CKME 136 – Final Project Report - Development of a Movie Recommender

Ademola Apata.
Student #: 501006476

Ryerson University

Table of Contents

Table of Figures

Introduction

Online streaming services such as Netflix, Disney plus and Amazon prime recommend movies to users based on watching patterns overtime. These systems are capable of learning and providing relevant suggestions to users based on their watching patterns. The application of recommender systems helps avoid information overload to users with far reach influences on everyday life. It is also often utilized on personalized environments to recommend items on platforms like Amazon and Google News (Bagher & Hassanpour et al., 2017).

The purpose of this project is to build a recommendation system which can suggest movies based on user preference. Some points which was taken into consideration while developing the recommender system includes the following.

The Implementation of a Collaborative recommender systems which creates and categorizes users based on similarities between profiles, behaviors in order to recommend similar products, services or content based on the group which the user belongs. The algorithm which will be implemented in this project takes account of user ratings and movie preferences and ultimately decides on a recommender.

The two main algorithms work based on the following principle:

- Item Based Collaborative Filtering Technique (IBCF): Item Based Collaborative Filtering is a model-based approach which makes predictions based on the relationship between items inferred from a ratings matrix. The concept of IBCFis that users will prefer items that are similar to the other items they like.
- User Based Collaborative Filtering Technique (UBCF): User Based Collaborative Filtering technique works on the premise that users with similar preferences will rate similarly. Thus, a missing rating from a user can be predicted by first finding the neighbor of similar users and these ratings are aggregated to forma prediction

Literature review analysis of various methods and studies to recommender systems.

Papers related to recommender systems using various filtering types and other techniques were reviewed and analyzed below is a summary of reviewed papers.

According to (Inana & Tekbacakb et al., 2018) paper on "A goal programming-based movie recommender system", this paper presents Moreopt as an enhanced recommender system which implements the combination of collaborative and content-based filtering approach. Collaborative filtering technique uses Item-based Pearson correlation between movies and user pairs, respectively. In addition, content based approach also used Item-based Pearson correlation to compute the similarity between movie pairs with the weights of features such as genre, cast, director applying linear optimization programing Language (OPL) to help overcome data sparsity problem as a result of not observing enough data to model in the corpus document. The results from the paper showed that when comparing the performance of Moreopt to other recommender systems, it was observed that the approach taken led to Moreopt outperforming traditional user-based and item-based collaborative filtering methods.

(Folajami & Isinkaye et al.) paper gives an in-depth analysis of the different characteristics and potentials of different prediction techniques in recommendation systems in order to serve as a compass for research

and practice in the field of recommendation systems. Furthermore, the paper gives a detailed breakdown of different phases involved in gathering information and preliminary analysis of data obtained for any recommender system. These include the Information collection phase, the explicit feedback phase, the implicit feedback phase, the hybrid feedback phase. Once preliminary analysis and modelling is accomplished, this paper then proposes the application of the learning phase where content-based filtering, collaborative based filtering and hybrid filtering can be applied to recommenders to predict what kind of items the user may prefer. Finally, the paper analyzes various methods to evaluate the quality of a recommendation algorithms using different types of measurements. Mean Absolute Error (MAE), Root Mean Square Error (RMSE) and Correlation are usually used as statistical accuracy metrics. Other metrics explored in the paper includes Decision support accuracy metrics that are popularly used such as Reversal rate, Weighted errors, Receiver Operating Characteristics (ROC) and Precision Recall Curve (PRC), Precision, Recall and F-measure. These metrics help users in selecting items that are of very high quality out of the available set of items.

(Uluyagmur & Cataltepe et al., 2012) paper on Content -Based Movie Recommendation Using Different Feature Sets introduces the application of content-based recommendation system with the application of various feature sets or attribute parameters such as genre, actors, directors amongst other attributes associated with a movie. The attributed weights to each feature and user are then based on attributed users' past behavior. Precision and Recall ratio and F-Measure of the system are the performance metric used to measure performance of the system.

(Li, Xu, & Wan et al., 2018) paper on Movie recommendation based on bridging movie feature and user interest proposed the implementation of a hybrid recommendation system to resolve issues of data sparsity and changes in user interest when developing recommender systems. The technique utilized involves the integration of movie features and user interests to derive the similarity between users in combination to utilizing interest vectors which is generated and regularly updated iteratively based on user movie preference. The combination of the interest vector and rating matrix of users generates a hybrid vector of users also utilized to calculate the similarity between users thereby adapting to changes in interests.

(Nilashi & Mohammad et al., 2016) paper on applying A Multi-Criteria Collaborative Filtering Recommender System Using Clustering and Regression Techniques investigates the application of Clustering techniques, Regression Trees and Expectation Maximization (EM) to improve predictive accuracy of Multi Criteria Collaborative Filtering. Principal Component Analysis(PCA) was also used for dimensionality reduction to address interdependencies among factors in multi -criteria collaborative filtering datasets. Relevancy of items and prediction accuracy and methods evaluation were carried out using precision and Mean Absolute Error (MAE). Based on the experimental results derived, the application of clustering algorithm, collaborative filtering technique and MAE together led to a significant improvement in predictive accuracy measured by standard accuracy metric mentioned earlier.

(Tewari & Singh et al., 2018) paper on Generating Top-N Items Recommendation Set Using Collaborative, Content Based Filtering and Rating Variance presents the application of collaborative and content-based filtering techniques collectively using rating variance of different items to generate more accurate recommendations. The author implements five building blocks into the models. These blocks are Profile Builder (PB) which constructs the profile of every user and item. Similarity Finder (SF) which collects keywords and vectors from the PB block. Collaborative Classifier (CC) block which uses collaborative Filtering to generate recommendations. Item Weight and Variance calculator(IWV) which facilitates finding the popularity of different items among all users in the form of weights. Final

Recommender(FR) generates targeted final recommendations for the user. The results gotten from this approach generates smaller recommendations list with more targeted items for the user. This approach also produced much higher precision factor than other benchmark recommendation method.

(Park & Pennok, 2007) paper on applying Collaborative Filtering Techniques to Movies proposes the use of a ranking system which combines recommender systems with information search tools for better search and browsing. Collaborative filtering algorithm was used to evaluate the approach both offline and online. A personalized search engine MAD6 an online research platform was built and with unique features such as web analysis which extract results from associated web engines and generate a relevance score amongst other capabilities. Based on the experimental findings, the MAD6 model seems to provide higher recall and quality in comparison to IMDB search and Yahoo search engines.

(Son & Kim, 2017) paper on Content-based filtering for recommendation systems using multi-attribute networks investigates the application of content-based filtering that uses multi-attribute (CBF-MN) networks several attributes when calculating correlation to recommend items to users. According to the study, the application of CBF-MN improves system performance overcoming the challenge of overspecialization by CBF approach because a few attributes are considered before characterizing items. The paper also details that CBF-MN also adopts a network analysis which considers relationships between all items in addition to direct and indirect relationships too.

Dataset

Two datasets which will be used for building a movie recommender system was posted on www.kaggle.com uploaded by Md Mahmud Ferdous (Kaggle, 2020). It contains dataset extracted from IMDb database, a popular movie website that majors in movies critics, movie plot descriptions, ratings ad reviews and many more aspects.

The data are contained in four files: *links.csv*, *movies.csv*, *ratings.csv* and *tags.csv*. I only use the files *movies.csv* and *ratings.csv* to build a recommendation system.

The *movies.csv*file contains 10329 observations of movies with attributes such as Movie ID:“movieid”, Title:‘title” and Movie Genre: “genres”. A summary of the dataset is shown in

	movieId <int>	title <chr>	genres <chr>
1	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
2	2	Jumanji (1995)	Adventure|Children|Fantasy
3	3	Grumpier Old Men (1995)	Comedy|Romance
4	4	Waiting to Exhale (1995)	Comedy|Drama|Romance
5	5	Father of the Bride Part II (1995)	Comedy
6	6	Heat (1995)	Action|Crime|Thriller

6 rows

Table 1.

	movieId <int>	title <chr>	genres <chr>
1	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
2	2	Jumanji (1995)	Adventure|Children|Fantasy
3	3	Grumpier Old Men (1995)	Comedy|Romance
4	4	Waiting to Exhale (1995)	Comedy|Drama|Romance
5	5	Father of the Bride Part II (1995)	Comedy
6	6	Heat (1995)	Action|Crime|Thriller

6 rows

Table 1: Movie dataset implemented for recommender system

	userId <int>	movieId <int>	rating <dbl>	timestamp <int>
1	1	16	4.0	1217897793
2	1	24	1.5	1217895807
3	1	32	4.0	1217896246
4	1	47	4.0	1217896556
5	1	50	4.0	1217896523
6	1	110	4.0	1217896150

6 rows

Table 2 below displays a summary of information on the *ratings.csv* file contains 105339 observations for different users rate multiple movies. Details of the dataset can be further described as the User ID: “userId” , Movies ID: “movieId” , Individual ratings and associated Timestamp detail.

	userId <int>	movieId <int>	rating <dbl>	timestamp <int>
1	1	16	4.0	1217897793
2	1	24	1.5	1217895807
3	1	32	4.0	1217896246
4	1	47	4.0	1217896556
5	1	50	4.0	1217896523
6	1	110	4.0	1217896150

6 rows

Table 2: Table representing each user rating for multiple movies

Table 3 below shows various attributes, sample information and description of the movie's dataset provided.

#	Attribute	Sample	Description
1	MovieID	1,2,3….10329	Discrete Variable
2	Title	Tom and Huck (1995), Golden Eye (1995)	Categorial Nominal variable
3	Genre	Adventure, Animation, Children, Comedy, Fantasy	Categorial Nominal variable

Table 3:Variable description in movies dataset

Table 4 below describes various attributes, sample information and description of the ratings dataset.

#	Attribute	Sample	Description
1	UserId	1,2,3….18,39	Discrete variable.
2	MovieId	16,1356	Discreet Variable.
3	Rating	0.5,2.64.6,3.9,7.9	Continuous variable.
4	Timestamp	1217896246, 217895786	Time stamp identifying different movies and ratings by each USERID in ratings.csv file

Table 4: Variable description in ratings dataset

Table 5 displays top 6 ranked movies and the view count in descending order with Pulp Fiction having a count of 325 followed by Forrest Gump with 311.

#	movie	View count	Title
1	296	325	Pulp Fiction (1994)
2	356	311	Forrest Gump(1994)
3	318	308	The Shawshank Redemption, (1994)
4	480	294	Jurassic Park (1993)
5	593	290	Silence of the Lambs, The (1991)
6	260	273	Star Wars: Episode IV - A New Hope (1977)

Table 5: Summary ranked Movies "movieId column" and Title with the highest number of views in the dataset in descending order.

A bar plot was also developed on R to visualize the top ranked movies.

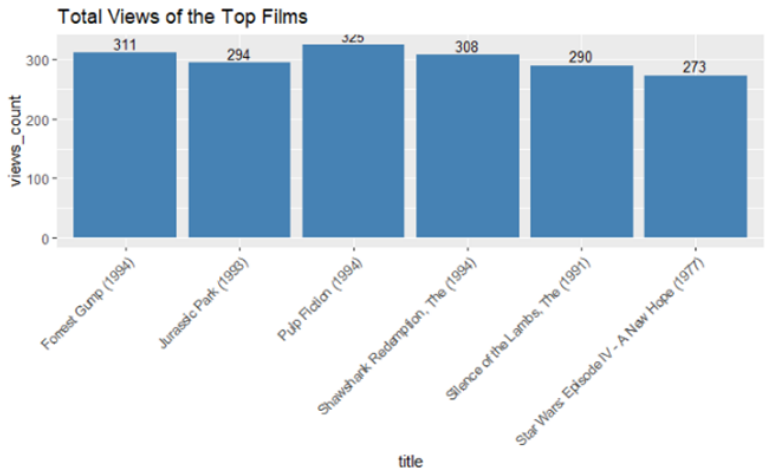


Figure 1: Bar plot of top viewed movies according to data summary shown in Table 5

Some pre-processing of the data available was required before creating the recommendation system. To accommodate this, the information on various genre type for each movie in the *movies.csv* file as shown in

	movieId <int>	title <chr>	genres <chr>
1	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
2	2	Jumanji (1995)	Adventure|Children|Fantasy
3	3	Grumpier Old Men (1995)	Comedy|Romance
4	4	Waiting to Exhale (1995)	Comedy|Drama|Romance
5	5	Father of the Bride Part II (1995)	Comedy
6	6	Heat (1995)	Action|Crime|Thriller

6 rows

Table 1 above. Information of movie genres are reorganized from a design perspective to make it much easier for users to be compared with each other from a very long list of movies available.

	movieId <int>	title <chr>	genres <chr>
1	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
2	2	Jumanji (1995)	Adventure|Children|Fantasy
3	3	Grumpier Old Men (1995)	Comedy|Romance
4	4	Waiting to Exhale (1995)	Comedy|Drama|Romance
5	5	Father of the Bride Part II (1995)	Comedy
6	6	Heat (1995)	Action|Crime|Thriller

6 rows

Table 1

Table 6 below gives a visual display of a one-hot encoding to create a matrix of corresponding genres for each movie.

movieId <int>	title <chr>	Action <int>	Adventure <int>	Animation <int>	Children <int>	Comedy <int>	Crime <int>	Documentary <int>	Drama <int>
1	1	Toy Story (1995)	0	1	1	1	1	0	0

2	2	Jumanji (1995)	0	1	0	1	0	0	0
3	3	Grumpier Old Men (1995)	0	0	0	0	1	0	0
4	4	Waiting to Exhale (1995)	0	0	0	0	1	0	0
5	5	Father of the Bride Part II (1995)	0	0	0	0	1	0	0
6	6	Heat (1995)	1	0	0	0	0	1	0

6 rows | 1-10 of 20 columns

Table 6:Visual display of Matrix encoding output based on movie genre.

Using the ggplot function in R, the distribution of the average ratings per user according to the dataset in Figure 2. Based on the histogram plot the average rating is between 3-4 at about 3.6/10.

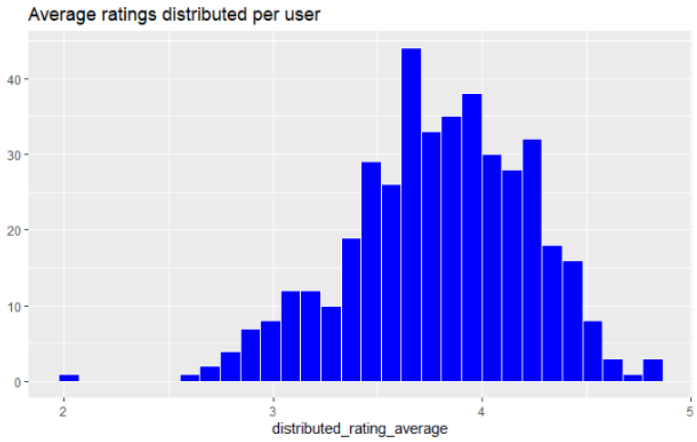
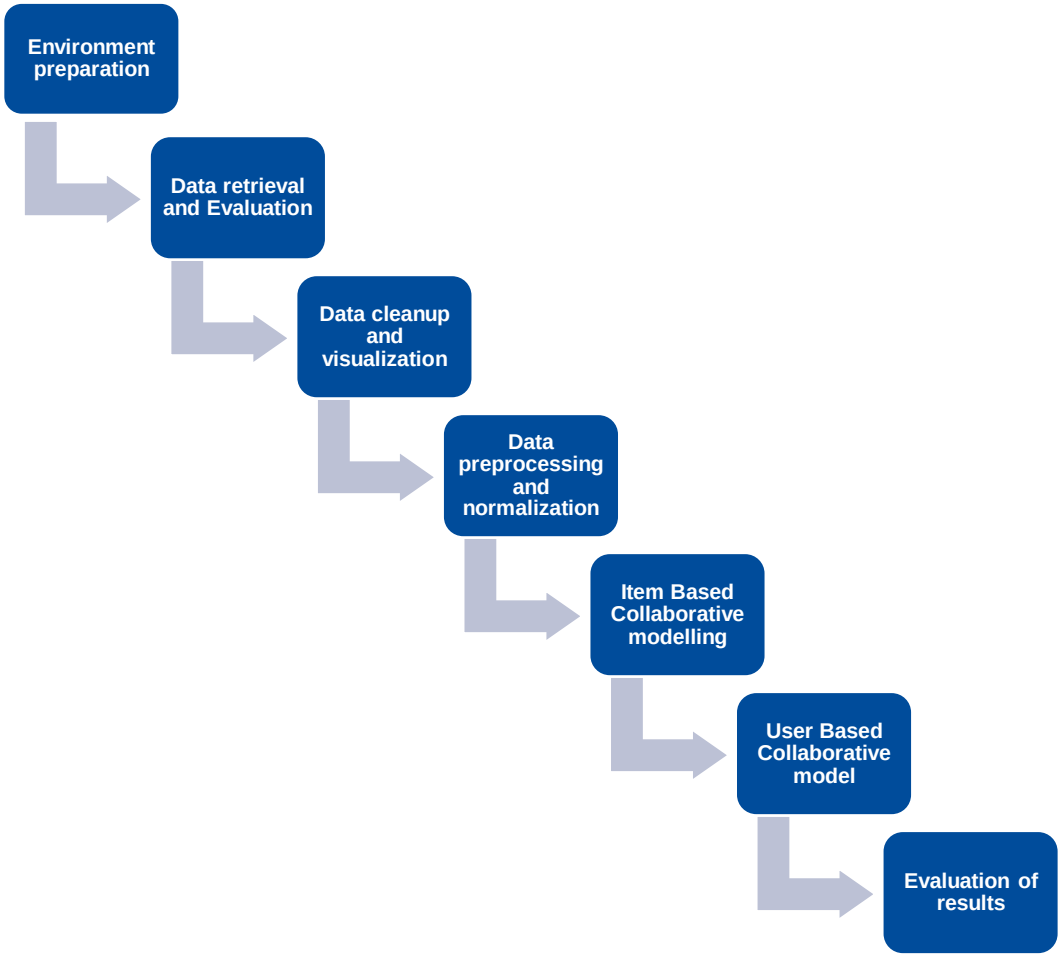


Figure 2: Distribution of the average ratings per user

Approach to developing the Movie Recommender in R



Step 1: Environment preparation

R will be the used to develop the recommender, appropriate packed installed includes "recommenderlab", "ggplot2", "data.table" and "reshape2".

Step 2: Data Retrieval and Evaluation

The dataset was retrieved from: https://www.kaggle.com/mdmahmudferdous/tutorial-how-to-build-a-movie-recommender-system/comments?select=movies.csv. Upon which it was loaded into the R environment as a data frame.

Step 3: Data Cleanup and visualization

The dataset extracted from Kaggle is quite clean, however understanding the dataset and attributes within was paramount before conducting further analysis. A brief summary of how this was executed can be viewed in Dataset section.

Step 4: Data preprocessing and normalization.

The “userId” and “movieId” column are integer variables. It is necessary to create a matrix of genres for each film which will identify which genre are associated to every movie as shown in Table 6 above.

Data preparation involves:

- Selecting useful data.
- Creation of Genre Matrix
- Normalization of dataset.

Furthermore, a minimum threshold of 50 users will be set for users who have rated a film, a similar criterion will be set for the minimum number of views per film thereby generating a list of films from those without enough views.

Step 5: Item Based Collaborative Modelling (IBCF)

The development and implementation of an IBCF similarity matrix containing all items-to-item similarity given a similarity measure. The similarity matrix is them used to recommend movies to various users based on their movie preference

Step 6: User Based collaborative Modelling

UBCF modelling can be implemented based on the principle of the neighborhood which is defined in terms of similarity between users by either taking a given number of most similar users or k-nearest neighbors or all users within a given similarity threshold.

Step 7: Evaluation of Models

The performance of each model will be evaluated at this step using measures such as True positive “TP”, False Positive “FP”, True Negative “TN”, False Negative “FN”, statistical accuracy metrics such as Root Mean Square Error “RMSE”, Mean Square Error “MAE” will be used to evaluate the deviation of recommendation from user’s specific value and finally, decision support accuracy metrices such as Receiver Operating Characteristics (ROC) and Precision Recall Curve (PRC), Precision, Recall will be used to assess the performance of IBCF and UBCF algorithms.

Methodology utilized to execute Recommender code.

The code is to be run sequentially as displayed in the "Ademola-Apata_Movie-Recommender_Final-Code_4.Rmd" file, please use this page as a reference of how to load the R Markdown package to load the Movie Recommender code.

→ Knit Set Up in R

Used Libraries

[Line:20] In this Data Science project, the following packages were used – 'recommenderlab', 'dplyr', 'ggplot2', 'data.table' and 'reshape2' was executed.

Loading Dataset into R

[Line:36 - 42] The following datasets were executed next to load into a data frame in R *movies.csv* as IMDb_movies and *ratings.csv* as IMDB_ratings data-frames.

→ Loads a summary of movies and ratings data-frames in R together with the first several first rows.

DATA PREPROCESING STEP

Extract a list of genres

[Line:54] Loads a matrix that reorganizes movie genre information for each of the films". This will allow future users search for movies they like within a specific genre.

Create a search matrix for a movie by genre

[Line:88] Load this to allow us to perform an easy search of the films by specifying its genre present in the list.

Converting ratings matrix in a proper format

[Line:97] Load this in order to use ratings data for building a recommendation engine with "recommenderlab" library package by creating a matrix of users and films and ratings of all films by each user.

Exploring Parameters of Recommendation Models

[Line:107] The next step is to load the options available for recommender model.

Similarity data between Users and Movies.

Exploring Similarity Data – Users

[Line:121] With the help of "recommenderlab", we can compute similarities between the first four users using various methods like cosine, pearson and jaccard and then visualize it as an image.

Exploring Similarity Data – Films

[Line:130] With the help of "recommenderlab", we can compute similarities between the first four films using various methods like cosine, pearson and jaccard and then visualize it as an image.

Further data exploration

[Line:139] Load to display the different ratings and frequency their values as a table in the ratings dataframe, “IMDb_ratings.csv”.

Distribution of ratings in the data frame.

[Line:150] According to the documentation, a rating equal to 0 represents a missing value, so it was removed from the dataset before visualizing the results.

Movies Visualization

Number of views of the top movies

[Line:158 & 177] Load to display section displays what the most viewed movies are and a bar plot of the total count of the top 6 viewed films.

Heatmap of Movie Ratings

[Line:201] Load this section is executed to display the “whole matrix of ratings” and “first 20 rows and columns” where each row represents users ,columns represents movies and color shade intensity represents ratings.

[Line:216] Load this section is executed to display the “whole matrix of ratings to display Heatmap of the top users and movies which identify and select the most relevant users and movies

DATA PREPARATION (Movie Ratings)

Selecting Useful Data

[Line:230] Load this to reduce the number of users and movies in the dataset based on a threshold of number of films rated and number of users who have rated a film.

Heat Map of top users and movies in the Movie rating threshold

[Line:236] Load this section selects the most relevant data based on the top 2 percent of users and movies in the new matrix of the most relevant data.

[Line:236] Load to view the Histogram plot of the average rating per user in the ratings dataframe.

Data Normalization

[Line:250] Load to view the Normalized dataset aimed at removing bias of high or low ratings provided to all films by a user.

Data Binarization and Heatmap of the top users and movies

[Line:259] Implemented make the classifier algorithm more efficient. By creating a rating threshold “0 – 3” as 0 and “3 – 5” as 1 : I define a matrix equal to 1 if the movie has been watched.

Defining Training and Test sets.

[Line:274] Displays and splits the training and test dataset parameters. The model uses 80% training set and 20% test set.

ITEM-based Collaborative Filtering Model Implementation

Building the IBCF recommendation system:

[Line:284] Load to display the rating matrix class and various parameters used in the Item based Collaborative Filter (IBCF) model.
[Line:291] Load to implement a recommender model using IBCF on the training dataset.

Applying recommender on the test set and result output

[Line:300] Load to explore the results and explore the reduced similarity matrix parameters.
[Line:309 & 317] Load to output top 10 movies in the dataset for the first user.
[Line:328] Load to view similarity matrix of the top 10 recommended movies to the first 4 users.
[Line:339] Load to view Distribution of the number of items for IBCF and the top recommended movies for each user.

User-based Collaborative Filtering Model Implementation

Building the UBCF recommendation system

[Line:360] Load to display the rating matrix class and various parameters used in the User based Collaborative Filter (UBCF) model.
[Line: 365] Load to implement a recommender model using UBCF on the training dataset.

Applying recommender on the test set and result output

[Line:374] Load to explore the results and display the distribution of items (movies) and the top recommended movies in the UBCF model.
[Line:381] Load to output top 10 movies in the dataset for the first user.
[Line:393] Load to view similarity matrix of the top 10 recommended movies to the first 4 users
[Line:403 & 412] Load to view Distribution of the number of items for IBCF and the top recommended movies for each user.

Evaluating the Recommender System

Training and Test split of the Recommender system

[Line:430, 450 & 465] Load to implement the several ways to implement the two training and testing data to evaluate the model. These include 1) splitting the data into training and test sets, 2) bootstrapping, 3) using k-fold.

Evaluating the Ratings - Choosing k-fold Cross validation technique to split data to evaluate the model

[Line:479] Load to use the k-fold cross validation approach to Evaluate the ratings.
[Line:504] Load to compute the accuracy measures for each user.

Comparing RMSEs (Root mean square errors) for both IBCF and UBCF users

Remember to change *model_to_evaluate* parameter in “Evaluate the Ratings code block” muse be changed to either IBCF or UBCF in order to measure the accuracy performance of the models.
[Line: 514] Load in order to have a performance index for the whole model.

Evaluating the recommendations

[Line: 532] Load to generate a Confusion Matrix for IBCF and UBCF to evaluate model performance depending on the number *n* of items to recommend to each user Approach. Remember to change *model_to_evaluate" parameter * in “Evaluate the Ratings code block” muse be changed to either IBCF or UBCF.

[Line:548] Load to plot the ROC and the precision/recall curve.

Comparing Models

[Line:565 & 580] Load to compare several model types in order to compare several model types and baseline them baseline model.

[Line: 600] Load to plot the ROC curves and Precision/recall curves of all multiple models.

Results

Matrix List of Genres for Each Movie

Below are Tail value of Matrix Output summary statistics of matrix genre dataframe which indicates the data type “Int” and example of each genre type.

```
##       Action Adventure Animation Children Comedy Crime Documentary Drama
## 10324      0         0         0        0      0     0           0     1
## 10325      0         0         1        1      1     0           0     0
## 10326      0         0         0        0      1     0           0     0
## 10327      0         0         0        0      1     0           0     0
## 10328      0         0         0        0      0     0           0     1
## 10329      0         0         0        0      0     0           0     0
##       Fantasy Film-Noir Horror Musical Mystery Romance Sci-Fi Thriller War
## 10324       0         0      0       0       0       0      0        0   0
## 10325       0         0      0       0       0       0      0        0   0
## 10326       0         0      0       0       0       0      0        0   0
## 10327       0         0      0       0       0       0      0        0   0
## 10328       0         0      0       0       0       0      0        0   0
## 10329       0         0      0       0       0       0      0        0   0
##       Western
## 10324       0
## 10325       0
## 10326       0
## 10327       0
## 10328       0
## 10329       0

## 'data.frame':    10329 obs. of  18 variables:
##  $ Action     : int  0 0 0 0 0 1 0 0 1 1 ...
##  $ Adventure  : int  1 1 0 0 0 0 0 1 0 1 ...
##  $ Animation  : int  1 0 0 0 0 0 0 0 0 0 ...
##  $ Children   : int  1 1 0 0 0 0 0 1 0 0 ...
##  $ Comedy     : int  1 0 1 1 1 0 1 0 0 0 ...
##  $ Crime      : int  0 0 0 0 0 1 0 0 0 0 ...
##  $ Documentary: int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Drama      : int  0 0 0 1 0 0 0 0 0 0 ...
##  $ Fantasy    : int  1 1 0 0 0 0 0 0 0 0 ...
##  $ Film-Noir  : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Horror     : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Musical    : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Mystery    : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Romance    : int  0 0 1 1 0 0 1 0 0 0 ...
##  $ Sci-Fi     : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Thriller   : int  0 0 0 0 0 1 0 0 0 1 ...
##  $ War        : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Western    : int  0 0 0 0 0 0 0 0 0 0 ...
```

Summary Statistic of Search Matrix result by genre

Below is a summary statistic of search matrix output allowing film search by specifying the genre present on the list.

	movieId	title	Action	Adventure	Animation	Children	Comedy	Crime
1	1	Toy Story (1995)	0	1	1	1	1	0
2	2	Jumanji (1995)	0	1	0	1	0	0
3	3	Grumpier Old Men (1995)	0	0	0	0	1	0
4	4	Waiting to Exhale (1995)	0	0	0	0	1	0
5	5	Father of the Bride Part II (1995)	0	0	0	0	1	0
6	6	Heat (1995)	1	0	0	0	0	1

	Documentary	Drama	Fantasy	Film-Noir	Horror	Musical	Mystery	Romance	Sci-Fi	Thriller	War	Western
1	0	0	1	0	0	0	0	0	0	0	0	0
2	0	0	1	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	1	0	0	0	0
4	0	1	0	0	0	0	0	1	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	1	0	0

```
>
```

Figure 3: summary statistic of search matrix output allowing film search by specifying the genre present on the list.

Exploring Similarity Data – Users

According to the Users similarity results, In the given matrix, each row and each column corresponds to a user, and each cell corresponds to the similarity between two users. The more red the cell is, the more similar two users are. Note that the diagonal is yellow, since it's comparing each user with itself.

```
##           1         2         3         4
## 1 0.0000000 0.9760860 0.9641723 0.9914398
## 2 0.9760860 0.0000000 0.9925732 0.9374253
## 3 0.9641723 0.9925732 0.0000000 0.9888968
## 4 0.9914398 0.9374253 0.9888968 0.0000000
```

Figure 4: Similarity between the first four users.

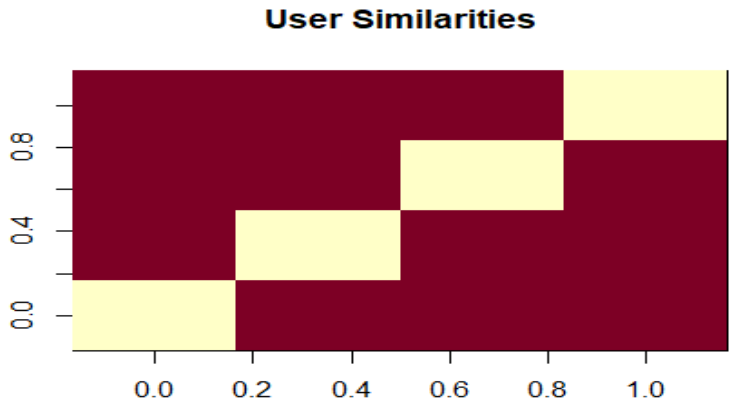


Figure 5: Similarity matrix plot of the first four users.

Exploring Similarity Data – Films

Likewise, each row and column correspond to a movie, and each cell corresponds to the similarity between two movies. The more red the cell is, the more similar two movies are. Note that the diagonal is yellow, since it's comparing each user with itself.

```
##           1         2         3         4
## 1 0.0000000 0.9669732 0.9559341 0.9101276
## 2 0.9669732 0.0000000 0.9658757 0.9412416
## 3 0.9559341 0.9658757 0.0000000 0.9864877
## 4 0.9101276 0.9412416 0.9864877 0.0000000
```

Figure 6: Similarity between the first four movies.

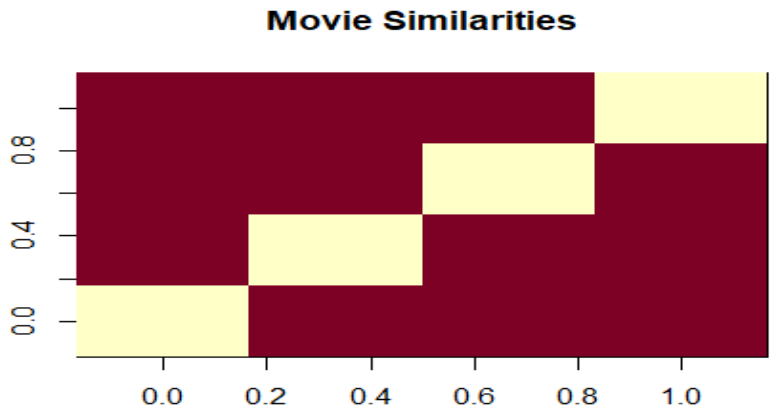


Figure 7: Similarity matrix plot of the first four movies.

Table and Distribution of the ratings

The table below shows each rating and their respective counts in an organized format

```
## ratingValues
##       0     0.5       1     1.5       2     2.5       3     3.5       4
4.5
## 6791761    1198    3258    1567    7943    5484   21729   12237   28880
8187
##       5
##   14856
```

Figure 8: Table of the different ratings and the count.

A Graphical Representation of distribution of the ratings, a rating of 0 represents the absence of a rating or missing value therefore it was removed from plot as shown below.

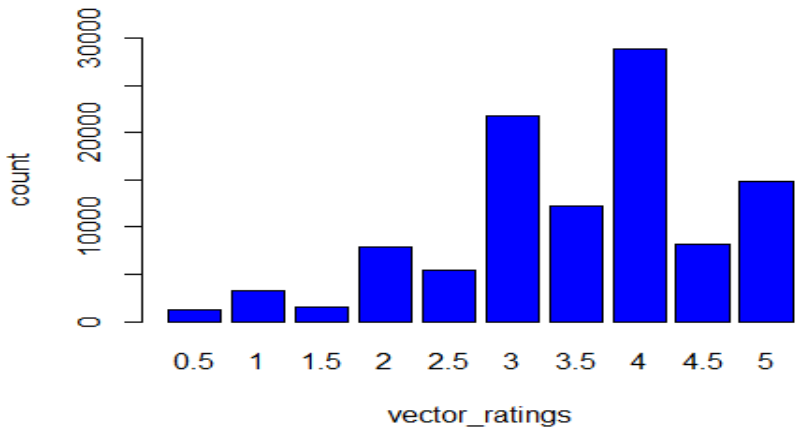


Figure 9: Plot of the different ratings and the count.

Most movies as shown in Figure 10 are rated with a score of 3 or higher. The most common rating is 4.

Number and plot of the top movies

Tabular view and bar plot of most viewed movies in the movies dataframe.

```
##     movie views_count                                        title
## 296   296         325                          Pulp Fiction (1994)
## 356   356         311                          Forrest Gump (1994)
## 318   318         308           Shawshank Redemption, The (1994)
## 480   480         294                        Jurassic Park (1993)
## 593   593         290           Silence of the Lambs, The (1991)
## 260   260         273 Star Wars: Episode IV - A New Hope (1977)
```

Figure 11: Tabular view and bar plot of most viewed movies in the movies dataframe.

We see that “Pulp Fiction (1994)” is the most viewed movie, exceeding the second-most-viewed “Forrest Gump (1994)” by 14 views.

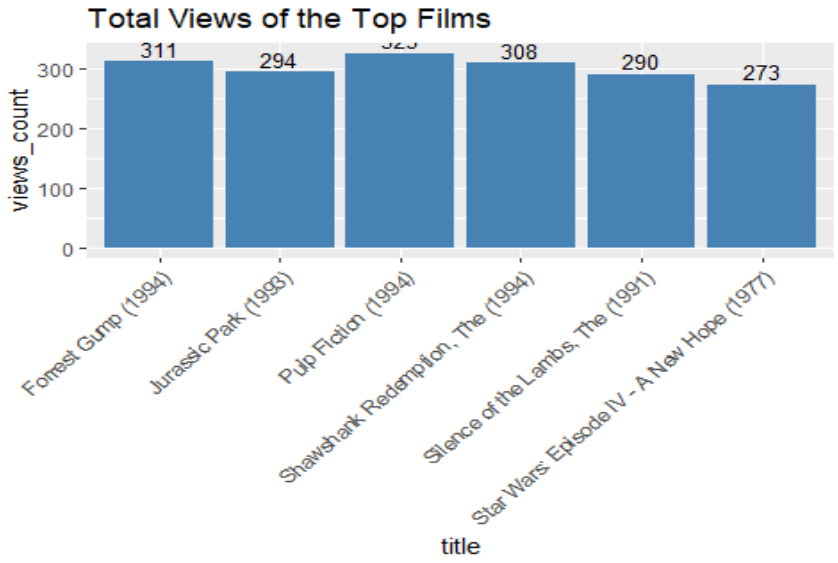


Figure 12: Bar plot of most viewed movies in the movies dataframe.

Heatmap of Movie Ratings

The table below represents a zoom in on the first 20 users -rows and movies- columns and the color intensity shade representing the intensity of the rating

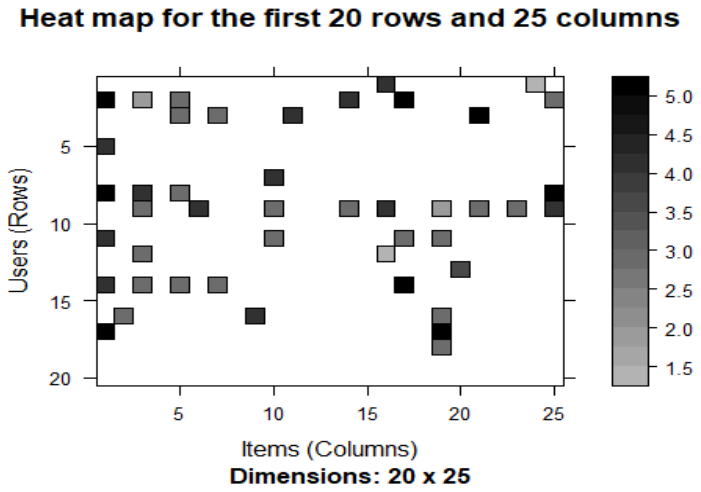


Figure 13: Heatmap of the top 20 rated movies and the users.

Data Preparation Results

The plot below indicates the distribution of the average movie ratings, the rankings are between 2 and 4.5. As expected, all extreme values are removed due to a defined threshold of 50, creating a subset of only relevant movies.

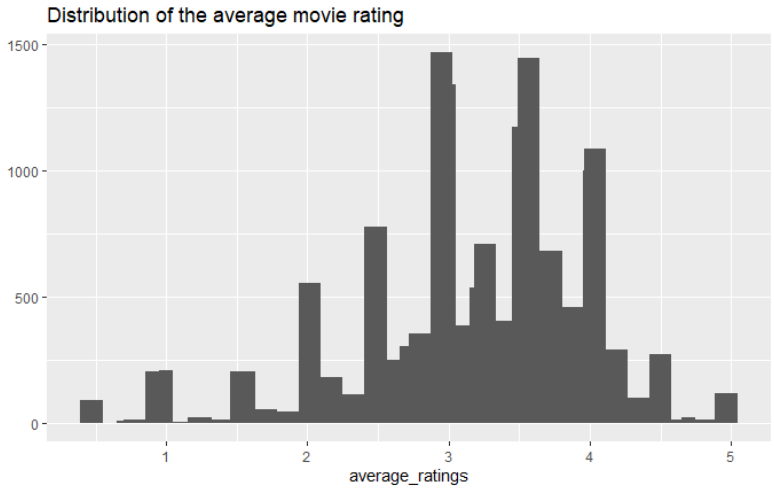


Figure 14: Distribution of average movie ratings.

Figure 15 above shows the Distribution of the average movie ratings. The highest value is around 3, and there are a few movies whose rating is either 1 or 5. Probably, the reason is that these movies received a rating from a few people only, so we shouldn't take them into account.

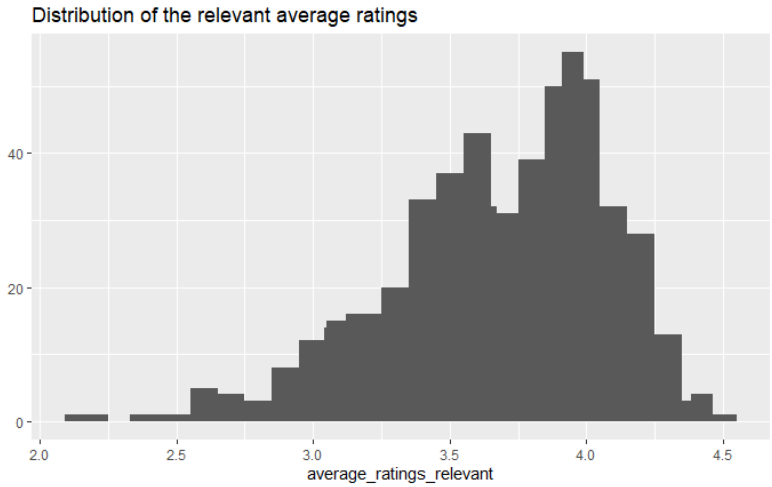


Figure 16: Distribution of relevant movie ratings.

Once movies whose number of views is below a defined threshold of 50 was removed, a subset of only relevant movies was created. Figure 17 above shows the distribution of the relevant average ratings. All the rankings are between 2.16 and 4.7. As expected, the extremes were removed. The highest value changes, and now it is around 4.

Selecting Useful Data

Upon selection of the most relevant data, the initial minimum threshold of users who have rated a film and the minimum number of views for a film was set as 50 to create a smaller dataset which minimizes data sparsity.

The heat map for the first 20 users / rows and 25 movies / columns shows the top two percent of users and movies of the relevant data

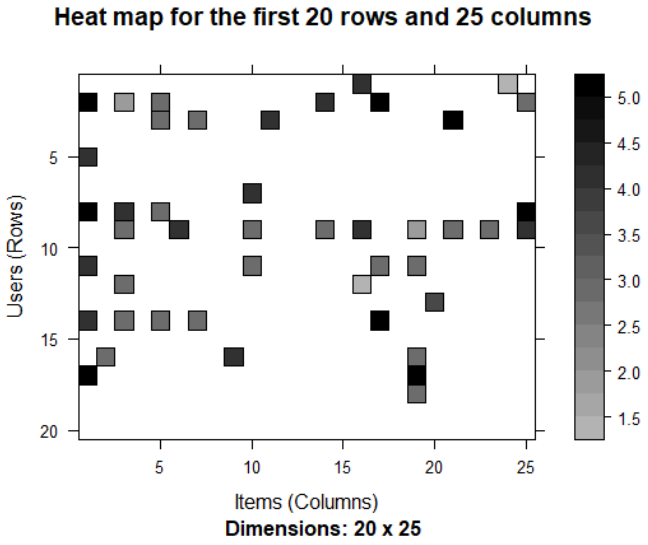


Figure 18: Heatmap of first 20 users and movies based on minimum threshold of users and movie limit.

Heatmap of Movie Rating

I visualize the whole matrix of ratings by building a heat map whose colors represent the ratings. Each row of the matrix corresponds to a user, each column to a movie, and each cell to its rating.

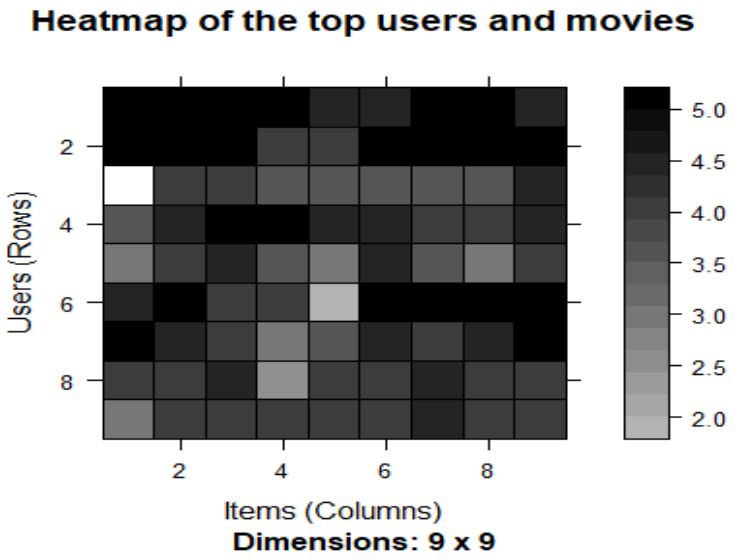


Figure 19: Heatmap of the top uses and movies in the new dataset .

In the heatmap show in Figure 20, some rows are darker than the others. This might mean that some users give higher ratings to all the movies. The distribution of the average rating per user across all the users varies a lot, as the Distribution of the average rating per user chart below shows.

Data Normalization

In order to remove the bias of high and low ratings from users, the data is normalized and the heatmap of top users and movies are shown below. The shades of blue or more red is a result of visualizing only the top movies. The average ratings by users as a result of normalization is 0 as expected. The visualized matrix for the top users is colored therefore the data is continuous as shown in Figure 21 below.

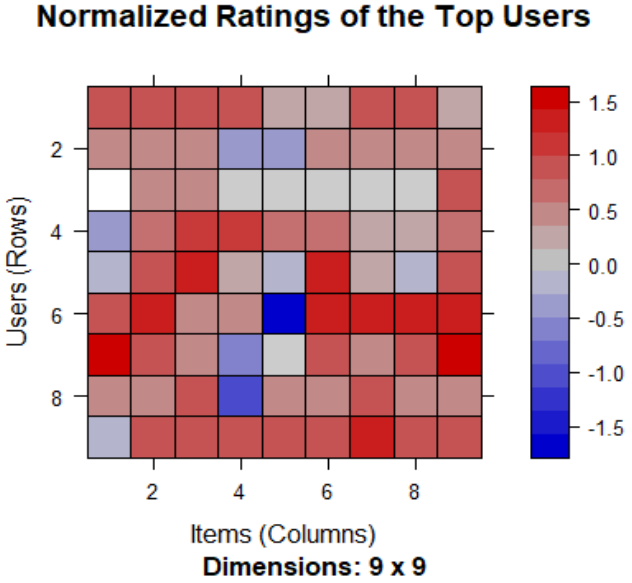


Figure 22: Normalized ratings of the top users.

Performing Data Binarization

According to the Binarized threshold defined in the code applied, we define a matrix equal to 1 if the cell has a rating above the threshold. The white cells in heatmap, shows that there are few movies with no or bad ratings as shown in.

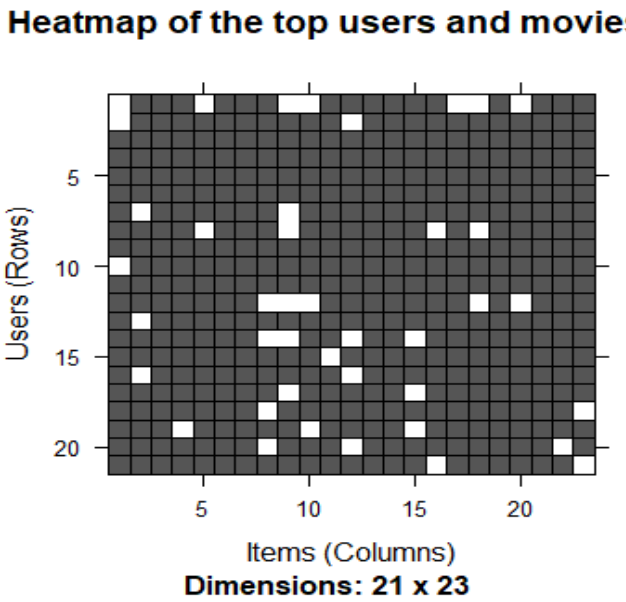


Figure 23: Heatmap of the top users and movies of binarized data.

ITEM-based Collaborative Filtering Results.

Item-based techniques Collaborative filtering technique analyze the user-item matrix to identify relationships between different items, and then use these relationships to indirectly compute recommendations for users.

The core algorithm is based on these steps:

1. For each two items, measure how similar they are in terms of having received similar ratings by similar users.
2. For each item, identify the k most similar items.
3. For each user, identify the items that are most similar to the user's purchases.

Defining Training and Test Dataset.

The model was built based on an 80% training set and 20% test set.

Result of Implementing IBCF the Recommender on various users.

```
[1] "Toy Story (1995)"              "Grumpier Old Men (1995)"
[3] "Leaving Las Vegas (1995)"      "First Knight (1995)"
[5] "Johnny Mnemonic (1995)"        "Species (1995)"
[7] "Santa Clause, The (1994)"      "Three Musketeers, The (1993)"
[9] "Dragonheart (1996)"            "Top Gun (1986)"
```

Figure 24: The results of the IBCF recommender for the top 10 movies is shown below for the first user.

```
 [1] "Mortal Kombat (1995)"
 [2] "Rumble in the Bronx (Hont faan kui) (1995)"
 [3] "Bad Boys (1995)"
 [4] "Crimson Tide (1995)"
 [5] "Client, The (1994)"
 [6] "Crow, The (1994)"
 [7] "Flintstones, The (1994)"
 [8] "Maverick (1994)"
 [9] "Naked Gun 33 1/3: The Final Insult (1994)"
[10] "Speed (1994)"
```

Figure 25: The results of the IBCF recommender for the top 10 movies is shown below for the second user.

```
[1] "It's a Wonderful Life (1946)"    "Cast Away (2000)"
[3] "Coneheads (1993)"                "Dogma (1999)"
[5] "Dark City (1998)"                "Pleasantville (1998)"
[7] "Godfather: Part III, The (1990)" "French Kiss (1995)"
[9] "Godfather: Part II, The (1974)"  "In the Line of Fire (1993)"
```

Figure 26: The results of the recommender for the top 10 movies is shown below for the third user.

It's also possible to define a matrix with the recommendations to each user. Below is a visualisation of top 10 movie Id for the first four users:

```
[1] 10 83
      [,1] [,2] [,3] [,4]
 [1,]    1   44  953    6
 [2,]    3  112 4022    7
 [3,]   25  145  435   62
 [4,]  168  161 3052  141
 [5,]  172  350 1748  151
 [6,]  196  353 2321  163
 [7,]  317  355 2023  196
 [8,]  552  368  236  236
 [9,]  653  370 1221  261
[10,] 1101  377  474  292
```

Figure 27: visualization of IBCF similarity matrix top 10 movie Id for the first four users.

The columns represent the first four users and the rows represents the movieId values for 10 movies to each user.
Based on IBCF Model implemented to the matrix defined for users, a few movies have been recommended more than 8 times as shown below:

	Movie title <chr>	No of items <fctr>
3	Grumpier Old Men (1995)	14
7	Sabrina (1995)	10
36	Dead Man Walking (1995)	10
21	Get Shorty (1995)	9

4 rows

Figure 28: visualization of the top viewed movies using IBCF recommender.

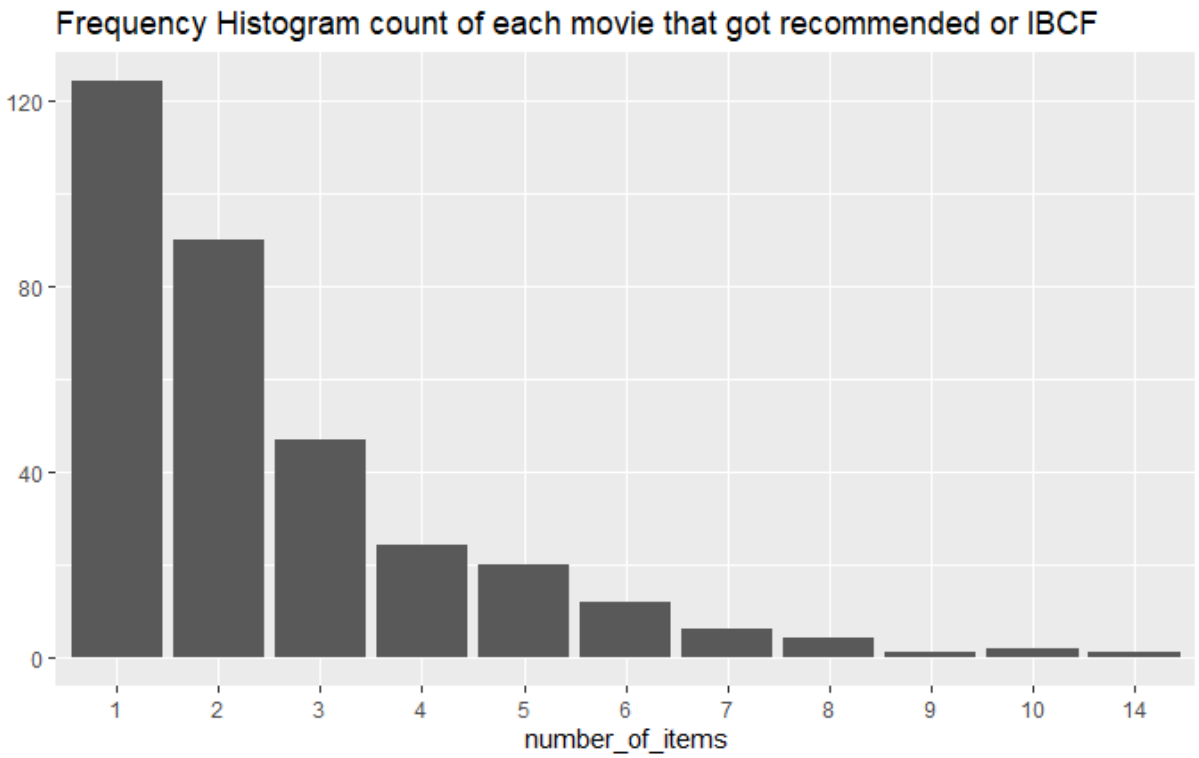


Figure 29:IBCF Frequency Histogram count of each movie that got recommended.

how many times each movie got recommended and build the related frequency histogram:

User-based Collaborative Filtering Results.

User Based Collaborative Filtering Approach works based on identifying similar users when a new user is given. Concurrently the top users are then identified,

The following steps were implemented in the model.

1. Measure the similarity of users to the new one
2. Identify the most similar users.
    a. Used the top (k-nearest neighbor approach)
    b. Defined a threshold in which users with similarities are identified.
3. Rate the movies by the most similar users
4. The top-rated movies are chosen

Result of Implementing UBCF the Recommender on various users.

The results of the recommender for the top 10 movies is shown below for the first user:

```
[1] "Cliffhanger (1993)"                    "Lost World: Jurassic Park, The
(1997)"
[3] "Devil's Advocate, The (1997)"          "Talented Mr. Ripley, The (1999)"
[5] "Animal House (1978)"                   "Alien (1979)"
[7] "Untouchables, The (1987)"              "Total Recall (1990)"
[9] "Sleepy Hollow (1999)"                  "Cape Fear (1991)"
```

Figure 30: The results of the UBCF recommender for the top 10 movies is shown below for the first user.

The results of the recommender for the top 10 movies is shown below for the second user:

```
[1] "Jungle Book, The (1967)"      "It's a Wonderful Life (1946)"
[3] "Heat (1995)"                  "Sneakers (1992)"
[5] "Boogie Nights (1997)"         "Traffic (2000)"
[7] "Demolition Man (1993)"        "Outbreak (1995)"
[9] "Chinatown (1974)"             "Vertigo (1958)"
```

Figure 31: The results of the UBCF recommender for the top 10 movies is shown below for the second user.

The results of the recommender for the top 10 movies is shown below for the third user:

```
[1] "Citizen Kane (1941)"        "Animal House (1978)"        "Juno (2007)"
[4] "Lost in Translation (2003)" "Boogie Nights (1997)"       "Amadeus
(1984)"
[7] "Chinatown (1974)"           "Few Good Men, A (1992)"     "Dragonheart
(1996)"
[10] "Phenomenon (1996)"
```

Figure 32: The results of the UBCF recommender for the top 10 movies is shown below for the tjird user.

The UBCF matrix below contains movieId of each recommended movie (rows) for the first four users (columns) in our test dataset.

```
      [,1] [,2]  [,3] [,4]
 [1,]  434 2078   923 1250
 [2,] 1544  953  3421  553
 [3,] 1645    6 56367  805
 [4,] 3176 1396  6711 1276
 [5,] 3421 1673  1673 1204
 [6,] 1214 4034  1225 7438
 [7,] 2194  442  1252 1252
 [8,] 2916  292  2268 2268
 [9,] 3081 1252   653 2640
[10,] 1343  903   802 2918
```

Figure 33: visualization of UBCF similarity matrix top 10 movie Id for the first four users.

In comparison to IBCF Model implemented, movies have been recommended more frequently to users with " 2268 - Few Good Men, A (1992)" recommended 21 times as compared to the most compared IBCF movie in "168 - First Knight (1995)" recommended just 12 times.

	Movie title <chr>	No of items <fctr>
2268	Few Good Men, A (1992)	21
3421	Animal House (1978)	19
5669	Bowling for Columbine (2002)	19
953	It's a Wonderful Life (1946)	13

4 rows

Figure 34: visualization of the top viewed movies using UBCF recommender.

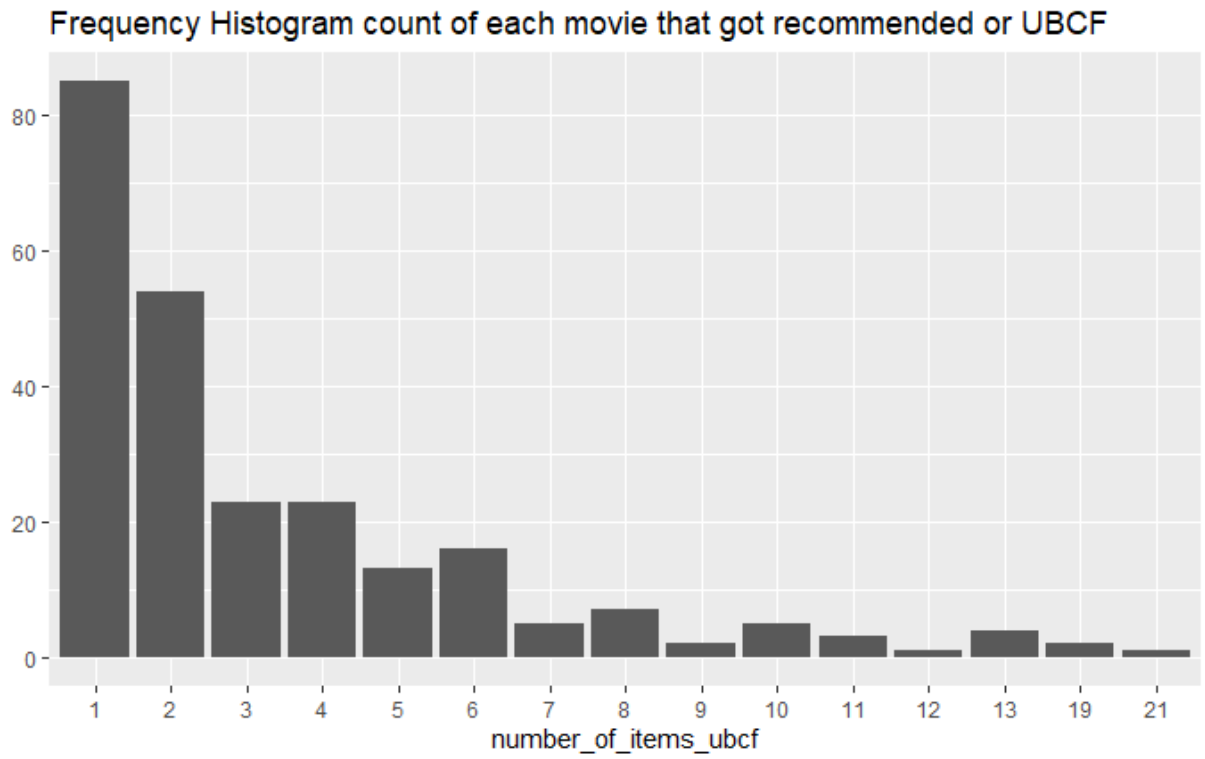


Figure 35:UBCF Frequency Histogram count of each movie that got recommended.

Compared with the IBCF, the distribution has a longer tail. This means that there are some movies that are recommended much more often than the others. The maximum is more than 20, compared to about 10 for IBCF.

Evaluation Metrics for IBCF and UBCF Recommenders.

The result is an object of class Evaluation Result which contains several confusion matrices. getConfusionMatrix() will return the confusion matrices for the 4 runs (we used 4-fold cross evaluation) as a list. In the following we look at the first element of the list which represents the first of the 4 runs as shown below.
Before performing the comparative analysis of the recommenders, it is important to define the parameters which will used to analyze the Recommenders.

The primary performance matrix of the confusion matrix results include TP, FP, FN and TN are the entries true positives, false positives, false negatives and true negatives, other precomputed performance measurement values generated by the confusion matrix also includes TPR, FPR which is the entries for true positive rate, false positive rate, precision and recall.
Metrics for measuring the accuracy of recommendation filtering systems are divided into statistical and decision support accuracy metrics (Folajami & Isinkaye et al.)

Statistical accuracy metrics evaluate accuracy of a filtering technique by comparing the predicted ratings directly with the actual user rating. Mean Absolute Error (MAE), Root Mean Square Error (RMSE) and Correlation are usually used as statistical accuracy metrics. MAE is the most popular and commonly used;

it is a measure of deviation of recommendation from users' specific value (Folajami & Isinkaye et al.), a general guiding summary of RMSE and MAE metrics are.

- → Root Mean Square Error (RMSE) puts more emphasis on larger absolute error and the lower the RMSE is, the better the recommendation accuracy.
- → Lower Mean Absolute Error (MAE) means more accurate the recommendation engine prediction of user ratings.

Decision support accuracy metrics that are popularly used are Reversal rate, Weighted errors, Receiver Operating Characteristics (ROC) and Precision Recall Curve (PRC), Precision, Recall and F-measure. These metrics help users in selecting items that are of very high quality out of the available set of items

- → ROC curves are very successful when performing comprehensive assessments of the performance of some specific algorithms.
- → Precision is the fraction of recommended items that is relevant to the user: $\text{Correctly recommended items} / \text{Total recommended items}$
- → Recall can be defined as the fraction of relevant items that are also part of the set of recommended items: $\text{Correctly recommended items} / \text{Total useful recommended items}$

IBCF & UBCF performance using TP, TN, FN and TN Evaluation performance

The function which evaluates the recommender performance depends on the number *n* of items to recommend to each user, an increment from 10 to a 100 items (movies) to recommend will be analyzed for each movie in other to measure the TP, FP, FN, TN parameters defined earlier on.

```
IBCF run fold/sample [model time/prediction time]
         1  [0.98sec/0.24sec]
         2  [0.9sec/0.06sec]
         3  [0.79sec/0.08sec]
         4  [0.76sec/0.08sec]
      TP        FP       FN       TN precision     recall        TPR        FPR
10  1.466667  8.247619 64.80952 367.4762 0.1509804 0.02028600 0.02028600 0.02213783
20  2.838095 16.571429 63.43810 359.1524 0.1464597 0.03830930 0.03830930 0.04422261
30  4.219048 24.752381 62.05714 350.9714 0.1473353 0.05644117 0.05644117 0.06586345
40  5.504762 32.657143 60.77143 343.0667 0.1475627 0.07623389 0.07623389 0.08665240
50  6.638095 40.171429 59.63810 335.5524 0.1469088 0.09219699 0.09219699 0.10657674
60  7.742857 47.152381 58.53333 328.5714 0.1477416 0.11082734 0.11082734 0.12515644
70  8.847619 53.371429 57.42857 322.3524 0.1499211 0.12815858 0.12815858 0.14187549
80  9.790476 58.847619 56.48571 316.8762 0.1514083 0.14458264 0.14458264 0.15635174
90 10.647619 63.809524 55.62857 311.9143 0.1526064 0.15707964 0.15707964 0.16934538
100 11.200000 68.676190 55.07619 307.0476 0.1511938 0.16431753 0.16431753 0.18207174
```

Figure 36:IBCF model performance evaluation result .

```
UBCF run fold/sample [model time/prediction time]
         1  [0.03sec/1.61sec]
         2  [0sec/0.81sec]
         3  [0sec/0.79sec]
         4  [0.01sec/0.81sec]
            TP        FP       FN       TN precision     recall        TPR        FPR
10    1.419048  8.580952 70.10476 361.8952 0.1419048 0.01843235 0.01843235 0.02331262
20    2.952381 17.047619 68.57143 353.4286 0.1476190 0.03882662 0.03882662 0.04626236
30    4.876190 25.123810 66.64762 345.3524 0.1625397 0.06891812 0.06891812 0.06823857
40    6.666667 33.333333 64.85714 337.1429 0.1666667 0.09242064 0.09242064 0.09022449
50    8.419048 41.580952 63.10476 328.8952 0.1683810 0.11415073 0.11415073 0.11236851
60   10.666667 49.333333 60.85714 321.1429 0.1777778 0.14781073 0.14781073 0.13305570
70   12.771429 57.228571 58.75238 313.2476 0.1824490 0.17773134 0.17773134 0.15435736
80   14.685714 65.314286 56.83810 305.1619 0.1835714 0.20464941 0.20464941 0.17626134
90   16.466667 73.533333 55.05714 296.9429 0.1829630 0.22741423 0.22741423 0.19841445
100  18.390476 81.609524 53.13333 288.8667 0.1839048 0.25749655 0.25749655 0.22014827
```

Figure 37:UBCF model performance evaluation result .

The primary point when analyzing the results are the low values of the confusion matrix, this is as a result of the limited amount of data present when developing both recommenders also known as data sparsity causing leading to limited accuracy.

Both IBCF and UBCF models display similar values in regard to the TP, FP,FN and TN performance metric, however IBCF has a much faster prediction time than UBCF which is to be expected because UBCF utilizes information from the dataset itself while IBCF utilises information from a set threshold for the minimum number of users who have rated a film as 50. This is also same for minimum number of views that are per film. This way, we have filtered a list of watched films from least-watched ones.

However, upon investigating the precision and recall values, UBCF performs better than IBCF with higher values for increments of 10 to a 100 items (movies).

Comparing the RMSE, MSE & MAE of both IBCF and UBCF

IBCF

Evaluating the IBCF model performance for the first 6 users.

```
         RMSE      MSE       MAE
[1,] 2.121320 4.500000 1.6666667
[2,] 1.110555 1.233333 0.8666667
[3,] 1.211060 1.466667 0.9333333
[4,] 1.172604 1.375000 0.8750000
[5,] 1.732051 3.000001 1.1430720
[6,] 2.915476 8.500000 2.3333333
```

Overall IBCF Model performance

```
    RMSE      MSE      MAE
1.372488 1.883722 1.008664
```

UBCF

Evaluating the UBCF model performance for the first 6 users.

```
          RMSE       MSE       MAE
[1,] 0.8991713 0.8085091 0.6842119
[2,] 0.9188139 0.8442190 0.6326462
[3,] 0.7692606 0.5917619 0.6477510
[4,] 0.7147985 0.5109368 0.5147303
[5,] 1.3017383 1.6945227 1.0880761
[6,] 0.9997881 0.9995762 0.8251448
```

Overall UBCF Model performance

```
     RMSE       MSE       MAE
1.0316689 1.0643408 0.7943234
```

Figure 38: Figure comparing the RMSE, MSE & MAE of both IBCF and UBC.

As mentioned before, Root Mean Square Error (RMSE) puts more emphasis on larger absolute error and the lower the RMSE is, the better the recommendation accuracy, also the lower the MAE, the more accurately the recommendation engine predicts user ratings. Therefore, UBCF is the better performing algorithm when comparing both individual RMSE and MAE values for each user in the recommender and the overall model performance for each recommender.

Identifying the most suitable model

A possible way to compare the efficiency of two systems is by comparing the size of the area under the Receiver Operative Characteristic (ROC)-curve, where a bigger area indicates better performance (Hahsler, 2020). A good performance index is the area under the curve, that is, the area under the ROC curve. The shows that the highest is UBCF with cosine distance, so it's the best-performing technique. The UBCF with cosine distance is still the top model as show in fig.

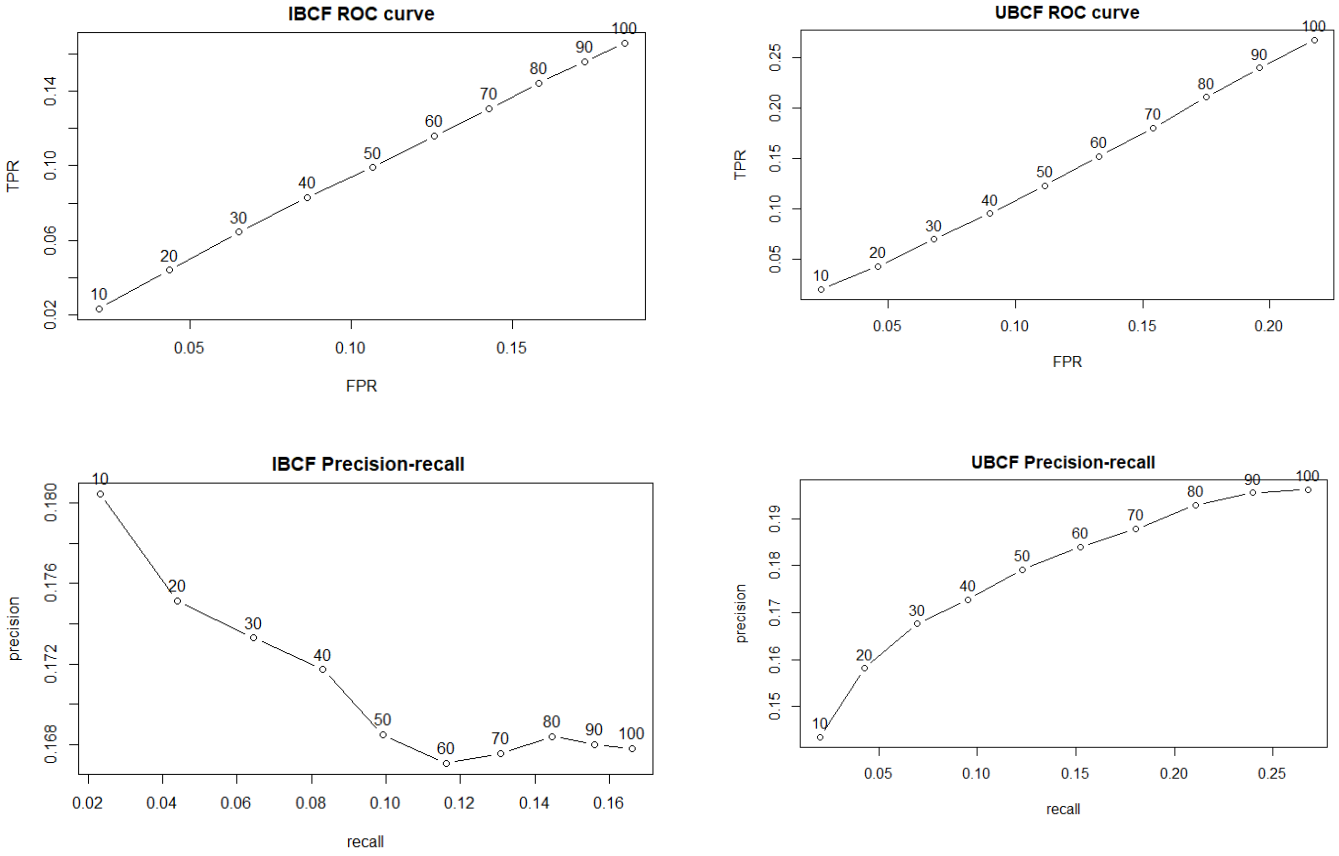


Figure 39: Comparing the Roc Curve and Precision- recall curve for different probability tresholds (IBCF and UBCF) .

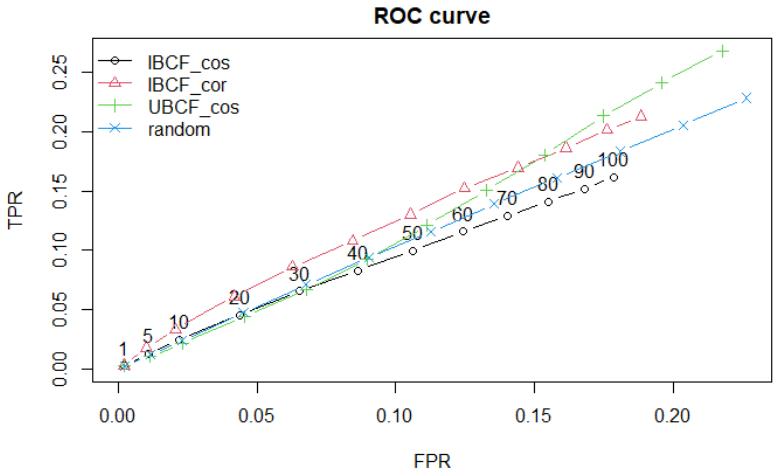


Figure 40: Comparing the Roc Curve for different models using different method parameters.

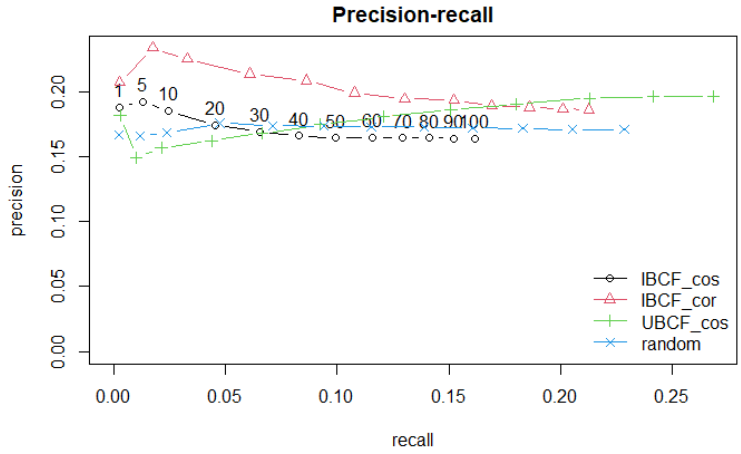


Figure 41: Comparing Precision-recall curve for different models using different method parameters.

Depending on what is the main purpose of the system is, an appropriate number of items to recommend should be defined in order to decide which recommender to choose.

Conclusion and Discussion

This project accomplished the development and evaluation of collaborative based filtering systems (CFR) for recommending movies with cosine method. Among all the various recommenders evaluated, the UBCF was the recommender of choice. Benefit of UBCF isn't that it gives recommendations that can be complements to the item the user was interacting with. This might be a stronger recommendation than what an item-based recommender can provide as users might not be looking for direct substitutes to a movie they had just viewed or previously watched. A weakness of a UBCF is a type of memory-based collaborative filtering that uses all user data in the dataframe to create recommendations. Therefore, comparing the pairwise correlation of every user in the dataset is not scalable. If there were millions of users, this computation would be very time consuming therefore, comparing the pairwise correlation of every user in the dataset is not scalable. This can be resolved using some form of dimensionality reduction, such as Principal Component Analysis. Since user-based collaborative filtering technique uses users' past choices to make a prediction. User choices can change over time making it more difficult to precompute user-similarities

Bibliography

Badrul, S., Karypis, G., & Karypis et al., G. (2001). Item-Based Collaborative Filtering Recommendation. *GroupLens Research Group/Army HPC Research Center*, 285-295.

Bagher , C. R., & Hassanpour et al., H. (2017). User trends modeling for a content-based recommender system. *Expert Systems With Applications*, 209-219. doi:10.1016/j.eswa.2017.06.020

Folajami, Y. O., & Isinkaye et al., F. O. (n.d.). Recommendation systems: Principles, methods and evaluation.

Hahsler, M. (2020). *recommenderlab: A Framework for Developing and Testing Recommendation Algorithms.* Southern Methodist University, Lyle School of Engineering.

Inana, E., & Tekbacakb et al., F. (2018). Moreopt: A goal programming based movie recommender systemEmrah. *Journal of Computational Science*, 43-50. doi:10.1016

Kaggle. (2020, June 06). *Movie Recommender System dataset*. (M. Ferdous, Editor) Retrieved from kaggle: https://www.kaggle.com/mdmahmudferdous/tutorial-how-to-build-a-movie-recommender-system/comments?select=movies.csv

Li, J., Xu, W., & Wan et al., W. (2018). Movie recommendation based on bridging movie feature and userinterest. *Journal of Computational Science*, 128–134. doi:1877-7503

Nilashi, M., & Mohammad et al., D.-E. (2016). A Multi-Criteria Collaborative Filtering Recommender System Using Clustering and Regression Techniques. *Journal of Soft Computing and Decision Support Systems*, 24-30. doi:E-ISSN: 2289-8603

Park, S.-T., & Pennok, D. M. (2007). Applying Collaborative Filtering Techniques to Movie. *Yahoo! Research*, 550-559. doi:ACM 978-1-59593-609-7/07/0008

Son, J., & Kim, S. B. (2017). Content-based filtering for recommendation systems using multiattribute networks. *Expert Systems With Applications*, 404-412. doi:10.1016/j.eswa.2017.08.008

Tewari, A. S., & Singh et al., P. J. (2018). Generating Top-N Items Recommendation Set Using Collaborative, Content Based Filtering and Rating Variance. *International Conference on Computational Intelligence and Data Science (ICCIDS 2018)*, 1678–1684. doi:10.1016/j.procs.2018.05.139

Uluyagmur, M., & Cataltepe et al., Z. (2012). Content -Based Movie Recommendation Using Different Feature Sets. *Proceedings of the World Congress on Engineering and Computer Science, 1*. doi:2078-0966