Predicting User Preference for Movies using NetFlix database

Dhiraj Goel and Dhruv Batra
Department of Electrical and Computer Engineering
Carnegie Mellon University
Pittsburgh, PA 15213
{dgoel,dbatra}@ece.cmu.edu

Abstract

Online content and service providers deal with the problem of providing *"relevant"* content on a regular basis, especially due to the sheer volume of data available. This work deals with one such problem, namely, that of predicting user preference for movies using the NetFlix database. We present a memory-based Collaborative Filtering (CF) algorithm that learns the personality traits of the users in a features space we call the Latent Genre Space (LGS). This representation allows us to use traditional clustering algorithms in this space, and overcome one of the biggest problems in these works – that of different lengths of user feature vectors in the vote space. Inference techniques in this space are discussed, and a kd-tree based nearest-neighbor scheme is implemented.

1 Introduction

With the advent of innumerable web-based content and service providers, the problem of providing *"relevant"* content becomes crucial. It is often necessary to make suggestions without sufficient knowledge of past behavior of particular users. An automated recommendation system can help in this context by aggregating the opinions of a community of users to help individuals in that community more effectively identify content of interest from a potentially overwhelming set of choices [1]. The developers of the first recommendation system Tapestry [2], coined the term "Collaborative Filtering (CF)", variants of which have become famous in this community. The basic assumption of collaborative filtering is that if users A and B have rated similarly in the past, then they will rate similarly in future. Based on the instantiations of 'similarity', 'past', and 'rating', the basic algorithm can take varied forms.

A popular classification of CF algorithms was proposed by Breese *et al* [3] – into Memory-based and Model-based methods. Memory-based methods work on the principal of aggregating the labeled data and attempt to match recommenders to those seeking recommendations. Most common memory-based methods works are based on the notion of nearest neighbor, using a variety of distance metrics. Model-based methods, on the other hand, try to learn a compact model from the training data, for example learn parameters of a parametric posterior distribution. From an operational point of view, memory-based methods potentially work with the entire training set and scale linearly with the amount of training

	The Godfather	Memento	Titanic	Star Wars
John	5	1	2	5
Dave	4		2	2
Susane	5	1	2	?
Mark	2	2		4

Table 1: Table showing the ratings that different users gave for the movies. Our recommendation scheme tries to predict the missing values

data, while model-based methods are constant time.

Pennock and Horvitz [4] suggest Personality Diagnosis (PD), a latent variable approach based on computing the probability that a new user is of an underlying "personality type". They then go on to find "similar" users in the personality type sense and thus estimate the probability that a new user will like an item. In a different paper, Pennock and Horvitz [5] note that "aggregation" of preferences has been studied in the Social Choice theory since 1960s [6], and argue that only a single nearest neighbor will satisfy the axiomatic foundations of CF.

Our proposed method lies at the intersection of memory-based and user personality approaches. The rest of this paper is organized as follows: Section 2 discusses the problem at hand, Section 3 explains in detail our proposed approach, Section 4 describes our experimental results, and Section 5 concludes with discussion.

2 Problem Space

The problem that any movie recommendation algorithm faces is to predict how well a user would like a movie he hasn't seen, given the response of the community for that movie. This can be thought of as predicting the missing values in the user-movie matrix. Consider an example shown in the Table 1. Each row of the matrix contains all the reviews by a single user while each column denotes responses/ratings for that movie from these users. The rating are on a scale of 1-5, with higher numbers denoting higher preferance.

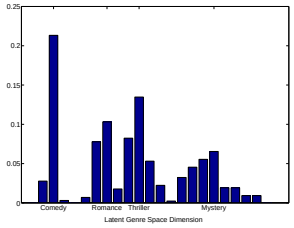
The matrix also contains missing values as every user hasn't seen every movie. Now, say we want to predict how much Susane would like the movie Star Wars. Given the preferences of other users, a reasonable guess would be "Like a Lot" (rating: 4) since John shares a similar taste for movies as Susane and hence, their future ratings should be similar. This kind of prediction is feasible in a small space of movies and users (like the given Table 1) but becomes hard when their numbers grow. The matrix tends to become increasingly sparse and it's not trivial to indentify users with similar taste.

The approach that we propose tries to circumvent the problem of sparsity of the user-movie matrix by projecting the preferences of the users into a fixed dimensionality space, thereby, learning the personality traits for each inidividual user.

3 Proposed Approach

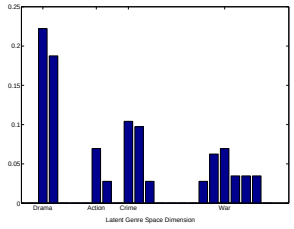
We propose a memory-based collaborative filtering approach for recommending new movies to users. In addition, our approach also learns the personality traits of the users in a new space, which we call the Latent Genre Space (LGS). The method can be broken into three main steps:

- **Learning Personalities:** Learning the preferences of the individual users based on the genres of the movies they have reviewed, i.e. projecting onto the Latent Genre Space.
- **Clustering Personalities:** Clustering the users in the Latent Genre Space so that the users in the same cluster exhibit similar tastes.
- **Predicting Responses:** Predicting the response of a user for an unseen movie depending on the responses of other users sharing the same cluster.



Movie	Genre	Rating
High Fidelity	Comedy,Drama,Romance	5
Chasing Amy	Drama,Romance	3
Bringing Out the Dead	Drama,Thriller	1

Figure 1: User1: User preferences as a vector in Latent Genre Space and the movie ratings



Movie	Genre	Rating
Braveheart	Drama,History,Action,War	5
Walk on Water	Drama,Thriller	5
The General's Daughter	Drama,Mystery,Thriller	4

Figure 2: User2: User preferences as a vector in Latent Genre Space and the movie ratings

The main idea behind learning user personalities is that the movie ratings given by the users tend to reflect their personality traits, specifically, their liking for certain themes. In general, it's reasonable to assume that each user would show a penchant for certian themes while exhibiting mild dislike, if not disaffection, for others. The caveat with relating movie responses with the genre alone is that it might lead to incorrect inference about the user taste. Strong inclination towards certain directors, actors etc. also influence how users rate movies. However, capturing additional information brings increased complexity and hence, is not considered for this project.

Assuming genre is sufficient to capture how users rate movies, we project the user responses in a new, low-dimensional space, called the Latent Genre Space. Each dimension in this space is represented by a genre. In all, 23 genres are considered (from IMDB database) - Action, Romance, Family, Crime, Thriller, Adventure, Horror, Western, Musical/Music, Fantasy, Sci-Fi, Mystery, War, Biography, Sport, History, Film-Noir. Each user is represented as a vector in this new space. For every movie that this user has rated, the corresponding genre is weighted by its rating. In case of multiple genres, all of them are assigned the same weightage as the rating, and are thus treated equally. Finally, the LGS vector is L_1 normalized. This step can be seen as a dimensionality reduction process, from the initial high-dimensional 'vote space' where the number of dimensions was equal to the number of movies in the database, and we had to deal with the problem of missing data, to this new space that is low-dimensional, and easier to work with. Most importantly, it overcomes the problem of clustering vectors of different lengths.

In our midterm report we had proposed that the next step would be to cluster users in this (LGS) space using either a naive algorithm like K-means or something sophisticated

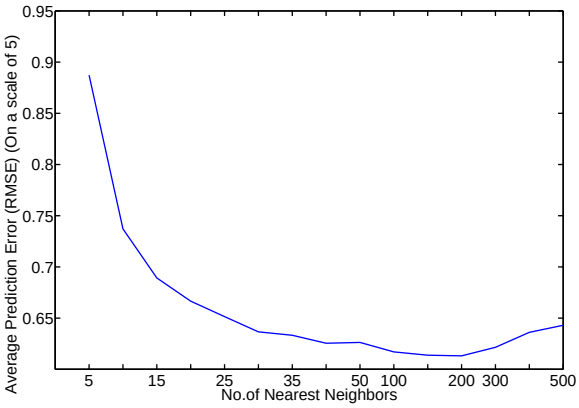


Figure 3: Validation Results for choosing the optimal value of K in KNN

like spectral clustering. The purpose of this clustering step would have been to discover cluster/cliques of "similar" users and the blanks in the user-movie matrix would be filled by averaging the reviews within a cluster. However, intuitively, we want to avoid "hard-binning" the data at an early stage, and want to consider *all* users within an 23-dimensional hypersphere centered at the user in question. Of course, a computational cost has to be paid, because in the former case, clustering can be performed once, while for the latter, a k-nearest neighbor query has to be made for every user in question. Kd-trees [7] are data structures that store the data in such a way so as to to facilitate fast k-nearest neighbor queries. A kd-tree for our problem, (like clustering) can be built once, and then used repeatedly for all queries. The kd-tree code used in our implementation is publically available at [8].

Thus, the predicted review for an unseen movie for a user would just be the weighted average of the responses of the k-nearest neighbors in the Latent Genre Space, where the weights are inversely proportional to their distances from the user in question, and 'k' (as we discuss in the next section) is picked by cross-validation.

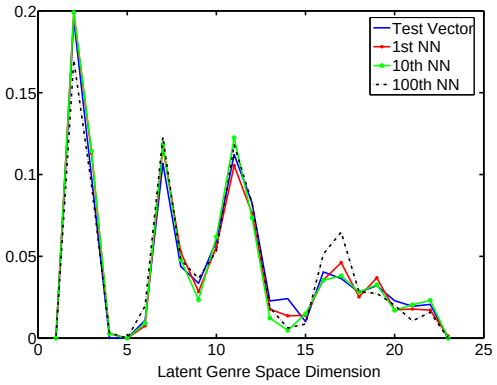
4 Experiments and Results

Our work/experiments can be categorized as:

- **Generating the Database and the Latent Genre Space:** The original training dataset from NetFlix (available for download from www.netflix.com) contains votes for 17,770 movies from approximately 480 million users (not all users vote for all movies). We obtained a list of genres from IMDB for approximately 5 million movies, 23 in total. A simple/naive search for genre labels was performed for all NetFlix movies. Due to irregularities in the movie names and the released dates between the databases, we could achieve exact match for only 6658 of the NetFlix movies. A list of 33941 users who have rated these movies was obtained. Out of these, 3941 users were set aside for the test purpose while the remaining 30000 users were used for finding the optimal value of K parameter in the KNN algorithm using the 5-fold cross validation. Each user is represented in the Latent Genre Space by a feature vector by accumulating the contribution of different genres of the movies rated by him. For the users in the test set, one movie per user is used for predicting the label while the remaining movies are used to generate the "partial" feature vector. The same approach is used for the validation set during

training set.

Fig 1 and Fig 2 show the feature vector generated for two users using their movie preferences. Higher weightage for genres like Comedy, Romance and Thriller for User 1 displays the liking for movies with these particular themes while User 2 has preference for Drama, Crime and War movies.



$k^{th}-$NN	Rating
1	4
5	4
10	5
50	3
100	3

Movie	Genre	Predicted Rating (k-NN)	Predicted Rating (Mean)	Actual Rating
Black Hawk Down	Action,Drama,History,War	3.72	3.6	4

Figure 4: Predicting the rating of an unrated movie for a new user. The top left figure is the feature vector in the Latent Genre Space genereated using the movies rated by this user till now. The top right figures shows the "closest" users obtained using KNN algorithm. The middle figure lists the rating for this movie by these matched users and the bottom table gives the predicted rating.

- **5-Fold Cross Validation to determine K:** Our cross-validation experiments consisted of using four of the five cut-out folds to prepare the kd-tree and use the fifth as a validation set to test the accuracy of our predictions. The error metric used in this work, and that promoted by the NetFlix Prize organizers [9] is Root Mean Square Error (RMSE). As we can see from Figure 3, the average prediction error across these folds decreases as we consider more neighbors, but starts increasing after a point. Intuitively, this makes sense, because at one extreme, we could consider *all* users (with K = ∞), and our prediction would just converge to the average vote for that movie, thereby becoming a reflection of the average popularity of that movie and *not* an indication of this particular user's preference (which was what we set out to do). At the other extreme, we could consider just one nearest neighbor and our prediction for this user would just become his "buddy's" vote.

At this end, our estimates would have a very high variance, and thus we need a optimal point, which in this case turns out be 200.

- **Testset Accuracy:** Figure 4 shows a sample query to our system – top plot shows the feature vectors in LGS of the query user and the retrieved nearest neighbors, the tables show the true review given by this user and that predicted by our system using various values of K. For our final evaluation, we considered two simple baselines for comparison: the first one being the mean of all votes for a movies (which reflects the average user-non-specific popularity of a movie), and a naive algorithm that always guesses 3 as a review. Our algorithm outperforms these two baselines, though a detailed discussion of why it does not beat them with a wider margin follows in the next section. Figure 5 shows the comparison of our testset error for all three methods.

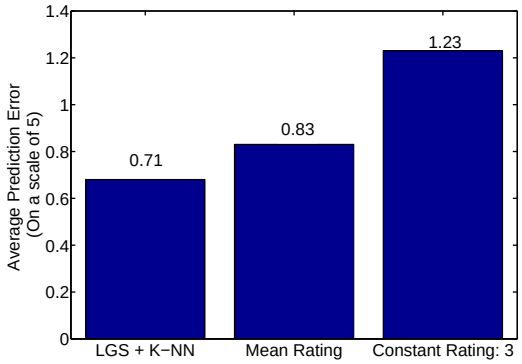


Figure 5: Error plots showing the comparison of our method with the two baseline

5 Conclusions and Discussions

We presented a novel memory-based Collaborative Filtering algorithm to predict user reviews for movies using the NetFlix dataset. In order to overcome the problem of different length movie-vote feature vectors, and to reduce computational complexity by working with a subset of "relevent" users, we introduced a feature space called the Latent Genre Space. Using a kd-tree to make efficient k-nearest neighbors queries in this space, we estimated the unseen review as a weighted combination of the reviews of these neighbors. Our results are promising and outperform the two baselines we compare to. Since all our results were obtained on a subset of the entire NetFlix database available for the NetFlix prize, we have not submitted our results on the website. Even though our algorithm performs better than the two baselines we compare to, we hypothesize two explanations for the fact that it does not outperform with a wider margin: the first is that the genre of the movie might not contain enough information for our task. After all, our model is completely oblivious to actor/director preference of users. Also, the mean across all votes is infact, we would argue, a very non-naive baseline. First order statistics contain a lot of information, and the user vote distributions are rarely, if ever, polarized; popular movies are usually globally popular.

References

[1] P. Resnick and H. R. Varian, *Special issue on recommender systems,* Communications of the ACM, 40(3), 1997.

[2] D. Goldberg, D. Nichols, B. Oki, and D. Terry, *Using collaborative ltering to weave an information tapestry*, Communications of the ACM, 35(12):61-70, 1992.

[3] J. Breese, D. Heckerman, and C. Kadie, *Empirical analysis of predictive algorithms for collaborative filtering*, In The Fourteenth Annual Conference on Uncertainty in Artificial Intelligence, pages 43–52, 1998.

[4] D. M. Pennock and E. Horvitz, *Collaborative Filtering by Personality Diagnosis: A Hybrid Memory- and Model-Based Approach*, In IJCAI Workshop on Machine Learning for Information Filtering, Stockholm, Sweden, August 1999. International Joint Conference on Artificial Intelligence.

[5] D. M. Pennock and E. Horvitz, *Analysis of the axiomatic foundations of collaborative filtering*, In AAAI Workshop on Artificial Intelligence for Electronic Commerce, Orlando, Florida, July 1999. National Conference on Artificial Intelligence.

[6] K. Arrow, *Social choice and individual values*, New Haven: Cowles Foundation, 2nd edition, 1963.

[7] J. L. Bentley, *Multidimensional binary search trees used for associative searching*, Commun. ACM 18, 9, 509-517, 1975.

[8] http://www.robots.ox.ac.uk/ abm/docs/vgg_kdtree2.tar.gz

[9] http://www.netflixprize.com/

Appendix A: Distribution of Work

Dhruv	Dhiraj
Obtaining Genre	Creating user feature vector
Cross-validation	Prediction

Table 2: Table showing the distribution of work