Barcelona School of Economics

Data Science Methodology Program

21D009 Networks: Concepts and Algorithms

Movie recommendations using networks

Authors:
Codd, Jonny
Chen, Joshua
Gallegos, Rafael
Pérez, Carlos

Professors:
Milán, Pau
Komander, Björn

December 20^{th}, 2023

Contents

List of Figures

List of Tables

1 Introduction

Popular recommendation techniques in Machine Learning often apply many network and network-adjacent concepts to create personalized recommendations. The matrix of user-to-item scores can be thought of as the adjacency matrix of a bipartite network - with the users representing one set of nodes and the items representing the other. The edges are represented either by binary or scalar relationships between users and items, implying unweighted or weighted graphs respectively.

Our project is two-fold. Using the concepts learned in this course, we will first analyze one of these networks to identify trends and patterns and extract information. We will then study and build recommendation systems and propose/implement new strategies and techniques derived from the first part to improve methodologies.

2 Literature Review

3 Dataset

MovieLens is popular movie recommender system dataset developed by GroupLens, a computer science research lab at the University of Minnesota. The goal of this challenge is to recommend movies to its users based on their movie ratings. Group Lens offers datasets of different sizes and their datasets are widely used in research and teaching contexts.

The selected dataset consists mainly on two files: movies.csv and ratings.csv. Movies dataset has 9,742 unique films and a column indicating the genres of the film. All possible genres are: 'Romance', 'Musical', 'Children', 'Documentary', 'Sci-Fi', 'Film-Noir', '(no genres listed)', 'Crime', 'Mystery', 'Drama', 'Western', 'Fantasy', 'Animation', 'Thriller', 'War', 'Action', 'Adventure', 'IMAX', 'Comedy', 'Horror'. The number of movies per genre is represented in Figure 3.2.

Ratings dataset consists of 100,836 ratings with 610 unique users that rated 9,724 movies. As it can be observed in Figure **??**, the ratings from users are right-skewed, which suggests that users tend to enter their rating on movies that they probably have liked. Ratings from users have been registered from 1996-03-29 until 2018-09-24. The most popular movies among users have been: Shawshank Redemption, The (1994), Godfather, The (1972), Fight Club (1999), Godfather: Part II, The (1974) and Goodfellas (1990).

The median user has rated 70 films, whereas the user with the lowest number of watched films was 20 movies and the user with the highest number of rated films is 2698.

4 Network analysis

With the MovieLens dataset, we created 4 different networks. The first, the user-movie network, is the aforementioned user-item network that will serve as the foundation for both the other networks and the recommendation systems that we build. From this network,

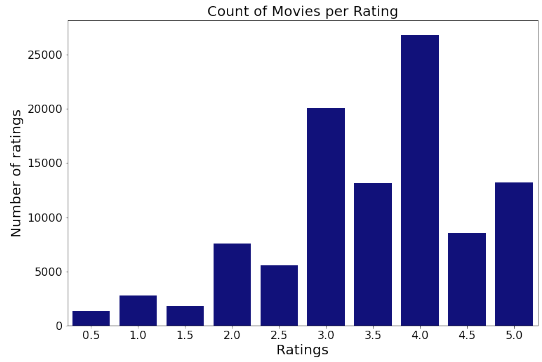

Figure 3.1: Count of Movies per Rating

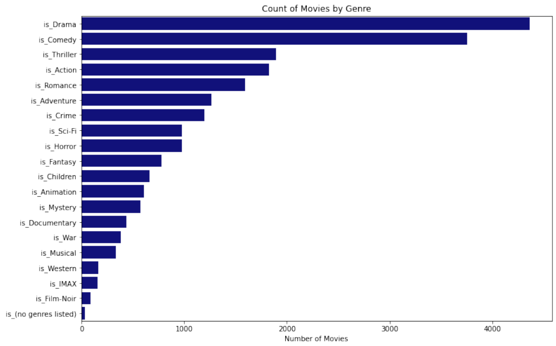

Figure 3.2: Count of Movies per Genre

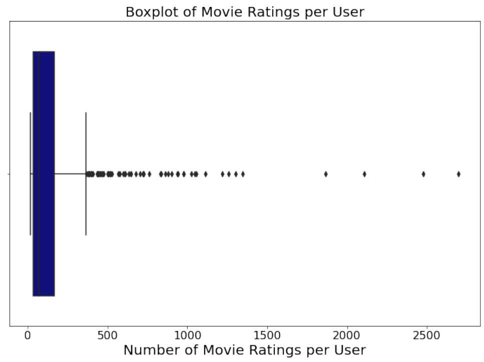

Figure 3.3: Count of Movies per User

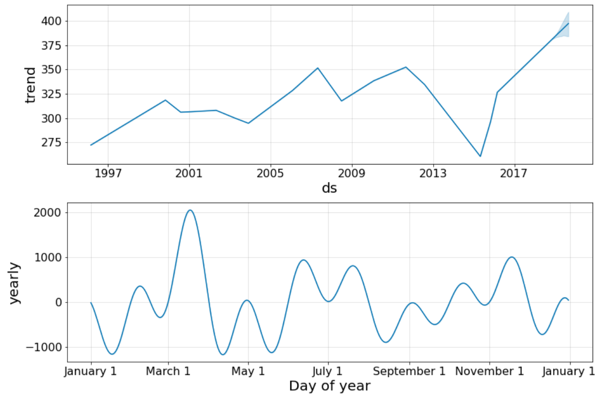

Figure 3.4: Time Series Analysis of Ratings with Prophet

we also constructed a user-to-user network - a symmetric unipartite network capturing the similarity between users. Similarly, we did this for the movies as well. Finally, we created another bipartite network that, instead of capturing relationships between users and specific movies, it describes the "fan score" between a user and a genre of movie.

4.1 User and movie bipartite network

To create the user/movie bipartite network, we used the Networkx package in Python. We selected the unique User Ids as one set of nodes and the movie titles as another. Edges were then added if a user has seen a movie. Note that this is a undirected, unweighted graph. Ratings are not considered here.

Despite an incredibly sparse matrix, understandably with close to 10,000 movies - this results in a connected graph - meaning there is only one component.

To better understand that users and movies, we calculated the centrality for each of them: With this graph - the degree centrality for users and movies are aligned with how many reviews they have. The centrality score for users is much higher than that for movies as

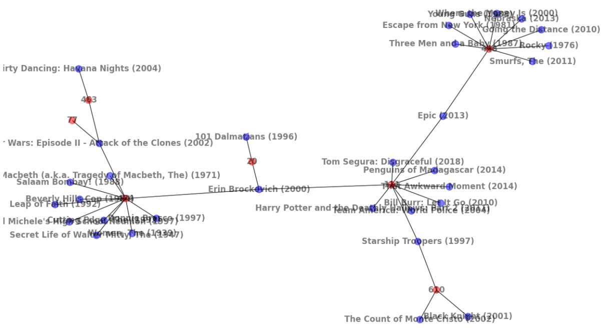

Figure 4.1: Subset of User to Movie Network

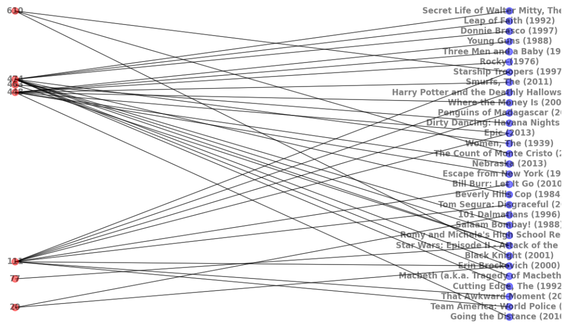

Figure 4.2: BipartiteLayout

there are many more reviews per user than there are per movie.

However, this may not be the best way to actually capture centrality as highly central movies and users might not just be how many direct nodes that they have. Therefore, we also calculated the betweenness and closeness centralities. Take, for example, The Avengers (2012). Regardless of how many reviews it may have, it will probably be an important bridge (betweeness) between the Marvel movies. It is relationships like these that we are hoping to capture. However, there seems to be very little difference in the rankings of these centrality metrics as the top scores keep generally the same order. A couple near the bottom were pushed down but there was no large shuffling.

It is important to note already that the most highly centralized movies are all produced before 2000 (in fact, this extends far beyond just the top 10), despite most of the reviews coming after the 2020s. This is understandable as movies that have been around longer are more likely to have more reviews.

4.2 User-to-user network

To construct the user-to-user network, we calculated the similarity between all users as our edges. We first pivoted the Ratings dataframe on the *UserId*. Our resulting dataframe has the UserId as the index, the Movies as the columns, and any user ratings inside. This is exactly the adjacency matrix for the User to Movie network that we created in the previous section; however, just weighted edges based on ratings.

We used cosine similarity as our node embedding technique to map user vectors into Euclidean space. Cosine similarity is a common space because it ignores magnitude and focuses only on directions in space. For example, A harsh critic might rate an average movie a 2 but a more generous critic might consider average to be 3. This would not affect at all the cosine similarity (if we kept the ratings positive). A similarity measure such as Euclidean distance would fail here because the more movies you add can only possibly add distance. Therefore people are punished for actually having seen more of the same films.

Because this is an incredibly sparse matrix, in order to calculate similarity scores, we only

Figure 4.3: Network Centrality Measures and Comparative Metrics for Movies

User Id	Degree Centrality	Betweeness Centrality	Closeness Centrality
599	4.067	0.141 (1)	0.406 (2)
414	4.429	0.131 (2)	0.413 (1)
474	3.460	0.120 (3)	0.395 (3)
448	3.061	0.110 (4)	0.388 (4)
274	2.210	-	0.373 (5)
610	2.136	0.060 (5)	0.372 (6)
68	2.067	-	0.371 (7)
380	2.000	0.034 (10)	0.370 (8)
606	1.829	0.050 (6)	0.367 (9)
288	1.731	-	0.365 (10)

Movie Title	Degree Centrality	Betweeness Centrality	Closeness Centrality
Forrest Gump (1994)	0.0339	0.0064	0.4824
Pulp Fiction (1994)	0.0316	0.0050	0.4650
Matrix, The (1999)	0.0286	0.0048	0.4694
The Silence of the Lambs (1991)	0.0287	0.0046	0.4624
Shawshank Redemption (1994)	0.0326	0.0042	-
Star Wars: Episode IV (1977)	0.0258	0.0041	0.4613
Jurassic Park (1993)	0.0245	-	-
Braveheart (1995)	0.0244	-	-
Terminator 2 (1991)	0.0231	-	-
Schindler's List (1993)	0.0226	-	-

used the films that people had shared reviews for. However, this could result in the case where two people are assigned similarities despite having very little in common. For example, if two people who have wildly different preferences watched one random movie and both gave it 4 stars, these two people would be assigned a perfect similarity when, clearly, this should not be the case. To mitigate this, we only kept similarity scores for people that have rated 10 or more movies in common.

We actually calculated two similarity scores. Because ratings are all positive, all vectors will only be in the positive space. Therefore, cosine similarities are limited to only positive values as any two vectors cannot exceed more than a right angle. We constructed a similarity matrix using this framework which will henceforth in the paper be referred to as simply the User Network.

However, we also constructed another network where we subtracted 2.5 from all ratings. This allows for negative cosine similarities but has two important implications. First, in the example above regarding the harsh vs the generous critic, this is no longer true as now, their vectors are directly opposite (-0.5 vs 0.5). This model will be less able to adjust for user biases within their personal ratings. However, a large benefit of this approach is that it also captures *dissimilarity* as well. For example, if two users have seen the same movie but have rated it 0.5 and 5.0, while these ratings are clearly diametrically opposed, this still add similarity towards their score. Now, when we subtract 2.5, essentially set 2.5 as the "neutral" threshold. Anything below is considered negative and anything above is considered positive. Users who share negative or positive scores will be rewarded with similarity and users who have disagreements will negative similarity.

4.3 Movie-to-Movie network

4.4 User-to-Genre network

5 Collaborative Filtering

6 Graphical Neural Networks

7 Conclusions