Movie Recommender System

Arya Shukla, Ariane Uwase, Kelly Uwase Rubangura
Northeastern University, Boston, MA
DS 5230: Unsupervised Machine Learning
Shukla.ar@northeastern.edu, Uwase.a@northeastern.edu, Uwaserubangura.k@northeastern.edu

Abstract

The film industry is a continuously growing entity in which the success of a movie depends on a variety of factors, including their popularity on different streaming services. However, success of a streaming service also depends on its ability to cater to diverse user preferences, provide extensive content so all users are satisfied with the selection, maintain user engagement, and generate revenue for sustainability. To fulfill all of these requirements, a platform must suggest movies that align with user taste, discover new and relevant content based on historically liked and disliked items, and foster a positive experience. The team merged two datasets to see how a recommender system may come to life. With approximately 27,000 unique movie titles available, movies were suggested depending on two different methods (user-user and item-item) in addition to the discovery of association rules and correlations between movies.

Introduction

In a diverse movie landscape with numerous film choices, a movie recommender system becomes a crucial tool to streamline the viewer's selection process. Similar to the recommendations we get on music platforms, this recommender system utilizes different algorithms such as KNNWithMeans and Singular Value Decomposition (SVD), collaborative filtering (CF) such as user-user and item-item, and different association rules to make predictions and recommendations. The use of a recommender system would allow different streaming platforms to enhance user satisfaction and increase engagement. Streaming platforms offer in-depth insights into user preferences by collecting data viewing history, genre preferences, ratings, watch times and more, enabling the recommender system to provide personalized recommendations. According to research conducted by Rovi, users typically spend an average of 19 minutes looking for something to watch. Implementing a movie recommender system would be beneficial and efficient for the users.

Movielens, the dataset used for various recommender systems projects, originated from the research group, GroupLens, from the University of Minnesota. The two acquired datasets came from Kaggle, in which the team later performed a dataframe merge on the common attribute of movieId. From our exploratory data analysis, we determined that the datasets were very robust. Each user represented in the dataset rated at least 20 movies, and the datasets were populated for about 20 years.

Background

Advances in the development of recommender methods and machine learning tools allow for better representations of large datasets that can suggest top movies more clearly. The team focused on CF, following the basic assumption that if users agree about the quality of some items, then they will also agree about other items.

User-user CF considers user u and finds a set N of other users whose ratings are similar to u's ratings. Item-item CF utilizes similarities between rating patterns of different items; for instance, systems will estimate the rating for item i based on the user's ratings on similar items. In practice, item-item CF often

works better than user-user, as it tends to be more stable compared to the fluctuating tastes of users over time.

Although CF works for any type of item and requires no feature selection, it may suffer from popularity bias in which the system tends to recommend frequently liked items as well as its inability to recommend to users with more unique movie preferences. CF also may struggle with the cold start problem, as there are not enough users or items in the system to find a match. For that reason, the team filtered and focused only on those movies that had 1000 or more ratings.

In addition to CF, the team also chose to include the testing of the Apriori algorithm for association rules. The goal here was to identify any interesting relations between items in the database to see which movies might have been frequently watched together. Doing so would allow designers of a movie recommendation platform to use targeted advertisement for people who view one of the movies to view the other. Streaming services represent this idea in different ways, as on Netflix it may manifest in match percentage whereas Amazon Prime may place these movies together on their carousel of entertainment options.

In the context of recommendation systems, both KNNWithMeans and SVD (Singular Value Decomposition) are algorithms used for CF. KNNWithMeans is an extension that considers the mean ratings of each user. The "Means" in KNNWithMeans refers to the inclusion of the mean rating for each user in the prediction. This helps in normalizing the ratings, as different users might have different rating scales. KNNWithMeans is feasible to understand and implement, but it can come with a large computational cost for massive datasets.

SVD is a matrix factorization technique that decomposes the user-item interaction matrix into three matrices: U (user matrix), Σ (diagonal matrix of singular values), and V (item matrix). This method is used to discover factors that represent underlying characteristics of users and items and uncover hidden patterns. While SVD can handle missing values nicely and provide accurate recommendations, it is like KNNWithMeans in the sense that it is also expensive to run.

Related Work

While our work has probably been replicated in the past by other scientists or streaming service designers, the goal of the team was to understand how larger movie platforms may build their own recommendation systems. For instance, systems created by Netflix, Amazon Prime, and Hulu consider metadata regarding viewing history, director, cast, and crew, and plot to see if a movie would be a match for a user. The datasets to which the team had access neglected to gather information about demographic information, as all users were solely represented by an identification number. Considering data about the other aspects of a user's identity could help to improve the recommendation system even further.

Netflix, for example, publishes different movies depending on which area of the world the user may live. This may be since Bollywood movies may be more popular in India, Anime may be more popular in Japan, and telenovelas may be more popular in Latin American or Spanish-speaking countries. Additionally, these streaming platforms may have restrictions that prevent younger kids from accessing movies designed for crowds over a certain age.

Considering factors such as regional preferences and age, observed in these professional platforms, further demonstrates the potential for refinement in the team's approach. Recognizing the importance of a user's broader identity and incorporating such data could pave the way for more sophisticated and culturally sensitive recommendation systems in the future.

Project Description

The Movielens dataset contains over 20 million movie ratings collected across 138493 users starting 1995 until 2015. The data includes over 27000 movies, with different genres such as comedy, drama, romantic, thriller and more.

Figure 1 demonstrates the percentage distribution of the top movie genres in the dataset. Drama is combined with a few other categories as well, but stand

alone, it is the top genre at 32.5%. Comedy is the second largest genre at 16.5% and documentaries consist of about 14.0%.

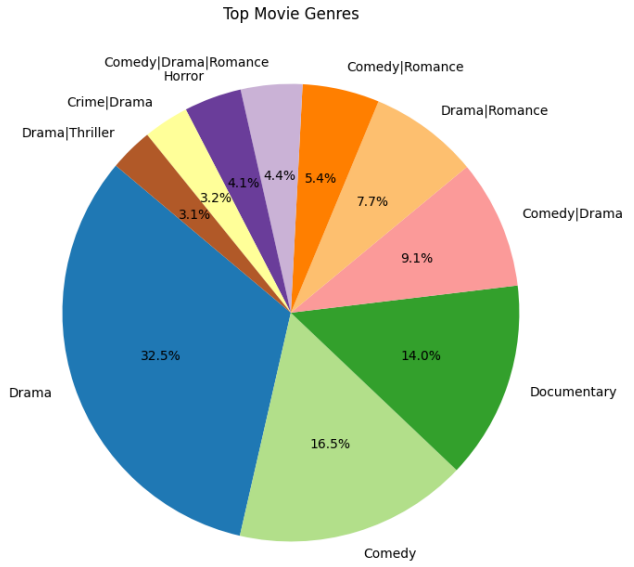

Figure 1: Percentage distribution of the top movie genres in the dataset.

Figure 2 provides a snapshot of the top movies across all genres, this figure may show some popularity bias, as *The Incredibles* show up in four out of the nine represented genres and *Revolutionary Girl Utena* is falls at the top in two out of the nine genres.

	Genre	Top Movie
0	Drama	Revolutionary Girl Utena: Adolescence of Utena (a.k.a. Revolutionary Girl Utena the Movie) (Shoujo kakumei Utena: Adolescence mokushiroku) (1999)
1	Comedy	Incredibles, The (2004)
2	Documentary	'Hellboy': The Seeds of Creation (2004)
3	Comedy|Drama	Incredibles, The (2004)
4	Drama|Romance	Ice Age: Dawn of the Dinosaurs (2009)
5	Comedy|Romance	Incredibles, The (2004)
6	Comedy|Drama|Romance|Horror	Incredibles, The (2004)
7	Crime|Drama	Lupin III: The Castle Of Cagliostro (Rupan sansei: Kariosutoro no shiro) (1979)
8	Drama|Thriller	Revolutionary Girl Utena: Adolescence of Utena (a.k.a. Revolutionary Girl Utena the Movie) (Shoujo kakumei Utena: Adolescence mokushiroku) (1999)

Figure 2: Top Movies for each of the top genres

User-user CF

The user-user collaborative filtering implementation approach began with the selection of a user identification number (ID) at random. The team was able to then find which movies that user watched, allowing them to gather the IDs of those who watched 60% or more of the same movies, as that was an acceptable threshold. A list was then generated of these similar users and ratings to determine the correlations between the representative user and the similar users.

Item-Item CF

As mentioned, in our item-item collaborative filtering approach, we first filtered out the less common movies in order to improve recommendation accuracy. Following that, we generated a user-movie matrix where the columns represent the movies, the rows represent users and their ratings. Using the movie matrix, we first pick a random movie and compute correlation with all the other movies. To calculate correlation between the movies, we used Cosine similarity because of its ease with sparse rating data. Using the similarity scores between the randomly chosen movie and all other movies, we are able to identify the top 10 movies with the highest correlation. This approach assumes that users who have previously enjoyed similar movies are likely to also enjoy the same movies in the future. In summary, item-item collaborative filtering provides a personalized approach to recommendations based on user's preferences and correlations with other movies.

Association rules

This analysis involves algorithms like Apriori and FP-growth tree algorithms, which are advanced techniques employed for transactional data. These algorithms are designed to calculate the degree of association between data values and their respective combinations. For our project, we opted for the Apriori algorithm

Algorithm 5.1 Frequent itemset generation of the *Apriori* algorithm.

1: $k = 1$.
2: $F_k = \{ i \mid i \in I \wedge \sigma(\{i\}) \geq N \times minsup\}$. {Find all frequent 1-itemsets}
3: **repeat**
4: $k = k + 1$.
5: $C_k = \text{candidate-gen}(F_{k-1})$. {Generate candidate itemsets.}
6: $C_k = \text{candidate-prune}(C_k, F_{k-1})$. {Prune candidate itemsets.}
7: **for** each transaction $t \in T$ **do**
8: $C_t = \text{subset}(C_k, t)$. {Identify all candidates that belong to t.}
9: **for** each candidate itemset $c \in C_t$ **do**
10: $\sigma(c) = \sigma(c) + 1$. {Increment support count.}
11: **end for**
12: **end for**
13: $F_k = \{ c \mid c \in C_k \wedge \sigma(c) \geq N \times minsup\}$. {Extract the frequent k-itemsets.
14: **until** $F_k = \emptyset$
15: Result $= \bigcup F_k$.

Figure 3: Apriori Algorithm Pseudocode

Steps to Calculate Association rules:

1. Generate Frequent Itemsets . This is where the Apriori algorithm is used to find patterns in the datasets and create itemsets to be used for further analysis.
   To implement the algorithm, we used "mlxtend.frequent_patterns" to discover the itemsets in the merged list. Using a minimum support threshold of 0.2 and a maximum length of 2

2. Apply Association rules on the itemsets
   Using both lift as a score evaluating method and filtering rules that contain a specific movie in their antecedents and then sorting the results

in descending order. We used confidence as a score evaluating measure as well to show that as the lift value increases, there is a corresponding rise in other measures such as confidence.

Prediction with KNNwithmeans and SVD

Additionally, out of the algorithms used to make predictions for Nan ratings, SVD was identified to have performed better than KNNWithMeans with an RMSE of 0.66 where KNNWithMeans had an RMSE of 0.80. This could be due to the fact that in general SVD handles sparse data well.

Empirical Results

User-User

The user-user CF method exploits similarities between users to make movie recommendations. By comparing the preferences of similar users, the system can suggest movies that the target user might enjoy based on the historical preferences of their peers. Based on the table below, the top 10 similar users and movies are recommended. Since we can see that the user and correlation in the top 10 is the same for every entry but the movieID and ratings are different, the data may be experiencing a low variability in user preferences. Since users have very similar preferences across the board, it could result in high correlations between many pairs of users, leading to similar top matches.

The user-user CF method exploits similarities between users to make movie recommendations. By comparing the preferences of similar users, the system can suggest movies that the target user might enjoy based on the historical preferences of their peers. Based on the table below, the top 10 similar users and movies are recommended. Since we can see that the user and correlation in the top 10 is the same for every entry but the movieID and ratings are different, the data may be experiencing a low variability in user preferences. Since users have very similar preferences across the board, it could result in high correlations between many pairs of users, leading to similar top matches.

The team then calculated a weighted rating by multiplying the correlation between users and the ratings given by those users, assuming that users with higher correlations have more influence on the recommendation. Movies that include a weighted rating above a threshold of 3.5 were then filtered and the top 10 movies were displayed. Adjustment of the threshold can occur depending on the motivations of the designer. The figure demonstrates the top 10 movies recommended after the randomly generated user.

	userId	corr	movieId	rating
0	8963.0	0.866025	1	2.5
1	8963.0	0.866025	4	3.5
2	8963.0	0.866025	6	3.5
3	8963.0	0.866025	7	3.0
4	8963.0	0.866025	9	4.0
5	8963.0	0.866025	10	3.0
6	8963.0	0.866025	11	3.0
7	8963.0	0.866025	14	3.0
8	8963.0	0.866025	16	3.0
9	8963.0	0.866025	17	4.0

Figure 4: User-User: randomly generated user

```
4093                Lilies of the Field (1963)
6474            What's Up, Tiger Lily? (1966)
7701            Day of the Jackal, The (1973)
7847     Christmas Carol, A (Scrooge) (1951)
7854                      Hitcher, The (1986)
10424                      It's a Gift (1934)
12597           Forbidden Kingdom, The (2008)
12604                       Religulous (2008)
12638                            Taken (2008)
14471                  Ninja Assassin (2009)
Name: title, dtype: object
```

Figure 5: User-User: Top 10 Recommended movies

Item-Item

The figure below demonstrates the resulting top 10 movies for the randomly selected movie, *Major*

Payne, using the item-item collaborative filtering approach we implemented.

These recommendations were generated by looking at the cosine similarity between the randomly selected movie and all other movies in the data and outputting the movies with highest correlation to the user.

These are the top 10 movies with high correlation for the movie Major Payne (1995)

title	Cosine similarity
'burbs, The (1989)	0.105932
(500) Days of Summer (2009)	0.036919
*batteries not included (1987)	0.051715
...And Justice for All (1979)	0.030521
10 Things I Hate About You (1999)	0.097017
10,000 BC (2008)	0.041254
101 Dalmatians (1996)	0.097108
101 Dalmatians (One Hundred and One Dalmatians) (1961)	0.080443
102 Dalmatians (2000)	0.051839
12 Angry Men (1957)	0.060142

Figure 6: Item-Item: Top 10 with high correlation with the movie Major Payne

Association Rules

The figures shown below are results for the generated frequent itemsets and the association rules for further analysis into movies that are likely to be viewed together. Using both lift and confidence to showcase movies closely related with "Forrest Gump"

	support	itemsets
0	0.263934	(Ace Ventura: Pet Detective (1994))
1	0.300000	(Aladdin (1992))
2	0.239344	(Alien (1979))
3	0.206557	(Aliens (1986))
4	0.334426	(American Beauty (1999))
...	...	...
305	0.219672	(Star Wars: Episode IV - A New Hope (1977), To...
306	0.265574	(Star Wars: Episode V - The Empire Strikes Bac...
307	0.211475	(Star Wars: Episode V - The Empire Strikes Bac...
308	0.206557	(Terminator 2: Judgment Day (1991), Star Wars:...
309	0.203279	(Terminator 2: Judgment Day (1991), True Lies ...

310 rows × 2 columns

Figure 7 : Generated frequent itemsets(10) with minimum support = 0.2 and maximum length= 2

antecedents	consequents	lift
(Forrest Gump (1994))	(Mrs. Doubtfire (1993))	1.738222
	(Pretty Woman (1990))	1.716762
	(Speed (1994))	1.572193
	(Lion King, The (1994))	1.552273
	(Jurassic Park (1993))	1.542489
	(Apollo 13 (1995))	1.522025
	(Dances with Wolves (1990))	1.514938
	(Men in Black (a.k.a. MIB) (1997))	1.505757
	(Mask, The (1994))	1.499816
	(Back to the Future (1985))	1.474609

Figure 8 : Top 10 movies related to Forrest Gump Using Lift

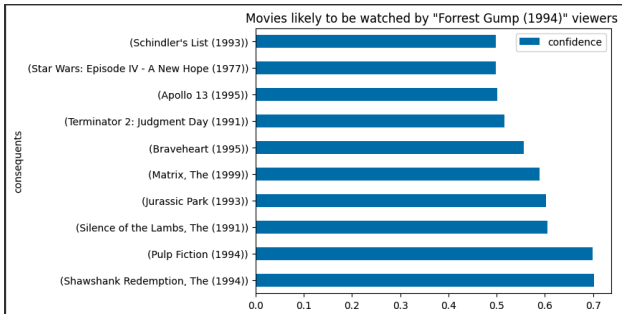

Figure 9 : Horizontal Bar graph for movies likely to be watched by "Forrest Gump" Viewers based on Confidence

Predictions

The table here represents the actual ratings and the predicted ratings by the model. Based on these predictions, it can be concluded that the model is reasonably accurate for all these movies. While some predicted ratings are lower than the actual, that may imply that user preferences have changed over time and that periodic retraining may be a solution here. However, some samples have a lower actual rating than the predicted rating, indicating that the user was moderately satisfied with the movie but perhaps found it slightly less enjoyable than the model predicted

User ID	Movie ID	Actual Rating	Predicted Rating
98604	79	3.128546	3.144258
70204	2	3.211977	3.014810
54732	502	2.284047	3.053858
88181	5	3.064592	3.078389
73042	232	4.035610	3.099101
116859	3	3.151040	3.046264
111237	1	3.921240	2.949241
82842	11	3.667713	3.161590
459	1103	3.812048	3.138959

Figure 10: User-User: Top 10 Recommended movies

Conclusions/ Future Directions

In this project, the team successfully understood and utilized user-user and item-item CF to suggest movies based on their respective methodologies.

Although the systems perform well, there are some directions for development if there were more time. It may be helpful to get an understanding about demographic data, as streaming services having access to age parameters, nationality, ethnicity, and gender of their users may allow for more personalized recommendations. With that in mind, the implementation of content-based recommendations is a much more personal technique for users to get suggestions, so the team would have tried to investigate that further. The addition of a dataset with information about user feedback such as clicks on certain movies and the duration of viewing could have been an asset to the existing data.

For future students taking the class, the team's advice about the project would be to start early and to find an efficient way to share the code you individually complete. The team utilized Google Colaboratory that code could simultaneously be completed, and updates were pushed immediately. GitHub was a useful tool for publishing the final source code.

While the rating.csv was too large to push to the GitHub repository, the team was still able to load the-information into the code with no issues of efficiency or additional errors**.**

Link to Git Repository

Link to Repository: https://github.com/shukla-arya/DS5230

References

F. Maxwell Harper and Joseph A. Konstan. 2015. The MovieLens Datasets: History and Context. ACM Transactions on Interactive Intelligent Systems (TiiS) 5, 4, Article 19 (December 2015), 19 pages. DOI=http://dx.doi.org/10.1145/2827872

Study: Netflix, hulu users spend 19 minutes a day searching for something to watch (2016) *CBS News*. Available at: https://www.cbsnews.com/newyork/news/netflix-streaming-search-app/ (Accessed: 11 December 2023).