Netflix Recommendation based on IMDB

Nipun Bhatia, Pratyus Patnaik
Department of Computer Science, Stanford University.
nipunb, pratyus@stanford.edu

March 12, 2008

Abstract

In this report we discuss a methodology to recommend movies for the Netflix challenge using information from the Internet Movie Database(IMDB). Most of the current approaches in solving this problem have revloved around using machine learning and clustering. We explore the idea how using information about genres, actors and directors can prove helpful in predicting a user rating. The main idea we develop in this report is how to build a user-profile based on the movies rated by him and utilizing the information about those movies from IMDB. Using the techniques described we obtain an RMSE of 0.8658 on a random subset($\sim$ 900,000) of the probe data provided by NetFlix.

1 Introduction

The Netflix prize seeks to predict whether someone will enjoy a movie based on how much they liked or disliked other movies. Netflix has developed its own movie recommendation system: Cinematch. The RMSE of Cinematch on the test subset, using the training data set is 0.9525. The challenge is to develop techniques that can further improve predictions on how much a user would like a movie based on their previous movie preferences. Netflix has provided with the training set of numerous anonymous user movie rating data. In our work we have used movie metadata to characterize each user's movie taste. Based on this characterization of user's movie taste, we predict how much the user would like a certain movie. A comprehensive catalogue of information on the movies is the Internet Movie Database(IMDB), which we used have used for the purpose of movie characterization.

Most of the approaches dealing with this issue have used concepts from machine learning, collaborative filtering and data clustering . We have developed a hybird approach in which we characterize each user's movie taste based on genre, director and actors in the movie. Using meta-data of the movies from IMDB, we categorize the movies rated by a user in NetFlix into different genres, extract information about its actors and directors. Based on this extracted information and user's rating in NetFlix we build comprehensive profile of a user, which we then use for predicting movie ratings.

In the remaining part of the paper, we begin by describing our data-sources, followed by a detailed explaination of our methodology. We then discuss the implementation challenges faced by such a system and provide an intuitive effective solution to these challenges. We then address one of the major issues facing any recommendation system, lack of substantial information about user's choice. In our particular case, we have instances where we do not have information about user's movie taste. We propose addressing this issue by using correlation calculated between different genres using the IMDB dataset. Using this co-relation we predict user ratings for missing genres.We also use collaborative filtering to find out average ratings of all Netflix users for other genres, who have given a particular rating to a certain genre. We conclude by providing RMSE values on the probe data-set given by NetFlix.

2 Data Sources

The Netflix data is readily available for download from the Internet. This data contains user ratings for movies. There are 17,770 movies rated by

480,189 users. Number of movies rated by each user varies from 1 to 17,770. The data is arranged by user ID, Movie ID, rating for the movie and the date on which the rating was given. A separate file mapped movie ID to movie names. The probe file for testing the algorithm contained nearly 1.4 million, movie-user, couples.

We obtained information about movies from the Internet Movie Database (IMDB). The files are available for download at the IMDB site [1]. Of all the available information about the movies we downloaded the necessary files, based on the information attributes we decided to consider in our algorithm to charachterize a movie/movie taste of the user. These files contain information about the genre, directors, actors and ratings of all the movies in IMDB. Datasets, hence, collected was cleaned and arranged to simplify the implementation of datamining algorithm in the pre-processing phase. After pre-processing we had a list of 27 Genres, 389,492 Movie-to-Genre, 190,957 Director-to-Movie, 647,697 Actor-to-Movie and 225,184 Movie-to-Rating couples.

One of the major issues we faced was the difficulty to map movies in Netflix to corresponding movie in IMDB. We carried out exact string matching. As a result, we could find matches for about 10,000 Netflix movies in IMDB.

3 Methodology

Figure 1 shows the complete pipeline used for predicting the rating of a movie for a user.

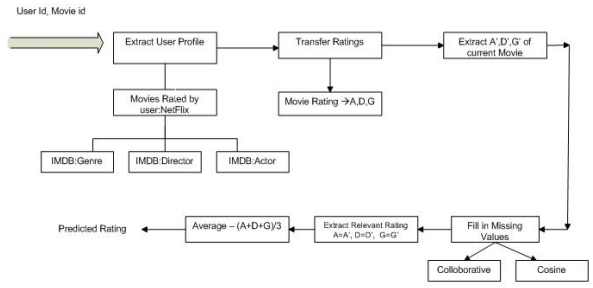

Figure 1: Pipeline for predicting of movie for a user

[1] http://www.imdb.com/interfaces#plain

1. The algorithm takes as input the user Id and the movie Id(*Current Movie*) for which the prediction is to be made. From the NetFlix data it retrieves all the movies rated by the user and their corresponding ratings.

2. For each of the movies rated by the user, it retrieves the genres, directors and actors from the IMDB data.

3. Transferring Ratings: After extracting the genres, directors and actors of the movie, the rating of the particular movie is transferred to these fields. If an user has rated multiple movies that lie in the same genre, the average rating for the movies is the rating for that genre for that user. Similar ratings for actor and director are calculated. Figure

4. Actor(A'), Director(D') and Genre(G') for the current movie are extracted from the IMDB data.

5. Fill in the missing values: In the situation that user hasn't rated any movie for a particular genre, the score for that genre would be 0. We explain in detail how we handle the missing values in the section Issues.

6. Extract Relevant Ratings: Extract rating for genres, rating[G] where G=G'. For Actors, we calculate a co-relation between the actor of the current movie,A' and the actors previously rated by the user. Using the co-relation values as weights we calculate the weighted mean. The weighted mean is the rating for the actor - rating[A']. The same approach is used for calculating the rating for the director - rating[D']. Calculation of co-relation is explained in the section below.

7. Predicted Rating: The predicted rating for the current movie would be (rating[A']+rating[G']+rating[D'])/3.

3.1 Calculating Co-relation between Actors/Directors

As the total number of directors and actors listed in IMDB is very large. In would have taken a long time to generate a cosine correlation between any two directors/actors. So, we decided to calculate

the co-relation between the directors/actors of the current movie and the directors/actors previously rated by the user on the fly.

We define directors/actors based on the following parameters:

1. Proportion of movies done for all the genre. Defined as nG/nT, where nG is the total number of movies directed/acted by an director/actor for a particular genre. nT is the total number of movies directed/acted.

2. Average IMDB movie rating for the all the movies of movies directed/acted by an director/actor (on the scale of 0 to 1).

Rating for directors/actors is calculated by taking the weighted average of the rating of all the directors/actors rated by the user, with cosine correlation between the them actors being the weight. eg. Let currentDirector be the director of the test movie. Director-1,.., Director-n be directors already rated by the user. Then the rating of the currentDirector is given by : Cosine(currentDirector,Director-1)* rating(Director-1) + ... + Cosine(current-Director,Director-n) * rating(Director-n)/Number of non-zero products in the numerator.

4 Implementation

One of the major hurdles for any algorithm attempting the NetFlix challenge is handling the humongous amount of user data. Our approach uses IMDB, which has a dataset even bigger than NetFlix. Effeciently processing these huge datasets in finite amount of time was a major design and implementation issue for our system.

The NetFlix data has over 100 Million unique (user Id, movie Id, ratings) triples. Our system requires that when a user Id for which prediction is to be made,comes in, we retrieve all the movies which have been rated by that user in the NetFlix data. Searching through the 100 Million entries across multiple files is not a reasonable option. To overcome this, we first sort the entire NetFlix data on the user Id field. As discussed in class, sorting can be performed using MapReduce. The sorted set is then partitioned into 9 mutually exclusive separate files. Each file for a particular range of user Id's, has all the movies and their ratings by those users. Also, each of the separate files is small enough to be kept as a hash map in-memory. Along with this, we store the starting and ending user Id of each file, along with the respective file names in a table. In a distributed setting, the master node would keep this table in memory. It would be associated with 9 slave nodes each of which keeps the partioned file as a hash-map, with the user Id as key. The map phase returns the appropriate file name which contains the user Id, as the intermediate result. The reduce phase takes this intermediate result, along with the user Id, and returns the movies and their ratings from the appropriate hash map.

For the RMSE calculations, there are 1 Million user Id and movie Id couples. We use a similar approach to what is explained above. We divide the probe file of 1 Million user's into 9 mutually exclusive sets. The range of user Ids for each of these files is equal to or less than the corresponding user profile file generated above. For eg. if the first split of the probe file has user ids from 10 - 9500, it is mapped to the corresponding user profile file which has ids from 1 to 10,000. As is intuitive, this approach allows us to parallelly process the nine split probe files.

5 Issues

In the course of our work, we faced with multiple issues that stymied the work. There were two main problems. We have briefly discussed them below and the ways we tackled them.

1. Handling missing values for genres: In several cases, the movie presented to be rated by the user, was from a genre of which the user had never rated a movie. In this case we could not trasfer the rating for the genre (which would be zero). So, rating were calculated in the following two ways and averaged.

   (a) Collaborative Matrices
   Average ratings by all Netflix users for other 26 genres, who have given a particular rating to a certain genre. Five (corresponding to each rating value) 27x27 matrices were generated. Missing genre rating is filled by averaging the rating of that genre (from the matrices) based

on the average rating of other genres the user has rated. Eg. genreRating[] = 1,2,0,0..,1 be the user's average rating for all the genres. Now,genreRating[0] corrensponds to Short; genreRating[1] corrensponds to Drama; genreRating[2] corrensponds to Comedy;....; genreRating[26] corrensponds to Film-Noir etc. Now, if we have to predict the rating for a movie of Comedy genre, which he hasnt rated yet. In the matrices we find the rating for Comedy genre by averaging the rating for the it given by all the users who have rated genreRating[0] (i.e. Short) as 1, genreRating[1] (i.e. Drama) as 2 and genreRating[26] (i.e. Film-Noir) as 1.

(b) Cosine relation between all the genre pairs in IMDB
We correlated all the genres using IMDB information. We assumed that all the movies made till date are present in IMDB. And IMDB classifies movies in multiple gernres as suited. Cosine Correlation between two genres was calculated based on how many movies were in IMDB were listed to be belonging to both the genres. Missing genre rating is filled by taking the weighted average of the rating of the genres rated by the user (correlation between the genres being the weight). Eg. genreRating[] = 1,2,0,0..,1 be the user's average rating for all the genres. Now rating for comedy Genre is given by (1xCor(Comedy,Short) + 2xCor(Comedy,Drama)+ ... + 1xCor(Comedy,Film-Noir))/Number of non-zero ratings by user.

2. Matching of movie names from NetFlix to IMDB. One of the other hurdle that we faced was difference in the name of the movies in IMDB and NetFlix. A movie appears by the name 'Character' in NetFlix and the same movie appears as 'Karakter' in IMDB. There certain other movie names that different arrangement of words. Major part of the problem can be solved using stop words and other popular natural language processing techniques. However, due to time constraints we were unable to effectively solve this problem. As a consequence out of the 17,770 movies in NetFlix, we could map about $\sim$ 10,000 movies. As a result the probe data set provided by NetFlix was reduced from 1.4 Million to about 900,000. However, as we point out in the next section, this doesn't have a major impact on RMSE results.

6 Results

In this section we present the results obtained for RMSE. The RMSE was calculated using the probe data-set provided by NetFlix. The probe data-set consists of about 1.4 Million user Id, movie Id couples. Since we were unable to map all NetFlix movies names to IMDB titles, our probe set was reduced to $\sim$900,000. It is important to note that the movies not mapped were totally random and we didn't in any ways control or influence which movies got dropped. Further, the table 1 & the table 2 show that RMSE values for each set of 100,000 users doesnot vary a lot. Thus it is not unreasonable to except that when the unmapped movies are included, RMSE won't change by a huge factor. Table 1 shows the RMSE obtained with different sets of the probe data-set when users movie choices were only categorized into genres. The average RMSE obtained is 0.8658. Table 2 shows the RMSE obtained when both genres and directors are used for categorizing user movie tastes. The RMSE obtained in this case was 0.916. When genres, directors and actors are used in conjunction, the RMSE obtained was 0.92.

Set	User,Movie Couples	RMSE
1	99674	0.864
2	99262	0.865
3	98979	0.862
4	99188	0.875
5	99262	0.865
6	99454	0.863
7	98558	0.860
8	99454	0.863
9	99350	0.872

Table 1: RMSE - Only Genre

Set	User,Movie Couples	RMSE
1	99674	0.915
2	99262	0.909
3	98979	0.913
4	99188	0.928
5	99262	0.916
6	99454	0.914
7	98558	0.914
8	99454	0.910
9	99350	0.923

Table 2: RMSE - Genre & Directors

7 Discussion

The metadata information about the movies seems a very strong indicator of an user's movie taste. The results of our algorithm are better than the results of the current best at Netfilx (RMSE 0.8675 obtained using ML techniques). Out of the information attributes we used, genres seemed to be more accurate in predicting the user's rating. We think that we can further improve the prediction by prudently selecting the weights for each of the parameters (namely, Genre, Directs and Actors) in the final computation of user rating, rather than simple average.

We would also like to incorporate better NLP techniques to increase the movie mappings between IMDB and NetFlix. An entity recognition match may increase the movie matches across the two datasets.

There are a some more improvements which can potentially have a positive impact on the results. One of them we think is to define confidence for the movie - genre relation. A movie lying in only one genre is more indicative of that genre than a movie which lies in couple of other genres too. Instead we giving 1 to every genre that the movie lies in, we could normalize it by dividing the number of genres the movie lies in. This could led to better co-relation between genres. Also, filtering out noisy user rating who could have given different extreme rating to the movies lying in the same genre.

All this will better connect people to the movies they love, the primary goal of Netflix. But before applying for the Netflix challenge we will have to look into the legal issues involved regarding the use of IMDB data for the Netflix Challenge.