CS6220 Final Report: Movie Recommendation System

Yash Kothari, Sanyam Harne, Ananth Narasimhan, Tarun Vishal, Ankur Yadav

I. Abstract

Several movie recommendations are available, each of which takes into account aspects of the film such as genre, actors, and so on. Although this is a great way to propose movies, we frequently find that some choices are available that we don't care for. Users may choose happy ending romantic movies to sad ending romantic movies, for example. Because both have the same genre, the customer may get a sad ending romantic movie if he likes the joyful ending romantic comedy. To address this, we've chosen to embark on a project that recommends movies based on movies watched by people with similar profiles and the ratings they gave those films.To do this we have decided to use Collaborative Filtering to find similar profiles as that of the user and recommend movies. In order to filter and rank the movies we have used Pearson Correlation.

II. Introduction

- Statement of the problem you are trying to solve

  In this project, we are trying to build a recommendation system that is more personalized and considers more fields specific to the user rather than just recommending the most popular movie present on the platform or in a similar genre.

- Why is it important to solve this problem?

  Today pop culture is an important social interaction and with the ease of making movies, several movies and videos are being produced. Also due to OTT platforms, the production of movies has taken a huge boost. The new era of movies does not involve only mainstream ideas but also targeted movies for a section of the audience. For streaming movie services like Netflix, recommendation systems are essential for helping users find new movies to enjoy. A lot of movie recommendations take into account only genres and not subtypes of the genre which may lead for a bad experience for the user. Due to these aforementioned factors, we have decided to work on this project.

- The reason we use data science:

  Data science is an excellent instrument for accomplishing this. To begin, we can use numerous programs, such as Pandas and Matlab, to undertake significant data analysis and visualization. Instead of displaying raw data, this visualization can show a clear understanding of what the data means by using graphs or plots to provide visual context. Users can better comprehend what current movie genre they want to see and more easily identify trends and patterns. To add on, by training the dataset, we can get a model to predict

possible movies and genres given a specific instance. For example, Jack would like to know which possible movies he can see after he has finished watching Avengers. Based on the previous movies he watched and liked, he will be able to see similar movies he would like to surely like based on his taste in previous movies.

As a result, I believe data science can assist us in mining and analyzing data, as well as predicting specific outcomes. It can also assist us in visualizing large volumes of data in order to gain insights into the dataset, which may include more information than raw data.

- Background information and a bit of literature survey to present what's already known about this problem.

  We have seen several models which recommend movies based on movies' recommended popularity, genre of the movie and the actors in the movie.

  Collaborative filtering has been used for similar fields like grocery shopping, online retail and we are planning to implement a movie recommendation system based on collaborative filtering.

III. Methodology - What's your solution, how do you propose to solve the identified problem?

We are planning to clean our dataset followed by getting the Pearson Correlation values for each of the movies using the rating and movies watched by the user. Using these Pearson correlation values we are using user-based collaborative filtering to recommend similar movies to the user.

- Pearson Correlation:

  Pearson correlation remains unchanged when all items are multiplied by a nonzero constant or when any constant is added to all elements. pearson(X, Y) == pearson(X, 2 * Y + 3), for example, if you have two vectors X and Y. This is a crucial trait in recommendation systems because, for example, two users may rate two series of objects in absolute terms that are completely different, yet they are comparable users (i.e., with similar thoughts) with similar rates on multiple scales.

  The formula returns values ranging from r = -1 to r = 1, with 1 indicating a direct correlation between the two entities (perfect positive correlation) and -1 indicating a perfect negative correlation.

  In our case, a 1 means that the two users have similar tastes while a -1 means the opposite.

- Collaborative Filtering:

  In this project we are using a technique called collaborative filtering (CF). Collaborative filtering has two meanings: a specific one and a broader one.

In the narrow sense Collaborative filtering is a method of creating automatic predictions (filtering) about a user's interests by collecting preferences or taste information from numerous users in a newer, narrower sense (collaborating). The collaborative filtering strategy is based on the notion that if two people have the same opinion on an issue, A is more likely to share B's opinion on a different problem than a randomly chosen person. A collaborative filtering recommendation system for television preferences, for example, may offer predictions about which television show a user would enjoy based on a partial list of that user's preferences (likes or dislikes). It's worth noting that these forecasts are unique to the user, although they're based on data from a large number of people. This is in contrast to the more straightforward strategy of assigning an average (non-specific) score to each item of interest, such as based on the number of votes.

In a broader sense Collaborative filtering is the process of searching for information or patterns using strategies that entail collaboration among various agents, viewpoints, data sources, and other factors. Collaborative filtering is most commonly used to filter very large data sets. Sensing and monitoring data, such as in mineral exploration, environmental sensing over large areas or multiple sensors; financial data, such as financial service institutions that integrate many financial sources; or electronic commerce and web applications where the focus is on user data, etc. Collaborative filtering methods have been applied to many different types of data.

Most recommendation systems employ broader collaborative filtering to detect comparable patterns or information from users. This technique can filter out goods that users enjoy based on ratings or reactions from similar users.

Collaborative filtering can be used to anticipate a user's rating based on user ratings for other movies and other users' ratings for all movies. This notion is used to propose movies, news, apps, and a variety of other goods.

Let's look at an example to better grasp what Collaborative Filtering is.

Assume I have a user named U1 who enjoys the movies m1,m2, and m4. User U2, who enjoys the films m1, m3, and m4, and user U3, who enjoys the film m1.So our job is to recommend which are the new movies to watch for the user U3 next.

- User-Based Filtering:

In user-based collaborative filtering we look at the past history of the user depending on the movies he has watched and suggest similar movies to him. We can use user based collaborative filtering to suggest movies to similar users. Let's pretend we wish to propose a film to our pal Stanley. We can expect that people with comparable tastes will have similar tastes. Assume that Stanley and I both saw the same movies and gave them nearly identical ratings. Stanley, on the other hand, hasn't seen 'The Godfather: Part II,' whereas I have. It seems reasonable to assume that if I enjoy the film, he will as well. As a result, we've developed a fictitious ranking based on our resemblance.

UB-CF, on the other hand, follows this reasoning and recommends goods by locating users who are similar to the active user (to whom we are trying to recommend a movie). The user-based Nearest Neighbor algorithm is one example of this. Two jobs are required for this algorithm:

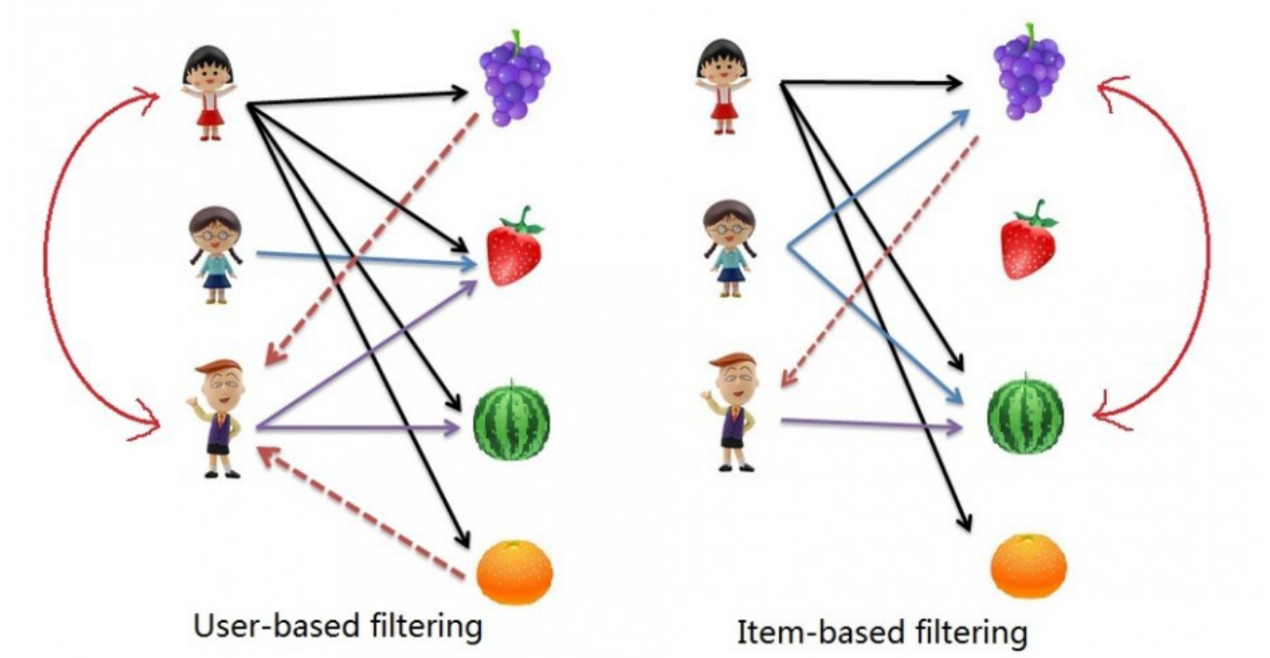
1. Find the K-nearest neighbors (KNN) to the user a, using a similarity function to identify similar users

2. In other words, we're building a User-Item Matrix that predicts ratings on products that the active user hasn't seen based on the ratings of other users who are similar. This is a memory-based method.

- Item-Based Filtering:

  We can design our system to recommend based on the likes of the movies as well. Like if some like two movies a lot and if we identify a movie similar to that movie we can recommend that movie to them as well.

  Returning to Stanley. Rather than focusing on his pals, we may consider which goods from the several alternatives are the most similar to what we know he appreciates. Item-Based Collaborative Filtering is the name of this new focus (IB-CF).



IV. Understanding the data

In this project we are referring to data from this dataset: https://grouplens.org/datasets/movielens/latest/.

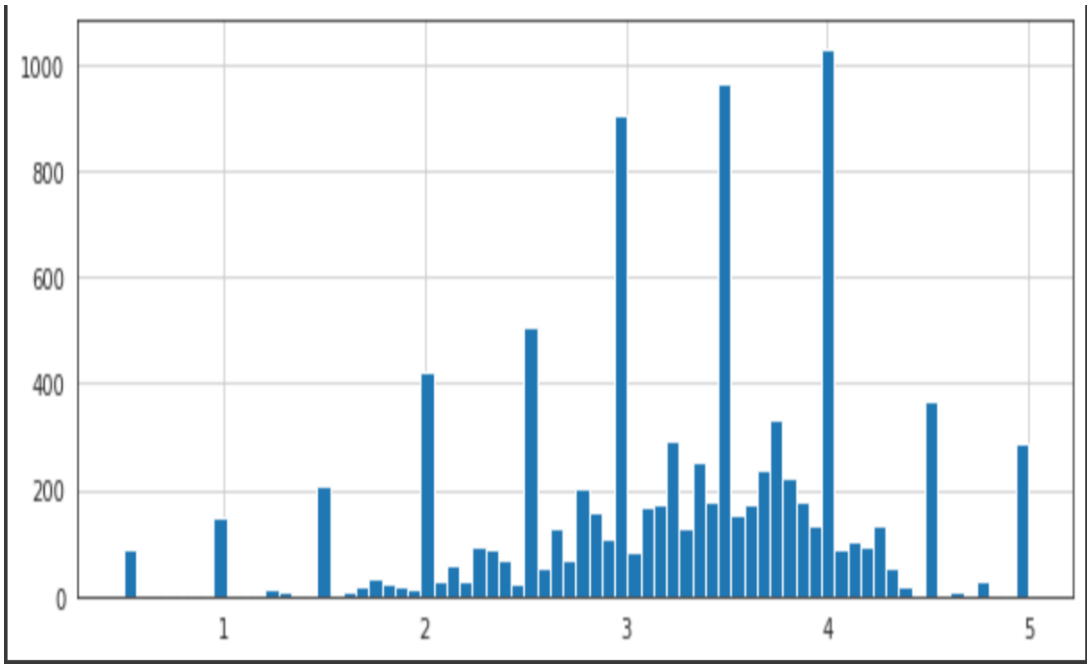
The data for movies:

	movieId	title	genres
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
1	2	Jumanji (1995)	Adventure|Children|Fantasy
2	3	Grumpier Old Men (1995)	Comedy|Romance
3	4	Waiting to Exhale (1995)	Comedy|Drama|Romance
4	5	Father of the Bride Part II (1995)	Comedy

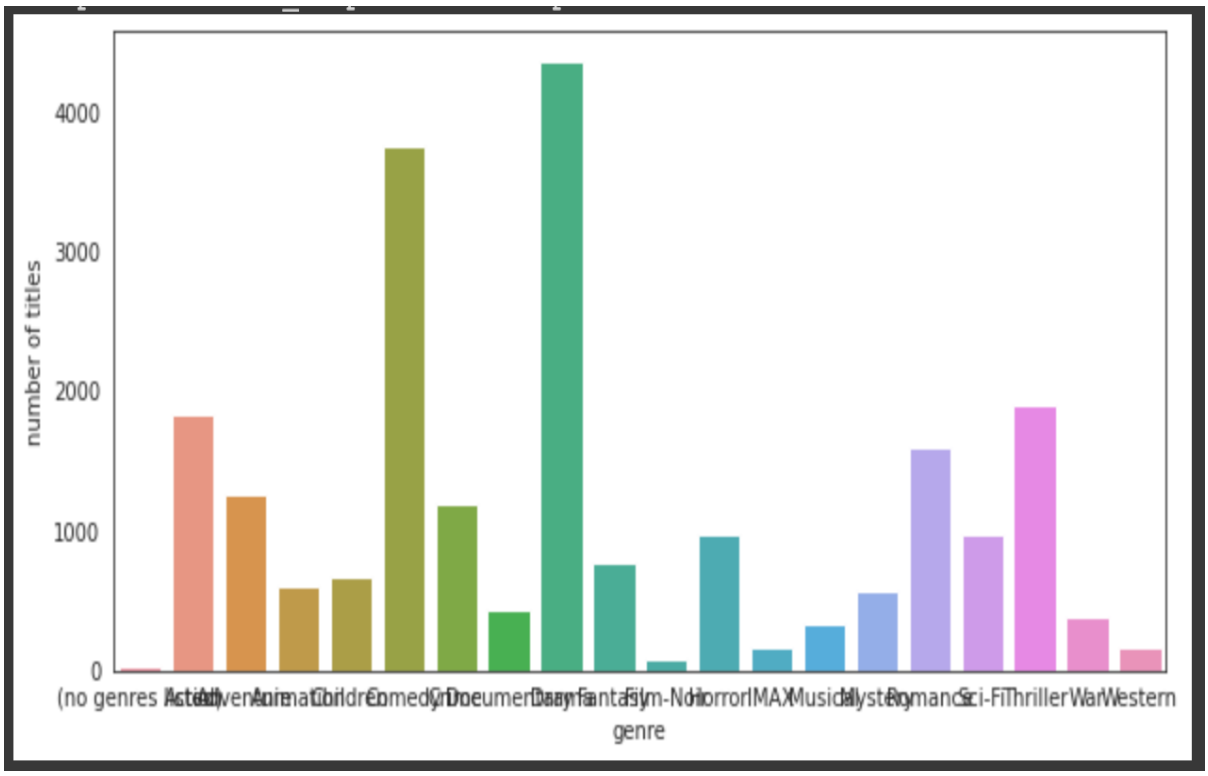
The data for ratings:

	userId	movieId	rating	timestamp
count	100836.000000	100836.000000	100836.000000	1.008360e+05
mean	326.127564	19435.295718	3.501557	1.205946e+09
std	182.618491	35530.987199	1.042529	2.162610e+08
min	1.000000	1.000000	0.500000	8.281246e+08
25%	177.000000	1199.000000	3.000000	1.019124e+09
50%	325.000000	2991.000000	3.500000	1.186087e+09
75%	477.000000	8122.000000	4.000000	1.435994e+09
max	610.000000	193609.000000	5.000000	1.537799e+09

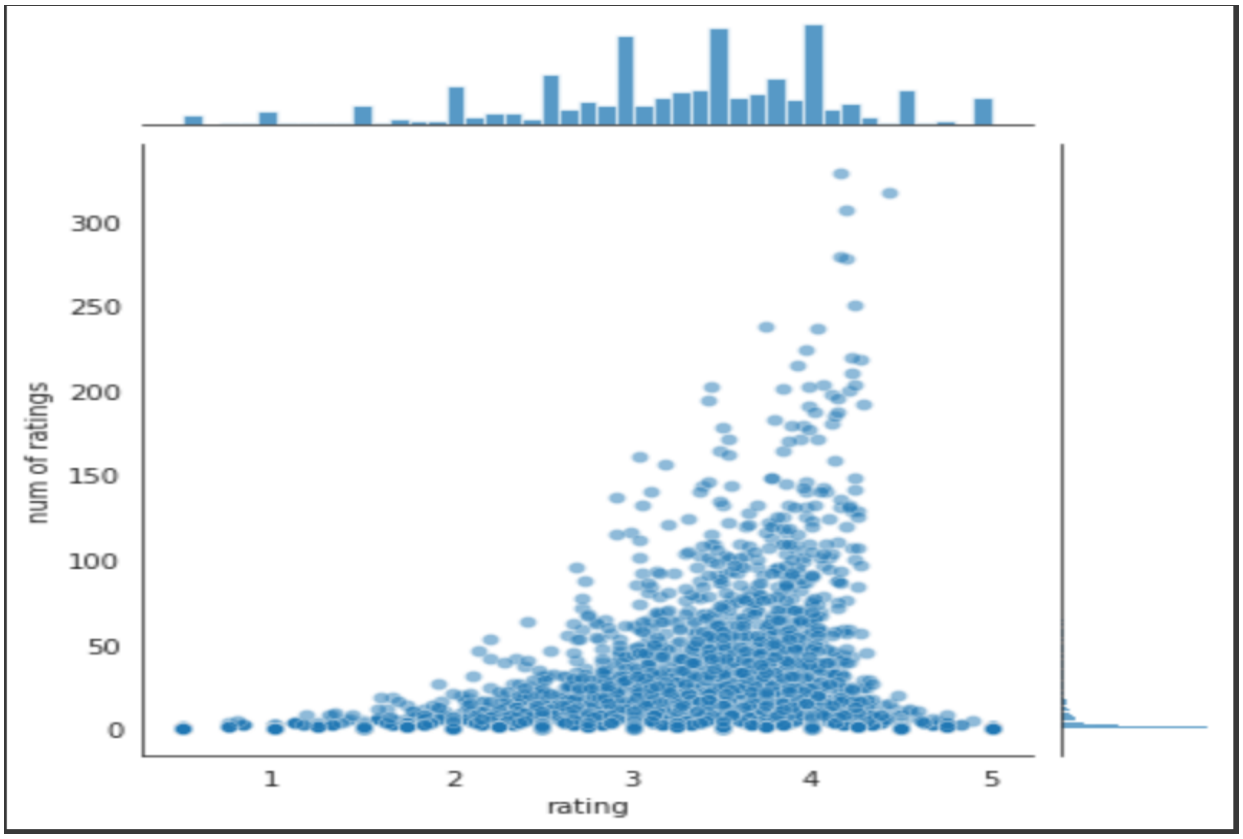
Movie rating:



Genre of movie vs Number of movies of each title



Rating vs number of votes for that rating



IV. Code

Preprocessing the data

- We remove the year from the title in the movie dataframe and add a separate year column using pandas' replace function. Regular expressions are used to do this. Between the parenthesis, look for a year. We use parentheses to avoid clashing with films that include years in their names.
- Collaborative filtering does not make recommendations based on the movie's features. The recommendation is based on the likes and dislikes of the neighbors or other users, as well as their ratings. As a result, the genre column will be removed because it is no longer useful.
- When it comes to the ratings dataframe, the movieId field is the same as it is in the movie dataframe. Each user has assigned several ratings to various films. The recommendation algorithm does not require the column Timestamp. As a result, we can let it go.

Processing the data

- Choose a user based on the movies they've seen. - Find the top X neighbors based on their movie ratings.
- For each neighbor, get the user's viewed movie record.
- Using a formula, calculate a similarity score.
- Recommend the things that received the highest rating.
- Using the rating dataframe to find users who have viewed the same movies We can now extract the subset of people who have watched and reviewed the movies in our input using the movie IDs.
- Groupby creates several sub dataframes where they all have the same value in the column specified as the parameter.
- Showing one such group example by getting all the users of a particular userId.
- Sorting it so users with the movie most in common with the input will have priority.
- Finding the Pearson coefficient to get the similarity values.

Deriving the results

- Storing the Pearson Correlation in a dictionary, where the key is the user Id and the value is the coefficient.
- Since all movies are rated by a group of users We'll do this by taking the weighted average of the movie ratings, with the Pearson Correlation serving as the weight. However, in order to do so, we must first obtain the movies seen by users in our pearsonDF from the ratings dataframe, and then store their correlation.
- Multiply the movie's rating by its weight (the similarity index), then add the new ratings together and divide by the sum of the weights.
- Multiplies the similarity by the user's ratings.
- Applies a sum to the topUsers after grouping it up by userId.
- Order the values to get a list of movies according to their rank based on the order of weighted average recommendation score.

V.Results

- We were able to identify the Pearson Correlation values for all of the movies and user information.
- Given a user profile we were able to recommend movies to users.
- User Input :

	title	rating
0	Breakfast Club, The	4.0
1	Toy Story	2.5
2	Jumanji	3.0
3	Pulp Fiction	4.5
4	Akira	5.0

- Similar movies based on the user:

	movieId	title	year
1261	1289	Koyaanisqatsi (a.k.a. Koyaanisqatsi: Life Out ...	1983
1324	1354	Breaking the Waves	1996
2725	2810	Perfect Blue	1997
3077	3163	Topsy-Turvy	1999
3943	4036	Shadow of the Vampire	2000
6332	6440	Barton Fink	1991
7592	7976	Ken Park	2002
9597	30803	3-Iron (Bin-jip)	2004
18301	90943	Into the Abyss	2011
24907	115210	Fury	2014

VI. Discussion

From the list of movies that we get we can see that the movies are similar to the one suggested in the input. We can also observe that when we see the list of these movies they are really similar to the movie for which we watched and required similar movies. One consideration we had was if a few movies did very well only appear since it could be possible that majority of the people watched that movie and hence the data could be skewed. On further testing and cleaning our dataset slightly more we were able to fix this.

VII. Future work

In future we would like to suggest movies by mapping the plot of the movie. We could analyze the plot of the movie and provide further depth in genre for the type of movie. For example let us consider Romantic movies. By analyzing the plot we can further determine if it is a Romantic movie with a happy ending or a Romantic movie with a sad ending. The main advantage of this would be that we would be able to analyze user preferences even better and provide more detailed recommendations.

Also we were short of data with respect to the ethnicity of the users. We could leverage the users ethnicity and correlate their preferences even more. This could help us show even more tailored movie recommendations to those ethnicities which could make our recommendation even more accurate.

VIII. Conclusion

In conclusion we were able to provide movie recommendations to users based on their preferences and rating given to those movies using the list of movies, rating of movies and user information who rated those movies.

IX.References

- **https://medium.com/@cfpinela/recommender-systems-user-based-and-item-based-collaborative-filtering-5d5f375a127f**
- **https://www.geeksforgeeks.org/user-based-collaborative-filtering/**
- **https://towardsdatascience.com/user-user-collaborative-filtering-for-jokes-recommendation-b6b1e4ec8642**
- **https://en.wikipedia.org/wiki/Collaborative_filtering**
- **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**