First publ. in: IEEE Transactions on Knowledge and Data Engineering ; 22 (2010), 3. - pp. 334-347

http://dx.doi.org/10.1109/TKDE.2009.91

Efficient Multidimensional Suppression for K-Anonymity

Slava Kisilevich, Lior Rokach, Yuval Elovici, *Member, IEEE*, and Bracha Shapira

Abstract—Many applications that employ data mining techniques involve mining data that include private and sensitive information about the subjects. One way to enable effective data mining while preserving privacy is to anonymize the data set that includes private information about subjects before being released for data mining. One way to anonymize data set is to manipulate its content so that the records adhere to k-anonymity. Two common manipulation techniques used to achieve k-anonymity of a data set are generalization and suppression. Generalization refers to replacing a value with a less specific but semantically consistent value, while suppression refers to not releasing a value at all. Generalization is more commonly applied in this domain since suppression may dramatically reduce the quality of the data mining results if not properly used. However, generalization presents a major drawback as it requires a manually generated domain hierarchy taxonomy for every quasi-identifier in the data set on which k-anonymity has to be performed. In this paper, we propose a new method for achieving k-anonymity named K-anonymity of Classification Trees Using Suppression (kACTUS). In kACTUS, efficient multidimensional suppression is performed, i.e., values are suppressed only on certain records depending on other attribute values, without the need for manually produced domain hierarchy trees. Thus, in kACTUS, we identify attributes that have less influence on the classification of the data records and suppress them if needed in order to comply with k-anonymity. The kACTUS method was evaluated on 10 separate data sets to evaluate its accuracy as compared to other k-anonymity generalization- and suppression-based methods. Encouraging results suggest that kACTUS' predictive performance is better than that of existing k-anonymity algorithms. Specifically, on average, the accuracies of TDS, TDR, and kADET are lower than kACTUS in 3.5, 3.3, and 1.9 percent, respectively, despite their usage of manually defined domain trees. The accuracy gap is increased to 5.3, 4.3, and 3.1 percent, respectively, when no domain trees are used.

Index Terms—Privacy-preserving data mining, k-anonymity, deindentified data, decision trees.

1 Introduction

Knowledge Discovery in Databases (KDDs) is the process of identifying valid, novel, useful, and understandable patterns from large data sets. Data Mining (DM) is the core of the KDD process, involving algorithms that explore the data, develop models, and discover significant patterns. Data mining has emerged as a key tool for a wide variety of applications, ranging from national security to market analysis. Many of these applications involve mining data that include private and sensitive information about users [1]. For instance, medical research might be conducted by applying data mining algorithms on patient medical records to identify disease patterns. A common practice is to deidentify data before releasing it and applying a data mining process in order to preserve the privacy of users. However, private information about users might be exposed when linking deidentified data with external public sources. For example, the identity of a 95-year-old patient may be inferred from deidentified data that include the patients' addresses, if she is known as the only patient at this age in her neighborhood. This is true even if sensitive details such as her social security number, her name, and the name of the street, where she lives, were omitted.

To avoid such situations, privacy regulations were promulgated in many countries (e.g., privacy regulation as part of HIPAA[1] in the USA). The data owner is required to omit identifying data so that to assure, with high probability, that private information about individuals cannot be inferred from the data set that is released for analysis or sent to another data owner. At the same time, omitting important fields from data sets, such as age in a medical domain, might reduce the accuracy of the model derived from the data by the DM process. Researchers have shown that the HIPAA privacy rules have affected significantly their ability to perform retrospective, chart-based research [2], [3].

Privacy-preserving data mining (PPDM) deals with the trade-off between the effectiveness of the mining process and privacy of the subjects, aiming at minimizing the privacy exposure with minimal effect on mining results.

K-anonymity is an anonymizing approach proposed by Samarati and Sweeney [4]. A data set complies with k-anonymity protection if each individual's record stored in the released data set cannot be distinguished from at least $k-1$ individuals whose data also appear in the data set. This protection guarantees that the probability of identifying an individual based on the released data in the data set does not exceed $1/k$. Generalization and suppression are

- *S. Kisilevich is with the Department of Computer and Information Science, University of Konstanz/Germany, Universitaets Strasse 10, Box 78, 78457 Konstanz, Germany. E-mail: slaks@dbvis.inf.uni-konstanz.de.*
- *L. Rokach, Y. Elovici, and B. Shapira are with the Department of Information Systems Engineering, Ben-Gurion University of the Negev, POB 653, Beer-Sheva 84105, Israel. E-mail: {liorrk, bshapira}@bgu.ac.il, elovici@inter.net.il.*

1. HIPPA—Health Insurance Portability and Accountability Act.

the most common methods used for deidentification of the data in k-anonymity-based algorithms [5], [6], [7], [8].

Generalization consists of substituting attribute values with semantically consistent but less precise values. For example, the month of birth can be replaced by the year of birth which occurs in more records so that the identification of a specific individual is more difficult. Generalization maintains the correctness of the data at the record level but results in less specific information that may affect the accuracy of machine learning algorithms applied on the k-anonymous data set. Different systems use various methods for selecting the attributes and records for generalization as well as the generalization technique [9].

Suppression refers to removing a certain attribute value and replacing occurrences of the value with a special value "?," indicating that any value can be placed instead. Suppression can drastically reduce the quality of the data if not properly used [6]. This is the reason why most k-anonymity-related studies have focused on generalization.

Quasi-identifier is a set of features whose associated values may be useful for linking with another data set to reidentify the entity that is the subject of the data [6]. One major drawback of existing generalization techniques is that manually generated domain hierarchy trees are required for every quasi-identifier attribute of data sets before k-anonymity can be applied [8], [7], [10], [11], [12], [13].

In this paper, we present kACTUS—Supervised Decision Tree-based K-Anonymity, a new algorithm that deidentifies (anonymizes) data sets so that to assure high degree of users' privacy when data mining is applied, while having minimal impact on accuracy of data mining results. The privacy of the users is measured by the compliant of the data set to k-anonymity. kACTUS was specifically designed to support classification, but can be extended to support other data mining methods.

The new algorithm we developed, kACTUS, wraps a decision tree inducer which is used to induce a classification tree from the original data set. The wrapping approach [14] refers to using a known algorithm as a black box so that only knowledge of the interface is needed. While the wrapping approach was used in other domains (such as feature selection), it was not used for anonymizing data sets, and presents important advantages for this application. The wrapping approach enables utilization of existing state-of-the-art practice in generating accurate classifiers to generate anonymized data sets with minimal effect on the accuracy. Another advantage of the Wrapping approach is that it does not require any revision of existing algorithms for adjusting it to k-anonymity.

The automatically induced tree is used by kACTUS to apply k-anonymity on the data set (instead of using a manual generated domain generalization tree). kACTUS generates a k-anonymous data set that can be used by external users that may utilize any mining algorithm for training a classifier over the anonymous data set. The output of our algorithm is an anonymous data set which can be transmitted to the data consumers for further mining.

The kACTUS algorithm takes the suppression approach to anonymize the data set. We developed an efficient multidimensional suppression method, where values are suppressed only on certain tuples, depending on other attribute values. We show that our new multidimensional suppression scheme significantly outperforms existing single-dimensional methods and existing multidimensional generalization methods that require manual-defined generalization trees. In this paper, we describe our new method in detail and explain its advantages. We present the comparisons to other known methods and explicate the comparison results. The paper concludes with limitations and future research issues.

2 Related Work

PPDM is a relatively new research area that aims to prevent the violation of privacy that might result from data mining operations on data sets [1], [15], [16], [17]. PPDM algorithms modify original data sets so that privacy is preserved even after the mining process is activated, while minimally affecting the mining results quality. Verykios et al. [15] classified existing PPDM approaches based on five dimensions:

1. data distribution, referring to whether the data are centralized or distributed;
2. data modification, referring to the modifications performed on the data values to ensure privacy. There are different possible operations such as aggregation (also called generalization) or swapping;
3. data mining algorithms referring to the target DM algorithm for which the PPDM method is defined (e.g., classification [13], [18]);
4. data or rule hiding referring to whether the PPDM method hides the raw or the aggregated data; and finally,
5. privacy preservation, referring to the type of technique that is used for privacy preservation: heuristic [19], cryptography [17], [18]; or reconstruction-based (i.e., perturbing the data and reconstructing the distributions to perform mining [20], [21]).

The study presented in this paper can be classified according to the above dimensions: it deals with centralized databases (dimension 1), on which suppression of the data (dimension 2) is performed. The DM algorithm that this study is targeting is classification (dimension 3), and part of the raw data is hidden (dimension 4). We use the k-anonymity method which is a heuristic-based technique (dimension 5).

One of the PPDM techniques is k-anonymity. The k-anonymity concept [4] requires that the probability to identify an individual by linking databases does not exceed $1/k$. Generalization is the most common method used for deidentification of the data in k-anonymity-based algorithms [5], [7], [13], [19], [22], [23], [24]. Generalization consists of replacing specific data with a more general value to prevent individual identification; for example, the address that includes (Street, City, and State) can be replaced by (City and State) which applies to more records so that identification of a specific individual is more difficult. Some known generalization-based k-anonymity algorithms that guarantee optimal anonymity (in respect of accuracy by enumerating all candidate generalizations) are impractical [25], [26]. Some other systems employ heuristic-based

practical algorithms, most often using greedy algorithms in order to select attributes/value tuples to generalize [19], [27], [28]. For example, the Datafly system generates frequency lists of attributes and values [29] and generalizes the attribute having the highest number of distinct values. Generalization continues until there remain k or fewer tuples having distinct values. The μ-Argus system is another example of a greedy approach that generalizes attributes whose combinations with other attributes do not appear at least k times.

The aforementioned systems do not consider any specific DM algorithm to be operated on the data sets. The study presented in this paper considers the anonymity problems in terms of classification, i.e., the operations on the data are performed while taking into account their effect on classification results. A few other studies address the same problem, namely, k-anonymity for classification [11], [22]. In one work [10], a random genetic algorithm is used to search for the optimal generalization of data. This algorithm seems to be impractical due to its computational extensiveness. The author reported an 18 hours run for 30k records. Wang et al. [8] presented a practical effective bottom-up generalization that aimed at preserving the information needed for inducing the classifier while preserving privacy. They defined the "information gain" metric to measure the privacy/information trade-off. The bottom-up generalization technique can generalize only for categorical attributes. Fung et al. [7] presented another practical generalization method for classification using k-anonymity: the "top-down specialization (TDS)" algorithm. This algorithm handles both categorical and continuous attributes. TDS starts from the most general state of the table and specializes it by assigning specific values to attributes until violation of the anonymity may occur. More recently, Fung et al. [11] presented an improved version of TDS which is called "Top-Down Refinement" (TDR). In addition to the capabilities of TDS, TDR is capable of suppressing a categorical attribute with no taxonomy tree. They use a single-dimension recoding [13], i.e., an aggressive suppression operator that suppresses a certain value in all records without considering values of other attributes so that data that might adhere to k-anonymity might be also suppressed. This "oversuppression" reduces the quality of the anonymous data sets.

Friedman et al. [13] present kADET, a decision tree induction algorithm that is guaranteed to maintain k-anonymity. The main idea is to embed the k-anonymity constraint into the groining phase of a decision tree. While kADET has shown accuracy superior to that of other methods, it is limited to decision trees inducers. It differs from other methods such as TDS and TDR by letting the data owners share with each other the classification models extracted from their own private data sets, rather than to let the data owners publish any of their own private data sets. Thus, the output of kADET is an anonymous decision tree rather than an anonymous data set.

Sharkey et al. [30] present the APT algorithm, which like kADET, it also outputs anonymous decision tree rather than an anonymous data set. In addition, the authors show how the classification model can be then used to generate some pseudodata set. However, the pseudodata set is tightly coupled to the classification model. Because the classifier is not an authentic anonymous copy of the original private data set, so does the pseudodata set. For example, the values of the nonquasi-identifiers attributes (which can be shared with no risk) are lost if they are not included in the classification model. Similarly, the actual distribution of nonbinary target attributes can be distorted (the number of tuples in each class is only roughly estimated).

One main common disadvantage of existing anonymization algorithms (except TDR with some reservations) is the need to perform manual preprocessing, i.e., generation of a domain generalization taxonomy to define the hierarchy of the categorical attribute values involving prior knowledge about the domain. The domain tree should be prepared separately for every domain. Moreover, there might be disagreements between domain experts about the correct structure of the taxonomy tree, which may lead to differences in the results. The TDR algorithm is the only study, to the best of our knowledge that relaxes the need for a taxonomy tree only for categorical attributes, by using a single-dimension suppression operator for attributes for which domain taxonomy does not exist. As explained above, the performance of a single-dimension suppression operator is quite limited. Because it suppresses a certain value in all tuples without considering values of other attributes. This "oversuppression" may lead to unnecessary loss of data. Our algorithm uses multidimensional recoding, i.e., suppression of values is applied only on certain tuples, depending on other attribute values.

We suggest a practical and effective approach that provides an improved predictive performance in comparing to existing approaches and does not require the generation of manual domain generalization taxonomy. Instead, our approach uses a classification tree that is automatically induced from the original data set in order to perform a multidimensional suppression having a performance comparable to that of generalization-based algorithms.

3 Problem Formulation

In this section, several basic definitions to be used later in the paper are introduced, and the problem formulation is presented.

3.1 Formulation of Classification Problem

In a typical classification problem, a training set of labeled examples is given. The training set can be described in a variety of languages, most frequently, as a collection of records that may contain duplicates (also know as bag). A vector of attribute values describes each record. The notation A denotes the set of input attributes containing n attributes: $A = \{a_1, \ldots, a_i, \ldots, a_n\}$, and y represents the class variable or the target attribute. Attributes (sometimes referred to as features) are typically one of two types: categorical (values are members of a given set), or numeric (values are real numbers). When the attribute a_i is categorical, it is useful to denote its domain values by $dom(a_i)$. Numeric attributes have infinite cardinalities.

The instance space X (the set of all possible examples) is defined as a Cartesian product of all the input attribute

domains: $X = dom(a_1) \times dom(a_2) \times \cdots \times dom(a_n)$. The universal instance space (or the labeled instance space) U is defined as a Cartesian product of all input attribute domains and the target attribute domain, i.e., $U = X \times dom(y)$. The training set consists of a set of m records and is denoted as $S = (<x_1, y_1>, \ldots, <x_m, y_m>)$, where $x_q \in X$. Usually, the training set records are distributed randomly and independently according to some fixed and unknown joint probability distribution D over U.

It is assumed that a given inducer I is used to build a classifier (also known as a classification model) by learning from S. The classifier can then be used for classifying unlabeled instances. The notation $I(S)$ represents a classifier which was induced by training I with data set S.

3.2 Bag Algebra Operation Notation

Next, we will adopt the common operation notation of bag algebra (i.e., duplicates are allowed) to present projection and selection of tuples [31], where S denotes a bag.

1. Selection: The selection operator σ with the form $\sigma p(S)$ is used to select tuples in S that satisfy a given predicate p.
2. Projection: The projection operator π with the form $\pi_{\mathrm{B}}(\mathrm{S})$ is used to project bag S onto a subset of features B.
3. Duplicate elimination in bags:[2] The duplicate elimination operator ε with the form $\varepsilon(S)$ creates bag containing exactly one occurrence of each object of S.
4. Update operator: The update operator δ with the form $\delta a \rightarrow w(S)$, where S is a bag with an attribute a, which is assigned the value of expression w.

3.3 The k-Anonymity Protocol

Given a population of entities E, an entity-specific table $S \in U$ with input feature set $A = \{a1, a2, \ldots, an\}$, $f1 : E \rightarrow S$ and $f2 : S \rightarrow E'$, where $E \subseteq E'$. Q is a *quasi-identifier* of S, if $Q \subseteq A$ and $\exists e \in E$; such that $f2(\pi_Q f1(e)) = e$.

The formulation defines a *quasi-identifier* as a set of features whose associated values may be useful for linking to reidentify the entity that is the subject of the data [6].

A data set S and the quasi-identifier Q associated with it is said to satisfy *k-anonymity* if and only if each tuple in $\varepsilon(\pi_Q(S))$ appears with at least k occurrences in $\pi_Q(S)$.

The following bag S represents the Adult data set from the UC Irvine Machine Learning Repository. This data set contains census data and has become a commonly used benchmark for k-anonymity. The Adult data set has six continuous attributes and eight categorical attributes. The class attribute is income level, with two possible values, <= 50K or >50K. In this data set, $Q =$ (*age, workclass, fnlwgt, edu, edu-nun, marital-status, occupation, relationship, race, sex, native-country*) is a quasi-identifier since the values of these attributes can be linked to identify an individual. As in previous PPDM studies, we assume that the set of quasi-identifiers is provided by the user, and that there is only one set of quasi-identifiers. Examples of several records in the Adult data set are presented.

2. In SQL, this operation is known as the **DISTINCT** clause which is used together with the SQL SELECT keyword, to return a data set with unique entries for certain database table column.

age	workclass	fnlwgt	edu	edu-num	marital-status	occupation	relationship	race	sex	capital-gain	capital-loss	hours-per-week	native-country	salary
39	Private	77516	BA	13	Married	Excecutive	Not-in-family	White	M	2174	0	40	US	<=50K
39	Private	83311	BA	13	Married	Excecutive	Husband	White	M	0	0	13	US	<=50K
38	Private	215646	BA	9	Divorced	Excecutive	Not-in-family	White	M	0	0	40	US	<=50K
53	Private	234721	BA	7	Married	Excecutive	Husband	Black	M	0	0	40	US	<=50K
28	Private	338409	BA	13	Married	Excecutive	Wife	Black	M	0	0	40	Cuba	<=50K
37	Private	284582	BA	14	Married	Excecutive	Wife	White	M	0	0	40	Cuba	<=50K
49	Private	160187	BA	5	Married	Excecutive	Not-in-family	Black	M	0	0	16	Cuba	<=50K
52	State-gov	209642	BA	9	Married	Excecutive	Husband	White	M	0	0	45	Cuba	>50K
31	State-gov	45781	BA	14	Married	Excecutive	Not-in-family	White	M	14084	0	50	Cuba	>50K
42	State-gov	159449	MA	13	Married	Excecutive	Husband	White	M	5178	0	40	Cuba	>50K
37	State-gov	280464	MA	10	Married	Excecutive	Husband	Black	F	0	0	80	Cuba	>50K
30	State-gov	141297	MA	13	Married	Sales	Husband	Asian	F	0	0	40	Cuba	>50K
30	State-gov	141297	MA	13	Married	Sales	Husband	Asian	F	0	2	60	Cuba	>50K
30	State-gov	141297	MA	13	Married	Sales	Husband	Asian	F	0	1	80	Cuba	<=50K
34	State-gov	245487	7th-8th	4	Married	Sales	Husband	Indian	F	0	0	45	Mexico	<=50K

age	workclass	fnlwgt	edu	edu-num	marital-status	occupation	relationship	race	sex	native-country
39	Private	77516	BA	13	Married	Excecutive	Not-in-family	White	M	US
39	Private	83311	BA	13	Married	Excecutive	Husband	White	M	US
38	Private	215646	BA	9	Divorced	Excecutive	Not-in-family	White	M	US
53	Private	234721	BA	7	Married	Excecutive	Husband	Black	M	US
28	Private	338409	BA	13	Married	Excecutive	Wife	Black	M	Cuba
37	Private	284582	BA	14	Married	Excecutive	Wife	White	M	Cuba
49	Private	160187	BA	5	Married	Excecutive	Not-in-family	Black	M	Cuba
52	State-gov	209642	BA	9	Married	Excecutive	Husband	White	M	Cuba
31	State-gov	45781	BA	14	Married	Excecutive	Not-in-family	White	M	Cuba
42	State-gov	159449	MA	13	Married	Excecutive	Husband	White	M	Cuba
37	State-gov	280464	MA	10	Married	Excecutive	Husband	Black	F	Cuba
30	State-gov	141297	MA	13	Married	Sales	Husband	Asian	F	Cuba
30	State-gov	141297	MA	13	Married	Sales	Husband	Asian	F	Cuba
30	State-gov	141297	MA	13	Married	Sales	Husband	Asian	F	Cuba
34	State-gov	245487	7th-8th	4	Married	Sales	Husband	Indian	F	Mexico

For example, assume that k = 2. The data set described bellow does not satisfy k-anonymity requirements for Q = (age, workclass, fnlwgt, edu, edu-nun, marital-status, occupation, relationship, race, sex, native-country), since $\pi_Q(R)$ results in 15 records (see bellow). The records 12-14 comply with the k-anonymity restriction because they are three records with the same values for the quasi-identifiers (k = 2 < 3). However, the remaining records are unique, and thus, do not comply with the k-anonymity restriction (k = 2 > 1).

3.4 Optimal k-Anonymity Transformation for a Classification Problem

The main goal of this study is to introduce a new k-anonymity algorithm which is capable of transforming a nonanonymous data set into a k-anonymity data set. The transformation is aimed to achieve a predictive performance of a classifier trained on the transformed data set as similar as possible to the performance of a classifier trained on the original data set. Consequently, the problem can be formally phrased as follows: Given $k \in [1, m]$, an inducer I, a data set S with input attribute set $A = \{a1, a2, \ldots, an\}$, and target feature y from a distribution D over the labeled instance space, the goal is to find an optimal transformation $T : S \rightarrow S'$such that S' satisfy k-anonymity. Optimality is defined in terms of minimizing the deterioration in the generalized accuracy over the distribution D as a result of replacing classifier $I(S)$ with classifier $I(S')$.

Finding the optimal solution for Multidimensional k-Anonymity is known to be NP-Hard [32]. This study aims at defining a new k-anonymity heuristics that is not based on a predefined generalization in order to overcome the need to generate manually domain hierarchy trees for every quasi-identifier attribute. This manual process entails prior specific knowledge for each domain.

```
01: marital-status = Married-civ-spouse
02: |   education = 11th: <=50K. (271)
03: |   education = Masters: >50K. (713)
04: |   education = 9th: <=50K. (158)
05: |   education = HS-grad: <=50K. (3436)
06: |   education = Some-college
07: |   |   occupation = Handlers-cleaners: <=50K. (69)
08: |   |   occupation = Exec-managerial
09: |   |   |   workclass = Private
10: |   |   |   |   race = Black: <=50K. (6)
11: |   |   |   |   race = White: >50K. (190)
12: |   |   |   |   race = Asian-Pac-Islander: <=50K. (3)
13: |   |   |   |   race = Other: <=50K. (1)
14: |   |   |   |   race = Amer-Indian-Eskimo: <=50K. (2)
15: |   |   |   workclass = Self-emp-not-inc: <=50K. (39)
16: |   |   |   workclass = State-gov: >50K. (8)
17: |   |   |   workclass = Federal-gov: >50K. (16)
18: |   |   |   workclass = Local-gov: <=50K. (23)
19: |   |   |   workclass = Self-emp-inc
20: |   |   |   |   sex = Male: >50K. (42)
21: |   |   |   |   sex = Female: <=50K. (4)
22: |   |   |   workclass = Without-pay: >50K. (0)
23: marital-status = Married-AF-spouse: <=50K. (16)
```

Fig. 1. Classification tree for adult data set.

```
01: marital-status = Married-civ-spouse
02: |   education = 11th: <=50K. (271)
03: |   education = Masters: >50K. (713)
04: |   education = 9th: <=50K. (158)
05: |   education = HS-grad: <=50K. (3436)
06: |   education = Some-college
07: |   |   occupation = Handlers-cleaners: <=50K. (69)
08: |   |   occupation = Exec-managerial
09: |   |   |   workclass = Private (100)
15: |   |   |   workclass = Self-emp-not-inc: <=50K. (39)
16: |   |   |   workclass = State-gov: >50K. (8)
17: |   |   |   workclass = Federal-gov: >50K. (16)
18: |   |   |   workclass = Local-gov: <=50K. (23)
19: |   |   |   workclass = Self-emp-inc
20: |   |   |   |   sex = Male: >50K. (42)
21: |   |   |   |   sex = Female: <=50K. (4)
22: |   |   |   workclass = Without-pay: >50K. (0)
23: marital-status = Married-AF-spouse: <=50K. (16)
```

Fig. 2. Revised classification tree after iteration 1.

4 Methods

4.1 Overview

In this section, we describe our new method for k-anonymity. Recall that the goal of this study is to create an anonymous data set where the predictive performance of a classifier trained on the anonymous data set is as similar as possible to the performance of a classifier trained on the original data set. In order to achieve this, one need to consider how the input attributes affect the target attributes (class).

It is possible to tackle the latter issue as part of the anonymization algorithm or by utilizing existing classifier inducers. In this study, we take the second approach, assuming that it will be more efficient to use existing mature inducers than to try to "invent the wheel," especially when the target is to create an anonymous data set and not a classifier.

Specifically, our approach wraps an existing classification tree induction algorithm (such as C4.5) and is referred to as K-Anonymity of Classification Trees Using Suppression (kACTUS). The classification tree inducer is used to induce a classification tree from the original data set which indicates how the attribute values affect the target class. The classification tree can be easily interpreted by a machine in order to perform the k-anonymity process.

Each path from the root to a leaf can be treated as a classification rule. A subset of the data set is ascribed with each leaf. If this subset size is greater than k, then this subset of instances can be easily anonymized by suppressing all the quasi-identifier attributes that are not referenced in one of the nodes along the path from the root. Assuming that all attributes are categorical, then all quasi-identifiers attributes that are referenced have the same value for all the instances of the subset, and thus, there is no need to suppress or generalize their values). In our terminology, a leaf node complies with the k-anonymity if the number of instances that ascribed to this leaf is higher or equal to k.

If all the leaves in the tree comply with the k-anonymity, the data set can be k-anonymized by suppression as described above. For leaves that do not comply with the k-anonymity, by adequately pruning them, one can obtain a new leaf which may comply with the k-anonymity. We utilize the fact that the order in which the attributes are selected for the decision tree usually implies their importance for predicting the class. Thus, by pruning the rules in a bottom-up manner, we suppress the least significant quasi-attributes. The remaining attributes are the quasi-attributes which will be left as is. In Section 4.2, we demonstrate the new method with an example and provide a detailed example of our algorithm. In Section 4.4, we present the correctness proof and complexity analysis of the new method.

4.2 Illustrative Example

The proposed anonymize procedure is illustrated on the adult data set assuming k-anonymity = 100 and the quasi-identifier which was used in Section 3.1. First, we build a classification tree using an existing classification tree algorithm. Fig. 1 describes the generated classification tree after applying C4.5 algorithm on the quasi-identifier attributes in the Adult data set. A number in a bracket is associated with each leaf. This number represents the number of instances in the training set that complies with the tests along the path.

Given the classification tree, we begin in the anonymization process by reviewing the nodes with height = 1. Nodes 9 and 19 have height = 1; thus, we arbitrarily select node 9. Its children (nodes 10-14) are examined indicating that only node 11 complies with k (because $190 > 100$). The remaining siblings (nodes 10, 12, 13, and 14) have $6 + 3 + 1 + 2 = 12$ instances in total, i.e., there are 88 instances missing. Node 11 has 90 extra instances ($190 - 100 = 90$). Thus, 88 instances of node 11 can be randomly selected and used to complement the uncomplying nodes. The remaining 102 instances of node 11 can be mounted as is to the anonymous data set with the following quasi-identifiers revealed (all nonquasi-identifier attributes, such as hours-per-week, can be revealed as well): marital-status = Married-civ-spouse, education = Some-college, occupation = Exec-managerial, workclass = Private, race = White.

Finally, node 9 is pruned resulted with the following revised tree presented in Fig. 2. Given the new tree, the next node to be examined is node 19. None of its siblings complies with k. Thus, no instances are mounted to the

```
01: marital-status = Married-civ-spouse
02: |   education = 11th: <=50K. (271)
03: |   education = Masters: >50K. (713)
04: |   education = 9th: <=50K. (158)
05: |   education = HS-grad: <=50K. (3436)
06: |   education = Some-college
07: |   |   occupation = Handlers-cleaners: <=50K. (69)
08: |   |   occupation = Exec-managerial
09: |   |   |   workclass = Private (100)
15: |   |   |   workclass = Self-emp-not-inc: <=50K. (39)
16: |   |   |   workclass = State-gov: >50K. (8)
17: |   |   |   workclass = Federal-gov: >50K. (16)
18: |   |   |   workclass = Local-gov: <=50K. (23)
19: |   |   |   workclass = Self-emp-inc (46)
22: |   |   |   workclass = Without-pay: >50K. (0)
23: marital-status = Married-AF-spouse: <=50K. (16)
```

Fig. 3. Revised classification tree after iteration 2.

anonymized data set, and node 19 is simply pruned, resulting with tree as shown in Fig. 3. Next node to be examined is node 8. Only the first sibling (node 9) complies with k; thus, its instances are mounted to the anonymized data set. Node 8 is pruned and accumulates the instances of the uncomplying nodes 15-19 and 22 (total of 132 instances), as presented in Fig. 4.

4.3 Supervised Decision Tree-Based K-Anonymity

The kACTUS algorithm is presented in Fig. 5. kACTUS consists of two main phases: In the first phase, a classification tree is induced from the original data set; in the second, the classification tree is used by a new algorithm developed in this study to k-anonymize the data set.

4.3.1 Phase 1: Deriving the Classification Tree

In this phase, we are employing a decision tree inducer (denoted by *CTI*) to generate a decision tree denoted by *CT*. The tree can be derived using various inducers. We concentrate on top-down univariate inducers which are considered the most popular decision tree inducers [33] and include the well-known algorithms C4.5 [34]. Top-down inducers are greedy by nature and construct the decision tree in a top-down recursive manner (also known as divide and conquer). Univariate means that the internal nodes are split according to the value of a single attribute.

The decision tree is trained over the projection of the quasi-identifiers, i.e., $(CT \leftarrow CTI(\pi_{Q \cup y}(S))$. The wrapped inducer *CTI* should be differentiated from the target inducer *I*. Inducer *I* is applied on the anonymous data set (i.e., after applying k-anonymization process). The aim of the *CTI* is to reveal which quasi-identifier is more relevant for predicting the class value.

```
01: marital-status = Married-civ-spouse
02: |   education = 11th: <=50K. (271)
03: |   education = Masters: >50K. (713)
04: |   education = 9th: <=50K. (158)
05: |   education = HS-grad: <=50K. (3436)
06: |   education = Some-college
07: |   |   occupation = Handlers-cleaners: <=50K. (69)
08: |   |   occupation = Exec-managerial (132)
23: marital-status = Married-AF-spouse: <=50K. (16)
```

Fig. 4. Revised classification tree after iteration 3.

```
kACTUS (S,Q,CT,k)
Input: S (Original dataset), Q (The quasi-identifier set),
k (Anonymity threshold)
Output: S' (Anonymous dataset)

1: CT← CTI(πQ∪yS) /* phase 1 */
2: Return Anonymize (SO,Q,CT,k) /* phase 2 */

Anonymize (SO,Q,CT,k)
Input: SO (Original dataset to be anonyimized), Q (The
quasi-identifier set), CT (Classification tree),
k (Anonymity threshold)
Output: S'        Anonymous dataset

3: S←SO /* work on a copy of the original dataset */
4: S'←∅
5: WHILE height(root(CT))>0
6:   p←node in CT whose height=1
7:   nui ← 0 /* number of uncomplying instances */
8:   SP ← ∅ /* candidate instances to be suppressed */
9:   Sextra ← ∅ /* extra complying instances */
10:  FOR each v∈ children(p) DO
11:     IF |σant(v)(S)| ≥ k THEN
12:        SV←randomSelect(|σant(v)(S)|-k, σant(v)(S))
13:        Sextra ← Sextra ∪ SV
14:        SP←SP∪ (σant(v) (S)-SV)
15:     ELSE
16:        nui←nui+|σant(v)(S)|
17:     END IF
18: END FOR
19: IF nui<k THEN
20:     requried←k-nui
21:     IF |Sextra| ≥ required THEN
22:        SNot_required←randomSelect(|Sextra|-required, Sextra)
23:     ELSE
24:        SNot required ← Sextra
25:     END IF
26:     SP←SP∪ SNot required
27: END IF
28:     suppressComplyingChildren(S',SP,Q, p)
29:     S←S-SP
30:     prune(CT,p)
31: END WHILE
32: IF |S| ≥ k THEN S' ← S'∪suppress (S,Q,∅)

Suppress (R,Q,pred)
Input: R (Dataset), Q (The quasi-identifier set), v
(Predicates)
Output: R' (Suppressed dataset)

33: R'←R
34: For each a∈Q Do
35:       If a does not appear in an antecedent in v
36:              δa → ν(R')
37:       End If
38: End For
39: Return R'

suppressComplyingChildren(S',S,Q, p)
Input: S' (anonotmous dataset), S (original dataset), Q (
quasi-identifier), p (parent node)

40: FOR each v∈ children(p) DO
41:     IF |σant(v)(S)| ≥ k THEN
42:              Sv← σant(v)(S)
43:              S' ← S'∪suppress (Sv,Q,ant(v))
44:     END IF
45: END FOR
```

Fig. 5. kACTUS Algorithm.

Any internal node (nonleaf) with less than k instances cannot be used by itself for generating the anonymous data set. Thus, even if such a node is provided in the classification tree it can be pruned in advance. In many decision trees inducers, such as C4.5, the user can control the tree growing process by setting the algorithm's parameters. Specifically, the parameter MinObj ("minimum number of instances") indicates the number of instances that should be associated with a node in order it to be considered for splitting. By setting MinObj to k, one ensures that there are no noncomplying internal nodes that are needed to be pruned. Thus, with this parameter setting, we can reduce the tree

size without sacrificing the accuracy performance of the k-anonymous procedure. Still in Phase 2 described next, no assumption regarding the internal nodes is made.

4.3.2 Phase 2: K-Anonymity Process

In this phase, we use the classification tree that was created in the first phase to generate the anonymous data set. We assume that the classification tree complies with the following properties:

1. The classification tree is univariate, i.e., each internal node in the tree refers to exactly one attribute.
2. All internal nodes refer to a quasi-identifier attributes. This is true because the decision tree was trained over the projection of the quasi-identifier set ($\pi_{Q \cup y}(S)$).
3. Assuming a top-down inducer, the attributes are sorted (from left to right) according to their significance for predicting the class (where the rightmost relates to the least significant attribute).
4. Complete Coverage: Each instance is associated with exactly one path from root to leaf.

In the next phase, we utilize these properties for the k-anonymity process. Given a tree *CT* and node v, we define the following functions and procedures. Because these functions are straightforward, they are used here without providing pseudocode.

1. root(CT)—returns the root node of CT.
2. parent(v)—returns the parent of v.
3. height(v)—returns the height (length of the longest path from that node to a leaf) of v.
4. children(v)—returns the set of immediate descendants of v.
5. ant(v)—returns the antecedent associated with node v. Conjoining (logically, ANDed together) the tests along the path from the root to the node forms the antecedent part. For example, the seventh leaf in Fig. 1 is associated with the antecedent "Marital-status = Married AND Education = Some-college AND Occupation = Handlers-cleaners."
6. prune(CT,v)—prunes all descendants of v from the tree CT.
7. randomSelect (k,S)—return k randomly selected instances from relation S.

Our supervised k-anonymity process is described in procedure *Anonymize* of Fig. 5. The input to the *Anonymize* procedure includes the original data set S, quasi-identifier set Q, classification tree *CT*, and the anonymity threshold k. The output of the algorithm is a k-anonymous data set denoted by S'. For the sake of simplicity, we first assume that all quasi-identifiers are categorical. We later relax this assumption.

While the tree is not fully pruned (i.e., the height of the root is greater than 0), we continue to iterate. In each iteration, we choose one internal node (denoted by p) of height = 1. Note that such a node can always be found.

In lines 9-18, we go over the children of p (which are leaves as height(p) = 1) and check if they comply with k, i.e., we verify that the number of records in the data set satisfying the leaf antecedents is higher than or equal to k. The instances associated with the complying leaves form the candidate set of instances to be suppressed in this iteration (denoted by *SP*). The number of instances associated with the noncomplying leaves is counted by the counter *nui* (line 16).

In lines 19-25, we check if the total number of instances associated with the noncomplying leaves is not jointly comply with k (i.e., $nui < k$). If not, we attempt to reach k by taking instances from the complying siblings. Siblings can waive instances only if their have extra instances. For example, if $k = 100$ and a certain sibling has 102 instances associated with, then two instances can be waived. The extra instances are randomly accumulated in S_{extra} in lines 9-17.

Finally, if the extra instances can actually help us to reach the threshold k (line 20), we randomly select the non-required instances from S_{extra} and add it to *SP*. This ensures that the postpruning p is associated with k instances (i.e., comply with k) and will be mounted to the anonymous data set as is in the subsequent iterations. On the other hand, if the extra instances are not sufficient to make the noncomplying leaves reaching the threshold k, then no instances are moved from the complying leaves to the noncomplying leaves. We decided not to move any instances, because even if we move all the extra instances, the postpruning p will not comply with k and it is needed to be furthered pruned.

In lines 28-30, we eventually suppress the complying children, removing suppressed instances from the data set S and prune p. It should be noted that our algorithm is different from bottom-up pruning procedures which are usually included in advanced decision trees inducers. Contrary to pruning procedures, the proposed algorithm may prune every path differently. Thus, instead of pruning all branches from a certain point to the leaves, some paths remain as is (if they comply with k-anonymity), while other are pruned. Moreover, instances that are associated with the same leaf can be pruned differently, because some may be randomly chosen to help their noncomplying siblings.

Finally, in line 32, we are left with the root of the tree and examine whether the number of instances left with the root node comply with k. If such a situation occurs, then we suppress them, and then, move them to the anonymous data set. Note that if all attributes are quasi-identifiers (i.e., Q = A), we copy these instances to the anonymous data set, but not before suppressing all input attributes. Thus, these instances are left only with the target attribute. Still some induction algorithms can benefit from these empty instances, as it provides additional information regarding the class distribution.

However, if the remaining instances do not comply with k, we do not copy these instances to the anonymous data set. Thus, it is obvious that the new anonymous data set may contain fewer instances than the original one, but our experiments show that the number of removed instances is very small compared to the total number of instances.

4.4 kACTUS Properties

Corollary 1. *The kACTUS algorithm is correct.*

Proof. In order to prove the correctness of the algorithm, we need to show that the data set S' complies with k-anonymity. However, this can be easily verified because just before calling the suppress procedure; we verify the k threshold (line 32 and line 41). □

Corollary 2. *The computational complexity of kACTUS algorithm overhead is linearly related to the number of instances.*

Proof. We need to find the computational overhead incurred by the kACTUS, in addition to the complexity of the decision tree inducer *CTI* (i.e., the complexity of the Anonymize procedure). □

We assume that we are given a tree structure such that each leaf holds the pointers to the instances complying with this leaf (i.e., complying with all the tests in the corresponding path from the root to the leaf). Note that all selection operations in kACTUS are performed only on the leaves. The number of iterations is bounded by the number of internal nodes, which cannot exceed the number of instances (i.e., m). On each iteration of the outer loop, we handle a single node. The number of children associated with the internal node is maximum $d_{\max}$, which represent the largest attribute domain. Regarding the operations on the instances:

1. Summing up all instances suppression, we maximally manipulate m instances, each with $|Q|$ suppressions and $|A| - |Q|$ values duplications. Thus, the suppression operations end up with $O(m|A|)$.
2. Each instance with $|A|$ attributes is added to an existing bag (line 13, line 14, and line 26) not more than $|Q|$ times. Note that the longest path is bounded by $|Q|$ because the tree is using only the quasi-identifiers. Moreover, some instances may left the next tree level. This totally ends up with $O(m|A||Q|)$.
3. As for the expensive minus operations—actually, there is no need to explicitly perform them. Lines 11-13 can be joined by shuffling the bag's instances and add the top $|\sigma_{ant(v)}(S)| - k$ instances to S_{extra} and the remaining to *SP*. Line 28 can be relaxed by adding the instances of the noncomplying leaves and the remaining extra required instances of the complying leaves, and associate them to the new leaf (note that the pruned parent becomes a leaf). Thus, the minus operations also end up with $O(m|A||Q|)$.

In summary, the complexity of kACTUS overhead is

$$O(m \cdot d_{\max}) + O(m|A|) + O(m|A||Q|) + O(m|A||Q|) = O(m \cdot \max(d\max, |A||Q|)).$$

Practically, the computational complexity is affected by the size of k. For a large number of k, more inner iterations are needed till the rule complies with k. However, k could not be compactly referred to in the complexity analysis without making major assumptions about the tree structure (for example, assuming that the tree is balanced).

Corollary 3. *The number of instances that might be lost due to kACTUS is bounded by k.*

Proof. The proof of this corollary is straightforward. The only cause of instances loss is Line 32 in the algorithm. Instances are lost if and only if the left over subset (instances that we pruned up to the root) has less than k instances. □

4.5 Handling Numeric Attributes

In this section, we relax the assumption that the quasi-identifier includes only categorical attributes. For this, we need to revise only the suppressed function in the algorithm. Fig. 6 presents the new suppression function. In this function, if the attribute a is included in v but it appears with an inequality predicate (i.e., it is a numeric attribute), then we fill all the instances of the attribute a in R with a value that is equal to the mean of all a values in R. We illustrate the above with the following node path:

```
Suppress (R,Q,v)
Input: R (Dataset), Q (The quasi-identifier set), v
(Antecedents)

Output: R' (Suppressed dataset)

1:R'←R
2:For each a∈Q Do
3:   IF a does not appear in a antecedent of v
4:        δ_{a→'All'}(R')
5:   Else
6:      If a doesn't appear in v with equality predicate
7:          val←mean value of a in R
8:          δ_{a→val}(R')
9:      End If
10:  End If
11:End For
12:Return R'
```

Fig. 6. Suppress for numeric attributes.

```
marital-status = Married-civ-spouse AND age >= 30 AND
occupation = Exec-managerial AND age < 50 AND race = Other
AND age < 40: <=50K. (5)
```

Note that the numeric attribute *age* appears in the rule three times. We also assume that k = 10. The path does not comply with k, so we prune it and obtain:

```
marital-status = Married-civ-spouse AND age >= 30 AND
occupation = Exec-managerial AND  age < 50 AND race =
Other: <=50K. (15)
```

The new node complies with k (because it has 15 instances). These instances will be added to the data set with the mean value which should be in the range [30,50).

Note that if there are numeric quasi-identifiers, then the computational complexity of the kACTUS is not bounded by Corollary 2 because the same attribute can appear several times in the same rule (as was demonstrated in the above example). However, the experimental study shows that practically, the inclusion of numeric attributes has no significant effect on the computational cost.

5 Experimental Evaluation

5.1 Overview

In order to evaluate the performance of the proposed method for applying k-anonymity to a data set used for classification tasks, a comparative experiment was conducted on benchmark data sets. Specifically, the experimental study has the following goals:

1. To compare the obtained classification accuracy to the original accuracy without applying k-anonymity.

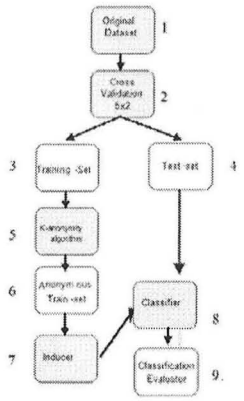


Fig. 7. The experimental process.

2. To compare the proposed method to existing k-anonymity methods in terms of classification accuracy.
3. To investigate the sensitivity of the proposed method to different classification methods.

5.2 Experimental Process

Fig. 7 is a graphic representation of the experimental process that was conducted. Unshaded boxes represent data sets. The main aim of this process is to estimate the generalized accuracy (i.e., the probability that an instance was classified correctly). First, the data set (box 1) was divided into training (box 3) and test sets (box 4) using five iterations of twofold cross validation (box 2—known as the 5×2CV procedure) as proposed by Alpaydin [35]. On each iteration, the data set is randomly partitioned into two equal-sized sets S1 and S2 such that the algorithm is evaluated twice: on the first evaluation, S1 is the training set and S2 the test set, and vice versa the second time. We apply (box 5) the k-anonymity method on the training set and obtain a new anonymous training set (box 6). An inducer is trained (box 7 over the anonymous training set to generate a classifier (box 8). Finally, the classifier is used to estimate the performance of the algorithm over the test set (box 9).

Note that in the kACTUS algorithm, the classifier can use the test set as is. In kACTUS, suppression is performed on the training set, i.e., some values are converted to missing values and all others left on their original values. Many of the existing induction algorithms are capable to train from data set with missing values. The test set attributes use the same domain values as in the train set. Nevertheless, TDS and TDR require fitting the test set to the classifier in order to ensure fair comparison with kACTUS. If, for example, the values "USA" and "Mexico" are generalized to the value "America," then all occurrences of "USA" and "Mexico" in the training data set will be replaced with the value "USA." A classifier trained on the anonymous training set will not be aware of the existence of the values "USA" and "Mexico" (note that the classifier is not aware of the generalization tree). Thus, if this classifier is now used to classify a test instance with the original value of "USA," then the classifier will not be able to utilize the fact that "USA" should be treated as "America." Because kADET is embedded in the decision tree algorithm, it is capable, like kACTUS, of handling the test set as is.

TABLE 1
The Properties of the Data Sets

Dataset	Application Area	Number of tuples	Number of attributes	Propoction of numeric attributes	Number of classes
Adult	Social - Predict whether income exceeds $50K/yr based on census data. Also known as "Census Income" dataset	48842	14	42.85%	2
German credit	Financial - classify people described by a set of attributes as good or bad credit risks.	1000	20	35%	2
Glass	Physical - USA Forensic Science Service; 6 types of glass; defined in terms of their oxide content	214	10	100%	7
Ionosphere	Physical - Classification of radar returns from the ionosphere	351	34	100%	2
Japanese Credit	Financial - classify people described by a set of attributes as good or bad credit risks.	125	15	40%	2
Nurse	Social - rank applications for nursery schools	12960	8	0%	5
Pima Diabetes	Life-The diagnostic, binary-valued variable investigated is whether the patient shows signs of diabetes	768	8	100%	2
TTT	Game - Binary classification task on possible configurations of tic-tac-toe game	958	9	0%	2
Vehicle	Image Processing - to classify a given silhouette as one of four types of vehicle.	946	18	100%	4
Waveform	Physical – classification of waveforms	5000	40	100%	3

The same cross-validation folds are implemented for all algorithms compared. Since the average accuracy is a random variable, the confidence interval was estimated by using the normal approximation of the binomial distribution. Moreover, we used the combined 5×2 CV F-test to accept or reject the hypothesis that the two methods have the same error rate with a 0.95 confidence level.

It should be noted that the above experimental process is slightly different from the process used in [11]. As Friedman et al. [13] already indicated, the experiment in [11] uses the entire data set to perform generalization and "this may provide different generalization results." In fact, according to the experimental design in [11], the split into training and test sets is performed after the generalization is applied. Thus, the estimated variance of the cross validation solely measures the inducer's variance and not the anonymity variance. In the experimental design that we use, the split is performed before generalization is applied; thus, we examine the anonymity on various samples.

5.3 Data Sets

The privacy-preserving classification algorithms are usually evaluated only on the Adult data set which has become a commonly used benchmark for k-anonymity [7], [13], [22]. Fung et al. [11] also evaluated the TDR algorithm on the German credit data set. In this experimental study, we used an additional eight data sets (10 data sets in total), which were also selected from the UCI Machine Learning Repository [36] and are widely used by the machine learning community for evaluating learning algorithms. The data sets vary across such dimensions as the number of target classes, of instances, of input features, and their type (nominal and numeric). Table 1 summarizes the properties of the data sets.

5.4 Algorithms Used

For the anonymity phase (box 5 in Fig. 7), we compared the proposed algorithm (denoted as kACTUS) to TDS [7],

TABLE 2
Accuracy versus K for Kactus Algorithm

Dataset	Inducer	K-Anonymity					
		1	5	10	15	20	30
Japanese Credit	C4.5	85.08±1.58	84.64±3.56	85.42±1.56	85.18±1.44	85.18±1.44	85.51±1.49
	PART	83.19±3.56	84.73±1.49	85.29±1.57	84.87±1.60	85.18±1.44	85.15±1.47
	NB	84.70±2.30	83.12±5.28	85.23±1.70	84.90±1.54	85.26±1.38	85.15±1.47
	Logistic	81.62±1.62	82.49±5.30	84.72±1.78	84.87±1.60	*85.18±1.44	*85.15±1.47
Pima Diabetes	C4.5	75.17±2.49	75.20±2.91	74.26±2.88	74.70±3.09	73.70±3.10	72.40±3.69
	PART	74.88±1.99	75.42±3.03	74.26±2.90	75.12±2.95	74.41±3.25	72.40±3.69
	NB	77.56±2.15	75.69±2.98	75.52±3.37	76.16±2.78	74.92±3.32	74.33±3.55
	Logistic	78.29±2.06	74.17±4.21	75.25±2.36	75.84±3.18	75.36±3.15	74.56±3.46
Glass	C4.5	67.63±3.18	*61.89±3.69	*59.47±2.98	*58.11±3.75	*55.13±4.59	*45.42±5.62
	PART	68.88±4.91	62.31±3.81	*60.33±2.72	58.52±3.80	*55.62±4.76	*46.70±9.65
	NB	71.96±3.34	*62.49±4.55	*60.35±3.58	56.57±6.76	*55.46±4.00	*46.15±9.26
	Logistic	69.93±3.16	*30.86±15.95	46.75±15.08	49.25±16.54	*54.50±7.17	*44.66±5.64
Ionosphere	C4.5	89.08±1.99	87.71±2.58	85.99±4.65	85.18±5.33	82.77±4.75	81.52±2.92
	PART	88.16±2.95	87.16±3.00	85.21±3.93	85.17±4.30	82.34±4.15	80.82±2.28
	NB	90.64±2.42	89.00±1.82	86.12±3.50	85.53±3.43	83.65±3.53	81.85±2.82
	Logistic	87.59±2.67	81.50±5.11	80.04±14.05	82.91±7.23	78.21±7.01	*29.79±2.38
Nurse	C4.5	95.84±0.20	*93.61±0.40	*92.52±0.34	*92.13±0.28	*91.59±0.30	*90.55±0.39
	PART	98.10±0.28	*94.37±0.26	*92.88±0.66	*92.53±0.42	*91.65±0.66	*91.01±0.54
	NB	90.19±0.31	*79.35±2.51	*77.76±0.48	*78.95±2.22	*81.22±2.59	*82.53±3.02
	Logistic	92.43±0.39	*50.07±6.88	*49.03±8.86	*52.59±10.59	58.60±15.09	55.58±11.58
TTT	C4.5	81.20±1.56	*72.89±2.48	*73.16±2.19	*70.49±1.52	*71.70±1.76	*70.30±1.65
	PART	89.24±4.13	*74.58±3.22	*72.24±3.03	*71.49±1.61	*71.83±1.86	*70.30±1.45
	NB	70.70±2.49	62.63±3.27	61.94±3.94	64.35±3.71	65.16±3.81	69.39±2.40
	Logistic	97.66±0.66	*48.25±10.92	*52.94±5.81	*63.01±6.22	*60.15±7.27	*66.06±3.04
Vehicle	C4.5	66.39±2.65	66.01±3.88	*63.42±3.41	62.13±3.11	*61.11±2.51	*57.64±2.73
	PART	68.57±1.41	*64.97±3.25	*63.26±3.63	*62.02±3.32	*61.04±2.64	58.48±3.46
	NB	63.06±2.86	*59.44±3.54	58.62±3.64	60.34±3.74	*57.29±4.82	*56.48±4.17
	Logistic	67.68±3.49	*32.29±7.31	*34.06±8.87	*30.04±6.18	*29.75±2.88	*29.35±6.20
Waveform	C4.5	74.78±0.61	74.64±0.74	73.38±1.24	*72.14±0.93	*71.39±0.77	*70.66±1.25
	PART	76.60±1.14	73.40±0.98	*73.22±0.67	*72.60±1.06	*71.98±0.75	*70.89±1.74
	NB	80.65±0.67	*75.42±3.11	*70.29±2.55	*66.25±1.77	*64.93±1.58	*63.90±2.85
	Logistic	85.07±0.74	*50.03±6.81	*46.08±6.16	*42.91±6.54	*35.35±3.06	*44.95±8.57
German credit	C4.5	71.67±1.28	*69.98±1.80	70.08±1.83	69.39±2.22	*69.19±1.49	69.97±2.12
	PART	69.91±1.97	70.83±1.69	69.78±2.31	69.25±1.96	68.81±2.18	70.23±1.54
	NB	74.67±1.89	70.56±2.89	*71.12±2.38	*71.08±2.23	*70.99±1.58	71.61±1.47
	Logistic	74.19±1.22	67.14±7.78	68.34±8.58	71.03±2.73	70.52±2.20	71.12±1.58
Adult	C4.5	85.96±0.19	85.88±0.21	85.73±0.22	*85.61±0.19	*85.48±0.20	*85.32±0.19
	PART	85.81±0.18	85.91±0.25	85.79±0.18	85.63±0.27	85.55±0.13	*85.28±0.23
	NB	83.77±0.19	85.79±0.76	*86.05±0.41	*85.98±0.49	*86.01±0.52	*85.88±0.36
	Logistic	86.92±0.12	78.31±18.25	84.34±3.12	85.30±0.88	84.80±1.03	*85.00±0.45

TDR [11], and kADET [13] in terms of classification accuracy. All experiments are based on the execution of the original software obtained from the corresponded algorithms inventors.

Because TDS, TDR, and kADET all require the user to provide a generalization taxonomy, we could use them only on the Adult data set for which a generalization taxonomy was previously provided by the algorithms inventors.

For the induction phase (box 7 in Fig. 7), we examined the following base induction algorithms: PART decision list [37], Naïve Bayes and C4.5 [34], and Logistics Regression.

The C4.5 algorithm was selected because it is considered a state-of-the-art decision tree algorithm and has been widely used in many other comparative studies. Moreover, in our method, it is also wrapped in order to generate the anonymity rules. Thus, we expected that C4.5 would show the minimal deterioration in classification accuracy. Naïve Bayes was selected due to its simplicity and the fact that it uses a quite different learning bias from the one used by C4.5. The PART algorithm was used due to its resemblances to C4.5, thus allowing us to examine the effectiveness of the proposed method on various induction biases: C4.5 (using exactly the same inducer as the one used internally (wrapped) by kACTUS), PART (using a similar inducer as wrapped by kACTUS), naïve Bayes, and Logistics Regression (which has quite a different induction bias). All experiments were performed in the WEKA environment [38]. The experiments with C4.5 took place using J48, the Java version of C4.5.

5.5 The Effect of k on the Accuracy

In this section, we analyze the effect of the value of k (anonymity level) on the accuracy. Table 2 shows the accuracy results obtained by the proposed algorithm on six different values of k for various data sets using different inducers. In this section, we assume that all the input attributes are quasi-identifiers. Note that the column with $k = 1$ represents the original result (i.e., when no anonymity has been performed) to make the examination of the effect of anonymity on the accuracy of results possible. The superscript "*" indicates that the degree of accuracy of the original data set was significantly different from the corresponding result at a confidence level of 95 percent.

As expected, the results indicate that there is a trade-off between accuracy performance and the anonymity level. Namely, increasing the anonymity level decreases accuracy. Moreover, usually, there is correlation between the accuracy deterioration and the resemblances of the inducer used for the classifier learning to the inner inducer (our algorithm wraps the C4.5 algorithm). However, surprisingly in the Japanese Credit, this correlation does not hold. In this data set, the logistics regression classifier improves its classification accuracy despite the anonymity constraint, while the C4.5 classifier does not manifest a similar behavior. This indicates that for noisy data sets, the fact that kACTUS wraps a decision tree inducer; make it useful as a feature selection for simple classifiers such as logistics regression and naïve Bayes.

5.6 Single Dimensional Generalization Schemes

In this section, we compare the proposed algorithm to other existing k-anonymity algorithms. First, we compare the performance on the Adult data set for different sizes of quasi-identifier sets and k-values on various inducers. Following [11], the various quasi-identifier sets are based on the impact of the attributes on classification. Specifically, the label "14/14" refers to a quasi-identifier set that contains all input attributes. The label "11/14" refers to a quasi-identifier set that contains the attributes {Age, Workclass, fnlwgt, Education, Education-num, Marital-status, Occupation, Sex, Capital-gain, Hours-per-week, Native-country}. The label "8/14" refers to a quasi-identifier set that contains the attributes {Age, Workclass, fnlwgt, occupation, sex, Capital-gain, Hours-per-week, Native-country}. All remaining attributes are included in the training set but are treated as nonquasi-identifiers. Table 3 specifies the results obtained on the Adult data set for comparing kACTUS with TDS, TDR, and kADET, respectively. Note that due to the restriction of kADET, we could only compare the performance on the C4.5 inducer (as kADET is embedded in this algorithm). The superscript "+" indicates that the accuracy of kACTUS was significantly higher than the corresponding algorithm (with the same k, the same base inducer, and the same data set) at a confidence level of 95 percent. The "−" superscript indicates that the accuracy was significantly lower. The results of the experimental study are encouraging. They indicate that there is only one significant case where TDS is more accurate and two cases when TDR is more accurate than kACTUS. On the other cases, kACTUS is significantly more accurate than TDS and TDR in 51 and 52 cases out of 72 cases, respectively. As expected, smaller quasi-identifier sets obtain a higher accuracy level. This can be explained by the fact that larger quasi-identifiers place a tighter restriction on the anonymity

TABLE 3
Comparing Accuracy with Generalization Methods

Case	Alg.	k=5	k=20	k=50	k=100	k=500	k=1000
C4.5 QE8/14	TDS	83.01±0.21	83.07±0.3	82.79±0.2	82.79±0.2	82.72±0.2	82.61±0.29
	TDR	84.63±0.92	84.59±0.89	82.91±1	82.47±0.2	82.47±0.2	82.51±0.21
	kADET	82.80±0.3	82.55±0.3	82.54±0.3	82.49±0.3	82.49±0.3	82.39±0.21
	kACTUS	86.01±0.19	85.74±0.28	85.31±0.27	84.62±0.27	84.61±0.26	83.00±1.05
C4.5 QE11/14	TDS	80.55±1.96	81.37±1.9	82.46±0.3	82.60±0.2	81.36±0.2	81.36±0.19
	TDR	82.56±0.2	82.56±0.3	82.58±0.2	82.45±0.2	82.60±0.2	81.37±0.31
	kADET	82.78±0.4	80.90±2.4	82.19±0.6	81.87±0.7	81.81±1.66	81.54±0.76
	kACTUS	86.01±0.29	85.75±0.22	85.47±0.23	84.79±0.24	84.08±0.23	80.95±1.43
C4.5 QE14/14	TDS	77.29±2.8	80.48±0.2	79.93±1.7	75.21±0.2	75.21±0.2	80.38±0.18
	TDR	77.45±2.2	77.37±0.8	78.51±3.1	77.24±1.6	75.21±0.2	75.21±0.2
	kADET	82.57±0.3	81.76±0.3	81.63±0.3	81.53±0.2	81.52±0.2	80.38±0.18
	kACTUS	85.88±0.21	85.48±0.18	84.89±0.23	84.25±0.24	82.43±0.19	79.27±1.42
Naive B. QE8/14	TDS	81.17±0.4	81.18±0.3	80.42±0.6	80.42±0.6	80.45±0.1	81.35±0.2
	TDR	82.75±0.7	82.73±0.6	80.84±0.2	80.83±0.2	80.91±0.3	80.93±1.4
	kACTUS	85.45±0.15	85.37±0.23	85.11±0.24	84.83±0.41	83.10±0.75	85.45±0.15
Naive B. QE11/14	TDS	77.79±3.42	79.50±3.27	81.43±0.2	75.41±3.2	78.29±3.64	78.29±3.64
	TDR	81.43±0.3	81.71±0.2	81.78±0.6	82.13±0.40	82.35±0.2	81.40±0.2
	kACTUS	84.79±0.42	84.54±0.56	83.59±1.19	83.59±1.19	77.41±0.16	76.66±0.42
Naive B. QE14/14	TDS	73.90±4.1	80.41±0.3	79.61±2.7	71.44±0.2	75.18±0.2	80.38±0.2
	TDR	76.43±2.8	74.71±0.9	76.61±3.94	74.68±1.9	72.53±0.2	73.35±1.3
	kACTUS	85.79±0.76	86.01±0.52	85.64±0.36	84.84±0.29	83.01±0.22	79.74±0.62
Logistics. QE8/14	TDS	83.69±0.1	83.57±0.2	83.25±0.2	83.25±0.2	83.21±0.2	83.03±0.23
	TDR	84.83±0.77	84.82±0.73	82.85±0.2	82.85±0.2	82.82±0.2	82.79±0.22
	kACTUS	86.02±0.38	85.94±0.30	85.50±0.39	84.45±0.57	83.75±0.18	82.98±0.47
Logistics. QE11/14	TDS	80.71±2.05	81.57±1.96	82.71±0.2	82.68±0.2	81.34±0.2	81.34±0.21
	TDR	82.82±0.16	82.64±0.2	82.67±0.2	82.67±0.19	82.66±0.20	82.10±0.64
	kACTUS	80.53±15.56	85.54±0.53	85.24±0.39	83.83±1.02	82.80±0.28	80.51±0.80
Logistics. QE14/14	TDS	77.28±2.86	78.31±0.6	79.93±1.7	75.18±0.2	75.18±0.2	80.38±0.18
	TDR	77.46±2.28	77.84±0.5	78.64±1.5	77.50±0.8	77.26±0.1	75.21±0.2
	kACTUS	78.31±18.25	84.80±1.03	84.62±0.34	82.15±0.98	81.42±0.23	78.89±0.82
PART QE8/14	TDS	83.08±0.1	83.19±0.2	82.86±0.2	82.86±0.2	82.88±0.2	82.83±0.24
	TDR	84.68±0.8	84.65±0.68	82.64±0.2	82.63±0.2	82.64±0.2	82.63±0.20
	kACTUS	85.92±0.28	85.66±0.24	85.46±0.24	84.76±0.29	84.71±0.22	83.14±1.03
PART QE11/14	TDS	80.60±2.02	81.43±1.91	82.61±0.2	82.62±0.2	81.34±0.2	81.34±0.19
	TDR	82.68±0.2	82.62±0.2	82.63±0.2	82.60±0.2	82.63±0.2	81.43±0.30
	kACTUS	86.04±0.21	85.85±0.18	85.46±0.25	84.97±0.20	84.00±0.24	81.69±1.78
PART QE14/14	TDS	77.29±2.8	80.48±0.2	79.93±1.7	75.21±0.2	75.21±0.2	80.38±0.2
	TDR	77.46±2.2	77.37±0.8	78.51±3.1	77.24±1.6	75.21±0.2	75.21±0.2
	kACTUS	85.91±0.25	85.55±0.13	84.84±0.33	83.90±0.24	83.20±0.15	78.28±0.44

process. Table 3 indicates that there is no significant case where kADET is more accurate than kACTUS. On the other hand, kACTUS is significantly more accurate than kADET in 13 cases out of 18 cases. The results indicate that from the accuracy perspective, kACTUS should be preferred for small values of k.

In order to conclude which algorithm performs the best over multiple cases, we followed the procedure proposed in [39], [40]. In the case of multiple classifiers, we first used the adjusted Friedman test in order to reject the null hypothesis that all algorithms perform the same, and then, the Bonferroni-Dunn test to examine whether the new algorithm performs significantly better than existing algorithms. The null hypothesis, that all classifiers perform the same and the observed differences are merely random, was rejected using the adjusted Friedman test. We proceeded with the Bonferroni-Dunn test and found that kACTUS statistically outperforms TDS, TDR, and kADET with a 95 percent confidence level.

5.7 Multidimensional Generalization Schemes

In the previous section, we have shown that kACTUS outperforms single-dimensional generalization schemes. In this section, we compare the predictive performance of kACTUS to the Mondrian algorithm which is a multi-dimensional k-anonymity [28].

Table 4 presents the obtained accuracy results. On average, the accuracy of kACTUS is 7.84 percent higher than the accuracy of Mondrian. There is no case in which Mondrian significantly outperforms kACTUS. On the other hand, kACTUS significantly outperforms Mondrian in 58 out of 72 cases. Using Bonferroni-Dunn test, we concluded that kACTUS performs significantly better than existing algorithms.

TABLE 4
Comparing Accuracy with a Multidimensional Method

Case	Alg.	k=5	k=20	k=50	k=100	k=500	k=1000
C4.5 QE8/14	kACTUS	86.01±0.19	85.74±0.28	85.31±0.27	84.62±0.27	84.61±0.26	83.00±1.05
	Mondrian	82.86±0.27	82.86±0.19	82.86±0.17	82.85±0.19	82.73±0.26	77.60±0.63
C4.5 QE11/14	kACTUS	86.01±0.26	85.75±0.22	85.47±0.23	84.79±0.24	84.08±0.23	80.95±1.43
	Mondrian	83.09±0.13	82.46±1.73	80.93±2.94	79.48±2.76	79.19±2.95	77.20±0.62
C4.5 QE14/14	kACTUS	85.88±0.21	85.48±0.18	84.89±0.23	84.25±0.24	82.43±0.19	79.27±1.42
	Mondrian	82.17±0.21	82.15±0.18	81.31±0.45	81.11±0.16	75.18±0.22	75.21±0.19
Naive B. QE8/14	kACTUS	85.45±0.15	85.37±0.23	85.11±0.24	84.83±0.41	83.10±0.75	85.45±0.15
	Mondrian	77.29±0.98	78.00±0.34	77.98±0.40	77.66±0.40	77.08±1.51	76.13±0.26
Naive B. QE11/14	kACTUS	84.79±0.42	84.54±0.56	83.59±1.19	83.59±1.19	77.41±0.16	76.66±0.42
	Mondrian	69.61±3.57	65.82±7.73	63.74±6.36	63.19±4.94	70.15±6.27	59.06±9.82
Naive B. QE14/14	kACTUS	85.79±0.76	86.01±0.52	85.64±0.36	84.84±0.29	83.01±0.22	79.74±0.62
	Mondrian	44.15±2.80	47.49±2.76	47.20±1.63	47.12±4.05	53.08±5.54	39.37±15.28
Logistics. QE8/14	kACTUS	86.02±0.38	85.94±0.30	85.50±0.39	84.45±0.57	83.75±0.18	82.98±0.47
	Mondrian	83.69±0.23	83.68±0.23	83.59±0.19	83.30±0.28	82.82±0.29	82.68±0.18
Logistics. QE11/14	kACTUS	80.53±15.56	85.54±0.53	85.24±0.39	83.83±1.02	82.80±0.28	80.51±0.80
	Mondrian	82.16±1.22	82.36±1.74	81.92±1.07	81.20±2.14	79.82±2.47	79.38±0.19
Logistics. QE14/14	kACTUS	78.31±18.25	84.80±1.03	84.62±0.34	82.15±0.98	81.42±0.23	78.89±0.82
	Mondrian	80.29±2.03	78.18±2.17	75.15±2.45	73.88±2.07	73.54±4.25	75.21±0.19
PART QE8/14	kACTUS	85.92±0.28	85.66±0.24	85.46±0.24	84.76±0.29	84.71±0.22	83.14±1.03
	Mondrian	82.29±0.84	82.53±0.59	82.52±0.46	82.67±0.41	82.56±0.48	82.83±0.24
PART QE11/14	kACTUS	86.04±0.21	85.85±0.18	85.46±0.25	84.97±0.20	84.00±0.24	81.69±1.78
	Mondrian	81.60±1.57	81.68±1.52	81.32±0.83	79.70±1.32	78.20±4.00	75.39±2.91
PART QE14/14	kACTUS	85.91±0.25	85.55±0.13	84.84±0.33	83.90±0.24	83.20±0.15	78.28±0.44
	Mondrian	81.64±0.64	80.81±0.64	80.96±0.89	80.34±0.88	77.21±0.19	75.21±0.19

TABLE 5
Comparing Accuracy with Suppression Methods

Case	Alg.	k=5	k=20	k=50	k=100	k=500	k=1000
C4.5 QE8/14	TDS	82.84±0.22	82.79±0.22	82.79±0.22	82.79±0.22	82.72±0.23	82.43±0.23
	TDR	85.29±0.37	84.71±0.23	84.66±0.23	84.65±0.23	82.54±0.20	82.53±0.21
	kADET	82.83±0.30	82.52±0.32	82.52±0.32	82.52±0.32	82.50±0.32	82.39±0.21
	kACTUS	86.01±0.19	85.74±0.28	85.31±0.27	84.62±0.27	84.61±0.26	83.00±1.05
C4.5 QE11/14	TDS	77.41±0.20	79.32±1.56	79.49±1.53	79.49±1.53	79.45±1.45	77.41±0.20
	TDR	82.26±0.41	82.17±0.37	82.08±0.31	81.64±0.26	77.41±0.20	77.41±0.20
	kADET	81.08±2.57	81.97±1.73	81.53±2.36	81.31±2.25	81.27±2.27	75.73±0.83
	kACTUS	86.01±0.20	85.75±0.22	85.47±0.23	84.79±0.24	84.08±0.23	80.95±1.43
C4.5 QE14/14	TDS	77.82±2.07	75.21±0.19	75.21±0.19	75.21±0.19	75.21±0.19	75.21±0.19
	TDR	78.16±2.63	80.96±0.50	79.89±0.32	79.44±0.25	75.21±0.19	75.21±0.19
	kADET	82.58±0.32	81.78±0.28	80.96±0.31	80.84±0.22	75.21±0.19	75.21±0.19
	kACTUS	85.88±0.21	85.48±0.18	84.89±0.23	84.25±0.24	82.43±0.19	79.27±1.42
Naive B. QE8/14	TDS	80.64±0.95	80.41±0.61	80.41±0.61	80.41±0.61	80.45±0.14	77.97±1.83
	TDR	82.93±0.48	79.93±0.38	81.34±1.56	82.58±0.57	77.98±1.06	78.52±0.49
	kACTUS	85.45±0.15	85.37±0.23	85.11±0.24	84.83±0.41	83.10±0.75	85.45±0.15
Naive B. QE11/14	TDS	73.04±0.24	76.63±2.92	76.88±2.71	76.88±2.71	79.19±1.69	73.36±0.26
	TDR	74.33±0.64	70.12±5.70	76.52±3.86	78.41±3.5	80.12±0.23	73.03±0.26
	kACTUS	84.79±0.42	84.54±0.56	83.59±1.19	83.59±1.19	77.41±0.16	76.66±0.42
Naive B. QE14/14	TDS	77.40±2.75	74.86±1.09	74.86±1.09	74.86±1.09	75.18±0.22	75.21±0.19
	TDR	76.42±4.69	78.87±0.58	79.53±0.52	77.92±1.21	77.34±0.22	65.28±3.02
	kACTUS	85.79±0.76	86.01±0.52	85.64±0.36	84.84±0.29	83.01±0.22	79.74±0.62
Logistics. QE8/14	TDS	83.26±0.19	83.25±0.16	83.25±0.16	83.25±0.16	83.21±0.16	82.76±0.29
	TDR	83.81±0.90	84.14±0.65	84.12±0.66	84.13±0.67	82.79±0.22	82.78±0.21
	kACTUS	86.02±0.38	85.94±0.30	85.50±0.39	84.45±0.57	83.75±0.18	82.98±0.47
Logistics. QE11/14	TDS	76.36±0.21	79.41±1.70	79.56±1.58	79.56±1.58	79.52±1.58	77.37±0.20
	TDR	80.52±0.88	80.55±0.85	80.52±0.89	80.36±0.50	77.37±0.20	77.37±0.21
	kACTUS	80.53±15.56	85.54±0.53	85.24±0.39	83.83±1.02	82.80±0.28	80.51±0.80
Logistics. QE14/14	TDS	77.79±2.68	75.18±0.22	75.18±0.22	75.18±0.22	75.18±0.22	75.21±0.19
	TDR	76.91±3.97	78.06±1.06	78.00±1.08	77.89±0.75	75.21±0.19	75.21±0.19
	kACTUS	78.31±18.25	84.80±1.03	84.62±0.34	82.15±0.98	81.42±0.23	78.89±0.82
PART QE8/14	TDS	82.87±0.19	82.86±0.22	82.86±0.22	82.86±0.22	82.88±0.19	82.57±0.26
	TDR	84.07±0.82	84.70±0.29	84.69±0.29	84.69±0.27	82.63±0.17	82.62±0.17
	kACTUS	85.92±0.28	85.66±0.24	85.46±0.24	84.76±0.29	84.71±0.22	83.14±1.03
PART QE11/14	TDS	77.38±0.19	79.35±1.67	79.49±1.61	79.49±1.61	79.47±1.55	77.35±0.20
	TDR	80.17±1.10	80.57±1.10	80.77±1.31	80.77±1.21	77.35±0.20	77.37±0.20
	kACTUS	86.04±0.21	85.85±0.18	85.46±0.25	84.97±0.20	84.00±0.24	81.69±1.78
PART QE14/14	TDS	77.85±2.05	75.21±0.19	75.21±0.19	75.21±0.19	75.21±0.19	75.21±0.19
	TDR	76.97±1.89	78.06±1.06	78.00±1.08	77.54±0.66	75.21±0.19	75.21±0.19
	kACTUS	85.91±0.25	85.55±0.13	84.84±0.33	83.90±0.24	83.20±0.15	78.28±0.44

5.8 Comparing to Suppression Methods

It is expected that generalization methods outperform suppression methods because in the former case, additional knowledge (the generalization tree) is provided to the k-anonymity algorithm and the algorithm has the choice for not entirely suppressing the datum but gently generalizes its value. In fact, one of the features of TDR is to automatically decide if it is better to perform suppression or generalization.

In this section, we examine if the improved performance of kACTUS is not derived from the fact that our algorithm uses suppression approach, while other use generalization. Specifically, in this section, all generalization algorithms are provided with one level of generalization trees, i.e., each value is generalized to the "?" value. Practically, this means that these algorithms become suppression algorithms. Table 5 presents the results. As can be seen, there is only

TABLE 6
Scalability Analysis Results

Scalability factor	C4.5 training time (sec)	k-anonymity time (sec)	Dataset size
5	6.88	44.32	226110
10	15.57	99.16	452220
15	28.48	192.82	678330
20	41.23	247.94	904440
25	62.47	409.32	1130550
30	69.43	483.01	1356660

one significant case where TDR is more accurate than kACTUS. On the other hand, there is no case in which TDS or kADET is significantly better than kACTUS. Moreover kACTUS is significantly more accurate than TDS and TDR in 62 and 59 cases out of 72 cases, respectively. Comparing this result to the previous one (when a full generalization tree is used) indicates that generalization tree does improve the performance of TDS, TDR, and kADET. More important when the contributing impact of the detailed generalization tree is deactivated, the superiority of kACTUS becomes clearer.

5.9 Scalability

The aim of this section is to examine the actual computational cost of the proposed algorithm by measuring the running time, and to examine its ability to handle a growing size of data set in a graceful manner. We performed a scalability test proposed in [11] to measure runtime cost of the algorithm on large data sets. The original Adult data set with 45,222 records and seven quasi-identifiers was expanded as follows: for every original record t, we added $\sigma - 1$ variations, where σ is a scale factor. Together with all original records, the enlarged data set had $\sigma \times 45{,}222$ records. Each variation of t was generated by combining ρ attributes from the original record, while the values of the remaining attributes were randomly drawn from the domain of the attributes.

We conducted all experiments on the following hardware configuration: a desktop computer implementing a Windows XP operating system with Intel Pentium 4-2.8 GHz and 2 GB of physical memory. The data set was loaded from MySql RDBMS Ver. 6 with indexes defined on all quasi-identifiers.

Table 6 presents the time measured (in seconds) for various values of σ with $\rho = 3$ and k-threshold $= 150$. We measured separately the time needed to generate the decision tree (referred to as C4.5 training time) and the time needed to perform our k-anonymity procedure. Model generation time reflects the runtime cost of J4.8 inducer in Weka package and it is beyond our control. The results indicate that the execution time is almost linear in the number of records. This agrees with Corollary 2 which indicates that the computational complexity overhead of the kACTUS algorithm is linear in the training set size. The largest execution time of 483 seconds was achieved for Scalability Factor of 30 with 1,356,660 records, which is comparable to the execution time reported by TDS.

5.10 Discussions

The advantages of the new kACTUS algorithm, as observed from the experimental study, can be summarized as following:

- kACTUS is capable of applying k-anonymity on a given table with no significant effect on classification accuracy.
- kACTUS scales well with large data sets and anonymity levels.
- When compared to the state-of-the-art k-anonymity methods, kACTUS anonymized data can be used to induce classifiers which are of an equivalent or slightly higher degree of accuracy.
- kACTUS, unlike other methods, does not use any prior knowledge. In TDS, TDR, kADET, GA-based anonymizer [10], and Incognito [12], the user is required to provide a taxonomy tree for categorical attributes. This makes it difficult to use. Additionally, it can become a source for disagreements among experts as described in [7], [13].
- When compared to the kADET algorithm, kACTUS is not restricted to a decision tree classifier, and its output can be used by any induction algorithms.
- Moreover, kADET embeds k-anonymity into the induction algorithm as part of the tree growing phase (via the split criterion). kACTUS, on the contrary, takes the pruning approach. Thus, kACTUS can wrap any top-down univariate decision tree inducer as a black box. It does not require revision of existing algorithms, where kADET, which embeds new k-anonymity splitting criteria into an existing decision trees inducer, requires increased effort from the practitioner when moving from one decision tree inducer to another. Moreover, some decision tree inducers are very complicated to manipulate. It is not surprising that to use kADET with the C4.5 algorithm, the authors were forced to disable a certain feature of the original C4.5. Taking the pruning approach, we could wrap C4.5 without disabling any feature and without revising its source.

The kACTUS algorithm has also several drawbacks:

- Overanonymity: When a certain node does not comply with the k-anonymity restriction, kACTUS prunes it. This might be too aggressive and might result in overanonymity of the data. Instead, one can generalize the attribute and not suppressing it. For example, consider an internal node with the following leaves "A1 = Yellow (6)," "A1 = Blue (7)," and "A1 = Red (10)." Assuming that k = 10, then the "A1" of the first two leaves will be totally suppressed. However, it is sufficient to replace this attributes with the value: "Yellow OR Blue" which implies a smaller data loss. A different origin for overgeneralization happens if the original classification tree is already overanonymous. The proposed algorithm is capable of pruning a node and making its scope wider, but it cannot expand the node to narrow its scope.
- Instances loss: While there are some cases when instances loss cannot be avoided, the greedy

nature of kACTUS can lead to unnecessary instances loss. In some decision trees, splitting criteria (such as the information gain in ID3 algorithm) attributes, which take on a large number of distinct values, tend to be selected near the root of the tree. This well-known observation can cause many nodes to be totally pruned in kACTUS (i.e., instances loss). Using information gain ratio instead (as in the C4.5 algorithm that has been used in the experimental study) biases the decision tree against considering attributes with a large number of distinct values, and thus, it reduces the need to remove instances. Nevertheless, as stated in Corollary 3, the instances loss is bounded by k.

- In two points, kACTUS performs random selection of instances. Results might be improved if a greedy selection rule is used. Moreover, this randomness injection may cause the performance to be unstable. Nevertheless, it should be noted that the last weakness has not manifested in the experimental studies presented above. As a matter of fact, the standard deviation of kACTUS is quite similar to that of other anonymization methods.

6 Conclusion

In this paper, we presented a new method for preserving the privacy in classification tasks using k-anonymity. The proposed method requires no prior knowledge regarding the domain hierarchy taxonomy and can be used by any inducer. The new method also shows a higher predictive performance when compared to existing state-of-the-art methods.

Additional issues to be studied further include: Examining kACTUS with other decision trees inducers; revising kACTUS to overcome its existing drawbacks; extending the proposed method to other data mining tasks (such as clustering and association rules) and to other anonymity measures (such as l-diversity) which respond to different known attacks against k-anonymity, such as homogeneous attack and background attack.

Acknowledgments

The authors gratefully thank Benjamin C.M. Fung, Ke Wang, and Philip S. Yu for providing their proprietary software packages for evaluating TDS and TDR algorithms. They gratefully thank Arik Friedman and Ran Wolff from Technion–Israel Institute of Technology and Assaf Schuster from Haifa University for providing their proprietary software packages for evaluating kADET. They also gratefully thank Kristen LeFevre for providing her proprietary software package for evaluating Mondrian algorithm.

References

[1] M. Kantarcioglu, J. Jin, and C. Clifton, "When Do Data Mining Results Violate Privacy?" *Proc. 2004 Int'l Conf. Knowledge Discovery and Data Mining*, pp. 599-604, 2004.

[2] L. Rokach, R. Romano, and O. Maimon, "Negation Recognition in Medical Narrative Reports," *Information Retrieval*, vol. 11, no. 6, pp. 499-538, 2008.

[3] M.S. Wolf and C.L. Bennett, "Local Perspective of the Impact of the HIPAA Privacy Rule on Research," *Cancer-Philadelphia Then Hoboken*, vol. 106, no. 2, pp. 474-479, 2006.

[4] P. Samarati and L. Sweeney, "Generalizing Data to Provide Anonymity When Disclosing Information," *Proc. 17th ACM SIGACT-SIGMOD-SIGART Symp. Principles of Database Systems*, vol. 17, p. 188, 1998.

[5] L. Sweeney, "k-Anonymity: A Model for Protecting Privacy," *Int'l J. Uncertainty, Fuzziness, and Knowledge-Based Systems*, vol. 10, no. 5, pp. 557-570, 2002.

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

[8] K. Wang, P.S. Yu, and S. Chakraborty, "Bottom-Up Generalization: A Data Mining Solution to Privacy Protection," *Proc. Fourth IEEE Int'l Conf. Data Mining*, pp. 205-216, 2004.

[9] L. Tiancheng and I. Ninghui, "Optimal K-Anonymity with Flexible Generalization Schemes through Bottom-Up Searching," *Proc. Sixth IEEE Int'l Conf. Data Mining Workshops*, pp. 518-523, 2006.

[10] S.V. Iyengar, "Transforming Data to Satisfy Privacy Constraints," *Proc. Eighth ACM SIGKDD*, pp. 279-288, 2002.

[11] B.C.M. Fung, K. Wang, and P.S. Yu, "Anonymizing Classification Data for Privacy Preservation," *IEEE Trans. Knowledge and Data Eng.*, vol. 19, no. 5, pp. 711-725, May 2007.

[12] K. LeFevre, D.J. DeWitt, and R. Ramakrishnan, "Incognito: Efficient Full Domain k-Anonymity," *Proc. 2005 ACM SIGMOD*, pp. 49-60, 2005.

[13] A. Friedman, R. Wolff, and A. Schuster, "Providing k-Anonymity in Data Mining," *Int'l J. Very Large Data Bases*, vol. 17, no. 4, pp. 789-804, 2008.

[14] R. Kohavi and G.H. John, "Wrappers for Feature Subset Selection," *Artificial Intelligence*, vol. 97, nos. 1/2, pp. 273-324, 1997.

[15] V.S. Verykios, E. Bertino, I.N. Fovino, L.P. Provenza, Y. Saygin, and Y. Theodoridis, "State-of-the-Art in Privacy Preserving Data Mining," *ACM SIGMOD Record*, vol. 33, no. 1, pp. 50-57, 2004.

[16] A. Agrawal and R. Srikant, "Privacy Preserving Data Mining," *ACM SIGMOD Record*, vol. 29, no. 2, pp. 439-450, 2000.

[17] B. Gilburd, A. Schuster, and R. Wolff, "k-TTP: A New Privacy Model for Large-Scale Distributed Environments," *Proc. 10th ACM SIGKDD*, pp. 563-568, 2004.

[18] Z. Yang, S. Zhong, and R.N. Wright, "Privacy-Preserving Classification of Customer Data without Loss of Accuracy," *Proc. Fifth Int'l Conf. Data Mining*, 2005.

[19] J. Roberto, Jr. Bayardo, and A. Rakesh, "Data Privacy through Optimal k-Anonymization," *Proc. Int'l Conf. Data Eng.*, vol. 21, pp. 217-228, 2005.

[20] A. Blum, C. Dwork, F. McSherry, and K. Nissim, "Practical Privacy: The SuLQ Framework," *Proc. 24th ACM SIGMOD-SIGACT-SIGART Symp. Principles of Database Systems*, pp. 128-138, June 2005.

[21] S. Chawla, C. Dwork, F. McSherry, A. Smith, and H. Wee, "Toward Privacy in Public Databases," *Proc. Theory of Cryptography Conf.*, pp. 363-385, 2005.

[22] K. Wang, B.C.M. Fung, and P.S. Yu, "Template-Based Privacy Preservation in Classification Problems," *Proc. Fifth IEEE Int'l Conf. Data Mining*, pp. 466-473, 2005.

[23] E. Bertino, B.C. Ooi, Y. Yang, and R.H. Deng, "Privacy and Ownership Preserving of Outsourced Medical Data," *Proc. Int'l Conf. Data Eng.*, vol. 21, pp. 521-532, 2005.

[24] G. Aggarwal, A. Feder, K. Kenthapadi, R. Motwani, R. Panigrahy, D. Thomas, and A. Zhu, "Approximation Algorithms for k-Anonymity," *J. Privacy Technology*, 2005.

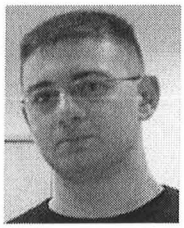
[25] A. Meyerson and R. Williams, "On the Complexity of Optimal k-Anonymity," *Proc. 23rd ACM SIGMOD-SIGACT-SIGART Symp. Principles of Database Systems*, pp. 223-228, 2004.

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

[27] K. LeFevre, D.J. DeWitt, and R. Ramakrishnan, "Mondrian Multidimensional k-Anonymity," *Proc. 22nd Int'l Conf. Data Eng.*, p. 25, Apr. 2006.

[28] K. LeFevre, D.J. DeWitt, and R. Ramakrishnan, "Workload-Aware Anonymization," *Proc. 12th ACM SIGKDD*, pp. 277-286, 2006.

[29] L. Sweeney, "Datafly: A System for Providing Anonymity in Medical Data," *Proc. IFIP TC11 WG11.3 11th Int'l Conf. Database Security XI: Status and Prospects*, pp. 356-381, 1997.
[30] P. Sharkey, H. Tian, W. Zhang, and S. Xu, "Privacy-Preserving Data Mining through Knowledge Model Sharing," *Privacy, Security and Trust in KDD*, pp. 97-115, Springer, 2008.
[31] S. Grumbach and T. Milo, "Towards Tractable Algebras for Bags," *J. Computer and System Sciences*, vol. 52, no. 3, pp. 570-588, 1996.
[32] Y. Du, T. Xia, Y. Tao, D. Zhang, and F. Zhu, "On Multidimensional k-Anonymity with Local Recoding Generalization," *Proc. Int'l Conf. Data Eng. (ICDE)*, pp. 1422-1424, 2007.
[33] L. Rokach, L. Naamani, and A. Shmilovici, "Pessimistic Cost-Sensitive Active Learning of Decision Trees," *Data Mining and Knowledge Discovery*, vol. 17, no. 2, pp. 283-316, 2008.
[34] J.R. Quinlan, *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.
[35] E. Alpaydin, "Combined 5×2 cv F Test for Comparing Supervised Classification Learning Classifiers," *Neural Computation*, vol. 11, no. 8, pp. 1885-1892, 1999.
[36] A. Asuncion and D.J. Newman, "UCI Machine Learning Repository," School of Information and Computer Science, Univ. of California, http://mlearn.ics.uci.edu/MLRepository.html, 2007.
[37] E. Frank and I.H. Witten, "Generating Accurate Rule Sets without Global Optimization," *Proc. 15th Int'l Conf. Machine Learning*, pp. 144-151, 1998.
[38] I.H. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools*. Morgan Kaufmann, 2005.
[39] J. Demsar, "Statistical Comparisons of Classifiers over Multiple Data Sets," *J. Machine Learning Research*, vol. 7, pp. 1-30, 2006.
[40] L. Rokach, "Genetic Algorithm-Based Feature Set Partitioning for Classification Problems," *Pattern Recognition*, vol. 41, no. 5, pp. 1693-1717, 2008.



Slava Kisilevich received the MSc degree in information systems from Ben-Gurion University, Beer-Sheva, Israel, in 2008. He is currently working toward the PhD degree in the "Databases, Data Mining and Visualization" Group of Professor Daniel Keim at Konstanz University, Germany. His research interests include data mining, information retrieval, recommender systems, geospatial analysis of moving objects, geographic data analysis, and visualization.

Lior Rokach received the PhD degree in industrial engineering from Tel Aviv University. He is a senior lecturer in the Department of Information System Engineering and the Program of Software Engineering at Ben-Gurion University. He is a recognized expert in intelligent information systems, and served in several leading positions in this field. Dr. Rokach is the author of more than 70 refereed papers in leading journals, conference proceedings, and book chapters. In addition, he has also authored six books including *Pattern Classification Using Ensemble Methods* (World Scientific Publishing, 2010), *Data Mining with Decision Trees* (World Scientific Publishing, 2008), and *Decomposition Methodology for Knowledge Discovery and Data Mining* (World Scientific Publishing, 2005). He is also the editor of *The Data Mining and Knowledge Discovery Handbook* (Springer, 2005), *Soft Computing for Knowledge Discovery and Data Mining* (Springer, 2007), and *Recommender Systems Handbook* (Springer, 2010).

Yuval Elovici received the BS and MSc degrees, both in computer and electrical engineering, from Ben-Gurion University and the PhD degree in information systems from Tel Aviv University, Israel. He is is a senior lecturer at Ben-Gurion University's Department of Information System Engineering and the former head of the university's software engineering program. His main areas of interest are computer and network security, information retrieval, and data mining. Dr. Elovici is one of the founding members of Deutsche Telekom Laboratories at Ben-Gurion University of the Negev (Israel) and serves as its director. Currently, 20 senior faculty members and 80 graduate students are involved in the laboratory's activities. Over the past 10 years, he has served as a consultant to several startup companies, to venture capital funds, and the Israeli government. He is the author of more than 90 refereed papers in major journals (e.g., *IEEE Transactions on Knowledge and Data Engineering*, *Physical Review E*, *Information Sciences*, and *IEEE Intelligent Systems*), conference proceedings and book chapters, 40 of which are the field of data mining for information security. He is a member of the IEEE.

Bracha Shapira received the MSc degree in computer science form the Hebrew University, Jerusalem, and the PhD degree in information systems from Ben-Gurion University. She is a senior lecturer in the Department of Information Systems Engineering at Ben-Gurion University of the Negev, Israel, where she leads the Information Retrieval Laboratory. She spent two years at Rutgers University in New Jersey, where she has conducted postdoctoral research in the School of Communication Information and Library Studies. Her articles have been published in referred journals (Jassist, DSS, IP&M, CACM, and more), and her work was presented at professional conferences. Her current research interests include information retrieval and filtering, especially user modeling, profiling, and personalization. She is currently a project manager in the Deutsche Telekom Laboratories at Ben-Gurion University, where she leads a project that deals with personalized content on mobile devices.