Utilizing Content Information in an Association Rule Based Recommender System

by

Michael A. Sao Pedro

A Thesis

Submitted to the Faculty

of the

WORCESTER POLYTECHNIC INSTITUTE

In partial fulfillment of the requirements for the

Degree of Master of Science

in

Computer Science

by

August 2001

APPROVED:

Professor Carolina Ruiz, Major Thesis Advisor

Professor Lee Becker, Thesis Reader

Professor Micha Hofri, Head of Department

Abstract

As the amount of information available through different media such as the Internet continues to multiply daily, the need for more efficient ways of finding information of interest easily is becoming increasingly important. One method for sifting out information is to use a recommender system, a system that makes predictions about a target user's tastes or opinions. Our research focuses on extending a system which mines association rules for collaborative filtering [Lin2000] by enhancing the framework to utilize available content information in formulating predictions. Association rule mining is well-suited for recommendation, because association rules can naturally find and describe multiple dependencies between those aspects which determine user preferences, such as article properties or other users' ratings of article properties. Additionally, mined rules come quantified by two measures, the confidence and the support, which measure the strength and the significance of the dependencies among factors, respectively.

To enhance recommendation, we propose mining two new types of associations: content associations, which relate a target user to the properties of articles they find of interest, and content-based collaborative associations, which attempt to find relationships between users through users' ratings of article properties. We test these approaches using the domain of movies. Our results show that we achieve high quality recommendations and in most cases with real time performance. Specifically, our results show that overall, content-based collaborative filtering yields the highest precision, but may fail to produce recommendations for some users. We therefore

combine our content-based collaborative filtering with other system modes, including regular collaborative associations, and content associations, which have slightly lower precision but yield recommendations for all users.

Acknowledgements

First, I would like to thank my advisor Carolina Ruiz for all of her patience, support, caring, and guidance throughout the entire year. All of your support and confidence in me definitely helped me get through the tough times.

I would also like to thank Professors Stanley Selkow, George Heineman, and Kathryn Fisler for giving their time to help me out in any way possible. Most of all, I appreciate the time you three took to listen to my problems and keep me sane through the seemingly infinite amount of work.

I also want to thank Lee Becker for reading the thesis, and saying those magical words "it looks like you are going to graduate".

A standing ovation also goes to Wendy Kogel whose assistance and illustrious Perl script helped me out more than she'll ever know.

A special order of appreciation also goes out to Ji Chen who not only devoted a lot of time to help me with modifying the association rule mining system, but gave me a lot of laughs and good fun throughout the year.

Last but not least, I would like to thank all of my friends, especially Amy and Sharad, and family for all their love, support, guidance and encouragement. You guys helped me pull it all together to see this project through. Finally, I totally appreciate all your understanding that graduate school really does suck up your entire life.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

As the amount of information available through different media such as the Internet continues to multiply daily, the need for more efficient ways of finding information of interest rapidly and easily is becoming increasingly important. One method for sifting out information relative to the current user, or *target user*, at hand is to utilize a recommender system. Recommender systems are tools which make predictions about a target user's tastes or opinions. Such systems are useful in domains such as e-commerce, because recommender systems can be used to suggest items the target user would like to buy based on some previous information either given explicitly by the target user, or implicitly determined through the target user's interactions with the system. Recommendation services range from suggestions about books given while browsing Amazon.com^{TM}, to suggesting one's favorite TV programs using TiVo^{TM}'s digital television system.

Recommender systems utilize two general strategies for formulating these predictions. These two strategies are content-based recommendation and collaborative recommendation. Content-based recommendation uses the properties of articles to recommend similar articles with those properties. For example, if a target user has bought two Backstreet Boys CDs and one NSYNC CD, a recommender system may use the properties of those CDs, namingly the fact that they are pop CDs by boy bands, to determine the target user may like another CD with those properties such as an O-Town CD. Collaborative recommendation on the other hand tries to group people who have similar tastes in order to make recommendations. Continuing with the boy band example, if the target user has purchased those two CDs and a group of other people have also purchased those CDs in addition to an O-Town CD, the O-Town CD will be recommended to the target user.

Both content-based and collaborative filtering have strengths and drawbacks. Collaborative filtering works well, because user's opinions tend to capture more information about unquantifiable properties such as emotions than content-based filtering. However, collaborative filtering fails to work when users do not rate or purchase the same articles. Content-based filtering works well when the user has not rated or purchased many items, however determining the best combinations of article properties to yield precise predictions is difficult. Because each method has their own respective strengths, many attempts such as [CGM99], [BPZ00], and [SN98] have been made to "combine the best of both worlds", thus developing some hybrid of the two approaches.

Our work utilizes association rules in order to construct recommendations. In particular, we continue the work presented in [LAR02] which used association rules for collaborative recommendation. We extend the model to handle content-based recommendation, and then a combination of the two approaches. Association rule mining [AIS93] is a data mining approach which attempts to find relationships between items in a transactional database. For example, association rules can be used to find trends between items purchased in a supermarket. If a record of all transactions made at the supermarket, namingly records of what items each person bought, that information can be used to find association rules of the form $X \Rightarrow Y$, where X and Y are disjoint sets of item, that describe these relationships between items. One such association rule r could be: r = *beer* $\wedge$ *diapers* $\Rightarrow$ *antacid* $\wedge$ *potato chips.*

Association rule mining is well-suited for the recommendation domain for a couple of reasons. The first reason is that association rules naturally capture relationships between items and preferences. Therefore, rules such as: r_1 = *[User1: Like]* $\wedge$ *[User5: Like]* $\Rightarrow$ *[Target User: Like]* for collaborative recommendation and r_2 = *[Genre = pop music]* $\wedge$ *[Group Type = boy band]* $\Rightarrow$ *[Target User: Like]* for content-based recommendation can be mined. The second reason is that association rules carry two measurements which quantify their significance. The first measurement, support, counts the frequency of rule items appearing together in the database of transactions. Therefore, if *Genre = pop music, Group Type = boy band*, and *Target User: Like* all appeared together in 30% of the transactions, then the support for r_2 would be 30%. The second measurement, confidence, determines the strength of the correlation between items in the left hand side of the rule, or *antecedent*, and right hand side, or *consequent* which contains the target user's preference. More precisely defined, confidence measures the frequency of the presence of *[Target User: Like]* among those transactions which contain the antecedent of the rule. Therefore, if 10 transactions contain *Genre = pop music*, and *Group Type = boy band*, and of those 10, *[Target User: Like]* appears in 8 transactions, the confidence would be of r_2 would be 80%.

The goal of our work is to evaluate the quality of the results returned by content-based recommendation and content-based collaborative recommendation, in which article properties are used to match user similarities instead of the articles themselves, via association rules. To this end, we chose to use the domain of movie recommendation in order to test our approaches. Three different datasets, the EachMovie dataset [McJ97], the Movies Database [Wie98], and the GroupLens dataset [Her98] were used to build the content base for our experiments. Ratings for movies were acquired from the EachMovie dataset.

Our experimental approach is decomposed into the following steps. First, we develop the framework for mining content-based rules. We then test two alternative methods for scoring those rules to make predictions. The first method uses a linear score threshold based on the number of rules returned by the mining algorithm. The second uses a combination of positive-class rules, those which contain *[Target User: Like]* in the consequent, and negative-class rules, those which contain *[Target User: Dislike]* in the consequent to make predictions. Finally, we construct and test the framework for mining content-based collaborative rules, which yield the same type of rules as regular collaborative recommendation, but are generated using derived user's ratings of movie properties instead of the explicit movie ratings themselves.

Our experiments show that association rule mining is well-suited for recommendation. In particular, the content-based collaborative approach yields a higher precision in recommendation than both the regular collaborative and content approaches. However, fewer target users receive results from the content-based collaborative approach. We therefore developed an algorithm to ensure that everyone receives predictions. This algorithm first tries to produce recommendations using the most precise method, in this case the content-based collaborative method. If users do not receive any predictions, we fall back on the second most precise method, regular collaborative associations. If some users still do not receive predictions, content associations, which can produce recommendations for everyone as shown through our experimentation, are used to make predictions.

The remaining portion of this document is organized as follows:

- *Chapter 2: Background*: contains the necessary background information required for understanding the framework and experiments.

- *Chapter 3: Related Work*: contains previous approaches to content-based and collaborative filtering and the results obtained from those approaches.

- *Chapter 4: Using Content Information for Recommendation*: discusses our framework for mining content-based and content-based collaborative rules.

- *Chapter 5: Experimentation and Evaluation*: contains an in-depth description of the datasets and testing protocol as well as the different experiments performed and results acquired.

- *Chapter 6: Our Recommendation Strategy* contains our overall approach to recommendation which combines content associations, collaborative associations [LAR02], and content-based collaborative associations in making predictions.

- *Chapter 7: Conclusions*: contains a summary of the experiments and future work.

Chapter 2

Background

This chapter presents the background material which provides the framework of our research. First, a discussion of recommender systems is presented, followed by an introduction to association rules and rule mining. Finally, an overview of the system developed in [LAR02] for mining association rules for use in recommender systems is presented.

2.1 Recommender Systems

In a nutshell, a recommender system is an entity that helps a user, called the *target user*, find information they would like to see. As a result, the recommender system filters out the information they would not like to see. For example, in e-commerce a recommender system might be used to suggest products the user is more likely to purchase, thus filtering out those products that are less interesting to the user. As another example, a recommender system could be used to recommend news articles a user is more likely to enjoy and display those before any other article.

Many different companies currently use various types of recommender systems to aid their users in finding the information most relevant to them. Amazon.com^{TM}

(www.amazon.com) recommends books to users frequently purchased by customers who purchased the selected book. Additionally, the recommender system also recommends other authors of books based on customers who purchased the selected book. These two recommendation schemes hence are based on the purchase histories of other users.

CDNOW^{TM} (www.cdnow.com) provides a feature called "My CDNOW" in which the user first creates a profile, and then tells the system which CDs they already own and if they like those CDs. With this information in conjunction with their purchase history, assuming that the user likes the CDs they bought from CDNOW, the system predicts another 6 albums that the user is most likely to enjoy. This system, in contrast to Amazon.com^{TM}'s system, allows the user to enter information which may aid in providing better recommendations.

Moviefinder.com (www.moviefinder.com) provides a feature named "We Predict" in which customers are asked to enter their ratings for movies they have seen using a 5-point scale. The recommender system then makes two different kinds of predictions using this information. The first type presents the user with unrated movies the system thinks they should either see or not see as they navigate through the system. The second type, called "Powerfind", lets the user search the database based on movie attributes such as genre, actors, or directors. "Powerfind" can either utilize the user's ratings to produce a personalized list of those movies matching the search criteria sorted according to the derived likelihood of the user enjoying the movie, or present those same movies sorted by other customers' average ratings. Therefore, Moviefinder.com's recommender system takes full advantage of both users' ratings and movie attributes [SKR01].

All of the recommender systems described take different forms of input information to yield various forms of output recommendations. Inputs can be entered by the user as ratings or search criteria. Any input given by the user to aid in recommendation is known as *explicit* information, because the user explicitly tells the recommender system their preferences. In contrast, CDNOW.com^{TM} makes an assumption that the user likes what they have purchased in order to make predictions. Such information is called *implicit* information, because the system derives information based on the user's interaction with the website. Other forms of implicit information include: time spent viewing an item and tracking mouse clicks. The output, or actual recommendations, range from a single bit (recommended or not) to lists of items. The items can be sorted based upon the system's predictions of the target user liking the item. The system could also present a fixed number of appealing items, much like a "top-10" list.

2.1.1 Recommender System Taxonomy

Recommender systems fall under two genres: *non-personalized* and *personalized.* A non-personalized recommender system suggests items to a target user based on what other users felt about those items on average. These recommendations are independent of the user; therefore each user gets the same recommendations. An example of a non-personalized recommendation scheme would be to always show the 10 most frequently viewed items first. [SKR01]

In contrast, personalized recommender systems make use of target user information in order to construct unique predictions for that target user. More generally, each person gets different predictions from the system. Personalized recommendation can further be broken up into three different schemes for producing predictions:

- *attribute-based*: An attribute-based recommender system shows items to the target user based on qualitative or quantitative descriptions of those items. These systems require users to explicitly state which items or characteristics of items interest them. Attribute-based recommender systems are similar to search engines in which the user types in specific "qualities" to match.

- *content-based*: A content-based recommender system yields predictions based on the attribute similarities between items. The system uses explicit or implicit information about the target user's interest in a small set of items and makes suggestions based on qualitative and quantitative similarities between items that the target user has not yet viewed.
- *collaborative-based*: A collaborative-based recommender system generates predictions based on the opinions of other users who have expressed the same feelings as the target user on a small set of items. For example, if a user A and a user B rate a set of items in the same way, and user B likes an item user A has not seen, then chances are user A will also like that item. This differs from the non-personalized approach, because the system attempts to group people with similar tastes and make similar predications only for people within a group, called the "collaborative group".

A recommender system can adopt any one or a combination of these schemes to make a prediction [SKR01].

Most research in recommender systems focuses on content and collaborative-based techniques. We will now discuss some of these techniques in detail.

2.1.2 Quality of a Recommender System

The quality of a recommender system can be defined in many ways. Most generally, recommender systems that can accurately and consistently predict what a user would find interesting are of high quality. This idea of quality can depend on the abilities of the algorithms to properly compute predictions, the speed of the computations, and even the space required to store what is necessary for the calculations. The quality can also depend on the meaning and presentation of the final output. Prediction algorithms have many dimensions on which quality of a recommender system can be measured. These dimensions are: coverage, statistical accuracy, decision-support accuracy, precision, and recall. These notions are defined as follows:

- *coverage*: Coverage is a measure of the percentage of items for which a recommender system can provide predictions. Coverage applies more specifically to collaborative filtering, because small groupings of neighbors and unrated items cause certain items to have no valid predictions, which affects coverage. For example, if nobody rated 10 items, then there is no way to determine if any of the users would like or dislike those items based on pure collaborative filtering, thus lowering the coverage for collaborative predictions.

- *Decision-support accuracy*: Decision-support accuracy measures how effectively predictions help a user select high-quality items. For example, a poor decision-support accuracy would be reflected when the highest rated item for a user is .65, and the user is only shown items which rate higher than .70 [HKBR99].

- *Precision*: Precision is a direct measure of how well the system can actually find items a user likes. A recommender system is precise when it recommends items a user likes and does not recommend items the user will not like. Specifically, precision is defined as the number of items the system correctly determines that the user likes out of all the items recommended by the system.

- *Recall*: Recall measures the ability of the system to find all the items a user likes. More precisely, recall is defined as the number of items the system correctly finds that the user likes out of all the items liked by the user. A perfect recall can easily be achieved by recommending every item to the user; however this will significantly lower the precision, because if the system recommends everything, the system will also recommend items the user does not like. Thus, there exists a tradeoff between precision and recall.

In addition to filtering algorithms, quality can also be defined in terms of the system's final output. There are two basic classes of displaying filtered information to users. The first class involves the presentation of items one-at-a-time to the users along with a rating indicating potential interest in the topic. The second class involves the presentation of some ordered list of recommended items. In the first case, the user can be presented with the actual raw score from the filtering engine. In the latter, the same scores could be used to sort the list, hiding the actual percentage match from the user. This sorted list can be presented to the user either in a linear fashion, or in some form of tree [BHK98]. Furthermore, only a limited number of items could be presented to the user, which may affect the precision of the entire system.

2.2 Association Rules

Association rule mining, introduced in [AIS93] is one procedure for performing knowledge discovery, a process which attempts to extract useful patterns or trends from a large dataset. In particular, association rules are known for their ability to inherently describe relationships and strengths of those relationships between items in a database. The definition of an association rule is given more formally as follows:

Given a database D of transactions where each transaction $t \in D$ is a subset of items from a set of items I, an association rule in D is described as $X \Rightarrow Y$, where $X \subset I$, $Y \subset I$, and $X \cap Y = \emptyset$. X is known as the body or antecedent of the rule, and Y is known as the head or consequent of the rule. For example, assume we have a database composed of all the sales made at a grocery store. Each transaction t would represent one person's basket of items they purchased. The set of items I would be all of the items that have been purchased from the store. An association rule r derived from the set of everyone's purchases could be: r = *beer* $\wedge$ *potato chips* $\wedge$ *buffalo wings* $\Rightarrow$ *cola* $\wedge$ *antacid*, where the antecedent of the rule is: *beer* $\wedge$ *potato chips* $\wedge$ *buffalo wings*, and the consequent is: *cola* $\wedge$ *antacid.*

Generally, an association rule is quantified by the support and confidence, two measurements which capture the statistical significance of the rule. The support of a rule, or $supp(X \Rightarrow Y)$, is the percentage of the transactions in D that contain both X and Y. Continuing with the grocery store example, the support of r would be the percentage of transactions in D that have beer, potato chips, buffalo wings, cola, and antacid in them. The confidence of the rule, or $conf(X \Rightarrow Y)$ is the percentage of the transactions that contain Y among those that contain X. Therefore, the confidence of r would beg the question: "out of the transactions containing beer, potato chips, and buffalo wings, what percentage of these also contain cola and antacid?" Another way to define the support and confidence is through probability with respect to the database D as follows:

- $supp(X \Rightarrow Y) = Pr(X \cup Y) = \frac{|X \cup Y|}{|D|}$
- $conf(X \Rightarrow Y) = Pr(Y|X) = \frac{Pr(X \cup Y)}{Pr(X)}$

To illustrate support and confidence, consider an example database of transactions from a grocery store:

In the example database, the support of the rule r = *beer* $\wedge$ *potato chips* $\wedge$ *buffalo wings* $\Rightarrow$ *cola* $\wedge$ *antacid* is equal to 40%, because two transactions (2 and 3), out of all five have all of the items in the antecedent and consequent of the rule.

transaction #	*items purchased*
1	beer, diapers, potato chips, buffalo wings, cola
2	beer, diapers, potato chips, buffalo wings, cola, antacid
3	beer, potato chips, buffalo wings, cola, antacid
4	beer, diapers, potato chips
5	buffalo wings, cola

Figure 2.1: Example database of transactions

The confidence of r is equal to 66%, because out of the transactions containing the antecedent of the rule r (1, 2 and 3), only two also contain the consequent of r (2 and 3).

2.2.1 Association Rules for Classification

One restriction that can be made about association rules is that their consequents can only contain only value coming from a separate set of possible classification values. Now, a transaction t composed of a subset of items from the itemset I also has one classification from a set of classifications $c \in C$. Such association rules are called classification association rules. For example, classification rule mining could be used to determine which factor or factors contribute to a person making more or less than $50,000 per year given a database of census information. The possible classification values in this case would be (1) making $50,000 or more per year and (2) making less than $50,000 per year. In recommendation, classification rule mining can be used to determine which properties pertain to a user liking or disliking a target item. Because classification rule mining is the essence of performing recommendations via association rules, the classification rule mining process is discussed in further detail below.

2.2.2 Classification-Based Rule Mining Algorithm (CBA-RG)

Classification-based rule mining, proposed in [LHM98], is based on the well-known Apriori algorithm [AS94] for mining regular association rules. The main difference between the two mining approaches lies in the fact that classification association rules have one and only one classification value in the consequent attained from a set of possible classification values which is separate from the set of items I, whereas regular association rules contain some subset of items from the itemset I. The fact that only one element can appear in the consequent of a rule allows for speedups in rule generation. In particular, once the Apriori algorithm finds a set of items which are frequent, meaning that the support for those items from I lies above the minimum support, the algorithm must try all possible combinations of these items to form rules which are above the minimum confidence. The CBA-RG algorithm, in contrast, does not need to manipulate the frequent items to generate rules, because the rules' consequents can only contains one element from a set of classification values. The algorithm for generating classification association rules is given below:

The purpose of the algorithm is to find all classification association rules that are above a minimum support and minimum confidence, which are both user-specified. The algorithm works by finding all rules whose antecedents are of length 1, and using those rules' antecedents to generate new rules with antecedents of length 2 and so on. Those rules of length k at a given stage of the algorithm which are above the minimum support contain frequent items in their antecedents. Those frequent items are then used to generate the rules of length $k+1$. Although the rules are above the minimum support, they may not be above the minimum confidence and thus may not be a part of the final set of rules produced from the mining algorithm.

```
$F_1 = \{\text{frequent 1-ruleitems}\}$;
$CAR_1 = genRules(F_1)$;
$prCAR_1 = pruneRules(CAR_1)$;
for ($k = 2$; $F_{k-1} \neq \emptyset$; $k++$) do Begin
  $C_k = candidateGen(F_{k-1})$;
  for each data case $d \in D$ do Begin
    $C_d = ruleSubset(C_k, d)$;
    for each candidate $c \in C_d$ do Begin
      $c$.condsupCount++;
      if $d$.class = $c$.class then $c$.rulesupCount++;
    end
  end
  $F_k = \{c \in C_k \mid c.rulsupCount \geq minsup\}$;
  $CAR_k = genRules(F_k)$;
  $prCAR_k = pruneRules(CAR_k)$;
end
$CARs = \bigcup_k CAR_k$;
$prCARs = \bigcup_k prCAR_k$;
```

Figure 2.2: The CBA-RG algorithm

Therefore the three key portions to this algorithm are (1) the candidate itemset generation step [step 4], the frequent rule generation step [steps 7 - 13], and (3) the rule generation [step 14].

Candidate Itemset Generation Step

A candidate itemset in this framework is composed of a group of items from I and one classification value from the set of possible classification values $c \in C$. As in the Apriori algorithm [AS94], CBA-RG generates new candidate itemsets using a "join" step and then a "prune" step. These two steps are used in order to reduce the possible search space for rules. In particular, this search space reduction works off of the Apriori Principle which states that an itemset of size k cannot be more frequent than any of its subsets of size $k-1$. For example, given two itemsets, $\{A, B\}$ with support 30%, and $\{B, C\}$ with support 20%, the candidate itemset $\{A, B, C\}$ can have a support of at most 20%. The significance of this fact is that pruning out items earlier in the process means that they do not have to be considered again to try and make rules, because if they are not frequent for smaller itemsets, they also will not be frequent in larger itemsets. Thus only itemsets and the items in them that were previously frequent need to be considered.

In the "join" phase, each of the frequent itemsets of size $k-1$ are joined with another item to form frequent itemsets of size k. These new candidate itemsets are generated according to the following formula:

$$C_k = \{p \cup q | p, q \in Freq_{k-1} \wedge$$

$$p.item_1 = q.item_1, ..., p.item_{k-2} = p.item_{k-2}, p.item_{k-1} < q.item_{k-1}\}$$

In other words, all the unique itemsets of size $k+1$ are derived by combining only the frequent items of size k.

After the candidate sets of length k have been generated, they are pruned. A candidate set of length k is pruned if any one subset of that candidate set is not found in the set of frequent itemsets of length $k-1$ used in the previous join step. This step also follows the Apriori Principle, because if the subset is not found, then there is no way that subset could have been frequent.

Frequent Rule Generation Step

Once the candidate itemsets have been generated, those which are frequent must be selected so that they can be used to generate rules and potentially larger frequent itemsets. In this stage, each transaction in the database is observed to tally, one by one, if (1) the antecedent of the rule appears in a transaction, and (2) if both the antecedent and consequent of the rule appears in a transaction. If the antecedent and consequent of a rule r appears more frequently than the minimum support, then the union of the two is added to the set of frequent itemsets, otherwise it is removed.

Rule generation step

Using both tallies from the previous frequent rule generation step, those rules which are valid classification association rules can be derived. A classification association rule is added to the final set if its confidence is greater than the user-specified minimum confidence. This value is calculated for a rule r in the following manner:

$$conf(r) = \frac{\#\text{ times both antecedent and consequent of } r \text{ found in all transactions}}{\#\text{ times antecedent of } r \text{ found in all transactions}}$$

Rule pruning step

An optional rule pruning step can be performed in order to reduce the number of rules generated. A rule is pruned based on the pessimistic error rate used in the decision tree generating algorithm C4.5 [Qui92]. A rule is pruned if its pessimistic error rate is greater than the pessimistic error rate of that same rule with one of its antecedent items randomly removed.

2.3 Using Classification Rule Mining to Produce Collaborative Recommendations

The basic idea behind this recommendation scheme starts with the rule representation. The target user, the person for whom the predictions are made, is the target attribute and thus will appear in the consequent of every rule mined. The antecedents of those rules are made up of the preferences of other users in the system. Therefore, a rule such as: $r = [user1 : like] \wedge [user5 : like] \rightarrow [targetuser : like]$ could be mined from the appropriate input data. These rules can then be used to make predictions about different items in the system. For example, if $user1$ and $user5$ have rated an article a as "like", and the target user has not rated a, then the article a may get recommended to the target user.

There are two advantages achieved when using classification rule mining to make collaborative predictions over the other methods mentioned [LAR02]. These are:

1. Rule mining has two measures which can be used to gauge correlations, namingly the support and confidence. More precisely, the confidence measures the degree of correlation between users whereas the support measures the significance of these correlations.

2. Rule mining naturally allows for exploring if multiple users' opinions together predict very well for the target user. As shown in the user association rule example above, multiple users' relation to the target user can be acquired

	user 1	*user 2*	*user 3*	*user 4*
article 1	1	0	?	?
article 2	0	1	1	?
article 3	0	1	1	?
article 4	1	1	0	1
article 5	0	1	0	?

Table 2.1: Example dataset for mining user associations.
Here, "1" means the user liked the article, "0" means the user disliked the article, and "?" means the user did not rate the article. Note that the target user does not appear in this table.

which may yield more useful information than solely looking at each other user separately with respect to the target user.

The work presented in [LAR02] describes the input data required to mine such rules, presents a variation of the CBA algorithm for mining collaborative classification rules quickly, and describes a method for calculating predictions. Each of these facets is described in further detail below.

2.3.1 User Associations

User associations are rules that describe the target user's preference based upon the preferences of other users. The example rule $r = [user1 : like] \wedge [user5 : like] \rightarrow [targetuser : like]$ mentioned previously is one such example of a user association. User associations are mined over a set of transactions, each of which represents an "article" or "product" rated by any user. Therefore, all of the users *excluding target user* make up each transaction. The target user liking or disliking in this case represent the target classes which will always appear in the consequent of the rules mined. Table 2.1 gives an example of such data. The example database presented in Table 2.1 can be used to generate transactions for mining. [LAR02] determined that only viewing what users liked was most effective in the recommendation process as

transaction #	*itemsets*
1	user 1
2	user 2, user 3
3	user 2, user 3
4	user 1, user 2, user 4
5	user 2

Table 2.2: Transactions generated from the example dataset for user associations. The presence of a user in a transaction indicates that this user liked the article represented by that transaction.

opposed to viewing what users liked and disliked. Therefore, if a user did not rate or did not like an article, the user did not appear in a transaction. For example, the Table 2.1 would yield the transactions presented in Table 2.2.

This information along with the target user's likes and dislikes for the articles can then be used as input to the data mining algorithm.

2.3.2 AR-CRS Algorithm

[Lin00] presents the AR-CRS algorithm for mining classification-based association rules, all of which contain a single, unique classification value. This is different from the CBA algorithm discussed previously, because CBA mines rules which can contain any one classification value from a set of possible classifications. Therefore, given data for mining user associations, the AR-CRS algorithm could only return sets of rules reflecting the target user's likes, or the target user's dislikes, but not a mix of rules where some contain "like" in the consequent and some contain "dislike".

Additionally, unlike CBA where the user is required to enter the minimum support and minimum confidence, AR-CRS finds the optimal minimum support for a desired minimum confidence within a minimum and maximum number of rules specified by the user. The motivation for letting the system find the minimum support for a rule range comes from the following observations as discussed in [LAR02]:

- Specifying both the appropriate minimum support and the minimum confidence for mining rules for recommender systems on a per-user basis is difficult. This difficulty exists, because users' tastes and articles' popularities vary widely. If the minimum support and confidence are set too high, too few rules will be mined for recommendation. If the minimum support and confidence are set too low, too many rules will be mined.
- A large number of rules are not needed for recommendation.

AR-CRS first determines the initial minimum support based on the like ratio of the target user, which is the prior probability of the target user liking a movie. From there, AR-CRS increases and decreases the minimum support, rerunning the mining algorithm, until the appropriate number of rules are found within the specified range and above the minimum confidence. If too few rules are mined for one iteration, the minimum support is decreased so that more rules can potentially be mined in the next iteration. Similarly, if too many rules are mined the minimum support is increased. The formal algorithm is presented Figure 2.3 and Figure 2.4:

In context of user associations, the transaction data and target user article rating data described previously, along with a minimum confidence and rule range are all specified to yield user association rules. In order to make recommendations, the rules are scored to yield final predictions.

2.3.3 Scoring the Rules

Given a set of rules mined for a particular target user, the recommendations are calculated using the following scoring function defined in [LAR02]:

$$score_a = \sum_{r \in FireRules(R,a)} support_r * confidence_r$$

```
minsupportCount = 0;
R = AR-CRS-1();
while (R.rulenum = maxRulenum) do
    minsupportCount++;
    R = AR-CRS-1();
end
while (R.rulenum < minRulenum) do
    minsupportCount--;
    R = AR-CRS-1();
end
return R;
```

Figure 2.3: The AR-CRS1 algorithm.

```
$F_1 = \{\text{frequent 1-condsets}\}$;
$R = \text{genRules}(F_1)$;
if R.rulenum = maxRulenum then return $R$;
for $(k = 2; F_{k-1} \neq \emptyset; k++)$ do Begin
    $C_k = \text{candidateGen}(F_{k-1})$;
    for each transaction $t \in D$ do Begin
        $C_t = \text{subset}(C_k, t)$;
        for each candidate $c \in C_t$ do Begin
            c.condsupCount++;
            if t.contain(targetItem) then c.rulesupCount++;
        end
    end
    $F_k = \{c \in C_k \mid c.\text{rulesupCount} \geq minsup\}$;
    $R = R \cup \text{genRules}(F_k)$;
    if R.rulenum = maxRulenum then return $R$;
end
return $R$;
```

Figure 2.4: The AR-CRS2 algorithm.

where:

- a is an article
- R is the set of rules returned by the mining algorithm
- $r \in FireRules(r, a)$ if the ratings for a satisfy the antecedent of r. For example, if $r = [user2 : like] \wedge [user3 : like] \rightarrow [TargetUser : like]$, then in order for r to fire for a, user 2 and user 3 must like article a.

Those articles which fall above a system-defined threshold are recommended to the user. The threshold can be determined in two ways as described in [LAR02]:

1. Use a constant as the threshold. Therefore, if the final score for an article is above this constant threshold, the article is recommended.
2. Use a linear function, based on the number of rules mined. This function can be defined as follows: $threshold = k * \#rules + b$, where k is a constant for the slope and b is an offset parameter. Again, if the score exceeds this threshold, the article is recommended.

Chapter 3

Related Work

This chapter presents different approaches for performing content-based and collaborative filtering. We present overviews of the different approaches and analyze their results. Finally, we give an overview of how our approaches using association rule mining compares to these other approaches.

3.1 Current Content Recommendation Techniques

In this section we observe different methods for determining the level of similarity between two items based on their attributes, as well as using this similarity to derive predictions.

3.1.1 Word Matching in an Online Newspaper

Claypool, et.al. in [CGM99] developed a recommender system for an online newspaper which combines predictions from both collaborative and content-based filters based on explicit and implicit information. A user's content profile is divided into sections, representing each part of the newspaper. A user can explicitly state different keywords of interest for each section. For example, a user could specify the Dallas Cowboys in the Sports section as a keyword, and 1997 Ford Explorer in the Classified section. The content profile also contains a list of implicit keywords, derived from articles the user has rated highly. This list of keywords is derived by (1) removing stop words such as "a", "an", and "the", (2) performing word stemming, for example removing the gerund form of verbs, and (3) taking the first quartile of those remaining words appearing most frequently in the highly rated articles.

Each different piece of content information: the section, the explicit list of keywords, and the implicit list of keywords, is treated separately. The predictions yielded from each piece of information is combined with equal weight to form the final content-based prediction. Choosing the articles based on section is trivial; simply return those articles from the appropriate section. In order to determine articles to recommend using the other two pieces of information, a word-similarity match between a target article and either keyword list is calculated. The degree of match M is calculated according to the following equation:

$$M = \frac{2(|D| \cap |Q|)}{min(|D|, |Q|)}$$

where:

- D: set of keywords extracted from the target article
- Q: set of keywords from a keyword list

Those with a high degree of similarity are recommended.

The results obtained from the content-based predictions are significant, because as people interact with the system longer, the amount of inaccuracy tended to increase. Additionally, this approach suffers from the fact that when calculating the degree of match, each relevant word is treated with equal weights. Therefore, even though the presence of a few keywords may be of great significance, this information cannot be taken into account when making recommendations. Our approach, in contrast, could determine the significance of the keywords and combinations of keywords using the support and confidence as the measurements.

3.1.2 A Content Based Filter for Recommending Greeting Cards

Bhojnagarwala, Sao Pedro, and Zebrowski in [BPZ00] created a recommender system utilizing collaborative and content-based filters to recommend greeting cards using a "top-N" approach. Unlike news articles which contain only words, greeting cards also have other discrete properties such as: price, width, height, and manufacturer. In order to use all content-based information, each different piece of content information is compared to that same information in the target user's profile. The final prediction for a target card is calculated using a weighted sums of the similarities between each piece of content data and the user's profile. This profile is constructed by assuming any cards purchased by the user were liked.

The uniqueness in this approach lies in how content similarity is calculated. Both the value of an attribute stored in a user's profile, and the similarity calculations depended on the type of attribute encountered. These different attribute types are outlined as follows:

- *categorical*: Categorical attributes are attributes coming from a discrete set of values that have no order. An example of a categorical attribute is a card's manufacturer. A categorical attribute in the user's profile is calculated by finding the mode value using each of the cards the user purchased. The similarity between the profile and target card is determined by checking for an exact match with the profile. If the two attribute values matched exactly, then they were 100% similar; otherwise they did not match at all.

- *numerical*: Numerical attributes are either discrete or continuous and have a total ordering. An example of a numerical attribute is a card's price. A numerical attribute in the user's profile is calculated by finding the median value of all the prices from each purchased card. The similarity between the attribute for the target card and the attribute in the user's profile is computed using the percent difference between both values. For example, if the profile's value for price is \$2.65 and the target card's value for price is \$2.45, then the similarity value is: $100\% - \frac{\$2.65-\$2.45}{\$2.65} = 92.5\%$ similarity.

- *text*: The same approach in [CGM99] described previously was used in creating this attribute's value in a profile. The calculation for percent similarity was also the same, except that the value for percent similarity was divided by two.

This approach suffers from the fact that the weights for determining the relevance of the content information must be specified. Additionally, the relevance of combinations of attributes is not taken into consideration. Our approach is able to determine the significance of relations of content attributes using association rules and the confidence and support measures. Furthermore, our approach can also compare to this method, because our approach can be extended to handle continuous data such as prices by discretizing that data [Mit97].

3.1.3 Latent Semantic Indexing (LSI)

Soboroff and Nicholas in [SN98] created a content-based collaborative filter using Latent Semantic Indexing for use in predicting users' taste in technical scientific abstracts. A content-based collaborative filter uses content information in each user's profile to determine which users have similar tastes. Each user's profile is constructed of all the documents they have rated. The SVD projects the rated documents into an r-dimensional space, also known as LSI space. Each singular value along the diagonal of gives the relative importance of the dimensions. Finally, the two profiles can be compared by taking the dot product of their LSI representation.

The LSI approach produced good average precision values for lower recall values, but much lower average precision for high recall values. The best average precision of about 57% was obtained for a recall of 10%. Our approach compares to the LSI approach, because both approaches attempt to find the significance of combinations of content attributes.

3.2 Current Collaborative Recommendation Techniques

In this section we observe different techniques used in computing predictions based on user similarities.

3.2.1 Correlation-based Method

Perhaps the most well-known approach used in collaborative-based recommendation is calculating correlations between users. In particular, the Pearson's Correlation coefficient is most commonly used to compute user similarity [RNM$^+$94]. Essentially, this equation creates a collaborative group using all of the users who have rated the target article t in order to make a prediction. If a particular user rates items in the same way as the target user, then they are highly correlated, and that user's opinion is weighted highly in making the prediction. In other words, the more closely the target user matches another user's predictions, the more that user's predictions will influence the final prediction for the target article t. The similarity value r_{ij} between two users i and j is defined as follows:

$$r_{ij} = \frac{\sum\limits_{x} (i_x - \bar{i})(j_x - \bar{j})}{\sqrt{\sum\limits_{x} (i_x - \bar{i})^2 \sum\limits_{x} (j_x - \bar{j})^2}}$$

where:

- x = set of articles both user i and j rated.
- $\bar{i}$ = average ratings given by user i using all articles i rated.
- $\bar{j}$ = average ratings given by user j using all articles j rated.
- i_x = rating given by user i for an article in x.
- j_x = rating given by user j for an article in x.

This correlation coefficient can then be used in another formula to derive a predicted rating for a target article t for a target user U as follows:

$$U_t = \bar{U} + \frac{\sum\limits_{\text{j} \in \text{raters of t}} (j_t - \bar{j}) r_{uj}}{\sum\limits_{\text{j} \in \text{raters of t}} |r_{uj}|}$$

where:

- r_{uj} = similarity calculation between users u and j
- $\bar{j}$ = average rating given by user j using all articles j rated
- $\bar{U}$ = average rating given by target user U using all articles U rated

According to the above equation, if no ratings for an item t are available, then the prediction is equal to the mean of all the ratings from the target user.

This procedure and those similar, suffer from three drawbacks:

1. The correlation approach forces one model of computing similarity between users, instead of treating positive and negative ratings separately. In other words, the correlation approach meshes the two ratings together instead of searching for each separately. For example, one user's negative ratings may be the perfect predictor for what another user likes if these users rated common articles oppositely. Thus, potential information may be lost. [BP98]

2. The correlation between two user profiles can only be determined using items both users have rated. Therefore, if both users have rated nothing in common, their similarity cannot be determined. Furthermore, if there are many items in the system, chances are users will not have a chance to rate many items in common. Therefore, the likelihood that similarity is based on a small number of items in common is high. [BP98]

3. The correlation approach only works in a system where the user must rate items in order to receive predictions. [FBH00]

3.2.2 Clustering Method

Ungar and Foster in [UF98] present formal statistical models for performing collaborative filtering. They propose that both articles and people fall into different classes. For example, if movies and people's ratings for movies are considered, the movies may fall into different classes such as: action, foreign, or documentary, and people may also fall into different classes such as: intellectual or melodramatic. The recommendation idea is that one class of people will tend to like one class of movies.

A statistical model is then estimated to try and group people and movies into different classes in order to make recommendations. These model estimations were generated using two different approaches: repeated K-means clustering, and Gibbs sampling. Tests were performed using purchase data from CDNow, but the results were not reported in [UF98].

This idea is somewhat similar to our content-based collaborative filtering method in that we also try to group movies into "classes", except that our "classes", which are the movie's properties, are well-defined a priori. Our approach also has the advantage that we do not force people into groups, but instead try to find dependencies between users' tastes.

3.2.3 Neural Networks with Input Data from the Singular Value Decomposition (SVD)

Billsus and Pazzani in [BP98] use a combination of neural networks and the Singular Value Decomposition (SVD) to yield collaborative predictions. The SVD reduces the dimensionality of the data, which essentially shrinks the size of the collaborative group. The neural network then uses this reduced data as its input nodes to yield collaborative predictions.

[BP98] uses the same database we use for attaining user ratings, in particular the EachMovie database [McJ97]. In particular, their results for this database showed a higher precision than the Pearson's Correlation approach discussed previously.

3.2.4 Using Association Rules for Web Mining

Fu et.al. in [FBH00] describe a collaborative recommendation system which is similar to the article associations described in [LAR02]. They attempt to mine associations of the form: *[Article 1: like]* $\wedge$ *[Article 2: like] Rightarrow [Article 3: like]*, where the article recommended would be the article in the rule's consequent, namingly *Article 3*. Their approach suffers from the following drawbacks:

- Support is the only method utilized for making recommendations.

- The relevance of the rules is not used to aid in recommendation. In other words, they do not use confidence and support to gauge the quality of the rule.

- Unlike the adaptive support algorithm, the minimum confidence and support must both be specified.

- Only rules with an antecedent of length 2 can be mined on the fly; all other rules must be mined offline.

In the following chapter, we present our framework for utilizing content information to aid recommendation via association rule-based systems. In particular, we present the ideas behind mining content-based associations and content-based collaborative associations.

Chapter 4

Using Content Information for Recommendation

The AR-CRS algorithm previously described can also be used to mine content-based classification rules which can be utilized to make predictions. As mentioned, collaborative-based techniques will fail to produce good quality results when:

1. the target user has not rated many items.
2. the target user has not rated items that other users have rated. In other words, there is little overlap between the ratings of the target user and those users who comprise the collaborative group.

Although user associations can effectively find similarities between multiple users and the target user, the system will still fail to yield quality results if the previous conditions exist. As an alternative, if there is content information, also known as "properties", about the articles present that property information can be used to make predictions.

There are also many advantages to mining content associations over the other content approaches mentioned. These are outlined as follows:

- As with user associations, content associations can effectively capture how multiple properties relate to the target users' preferences, instead of treating each property separately.

- The significance and usefulness of how these properties relate to predicting preferences is also determined more precisely using a rule's support and confidence. This has the advantage of not needing to determine a priori a particular property's significance as required in [BPZ00].

- The significance and usefulness of different properties can be determined on a per-user basis instead of a global basis.

Additionally, the content and collaborative rule-based techniques can be combined to further enhance the quality of recommendation. In particular, we propose mining content-based collaborative associations. Content-based collaborative associations are of the same form as regular collaborative associations. For example, the rule $r =$ *[user 1: like]* $\wedge$ *[user 2: like]* $\Rightarrow$ *[target user: like]* could result from mining either type of association. The difference lies in the data used to construct each type of rule. Regular collaborative associations are constructed based upon user's ratings of articles. Therefore, the rule r would be constructed if *user 1* and *user 2* generally rated articles in the same way as the *target user*. Content-based collaborative associations, on the other hand, are constructed based upon user's ratings of article *properties*. For example, the rule r would be constructed if *user 1* and *user 2* generally liked the same types of article properties, such as color, cost, or material, as the *target user*.

This chapter presents the framework and modifications made in order to produce both content-based recommendations and combined recommendations. In order to mine content-based associations, the data representation must be changed so that each transaction is valid for mining using AR-CRS. A new technique for reducing the size of these transactions so that the mining can be performed more efficiently is also presented. Then, a new scoring method is introduced which makes use of both positive-class and negative-class rules. Finally the framework is given for combining the collaborative and content-based techniques.

4.1 Content-based data representation

Essentially, we want to format the data so that the rules produced have the target user's preference as the consequent and some combination of article properties as the antecedent. Therefore, an example content rule would be: $r = Property1 \wedge Property4 \wedge Property6 \Rightarrow [TargetUser : like]$. In order to mine such rules, each input transaction represents an article *rated by the target user*, and the elements of a transaction represent the properties of that article. We use an open-world assumption, meaning that the fact that certain properties do not appear in transactions has no significant meaning. Therefore, if an article is missing a property from a transaction, a deduction cannot be made that the article does not have that property. For example, if a transaction representing a movie has the following properties: *[actor: Tom Hanks]*, *[genre: comedy*, and *[year: 1997]*, the fact that *[actor: Goldie Hawn]* is not in the transaction will never be used to mine rules. Thus, rules like $r =$ *[year: 1997]* $\wedge$ *[actor != Goldie Hawn]* $\Rightarrow$ *[target user: like]*, where the lack of a property is present in the rule, would never be mined.

One problem with using properties is that properties' values come from a discrete set, whereas the user association data is boolean. For example, a property 'genre' could realize values such as: comedy, romance, cartoon, or horror, not just 'yes/no', or 'like/dislike'. Because the input to AR-CRS is assumed to be of boolean

	genre	*year*	*actor*
article 1	comedy	1998	?
article 2	comedy	1998	T.Hanks
article 3	romance	?	G.Hawn
article 4	comedy	1996	?
article 5	romance	1998	T.Hanks

Becomes

	gen_cmdy	*gen_rom*	*yr_1998*	*yr_1996*	*actr_Hanks*	*actr_Hawn*
art 1	Y	-	Y	-	-	-
art 2	Y	-	Y	-	Y	-
art 3	-	Y	-	-	-	Y
art 4	Y	-	-	Y	-	-
art 5	-	Y	Y	-	Y	-

Figure 4.1: Example dataset for creating transactions for mining content associations.
Here, "Y" means that the property is present, "-" means the property is not present, and "?" means that the article does not have that property information available.

format, each property-value for a property is broken up and treated as an individual property so that this boolean format may be achieved. Therefore, the property 'genre' gets mapped to the following set of properties: genre_comedy, genre_romance, genre_cartoon, and genre_horror. These new properties can then be assigned on a "yes/no" basis depending on if the article has that particular feature or not. An example of this translation is given in Figure 4.1.

In the example, if an article has no value for a property, such as the case with article 1's *actor*, then each of the newly derived properties, in this case *actor_T.Hanks*, and *actor_G.Hawn*, also have no values.

The mapped example dataset can then be used to produce the transactions shown in Figure 4.2.

Those transactions, along with the target user's ratings on the articles can then be used to produce content associations.

transaction #	*itemsets*
1	genre_comedy, year_1998
2	genre_comedy, year_1998, actor_T.Hanks
3	genre_romance, actor_G.Hawn
4	genre_comedy, year_1996
5	genre_romance, year_1998, actor_T.Hanks

Figure 4.2: Transactions generated from the mapped example dataset for content associations.
The presence of a mapped property in a transaction indicates that the associated article has this mapped property.

4.2 Reducing the number of mapped properties

As the number of ratings for articles increases, the number of possible property values, which become properties themselves in the transactions, may also increase dramatically. Because the complexity of the AR-CRS algorithm is exponential with respect to the number of these unique property-values in the worst case, reducing this number speeds up rule mining dramatically.

As a result, a new measurement for selecting the most useful property values was developed. Each property value is associated with a score and those that score highest are used to build up the transactions. The score is derived using all of the articles a user rated, and is based on the information gain formula defined in [Mit97]. The information gain measures how well a property predicts the user's preferences. For example, the property with high information gain would be a property that consistently correlates with the user liking or disliking an article. In this case however, we are using information gain to determine how well a *property value*, such as *actor* = *T.Hanks*, or *actor* = *G.Hawn* correlates with the user liking or disliking an article, versus the property *actor*.

The score for a property value p found in a set of A rated articles, where:

- each article $a \in A$ is either rated as "like", a^+, or "dislike", a^-, by the target user

- p is liked, p^+, by the target user if an article $a \in A$ has the property value p, and the target user rated a as "like"
- p is disliked, p^-, by the target user if an article $a \in A$ has the property value p, and the target user rated a as "dislike"

is given as follows:

$$score_p = \begin{cases} 0, & p = 0 \\ log_2(|p|) * Gain(p, A), & p > 0 \end{cases}$$

where:

- $|p|$ is the number of times p is found in the set of articles A
- $|A|$ is the number of articles rated by the target user
- $|p| <= |A|$
- $Gain(p, A)$ is defined as:

$$Gain(p, A) = Entropy(A) - \frac{|p|}{|A|} Entropy(p)$$

The information gain measures how well a property value is a predictor by calculating the expected reduction in entropy caused by partitioning the articles on this attribute [Mit97]. The entropy thus measures the stability of the input v, meaning that v is totally stable if the target user rated everything in v as "like" or "dislike". The following equation shows the definition of $Entropy(v)$, where v is either p or A

as noted in the previous information gain:

$$Entropy(v) = -1 * (\frac{|v^+|}{|v|} log_2(\frac{|v^+|}{|v|}) + \frac{|v^-|}{|v|} log_2(\frac{|v^-|}{|v|}))$$

where:

- $|v^+|$ is the number of times v was rated as "like"
- $|v^-|$ is the number of times v was rated as "dislike"
- $|v|$ is the size of the input

Essentially, the scoring function presented is a weighted information gain, weighted by the frequency of occurrence of the property value. This modification to the original information gain equation was made because not only do we want such property values with high information gain, but we also want them to be frequent so that they can be mined. However, as the frequency increases, we also want information gain to count more profoundly in choosing the correct property values. The formula described exhibits both properties.

Once each property value has been scored, the top N property values can be chosen to create the transactions necessary to mine content associations.

4.3 New scoring method

Once rules have been mined, a scoring method must be developed in order to produce recommendations. One way to score each article is to use the linear scoring method defined for user associations in [LAR02] as described previously. An alternate method for scoring articles using content rules is to use the scoring function defined in [MLW$^+$00] which makes use of positive and negative class rules. For our purposes, a positive class rule contains *[target user: like]* in the consequent, and a negative class rule contains *[target user: dislike]* in the consequent. In order to acquire both sets of rules, AR-CRS is run twice, once for mining "like" rules, and once for mining "dislike" rules. This method is particularly useful when the user has rated many items as "dislike".

The positive and negative class rule scoring function is a weighted average of the combined significance of the positive and negative class rules. The significance of each rule is determined by that rule's support and confidence values. A rule's support and confidence are used in determining the final score of an article a if a "fires" for that rule. An article a fires for a rule r if the rule's antecedent is satisfied by a. For example, if the rule $r = \textit{comedy} \wedge \textit{T.Hanks} \Rightarrow$ *[target user: dislike]*, article a satisfies r if a contains *comedy* and *T.Hanks*.

One interesting feature about the positive and negative class rule scoring method is that a negative ("dislike") rule can be used to provide insight about a user's "likes". This feature can be seen in the scoring method where a negative class rule is converted to a positive class rule by reversing that rule's confidence. For example, if a negative rule has a confidence of 30%, part of the calculation views that rule as a positive class rule of 70%. Therefore, negative rules with lower confidences in conjunction with positive rules are used to strengthen recommendation capabilities.

Overall, this positive and negative class rule scoring method attempts to provide the following two features:

1. If there are many confident "like" rules that fire for an article, then that article should receive a high score. An article satisfying many confident "like" rules indicates that there is a strong possibility of the user liking the article; thus the article should receive a high score.

2. When “like” rules with low confidences fire for an article, but “dislike” rules with high confidences also fire for that same article, that article should receive a low score.

This scoring function for an article a is calculated as follows:

$$score_a = \frac{\sum\limits_{r \in POS(R,a)} VPOS + \frac{1}{k} \sum\limits_{r \in NEG(R,a)} VNEG}{\sum\limits_{r \in POS(r,a)} supp_r * conf_r + \frac{1}{k} \sum\limits_{r \in NEG(R,a)} supp_r * conf_r}$$

where:

- $conf_r$ is the confidence of rule r
- $supp_r$ is the support of rule r
- R is the set of rules returned by the mining process. In this case, R contains both positive and negative class rules.
- $r \in POS(R,a)$ if the consequent of r is ’like’ and a satisfies r’s antecedent.
- $r \in NEG(R,a)$ if the consequent of r is ’dislike’ and a satisfies r’s antecedent.
- $VPOS = supp_r * conf_r^2$
- $VNEG = supp_r * conf_r * (1 - conf_r)$. This calculation is where a negative class rule gets viewed as a positive class rule as discussed previously.
- k is a user-specified constant which lessens the impact of negative rules. The impact of negative rules on the final score would be great, because we expect more negative rules with stronger confidences and supports to be mined than positive class rules.

4.4 Modifications to the AR-CRS algorithm

The AR-CRS algorithm was modified in three ways. The first two were attempts at lowering the number of iterations required to converge at the minimum support necessary to mine the number of rules within the range for the given minimum confidence. These modifications were made, because reducing the number of iterations increases the overall recommendation speed. The final modification was made to account for mining both "like" and "dislike" rules so that the new scoring method previously mentioned could be used. These modifications are described below:

- *Increasing the minimum support more rapidly when too many rules are mined.* Instead of increasing the minimum support count by one as done in the AR-CRS1 algorithm [LAR02] shown in Figure 2.3, each time too many rules are mined, the minimum support count is increased by two.

- *Changing how the initial minimum support is chosen.* Instead of selecting the minimum support as a function of the number of articles the user rated as "like", the minimum support was determined using the supports of rules of length 1. To calculate the initial minimum support, we find the median support of all the rules of length 1 and then multiply that median support by a frequency percentage. We use this frequency constant, because we expect that the support for larger length rules will be significantly lower than the median support for the rules of length 1. At the same time, however, we want the frequency constant to be large enough to prevent mining too many rules. This process acts as an aid to decrease the number of iterations required to find the optimal support for attaining the correct number of rules above the minimum confidence. In our experiments, we set this frequency constant to .15.

- *Allowing the system to also take the range of negative class rules to mine as input.* When using the new scoring method, an addition was made to provide different ranges of rules to be mined for "like" classification rules and "dislike" classification rules. The minimum confidence, however, was not allowed to vary between the two mining processes.

4.5 Combining Collaborative and Content based recommendations

As stated previously, we propose mining content-based collaborative associations in order to combine collaborative and content-based methods. In this approach, we use users' opinions of article properties as the basis for determining similarity between users, instead of users' opinions of the articles themselves. This approach has the following advantages over mining only content associations or collaborative associations:

- we use all the information available, namingly article properties and user ratings of articles, to make predictions.
- we still capture hard-to-quantify user feelings such as "emotion" just as done with regular collaborative associations.
- we attack the sparsity problem, or the problem where users' ratings of articles do not overlap thus causing problems when recommending via collaborative filtering, by potentially shrinking the space over which user associations are mined. In other words, since there may be less properties than articles over which the user can rate, and since articles may have overlapping properties, a user's rating of one article with a certain set of properties can now be used to

	gen_comedy	*gen_romance*	*actor_T.Hanks*	*actor_G.Hawn*
movie 1	Y	-	Y	-
movie 2	Y	-	-	-
movie 3	Y	-	-	Y
movie 4	-	Y	-	Y
movie 5	-	Y	-	Y
movie 6	Y	-	Y	-
movie 7	-	Y	-	-

	user 1	*user 2*	*user 3*	*user 4*	*user 5*
movie 1	L	-	-	L	-
movie 2	-	L	-	-	-
movie 3	L	-	-	-	L
movie 4	-	L	-	-	L
movie 5	-	-	L	-	-
movie 6	-	-	-	L	-
movie 7	-	-	L	-	-

Figure 4.3: Example database used to construct associations.
In the tables, "Y" means the movie had the corresponding property, and "L" means the user liked the corresponding movie.

derive predictions about other articles with similar properties that the user has not rated. Therefore, since the space over which ratings are present is mapped to a smaller space, there exists a greater chance of users' ratings overlapping.

The third benefit to mining content-based collaborative associations, namingly attacking the sparsity problem, is perhaps the most important aspect of this combination approach. We therefore show an example highlighting this point.

Figure 4.3 shows an example database over which any of the three type of associations, content associations, regular collaborative associations, and content-based collaborative associations, could be mined to eventually yield recommendations.

In particular, the table with *movies* as rows and *users* as columns would be used to mine normal collaborative associations. The problem with mining normal collaborative associations with those data is that the users barely rated anything in common. Therefore, not many associations can be mined. On the other hand,

	user 1	*user 2*	*user 3*	*user 4*	*user 5*
gen_comedy	L	L	-	L	L
gen_romance	-	L	L	-	L
actor_T.Hanks	L	-	-	L	-
actor_G.Hawn	L	L	L	-	L

Table 4.1: Example database used to construct associations. "L" means the user liked the corresponding property.

if we replace those *movies* with *movie properties* we may be able to shrink up the space over which collaborative rules are mined. We can derive users' preferences for properties by looking at the properties of the movies each user rated. For example, *user 1* rated *movie 1* and *movie 3* as like. Each of these movies has properties *genre_comedy* and *actor_T.Hanks.* We can therefore deduce that *user 1* likes comedies and the actor *T.Hanks.* If we repeat this process for all users, we can create a new database over which collaborative associations can be mined. This table is shown in Table 4.1.

As a result of this mapping, users who have rated no articles in common, such as *user 1* and *user 2*, now have overlapping ratings of properties. Because the space over which users ratings exist has shrunk, in this case from 7 movies to 4 properties, more associations can be derived.

Three parts of this process have not yet been fully quantified:

1. Which properties should be used in the mapping?

2. How do we calculate which properties a user likes and dislikes?

3. How do we make recommendations? These aspects are discussed in more detail below.

4.5.1 Choosing the properties for the content-based collaborative approach

In order to select the properties, we use the property scoring method based on the weighted information gain discussed previously. In particular, we view each of the articles every user in the collaborative group has rated, pick the top N scoring properties for each user, and choose those unique properties to represent the transactions over which we will mine content-based collaborative associations. Therefore, each collaborative user's top N attributes are used to construct the dataset.

4.5.2 Determining if a user likes a property

Once we have selected a collaborative user's top N scoring properties, we utilize the user's ratings of articles to discern which properties a user likes or dislikes. Each of the articles a user has rated contains a set of properties. If an article contains one of the properties that is found in the top N list of properties, we note whether or not the user rated that article as "like" or "dislike". If the user liked the article, then we say the user also liked the property for that instance, and vice versa for "dislike". We then tally the number of occurrences of "like" and "dislike" for each property to determine overall if the user liked or disliked the property. The following equation is used to determine if the user likes or dislikes a property p:

$$preference(p) = \begin{cases} like, & \frac{\text{occurrences of liking } p}{\text{total occurrences of } p} > threshold \\ dislike, & otherwise \end{cases}$$

where the threshold is system-specified.

4.5.3 Making recommendations

Once the table has been constructed, content-based collaborative associations such as r = *[user 1: like]* $\wedge$ *[user 2: like]* $\Rightarrow$ *[target user: like]* can be mined. The set of rules obtained can then be scored using the linear scoring method discussed previously in order to generate predictions.

Chapter 5

Experimentation and Evaluation

In this chapter, we provide the results obtained from testing the classification rule-based recommender system which utilizes content information. In the next section, the datasets used for generating the content base and target user ratings, along with the testing process are described. Following the discussion of the datasets, a description of the metrics used to measure the quality of our recommender system is presented. Then, experiments using the linear score threshold are presented. Next, experiments utilizing the new content association scoring method involving positive and negative rules are shown. The results from running the content-based collaborative system are then presented. Finally, all of the results are compared to the results acquired from the collaborative classification rule-based recommender system discussed in [LAR02] and [Lin00].

5.1 The Datasets

This section describes the datasets used for constructing classification rules and for testing the system. Additionally, the processes used to select the training and test data are presented.

5.1.1 Data Construction

Three datasets were used to create the content base for all movies. A majority of the content information was obtained from the KDD Movie Database, an online data source provided by the University of California at Irvine (UCI) [Wie98]. The GroupLens dataset, an online data source provided by the University of Minnesota [Her98] was also added to the content base, because the genre classifications for movies were more detailed than the content information of UCI. Finally, the EachMovie dataset, an online data source provided by the Compaq Systems Research Center [McJ97] was used in the content base, because movie production dates not found in the other datasets were present.

The final content base consisted of the following attributes:

- Year the movie was made
- Director's name
- Director's date of birth
- Director's date of death
- Director's nationality
- Four leading actors' names
- Four leading actors' role types, such as "villian" or "hero"
- Four leading actors' nationalities
- Process code, such as "Technicolor_R" or "black and white"
- Eighteen movie genres, including "horror", "cartoon", and "adventure"

Using this setup, a transaction can contain a variable number of actors' names, actors' roletypes, actors' nationalities, and combination of genres. For example, one transaction in our dataset for the movie "Close Encounters of the Third Kind" would be: *movie 1 = Year:1977, Director:Spielberg, Director_Nationality: USA, Actor: Richard Dreyfuss, Actor: Melinda Dillon, genre: sci-fi.*

All missing values from the original content base were ignored when converting the data to transactions for mining content associations, or building the base for determining user's preferences towards properties for mining content-based collaborative associations. An example of handling missing values for content associations was shown earlier in Figure 4.1.

The Eachmovie dataset [McJ97] was used to acquire ratings for the movies. This dataset was used to compare our results to those obtained in [LAR02]. Each user in the Eachmovie dataset rated movies on a five point scale with 0 being the lowest and 1 being the highest. In all our experiments, just as in [LAR02] we chose .7 as the threshold for a user liking a movie. Therefore, if a user rated a movie as .8 or 1, the user was said to like that movie. Otherwise, they disliked the movie.

5.1.2 Data Selection

In order to select the most relevant content properties for mining, the 100 top scoring properties acquired from the calculation which yields the most relevant properties discussed in Section 3.2 were found on a per-user basis. Therefore, each target user may have a different set of properties over which rules were mined. The 18 movie genres were *not* included in this selection process but were added automatically, because we feel that they are very significant in constructing good content-based rules. Therefore, each transaction can have at most 118 items.

In all experiments, 100 test users were chosen from the EachMovie dataset whose

ID numbers were above 70,000 and who rated at least 100 movies. These users were selected to correspond with those test users used in [LAR02].

In order to construct the collaborative group for the combination approach, the first 1000 users who rated over 100 movies were chosen from the EachMovie dataset as done in the collaborative experiments of [LAR02]. To select the properties which represent the transactions in the property-user matrix for mining, each collaborative group user's top 100 scoring attribute-values were found. Finally, in these experiments a collaborative group user was present in a transaction if out of those transactions containing that property, over 50% were rated as "like" by that user. Again, the movie genres were treated separately and added automatically, resulting in an additional 18 transactions used for mining.

5.1.3 Training and Testing the System

For the two content-based approaches, the first using the linear scoring method and the second using the positive and negative class rule scoring method, four-fold cross validation was used. In order to perform four-fold cross validation, the movies a test user has rated is randomly divided into four sections. Three of the four sections are then used to mine rules, while the fourth section is used to test. This process is repeated four times, each time using a different portion as the testing set, where the testing set contains those movies for which the system will attempt to provide correct recommendations. Therefore, the system is run four different times per target user, and every movie is used as a test movie once. During our experiments using four-fold cross validation, we report the time taken for the four-fold to perform. *Therefore, the time the system would take to yield predictions for a target user if the system was running without four-fold cross validation would be approximately 25% of the times we reported.*

The combination approach, on the other hand, did not use four-fold cross validation. Instead, a "70-30" approach was used, meaning that the first 70% of the ratings were used to construct rules, and the last 30% were used to test. Since four-fold cross validation is not used, the times reported by the system are the actual expected runtimes of the system.

5.2 Metrics used to measure the quality of our recommender system

The quality of the results is analyzed using four metrics:

- *precision*: Precision is defined as the number of test movies the system correctly recommends to the target user out of all the test movies the system recommends. For example, if the system recommends 20 test movies to the target user, and the target user likes 15 of those 20 test movies, then the precision would be 75%.

- *recall*: Recall is the percentage of test movies the system recommends from the target user's "like" test movie list. For example, if the target user likes 10 test movies and the system recommends 8 out of the 10 test movies the user likes, then the recall would be 80%.

- *speed*: Speed is the time elapsed for returning recommendations, including mining time.

- *number of occurrences of no recommendations*: The number of occurrences of no recommendations is a count of the number of target users for which the system does not provide any recommendations. This situation occurs when every test movie does not accrue enough score to surpass the threshold.

5.3 Content Experiments using the Linear Scoring Method

The following experiments use the linear scoring method presented in [LAR02] in which the score threshold is determined by a linear function of the number of rules mined. Each experiment presents the effects of varying each of the parameters of the mining and scoring processes on the final recommendations. For each experiment performed, two graphs are presented. The first, called the "unadjusted" graph, shows the average precision and recall using all results obtained, even those results for which target users did not receive recommendations. In cases where target users did not receive recommendations, both the precision and recall are zero, and thus lower the average precision and recall values presented in the graphs. The second graph, called the "adjusted" graph, shows the average precision and recall using only those people who received recommendations from the system.

The following parameter values were used as the base values for performing the experiments:

- minimum confidence: 30%
- minimum number of rules: 5
- maximum number of rules: 50
- rule length: 6
- score threshold: 0.01*number of rules mined

In each of the following experiments, one of each of the parameters is varied, with the exception of rule length, while keeping the other parameters constant. Rule length was not modified, because (1) rules of larger length are highly infrequent, and (2) if such rules were to be mined, more time would be needed to mine those longer rules. We perform these experiments to test exactly how the quality of the recommender system which uses the linear score threshold to score content associations is affected by each of the parameters.

5.3.1 Experiment 1-1: Varying Confidence

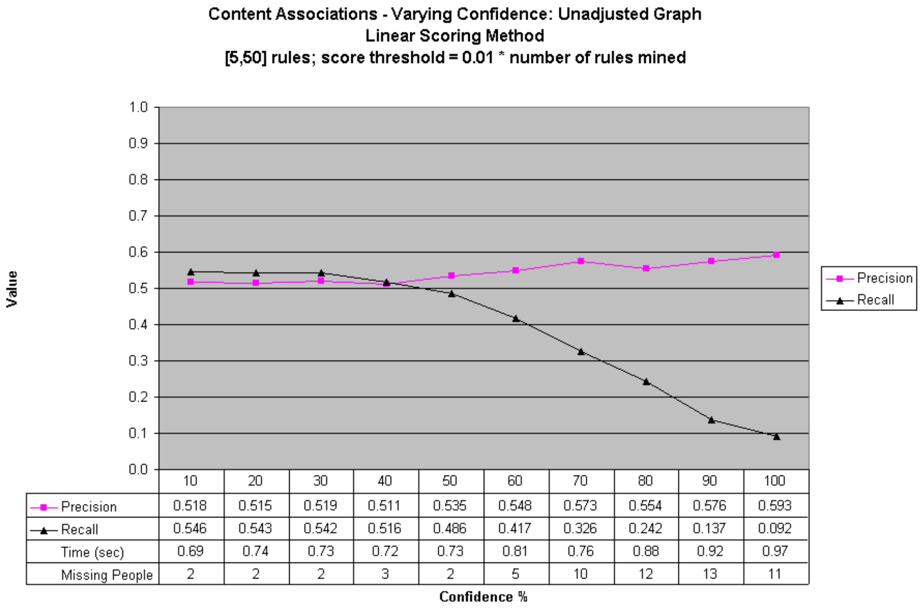
In this experiment, different results were obtained by varying the minimum confidence between 10% and 100% while keeping the other mining parameters constant. The graphs presented in Figure 5.1 show the effects of this modification on the precision and recall.

Both graphs show an increase in precision as the confidence increases. This phenomenon matches expectations, because the confidence measures the strength of rules with respect to the data. Therefore, if the movie satisfies only a few of these strong rules, the result should be a correct prediction.

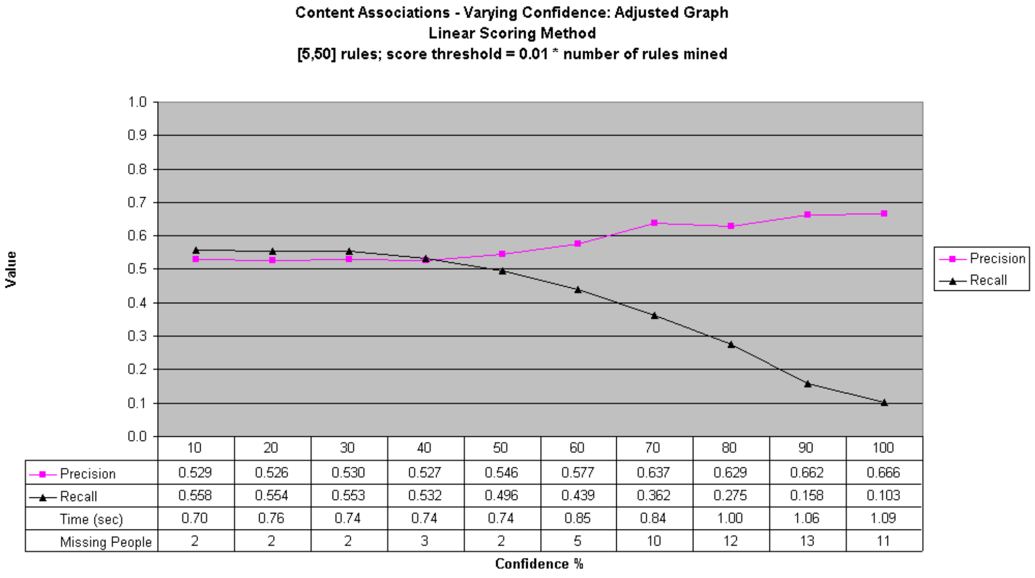
Additionally, the recall decreases sharply as the confidence increases. This result also matches expectations for two reasons. First, if the rules are so highly confident, there is a chance that the support for these rules is very low. Therefore, there is less of a chance of a movie accruing a large enough score to surpass the threshold and thus get recommended. Second, as the confidence increases, there is less of a chance of mining a large number of rules at that confidence. Thus, there is also less of a chance of movies satisfying those rules.

Another observation is that as the minimum confidence increases, the number of people not receiving predictions tends to increase. This suggests that in order to yield predictions for all target users using this scoring method for content associations, one of two options should be performed:

1. The score threshold can be set on a per-user basis so that everyone can receive



	10	20	30	40	50	60	70	80	90	100
Precision	0.518	0.515	0.519	0.511	0.535	0.548	0.573	0.554	0.576	0.593
Recall	0.546	0.543	0.542	0.516	0.486	0.417	0.326	0.242	0.137	0.092
Time (sec)	0.69	0.74	0.73	0.72	0.73	0.81	0.76	0.88	0.92	0.97
Missing People	2	2	2	3	2	5	10	12	13	11



	10	20	30	40	50	60	70	80	90	100
Precision	0.529	0.526	0.530	0.527	0.546	0.577	0.637	0.629	0.662	0.666
Recall	0.558	0.554	0.553	0.532	0.496	0.439	0.362	0.275	0.158	0.103
Time (sec)	0.70	0.76	0.74	0.74	0.74	0.85	0.84	1.00	1.06	1.09
Missing People	2	2	2	3	2	5	10	12	13	11

Figure 5.1: Content associations linear scoring method results from varying the minimum confidence. Time reported is for all four folds. Actual response time is approximately 25% of the times reported.

predictions.

2. When no recommendations are made, instead of using a threshold to recommend items, the top-N scoring movies should be returned to the target user. An experiment which returns the top-3 articles is presented in section 5-7.

In trials where the minimum confidence was high, the precision was high in comparison with the recall. This observation means that the system is not recommending many items, but the recommendations made are precise. Therefore, as the confidence increases, there is a tradeoff between attaining more precise results and having the system recommending more movies, which may increase the recall.

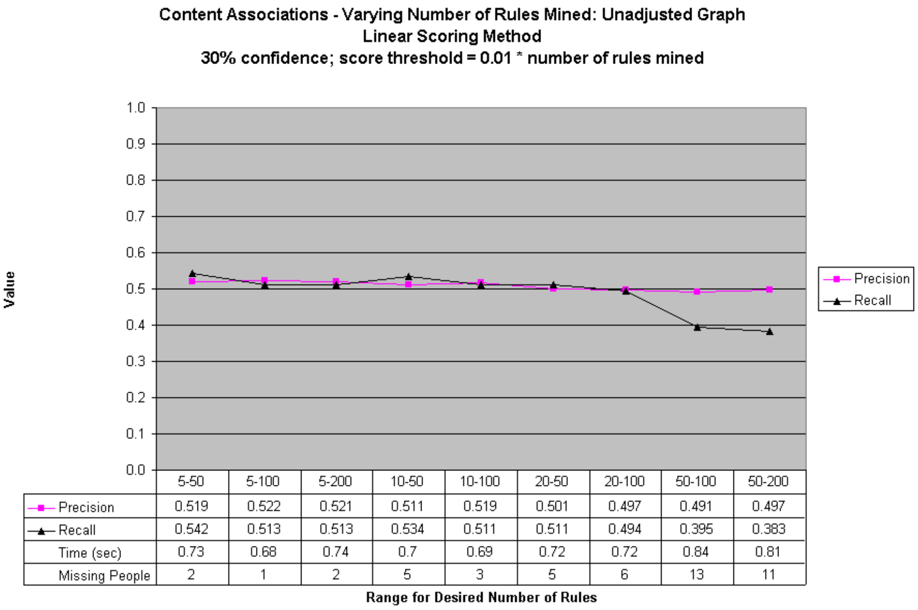
5.3.2 Experiment 1-2: Varying the Number of Rules Mined

In these experiments, the minimum and maximum number of rules mined was varied for 30% minimum confidence and then for 90% minimum confidence. We chose to run the additional experiment at 90% minimum confidence, because the precision yielded from that confidence as shown in Figure 5.1 was high. We therefore wanted to view the effects of modifying the rule range on a low confidence and high confidence. The results from this experiment are shown in Figure 5.2 and Figure 5.3.

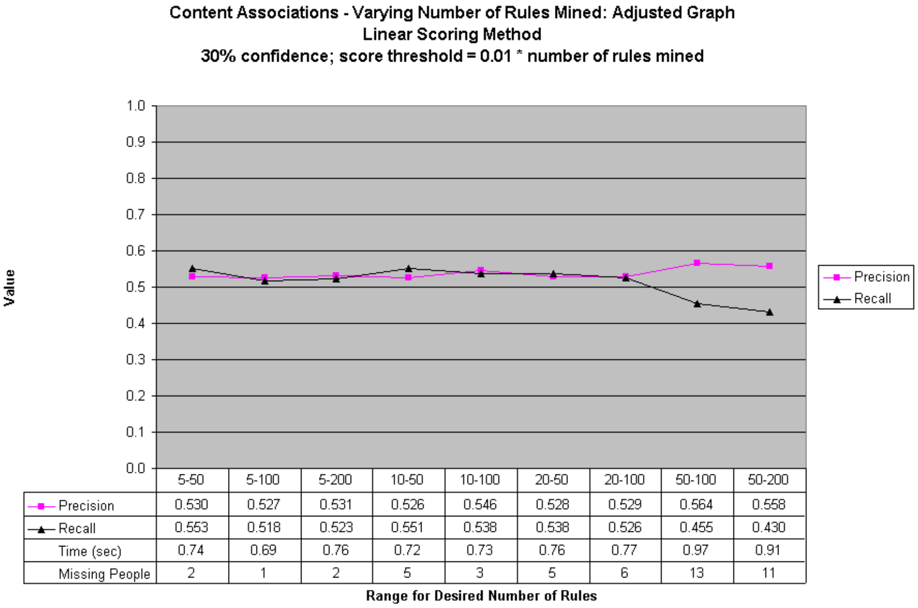
Both the recall and precision drop as the minimum number of rules increases for the unadjusted graphs of both 30% minimum confidence and 90% confidence. However, if the people who receive no predictions are ignored as shown in the adjusted graphs, then as the minimum number of rules increases, the precision increases and the recall drops.

A precision increase occurred as the minimum number of rules increased, more noticeably at the 90% confidence level, for the following reasons:

1. At the 90% confidence level, rules which show a high correlation between the

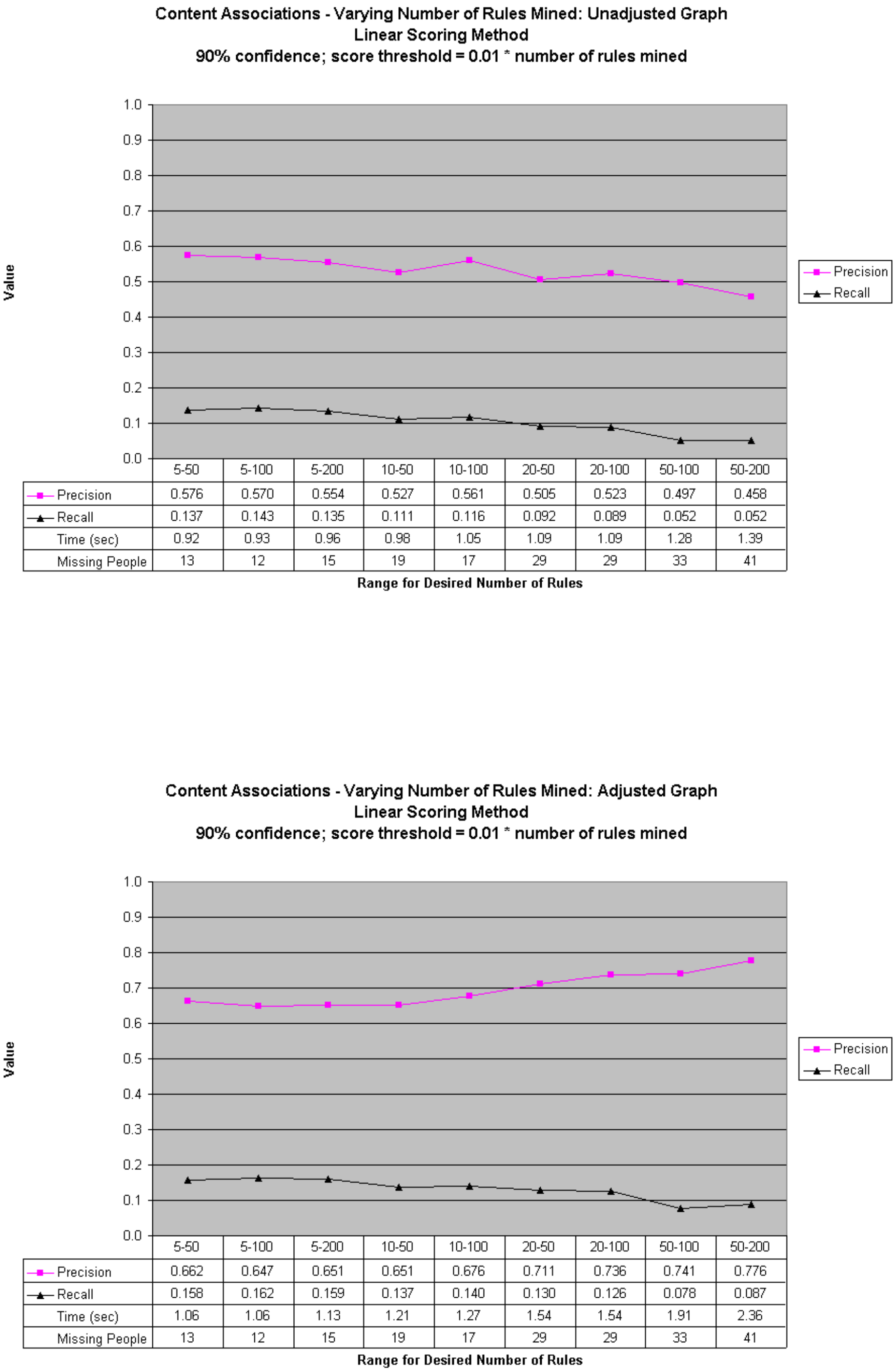


	5-50	5-100	5-200	10-50	10-100	20-50	20-100	50-100	50-200
Precision	0.519	0.522	0.521	0.511	0.519	0.501	0.497	0.491	0.497
Recall	0.542	0.513	0.513	0.534	0.511	0.511	0.494	0.395	0.383
Time (sec)	0.73	0.68	0.74	0.7	0.69	0.72	0.72	0.84	0.81
Missing People	2	1	2	5	3	5	6	13	11



	5-50	5-100	5-200	10-50	10-100	20-50	20-100	50-100	50-200
Precision	0.530	0.527	0.531	0.526	0.546	0.528	0.529	0.564	0.558
Recall	0.553	0.518	0.523	0.551	0.538	0.538	0.526	0.455	0.430
Time (sec)	0.74	0.69	0.76	0.72	0.73	0.76	0.77	0.97	0.91
Missing People	2	1	2	5	3	5	6	13	11

Figure 5.2: Content associations linear scoring method results from varying the range for the number of rules, mined at a minimum confidence of 30%.


Content Associations - Varying Number of Rules Mined: Unadjusted Graph
Linear Scoring Method
90% confidence; score threshold = 0.01 * number of rules mined

	5-50	5-100	5-200	10-50	10-100	20-50	20-100	50-100	50-200
Precision	0.576	0.570	0.554	0.527	0.561	0.505	0.523	0.497	0.458
Recall	0.137	0.143	0.135	0.111	0.116	0.092	0.089	0.052	0.052
Time (sec)	0.92	0.93	0.96	0.98	1.05	1.09	1.09	1.28	1.39
Missing People	13	12	15	19	17	29	29	33	41

Content Associations - Varying Number of Rules Mined: Adjusted Graph
Linear Scoring Method
90% confidence; score threshold = 0.01 * number of rules mined

	5-50	5-100	5-200	10-50	10-100	20-50	20-100	50-100	50-200
Precision	0.662	0.647	0.651	0.651	0.676	0.711	0.736	0.741	0.776
Recall	0.158	0.162	0.159	0.137	0.140	0.130	0.126	0.078	0.087
Time (sec)	1.06	1.06	1.13	1.21	1.27	1.54	1.54	1.91	2.36
Missing People	13	12	15	19	17	29	29	33	41

Figure 5.3: Content associations linear scoring method results from varying the range for the number of rules, mined at a minimum confidence of 90%. Time reported is for all four folds. Actual response time is approximately 25% of the times reported.

movie properties and user preference are mined. If the correlations are strong, then we expect that if enough of those rules are satisfied by the movie, then the recommendation of that movie should be correct, thus increasing precision.

2. At the 30% confidence level, many rules can be found, because the confidence is so low. Furthermore, because the confidence is low, a good chance exists that the supports of these rules can be high. If many of these rules are satisfied, then a movie could accrue enough score to be recommended. We expect these recommendations to be correct, because in order for a movie to be recommended, the score would have to be great enough to exceed the linear threshold based on the number of rules mined. The score required for the system to recommend would be high, because many rules would be mined. Therefore, assuming our scoring method is correct in that high scoring movies are those which the target user would like, if many of these low confidence rules are satisfied by a movie, we expect that the system recommends correctly, thus increasing the precision. Later in Section 5.7, we specifically test the scoring method's ability to determine correct predictions by seeing if the highest scoring articles are in fact correct predictions.

The drop in recall is evident, because as the minimum number of rules increases, the support must be lowered to accommodate for the confidence. This drop in support affects the final scores of the movies, because now more rules would need to be satisfied in order for a movie to be recommended, because the score threshold is based on the number of rules mined. Since the threshold is larger due to mining more rules, a good chance exists that any movie cannot accrue enough score to be recommended, thus increasing the number of people not receiving predictions.

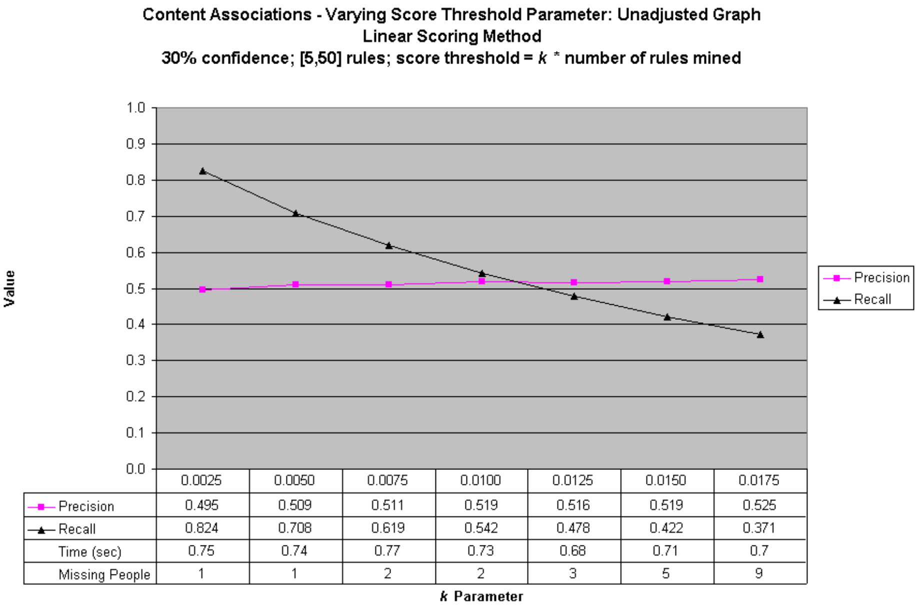
A key result is found when 50 to 200 rules are mined for a minimum confidence of 90%. As shown in the adjusted graph, the highest precision is yielded at this point. This suggests that although less people will receive predictions from the system, those predictions will be precise when using a minimum confidence of 90% and mining with a large minimum number of rules.

5.3.3 Experiment 1-3: Varying the Score Threshold Parameter

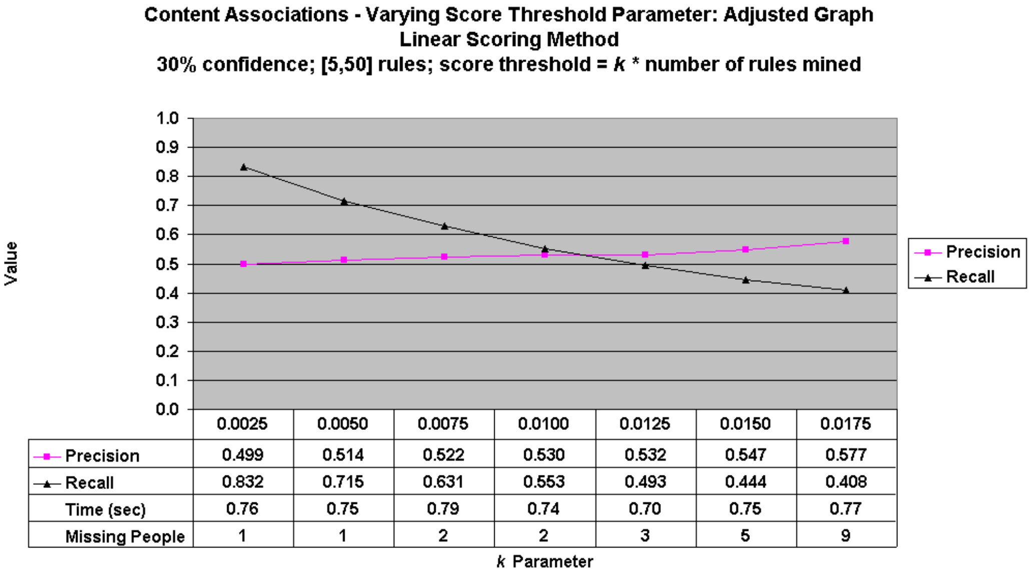
Finally, the score threshold parameter was modified using values ranging from 0.0025 to 0.0175. The score threshold was varied using a minimum confidence of 30% and then a minimum confidence of 90%. Again, the additional experiment at 90% was run, because we wanted to see if we could further increase the high precision attained at this confidence as shown in Figure 5.1. The results from this experiment are shown in Figure 5.4 and Figure 5.5.

For a minimum confidence of 30%, both the adjusted and unadjusted graphs show an increase in precision and a decrease in recall as the score threshold increases. For a minimum confidence of 90%, the adjusted graph shows an increase in precision and a decrease in recall as the score threshold increases; however the same is not true for the unadjusted version. The unadjusted precision decreases as the score threshold increases, because the number of people without predictions is larger.

These observations confirm expectations for two reasons. First, as the score threshold increases, the ability to yield predictions should become more difficult, because each article must acquire a high score to be recommended. Since many articles may not be able to achieve such a high score, less articles will be recommended, which explains both why the recall decreases, and why more people do not receive predictions as the score threshold increases. Additionally if the score threshold is high, then those movies exceeding the threshold should be correctly

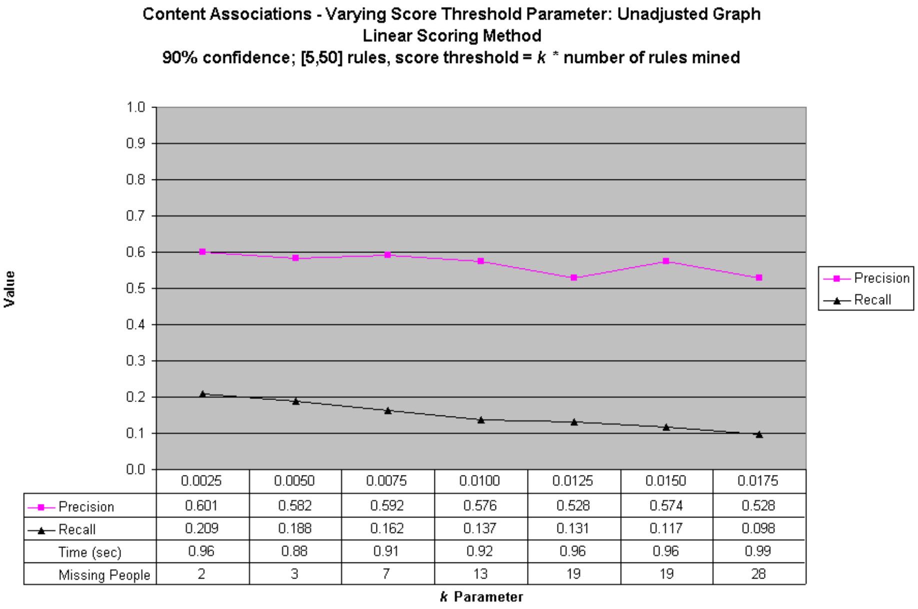


	0.0025	0.0050	0.0075	0.0100	0.0125	0.0150	0.0175
Precision	0.495	0.509	0.511	0.519	0.516	0.519	0.525
Recall	0.824	0.708	0.619	0.542	0.478	0.422	0.371
Time (sec)	0.75	0.74	0.77	0.73	0.68	0.71	0.7
Missing People	1	1	2	2	3	5	9

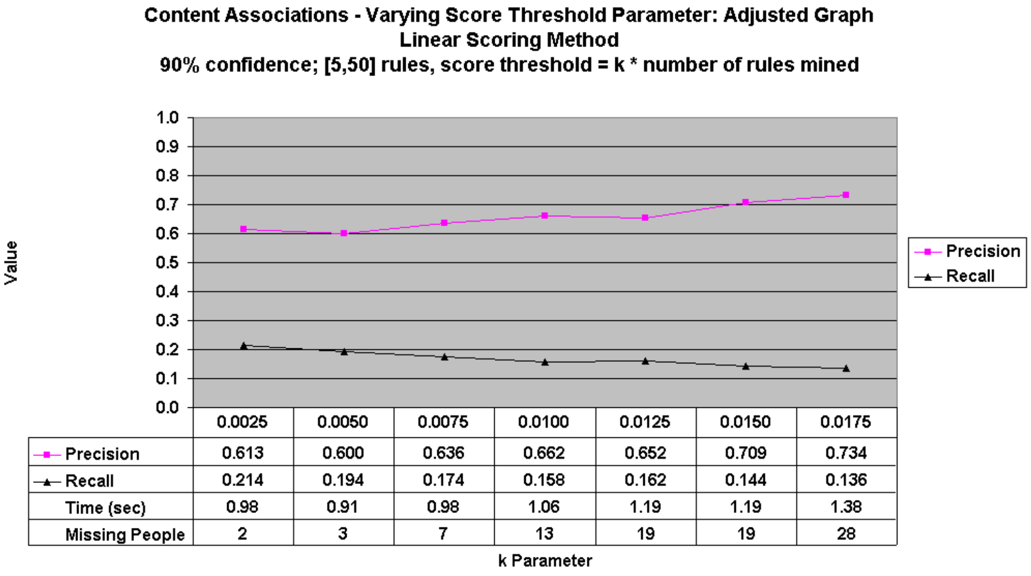


	0.0025	0.0050	0.0075	0.0100	0.0125	0.0150	0.0175
Precision	0.499	0.514	0.522	0.530	0.532	0.547	0.577
Recall	0.832	0.715	0.631	0.553	0.493	0.444	0.408
Time (sec)	0.76	0.75	0.79	0.74	0.70	0.75	0.77
Missing People	1	1	2	2	3	5	9

Figure 5.4: Content associations linear scoring method results from varying the score threshold parameter, mined at a minimum confidence of 30%.


Content Associations - Varying Score Threshold Parameter: Unadjusted Graph
Linear Scoring Method
90% confidence; [5,50] rules, score threshold = *k* * number of rules mined

	0.0025	0.0050	0.0075	0.0100	0.0125	0.0150	0.0175
Precision	0.601	0.582	0.592	0.576	0.528	0.574	0.528
Recall	0.209	0.188	0.162	0.137	0.131	0.117	0.098
Time (sec)	0.96	0.88	0.91	0.92	0.96	0.96	0.99
Missing People	2	3	7	13	19	19	28


Content Associations - Varying Score Threshold Parameter: Adjusted Graph
Linear Scoring Method
90% confidence; [5,50] rules, score threshold = k * number of rules mined

	0.0025	0.0050	0.0075	0.0100	0.0125	0.0150	0.0175
Precision	0.613	0.600	0.636	0.662	0.652	0.709	0.734
Recall	0.214	0.194	0.174	0.158	0.162	0.144	0.136
Time (sec)	0.98	0.91	0.98	1.06	1.19	1.19	1.38
Missing People	2	3	7	13	19	19	28

Figure 5.5: Content associations linear scoring method results from varying the score threshold parameter, mined at a minimum confidence of 90%. Time reported is for all four folds. Actual response time is approximately 25% of the times reported.

recommended providing the scoring function works correctly, thus explaining the increasing precision.

Another important discovery from this experiment is that higher confidences yield higher precisions and lower recalls irrespective of the score threshold. Therefore, if we keep the precision and score threshold high, we will receive few, but precise recommendations.

5.3.4 Summary of Linear Scoring Method Experiments

We summarize the effects each parameter has on precision, recall, and the number of people not receiving predictions below:

1. Increasing the minimum confidence increases the precision, decreases the recall, and increases the number of people not receiving predictions.

2. Increasing the minimum number of rules to mine increases the precision, decreases the recall, and increases the number of people not receiving predictions.

3. Increasing the score threshold parameter increases the precision, decreases the recall, and increases the number of people not receiving predictions.

If we (1) set the minimum confidence to be 90% or 100%, (2) set the rule range to be between 50 and 200, and (3) set the score threshold to be 0.0175, we should receive the most precise results. However, at these high mining parameter values, as backed up by the previous experiments a good chance exists that many people will not receive predictions. Since the score threshold parameter seems to most greatly affect the number of people receiving predictions, we feel that lowering the score parameter may be important even though we sacrifice some precision. In order to test these hypotheses, in particular to see how much precision will be negatively

confidence	*rule range*	*threshold param*	*precision*	*recall*	*time (sec)*	*missing people*
90%	[50,200]	0.05	0.670	0.117	1.75	21
100%	[50,200]	0.05	0.731	0.075	1.89	21
90%	[50,200]	0.175	0.811	0.066	3.78	63
100%	[50,200]	0.175	0.844	0.029	4.42	67

Table 5.1: Summary of adjusted results obtained when mining with the best parameters for content associations with the linear score method. Time reported is for all four folds. Actual response time is approximately 25% of the times reported.

affected if we lower the score threshold parameter, we ran an additional experiment using these high mining values. The results, not counting those who did not receive predictions, are summarized in Table 5.1.

Table 5.1 shows that if the lower score threshold parameter is used with 100% confidence and 50 to 200 rules, the precision only drops by 8%, while 42 more people receive predictions from the system. Therefore, we can get precise results and have many people receive recommendations from the system. To further back this claim, we would like to see if many target users receive these high quality predictions. In order to do this, we created four distributions to see the number of high quality recommendations returned by the system for all target users using the mining parameters mentioned in Table 5.1. These results, which ignore those people who did not receive predictions, are shown in Figure 5.6 and Figure 5.7.

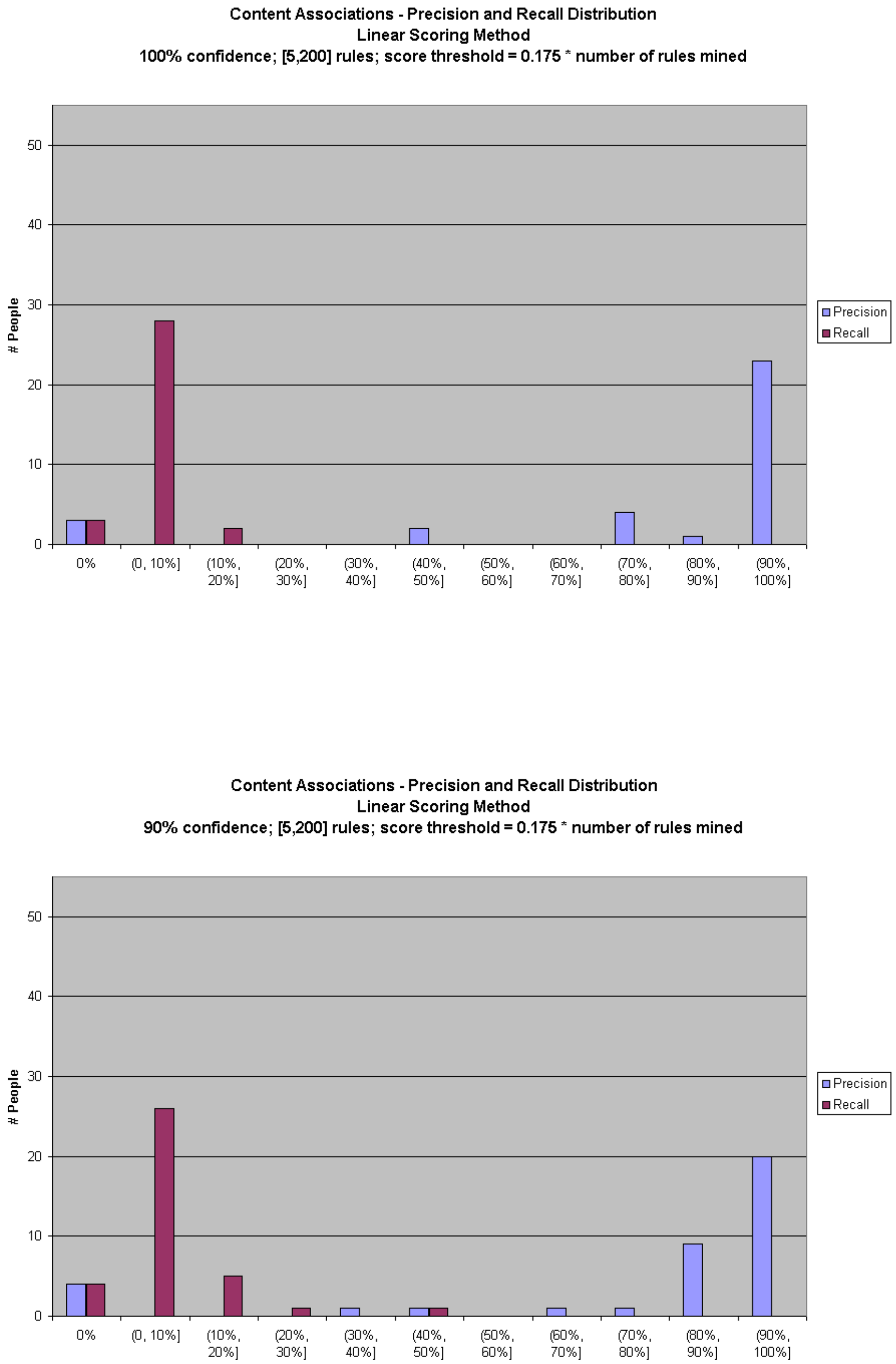


Figure 5.6: Distributions of results for high-valued mining parameters with score threshold parameter = 0.175

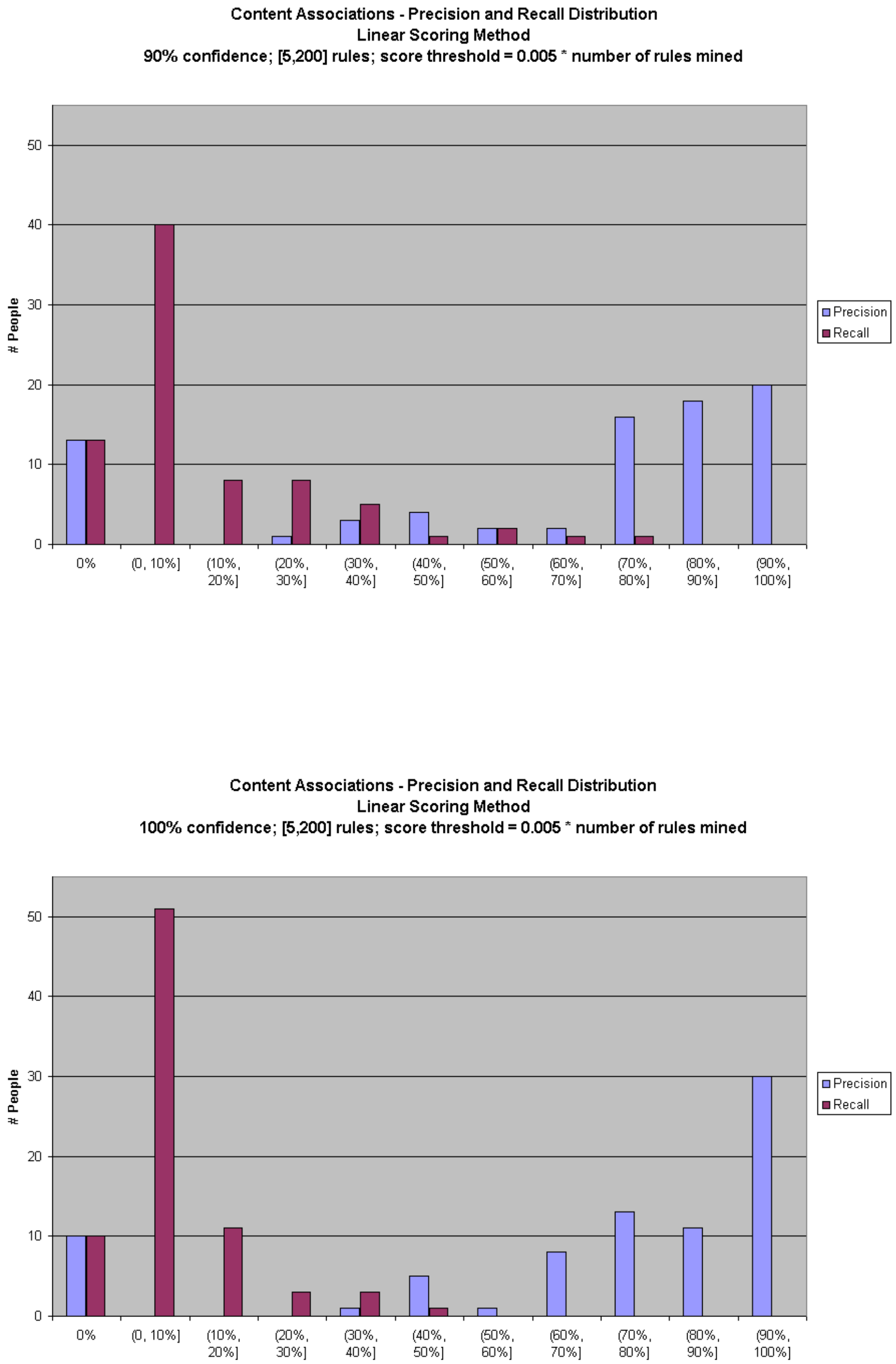
The distributions yield some interesting observations:

- The number of people for which completely incorrect results, where the precision and recall are zero, is higher for a lower score threshold. To see this, we calculate the percentage of people who received completely wrong predictions out of all those who received predictions. With a score threshold of 0.005, 16.5% of the people received completely wrong predictions at 90% confidence and 12.7% received completely wrong predictions at 100% confidence. With a score threshold of 0.175, 10.8% of the people received completely wrong predictions at 90% confidence, and 9.1% received a zero precision at 100% confidence.

- The number of people receiving higher precisions, precisions above 0.8, is higher for the score threshold of 0.175. To see this, we view the percentage of people receiving high precisions out of all the people who received predictions. With a score threshold of 0.005, 48.1% of the people attained precisions of over 0.8 at a confidence of 90%, and 51.9% of the people attained precisions over 0.8 at a confidence of 100%. For a score threshold of 0.175, 78.4% of the people attained this high precision at a confidence of 90%, and 72.7% of the people attained this high precision at a confidence of 100%.

- In all four distributions, most people who were given recommendations received a low recall. In particular, for all distributions, the 0.0 and 0.1 recall group accrued the most number of people.

These distributions, along with the information given in Table 5.1 show that content associations scored with the linear method can indeed yield precise results. In particular, using the correct parameters can yield high precisions as shown in the distributions. Although the precision is high, we still want the system to be


Content Associations - Precision and Recall Distribution
Linear Scoring Method
90% confidence; [5,200] rules; score threshold = 0.005 * number of rules mined

Content Associations - Precision and Recall Distribution
Linear Scoring Method
100% confidence; [5,200] rules; score threshold = 0.005 * number of rules mined

Figure 5.7: Distribution of results for high-valued mining parameters with score threshold parameter = 0.005

able to recommend for everyone, and also not have the system return completely incorrect results. We therefore need an alternate method for producing results for those who did not receive predictions or received completely wrong predictions. In the next section, we see if the alternate scoring method will increase the precision and recall, and decrease the number of people not receiving predictions and the number of people receiving wrong recommendations.

5.4 Content Experiments using Positive and Negative Rules

The following experiments use the scoring method presented in [MLW$^+$00] in which the score threshold is determined using both positive rules, those which represent the target user liking a movie, and negative rules, those which represent the target user disliking a movie as explained in Section 4.3. Each experiment presents the effects of varying each of the parameters of the mining and scoring processes on the final recommendations. The following parameter values were used as the base values for performing the experiments:

- minimum confidence for mining both positive and negative rules: 30%
- minimum number of positive rules: 5
- maximum number of positive rules: 50
- minimum number of negative rules: 5
- maximum number of negative rules: 50
- rule length: 6

- score threshold: 0.5
- negative rule weighting constant: 3

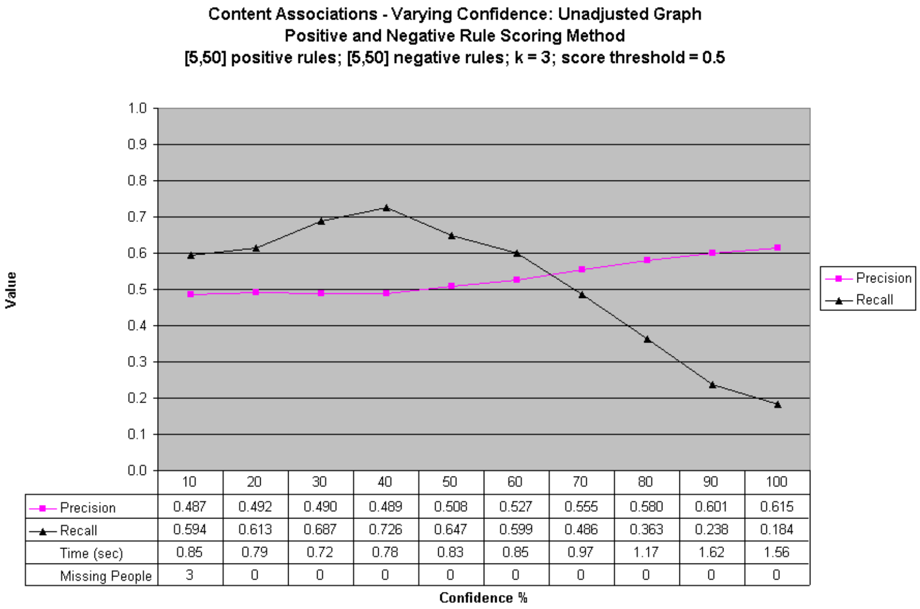
As in the previous set of experiments, we do not modify the rule length parameter. We run these experiments to find the best parameters over which to mine in order to derive a distribution. Using this distribution we will compare the quality of the results returned by this positive and negative rule scoring method with those results yielded from the linear scoring method presented in the previous section.

5.4.1 Experiment 2-1: Varying Confidence for the New Scoring Method

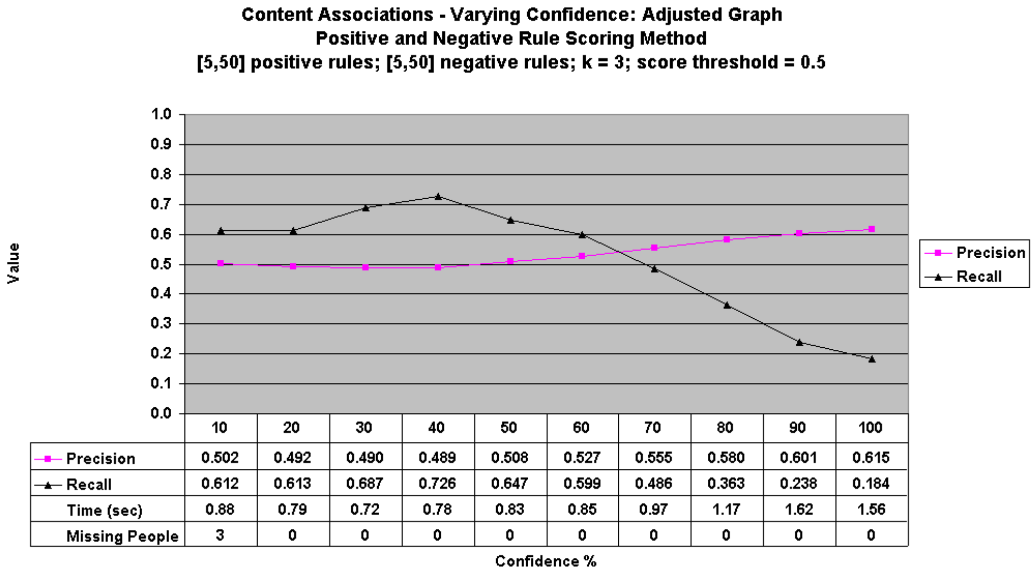
This experiment presents the effect that confidence has on precision and recall for the new scoring method. Results were obtained by varying the minimum confidence between 10% and 100%. Figure 5.8 shows the effects of this parameter modification:

As shown in Figure 5.8, as the confidence increases beyond 40%, the precision increases while the recall decreases. This is the same trend observed previously for the linear scoring method when modifying the minimum confidence. However, there are three key differences to note between the two scoring methods when varying confidence:

1. More people received predictions using the positive and negative rule scoring method over the linear method discussed previously when modifying the minimum confidence parameter. As shown in Figure 5.8, everyone received predictions from the new scoring method for all confidence values, except 10%, where only three people did not receive predictions. In contrast, as shown in Figure 5.1 *every* confidence value for the linear scoring method yielded people who did not receive predictions. 13 people did not receive predictions at 100%



	10	20	30	40	50	60	70	80	90	100
Precision	0.487	0.492	0.490	0.489	0.508	0.527	0.555	0.580	0.601	0.615
Recall	0.594	0.613	0.687	0.726	0.647	0.599	0.486	0.363	0.238	0.184
Time (sec)	0.85	0.79	0.72	0.78	0.83	0.85	0.97	1.17	1.62	1.56
Missing People	3	0	0	0	0	0	0	0	0	0



	10	20	30	40	50	60	70	80	90	100
Precision	0.502	0.492	0.490	0.489	0.508	0.527	0.555	0.580	0.601	0.615
Recall	0.612	0.613	0.687	0.726	0.647	0.599	0.486	0.363	0.238	0.184
Time (sec)	0.88	0.79	0.72	0.78	0.83	0.85	0.97	1.17	1.62	1.56
Missing People	3	0	0	0	0	0	0	0	0	0

Figure 5.8: Content associations positive and negative rule scoring method results from varying the minimum confidence parameter. Time reported is for all four folds. Actual response time is approximately 25% of the times reported.

minimum confidence for the linear method, which is significantly larger than the positive and negative rule scoring method.

2. The precision for the new scoring method is higher for a confidence above 80% than the unadjusted confidence of the linear method. However, the precision for the new scoring method is lower for all confidence values than the unadjusted graph for the linear scoring method. Therefore, overall the new scoring method yields less precise results as compared to only those people who receive predictions via the linear method.

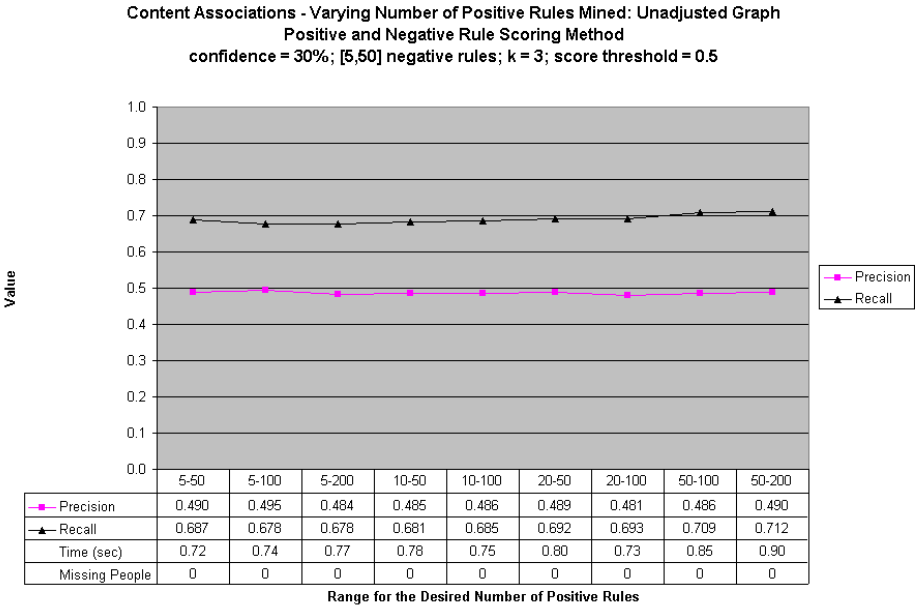
3. The new scoring method yields a consistently higher recall as compared to the linear scoring approach when varying the confidence.

5.4.2 Experiment 2-2: Varying the Number of Positive and Negative Rules Mined

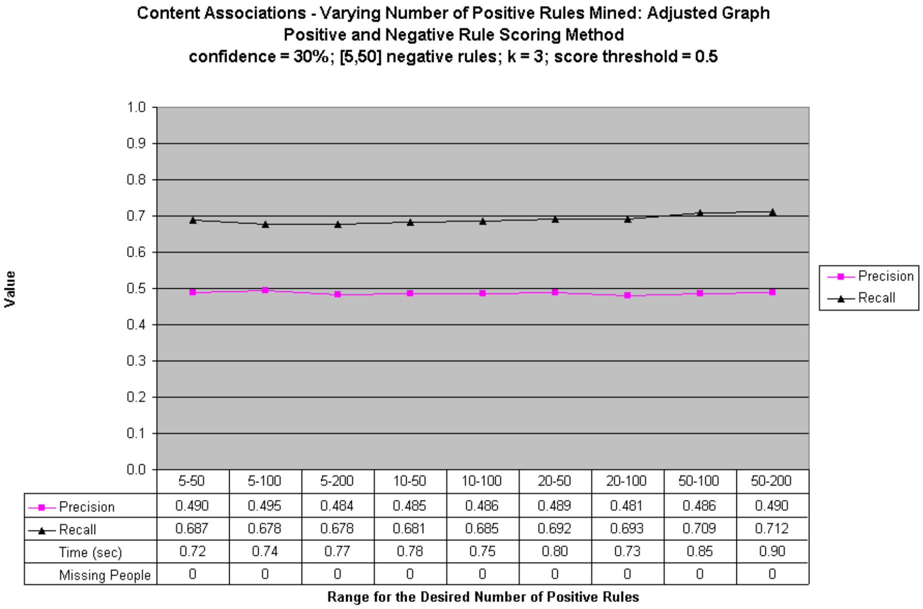
In these experiments the minimum and maximum number of positive and negative rules were varied independently. Both the positive and negative rule numbers were modified with a minimum confidence of 30% and minimum confidence of 90%. A minimum confidence of 30% and 90% were chosen to compare our results with the previous results acquired by scoring rules using the linear scoring method. Figure 5.9 and Figure 5.10 shows the results from modifying the rule range for positive rules; Figure 5.11 and Figure 5.12 shows the results from modifying the rule range for negative rules.

Varying the number of positive and negative rules does *not* greatly affect precision as shown in the graphs. There are some other key observations acquired from these graphs:

1. The recall tends to fluctuate more heavily than the precision when varying

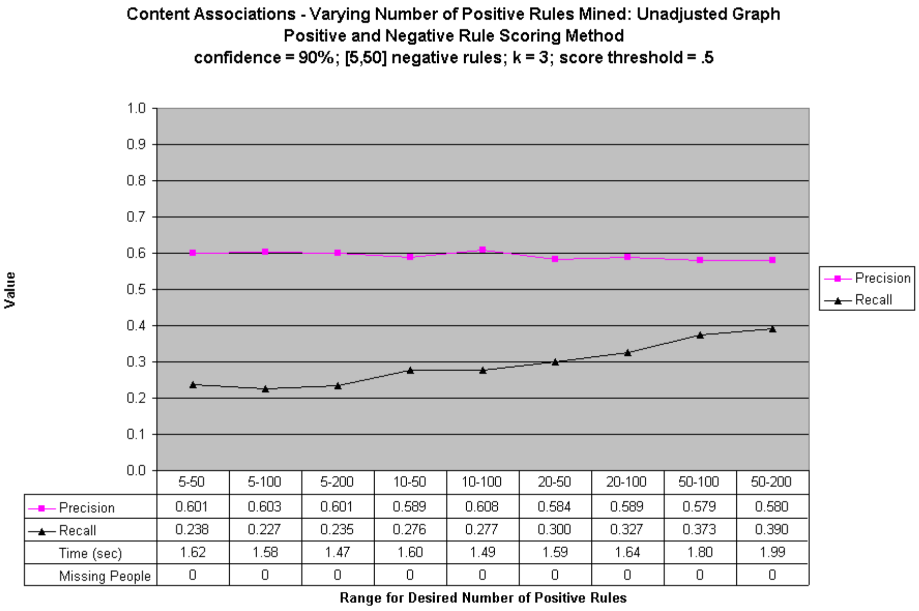


	5-50	5-100	5-200	10-50	10-100	20-50	20-100	50-100	50-200
Precision	0.490	0.495	0.484	0.485	0.486	0.489	0.481	0.486	0.490
Recall	0.687	0.678	0.678	0.681	0.685	0.692	0.693	0.709	0.712
Time (sec)	0.72	0.74	0.77	0.78	0.75	0.80	0.73	0.85	0.90
Missing People	0	0	0	0	0	0	0	0	0



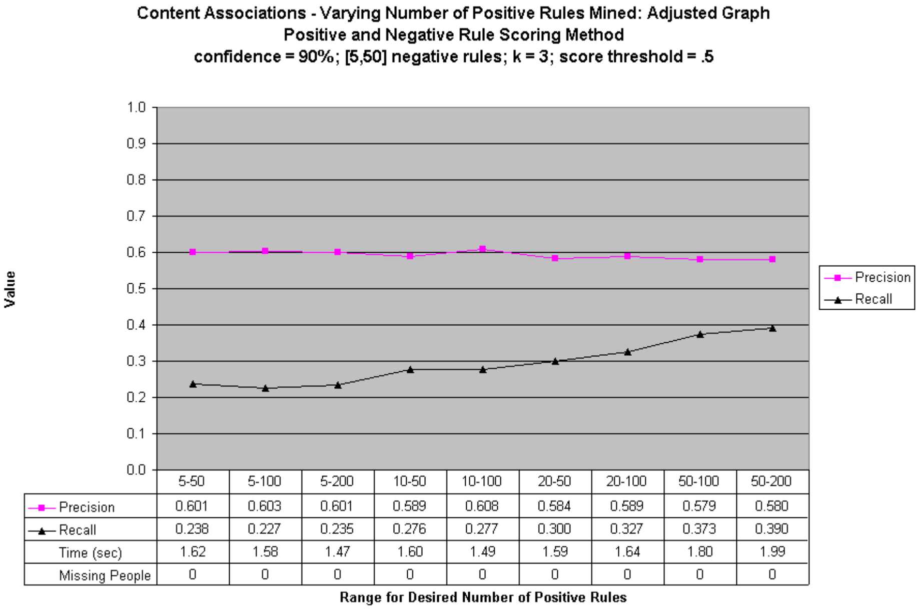
	5-50	5-100	5-200	10-50	10-100	20-50	20-100	50-100	50-200
Precision	0.490	0.495	0.484	0.485	0.486	0.489	0.481	0.486	0.490
Recall	0.687	0.678	0.678	0.681	0.685	0.692	0.693	0.709	0.712
Time (sec)	0.72	0.74	0.77	0.78	0.75	0.80	0.73	0.85	0.90
Missing People	0	0	0	0	0	0	0	0	0

Figure 5.9: Content associations positive and negative rule scoring method results from varying the range for the number of positive rules, mined at a minimum confidence of 30%.



Content Associations - Varying Number of Positive Rules Mined: Unadjusted Graph
Positive and Negative Rule Scoring Method
confidence = 90%; [5,50] negative rules; k = 3; score threshold = .5

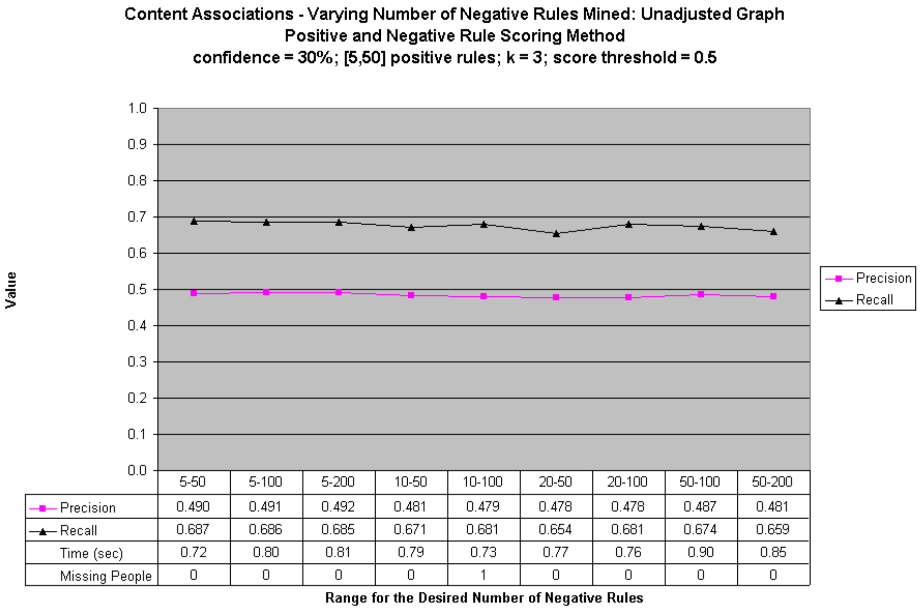
	5-50	5-100	5-200	10-50	10-100	20-50	20-100	50-100	50-200
Precision	0.601	0.603	0.601	0.589	0.608	0.584	0.589	0.579	0.580
Recall	0.238	0.227	0.235	0.276	0.277	0.300	0.327	0.373	0.390
Time (sec)	1.62	1.58	1.47	1.60	1.49	1.59	1.64	1.80	1.99
Missing People	0	0	0	0	0	0	0	0	0



Content Associations - Varying Number of Positive Rules Mined: Adjusted Graph
Positive and Negative Rule Scoring Method
confidence = 90%; [5,50] negative rules; k = 3; score threshold = .5

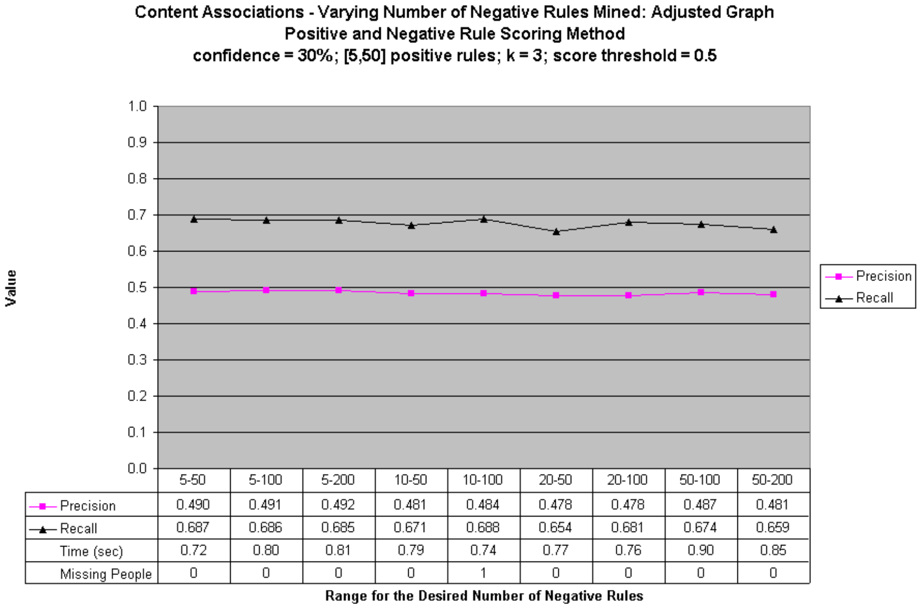
	5-50	5-100	5-200	10-50	10-100	20-50	20-100	50-100	50-200
Precision	0.601	0.603	0.601	0.589	0.608	0.584	0.589	0.579	0.580
Recall	0.238	0.227	0.235	0.276	0.277	0.300	0.327	0.373	0.390
Time (sec)	1.62	1.58	1.47	1.60	1.49	1.59	1.64	1.80	1.99
Missing People	0	0	0	0	0	0	0	0	0

Figure 5.10: Content associations positive and negative rule scoring method results from varying the range for the number of positive rules, mined at a minimum confidence of 90%. Time reported is for all four folds. Actual response time is approximately 25% of the times reported.



Content Associations - Varying Number of Negative Rules Mined: Unadjusted Graph
Positive and Negative Rule Scoring Method
confidence = 30%; [5,50] positive rules; k = 3; score threshold = 0.5

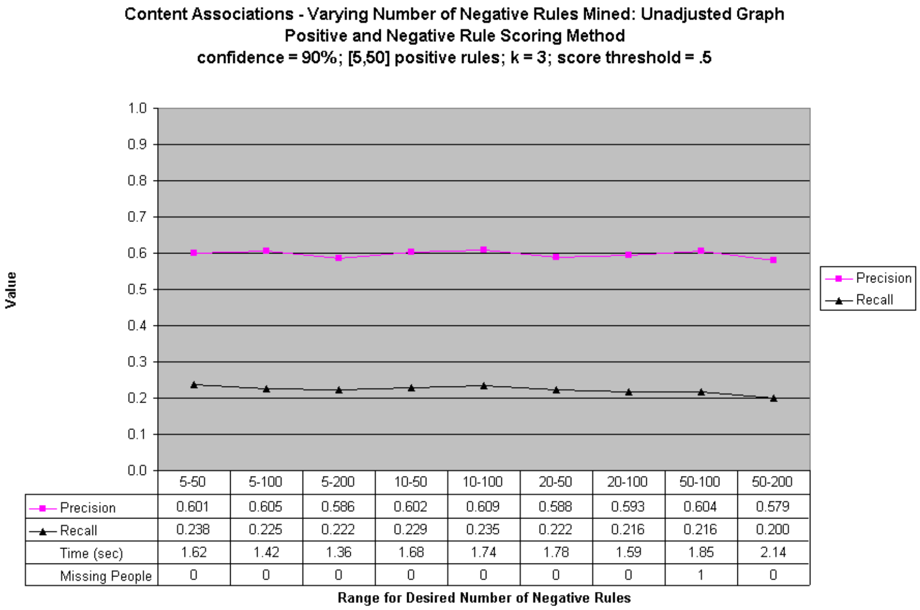
	5-50	5-100	5-200	10-50	10-100	20-50	20-100	50-100	50-200
Precision	0.490	0.491	0.492	0.481	0.479	0.478	0.478	0.487	0.481
Recall	0.687	0.686	0.685	0.671	0.681	0.654	0.681	0.674	0.659
Time (sec)	0.72	0.80	0.81	0.79	0.73	0.77	0.76	0.90	0.85
Missing People	0	0	0	0	1	0	0	0	0



Content Associations - Varying Number of Negative Rules Mined: Adjusted Graph
Positive and Negative Rule Scoring Method
confidence = 30%; [5,50] positive rules; k = 3; score threshold = 0.5

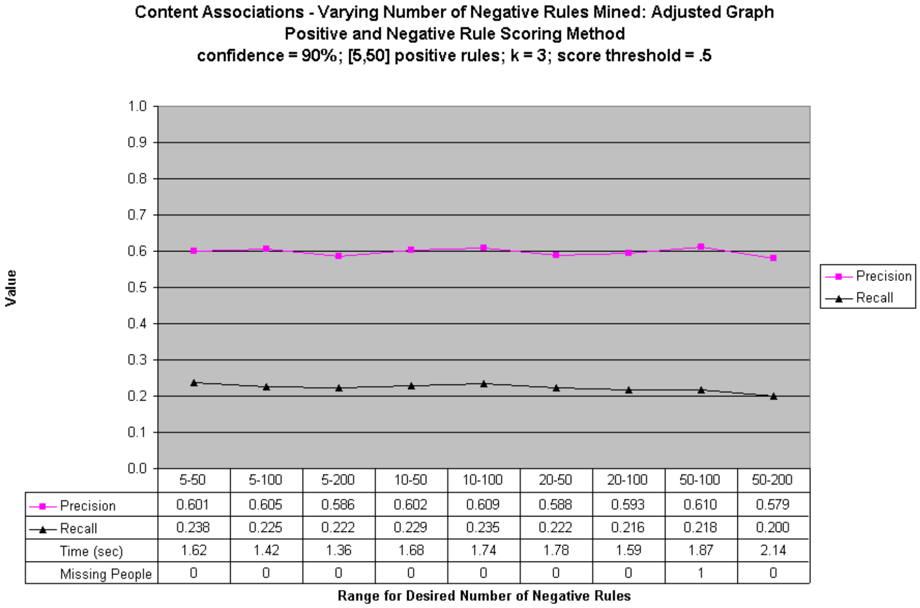
	5-50	5-100	5-200	10-50	10-100	20-50	20-100	50-100	50-200
Precision	0.490	0.491	0.492	0.481	0.484	0.478	0.478	0.487	0.481
Recall	0.687	0.686	0.685	0.671	0.688	0.654	0.681	0.674	0.659
Time (sec)	0.72	0.80	0.81	0.79	0.74	0.77	0.76	0.90	0.85
Missing People	0	0	0	0	1	0	0	0	0

Figure 5.11: Content associations positive and negative rule scoring method results from varying the range for the number of negative rules, mined at a minimum confidence of 30%. Time reported is for all four folds. Actual response time is approximately 25% of the times reported.



Content Associations - Varying Number of Negative Rules Mined: Unadjusted Graph
Positive and Negative Rule Scoring Method
confidence = 90%; [5,50] positive rules; k = 3; score threshold = .5

	5-50	5-100	5-200	10-50	10-100	20-50	20-100	50-100	50-200
Precision	0.601	0.605	0.586	0.602	0.609	0.588	0.593	0.604	0.579
Recall	0.238	0.225	0.222	0.229	0.235	0.222	0.216	0.216	0.200
Time (sec)	1.62	1.42	1.36	1.68	1.74	1.78	1.59	1.85	2.14
Missing People	0	0	0	0	0	0	0	1	0



Content Associations - Varying Number of Negative Rules Mined: Adjusted Graph
Positive and Negative Rule Scoring Method
confidence = 90%; [5,50] positive rules; k = 3; score threshold = .5

	5-50	5-100	5-200	10-50	10-100	20-50	20-100	50-100	50-200
Precision	0.601	0.605	0.586	0.602	0.609	0.588	0.593	0.610	0.579
Recall	0.238	0.225	0.222	0.229	0.235	0.222	0.216	0.218	0.200
Time (sec)	1.62	1.42	1.36	1.68	1.74	1.78	1.59	1.87	2.14
Missing People	0	0	0	0	0	0	0	1	0

Figure 5.12: Content associations positive and negative rule scoring method results from varying the range for the number of negative rules, mined at a minimum confidence of 90%. Time reported is for all four folds. Actual response time is approximately 25% of the times reported.

the negative rule range for both confidence levels. However this fluctuation is more pronounced for a minimum confidence of 90%. The best precision and recall combination for 30% confidence is found when mining between 5 and 50 negative rules or 50 and 100 negative rules. The best precision and recall combination for 90% confidence is found when mining between 10 and 100 negative rules.

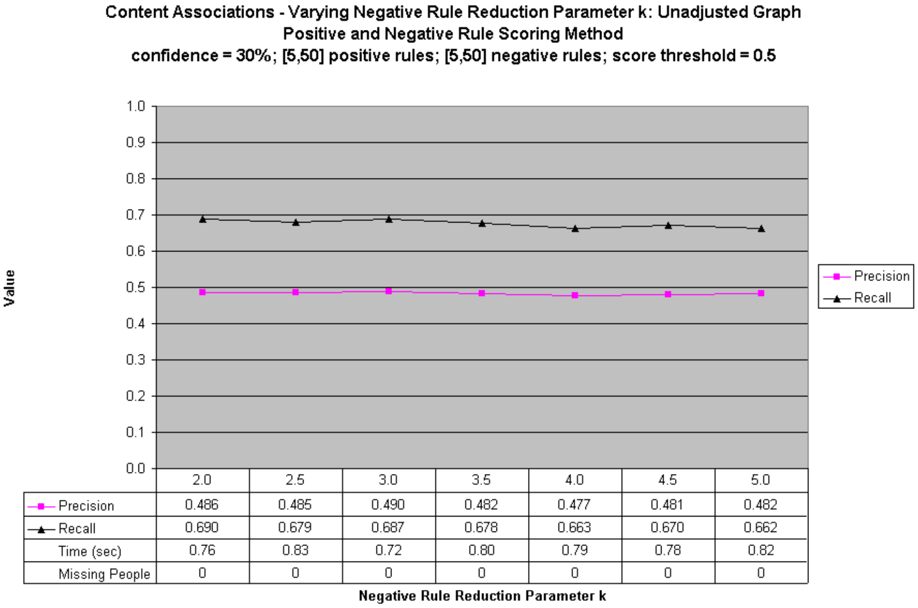
2. As the minimum number of positive rules increases, the recall increases while the precision decreases slightly. This phenomenon, seen in both graphs, is more visible for a minimum confidence of 90%. Additionally, these great increases in recall only slightly diminish the precision.

3. Varying the rule range for either class of rules does little affecting the number of people who do not receive recommendations. In Experiment 2-1 in which the minimum confidence was varied, everyone received predictions for a minimum confidence of 30% and 90%. At both these minimum confidences, when the positive and negative rule ranges were modified, only in two trials did people not receive predictions. Specifically, only 1 person did not receive predictions for a positive rule range of [10,100] at a minimum confidence of 90%, and 1 person did not receive predictions for a negative rule range of [50,100] at a minimum confidence of 90%.

5.4.3 Experiment 2-3: Varying the negative rule reduction constant

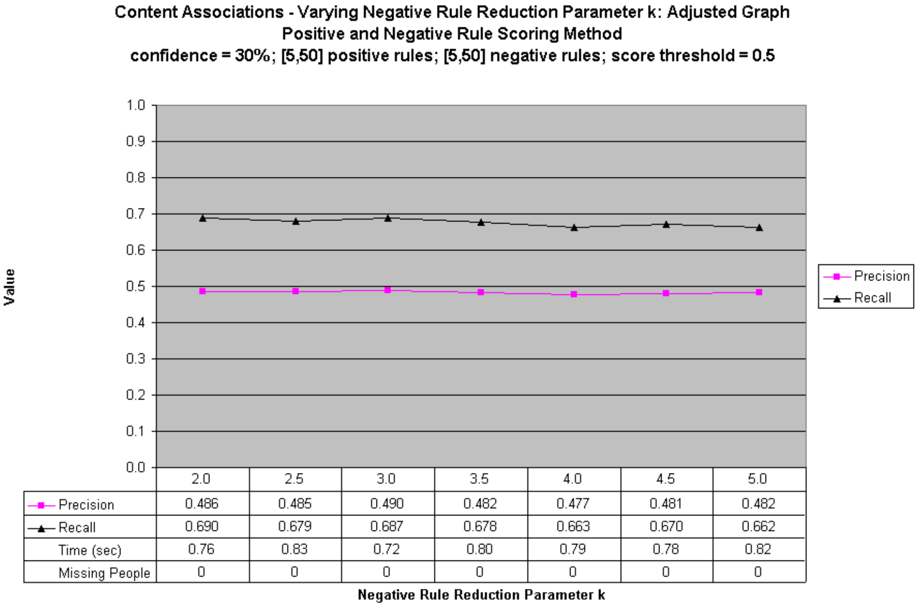
These results show how varying the negative rule reduction constant affects precision and recall. The constant was assigned values between 2.0 and 5.0 using first a minimum confidence of 30%, and then 90%. The following graphs in Figure 5.13

and Figure 5.14 depicts these results:



Content Associations - Varying Negative Rule Reduction Parameter k: Unadjusted Graph
Positive and Negative Rule Scoring Method
confidence = 30%; [5,50] positive rules; [5,50] negative rules; score threshold = 0.5

	2.0	2.5	3.0	3.5	4.0	4.5	5.0
Precision	0.486	0.485	0.490	0.482	0.477	0.481	0.482
Recall	0.690	0.679	0.687	0.678	0.663	0.670	0.662
Time (sec)	0.76	0.83	0.72	0.80	0.79	0.78	0.82
Missing People	0	0	0	0	0	0	0



Content Associations - Varying Negative Rule Reduction Parameter k: Adjusted Graph
Positive and Negative Rule Scoring Method
confidence = 30%; [5,50] positive rules; [5,50] negative rules; score threshold = 0.5

	2.0	2.5	3.0	3.5	4.0	4.5	5.0
Precision	0.486	0.485	0.490	0.482	0.477	0.481	0.482
Recall	0.690	0.679	0.687	0.678	0.663	0.670	0.662
Time (sec)	0.76	0.83	0.72	0.80	0.79	0.78	0.82
Missing People	0	0	0	0	0	0	0

Figure 5.13: Content associations positive and negative rule scoring method results from varying the negative rule reduction parameter at 30% minimum confidence. Time reported is for all four folds. Actual response time is approximately 25% of the times reported.

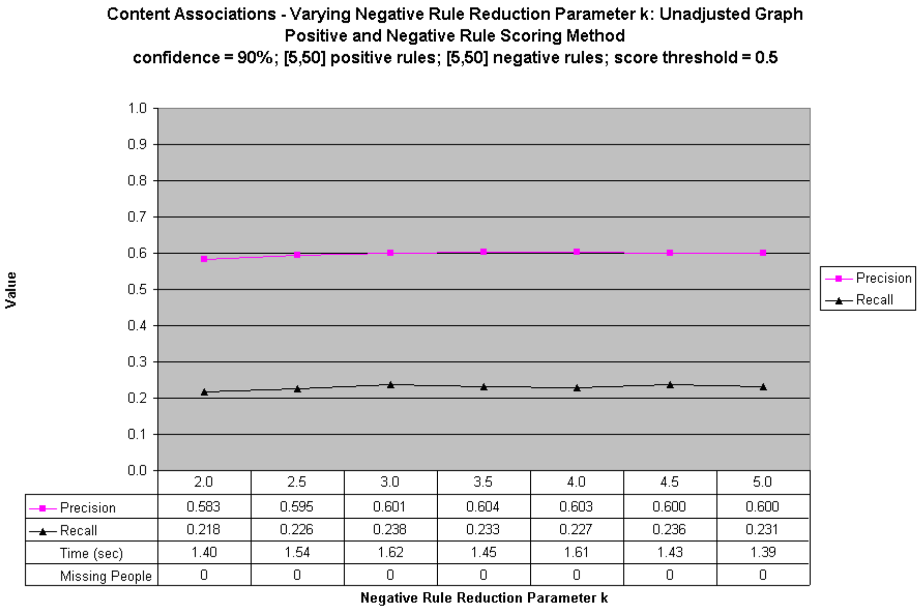
As shown in Figure 5.13 and Figure 5.14, this rule reduction constant barely affected the precision and recall. Additionally, modification to this constant did not alter the number of people not receiving predictions. In fact, everyone received predictions in this setup, which is why no adjusted graphs are presented. The best precision and recall combination found in both experiments was obtained for a negative reduction constant of 3.0, which coincides with the findings from [MLW$^+$00].

5.4.4 Experiment 2-4: Varying the score threshold

In this experiment, the score threshold was varied between 0.3 and 0.8 for 4 different minimum confidence values: 30%, 50%, 70%, and 90%. We chose to run experiments with more minimum confidences, because the minimum confidence along with the score threshold appear to be the most significant factors affecting precision and recall. Furthermore, we wish to get a better feel of how the scoring method behaves in terms of precision, recall, and the number of people not receiving predictions at different confidence levels. Figure 5.15 shows the results for a minimum confidence of 30%, Figure 5.16 shows the results for a minimum confidence of 50%, Figure 5.17 shows the results for a minimum confidence of 70%, and Figure 5.18 shows the results for a minimum confidence of 90%.

Many interesting observations can be made from these experiments:

1. When the score threshold is high and the minimum confidence is low, few people receive recommendations from the system. For a score threshold of 0.80 and a minimum confidence of 30%, only 26 people receive predictions. However those people who receive recommendations however receive the most precise recommendations on average as compared to any of the experiments run thus far. These precise results may be attributed to the fact that we mine negative rules at a low confidence. Because the positive and negative rule scoring



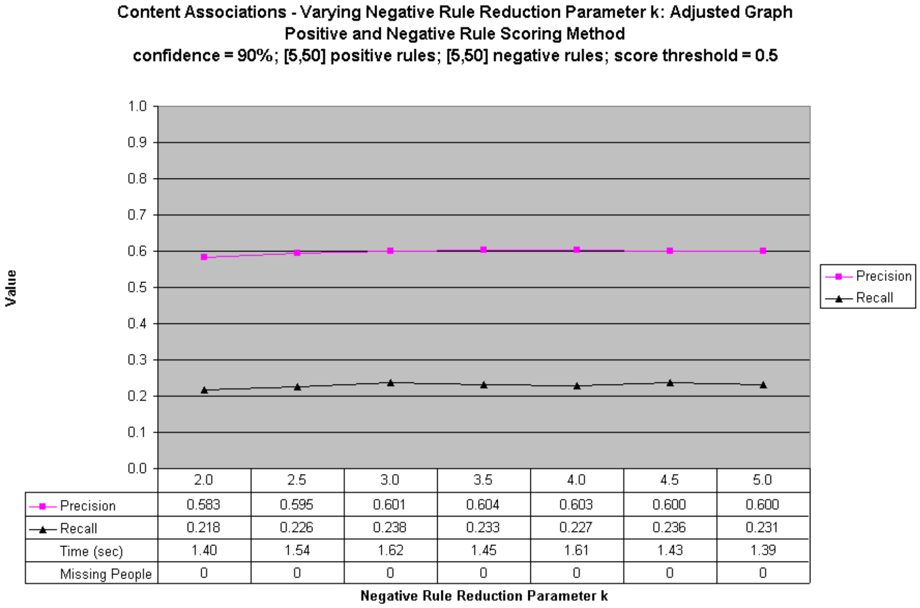
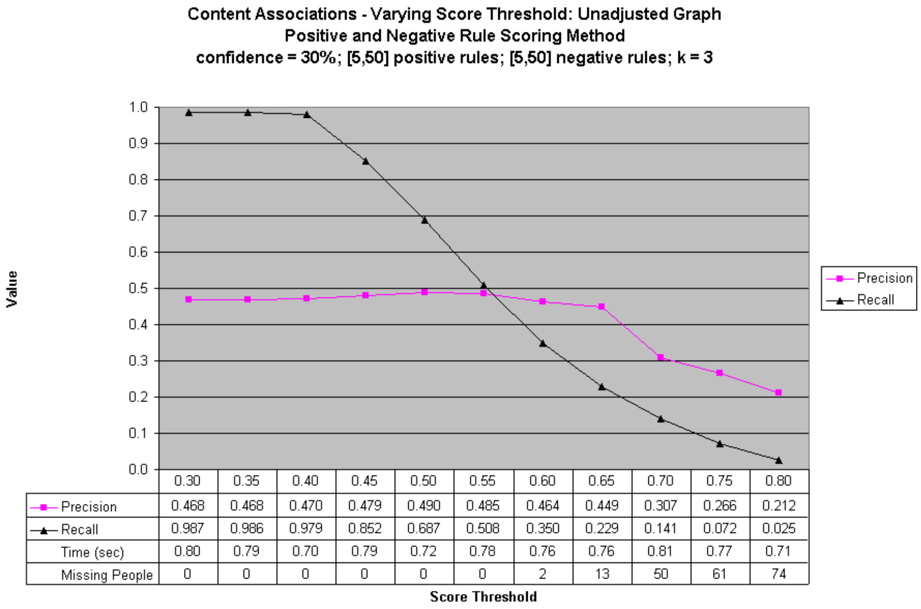
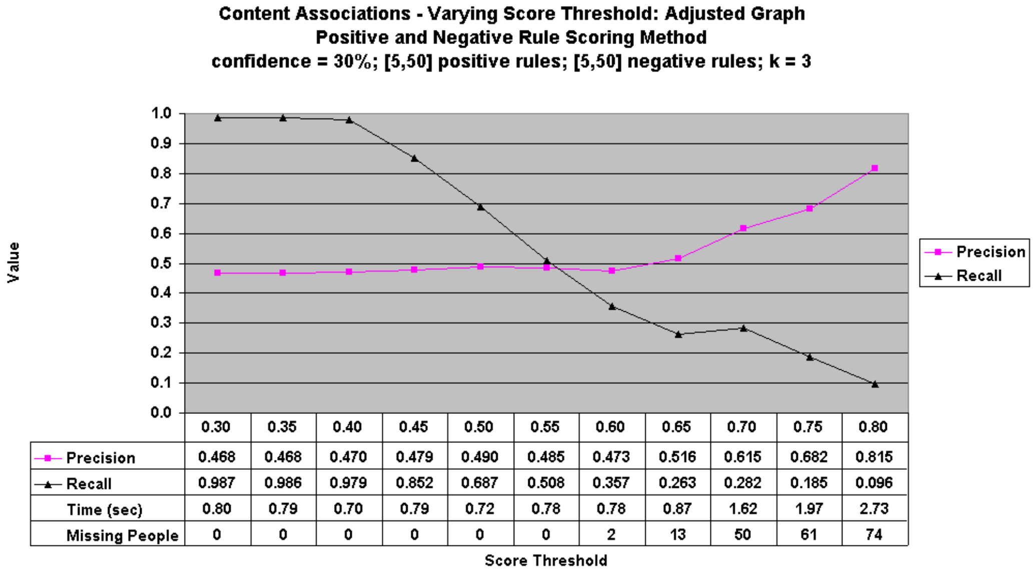


Figure 5.14: Content associations positive and negative rule scoring method results from varying the negative rule reduction parameter at 90% minimum confidence. Time reported is for all four folds. Actual response time is approximately 25% of the times reported.

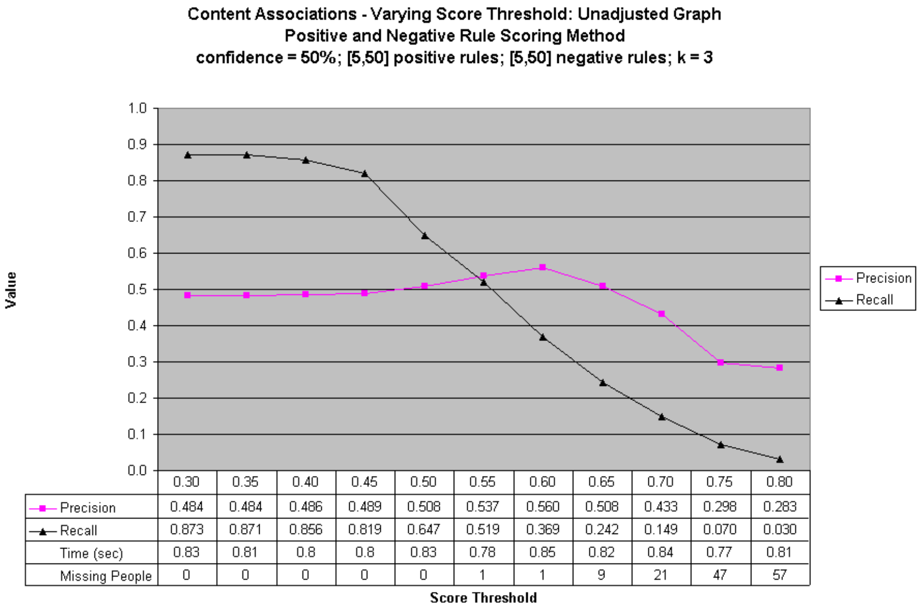


	0.30	0.35	0.40	0.45	0.50	0.55	0.60	0.65	0.70	0.75	0.80
Precision	0.468	0.468	0.470	0.479	0.490	0.485	0.464	0.449	0.307	0.266	0.212
Recall	0.987	0.986	0.979	0.852	0.687	0.508	0.350	0.229	0.141	0.072	0.025
Time (sec)	0.80	0.79	0.70	0.79	0.72	0.78	0.76	0.76	0.81	0.77	0.71
Missing People	0	0	0	0	0	0	2	13	50	61	74

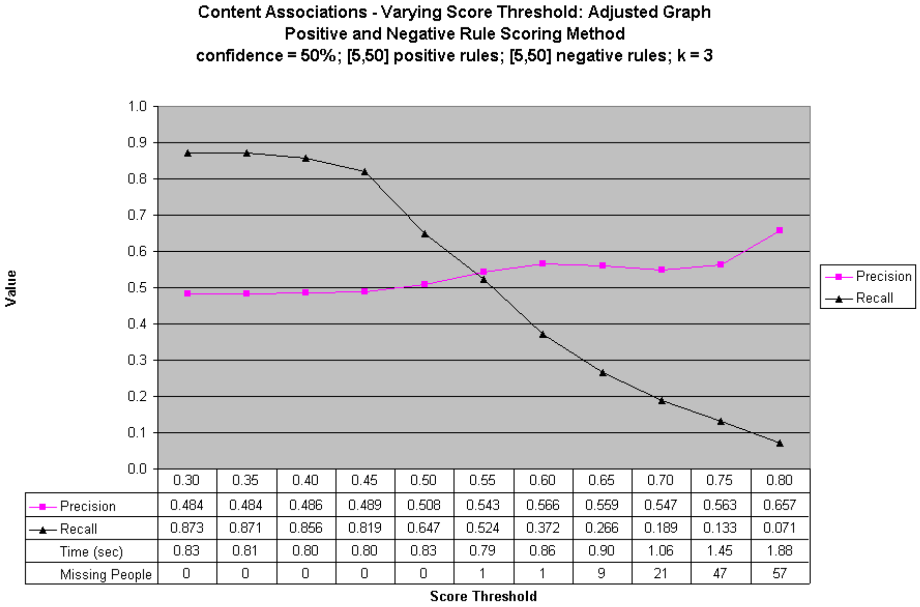


	0.30	0.35	0.40	0.45	0.50	0.55	0.60	0.65	0.70	0.75	0.80
Precision	0.468	0.468	0.470	0.479	0.490	0.485	0.473	0.516	0.615	0.682	0.815
Recall	0.987	0.986	0.979	0.852	0.687	0.508	0.357	0.263	0.282	0.185	0.096
Time (sec)	0.80	0.79	0.70	0.79	0.72	0.78	0.78	0.87	1.62	1.97	2.73
Missing People	0	0	0	0	0	0	2	13	50	61	74

Figure 5.15: Content associations positive and negative rule scoring method results from varying the score threshold, mined at a minimum confidence of 30%. Time reported is for all four folds. Actual response time is approximately 25% of the times reported.

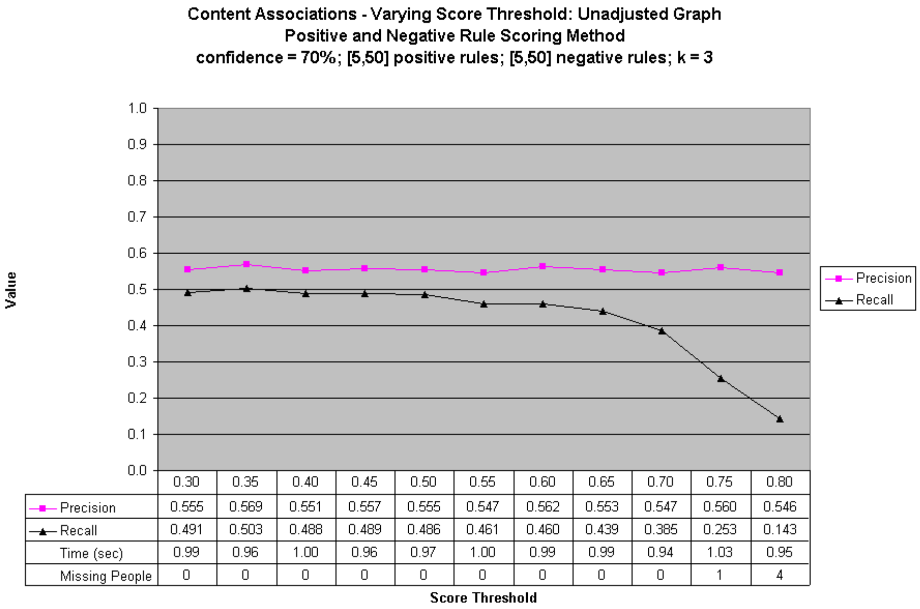

Content Associations - Varying Score Threshold: Unadjusted Graph
Positive and Negative Rule Scoring Method
confidence = 50%; [5,50] positive rules; [5,50] negative rules; k = 3

	0.30	0.35	0.40	0.45	0.50	0.55	0.60	0.65	0.70	0.75	0.80
■ Precision	0.484	0.484	0.486	0.489	0.508	0.537	0.560	0.508	0.433	0.298	0.283
▲ Recall	0.873	0.871	0.856	0.819	0.647	0.519	0.369	0.242	0.149	0.070	0.030
Time (sec)	0.83	0.81	0.8	0.8	0.83	0.78	0.85	0.82	0.84	0.77	0.81
Missing People	0	0	0	0	0	1	1	9	21	47	57


Content Associations - Varying Score Threshold: Adjusted Graph
Positive and Negative Rule Scoring Method
confidence = 50%; [5,50] positive rules; [5,50] negative rules; k = 3

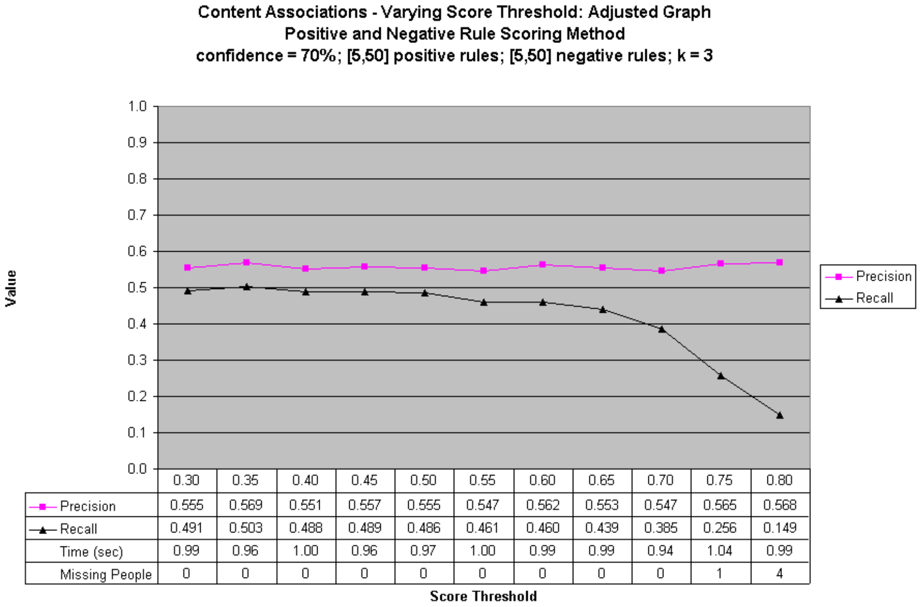
	0.30	0.35	0.40	0.45	0.50	0.55	0.60	0.65	0.70	0.75	0.80
■ Precision	0.484	0.484	0.486	0.489	0.508	0.543	0.566	0.559	0.547	0.563	0.657
▲ Recall	0.873	0.871	0.856	0.819	0.647	0.524	0.372	0.266	0.189	0.133	0.071
Time (sec)	0.83	0.81	0.80	0.80	0.83	0.79	0.86	0.90	1.06	1.45	1.88
Missing People	0	0	0	0	0	1	1	9	21	47	57

Figure 5.16: Content associations positive and negative rule scoring method results from varying the score threshold, mined at a minimum confidence of 50%. Time reported is for all four folds. Actual response time is approximately 25% of the times reported.



Content Associations - Varying Score Threshold: Unadjusted Graph
Positive and Negative Rule Scoring Method
confidence = 70%; [5,50] positive rules; [5,50] negative rules; k = 3

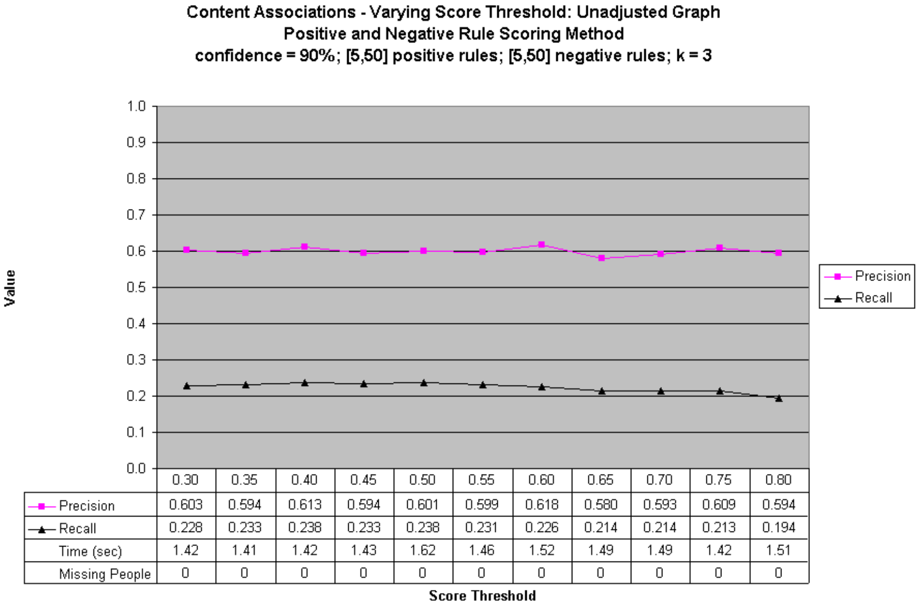
	0.30	0.35	0.40	0.45	0.50	0.55	0.60	0.65	0.70	0.75	0.80
Precision	0.555	0.569	0.551	0.557	0.555	0.547	0.562	0.553	0.547	0.560	0.546
Recall	0.491	0.503	0.488	0.489	0.486	0.461	0.460	0.439	0.385	0.253	0.143
Time (sec)	0.99	0.96	1.00	0.96	0.97	1.00	0.99	0.99	0.94	1.03	0.95
Missing People	0	0	0	0	0	0	0	0	0	1	4



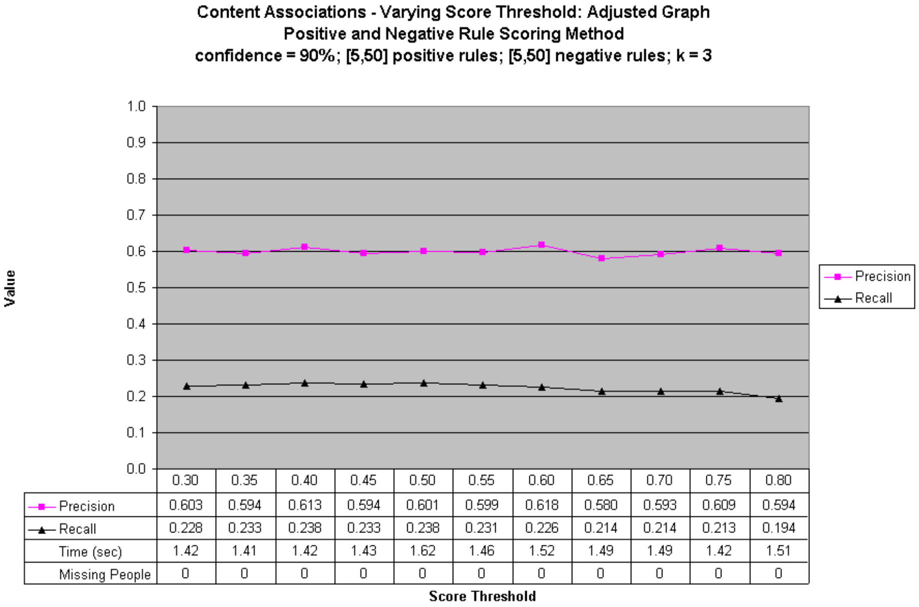
Content Associations - Varying Score Threshold: Adjusted Graph
Positive and Negative Rule Scoring Method
confidence = 70%; [5,50] positive rules; [5,50] negative rules; k = 3

	0.30	0.35	0.40	0.45	0.50	0.55	0.60	0.65	0.70	0.75	0.80
Precision	0.555	0.569	0.551	0.557	0.555	0.547	0.562	0.553	0.547	0.565	0.568
Recall	0.491	0.503	0.488	0.489	0.486	0.461	0.460	0.439	0.385	0.256	0.149
Time (sec)	0.99	0.96	1.00	0.96	0.97	1.00	0.99	0.99	0.94	1.04	0.99
Missing People	0	0	0	0	0	0	0	0	0	1	4

Figure 5.17: Content associations positive and negative rule scoring method results from varying the score threshold, mined at a minimum confidence of 70%. Time reported is for all four folds. Actual response time is approximately 25% of the times reported.



	0.30	0.35	0.40	0.45	0.50	0.55	0.60	0.65	0.70	0.75	0.80
Precision	0.603	0.594	0.613	0.594	0.601	0.599	0.618	0.580	0.593	0.609	0.594
Recall	0.228	0.233	0.238	0.233	0.238	0.231	0.226	0.214	0.214	0.213	0.194
Time (sec)	1.42	1.41	1.42	1.43	1.62	1.46	1.52	1.49	1.49	1.42	1.51
Missing People	0	0	0	0	0	0	0	0	0	0	0



	0.30	0.35	0.40	0.45	0.50	0.55	0.60	0.65	0.70	0.75	0.80
Precision	0.603	0.594	0.613	0.594	0.601	0.599	0.618	0.580	0.593	0.609	0.594
Recall	0.228	0.233	0.238	0.233	0.238	0.231	0.226	0.214	0.214	0.213	0.194
Time (sec)	1.42	1.41	1.42	1.43	1.62	1.46	1.52	1.49	1.49	1.42	1.51
Missing People	0	0	0	0	0	0	0	0	0	0	0

Figure 5.18: Content associations positive and negative rule scoring method results from varying the score threshold, mined at a minimum confidence of 90%. Time reported is for all four folds. Actual response time is approximately 25% of the times reported.

method views negative rules as positive rules when scoring, a negative rule with low confidence, for example 30%, gets converted to a positive rule of high confidence, in this case 70%. Viewing negative rules with low confidence as positive rules with high confidence in fact may help recommendations greatly. This observation motivated us to try and mine content associations via the positive and negative rule scoring method using two confidence values, a high confidence for the positive rules and a low confidence for the negative rules. These results actually yield higher precisions and are discussed in detail in the next section.

2. As the confidence increases, more people receive predictions from the system, even with higher score thresholds.

3. As the confidence and score threshold both increase, the positive effect on precision decreases. As shown in the graphs, as the confidence increases, the increase in precision on the adjusted graphs is not as great, and when a 90% minimum confidence is reached, the precision just perturbates around a precision of 60%.

4. As the confidence and score threshold both increase, the negative affect on recall decreases. As shown in the graphs, the recall tends to become more of a horizontal line for different confidence values, instead of yielding high values for low score thresholds and then decreasing sharply for large score thresholds.

5. As the confidence increases, the overall recall decreases. This phenomenon has already been seen in previous experiments for a constant score threshold. The same trend is also visible with modifications to the positive and negative rule scoring method threshold.

5.4.5 Experiment 2-5: Mining with different positive and negative rule confidences

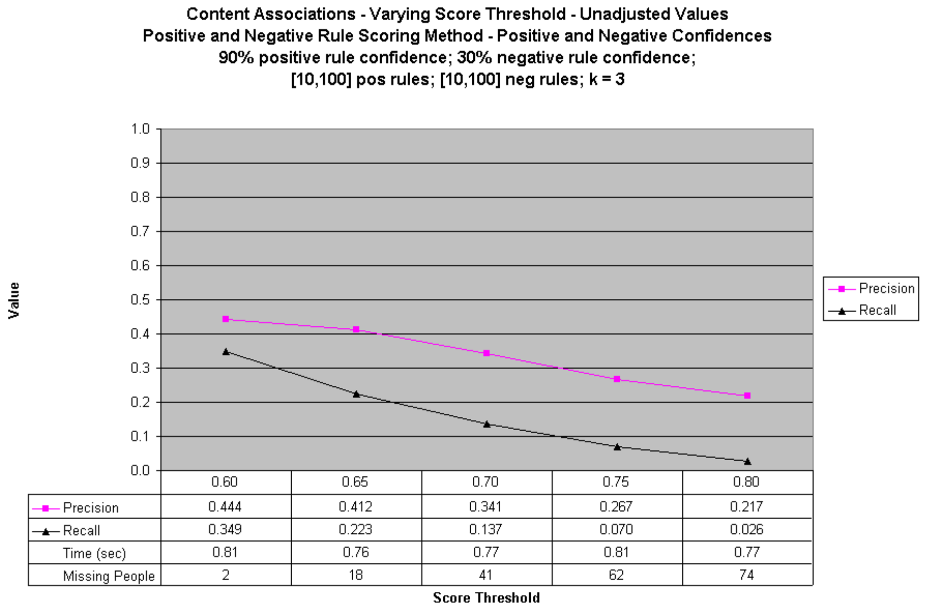
After seeing that a minimum confidence of 30% yielded a high precision in Experiment 2-4 for a high score threshold, we decided to mine using two different confidence values: one for the positive rules and one for the negative rules. Essentially, we test to see if viewing negative rules with low confidences as positive rules with high confidences helps in recommendation. To this end, we chose 90% for the positive rule confidence and 30% for the negative rule confidence. We run a series of tests, varying the score threshold between 0.6 and 0.8, since the higher score threshold values produce more precise results as shown in Experiment 2-4.

These results show that negative rules can be viewed and effectively viewed as positive rules if the score threshold is high. In particular, for a score threshold of 0.75 and 0.80 the precisions are greater than the precisions for those same score threshold values when mining at 30% minimum confidence for both types of rules shown in Figure 5.15. Additionally, the greatest precision, 0.836, for content that we have seen is attained for a score threshold of 0.80 in this experiment.

5.4.6 Summary of results for the new score threshold experiments

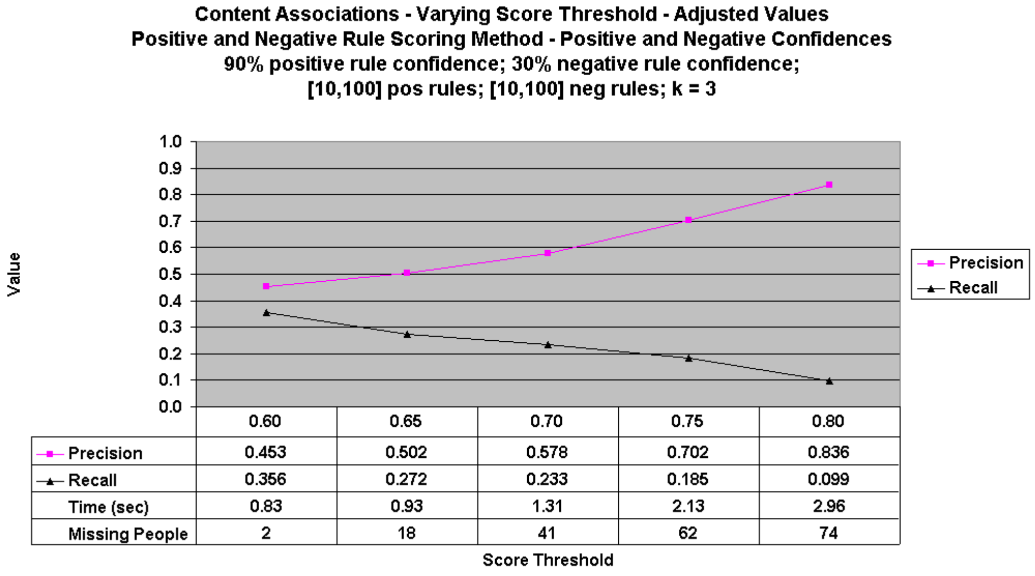
We summarize the effects each parameter has on precision, recall, and the number of people not receiving predictions below:

1. For a lower score threshold, for example 0.5 as shown in Experiment 2-1, increasing the minimum confidence increases the precision and decreases the recall. When using just one confidence value for both positive and negative rules, mining with a higher minimum confidence ensures that everyone will



Content Associations - Varying Score Threshold - Unadjusted Values
Positive and Negative Rule Scoring Method - Positive and Negative Confidences
90% positive rule confidence; 30% negative rule confidence;
[10,100] pos rules; [10,100] neg rules; k = 3

	0.60	0.65	0.70	0.75	0.80
Precision	0.444	0.412	0.341	0.267	0.217
Recall	0.349	0.223	0.137	0.070	0.026
Time (sec)	0.81	0.76	0.77	0.81	0.77
Missing People	2	18	41	62	74



Content Associations - Varying Score Threshold - Adjusted Values
Positive and Negative Rule Scoring Method - Positive and Negative Confidences
90% positive rule confidence; 30% negative rule confidence;
[10,100] pos rules; [10,100] neg rules; k = 3

	0.60	0.65	0.70	0.75	0.80
Precision	0.453	0.502	0.578	0.702	0.836
Recall	0.356	0.272	0.233	0.185	0.099
Time (sec)	0.83	0.93	1.31	2.13	2.96
Missing People	2	18	41	62	74

Figure 5.19: Content associations positive and negative rule scoring method results from varying the score threshold using a 90% minimum confidence when mining "like" rules and a 30% minimum confidence when mining "dislike" rules. Time reported is for all four folds. Actual response time is approximately 25% of the times reported.

receive predictions according to our experiments. When using just one confidence value for both positive and negative rules, lower confidences, such as 30% as shown in Experiment 2-4, yield higher precisions with less people receiving predictions as the score threshold increases.

2. Mining with a high confidence for positive rules and a low confidence for negative rules yields very high precision, a low recall, and many people not receiving predictions.

3. Varying the number of positive and negative rules does not greatly affect either the precision, or the number of people receiving predictions. However, increasing the minimum and maximum number of positive rules increases the recall as shown in Experiment 2-2.

4. Varying the negative rule reduction parameter also does not greatly affect either the precision, or the recall as shown in Experiment 2-3.

We try to maximize precision by (1) setting the minimum confidence to be 30%, 90% or 100%, (2) setting both the positive and negative rule ranges to be between 10 and 100, and (3) set the score threshold to be 0.8. We expect that many people at the 30% confidence will not receive predictions as shown in previous experiments, but we do expect everyone at the 90% and 100% confidence level to receive predictions. Since a high score threshold tends to yield higher predictions at different confidence values, we aim to see the quality of predictions returned by the system at different confidence values, keeping the high score threshold fixed. In order to compare this content scoring method with the linear method discussed previously, we create three distributions using the three confidence values: 30%, 90%, and 100%. The results, not counting those who did not receive predictions, are summarized in Table 5.1.

conf	*pos rules*	*neg rules*	*thresh*	*precision*	*recall*	*time (sec)*	*missing people*
30%	[10,100]	[10,100]	0.8	0.742	0.091	3.24	75
90%	[10,100]	[10,100]	0.8	0.580	0.213	1.60	0
100%	[10,100]	[10,100]	0.8	0.594	0.178	1.83	0

Table 5.2: Summary of adjusted results obtained when mining with the best parameters for content associations with the positive and negative rule score method. Time reported is for all four folds. Actual response time is approximately 25% of the times reported.

Table 5.2 shows that using a low confidence, just as before, yields a high precision with fewer people receiving predictions. Using a low confidence on the other hand yields a lower precision, but everyone receives predictions. Although the higher confidences have a lower average precision than the 30% confidence trial, we still would like to see how many people are actually receiving predictions which yield over 80% precision as done with the linear scoring method. Again, we created distributions to see the number of high quality recommendations returned by the system for all target users using the mining parameters mentioned in Table 5.2. These results, which ignore those people who did not receive predictions, are shown in Figure 5.20, Figure 5.21, and Figure 5.22.

The distributions yield some interesting observations:

- The number of people for which completely incorrect results, where the precision and recall are zero, is higher for lower confidence values. To see this, we calculate the percentage of people who received completely wrong predictions out of all those who received predictions. With a minimum confidence of 30%, 12% of the people received completely wrong predictions; at 90% confidence 4% received completely wrong predictions; and at 100% confidence, 2% received completely wrong predictions. All of these percentages are *lower* than the percentages from the linear scoring method.

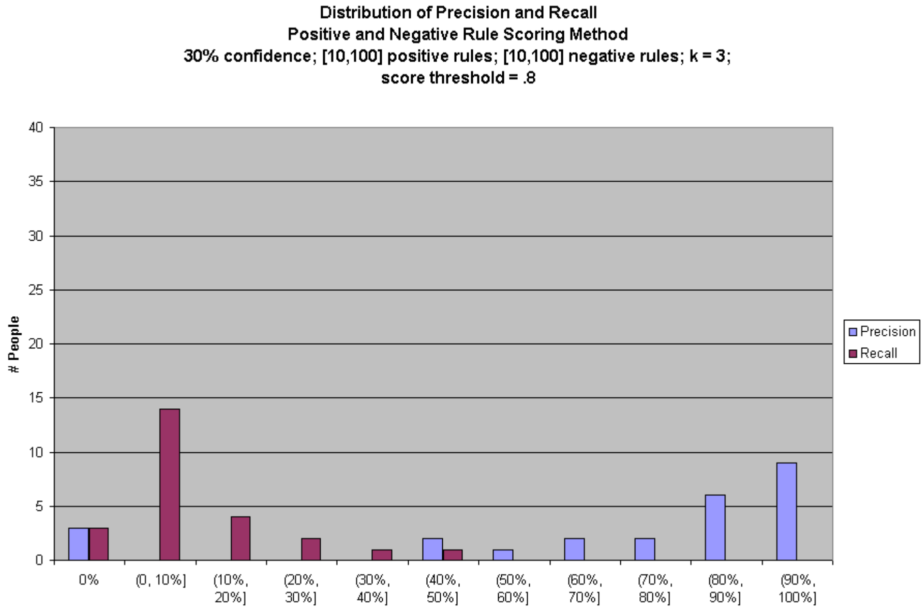


Figure 5.20: Distribution of results for high-valued mining parameters; minimum confidence of 30%

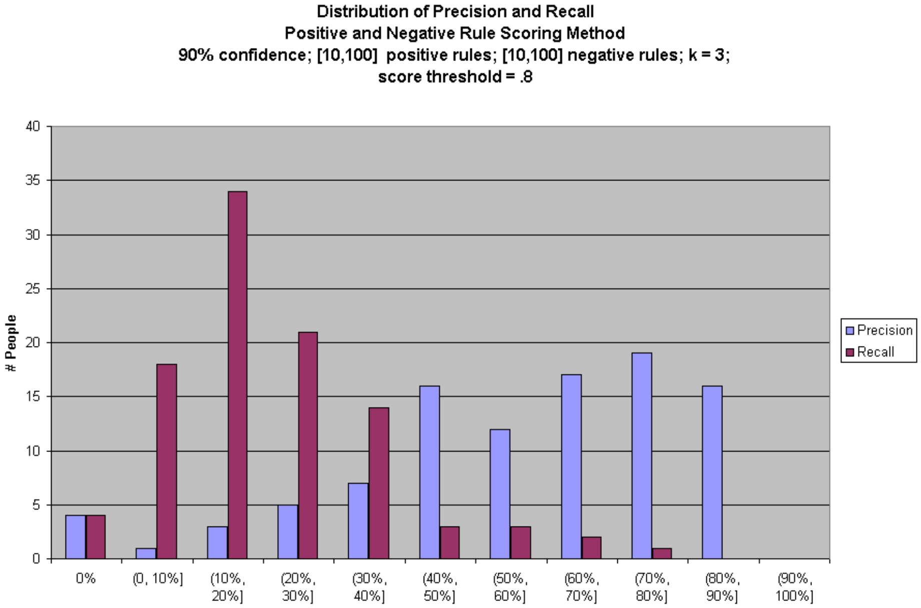


Figure 5.21: Distribution of results for high-valued mining parameters; minimum confidence of 90%

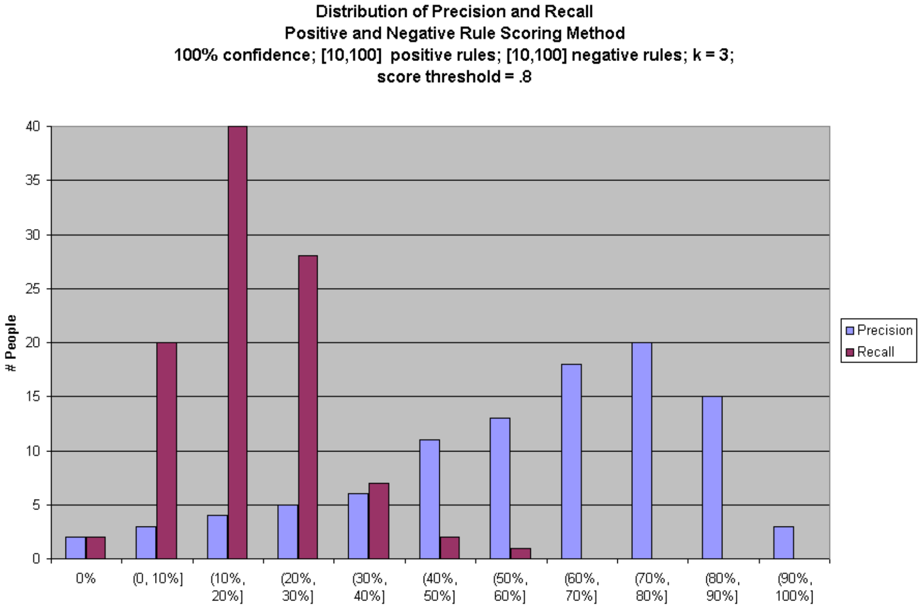


Figure 5.22: Distribution of results for high-valued mining parameters; minimum confidence of 100%

- The number of people receiving higher precisions, precisions above 0.8, is highest for a minimum confidence of 30%. With a minimum confidence of 30%, 60% of the people receiving recommendations attain a precision greater than 0.8. The percentage of people receiving high quality recommendations is *lower* than using the linear scoring method or content associations.

- The distribution for recall with higher minimum confidences for the positive and negative rule scoring method was much greater than distribution for the linear scoring method. In particular, if we view the 30% confidence distribution where high precision is yielded, more people receive higher recall values as compared to the high score threshold combined with high minimum confidence, which returned high precision values for the linear score method.

These distributions, along with the information given in Table 5.1 show that content associations scored with the positive and negative rule scoring method can also yield precise results. In particular, using the correct parameters can yield high precisions as shown in the distributions. If we view the minimum confidence of 90% and 100%, everyone receives predictions as compared to the linear scoring method. However, the overall precision returned by the positive and negative rule scoring method is *lower* than the linear scoring method, whereas the overall recall returned by the positive and negative rule scoring method is *higher* than the linear scoring method. Therefore, the linear scoring method should be used over the positive and negative method when trying to make more precise recommendations. If our aim is to return more predictions, then the positive and negative rule scoring method should be used.

5.5 Comparing Content Associations with Regular Collaborative Associations

In this set of experiments, we compare the precision and recall attained from recommendation via content associations with the collaborative-based recommendation scheme performed in [LAR02]. When the number of rated movies is high, which is true in this case since everyone in our dataset rated over 100 items, collaborative methods are expected to outperform content methods, because collaborative better captures the quality of "liking" a movie than content. However, we still run these experiments to see close the precision and recall are between the content associations method and the collaborative associations method.

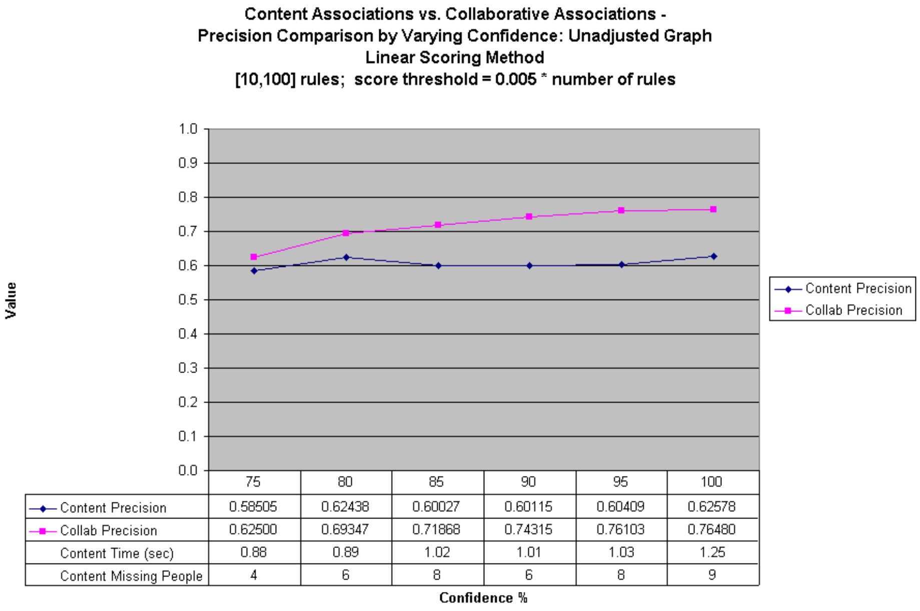
We compare the linear scoring method for content associations with the linear scoring method for collaborative associations in [LAR02]. We did not compare with the positive and negative rule scoring method for content associations, because the two methods have different parameters; therefore the parameters, in particular the score threshold, cannot be set evenly to make a direct comparison.

We compare these two approaches by first varying the confidence values and then the score threshold values, varying each separately. The following parameter values were used as the base values for performing the experiments:

- minimum confidence = 90%
- minimum number of rules = 10
- maximum number of rules = 100
- rule length = 8
- score threshold = 0.005 * number of rules mined

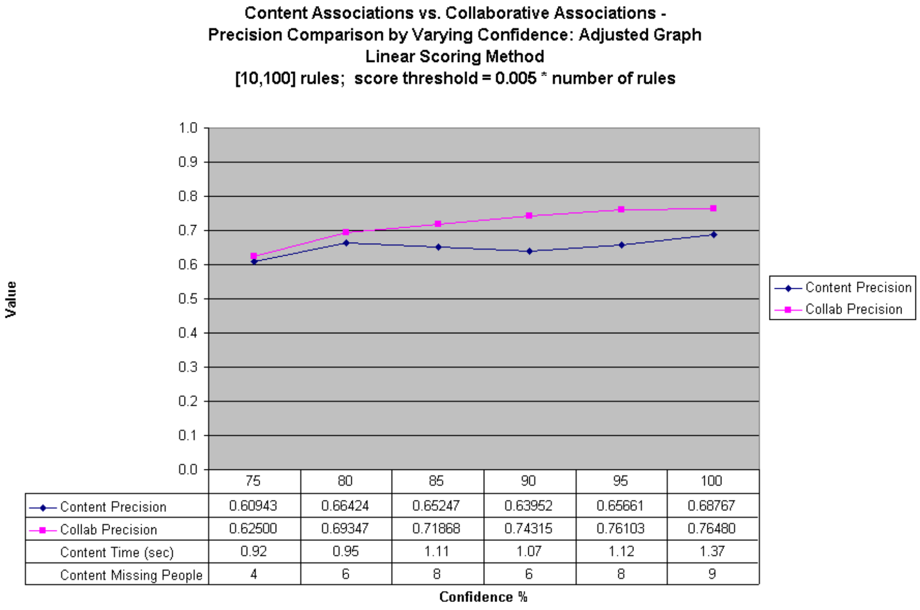
5.5.1 Experiment 3-1: Comparing Content with Collaborative - Varying the minimum confidence

In this experiment, different results were obtained by varying the minimum confidence between 75% and 100% while keeping the other mining parameters constant. The graphs presented in Figure 5.23 and Figure 5.24 show the effects of this modification on the precision and recall.



Content Associations vs. Collaborative Associations - Precision Comparison by Varying Confidence: Unadjusted Graph Linear Scoring Method [10,100] rules; score threshold = 0.005 * number of rules

	75	80	85	90	95	100
Content Precision	0.58505	0.62438	0.60027	0.60115	0.60409	0.62578
Collab Precision	0.62500	0.69347	0.71868	0.74315	0.76103	0.76480
Content Time (sec)	0.88	0.89	1.02	1.01	1.03	1.25
Content Missing People	4	6	8	6	8	9



Content Associations vs. Collaborative Associations - Precision Comparison by Varying Confidence: Adjusted Graph Linear Scoring Method [10,100] rules; score threshold = 0.005 * number of rules

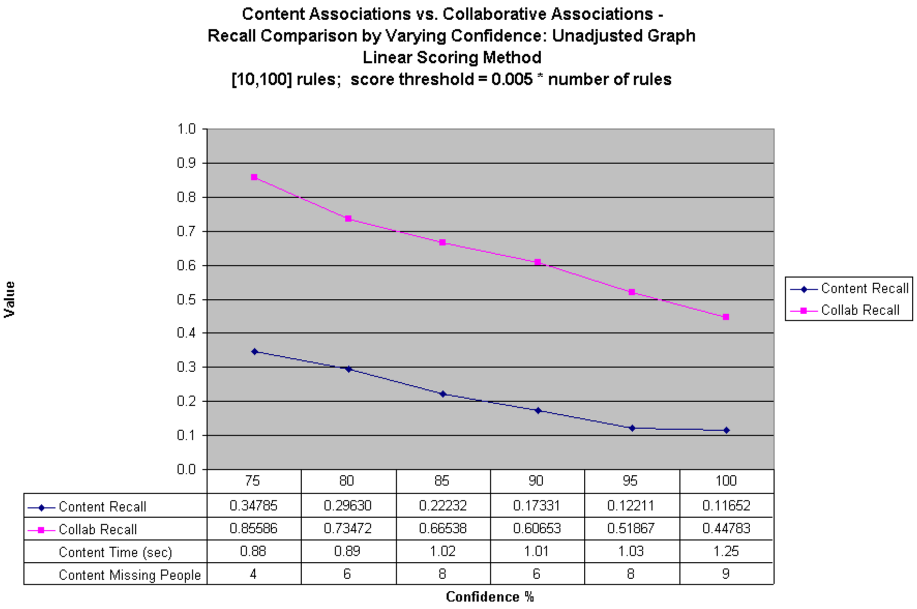
	75	80	85	90	95	100
Content Precision	0.60943	0.66424	0.65247	0.63952	0.65661	0.68767
Collab Precision	0.62500	0.69347	0.71868	0.74315	0.76103	0.76480
Content Time (sec)	0.92	0.95	1.11	1.07	1.12	1.37
Content Missing People	4	6	8	6	8	9

Figure 5.23: Content associations vs. Collaborative associations: precision results from varying the minimum confidence.Time reported is for all four folds. Actual response time is approximately 25% of the times reported.

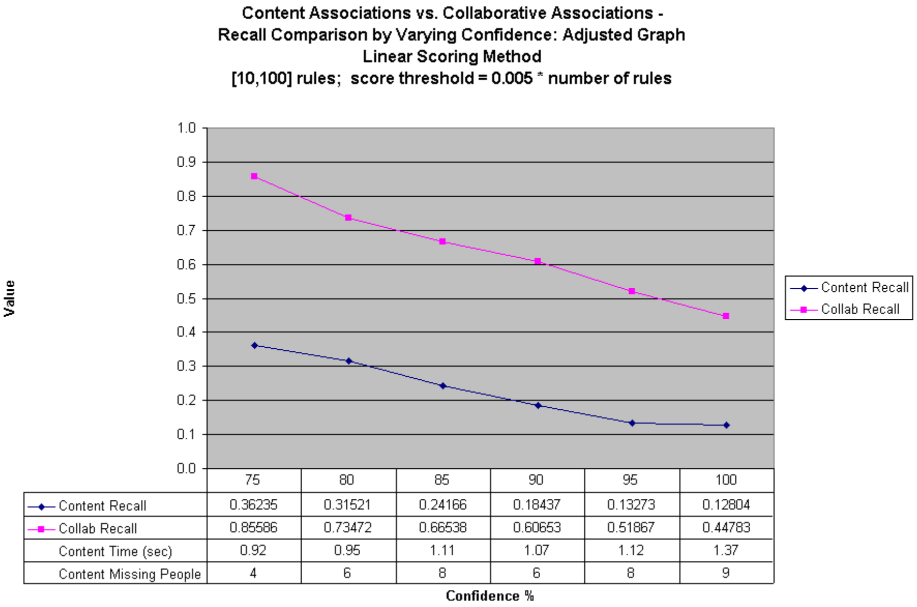
As shown in Figure 5.23 and Figure 5.24 for lower confidence values, the precision between the content and collaborative approaches is very close, however as the confidence increases, the precision distance between the two increases. The collaborative approach does outperform the content approach for all confidence values. Additionally, the recall for content is much lower than the collaborative's recall for all confidence values.

5.5.2 Experiment 3-2: Comparing Content with Collaborative - Varying the score threshold parameter

In this experiment, different results were obtained by varying the score threshold parameter between 0.0025 and 0.2 while keeping the other mining parameters constant. The graphs presented in Figure 5.25 and Figure 5.26 show the effects of this modification on the precision and recall.

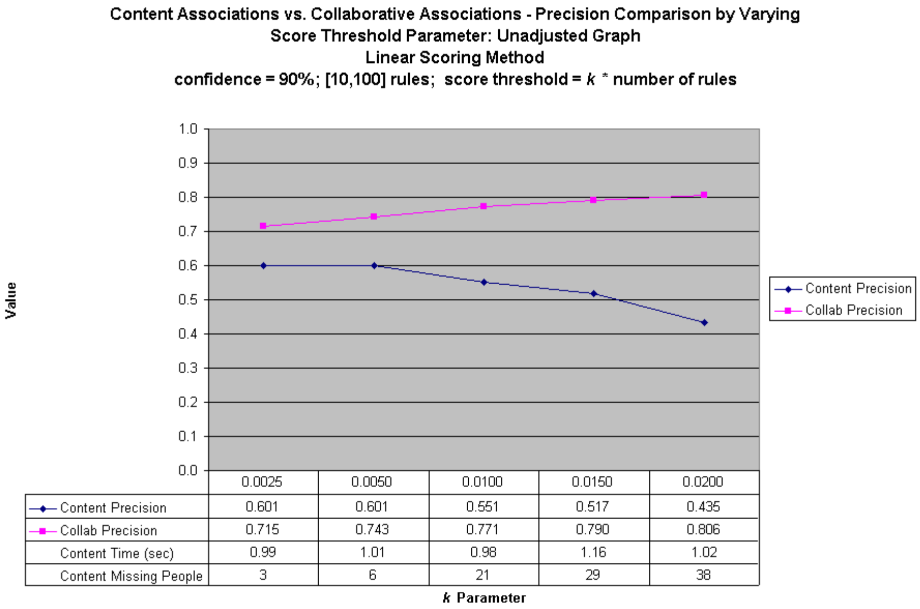


	75	80	85	90	95	100
Content Recall	0.34785	0.29630	0.22232	0.17331	0.12211	0.11652
Collab Recall	0.85586	0.73472	0.66538	0.60653	0.51867	0.44783
Content Time (sec)	0.88	0.89	1.02	1.01	1.03	1.25
Content Missing People	4	6	8	6	8	9



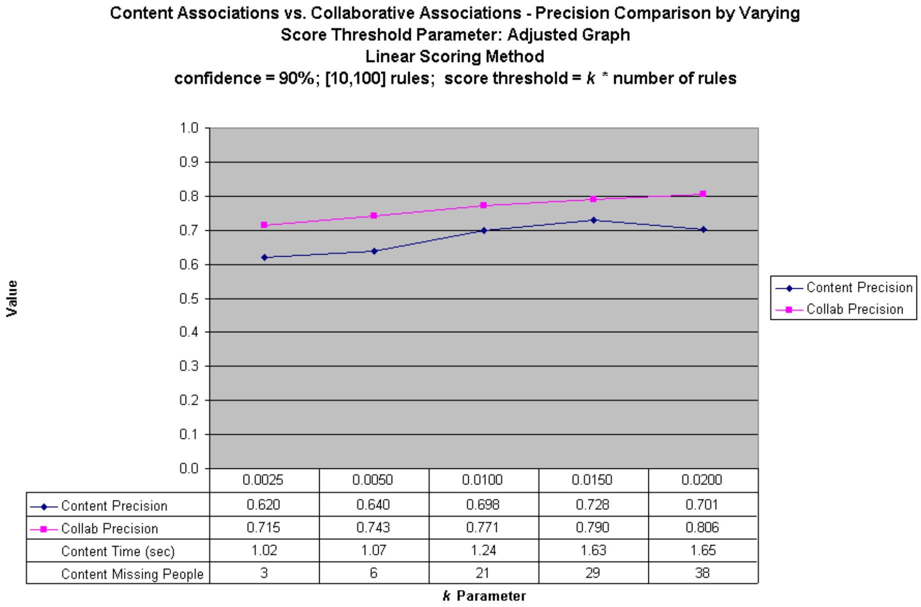
	75	80	85	90	95	100
Content Recall	0.36235	0.31521	0.24166	0.18437	0.13273	0.12804
Collab Recall	0.85586	0.73472	0.66538	0.60653	0.51867	0.44783
Content Time (sec)	0.92	0.95	1.11	1.07	1.12	1.37
Content Missing People	4	6	8	6	8	9

Figure 5.24: Content associations vs. Collaborative associations: recall results from varying the minimum confidence. Time reported is for all four folds. Actual response time is approximately 25% of the times reported.



Content Associations vs. Collaborative Associations - Precision Comparison by Varying Score Threshold Parameter: Unadjusted Graph
Linear Scoring Method
confidence = 90%; [10,100] rules; score threshold = *k* * number of rules

	0.0025	0.0050	0.0100	0.0150	0.0200
Content Precision	0.601	0.601	0.551	0.517	0.435
Collab Precision	0.715	0.743	0.771	0.790	0.806
Content Time (sec)	0.99	1.01	0.98	1.16	1.02
Content Missing People	3	6	21	29	38



Content Associations vs. Collaborative Associations - Precision Comparison by Varying Score Threshold Parameter: Adjusted Graph
Linear Scoring Method
confidence = 90%; [10,100] rules; score threshold = *k* * number of rules

	0.0025	0.0050	0.0100	0.0150	0.0200
Content Precision	0.620	0.640	0.698	0.728	0.701
Collab Precision	0.715	0.743	0.771	0.790	0.806
Content Time (sec)	1.02	1.07	1.24	1.63	1.65
Content Missing People	3	6	21	29	38

Figure 5.25: Content associations vs. Collaborative associations: precision results from varying the score threshold parameter. Time reported is for all four folds. Actual response time is approximately 25% of the times reported.

As shown in Figure 5.25 and Figure 5.26 collaborative's precision again outperforms the content for all trials. However, for a score threshold parameter of 0.015, the two precision values are relatively close. Also, the recall again for collaborative is much greater than content.

One piece of information not considered up until this point is the speed at which recommendations are generated. Although the precision and recall for the content associations method are lower than the collaborative associations method's precision and recall, the speed at which the predictions are generated is much greater. [LAR02] reported an average prediction generation time of 14.2 seconds for producing recommendations via collaborative associations, whereas our content associations method produces recommendations in under one second on average, thus meeting the real time requirement.

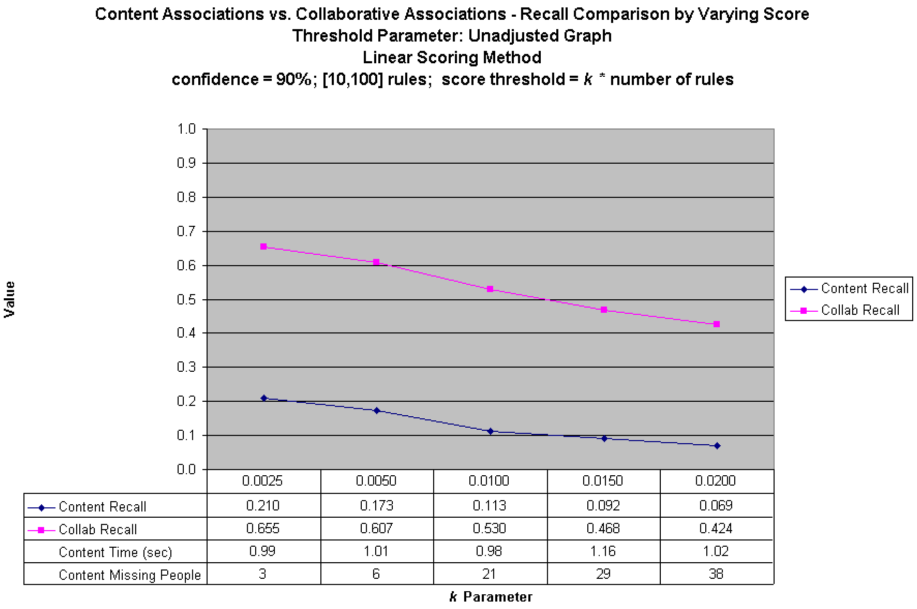
5.6 Content-Based Collaborative Experiments

In these experiments, we compare our content-based collaborative approach with the regular collaborative approach presented in [LAR02]. The content-based collaborative approach uses the same linear scoring method as [LAR02]. We did not use the positive and negative rule scoring method in testing the content-based collaborative associations, because mining two sets of rules in this case took too long.

Our setup for the content-based collaborative experiments is as follows:

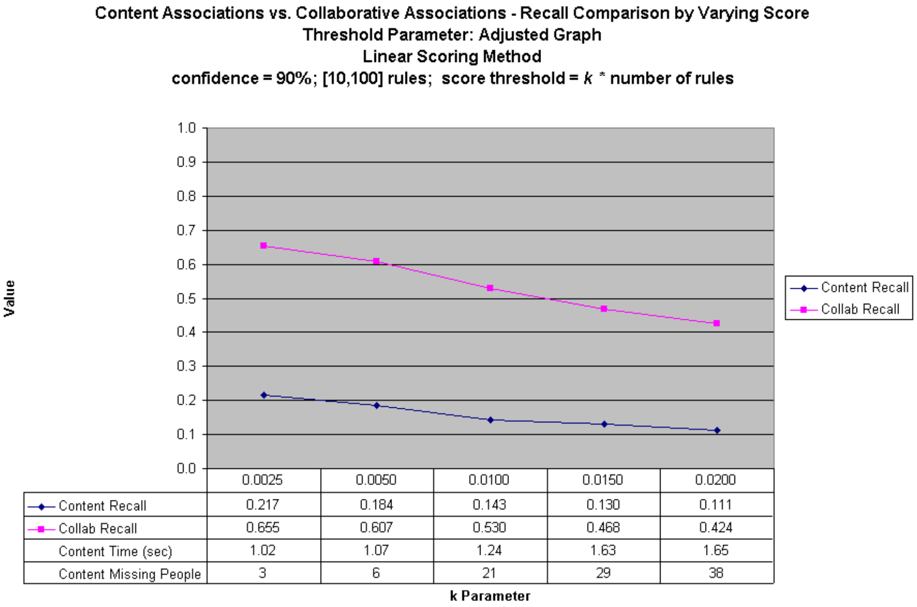
- We selected the same 1000 collaborative users as [LAR02]. In particular, these users were the first 1000 users who rated over 100 movies in the Eachmovie dataset.

- In order to choose the properties used to mine content-based collaborative associations, we combined each collaborative users' top 100 scoring properties,



Content Associations vs. Collaborative Associations - Recall Comparison by Varying Score Threshold Parameter: Unadjusted Graph
Linear Scoring Method
confidence = 90%; [10,100] rules; score threshold = *k* * number of rules

	0.0025	0.0050	0.0100	0.0150	0.0200
Content Recall	0.210	0.173	0.113	0.092	0.069
Collab Recall	0.655	0.607	0.530	0.468	0.424
Content Time (sec)	0.99	1.01	0.98	1.16	1.02
Content Missing People	3	6	21	29	38



Content Associations vs. Collaborative Associations - Recall Comparison by Varying Score Threshold Parameter: Adjusted Graph
Linear Scoring Method
confidence = 90%; [10,100] rules; score threshold = *k* * number of rules

	0.0025	0.0050	0.0100	0.0150	0.0200
Content Recall	0.217	0.184	0.143	0.130	0.111
Collab Recall	0.655	0.607	0.530	0.468	0.424
Content Time (sec)	1.02	1.07	1.24	1.63	1.65
Content Missing People	3	6	21	29	38

Figure 5.26: Content associations vs. Collaborative associations: recall results from varying the score threshold parameter. Time reported is for all four folds. Actual response time is approximately 25% of the times reported.

using the property scoring method discussed in Section 3.3. Again, since we feel that genre is the most important content property, we do not include genre in selection of the top-100 properties. Instead, we include genre automatically.

- Instead of selecting the target user's top-100 scoring properties, we use all of their properties. We use all the properties in order to increase the number of transactions for which content-based collaborative associations are mined as discussed in Section 3.5.
- A user, either collaborative or target, was considered to like a movie if more than 50% of the movies having that property were rated as "like" as discussed in Section 3.5.

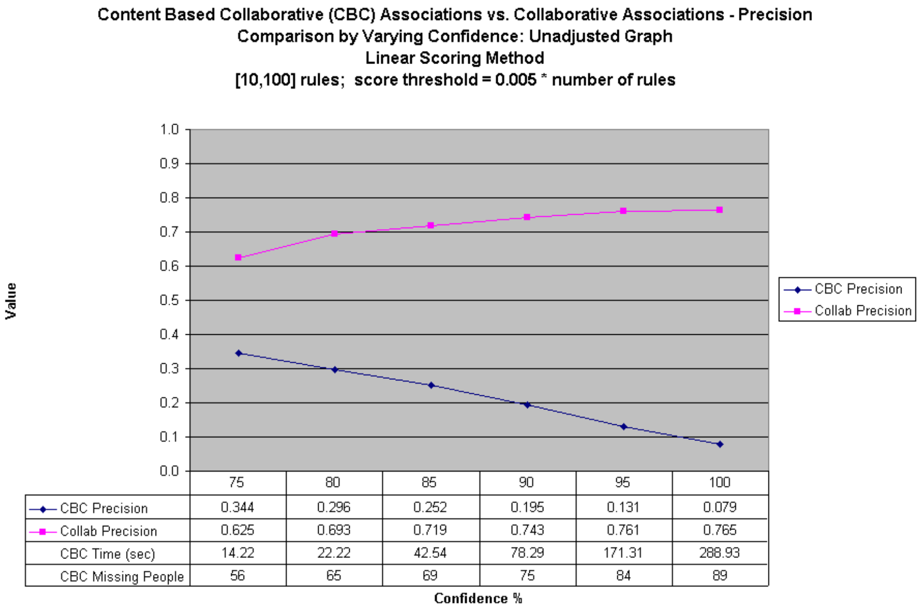
To compare the two approaches, we use the same mining parameters as [LAR02]. We first vary the minimum confidence, and then the linear score threshold parameter, as done in Section 5.5.

The following parameter values were used as the base values for performing the experiments:

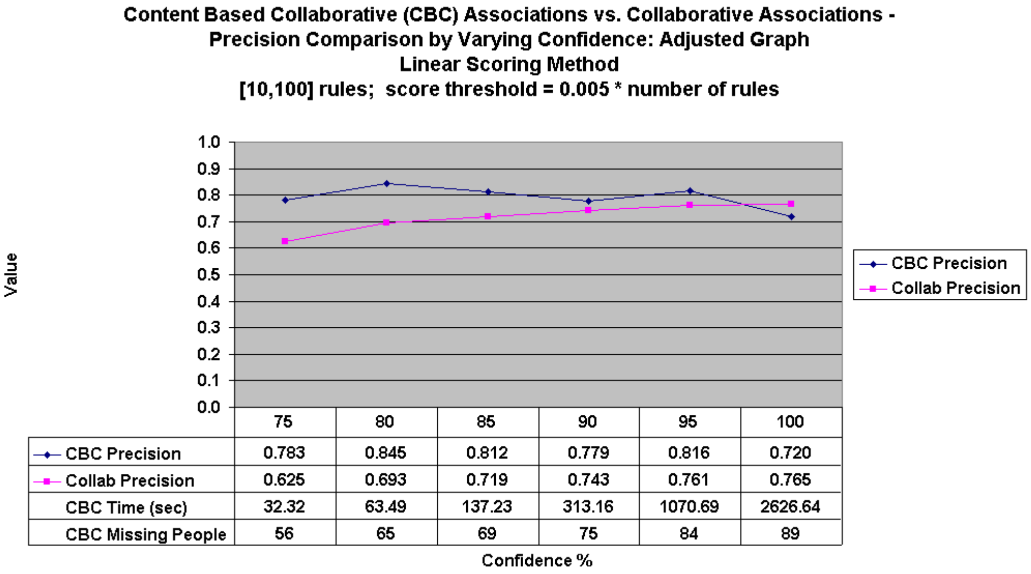
- minimum confidence = 90%
- minimum number of rules = 10
- maximum number of rules = 100
- rule length = 8
- score threshold = 0.005 * number of rules mined

5.6.1 Experiment 4-1: Comparing Content-based Collaborative with Regular Collaborative - Varying the minimum confidence

In this experiment, different results were obtained by varying the minimum confidence between 75% and 100% while keeping the other mining parameters constant. The graphs presented in Figure 5.27 and Figure 5.28 show the effects of this modification on the precision and recall.



	75	80	85	90	95	100
CBC Precision	0.344	0.296	0.252	0.195	0.131	0.079
Collab Precision	0.625	0.693	0.719	0.743	0.761	0.765
CBC Time (sec)	14.22	22.22	42.54	78.29	171.31	288.93
CBC Missing People	56	65	69	75	84	89



	75	80	85	90	95	100
CBC Precision	0.783	0.845	0.812	0.779	0.816	0.720
Collab Precision	0.625	0.693	0.719	0.743	0.761	0.765
CBC Time (sec)	32.32	63.49	137.23	313.16	1070.69	2626.64
CBC Missing People	56	65	69	75	84	89

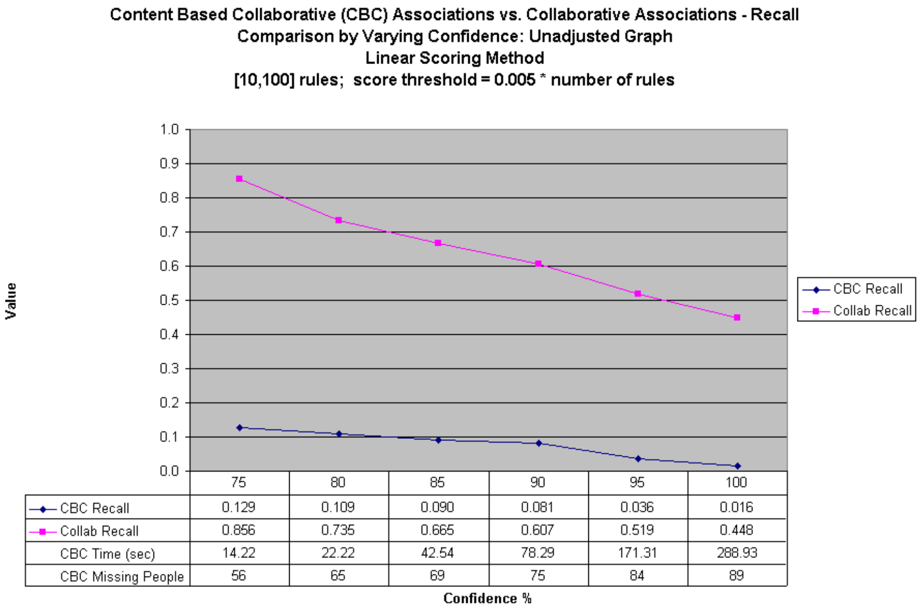
Figure 5.27: Content-based Collaborative associations vs. Regular Collaborative associations: precision results from varying the minimum confidence.

As expected, overall the content-based collaborative approach outperforms the regular collaborative approach. In particular, at a confidence of 80%, the precision, 0.845, is the highest precision found thus far in all experiments. Additionally, unlike the content association approaches where a high precision is paired with a very low recall, the recall at a confidence of 80% is much higher: 0.312.

Two downsides to the content-based collaborative approaches are described as follows:

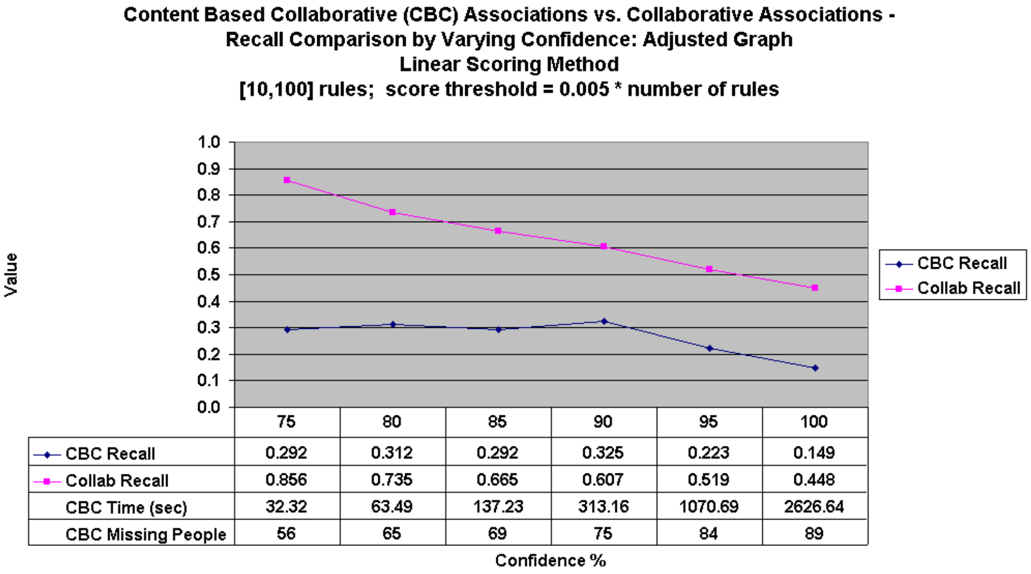
1. *Many people do not receive predictions from the system.* We can hypothesize that the lack of predictions can be caused by any one or a combination of many factors. First, our testing approach does not make predictions for all movies as done in the four-fold cross validation approach; only 30% of the total number of movies are given predictions. Therefore, there is less of a chance of receiving at least one prediction with which we can calculate the precision and recall. Another reason may be that the supports for the rules that are mined are very low. If the supports are very low, a movie may not be able to accrue enough score to be recommended using the linear scoring method. Finally, we feel that the score threshold should be set on a per-user basis, rather than globally. For example, a movie's score of 0.005 for one user may mean that one target user likes that movie, whereas for another target user, that score may not be high enough to signify liking that movie. We do note however, that when target users do receive predictions from this system, on average the precision is higher than either the regular collaborative approach or the content approach.

2. *The system returns predictions in an unacceptable amount of time.* The times returned by the content-based collaborative approach are much greater than



Content Based Collaborative (CBC) Associations vs. Collaborative Associations - Recall Comparison by Varying Confidence: Unadjusted Graph
Linear Scoring Method
[10,100] rules; score threshold = 0.005 * number of rules

	75	80	85	90	95	100
CBC Recall	0.129	0.109	0.090	0.081	0.036	0.016
Collab Recall	0.856	0.735	0.665	0.607	0.519	0.448
CBC Time (sec)	14.22	22.22	42.54	78.29	171.31	288.93
CBC Missing People	56	65	69	75	84	89



Content Based Collaborative (CBC) Associations vs. Collaborative Associations - Recall Comparison by Varying Confidence: Adjusted Graph
Linear Scoring Method
[10,100] rules; score threshold = 0.005 * number of rules

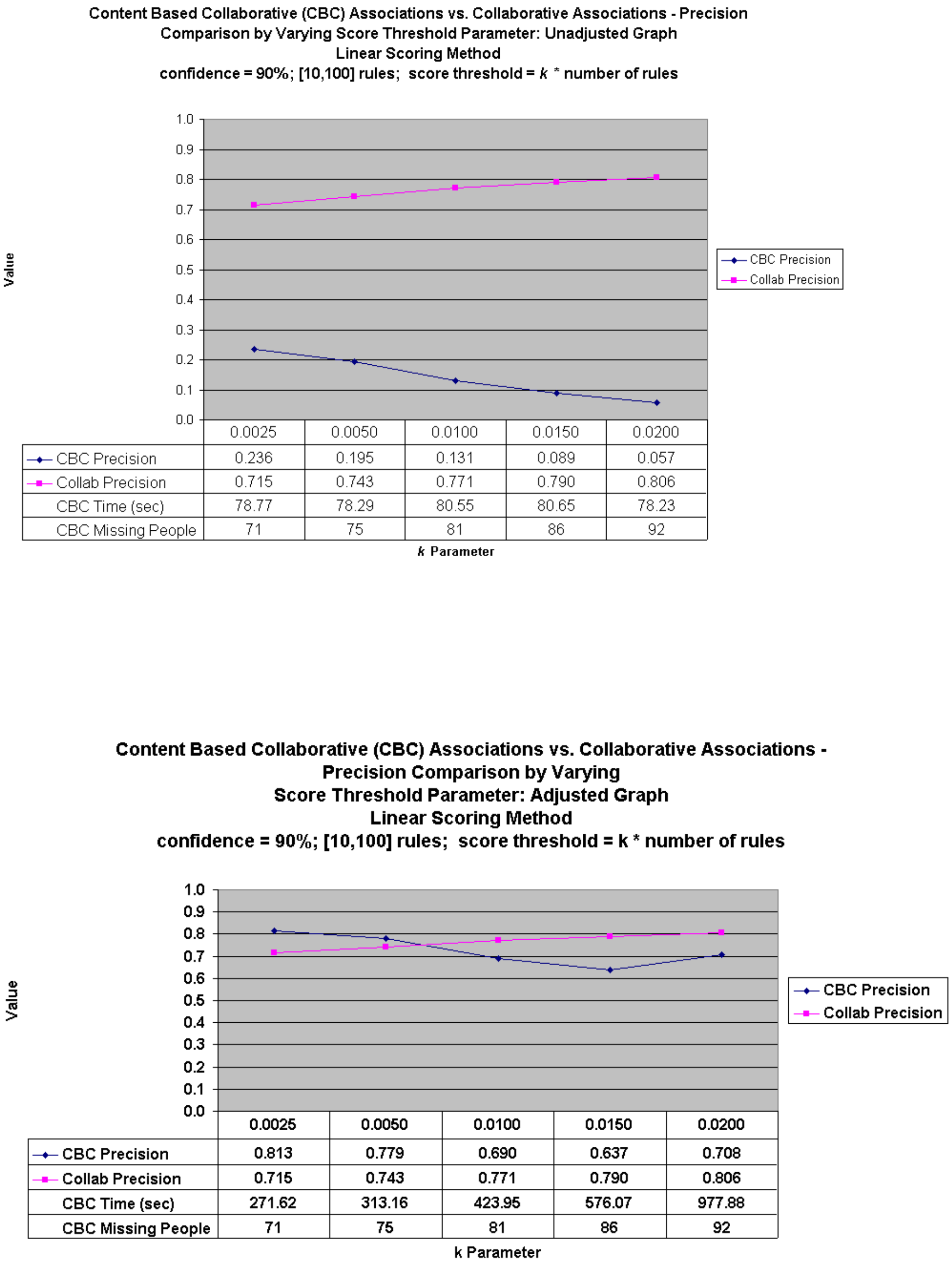
	75	80	85	90	95	100
CBC Recall	0.292	0.312	0.292	0.325	0.223	0.149
Collab Recall	0.856	0.735	0.665	0.607	0.519	0.448
CBC Time (sec)	32.32	63.49	137.23	313.16	1070.69	2626.64
CBC Missing People	56	65	69	75	84	89

Figure 5.28: Content-based Collaborative associations vs. Regular Collaborative associations: recall results from varying the minimum confidence.

both the regular collaborative approach and the content approach. We can attribute the increase in time as compared to the average collaborative associations runtime reported in [LAR02] of 14.2 seconds to an increase in the number of overlapping property ratings versus the number of overlapping article ratings. Since the maximum size of a transaction is the same in both cases, namingly 1000 collaborative users, if there is more overlap in the ratings, the mining process will take longer, because more frequent itemsets (see Section 2.3) can be found.

5.6.2 Experiment 4-2: Comparing Content-based Collaborative with Regular Collaborative - Varying the score threshold parameter

In this experiment, different results were obtained by varying the score threshold parameter between 0.0025 and 0.2 while keeping the other mining parameters constant. The graphs presented in Figure 5.29 and Figure 5.30 show the effects of this modification on the precision and recall.



	0.0025	0.0050	0.0100	0.0150	0.0200
CBC Precision	0.236	0.195	0.131	0.089	0.057
Collab Precision	0.715	0.743	0.771	0.790	0.806
CBC Time (sec)	78.77	78.29	80.55	80.65	78.23
CBC Missing People	71	75	81	86	92

	0.0025	0.0050	0.0100	0.0150	0.0200
CBC Precision	0.813	0.779	0.690	0.637	0.708
Collab Precision	0.715	0.743	0.771	0.790	0.806
CBC Time (sec)	271.62	313.16	423.95	576.07	977.88
CBC Missing People	71	75	81	86	92

Figure 5.29: Content-based Collaborative associations vs. Regular Collaborative associations: precision results from varying the score threshold parameter.

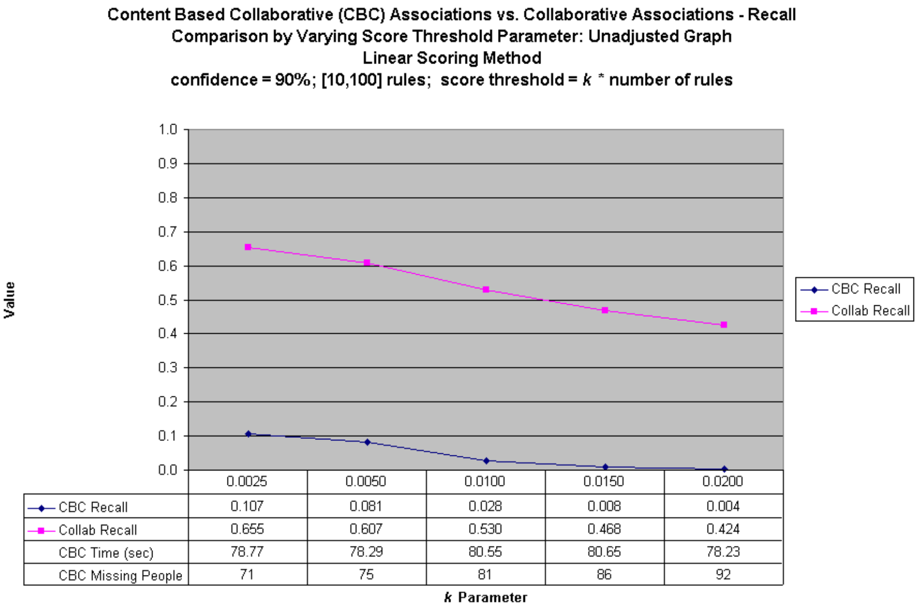
As shown in Figure 5.29 and Figure 5.30, lower score threshold values for a minimum confidence of 90% yield a higher precision for the content-based collaborative associations. As the score threshold increases however, the regular collaborative associations outperform the content-based collaborative associations in terms of precision. Again, we may attribute this not to a flaw in the scoring method, but rather setting the score threshold globally versus on a per-user basis. As in Experiment 4-1, the recall for content-based collaborative associations is consistently lower than for regular collaborative associations.

5.7 "Top-3" Method Comparisons

In this section we attempt to compare the overall ability of each recommendation method: (1) the content associations paired with linear scoring, (2) the content associations paired with positive and negative rule scoring, and (3) the content-based collaborative associations paired with linear scoring. In order to make these comparisons, we first eliminate the notion of recommendation if the score for a movie exceeds some threshold value. We instead recommend by returning the three highest scoring movies attained. This process is justified, because if the scoring method is working correctly, then the highest scoring movies should be those that the target user will like. We therefore test the ability of the scoring method to make those movies that the target user likes stand out through a high score relative to the other movies.

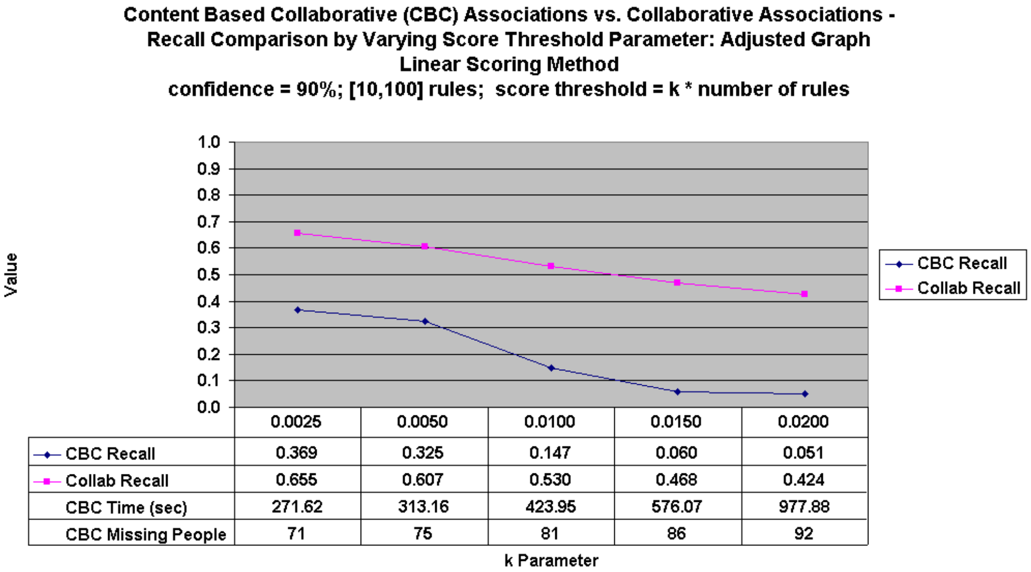
In order to compare the approaches, we chose the best mining parameters for each approach. These values are listed below:

- *content associations: linear scoring method.* minimum confidence = 90%; rule range = [50,200].



Content Based Collaborative (CBC) Associations vs. Collaborative Associations - Recall Comparison by Varying Score Threshold Parameter: Unadjusted Graph
Linear Scoring Method
confidence = 90%; [10,100] rules; score threshold = *k* * number of rules

	0.0025	0.0050	0.0100	0.0150	0.0200
CBC Recall	0.107	0.081	0.028	0.008	0.004
Collab Recall	0.655	0.607	0.530	0.468	0.424
CBC Time (sec)	78.77	78.29	80.55	80.65	78.23
CBC Missing People	71	75	81	86	92



Content Based Collaborative (CBC) Associations vs. Collaborative Associations - Recall Comparison by Varying Score Threshold Parameter: Adjusted Graph
Linear Scoring Method
confidence = 90%; [10,100] rules; score threshold = k * number of rules

	0.0025	0.0050	0.0100	0.0150	0.0200
CBC Recall	0.369	0.325	0.147	0.060	0.051
Collab Recall	0.655	0.607	0.530	0.468	0.424
CBC Time (sec)	271.62	313.16	423.95	576.07	977.88
CBC Missing People	71	75	81	86	92

Figure 5.30: Content-based Collaborative associations vs. Regular Collaborative associations: recall results from varying the score threshold parameter.

	content: linear	*content: pos/neg*	*CBC: linear*
precision	0.655	0.593	0.778
time	1.24	1.73	45.02
missing people	0	0	52

Table 5.3: Comparing our approaches: top-3 experiment results.

- *content associations: positive and negative rule scoring method.* minimum confidence = 90%; positive rule range = [10,100]; negative rule range = [10,100]; negative rule reduction parameter = 3.

- *content-based collaborative (CBC) associations: linear scoring method.* minimum confidence = 80%; rule range = [10,100]

The results acquired from running the top-3 experiments are shown in Table 5.3. As shown in Table 5.3 content-based collaborative associations yield the highest top-3 precision, but take the longest to create predictions. Additionally, content-based collaborative yielded many people who received no predictions, meaning that none of the test movies satisfied any rules.

Now that we see content-based collaborative returns the best results, but leaves many without predictions, we need to find a way to get those other target users results. In the next chapter, we present our final recommendation framework which attempts to provide high quality predictions for all target users.

Chapter 6

Final Recommendation Strategy

In this chapter, we present our framework for yielding precise recommendations for everyone. For the moment, we relax the real-time constraint, because we want to make use of the content-based collaborative associations which yield the highest precision. In particular, we use the results from Section 5.6 which showed that content-based collaborative associations paired with the linear scoring method yield the highest precision using a top-3 recommendation comparison approach. As shown in Table 5.3, however, more than half of the 100 test target users did not receive predictions using content-based collaborative associations. Therefore, we need to devise a "backup plan" for these users.

In particular, we propose mining other types of associations and using those to make predictions when content-based collaborative fails to yield recommendations. This process is outlined specifically in Figure 6.1. The order in which the associations are used to make predictions was derived from the experiments in the previous chapter. Overall, content-based collaborative yielded the most precise results, then regular collaborative, and finally content. Content also appears last in the list, because we can use the positive and negative rule scoring method to ensure that

```
1. Recommend using content-based collaborative associations.
   If no recommendations are returned,

   2. Recommend using regular collaborative associations.
      If no recommendations are returned,

      3. Recommend using the positive and negative rule
         scoring method for content.
```

Figure 6.1: Algorithm for combining all modes of recommendation

method	*% users receiving predictions*	*precision*	*recall*	*values taken from*
CBC	35%	.845	.312	Figure 5.27
Collab	96%	.806	.424	[Lin00], pg. 51
Content	100%	.615	.184	Figure 5.8
Combined	100%	.815	.379	Table 6.1

Table 6.1: Projected precision and recall for all methods combined so that all target users receive predictions.

everyone receives predictions as shown in previous experiments.

Table 6.1 shows some of the best values obtained from each of the mining methods. We can extrapolate the final precision and recall if the algorithm described in Figure 6.1 is employed. Although the combination of all methods yields a lower precision than the content-based collaborative method, the recall is higher and everyone receives predictions from the system.

Chapter 7

Conclusions

This thesis has shown that association rule mining can effectively be used for either content-based or collaborative-based recommendation purposes. In particular, we created the framework for extending the collaborative system described in [LAR02] to make use of available content information to improve the precision and recall of the system. We developed two new types of associations which can be used to produce recommendations: content associations, which map article properties to user preferences, and content-based collaborative associations, which uses ratings of article properties instead of ratings of articles as the basis for determining similarity between collaborative users and the target user. We also developed a new method for selecting the best properties on a per-user basis to use for creating content associations, and for selecting the properties used to determine similarity between users. This feature selection method was essential in obtaining content-based predictions in real-time. Additionally, we created a heuristic for mapping ratings of articles to ratings of article properties for use in mining content-based collaborative associations.

Through our experimentation, we discovered that all three forms of associations:

content associations, regular collaborative associations, and content-based collaborative associations can be configured correctly to achieve precise recommendations. Overall, the content-based collaborative approach yielded the most precise results, and also had a decent recall. However, many people did not receive recommendations for the content-based collaborative scheme. Following this observation, we devised a method to ensure that everyone received predictions. In particular, if a person did not receive any predictions from the best method, the content-based collaborative method, we attempt to attain predictions from the next-best method, in this case regular collaborative associations, and so-on. We projected results that show that not only can everyone receive predictions, but the average precision is higher when recommending in this way as compared to recommending with only content associations or regular collaborative associations.

7.1 Future Work

The following list illustrates some of the work that can be performed to further enhance the systems speed and prediction precision and recall:

- *Selecting the mining parameters on a per-user basis.* We feel that in particular for content-based collaborative associations, selecting the parameters on a per-user basis will aid in recommendation. In particular, we feel that one minimum confidence, rule range, or score threshold which may yield excellent predictions for one target user may produce bad predictions for another target user.

- *Finding a way to pick the best collaborative users.* We used a feature selection method in order to reduce the number of properties with which to mine in order to produce content associations. We chose the "best" properties based on two factors: (1) their high correlation with the target user liking or disliking movies, and (2) their frequency in the data. In particular, we would like to choose a subset of the collaborative users to utilize in mining rules by selecting those collaborative users based on the same two criteria. Since mining is exponential to the number of items (for content the items are properties, and for collaborative the items are users), reducing the number of items over which to mine will greatly speed up the production of associations.

- *Determining if users disliking an article property can be used to produce better recommendations using content-based collaborative associations.* This type of exploration would require that the best collaborative users be chosen, because since the association rule mining algorithm expects boolean input, we would need to double the number of items over which to mine. In particular, each user would be split into two different items: *user a_like* and *user a_dislike* for example as done for the content associations. Therefore, if we were to mine with 1000 collaborative users, we would need 2000 items to represent all of the possible combinations of ratings for each property, which would take even longer to mine. In order to test this approach, we would need to select the best users to utilize in order to mine content-based collaborative associations in this manner.

- *Testing the system with another dataset.* In order to tell if these recommendation methods work consistently, we could test using different datasets over different domains.

- *Running real tests with our recommendation strategy.* We did not test our approach of yielding everyone recommendations using the algorithm presented in Figure 6.1. Therefore, we would like to see if an actual implementation of this approach would match our expected precision and recall.

Appendix A

Other Experiments Performed

In this appendix, we present other experiments performed using our system. In particular, we ran some additional content association and content-based collaborative association experiments using different subsets of the EachMovie dataset in order to see if predictions were affected by the change in data.

A.1 Content Associations Positive and Negative Rule Scoring Method: Testing with Target Users Who Have not Rated Many Items

In this experiment, we chose the first 100 target users in the EachMovie database who rated between 15 and 50 movies in order to see if the system could produce good recommendations even when the users have not rated many items. We mined content associations using the following parameters:

- minimum confidence = 90%
- positive rule range = [5,50]

- negative rule range = [5,50]

- negative rule reduction parameter = 3

- score threshold = 0.7

The results from this experiment are summarized below:

- adjusted precision = 0.668

- adjusted recall = 0.479

- adjusted time = 0.229 sec, total time for four-fold cross validation

- number of missing people = 4

Even though the number of ratings a target user has given the system is smaller, the precision and recall are still on par with the other experiments.

A.2 Content-Based Collaborative Top-3: 2000 Collaborative Users

In this experiment, we use the same dataset as [BP98] in which the first 2000 users in the EachMovie dataset were used as collaborative users, and a random 91 users who have rated between 50 and 100 movies were used as target users. We mined content-based collaborative associations using the following parameters:

- minimum confidence = 70%

- rule range = [400,499]

The top-3 results from this experiment are summarized below:

- adjusted precision = 0.588
- adjusted time = 35.47 sec
- number of missing people = 55

The fact that the collaborative users may not have rated many items may contribute to the lower precision for the top-3 predictions.

Bibliography

[AIS93] Rakesh Agrawal, Tomasz Imielinski, and Arun Swami. Mining association rules between sets of items in large databases. In *Proceedings of the 1993 ACM SIGMOD Conference*, 1993.

[AS94] Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules. In *Proceedings of the 20th VLDB Conference*, 1994.

[BHK98] John S. Breese, David Heckerman, and Carl Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, 1998.

[BP98] Daniel Billsus and Michael Pazzani. Learning collaborative information filters. In *Proceedings of the Fifteenth International Conference on Machine Learning*, 1998.

[BPZ00] Sharad Bhojnagarwala, Michael Sao Pedro, and Zachary Zebrowski. Filtering greeting cards @ sparks.com. Major qualifying project, Computer Science Department Worcester Polytechnic Institute, 2000.

[CGM99] Mark Claypool, Anuja Gokhale, and Tim Miranda. Combining content-based and collaborative filters in an online newspaper. In *ACM SIGIR Workshop on Recommender Systems - Implementation and Evaluation*, 1999.

[FBH00] Xiaobin Fu, Jay Budzik, and Kristian J. Hammond. Mining navigation history for recommendation. In *The 2000 International Conference on Intelligent User Interfaces*, 2000.

[Her98] Jon Herlocker. Grouplens research project at the university of minnesota. http://www.cs.umn.edu/Research/GroupLens/data/, 1998.

[HKBR99] J. Herlocker, J. Konstan, A. Borchers, and J. Riedl. Framework for performing collaborative filtering. In *Proceedings of the 1999 Conference on Research and Development in Information Retrieval*, 1999.

[LAR02] W. Lin, S.A. Alvarez, and C. Ruiz. Efficient adaptive–support association rule mining for recommender systems. *Data Mining and Knowledge Discovery Journal*, January 2002. See also [Lin00].

[LHM98] Bing Liu, Wynne Hsu, and Yiming Ma. Integrating classification and association rule mining. In *Proceedings of the Fourth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1998.

[Lin00] Weiyang Lin. Association rule mining for collaborative recommender systems. Master's thesis, Computer Science Department Worcester Polytechnic Institute, 2000.

[McJ97] P. McJones. EachMovie collaborative filtering data set. http://www.research.compaq.com/SRC/eachmovie, 1997. Compaq Systems Research Center.

[Mit97] Tom M. Mitchell. *Machine Learning.* McGraw-Hill, 1997.

[MLW$^+$00] Yiming Ma, Bing Liu, Ching Kian Wong, Philip S. Yu, and Shuik Ming Lee. Targeting the right students using data mining. In *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2000.

[Qui92] J.R. Quinlan. *C4.5: program for machine learning.* Morgan Kaufmann, 1992.

[RNM$^+$94] P. Resnick, I. Neophytos, S. Mitesh, P. Bergstrom, and J. Riedl. Grouplens: An open architecture for collaborative filtering of netnews. In *Proceedings of CSCW94: Conference on Computer Supported Cooperative Work*, 1994.

[SKR01] J. Ben Schafer, Joesph Konstan, and John Riedl. E-commerce recommendation applications. *Data Mining and Knowledge Discovery*, pages 115–153, 2001.

[SN98] Ian M. Soboroff and Charles K. Nicholas. Combining content and collaboration in text filtering. Technical report, Department of Computer Science and Electrical Engineering University of Maryland, Baltimore County, 1998.

[UF98] Lyle H. Ungar and Dean P. Foster. Clustering methods for collaborative filtering. Technical report, CIS Dept. and Dept. of Statistics University of Pennsylvania, 1998.

[Wie98] Gio Wiederhold. University of California at Irvine KDD Archive: Movies Database. http://kdd.ics.uci.edu/databases/movies/movies.html, 1998.