Adaptive Recommendation: Putting the Best Foot Forward

John O'Donovan and John Dunnion

Department of Computer Science,
University College Dublin,
Dublin, Ireland.
{John.ODonovan,John.Dunnion}@ucd.ie

Abstract. Information is becoming more increasingly available in digital formats such as Web Pages, MP3 files and many others. This puts more emphasis on the need for reliable information filtering techniques. New recommendation algorithms are being developed frequently to deal with the problem of information overload. In this paper we present a new, regression-based approach to the application of recommendation algorithms. We classify five different datasets based on a range of metrics, including sparsity, user-item ratio, and the distribution of user ratings. From performance analysis tests of four predictive algorithms over these sets, we develop a regression function to predict the suitability of a particular recommendation algorithm for a previously unseen dataset. Our results show that the best-performing algorithm on the new set is the same as predicted by our regression analysis.

1 Introduction

Information filtering techniques are becoming more widely used as available information spaces grow ever larger. New techniques for filtering information are being developed to tackle the information overload problem. We present an assessment of the performance of three popular recommendation algorithms over a range of diverse data. Our aim is to show that the relative performance of these algorithms varies as they are applied to different data, and that these differences can be harnessed to develop an Adaptive Recommender System. If we can successfully perform this algorithm prediction task, we can form the basis of a generic recommender system, which can employ cutting-edge filtering techniques to a given system without having to manually tailor the recommendation engine for that system.

We classify our datasets based on a set of their salient features, including user-item ratio, rating distribution, data type, and sparsity of the data. We run performance tests on Item-Based Collaborative Filtering (IBCF) [9] [16] [4], Pure Collaborative Filtering(UBCF) [15] [**?**][8], and an Association-Rule Based Filtering algorithm (RBCF) [13] [2] and a base-line classification algorithm: Zero-r [1], using four experimental datasets: PTV [3], Movielens [15], Jester [5] and EachMovie.

Each of our recommendation algorithms differs in the manner in which it gathers similarity information about users. IBCF compiles a model of item similarity, based on the number of users who have co-rated each pair of items. UBCF is based on a model of user similarity, which is acquired by forming peergroups of users with high overlap of items in their profiles. The RBCF algorithm is similar to IBCF, except it employs data mining techniques (the Apriori association rule algorithm) to form associations between items, which are then used in an IBCF-like model of item-item similarity. These rules take the form: $A \Longrightarrow B$, with associated support and confidence levels.

Based on the results of our performance analysis, and the values each dataset produces for the classification metrics, we develop a multi-variable linear regression function for the purpose of predicting the suitability of a particular recommendation algorithm to a dataset

previously unseen by the system, (The SmartRadio music ratings dataset). We plot a response-surface graph of the predictive accuracy of each recommendation algorithm with respect to the classification metrics for our different datasets.

Using this graph and the new datasets classification values, we aim to, and succeed in predicting the best performing algorithm for the new dataset. Our empirical results show that there is a significant difference in the relative performance of these algorithms over our four data platforms, and that the resulting regression function not only correctly predicted the best-performing algorithm for the new dataset, but also the second best performer.

2 The AdRec System

AdRec is a framework wherein recommendation techniques can be dynamically assigned to datasets based on a lightweight classification of that set. By adopting this approach to the recommendation task, we hope to minimise the effects of inherent CF problems such as sparsity and latency [13].

Figure 1 below, outlines the architecture of this adaptive system. Data is sent to a classification module, which records the salient features of the dataset. Each of the recommendation techniques in the system then generates its own recommendations on the data. The recommendations from each technique are passed to the analysis module which performs accuracy testing on the recommendations, using train-test sets from the data. The performance of each algorithm, and the associated dataset classification is stored in an SQL database. A regression component then interpolates this data and outputs a regression function which can predict algorithm performance on a new dataset based only on its classification metrics.

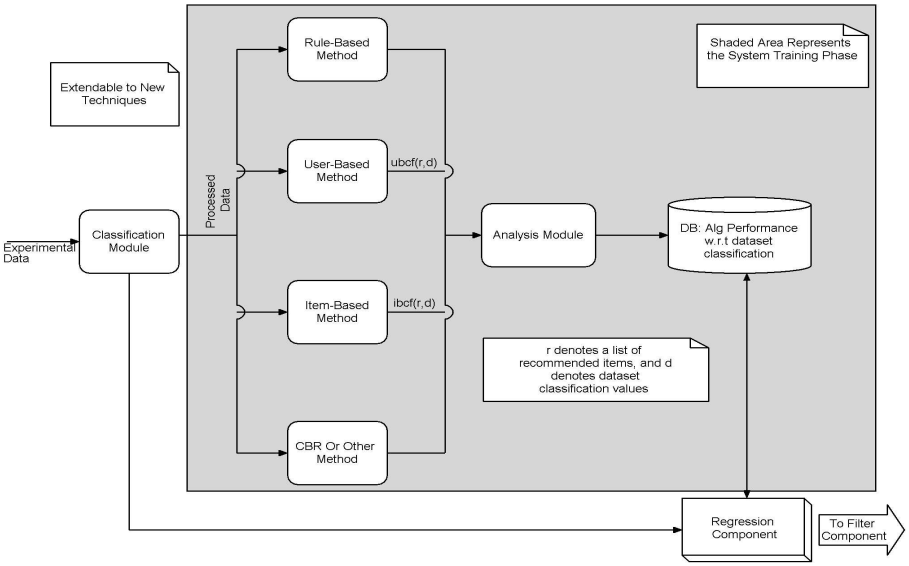

Fig. 1. AdRec System Architecture.

2.1 Regression

A linear regression model [16] is built up using our evaluations in [11]. This is based on a predictive function of several variables, as described in [14]:

$$E\{Y\} = \beta_0 + \beta_1 X_1 + \beta_2 X_2$$

Values for each β_i are obtained by taking the classification metric values for each dataset, together with the best performing algorithm for each set, and solving the resulting system of simultaneous equations.

2.2 Java Implementation

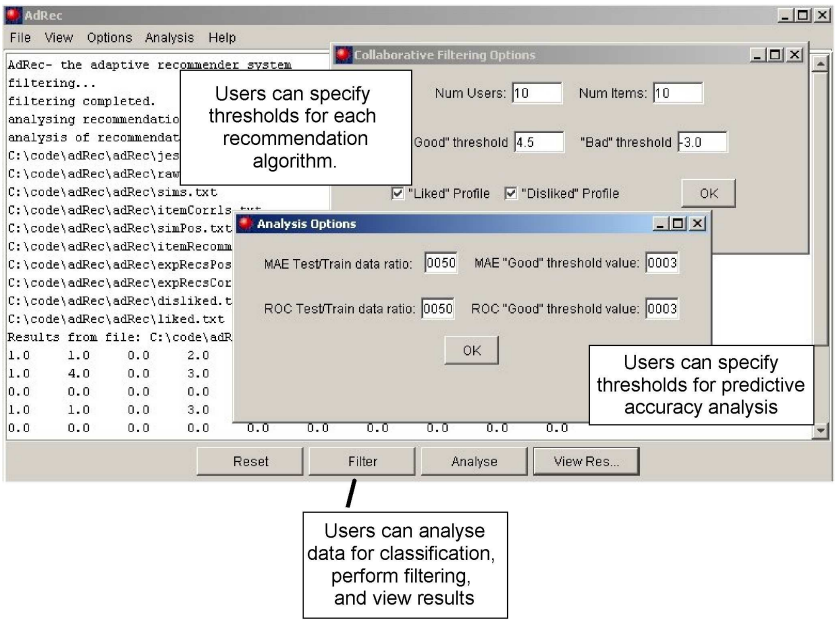

Fig. 2. The AdRec GUI

The AdRec system has three interfaces, a simple command-line interface in which parameters are passed directly to the system via the javac command; a GUI implemented in java swing technology, designed to give a better visual impression of the system; a web-based front-end, in which users can receive recommendations from each of the algorithms in the system. These recommendations are based on a short profile of items which the user has selected from one of the available datasets. Users also have an option to rebuild the similarity model upon which their recommendations are based, and specify its training set size. This feature of the system uses Java Servlet technology over an Apache Tomcat web-server.

Unlike Herlocker's CFEngine [7], which implements a full CF system based on a Java client-server model using Java RMI and CORBA technologies, the AdRec system is implemented using a simple three tier model, using an SQL back end, Java and Perl for middleware, and the three Java-based interfaces mentioned above.

2.3 Similarity Capture in CF Systems

Similarity capture is a vital feature of any CF algorithm, and the manner in which this is performed can have significant effects on system performance. Similarity can be computed for CF by several well-known techniques, such as Cosine Similarity, Spearman's Correlation and Pearson's Correlation [12]. For our similarity calculations we employ Pearsons Correlation, as it is the most widely used and allows for better comparison with other systems. It is defined as follows:

$$corr_{x,y} = \frac{\sum_{u \in U}(R_{u,x} - \overline{R}_x)(R_{u,y} - \overline{R}_y)}{\sqrt{\sum_{u \in U}(R_{u,x} - \overline{R}_x)^2 \cdot \sum_{u \in U}(R_{u,y} - \overline{R}_y)^2}}$$

where $corr_{x,y}$ is the Pearson correlation coefficient between users x and y, $R_{i,j}$ is the rating of item i by user j, and $\overline{R}_i$ is the average item rating by user i.

2.4 Experimental Data and Dataset Classification

For the initial training phase of the system, four experimental datasets were used: Jester [5] (an experimental dataset of jokes ratings, consisting of 21800 users ratings of 100 jokes); EachMovie (73000 user ratings of 1628 movies); PTV [3] (622 user ratings of TV programmes); and MovieLens [15] (100000 user ratings in the movie domain). In future work, it is hoped to also include a customer-product purchase database from an on-line sales company. For the purposes of our testing we selected subsets of 900 profiles from each of the above datasets comprised of the largest profiles, ie those users who had rated 20 items or more (with the exception of PTV, which only contains 622 profiles, and SmartRadio, which has 395). The datasets were parsed, converted into the same format and stored in an SQL database. The SmartRadio dataset was classified according to the same metrics as the others and was found to be over 99% sparse and to have a user-item ratio of 1:9 ($\beta_1 = 0.99$ and $\beta_2 = 0.111$). A summary of the dataset classification is presented in Table 2.4.

Dataset	*User:Item*	*Sparsity (%)*	*Type*
PTV	1:6	94.25	TV Programmes
MovieLens	9:13	63.86	Movie Ratings
Jester	9:1	54	Jokes Ratings
EachMovie	9:17	33.97	Movie Ratings
SmartRadio	1:9 (approx)	99.98	Music Ratings

Table 1. Classification of Experimental Datasets.

3 Experimental Evaluation

We designed a suite of tests that would empirically show that it is possible for the system to predict the best-performing algorithm using only the regression function learned from the classification metrics of the other datasets, and the values the new dataset has for these metrics. Due to space restrictions, all of the experimentation and tests used for construction of the regression function cannot be detailed here. See [11] for detail on this. Figure 3 shows a response surface graph of the regression function which we used for the predictions for the IBCF algorithm.

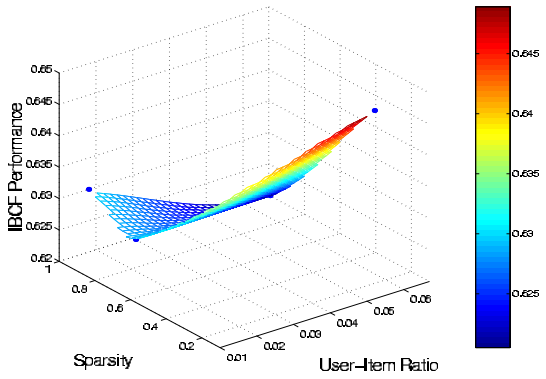

Fig. 3. Response surface graph for the IBCF algorithm

Figure 3 shows the predicted performance levels for the IBCF algorithm based on two seemingly independent variables: user-item ratio and sparsity. This graph shows that the predicted

performance increases with density of ratings, as one would expect. The surface is skewed however, by the ratio of users to items in the dataset. For further work, we would like to introduce a weighting scheme for these variables, as the sparsity measure is has more of an effect on the overall performance of the algorithm.

Having calculated our regression function from algorithm performance and dataset classification metrics over four training datasets, we introduce the SmartRadio dataset [6]. We run each of the prediction algorithms individually on this dataset, previously unseen by the system, and evaluate the results using predictive accuracy (mean absolute error and ROC Sensitivity [10]). However, here we present only a basic predictive accuracy test. These tests are simplified by keeping the neighbourhood size k and the test-train ratio constant at 30 and 80, respectively. These are optimal values we discovered empirically and reported in [11].

3.1 Experimental Results

To re-iterate: the goal of the AdRec system is to predict the best performing algorithm for a dataset, based solely on our regression model and on the classification metrics for the new dataset.

The graph in Figure 4 shows that the user-based CF algorithm has a better predictive accuracy than its competitors. This algorithm was correctly predicted as the best performer by our regression function. We can see that the performance of all of the algorithms is significantly more poor on the SmartRadio set than the others. This is due to the high sparsity level in the dataset, as we can see from table 2.4

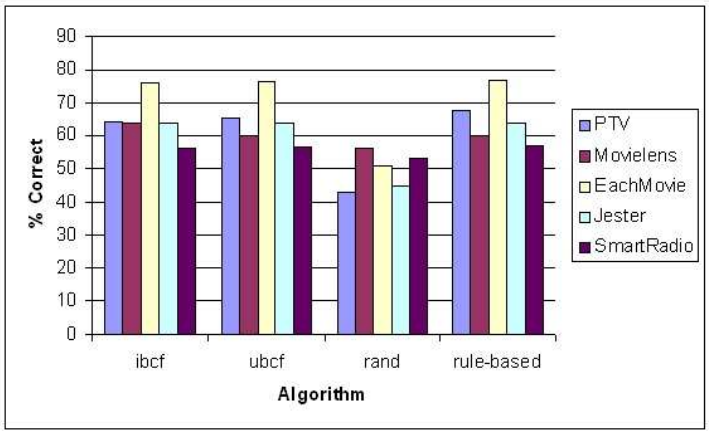

Fig. 4. Recommendation Accuracy for each algorithm, keeping k constant at 30 and the test-train ratio at 80%.

4 Conclusions and Future Work

This paper evaluated three recommendation algorithms within the framework of AdRec, a system which compiles a predictor function for algorithm performance based on classification of, and algorithm performance analysis on four training datasets. We then evaluated the performance of the prediction function on a new dataset. Our performance results show that the system was able to predict both the first and second best performing algorithms on the new dataset. Our testing procedure will be modified to include a ten-fold cross validation as in [17], decision support metrics, such as Receiver Operator Characteristic (ROC) as in [10], and better statistical accuracy in the form of mean absolute predictive error. One of the main drawbacks of

this area is the lack of available experimental datasets. We hope to introduce a new jokes dataset upon which to further test the system, and incorporate new (customer-product)recommendation domains in our tests.

Possible deployments of such a system include applications which require a recommendation engine, but a there is not enough knowledge of the area to know which is the best strategy to employ. This system can essentially personalise a recommendation strategy to suit the requirements of a particular application.

Acknowledgements

The support of the Enterprise Ireland Informatics Research Initiative is gratefully acknowledged. The research was funded under grant PRP/00/INF/06. We would also like to thank all those who participated in making our test datasets available.

References

1. Syed S. Ali and Susan McRoy. Links: Java resource for artificial intelligence. *intelligence*, 11(2):15–16, 2000.
2. Christian Borgelt and Rudolf Kruse. Induction of association rules: Apriori implementation. In *Proc. 15th Conf. on Computational Statistics (Compstat 2002, Berlin, Germany)*, Heidelberg, Germany, 2002. Physika Verlag.
3. Paul Cotter and Barry Smyth. PTV: Intelligent personalised TV guides. In *Proceedings of the 7th Conference on Artificial Intelligence (AAAI-00) and of the 12th Conference on Innovative Applications of Artificial Intelligence (IAAI-00)*, pages 957–964, Menlo Park, CA, July 30– 3 2000. AAAI Press.
4. D. Fisk. An application of social filtering to movie recommendation. *Lecture Notes in Computer Science*, 1198:116–131, 1997.
5. Ken Goldberg, Theresa Roeder, Dhruv Gupta, and Chris Perkins. Eigentaste: A constant time collaborative filtering algorithm. *Information Retrieval*, 4(2):133–151, 2001.
6. C. Hayes, P. Cunningham, P. Clerkin, and M. Grimaldi. Programme-driven music radio, 2002.
7. Jonathan L. Herlocker and Joseph A. Konstan ans John Riedl. An empirical analysis of design choices in neighborhood-based collaborative filtering algorithms. In *Journal of Information Retrieval.*, pages Inf. Retr. 5(4), p. 287–310, 2002.
8. Jonathan L. Herlocker, Joseph A. Konstan, and John Riedl. Explaining collaborative filtering recommendations. In *Proceedings of ACM CSCW'00 Conference on Computer-Supported Cooperative Work*, Expertise and Explanation, pages 241–250, 2000.
9. George Karypis. Evaluation of item-based top-n recommendation algorithms. In *CIKM*, pages 247–254, 2001.
10. P. Melville, R. Mooney, and R. Nagarajan. Content-boosted collaborative filtering, 2001.
11. John O'Donovan and John Dunnion. A comparison of collaborative recommendation algorithms over diverse data. In *Proceedings of the National Conference on Artificial Intelligence and Cognitive Science (AICS), Ireland*, pages 101–104, September 17– September 19 2003.
12. Michael P. O'Mahony, Neil Hurley, and Guenole C. M. Silvestre. An attack on collaborative filtering. In *Proceedings of the 13th International Conference on Database and Expert Systems Applications*, pages 494–503. Springer-Verlag, 2002.
13. Derry O'Sullivan, David C. Wilson, and Barry Smyth. Improving case-based recommendation: A collaborative filtering approach. In *Proceedings of the Sixth European Conference on Case Based Reasoning.*, pages LNAI 2416, p. 278 ff., 2002.
14. Adrian E. Raftery, David Madigan, and Jennifer A. Hoeting. Bayesian model averaging for linear regression models. *Journal of the American Statistical Association*, 92(437):179–191, 1997.
15. Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. Grouplens: An open architecture for collaborative filtering of netnews. In *Proceedings of ACM CSCW'94 Conference on Computer-Supported Cooperative Work*, Sharing Information and Creating Meaning, pages 175–186, 1994.
16. Badrul M. Sarwar, George Karypis, Joseph A. Konstan, and John Reidl. Item-based collaborative filtering recommendation algorithms. In *World Wide Web*, pages 285–295, 2001.
17. Badrul M. Sarwar, Joseph A. Konstan, Al Borchers, Jon Herlocker, Brad Miller, and John Riedl. Using filtering agents to improve prediction quality in the grouplens research collaborative filtering system. In *Proceedings of ACM CSCW'98 Conference on Computer-Supported Cooperative Work*, Social Filtering, Social Influences, pages 345–354, 1998.