Using Web Services to Support a Shared Knowledge Repository for Market Basket Analysis and Suggestive Sell Strategies

Marcia Vaughn
University of Wisconsin – Eau Claire
Eau Claire, WI 54701
vaughnma@uwec.edu

Paul J. Wagner
University of Wisconsin – Eau Claire
Eau Claire, WI 54701
wagnerpj@uwec.edu

Michael R. Wick
University of Wisconsin – Eau Claire
Eau Claire, WI 54701
wickmr@uwec.edu

Abstract

Any business marketing products or services can benefit from a data analysis of their customer purchase records looking for patterns in customer purchases. The results of this analysis can be used as a marketing strategy in future transactions to help suggest additional purchases to a customer prior to checkout. For the vast majority of companies, the cost to purchase and/or the expertise needed to develop this software is prohibitive. Further, even when a company can afford the software, smaller companies lack the volume of purchase records needed to make the analysis effective. This paper describes the development of a system that performs a probabilistic analysis of arbitrary purchase records for multiple clients, and captures the results in a shared knowledge repository. In addition, it describes the development of a suggestive sell component that uses this shared repository to suggest additional items for partially completed customer purchases.

Introduction

The Data Analysis Web Service (DAWS) system with a shared knowledge repository developed as part of this project is generic enough to work independent of the application domain, programming language, software, and database management system of the client.

DAWS uses a data mining technique called *market basket analysis* [1] to produce association rules for a given data set, and consists of two key software components: 1) the Market Basket Analysis (MBA) component, and 2) the Suggestive Sell (SS) component. Both components are deployed as a web service using the Internet for communication with clients. The MBA component accepts requests for data analysis. Each request includes a collection of completed purchase transactions and produces a set of association rules that can be used as a marketing tool. The SS component takes the contents of a specific customer's partial purchase and produces a list of probabilistically suggested additional products. Each component uses a custom protocol for communicating information between the client and DAWS. Both components are written in the C# programming language and run within the Microsoft .NET framework. The shared knowledge repository, which stores the association rules and can quickly produce a probabilistically likely suggestion given a set of purchases already selected by a customer, is a powerful source of information benefiting all clients that tap into its knowledge base.

Data Mining

Data mining refers to extracting or "mining" knowledge from large amounts of data. Data mining tasks can be classified into two categories: *descriptive* and *predictive*. Descriptive mining tasks characterize the general properties of the data in the database, whereas predictive mining tasks perform inference on the current data in order to make predictions. Predictive mining is used in this project by performing an *association analysis* on a given set of data. An association analysis is the discovery of *association rules* showing attribute-value conditions that occur frequently together in a given set of data; it is widely used for *market basket analysis*, a technique that analyzes customer buying habits based upon association rules. Association rules are of the form "$A_1 \Lambda \ldots \Lambda A_m \rightarrow B_1 \Lambda \ldots \Lambda B_n$" where A_i (for $i \epsilon \{1,\ldots,m\}$) and B_j (for $j \epsilon \{1,\ldots,n\}$) are attribute-value pairs. The association rule A=>B is interpreted as "database tuples that satisfy the conditions in A are also likely to satisfy the conditions in B". For example, a data mining system may find association rules like "popcorn => soda [7%, 60%]", where the support is 7%, and the confidence is 60%. The rule indicates that if a transaction contains "popcorn", there is a 60% chance that it contains "soda" as well, and 7% of all transactions contain both. "Strong" association rules satisfy both a minimum confidence and minimum support threshold. as set by the user.

A group of items purchased together is known as an *itemset*. If an itemset contained *k* items, it is called a *k*-itemset. An itemset satisfies minimum support if the number of individual transactions the itemset occurs in is greater than or equal to the product of the minimum support and the total number of transactions in the entire historical set of transactions. The minimum support count is the number of transactions required for the itemset to satisfy minimum support. The 2-itemset {popcorn, soda} is a *frequent itemset* if it satisfies the minimum support.

Association rule mining involves the process of first finding all frequent itemsets that occur at least as frequently as a pre-determined minimum support count, and then generating strong association rules from the frequent itemsets that satisfy minimum support and minimum confidence.

Market Basket Analysis

Market Basket Analysis is a boolean association rule mining technique that analyzes customer buying habits by finding associations between the items that customers place in their "shopping carts". The binary purchase records only take into account whether the item was purchased or not – the quantity of each individual item purchased is irrelevant (thus the term Boolean association). The discovery of such associations can help retailers develop effective marketing strategies by gaining insight into which items are frequently purchased together. This modeling technique is based on the theory that if you buy a certain group of items, you are likely to buy another group of items as well. For example, if you are at a hardware store and you buy a hammer, you are more likely to buy nails at the same time than somebody who didn't buy a hammer. In retail, many purchases are bought on impulse. Market Basket Analysis gives clues as to what a customer might have bought if the idea occurred to them. A marketing strategy, such as showing targeted advertisements on a web page, can then be used to attempt to encourage the consumer into purchasing the related item.

Market Basket Analysis often uses the well-known Apriori algorithm, who's name comes from the fact that it uses *prior* knowledge of frequent itemset properties, for mining frequent itemsets of Boolean association rules using candidate generation [1]. Apriori makes use of an iterative approach known as a level-wise search, where k-itemsets are used to determine (k+1) itemsets. Thus the Apriori algorithm is a variation of a dynamic programming algorithm in which frequent itemsets of size k+1 are determined from frequent itemsets of size k. The *Apriori property*, which states that all subsets of a frequent itemset must also be frequent, is used to improve the efficiency of the search. This property belongs to a special set of properties called *anti-monotone* because if a set cannot pass a test, all of its supersets will fail the same test as well. A two-step process consisting of *join* and *prune* actions is used to generate candidates:

> 1) The join is performed where members of the set are joinable if their first (k-2) items are in common. The assumption is made that all items within an itemset are sorted lexicographically.
>
> 2) Pruning reduces the size of the candidate set generated by using the Apriori property. Any (k-1)-itemset that is not frequent cannot be a subset of a frequent k-itemset. Therefore, if any (k-1)-subset of a candidate k-itemset is not in the previously generated (k-1)-itemset, then the candidate cannot be frequent either so it can be removed from the current candidate set.

Notice how the process is a combination of generate-and-test and dynamic programming. From the k-itemsets, the system produces a list of all possible k+1-itemsets. If any of these k+1-itemsets of a k-itemset as a subset that is not frequent, it is impossible for the k+1-itemset to be frequent and it is eliminated from consideration. In the end, the remaining k+1-itemsets that are possible must be compared to the full transaction set to see if they actually occur more than the minimum support necessary.

An example of this process is shown below. Using the following set of 9 transactions, a support of 22%, and a confidence of 70%:

Transactions		
A, B, E	B, D	B, C
A, B, D	A, C	B, C
A, C	A, B, C, E	A, B, C

First, the data is scanned to count the number of transactions containing each item, or candidate. Item A occurs in 6 transactions. Since a support of 22% is used, the minimum support count would be 22% of the 9 transactions, or 2. The candidate support count is then compared with the minimum support count. All of the items have a count of at least 2, so they are all kept.

Itemset	Count	Itemset	Count	Itemset	Count
{A}	6	{C}	6	{E}	2
{B}	7	{D}	2		

Next, a candidate set of 2-itemsets is generated by joining the frequent 1-itemsets with themselves. The transaction set is scanned and the support count is accumulated for each 2-itemset, or itemset containing 2 items. The 2-itemsets that do not meet the minimum support count of 2 are then removed.

Itemset	Count	Itemset	Count	Itemset	Count
{A, B}	4	{A, E}	2	{B, E}	2
{A, C}	4	{B, C}	4	{C, D}	0
{A, D}	1	{B, D}	2	{C, E}	1
{D, E}	0				

The process continues until all the frequent itemsets are found.

Itemset	Count	Itemset	Count	Itemset	Count
{A}	6	{C}	6	{E}	2
{B}	7	{D}	2	{A, B}	4
{A, E}	2	{B, E}	2	{A, D}	1
{A, C}	4	{B, C}	4	{C, E}	1
{B, D}	2	{A, B, C}	2	{A, B, E}	2

Then it is straightforward to generate strong association rules from them using the confidence equation:

Confidence (A => B) = P(B|A) = support count (A U B)/support count (A)

If all non-empty subsets are generated from one of the frequent 3-itemsets, { A, B, C }, the following association rules are generated:

A ^ B => E confidence = 2/4 = 50%	A ^ E => B confidence = 2/2 = 100%
B ^ E => A confidence = 2/2 = 100%	A => B ^ E confidence = 2/6 = 33%
B => A ^ E confidence = 2/7 = 29%	E => A ^ B confidence = 2/2 = 100%

What these rules state is that for every transaction that contains both A and E, 100% of the time it also contains **B**. All rules that do not meet the minimum confidence of 70% are discarded.

Data Analysis Web Service

Web services allow for the creation of components whose methods can be easily invoked over the Internet. A web service is the ideal solution for providing access to this data analysis system, as any application that understands and can communicate via *Simple Object Access Protocol (SOAP)* [2] can take advantage of it. Other advantages of web services include their scalability, accessibility, ability to provide a means of advertising their existence via *Universal Description, Discovery, and Integration (UDDI)* [2], and allow for implementation of a custom protocol via *Web Service Description Language (WSDL)* [2].

SOAP is a lightweight, message-based platform-independent protocol built on XML, HTTP, and SMTP designed for communication with web services. Two other protocols offer additional convenience and functionality for a client using a web service: 1) UDDI provides a description of all web services available at a particular site or URL, and 2) WSDL provides a description of the available methods, or the interface, provided by a web service that can be used by clients.

Using a web service is straightforward – client code is written as though it were communicating directly with the host server by means of a URL. However, in reality the client interacts with a *proxy*, whose only responsibility is to represent the server on the client machine, create SOAP messages out of client requests that are sent on to the server, and to retrieve the responses containing the result.

The DAWS implementation involved creating two web methods – one for clients requesting a data analysis of their entire historical transaction set, and one for clients requesting a list of suggestions for a specific "shopping cart". These two aspects are described in the following sections.

Market Basket Analysis Component

The Market Basket Analysis (MBA) component, which is deployed as a web service, accepts requests for data analysis over the Internet. An accepted request consists of a collection of prior completed

"shopping cart" transactions. An analysis is then done on these transactions, with the result being a set of association rules containing antecedents and consequents. These rules are then saved in the shared knowledge repository, further increasing its knowledge base, and could also be returned to the client upon request to use in its own client-specific marketing strategies. The use of a common *product code* allows all clients to benefit from the increased validity of the suggestions produced as the data sample continually increases in size with each additional client requesting a data analysis. The product code used is the Universal Product Code (UPC), as it is unique and available for all items. This is a key point to the uniqueness of this system, as it allows all clients using UPC codes for the items they market to participate in the sharing of their data analysis to benefit all.

The requests are accepted via the web method DataAnalysisRequest() within the MBA system, which takes in a transaction set to be analyzed as a matrix of integers. Each row in the matrix corresponds to the product codes purchased in a single transaction. The client will be returned a client identification number in the form of a positive integer if the request was received and sent off to be processed successfully. A negative result being returned signifies there were problems in receiving or processing the request. The matrix is read into a specialized object used by the MBA component. Each itemset within the object holds a sorted list of items, with each item consisting of a unique item id. The item id is the unique UPC code assigned to that item.

A configuration interface is provided that allows DAWS to be flexible enough to be used with any available combination of data analysis, algorithm, and shared repository implemented. The MBA component is implemented using market basket analysis, the Apriori algorithm, and a specifically designed database shared repository. Other information specified in the configuration object includes the minimum support count, minimum confidence, and maximum consequent size. The system was designed with reusability in mind to allow other data analysis implementations to provide their own custom types of analysis, algorithms, repositories, or data structures.

The MBA component then uses the Apriori algorithm to generate candidate sets and passes the resulting frequent itemsets in to a rule set miner where the strong association rules are generated. The rules are then added to the shared knowledge repository (described in detail in Section 5).

When adding rules, the database is first queried to see if the rule currently exists using the SQL query "SELECT distinct {*RuleConsequentTable*} from {*RuleConsequentTable*} WHERE {*RuleID*} in (SELECT {*RuleID*} from {*RuleAntecedentTable*} WHERE {*AntecedentItemNumber*} in ({*AntecedentItemNumbers List*}) GROUP by {*RuleID*} HAVING COUNT(*) = {*Size of AntecedentItemNumbers List*})". If the rule does not exist, then the SQL query used to insert rules into the database is "INSERT into {*RuleAntecedentTable*} VALUES ({*RuleID*}, {*AntecedentItemNumber*})".

Suggestive Sell Component

The Suggestive Sell (SS) component likewise accepts requests over the Internet via a specialized protocol. This component takes the current "shopping cart" contents and sends them to the shared knowledge repository, which then returns a list of probabilistically selected additional products to suggest.

The requests are accepted via the web method SuggestionsRequest(), which takes in the "shopping cart", to be analyzed as an array of integers. The "shopping cart" is then used by the SS component to create a basket containing an itemset that holds on to the items in the current shopping cart contents. These items contain the UPC code just as they do in the MBA component. The suggestion set returned by the SS component is in the form of an array of integers representing UPC codes that could be suggested as possible additional purchases.

The SQL query used to retrieve this information is "SELECT distinct {*ConsequentItemNumber*} from {*RuleConsequentTable*} where {*RuleID*} in (SELECT {*RuleID*} from {*RuleAntecedentTable*} minus SELECT distinct {*RuleID*} from {*RuleAntecedentTable*} where {*AntecedentItemNumber*} not in ({*List of ItemsPurchased*})".

Shared Knowledge Repository

The shared knowledge repository is a multi-user, high-performance, scalable system that can handle updating the repository and generating suggestive sell suggestions for an arbitrary number of clients simultaneously.

The schema for the suggestive sell rules database repository is fairly simple, and includes the following three tables:

RULE table		
RuleID	MinConfidence	MinSupport

RULE-ANTECEDENT table	
RuleID	AntecedentItemNumber

RULE-CONSEQUENT table	
RuleID	ConsequentItemNumber

New rules are stored as they are generated by an arbitrary rule number in the Rule table with their associated confidence and support levels. The antecedent and consequent item numbers are each stored individually in a row of the Rule-Antecedent or Rule-Consequent tables, in order to simplify rule matching for the Suggestive Sell component. Multiple antecedents are currently supported, but only a single consequent per rule. Conjunction of all antecedents is assumed at this point. More complex antecedent expressions would require further evolution of the database schema.

This repository was implemented using the Oracle 9.2.0 database management system.

Conclusions

The development of this project has produced significant outcomes. First, it makes the powerful marketing tools of market basket analysis and suggestive sell available to even the smallest companies. Second, the development of a shared knowledge repository of association rules that can be used by multiple simultaneous clients significantly increases the quantity of suggestive sell rules that are produced and available to all clients. Thus even larger companies will benefit from contribution to and application of this shared repository. Third, the centralized definition of the MBA and SS components offloads the responsibility for maintenance and improvement of the probabilistic algorithms to a single agent. The resulting savings to companies that currently maintain their own similar components will be significant. Finally, developing this project using design patterns [3] creates a system that can be easily adapted to varying data analysis and shared knowledge repository implementations.

References

1. Han, J. and Kamber, M., Data Mining – Concepts and Techniques, Morgan Kaufmann Publishing, 2001.
2. Anderson, R., Francis, B., Homer, A., Howard, R., Sussman, D., Watson, K. Professional ASP.NET 1.0, Wrox Press, 2002.
3. Gamma, E., Helm, R., Johnson, R., and Vlissides, J., Design Patterns, Elements of Reusable Object-Oriented Software, Addison-Wesley Publishing, 1995.