Workload-Aware Anonymization Techniques for Large-Scale Datasets

KRISTEN LeFEVRE
University of Michigan
DAVID J. DeWITT
Microsoft
and
RAGHU RAMAKRISHNAN
Yahoo! Research

Protecting individual privacy is an important problem in microdata distribution and publishing. Anonymization algorithms typically aim to satisfy certain privacy definitions with minimal impact on the quality of the resulting data. While much of the previous literature has measured quality through simple one-size-fits-all measures, we argue that quality is best judged with respect to the workload for which the data will ultimately be used.

This article provides a suite of anonymization algorithms that incorporate a target class of workloads, consisting of one or more data mining tasks as well as selection predicates. An extensive empirical evaluation indicates that this approach is often more effective than previous techniques. In addition, we consider the problem of scalability. The article describes two extensions that allow us to scale the anonymization algorithms to datasets much larger than main memory. The first extension is based on ideas from scalable decision trees, and the second is based on sampling. A thorough performance evaluation indicates that these techniques are viable in practice.

Categories and Subject Descriptors: H.2.8 [**Database Management**]: Database Applications

General Terms: Algorithms, Performance, Security

Additional Key Words and Phrases: Databases, privacy, anonymity, data mining, performance, scalability

ACM Reference Format:

LeFevre, K., DeWitt, D. J., and Ramakrishnan, R. 2008. Workload-aware anonymization techniques for large-scale datasets. ACM Trans. Datab. Syst. 33, 3, Article 17 (August 2008), 47 pages. DOI = 10.1145/1386118.1386123 http://doi.acm.org/10.1145/1386118.1386123

This research was conducted while the authors were at the University of Wisconsin – Madison. The work was partially supported by a grant from NSF CyberTrust and an IBM Ph.D. Fellowship.
Author's address: K. LeFevre (corresponding author), University of Michigan; email: klefevre@eecs.umich.edu.

1. INTRODUCTION

Numerous organizations collect, distribute, and publish microdata (personal data in its nonaggregate form) for purposes that include demographic and public health research. Typically, attributes that are known to uniquely identify individuals (e.g., name or social security number) are removed from these datasets. However, data distributors are often concerned with the possibility of using external data to uniquely identify individuals. For example, according to one study, 87% of the population of the Unites States can be uniquely identified on the basis of their 5-digit zip code, sex, and date of birth [Sweeney 2002b]. Thus, even if a person's name is removed from a published record, it may be possible to learn the identity of the person by linking the values of these attributes to an external source of information (e.g., a voter registration database [Sweeney 2002b]).

Conventional wisdom indicates that we can limit the risk of this kind of reidentification by removing certain attributes and coarsening the values of other attributes (e.g., time and geography). For example, the U.S. Health Insurance Portability and Accountability Act (HIPAA) Privacy Rule [HIP 2002] lays out specific attributes that must be removed from medical records when distributing data under a limited use agreement, or more strictly when constructing a deidentified dataset for broader distribution. While they do not satisfy the letter of the HIPAA regulation, k-Anonymity [Samarati 2001; Sweeney 2002b], ℓ-diversity [Machanavajjhala et al. 2006], and extensions [Xiao and Tao 2006; Martin et al. 2007; Chen et al. 2007] represent formalizations of the same intuition.

Subject to the given anonymity constraints, it is of course important that the data remain as useful as possible. It is our position that the best way of measuring quality is based on the task(s) for which the data will ultimately be used. This article proposes using a workload (including data mining tasks and queries) as an evaluation tool. We then develop a suite of techniques for incorporating a target family of workloads into the anonymization process. In addition, we substantially extend our previous work [LeFevre et al. 2006a; LeFevre et al. 2006b] by providing a unified framework that allows these algorithms to scale to datasets significantly larger than main memory.

1.1 Problem Setting

Our proposal is best illustrated through a series of examples. Consider an organization that compiles a database of disease information which could prove useful to external researchers. At the same time, it is important for the agency to take precautions protecting the privacy of patients, for example, hiding the identities of individuals and protecting other sensitive information such as HIV status.

The data distribution procedure can take on a variety of forms, and the amount of trust placed in the data recipient(s) can vary significantly. At one extreme, the data recipient could be a researcher whose work is subject to approval by an Institutional Review Board (IRB) and who has signed a limited use agreement. While it is prudent to apply some anonymization to this data (e.g.,

as indicated informally by the HIPAA rule) to prevent passive reidentification by a rogue user, the risk is fairly low. At the other extreme, the agency might wish to post its data on a public website, in which case the eventual recipient(s) are unknown and comparatively less trustworthy.

Now consider Alice, a researcher who is directing two separate studies. As part of the first study, Alice wants to build a classification model that uses age, smoking history, and HIV status to predict life expectancy. In the second study, she would like to find combinations of variables that are useful for predicting elevated cholesterol and obesity in males over age 40. We consider the problem of producing a single sanitized snapshot that satisfies a given set of privacy requirements but that is useful for the set of tasks in the specified workload.

One might envision a simpler protocol in which Alice requests specific models constructed entirely by the agency. However, in many exploratory data mining tasks (e.g., Chen et al. [2005]), the tasks are not fully specified ahead of time. Perhaps more importantly, the inference implications of releasing multiple predictive models (e.g., decision trees, Bayesian networks, etc.) are not well understood. It is well known that answering multiple aggregate queries may reveal more precise information about the underlying database [Adam and Wortmann 1989; Kenthapadi et al. 2005]. Similarly, each predictive model reveals something about the agency's data. On the other hand, it is appealing to release a single snapshot because there are well-developed notions of anonymity, and the best Alice can do is approximate the distribution in the data she is given.

Finally, it is important to consider the potential risk and implications of collusion. For example, suppose that Bob is the director of another laboratory carrying out a different set of studies and that he requests a separate dataset tailored towards his work. If Alice and Bob were to combine their datasets, they would likely be able to infer more precise information than was intended by the agency. In certain cases (e.g., in the presence of limited use agreements), the risk of collusion by two or more malicious recipients may be sufficiently low to be considered acceptable. In other cases (e.g., publication of data on the web), this is unrealistic. Nevertheless, it is often still useful to tailor a single snapshot to an anticipated class of researchers who are expected to use the data for similar purposes (e.g., AIDS or obesity research). In other words, in this case, we can produce a sanitized snapshot based on the anticipated workloads of all expected recipients with the expectation that this is the only snapshot to be released.

1.2 Article Overview and Contributions

This article is organized as follows. Section 2 reviews some common definitions of anonymity with respect to linking attacks and gives a brief overview of generalization and recoding techniques.

Our first main technical contribution, described in Section 3, is a simple language for describing a family of target workloads and a suite of algorithms for incorporating these workloads into the anonymization process when generating a single anonymized snapshot. It is important to note that, unless special care is taken, publishing multiple sanitized versions of a dataset may be disclosive [Kifer and Gehrke 2006; Wang and Fung 2006; Yao et al. 2005; Xiao

and Tao 2007]. In this article, we do not address the complementary problem of reasoning about disclosure across multiple releases.

We provide an extensive experimental evaluation of data quality in Section 4. The results are promising, indicating that often we do not need to compromise too much utility in order to achieve reasonable levels of privacy.

In Section 5, we turn our attention to performance and scalability, and we describe two techniques for scaling our proposed algorithms to datasets much larger than main memory. An experimental performance evaluation in Section 6 indicates the practicality of these techniques.

The article concludes with discussions of related and future work in Sections 7 and 8.

2. PRELIMINARIES

Consider a single input relation R, containing nonaggregate personal data. As in the majority of previous work, we assume that each attribute in R can be uniquely characterized by at most one of the following types based on knowledge of the application domain.

—*Identifier.* Unique identifiers (denoted *ID*), such as name and social security number are removed entirely from the published data.

—*Quasi-Identifier.* The quasi-identifier is a set of attributes available to the data recipient through other means. Examples include the combination of birth date, sex, and zip code.

—*Sensitive Attribute.* An attribute S is considered sensitive if an adversary is not permitted to uniquely associate its value with an identifier. An example is a patient's disease attribute.

Throughout this article, we consider the problem of producing a sanitized snapshot R^* of R. It is convenient to think of R^* as dividing R into a set of nonoverlapping equivalence classes, each with identical quasi-identifier values. Throughout this article, we will assume multiset (bag) semantics unless otherwise noted.

2.1 Anonymity Requirements

The k-anonymity requirement is quite simple [Samarati 2001; Sweeney 2002b]. Intuitively, it stipulates that no individual in the published data should be identifiable from a group of size smaller than k on the basis of its quasi-identifier values.

Definition 2.1 (*k-Anonymity*). Sanitized view R^* is said to be k-anonymous if each unique tuple in the projection of R^* on $Q_1, \ldots, Q_d$ occurs at least k times.

Although k-anonymity is effective in protecting individual identities, it does not take into account protection of one or more sensitive attributes [Machanavajjhala et al. 2006]. ℓ-Diversity provides a natural extension, incorporating a nominal (unordered categorical) sensitive attribute S. The ℓ-diversity principle requires that each equivalence class in R^* contain at least ℓ well-represented values of sensitive attribute S and can be implemented in several ways

[Machanavajjhala et al. 2006]. Let D_S denote the (finite or infinite) domain of attribute S. The first proposal requires that the entropy of S within each equivalence class be sufficiently large. (We adopt the convention $0 \log 0 = 0$.)

Definition 2.2. Entropy ℓ-Diversity [Machanavajjhala et al. 2006] R^* is entropy ℓ-diverse with respect to S if, for every equivalence class R_i in R^*, $\sum_{s \in D_S} -p(s|R_i) \log p(s|R_i) \geq \log(\ell)$, where $p(s|R_i)$ is the fraction of tuples in R_i with $S = s$.

Entropy ℓ-diversity is often quite restrictive. Because the entropy function is concave, in order to satisfy ℓ-diversity, the entropy of S within the entire dataset must be at least $\log(\ell)$. For this reason, they provide an alternate definition motivated by an elimination attack model. The intuition informing the following definition is as follows: the adversary must eliminate at least $\ell - 1$ sensitive values in order to conclusively determine the sensitive value for a particular individual.

Definition 2.3. Recursive (c, ℓ)-Diversity [Machanavajjhala et al. 2006]. Within an equivalence class R_i, let x_i denote the number of times the ith most frequent sensitive value appears. Given a constant c, R_i satisfies recursive (c, ℓ)-diversity with respect to S if $x_1 < c(x_\ell + x_{\ell+1} + \cdots + x_{|D_S|})$. R^* satisfies recursive (c, ℓ)-diversity if every equivalence class in R^* satisfies recursive (c, ℓ)-diversity. (We say $(c, 1)$-diversity is always satisfied.)

When S is numerically-valued, the definitions provided by Machanavajjhala et al. [2006] do not fully capture the intended intuition. For example, suppose $S = Salary$, and that some equivalence class contains salaries $\{100K, 101K, 102K\}$. Technically, this is considered 3-diverse; however, intuitively, it does not protect privacy as well as an equivalence class containing salaries $\{1K, 50K, 500K\}$.

For this reason, we proposed an additional requirement, which is intended to guarantee a certain level of dispersion of S within each equivalence class.[1] Let $Var(R_i, S) = \frac{1}{|R_i|} \sum_{t \in R_i} (t.S - \overline{S}(R_i))^2$ denote the variance of values for sensitive attribute S among tuples in equivalence class R_i. (Let $\overline{S}(R_i)$ denote the mean value of S in R_i.)

Definition 2.4. Variance Diversity [LeFevre et al. 2006b] An equivalence class R_i is variance diverse with respect to sensitive attribute S if $Var(R_i, S) \geq v$, where v is the diversity parameter. R^* is variance diverse if each equivalence class in R^* is variance diverse.

2.2 Recoding Framework and Greedy Partitioning

In previous work, we proposed implementing these requirements using a multidimensional partitioning approach [LeFevre et al. 2006a]. The idea is to divide

[1]Variance diversity also satisfies the monotonicity property described in Machanavajjhala et al. [2006]. See the appendix for a short proof. In concurrent work, Li et al. [2007] also considered a numeric sensitive attribute and proposed a related privacy requirement called t-closeness.

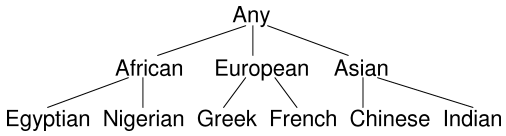

Fig. 1. Generalization hierarchy for nationality attribute.

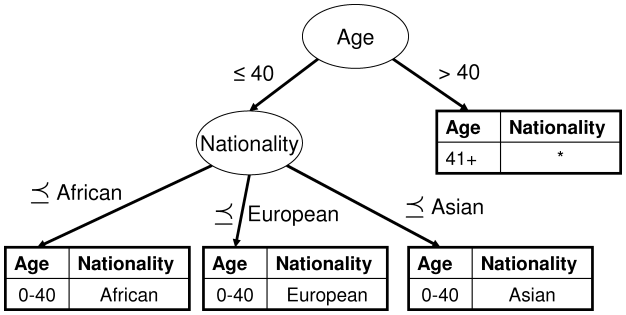

Fig. 2. Example partition tree.

the d-dimensional quasi-identifier domain space into nonoverlapping rectangular regions. This partitioning is then used to define a *global recoding function* ($\phi : D_{Q_1} \times \cdots \times D_{Q_d} \rightarrow D^d$) that maps each domain tuple to the region in which it is contained.[2] ϕ is then applied to the input relation R to produce R^*.

A partitioning is said to be *allowable* with respect to a particular input relation R if the recoded relation R^*, resulting from applying ϕ to the quasi-identifier attributes of R, satisfies all given anonymity requirements. The proposed algorithm (Mondrian) is based on greedy recursive partitioning [LeFevre et al. 2006a]. Briefly, the recursive procedure takes as input a (potentially infinite) d-dimensional rectangular domain and a set of tuples, R. The algorithm chooses a quasi-identifier split attribute (dimension of the domain space). When the split attribute is numeric, the algorithm also chooses a binary split threshold (e.g., Age $\leq$ 40; Age $>$ 40). For categorical attributes, the split is defined by specializing a user-defined generalization hierarchy (e.g., Figure 1) as originally proposed by Samarati [2001] and Sweeney [2002a]. We use the notation $\preceq$ to indicate a generalization relationship. See Figure 2.

The split attribute (and threshold) define a division of the input domain into m nonoverlapping regions that cover the input domain. The split also defines a corresponding partitioning of the input data (R) into disjoint subsets, $R_1, \ldots, R_m$. The split is said to be allowable if each R_i satisfies the given anonymity requirement(s). For example, under k-anonymity, a split is allowable if each R_i contains at least k tuples. The procedure is executed recursively on each resulting partition (R_i) until there no longer exists an allowable split.

[2]This is in contrast to other single-dimensional global recoding techniques which are defined by a set of functions $\phi_1, \ldots, \phi_d$ such that each $\phi_i : D_{Q_i} \rightarrow D'_{Q_i}$. R^* is obtained by applying each ϕ_i to the value of Q_i in each tuple of R.

Informally, a partitioning is said to be *minimal* if it satisfies the given anonymity requirement(s), and there exist no further allowable splits.

When there is no known target workload, LeFevre et al. [2006a] proposed choosing the allowable split attribute with the widest (normalized) range of values and (for numeric attributes) used the median value as the threshold. We will refer to this simple algorithm as Median Mondrian. It is appealing because, under k-anonymity, if there exists an allowable split perpendicular to a particular axis, the split at the median is necessarily allowable and can be found in linear time. However, as we will show throughout this article, when there is a known workload, we can often achieve better data quality by replacing this split heuristic.

3. WORKLOAD-AWARE ANONYMIZATION

Our goal is to allow the data recipient to specify a *family* of target workloads, and we introduce a simple language for this purpose. In particular, a workload family is specified by one or more of the following tasks.

—*Classification Tasks.* A (set of) classification task(s) is characterized by a set of predictor attributes $\{F_1, \ldots, F_n\}$ (also commonly called *features*), and one or more nominal class labels C.

—*Regression Tasks.* A (set of) regression task(s) is characterized by a set of features $\{F_1, \ldots, F_n\}$ and one or more numeric target attributes T.

—*Selection Tasks.* A set of selection tasks is defined by a set of selection predicates $\{P_1, \ldots, P_n\}$, each of which is a boolean function of the quasi-identifier attributes.

—*Aggregation Tasks.* Each aggregation task is defined by an aggregate function (e.g., SUM, MIN, AVG, etc.).

In the remainder of this section, we describe techniques for incorporating each of these components into the anonymization process.

3.1 Classification and Regression

We begin by considering workloads consisting of one or more classification or regression tasks. To lend intuition to the problem, consider a single classification or regression task and an anonymity requirement. Notice that each attribute has two characterizations: one for anonymity (QID, sensitive, or other), and one for the task (feature, target, or other).

For simplicity, suppose $\{Q_1, \ldots Q_d\} = \{F_1, \ldots, F_n\}$ (features and quasi-identifiers are the same), and consider a discrete class label C. Intuitively, our goal in this case is to produce a multidimensional partitioning of the quasi-identifier domain space (also the feature space in this case) into regions containing disjoint tuple multisets $R_1, \ldots, R_m$. Each of these partitions should satisfy the given anonymity requirements. At the same time, because of the classification task, we would like these partitions to be homogeneous with respect to C. One way to implement this intuition is to minimize the conditional

entropy of C given membership in a particular partition:

$$H(C|R^*) = \sum_{i=1}^{m} \frac{|R_i|}{|R|} \sum_{c \in D_C} -p(c|R_i) \log p(c|R_i). \quad (1)$$

This intuition is easily extended to a regression task with numeric target T. In this case, we seek to minimize the weighted mean squared error which measures the impurity of T within each partition (weighted by partition size). In the following, $\overline{T}(R_i)$ denotes the mean value of T in data partition R_i.

$$WMSE(T, R^*) = \frac{1}{|R|} \sum_{i=1}^{m} \sum_{t \in R_i} (t.T - \overline{T}(R_i))^2. \quad (2)$$

The simple case, where the features and quasi-identifiers are the same, arises in two scenarios: It is likely to occur when the anonymity requirement is k-anonymity (i.e., no sensitive attribute), and it may also occur when the target attribute (C or T) and sensitive attribute (S) are the same.[3] Intuitively, the latter case appears problematic. Indeed, entropy ℓ-diversity requires that $H(C|R^*) \geq log(\ell)$, and variance diversity requires that $WMSE(T, R^*) \geq v$. That is, the anonymity requirement places a bound on quality.

In the remainder of this section, we first describe algorithms for incorporating a single classification/regression model under the assumptions described (Section 3.1.1), and we extend the algorithm to incorporate multiple models (Section 3.1.2). Finally, in Section 3.1.3, we relax the initial assumption and describe a case where we can achieve high privacy and utility even when the sensitive attribute and target attribute are the same.

3.1.1 *Single Target Model.* Consider the case where $\{F_1, \ldots, F_n\} = \{Q_1, \ldots, Q_d\}$, and consider a single target classifier with class label C. In order to obtain heterogenous class label partitions, we propose a greedy splitting algorithm based on entropy minimization, which is reminiscent of algorithms for decision tree construction [Breiman et al. 1984; Quinlan 1993]. At each recursive step, we choose the candidate split that minimizes the following function without violating the anonymity requirement(s). Let V denote the current (recursive) tuple set, and let $V_1, \ldots, V_m$ denote the set of data partitions resulting from the candidate split. $p(c|V_i)$ is the fraction of tuples in V_i with class label $C = c$. We refer to this algorithm as InfoGain Mondrian.

$$Entropy(V, C) = \sum_{i=1}^{m} \frac{|V_i|}{|V|} \sum_{c \in D_C} -p(c|V_i) \log p(c|V_i). \quad (3)$$

When we have a continuous target attribute T, the recursive split criterion is similar to the CART algorithm for regression trees [Breiman et al. 1984], choosing the split that minimizes the following expression (without violating

[3]Our approach is to release all attributes that are not part of the quasi-identifier (that are involved in one of the tasks) without modification. In some cases, this includes the sensitive attribute. Thus, we must apply generalization to an attribute only if it is part of the quasi-identifier and either a feature or the target.

the anonymity requirement). We call this Regression Mondrian.

$$Error^2(V, T) = \sum_{i=1}^{m} \sum_{t \in V_i} (t.T - \overline{T}(V_i))^2. \quad (4)$$

Each of the two algorithms handles continuous quasi-identifier values by partitioning around the *threshold* value that minimizes the given expression without violating the anonymity requirement. In order to select this threshold, we must first sort the data with respect to the split attribute. Thus, the complexity of both algorithms is $O(|R| \log^2 |R|)$.

3.1.2 *Multiple Target Models.* In certain cases, we would like to allow the data recipient to build several models to accurately predict the marginal distributions of several class labels $(C_1, \ldots, C_n)$ or numeric target attributes $(T_1, \ldots, T_n)$. The heuristics described in the previous section can be extended to these cases. (For simplicity, assume $\{F_1, \ldots, F_n\} = \{Q_1, \ldots, Q_d\}$.)

For classification, there are two ways to make this extension. In the first approach, the data recipient would build a single model to predict the vector of class labels, $\langle C_1, \ldots, C_n \rangle$, which has domain $D_{C_1} \times \cdots \times D_{C_n}$. A greedy split criterion would minimize entropy with respect to this single variable.

However, in this simple approach, the size of the domain grows exponentially with the number of target attributes. To avoid potential sparsity problems, we instead assume independence among target attributes. This is reasonable because we are ultimately only concerned about the marginal distribution of each target attribute. Under the independence assumption, the greedy criterion chooses the split that minimizes the following without violating the anonymity requirement(s):

$$\sum_{i=1}^{n} Entropy(V, C_i). \quad (5)$$

In regression (the squared error split criterion in particular), there is no analogous distinction between treating the set of target attributes as a single variable and assuming independence. For example, if we have two target attributes, T_1 and T_2, the joint error is the distance between an observed point (t_1, t_2) and the centroid $(\overline{T_1}(V), \overline{T_2}(V))$ in 2-dimensional space. The squared joint error is the sum of individual squared errors, $(t_1 - \overline{T_1}(V))^2 + (t_2 - \overline{T_2}(V))^2$. For this reason, we choose the split that minimizes the following without violating the anonymity requirement(s):

$$\sum_{i=1}^{n} Error^2(V, T_i). \quad (6)$$

3.1.3 *Relaxing the Assumptions.* Until now, we have assumed that $\{F_1, \ldots, F_n\} = \{Q_1, \ldots, Q_d\}$. In this case, if we have a sensitive attribute S such that $S = C$ or $S = T$, then the anonymity requirement may be at odds with our notion of data quality.

However, consider the case where $\{Q_1, \ldots, Q_d\} \subset \{F_1, \ldots, F_n\}$. In this case, we can draw a distinction between partitioning the quasi-identifier space

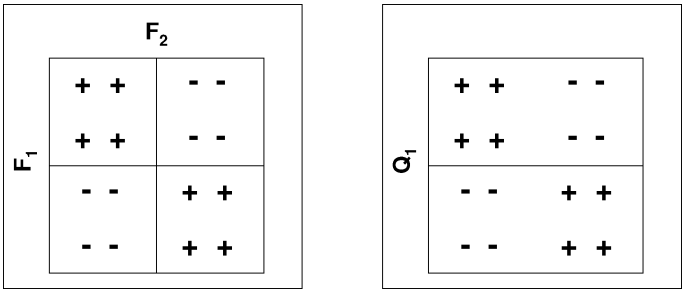

Fig. 3. Features vs. quasi-identifiers in classification-oriented anonymization.

and partitioning the feature space. For example, consider the partitioning in Figure 3 (with features F_1, F_2, and class labels/sensitive values $+$ and $-$). The feature space partitions are homogeneous with respect to the class label. However, suppose there is just one quasi-identifier attribute $Q_1 = F_1$. Clearly, the partitioning is 2-diverse with respect to Q_1.

This observation leads to an interesting extension of the greedy split heuristics. Informally, at each recursive step the extended algorithm chooses the (quasi-identifier) split that minimizes the entropy of C (or squared error of T) across the resulting feature space partitions without violating the given anonymity requirement(s) across the quasi-identifier partitions.

3.2 Selection

Sometimes one or more of the tasks in the target workload will use only a subset of the released data, and it is important that this data can be selected precisely, despite recoding. For example, in Section 1.1, we described a study that involved building a model for only males over age 40, but this is difficult if the ages of some men are generalized to the range 30–50.

Consider a set of selection predicates $\{P_1, \ldots, P_n\}$ defined by a boolean function of the quasi-identifier attributes $\{Q_1, \ldots, Q_d\}$. Conceptually, each P_i defines a query region X_i in the domain space such that $X_i = \{x : x \in D_{Q_1} \times \cdots \times D_{Q_d}, P_i(x) = true\}$. For the purposes of this work, we only consider selections for which the query region can be expressed as a d-dimensional rectangle. (Of course, some additional selections can be decomposed into two or more hyper-rectangles and incorporated as separate queries.)

A multidimensional partitioning (and recoding function ϕ) divides the domain space into nonoverlapping rectangular regions $Y_1, \ldots, Y_m$. The recoding region $Y_i = \{y : y \in D_{Q_1} \times \cdots \times D_{Q_d}, \phi(y) = y_i^*\}$, where y_i^* is a unique generalization of the quasi-identifier vector. When evaluating P_i over the sanitized view R^*, it may be that no set of recoding regions can be combined to precisely equal query region X_i. Instead, we need to define the semantics of selection queries on this type of imprecise data. Clearly, there are many possible semantics but, in the rest of this chapter, we settle on one. Under this semantics, a selection with predicate P_i returns all tuples from R^* that are contained in any

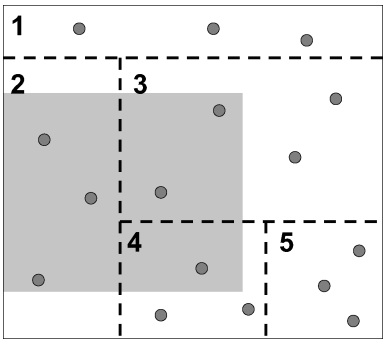

Fig. 4. Evaluating a selection over generalized data.

recoding region overlapping the corresponding query region X_i. More formally,

$$\begin{aligned} \mathit{Overlap}(X_i, \{Y_1, \ldots, Y_m\}) &= \cup\{Y_j : Y_j \in \{Y_1, \ldots, Y_m\}, Y_j \cap X_i \neq \emptyset\} \\ P_i(R^*) &= \{\phi(t) : \phi(t) \in R^* \wedge t \in \mathit{Overlap}(X_i, \{Y_1, \ldots, Y_m\})\}. \end{aligned}$$

Notice that this will often produce a larger result set than evaluating P_i over the original table R. We define the *imprecision* to be the difference in size between these two result sets.

$$\begin{aligned} P_i(R) &= \{t : t \in R, P_i(t) = \mathit{true}\} \\ \mathit{imprecision}(P_i, R^*, R) &= |P_i(R^*)| - |P_i(R)|. \end{aligned}$$

For example, Figure 4 shows a 2-dimensional domain space. The shaded area represents a query region, and the tuples of R are represented by points. The recoding regions are bounded by dotted lines and numbered. Recoding regions 2, 3, and 4 overlap the query region. If we evaluated this query using the original data, the result set would include 6 tuples. However, evaluating the query using the recoded data (under the given semantics) yields 10 tuples, an imprecision of 4.

Ideally, the goal of selection-oriented anonymization is to find the safe (k-anonymous, ℓ-diverse, variance-diverse, etc.) multidimensional partitioning that minimizes the (weighted) sum of imprecision for the set of target predicates. (We assign each predicate P_i a positive weight w_i.)

We incorporate this goal through another greedy splitting heuristic. Let V denote the current (recursive) tuple set, and let $V_1, \ldots, V_m$ denote the set of partitions resulting from the candidate split. Our heuristic minimizes the sum of weighted imprecisions:

$$\sum_{i=1}^{n} w_i * \mathit{imprecision}(P_i, V^*, V). \tag{7}$$

The algorithm proceeds until there is no allowable split that reduces the imprecision of the recursive partition. We will call this algorithm Selection Mondrian. In practice, we expect this technique to be used most often for simple selections, such as breaking down health data by state. Following this,

we continue to divide each resulting partition using the appropriate splitting heuristic (i.e., InfoGain Mondrian, etc.).

3.3 Aggregation and Summary Statistics

In multidimensional global recoding, individual data points are mapped to one multidimensional region in the set of disjoint rectangular regions covering the domain space. To this point, we have primarily considered representing each such region as a relational tuple based on its conceptual bounding box (e.g., see Figure 2).

However, when we consider the task of answering a set of aggregate queries, it is also beneficial to consider alternate ways of representing these regions using various summary statistics, which is reminiscent of ideas used in microaggregation [Domingo-Ferrer and Mateo-Sanz 2002].[4] In particular, we consider two types of summary statistics which are computed based on the data contained within each region (partition). For each attribute A in partition R_i, consider the following.

—*Range Statistic (R).* Including a summary statistic defined by the minimum and maximum value of A appearing in R_i allows for easy computation of MIN and MAX aggregates.

—*Mean Statistic (M).* We also consider a summary statistic defined by the mean value of A appearing in R_i, which allows for the computation of AVG and SUM.

When choosing summary statistics, it is important to consider potential avenues for inference. Notice that releasing minimum and maximum statistics allows for some inference about the distribution of values within a partition. For example, consider an integer-valued attribute A, and let $k = 2$. Suppose that an equivalence class contains two tuples with minimum $= 0$ and maximum $= 1$. It is easy to infer that one of the original tuples has $A = 0$ and, in the other, has $A = 1$. However, this type of inference is not problematic in preventing joining attacks because it is still impossible for an adversary to distinguish the tuples within a partition from one another.

4. EXPERIMENTAL EVALUATION OF DATA QUALITY

In this section, we describe an experimental evaluation of data quality; evaluation of runtime performance is postponed until Section 6.

Our experimental quality evaluation had two main goals. The first goal was to provide insight into experimental quality evaluation methodology. We outline an experimental protocol for evaluating an anonymization algorithm with respect to a workload of classification and regression tasks. A comparison with the results of simpler general-purpose quality measures indicates the importance of evaluating data quality with respect to the target workload when it is known.

[4]Certain types of aggregate functions (e.g., MEDIAN) are ill-suited to these types of computations. We do not know of any way to compute such functions from this type of summary statistics.

The second goal is to evaluate the extensions to Mondrian for incorporating workload. We pay particular attention to the impact of incorporating one or more target classification/regression models and the effects of multidimensional recoding. We also evaluate the effectiveness of our algorithms with respect to selections and projections.

4.1 Methodology

Given a target classification or regression task, the most direct way of evaluating the quality of an anonymization is by training each target model using the anonymized data and evaluating the resulting models using *predictive accuracy* (classification), *mean absolute error* (regression), or similar measures. We will call this methodology *model evaluation*. All of our model evaluation experiments follow a common protocol. In particular, we consider it important to hold out the test set during both anonymization and training.

(1) The data is first divided into training and testing sets (or 10-fold cross-validation sets), R_{train} and R_{test}.
(2) The anonymization algorithm determines recoding function ϕ using only the training set R_{train}. Anonymous view R^*_{train} is obtained by applying ϕ to R_{train}.
(3) The same recoding function ϕ is then applied to the testing set (R_{test}), yielding R^*_{test}.
(4) The classification or regression model is trained using R^*_{train} and tested using R^*_{test}.

Unless otherwise noted, we used k-anonymity as the anonymity requirement. We fixed the set of quasi-identifier attributes and features to be the same, and we used the implementations of the following learning algorithms provided by the Weka software package [Witten and Frank 2005]:

—*Decision Tree* (*J48*). Default settings were used.
—*Naive Bayes.* Supervised discretization was used for continuous attributes; otherwise all default settings were used.
—*Random Forests.* Each classifier was comprised of 40 random trees, and all other default settings were used.
—*Support Vector Machine* (*SMO*). Default settings, including a linear kernel function.
—*Linear Regression.* Default settings were used.
—*Regression Tree* (*M5*). Default settings were used.

In addition to model evaluation, we also measured certain characteristics of the anonymized training data to see if there was any correlation between these simpler measures and the results of the model evaluation. Specifically, we measured the average equivalence class size [LeFevre et al. 2006a], and for classification tasks, we measured the conditional entropy of the class label C, given the partitioning of the full input data R into $R_1, \ldots, R_m$ (see Equation (1)).

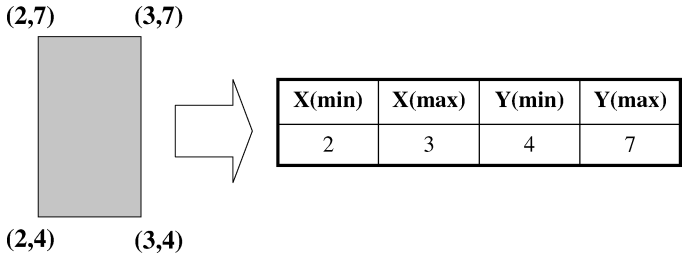

Fig. 5. Mapping a d-dimensional rectangular region to $2 * d$ attributes.

4.2 Learning from Regions

When single-dimensional recoding is used, standard learning algorithms can be applied directly to the resulting point data, notwithstanding the coarseness of some points [Fung et al. 2005]. Although multidimensional recoding techniques are more flexible, using the resulting hyper-rectangular data to train standard data mining models poses an additional challenge.

To address this problem, we make a simple observation. Because we restrict the recoding regions to include only d-dimensional hyper-rectangles, each region can be uniquely represented as a point in $(2 * d)$-dimensional space. For example, Figure 5 shows a 2-dimensional rectangle, and its unique representation as a 4-tuple. This assumes a total order on the values of each attribute similar to the assumption made by support vector machines.

Following this observation, we adopt a simple preprocessing technique for learning from regions. Specifically, we extend the recoding function ϕ to map data points to d-dimensional regions, and, in turn, to map these regions to their unique representations as points in $(2 * d)$-dimensional space.

Our primary goal in developing this technique is to establish the utility of our anonymization algorithms. There are many possible approaches to the general problem of learning from regions. For example, Zhang and Honavar [2003] proposed an algorithm for learning decision trees from attribute values at various levels of a taxonomy tree. Alternatively, we could consider assigning a density to each multidimensional region, and then sampling point data according to this distribution. However, a full comparison is beyond the scope of this work.

4.3 Experimental Data

Our experiments used both synthetic and real-world data. The synthetic data was produced using an implementation of the generator described by Agrawal et al. [1993] for testing classification algorithms. This generator is based on a set of predictor attributes, and class labels are generated as functions of the predictor attributes (see Tables I and II).

In addition to the synthetic data, we also used two real-world datasets. The first (Table III) was derived from a sample of the 2003 Public Use Microdata, distributed by the United States Census American Community Survey[5] with target attribute Salary. This data was used for both classification and regression

[5]http://www.census.gov/acs/www/index.html.

Table I. Synthetic Features/Quasi-Identifier Attributes

Attribute	Distribution
salary	Uniform in [20,000, 150,000]
commission	If salary $\geq$ 75,000, then 0 Else Uniform in [10,000, 75,000]
age	Uniform integer in [20,80]
elevel	Uniform integer in [0, 4]
car	Uniform integer in [1, 20]
zipcode	Uniform integer in [0, 9]
hvalue	zipcode * h * 100,000 where h uniform in [0.5, 1.5]
hyears	Uniform integer in [1, 30]
loan	Uniform in [0, 500,000]

Table II. Synthetic Class Label Functions

Function	Class A
C2	$((age < 40) \wedge (50K \leq salary \leq 100K)) \vee$ $((40 \leq age < 60) \wedge (75K \leq salary \leq 125K)) \vee$ $((age \geq 60) \wedge (25K \leq salary \leq 75K))$
C4	$((age < 40) \wedge$ $(((elevel \in \{0, 1\})?(25K \leq salary \leq 75K)) : (50K \leq salary \leq 100K)))) \vee$ $((40 \leq age < 60) \wedge$ $(((elevel \in \{1, 2, 3\})?(50K \leq salary \leq 100K)) : (75K \leq salary \leq 125K)))) \vee$ $((age \geq 60) \wedge$ $(((elevel \in \{2, 3, 4\})?(50K \leq salary \leq 100K)) : (25K \leq salary \leq 75K))))$
C5	$((age < 40) \wedge$ $(((50K \leq salary \leq 100K)?(100K \leq loan \leq 300K) : (200K \leq loan \leq 400K)))) \vee$ $((40 \leq age < 60) \wedge$ $(((75K \leq salary \leq 125K)?(200K \leq loan \leq 400K) : (300K \leq loan \leq 500K)))) \vee$ $((age \geq 60) \wedge$ $(((25K \leq salary \leq 75K)?(300K \leq loan \leq 500K) : (100K \leq loan \leq 300L))))$
C6	$((age < 40) \wedge (50K \leq (salary + commission) \leq 100K)) \vee$ $((40 \leq age < 60) \wedge (75K \leq (salary + commission) \leq 125K)) \vee$ $((age \geq 60) \wedge (25K \leq (salary + commission) \leq 75K))$
C7	$disposable = .67 \times (salary + commission) - .2 \times loan - 20K$ $disposable > 0$
C9	$disposable = (.67 \times (salary + commission) - 5000 \times elevel - .2 \times loan - 10K)$ $disposable > 0$

and contained 49,657 records. For classification, we replaced the numeric Salary with a Salary class ($<$ 30K or $\geq$ 30K); approximately 56% of the data records had Salary $<$ 30K. For classification, this is similar to the Adult database [Blake and Merz 1998]. However, we chose to compile this new dataset that can be used for both classification and regression.

The second real dataset is the smaller Contraceptives database from the UCI Repository (Table IV), which contained 1,473 records after removing those with missing values. This data includes nine socio-economic indicators, which are used to predict the choice of contraceptive method (long-term, short-term, or none) among sampled Indonesian women.

Table III. Census Data Description

Attribute	Distinct Vals	Generalization
Region	57	hierarchy
Age	77	continuous
Citizenship	5	hierarchy
Marital Status	5	hierarchy
Education (years)	17	continuous
Sex	2	hierarchy
Hours per week	93	continuous
Disability	2	hierarchy
Race	9	hierarchy
Salary	2/continuous	target

Table IV. Contraceptives Data Description

Attribute	Distinct Vals	Generalization
Wife's age	34	continuous
Wife's education	4	hierarchy
Husband's education	4	hierarchy
Children	15	continuous
Wife's religion	2	hierarchy
Wife working	2	hierarchy
Husband's Occupation	4	hierarchy
Std. of Living	4	continuous
Media Exposure	2	hierarchy
Contraceptive	3	target

4.4 Comparison with Previous Algorithms

InfoGain Mondrian and Regression Mondrian use both multidimensional recoding and classification- and regression-oriented splitting heuristics. In this section, we evaluate the effects of these two components through a comparison with two previous anonymization algorithms. All of the experiments in this section consider a single target model constructed over the entire anonymized training set.

Several previous algorithms have incorporated a single-target classification model while choosing a single-dimensional recoding [Fung et al. 2005; Iyengar 2002; Wang et al. 2004]. To understand the impact of multidimensional recoding, we compared InfoGain Mondrian and the greedy Top-Down Specialization (TDS) algorithm [Fung et al. 2005]. Also, we compare InfoGain and Regression Mondrian to Median Mondrian to measure the effects of incorporating a single-target model.

The first set of experiments used the synthetic classification data. Notice that the basic labeling functions in Table II include a number of constants (e.g., 75K). In order to get a more robust understanding of the behavior of the various anonymization algorithms, for functions 2, 4, and 6, we instead generated many independent datasets, varying the function constants independently at random over the range of the attribute. Additionally, we imposed hierarchical generalization constraints on attributes elevel and car.

Figure 6 compares the predictive accuracy of classifiers trained on data produced by the different anonymization algorithms. In these experiments, we

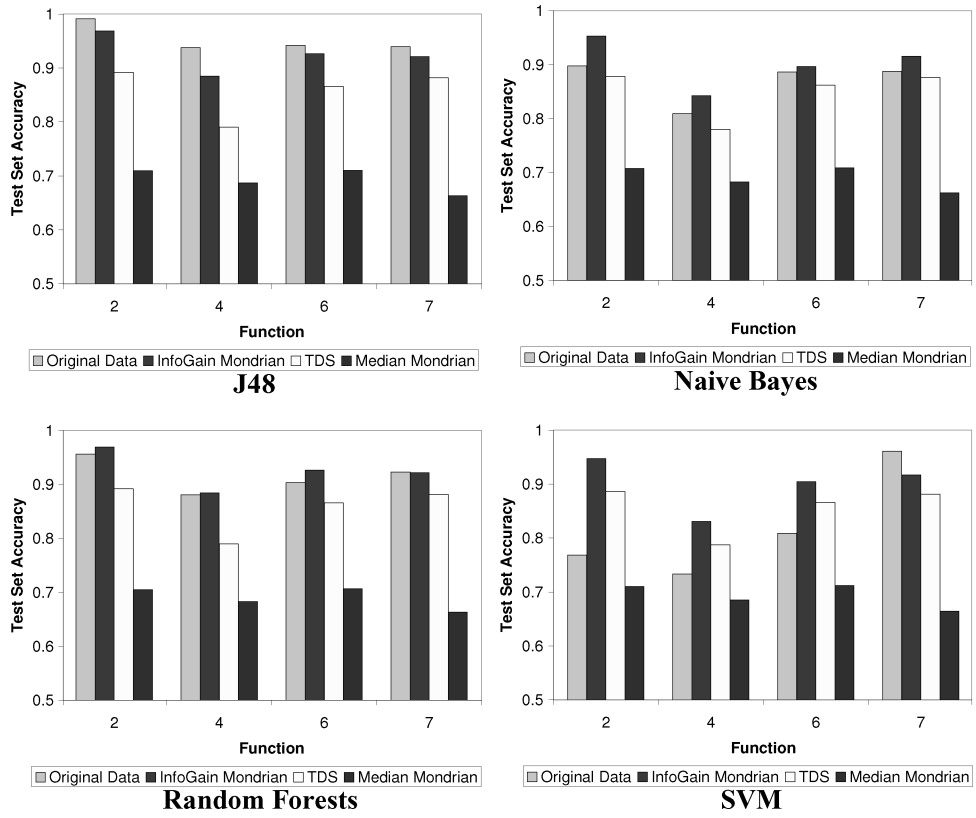

Fig. 6. Classification-based model evaluation using synthetic data ($k = 25$).

generated 100 independent training and testing sets, each containing 1,000 records, and we fixed $k = 25$. The results are averaged across these 100 trials. For comparison, we also include the accuracies of classifiers trained on the (not anonymized) original data.

InfoGain Mondrian consistently outperforms both TDS and Median Mondrian, a result that is overwhelmingly significant based on a series of paired t-tests.[6] It is important to note that the preprocessing step used to convert regions to points (Section 4.2) is only used for the multidimensional recodings; the classification algorithms run unmodified on the single-dimensional recodings produced by TDS [Fung et al. 2005]. Thus, should a better technique be developed for learning from regions, this would improve the results for InfoGain Mondrian, but it would not affect TDS.[7]

We performed a similar set of experiments using the real-world data. Figure 7 shows results for the Census classification data for increasing k. The graphs

[6]For example, when comparing InfoGain Mondrian and TDS on the J48 classifier, the one-tailed p-value is $< .001$ for each synthetic function.

[7]Note that by mapping to $2 * d$ dimensions, we effectively expand the hypothesis space considered by the linear SVM. Thus, it is not surprising that this improves accuracy for the nonlinear class label functions.

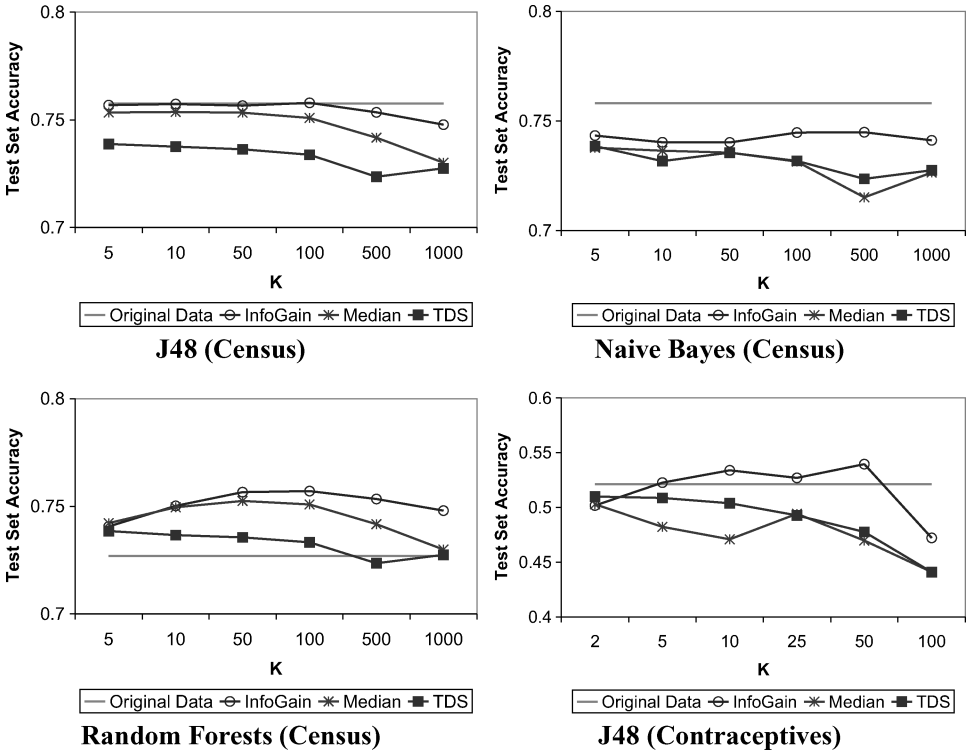

Fig. 7. Classification-based model evaluation using real-world data.

show test set accuracy (averaged across 10 folds) for three learning algorithms. The variance across the folds was quite low, and the differences between InfoGain Mondrian and TDS, and between InfoGain Mondrian and Median Mondrian, were highly significant based on paired t-tests.[8]

It is important to point out that in certain cases, notably Random Forests, the learning algorithm overfits the model when trained using the original data. For example, the model for the original data gets 97% accuracy on the training set but only 73% accuracy on the test set. When overfitting occurs, it is not surprising that the models trained on anonymized data obtain higher accuracy because anonymization serves as a form of feature selection/construction. Interestingly, we also tried applying a traditional form of feature selection (ranked feature selection based on information gain) to the original data, and this did not improve the accuracy of Random Forests for any number of chosen attributes. We suspect that this discrepancy is due to the flexibility of the recoding techniques. Single-dimensional recoding (TDS) is more flexible than traditional feature selection because it can incorporate attributes at varying levels of granularity. Multidimensional recoding is more flexible still because it

[8]On the Census data, for J48 and Naive Bayes, when comparing InfoGain Mondrian vs. Median Mondrian, and InfoGain Mondrian vs. TDS, the one-tailed p-value is always $< .02$ for $k = 5$. For Random Forests, the p-value is $< .02$ for $k = 10$.

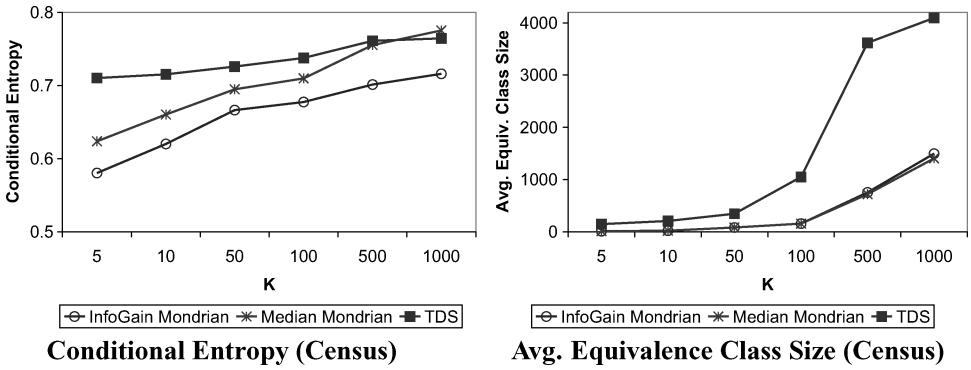

Fig. 8. General-purpose quality measures using real-world data.

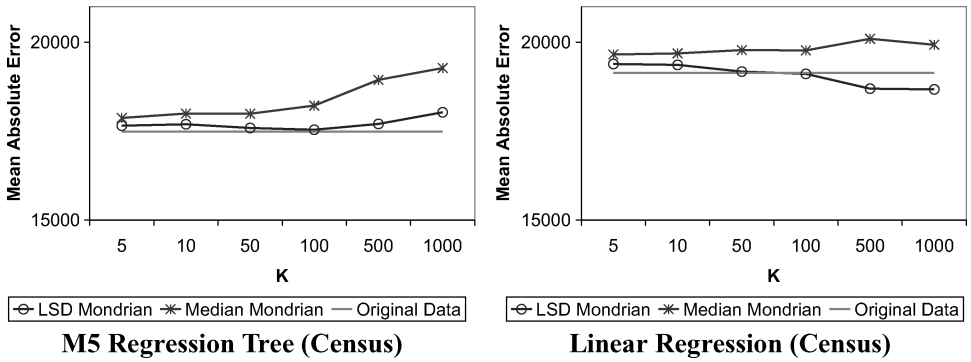

Fig. 9. Regression-based model evaluation using real-world data.

incorporates different attributes (at different levels of granularity) for different data subsets.

We performed the same set of experiments using the Contraceptives database and observed similar behavior. InfoGain Mondrian yielded higher accuracy than TDS or Median Mondrian. Results for J48 are shown in Figure 7.[9]

Next Figure 8 shows conditional entropy and average equivalence class-size measurements averaged across the ten anonymized training folds of the Census classification data. Notice that Median Mondrian and InfoGain Mondrian consistently produce equivalence classes of comparable size despite the difference in predictive accuracy; this indicates that average equivalence class size is not a very good indicator of data quality with respect to this particular workload. Conditional entropy, which incorporates the target class label, is better; low-conditional entropy generally indicates a higher-accuracy classification model.

For regression, we found that Regression Mondrian generally led to better models than Median Mondrian. Figure 9(a) shows the mean absolute test set error for the M5 regression tree and a linear regression using the Census regression data.

[9]On the contraceptives data, when comparing InfoGain Mondrian vs. Median Mondrian and InfoGain Mondrian vs. TDS, the one-tailed p-value is $< .05$ for $k = 10$.

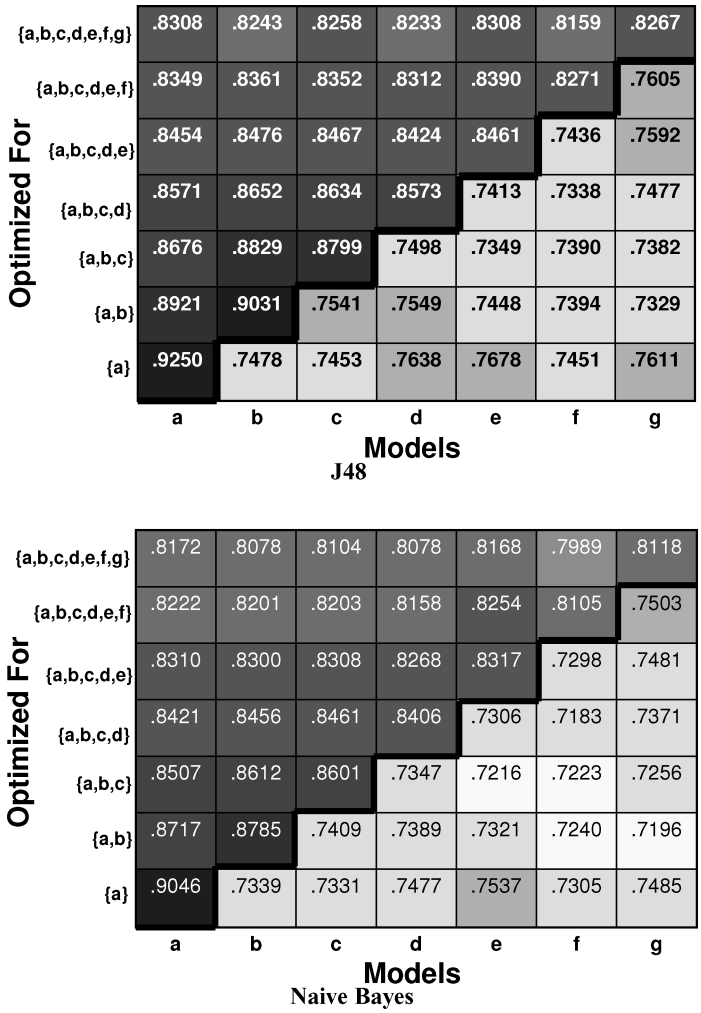

Fig. 10. Classification-based model evaluation for multiple models ($k = 25$).

4.5 Multiple Target Models

In Section 3.1.2, we described a simple adaptation to the basic InfoGain Mondrian algorithm that allowed us to incorporate more than one target attribute, expanding the set of models for which a particular anonymization is optimized. To evaluate this technique, we performed a set of experiments using the synthetic classification data, increasing the number of class labels.

Figure 10 shows average test set accuracies for J48 and Naive Bayes. We first generated 100 independent training and testing sets, containing 1,000 records each. We used synthetic labeling functions 2-6,7, and 9 from the Agrawal generator [Agrawal et al. 1993], randomly varying the constants in functions 2-6 as described in Section 4.4.

Each column in the figure (models A-G) represents the average of 25 random permutations of the synthetic functions. (i.e., to get a more robust result, we randomly re-ordered the class labels.) The anonymizations (rows in the figure) are optimized for an increasing number of target models. (e.g., the anonymization in the bottom row is optimized exclusively for model A.) There are two important things to note from the chart, and similar behavior was observed for the other classification algorithms.

—Looking at each model (column) individually, when the model is included in the anonymization (above the bold line), test set accuracy is higher than when the model is not included (below the line).

—As we increase the number of included models (moving upward above the line within each column), the test set accuracy tends to decrease. This is because the quality of the anonymization with respect to each individual model is diluted by incorporating additional models.

4.6 Privacy-Utility Trade off

In Section 3.1, we noted that there are certain cases where the trade off between privacy and utility is (more or less) explicit, provided that conditional entropy is a good indicator of classification accuracy. In particular, this occurs when the set of features is the same as the set of quasi-identifiers, and the sensitive attribute is the same as the class label or numeric target attribute.

In this section, we illustrate this empirically. Specifically, we conducted an experiment using entropy ℓ-diversity as the only anonymity requirement (i.e., $k = 1$) for increasing values of parameter ℓ. We again used the Census classification data, and this time let the salary class attribute be both the sensitive attribute and the class label. For each ℓ value, we conducted an anonymization experiment, measuring the average conditional entropy of the resulting data (across the 10 folds) as well as the average test set classification accuracy.

The results are shown in Figure 11. As expected, the conditional entropy (across resulting partitions) increases for increasing ℓ.[10] Also, it is not surprising that the classification accuracy slowly deteriorates with increasing ℓ.

4.7 Selection

In Section 3.2, we discussed the importance of preserving selections and described an algorithm for incorporating rectangular selection predicates into an anonymization. We conducted an experiment using the synthetic data (1,000 generated records), but treating synthetic Function C2 as a selection predicate. Figure 12 shows the imprecision of this selection when evaluated using the recoded data. The figure shows results for data recoded using three different anonymization algorithms. The first algorithm is Median Mondrian with greedy recursive splits chosen from among all of the quasi-identifier attributes. It also shows a restricted variation of Median Mondrian where splits

[10]When $\ell = 1$, the conditional entropy is greater than 0 due to a small number of records in the original data with identical feature vectors but differing class labels.

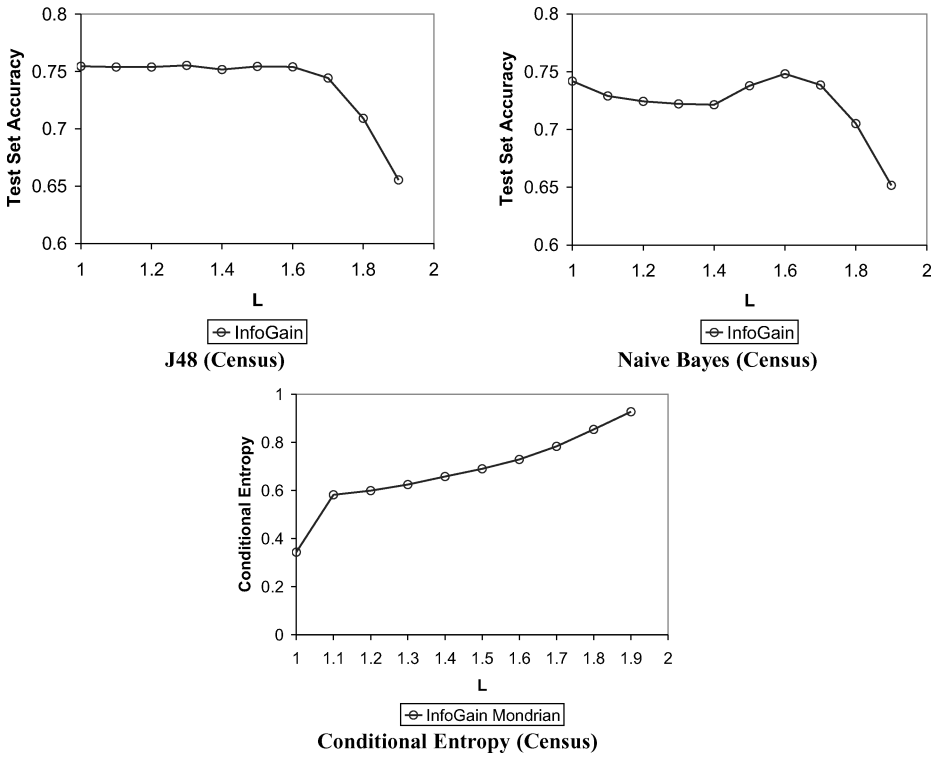

Fig. 11. ℓ-Diversity experiment.

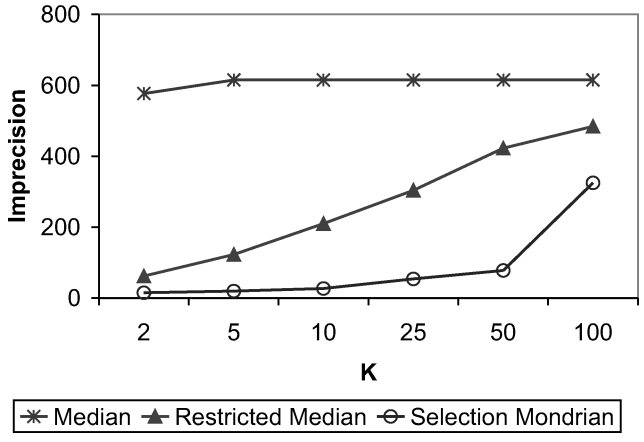

Fig. 12. Imprecision for synthetic Function C2.

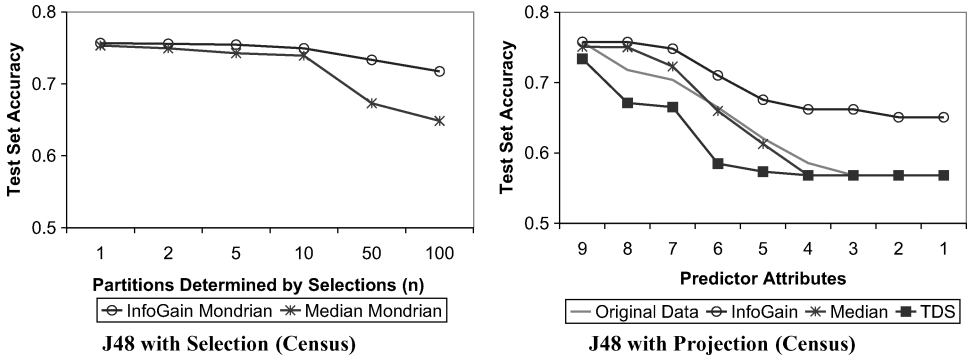

Fig. 13. Selection and projection experiment.

are made with respect to only Age and Salary. Finally, it shows the results of Selection Mondrian, incorporating Function C2 as three separate rectangular query regions (each with equal weight). It is intuitive that imprecision increases with k and that imprecision is reduced by incorporating the selection into the anonymization.

Incorporating selections can also affect model quality. In the absence of selections, InfoGain and Regression Mondrian choose recursive splits using a greedy criterion driven by the target model(s). When selections are included, the resulting partitions may not be the same as those that would be chosen based on the target model(s). In the worst case, there may be a selection on an attribute that is uncorrelated with the target attribute.

To demonstrate this intuition, we performed an experiment using the Census classification data. To simulate the effect of selections that are uncorrelated with the target model, we first assigned each training tuple to one of n groups, chosen uniformly at random. (We assume $\frac{|R|}{n} \geq k$.) We then anonymized each group independently, using either InfoGain Mondrian or Median Mondrian. Once recodings were determined for each training group, we randomly assigned each test tuple to one of the n groups and recoded the tuple using the recoding function for that group. Finally, we trained a single classification model using the full recoded training set (union of all training groups) and tested using the full recoded test set. This process was repeated for each of ten folds.

The results of this selection experiment for J48 are shown in Figure 13 for increasing n and $k = 50$. As expected, accuracy decreases slightly as the number of selections (n) increases. However, several selections can be incorporated without large negative effects. Similar results were observed for the other classification algorithms.

4.8 Projection

In certain cases, the data recipient will not use all released attributes when constructing a model. Instead, he or she will build the model using only a projected subset of attributes. In our experiments, we have found that single-dimensional recoding often preserves precise values for fewer attributes than does multidimensional recoding. (This was also observed in LeFevre et al. [2006a].)

In this section, we describe an experiment comparing anonymization algorithms when only a subset of the released features is used in constructing a particular model. In this experiment, we first ranked the set of all features using the original data and a greedy information gain heuristic. We then removed the features in order, from most to least predictive, and constructed classification models using the remaining attributes. We fixed $k = 100$.

As expected, test set accuracy decreases as the most predictive features are dropped. However, the rate of this decline varies depending on the anonymization algorithm used. Figure 13 shows the observed accuracies for J48 using the Census database. Because of the single-dimensional recoding pattern which preserves fewer attributes, this rate of decay is the most precipitous for TDS. The results were similar for the other classification algorithms and the Contraceptives data.

5. INCORPORATING SCALABILITY

While numerous anonymization algorithms have recently been proposed, few have considered datasets larger than main memory. Proposed scalable techniques [Mokbel et al. 2006; Iwuchukwu and Naughton 2007] based on spatial indexing do not support workload-oriented splitting heuristics. For this reason, we introduce and evaluate two external adaptations of the Mondrian algorithmic framework (Median Mondrian, InfoGain Mondrian, and Regression Mondrian). For clarity, we refer to the scalable variations as *Rothko*.[11]

The first adaptation is based on ideas from the RainForest scalable decision tree algorithms [Gehrke et al. 1998]. Although the basic structure of the algorithm is similar to RainForest, there were several technical problems we had to address. First, in order to choose an allowable split (according to a given split criterion and anonymity requirement), we need to choose an appropriate set of count statistics since those used in RainForest are not always sufficient. Also, we note that in the anonymization problem, the resulting partition tree does not necessarily fit in memory, and we propose techniques addressing this problem.

The second adaptaion takes a different approach based on sampling. The main idea is to use a sample of the input dataset R (that fits in memory) and to build the partition tree optimistically according to the sample. Any split made in error is subsequently undone; thus, the output is guaranteed to satisfy all given anonymity requirements. We find that, for reasonably large sample sizes, this algorithm also generally results in a minimal partitioning.

5.1 Exhaustive Algorithm (Rothko-T)

Our first algorithm, which we call *Rothko-Tree* (or *Rothko-T*), leverages several ideas originally proposed as part of the RainForest scalable decision tree framework [Gehrke et al. 1998]. Like Mondrian, decision tree construction typically involves a greedy recursive partitioning of the domain (feature) space. For decision trees, Gehrke et al. [1998] observed that split attributes (and thresholds)

[11]Mark Rothko (1903-1970) was a Latvian-American painter whose late abstract expressionist work was influenced by Piet Mondrian among others.

could be chosen using a set of count statistics, typically much smaller than the full input dataset.

In many cases, allowable splits can be chosen greedily in Mondrian using related count statistics, each of which is typically much smaller than the size of the input data.

—*Median / k-Anonymity.* Under k-anonymity and Median partitioning, the split attribute (and threshold) can be chosen using what we will call an *AV group*. The *AV set* of attribute A for tuple set R is the set of unique values of A in R, each paired with an integer indicating the number of times it appears in R (i.e., SELECT A, COUNT(*) FROM R GROUP BY A). The AV group is the collection of AV sets, one per quasi-identifier attribute.

—*InfoGain / k-Anonymity.* When the split criterion is InfoGain, each AV set (group) must be additionally augmented with the class label, producing an AVC set (group) as described in Gehrke et al. [1998] (i.e., SELECT A, C, COUNT(*) FROM R GROUP BY A, C.).

—*Median / ℓ-Diversity.* In order to determine whether a candidate split is allowable under ℓ-diversity, we need to know the joint distribution of attribute values and sensitive values, for each candidate split attribute (i.e., SELECT A, S, COUNT(*) FROM R GROUP BY A, S). We call this the AVS set (group).

—*InfoGain / ℓ-Diversity.* Finally, when the split criterion is InfoGain, and the anonymity constraint is ℓ-diversity, the allowable split yielding maximum information gain can be chosen using both the AVC and AVS groups.

Throughout the rest of the article, when the anonymity requirement and split criterion are clear from context, we will interchangeably refer to them as *frequency sets* and *frequency groups*.

When the anonymity requirement is variance diversity or the split criterion is Regression, the analogous summary counts (e.g., the joint distribution of attribute A and a numeric sensitive attribute S or numeric target attribute T) are likely to be prohibitively large. We return to this issue in Section 5.2.

In the remainder of this section, we describe a scalable algorithm for k-anonymity and/or ℓ-diversity (using Median or InfoGain splitting) based on these summary counts. In each case, the output of the scalable algorithm is identical to the output of the corresponding in-memory algorithm.

5.1.1 *Algorithm Overview.* The recursive structure of Rothko-T follows that of Rain-Forest [Gehrke et al. 1998], and we assume that at least one frequency group will fit in memory. In the simplest case, the algorithm begins at the root of the partition tree and scans the input data (R) once to construct the frequency group. Using this, it chooses an allowable split attribute (and threshold) according to the given split criterion. Then, it scans R once more and writes each tuple to a disk-resident child partition as designated by the chosen split. The algorithm proceeds recursively in a depth-first manner, dividing each of the resulting partitions (R_i) according to the same procedure.

Once the algorithm descends far enough into the partition tree, it will reach a point where the data in each leaf partition is small enough to fit in memory.

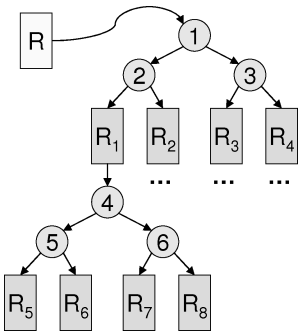

Fig. 14. Rothko-T example.

At this point, a sensible implementation loads each partition (individually) into memory and continues to apply the recursive procedure in memory.

When multiple frequency groups fit in memory, the simple algorithm can be improved to take better advantage of the available memory using an approach reminiscent of the RainForest hybrid algorithm. In this case, the algorithm first scans R, choosing the split attribute and threshold using the resulting frequency group. Now, suppose that there is enough memory available to (simultaneously) hold the frequency groups for all child partitions. Rather than repartitioning the data across the children, the algorithm proceeds in a breadth-first manner, scanning R once again to create frequency groups for all of the children.

Because the number of partitions grows exponentially as the algorithm descends in the tree, it will likely reach a level at which all frequency groups no longer fit in memory. At this point, it divides the tuples in R across the leaves, writing these partitions to disk. The algorithm then proceeds by calling the procedure recursively on each of the resulting partitions. Again, when each leaf partition fits in memory, a sensible implementation switches to the in-memory algorithm.

Example (*Rothko-T*). Consider input tuple set (R), and suppose there is enough memory available to hold 2 frequency groups for R. The initial execution of the algorithm is depicted in Figure 14.

Initially, the algorithm scans R once to create the frequency group for the root (1) and chooses the best allowable split (provided that one exists). (In this example, all of the splits are binary.) Then, the algorithm scans R once more to construct the frequency groups for the child nodes (2 and 3) and chooses the best allowable splits for these nodes.

Following this, the four frequency groups for the next level of the tree will not fit in memory so the data is divided into partitions $R_1, \ldots, R_4$. The procedure is then called recursively on each of the resulting partitions.

5.1.2 *Recoding Function Scalability.* The previous section highlights an additional problem. Because the decision trees considered by Gehrke et al. [1998] were of approximately constant size, it was reasonable to assume that the resulting tree structure itself would fit in memory. Unfortunately, this is often not true of our problem.

Instead, we implemented a simple scalable technique for materializing the multidimensional recoding function ϕ. Notice that each path from root to leaf in the partition tree defines a rule, and the set of all such rules defines global recoding function ϕ. For example, in Figure 2, $(Age < 40) \wedge (Nationality \preceq European) \rightarrow \langle [0-40], European \rangle$ is one such rule.

The set of recoding rules can be constructed in a scalable way, without fully materializing the tree. In the simplest case, when only one frequency group fits in memory, the algorithm works in a purely depth-first manner. At the end of each depth-first branch, we write the corresponding rule (the path from root to leaf) to disk. This simple technique guarantees that the amount of information stored in memory at any one time is proportional to the height of the tree, which grows only as a logarithmic function of the data.

When more memory is available for caching frequency groups, the amount of space is slightly larger due to the periods of breadth-first partitioning, but the approach still consumes much less space than materializing the entire tree.

Finally, note that the tree structure is only necessary if it is used to define a global recoding function that covers the domain space. If we instead choose to represent each resulting region using summary statistics, then the tree structure need not be materialized. Instead, the summary statistics can be computed directly from the resulting data partitions.

5.2 Sampling Algorithm (Rothko-S)

In this section, we describe a second scalable algorithm, this time based on sampling. Rothko-Sampling (or Rothko-S) addresses some of the shortcomings of Rothko-T. Specifically, because splits are chosen using only memory-resident data, it provides us with the ability to choose split attributes using the Regression split criterion and to check variance diversity. The sampling approach also often leads to better performance.

The main recursive procedure consists of three phases.

(1) (*Optimistic*) *Growth Phase*. The procedure begins by scanning input tuple set R to obtain a simple random sample (r) that fits in the available memory. (If R fits in memory, then $r = R$.) The procedure then grows the tree, using sample r to choose split attributes (thresholds). When evaluating a candidate split, it uses the sample to estimate certain characteristics of R, and using these estimates, it will make a split (optimistically) if it can determine with high confidence that the split will not violate the anonymity requirement(s) when applied to the full partition R. The specifics of these tests are described in Section 5.2.1.

(2) *Repartitioning Phase*. Eventually, there will be no more splits that can be made with high confidence based on sample r. If $r \subset R$, then input tuple set R is divided across the leaves of the tree built during the growth phase.

(3) *Pruning Phase*. When $r \subset R$, there is the possibility that certain splits were made in error during the growth phase. Given a reasonable testing procedure, this won't happen often, but when a node in the partition tree is found to violate (one of) the anonymity requirement(s), then all of the

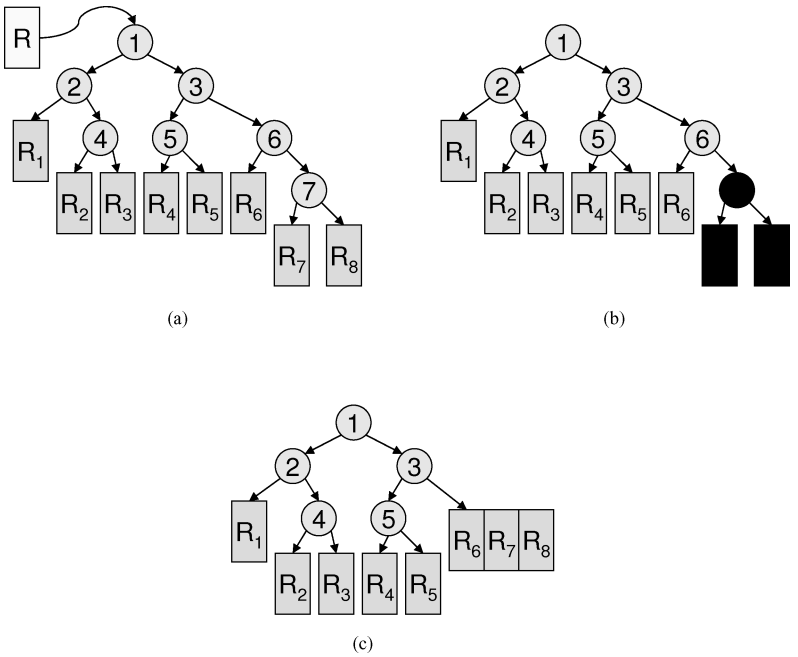

Fig. 15. Rothko-S example.

partitions in the subtree rooted at the parent of this node are merged. To do this, during the repartitioning phase, we maintain certain population statistics at each node. (For k-anonymity, this is just a single integer count. For ℓ-diversity or variance diversity, we construct a frequency histogram over the set of unique sensitive values.)

Finally, the procedure is executed recursively on each resulting partition, $R_1, \ldots, R_m$. In virtually all cases, the algorithm will eventually reach a base case where each recursive partition R_i fits entirely in memory. (There are a few pathological exceptions, which we describe in Section 5.2.2. These cases typically only arise when an extremely small amount of memory is available.)

Recoding function scalability can be implemented as described in Section 5.1.2. In certain cases, we stop the growth phase early for one of three possible reasons. First, if we are constructing a global recoding function and the tree structure has filled the available memory, we then write the appropriate recoding rules to disk. Similarly, we repartition the data if the statistics necessary for pruning (e.g., sensitive frequency histograms) no longer fit in memory. Finally, notice that repartitioning across a large number of leaves may lead to a substantial amount of nonsequential I/O if there is not enough memory available to adequately buffer writes. In order to prevent this from occurring, the algorithm may repartition the data while there still exist high-confidence allowable splits.

Example (*Rothko-S*). Consider an input tuple set R. The algorithm is depicted in Figure 15.

The growth phase begins by choosing sample r from R and growing the partition tree accordingly. When there are no more (high-confidence) allowable splits, R is repartitioned across the leaves of the tree (e.g., Figure 15(a)).

During repartitioning, the algorithm tracks necessary population statistics for each node (e.g., total count for k-anonymity). In the example, suppose that Node 7 violates the anonymity requirement (e.g., contains fewer than k tuples). In this case, the tree is pruned and partitions R_6, R_7, R_8 combined.

The procedure is then executed recursively on data partitions $R_1, \ldots, R_5, R_6 \cup R_7 \cup R_8$.

5.2.1 *Estimators and Hypothesis Tests.* Rothko-S must often use a sample to check whether a candidate recursive split satisfies the given anonymity requirement(s). A naive approach performs this check directly on the sample. For example, under k-anonymity, if input data R contains N tuples and we have selected a sample of size n, the naive approach makes a split (optimistically) if each resulting sample partition contains at least $k\left(\frac{n}{N}\right)$ tuples.

Unfortunately, we find that this naive approach can lead to an excessive amount of pruning in practice (Section 6.6). Instead, we propose to perform this check based on a statistical hypothesis test. In this section, we outline some preliminary methods for performing these tests. We find that, while our tests for variance diversity and ℓ-diversity do not make strong guarantees, these tests produce quite favorable results in practice. Most importantly, the test will never affect the anonymity of the resulting data because the algorithm always undoes any split made in error.

In the context of splitting, the *null hypothesis* (H_0) can be described informally as stating that the candidate split is *not* allowable under the given anonymity requirement. An ideal test would reject H_0 if it can determine (using realistic assumptions) that there is only a small probability ($\leq \alpha$) of the split violating the anonymity requirement. During the growth phase, Rothko-S will make a split (optimistically) if H_0 can be rejected with high confidence.

In the following, let R denote the input data (a finite population of tuples), and let N denote the size (number of tuples) of R. Let r denote a simple random sample of n tuples, drawn uniformly without replacement from R ($n \leq N$). Consider a candidate split, which divides R into m partitions $R_1, \ldots, R_m$. (When applied to sample r, the split yields sample partitions $r_1, \ldots, r_m$.)

k-Anonymity. We begin with k-anonymity. Let $p = |R_i|/N$ denote the proportion of tuples from R that would fall in partition R_i after applying a candidate split to R. Under k-anonymity, H_0 and H_1 can be expressed (for R_i) as follows, where $p_0 = k/N$.

$$H_0 : p = p_0$$
$$H_1 : p \geq p_0$$

Similarly, let $\widehat{p} = |r_i|/n$. We use proportion $\widehat{p}$ to estimate p. Regardless of the underlying data distribution, we know by the Central Limit Theorem that $\widehat{p}$ is approximately normally distributed (for large samples). Thus, we use the

following test, rejecting H_0 when the expression is satisfied.[12]

$$Var_{H_0} = \frac{p_0(1-p_0)}{n-1}\left(\frac{N-n}{N}\right)$$
$$\widehat{p} - p_0 \geq z_{\alpha/m}\sqrt{Var_{H_0}}.$$

There are three important things to note about this test. First, notice that we are simultaneously testing all m partitions resulting from the split. That is, we want to construct the test so that the total probability of accepting any R_i containing fewer than k tuples is α. For this reason we use the Bonferroni correction (α/m).

Also, it is important to remember that we are sampling from a finite population of data (R), and the fraction of the population that fits in memory (and is included in the sample) grows each time the algorithm repartitions the data. For this reason, we have defined Var_{H_0} in terms of the sampling process, incorporating a finite population correction. Given this correction, notice that when $N = n$ (i.e., the entire partition fits in memory), then $Var_{H_0} = 0$.

Finally, as the growth phase progresses (prior to repartitioning), note that the population (R), and the sample (r), do not change. The only component of the hypothesis test that changes during a particular instantiation of the growth phase is $\widehat{p}$, which decreases with each split. Thus, as the growth phase progresses, it becomes increasingly likely that we will be unable to reject H_0, at which point we repartition the data.

Recursive (c, ℓ)*-Diversity.* When the anonymity requirement is recursive (c, ℓ)-diversity, it is substantially more difficult to construct the hypothesis test with strong guarantees about α. The technique described in this section is simply a rule of thumb that accomplishes our practical goals.[13]

We must use each sample partition (r_i) to estimate certain characteristics of the sensitive attribute S within the corresponding population partition (R_i). Let N_i denote the size of population partition R_i, and let n_i denote the size of sample partition r_i.

Recursive (c, ℓ)-diversity can be expressed in terms of two proportions. Let X_j denote the frequency of the jth most common sensitive value in R_i. Let $p_1 = X_1/N_i$ and $p_2 = (X_\ell + \cdots + X_{|D_S|})/N_i$. Using these proportions,

$$H_0 : p_1 = c * p_2$$
$$H_1 : p_1 < c * p_2.$$

We use the sample partition (r_i) to estimate these proportions. Let x_j denote the frequency of the jth most common sensitive value in r_i, and let $\widehat{p_1} = x_1/n_i$ and $\widehat{p_2} = (x_\ell + \cdots + x_{|D_S|})/n_i$.

Notice that these estimates make several implicit assumptions. First, they assume that the domain of sensitive attribute S is known. More importantly,

[12] z_α is the number such that the area beneath the standard normal curve to the right of $z_\alpha = \alpha$.
[13] We also considered entropy ℓ-diversity [Machanavajjhala et al. 2006] and found it equally difficult to develop a precise test without simulating H_0 which is computationally costly.

they assume that the ordering of sensitive value frequencies is the same in R_i and r_i. (Clearly, this is not true, but in fact leads to a conservative bias.) Nonetheless, this is a good starting point.

In order to do the test, we need to estimate the sample variance of $c\widehat{p_2} - \widehat{p_1}$. If we assume that $\widehat{p_1}$ and $\widehat{p_2}$ are independent (also not true), then

$$Var(c\widehat{p_2} - \widehat{p_1}) = c^2 Var(\widehat{p_2}) + Var(\widehat{p_1}).$$

An estimator for $Var(\widehat{p})$ is

$$\frac{\widehat{p}(1-\widehat{p})}{n_i - 1}\left(\frac{N_i - n_i}{N_i}\right),$$

so we estimate the variance as follows:

$$\frac{c^2\widehat{p_2}(1-\widehat{p_2}) + \widehat{p_1}(1-\widehat{p_1})}{n_i - 1}\left(\frac{N_i - n_i}{N_i}\right).$$

Of course, when choosing a candidate split, we do not know N_i, the size of the ith resulting population partition. Instead, we use the overall sampling proportion ($\frac{n}{N}$) to guide the finite population correction, which gives us the following estimate.

$$\widehat{Var}_{H_0} = \frac{c^2\widehat{p_2}(1-\widehat{p_2}) + \widehat{p_1}(1-\widehat{p_1})}{n_i - 1}\left(\frac{N - n}{N}\right).$$

Finally, we reject H_0 in favor of H_1 when the following expression is satisfied, again using the Bonferroni correction.

$$c\widehat{p_2} - \widehat{p_1} > z_{\alpha/m}\sqrt{\widehat{Var}_{H_0}}.$$

Variance Diversity. When the anonymity requirement is variance diversity, our test is again just a rule of thumb. We again use the sample partition r_i to estimate certain characteristics of R_i, namely, the variance of sensitive attribute S. The null and alternative hypotheses (for population partition R_i) can be expressed as follows.

$$H_0 : Var(R_i, S) = v$$
$$H_1 : Var(R_i, S) \geq v.$$

We use the variance of S within sample partition r_i as an estimate of the variance in population partition R_i.

$$\widehat{Var}(R_i, S) = \frac{1}{n_i - 1}\sum_{j=1}^{n_i}(s_j - \overline{s})^2.$$

Recall that if each s_j is an independent normally-distributed random variable, then the sample variance of S follows a chi-square distribution. Under

this assumption, we reject H_0 (for R_i) if the following holds.[14]

$$\frac{(n_i - 1)\widehat{Var}(R_i, S)}{v} \geq \chi^2_{\alpha/m}\,(n_i - 1\ df).$$

In reality, S may follow an arbitrary distribution, and because we are sampling from a finite population, the elements in the sample are not independent. Because this test does not include a finite population correction as such, when the overall sampling proportion $\frac{n}{N} = 1$ (which means that the algorithm is operating on the full data partition), we instead reject H_0 when $Var(R_i, S) \geq v$, according to the definition of variance diversity.

5.2.2 *Discussion.* Partitionings produced by Rothko-S are always guaranteed to satisfy the given anonymity requirement(s) provided that the entire input database satisfies the requirement(s). In virtually all cases (i.e., when the sample size is not extremely small), the resulting partitioning is also minimal (see Section 6). Potential nonminimality can, however, occur in the following scenario. Suppose the algorithm is operating on only a sample in some recursive instantiation (i.e., R is larger than memory). If there does not exist a single (high-confidence) split that can be made during the growth phase, then it is possible that the resulting partitioning is nonminimal.[15] In this sense, the potential for nonminimality can be roughly equated with the power of the test. Similarly, if all splits made during the growth phase are undone during the pruning phase, we stop the algorithm to avoid thrashing.

There are two other important issues to consider. First, as we mentioned previously, our application can withstand some amount of imprecision and bias in the hypothesis test routine because splits that are made incorrectly based on a sample are eventually undone. However, it is important for efficiency that this does not happen too often. We continue to explore this issue in Section 6.6 as part of the experimental evaluation.

The second important issue to consider is the precision of the sampling-based algorithm with respect to workload-oriented splitting heuristics (InfoGain and Regression). It is clear that the split chosen using sample r is not guaranteed to be the same as the split that would be chosen according to the full partition R. This problem has been studied in the context of a sampling-based decision-tree construction algorithm (BOAT) [Gehrke et al. 1999] and could be similarly addressed in the anonymization setting using bootstrapping for splits and subsequent refinement.[16]

From a practical perspective, however, we find that it is less important in our problem to choose the optimal split (according to the population) at every step. While decision trees typically seek to construct a compact structure that

[14] $\chi^2_\alpha\,(ndf)$ is the number such that the area beneath the chi-square density function (with n degrees of freedom) to the right is α.

[15] Of course, in the rare event that this scenario arises in practice, it is easily detected. For k-anonymity, ℓ-diversity, Median and InfoGain splitting, a reasonable implementation would simply switch to Rothko-T for the offending partition.

[16] The techniques proposed as part of BOAT would also have to be extended to handle the case where the entire partition tree does not fit in memory.

|R|	*Number of disk blocks in input relation R*
||R||	*Number of data tuples in input relation R*
TM	*Number of data tuples that fit in memory*
F_CACHE	*Number of frequency groups that fit in memory* (≥ 1)
height	*Height of the partition tree before each leaf partition fits in memory*

Fig. 16. Notation for analytical comparison.

expresses an underlying concept, the anonymization algorithm continues partitioning the domain space until no allowable splits remain. We return to the issue of sampling and data quality in the experimental evaluation (Section 6.5).

5.3 Analytical Comparison

In order to lend insight into the experimental evaluation, this section provides a brief analytical comparison of the I/O behavior of Rothko-T and Rothko-S. For simplicity, we make this comparison for numeric data (binary splits), k-anonymity, and partition trees that are balanced and complete. Obviously, these assumptions do not hold in all cases. Under Median splitting, the partition tree will be only approximately balanced and complete due to duplicate values; for InfoGain and Regression splitting, the tree is not necessarily balanced or complete. Under ℓ-diversity and variance diversity, the analysis additionally depends on the distribution of sensitive attribute S. Nonetheless, the analytical comparison provides valuable intuition for the relative performance of the two scalable algorithms.

We use the notation described in Figure 16, and we count the number of disk blocks that are read and written during the execution of each algorithm.

5.3.1 *Rothko-T.* We begin with Rothko-T. Recall that once each leaf contains $\leq TM$ tuples, we switch to the in-memory algorithm. The height of the partition tree, prior to this switch, is easily computed. (We assume that $k \ll TM$.)

$$height = max\left(0, \left\lceil \log_2\left(\frac{\|R\|}{TM}\right)\right\rceil\right).$$

Regardless of the available memory, the algorithm must scan the full dataset $height + 1$ times. (The final scan imports the data in each leaf before executing the in-memory algorithm.) As F_CACHE increases, an increasing number of repartitions are eliminated.[17] Thus, the total number of reads and writes (disk blocks) is as follows:

$$\begin{aligned} repartitions_T &= \left\lceil \frac{height}{\lfloor \log_2(F\_CACHE) \rfloor + 1} \right\rceil \\ reads_T &= |R| * (height + repartitions_T + 1) \\ writes_T &= |R| * repartitions_T. \end{aligned}$$

[17]For simplicity, we assume that the size of a frequency group is approximately constant for a given dataset. In reality, the number of unique values per partition decreases as we descend in the tree.

It is important to note that, unlike scalable decision trees [Gehrke et al. 1998], Rothko-T does not scale linearly with the size of the data. The reason for this is simple: decision trees typically express a concept of fixed size, independent of the size of the training data. In the anonymization algorithm, however, the height of the partition tree grows as a function of the input data and parameter k. For Median partitioning, the height of the full partition tree is approximately $\lfloor \log_2(\frac{\|R\|}{k}) \rfloor$.

5.3.2 *Rothko-S.* In the case of Rothko-S, the number of repartitions is a function of the estimator (rather than *F_CACHE*). The following recursive function counts the number of times the full dataset is repartitioned under k-anonymity.

repartitions$_S(N)$
if $(N \leq TM)$
return 0
else
$p_0 = k/N$
$n = min(TM, N)$
$levels = max\left(x \in \mathbb{Z} : \frac{1}{2^x} - p_0 \geq z_{\alpha/2}\sqrt{\frac{p_0(1-p_0)}{n-1}\left(\frac{N-n}{N}\right)}\right)$
if $(levels > 0)$
return 1+ repartitions$_S\left(\frac{N}{2^{levels}}\right)$
else // *non-minimal partitioning*
return 0.

The data is scanned once to obtain the initial sample. Each time the data is repartitioned, the entire dataset is scanned and the new partitions written to disk. Then, each of the resulting partitions is scanned to obtain the random sample. Thus, the total number of reads and writes (disk blocks) is as follows:

$$reads_S = |R| * (2 * repartitions_S(\|R\|) + 1)$$
$$writes_S = |R| * repartitions_S(\|R\|).$$

Although the function is complicated, in practice we observe that the total number of repartitions is often just one. In this case, the entire dataset is read three times and written once.

6. EXPERIMENTAL PERFORMANCE EVALUATION

To evaluate the scalable algorithms (Rothko-T and Rothko-S), we conducted an extensive experimental evaluation. The evaluation is intended to address the following high-level questions.

—*Need for Scalable Algorithm*. We first seek to demonstrate the need to explicitly manage memory and I/O when anonymizing large datasets. (Section 6.2)

—*Evaluate and Compare Algorithms*. One of our main goals is to evaluate and compare our scalable algorithms (Rothko-T and Rothko-S). To this end, we perform an extensive experimental comparison of I/O behavior (Section 6.3) and total execution time (Section 6.4).

Table V. Experimental System Configuration

CentOS Linux (xfs file system)
512 MB memory
Intel Pentium 4 2.4 GHz processor
40 GB Maxtor IDE hard drive
(measured 54 MB/sec sequential bandwidth)
gcc version 3.4.4

Table VI. Synthetic Numeric Target Functions

	Target Function T
R7	$T = 0.67 \times (salary + commission)$ $- 0.2 \times loan - 20K$
R10	**if** $hyears < 20$ **then** $equity = 0$ **else** $equity = 0.1 \times hvalue \times (hyears - 20)$ $T = 0.67 \times (salary + commission)$ $- 5000 \times elevel + 0.2 \times equity - 10K$

—*Sampling and Data Quality*. When using a sample, the splits chosen according to the InfoGain and Regression split heuristics may not be identical to those chosen using the entire dataset. Section 6.5 evaluates the practical implications.

—*Evaluate Hypothesis Tests*. Our final set of experiments (Section 6.6) evaluates the effectiveness of the optimistic hypothesis-based splitting approach using a sample. By measuring the frequency of pruning, we show that the approach is quite effective. Also, though just rules of thumb, the tests described in Section 5.2.1 work quite well.

6.1 Experimental Set Up

We implemented each of the scalable algorithms using C++. In all cases, disk-resident data partitions were stored as ordinary files of fixed-width binary-encoded tuples. File reads and writes were buffered into 256K blocks. Table V describes our hardware/software configuration. In each of the experiments, we used a dedicated machine with an initially cold buffer cache.

Our experiments again made use of the synthetic data generator described in Section 4. The quasi-identifier attributes were generated following the distributions described in Table I. When necessary, categorical class labels were generated as a function of these attributes (see Table II). In addition, for regression tasks, numeric target attributes were generated using the functions described in Table VI. For these experiments, each quasi-identifier was treated as numeric (without user-defined generalization hierarchies), and each tuple was 44 bytes.

For the synthetic data, the size of an AV group (Median splitting) was approximately 8.1MB. Because the class label attribute has two distinct values, the size of an AVC group (InfoGain splitting) was approximately 16.2MB. Also, for the sampling-based algorithm, we fixed $\alpha = 0.05$ throughout the experimental evaluation.

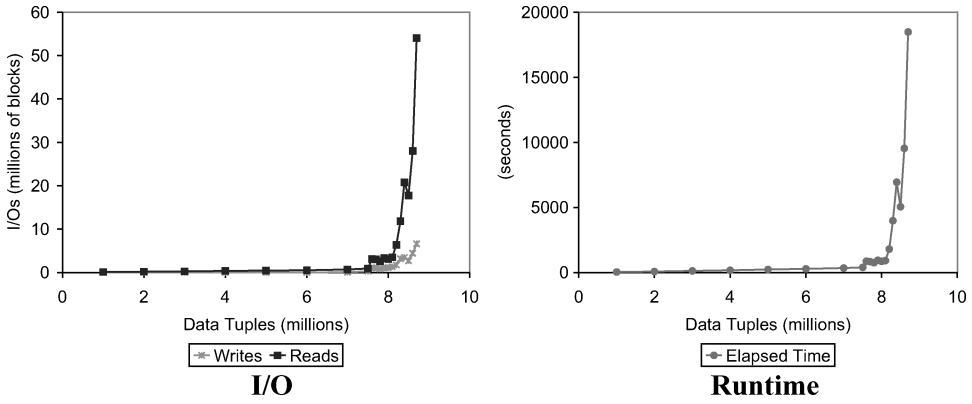

Fig. 17. In-memory implementation for large datasets.

6.2 Need for a Scalable Algorithm

When applied naively to large datasets, the Mondrian algorithms will often lead to thrashing and the expected poor performance. To illustrate the need to explicitly manage memory and I/O, we performed a simple experiment. We ran an in-memory implementation (also in C++), allowing the virtual memory system to manage memory and I/O. Figure 17 shows I/O behavior and runtime performance, respectively, for Median splitting and k-anonymity ($k = 1000$). As expected, the system begins to thrash for datasets that do not fit entirely in memory. These figures show performance for datasets containing up to 10 million records; in the remainder of this section, we will show that the scalable algorithms are easily applied to much larger datasets.

6.3 Counting I/O Requests

We begin by focusing on the I/O incurred by each of the two proposed algorithms. Each of the experiments in this section uses Linux `/proc/diskstat` to count the total number of I/O requests (in 512-byte blocks) issued to the disk. We also compare the experimental measurements to the values predicted by the analytical study in Section 5.3. All of the experiments in this section use Median partitioning and k-anonymity. The results are shown in Figure 18.

The first two experiments each used 50 million input tuples, and $k = 1000$. For Rothko-T, we fixed $TM = 2$ million and varied parameter F_CACHE. As expected, increasing F_CACHE reduces the number of I/O requests. However, the marginal improvement obtained from each additional frequency group is decreasing. In some cases, the observed number of disk reads is smaller than expected due to file system buffering.

For Rothko-S, we varied the sample size. Notice that for this wide range of sample sizes, the data was repartitioned just once, meaning that the algorithm read the entire dataset 3 times and wrote it once. Also, the total amount of I/O is substantially less than that of Rothko-T.

Finally, we performed a scale-up experiment, increasing the data size, and fixing $TM = 2$ million, $k = 1000$. Rothko-T is able to exploit the buffer

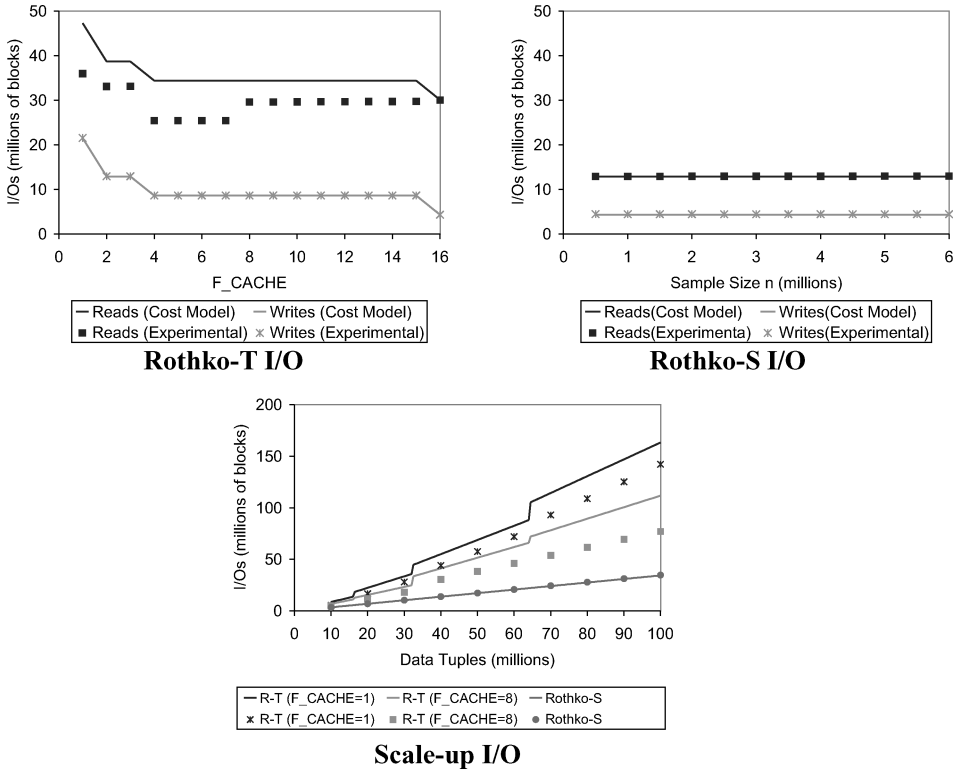

Fig. 18. I/O cost comparisons.

cache to some extent, but the total amount of I/O is substantially more than Rothko-S.

6.4 Runtime Performance

Perhaps more importantly, we evaluated the runtime performance of both proposed algorithms. All of the experiments in this section use k-anonymity as the anonymity requirement. In each case, we break down the execution time into three components: (1) user space CPU time, (2) kernel space CPU time, and (3) I/O wait time. These statistics were gathered from the system via `/proc/stat`.

We begin with Median splitting. The first set of experiments measured scale-up performance, fixing $TM = 2$ million, and $k = 1000$. Figure 19 shows results for Rothko-T ($F\_CACHE = 1$ and 8) and for Rothko-S. As expected, the sampling-based algorithm was faster both in terms of total execution time and CPU time. Additionally, each of the algorithms goes through periods where execution is I/O-bound. Interestingly, the I/O wait times are similar for Rothko-T ($F\_CACHE = 8$) and Rothko-S. However, this is deceptive. Although Rothko-T does more I/O, it also performs more in-memory calculations, thus occupying the CPU while the file system flushes the buffer cache asynchronously.

The second set of experiments considered the effects of parameter k. Results for these experiments are shown in Figure 20. As expected, a decreasing value

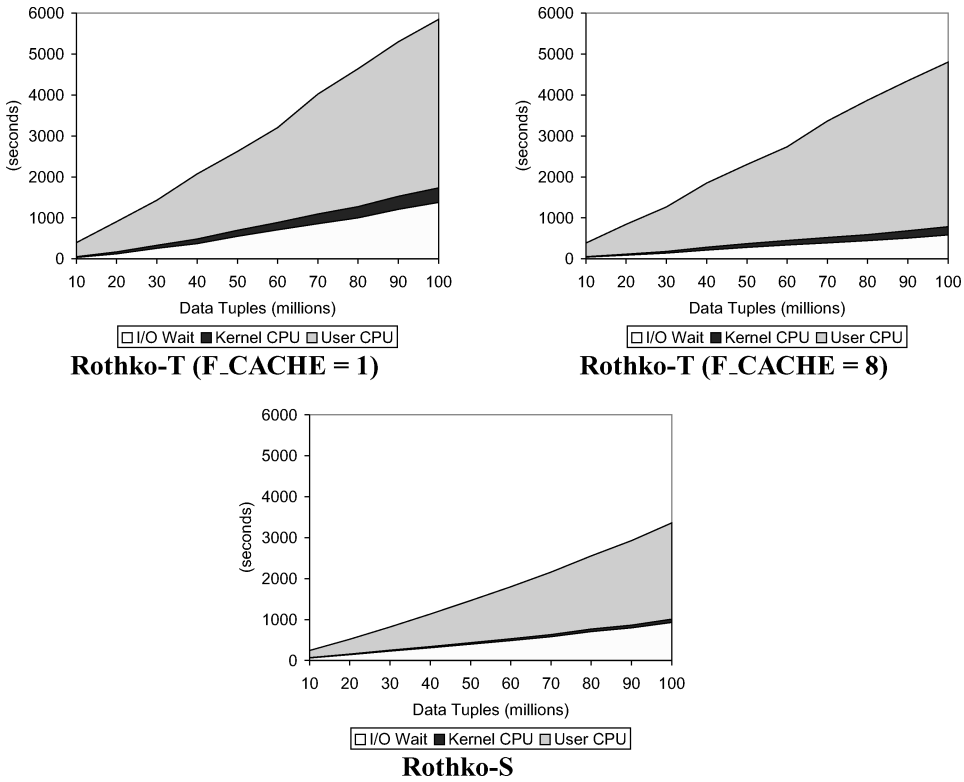

Fig. 19. Scale-up performance for Median splitting.

of k leads to more computation. However, because the algorithms all switch to the in-memory algorithm after some number of splits, this additional cost falls to the CPU.

Finally, we compared scale-up performance using the InfoGain split criterion, again fixing $TM = 2$ million, and $k = 1000$. For these experiments, we used label function C2 to generate the class labels. Figure 21 shows results for Rothko-T ($F\_CACHE = 1, 4$) and Rothko-S. As expected, the CPU cost incurred by these algorithms is greater than Median partitioning, particularly due to the extra cost of finding safe numeric thresholds that maximize information gain. However, Rothko-S consistently outperforms Rothko-T.[18]

6.5 Effects of Sampling on Data Quality

In Section 5.2.2, we discussed some of the potential shortcomings of the sampling-based algorithm, and we noted that one primary concern is imprecision with respect to the InfoGain and Regression split criteria. In this section,

[18]For efficiency, in each case, the recursive partitioning procedure switched to Median partitioning when the information gain resulting from a new split dipped below a 0.01. We note that continuing the InfoGain splitting all the way to the leaves is very CPU-intensive, particularly for numeric attributes because of the required sorting.

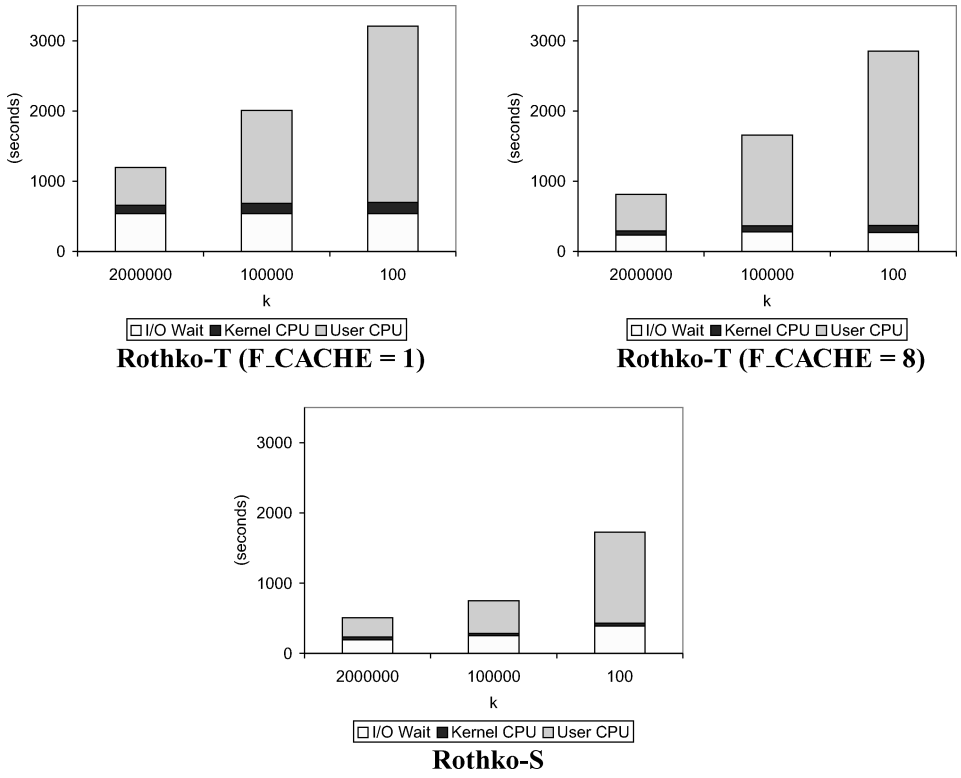

Fig. 20. Runtime performance for varied k.

we evaluate the effects of sampling with respect to data quality. For reasonably large sample sizes, we find that in practice the effect is often minimal.

In the interest of simplicity, in these experiments, when using the InfoGain split criterion, we measured the conditional entropy of the class label (C) with respect to the partitioning (see Equation (1)). For Regression splitting, we measured the weighted mean-squared error (see Equation (2)). Both of these measures relate directly to the given task.

We performed experiments using both synthetic and real-life data. Results for InfoGain splitting are shown in Figure 22. Results for Regression splitting are shown in Figure 23. For each experiment using synthetic data, we generated 10 datasets (each containing 100,000 records), and we increased the sample size. The reported results are averaged across the ten datasets. In the figures, we circled partitionings that are potentially nonminimal.

Increasing the sample size does lead to small improvement in quality (decreased entropy or error). However, for large sample sizes, the difference is very small. In all of our experiments the sample size had a much smaller impact on quality than the anonymity parameter k.

We also conducted a similar experiment using the Census database described in Table III. Again, the improvement in quality gained from increasing the sample size is small.

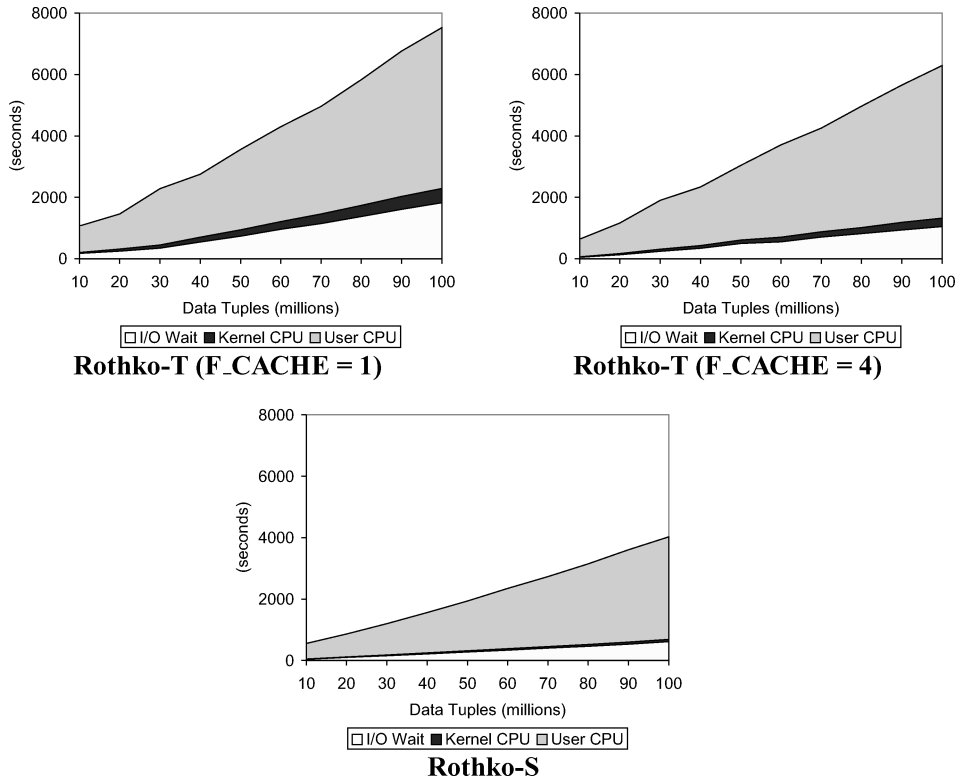

Fig. 21. Scale-up performance for InfoGain splitting.

6.6 Hypothesis Tests and Pruning

One of the important components in the design of the sampling-based algorithm is choosing an appropriate means of checking each anonymity requirement (k-anonymity, ℓ-diversity, and variance diversity) using a sample. Although the algorithm will always undo splits made in error, it is important to have a reasonable procedure in order to avoid excessive pruning.

In this section, we evaluate the effectiveness of the hypothesis-based approach described in Section 5.2.1. As mentioned previously, our hypothesis tests for ℓ-diversity and variance diversity are just rules of thumb. Nonetheless, we find that the approach of using a hypothesis test, as well as the specific tests outlined in Section 5.2.1, actually work quite well in practice.

We again used the synthetic data generator and the Median split criterion. For each experiment, we used an input of 100,000 tuples and varied the sample size. For each experiment, we repartitioned the data automatically when the height of the tree reached 8 (due to memory limitations for storing sensitive value histograms under variance diversity).

We conducted experiments using k-anonymity, ℓ-diversity, and variance diversity, each as the sole anonymity requirement in the respective experiment.

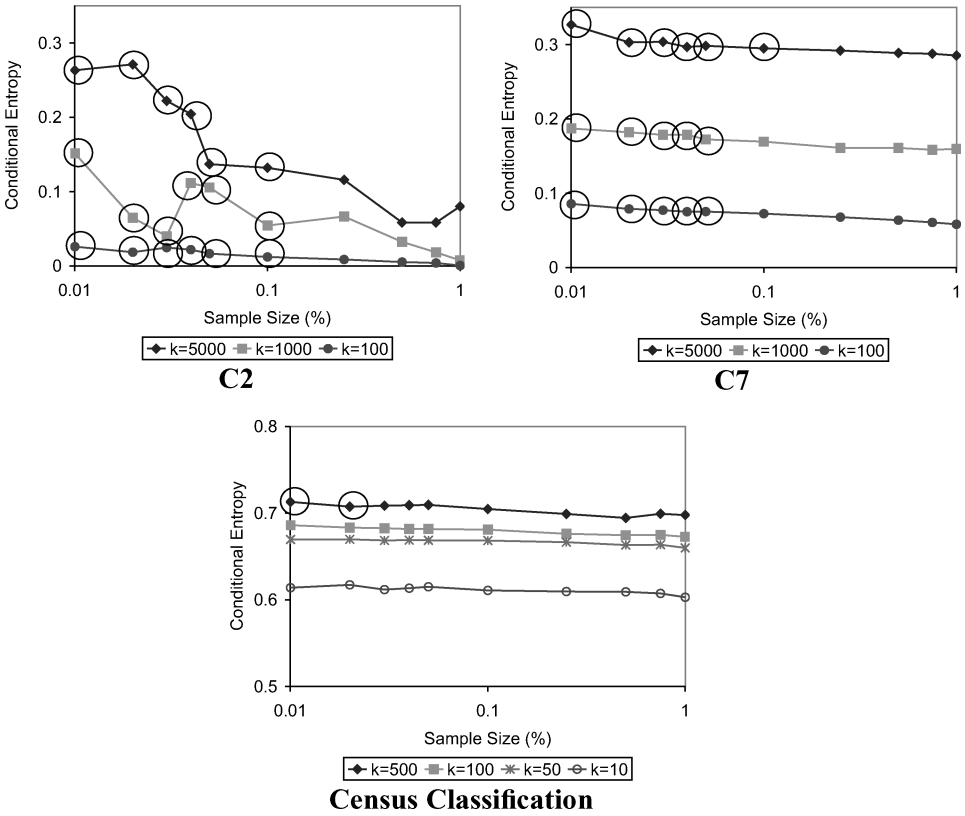

Fig. 22. Conditional entropy (InfoGain splitting).

For ℓ-diversity, we used zipcode as the sensitive attribute, and fixed $c = 1$. For variance diversity, we used salary as the sensitive attribute. In addition to the uniform salary distribution, we also considered a normal distribution. (The population variance of the uniform salary is approximately 1.4e9; the population variance of the normal salary is approximately 1.1e8.)

Figure 24 shows our results. Each entry indicates the total number of nodes that were pruned during the algorithm's entire execution. The numbers in parentheses indicate the number of nodes that are pruned when we use a naive approach that does not incorporate hypothesis tests (see Section 5.2.1). An x indicates that the resulting partitioning was (potentially) nonminimal as described in Section 5.2.2.

There are two important things to note from these results. First and foremost, the estimates are reasonably well behaved and do not lead to an excessive amount of pruning even for small samples. Similarly, although our hypothesis tests are just rules of thumb, they provide for much cleaner execution (less pruning) than the naive approach of using no hypothesis test.

As expected, the incidence of both nonminimality and pruning decreases with increased sample size.

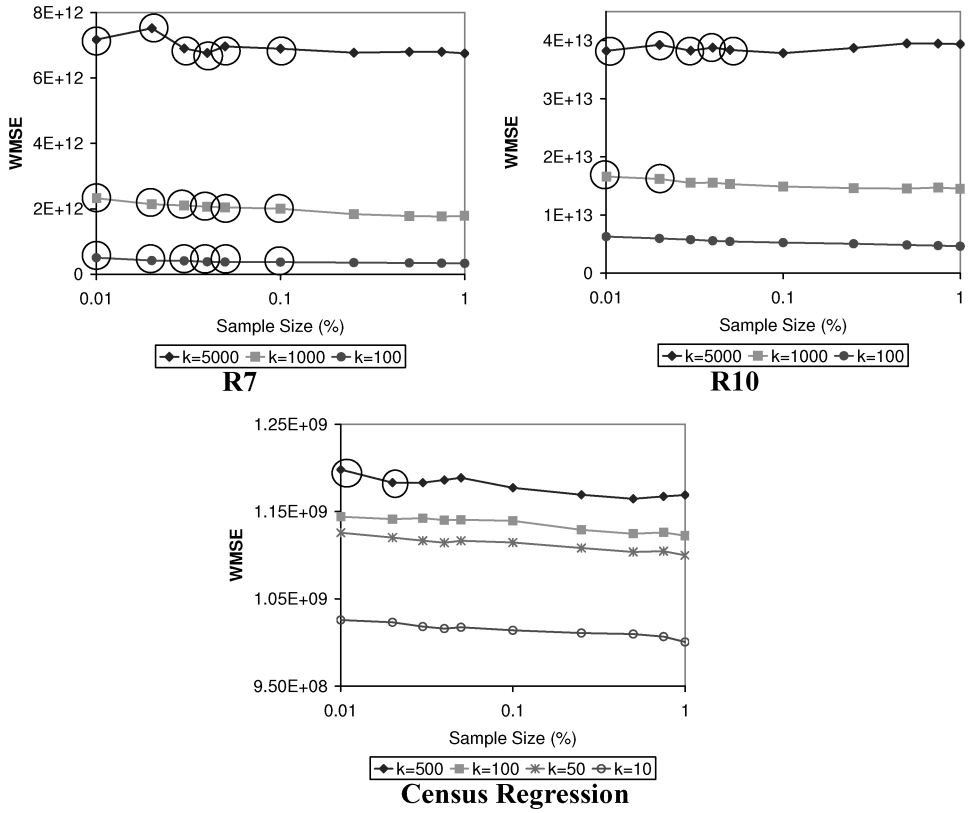

Fig. 23. WMSE (Regression splitting).

7. RELATED WORK

There has been substantial recent work on the problem of anonymization. In the context of k-anonymity/ℓ-diversity, much of the work initially focused on optimizing simple general-purpose measures of quality, such as the generalization height [Samarati 2001], the number of suppressed or generalized cells [Sweeney 2002b; Meyerson and Williams 2004; Aggarwal et al. 2005], the discernability penalty [Bayardo and Agrawal 2005], or the average equivalence class size [LeFevre et al. 2006a].

Subsequently, there has been interest in incorporating an understanding of workload when anonymizing data. This idea was first proposed by Iyengar [2002], who developed a genetic algorithm for single-dimensional recoding, incorporating a single target classification model. This approach proved costly, and subsequently there have been several proposed heuristic algorithms that also incorporate a single target classifier when performing k-anonymous single-dimensional recoding [Wang et al. 2004; Fung et al. 2005]. However, previous work has not considered incorporating a large family of workload tasks, including multiple predictive models and selection predicates.

In addition to generalization, related techniques have also been proposed, including microaggregation [Domingo-Ferrer and Mateo-Sanz 2002],

	k			
n	10	100	1000	10000
100	68 (1384)	x (x)	x (x)	x (x)
250	30 (1110)	7 (97)	x (x)	x (x)
500	11 (419)	12 (55)	x (x)	x (x)
1000	0 (0)	5 (6)	x (x)	x (x)
2500	0 (0)	2 (1)	1 (3)	x (x)
5000	0 (0)	0 (0)	1 (4)	x (x)
10000	0 (0)	0 (0)	0 (0)	x (x)
25000	0 (0)	0 (0)	0 (0)	0 (0)

(a) k-Anonymity

	ℓ			
n	2	4	6	8
100	97 (631)	x (x)	x (x)	x (x)
250	65 (87)	x (x)	8 (67)	x (x)
500	2 (763)	x (111)	2 (32)	x (x)
1000	112 (91)	1 (170)	1 (33)	x (16)
2500	0 (0)	0 (2)	1 (0)	0 (2)
5000	0 (0)	0 (2)	0 (0)	0 (9)
10000	0 (0)	0 (1)	0 (0)	0 (0)
25000	0 (0)	0 (1)	0 (0)	0 (0)

(b) ℓ-Diversity

	v		
n	1.1e9	1.2e9	1.3e9
100	x (x)	x (x)	x (x)
250	x (510)	x (x)	x (x)
500	x (443)	x (434)	x (x)
1000	0 (456)	x (372)	x (x)
2500	0 (0)	0 (87)	x (146)
5000	0 (0)	0 (0)	0 (47)
10000	0 (0)	0 (0)	0 (5)
25000	0 (0)	0 (2)	0 (7)

(c) Variance Diversity (Uniform S)

	v		
n	7e7	8e7	9e7
100	x (x)	x (x)	x (x)
250	x (396)	x (x)	x (x)
500	x (599)	x (457)	x (x)
1000	0 (402)	0 (405)	x (278)
2500	0 18)	0 (66)	x (169)
5000	0 (0)	0 (0)	0 (6)
10000	0 (0)	0 (0)	0 (13)
25000	0 (0)	0 (0)	0 (17)

(d) Variance Diversity (Normal S)

Fig. 24. Number of nodes pruned by Rothko-S as a function of the sample size n.

k-anonymous clustering [Aggarwal et al. 2006], and histogram sanitization [Chawla et al. 2005]. Kifer and Gehrke [2006] additionally considered releasing multiple (generalized) projections.

Aside from k-anonymity/ℓ-diversity, a variety of other techniques have been proposed for protecting (various definitions of) privacy while allowing certain data mining tasks. One widely-studied approach is based on the idea of adding random noise to the data, and then reconstructing attribute distributions in order to carry out one or more data mining tasks [Agrawal and Srikant 2000; Evfimievski et al. 2002; Rizvi and Haritsa 2002]. Also, Aggarwal and Yu [2004] propose first clustering the data into groups of required minimum occupancy, and then generating synthetic data based on summary statistics of each group. Pseudorandom sketches [Mishra and Sandler 2006] and answering statistical

and data mining queries using output perturbation primitives [Blum et al. 2005; Dwork et al. 2006; Dwork 2006] have also been proposed. This work is all related to ours because it also considers the effects of privacy and anonymization on various data mining and learning tasks. One appealing characteristic of the generalization approach is that the released data is semantically consistent with the original data, though at a coarsened level of granularity. This allows additional workloads to be carried out using the data, including selections.

Only a few of the proposed anonymization algorithms have specifically considered datasets larger than main memory. Incognito [LeFevre et al. 2005] operated on external (disk-resident) data, but the complexity of the algorithm was exponential in the number of attributes in the quasi-identifier, making it impractical in many situations.

In the context of location-based services, Mokbel et al. [2006] proposed using a scalable grid-based structure to implement k-anonymity. However, the proposed algorithms were not designed to incorporate additional anonymity requirements (e.g., ℓ-diversity) or workload-oriented splitting heuristics (e.g., InfoGain splitting). Also, they were designed to handle 2-dimensional spatial data, and it is not immediately clear how they would scale to data with higher dimensionality. Iwuchukwu and Naughton [2007] developed an algorithm for k-anonymity using incrementally-constructed R-trees, but this also does not support workload-oriented splitting.

To the best of our knowledge, all of the other proposed algorithms were designed to handle only memory-resident data, and none has been evaluated with respect to data substantially larger than the available memory.

8. CONCLUSIONS AND FUTURE WORK

This article has considered the problem of measuring the quality of anonymized data. It is our position that the most direct way of measuring quality is with respect to the purpose for which the data will be used. For this reason, we developed a suite of techniques for incorporating a family of tasks (comprised of queries, classification, and regression models) directly into the anonymization procedure. An extensive empirical study indicates that this typically leads to high-quality data. Further, the quality of the data with respect to a particular workload is not necessarily correlated with simple general-purpose measures that have been proposed in the previous literature.

In the second half of the article, we addressed the problem of scalability. We developed two techniques that allow our anonymization algorithms to be applied to datasets much larger than main memory. The first technique is based on ideas from scalable decision trees [Gehrke et al. 1998], and the second is based on sampling. An experimental evaluation and analytical study indicate that these techniques work very well in practice.

There are several interesting opportunities for future work both in data quality and performance. In this article, we developed techniques for incorporating classification and regression tasks as well as selection queries defined by rectangular regions. In the future, there are many other types of tasks one might be interested in performing on anonymized data.

Also, we considered using sampling as a way to scale an anonymization algorithm to datasets larger than main memory. The hypothesis tests we developed (Section 5.2.1) for k-anonymity, recursive (c, ℓ)-diversity, and variance diversity are reasonable rules of thumb particularly for the large samples encountered in the external algorithm. However, if we had a more precise set of tests (and precise characterizations of power and significance levels), it is reasonable to believe that we could also apply a sampling-based algorithm to enhance the performance of the in-memory case, choosing sample sizes in accordance with the given test(s).

REFERENCES

ADAM, N. AND WORTMANN, J. 1989. Security-control methods for statistical databases. *ACM Comput. Surv. 21*, 4, 515–556.

AGGARWAL, C. AND YU, P. 2004. A condensation approach to privacy-preserving data mining. In *Proceedings of the 9th International Conference on Extending Database Technology (EDBT)*.

AGGARWAL, G., FEDER, T., KENTHAPADI, K., MOTWANI, R., PANIGRAHY, R., THOMAS, D., AND ZHU, A. 2005. Anonymizing tables. In *Proceedings of the 10th International Conference on Database Theory (ICDT)*.

AGGARWAL, G., FEDER, T., KENTHAPADI, K., PANIGRAHY, R., THOMAS, D., AND ZHU, A. 2006. Achieving anonymity via clustering in a metric space. In *Proceedings of the 25th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS)*.

AGRAWAL, R., GHOSH, S., IMIELINSKI, T., AND SWAMI, A. 1993. Database mining: A performance perspective. In *IEEE Trans. Knowl. Data Engin. 5*.

AGRAWAL, R. AND SRIKANT, R. 2000. Privacy-preserving data mining. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*.

BAYARDO, R. AND AGRAWAL, R. 2005. Data privacy through optimal k-anonymity. In *Proceedings of the 21st International Conference on Data Engineering (ICDE)*.

BLAKE, C. AND MERZ, C. 1998. UCI repository of machine learning databases. University of California Irvine.

BLUM, A., DWORK, C., MCSHERRY, F., AND NISSIM, K. 2005. Practical privacy: The SuLQ framework. In *Proceedings of the 24th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS)*.

BREIMAN, L., FREIDMAN, J., OLSHEN, R., AND STONE, C. 1984. *Classification and Regression Trees*. Wadsworth International Group, Belmont, CA.

CHAWLA, S., DWORK, C., MCSHERRY, F., SMITH, A., AND WEE, H. 2005. Toward privacy in public databases. In *Proceedings of the 2nd Theory of Cryptography Conference*.

CHEN, B., CHEN, L., LIN, Y., AND RAMAKRISHNAN, R. 2005. Prediction cubes. In *Proceedings of the 31st International Conference on Very Large Databases (VLDB)*.

CHEN, B., LEFEVRE, K., AND RAMAKRISHNAN, R. 2007. PrivacySkyline: Privacy with multidimensional adversarial knowledge. In *Proceedings of the 33rd International Conference on Very Large Databases (VLDB)*.

DOMINGO-FERRER, J. AND MATEO-SANZ, J. 2002. Practical data-oriented microaggregation for statistical disclosure control. *IEEE Trans. Knowl. Data Engin. 4*, 1.

DWORK, C. 2006. Differential privacy. In *Proceedings of the 33rd International Colloquium on Automata, Languages, and Programming (ICALP)*.

DWORK, C., MCSHERRY, F., NISSIM, K., AND SMITH, A. 2006. Calibrating noise to sensitivity in private data analysis. In *Proceedings of the 3rd Theory of Cryptography Conference*.

EVFIMIEVSKI, A., SRIKANT, R., AGRAWAL, R., AND GEHRKE, J. 2002. Privacy preserving mining of association rules. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

FUNG, B., WANG, K., AND YU, P. 2005. Top-down specialization for information and privacy preservation. In *Proceedings of the 21st International Conference on Data Engineering (ICDE)*.

GEHRKE, J., GANTI, V., RAMAKRISHNAN, R., AND LOH, W. 1999. BOAT: Optimistic decision tree construction. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*.

GEHRKE, J., RAMAKRISHNAN, R., AND GANTI, V. 1998. RainForest: A framework for fast decision tree construction of large datasets. In *Proceedings of the 24th International Conference on Very Large Databases (VLDB)*.

HIP. 2002. Standards for privacy of individuals identifiable health information. U.S. Department of Health and Human Services.

IWUCHUKWU, T. AND NAUGHTON, J. 2007. K-anonymization as spatial indexing: Toward scalable and incremental anonymization. In *Proceedings of the 33rd International Conference on Very Large Databases (VLDB)*.

IYENGAR, V. 2002. Transforming data to satisfy privacy constraints. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

KENTHAPADI, K., MISHRA, N., AND NISSIM, K. 2005. Simulatable auditing. In *Proceedings of the 24th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS)*.

KIFER, D. AND GEHRKE, J. 2006. Injecting utility into anonymized datasets. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*.

LEFEVRE, K., DEWITT, D., AND RAMAKRISHNAN, R. 2005. Incognito: Efficient full-domain k-anonymity. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*.

LEFEVRE, K., DEWITT, D., AND RAMAKRISHNAN, R. 2006a. Mondrian multidimensional k-anonymity. In *Proceedings of the 22nd International Conference on Data Engineering (ICDE)*.

LEFEVRE, K., DEWITT, D., AND RAMAKRISHNAN, R. 2006b. Workload-aware anonymization. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

LI, N., LI, T., AND VENKATASUBRAMANIAN, S. 2007. t-Closeness: Privacy beyond k-anonymity and l-diversity. In *Proceedings of the IEEE International Conference on Data Engineering (ICDE)*.

MACHANAVAJJHALA, A., GEHRKE, J., KIFER, D., AND VENKITASUBRAMANIAM, M. 2006. l-Diversity: Privacy beyond k-anonymity. In *Proceedings of the 22nd International Conference on Data Engineering (ICDE)*.

MARTIN, D., KIFER, D., MACHANAVAJJHALA, A., GEHRKE, J., AND HALPERN, J. 2007. Worst-case background knowledge in privacy. In *Proceedings of the IEEE International Conference on Data Engineering (ICDE)*.

MEYERSON, A. AND WILLIAMS, R. 2004. On the complexity of optimal k-anonymity. In *Proceedings of the 23rd ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS)*.

MISHRA, N. AND SANDLER, M. 2006. Privacy via pseudorandom sketches. In *Proceedings of the 25th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS)*.

MOKBEL, M., CHOW, C., AND AREF, W. 2006. The new casper: Query processing for location services without compromising privacy. In *Proceedings of the 32nd International Conference on Very Large Databases (VLDB)*.

QUINLAN, R. 1993. *C4.5 Programs for Machine Learning*. Morgan Kaufmann, San Francisco, CA.

RIZVI, S. AND HARITSA, J. R. 2002. Maintaining data privacy in association rule mining. In *Proceedings of the 28th International Conference on Very Large Databases (VLDB)*.

SAMARATI, P. 2001. Protecting respondants' identities in microdata release. *IEEE Trans. Knowl. Data Engin. 13*, 6.

SWEENEY, L. 2002a. Achieving k-anonymity privacy protection using generalization and suppression. *Inter. J. Uncertainty, Fuzziness, Knowl.-Based Syst. 10*, 5, 571–588.

SWEENEY, L. 2002b. K-anonymity: A model for protecting privacy. *Inter. J. Uncertainty, Fuzziness, Knowl.-Based Syst. 10*, 5, 557–570.

WANG, K. AND FUNG, B. 2006. Anonymizing sequential releases. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

WANG, K., YU, P., AND CHAKRABORTY, S. 2004. Bottom-up generalization: A data mining solution to privacy protection. In *Proceedings of the 4th IEEE International Conference on Data Mining (ICDM)*.

WITTEN, I. AND FRANK, E. 2005. *Data Mining: Practical Machine Learning Yools and Techniques* 2nd Ed. Morgan Kaufmann, San Francisco, CA.

XIAO, X. AND TAO, Y. 2006. Personalized privacy preservation. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*.

XIAO, X. AND TAO, Y. 2007. m-Invariance: Towards privacy preserving re-publication of dynamic datasets. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*.

YAO, C., WANG, X., AND JAJODIA, S. 2005. Checking for k-anonymity violation by views. In *Proceedings of the 31st International Conference on Very Large Databases (VLDB)*.

ZHANG, J. AND HONAVAR, V. 2003. Learning decision tree classifiers from attribute value taxonomies and partially specified data. In *Proceedings of the 20th International Conference on Machine Learning (ICML)*.

Received June 2007; revised Januray 2008, May 2008; accepted June 2007