Plausible Deniability for Privacy-Preserving Data Synthesis

Vincent Bindschaedler
UIUC
bindsch2@illinois.edu

Reza Shokri
Cornell Tech
shokri@cornell.edu

Carl A. Gunter
UIUC
cgunter@illinois.edu

ABSTRACT

Releasing full data records is one of the most challenging problems in data privacy. On the one hand, many of the popular techniques such as data de-identification are problematic because of their dependence on the background knowledge of adversaries. On the other hand, rigorous methods such as the exponential mechanism for differential privacy are often computationally impractical to use for releasing high dimensional data or cannot preserve high utility of original data due to their extensive data perturbation.

This paper presents a criterion called *plausible deniability* that provides a formal privacy guarantee, notably for releasing sensitive datasets: an output record can be released only if a certain amount of input records are indistinguishable, up to a privacy parameter. This notion does not depend on the background knowledge of an adversary. Also, it can efficiently be checked by privacy tests. We present mechanisms to generate *synthetic datasets* with similar statistical properties to the input data and the same format. We study this technique both theoretically and experimentally. A key theoretical result shows that, with proper randomization, the plausible deniability mechanism generates differentially private synthetic data. We demonstrate the efficiency of this generative technique on a large dataset; it is shown to preserve the utility of original data with respect to various statistical analysis and machine learning measures.

1. INTRODUCTION

There is tremendous interest in releasing datasets for research and development. Privacy policies of data holders, however, prevent them from sharing their sensitive datasets. This is due, to a large extent, to multiple failed attempts of releasing datasets using imperfect privacy-preserving mechanisms such as de-identification. A range of inference attacks on, for example, AOL search log dataset [2], Netflix movie rating dataset [39], Genomic data [48, 22], location data [18, 46], and social networks data [40], shows that simple modification of sensitive data by removing identifiers or by generalizing/suppressing data features results in major information leakage and cannot guarantee meaningful privacy for data owners. These simple de-identification solutions, however, preserve data utility as they impose minimal perturbation to real data.

Rigorous privacy definitions, such as differential privacy [15], can theoretically guarantee privacy and bound information leakage about sensitive data. However, known mechanisms, such as the Laplacian mechanism [15] or the exponential mechanism [37], that achieve differential privacy through randomization, have practical limitations. The majority of scenarios, where they have been applied, are limited to interactive count queries on statistical databases [14]. In a non-interactive setting for releasing generic datasets, these mechanisms are either computationally infeasible on *high-dimensional* data, or practically ineffective because of their large *utility* costs [26]. At best, these methods are used to release some privacy-preserving statistics (e.g., histograms [6, 51]) about a dataset, but not *full* data records. It is not obvious how to protect the privacy of full records as opposed to that of aggregate statistics (by adding random noise).

Despite all these obstacles, releasing full data records is firmly pursued by large-scale data holders such as the U.S. Census Bureau [21, 28, 27]. The purpose of this endeavor is to allow researchers to develop analytic techniques by processing full synthetic data records rather than a limited set of statistics. Synthetic data could also be used for educational purpose, application development for data analysis, sharing sensitive data among different departments in a company, developing and testing pattern recognition and machine learning models, and algorithm design for sensitive data. There exists some inference-based techniques to *assess* the privacy risks of releasing synthetic data [42, 43]. However, the major open problem is how to *generate* synthetic full data records with *provable privacy*, that experimentally can achieve acceptable utility in various statistical analytics and machine learning settings.

In this paper, we fill this major gap in data privacy by proposing a generic theoretical framework for generating synthetic data in a privacy-preserving manner. The fundamental difference between our approach and that of existing mechanisms for differential privacy (e.g., exponential mechanism) is that we disentangle the data generative model from privacy definitions. Instead of forcing a generative model to be privacy-preserving by design, which might significantly degrade its utility, we can use a utility-preserving generative model and release only a subset of its output that satisfies our privacy requirements. Thus, for designing a generative

Proceedings of the VLDB Endowment, Vol. 10, No. 5

model, we rely on the state-of-the-art techniques from data science independently from the privacy requirements. This enables us to generate high utility synthetic data.

We formalize the notion of *plausible deniability* for data privacy [3], and generalize it to any type of data. Consider a probabilistic generative model that transforms a real data record, as its seed, into a synthetic data record. We can sample many synthetic data records from each seed using such a generative model. According to our definition, a synthetic record provides plausible deniability if there exists a set of real data records that could have generated the same synthetic data with (more or less) the same probability by which it was generated from its own seed. We design a privacy mechanism that provably guarantees plausible deniability. This mechanism results in *input indistinguishability*: by observing the output set (i.e., synthetics), an adversary cannot tell for sure whether a particular data record was in the input set (i.e., real data). The degree of this indistinguishability is a parameter in our mechanism.

Plausible deniability is a property of the overall process, and similar to differential privacy, it is independent of any adversary's background knowledge. In fact, we prove that our proposed plausible deniable data synthesis process can also satisfy differential privacy, if we randomize the indistinguishability parameter in the privacy mechanism. This is a significant theoretical result towards achieving strong privacy using privacy-agnostic utility-preserving generative models. Thus, we achieve differential privacy *without* artificially downgrading the utility of the synthesized data through output perturbation.

The process of generating a single synthetic data record and testing its plausible deniability can be done independently from that of other data records. Thus, millions of data records can be generated and processed in parallel. This makes our framework extremely efficient and allows implementing it at a large scale. In this paper, we develop our theoretical framework as an open-source tool, and run it on a large dataset: the American Community Survey [47] from the U.S. Census Bureau which contains over 3.1 million records. In fact, we can generate over one million privacy-preserving synthetic records in less than one hour on a multi-core machine running 12 processes in parallel.

We analyze the utility of synthetic data in two major scenarios: extracting statistics for data analysis, and performing prediction using machine learning. We show that our privacy test does not impose high utility cost. We also demonstrate that a significant fraction of candidate synthetic records proposed by a generative model can pass the privacy test even for strict privacy parameters.

We show that a strong adversary cannot distinguish a synthetic record from a real one with better than 63.0% accuracy (baseline: 79.8%). Furthermore, when it comes to classification tasks, the accuracy of the model learned on a synthetic dataset is only slightly lower than that of model trained on real data. For example, for Random Forest the accuracy is 75.3% compared to 80.4% when trained on real data (baseline: 63.8%); whereas for AdaBoostM1 the accuracy is 78.1% compared to 79.3% when trained on real data (baseline: 69.2%). Similar results are obtained when we compare logistic regression (LR) and support vector machine (SVM) classifiers trained on our synthetic datasets with the same classifiers trained (on real data) in a differential private way (using state-of-the-art techniques). Concretely, the accuracy of classifiers trained on our synthetic data is 77.5% (LR) and 77.1% (SVM); compared to 76.3% (LR) and 78.2% (SVM) for objective-perturbation ε-DP classifiers.

Contributions. We introduce a formal framework for plausible deniability as a privacy definition. We also design a mechanism to achieve it for the case of generating synthetic data. We prove that using a randomized test in our plausible deniability mechanism achieves differential privacy (which is a stronger guarantee). We also show how to construct generative models with differential privacy guarantees. The composition of our generative model and plausible deniability mechanism also satisfies differential privacy. We show the high accuracy of our model and utility of our generated synthetic data. We develop a generic tool and show its high efficiency for generating millions of full data records.

2. PLAUSIBLE DENIABILITY

In this section, we formalize plausible deniability as a new privacy notion for releasing privacy-preserving synthetic data. We also present a mechanism to achieve it. Finally, we prove that our mechanism can also satisfy differential privacy (which is a stronger guarantee) by slightly randomizing our plausible deniability mechanism.

Informally, plausible deniability states that an adversary (with any background knowledge) cannot deduce that a particular record in the input (real) dataset was significantly more responsible for an observed output (synthetic record) than was a collection of other input records. A mechanism ensures plausible deniability if, for a privacy parameter $k > 0$, there are at least k input records that could have generated the observed output with similar probability.

Unlike the majority of existing approaches (e.g., to achieve differential privacy), designing a mechanism to satisfy plausible deniability for generative models does not require adding artificial noise to the generated data. Instead, we separate the process of releasing privacy-preserving data into running two independent modules: (1) generative models, and (2) privacy test. The first consists in constructing a utility-preserving generative data model. This is ultimately a data science task which requires insight into the type of data for which one wants to generate synthetics. By contrast, the privacy test aims to safeguard the privacy of those individuals whose data records are in the input dataset. Every generated synthetic is subjected to this privacy test; if it passes the test it can be safely released, otherwise it is discarded. This is where the plausible deniability criterion comes into the frame: the privacy test is designed to ensure that any released output can be plausibly denied.

In this section, we assume a generic generative model that, given a data record in the input dataset as seed, produces a synthetic data record. In Section 3, we present a generic generative model based on statistical models, and show how it can be constructed in a differentially-private manner, so that it does not significantly leak about its own training data. Plausibly deniable mechanisms protect the privacy of the seeds, and are not concerned about how the generative models are constructed.

Let $\mathcal{M}$ be a probabilistic generative model that given any data record d can generate synthetic records y with probability $\Pr\{y = \mathcal{M}(d)\}$. Let $k \geq 1$ be an integer and $\gamma \geq 1$ be a real number. Both k and γ are privacy parameters.

Definition 1 (Plausible Deniability). *For any dataset D with $|D| \geq k$, and any record y generated by a probabilistic generative model $\mathcal{M}$ such that $y = \mathcal{M}(d_1)$ for $d_1 \in D$, we state that y is releasable with (k, γ)-*plausible deniability*, if there exist at least $k-1$ distinct records $d_2, ..., d_k \in D \setminus \{d_1\}$ such that*

$$\gamma^{-1} \leq \frac{\Pr\{y = \mathcal{M}(d_i)\}}{\Pr\{y = \mathcal{M}(d_j)\}} \leq \gamma, \quad (1)$$

for any $i, j \in \{1, 2, \ldots, k\}$.

The larger privacy parameter k is, the larger the indistinguishability set for the input data record. Also, the closer to 1 privacy parameter γ is, the stronger the indistinguishability of the input record among other plausible records.

Given a generative model $\mathcal{M}$, and a dataset D, we need a mechanism $\mathcal{F}$ to guarantee that the privacy criterion is satisfied for any released data. Specifically $\mathcal{F}$ produces data records by using $\mathcal{M}$ on dataset D. The following mechanism enforces (k, γ)-plausible deniability by construction.

Mechanism 1 ($\mathcal{F}$ with Plausible Deniability). *Given a generative model $\mathcal{M}$, dataset D, and parameters k, γ, output a synthetic record y or nothing.*

1. *Randomly sample a* seed *record $d \in D$.*
2. *Generate a* candidate *synthetic record $y = \mathcal{M}(d)$.*
3. *Invoke the* **privacy test** *on $(\mathcal{M}, D, d, y, k, \gamma)$.*
4. *If the tuple passes the test, then release y. Otherwise, there is no output.*

The core of Mechanism 1 ($\mathcal{F}$) is a privacy test that simply rejects a candidate synthetic data record if it does not satisfy a given privacy criterion.

We can think of Definition 1 as a *privacy criterion* that can be efficiently checked and enforced. So, instead of trying to measure how sensitive the model $\mathcal{M}$ is with respect to input data records, we test if there are enough indistinguishable records in the input dataset that could have (plausibly) generated a candidate synthetic data record.

Privacy Test 1 (Deterministic test $\mathcal{T}$). *Given a generative model $\mathcal{M}$, dataset D, data records d and y, and privacy parameters k and γ, output* pass *to allow releasing y, otherwise output* fail.

1. *Let $i \geq 0$ be the (only) integer that fits the inequalities*
$$\gamma^{-i-1} < \Pr\{y = \mathcal{M}(d)\} \leq \gamma^{-i} \ .$$
2. *Let k' be the number of records $d_a \in D$ such that*
$$\gamma^{-i-1} < \Pr\{y = \mathcal{M}(d_a)\} \leq \gamma^{-i} \ .$$
3. *If $k' \geq k$ then return* pass*, otherwise return* fail.

Step 2 counts the number of *plausible* seeds, i.e., records in D which could have plausibly produced y. Note that for a given y, there may exist some records $d_a \in D$ such that $\Pr\{y = \mathcal{M}(d_a)\} = 0$. Such records cannot be plausible seeds of y since no integer $i \geq 0$ fits the inequalities.

Remark that Privacy Test 1 ($\mathcal{T}$) enforces a stringent condition that the probability of generating a candidate synthetic y given the seed d and the probability of generating the same record given another plausible seed d_a both fall into a geometric range $[\gamma^{-i-1}, \gamma^{-i}]$, for some integer $i \geq 0$, assuming $\gamma > 1$. Notice that, under this test, the set of $k-1$ different d_as plus d satisfies the plausible deniability condition (1).

Informally, the threshold k prevents releasing the *implausible* synthetics records y. As k increases the number of plausible records which could have produced y also increases. Thus, an adversary with only partial knowledge of the input dataset cannot readily determine whether a particular input record d was the seed of any released record y. This is because there are at least $k-1$ other records $d_i \neq d$ in the input dataset which could *plausibly* have been the seed. However, whether y passes the privacy test itself reveals something about the number of plausible seeds, which could potentially reveal whether a particular d is included in the input data. This can be prevented by using a privacy test which randomizes the threshold k (as Section 2.1 shows) in which case the mechanism achieves (ε, δ)-differential privacy.

2.1 Relationship with Differential Privacy

We show a connection between Plausible Deniability and Differential Privacy, given the following definition.

Definition 2 (Differential Privacy [16]). *Mechanism F satisfies* (ε, δ)-differential privacy *if for any neighboring datasets D, D', and any output $S \subseteq$ Range(F):*

$$\Pr\{F(D') \in S\} \leq e^{\varepsilon} \Pr\{F(D) \in S\} + \delta \ .$$

Typically, one chooses δ smaller than an inverse polynomial in the size of the dataset, e.g., $\delta \leq |D|^{-c}$, for some $c > 1$.

In this section, we prove that if the privacy test is randomized in a certain way, then Mechanism 1 ($\mathcal{F}$) is in fact (ε, δ)-differentially private for some $\delta > 0$ and $\varepsilon > 0$. Privacy Test 1 simply counts the number of plausible seeds for an output and only releases a candidate synthetic if that number is at least k. We design Privacy Test 2 which is identical except that it randomizes the threshold k.

Privacy Test 2 (Randomized test $\mathcal{T}_{\epsilon_0}$). *Given a generative model $\mathcal{M}$, dataset D, data records d and y, privacy parameters k and γ, and randomness parameter ϵ_0, output* pass *to allow releasing y, otherwise output* fail.

1. *Randomize k by adding fresh noise: $\tilde{k} = k + \text{Lap}(\frac{1}{\epsilon_0})$.*
2. *Let $i \geq 0$ be the (only) integer that fits the inequalities*
$$\gamma^{-i-1} < \Pr\{y = \mathcal{M}(d)\} \leq \gamma^{-i} \ .$$
3. *Let k' be the number of records $d_a \in D$ such that*
$$\gamma^{-i-1} < \Pr\{y = \mathcal{M}(d_a)\} \leq \gamma^{-i} \ .$$
4. *If $k' \geq \tilde{k}$ then return* pass*, otherwise return* fail.

Here $z \sim \text{Lap}(b)$ is a sample from the Laplace distribution $\frac{1}{2b} \exp(\frac{-|z|}{b})$ with mean 0 and shape parameter $b > 0$.

Theorem 1 (Differential Privacy of $\mathcal{F}$). *Let $\mathcal{F}$ denote Mechanism 1 with the (randomized) Privacy Test 2 and parameters $k \geq 1$, $\gamma > 1$, and $\varepsilon_0 > 0$. For any neighboring datasets D and D' such that $|D|, |D'| \geq k$, any set of outcomes $Y \subseteq \mathcal{U}$, and any integer $1 \leq t < k$, we have:*

$$\Pr\{\mathcal{F}(D') \in Y\} \leq e^{\varepsilon} \Pr\{\mathcal{F}(D) \in Y\} + \delta \ ,$$

for $\delta = e^{-\varepsilon_0(k-t)}$ and $\varepsilon = \varepsilon_0 + \ln(1 + \frac{\gamma}{t})$.

The privacy level offered by Theorem 1 is meaningful provided k is such that δ is sufficiently small. For example, if we want $\delta \leq \frac{1}{n^c}$ for some $c > 1$, then we can set $k \geq t + \frac{c}{\varepsilon_0} \ln n$. Here t provides a trade-off between δ and ε.

The proof of Theorem 1 can be found in the extended version of the paper ([4] Appendix C). Roughly speaking, the theorem says that, except with some small probability δ, adding a record to a dataset cannot change the probability that any synthetic record y is produced by more than a small multiplicative factor. The intuition behind this is the following.

Fix an arbitrary synthetic record y. Observe that given y, all the records in the dataset are partitioned into (disjoint) sets according to their probabilities of generating y (with respect to $\mathcal{M}$). That is, the i^{th} partition (or set) contains those records d such that $\gamma^{-(i+1)} < \Pr\{y = \mathcal{M}(d)\} \leq \gamma^{-i}$. (Records d such that $\Pr\{y = \mathcal{M}(d)\} = 0$ can be ignored.) Note that: for y to be released from partition i, the seed must be in partition i, and it must pass the privacy test; and the probability of passing the privacy test depends only on the number of records in the partition of the seed.

Suppose we add d' to the dataset and let j be the partition that d' falls into. The number of plausible seeds can increase by at most one (this occurs when the seed is in partition j) and so the probability of passing the privacy test changes by a factor of at most e^{ε_0} due to adding Laplacian noise to the threshold k. Now, suppose the seed belongs to partition j. One the one hand, if partition j contains only l records, such that $l \ll k$, then the change in probability (due to adding d') could be unbounded. (For example, it could be that d' is the only record for which $\Pr\{y = \mathcal{M}(d')\} > 0$.) However, in this case, the probability of passing the privacy test is negligible (at most δ). On the other hand, if l is large enough (say $l \approx k$) so that passing the privacy test is likely, then the probability of generating y from partition j can only change by a small multiplicative factor. Indeed, the probabilities of generating y from d' or from any of the other l records in partition j are γ-close.

3. GENERATIVE MODEL

In this section, we present our generative model, and the process of using it to generate synthetic data. The core of our synthesizer is a probabilistic model that captures the joint distribution of attributes. We learn this model from training data samples drawn from our real dataset $\mathbb{D}$. Thus, the model itself needs to be privacy-preserving with respect to its training set. We show how to achieve this with differential privacy guarantees.

Let $\mathbb{D}_{\mathsf{S}}$, $\mathbb{D}_{\mathsf{T}}$, and $\mathbb{D}_{\mathsf{P}}$ be three non-overlapping subsets of dataset $\mathbb{D}$. We use these datasets in the process of synthesis, structure learning, and parameter learning, respectively.

3.1 Model

Let $\{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_m\}$ be the set of random variables associated with the attributes of the data records in $\mathbb{D}$. Let $\mathcal{G}$ be a directed acyclic graph (DAG), where the nodes are the random variables, and the edges represent the probabilistic dependency between them. A directed edge from $\mathbf{x}_j$ to $\mathbf{x}_i$ indicates the probabilistic dependence of attribute i to attribute j. Let $P_{\mathcal{G}}(i)$ be the set of parents of random variable i according to the dependency graph $\mathcal{G}$. The following model, which we use in Section 3.2 to generate synthetic data, represents the joint probability of data attributes.

$$\Pr\{\mathbf{x}_1, ..., \mathbf{x}_m\} = \prod_{i=1}^{m} \Pr\{\mathbf{x}_i \mid \{\mathbf{x}_j\}_{\forall j \in P_{\mathcal{G}}(i)}\} \tag{2}$$

This model is based on a structure between random variables, captured by $\mathcal{G}$, and a set of parameters that construct the conditional probabilities. In Section 3.3 and Section 3.4, we present our differentially-private algorithms to learn the structure and parameters of the model from $\mathbb{D}$, respectively.

3.2 Synthesis

Using a generative model, we probabilistically transform a real data record (called the seed) into a synthetic data record, by updating its attributes. Let $\{x_1, x_2, ..., x_m\}$ be the values for the set of data attributes for a randomly selected record in the seed dataset $\mathbb{D}_{\mathsf{S}}$. Let ω be the number of attributes for which we generate new values. Thus, we keep (i.e., copy over) the values of $m - \omega$ attributes from the seed to the synthetic data. Let σ be a permutation over $\{1, 2, ..., m\}$ to determine the re-sampling order of attributes.

We set the re-sampling order σ to be the dependency order between random variables. More precisely, $\forall j \in P_{\mathcal{G}}(i)$: $\sigma(j) < \sigma(i)$. We fix the values of the first $m - \omega$ attributes according to σ (i.e., the synthetic record and the seed overlap on their $\{\sigma(1), ..., \sigma(m - \omega)\}$ attributes). We then generate a new value for each of the remaining ω attributes, using the conditional probabilities (2). As we update the record while we re-sample, each new value can depend on attributes with updated values as well as the ones with original (seed) values.

We re-sample attribute $\sigma(i)$, for $i > m - \omega$, as

$$x'_{\sigma(i)} \sim \Pr\{\mathbf{x}_{\sigma(i)} \mid \{\mathbf{x}_{\sigma(j)} = x_{\sigma(j)}\}_{\forall j \in P_{\mathcal{G}}(i), j \leq m-\omega}, \{\mathbf{x}_{\sigma(j)} = x'_{\sigma(j)}\}_{\forall j \in P_{\mathcal{G}}(i), j > m-\omega}\} \tag{3}$$

In Section 2, we show how to protect the privacy of the *seed* data record using our plausible deniability mechanisms.

Baseline: Marginal Synthesis. As a baseline generative model, we consider a synthesizer that (independently from any seed record) samples a value for an attribute from its marginal distribution. Thus, for all attribute i, we generate $x_i \sim \Pr\{\mathbf{x}_i\}$. This is based on an assumption of independence between attributes' random variables, i.e., it assumes $\Pr\{\mathbf{x}_1, ..., \mathbf{x}_m\} = \prod_{i=1}^{m} \Pr\{\mathbf{x}_i\}$.

3.3 Privacy-Preserving Structure Learning

Our generative model depends on the dependency structure between random variables that represent data attributes. The dependency graph $\mathcal{G}$ embodies this structure. In this section, we present an algorithm that learns $\mathcal{G}$ from real data, in a privacy-preserving manner such that $\mathcal{G}$ does not significantly depend on individual data records.

The algorithm is based on maximizing a scoring function that reflects how correlated the attributes are according to the data. There are multiple approaches to this problem in the literature [35]. We use a method based on a well-studied machine learning problem: *feature selection.* For each attribute, the goal is to find the best set of features (among all attributes) to predict it, and add them as the attribute's parents, under the condition that the dependency graph remains acyclic.

The machine learning literature proposes several ways to rank features in terms of how well they can predict a particular attribute. One possibility is to calculate the information gain of each feature with the target attribute. The major downside with this approach is that it ignores the redundancy in information between the features. We propose to use a different approach, namely Correlation-based Feature Selection (CFS) [20] which consists in determining the best subset of predictive features according to some correlation measure. This is an optimization problem to select a subset of features that have high correlation with the target attribute and at the same time have low correlation among themselves. The task is to find the best subset of features which maximizes a merit score that captures our objective.

We follow [20] to compute the merit score for a parent set $P_{\mathcal{G}}(i)$ for attribute i as

$$\text{score}(P_{\mathcal{G}}(i)) = \frac{\sum_{j \in P_{\mathcal{G}}(i)} \text{corr}(\mathbf{x}_i, \mathbf{x}_j)}{\sqrt{|P_{\mathcal{G}}(i)| + \sum_{j,k \in P_{\mathcal{G}}(i)} \text{corr}(\mathbf{x}_j, \mathbf{x}_k)}}, \quad (4)$$

where $|P_{\mathcal{G}}(i)|$ is the size of the parent set, and corr() is the correlation between two random variables associated with two attributes. The numerator rewards correlation between parent attributes and the target attribute, and the denominator penalizes the inner-correlation among parent attributes. The suggested correlation metric in [20], which we use, is the symmetrical uncertainty coefficient:

$$\text{corr}(\mathbf{x}_i, \mathbf{x}_j) = 2 - 2\frac{\text{H}(\mathbf{x}_i, \mathbf{x}_j)}{\text{H}(\mathbf{x}_i) + \text{H}(\mathbf{x}_j)}, \quad (5)$$

where H() is the entropy function.

The optimization objective in constructing $\mathcal{G}$ is to maximize the total score$(P_{\mathcal{G}}(i))$ for all attributes i. Unfortunately, the number of possible solutions to search for is exponential in the number of attributes, making it impractical to find the optimal solution. The greedy algorithm, suggested in [20], is to start with an empty parent set for a target attribute and always add the attribute (feature) that maximizes the score.

There are two constraints in our optimization problem. First, the resulting dependency graph obtained from the set of best predictive features (i.e., parent attributes) for all attributes should be acyclic. This would allow us to decompose and compute the joint distribution over attributes as represented in (2).

Second, we enforce a maximum allowable complexity cost for the set of parents for each attribute. The cost is proportional to the number of possible joint value assignments (configurations) for the parent attributes. So, for each attribute i, the complexity cost constraint is

$$\text{cost}(P_{\mathcal{G}}(i)) = \prod_{j \in P_{\mathcal{G}}(i)} |\mathbf{x}_j| \leq \texttt{maxcost} \quad (6)$$

where $|\mathbf{x}_j|$ is the total number of possible values that the attribute j takes. This constraint prevents selecting too many parent attribute combinations for predicting an attribute. The larger the joint cardinality of attribute i's parents is, the fewer data points to estimate the conditional probability $\mathbb{P}\text{r}\{\mathbf{x}_i \,|\, \{\mathbf{x}_j\}_{\forall j \in P_{\mathcal{G}}(i)}\}$ can be found. This would cause overfitting the conditional probabilities on the data, that results in low confidence parameter estimation in Section 3.4. The constraint prevents this.

To compute the score and cost functions, we discretize the parent attributes. Let bkt() be a discretizing function that partitions an attribute's values into buckets. If the attribute is continuous, it becomes discrete, and if it is already discrete, bkt() might reduce the number of its bins. Thus, we update conditional probabilities as follows.

$$\mathbb{P}\text{r}\{\mathbf{x}_i \,|\, \{\mathbf{x}_j\}_{\forall j \in P_{\mathcal{G}}(i)}\} \approx \mathbb{P}\text{r}\{\mathbf{x}_i \,|\, \{\text{bkt}(\mathbf{x}_j)\}_{\forall j \in P_{\mathcal{G}}(i)}\} \quad (7)$$

where the discretization, of course, varies for each attribute. We update (4) and (6) according to (7). This approximation itself decreases the cost complexity of a parent set, and further prevents overfitting on the data.

3.3.1 Differential-Privacy Protection

In this section, we show how to safeguard the privacy of individuals whose records are in $\mathbb{D}$, and could influence the model structure (which might leak about their data).

All the computations required for structured learning are reduced to computing the correlation metric (5) from $\mathbb{D}$. Thus, we can achieve differential privacy [16] for the structure learning by simply adding appropriate noise to the metric. As, the correlation metric is based on the entropy of a single or a pair of random variables, we only need to compute the entropy functions in a differentially-private way. We also need to make sure that the correlation metric remains in the $[0, 1]$ range, after using noisy entropy values.

Let $\tilde{H}(\mathbf{z})$ be the noisy version of the entropy of a random variable $\mathbf{z}$, where in our case, $\mathbf{z}$ could be a single or pair of random variables associated with the attributes and their discretized version (as presented in (7)). To be able to compute differentially-private correlation metric in all cases, we need to compute noisy entropy $\tilde{H}(\mathbf{x}_i)$, $\tilde{H}(\text{bkt}(\mathbf{x}_i))$, $\tilde{H}(\mathbf{x}_i, \mathbf{x}_j)$, and $\tilde{H}(\mathbf{x}_i, \text{bkt}(\mathbf{x}_j))$, for all attributes i and j. For each of these cases, we generate a *fresh* noise drawn from the Laplacian distribution and compute the differentially-private entropy as

$$\tilde{H}(\mathbf{z}) = \text{H}(\mathbf{z}) + \text{Lap}(\frac{\Delta_H}{\varepsilon_H}) \quad (8)$$

where Δ_H is the sensitivity of the entropy function, and ε_H is the differential privacy parameter.

It can be shown that if $\mathbf{z}$ is a random variable with a probability distribution, estimated from $n_T = |\mathbb{D}_\mathsf{T}|$ data records, then the upper bound for the entropy sensitivity is

$$\Delta_H \leq \frac{1}{n_T}[2 + \frac{1}{\ln(2)} + 2\log_2 n_T] = O(\frac{\log_2 n_T}{n_T}) \quad (9)$$

The proof of (9) can be found in the extended version of the paper ([4] Appendix B). Remark that Δ_H is a function of n_T (the number of records in $\mathbb{D}_\mathsf{T}$) which per se needs to be protected. As a defense, we compute Δ_H in a differentially-private manner, by once randomizing the number of records

$$\tilde{n}_T = n_T + \text{Lap}(\frac{1}{\varepsilon_{n_T}}) \quad (10)$$

By using the randomized entropy values, according to (8), the model structure, which will be denoted by $\tilde{\mathcal{G}}$, is differentially private. In Section 3.5, we use the composition theorems to analyze the total privacy of our algorithm for obtaining a differentially-private structure.

3.4 Privacy-Preserving Parameter Learning

Having a dependency structure $\tilde{\mathcal{G}}$, we need to compute the conditional probabilities for predicting each of the attributes given its parent set (see (2)). This is a well-known problem in statistics. In this section, we show how to learn the parameters that represent such conditional probabilities, from $\mathbb{D}_\mathsf{P}$, in a differentially private manner.

The problem to be solved is to first learn a prior distribution over the parameters of the conditional probabilities. To do so, we learn the hyper-parameters (the parameters of the prior distribution over the model's parameters) from data. Only then, we can compute the parameters that form the conditional probabilities from the prior distribution.

Let us take the example of computing the parameters for predicting discrete/categorical attributes. In this case, we assume a multinomial distribution over the attribute's values (that fall into different bins). The conjugate prior for multinomials comes from a Dirichlet family. The Dirichlet distribution assigns probabilities to all possible multinomial distributions, according to the statistics obtained from a set of data records.

Let $|\mathbf{x}_i|$ be the number of distinct values that attribute i can take. The probability of some multinomial distribution parameters $\vec{p}_i^{\,c} = p_{i,1}^c, p_{i,2}^c, ..., p_{i,|\mathbf{x}_i|}^c$ to predict attribute i, under configuration c for $P_\mathcal{G}(i)$, is

$$\Pr\{\vec{p}_i^{\,c} \,|\, \tilde{\mathcal{G}}, \mathbb{D}_\mathsf{P}\} = \mathrm{Dir}(\vec{\alpha}_i^{\,c} + \vec{n}_i^{\,c}) \tag{11}$$

where $\vec{\alpha}_i^{\,c}$ is the vector of default hyper-parameters for the Dirichlet distribution, and $\vec{n}_i^{\,c}$ is the vector for the number of data records in $\mathbb{D}_\mathsf{P}$ with $P_{\tilde{\mathcal{G}}}(i)$ configuration c with different values for attribute i (i.e., element $n_{i,l}^c$ is the number of records for which $x_i = l$ and $P_{\tilde{\mathcal{G}}}(i)$ configuration is c). The Dirichlet distribution is computed as

$$\mathrm{Dir}(\vec{\alpha}_i^{\,c} + \vec{n}_i^{\,c}) = \prod_{i=1}^{m} \prod_{c=1}^{\#c} \Gamma(\alpha_i^c + n_i^c) \prod_{l=1}^{|\mathbf{x}_i|} \frac{(p_i^c)^{\alpha_{i,l}^c + n_{i,l}^c - 1}}{\Gamma(\alpha_{i,l}^c + n_{i,l}^c)} \tag{12}$$

where $\alpha_i^c = \sum_l \alpha_{i,l}^c$, and $n_i^c = \sum_l n_{i,l}^c$, and the number of configurations $\#c$ is $\prod_{j \in P_{\tilde{\mathcal{G}}}(i)} |\mathrm{bkt}(\mathbf{x_j})|$, which according to constraint (6) can at most be `maxcost`.

Learning the parameters of the model, in the case of a Dirichlet prior for multinomial distribution, is simply computing $\vec{n}_i^{\,c}$ from the data records in $\mathbb{D}_\mathsf{P}$. Given the probability distribution (11) over the multinomial parameters, we can compute the most likely set of parameters as

$$p_{i,l}^c = \frac{\alpha_{i,l}^c + n_{i,l}^c}{\alpha_i^c + n_i^c} \tag{13}$$

or, we can sample a set of multinomial parameters according to (12). This is what we do in our generative model, in order to increase the variety of data samples that we can generate.

Note that for computing the marginal distributions, that are needed for the baseline, we perform the same computations by setting the parent sets to be empty.

If an attribute is continuous, we can learn the parameters of a Normal distribution or learn a regression model from our data to construct its conditional probability. We omit the details here (as in the dataset we evaluate in Section 6 all attributes are discrete).

3.4.1 Differential-Privacy Protection

The parameters of the conditional probabilities depend on the data records in $\mathbb{D}_\mathsf{P}$, thus they can leak sensitive information about individuals who contributed to the real dataset. In this section, we show how to learn parameters of the attribute conditional probabilities (i.e., $\vec{p}_i^{\,c}$ values) with differential privacy guarantees.

Note that in (11), the only computations that are dependent on $\mathbb{D}_\mathsf{P}$ are the $\vec{n}_i^{\,c}$ counts (for all c and i). To find the variance of the noise to be added to these counts, to achieve differential privacy, we need to compute their sensitivity with respect to one individual's data record.

Suppose we are computing the parameters associated with predicting a given attribute i given its parent set $P_{\tilde{\mathcal{G}}}(i)$. Note that adding a record to $\mathbb{D}_\mathsf{P}$ increases exactly a single component $n_{i,l}^c$, for which it matches value l for attribute i and configuration c for its parent set. So, only one single element among all $\#c \times |\mathbf{x}_i|$ elements of $\vec{n}_i = \vec{n}_i^{\,1}, \vec{n}_i^{\,2}, ..., \vec{n}_i^{\,\#c}$ changes. This implies that the L1 sensitivity of $\vec{n}_i$ is 1. Consequently, a random noise drawn from $\mathrm{Lap}(\frac{1}{\varepsilon_p})$ can be added to each component of $\vec{n}_i$ independently. More precisely, for any attribute i value l, and configuration c, we randomize counts as

$$\tilde{n}_{i,l}^c = \max(0, n_{i,l}^c + \mathrm{Lap}(\frac{1}{\varepsilon_p})) \tag{14}$$

and use them to compute (11) with differential privacy.

3.5 Differential Privacy Analysis

In this section, we compute the differential privacy level that we can guarantee for the whole dataset $\mathbb{D}$ for learning the structure and the parameters of the model. We compute the total (ϵ, δ) privacy by composing the differentially-private mechanisms in Section 3.3.1 and Section 3.4.1.

Remark that we often protect the output $f(x)$ of some function f by adding noise from $\mathrm{Lap}(\frac{\Delta_f}{\varepsilon})$, where Δ_f is the L1 sensitivity of f. This mechanism is known as the Laplace mechanism and it satisfies ε-differential privacy (Theorem 3.6 of [16]).

Thus the $m(m+1)$ entropy values, $\tilde{H}(\mathbf{z})$, needed for structure learning (Section 3.3.1) are obtained in an a way that satisfies ε_H-differential privacy. This is also the case for the number of records n_T, i.e., it satisfies ε_{n_T}-differential privacy. Similarly, the counts $n_{i,l}^c$ parameters learned for each configuration (Section 3.4.1) satisfy ε_p-differential privacy.

For structure learning, we make use of both sequential composition (Theorem 2 in [4] Appendix A) and advanced composition (Theorem 3 in [4] Appendix A). Specifically, we use advanced composition for the $m(m+1)$ entropy values and sequential composition with the number of records. That is, the overall privacy achieved (of structure learning) is $(\varepsilon_L, \delta_L)$-differential privacy for a fixed $\delta_L \ll \frac{1}{n_T}$ and $\varepsilon_L = \varepsilon_{n_T} + \varepsilon_H \sqrt{2m(m+1)\ln(\delta_L^{-1})} + m(m+1)\varepsilon_H(e^{\varepsilon_H} - 1)$.

For the parameter learning (as explain in Section 3.4.1), for a given attribute i, the L1 sensitivity of all configurations of the parent set of i, i.e., $P_{\tilde{\mathcal{G}}}(i)$, is 1. The overall privacy achieved (of parameter learning) is $(\varepsilon_P, \delta_P)$-differential privacy using advanced composition over the m attributes. Here, $\delta_P \ll \frac{1}{n_p}$ (where n_p is the number of records in $\mathbb{D}_\mathsf{P}$) and $\varepsilon_P = \varepsilon_p \sqrt{2m\ln(\delta_P^{-1})} + m\varepsilon_p(e^{\varepsilon_p} - 1)$.

Table 1: Pre-processed ACS13 dataset attributes.

Name	Type	Cardinality (Values)
Age (AGEP)	Numerical	80 (17 to 96)
Workclass (COW)	Categorical	8
Education (SCHL)	Categorical	24
Martial Status (MAR)	Categorical	5
Occupation (OCCP)	Categorical	25
Relationship (RELP)	Categorical	18
Race (RAC1P)	Categorical	5
Sex (SEX)	Categorical	2 (male or female)
Hours Worked per Week (WKHP)	Numerical	100 (0 to 99+)
World Area of Birth (WAOB)	Categorical	8
Income Class (WAGP)	Categorical	2 ($\leq$ 50K, $>$ 50K)[USD]

Table 2: ACS13 data extraction and cleaning statistics.

Records	$3,132,796$ (clean: $1,494,974$)
Attributes	11 (numerical: 2, categorical: 9)
Possible Records	$540,587,520,000$ ($\approx 2^{39}$)
Unique Records	$1,022,718$ (68.4%)
Classification Task	Income class

Given that $\mathbb{D}_\mathsf{T}$ and $\mathbb{D}_\mathsf{P}$ are non-overlapping, the privacy obtained for the generative model is differentially private with parameters $(\max\{\varepsilon_L, \varepsilon_P\}, \max\{\delta_L, \delta_P\})$. Due to random subsampling of $\mathbb{D}_\mathsf{T}$ and $\mathbb{D}_\mathsf{P}$ from $\mathbb{D}$, the privacy parameters can be further improved by using the amplification effect of sampling (Theorem 4 in [4] Appendix A) to obtain (ε, δ)-differential privacy.

4. DATA

For validation, we use the 2013 American Community Survey (ACS) [47] from the U.S. Census Bureau. The dataset contains upwards of 3 million of individual records. Each record includes a variety of demographics attributes such as age, sex, race, as well as attributes related to the individual's income such as yearly income in USD.

The ACS dataset has been used for various purposes ranging from examining the relationship between education and earnings [24] to looking at current language use patterns of children of immigrants [38]. Furthermore, the prominent UCI Adultdataset, which provides a well-established benchmark for machine learning tasks, was extracted from the 1994 Census database. The 2013 ACS dataset contains similar attributes so we process it in a manner similar to how the Adult dataset was extracted. In particular, we extract the same attributes whenever possible.

As pre-processing, we discard records with missing or invalid values for the considered attributes (Table 1). Table 2 shows some statistics of the data cleaning and extracted dataset. This is a highly dimensional dataset despite having only 11 attributes, there are more than half a trillion *possible* records and out of the roughly 1.5 million records obtained after cleaning, approximately 2/3 are unique.

We bucketize (Section 3.3) values of the age attribute in bins (i.e., buckets) of 10, i.e., 17 to 26, 27 to 36, etc. (Following the rules used to extract the Adult dataset, we only consider individuals older than 16.) We also bucketize the values of: hours worked per weeks (HPW), in bins of 15 hours; education, to aggregate education level below a high-school diploma in a single bin, and high-school diploma but not college into (another) single-bin. Bucketization is performed based on the data format and the semantics of attributes (and thus is privacy-preserving). It is done only for structured learning (Section 3.3); both the input and output data format remain the same.

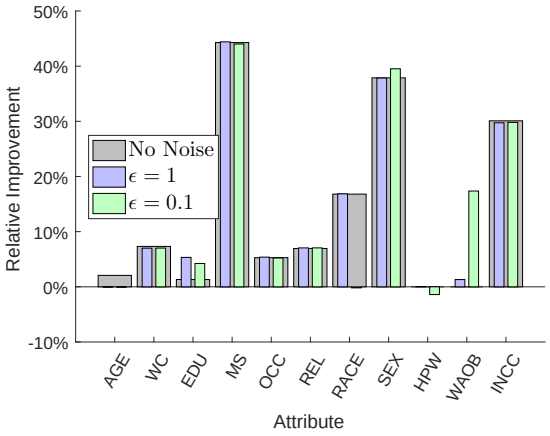

Figure 1: Relative Improvement of Model Accuracy of the un-noised, $\varepsilon = 1$-DP, and $\varepsilon = 0.1$-DP models, with respect to the baseline (marginals). Overall, the improvement for $\varepsilon = 1$ or $\varepsilon = 0.1$ is comparable to that for the un-noised version. Adding noise to achieve DP for structure learning (Section 3.3) can lead to a different acyclic graph of the model. (This is why there is a significant difference in improvement for attributes RACE and WAOB between $\varepsilon = 1$-DP and $\varepsilon = 0.1$-DP.)

5. SYNTHETICS GENERATOR TOOL

The synthetic generator [5] is implemented as C++ tool which takes as input: a dataset represented as a CSV file, a few metadata text files describing the dataset, and a config file. As output, the tool produces a synthetic dataset of the requested size and some metadata.

The generation process is defined by the config file, i.e., parameters defined within in control various aspects of the generation process. The parameters are the privacy parameters k, γ, ε_0, and also parameters of the generative model such as ω. In addition, the tool takes two optional parameters to control the privacy test: `max_plausible` and `max_check_plausible`, which allow the test to terminate early. Specifically, the implementation initially sets $k' = 0$, and iterates over the records of D in a random order, incrementing k' for each plausible seed record d_a encountered. The process terminates whenever $k' \geq$ `max_plausible` or if `max_check_plausible` records have been examined (whichever occurs first). Note that this affects performance (but not privacy); lower values lead to faster generation time in cases where plausible seeds are abundant, at the cost of fewer synthetics passing the test (potentially lowering utility).

The synthesis process, given a chosen seed, is independent of other seeds (Section 2); so the generation process itself is embarrassingly parallel. One hurdle with running multiple concurrent instances is implementing differentially-private parameters learning (Section 3.4.1). In general, the number of configuration (in the sense of Section 3.4) of the model is too large (i.e., exponential in the number of attributes) to learn the model as a pre-processing step. So we design the tool to learn the model for each configuration as it encounters it. To ensure that the privacy guarantee holds we set the RNG seed number to be a deterministic function (i.e., a hash) of the configuration.

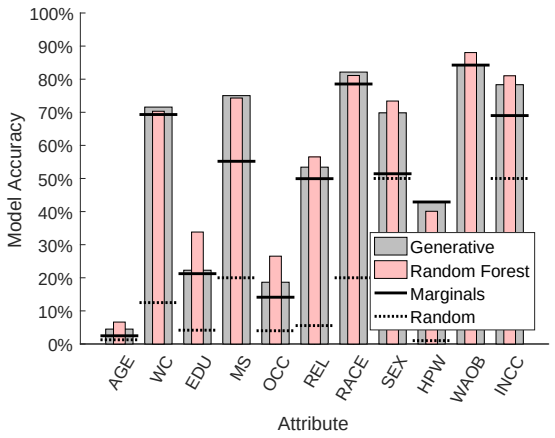

Figure 2: Model Accuracy. The difference between the random forest accuracy and the marginals accuracy indicates how informative the data is about each attribute.

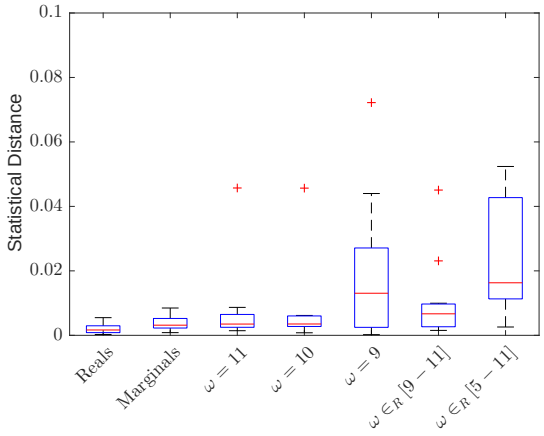

Figure 3: Statistical Distance for individual attributes of two distributions: reals and (other) reals; reals and marginals; reals and synthetics (for varying ω). The smaller the statistical distance the more information is preserved. The distance of reals and $\omega = 11$ and $\omega = 10$ synthetics is similar to that of reals and marginals.

6. EVALUATION

We feed the 2013 ACS dataset (Section 4) as input to our tool and generate millions of synthetic records. We start with a description of the experimental setup. The evaluation itself is divided into four logical parts: (Section 6.2) statistical measures (how good are the synthetics according to well-established statistical metrics); (Section 6.3) machine learning measures (how good are the synthetics for machine learning tasks, specifically classification); (Section 6.4) distinguishing game (how successful is an adversary at distinguishing between a real record and a synthetic one); and (Section 6.5) performance measures (how computationally complex it is to generate synthetics).

6.1 Setup

To achieve differential privacy we sampled the input dataset into disjoint sets of records. Each of $\mathbb{D}_T$ and $\mathbb{D}_P$ contains roughly 280,000 records, whereas $\mathbb{D}_S$ contains roughly 735,000 records (Section 3.5). For differential privacy of the generative model, we set $\varepsilon = 1$ (though we give some results for $\varepsilon = 0.1$) and always set δ to be at most $2^{-30} \approx 10^{-9}$.

We typically compare the quality of our generated synthetics with real records (coming from the input dataset) and privacy-preserving marginals (Section 3.2) which we refer to as *reals* and *marginals*, respectively. The synthetics we generate are referred by their generation parameters (e.g., $\omega = 10$). Unless otherwise stated, we set $k = 50$, $\varepsilon_0 = 1$, $\gamma = 4$, and ω is set to vary between 5 and 11.

We maintain a testing set of roughly 100,000 records. Evaluation of classifiers (in this section) uses at least 100,000 records for training and a (disjoint) testing set of size at least 30% of the size of the aforementioned training set.

6.2 Statistical Measures

We evaluate the quality of the synthetics in terms of their statistical utility, i.e., the extent to which they preserve the statistical properties of the original (input) dataset. We can do this at the level of the generative model (Section 3.1) itself. Concretely, we directly quantify the error of the privacy-preserving generative model before any synthetic record is generated. We do this for each attribute by repeatedly selecting a record from the input dataset (uniformly at random) and using the generative model to find the most likely attribute value (of that attribute) given the other attributes. The generative model error is then measured as the proportion of times that the most likely attribute value is *not* the correct (i.e., original) one. We repeat this procedure millions of times to quantify the average error of the model for each attribute. Because the generative model is made differentially private by adding noise (Section 3.4.1) we additionally repeat the whole procedure 20 times (learning a different private model each time) and take the average.

The results are shown in Figures 1 and 2. Figure 1 shows the relative decrease in model error (i.e., improvement of model accuracy) over the (privacy-preserving) marginals; it shows this improvement for the un-noised, (ε=1)-differential private, and (ε=0.1)-differential private generative models. There is a clear accuracy improvement over marginals, in addition to a low decrease in improvement between the un-noised model and the ε=1 and ε=0.1 noisy versions.

Figure 2 shows the accuracy of the un-noised generative model against the (un-noised) marginals, random guessing (baseline), and the best classifier we could find (trained on as many records as the generative model), the random forest (RF). While RF's accuracy is sometimes higher than that of the generative model, the accuracy of the latter is in many cases significantly higher than that of marginals and random guessing. We conclude that while the proposed generative model does not perform as well as RF (though making RF differentially private would certainly lower its performance) it does perform significantly better than marginals (or random guessing).

In addition to the error of the generative model, we can more directly evaluate the extent to which the generated synthetics preserve the statistical properties of the original (input) dataset. To do this, we compare the probability distributions of the synthetics with the reals and marginals. Specifically, for reals, marginals and synthetics datasets, we compute the distribution of each attribute and of each pair of attributes. We compare each of these distributions to those computed on (other) reals and quantify their distance. We use a well-established statistical distance metric called "the"

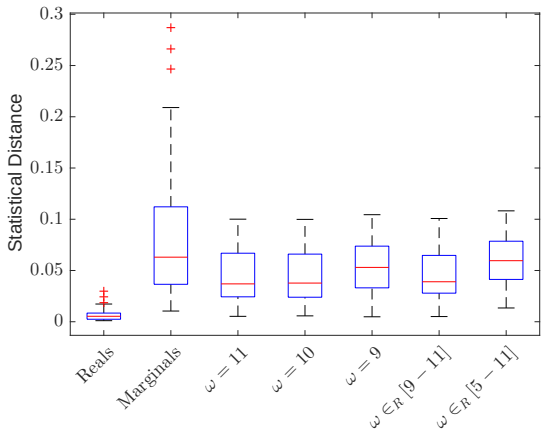

Figure 4: Statistical Distance for pairs of attributes of two distributions: reals and (other) reals; reals and marginals; reals and synthetics (for varying ω). The smaller the statistical distance the more information is preserved. The distance of reals and synthetics is significantly smaller than that of reals and marginals.

statistical distance (a.k.a. total variation distance [17, 30]).

The results are shown in Figures 3 and 4, where Figure 3 shows box-and-whisker plots for the distance of the distributions of each attribute separately, and Figure 4 shows box-and-whisker plots for the distance of the distributions of all pairs of attributes. While marginals do well for single attribute and sometimes outperform our synthetics (though the statistical distance for all datasets is small), synthetics clearly outperform marginals for pairs of attributes. We conclude that the generated synthetics preserve significantly more statistical information than marginals.

6.3 Machine Learning Measures

In addition to preserving statistical properties of the original (input) dataset, the synthetics should also be suitable to various machine learning tasks. In particular, given a learning task, we can evaluate the extent to which synthetics are suitable replacements for a real dataset. For the ACS dataset, a natural and well-establish classification task is to predict a person's income class (i.e., $\geq$ 50K or $<$ 50K) using the other attributes as features (Section 4).

We train various classifiers on the synthetic datasets and on the real (input) dataset. We then compare: the classification accuracy obtained, and the agreement rate of the learned classifiers. Specifically, for two classifiers trained on different datasets (but with the same classification task), we define the *agreement rate* to be the percentage of records for which the two classifiers make the same prediction (regardless of whether the prediction is correct). Given that we look at the agreement rate of classifiers trained on reals and synthetics, the agreement rate reveals the extent to which the classifier trained on synthetic data has learned the *same* model as the classifier trained on real data.

Table 3 shows the obtained results for three (best) classifiers: Classification Tree (Tree), Random Forest (RF), and AdaBoostM1 (Ada). The accuracy and agreement rate are calculated as the average over 5 independent runs, that is, for each run, we use different (randomly sampled) training and testing datasets. Overall, we see that both the accuracy

Table 3: Classifier Comparisons. The agreement rate is the proportion of times that the classifier makes the same prediction as a classifier trained on real data.

	Accuracy			Agreement Rate		
	Tree	RF	Ada	Tree	RF	Ada
Reals	77.8%	80.4%	79.3%	80.2%	86.4%	92.4%
Marginals	57.9%	63.8%	69.2%	58.5%	65.4%	75.6%
$\omega = 11$	72.4%	75.3%	78.0%	73.9%	79.0%	83.0%
$\omega = 10$	72.3%	75.2%	78.1%	73.8%	78.9%	83.6%
$\omega = 9$	72.4%	75.2%	77.5%	73.9%	79.2%	82.4%
$\omega \in_R [9-11]$	72.3%	75.2%	78.1%	73.7%	79.0%	83.9%
$\omega \in_R [5-11]$	72.1%	75.2%	78.1%	73.6%	79.2%	83.3%

and the agreement rates of the synthetics are significantly closer to that of the reals than the marginals are.

In addition to comparing the best classifiers trained on real data versus those trained on synthetic data, we can also compare privacy-preserving classifiers trained on real data versus non-private classifiers trained on (privacy-preserving) synthetic data. In particular, Chaudhuri et al. [10] propose two techniques based on empirical risk minimization to train logistic regression (LR) and support vector machines (SVM) binary classifiers: output perturbation (noise is added to the learned model), and objective perturbation (noise is added to the objective function of the minimization problem). To train such classifiers, we first pre-process our datasets following the instructions in [10]: we transform each categorical attribute into an equivalent set of binary attributes, and normalize features so that each feature takes values in $[0, 1]$ and subsequently further normalize each training example such that its norm is at most 1. The target attribute for classification is again the person's income class. The method proposed in [10] has two parameters: the privacy budget ε which we set to 1 (the same as for our generative model), and λ which is a regularization parameter. We use the code of [10], which we obtain courtesy of the authors, to train the LR and SVM classifiers. Because the classification models vary greatly depending on λ, we vary its value in the set $\{10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}\}$ and (optimistically) pick whichever value maximizes the accuracy of the non-private classification model.

We report the accuracy obtained in each case in Table 4, where we contrast non-private, output perturbation DP, and objective perturbation DP classifiers trained on real data with non-private classifiers trained on our synthetic datasets (for various values of ω). Remark that for the case $\omega = 11$, for example, this is a fair comparison as the obtained LR and SVM classifiers are $\varepsilon = 1$-DP and thus provides the exact same privacy guarantee as the output perturbation and objective perturbation LR and SVM classifiers. Non-private LR and SVM classifiers trained on our (privacy-preserving) synthetic datasets are competitive with differentially private LR and SVM classifiers trained on real data.

We emphasize that the results should be interpreted in favor of our proposed framework. Indeed, the classifiers trained on our privacy-preserving synthetics outperforms ε-DP LR classifier and only achieves about 1% lower accuracy than the objective-perturbation ε-DP SVM. This is significant because the technique to train the ε-DP LR and SVM is specifically optimized for that task. In contrast, our synthetics are *not* specifically generated to optimize any particular classification task; instead the general objective is to preserve the statistical properties of real data.

Table 4: Privacy-Preserving Classifier Comparisons.

	LR	SVM
Non Private	79.9%	78.5%
Output Perturbation	69.7%	76.2%
Objective Perturbation	76.3%	78.2%
Marginals	68.9%	68.9%
$\omega = 11$	77.6%	77.2%
$\omega = 10$	77.7%	77.1%
$\omega = 9$	77.5%	77.1%
$\omega \in_R [9-11]$	77.5%	76.9%
$\omega \in_R [5-11]$	77.7%	77.3%

Table 5: Distinguishing Game. Random Forest (RF) and Classification Tree (Tree) can easily distinguish marginals from reals but perform significantly less well when trying to distinguish synthetics from reals.

	Reals	Marginals	$\omega =$ or $\in_R$				
			11	10	9	$[9-11]$	$[5-11]$
RF	50%	79.8%	62.3%	61.8%	63.0%	60.1%	61.4%
Tree	50%	73.2%	58.9%	58.6%	59.8%	57.9%	58.4%

6.4 Distinguishing Game

A different way to evaluate the quality of synthetic datasets is to quantify the extent to which the synthetics can "pass off" as real records. In other words, we can imagine a game in which the participant is given a random record either from a real dataset or a synthetic dataset (but doesn't know which) and is asked to distinguish between the two possibilities. In this case, the utility is measured by how likely a sophisticated participant (e.g., a well-established learning algorithm) is to make a mistake, i.e., confuse a synthetic record with a real record or vice-versa.

For our purpose the role of the participant is played by the two best classifiers (those that best distinguish synthetics from reals): Random Forest (RF) and Classification Tree (Tree). Specifically, we provide $50,000$ records from both a real dataset and a synthetic dataset (i.e., $100,000$ total) as training examples to the (binary) classifier. We then evaluate the accuracy on a 50% mix of real and synthetic records which were not part the training set. Table 5 shows the results: both classifiers obtain reasonably high (79.8% and 73.2%) accuracy in distinguishing marginals from real records. However, both classifiers obtain much lower accuracy (i.e., 63%) when trying to distinguish synthetics from reals.

6.5 Performance Measures

In addition to how much utility they preserve, synthetics also need to be easy to generate. The generation is a parallel process, so we measure the time taken for both the learning of the privacy-preserving generative model (model learning) and the synthetics generation (synthesis) itself. Figure 5 shows the time taken to produce various number of synthetics records (totaling over 1 million). The parameters `max_plausible` and `max_check_plausible` (Section 5) were set to 100 and 50000 respectively. The machine used for the experiment runs Scientific Linux and is equipped with an Intel Xeon E5-2670 processor (2.60GHz) with 32 processing units and 128GB of RAM. We ran 96 instances (16 in parallel at a time) and picked a random maximum runtime for each instance between 3 and 15 minutes.

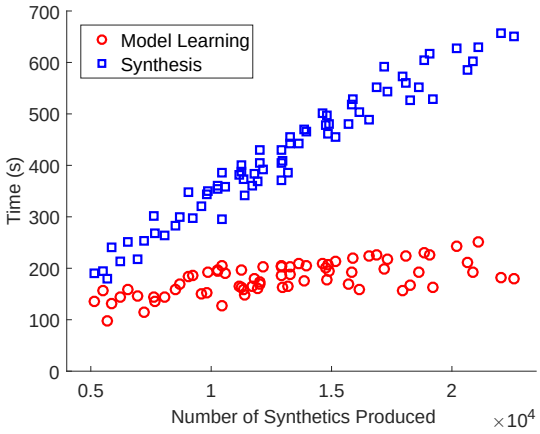

Figure 5: Synthetic Generation Performance. The parameters are: $\omega = 9$, $k = 50$, $\gamma = 4$. The time to generate $10,000$ synthetic records on a single-core is less than 10 minutes. Thus, in the same time frame we can generate 1 million records with 100 parallel instances.

The generator outputs all synthetics produced regardless of whether they pass the privacy test. Naturally, only those which pass the test should to be released. Thus, the extent to which we can synthesize large (privacy-preserving) datasets depends on how easy it is to find synthetics that pass the privacy-test (Section 2). To evaluate this, we set $\gamma = 2$ and `max_check_plausible` $= 100,000$, and vary k and ω. We measure the proportion of synthetics which pass the privacy test. The results are shown in Figure 6: even for stringent privacy parameters (e.g., $k = 100$) a significant proportion (i.e., over 50% for $\omega \in_R [5-11]$) of synthetics pass the test.

7. RELATED WORK

Data synthesis is the process of creating artificial data that can be used to serve some key purposes of real data. For example, it can be used to test software [50, 41] when there is an inadequate supply of real tests available from operations. It can be used to evaluate the performance of database systems [19] when obtaining real data at sufficient scale is difficult or expensive. It can also be used to protect privacy. In 1993, Rubin [44] proposed the idea of creating synthetic data based on multiple imputation, that is, on repeated use of a function that proposes values for missing fields in a record. The generative model we presented in Section 3.1 uses a similar technique. This and related work have given rise to a substantial body of research on the release of synthetic data [25, 13]. Such techniques have achieved significant deployment; for example, they have been adopted by the U.S. Census Bureau [21, 28].

An alternative to data synthesis sometimes called *syntactic* privacy protection transforms the sensitive data using a combination of aggregation, suppression, and generalization, to achieve criteria such as k-anonymity [45] or l-diversity [34]. Although these techniques support privacy protected data publishing without synthesis, the degree of privacy protection they provide depends on the background knowledge of adversaries. The key difference between (k, γ)-plausible deniability and k-anonymity is that the latter is a

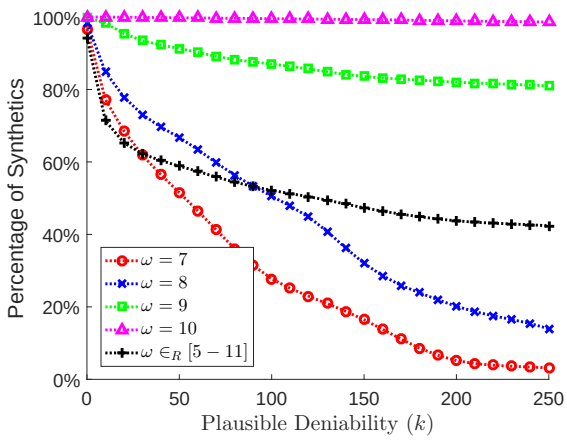

Figure 6: Percentage of candidates which pass the privacy test for various values of k and ω ($\gamma = 2$). The percentage decreases for higher privacy (i.e., larger k) but remains significant even for combinations of parameters yielding high privacy. The conclusion is that (very) large synthetic datasets can efficiently be generated.

syntactic condition on the *output* of a *sanitization process*, whereas plausible deniability is a condition on a *synthetic generator mechanism* with respect to its *input* data.

Plausible deniability as a privacy notion for synthetic data was proposed by Bindschaedler and Shokri in [3] which describes a technique to synthesize location trajectories in a privacy-preserving way. The use of plausible deniability in [3] is specific to location privacy as it is defined in terms of a semantic distance between two location trajectories. In contrast, this work generalizes the notion of plausible deniability for *general* data synthesis by establishing it as a privacy criterion in terms of the underlying synthesis probabilities. Consequently, this criterion is applicable to any system and any (kind of) data. The generative framework described in this paper enables us to formally connect plausible deniability to differential privacy (Theorem 1).

Differential privacy provides guarantees even against adversaries with (almost) unlimited background knowledge. However, popular differentially private mechanisms such as the Laplacian mechanism target the release of aggregate statistics. By contrast, we focus on synthesizing data with the *same format* as the (sensitive) input data. Preserving the data format is valuable for many reasons, such as enabling the use of applications and code that are targeted at raw or sanitized data. There is a line of work on mechanisms that are differentially private and provide data as an output. Some of these techniques have theoretical properties that may make them impractical in important cases [7].

A prominent example is the Exponential Mechanism [37]. Informally, the mechanism induces a distribution on the space of output records by assigning a weight to each such record and then producing output records by sampling from that distribution. The mechanism is of great importance for algorithm design due to its generality. However, as several researchers have pointed out [12, 29, 6], a direct application is too costly to be practical for high-dimensional datasets due to the complexity of sampling, which grows exponentially in the dimension of the data records. Concretely, a straightforward implementation of the exponential mechanism to generate synthetic records from the ACS dataset (Section 4) would sample from a universe of records of size roughly 2^{39} (Table 2). This would require pre-computing that many probabilities. If we assume we can store each value in four bytes this would require about 2TB of memory. In contrast, the complexity of synthesizing a record with our framework depends only on the number of records in the dataset and on the complexity of our generative model and thus the process is very efficient in practice (Section 6.5).

There is a growing collection of mechanisms and case studies for differentially private release of data [1, 9, 36, 33, 49, 11, 23], although some of these are based on a broad view of data release, such as the release of histograms or contingency tables. Our use of plausible deniability to achieve differentially private data adds to this body of work. The typical approach to protect privacy in this context is to add noise directly to the generative model. For example, this is the approach taken by [31, 8, 32, 52]. In particular, [52] constructs a generative model based on Bayesian networks similar to the generic generative model of Section 3.

Our work takes a novel approach: instead of trying to achieve differential privacy directly, we design the generative framework to achieve plausible deniability. A major step towards achieving plausible deniability and a key novelty is the idea of *testing* privacy. That is, instead of designing the mechanism so it achieves that notion by design, we use a privacy test which rejects "bad" samples. As a side effect, the generative model is decoupled from the framework. Privacy is guaranteed in a way that is oblivious to the specifics of the generative model used for synthesis. Furthermore, the guarantee offered by our proposed (k, γ)-plausibly deniable mechanism is surprisingly close to that of differential privacy, as evidenced by the fact that merely randomizing the threshold yields a differentially private mechanism.

The idea of running data synthesis and then testing privacy has been used before. For example, Reiter et al. in [42] and [43] use inference to evaluate privacy risk for a synthetic data release. However, there is no proof of privacy, and it is not efficient to run a set of inference attacks to estimate the risk before releasing a dataset.

8. DISCUSSION

Regardless of whether one intends to release a set of aggregate statistics or a synthetic dataset, there is no privacy protection technique that can preserve utility for all meaningful utility functions. However, one key feature of our generative framework is that, unlike other approaches based on differential privacy, any generative model $\mathcal{M}$ can be used while keeping the same privacy guarantees. As a result, designing $\mathcal{M}$ is a data science problem which need not involve considerations of the privacy of the seeds.

Parameter ω (Section 3) controls the closeness of synthetics to their seeds. (Lower ω means more dependence on the seed but it is harder to pass the privacy test.) A pragmatic approach is to generate synthetics for various values of ω and then randomly sample a subset of those synthetics which pass the privacy test (this is evaluated in Section 6). Note that no matter what the value of ω is, the privacy of the seeds is ensured because of the privacy test.

In the special case where the generative model $\mathcal{M}$ is independent of the seed, the privacy guarantee applies to any output from Mechanism 1 because the privacy test always passes. However, for a seed-dependent generative model,

the privacy of the seeds is safeguarded by rejecting synthetics which do not pass the privacy test. So, when generating several synthetics using the same input dataset, the privacy obtained depends on the number of synthetics released. In fact, when Privacy Test 2 is used, the (ε, δ)-differential privacy guarantee applies to a single (released) synthetic record y. That said, the composition theorems for differential privacy can be used to extend the guarantee to arbitrarily large synthetic datasets, provided the privacy budget is appropriately increased. We leave as future work the design of improved composition strategies.

9. CONCLUSIONS

We have presented the first practical data synthesis tool with strong privacy guarantees. We have formalized plausible deniability for generating generic data records, and have proven that our mechanisms can achieve differential privacy without significantly sacrificing data utility.

Acknowledgments. This work was supported in part by NSF CNS grants 13-30491 and 14-08944. The views expressed are those of the authors only.

10. REFERENCES

[1] J. M. Abowd and L. Vilhuber. How protective are synthetic data? In *ICPSD*, 2008.

[2] M. Barbaro and T. Z. Jr. A face is exposed for aol searcher no. 4417749, 2006.

[3] V. Bindschaedler and R. Shokri. Synthesizing plausible privacy-preserving location traces. In *IEEE S&P*, 2016.

[4] V. Bindschaedler, R. Shokri, and C. Gunter. Plausible Deniability for Privacy-Preserving Data Synthesis (Extended Version). *arXiv*, 2016.

[5] V. Bindschaedler, R. Shokri, and C. Gunter. Synthetics Generation Tool. `https://vbinds.ch/projects/sgf`, 2016.

[6] J. Blocki, A. Datta, and J. Bonneau. Differentially private password frequency lists. In *NDSS*, 2016.

[7] A. Blum, K. Ligett, and A. Roth. A learning theory approach to noninteractive database privacy. *JACM*, 2013.

[8] C. M. Bowen and F. Liu. Differentially private data synthesis methods. *arXiv preprint*, 2016.

[9] A.-S. Charest. How can we analyze differentially-private synthetic datasets? *JPC*, 2011.

[10] K. Chaudhuri, C. Monteleoni, and A. D. Sarwate. Differentially private empirical risk minimization. *JMLR*, 2011.

[11] R. Chen, N. Mohammed, B. C. Fung, B. C. Desai, and L. Xiong. Publishing set-valued data via differential privacy. *PVLDB*, 2011.

[12] G. Cormode, M. Procopiuc, D. Srivastava, and T. Tran. Differentially private publication of sparse data. In *ICDT*, 2012.

[13] J. Drechsler. *Synthetic datasets for statistical disclosure control: theory and implementation*. Springer Science & Business Media, 2011.

[14] C. Dwork. Differential privacy: A survey of results. In *TAMC*, 2008.

[15] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *TCC*, 2006.

[16] C. Dwork and A. Roth. The algorithmic foundations of differential privacy. *FnT-TCS*, 2014.

[17] A. L. Gibbs and F. E. Su. On choosing and bounding probability metrics. *ISR*, 2002.

[18] P. Golle and K. Partridge. On the anonymity of home/work location pairs. In *PerCom*, 2009.

[19] J. Gray, P. Sundaresan, S. Englert, K. Baclawski, and P. J. Weinberger. Quickly generating billion-record synthetic databases. In *ACM SIGMOD*, 1994.

[20] M. A. Hall. *Correlation-based feature selection for machine learning*. PhD thesis, The University of Waikato, 1999.

[21] S. Hawala. Producing partially synthetic data to avoid disclosure. In *PJSM*, 2008.

[22] M. Humbert, E. Ayday, J.-P. Hubaux, and A. Telenti. Addressing the concerns of the lacks family: quantification of kin genomic privacy. In *ACM CCS*, 2013.

[23] G. Jagannathan and R. N. Wright. Privacy-preserving imputation of missing data. *DKE*, 2008.

[24] T. Julian and R. Kominski. Education and synthetic work-life earnings estimates. american community survey reports. acs-14. *US Census Bureau*, 2011.

[25] S. A. Keller, S. Shipp, and A. Schroeder. Does big data change the privacy landscape? a review of the issues. *Annual Review of Statistics and Its Application*, 2016.

[26] D. Kifer and A. Machanavajjhala. No free lunch in data privacy. In *ACM SIGMOD*, 2011.

[27] S. K. Kinney, J. P. Reiter, and J. Miranda. Synlbd 2.0: improving the synthetic longitudinal business database. *Statistical Journal of the IAOS*, 2014.

[28] S. K. Kinney, J. P. Reiter, A. P. Reznek, J. Miranda, R. S. Jarmin, and J. M. Abowd. Towards unrestricted public use business microdata: The synthetic longitudinal business database. *ISR*, 2011.

[29] E. Lantz, K. Boyd, and D. Page. Subsampled exponential mechanism: Differential privacy in large output spaces. In *AISec*, 2015.

[30] D. A. Levin, Y. Peres, and E. L. Wilmer. *Markov chains and mixing times*. American Mathematical Soc., 2009.

[31] H. Li, L. Xiong, and X. Jiang. Differentially private synthesization of multi-dimensional data using copula functions. In *EDBT*, 2014.

[32] F. Liu. Model-based differential private data synthesis. *arXiv*, 2016.

[33] A. Machanavajjhala, D. Kifer, J. Abowd, J. Gehrke, and L. Vilhuber. Privacy: Theory meets practice on the map. In *ICDE*, 2008.

[34] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam. l-diversity: Privacy beyond k-anonymity. *TKDD*, 2007.

[35] D. Margaritis. *Learning Bayesian network model structure from data*. PhD thesis, US Army, 2003.

[36] D. McClure and J. P. Reiter. Differential privacy and statistical disclosure risk measures: An investigation with binary synthetic data. *TDP*, 2012.

[37] F. McSherry and K. Talwar. Mechanism design via differential privacy. In *FOCS*, 2007.

[38] M. T. Mora, D. J. Villa, and A. Dávila. Language maintenance among the children of immigrants: A comparison of border states with other regions of the us. *Southwest Journal of Linguistics*, 2005.

[39] A. Narayanan and V. Shmatikov. Robust de-anonymization of large sparse datasets. In *IEEE S&P*, 2008.

[40] A. Narayanan and V. Shmatikov. De-anonymizing social networks. In *IEEE S&P*, 2009.

[41] K. H. Pedersen, K. Torp, and R. Wind. Simple and realistic data generation. In *PVLDB*, 2006.

[42] J. P. Reiter and R. Mitra. Estimating risks of identification disclosure in partially synthetic data. *JPC*, 2009.

[43] J. P. Reiter, Q. Wang, and B. Zhang. Bayesian estimation of disclosure risks for multiply imputed, synthetic data. *JPC*, 2014.

[44] D. B. Rubin. Statistical disclosure limitation. *Journal of official Statistics*, 1993.

[45] P. Samarati and L. Sweeney. Generalizing data to provide anonymity when disclosing information. In *PODS*, 1998.

[46] A. Tockar. Riding with the stars: Passenger privacy in the nyc taxicab dataset. `http://research.neustar.biz/2014/09/15/riding-with-the-stars-passenger-privacy-in-the-nyc-taxicab-dataset/`, 2014.

[47] US Census Bureau. American community survey (acs). `http://www.census.gov/programs-surveys/acs/`.

[48] R. Wang, Y. F. Li, X. Wang, H. Tang, and X. Zhou. Learning your identity and disease from research papers: information leaks in genome wide association study. In *ACM CCS*, 2009.

[49] L. Wasserman and S. Zhou. A statistical framework for differential privacy. *JASA*, 2010.

[50] M. A. Whiting, J. Haack, and C. Varley. Creating realistic, scenario-based synthetic data for test and evaluation of information analytics software. In *BELIV*, 2008.

[51] J. Xu, Z. Zhang, X. Xiao, Y. Yang, G. Yu, and M. Winslett. Differentially private histogram publication. *VLDB*, 2013.

[52] J. Zhang, G. Cormode, C. M. Procopiuc, D. Srivastava, and X. Xiao. Privbayes: Private data release via bayesian networks. In *ACM SIGMOD*, 2014.