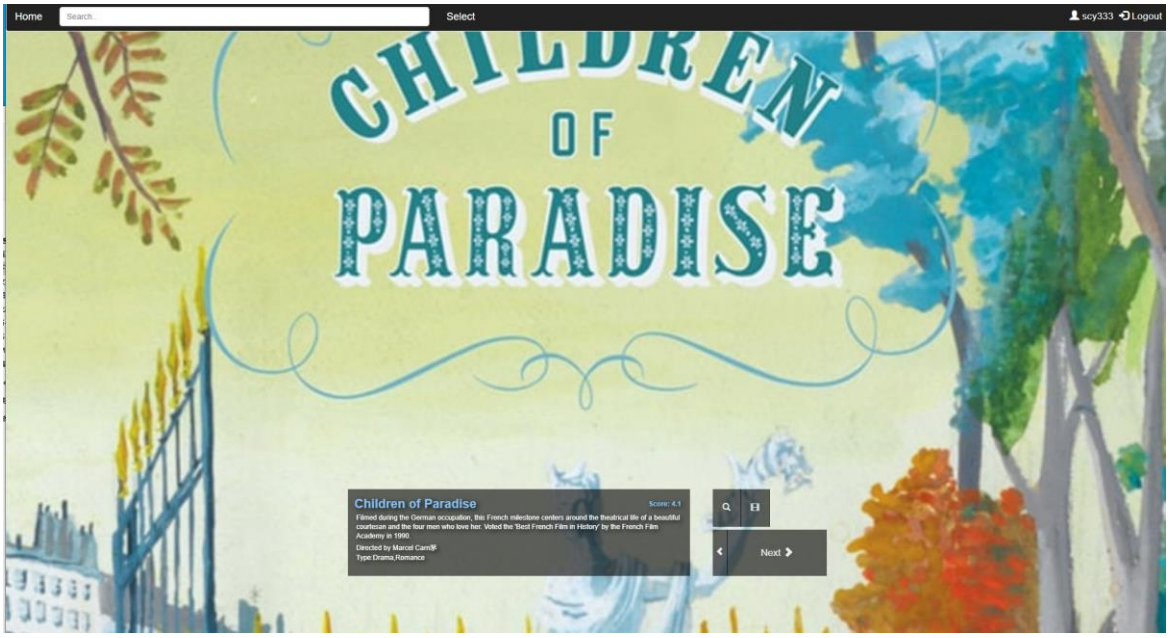


Figure 12

And in Movie select page. We will also show theses recommendation movie to user to make sure user and see the recommendation everywhere. In this page we can let user select their prefer movie by sorting category.

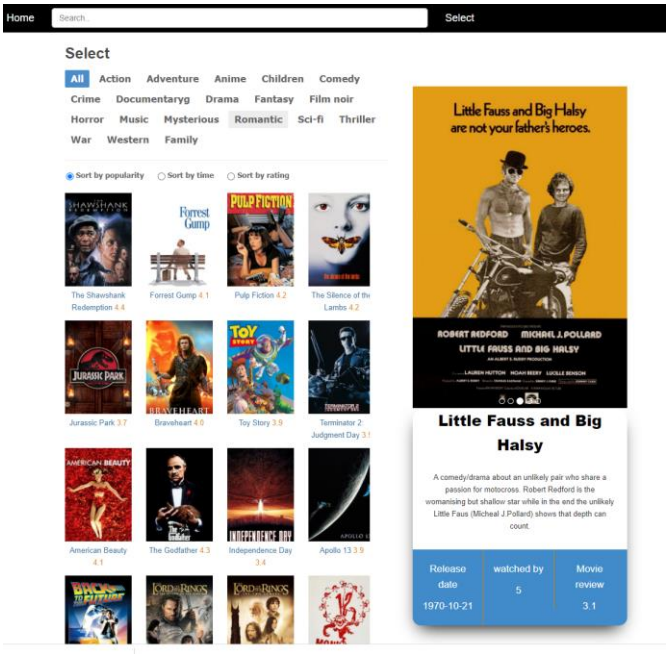


Figure 13

From Movie`s description page, Keypoints are we can make movie rating submit and like/unlike movie and show the movie`s similar movie list. All these similar movies are extract from our similarity table which is build by ItemCF. And The way to calculate similarity is co-occurrence similarity calculation.

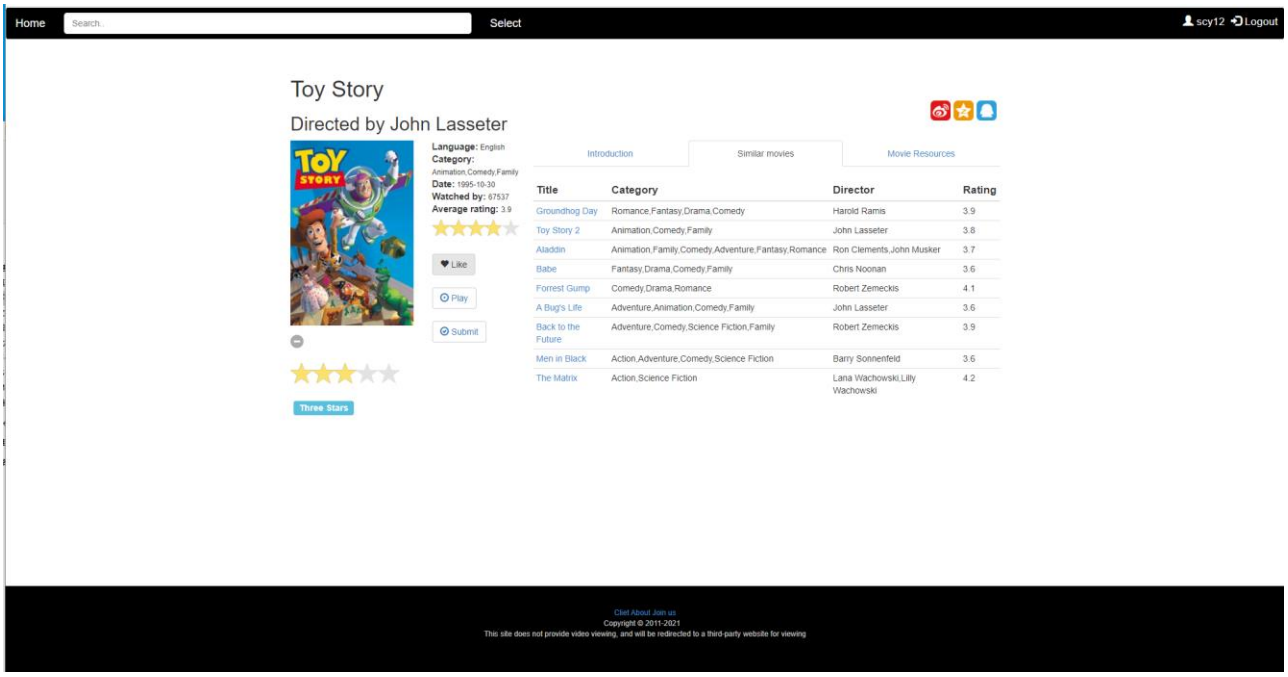


Figure 14

4.2 Project Scope

Login/Signup page.

We allow user to register or login an account to view website, User can enter this page in any page`s right top corner, also can logout there.

When user signup a new account, system will let user to choose any random movie as user`s like movie to calculate the recommendation movie.

Home page.

User can view 5 movies in home page. Non-login user will view random 5 movies, logged-in user will display the recommended 5 movies.

User can click Next or previous to check different movie information.

User can type movie name in search bar, it can show all related search result. Click one of the result will go to co-responding Movie description page.

User can click their name in right top corner to go to user profile page.

User can click select beside search bar to go to Movie select page.

Movie Select page.

User can select all kinds of movie category in this page. And it will display co-responding movie.

User can view 5 recommendation movies in the right part of page which is the same recommended in home page and user profile page.

User can sort movies result.

User profile page.

User can check their basic information in this page, name and profile image.

User can click edit profile to edit account`s password and email.

User can view all his/her Like/Unlike movies record in this page.

User can view all his/her rating movies record in this page. It will show movie`s information and user`s rate.

Movie description page.

User can check all movie detail description in this page.

User can rating this movie.

User can like/unlike this movie.

User can view this movie`s similar movie table in this page.

User can Share this movie and go to some reference link.

User stay in this page more than 10s will record into recommendation table to show user like this movie.

4.3 LIMITATIONS

The disadvantages of the ALS algorithm:

1. It is an offline algorithm. So If the amount of data grows very rapidly, then this will be a tricky problem to update ALS model.
2. Unable to accurately evaluate newly added users or products. This problem is also known as the Cold Start problem

5.0 CONCLUSION & REFRENCES

The construction of movie websites has already given users a complete movie watching process and experience. Our website has user management system, movie management system and movie recommendation system. Through the various behaviors of users watching movies and browsing on the website, we can effectively capture and transmit them to the recommendation system in real time to make recommendation feedback.

And through the offline model that has been trained and the real-time feedback recommendation model, it can accurately predict the movies that the user may like based on the user's viewing history, rating, collection and browsing time. This is not only due to the successful construction of the recommendation system, but to a large extent also has a great relationship with the accuracy and size of the crawled data. Although the lack of viewing records of some movie data causes the table of similar movies to be incomplete, but in general Has performed very well.

We invite our friends to try our movie website to simulate watching their favorite movie records. When they check the recommended movies, most people think they can really recommend the movies they are more interested in, and want to go to the movie description page to browse further.

5.1 IMPROVEMENTS:

When we obtained the data, we found that the picture information of many movies was missing. More importantly, many movies could not crawl their historical rating information, etc., so that when we train to create movie similar data, many movies will not have their own Similar movie data, which will also affect the performance of the push line system.

And the missing data will also affect the movie image`s display, some movie has no image, it should decrease the user experience.

Especially for a new user, we let him select a few of the ten randomly generated movies when creating a user, so when the user selects no similar movies, and in general, the new user will not have The scoring record of the movie. Therefore, in this case, the user will not be able to recommend because there is no similarity recommendation and no movie scoring record. We will recommend 5 random movies to the user every time the user enters the movie homepage, just like the way we do not log in users.

6.0 APPENDICES

APPENDICES 1: Project proposal

What we do

In this project, we designed a system which based on big data filtering engine-movie website, including movie website (front-end and back-end), recommendation system (Spark) and a Python crawl code.

Why we do it

We hope to build a movie website with a lot of data, where users can find their favorite movies in a short time, and the recommendation algorithm in this field shows great potential.

We hope that we can take advantage of the user's behavior and scoring preferences, as well as the recommendation system knowledge we learned in class, to design a recommendation system to improve the user's watching experience.

How we do it

We use the user's viewing records and movie ratings as data to perform item-based collaborative filtering, and predict the user's ratings of other movies by calculating the similarity between movies... and then based on the prediction Sort the ratings and recommend these movies to users who have watched the movies.

If it is a user who has not watched a movie, it will be given a default or selected favorite movie for calculation.

APPENDICES 2: Survey For Expert Knowledge

Nowadays, watching movies is a very common way of entertainment. In order to understand how users choose to watch movies, we surveyed an expert. The following is a conclusion from our survey:

We: What is the first step in picking up a movie to watch?

Expert: Usually, I will go to the movie website where I have a member to check if there are any new movies I am looking forward to. If not, I will browse the homepage a little bit to see if there are any movies I like. If not, I usually look up the movie category to continue. Browse

We: So if there is no target movie at the beginning, you are actually in the process of looking for a movie aimlessly?

Expert: Yes. Because I have watched so many movies. Although I like to watch some Movies in a specific category, but I still want to see what I may like to watch.

We: How long do you usually look for movies in your favorite movie category?

Expert: Sometimes it takes long sometimes not too much time. It`s based on my interested. But every time I browse the specific category, The movies always the same, especially some new movie come out. That will waste my time to pick the movie I did not watched before.

We: OK, and what kind of movie you like?

Expert: Generally speaking, I prefer to watch action movies and crime movies, but I also watch romance movies and documentaries. I like many kinds of movies, so this also caused my difficulty in choosing movies.

We: So whenever you want to watch a movie, I hope to save the time of choosing a movie, right?

Expert: Yes.

We: Do you have the habit of commenting and scoring movies?

Colleague: I usually rate the movie after I watch it, but film reviews are based on my mood and only write when I really like this video or really hate the video.

We: If the system recommends a movie to you, would you take the time to browse it?

Expert: Yes, of course. But I hope I don't recommend too many movies at once. If the recommendations are all the movies I like to watch, I may have a hard time picking them out, but I will recommend them to me at the same time I also like movies in my favorites to watch later

We: That's really helpful. We have already taken too much for your time. Thanks for your answers. Hope you can make a hit video in the future.

APPENDICES 3: Individual project report

Sun Chengyuan:

personal contribution

I use data crawling to crawl movie information from https://movielens.org/, including movie ratings, pictures and user viewing records, etc. These are all for creating and training Item based movie similar tables, as well as creating and training ALS The model refers to a new recommendation table, so that every time the user opens the movie homepage, it will input according to the user's id and use the ALS model to calculate and update the recommendation table, and then recommend the top 5 movies with the highest value.

If the user does not have a watch history, showing that there is no favorite movie, no review rating, etc., we will randomly show the user 5 high-rated movies.

What I learn

Although we think ideas are good, when we actually apply them, we will encounter a series of difficulties. At the beginning, We have had a lot of discussions on the front-end display, and we have been thinking for a long time about what specific recommended solution we need to show to users. Later, we plan to display the movies recommended by the system on the movie home page, movie category page (select page) and user profile page.

Secondly, when we specified the recommendation display, because we would always get the data in the same table, but we made more than one recommendation based on user behavior, which made us often unable to let others when testing recommendations. Can not Recommend effective insertion recommendations. This also wastes a lot of our time to adjust the code to how to implement recommendations caused by multiple user behaviors

Finally, I think teamwork is very important. We can learn a lot from the team members, or they can supplement or improve our ideas, and use a better structure to accomplish our goals. Although the progress of our project was seriously slowed down due to the excessive work pressure of the team members, we finally completed the code improvement and deployment after staying up all night every day. This makes us understand that we must do a good job of project schedule and node inspection in advance, otherwise it will seriously disrupt the original plan

Skills in other situation

In this process, we have used many technologies which we learn from class, such as ItemCF, cosine similarity calculate, ALS, and knowledge engine. Finally, we used the co-occurrence similarity calculation to calculate the movie similarity table.

itemCF is more suitable for personalized recommendation scenarios (such as commodities, movies, short videos, etc.). It focuses on the current behavior of the current user, so that it can recommend the content of the current related movies, while UserCF is more group-oriented (for example, news Recommended scenario), when calculating the K similar users of

the current user, when the K value is too large, UserCF recommends more hot spots; so we finally plan to use itemCF.

Since it is impossible to rate all the movies in the user's historical viewing records, we used the ALS algorithm learned in class to use the user's existing viewing records as a benchmark to predict the ratings of other unwatched movies

Xu Mingjie:

personal contribution

I was in charge of building docker cluster and also establish communication between Spark/Hadoop implantation, and communication between the extension and other components including Kafka and MySQL. And I was also responsilble for the Java SSM web system, including a tomcat server, nginx, backend and frontend.

What I learn

In this project, I first learned that for a recommender system or intelligent system, it is usually not simply done through one method, but through a combination of multiple methods, and different methods face the situation is often different, depending on the actual situation and user needs. For example, in this project, we not only use collaborative filtering to predict users' ratings of movies, but we also use the same similarity to get similar movies. , The first method is used for offline recommendation, so as to find some long-term user preferences, while the second method is used to respond to the user's short-term exploration desire.

Secondly, I learned that when dealing with massive amounts of data, the hdfs file system references relational databases, which is more advantageous.
Relational databases such as MySQL used in this project are better than hdfs in the face of high-frequency requests. This is why we use MySQL and Hadoop for real-time processing and offline computing, respectively.

Finally, I think that when building a complete system, we need to do a complete and detailed decomposition of the system at the beginning, so as to understand the time required to complete each item. In this project, considering the limited team members, we Just breaking it down into a recommendation system, modularizing it, and roughly estimating the time resulted in the actual time being much longer than the expected time, thus delaying the delivery of the product.

Skills in other situation

In this project, I think I have gained a lot of experience that is beneficial to the future work in the direction of artificial intelligence or data processing. For example, related to the calculation of item relevance, the establishment of recommendation models, these methods can not only be used for the movie recommendation described in this article, it is also suitable for book recommendation, music recommendation, and even all items related to evaluation. I think that such a system can almost find applications in any industry, so it is helpful for me to find the implementation of intelligent systems in various industries and directions. point. And in this project, the Spark recommendation system I implemented is also often used in

actual industrial environments. or example, in an industrial environment, processing a large amount of data is often achieved through distributed clusters, and a lot of knowledge about the Hadoop/Spark system that I learned in this project can be helpful in my actual work in the future

APPENDICES 4: Installation and User Guide

Environmental requirements: Win10 / Linux with 16GB memory or higher

1. Install docker 20.10.5 in your computer.

2. Download and unzip dockerMovieRec.zip into current work directory (\MovieRecommend\SourceCode\MovieRecommend)

3.CD to \MovieRecommend\SourceCode\MovieRecommend, then run command "docker compose up -d"

4. After start all container in docker, Open localhost:85/movie/

5.Open terminal on your machine and key in command "docker exec -it spark1 bash", after connected with spark1,
then type "nohup /usr/local/spark-1.6.2-bin-hadoop2.6/bin/spark-submit --class com.cloud.streaming.SparkDrStreamALS --jars /data/traintools/kafka-clients-0.8.2.0.jar,/data/traintools/metrics-core-2.2.0.jar,/data/traintools/kafka_2.10-0.8.2.1.jar,/data/traintools/spark-streaming-kafka_2.10-1.6.2.jar,lib/mysql-connector-java-5.1.49-bin.jar /data/traintools/SparkDrStreamALS.jar 1>/data/Log/task.log 2>&1 &"
then press enter and close terminal, now the spark will update recommend movies based on your browse history.

6. Some account for testing:
username:test1 password:abc123456 remarks: This account is a prebuild account, and included in the training model, so will recommend movies based on the model when login
username:test2 password:abc123456 remarks: This account is a new account, and not included in the training model, so will recommend movies based on favorites.

7. Commands for model updating (Open terminal first):

1) docker exec -it spark1 bash
2) source /etc/profile
3) spark-submit --class com.cloud.datacleaner.RatingDataSQL --jars lib/mysql-connector-java-5.1.49-bin.jar /data/traintools/RatingDataSQL.jar
4) spark-submit --class com.cloud.modeltrain.ModelTraining /data/traintools/ModelTraining.jar
5) spark-submit --class com.cloud.modeltrain.RecommendForAllUsers --jars lib/mysql-connector-java-5.1.49-bin.jar data/traintools/RecommendForAllUsers.jar

8. Commands for updating item similarity table (may report error because of memory limit):

1) docker exec -it spark1 bash
2) source /etc/profile

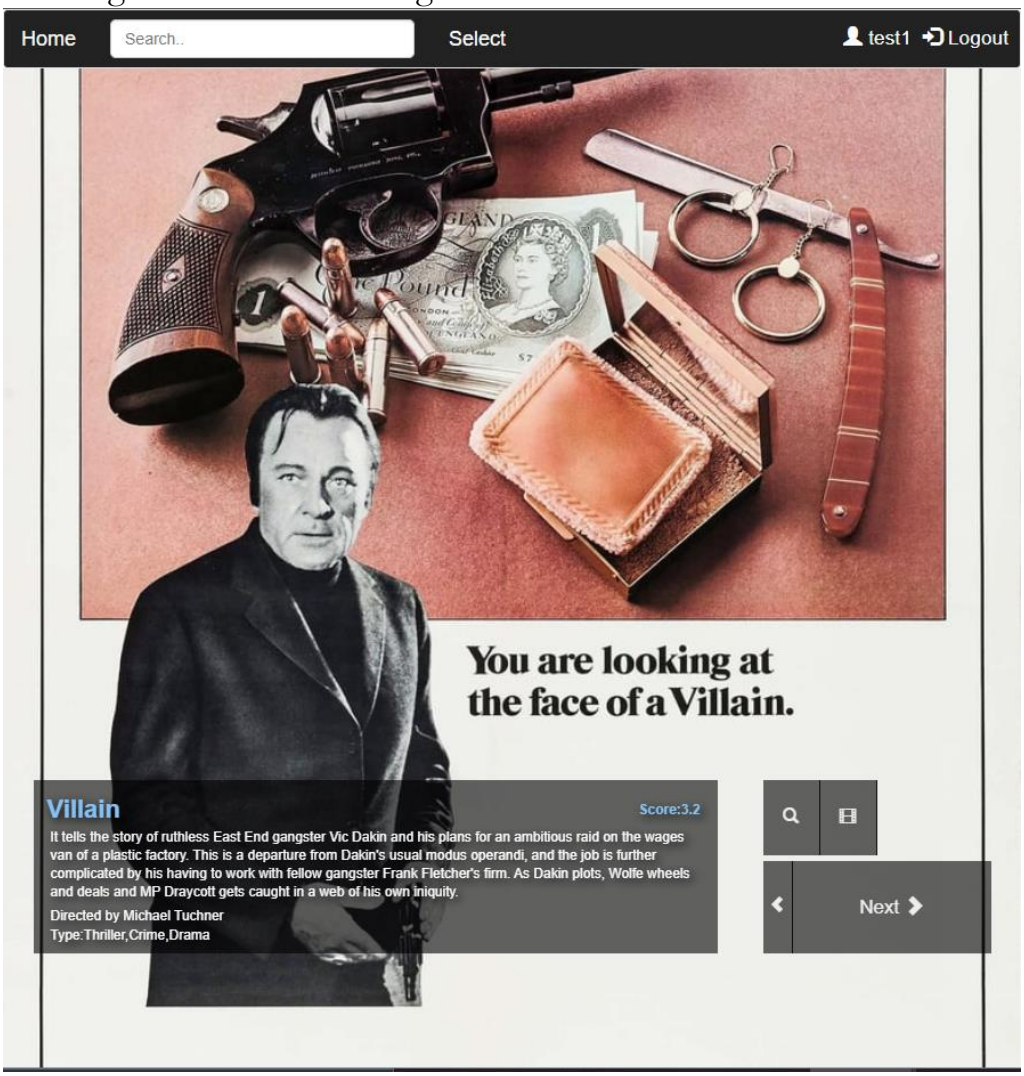
3) spark-submit --class com.cloud.datacleaner.RatingDataSQL --jars lib/mysql-connector-java-5.1.49-bin.jar /data/traintools/RatingDataSQL.jar
4) spark-submit --class com.cloud.similaritycacu. ItemCF /data/traintools/ItemCF.jar

Running System Captures:
Start docker cluster and Spark streaming

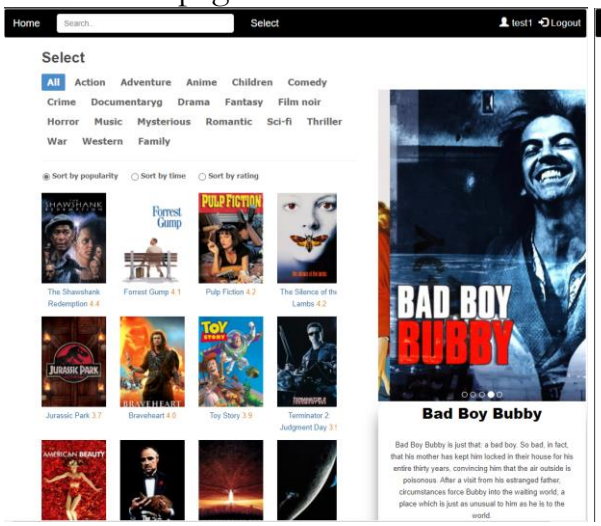
```
(base) PS C:\Users\11351> cd F:\MovieRecommend
(base) PS F:\MovieRecommend> docker compose up -d
[+] Running 10/10
 - Network "movierecommend_movierec"  Created                                    0.2s
 - Container mysql                    Started                                  32.5s
 - Container tomcat                   Started                                  34.0s
 - Container nginx                    Started                                  35.2s
 - Container zookeeper                Started                                  34.4s
 - Container kafka                    Started                                  33.8s
 - Container flume                    Started                                  34.2s
 - Container spark2                   Started                                  37.7s
 - Container spark3                   Started                                  35.9s
 - Container spark1                   Started                                  38.2s
(base) PS F:\MovieRecommend> docker exec -it spark1 bash
 * Starting OpenBSD Secure Shell server sshd                                  [ OK ]
root@663e5093a465:/# nohup /usr/local/spark-1.6.2-bin-hadoop2.6/bin/spark-submit --class com.cloud.streaming.SparkDrStreamALS --jars /data/traintools/kafka-clients-0.8.2.0.jar,/data/traintools/metrics-core-2.2.0.jar,/data/traintools/kafka_2.10-0.8.2.1.jar,/data/traintools/spark-streaming-kafka_2.10-1.6.2.jar,lib/mysql-connector-java-5.1.49-bin.jar /data/traintools/SparkDrStreamALS.jar 1>/data/Log/task.log 2>&1 &
[1] 1154
root@663e5093a465:/#
```

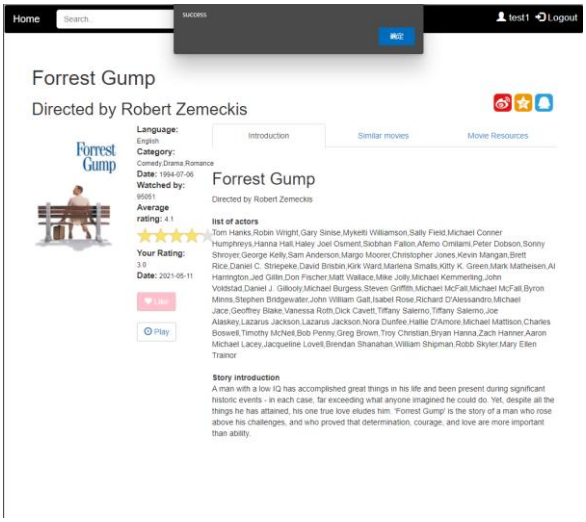
Login page(May take a while for first launch):

Home page after login with test1 showing the recommend movies based on ALS model:

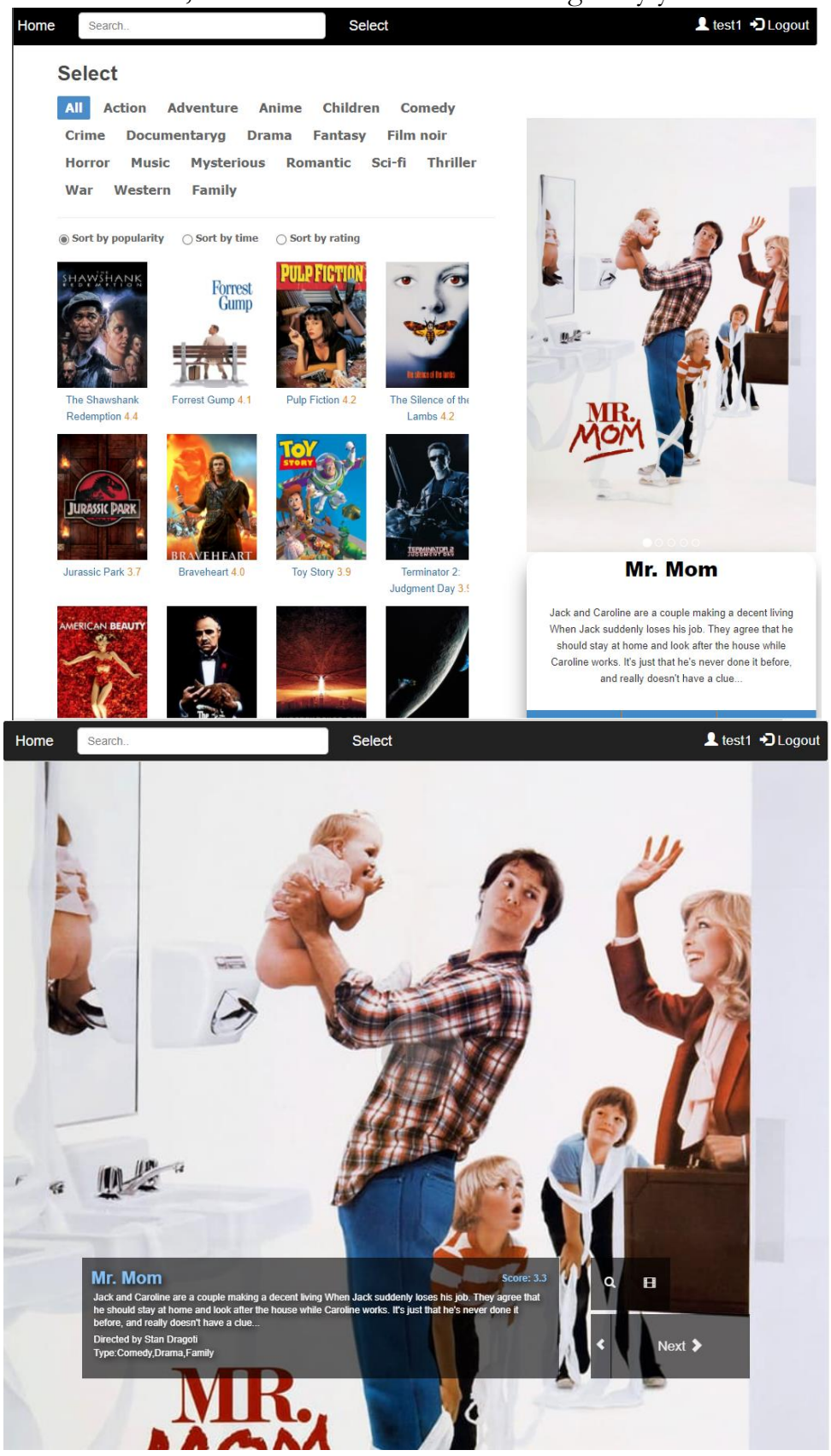


Go to *Select* page and select one movie and add to like:

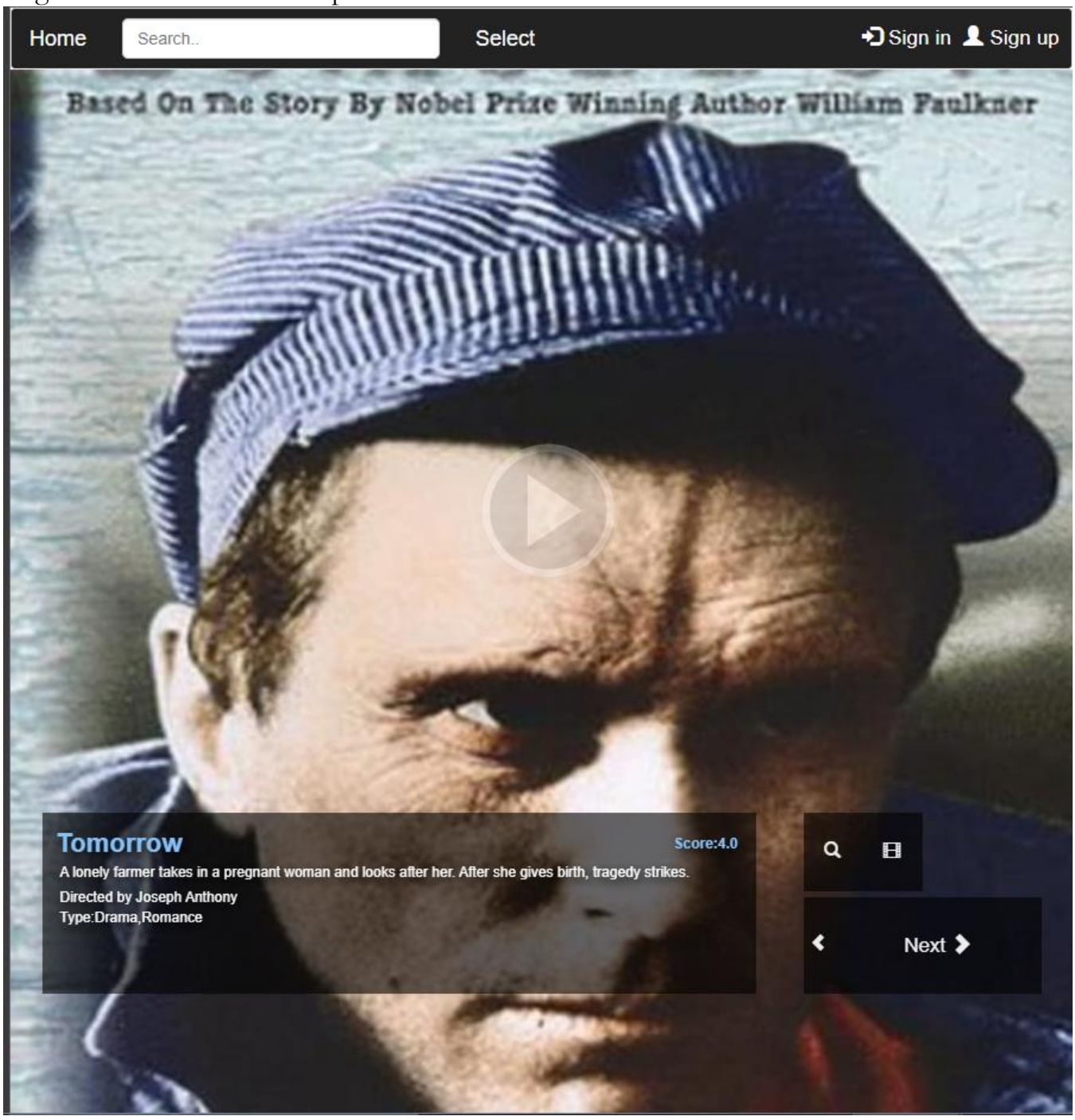




Go back to Select or Home, the recommandation are changed by your like movie:



Logout and movies will be updated



Note: Some movies has no similar movies , so the page will not be refreshed if you select those standalone movies lol~