5-2017

DPWeka: Achieving Differential Privacy in WEKA

Srinidhi Katla
University of Arkansas, Fayetteville

DPWeka: Achieving Differential Privacy in WEKA

A thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Science in Computer Science

by

Srinidhi Katla
Jawaharlal Nehru Technological Institute
Bachelor of Technology in Electrical and Electronics Engineering, 2010

May 2017
University of Arkansas

This thesis is approved for recommendation to the Graduate Council.

_____
Dr. Xintao Wu
Thesis Director

_____
Dr. Wing Ning Li
Committee Member

_____
Dr. Qinghua Li
Committee Member

Abstract

Organizations belonging to the government, commercial, and non-profit industries collect and store large amounts of sensitive data, which include medical, financial, and personal information. They use data mining methods to formulate business strategies that yield high long-term and short-term financial benefits. While analyzing such data, the private information of the individuals present in the data must be protected for moral and legal reasons. Current practices such as redacting sensitive attributes, releasing only the aggregate values, and query auditing do not provide sufficient protection against an adversary armed with auxiliary information. In the presence of additional background information, the privacy protection framework, differential privacy, provides mathematical guarantees against adversarial attacks.

Existing platforms for differential privacy employ specific mechanisms for limited applications of data mining. Additionally, widely used data mining tools do not contain differentially private data mining algorithms. As a result, for analyzing sensitive data, the cognizance of differentially private methods is currently limited outside the research community.

This thesis examines various mechanisms to realize differential privacy in practice and investigates methods to integrate them with a popular machine learning toolkit, WEKA. We present DPWeka, a package that provides differential privacy capabilities to WEKA, for practical data mining. DPWeka includes a suite of differential privacy preserving algorithms which support a variety of data mining tasks including attribute selection and regression analysis. It has provisions for users to control privacy and model parameters, such as privacy mechanism, privacy budget, and other algorithm specific variables. We evaluate private algorithms on real-world datasets, such as genetic data and census data, to demonstrate the practical applicability of DPWeka.

Acknowledgements

There are a number of people I want to thank for their help with my thesis.

First of all, I would like to express my immense gratitude to my advisor, Dr. Xintao Wu. This project would not have been possible without his support and encouragement. He contributed virtually to all ideas in this thesis. In spite of his overwhelming schedule, he made time to thoroughly review my work, pointing out when I had gone wrong, and when things could be improved. I also knew when I did well, because he let me know. His wisdom, passion, dedication, and attention to detail are awe-inspiring and constantly motivated me to raise my standards. He definitely provided more than any supervisor would.

Secondly, I would like to thank my lab-mates, a lot. They are a daily source of reminder that hard work and persistence leads to success. In fact, Shuhan Yuan's and Qiuping Pan's constant question of "Have you finished your thesis?" has been motivation to push through the progress of my thesis. I would like to thank Depeng Xu and Qiuping Pan for all the discussions on differential privacy and genetic database. They certainly made wading through the dense concepts lot easier. As a matter of fact, the general discussions and the regular group meetings with everyone in the lab (Yongkai Wu, Panpan Zheng, Dr. Lu Zhang, Nghia Nguyen) were a constant source of learning.

I am grateful to my thesis committee members, Dr. Wingning Li and Dr. Qinghua Li, for recognizing and approving the thesis. I would also like to thank everyone in the University of Arkansas' Department of Computer Science and Computer Engineering for their help and support. I cannot thank enough the WEKA community for tremendous help in figuring out the details of WEKA.

Finally, I like to express my profound gratitude to my family for their unfailing love, encouragement and support. To all other friends and relatives, who in one way or another shared their wisdom and support, thank you.

My research is supported by U.S. National Institute of Health (1R01GM103309), and I am grateful for this support.

Table of Contents

I. INTRODUCTION 1
A. Overview 1
B. Thesis Statement 2
C. Contributions 3
D. Organization of the Thesis 4
II. DIFFERENTIAL PRIVACY 5
A. Motivation 5
B. Introduction to Differential Privacy 6
C. Properties of Differential Privacy 8
D. Achieving Differential Privacy 8
E. Tools to Achieve Differential Privacy 12
F. Summary 14
III. WEKA 15
A. Introduction 15
B. Interfaces in WEKA 16
C. Anatomy of Source Code 20
D. Development in WEKA 22
E. WEKA Package 23
F. Development and Installation of DPWeka 24
G. Summary 26
IV. IMPLEMENTATION OF DIFFERENTIALLY PRIVATE STATISTICS 28
A. Introduction 28
B. Genome Wide Association Studies 29
C. Significance Statistics 31
D. Literature Review 32
E. Implementation in WEKA 33
F. Datasets 38
G. Evaluation 38
H. Summary 41
V. IMPLEMENTATION OF DIFFERENTIALLY PRIVATE LOGISTIC REGRESSION 42
A. Introduction 42

B. Logistic Regression 43
C. Literature Review 45
D. Implementation in WEKA 46
E. Datasets 50
F. Evaluation 51
G. Summary 52
VI. CONCLUSIONS AND FUTURE WORK 54
A. Conclusions 54
B. Future Work 56
VII. REFERENCES 57
VIII. APPENDIX 61
A. Algorithm to Release Significant Attributes Using Laplace Mechanism for χ^2 values 61
B. Algorithm to Release Significant Attributes Using Laplace Mechanism for p-values 61
C. Algorithm to Release Significant Attributes Using Exponential Mechanism for p-values 62
D. Cost Function of Logistic Regression 62
E. Algorithm to Build Logistic Regression Model Using Functional Mechanism 63
F. Algorithm to Build Logistic Regression Model Using Exponential Mechanism 64
G. Access DPWeka 65

List of Figures

Figure 1 Differential Privacy Layer for Data Mining 3
Figure 2 Sample and Aggregate Framework [33] 11
Figure 3 GUI Chooser Panel 17
Figure 4 Preprocess Panel in Explorer Interface 18
Figure 5 Menu for Filter Methods in Preprocess Panel 19
Figure 6 Property Form for AttributeSelection Method 19
Figure 7 WEKA Source Code Directory 21
Figure 8 Similarities in WEKA's Directory Structure and Graphical Interface 21
Figure 9 Contents of DPWeka Package 25
Figure 10 Excerpt from Description.props File 25
Figure 11 WEKA Package Manager Graphical Interface 26
Figure 12 WEKA Unofficial Package Installation Steps 26
Figure 13 Access PrivStats from Select Attributes Panel 34
Figure 14 Property Sheet of PrivateStatistics Method 35
Figure 15 Property Sheet of χ^2 values 36
Figure 16 Property Sheet of p-values 37
Figure 17 Evaluation of Laplace Mechanism for χ^2-values 39
Figure 18 Unperturbed Top p-values in Ascending Order 40
Figure 19 Evaluation of Exponential Mechanism for p-values 41
Figure 20 Logistic Mapping of Single Variable Input and Output 44
Figure 21 Access PrivClassifier from Classify Panel 46
Figure 22 Property Sheet of FunctionalMechanism 48
Figure 23 Property Sheet of EMGeneticAlgorithm 50
Figure 24 Evaluation of FunctionalMechanism for Varying Privacy Budget 52
Figure 25 Evaluation of EMGeneticAlgorithm for Varying Privacy Budget 52

Abbreviations

JAR: Java Archive

NIH: National Institutes of Health

PII: Personally Identifiable Information

SNP: Single Nucleotide Polymorphism

XML: eXtensible Markup Language

GWAS: Genome Wide Association Studies

WEKA: Waikato Environment of Knowledge Analysis

ARFF: Attribute Relation File Format

XRFF: XML Relation File Format

PINQ: Privacy Integrated Queries

IPUMS: Integrated Public Use Microdata Series

I. INTRODUCTION

A. Overview

The affordability of digital storage space has prompted organizations across several industries to collect and store users' information with the intention of unveiling interesting patterns within the data. The users' sensitive information present in these datasets can be misused by an ill-intentioned adversary. This fear of privacy breach can prevent people from providing true answers to the data collection systems, which in turn affects the systems' utility. However, with many companies collecting customers' data, an individual's information can be expected to be available in several independent sources. The information that can be obtained from external sources, including social media, alternate databases or even general observations, is called auxiliary information. Incidents such as re-identification of individuals from the "anonymized" Netflix dataset [1] and AOL's web search dataset [2] stand as classic examples to depict that simply anonymizing the data does not ensure privacy when auxiliary information is taken into account.

Additionally, to protect the privacy of the dataset, data curators may only release the models to make predictions, or to draw inferences. However, when the models are built on the sensitive dataset, privacy breach can occur. An adversary with auxiliary knowledge can infer the sensitive attributes of the individual with enough brute force attempts on the model [3].

Hence, methods that protect against privacy breach, even when auxiliary information is available, are necessary while working with sensitive data. This can be achieved by the differential privacy framework, which provides strong privacy guarantees by randomizing the results. Publications such as [4]–[18] indicate an actively growing interest in the differentially private methods for data mining. Some works proposed approaches to enforce differential

privacy at the initial stages of data mining, such as attribute selection [6]–[8] and dimensionality reduction [4], [5]. Some others such as [9]–[13], [16], concentrated on building differentially private learning schemes, including decision trees, classification, regression and deep learning. Others developed approaches to achieve differentially private diagnostic methods [14]. Some authors concentrated on achieving differential privacy in graph generation [17] and spectral graph analysis [18]. Works such as [13] have also proved that embracing a differentially private algorithm can yield utility comparable to non-private methods with fewer training samples, when the mechanism is carefully chosen.

Although differential privacy provides strong privacy guarantees, there are only a few platforms to support practical differentially private data mining. They employ specific differentially private mechanisms for limited applications of data mining. Additionally, popular data mining tools do not contain differentially private data mining algorithms. As a result, the cognizance of differentially private methods for analyzing sensitive data is currently limited outside the research community. We address this gap by implementing DPWeka, a package containing several differentially private algorithms for the open source machine learning tool, WEKA. DPWeka can be considered as a privacy preserving layer that provides access to the sensitive database through a query interface. This concept, as illustrated in Figure 1, is similar to Privacy Integrated Queries (PINQ) platform [19] and the privacy layer proposed in [13].

B. Thesis Statement

This thesis consists of two parts: (1) Study the various mechanisms to achieve differential privacy for data mining tasks and (2) Realize these mechanisms through DPWeka, a plugin designed to provide comprehensive differential privacy capabilities to the machine learning platform, WEKA.

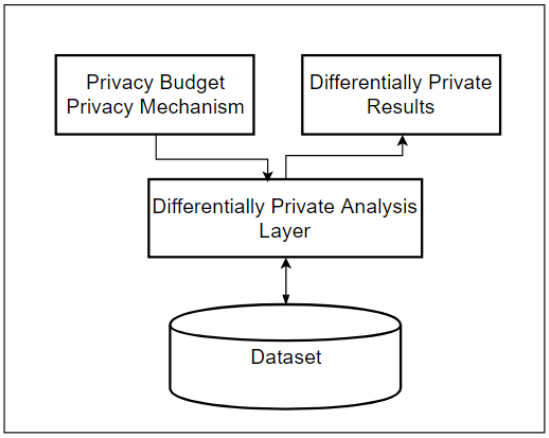

Figure 1 Differential Privacy Layer for Data Mining

C. Contributions

The main contribution of this work is the development of DPWeka, which can be used either independently or as an extension to a popular data mining tool, WEKA. DPWeka can be considered as an interface that allows users to query datasets in a differentially private manner. It is a package that provides comprehensive differential privacy capabilities to WEKA for practical data mining purposes. Currently, DPWeka includes a suite of differential privacy preserving algorithms which support data mining tasks such as attribute selection and classification. However, it can be easily extended to include other data mining tasks such as clustering and association rule mining. DPWeka has provisions for users to control various privacy and model parameters, such as privacy mechanism, privacy budget, and other algorithm specific variables. It allows users to perform privacy preserving data mining using a graphical interface, which does not require users to be programmers or privacy experts. It enforces privacy while providing users with wide range of implementation choices.

Currently, DPWeka consists of two components. The PrivStats component is designed to perform differentially private attribute selection. As an example, we have implemented differentially private methods to identify relevant attributes based on the significance statistics

such as χ^2 values, p-values, and odds ratio. The PrivClassifier component within DPWeka is designed to perform differentially private classification. As an example, we have implemented methods to perform logistic regression using various differential privacy mechanisms.

D. Organization of the Thesis

The remainder of this thesis is organized as follows:

- In Chapter II, we discuss differential privacy and its motivation. We also survey the mechanisms for realizing differential privacy in practice. We later discuss the tools to achieve differential privacy and compare them with DPWeka.
- In Chapter III, we explore WEKA and its features to perform end-to-end data mining. We then discuss the methods to build and configure new modular capabilities for WEKA. Finally, we discuss development of DPWeka and illustrate the steps to integrate it with WEKA.
- In Chapter IV, we study the motivations for privacy in attribute selection and examine differentially private methods to achieve it. As an example, we examine the motivations and methods to achieve privacy in Genetic Wide Association Studies (GWAS). We then discuss the implementation of some of the differentially private methods in PrivStats component of DPWeka. Further, we demonstrate their applicability on a genetic dataset.
- In Chapter V, we focus on achieving differential privacy in learning schemes. While considering logistic regression model as an example, we explore various methods to build differentially private regression models. We then discuss their implementation in PrivClassifier component of DPWeka and evaluate how well the methods preserve statistical utility while preserving privacy.
- In Chapter VI, we conclude the work and identify the areas for future work.

II. DIFFERENTIAL PRIVACY

A. Motivation

Data mining is performed to extract the underlying information of a dataset. On the other hand, data privacy is enforced to prevent an adversary from inferring anything significant about the individuals present in the dataset. Although these exercises seem to contradict one another, it is not quite so. The aim of data mining is to find the general trends and not directed at identifying any single individual's behavior. Data mining tasks generate models to emulate the commonly occurring patterns in the dataset. However, when sensitive data is involved, the models may become vulnerable to attacks and cause privacy leakage.

Anonymizing the sensitive data before constructing the model seems like an obvious solution to prevent privacy leaks. In fact, current guidelines enforce that data containing any of the listed categories of personally identifiable information (PII) must be appropriately redacted before being published for research purposes. In spite of such regulations, researchers were able to identify Massachusetts governor's personal health information. The redacted health data was re-identified by taking the voter registry data into account. Such linkage attacks highlight the limitation of anonymization methods when the adversary has auxiliary information [20].

Various methods including k-anonymity [20], l-diversity, [21] and t-closeness [22] have been proposed to combat linkage attacks for data publishing tasks. In these methods, samples containing similar values of sensitive data are grouped together and released only if the sample size is sufficiently large. However, an adversary with knowledge of all but one individual of the dataset can still infer the individual's data based on the query result [23]. Alternatively, query auditing methods can be used, in which every query presented to the database is evaluated. If a query response in combination with past responses is found to disclose private information, then

the query is refused. However, the denial of the request can then reveal information about the dataset. Moreover, for large datasets that are queried frequently, auditing every query may be computationally infeasible. These challenges call for a more robust definition of privacy that is not affected by the auxiliary information.

B. Introduction to Differential Privacy

Privacy is guaranteed if after querying the dataset, an adversary's posterior knowledge of any individual in a dataset is no more than his prior knowledge of the same individual. Differential privacy provides strong privacy guarantees for an individual who has participated in the dataset while allowing the data miner to learn general trends presented by the data. Differential privacy relies on the idea that privacy is ensured when a query mechanism is resilient to changes in even a single instance of the dataset. Formally, it is defined as

Definition 1 (ε-differential privacy [24]): A randomized mechanism $\mathcal{M}: D^n \rightarrow \mathbb{R}^d$ is said to be ε-differentially private, if for any two neighboring datasets D and D' differing by a single element, and for all the outcomes $S \subseteq \mathbb{R}^d$,

$$\Pr[\mathcal{M}(D) \in S] \leq e^{\varepsilon} \Pr[\mathcal{M}(D') \in S] \qquad (1)$$

Here, the output distribution of $\mathcal{M}$ (D) depends on the actual query output distribution. The parameter ε quantifies the privacy leak. Small value of ε indicates that the mechanism is more private. It implies that for all pairs of neighboring datasets, the differentially private mechanism's output distributions are approximately same. Due to similar output distributions, an adversary, who even has information about all but one individual of the dataset, cannot infer anything about the individual from the query output. The constraint (1) ensures that the randomized mechanism is insensitive to the presence of an individual, and hence provides opt-out option, i.e., a participant can choose to either participate or not participate in a data analysis

without significantly affecting the outcome of the analysis. This guarantees that a differentially private computation also obscures the information about the presence of an individual in the dataset. However, privacy guarantee does not mean that the participating individual is not affected by the decisions made using the query results. On the other hand, ensuring privacy means that the effect of the business decision on an individual present in the dataset is no different than on a similar individual who is not present in that dataset. However, it should be noted that differential privacy does not provide absolute privacy guarantees as any non-trivial utility implies compromised privacy [25].

A weaker privacy guarantee is given by (ε-δ)-differential privacy [26].

Definition 2: A randomized mechanism $\mathcal{M}: D^n \rightarrow \mathbb{R}^d$ is said to satisfy (ε -δ)-differentially privacy for two non-negative numbers ε and δ iff for all neighboring datasets where $d(D, D') = 1$, and all subset outcomes $S \subseteq \mathbb{R}^d$,

$$\Pr[\mathcal{M}(D) \in S] \leq \delta + e^{\varepsilon} \Pr[\mathcal{M}(D') \in S] \qquad (2)$$

Here, δ denotes the probability by which the mechanism's outputs vary by a factor of e^{ε} for any two neighboring datasets. In other words, an (ε - δ)-differentially private algorithm ensures that for neighboring datasets, the privacy loss is bound by ε with a probability of at least (1- δ) [26]. Lower values of δ signify greater confidence that the neighboring datasets vary only by a factor of e^{ε}. Conversely, for a sufficiently large δ value, the mechanism can breach privacy even at smaller ε values. Hence, δ is chosen heuristically to be $\delta \in o\left(\frac{1}{n}\right)$, where *n* indicates the size of the dataset [27].

Two datasets are neighboring datasets if they differ by a single entry. The difference can be due to the replacement of an entry with another value while maintaining the same data size among the neighbors. Another interpretation is that an entry is either added into or deleted from

the neighboring dataset, thus varying the dataset size. The difference in interpretation leads to the variation in calculations of sensitivity.

C. Properties of Differential Privacy

Differentially private mechanisms are immune to post-processing [26]. That is, processing a differentially private output using a data independent mechanism does not incur additional privacy loss. Formally,

Property 1: If $\mathcal{M}: D^n \rightarrow \mathbb{R}^d$ is an ε-differentially private mechanism, and $f: \mathbb{R}^d \rightarrow R'$ is an arbitrary randomized mapping, then their composition $f\ (\mathcal{M}(D^n))$ is ε-differentially private.

Differentially private mechanisms exhibit sequential composition [28]. When multiple differentially private mechanisms, each having its own privacy parameter, access the same database, then the privacy of the overall operation degrades equivalent to the sum of all privacy parameters.

Property 2: Given two randomized mechanisms, f_1 and f_2, which are $\varepsilon_1$and ε_2-differentially private respectively, a mechanism $f = g(f_1(D), f_2(D, f_1(D)),$ is $(\varepsilon_1 + \varepsilon_2)$-differentially private.

When the sequential property is applied repeatedly, the overall privacy budget of the resultant mechanism is the sum of privacy budgets consumed by each mechanism. In general, both the properties can be extended to (ε - δ)-differentially private algorithms.

D. Achieving Differential Privacy

Differential privacy describes the behavior of a mechanism and is not the algorithm itself. Nevertheless, a query can be made differentially private by randomizing it. However, while randomization preserves the privacy, it can also adversely affect the utility. Hence, although a data mining algorithm may have several mechanisms to achieve ε differential privacy, the

challenge is to find a mechanism that yields higher utility for smaller values of ε [26]. Some of the mechanisms to achieve differential privacy are discussed below.

Output perturbation methods add noise to the query output to preserve privacy. The added noise is such that it masks the impact any single individual can have. Hence, whenever the contribution of any individual is lost in the noise, their private information, that is not generally known, cannot be revealed. Output perturbation methods are often used for queries that yield numerical results, such as aggregate functions. For an output perturbation method, the amount of added noise is dependent on the sensitivity of the query. The sensitivity [26] of a function is the maximum value by which the function's output can vary for any pair of neighboring datasets. It is a measure of the largest change a single participant of the dataset can have on the query output.

Definition 3: For a function $f: D^n \rightarrow \mathbb{R}^d$, sensitivity is defined as

$$\Delta f = \max ||f(D) - f(D')|| \qquad (3)$$

where D and D' are neighboring datasets differing by one tuple and $||.||$ is the ℓ_1 norm.

Laplace mechanism [28], the most popular of noise adding mechanisms, involves adding independently generated noise sampled from Laplace distribution with mean 0 and scale $\Delta f/\varepsilon$, to maintain ε-differential privacy. The noise distribution is then given by

$$f(x \mid \varepsilon, \Delta f) = \frac{\varepsilon}{2\Delta f} exp\left(-\frac{\varepsilon|x|}{\Delta f}\right) \qquad (4)$$

The magnitude of the noise added depends on the desired privacy levels and the query type. A query with higher sensitivity requires more noise to attain same privacy levels as a query with lower sensitivity.

Adding noise sampled from Gaussian distribution may need the privacy definitions to be relaxed to obtain similar utility [29]. Geometric mechanism [30], a discrete variant of Laplace mechanism, can be used for count queries. An α-differentially private geometric mechanism adds random noise based on a two-sided geometric distribution centered over 0. Its probability distribution function is given by

$$p(x|\,\alpha) = \frac{1-\alpha}{1+\alpha}\,\alpha^{|x|} \tag{5}$$

where $\alpha \in [0, 1]$ indicates the privacy level.

Another noise adding mechanism, the staircase mechanism [31], [32], can be viewed as geometric mixture of random variables and adds noise with probability distribution given by

$$p_\gamma(x|\varepsilon,\Delta) = \frac{1-\alpha}{2\,\Delta\,\sqrt{\alpha}}\,e^{-\varepsilon(k+[x]_\gamma)} \tag{6}$$

$$[x]_\gamma = \begin{cases} 0, & |x| \in [k\Delta,\ (k+\gamma)\Delta) \\ 1, & |x| \in [(k+\gamma)\Delta,\ (k+1)\Delta] \end{cases} \tag{7}$$

where $k \in \mathbb{N}$, $\gamma \in [0,1]$ controls the shape of the staircase and $[x]_\gamma$ is the rounding function. For larger values of ε, i.e., at medium to low privacy regimes, staircase mechanism performs significantly better than Laplace mechanism [31], [32].

The output perturbation mechanisms consider only the query function and its sensitivity to derive the degree of noise. However, for certain queries, such as median calculation and cluster center calculation, the large noise magnitude due to the global sensitivity can render the query results useless. For such queries, Nissim et. al [33] proposed sample and aggregate framework, which adds query-based noise as well as instance based noise to the result being released. To ensure that the noise magnitude does not adversely affect the utility, smooth sensitivity, which smoothens the local sensitivity bounds, is considered. The sample and aggregate framework

treats the query function as a black box and uses a smooth sensitivity framework for ensuring differential privacy. The mechanism consists of two steps. In the sample step, the query, *f*, is evaluated on random samples that are the subset of the private dataset. The query results of the subsets are then combined by an aggregation function, assuming that the desired result can be measured well with samples. The output of the aggregation function, $\bar{f}$, is then perturbed with the noise calibrated to smooth sensitivity of aggregate function (Figure 2) [33]. The main challenge in this framework is to find an efficient aggregation function.

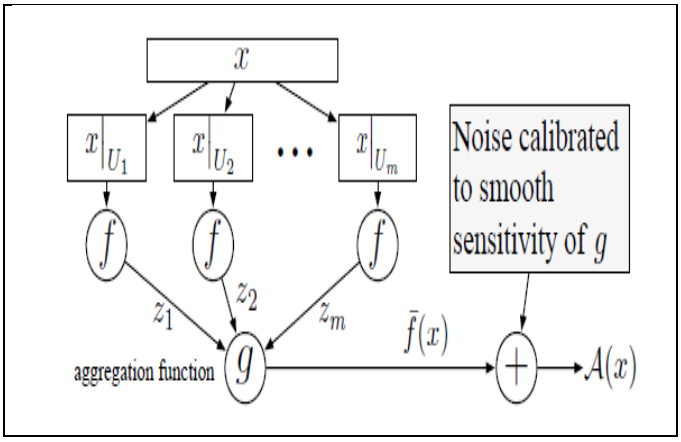

Figure 2 Sample and Aggregate Framework [33]

The output perturbation and sample and aggregate framework mechanisms are employed when the query results are numerical and continuous values. However, for queries yielding non-continuous output, adding noise destroys the value of the output. Exponential mechanism [34] can be employed for queries whose output space is non-continuous. If R is the set of all possible responses to a query on dataset D, then the exponential mechanism involves defining a quality score function q, which assigns scores to each possible response $r \in R$. The optimal scoring function is the one that assigns the scores proportional to the probability of their being the correct solution to the query. Exponential mechanism, $\xi^{\varepsilon}_{q,\Delta q}$, then has the output distribution

$$\Pr\left[\xi^{\varepsilon}_{q,\Delta q}(D) = r\right] \propto e^{q(r,D)\varepsilon/2\Delta q} \quad (8)$$

where Δq is the sensitivity of the scoring function given by

$$\Delta q = max\ ||q(D) - q(D')|| \quad (9)$$

where D and D' are neighboring datasets.

Since the probability assigned reduces exponentially for each less probable result, the method is called exponential mechanism [26]. This method has been applied for building differentially private models including support vector machines [35], auction mechanisms [36] and ID3 decision trees [13]. In some cases, such as differentially private ID3 [13], the private mechanism was shown to perform better than the non-private mechanism for fewer training tuples.

Besides aggregation queries, data mining procedures also comprise model generation techniques for prediction and inference purposes. Objective perturbation method [9] can be used to release the model parameters within a differentially private framework. In this method, noise is added to the objective function, which is then optimized to obtain the private model parameters. This method has been used to achieve differentially private logistic regression and linear regression [10], [12], [15].

E. Tools to Achieve Differential Privacy

Systems such as PINQ [37], Airavat [38], and GUPT [39] have been developed to realize differential privacy. Although the motivation behind building the systems is same, they are designed to enforce differential privacy using different frameworks. Privacy Integrated Queries (PINQ) is an application program interface (API) built upon C#'s LINQ (Language Integrated Query). It contains differentially private functional blocks, such as *Where*, *Groupby*, *Join*, and *Partition* transformations, which provide basic data access functionalities. These blocks can be used by a data miner to develop differentially private programs. The private data is wrapped in a

PINQueriable object, which acts as the privacy layer to access the data. The PINQueriable object evaluates the amount of privacy budget available after each operation involving the private data and denies further access once the budget is exhausted. Apart from being built only for Windows platform, PINQ cannot calculate the amount of privacy budget required for the entire program in advance, which may cause the privacy budget to be exhausted before the program is completed.

Airavat executes unmodified programs in Map-Reduce programming paradigm with an untrusted mapper computation and a differentially private reduce computation. To estimate the sensitivity and the amount of random noise to be added at the reducer, the range of the mapper outputs must be declared initially. Since the range of mapper outputs can be estimated only for specific data analysis tasks, Airavat supports only limited types of data mining operations.

GUPT enforces differential privacy using sample and aggregate framework. To reduce the amount of perturbation at the aggregate step, GUPT uses an aging model of data sensitivity to select the sample size. Moreover, GUPT automatically distributes the privacy budget among the queries of a data mining task to maximize the overall accuracy of the task. However, GUPT assumes that the dimensions of the query results are known in advance, which may not always hold true. Moreover, when the query range is non-numerical, adding noise to the aggregated result will destroy the results' value. Due to these limitations, the range of data mining operations supported by GUPT are limited.

Our DPWeka provides a platform for users to select the desired differentially private mechanism for a data mining task. Unlike the other platforms that enforce privacy at the programming level or at the aggregation step, DPWeka enforces privacy within the data mining task. It also supports more complex data mining tasks than simple aggregate functions. DPWeka allows users to control privacy budget and other model parameters. It also offers facilities to

compare the performance of various mechanisms simultaneously. As compared to PINQ and Airavat, DPWeka does not require data miners to be privacy experts or programmers.

F. Summary

To summarize, differential privacy framework provides mathematical guarantees that an adversary cannot infer the presence/absence of an individual in the dataset. Mechanisms such as Laplace mechanism, exponential mechanism, objective function perturbation, and sample and aggregate framework can be used to design a differentially private algorithm. However, not all mechanisms can ensure high accuracy for every data analysis algorithm. The challenge is to identify a differentially private mechanism which maintains the utility of the algorithm. Several tools have been developed to achieve differential privacy. However, they are designed to enforce differential privacy using specific mechanisms. Moreover, some of these tools require users to have prior programming knowledge. On the other hand, DPWeka is designed to support a wide range of differential privacy mechanisms and does not require users to be programmers or privacy experts.

III. WEKA

A. Introduction

Data mining is an elaborate and experimental process of finding patterns within data. It involves performing one or many steps, including cleaning up the data using domain specific metrics, performing data transformation and feature selection, applying learning schemes and evaluating them to find suitable parameters. With the evolution in the nature of the data, advanced analytical techniques are being developed constantly. This in turn has led to the emergence of an assortment of tools designed to perform either a specific data mining task or comprehensive data mining. These tools are available in a wide range of formats including licensed software such as SAS; libraries for existing programming languages, such as Pandas and Deeplearning4j; and open source toolkits, such as WEKA. In this chapter, we discuss WEKA and its exhaustive range of data mining features. We also examine the techniques to develop WEKA packages for installing new features into WEKA. Finally, we describe the elements of DPWeka package and illustrate the steps to integrate DPWeka with WEKA.

The WEKA workbench [40] is a collection of machine learning and data preprocessing tools, distributed under GNU General Public License. WEKA, which is an acronym for *Waikato Environment for Knowledge Analysis*, was developed at the University of Waikato in New Zealand. It is written in Java and can run on Linux, Windows, and Macintosh operating systems. The current stable version, 3.8.0, is compatible with Java 1.7 or later [41].

WEKA provides comprehensive support for the entire data mining process, including methods for preparing the input data using data transformation and preprocessing, analyzing the data using learning schemes, and visualizing the data. All these features are accessed by an interactive interface, which does not require users to possess any programming knowledge. The

tunable parameters of the methods are presented as forms to be filled by users. For ease of use, the default values are already filled, and users can examine the methods by varying the values. Advanced users can access the learning schemes and other preprocessing methods available in WEKA to reduce their programming effort when developing their own algorithms.

B. Interfaces in WEKA

WEKA's initial screen, the *GUI Chooser* panel, serves as a gateway to access its five interfaces: *Explorer*, *Experimenter*, *Knowledge Flow*, *Workbench*, and *SimpleCLI* as shown in Figure 3. The first four are interactive interfaces, which contain menus to select the desired methods. The tunable parameters of the methods can be assigned values through a property sheet or *object editor*.

To be processed by any of the interfaces, data must be in the form of a single relational table. Data can be read from a file on the local machine, or from a remote location, or generated from a database query. Each row of the data is referred as an instance and each column as an attribute. The attributes of the data must belong to numeric, nominal, string, or date types, and the missing values must be denoted by '?'. Although WEKA can process data in Attribute Relation File Format (ARFF), facilities exist to convert other file formats to ARFF format. This support exists for files of CSV format, C4.5 format, LIBSVM format, XML Relation File Format (XRFF), JSON based ARFF format, and MATLAB files [41].

The five interfaces of WEKA are designed to accommodate different styles of data mining process. The *Workbench* interface combines all the interfaces into a single application, whereas the command line interface, *SimpleCLI*, is used to directly access WEKA's basic functionalities by calling its required Java classes and providing appropriate arguments.

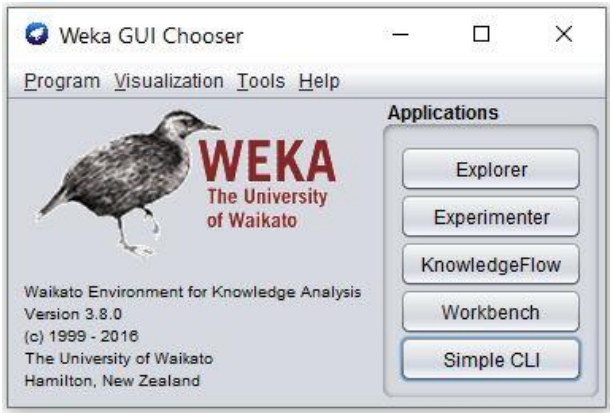

Figure 3 GUI Chooser Panel

The *Experimenter* interface has facilities to identify the optimal learning methods and parameters by setting up experiments involving multiple learning methods and datasets simultaneously. Once the desired set up is arranged, the large scale experiments can run without requiring any further human intervention. Their results can be saved as ARFF files which can be analyzed at later time. The *Experimenter* interface also contains facilities to distribute the computing loads across multiple machines, thus reducing the run time.

The *Knowledge Flow* interface is used to set up step-by-step workflow for the data to pass through. It presents facilities to define a data flow procedure by connecting various data mining methods. The procedure can be saved as model and used whenever required. It also allows for incremental learning, provided that the methods have capabilities to process incremental data. Thus, this interface can be used for processing streaming data and large data files that cannot fit in the memory.

WEKA's main graphical interface, the *Explorer*, has panels corresponding to various data mining tasks supported by WEKA, such as preprocessing, classification and regression, clustering, attribute selection, association rule mining, and data visualization (Figure 4). The panels can be accessed by selecting the appropriate tabs. Each panel contains menus to select the

desired methods (Figure 5). Furthermore, each method has property form (or *object editor*) to assign values to method specific parameters (Figure 6). The facilities provided by the panels can also be accessed from other interfaces. Therefore, although DPWeka is developed with the intentions of using in *Explorer* interface, DPWeka can also be accessed by other interfaces.

In the initial screen of *Explorer* interface, the *Preprocess* panel, appropriate buttons can be clicked to load data files either located on the local machine, from a website, or from a database. The *Preprocess* panel provides access to data preprocessing algorithms including imputation, normalizing, randomizing, resampling, and removing selected instances and attributes, which are all bundled as *Filters*. A desired filter can be selected from the menu and its tuning parameters can be modified as required. The transformed data can then be saved and/or used for further data mining activities such as classification, clustering, or association analysis. Provisions are also available for generating artificial data suitable for classification, regression, and clustering purposes.

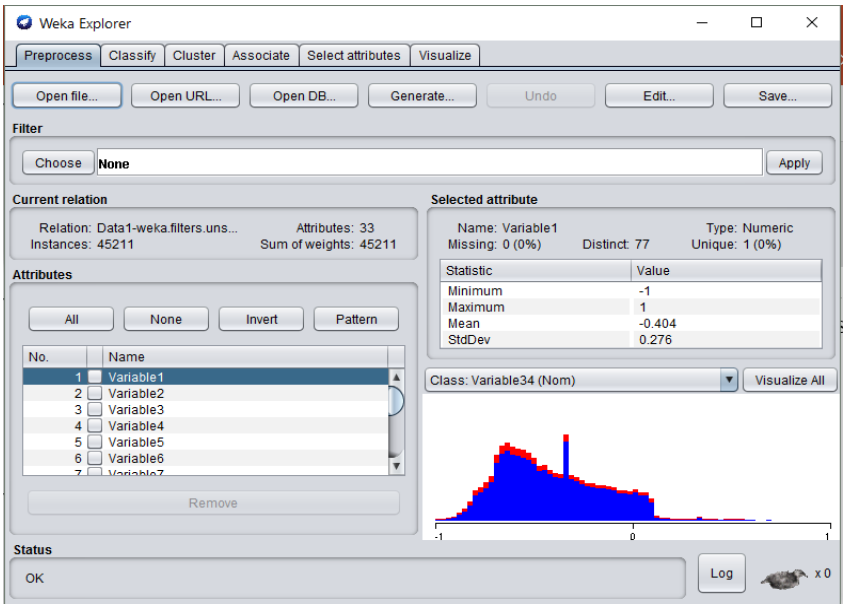

Figure 4 Preprocess Panel in Explorer Interface

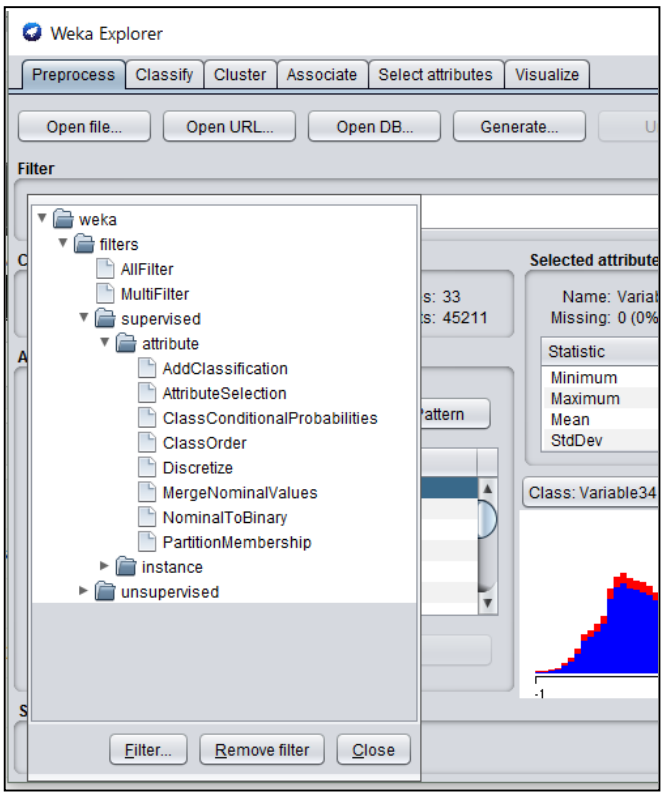

Figure 5 Menu for Filter Methods in Preprocess Panel

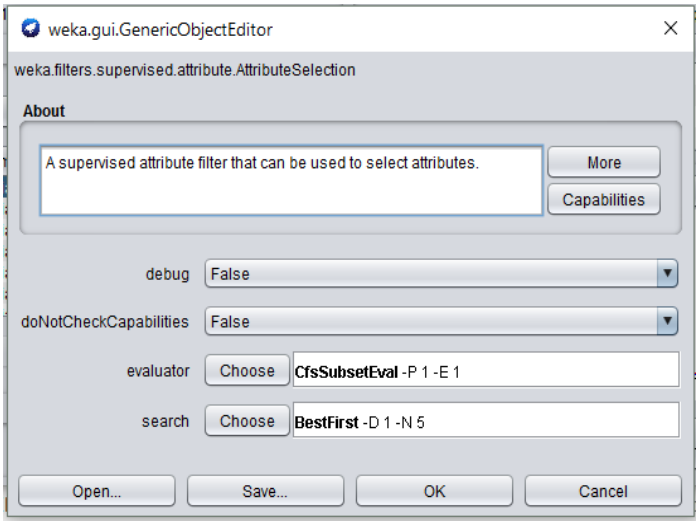

Figure 6 Property Form for AttributeSelection Method

The *Classify* and *Cluster* panels provide options to choose from various classification and clustering algorithms respectively. In *Classify* panel, various classification methods such as rule-based, tree-based, Bayes, regressions, and meta classifiers are available from the drop-down

menu. Similarly, in *Cluster* panel, methods including k-means, hierarchical clustering, and canopy clustering are available from the menu for unsupervised learning purposes. For each of the methods, users can assign method-specific parameters as well as select the general settings, such as training and testing data files, percentage split, and cross validation values.

The *Select Attributes* panel offers methods to evaluate and identify the most relevant attributes based on characteristics such as principle components, subset evaluations, correlation analysis and gain ratio. Furthermore, features to rank or select the best attributes are also available.

The *Associate* panel has methods for association rule learning. Finally, the *Visualize* panel can be used to visualize the data and the results of learning methods.

C. Anatomy of Source Code

WEKA heavily adopts Object Oriented Programming (OOP) concepts of Java in its implementation. Algorithms belonging to a specific data mining task are implemented within the same Java package (a Java package is physically represented as a folder). Hence, data mining algorithms associated with each panel of the *Explorer* interface are implemented in their respective Java packages as shown in Figure 7. Moreover, the algorithms adopting similar techniques to solve a task may be further grouped into subpackages. The directory structure seen within the packages is reflected in WEKA's graphical interface. Figure 8 depicts the similarities in the hierarchy within WEKA's physical directory structure and the graphical interface for classifiers. For each data mining task, WEKA provides an abstract class. Every algorithm inherits from its task specific abstract class and then implements the available abstract methods. In addition, the algorithms also import and inherit from classes and interfaces of *weka.core* package, which provides commonly used features for data mining tasks supported by WEKA.

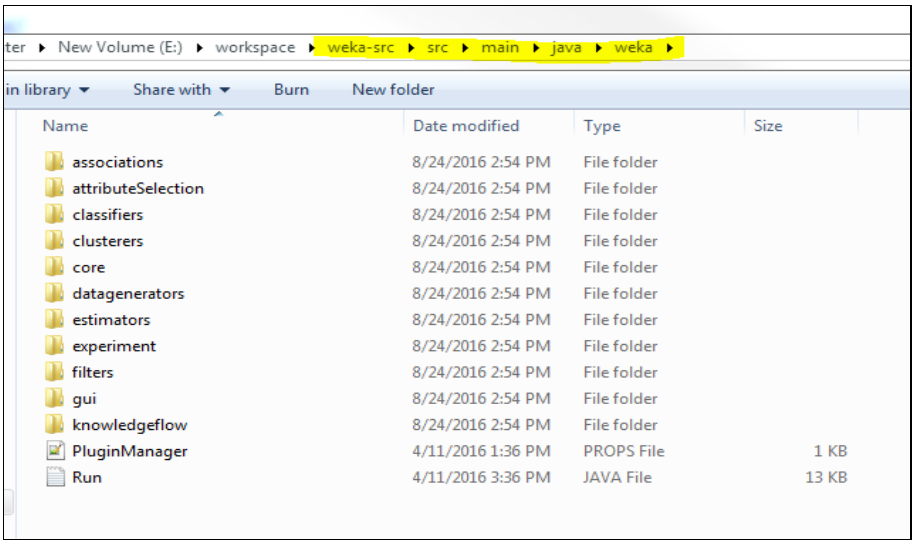

Figure 7 WEKA Source Code Directory

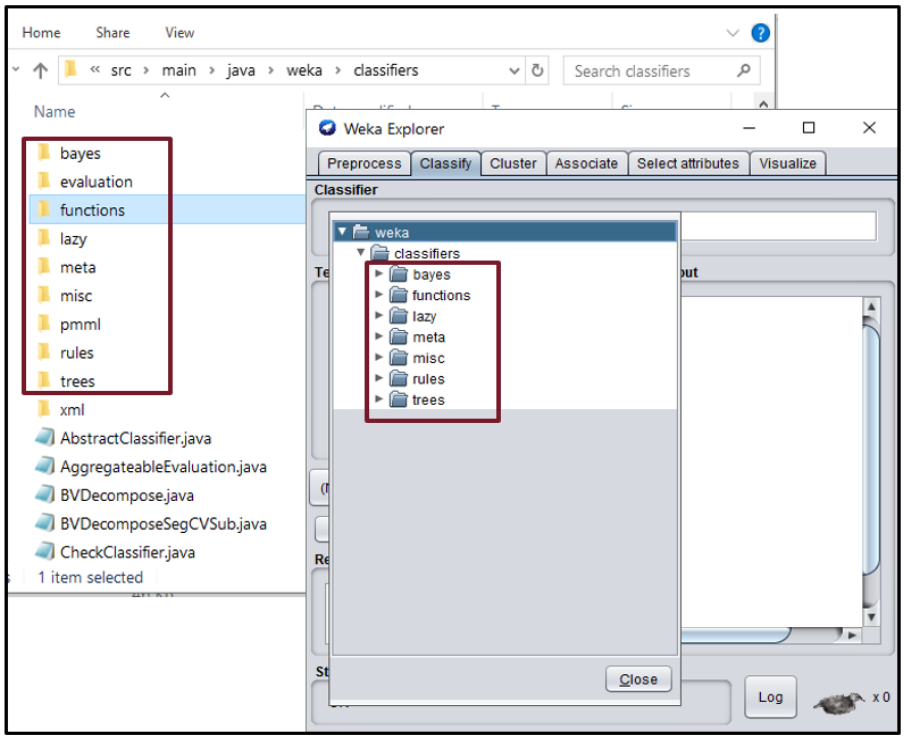

Figure 8 Similarities in WEKA's Directory Structure and Graphical Interface

For example, all the classification algorithms are available in *weka.classifiers* package. Further, the classification algorithms solved using different techniques are implemented in different subpackages within the *weka.classifiers* package. For instance, an objective function

based classification such as logistic regression is implemented in *weka.classifiers.functions* package. Moreover, all the classifiers inherit from *weka.classifiers.AbstractClassifier* abstract class.

Similarly, all the algorithms for performing attribute selection are available in *weka.attributeSelection* package. Since attribute selection involves evaluating the attributes and then selecting them, two abstract classes are provided by WEKA. Algorithms for calculating the metrics inherit from *weka.attributeSelection.AttributeEvaluator* abstract class, while the algorithms for selection inherit from *weka.attributeSelection.ASSearch* abstract class. These algorithms can also inherit from appropriate Java interfaces provided by WEKA.

Additionally, the features that are specific to *Knowledge Flow* and *Experimenter* interfaces are implemented in *weka.knowledgeflow* and *weka.experiment* packages. Furthermore, WEKA's graphical user interface features, such as click, change of tabs, and select actions, are implemented in *weka.gui* package.

D. Development in WEKA

WEKA can be used for performing data mining tasks by invoking appropriate methods from its graphical interface or command line. This approach does not require users to have any programming knowledge. WEKA's data mining algorithms can also be used as components of a development project by including WEKA's source files, *weka.jar* and *weka-src.jar*, into the project library. In order to contribute an algorithm that is not available in WEKA's collection, the algorithm's implementation design must conform to the general design structure of WEKA. The algorithm must inherit from WEKA's abstract class for the data mining task that the algorithm aims to solve. It may also inherit from WEKA's Java interfaces for additional features that are applicable to the algorithm. For example, a new classifier must extend from the abstract

class, *weka.classifier.AbstractClassifier*. The abstract class has abstract methods, such as *getCapabilities()* for defining the scope of the classifier, *buildClassifier()* for describing the procedure to build the model, and *classifyInstance()* for defining the procedure to make predictions. The new classifier must update the implementation details for these methods as per its algorithm. The classifier can optionally inherit from interfaces, such as *weka.core.TechnicalInformationHandler , weka.classifier.Randomizable,* and *weka.core.OptionHandler*. The classifier inheriting from *Randomizable* interface defines the method to randomize the data before constructing the model. Similarly, the classifier can define the parameters for its properties form using the *OptionHandler* interface and display the bibliographic references using the *TechnicalInformationHandler* interface.

For other data mining tasks, such as attribute selection, clustering, and data preprocessing, new algorithms can be developed similarly by inheriting from the task-specific classes and relevant interfaces.

E. WEKA Package

The newly implemented algorithm must be deployed as a WEKA package to be integrated with WEKA. A WEKA package is a zipped archive containing the implementation's Java binaries compiled into JAR (Java Archive) file, *Description.props* file, and other property files. The package, which can optionally contain source files and other user documentation files, should unpack to the current active directory. The *Description.props* file contains the metadata of the package, such as the package name, version, brief description and category of the implementation, maintainer's name and contact, and the link for the availability of the source code. For the algorithm to be accessible from the WEKA menus (Figure 5), the *GUIObjectEditor.props* file must be updated with each Java package name and included in the

WEKA package. Further, the algorithm can be made available as an option in the WEKA property forms' (Figure 6), by adding the algorithm's class or its parent class to the *GUIEditors.props* file, which must then be included in the WEKA package.

The package can be officially made available to the community by supplying the *Description.props* file to the WEKA admin team, which will deploy the package on WEKA's central package repository after performing tests on the package. Alternatively, authors can make the package available unofficially and avoid the wait time, by uploading the package on public repositories, such as GitHub, for the community to download. Users can then integrate the WEKA package through WEKA's *Package Manager* interface, which is accessed from the *Tools* menu in *GUI Chooser panel.* Multiple Java packages can be bundled under the same WEKA package for the ease of deployment, making a WEKA package different from the regular Java package.

F. Development and Installation of DPWeka

DPWeka is designed to provide comprehensive differential privacy capabilities for end-to-end data mining in WEKA. Figure 9 shows the contents of our DPWeka package. The package contains the *DPWEKA.jar* file, which is the archive of the application; the property files; the source code; and the user instruction *README.md* file. An excerpt of *Description.props* file containing the metadata of DPWeka is shown in Figure 10. The *GUIObjectEditor.props* file contains an entry for *DPWeka.PrivStats* package under *weka.attributeSelection.ASEvaluation* and an entry for *DPWeka.PrivClassifier.Logistic* package under *weka.classifiers.Classifier*. Similarly, to make the various metrics available in the properties form, the *GUIEditors.props* file contains an entry *DPWeka.PrivStats.CalculateStats* mapping to *weka.gui.GenericObjectEditor*. Since DPWeka is currently being deployed as an unofficial package, it can be installed by

clicking the button at the top right corner of the package manager and then browsing to the local folder path containing the package or by providing the URL of the package. After its successful installation, the features of the package become available upon the restart of WEKA. The steps for installation are shown in Figure 11 and Figure 12.

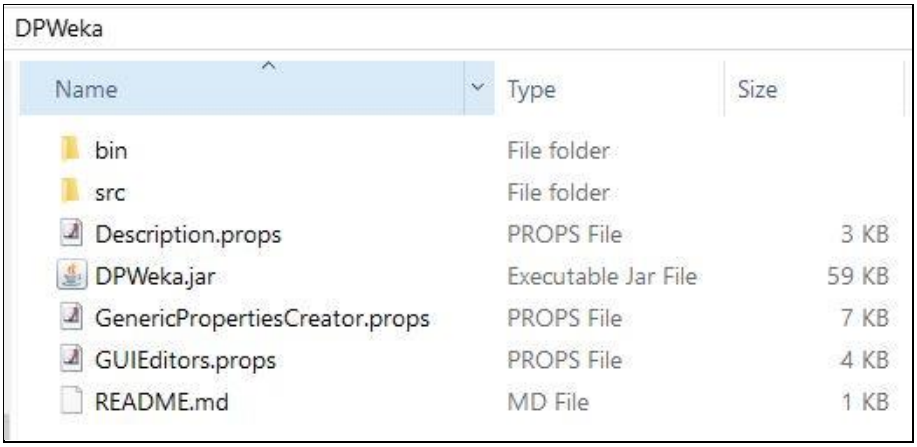

Figure 9 Contents of DPWeka Package

```
Date=2017-02-01
↓
# Title (required)
Title=Private Classifier and Attribute Selection demo
↓
# Category (recommended)
Category= Classifier, Attribute Selection
↓
# Author (required)
Author=Srinidhi Katla <skatla@email.uark.edu>
↓
# Maintainer (required)
Maintainer=Srinidhi Katla <skatla@email.uark.edu>
↓
# License (required)
License=GPL 2.0|Mozilla
↓
# Description (required)
Description=Differential Privacy demo
↓
# Package URL for obtaining the package archive (required)
PackageURL=https://bitbucket.org/
↓
# URL for further information
URL=https://bitbucket.org/
↓
↓
↓
# Dependencies (format: packageName (equality/inequality version_number)
Depends=weka (>=3.7.1),
```

Figure 10 Excerpt from Description.props File

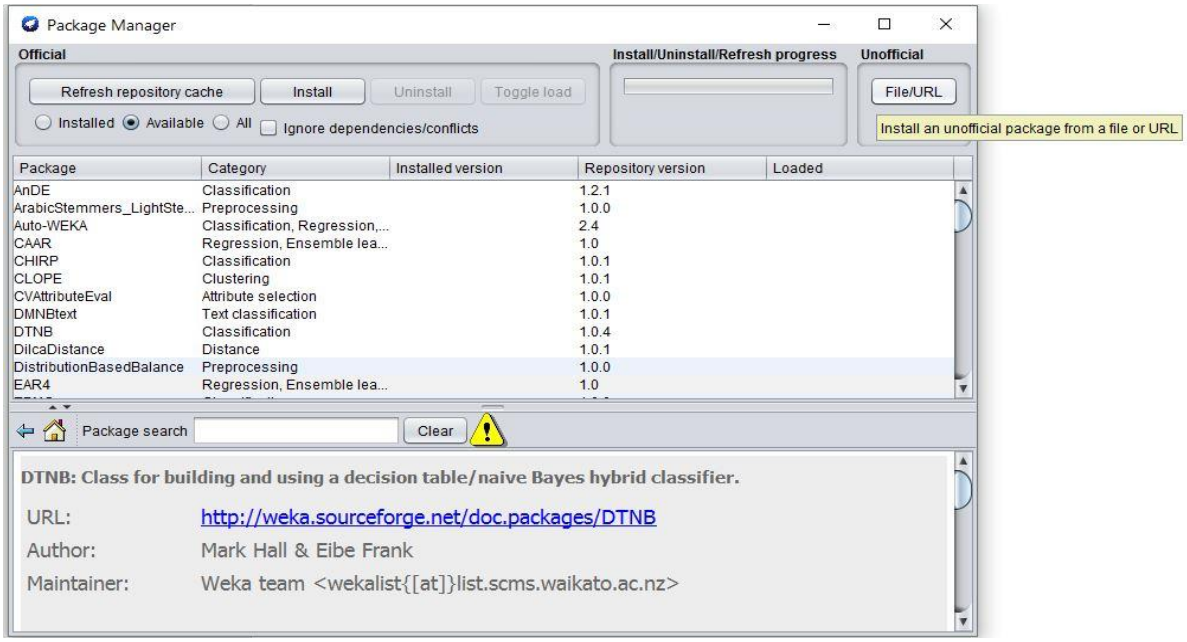

Figure 11 WEKA Package Manager Graphical Interface

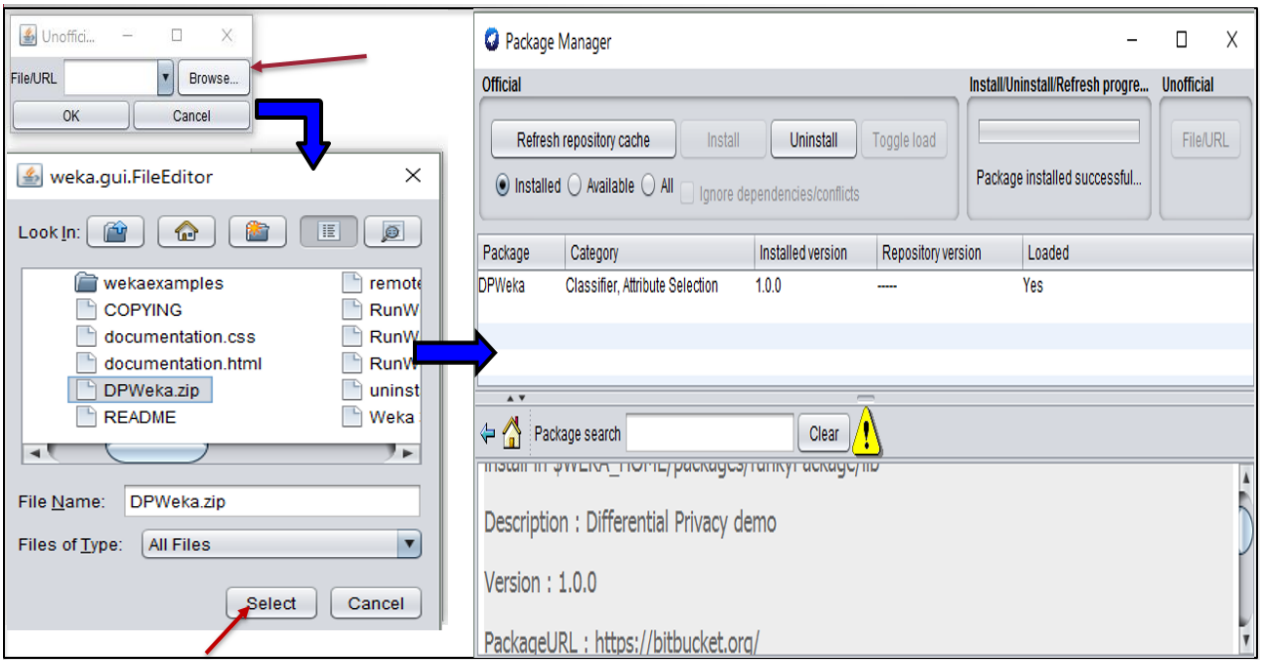

Figure 12 WEKA Unofficial Package Installation Steps

G. Summary

To summarize, WEKA is an open-source, comprehensive machine learning toolkit implemented in Java. New features can be deployed into WEKA in the form of plugins, which

are also called WEKA packages. However, for successful integration with WEKA, the implementation details of the new algorithms must adhere to WEKA's design. Then, the application is compiled into a JAR file. The zipped archive of the JAR file along with the property files is the WEKA package. The package is then imported from the *Package Manager* graphical interface. DPWeka is deployed as one such package containing methods belonging to *Attribute Selector* and *Classify* categories.

IV. IMPLEMENTATION OF DIFFERENTIALLY PRIVATE STATISTICS

A. Introduction

The data collected for data mining purposes may contain redundant and irrelevant attributes, which impact the accuracy of the prediction models by introducing noise. Hence, selecting relevant attributes is imperative to building more accurate models. Additionally, when processing high dimensional data, employing attribute selection methods avoids the curse of dimensionality, reduces the training time, and avoids overfitting. Based on the selection methods and learning algorithms used in the data mining process, attribute selection can be performed using filter method, wrapper method, or embedded method. In the filter method, general metrics such as association and correlation are used to select the relevant attributes. The wrapper method is used for evaluating subset of attributes by detecting possible interactions between them. In the embedded method, the learning algorithm performs the selection and classification simultaneously.

For high dimensional datasets, filter methods are used to identify relevant attributes. Typically, metrics of independence tests such as χ^2-values and p-values, correlation statistics such as Pearson r correlation and Spearman rank correlation, and association statistics such as odds ratio, are used to identify the attributes related to the predicted variable. To account for the reproducibility, the aggregate values (metric values) of the relevant attributes are released. However, these values may be used by an adversary to identify the entities of the dataset, leading to privacy breach.

In the following sections, we consider genetic data analysis as an example and study the privacy violations due to published aggregate values. Furthermore, we examine various differentially private algorithms for publishing the aggregate values. We then discuss the

implementation details of some of these algorithms within PrivStats component of DPWeka and finally evaluate them on a genetic dataset.

B. Genome Wide Association Studies

Genome Wide Association Studies (GWAS) are performed to identify the associations between genetic variants and traits in organisms. The human genome is encoded as 3 billion DNA (Deoxyribonucleic acid) base pairs within 23 chromosome pairs in the cell nuclei. Genome is the genetic sequence of an organism containing information about its growth, development, and health. In fact, it is a sequence of protein molecules, each of which could be made up of one of the four chemical bases: Cytosine (C), Adenine (A), Guanine (G) or Thymine (T), also called allele. The genome sequence varies by less than 0.1% between any two persons in the world. The smallest possible variation involving just one allele is called Single Nucleotide Polymorphism (SNP). Generally, the variation is in the form of the presence of an alternate allele at the SNP location. The allele that occurs in minority of the population is minor allele, while the one that is more frequently prevalent is called major allele. Scientists believe that these polymorphisms affect the characteristics in terms of appearance, personality, and physiology, and their study may reveal interesting facts about organisms [42][43].

GWAS is a type of case-control study, which examines the associations between the SNP variations (genotypes) and traits, characteristics, or diseases (collectively called phenotypes). A case-control study compares the subjects exhibiting the trait condition (called case group) with the subjects not exhibiting the condition (called control group). The applications of these studies range from genome mapping to screening for genetic diseases to forensic technologies, such as DNA fingerprinting diseases. Additionally, the genetic variations are thought to contribute to complex traits such as personality, weight, body mass index, and susceptibility to cancer and

heart diseases. Identifying the associated SNPs could provide better understanding in developing effective treatments for such conditions.

The progress in genotyping technology and the reduced cost of sequencing has motivated many researchers to contribute to these studies [43]. For each of the studies, the aggregate statistical values such as minor allele frequencies, p-values, and χ^2 statistics for all the SNP-phenotype combinations were released. Due to the sensitive nature of genome data, policies for data sharing, such as anonymizing, were enforced. In spite of the privacy measures, researches proved that it is not only possible to re-identify the redacted subjects, but also do it with a surprisingly small number of SNPs [44]. In 2008, Homer et al. [45] devised an attack where an adversary with auxiliary knowledge about the target's genetic profile would be able to determine if the target belonged to the case group or control group using the aggregated frequencies. This attack approach compares the target's allele with the minor allele frequency distributions, and uses T-test to determine the target's group. In response, National Institutes of Health (NIH) limited the access to the statistics of only a few significant SNPs in GWAS catalog for public use. Any research group now interested in accessing all the details has to undergo tedious bureaucratic processes explaining the committee of their altruistic intentions in conducting experiments with the genome data. Only after the committee is convinced that the data will not be abused, the research group can access it [46]. Moreover, [47]–[49] studied methods to construct Bayesian networks for inference using the publicly available GWAS statistics and evaluated the potential risk of the released GWAS statistics on individual privacy. We refer to [50] to survey the risks and protection methods of human genetic privacy.

C. Significance Statistics

In GWAS analysis, χ^2 test of significance is performed to identify the SNPs that are associated with a phenotype. χ^2 tests are designed to verify the independence of the variables by comparing their observed distribution with expected distribution under the independence assumption. If the distributions do not match, then the null hypothesis is rejected. In GWAS context, the null hypothesis H_0, is that the SNP is independent of the phenotype, and the alternate hypothesis H_a, is that the SNP is associated with the phenotype. The match in the distributions is quantitatively determined using the p-value and a predefined significance level. The significance level, denoted by α, indicates the probability of rejecting the null hypothesis when it is true. The p-value is the probability of obtaining an effect at least as extreme as the sample data, assuming that the null hypothesis is true. Thus, if the p-value for the sample is less than the significance level, the null hypothesis can be rejected and an alternate hypothesis can be entertained. Otherwise, the test is considered inconclusive.

The p-values for χ^2 tests are calculated based on the degree of freedom (*df*) and χ^2 values, which are calculated using the observed and expected distributions of the attributes whose association is being determined. A contingency table is used to display the observed distribution of the attributes and represents the first attribute as rows and the second attribute as columns. Each cell of the expected distribution table is then calculated by

$$Expected = \frac{rowtotal * columntotal}{N} \quad (10)$$

where $rowtotal$ is the sum of elements in the cell's corresponding row in the observed distribution (contingency table), $columntotal$ is the sum of elements in the cell's corresponding column in the observed distribution, and N represents the total number of records.

The degree of freedom (*df*) is then given by equation (11), and the χ^2 value is calculated using equation (12).

$$df = (number\ of\ \ rows - 1)(number\ of\ columns - 1) \quad (11)$$

$$\chi^2 = \sum_i \frac{(Observed - Expected)^2}{Expected} \quad (12)$$

Odds ratio (OR) is used to determine the strength of association between two attributes. In the context of GWAS, OR is employed to determine if a SNP is associated with the trait. When the SNP allele distribution is of the form given in Table 1, OR is given by equation (13)

	Case	Control	Total
Allele 1	a	b	$a + b$
Allele 2	c	d	$c + d$

Table 1 2x2 Contingency Table for SNP

$$Odds\ Ratio\ (OR) = \frac{Odds\ that\ allele\ 1\ occurs\ in\ a\ case}{Odds\ that\ allele\ 2\ occurs\ in\ a\ case} = \frac{(a/b)}{(c/d)} = \frac{ad}{bc} \quad (13)$$

OR = 1 indicates that there is no association between the SNP and the trait, whereas OR >1 indicates that the risk of the trait occurring is higher if allele 1 occurs while $OR < 1$ indicates that the risk of the trait is higher if allele 2 occurs.

D. Literature Review

NIH's decision to limit the access to genome data motivated researchers to formulate methods for sharing the aggregate values without violating privacy [6]–[8], [51]–[55]. Some of these works developed methods highly motivated by differential privacy.

Fienberg et al. [6] proposed methods to release the minor allele frequencies, p-values, and χ^2 values in an ε-differentially private manner by employing output perturbation mechanism. Their

approach is to add randomly generated Laplace noise of scale $2M/\epsilon$ to minor allele frequencies, $4M\epsilon(\frac{4N}{N+2})$ to χ^2 values, and $4M\epsilon\left(e^{-\frac{2}{3}}\right)$ to p-values, where M represents the number of significant SNPs to be released and N indicates the number of records (population size). The M most relevant SNPs are then selected based on the perturbed values. However, they assume that the size of the case and control groups are same which may not always hold true in the real world.

Johnson and Shmatikov [7] and Yu et al. [8] proposed methods to release p-values by employing exponential mechanism and without limitations on the case and control group size. While [8] and [6] assumed that the number of most significant SNPs was already known, [7] proposed a framework for the exploration of GWAS data in differentially private manner. [7] proposed algorithms to find the count of most significant SNPs associated with the phenotype, compute the location of these SNPs, identify the longest block of correlated SNPs, release the p-values, and detect the correlation between a pair of SNPs using exponential mechanism. The scores for exponential mechanism are assigned using a distance score function, which counts the minimum number of alleles to be altered for the SNP's significance to change. Since finding the nearest neighbor where the significance of the SNP flips is a NP-hard problem (non-deterministic polynomial-time hard), works such as [8] proposed heuristics for its approximation. By assuming that the control group size is already known, [8] weakens the privacy constraints to obtain better results.

E. Implementation in WEKA

We design the PrivStats component of DPWeka to calculate the attribute selection metrics in a differentially private manner. As an example, we implement methods to evaluate the attributes

based on significance statistics, in both non-private and differentially private manners. However, PrivStats can be easily extended to include other statistics. PrivStats is available as an *Attribute Evaluator* under the *Select Attributes* panel (Figure 13). It is also available as a *Filter* by navigating to *weka> filter> supervised >attribute > AttributeSelection* in the *Preprocess* panel of WEKA's graphical interface. In both panels, PrivStats can be used along with the *Ranker* of *Search Method* to select the desired number of relevant attributes for further analysis.

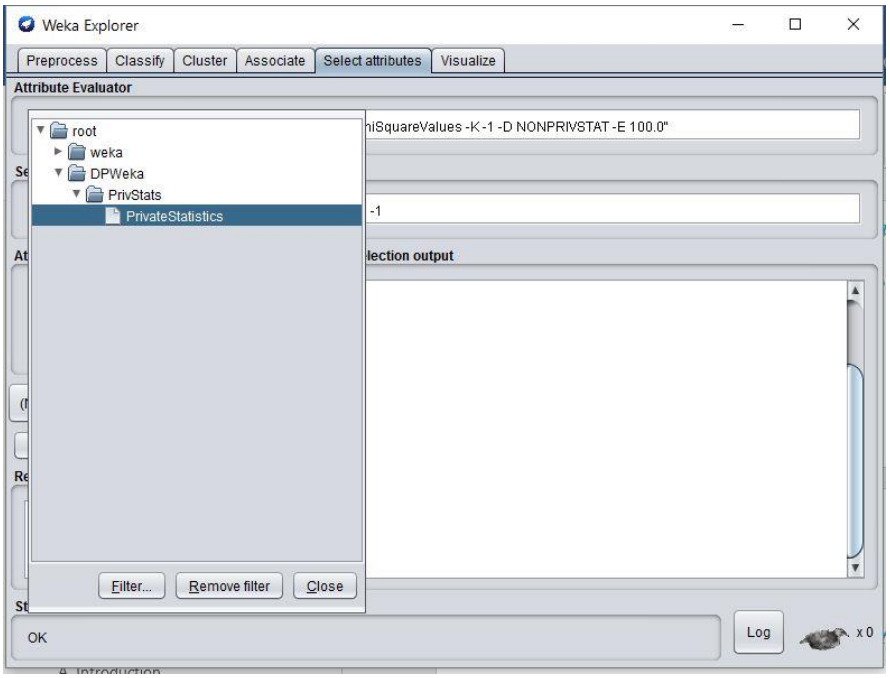

Figure 13 Access PrivStats from Select Attributes Panel

The PrivStats component contains *PrivateStatistics* method, which is implemented as a Java class that inherits from WEKA's *ASEvaluation*, *ASEvaluator*, and *OptionHandler*. From the property sheet of *PrivateStatistics* method, users can select the desired statistic for evaluation, as depicted in Figure 14. As an example, we implement methods to calculate χ^2 values, p-values, and odds ratio based on the 3x2 contingency matrix of each SNP and the trait. However, PrivStats can be easily extended to include other metrics (statistics). The statistic must be implemented as a Java class, which inherits from *CalculateStats* and *StatisticsToBeMeasured*

Java classes. Furthermore, from the property sheet of each statistic, users can select the desired privacy mechanism, control the privacy budget (ε), and assign the number of relevant attributes to be selected (k). For evaluating p-value using exponential mechanism, users can also assign the threshold of p-value (or significance value) (τ) to calculate the distance scores.

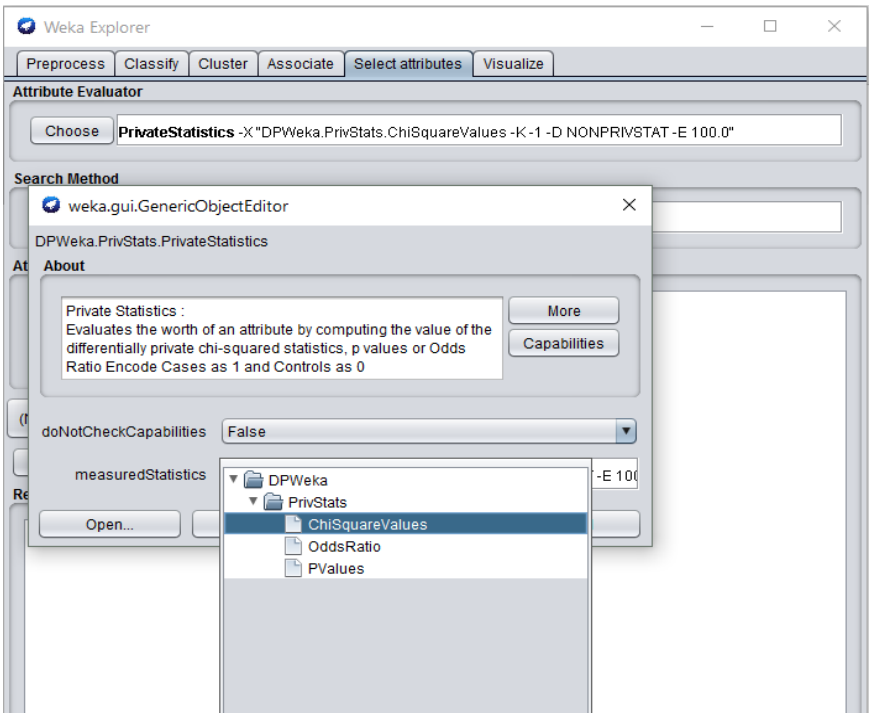

Figure 14 Property Sheet of PrivateStatistics Method

In our implementation, users can choose to evaluate the attributes based on χ^2 values using non-private or Laplace mechanisms as shown in Figure 15. We employ WEKA's *Statistics* class to calculate the true (non-private value) χ^2 value of each attribute. However, to calculate differentially private χ^2 values, we implement the approach proposed in [8]. In this approach, randomly generated Laplace noise of scale $4kt/\varepsilon$ is added to the true χ^2 values, where t is the sensitivity given by $\frac{N^2}{RS}\left(1-\frac{1}{\max(R,S)+1}\right)$, where N is the number of records, R is the control group size, and S is the case group size. The k most relevant attributes are selected based on the

perturbed χ^2 values. Laplace noise of $2kt/\varepsilon$ is added to the true χ^2 values of these selected attributes before releasing them. This algorithm is formally presented in Appendix VIII.

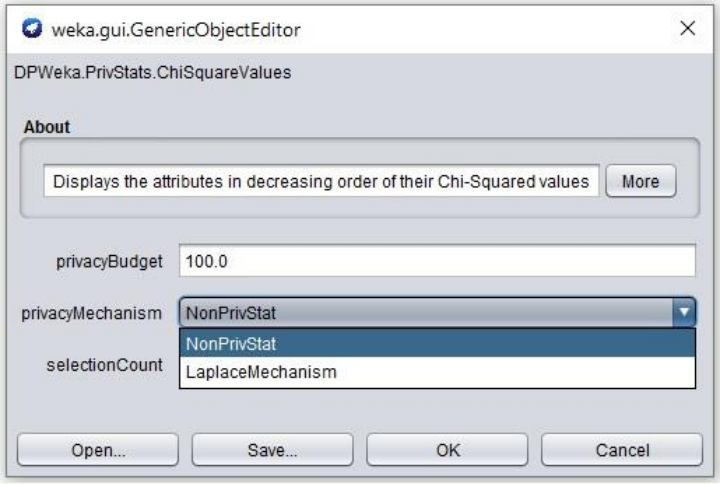

Figure 15 Property Sheet of χ^2 values

Furthermore, for evaluating attributes using p-values, users can choose from non-private, Laplace, or exponential mechanisms (Figure 16). We use WEKA's *Statistics* class to calculate p-values non-privately. However, we implement the Laplace mechanism for p-value based on the approach proposed in [6]. In this technique, the original p-values of all the attributes are perturbed by Laplace noise of scale $\frac{4k}{\varepsilon}e^{-\frac{2}{3}}$. Based on the perturbed values, *k* most relevant attributes are selected. Laplace noise of scale $\frac{2k}{\varepsilon}e^{-\frac{2}{3}}$ is then added to the true p-values of the selected attributes before releasing the selected attributes. This algorithm is formally presented in Appendix B.

For exponential mechanism, we implement the algorithm proposed in [7]. In this approach, each attribute is assigned a score based on smallest distance, *d*, to its neighboring contingency matrix where the significance of p-value changes. The significance is dependent on the threshold value assigned by the user (τ). The *k* most relevant attributes are then selected using exponential mechanism. Mathematically, the distance function [7], *d*, is given by

$$d(D) = \begin{cases} max\{r: |\{j: D_j \neq \acute{D}_j\}| < r \Rightarrow p(\acute{D}) \geq \tau\} & if\ p(D) \geq \tau \\ -max\{r: |\{j: D_j \neq \acute{D}_j\}| < r \Rightarrow p(\acute{D}) \geq \tau\} & otherwise \end{cases} \quad (14)$$

where the sign of d indicates the significance of the attribute. D is the actual contingency matrix of the attribute, $\acute{D}$ is the neighboring contingency matrix where the significance flips, $p()$ is a function that calculates the p-value. Formally, the scoring function [7], q, of sensitivity 1 is then given by

$$q(D) = \begin{cases} -d(D) - 1 & if\ p(D) < \tau \\ -d_i(D) & otherwise \end{cases} \quad (15)$$

Appendix C presents the algorithm to evaluate the attributes based on differentially private p-values using exponential mechanism.

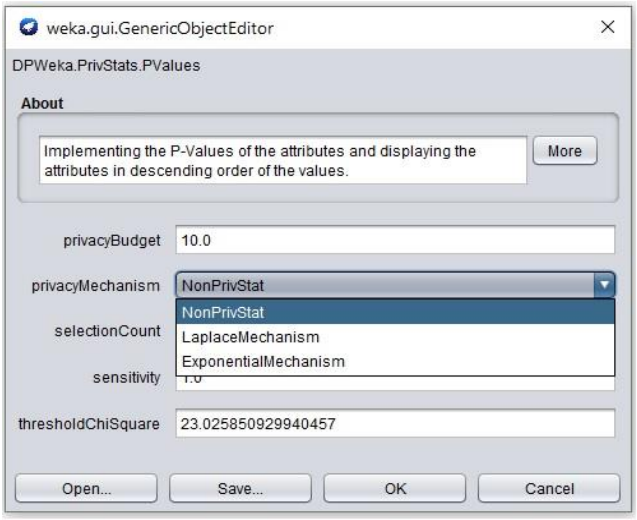

Figure 16 Property Sheet of p-values

These statistics can also be calculated independent of WEKA by passing the selection parameters to the Java class, *callStatistics*, provided in the source code. However, users must then include appropriate methods if filtering is needed.

The PrivStats component contains a Java class, *NoiseDistributions*, which can be employed to generate random numbers from Laplace and exponential distributions. This class can also be easily extended to include other distributions.

F. Datasets

The GWAS dataset for each phenotype gives a list of subjects associated with the case and control group of the phenotype. Each subject has a text file containing their rsids (chromosome ID) and the allele values. Since a single flat file containing the SNP details of all the subjects must be loaded for processing in WEKA, we develop a Java package, *SNPDataPreProcessing*, with functions to parse through the files, filter the SNPs and subjects as per the quality control procedures, and generate a master ARFF file for the phenotype. Each individual's genome sequence is considered as an instance of the dataset and each SNP is considered as an attribute.

For our experiments, we choose the subject files from openSNP database [56] for tongue roller phenotype. We identify over 1,750,000 unique SNPs among the 330 subjects belonging to the case and control groups of this study. The SNPs with call rate less than 95 % are filtered. Call rate is the percentage of subjects in whom the SNPs are present. Thus, any SNP not present in more than 95% of population is removed. Further, subjects with more than 5% of missing data are not considered in the experiment. Finally, SNPs with minor allele frequency less than 0.01, and SNPs with any of the alleles occurring in less than 10 subjects are filtered out. After these pre-processing tasks, our dataset has 300 instances and 262,622 SNPs.

G. Evaluation

We evaluate the accuracy of the differentially private mechanisms by comparing the privately identified SNPs with the true (non-privately identified) relevant SNPs. Experiments are conducted by varying the privacy budget values (ε) and the number of relevant SNPs to be identified (k). Each experiment is executed for 100 times, and the number of relevant SNPs correctly identified are averaged. We plot the privacy budget on X-axis, and on Y-axis we plot the percentage of relevant SNPs that are correctly identified. The percentage is calculated by

dividing the average number of SNPs correctly identified by the number of relevant SNPs to be identified and multiplying the ratio with 100. For example, when *k*=3 and ε = 50, if on average, 1 out of 3 most relevant SNPs is identified, the percentage is calculated as (1/3*100) = 33.33%. Since in GWAS research, the relevant SNPs will be processed further, we do not take the order or ranks of SNPs into account.

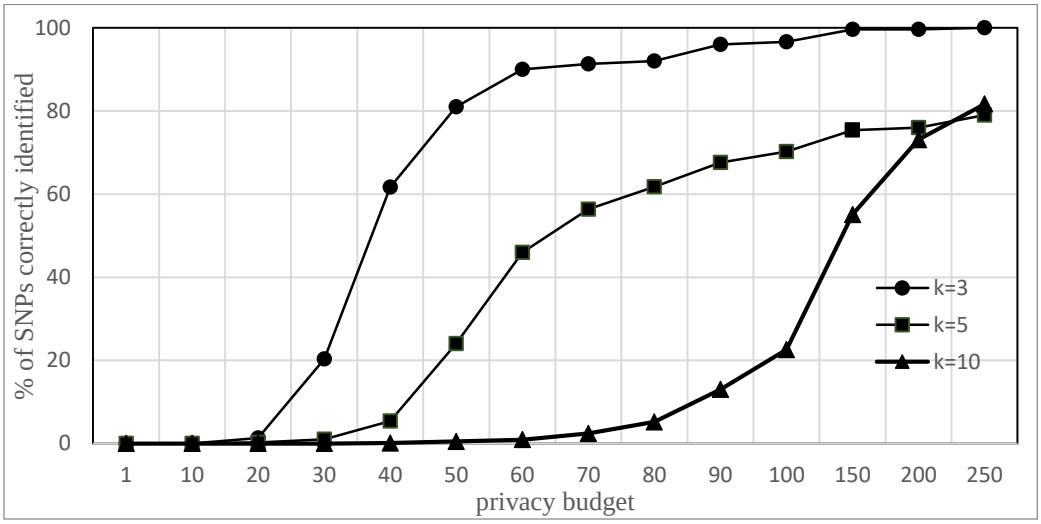

Figure 17 Evaluation of Laplace Mechanism for χ^2-values

Figure 17 depicts the evaluation of Laplace mechanism for χ^2 values. As expected, for larger privacy budget, and thus weaker privacy guarantees, the chance of identifying the truly relevant SNPs is higher. Also, for smaller selection count values (*k*), the relevant SNPs are accurately identified at smaller privacy budgets. This can be attributed to the sequential property of differential privacy. When the selection count is smaller, privacy budget assigned for each SNP is larger. For example, for an overall budget of 100, when the number of SNPs to be released is 10, the privacy budget of 10 is assigned for each SNP to be identified and released, whereas when the number of SNPs to be released is 5, each SNP gets a budget of 20 for the same.

We evaluate the exponential mechanism for p-values similarly. Figure 19 depicts the trend when the threshold p-value (τ) is 10^{-5}. The graph follows similar patterns as that in Figure 17.

The percentage of SNPs that are correctly identified increases as the privacy budget is increased, and when the *k* values are smaller, the SNPs are accurately identified at smaller privacy budget. However, when $k = 5$, there seems to be no improvement in the identification despite increasing the privacy budget. This can be explained by considering the graph in Figure 18, which plots the true p-values of SNPs in increasing order and has sharp increase at SNP numbers 3 and 10. At these points, the probabilities assigned to the SNPs can be distinguished clearly from their neighbors; thus the accuracy of exponential mechanism is better. However, at other points such as $k = 5$, the neighboring SNPs have approximately similar p-values, and hence the scores assigned are similar, which in turn makes it difficult for the mechanism to distinguish between the neighboring SNPs. When we relax the SNP identification constraint by identifying any 5 of the 10 most relevant SNPs, the accuracy improves. This is shown in the *k = 5 weak* plot in Figure 19. The plot depicts that when $\varepsilon = 80$, all the 5 identified SNPs are among the top 10 relevant SNPs. The accuracy of this method depends on how closely the distance trend follows the p-value trend.

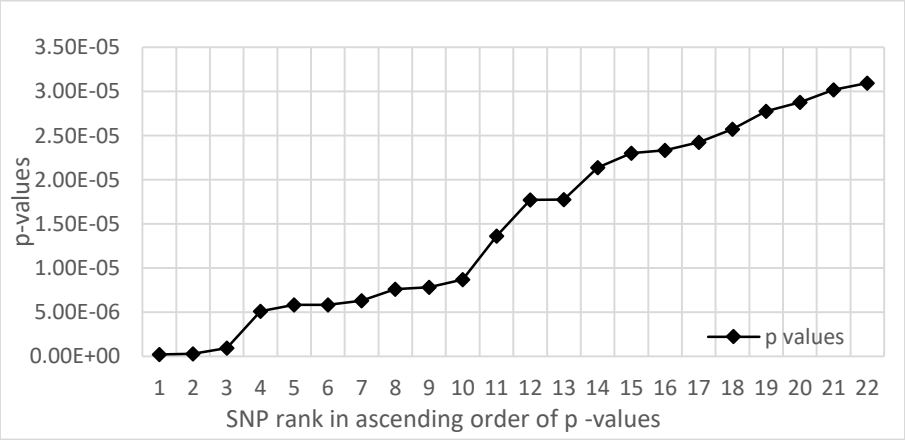

Figure 18 Unperturbed Top p-values in Ascending Order

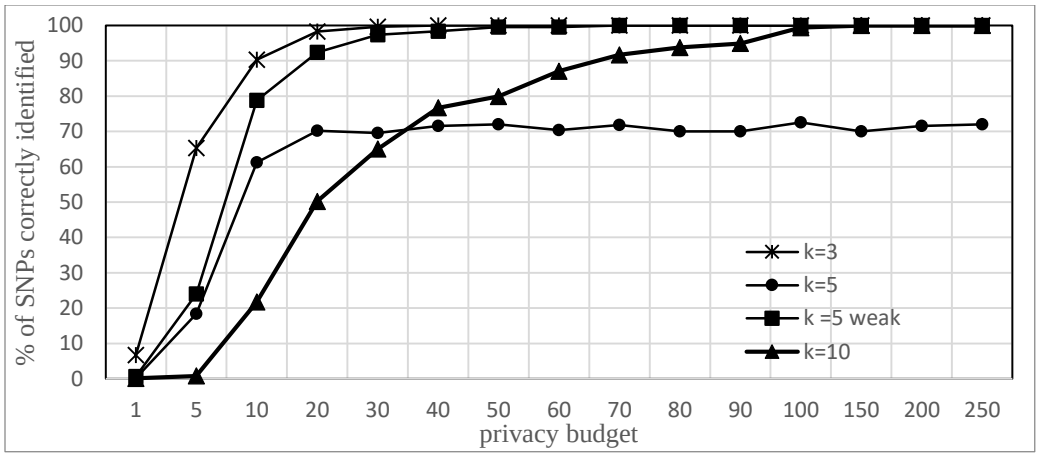

Figure 19 Evaluation of Exponential Mechanism for p-values

Since the assumption for calculating p-values using Laplace mechanism is equal number of subjects in the case and control groups, we modify our dataset to meet the condition. We find in our evaluations that this mechanism does not perform as well as other mechanisms.

H. Summary

To summarize, attribute selection is performed in data mining to identify the relevant attributes of the dataset. It reduces the training time for generating data models and improves the accuracy of the models. In this chapter, we present PrivStats component, which is designed to calculate the metrics for attribute selection in differentially private manner. The component has methods to calculate the χ^2 values, and p-values, and it provides user support to control various privacy and method-specific parameters. We consider a genetic dataset to demonstrate that the performance of the private algorithms depends on the privacy budget and the method parameters.

V. IMPLEMENTATION OF DIFFERENTIALLY PRIVATE LOGISTIC REGRESSION

A. Introduction

One of the primary tasks in data mining is generating models using learning algorithms. Input data, also called training data, are presented to the learning algorithms to construct models that fit the training data. In supervised learning, the learning algorithms use labelled data to infer the mapping function for model construction, whereas in unsupervised learning, the learning algorithms construct the models with unlabeled data. Classification and regression are examples for supervised learning methods, whereas clustering and association rule mining are examples for unsupervised learning.

The training data presented to a supervised learning algorithm consists of predictor attributes (also called predictors) denoted by X and response attributes (also called labels) denoted by Y. Typically, a regression model is used when the response attribute is continuous, whereas a classification model is used when the response attribute is categorical. However, when a classification problem is solved by assigning probabilities for each category, it can be considered as a special form of regression.

Models generated by a supervised learning algorithm can be employed to analyze the simultaneous effect of multiple predictor attributes on labels. Supervised learning models are primarily used for inference and prediction purposes. Under an inference paradigm, the model parameters are used to identify the predictors related to the labels and determine the nature of their relationship. On the other hand, under a prediction paradigm, models are used to determine the label when the predictor attributes are given.

When data curators release the model parameters, an adversary with access to the model can abuse it to extract private sensitive information of the individuals in training data. The technique used to compromise the privacy of training data using model parameters is called model inversion attack [57]. Models constructed employing the sensitive data, such as healthcare data, are particularly vulnerable to these attacks and can compromise the privacy of individuals present in the dataset. Thus, privacy preserving methods are imperative while constructing the models involving sensitive data.

In the following sections, we examine how differential privacy mechanisms can be applied to preserve the model's privacy. As an example, we study logistic regression and examine methods to release differentially private parameters of logistic regression models. Furthermore, we discuss implementation details of some of these methods within PrivClassifier component of DPWeka. Finally, we present the evaluations of these mechanisms on multiple census and marketing datasets.

B. Logistic Regression

Logistic Regression is commonly used to model datasets containing binary responses such as yes or no; pass or fail. Its prediction function, given by logistic function (also sigmoid function), depicts the curvilinear relationship between the inputs and outputs (Figure 20). Thus, the probability of response for logistic regression is then given by

$$\Pr(y_i = 1|\boldsymbol{x_i}) = \frac{1}{1+\exp(-\boldsymbol{x_i^T\omega})} \quad \text{and}$$

$$\Pr(y_i = 0|\boldsymbol{x_i}) = \frac{1}{1+\exp(\boldsymbol{x_i^T\omega})} \qquad (16)$$

where the predictor attributes (features) are denoted by $\boldsymbol{x_i}$, and the predicted label is denoted by y_i. The parameter vector $\boldsymbol{\omega}$ is selected such that the cost function (also referred as objective

function or loss function), which evaluates how well the model fits the data, is minimized. The cost function for logistic regression is given in equation (17), and its derivation is discussed in Appendix D.

$$\sum_{i=1}^{n} f\,(\,\boldsymbol{\omega}) = \sum_{i=1}^{n} ln(\,1 + \exp\left(\boldsymbol{x_i^T \omega}\right)) - y_i \boldsymbol{x_i^T \omega}. \tag{17}$$

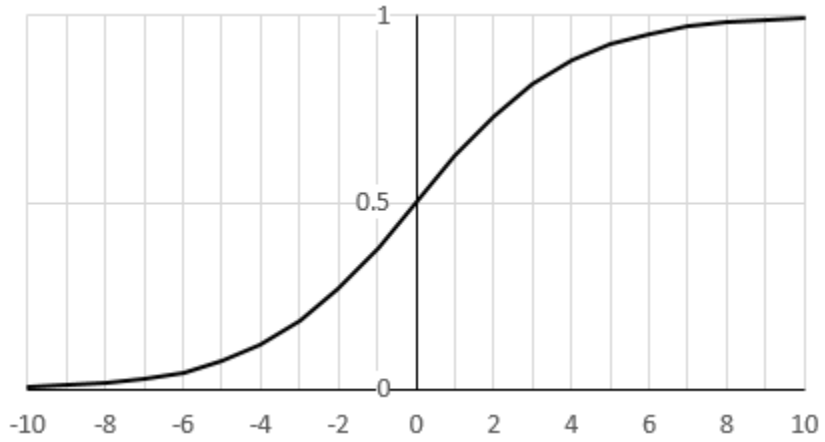

Figure 20 Logistic Mapping of Single Variable Input and Output

Fitting the model to the training data can sometimes cause overfitting, which leads to poor prediction performance. Regularization is one of the techniques to reduce overfitting. It improves the generalization of the model by adding controlled noise to the cost function. Some common forms of regularization include L1- regularization (also called Lasso), L2-regularization (also called ridge), and elastic net regularization. In L1-regularization, noise equivalent to the sum of parameters is added to the cost function. On the other hand, in L2-regularization, noise equivalent to the sum of the square of parameters is added to the cost function. In elastic net regularization method, a combination of both L1 and L2 noise is added to the cost function. Mathematically, L1-regularization is given by

$$\sum_{i=1}^{n} ln(\,1 + \exp\left(\boldsymbol{x_i^T \omega}\right)) - y_i \boldsymbol{x_i^T \omega} + \lambda_1 \sum_{j=1}^{n} |\omega_j| \tag{18}$$

and L2-regularization is given by

$$\sum_{i=1}^{n} ln(1 + \exp(\boldsymbol{x_i^T \omega})) - y_i \boldsymbol{x_i^T \omega} + \lambda_2 \sum_{j=1}^{n} \omega_j^2 \quad (19)$$

where $\lambda_1$and λ_2 control the amount of regularization.

The regularized cost function is then optimized to obtain the optimal model parameters, which can be used for prediction or inference purposes.

C. Literature Review

Chapter II of this thesis discusses various mechanisms for achieving differential privacy in practice. To directly apply Laplace mechanism to the regression model parameters, the mechanism's sensitivities must be calculated. However, due to the complex correlations of the labels and predictors, the computations of the sensitivities become complicated, which in turn deteriorates the differentially private model's performance. Hence, direct application of Laplace mechanism for regression models is disadvantageous.

Chaudhuri et al. [9] proposed the objective perturbation method which entails adding noise to the objective function before solving for optimal $\boldsymbol{\omega}$ values. However, this approach requires that the regularization and the loss functions satisfy specific differentiability and convexity properties. Kifer et al. [10] later improved this approach to include non-differentiable regularization methods. Yu et al. [55] combined the improved approach in [10] with elastic net regularization to perform privacy-preserving penalized logistic regression.

In functional mechanism approach [12], Zhang et al. enforced differential privacy by perturbing the approximate objective function of regression analysis with Laplace noise. The sensitivity for the Laplace noise is obtained as $\frac{d^2}{4} + 3d$, where d is the number of attributes in dataset. Thus, the noise is dependent only on the number of attributes and independent of the size

of dataset. Wang et al. [15] improved the approach by adding more noise to sensitive attributes than non-sensitive attributes, while maintaining the accuracy under model inversion attacks.

In the PrivGene method [11], Zhang et al. proposed to optimize the objective function in a differentially private manner. In this approach, a differentially private genetic algorithm is used for selecting the optimal parameters. To enforce privacy, a variation of exponential mechanism called enhanced exponential mechanism is used in selecting the optimal parameter vector.

D. Implementation in WEKA

The PrivClassifier component in DPWeka is designed to implement differentially private classification and regression models. As an example, we implement methods to perform logistic regression using functional mechanism [12] and enhanced exponential mechanism (based on PrivGene) [11] . Both the methods inherit from WEKA's *AbstractClassifier* and *OptionHandler* classes to be accessible from *Classify* panel of the graphical interface (Figure 21).

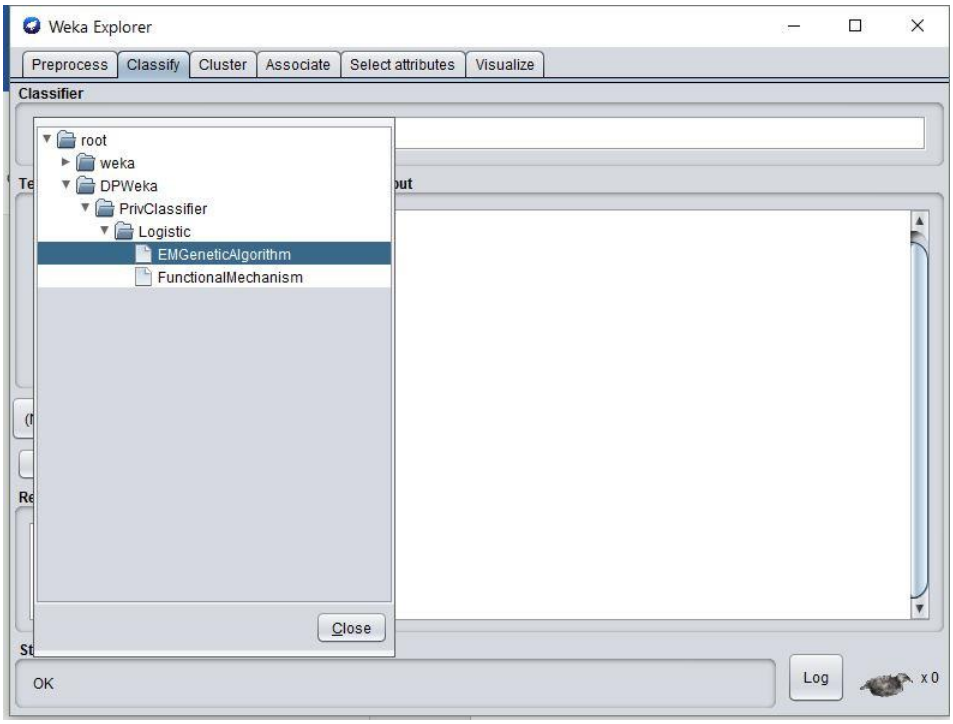

Figure 21 Access PrivClassifier from Classify Panel

The functional mechanism algorithm is implemented as *FunctionalMechanism* Java class in PrivClassifier. The *FunctionalMechanism* method allows users to control the privacy budget (ε) and the number of iterations (*t*) through its property sheet (Figure 22). In this approach, the cost function of the logistic regression is approximated to polynomial form of the second degree based on Taylor expansions. The approximated cost function is given by

$$\tilde{f}_D(\omega) = \sum_{i=1}^{n} (log2) * (x_i^T \omega)^2 + \frac{1}{2} x_i^T \omega + \frac{1}{8} - \sum_{i=1}^{n} y_i x_i^T \omega \quad (20)$$

Then, we perturb coefficients of the parameter vector with Laplace noise of scale $(\frac{\frac{d^2}{4}+d}{\varepsilon})$, where *d* is the number of attributes in the dataset. The matrix representation of the noisy quadratic cost function is then of the form

$$\bar{f}_D(\omega) = \omega^T M^* \omega + \alpha^* \omega + \beta^* \quad (21)$$

To obtain a bounded cost function, L2 regularization and spectral trimming are performed. The regularized cost function transforms to

$$\hat{f}_D(\omega) = \omega^T (M^* + \lambda I)\, \omega + \alpha^* \omega + \beta^* \quad (22)$$

where λ denotes the regularization term.

For spectral trimming, the eigen decomposition of coefficient matrix, $(M^* + \lambda I)$, is performed to obtain eigenvalue matrix (Λ) and eigenvector matrix (Q). Then, from the eigenvalue matrix, the rows containing non-positive eigenvalues in the diagonal are removed to obtain a modified diagonal matrix (Λ'). The corresponding rows from the eigenvector matrix are deleted to obtain a modified matrix Q'. The equation is then rearranged to obtain the cost function as

$$\hat{g}_D(Q'\omega) = (Q'\omega)^T \Lambda' (Q'\omega) + \alpha^* Q'^T (Q'\omega) + \beta^* \quad (23)$$

This resultant function is optimized to find $Q'\omega$ using WEKA's Active-sets method with BFGS update optimizer. The optimization runs for *t* iterations or until the optimal values are found,

whichever is earlier. The optimal values are then post-processed to obtain the differentially private model parameters. The algorithm is presented in Appendix E.

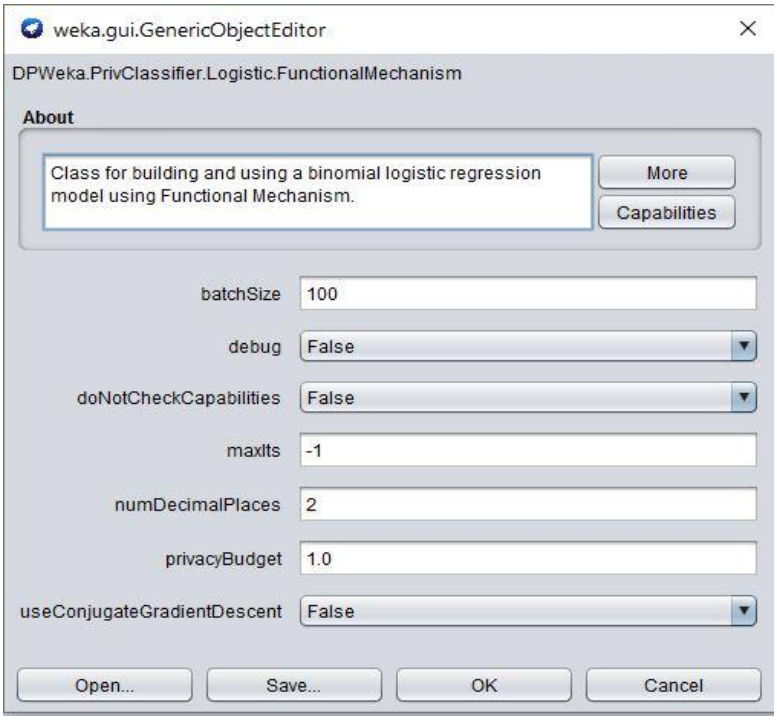

Figure 22 Property Sheet of FunctionalMechanism

The PrivGene algorithm is implemented as *EMGeneticAlgorithm* Java class in PrivClassifier. This method employs differentially private genetic algorithm for optimization. A genetic algorithm involves initializing a set of m' random vectors. These vectors are then crossed over to create offspring parameter vectors, which are then mutated to generate a candidate set of m mutations denoted by Ω_{m}. From this set, m' mutations which best fit the cost function are selected to form the selected set, $\Omega'_{\mathrm{m}'}$. The crossover and mutation operations are performed on the selected set to form new candidate set, from which a new selected set is generated. This process is performed for desired number of iterations or till the optimal parameter vector is obtained. Differential privacy is enforced by employing a variation of exponential mechanism, called enhanced exponential mechanism, for generating the selected set $\Omega'_{\mathrm{m}'}$ in each iteration.

Enhanced exponential mechanism can be applied for applications whose fitting functions can be expressed as containing a component independent of sensitive data and another data dependent component. It differs from exponential mechanism only in the computation of sensitivity for scoring functions; the scoring methods and the selection methods for enhanced exponential mechanism is same as exponential mechanism. For logistic regression, when the selected set size m' is 1, the sensitivity of scoring function for enhanced exponential mechanism, is bound to 4 times the mutation scale [11].

In the *EMGeneticAlgorithm* method, we provide users the ability to control the privacy budget (ε), the mutation scale (σ) and the mutation resize value (*s*) through the property sheet (Figure 23). In the implementation, we generate a random vector of size *d*, where *d* is the number of attributes of the dataset. Mutation operation is performed by adding a noise of + σ and – σ to a single element at a time, generating a total of 2*d* vectors in the candidate set. Each candidate vector $\boldsymbol{\omega}'$ is then assigned scores based on the scoring function given by

$$q(\boldsymbol{\omega}') = \sum_{i=1}^{n} y_i \boldsymbol{x_i^T} \boldsymbol{\omega}' - ln(1 + exp(\boldsymbol{x_i^T} \boldsymbol{\omega}')) \quad (24)$$

The sensitivity of the scoring function is given by 4σ. Exponential mechanism is then used to select a single vector based on the scores. The selected vector then undergoes mutation, and the process is repeated for a given number of iterations. However, in each iteration, the noise added in the mutation operation varies by a factor of resize value (*s*), i.e., $\sigma * s$ amount of noise is added in the second iteration, and noise of $\sigma * s^2$ is added in third iteration, and so on. Similarly, the sensitivity for each iteration varies by a factor of *s*. The parameter vector selected in the final iteration is released as the model parameter. The algorithm is presented in Appendix F.

To employ these methods, each predictor attribute of the dataset must be normalized to [-1,1], and the response attribute must be a binary attribute containing either 0 or 1 values. A dataset can be easily transformed to meet these criteria using WEKA *filters*.

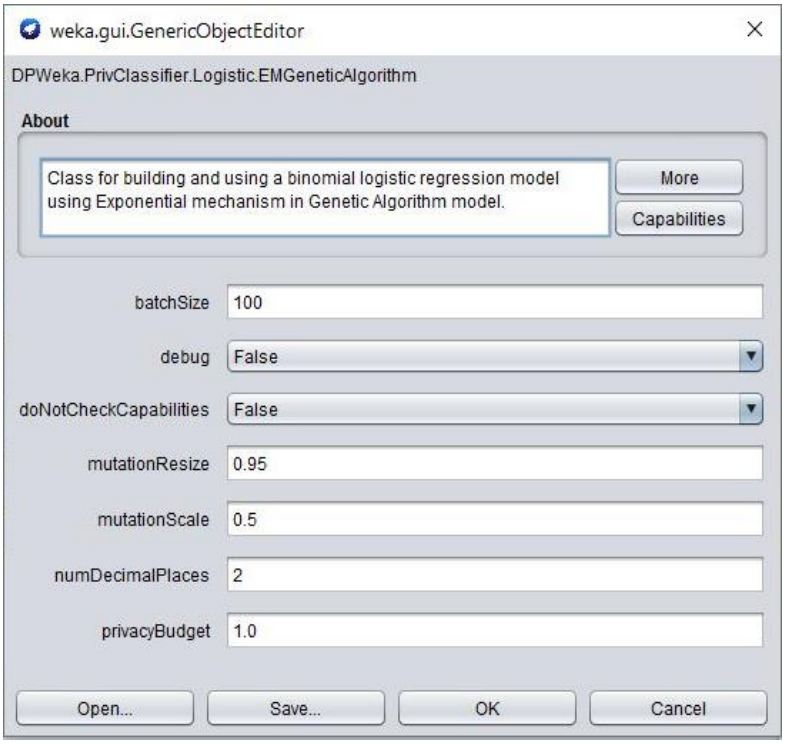

Figure 23 Property Sheet of EMGeneticAlgorithm

E. Datasets

For evaluating our implementation, we use five data sets: (i) Adult dataset [58], which contains data from 1994 household census to predict if the annual household income is greater than $50,000. (ii) Banking dataset [59], which contains marketing data from banking institution to predict if a customer subscribes to a term deposit. (iii) Three datasets from Integrated Public Use Microdata Series (IPUMS) [60], for US and Brazil containing census records for years 2000 and 2009 to predict if the annual income of an individual is greater than a predefined threshold value.

These datasets contain both numerical and categorical attributes. Each categorical attribute, containing *l* distinct values, is transformed into *l* binary attributes. Each predictor attribute is then normalized to range [-1, 1]. The response attribute is set to be a binary attribute. Table 2 summarizes the properties of each of these datasets. We use the datasets preprocessed by Zhang et al. in [12] and [11].

S no.	**Dataset**	**Number of tuples**	**dimensions**
1	Adult dataset	48,842	14 (Original) 124 (Converted)
2	Banking	45,211	17 (Original) 33 (Converted)
3	IPUMS- US -2009	370044	14
4	IPUMS-BR2000	38,000	53
5	IPUMS-US2000	40,000	58

Table 2 Datasets properties

F. Evaluation

We evaluate the performance of the differentially private logistic regression by plotting the graph of misclassification rate for varying values of privacy budget (ε). For each experiment, the dataset is split into 80 percent training data and 20 percent test data. The misclassification rates of 500 such experiments are then averaged. Figure 24 and Figure 25 plot the misclassification rates of *FunctionalMechanism* and *EMGeneticAlgorithm* for all the five datasets. As expected, the rate of misclassification reduces as the privacy budget is increased.

For the evaluation of *EMGeneticAlgorithm*, we follow the authors' convention in [11] and set the number of iterations to $0.125 * \varepsilon * number\ of\ instances$. The slight increase in the misclassification rate can be attributed to the tuning parameters associated with the mutation

parameters. We demonstrate that the *EMGeneticAlgorithm* achieves similar accuracy values for lower privacy budgets as compared to Functional Mechanism.

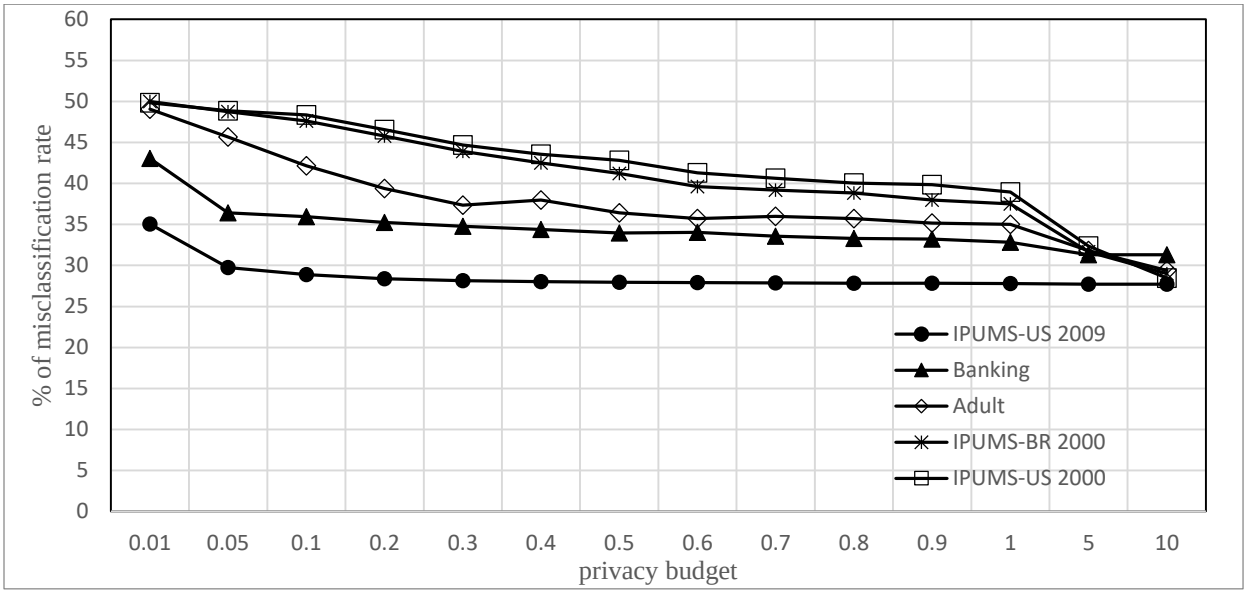

Figure 24 Evaluation of FunctionalMechanism for Varying Privacy Budget

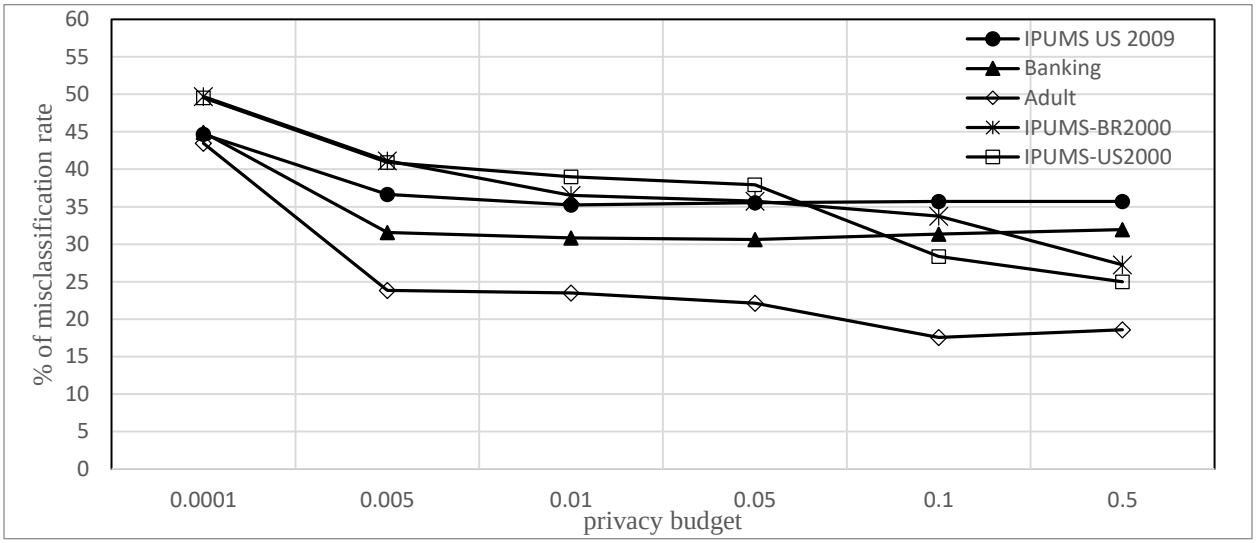

Figure 25 Evaluation of EMGeneticAlgorithm for Varying Privacy Budget

G. Summary

To summarize this chapter, the models built by learning algorithms using sensitive data are vulnerable to privacy leaks. Hence, various approaches have been proposed to make the learning

algorithms differentially private. In this chapter, we discuss the implementation of PrivClassifier component of DPWeka. The PrivClassifier contains differentially private logistic regression methods *FunctionalMechanism* and *EMGeneticAlgorithm*, which are based on objective function perturbation and exponential mechanism respectively. The *FunctionalMechanism* method ensures privacy by perturbing the approximated cost function, whereas the *EMGeneticAlgorithm* method ensures privacy by making the optimization process differentially private. These methods provide support to control the privacy budget and model parameters such as the number of iterations. We evaluate these methods on several datasets and demonstrate that their performance is determined by the privacy budget and the model parameters.

VI. CONCLUSIONS AND FUTURE WORK

The implementations discussed in this thesis are based on current research in differential privacy. Hence, much of the richness of the differentially private solutions to machine learning problems is left unexplored. In this chapter, we first summarize the entirety of the thesis. Then we discuss the scope of extending this work to provide exhaustive support for privacy-preserving data mining tasks.

A. Conclusions

In this thesis, we discussed differential privacy and various mechanisms to achieve it in practice (Chapter II). Differential privacy provides formal privacy guarantees against an adversary with arbitrary background knowledge. Typically, differential privacy is achieved through randomization, which restricts the adversary from inferring private information of the individual present in the dataset. Output perturbation, exponential mechanism, sample and aggregate framework, and objective function perturbation are some of the mechanisms to realize differential privacy. Tools that ensure differential privacy in data mining methods, including PINQ, GUPT, and Airavat, employ only certain mechanisms in their design which limits their scope to a few data mining tasks. Some of these tools need users to possess programming skills for operational purposes. On the other hand, DPWeka, an extension to WEKA, is a comprehensive GUI based tool with options to select various differentially private mechanisms for data mining tasks. The graphical interface caters to non-programmers and non-privacy experts to perform privacy preserving data mining.

In Chapter III, we discussed the features available within WEKA, an open source machine learning tool, to perform comprehensive data mining tasks. New features can be integrated with WEKA by creating WEKA packages. A WEKA package is an archive containing the application

JAR file along with the property files. The application design must adhere to WEKA implementation structure to be accessible within WEKA. DPWeka is designed conforming to WEKA's implementation standards.

In data mining, attribute selection is performed to identify the most relevant attributes of the dataset. Metrics such as significance statistics and correlation statistics are used to perform attribute selection. In Chapter IV, we discussed DPWeka's PrivStats component, which is designed to calculate the metrics for attribute selection methods. As an example, we developed methods to calculate significance statistics such as χ^2 values and p-values in differentially private manner. The PrivStats component provides support to select the desired mechanism, and control the privacy budget and other model parameters.

In Chapter V, we discussed DPWeka's PrivClassifier component, which is designed to perform classification in differentially private manner. As an example, we implemented methods to perform logistic regression employing two different mechanisms. The *FunctionalMechanism* method is based on the functional mechanism, which ensures differential privacy by perturbing the approximated cost function. The *EMGeneticAlgorithm* method ensures privacy by making the optimization process of the cost function differentially private. The optimization process is based on a genetic algorithm, which mimics the process of evolution. The PrivClassifier component provides support to select the desired mechanism, and control the privacy budget and other model parameters.

We evaluated the differentially private methods on various real world datasets, such as genetic data, census data, and banking data. In our evaluations, we demonstrated that the performance of private methods depends on the model parameters as well as the privacy budget.

B. Future Work

This work offers several future directions. One direction is to implement additional differentially private methods to support comprehensive privacy preserving data mining process. It includes implementing an exhaustive range of differentially private pre-processing methods, learning schemes, and diagnostics methods. The extensive tool would then support users to set privacy budget for the entire data mining process or for each component. Based on the user setting, DPWeka will then allocate the budget for each component automatically or estimate the overall privacy budget.

Another direction is to extend DPWeka as a dedicated tool for specific application such as genetic data analysis. In this approach, differentially private methods designed particularly for each step of application's analysis are implemented within the tool. The package being application specific allows users to focus on the functional aspects of the application.

VII. REFERENCES

[1] A. Narayanan and V. Shmatikov, "Robust De-anonymization of Large Sparse Datasets," in *IEEE Symposium on Security and Privacy*, 2008, pp. 111–125.

[2] M. Barbaro and T. Zeller Jr, "A Face Is Exposed for AOL Searcher No. 4417749 - The New York Times," *New York Times*, 2006. [Online]. Available: http://www.nytimes.com/2006/08/09/technology/09aol.html?_r=0. [Accessed: 10-Mar-2017].

[3] M. Kantarcioğlu, J. Jin, and C. Clifton, "When do data mining results violate privacy?," in *Proceedings of the 2004 ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '04*, 2004, p. 599.

[4] K. Chaudhuri, K. Edu, A. D. Sarwate, and K. Sinha, "A Near-Optimal Algorithm for Differentially-Private Principal Components *," *J. Mach. Learn. Res.*, vol. 14, pp. 2905–2943, 2013.

[5] Y. Wang, X. Wu, and D. Hu, "Using Randomized Response for Differential Privacy Preserving Data Collection," in *9th International Workshop on Privacy and Anonymity in the Information Society (PAIS)*, 2016.

[6] S. E. Fienberg, A. Slavkovic, and C. Uhler, "Privacy Preserving GWAS Data Sharing," in *2011 IEEE 11th International Conference on Data Mining Workshops*, 2011, pp. 628–635.

[7] A. Johnson and V. Shmatikov, "Privacy-Preserving Data Exploration in Genome-Wide Association Studies.," *KDD proceedings. Int. Conf. Knowl. Discov. Data Min.*, vol. 2013, pp. 1079–1087, Aug. 2013.

[8] F. Yu, S. E. Fienberg, A. B. Slavković, and C. Uhler, "Scalable privacy-preserving data sharing methodology for genome-wide association studies," *J. Biomed. Inform.*, vol. 50, pp. 133–141, Aug. 2014.

[9] K. Chaudhuri, C. Monteleoni, and A. D. Sarwate, "Differentially Private Empirical Risk Minimization," *J. Mach. Learn. Res.*, vol. 12, pp. 1069–1109, 2011.

[10] D. Kifer, A. Smith, A. Thakurta, S. Mannor, N. Srebro, and R. C. Williamson, "Private Convex Empirical Risk Minimization and High-dimensional Regression," in *25th Annual Conference on Learning Theory, PMLR*, 2012, vol. 2340, no. 25, pp. 1–25.

[11] J. Zhang, X. Xiao, Y. Yang, Z. Zhang, and M. Winslett, "PrivGene," in *Proceedings of the 2013 international conference on Management of data - SIGMOD '13*, 2013, p. 665.

[12] J. Zhang, Z. Zhang, X. Xiao, Y. Yang, and M. Winslett, "Functional mechanism," *Proc. VLDB Endow.*, vol. 5, no. 11, pp. 1364–1375, Jul. 2012.

[13] A. Friedman and A. Schuster, "Data mining with differential privacy," in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '10*, 2010, p. 493.

[14] Y. Chen, A. Machanavajjhala, J. P. Reiter, and A. F. Barrientos, “Differentially Private Regression Diagnostics.”

[15] Y. Wang, C. Si, and X. Wu, “Regression Model Fitting under Differential Privacy and Model Inversion Attack,” *Proc. Twenty-Fourth Int. Jt. Conf. Artif. Intell.*, no. Ijcai, pp. 1003–1009, 2015.

[16] N. Phan, Y. Wang, X. Wu, and D. Dou, “Differential privacy preservation for deep auto-encoders: an application of human behavior prediction,” in *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, 2016, pp. 1309–1316.

[17] Y. Wang and X. Wu, “Preserving Differential Privacy in Degree-Correlation based Graph Generation.,” *Trans. data Priv.*, vol. 6, no. 2, pp. 127–145, Aug. 2013.

[18] Y. Wang, X. Wu, and L. Wu, “Differential Privacy Preserving Spectral Graph Analysis,” in *Proceedings of the 17th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*, 2013, pp. 329–340.

[19] F. D. McSherry, “Privacy integrated queries,” in *Proceedings of the 35th SIGMOD international conference on Management of data - SIGMOD ’09*, 2009, pp. 19–30.

[20] L. Sweeney, “k-ANONYMITY: A MODEL FOR PROTECTING PRIVACY,” *Int. J. Uncertainty, Fuzziness Knowledge-Based Syst.*, vol. 10, no. 5, pp. 557–570, Oct. 2002.

[21] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam, “*L* -diversity,” *ACM Trans. Knowl. Discov. Data*, vol. 1, no. 1, p. 3–es, Mar. 2007.

[22] N. Li, T. Li, and S. Venkatasubramanian, “t-Closeness: Privacy Beyond k-Anonymity and l-Diversity,” in *2007 IEEE 23rd International Conference on Data Engineering*, 2007, pp. 106–115.

[23] Z. Ji, Z. C. Lipton, and C. Elkan, “Differential Privacy and Machine Learning: a Survey and Review,” Dec. 2014.

[24] C. Dwork, “Differential Privacy,” in *Proceedings of the 33rd international conference on Automata, Languages and Programming - Volume Part II*, Springer-Verlag, 2006, pp. 1–12.

[25] C. Dwork, “Differential Privacy: A Survey of Results,” in *Theory and Applications of Models of Computation*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 1–19.

[26] C. Dwork and A. Roth, “The Algorithmic Foundations of Differential Privacy,” *Found. Trends® Theor. Comput. Sci.*, vol. 9, no. 3–4, pp. 211–407, 2013.

[27] S. Ranjit Ganta, S. Kasiviswanathan, and A. Smith, “Composition Attacks and Auxiliary Information in Data Privacy.”

[28] C. Dwork, F. McSherry, K. Nissim, and A. Smith, “Calibrating Noise to Sensitivity in Private Data Analysis,” in *Proceedings of the Third conference on Theory of Cryptography*, Springer-Verlag, 2006, pp. 265–284.

[29] C. Dwork, “A firm foundation for private data analysis,” *Commun. ACM*, vol. 54, no. 1, p. 86, Jan. 2011.

[30] A. Ghosh, T. Roughgarden, and M. Sundararajan, “Universally Utility-maximizing Privacy Mechanisms,” *SIAM J. Comput.*, vol. 41, no. 6, pp. 1673–1693, 2012.

[31] Q. Geng, P. Kairouz, S. Oh, and P. Viswanath, “The Staircase Mechanism in Differential Privacy,” *IEEE J. Sel. Top. Signal Process.*, vol. 9, no. 7, 2015.

[32] Q. Geng and P. Viswanath, “The Optimal Mechanism in Differential Privacy,” Dec. 2012.

[33] K. Nissim, S. Raskhodnikova, and A. Smith, “Smooth sensitivity and sampling in private data analysis,” in *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing - STOC ’07*, 2007, p. 75.

[34] F. McSherry and K. Talwar, “Mechanism Design via Differential Privacy,” in *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS’07)*, 2007, pp. 94–103.

[35] P. Jain and A. Thakurta, “Differentially private learning with kernels,” *Proceedings of the 30th International Conference on Machine Learning - Volume 28*. JMLR.org, pp. 118–126, 2013.

[36] Z. Huang and S. Kannan, “The Exponential Mechanism for Social Welfare: Private, Truthful, and Nearly Optimal,” in *2012 IEEE 53rd Annual Symposium on Foundations of Computer Science*, 2012, pp. 140–149.

[37] F. McSherry, “Privacy Integrated Queries: An Extensible Platform for Privacy-Preserving Data Analysis,” *Commun. ACM*, vol. 53, no. 8, pp. 89–97, 2009.

[38] I. Roy, S. T. V. Setty, A. Kilzer, V. Shmatikov, and E. Witchel, “Airavat: security and privacy for MapReduce,” *Proceedings of the 7th USENIX conference on Networked systems design and implementation*. USENIX Association, pp. 20–20, 2010.

[39] P. Mohan, A. Thakurta, E. Shi, D. Song, and D. Culler, “GUPT,” in *Proceedings of the 2012 international conference on Management of Data - SIGMOD ’12*, 2012, p. 349.

[40] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, “The WEKA data mining software,” *ACM SIGKDD Explor. Newsl.*, vol. 11, no. 1, p. 10, Nov. 2009.

[41] E. Frank, M. A. Hall, and I. H. Witten, *The WEKA Workbench Online Appendix for “Data Mining: Practical Machine Learning Tools and Techniques,”* Fourth Edi. Morgan Kaufmann, 2016.

[42] S. E. DeWeerdt and B. J. Culliton, “What’s a Genome?,” 2003. [Online]. Available: http://www.genomenewsnetwork.org/resources/whats_a_genome/Chp4_2.shtml#chp4#7. [Accessed: 12-Mar-2017].

[43] “Help Me Understand Genetics page: National Library of Medicine (US). Genetics Home Reference [Internet]. Bethesda (MD): The Library;,” *March 7*, 2017. [Online]. Available: https://ghr.nlm.nih.gov/primer. [Accessed: 12-Mar-2017].

[44] Z. Lin, A. B. Owen, and R. B. Altman, “Genomic Research and Human Subject Privacy,” *Science (80-.).*, vol. 305, no. 5681, pp. 183–183, Jul. 2004.

[45] N. Homer *et al.*, “Resolving Individuals Contributing Trace Amounts of DNA to Highly

Complex Mixtures Using High-Density SNP Genotyping Microarrays," *PLoS Genet.*, vol. 4, no. 8, p. e1000167, Aug. 2008.

[46] National Institutes of Health, "Modifications to Genome-Wide Association Studies (GWAS) Data Access," pp. 1–2, 2008.

[47] L. Zhang, Q. Pan, X. Wu, and X. Shi, "Building Bayesian networks from GWAS statistics based on Independence of Causal Influence," in *IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, 2016, pp. 529–532.

[48] Y. Wang, X. Wu, and X. Shi, "Using aggregate human genome data for individual identification," in *IEEE International Conference on Bioinformatics and Biomedicine*, 2013, pp. 410–415.

[49] Y. Wang, J. Wen, X. Wu, and X. Shi, "Infringement of individual privacy via mining differentially private GWAS statistics," in *Proceedings of the 2nd International Conference on Big Data Computing and Communications (BIGCOM)*, 2016, vol. 9784, pp. 355–366.

[50] X. Shi and X. Wu, "An overview of human genetic privacy," *Ann. N. Y. Acad. Sci.*, vol. 1387, no. 1, pp. 61–72, Jan. 2017.

[51] M. Kantarcioglu, Wei Jiang, Ying Liu, and B. Malin, "A Cryptographic Approach to Securely Share and Query Genomic Sequences," *IEEE Trans. Inf. Technol. Biomed.*, vol. 12, no. 5, pp. 606–617, Sep. 2008.

[52] M. Canim, M. Kantarcioglu, and B. Malin, "Secure Management of Biomedical Data With Cryptographic Hardware," *IEEE Trans. Inf. Technol. Biomed.*, vol. 16, no. 1, pp. 166–175, Jan. 2012.

[53] F. Tramèr, Z. Huang, J.-P. Hubaux, and E. Ayday, "Differential Privacy with Bounded Priors: Reconciling Utility and Privacy in Genome-Wide Association Studies."

[54] S. Simmons and B. Berger, "Realizing privacy preserving genome-wide association studies," *Bioinformatics*, vol. 32, no. 9, pp. 1293–1300, May 2016.

[55] F. Yu, M. Rybar, C. Uhler, and S. E. Fienberg, "Differentially-Private Logistic Regression for Detecting Multiple-SNP Association in GWAS Databases," Jul. 2014.

[56] B. Greshake, P. E. Bayer, H. Rausch, and J. Reda, "openSNP–A Crowdsourced Web Resource for Personal Genomics," *PLoS One*, vol. 9, no. 3, p. e89204, Mar. 2014.

[57] M. Fredrikson, E. Lantz, S. Jha, D. Page, T. Ristenpart, and S. Lin, "Privacy in Pharmacogenetics: An End-to-End Case Study of Personalized Warfarin Dosing."

[58] C.-C. Chang and C.-J. Lin, "LIBSVM: A Library for Support Vector Machines."

[59] M. Lichman, "{UCI} Machine Learning Repository," 2013. [Online]. Available: http://archive.ics.uci.edu/ml.

[60] "IPUMS International." [Online]. Available: https://international.ipums.org/international/. [Accessed: 10-Mar-2017].

VIII. APPENDIX

A. Algorithm to Release Significant Attributes Using Laplace Mechanism for χ^2 values

Input: *k*: the number of attributes to release, ε: the privacy budget, *R*: number of instances with predictor value 1, *S*: number of instances with predictor value 0, *N*: total number of instances = *R*+*S*

Output: *k* attributes and their noisy statistics

1. calculate sensitivity $t = \frac{N^2}{RS}\left(1 - \frac{1}{\max(R,S)+1}\right)$
2. add random noise with Laplace distribution (Laplace noise) with mean zero and scale $\frac{4kt}{\varepsilon}$ to the true χ^2 values
3. pick the top *k* attributes with respect to the perturbed values
4. add new Laplace noise with mean zero and scale $\frac{2kt}{\varepsilon}$ to the true statistics of *k* selected attributes in step 3
5. return *k* attributes with values calculated in step 4

B. Algorithm to Release Significant Attributes Using Laplace Mechanism for p-values

Input: *k*: number of attributes to release, ε: the privacy budget

Output: *k* attributes and their noisy statistics

1. add Laplace noise with mean zero and scale $\frac{4k}{\varepsilon}e^{-\frac{2}{3}}$ to the true p-values
2. pick the top *k* attributes with respect to the perturbed statistics
3. add new Laplace noise with mean zero and scale $\frac{2k}{\varepsilon}e^{-\frac{2}{3}}$ to the true statistics of *k* selected attributes in step 2
4. return the *k* attributes with values calculated in step 3

C. Algorithm to Release Significant Attributes Using Exponential Mechanism for p-values

Input: *k*: number of attributes to release, ε: privacy budget, τ: threshold p-value to calculate the distance score, *selectedAttribute*: empty array of size *k* to hold the selected attributes

Output: array *selectedAttribute* containing *k* relevant attributes

1. for each attribute, *m*:

   calculate the distance, *d*, for the attribute to flip its significance based on the threshold value, τ

2. for each attribute, *m*:

   score is,

   $q[m] = -d \;\; if \;\; p < \tau$ and

   $q[m] = -d - 1 \;\; if \;\; p > \tau$

3. for each attribute, *m*:

   weight based on score, $w[m] = exp(\frac{\varepsilon . q[m]}{k})$

4. for j :1 to *k*
   a. for each attribute, *m*:

      probability assigned , $p[m] = w[m]/\Sigma_i w[i]$

   b. *selectedAttribute[j]* = select an attribute based on the probability

   c. $w\big[selectedAttribute[j]\big] = 0$

   d. $p\big[selectedAttribute[j]\big] = 0$

5. return *selectedAttribute* containing *k* relevant attributes

D. Cost Function of Logistic Regression

For logistic regression, the probability that the response is true is given by $h_\theta(\boldsymbol{x_i}) = \Pr(y_i = 1|\boldsymbol{x_i}) = \frac{1}{1+\exp(-\boldsymbol{x_i^T}\omega)}$. For a binary model, the probability of response being false is then

given by $(1 - h_\theta(\boldsymbol{x_i})) = \frac{1}{1+\exp(x_i^T \omega)}$, where the independent variables vector (features) are represented by $\boldsymbol{x_i}$, and the categorical label predicted by the model is represented by $\boldsymbol{y_i}$. The parameter vector ω must be selected such that data fits the model. One approach is to identify a cost function which satisfies the below conditions. The parameters that minimizes the cost function will fit the data.

$actual\ y_i = 1$ and $h_\theta(\boldsymbol{x_i}) \to 0\ then\ \ Cost \to \infty$

$actual\ y_i = 0$ and $h_\theta(\boldsymbol{x_i}) \to 1\ then\ Cost \to \infty$

$actual\ y_i = h_\theta(\boldsymbol{x_i})\ then\ Cost = 0$

This can be attained by considering

$$Cost = \begin{cases} -\ln\big(h_\theta(\boldsymbol{x_i})\big)\ when\ y_i = 1 \\ -\ln\big(1 - h_\theta(\boldsymbol{x_i})\big)\ when\ y_i = 0 \end{cases}$$

Combining them over all the training set, the cost function can be written as

$$Cost = \sum_{i=1}^{n} -\mathrm{y_i}(\ln\big(h_\theta(\boldsymbol{x_i})\big) - (1 - y_i)(\ln\big(1 - h_\theta(\boldsymbol{x_i})\big))$$

Replacing $h_\theta(\boldsymbol{x_i})$

$$Cost = \sum_{i=1}^{n} -\mathrm{y_i}(\ln\left(\frac{1}{1 + \exp(-\boldsymbol{x_i^T}\omega)}\right)) - (1 - y_i)(\ln\left(\frac{1}{1 + \exp(\boldsymbol{x_i^T}\omega)}\right))$$

$$\Rightarrow Cost = \sum_{i=1}^{n} ln(1 + \exp(x_i^T \omega)) - y_i x_i^T \omega$$

E. Algorithm to Build Logistic Regression Model Using Functional Mechanism

Input: *D*: Sensitive dataset, ε: privacy budget, *d:* number of attributes of dataset

Output: $\bar{\omega}$, the parameter vector for logistic regression as identified by functional mechanism

1. build the approximate objective function

$$\tilde{f}_D(\omega) = \sum_{i=1}^{n} (\log 2) * \left(x_i^T \omega\right)^2 + (y_i - 0.5)\, x_i^T \omega + \frac{1}{8}$$

2. set sensitivity, $\Delta = \frac{d^2}{4} + d$
3. for each coefficient of ω , λ_φ:

   set $\lambda_\varphi = \sum_{t_i \epsilon D} \lambda_{\varphi t_i} + Lap\left(0, \frac{\Delta}{\varepsilon}\right)$
4. perform regularization and spectral trimming to make $\tilde{f}_D(\omega)$ positive definite
5. compute $\bar{\omega} = arg\ min_\omega \tilde{f}_D(\omega)$
6. return $\bar{\omega}$

F. Algorithm to Build Logistic Regression Model Using Exponential Mechanism

Input: *D*: Sensitive dataset, *d*: number of attributes and instances of sensitive dataset, *n*: number of instances of sensitive dataset respectively, ε: privacy budget, σ: the mutation scale, *s*: the mutation resize value, Ω : candidate set, ω: intermittent selected vector

Output: $\bar{\omega}$, the parameter vector for logistic regression as identified by PrivGene

1. initialize ω with a randomly generated vector
2. set the number of iterations $j = 0.125 * n * \varepsilon$
3. set privacy budget per iteration $\varepsilon_s = \varepsilon/j$
4. for *j* iterations:
   a. generate candidate set, Ω, of 2*d* vectors such that it contains

      d mutations of ω by adding σ noise to each element of ω one at a time

      d mutations of ω by adding -σ noise to each element of ω one at a time
   b. set sensitivity, $\Delta = 4 * \sigma$
   c. set ω = DP_Select(D, f, Ω, ε_s, Δ)

d. $\sigma = \sigma * s$

end for

5. set $\bar{\omega} = \omega$
6. return $\bar{\omega}$

DP_Select(D, f, Ω, ε_s, , Δ)

Input: *D*: Sensitive dataset, *f*: fitting function of logistic regression, Ω: candidate set of parameter vectors, ε_s: privacy budget

Output: ω^*: selected parameter vector

1. for each mutation, $\ddot{\omega} \in \Omega$:

   score, $q[\ddot{\omega}] = -f(D, \ddot{\omega})$

2. for each mutation, $\ddot{\omega}$:

   weight based on score, $w[\ddot{\omega}] = exp(\frac{\varepsilon_s * q[\ddot{\omega}]}{\Delta})$

3. for each mutation, $\ddot{\omega}$:

   probability assigned , $p[\ddot{\omega}] = w[\ddot{\omega}]/\Sigma_i w[i]$

4. ω^* = select a mutation based on the probability
5. return ω^*

G. Access DPWeka

The DPWeka package implemented for this thesis is available at *https://github.com/NidhiKat/DPWeka*. For further details on the project, please refer to *http://csce.uark.edu/~xintaowu/DPWeka/index.htm*.