www.ijecs.in
International Journal Of Engineering And Computer Science ISSN:2319-7242
Volume 3 Issue 7 July, 2014 Page No. 7193-7198

Mining Frequent Patterns from Uncertain Databases

A. Chandrakanth[1], R. Sateesh Kumar[2], Yella Anitha[3]

[1]Computer Science and Engineering
Vasavi College of Engineering
Hyderabad, INDIA
balram43@gmail.com
[2]Computer Science and Engineering
Vasavi College of Engineering
Hyderabad, INDIA
sateeshramatenki@gmail.com
[3]Computer Science and Engineering
Vasavi College of Engineering
Hyderabad, INDIA
yella.anitha@gmail.com

Abstract: — ***In Many Real time application like sensors monitoring system, location based system and data integration are inexact in nature. It is difficult to extract frequent Item sets from these kind of application because of uncertainty. We study the problem of mining frequent itemsets from uncertain data under Possible Semantic Worlds (PSW). Uncertain database contain exponential large number of possible semantic worlds. We consider transactions whose items are associated with existential probabilities and give a formal definition of frequent patterns from uncertain data model.***

By observing that the mining process can be modelled as a Poisson binomial distribution, an algorithm was developed, which can efficiently and accurately discover frequent item sets in a large uncertain database. We are adopting mining algorithm which identifies Probabilistic Frequent Itemset (PFI) from evolving database. Traditional algorithms for mining frequent itemsets are inapplicable or computationally inefficient for uncertain database. Implementing incremental algorithm which can efficiently accurately discovered frequent item set in large uncertain database.

Keywords: ***PFI, PWS, S-PMF.***

1. Introduction

Nowadays, mass data is generated every minute. Data mining, which is to find useful information and patterns among large databases, has been studied popularly in databases analysis. Data mining which makes to analyse and learn uncertain database for pattern. There are many aspects in data mining field, such as retrieving association rules, classification, clustering, and estimation [2] [3]. In traditional databases, where the data is clear-cut (correct), those approaches have been considered for many years. But, in many applications we may find that the databases are uncertain. The locations of users obtained through RFID and GPS systems, for instance, are not correct due to evaluation errors. Customer purchase behaviours, as captured in supermarket basket databases, contain statistical information for predicting what a customer will buy in the future [6]. Integration and record linkage tools associate confidence values to the output tuples according to the quality of matching. In structured information extractors, confidence values are appended to rules for extracting patterns from unstructured data. Recently, uncertain databases have been proposed to offer a better support for handling imprecise data in these applications, where uncertainty is treated as a first-class citizen".

Databases used in many important applications are often uncertain. For example, the Customer acquisition behaviours, as grab in supermarket basket databases, contain statistical information for predicting what a customer will buy in the future. In structured information extractors, confidence values are appended to rules for extracting patterns. In unstructured information to handle a large amount of uncertain data is recently developed.

It is generally assumed that the items occurring in a transaction are known for certain. However, this is not always the occurrence. For detail;

- In many functions the data is inherently noisy, such as data collected by sensors or in satellite images.
- In privacy protection applications, artificial noise can be added intentionally. Finding patterns despite this noise is a challenging problem.

By aggregating transactions by customer, we can mine patterns across customers instead of transactions. This produces

estimated purchase probabilities per item per customer rather than certain items per transaction.

Considering in Uncertain transaction database all tuple are mutually independent. Thus, the decision by customer A has no influence on customer B. This expectation is reasonable in real world applications[10][15]. This can be justified by the assumption that the items are observed independently. In this case, the probability of a world w is given by

$$P(w) = (\prod_{t \in I}(\prod_{x \in t} P(x \in t) * \prod_{x \notin t}(1 - P(x \in t))) \quad (1)$$

Any query evaluation algorithm for an uncertain database has to be correct under PWS. That is, the results produced by the algorithm should be the same as if the query is evaluated on every possible world. Although PWS is perceptive and useful, querying or mining under this notion is costly. This is because an uncertain database has an exponential number of possible worlds. Performing data mining under PWS can thus be technically challenging. In fact, the mining of uncertain data has recently attracted research attention. For example, in efficient clustering algorithms were developed for uncertain objects; in and naive Bayes and decision tree classifiers designed for uncertain data were studied.

In this paper, incremental mining algorithms is used for finding frequent itemsets evolving certain databases. Our algorithms can be applied to two important uncertainty models: attribute uncertainty and tuple uncertainty, where every tuple is associated with a probability to indicate whether it exists. The frequent item sets discovered from uncertain data are naturally probabilistic, in order to reflect the confidence placed on the mining results for a Probabilistic Frequent Item set (PFI) extracted. A PFI is a set of attribute values that occurs frequently with a sufficiently high probability. The support probability mass function (s-pmf) for the PFI. This is the pmf for the number of tuples (or support count) that contain an item set. Under PWS, a database induces a set of possible worlds, each giving a (different) support count for a given item set. Hence, the support of a frequent item set is described by a pmf.

A simple way of finding PFIs is to mine frequent patterns from every possible world, and then record the probabilities of the occurrences of these patterns. This is impractical, due to the exponential number of possible worlds. To solve this, some algorithms have been recently developed to successfully retrieve PFIs without instantiating all possible worlds[7]. These algorithms can verify whether an item set is a PFI in $O(n^2)$ time (where n is the number of tuples contained in the database).

The s-pmf of a PFI can be captured by a Poisson binomial distribution, for attribute and tuple-uncertain data. The model based algorithm can verify a PFI in $O(n)$ time, and is thus more suitable for large databases. The algorithm can be used to mine threshold-based PFIs, whose probabilities of being true frequent item sets are larger than some user-defined threshold. It takes very less time to find all PFI.

1.1 Mining evolving databases

In real time the important problem is maintaining mining results for changing, or evolving, databases. The type of evolving data that we address here is about the appending, or insertion of a batch of tuples to the database. Tuple insertion is common in the applications that we consider. For example, a GPS system may have to handle location values due to the registration of a new user; in an online marketplace application, information about new purchase transactions may be appended to the database for further analysis. Notice that these new tuples may induce changes to the mining result[12]. A straightforward way of refreshing the mining results is to reevaluate the whole mining algorithm on the new database. This can be costly, however, when new tuples are appended to the database at different time instants. In fact, if the new database D is similar to its older version, D, it is likely that most of the PFIs extracted from D remain valid for D^+. Based on this perception, the developed mining algorithm uses the PFIs of D to derive the PFIs of D^+, instead of finding them from scratch. In this paper, the adopting mining algorithm to study and discovers the PFIs. The algorithm discovers PFIs, which can be extended to handle evolving data. As the experiments show, when the change of the database is small, running the mining algorithm on D^+ is much faster than finding PFIs on D^+ from scratch. In an experiment on a real data set, the mining algorithm addresses a fivefold performance improvement over its non counterpart.

To summarize, an algorithm was adopting, which can reduce the amount of effort of scanning the database for mining threshold− based PFIs. The algorithm can support for attribute and tuple uncertainty models. The comparison has made incremental mining and Apriori approach is presented. Experiments on the real data set reveal that the adopting incremental mining method significantly improves the performance of PFI discovery, with a high degree of efficiency

2. RELATED WORK

Mining frequent item sets is an important problem in data mining, and is also the first step of deriving association rules. Hence, many efficient item set mining algorithm(e.g., Apriori and FP− growth) have been proposed. While these algorithm work well for databases with correct values, it is not clear how they can be used to mine probabilistic data. Here an algorithm for extracting frequent item sets from uncertain databases was developed. Although the algorithm is developed based on the Apriori framework, they can be considered for supporting other algorithm (e.g., FP−growth) for handling uncertain data. For uncertain databases, Aggarwal et al. and Chui et al developed efficient frequent pattern mining algorithm based on the expected support counts of the patterns. However, Bernecker et al. Sun et al. and Yiu et al. found that the use of expected support may render important patterns missing. Hence, they proposed to compute the probability that a pattern is frequent, and introduced the notion of PFI. In dynamic programming based solutions were developed to retrieve PFIs from attribute uncertain databases. However, their algorithm compute probabilities, and verify that an item set is a PFI in $O(n^2)$ time. the algorithm avoid the use of dynamic programming, and are able to verify a PFI much faster in O(n) time. Zhang et al. only considered the extraction of singletons (i.e., sets of single items), the solution discovers patterns with more than one item. Recently, Sun et al. developed an threshold based PFI mining algorithm. However, it does not support attribute uncertain data considered in this paper other works on the retrieval of frequent patterns from imprecise data include, which studied frequent patterns on noisy data; which examined association rules on fuzzy sets; and which proposed the notion of a "vague association rule." However, previous algorithm are not suitable for the uncertainty models. For evolving databases, A few mining algorithm that work for data have been developed. For example, the Fast Update algorithm (FUP) was proposed to efficiently maintain frequent item sets, for a database to which new tuples are inserted. the mining framework is inspired by FUP. The FUP2 algorithm was developed to handle both addition and deletion of tuples. ZIGZAG also examines the efficient maintenance of maximal frequent item sets for

databases that are constantly changing. In a data structure, called CATS Tree, was introduced to maintain frequent item sets in evolving databases. Another structure, called CanTree, arranges tree nodes in an order that is not affected by changes in item frequency[4][6]. The data structure is used to support mining on a changing database. To the best knowledge, maintaining frequent item sets in evolving uncertain databases has not been examined before. The proposed algorithm can also support attribute and tuple uncertainty models

3. PROBLEM DEFINITION

Let V be a set of items. Here, we acquire a database D contains n tuples, or transactions. Each transaction, t_j is associated with a set of items taken from V. Each item $v \in V$ exists in t_j with an existential probability $P(v \in t_j) \in (0,1]$, which denotes the chance that u belongs to t_j . Under Possible Semantic Worlds, D generates a set of possible worlds W. In each world subset of attributes from each transaction. The sum of all possible world is one, and the no of possible worlds is exponentially large[10][5]. The goal is to discover frequent patterns without expanding D into possible worlds. Each transaction is associated with a set of items and an existential probability $P(t_j) \in (0,1]$. Again, the number of possible worlds for this model is exponentially large. The problem of mining approximate probability frequent itemsets and support for every item described in below Table 1 summarizes the symbols used in this paper.

3.1 Support of I

In uncertain transaction database, the support of an item or itemset cannot be unique value, but rather, must be represented by a discrete probability distribution.

DEFININTION 1: In uncertain database T and the set W of possible worlds of T, the support probability P(i) of an itemset X is the probability that X has the support i.

$$P(i) = \sum_{w_j \in W,(S(w_j,I)=i)} P(w_j) \quad (2)$$

Where $S(W_j,I)$ is the support of X in world W_j

Intuitively, P(i) denotes the probability that the support of i. The support probabilities associated with an itemset I for different support values form the support probability distribution of the support of X.

Definition1: The probabilistic support of an itemset X in an uncertain transaction database T is defined by the support probabilities of X (P(X)) for all possible support values $I \in \{0,\ldots.|T|\}$. This probability distribution is called support probability distribution.

3.2 Probability Frequent Item Sets

Let S(wj, I) be the support count of I in possible world wj. Then, the probability that s(I) has a value of I, denoted by P(i),is calculated from Equation (2). Hence, P(i)($i = 1,\ldots,n$) form a probability mass function (pmf) of S(I), where n is the size of database D. Now, let minsup $\in (0, 1]$ be a percentage value, which is generally used to define minimal support in a deterministic database. An item set I is said to be frequent in a database D if $s(I) \geq msc(D)$, where msc (D) = minsup x n is called the minimal support count of D. For uncertain databases the frequentness probability of I, denoted by $P_{freq}(I)$ is the probability that an item set is frequent. Notice that $P_{freq}(I)$ can be expressed as

$$P_{freq}(I) = \sum_{i \geq msc(D)} P(i) \quad (3)$$

Using frequentness probabilities, the proposed algorithm can determine whether an item set I is frequent. I is a Threshold-based PFI if its Frequentness probability is larger then some user-defined threshold. If $P_{freq}(I) \geq$ minprob. We call minprob the frequentness probability threshold. Here the minimum probability (minprob) is the frequentness probability threshold.

Notation	Description
D	*An uncertain database of n tuples*
V	*The set of items that appear in D, d,D +*
v	*An item, where v ∈ V*
t_j	*The jth tuple in D*
W	*The set of all possible worlds.*
W_j	*A possible world wj ∈ W*
I	*An item set, where I ⊆ V*
$minsup$	*A real value between (0; 1]*
$msc(D)$	*The minimal support count in D*
$s(I)$	*The support count of I in D*
$minprob$	*A real value between (0,1],used in threshold − based PFI*
$P^I(i)$	*Support probability (i.e. probability that I has a support count of i)*
$P_{freq}(I)$	*Frequentness probability of I*
P_j^I	*Pr(I ⊆ t_j)*
μ^I	*Expected value of XI in D*

Table 1
Summary of Notation

4. Estimating S-PMF

In this section we estimate the s-pmf s(I) of item set I plays an important role in determining whether I is a PFI. However, directly computing s(I) can be expansive. An interesting observation about s(I) is that it is essentially the number of successful Poisson trials[13]. To explain, let X_j^I be a random variable, which is equal to one if I is a subset of the items associated with transaction t_j, or zero otherwise. Given a database of size n, each I is related with random variables $X_1^I, X_2^I, \ldots, X_n^I$,all tuples are discrete. Therefore, these n variables are discrete, and they represent n Poisson trials. Moreover, $X^I = \sum_{j=1}^{n} X_j^I$ follow a Poisson binomial distribution. Next we evaluate relationship between X^I and $P^I(i)$, $P^I(i) = P(X^I = i)$.This is simply because X^I is the number of items that I exists in the database. Hence, the s-pmf of I, I.e.,

$P^I(i)$ is the pmf of X^I, a Poisson binomial distribution. Using (5) we can write (2), which computes the frequentness probability of I,

$P_{freq}(I) = \sum_{i \geq msc(D)} \Pr(X^I = i)$

$= P(X^I \geq msc(D))$ (4)

Therefore, if the cumulative distribution function (cdf) of X^I is known, $P_{freq}(I)$ can also be evaluated. Next, another approach to this cdf, in order to compute $P_{freq}(I)$ efficiently.

A. Approximating s-pmf

we can express $P_{freq}(I)$ as

$P_{freq}(I) = 1- P(X^I \geq msc(D) -1)$, (5)

For notational convenience, let P^I_j be $P(I \subseteq t_j)$. Then, the expected value of X^I. In D, denoted by μ^I, can be computed by

$\mu^I = \sum_{j=1}^{n} p_j^I$ (6)

Since a Poisson binomial distribution can be well approximated by a Poisson distribution [8], (5) can be written as

$P_{freq}(I) \approx 1- F(msc(D)-1, \mu^I)$, (7)

Where F is the cdf of the Poisson distribution can be well approximated by a Poisson distribution with mean μ^I , i.e.,

$F(msc(D)-1, \mu^I) = 1- \frac{\tau(msc(D),\mu^I)}{msc(D)-1}$, (8)

expressed using the incomplete gamma function $\Gamma_{(s,x)} = \int_x^{\infty} t^{s-1} e^{-t} dt.$

To estimate the $P_{freq}(I)$, we can compute μ^I by scanning D once and summing up P^I_j 's for all tuples t_j in D. Then, $F(msc(D)-1, \mu^I)$, is evaluated and approximate is estimated from equation .

Theorem 1: P_{freq} (I), if approximated by (7), increase monotonically with μ^I.

Proof: The cdf of a Poisson distribution, *F(I,μ)* can be written as

$$F(i,\mu) = \frac{\Gamma(i+1,\mu)}{i!} = \frac{\int_{\mu}^{\infty} t^{(i+1)-1} e^{-1}\, dt}{i!}$$

Since minsup is fixed and independent of μ, let us examine the partial derivate w.r.t. μ

$$= \frac{\partial F(i,\mu)}{\delta \mu} = \frac{\partial}{\partial \mu}\left(\frac{\int_{\mu}^{\infty} t(i+1)-1 e^{-t}}{i!}\right)$$

$$= \frac{1}{i!}\frac{\partial}{\partial \mu}\left(\int_{\mu}^{\infty} t^i e^{-t}\, dt\right)$$

$$= \frac{1}{i!}(-\mu^i e^{-\mu})$$

$$= -f(i,\mu) \leq 0.$$

Thus, the cdf of the passion distribution F(i,μ) is monotonically decreasing w.r.t. μ when I is fixed.

Consequently, 1-F(i-1,μ)increase monotonically with μ. Theorem1follow immediately by substituting i, i=msc(D).From above theorem states that the higher value of μ^I, the higher is the chance that *I* is a PFI.

5. THRESHOLD-BASED PFI MINING

In an uncertain database, it is difficult quickly determine whether an itemset I is a threshold-based PFI? Answering this question is critical part. We develop a simple method of testing whether I is a threshold-based PFI, without computing its frequentness probability.

5.1 PFI Testing

Given the values of minsup and minprob, we can test whether I is a threshold-based PFI, in three steps:

Step1. Find a real number μ_m satisfying the equation: $minprob = 1 - F(msc(D), 1; \mu_m).$ From Equation (9) can be solved efficiently by employing numerical methods.

Step2. Use Equation (6) to compute μ^I. Notice that the database D has to be scanned once.

Step3. If $\mu^I \geq \mu_m$, we conclude that I is a PFI. Otherwise, I must not be a PFI.

To understand why this works, first notice that the right hand side of Equation(9) is the same as that of Equation (7), an expression of frequentness probability. Essentially, Step 1 finds out the value of μ_m that corresponds to the frequentness probability threshold (i.e., minprob). In Steps 2 and 3, if $\mu^I \geq \mu_m$,Theorem 1 allows us to deduce that $P_{freq}(I)$ > minprob. Hence, these steps together can test whether an itemset is a PFI.

In order to verify whether I is a PFI, once μ_m is found, we do not have to evaluate $P_{freq}(I)$. Instead, we compute μ^I in Step 2, which can be done in O(n) time. This is a more scalable method compared with solutions in, which evaluate P_{freq} in $O(n^2)$time. Next, we study how this method can be further improved.

5.2 Improving the PFI Testing Process

Database D has to be scanned once to obtain μ^I , for every item set I. This can be costly if D is large, and if many item sets need to be tested. For example, in the Apriori algorithm [6], many candidate item sets are generated first before testing whether they are PFIs. Next the process of how the PFI testing can still be carried out without scanning the whole database was explained.

Let $\mu^I = \sum_{j=1}^{l} p_j$, where $l \in (0, n]$., Essentially, μ^I is the "partial value" of μ^I ,which is obtained after scanning 1 tuples. Notice that $\mu^I{}_j = \mu^I$. Suppose that μ_m has been obtained from (11).

Corollary 1. An item set I cannot be a PFI if there exist

$$i \in (0, [\mu m]]$$

such that

$$\mu^I_{n-i} < \mu_m - i$$

.

The above equations can be used to improve the speed of the PFI testing process[11]. Specifically, after a tuple has been scanned, The algorithm checks whether the s-pmf value of the item set exceeds the threshold value; if so, it immediately conclude that I is a PFI. After scanning n- μ_m or more tuples, we examine whether I is not a PFI, by using

Corollary 1: These testing procedures continue until the whole database is scanned, yielding μ^I, Then, the algorithm execute Step 3 (Section A) to test whether I is a PFI

6. Incremental Mining Framework

Now we present evolving database D contains n tuples. D is also called old database. A small uncertain database d, which contains n' tuples, will be appended to D. d is called delta database. Let new database be $D^+ = D + d$, which contains $n^+ = n + n'$ tuples. Our goal is to maintain all threshold-based PFIs of D^+, under the same values of minsup or minprob. Notice that the minimal support count changes with the size of database[10]. We assume that the complete set of PFIs and some essential support information (e.g. the s-pmfs, or the expected support values) of them from the old database are saved on disk. We using an incremental mining algorithm for maintaining frequent itemsets in evolving databases with improper data, called uFUP. Based on the framework, we

maintain uncertain database with help of algorithms called uFUP respectively. Note that the algorithms support both attribute and tuple uncertainty models.

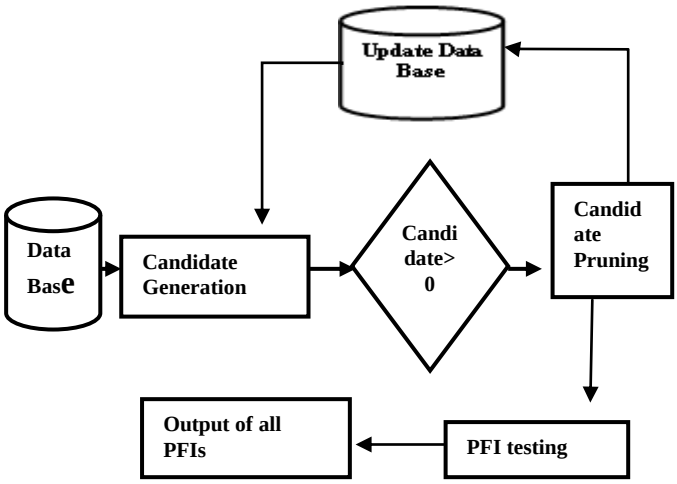

Fig 1: System Architecture

Our incremental mining framework (Figure -1) is a candidate set generation and test" approach which is based on the Apriori. It is a bottom-up framework, which generates k-PFIs with ($k-1$) PFIs. The algorithm contains several iterations. Starting from k = 1, there are three steps in each iteration.

Step1: Candidate Generation, Size-k candidates are generated in this step. The algorithm will be stopped when there are no new candidates. In the first iteration, the generation is based on the old PFIs and the delta database. When k > 1, the generation is based on (K-1)PFIs.

Step2: Candidate Pruning, A pruning method is used to reduce the size of candidates by scanning the delta database. The target is to obtain a small set of candidates.

Step3: PFI Testing, All remaining candidates are tested to get the k-PFIs. We have mentioned that re-evaluating mining algorithms on the new database directly is inefficient, since it would calculate many results which have been calculated before. And, it costs much time to scan the new database when testing PFI. Notice that in step 1 and 2, only d and the PFIs of Dare needed. Since these pieces of information are relatively small in size (compared with D or D^+), they are usually not very expensive to evaluate. step 3 involves deriving the s−pmf s of item sets, with the use of D^+ , and is thus more expensive than other phases. If step 2 successfully removes a lot of candidates from consideration, the cost of executing step 3 can be reduced. The above discussion is formalized in the Algorithm-1, which uses the databases D and d, as well as the set of PFIs F^D collected from D (e.g., using the method of [6]). The output of the algorithm is a set F^+ of PFIs for D^+, $F_k^+ = F_1^+, F_2^+, F_3^+, \ldots .. F_n^+$ and F^+ is the set of „k− PFIs" for D^+

Let C_k^+ be a set of size− k candidates found from D^+. Initially k=1. The algorithm generates C_+^1 (Step 1). In the kth iteration, first the algorithm removes candidate item sets that cannot be k− PFIs, from C_k^+ (Step2).if C_k^+ is not empty, the algorithm performs testing on these candidates, in order to find out the true k− PFIs (i.e., F_k^+,) Step 3 generates size (k+1)−candidate item sets by using the k− PFIs. The whole process is repeated until no more candidates are found, The algorithm takes D, d, minsup, minprob as inputs and produces the output as a set of Probabilistic Frequent item sets (PFIs) in the function F, which is given as follows

Algorithm uFUP

uFUP runs several iterations. Each iteration contains three steps. Starting from k = 1, Candidate Generation generates C_k^+ which is size-k candidates of D^+ (Lines 3 { 4 and 12). The algorithm is stopped when $|C_k^+|$ = 0. Otherwise, Candidate Pruning is adopted to prune candidates (Line 6). Finally, k-PFIs (F_k^+) are retrieved from C^+_k by PFI Testing (Line 8).We can see that the old PFIs F^D and the delta database d are needed in every step. However, only PFI testing step requires scanning the old database D.

Algorithm uFUP

Input: D, d, F^D, minsup, minprob

Output: Exact PFIs of D^+: $F^+ = \{F_1^+, F_2^+, \ldots\ldots, F_m^+\}$ // F_k^+ is set of k-PFIs

1.begin
2. $F^+ = 0$;
3. $C_1^+.GenerateSingleton(d, F_1^D)$;
4. $k = 1$; while$|C_k^1| \neq 0$ do;
5. $C_k^+.Pruned(d, F_k^D, minsup)$;
6. $if|C_k^+| \neq 0$ then
7. $F_k^+ \leftarrow C_k^+.Test(D, d, F_k^D, minsup, minprob)$;
8. $else$
9. Break;
10. $C_{k+1}^+.GenerateCandidate(F_k^+)$;
11. $k = k + 1$;
12. $return\ F^+ = \{F_1^+, F_2^+, \ldots\ldots, F_{k-1}^+\}$
13. end

7. RESULT

Now the experimental results on the data set, called road safety accidents. This data set is obtained from http://data.gov.uk/. Provided by coroners in England and Wales and procurators Fiscal in Scotland for the period of 1/1/1979 - 31/12/2013. The data are obtained from the "Great Britain (England, Scotland, Wales)," which are filled out by a police officer for each traffic with STATS 19 form accident occurring on a public road in Great Britain . The data set contains 1,40,184 accident records, with a total of 457attribute values. On average, each record has 40 attributes. The algorithm uses the first 10k tuples as the default data set as its input. The default value of minsup is 20 percent. To test the mining algorithm, it uses the first 10k tuples as the old data base D, and the subsequent tuples as the delta database d. The default size of d is 5 percent of D, it considers both attribute and tuple uncertainty models. The default value of minprob is 0.4.In the results presented, minsup is shown as a percentage of the data set size n. Notice that when the values of minsup or minprob are large, no PFIs can be returned; it do not show the results for these values. The experiments were carried out on the Windows XP operating system, on a machine with a 2.66GHz Intel Core 2 Duo processor and 4 GB memory. The programs were written in Java, for data set is maintained by Wamp Server written queries operation to get probability values, MySQL java connector used for connections and compiled in Net beans. The proposed algorithm is implemented under the java runtime environment. The algorithm extracts the probabilistic frequent item sets within a fraction of seconds

Comparison on uFUP vs. Apriori:

Next compare uFup and Apriori which both yield PFIs. The given Figure 2 shows that uFUP is faster than Apriori over different minsup values. minsup is taken on the X-axis and runtime is taken on the Y-axis. On increasing the minsup value the runtime decreases.

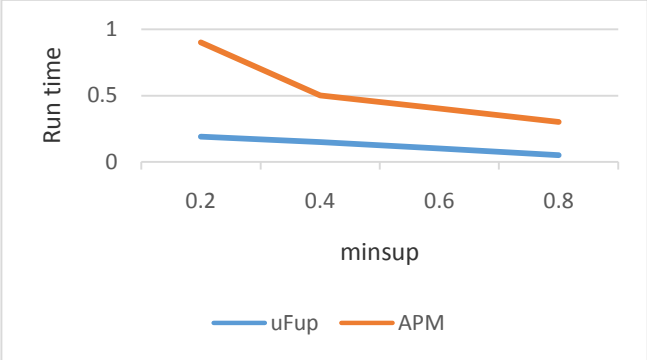

Figure 2. Comparison On Ufup Vs. Apriori(minsup)

8. CONCLUSION AND FUTURE WORK

In this paper, we have studied the problem of mining frequent itemsets from uncertain databases. We adopting an algorithm to retrieve PFIs using model based approach for incremental uncertain databases. Model-based method calculates the s-pmf of an itemsets, in order to find PFIs. This method can efficiently extract threshold-based PFIs. It also supports attribute and tuple uncertainty. We used approximate methods to evaluate frequentness probabilities efficiently. Then, we adopting an incremental mining framework for retrieving threshold based PFIs from incremental uncertain databases. Based on this framework, we proposed an exact incremental mining algorithm, which identifies the complete set of PFIs when new tuples are being inserted into the databases. The experimental results show that our incremental mining algorithms are efficient and effective.

8.1 FUTURE WORK

We will also study the other method to model the support probability mass function, such as normal distribution, which is a common distribution. On the other hand, sampling can also be used to retrieve probabilistic frequent itemsets. The main challenge of the sampling method is how to balance the efficiency and accuracy, and how to give an appropriate theoretical error found.

References

[1] Chin-Chen Chang, Yu-Chiang Li, Jung-San Lee,Taichung,"An Efficient Algorithm for Incremental Mining of Association Rules," Proc.15th International Workshop on Research Issues in Data Engineering: Stream Data Mining and Applications (RIDE-SDMA'05) 1097-8585/05 $20.00 © 2005 IEEE.

[2] Thomas Bernecker, Hans-Peter Kriegel, Matthias Renz, Florian Verhein, Andreas Zuefle, "Probabilistic Frequent Itemset Mining in Uncertain Databases," Proc. 15th ACM SIGKDD Conf. on Knowledge Discovery and Data Mining (KDD'09), Paris, France, 2009.

[3] Yongxin Tong, Lei Chen, Yurong Cheng, Philip S. Yu,"Mining Frequent Itemsets over Uncertain Databases," Proceedings of the VLDB Endowment, Vol. 5, No. 11Copyright 2012 VLDB Endowment 21508097/12/07, August 27th 31st 2012, Istanbul, Turkey.

[4] Mr. Avinash, A. Powar, A.S. Tamboli, "Incremental Mining for Frequent Item set on Large Uncertain Databases," International Journal of Advanced Research in Computer Engineering & Technology (IJARCET) Volume 2, Issue 11, November 2013.

[5] Liwen Sun, Reynold Cheng, David W. Cheung, Jiefeng Cheng, "Mining Uncertain Data with Probabilistic Guarantees," KDD'10, July 25–28, 2010, Washington, DC, USA.

[6] Liwen Sun, Reynold Cheng, David W. Cheung, Jiefeng Cheng, "Mining Uncertain Data with Probabilistic Guarantees," KDD'10, July 25–28, 2010, Washington, DC, USA.

[7] Rakesh Agrawal, Tomasz Imielinski, Arun Swami, "Mining Association Rules between Sets of Items in Large Databases," Proceedings of the 1993 ACM SIGMOD Conference Washington DC, USA, May 1993.

[8] Qin Zhang, Feifei Li2, Ke Yi, "Finding Frequent Items in Probabilistic Data,"*SIGMOD'08,* June 9–12, 2008, Vancouver, BC, Canada.

[9] H. Cheng, P. Yu, and J. Han, "Approximate Frequent Itemset Mining in the Presence of Random Noise," Proc. Soft Computing for Knowledge Discovery and Data Mining, pp. 363-389, 2008.

[10] D. Cheung, J. Han, V. Ng, and C. Wong, "Maintenance of Discovered Association Rules in Large Databases: An Incremental Updating Technique," Proc. 12th Int'l Conf. Data Eng. (ICDE), 1996.

[11] N. Dalvi and D. Suciu, "Efficient Query Evaluation on Probabilistic Databases," Proc. 13th Int'l Conf. Very Large Data Bases (VLDB), 2004.

[12] A. Deshpande, C. Guestrin, S. Madden, J. Hellerstein, and W. Hong, "Model-Driven Data Acquisition in Sensor Networks,"Proc. 13th Int'l Conf. Very Large Data Bases (VLDB), 2004.

[13] J. Huang, "MayBMS: A Probabilistic Database Management System," Proc. 35th ACM SIGMOD Int'l Conf. Management of Data, 2009.

[14] J. Ren, S.D. Lee, X. Chen, B. Kao, R. Cheng, and D.W. Cheung, "Naive Bayes Classification of Uncertain Data," Proc. IEEE NinthInt'l Conf. Data Mining (ICDM), 2009.

[15] N. Khoussainova, M. Balazinska, and D. Suciu, "Towards Correcting Input Data Errors Probabilistically Using Integrity Constraints," Proc. Fifth ACM Int'l Workshop Data Eng. for Wireless and Mobile Access (MobiDE), 2006.

[16] H. Kriegel and M. Pfeifle, "Density-Based Clustering of Uncertain Data," Proc. ACM SIGKDD Int'l Conf. Knowledge Discovery in Data Mining (KDD), 2005.