Movie Recommender System

CS 487 – Final Project - Report

Devon Miller

Kitt Phi

Problem

Most modern streaming services, such as Netflix and Hulu, use machine learning algorithms to attempt to find similarities between movies based on a wide assortment of criteria. These criteria include, but are not limited to, genre, word frequency in titles, and release date. Additionally, most services track statistics such as rating and number of views. Combining these criteria allows services to quantitatively evaluate the best recommendations for users based on their previous watch history, thus improving usability and user experience.

Our goal is to create a simple movie recommender system using data from the MovieLens dataset, one of the largest and most widely used datasets designed for this purpose. Our initial prototype will be based on item-based similarity, using both ratings and movie tags as a metric to evaluate the correlation between movies.

A user-based similarity approach is possible, but requires the logging of user-based information, and is therefore less in line with our scope. Designing the initial prototype with a smaller scope in mind allows us to be more flexible and augment the system later without worrying about leaving the system incomplete and non-functional.

If our work timeline allows, we plan on potentially incorporating more metrics into our recommender system, in addition to rating and tags. The main challenge with this project is that many implementations of recommender systems already exist, so our goal is to try and incorporate as many different item-based metrics as we can in order increase the robustness of our system. Finding metrics which correlate with each other and provide useful information for predicting similarities between movies will be one of the hardest aspects of this project.

Motivation

Much of the success of modern streaming services comes from the implementation of recommendation algorithms. The movie industry is highly saturated, and therefore it can be very difficult for less popular movies to be discovered by audiences, even when viewers might have a strong liking for the movie's content. If streaming services can effectively recommend fresh and relevant content to users, then users are more likely to stay on the service and watch the content being provided to them, thereby increasing the revenue of subscription-based services (or other services which make profit solely off of advertising).

Given that many variables are in play when evaluating a viewer's content preferences, the mission to create a "perfect" recommendation algorithm is continuously ongoing. Any research and analysis in this area is useful, which is why we are hoping to create our own implementation to gain a better understanding of the problem, and gain more experience for future work in the field.

Furthermore, a basic backend algorithm capable of determining correlation between movies can be easily applied to front-end applications designed to recommend movies to users. Such is the case for a web development project required for another course, so implementing this algorithm allows us to complete multiple tasks at once.

Solution

Our solution is to build a basic item-item collaborative filtering system, capable of determining top recommendations for specific films based on two important criteria. These criteria are the ratings of each movie, and the tags associated with each film. These correspond to the ratings.csv and tags.csv files, respectively.

We aren't using any specific user information in this application, so recommendations are based purely on calculated similarities to the chosen source film, and not any specific user.

The primary dataset we are using can be found at: https://grouplens.org/datasets/movielens/.

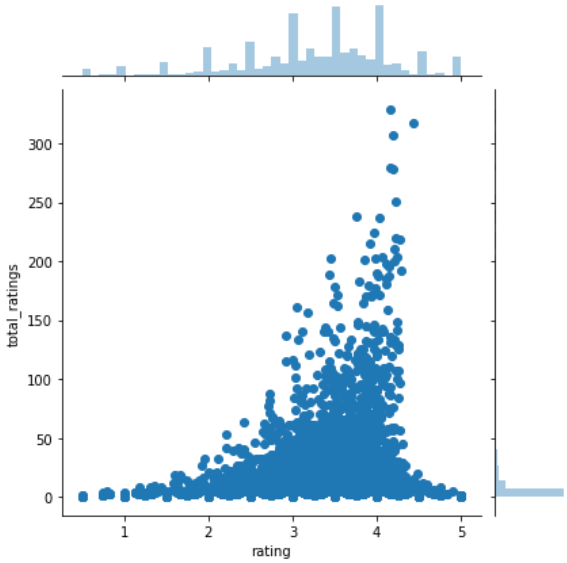

Fig. 1

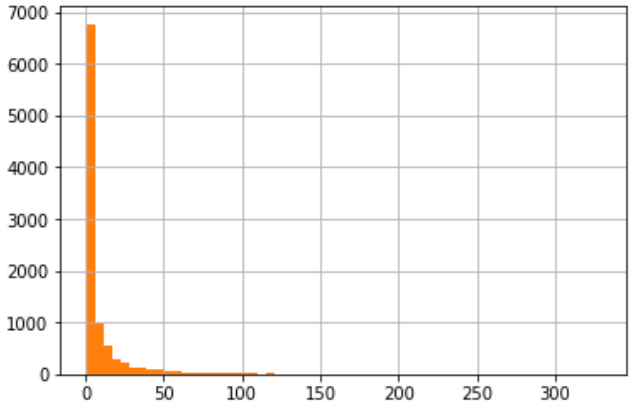

Fig. 2

The specific dataset we are using for preliminary testing is ml-latest-small.zip.

In order to get some initial impressions of the dataset being used, we generated the plot shown in Figure 1, which graphs the overall rating of each movie against the total number of ratings it received. What is most useful about graphing average rating against total ratings is that it allows us to see the most important trend in the rating data, which is that movies with a high number of ratings tend to also have higher ratings overall. This trend makes sense, considering that more popular movies which receive lots of ratings tend to also be very highly received by viewers. Good critical responses typically result in more popular films. Figure 2 further shows that a large majority of films have less than 50 or so ratings.

Based on this knowledge, we can now start to filter out recommendations based on whether they have a sufficient number of ratings. A quite high number of films have very high ratings but a low number of ratings overall. For example, if only a single person watches a film and gives it a perfect rating, then that movie will appear to be very well-received based on overall rating alone, but not necessarily based on number of ratings.

We want to only recommend films that are certified ‘good’ films by utilizing both number of ratings and overall rating, so we establish a threshold value such that only movies with a number of ratings higher than this value will be recommended. By looking at Figure 1 we can see that a majority of the ratings per movie are under 100. Therefore, we only search for movies that have at least 100 ratings.

After we filter movies based on number of ratings (or before; order is not important here), we can calculate the correlation between these movies by comparing the similarity of their tags and genres, and how often they match. When we generate a list of movies that have a high tag correlation with the source film, we can then apply the actual average rating of each film to filter a ‘top-n' list of movies which are best to recommend based on the source film. Evaluating tag similarity before filtering on rating is actually very important here, since we want to prioritize movies that are actually similar to the source film, and sorting based on ratings first means that more similar films might be lost in place of more overall critically-acclaimed ones. Alternatively, if we wanted to prioritize higher rated movies over more similar ones, we would flip this ordering.

Data Description

For this project we used the Movie Lens Dataset. The dataset was built by a research group at the University of Minnesota. It contains 100,000 ratings applied to 9,000 movies by 600 users. This data set is a continuously changing dataset and is continually being updated with more movies and metadata. As stated previously we are using the ml-latest-small subset of the Movie Lens dataset. As the name suggests this data set is among the most recently updated datasets, and is also the smaller of the more recent datasets. We can break down the dataset into the specific features which we are using for classification.

Movies.csv contains the primary movie information, which includes the title of the movie, it's corresponding movie ID (for use in indexing the other data tables), and it's assigned genre. The list of possible genres is given in README.txt (extracted from the Movie Lens website). Movies.csv contains three total features.

Tags.csv contains all tag information for each movie contained in Movies.csv. Each movie in Tags.csv is indexed according to it's corresponding movie ID in Movies.csv. These tags are generated by users, and are therefore not entirely consistent. However, we can use any potential matches between these tags to further influence the correlation between each movie. Tags.csv contains four features, including timestamp and user ID, which are not pertinent to our project.

Ratings.csv contains all rating information for each movie contained in Movies.csv. This includes each movie's individual ratings from users and includes four different features, including timestamp and user ID. We don't need either of these features since only the average rating for each film, and the total number of ratings for each film is pertinent in the scope of our project.

Results

We used a dataset to find the number of ratings and the average rating for each movie. We then used the ratings to find a correlation between the movies. We used a correlation function to indicate how the two variables fluctuate together. The movies with a high correlation are the movies that have the most similarity with one another. The correlation numbers lie between –1 and 1.

-1 Indicates a strong negative correlation, zero indicates no correlation (not similar at all) and 1 indicates a positive correlation. The most similar movies to Forrest Gump was Good Will Hunting; with the highest correlation of 0.48, Aladdin and American History X. The most similar movies to Pulp Fiction was Fight Club, with the highest correlation of 0.54, Kill Bill 1 and Trainspotting.

```
Top 3 most recommended movie out of 100,000 films most similar to
Forrest Gump
                          Correlation  total_ratings
title
Good Will Hunting (1997)     0.484042            141
Aladdin (1992)               0.464268            183
American History X (1998)    0.457287            129


Top 3 most recommended movie out of 100,000 films most similar to
Pulp Fiction
                         Correlation  total_ratings
title
Fight Club (1999)           0.543465            218
Kill Bill: Vol. 1 (2003)    0.504147            131
Trainspotting (1996)        0.437714            102
```