e-PGPathshala
Subject : Computer Science
Paper: Data Analytics
Module No 23: CS/DA/23 - Data Analytics – Association Rule Mining - I
Quadrant 1 – e-text

1.1 Introduction

Mining Association Rules is one of the most used functions in data mining. This chapter gives the basics of association rule analysis (ARA) and applications of ARA and also gives a glance on apriori approach for ARA.

1.2 Learning Outcomes

- To Understand the basics of association rule analysis (ARA)
- To know about applications of ARA
- To gain knowledge about apriori approach for ARA

1.3 Association rule mining

Most of the machine learning algorithms that are used for data mining and data science also work with numeric data. But, association rule mining is perfect for categorical (non-numeric) data and it involves little more than simple counting. Association mining can be defined as finding frequent patterns, associations, correlations, or causal structures among sets of items or objects in transaction databases, relational databases, and other information repositories. Association rule mining is mainly focused on finding frequent co-occurring associations among a collection of items. It also used to understand customer buying habits by finding associations and correlations between the different items that customers place in their shopping basket.

Basket data analysis, cross-marketing, catalog design, loss-leader analysis, web log analysis, fraud detection (supervisor->examiner) are some applications of association rule mining

1.4 Association rule discovery

Association rules are if/then statements that help uncover relationships between seemingly unrelated data in a relational database or other information repository. An association rule has two parts, an antecedent (if) and a consequent (then). An antecedent is an item found in the data. A consequent is an item that is found in combination with the antecedent**.**

The set of items a customer buys is referred to as an itemset, and market basket analysis seeks to find relationships between purchases. Typically the relationship will be in the form of a rule:

- Goal: Identify items that are bought together by sufficiently many customers
- Approach: Process the sales data collected with barcode scanners to find dependencies among items
- A classic rule:If someone buys toothpaste and brush, then he/she is likely to buy mouth cleaner

Example: IF {toothpaste, brush} THEN {mouth cleaner}

“Body → Head [support, confidence]”

– buys(x, “Shaving Razor”) → buys(x, “Shaving Gel”) [0.5%, 60%]

– major(x, “CS”) ^ takes(x, “DB”) → grade(x, “A”) [1%, 75%]

1.4.1 Supermarket shelf management – Market-basket model:

Market Basket Analysis is a modelling technique based upon the theory that if you buy a certain group of items, you are more (or less) likely to buy another group of items.Market Basket Analysis is one of the most common and useful types of data analysis for marketing and retailing. The purpose of market basket analysis is to determine what products customers purchase together. It takes its name from the idea of customers throwing all their purchases into a shopping cart (a "market basket") during grocery shopping. Knowing what products people purchase as a group can be very helpful to a retailer or to any other company. A store could use this information to place products frequently sold together into the same area, while a catalog or World Wide Web merchant could use it to determine the layout of their catalog and order form. Direct marketers could use the basket analysis results to determine what new products to offer their prior customers.

In market-basket model we have mainly 3 main concepts as shown in figure 1.

- A large set of **items** e.g., things sold in a supermarket
- A large set of **baskets** where, each basket is a **small subset of items,** e.g., the things one customer buys on one day
- **Association rules**, e.g., People who bought {x,y,z} tend to buy {v,w}

TID	*Items*
1	Bread, Biscut, Milk
2	Sugar, Bread
3	Sugar, Biscut, CoffeePowder, Milk
4	Sugar, Bread, CoffeePowder, Milk
5	Biscut, CoffeePowder, Milk

Rules Discovered:
{Milk} --> {Biscut}
{CoffeePowder, Milk} --> {Sugar}

Figure 1. Market basket model

In figure one transactions (indicated by TID) indicate baskets and each basket has small subset of items. The rules discovered are the association rules, which

indicates that if milk is purchased then biscut will also be purchased and similarly if coffee powder and milk are purchased then sugar will be purchased.

The strength of market basket analysis is that by using computer data mining tools, it's not necessary for a person to think of what products consumers would logically buy together – instead, the customers' sales data is allowed to speak for itself. This is a good example of data-driven marketing.

Once it is known that customers who buy one product are likely to buy another, it is possible for the company to market the products together, or to make the purchasers of one product the target prospects for another. Likewise, if it's known that customers who buy a sweater and casual pants from a certain mail-order catalog have a propensity toward buying a jacket from the same catalog, sales of jackets can be increased by having the telephone representatives describe and offer the jacket to anyone who calls in to order the sweater and pants. Still better, the catalogue company can provide an additional 5% discount on a package containing the sweater, pants, and jacket simultaneously and promote well the complete package. The dollar amount of sales is guaranteed to go up. By targeting customers who are already known to be likely buyers, the effectiveness of marketing is significantly increased – regardless of if the marketing takes the form of in-store displays, catalog layout design, or direct offers to customers. This is the purpose of market basket analysis – to improve the effectiveness of marketing and sales tactics using customer data already available to the company.

Application

Although Market Basket Analysis conjures up pictures of shopping carts and supermarket shoppers, it is important to realize that there are many other areas in which it can be applied. These include:

- Telecommunication (each customer is a transaction containing the set of phone calls)

- Credit Cards/ Banking Services (each card/account is a transaction containing the set of customer's payments)

- Medical Treatments (each patient is represented as a transaction containing the ordered set of diseases)

- Basketball-Game Analysis (each game is represented as a transaction containing the ordered set of ball passes) Analysis of credit card purchases.

Note that despite the terminology, there is no requirement for all the items to be purchased at the same time. The algorithms can be adapted to look at a sequence of purchases (or events) spread out over time. A predictive market basket analysis can be used to identify sets of item purchases (or events) that generally occur in sequence — something of interest to direct marketers, criminologists and many others.

Application Example 1: If Items = products; Baskets = sets of products someone bought in one trip to the store. In **Real market baskets,** Chain stores keep TBs of data about what customers buy together which:

- Tells how typical customers navigate stores, lets them position tempting items
- Suggests tie-in "tricks", e.g., run sale on Sugar and raise the price of Coffee Powder
- Need the rule to occur frequently, or no profit

Application Example 2: Consider a text database, where Baskets = sentences and Items = documents containing those sentences, the the following may be inferred:

- Items that appear together too often could represent plagiarism
- Notice items do not have to be "in" baskets

Application Example 3: In healthcare domain, if Baskets = patients; Items = drugs & side-effects, then we can detect combinations of drugs that result in particular side-effects. But thus requires extension, that, absence of an item needs to be observed as well as presence.

More generally, it is a many-to-many mapping (association) between two kinds of things. But we ask about connections among "items", not "baskets", for example: Finding communities in graphs (e.g., Twitter).

Application Example 4: Finding communities in graphs (e.g., Twitter) can be visualized when Baskets = nodes and Items = outgoing neighbors. It is more like searching for complete bipartite subgraphs $K_{s,t}$ of a big graph as shown in figure 2 and the steps given below:

- View each node ***i*** as a basket B_i of nodes it points to
- $K_{s,t}$ = a set ***Y*** of size ***t*** that occurs in *s* buckets B_i
- Looking for $K_{s,t}$ → set of support ***s*** and look at layer ***t*** – all frequent sets of size ***t***

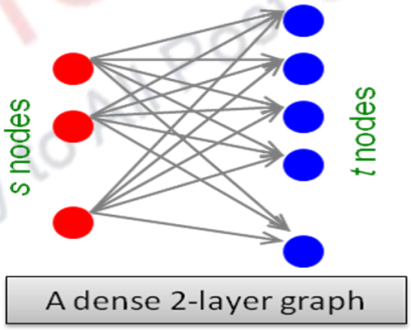


Figure 2. Complete bipartite subgraphs

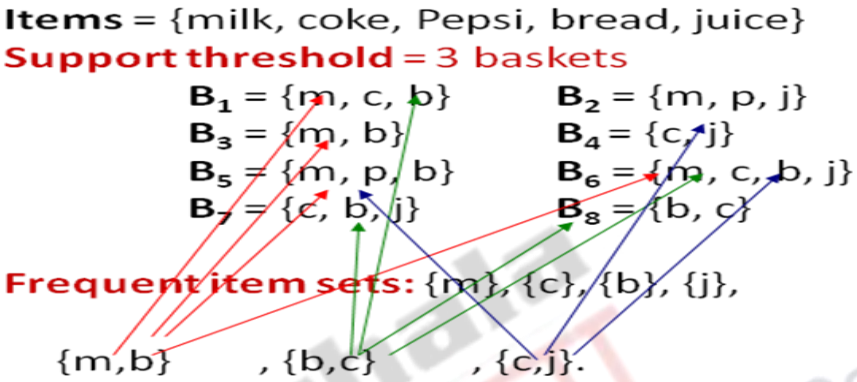
1.5 Association Rule: Basic Concepts

- Itemset - A set of items
- k-itemset - An item set that contains k items, e.g. {bread, jam} is a 2 - itemset
- Occurrence frequency of an itemset – no. of transactions that contain the itemset

- An itemset satisfies minimum support if Occurrence freq. >=min. support X no.of transactions

- If an itemset satisfies minimum support, then it is a frequent itemset

Example for finding frequent itemsets is given below:



An association rule has two parts, an antecedent (if) and a consequent (then). An antecedent is an item found in the data. A consequent is an item that is found in combination with the antecedent.

An implication expression of the form X → Y, where X and Y are itemsets

Example:{Milk} → {Sugar}

Association rules are created by analyzing data for frequent if/then patterns and using the criteria support and confidence to identify the most important relationships. Support(s) is an indication of how frequently the items appear in the database. Confidence(c) indicates the number of times the if/then statements have been found to be true.

If the association is denoted by A →B [s, c]

Support (s): Fraction of transactions that contain both A and B.Or it denotes the frequency of the rule within transactions. A high value means that the rule involves a great part of database.

Support (A → B [s, c]) = p(A ∪ B)

Confidence (c): Measures how often items in Y appear in transactions that contain X

T denotes set if Transactions transactions or it denotes the percentage of transactions containing A which also contain B. It is an estimation of conditioned probability .

confidence(A $\rightarrow$ B [s, c]) = p(B|A) = sup(A,B)/sup(A).

Example: Support and Confidence – Consider the list of transactions given below:

Transaction ID	Items Bought
2000	A,B,C
1000	A,C
4000	A,D
5000	B,E,F

Let minimum support 50%, and minimum confidence 50%, then we have two rules: A $\Rightarrow$ C (50%, 66.6%) and C $\Rightarrow$ A (50%, 100%)

Rule Evaluation:

In practice there are many rules, and if we want to find significant/interesting ones, we can use confidence of an association rule, which is the probability of j given I = $\{i_1,\ldots,i_k\}$

$$\text{conf}(I \rightarrow j) = \frac{\text{support}(I \cup j)}{\text{support}(I)}$$

However, not all high-confidence rules are interesting. For example, the rule ***X*** $\rightarrow$ ***milk*** may have high confidence for many itemsets ***X***, because milk is just purchased very often (independent of ***X***) and the confidence will be high. Hence we can make use of another metric called Interest.

Interest of an association rule ***I*** $\rightarrow$ ***j*** is the difference between its confidence and the fraction of baskets that contain ***j*** **.** Interesting rules are those with high positive or negative interest values (usually above 0.5)

Consider the example given below highlighting the importance of interest metric with respect to confidence. Assume the following 8 baskets:

B_1 = {m, c, b} B_2 = {m, p, j}

B_3 = {m, b} B_4= {c, j}

B_5 = {m, p, b} B_6 = {m, c, b, j}

B_7 = {c, b, j} B_8 = {b, c}

For the association rule: {m, b} →c, derived from the above data, Confidence = 2/4 = 0.5 and Interest = |0.5 – 5/8| = 1/8. From which we can understand that, item c appears in 5/8 of the baskets and therefore we can conclude that the rule is not very interesting!

1.5.2 Mining single-dimensional Boolean association rules from transactional databases

If the items or attributes in an association rule reference only one dimension,then it is a single-dimensional association rule. If a rule references two or more dimensions, such as the dimensions buys, time_of_transaction and customer_category, then it is a multidimensional association rule.

Mining Association Rules—An Example: For the below given transactional data (Table 1 (a)), if min_sup=50% and min_confidence=50%, frequent itemsets are as listed below in table 1(b):

Table 1 (a): Transactional Data

Transaction ID	Items Bought
2000	A,B,C
1000	A,C
4000	A,D
5000	B,E,F

Table 1 (b): Frequent Itemsets

Frequent Itemset	Support
{A}	75%
{B}	50%
{C}	50%
{A,C}	50%

Consider a rule $A \Rightarrow C$ derived from frequent 2-itemset in table 1(b):

support = support({$A \wedge C$}) = 50%

confidence = support({$A \wedge C$})/support({A}) = 66.6%

1.6 The Apriori Algorithm:

Apriori is a classic algorithm for learning association rules. It is designed to operate on databases containing transactions (for example, collections of items bought by customers, or details of a website frequentation). Apriori is an influential algorithm for mining frequent itemsets for boolean association rules. The name of the algorithm is based on the fact that the algorithm uses prior knowledge of frequent itemset.

General Process

Association rule generation is usually split up into two separate phases:

Phase 1: Find the frequent itemsets: the sets of items that have minimum support

- A subset of a frequent itemset must also be a frequent itemset
  - i.e., if {AB} is a frequent itemset, both {A} and {B} should be a frequent itemset
- Iteratively find frequent itemsets with cardinality from 1 to k (k-itemset)

Phase 2: Use the frequent itemsets to generate association rules.

While the second phase is straight forward, the first phase needs more attention. Finding all frequent itemsets in a database is difficult since it involves searching all possible itemsets(item combinations). The set of possible itemsets is the power set over I and has size $2^n - 1$ (excluding the empty set which is not a valid itemset). Although the size of the powerset grows exponentially in the number of items n in I, efficient search is possible using the ***downward-closure property*** of support (also called *anti-monotonicity*) which guarantees that for a frequent itemset, all its subsets are also frequent and thus for an infrequent itemset, all its supersets must also be infrequent. Exploiting this property, efficient algorithms (e.g., Apriori and Eclat) can find all frequent itemsets.

Apriori Algorithm Pseudocode

procedure **Apriori** (T, *minSupport*) { //T is the database and *minSupport* is the minimum support

L1= {frequent items};

for (k= 2; L_{k-1} != ϕ ; k++) **{**

C_k= candidates generated from L_{k-1}

//that iscartesian product L_{k-1} x L_{k-1} and eliminating any k-1 size itemset that is //not frequent

for each transaction **t** in database **do{**

#increment the count of all candidates in Ck that are contained in t
L_k = candidates in C_k with *minSupport*

}//end for each
}//end for
return U_k L_k;

}

As is common in association rule mining, given a set of *itemsets* (for instance, sets of retail transactions, each listing individual items purchased), the algorithm attempts to find subsets which are common to at least a minimum number C of the itemsets. Apriori uses a "bottom up" approach, where frequent subsets are extended one item at a time (a step known as *candidate generation*), and groups of candidates are tested against the data. The algorithm terminates when no further successful extensions are found.

Apriori uses breadth-first search and a tree structure to count candidate item sets efficiently. It generates candidate item sets of length *k* from item sets of length *k* − 1. Then it prunes the candidates which have an infrequent sub pattern. According to the downward closure lemma, the candidate set contains all frequent *k*-length item sets. After that, it scans the transaction database to determine frequent item sets among the candidates.

Apriori, while historically significant, suffers from a number of inefficiencies or trade-offs, which have spawned other algorithms. Candidate generation generates large numbers of subsets (the algorithm attempts to load up the candidate set with as many as possible before each scan). Bottom-up subset exploration (essentially a breadth-first traversal of the subset lattice) finds any maximal subset S only after all $2^{|S|} - 1$ of its proper subsets.

Sample usage of Apriori algorithm

A large supermarket tracks sales data by Stock-keeping unit (SKU) for each item, and thus is able to know what items are typically purchased together. Apriori is a moderately efficient way to build a list of freq. Let the database of transactions consist of the sets {1,2,3,4}, {1,2,3,4,5}, {2,3,4}, {2,3,5}, {1,2,4}, {1,3,4}, {2,3,4,5}, {1,3,4,5}, {3,4,5}, {1,2,3,5}. Each number corresponds to a product such as "butter" or "water". The first step of Apriori is to count up the frequencies, called the supports, of each member item separately:

Item	Support
1	6
2	7
3	9
4	8
5	6

We can define a minimum support level to qualify as "frequent," which depends on the context.For this case, let min support = 4. Therefore, all are frequent. The next step is to generate a list of all 2-pairs of the frequent items. Had any of the above items not been frequent, they wouldn't have been included as a possible member of possible 2-item pairs. In this way, Apriori *prunes* the tree of all possible sets. In next step we again select only these items (now 2-pairs are items) which are frequent .

Item	Support
{1,2}	4
{1,3}	5
{1,4}	5
{1,5}	3
{2,3}	6
{2,4}	5
{2,5}	4
{3,4}	7
{3,5}	6
{4,5}	4

We generate the list of all 3-triples of the frequent items (by connecting frequent pair with frequent single item).

Item	Support
{1,3,4}	4
{2,3,4}	4
{2,3,5}	4
{3,4,5}	4

The algorithm will end here because the pair {2,3,4,5} generated at the next step does not have the desired support. We will now apply the same algorithm on the same set of data considering that the min support is 5. We get the following results:

Step 1:

Item	Support
1	6
2	7
3	9
4	8
5	6

Step 2:

Item	Support
{1,2}	4
{1,3}	5
{1,4}	5
{1,5}	3
{2,3}	6
{2,4}	5
{2,5}	4
{3,4}	7
{3,5}	6
{4,5}	4

The algorithm ends here because none of the 3-triples generated at Step 3 have de desired support.

Case Studies

A) **Case Studies in Association Rule Mining for Recommender Systems**

https://www.researchgate.net/publication/220835452_Case_Studies_in_Association_Rule_Mining_for_Recommender_Systems

Recommender systems combine ideas from information retrieval, machine learning and user profiling research in order to provide end-users with more proactive and personalized information retrieval applications. Two popular approaches have come to dominate. Content-based techniques leverage the availability of rich item descriptions to identify new items that are similar to those that a user has liked in the past. In contrast, collaborative filtering techniques rely on the availability of user profiles in which sets of items have been rated. They recommend new items to a target user on the basis that similar users have preferred these items in the past. This paper presents two case-studies of how association rule mining techniques have been used to significantly enhance the power of content-based and collaborative filtering recommender systems.

Summary

- In real world associations between item pairs provide great insight to business
- Associations can be effectively captures using apriori method