Enhancing Product Recommender Systems on Sparse Binary Data

Ayhan Demiriz
Information Technology, Verizon Inc.,
919 Hidden Ridge, Irving, TX 75038
E-mail:ayhan.demiriz@verizon.com

Abstract

Commercial recommender systems use various data mining techniques to make appropriate recommendations to users during online, real-time sessions. Published algorithms focus more on the discrete user ratings instead of binary results, which hampers their predictive capabilities when usage data is sparse. The system proposed in this paper, e-VZpro, is an association mining-based recommender tool designed to overcome these problems through a two-phase approach. In the first phase, batches of customer historical data are analyzed through association mining in order to determine the association rules for the second phase. During the second phase, a scoring algorithm is used to rank the recommendations online for the customer. The second phase differs from the traditional approach and an empirical comparison between the methods used in e-VZpro and other collaborative filtering methods including dependency networks, item-based, and association mining is provided in this paper. This comparison evaluates the algorithms used in each of the above methods using two internal customer datasets and a benchmark dataset. The results of this comparison clearly show that e-VZpro performs well compared to dependency networks and association mining. In general, item-based algorithms with cosine similarity measures have the best performance.

Keywords: Recommender Systems, Association Mining, Dependency Networks, e-commerce, Collaborative Filtering, Customer Relationship Management.

1 Introduction

Customers on the web are often overwhelmed with options and flooded with promotional messages for products or services they neither need nor want. When users cannot find what they are searching for, the e-commerce site struggles to maintain good customer relations.

Employing a recommender system as part of a site's Customer Relationship Management (CRM) activities can overcome the problems associated with providing users with too little information, or too much of the wrong information. Recommender systems are able to assist customers during catalog browsing and are an effective way to cross-sell and improve customer loyalty.

In this paper, we will compare several recommender systems being used as an essential component of CRM tools under development at Verizon. Our solutions are purposely for the current customers and current products - recommendations for new customers and new products are out of the scope of this paper. Thus every customer has at least some of the products in his/her profile (customer profile).

According to [Breese et al., 1998], there are two main approaches used in recommender systems: memory and model-based systems. Memory-based recommender systems, such as correlation analysis and vector similarity, search the customer database for customer profiles that are similar to the profile of the active customer that the recommendation is made for. In this type of recommender system, it is important that the customer database remain in system memory during the algorithm's runtime. Model-based methods, such as Bayesian networks and Clustering models, approach the problem from a probabilistic perspective to find the best product for a given customer profile [Breese et al., 1998] and need only keep the resulting model in memory while the algorithm runs. The above methods will be described in the following sections.

Because memory-based approaches make predictions based on the local neighborhood of the active user and the model-based approach bases its predictions on the similarities between items (products), recommender systems can be grouped into user-based and item-based systems [Sarwar et al., 2001, Sarwar et al., 2000]. Note that since the summary of the similar customer profiles are stored in the memory rather than the profiles themselves, clustering is considered as a model-based approach. Thus a user-based model is not considered as a model-based one.

User-based systems, as the name suggests, use historical information to identify the neighborhood for the active customer. Products are then recommended according to their similarities to this neighborhood. Because user-based recommender systems use the customer profile data, they can incorporate demographics data along with the historical purchase data. In contrast, item-based method use only historical purchase data to identify similarities between different items. Unfortunately, when the size and the number of dimensions of a customer database are very large, the time required to perform a search utilizing user-based methods may become prohibitive.

Neighborhoods in user-based algorithms are defined according to several measures. Two of these measures are used frequently: Pearson's correlation coefficient and cosine. Pearson's correlation coefficient is computed between the active customer and the rest of the customers in the database. Based on their similarity to the active customer, k customers are selected. Statistics regarding this neighborhood are calculated later in order to accurately recommend products. Pearson's correlation coefficient between customer a and customer b is computed using the following formula:

$$corr_{a,b} = \frac{\sum_i (u_{ai} - \bar{u}_a)(u_{bi} - \bar{u}_b)}{\sqrt{\sum_i (u_{ai} - \bar{u}_a)^2 \sum_i (u_{bi} - \bar{u}_b)^2}} \tag{1}$$

where u_{ai} corresponds to the a^{th} row and i^{th} column of the user-item matrix and u_{ai} is equal to 1 if the customer has the i^{th} product or 0 otherwise. These binary values are equivalent to the user ratings (votes) in the discrete valued data. An average vote for the customer a in this instance is indicated as $\bar{u}_a$. It is also possible to form the cosine between two customers (a and b) in neighborhood k using the following equation:

$$\cos(\vec{a}, \vec{b}) = \frac{\vec{a} \cdot \vec{b}}{||\vec{a}||_2 * ||\vec{b}||_2} \tag{2}$$

The similarity measures defined in Equations 1 and 2 can also be used in item-based recommender systems. In this case, similarities between the items are found first. During the prediction phase, the items that are most similar to the items found in the active user's profile are recommended. Because of the nature of the binary user-item data, these similarity measures may perform poorly for prediction purposes. Several modifications to these measures are proposed in [Breese et al., 1998] to improve the performance of user-based recommender systems. In this paper, both correlation and cosine-based similarity measures are used in the context of item-based recommender systems.

Dependency networks are used to increase the efficiency of collaborative filtering. Each node in a dependency network represents an item (product). In this type of modeling, items missing from the customer profile data are also considered, so that the models found by a dependency network will include some items that a customer does not have. Dependency network modeling is available with Analysis Services of MS SQL

Server 2000 and is also hard-coded into MS Commerce Server 2000 as the standard product recommendation system. The dependency network model is discussed in Section 2.2.

Association mining models have been successfully used in product recommendation systems. In terms of the memory and model-based taxonomy, association mining does not fit easily into either group. Finding association rules based on historical data is similar to the methodology of the model-based approach, but the size and exact search mechanisms between customer profile and association rules resemble item-based approaches. In this manner, it is better to classify association mining as an item-based approach with higher memory requirements. Our proposed method, e-VZpro, does not use the exact match between active user profile and association rules, but instead searches for those rules that are the most similar with the higher confidence levels.

When implementing a recommender system, there are two important issues to consider: the scalability of the filtering algorithm and the quality of the recommendations [Sarwar et al., 2001]. Successful e-commerce sites may have several million customers and product catalogs that include thousands of items. When each customer is interested in and purchases only a small number of items, the user-item matrix used by collaborative filtering algorithms may be very sparse - which, in turn, produces poor recommendations for customers. To increase the quality of recommendations, a user-based collaborative filtering algorithm may be introduced, which compromises scalability. It can be seen, then, that the quality of recommendations and the need for scalability in a collaborative filtering system are often in conflict with each other.

This paper seeks to make four major contributions. We use a similarity measure for finding the best rule instead of an exact match. We show that recommender systems based on association mining have stronger predictive abilities when compared to dependency network systems. We also demonstrate how clustering data improves the predictive qualities of association rules. Finally, we conduct an empirical study to compare several algorithms on very sparse and binary user-item data.

The paper is organized as follows: Section 2 contains a brief discussion of some related work and other recommender systems. The methodology of our approach is discussed in Section 3, along with a toy example. There are two internal Verizon datasets and the MSWeb benchmark dataset (provided in [Hettich and Bay, 1999, Breese et al., 1998, Heckerman et al., 2000]) for empirical comparisons of various recommender systems. The summary of the study's findings is reported in Section 4, along with details of our implementation. Section 5 contains our conclusions and information on future work directions.

2 Related Work

The introduction of customer-centric management techniques and the emergence of a web-based economy have forced the creation of a new generation of companies and the evolution of existing, traditional companies. The deregulation of previously protected industries has also led to increased competition for many companies, who now seek to improve their service quality and customer loyalty. Companies in every industry are finding that their customers increasingly demand products, services, and solutions that are tailored to their specific needs.

Rapid changes in internet technologies, especially the increasing availability of high-speed broadband connections, allow customers to go online and easily research the products and services they desire. Customers are able to compare competing items through corporate web sites, so much so that e-commerce sites now play a role as important as the traditional customer touch-points such as call centers.

In every contact with a customer, companies need to offer their best products and services to gain customer confidence and retain consumer loyalty. The automated nature and frequent use of e-commerce sites highlights the need for recommender systems that can assist the customer. But recommender systems are also useful in call centers, where the staff is not always as experienced or trained in sales as would be ideal and the system could assist the customer service and sales representatives. In these cases, a recommender system should be designed in accordance with the company's overall marketing strategy and a successful system will also benefit employees by helping them attain larger performance bonuses.

2.1 Common Recommender Systems

In the past decade, recommender systems have been put to successful use in the publishing, news, film, music, and video industries. Although "collaborative filtering" was first coined in [Goldberg et al., 1992], communications of the ACM featured several pioneering works in the third issue of Volume 40 [Resnick and Varian, 1997] in 1997, including the GroupLens Project [Konstan et al., 1997], and Siteseer [Rucker and Polanco, 1997].

GroupLens Project is a recommender system based on the Usenet news groups wherein recommendations are made according to the correlations among the news ratings provided by the users. Depending on the user's ratings, he/she gets recommendations of the unread messages and these unread messages can then be sorted according to their predicted rankings. Although news groups are clustered well in terms of the content and the user preferences, each user can only read and evaluate (rate) very few messages, which causes an obvious sparsity problem for predictions.

Siteseer, on the other hand, recommends web pages based on bookmark similarities, eliminating the need for user ratings. Early recommender systems had the advantage of working with communities wherein the users are very similar to one another, as is the case with newsgroups and other internet-based communities.

A recent work by Shafer, Konstan, and Riedl [Schafer et al., 2001] summarizes the taxonomy of recommender systems and their use in e-commerce applications. Recommender systems have the potential for greater predictive ability than database marketing techniques, since recommendations - depending on the method used - are made in highly similar, homogenous, and small neighborhoods rather than in segments consisting of much larger populations. Although one can come up with an extremely successful segmentation scheme to have very homogenous segments, it still lacks the finer granularity of personalized recommendations. Moreover, in practice segments correspond to very high level summary of customer base.

The recommender systems from six e-commerce sites, including Amazon.com, CDNOW, Drugstore.com, eBay, MovieFinders.com, and Reel.com are given in [Schafer et al., 2001] as examples of the currently popular recommender systems. These e-commerce sites use a number of recommendations, such as Customers Who Bought, Customers' Ratings, Editors' Picks, Top 100, and Customer Preferences. An important consideration when deploying a recommender system is the method of recommendation delivery. An online, real-time solution requires an extremely fast response from the system, but recommendations delivered via e-mail do not require great speed, but must insure good quality.

The "keyword search" interfaces commonly used on e-commerce sites can be considered one of the simplest types of recommender systems. Virtually all e-commerce sites have this feature by default, allowing any customer to search for information related to specific products or product categories. High volume, standardized products or highly recognized brands are the most likely to see significant benefits from using a simple search utility as a recommender system.

Recommender systems based around "editors' picks" are also very common. These systems require "manual rules" for their product recommendations and the expert opinions are very useful in extremely connected communities with very similar member preferences.

A recommender system based on general public opinion is also useful in many domains. Basic statistical summaries such as the most popular products or Top 100 music charts will be sufficient for this purpose. Statistical summaries are best used for non-personalized recommendations [Schafer et al., 2001].

As we briefly mentioned in our discussion of user-based recommender systems in Section 1, the goal of these recommender systems is to predict the utility of the product offerings to the active customer based on the preferences of either a select group of customers or all customers [Breese et al., 1998]. As given in Equations 1 and 2, the most popular methods for computing similarity are correlation coefficient and cosine. Of course, the vector similarity is a naïve way of finding customers similar to the active customer; inverse user frequency and case amplification are modifications to similarity methods that can be used to reduce the weights of highly used (preferred) products. In cases where there are only a few votes or purchased items common among users, similarity measure based on correlation may be problematic. Default voting, to assign minimal weights to some items, was proposed in [Breese et al., 1998] as one method to prevent poor performance in the course of intersecting common votes (purchased products) and, instead of the intersection, the union of voted items could be used [Breese et al., 1998].

A number of successful e-commerce sites have employed user-based methods with the assumption that each customer or individual belongs to a larger group of customers [Karypis, 2001]. The scalability and online performance of these algorithms are considered as the main limitations and the sparsity of the user-item matrix may require the use of some of the remedies mentioned above, such as inverse user frequency. On the other hand, the sparseness may result in shortened recommendation times with the implementation of the sparse matrix operations for calculating similarities.

Item-based recommender systems are very effect when a large customer database is available [Sarwar et al., 2001, Karypis, 2001]. Like user-based recommender systems, cosine and correlation based similarities are used to find the item similarities. Conditional probability [Karypis, 2001] is another way of computing item similarities and modified conditional probabilities and adjusted cosine similarities [Sarwar et al., 2001] are often used to minimize the effect of high-frequency items. Frequency scaling is a common method used in information retrieval systems as well, and remarkably short recommendation times have been reported [Karypis, 2001] when using this methodology in item-based recommender systems. In [Karypis, 2001] a regression model was used to predict the recommendations after finding items similar to the active customer profile, showing that item-based recommender systems compromise very little on recommendation quality while providing substantial gains in computational efficiency [Sarwar et al., 2001].

Statistical and Machine Learning techniques have been successfully used for the model-based recommender systems. Indeed, collaborative filtering is a major research topic in statistical and machine-learning communities. The probabilistic nature of the problem requires robust and efficient techniques to recommend products accurately. In the following subsection, we will discuss a well-known collaborative filtering technique based on the dependency networks introduced in [Breese et al., 1998, Heckerman et al., 2000].

2.2 Dependency Networks

Bayesian networks have been successfully used in the visualization of the predictive relationships between the variables of a particular dataset. There is a problem, however, in that Bayesian networks can be very confusing for an untrained person to interpret [Heckerman et al., 2000]. Dependency networks are proposed in [Heckerman et al., 2000, Breese et al., 1998] to represent predictive or correlational relationships that exist in data in a more easily comprehensible manner. On the other hand, it is computationally very easy to construct a dependency network compared to the effort required to create similar Bayesian networks. In addition, it is a simple task for any probabilistic classification or regression model [Heckerman et al., 2000] to estimate the conditional distribution of each variable in relation to the rest of the variables. Because efficient probabilistic classification and regression models exist, it takes less time to construct a dependency network than to construct a comparable Bayesian network.

To illustrate the computational efficiency of the dependency network, we have adopted the following simple example from [Heckerman et al., 2000]. Assume the problem is to find the dependency networks from the domain $\mathbf{X} = (X_1, X_2, X_3)$. In this case, all that is required is to estimate three conditional probability distributions $p(x_1|x_2, x_3)$, $p(x_2|x_1, x_3)$, and $p(x_3|x_1, x_2)$. These can be estimated using any classification method, such as logistic regression, neural networks, and probabilistic decision trees. For illustration purposes, assume these methods are capable of variable selection and that one of these methods, let's say the probabilistic decision trees, found that X_1 is not a significant predictor of X_3 and X_3 is not a significant predictor of X_1. In this case, we can easily construct the following dependency network: $X_1 \leftrightarrow X_2 \leftrightarrow X_3$. This example shows how dependency networks are constructed in a very precise and simple manner. Note that the variable selection determines the structure of the dependency network [Heckerman et al., 2000].

Though constructing the dependency network is computationally very reasonable in general, inconsistent conditional probability distributions - due to the heuristic search and finite-data effects - are potentially major drawbacks to this method [Heckerman et al., 2000]. Nevertheless, having large datasets in this particular application gives us the confidence to apply a collaborative filtering based on the dependency networks, since strong inconsistencies will be rare when dealing with large datasets. Thus, we expect that resulting dependency networks will be "almost" consistent according to the discussion in [Heckerman et al., 2000]. Another possible issue is that, in some cases, it is prohibitive to construct dependency networks like other model-based methods. Moreover, each additional data requires full training as in the case of many model-based methods. In the following subsection, we will briefly discuss some of the other model-based approaches.

2.3 Other Model-Based Recommender Systems

A very early work on recommender systems based on machine learning algorithms is introduced in [Billsus and Pazzani, 199
The concept is to first manipulate the data by transforming the user-item matrix into Boolean feature vectors in order to use Singular Value Decomposition to find important features. Finally, a neural network is trained based on these important features - unfortunately, this method lacks the scalability offered by other methodologies.

Clustering is successfully used to summarize and analyze data in many different domains. As a natural extension, it has been used in various recommender systems [Ungar and Foster, 1998, Breese et al., 1998] by grouping customers with similar profiles into the same cluster and then recommending products based on the popularity of each product within that cluster. Expectation-Maximization (EM) based clustering algorithms are used in both [Ungar and Foster, 1998, Breese et al., 1998] of the above methods. The Bayesian Clustering method performed very well in terms of accuracy in [Breese et al., 1998] as an alternative to the dependency networks and user-based methods.

A hybrid method based on Personality Diagnosis (PD) is proposed in [Pennock et al., 2000]. In this method, the personality type of the active user is determined and used to compute the probability of the active user having new items. PD-based collaborative filtering requires using all of the available data throughout the process, though new data can be added incrementally [Pennock et al., 2000] to adjust model parameters.

Association mining is best used to perform traditional market basket analysis where the rule discovery is conducted to uncover the associations between various products. Recommender systems based on association mining are very common in e-commerce applications [Sarwar et al., 2000]. The algorithm Apriori and its derivatives are commonly used in association mining. The implementation in this paper, however, differs from previous approaches in an important point - rather than finding exact matches between the active user and the rule base, e-VZpro finds the most similar rule with the highest confidence rate by scoring each rule in terms of both its similarity to the active user and its confidence rate. Association mining can also be used to predict certain phenomenon (classification problem); a case study in the telecommunication industry [Ali et al., 1997] uses association mining to predict whether manual assistance will be required for a specific order. Based on the Universal Service Order Codes (USOCs) attached to an order, a prediction model determines the probability of the need for manual assistance with that order. We postpone further explanation of association mining until toy example in the next section. In the following section, we will discuss the details of methodology used in proposed algorithm.

3 Methodology Used in e-VZpro

The main purposes of this section are to introduce the methodology used in e-VZpro and to give a simple example. There are two phases to our methodology: Discovery and Scoring. In the first phase, association mining is used to discover the rules that exist in historical customer data in a manner similar to that used in item-based recommender systems. During the second phase, a scoring algorithm searches the rule base according to the active customer profile to find products that can be offered to that particular customer. Since all rules are kept in memory and a subset of the rules are scored sequentially, this phase is most similar to the user-based recommender systems discussed earlier.

In the second phase, each product that the customer does not have is scored by finding corresponding rules for the active product. Scores for the active product are then computed by multiplying the similarity measure between the rules and the active customer profile by the confidence rate of the rules. More formally scores are computed using the following equation:

$$score(u, r) = similarity(u, r) \times confidence(r) \tag{3}$$

In Equation 3, u and r represent the active user and the rule, respectively. The similarity measure is computed based on a normalized Euclidian distance measure given in following equation.

$$similarity(u, r) = 1 - \sqrt{\frac{\sum_{i:r_i>0}(u_i - r_i)^2}{4 * \sum_{i:r_i>0} r_i}} \tag{4}$$

The rationale behind this formulation is as follows: Euclidian distance is a measure of dissimilarity and, in order to have similarity measures between 0 and 1, it is necessary to normalize Euclidian distance by dividing it by the maximum discrepancy and then subtracting this normalized distance from 1. Because some products may not be available in certain areas, e-VZpro was designed to handle these cases by including an option in the customer profile for “-1” to represent an unavailable product in addition to the standard entries of “0” and “1”. Thus, the maximum discrepancy might be equal to $4 * \sum_{i:r_i>0} r_i$. Where a perfect match between customer profile and rule are found, the similarity is equal to 1. Again, u and r represent the active user and the rule respectively in Equation 4.

Because the rules and customer profiles are composed of 0's and 1's it is problematic to use correlation coefficient and cosine directly. Since the rules are sparser than customer profiles, using correlation coefficient or cosine as a similarity measure is misleading. As the algorithm tries to find rules that are similar to the active user profile, the similarity measure between a rule and the active customer profile is dependent on the magnitude of the left-hand side of the rule. Association rules might have multiple items on the right hand side of the rules but, due to the nature of the prediction problem in this paper (recommendations are independent of one another and customers will select only one of several recommendations) we only use rules that have singleton right-hand sides.

In practice, there will be several rules that their right hand sides point to the same product. But these rules might certainly have different confidence rates. Thus their similarities and confidence rates will be different i.e. scores from these rules will vary. The highest score for offered products determines the ranking of each product. In addition, revenue information (recurring charges) can be used to weight the scores to find the final ranking of each product - this method is not explored in this paper. Finally, the Top-N products as determined above are offered to the active user. While scoring is done on the fly, association mining is a batch process and, thus, e-VZpro clearly resembles both the memory-based and user-based methods. In the rest of this section, we will explain our methodology with a toy example.

Given the customer purchase data (such as in Table 1), association mining, also known as market basket analysis, discovers the rules (relationships) that exist within the historical customer purchase (service order) data. Each of the rules include "if" clauses by default and the structure of the rules is as follows: If the customer purchases Product A, then with $C\%$ probability he/she will buy Product B. This probability is called the confidence rate. More formally, the confidence rate can be computed as:

$$C = \frac{frequency(A \cup B)}{frequency(A)}$$

where $A \cup B$ indicates transactions that include both Product A and Product B. Confidence rate is also equivalent to the conditional probability of a customer having Product B if they already have Product A. Another important statistic in association mining is the support level, which is essentially equal to the fraction of transactions that have both Product A and Product B. Thus, the support level S can be computed as:

$$S = \frac{frequency(A \cup B)}{T}$$

where T is equal to the total number of transactions. The support level is the most important factor when pruning the association rule base.

To depict our analysis with a simple example, a toy problem is provided in Table 1. USOCs given in Table 1 are actual values extracted from our databases. The descriptions of the USOCs are found in Table 2.

After the analysis with 100% confidence rate, we found the following rules using association mining:

Table 1: Sample Customer Data

Cust-id	USOC
1	47036
1	41368
1	55822
2	42267
2	41368
3	47036
3	55822
3	42267
4	42267
4	55822
4	47036
4	41368

Table 2: Descriptions of USOCs used in the Sample Data

USOC	Description
41368	Personal 800 Number
42267	Basic Voice Mail Service
47036	Caller ID -Name and Number
55822	Non-published Listing

$47036 \rightarrow 55822$

$55822 \rightarrow 47036$

$47036\ 41368 \rightarrow 55822$

$41368\ 55822 \rightarrow 47036$

$47036\ 42267 \rightarrow 55822$

$55822\ 42267 \rightarrow 47036$

For example, the first rule implies that if the customer has Caller ID - Name and Number, then it is 100% likely that he/she will have or by Non-Published Listing. On the other hand, the last rule implies that if the customer has both Non-published Listing and Basic Voice Mail Service then there is a 100% probability that he/she will have/buy Caller ID - Name and Number. Assuming that we want to offer a product to a customer with only the Basic Voice Mail Service in his/her profile, we can offer the customer any or all of the other three products. Since Basic Voice Mail Service is not found on the left side of any of the first 4 rules, they receive the lowest scores. The first two rules have a similarity measure of $1 - \sqrt{\frac{1}{4*1}} = 0.5$. The third and fourth rules have a similarity measure of $1 - \sqrt{\frac{1+1}{4*2}} = 0.5$. But the last two rules are some how similar to the customer profile. Their similarity measures are $1 - \sqrt{\frac{1+0}{4*2}} = 0.6465$. The final scores are also equal to 0.6465 because the confidence rates are equal to 1 for both rules. Finally, we are able to offer the Caller ID - Name and Number as well as the Non-Published Listing.

We used 100% confidence rating to illustrate the algorithm; while it would have been possible to use a lower confidence rate, it would have required a prohibitive number of rules to illustrate in this toy example. An empirical study is provided in the next section to compare e-VZpro and dependency networks.

4 Experimental Results

As was mentioned in the previous sections, there are many factors that affect the performance of a given recommender system. Previous empirical studies [Breese et al., 1998, Karypis, 2001, Sarwar et al., 2001] indicate that item-based approaches are very scalable and result in very negligible losses in recommendation quality when compared to user-based approaches. Our goal in this section is to demonstrate that association mining-based recommender systems are comparable with the well-known recommender system introduced in [Breese et al., 1998, Heckerman et al., 2000] and other item-based recommender systems.

Problems studied in this paper do not include user ratings, as it is simple to determine whether or not users have a specific product. Thus, the user-item matrix is designed to include only 0-1 entries. It is assumed that all products are available to all customers in this study and that users cannot have multiple instances of the same product.

Three different datasets were used in this study - two of these datasets contain internal Verizon customer information and the third is the MSWeb dataset from the UCI KDD archive [Hettich and Bay, 1999]. One of the Verizon customer datasets was sampled from a pool of residential customers located in the western states (WESTDB2) and the second dataset was sampled from a pool of general business customers located in the state of Texas (GENBUS). The samples were taken from the working telephone numbers (WTN) from the two different data sources. In reality, customers can have multiple telephone lines, but the current study treats each line as a unique customer. A transaction (customer profile) is defined as a set of products (services) that belongs to an associated customer (WTN). The MSWeb dataset was used in [Breese et al., 1998, Heckerman et al., 2000] and details about this dataset can be found there. We use MSWeb as a benchmark dataset that consists of user activities on certain Microsoft pages.

As we mentioned in the previous section, the orders in the telecommunication industry are processed at the USOC level. Many orders contain multiple USOCs and, depending on the geographical location, price, discount, and regulations, the same product may have different USOCs. Some USOCs are associated with certain products - such as Caller ID and Call Waiting - but may have no additional charges at all. On the other hand, certain products may have different features and different pricing. Before performing any type of analysis for this study, the data was prepared by grouping the USOCs into 70 meta-product groups and eliminating the most frequent products such as residential and business lines. Some products did not have a high enough frequency to be included in the analysis, so the WESTDB2 and GENBUS datasets consist of 51 and 61 product groups respectively. The characteristics of the datasets are summarized in Table 3. MSWeb originally had 5,000 test instances, but we omitted the test cases that had only one page in the user profile. The sparsity of each dataset is defined as $1 - \frac{\text{nonzero entries}}{\text{total entries}}$ in [Sarwar et al., 2001].

The original WESTDB2 and GENBUS datasets have 174,933 and 205,882 customers respectively. Since we do not have repeating customers in these datasets and customer profiles are constant at the time of association mining, a customer profile can be considered as a transaction. As with the evaluation of any other machine learning method, we divided our datasets into two subsets: training and test data. Approximately 85% of the original data was reserved as the training data; those transactions with at least 2 products in the customer profiles were used as the training data from the reserved data. The training data was used to construct the dependency networks or to discover the association rules used to create the rule base for e-VZpro. The remaining data was used as test data.

Because our focus in this paper is to compare the predictive abilities of two different recommender systems, we adopted the *all-but-1* protocol used in [Breese et al., 1998]. As a significant amount of the test

Table 3: Summary of the Datasets

	Dataset		
Statistics	WESTDB2	GENBUS	MSWeb
Number of Distinct USOCs	1789	1602	NA
Number of Transactions (WTNs)	174933	205882	37711
Number of Transactions in Training Data	130379	151921	32711
Number of Transactions in Test Data ($\|U\| > 1$)[1]	22582	18913	3453
Ave. Number of Items across All Data[2]	3.82 ± 2.2	2.85 ± 1.6	3.02 ± 2.5
Ave. Number of Items across Training Data	4.21 ± 2.1	3.12 ± 1.5	3.02 ± 2.5
Ave. Number of Items across Test Data	4.21 ± 2.1	3.13 ± 2.1	3.95 ± 2.5
Number of Items in Data	51	61	294
Number of Nonzero Entries in Training Data	548944	473887	98654
Sparsity of Training Data	0.9174	0.9489	0.9897
Number of Nonzero Entries in Test Data	98446	84887	13644
Sparsity of Test Data	0.9145	0.9264	0.9866

Notes:
[1] $|U|$ is equal to the number of products in a transaction (customer profile) .
[2] Standard Deviations are reported after ± symbols.

data is only suitable for the *all-but-1* protocol, other test protocols were not used. Some of the products might be used in the models to predict for other products, but we chose to not offer such products to the customer through the recommender system in this study. For example, Product 26 in our dataset is not offered by the recommender systems in this study, so the test data contains only those transactions that have at least 2 products other than Product 26 before one of the products was removed according to the *all-but-1* protocol.

The accuracy rate of the systems was defined as $\frac{\text{number of successes}}{\text{number of transactions}}$ where predicting the removed product from the test data within the Top-5 offers is defined as a success. Evaluating less than 5 predictions will result evidently lower accuracies. Choosing Top-5 recommendations is operationally motivated not arbitraryly. We are required to show as many recommendations as possible to the clients by utilizing the very limited area in the user interface. The predictive abilities of the two different methods are evaluated by their accuracy rates on the test datasets. The specific details of the dependency networks implementation are given in the following subsection.

4.1 Implementing The Dependency Networks

Constructing the dependency networks is an important feature of MS SQL Server 2000. Unlike other modeling software, SQL Server allows users to construct multiple prediction models in a single step using the same dataset. Collaborative filtering is a special implementation of this type of modeling that works very well and is easy to construct. MS Commerce Server 2000 also comes with a recommender system that uses dependency networks, but is limited by its ability to use only 20,000 cases. It was decided that MS Commerce Server 2000 would not be used in this study due to this limitation.

Training the dependency-networks-based recommender systems took around 48, 74 and 194 minutes of computation for the WESTDB2, GENBUS, and MSWeb datasets respectively. All computations were performed on a dedicated SQL Server 2000 machine with two Pentium III 1 GHz CPUs and 4 GB of physical memory. Note that the current data mining models in SQL Server 2000 are not parallel algorithms. The resulting dependency networks are depicted in Figure 1 and Figure 2.

An important visualization feature of dependency networks is that they can be filtered to remove the weaker links [Heckerman et al., 2000] and related products are shown closer to one another. Due to the inadequate frequencies of certain products, dependency networks have no links for some products as seen in Figure 1. Dependency networks resulted in 41.93%, 47.59%, and 31.07% accuracy rates for the WESTDB2, GENBUS, and MSWEB test datasets respectively.

It should be noted this paper uses collaborative filtering based on dependency networks as a benchmark method. The superiority of this method is discussed in [Breese et al., 1998, Heckerman et al., 2000]. In the MS SQL Server 2000 data mining models, prediction is defined as a join operation [Microsoft, 2000] and is thus optimized very well, taking less than 5 minutes to find the recommendations for the test datasets. Of course, there is a need for post-processing to find the Top-5 offers. The specification for the `Prediction Join` can be found in [Microsoft, 2000].

For many cases from the test data, we encountered that dependency network models returned NULL predictions, meaning the recommender systems did not find any product to offer some customers. The fractions of the NULL predictions are 26.13%, 36.88%, and 41.47% for the WESTDB2, GENBUS, and MSWeb datasets respectively - these null predictions were not considered as successes. The accuracy rates reported above reflect this reality and a similar evaluation approach was used for e-VZpro.

The results from our experiments with e-VZpro and association mining are reported in the following section. Several experiments were conducted by changing the support level (S) and confidence rating (C). The results indicated that e-VZpro is very competitive with the collaborative filtering system that uses the dependency networks approach. An extension of e-VZpro and association mining that clustered the data prior to the analysis also improved the results significantly.

4.2 Implementing e-VZpro and Association Mining

The current implementation of e-VZpro was designed to investigate the predictive ability of the method. As reported in [Karypis, 2001] we implemented e-VZpro and association mining in sparse matrix format, keeping the rules and customer profiles in sparse matrix format in memory. Customer profiles from the test datasets were read one at a time, which reduced the memory requirements for the experimental software, but also had some negative impact on the I/O times. As indicated in previous sections, association mining has been used successfully as a recommender system and experimental results for the traditional association mining recommender system are reported.

The results from the experiments with e-VZpro and association mining on the WESTDB2 and GEBUS datasets are show in Table 4 and Table 5, following. As previously mentioned, support level (S) and confidence rate (C) are important parameters in association mining. To investigate the effects of changing these factors, the association rules were generated according to 12 different parameter sets. We used 3 different support levels (0.01%, 0.05%, and 0.1%) and 4 different confidence rates (55%, 60%, 65%, and 75%). Corresponding number of rules are also reported in Tables 4 and 5 for the WESTDB2 and GENBUS datasets respectively. Decreasing the support level increases the number of rules and increasing the confidence rate decreases the number of rules for any given support level. In a sense, support level and confidence rate are used as two important knobs to adjust the overfitting or underfitting of e-VZpro. Certainly decreasing both support level and confidence rates will generate more rules. The more rules we have, the more precise rec-

Table 4: Accuracy Results from e-VZpro and association mining on WESTDB2 dataset

			e-VZpro		Assoc. Mining	
Support Level (%)	Confidence Rate (%)	Number of Rules	Time (sec.)	Accuracy (%)	Time (sec.)	Accuracy (%)
0.01	55	394203	11915	52.99	**8442**	**45.03**
0.01	60	345173	10077	52.28	7221	36.50
0.01	65	294471	8498	51.92	6044	26.62
0.01	75	209445	6045	49.88	4182	14.08
0.05	55	63611	1848	55.61	1390	44.54
0.05	60	53396	1533	53.13	1152	36.00
0.05	65	43786	1253	50.24	945	26.04
0.05	75	27049	796	47.64	602	13.39
0.1	**55**	**26850**	**741**	**58.87**	563	44.19
0.1	60	22040	599	55.66	455	35.68
0.1	65	17758	477	53.06	364	25.77
0.1	75	10366	292	51.11	223	13.04

ommender system we can build. But excessive number of rules will reduce both computational performance and the accuracy due to the overfitting. Finding right combinations of support level and confidence rates is certainly a tradeoff decision.

The running time of e-VZpro and association mining (in seconds) is reported in Table 4 and Table 5. The experiments were conducted on the same machine as dependency networks based recommender systems. Because computational times are dependent on the number of rules, the advantage of e-VZpro should be apparent with a smaller number of association rules. Note that it took only a few minutes to discover the association rules for any experiment in this study and the reported times do not include the discovery time. When the number of rules is less than the number of customers in the training data, e-VZpro is computationally better than memory-based methods. For the best results, e-VZpro ended up using 26,850 and 92,389 rules on the WESTDB2 and GENBUS datasets, respectively; association mining takes less time than e-VZpro. The accuracy results are shown in Tables 4 and 5 and best performances are shown in **bold**.

e-VZpro and association mining were run on the MSweb dataset using support level and confidence rate. We arbitrarily chose 0.05% and 55% as support level and confidence rate. e-VZpro and association mining used 30,181 rules to recommend pages for the MSWeb dataset. e-VZpro took 149 seconds to run and resulted in 50.51% accuracy while association mining took 115 seconds to and achieved 31.94% accuracy.

These results clearly indicate that e-VZpro made consistently better predictions when compared to association mining. For the WESTDB2 dataset, e-VZpro performed remarkably well (58.7% compared to 41.93%) and its best result, (46.20% compared to 47.59%) for the GENBUS dataset was very close to the result of the dependency networks based recommender system discussed earlier. Note that e-VZpro performed very well for the less sparse dataset. At a fixed support level, increasing the confidence rate diminishes the accuracy rates of e-VZpro in general; it can be concluded that e-VZpro can make better predictions when its association rules are discovered at lower confidence rates. Changing confidence has a greater effect on performance of association mining than changing the support level and rules found at lower confidence levels have better predictive abilities.

Since similarity is used in e-VZpro rather than exact match, the rules with at least one item (product) matching in left-hand side will be considered. There will be at least one rule that will always satisfy

Table 5: Accuracy Results from e-VZpro and association mining on GENBUS dataset

			e-VZpro		Assoc. Mining	
Support Level (%)	Confidence Rate (%)	Number of Rules	Time (sec.)	Accuracy (%)	Time (sec.)	Accuracy (%)
0.01	55	103392	2856	46.11	**2165**	**35.66**
0.01	**60**	**92389**	**2561**	**46.20**	1927	34.58
0.01	65	82476	2269	46.18	1714	33.16
0.01	75	65491	1807	43.97	1371	28.51
0.05	55	13127	333	44.58	253	35.33
0.05	60	11642	290	44.39	221	34.15
0.05	65	10335	255	44.35	194	32.77
0.05	75	7491	183	41.96	139	28.18
0.1	55	5727	133	44.32	98	35.05
0.1	60	5101	115	44.08	83	33.88
0.1	65	4483	97	43.85	71	32.59
0.1	75	3196	67	40.96	50	28.10

this condition. Thus e-VZpro does not return NULL predictions. Association mining, however, resulted in 12.86%, 38.98% and 41.01% NULL predictions for the WESTDB2, GENBUS, and MSWeb database respectively, pointing out a significant difference between e-VZpro and association mining.

We can also easily determine that the support level used for the GENBUS dataset is not low enough. In other words, there are not sufficient association rules to represent the underlying relationships that exist within the data. There are two approaches to handling this problem. We can either lower the support level to generate a larger set of association rules, or we can cluster the training data and find the association rules for each of the clusters separately. This situation will definitely reduce the computational performance of e-VZpro and the following subsection will investigate the effect of clustering on the predictive abilities of e-VZpro and association mining.

4.3 Using Clustering in Association Mining

As noted above, to prevent the possibility of poor performance of e-VZpro and association mining in terms of accuracy at low support levels and to investigate the clustering approach, we grouped the user-item matrix of the datasets into 5 clusters. This was accomplished using the clustering algorithm provided by MS SQL Server 2000 and we intentionally chose a small number of clusters to keep the experiment within manageable levels. For the sake of simplicity, we set the confidence rate to 55% for all datasets.

As shown in Tables 4 and 5, e-VZpro and association mining take longer to run when there are too many rules. In contrast, these methods perform poorly with very few rules but take a very short time to run. Because of the behavior of the clustering algorithm in MS SQL Server 2000, the first cluster always has the highest number of cases with later clusters containing less data. The last cluster, Cluster 5 in our case, has the lowest number of cases, but very dense data. In other words, the first cluster has the greatest number of cases, but the fewest number of products included in customer profiles. Cluster 5, on the other hand, has a very small number of customers with highly dense profiles containing numerous products.

Because of these discrepancies, it was necessary to set the support levels of the clustered data differently. For the WESTDB2 dataset, support levels were set to 0. 01%, 0.1%, 0.1%, 0.1%, and 0.1% respectively, for

Table 6: Results on Clustered Data

	WESTDB2			GENBUS			MSWeb		
	Accuracy %		No of	Accuracy %		No of	Accuracy %		No of
	e-VZpro	Assoc.	Cases	e-VZpro	Assoc.	Cases	e-VZpro	Assoc.	Cases
Cluster 1	67.89	36.23	14529	65.43	33.01	8930	51.81	22.56	1241
Cluster 2	72.27	50.30	3549	68.35	34.81	6078	51.64	32.48	856
Cluster 3	69.59	67.69	1825	60.34	54.00	1702	56.67	39.82	540
Cluster 4	66.89	61.75	1676	71.45	47.56	1373	47.39	39.78	460
Cluster 5	61.12	60.92	1003	63.98	59.88	830	47.47	46.63	356
Overall	68.34	43.97	22582	66.28	37.71	18913	51.49	32.49	3453

Table 7: Top-5 Recommendations for the Test Case Number 3330

Method	Recom. 1	Recom. 2	Recom. 3	Recom. 4	Recom. 5
e-VZpro	15 - 1	8 - 0.6723	9 - 0.6723	46 - 0.6655	29 - 0.6631
Dependency Networks	29 - 0.7737	40 - 0.5048	NULL	NULL	NULL

each of the clusters from 1 to 5. These values were set to 0.001%, 0.01%, 0.01%, 0.01%, and 0.1% for each of the clusters in the GENBUS dataset and 0.05%, 0.05%, 0.05%, 0.05%, and 0.1% for the MSWeb dataset. As seen from these support levels, higher levels were set for Cluster 5 in each dataset to prevent an excessive number of association rules.

Results after running e-VZpro and association mining on clustered data are reported in Table 6. The accuracy of e-VZpro and association mining with the number of test cases in each of the clusters is reported in Table 6 on WESTDB2, GENBUS, and MSWeb datasets. Overall accuracy of the e-VZpro has jumped significantly on WESTDB2 (From 58.87% to 68.34%) and GENBUS (from 46.20% to 66.23%) after clustering the data.

The results of association mining either improve or degrade a bit after clustering. Both e-VZpro and association mining improve somewhat after clustering on the MSWeb dataset. We conclude that clustering data and then finding association rules for each cluster helps to improve e-VZpro significantly on the less sparse data but has no effect on very sparse data. This point deserves greater investigation to determine if clustering algorithms perform less effectively on data with a large number of dimensions. In short, e-VZpro resulted in significantly better predictions than the dependency networks based recommender systems. In the following subsection we discuss the differences between the two methods and their effects on the corresponding performance of these two methods.

4.4 Further Discussions on e-VZpro and Dependency Networks

To better depict the underlying methods, this section contains a test case from the WESTDB2 dataset. The test case number 3330 originally had Product 1, Product 26, Product 29, and Product 51 in the profile. Product 29 was randomly removed from the customer profile according to the *all-but-1* protocol. Thus the remaining products were used to find the Top-5 recommendations given in Table 7 for this customer profile.

In Table 7, the numbers on the left side and the right side of the hyphen correspond to the recommended product and the probability (Score) of the prediction. Because both methods predicted Product 29 within the Top-5 recommendations, they are both successful for this test case. e-VZpro used the association rule Product 1, Product 26, Product 51 $\rightarrow$ Product 29 with a confidence rate of 0.6631. Since the rule and the customer profile match exactly, the similarity between this particular profile and the rule is equal to 1. The score for this case then becomes 0.6631 for Product 29.

Based on the decision tree depicted in Figure 3, dependency networks predicted Product 29 as the top recommendation. From the Node Path window, we can see that the corresponding rule for this particular case is "if Product 1 > 0.334 and Product 26 > 0.221 and Product 33 ≤ 0.159 and Product 51 > 0.147 then with 0.7736 probability the case 0.606 (means 1) and 0.2248 probability the case is 0.106 (means 0)". Since 0.7736 is greater than 0.5, the prediction is 1 for this particular case.

Many data mining and machine learning models might fail to perform well in certain cases due to the overfitting or the underfitting problems. Recommender systems are no exception and changing the support level and confidence rate in our approach controls the degree of the training process. Pruning is an important step in constructing decision trees to prevent overfitting. The decision tree model used in MS SQL Server 2000 does not allow the users to prune the trees, as the model is not flexible enough to allow the use of any pruning method. As it is seen from the unbalanced and very deep decision tree in Figure 3, the decision trees found in MS SQL Server 200 are highly overfitted. The only option we have in this case is to limit the sample size, which explains why MS Commerce Server 2000 has an upper limit on the number of cases used in training the recommender system. Overfitting could also be one of the reasons that recommender systems based on the dependency networks had poor results.

Overfitting in the dependency networks also caused the NULL prediction problem as was indicated in Section 4.1. This resulted in a poor performance compared to the e-VZpro, which is designed not to predict NULLs. However, higher NULL prediction rates in the dependency networks model indicate that the quality of the recommendation is higher for regular recommendations.

One should consider the difference between the product recommendation problem based on the historical purchase data and on the customer ratings. Since training e-VZpro depends on the association mining, it is not clear how e-VZpro is used when ratings are present. Recommender systems based on the dependency network are very straightforward to use for these issues and show remarkable results [Breese et al., 1998, Heckerman et al., 2000].

The biggest advantage of e-VZpro over dependency networks is the very short training times of e-VZpro. Practically all the training time, e.g. discovering the association rules, of e-VZpro is negligible. However the prediction time of the dependency networks is very short due to their very small stored models. The optimized nature of the Prediction Join operation in a relational database environment also contributes to these short prediction times [Microsoft, 2000]. Nevertheless, e-VZpro performed well in terms of computation time when using the sparse format.

Experiments were also run using other well-known recommender systems. To avoid excessive computational times associated with the user-based methods, only item-based methodologies were used. In the following subsection, we report results from the simple, yet very powerful, item-based recommender systems that we tested. Our results confirm again that these methods are very robust and superior to others. The algorithm based on the cosine similarity performed the best or second best across all datasets.

Table 8: Results from Cosine, Correlation and Popularity Based Recommender Systems

Method	WESTDB2		GENBUS		MSWeb	
	Accuracy (%)	Time (sec.)	Accuracy (%)	Time (sec.)	Accuracy (%)	Time (sec.)
Cosine	73.74	5	73.39	7	62.41	6
Correlation	46.55	6	44.31	7	50.51	6
Popularity	64.37	4	75.62	5	50.13	4

4.5 Results from Other Recommender Systems

This section contains the results from item-based algorithms using similarity measures based on cosine (Equation 2) and Pearson's correlation coefficient (Equation 1) in this section. Of course, in item-based cases similarities are computed between the items without changing the formulas given in Equations 1 and 2. This section also contains results from a popularity-based recommender system - while this last type of system is the most naïve, it is also very simple to implement and can be useful and powerful in certain cases.

Each of the methods described in this section is implemented in sparse format by adhering to the findings in [Karypis, 2001]. We report accuracy and computational time in Table 8. Computational time includes both finding similarities between items and evaluating each of the test cases. Sparse implementation is very effective in terms of the computational time as seen from the results.

As indicated in [Sarwar et al., 2001], there are two steps involved in item-based algorithms:

- Similarity computations between items.
- Prediction.

The prediction step is slightly different in our implementation. As in [Sarwar et al., 2001], prediction was based on a weighted sum. But, to find the overall similarity between the items in the active customer profile and a given item, we used only the non-negative correlation coefficients and discarded the negative correlations. Thus, normalization of the overall similarity was accomplished by dividing the number of non-zero correlation coefficients between the active customer profile and a given item. For the cosine similarity measure, no such change was needed, because these measures are between 0 and 1 for our datasets.

As the results in Table 8 indicate, item-based recommender systems are very efficient and robust in terms of both scalability and prediction accuracy. Cosine based algorithms, especially, are superior to other methods mentioned in this paper.

5 Future Work and Conclusion

A thorough comparison of several recommender systems is presented in this paper. Experimental studies on both internal Verizon data and publicly available benchmark data were conducted using sparse binary data. The results of the experiments indicate that simple methods recommend products very accurately when compared to well-known recommender systems based on machine learning communities and dependency networks introduced in [Breese et al., 1998, Heckerman et al., 2000]. We also proposed a new recommender system, e-VZpro, that is based on association mining. Although the training time of e-VZpro is negligible and it has provided effective predictions (which were improved further by the use of clustering), the current implementation does have the drawback of longer prediction times.

For modeling purposes, we used customer profiles without a time dimension. Incorporating the time information (i.e. purchase or service order times), the problem becomes one of sequence mining. This type of analysis may answer questions about what product/service the customer will purchase next and recommender systems using the two-step method of e-VZpro can be constructed to easily perform sequence mining. One important benefit of this type of recommender system is that it can be used to predict potential service cancellations and offer retention incentives to those at-risk customers.

A successful recommender system should also consider and model the acceptance and the decline rates of the products. At the beginning of the deployment of any recommender system such modeling perspective would be problematic due to the unavailability of the historical acceptance and decline rates. Modeling the acceptance and the decline rates will allow showing the different recommendations each time the customer interacts with the system. Recommender systems should also avoid from the NULL predictions. Na recommender systems such as the most frequent items should be used in this case.

A successful recommender system should also consider and model the acceptance and decline rates of the offered products. While the lack of historical acceptance/decline rates for newly deployed recommender systems can be problematic, modeling these rates will allow different recommendations to be presented to the customer each time they interact with the system. NULL predictions should be avoided in recommender systems; in these cases, a more naïve recommender system, such as a most frequent items system, should be used.

The most important consideration when deploying a commercial recommender system lies in determining the success metrics. Web-based recommender systems can track successes by measuring the clicks (convergence rates) but recommender systems for e-commerce sites should also incorporate the click-stream data in order to predict user needs and behavior to provide more accurate recommendations.

Acknowledgements

I would like to thank to Dr. Mohammed J. Zaki for his comments on an early draft of this paper, Dr. David Heckerman for his directions regarding the implementation of the dependency networks, Syed Rafiuddin and Wahib Omran for their valuable contributions in the preparation of the data, and numerous people in (former) E-Business Department for very productive discussions. Finally, I want to thank to Sam Witt and Ronald Brown for editing this paper.

References

[Ali et al., 1997] Ali, K., Manganaris, S., and Srikant, R. (1997). Partial classification using association rules. In Heckerman, D., Mannila, H., Pregibon, D., and Uthurusamy, R., editors, *Proceeding of Third International Conference on Knowledge Discovery in Databases and Data Mining*, pages 115–118. AAAI Press.

[Billsus and Pazzani, 1998] Billsus, D. and Pazzani, M. J. (August, 1998). Learning collaborative information filters. In *Proceedings of the 1998 Workshop on Recommender Systems.* AAAI Press.

[Breese et al., 1998] Breese, J., Heckerman, D., and Kadie, C. (July, 1998). Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence (UAI-98)*, pages 43–52.

[Goldberg et al., 1992] Goldberg, D., Nichols, D., Oki, B. M., and Terry, D. (1992). Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35(12):61–70.

[Heckerman et al., 2000] Heckerman, D., Chickering, D., Meek, C., Rounthwaite, R., and Kadie, C. (2000). Dependency networks for density estimation, collaborative filtering, and data visualization. *Journal of Machine Learning Research*, 1(Oct):49–75.

[Hettich and Bay, 1999] Hettich, S. and Bay, S. D. (1999). The UCI KDD archive. University of California, Irvine, Dept. of Information and Computer Sciences. http://kdd.ics.uci.edu.

[Karypis, 2001] Karypis, G. (2001). Evaluation of item-based top-n recommendation algorithms. In *Proceedings of the Tenth International Conference on Information and Knowledge Management (CIKM)*, Atlanta.

[Konstan et al., 1997] Konstan, J. A., Miller, B. N., Maltz, D., Herlocker, J. L., Gordon, L. R., and Riedl, J. (1997). Applying collaborative filtering to usenet news. *Communications of the ACM*, 40(3):77–87.

[Microsoft, 2000] Microsoft (2000). Introduction to ole db for data mining specification, version 1.0. Technical Manual. Microsoft Corporation, Redwood, WA.

[Pennock et al., 2000] Pennock, D. M., Horvitz, E., Lawrence, S., and Giles, C. L. (June, 2000). Collaborative filtering by personality diagnosis: A hybrid memory- and model-based approach. In *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence (UAI-2000)*, pages 473–480, Stanford, CA.

[Resnick and Varian, 1997] Resnick, P. and Varian, H. R. (1997). Recommender systems. *Communications of the ACM*, 40(3):56–58.

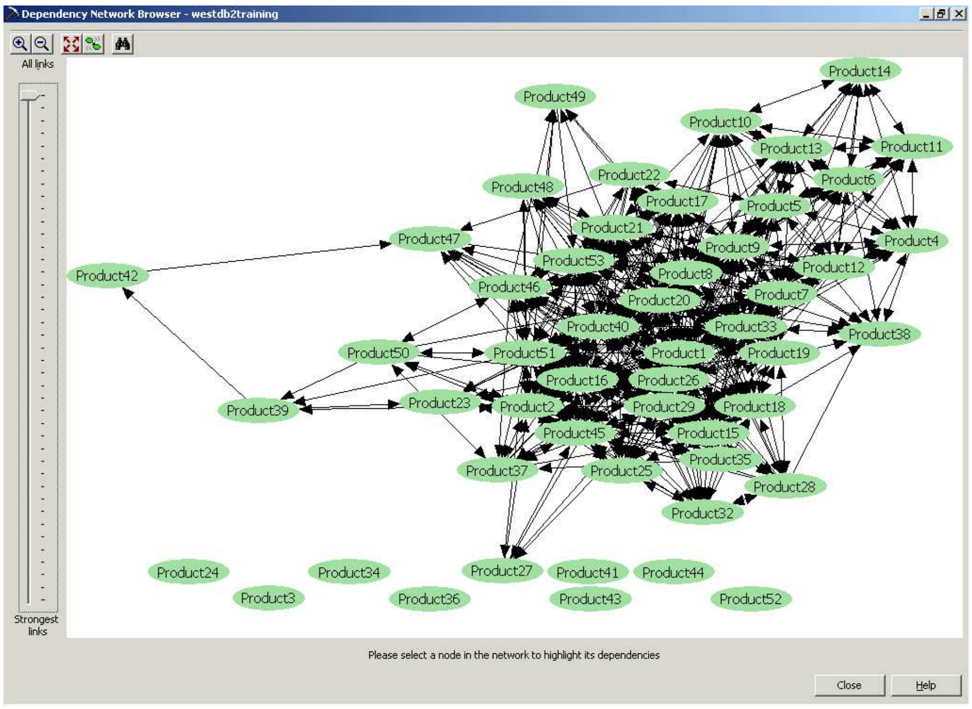
[Rucker and Polanco, 1997] Rucker, J. and Polanco, M. J. (1997). Personalized navigation for the web. *Communications of the ACM*, 40(3):73–75.

[Sarwar et al., 2001] Sarwar, B., Karypis, G., Konstan, J., and Reidl, J. (2001). Item-based collaborative filtering recommendation algorithms. In *Proceedings of the Tenth International World Wide Web Conference on World Wide Web*, pages 285–295.

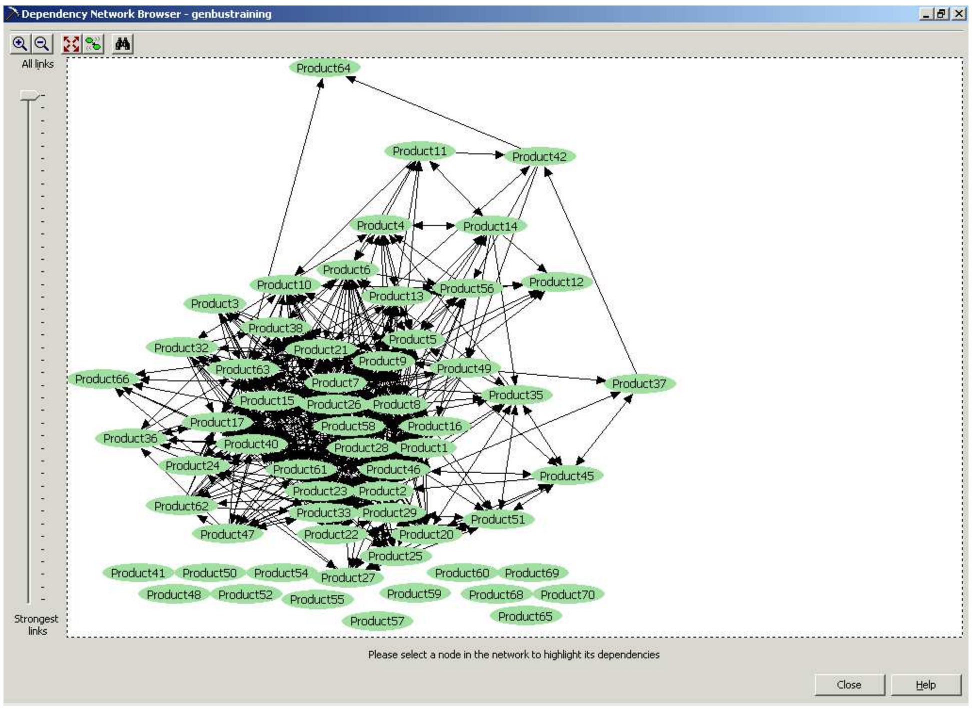
[Sarwar et al., 2000] Sarwar, B. M., Karypis, G., Konstan, J. A., and Riedl, J. (2000). Analysis of recommender algorithms for e-commerce. In *Proceedings of the 2nd ACM E-commerce Conference (EC'00).*

[Schafer et al., 2001] Schafer, J. B., Konstan, J., and Riedl, J. (2001). Electronic commerce recommender applications. *Journal of Data Mining and Knowledge Discovery*, 5(1/2):115–152.

[Ungar and Foster, 1998] Ungar, L. H. and Foster, D. P. (August, 1998). Clustering methods for collaborative filtering. In *Proceedings of the 1998 Workshop on Recommender Systems.* AAAI Press.



Please select a node in the network to highlight its dependencies



Please select a node in the network to highlight its dependencies

Figure 1: Dependency Networks: (A)- WESTDB2, (B)- GENBUS

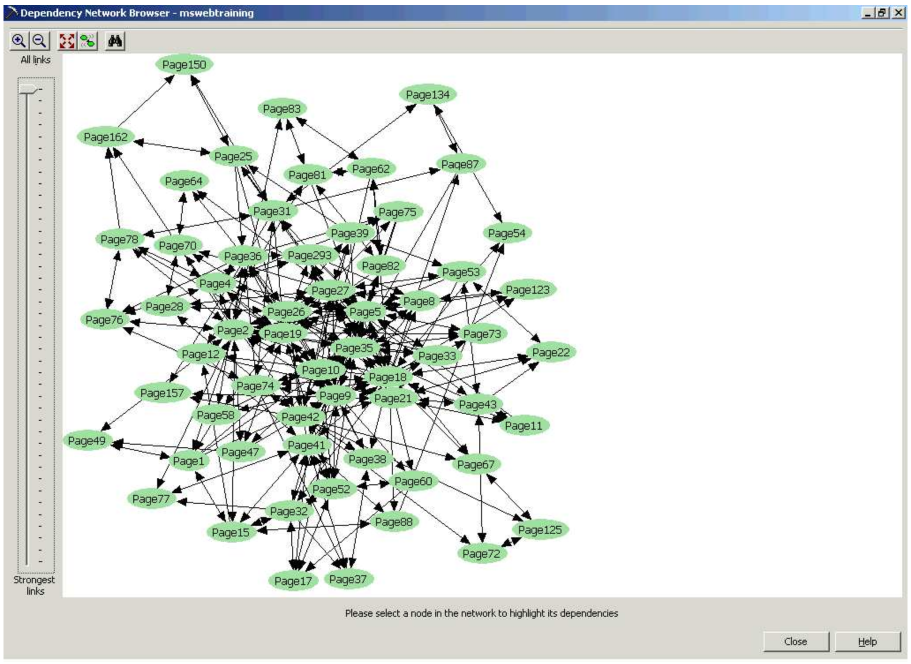
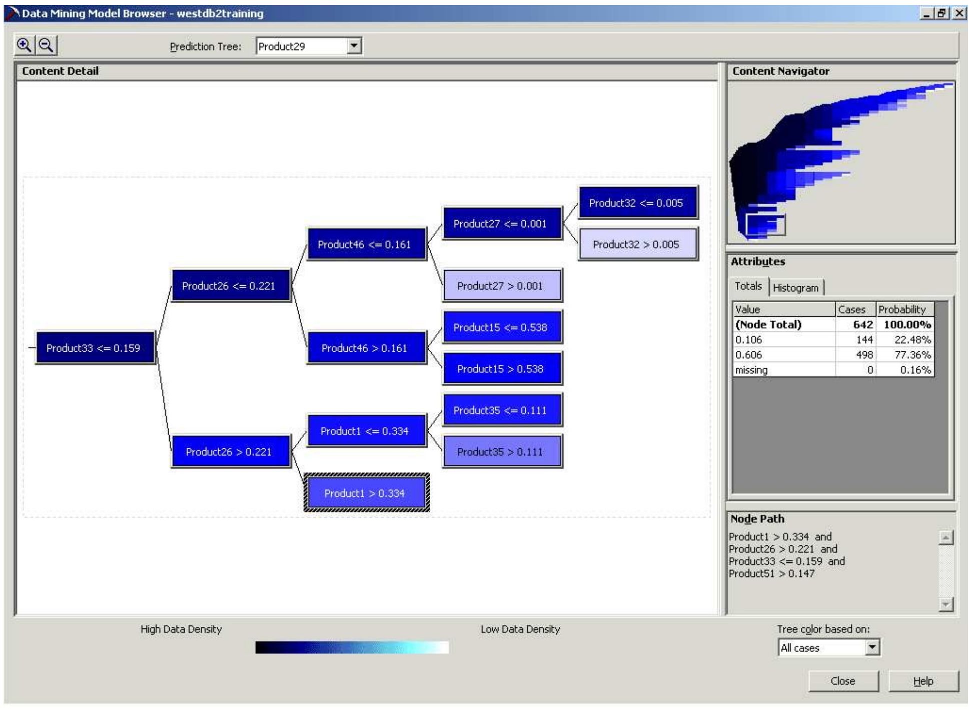


Figure 2: Dependency Networks: MSWeb



Prediction Tree: Product29

Value	Cases	Probability
(Node Total)	**642**	**100.00%**
0.106	144	22.48%
0.606	498	77.36%
missing	0	0.16%

Node Path

Product1 > 0.334 and
Product26 > 0.221 and
Product33 <= 0.159 and
Product51 > 0.147

Tree color based on: All cases

Figure 3: A Sample Decision Tree from WESTDB2 Model