DS 5110- Project

MOVIE RECOMMENDATION SYSTEM

Table of Contents

Contents

Introduction

We use sophisticated techniques to build a personalized movie recommendation system based on the user's previous movie ratings and similar users' movie interests. Different people have different tastes in movies, and this is not reflected in a single score that we see when a movie is looked up on the internet. Our system helps users instantly discover movies to their liking, regardless of how distinct their tastes may be. Current recommender systems generally fall into two categories: content-based (item) filtering and user-based collaborative filtering. We experiment with both approaches in our project.

The objective is to recommend movies to users at two levels: first, based on their past rating history and second, based on impromptu genre-movie selections. The first prediction model would take into consideration the rating history of all customers, and having understood the watching patterns of similar users, recommend the user a movie he/she may be interested in. The second model would take genre as input and 3 movies he/she likes in that genre, thereafter recommend movies like the movies in the selected genre.

Methods

This section includes an overview of the main steps involved in this project,

- The first step is to formulate/clean the data. It includes the treatment of null values (if any) in the data
- Once data is pre-processed, obtain insights to have a direction in which the data can be modeled
- Finally, use appropriate models to fit into the data for proper recommendations

The below sections delve into each of the steps in detail

Pre-Processing:

Real-life data typically needs to be preprocessed (e.g. cleansed, filtered, transformed) for analysis and machine learning techniques to be used. In this section, we will focus mainly on data preprocessing techniques that were important while designing the recommender system.

We used the **MovieLens** [1] movie rating dataset for our project. The data consists of the ~ 27 Million user ratings across ~53.9K movies (20 genres) collected from 1995-2018. It contains 6 CSV files containing attributes about movies, ratings, tags of user ratings (3 files), and movie links. The entity-relationship diagram is as follows,

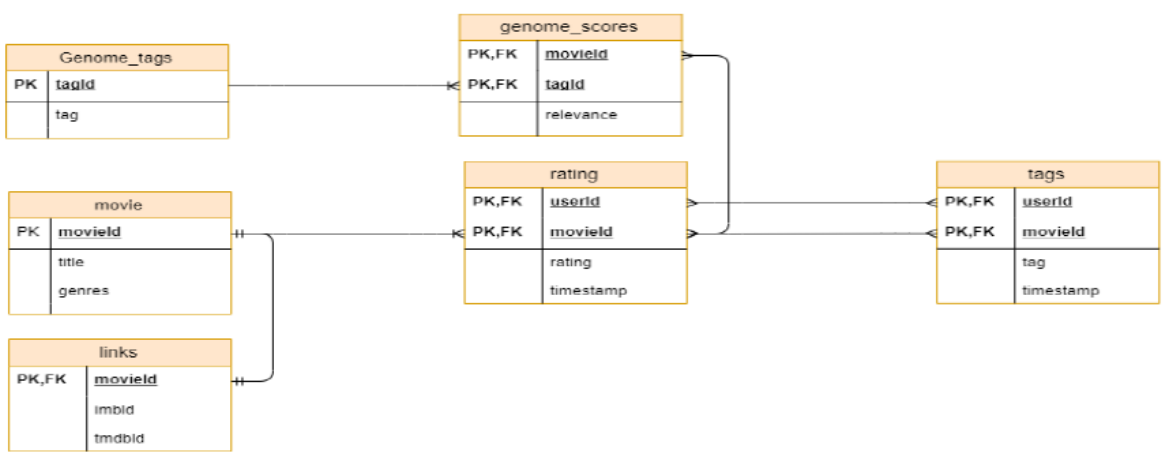

Fig 1

We only use the ratings and movies data as the tags data wasn't relevant for generating recommendations. This was the case as the tags were not metadata about the movies but text summarizing user comments. Thus, the ratings (numeric) provides us a better understanding of user preference than the tags, which were ambiguous.

The movies file has combined genre values, for example; "Action|Comedy|Drama". This makes sense for a movie that belongs to all 3 genres. This is difficult to model and hence must be separated into different rows for easier modeling. With the use of "*splitstackshape*" [2] function, we segregated data into different rows based on a delimiter (in this case "|"). Along with mixed values, there are certain null values in the genres column. This was resolved by merging null-values with "no-genres-listed".

Finally, we can move ahead and explore the dataset at hand by visualizing it to obtain insights. Since we are unaware of the distribution of data, we will evaluate every feature to gain some perspective.

Exploratory Data Analysis:

Fig 1 shows the relationship between genre and ratings. This graph depicts the average ratings for each genre. From the graph, the horror genre has the lowest rating whereas film-noir has the highest average rating.

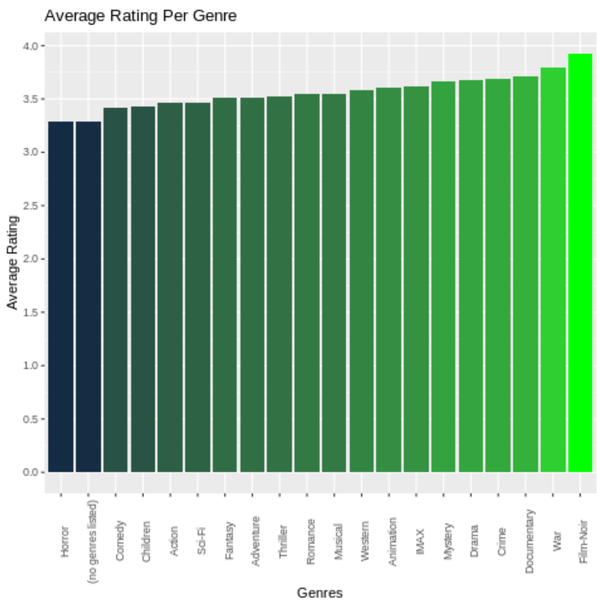

Fig 2: Average Rating v/s Genres

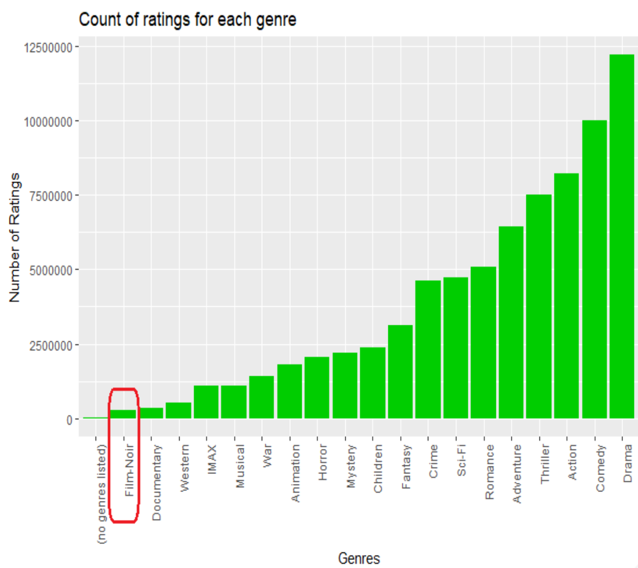

Fig 3: Count of Ratings for each genre

Fig 3 represents the count of the number of ratings for each genre. We observe that more common genres like Drama, Comedy, and action are rated more frequently when compared to the *film-noir* genre. This means that data is skewed as the film-noir genre has a very small sample size of users' rating, thus the average rating for this genre looks high. Thus, we can't recommend movies based solely on the average rating for a genre. We need to find the relationship between each genre so that the distribution of the number of ratings for every genre is uniform.

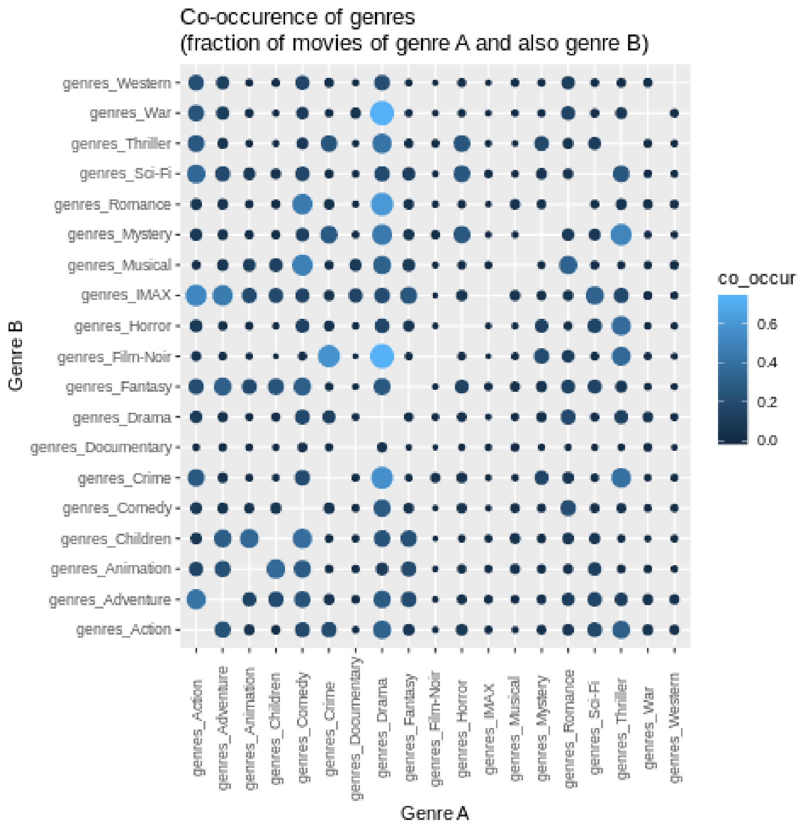

Fig 4: Co-occurrence of Genres

Here, the plot of co-occurrence will help in finding the relation between each genre. The size of dots represents the fraction of movies that have both genres listed. Bigger the size of dots, higher are the movies where the genres co-exist. As we can see from graph 4, film-noir and Drama genre have a big bubble size. This means that there are many movies with film-noir and drama as their genre. We can also see that there are many other genres co-existing with drama. Hence, we don't have to skew data for film-noir.

Collaborative Filtering Algorithms

Collaborative filtering [3] algorithms are mainly divided into two main categories – Memory-based (user-based) and Model-based (item-based) algorithm [4].

Item-based Collaborative Filtering Algorithm

Item-based collaborative filtering [5] [6] is a model-based collaborative filtering algorithm for producing predictions for users. It looks for items that are like the articles that the user has already rated and recommend most similar articles.

Algorithm

1. The first step is creating a document-term matrix with users as rows and movies as columns and ratings as values. The similarity is found between two movies (column) using only pairwise complete observations.

Movies →

Users ↑

	Three Colors: Blue (Trois couleurs: Bleu) (1993)	Kalifornia (1993)	Weekend at Bernie's (1989)	Better Off Dead... (1985)	Waiting for Guffman (1996)	Event Horizon (1997)	Spawn (1997)	Weird Science (1985)	¡Three Amigos! (1986)	Stigmata (1999)	RoboCop 2 (1990)	Falling Down (1993)
1	3.5	3.5	1.5	4.5	4.5	2.5	1.5	4.5	4	3.0	2.5	4.0
2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0	0.0	0.0	0.0
3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0	0.0	0.0	0.0
4	0.0	4.0	1.0	0.0	0.0	3.5	3.5	0.0	3	3.5	1.5	3.5
5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0	0.0	0.0	0.0
6	4.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0	0.0	0.0	0.0
7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0	0.0	0.0	0.0

Table 5

2. Based on movies (3 in the shiny dashboard) input by the user, find the correlation with all other movies.

Movies input by the user ←

	Sudden Death (1995)	Young Guns (1988)	Inside (1996)
Three.Colors..Blue..Trois.couleurs..Bleu...1993.	0.018993450	0.03359187	-0.0027457683
Kalifornia..1993.	0.084977478	0.16070660	0.0733407965
Weekend.at.Bernie.s..1989.	0.039078061	0.20986873	0.0381750100
Better.Off.Dead.....1985.	0.026683198	0.18863536	0.0131101340
Waiting.for.Guffman..1996.	0.018734279	0.07443256	0.0212138898
Event.Horizon..1997.	0.056234553	0.15227612	-0.0026973300
Spawn..1997.	0.073849384	0.15441866	0.0336255190
Weird.Science..1985.	0.028332215	0.25392346	-0.0024027108

Table 6

3. Filter the dataset and remove the row where the movie is one of the input movies.
4. Sum the correlation values and sort the final column descending order.

	title	sum
1	Three Colors: Blue (Trois couleurs: Bleu) (1993)	0.8893434
2	Kalifornia (1993)	0.8893434
3	Weekend at Bernie's (1989)	0.7941189
4	Better Off Dead... (1985)	0.7139037
5	Waiting for Guffman (1996)	0.7106683
6	Event Horizon (1997)	0.6478365
7	Spawn (1997)	0.6371048

Table 7

5. Recommend the top 10 movies

User-based Collaborative Filtering

`User-based collaborative filtering [7][8] is a memory-based collaborative filtering algorithm that looks at the entire dataset to find similar entities or users and recommends articles or movies liked by similar users.

Algorithm

1. Transpose the IBCF document-term matrix created in item-based collaborative filtering to obtain movies as rows and users in columns
2. Based on input *UserID*, find correlations with all other users
3. Map users with the correlation values

	userId	cor_values
1	1	0.1091536142
2	2	-0.0017446418
3	3	-0.0014708181
4	4	0.2528965582
5	5	0.0059723657

Table 8

4. Filter the table for the users with the top 3 correlation values. Fetch the movies the similar users have watched by adding a column "UserId" in the IBCF matrix and joining (inner join) it to the top 3 correlations table. Transpose it to get users as columns and movies as rows

Input: User 5

	5	1946	5550	5573
Major League II (1994)	0.0	0.0	0.0	0.0
Old Boy (2003)	0.0	0.0	4.5	0.0
Corporation, The (2003)	0.0	0.0	0.0	0.0
Casshern (2004)	0.0	0.0	0.0	0.0
Appleseed (Appurushîdo) (2004)	0.0	0.0	0.0	0.0
Constantine (2005)	0.0	0.0	0.0	0.0
Night Watch (Nochnoy dozor) (2004)	0.0	0.0	0.0	0.0
Kung Fu Hustle (Gong fu) (2004)	0.0	0.0	0.0	0.0
League of Ordinary Gentlemen, A (2004)	0.0	0.0	0.0	0.0
Sin City (2005)	4.5	5.0	5.0	4.5

Table 9

5. Add movies as a column, and filter for movies that the user in consideration has not watched but at least one of the similar users have watched. This is done by filtering for rows where the rating of the user is 0 and at least one of the others is non-zero
6. Thereafter, ratings for movies are added across the similar users and top 10 movies based on their cumulative sum are recommended to the user in consideration (user 5 in this case). The below movies will be recommended to the user.

Movies not seen by User 5

title	5	1946	5550	5573	movieScore	movieId	genres
Matrix, The (1999)	0	5	5	4	14.0	2571	Action|Sci-Fi|Thriller
Silence of the Lambs, The (1991)	0	4.5	5	4	13.5	593	Crime|Horror|Thriller
Twelve Monkeys (a.k.a. 12 Monkeys) (1995)	0	4.5	4.5	4	13.0	32	Mystery|Sci-Fi|Thriller
Godfather: Part II, The (1974)	0	5	4	3.5	12.5	1221	Crime|Drama
Star Wars: Episode V - The Empire Strikes Back (1980)	0	4.5	5	0	9.5	1196	Action|Adventure|Sci-Fi
Lucky Number Slevin (2006)	0	5	4.5	0	9.5	44665	Crime|Drama|Mystery
Star Wars: Episode IV - A New Hope (1977)	0	4.5	4.5	0	9.0	260	Action|Adventure|Sci-Fi
Lord of the Rings: The Fellowship of the Ring, The (2001)	0	0	5	4	9.0	4993	Adventure|Fantasy
Saving Private Ryan (1998)	0	4.5	0	4	8.5	2028	Action|Drama|War
Black Hawk Down (2001)	0	0	4.5	4	8.5	5010	Action|Drama|War

Table 10

Results

Shiny Dashboard

Both the above approaches (item and user-based collaborative filtering) were combined to an interactive shiny dashboard [9] to recommend movies to users based on their viewership history and impromptu mood (based on genre and movie(s) selection.

Below are screenshots from the dashboard,

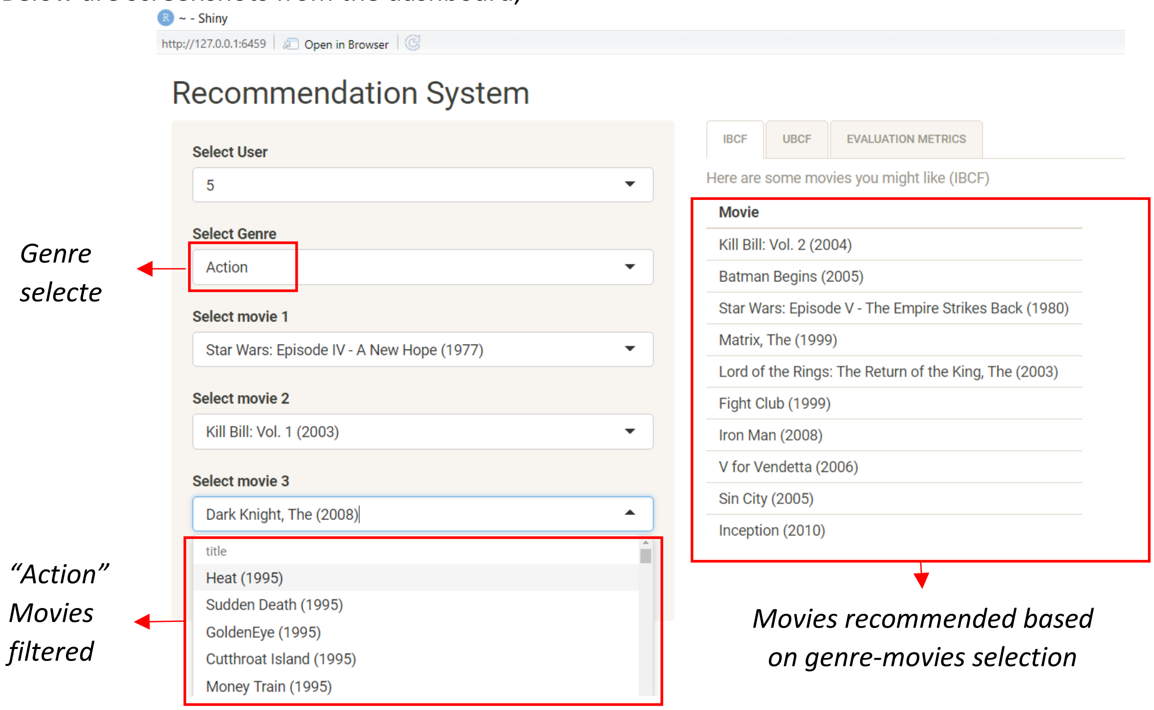

Fig 5

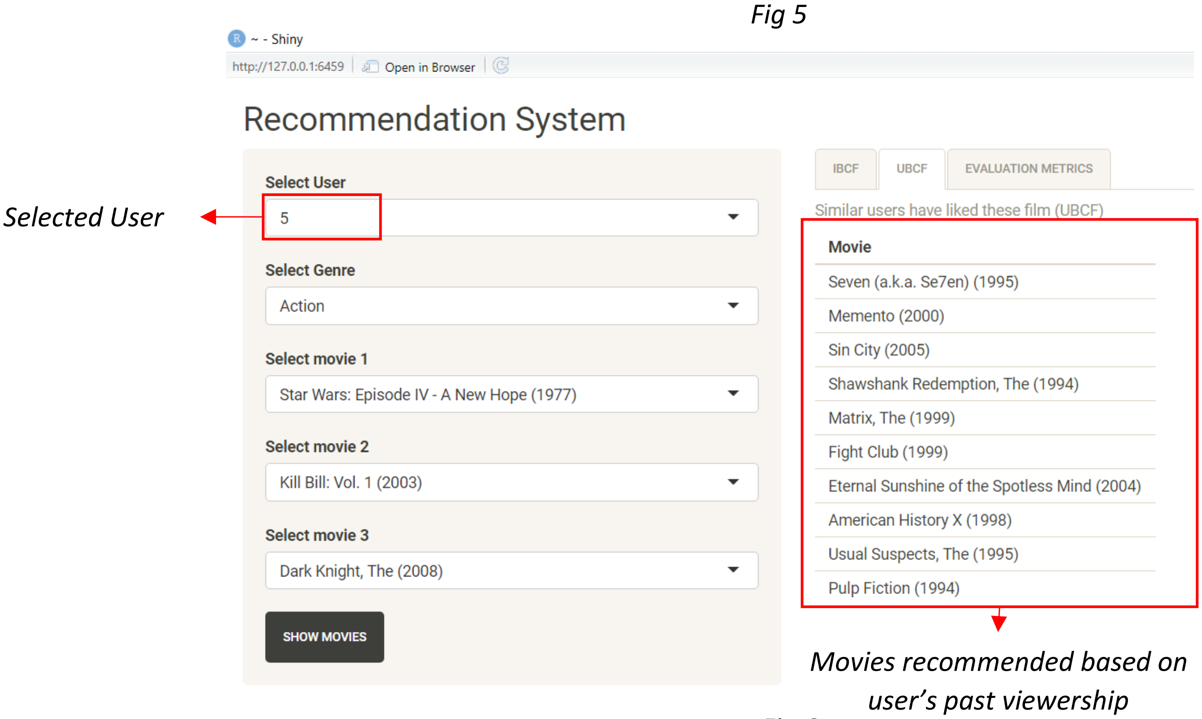

Fig 6

To compare the results, below are the movies that were rated well by the user,

title	genres
Sex, Lies, and Videotape (1989)	Drama
Seven (a.k.a. Se7en) (1995)	Mystery|Thriller
Usual Suspects, The (1995)	Crime|Mystery|Thriller
Léon: The Professional (a.k.a. The Professional) (Léon) (1994)	Action|Crime|Drama|Thriller
Pulp Fiction (1994)	Comedy|Crime|Drama|Thriller
Shawshank Redemption, The (1994)	Crime|Drama
Schindler's List (1993)	Drama|War
Trainspotting (1996)	Comedy|Crime|Drama
Godfather, The (1972)	Crime|Drama
Full Metal Jacket (1987)	Drama|War

Table 11

The movies the user rated well indicate that he/she likes action/thriller/war/drama. The recommended movies in *fig 6* are also highly rated films of similar genres.

Evaluation

By using user-based collaborative filtering, the ratings of the top 10 similar users are obtained. The weighted product of ratings *(correlation with user) * rating* predicts what the user will rate a movie.

The model was evaluated using several metrics. A subset of data (10k users) was taken as test data. It was assumed that ratings were not known, and then ratings were predicted. A threshold of 3.5 was used to classify rating as good or bad. The table below shows evaluation metrics for a sample user (userId -26) in our test dataset.

IBCF | UBCF | EVALUATION METRICS

Show 10 entries Search:

	title	userActualRating	predictedRating	How did the user rate the movie?	Prediction: Will the user like the movie?	Error
1	Seven (a.k.a. Se7en) (1995)	4	4.83333333333333	Good	Good	0
2	Usual Suspects, The (1995)	5	4.5	Good	Good	0
3	Pulp Fiction (1994)	5	4.5	Good	Good	0
4	Shawshank Redemption, The (1994)	5	4.66666666666667	Good	Good	0
5	Schindler's List (1993)	4.5	4.5	Good	Good	0
6	Trainspotting (1996)	5	4	Good	Good	0
7	Godfather, The (1972)	4.5	4.5	Good	Good	0
8	Full Metal Jacket (1987)	5	4.5	Good	Good	0
9	L.A. Confidential (1997)	4.5	4.75	Good	Good	0
10	Big Lebowski, The (1998)	5	4.5	Good	Good	0

Previous 1 2 3 4 5 Next

Fig 7

Confusion Matrix:

	Actual Values	
Predicted Values	Bad (0)	Good (1)
Bad (0)	1	6
Good (1)	6	61

Accuracy: 0.8378
Sensitivity: 0.9104
Specificity: 0.1429

Similar calculations were done for 10k test users, and **accuracy of 79.23%** was obtained on entire test data.

Discussion

We can understand from the results, how user and item-based collaborative filtering can be used to generate insightful recommendations for users.

Future scope

- Scrape summaries of every movie, match keywords and bigrams to obtain more accurate similarity scores
- Create a hybrid recommender system based on content-based filtering and collaborative filtering
- User ensemble models along with neural networks to predict movies better
- Give more weight to the user's recent watching history compared to past history as current watching patterns are more indicative of preferences

References

1. Data set: https://grouplens.org/datasets/movielens/
2. https://cran.r-project.org/web/packages/splitstackshape/index.html
3. http://cs229.stanford.edu/proj2018/report/128.pdf
4. https://pdfs.semanticscholar.org/767e/ed55d61e3aba4e1d0e175d61f65ec0dd6c08.pdf
5. http://www.cs.carleton.edu/cs_comps/0607/recommend/recommender/itembased.html
6. http://files.grouplens.org/papers/www10_sarwar.pdf
7. https://www.seas.upenn.edu/~cse400/CSE400_2006_2007/ChenJatia/Writeup.pdf
8. https://www.researchgate.net/publication/316107913_User-item_based_Collaborative_Filtering_for_Improved_Recommendation
9. https://shiny.rstudio.com/

Appendix

Preliminary EDA

Preliminary visualizations are below,

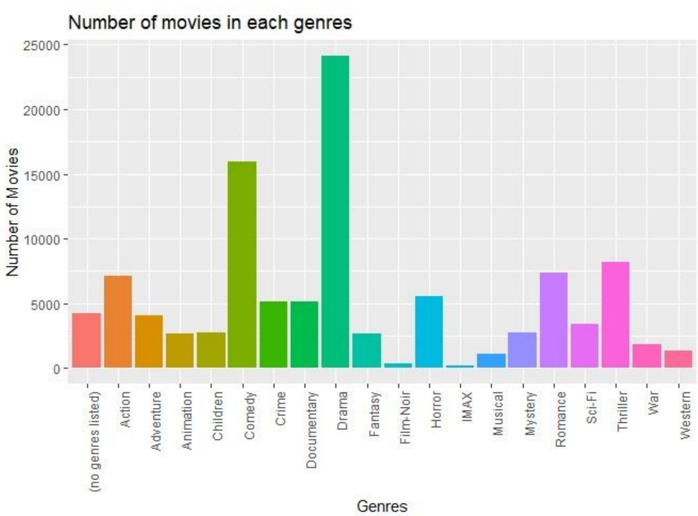

Fig 8

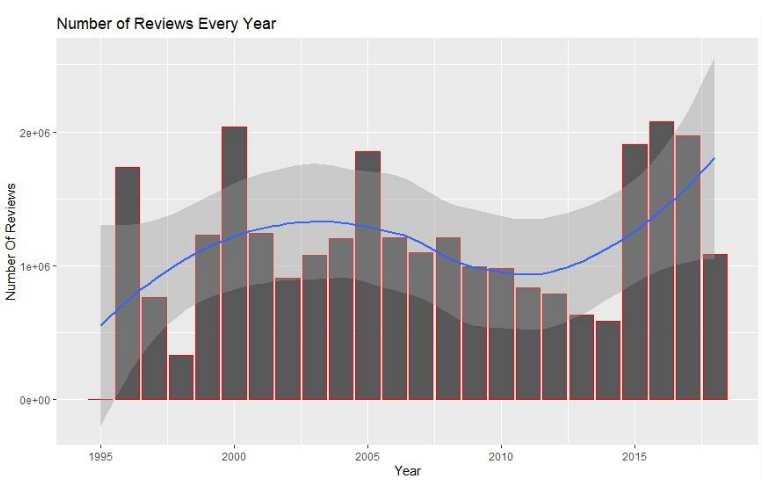

Fig 9

The above graph shows the number of movies in each genre and the number of reviews every year. As observed, Drama has the highest number of movies, while genres like Film-Noir and Imax have the lowest. Also, we have no-genres listed in the charts, thus the data requires a bit of pre-processing.

From the second graph, we can infer that at the turn of the century the number of reviews jumped high, however, after that, the count of reviews plummeted. Thus, we can say that movies released/reviewed during this time (2001-2014) have a smaller number of samples.

Two graphs below depict the average rating by genre and count of rating for each genre. The first graph shows that for Film-Noir, the rating is the highest, hence, it can be easily interpreted that films that fall under this genre are significantly better, however, the second graph shows that the number of reviews for this genre, Film-Noir, is way less than the number of reviews for some other genre.

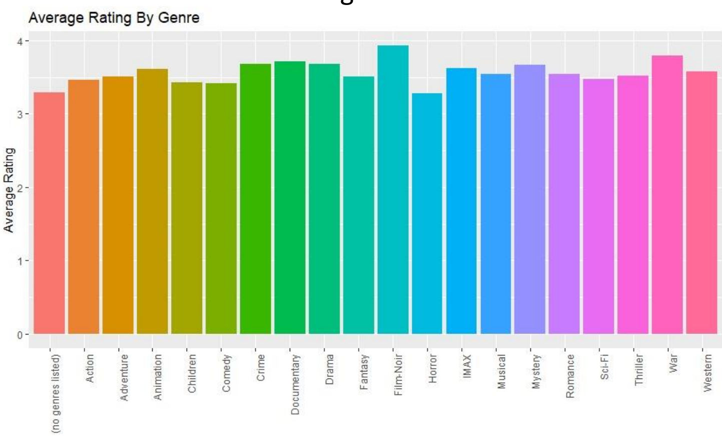

Fig 10

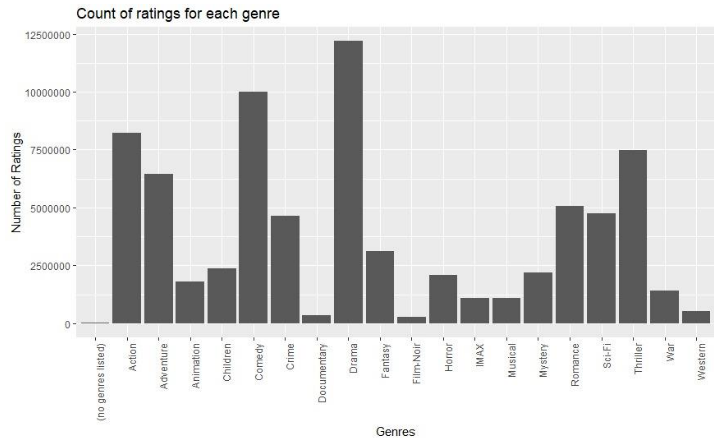

Fig 11

Collaborative Filtering Algorithms

Memory-based Collaborative Filtering Algorithm

Memory-based algorithms utilize the entire user-item database to generate a prediction. These systems employ statistical techniques to find a set of users, known as neighbors, that have a history of agreeing with the target user (i.e., they either rate different items similarly or they tend to buy a similar set of items). Once a neighborhood of users is formed, these systems use different algorithms to combine the preferences of neighbors to produce a prediction or top-N recommendation for the active user. The techniques, also known as nearest-neighbor or user-based collaborative filtering are more popular and widely used in practice. This method is quite stable as compared to User-based collaborative filtering because the average item has a lot more ratings than the average user. Unlike the user-based collaborative filtering algorithm, the item-based approach investigates the set of items the target user has rated and computes how similar they are to the other items *i* and then select k most similar items $\{i_1, i_2, ..., i_k\}$. At the same time, their corresponding similarities are also computed. Once the most similar items are found, the prediction is then computed by taking a weighted average of ratings on these similar items.

Model-based Collaborative Filtering Algorithm

Model-based collaborative filtering algorithms provide item recommendations by first developing a model of user ratings. Algorithms in this category take a probabilistic approach and envision the collaborative filtering process as computing the expected value of a user prediction, given his/her ratings on other items. The model-based algorithm tries to compress a huge database into a model and perform a recommendation task by applying the reference mechanism into this model. Model-based collaborative filtering can respond to user's requests instantly.

Types of similarity scores

Cosine based similarity:

In this case, two items are thought of as two vectors in the m dimensional user-space. The similarity between them is measured by computing the cosine of the angle between these two vectors. The similarity between items 'i' and 'j' (where 'i' and 'j' are vectors with pairwise complete observations) shown in the above figure is given by,

$$sim(i,j) = cos(i,j) = \frac{i.j}{\|i\|\|j\|}$$

Correlation-based similarity:

In this case, the similarity between two items 'i' and 'j' is measured by computing the Pearson-r correlation. The similarity between items 'i' and 'j' (where 'i' and 'j' are vectors with pairwise complete observations) shown in the figure above is given by,

$$corr(i,j) = \rho_{i,j} = \frac{cov(i,j)}{\sigma_i \sigma_j}$$

Where:

- cov = covariance
- σ_i, σ_j are standard deviations of 'i' and 'j' respectively.

Above formula can also be written as,

$$corr(i,j) = \frac{\sum_{u \in U}(R_{u,i} - \bar{R}_i)(R_{u,j} - \bar{R}_j)}{\sqrt{\sum_{u \in U}(R_{u,i} - \bar{R}_i)^2}\sqrt{\sum_{u \in U}(R_{u,j} - \bar{R}_j)^2}}$$

Here $R_{u,i}$ denotes the rating of user u on item i, $\bar{R}_i$ is the average rating of the i-th item.

- After finding the similarities of our input movie 'i' with all other movies, we sort the resulting values in descending order.
- Based on this, the top 10 movies are recommended.

Pros and Cons:

- Item-based collaborative filtering is faster compared to the user-based approach
- Easy to implement and maintain
- It works with little user feedback (New users will have no to little information about them to be compared with other users and hence item-based collaborative filtering is preferred)
- Predictions are less accurate compared to memory-based algorithms