Available at
www.ElsevierComputerScience.com
POWERED BY SCIENCE DIRECT®

Information Systems 29 (2004) 343–364

www.elsevier.com/locate/infosys

Privacy preserving mining of association rules☆

Alexandre Evfimievski[a], Ramakrishnan Srikant[b], Rakesh Agrawal[b], Johannes Gehrke[a,*]

[a] *Department of Computer Science, Cornell University, Ithaca, NY 14853, USA*
[b] *IBM Almaden Research Center, 650 Harry Road, San Jose, CA 95120, USA*

Abstract

We present a framework for mining association rules from transactions consisting of categorical items where the data has been randomized to preserve privacy of individual transactions. While it is feasible to recover association rules and preserve privacy using a straightforward "uniform" randomization, the discovered rules can unfortunately be exploited to find privacy breaches. We analyze the nature of privacy breaches and propose a class of randomization operators that are much more effective than uniform randomization in limiting the breaches. We derive formulae for an unbiased support estimator and its variance, which allow us to recover itemset supports from randomized datasets, and show how to incorporate these formulae into mining algorithms. Finally, we present experimental results that validate the algorithm by applying it on real datasets.
© 2003 Published by Elsevier Ltd.

Keywords: Data mining; Privacy; Association rule; Privacy breach

1. Introduction

The explosive progress in networking, storage, and processor technologies is resulting in an unprecedented amount of digitization of information. It is estimated that the amount of information in the world is doubling every 20 months [1]. In concert with this dramatic and escalating increase in digital data, concerns about privacy of personal information have emerged globally [1–4]. Privacy issues are further exacerbated now that the internet makes it easy for the new data to be automatically collected and added to databases [5–10]. The concerns over massive collection of data are naturally extending to analytic tools applied to data. Data mining, with its promise to efficiently discover valuable, nonobvious information from large databases, is particularly vulnerable to misuse [1,11–13].

An interesting new direction for data mining research is the development of techniques that incorporate privacy concerns [14]. The following question was raised in [15]: since the primary task in data mining is the development of models about aggregated data, can we develop accurate models without access to precise information in individual data records? Specifically, they studied the technical feasibility of building accurate classification models using training data in which the sensitive numeric values in a user's record have been

☆Recommended by D. Shasha.
*Corresponding author.
E-mail addresses: aevf@cs.cornell.edu (A. Evfimievski), srikant@almaden.ibm.com (R. Srikant), ragrawal@almaden.ibm.com (R. Agrawal), johannes@cs.cornell.edu (J. Gehrke).

0306-4379/$ - see front matter © 2003 Published by Elsevier Ltd.
doi:10.1016/j.is.2003.09.001

randomized so that the true values cannot be estimated with sufficient precision. Randomization is done using the statistical method of value distortion [16] that returns a value $x_i + r$ instead of x_i where r is a random value drawn from some distribution. They proposed a Bayesian procedure for correcting perturbed distributions and presented three algorithms for building accurate decision trees [17,18] that rely on reconstructed distributions.[1] In [20], the authors derived an Expectation Maximization (EM) algorithm for reconstructing distributions and proved that the EM algorithm converged to the maximum likelihood estimate of the original distribution based on the perturbed data. They also pointed out that the EM algorithm was in fact identical to the Bayesian reconstruction procedure in [15], except for an approximation (partitioning values into intervals) that was made by the latter.

1.1. Contributions of this paper

We continue the investigation of the use of randomization in developing privacy preserving data mining techniques, and extend this line of inquiry along two dimensions:

- categorical data instead of numerical data, and
- association rule mining [21] instead of classification.

We will focus on the task of finding frequent itemsets in association rule mining, which we briefly review next.

Definition 1. Suppose we have a set $\mathcal{I}$ of n items: $\mathcal{I} = \{a_1, a_2, \ldots, a_n\}$. Let T be a sequence of N transactions $T = (t_1, t_2, \ldots, t_N)$ where each transaction t_i is a subset of $\mathcal{I}$. Given an itemset $A \subset \mathcal{I}$, its *support* $\operatorname{supp}^T(A)$ is defined as

$$\operatorname{supp}^T(A) := \frac{\#\{t \in T \mid A \subseteq t\}}{N}. \quad (1)$$

An itemset $A \subset \mathcal{I}$ is called *frequent* in T if $\operatorname{supp}^T(A) \geqslant \tau$, where τ is a user-defined parameter.

[1] Once we have reconstructed distributions, it is straightforward to build classifiers that assume independence between attributes, such as Naive Bayes [19].

We consider the following setting. Suppose we have a server and many clients. Each client has a set of items (e.g., books or web pages or TV programs). The clients want the server to gather statistical information about associations among items, perhaps in order to provide recommendations to the clients. However, the clients do not want the server to know with certainty who has got which items. When a client sends its set of items to the server, it modifies the set according to some specific randomization policy. The server then gathers statistical information from the modified sets of items (transactions) and recovers from it the actual associations.

The following are the important results contained in this paper:

- In Section 2, we show that a straightforward uniform randomization leads to privacy breaches.
- We formally model and define privacy breaches in Section 3.
- We present a class of randomization operators in Section 4 that can be tuned for different tradeoffs between discoverability and privacy breaches. We derive formulae for the effect of randomization on support, and show how to recover the original support of an association from the randomized data.
- An estimator for the confidence of association rules is given as well, and its precision evaluated (see Section 4.6).
- We present experimental results on two real datasets in Section 5, as well as graphs showing the relationship between discoverability, privacy, and data characteristics.

1.2. Related work

There has been extensive research in the area of statistical databases motivated by the desire to provide statistical information (sum, count, average, maximum, minimum, pth percentile, etc.) without compromising sensitive information about individuals (see surveys in [22,23]). The proposed techniques can be broadly classified into query restriction and data perturbation. The query restriction family includes restricting the size of query result, controlling the overlap amongst successive queries, keeping audit trail of all answered queries and constantly checking for possible compromise, suppression of data cells of small size, and clustering entities into mutually exclusive atomic populations. The perturbation family includes swapping values between records, replacing the original database by a sample from the same distribution, adding noise to the values in the database, adding noise to the results of a query, and sampling the result of a query. There are negative results showing that the proposed techniques cannot satisfy the conflicting objectives of providing high quality statistics and at the same time prevent exact or partial disclosure of individual information [22].

The most relevant work from the statistical database literature is the work by Warner [24], where he developed the "randomized response" method for survey results. The method deals with a single boolean attribute (e.g., drug addiction). The value of the attribute is retained with probability p and flipped with probability $1 - p$. Warner then derived equations for estimating the true value of queries such as COUNT (Age = 42 & Drug Addiction = Yes). The approach we present in Section 2 can be viewed as a generalization of Warner's idea.

Another related work is [25], where they consider the problem of mining association rules over data that is vertically partitioned across two sources, i.e., for each transaction, some of the items are in one source, and the rest in the other source. They use multi-party computation techniques for scalar products to be able to compute the support of an itemset (when the two subsets that together form the itemset are in different sources), without either source revealing exactly which transactions support a subset of the itemset. In contrast, we focus on preserving privacy when the data is horizontally partitioned, i.e., we want to preserve privacy for individual transactions, rather than between two data sources that each have a vertical slice.

Related, but not directly relevant to our current work, is the problem of inducing decision trees over horizontally partitioned training data originating from sources who do not trust each other. In [12], each source first builds a local decision tree over its true data, and then swaps values amongst records in a leaf node of the tree to generate randomized training data. Another approach, presented in [26], does not use randomization, but makes use of cryptographic oblivious functions during tree construction to preserve privacy of two data sources.

This publication extends our conference paper [27]. At the same time, independently from our work, there was a paper [28] that considered another algorithm for randomizing transactions for privacy preserving mining of association rules. Some additional research in the framework of this paper is published in [29].

2. Uniform randomization

A straightforward approach for randomizing transactions would be to generalize Warner's "randomized response" method, described in Section 1.2. Before sending a transaction to the server, the client takes each item and with probability p replaces it by a new item not originally present in this transaction. Let us call this process *uniform* randomization.

Estimating true (nonrandomized) support of an itemset is nontrivial even for uniform randomization. Randomized support of, say, a 3-itemset depends not only on its true support, but also on the supports of its subsets. Indeed, it is much more likely that only one or two of the items are inserted by chance than all three. So, almost all "false" occurrences of the itemset are due to (and depend on) high subset supports. This requires estimating the supports of all subsets simultaneously. (The algorithm is similar to the algorithm presented in Section 4 for select-a-size randomization, and the formulae from Statements 1, 3 and 4 apply here as well.) For large values of p, most of the items in most randomized transactions will be "false", so we seem to have obtained a reasonable privacy protection. Also, if there are enough clients and transactions, then frequent itemsets will still be "visible", though less frequent than originally. For instance, after uniform randomization with $p = 80\%$, an itemset of 3 items that originally occurred in 1% transactions will occur in about $1\% \cdot (0.2)^3 = 0.008\%$ transactions, which is about 80 transactions per each million. The opposite effect of "false" itemsets becoming more frequent is comparatively negligible if there are many possible items: for 10,000 items, the probability that, say, 10 randomly inserted items contain a given 3-itemset is less than $10^{-7}\%$.

Unfortunately, this randomization has a problem. If we know that our 3-itemset escapes randomization in 80 per million transactions, and that it is unlikely to occur even once *because* of randomization, then every time we see it in a randomized transaction we know with near certainty of its presence in the nonrandomized transaction. With even more certainty we will know that at least one item from this itemset is "true": as we have mentioned, a chance insertion of only one or two of the items is much more likely than of all three. In this case we can say that a *privacy breach* has occurred. Although privacy is preserved on average, personal information leaks through uniform randomization for some fraction of transactions, despite the high value of p.

The rest of the paper is devoted to defining a framework for studying privacy breaches and developing techniques for finding frequent itemsets while avoiding breaches.

3. Privacy breaches

Definition 2. Let $(\Omega, \mathcal{F}, \mathbf{P})$ be a probability space of elementary events over some set Ω and σ-algebra $\mathcal{F}$. A *randomization operator* is a measurable function

$$R : \Omega \times \{\text{all possible } T\} \rightarrow \{\text{all possible } T\}$$

that randomly transforms a sequence of N transactions into a (usually) different sequence of N transactions. Given a sequence of N transactions T, we shall write $T' = R(T)$, where T is constant and $R(T)$ is a random variable.

Definition 3. Suppose that a nonrandomized sequence T is drawn from some known distribution, and $t_i \in T$ is the ith transaction in T. A *general privacy breach of level* ρ with respect to a property $P(t_i)$ occurs if

$$\exists T' : \mathbf{P}[P(t_i) \mid R(T) = T'] \geqslant \rho.$$

We say that a property $Q(T')$ *causes a privacy breach* of level ρ with respect to $P(t_i)$ if

$$\mathbf{P}[P(t_i) \mid Q(R(T))] \geqslant \rho.$$

When we define privacy breaches, we think of the prior distribution of transactions as known, so that it makes sense to speak about a posterior probability of a property $P(t_i)$ versus prior. In practice, however, we do not know the prior distribution. In fact, there is no prior distribution; the transactions are not randomly generated. However, modeling transactions as being randomly generated from a prior distribution allows us to cleanly define privacy breaches.

Consider a situation when, for some transaction $t_i \in T$, an itemset $A \subseteq \mathcal{I}$ and an item $a \in A$, the property "$A \subseteq t'_i \in T'$" causes a privacy breach w.r.t. the property "$a \in t_i$." In other words, the presence of A in a randomized transaction makes it likely that item a is present in the corresponding nonrandomized transaction.

Definition 4. We say that itemset A *causes a privacy breach of level* ρ if for some item $a \in A$ and some $i \in 1 \ldots N$ we have $\mathbf{P}[a \in t_i \mid A \subseteq t'_i] \geqslant \rho$.

We will focus on controlling the class of privacy breaches given by Definition 4. Thus we ignore the effect of other information the server obtains from a randomized transaction, such as which items the randomized transaction does not contain, or the randomized transaction size. We also do not attempt to control breaches that occur because the server knows other randomized transactions, or some other information about items and clients besides the transactions. For example, the server may know some geographical or demographic data about the clients. Finally, in Definition 4, we only considered positive breaches, i.e., we know with high probability that an item was present in the original transaction. In some scenarios, being confident that an item was *not* present in the original transaction may also be considered a privacy breach.

4. Algorithm

"Where does a wise man hide a leaf? In the forest. But what does he do if there is no forest?" ... "He grows a forest to hide it in."—G.K. Chesterton, "The Sign of the Broken Sword"

The intuition of breach control is quite simple: in addition to replacing some of the items, we shall insert so many "false" items into a transaction that one is as likely to see a "false" itemset as a "true" one.

4.1. Randomization operators

Definition 5. We call randomization R a *per-transaction* randomization if, for $T = (t_1, t_2, \dots, t_N)$, we can represent $R(T)$ as

$$R(t_1, t_2, \dots, t_N) = (R(1, t_1), R(2, t_2), \dots, R(N, t_N)),$$

where $R(i, t)$ are independent random variables whose distributions depend only on t (and not on i). We shall write $t'_i = R(i, t_i) = R(t_i)$.

Definition 6. A randomization operator R is called *item-invariant* if, for every transaction sequence T and for every permutation $\pi : \mathcal{I} \to \mathcal{I}$ of items, the distribution of $\pi^{-1}R(\pi T)$ is the same as of $R(T)$. Here πT means the application of π to all items in all transactions of T at once.

Definition 7. A *select-a-size* randomization operator has the following parameters, for each possible input transaction size m:

- Default probability of an item (also called randomization level) $\rho_m \in (0, 1)$;
- Transaction subset size selection probabilities $p_m[0], p_m[1], \dots, p_m[m]$, such that every $p_m[j] \geqslant 0$ and

  $$p_m[0] + p_m[1] + \dots + p_m[m] = 1.$$

Given a sequence of transactions $T = (t_1, t_2, \dots, t_N)$, the operator takes each transaction t_i independently and proceeds as follows to obtain transaction t'_i ($m = |t_i|$).

(1) The operator selects an integer j at random from the set $\{0, 1, \dots, m\}$ so that $\mathbf{P}[j \textit{ is selected}] = p_m[j]$.
(2) It selects j items from t_i, uniformly at random (without replacement). These items, and no other items of t_i, are placed into t'_i.
(3) It considers each item $a \notin t_i$ in turn and tosses a coin with probability ρ_m of "heads" and $1 - \rho_m$ of "tails". All those items for which the coin faces "heads" are added to t'_i.

Remark 4.1. Both uniform (Section 2) and select-a-size operators are per-transaction because they apply the same randomization algorithm to each transaction independently. They are also item-invariant since they do not use any item-specific information (if we rename or reorder the items, the outcome probabilities will not be affected).

Definition 8. A *cut-and-paste* randomization operator is a special case of a select-a-size operator (and which we shall actually test on datasets). For each possible input transaction size m, it has two parameters: $\rho_m \in (0, 1)$ (randomization level) and an integer $K_m > 0$ (the cutoff). The operator takes each input transaction t_i independently and proceeds as follows to obtain transaction t'_i (here $m = |t_i|$):

(1) It chooses an integer j uniformly at random between 0 and K_m; if $j > m$, it sets $j = m$.
(2) The operator selects j items out of t_i uniformly at random (without replacement). These items are placed into t'_i.
(3) Each other item (including the rest of t_i) is placed into t'_i with probability ρ_m, independently.

Remark 4.2. For any m, a cut-and-paste operator has only two parameters, ρ_m and K_m, to play with; moreover, K_m is an integer. Because it is easy to find optimal values for these parameters (Section 4.4), we chose to test this operator, leaving open the problem of optimizing the m parameters of the "unabridged" select-a-size. To see that cut-and-paste is a case of select-a-size, let us write down the formulae for the $p_m[j]$'s:

$$p_m[j] = \sum_{i=0}^{\min\{K,j\}} \binom{m-i}{j-i} \rho^{j-i}(1-\rho)^{m-j} \times \begin{cases} 1 - m/(K+1) & \text{if } i = m \text{ and } i < K, \\ 1/(K+1) & \text{otherwise.} \end{cases}$$

Now let us give one example of a randomization operator that is not a per-transaction randomization, because it uses the knowledge of several transactions per each randomized transaction.

Example 4.1. The mixing randomization operator has one integer parameter $K \geqslant 2$ and one real-valued parameter $p \in (0, 1)$. Given a sequence of transactions $T = (t_1, t_2, \ldots, t_N)$, the operator takes each transaction t_i independently and proceeds as follows to obtain transaction t'_i:

(1) Other than t_i, pick $K - 1$ more transactions (with replacement) from T and union the K transactions as sets of items. Let t''_i be this union.
(2) Consider each item $a \in t''_i$ in turn and toss a coin with probability p of "heads" and $1 - p$ of "tails".
(3) All those items for which the coin faces "tails" are removed from the transaction. The remaining items constitute the randomized transaction.

For the purpose of privacy-preserving data mining, it is natural to focus mostly on per-transaction randomizations, since they are the easiest and safest to implement. Indeed, a per-transaction randomization does not require the users (who submit randomized transactions to the server) to communicate with each other in any way, nor to exchange random bits. On the contrary, implementing mixing randomization, for example, requires to organize an exchange of nonrandomized transactions between users, which opens an opportunity for cheating or eavesdropping.

4.2. *Effect of randomization on support*

Let T be a sequence of transactions of length N, and let A be some subset of items (that is, $A \subseteq \mathcal{I}$). Suppose we randomize T and get $T' = R(T)$. The support $s' = \mathrm{supp}^{T'}(A)$ of A for T' is a random variable that depends on the outcome of randomization. Here we are going to determine the distribution of s', under the assumption of having a per-transaction and item-invariant randomization.

Definition 9. The fraction of the transactions in T that have intersection with A of size l among all transactions in T is called *partial support* of A for intersection size l:

$$\mathrm{supp}_l^T(A) := \frac{\#\{t \in T \mid \#(A \cap t) = l\}}{N}. \tag{2}$$

It is easy to see that $\mathrm{supp}^T(A) = \mathrm{supp}_k^T(A)$ for $k = |A|$, and that

$$\sum_{l=0}^{k} \mathrm{supp}_l^T(A) = 1$$

since those transactions in T that do not intersect A at all are covered in $\mathrm{supp}_0^T(A)$.

Definition 10. Suppose that our randomization operator is both per-transaction and item-invariant. Consider a transaction t of size m and an itemset $A \subset \mathcal{I}$ of size k. After randomization, transaction t becomes t'. We define

$$p_k^m[l \to l'] = p[l \to l'] := \mathbf{P}[\#(t' \cap A) = l' \mid \#(t \cap A) = l]. \tag{3}$$

Here both l and l' must be integers in $\{0, 1, \ldots, k\}$.

Remark 4.3. The value of $p_k^m[l \to l']$ is well-defined (does not depend on any other information about t and A, or other transactions in T and T' besides t and t'). Indeed, because we have a per-transaction randomization, the distribution of t' depends neither on other transactions in T besides t, nor on their randomized outcomes. If there were other t_1 and B with the same (m, k, l), but a different probability (3) for the same l', we could consider a permutation π of $\mathcal{I}$ such that $\pi t = t_1$ and $\pi A = B$; the application of π or of π^{-1} would preserve intersection sizes l and l'. By item-invariance we have

$$\mathbf{P}[\#(t' \cap A) = l'] = \mathbf{P}[\#(\pi^{-1} R(\pi t) \cap A) = l'],$$

but by the choice of π we also have

$$\begin{aligned}&\mathbf{P}[\#(\pi^{-1}R(\pi t)\cap A)=l'] \\ &\quad=\mathbf{P}[\#(\pi^{-1}R(t_1)\cap\pi^{-1}B)=l'] \\ &\quad=\mathbf{P}[\#(t'_1\cap B)=l']\neq\mathbf{P}[\#(t'\cap A)=l'],\end{aligned}$$

a contradiction.

Statement 1. *Suppose that our randomization operator is both per-transaction and item-invariant. Suppose also that all the N transactions in T have the same size m. Then, for a given subset $A\subseteq\mathcal{I}$, $|A|=k$, the random vector*

$$N\cdot(s'_0,s'_1,\ldots,s'_k),\quad \textit{where } s'_l:=\mathrm{supp}_l^{T'}(A) \tag{4}$$

is a sum of $k+1$ independent random vectors, each having a multinomial distribution. Its expected value is given by

$$\mathbf{E}(s'_0,s'_1,\ldots,s'_k)^{\mathrm{T}}=P\cdot(s_0,s_1,\ldots,s_k)^{\mathrm{T}} \tag{5}$$

where P is the $(k+1)\times(k+1)$ matrix with elements $P_{l'l}=p[l\to l']$, and the covariance matrix is given by

$$\mathbf{Cov}(s'_0,s'_1,\ldots,s'_k)^{\mathrm{T}}=\frac{1}{N}\cdot\sum_{l=0}^{k}s_lD[l] \tag{6}$$

where each $D[l]$ is a $(k+1)\times(k+1)$ matrix with elements

$$D[l]_{ij}=p[l\to i]\cdot\delta_{i=j}-p[l\to i]\cdot p[l\to j]. \tag{7}$$

Here s_l denotes $\mathrm{supp}_l^T(A)$, and the T over vectors denotes the transpose operation; $\delta_{i=j}$ is one if $i=j$ and zero otherwise.

Proof. See Appendix A.1.

Remark 4.4. In Statement 1 we have assumed that all transactions in T have the same size. If this is not so, we have to consider each transaction size separately and then use per-transaction independence.

Statement 2. *For a select-a-size randomization with randomization level ρ and size selection probabilities $\{p_m[j]\}$, we have*

$$p_k^m[l\to l']=\sum_{j=0}^{m}p_m[j]\cdot\sum_{q=\max\{0,j+l-m,l+l'-k\}}^{\min\{j,l,l'\}}\frac{\binom{l}{q}\binom{m-l}{j-q}}{\binom{m}{j}}\times\binom{k-l}{l'-q}\rho^{l'-q}(1-\rho)^{k-l-l'+q}. \tag{8}$$

Proof. See Appendix A.2.

4.3. Support recovery

Let us assume that all transactions in T have the same size m, and let us denote

$$\vec{s}:=(s_0,s_1,\ldots,s_k)^{\mathrm{T}},\quad \vec{s}':=(s_0,s_1,\ldots,s_k)^{\mathrm{T}};$$

then, according to (5), we have

$$\mathbf{E}\vec{s}'=P\cdot\vec{s}. \tag{9}$$

Denote $Q=P^{-1}$ (assume that it exists) and multiply both sides of (9) by Q:

$$\vec{s}=Q\cdot\mathbf{E}\vec{s}'=\mathbf{E}Q\cdot\vec{s}'.$$

We have thus obtained an unbiased estimator for the original partial supports given randomized partial supports:

$$\vec{s}_{\mathrm{est}}:=Q\cdot\vec{s}'. \tag{10}$$

Using (6), we can compute the covariance matrix of $\vec{s}_{\mathrm{est}}$:

$$\begin{aligned}\mathbf{Cov}\,\vec{s}_{\mathrm{est}}&=\mathbf{Cov}(Q\cdot\vec{s}')=Q(\mathbf{Cov}\,\vec{s}')Q^{\mathrm{T}} \\ &=\frac{1}{N}\cdot\sum_{l=0}^{k}s_lQD[l]Q^{\mathrm{T}}.\end{aligned} \tag{11}$$

If we want to estimate this covariance matrix by looking only at randomized data, we may use $\vec{s}_{\mathrm{est}}$ instead of $\vec{s}$ in (11):

$$(\mathbf{Cov}\,\vec{s}_{\mathrm{est}})_{\mathrm{est}}=\frac{1}{N}\cdot\sum_{l=0}^{k}(\vec{s}_{\mathrm{est}})_lQD[l]Q^{\mathrm{T}}.$$

This estimator is also unbiased:

$$\mathbf{E}(\mathbf{Cov}\,\vec{s}_{\mathrm{est}})_{\mathrm{est}}=\frac{1}{N}\cdot\sum_{l=0}^{k}(\mathbf{E}\ \vec{s}_{\mathrm{est}})_lQD[l]Q^{\mathrm{T}}=\mathbf{Cov}\,\vec{s}_{\mathrm{est}}.$$

In practice, we want only the kth coordinate of $\vec{s}$, that is, the support $s=\mathrm{supp}^T(A)$ of our itemset A

in T. We denote by $\tilde{s}$ the kth coordinate of $\vec{s}_{\text{est}}$, and use $\tilde{s}$ to estimate s. Let us compute simple formulae for $\tilde{s}$, its variance and the unbiased estimator of its variance. Denote

$$q[l \leftarrow l'] := Q_{ll'}.$$

Statement 3.

$$\tilde{s} = \sum_{l'=0}^{k} s'_{l'} \cdot q[k \leftarrow l'];$$

$$\mathbf{Var}\,\tilde{s} = \frac{1}{N} \sum_{l=0}^{k} s_l \left(\sum_{l'=0}^{k} p[l \rightarrow l'] q[k \leftarrow l']^2 - \delta_{l=k} \right);$$

$$(\mathbf{Var}\,\tilde{s})_{\text{est}} = \frac{1}{N} \sum_{l'=0}^{k} s'_{l'} (q[k \leftarrow l']^2 - q[k \leftarrow l']).$$

Proof. See Appendix A.3.

We conclude this subsection by giving a linear coordinate transformation in which the matrix P from Statement 1 becomes triangular. (We use this transformation for privacy breach analysis in Section 4.4.) The coordinates after the transformation have a combinatorial meaning, as given in the following definition.

Definition 11. Suppose we have a transaction sequence T and an itemset $A \subseteq \mathcal{I}$. Given an integer l between 0 and $k = |A|$, consider all subsets $C \subseteq A$ of size l. The sum of supports of all these subsets is called the *cumulative support* for A of order l and is denoted as follows:

$$\Sigma_l = \Sigma_l(A, T) := \sum_{C \subseteq A,\ |C| = l} \operatorname{supp}^T(C),$$

$$\vec{\Sigma} := (\Sigma_0, \Sigma_1, \ldots, \Sigma_k)^{\mathrm{T}}. \tag{12}$$

Statement 4. *The vector $\vec{\Sigma}$ of cumulative supports is a linear transformation of the vector $\vec{s}$ of partial supports, namely,*

$$\Sigma_l = \sum_{j=l}^{k} \binom{j}{l} s_j \quad \textit{and} \quad s_l = \sum_{j=l}^{k} (-1)^{j-l} \binom{j}{l} \Sigma_j; \tag{13}$$

in the $\vec{\Sigma}$ and $\vec{\Sigma}'$ space (instead of $\vec{s}$ and $\vec{s}'$) matrix P is lower triangular.

Proof. See Appendix A.4.

4.4. Limiting privacy breaches

Here we determine how privacy depends on randomization. We shall use Definition 4 and assume a per-transaction and item-invariant randomization.

Consider some itemset $A \subseteq \mathcal{I}$ and some item $a \in A$; fix a transaction size m. We shall assume that m is known to the server, so that we do not have to combine probabilities for different nonrandomized sizes. Assume also that a partial support $s_l = \operatorname{supp}_l^T(A)$ approximates the corresponding prior probability $\mathbf{P}[\#(t \cap A) = l]$.[2] Suppose we know the following prior probabilities:

$$s_l^+ := \mathbf{P}[\#(t \cap A) = l,\ a \in t],$$
$$s_l^- := \mathbf{P}[\#(t \cap A) = l,\ a \notin t].$$

Notice that $s_l = s_l^+ + s_l^-$ simply because

$$\#(t \cap A) = l \ \Leftrightarrow \ \left[\begin{array}{l} a \in t \ \& \ \#(t \cap A) = l, \ \text{or} \\ a \notin t \ \& \ \#(t \cap A) = l. \end{array} \right.$$

Let us use these priors and compute the posterior probability of $a \in t$ given $A \subseteq t'$:

$$\begin{aligned} \mathbf{P}[a \in t \mid A \subseteq t'] &= \frac{\mathbf{P}[a \in t, A \subseteq t']}{\mathbf{P}[A \subseteq t']} \\ &= \frac{\sum_{l=1}^{k} \mathbf{P}[\#(t \cap A) = l,\ a \in t, A \subseteq t']}{\sum_{l=0}^{k} s_l \cdot p[l \rightarrow k]} \\ &= \frac{\sum_{l=1}^{k} \mathbf{P}[\#(t \cap A) = l,\ a \in t] \cdot p[l \rightarrow k]}{\sum_{l=0}^{k} s_l \cdot p[l \rightarrow k]} \\ &= \frac{\sum_{l=1}^{k} s_l^+ \cdot p[l \rightarrow k]}{\sum_{l=0}^{k} s_l \cdot p[l \rightarrow k]}. \end{aligned}$$

Thus, in order to prevent privacy breaches of level 50% as defined in Definition 4, we need to ensure that always

$$\sum_{l=1}^{k} s_l^+ \cdot p[l \rightarrow k] < 0.5 \cdot \sum_{l=0}^{k} s_l \cdot p[l \rightarrow k]. \tag{14}$$

[2] Here I denotes all transactions of size m.

The problem is that we have to randomize the data *before* we know any supports. Also, we may not have the luxury of setting "oversafe" randomization parameters because then we may not have enough data to perform a reasonably accurate support recovery. One way to achieve a compromise is to:

(1) Estimate maximum possible support $s_{\max}(k,m)$ of a k-itemset in the transactions of given size m, for different k and m;
(2) Given the maximum supports, find values for s_l and s_l^+ that are most likely to cause a privacy breach;
(3) Make randomization just strong enough to prevent such a privacy breach.

Since $s_0^+ = 0$, the most privacy-challenging situations occur when s_0 is small, that is, when our itemset A and its subsets are frequent.

In our experiments we consider a privacy-challenging k-itemset A such that, for every $l > 0$, all its subsets of size l have the maximum possible support $s_{\max}(l,m)$. The partial supports for such a test-itemset are computed from the cumulative supports Σ_l using Statement 4. By it and by (12), we have ($l > 0$)

$$s_l = \sum_{j=l}^{k} (-1)^{j-l} \binom{j}{l} \Sigma_j, \quad \Sigma_j = \binom{k}{j} s_{\max}(j,m) \tag{15}$$

since there are $\binom{k}{j}$ j-subsets in A. The values of s_l^+ follow if we note that all l-subsets of A, with a and without, appear equally frequently as $t \cap A$:

$$\begin{aligned} s_l^+ &:= \mathbf{P}[\#(t \cap A) = l,\ a \in t] \\ &= \mathbf{P}[a \in t \mid \#(t \cap A) = l] \cdot s_l = l/k \cdot s_l. \end{aligned}$$

While one can construct cases that are even more privacy-challenging (for example, if $a \in A$ occurs in a transaction every time any nonempty subset of A does), we found the above model (15) and (16) to be sufficiently pessimistic on our datasets.

We can now use these formulae to obtain cut-and-paste randomization parameters ρ_m and K_m as follows. Given m, consider all cutoffs from $K_m = 3$ to some $K_{\max}$ (usually this $K_{\max}$ equals the maximum transaction size) and determine the smallest randomization levels $\rho_m(K_m)$ that satisfy (14). Then select (K_m, ρ_m) that gives the best discoverability (by computing the lowest discoverable supports, see Section 5.1).

4.5. Discovering associations

We show how to discover itemsets with high true support given a set of randomized transactions. Although we use the Apriori algorithm [30] to make the ideas concrete, the modifications directly apply to any algorithm that uses Apriori candidate generation, i.e., to most current association discovery algorithms.[3] The key *lattice property* of supports used by Apriori is that, for any two itemsets $A \subseteq B$, the true support of A is equal to or larger than the true support of B. A simplified version of Apriori, given a (nonrandomized) transactions file and a minimum support $s_{\min}$, works as follows:

(1) Let $k = 1$, let "candidate sets" be all single items. Repeat the following until no candidate sets are left:

(a) Read the data file and compute the supports of all candidate sets;
(b) Discard all candidate sets whose support is below $s_{\min}$;
(c) Save the remaining candidate sets for output;
(d) Form all possible $(k+1)$-itemsets such that all their k-subsets are among the remaining candidates. Let these itemsets be the new candidate sets.
(e) Let $k = k + 1$.

(2) Output all the saved itemsets.

It is (conceptually) straightforward to modify this algorithm so that now it reads the randomized dataset, computes partial supports of all candidate sets (for all nonrandomized transaction sizes) and recovers their predicted supports and sigmas using the formulae from Statement 3. However, for the predicted supports the lattice property is no longer

[3] The main class of algorithms where this would not apply are those that find only maximal frequent itemsets, e.g., [31]. However, randomization precludes finding very long itemsets, so this is a moot point.

true. It is quite likely that for an itemset that is slightly above minimum support and whose predicted support is also above minimum support, that one of its subsets will have predicted support below minimum support. So if we discard all candidates below minimum support for the purpose of candidate generation, we will miss many (perhaps even the majority) of the longer frequent itemsets. Hence, for candidate generation, we discard only those candidates whose predicted support is "significantly" smaller than $s_{\min}$, where significance is measured by means of predicted sigmas. Here is the modified version of Apriori:

(1) Let $k = 1$, let "candidate sets" be all single-item sets. Repeat the following until k is too large for support recovery (or until no candidate sets are left):

(a) Read the randomized data file and compute the partial supports of all candidate sets, separately for each nonrandomized transaction size[4];
(b) Recover the predicted supports and sigmas for the candidate sets;
(c) Discard every candidate set whose support is below its *candidate limit*;
(d) Save for output only those candidate sets whose predicted support is at least $s_{\min}$;
(e) Form all possible $(k+1)$-itemsets such that all their k-subsets are among the remaining candidates. Let these itemsets be the new candidate sets.
(f) Let $k = k + 1$.

(2) Output all the saved itemsets.

We tried $s_{\min} - \sigma$ and $s_{\min} - 2\sigma$ as the candidate limit, and found that the former does a little better than the latter. It prunes more itemsets and therefore makes the algorithm work faster, and, when it discards a subset of an itemset with high predicted support, it usually turns out that the true support of this itemset is not as high.

[4] In our experiments, the nonrandomized transaction size is always known and included as a field into every randomized transaction.

4.6. Estimating confidence of association rules

Now we would like to see what happens if the support estimators from a randomized dataset are used for the computation of confidence for association rules.

Definition 12. Consider two disjoint itemsets A and B; the confidence of the association rule "$A \Rightarrow B$" is defined as

$$\mathrm{conf}^T(A \Rightarrow B) := \frac{\mathrm{supp}^T(A \cup B)}{\mathrm{supp}^T(A)}. \tag{16}$$

Perhaps the simplest way to estimate $\mathrm{conf}^T(A \Rightarrow B)$ is by replacing the actual supports in 16 with the predicted supports estimated from a randomized dataset as described in Section 4.3. Denote by X the unbiased estimator for the support of $A \cup B$, and by Y the unbiased estimator for the support of A; then the actual confidence $\mathrm{conf}^T(A \Rightarrow B)$ equals $\mathbf{E}X/\mathbf{E}Y$, and the confidence estimator $(\mathrm{conf}^T(A \Rightarrow B))_{\mathrm{est}}$ equals X/Y. Mathematical expectation of this estimator does not behave well, because Y may sometimes fall too close to zero (though it happens rarely if A is a frequent itemset). However, one can compute the probability $p = f(\delta)$ that the actual confidence is within a given distance $\delta > 0$ from the estimator. Below we shall give the formulae needed for computing $f(\delta)$.

Let us denote

$$\alpha := \frac{\mathbf{E}X}{\mathbf{E}Y} - \delta, \quad \beta := \frac{\mathbf{E}X}{\mathbf{E}Y} + \delta.$$

Then the probability $f(\delta)$ that our estimator X/Y gets within δ from $(\mathbf{E}X/\mathbf{E}Y)$ is

$$\begin{aligned} f(\delta) &:= \mathbf{P}\left[\frac{X}{Y} - \delta \leqslant \frac{\mathbf{E}X}{\mathbf{E}Y} \leqslant \frac{X}{Y} + \delta\right] \\ &= \mathbf{P}[\alpha \leqslant X/Y \leqslant \beta] \\ &= \mathbf{P}[\alpha Y \leqslant X \leqslant \beta Y,\ Y > 0] \\ &\quad + \mathbf{P}[\beta Y \leqslant X \leqslant \alpha Y,\ Y < 0]. \end{aligned}$$

For simplicity, we assume that the distribution of (X, Y) is a two-dimensional Gaussian. The assumption seems justified because both X and Y are linear combinations of multinomials (see Statement 1). In this case,

$$f(\delta) = \int_{-\infty}^{+\infty} \left| \int_{\alpha y}^{\beta y} \frac{1}{2\pi\,|\Sigma|^{1/2}} \times \mathrm{e}^{-\frac{1}{2}\begin{pmatrix} x-\mathbf{E}X \\ y-\mathbf{E}Y \end{pmatrix}^{\mathrm{T}} \Sigma^{-1} \begin{pmatrix} x-\mathbf{E}X \\ y-\mathbf{E}Y \end{pmatrix}} \,\mathrm{d}x \right| \mathrm{d}y,$$

where Σ is the 2×2 matrix

$$\Sigma := \begin{pmatrix} \mathbf{Var}\,X & \mathbf{Cov}(X,Y) \\ \mathbf{Cov}(X,Y) & \mathbf{Var}\,Y \end{pmatrix}.$$

The formulae for $\mathbf{Var}\,X$ and $\mathbf{Var}\,Y$ are given in Statement 3. The computation of $\mathbf{Cov}(X, Y)$ is a little more involved; intuitively, it requires to do Sections 4.2 and 4.3 for two itemsets A and B simultaneously. Let us start with a notion of "two-set partial supports."

Definition 13. The *two-set partial supports* for disjoint itemsets A and B are defined as follows:

$$s_{l_A,l_B} = \mathrm{supp}^T_{l_A,l_B}(A;B) := \frac{\#\left\{ t \in T : \begin{array}{c} |t \cap A| = l_A, \\ |t \cap B| = l_B \end{array} \right\}}{N}.$$

Here $l_A = 0, 1, \ldots, |A|$ and $l_B = 0, 1, \ldots, |B|$.

The vector of two-set partial supports is denoted by $\vec{s}_{\mathrm{II}}$ and has $(|A|+1)(|B|+1)$ coordinates that sum up to 1. The vector of two-set partial supports for the randomized dataset is denoted by $\vec{s}\,'_{\mathrm{II}}$. Analogously to (3), there is a two-set version of $p[l \to l']$:

$$p^m_{|A|,|B|}\begin{bmatrix} l_A \to l'_A \\ l_B \to l'_B \end{bmatrix} := \mathbf{P}\begin{bmatrix} |t' \cap A| = l'_A & |t \cap A| = l_A \\ |t' \cap B| = l'_B & |t \cap B| = l_B \end{bmatrix}.$$

For a per-transaction, item-invariant randomization operator, this probability depends only on m, $|A|$, $|B|$, l_A, l_B, l'_A, and l'_B by the same argument as for $p[l \to l']$ (see Remark 4.3). If our randomization operator is a select-a-size with parameters ρ and $p_m[i]$ (see Definition 7), then

$$p^m_{|A|,|B|}\begin{bmatrix} l_A \to l'_A \\ l_B \to l'_B \end{bmatrix} = \sum_{i=0}^{m-l_A-l_B} \sum_{\substack{j_A = \max\{0, l_A + l'_A - |A|\} \\ j_B = \max\{0, l_B + l'_B - |B|\}}}^{\substack{j_A \leqslant \min\{l_A, l'_A\} \\ j_B \leqslant \min\{l_B, l'_B\}}} p_m[i + j_A + j_B]$$
$$\times \binom{m}{i + j_A + j_B}^{-1} \binom{l_A}{j_A} \binom{l_B}{j_B} \times \binom{m - l_A - l_B}{i} \times \binom{|A| - l_A}{l'_A - j_A} \binom{|B| - l_B}{l'_B - j_B}$$
$$\times \rho^{l'_A + l'_B - j_A - j_B} (1-\rho)^{|A| + |B| - l_A - l_B - l'_A - l'_B + j_A + j_B}.$$

We denote the $(|A|+1)(|B|+1) \times (|A|+1)(|B|+1)$ matrix of these two set probabilities by P_{II}, and its inverse P^{-1}_{II} by Q_{II}. Then, analogously to Statement 1, we have

$$\mathbf{E}\vec{s}\,'_{\mathrm{II}} = P_{\mathrm{II}}\vec{s}_{\mathrm{II}}, \quad \mathbf{Cov}\,\vec{s}\,'_{\mathrm{II}} = \frac{1}{N} \times \sum_{\substack{0 \leqslant l_A \leqslant |A| \\ 0 \leqslant l_B \leqslant |B|}} s_{l_A,l_B} D_{\mathrm{II}}[l_A, l_B],$$

where

$$D_{\mathrm{II}}[l_A, l_B]^{i_A,i_B}_{j_A,j_B} = p\begin{bmatrix} l_A \to i_A \\ l_B \to i_B \end{bmatrix} \cdot \delta_{\substack{i_A = j_A \\ i_B = j_B}} - p\begin{bmatrix} l_A \to i_A \\ l_B \to i_B \end{bmatrix} \cdot p\begin{bmatrix} l_A \to j_A \\ l_B \to j_B \end{bmatrix}.$$

Now we define the two-set partial support estimator and compute its covariance matrix:

$$(\vec{s}_{\mathrm{II}})_{\mathrm{est}} = Q_{\mathrm{II}}\vec{s}\,'_{\mathrm{II}},$$

$$\mathbf{Cov}(\vec{s}_{\mathrm{II}})_{\mathrm{est}} = \frac{1}{N} \cdot \sum_{\substack{0 \leqslant l_A \leqslant |A| \\ 0 \leqslant l_B \leqslant |B|}} s_{l_A,l_B} Q_{\mathrm{II}} D_{\mathrm{II}}[l_A, l_B] Q^{\mathrm{T}}_{\mathrm{II}}. \quad (17)$$

The "one-set" partial support estimators for sets $A \cup B$ and A can be obtained from $(\vec{s}_{\mathrm{II}})_{\mathrm{est}}$ by linear combination of its coordinates. In particular,

X and Y can be computed as follows:

$$X = (\vec{s}_{\mathrm{II}})_{\mathrm{est}}[|A|, |B|], \quad Y = \sum_{l_B=0}^{|B|} (\vec{s}_{\mathrm{II}})_{\mathrm{est}}[|A|, l_B]. \tag{18}$$

Remark 4.5. These estimators are exactly the same as those computed in Section 4.3. Indeed, consider for example set A; let an $(|A|+1)\times(|A|+1)(|B|+1)$-matrix M be such that $M\vec{s}_{\mathrm{II}}$ equals the "one-set" vector $\vec{s}$ of partial supports for A. Since we have

$$\mathbf{E}\vec{s}\,' = P\vec{s}, \quad \mathbf{E}\vec{s}\,'_{\mathrm{II}} = P_{\mathrm{II}}\vec{s}_{\mathrm{II}}, \quad \vec{s} = M\vec{s}_{\mathrm{II}}, \quad \vec{s}\,' = M\vec{s}\,'_{\mathrm{II}}$$

for all possible partial support vectors, hence we also have

$$PM = MP_{\mathrm{II}} \;\Rightarrow\; MQ_{\mathrm{II}} = QM,$$

i.e. both estimators do the same thing.

By combining Eqs. (17) and (18), the formula for $\mathbf{Cov}(X, Y)$ is given by

$$\mathbf{Cov}(X, Y) = \frac{1}{N}\cdot \sum_{\substack{0\leqslant l_A\leqslant |A|\\ 0\leqslant l_B\leqslant |B|}} s_{l_A,l_B}\vec{q}_X^{\,\mathrm{T}} D_{\mathrm{II}}[l_A, l_B]\vec{q}_Y,$$

where (the $q[\ldots]$'s are the coordinates of Q_{II})

$$\vec{q}_X := \left(q\begin{bmatrix}|A|\leftarrow l'_A\\ |B|\leftarrow l'_B\end{bmatrix} : \begin{matrix}l'_A = 0, 1, \ldots, |A|\\ l'_B = 0, 1, \ldots, |B|\end{matrix}\right)^{\mathrm{T}},$$

$$\vec{q}_Y := \sum_{l_B=0}^{|B|}\left(q\begin{bmatrix}|A|\leftarrow l'_A\\ l_B\leftarrow l'_B\end{bmatrix} : \begin{matrix}l'_A = 0, 1, \ldots, |A|\\ l'_B = 0, 1, \ldots, |B|\end{matrix}\right)^{\mathrm{T}}.$$

5. Experimental results

Before we come to the experiments with datasets, we first show in Section 5.1 how our ability to recover supports depends on the permitted breach level, as well as other data characteristics. The same is shown for the confidence of association rules in Section 5.2. We then describe the real-life datasets in Section 5.3, and present results for support estimation on these datasets in Section 5.4.

5.1. Privacy, discoverability and dataset characteristics

We define the *lowest discoverable support* as the support at which the predicted support of an itemset is four sigmas away from zero, i.e., we can clearly distinguish the support of this itemset from zero. In practice, we may achieve reasonably good results even if the minimum support level is slightly lower than four sigma (as was the case for 3-itemsets in the randomized `soccer`, see below). However, the lowest discoverable support is a nice way to illustrate the interaction between discoverability, privacy breach levels, and data characteristics.

In order to compute the cut-and-paste randomization parameters for the graphs, we have to fix some values for maximum supports $s_{\max}(k, m)$. We do it as follows:

Itemset size:	1	2	3	4	5	6	7	8	9	10
Support(%):	20.0	10.0	5.0	2.0	1.0	0.5	0.2	0.1	0.05	0.02

We set the cutoff value to 5 (except in Fig. 3, where it is 7). To set the partial supports of the itemset being mined, we use *independence assumption*: all items within the itemset occur independently of each other and with the same frequency.

Fig. 1 shows how the lowest discoverable support changes with the privacy breach level. For higher privacy breach levels such as 95% (which could be considered a "plausible denial"

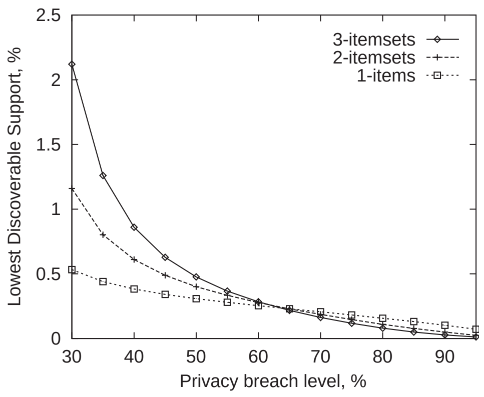

Fig. 1. Lowest discoverable support for different breach levels. Transaction size is 5, five million transactions.

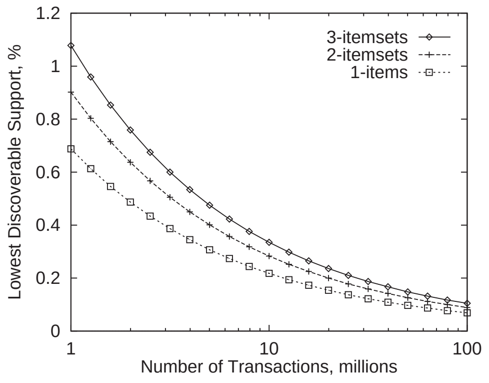

Fig. 2. Lowest discoverable support versus number of transactions. Transaction size is 5, breach level is 50%.

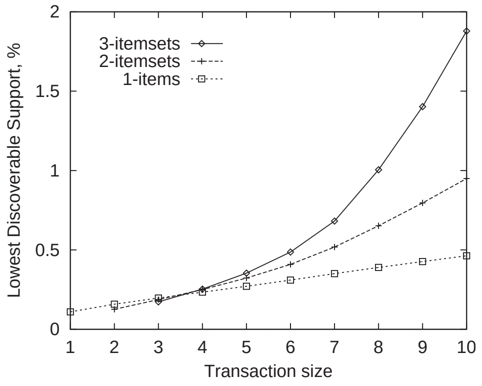

Fig. 3. Lowest discoverable support for different transaction sizes. Five million transactions, breach level is 50%.

breach level), we can discover 3-itemsets at very low supports. For more conservative privacy breach levels such as 50%, the lowest discoverable support is significantly higher. It is interesting to note that at higher breach levels (i.e. weaker randomization) it gets harder to discover 1-itemset supports than 3-itemset supports. This happens because the variance of a 3-itemset predictor depends highly nonlinearly on the amount of false items added while randomizing. When we add fewer false items at higher breach levels, we generate so much fewer false 3-itemset positives than false 1-itemset positives that 3-itemsets get an advantage over single items.

Fig. 2 shows that the lowest discoverable support is roughly inversely proportional to the square root of the number of transactions. Indeed, the lowest discoverable support is defined to be proportional to the standard deviation (square root of the variance) of this support's prediction. If all the partial supports are fixed, the prediction's variance is inversely proportional to the number N of transactions according to Statement 3. In our case, the partial supports depend on N (because the lowest discoverable support does), i.e. they are not fixed; however, this does not appear to affect the variance very significantly (but justifies the word "roughly").

Finally, Fig. 3 shows that transaction size has a significant influence on support discoverability. In fact, for transactions of size 10 and longer, it is typically not possible to make them both breach-safe and simultaneously get useful information for mining transactions. Intuitively, a long transaction contains too much personal information to hide, because it may contain long frequent itemsets whose appearance in the randomized transaction could result in a privacy breach. We have to insert a lot of false items and cut off many true ones to ensure that such a long itemset in the randomized transaction is about as likely to be a false positive as to be a true positive. Such a strong randomization causes an exceedingly high variance in the support predictor for 2- and especially 3-itemsets, since it drives down their probability to "tunnel" through while raising high the probability of a false positive. In both our datasets we discard long transactions. The question of how to safely randomize and mine long transactions is left open.

5.2. *Discoverability of confidence*

The formulae from Section 4.6 can be used for computing the radius δ of the interval

$$[\mathrm{conf}^T(A \Rightarrow B) - \delta,\ \mathrm{conf}^T(A \Rightarrow B) + \delta]$$

around the actual confidence of the rule "$A \Rightarrow B$" such that our estimator $(\mathrm{conf}^T(A \Rightarrow B))_{\mathrm{est}}$ is p%-likely to fall into this interval, where p is any given probability. To find the δ, we should solve the equation $f(\delta) = p$ by means of any suitable numerical algorithm. The δ depends on p, on transaction size m, on sizes of itemsets $|A|$ and $|B|$, on the number N of transactions, and on the parameters of the randomization operator. It also depends on all two-set partial supports for A and B (see Definition 13).

For our confidence graphs we use the interval probability value of 95%. Like we did in the case of support prediction, we define the *lowest discoverable confidence* as the smallest confidence $\text{conf}^T(A \Rightarrow B)$ that is twice as large as the radius of the 95%-interval for its predictor.

In order to set the partial supports for our graphs, we shall use *independence assumption*: all items in $A \cup B$ occur in transactions independently from each other, all items in A have the same frequency, and all items in B have the same frequency. The two frequencies are chosen according to the given $\text{supp}^T(A)$ and $\text{conf}^T(A \Rightarrow B)$. We have also tried *maximum dependence assumption*:

- All items in A and in B are independent from other items in their set and have the same frequency;
- Always if $B \subset t$ then $A \subset t$. Thus, $\text{supp}^T(B) = \text{supp}^T(A) \cdot \text{conf}^T(A \Rightarrow B)$;
- Transactions t are "filled" with items from B in the order of decreasing $|t \cap A|$, observing the above constraints.

Graph in Fig. 5 is computed under maximum dependence assumption, the other graphs are computed under independence assumption. As it turns out, the two assumptions produce similar results, the maximum dependence assumption usually giving a little higher values of lowest discoverable confidence (by several percent or less).

The default parameter settings for the graphs are the same as in Section 5.1, including 5 million transactions, each of size 5 items, and privacy breach level of 50%. The default value for the support of A is set to 2%. We plot the results for four sizes of A and B, as given in the graphs. The cutoff for cut-and-paste is set to 5, except in graphs in Figs. 7 and 8, where the cutoff is set to 7 (for Fig. 7) or to the transaction size (for Fig. 8).

Graphs in Figs. 4 and 5 show how the lowest discoverable confidence depends on the privacy breach level; graph on Fig. 6 shows its dependence on the support of the set A; graphs on

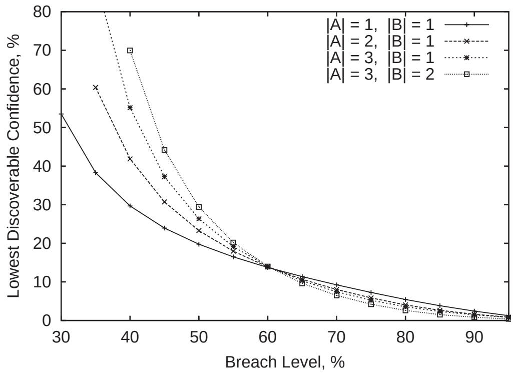

Fig. 4. Lowest discoverable confidence for different breach levels. Five million transactions, transaction size is 5, $\text{supp}^T(A) = 2\%$.

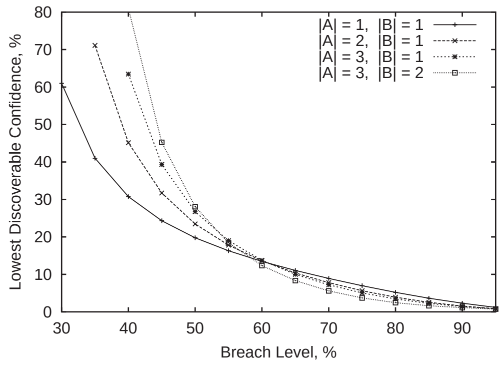

Fig. 5. Lowest discoverable confidence for different breach levels, under "maximum dependence" assumption. Five million transactions, transaction size is 5, $\text{supp}^T(A) = 2\%$.

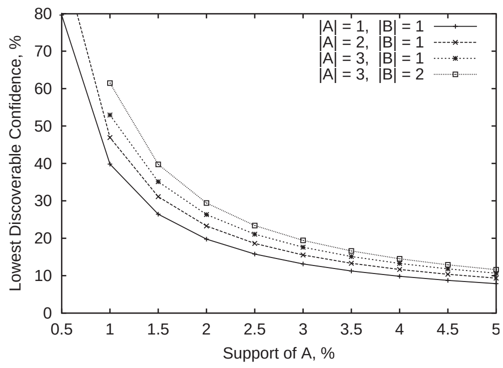

Fig. 6. Lowest discoverable confidence for different values of $\text{supp}^T(A)$. Five million transactions, transaction size is 5, breach level is 50%.

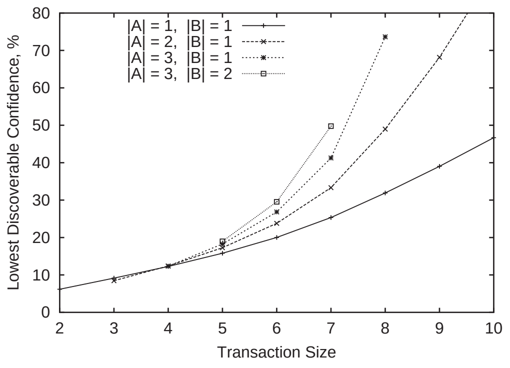

Fig. 7. Lowest discoverable confidence for different transaction sizes. Five million transactions, breach level is 50%, $\mathrm{supp}^T(A) = 2\%$, cutoff is 7.

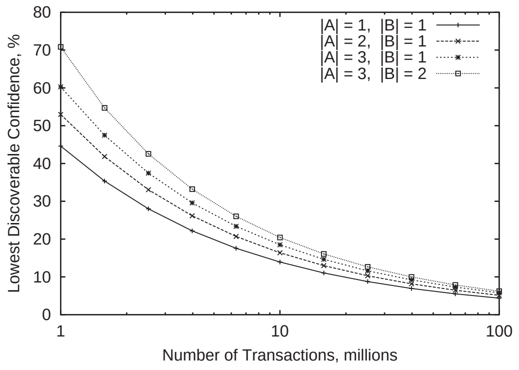

Fig. 9. Lowest discoverable confidence for different numbers of transactions. Transaction size is 5, breach level is 50%, $\mathrm{supp}^T(A) = 2\%$.

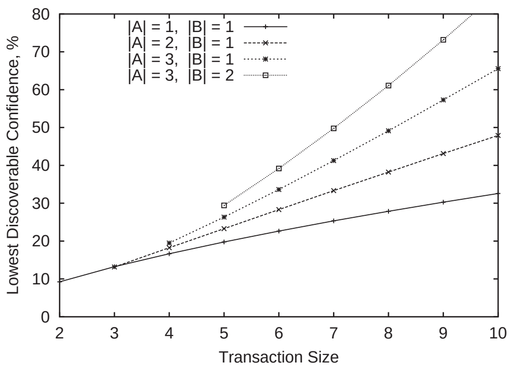

Fig. 8. Lowest discoverable confidence for different transaction sizes. Five million transactions, breach level is 50%, $\mathrm{supp}^T(A) = 2\%$, cutoff equals transaction size.

Figs. 7 and 8 demonstrate its dependence on transaction size; finally, graph on Fig. 9 shows the influence of the number of transactions. The tendencies here are similar to the ones for the lowest discoverable support, with some additional sensitivity to the support of A since its predictor appears in the denominator of the confidence estimator. The graphs demonstrate that our estimator can be used, as long as the confidence being predicted is not low and the support of A is predicted well above zero.

5.3. The datasets

We experimented with two "real-life" datasets. The `soccer` dataset is generated from the click-stream log of the 1998 World Cup web site, which is publicly available at `ftp://researchsmp2.cc.vt.edu/pub/worldcup/`.[5] We scanned the log and produced a transaction file, where each transaction is a session of access to the site by a client. Each item in the transaction is a web request. Not all web requests were turned into items; to become an item, the request must satisfy the following:

(1) Client's request method is `GET`;
(2) Request status is `OK`;
(3) File type is `HTML`.

A session starts with a request that satisfies the above properties, and ends when the last click from this ClientID timeouts. The timeout is set as 30 minutes. All requests in a session have the same ClientID. The `soccer` transaction file was then processed further: we deleted from all transactions the items corresponding to the French and English front page frames, and then we deleted all empty transactions and all transactions of size above 10. The resulting `soccer` dataset consists of 6,525,879 transactions, distributed as shown in Fig. 10.

The `mailorder` dataset is the same as that used in [32]. The original dataset consisted of around 2.9 million transactions, 15,836 items, and around 2.62 items per transaction. Each transaction was

[5] M. Arlitt and T. Jin, "1998 World Cup Web Site Access Logs", August 1998. Available at `http://www.acm.org/sigcomm/ITA/`

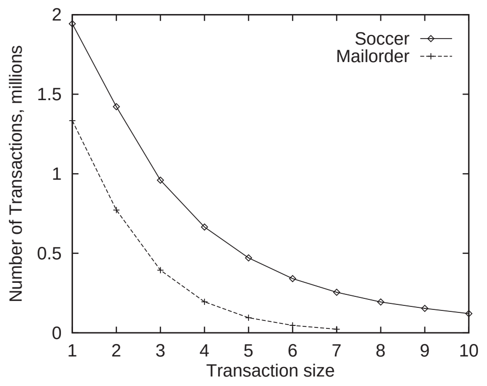

Fig. 10. Number of transactions for each transaction size in the `soccer` and `mailorder` datasets.

the set of items purchased in a single mail order. However, very few itemsets had reasonably high supports. For instance, there were only two 2-itemsets with support $\geqslant 0.2\%$, only five 3-itemsets with support $\geqslant 0.05\%$. Hence we decided to substitute all items by their parents in the taxonomy, which had reduced the number of items from 15836 to 96. It seems that, in general, moving items up the taxonomy is a natural thing to do for preserving privacy without losing aggregate information. We also discarded all transactions of size $\geqslant 8$ (which was less than 1% of all transactions) and finally obtained a dataset containing 2,859,314 transactions (Fig. 10).

5.4. The results

We report the results for both datasets at a minimum support that is close to the lowest discoverable support, in order to show the resilience of our algorithm even at these very low support levels. We targeted a conservative breach level of 50%, so that, given a randomized transaction, for any item in the transaction it is at least as likely that someone did not buy that item (or access a web page) as that they did buy that item.

We used cut-and-paste randomization (see Definition 8) that has only two parameters, randomization level and cutoff, per each transaction size. We chose a cutoff of 7 for our experiments as a good compromise between privacy and discoverability. Given the values of maximum supports, we then used the methodology from Section 4.4 to find the lowest randomization level such that the breach probability (for each itemset size) is still below the desired breach level. The actual parameters (K_m is the cutoff, ρ_m is the randomization level for transaction size m) for `soccer` were:

m	1	2	3	4	5	6	7	8	9	10
K_m	7	7	7	7	7	7	7	7	7	7
$\rho_m\%$	4.7	16.8	21.4	32.2	35.3	42.9	46.1	42.0	40.9	39.5

and for `mailorder` were:

m	1	2	3	4	5	6	7
K_m	7	7	7	7	7	7	7
$\rho_m\%$	8.9	20.4	25.0	33.4	43.5	50.5	59.2

Table 1 shows what happens if we mine itemsets from both randomized and nonrandomized files and then compare the results. We can see that, even for a low minimum support of 0.2%, most of the itemsets are mined correctly from the randomized file. There are comparatively few false positives (itemsets wrongly included into the output) and even fewer false drops (itemsets wrongly omitted). The predicted sigma for 3-itemsets ranges in 0.066–0.07% for `soccer` and in 0.047–0.048% for `mailorder`; for 2- and 1-itemsets sigmas are even less.

One might be concerned about the true supports of the false positives. Since we know that there are *many* more low-supported itemsets than there are highly supported, we might wonder whether most

Table 1
Results on real datasets

Itemset size	True itemsets	True positives	False drops	False positives
(a) Mailorder, 0.2% minimum support				
1	65	65	0	0
2	228	212	16	28
3	22	18	4	5
(b) Soccer, 0.2% minimum support				
1	266	254	12	31
2	217	195	22	45
3	48	43	5	26

Table 2
Analysis of false drops

Size	Itemsets	Predicted support			
		<0.1	0.1–0.15	0.15–0.2	⩾0.2
(a) Mailorder, ⩾0.2% true support					
1	65	0	0	0	65
2	228	0	1	15	212
3	22	0	1	3	18
(b) Soccer, ⩾0.2% true support					
1	266	0	2	10	254
2	217	0	5	17	195
3	48	0	1	4	43

Table 3
Analysis of false positives

Size	Itemsets	True support			
		<0.1	0.1–0.15	0.15–0.2	⩾0.2
(a) Mailorder, ⩾0.2% predicted support					
1	65	0	0	0	65
2	240	0	0	28	212
3	23	1	2	2	18
(b) Soccer, ⩾0.2% predicted support					
1	285	0	7	24	254
2	240	7	10	28	195
3	69	5	13	8	43

of the false positives are outliers, that is, have true support near zero. We have indeed seen outliers; however, it turns out that most of the false positives are not so far off. Tables 2 and 3 show that usually the true supports of false positives, as well as the predicted supports of false drops, are closer to 0.2% than to zero. This good news demonstrates the promise of randomization as a practical privacy-preserving approach.

Privacy analysis: We evaluate privacy breaches, i.e., the conditional probabilities from Definition 4, as follows. We count the occurrences of an itemset in a randomized transaction and its sub-items in the corresponding nonrandomized transaction. For example, assume an itemset $\{a, b, c\}$ occurs 100 times in the randomized data among transactions of length 5. Out of these 100 occurrences, 60 of the corresponding original transactions had the item b. We then say that this itemset caused a 60% privacy breach for transactions of length 5, since for these 100 randomized transactions, we estimate with 60% confidence that the item b was present in the original transaction.

Out of all sub-items of an itemset, we choose the item that causes the worst privacy breach. Then, for each combination of transaction size and itemset size, we compute over all frequent[6] itemsets the worst and the average value of this breach level. Finally, we pick the itemset size that gave the worst value for each of these two values.

Table 4 shows the results of the above analysis. To the left of the semicolon is the itemset size that was the worst. For instance, for all transactions of length 5 for `soccer`, the worst average breach was with 4-itemsets (43.9% breach), and the worst breach was with a 5-itemset (49.7% breach). We can see that, apart from fluctuations, the 50% level is observed everywhere except of a little "slip" for 9- and 10-item transactions of `soccer`. The "slip" resulted from our decision to use the corresponding maximal support information only for itemset sizes up to 7 (while computing randomization parameters). In another experiment, when we used the maximal supports for itemset sizes up to 10, we compensated for the increased privacy requirements with the following cut-and-paste parameters:

m	1	2	3	4	5	6	7	8	9	10
K_m	7	7	7	7	7	7	7	10	10	10
ρ_m%	4.7	16.8	21.4	32.2	35.3	42.9	46.1	52.5	57.8	59.7

Also, for this experiment we have used a different (better quality) pseudorandom number generator.[7] The results of this `soccer` dataset experiment are given in Table 5. Some details on false drops are in Table 6, on false positives are in Table 7. Privacy analysis is given in Table 8. We can see that the "slip" for 9- and 10-item transactions is no longer observed.

[6] If there are no frequent itemsets of certain size, we pick the itemsets with the highest support.

[7] "Mersenne Twister", http://www.math.keio.ac.jp/matumoto/emt.html

Table 4
Actual privacy breaches

	Transaction size									
	1	2	3	4	5	6	7	8	9	10
Soccer										
Worst average:	1: 4.4%	2: 20.2%	3: 39.2%	4: 44.5%	4: 43.9%	4: 37.5%	4: 36.2%	4: 38.7%	8: 51.0%	10: 49.4%
Worst of the worst:	1: 45.5%	2: 45.4%	3: 53.2%	4: 49.8%	5: 49.7%	5: 42.7%	5: 41.8%	5: 44.5%	9: 66.2%	10: 65.6%
Mailorder										
Worst average:	1: 12.0%	2: 27.5%	3: 48.4%	4: 51.5%	5: 51.7%	5: 51.9%	6: 49.8%			
Worst of the worst:	1: 47.6%	2: 51.9%	3: 53.6%	4: 53.1%	5: 53.6%	6: 55.4%	7: 51.9%			

Table 5
Soccer results, "no-slip" experiment. Soccer, 0.2% minimum support

Itemset size	True itemsets	True positives	False drops	False positives
1	266	250	16	33
2	217	197	20	38
3	48	39	9	27

Table 6
Analysis of false drops, "no-slip" experiment. Soccer, ⩾0.2% true support

Size	Itemsets	Predicted support			
		<0.1	0.1–0.15	0.15–0.2	⩾0.2
1	266	0	1	15	250
2	217	0	4	16	197
3	48	0	0	9	39

Table 7
Analysis of false positives, "no-slip" experiment. Soccer, ⩾0.2% Predicted support

Size	Itemsets	True support			
		<0.1	0.1–0.15	0.15–0.2	⩾0.2
1	283	0	7	26	250
2	235	4	7	27	197
3	66	6	11	10	39

Summary: Despite choosing a conservative privacy breach level of 50%, and further choosing a minimum support around the lowest discoverable support, we were able to successfully find most of the frequent itemsets, with relatively small numbers of false drops and false positives.

6. Conclusions

In this paper, we have presented three key contributions toward mining association rules while preserving privacy. First, we pointed out the problem of privacy breaches, presented their formal definitions and proposed a natural solution. Second, we gave a sound mathematical treatment for a class of randomization algorithms and derived formulae for support and variance prediction, and showed how to incorporate these formulae into mining algorithms. Finally, we presented experimental results that validated the algorithm in practice by applying it to two real datasets from different domains.

We conclude by raising three interesting questions for future research. Our approach deals with a restricted (albeit important) class of privacy breaches; can we extend it to cover other kinds of breaches and other assumptions on the dataset? Second, what are the theoretical limits on discoverability for a given level of privacy (and vice versa)? Finally, can we combine randomization and cryptographic protocols to get the strengths of both without the weaknesses of either?

Table 8
Actual privacy breaches, "no-slip" experiment. Soccer

	Transaction size									
	1	2	3	4	5	6	7	8	9	10
Worst Average:	1: 4.4%	2: 20.2%	3: 39.0%	4: 35.8%	4: 36.0%	4: 31.3%	3: 30.6%	3: 31.6%	3: 30.1%	3: 30.5%
Worst of the Worst:	1: 45.6%	2: 45.6%	3: 52.5%	4: 49.9%	5: 50.4%	5: 44.0%	7: 44.3%	7: 45.4%	9: 40.4%	10: 42.3%

Appendix A. Proofs

A.1. Proof of Statement 1

Each coordinate $N \cdot s'_{l'}$ of the vector in (4) is, by definition of partial supports, just the number of transactions in the randomized sequence T' that have intersections with A of size l'. Each randomized transaction t' contributes to one and only one coordinate $N \cdot s'_{l'}$, namely to the one with $l' = \#(t' \cap A)$. Since we are dealing with a per-transaction randomization, different randomized transactions contribute independently to one of the coordinates. Moreover, by item-invariance assumption, the probability that a given randomized transaction contributes to the coordinate number l' depends only on the size of the original transaction t (which equals m) and the size l of intersection $t \cap A$. This probability equals $p[l \to l']$.

So, for all transactions in T that have intersections with A of the same size l (and there are $N \cdot s_l$ such transactions) the probabilities of contributing to various coordinates $N \cdot s'_{l'}$ are the same. We can split all N transactions into $k+1$ groups according to their intersection size with A. Each group contributes to the vector in (4) as a multinomial distribution with probabilities

$$(p[l \to 0], p[l \to 1], \ldots, p[l \to k]),$$

independently from the other groups. Therefore, the vector in (4) is a sum of $k+1$ independent multinomials. Now it is easy to compute both expectation and covariance.

For a multinomial distribution $(X_0, X_1, \ldots, X_k)$ with probabilities $(p_0, p_1, \ldots, p_k)$, where $X_0 + X_1 + \cdots + X_k = n$, we have $\mathbf{E}X_i = n \cdot p_i$ and

$$\begin{aligned}\mathbf{Cov}(X_i, X_j) &= \mathbf{E}(X_i - np_i)(X_j - np_j)\\ &= n \cdot (p_i \delta_{i=j} - p_i p_j).\end{aligned}$$

In our case, $X_i = l$'s part of $N \cdot s'_i$, $n = N \cdot s_l$, and $p_i = p[l \to i]$. For a sum of independent multinomial distributions, their expectations and covariances add together:

$$\mathbf{E}(N \cdot s'_{l'}) = \sum_{l=0}^{k} N \cdot s_l \cdot p[l \to l'],$$

$$\begin{aligned}\mathbf{Cov}(N \cdot s'_i, N \cdot s'_j) = \sum_{l=0}^{k} N \cdot s_l \cdot (&p[l \to i] \cdot \delta_{i=j}\\ &- p[l \to i] \cdot p[l \to j])\end{aligned}$$

Thus, after dividing by an appropriate power of N, the formulae in the statement are proven. □

A.2. Proof of Statement 2

We are given a transaction $t \in T$ and an itemset $A \subseteq \mathcal{I}$, such that $|t| = m$, $|A| = k$, and $\#(t \cap A) = l$. In the beginning of randomization, a number j is selected with distribution $\{p_m[j]\}$, and this is what the first summation takes care of. Now assume that we retain exactly j items of t, and discard $m - j$ items.

Suppose there are q items from $t \cap A$ among the retained items. How likely is this? Well, there are $\binom{m}{j}$ possible ways to choose j items from transaction t; and there are $\binom{l}{q}\binom{m-l}{j-q}$ possible ways to choose q items from $t \cap A$ and $j - q$ items from $t \backslash A$. Since all choices are equiprobable, we get $\binom{l}{q}\binom{m-l}{j-q} / \binom{m}{j}$ as the probability that exactly q A-items are retained.

To make t' contain exactly l' items from A, we have to get additional $l' - q$ items from $A \backslash t$. We know that $\#(A \backslash t) = k - l$, and that any such item has probability ρ to get into t'. The last terms in (8) immediately follow. Summation bounds restrict q

to its actually possible (= nonzero probability) values. □

A.3. Proof of Statement 3

Let us denote

$$\vec{p}_l := (p[l\to 0], p[l\to 1], \ldots, p[l\to k])^{\mathrm{T}},$$
$$\vec{q}_l := (q[l\leftarrow 0], q[l\leftarrow 1], \ldots, q[l\leftarrow k])^{\mathrm{T}}.$$

Since $PQ = QP = I$ (where I is the identity matrix), we have

$$\sum_{l=0}^{k} p[l\to i]q[l\leftarrow j] = \sum_{l'=0}^{k} p[i\to l']q[j\leftarrow l'] = \delta_{i=j}.$$

Notice also, from (7), that matrix $D[l]$ can be written as

$$D[l] = \mathrm{diag}(\vec{p}_l) - \vec{p}_l\vec{p}_l^{\mathrm{T}},$$

where $\mathrm{diag}(\vec{p}_l)$ denotes the diagonal matrix with $\vec{p}_l$-coordinates as its diagonal elements. Now it is easy to see that

$$\tilde{s} = \vec{q}_k^{\mathrm{T}}\vec{s}' = \sum_{l'=0}^{k} q[k\leftarrow l']\cdot s'_{l'};$$

$$\begin{aligned}\mathbf{Var}\,\tilde{s} &= \frac{1}{N}\sum_{l=0}^{k} s_l\vec{q}_k^{\mathrm{T}}D[l]\vec{q}_k\\ &= \frac{1}{N}\sum_{l=0}^{k} s_l\vec{q}_k^{\mathrm{T}}(\mathrm{diag}(\vec{p}_l) - \vec{p}_l\vec{p}_l^{\mathrm{T}})\vec{q}_k\\ &= \frac{1}{N}\sum_{l=0}^{k} s_l(\vec{q}_k^{\mathrm{T}}\mathrm{diag}(\vec{p}_l)\vec{q}_k - (\vec{p}_l^{\mathrm{T}}\vec{q}_k)^2)\\ &= \frac{1}{N}\sum_{l=0}^{k} s_l\left(\sum_{l'=0}^{k} p[l\to l']q[k\leftarrow l']^2 - \delta_{l=k}\right);\end{aligned}$$

$$\begin{aligned}(\mathbf{Var}\,\tilde{s})_{\mathrm{est}} &= \frac{1}{N}\sum_{l=0}^{k}(\vec{q}_l^{\mathrm{T}}\vec{s}')\left(\sum_{l'=0}^{k} p[l\to l']\right.\\ &\qquad\left.\times\, q[k\leftarrow l']^2 - \delta_{l=k}\right)\\ &= \frac{1}{N}\sum_{j=0}^{k} s'_j\left(\sum_{l,l'=0}^{k} q[l\leftarrow j]p[l\to l']\right.\\ &\qquad\left.\times\, q[k\leftarrow l']^2 - \sum_{l=0}^{k}\delta_{l=k}q[l\leftarrow j]\right)\\ &= \frac{1}{N}\sum_{j=0}^{k} s'_j\\ &\qquad\times\left(\sum_{l'=0}^{k}\delta_{l'=j}q[k\leftarrow l']^2 - q[k\leftarrow j]\right)\\ &= \frac{1}{N}\sum_{j=0}^{k} s'_j(q[k\leftarrow j]^2 - q[k\leftarrow j]).\end{aligned}$$

A.4. Proof of Statement 4

We prove the left formula in (13) first, and then show that the right one follows from the left one. Consider $N\cdot\Sigma_l$; it equals

$$\begin{aligned}N\cdot\Sigma_l &= N\cdot\sum_{C\subseteq A,|C|=l}\mathrm{supp}^{\mathrm{T}}(C)\\ &= \sum_{C\subseteq A,|C|=l}\#\{t_i\in T\mid C\subseteq t_i\}\\ &= \sum_{i=1}^{N}\#\{C\subseteq A\mid |C| = l, C\subseteq t_i\}.\end{aligned}$$

In other words, each transaction t_i should be counted as many times as many different l-sized subsets $C\subseteq A$ it contains. From simple combinatorics we know that if $j = \#(A\cap t_i)$ and $j\geqslant l$, then t_i contains $\binom{j}{l}$ different l-sized subsets of A. Therefore,

$$\begin{aligned}N\cdot\Sigma_l &= \sum_{i=l}^{N}\binom{\#(A\cap t_i)}{l}\\ &= \sum_{j=1}^{k}\binom{j}{l}\cdot\#\{t_i\in T\mid\#(A\cap t_i) = j\}\\ &= \sum_{j=l}^{N}\binom{j}{l}N\cdot s_j,\end{aligned}$$

and the left formula is proven. Now we can check the right formula just by replacing the Σ_j's according to the left formula. We have

$$
\begin{aligned}
\sum_{j=l}^{k}(-1)^{j-l}\binom{j}{l}\Sigma_j &= \sum_{j=l}^{k}(-1)^{j-l}\binom{j}{l}\sum_{q=j}^{k}\binom{q}{j}s_q\\
&= \sum_{l\leqslant j\leqslant q\leqslant k}(-1)^{j-l}\binom{j}{l}\binom{q}{j}s_q\\
&= \sum_{q=l}^{k} s_q\sum_{j=l}^{q}(-1)^{j-l}\binom{j}{l}\binom{q}{j}\\
&= \sum_{q=l}^{k} s_q\sum_{j'=0}^{q-l}(-1)^{j'}\frac{(j'+l)!}{l!j'!}\,\frac{q!}{(j'+l)!(q-j'-l)!}\\
&= \sum_{q=l}^{k} s_q\cdot\frac{q!}{l!(q-l)!}\sum_{j'=0}^{q-l}(-1)^{j'}\frac{(q-l)!}{j'!(q-l-j')!}\\
&= \sum_{q=l}^{k} s_q\binom{q}{l}\sum_{j'=0}^{q-l}(-1)^{j'}\binom{q-l}{j'} = s_l,
\end{aligned}
$$

since the sum $\sum_{j'=0}^{q-l}(-1)^{j'}\binom{q-l}{j'}$ is zero whenever $q-l>0$.

To prove that matrix P becomes lower triangular after the transformation from $\vec{s}$ and $\vec{s}'$ to $\vec{\Sigma}$ and $\vec{\Sigma}'$, let us find how $\mathbf{E}\vec{\Sigma}'$ depends on $\vec{\Sigma}$ using the definition (12).

$$
\begin{aligned}
\mathbf{E}\Sigma'_{l'} &= \sum_{C\subseteq A,|C|=l'} \mathbf{E}\,\mathrm{supp}^{T'}(C)\\
&= \sum_{C\subseteq A,|C|=l'}\ \sum_{l=0}^{l'} p_{l'}^{m}[l\to l']\cdot \mathrm{supp}_l^{T}(C)\\
&= \sum_{C\subseteq A,|C|=l'}\ \sum_{l=0}^{l'} p_{l'}^{m}[l\to l']\\
&\quad\times\sum_{j=l}^{l'}(-1)^{j-l}\binom{j}{l}\Sigma_j(C,T)\\
&= \sum_{j=0}^{l'}\underbrace{\sum_{l=0}^{j}(-1)^{j-l}\binom{j}{l}p_{l'}^{m}[l\to l']}_{c_{l'j}}\\
&\quad\times\sum_{C\subseteq A,|C|=l'}\Sigma_j(C,T)\\
&= \sum_{j=0}^{l} c_{l'j}\sum_{C\subseteq A,|C|=l'}\ \sum_{B\subseteq C,|B|=j}\mathrm{supp}^T(B)\\
&= \sum_{j=0}^{l'} c_{l'j}\sum_{B\subseteq A,|B|=j}\#\{C\mid B\subseteq C\subseteq A,|C|\\
&= l'\}\cdot\mathrm{supp}^T(B)\\
&= \sum_{j=0}^{l'} c_{l'j}\sum_{B\subseteq A,|B|=j}\binom{k-j}{l'-j}\mathrm{supp}^T(B)\\
&= \sum_{j=0}^{l'} c_{l'j}\binom{k-j}{l'-j}\cdot\Sigma_j.
\end{aligned}
$$

Now it is clear that only the lower triangle of the matrix can have nonzeros. □

References

[1] Office of the Information and Privacy Commissioner, Ontario, Data Mining: Staking a Claim on Your Privacy, January 1998.

[2] The Economist, The End of Privacy, May 1999.

[3] European Union, Directive on Privacy Protection, October 1998.

[4] Time, The Death of Privacy, August 1997.

[5] Business Week, Privacy on the Net, March 2000.

[6] L. Cranor, J. Reagle, M. Ackerman, Beyond concern: understanding net users' attitudes about online privacy, Technical Report TR 99.4.3, AT&T Labs-Research, April 1999.

[7] L.F. Cranor (Ed.), Special Issue on Internet Privacy, Commun. ACM 42(2) (1999).

[8] A. Westin, E-commerce and privacy: what net users want, Techical Report, Louis Harris & Associates, June 1998.

[9] A. Westin, Privacy concerns & consumer choice, Technical Report, Louis Harris & Associates, December 1998.

[10] A. Westin, Freebies and privacy: what net users think, Technical Report, Opinion Research Corporation, July 1999.

[11] C. Clifton, D. Marks, Security and privacy implications of data mining, in: ACM SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery, 1996, pp. 15–19.

[12] V. Estivill-Castro, L. Brankovic, Data swapping: balancing privacy against precision in mining for logic rules, in: M. Mohania, A. Tjoa (Eds.), Data Warehousing and Knowledge Discovery DaWaK-99, Springer-Verlag Lecture Notes in Computer Science, Vol. 1676, Springer, Berlin, 1999, pp. 389–398.

[13] K. Thearling, Data mining and privacy: A conflict in making, DS*.

[14] R. Agrawal, Data mining: crossing the chasm, in: Fifth International Conference on Knowledge Discovery in Databases and Data Mining, San Diego, California, 1999, available from http://www.almaden.ibm.com/cs/quest/papers/kdd99_chasm.ppt.

[15] R. Agrawal, R. Srikant, Privacy preserving data mining, in: ACM SIGMOD Conference on Management of Data, Dallas, TX, 2000, pp. 439–450.

[16] R. Conway, D. Strip, Selective partial access to a database, in: Proceedings of the ACM Annual Conference, 1976, pp. 85–89.

[17] L. Breiman, J.H. Friedman, R.A. Olshen, C.J. Stone, Classification and Regression Trees, Wadsworth, Belmont, 1984.

[18] J.R. Quinlan, Induction of decision trees, Mach. Learn. 1 (1986) 81–106.

[19] T.M. Mitchell, Machine Learning, McGraw-Hill, New York, 1997 (chapter 6).

[20] D. Agrawal, C.C. Aggarwal, On the design and quantification of privacy preserving data mining algorithms, in: Proceedings of the 20th ACM Symposium on Principles of Database Systems, Santa Barbara, CA, 2001, pp. 247–255.

[21] R. Agrawal, T. Imielinski, A. Swami, Mining association rules between sets of items in large databases, in: Proceedings of the ACM SIGMOD Conference on Management of Data, Washington, DC, 1993, pp. 207–216.

[22] N.R. Adam, J.C. Wortman, Security-control methods for statistical databases, ACM Comput. Surv. 21 (4) (1989) 515–556.

[23] A. Shoshani, Statistical databases: characteristics, problems and some solutions, in: Proceedings of the Eighth International Conference on Very Large Databases, Mexico City, Mexico, 1982, pp. 208–213.

[24] S. Warner, Randomized response: a survey technique for eliminating evasive answer bias, J. Am. Stat. Assoc. 60 (309) (1965) 63–69.

[25] J. Vaidya, C.W. Clifton, Privacy preserving association rule mining in vertically partitioned data, in: Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Edmonton, Canada, 2002. URL citeseer.nj.nec.com/492031.html.

[26] Y. Lindell, B. Pinkas, Privacy preserving data mining, in: CRYPTO, 2000, pp. 36–54. URL citeseer.nj.nec.com/lindell00privacy.html.

[27] A. Evfimievski, R. Srikant, R. Agrawal, J. Gehrke, Privacy preserving mining of association rules, in: Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery in Databases and Data Mining, Edmonton, Alberta, Canada, 2002, pp. 217–228.

[28] S.J. Rizvi, J.R. Haritsa, Maintaining data privacy in association rule mining, in: Proceedings of the 28th International Conference on Very Large Data Bases, Hong Kong, China, 2002.

[29] A. Evfimievski, J. Gehrke, R. Srikant, Limiting privacy breaches in privacy preserving data mining, in: Proceedings of the 22nd ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS 2003), San Diego, California, USA, 2003, pp. 211–222.

[30] R. Agrawal, H. Mannila, R. Srikant, H. Toivonen, A.I. Verkamo, Fast discovery of association rules, in: U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, R. Uthurusamy (Eds.), Advances in Knowledge Discovery and Data Mining, AAAI/MIT Press, Cambridge, 1996, pp. 307–328 (Chapter 7).

[31] R. Bayardo, Efficiently mining long patterns from databases, in: Proceedings of the ACM SIGMOD Conference on Management of Data, Seattle, WA, 1998.

[32] R. Agrawal, R. Srikant, Fast algorithms for mining association rules, Research Report RJ 9839, IBM Almaden Research Center, San Jose, CA, June 1994.