Improving Movie Recommendation System by grouping users

Rohit Sharma / Avineet Kumar Singh

College of Engineering and Computing

University of South Carolina

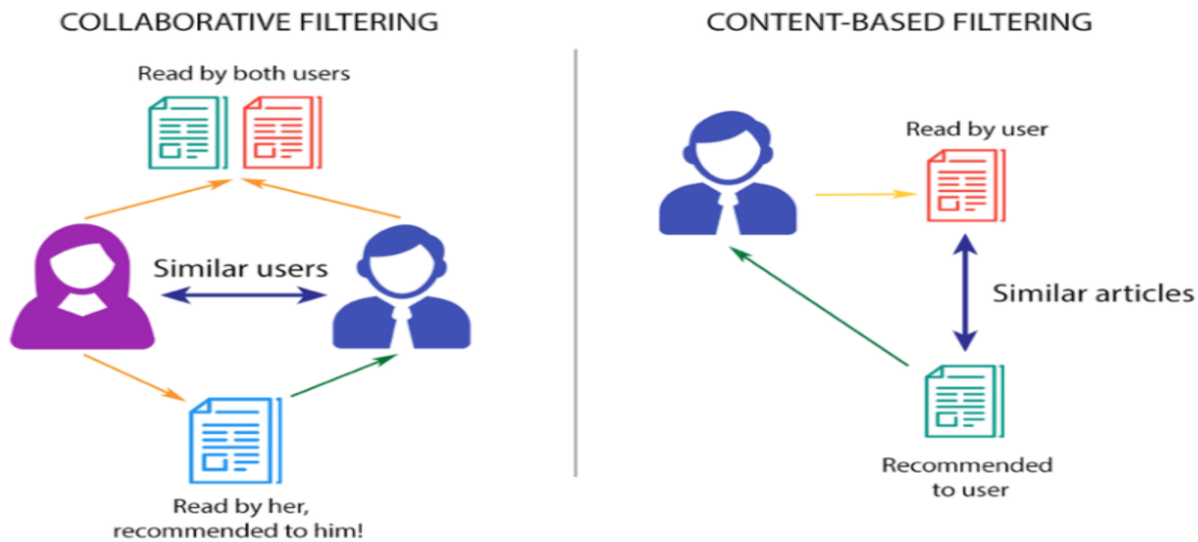
Email: ROHITS@email.sc.edu / AS89@email.sc.edu

BACKGROUND

In today's world there is lot of options for movies to watch for. A movie recommendation application provides choices of movies of specific genre (information content) as per the demand of the user. For any input, it computes a similarity score based on genre of the movie with the input movie. However, to recommend a desired movie to the user, recommendation has to deal simultaneously with user's clearly defined preferences and user's past behaviors.

A plethora of algorithms and approaches has been recommended by various researchers to create personalized recommendations. The core of recommendation systems is the recommendation algorithm. In literature, most of the recommendation algorithms are based on collaborative filtering and content-based filtering approaches. However, to overcome the shortcoming of both approaches, most modern recommender systems utilize hybrid approach. Let's understand briefly how these algorithms work.

Content-based filtering: This type of recommendation is the most popular approach among the traditional approaches utilized for movie recommendation. The principle of this approach is to recommend an object that has similarities to some other object, the user preferred in the past. Though every content-based recommendation approach is based on simple concept of analyzing movie descriptions to identify movies that are of particular interest to the user, still they are not sensitive to the changes of user interest.



Collaborative filtering method: The collaborative filtering bases its predictions and recommendations on ratings or opinions of other users in collaboration with users' preferences and their historical information. The fundamental assumption of this approach is that the preferences of other users can be selected and assembled in such a way to give a reasonable prediction of the active user's preference. CF algorithms are primarily based on Euclidean embedding and Matrix Factorization models based on Principal Component Analysis (PCA), Singular Value Decomposition (SVD) and Probabilistic Matrix Factorization (PMF).

Hybrid model: Many researchers define the Hybrid Recommender Systems as a technology that applies two or more Recommender System techniques as described before. Usually, Collaborative filtering technique along with another techniques which has a better performance than traditional one-based techniques.

Most recommender systems now use a hybrid approach, combining collaborative filtering, content-based filtering, and other approaches . There is no reason why several different techniques of the same type could not be hybridized. Hybrid approaches can be implemented in several ways: by making content-based and collaborative-based predictions separately and then combining them; by adding content-based capabilities to a collaborative-based approach (and vice versa); or by unifying the approaches into one model. Several studies that empirically compare the performance of the hybrid with the pure collaborative and content-based methods and demonstrated that the hybrid methods can provide more accurate recommendations than pure approaches.

Netflix is a good example of the use of hybrid recommender systems. The website makes recommendations by comparing the watching and searching habits of similar users (i.e., collaborative filtering) as well as by offering movies that share characteristics with films that a user has rated highly (content-based filtering).

For grouping users, we have used below methods-

Cosine similarity is one of the most widely used and powerful similarity measures in Data Science. It is used in multiple applications such as finding similar documents in NLP, information retrieval, finding similar sequence to a DNA in bioinformatics, detecting plagiarism and may more. It takes the angle between two non-zero vectors and calculates the cosine of that angle, and this value is known as the similarity between the two vectors. This similarity score ranges from 0 to 1, with 0 being the lowest (the least similar) and 1 being the highest (the most similar).

Cosine similarity is calculated as follows,

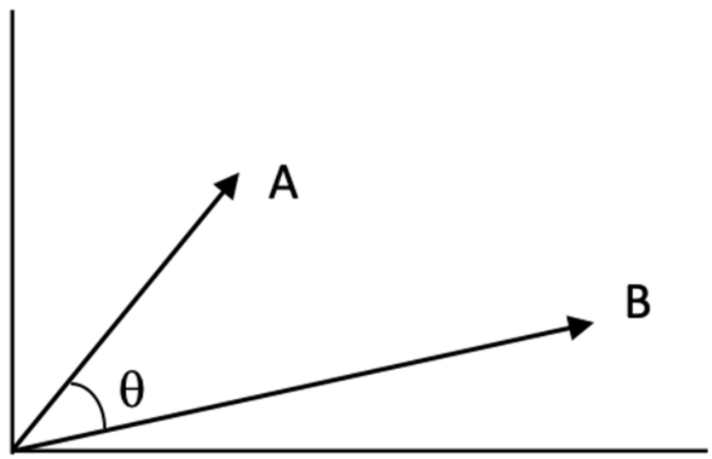


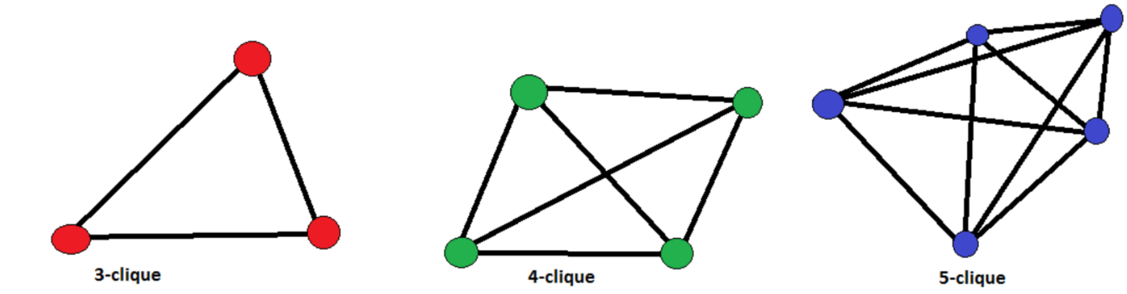
Fig: Angle between two 2-D vectors A and B

$$\cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|} = \frac{\sum\limits_{i=1}^{n} A_i B_i}{\sqrt{\sum\limits_{i=1}^{n} A_i^2} \sqrt{\sum\limits_{i=1}^{n} B_i^2}}$$

Formula: calculation of cosine of the angle between A and B.

Pearson Similarity: Similarity is the Pearson coefficient between the two vectors. For the purpose of diversity, we have also used Pearson Similarity in this implementation.

Undirected connected graph (network analysis): An undirected graph is graph, i.e., a set of objects (called vertices or nodes) that are connected together, where all the edges are bidirectional.
An undirected graph is sometimes called an undirected network. In contrast, a graph where the edges point in a direction is called a directed graph.



PROBLEM

The Collaborative Filtering Recommender is entirely based on the past behavior and not on the context. More specifically, it is based on the similarity in preferences, tastes and choices of two users. It analyses how similar the tastes of one user is to another and makes recommendations on the basis of that. This has created a less accurate recommendation system.

We have tried to target this issue by using Improved Hybrid model. This is done by clustering existing users in the system and then recommending movies. We have also improved that algorithm by using undirected connected graph at clustering stage.

DATASET

In our experiment we used 100k movie lens dataset for recommendations in python platform which was readily available and handled by Group lens organization over several duration of time. This dataset consists of different ratings of 1682 movies given by 943 users. The file ratings.csv in our dataset consist of user's ratings given to different movies in the following format as UserId, MovieId, Ratings, and Timestamp (i.e. the time at which user provide the ratings). The rating scale of the movies range from 1 to 5 while MovieId is between 1 to 1682 Ids.

In our dataset, most of the movies have received less than 50 ratings but each user provides ratings to at least 20 movies.

userId	age	gender	occupation	zip code
1	24	M	technician	85711
2	53	F	other	94043
3	23	M	writer	32067
4	24	M	technician	43537
5	33	F	other	15213
6	42	M	executive	98101

userId	**movieId**	**rating**	**timestamp**
1	31	2.5	1260759144
1	1029	3	1260759179
1	1061	3	1260759182
1	1129	2	1260759185
1	1172	4	1260759205
1	1263	2	1260759151

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

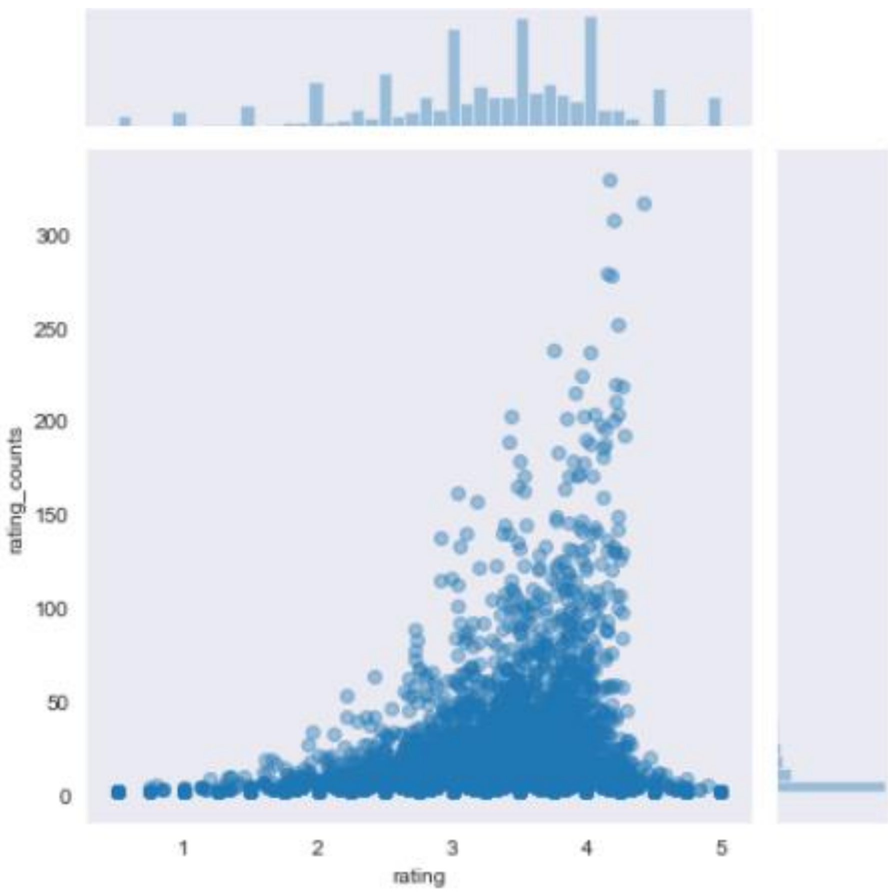


Fig: Average ratings against number of ratings.

APPROACH

In this project we have improved existing hybrid recommendation system by making user clusters based on their personal information and then improving the same algorithm by using connected graph for creating cluster of users. Below are the steps we followed in our two approaches.

Approach 1

1. Clubbed user details like gender, age, and occupation.

2. Used **TF-IDF vectorizer** to create embeddings from clubbed user details.

3. Applied cosine similarity among users.

4. Created clusters for most similar users.

5. Collected movie data with ratings for those group of users.

6. From those movies we recommended most similar movies based on Pearson similarity.

7. Multiplied the similarity score with the ratings and then recommended movies to the user.

8. For prediction we have used userid and movie title to predict movie rating for a new user.

<u>Approach 2 (Improved)</u>

1. Clubbed user details like gender, age, and occupation.

2. Used **TF-IDF vectorizer** to create embeddings from clubbed user details.

3. Applied cosine similarity among users.

4. Created undirected graph for similar users using **k-clique** algorithm.

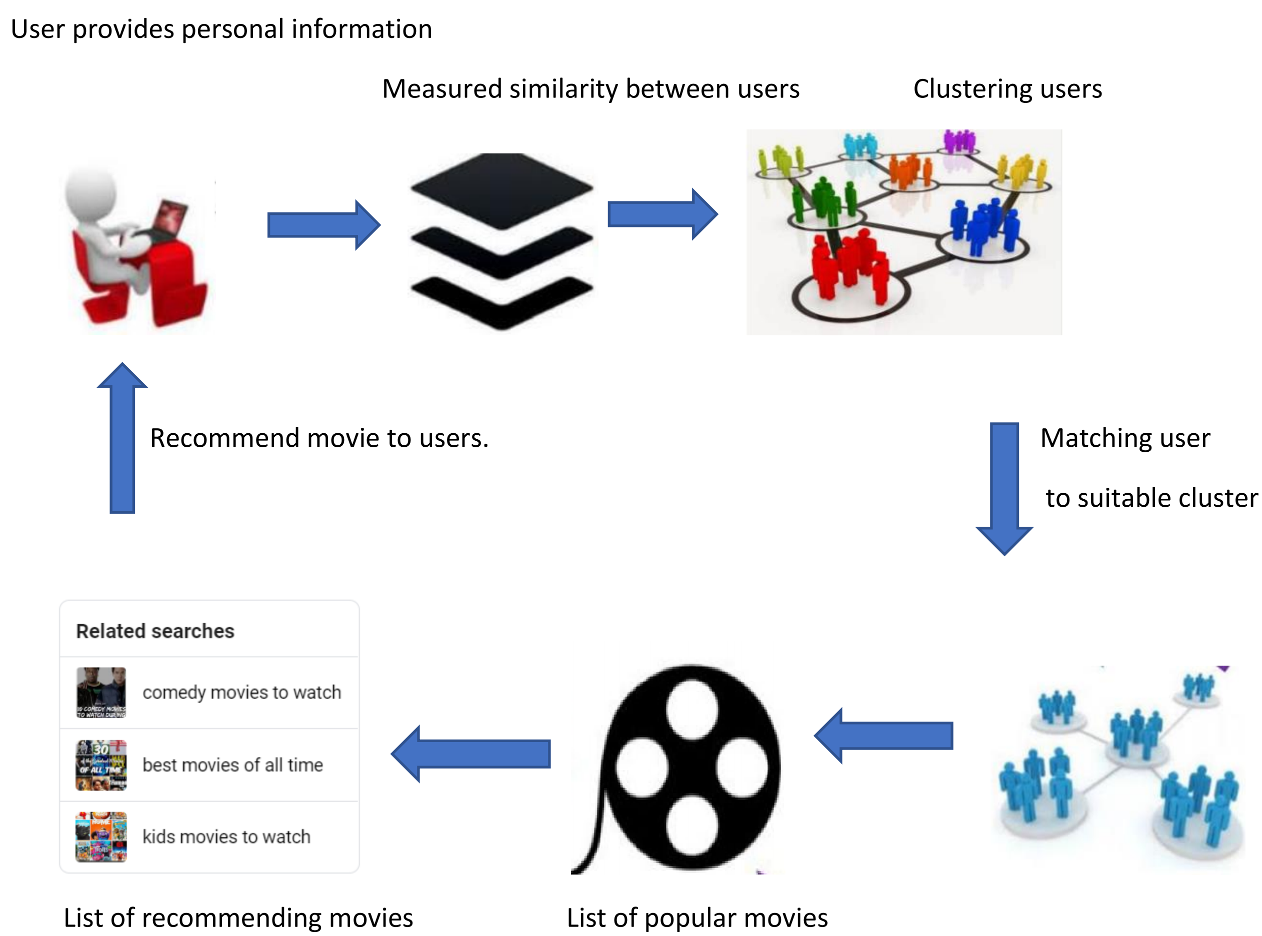
5. Found the most similar cluster from undirected graph for new user using cosine similarity.

6. Collected movie data with ratings for those group of users.

7. From those movies we recommended most similar movies based on Pearson similarity.

8. Multiplied the similarity score with the ratings and then recommended movies to the user.

9. For prediction we have used userid and movie title to predict movie rating for a given user.


User provides personal information
Measured similarity between users
Clustering users
Recommend movie to users.
Matching user
to suitable cluster
Related searches
comedy movies to watch
best movies of all time
kids movies to watch
List of recommending movies
List of popular movies

RESULTS and EVALUATION

Movie recommended by different methods.

Approach 1:

For UserId: 943

Users who rated more than 50 movies.

```
[359, 442, 501, 245, 304, 327, 361, 377, 459]
[442, 501, 245, 304, 327, 361]
```

Recommended Movies	Total Score	Predicted Rating
Reservoir Dogs (1992)	228.609784	4.17558033728839
Goodfellas (1990)	221.945500	4.130840312520293
Jackie Brown (1997)	219.388194	3.7429466560702354
O Brother, Where Art Thou? (2000)	218.369608	3.9676977339628614
Ocean's Eleven (2001)	217.387415	3.951575342357899
Jerry Maguire (1996)	215.236588	3.7827911524297204
Insomnia (2002)	215.113956	3.5159992659602954
Untouchables, The (1987)	214.845558	3.913938619149496
Groundhog Day (1993)	214.387803	3.8546079692362083
Big (1988)	210.148683	3.8999229231800436

Approach 2:

For UserId: 943

- User Cluster using Unconnected Graph with size greater than 5.

user_group_list	combined_feature
943	22 M student
0|3|455|888|716|831	24 M technician
736|16|675|474|794	30 M programmer
32|65|705|36|837|134|390|48|407|476|158	23 M student
81|641|452|587|269|367|527|591|786|51|630|631...	18 F student
624|513|113|159|644|57|863	27 M programmer
56|66|581|645|903|618|396|620|366|340|760|699...	17 M student
434|626|68|104|267	24 M engineer
640|516|72|874|300|368|471|347	24 M student
04|258|323|322|227|197|495|80|275|724|922|541...	21 F student
240|770|659|756|102|938|93	26 F student
848|100|280|617|460	15 F student
912|483|757|103|285|428|653	27 M student
788|923|108|477|511	29 M other
546|677|136|908|589|877|622|157	50 M educator
248|354|583|202|306|371|726|247|152|153|892	25 M student
402|580|166|668|701|718	37 M other
224|787|437|327|890	51 F administrator
594|550|284|621|799	25 M programmer
688|341|918|696|906|733|398	25 M other
416|804|436|904|504	27 F other
913|732|698|459|796|543	44 F other

Users who rated more than 50 movies.

```
[641, 73, 369, 472, 348, 104, 654]
```

Recommended Movies	Total Score	Predicted Rating
Fisher King, The (1991)	755.188866	3.6924078275856727
Unforgiven (1992)	744.529194	3.874908193747084
Fast Times at Ridgemont High (1982)	730.106762	3.6367614026421577
Misery (1990)	720.259366	3.8125068942056446
Three Kings (1999)	718.417139	3.828825197257265
Jaws (1975)	713.815785	3.8666128207107584
Jackie Brown (1997)	711.431124	3.73762570844318
Ghostbusters II (1989)	709.595190	3.008067633269698
Insomnia (2002)	706.948516	3.546193500054592
Untouchables, The (1987)	688.182905	3.8937178631968847

We then calculated Root Mean Square Error (**RMSE)** and Mean Absolute Error (**MAE**) based on the predicted movie ratings for the userId 943.

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{j=1}^{n}(y_j - \hat{y}_j)^2}$$

$$\mathrm{MAE} = \frac{1}{n}\sum_{j=1}^{n}|y_j - y_j|$$

Where yi refers to prediction or yj refers to actual data or vice-versa.

Approach-1		
Recommended Movies	Actual Rating-IMDB	Predicted Ratings
Reservoir Dogs	4.15	4.17558
Goodfellas	4.35	4.13084
Jackie Brown	3.75	3.742947
O Brother, Where Art Thou?	3.85	3.967698
Ocean's Eleven	3.85	3.951575
Jerry Maguire	3.65	3.782791
Insomnia	3.6	3.515999
Untouchables, The	3.15	3.913939
Groundhog Day	4	3.854608
Big	3.65	3.899923

Approach-2		
Recommended Movies	Actual Rating-IMDB	Predicted Ratings
The Fisher King	3.75	3.654977
Unforgiven	4.1	3.874431
Fast Times at Ridgemont High	3.6	3.680693
Misery	3.9	3.832591
Three Kings	3.55	3.816764
Jaws	4	3.832446
Jackie Brown	3.75	3.702675
Ghostbusters II	3.3	3.00048
Insomnia	3.6	3.570056
Untouchables, The	3.15	3.93728

Approach	RMSE	MAE
Approach 1	0.29713	0.093590031
Approach 2	0.2765	0.020239256

CONCLUSION

Movie recommendation system is a complex application of machine learning which has wide scope of improvement. Various movie recommendation approaches have been explored earlier, and from this project it was found that approaches which combine both the content and collaborative approaches based on improved user clustering methods fare with better accuracy. These methods can also be used to overcome some of the common problems in recommender systems such as cold start and the sparsity problem, as well as the knowledge engineering bottleneck in knowledge-based approaches.

FUTURE WORK

1. We could add demographics information of user to cluster users and suggest movies based on the locally watched movies.
2. Including personality data by collecting user information either through social media or asking questions during login could help us understand user more, and recommend movies based on current mood.
3. Using the time/day at which user logs in could also be used to improve this recommender system, as the user preferences could be different during different time of the day or different day of the week.

REFERENCES and LINKS

1. An Exploratory Study of Collaborative Filtering Vs. Content Based Movie Recommendation by Dr. Kumud Kundu, Rahul Pandey , Pankaj Bhatt, Rahul, Riya kakar.
2. Item Based Collaborative Filtering Approach in Movie Recommendation System Using Different Similarity Measures by Jamilu Maaruf Musa and XU Zhihong .
3. **Dataset:** https://grouplens.org/datasets/movielens/latest/
4. **Code(GitLab):** https://gitlab.com/er.sharmarohit22/machine-learning-final-project
5. **Video demo:** https://youtu.be/SDc2vCNqkvk