Analysis of Netflix Data (Independent Study)

Abhishek Gupta, Abhijeet Mohapatra
under Prof. Jeffrey Ullman
Stanford University

June 8, 2009

1 Introduction

Today Recommendation systems [6] have become indispensable because of the sheer overload of information made available to a user from web-services(Netflix, IMDB, Amazon, Yelp and many others). Recommendation systems are a well studied research area. In the following work, we present our study on the Netflix Challenge [3]. The Netflix Challenge can be summarized in the following way : Given a movie m, predict the rating of a particular user u, given a list of user-movie ratings which may not contain the $u; m$ pair. The performance of all such approaches is measured using the RMSE (root mean-squared error) of the submitted ratings from the actual ratings. Currently, the best system has an RMSE of 0.8585 [4]. In our attempt, we tried a a variety of enhancements to the approach that we followed in CS345a class project. Our previous approach initially seemed like a very promising apporach since we obtained an RMSE of 0.8312 for close to 70% of the validation set data. Unfortunately, none of our variants lead to an RMSE value below the required value of 0.85626.

Section 2 gives some background on our approach last quarter by summmarizing our results for last quarter. Section 3 contains a detailed description of our current approach with Section 3.2 focussing on the two variants of collaborative filtering that we experimented with. Finally, Section 4 contains design choices that motivated us to choose parameters, error analysis and our final results.

2 Background

Last quarter, we implemented a hybrid approach for Netflix Challenge.Summary of the results is as follows :

Tab. 1 – Last quarter approach : Results on probe set

Description	RMSE	Number of user-movie pairs
The global rating of movies	1.0537	1408395
Content Based Approach based on genres, director	0.6955	172641
Collaborative filtering	0.9602	992681
The weighted sum of the 3 ratings above	0.8339	121722

However, there were certain pitfalls in this approach and that eventually lead to the current approach. (section 3)

3 Our Aproach

Last quarter, we applied a hybrid aproach but failed to identify classes of users exhibiting *special behaviour* for e.g. users who rate movies consistently, movies that are globally acknowledged good or bad. Instead, we focussed on the aggregated statistics. Therefore, we turned our focus towards an integrated approach, where our aim is to identify classes of *special* users and movies and devise heuristics to explain their behaviour. Further, we had not taken into account the variation of an user's ratings within a genre, which we feel could improve the quality of results from the collaborative filtering approach.

In the sections below, we describe how different ratings are assigned to different classes of users.

3.1 Special Classes of users

These are the *easy cases* where we can predict the user's ratings with a high degree of accuracy.

1. **New Users** : $|\{m|(u,m,r) \in \text{Training-Set}\}| < \delta_{new} \implies u \in$ New-Users
2. **Consistent Users** : $|\{m|(u,m,r) \in \text{Training-Set}\}| \geq \delta_{consistent} \wedge \sigma_{(u,m,r)}(u.r) \leq \delta_{\sigma_{consistent}} \implies u \in$ Consistent-Users.
3. **Genre-Consistent Users** : $|\{m|(\{u,g\},m,r) \in \text{Training-Set}\}| \geq \delta_{genre\_consistent} \wedge \sigma_{(\{u,g\},m,r)}(\{u,g\}.r) \leq \delta_{\sigma_{genre\_consistent}} \implies u \in$ Genre-Consistent Users.

Here σ stands for standard deviation. We remove these classes of users from the training set and build a *Pruned Training Set*. We apply collaborative filtering to this *Pruned Training Set*. For the cases, in which we don't have much data to build a comprehensive user profile or find a sufficiently similar set of users, we use *global rating* of the movie to predict ratings. Global rating is defined as,

$GlobalRating(m) = \forall(u,m,r) \in \text{Training-set}, AVG(rating)$

We also tried incorporating IMDB [2] data into the global rating of a movie. RMSE values with only Netflix data came out to be better than the RMSE values when we incroporated IMDB data.

3.1.1 New User

To predict ratings for a $NewUser\ u$ for a movie m, we tried the following two approaches

1. Predicted rating(u, m) = $GlobalRating(m)$
2. Predicted rating(u, m) = $(GlobalRating(m) + UserAverage(u))/2$
   where $UserAverage(m) = \forall(m,r) \in$ Training-set for a user $u, AVG(rating)$

The latter approach yielded poor results on the validation set. Therefore, we used the former approach in our final evluation. The exact value of the threshold is present in (section 4.1).

3.1.2 Consistent Users

We take the $\delta_{\sigma_{consistent}}$ sufficiently small. Our intention in doing so is to use the mean rating of this user for predicting ratings for the user. We have empirically verfied using the probe set that using this heuristic actually yields good results. The exact value of the threshold and a brief discussion on it is present in (section 4.2).

3.1.3 Genre-Consistent Users

To predict ratings of a movie m for such a user $u \in$ Genre-Consistent Users,

1. Obtain Genre(m) = the set of genres of movie m
2. Obtain the $ratings(u,g)$ where $g \in$ Genre(m)
3. Predict rating $AVG_g(ratings(u,g))$

3.2 Collaborative Filtering

It is generally the case in social networks(Facebook, Twitter etc.) that users tend to show common patterns of behavior as their friends and acquantaince's. This applies even to the domain of movie ratings, since the number of users are far greater than the number of movies, we expect users to have common movies and furthermore, exhibit similar responses to movies. Therefore, we would like to cluster *similar* users and study the behaviour of these clusters with respect to movie ratings.

Since the number of possible neighbors of an user can be large, Locality Sensitive hashing [4] seemed an obvious choice for clustering. Genres are the ideal choice of signatures since they are highly representative of the content of a movie. To build movie profile we have used IMDB [2] data. IMDB data contains 28 distinct genres so we define a user as a feature vector of 28 features where each feature corresponds to a distinct genre and is representative of a user's behaviour towards that genre. Two users are considered similar based on the closeness of the distance between their repspective dimensions.

In the following study, we have experimented with two clustering techniques

1. *Euclidean LSH* : For every user, each feature point is represented as user's average rating for the corr esponding genre. We use L2-norm as a distance measure. To compute *similar* neighbours we have used E2LSH [1] implementation of LSH [7]. If a user has not seen any movie of a genre, we assign a constant rating of 3 out of 5 for that genre assuming neutral behavior.
2. *k-NN Clustering* : Feature corresponding to a user u and genre g is represented the interval i.e. $(\mu_{u,g} - \sigma_{u,g}, \mu_{u,g} + \sigma_{u,g})$ and the distance is computed using the following scheme.

Algorithm 3.1: DISTANCEBETWEENTWOUSERS($user_1, user_2$)

```
$common\_genre\_list \leftarrow FindCommonGenres(user_1, user_2)$
$jaccard\_distance \leftarrow 0$
$g \leftarrow common\_genre\_list$
while $g \neq NULL$
    do $\begin{cases} jaccard\_dist \leftarrow jaccard\_dist + 1 - \frac{\bigcap\{(\mu_{u1:g}-\sigma_{u1:g},\mu_{u1:g}+\sigma_{u1:g}),\ (\mu_{u2:g}-\sigma_{u2:g},\mu_{u2:g}+\sigma_{u2:g})\}}{\bigcup\{(\mu_{u1:g}-\sigma_{u1:g},\mu_{u1:g}+\sigma_{u1:g}),\ (\mu_{u2:g}-\sigma_{u2:g},\mu_{u2:g}+\sigma_{u2:g})\}} \\ g \leftarrow g.next \end{cases}$
$num\_effective\_dim \leftarrow size(common\_genre\_list)$
return $(\frac{jaccard\_distance}{num\_effective\_dim} * \sqrt{num\_effective\_dim})$
```

For the first approach we directly used the LSH implementation. The LSH implementation requires two paramters as inputs. The first being the radius threshold within which it finds the nearest neighbours. The second being the number of neighbours that it reports within the hyper-sphere of radius r given as the input. Although LSH algorithm is independent of the distance measure but E2LSH package uses Eucllidean distance as a distance measure.

For the second approach, we could not use the default LSH implementation as it only considers features as values and does not handle features as intervals. Further, even by modifying the distance function we would run into the problem that the LSH implementation that we were using was implemented to probabilistically compute neighbours for an Eucliden distance measure and not for a Jacccard distance measure. Using this implementation for a non-eucliedean distance, would miss out a large fraction of good neighbours. We empirically established this fact before moving on to the Basic K-Nearest neighbour ($O(n*logk)$) algorithm.

4 Results

4.1 New Users

We used $\delta_{new\_threshold}$ of 10. By using the latter method, as described in section 3.1.1, we get an RMSE of 1.09. Our RMSE improves to 1.0537 when we use the former method. Around 3% of our probe set contains

New Users.

4.2 Consistent Users

We chose $\delta_{consistent}$ as 10. We initially set $\delta_{\sigma_{consistent}}$ to 0.3 and got an RMSE of 0.7612 for thse users on the probe set. We then tried higher thresholds to find consistent users. A threshold of 0.35 or higher gave an RMSE close to 0.84 for the validation set. We eventually settled for a threshold of 0.3. For a threshold of 0.35 or higher we were doing worse than collaborative filtering for these users.

For users consistent within a genre we used $\delta_{genre\_consistent}$ of 5. There are 2% of users who give give consistent rating for atleast one genre.

4.3 Locality Sensistive Hashing

Even with exact neighbours, our results did not improve much as compared to our first approach in which we were not considering interval of ratings for every genre. This lead us to do some further data analysis. We observed that around $1.5K$ users were such that the standard deviation of their ratings was less than or equal to 0.3 and almost all the rest of the users had a standard deviation almost the same i.e. between 0.8 to 1.1. With this new insight, we concluded that since most users in the *pruned training set* have more or less similar standard deviation of ratings so it might not be worthwhile to consider them as intervals of ratings within every genre and that only considering their mean ratings for every genre migt be just as good a measure.

Tab. 2 – LSH results on probe set

Sl No.	Data Set	RMSE	number of user-movie pairs
1	Training Set - Consistent Users	0.992197	1048274
2	Training Set - New Users	0.987475922	1033740

Tab. 3 – Overall Results on qual set with 28 features

Sl No.	Number of neighbours	Radius threshold	RMSE on Test Set
1	250	2.3	1.03
2	500	2.6	1.0282

Tab. 4 – LSH for pruned training set on Probe Set

Actual Rating	RMSE	number of user-movie pairs
1	2.3137	40208
2	1.4143	95748
3	0.7450	273000
4	0.6213	359142
5	1.0875	261646
Overall	0.9886	1029744

RMSE for ratings ≥ 3 is 0.8191 and it is predicted for 87% of the pruned training set. RMSE for ratings ≥ 2 is 0.8941 and it is predicted for 97% of the pruned training set. We can see that the user-movie pairs with an actual rating of 1 severely affect our overall RMSE.

Our final results came out to be even poorer than the cinematch's score of 0.9515. We suspect that this might be due to the following reasons :

1. We were performing rather poorly on the probe set for movies which were rated 1 by the users. But for all other movies in the probe set our RMSE was close to 0.88. In our next section, we describe in detail about this curious case of ones.
2. While fine tuning our method on the probe set we were able to compute neighbours for a larger fraction of the probe set but when we used the same thresholds for the qual set we were able to compute neighbours for close to 70% of the data only. On increasing the threshold to 2.6 we were able to compute neighbours for 79% of the data. We have finally increased the threshold to 3.0 and are awaiting the results. For the probe set for which we can neither predict ratings via consistent users within a genre, nor by new users and nor by LSH, we simply take the average of users rating and movie's global rating.

4.4 Curious Case of ones

After running our final setup we got an RMSE of 0.8923 on the probe set for all user-movie pairs except for the user-movie pairs which had an actual rating of 1. We found out that 73211 (around 5%) user-movie pairs in the probe set were rated as all 1s. Some of the interesting statistics about *such* user-movie pairs is described below :

1. Number of unique users within this user-movie set is 9205.
2. Number of unique movies within this user-movie set is 61358.
3. Percentage of user-movie pairs with actual rating as 1 but with the global rating of the movie ≥ 3.0 is around 81% of total user-movie pairs with 1s, whereas percentage of user-movie pairs with actual rating as 1 but with the global rating of the movie ≤ 1.85 is 0.25%. This seems to suggest that a large fraction of these movies are such that they are not in general rated ones by most users.
4. Number of unique users with an average rating ≤ 1.5 is only 158 and the number of user-movie pairs corresponding to these users is only 0.61% of all such use-movie pairs. This seems to suggest that a lot of the users within the above mentioned 73211 user-movie pairs are such that they don't normally give ones but there is something about these movies which made them give rating 1 to these movies.

Even up till now, we have not been able to identify why these movies are rated as ones even though average rating the movies of over 81% of these user-movie pairs are over 3.What is even more surprising is that most user's who rate them as ones are in general not the kind who give ones to these movies. Another noteworthy point is that even though the fraction of such user-movie pairs is actually very small but they affect the overall ratings greatly.

Further, there was a user who has rated around 16560 movies. if a user watched roughly 9 movies per day continuously for 5 years then and only then can the user watch 16560 movies. Since, we only have data for user's ratings given in 5 consecutive years, this clearly suggests that the user has simply rated a lot of the movies and has not actually watched them. There were 12 other users who have rated more than 8000 movies. We considered users with average rating ≤ 2.5 and predicted for them by taking the average of their average rating and movies' average rating. We got an RMSE of 1.36 for 8800 user-movie pairs in the validation set. Earlier, we were getting an RMSE ≥ 2 for these set of users. Our main effort here was to somehow get an RMSE close to 1.3 for these sets of users for these sets of movies. [5]

5 Future Work

Références

[1] E2lsh. http://www.mit.edu/~andoni/LSH/.

[2] Imdb. http://www.imdb.com/.

[3] Netflix challenge. http://www.netflixprize.com/.

[4] Netflix leaderboard. http://www.netflixprize.com//leaderboard.

[5] Netflix prize. http://en.wikipedia.org/wiki/Netflix_Prize.

[6] Gediminas Adomavicius and Er Tuzhilin. Toward the next generation of recommender systems : A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, 17, 2005.

[7] Aristides Gionis, Piotr Indyk, and Rajeev Motwani. Similarity search in high dimensions via hashing. In *VLDB*, 1999.