CHAPTER 14
Association Rules and Collaborative Filtering

In this chapter, we describe the unsupervised learning methods of association rules (also called "affinity analysis" and "market basket analysis") and collaborative filtering. Both methods are popular in marketing for cross-selling products associated with an item that a consumer is considering.

In association rules, the goal is to identify item clusters in transaction-type databases. Association rule discovery in marketing is termed "market basket analysis" and is aimed at discovering which groups of products tend to be purchased together. These items can then be displayed together, offered in post-transaction coupons, or recommended in online shopping. We describe the two-stage process of rule generation and then assessment of rule strength to choose a subset. We look at the popular rule-generating Apriori algorithm, and then criteria for judging the strength of rules.

In collaborative filtering, the goal is to provide personalized recommendations that leverage user-level information. User-based collaborative filtering starts with a user, then finds users who have purchased a similar set of items or ranked items in a similar fashion, and makes a recommendation to the initial user based on what the similar users purchased or liked. Item-based collaborative filtering starts with an item being considered by a user, then locates other items that tend to be co-purchased with that first item. We explain the technique and the requirements for applying it in practice.

14.1 Association Rules

Put simply, association rules, or *affinity analysis*, constitute a study of "what goes with what." This method is also called *market basket analysis* because it originated with the study of customer transactions databases to determine dependencies between purchases of different items. Association rules are heavily used in retail for learning about items that are purchased together, but they are also useful in other fields. For example, a medical researcher might want to learn what symptoms appear together. In law, word combinations that appear too often might indicate plagiarism.

Discovering Association Rules in Transaction Databases

The availability of detailed information on customer transactions has led to the development of techniques that automatically look for associations between items that are stored in the database. An example is data collected using bar-code scanners in supermarkets. Such *market basket databases* consist of a large number of transaction records. Each record lists all items bought by a customer on a single-purchase transaction. Managers are interested to know if certain groups of items are consistently purchased together. They could use such information for making decisions on store layouts and item placement, for cross-selling, for promotions, for catalog design, and for identifying customer segments based on buying patterns. Association rules provide information of this type in the form of "if–then" statements. These rules are computed from the data; unlike the if–then rules of logic, association rules are probabilistic in nature.

Association rules are commonly encountered in online *recommendation systems* (or *recommender systems*), where customers examining an item or items for possible purchase are shown other items that are often purchased in conjunction with the first item(s). The display from Amazon.com's online shopping system illustrates the application of rules like this under "Frequently bought together." In the example shown in Figure 14.1, a user browsing a Samsung Galaxy S5 cell phone is shown a case and a screen protector that are often purchased along with this phone.

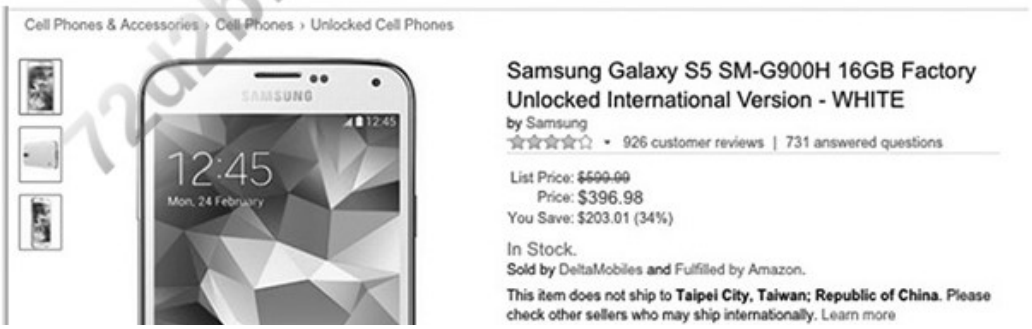

Figure 14.1 Recommendations under "Frequently bought together" are based on association rules

We introduce a simple artificial example and use it throughout the chapter to demonstrate the concepts, computations, and steps of association rules. We end by applying association rules to a more realistic example of book purchases.

Example 1: Synthetic Data on Purchases of Phone Faceplates

A store that sells accessories for cellular phones runs a promotion on faceplates. Customers who purchase multiple faceplates from a choice of six different colors get a discount. The store managers, who would like to know what colors of faceplates customers are likely to purchase together, collected the transaction database as shown in Table 14.1.

Table 14.1 Transactions Database for Purchases of Different-Colored Cellular Phone Faceplates

Transaction	Faceplate Colors Purchased			
1	red	white	green	
2	white	orange		
3	white	blue		
4	red	white	orange	
5	red	blue		
6	white	blue		
7	red	blue		
8	red	white	blue	green

9	red	white	blue	
10	yellow			

Generating Candidate Rules

The idea behind association rules is to examine all possible rules between items in an if–then format, and select only those that are most likely to be indicators of true dependence. We use the term *antecedent* to describe the IF part, and *consequent* to describe the THEN part. In association analysis, the antecedent and consequent are sets of items (called *itemsets*) that are disjoint (do not have any items in common). Note that itemsets are not records of what people buy; they are simply possible combinations of items, including single items.

Returning to the phone faceplate purchase example, one example of a possible rule is "if red, then white," meaning that if a red faceplate is purchased, a white one is, too. Here the antecedent is *red* and the consequent is *white*. The antecedent and consequent each contain a single item in this case. Another possible rule is "if red and white, then green." Here the antecedent includes the itemset {*red*, *white*} and the consequent is {*green*}.

The first step in association rules is to generate all the rules that would be candidates for indicating associations between items. Ideally, we might want to look at all possible combinations of items in a database with p distinct items (in the phone faceplate example, $p = 6$). This means finding all combinations of single items, pairs of items, triplets of items, and so on, in the transactions database. However, generating all these combinations requires a long computation time that grows exponentially[1] in p. A practical solution is to consider only combinations that occur with higher frequency in the database. These are called *frequent itemsets*.

Determining what qualifies as a frequent itemset is related to the concept of *support*. The support of a rule is simply the number of transactions that include both the antecedent and consequent itemsets. It is called a support because it measures the degree to which the data "support" the validity of the rule. The support is sometimes expressed as a percentage of the total number of records in the database. For example, the support for the itemset {red,white} in the phone faceplate example is 4 (or, $100 \times \frac{4}{10} = 40\%$).

What constitutes a frequent itemset is therefore defined as an itemset that has a support that exceeds a selected minimum support, determined by the user.

The Apriori Algorithm

Several algorithms have been proposed for generating frequent itemsets, but the classic algorithm is the *Apriori algorithm* of Agrawal et al. (1993). The key idea of the algorithm is to begin by generating frequent itemsets with just one item (one-itemsets) and to recursively generate frequent itemsets with two items, then with three items, and so on, until we have generated frequent itemsets of all sizes.

It is easy to generate frequent one-itemsets. All we need to do is to count, for each item, how many transactions in the database include the item. These transaction counts are the supports for the one-itemsets. We drop one-itemsets that have support below the desired minimum support to create a list of the frequent one-itemsets.

To generate frequent two-itemsets, we use the frequent one-itemsets. The reasoning is that if a certain one-itemset did not exceed the minimum support, any larger size itemset that includes it will not exceed the minimum support. In general, generating k-itemsets uses the frequent $(k - 1)$-itemsets that were generated in the preceding step. Each step requires a single run through the database, and therefore the Apriori algorithm is very fast even for a large number of unique items in a database.

Selecting Strong Rules

From the abundance of rules generated, the goal is to find only the rules that indicate a strong dependence between the antecedent and consequent itemsets. To measure the strength of association implied by a rule, we use the measures of *confidence* and *lift ratio*, as described below.

Support and Confidence

In addition to support, which we described earlier, there is another measure that expresses the degree of uncertainty about the if–then rule. This is known as the *confidence*[2] of the rule. This measure compares the co-occurrence of the antecedent and consequent itemsets in the database to the occurrence of the antecedent itemsets. Confidence is defined as the ratio of the number of transactions that include all antecedent and consequent itemsets (namely, the support) to the number of transactions that include all the antecedent itemsets:

support) to the number of transactions that include all the antecedent itemsets:

$$\text{Confidence} = \frac{\text{no. transactions with both antecedent and consequent itemsets}}{\text{no. transactions with antecedent itemset}}.$$

For example, suppose that a supermarket database has 100,000 point-of-sale transactions. Of these transactions, 2000 include both orange juice and (over-the-counter) flu medication, and 800 of these include soup purchases. The association rule "IF orange juice and flu medication are purchased THEN soup is purchased on the same trip" has a support of 800 transactions (alternatively, 0.8% = 800/100,000) and a confidence of 40% (= 800/2000).

To see the relationship between support and confidence, let us think about what each is measuring (estimating). One way to think of support is that it is the (estimated) probability that a transaction selected randomly from the database will contain all items in the antecedent and the consequent:

$$\text{Support} = \hat{P}(\text{antecedent AND consequent}).$$

In comparison, the confidence is the (estimated) *conditional probability* that a transaction selected randomly will include all the items in the consequent *given* that the transaction includes all the items in the antecedent:

$$\text{Confidence} = \frac{\hat{P}(\text{antecedent AND consequent})}{\hat{P}(\text{antecedent})} = \hat{P}(\text{consequent} \mid \text{antecedent}).$$

A high value of confidence suggests a strong association rule (in which we are highly confident). However, this can be deceptive because if the antecedent and/or the consequent has a high level of support, we can have a high value for confidence even when the antecedent and consequent are independent! For example, if nearly all customers buy bananas and nearly all customers buy ice cream, the confidence level of a rule such as "IF bananas THEN ice-cream" will be high regardless of whether there is an association between the items.

Lift Ratio

A better way to judge the strength of an association rule is to compare the confidence of the rule with a benchmark value, where we assume that the occurrence of the consequent itemset in a transaction is independent of the occurrence of the antecedent for each rule. In other words, if the antecedent and consequent itemsets are independent, what confidence values would we expect to see? Under independence, the support would be

$$P(\text{antecedent AND consequent}) = P(\text{antecedent}) \times P(\text{consequent}),$$

and the benchmark confidence would be

$$\frac{P(\text{antecedent}) \times P(\text{consequent})}{P(\text{antecedent})} = P(\text{consequent}).$$

The estimate of this benchmark from the data, called the *benchmark confidence value* for a rule, is computed by

$$\text{Benchmark confidence} = \frac{\text{no. transactions with consequent itemset}}{\text{no. transactions in database}}.$$

We compare the confidence to the benchmark confidence by looking at their ratio: this is called the *lift ratio* of a rule. The lift ratio is the confidence of the rule divided by the confidence, assuming independence of consequent from antecedent:

$$\text{lift ratio} = \frac{\text{confidence}}{\text{benchmark confidence}}.$$

A lift ratio greater than 1.0 suggests that there is some usefulness to the rule. In other words, the level of association between the antecedent and consequent itemsets is higher than would be expected if they were independent. The larger the lift ratio, the greater the strength of the association.

To illustrate the computation of support, confidence, and lift ratio for the cellular phone faceplate example, we introduce an alternative presentation of the data that is better suited to this purpose.

Data Format

Transaction data are usually displayed in one of two formats: a transactions database (with each row representing a list of items purchased in a single transaction), or a binary incidence matrix in which columns are items, rows again represent transactions, and each cell has either a 1 or a 0, indicating the presence or absence of an item in the transaction. For example, Table 14.1 displays the data for the cellular faceplate purchases in a transactions database. We translate these into binary incidence matrix format in Table 14.2.

Table 14.2 Phone Faceplate Data in Binary Incidence Matrix Format

Transaction	Red	White	Blue	Orange	Green	Yellow
1	1	1	0	0	1	0
2	0	1	0	1	0	0
3	0	1	1	0	0	0
4	1	1	0	1	0	0
5	1	0	1	0	0	0
6	0	1	1	0	0	0
7	1	0	1	0	0	0
8	1	1	1	0	1	0
9	1	1	1	0	0	0
10	0	0	0	0	0	1

Now suppose that we want association rules between items for this database that have a support count of at least 2 (equivalent to a percentage support of 2/10 = 20%): In other words, rules based on items that were purchased together in at least 20% of the transactions. By enumeration, we can see that only the itemsets listed in Table 14.3 have a count of at least 2.

Table 14.3 Itemsets with Support Count of At Least Two

Itemset	Support (Count)
{red}	6
{white}	7
{blue}	6
{orange}	2
{green}	2
{red, white}	4
{red, blue}	4
{red, green}	2
{white, blue}	4
{white, orange}	2
{white, green}	2
{red, white, blue}	2
{red, white, green}	2

The first itemset {red} has a support of 6, because six of the transactions included a red faceplate. Similarly, the last itemset {red, white, green} has a