Chapter 2

Literature Survey

2.1 Different Set Representation Techniques available

In programming, the sets are the fundamental visualization. A representation is considered the best depending on the type of operations used, time, and the memory cost. One such representation is the sparse-set representation consisting of the three components: scalar and two vectors. A scalar stores many members present in the set [14]. Figure 2-1 illustrates an example set with a single member 3. A set can be represented in an unordered fashion. The operations such as union, set difference, and intersection of the two sets are time-consuming. Another method for storing the element is representing an arbitrary order of the elements. Using the representation, the computation of the set becomes effortless. Assume that U is the universal set that is finite and $\{a_1, a_2, a_3, ..., a_n\}$ is the arbitrary

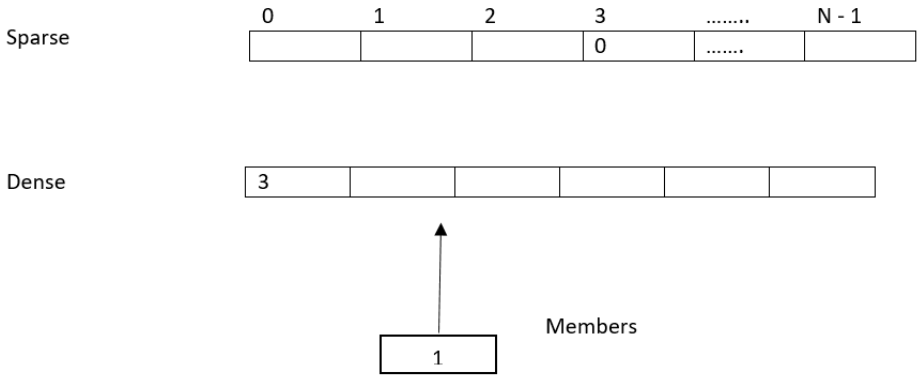

Figure 2-1: An example set of an illustration with a single member 3

ordering of elements. To represent a subset Z of U consisting of bit strings of say length n where bit of i^{th} position of the string is 1 if $a_i \in$ A and the bit of i^{th} position of the string is 0 if $a_i \notin$ A. A set enumeration tree is another alternative to representing the sets for an itemset I ={ A, B, C, D}. The subtree lists all the supersets in a lexicographic of the root node. The children of the node are represented using a linked list connecting as siblings. This representation needs two pointers for storage for each node[68]. A structure called free sets efficiently extracts dense datasets. The algorithm works on the principle of anti-monotonic that if the set is not a free-set all of its supersets are also not free-set. The algorithm starts with the set size of 0 or empty itemset. Then, it considers the set size of 1 and continues so on. When a set say A is not a free set then the set is pruned. This reduces the search space because it will consider supersets of any of set A. For example, the set A= {1, 4, 6, 7} is not considered because some of the subsets are not free set [13].

2.2 Generation of Itemsets with Candidate Generation

Candidate generation of the itemsets is based on the two methods. One is with generation of frequent itemsets with candidate itemsets and the other generation of frequent itemsets without generation of candidate itemsets.

2.2.1 Apriori

The Apriori Algorithm is one of the algorithms given by R. Agarwal and R. Srikant [1]. It is one of the most efficient association mining algorithms for rule discovery. The algorithm utilizes the previous information of the frequent itemsets. It adopts an iterative approach where k+1 itemsets are generated from k itemsets. The Apriori Algorithm reduces the search space. This property stated that the subset of any frequent itemset should also be frequent. If the itemset is infrequent then its subset can never be frequent [10].

The Apriori algorithm takes numerous passes on the database. During each pass, each itemset is extended. A counter is associated with an itemset. The purpose of the counter is to keep track of the count on the number of times a particular itemset has occurred in the database. Initially, the value of the counter was

zero when the itemset was built. In each pass, the support count of the itemsets is measured. The itemset's support count is compared with the threshold value of the support count to determine whether the itemset is large or not. Depending on the support count, it is determined if the itemset will be added to the set needed for the next pass. When the set becomes empty the algorithm terminates [1].

2.2.2 DIC

The number of passes in this algorithm is reduced. At each pass, the number of itemsets maintained is comparatively very low. The DIC runs over the database and stops at M intervals where M is a parameter value that ranges from 100 to 10,000. When the algorithm passes over the data and reaches the end of the transaction file then it starts again from the beginning for the succeeding pass. A count is maintained for the occurrence of the itemsets that occurred in the transactions. The itemsets are marked in the following four different ways:

- Solid box: contains itemsets whose support surpasses the threshold.
- Solid circle - itemsets whose support count is below a threshold value.
- Dashed box - contains itemsets that are still in the process of counting and suspected to exceed the threshold.
- Dashed circle - contains itemsets that are still in the process of counting and suspected to be below the threshold.

In the beginning, the Solid Circle, Dashed Box, and Solid Box are empty and Dashed Circle consists of 1-itemsets. For each transaction, DIC starts counting the support of the itemsets from the "dashed" sets. DIC moved the itemsets whose support count surpasses the threshold from the Dashed Circle to the Dashed Box. When the new itemsets are added to the Dashed Circle, they become the immediate supersets of those itemsets that are from the Dashed Box including all of its subsets. After the completion of the pass, the itemsets are moved from dashed to solid set. The DIC continues if any of the itemset are still present in the dashed sets [15] [48] [84].

2.2.3 SETM

The Set-Oriented Mining for Association Rules algorithm was introduced by Swami and Houtsma. Using relational operations, the algorithm can optimize the operations. The rule discovery can be integrated with the database system. The algorithm utilizes the sorting and the merge-scan join. SETM uses query language such as SQL for developing black box algorithms. The first sort is used for implementing merge and scan join. The second sorting is used for generating the support counts. A sequential scan the data is used for generating the support counts of the itemset. The tuples which do not meet the requirement are deleted using the look-up table. The tuples show the relations that are of interest for the rule generation[41]. The candidate itemsets generation process is separated from the counting process. The generated candidate itemsets are counted after the end of each pass. The identifier TID for each transaction is also saved sequentially. One of the disadvantages of this algorithm is that for a single candidate itemset there are many entries for support count [49].

2.2.4 AIS

The algorithm was introduced by Agarwal et .[1]. The algorithm focuses on the improvement of the database quality and the decision support system. It generates only one item in the consequent part algorithm. For example, the rules A $\cup$ B $\rightarrow$ C but not A$\rightarrow$ B $\cup$ C. To generate the frequent itemsets, the algorithm scans the database many times. A method such as the estimation method was incorporated into the algorithm for pruning the itemsets whose support is less than the threshold. This process consequently avoids the unnecessary steps of counting the itemsets again. One assumption is that all the candidate and frequent itemsets can fit in the main memory. This is usually a disadvantage if the database is too large.

2.3 Generation of Itemsets without Candidate Generation

2.3.1 FP-growth

The algorithm generates frequent itemsets without the process of generating the candidate itemsets. In this approach, the FP is a tree structure that is an extended tree structure for storing the crucial and compressed information on frequent itemsets. The three techniques which increase the efficiency of the algorithm are as follows:-

- The database is compressed into smaller structured and compressed data to avoid repeated database scans.
- The mining algorithms adopted the fragment pattern method for the generation of frequent instead of generating large amounts of candidate itemsets.
- A method of divide and conquer partitioning is introduced in the mining task which reduces the search space [37].

One of the challenges faced by the FP-Growth is storage. If the database is huge, the generated FP is large and cannot fit in the main memory[52].

2.3.2 ECLAT

The algorithm uses the vertical format for storing the information. Vertical format is expressed in the form of an item TID_set which is a set of items in the database and TID_set is the set of transactions that contains the item. For the generation of support degree of itemsets, computation of the intersection of all TID_sets is needed [61].

2.3.3 LCOFI

LCOFI is an algorithm that is built on the LCO algorithm. The representation of itemset is similar to the binary representation of the dataset. The algorithm scans the database to build a bi-partite graph of 1-frequent itemset. It does not generate candidate itemset [4].

2.3.4 IR-Eclat

IR-Eclat is a rare equivalence class transformation algorithm designed for mining for infrequent itemset. The itemsets are mined through different forms such as R-Eclat- Diffset, R-Eclat-Tidset, R-Eclat-Postdiffset, and REclat-Sortdiffset. Lastly, they are mined with the IR-Eclat engine. The algorithm is beneficial for the incremental database including deletion, addition of records, or transaction in the database [53].

2.3.5 CAFP

A Cellular learning automata (CLA) and multiple FP tree structures (CAFP) are used for mining itemset from the dataset. CLA scans the itemsets and transports them to the suitable cells. The cell processes the itemset and updates itself simultaneously. Each cell serves as a root in an FP-Tree that forms like a neighborhood that updates from the transactions. Later, a conditional pattern base is constructed for each 1-frequent item. Consequently, the itemsets are pruned from the tree based on a threshold value. Then, this is followed by generating all combinations of the frequent patterns. Lastly, the frequent itemsets are forwarded to the environment [33].

2.3.6 Meta-PCP

This algorithm used the concept of frequent closed itemsets for the data that is uncertain. This happens in two ways: Meta-PCP is a concise representation that eliminates redundant information. It is a lossy representation that is controlled by users according to domain experts. Furthermore, a query-based mining algorithm is used to improve the performance when the threshold value is set to a very small value. It generates frequent itemsets without generating candidate itemsets. the closed itemsets and generators are considered based on the probability of frequent itemsets. The closed itemsets are associated with the generators based on the concept of equivalence class [74].

2.4 Itemset Generation

The itemset is generated starting with the set size k=1 say a_{k-1}. The next itemset is generated using the join operation. The itemset a_{k-1} is joined to produce the next itemset a_k. The generation of itemsets is based on the metrics support threshold. If the support count of the itemset is greater than the threshold level, then the itemset is considered for the generation of itemset in the next level. This process of generating the itemsets continues until no itemset is left.

2.4.1 Different Types of Set Representation used in ARM

A set is a collection of well defined objects. It is usually denoted by capital letters such as A, B,...Z and the elements are denoted by small letters such as a, b,..., z, etc. If an element a belongs to set Z, it is represented as $a \in Z$ [24].

Suppose we are given a database D, each transaction in the database represents an itemset. An itemset $I = \{ i_1, i_2, .., i_n\}$ where each element represents an item $i \in I$ [20]. Set representations that are commonly used in association rule mining can be broadly divided into the following main categories:-

- Closed sets: This is a representation that is based on the concept of formal analysis [78]. This analysis is related to the branch of lattice theory. Its application is applied to the mining of frequent itemsets [63]. In this concept, an itemset I is considered to be closed in the database D if there exists a superset of I that has the same support other than I in D. For example, if the itemset A has no superset, then it means that $cl(A)$ is not frequent. Thus, A cannot be frequent.

- Free set: This representation is based on the δ- *strongrule* notation. The association is in the form of $A \rightarrow a$ such that $A \subseteq I$, a $\in I/\ A$ where δ denotes a natural number. The rule is treated as valid if the Support of(A)-Support(X $\cup$ {a}) $\leq \delta$ [12].

- Disjunction-Free Sets : This representation is a generalization of the free set. The rules are in the form of $A \rightarrow$a $\bigvee$ b, $A \subseteq I$ and a,b $\in I\ A$. The rule is valid if the transaction that contains a also comprises a or b [16].

- Generalized Disjunction-Free Sets: The representation of the rule is in the form of $A \rightarrow a_1 \bigvee ... \bigvee .... \bigvee a_i \bigvee ... \bigvee a_n$. An itemset A is said to be a

$$NDIRep(\mathcal{D}, \sigma) := \{(I, supp(I, \mathcal{D})) \mid supp(I, \mathcal{D}) \geq \sigma, LB(I) \neq UB(I)\}$$

Figure 2-2: The NDI-representation

disjunction-free set if there is no valid rule $A \ \{a_1,...a_i...a_n,\} \rightarrow a_1 \bigvee ... \bigvee a_i \bigvee ... \bigvee a_n$ where $\{a_1,...a_i...a_n,\} \subseteq A$ for any value of $n > 0$ [17].

- Non-Derivable Itemsets: a condensed representation that is based on a set of deduction rules that are derived from the itemset's support. The NDI representation is defined in Figure 2-2. The support can be derived and the decision can be made whether the itemset is frequent or not. An itemset I that is infrequent is not in NDI or $LB(I) \neq UB(I)$. If $LB(I) = UB(I)$, then the support(I) = LB(I) = UB(I) [18].

- Unified View: In this approach, 0-free, non-derivability, and disjunction-freeness were incorporated. A unified view includes the representation which is expressed as a main component of the border and frequent k-free [19].

- NegNodesets: employs bitmap representation for concise itemset representation. The representation collaborates with N-list which is a list-based structure called Hybrid Framework. It has the advantage of both the data structures, i.e., tree and list based for mining the frequent itemsets [45].

2.5 Conclusion

The state-of-the-art survey on the different itemset representation techniques was elaborated. It provides exhaustive detail on the algorithms that generate the itemsets with candidate generation and without candidate generation. The chapter discusses the different types of itemset representation that are used in Association Rule Mining.