A Sequential Approach for Predicting Missing Items in Shopping Cart Using Apriori Algorithm

Ruchira Bodakhe\textsuperscript{1}, Prajakta Gotarkar\textsuperscript{2}, Ashvini Dahiwade\textsuperscript{3}
Pooja Gosavi\textsuperscript{4}, Prof. J.I.Syed\textsuperscript{5}
\textsuperscript{1,2,3,4}Student, Computer Science and Engineering, H.V.P.M, COET, Amravati, India
\textsuperscript{5}Assistant Professor, Computer Science and Engineering, H.V.P.M, COET, Amravati, India

Abstract: In today’s ever growing market it is very essential to keep track of the customer’s interests and keep them updated about the trends and products in the market. In this paper we aim to suggest a shopping portal and have a database that records the item sets that are frequently bought together using ‘APRIORI ALGORITHM’. This information will be used to flash advertisements and offers on the products of their interests. Also to promote new products relating to their requirements and mainly to suggest and prompt them about the products which are often bought along with the product already present in their shopping cart. The frequently co-occurring group of items is determined by frequent pattern mining in the databases. Here the major contributing task is expediting the frequent item sets by proposing a technique that uses the minimal data available in the shopping cart for the prediction of what other items the customer can choose to buy.

Keywords— Apriori Algorithm, Association rule mining, Data mining, frequent itemsets.

1. Introduction

The primary task of association mining is to detect frequently co-occurring groups of items in transactional databases. The intention is to use this knowledge for prediction purposes: if bread, butter, and milk often appear in the same transactions, then the presence of butter and milk in a shopping cart suggests that the customer may also buy bread. More generally, knowing which items a shopping cart contains, we want to predict other items that the customer is likely to add before proceeding to the checkout counter. This paradigm can be exploited in diverse applications For example, in the domain discussed in [1], each “shopping cart” contained a set of hyperlinks pointing to a Web page [1]; in medical applications, the shopping cart may contain a patient’s symptoms, results of lab tests, and diagnoses; in a financial domain, the cart may contain companies held in the same portfolio; and Bollmann-Sdorra et al. [2] proposed a framework that employs frequent item sets in the field of information retrieval. The prediction task was mentioned as early as in the pioneering association mining paper by Agrawal et al. [3], but the problem is yet to be investigated in the depth it deserves.

The literature survey in [4] indicates that most authors have focused on methods to expedite the search for frequent item sets, while others have investigated such special aspects as the search for time-varying associations [6] or the identification of localized patterns [8]. Still, some prediction-related work has been done as well. In our work, we wanted to make the next logical step by allowing any item — its value is to be predicted based on the presence or absence of other items. Put another way, knowing a subset of the shopping cart’s contents, we want to “guess” (predict) the rest. It is important to understand that allowing any item presents serious challenges as compared with the case of just a single item. The number of different items can be very high, perhaps hundreds, or thousand, or even more. To generate association rules for each of them separately would give rise to great many rules with two obvious consequences: first, the memory space occupied by these rules can be many times larger than the original database. second, identifying the most relevant rules and combining their sometimes conflicting predictions may easily incur prohibitive computational costs. In our work, we sought to solve both of these problems by developing a technique that answers user’s queries (for shopping cart completion) in a way that is acceptable not only in terms of accuracy, but also in terms of time and space complexity.

2. Implementation

2.1 Apriori Algorithm

Apriori algorithm (Agrawal et al. 1993) is easy to execute and very simple, is used to mine all frequent item sets in database [4]. In the process of Apriori,
the following definitions are needed [4]: Definition 1: Suppose \( T = \{ T_1, T_2, \ldots, T_m \} \), \( (m_1) \) is a set of transactions, \( T_i = \{ I_1, I_2, \ldots, I_n \} \), \( (n_1) \) is the set of items, and \( k \)-item set \( = \{ i_1, i_2, \ldots, i_k \} \), \( (k_1) \) is also the set of \( k \) items, and \( k \)-item set \( \subseteq I \).

Definition 2: Suppose \( (item \ set) \), is the support count of item set or the frequency of occurrence of an item set in transactions.

Definition 3: Suppose \( C_k \) is the candidate item set of size \( k \), and \( L_k \) is the frequent item set of size \( k \).

The key idea of Apriori algorithm is to make multiple passes over the database. It employs an iterative approach known as a breadth-first search (level-wise search) through the search space, where \( k \)-item sets are used to explore \( (k+1) \)-item sets [7].

It is used to generate all frequent item set (i.e. an item set [7] whose support is greater than some user-specified minimum support denoted \( L_k \), where \( k \) is the size of the item set). A Candidate item set is [7] a potentially frequent item set (denoted \( C_k \), where \( k \) is the size of the item set).

1. Pass 1
   1. Generate the candidate item sets in \( C_1 \)
   2. Save the frequent item sets in \( L_1 \)
2. Pass \( k \)
   (i). Generate the candidate item sets in \( C_k \) from the frequent item sets in \( L_{k-1} \)
   Join \( L_{k-1}p \) with \( L_{k-1}q \), as follows: insert not \( C_k \) select \( p.item1, p.item2,P.itemk-1, q.itemk-1 \) from \( L_{k-1}p, L_{k-1}q \) where, \( p.item1 = q.item1, \ldots p.itemk-2 = q.itemk-2, p.itemk-1<q. itemk-1 \)
   • Generate all \((k-1)\)-subsets from the candidate item sets in \( C_k \)
   • Prune all candidate item sets from \( C_k \) where, some \((k-1)\)-subset of the candidate item set is not in the frequent item set \( L_{k-1} \)
   (ii). Scan the transaction database to determine the support for each candidate item set in \( C_k \)
   (iii). Save the frequent item sets in \( L_k \) [7].

2.2 Limitations

Apriori algorithm too shows some loopholes in spite of being simple and clear. The main limitation is excessive wastage of time to hold a huge number of candidate item sets with [4] much frequent itemsets, low minimum support or large itemsets. For example, if there are 104 frequent 1-item sets, the Apriori algorithm will need to generate more than 107 length-2 candidates and accumulate and test their occurrence frequencies [7]. Furthermore, to detect frequent pattern in size 100 (e.g.) \( v_1, v_2 \ldots v_{100} \), it will be required to generate 2100 candidate item sets that yield on costly and wasting of time of candidate generation [7], no matter what implementation technique is applied [4]. Thus from candidate itemsets, it will check for multiple sets and also [4] scan database many times repeatedly for finding candidate itemsets. When the database is storing a large number of data services, the limited memory capacity, the system I/O load, considerably very long time will be consumed in scanning the database, so efficiency is very low[7].

3. Proposed Approach

The shopping cart prediction architecture is proposed in Fig.1. Based on passed transaction we can easily construct a Graph structure from which association rules are generated in consideration of new incoming instances in new transaction. Then the threshold value is set by the user and is kept dynamic, the prediction algorithm is executed which predicts the next new item set to be considered for purchase. Threshold value is the minimum support value that a particular pair has to be present before getting predicted.

Fig.1: Shopping cart prediction architecture

Input: set of items from database
Output: predicted items
1. For the given set of items item table is generated.
2. The item table is then converted to matrix form
3. Prediction of the items is done
4. The choice of customer is obtained.
5. Find the list of items that the customer has selected with the threshold value
6. Display this predict key item as the next item to be purchased.

Here, the set of input items is encapsulated into an Item table where unique key values are generated for each of the items in the database. Next the Association graph is constructed for every item purchased by the customer. The edge value denotes the number of occurrences or frequencies of the items purchased by the same or different customers.
At this stage the support value is defined. From the already generated frequent itemsets the association rules are to be generated. It takes minimum confidence from the user and discovers all rules with a fixed antecedent and with different consequent. The association rules generated form the basis for prediction.

4. Conclusion

In this paper, based on incomplete Information about the constituents of shopping cart, we can predict which other items the shopping cart contains. By using the apriori algorithm when presented with an incomplete list of items in a shopping cart, our program first identifies all high-support, high-confidence rules using Item set trees. Then, it combines the consequents of all these (sometimes conflicting) rules and creates a set of items most likely to complete the shopping cart.

The main objective of this paper is that implementation of architecture that predict the missing item more accurately and within limited time. By using the IT-tree data structure, rearranging of database faster is possible. This technique proves to appear better than the traditional techniques in association rule mining. Costs associated with rule pruning and rule selection can be effectively minimized by better organization of rules. The main advantage of this approach is that it can applied to the large dataset or any kind of dataset. It can surmise the multiple missing items while other technique can guess only single missing item. It gives the approximate 65% accurate results for prediction.

5. References


