PrivGene: Differentially Private Model Fitting Using Genetic Algorithms

Jun Zhang[1] Xiaokui Xiao[1] Yin Yang[2,3] Zhenjie Zhang[2] Marianne Winslett[2,3]

[1]School of Computer Engineering
Nanyang Technological University
{jzhang027, xkxiao}@ntu.edu.sg

[2]Advanced Digital Sciences Center
Illinois at Singapore Pte. Ltd.
{yin.yang, zhenjie}@adsc.com.sg

[3]Department of Computer Science
University of Illinois at Urbana-Champaign
winslett@illinois.edu

ABSTRACT

ε-differential privacy is rapidly emerging as the state-of-the-art scheme for protecting individuals' privacy in published analysis results over sensitive data. The main idea is to perform random perturbations on the analysis results, such that any individual's presence in the data has negligible impact on the randomized results. This paper focuses on analysis tasks that involve model fitting, i.e., finding the parameters of a statistical model that best fit the dataset. For such tasks, the quality of the differentially private results depends upon both the effectiveness of the model fitting algorithm, and the amount of perturbations required to satisfy the privacy guarantees. Most previous studies start from a state-of-the-art, non-private model fitting algorithm, and develop a differentially private version. Unfortunately, many model fitting algorithms require intensive perturbations to satisfy ε-differential privacy, leading to poor overall result quality.

Motivated by this, we propose *PrivGene*, a general-purpose differentially private model fitting solution based on *genetic algorithms* (GA). PrivGene needs significantly less perturbations than previous methods, and it achieves higher overall result quality, even for model fitting tasks where GA is not the first choice without privacy considerations. Further, PrivGene performs the random perturbations using a novel technique called the *enhanced exponential mechanism*, which improves over the exponential mechanism [26] by exploiting the special properties of model fitting tasks. As case studies, we apply PrivGene to three common analysis tasks involving model fitting: logistic regression, SVM classification, and k-means clustering. Extensive experiments using real data confirm the high result quality of PrivGene, and its superiority over existing methods.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database applications—*statistical databases*

Keywords

Differential privacy, genetic algorithms, model fitting

1. INTRODUCTION

Releasing sensitive information while preserving individuals' privacy has been an active research subject for decades. The recently proposed notion of ε-differential privacy [9, 10] is rapidly emerging as the state-of-the-art scheme for this purpose, due to its strong privacy guarantees, and robustness against adversaries with background knowledge. ε-differential privacy publishes randomly perturbed analysis results performed on a sensitive dataset, rather than the dataset itself. The randomized results must ensure that it is hard for the adversary to infer any attribute value of any individual record in the dataset, even if the adversary knows the exact details of all remaining records.

This paper focuses on enforcing ε-differential privacy on analysis tasks involving *model fitting*. Given a statistical model M, a sensitive database D, and a *fitting function* $f(D, \omega)$ that measures how well M with parameter vector ω fits dataset D, model fitting aims to find the best parameter vector ω^* for M that maximizes $f(D, \omega^*)$. Model fitting is used in a broad class of analysis tasks, including classification (which searches for the best way to distinguish different classes of objects), regression (which finds a mathematical model that describes the data best), and clustering (which computes the optimal assignment of objects to groups). As a real world scenario, consider an electronic medical records database, where it is highly beneficial to publish statistical analysis results, provided that individuals' information is kept private. Since classification, regression, and clustering are common tools [11, 22, 28] for analyzing medical data, privacy-preserving methods for such analysis are highly valuable.

As we review in Section 2.2, existing studies on differentially private model fitting mostly develop the private version of a state-of-the-art, non-private model fitting algorithm. This methodology incurs two serious drawbacks. First, each such method is limited to a narrow range of applications that use a specific type of model fitting algorithms. Second and more importantly, *the differentially private version of a good non-private algorithm does not necessarily yield high-quality results*, because such an algorithm may need large amounts of random perturbations to satisfy ε-differential privacy, which dominates overall result inaccuracy. Many existing model fitting algorithms suffer from this problem due to their inherent complexity, which we elaborate further in Section 2.2. Consequently, the result quality of their differentially private versions

SIGMOD'13, June 22–27, 2013, New York, New York, USA.

often lags far behind the non-private versions. Finally, although generic solutions exist that can in theory render any analysis results differentially private, the quality of their results tends to be rather poor, if usable at all.

Motivated by this, we propose *PrivGene*, a novel framework for differentially private model fitting based on genetic algorithms (GA) [14, 16]. Similar to conventional GA, PrivGene starts with a set of seed parameter vectors, and iteratively improves them by emulating natural evolutions. In each iteration, PrivGene recombines and modifies existing parameter vectors through *crossover* and *mutation* operations, and subsequently filters them through *selection* of the top ones that best fit the given dataset. Among these operations, crossover and mutation are performed independently of the sensitive data, and, thus, do not have privacy issues at all. Only the selection step involves the data, and requires random perturbations to satisfy ε-differential privacy. PrivGene performs differentially private selections with high accuracy through a novel *enhanced exponential mechanism (EEM)*, which improves over the exponential mechanism [26] by exploiting the special properties of the selection operations.

We define a broad class of model fitting tasks that can be solved by PrivGene, and demonstrate its use on three common analysis tools, namely *logistic regression, SVM classification* and *k-means clustering*. Although GA is usually not the first choice to solve these problems in the non-private setting, PrivGene outperforms all existing private solutions in terms of overall result quality, due to the former's low accuracy loss during perturbations. Extensive experiments using 6 real datasets confirm the high accuracy achieved by PrivGene and its superiority over previous methods.

The contributions of this paper are summarized as follows: (i) We propose PrivGene, a novel framework for model fitting under ε-differential privacy, which applies to a broad class of analysis tasks. (ii) We design EEM, a novel mechanism for differentially private parameter vector selection for PrivGene. EEM significantly improves over the exponential mechanism by utilizing the special properties of PrivGene. (iii) We apply PrivGene to three common analysis tasks: logistic regression, SVM classification and k-means clustering. (iv) We demonstrate through extensive experiments that PrivGene significantly outperform existing algorithms for enforcing differential privacy on these types of analyses.

2. BACKGROUND

In this section, we introduce the basic concepts of differential privacy in Section 2.1, and then present an overview of existing solutions for model fitting under differential privacy.

2.1 Differential Privacy

As stated in Section 1, ε-differential privacy guarantees the hardness for inferring any attribute value of any record in the dataset, provided that the adversary knows all attribute values of all remaining records. This hardness is controlled by the parameter ε called the *privacy budget*. Lower privacy budget indicates stronger privacy protection, but also noisier results. Formally, let D be any dataset in the application domain, and D' be its *neighbor database* obtained by replacing a record of D with a new one. Meanwhile, let $\mathcal{A}$ be a randomized algorithm, and O be any possible output of $\mathcal{A}$. Then, $\mathcal{A}$ satisfies ε-differential privacy, if and only if the following inequality holds [10].

$$\Pr\left[\mathcal{A}(D) = O\right] \le e^{\varepsilon} \cdot \Pr\left[\mathcal{A}(D') = O\right]. \qquad (1)$$

According to the above inequality, a deterministic algorithm cannot possibly satisfy ε-differential privacy for any value of ε. Hence, a deterministic analysis algorithm must be randomly perturbed to satisfy the privacy requirement. Two fundamental approaches for this purpose are the Laplace mechanism [10] and the exponential mechanism [26]. The former is limited to analysis tasks that return numeric results, whereas the latter mainly targets tasks with categorical outputs. For simplicity, we present these mechanisms for analysis tasks with a single output – multiple outputs can be handled by utilizing the *composability* property of differential privacy [9]. In particular, composability indicates that when a set of (say, m) random algorithms satisfy differential privacy with parameters $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_m$, respectively, the set of algorithms as a whole satisfies $\left(\sum_i \varepsilon_i\right)$-differential privacy.

Given an analysis task with a numeric output $\mathcal{F}$ and a privacy budget ε, the Laplace mechanism injects into $\mathcal{F}$ random Laplace noise [9] of scale $\Delta_{\mathcal{F}}/\varepsilon$, where $\Delta_{\mathcal{F}}$ is called the *sensitivity* of $\mathcal{F}$. In particular, $\Delta_{\mathcal{F}}$ is defined as:

$$\Delta_{\mathcal{F}} = \max_{D,D'} \mathcal{F}(D) - \mathcal{F}(D'), \qquad (2)$$

where D, D' are two arbitrary neighbor databases.

For an analysis task with a categorical output (e.g., an ID), injecting random noise no longer yields meaning results. The exponential mechanism tackles this problem by performing random perturbations during the selection of the output. Specifically, given the set Ω of all possible output values, the user assigns each possible output value $\omega \in \Omega$ a *quality score* $\mathcal{Q}(\omega)$; higher scores correspond to better values. Then, the exponential mechanism computes the probability for selecting each possible output value in the domain, and selects one randomly based on these probabilities. In particular, the probability for choosing a value ϖ satisfies:

$$\Pr\left[\varpi \text{ is selected}\right] \propto \exp\left(\frac{\varepsilon \cdot \mathcal{Q}(\varpi)}{2\max_{\omega\in\Omega}\Delta_{\mathcal{Q}(\omega)}}\right), \qquad (3)$$

where $\Delta_{\mathcal{Q}(\omega)}$ is the sensitivity of $\mathcal{Q}(\omega)$ as a function of the input database, defined similarly to Equation 2, i.e., the maximum possible difference of $\mathcal{Q}(\omega)$ for any two neighbor databases. Note that the exponential mechanism does not make any assumptions on the quality function $\mathcal{Q}(\omega)$. However, as we show later in Section 4, the effectiveness of the exponential mechanism can be significantly improved, by exploiting special properties of $\mathcal{Q}(\omega)$.

Based on the Laplace mechanism and the exponential mechanism, a plethora of differentially private methods have been proposed for various analysis tasks. A large number of existing solutions focus on simple aggregate queries such as COUNT and SUM. For instance, Xiao et al. [32] investigate the problem of range-count answering under differential privacy, and propose *Privlet*, an effective synopsis based on the Laplace mechanism for answering arbitrary range-counts with a fixed privacy budget. Hay et al. [15] also study count queries, and propose a post-processing method for improving the accuracy of differentially private synopsis, based on consistency conditions of the application domain. Cormode et al. [4] design differentially private data structures for accurately answering multi-dimensional range-count queries. Li et al [23] and Yuan et al. [34] propose optimized solutions for answering a batch of linear counting queries under differential privacy. Besides simple counting queries, there is also work on publishing data structures consisting of multiple counts. For example, Xu et al. [33] propose effective methods for publishing differentially private histograms. Ding et al. [8] publish data cubes with both privacy and consistency guarantees. None of these methods, however, applies to our problem, as model fitting involves a complex optimization solving process, which is inherently more challenging than COUNT/SUM queries and their derivatives.

Differential privacy has also been applied to complex data min-

Table 1: List of common notations

Notation	Description
D, n	Sensitive database and its cardinality
$\mathcal{T}$	Domain of a tuple in D
ω, d	Parameter vector for a model, and its dimensionality
$f(D, \omega)$	Fitting function that quantifies how well a model, parameterized by ω, fits D
$q(t, \omega)$	Tuple fitting function that describes the quality of a model, parameterized by ω, fits tuple t
Ω, m	Candidate set of parameter vectors, and its cardinality
Ω', m'	Selected set of parameter vectors, and its cardinality
r	Number of iterations in PrivGene

ing tasks. For instance, Li et al. [24] study privacy-preserving frequent item mining. Friedman et al. [13] investigate private decision trees. Inan et al. [18] design solutions for record matching under differential privacy. These problems are orthogonal to ours. Finally, Kifer and Machanavajjhala point out the limitations of differential privacy [19], and explore alternative privacy definitions [20]. This work focuses on the original definition of differential privacy.

2.2 Differentially Private Model Fitting

A large class of analysis tasks, including regression, classification, and clustering, require model fitting to determine the best parameter set. Existing approaches to differentially private model fitting generally follow the methodology of developing the differentially private version of a commonly used algorithm in the non-private setting. The main challenge faced by this methodology is that the sensitivity of such algorithms (and, consequently, the scale of perturbations) is usually prohibitively high, to the extent that direct use of the Laplace mechanism or the exponential mechanism simply returns noise. For instance, Zhang et al. [35] investigate linear and logistic regressions under differential privacy. In the non-private case, the optimal parameter set of linear regression can be solved trivially, with a small number of matrix operations [35]. Yet, as shown in [35], even this simple solver incurs prohibitively high sensitivity. Popular model fitting algorithms for logistic regression, such as iteratively re-weighted least squares [17], are much more complex, and incur prohibitive high sensitivity, too.

The solution proposed in [35], called functional mechanism (FM), injects random noise into the fitting function f, which is the only part of the model fitting task that involves the sensitive data. The model fitting task with the noisy f is then published, and subsequently solved by a standard algorithm. The idea is that when the noisy f is close to the original one, hopefully (but there is no guarantee) the former leads to parameters of comparable quality as the latter. The fitting function of linear regression is simple, and applying FM to it is relatively straightforward. The fitting function of logistic regression, however, still incurs prohibitively high sensitivity. [35] tackles this problem by applying FM to a truncated version of the fitting function consisting of the first few terms of its Taylor expansion. This approach imposes considerable information loss. Further, it is limited to fitting functions with a closed-form Tayler expansion. As we show in Section 5.2 and 5.3, the fitting functions of SVM classification and k-means clustering cannot be handled this way as neither of them is differentiable.

Similar to FM, Chaudhuri et al. [3] solves a restricted class of empirical risk minimization problems under differential privacy, by injecting noise into the fitting function f. The methods in [3] relies on rather strong assumptions about f, e.g., f must be strongly concave and doubly differentiable. [3] demonstrates two specific applications of their methods: logistic regression with a non-zero regularization term (logistic regression with zero regularization, however, fails the strongly concave condition), and SVM classification with certain special loss functions such as *Huber loss* (the more commonly used *hinge loss* function is not differentiable [29]). Even for these two applications, however, the effectiveness of their methods is relatively unstable, and highly sensitive to the choice of the regularization factor. The parameter tuning algorithm in [3] for selecting an appropriate regularization factor consumes a considerable portion of the privacy budget, reducing the overall accuracy of their methods.

Rubinstein et al. [29] investigate differentially private kernel SVMs. Their main focus lies in tackling different types of kernels, including those that involves infinite dimensionality. However, the solutions in [29] still assume a standard SVM solver, which incurs high sensitivity and large amounts of noise, as we show in the experiments. Finally, Kifer et al. [21] improve both the accuracy and the applicability of [3], by using a relaxed privacy definition. This is orthogonal to our work since we focus on the stronger ε-differential privacy definition.

GUPT [27] is a general-purpose differentially private analysis system based on the sample-and-aggregate framework [31], which applies to all analysis tasks whose results are not affected by the number of records in the dataset (e.g., AVG is supported, but COUNT and SUM are not). The main idea is to partition the dataset into smaller blocks, and run the analysis algorithm on each of the blocks *without privacy considerations*. Then, GUPT computes a differentially private average on the results from different blocks. The main strength of GUPT is its general applicability and ease of use. These come at a price of low result quality, however, as the differentially private averaging step can still incur high sensitivity for larger output domains. Overall, all existing solutions are limited to the implicit assumption that *a model fitting task should be solved using the same algorithm as in the non-private case*. PrivGene, presented next, lifts this restriction, and achieves high result quality for a broad class of model fitting tasks.

3. PRIVGENE

Let $D \in \mathcal{T}^n$ be a sensitive database containing n tuples sampled from domain $\mathcal{T}$, and $f(D, \omega)$ be the fitting function that measures how well a parameter vector ω fits the data D. Our goal is to find the best parameter vector ω^* that maximizes $f(D, \omega^*)$. Unlike many existing solutions reviewed in Section 2, PrivGene does not rely on any restrictive assumptions on f, except that the sensitivity of f should be bounded and reasonably small. Table 1 summarizes frequent notations used throughout the paper.

Algorithm 1 shows the general framework of PrivGene, which involves three main inputs: D (the sensitive database), f (the fitting function), ε (the privacy budget), as well as three more system parameters: m (the size of the candidate set), m' (the size of the selected set), and r (the number of iterations). The choice of m, m', and r is discussed towards the end of this section. PrivGene initializes the candidate set Ω with m random parameter vectors (Line 1 in Algorithm 1), and refines them with r iterations. In each iteration (Lines 3-9), PrivGene chooses m' best parameter vectors from Ω in a differentially private manner, and places them into the selected set Ω' (Line 3). Then, the algorithm generates new parameter vectors by performing *crossover* and *mutation* operations over existing ones in Ω', and forms a new candidate set with the new vectors (Lines 6-9). Finally, after the last iteration, the best parame-

Algorithm 1 PrivGene ($D, f, \varepsilon, m, m', r$): returns ω

```
Input: $D$, $f$: sensitive dataset and its fitting function
    $\varepsilon$: privacy budget
    $m$, $m'$: sizes of candidate set $\Omega$ and selected set $\Omega'$, respectively
    $r$: number of iterations
Output: $\omega$: best parameter vector identified by PrivGene
 1: Initialize candidate set $\Omega$ with $m$ randomly generated vectors
 2: for $i = 1$ to $r - 1$ do
 3:     Compute $\Omega' = DP\_Select(D, f, \Omega, m', \varepsilon/r)$
 4:     Set new candidate set $\Omega$ to empty
 5:     for $j = 1$ to $m/2$ do
 6:         Randomly choose two vectors $\omega^1, \omega^2 \in \Omega'$
 7:         Compute $(v^1, v^2) = Crossover(\omega^1, \omega^2)$
 8:         Call $Mutate(v^1)$ and $Mutate(v^2)$
 9:         Add $v^1, v^2$ to $\Omega$
10:     end for
11: end for
12: Compute $\{\omega\} = DP\_Select(D, f, \Omega, 1, \varepsilon/r)$
13: return $\omega$
```

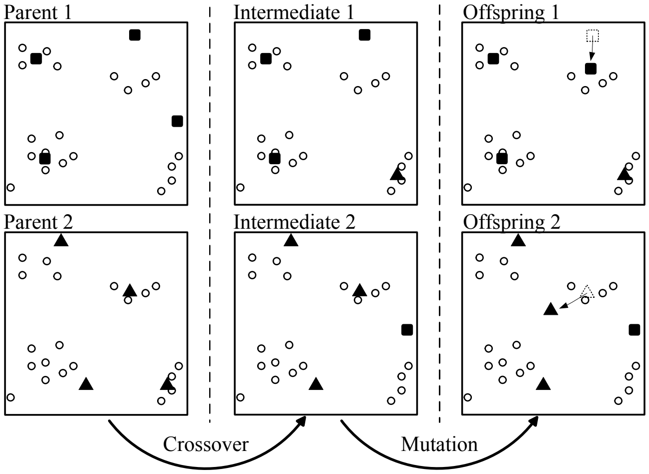


Figure 1: An example of crossover and mutation on 4-means clustering.

ter vector is selected (again with perturbations to satisfy differential privacy), and returned as the final result (Lines 10-11).

There are three major components in PrivGene: $Crossover$, $Mutate$, and DP_Select. Crossover and Mutate operate entirely on the parameter vectors, and do not involve the sensitive data at all. Hence, algorithms in the traditional, non-private setting apply to them. In the following we describe Crossover and Mutate in our specific implementation; we emphasize that details of these two functions are not the main contribution of this paper, and PrivGene works with any algorithms for them as long as they do not involve the dataset D. In particular, our realization of Crossover takes as input two *parent vectors* ω^1 and ω^2, and recombines their elements to obtain two new vectors v^1 and v^2. Let d be the dimensionality of a parameter vector. We perform the recombination by (i) choosing a random number $d' < d$, and subsequently (ii) computing $v^1 = (\omega^1_1, \ldots, \omega^1_{d'}, \omega^2_{d'+1}, \ldots, \omega^2_d)$, and $v^2 = (\omega^2_1, \ldots, \omega^2_{d'}, \omega^1_{d'+1}, \ldots, \omega^1_d)$, where ω_i means the i-th value in vector ω. The Mutate operation on a parameter vector ω is implemented by (i) choosing a random $d' < d$, and (ii) adding random noise to $\omega_{d'}$. In our implementation, the noise's absolute value equals 5% of the domain of $\omega_{d'}$ in the first iteration of PrivGene, and decreases by 5% after each iteration; the sign of the noise is also random. The parameter vectors after the Crossover and Mutate operations are called the *offsprings*.

Algorithm 2 DP_Select ($D, f, \Omega, m', \varepsilon_s$): returns Ω'

```
Input: $D$, $f$: sensitive dataset and its fitting function
    $\Omega$: candidate set of parameter vectors
    $m'$: number of parameter vectors to select from $\Omega$
    $\varepsilon_s$: total amount of privacy budget used for selecting $\Omega'$
Output: $\Omega'$: set of selected parameter vectors
1: Initialize $\Omega'$ to empty
2: For each $\omega \in \Omega$, compute $f(D, \omega)$
3: for $i = 1$ to $m'$ do
4:     Use privacy budget $\varepsilon_s/m'$ to apply the exponential mechanism (Section 2.1) or the enhanced exponential machanism (Section 4) to select the parameter vector $\omega^*$ from $\Omega$ that aims to maximize $f(D, \omega^*)$
5:     Remove $\omega^*$ from $\Omega$, and add $\omega^*$ to $\Omega'$
6: end for
7: return $\Omega'$
```

Figure 1 illustrates an example of crossover and mutation in a 4-means clustering problem. Tuples in the database are projected to a 2-dimensional space, denoted by circles. On the left side of the figure, there are two parent center sets, with centers denoted by solid squares and triangles respectively. After crossover, these two parent center sets exchange a pair of centers, generating two intermediate center sets shown in the middle of the figure. Another mutation operation follows, by randomly moving one of the centers to a new location in the space. This leads to the final output of new offspring center sets on the right side of the figure. Note that crossover and mutation do not necessarily enhance the quality of input solutions. In Figure 1, intermediate 1 and offspring 1 benefit from these genetic operations while intermediate 2 and offspring 2 have lower quality than their parents. Hence, it is necessary to perform an effective selection step, presented next.

The selection of top-quality parameter vectors is more complicated since it requires access to the sensitive dataset D, and thus, must be performed in a randomized manner to satisfy differential privacy. Algorithm 2 shows the DP_Select algorithm for this purpose, which takes 5 inputs: the data D, the fitting function f, the candidate set Ω, the number m' of vectors to select from Ω, and the amount of privacy budget ε_s for this operation. DP_Select divides ε_s into m' equal shares, and uses each share to select one parameter vector ω^* under differential privacy, with the goal of maximizing the $f(D, \omega^*)$, i.e., ideally ω^* should be the best fit for the data among all parameter vectors in Ω. The selection of ω^* is performed using either the exponential mechanism (EM) described in Section 2.1 or the enhanced exponential mechanism (EEM), which we elaborate in Section 4.

The correctness of PrivGene directly follows the composability property of differential privacy, described in Section 2.1. Specifically, PrivGene performs DP_Select (the only step that involves the sensitive data) r times, each with privacy budget ε/r, which consumes ε overall. Inside DP_Select, its privacy budget (i.e., $\varepsilon_s = \varepsilon/r$) is used for m' invocations of EM or EEM, each with budget $\varepsilon_s/m' = \varepsilon/(r \cdot m')$. Hence, we reach the following lemma.

LEMMA 1. *PrivGene satisfies ε-differential privacy.*

Parameter Settings. Next, we clarify the selection of parameters m, m', and r, which are the size of the candidate set, the number of parameter vectors selected in each iteration, and the total number of iterations, respectively. Existing studies [6, 7, 30] on robust parameters for GA have suggested that without domain-specific information, $m = 200$ and $m' = 10$ generally lead to good results. We adopt the same setting in PrivGene, except that we set $m' = 1$ when the enhanced exponential mechanism is adopted (for selection of top-quality parameter vectors). We will explain the reason in Section 4.

On the other hand, the choice of r for PrivGene is more sutble. In the non-private setting, GA is usually run until convergence, or for a large number of iterations, in order to find high-quality solutions. However, in PrivGene, a larger value of r also has the negative effect of reducing the privacy budget $\varepsilon_s = \varepsilon/r$ used in each invocation of DP_Select (to select m' top-quality parameter vectors), leading to more noisy selections. Hence, the choice of r should balance the extensiveness of the search and the noisiness in the selections. This balancing is affected by three parameters: the total privacy budget ε, the total number of records n in the dataset D, and the number of top-quality parameter vectors to be selected in each iteration of PrivGene. Intuitively, a higher amount of total budget ε as well as a larger number of data samples reduce the randomness in the selection process, which allows using a larger r. Hence, we heuristically set r to $\mathtt{c} \cdot (n \cdot \varepsilon)/m'$, where $\mathtt{c}$ is a constant determined by the experiments. We will choose an appropriate value for $\mathtt{c}$ in the experimental section.

Finally, the performance of PrivGene is also affected by how the initial candidate set Ω is generated (see Line 1 in Algorithm 1). Intuitively, if Ω contains a parameter vector ω that is reasonably good, then, ω has a high chance to be chosen by DP_Select in the first iteration, and subsequently improved in later iterations into a near-optimal solution. Conversely, if all parameter vectors in Ω are of poor quality, PrivGene may not be able to refine them into good solutions before depleting its privacy budget. As we discuss in Section 5, for certain model fitting tasks (e.g., logistic regression and SVM classification), it is possible to heuristically insert into Ω some parameter vectors that provide relatively good initial solutions without using any privacy budget. When this cannot be done (e.g., for k-means clustering), we simply initiates Ω with random parameter vectors.

4. ENHANCED EXPONENTIAL MECHANISM

This section focuses on EEM, an enhanced version of the exponential mechanism (described in Section 2.1) for PrivGene's settings. EEM aims at applications whose fitting function f can be expressed in the following form:

$$f(D, \omega) = h(\omega) + \sum_{t \in D} q(t, \omega). \tag{4}$$

In the above equation, $h(\omega)$ is a function whose result is independent of the sensitive data D; in other words, releasing the output of h does not violate personal privacy. The *tuple fitting function* $q(t, \omega)$ measures how well the model fits a tuple t in the data. As we show later in Section 5, a broad class of model fitting tasks can be expressed in the form of Equation 4. It is worth mentioning that when the above assumption does not hold, EEM gracefully reduces to the exponential mechanism, as we show soon.

EEM follows the same general idea as the exponential mechanism, which consists of three steps: (i) assign each possible output value ω a fitting score $f(D, \omega)$; (ii) compute a probability for each possible output ω; and (iii) select an output value randomly based on the probabilities calculated in Step (ii). The main difference between EEM and the exponential mechanism is in Step (ii), i.e., probability computation. This is a vital step that determines the result quality of the whole mechanism. In general, the higher the impact of f on the probability of ω to be selected, the better the results, but also the higher amount of released private information. To see this, consider two extreme cases. First, without privacy considerations, we can simply assign probability 1 to the output value ω^* with the highest fitting score $f(D, \omega^*)$, and 0 to all other possible outputs. Clearly, this assignment always returns the result of the highest fitting score. Second, when $f(D, \omega)$ has no impact at all on the probability of ω, the probabilities for all output values are identical, leading to no leakage of private information, but completely random results. The goal of EEM, thus, is to maximize the impact of $f(D, \omega)$ in the probability assignment, while satisfying differential privacy requirements.

Recall from Section 2.1 that, to achieve ε-differential privacy with the exponential mechanism, the probability of an output value $\overline{\omega}$ is proportional to $\exp\left(\varepsilon \cdot f(D, \overline{\omega})/\Delta\right)$, where

$$\Delta \geq 2 \max_{\omega \in \Omega, D, D'} f(D, \omega) - f(D', \omega), \tag{5}$$

and D, D' are two arbitrary neighbor databases.

Observe that the impact of $f(D, \overline{\omega})$ is negatively correlated to Δ, which we call the *dampening factor*. As an extreme case, when the dampening factor far exceeds $\varepsilon \cdot f(D, \overline{\omega})$, the fraction $\varepsilon \cdot f(D, \overline{\omega})/\Delta$ approaches 0, and, consequently, each output value $\overline{\omega}$ is assigned an almost identical probability. The major difference between EEM and the exponential mechanism is that the former uses a different dampening factor that is no larger than the latter does. Specifically, when Equation 4 holds, EEM computes the probability of selecting output value ω with the following dampening factor,

$$\Delta \geq \min\left\{ 2 \max_{t, t' \in \mathcal{T}, \omega \in \Omega} q(t, \omega) - q(t', \omega), \right. \\ \left. 2 \max_{t \in \mathcal{T}, \omega, \omega' \in \Omega} q(t, \omega) - q(t, \omega') \right\}. \tag{6}$$

Otherwise, EEM simply follows the exponential mechanism by setting the dampening factor as inequality 5.

To prove the correctness of EEM with the assumption in Equation 4, we first introduce the following two lemmas.

LEMMA 2. *EEM satisfies ε-differential privacy, if*

$$\Delta \geq 2 \max_{t, t' \in \mathcal{T}, \omega \in \Omega} q(t, \omega) - q(t', \omega). \tag{7}$$

PROOF. Let $D, D' \in \mathcal{T}^n$ be any neighbor databases and t and t' denote the differing tuples in D and D' respectively. Given any $\omega \in \Omega$, we have

$$\begin{aligned} \max_{D, D'} f(D, \omega) - f(D', \omega) &= \max_{D, D'} q(t, \omega) - q(t', \omega) \\ &= \max_{t, t' \in \mathcal{T}} q(t, \omega) - q(t', \omega). \end{aligned}$$

Thus, the dampening factor in the lemma satisfies Equation 5 as

$$\begin{aligned} \Delta &\geq 2 \max_{t, t' \in \mathcal{T}, \omega \in \Omega} q(t, \omega) - q(t', \omega) \\ &= 2 \max_{\omega \in \Omega, D, D'} f(D, \omega) - f(D', \omega). \end{aligned}$$

The proof is complete. □

LEMMA 3. *EEM satisfies ε-differential privacy, if*

$$\Delta \geq 2 \max_{t \in \mathcal{T}, \omega, \omega' \in \Omega} q(t, \omega) - q(t, \omega'). \tag{8}$$

PROOF. Let D and D' be any neighbor databases and t and t' denote the differing tuples in D and D' respectively. For any output $\omega \in \Omega$ of enhanced exponential mechanism $\mathcal{E}$, we have

$$\begin{aligned}
&\frac{\Pr\left(\mathcal{E}(D)=\omega\right)}{\Pr\left(\mathcal{E}(D')=\omega\right)} \\
=\ & \frac{\exp\left(\varepsilon\cdot f(D,\omega)/\Delta\right)}{\sum_{\omega'\in\Omega}\exp\left(\varepsilon\cdot f(D,\omega')/\Delta\right)} \Big/ \frac{\exp\left(\varepsilon\cdot f(D',\omega)/\Delta\right)}{\sum_{\omega'\in\Omega}\exp\left(\varepsilon\cdot f(D',\omega')/\Delta\right)} \\
\leq\ & \frac{\exp\left(\varepsilon\cdot\left(f\left(D,\omega\right)-f\left(D',\omega\right)\right)/\Delta\right)}{\min_{\omega'\in\Omega}\exp\left(\varepsilon\cdot\left(f\left(D,\omega'\right)-f\left(D',\omega'\right)\right)/\Delta\right)}.
\end{aligned}$$

With the assumption given in Equation 4, the above inequality can be rewritten as:

$$\begin{aligned}
\frac{\Pr\left(\mathcal{E}(D)=\omega\right)}{\Pr\left(\mathcal{E}(D')=\omega\right)} &\leq \frac{\exp\left(\varepsilon\cdot\left(q\left(t,\omega\right)-q\left(t',\omega\right)\right)/\Delta\right)}{\min_{\omega'\in\Omega}\exp\left(\varepsilon\cdot\left(q\left(t,\omega'\right)-q\left(t',\omega'\right)\right)/\Delta\right)} \\
&= \max_{\omega'\in\Omega}\exp\left(\varepsilon\cdot\left(\left(q\left(t,\omega\right)-q\left(t',\omega\right)\right)-\left(q\left(t,\omega'\right)-q\left(t',\omega'\right)\right)\right)/\Delta\right) \\
&= \max_{\omega'\in\Omega}\exp\left(\varepsilon\cdot\left(q\left(t,\omega\right)-q\left(t,\omega'\right)+q\left(t',\omega'\right)-q\left(t',\omega\right)\right)/\Delta\right).
\end{aligned}$$

On the other hand,

$$\begin{aligned}
\Delta &\geq 2\max_{t\in\mathcal{T},\omega,\omega'\in\Omega} q(t,\omega)-q(t,\omega') \\
&\geq \max_{t\in\mathcal{T},\omega,\omega'\in\Omega} q(t,\omega)-q(t,\omega') \\
&\qquad\qquad + \max_{t'\in\mathcal{T},\omega,\omega'\in\Omega} q(t',\omega')-q(t',\omega) \\
&\geq \max_{t,t'\in\mathcal{T},\omega,\omega'\in\Omega} q(t,\omega)-q(t,\omega')+q(t',\omega')-q(t',\omega).
\end{aligned}$$

Thus, $\Pr\left(\mathcal{E}(D)=\omega\right)$ is no more than to $e^{\varepsilon}\cdot\Pr\left(\mathcal{E}(D')=\omega\right)$, which completes the proof. □

Combing the results of Lemma 2 and Lemma 3, we prove the correctness of EEM, formally stated in Theorem 1.

THEOREM 1. *EEM satisfies ε-differential privacy with dampening factor in Inequality 6.*

Let Δ_1 denote the right hand side of Inequality 7 and Δ_2 denote that in Inequality 8. According to proofs of Lemmas 2 and 3, Δ_1 is exactly the dampening factor used in the exponential mechanism while Δ_2 is a new dampening factor designed specifically for PrivGene. The dampening factor of EEM is the smaller of the two, which is no larger than Δ_1. Further, when used in PrivGene, Δ_2 is usually smaller than Δ_1. The intuition is that as more iterations are performed, the quality of the parameter vectors in the candidate set becomes increasingly close to each other, since it converges to (possibly local) optimal. This means that it is likely that the maximum value of $q(t,\omega)-q(t,\omega')$ gradually decreases with the number of iterations performed, leading to decreasing Δ_2. Δ_1, on the other hand, is not significantly affected by this phenomenon. Hence, the gap between Δ_1 and Δ_2 expands with the number of iterations, as confirmed by our experiments. The following example illustrates that Δ_2 can be considerably smaller than Δ_1 with candidates of similar fitting quality.

EXAMPLE 1. *Consider that we have a database D that contains one-dimensional tuples in the integer domain $\mathcal{T}=[0,10]$, and candidate set $\Omega=\{6,7,8\}$. The fitting function $f(D,w)$ is defined as $-\sum_{t\in D}(t-\omega)^2$, i.e., we aim to identify the ω that best approximates the mean of the tuples. To enforce ε-differential privacy using EEM, the dampening factor $\Delta \geq \min\{\Delta_1,\Delta_2\}$, where $\Delta_1=128$ under the worst case $t=8, t'=0, \omega=8$ and $\Delta_2=56$ under the worst case $t=0, \omega=6, \omega'=8$.*

In general, the benefit of Δ_2 is more pronounced when the candidate set does not contain two parameter vectors that differ significantly from each other. Therefore, we set $m'=1$ for PrivGene+EEM, i.e., each iteration selects exactly one parameter vector ω from the candidate set to generate offsprings for the next iteration. This ensures that no two offsprings would have significant differences (as they are all mutated from ω). As such, when those offsprings are given to the tuple fitting function as inputs, the outputs of the function would be similar, leading to a small value of Δ_2, and thus, high accuracy of EEM. In particular, as we show in Section 5, when we set $m'=1$ for for logistic regression and SVM classification, Δ_2 is bounded by a multiple of the mutation scale, which leads to a dramatic accuracy boost compared to EEM with other values of m' as well as EM. Hence, in these applications, it is strongly preferred to use EEM and set $m'=1$ in PrivGene.

5. APPLICATIONS

This section applies PrivGene to three common model fitting tasks: logistic regression, SVM classification, and k-means clustering.

5.1 Logistic Regression

Let D be a database containing n tuples from a domain $\mathcal{T}$, such that each tuple has d attributes $X_1, X_2, \ldots, X_{d-1}, Y$, and attribute Y has a binary domain $\{0,1\}$. For each $t=(x,y)=(x_1,x_2,\ldots,x_{d-1},y)$, we assume without loss of generality[1] that $|x_k|\leq 1$ for $k\in\{1,2,\ldots,d-1\}$, i.e., $\mathcal{T}=[-1,1]^{d-1}\times\{0,1\}$. A logistic regression model built on D is parameterized by a vector α and a constant β (called the *bias*), as formalized in Definition 1.

DEFINITION 1 (LOGISTIC REGRESSION). **Logistic regression** *on D predicts $\hat{y}=1$ given $\hat{x}=(\hat{x}_1,\hat{x}_2,\ldots,\hat{x}_{d-1})$ with probability*

$$Pr\left\{\hat{y}=1 \mid \hat{x}\right\} = 1/\left(1+\exp(-\hat{x}^T\alpha^*-\beta^*)\right),$$

where α^* is a vector of $d-1$ real numbers and β^* is a real number, such that

$$(\alpha^*,\beta^*) = \arg\max_{\alpha,\beta}\sum_{t\in D}\left(y\left(x^T\alpha+\beta\right)-\log\left(1+\exp\left(x^T\alpha+\beta\right)\right)\right).$$

To apply PrivGene to logistic regression, each parameter vector in PrivGene has d elements: the first $d-1$ elements represent α and the last one represents β. In what follows, we focus on deriving the dampening factor Δ used in EEM for selecting the top parameter vectors.

First, given the fitting function in Definition 1, the tuple fitting function for $t=(x,y)$ can be expressed as

$$q(t,\omega)=y\left(x^T\alpha+\beta\right)-\log\left(1+\exp\left(x^T\alpha+\beta\right)\right). \tag{9}$$

To derive Δ, it suffices to derive Δ_1 and Δ_2, i.e., the right hand sides of Inequalities 7 and 8, respectively.

Consider Δ_1. Recall that

$$\begin{aligned}
\Delta_1 &= 2\max_{t,t'\in\mathcal{T},\omega\in\Omega} q(t,\omega)-q(t',\omega) \\
&= 2\max_{\omega\in\Omega}\left(\max_{t\in\mathcal{T}} q(t,\omega)-\min_{t\in\mathcal{T}} q(t,\omega)\right). \qquad (10)
\end{aligned}$$

[1] This assumption can be easily enforced by changing each x_k to $\frac{x_k-0.5(\max_k-\min_k)}{(\max_k-\min_k)}$, where $\min_k$ and $\max_k$ denote the minimum and maximum values in the domain of X_k.

By Equation 9, we have $q(t,\omega) \le 0$ for any $t \in \mathcal{T}$. Therefore, $\max_{t\in\mathcal{T}} q(t,\omega) = 0$. To derive $\min_{t\in\mathcal{T}} q(t,\omega)$, we differentiate two cases: $x^T\alpha + \beta \ge 0$ and $x^T\alpha + \beta < 0$.

When $x^T\alpha + \beta \ge 0$, by Equation 9, we have

$$\begin{aligned}
\min_{t\in\mathcal{T}} q(t,\omega) &= \min_{t\in\mathcal{T}} y\left(x^T\alpha+\beta\right) - \log\left(1+\exp\left(x^T\alpha+\beta\right)\right)\\
&= \min_{t\in\mathcal{T}} -\log\left(1+\exp\left(x^T\alpha+\beta\right)\right)\\
&\ge \min_{t\in\mathcal{T}} -\left(x^T\alpha+\beta+1\right)\\
&= -\left(\sum_{k=1}^{d-1}|\alpha_k| + \beta + 1\right) \quad \ge -\left(\sum_{k=1}^{d}|\omega_k|+1\right).
\end{aligned}$$

On the other hand, when $x^T\alpha+\beta<0$, we have

$$\begin{aligned}
\min_{t\in\mathcal{T}} q(t,\omega) &= \min_{t\in\mathcal{T}} y\left(x^T\alpha+\beta\right) - \log\left(1+\exp\left(x^T\alpha+\beta\right)\right)\\
&\ge \min_{t\in\mathcal{T}} y\left(x^T\alpha+\beta\right) - 1\\
&= \left(-\sum_{k=1}^{d-1}|\alpha_k| + \beta\right) - 1 \quad \ge -\left(\sum_{k=1}^{d}|\omega_k|+1\right).
\end{aligned}$$

Combining the above inequalities, we have

$$\Delta_1 \le 2\max_{\omega\in\Omega}\left(\sum_{k=1}^{d}|\omega_k|+1\right). \tag{11}$$

Now consider Δ_2. Recall that

$$\begin{aligned}
\Delta_2 &= 2\max_{t\in\mathcal{T},\omega,\omega'\in\Omega} q(t,\omega) - q(t,\omega')\\
&= 2\max_{\omega,\omega'\in\Omega}\left(\max_{t\in\mathcal{T}} q(t,\omega) - q(t,\omega')\right).
\end{aligned} \tag{12}$$

By Equation 9,

$$q(t,\omega) - q(t,\omega') =$$
$$y\left(\left(x^T\alpha+\beta\right) - \left(x^T\alpha'+\beta'\right)\right) - \log\left(\frac{1+\exp\left(x^T\alpha+\beta\right)}{1+\exp\left(x^T\alpha'+\beta'\right)}\right).$$

If $x^T\alpha+\beta \ge x^T\alpha'+\beta'$, then

$$\begin{aligned}
\max_{t\in\mathcal{T}} q(t,\omega) - q(t,\omega') &\le \max_{t\in\mathcal{T}} y\left(\left(x^T\alpha+\beta\right) - \left(x^T\alpha'+\beta'\right)\right)\\
&= \max_{t\in\mathcal{T}} x^T\left(\alpha-\alpha'\right) + \left(\beta-\beta'\right) \quad \le \sum_{k=1}^{d}\left|\omega_k-\omega'_k\right|.
\end{aligned}$$

On the other hand, if $x^T\alpha+\beta < x^T\alpha'+\beta'$, then

$$\begin{aligned}
\max_{t\in\mathcal{T}} q(t,\omega) - q(t,\omega') &\le \max_{t\in\mathcal{T}} -\log\left(\frac{1+\exp\left(x^T\alpha+\beta\right)}{1+\exp\left(x^T\alpha'+\beta'\right)}\right)\\
&\le \max_{t\in\mathcal{T}} -\log\left(\frac{\exp\left(x^T\alpha+\beta\right)}{\exp\left(x^T\alpha'+\beta'\right)}\right)\\
&= \max_{t\in\mathcal{T}}\left(x^T\alpha'+\beta'\right) - \left(x^T\alpha+\beta\right) \quad \le \sum_{k=1}^{d}\left|\omega_k-\omega'_k\right|.
\end{aligned}$$

Based on the above inequalities, we have

$$\Delta_2 \le 2\max_{\omega,\omega'\in\Omega}\sum_{k=1}^{d}\left|\omega_k-\omega'_k\right|. \tag{13}$$

Combining Inequalities 6, 11, and 13, we have

$$\Delta \ge \min\left\{2\max_{\omega\in\Omega}\left(\sum_{k=1}^{d}|\omega_k|+1\right), 2\max_{\omega,\omega'\in\Omega}\sum_{k=1}^{d}\left|\omega_k-\omega'_k\right|\right\}.$$

Observe that when $m' = 1$ (i.e., only one parameter vector is selected in each iteration), after the first iteration, all candidate parameter vectors are generated from the same parent. Hence, the L_1 distance between any two of them is bounded by twice the mutation scale. Therefore, Δ_2 is bounded by 4 times the mutation scale. For any $m' \ne 1$, this bound no longer holds, and Δ_2 can be as large as twice the size of the parameter vector domain. Hence, setting $m' = 1$ provides a significant accuracy boost to EEM. The accuracy of EM, however, is not dramatically affected by m', since Δ_1 can always be as large as twice the domain size for any m'.

5.2 SVM Classification

Support vector machine (SVM) [5] is a popular tool for *classification*, which predicts the labels of new observations based on existing observations with labels. For simplicity, we focus on SVM with linear *kernels* [29]; our solution can be extended to SVMs with non-linear kernels similarly as in [29].

Let $D \in \mathcal{T}^n$ be a database containing n tuples sampled from a d-dimensional domain $\mathcal{T} = [-1,1]^{d-1} \times \{-1,1\}$. We denote the i-th ($i \in [1, d-1]$) dimension of $\mathcal{T}$ as X_i, and the last dimension of $\mathcal{T}$ as Y. For ease of exposition, we use x to denote a vector in $[-1,1]^{d-1}$, and use $t = (x,y)$ to denote a tuple in D. A linear SVM classifier on D is defined as follows.

DEFINITION 2 (SVM CLASSIFICATION). *Given $\hat{x} = (\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_{d-1})$, a* **linear SVM classifier** *on D predicts the Y value associated with $\hat{x}$ as:*

$$\hat{y} = \begin{cases} 1, & \textit{if } \hat{x}^T\alpha^* + \beta^* > 0\\ -1, & \textit{otherwise}\end{cases}$$

where α^* is a vector of $d-1$ real numbers and β^* is a real number, such that

$$(\alpha^*,\beta^*) = \arg\max_{\alpha,\beta} -\left(\frac{1}{2}||\alpha||^2 + C\sum_{(x,y)\in D}\xi(x,y)\right),$$

where $\xi(x,y) = \max\left\{1 - y(x^T\alpha+\beta), 0\right\}$, referred to as the **hinge-loss** *function.*

To solve SVM classification with PrivGene, we define each parameter vectors as a d-dimensional vector, such that the first i ($i \in [1, d-1]$) dimensions correspond to α and the last dimension correspond to β. In addition, the tuple fitting function is defined as

$$q(t,\omega) = -C\max\left\{1 - y(x^T\alpha+\beta), 0\right\}, \tag{14}$$

which is in accordance with Definition 2. As for the damping factor Δ for EEM, we derive it based on an analysis of Δ_1 and Δ_2, as formulated in Equations 10 and 12.

First, let us consider Δ_1. By Equation 10, we can calculate an upperbound for Δ_1 based on the maximum and minimum possible values of $q(t,\omega)$ for $t \in \mathcal{T}$. By Equation 14, $\max_{t\in\mathcal{T}} q(t,\omega) = 0$ trivially holds. Meanwhile,

$$\begin{aligned}
\min_{t\in\mathcal{T}} q(t,\omega) &= \min_{t\in\mathcal{T}} -C\max\left\{1 - y(x^T\alpha+\beta), 0\right\}\\
&= -C\max_{t\in\mathcal{T}}\left(1 + \left|x^T\alpha+\beta\right|\right)\\
&\ge -C\left(\sum_{k=1}^{d}|\omega_k|+1\right).
\end{aligned}$$

Therefore, we have an upperbound of Δ_1 as follows:

$$\Delta_1 \leq 2C \max_{\omega \in \Omega} \left(\sum_{k=1}^{d} |\omega_k| + 1 \right). \tag{15}$$

Next, we derive an upperbound of Δ_2. This, by Equation 12, can be achieved by upperbouding $q(t, \omega) - q(t, \omega')$ for any pair of parameter vectors ω, ω' and any tuple t. Observe that, for any $t = (x, y)$, we have

$$\begin{aligned} & q(t, \omega) - q(t, \omega') \\ & = C \max \left\{ 1 - y(x^T \alpha' + \beta'), 0 \right\} - C \max \left\{ 1 - y(x^T \alpha + \beta), 0 \right\} \\ & = C \cdot \frac{1 - y(x^T \alpha' + \beta') + \left| 1 - y(x^T \alpha' + \beta') \right|}{2} \\ & \qquad - C \cdot \frac{1 - y(x^T \alpha + \beta) + \left| 1 - y(x^T \alpha + \beta) \right|}{2} \\ & \leq C \cdot \frac{1}{2} y \left(\left(x^T \alpha + \beta \right) - \left(x^T \alpha' + \beta' \right) \right) \\ & \qquad + C \cdot \frac{1}{2} \left| y \left(\left(x^T \alpha + \beta \right) - \left(x^T \alpha' + \beta' \right) \right) \right| \\ & \leq C \left| y \left(\left(x^T \alpha + \beta \right) - \left(x^T \alpha' + \beta' \right) \right) \right|. \end{aligned}$$

Meanwhile,

$$\begin{aligned} & \max_{t \in \mathcal{T}} C \left| y \left(\left(x^T \alpha + \beta \right) - \left(x^T \alpha' + \beta' \right) \right) \right| \\ & = \max_{t \in \mathcal{T}} C \left| y \left(x^T \left(\alpha - \alpha' \right) + \left(\beta - \beta' \right) \right) \right| \quad = C \sum_{k=1}^{d} \left| \omega_k - \omega'_k \right|. \end{aligned}$$

Therefore, we have the following upperbound for Δ_2:

$$\Delta_2 \leq 2C \max_{\omega, \omega' \in \Omega} \sum_{k=1}^{d} \left| \omega_k - \omega'_k \right|. \tag{16}$$

By combining the upperbounds for Δ_1 and Δ_2, we obtain the following dampening factor Δ:

$$\Delta \geq 2C \cdot \min \left\{ \max_{\omega \in \Omega} \left(\sum_{k=1}^{d} |\omega_k| + 1 \right), \max_{\omega, \omega' \in \Omega} \sum_{k=1}^{d} \left| \omega_k - \omega'_k \right| \right\}.$$

Similar to the case of logistic regression, when (and only when) $m' = 1$, Δ_2 is bounded by $2C$ times the mutation scale. Hence, EEM with $m' = 1$ leads to high accuracy for PrivGene.

5.3 *k*-means Clustering

Let k be a positive integer, and D be a private database containing n tuples from a domain $\mathcal{T} = [-1, 1]^d$. Given k and D, a *k-means clustering* [25] on D identifies k elements $c_1, c_2, \ldots, c_k \in \mathcal{T}$ (referred to as *centers*), such that each tuple in D has a small distance to at least one center. The formal definition is as follows:

DEFINITION 3 (k-MEANS CLUSTERING). *A k-means clustering on D returns a set $\mathcal{C} = \{c_1, c_2, \ldots, c_k\}$, such that $c_i \in \mathcal{T}$ ($i \in [1, d]$) and*

$$\mathcal{C}^* = \arg\max_{\mathcal{C}} - \sum_{t \in D} dist(t, \mathcal{C}), \tag{17}$$

where $dist(t, \mathcal{C}) = \min_{c \in \mathcal{C}} ||t - c||_2^2$.

To adopt PrivGene for k-mean clustering, we define each parameter vector ω as a $k \cdot d$-dimensional vector from $[-1, 1]^{kd}$, such that the $(k \cdot i + j)$-th element in ω represents the j-th coordinate of the i-th center c_i. For convenience, we abuse notation and define ω_i as a d-dimensional vector whose j-th element equals the $(k \cdot i + j)$-th element in ω, i.e., ω_i represents c_i. Then, the tuple fitting function for any tuple $t \in D$ is defined as

$$q\left(t, \omega\right) = - \min_{i \in [1, d]} ||t - \omega_i||_2^2.$$

To derive the dampening factor Δ for EEM, we need to calculate an upperbound for Δ_1 (in Equation 10) and Δ_2 (in Equation 12).

For Δ_1, we have

$$\begin{aligned} \Delta_1 \quad & \leq \max_{t, t' \in \mathcal{T}, \omega \in \Omega} q(t, \omega) - q(t', \omega) \quad \leq \max_{t \in \mathcal{T}, \omega \in \Omega} -q(t, \omega) \\ & = \max_{\mathcal{C} \in \Omega} \max_{t \in \mathcal{T}} \min_{c \in \mathcal{C}} ||t - c||_2^2 \quad \leq \max_{\mathcal{C} \in \Omega} \min_{c \in \mathcal{C}} \max_{t \in \mathcal{T}} ||t - c||_2^2. \end{aligned} \tag{18}$$

On the other hand, we find it difficult to derive a non-trivial upperbound for Δ_2, as it is hard to quantify the maximum value of $q\left(t, \omega\right) - q\left(t, \omega'\right)$ for any $t \in \mathcal{T}$ and any $\omega, \omega' \in \Omega$. To explain, observe that for each tuple $t \in \mathcal{T}$, the value of $q\left(t, \omega\right)$ is decided by the center closest to t (among all d centers represented by ω). Assume without loss of generality that, among the centers represented by ω, the i-th center is closet to t. Suppose that we replace ω with ω'. In that case, the closet center to t might change from the i-th center c_i to the j-th center c_j ($i \neq j$), in which case $q\left(t, \omega'\right)$ is decided by the c_j instead of c_i. Therefore, if we are to derive the maximum value of $q\left(t, \omega\right) - q\left(t, \omega'\right)$, then we must take into account all possible changes in the center closest to t, which leads to highly complicated analysis.

Due to the difficulty in upperbouding Δ_2, we use only Δ_1 to derive Δ, resulting in $\Delta \geq \min\{\Delta_1, \Delta_2\} = \Delta_1$. In that case, EEM is degenerated to the exponential mechanism (EM), since Δ_1 is exactly the dampening factor used by the latter. In other words, our solution for k-means clustering adopts EM instead of EEM.

5.4 Choosing Initial Candidate Set

As discussed in Section 3, the performance of PrivGene can be improved if the initial candidate set Ω contains at least one reasonably good parameter vector. Towards this end, for logistic regression and SVM classification, we heuristically generate $m = 200$ parameter vectors in Ω as follows. First, we insert into Ω 180 vectors that are randomly selected from the solution space. After that, we add another 10 vectors to Ω, such that the first $d - 1$ elements in each vector equal 0, while the last element is a random positive number. We denote this set of 10 vectors as Ω^+. Finally, we add 10 vectors (denoted by Ω^-) to Ω with a random negative number in the last dimension, and 0 in all other dimensions.

Essentially, each parameter vector in Ω^+ (resp. Ω^-) gives a naive logistic or SVM model that predicts $Y = 1$ (resp. $Y \neq 1$) for any given tuple. Although such naive models are not accurate in general, they can sometimes serve as a good starting point for PrivGene. To explain, let us consider a dataset D where 70% of the tuples have the same Y values (regardless of whether they all have $Y = 1$ or $Y \neq 1$). Then, at least 10 parameter vectors in Ω provide a model that can correctly predict the Y value of 70% of the tuples, resulting in 70% predication accuracy. This, intuitively, gives PrivGene a good initial set of solutions. In general, Ω^+ and Ω^- tend to improve the performance of PrivGene for logitic regression and SVM classification, especially when a large majority of the tuples in the given dataset have the same values on Y. Note that the generation of Ω^+ and Ω^- incur no privacy cost, as they are independent of the input data.

For k-means clustering, however, there is no obvious way to generate parameter vectors that correspond to reasonably good solutions. In that case, we resort to populating Ω with 200 vectors randomly sampled from the solution space.

Table 2: Datasets properties.

Dataset	Number of tuples	Dimensionality	Task
Adult	$48,842$	124	Predict if a person makes over 50k USD per year
Banking	$45,211$	33	Predict if a client subscribes a term deposit
US	$40,000$	58	Predict if a person makes over 25k USD per year
BR	$38,000$	53	Predict if a person makes over 300 USD per month
Lifesci	$26,733$	10	k-means clustering with k equals 3 and 5
Image	$34,112$	3	k-means clustering with k equals 10 and 15

6. EXPERIMENTS

This section experimentally evaluates PrivGene on six real datasets: (i) Adult [2, 12], which includes information of $48,842$ individuals extracted from the 1994 US Census database, (ii) Banking [12], a marketing dataset from a banking institution on $45,211$ individuals, (iii) US [1], containing $40,000$ US census records, (iv) BR [1], consisting of $38,000$ Brazilian census records, (v) Lifesci, a life sciences dataset, available at http://komarix.org/ac/ds/, (vi) Image, an image dataset with $34,112$ RGB vectors retrieved from http://cs.joensuu.fi/sipu/datasets/. Adult, Banking, BR and US are used for logistic regression and SVM classification; Lifesci and Image are used in the k-means clustering experiments.

Adult, Banking, US, and BR contain both continuous and categorical attributes. Following common practice in regression and classification, we transform each categorical attribute with l possible values into l binary attributes. We also normalize the values of each attribute to the range $[-1, 1]$. After these preprocessing steps, the dimensionalities of datasets Adult, Banking, US, and BR become 124, 33, 58 and 53, respectively. For k-means clustering, we choose $k = 3, 5$ for Lifesci and $k = 10, 15$ for Image. Table 2 summarizes the properties of each dataset.

As shown in Table 2, in each regression or classification task, we label a tuple with $Y = 1$ iff. it belongs to a specific class, e.g., yearly income over 50k in Adult. In particular, in logistic regression, we predict a tuple t to be in the $Y = 1$ class, whenever t has over 50% probability to have $Y = 1$ under the logistic model. We measure the performance of a logistic model or an SVM classifier by its *misclassification rate*, i.e., the fraction of tuples in the testing dataset that are incorrectly classified. The performance of a k-means clustering method is evaluated by its average *intra-cluster variance* (also used in [27]), defined as $\frac{1}{|D|}\sum_{t\in D}\min_{c\in\mathcal{C}}||t-c||_2^2$, where D is the testing data and $\mathcal{C}$ is the set of k cluster centers.

We compare PrivGene against six competitors, namely, *GUPT* [27], *Functional Mechanism (FM)* [35], *PrivateERM* [3], *PrivateSVM* [29], *NoPrivacy*, and *Majority*. As explained in Section 2, GUPT is a generic method that can handle all three model fitting tasks in our experiments. FM and PrivateSVM are limited to logistic regression and SVM classification, respectively. PrivateERM is limited to specific classes of logistic regression (i.e., with a non-zero regularization term as explained in Section 2) and SVM classification (using the Huber loss function rather than the more popular hinge loss). We include it in our evaluations anyway, with a very small (10^{-12}) regularization factor in logistic regression, and Huber loss in SVM classification. The remaining SVM classification solutions (i.e., PrivGene, PrivateSVM and NoPrivacy) employ hinge loss. Note that the effectiveness of PrivateERM is sensitive to the regularization factor. In our experiments, we use fixed values of the factor that lead to relatively good performance; using the auto-tuning algorithm in [3] leads to strictly and significantly worse results than what we report. These settings are in favor of PrivateERM.

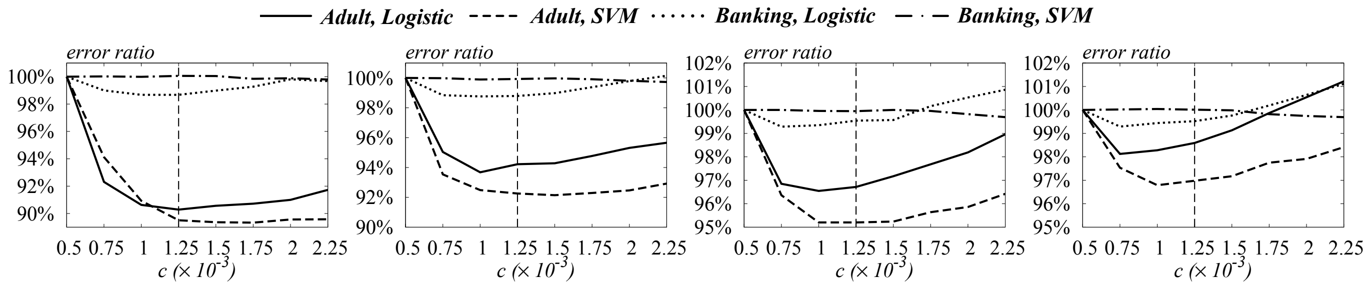
NoPrivacy directly releases the best parameters without any privacy consideration. Finally, Majority is a naive differentially private classification method: it first counts the number of tuples in the training data with $Y = 1$, and then adds Laplace noise to the count to ensure ϵ-differential privacy; if the noisy count is larger than $n/2$ (n is the number of tuples in the training data), Majority always outputs $Y = 1$; otherwise, it always outputs $Y \neq 1$. Clearly, NoPrivacy provides the best possible accuracy for any private method, whereas Majority indicates an upper-bound for the misclassification rate in logistic regression and SVM classification.

Unless specified otherwise, we use the default parameter settings for each method as in previous work [3,27,29,35]. GUPT requires an explicitly defined search space for parameter vectors. For regression and classification, we set the search space to $[-5, 5]^d$, where d is the number of elements in each parameter vector ω. For k-means clustering, we naturally set the search space to the domain of a data tuple. In addition, SVM classification requires a regularization parameter c. For NoPrivacy, we test a range of c's and select the best one. For all other methods, we arbitrarily set $C = 10$ without tuning, so as to avoid leaking private information. Finally, in every regression/classification experiment, we evaluate each method by repeating 5-fold cross-validation 500 times, and report the average result. In each k-means clustering experiment, we run every method 500 times and report the average result.

6.1 Effect of Number of Iterations

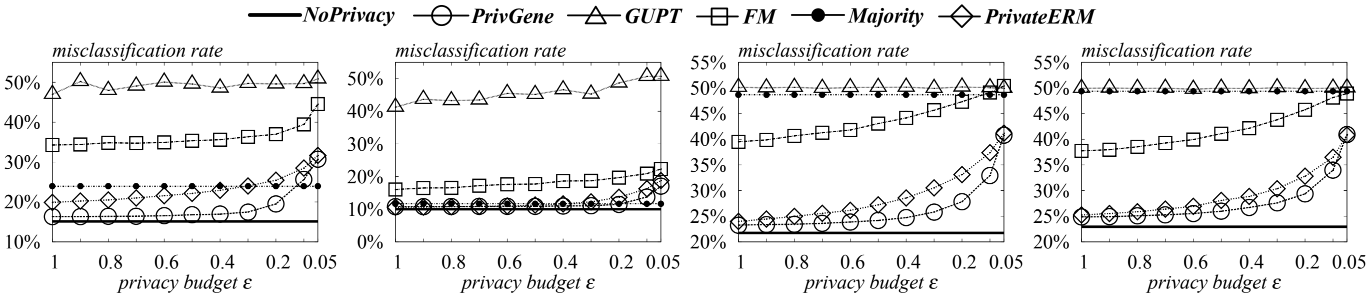
PrivGene has several internal parameters, which are all fixed to values suggested in the genetic algorithms literature, except for the number of iterations r. As discussed in Section 3, we adopt a heuristic $r = \mathsf{c}\cdot(n\cdot\varepsilon)/m'$, where c is a parameter to be tuned, n is the number of tuples in the input data, and $m' = 1$ (resp. $m' = 10$) when EEM (resp. EM) is incorporated with PrivGene. To choose an appropriate value for c, we conduct experiments as follows. First, we run PrivGene+EEM for logistic regression and SVM classification on Adult and Banking, based on which we select a fixed value for c, without looking at the other datasets or the task of k-means clustering. After that, we use the selected c for all experiments. This ensures (i) that our choice of c does not reveal private information on datasets US and BR, and (ii) that PrivGene can compete against other methods without relying on manual tuning of parameters. Meanwhile, since our parameter tuning is performed based on Adult and Banking, the experimental results on the other four datasets are more important.

Figure 2 illustrates the misclassification rate of PrivGene with varying values of c, as a ratio of the misclassification rate when $\mathsf{c} = 0.5 \times 10^{-3}$. Observe that the misclassification rate of PrivGene tends to be high when c is either excessively small or excessively large. This is consistent with our analysis in Section 3 that (i) a small c prevents PrivGene from converging to the optimal so-



(a) $n \cdot \varepsilon / m' = 16000$ (b) $n \cdot \varepsilon / m' = 20000$ (c) $n \cdot \varepsilon / m' = 24000$ (d) $n \cdot \varepsilon / m' = 28000$

Figure 2: Number of iterations in Logistic regression and SVM classification.



(a) Adult (b) Banking (c) US (d) BR

Figure 3: Logistic regression on different datasets.

lutions, and (ii) a large c renders it difficult for PrivGene to choose the top-quality offsprings in each iteration, both of which lead to inferior overall performance. Based the experimental results, we set $c = 1.25 \times 10^{-3}$ (the dotted vertical line in Figure 2), and use this value in all remaining experiments.

6.2 Comparisons with Existing Solutions

This section compares PrivGene with existing solutions on three model fitting tasks: logistic regression, SVM classification, and k-means clustering. Note that previous work [27] evaluates GUPT on exactly the same three tasks.

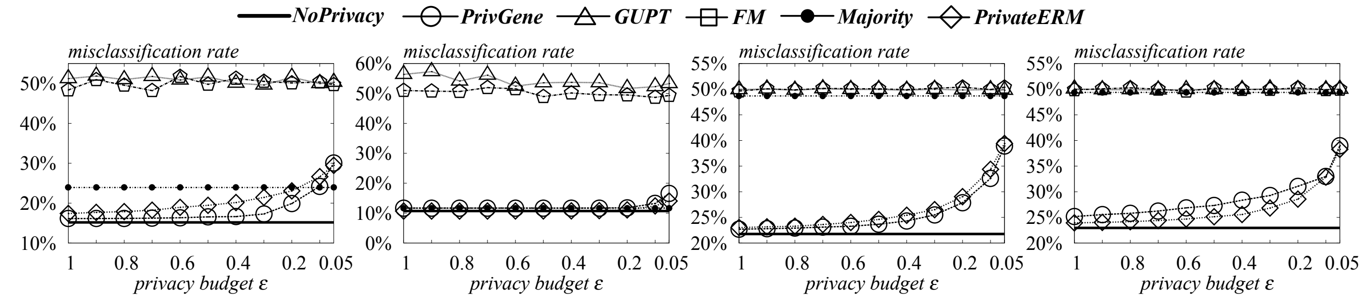
Logistic Regression. Figure 3 shows the misclassfication rate of each algorithm, with varying privacy budget ε. The error of NoPrivacy remains unchanged for all values of ε since it does not enforce ε-differential privacy at all. Clearly, both PrivGene and PrivateERM outperform FM and GUPT in all settings. Comparing PrivGene and PrivateERM, the former achieves consistently better accuracy than the latter, except when the privacy budget ε becomes extremely small (i.e., 0.05), in which case both methods obtain comparable (and relatively low) accuracy. For reasonably large ε, both PrivGene and PrivateERM achieve accuracy close to NoPrivacy. This result reassures that differential privacy is indeed a practical technique for model fitting on sensitive data, provided that an appropriate solution is used for the task.

Regarding Majority, its accuracy is not sensitive to the value of ε, because (i) there are large gaps between the number of records in the majority and minority classes on the Adult and Banking datasets, and (ii) the two classes have similar number of records on US and BR; consequently, Majority's accuracy is close to that of a wild guess (i.e., 50%). PrivGene outperforms Majority, except when ε is very small (i.e., < 0.01), or when there is a dominating majority class (e.g., on Banking, since few people subscribe to a term deposit), where the Majority's accuracy is already close to that of NoPrivacy, leaving little space for improvement.

SVM Classification. This set of experiments compares PrivGene with existing methods on SVM classification. Figure 4 illustrates the average misclassification rate of SVM classifiers trained by different algorithms. For all 4 datasets, PrivGene and PrivateERM are highly competitive, whereas GUPT and PrivateSVM report close-to-random class labels. Meanwhile, the accuracy of the former two is again close to that of NoPrivacy in all settings, and higher than Majority in most cases, which confirms their practical usefulness. PrivGene slightly outperforms PrivateERM on Adult and US, and the reverse is true on BR. This is because PrivGene's accuracy is slight better on Adult and US than on BR, which is also evident in the set of logistic regression experiments. The reason is that the parameter c is tuned based on Adult and Banking is sub-optimal on BR. Nevertheless, the accuracy loss due to this sub-optimal c is small. Concerning Banking, the difference in accuracy of PrivGene, Majority and PrivateERM is negligible, due to the presence of a dominating majority class.

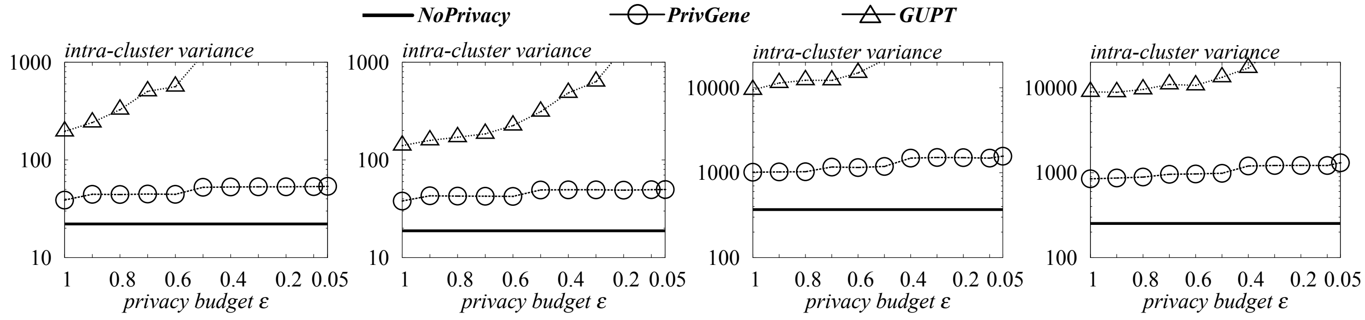
k-means Clustering. Figure 5 exhibits the average intra-cluster variance of each algorithm. Note that the accuracy results for Image are shown in log scale. PrivGene is again the clear winner in all settings, and the performance gap between PrivGene and GUPT increases rapidly with decreasing ε. At $\varepsilon = 0.1$, the accuracy of PrivGene is two orders of magnitude better than that of GUPT. Both PrivGene and GUPT are more sensitive to ε compared to logistic regression and SVM classification tasks, which suggests that differentially private k-means clustering is a more difficult problem.

In summary, PrivGene outperforms both the general solution GUPT and specialized solutions FM (for logistic regression) and PrivateSVM (for SVM classification). The only method that can



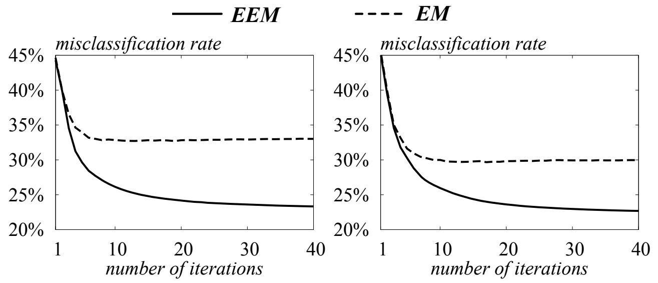
(a) Adult (b) Banking (c) US (d) BR

Figure 4: SVM classification on different datasets.



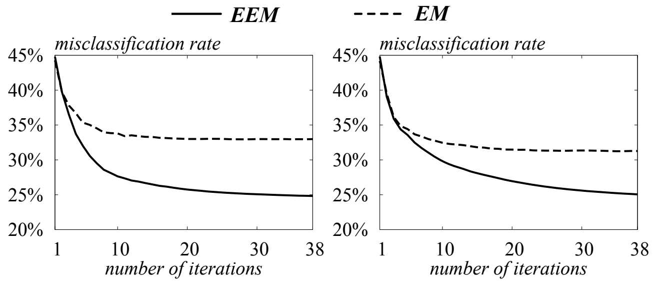
(a) Lifesci, $k = 3$ (b) Lifesci, $k = 5$ (c) Image, $k = 10$ (d) Image, $k = 15$

Figure 5: k-means clustering on different datasets.



(a) Logistic regression (b) SVM classification

Figure 6: Improvement of EEM on the US dataset



(a) Logistic regression (b) SVM classification

Figure 7: Improvement of EEM on the BR dataset

compete with PrivGene is PrivateERM, which, however, relies on rather strong assumptions of the model fitting task as well as a good selection of the regularization factor. The accuracy of PrivGene is often close to that of NoPrivacy, and is generally robust against data dimensionality and (to a lesser degree) the amount of privacy budget ε. These results suggest that PrivGene is the method of choice for all three model fitting tasks.

6.3 Improvement of EEM

Having established the superiority of PrivGene over existing solutions, we next investigate the intrinsic characteristics of PrivGene. In particular, we demonstrate the effectiveness of using EEM in comparison with vanilla exponential mechanism. Figure 6a and 6b illustrate the performance of the exponential mechanism (with $m' = 1$ in order to facilitate the comparison) and EEM in terms of misclassification rate in the tasks of logistic regression and SVM classification, respectively, using the US dataset with ε fixed to its maximal value 1. Using a different value for ε leads to similar conclusions. There is a gap between the misclassification rate of PrivGene with EEM and with exponential mechanism, and the gap increases rapidly with the number of iterations. When PrivGene finishes all iterations, EEM wins by over 10% in terms of misclassification rate for logistic regression, and 7% for SVM classification.

Figure 7 demonstrates the results from the BR dataset with same settings as US. Again, EEM is highly effective for PrivGene compared to the original exponential mechanism. In particular, the use of EEM reduces the misclassification rate by up to 8% for logistic regression, and up to 6% for SVM classification. Considering that the accuracy of PrivGene is already close to that of NoPrivacy, improvement in PrivGene's misclassification rate is difficult; hence, the use of EEM is strongly recommended.

7. CONCLUSIONS AND FUTURE WORK

This paper presents PrivGene, a general framework for differentially private model fitting. Unlike most existing solutions that apply the differentially private version of a popular algorithm in the non-private setting to a restricted class of problems, PrivGene is based on the happy marriage of differential privacy with genetic algorithms, which achieves high accuracy for a broad class of model fitting problems. In addition, we propose EEM, an improved version of the exponential mechanism for PrivGene. We show that PrivGene outperforms existing solutions on three common model fitting tasks: logistic regression, SVM classification, and k-means clustering, and the accuracy of PrivGene is often close to the baseline approach without privacy considerations. Regarding future work, we plan to investigate the possibility of applying PrivGene to problems besides model fitting, e.g., dimensionality reduction and frequent pattern mining. Additionally, we are also interested in generalizing EEM to improve the accuracy of other differentially private algorithms.

Acknowledgement

This work was supported by the Nanyang Technological University under SUG Grant M58020016, and by A*STAR under SERC Grant 102-158-0074. The authors would like to thank the anonymous reviewers for their insightful comments.

8. REFERENCES

[1] https://international.ipums.org.

[2] C.-C. Chang and C.-J. Lin. Libsvm: A library for support vector machines. page 27, 2011.

[3] K. Chaudhuri, C. Monteleoni, and A. D. Sarwate. Differentially private empirical risk minimization. *Journal of Machine Learning Research*, 12:1069–1109, 2011.

[4] G. Cormode, C. M. Procopiuc, E. Shen, D. Srivastava, and T. Yu. Differentially private spatial decompositions. In *ICDE*, 2012.

[5] C. Cortes and V. Vapnik. Support-vector networks. *Mach. Learn.*, 20(3):273–297, Sept. 1995.

[6] L. Davis. Adapting operator probabilities in genetic algorithms. In *Proceedings of the third international conference on Genetic algorithms*, pages 61–69, San Francisco, CA, USA, 1989. Morgan Kaufmann Publishers Inc.

[7] L. Davis. *Handbook of Genetic Algorithms*. Van Nostrand Reinhold, New York, New York, USA, 1991.

[8] B. Ding, M. Winslett, J. Han, and Z. Li. Differentially private data cubes: optimizing noise sources and consistency. In *SIGMOD*, pages 217–228, 2011.

[9] C. Dwork. Differential privacy. In *ICALP*, pages 1–12, 2006.

[10] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *TCC*, pages 265–284, 2006.

[11] P. Elliot, J. C. Wakefield, N. G. Best, and D. J. Briggs. *Spatial Epidemiology; Methods and applications*. Oxford University Press, 2000.

[12] A. Frank and A. Asuncion. UCI machine learning repository, 2010.

[13] A. Friedman and A. Schuster. Data mining with differential privacy. In *KDD*, pages 493–502, 2010.

[14] D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1st edition, 1989.

[15] M. Hay, V. Rastogi, G. Miklau, and D. Suciu. Boosting the accuracy of differentially private histograms through consistency. *PVLDB*, 3(1):1021–1032, 2010.

[16] J. H. Holland. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence*. MIT Press, Cambridge, MA, USA, 1992.

[17] D. Hosmer and S. Lemeshow. *Applied Logistic Regression*. Wiley Series in Probability and Statistics: Texts and References Section. Wiley, 2000.

[18] A. Inan, M. Kantarcioglu, G. Ghinita, and E. Bertino. Private record matching using differential privacy. In *EDBT*, pages 123–134, 2010.

[19] D. Kifer and A. Machanavajjhala. No free lunch in data privacy. In *SIGMOD*, pages 193–204, 2011.

[20] D. Kifer and A. Machanavajjhala. A rigorous and customizable framework for privacy. In *PODS*, 2012.

[21] D. Kifer, A. D. Smith, and A. Thakurta. Private convex optimization for empirical risk minimization with applications to high-dimensional regression. *Journal of Machine Learning Research - Proceedings Track*, 23:25.1–25.40, 2012.

[22] B. R. Kirkwood. *Essentials of medical statistics*. Blackwell Scientific Publications, 1988.

[23] C. Li, M. Hay, V. Rastogi, G. Miklau, and A. McGregor. Optimizing linear counting queries under differential privacy. In *PODS*, pages 123–134, 2010.

[24] N. Li, W. Qardaji, D. Su, and J. Cao. Privbasis: Frequent itemset mining with differential privacy. *PVLDB*, 5(11):1340–1351, 2012.

[25] S. Lloyd. Least squares quantization in pcm. *IEEE Trans. Inf. Theor.*, 28(2):129–137, Sept. 2006.

[26] F. McSherry and K. Talwar. Mechanism design via differential privacy. In *FOCS*, pages 94–103, 2007.

[27] P. Mohan, A. Thakurta, E. Shi, D. Song, and D. Culler. Gupt: privacy preserving data analysis made easy. In *SIGMOD*, pages 349–360, 2012.

[28] M. S. Pepe. *The Statistical Evaluation of Medical Tests for Classification and Prediction*. Oxford Statistical Science Series. Oxford University Press, 2004.

[29] B. I. P. Rubinstein, P. L. Bartlett, L. Huang, and N. Taft. Learning in a large function space: Privacy-preserving mechanisms for svm learning. *Journal of Privacy and Confidentiality*, 4(1):65–100, 2012.

[30] J. D. Schaffer, R. A. Caruana, L. J. Eshelman, and R. Das. A study of control parameters affecting online performance of genetic algorithms for function optimization. In *Proceedings of the third international conference on Genetic algorithms*, pages 51–60. Morgan Kaufmann Publishers Inc., 1989.

[31] A. Smith. Privacy-preserving statistical estimation with optimal convergence rate. In *STOC*, 2011.

[32] X. Xiao, G. Wang, and J. Gehrke. Differential privacy via wavelet transforms. In *ICDE*, pages 225–236, 2010.

[33] J. Xu, Z. Zhang, X. Xiao, Y. Yang, and G. Yu. Differentially private histogram publication. In *ICDE*, 2012.

[34] G. Yuan, Z. Zhang, M. Winslett, X. Xiao, Y. Yang, and Z. Hao. Low-rank mechanism: Optimizing batch queries under differential privacy. *PVLDB*, 5(11):1352–1363, 2012.

[35] J. Zhang, Z. Zhang, X. Xiao, Y. Yang, and M. Winslett. Functional mechanism: Regression analysis under differential privacy. *PVLDB*, 5(11):1364–1375, 2012.