COLUMBIA UNIVERSITY

MS Machine Learning

Movie Recommendation System

Anunay Sanganal – avs2160

1. Abstract

Recommendation systems are the backbone of streaming services like Netflix, YouTube, Amazon Prime, Hulu, etc. as they need to keep the users constantly engaged so that they keep using their services. The most important way of engagement is by recommending movies or TV shows that the users haven't watched before. There are various types of recommendation systems being used nowadays like Collaborative Filtering Models, Content-Based Recommender Systems, Knowledge-Based Recommender Systems, Hybrid and Ensemble-Based Recommender Systems and many more. These systems utilize various sources of data like user ratings, user viewing history user demographics, item descriptions, etc. to infer user interests to make recommendations.

Having learnt about these methods, I was motivated to build my own recommendation system to recommend movies. I have tried to build a Collaborative and Content based recommendation system using Cosine Similarity and K-Nearest Neighbours respectively using the MovieLens data. MovieLens is a research site run by GroupLens Research at the University of Minnesota that provides comprehensive data on movies for research and educational purposes. With the help of this data I have tried to achieve the following two objectives:

- *Recommend movies for the target user by finding similar users based on ratings they have rated in the past*
- *Recommend similar movies based on the movies you have liked before*

2. Data Collection and Manipulation

The data was downloaded from the MovieLens website [1] which includes data on movies, ratings, movie genome, etc. The data was created by randomly choosing 283228 users between January 09, 1995 and September 26, 2018. The data [2] consists of three datasets:

- Movie dataset containing information on movies
- Ratings datasets containing ratings provided by each user
- Movie Genome dataset

2.1 Movie Dataset

The dataset contains information for 58098 movies from year 1954. Only movies which have at least 1 rating in the Ratings datasets have been included. Each observation in this dataset represents one movie, and has the following information for each movie: movieId, title, genres. Movie ID is a unique identifier for each movie and is consistent across Ratings and Move Genome datasets. Each movie has genre associated with it and it is recorded in the genre variable. A snapshot of the data:

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

2.2 Ratings Dataset

Ratings dataset contains ratings provided by a user for a move. Each observation in this dataset represents one rating of one movie by one user, and has the following format: userId, movieId, rating, timestamp. User ID is a unique identifier for each user and there are 283228 unique users. The observations are ordered first by userId, and then, within each user, by movieId. Ratings variable contains ratings provided by the user for a movie on a 5-star scale, with half star increments (0.5 stars - 5.0 stars). The timestamp variable indicates when the user rated the movie but it's not useful for our analysis. A snapshot of the data:

userId	movieId	rating	timestamp
1	307	3.5	1256677221
1	481	3.5	1256677456
1	1091	1.5	1256677471
1	1257	4.5	1256677460
1	1449	4.5	1256677264

2.3 Genome Dataset

The genome dataset contains tag genome relevance scores for movies based on the Tag Genome paper [3]. The tag genome encodes how strongly a movie exhibits particular property represented by the tags; for example, atmospheric, thought-provoking, realistic, etc. It is similar to the Pandora Genome project in which each song is represented by a vector containing 150-400 genes with each gene corresponding to a characteristic. The structure is a dense matrix; containing 1128 values for each movie. The dataset is split into two files: a file containing the movieId, tagId, relevance and another file having information on each tag. A snapshot of the data:

movieId	tagId	relevance
1	1	0.02900
1	2	0.02375
1	3	0.05425
1	4	0.06875
1	5	0.16000
1	6	0.19525
1	7	0.07600
1	8	0.25200

tagId	tag
40	alaska
41	alcatraz
42	alcoholism
43	alien
44	alien invasion
45	aliens
46	allegory

3. Movie Recommendation using K-Nearest Neighbours

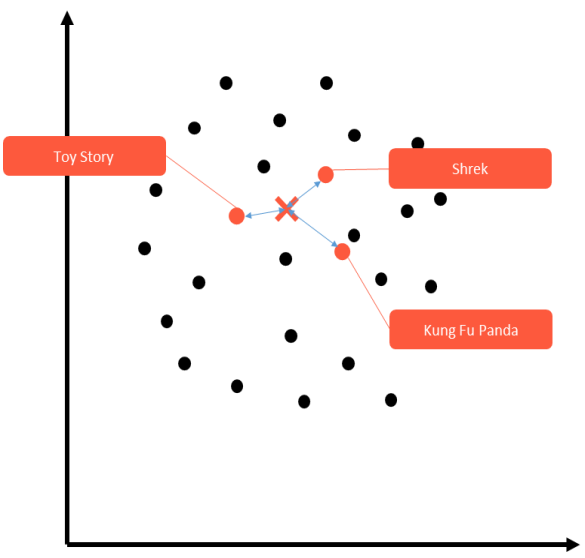
The objective of building this recommendation model was to recommend movies based on the movies a user has liked before. For this analysis, I have used the movie genome dataset. As we are using the attributes of the item, in our case a movie, to make a recommendation, it can be thought of as a Content based collaborative system. The genome was converted into a wide format matrix with each row representing a movie and the column representing its genes (features):

	1	2	3	4	5	6	7	8
1	0.02900	0.02375	0.05425	0.06875	0.16000	0.19525	0.07600	0.25200
2	0.03625	0.03625	0.08275	0.08175	0.10200	0.06900	0.05775	0.10100
3	0.04150	0.04950	0.03000	0.09525	0.04525	0.05925	0.04000	0.14150
4	0.03350	0.03675	0.04275	0.02625	0.05250	0.03025	0.02425	0.07475
5	0.04050	0.05175	0.03600	0.04625	0.05500	0.08000	0.02150	0.07375

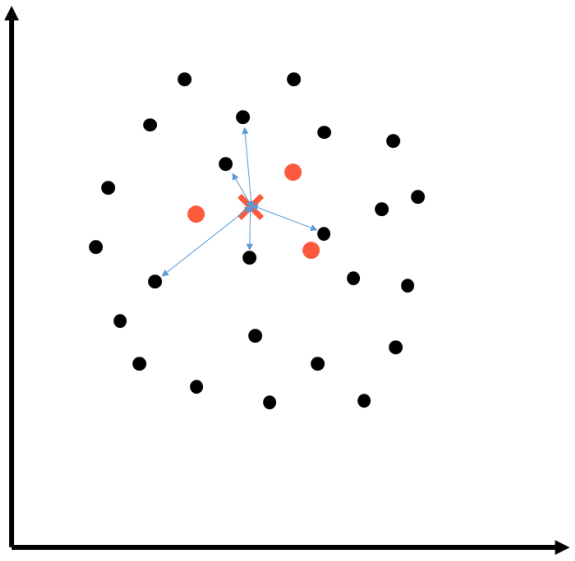
Each movie row in the matrix can be thought of as a point on a multi-dimensional plane with its features as the coordinates. So, given a set of movies the user liked, we can find the centroid of these movies. Then the centroid would indicate the features of an ideal movie the user likes. Hence, by finding the closest points (Movies) to this centroid, we can recommend those movies to the user. The step are as follows to find the top 5 recommendations:

- Get the coordinates (features) of the movie the user liked, for simplicity let's assume 3 movies; "Toy Story", "Kung Fu Panda" and "Shrek"
- Find the centroid of the three points by taking the mean of the features
- Find the closest points to the centroid which can be then recommended to the user

A demonstration of the idea in 2D is illustrated below:



Centroid of the liked Movies



Nearest Points to the Centroid

After taking the centroid and finding the nearest neighbours, the results were the following movies: “Bug's Life”, “Toy Story 2”, “Monsters, Inc.”, “Ice Age” and “How to Train Your Dragon”. As can be seen from the results, the movies are similar to the movies which were liked by the user. The user liked “Toy story”, so “Toy story 2” was recommended by KNN which makes sense. Similarly, other movies are also animated kids’ movies. Hence, though the model is quite simple, it gives surprisingly good results. This model can further be improved by incorporating ratings provided by the user for the movies to be used as weights while calculating the nearest neighbours. Hence, more emphasis can be given to movies which were rated highly among the liked movies.

Movies liked

Recommended Movies

4. Movie Recommendation Using Cosine Similarity

The objective of this recommender system was to recommend movies for the target user by analysing the movies watched by the users similar to them. How are these similar users identified? This is where Cosine similarity comes in. The similarity between the users is found by analysing the ratings they have given for movies they have both watched. The intuition being, the users who like and dislike similar movies have same taste. For example, suppose Bob and Janice have rated similarly in the past for the movies they have both watched, then one can use movies that Bob has liked to recommend to Janice which she hasn’t watched. This is a Collaborative filtering type of recommendation system which is very widely used. For building this model, I have used the ratings dataset which contains the ratings given by each user for movies. The data has been filtered to consider only those users who have rated more than 400 movies. Also, movies that have been rated less than 500 times have been filtered out. The steps followed are as follows:

- Prepare the data for analysis by filtering out the unwanted movies and users
- Create a user-movies ratings matrix with each row representing one user and each column representing one movie. The values are the ratings provided by the users for that movie
- Demean each row by subtracting the average rating of each user form the ratings the user has given for each movie. This helps address the problem when different users may provide ratings on different scale, i.e., some users may rate all movies highly whereas some users may rate all movies negatively. Also, the NA values are replaced by 0.
  A snapshot of the ratings matrix:

	1	2	3	4	5	6	7	8	9	10
4	0.5965278	0.5965278	0.0000000	0	-1.403472	1.0965278	0	0	0.000000	0.5965278
56	-0.2176634	0.7823366	0.0000000	0	0.000000	1.7823366	0	0	0.000000	0.0000000
73	0.3188073	0.0000000	0.0000000	0	0.000000	-0.6811927	0	0	0.000000	0.0000000
79	0.8047619	-0.1952381	0.0000000	0	0.000000	0.0000000	0	0	0.000000	0.0000000
81	0.0000000	-0.0391595	0.4608405	0	0.000000	0.9608405	0	0	0.000000	-0.5391595
100	0.0661323	-0.4338677	0.0000000	0	0.000000	-0.4338677	0	0	0.000000	0.0000000
134	1.1963134	0.1963134	0.0000000	0	0.000000	0.0000000	0	0	0.000000	0.1963134
138	0.0000000	0.0000000	0.0000000	0	0.000000	0.0000000	0	0	0.000000	0.0000000
160	0.4813596	0.0000000	0.0000000	0	0.000000	0.9813596	0	0	0.000000	0.0000000
173	0.0000000	-0.0493238	0.0000000	0	-1.049324	-0.0493238	0	0	-1.049324	-0.0493238

- Then the cosine similarity between each user is computed to generate the final similarity matrix

$$similarity(A,B) = \frac{A \cdot B}{\|A\| \times \|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \times \sqrt{\sum_{i=1}^{n} B_i^2}}$$

- Finally, the similarity matrix can be analysed to find the most similar users to the target user to generate recommendations. A snapshot of the final similarity matrix:

	4	56	73	79	81	100	134	138	160	173
4	1.0000000	0.0838837	0.0566634	0.0689402	0.1045462	0.0735250	0.1330891	0.1170396	0.1051581	0.1630854
56	0.0838837	1.0000000	0.0650693	-0.0152543	0.1122352	0.1017981	-0.0086480	0.1110676	0.0771408	0.1440386
73	0.0566634	0.0650693	1.0000000	0.0467668	0.0885465	0.1204698	0.0344008	0.0584647	0.0598103	0.1435137
79	0.0689402	-0.0152543	0.0467668	1.0000000	0.0332469	0.0765528	0.0702572	0.0393635	0.0233951	0.0370079
81	0.1045462	0.1122352	0.0885465	0.0332469	1.0000000	0.1108104	0.0803149	0.0877542	0.0504458	0.3020629
100	0.0735250	0.1017981	0.1204698	0.0765528	0.1108104	1.0000000	0.0376119	0.0939703	0.1105957	0.1284996
134	0.1330891	-0.0086480	0.0344008	0.0702572	0.0803149	0.0376119	1.0000000	0.0656371	0.0640269	0.1976704
138	0.1170396	0.1110676	0.0584647	0.0393635	0.0877542	0.0939703	0.0656371	1.0000000	0.1154325	0.1140005
160	0.1051581	0.0771408	0.0598103	0.0233951	0.0504458	0.1105957	0.0640269	0.1154325	1.0000000	0.1105788
173	0.1630854	0.1440386	0.1435137	0.0370079	0.3020629	0.1284996	0.1976704	0.1140005	0.1105788	1.0000000

Now how do we use the similarity matrix? Suppose we want to recommend a movie for say user A; then we find the user (say B) who has the highest similarity value with user A and recommend movies which B has rated highly to user A. For example, let us consider two users form our similarity matrix, users with userID 173 and 11560. They have cosine similarity score of 0.4258355. Hence, these two users can be considered as like minded with similar tastes. When we analyse the movies, they have both watched and rated, we find that they have both rated them similarly. Hence, in the following scenario, user 11560 hasn't watched "Schindler's List" which user 173 has rated highly. Hence this movie can be recommended to him.

UserID	Shawshank Redemption, The (1994)	Forrest Gump (1994)	Jurassic Park (1993)	Schindler's List (1993)	Blade Runner (1982)	Godfather, The (1972)	Reservoir Dogs (1992)	Platoon (1986)	Monty Python and the Holy Grail (1975)
173	4.5	5	5	5	5	5	4.5	5	5
11560	4.5	4	4	NA	4	4.5	4	4	4.5

Snapshot of the movies rated highly by both users

The similarity matrix can be used many other applications, for example, it can be used to predict the unobserved ratings for users by analysing the ratings of users similar to them.

5. Conclusion and Future Scope

As seen, by utilising simple models like KNN and Cosine similarity, we were able to give very good recommendations. But we have only scratched the surface of numerous different methods the streaming giants utilize to generate recommendations. Even for the above two methods that I have explored, there are many ways to further improve them. Future scope to explore:

- We can include user ratings as weights in the KNN model to give more emphasis on movies which have been rated highly.
- We can use the similarity matrix to make predictions of unobserved ratings for a user

6. References

1. MovieLens website - (http://movielens.org) (http://grouplens.org/datasets/)
2. F. Maxwell Harper and Joseph A. Konstan. 2015. The MovieLens Datasets: History and Context. ACM Transactions on Interactive Intelligent Systems (TiiS) 5, 4: 19:1–19:19. https://doi.org/10.1145/2827872
3. Jesse Vig, Shilad Sen, and John Riedl. 2012. The Tag Genome: Encoding Community Knowledge to Support Novel Interaction. ACM Trans. Interact. Intell. Syst. 2, 3: 13:1–13:44. https://doi.org/10.1145/2362394.2362395
4. Charu C. Aggarwal - Recommender Systems
5. StackOverflow – For various coding doubts
6. Collaborative Filtering – Artificial Intelligence - All in One (https://www.youtube.com/watch?v=h9gpufJFF-0)