BIPARTITE GRAPH SAMPLING METHODS FOR SAMPLING RECOMMENDATION DATA

Zan Huang
Supply Chain and Information Systems
Smeal College of Business
Pennsylvania State University
zanhuan@psu.edu

Abstract
Sampling is the common practice involved in academic and industry efforts on recommendation algorithm evaluation and selection. Experimental analysis often uses a subset of the entire user-item interaction data available in the operational recommender system, often derived by including all transactions associated with a subset of uniformly randomly selected users. Our paper formally studies the sampling problem for recommendation to understand to what extent population-based algorithm evaluation results correspond with sample-based results using different sampling methods. We use a bipartite graph to represent the key input data of user-item interaction for recommendation algorithms and build on the literature on unipartite graph sampling to develop sampling methods for our context of bipartite graph sampling. We also developed several metrics for assessing the quality of a given sample, including performance recovery and ranking recovery measures for assessing both single-sample and multiple-sample recovery performances. Based on the empirical results from two real-world datasets we provide some general recommendations for sampling for recommendation algorithm evaluation.

Keywords: Graph sampling, recommender systems, algorithm evaluation and selection

1. Introduction
Significant academic efforts have been devoted to recommender system-related research since the concept of collaborative filtering, which makes recommendations only using the user-item interaction data, was first implemented and popularized around 1995. There is also ever increasing industry interest and efforts in improving the performance of real-world recommender systems, highlighted by the Netflix Prize launched in October (Bennett et al. 2007). A wide range of recommendation algorithms have been developed and numerous evaluation studies have been conducted to compare the performances of these algorithms on different recommendation datasets. Many evaluative studies (e.g. Deshpande et al. 2004; Huang et al. 2007) have collectively revealed that there seems to be no single best algorithm and relative performances of different algorithms are largely domain- and data-dependent. For practitioners who need to select the best algorithm to be applied to an operational recommender system, there is the need to assess a wide range of algorithms on their data and choose the best. This requires numerous runs of the recommendation data resulted from potentially very large set of candidate algorithms. For designing new algorithms, numerous runs of the data on different versions of the new algorithm and benchmark algorithms are also needed.

As recommender systems are being increasingly used in many application domains and become a key competitive technology for many retailers and service providers, the scale of these systems increase continuously as new users and items enter the system and user-item interactions cumulate over time. The sheer size of recommendation data makes it prohibitively expensive and time-consuming to perform algorithm evaluation on the entire dataset. For such large datasets of companies like Netflix and Amazon, it is unacceptably time-consuming to assess the performance of even relatively simple algorithms such as the commonly used user-based or item-based neighborhood algorithms, not to mention those more complex model-based algorithms that require computationally intensive model estimation and prediction.

The common practice is to evaluate algorithms on a sample of the population data. The 100 million ratings Netflix provided for the Netflix Prize Competition is a sample of about 480 thousand randomly-
chosen users from the population data. Most research papers that use the Netflix dataset typically use a subsample of that sample. Other datasets used in the literature are also samples of population data in almost all cases. Although using the sample data has long been the standard practice, there is no formal study on sampling of the recommendation data. Creating a sample by including a random subset of users and all items they have interacted with (rating, purchase, browsing, etc.) seems to be the most common approach to obtaining the sample data. The understanding on how well the algorithm evaluation results obtained from such a sample correspond to the results would be obtained from the population data is lacking. There are many alternative approaches for obtaining the sample. A straightforward second approach would be selecting a random subset of items and include all users interact with them. There is no formal investigation into these sampling methods and the quality of the samples the produce.

This paper formally studies the sampling problem for recommendation. Building on the literature on network/graph sampling, we investigate a number of methods to sample the bipartite graph representing the user-item interaction data. We also introduce various measures to assess the correspondence of a sample with the population data with respect to recommendation algorithm performances. In this study, we focus on collaborative filtering algorithms that only use user-interaction data (the bipartite graph itself without the user/item attributes). We also focus on transaction-based recommendation where recommendations are made based on observed transactions without rating information.

2. Background and Related Work

Subgraph sampling studies date back to 1960s in mathematics and sociology. Many sampling methods were introduced and analyzed since then. These methods primarily focus on unipartite graphs where there is only one type of nodes and links are possible between any pair of nodes. Frank (Frank 1978) studied the random node sampling method, which obtains a random subset of nodes and includes all edges among these selected nodes to form the sample graph. Capobianco and Frank (Capobianco et al. 1982) studied a commonly used sampling method for studying social networks in sociology literature, the star or ego-centric sampling method. Under this method, seed nodes are randomly sampled from the population and then all neighbors of the seeds and the seed-neighbor and neighbor-neighbor links are included to form the sample graph. Another commonly used sampling method in sociology studies called snowball or chain-referral sampling was studied in (Goodman 1961). Under this approach, a seed node is selected and the population graph is traversed in a breadth-first fashion (including all neighbors of the seed and then all neighbors of the neighbors of the seed and so forth). Klovdahl (Klovdahl 1977) studied the method of random walk sampling, which received substantial interest in follow-up studies. It differs from snowball sampling in that only one neighbor of the current node is randomly chosen into the sample.

Several recent studies have addressed the graph sampling problem on large-scale graphs with a focus on recovering more complex graph topological characteristics such as degree distribution, path length, and clustering. Many of these studies focus on a single graph characteristic or focus on just a few sampling methods (e.g., (Lee et al. 2006)). Several recent studies provide comprehensive assessment of a wide range of sampling methods for recovering many graph topological characteristics. Leskovec and Faloutsos (Leskovec et al. 2006) evaluate the samples generated from a large number of sampling methods on a number of real-world networks. The sampling methods they studied include random node selection, random edge selection, random walk, random jump, snowball, and a new method they designed called forest fire sampling.

3. Sampling Recommendation Data

Given a population recommendation data, we define a bipartite graph $G = (C, P, E)$ with the user node set $C = \{c_1, ..., c_M\}$, item node set $P = \{p_1, ..., p_N\}$, and edge set $E = \{<c_i, p_j>\}$ where $c_i \in C$ and $p_j \in P$. The recommendation sampling problem is to find what sampling method produces a subgraph of $G$ that provides best correspondence between the sample-based algorithm evaluation results and population-based results. Sample size in the graph sampling literature typically refers to the number of nodes included in the sample graph. In our context of sampling for recommendation, we chose to define sample size as the number of edges (or user-item interactions) in the sample bipartite graph. Under this definition the samples of the same size have comparable file size under the relational representation.
The main departure from unipartite graph sampling in our study is that we have two sets of nodes to sample from, users and items. For the current paper, we study three types of sampling methods: user-oriented, item-oriented, and edge-oriented. Under the user-oriented sampling, we identify a subset \( C' \) of users from the population user set and include all items interacting with the selected users to form the sample such that the sample contains \( \lambda|E| \) edges where \( \lambda \) is the pre-specified sampling ratio. Under the graph representation, this approach is to include \( C' \), links from nodes in \( C' \) and item nodes reached by these links. Similarly, under the item-oriented sampling, we identify a subset \( P' \) of items from the population item set and include all users interacting with the selected items to form the sample such that the sample contains \( \lambda|E| \) edges. Under the edge-oriented sampling, we select a subset \( E' \) of the edges from the population edge set with size \( \lambda|E| \) and include all user and item nodes incident on these edges.

In this paper, we study the following sampling methods: random, random walk, snowball, and forest fire sampling methods under user-oriented, item-oriented, and edge-oriented sampling.

Random sampling: Under random sampling the subset of users/items/edges is selected uniformly randomly from the population user/item/edge set under user-oriented/item-oriented/edge-oriented sampling. Since we try to obtain a sample containing a specific number of edges, under the user-oriented (item-oriented) sampling, we include one user (item) at a time and all user-item edges involved. When adding a user (item) results in more edges than needed, a random subset of user-item edges involved with that user (item) is included into the sample. The recommendation data used for Netflix Prize Competition is obtained through user-oriented random sampling in our terms.

Graph exploration sampling: Under methods in this approach, we select the nodes to include in the sample by navigating the population network following the links from a randomly selected starting seed node, \( s \). Under the user-oriented (item-oriented) sampling, we operate this graph navigation on the projected user (item) graphs. A user (item) graph projected from the user-item graph forms an edge between two users (items) if the two users (items) interact with at least one common item (user). Being consistent with the user-oriented (item-oriented) random sampling, we include one user (item) at a time and the user-item edge involved until reaching the sample size. Under the edge-oriented sampling, the graph navigation is operated on the population user-item graph. Specifically, graph navigation sampling works as follows. Starting from \( s \), these methods select either a subset or the entire set of unselected neighbors of \( s \) and repeat this process for each node just included into the sample. If the desired number of edges cannot be reached (i.e., the exploration process has fallen into a small component of the population graph isolated from others) a new seed node is selected at random to re-start the exploration.

There are several variations of the graph-exploration methods depending on how we select the neighbors of a node \( v \) that is reached via the exploration process. The snowball method includes all unselected neighbors of \( v \). The random walk method selects exactly one neighbor uniformly at random from all the unselected neighbors. These two extremes can also be understood as the well-studied breadth-first and depth-first searches from the graph search literature respectively. Between these two extremes is the forest fire method (Leskovec et al. 2006), which selects \( x \) unselected neighbors of \( v \) uniformly randomly. In our study we set this number \( x \) to be uniformly drawn from \( [1, 2, 3] \). In our future study we will investigate additional settings for \( x \). When implementing these methods, due to the nature of the snowball method, we can simply forbid revisiting the nodes. For random walk and forest fire, nodes need to be revisited for the exploration to carry on. In order to avoid being stuck in a small component of the population graph, we set the exploration process to jump to another randomly chosen seed with a probability of 0.05.

4. Evaluation of Recommendation Samples

In the context of our study, the quality of a sample should reveal the extent to which the sample-based recommendation algorithm evaluation results align with the population-based results. Given a recommendation algorithm performance measure of interest, we consider two types of sample quality measures, performance recovery measure and ranking recovery measures. We denote the chosen performance measure as \( m \) and the measures obtained using a particular algorithm \( j \) on the sample \( s \), and population data as \( m_j(s) \) and \( m_j(p) \), respectively. The performance recovery measure for measure \( m \),
algorithm \( j \), and sample \( i \) is straightforwardly defined as: \( \text{prm}_{m,j,i} = m_j(s_i) - m_j(p) \). The ranking recovery measure reflects the alignment between the ranking of the relative performances of \( J \) algorithms considered for the sample and for the population data. We denote the ranks of algorithm \( j \) in terms of measure \( m \) for the population \( p \) and sample \( s_i \) as \( r_{j,m}(p) \) and \( r_{j,m}(s_i) \), respectively. The ranking recovery measure for measure \( m \), and sample \( i \) is defined as: \( r_{m,j,i} = \sum_j (r_{j,m}(s_i) - r_{j,m}(p))^2 \). Under this definition, we penalize large ranking differences. For example, suppose the ranking lists of four algorithms for population data is [1,2,3,4] and two samples produce ranking lists of [2,1,4,3] and [2,3,4,1], the ranking recovery measures for the two samples will be 4 and 12, respectively. We consider the first sample recovers the rankings relatively better. For four algorithms, the ranking recovery measure ranges from 0 to 20. It may be argued that in the previous example, the second sample recovers the ranking better as the ranking among the first three algorithms are the same as the population results. We leave investigation on other definitions of ranking quality to future work.

For given sample ratio and sample method, the quality of the sample may vary across different instances of the samples obtained. In order to estimate the quality of single sample, we use multiple samples in the experiment and use the average recovery measures across the samples to obtain an estimate of the expected recovery measures. For the performance recovery measure, we define mean absolute performance recovery measure for method \( m \) and algorithm \( j \) as: 
\[
\text{mapr}_{m,j} = \sum_i \left| m_j(s_i) - m_j(p) \right| / I,
\]
where \( I \) is the number of samples. In order to reflect the quality of each given sample rather than the mean across multiple sample instances, we may also look at the worst apr among the multiple samples obtained or a percentile apr measure that take the variance of multiple samples into consideration.

When multiple samples are available, there is yet another way to recover the population evaluation results from the results on multiple samples. For performance recovery measure, we can average across multiple samples and use the mean performance recovery measure. This makes intuitive sense as we expect the mean to have smaller variance. Note this measure represents how a set of samples, rather than a single sample, can recover the population evaluation results. Specifically, we define the mean performance recovery measure for method \( m \) and algorithm \( j \) as: 
\[
\text{mpr}_{m,j} = \sum_i \frac{m_j(s_i) - m_j(p)}{I}.
\]
We can similarly obtain the ranking recovery measure based on multiple samples. We may obtain the ranking list based on the mean performance recovery measure across multiple samples, which we denote as ranking of mean recovery measure, \( \text{rmr}_{m} \).

5. Experimental Study

We used two datasets in our experimental study: a retail dataset provided by a leading U.S. online clothing merchant and a sample of movie rating data from MovieLense project. The details about the datasets we used are shown in Table 1. The movie dataset is treated as transaction data and our recommendation is on the “Who Rated What” task.

<table>
<thead>
<tr>
<th>Dataset</th>
<th># of Consumers</th>
<th># of Products</th>
<th># of Transactions</th>
<th>Density Level</th>
<th>Avg. # of purchases per consumer</th>
<th>Avg. sales per product</th>
</tr>
</thead>
<tbody>
<tr>
<td>Retail</td>
<td>1,000</td>
<td>7,328</td>
<td>9,332</td>
<td>0.13%</td>
<td>9.33</td>
<td>1.27</td>
</tr>
<tr>
<td>Movie</td>
<td>1,000</td>
<td>2,900</td>
<td>50,748</td>
<td>1.75%</td>
<td>50.75</td>
<td>17.5</td>
</tr>
</tbody>
</table>

For algorithm performance evaluation on the population datasets, we first randomly selected 2/3 of the data into the training set. Within the remaining 1/3 of the transactions, only those involving users and items appeared in the training set were included into the testing set. We applied the user-based and item-based neighborhood algorithm, spreading activation algorithm, and generative model algorithm and compute the precision, recall, F, rank score, and AUC measures Huang, 2007 #152]. We have set the number of recommendations for each user to be 10 in our study. Table 2 shows the population algorithm evaluation results. With a given sampling ratio we obtain 20 samples using random (\( \text{rand} \)), random walk (\( \text{rw} \)), snowball (\( \text{sb} \)), and forest fire (\( \text{ff} \)) methods under the user-oriented, item-oriented, and edge-oriented sampling. With each sample data, we follow exactly the same procedure as the population data to obtain
training and testing sets and obtain algorithm evaluation results with the four recommendation algorithms. We then obtain the sample quality measures introduced earlier for comparison. In our study we focus on relatively small sampling ratios as the performance of sampling methods for small samples are most interesting. Specifically, we studied samples of sizes 4%, 6%, 8%, 10%, and 20%.

Figure 1: Recovery measures

![Graphs showing recovery measures for retail and movie datasets.](image)

(a) Retail dataset

(b) Movie dataset

Figure 2 shows the sampling results for the retail and movie datasets. We summarize the main findings from these results below. We observe that the relative performances of individual sampling methods from the results on the two datasets we studied are far from a clear picture. The best sampling method varies with respect to dataset, performance measure interested (F or AUC), and whether performance recovery or ranking recovery is the goal. We observe quite different patterns of relative performance of the sampling methods for the retail and movie datasets. While further investigation is needed on more datasets, we conjecture that the different results we see on the retail and movie datasets may be due to the very different sparsity level of the two datasets.

Comparing the mapr and mnr measures with mpr and mnr measures we see that using mean from multiple samples generally helps to recover performance measure and this is even more so the case for
recovering ranking of algorithms. For example, for the F measure of the movie dataset, we see that the
four user-oriented sampling methods delivered best result for mean rank recovery measures and using
mean of multiple samples help these methods to reach perfect recovery of ranking fairly soon. All four
methods had perfect ranking of mean recovery measure for sample sizes greater than 6% and user
oriented random walk even had perfect ranking of mean recovery measure for sample size of 4%.

User-oriented random sampling is used in almost all academic studies and practical research on
recommendation algorithm evaluation (include the Netflix Prize dataset). Based on our results, this
sampling method is far from the overall best. Particularly for the retail dataset, it is among the two or four
worst methods for the four recovery measures for individual samples. When using mean from multiple
samples, it had the best performance for recovery of F measure and ranking based on F measure, medium
performance for recovery of AUC measure and worst performance for recovery of ranking based on AUC
measure. For the movie dataset, using user-oriented random sampling is acceptable for recovery of F
measure and ranking based on F or AUC measure but not for recovery of AUC measure using single
samples or mean from multiple samples.

6. Conclusion and Future Research
To the best of our knowledge, there is no prior formal study attempted to understand to what extent
sample-based algorithm evaluation results correspond with population-based results. Our paper introduces
this sampling problem for recommendation. We use a bipartite graph to represent the key input data of
user-item interactions for recommendation algorithms and build on the literature on graph sampling to
develop our analytical framework. We adapted the sampling methods reported to be of good quality on
unipartite graph to our context of bipartite graph sampling. The sampling methods included in our study
are random, random walk, snowball, and forest fire methods under user-oriented, item-oriented, and edge-
oriented sampling. We also developed a series of metrics of the quality of a given sample with respect to
recommendation algorithm evaluation and selection. These metrics include performance recovery and
ranking recovery measures for assessing both single-sample and multiple-sample recovery performances.
Our key findings from preliminary empirical results are that for population data with different density
level different sampling methods are recommended: edge-oriented forest fire or edge-oriented random
walk for sparse data and user-oriented methods for dense data and that using mean from multiple smaller
samples often works better than using a single large sample. The current work is limited in several
aspects, each of which points to need for extensive future research. Firstly, we need to further our
understanding on why certain sampling methods excelled for certain algorithm evaluation goals.
Secondly, we need to use additional datasets of different characteristics and different scale to evaluate the
generalizability of our current conclusions. Thirdly, we also want to expand our study to include more
recommendation algorithms, to investigate the cases of rating-based collaborative filtering
recommendation and even hybrid recommendation methods that also utilize user/item attributes.

References
Deshpande, M., and Karypis, G. "Item-based top-N recommendation algorithms," ACM Transactions on
Huang, Z., Zeng, D., and Chen, H. "A comparative study of recommendation algorithms for e-commerce
Klovdahl, A.S. "Social networks in an urban area: First Canberra study," Australian and New Zealand Journal of
Sociology (13) 1977, pp 169-175.
Leskovec, J., and Faloutsos, C. "Sampling from large graphs," 12th ACM SIGKDD international conference on
Optimizing Offer Sets Based on User Profiles

Pelin Atahan, Monica Johar, and Sumit Sarkar
Özyeğin University, University of North Carolina at Charlotte, University of Texas at Dallas
pelin.atahan@ozyegin.edu.tr, msjohar@uncc.edu, sumit@utdallas.edu

Abstract

Personalization and recommendation systems are being increasingly utilized by ecommerce firms to provide personalized product offerings to visitors at the firm’s website. These systems often recommend, at each interaction, multiple items (referred to as an offer set) that might be of interest to a visitor. When making recommendations, firms typically attempt to maximize their expected payoffs from the offer set. This paper examines how a firm can maximize its expected payoffs by leveraging the knowledge of the profiles of visitors to their site. We provide a methodology that accounts for the interactions among items in an offer set in order to determine the expected payoff. Identifying the optimal offer set is a difficult problem when the number of candidate items to recommend is large. We develop an efficient heuristic for this problem, and show that it performs well for both small and large problem instances.

Keywords: Personalization, recommendation, e-commerce, probability theory

1. Introduction

Effective personalization can help firms reduce their customers’ search costs and enhance customer loyalty. This, in turn, translates into increased cash inflows and enhanced profitability (Ansari and Mela 2003). Extant research has shown that in electronic shopping environments, personalized product recommendation enable customers to identify superior products with less effort (Häubl and Trifts 2000). These works have demonstrated that personalization can be an effective tool for firms.

The personalization process consists of two important activities, learning and matching. Learning involves collecting data from a customer’s interactions with the firm and then making inferences from the data about the customer’s profile. For instance, the relevant profile for a customer may be her membership in one of several possible demographic or psychographic segments, which could be based on age, gender, zip code, income, political beliefs, etc. (Montgomery et al. 2001, Wall Street Journal October 17 2007). Matching is the process of identifying products to recommend based on what is known about the customer’s profile. Naturally, the quality of a customer’s profile should impact the ability of the firm to provide high quality recommendations targeted towards sales (viz., the matching ability).

In this research, we examine how a firm can maximize its expected payoffs when making recommendations to users by leveraging the knowledge of the profiles of visitors to the site. In order to identify the best set of items to offer (e.g., links to a set of recommended items on a page that we call the offer set), a firm would first need a methodology to evaluate the expected payoff given an offer set. Then, the optimal offer set can be determined by selecting the set of items that maximizes the expected payoff for each page requested by the visitor based on what the firm knows about the visitor’s item history (denoted by IH) and the profile. To evaluate the expected payoffs from an offer set the firm would need to evaluate the likelihood of each offered item being viewed and eventually purchased. The probability that an item will be viewed when provided in an offer set depends not only on the probability parameters associated with the item itself, but also on the other items in the offer set. Therefore, the interaction among items in an offer set should be accounted for when evaluating the expected payoffs from that offer set.

Extant literature has not formally analyzed the impact of the composition of an offer set on the resultant expected payoffs. Existing approaches that consider multiple recommendations typically sort association rules by some criteria like confidence or lift and simply take the top n items to recommend (Huang et al.
A novelty of the proposed approach is that it explicitly studies the impact of an item in the offer set on the probability of other items in the offer set being viewed and ultimately purchased when calculating the expected payoffs from that offer set.

In the next section, we present the framework to evaluate the expected payoffs from an offer set. A firm can evaluate all feasible offer sets using this framework and select the optimal one. We present in Section 3 an efficient heuristic approach to determine the offer sets quickly when the number of sets to evaluate is large. Section 4 discusses experiments to evaluate the performance of the proposed approach. Concluding remarks are provided in Section 5.

2. Evaluating the Expected Payoff from an Offer Set

The interactions between a visitor and the site are iterative in nature, with the firm providing a new offer set at each interaction (i.e., each time the visitor makes a page request). Figure 1 shows the choices faced by the visitor when provided with an offer set.

Given an offer set, the visitor may either view detailed information on one of the offered items or ignore the offer set. When the visitor views information on one of the items (say \(i_j\)) by clicking on the appropriate link, the site provides detailed product information for item \(i_j\), along with a new offer set (i.e., a new set of recommendations) in case the visitor does not like the product. If, on viewing the information on item \(i_j\), the visitor decides to purchase that item, it results in a payoff to the firm. If the visitor does not purchase that item, then the visitor has the option of selecting an item from the new offer set for further evaluation, and the process repeats.

A visitor’s decisions are driven by the visitor’s profile and the items previously viewed by the visitor. A visitor’s profile is represented by the set of possible classes (\(a_i\)) the visitor may be a member of, accompanied by the probability associated with each class. At any point in time, the visitor’s item history is known to the site; and the site can drive a probability distribution of the visitor’s profile information given the visitor’s item history, i.e., the probability \(P(a_i|IH)\) for each \(a_i\) (details of the belief revision process are suppressed for lack of space). To estimate the probability that a given visitor purchases an offered item \(i_j\), the site needs to estimate the joint probability distribution of the visitor viewing the item (\(v_j\)), purchasing the item (\(s_j\)) and the visitor’s profile, i.e., the site needs the joint probability \(P(s_j, v_j, a_i|IH)\) for each \(a_i\). This probability can be expressed as:

\[
P(s_j, v_j, a_i|IH) = P(s_j|IH, v_j, a_i)P(v_j|IH, a_i)P(a_i|IH).
\]

Given an offer set \((O)\) and the knowledge about the visitor’s profile, the firm can calculate the expected payoff from that offer set \(EP(O)\) in the following manner:

\[
EP(O) = \sum_{a_i} \sum_{i_j \in O} P(s_j|IH, a_j, v_j, O)P(v_j|IH, a_i, O)P(a_i|IH)\omega_j.
\]

\(\omega_j\) is the profit realized from sales of item \(i_j\). To simplify the exposition the profit from each item is assumed to be the same and equal to 1; we should point out that our approach can accommodate differentiated values for \(\omega_j\).
To operationalize this framework, the firm would need to estimate the following probability parameters associated with the choices made by the visitor.

- The probability that a visitor associated with a given profile and item history will view item $i_j$ when presented with an offer set $O=\{i_1, \ldots, i_n\}$, i.e., $P(v|IH, a_i, O)$.
- The probability that such a visitor will purchase item $i_j$ after viewing information on that item, i.e., $P(s_j|IH, v_j, a_i, O)$.

One approach to obtain the necessary parameters is by directly estimating them based on the historical data on customer interactions at that site. While that could be feasible for some of the above parameters, it would be very difficult for others because the number of feasible item histories and offer sets would be typically very large.

To help estimate the probability that a visitor will view information about an item that is part of the offer set we consider the use of association rules. For example, if a user has viewed items $i_1$ and $i_2$ and there exists a rule of the form \{ $v_1, v_2$ $\Rightarrow v_j$ $\}$ this rule would provide $P(v_j|v_1, v_2)$.

To leverage the profile information of its visitors when making recommendations, a firm would need profile specific probabilities associated with user actions. For instance, to make gender specific recommendations a firm would need probabilities associated with male ($m$) and female ($f$) visitors’ decisions to view and to purchase each item. For example, for the aforementioned rule, the firm would need the probabilities item $i_1$ will be viewed by male and female visitors who have previously viewed items $i_1$ and $i_2$, i.e., $P(v_j|v_1, v_2, m)$ and $P(v_j|v_1, v_2, f)$.

Using the data from site’s log files, the firm can also estimate the probability associated with item $i_j$ being purchased by male and female visitors who have viewed the items in the rule antecedent, i.e., $P(s_j|v_1, v_2, m)$ and $P(s_j|v_1, v_2, f)$. The probability of purchasing item $i_j$ is assumed to be independent of the other offered items conditioned on the visitor’s class, visitor item history and the fact that the item has been viewed, i.e., $P(s_j|IH, v_j, a_i, O)=P(s_j|IH, v_j, a_i)$.

We next illustrate using an example how a site can estimate the probability that a visitor with a specific profile (e.g., gender) will view an item that is part of the offer set. The firm chooses a set of items to offer to a visitor based on the visitor’s item history; the item history can be used to identify the eligible rules and the current belief about the visitor’s gender. A rule is considered to be eligible, if its antecedent is a subset of the visitor’s item history and the consequent is not a subset of the visitor’s item history. We first discuss the methodology where there exist several rules with antecedents that match the visitor’s item history. The situation where antecedents of rules are proper subsets of the visitor’s item history is similar and discussed later.

Imagine that the firm has two eligible rules $R_1$: $IH \Rightarrow v_1$ and $R_2$: $IH \Rightarrow v_2$, and is considering offering $i_1$ and $i_2$. The site would need to determine the probability the visitor will view either of the offered items or ignore the offer set. The class specific probabilities associated with these rules are $P(v_1|IH, m)$, $P(v_1|IH, f)$, $P(v_2|IH, m)$, and $P(v_2|IH, f)$. A male visitor’s likelihood of viewing information on item $i_1$ when presented in offer set $O$ is the probability $P(v_1|IH, O, m)$. Assuming that the visitor views one of the two items (event $V$), the probability that the user will view item $i_1$ is

$$P(v_1|IH, m, O, V) = \frac{P(v_1|IH, m)}{P(v_1|IH, m) + P(v_2|IH, m)} = \frac{P(v_1|IH, m)}{P(v_1|IH, m) + P(v_2|IH, m)}.$$

---

1 A firm can estimate profile specific probabilities from the profile information of a subset of its users. Such data is available from market research agencies like comScore or AC Nielsen which collect personal information from a large panel of users and track their online behavior.
Consequently, \( P(v_2|IH,m,O,V) = 1 - P(v_1|IH,m,O,V) \). The corresponding probabilities for female visitors are obtained analogously.

We next consider the situation where the visitor does not view either item. Based on the available rules, the firm knows the probability that each item is of interest to a male user with item history \( IH \), i.e., \( P(v_1|IH,m) \) and \( P(v_2|IH,m) \). Then, the probability that a male visitor with item history \( IH \) is not interested in either of the items offered, denoted \( P(\emptyset|IH,m,O) \), can be estimated as follows:

\[
P(\emptyset|IH,m,O) = 1 - P(v_1|IH,m) - P(v_2|IH,m) + P(v_1,v_2|IH,m).
\]

Using the chain rule, the term \( P(v_1,v_2|IH,m) \) can be written as \( P(v_1|v_2,IH,m)*P(v_2|IH,m) \) or as \( P(v_2|v_1,IH,m)*P(v_1|IH,m) \). The joint probability \( P(v_1,v_2|IH,m) \) can be calculated directly if the firm has rules of the form \( \{IH,v_1\} \Rightarrow v_2 \) or \( \{IH,v_2\} \Rightarrow v_1 \). If neither of these rules is available, this implies a weak dependency between the two items given the profile and the item history. In that case, it is reasonable to assume the probability of viewing item \( i_1 \) is independent of the probability of viewing item \( i_2 \) conditioned on the profile and the item history, i.e., \( P(v_1|v_2,IH,m) = P(v_1|IH,m) \). We then have

\[
P(v_1,v_2|IH,m) = P(v_1|IH,m)*P(v_2|IH,m).
\]

The probability that a male user with item history \( IH \) will view an offered link is then \( P(V|IH,m,O) = 1 - P(\emptyset|IH,m,O) \). The unconditional probability that a male visitor with item history \( IH \) will view item \( i_1 \) (i.e., without assuming the visitor must view an offered item), \( P(v_1|IH,m,O) \), is then obtained as

\[
P(v_1|IH,m,O) = P(v_1|IH,m,O,V)*P(V|IH,m,O).
\]

The probability \( P(V|IH,m,O) \) can be estimated similarly. The above analysis easily extends to offer sets comprising of any number of items, and for profile attributes that can take any number of values.

As the size of the item history increases it will be difficult to find association rules with antecedents that perfectly match the entire item history. However, there will usually exist many eligible rules when the item history is large. The firm can then consider for inclusion in the offer set the consequents of eligible rules which have maximal antecedents (an antecedent is maximal if there does not exist another eligible rule whose antecedent is a superset of the target rule’s antecedent). The rest of the procedure will remain unchanged.

3. Determining the Optimal Offer Set

The firm’s objective is to select the offer set (including a predetermined number of items \( n \)) that maximizes its expected payoff. The items are chosen from consequents of eligible rules at each interaction. An obvious way to identify the offer set that maximizes the firm’s expected payoffs would be to evaluate all feasible offer sets and then provide the offer set that leads to the highest expected payoff. However, when the number of items for consideration is large it may not be feasible to evaluate all possible offer sets in real time. We develop an efficient heuristic approach to determine the offer set in such situations.

3.1 Algorithm to Determine Offer Sets

Our approach selects items to include in the offer set in an iterative manner. It identifies items that have high probability of being viewed and purchased by visitors of each class, so that they contribute highly to the expected payoff from the corresponding class. It creates as many lists as the number of classes, where each list includes items more likely to be viewed by members of that class, i.e., items for which \( P(a_i|IH,v_i) > P(a_i|IH) \). Then, items in each of the lists are sorted by their item value. An item’s value for a given class is calculated by the product of an item’s likelihood of being viewed and purchased by members of that class, i.e., item value of \( i_j \) in the list associated with class \( a_i \) is calculated as \( P(s_j|IH,v_i,a_i)*P(v_i|IH,a_i) \). The algorithm then compares the expected payoffs from offer sets that are created by adding the highest contributing item from each class. When comparing expected payoffs it disregards the likelihood of a user ignoring the entire offer set. Otherwise, the algorithm would be overly biased in earlier iterations to select links that have a high probability of being viewed.
4. Experiments

To validate our approach we have performed simulated experiments (we do not have access to real world data). We use expected payoffs from the identified offer sets as a measure of performance. We compare the performance of the proposed approach with that of the optimal offer set for many problem instances.

In our experiments, we used a binary class attribute for a visitor’s profile. To generate the probabilities associated with a member of a class viewing an item, we generated the distribution of the profile of visitors $P(a_i|v_j)$ who view each item and each item’s overall popularity $P(v_j)$ based on uniform distributions. We then obtained probabilities associated with each item being viewed by members of a specific class assuming a population prior $P(a_i)=P(a_i)=0.5$.

We expect the profiles of visitors who view an item to be correlated with the profile of visitors who purchase that item. The probabilities associated with purchasing the item $P(s_j|a_i,v_j)$ were generated by mixing a uniform distribution with the distribution of $P(v_j|a_i)$ associated with members of that class viewing the item where specific levels of correlation were created between the two probabilities. Then, the purchase probabilities were normalized to be between 0 and 0.3. We performed experiments on different datasets which had correlation levels of 0.6, 0.8 and 1.

To determine the optimal solution, we evaluated the expected payoffs from all possible offer sets and select the one that provides the highest expected payoff. In these experiments, the cardinality of the offer set is 8 and there are 40 candidate items. This leads to 76,904,685 possible offer sets to evaluate. We randomly generated 5 different datasets for each correlation level considered. On each dataset, for a given profile distribution, the proposed approach was implemented first to determine the offer set. Then each possible offer set was enumerated. The expected payoff from the offer set identified by the proposed approach was compared with the expected payoff from each of the other offer sets. We recorded the rank of the expected payoff from this offer set compared to all other offer sets and the percentage difference of the expected payoff from this offer set from that of the optimal offer set. We repeated the experiments on the same dataset for 11 different user profile distributions (profile probability for one class ranging from 0 to 1 in increments of 0.1). Then we conducted the same set of experiments on the datasets for each correlation level.

<table>
<thead>
<tr>
<th>User Profile Probability</th>
<th>Rank</th>
<th>Difference in Expected Payoff</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Correlation Level</td>
<td>0.6</td>
</tr>
<tr>
<td>0</td>
<td>25,652</td>
<td>22,326</td>
</tr>
<tr>
<td>0.1</td>
<td>3,186</td>
<td>3,572</td>
</tr>
<tr>
<td>0.2</td>
<td>1,028</td>
<td>1,391</td>
</tr>
<tr>
<td>0.3</td>
<td>295</td>
<td>284</td>
</tr>
<tr>
<td>0.4</td>
<td>351</td>
<td>388</td>
</tr>
<tr>
<td>0.5</td>
<td>11</td>
<td>716</td>
</tr>
<tr>
<td>0.6</td>
<td>37</td>
<td>975</td>
</tr>
<tr>
<td>0.7</td>
<td>71</td>
<td>134</td>
</tr>
<tr>
<td>0.8</td>
<td>627</td>
<td>641</td>
</tr>
<tr>
<td>0.9</td>
<td>1,446</td>
<td>1,373</td>
</tr>
<tr>
<td>1</td>
<td>5,745</td>
<td>2,617</td>
</tr>
<tr>
<td>Average</td>
<td>3,495</td>
<td>3,129</td>
</tr>
</tbody>
</table>
The results of these five sets of experiments are averaged for each correlation level in Table 1. Each row reports, for a given profile distribution, the rank of the solution in terms of expected payoff provided by the proposed approach among all possible offer sets and the average percentage difference in expected payoffs between the proposed offer set and the optimal offer set. The last row provides the results averaged over the different correlations considered.

The proposed approach performs well in all the experiments. In the worst case, the rank of the solution provided by the proposed approach is 25,652, which is in the top 1% of all possible offer sets. The overall performance of the proposed approach is even better in the experiments where the correlation level is higher, e.g., when the correlation is 1, the rank of the solution is within the top ten (out of more than 76 million) in several of the experiments. The performance can be explained as follows. Our approach considers the potential value of an item for one class and ignores the potential value of the item for the other class. In some cases, instead of including an item with the highest potential value for one class, it may be more profitable to include an item that has slightly lower potential value for that class, but much higher potential value for the other class. In such situations, because the proposed approach will fail to identify and include such items, the expected payoffs for the solution provided by the proposed approach may deviate more from the optimal solution. When the correlation level is high having items valuable for both classes is less likely. Therefore, the items identified by the proposed approach are more likely to be the most valuable items to include and the expected payoffs for the solution from the proposed approach will be much closer to the optimal expected payoff.

The percentage difference in expected payoff between the solution provided by the proposed approach and the optimal solution is quite small in general. It is around 2% or less on average. The performance of the proposed approach degrades slightly compared to the optimal approach at more extreme user profiles.

5. Conclusion and Discussions

Firms typically make multiple recommendations to visitors traversing their sites. However, extant research has not addressed how the multiple items in an offer set impact each other’s view and purchase probabilities and hence a firm’s expected payoffs from an offer set. We study how a firm should compose the offer set to maximize its payoffs from the recommendations. The framework presented would allow the firm to select the offer set that maximizes its expected payoffs based on the visitor’s item history and the current beliefs regarding the visitor’s profile. We propose an efficient heuristic algorithm to determine the offer sets quickly when there are a large number of items that are considered for inclusion in the offer set. Simulated experiments demonstrate that the heuristic performs well compared to the optimal approach. Ongoing experiments (not reported here) show that the performance of the proposed approach can be markedly better compared to that of a benchmark approach.

References

RECOMMENDATIONS USING INFORMATION FROM MULTIPLE ASSOCIATION RULES

Abhijeet Ghoshal, Syam Menon, Sumit Sarkar
School of Management, University of Texas at Dallas, Richardson, TX, 75080
axg053200@utdallas.edu, syam@utdallas.edu, sumit@utdallas.edu

Abstract

Personalized recommendations enable firms to effectively target customers with products and services. Such systems are often based on association rules. While there has been considerable work done on mining rules more efficiently, there is very little prior research that examines how to use rules effectively when making recommendations. Traditional association rule-based recommendation systems have relied on identifying one rule from the several eligible ones to make the recommendation. This ignores information from other eligible rules that can potentially improve the recommendation. We propose a method to combine multiple rules when making recommendations. In doing so, we also present an approach to select the best combination of rules from the many that might be available.

Keywords: Personalization, Bayesian Estimation, Maximum Likelihood, Mutual Information

1. Introduction

As Internet based applications gain in popularity, firms are increasingly resorting to personalized recommendations to improve their customers' online shopping experience. Several studies have shown that personalized recommendations can enable firms to effectively target customers with products and services (Häubl and Trifts 2000, Tam and Ho 2003). Personalization has been found to be an effective tool for consumer marketing, building customer loyalty, improving merchandising and elevating the online customer experience (Lovett 2007). Recognizing the benefits of personalization, many firms are adopting technologies that provide useful recommendations.

Recommendation systems often use association rules (Hastie et. al. 2009). Association rules are implications of the form \{bread, milk\} → \{yogurt\}, where \{bread, milk\} is called the antecedent of the rule and \{yogurt\} is called its consequent. In a typical dataset, millions of such implications are possible. However, not all of them are useful for providing recommendations. Association rule mining (Agrawal et al. 1993) identifies those rules where the items in the rules appear in reasonably large numbers of transactions (termed the support of the rule), and where a consequent has a high probability of being chosen when the items in the antecedent have already been chosen (termed the confidence of the rule). Every mined rule must meet minimum thresholds for both support and confidence. Association rules compactly express how products group together (Berry and Linoff 2004). Further, they can be used unobtrusively in automated systems to provide recommendations to customers in real time.

When a customer is shopping online, the recommender engine identifies items to recommend based on the customer's basket and available rules. While there has been considerable work done on mining rules more efficiently, there is very little prior research that examines how to use rules effectively when making recommendations. Zaïane (2002) proposed a method that finds all rules whose antecedents are subsets of the basket and whose consequents are not (i.e., the eligible rules) and recommends the consequent of the eligible rule with the highest confidence. Wang and Shao (2004) suggest considering only eligible rules whose antecedents are maximal-matching subsets of the basket; we call such rules maximal rules.

---

1 An antecedent is maximal-matching if no superset of the antecedent is a subset of the basket.
All these approaches focus on identifying a single rule from the eligible ones to make the recommendation. The final recommendation, therefore, is often made on the basis of only a few items from the basket, as items not present in the antecedent of the rule being used for recommendation are ignored. If we consider the probability a customer will choose the recommended item given all the items in the basket, it is possible that the item to be recommended will be a different one. Interestingly, the set of eligible rules often contain multiple rules with the same consequent. The confidence associated with each such rule conveys information about the probability of the customer choosing the consequent. The quality of recommendations could be improved by combining such rules in an appropriate manner.

We propose a method to combine multiple rules in order to make recommendations based on as many items in the basket as possible. We assume that the rules have been mined beforehand. For every item that can potentially be recommended, we estimate the probability that it will be selected by the customer, given the basket. In order to do so, it is necessary to identify the best combination of rules from the many combinations that might exist, and we show how to identify such a combination. The consequent that has the highest estimated probability is recommended to the customer.

We describe the problem in detail in the next section. Section 3 discusses the proposed methodology. Section 4 presents results of experiments conducted to validate our approach.

2. Problem Description

We illustrate the problem using an example. Consider recommending an item to a customer who has three items $i_1, i_2$ and $i_3$ in her basket $B$. Table 1 shows the eligible rules in our example.

<table>
<thead>
<tr>
<th>Rule</th>
<th>Items in antecedent</th>
<th>Item in consequent</th>
<th>Confidence</th>
</tr>
</thead>
<tbody>
<tr>
<td>$R_1$</td>
<td>$i_1, i_2$</td>
<td>$x_1$</td>
<td>60%</td>
</tr>
<tr>
<td>$R_2$</td>
<td>$i_2, i_3$</td>
<td>$x_1$</td>
<td>40%</td>
</tr>
<tr>
<td>$R_3$</td>
<td>$i_1$</td>
<td>$x_1$</td>
<td>54%</td>
</tr>
<tr>
<td>$R_4$</td>
<td>$i_3$</td>
<td>$x_1$</td>
<td>45%</td>
</tr>
<tr>
<td>$R_5$</td>
<td>$i_1, i_3$</td>
<td>$x_2$</td>
<td>42%</td>
</tr>
<tr>
<td>$R_6$</td>
<td>$i_2, i_3$</td>
<td>$x_2$</td>
<td>50%</td>
</tr>
<tr>
<td>$R_7$</td>
<td>$i_1$</td>
<td>$x_2$</td>
<td>58%</td>
</tr>
<tr>
<td>$R_8$</td>
<td>$i_2$</td>
<td>$x_2$</td>
<td>62%</td>
</tr>
</tbody>
</table>

The approach proposed by Zaïane (2002) ranks the eligible rules based on their confidences, and the highest ranked rule is fired to make the recommendation. In our example, rule $R_8$ has the highest confidence, and therefore item $x_2$ would be recommended.

While Wang and Shao (2004) suggest using only maximal eligible rules, they do not discuss how to identify the specific maximal rule to be used for making the recommendation. One approach could be to use the maximal rule with the highest confidence. In that case, $R_1$ would be used to recommend $x_1$. Another possibility is to use the maximal rule that has the largest number of items in the antecedent, with ties being broken based on confidence. This approach would also result in rule $R_1$ being used.

In each approach, items not present in the antecedent of the fired rule get ignored. For example, recommending $x_2$ based on rule $R_8$ ignores the presence of $i_1$ and $i_3$ in the basket, even though another rule exists with antecedent containing $i_1$ and $i_3$ ($R_3$). Thus, even though the probability that $x_2$ will be chosen...
given i_1 and i_3 is known, this information is ignored in this approach. Similarly, if x_1 is recommended using R_1, the impact of i_3 is ignored even though it is the antecedent of R_4.

Our goal is to develop an approach that utilizes as much information from items in the basket as possible. If we could somehow combine multiple rules to cover as many items of the basket as possible, and estimate the probabilities of x_1 and x_2 for the complete basket before deciding which one to recommend, our recommendation would be more informed. To do this, the question is how to combine information in multiple rules to find the best item for recommendation.

Note that the antecedents of R_1 and R_4 together constitute the complete basket, as do the antecedents of R_2 and R_3. This raises another question. Which set of rules - \{R_1,R_4\} or \{R_2,R_3\} - should be used for estimating the probability of x_1? Similarly, should \{R_5,R_6\} or \{R_6,R_7\} be used to estimate the probability of x_2? Our methodology also shows how to compare alternative sets of rules that could be used to estimate the probability of a consequent.

3. Proposed methodology for recommending an item

In our approach, the recommendation of an item is a three step process. The first step identifies all the eligible rules and from them, the feasible consequents. The second step identifies, for each consequent, the best probability estimate conditioned on the basket (given the relevant rules for that consequent). The third step selects the consequent with the highest probability.

3.1 Identifying eligible rules and consequents

We first find all the eligible rules by ensuring that all items in the antecedent of a selected rule appear in the basket while its consequent does not. The consequents of the eligible rules are added to a consequent list M. In our example, the list M contains two consequents (x_1,x_2).

3.2 Computing the probability of a consequent given the basket

Suppose we have to estimate the probability that a customer with a given basket B will choose item x_i from M. Ideally, we would like to use a rule that has x_i as the consequent and an antecedent identical to B. However, when B includes several items, we are unlikely to find such a rule. On the other hand, we are more likely to find multiple rules with consequent x_i, whose antecedents are subsets of B and that together cover many, if not all, the items in B. These rules can be used to arrive at a better estimate for \( P(x_i|B) \) with appropriate conditional independence assumptions.

Suppose \( r \) eligible rules with disjoint antecedents have the consequent x_i. We denote the antecedent of rule R_i as A_i and use A to denote \( \bigcup_{j=1}^{r} A_i \). We can then approximate \( P(x_i|B) \) by \( P(x_i|A) \) using conditional independence assumptions as

\[
P(x_i|A) = \frac{P(A|x_i)P(x_i)}{P(A)} = \frac{P(A|x_i)P(x_i)}{P(x_i|A) + P(\bar{x}_i|A)} = \frac{(\prod_{j=1}^{r} P(A_j|x_i))P(x_i)}{(\prod_{j=1}^{r} P(A_j|x_i))P(x_i) + (\prod_{j=1}^{r} P(A_j|\bar{x}_i))P(\bar{x}_i)}
\]

Consider x_1 from our earlier example. We can use rules R_1 and R_4 to compute \( P(x_1|B) \). Apart from the confidences of rules R_1 and R_4, equation 1 requires some additional probability parameters - P(x_i) and P(A_j) which are primitives, and P(\bar{x}_i), \forall j and P(A_j|\bar{x}_i) \forall j. Using the primitives and rule confidences, all other parameters required in equation 1 can be calculated. For example, if P(x_1)=0.2, P(A_1)=0.4 and P(A_4)=0.2, we obtain \( P(x_1|B)=0.83 \) from equation 1. This illustrates how the probability of the customer choosing x_1 is quite different when both rules R_1 and R_4 are considered, relative to when either rule is considered by itself.

3.3 Multiple ways of computing the probability of a consequent

The group of rules \{R_2,R_3\} in Table 1 can also be used to estimate the probability that x_1 is chosen given the basket B. To compute \( P(x_1|B) \) using these rules, we need P(x_1|A_2), P(x_1|A_3), which are available in
Table 1, and the additional parameters $P(A_2)$ and $P(A_3)$. Suppose $P(A_2)=0.25$ and $P(A_3)=0.21$. Using equation (i), we find $P(x_1|B)=0.76$. We observe that this estimate for $P(x_1|B)$ is also quite different from when rules $\{R_1,R_4\}$ were used.

Given a set of eligible rules, typically we could combine the information from one or more groups of rules to obtain better probability estimates for a feasible consequent. We call each such group of rules as an admissible rule group (or admissible group for short). Formally an admissible rule group is defined as a set of eligible rules that have the same consequent and disjoint antecedents. An admissible rule group to which no other eligible rule can be added is called a maximal admissible rule group (or maximal admissible group for short). When the union of the antecedents of the rules in the admissible group is equal to the basket $B$, we say that the group covers the basket. The collection of all the eligible rules for a given consequent $x_i$ is called a consequent rule set (or consequent set for short), and is denoted by $\mathcal{C}(x_i)$.

### 3.4 Comparing maximal admissible rule groups

As seen earlier, the confidence of $x_i$ can be computed using multiple admissible groups. We first discuss how to compare maximal admissible rule groups that cover the entire basket; we then extend our findings to maximal admissible rule groups that do not cover the basket. A natural question is, which admissible group should be used to estimate $P(x_i|B)$? Ideally, we would like to use that admissible group which can best approximate the true underlying distribution of $P(B,x_i)$. Therefore, we compare the admissible groups using the likelihood of each admissible group generating the true underlying distribution $P(B,x_i)$. In our case, the likelihoods of interest are those that are associated with the probability models implied by the collection of rules for each admissible group. Hence our problem can be viewed as one of maximizing the likelihood that the observed data is generated from the competing probability models represented by the admissible groups. We derive an important property of our problem that is summarized in Proposition 1.

**Proposition 1:** When admissible groups cover the entire basket, the admissible group that maximizes the likelihood has the highest sum of mutual information (MI) terms associated with the participating rules. (All proofs are suppressed for lack of space)

The MI can be pre-computed for every rule, and kept available for use at run-time. Given a consequent rule set, finding the admissible group that maximizes $\Sigma$MI is a set packing problem, and therefore NP-Hard. In Section 3.5, we propose a heuristic to solve the problem.

So far we have considered maximal admissible groups that cover the basket. However, there may exist maximal admissible groups that cover only a subset of the basket; indeed, it is possible that none of the maximal admissible groups cover the entire basket. In such cases, when considering an admissible group, we assume that the uncovered items and the consequent are independent of each other, and that the corresponding MI terms are zero. While this may not be strictly true, the fact that such rules were not retained after mining suggests that the dependence is weak at best.

### 3.5 Finding a good admissible group

Since the problem of finding the best admissible group is a hard one, we propose a heuristic to solve this problem. The heuristic exploits the following properties of the optimal solution.

**Proposition 2:** The MI corresponding to a rule $(A_j \to x_i)$ is always greater than or equal to the sum of the MI corresponding to rules $(A_{j1} \to x_i), (A_{j2} \to x_i), \ldots, (A_{jn} \to x_i)$ if the antecedents $A_{j1}, A_{j2}, \ldots, A_{jn}$ are mutually disjoint and $\bigcup_{k=1}^{n} A_{jk} = A_j$.

**Corollary 1:** An admissible group $\mathcal{S}_1$ will always dominate another admissible group $\mathcal{S}_2$ if the antecedents of two or more rules in $\mathcal{S}_2$ are subsumed by a rule in $\mathcal{S}_1$.

The algorithm to find an admissible rule group for a consequent $x_i$ is as follows. The rules in the consequent set $\mathcal{C}(x_i)$ are sorted in decreasing order of their mutual information. Going down this list, a rule is added to the admissible group if the rule’s antecedent does not overlap the antecedent of any rule.
that has already been included. The algorithm stops when no other rule can be added to the admissible group.

4. Experiments

We have implemented the algorithm and performed experiments on a real dataset “Retail” available via the FIMI repository (http://fimi.cs.helsinki.fi/data/). From the dataset, 80% of the transactions are chosen randomly for mining the rules (training data), and the remaining 20% are used for testing. Rules are mined using a support threshold of 0.1% and three separate confidence thresholds: 20%, 30% and 50%. We compute the MI and other required probabilities for all the rules generated at the different confidence thresholds from the training data.

For testing purposes, we consider as the relevant basket half the items in each transaction of the test data (transactions with only one item are removed from the test dataset). Our approach observes the basket, identifies eligible rules, determines the consequent set, identifies for each feasible consequent a good admissible group and computes its probability, and recommends the consequent with the highest probability. If there are no eligible rules for a basket, our approach is unable to make any recommendations. When a recommendation is made for a basket, it is considered successful if the recommended item appears among the remaining items in the corresponding transaction. We compare our approach with a benchmark which calculates from the training dataset the conditional probability associated with every item not in the basket given the items in the basket, and picks the item with the highest probability. The benchmark ensures that all items in the basket are explicitly considered when making the recommendations. The benchmark cannot make any recommendations if the training data does not include any transaction that covers the basket. Table 3 summarizes the results for the experiments.

<table>
<thead>
<tr>
<th>Table 3: Summary of results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Confidence thresholds</td>
</tr>
<tr>
<td>Total number of transactions</td>
</tr>
<tr>
<td># (%) of instances heuristic can recommend</td>
</tr>
<tr>
<td># (%) of successes when heuristic recommends</td>
</tr>
<tr>
<td>Aggregate success % for heuristic</td>
</tr>
<tr>
<td># (%) of instances benchmark can recommend</td>
</tr>
<tr>
<td># (%) of successes when benchmark recommends</td>
</tr>
<tr>
<td>Aggregate success % for benchmark</td>
</tr>
<tr>
<td># of instances (RR) when both can recommend</td>
</tr>
<tr>
<td># (%) of successes for heuristic in RR</td>
</tr>
<tr>
<td># (%) of successes for benchmark in RR</td>
</tr>
</tbody>
</table>

The proposed approach vastly outperforms the benchmark both in terms of the ability to make recommendations as well as the accuracy of predictions. For example, when rules are mined using a confidence threshold of 50%, our approach can make recommendations for 85% of the test cases as compared to only 22% for the benchmark. When our approach makes a recommendation it is correct 40% of the time, whereas the benchmark is correct only 16% of the time it can make a recommendation. We

[2] Our approach made correct recommendations 28% of the time when it used only one rule and 46% of the time when it was able to combine two or more rules.
also compared the performance of the two methods for test instances on which both could make recommendations. Once again, our approach outperformed the benchmark 24% to 19%. The change in the performance of our approach when rules are mined at different confidence levels are as expected. When the confidence threshold is lower, the number of rules mined is higher and therefore recommendations are made for more test instances. However, this comes with a reduced accuracy of recommendations.

References