Netflix Rating Prediction Midterm Report

Shupeng Niu : sniu
Kartikeya Pharasi : kpharas
Alok Kucheria : akucher

November 16, 2015

1 Background

The Netflix Prize was a contest Netflix sponsored, which sought to substantially improve the accuracy of predictions about how much someone would enjoy a particular movie and rate it based on their previous movie preferences.The participants were provided with a Dataset consisting of 100,480,507 ratings that 480,189 users gave to 17,770 movies. The participants who developed a system that beat the predicting accuracy of Netflix's own algorithm, Cinematch by bringing about a 10% improvement over it were declared as the ultimate winners.

2 Introduction

2.1 Objective

Predict the rating a user will give a movie based on the movies that user has rated, as well as the ratings similar users have given to similar movies.

The overview of the steps involved to achieve this goal are:
1. Discover clusters of similar movies and similar users.
2. Find various parameters relevant to prediction of ratings.
3. Define methods of prediction and select the best possible method.
4. Make the prediction for the rating a user would give to a movie not yet rated by them.

2.2 Description of the Dataset

1. Training data set: This is a directory containing 17,770 files with a single file per movie. The first line of each file contains the movie id, each subsequent line in the file corresponds to a rating from a customer, as shown below:
movie id
[User ID,Rating,Date]

- The User IDs are integers
- The Ratings are on a five star scale from 1 to 5.
- Dates have the format YYYY-MM-DD.

```
1
1488844    3    2005-09-06
822109     5    2005-05-13
885013     4    2005-10-19
...        ...  ...
```

2. Movie Titles data set: This is a dictionary file which consists of information about each movie

i.e the movie id followed by year of release and title of the movie.
[Movie ID, Year of release, Movie title]

- The Movie IDs are integers
- The Year of release is integer type
- The Movie titles are string type

```
1    2003    Dinosaur Planet
2    2004    Isle of Man TT 2004 Review
...  ...     ...
```

3. Probe data set: This file contains some Movie IDs on one single line followed by some User IDs.

```
1:
30878
...
10:
1952305
...
```

4. Qualifying data set: The first line is the Movie ID followed by a colon and subsequent lines consist of User ID and the rating date.

```
1:
1046323    2005-12-19
...        ...
10101:
797782     2005-12-06
...        ...
```

3 Method

3.1 Problem analysis

Our objective is to predict the rating a user will give to a movie based on the movie ratings given by similar users to similar movies.

The Tasks involves:
1. Splitting all the movies and users into clusters, containing movies and users having some similarity so that they can be classified as similar movies or similar users.
2. Based on the movie ratings given by similar users to similar movies, develop a method to give the final prediction different users would give to different movies they have not rated yet.

3.2 Clustering

3.2.1 Parameters selection

Based on the information given in the Netflix dataset, for a single movie, we have all the users who rated this movie, the ratings, the date of rating, the release date of the movie and the movie title.

- The movie IDs, movie title and user IDs are not chosen as the parameters for clustering here, but are used for identification.
- The release date itself is trivial, but when combined with the rating date of users, it will give us information regarding when the user usually rates the movie after its release.
- Apply mathematical methods to the ratings. Given a movie, we could get useful information such as the number of ratings(which implies the popularity),the standard deviation, average,

difference, etc. Similar information could be found for each user after extracting the rating information for the user.

After comparing all these factors, we decide to use the following parameters to cluster movies and users.
Movies: the number of ratings, the average ratings, standard deviation for each movie.
Users: the number of ratings, average ratings and standard deviation for each user, the time delay(year) between the rating year and the release year of each movie.

3.2.2 Clustering algorithm

Our first step is pre-processing the entire dataset and extracting the parameters we need for all the movies and users.For the midterm report, we used a smaller size of the training data set. We use 25 movies instead of the entire data set. This smaller dataset will help in making decisions which would otherwise take a long time on the entire dataset.

Based on the available data, we go for clustering to refine our rating prediction system. There are multiple ways to cluster data, namely, Connectivity based, Centroid based, Distribution based and Density based.

Connectivity based, eg. Hierarchical clustering cannot be used as it results in overlapping clusters. Nor is the nature of our data hierarchical. Even density based clusters have the drawback of improperly defined borders.That leaves us with Centroid based and Distribution based clustering. As our data is in a small range, the average values can be close. This might lead to over-fitting if we use Distribution Clustering.

Hence we go with Centroid Clustering. As clusters are easy to define, we can overcome the biggest drawback of K-Means.

3.2.3 K-Means

K Means is used when we have to define k clusters for n observations. Our data and the prediction required hinges mainly on ratings. They are divided on a scale from 1 to 5. This means we already have a basic classification ready. We can easily cluster our observation based on these ratings or their subsets for finer classification.

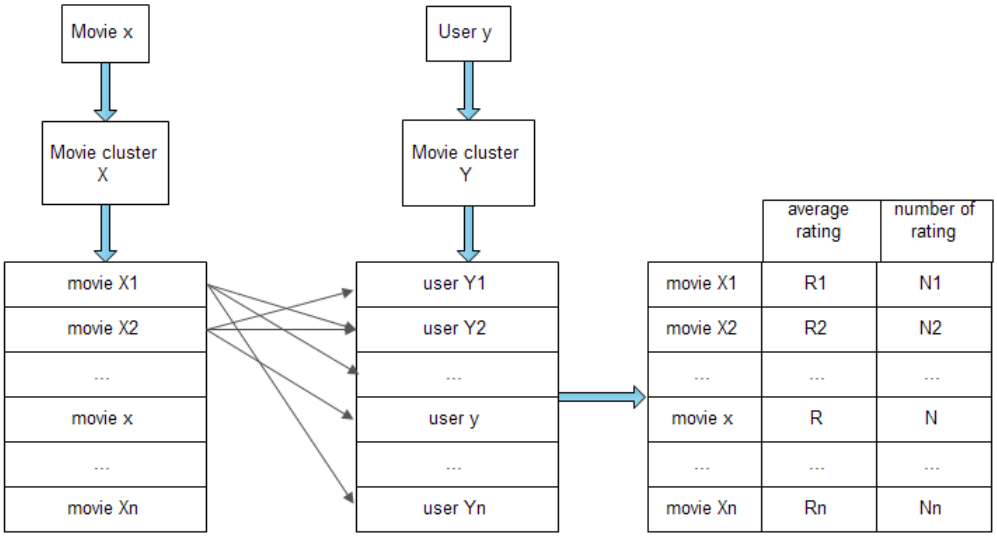
Thus, we chose the average, standard deviation and number of ratings to make clusters of movies. Similarly, use the user information data like average rating, standard deviation of ratings, number of ratings and the difference in the time they rate the movie after its release. Our final output, a rating prediction will finally have values ranging from 1 to 5.

3.3 Prediction algorithm

After clustering, we need predict the movie rating from specific users for specific movies (which they have not yet rated).

Firstly, we would find the movie cluster and user cluster which the user and movie belongs to. Then for each movie in the movie cluster, we look at the movie ratings from users in the user cluster. Now we have, for all the similar movies, the ratings from similar users.

Based on the average rating for each movie, the corresponding weight is assigned according to the number of ratings each movie gets. We can get a prediction by calculating the weighted average of all these ratings of similar movies from similar users.



$$Predicated\ rating = \frac{R1 * N1 + R2 * N2 + ... + Rn * Nn}{N1 + N2 + ... + Nn}$$

For user y, since it may not be in the center of the cluster, we can add a correction to the prediction. For all the movies user y has rated in the movie cluster,(say movie X2,X2...), the ratings are r1,r2... We define

$$\alpha = \frac{(r1 - R1) * N1 + (r2 - R2) * N2 + ...}{N1 + N2 + ...}$$

Then

$$Predicated\ rating = \frac{R1 * N1 + R2 * N2 + ... + Rn * Nn}{N1 + N2 + ... + Nn} + \alpha$$

4 Experiment & Results

We have made clusters on a smaller dataset for both the movies and the users using k-means clustering.
The number of clusters for the entire dataset will be determined according to the accuracy.
Basic code for data extraction and storage.

```
dataset<-read.table(text=readLines(file)[-1],header=FALSE,sep=",")
ratings<-dataset[,c(2)]
average<-mean(ratings)
standard_deviation<-sd(ratings)
number<-length(ratings)
```

Extraction of Movie summary:

```
Movie_Summary_temp<-data.frame(Movie_ID=i,Average_Movie_Rating=average,
Standard_Deviation=standard_deviation,Number_of_Ratings=number)
Movie_Summary<-rbind(Movie_Summary,Movie_Summary_temp)
rm(Movie_Summary_temp)
```

Extraction of User summary:

```
release_year<- Movie_Titles[,c(2)]
rating_year<- substr(dataset[,c(3)],1,4)
difference=strtoi(rating_year[j])-release_year[i]
user_temp<-data.frame(Movie_ID=i,Rating=ratings[j],Difference=difference)
temp<-get(paste("User",user_id[j],sep=""))
```

```
temp<-rbind(temp,user_temp)
assign(paste("User",user_id[j],sep=""),temp)
rm(user_temp)
rm(temp)
user_info_summary_temp<-data.frame(User_id=user_summary[i],
Avg_Rating=average1, N_Rating=number_of_ratings,
SD_Rating=standard_deviation1, Date_Difference_Avg=average_date_diff)
user_info_summary<-rbind(user_info_summary,user_info_summary_temp)
rm(user_info_summary_temp)
```

K-Means Clustering using R functions:

```
(kc<-kmeans(Movie_Summary[,c(2,3,4)],5))
(User_cluster<-kmeans(user_info_summary[,c(2,3,4,5)],5))
```

4.1 Output

4.1.1 Movie Cluster

When setting the number of clusters to 5,the K-means clustering algorithm returns 5 clusters of sizes 12, 8, 1, 2, 2
One of the returning value is a clustering vector:

Clustering vector (For the first 10 values):

1	2	3	4	5	6	7	8	9	10
2	1	5	1	2	2	1	3	1	1

As shown above, the clustering vector contains the index of the movie or user and the cluster it belongs to.We can easily find the cluster a movie or user belongs to using this data.

4.1.2 User Cluster

Clustering vector(for random 10 values):

33685	33686	33687	33688	33689	33690	33691	33692	33693	33694
5	3	3	5	5	5	3	3	3	3

5 Conclusion

We have devised a method to predict the rating a user will give to a movie which he has not rated yet based on the method of prediction described above.

Current Computations were made on a smaller training set. We now plan to apply clustering on the entire dataset and make the prediction based on the resultant clusters. The cluster size will be decided based on various iterations to compute more accurate results.

References

[1]Ted Hong, Dimitris Tsamis. *Use of KNN for the Netflix Prize*
[2]Cremonesi, Paolo, Yehuda Koren, and Roberto Turrin. *Performance of recommender algorithms on top-n recommendation tasks.*

Appendix

Revisions
Earlier we had planned to cluster the movies based on the genres which we would have retrieved from public domain such as IMDB. But since there could be a movie which could have multiple genres tagged to them, they would be present in multiple clusters. This would have created a problem. Therefore, we decided to change the parameters based on which we would cluster the movies.