DATA SHARING TAXONOMY RECORDS FOR SECURITY CONSERVATION

Rajeswari Chandrasekaran[1] and Chandrasekaran Nammalwar[2]

[1,2]Faculty of Computing,
Botho University, Gaborone, Botswana

ABSTRACT

Here, we discuss the Classification is a fundamental problem in data analysis. Training a classifier requires accessing a large collection of data. Releasing person-specific data, such as customer data or patient records, may pose a threat to an individual's privacy. Even after removing explicit identifying information such as Name and SSN, it is still possible to link released records back to their identities by matching some combination of non identifying attributes such as {Sex,Zip,Birthdate}. A useful approach to combat such linking attacks, called k-anonymization is anonymizing the linking attributes so that at least k released records match each value combination of the linking attributes. Our goal is to find a k-anonymization which preserves the classification structure. Experiments of real-life data show that the quality of classification can be preserved even for highly restrictive anonymity requirements.

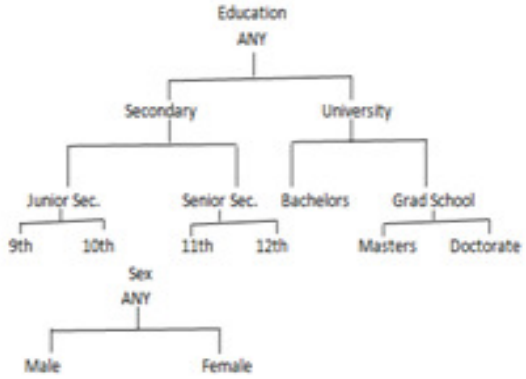
KEYWORDS

Privacy protection, Anonymity, Security integrity, Data mining, classification, Data sharing.

1. INTRODUCTION

DATA sharing in today's globally networked systems poses a threat to individual privacy and organizational confidentiality. An example by Samarati [2] shows that linking medication records with a voter list can uniquely. Identify a person's name and medical information. New privacy acts and legislations are recently enforced in many countries. In 2001, Canada launched the Personal Information Protection and Electronic Document Act [3] to protect a wide spectrum of information, such as age, race, income, evaluations, and even intentions to acquire goods or services. This information spans a considerable portion of many databases. Government agencies and companies have to revise their systems and practices to fully comply with this act in three years. Consider a table T about a patient's information on Birthplace, Birth year, Sex and Diagnosis. If a description on fBirthplace; Birth year; Sexg is so specific that not many people match it, releasing the table may lead to linking a unique record to an external record with explicit identity, thus identifying the medical condition and compromising the privacy rights of the individual [2]. Suppose that the attributes Birthplace, Birth year, Sex and Diagnosis must be released (say, to some health research institute for research purposes). One way to prevent such linking is masking the detailed information of these attributes as follows:

Jae-Kwang Lee et al. (Eds) : CCSEA, AIFU, DKMP, CLOUD, EMSA, SEA, SIPRO - 2017
pp. 49– 58, 2017. © CS & IT-CSCP 2017 DOI : 10.5121/csit.2017.70205



1. If there is a taxonomical description for a categorical attribute (for example, Birthplace), we can generalize a specific value description into a less specific but semantically consistent description. For example, we can generalize the cities San Francisco, San Diego, and Berkeley into the corresponding state California.

2. If there is not taxonomical description for a categorical attribute, we can suppress a value description to a "null value" demoted? For example, we can suppress San Francisco and San Diego to the null value? While keeping Berleley.

3. If the attribute is a continuous attribute (for example, Birth year), we can discredited the range of the attribute into a small number of intervals. For example, we can replace specific Birth year values from 1961 to 1965 with an interval [1961-1966].

By applying such masking operations, the information on fBirthplace; Birth year; Sexg is made less specific, and a person tends to match more records. For example, a male born in San Francisco in 1962 will match all records that have the values HCA; {1961-1966}; Mi; clearly, not all matched records correspond to the person. Thus, the masking operation makes it more difficult to tell whether an individual actually has the diagnosis in the matched records. Protecting privacy is one goal. Making the released data useful to data analysis is another goal. In this paper, we consider classification analysis [4].

2. PROBLEM STATEMENT

A data provider wants to release a person-specific table T (d1;…….Dm; Class) to the public for modeling the class label Class. Each Di is either a categorical or a continuous attribute. A record has the form hv1… vm;clsi, where vi is a domain value for Di and cls is a class for Class. Att(v) denotes the attribute of a value v. The data provider also wants to protect against linking an individual to sensitive information either within or outside T through some identifying attributes, called QID. A sensitive linking occurs if some value of the QID identifies a "small" number of records in T. This requirement is formally defined below. Definition 1 (anonymity requirement). Consider p QIDs QID1;……;QIDp on T. a(qidi) denotes the number of data records in T that share the value qidi of QIDi. The anonymity of QIDi, denoted A(QIDi), is the smallest a(qidi) for any value qidi on QIDi. A table T satisfies the anonymity requirement fhQID1; k1i…hQIDp; kpig if A(QIDi) _ki for 1_i_p, where ki is the anonymity threshold on QIDi specified by the data provider. It is not hard to see that if QIDj is a subset of QIDi, A(QIDi)_A(QIDj). Therefore, if kj_ki, A(QIDi)_ki implies A(QIDj)_kj, and hQIDj; kji can be removed in the presence of hQIDi; kii. Following a similar argument, to prevent a linking through any QID, that is, any subset of

attributes in QID1 [----] QIDp, the single QID [---]QIDp and $k^{1/4}$ maxfkjg, can be specified. However, a table satisfying fhQID1; k1i…hQIDp; kpig does not have to satisfy hQID; ki.

2.1 Masking Operations

A. Generalize Dj if Dj is a categorical attribute with a taxonomy tree. A leaf node represents a domain value and a parent node represents a less specific value. Fig. 2 shows a taxonomy tree for Education. A generalized Dj can be viewed as a "cut" through its taxonomy tree. A cut of a tree is a subset of values in the tree, denoted Cutj, which contains exactly one value on each root-to-leaf path. This type of generalization does not suffer from the interpretation difficulty discussed in Section 1.

B. Suppress Dj if Dj is a categorical attribute with not taxonomy tree. The suppression of a value on Dj means replacing all occurrences of the value with the special value?j. All suppressed values on Dj are represented by the same?j, which is treated as a new Supj to denote the set of values suppressed by ?j. This type of suppression is at the value level in that Supj is in general, a subset of the values in the attribute Dj.

C. Discredited Dj if Dj is a continuous attribute. The discretization of a value v on Dj means replacing all occurrences of v with an interval containing the value. Our algorithm dynamically grows a taxonomy tree for intervals at runtime, where each node represents an interval, and each nonleaf node has two child nodes representing some "optional" binary split of the parent interval. More details will be discussed in Section 3. A discredited Dj can be represented by the set of intervals, denoted Intj, corresponding to the leaf nodes in the dynamically grown taxonomy tree of Dj.

Definition 2 (Anonymity for Classification). Given a table T, an anonymity requirement fhQID1; k1i…hQIDp; kpig and an optional taxonomy tree for each categorical attribute contained in [QIDi mask T on the attributes [QIDi to satisfy the anonymity requirement while preserving the classification structure in the data (that is, the masked table remains useful for classifying the Class column). A masked table T can be represented by h[Cutj; [Supj;] Inti; where Cutj, Supj, and Intj are defined as above. If the masked T satisfied the anonymity requirement, h[Cutj; [supj; [Intji is called a solution set.

3. SEARCH METHODS

A table T can be masked by a sequence of refinements starting from the most masked state in which each attribute is either generalized to the topmost value, suppressed to the special value? Our method iteratively refines a masked value selected from the current set of cuts, suppressed values, and intervals, until violating the anonymity requirement. Each refinement increases the information and decreases the anonymity since records with specific values are more distinguishable. The key is selecting the "best" refinement at each step with both impacts considered.

3.1. Modifications

Below, we formally describe the notion of refinement on different types of attributes Dj 2[QIDi and define a selection criterion for a single refinement.

3.1.1 Refinement for Generalization

Consider a categorical attribute Dj with a user-specified taxonomy tree. Let child(v) be the set of child values of v in a user-specified taxonomy tree. A refinement, written v! Child (v) replaces the parent value v with the child value in child (v) that generalized the domain values in each (generalized) record.

3.1.2 Refinement for Suppression

For a categorical attribute Dj without taxonomy tree, a refinement! fv;? Jg refers to disclosing one value v from the set of suppressed values Supj. Let R?j denotes the set of suppressed records that currently contain? j. Disclosing v means replacing? j with v in all records in R?j that originally contain v.

3.1.3 Refinement for Discretization

For a continuous attribute, refinement is similar to that for generalization except that no prior taxonomy tree is given and the taxonomy tree has to be grown dynamically in the process of refinement. Initially, the interval that covers the full range of the attribute forms the root. The refinement on an interval v, which is written as v! Child (v) refers to the optimal split of v into two child intervals child (v) that maximizes the information gain. The anonymity is not used for finding a slit good for classification. This is similar to defining a taxonomy best describes the application. Due to this extra step of identifying the optimal split of the parent interval, we treat continuous attributes separately from categorical attributes with taxonomy trees.

A refinement is valid (with respect to T) if T satisfied the anonymity requirement after the refinement. A refinement is beneficial (with respect to T) if more than one class is involved in the refined records. A refinement is performed only if it is both valid and beneficial. Therefore, a refinement guarantees that every newly generated qid has a (qid) _k.

3.2 Selection Criterion

We propose a selection criterion for guiding our TDR process to heuristically maximize the classification goal. Consider a refinement v ! Child (v), where v 2 Dj and Dj is a categorical attribute with a user-specified taxonomy tree or Dj is a continuous attribute with a dynamically grown taxonomy tree. The refinement has two effects: it increases the information of the refined records with respect to classification, and it decreases the anonymity of the refined records with respect to privacy. These effects are measured by “information gain”, denoted AnonyLoss(v). v is a good candidate for refinement if InfoGain(v) is large and AnonyLoss(v) is small. Our selection criterion is choosing the candidate v, for the next refinement, that has the maximum informationgain/ anonymity-loss trade-off. To avoid division by zero, 1 is added to AnonyLoss(v). Each choice of InfoGain(v) and anonyLoss(v) gives a trade-off between classification and anonymixation. It should be noted that Score is not a goodness metric of k-anonymixation. In fact, it is difficult to have a closer-form metric to capture the classification goal (on future data). We achieve this goal through this heuristic selection criterion. For concreteness, we borrow Shannon’s information theory to measure information gain [26]. Let Rv denote the set of records masked to the value v, and let Rc denote the set of records masked to a child value c in child(v) after refining v. Let jxj be the number of elements in a set of x.

AnonyLoss(v): Defined as AnonyLoss(v) ¼ avgfA(QIDj)_AV(QIDj)g; (4) where A(QIDj) and AV(QIDj) represent the anonymity before and after refining v. avgfA(QIDj)_AV(QIDj)g is the average loss of anonymity for all QIDj that contain the attribute of v if Dj is a categorical attribute without taxonomy tree, the refinement ?j ! fv;? jg means refining R?j into Rv and R0 ?j, where

R?j denotes the set of records containing?j before the refinement. Rv and R0? J denote the set of records contain v and? j after the refinement, respectively. We employ the same Score(v) function to measure the goodness of the refinement?j ! fv;?jg.

3.3 InfoGain versus Score

An alternative to Score is using InfoGain alone, that is, maximizing the information gain produced by a refinement without considering the loss of anonymity. This alternative may pick a candidate that has a large reduction in anonymity, which may lead to a quick violation of the anonymity requirement, thereby, prohibiting refining the data to a lower granularity. Table 2b shows the calculated InfoGain, AnonyLoss, and Score of the three candidate refinements. According to the InfoGain criterion, ANY Edu will be first refined because it has the highest InfoGain. The result is shown in Table 2c with A(QID)1/4 4. After that, there is no further valid refinement because refining either ANY Sex or [1-99] will result in a violation of 4-anonymity. Note that the first 24 records in the table fail to separate the 4N from the other 20Y. In contrast, according to the Score criterion, ANY Sex will be first refined. The result is shown in Table 2d, and A(QID)1/4 14. Subsequently, further refinement on ANY Edu is invalid because it will result in a (h9th;M;$^{1/2}$1_99)i)1/4 2<k, but the refinement on [1-99] is valid because it will result in A(QID) ¼ 6_k. The final masked table is shown in Table 2e where the information for separating the two classes is preserved. Thus, by considering the information / anonymity trade-off, the Score criterion produces a more desirable sequence of refinements for classification.

4. TOP DOWN REFINEMENT

4.1 The Algorithm

We present our algorithm TDR. In a preprocessing step, we compress the given table T by removing all attributes not in [QIDi and collapsing duplicates into a single row with the Class column storing the class frequency as in Table 1. The compressed table is typically much smaller than the original table. Below, the term “data records” refers to data records in this compressed form. There exists a masked table satisfying the anonymity requirement if and only if the most masked table does that is, jTj_k. this condition is checked in the preprocessing step as well. To focus on main ideas, we assume that jTj_k and the compressed table first in the memory. In Section 4.5, we will discuss the modification needed if the compressed table does not fit in the memory.

Algorithm: [Top-down Refinement (TDR)]

1. Initialize every value of Dj to the topmost value, suppress every value of dj to ?j, or include every continuous value of Dj into the full-range interval, where Dj 2[QIDi].

2. Initialize cut j of Dj to include the topmost value, Supj of Dj to include all domain values of Dj, and Intj of Dj to include the full-range interval, where Dj 2[QIDi].

3. While some x 2 h [cut j]; [Supj]; [Intj is valid and beneficial].

4. Find the Best refinement from h[cutj]; [Supj]; [Intji].

5. Perform Best on T and update h[cutj]; [supj]; [Intji].

6. Update Score(x) and validity for x 2 h[cutj]; [supj]; [Intji].

7. End while

8. Return Masked T and h[cutj; [Supj; [Intji].

High level description of our algorithm. Algorithm summarizes the conceptual algorithm. Initially, cutj contains only the topmost value for a categorical attribute Dj with a taxonomy tree, Supj contains all domain values of a categorical attribute Dj without a taxonomy tree, and Intj contains the full-range interval for a continuous attribute Dj. The valid beneficial refinements in h[cutj; [Supj; [Intji form the set of candidates. At each iteration, we find the candidate of the highest Score, denoted Best (Line 4), apply Best to T, update h[cutj; [Supj; [Intji (Line 5), and update Score and the validity of the candidates in h[cutj; [Supj; [Intji (Line 6). The algorithm terminates when there is no more candidate in h[cutj; [Supj; [Intji, in which case it returns the masked table together with the solution set h[cutj; [supj; [Intji. Our algorithm obtains the masked T by iteratively refining the table form the most masked state. An important property of TDR is that the anonymity requirement is antimonotone with respect to the TDR. If it s violated before a refinement, it remains violated after the refinement. This is because a refinement never equates distinct values; therefore it never increased the count of duplicates a(qid). Hence, the hierarchically state at the top is separated by a border above which lie all satisfying states and below which lie all violating states. The TDR finds a state on the border, and this state is maximally refined in that any further refinement of it would cross the border and violate the anonymity requirements. Note that there may be more than one maximally refined state on the border. Our algorithm finds the one based on the heuristic selection criterion of maximizing Score at each step. Samarati[2] presents some results related to ntimonotonicity, but the results are based on a different masking model that generalizes all values in an attribute to the same level and suppresses data at the record level. Theorem 1. Algorithm 1 finds a maximally refined table that satisfied the given anonymity requirement. Algorithm 1 makes no claim on efficiency. In fact, in a straightforward implementation, Lines 4,5 and 6 require scanning all data records and recomputing Score for all candidates in h[cutj; [Supj; [Intji. Obviously, this is not scalable. The key to efficiency of our algorithms is directly accessing the data records to be refined and updating Score based on some statistics maintained for candidate in h[cutj; [Supj; [Intji. In the rest of the section, we explain a scalable implementation of Lines 4, 5, and 6.

4.2 Find the Best Refinement (Line 4)

This step makes use of computed InfoGain(x) and Ax(QIDi) for all candidates x in h[cutj; [Supji; [Intji and computed A(QIDi) for each QIDi. Before the first iteration, such information is computed in an initialization step for every topmost value, every suppressed value, and every full-range interval. For each subsequent iteration, such information comes from the update in the previous iteration (Line 6). Findings the best refinement Best involves at almost j [cutjj) j [supjj) j [Intjj computations of Score without accessing data records. Updating InfoGain(x) and Ax(QIDi) will be considered in section 4.4

4.3 Perform the Best Refinement(Line 5)

We consider two cases of performing the Best refinement, corresponding to whether a taxonomy tree is available for the attribute Dj for Best. Case 1: Dj has a taxonomy tree. Consider the refinement Best ! child(Best) where Best 2 Dj and Dj is either a categorical attribute with a specified taxonomy tree or a continuous attribute with a dynamically grown taxonomy tree. First, we replace Best with child(Best) in h[cutj; [Intji. Then, we need to retrieve RBest, the set of data records masked to Best, to tell the child value in child(Best) for each individual data records. We

present a data structure Taxonomy Indexed PartitionS (TIPS) to facilitate this operations. This data structure is also crucial for updating InfoGain(x) and Ax(QIDi) for candidate x. the general idea is to group data records according to their masked records on [QIDi. Definitions 3 (TIPS). TIPS is a tree structure with each node representing a masked record over [QIDi and each child node representing a refinement of the parent node on exactly one attribute. Stored with each leaf node is the set of (compresses) data record having the same masked record, called a leaf partition. For each candidate refinement x, Px denotes a leaf partition whose masked record contains x, and Linkx denotes the link of all such Px. The head of Linx is stored with x. the masked table is represented by the leaf partitions of TIPS. Linkx provides a direct access to Rx, the set of (original) data records masked by the value x. initially, TIPS has only one leaf partition containing all data records, masked by the topmost value or interval on every attribute in [QIDi. In each iteration, we perform the best refinement Best by refining the leaf partition on LinkBest. Refine Best in TIPS. We refine each leaf partition PBest found on LinkBest as follows:

For each value c in child(Best), a child portion Pc is created under PBest and data record in PBest are split among the child partitions. Pc contains the data records in PBest if a categorical value c generalized the corresponding domain value in the record or if an interval c contains the corresponding domain value in the record, an empty Pc is removed. Linkc is created to link up all Pcs for same c. Also, link Pc to every Linkx to which PBest was previously linked, except for LinkBestFinally, mark c as "beneficial" if Rc has more than one class, where Rc denotes the set of data records masked to c. This is the only operation that actually accesses data records in the whole algorithm. The overhead is maintaining Linkx. For each attribute in [QIDi and each leaf partition on LinkBest, there are at most jchild(Best)j "relinking." Therefore, there are at most j QIDjj _ jLinkBestj _ jchild(Best)j"relinkings" for applying Best.

A TIPS has several useful properties:

1) All data records in the same leaf partition have the same masked record, although they may have different refined values.

2) Every data record appears in exactly one leaf partition.

3) Each leaf partition Px has exactly one masked qidj on QIDj and contributes the count jPxj towards a(qidj). Later, we use the last property to extract a(qidj) from TIPS.

4.4 Update Score and Validity (Line 6)

This step updates Score(x) and validity for candidates x in h[cutj; [Supj; [Intji to reflect the impact of the Best refinement. The key is computing Score(x) from the count statistics maintained in Section 4.3 without accessing data records. We update InfoGain(x) and Ax(QIDi) separately. Note that the updated A(IDi) is obtained from ABest(QIDi).

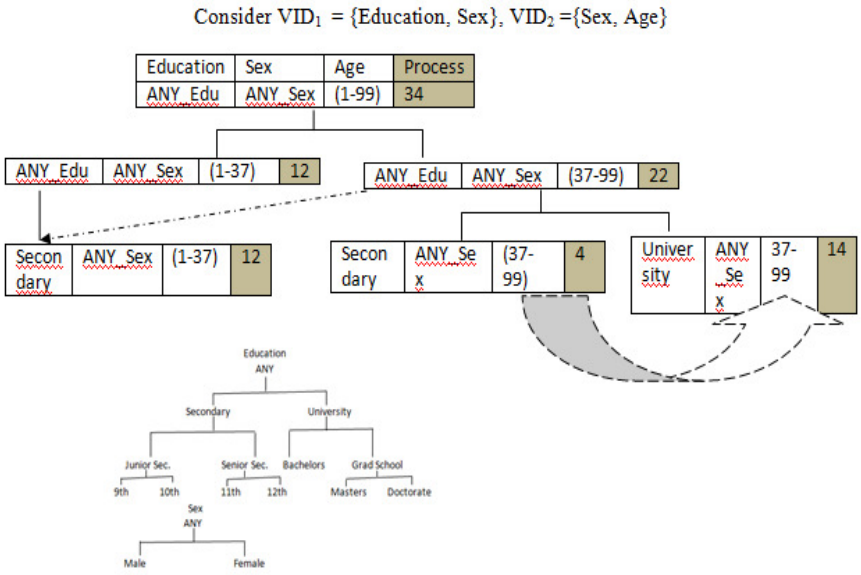
4.4.1 Update InfoGain(x)

An observation is that InfoGain(x) is not affected by Best ! child(Best), except that we need to compute InfoGain(c) for each newly added value c in child(Best). InfoGain(c) can be computed while collecting the count statistics for c in Case 1 of section 4.3. in case the refined attribute has not taxonomy tree, InfoGain(x) can be computed from the count statistics for x in Case 2 of Section 4.3.

4.4.2 Update AnonyLoss(x)

Again, we consider the two cases:

Case1: Dj has a taxonomy tree. Unlike information gain, it is not enough to compute Ac(QIDi) only for the new values c in child(Best). Recall that Ax(QIDi) is equal to the minimum a(qidi) after refining x. if both att(x) and att(Best) are contained in QIDi, the refinement on Best may affect this minimum hence, Ax(QIDi).



The above Fig presents the TIPS data structure presents the data structure Quasi-Identifier TreeS (QITS) to extract a(qidi) efficiently from TIPS for updating Ax(QIDi). Definition 4 (QITS) QITi for DIDi ¼ fD1;…;Dwg is a tree of w levels. The level p > 0 represents the masked values for Dp. Each root-to-leaf path represents an existing qidi on DIDi in the masked data, with a(qidi) stored at the leaf node. A branch is trimmed if it's a(qidi)$^{1/4}$ 0. A(QIDi) is equal to the minimum a(qidi) in QITi. In other words, QITi provides an index of a(qidi) by qidi. Unlike TIPS, QITS does not maintain data records. On applying Best ! child(Best), we update every QITi such that QIDi contains the attribute att(Best). Update QITi, for each occurrence of Best in QITi, create a separate branch for each c in child(Best). The procedure in algorithm 2 computes a(qidi) for the newly created qidis on such branches. The general idea is to loop through each Pc on Linkc in TIPS, increment a(qidi) by jPcj. This step does not access data records because jPcj was part of the count statistics of Best. Let r be the number of QIDi containing att(Best). The number of a(qidi) to be computed is at most r_jLinkBestj_jchild(Best)j.

5. SUMMARY

Our experiments verified several claims about the proposed TDR method. First, TDR masks a given table to satisfy a broad range of anonymity requirements without sacrificing significantly the usefulness to classification. Second, while producing a comparable accuracy, TDR is much more efficient than previously reported approaches, particularly, the genetic algorithm in [12]. Third, the previous optimal k-anonymization [7], [16] does not necessarily translate into the optimality of classification. The proposed TDR finds a better anonymization solution for classification. Fourth, the proposed TDR scales well with large data sets and complex anonymity

requirements. These performances together with the features discussed in Section 1 make TDR a practical technique for privacy protection while sharing information.

6. CONCLUSION

We considered the problem of ensuring an individual's anonymity while releasing person-specific data for classification analysis. We pointed out that the previous optimal k-anonymization based on a closed-form cost metric does not address the classification requirement. Our approach is based on two observations specific to classification: Information specific to individuals tends to be over fitting, thus of little utility, to classification; even if a masking operation eliminates some useful classification structures, alternative structures in the data emerge to help. Therefore, not all data items are equally useful for classification and less useful data items provide the room for anonymizing the data without compromising the utility. With these observations, we presented a top-down approach to iteratively refine the data from a general state into a special state, guided by maximizing the trade-off between information and anonymity. This top-down approach serves a natural and efficient structure for handling categorical and continuous attributes and multiple anonymity requirements. Experiments showed that our approach effectively preserves both information utility and individual's privacy and scales well for large data sets.

REFERENCES

[1] P. Samarati and L.Sweeney, "Generalizing Data to provide Anonymity when Disclosing Information," Proc. 17th ACM SIGACT-SIGMOD-SIGART Symp. Principles of Database Systems (PODS '98), p. 188, June 1998.

[2] P.Samarati, "Protecting Respondents' Identities in Microdata Release," IEEE Trans. Knowledge Eng., vol. 13, no.6, pp.1010-1027, Nov/Dec. 2001

[3] The House of Commons in Canada, "The Personal Information Protection and Electronic Documents Act," 1991, http://www.privcom.gc.ca/.

[4] S.M. Weiss and C.A. Kulikowski, Computer Systems that Learn: Classification and Prediction Methods from Statistics, Machine Learning, and Expert Systems. Morgan Kaufmann, 1991.

[5] T.Dalenius, "Finding a Needle in a Haystack or Identifying Anonymous Census Record," J. Official Statistics, vol.2, no.3, pp.329-336, 1986.

[6] L.Sweeney, "Achieving k-Anonymity Privacy Protection Using Generalization and Suppression," Int'l J. Uncertainty, Fuzziness, and Knowledge-Based Systems, vol.10, no.5 pp.571-588, 2002.

[7] R.J. Bayardo and R.Agrawal, "Data Privacy through Optimal k-Anonymization," Proc.21st Int'l Conf. Data Eng. (ICDE '05), pp.217-228. April 2005.

[8] G.Aggarwal, T.Feder, K.Kenthapadi, R. Motwani, R.Pamigraphy, D. Thomas, and A. Zhu, "Approximation Algorithms for k-Anonymity," J.Privacy Technology, no. 2005``1000`, Nov. 2005.

[9] A. Meyerson and R. Williams, "On the Complexity of Optimal k-Anonymity," Proc. 23rd ACM Symp. Principles of database Systems (PODS'04), pp. 223-228, 2004.

[10] L. Sweeney, "Datafly: A System for providing anonymity in Medical Data," Proc. Int'l conf. Database Security, pp. 356-381, 1998.

[11] A. Hundepool and L. Willenborg," and Argus : Software for Statistical Disclosure Control," Proc. Third Int'l Seminar on Statistical Confidentiality, 1996.

[12] V.S. Iyengar, "Transforming Data to Satisfy Privacy Constraints," Proc. Eighth ACM SIGKDD Int'l conf. Knowledge Discovery and Data Mining, pp.279-288, July 2002.

[13] K.Wang, P. Yu, and S. chakraborty, "Bottom-Up Generalization: A Data Mining Solution to Privacy Protection," Proc. Fourth IEEE Int'l conf. Data Mining (ICDM '04), Nov. 2004.

[14] B.C.M. Fung, K.Wang, and P.S. Yu, "Top-Down Specialization for Information and Privacy Preservation," Proc. 21st Int'l Conf. Data Eng. (ICDE '05), pp.205-216, April 2005.