

Estimating Joint Probabilities from Marginal Ones^{*}

Tao Li, Shenghuo Zhu, Mitsunori Ogihara, and Yinhe Cheng

Computer Science Department
University of Rochester
Rochester, New York 14627-0226
{taoli,zsh,ogihara,cheng}@cs.rochester.edu

Abstract. Estimating joint probabilities plays an important role in many data mining and machine learning tasks. In this paper we introduce two methods, *minAB* and *prodAB*, to estimate joint probabilities. Both methods are based on a light-weight structure, *partition support*. The core idea is to maintain the partition support of itemsets over logically disjoint partitions and then use it to estimate joint probabilities of itemsets of higher cardinalities. We present extensive mathematical analyses on both methods and compare their performances on synthetic datasets. We also demonstrate a case study of using the estimation methods in *Apriori* algorithm for fast association mining. Moreover, we explore the usefulness of the estimation methods in other mining/learning tasks [9]. Experimental results show the effectiveness of the estimation methods.

Keywords: *Joint Probability, Estimation, Association Mining*

1 Introduction

Estimating the joint probabilities in a collection of N observations on M events is the problem of estimating the joint probabilities of events, given the probabilities of single events. Generally the collection of N observations on M events is represented by a $N \times M$ binary table D where $D_{ij} = 1$ denotes that the i -th event occurs in the j -th observation and $D_{ij} = 0$ otherwise. Let $\{\mathcal{I}_j\}$, $j = 1, \dots, M$, represent the events. $P(\mathcal{I}_j)$ can be estimated by computing its occurrence frequency in the table. Thus given $P(\mathcal{I}_j)$, $j = 1, \dots, M$, the goal is to estimate the joint probability $P(\mathcal{I}_{j_1}, \dots, \mathcal{I}_{j_l})$, $l \geq 2, 1 \leq j_1, \dots, j_l \leq M$.

A simple way to estimate joint probabilities is, like we approximate the probabilities of a single event, to just count the co-occurrences of the events in the table D (i.e., via combinatory counting). Although in many cases this simple method does provide satisfactory solutions, there are cases in which the time

^{*} The project is supported in part by NIH Grants 5-P41-RR09283, RO1-AG18231, and P30-AG18254 and by NSF Grants EIA-0080124, NSF CCR-9701911, and DUE-9980943. We would also like to thank Dr. Meng Xiang Tang and Xianghui Liu for their helpful discussions.

or space complexity of combinatory counting is very large even unacceptable. For example, in a large dataset which cannot fit in the main memory, combinatory counting would incur considerable overheads. Even in cases in which the complexity of combinatory counting is manageable, there can also be reasons to consider the estimation methods. First, the given dataset can be viewed as a sample of some source distribution. So, even the exact counting just provides an approximation to the source distribution. On the other hand, in many application domains, the goals are finding the interesting patterns which satisfies some given thresholds to support the decision processes. Most of those thresholds are given by estimations or specified manually. Hence, the combinatory counting may not be worth the computation cost in these cases.

Since $P(A|B) = \frac{P(A,B)}{P(B)}$, knowing the joint probabilities is useful to infer the intrinsic relations between events such as associations [1], correlations [3], causalities [10] and multidimensional patterns [7]. Hence, it plays an important role in many data mining tasks. For example, frequent itemset mining in the discovery of association rules [1,4] can be thought as finding the set of items whose joint probabilities satisfy the given parameters. In the rest of the paper, we use D to denote the given $N \times M$ dataset, and $I = \mathcal{I}_1, \mathcal{I}_2, \dots, \mathcal{I}_M$ be a set of events¹. Each row, R_i , in the dataset, D , is referred as an observation (a record). An itemset with k items is called k -itemset.

In this paper we present two methods, *minAB* and *prodAB*, to estimate the joint probabilities without combinatory counting and explores their applications in data mining. Both methods are based on a structure called *partition support*. The main idea is to maintain the partition support information of items (or itemsets) over each logically disjoint partition and then use it to carry out the estimation. The rest of the paper is organized as follows: Section 2 introduces the basic concepts of *partition support*. Section 3 describes the *minAB* estimation methods and analyzes its properties. Section 4 presents *prodAB* estimation methods. Section 5 shows the performances of the two estimation methods on synthetic datasets. Section 6 gives a case study on using the estimation methods for fast association mining. Section 7 concludes and proposes our future work.

2 Partition Support

Definition 1. *The support count $\lambda(S, D)$ of itemset S in dataset D is the number of records in D containing S . If we logically divided the datasets D into disjoint partition $D_1, \dots, D_n, n \geq 1$, the **partition support** $PS(S, D, n)$ of an itemset S over D is a n -tuple $(\lambda(S, D_1), \lambda(S, D_2), \dots, \lambda(S, D_n))$.*

The support count of itemset S in D , $|\lambda(S, D)|$, is the sum of all the elements in $PS(S, D, n)$. The partition support $PS(S, D, n)$ is a structure consisting of the support counts of S in each partition. Figure 2 shows an example of the dataset. It has five items (A, B, C, E, F) and six records (1, 2, 3, 4, 5, 6). Clearly,

¹ items or attributes.

	A	B	C	E	F
1	0	1	1	1	1
2	1	0	0	1	1
3	1	1	0	1	1
4	0	1	1	1	1
5	1	1	1	0	0
6	0	1	1	0	1

Fig. 1. An Example Dataset

$$\begin{aligned}
 PS(\{A\}, D, 3) &= (\lambda(\{A\}, D_1), \lambda(\{A\}, D_2), \lambda(\{A\}, D_3)) = (1, 1, 1), \\
 PS(\{B\}, D, 3) &= (\lambda(\{B\}, D_1), \lambda(\{B\}, D_2), \lambda(\{B\}, D_3)) = (1, 2, 2), \\
 PS(\{A, B\}, D, 3) &= (\lambda(\{A, B\}, D_1), \lambda(\{A, B\}, D_2), \lambda(\{A, B\}, D_3)) \\
 &= (0, 1, 1), \\
 |\lambda(\{A, B\}, D)| &= 2; \\
 PS(\{B, C\}, D, 3) &= (1, 2, 2), \quad |\lambda(\{A, B, C\}, D)| = 2.
 \end{aligned}$$

Fig. 2. The Partition Support

$\lambda(\{A\}, D) = 3, \lambda(\{B\}, D) = 5, \lambda(\{A, B\}, D) = 2$. If we divide D into $D_1 = \{1, 2\}, D_2 = \{3, 4\}, D_3 = \{5, 6\}$, then the partition support is given in Fig. 2.

3 *minAB* Estimation

3.1 Method Description

Note that in the above example,

$$|\lambda(\{A, B, C\}, D)| \leq \sum_{i=1}^3 \min\{\lambda(\{A\}, D_i), \lambda(\{B\}, D_i), \lambda(\{C\}, D_i)\} = 3, \quad (1)$$

where $\min(x, y)$ returns the smaller value of x and y . In general, given the partition support $PS(\{\mathcal{I}_i\}, D)$ of any singleton itemset $\{\mathcal{I}_i\}, i = 1, \dots, M$, we can get an upper estimation of the support count of any itemset $S = \{\mathcal{I}_{i_1}, \mathcal{I}_{i_2}, \dots, \mathcal{I}_{i_k}\}$ since $\lambda(S, D) \leq \sum_i \min\{\lambda(\{\mathcal{I}_{i_1}\}, D_i), \dots, \lambda(\{\mathcal{I}_{i_k}\}, D_i)\}$. In other words, knowing the approximate distribution of each singleton item over the partitions, we can estimate the joint probabilities. Clearly we can also maintain the partition support of size $k, k \geq 1$, itemsets and use it to get an upper estimation of the support counts of itemsets with higher cardinalities. For example, if we have the support count of any 2-itemset, we can use it to obtain a tighter upper bound of the itemset $S : \sum_i \min\{\lambda(S_1^{(2)}, D_i), \dots, \lambda(S_t^{(2)}, D_i)\}$, where $S_j^{(2)}, j = 1, \dots, t$, are all the 2-subsets of S . For previous example, using the partition support of singleton sets, we have Eq. (1). We get a tighter estimate Eq. (2) by using the partition support of 2-sets.

$$|\lambda(\{A, B, C\}, D)| \leq \sum_{i=1}^3 \min\{\lambda(\{A, B\}, D_i), \lambda(\{A, C\}, D_i), \lambda(\{B, C\}, D_i)\} = 2. \quad (2)$$

The *minAB* estimation method can be described as follows: divide the dataset into logical partitions, maintain the partition support of itemsets and then use summation of the minimums of the partition support of itemsets to estimate the joint probability of itemsets with higher cardinalities. Note that the space complexity for partition support is increased exponentially with respect to the size of the itemsets (and the time complexity is increased too). Hence in most applications, we only maintain the partition support for singleton sets.

3.2 Method Analysis

In this section, we analyze the relationship between the number of partitions and the accuracy of the estimate and try to come up with the methods to determine the approximate number of partitions k . We model their relationship in two different ways. One is to establish the expression of the probability of correct estimate in terms of number of partitions and the other is to establish the relationship between the estimation error and the number of partitions.

Probability of Correct Estimation. We first analyze the case of using the partition support of singleton sets to estimate the joint probability of 2-sets. The problem can be described as follows: given the dataset D with N records, consider two boolean attributes a and b , i.e., a and b are either 0 or 1. Let the number of records in D such that $a = 1$ be $\lambda(\{a\}, D) = A$, i.e., $P(a = 1) \approx \frac{A}{N}$, the number of records with $b = 1$ be $\lambda(\{b\}, D) = B$, i.e., $P(b = 1) \approx \frac{B}{N}$, then we use $\min(A, B)$ to estimate $\lambda(\{a, b\}, D)$, the number of records with both $a = 1$ and $b = 1$. In other words, we use $\min(A, B)$ to estimate $P(a = 1, b = 1) \leq \frac{\min(A, B)}{N}$. The following lemma (proved in [9]) gives the probability of correct estimate.

Lemma 1. *Suppose that A, B subject to the binomial probability $\text{Binomial}(N, p_a)$ and $\text{Binomial}(N, p_b)$, respectively, and $P(a = 1, b = 1) = q$. Let P_N be the probability of the correct estimation, i.e., $\min(A, B) = \lambda(\{a, b\}, D)$. We have,*

$$\begin{aligned} P_N &\equiv P(\min(A, B) = \lambda(\{a, b\}, D) | N) \\ &= (1 + q - p_a)^N + (1 + q - p_b)^N - (1 + 2q - p_a - p_b)^N. \end{aligned} \quad (3)$$

In particular, $P_1 = 1$. Observe that the above lemma can be easily extend to more attributes. If we divide the dataset into two disjoint partitions and assume the partitions are independent, then the probability of correct estimation is $P_{N/2} \times P_{N/2}$.

Lemma 2. $P_{N/2} \times P_{N/2} > P_N$.

If we divide the database into k partitions and assume the independence among the partitions, the probability of the correct estimation of the $\min AB$ method is $P_{N/k}^k$. In general, we have

Proposition 1. $P_{N/(k+1)}^{k+1} > P_{N/k}^k, k \geq 1$.

The above proposition (proved in [9]) illustrates that the more the number of partitions, the more accurate the estimation. In particular, if $k = N$, i.e., if we view each record as a partition, then we always get the correct estimation. However, it would cost more time and space for larger k . Hence, from practical perspective, we need choose the proper k to reduce the computation cost while ensure the enough accuracy. Figure 3(a) and 3(b) depict the probabilities of correct estimate for $N = 10000, p_a = p_b = 0.005, q = 0.0045$, and $N = 100000, p_a = p_b = 0.001, q = 0.0009$, respectively, on different partitions.

So given two itemsets S_1 and S_2 , suppose that $\lambda(S_1, D)$ and $\lambda(S_2, D)$ satisfy the conditions of Lemma 1, then when we use the partition support of S_1 and S_2 to estimate the actual support count of $S_1 \cup S_2$, the probability of correct estimation is $P_{N/k}^k$. Therefore we can choose the number of partitions based on the probability of the correct estimation: given a pre-defined parameter σ , we want the probability of correct estimation to be no less than σ , i.e., $P_{N/k}^k \geq \sigma$. To compute $P_{N/k}^k$, we use Lemma 1. p_a and p_b can be approximated by the number of occurrences of the respective attributes, and let q be some given parameter (say, the minimum support of the association rules in Section 6). Hence the above inequality gives us a principle to choose k .

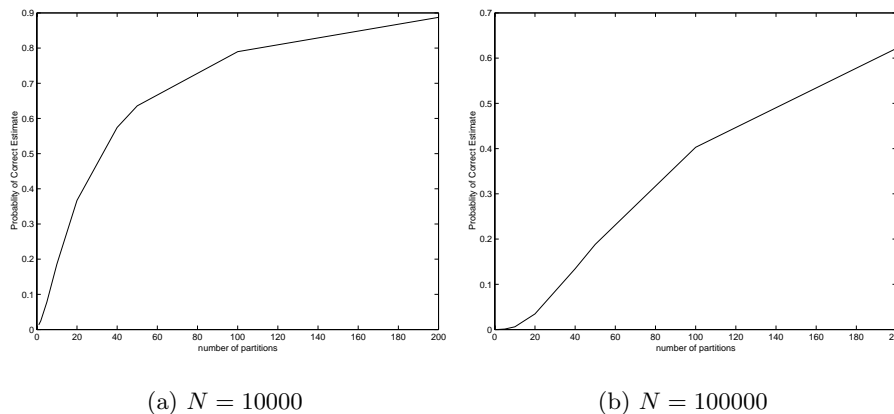


Fig. 3. Probability of the Correct Estimation

Estimation Error. Another way to model the relation between the number of partitions and the estimate is to compute the expectation of the estimation error between $\min(A, B)$ and $\lambda(\{a, b\}, D)$. It's not difficult to get the distribution function for $\min(A, B)$, where A is $\text{Binomial}(N, p_a)$, B is $\text{Binomial}(N, p_b)$, $P(a = 1, b = 1) = q$. Denote $X = \min(A, B)$, then X assumes discrete value $0, 1, \dots, N$. We could directly calculate the probability of $P(X = t)$, where $t = 0, 1, \dots, N$. More details of the computation can be found in [9]. The expectation of $\min(A, B)$ is $E(\min(A, B)) = \sum_{t=0}^N tP(X = t)$ and the expectation of the number of transactions with both $a = 1$ and $b = 1$ is $E_{ab} = Nq$. Hence we get the mean value of the estimate error $e(N) = |E(\min(A, B)) - E_{ab}|$. If we divide the database into two partitions, the estimation error is then $2 \times e(N/2)$. In general the estimation error of the $\min AB$ method is $k \times e(\frac{N}{k})$ if we have $k \geq 1$ partitions. The estimation error also gives us another heuristic to pick the number of database partitions: given a pre-defined error bound γ , we want the estimation error to be smaller than γ . We can also derive the *probability error* of the $\min AB$ estimation method by $\frac{k \times e(\frac{N}{k})}{N}$.

4 *prodAB* Estimation

The *prodAB* estimation is also based on partition support. Suppose there are N data points, which are randomly divided into k partitions and each partition contains $n = N/k$ points. Let $X_{ij} = 1$ be the event that the attribute a of the j -th point in i -th partition is active, $X_{ij} = 0$ otherwise. Let $Y_{ij} = 1$ be the event that the attribute b of the j -th point in i -th partition is active, $Y_{ij} = 0$ otherwise. X_{ij} 's are *i.i.d.*(independent identically distributed) and obey zero-one distribution with probability p_a . Y_{ij} 's are *i.i.d.* and obey zero-one distribution with p_b . X_{ij} and $Y_{i'j'}$ are independent if $(i, j) \neq (i', j')$. $P(X_{ij} = 1, Y_{ij} = 1) = q$. Of course, $q \leq p_a$ and $q \leq p_b$. Knowing $A_i = \sum_{j=1}^n X_{ij}$ and $B_i = \sum_{j=1}^n Y_{ij}$, which is just the *partition support* of $\{a\}$ and $\{b\}$, the goal is to estimate q .

$$EA_i = E \sum_j X_{ij} = \sum_j EX_{ij} = np_a, \quad EB_i = np_b \quad (4)$$

$$\begin{aligned} EA_i B_i &= E \left(\sum_j X_{ij} \sum_j Y_{ij} \right) = E \left(\sum_j X_{ij} Y_{ij} + \sum_{j \neq j'} X_{ij} Y_{ij'} \right) \\ &= \sum_j EX_{ij} Y_{ij} + \sum_{j \neq j'} EX_{ij} EY_{ij'} = nq + n(n-1)p_a p_b \end{aligned} \quad (5)$$

$$\begin{aligned} E \left(\sum_i A_i \sum_i B_i \right) &= E \left(\sum_{ij} X_{ij} \sum_{ij} Y_{ij} \right) = E \left(\sum_{ij} X_{ij} Y_{ij} + \sum_{(i,j) \neq (i',j')} X_{ij} Y_{i'j'} \right) \\ &= \sum_{ij} E(X_{ij} Y_{ij}) + \sum_{(i,j) \neq (i',j')} EX_{ij} EY_{i'j'} = nkq + nk(nk-1)p_a p_b \end{aligned} \quad (6)$$

Inserting Eq. (4) into Eq. (5), Eq. (7) can be derived.

$$q = \frac{nEA_i B_i - (n-1)EA_i EB_i}{n^2} \quad (7)$$

If we estimate $E(A_i B_i)$ with $\frac{1}{k} \sum_i A_i B_i$, EA_i with $\frac{1}{k} \sum_i A_i$ and EB_i with $\frac{1}{k} \sum_i B_i$, we can estimate q by

$$\hat{q} = \frac{nk \sum_i A_i B_i - (n-1) \sum_i A_i \sum_i B_i}{n^2 k^2} \quad (8)$$

In the case $n = 1$, $\hat{q} = \frac{1}{k} \sum_i A_i B_i$ is the exact rate of the points whose attributes \mathcal{A} and \mathcal{B} are active. In the case $k = 1$, $\hat{q} = \frac{1}{n^2} \sum_i A_i B_i$ is the rate of the points whose attribute \mathcal{A} is active times the rate of the points whose attribute \mathcal{B} is active, i.e., it is the estimation by assuming attributes \mathcal{A} and attribute \mathcal{B} are independent.

Unfortunately, Eq. (8) is a biased estimation, i.e., $E\hat{q} \neq q$. Now, replacing $p_a p_b$ in Eq. (5) with the one in Eq. (6), we have

$$q = \frac{k(nk-1)E(A_i B_i) - (n-1)E(\sum_i A_i \sum_i B_i)}{n^2 k(k-1)} \quad (9)$$

If we estimate $E(A_i B_i)$ with $\frac{1}{k} \sum_i A_i B_i$ and $E(\sum_i A_i \sum_i B_i)$ with $\sum_i A_i \sum_i B_i$, we estimate q by

$$\hat{q} = \frac{(nk - 1) \sum_i A_i B_i - (n - 1) \sum_i A_i \sum_i B_i}{n^2 k (k - 1)} \quad (10)$$

This estimation Eq. (10) is referred as *prodAB* estimation and it is an unbiased estimation because

$$E\hat{q} = \frac{(nk - 1) \sum_i E(A_i B_i) - (n - 1) E(\sum_i A_i \sum_i B_i)}{n^2 k (k - 1)} = q \quad (11)$$

The *probability error* of *prodAB* method can be estimated by Eq. (12) (proof in [9]). The error goes to zeros as the number of partitions, k , goes to infinite.

$$E(\hat{q} - q)^2 = \frac{(nk - 1)(n - 1)}{n^2 k (k - 1)} [(q - p_a p_b)^2 - p_a p_b (p_a + p_b)] + \frac{1}{nk} q - \frac{1}{n^2 k (k - 1)} q^2 \quad (12)$$

Moreover, *minAB* can be used to improve the accuracy of *prodAB* by restricting the estimation with the upper bound provided by *minAB*.

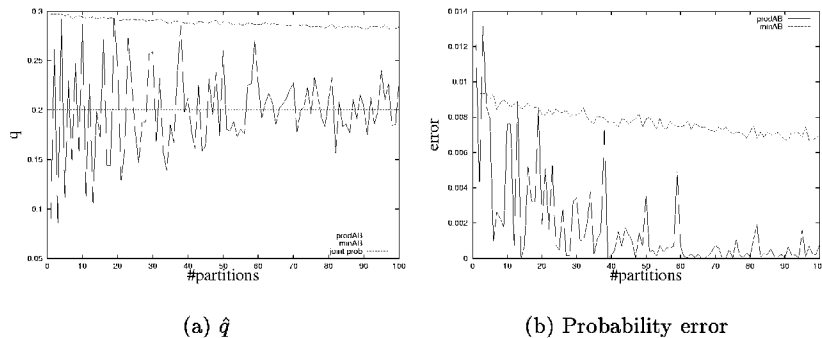


Fig. 4. Experimental results of *prodAB* and *minAB*, $p_a = p_b = 0.3, q = 0.2$

5 Experimental Results of Two Estimation Methods

In this section, we present the experimental results of two estimation methods: *prodAB* and *minAB*. We generate synthetic data sets with attributes a and b according to the known probabilities p_a , p_b and q . Each data set has $N = 100000$ data points. Fig.4 and Fig.5 illustrate the result of the experiments. It is clear that the *probability error* of the estimations decreases as the number of partitions increases. Also, we find that when a and b are strongly correlated, i.e., $q \sim \min(p_a, p_b)$ (e.g., Fig.5), the *minAB* method produces more stable results than the *prodAB* method. Otherwise, (e.g., Fig.4), the *prodAB* method provides more accurate estimations.

6 A Case Study for Fast Association Mining

In this section, we demonstrate a case study of using estimation methods for fast association mining. More applications of using estimation methods for rule pruning, online and distributed mining can be found in [9].

Table 1. Comparison of *MPSE1* and *Apriori*

Setting	Size of Itemsets	Number of Candidates Eliminated	Number of Candidates Remained	Total Number of Candidates	Number of Frequent Itemsets
D_1 with $k = 10$	2	955	72581	73536	1245
	3	1602	1352	2954	145
	4	17	62	79	62
	5	6	14	20	14
	6	0	1	1	1
D_1 with $k = 100$	2	10998	62538	73536	1245
	3	2788	166	2954	145
	4	17	62	79	62
	5	6	14	20	14
	6	0	1	1	1
D_2 with $k = 50$	2	919	72617	73536	1164
	3	1714	997	2711	63
	4	3	25	28	25
	5	1	7	8	7
	6	0	0	0	0
D_3 with $k = 100$	2	899	72264	73153	1143
	3	1702	971	2673	75
	4	19	20	39	20
	5	0	5	5	5
	6	0	0	0	0

6.1 Background on Association Mining

The association mining task, first introduced in [1] can be broken into two steps: The first step is to identify all frequent itemsets and the second step is to generate association rules with high confidence. There is a vast literature on this topic (for a survey, see [6]). Among the existing algorithms, *Apriori* [2] is a standard one and is used as the basis for many other existing algorithms. *Apriori* effectively uses the fact that the collection of frequent itemsets is a set lattice. To discover all frequent itemsets, *Apriori* goes by the size k of the itemsets that are examined. For $k = 1, 2, \dots$, one finds all frequent itemsets having cardinality exactly k , and then, based on the lattice property, generates the candidate for the next value of k . The search is entirely terminated when no frequent itemsets are discovered at any value of k , when, by the lattice property, one can guarantee that there is no frequent itemset of a larger size. Our estimation methods can be used

in the *Apriori* algorithm for fast association mining. In [8], a novel structure *segment support map (SSM)* is proposed to improve the performance of frequent-set mining algorithms. An SSM is a structure consisting of the support counts for all singleton itemsets in each segment of the transaction database. The authors also show how to use SSM in *Carma* [5] for efficient on-line mining. In this paper we extend the concept of SSM to partition support count and explore how to use the partition support to estimate the joint probabilities. We will show how to use the partition support estimate in *Apriori* for fast association mining.

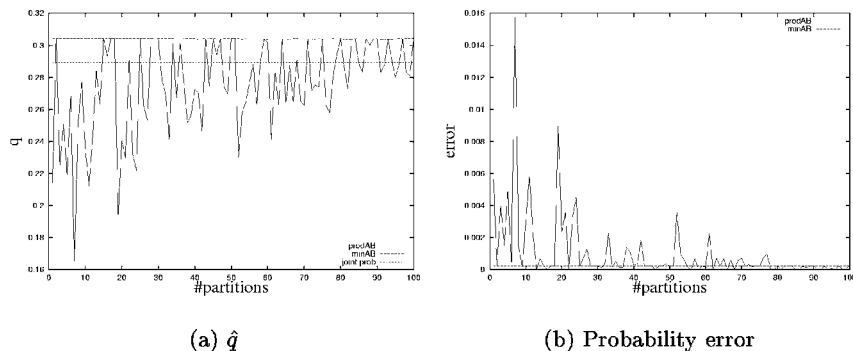


Fig. 5. Experimental results of *prodAB* and *minAB*, $p_a = 0.7, p_b = 0.3, q = 0.29$

6.2 Use Estimates for Fast Mining

Partition support based estimate methods can be used in *Apriori* to reduce the size of candidate itemsets or reduce the scans of the databases. The algorithm *MPSE1* (Mining with Partition Support Estimate 1) uses the *minAB* estimation method to prune the candidate itemsets. The underlying framework of *MPSE1* is the same as that of *Apriori* except that a process of partition support estimating is inserted between the candidate generation phase and the counting phase². In general, *MPSE1* can maintain the partition support of frequent i -itemsets for some $i \geq 1$. However, as described in Section 3.1, in our implementation we only maintain the partition support for singleton sets with the consideration of space and time complexity.

The estimation methods can be used more aggressively to reduce the number of scans. Algorithm *MPSE2* uses *Apriori* as the basis, but it scans the database only twice, at the beginning and the end. First, *MPSE2* scans the database to compute all frequent 1-itemsets and the partition support of all singleton sets. Then it executes *Apriori* for itemsets of size greater than one, but database scan is postponed. Instead, *MPSE2* estimates the support of each candidate

² Since the *minAB* estimate is an upper bound, so we will not eliminate any valid frequent itemsets.

itemset using *minAB* method. Eventually, *MPSE2* checks for each candidate that remained after pruning whether it is frequent or not.³

6.3 Experimental Result and Analysis

Test Organization. We evaluated the algorithms using synthetic datasets as well as real database. We used three synthetic datasets *T10.I5.D10k*, *T10.I5.D50k*, and *T10.I5.D100k* with 500 items as described in [2]. We denote them by D_1, D_2, D_3 . These database model supermarket basket data and have been used as benchmarks for association rule algorithms. We also tested our algorithms on a real database, called the Perinatal Database(*PD*)⁴. After preprocessing, the database has information of 16304 different birth records with 197 binary attributes. Unlike the sales market database, associations in the Perinatal Database are relatively large. The size of the maximal frequent itemset is twelve with the minimum support of 15%. All the experiments are performed on a Sun *U5/333* machine with 128 MB memory, running on Sun OS 5.7.

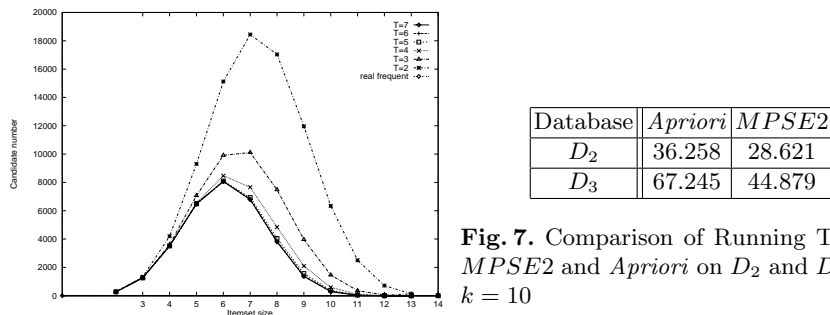


Fig. 6. Candidate Number Generated by *MPSE2* on *PD*

Fig. 7. Comparison of Running Time of *MPSE2* and *Apriori* on D_2 and D_3 with $k = 10$

Candidate Reducing using *MPSE1*. Table 1 shows the results on D_1 when the database is divided into ten parts (i.e., $k = 10$) and one hundred parts (i.e., $k = 100$), and the results on D_2 with $k = 50$ and D_3 with $k = 100$. The columns are the setting, the size of itemsets, the number of candidates eliminated by *MPSE1*, the number of candidates that remained in C after the pruning phase of *MPSE2*, the total number of candidates before pruning, and the number of frequent itemsets, respectively. There is a significant amount of reduction in the number of candidates for 2-itemsets and 3-itemsets.

³ *MPSE2* can be modified so that it takes as parameter T and switch from *Apriori* to *MPSE2* at level T . By default $T = 2$.

⁴ This database contains information of newborns in the Great Rochester area in the calendar year of 1998.

Candidate Reducing using *MPSE2*. Figure 6 shows the results of running *MPSE2* on *PD* with the minimum support of 15%, $k = 100$, for various values of T . We observe that with $T > 4$, the candidate set gets very close to the actual collection of frequent itemsets. This is very effective, given that the size of the maximal large itemsets is 12. Figure 7 shows the running time (in seconds) of *MPSE2* on D_2, D_3 with $k = 10$ and minimum support of 0.1% comparing with *Apriori*.

7 Conclusion

In this paper we present two methods of estimating joint probabilities and discuss their applications in data mining. We are currently implementing the idea of using the estimation methods for distributed and online mining.

References

1. R. Agrawal, T. Imielinski, and A. Swami. Mining associations between sets of items in massive databases. In *Proc. of ACM SIGMOD*, 1993.
2. R. Agrawal and R. Srikant. Fast algorithms for mining association rules in large databases. In *VLDB*, 1994.
3. S. Brin, R. Motwani, and C. Silverstein. Beyond market baskets: generalizing association rules to correlation. In *Proc. of ACM SIGMOD*, 1997.
4. Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, Prediction*. Springer, 2001.
5. C. Hidber. Online association rule mining. In *Proc. of ACM SIGMOD*, 1999.
6. J. Hipp, U. Güntzer, and G. Nakhaeizadeh. Algorithms for association rule mining: a general survey and comparison. *SIGKDD Explorations*, 2(1):58–63, 2000.
7. M. Kamber, J. Han, and J. Y. Chiang. Metarules-guided mining of multi-dimensional association rules using data cubes. In *Proc. of ACM SIGKDD*, 1997.
8. L. V. S. Lakshmanan, C. K. S. Leung, and R. T. Ng. The segment support map: Scalable mining of frequent itemsets. *SIGKDD Explorations*, 2(2), December 2000.
9. Tao Li, Shenghuo Zhu, Mitsunori Ogihara, and Yinhe Cheng. Estimating joint probabilities without combinatorial counting. Technical Report TR-764, Computer Science Department, University of Rochester, 2002.
10. C. Silverstein, S. Brin, R. Motwani, and J. Ullman. Scalable techniques for mining causal structures. In *Proc. of VLDB*, 1998.