Movie recommender application

AMOD-5250H-Termpoject-HoangHaiThanh-0772659

Hoang Hai THanh

2023-04-12

Contents

1 Introduction

In today's digital age, movie streaming services offer their users a vast library of movies. However, the abundance of choices often leads to indecisiveness and a time-consuming browsing experience. In addition, users may need help choosing a movie that aligns with their interests and preferences, leading to frustration and dissatisfaction with the movie-watching experience.

To address the above issue, we propose a movie recommendation application that utilizes advanced algorithms based on item-based and user-based collaborative filtering. Collaborative filtering techniques enable our application to analyze the user's previous movie-watching behaviour and preferences to recommend movies most likely to interest them.

User-based collaborative filtering works by identifying other users with similar movie preferences to the user and recommending movies they have enjoyed. On the other hand, item-based collaborative filtering works by analyzing the similarities between different movies and recommending movies similar to the ones the user has already watched and enjoyed.

Our application combines these techniques to provide users with personalized recommendations that cater to their tastes and preferences. Furthermore, by utilizing collaborative filtering, our application helps users overcome the challenge of browsing through an extensive movie library and ensures a satisfying movie-watching experience.

The data used in this application was extracted from MovieLens 100K Dataset[1], which was collected by the GroupLens Research Project at the University of Minnesota throught the MovieLens website (movielens.umn.edu) during the seven-month period from September 19th, 1997 through April 22nd, 1998.

2 Background

Recommender systems [2] are a type of information filtering system that provides personalized recommendations to users based on their preferences and past behavior. They are widely used in various applications, including e-commerce, social networks, and entertainment platforms. In recent years, the development of recommender systems has been driven by advances in machine learning and data mining, as well as the increasing availability of large-scale user data.

Collaborative filtering (CF)[3] is a widely used technique in recommender systems that leverages the similarities between users and/or items to provide personalized recommendations. There are two main approaches to CF: user-based and item-based. User-based CF predicts a user's preferences based on the preferences of similar users, while item-based CF predicts a user's preferences based on the similarity of items they have interacted with.

User-based CF[4] algorithms typically involve calculating similarity between users based on their past item interactions. Similar users are identified by computing the cosine similarity between their item vectors, where each entry in the vector corresponds to a rating or interaction with a particular item. Once similar users are identified, the algorithm recommends items that have been rated highly by these users but not by the target user.

Item-based CF[5] algorithms, on the other hand, identify similar items based on the overlap of users who have interacted with them. The similarity between items is computed by comparing the sets of users who have interacted with each item. Once similar items are identified, the algorithm recommends items that are similar to those that the target user has rated highly.

Both user-based and item-based CF have been widely studied in the literature, and several variants of these algorithms have been proposed to improve their performance. One common variant is the use of neighborhood-based approaches, such as k-nearest neighbors [6], to identify similar users or items. Another variant is the use of matrix factorization techniques[7], such as singular value decomposition (SVD)[8], to reduce the dimensionality of the user-item interaction matrix and to learn latent factors that capture the underlying preferences of users and items.

User-based and item-based CF have been applied in various domains, including e-commerce[9], social networks[10], and online advertising[11]. They have shown to be effective in improving user engagement and satisfaction, and have also been used to drive business revenue by increasing sales and customer retention.

However, both approaches have some limitations. One of the main limitations of user-based CF is the scalability problem, where the computation of user similarity becomes computationally expensive as the number of users and items in the system grows. Item-based CF, on the other hand, suffers from the sparsity problem, where the similarity between items cannot be computed for items that have not been interacted with by any user.

To address these limitations, hybrid CF[12] approaches have been proposed that combine both user-based and item-based CF, as well as other recommendation techniques such as content-based filtering or knowledge-based recommendation. Hybrid approaches can improve the performance of CF by leveraging the strengths of different techniques and mitigating their weaknesses.

In conclusion, user-based and item-based CF are important techniques in recommender systems that have been extensively studied in the literature. Ongoing research is focused on developing more accurate and efficient CF algorithms, as well as addressing ethical and fairness considerations.

2.1 Recommender algorithms used in the prototype

First of all, let's define some notations used in this report. Let $R = r_1, r_2, ..., r_N$ be the set of N users, $C = c_1, c_2, ..., c_M$ be the set of M items (items in the prototype are the movies). We use matrix $I = (i_{nm})_{N\times M}$ to store data about interaction between users and items, where $i_{nm} = 1$ denotes n-th user had interaction (saw, clicked, liked, etc.) with m-th item, while $i_{nm} = 0$ might mean either 1) missing data or 2) n-th user had no interaction with c-th item. For an active user $r_a \in R$, we define the set of unknown items to user u_a as $C_a = C \backslash \{c_l \in C | i_{al} = 1\}$.

2.1.1 Popular items

This is a non-personalized recommender stragedy that recommends to the users the most popular items they have no interaction with.

2.1.2 Collaborative filtering

Collaborative filtering is algorithm that predicts or recommends about user's preferences based on their past behavior, such as item ratings or purchase histories.

There are two categories of collaborative filtering algorithms: memory-based CF and model-based CF algorithms, as classified by Breese et al. (1998) [13]. Memory-based CF algorithms utilize the entire user database, or at least a significant sample of it, to generate recommendations. The user-based collaborative filtering algorithm is the most widely used memory-based CF algorithm. However, this approach is not very scalable since the entire user database must be processed in real-time to create recommendations. Meanwhile, algorithms based on models utilize the user database for acquiring knowledge of a more condensed model such as clusters containing users who have similar preferences. This model is then employed to generate recommendations.

2.1.2.1 User-base User-base CF is a popular memory-based algorithm used in recommender systems. It works base on the assumption that users with the near preferences would interact similarly with new items. Consequently, it is possible to make predictions for missing data of a user by initially identifying a group of users (or *neighborhood*) with the same interests and subsequently consolidating their interactions to calculate a prediction.

The neighborhood is defined in terms of similarity between users, by taking a given number of most similar users (k nearest neighbors, or *knn*). One of the most popular similarity measures for CF are the Pearson correlation coefficient and the Cosine similarity, which are defined as:

$$m_{Pearson}(x,y) = \frac{1}{n-1} \sum_{l=1}^{n} \left(\frac{x_l - \bar{x}}{s_x} \right) \left(\frac{y_l - \bar{y}}{s_y} \right)$$

and

$$m_{Cosine}(x,y) = \frac{x \cdot y}{\|x\| \|y\|}$$

To calculate the similarity between two users x and y, we use calculate $m_{Pearson}(r_x, r_y)$ or $m_{Cosine}(r_x, r_y)$ where (r_x, r_y) are x-th and y-th row of matrix R, representing user x and y.

Once the similarities between users are populated, we can find k neighborhood of an active user u_a, $N(a)$ by taking k nearest neighbors. Then, we can use the interaction data of the neighborhood to predict interaction for the active user u_a as:

$$\hat{i}_{am} = \frac{1}{\sum_{n \in N(a)} s_{am}} \sum_{n \in N(a)} s_{am} i_{nm}$$

where s_{am} is the similarity coefficient between user a and user m.

2.1.2.2 Item-base Item-based CF, as described in [14], is a model-driven technique for generating recommendations. It works by analyzing the rating matrix to identify connections between different items, and assumes that users tend to favor items that share similarities with those they have already enjoyed.

In this method, instead of calculating similarity between users, we would calculate similarity between items. To keep the model size optimal, for each item we find k most similar items and their similarity and store them to a $n \times k$ matrix.

To calculate prediction on interaction of user u_a, we use the following formula:

$$\hat{i}_{am} = \frac{1}{\sum_{i \in S(m)} s_{mi}} \sum_{i \in S(m)} s_{mi} i_{ai}$$

Where $S(m)$ is the item m's set of the k most similar items, and s_{mi} is the similarity coefficient between item m and item i.

2.1.3 Hybrid recommendation

Hybrid recommendation is the strategy that aggregates the recommendations of several algorithms to make the final recommendation to the users.

In this prototype, we created hybrid recommender by aggregating recommendations from a popular-item model, an item-based CF model and an user-based CF model, then recommend the top k items to the user.

3 Instruction

3.1 Getting started

3.1.1 Run from pre-installed application

The application is available at the following URL and can be accessed via most of the popular web browsers:

`https://haithanhhoang.shinyapps.io/MovieRecommendation/`

3.1.2 Run from sourcecode

3.1.2.1 Prerequisites

The following softwares and modules are required:

- git: Installation manual is available at Installing Git
- R: Installation manual is available at https://cran.r-project.org/
- R modules:
  - shiny
  - recommenderlab
  - shinyjs

3.1.2.2 Download the sourcecode

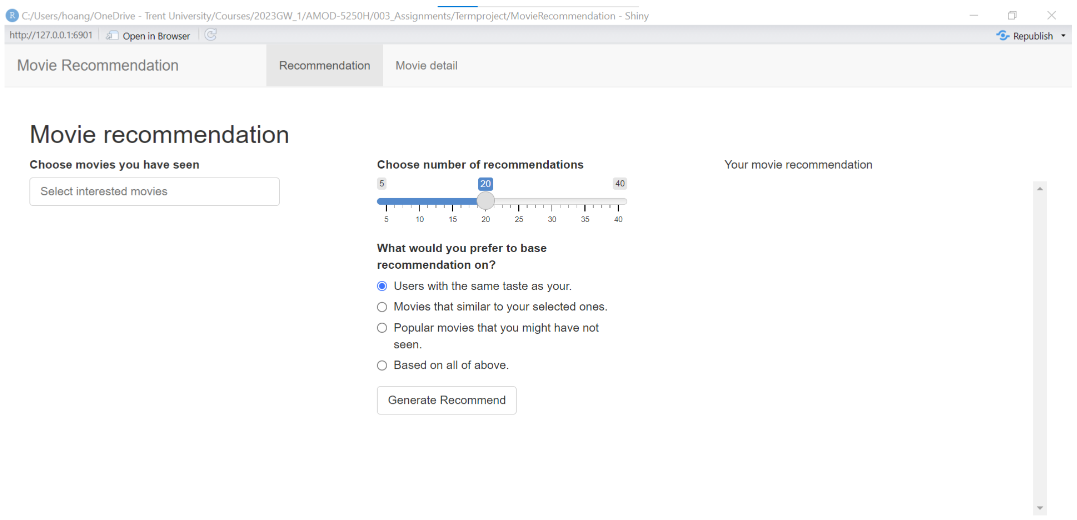
1. Open git bash
2. Run following commands in the git bash:
   - Navigate to desired parent folder:
     `cd [path/to/parent/folder]`
   - Download sourcecode
     ```
     git clone -n -b termproject  --depth=1 --filter=tree:0 \
       https://github.com/haithanhhoang/AMOD-5250H

     cd AMOD-5250H
     git sparse-checkout set --no-cone 003_Assignments/Termproject/MovieRecommendation
     git checkout
     ```

3.1.2.3 Run the application

1. Open R console
2. Run following commands in the R console
   - Navigate to Application folder
     `setwd([path/to/parent/folder]/AMOD-5250H/003_Assignments/Termproject/)`
   - Run the application
     ```
     library(shiny)

     # Run application at port 6901, you can change port number as you wish
     runApp('MovieRecommendation', port = 6901)
     ```
   - Access the application
     If the application is initiated successfully, it should be accessed by entering the following address in the internet browser:
     `http://localhost:6901`
     The homepage of the application is as follow:


C:/Users/hoang/OneDrive - Trent University/Courses/2023GW_1/AMOD-5250H/003_Assignments/Termproject/MovieRecommendation - Shiny
http://127.0.0.1:6901
Open in Browser
Republish
Movie Recommendation
Recommendation
Movie detail

Movie recommendation

Choose movies you have seen

Select interested movies

Choose number of recommendations

5 20 40

What would you prefer to base recommendation on?

- [✓ Users with the same taste as your.]
- Movies that similar to your selected ones.
- Popular movies that you might have not seen.
- Based on all of above.

Generate Recommend

Your movie recommendation

3.2 Generate recommendation

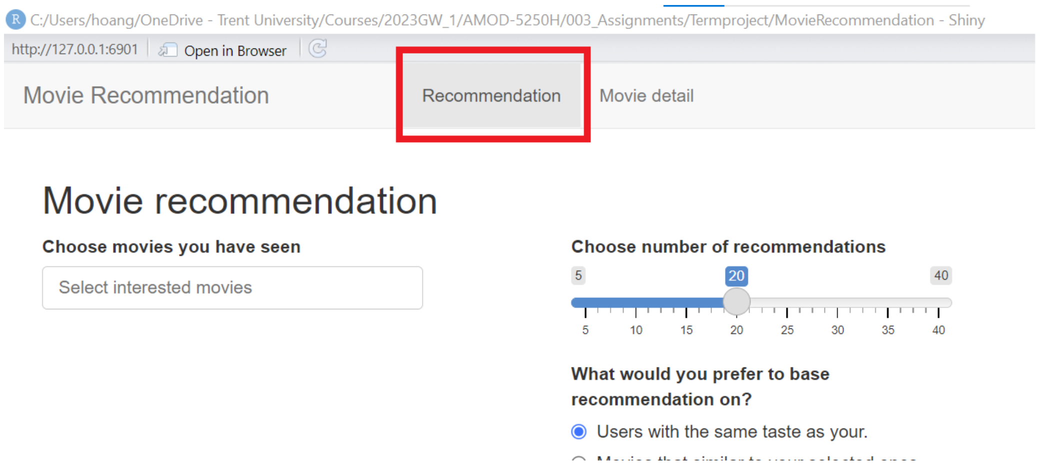
To Generate recommendation, users need to proceed the following steps:

1. Choose the movies that the user has interested in.
2. Choose the number of recommendation to generate
3. Select the recommend stragedy
4. Generate the recommendation

3.2.1 Choosing movies

To choose the movie, users operate as follow:

1. Navigate to the "Recommendation" tab under the home page


C:/Users/hoang/OneDrive - Trent University/Courses/2023GW_1/AMOD-5250H/003_Assignments/Termproject/MovieRecommendation - Shiny
http://127.0.0.1:6901
Open in Browser
Movie Recommendation
Recommendation
Movie detail

Movie recommendation

Choose movies you have seen

Select interested movies

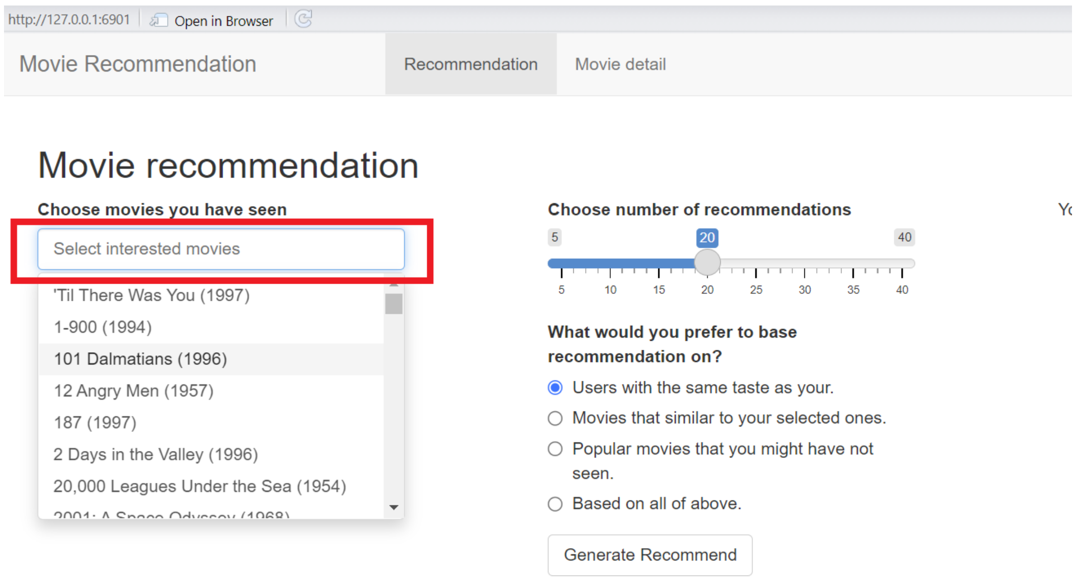
Choose number of recommendations

5 20 40

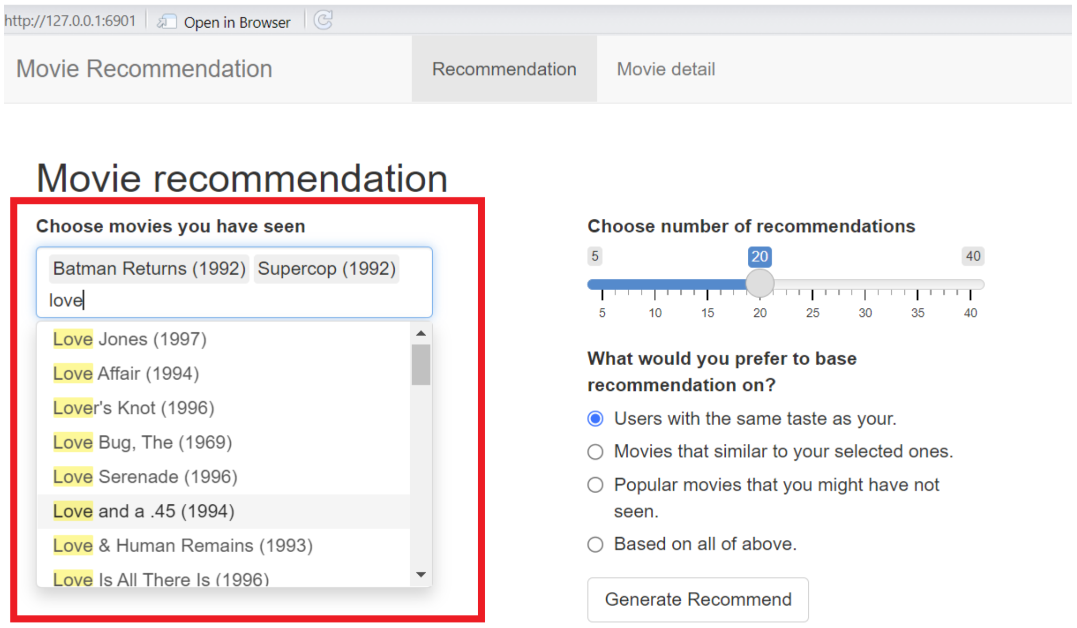
What would you prefer to base recommendation on?

- [✓ Users with the same taste as your.]

2. Click on the text input area under the "Choose movies you have seen" label to view the movie list.

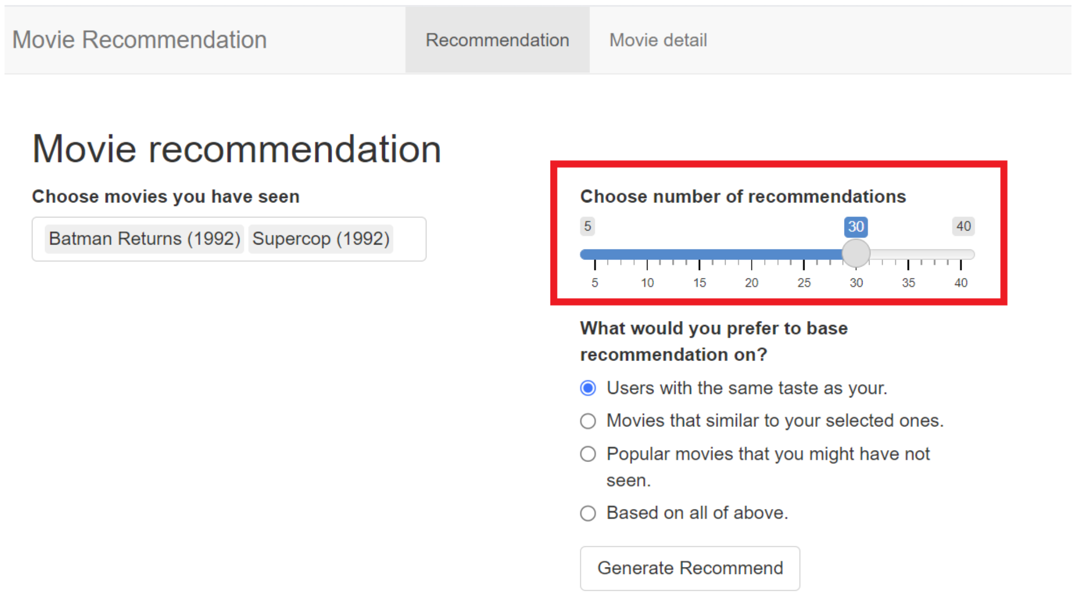

http://127.0.0.1:6901
Open in Browser
Movie Recommendation
Recommendation
Movie detail
Movie recommendation
Choose movies you have seen
Select interested movies
'Til There Was You (1997)
1-900 (1994)
101 Dalmatians (1996)
12 Angry Men (1957)
187 (1997)
2 Days in the Valley (1996)
20,000 Leagues Under the Sea (1954)
2001: A Space Odyssey (1968)
Choose number of recommendations
5
20
40
What would you prefer to base recommendation on?
Users with the same taste as your.
Movies that similar to your selected ones.
Popular movies that you might have not seen.
Based on all of above.
Generate Recommend

3. Select movies that the user has interested in. The user can choose multiple movies. To find the movie by its title, the user can input text to the field to filter the list as shown in the picture.



3.2.2 Choose number of recommendation

Users can adjust the slider to choose the number of movies to be recommended. For example, to generate recommendation that includes 30 movies, a user can set the slider as follow:



Movie recommendation

Choose movies you have seen: Batman Returns (1992), Supercop (1992)

Choose number of recommendations: 30 (range 5–40)

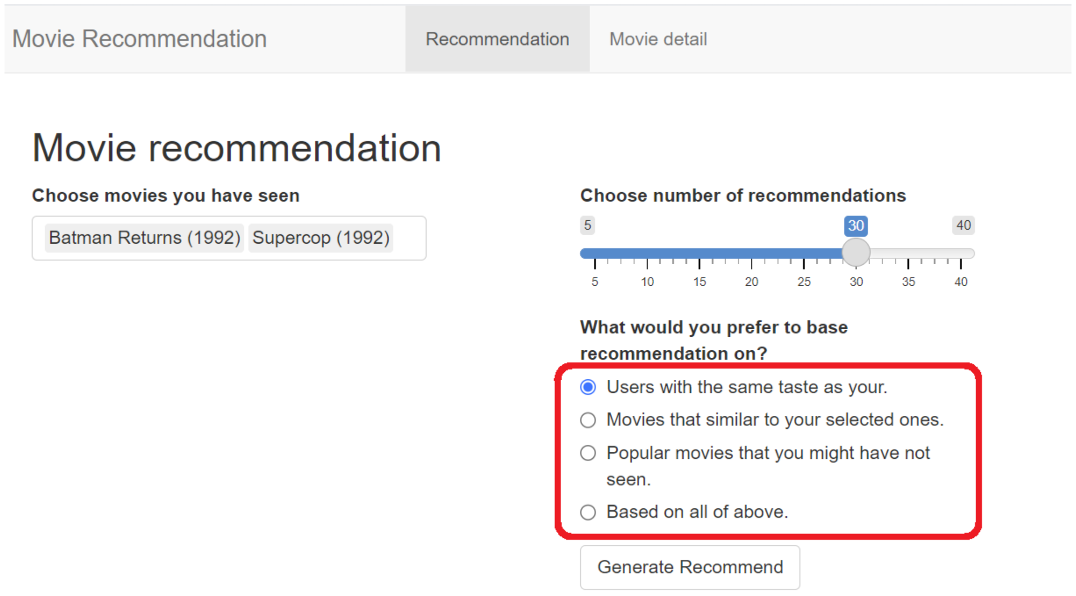
What would you prefer to base recommendation on?

- [✓ Users with the same taste as your.]
- Movies that similar to your selected ones.
- Popular movies that you might have not seen.
- Based on all of above.

3.2.3 Select recommend algorithm

To select recommend algorithm, users can select one of the four options under "What would you prefer to base recommendation on?" label.

1. The first option will generate recommendation using user-based strategy.
2. The second option will generate recommendation using item-based strategy.
3. The third option will generate recommendation using popular-item strategy.
4. The fourth option will generate recommendation using hybrid strategy.



Movie recommendation

Choose movies you have seen: Batman Returns (1992), Supercop (1992)

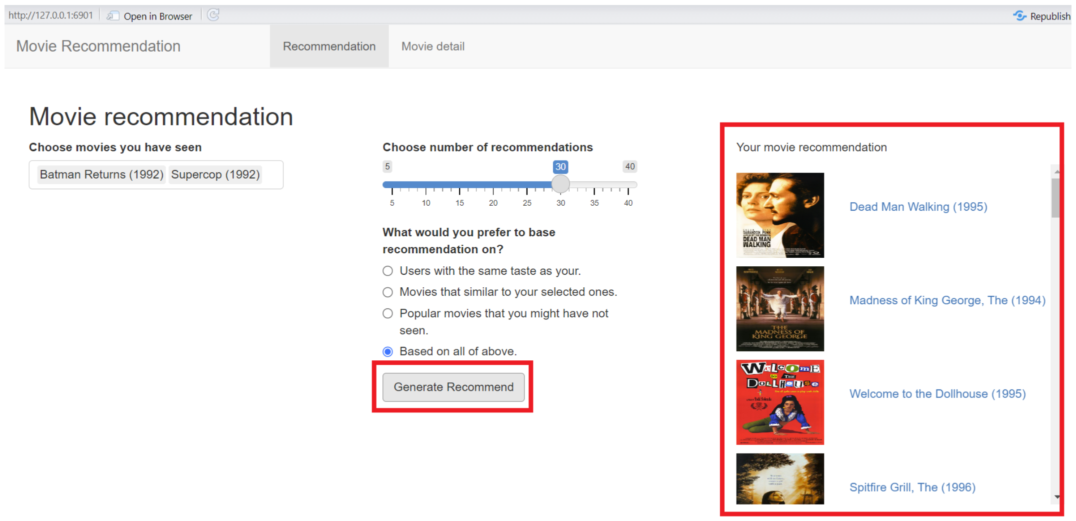
Choose number of recommendations: 30 (range 5–40)

What would you prefer to base recommendation on?

- [✓ Users with the same taste as your.]
- Movies that similar to your selected ones.
- Popular movies that you might have not seen.
- Based on all of above.

3.2.4 Generate and browse the recomendation

After finish all the steps above, users can click the button "Generate Recommend" to generate recommendation. The list of movie that are recommended will be shown on the right area as follow:


http://127.0.0.1:6901
Open in Browser
Republish
Movie Recommendation
Recommendation
Movie detail
Movie recommendation
Choose movies you have seen
Batman Returns (1992)
Supercop (1992)
Choose number of recommendations
30
What would you prefer to base recommendation on?
Users with the same taste as your.
Movies that similar to your selected ones.
Popular movies that you might have not seen.
Based on all of above.
Generate Recommend
Your movie recommendation
Dead Man Walking (1995)
Madness of King George, The (1994)
Welcome to the Dollhouse (1995)
Spitfire Grill, The (1996)

Users can click on the image or name of the movie to see the movie's details. The movie detail page will be described in the next section.

3.3 View movie details

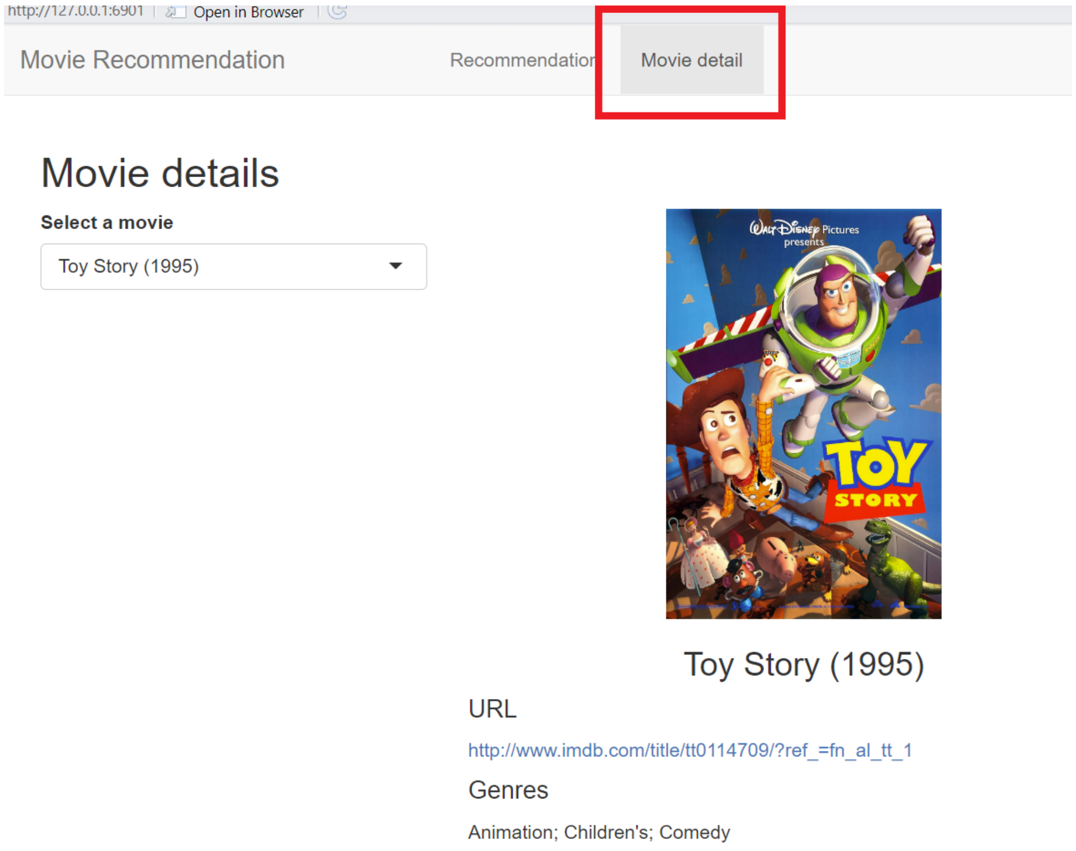
3.3.1 Navigate to movie detail page

Users can access movie detail page by one of the following method:

1. Select movie to see detail by selection menu.
2. Click on a movie poster image or name.

3.3.1.1 Using selection menu To see a movie detail information by using the selection menu, users proceed as follow:

1. Navigate to the "Movie detail" tab under the home page



Movie details

Select a movie

Toy Story (1995)

Toy Story (1995)

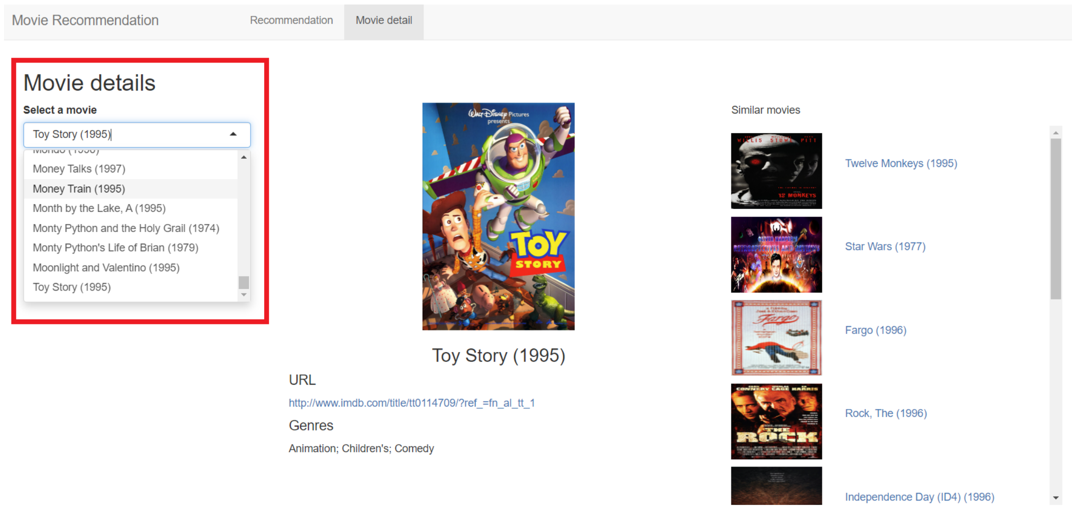
URL

http://www.imdb.com/title/tt0114709/?ref_=fn_al_tt_1

Genres

Animation; Children's; Comedy

2. Click on the selection input area under the "Select a movie" label to view the movie list.



Movie details

Select a movie

Toy Story (1995)

Money Talks (1997)
Money Train (1995)
Month by the Lake, A (1995)
Monty Python and the Holy Grail (1974)
Monty Python's Life of Brian (1979)
Moonlight and Valentino (1995)
Toy Story (1995)

Toy Story (1995)

URL

http://www.imdb.com/title/tt0114709/?ref_=fn_al_tt_1

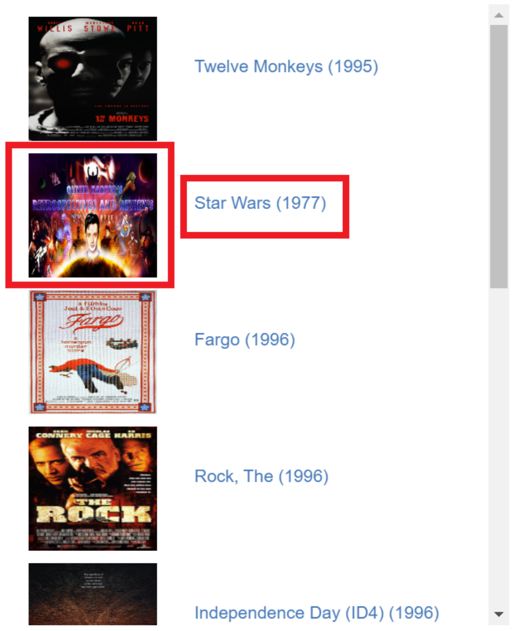
Genres

Animation; Children's; Comedy

Similar movies

Twelve Monkeys (1995)
Star Wars (1977)
Fargo (1996)
Rock, The (1996)
Independence Day (ID4) (1996)

3.3.1.2 Click on movie poster image or name Users can click on the image or name of the movie to see the movie's details.



3.3.2 Browse movie's information

The movie detail page includes:

1. Title
2. Poster image
3. IMDB url
4. Genre(s)
5. Similar movies (top 10 most similar movies generated by item-based model)

Bibliography

[1] F. M. Harper and J. A. Konstan, "The MovieLens datasets: History and context," *ACM Trans. Interact. Intell. Syst.*, vol. 5, no. 4, Dec. 2015, doi: 10.1145/2827872.

[2] P. Resnick and H. R. Varian, "Recommender systems," *Communications of the ACM*, vol. 40, no. 3, pp. 56–58, 1997.

[3] X. Su and T. M. Khoshgoftaar, "A survey of collaborative filtering techniques," *Advances in artificial intelligence*, vol. 2009, 2009.

[4] J. Wang, A. P. De Vries, and M. J. Reinders, "Unifying user-based and item-based collaborative filtering approaches by similarity fusion," in *Proceedings of the 29th annual international ACM SIGIR conference on research and development in information retrieval*, 2006, pp. 501–508.

[5] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in *Proceedings of the 10th international conference on world wide web*, 2001, pp. 285–295.

[6] J. M. Keller, M. R. Gray, and J. A. Givens, "A fuzzy k-nearest neighbor algorithm," *IEEE transactions on systems, man, and cybernetics*, no. 4, pp. 580–585, 1985.

[7] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, 2009.

[8] D. Kalman, "A singularly valuable decomposition: The SVD of a matrix," *The college mathematics journal*, vol. 27, no. 1, pp. 2–23, 1996.

[9] Z. Huang, D. Zeng, and H. Chen, "A comparison of collaborative-filtering recommendation algorithms for e-commerce," *IEEE Intelligent Systems*, vol. 22, no. 5, pp. 68–78, 2007.

[10] H. Kautz, B. Selman, and M. Shah, "Referral web: Combining social networks and collaborative filtering," *Communications of the ACM*, vol. 40, no. 3, pp. 63–65, 1997.

[11] E. Vargiu and M. Urru, "Exploiting web scraping in a collaborative filtering-based approach to web advertising." *Artif. Intell. Res.*, vol. 2, no. 1, pp. 44–54, 2013.

[12] E. Çano and M. Morisio, "Hybrid recommender systems: A systematic literature review," *Intelligent Data Analysis*, vol. 21, no. 6, pp. 1487–1524, 2017.

[13] J. S. Breese, D. Heckerman, and C. Kadie, "Empirical analysis of predictive algorithms for collaborative filtering," in *Proceedings of the fourteenth conference on uncertainty in artificial intelligence*, 1998, pp. 43–52.

[14] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in *Proceedings of the 10th international conference on world wide web*, 2001, pp. 285–295. doi: 10.1145/371920.372071.