Chapter 6

# Association and Market Baskets

## Chapter Outline

## What You Will Learn in This Chapter

- What products are purchased together?
- What is association or market basket analysis?
- Does association analysis really find useful rules?
- Why do association routines have parameters and what values should be selected?
- Do association rule systems return all of the interesting rules?
- How does data need to be organized for association analysis?
- How difficult is it to create and interpret association rules?
- How does the Microsoft Association Rules tool differ from traditional tools?

**Netflix**

Amazon.com demonstrated to other vendors the importance of its recommendation engine. By showing books and other items purchased by other customers, sales can be increased by bringing new items to the attention of customers. Netflix relies on a recommendation system to help customers find movies to rent. Movie rental is an interesting business: millions of people want to rent the same movie, when it is released. Then demand for a specific movie quickly drops off to low levels. Companies that rely on physical media (DVD and Blu-Ray) face the problem of spending money for a large number of initial copies to keep customers happy during the initial rush. (U.S. rental companies are required by law to rent original discs, not copies.) Netflix attempts to manage the demand for new releases through control lists and by encouraging customers to rent other movies. A key step to encouraging the adoption of other movies is the recommendation engine which suggests movies that the viewer might like—based on ratings of previous movies. In late 2006, Netflix created a contest to encourage people to create a new recommendation engine. Anyone who could beat the existing approach by at least 10 percent would win $1 million. In the end, in 2009, a team of researchers won the prize—by combining a variety of approaches. Most of the researchers began the contest separately, and each achieved small improvements. By combining several approaches and working together, the BellKor Pragmatic Chaos seven-man multinational team, largely led by AT&T Research found a way to beat the older models. In the last 30 days, the other competitors all joined together to form a second team (Ensemble)—which also beat the 10 percent requirement and almost won the prize. [Copeland 2009] It is difficult to determine the ultimate impact of the algorithm on Netflix because many factors have changed over time, but there is little doubt that Netflix is the current leader in movie rentals, and total revenue increased from $1.7 million in 2009 to $3.2 million in 2011. [Income statements]

Researchers continue to develop new data mining techniques and better methods for analyzing data. The association problem and recommendation engines are key tools for many businesses.

Michael V. Copeland, "Box Office Boffo for Brainiacs: The Netflix Prize," *Fortune*, September 21, 2009. http://tech.fortune.cnn.com/2009/09/21/box-office-boffo-for-brainiacs-the-netflix-prize/

## Introduction

**What products are purchased together?** This classic business question seems easy to answer. It certainly has many applications in business—from determining product placement in a store to making recommendations for additional purchases. Amazon was one of the first vendors to highlight its strength by recommending additional books to customers. Netflix pushed the curve with a million-dollar contest to anyone who created a substantially better recommendation engine. Better answers to the question can improve ordering and stocking decisions, reduce inventory costs, increase sales, and improve advertising targets.

The one catch is that the problem is relatively hard to answer—particularly with giant datasets available to retail merchants. The best computational algorithms are amazingly fast, but they gain speed by making simplifying assumptions. Still, the market basket algorithms are useful. Plus, once the data is organized correctly, the routines are almost completely automatic. The tools in this chapter can be run as **unsupervised learning** systems. Additionally, the output is relatively easy to understand, where associations are described as **rules**, such as customers who purchased Pies are also likely to purchase Cakes, and the strength of the association is given a number expressing the degree of confidence in the rule.

The basic output of association analysis consists of a set of rules of the form A → B which can be read as "A implies B" or in market terms: If a customer buys A, then they are likely to buy B. The rule also contains one or two numbers describing the strength of the relationship. Often the numbers are conditional probabilities, describing the frequency in the database where B followed A.

Dimensionality is the biggest issue faced with association tools. A sales database could have billions of rows of sales, with hundreds of thousands of products. The number of rows is less of a problem than the number of products (dimensions). Rows generally increase processing time linearly. Dimensions increase processing time exponentially. Even if an algorithm and system are fast enough to compare and identify relationships among so many products and sales, how can a human ever evaluate or understand the resulting rules? Hence, a key step in most association analyses is to find a way to reduce the complexity of the data. For instance, when examining sales in a hardware store, does it really matter if a customer buys 3-inch nails versus 2-inch nails? Or, is it sufficient to know that the customer purchased nails (with that hammer)? In other words, the analyst has to determine the appropriate product level for the research. If the size of the nails is unimportant, the analyses can be conducted at the level of product category. If the size is critical, the problem can be reduced by looking at fewer items, instead of the entire inventory.

The point is that analysis often has to be performed at multiple levels. In the hardware example, with tens of thousands of product variations, a high-level analysis might examine the categories of items purchased (hammer, nail, board, and so on). Then, departmental managers might want to examine sales at more detailed levels. For instance, did people buy the right hammers for the nails purchased? In this case, a lower-level analysis can look at the detailed items (SKUs) by limiting the items to those within a specific department.

Most of the examples in this chapter are oriented towards traditional market basket analyses—because the results are easy to understand. However, keep in mind that the tools can be used for more complex problems. Think in terms of association—identifying which events tend to happen together. In this context, the tools could be used to examine things such as customer attributes, stock price events, medical treatments, and even crime events.

## Business Situation

**What is association or market basket analysis? Data associations** can be defined loosely as events or situations that tend to happen together. In business, these events are often captured as transactions. In an accounting context, transactions typically consist of sales and purchases. **Market basket** analysis is a subset of association analysis that focuses specifically on products that were purchased at the same time—in the same market basket. In a broader context, the events could include almost anything—such as responses to marketing campaigns (marketing), product failures in different environments (operations), sales of stock to various groups (finance), security attacks or technical support requests (MIS), and evaluations and promotions (HRM). Sales and purchases (costs) are the most commonly studied business events—because the data is readily available. Bar code scanners automatically capture the list of items purchased by each customer in one sale. The detailed data is generally available in giant databases organized by store, date, region, and sometimes by customer—when customers use frequent-buyer cards. An increasingly popular example is to use association rules as the foundation of a **recommendation engine** such as those used by Amazon to suggest similar books and Netflix to recommend movies.

The classic example of market basket data mining might not be true (no one seems to know who actually ran the study). Nonetheless, it represents an interesting example of the potential results. Supposedly a chain of stores (e.g., 7-eleven) analyzed sales on weekend nights and the association tool found a strong rule that baby diapers and beer were commonly purchased together. Presumably father went out for diapers and picked up beer at the same time. The story sounds plausible, and lacking real data to test it (or a *Mythbusters* evaluation), it is likely to remain as the classic example. The beauty of the story is that (1) it could be true, (2) the results are something that could be found through market basket analysis, and (3) everyone can understand the implications. So, even if it is not true, it makes a good example. If you run a market basket association analysis and see a rule of the form (diapers => beer), the manager needs to think about what the rule means in business terms. For example, moving beer and diapers to the same aisle might increase sales of both. Or, perhaps putting snack items between the two products would lead to increased sales of snacks because of the improved visibility to these customers. The key point is that identifying potential rules is a good first step, but background knowledge of the topic is required to understand the value of the rule and put it to good use.

One of the issues quickly faced by anyone running association analysis is that the tools tend to turn out dozens, hundreds, or even thousands of rules. Some of the rules are interesting, some are not. Some can be used to increase sales or reduce costs, but others are simply curiosities. It is important for managers to read the rules and determine which relationships make sense for the particular situation. Fortunately, the tools are easy enough to set up and run that a manager with little formal training in data mining can make use of the tool. Today, some specialized firms provide data mining services where they take copies of the sales data and run the results on their customized high-speed servers. So managers only deal with the results. Of course, these firms charge fees, so most smaller problems could be handled without them—as long as the amount of data is reasonable.

| | |
|---|---|
| Bread | Muffin |
| Cake | Pastry |
| Candy | Pie |
| Cookie | Rolls |
| Crepes | Scone |
| Cupcake | Sweet Rolls |
| Miscellaneous | |

**Figure 6.1**

Categories for the bakery. Each category has many products. For example, pies can be chocolate, berry, lemon, coconut, and so on. Categories and important method for reducing dimensionality.

Market basket rules for shopping need a fairly large amount of data to estimate and the products need to be reasonably consistent over the time frame of the analysis. The Rolling Thunder Bicycle case has plenty of data, but the actual products change every couple of years as the vendors put out new versions of components. Additionally, the bicycle component data is usually constrained by the groups—customers almost always buy all of the products within one of the primary Shimano or Campagnolo groups. Consequently, the strongest associations are for products within the groups—which is a dull result.

## The Bakery

To provide a more interesting set of results—particularly for association—the bakery case was created with simulated sales over many years. A bakery is similar to the classic market basket problem: Many different products, but they fall into defined categories. Figure 6.1 lists the categories for the bakery—they include the common items encountered at most bakeries. Each individual product is classified into one of the categories. For example, the bakery produces 22 varieties of cakes including White, Angel, Sponge, Carrot, and German Chocolate. The initial bakery data contains over 140 individual products, but they are grouped into these 13 categories. The categories are important in determining association rules. As explained in the Model section, having too many dimensions makes the problem exceedingly difficult to solve—even with high-speed computers. Grouping products into a smaller number of categories speeds the analysis.

In reality, 140 items is a relatively small number and can be handled by the systems in a short time. More importantly, it is unlikely that important relationships exist at the detailed product level. And if they do exist, is it possible to determine the meaning? What does it mean if a rule states that customers who buy Spice cake often buy chocolate chip cookies? And, how would managers deal with the potentially hundreds or thousands of similar rules? For books or movies these detailed associations would be useful. If a customer buys a specific book or movie, the customer will want to see specific other books. But at the bakery, if a customer buys a Spice cake, it makes more sense to suggest cookies in general instead of just one variety of cookies.

| | |
|---|---|
| Cookie -> Muffin | (13.3, 30.8) |
| Rolls -> Muffin | (10.5, 24.4) |
| Cake -> Pie | ( 8.1, 19.7) |

**Figure 6.2**

Sample rules for bakery. The rule is read from left-to-right, so that customers who purchased a cake tended to also purchase a pie. This rule lists support of 19.5 percent with a confidence of 41.5 percent.

Figure 6.2 shows sample results based on a limited set of data for the bakery categories. The **association rules** are read in the direction of the implication arrow (->). A few tools write the rules from right-to-left instead, but left-to-right is easier to read. For example, the first rule states that customers who bought a cookie were likely to purchase a muffin. The numbers in parentheses are the support and confidence values which are explained in detail in the Model section. Essentially, the support is the frequency of that combination appearing in the sample. So 13.3 percent of the transactions included purchases of both cookies and muffins. The confidence is an estimate of the conditional probability of the outcome given that the first event has already occurred. For instance, the probability that someone will purchase a muffin given that a cookie has already been purchased is estimated to be 30.8 percent. Of course, the numbers apply only to the sample and there is no guarantee they will apply to future purchases. This point is particularly critical if other factors might be influencing the choices.

These sample rules are relatively simple in that they represent pairs of item categories. More complex rules can include multiple categories on the left side, such as

```
Scone, Sweet Rolls -> Muffin
```

This sample rule reads: customers who purchased both a scone and sweet rolls were also likely to purchase muffins. In theory, any combination of the dimensions could form the **antecedent** (left side) of the rule. Multiple dimensions quickly lead to a huge number of possible combinations. Beyond the computational problems, the rules are also difficult to understand. Picture the challenges of examining the bakery data with 141 products if the results led to rules with 100 items in the antecedent list. Now think about a modern superstore with hundreds of thousands of detailed products and imagine the potentially convoluted results.

## Product and Dimension Levels

The analyst must choose the level at which to conduct the analysis to match the business problem. This decision can be based on business factors with input from managers. Of course, in many situations it is possible to conduct the analysis at several levels. Start at the top (category) level and work down to more detail. For huge stores, detailed analyses can be conducted within groupings. For instance, in a discount store it might make sense to do market basket analysis just within the toy department. Customers who purchased a specific doll might also purchase specific clothing or accessories. Working within the department would miss rules that cross boundaries. Perhaps those customers who bought that doll and clothing

| | | |
|---|---|---|
| 1132, 1053, 893, 757, … | Store | Are sales at one store associated with sales at another store? |
| Toys, Garden, Auto, Shoes,… | Department | Do sales in one department increase sales in others? Good for loss leaders. |
| Dolls, Balls, Trucks, Stuffed, … | Category | Are categories of items commonly purchased together? Product display. |
| Girls, Boys, Baby, Fashion, … | Subcategory | Do subcategories affect each other? Handled within departments. |
| Barbie, Princess, Corolle, … | Product | Which items are purchased together? Usually within department or category. |
| Thumbelina, Supergirl, … | Size, Color | Which color items are associated? Useful for clothing. |

**Figure 6.3**

Sample levels of analysis. Start at the top and work down to more detail. Detail levels might have to be run within a sublevel instead of across the entire data set. For example, product sales can be examined within a department, but probably not at the store level.

also purchased cookie dough at the same time. But, because the store is organized in departments, even if some unusual cross-departmental patterns exist, it would be difficult to act on them. Figure 6.3 presents some examples of levels for a department or discount store. The questions to be answered vary by level or grouping. The Figure provides some examples to highlight the differences, but the list is endless.

Product hierarchies are important in most businesses. Simply to keep the data easier to understand, most organizations section the data into groups. Products are sold by country, region, store, and department. Products are grouped into departments, manufacturers, and subcategories. Individual items often have multiple sizes or colors. Sometimes the variations get out of hand. Does any store really need to carry 30 or more varieties of Oreo cookies? All of these groupings are useful candidates for aggregating sales before attempting to run the association analysis. The results for each grouping can provide different interpretations. Are customer choices of vendors related? Do some departments bring in customers who then buy profitable items from other areas?

Hierarchies exist in most other organizations—because humans routinely use them to reduce complexity. Workers are organized into groups—do some groups affect outcomes (e.g., quality) with other groups? Are some managers more closely associated with other managers in terms of assigning raises? Similarly, finance divides investments into categories, including broad categories of stocks, bonds, and derivatives. Each major grouping can be divided into sectors, duration, rating, or other categories. Starting with the top categories can provide guidance on where to look within sublevels.

## Model

**Does association analysis really find useful rules?** This particular question is probably the most important one regarding association analysis, but it is extremely difficult to answer. One approach might be to say that based on ex-

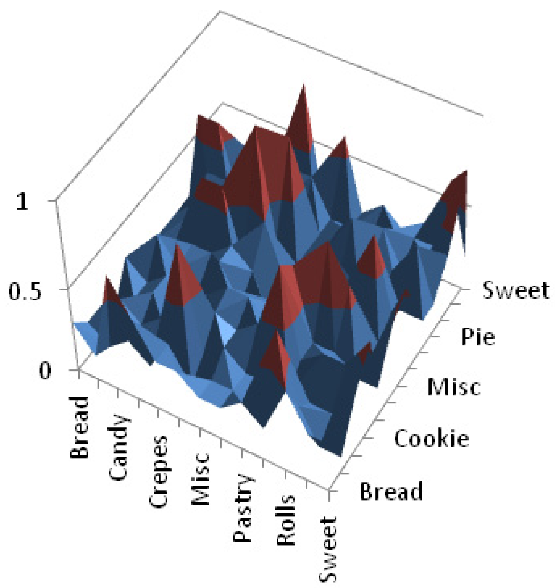


**Figure 6.4**

Frequency for pairs. The peaks or bumps show points where items are often purchased together. The chart shows the correlations between categories. The goal of association analysis is to find the peaks and identify interesting pairs.

perience, yes, association analysis has found useful rules in many situations. But, at heart, association analysis searches for patterns that arise often in the sample data set. There is no assurance that high frequency is associated with usefulness or even interestingness. For example, a sales analysis of a fast food chain would most likely reveal that people who purchase burgers (or almost any other entrée) also purchase fries. Having the sales clerks always ask “Would you like fries with that?” is going to skew the results. But, that is the whole reason for continually asking customers—to increase the sales of profitable products. Is there usefulness or interestingness or surprise in having a data mining tool find that relationship? Perhaps confirmation is important. But, what if that single rule is so dominant that it hides other associations?

The fry association is a fairly blatant example, but similar outcomes can arise in other industries. More importantly, the question and the example highlight the key aspect of association rules: The tools need to assign numbers to the relationships that have value to managers, are easy (fast) to compute, and can sort and differentiate the rules. These questions are covered in the model section, and each tool has methods of trying to handle the complex interactions.

## Goal

Ultimately, the goal of association data mining is to find “bumps” in the frequency distribution of the transactions data. The objective is to find those points or combination of items that stand out from the others. Figure 6.4 illustrates the concept by

| Measure | Definition |
|---|---|
| Support | P(A and B) |
| Confidence | P(B \| A)<br>P(A and B) / P(A) |
| Lift | P(A and B) / [P(A)P(B)] |
| Relative Risk | P(B \| A) / P(B \| ~A) |

**Figure 6.5**

Traditional measures and common definitions. A and B represent events or itemsets of products. Support is the number of times the events occur in the sample divided by the total number of transactions (frequency). Confidence is an estimate of the conditional probability. Lift is an estimate of the gain that might be expected for B if event A occurs.

comparing sample sales for pairs of categories. The peaks are likely to be the most interesting points to business managers. They represent combinations of product categories that are often sold together. Note that this figure represents the simplest combination of categories—pairs. And it still requires combining every category with every other category and finding the resulting correlation data for each point.

Even if it were possible to create a multidimensional picture similar to Figure 6.4 for every combination of categories, it would still be difficult to interpret. The visualization of the peaks is interesting, but if hundreds or thousands of peaks (rules) appear, it would be difficult to examine or understand them. Instead, the data mining tools assign numbers to the combinations which enable the analyst to sort and compare the various rules. The numbers are also commonly used to cut off the display lists to keep them to manageable size. Hence, it is important to understand what the numbers mean because they control which rules are displayed.

## Assigning Values to Rules

At heart, association rule algorithms examine combinations of attributes, assign a value, and then display the combinations with the highest values. It is then up to the analyst and managers to decide if the resulting rules make any sense, and can inspire changes that will increase sales and profits. Yet, the two main steps of the algorithms leave plenty of room for differences among tools. They differ based on how the search is performed and on the measures used to value each combination. The measurement issue is the most important, because it affects the ultimate question of whether the rules are meaningful. The objective is to find an easy-to-compute value that identifies interesting and meaningful rules for managers.

Several measures are used for identifying frequent combinations. The confusing part is that data mining research has created names for them, but some systems use the same names to represent different calculations. Hence, it is helpful to look at the mathematical definitions because they are straightforward and easier to understand when comparing tools. Figure 6.5 shows the most common measures used by association tools. The first two were defined in the original association analysis paper by Agrawal (1995). The relative risk term is less common, but it is a key element in Microsoft's association model. This list is a small portion of the concepts that have been proposed. Geng and Hamilton (2006) provide a good introduction to the concept of **interestingness** measures and list almost 30 variations.

Combinations of attributes or products are called **itemsets** and the letters A and B are commonly used to represent different itemsets. In market basket terms, an itemset consists of combinations of different products. Typically, the A itemset is the antecedent of a proposed rule, and the B itemset (often a single item) is the proposed result. So, a rule might be evaluated that proposes the presence of apples (A) leads to the purchase of bananas (B).

### *Support*

**Support** is a critical measure. It is the frequency of the itemset within the sample: The number of times the specified set occurs divided by the total number of observations. Support can be computed by counting the number of transactions containing all of the itemsets specified in the parameter. Support is sometimes denoted with a sigma function, such as $\sigma(A)$, but because it represents an estimate of the probability, it is most commonly written as a probability measure P(A). In association analysis, the most important support measure is $P(A \cap B)$ which is the probability or frequency of both A and B itemsets appearing in transactions together. In words, support represents the relative number of times the proposed rule actually occurs in the sample. So, high support values indicate that the combined itemsets often appear together. Support is a critical measure in most of the data mining algorithms. It is one of the primary tuning parameters used to reduce the number of rules to examine. Potential rules with low support are discarded from the analysis.

### *Confidence*

Remembering probability rules, the two itemsets can be interchanged and the support level will be the same: $P(A \cap B) = P(B \cap A)$. So how can a rule claim that one product leads to the other? By itself, support cannot. Support simply indicates that the two itemsets occur together, it says nothing about which might lead to the other. On the other hand, **confidence** is an estimate of $P(B \mid A)$ and equals support(A and B)/support(A). Because of the divisor, $P(B \mid A)$ is different from $P(A \mid B)$.The conditional values is interpreted as the probability that itemset B will arise given that itemset A has already happened. Hence the name (conditional). High values of the conditional probability provide confidence that the combination of events will happen in future transactions. Because of its one-way nature, confidence is a better measure of interestingness than support.

### *Lift*

It is possible to build an association data mining tool using just support and confidence measures. The results are ranked by confidence values and they are relatively easy to interpret. However, most tools include more measures. Lift is a common term that attempts to measure the impact of the rule. **Lift** is best defined as $P(B \mid A) / P(B)$. This ratio measures the probability of item B being purchased with the rule (A already chosen) versus without the rule—B by itself. Values greater than one indicate a positive effect, and values less than one reveal that the presence of itemset A reduces the probability of purchasing itemset B. The ratio is read as a measure of the probability gain. For example, if lift = 1.2, then the presence of the rule and itemset A leads to a 20 percent increase in the probability that B will be purchased, compared to a customer who does not already have itemset A in the basket. This estimate can be used to decide how much value might be gained by

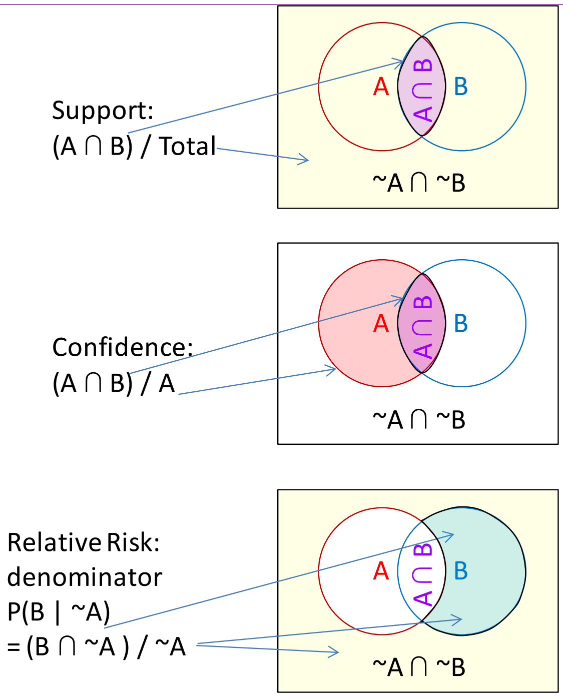


**Figure 6.6**

Venn diagram of primary measures. The values are the ratios of the indicated colored areas.

convincing customers to purchase itemset A. The accompanying 20 percent probability increase can be used to estimate the expected gain in sales of B, hence the expected gain in profit. A portion of this expected gain could be spent on marketing A, reorganizing the store or Web site to increase sales, or decreasing the price of A.

## *Relative Risk*

**Relative risk** is another of many interestingness measures. It is included in this chapter (versus the 20 or 30 other measures not included) because it is a major element in Microsoft association analysis. It is conceptually similar to lift because it is trying to measure the gain in probability of purchasing B when itemset A is present. The difference is in the comparator. Lift compares the gain to the total probability that B is purchased. Relative risk computes the gain from the probability that B is included when A is not purchased. The formula for relative risk is: P(B | A) / P(B | ~A). Recall that lift just used P(B) in the denominator. Here, P(B | ~A) is the probability that B is purchased given that itemset A is not (~) present. The interpretation is similar to that for lift. The ratio measures the relative difference in the probability of purchasing B with and without A present. If the ratio is larger than one, then the effect of A is positive, otherwise it is negative. The ratio measures the percentage gain in the probability of purchasing itemset B when itemset A is present.

For the most part, lift and relative risk are comparable. However, relative risk does not work well if A and B consist of all possible items. If it seems unlikely

| | | | |
|---|---|---|---|
| Pie | Sweet Rolls | Cake | Sweet Rolls |
| Sweet Rolls | Cookie | Cookie | Muffin |
| Pie | Pastry | Cookie | Sweet Rolls |
| Pastry | Cookie | Cookie | Muffin |
| Cupcake | Cake | Bread | Pie |
| Cookie | Sweet Rolls | Sweet Rolls | |
| Muffin | Cake | Cake | Pie |
| Cake | Cupcake | Pastry | |
| Rolls | Cake | Pie | Pie |
| Pastry | Cake | Rolls | |

**Figure 6.7**

Small sample of 10 transactions and 6 categories. Sample single support: Muffin shows up in 3 of the 10 transactions for a support of 0.3. As an example of pairs, Pie and Cake appear together 4 times for a support of 0.4.

that someone would conduct data mining on only two items, remember that A and B are itemsets, and itemset A could consist of everything except B. A little probability manipulation is needed. Look at the denominator:

```
P(B | ~A) = P(B ∩ ~A) / P(~A)
P(~A) = P(B ∩ ~A)  + P(~B ∩ ~A)
If A and B are all the items, then P(~B ∩ ~A) = 0, so
P(B|~A)=1
Reducing relative risk to P(B | A) or confidence.
```

Ending up with relative risk equal to confidence is not a major disaster, but it does change the interpretation of the concept. Just keep in mind that if the itemsets A and B are large and represent a substantial portion of all transactions, relative risk is less useful.

### *Venn Diagram*

Sometimes visual depictions of the concepts are easier to understand and remember. Besides, if the SAT was the last time you saw Venn diagrams you should know that they are relevant and occasionally useful. Figure 6.6 uses three Venn diagrams to show how the measures are calculated and how they are different from each other. Note one key aspect of the diagrams is the explicit inclusion of the ~A ∩ ~B component—the itemsets that are in the universe of transactions but not in either A or B sets.

Support is the percentage of transactions that include both A and B (the intersection of the circles), divided by the total number of all transactions. The comparison with confidence is clear because although the numerator is the same, the denominator for confidence is all of circle A and nothing else. The computation for lift is not shown because it is a different type of ratio (not a percentage). It is the confidence value divided by the B over everything ratio.

Likewise, only the denominator for relative risk is shown in the diagram. The numerator is the part of B that is not included in A. The denominator is everything

| Support | Cake | Pie | Sweet Rolls | Cookie | Muffin | Pastry |
|---|---|---|---|---|---|---|
| Cake | 0.1 | 0.4 | 0.1 | 0.0 | 0.1 | 0.2 |
| Pie | 0.4 | 0.1 | 0.2 | 0.1 | 0.1 | 0.1 |
| Sweet Rolls | 0.1 | 0.2 | 0.2 | 0.3 | 0.1 | 0.1 |
| Cookie | 0.0 | 0.1 | 0.3 | 0.2 | 0.2 | 0.2 |
| Muffin | 0.1 | 0.1 | 0.1 | 0.2 | 0.0 | 0.1 |
| Pastry | 0.2 | 0.1 | 0.1 | 0.2 | 0.1 | 0.0 |

**Figure 6.8**

Support values for pairs of categories. The values are derived from simple counts of the transactions containing both items. The highlighted values are greater than 0.2 and would be the initial cut off for search algorithms.

not in A, so it includes the numerator part plus all of the transactions outside both A and B. It is clear from the diagram that if nothing exists outside of A and B, then the denominator for relative risk reduces to one (the green part of B divided by the green part of B).

### *Examples*

With a small number of dimensions, all of the measures presented in this section are relatively easy to calculate. Actually, computability is an important aspect of the measures. The search algorithms spend a huge amount of time retrieving, sorting, and comparing combinations of dimensions. If the value computations are extensive as well, performance will fall dramatically. Figure 6.7 shows a tiny sample of 10 transactions with 6 categories. Each row contains data for a single transaction or basket. The items purchased in each transaction are listed in columns. As an example of support for a single item, the Muffin entries are highlighted to yield a support of 0.3 for muffins (3 of 10 rows contain muffins). The support values for the individual categories are: Cake=0.6, Pie=0.5, Sweet Rolls=0.4, Cookie=0.4, Muffin=0.3, and Pastry=0.4. The simplest rules consist of pairs, so the transactions that contain both Pie and Cake are highlighted, showing a support of $P(\text{Pie} \cap \text{Cake})$ of 0.4 or 40 percent.

Figure 6.8 shows the support values for all of the pairs of categories. It is created from simple counts of the transactions that include each pair. Even that simple table requires some effort to create by hand. The highlighted values are greater than 0.2 and would be the ones that would be selected for further analysis. Most algorithms have an option to set the cut off level. Only items above this threshold are analyzed further.

Figure 6.9 shows the computed confidence values. The easiest computation method is to take the support values for each pair and divide by the individual support value for the row item. For example, $P(\text{Cake} \cap \text{Pie})/P(\text{Cake}) = 0.4/0.6 = 0.67$ for the row=Cake, column=Pie entry. The values greater than 0.4 are highlighted. Again, the analyst sets this cutoff value to keep the number of displayed rules to a manageable level so that the most valuable ones are easier to find. However, most systems apply the support rule first and only compute confidence (or perhaps lift) for the items that pass the first test. Hence, only the four rules highlighted in Figure 6.8 would be evaluated and displayed.

| Confidence | Cake | Pie | Sweet Rolls | Cookie | Muffin | Pastry |
|---|---|---|---|---|---|---|
| Cake | 0.17 | 0.67 | 0.17 | 0.00 | 0.17 | 0.33 |
| Pie | 0.80 | 0.20 | 0.40 | 0.20 | 0.20 | 0.20 |
| Sweet Rolls | 0.25 | 0.50 | 0.50 | 0.75 | 0.25 | 0.25 |
| Cookie | 0.00 | 0.25 | 0.75 | 0.50 | 0.50 | 0.50 |
| Muffin | 0.33 | 0.33 | 0.33 | 0.67 | 0.00 | 0.33 |
| Pastry | 0.50 | 0.25 | 0.25 | 0.50 | 0.25 | 0.00 |

**Figure 6.9**

Confidence values for pairs where row -> column. Values greater than 0.4 are highlighted. But only the items that first passed the support criteria would be evaluated.

Figure 6.10 shows the computed lift values for all of the pairs. The lift can be computed as the confidence divided by the support for each column. For example, Confidence(Pie|Cake)/P(Pie) = 0.67/0.6 = 1.33, which is the lift for the Cake->Pie rule. The four highest lift values are highlighted. Notice that only the Cookie and Sweet Rolls rules are also highlighted in the support table. This difference is important because many tools use the support rule to cut off further analysis. Hence, the Cookie->Muffin and Muffin->Cookie rules with their relatively high lift will probably not be displayed unless the support cutoff is dropped to a low enough level.

The point of the examples is to illustrate how the standard numbers of support, confidence, and lift are computed. For pairs of items, the numbers are relatively easy to compute. Keep in mind that itemsets can consist of any combination of categories. The example also highlights the difficulties in finding the important relationships within the data. The values enable automated search systems to work, but they still require interpretation and some guidance by the analysts and managers.

### *Summary of Values*

It is important to understand why different measures exist. The goal is to find a measure that can find the interesting itemsets in the data. The challenge is that interestingness is hard to define. It might be highly subjective, and several researchers are trying to find useful ways to automatically incorporate subjective measures from managers. Interestingness might also require surprise value—finding patterns that indicate a rule is unexpected or different from typical rules. Researchers are working in all of these areas and more. But, for now, the tools generally rely on the primary measures: (1) support, (2) confidence, and (3) lift. Variations exist among these three measures, but the conceptual goals are similar. Just be careful when evaluating results from different tools—be sure to verify the exact definition of the measures used. These concepts will become clear in the examples later in this chapter.

| Lift | Cake | Pie | Sweet Rolls | Cookie | Muffin | Pastry |
|---|---|---|---|---|---|---|
| Cake | 0.28 | 1.33 | 0.42 | 0.00 | 0.56 | 0.83 |
| Pie | 1.33 | 0.40 | 1.00 | 0.50 | 0.67 | 0.50 |
| Sweet Rolls | 0.42 | 1.00 | 1.25 | 1.88 | 0.83 | 0.63 |
| Cookie | 0.00 | 0.50 | 1.88 | 1.25 | 1.67 | 1.25 |
| Muffin | 0.56 | 0.67 | 0.83 | 1.67 | 0.00 | 0.83 |
| Pastry | 0.83 | 0.50 | 0.63 | 1.25 | 0.83 | 0.00 |

**Figure 6.10**

Lift values for rules Row -> Column. The four highest values are highlighted. Only the Cookie and Sweet Rolls rules are also highlighted in the support table.

## Problems with Dimensions

**Why do association rule routines have parameters and what values should be selected?** Association or market basket analysis is one of the data mining tools that is highly sensitive to the number of dimensions. Remember that rules consist of two itemsets A -> B, where the set of items on the antecedent or left (A) implies the set of items on the **consequent** or right side (B). With d total dimensions (or items), these itemsets can consist of any combination of 1, 2, 3, … d-1 attributes. In a full comparison, each possible itemset on the left should be compared with every possible itemset on the right. As Tan (2005) notes, some algebra reveals that this full comparison using brute force amounts to $3^d - 2^{d+1} + 1$ comparisons. To reduce the number of computations, most systems assume problems are somewhat simpler; so most tools assume that the predicted side (B) contains only one element. Still, the left side (A) consists of every combination of 1, 2, 3, … d items; and the sum of all of these combinations is $2^d$. The point is that the number of rules to be searched is exponential in the number of dimensions. For example, the six categories in the example problem would require $2^6$ or 64 comparisons, which is relatively small. A problem with only 20 dimensions quickly jumps to $2^{20}$ or one million comparisons. How many problems have more than 20 dimensions? Even if the problem is reduced by using departments and categories, there could be hundreds of dimensions. Even if d is 100, $2^{100}$ is a huge number ($1.27 \times 10^{30}$). Forget about trying to search all of those combinations. The problem is sometimes known as the **curse of dimensionality**. Now think about trying to compare every combination of products in a huge discount store. With existing computing systems, these problems are not feasible if the computer has to evaluate every possible combination of dimensions.

Two solutions exist to this problem. The first is efficient search algorithms as explained in the following paragraphs. The second solution is to reduce the number of dimensions. No matter how efficient the algorithm or how fast the computer, at some point it becomes necessary to reduce the number of dimensions being compared. Typically, that means switching from product-level comparisons to category comparisons. Similarly, analysts often restrict the number of products to those that fall within a specific set—such as comparing detailed items only within a department. In cases where detailed product comparisons are needed (recommendations for books and movies), the algorithm needs to be restricted to examining the data just as single pairs, which requires in $d^2$ comparisons.

## The A Priori Algorithm

It is helpful to understand a few concepts of the most common association algorithm. The most critical aspect is that parameter options are used to control the level of the search. The tools have default values for these parameters that try to strike a balance between finding useful rules and running within a reasonable time. These parameters are related to the issue of too many dimensions, but the parameters are based on the value computations that attempt to measure interestingness. In practice, if the model runs within short time frames, it is possible to experiment and try different parameter values. However, for large problems, it is important to begin with conservative values to first see how many rules are returned and to determine how long the search takes.

Agrawal (2005) first developed the **a priori algorithm** which opened the door to large-scale association analysis and remains one of the most popular algorithms in use. Yet, it is not perfect, so most tools modify it to improve performance. Also, many variations exist largely because of the number of possible measures that can be used to evaluate interestingness. The basic original model is briefly described here. From that point, it is easy to see the variations.

The basic process of the a priori algorithm is to start with a seed set of candidate itemsets that occur frequently in the dataset on the assumption these might lead to strong rules. The computer then goes through all of the transactions and computes the value (typically support) for each rule and keeps the ones that are high enough. A new set of seed itemsets is generated from the ones that pass the threshold test and the process repeats until the possible itemsets have been searched. The critical feature of the algorithm is that new candidate itemsets are based only on the prior itemsets that are sufficiently strong.

The algorithm begins its search by looking only at single-item sets. It keeps the set only if its support exceeds some threshold value. Consequently, this cutoff value is the most critical parameter of the method. Any potential rules that occur less often than the specified value are discarded from further analysis. Remember that support is the frequency or percent of times that the itemset appears in the transactions. So, if the minimum support level is 0.4, and some category or dimension does not occur in at least 40 percent of the transactions it is tossed.

After the single-item sets have been evaluated, the algorithm moves to the combination of two-level sets. But, if category A (say apples) appears less than 40 percent of the time, then it is clear that searching for A with any other set will also be less than 40 percent. Subsets are never greater than the original set, so adding B (bananas) to the condition will reduce the number of potential transactions. In the example, if apples appear in less than 40 percent of the sales, then the combination “apples and bananas” cannot appear in 40 percent either. Consequently, any single-level itemset thrown out at the first step will not be used at other levels. This decision substantially prunes the list of itemsets to be considered. The process continues, so that only two-level itemsets that are significant are examined for three-level items and so on.

The algorithm still needs to iterate through all of the data at each step to compute the support values for the candidate itemsets. If the transaction dataset consists of millions or billions of observations, this step can be time consuming. The a priori algorithm still requires several passes through the data. This is one area where other algorithms have found substantial improvements in performance.

Once all of the itemsets have been found that have support values above the minimum level, the algorithm loops through the data set one more time to compute the confidence value for each of the remaining rules. These values are often sorted and the results displayed in terms of rules in descending order of confidence. Most algorithms also specify a cutoff value for the confidence level.

## Issues in Setting Minimum Support and Confidence

All tools that function similar to the a priori algorithm use two critical parameters: (1) **minimum support**, and (2) **minimum confidence**. Some systems use different terms, sometimes different definitions, and occasionally different values; but the overall concepts and tradeoffs are the same. At the start, the analyst must specify the cutoff values. Any potential rule falling below the threshold will be discarded. Because of the way the algorithm works to reduce the number of itemsets, all subsets of that itemset will also be discarded.

Choosing a low value for minimum support will keep many more itemsets and potential rules in the analysis. But, particularly with huge transaction sets, a low support threshold exponentially increases the number of itemsets to be considered and it might be impossible to examine all of the requested possibilities in a reasonable time.

Selecting a low value for the second parameter (usually confidence, but sometimes lift) is less drastic. It is used to limit the number of rules displayed. If a low value is selected, the system might attempt to display hundreds or thousands of rules. But most systems sort the list in descending order. Many tools also export the rules to a database and provide options to search and sort the rules on various criteria. Overall, the effect on performance is small and it is just a question of analyst time to deal with the extra potential rules.

So what values of support should be selected? Will some values work in all cases? The answer to these questions is that each problem is unique. Values that work in some cases will be unhelpful in others. A better way to approach the issue is to determine a process for finding good values of support and confidence with each problem. The first step in the process is to recognize that for small problems, the impact of the choices is relatively minor, but large problems can result in long computation times unless parameters are chosen carefully.

Small problems can be rerun quickly, so analysts can experiment with different values until they are happy with the resulting rules. But how do you know if a problem is small? The number of dimensions is one important measure, but there is no specific rule defining small and large. Plus, computational time depends on the hardware as well. So, in most cases, it is best to begin by assuming any problem could be big.

Large problems typically have many dimensions and huge transaction data sets. Always remember that the support cutoff value is a critical element used to reduce the number of itemsets to be searched. Setting a value that is too low can result in huge performance penalties. Consequently, the process for large problems is to start with a relatively high value for the minimum support and then slowly reduce it to add more rules to the results. The confidence parameter can start relatively low since it primarily controls the output display. Some algorithms begin with minimum support values of 40 or even 60 percent. With a large number of dimensions and evenly distributed sales, it is possible that no rules will exceed this initial threshold. But, the analysis will run quickly. At that point, the minimum support can be reduced and the analysis rerun. But, be careful to leave the confidence minimum relatively low. If confidence is set too high, the support parameter might be at a reasonable level, but the confidence minimum is blocking the display. The first goal is to find a reasonable value for minimum support, the confidence minimum can be found as a second step.

There is no solid rule for decrementing the support cutoff, but larger problems require more cautious changes. Values of ten percentage points might be reasonable at the high end. However, support is rarely evenly distributed. Typically a few dimensions will have high support values, then when support drops below a certain point, almost all of the dimensions will come into play at the same time. For example, a dataset might contain a handful of dimensions with 40 percent support, a few more at 30 percent, and then almost half the dimensions at 20 percent support. Even small changes in the minimum support could suddenly result in an intractable problem. For large problems, it would be useful to compute and display the support values for each of the individual items before proceeding with the analysis. A fairly straightforward query can be used to count the number of baskets (rows) containing each item. With this information, the analyst can begin with a cutoff value that will include a few of the top-most items.

Once the support cutoff returns a reasonable number of rules, it is possible to increase the confidence threshold to reduce the display to those that are potentially more important and more interesting. Some systems provide interactive queries to make it easier to explore the rules that pass the support test. The key is to find rules that are interesting and provide insight into the underlying process. Because no single measure conveys all of this information, the process becomes subjective and requires analysts and managers to evaluate the rules individually. Typically the rules with higher importance values make the best starting points.

## Potential Problems

**Do association rule systems return all of the interesting rules?** The obvious answer is that it is not possible to completely automate the search process, so some rules are going to be missed. Random errors are always going to occur. The bigger issue is to identify specific types of problems that might arise. Two of the biggest problems arise from Simpson's paradox and skewed data. Continuous data is also an issue that needs to be considered for many problems. All of these issues commonly arise in practice and you need to be able to recognize the potential problems.

### Simpson's Paradox

Simpson's Paradox is named after E.H. Simpson (1951), but the concept was described earlier by Yule (1903) and it is not exactly a paradox, so it is sometimes named the Yule-Simpson effect. **Simpson's paradox** states that comparisons or associations for groups can be reversed when the groups are combined. A classic example consists of men and women applying for jobs, or for admission to programs at Berkeley as revealed in a lawsuit explained by Bickel (1975). Figure 6.11 shows a simple example with hypothetical data. In total, men were hired 73.9 percent of the time versus 65.8 percent for women. But, looking at the detailed departments, in every case, women were more likely to be hired than men. How could this result arise? The difference exists because the majority of the women applied for Department A, which had a much lower overall acceptance rate.

| | Men | | Women | |
|---|---|---|---|---|
| | Hired | Applied | Hired | Applied |
| Dept A | 1 | 3 | 18 | 30 |
| Dept B | 16 | 20 | 7 | 8 |
| Total | 17 | 23 | 25 | 38 |

| | Men | Women |
|---|---|---|
| Dept A | 33.3% | 60.0% |
| Dept B | 80.0% | 87.5% |
| Total | 73.9% | 65.8% |

**Figure 6.11**

Example of Simpson's paradox. In each detail department, women were accepted at a higher rate than men. But the rate for the aggregate total is reversed.

Think of the percentages as conditional probabilities: P(Hired | Man) and P(Hired | Woman). These probabilities or confidence values reverse from the detail to the aggregate grouping.

How can the analyst spot potential conflicts caused by Simpson's paradox? Clearly, if the data rules are tested at both the detail and aggregate levels, it is straightforward to compare the two and see if reversals exist. If they do arise, the key is to recognize that they do not represent an error in computations but arise from Simpson's paradox. Then a decision has to be made whether the detail or aggregate results more accurately answer the questions being asked. A bigger problem arises when data is only analyzed at the aggregate level. Conclusions based on aggregate results might lead to incorrect decisions if there is a hidden factor affecting the results that could lead to Simpson's paradox. Hence, it is important to test association rules at detail/subgroup levels if these groups exist. Any time information exists, it should be incorporated into the analysis. The results will either confirm the conclusions from the aggregate rules or Simpson's paradox will arise, leading the analyst to consider more complex interaction effects.

## Skewed Support Data

Skewed support is likely to be relatively common in business—particularly in sales. Skew is a statistical term that refers to the asymmetry of the distribution (biased towards one end). **Skewed support** appears when the bulk of the items have few sales and a handful of them are sold in every basket. Think about a huge discount store—it is not possible for all 50,000 plus items to sell equally. Among other reasons, expensive items are not likely to sell as quickly as cheaper items. Think about a corner gas station/convenience store. Gas is probably sold in almost every transaction. A few items within the store probably sell reasonably well: perhaps soft drinks, beer (if available), or milk. The rest of the items sell occasionally, but as a percentage of the total number of transactions the support is low. Figure 6.12 illustrates the skew. Compare the approximate distribution to a normal "bell-shaped" distribution.

The effects of skewed support are interesting. The first effect is straightforward. It is going to be difficult to choose a value for minimum support. In the example, any number between about 0.1 and 0.8 will return only the few items that appear

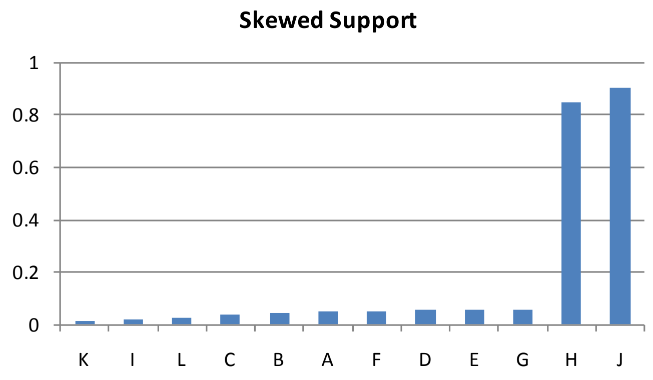


**Figure 6.12**

Skewed support. A few items are purchased by almost every customer. Other items are bought occasionally, but the total support is low. Think about what items might be sold at a local gas station convenience store. Gas and what else?

in most transactions. Remember that once an item is dropped for falling below the minimum support, it is not considered in any other itemsets. The analysis includes only the few items that appear regularly. But, there might be interesting and useful relationships among the other items. Despite the fact that the support level is low compared to the high-volume items, they might still be important to the business. The only solution to this problem is to drop the support level to almost zero to bring in the other values. Of course, that will bring in almost all of the items—which could adversely affect performance if thousands of items exist.

A more challenging problem arises because of the mathematics. **Cross-support** consists of an itemset that contains one item from the high-volume group and one (or more) from the low-support group. Figure 6.13 shows the problem, by displaying values for a typical relationship and comparing them to values generated from cross-support. When support for A and B are roughly equal, the computation of P(B|A) represents confidence in a prediction and explains a relationship and perhaps causality between the two items. In cross-support, where B occurs in almost every transaction, this relationship no longer holds. Because B arises almost all the time, the few times that A shows up, it is likely to already have B in the basket. If 95 percent of the customers buy gas, then if anyone buys milk it appears that the two items were purchased together. But there is no correlation and certainly no causality. Simply by random chance, if almost everyone buys item B (gas), anything else that is purchased will appear alongside the gas item. This **spurious correlation** arose because of the extreme popularity of the B item. In fact, the results would actually be more interesting if the conditional probability were low. Since random chance indicates B should almost always appear with A, when they do not appear together, it would indicate some underlying effect keeping them apart is more important.

The cross-support effect of skewed distributions is difficult to correct. A few tools have options to check for and filter out these effects. In other cases, the analyst must simply be aware of the problem—keep an eye on support values for each item. When a rule appears with large disparities in support for the items, discount

| | A | B | A & B | P(B\|A) | P(A\|B) |
|---|---|---|---|---|---|
| **Typical** | 0.4 | 0.5 | 0.3 | 0.75 | 0.6 |
| **Cross-support** | 0.1 | 0.9 | 0.09 | 0.9 | 0.1 |

Figure 6.13

Cross-support with skewed data. When support for A and B are roughly the same, P(B|A) represents confidence in the prediction or potential causality. If P(B) is high, it will appear in almost any transaction. Consequently, by random chance, A and B can show up together almost every time A appears simply because B will always be there. The value for P(B|A) is spurious correlation and cannot be interpreted as causation.

the value of the rule. Also, take a look at the lift. Because lift is confidence/P(B), the value of lift will be close to 1.0 if A and B occur randomly in the cross-support example. In most cases of cross support, lift cannot get much higher than 1.0.

## Continuous Data

Association rule and market basket analysis can be applied only to discrete data. It functions by counting the number of transactions that contain each itemset. With continuous data (such as income), each value would represent a separate item. This approach could potentially create millions of dimensions, making the problem virtually unsolvable. Additionally, would a rule using income=50,101 really be different from a rule using income=50,103? Even if the tool found differences, the business interpretation would be meaningless.

If it truly is important to use continuous data in association analysis, the data series must be converted to discrete groups. For instance, income could be split into Low (Income < 30,000), medium (30,000-80,000), and high (Income>80,000). Each group would be treated as a separate dimension, which raises the critical question of the number of categories to create. Create too many categories and the dimensionality gets out of control. Plus, the results would be difficult to interpret. Create too few, or create them at the wrong split points, and the analysis will miss useful effects and rules.

If some existing business definition exists, it can be used to define split points. For example, the size of a firm is often defined by the number of employees, and definitions exist for small businesses (less than 500 employees), with the high-end defined by Fortune 500 level companies with tens of thousands of employees. Similar examples might be used to define splits for income levels, education, city population, and so on. In other cases, many choices exist for discretization. One approach is to determine the number of categories and find the split points that assign the observations equally to each group. Figure 6.14 shows an example of splitting observations on income into equal-sized groups. Based on the hypothetical data, each group contains about 3,000 observations. The split points are probably reasonable, but think about potential loss of data from the grouping. With income data, it might be important to include a very-high income group. Although few people would fall into the group, it could be important to the company and might generate useful rules for a group that could substantially improve profitability.

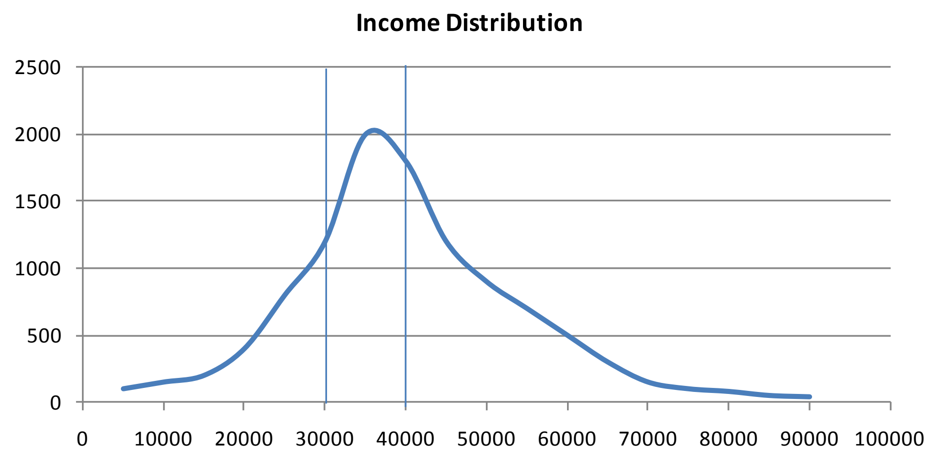


**Figure 6.14**

Equal-size groups for continuous data. Based on the total number of observations, this split puts about 3,000 observations in each group. Be careful that grouping does not lose important data. Perhaps with income, it would be important to include a very-high group.

Clustering is another common approach to defining groups. If the data fall into natural groupings that can be identified by clustering algorithms, these clusters will be more likely to yield useful results. Once the clusters are identified and interpreted, the results will also be easier to understand and explain to managers.

Yet another approach might be to use standard deviation. Any observations that are one or two standard deviations below (or above) the mean might be considered as extremes. This partitioning could be useful to compare the "average" group against the lower and higher extremes. Ultimately, the method of discretizing the data depends on the problem and the types of questions to be addressed by the analyst.

## Quantity

True market basket analysis examines sales of items or categories of items. For example, it compares cupcakes to bananas. In most cases, each item is treated equally in a binary fashion—the item is either present or absent from the basket. The analytical tools do not look at the quantity of items purchased. So, someone who buys 20 bananas is considered the same as a person who buys one. For grocery items, this distinction is probably reasonable. It is likely that most people fall within similar limits in terms of quantities. However, other types of businesses might care greatly whether a person buys one item or 20 at a time. Although the tools have no options to handle quantity, it could be added as a new dimension. But, remember that quantity is a continuous variable so it needs to be grouped into categories.

For each product that might have different interpretations due to quantity purchases, define new categories and use SQL or the data mining tool to assign the new values. For instance, separate categories could be defined for 1 jacket, 2-5

Sale

| **SaleID** | SaleDate | CustomerID | EmployeeID |
|---|---|---|---|
| 1101 | 7/8/2010 | 393 | 115 |
| 1102 | 7/9/2010 | 559 | 089 |
| 1103 | 7/9/2010 | 198 | 115 |

SaleItem

| **SaleID** | **ItemID** | Quantity | SalePrice |
|---|---|---|---|
| 1101 | 22902 | 1 | $3.95 |
| 1101 | 11983 | 5 | $10.00 |
| 1102 | 22902 | 2 | $3.75 |
| 1103 | 83932 | 3 | $2.00 |

Item

| **ItemID** | Description | Category | ListPrice |
|---|---|---|---|
| 11983 | Blackberry | Pie | $10.95 |
| 22902 | Raisin | Bread | 3.95 |
| 83932 | Blueberry | Muffin | 2.50 |

Figure 6.15

Typical database tables for Sale, SaleItem, and Item. Market basket analysis needs the transaction items from the SaleItem table, but also generally needs the category from the Item table.

jackets, and 6 or more jackets. Each category becomes a new dimension and the analysis runs normally. Again, the challenge lies in identifying the products for which quantity might be interesting, and then identifying useful split points for the categories. All of these operations require more passes through the data. But, once the categories are defined, the association analysis and interpretation is straightforward.

## Data

### How does data need to be organized for association analysis?

Unfortunately, two different methods are used to organize data for association rule analysis. One method is easy; the other can be difficult when starting with traditional database transactions. When data begins in a DBMS, the transactions are stored in a normalized format. This data lists each item purchased on a separate row. Some tools, particularly those within a DBMS such as the SQL Server association method, use the data in this form. However, some tools that started with the market basket approach require the data to be organized by basket or transaction—where one row represents a transaction and all items are listed on that row. Note that missing data is not supported by the association rules techniques. But, there should never be missing data—either a sale lists the item or category being sold or it does not. If the item is not listed, it must not have been important enough to keep.

Baskets with items

```
22902, 11983
22902
83932
19384, 44032, 39382
```

Baskets with categories

```
Bread, Pie
Bread
Muffin
Scone, Cake, Muffin
```

**Figure 6.16**

Market basket structure for data. Each row contains items in one basket. The items or categories need a separator—often a comma or tab character. Converting from a DBMS to this format is challenging.

## Database Structure

Databases are carefully designed to efficiently handle transaction data. A specific set of rules defines the best way to organize data. Following these rules, the common business data for a Sale is broken into two tables: Sale and SaleItem. The Item data is stored in a third table, and all of them are connected through the primary keys. The SaleItem table has two columns in the primary key: SaleID and ItemID. Figure 6.15 shows sample data with a couple of rows of data. The SaleItem table is the key table for market basket analysis. Each SaleID is the transaction or basket. Each item in the basket is listed in a separate row. However, to perform the analysis by category instead of by each item, a query is built that links the Item table to the SaleItem table, making Category available to the analysis.

This format is the standard method of collecting transaction data in any DBMS or enterprise system. Data mining tools that are integrated with a DBMS generally use this format directly, or have tools to automatically convert data from this format. Sometimes queries are used to limit the data to specific stores, departments, regions and so on.

## Market Basket Structure

Early market basket tools were not designed to pull data from a DBMS. Many of them assumed data would be stored in simple text files—with all items in one basket listed on a single row in the text file. Figure 6.16 shows small examples for listings by item and category. Because one row contains multiple items, the items must be separated by a standard character—usually a comma or tab. This character cannot appear within the text of the items or categories.

Converting from the DBMS transaction format into this market basket format is a challenge. Standard database queries are not designed to repeat items across a single row of output. Custom programming code is needed to read items one row at a time and display them onto a single row of output, adding the separate character along the way. Figure 6.17 provides sample code that runs within SQL Server to create the basket listing for the Bakery case. Because the code uses an nvarchar variable to construct each line, the line is limited to a maximum of 4.000 Unicode characters. With a limited number of items sold on each sale and relatively short category names this limit is not a problem for the bakery case. Perhaps a bigger problem is the number of output rows. With almost two million rows, it is going to be a challenge to save the output. It would be better to use the SQL Server pro-

```
CREATE PROCEDURE BuildBasket
AS
BEGIN
    DECLARE db_cursor CURSOR FOR
    SELECT SaleID, Category
    FROM SaleItem
    INNER JOIN Product ON SaleItem.ProductID = Product.ProductID
    INNER JOIN ProductCategory ON Product.CategoryID=ProductCategory.CategoryID
    ORDER BY SaleID, Seq;
    DECLARE @SaleID integer, @Category nvarchar(250), @PriorSaleID integer;
    DECLARE @iCount integer, @strLine nvarchar(4000);
    OPEN db_cursor;
    FETCH NEXT FROM db_cursor INTO @SaleID, @Category;
    SELECT @iCount = 0, @strLine = '';
    WHILE @@FETCH_STATUS = 0
    BEGIN
        IF (@iCount > 0) SELECT @strLine = @strLine + ',';
        SELECT @strLine=@strLine + @Category;

        SELECT @PriorSaleID = @SaleID;
        select @iCount = @iCount + 1;
        FETCH NEXT FROM db_cursor INTO @SaleID, @Category;
        IF (@@FETCH_STATUS = 0)
        BEGIN
            IF (@PriorSaleID <> @SaleID)
            BEGIN
                PRINT @strLine
                SELECT @iCount = 0, @strLine ='';
            END
        END
    END
    CLOSE db_cursor;
    DEALLOCATE db_cursor;
END
```

**Figure 6.17**

Sample code to create market basket layout for the Bakery. The output line is limited to 4,000 Unicode characters which is sufficient for the Bakery case with a limited number of items sold at each sale.

cess to create a text file and write the lines to that text file. However, the code to handle files is too long to list here. It can be found on the Web. In most cases, big problems would be easier to handle with an external, compiled language such as C# or Visual Basic. The logic is similar, but file handling is simpler and it would be easier to create forms to let users pick the tables and columns interactively.

## Traditional Tools for Association Rules

**How difficult is it to create and interpret association rules?** The best way to understand market basket analysis and association rules is to work with several examples. Experiment with different settings of the support and confidence parameters and see how the results change. Although every problem is different, it takes some practice to understand how to use the parameters to control the results. The sensitivity of each data set is different, but the concepts are the same. One catch is that many variations of the tools exist and each is somewhat different. One of the earliest tools is available on the Internet. Christian Borgelt provides several tools on his Web site (http://www.borgelt.net/software.html). His apriori tool searches for association rules and was the foundation for the SPSS Clementine routine. He even provides the source code, so programmers can modify or improve the routines.

## Goals

The a priori tool is a relatively straightforward implementation of the a priori algorithm. It takes input from a simple text file and writes rules to an output text file. It uses a measure of support as the initial filtering mechanism. The standard confidence measure is used to determine which rules to display. The one catch with this version of the tool is that it uses a version of support that is different from the traditional algorithm. In the common approach, support is computed as the percentage of transactions containing all of items in the rule: the antecedent and the consequent. The default in Borgelt's system is to compute the support only for the antecedent or left side of the rule. The routine does have a flag (-o) to tell it to use the original definition, but it does not work in at least some versions. It is not a critical problem, but it requires caution in setting the minimum support level and when interpreting the results.

## Data

Many of the early association algorithms require data to be stored as baskets—where each row in the text file contains a list of items in that market basket. Figure 6.16 provides an example. Each item must be separated by some character. Spaces and tabs are the default separators in the apriori code. These values can be changed with the –b flag. Use caution if spaces are used as separators—particularly when using text entries such as categories. If the text contains a space character, it needs to be converted to something else, typically an underscore ( _ ). In the bakery case, the ProductCategory table should be edited and Sweet Rolls changed to Sweet_Rolls.

As described in the Model section, when transaction data is stored in a DBMS, it is relatively difficult to convert it to the one line equals one basket data file. Usually, the easiest method is to write code that can read all of items for one sale and print the results to a file as a single row. The file can be large. The text file containing all of the Bakery data exceeds 50 megabytes.

## Results

Remember that it is best to start with low values for the minimum confidence, then decrease the support level until enough rules appear to provide results. In the end, a support level of 10 percent and confidence of 20 percent yields 115 rules. The list undoubtedly contains some rules that are weak, but it is short enough for an analyst to read through. The list can be imported into an Excel file and sorted. Alternatively, it could be imported into a new database table and SQL could be used to search the results.

Figure 6.18 shows the top results from the aprioir tool sorted by confidence. Because the support definition includes only the antecedent, it provides minimal information. Also, note the inclusion of the empty set (null) as an antecedent. Because the empty set exists in every case, its support is always 100. Although its inclusion might appear strange at first, it is an interesting way to determine which item categories are most popular on their own. In the example, Muffin has the highest confidence (40.9), followed by Pie (36.2), Bread (33.4), and Cupcake (32.8). These are the items that initially draw customers to the bakery.

The top paired rules also provide useful business information. Check the top three rules that show customers who buy Bread tend to buy Muffins 47.6 percent of the time, Cake implies Pie 42.9 percent of the time, and buying Rolls leads to Muffins 41.1 percent of the time. Managers could use these rules to arrange the store to entice additional sales, or perhaps change the pricing and profit margins. For instance, Bread was a relatively high standalone item and Bread leads to Muffin purchases. Perhaps a discount on Bread prices would attract more customers who would then purchase Muffins with a higher profit margin. In the end, it is up to managers to understand the rules and determine how to apply them to increase sales and profits.

Working down the list of rules, you eventually encounter a few rules with multiple items in the antecedent. The portion of the results shown in the table display an interesting relationship among Cupcakes, Muffins, and Pies. A third rule that is not shown also states that a basket containing Pie, and Muffins leads to the purchase of Cupcakes 28.3 percent of the time. These three rules imply that many customers come in to purchase these three items together. If business slows down for any reason, these three would be a good target for an advertised package.

These rules are only the beginning of the analysis, but they illustrate the basic process. Many other analyses should be run, such as comparing sales by day of week or perhaps certain seasons. These are left as exercises for the reader.

## Microsoft Association Rules

**How does the Microsoft Association Rules tool differ from traditional tools?** One of the main differences with Microsoft's Association tool is that it uses confidence instead of support as the primary filter. In terms of processing and eventual results, the difference is not huge, but it means the analyst has to think more carefully about how to set the minimum value. Microsoft also uses the name probability instead of confidence, which is reasonable since confidence is defined as P(B|A). Microsoft then defines Importance as the measure to evaluate the returned rules. Microsoft's definition of Importance is hard to find but it is similar to lift with a few twists. The relative risk measures forms the foundation of Importance: P(B|A) / P(B|~A). That is, the ratio of the probability of B given that A exists to the probability of B given that A does not exist. The formula translates to the probability of selecting B when A is present versus selecting B when A is not present. For mostly technical reasons, Microsoft adds a couple of tweaks. First, to avoid some specific situations with zero entries, the tool adds 1 to the count of each category. This tweak is usually invisible but required if trying to hand calculate the value from a small dataset. The other change is to take the log of the ratio—to convert the ratio into a number that can be plotted:

$$\text{Importance} = \log_{10}(P(B|A) \;/\; P(B|\sim A))$$

Values greater than zero correspond to ratios greater than one—which are positive effects on lift. Log values less than zero would indicate lift ratios less than one, or negative impacts on sales.

| | | | | |
|---|---|---|---|---|
| Muffin | <- | Bread | 33.4 | 47.6 |
| Pie | <- | Cake | 17.5 | 42.9 |
| Muffin | <- | Rolls | 21.6 | 41.1 |
| Muffin | <- | | 100.0 | 40.9 |
| Bread | <- | Rolls | 21.6 | 40.2 |
| Bread | <- | Muffin | 40.9 | 38.9 |
| Muffin | <- | Pie | 36.2 | 38.3 |
| Muffin | <- | Cookie | 28.3 | 38.0 |
| Pie | <- | Cupcake | 32.8 | 37.1 |
| Muffin | <- | Cupcake | 32.8 | 37.0 |
| Pie | <- | | 100 | 36.2 |
| Muffin | <- | Pastry | 25.6 | 35.4 |
| Scone | <- | Sweet_Rolls | 26.3 | 34.9 |
| Pastry | <- | Crepes | 25.4 | 34.9 |
| Crepes | <- | Pastry | 25.6 | 34.7 |
| Muffin | <- | Candy | 21.3 | 34.7 |
| Bread | <- | Candy | 21.3 | 34.5 |
| Pie | <- | Muffin | 40.9 | 33.9 |
| Cupcake | <- | Pie | 36.2 | 33.6 |
| Bread | <- | | 100.0 | 33.4 |
| Cupcake | <- | | 100.0 | 32.8 |
| Cookie | <- | Pastry | 25.6 | 32.8 |
| Pie | <- | Cupcake | 12.2 | 32.3 |
| Muffin | <- | Cupcake | 12.2 | 32.2 |

**Figure 6.18**

Top results from apriori routine sorted by confidence. The Support definition uses only the antecedent.

## Goals

The basic objective of the Association tool is to find the rules or items that appear together most frequently. The tool uses the a priori algorithm, but does the initial cutoff based on minimum confidence instead of minimum support. The resulting rules are displayed with the confidence (Probability) estimate and the Importance estimate. By using a log transformation, the Importance values are charted to make it easier to see the rules with the highest lift.

## Data

Because the data mining tools are integrated with the SQL Server database, the association tool can read market baskets directly from the database transaction tables. It is important to specify the key columns. Typical columns consist of items such as SaleID for the transaction identifier and ProductID or Category depending on whether the analysis uses individual items or categories.

```
SELECT        dbo.SaleItem.SaleID, dbo.SaleItem.ProductID,
              dbo.SaleItem.Quantity, dbo.SaleItem.SalePrice,
              dbo.Product.CategoryID, dbo.ProductCategory.Category,
              dbo.Product.ProductName
FROM          dbo.SaleItem
INNER JOIN    dbo.Product
    ON dbo.SaleItem.ProductID = dbo.Product.ProductID
INNER JOIN    dbo.ProductCategory
    ON dbo.Product.CategoryID = dbo.ProductCategory.CategoryID
```

**Figure 6.19**

Named query to combine SaleItem, Product, and ProductCategory tables. Name it SaleItemProductCategory.

The Bakery case was designed to illustrate the uses of the market basket tool. Create a new data source that connects to the Bakery database. Because the database has few tables, create a new dataset view that contains all four tables: Sale, SaleItem, Product, and ProductCategory.

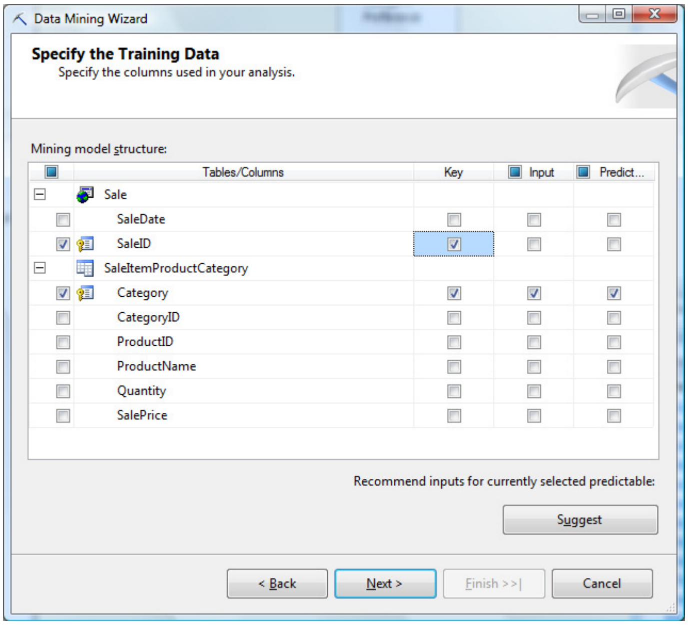
The analysis is easier to configure if all of the main columns are in one named query instead of scattered across the SaleItem, Product, and ProductCategory tables. Create a named query (SaleItemProductCategory) that links SaleItem with the Product and ProductCategory tables. It simply needs to display the SaleID, ProductID, and Category columns, but the others can be included as well. Figure 6.19 shows the simple query. In some cases, the combined data might already exist in a data cube.

Once the named query is created, assign logical primary keys to it in the data source view: SaleID + ProductID. Then create a relationship from the query to the Sale table based on the SaleID. Remember if the analysis is performed on a different server than your development computer (localhost), use the Deployment properties of the project to set the name of the server.

Microsoft's Association Rules model has an additional feature that is useful in some problems. So far, all of the examples in this chapter have pulled dimension attributes from a single column, such as Category or ProductID. Microsoft's configuration wizard makes it easy to use multiple columns. For instance, the Dining case has one table that includes day of week, gender of the customers, and the meal time (breakfast, lunch, dinner). All of these attributes can be combined into itemsets. With the Microsoft wizard, simply specify each attribute as both an input and predictable column. For traditional tools, it would be necessary to create a query that defines a new column to combine the attributes. For example, Meal = DOW + '_' + MealTime + '_' + Gender. On the flip side, the transaction key needs to consist of a single column and the tool is easiest to configure if all of the data reside in a single table or query.

## Results

The tool is straightforward to run. Create a new mining structure using the Microsoft Association Rules. To ensure the SaleID key is unique, select the Sale table as the Case table and the new named query (SaleItemProductCategory) as a Nested table. Figure 6.20 shows the selection of the columns for the analysis. The SaleID must be set as the Key value because it defines the transaction or market basket.



Data Mining Wizard

**Specify the Training Data**

Specify the columns used in your analysis.

Mining model structure:

| Tables/Columns | Key | Input | Predict... |
|---|---|---|---|
| Sale | | | |
| ☐ SaleDate | ☐ | ☐ | ☐ |
| ☑ SaleID | ☑ | ☐ | ☐ |
| SaleItemProductCategory | | | |
| ☑ Category | ☑ | ☑ | ☑ |
| ☐ CategoryID | ☐ | ☐ | ☐ |
| ☐ ProductID | ☐ | ☐ | ☐ |
| ☐ ProductName | ☐ | ☐ | ☐ |
| ☐ Quantity | ☐ | ☐ | ☐ |
| ☐ SalePrice | ☐ | ☐ | ☐ |

Recommend inputs for currently selected predictable:

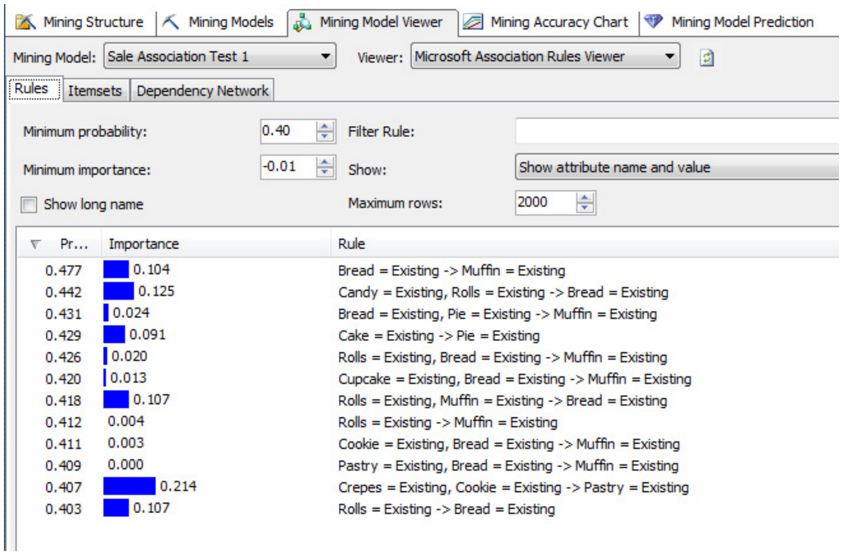
**Figure 6.20**

Selecting columns for association rules mining. Set the SaledID as the key to define the transaction or market basket. Select the Category column as the prediction data and the input. Check drill through on next screen.

Every item purchased with the same SaleID is purchased at the same time. The Category is specified as the predictable and input column. The Category column contains names so it is more useful than the CategoryID column which consists of simple numbers. This choice performs the analysis with categories as the dimensions. Selecting ProductID instead of Category would perform the analysis on all of the individual items. Avoid starting with ProductID because it is a much bigger problem and could take a long time to run. Finish the wizard to complete the model definition. Right-click the model in the Solution explorer and choose the option to Process it and run the processor. When processing is complete, right-click the model and choose the option to Browse the results.

Figure 6.21 shows the initial results from the Association Rules model. Clicking on the heading causes the rules to be resorted. Selecting minimum probability and importance in the filter boxes reduces the number of rules displayed. These tools provide a filter for searching for rules—particularly useful when hundreds or thousands of rules appear. The charts for importance make it easier to see which rules have the greatest lift. Note that negative values will appear in red—they represent lift values below 1—or negative correlations.

Notice that the cutoff value for probability can only be increased on this form. Changing it here only changes the display results—it does not rerun the analysis. Decreasing the numbers requires changing the parameters and re-computing the results. Figure 6.22 shows the basic process for changing the underlying parameters. On the Mining Models tab, right-click the Microsoft_Association_Rules model and choose the option to Set Algorithm Parameters. Remember to be cautious when making changes. The minimum probability (confidence) is used to



Mining Model: Sale Association Test 1
Viewer: Microsoft Association Rules Viewer

Minimum probability: 0.40
Filter Rule:
Minimum importance: -0.01
Show: Show attribute name and value
Show long name
Maximum rows: 2000

| Pr... | Importance | Rule |
|---|---|---|
| 0.477 | 0.104 | Bread = Existing -> Muffin = Existing |
| 0.442 | 0.125 | Candy = Existing, Rolls = Existing -> Bread = Existing |
| 0.431 | 0.024 | Bread = Existing, Pie = Existing -> Muffin = Existing |
| 0.429 | 0.091 | Cake = Existing -> Pie = Existing |
| 0.426 | 0.020 | Rolls = Existing, Bread = Existing -> Muffin = Existing |
| 0.420 | 0.013 | Cupcake = Existing, Bread = Existing -> Muffin = Existing |
| 0.418 | 0.107 | Rolls = Existing, Muffin = Existing -> Bread = Existing |
| 0.412 | 0.004 | Rolls = Existing -> Muffin = Existing |
| 0.411 | 0.003 | Cookie = Existing, Bread = Existing -> Muffin = Existing |
| 0.409 | 0.000 | Pastry = Existing, Bread = Existing -> Muffin = Existing |
| 0.407 | 0.214 | Crepes = Existing, Cookie = Existing -> Pastry = Existing |
| 0.403 | 0.107 | Rolls = Existing -> Bread = Existing |

**Figure 6.21**

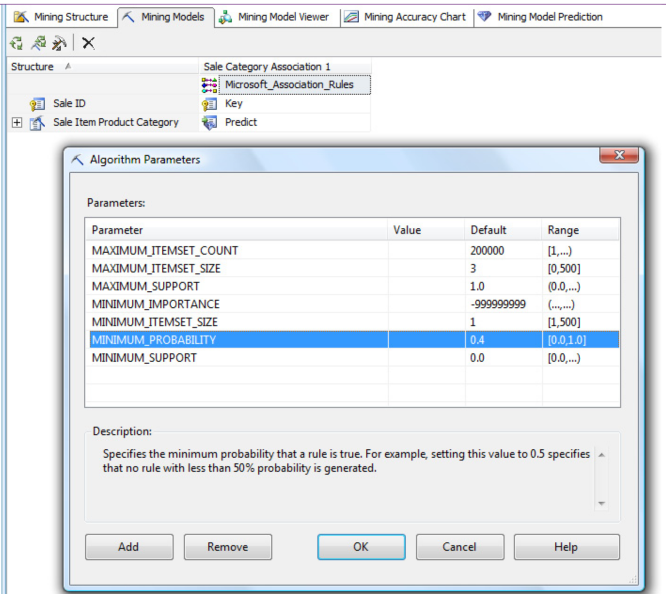
Initial results from Microsoft Association Rules. Note the minimum confidence can be increased to reduce the number of rules displayed. To decrease it, the model needs to be re-processed with new parameters.

limit the number of itemsets that are evaluated. Setting a low value causes the result set and processing time to increase exponentially. For the full collection of data in the Bakery problem, the default values appear to produce a reasonable number of results. Judging from the confidence values in the traditional analysis, some rules might be missed, so it could be useful to rerun the analysis at 30 percent.

As shown in Figure 6.23, the Microsoft tool also generates a dependency diagram. The arrows show the dependency of the nodes. For instance, purchase of a Cake leads to a purchase of Pie. The slider on the left is used to cut off weaker links. Clicking a single node causes the others to be color coded to show the direction of dependency. For instance, select the Muffin node to see that purchases of both Rolls and Bread have an effect on the sale of Muffins.

## Comparing Results

At first glance, the results from the traditional and Microsoft tools might appear to be different. To see similarities, sort the results from both tools by confidence (probability). Both tools list Bread -> Muffin at the highest confidence of 47.6 percent, Cake->Pie at about 43 percent, Rolls -> Muffin around 41 percent, and Rolls -> Bread around 40 percent. The probability values might differ slightly from the traditional results because the Microsoft tool holds out 30 percent of the observations to be used for testing. If this holdout set is decreased to zero, the numbers should match those from other tools. Also, if the Microsoft results are reprocessed with a lower minimum confidence, the results should continue to match those from the traditional tool. However, note that the Microsoft tool has found additional results with relatively high confidence that were not found by the



Parameters:

| Parameter | Value | Default | Range |
|---|---|---|---|
| MAXIMUM_ITEMSET_COUNT | | 200000 | [1,...) |
| MAXIMUM_ITEMSET_SIZE | | 3 | [0,500] |
| MAXIMUM_SUPPORT | | 1.0 | (0.0,...) |
| MINIMUM_IMPORTANCE | | -999999999 | (...,...) |
| MINIMUM_ITEMSET_SIZE | | 1 | [1,500] |
| MINIMUM_PROBABILITY | | 0.4 | [0.0,1.0] |
| MINIMUM_SUPPORT | | 0.0 | [0.0,...) |

Description:

Specifies the minimum probability that a rule is true. For example, setting this value to 0.5 specifies that no rule with less than 50% probability is generated.

**Figure 6.22**

Changing the minimum confidence parameter. On the Mining Model tab, right-click the Association Rules entry and choose the option to Set Algorithm Parameters. Try minimum probability at 30 percent.

traditional tool. In particular, Crepes, Cookie -> Pastry (0.407) and Candy, Rolls -> Bread (0.442) have the two highest Importance measures. Neither rule appears in the list of items from the traditional tool.

How can the Microsoft tool include (useful) results that are not in the results from the traditional tool? Because the traditional tool uses support as a filter and Microsoft uses confidence as a filter, the most likely cause is that the support for the items is too low to meet the cutoff value in the traditional tool. An SQL query could be used to count the number of cases, but Microsoft output provides an Itemset search tool. To check the rule for Crepes, Cookie, Pastry, set the minimum itemset size to 3 and filter on the word Crepes. The resulting list shows that the three items appear together in 23,728 market baskets, out of about 2 million total transactions. Because the traditional tool used in the example defines support only on the first two items, change the itemset size to 2 and find the entry for Crepes and Cookie. These two categories appear together in 58,250 baskets. Divide by 1,925,819 transactions to get a support percent of slightly over 3 percent. The cutoff used in the traditional tool was 10 percent—and that created 115 potential rules. Dropping to the 3 percent level would bring in the missing rule, but would also bring in hundreds of additional rules to sort through.

The point of the comparison example is to emphasize that any association tool can miss rules that might be useful. It is the reason there are so many different measures of interestingness and importance. It is a good idea to look at large datasets from several different perspectives. Bring in extra rules, compute multiple measures of interestingness and sort the results. Compare them and think about the potential interpretations. Remember that the purpose of data mining is exploration.

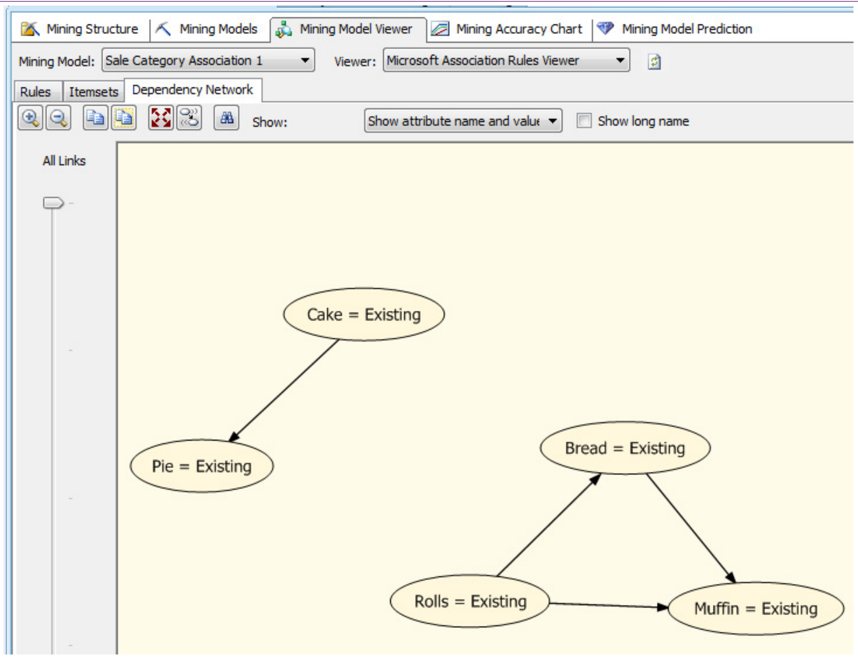


**Figure 6.23**

Dependency Network. The arrows show dependency. For example, purchasing cakes leads to purchases of pies. Select one of the nodes to see a color-coded reference of how the nodes are linked.

## Summary

Association rules or market basket analysis is one of the methods that started the data mining industry. The underlying concept is straightforward: Find the items or events that occur together. In market basket settings, the goal is to find items that are commonly purchased together. The tool can be used for many tasks in business, including store layout, pricing, and cross-selling. The tool is the foundation of many recommendation engines, such as those used by Amazon and Netflix.

In practice, the model is challenging to implement because of the number of comparisons required. If d is the number of items or dimensions, the potential number of comparisons assuming a single output item is 2 raised to the d-power. The a priori algorithm significantly simplifies the search and makes the method feasible even for relatively large datasets. Still, problems with a large number of items often need to be reduced by using item categories instead of detailed products. In many cases, these aggregate comparisons are more useful because the interpretation is more rational. For instance, if you are looking for a comparison between Cakes and Pies, is it really useful to search for differences between types of cakes and pies?

The a priori algorithm gains performance by cutting off potential rules early. This pruning is based on a parameter—typically the support or frequency of the initial items in the basket. Once an item or category is dropped from the analysis, all combinations of that item are also dropped. Choosing the threshold value for this parameter is a critical step in running market basket analysis. Setting the value too high blocks any potentially interesting rules from entering the analysis. Setting the value too low brings in too many itemsets to be solved in a reasonable time.

Categories are often a useful compromise when the number of products is too large. Yet, Simpson's paradox can result in unexpected shifts in the results. Aggregate results do not necessarily match the results from the detailed subgroups. Consequently, it is important to perform analyses at multiple levels and to evaluate the results of subgroups as well as the total. Different tools use different measures of interestingness, so results sometimes vary across tools. The problem is that no perfect measure of interestingness exists, so any tool can miss rules that might be useful. In the end, analysts often need to perform many different searches for rules, sorting results by various measures and evaluating rules for potential value. Market basket analysis is an exploratory tool that provides new perspectives on combinations of items.

## Key Words

a priori algorithm
antecedent
association rules
confidence
consequent
cross-support
curse of dimensionality
data associations
interestingness
itemsets
lift
market basket
minimum confidence
minimum support
recommendation engine
relative risk
rules
Simpson's paradox
skewed support
spurious correlation
support
unsupervised learning

## Review Questions

1. What business tasks are suited to association analysis?

2. What are the statistical definitions of support, confidence, and lift?

3. Why are different measures needed (and others created) for evaluating association rules?

4. How does the a priori algorithm solve the problem of the curse of dimensions? What problems does its solution create?

5. What steps can be taken to reduce problems with dimensionality on huge datasets?

6. How can Simpson's paradox affect market basket analysis and what steps should be taken to reduce its impact?

7. What problems are created when a few items or categories appear in most of the transactions?

8. Why do the algorithms only use discrete data and how can continuous data be used in an association rules problem?

9. Why is transaction data from a DBMS often a problem for tools that estimate association rules?

10. How is the Microsoft Association Rules tool different from traditional tools?

## Exercises

### Book

1. Perform the traditional association rules analysis on the Bakery dataset by category. Comment on the results.

2. Perform the Microsoft Association Rules analysis on the Bakery dataset. Comment on the results.

3. Reduce the minimum confidence level in the Microsoft Association Rules and comment on the number of rules added to the result set and the processing time.

4. Run the Microsoft Association Rules analysis at the product level instead of category. Comment on the results and the processing time. If no rules appear on the initial run, reduce the minimum confidence level.

### Rolling Thunder Database

5. Run an association analysis on the Rolling Thunder component sales. Do the results match the component groups?

6. Create a query that lists ModelType purchased by customer—at any time. Run an association analysis to see if some model types are more commonly purchased together.

7. Create a query that lists ModelType purchased within a state for a specific month. For example, create StateYearMonth = SaleState + '_' +Year(OrderDate)*100 +Month(OrderDate). Run an association analysis using StateYearMonth as the transaction key and ModelType as the dimensions. Comment on any results. How would these results be interpreted differently from those in the prior exercise?

8. Create a query that defines a new column to combine SaleState and ModelType. For example, create StateModel = SaleState + '_' + ModelType. Also define a column that includes the year and month of the sale as the transaction key: YearMonth = Year(OrderDate)*100 + Month(OrderDate). Run an association analysis and comment on the results. What is the goal of this specific model?

## Diner

9. Are there associations between type of meal, day of week, and gender? With Microsoft Association, you can simply add all three as predictable and input columns. With traditional tools, use DinerID as the key and define a new column to use for the analysis as: DOW + MealTime + Gender. Explain how at least one of the rules could be useful to the business. What other data might you want to include to make the results more meaningful?

10. Are there any useful associations between type of meal, gender, and the total bill? See the hint in the prior question, but also recognize that the BillTotal is continuous. Compute the average to discretize the bill into low and high.

## Corner Med

11. What diagnoses codes commonly occur together? Hint: Diagnoses codes are hierarchical. The first (left) character is the primary level. The first three are the main category.

12. What procedures commonly occur together? Hint: Procedure codes are hierarchical. The first (left) character is the primary level. The first three are the main category.

13. What procedures have commonly been done on patients at any time?

14. What drugs are commonly prescribed in the same visit?

15. What drugs are commonly taken by the same patients at any time?

16. Are there procedure associations among employees? That is, using employees as the transaction key, are there association rules? What do they mean?

## Basketball

17. Using Player and Game as the transaction ID, discretize the data and determine if there are associations between field goals attempted, three-points attempted, free throws attempted, and assists (in one season).

18. Using the team as the transaction key, are there associations among the colleges attended by the players on the teams.

19. Using GameID and TeamID as the transaction, are there any association rules between IsHomeTeam, IsPlayoff, and WonLoss?

## Bakery

20. Which products are commonly purchased together?

21. Split the data into three sets based on time of day (breakfast, lunch, and afternoon. Determine which products are purchased together in each of the three sets. Comment on the results.

22. Split the data into two sets based on day of week: Weekend (including Friday) versus the rest of the week. Determine which items are commonly purchased together in both of the data sets. Comment on the results.

## Cars

23. What associations exist between attributes in the cars database?

## Teamwork

24. Use queries to filter the Bakery data to days of the week, or at least certain groups such as work days versus the weekend. Assign one day to each team member and run the association analysis. Comment on any differences in rules among the team's results and the overall results.

25. Use queries to filter the Bakery data by seasons—particularly holidays and summer. Assign team members to each set of data and run the association analysis. Comment on any differences in rules among the team's results and the overall results.

26. Use queries to filter the Corner Med data by day of week. Assign each day to a team member and see if the associations among procedures vary by day.

27. Using the basketball database, use queries to assign one basketball team to each team member. Discretize the data and determine if there are associations between minutes played and field goal percentage. Compare the results for each team.

## Additional Reading

Agrawal, R., H. Mannila, R. Srikant, H. Toivonen and A.I. Verkamo, 1995, Fast Discovery of Association Rules, *Advances in Knowledge Discovery and Data Mining*, AAAI/MIT Press, Cambridge, MA. [The article that first described the a priori algorithm for fast association searching.]

Bickel P., E. Hammel, J. O'Connell, 1975, Sex Bias in Graduate Admissions: Data from Berkeley, Science (187), 398-404. [Classic example: Berkeley lawsuit on gender discrimination.]

Geng, Liqiang and Howard J. Hamilton, 2006, Interestingness Measures for Data Mining: A Survey, *ACM Computing Surveys*, 38(3),

Hastie, Trevor, Robert Tibshirani, and Jerome Friedman, 2009, *The Elements of Statistical Learning/2e*, Springer: New York. [An outstanding book on data mining, with an emphasis on theory. A graduated-level book that requires a strong mathematics background.]

E.H. Simpson, 1951, The Interpretation of Interaction in Contingency Tables, Journal of the Royal Statistical Society, Series B, (13), 238-241. [Original description of Simpson's paradox. Many Web sites provide analysis and examples.]

Tan, Pang-Ning, Michael Steinbach, Vipin Kumar, 2005, Introduction to Data Mining, Addison Wesley: New York. [A computer-science approach to data mining with algorithms and computational analysis.]

G.H. Yule, 1903, Notes on the Theory of Association of Attributes in Statistics, Biometrika, (2), 121-134. [The earliest version of the Yule-Simpson effect or Simpson's paradox.]