Collaborative Recommendation via Adaptive Association Rule Mining

Weiyang Lin   Sergio A. Alvarez *   Carolina Ruiz

Department of Computer Science
Worcester Polytechnic Institute
Worcester, MA 01609 USA
{wy_lin,alvarez,ruiz}@cs.wpi.edu
http://www.cs.wpi.edu/~{wy_lin,alvarez,ruiz}

Abstract

Collaborative recommender systems allow personalization for e-commerce by exploiting similarities and dissimilarities among users' preferences. We investigate the use of association rule mining as an underlying technology for collaborative recommender systems. Association rules have been used with success in other domains. However, most currently existing association rule mining algorithms were designed with market basket analysis in mind. Such algorithms are inefficient for collaborative recommendation because they mine many rules that are not relevant to a given user. Also, it is necessary to specify the minimum support of the mined rules in advance, often leading to either too many or too few rules; this negatively impacts the performance of the overall system. We describe a collaborative recommendation technique based on a new algorithm specifically designed to mine association rules for this purpose. Our algorithm does not require the minimum support to be specified in advance. Rather, a target range is given for the number of rules, and the algorithm adjusts the minimum support for each user in order to obtain a ruleset whose size is in the desired range. Rules are mined for a specific target user, reducing the time required for the mining process. We employ associations between users as well as associations between items in making recommendations. Experimental evaluation of a system based on our algorithm reveals performance that is significantly better than that of traditional correlation-based approaches.

*Corresponding author. Present affiliation: Department of Computer Science, Wellesley College, 106 Central Street, Wellesley, MA 02481 USA, e-mail: salvarez@wellesley.edu

1 Introduction

Collaborative recommender systems offer personalized recommendations of articles to users based on information about similarities among users' tastes (see e.g. [16], [15], [3]). Most available systems rely on simple methods to represent similarities among users, e.g. the linear correlation coefficient for a given pair of users. Concurrently, the field of data mining has given rise to effective methods for identifying patterns in large datasets. We are particularly interested here in the data mining paradigm of association rules [1]. Conditions are ripe for the application of these new methods to recommender systems.

1.1 Contributions of this paper

We have designed and implemented a method for collaborative recommendation based on a new association rule mining algorithm [12]. Our approach has the following advantages.

1. The association rules framework provides two useful measures for evaluating the association expressed by a rule. The *confidence* of a rule measures the degree of the correlation among users or among articles, while the *support* of a rule measures the significance of the correlation.

2. We can use overlaps of several users' tastes to match a given user's taste. This allows making good recommendations to users whose tastes don't correlate strongly with those of other individual users.

3. Our method can be configured to employ either associations among users, associations among articles, or a combination of the two.

Our mining algorithm automatically selects the minimum support so that an appropriate number of rules is produced for each target user or article. We include experimental results that show that the resulting recommendation performance is significantly better than that obtained using traditional recommender system technologies.

1.2 Relation to other work

We briefly describe some widely used techniques for collaborative recommendation below.

1.2.1 Linear correlation-based method

[16] and [15] put forth several variants of a technique based on the statistical (Pearson) correlation between two users' ratings. This technique has since become widely used. A disadvantage of the correlation-based techniques lies in the simple linear formulas that they use to make predictions. Users whose preferences are very clearly related, except not via a linear relation, may not be correlated at all and thus the predictive information present in their behavior patterns will be ignored altogether by correlation-based methods. Other drawbacks of the correlation approach are mentioned in [4]: the significance of the correlations between users is not measured, and, most importantly, if two users do not rate articles in common, they can not be similar under the correlation method even if they share common interests. Our approach can possibly overcome these drawbacks.

1.2.2 Neural Networks Paired with Feature Reduction Techniques

Billsus and Pazzani [4] present a framework for applying machine learning algorithms paired with feature reduction techniques, such as singular value decomposition (SVD) or information gain, for collaborative recommendation. They use feature reduction techniques to reduce the dimension of the rating data, and then neural networks are applied to the simplified data to construct a model for recommendation. In section 4 we compare our approach with this approach as well as the correlation-based method under similar experimental conditions.

1.2.3 Association Rules

Association rules have previously been used in web mining. They have been used to mine path traversal patterns and to facilitate the best design and organization of web pages [7, 8, 6]. For the domain of recommender systems, Fu, Budzik and Hammond[9] have recently developed a system for recommending web pages using the Apriori algorithm to mine association rules over users' navigation histories.

However, previously proposed association rule mining algorithms, Apriori included, are not well-suited for collaborative recommender systems. Two significant reasons for this are:

- Previous algorithms do not provide a mechanism to choose a proper minimum support for the given minimum confidence and the desired range for the number of rules. This often leads to either too many or too few rules, and thus to either excessive computation time or else poor recommendation performance.
- Most existing algorithms do not allow the heads of the rules to be specified in advance. Although CBA-RG [13] has addressed the problem of mining rules for a single target class, the recommendation problem is even more focussed since we need to mine rules for only one target class value.

2 Association Rules

The framework of association rules was introduced into the data mining community at large by Agrawal et al. [1]. Much earlier, P. Hájek et al. [10, 11] had anticipated many of the same concepts and approaches, but had focused on the representational power of association rules rather than the algorithmic aspects of rule mining. A large variety of association rule mining algorithms have been published in the literature, including Apriori [2] and DIS [5]. One extension of the basic binary association rules, called categorical association rules, [17] finds associations between attributes with categorical values. Some results of adapting those rules to classification tasks are shown in [13, 11]. [13] presents the CBA-RG algorithm (which is based on the Apriori algorithm) and a good framework to perform the so-called associative classification. Our adaptive-support algorithm to mine association rules for recommender systems is adapted from the Apriori and CBA-RG algorithms.

2.1 Definitions

We now introduce some of the basic terminology of association rules. A *transaction* is a set of items. An *association rule* is a rule of the form X $\Rightarrow$ Y, where X and Y are sets of items; X and Y are called respectively the *body* and the *head* of the rule. The intended meaning of this rule is that the presence of (all of the items of) X in a transaction implies the presence of (all of the items of) Y in the same transaction with some probability. Each association rule has two measures relative to a given set of transactions: its *confidence* and its *support*. Confidence is the percentage of transactions that contain Y among transactions that contain X; support is the percentage of transactions that contain both X and Y among all transactions in the input data set. In other words, the confidence of a rule measures the degree of the correlation between itemsets, while the support of a rule measures the significance of the correlation between itemsets.

2.1.1 Traditional Association Rule Mining Problem Definition

The traditional association rule mining problem definition is: *given a set of transactions, a user-specified minimum support and minimum confidence, find all association rules that are above the user-specified minimum support and minimum confidence.* This problem definition is well suited to uncovering aggregate user behaviors as required in traditional market basket analysis. However, for reasons explained below, we believe that this problem definition should be modified if the goal is to mine association rules for collaborative recommender systems.

2.1.2 Our New Association Rule Mining Problem Definition

It is difficult to choose a proper minimum confidence and support for each user/article before the mining process because users' tastes and articles' popularities vary widely. If the minimum confidence and support for mining are set too high, we cannot obtain enough rules for accurate recommendation; If they are set too low, the runtime may be unacceptably long. Also, an excessive number of rules may lead to decreased performance. This suggests the following new goal for association rule mining for recommender systems: *Given a transaction dataset, a target item, a specified minimum confidence and a desired range [minNumRules,maxNumRules] for the number of rules, find a set S of association rules with the target item in the heads of the rules such that the total number of rules in S is in the given range, the rules in S satisfy the minimum confidence constraint, and the rules in S have higher support than all rules not in S that are of the given form and satisfy the minimum confidence constraint.*

2.2 Our New Association Rule Mining Algorithm

In this section we describe our algorithm to mine association rules. See [12] for further details. Our algorithm adjusts the minimum support of the rules during mining in order to obtain an appropriate number of significant rules for the target user or article. We henceforth use the term *target item* interchangeably for both "target user" and "target article".

Our mining algorithm consists of an outer loop and an inner loop. The overall process consists of three parts, as follows. First, the outer loop initializes the minimum support count (product of the minimum support and the total number of transactions) according to the frequency of the target item and calls the inner loop to mine rules. When the inner loop terminates, the outer loop checks if the number of rules returned exceeds $maxNumRules$. If it does, the outer loop increases the minimum support count and calls the inner loop until the number of rules is $maxNumRules$ or less. Finally, the outer loop checks if the number of rules is less than $minNumRules$; if it is, it decreases the minimum support count until the rule number is greater than or equal to $minNumRules$. For a given support, rules with shorter bodies are mined first. If the number n of rules is out of range for consecutive values of the minimum support count, with $n > maxNumRules$ at support count s and $n < minNumRules$ at support count $s + 1$, then the shortest $maxNumRules$ rules with the smaller support count are returned.

The inner loop of our algorithm is a variant of CBA-RG [13] and therefore of the Apriori algorithm [2] as well. It is a variant of CBA-RG in the sense that instead of mining rules for all target classes, it only mines rules for one target item. It differs from CBA-RG in that it will only mine a number of rules within a certain range. For further details, see [12]. A different rule mining algorithm that produces rulesets of constrained size is presented in [18].

3 Recommendation using our Mining Algorithm

We now describe how our algorithm to mine association rules may be used for collaborative recommendation. The same mining process may be used to mine a certain number of rules for each user/article for both user associations and article associations. The main differences between the implementation of these two association types are that we use different transaction data to mine the association rules, and different recommendation strategies.

3.1 Mapping Ratings to Transactions

The conversion from item ratings available for recommendation tasks to "transactions" as required for association rule mining is determined by what kind of associations and how many levels of associations we want to discover. We map the numeric ratings for an item into two categories: *like* and *dislike* according to whether the rating for the item is greater than or less than some chosen threshold value. Then we convert the chosen *like* and *dislike* ratings into transactions:

- In order to obtain *like* associations among *users*, we have each user correspond to an "item" and each article rated by users correspond to a "transaction". If a user likes an article, then the transaction corresponding to the article contains the item corresponding to the user liking the article; If the user dislikes or did not rate the article, then the corresponding transaction does not contain the corresponding item. The mined rules will then be of the following form: "90% of articles liked by user A and user B are also liked by user C, 30% of all articles are liked by all of them", or, in simpler notation,

  "$[user_a : like]$ AND $[user_b : like] \Rightarrow [user_c : like]$ with confidence 90% and support 30%".

- In order to mine *like* associations among *articles*, we have each article correspond to an "item" and each training user who rated the target item correspond to a "transaction". If a training user likes an article, then the transaction corresponding to the user contains the item corresponding to the article; If the user dislikes or didn't rate the article, then the corresponding transaction does not contain the corresponding item. From here, we can mine *like* associations among articles, such as:

  "$[article_1 : like]$ AND $[article_4 : like] \Rightarrow [target\_article : like]$" with confidence 100% and support 40%.

3.2 Recommendation Strategy

3.2.1 User associations

For user associations, the rules mined are akin to $[training\_user_1 : like]$ AND $[training\_user_2 : like] \Rightarrow [target\_user : like]$. If $training\_user_1$ likes a test article and $training\_user_2$ also likes this article, then we say this rule *fires* for this article. We associate each rule with a score, which is the product of the support and the confidence of the rule. We also assign a score to each article, which is the sum of the scores of all the rules that fire for that article. If the score for $article_i$ is greater than a threshold, then we recommend $article_i$ to the target user.

3.2.2 Article associations

For article associations, the rules we have are of the form: $[article_1 : like]$ AND $[article_2 : like] \Rightarrow [target\_article : like]$. For a test article of the target user, if the user likes $article_1$ and $article_2$ (which could be known from the training articles of the user), then we say this rule fires for this article.

Our recommendation strategy for article associations is different than for user associations. We only recommend articles whose rules' supports are above a cutoff. The support cutoff is adjusted during system tuning; the mining process is then restricted to rules whose support is above the cutoff. Our mining process has the following advantages over algorithms such as Apriori or CBA:

- By mining article associations for one article at a time, only ratings related to the target article are used for mining, which is only a small subset of the whole rating data. The support of a rule is calculated over the small subset of the whole rating data, which enables us to obtain rules for articles that have only received a limited number of ratings, for example a new movie.

- A significant amount of runtime is saved by mining rules only over the subset of the rating data that is related to the target article instead of over the whole data. Systems that mine rules with unrestricted heads such as IBM's Intelligent Miner can easily take several days to mine article associations for all articles at once.

3.2.3 Combined associations mode

We use the following strategy to combine user and article associations: If a user's minimum support is greater than a threshold, then we use user associations for recommendation, otherwise we use article associations for recommendation.

4 Experimental Evaluation

In this section, we describe experimental results obtained with a collaborative recommendation method based on our association rule mining techniques.

4.1 Training and Test Data

We use the EachMovie Dataset [14] as the test-bed of our approaches. This dataset contains ratings from 72,916 users for 1,628 movies. Ratings are given on a numeric six–point scale (0.0, 0.2, 0.4, 0.6, 0.8, 1.0).

4.2 Collaborative and Target Users

We performed two groups of experiments. In the first group, we chose the first 1000 users in the EachMovie dataset who rated more than 100 movies as collaborative users, and the first 100 users whose userIDs are greater than 70,000 and who rated more than 100 movies as target users. By choosing collaborative users who rated over 100 movies, enough movies are available for our cross-validation tests. In order to compare our approach with other approaches, we performed a second group of experiments, for which we chose the first 2000 users in the database as the collaborative user group, and 91 random users whose like ratios are less than 0.75 and who have rated 50 to 100 movies as target users. See section 4.6.

4.3 Experimental Protocol

System parameters that were adjusted during the experiments reported in this paper are the minimum confidence for rule mining, and the score threshold for recommendation. See [12] for results involving additional parameters used in the mining process. For the experiments described below, the maximum number of terms allowed per rule was 8 and the target range for the number of rules to be mined was $10-100$. We use a *like* threshold of 0.7, i.e., if a user's rating for an article is greater than 0.7, then we assume that the user likes the article. With this *like* threshold, the ratio of the number of movies liked to the total number of movies rated among all the test users is 0.45. Test results for each target user are computed using a *4-fold cross-validation* approach.

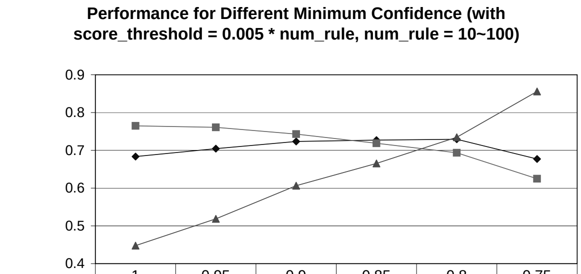
4.4 Performance Measurement

We use the *accuracy*, a commonly used performance measure in machine learning, and two standard information retrieval measures, *precision* and *recall*. Accuracy is the percentage of correctly classified articles among those classified by the system; Precision is the percentage of articles recommended to a user that the user likes; Recall is the percentage of articles liked by a user that are recommended to him/her. The best performance measure may depend on the domain. Our approach includes several parameters that may be tuned to optimize a particular measure.

4.5 Performance Results

4.5.1 Minimum Confidence

Recommendation performance results as a function of the minimum confidence are shown in Figure 1. The



	1	0.95	0.9	0.85	0.8	0.75
Accuracy	0.683552	0.704511	0.723359	0.727041	0.729348	0.676991
Precision	0.764803	0.76103	0.743149	0.718678	0.693465	0.625
Recall	0.447833	0.518673	0.606528	0.665383	0.734724	0.855856

Minimum Confidence

Figure 1: Performance vs. Minimum Confidence

minimum confidence has a significant impact on performance: the higher the minimum confidence, the higher the precision but the lower the recall. In contrast, the highest *accuracy* is obtained not with minimum confidence 100% but with minimum confidence from 80% to 90%. We use a minimum confidence of 90% for the remaining experiments.

4.5.2 Score Threshold

We use a score threshold for recommendation that is a linear function of the number of rules. Figure 2 gives the performance for different values of the slope of this function; the dependence of performance on the base value (not shown) is qualitatively similar.

Notice that the score threshold has a similar impact on the performance as the minimum confidence, i.e., the higher the score threshold, the higher the precision but the lower the recall. However, by simultaneously adjusting the minimum confidence we can increase both precision and recall: in Figure 2, we achieve a precision of 0.77 and a recall of 0.53 for a score threshold of $0.01 * rule\_num$ (and minimum confidence of 90%); c.f. the results for minimum confidence 100% and score threshold 0.005 (Section 4.5.1).

4.5.3 Performance Distribution

Figure 3 presents the distribution of precision and recall over the set of test users for a score threshold of

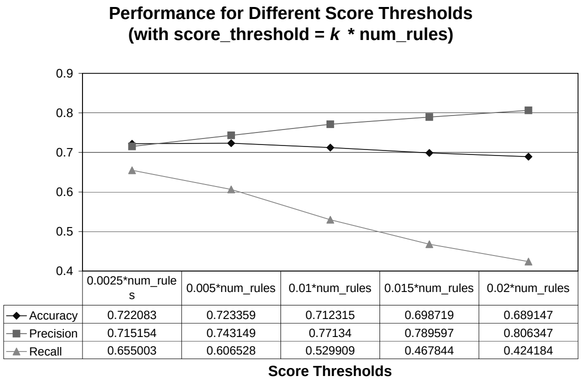


Figure 2: Performance for Different Score Thresholds

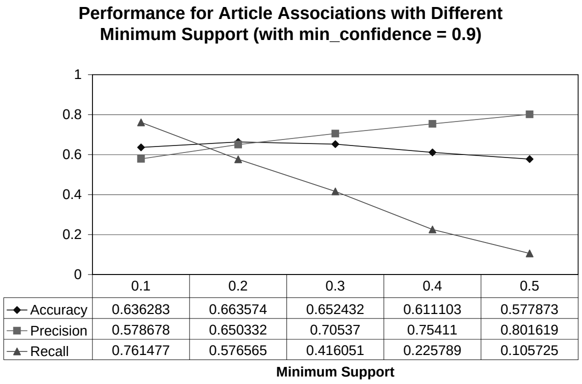


Figure 4: Performance for Article Associations

$0.005 * rule\_num$. We note that if the score threshold is set too high, for example $0.02 * rule\_num$, some users receive no recommendations. Some tuning is required to find a suitable threshold value.

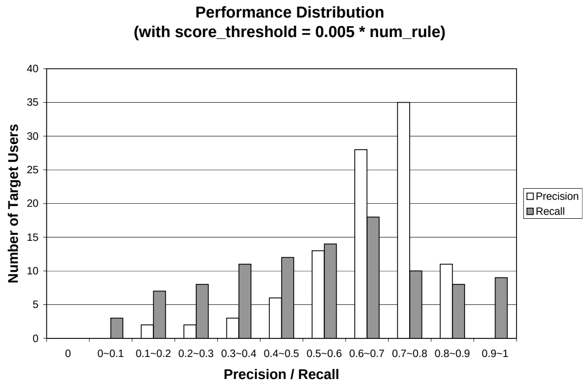


Figure 3: Performance Distribution

4.5.4 Article Associations

Performance for article associations is shown in Figure 4. Performance degrades slightly relative to the user associations mode. However, the running time is reduced considerably, as shown in Table 1.

4.5.5 Combining User and Article Associations

Table 1 lists the performance for user associations, article associations, and combined associations as described in the previous section. Running times shown in the table are in seconds on a 463 MHz Pentium family PC with 128 MBytes of RAM. We can see that when the two types of associations are combined, performance degrades a little bit with respect to the user associations mode, but we achieve a much faster response time. We note that running times are significantly lower than those shown if the size of the training set is reduced. For example, for a training set containing an average of 53 training movies per user, the running time for user associations (with a collaborative user group of 2000 users) is $0.55s$ instead of the value $14.2s$ reported in the table.

	Users	Articles	Combined	Combined
Threshold			0.075	0.1
Accuracy	0.720	0.611	0.717	0.712
Precision	0.751	0.754	0.745	0.723
Recall	0.584	0.226	0.582	0.602
Runtime	14.2s	0.06s	5.2s	4.6s

Table 1: Combining User and Article Associations

4.6 Comparison with other systems

Billsus and Pazzani [4] tested three collaborative recommendation techniques on the EachMovie dataset: a correlation-based method, neural networks paired with Information Gain, and neural networks paired with Singular Value Decomposition. The resulting accuracies are listed in Table 2.

	Correlation	InfoGain/ANN	SVD/ANN
Accuracy	0.644	0.67	0.679

Table 2: Accuracy of Collaborative Approaches

We tested our approach under similar experimental conditions. For the collaborative user group, we tried both the first 1000 users who have rated more than 100 movies, as well as the first 2000 users. We chose 91 random users who have rated 50 to 100 movies as target users and whose like ratios are below 0.75.

We employed the 4-fold cross-validation approach for our tests, which results in that the average number of training movies for each user is 53. The accuracy achieved for the user associations mode of our system varied between 0.67 and 0.69 depending on what sets of training users were selected. In all cases, the performance of our approach was superior to that of the correlation-based method tested in [4]. Our results are inconclusive as regards comparing our system with the SVD-based system described in [4]. Performance depends on the specific test users selected, and no information was available regarding what test users were selected for the results reported in [4].

5 Conclusions

We have described a new collaborative recommendation technique based on a specialized algorithm for mining association rules. Our algorithm automatically adjusts the minimum support so that the number of rules generated lies within a specified range. This reduces running time and provides enough rules for good recommendation performance. We can achieve a faster response by combining user associations and article associations. The performance of our approach is significantly better than that of traditional correlation-based methods.

References

[1] R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. In *Proc. of the ACM SIGMOD Conference on Management of Data*, pages 207–216, Washington, D.C., May 1993. ACM.

[2] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In *Proc. 20th VLDB Conference*, pages 487–499, Santiago, Chile, 1994.

[3] M. Balabanovic and Y. Shoham. Fab: Content-based, collaborative recommendation. *Communications of the ACM*, 40(3):66–72, March 1997.

[4] D. Billsus and M. J. Pazzani. Learning collaborative information filters. In *Proc. Fifteenth International Conference on Machine Learning*, Madison, Wisconsin, 1998. Morgan Kaufmann Publishers.

[5] S. Brin, R. Motwani, J. D. Ullman, and S. Tsur. Dynamic itemset counting and implication rules for market basket data. In *Proc. ACM SIGMOD International Conference on Management of Data*, pages 255–264, New York, May 1997. ACM, ACM Press.

[6] M.-S. Chen, J. S. Park, and P. S. Yu. Efficient data mining for path traversal patterns. *IEEE Trans. Knowledge Data Eng.*, 10(2):209–221, March 1998.

[7] R. Cooley, J. Srivastava, and B. Mobasher. Web mining: Information and pattern discovery on the world wide web. In *Proceedings of the 9th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'97)*, November 1997.

[8] R. Cooley, P.-N. Tan, and J. Srivastava. Web-SIFT: The web site information filter system. In *Proc. of the Workshop on Web Usage Analysis and User Profiling (WebKDD99)*. Available at http://www.acm.org/sigkdd/proceedings/webkdd99/, Aug. 1999.

[9] X. Fu, J. Budzik, and K. J. Hammond. Mining navigation history for recommendation. In *Proc. 2000 international conf. intelligent user interfaces*, pages 106–112, New Orleans, LA, January 2000. ACM.

[10] P. Hájek, I. Havel, and M. Chytil. The guha method of automatic hypotheses determination. *Computing*, 1:293–308, 1966.

[11] P. Hájek and T. Havranek. On generation of inductive hypotheses. *Int. J. Man-Machine Studies*, 9:415–438, 1977.

[12] W. Lin, C. Ruiz, and S. A. Alvarez. A new adaptive-support algorithm for association rule mining. Technical Report WPI-CS-TR-00-13, Department of Computer Science, Worcester Polytechnic Institute, May 2000.

[13] B. Liu, W. Hsu, and Y. Ma. Integrating classification and association rule mining. In *Proc. Fourth International Conference on Knowledge Discovery and Data Mining*, pages 80–86, New York, August 1998.

[14] P. McJones. Eachmovie collaborative filtering data set. http://www.research.digital.-com/SRC/eachmovie, 1997. DEC Systems Research Center.

[15] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. Grouplens: an open architecture for collaborative filtering of netnews. In *Proceedings of the Conference on Computer Supported Cooperative Work (CSCW94)*, pages 175–186, 1994.

[16] U. Shardanand and P. Maes. Social information filtering: Algorithms for automating "word of mouth". In *Proc. Conf. on Human Factors in Computing Systems (CHI95)*, pages 210–217, 1995.

[17] R. Srikant and R. Agrawal. Mining quantitative association rules in large relational tables. In *Proc. 1996 ACM SIGMOD Conference on Management of Data*, Montreal, Canada, June 1996. ACM.

[18] G. I. Webb. Efficient search for association rules. In *Proc. KDD-2000*, Boston, August 2000, to appear.