Rating Prediction and Recommendation for new movies

[A solution to the cold start problem in recommender system]

Shipra Gupta
Stony Brook University
700 Health Science Drive
Stony Brook, NY 11790
shipra.gupta@stonybrook.edu

Lokesh Gyanchandani
Stony Brook University
700 Health Science Drive
Stony Brook, NY 11790
lokesh.gyanchandani@stonybrook.edu

ABSTRACT

The internet today has an overwhelmingly large amount of information available, and with such abundancy of data, the problem of finding relevant content becomes tougher and time consuming for the user. This has given rise to the necessity of good recommender systems. Recommendation system is a kind of an intelligent technique which provides personalised suggestions to people, aiming at saving their time and also improve the quality of decisions made by them.

Various techniques including collaborative and content based filtering have been applied to generate recommendations for movies and some of them have even showed how to integrate the two techniques to generate a hybrid approach which can be used to generate more effective results.

This project aims to follow the latter approach in order solve a subset of the cold start problem and predict the ratings and the recommendation scope of a recently released movie for which there are no ratings at present. Experiments have been conducted on IMDB and MovieTweetings data to analyse the characteristics of our technique. The results show that our approach produces quality predictions and contributes effectively to solving the item side cold start problem.

General Terms

Algorithms, Performance, Experimentation

Keywords

Recommender System, Rating Prediction, Cold Start

1. INTRODUCTION

Even though movie recommender systems are quite useful, they suffer from a major drawback called "cold start" in which a relatively new movie will generally not be put forth (recommended) to any user, even if it would turn out to do well in the future, because of lack of any prior ratings for that movie.

We want to design a system such that when new movies, (which have not yet received any ratings from the community) are advertised in a catalog, they should automatically be assigned a rating, based on the ratings assigned by the community to similar movies. In order to offer a potential solution to this problem, we aim to exploit the metadata(content) of the movie such as director, genre, actor, etc to gain an insight into its content and try to predict its rating even before it is released. The data consisting of the user ratings of the movies released in the past is also equally utilized in an effort to make our prediction more accurate. As a result, the ratings predicted for any user based is the combined result of the ratings obtained from the past movies which have similar metadata characteristics as well those users who are similar to the concerned person in terms of their rating habits.

Since we have most of the metadata available for any upcoming movie other than the its user ratings, our recommender system has been designed based on this available information. We recommend movies based on user's preferences, his neighborhood preferences and the predicted rating from our system. By appropriately weighting these, a ranked list of movies is recommended to the user.

2. PRIOR WORK

A lot of work has been done in the past to solve the cold start problem, each approach has its own pros and cons. But "cold start" still remains an open problem and needs research.

"Combining content-based and collaborative filters in an online newspaper." by Claypool, Mark, et al [5] uses a hybrid systems and applies collaborative and content based filtering in a sequential manner, and then combines the two results by assigning weights to each. It is not very clear though how the weights are decided.

"Fab: content-based, collaborative recommendation." by Balabanovic, Marko, et al [1] only takes features into account and ignores the user rating data. Such a technique has certain drawbacks, and hence it can not provide very good quality costumized recommendations.

"A framework for collaborative, content-based and demographic filtering." by Pazzani, Michael J. et al [2] focuses on using demographics information of an user to assist in

cold-start problems. This paper assumes that users from the same demographics will have similar preferences.

"Variational Bayesian approach to movie rating prediction." by Lim, et al [4] tries to follow a bayesian approach of low-rank decomposition for collaborative filtering.

In our project, we are trying to develop over a *hybrid* technique which takes the item features into account for collaborative filtering and this is combined with the user rating data for items to calculate similarity between items [3], which solves the cold start problem and improves the prediction quality.

3. METHOD

An overview of the hybrid approach followed in this project is given in Figure 1. This is followed by a detailed description of our method.

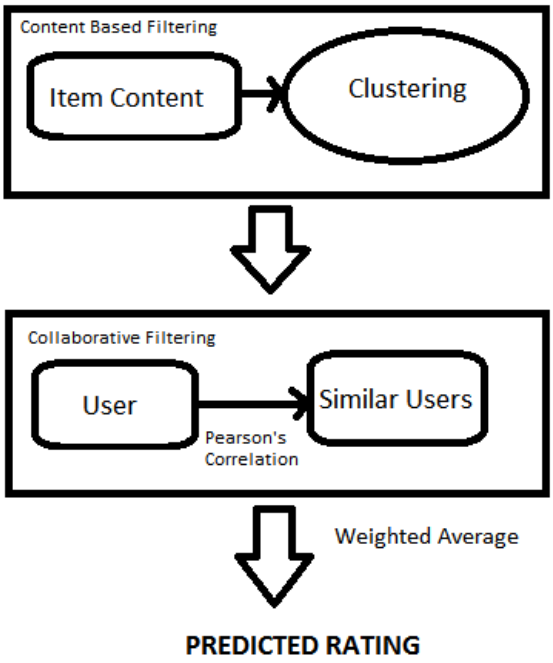

Figure 1: Approach Followed in the Project

In our project, we have used the following method to obtain the results once we were done with cleaning and organising the data that we pulled:

- Based on the features (including genre, director, producer, actors and writer), group the movies in clusters using simple **K-means**.

  [Depending on number and type of features included, the results vary. In general, the more the features included, the more accurate are the rating predictions obtained by this method]

- Generate an **item-rating matrix** which stores the rating R_{uk} of user u for movie k.

  [We used the user rating data that we cleaned after pulling it from the MovieTweetings database]

- Calculate **user-user similarity** using Pearson's correlation coefficient.

  The similarity measure considers the number of common movies rated by two users and also the kind of rating given by them. It is calculated using equation (1):

$$sim(u,k) = \frac{\sum_{i \in C_{u,k}} (r_{u,i} - \bar{r_u})(r_{k,i} - \bar{r_k})}{\sqrt{\sum_{i \in C_{u,k}} (r_{u,i} - \bar{r_u})^2 \sum_{i \in C_{u,k}} (r_{k,i} - \bar{r_k})^2}} \quad (1)$$

  Here, $sim(u,k)$ is the similarity measure between users u and k. $C_{u,k}$ is the intersection of items both rated by user u and k, r_u denotes the average rating value of the user u across all co-rated items in $C_{u,k}$

- Identify the cluster,(say C_n) to which a new movie which has no ratings belongs to.

- The rating for that movie for a user u, is then calculated using the average of the ratings given by him for the movies present in that cluster that he has watched and the weighted average of ratings calculated in the same way for the top-N similar users to user u.

- Thus an approximate rating for the movie is obtained, the formula used for prediction is given by equation (2).

$$P_{uk} = \frac{1}{2}(\overline{R_u} + \frac{\sum_{i=0}^{N} \bar{R_i} * sim(u,i)}{\sum_{i=0}^{N} sim(u,i)}) \quad (2)$$

  Here, P_{uk} is the predicted rating od a user u, for a new movie k. $\overline{R_u}$ is the average rating given by user u for movies present in the same cluster as the new movie. $\overline{R_i}$ is the average rating given by a user i to movies present in the same cluster as the new movie. $sim(u,k)$ is the similarity measure of user u with user k. We calculate this factor for the top N similar user to user u, which we obtain from user-user similarity matrix.

- Based on the prediction results calculated, recommend top N movies to a user.

To sum up, we can say that our technique uses the content information of a movie as well as user rating similarity to calculate rating of a new movie for an user, which he has not actually rated; using these ratings, good quality recommendations can be made.

4. AN EXAMPLE OF OUR APPROACH

- **User Id:** 3468
- **Movie:** Ice Age (2002)
- **Actual rating:** 4

A simple procedure of our approach is as follows:

1. Based on the movie contents, we generated clusters using K-means algorithm. After clustering, the movie *Ice Age (2002)* belonged to cluster 5.

2. Using Pearson's Correlation Coefficient, we calculated the user-user similarity and as a result we found the top 5 most similar users to our user, 3468, to be user ids $7477, 2477, 1772, 2848, 7788$, all the similarity measures equal to 1 in this case.

3. We then find the average ratings for all the movies in cluster 5, rated by the users including our user, 3468 and his top-5 similar users.

4. From our data, we found the ratings to be: 4.18(for 3468) and $4.63, 4.22, 4.07, 3.93, 3.88$ for the similar users.

5. Hence, the formula for prediction is applied and we get a rating as follows:

$$PredictedRating = \frac{1}{2}(4.18 + w_n)$$

Where w_n = Weighted average of neighbour ratings

$$w_n = \frac{(4.6 + 4.2 + 4.07 + 3.93 + 3.8)(1)}{(1 + 1 + 1 + 1 + 1)} = 4.12$$

$$\textbf{Predicted Rating} = \frac{1}{2}(4.18 + 4.12) = 4.15$$

In this manner, we calculate the ratings of a new movie. We can observe that the rating obtained for the sample movie we took in the example is also very close to the actual rating.

5. EXPERIMENTS

For the evaluation of our method, we used the dataset from IMDB and MovieTweetings to conduct experiments. We fetched the logistics (e.g director, actor, genre, plot summary) of the movie from IMDB using IMDbPy, a python script. And we took the user rating data for the movies from the Movie Tweeting dataset. We worked on approximately 3k movies and 5,70,000 user ratings. We randomly selected 50 movies and then deleted all their ratings and treated them like new movies for the experiments.

Initially the data gathering and cleaning was challenging and there were few difficulties in parsing and loading the data. IMDB data was organised in very small chunks, like directors.list, ratings.list etc. and different naming standards and norms were used for genre, movie names etc, which made the cleaning of data a difficult and overall time consuming task.

In the beginning we started with taking just the user ratings for computing movie similarity, but that did not give very good results and we found it to be a poor strategy as it does not capture much of the underlying relationships. Hence we began with considering the genre and director of the movie to further calculate similarity in two movies, which improved the results a bit. And finally, we moved to adding more features of a movie to the set including the producer, actors and writer, to basically observe what effect(positive or negative) does it make to the prediction quality of our rating system.

In all of the above cases, we clustered the data using *K-means*, over different values of K ranging from 2 to 20.

K-means clustering algorithm used for clustering the movies is described as follows:

1. To partition the movie-feature sets, represented by 2-dimensional real vectors into a partition of k sets so as to minimize the within-cluster sum of squares.

2. We input the number of clusters and item features.

3. Initially k movies are chosen arbitrarily as the centers of initial clusters.

4. Every movie is assigned to a cluster by measuring the similarity based on the mean value of cluster objects. Cluster mean value is recalculated.

5. Step 3 is performed repeatedly until it converges and the reassignments stop or come to a very small value. The final clusters are hence obtained as the output.

We then created the item-rating matrix from the ratings data and then calculated user-user similarity by employing Pearson's correlation coefficient. Based on the cluster to which a new movie belongs to, we calculated the average ratings of a user for movies he has watched in that cluster and also the same average ratings for his top-N similar users. Using all these values, we calculated the predicted rating taking the average of these two. For all the various attributes that we experimented upon, we calculated the Mean Absolute Error, using the formula:

$$MAE = \frac{\sum_{i \subset D} |p_i - r_i|}{D}$$

Here, p_i is the predicted rating and r_i is the actual rating. D is the test data collection and $i \subset D$
Once we obtained the predicted ratings for new movies, we took the top-N m for a user and generated recommendations.

6. RESULTS AND FINDINGS

In our technique, while calculating the predicted ratings of a movie, we have considered both, the content information (attributes) of a movie as well as user ratings for similar movies. We started with clustering based on few movie-attribute values - producer and genre of the movie. We then included more attributes such as the director, actors, writer, producer and genre of the movie, in order to observe the effects of movie attributes on the rating predictions. Through our experiment results, we found that correct and meaningful attributes of movies improve the performance of the recommender system.
Figure 2 shows the comparison of error in the predicted ratings upon changing the attributes being considered for clustering. It can be observed that as we include more and more attributes that play a role in the success or failure of a movie, the error keeps on decreasing and we get more and more accurate ratings for a given user.

In Figure 3, we have compared the ratings obtained by taking different attribute againts the actual rating for a movie using the bar graph. It can be seen from the graph that the ratings predicted after considering all the extracted attribute values is more closer to the actual rating than the predicted ratings using fewer attributes.

The Mean Absolute Error Percentage is given in Table 1.
A lot of interesting observations can be made from the above obtained results, like, who or what plays more significant role in the success of a movie than the other, we will

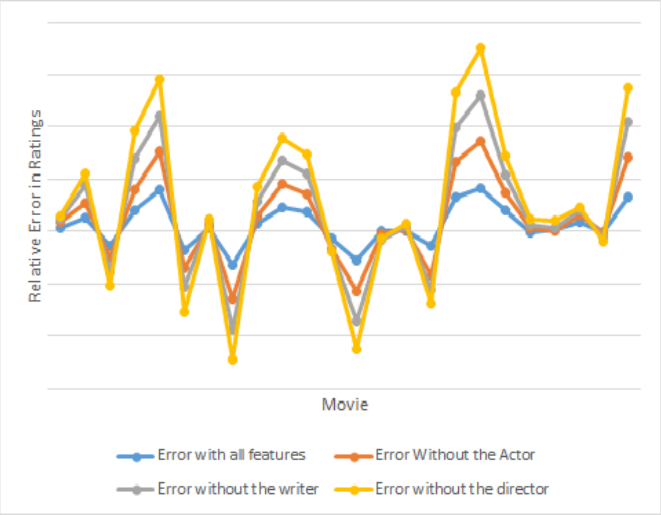

Figure 2: Comparison of error found using different features

Table 1: Mean Absoulte Error

All Features	No Writer	No Director	No Actor
24.4%	22.86%	26.78%	27.24%

discuss this further in our conclusion.

We took a random user from our dataset and generated ratings for him for some of the movies. The result of the ratings can be observed from Table 2. The predicted ratings are quite close to the actual rating given by the user. However, there could be some outliers, like for example in this case, "Broken Arrow" was given a low rating (2) by the user, whereas our method generated the predicted rating as 3.04. This can happen in cases such as, when a user who generally likes a particular genre of movies, say "Action", did not like an Action movie for some reason. In such cases, our system will give a good rating for a movie because of the genre being Action, which has been found to be liked by the user based on his past ratings. (One factor that might have reduced the amount of introduced error could be inclusion of the year of release of the movie here, which we have not considered for our experiments. This can be understood intuitively because movies made in a particular "era" may not be liked by someone who was born in a different "era")

Table 2: Sample Predicted Rating for a User

Movie	Actual Rating	Predicted Rating
Broken Arrow(1996)	2	3.04
Men in Black(1997)	4	3.961
Almost Famous(2000)	3.5	3.38
Bewitched(2005)	3	3.17

After the calculation of predicted ratings, we sorted the new movies based on the ratings for each user and took top-N movies from the sorted list and recommended those to the user. Upon verification, we found that these recommendations were indeed good quality recommendations and

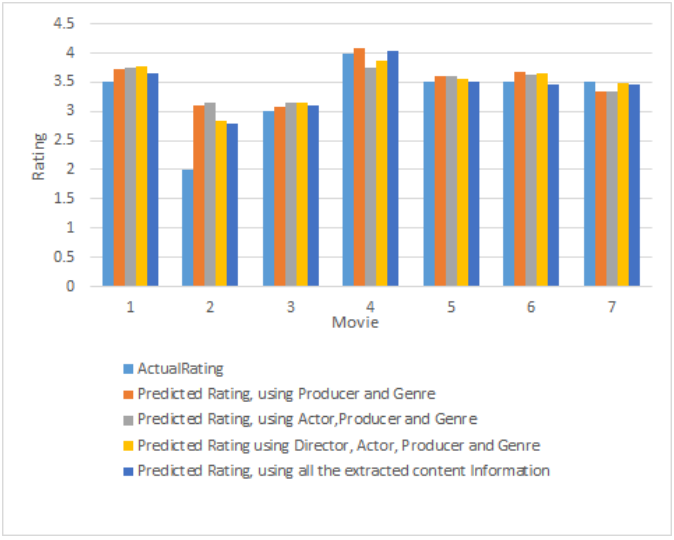

Figure 3: Comparison of predicted ratings using different features with the actual ratings

matched the user's preferences. Hence, we successfully recommended the new movies to users.

Part of our experimentation also involved keeping all the movie attributes in the clustering input and then varying the value of number of clusters that we were given as an input for generating the clusters. K-means clustering was run on a range of values and tested with changing size of clusters and the corresponding prediction error(Mean Average) of a new movie over a sample of 50 such movies were analysed. The results are plotted as shown in Figure 3. As it can observed from the plot, error was large when smaller values of K (2 or 3)were used. However, it kept on decreasing till K = 9, where it was minimum and on further increasing the value, the error started growing again. It shows us that even the size of cluster does play a vital role in deciding the accuracy of a rating predictor.

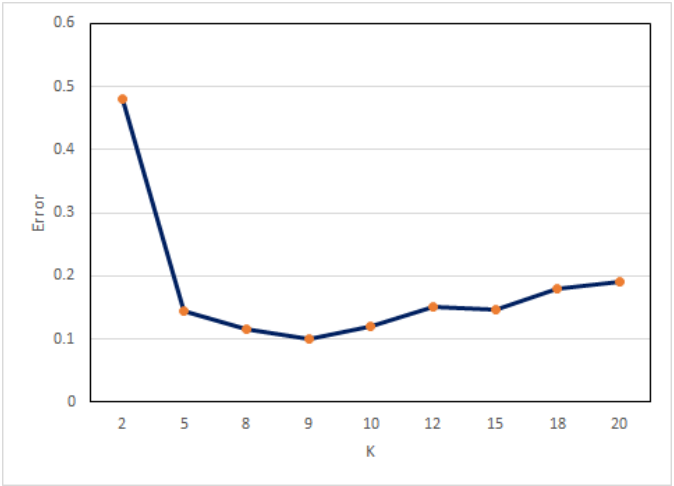

Figure 4: Mean Absolute Error over different values of cluster size,K

7. CONCLUSION

In the project, we have presented a method to solve the item side cold start problem by using a hybrid approach which takes both, the content information of the item (in our case, movie) and the user ratings for movies similar to that for the purpose of predicting the ratings. And we have found that our method gives good quality results. As a conclusion to this project we would like to comment on few of our observations about the contribution of different parameters of a movie on its ratings. Introduction of actors without any filter introduced irrelevant information into the clusters. Intuitively, the work of directors, writers and producers is 100% involved in the movie unlike actors and actresses, who have a different participation based on the importance of the character. We have seen that our experimental results agree with these intuitive facts. Broadly speaking, by including more and more meaningful features of a movie in our technique, we can keep on improving the results.

Rating predictors and recommenders require lots of statistical techniques to achieve minimum error, and therefore this problem becomes open ended in a sense and there always remains a scope of improvement. As an extension to our work, in the future, we can consider the actors/actresses dataset and evaluate an individual's amount of work on each movie (leading role versus supporting role or just a cameo in the movie). Data can be extracted from wikipedia to get the plot of the movie and then comparisons on how many times the actors role appears within the description to get the amount of work can be made. Our assumption is if the actor's character appears often in the plot, then that is an indication of how important his role is to the movie, and hence decides the weight that the actor attribute should be given while calculating the ratings.

8. REFERENCES

[1] M. Clark. Fab: content-based, collaborative recommendation. pages 66–72. ACM New York, NY, USA, March 1997.

[2] M. Herlihy. A framework for collaborative, content-based and demographic filtering. *ACM Trans. Program. Lang. Syst.*, 13:393–408, November 1999.

[3] Q. Li and B. M. Kim. Clustering approach for hybrid recommender system. pages 33–38. Web Intelligence, 2003. WI 2003. Proceedings. IEEE/WIC International Conference, October 2003.

[4] Y. J. Lim and Y. W. Teh. Variational bayesian approach to movie rating prediction. volume 7. Citeseer, Proceedings of KDD Cup and Workshop, 2007.

[5] T. Miranda, M. Claypool, A. Gokhale, P. Murnikov, D. Netes, and M. Sartin. *Combining content-based and collaborative filters in an online newspaper*, volume 60. Proceedings of ACM SIGIR workshop on recommender systems, June 1999.