SECURE TWO-PARTY PROTOCOL FOR PRIVACY-PRESERVING CLASSIFICATION VIA DIFFERENTIAL PRIVACY

by
Manish Kumar

A thesis

submitted in partial fulfillment

of the requirements for the degree of

Master of Science in Computer Science

Boise State University

December 2019

BOISE STATE UNIVERSITY GRADUATE COLLEGE

DEFENSE COMMITTEE AND FINAL READING APPROVALS

of the thesis submitted by

Manish Kumar

Thesis Title: Secure Two-Party Protocol for Privacy-Preserving Classification via Differential Privacy

Date of Final Oral Examination: 08 August 2019

The following individuals read and discussed the thesis submitted by student Manish Kumar, and they evaluated the presentation and response to questions during the final oral examination. They found that the student passed the final oral examination.

Gaby Dagher, Ph.D.	Chair, Supervisory Committee
Bogdan Dit, Ph.D.	Member, Supervisory Committee
Jyh-haw Yeh, Ph.D.	Member, Supervisory Committee

The final reading approval of the thesis was granted by Gaby Dagher, Ph.D., Chair of the Supervisory Committee. The thesis was approved by the Graduate College.

In loving memory of “my father”

ACKNOWLEDGMENTS

I'm extremely grateful and thankful to my advisor Dr. Gaby Dagher. My success in completing this thesis would not have been possible without his continuous guidance, tireless effort, amazing insights, encouragement, and gift of opportunities to me throughout this research. His academic excellence and technical expertise were very helpful throughout the research and in the writing. His generous attitude made the research journey comfortable and smooth.

I would also like to thank my committee members, Dr. Bogdan Dit and Dr. Jyh-haw Yeh for their great support and invaluable advice during this journey.

I am fortunate to be a part of Boise State University with an incredible group of fellow students. I am grateful to them for their support and help throughout my graduate studies, especially, my esteemed friends Joshua Holmes and Anil Acharya.

Last but not the least, I would like to express deepest gratitude to my always supportive parents and other family members. Most importantly, I wish to thank my loving and supportive wife, Manisha, who is always there for me at any situation. It is because of her I am able to achieve this success. Special mention goes to my source of inspiration my lovely daughter, Miraaya whose arrival in the middle of my graduate studies changed my perception of life and happiness.

ABSTRACT

Privacy-preserving distributed data mining is the study of mining on distributed data—owned by multiple data owners—in a non-secure environment, where the mining protocol does not reveal any sensitive information to the data owners, the individual privacy is preserved, and the output mining model is practically useful. In this thesis, we propose a secure two-party protocol for building a privacy-preserving decision tree classifier over distributed data using differential privacy. We utilize secure multiparty computation to ensure that the protocol is privacy-preserving. Our algorithm also utilizes parallel and sequential compositions, and applies distributed exponential mechanism to ensure that the output is differentially-private. We implemented our protocol in a distributed environment on real-life data, and the experimental results show that the protocol produces decision tree classifiers with high utility while being reasonably efficient and scalable.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

PPDP – Privacy-Preserving Data Publishing

PPDM – Privacy-Preserving Data Mining

DPPDM – Distributed Privacy-Preserving Data Mining

DPPDP – Distributed Privacy-Preserving Data Publishing

S-DPDT – Secure-Differentially-Private Decision Tree

LIST OF SYMBOLS

$\mathcal{D}$	denotes dataset
$[\![\mathcal{D}]\!]$	denotes encrypted dataset
D_1	denotes party 1 dataset
D_2	denotes party 2 dataset
$\mathcal{P}_1$	denotes the Party 1
$\mathcal{P}_2$	denotes the Party 2
A	denotes attribute
n	denotes total number of attributes
c	denotes class value
cls	denotes class attribute
B	denotes budget
ϵ	denotes epsilon
$\mathbb{T}$	denotes the differentially-private decision tree
IG_A	denotes information gain of an attribute
C_A^c	denotes the count table
A^w	denotes the winner attribute
X_w	denotes the winner split for an attribute

X_k denotes split combination pairs of an attribute

π denotes permutation

$\mathbb{T}^{\pi}_{A,c}$ denotes shuffled and randomize class count table of an attribute

v denotes the split value

Ω denotes domain

lc denotes left child

rc denotes right child

CHAPTER 1

INTRODUCTION

Due to the increase in data collection and storage technology, more information about individuals, such as personal, financial and health, are being collected and analyzed. When distributed data about individuals is brought together, businesses, organizations and government agencies can mine it to extract knowledge and learn patterns and behaviors about individuals, to provide better services and make better decisions. The term *data mining* refers to a wide range of techniques, including classification, clustering, pattern mining and regression models. When data about the same set of individuals is distributed between distrusting data owners, preserving the privacy of their data–no sensitive information about individuals is revealed–becomes a major concern if they collectively decide to mine their data. The main challenge in this protocol is *privacy-preservation*, during the execution of the protocol as well as after the generation of output. The goal here is to build a secure two party protocol that remains secure from beginning to end of the whole execution and how to securely build a privacy-preserving decision tree classifier that does not leak information about the individuals in the original data. A straightforward construction of a decision tree on the distributed and private data between two parties, without applying any anonymization technique, would typically lead to information leakage based on the output classifier only.

Distributed privacy-preserving data mining (DPPDM) is an advanced research area in

data mining where two or more parties jointly execute a protocol to perform data mining tasks on their private data such that none of the participating parties learn anything new from the protocol besides the final output. DPPDM is a difficult task to achieve, due to the requirement that protocol should be secure and privacy-preserving, and that the individual's identity and sensitive information remain protected against adversaries after the data mining task is computed. To achieve anonymity, DPPDM protocols usually utilize different privacy models such as k-anonymity, ℓ-diversity, *LKC*-privacy, and differential privacy. On the other hand, adversaries typically use linkage attacks on the output to trace back personal and sensitive information about individuals. But *differential privacy* stands out from the other privacy-preserving techniques mentioned above. Differential-privacy guarantees that any individual data used in a data mining task will not be revealed during or after the execution of protocol. In other words, whether an individual's data participates in the dataset or not the outcome of the data mining task will not be affected by its absence or presence.

1.1 Motivation

Digital data about individuals has ballooned, with more sensitive and personal details about the same individuals distributed at several places. Integrating the individuals data and then applying data mining tools to build a data mining model will have better accuracy than single party data based mining models. In a data mining classification technique, the more records we have, the higher the chance of achieving better accuracy and this is why it is called a supervised learning technique. The distributed data could be vertically partitioned where each record is of the same individual with different attributes in the datasets or horizontally partitioned for data with the same set of attributes but with a different set

of records.

Example 1.1.1. A hospital and a pathological clinic have the records for the same patients, with a different set of attributes, except a common patient ID. If we build a data mining model for predicting whether the patients are vulnerable to a life threatening disease or not and simply apply data mining tools on their separate data, the result might not be very convincing. On the other hand, if we build a model by integrating their data or jointly use both datasets in building the mining model, the output model will predict with significantly better accuracy.

Example 1.1.2. Assume that two credit card companies have a different set of records of their consumers with the same attributes. They jointly want to build a decision making data model to make decisions for new applicants for credit cards. Since the classifier accuracy depends upon a training data, the larger the data size the better the prediction will be. So, the integration of horizontal datasets will provide a large training data when compared to the isolated data of a single party.

At the same time, distributed data of the individual will provide a data miner great flexibility in achieving high accuracy through their data model rather than a single party's data. Integrating the private data owned by different data owners and achieving data mining tasks will solve many problems but will raise data privacy concerns for the data owners.

1.2 Challenges & Concerns

In this thesis, we propose a DPPDM protocol to construct a decision tree-based classifier over two-party private data. The protocol is secure (privacy-preserving) in the semi-honest adversarial model, where both parties are assumed to correctly execute the protocol but may try to infer information about each other's data during the execution of the protocol.

The main challenge in this protocol is *privacy-preservation*. That is, how to securely build a privacy-preserving decision tree classifier that does not leak information about the individuals in the original data. A straightforward construction of a decision tree on the distributed and private data between two parties, without applying any anonymization technique, would typically lead to information leakage based on the output classifier only, as illustrated in Example 1.2.1

Example 1.2.1. Let $\mathcal{P}_1$ and $\mathcal{P}_2$ be two data owners owning datasets D_1=(UID, Age, Job, $Class$) and D_2=(UID, $MaritalStatus$, $Class$), where UID and $Class$ are shared attributes between the two datasets, as shown in the Table 1.1. The data in each row corresponds to the same individual, but it is split between the two datasets.

Let $\mathbb{T}$ be the decision tree classifier which $\mathcal{P}_1$ and $\mathcal{P}_2$ jointly constructed based on D_1 and D_2, as shown in Figure 1.1. Even if the protocol for constructing $\mathbb{T}$ is privacy-preserving, the two parties can learn sensitive information about each others data merely based on $\mathbb{T}$. More specifically, $\mathcal{P}_2$ can learn that all *Assistants* are female and all *Professors* are male in the $\mathcal{P}_1$ dataset. Since $Class$ attribute is also known to $\mathcal{P}_1$, it can also learn that none of the males in D_2 is an *Assistant*, and none of the females in D_2 is a *Professor*. ■

On the other hand, the application of an anonymization technique, such as k-anonymity or ℓ-diversity, to construct an anonymous decision tree on the distributed and private data might also have the risk of information leakage from the output classifier, as illustrated in Example 1.2.2.

Example 1.2.2. Assume that $\mathcal{P}_1$ and $\mathcal{P}_2$ want to utilize k-anonymity to construct the decision tree classifier $\mathbb{T}$. Let Table 1.2 represent the 2-anonymous data used by $\mathcal{P}_1$ and $\mathcal{P}_2$ to jointly construct the decision tree classifier $\mathbb{T}$ illustrated in Figure 1.1. Although $\mathbb{T}$ is 2-anonymous, the two parties can still learn sensitive information about each others data.

Table 1.1: Raw datasets D_1 and D_2 owned by parties $\mathcal{P}_1$ and $\mathcal{P}_2$ respectively.

Shared		D_1		D_2
UID	*Class*	Age	Job	Marital Status
1	Male	54	Professor	Divorced
2	Female	26	Student	Single
3	Female	39	Assistant	Married
4	Male	67	Professor	Married
5	Female	32	Assistant	Married
6	Male	40	Engineer	Divorced
7	Male	50	Student	Divorced
8	Female	29	Student	Married
9	Female	43	Engineer	Married
10	Male	46	Student	Single

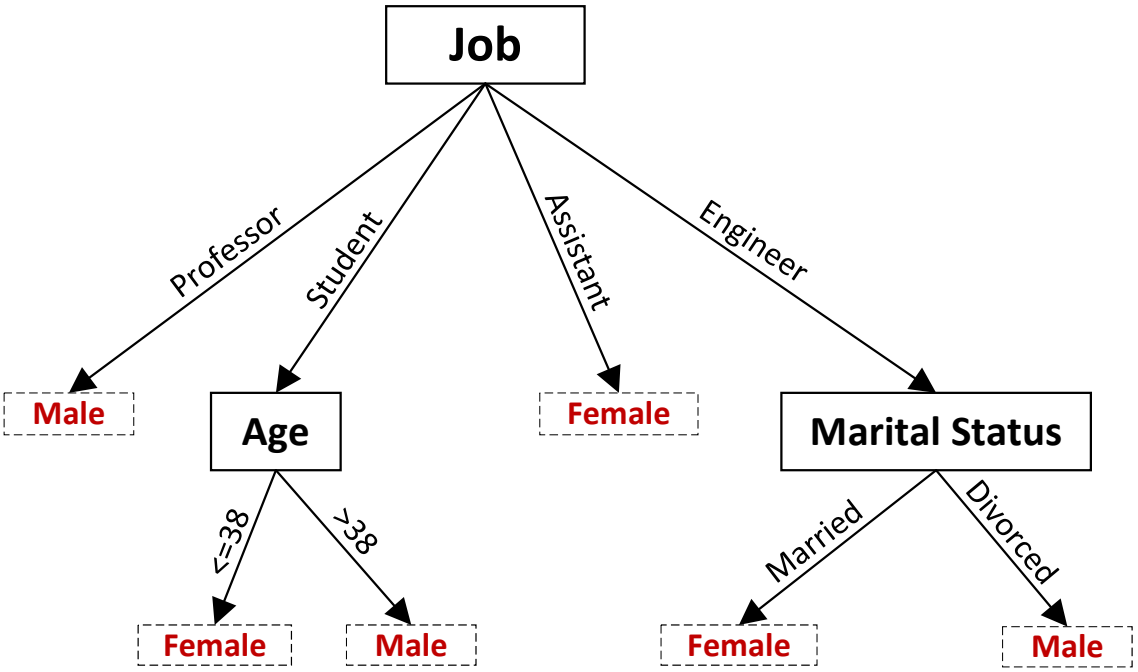

Figure 1.1: Decision tree based on datasets D_1 and D_2 from Table 1.1

For example, $\mathcal{P}_2$ can learn that the *age* of all males in its dataset D_1 is above 45, and the *age* of all females in D_1 is less or equal to 45. ■

Table 1.2: Anonymized datasets $\hat{D}_1$ and $\hat{D}_2$ based on raw data from Table 1.1.

Shared		$\hat{D}_1$		$\hat{D}_2$
UID	*Class*	Age	Job	Marital Status
1	Male	>45	Professional	Any
2	Female	<=45	Non-Professional	Any
3	Female	<=45	Non-Professional	Any
4	Male	>45	Professional	Any
5	Female	<=45	Non-Professional	Any
6	Male	<=45	Professional	Any
7	Male	>45	Non-Professional	Any
8	Female	<=45	Non-Professional	Any
9	Female	<=45	Professional	Any
10	Male	>45	Non-Professional	Any

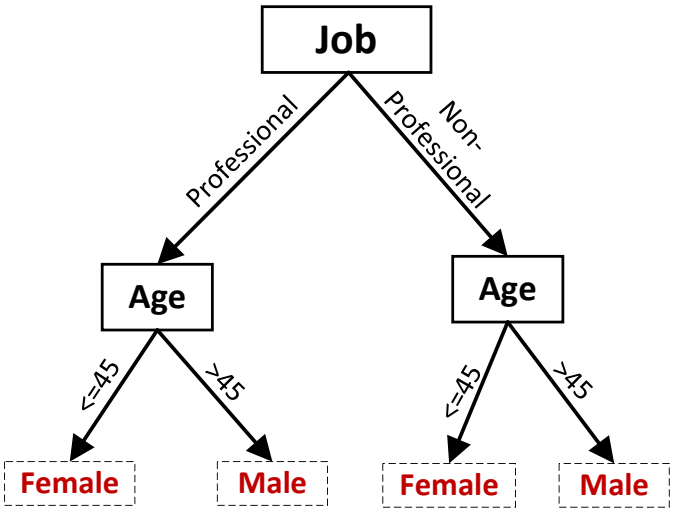

Figure 1.2: Anonymous decision tree based on 2-anonymous data from Table 1.2

Several works in the literature proposed to protect data privacy while performing distributed data mining tasks. Lindell and Pinkas [35][36] proposed to build a decision tree from distributed data between two parties, and Emekci *et al.* [22] gave protocol to build decision trees over distributed data among multiple data owners based on secure multi-party computation protocol [27]. Agrawal and Srikant [1] gave protocol to add

noise before performing a data mining task, and Blum et al. [6] proposed a protocol called *SuLQ* primitive which can be used for various data mining tasks. McSherry [39] also uses differential privacy to achieve privacy-preserving output, similarly, Friedman and Schuster [23] proposed a protocol to build decision trees using an exponential mechanism. Most recently, Vaidya *et al* [52] has proposed a protocol to build decision tree classifiers over distributed data among multiple data owners.

Other works in the literature used different privacy-preserving techniques. Nonetheless, most of these protocols are vulnerable to either attribute or record linkage attacks due to the fact that there is no privacy guarantee on the output. Several researchers proposed different protocols to prevent such attacks. Mohammed *et al.* [42] and Jiang and Clifton [30] introduced protocol to generalize data attributes using k-anonymity [48]. Later, Machanavajjhala *et al.* [37] showed that protocols based on k-anonymity are vulnerable to background attacks and presents a protocol based on ℓ-diversity to achieve privacy. However, several researchers found that these protocols are vulnerable to different kinds of privacy violating attacks, and lack in providing sufficient data privacy.

To achieve privacy-preserving distributed data mining, there is a need for robust protocols that provides strong immunity against linkage and statistical attacks. In this thesis, we utilize differential privacy [18], a privacy model that ensures rigorous privacy on the output irrespective of an adversary's background knowledge. Differential privacy provides a strong guarantee that the outcome of the query will not be affected significantly based on whether or not the record of a participating individual is included in the data. In other words, the output datasets, with or without the participant's data, will be almost identical, and hence guarantee that the individual's privacy is not at risk. We named our approach Secure-Differentially-private decision tree (S-DPDT) because the protocol is secure from beginning to end with no private information leakage during or after the execution of

protocol. The output decision tree we get is also differentially-private.

The synopsis of this thesis can be viewed as follows:

- We propose S-DPDT, a top-down secure two-party protocol for building a differentially-private decision tree from two datasets without revealing any sensitive information to the other party.
- For both numerical and categorical attributes, we propose an exponential mechanism-based protocol to efficiently split the data in a differentially-private manner.
- We utilize exponential ElGamal encryption schemes to share the participants' data and ensure the protocol is privacy-preserving.
- We implemented our approach and performed exhaustive experiments on real-life data with a sufficient number of categorical and numerical attributes. The results show that our approach achieves high utility, is scalable and efficient with respect to number of records and attributes in the data.

1.3 Thesis Statement

The objective of this thesis is to answer the following question: **Given two data owners, how can they perform a data mining task on their distributed data such that data privacy is maintained and the output model is privacy-preserving?**

More specifically, given two datasets D_1 and D_2 owned by $\mathcal{P}_1$ and $\mathcal{P}_2$ respectively, and a privacy budget ϵ, the goal of this thesis is to design a protocol for constructing a privacy-preserving decision tree-based classifier $\mathbb{T}$ for the purpose of data mining and analysis such that:

1. The protocol is secure (privacy-preserving) in the semi-honest adversarial model.

2. The output classifier $\mathbb{T}$ is ϵ-differentially-private while maintaining high utility.

3. The proposed approach is scalable and efficient.

1.4 Organization of the Thesis

The thesis is organized as follows:

- Chapter 2 introduces the preliminaries required for the protocols such as classification in data mining using a decision tree, several algorithms for building a decision tree, differential privacy, exponential mechanism, and several other cryptographic primitives with homomorphic encryption such as distributed exponential ElGamal, mix and match and plain-text equality text.

- Chapter 3 is an in-depth literature review of the previous privacy-preserving algorithms, categories into privacy-preserving data mining with interactive and non-interactive approach for centralized datasets, privacy-preserving data mining for distributed datasets and privacy-preserving data publishing for centralized and distributed datasets.

- Chapter 4 provides details of the proposed protocol for secure and distributed privacy-preserving data mining tasks. We proposed a secure two-party differentially-private protocol for building a decision tree, which not only preserves data privacy and security during the computation but the output it generates is differentially-private.

- Chapter 5 provides the algorithm analysis of our proposed approach and discusses the privacy and complexity analysis of our approach.

- Chapter 6 illustrates the experiments and performance evaluation of our proposed solution.
- Chapter 7 concludes our work and discusses the future work.

CHAPTER 2

BACKGROUND

This chapter provides details of preliminary components required for achieving the objective of building a decision tree, applying cryptographic schemes, and achieving differential privacy on the output decision tree.

2.1 Classification via Decision Tree

Classification is one of the important tasks in data mining. Given a set of records, called a training data set, where each record in a row has multiple attributes (categorical or numerical) and one class attribute which is categorical that categories the record. The goal of classification is to build a classifier model that will assign a correct class label to new data being inserted in the record set without a known class label. There are several types of classification algorithms that are widely used, such as, statistical models, genetic models, neural networks and decision trees.

A decision tree [46] is one of the most popular and inexpensive classification algorithms used in many real world applications for decision making, such as banking (credit card decision making, loan sanction), stock buying, and planning and disease diagnosis, etc.

There are several decision tree algorithms that have been developed over the years such as Iterative Dichotomiser 3 (ID3) [46], Improved ID3 (C4.5) [47], Classification and Regression Tree (CART) [9] algorithm, Chi-squared Automatic Interaction Detector

(CHAID) [32], and MARS [24] developed for handling numerical data. A decision tree is based on a greedy algorithm approach in which the dataset is recursively split into sub-datasets based on attribute value as the node moves from the top down until the leaf nodes are left with a class value. There are several methods to determine the best attribute to split the dataset, including Gini Index, Information Gain, Max Operator and Gain Ratio, etc.

- **Gini Index** [9]. It is an impurity measure algorithm mainly used by CART algorithm. It computes the probability of choosing an incorrect class value for an item when the class value is randomly picked from a majority of class value for a given attribute. Gini index is biased towards an attribute with large domain size.

$$Gini(D) = 1 - \sum_{cls} p_{cls}^2$$

The gini index computes the binary split of attribute A, such as D_1 and D_2. The gini index of D is given as:

$$Gini_A(D) = \frac{D_1}{D} Gini(D_1) + \frac{D_2}{D} Gini(D_2)$$

The reduction in impurity is calculated as:

$$\Delta Gini(A) = Gini(D) - Gini_A(D)$$

The attribute that maximizes the reduction in impurity is chosen as the split attribute.

- **Information gain** [46]. It is the method to split the dataset based on the maximum information given by an attribute. Information gain is expressed as the entropy of current item minus the total entropy of child nodes created by splitting current item for each value of an attribute. Information Gain(IG) of each attribute A is computed as:

$$E(T) = \sum -\frac{|T(c)|}{|T|} \lg \frac{|T(c)|}{|T|}$$

$$E(T|A) = \sum_{a \in A} -\frac{|T(a)|}{|T|} E(T(a))$$

$$IG(A) = E(T) - E(T|A)$$

- **Max Operator** [9]. This function corresponds to the node misclassification rate by picking the class with the highest frequency:

$$_{qMax}(\mathcal{T}, A) = \sum_{j \in A} (max(\mathcal{T}_{j,c}^{A}))$$

The sensitivity of this function is $S(qMax) = 1$

- **Gain Ratio** [47] the gain ratio is determined by dividing the information gain by information value. It reduces the bias of information gain towards the large multi-valued attributes. Used in improved ID3 (C4.5). Information value is defined as:

$$IV(A) = -\sum_{j \in A} \frac{\mathcal{T}_{j,c}^{A}}{\mathcal{T}} . log \frac{\mathcal{T}_{j,c}^{A}}{\mathcal{T}}$$

$$GainRatio(A) = \frac{Gain(A)}{IV(A)}$$

Whenever, *IV(A)*≈0, when $\mathcal{T}_{j,c}^{A} \approx \mathcal{T}$, the gain ratio becomes very large or undefined. The attribute with the highest gain ratio is taken as the split attribute.

2.2 Differential Privacy

Differential privacy is a strong privacy model introduced by Dwork *et al.* [19] for the purpose of preserving data confidentiality without making assumptions about the attacker's background knowledge. "You will not be affected, adversely or otherwise, by allowing your data to be used in any analysis of the data, no matter what other analyses, datasets, or information sources are available" [20]. Differential privacy ensures a strong guarantee to the participating individual in the dataset that their presence or absence will not affect the final output of the query significantly. That is, the output dataset with his data or without his data will be almost identical.

2.2.1 ε-Differential Privacy

Definition 2.2.1. ***ε-Differential Privacy [19].*** Given any two neighboring datasets D_1 and D_2 that differ on at most one record, a sanitizing mechanism M preserves ε-differential privacy if for any output $\hat{D} \in Range(M)$:

$$Pr[M(D_1) = \hat{D}] \leq e^{\varepsilon} \times Pr[M(D_2) = \hat{D}] \tag{2.1}$$

where the probabilities are taken over the randomness of M. □

To achieve differential privacy a random noise is added to the actual output, the noise value is determined with standard deviation with sensitivity of the function. Noise value changes accordingly when the input value is changed.

2.2.2 Sensitivity

Random noise is added to the dataset to achieve differential privacy by getting perturbed output. The quantity of noise added depends upon the sensitivity of the function.

Definition 2.2.2. ***Global Sensitivity [19].*** Given a query function $f : D \rightarrow \mathbb{R}^d$, the global sensitivity of f is:

$$S(f) = \max_{D_1, D_2} ||f(D_1) - f(D_2)||_1 \tag{2.2}$$

where D_1 and D_2 are any two neighboring datasets that differ on at most one record. □

2.2.3 Lapalce Distribution

The Laplace mechanism, proposed by Dwork *et al.* [19], adds random noise to the actual data, following the Laplace statistical distribution to determine the amount of noise to be added. In simple language this is how much noise to be added to the data so that the perturbed data does not lose it's utility. It is directly proportional to its standard deviation or noisiness. To achieve differential privacy when using a query function f, the principal approach is to perturb the true output of f by adding a random noise that is adjusted based on $S(f)$. The authors propose to generate the noise according to Laplace distribution, $Lap(\lambda)$, where the probability distribution function is

$Pr(x|\lambda) = \frac{1}{2\lambda} e^{\frac{|x|}{\lambda}}$,

the mean is 0, and the standard deviation is λ which is determined based on the global sensitivity $S(f)$ and the privacy level ε.

Theorem 2.2.1. ***[19].*** For any function $f : D \rightarrow \mathbb{R}^d$ that maps datasets to reals, the privacy mechanism M:

$$M(D) = f(D) + Lap(S(f)/\varepsilon) \tag{2.3}$$

satisfies ε-differential privacy. □

2.2.4 Exponential Mechanism

Proposed by McSherry and Talwar [38], exponential mechanism is one of the popular methods to achieve differential privacy without adding noise to output. There are several types of datasets that are not sensitive to noise addition meaning perturbation doesn't make sufficient change in the data. Thus, to achieve differential privacy in this type of dataset, exponential mechanism plays an important role. It allows the user to choose an output from a range of outputs which are close to optimum or best value with respect to a utility function while preserving differential privacy.

It takes input as follows: a data set D, output range $\mathcal{T}$, privacy parameter ϵ, and a utility function $u : (D \times \mathcal{T}) \rightarrow \mathbb{R}$ that assigns a real value to each output $t \in \mathcal{T}$, where a higher score means better utility.

The mechanism further applies probability distribution function over the range of output $\mathcal{T}$ and then result out an output t.

Let $\Delta u = max_{\forall t,D,D'}|u(D,t) - u(D',t)|$ be the sensitivity of the utility function.

The probability associated with each output is proportional to,

$$t \propto exp(\frac{\epsilon u(D,t)}{2\Delta u}); \tag{2.4}$$

i.e. output with higher score is exponentially more likely to be chosen.

Theorem 2.2.2. *exponential mechanism [38] for any function* $u : (D \times \mathcal{T}) \rightarrow \mathbb{R}$*, a utility algorithm A that chooses an output t with a probability proportional to* $exp(\frac{\epsilon u(D,t)}{2\Delta u})$ *satisfy the* ϵ*-differential privacy.* □

2.2.5 Distributed Exponential Mechanism

Proposed by Mohammad *et al.* [40], distributed exponential mechanism is used to achieve differential privacy when the data of the same individuals is distributed between two parties in a vertically partitioned fashion. Hence they introduced distributed exponential mechanism to determine the winner attribute based on the utility score generated by each party for each attribute they own in a secure fashion so that no information leakage happens except the winner attribute. The mechanism takes input (v_i, u_i) pairs where v_i represents candidates and u_i represents the scores. The score is calculated using utility function, in this case information gain is to calculate the score. Both parties calculate the score locally for each of their attributes. Then they jointly execute distributed exponential mechanism to determine the winner attribute and winner party.

2.3 Cryptographic Primitives

2.3.1 Encryption Scheme

Our protocol requires an additive homomorphic encryption scheme that allows ciphertexts to be re-randomized without revealing any private information to participating parties. Ours is a secure two-party protocol. Consequently, it must acknowledge the distributed key generation (DKG) and the distributed decryption, enabling participants to use key shares to perform a decryption operation. The best fit for our work is a variation of ElGamal [21] called Exponential ElGamal [12], it is fast when implemented over elliptic

curves, distributed key generation is straightforward, and decryption is feasible for our plain-text space. For brevity, we denote the encryption of a message m as $[\![m]\!]$. To this extent, the encrypted dataset in this thesis is denoted as mentioned in previous lines like dataset D referred $[\![D]\!]$ after encryption.

2.3.2 ElGamal Encryption

The ElGamal encryption scheme was proposed by Tahir El Gamal [21] which belongs to the family of public key cryptosystems based on discrete logarithmic problems. The key is separated in two parts, the public key and the private key. Knowledge of the private key is important for decryption, otherwise it will be almost impossible to decrypt the message in appropriate time line. An attacker who tries to break the encryption always looks to access the private key, else, he needs to compute the logarithm problem to decrypt.

2.3.3 Distributed key generation

Distributed key generation [26] is the principle building block of any symmetric and asymmetric threshold cyrptosystem [15][16]. In this encryption procedure two or more parties together compute and generate shared public and private key pair sets. The honest participating parties follow the threshold to determine the generation of a key pair.

2.3.4 Distributed Exponential ElGamal Decryption

In Distributed Exponential ElGamal, [8] n parties generate a private key x. Given ElGamal cipher-text (α, β), where the secret key $x \in \mathbb{Z}_p$ is shared between n parties according to (k, n)-threshold such that $k \leq n$, each participant $\mathcal{P}_i$ from any group of k participants $\mathcal{P}_1, \ldots, \mathcal{P}_k$ publishes β^{x_i}, where x_i is a private key share of $\mathcal{P}_i$. The plain-text can then be

derived by computing: $\alpha / \prod_{i=1}^{n} \beta^{x_i}$. For the purpose of this proposal, we assume that x is shared according to $(2, n)$-threshold.

2.3.5 Mix and Match

Proposed by Jakobsson and Juels, Mix and Match [28] is a secure multi-party computation protocol for obliviously evaluating an input against a lookup table, where all values are encrypted with exponential ElGamal as mentioned in our previous section 2.3.1. Mix and Match alone could realize our entire protocol given that lookup tables are sufficient for implementing general computing. However, such an approach would be expensive, i.e., the complexity would be exponential in the number of input variables. In our protocol, we use a two column noisy mix and match table with single-input lookup tables sparingly. Like our overall protocol, Mix and Match itself is publicly verifiable, secure against malicious adversaries, and secure with a dishonest majority.

2.3.6 Plaintext Equality Test (PET)

Computation in Mix and Match is a matching of cipher-text. *PET* provides a method to participating parties to determine, in a distributed fashion, whether the two cipher-text is or is not the same plain-text. *PET* uses algebraic division operations on the given two cipher-text to match them. If the two cipher-text is of the same plain-text the output of the division will be an encryption of 1, hence a match is found. Otherwise, the two cipher-text is of different plain-text. Let (x, y) and (x', y') be the ElGamal cipher-texts for plain-text l_1 and l_2 respectively. With PET protocol two parties jointly check whether $l_1 = l_2$ i.e whether $(x/x', y/y') = [\![1]\!]$.

2.3.7 Mix Network

A mix network, introduced by Chaum [29], is a multi-party protocol which allows a sequence of encrypted messages to be taken as input and generates a new shuffled and re-randomized output list, such that none of the participants know the permutation mapping of the corresponding inputs to outputs (except participants own contribution). A verifiable mix network produces a publicly verifiable result, that can be verified and proven that the output list is true (i.e., the messages were only randomized, not modified nor fabricated). The security of the Mix network lies in the feasibility of adversaries to find which output belongs to which input.

CHAPTER 3

LITERATURE REVIEW

In this chapter, we illustrate the related work and present an abstract view of related papers into Table 1.1 for comparative evaluation based on different main features with our proposed work.

A lot of work has been done in data privacy over the years for effective data mining from private and sensitive data. Researchers have come up with several good techniques to achieve privacy of data, privacy of individual identity, and sensitive data. Privacy could be achieved by many existing protocols like *differential privacy* [18], *k-anonymity* [48], *LKC-privacy* [43], and ℓ*-Diversity* [37], (a few of the widely used and efficient). However, not all the protocols provide guaranteed privacy of the dataset, but the *differential privacy* stands out from them and provides better privacy of the dataset. Such that the data of an individual entity was not present at all in the dataset.

Data Privacy is a broader concept and can be distributed into groups based on the protocols, privacy, security, dataset, and output.

3.1 Privacy-Preserving Data Mining (PPDM)

In privacy-preserving data mining, a single party applies a privacy model on the input data in order to perform a data mining task, i.e. building a decision tree, and output a privacy-preserving result. In PPDM, the data miner performs queries or mining operations

on the data and establishes, through his protocol, that the result of the query or mining operation will not violate the privacy of entities in the data. Over the years several approaches in the literature have been proposed for achieving privacy-preserving data mining, where [1][23][5][49] propose non-interactive approaches and [6][39][45][57][49][34] propose interactive approaches. Figure 3.1 and Figure 3.4 show privacy-preserving data mining in a non-interactive and interactive structure.

3.1.1 Non-Interactive PPDM

Agrawal and Srikant were the first to introduce the concept of privacy-preserving data mining in [1], where they present an approach to construct a privacy-preserving decision tree. They were first to introduce to perturb the data using Gaussian and uniform distribution, and then construct the decision tree on the perturbed data. Friedman and Schuster [23] proposed a decision tree classifier based on ID3 with differential privacy called *DiffPID3*, with limited budget allocation to the data miner, and used *Exponential mechanism* to achieve differential privacy. They used different quality functions which affects the sensitivity of the split- *information gain, Gini index and Max operator*, for finding the best split for the attribute for a decision tree induction. According to them, *Information gain* is the most sensitive to noise and *max operator* function is least sensitive to noise. Balu *et al.* [5] introduces a recommender system that uses differential privacy in a non-interactive manner to sanitize the user profile before publishing. To achieve privacy they use two differentially-private non-interactive mechanisms for profile representation, Bloom-and-Flip (BLIP) [2] and Johnson-Lindenstrauss Transform (JLT) [33].

Su *et al.* [49] presents a non-interactive approach called Extended Uniform Grid K-means algorithm to generate differentially-private K-means clustering.

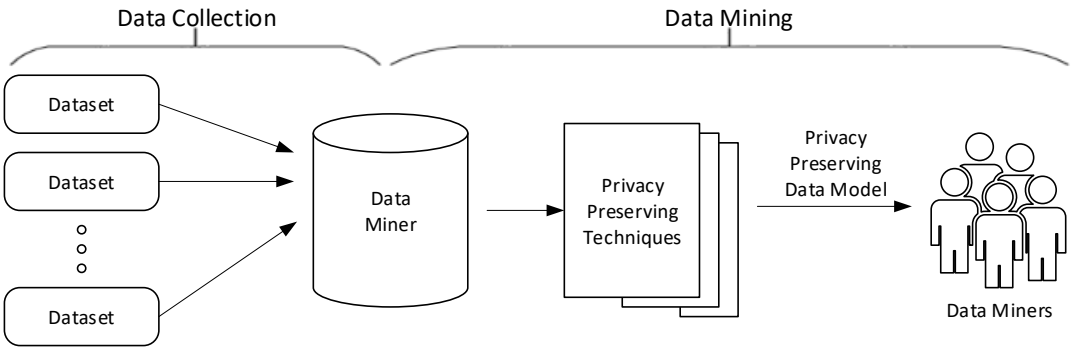

Figure 3.1: Non-Interactive Privacy-Preserving Data Mining

3.1.2 Interactive PPDM

Blum *et al.* [6] propose a protocol called *(Sub-Linear Queries) SuLQ* primitive, which results in a noisy output to the called queries to the database. Their approach is an interactive one which means each time a query is fired to a database, the query function adds a little noise to the output. They used this *SuLQ primitive* to various data mining protocols like *k-means clustering* called *DPLloyd*, *principal component analysis*, *the perception algorithm*, and the *ID3 classifier*, etc. and found the output to be a differentially-private. McSherry [39] presents a programming kind of a data analysis language called *Privacy Integrated queries* (PINQ) platform for privacy-preserving data analysis. It is a SQL kind of interactive language built over $C\sharp's$ LINQ. PINQ uses PINQueryable$\langle T \rangle$ which supports aggregations and transformations. PINQ uses a PINQAgent which takes a method, Alert (ε) as an input for generating a differentially-private aggregation with respect to its immediate data source. Mohan *et al.* [45] present a system called *GUPT* which uses a sample and an aggregate framework for achieving differential privacy. Zhang *et al.* [57] introduce a general purpose differentially-private model based on a genetic algorithm called *PrivGene*. PrivGene uses an exponential mechanism to achieve differential privacy. Su *et al.* [49] also

proposes an *improved DPLlyod* an interactive approach based on DPLlyod in same paper. Kotsogiannis *et al.* [34] introduce a meta-algorithm that selects a differentially-private algorithm based on a dataset and then returns a differentially-private result. The proposed method works in three steps, it extracts a set of features values from the input dataset, then selects an algorithm to be applied on the extracted set of features and at the end, it executes the algorithm to get the differentially-private output.

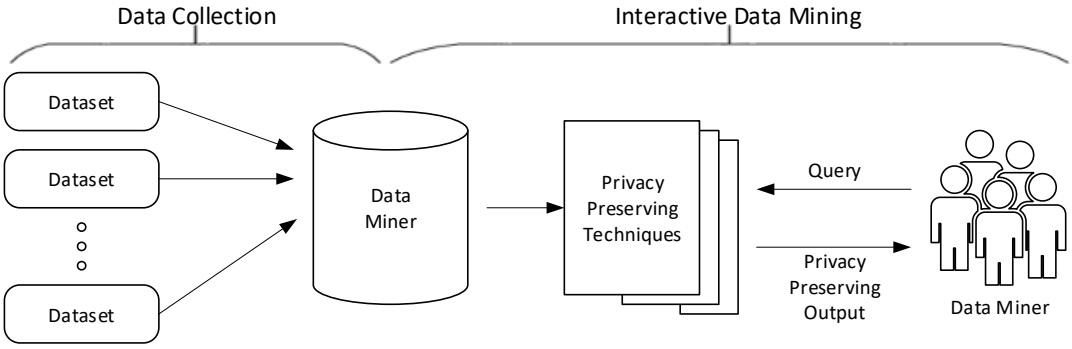

Figure 3.2: Interactive Privacy-Preserving Data Mining

3.2 Distributed Privacy-Preserving Data Mining (DPPDM)

In DPPDM, multiple parties want to run a protocol to jointly perform a data mining task on their private data such that no party will learn anything new from the protocol besides the final output. Figure 3.3 shows the distributed privacy-preserving data mining concepts. DPPDM can be achieved by two different approaches. In the first approach, multiple parties want to build or generate a privacy-preserving data model. Aa a result, they securely integrate their private datasets by generalizing or by anonymizing the sensitive information and then applying the privacy-preserving techniques over the integrated data to achieve

their goal. Then these data models can be used by the miners to extract different data mining tasks. On the other hand, the second approach is more complex, where each participating party's involved in building the privacy-preserving data model. They use secure multi-party computation protocols to jointly perform the data mining task by applying a privacy-preserving mechanism to achieve their privacy data model. Lindell and Pinkas [35][36] propose a secure protocol to build a decision tree using the ID3 algorithm over data that is distributed between two parties. The protocol is built in such a way that during the execution of protocol parties can not learn anything about the inputs of the other party or vice versa. I.e. there is no information leakage during the protocol, anything they learn is after the output is generated. Du and Zhan [17] propose a classifier to build a decision tree over two-party distributed data. They have used a trusted third-party server using *scalar product protocol* to privately and efficiently execute the protocol. In this protocol, both the participant parties do not trust the third party server completely. Therefore, they use the server to do only the necessary computations to get the result i.e to build a decision tree on their distributed data. Jiang and Clifton [30] propose a secure distributed framework, called D*k*A(*Distributed k-anonymity*) that generates a k-anonymous dataset from a two-party dataset. They achieve data security and privacy using secure multi-party security. Kantarcioglu and Kardes [31] propose a two-party protocol in the malicious model for equality and dot product. They have proposed two algorithms that are secure against malicious parties. First, they have modified the semi-honest model based on zero-knowledge proof, the second protocol is novel design specially for malicious adversaries. Similarly, Vaidya *et al.* [51] propose a privacy-preserving decision tree over vertically partitioned data over multiple parties based on the ID3 algorithm called *PPID3*. Their approach differs from all other protocols discussed above. As in all other protocols, the class attribute is shared by all parties, while in their approach only one party has

the class attribute. This protocol is not completely secure thus they proposed a secure algorithm using *secure multi-party dot product with homomorphic encryption*. Since only one party has the class attribute they use a cardinality set of intersection protocols to execute the protocol completely. Boutet *et al.* [7] propose a mechanism for a privacy-preserving collaborative that leverages recommender systems in a distributed environment. Based on the user interest the recommender system privately recommends items similar to his/her interest with optimum accuracy. They propose a twofold mechanism. First, they apply a technique to obscure the user profiles then they apply strong privacy protected random communication techniques. Emekci *et al.* [22] propose a protocol to build a privacy-preserving decision tree where data is distributed among several parties. In this protocol, during the initial phase of this protocol, the author assumes that all parties are semi-honest. Later on, during the intermediate phase, one or many parties could be malicious and could affect the final output of the protocol, and their computation is strong enough to verify the correctness. The primary goal of our approach in this thesis is to perform a distributed data mining task to achieve a differentially-private output with the security of protocol (e.g., a classifier) as well as the privacy of data (e.g., Private data). In contrast, the papers discussed above achieves either the security of protocol or the privacy of data using different types of privacy-preserving data mining techniques. Though many of the protocols discussed above achieve both the security of protocol and the data privacy, the output they produce is not privacy-preserving. Differential privacy has an upper edge in DPPDM, which produces a privacy-preserving output. Our protocol works in a semi-honest model and provides altogether the security of protocol, data privacy and privacy-preserving output as well.

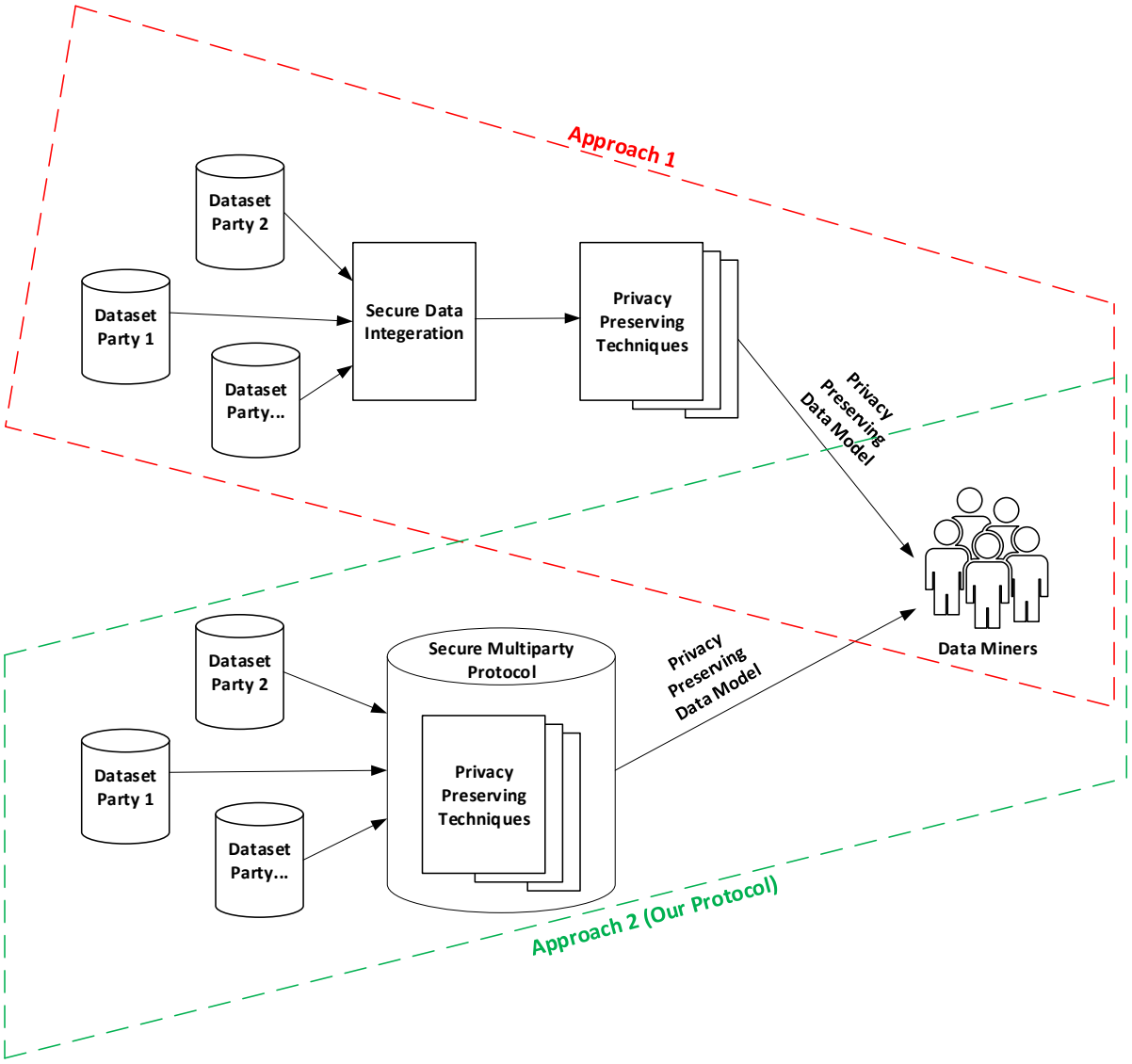

Figure 3.3: Distributed Privacy-Preserving Data Mining

3.3 Privacy-Preserving Data Publishing (PPDP)

In privacy-preserving data publishing, a single party applies a privacy model on the input of raw data in order to release an anonymized and privacy-preserving data. Since it is published after ensuring that the privacy is protected and preserved, it gives researchers

greater flexibility to do better data analysis. Several approaches have been used to achieve PPDP using different techniques and methods. Machanavajjhala *et al.* [37] propose a protocol called ℓ-*Diversity* to safeguard from attacks on privacy while using *k*-anonymity for privacy-preserving in data integration and release. The principle of ℓ-*Diversity* is that the sensitive values in each QI-group are sufficiently diverse. Xiao and Tao [55] propose a new technique called *anatomy*, using a *generalization* method based on *personalized anonymity*. They use ℓ-*Diversity* for strong privacy-preservation.

In this approach, the owners of data want to apply some privacy mechanism to anonymize or randomize the data before publishing for analysis and mining so that the privacy of data is preserved. Differential privacy is one of the strongest methods to achieve privacy. The following protocols use *differential privacy* [41][44][52][56][14] for achieving privacy. Mohammed *et al.* [41][44] proposed an algorithm, called *DiffGen*, that achieves ϵ-differential privacy by generalizing the raw data and then adding noise. The algorithm is based on a non-interactive approach, which means once the data is anonymized and satisfies ϵ-differential privacy, it is published for classification analysis. *k*-anonymity is another popular method to achieve privacy. The following papers used the same technique to achieve privacy in their protocol and to publish heterogeneous health data ϵ-differential private using *DiffGen*. This technique could be used in interactive as well as non-interactive frameworks. Vaidya *et al.* [52] propose a method to differentially-private *Naive Bayes* classification model to release a differentially-private data to be deployed as a PaaS service in the cloud. Clifton and Tassa [11] propose a PPDP model using syntactic anonymity. Based on the criticism of syntactical models the author has justified that syntactical models are still relevant in privacy-preserving data publishing using anonymization. Jun *et al.* [56] propose *PrivBayes*, a differentially-private method to release high dimensional data based on a Bayesian Network model. PrivBayes uses Exponential mechanism to

achieve differential privacy. Day and Li [14] present a protocol called *DPSense* to publish differentially-private statistical information from high dimensional data using sensitivity control. Based on the sensitivity of the dataset the algorithm applies noise addition and uses an exponential mechanism to achieve privacy.

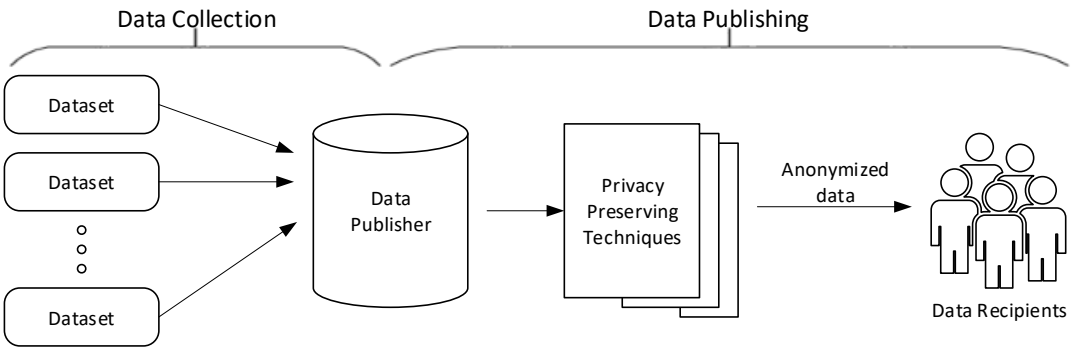

Figure 3.4: Privacy-Preserving Data Publishing

3.4 Distributed Privacy-Preserving Data Publishing (DPPDP)

In distributed privacy-preserving data publishing, multiple parties jointly execute a protocol on their private data in order to output an integrated and privacy-preserving data.

Vaidya and Clifton [50] propose a protocol to publish k-means clustering over vertically distributed data with different attributes distributed among multiple data owners. Mohammed *et al.* [42] propose two protocols to securely integrate private data from multiple users in semi-honest and malicious user models. For semi-honest parties, they achieved privacy using *k*-anonymity by the process of specialization and created a taxonomy tree and then calculated the score to find the goodness of specialization. In malicious parties the algorithm is almost the same as in the semi-honest model except the addition of a

participation strategy to check the malicious user deviation and add a score to a fixed value after each iteration of contribution from participant parties. Fung *et al.* [25] propose a protocol for private data mashup for SOA. Their approach is on the data mash-up based on the service request of the user, and then collecting and integrating required data from multiple data providers in privacy-preserving manner using a generalization approach. For privacy they used *LKC-privacy* [43] for multi-party data mashup. Vatsalan *et at.* [53] propose an algorithm to integrate databases from various parties with data privacy. They propose a technique called *privacy-preserving record linkage(PPRL)*. They built a taxonomy of PPRL techniques and check the performance based on experimental data.

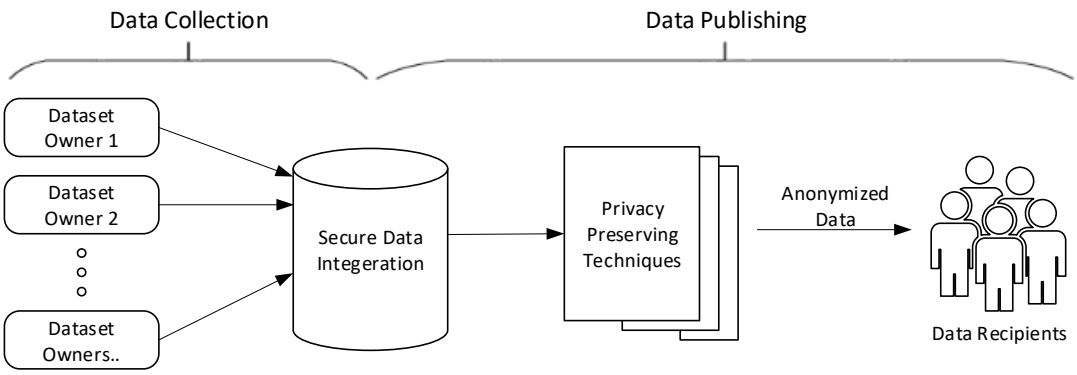

Figure 3.5: Distributed Privacy-Preserving Data Publishing

Mohammed *et al.* [40] propose an algorithm to release differentially-private data for vertically partitioned data called *DistDiffGen*, based on generalization of preprocessed data then adding noise using an exponential mechanism to make ε-differentially-private. The total protocol is distributed in one main protocol and two sub-protocols. The algorithm first generalizes the raw data, then by using the sub-protocol *distributed exponential mechanism* calculates the score of the candidates, and choose the best candidate to specialize and cut by applying exponential mechanism. To count the actual number of data in the leaf node

they used a *secure scalar product protocol* [30] [54] that securely calculates the shares of data from parties. Chen *et al.* [10] propose a non-interactive network data publication using differential privacy to achieve privacy in correlated datasets. Their approach is to first generate a private labeling of vertex to given network datasets and build an adjacency matrix to form clusters, then to re-build a noisy adjacency matrix by using an exponential mechanism. Dagher *et al.* [13] present a protocol on the multiparty data mashup called *Fusion* to come up with privacy-preserving data mashup where data is distributed among multiple parties. Their technique is to first compute the score in a distributed fashion using information gain called *distributed specialization score*, then apply it to another protocol to build a taxonomy tree in a hierarchical manner for data mashup using LKC-privacy [43] protocol.

Table 3.1 summarizes the features of the representative approaches, including our proposed protocol.

Table 3.1: Comparative evaluation of main features in related approaches including our proposed approach

Approach	**Environment**			**Domain**					**Security Model**		**Output Privacy**	
	One	**Two**	**Multi**	**Data Mining**				**Data Publishing**	**Semi-honest**	**Malicious**	**Differential Privacy**	**Others**
				Classification	**Clustering**	**Frequent Pattern**	**Other**					
Zhang *et al.* [57]	●							●			●	
Su *et al.* [49]	●						●				●	
Balu *et al.* [5]	●				●						●	
Blum *et al.* [6]	●				●						●	
McSherry [39]	●						●				●	
Mohan*et al.* [45]	●						●				●	
Kotsogiannis *et al.* [34]	●						●				●	
Jiang and Clifton [30]		●			●				●			●
Vaidya *et al.* [52]	●							●			●	
Boutet *et al.* [7]		●	●			●			●		●	
Xiao and Tao [55]	●							●				●
Machanavajjhala *et al.* [37]	●							●				●
Mohammed *et al.* [41][44]	●							●			●	
Mohammed *et al.* [40]		●						●	●		●	
Vaidya and Clifton [50]			●					●	●			●
Clifton and Tassa [11]	●							●				●
Jun *et al.* [56]	●							●			●	
Day and Li [14]	●							●			●	
Fung *et al.* [25]			●					●	●			●
Vatsalan *et at.* [53]			●					●	●			●
Chen *et al.* [10]			●					●	●		●	
Dagher *et al.* [13]		●	●					●	●		●	
Alhadidi *et al.* [3]		●						●	●		●	
Emekci *et al.* [22]			●					●	●			●
Kantarcioglu and Kardes [31]		●					●			●		●
Agrawal and Srikant [1]	●			●								●
Friedman ans Schuster [23]	●			●							●	
Lindell and Pinkas [35][36]		●		●					●			●
Du and Zhan [17]		●		●					●			●
Vaidya *et al.* [51]		●	●	●					●			●
Mohammed *et al.* [42]			●	●					●	●		●
Our proposed protocol 4.1		●		●					●		●	

CHAPTER 4

S-DPDT PROTOCOL

Let parties $\mathcal{P}_1$ and $\mathcal{P}_2$ be the data owners of dataset $\mathcal{D}_1$ and $\mathcal{D}_2$ respectively. We assume that both parties $\mathcal{P}_1$ and $\mathcal{P}_2$ have the same set of records in their data table D_1 and D_2 respectively. The attribute *UID* and *Class* are common in both the data table D_1 and D_2. The *UID* attribute could be either categorical or numerical but the $class$ attribute should be categorical, and the rest of the attributes in both the datasets are different and private. The purpose of our work is to propose a protocol for building a classifier for generating a decision tree over the distributed datasets in a differentially-private manner. The objective is to maintain the privacy of the private data as well as the security of the protocol, so that there is no leakage of information to each of the participating parties during the execution of the protocol. Also, to generate an output which is differentially-private in such a manner that we can protect it from any kind of linkage attack. A decision tree is important in performing decision making tasks and in finding the class of a new data item without knowing all the attributes of the entity.

In this chapter, we first give a general view of our proposed privacy-preserving protocols approach to build a decision tree in a differentially-private manner. We then present the details of each key step in our protocols. The objective of our proposed solution is to build a classifier from the distributed data of the participating two parties using differential privacy for maintaining strong privacy while maintaining the utility of the output. Our

whole solution includes four protocols:

Protocol 4.1—Two Party Differentially-Private Decision Tree: This is the main protocol that calls and executes other protocols *distributed exponential mechanism*, 2, 3, and 4 to build a differentially-private decision tree from the distributed dataset D_1 and D_2. To achieve differential privacy it uses an exponential mechanism to choose a random winner attribute.

Protocol 4.2—Categorical Split Protocol: This protocol is called by the main protocol when the winner attribute is categorical and is using the exponential mechanism to choose the best random split of the winner node.

Protocol 4.3—Numerical Split Protocol: This Protocol is executed by the main protocol when the winner attribute is numerical and is using the exponential mechanism to choose the best random split point of the winner node.

Protocol 4.4—Score Protocol: This protocol is used to calculate the scores of all sub-datasets of all possible split. This protocol is called and executed by categorical split protocol and numerical split protocol.

4.1 Two Party Differentially-Private Decision Tree

Our main protocol determines the whole solution for building a secure two-party differentially-private decision tree using three other protocols and a distributed exponential mechanism (DEM) protocol. This main protocol, which takes as input the datasets D_1 and D_2 of parties $\mathcal{P}_1$ and $\mathcal{P}_2$, respectively, where both datasets contain different sets of attributes for the same set of records (individuals), except for a common UID and a class attribute. The attributes could be numerical or categorical but the class attribute must be categorical. For example: Party $\mathcal{P}_1$ owns a dataset $D_1(ID, Age, Job, ..., Class)$, while party $\mathcal{P}_2$ owns a

Two Party Differentially-Private Decision Tree Protocol

Input: Datasets $D_1 = \{A_1, \ldots, Aj\}$ and $D_2 = \{A_{j+1}, \ldots, A_n\}$ owned by parties $\mathcal{P}_1$ and $\mathcal{P}_2$ respectively, Class attribute cls owned by both parties, privacy budget B.
Output: Differentially-private decision tree $\mathbb{T}$.

Initial Step: Each Party $\mathcal{P}_i : i \in \{1, 2\}$:

1. Generate an encrypted version of its dataset $[\![D_i]\!]$.
2. For each attribute $A \in D_i$ based on $\Omega(cls)$, generate an encrypted two-column Count table C_A^c using Laplace Mechanism.
3. Send $[\![D_i]\!]$ and all Count tables $[\![C_A^c]\!]$to the other party.

Main Protocol:

1. Each Party $\mathcal{P}_i$ computes information gain IG_A for each attribute $A \in D_i$.
2. Both parties $\mathcal{P}_1$ and $\mathcal{P}_2$ jointly execute the *Distributed Exponential Mechanism* protocol to determine the winner attribute A^w.
3. If $A^w \in D_1$ (i.e. owned by $\mathcal{P}_1$):
   (a) If A^w is *categorical*:
      i. $\mathcal{P}_1$ executes Protocol 4.2 to determine in a differentially-private manner the winner split X_w for attribute A^w.
      ii. Based on $X_w = (X'_w, X''_w)$, $\mathcal{P}_1$ splits D_1 into two partitions $D'_1 = D_1[A^w \in X'_w]$ and $D''_1 = D_1[A^w \in X''_w]$, and similarly splits $[\![D_2]\!]$ into $[\![D'_2]\!]$ and $[\![D''_2]\!]$.
   (b) Otherwise, if A^w is *numerical*:
      i. $\mathcal{P}_1$ executes Protocol 4.3 to determine in a differentially-private manner the split value v for attribute A^w.
      ii. Based on v, $\mathcal{P}_1$ splits D_1 into two partitions $D'_1 = \{\ D_1[A^w \leq v]\ \}$ and $D''_1 = \{\ D_1[A^w > v]\ \}$. Similarly, $\mathcal{P}_1$ splits $[\![D_2]\!]$ into $[\![D'_2]\!]$ and $[\![D''_2]\!]$.
   (c) $\mathcal{P}_1$ adds to $\mathbb{T}$ the attribute A^w as a parent node with two child nodes: left child lc and right child rc such that D'_1 and $[\![D'_2]\!]$ belong to lc, and D''_1 and $[\![D''_2]\!]$ belong to rc.
   (d) $\mathcal{P}_1$ instructs $\mathcal{P}_2$ on how to split $[\![D_1]\!]$ and D_2.
4. Otherwise, if $A^w \in D_2$ (i.e. owned by $\mathcal{P}_2$), then $\mathcal{P}_1$ waits for the instructions from $\mathcal{P}_2$ to split D_1 and $[\![D_2]\!]$.
5. Both parties repeat Steps 1–4 on each child node's data until all attribute are used to split or each leaf node has the same class value.
6. Return $\mathbb{T}$.

Protocol 4.1: Differentially-Private Decision Tree

dataset $D_2(ID, Sex, Salary, MaritalStatus, ..., Class)$. A privacy budget B is set for the whole execution of protocol that is evenly distributed during the making of each level of the decision tree. As an output of the solution our objective is to produce a differentially-private decision tree $\mathbb{T}$.

However, before executing this main protocol both the participating parties have to fulfill some pre-requisites. We call this Initial Steps. Since both parties' main aim is to protect the privacy of their data, they use a strong encryption scheme to share their encrypted private data with each other. In Step 1, both parties $\mathcal{P}_1$ and $\mathcal{P}_2$ encrypt their dataset using distributed ElGamal encryption to ensure that both parties do not infer anything about the data. Since, the class count of each attribute's unique value is required in calculating the information gain (the utility function used for the calculation of score). Therefore, for each attribute's unique value in their dataset and their respective class value, both parties generate an encrypted two-column count table $[\![C_A^c]\!]$: $\langle [\![AttributeValue]\!], [\![ClassCount]\!] \rangle$ added with random Laplace noise. Finally, they share their respective encrypted dataset and encrypted count tables of all the attributes in their dataset with each other through a secure channel. Now, they are ready for the execution for the main protocol.

After performing the initial steps successfully, both parties proceed and simultaneously execute S-DPDT at their respective machines. The general idea of the whole protocol is that the owner of the winner attribute will always proceed with the protocol and the other party will wait for the instruction at each iteration of the protocol. Therefore, to determine the winner attribute, the first step is to calculate the information gain of each attribute of their respective dataset at their own end. Now, to determine the winner attribute randomly both parties jointly execute the *distributed exponential mechanism protocol* 2.2.5 which takes as an input $\langle Attribute, Score \rangle$ pairs for each attribute owned by each of the parties and part of the privacy budget B. Here $score$ refers to the information gain of attributes.

After getting the winner attribute, the winning party will proceed with the protocol and execute the next steps.

In Step 3, the protocol is further executed by party $\mathcal{P}_1$ (since the winner attribute A^w belongs to $\mathcal{P}_1$), execution will remain the same if the winner attribute belongs to $\mathcal{P}_2$. Since, the further step of execution is to be applied on the winner attribute, $\mathcal{P}_1$ can directly determine the attribute property (categorical or numerical), since he is the owner of the dataset. If $\mathcal{P}_1$ finds that the winner attribute is categorical, he will execute Protocol 4.2 (explained in 4.2) to determine the X_w (split) of the winner attribute in a differentially-private manner. At this point, after getting the split party $\mathcal{P}_1$ will first split its own dataset D_1 into two child partitions D_1' and D_1'' based on the X_W',X_W'' (split combination). Since the set of records in the dataset in D_1 and D_2 is of the same individual, $\mathcal{P}_1$ will also split the encrypted dataset $[\![D_2]\!]$ into two child partitions $[\![D_2']\!]$ and $[\![D_2'']\!]$ based on its own's data split.

Otherwise, if the $\mathcal{P}_1$ finds that the winner attribute A^w is numerical, he will apply a different protocol, called a Numerical split protocol 4.3 (explained in 4.3) to determine the split value v of the winner attribute in a differentially-private manner. Based on the split value v the party $\mathcal{P}_1$ determines to split its own dataset into two child partitions. For any value of the winner attribute which is less than or equal to the split value v is assigned to the partition D_1' and any value which is greater than the split value v is assigned to the other partition D_1''. After getting the two partitions of its own dataset, party $\mathcal{P}_1$ will similarly split party $\mathcal{P}_2$ encrypted dataset $[\![D_2]\!]$ into $[\![D_2']\!]$ and $[\![D''2]\!]$.

After getting the parent node (winner attribute) and its splits (winner split), party $\mathcal{P}_1$ will start building the differentially-private decision tree $\mathbb{T}$. The first node, the winner attribute A^w, is assigned as a root node and it's split into child nodes: left child and right child with dataset $\langle D_1', [\![D_2']\!] \rangle$ and $\langle D_1'', [\![D_2'']\!] \rangle$ to left child node and right child node respectively. Straight away after creating the first level of the decision tree, the winning party will instruct

the other party to split the dataset $[\![D_1]\!]$ and D_2 similarly as the left child and right child datasets.

On the other hand, if after executing DEM at Step 2 above, the winner attribute A^w belongs to party $\mathcal{P}_2$, the whole solution will be executed as explained above, except that the party $\mathcal{P}_2$ will proceed with each step and $\mathcal{P}_1$ will have to wait for the instruction from $\mathcal{P}_2$ to split the dataset D_1 and $[\![D_2]\!]$. Since this is a recursive process, we will not get the final decision tree until we use all attributes from both datasets D_1 and D_2 or we get homogeneous class values at each leaf node. Hence, in Step 5, both parties will again start the main protocol from Step 1. They will compute information gain for the remaining attributes and again apply DEM to determine the winner attribute. Whoever owns the winner attribute will proceed with the protocol and perform Step 3, and the other party will wait for the instruction. After all attributes are used, split, and added to $\mathbb{T}$ after each iteration of the protocol to build each level of the decision tree, it grows from top to bottom. We get a differentially-private decision tree $\mathbb{T}$ after the whole execution and iterations have run smoothly. Forthwith, we can use $\mathbb{T}$ classifier as a data model to determine the $Class$ attribute of any new row of data inserted in the dataset without knowing all the attributes.

4.2 Categorical Split Protocol

This protocol is executed to determine the best split for a categorical attribute. The split is determined by finding all the possible binary combinations of unique values of the winner attribute, then choosing the best split in a differentially-private manner. This protocol takes as input, the winner party $\mathcal{P}_i$, the winner attribute A^w, and the datasets D_1 and $[\![D_2]\!]$. The execution of this protocol is required when the winner party $\mathcal{P}_i$ owns the winner attributes A^w and finds that the A^w is categorical, this protocol is executed at

Categorical Split Protocol

Input: Winner Party $\mathcal{P}_i$, winner attribute A^w, Data table D_i, Data table $[\![D_j]\!]$
Output: Winner split $X_w = (X'_w, X''_w)$

1. Winner Party $\mathcal{P}_i$ constructs the set X of all possible split combination pairs based on $\Omega(A^w)$: $X = \{X_k = (X'_k, X''_k) : 1 \leq k \leq |X|\}$.
2. For each $X_k \in X$:
   (a) $\mathcal{P}_i$ splits D_i into two partitions $D'_i = D_i[A^w \in X'_k]$ and $D''_i = D_i[A^w \in X''_k]$, and similarly splits $[\![D_j]\!]$ into $[\![D'_j]\!]$ and $[\![D''_j]\!]$.
   (b) Both parties jointly execute Protocol 4.4 to compute the *noisy* score of X_k, denoted by $Score(X_k)$.
3. The winner split X_w is the one that has the highest score among all possible splits:
$$X_w = (X'_w, X''_w) \in X : Score(X_w) = \max_{1 \leq k \leq |X|} Score(X_k)$$
4. Return X_w.

Protocol 4.2: Categorical Split Protocol

Step 3, in the main protocol 4.1. Determining the split of the categorical attribute is more complex than the numerical attribute and takes most of the execution run-time of the whole solution. Therefore, to determine the split, party $\mathcal{P}_i$ first determines the domain of the winner attribute, then based on its domain size it generates pairs of combination sets, such that each pair will only have a unique item. Thus, if an attribute domain size is n, the number of unique combination pairs will be $2^{n-1} - 1$. For example, if the domain size of an attribute is 4, then the number of the combination will be $2^{4-1} - 1 = 7$, explained with an example below. Consequently, if the number of unique values grows in the domain set of an attribute the combination pairs will grow exponentially. This makes determining the categorical split very complex for attributes having a large domain size.

Example 4.2.1. Let us suppose the winner attribute is *Job* and it belongs to *Party* $\mathcal{P}_1$. $\mathcal{P}_1$

will go ahead with the protocol and execute the rest of the steps to determine the winner split for the winner attribute.

Since the attribute *Job* is a categorical attribute, to determine the best split, we first need to find all possible binary combination pairs. The attribute Job has 4 distinct values *(Professor, Student, Assistant, and Engineer)* and the possible number of combination pairs for the binary split are as follows:

$Combination\ Pair\ 1$: [$\langle Student \rangle$, $\langle Professor, Assistant, Engineer \rangle$]

$Combination\ Pair\ 2$: [$\langle Assistant \rangle$, $\langle Professor, Student, Engineer \rangle$]

$Combination\ Pair\ 3$: [$\langle Professor \rangle$, $\langle Student, Assistant, Engineer \rangle$]

$Combination\ Pair\ 4$: [$\langle Engineer \rangle$, $\langle Professor, Student, Assistant \rangle$]

$Combination\ Pair\ 5$: [$\langle Professor, Student \rangle$, $\langle Assistant, Engineer \rangle$]

$Combination\ Pair\ 6$: [$\langle Professor, Assistant \rangle$, $\langle Student, Engineer \rangle$]

$Combination\ Pair\ 7$: [$\langle Professor, Engineer \rangle$, $\langle Student, Assistant \rangle$] □

At Step 1, after getting the split combination pairs for the attribute A^w, as explained in the example above, and based on each split combination pair, $\mathcal{P}_1$ splits its own dataset D_1 into two partitions and similarly splits the other party's encrypted dataset into two partitions, respectively. At this point, choosing the best split combination from all the split combinations, a noisy score is calculated using the score protocol 4.4 for each combination pair termed as $Score(X_k)$, where k is each split combination. In this Step, both parties have to collaborate to execute the score protocol and to calculate the score. At the end of the protocol, after the score of each combination pair is calculated, the combination pair having the highest noisy score is chosen as the winner split X_w for the winner attributes A^w and the main protocol proceeds to further steps.

Numerical Split Protocol

Input: Winner Party $\mathcal{P}_i$, winner attribute A^w, Data table D_i, Data table $[\![D_j]\!]$
Output: Split value v

1. Winner Party $\mathcal{P}_i$ constructs the set X of all possible split combination pairs based on the distinct values in $\Omega(A^w)$:
   $X = \{X_k = (X'_k, X''_k) : k \in \Omega(A^w)\}$, where $X'_k = \{\Omega(A^w)[\leq k]\ \}$ and $X''_k = \{\Omega(A^w)[> k]\ \}$.
2. For each $X_k \in X$:
   (a) $\mathcal{P}_i$ splits D_i into two partitions $D'_i = D_i[A^w \in X'_k]$ and $D''_i = D_i[A^w \in X''_k]$, and similarly splits $[\![D_j]\!]$ into $[\![D'_j]\!]$ and $[\![D''_j]\!]$.
   (b) Both parties jointly execute Protocol 4.4 to compute $NoisyScore(X_k)$.
   (c) $\mathcal{P}_i$ determines the final noise score of X_k: $Score(X_k) = Score(X_k).|I_k|$, where $|I_k|$ denotes the size of the interval of element $k \in \Omega(A^w)$.
3. Select the X_w that has the highest score:
   $$X_w = (X'_w, X''_w) \in X : Score(X_w) = \max_{1 \leq k \leq |X|} Score(X_k)$$
4. The split value v is uniformly chosen from the range $[x, y)$, where $x = Max(X'_w)$ and $y = Min(X''_w)$.
5. Return v.

Protocol 4.3: Numerical Split Protocol

4.3 Numerical Split Protocol

Otherwise, if the winner attribute obtained by party $\mathcal{P}_1$ is numerical, this protocol is called and executed by the main protocol to determine the split value v for the winner attribute to split the dataset into two partitions. Since the operation on numeric value is easier and more straight forward than it is for categorical data, all we need is a value within that column chosen as the split point and any value which is less than or equal to that split point is added to partition 1 and any value which is greater than this split point is added to partition 2. In the first Step, party $\mathcal{P}_i$ is doing the same thing based on each distinct value

K_i in the winner attribute A^w. Therefore, each distinct value is chosen as the split value, and a combination pair $\langle X'_k, X''_k \rangle$ is determined by taking any value which is $<= K$ in the partitioned subset and added to X'_k, and any value which $> K$ is partitioned and added to X''_k. Based on each split combination pair for each distinct value, $\mathcal{P}_i$ splits its own dataset D_i into two partitions D'_i and D''_i and also splits the other party data into $[\![D'_j]\!]$ and $[\![D''_j]\!]$. To calculate the noisy score for each split, based on the distinct value of both parties, they have to collaborate together and jointly execute the score protocol 4.4. After getting the noisy score, the winner party $\mathcal{P}_i$ multiplies each distinct value score by the number of elements having the same frequency count in the winner attribute A^w column and determines it as the final score of each combination pair. The distinct element whose frequency is highest in the dataset will probably have the highest score. In Step 3, the protocol $\mathcal{P}_i$ will choose the split X^w having the highest score among all the scores as the winner score and the winner split value v is uniformly chosen from the range of maximum and minimum scores of the bucket set. After getting the split value v, $\mathcal{P}_i$ proceeds to the next step of the main protocol.

4.4 Score Protocol

This is the most important protocol, the whole solution of S-DPDT depends on its smooth and accurate execution. Both categorical and numerical attributes use this protocol interactively to get their splits value based on the score calculated for each of the split combination pairs. As an input, this protocol takes the winner party $\mathcal{P}_i$ and the winner attribute A^w. Irrespective of attribute property (Categorical or numeric) this protocol computes the score efficiently. In Step 1, the winner party $\mathcal{P}_i$ takes the combination pair sub-dataset D'_i and D''_i and calculates *information gain* of each combination pair sub-dataset of its own winner attribute. The score of each attribute in each of the sub-datasets D'_i and D''_i is added together

Score Protocol

Input: Winner Party $\mathcal{P}_i$, winner attribute A^w
Input: Datasets: D'_i, D''_i, $[\![D'_j]\!]$, $[\![D''_j]\!]$, $[\![D_j]\!]$
Output: $NoisyScore(X_k)$

1. $\mathcal{P}_i$ computes *Information Gain* (IG) for all attributes in D'_i and D''_i to determine the total score of its attributes:
$$Score_i := \sum_{A \in D'_i} IG(A) + \sum_{A \in D''_i} IG(A)$$
2. For each attribute $A \in [\![D_j]\!]$:
   (a) Both parties jointly define random permutation π such that no single participant knows the permutation.
   (b) For each class value $c \in \Omega(cls)$:
      i. $\mathcal{P}_j$ sends a two-column count table $\mathbb{T}_{A,c} = \{in, out\}$ encrypted using Exponential Elgamal, where each row contains an encrypted value from $\Omega(A)$ in the in column, and an encrypted *Laplace* noise for that value in the out column.
      ii. For each encrypted value $[\![v]\!] \in A$ in $[\![D'_j]\!]$ such that $cls = c$:
         A. Both parties jointly perform *plaintext equality test* PET on column in to determine a match. When a match is found, $\mathcal{P}_i$ increments by 1 the corresponding noise $[\![v.noise]\!]$ in the out column by homomorphically multiplying the noise with Elgamal ciphertext of 1:
$$[\![v.noise + 1]\!] := [\![v.noise]\!] \times [\![1]\!]$$
         B. Both parties jointly apply the Verifiable Mix Network protocol [29] on $\mathbb{T}_{A,c}$ that uses the random permutation π to generate shuffled and randomized table $\mathbb{T}^{\pi}_{A,c}$.
      iii. Repeat Step 2(b)ii for each encrypted value $[\![v]\!] \in A$ in $[\![D''_j]\!]$ such that $cls = c$.
   (c) Party $\mathcal{P}_j$ uses its private key share to partially decrypt the noise count column in each table $\mathbb{T}^{\pi}_{A,c}$ generated in Step 2b, and send the results to $\mathcal{P}_i$.
   (d) $\mathcal{P}_i$ uses its private key share to fully decrypt the ciphertexts received from $\mathcal{P}_j$ in Step 2c.
   (e) $\mathcal{P}_i$ computes information gain of A in dataset $[\![D'_j]\!]$, i.e., $IG(A \in [\![D'_j]\!])$, and information gain of A in dataset $[\![D''_j]\!]$, i.e., $IG(A \in [\![D''_j]\!])$.
3. $\mathcal{P}_i$ computes *Information Gain* (IG) for all attributes in $[\![D'_j]\!]$ and $[\![D''_j]\!]$ to determine the total score of the attributes owned by $\mathcal{P}_j$:
$$Score_j := \sum_{A \in [\![D'_j]\!]} IG(A) + \sum_{A \in [\![D''_j]\!]} IG(A)$$
4. Return the noisy score $NoisyScore(X_k)$ of split combination X_k : $NoisyScore(X_k) = Score_i + Score_j$.

Protocol 4.4: Score Protocol

and computed as the total score of its owned attributes as $Score_i$.

The real challenge is to compute information gain for the attributes owned by the other party $\mathcal{P}_j$. Both parties join together to execute Step 2, First, they jointly establish a random permutation for each of the attributes owned by the party $\mathcal{P}_j$ in such a way that it remains unknown for both parties. For each class value c of class attribute cls we used *adult dataset*, where class attribute cls has two values $\langle <= 50K, > 50K \rangle$. The other party $\mathcal{P}_j$ sends an encrypted class count table having two columns, column in for class value and column out total number of class count for particular attribute elements. For calculating information gain all we need is the class count of each attribute's unique element. Party $\mathcal{P}_j$ also adds random noise to the true count in the out column before encryption, to maintain the privacy of the count. Further, both parties jointly perform *plaintext equality test* (PET) 2.3.6 over in column of class count table to find a match. For each class count of each attribute, a unique value of the sub-dataset D'_j they perform PET and whenever a match is found the out column value is incremented by 1 using a multiplicative homomorphic encryption scheme. Party $\mathcal{P}_i$ encrypts number *1* using ElGamal encryption and multiplies its ciphertext with the ciphertext in the respective out column. To protect from predictable ciphertext types of attack and information leakage both parties jointly re-randomized and shuffle the class count using the random permutation computed in the beginning of Step 2, by applying a verifiable mix network protocol 2.3.7. Once all the attributes of sub-dataset $[\![D'_i]\!]$ are executed and the class count is updated with all the increments, the steps are repeated for the sub-dataset $[\![D''_j]\!]$) to compute the updated class count with all attributes and increments. At this point, we need the noisy class count of each attribute element to compute the information gain. In order to do so, we need to decrypt the class count table which can only be performed when both parties decrypt them using their share. In Step 2(c), the other party $\mathcal{P}_j$ applies its private key share to partially decrypt the noisy count from the

class count for each attribute. Party $\mathcal{P}_j$ then shares each partially decrypted count of each attribute to party $\mathcal{P}_i$ using the secure channel communication.

At this point, on each partially decrypted count column for each attributes in $[\![D'_i]\!]$ and $[\![D''_j]\!]$), party $\mathcal{P}_i$ received from $\mathcal{P}_j$, applies his share of the private key to fully decrypt it. After the decryption of each count column, its time to compute the information gain for party $\mathcal{P}_j$ attributes. Therefore, the information gain of each attribute belongs to the sub-datasets ($[\![D'_i]\!]$ and $[\![D''_j]\!]$) is added together respectively and computed as $Score_j$. In the last Step, the protocol returns the noisy score to the categorical split or the numerical split based on whoever calls it by adding the score for party $\mathcal{P}_i$'s $Score_i$ with the $Score_j$ of $\mathcal{P}_j$'s to be called as $NoisyScore(X_k)$ for each attribute's split combination.

CHAPTER 5

PROTOCOL ANALYSIS

In this chapter, we perform the privacy and complexity analysis of our proposed approach.

5.1 Privacy Analysis

The privacy analysis of our proposed protocol finds that the classifier we built is ϵ-differential private and based on the composition properties of differential privacy. The composition property states that any sequence of computations that provides differential privacy in segregation will also result in differential privacy even when computed in any sequence. This is called the *sequential composition*. This is not only true with individual computation, but also in successive computation when current computation depends upon the results of preceding computations.

Lemma 1. *Sequential composition [39]. Let algorithm M_i each provides B_i-differential privacy. The sequence of $M_i(D)$ on the dataset D provides ($\sum_i B_i$)-differential privacy.* □

In the case of sequential composition the privacy cost of each computation of sequence is added together to compute overall privacy cost. Nevertheless, if the sequence of computation is applied on the disjointed subsets of the dataset, the privacy cost depends on the worst guarantees of each computation, not the sum of the privacy cost. This is called the *parallel composition*

Lemma 2. *Parallel composition [39]. Let algorithm M_i each provides B_i-differential privacy. Let D'_i, be the disjoint subsets of dataset D. The sequence $M_i(D \cap D'_i)$ is B_i-differential privacy.* □

Sequential composition is essential to compute the privacy cost of any approach, however, parallel composition is required to get the best of the cost from the sequence of computation privacy cost.

Proposition 5.1.1. *Given a privacy budget B, our approach outputs B-differentially-private decision tree.*

Proof: In our solution the main protocol produces the output of a differentially-private decision tree using three other protocols. It works in two phases: determining the winner attribute and then determining the winner split for that winning attribute. In our protocol we have a fan-out f = 2. According to lemma 2, we can use the same privacy budget in determining each level of the decision tree. Consequently, if we can prove that the summation of the budget at each level is less than or equal to the initial privacy budget B, we can establish that our approach satisfies B-differential privacy.

In the decision tree generation, the privacy budget distributed at each level is $B_l = B/n$, where n is the total number of attributes in the datasets D_1 and D_2, n will also be the height of the decision tree $\mathbb{T}$. The half of B_l at each level is needed to find the winner attribute A^w by applying distributed exponential mechanism (DEM) is $\sqrt{B_l}$. And remaining budget $B_l - \sqrt{B_l}$ is needed to determine the winner split X_w for the winner attribute A^w .

Thus, the use of an exponential mechanism guarantees that the selection of the winner attribute and the winner split at each level of the tree requires B_l, a part of the privacy budget and satisfies B-differential privacy. Therefore, the total privacy budget consumed is:

$$\sum_{l=1}^{n} \underbrace{\sqrt{B_l}}_{\text{winnerAttribute}} + \underbrace{(B_l - \sqrt{B_l})}_{\text{WinnerSplit}} = n \times B_l \leq B$$

Hence, our protocol satisfies B-differential privacy. □

Proposition 5.1.2. *Privacy-preserving. The overall approach is privacy-preserving, such that the private data is protected throughout the entire execution of protocol.*

Proof: (Sketch) To prove that out protocol is privacy-preserving, we will show that the private data is always protected.

Input Data: Both parties $\mathcal{P}_1$ and $\mathcal{P}_2$, encrypts their data, proves knowledge of it, sends it to each other and then inputs to the protocol. Both parties $\mathcal{P}_i$ prove that they know the implicit plaintext of the encrypted dataset without leaking any information to the other party about the plaintexts.

Encrypted Data: On the other hand, the encrypted data is protected due to a strong encryption scheme (DDH for ElGamal). The adversaries can in no way decrypt the dataset or any part of it, as it can only be decrypted using the shared key (*n,n*) between the two parties.

Decrypted Data: The dataset of both parties remains encrypted throughout the protocol, except when they apply a plaintext equality test (PET) on the count column. Both parties jointly apply distributed exponential ElGamal to find a match, which returns 1 for match, it otherwise a garbled output when PET is not successful. A verifiable mix network is used to shuffle and re-randomize the output list after every match is found. This security makes it infeasible for adversaries to find which output belongs to which input. □

5.2 Integrity Analysis

Proposition 5.2.1. *Integrity. The overall protocol is robust against the malicious adversarial under a semi-honest model.*

Proof (Sketch) All the steps of the protocol are publicly verifiable, which prevents the malicious participant party from veering away from the protocol with intent without getting caught. The party, who is honest with the protocol, can withdraw in the middle of the execution as soon as it detects any ill behavior of the other party. Withdrawal of one party will obstruct the completion of the protocol (like the distributed decryption, DEM operations need until the last step of the protocol both parties participation). We derive integrity against a malicious participant from our cryptographic primitives mentioned in Chapter 2. Table 5.1 shows publicly verifiable techniques used at different security-sensitive steps of our proposed protocols. We certify that all data inputs to the protocols are accurately devised. In the initial phase, both the participating parties generate a public key, the distributed key generation (DKG) scheme that guarantees that the output will be uniformly distributed at random [26]. The ciphertext generation must be within $\langle \mathbb{G}_q \text{ x } \mathbb{G}_q \rangle$ for encryption of each dataset in such a way that data owner can check the independency of the ciphertexts. Each of the participating parties inputs a random exponent from the $\mathbb{Z}_q$ whenever they execute operations like mixing(shuffling and re-randomization), plain-text equality testing.

□

5.3 Complexity Analysis

Proposition 5.3.1. ***(Complexity).*** *The overall complexity of our proposed approach is* $\mathcal{O}(n(d + n \log C + nK \log C))$.

Table 5.1: The Primitive verifiable primitives used at different security-sensitive steps of our proposed protocols

P.V.Primitive	**Intial Step**	**Main Protocol**	**Categorical Split Protocol**	**Numerical Split Protocol**	**Score Protocol**
Public Encryption	1				2.b.i
Distributed Decryption					2.c
Mix Network					2.a.i, 2.b.ii.B
Mix and Match					2.b.ii.A
Homomorphic Operation					2.b.ii.A
Cleartext Operation		1, 3.a, 3.b, 6	1, 2.a, 3	1, 2.a, 3, 4	1, 4

Proof. We determine the run-time complexity of our proposed approach based on the Initial step (encryption of dataset, encrypted count table) and main protocol.

Initial Step (Encryption phase): In this phase, both participating parties encrypt their dataset and encrypt a count table. To encrypt dataset D with d records and t columns requires $\mathcal{O}(t \times d)$. The encryption of a class count table for d records also requires $\mathcal{O}(t \times d)$. Since $n > t$, where n is the total number of attributes, then the initial phase runtime complexity is $\mathcal{O}(n \times d)$.

Decision tree generation phase: In the decision tree generation phase, the runtime complexity is the cost of generating each winner node using a distributed exponential mechanism, and determining the winner split of each winner node.

Distributed Exponential Mechanism: Both parties jointly execute *DEM* to determine the winner attribute. Since this is performed n times i.e. height of the tree, which is the total number of attributes in D_1 and D_2, then according to [40], the complexity of encryption and communication is $\mathcal{O}(n^2 \log C)$ and $\mathcal{O}(n^2 K \log C)$, respectively.

Determining Split of winner attribute: The total number of splits is 2^l. The total number of the attributes in dataset D is n, and the number of categorical and numerical is $n/2$ and $n/2$ respectively. For categorical splits, the number of operations depends upon the total number of combination pairs of all categorical attributes. The sum of unique values in each attribute is 64. The class attribute has 2 unique values. So the number of operations for categorical split is 64 X 2 = 128. Therefore, runtime complexity is 128 X $(\mathcal{O}(n/2))$ = $\mathcal{O}(n)$.

For the Numerical attribute, in the worst case scenario (where each value could be unique in numerical the attribute column), the maximum number of unique values for a numerical attribute is size of d. The class attribute has 2 unique values. So the number of operation is 2 X $(\mathcal{O}(d \times n/2))$ = $\mathcal{O}(n \times d)$. Therefore, total runtime complexity of determining the Split of winner attribute is : $\mathcal{O}(n + n \times d$ = $\mathcal{O}(n \times d)$

Thus, the total time complexity of S-DPDT is $\mathcal{O}(n \times d + n \times d + n^2 \log C + n^2 K \log C)$ = $\mathcal{O}(n \times d + n^2 \log C + n^2 K \log C)$ = $\mathcal{O}(n(d + n \log C + nK \log C))$

□

CHAPTER 6

EXPERIMENTAL EVALUATION

In this section, we evaluate the performance of our Secure Two-Party Protocol for Decision Tree classification via Differential Privacy. First, we discuss the implementation details. Second we discuss the results of the experiment that include the utility of the protocols, record scalability and efficiency.

6.1 Dataset

We use a real-life data set called *Adult* [4] data set in our experiment to illustrate the evolution and performance of S-DPDT. The Adult data set consists of 45,222 census records, containing eight *categorical* attributes, six *numerical* attributes, and a *class* attribute. Table 6.1 illustrates the attributes distribution between the two parties in distributed settings.

6.2 Implementation and Setup

S-DPDT is implemented using SCAPI[1], an open source Java based library for implementing secure multiparty computation protocols. We use socket-based communication channels of SCAPI that is based on Socket and ServerSocket of Java.net package. Socket-based communication uses two channels to send and receive ciphertexts between the two parties. We use Group-ElGamal of the ElGamal encryption scheme to encrypt the parties' data.

[1]SCAPI: https://.readthedocs.org/

The experiments were conducted on a machine equipped with Intel(R) Xeon(R) 3.5GHz CPU with 8GB RAM, running on 64-bit Windows 10 Enterprise edition using Java Eclipse IDE.

Table 6.1: *Adult* Dataset, and the distribution of attributes between two parties

Attribute	**Type**	**Owner**
Age	Numerical	*P1*
Workclass	Categorical	*P1*
Fnlwgt	Numerical	*P2*
Education	Categorical	*P1*
Education-Num	Numerical	*P1*
Marital-status	Categorical	*P2*
Occupation	Categorical	*P2*
Relationship	Categorical	*P1*
Race	Categorical	*P2*
Sex	Categorical	*P2*
Capital-Gain	Numerical	*P2*
Capital-Loss	Numerical	*P1*
Hours-per-week	Numerical	*P1*
Native-Country	Categorical	*P2*

6.3 Utility

To determine the utility of our classifier we did a classification analysis, where we compared our classifier accuracy with a Non-DP classifier taken as a baseline. We compared classification in terms of accuracy achieved with respect to a different number of attributes with three different ϵ (epsilon) values.

We use 2/3 of the records of adult data to train the classifier and 1/3 of the records for testing the classifier.

Figure 6.1 depicts the classification accuracy with respect to a different number of attributes in the dataset with three different ϵ (epsilon) values for each type of dataset.

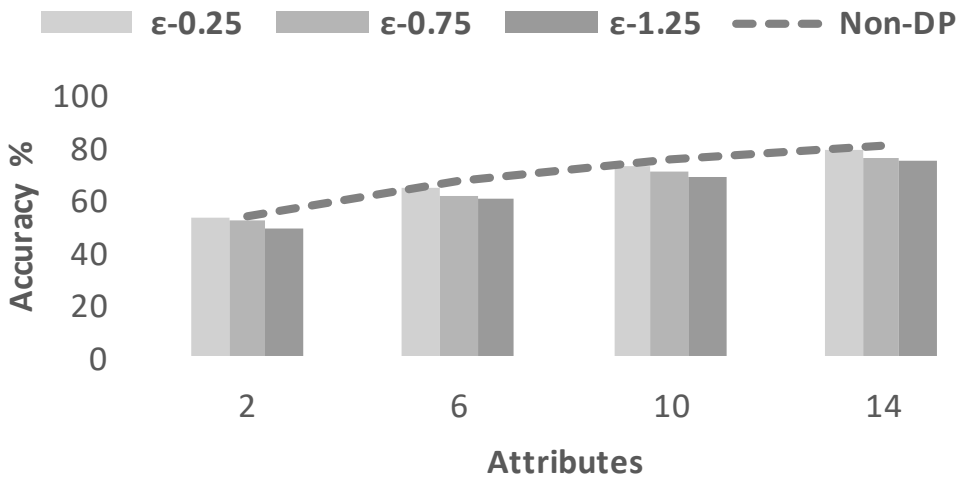

Figure 6.1: Utility of Classification with respect to number of attributes and different epsilon values

The classification accuracy is measured w.r.t to the number of attributes in the dataset, which are set from 2 to 14 and increased linearly. We use three different ϵ values for each number of attributes in the data set, which varies from 0.25 to 1.25. We also increased this value linearly. In our approach, we observe that the accuracy increases linearly from 54% to 74% when the number of attributes increases in the data set. Alternately, when we increase the ϵ values the accuracy decreases slightly in all cases. We set a baseline classifier without differential privacy for comparison of classification utility with our protocol and observe that the accuracy achieved by our protocol was marginally less than the accuracy achieved by baseline without differential privacy. This is expected when we apply differential privacy as accuracy decreases marginally but this doesn't affect the utility of our protocol. However, we also observed that the accuracy of our protocol is consistent w.r.t the baseline, the accuracy increases linearly when the number of attributes is increased.

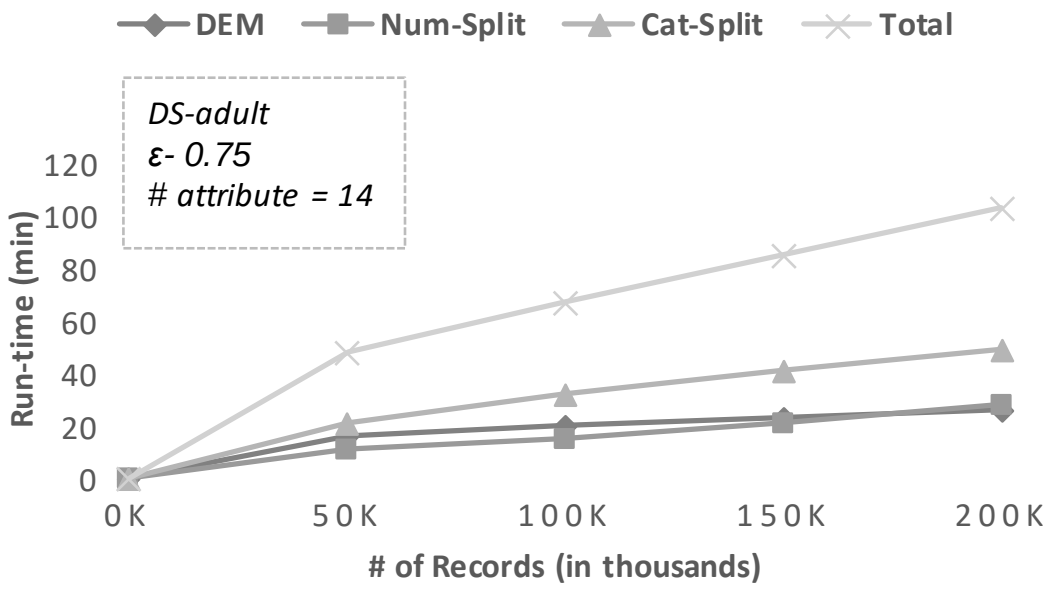

Figure 6.2: Scalability with respect to number of records

6.4 Scalability

We measured the scalability of S-DPDT with respect to the number of records. We observed the runtime of our three main protocols DEM (distributed exponential mechanism), Numerical-split and Categorical-Split with ϵ value set at 0.75. We chose 0.75, as it is the median value of our epsilon range from 0.25 to 1.25 used in testing the experiment.

We observed that to run an adult dataset with 50k records with all 14 attributes it takes up to 48 min to run S-DPDT. The runtime increases linearly as the record size increases linearly from 50k to 200k. Figure 6.2 depicts the runtime in minutes for the number of records from 50,000 to 200,000 in each dataset. We also observed that the most dominant protocol of our experiment is Categorical-split protocol which took the majority of run time, approximately 46% of total time taken by the three main protocols, because the categorical-

split has to find all the possible combinations of domain of each attribute, calculate a score for each combination, and then chose the best score by applying an exponential mechanism on the set of scores.

We observed that there is a linear increase in total runtime w.r.t to a linear increase in the number of records. However, we also observed that the protocol DEM (Distributed Exponential Mechanism) runtime didn't grow linearly w.r.t a linear increase in records, rather it tends to remain constant. This is because the DEM is applied to the score generated by the numerical-split or categorical-split so no matter what the number of records grows to it doesn't affect the runtime of the DEM protocol.

6.5 Efficiency

We tested the efficiency of our proposed approach with Differential Privacy overhead, with respect to the number of attributes in the data set, comparing the runtime to a protocol without differential privacy. The Adult dataset was chosen for testing the efficiency, the ϵ value is set at 0.75. The Figure 6.3 illustrates the runtime for different datasets with a different number of attributes. Here the X-axis represents the number of attributes in the dataset, while the Y-axis shows the time taken in minutes to execute each of the datasets with differential privacy and without differential privacy.

We observed that the runtime increases considerably with an increase in the number of attributes in the data set. Our approach with differential privacy takes more execution time with respect to an approach without differential privacy which is consistent with our expectations. Our approach uses an exponential mechanism to achieve differential privacy which is bound to take more time to randomly pick the best score for the split. We termed it as differential privacy overhead, and to achieve foolproof privacy like differential privacy

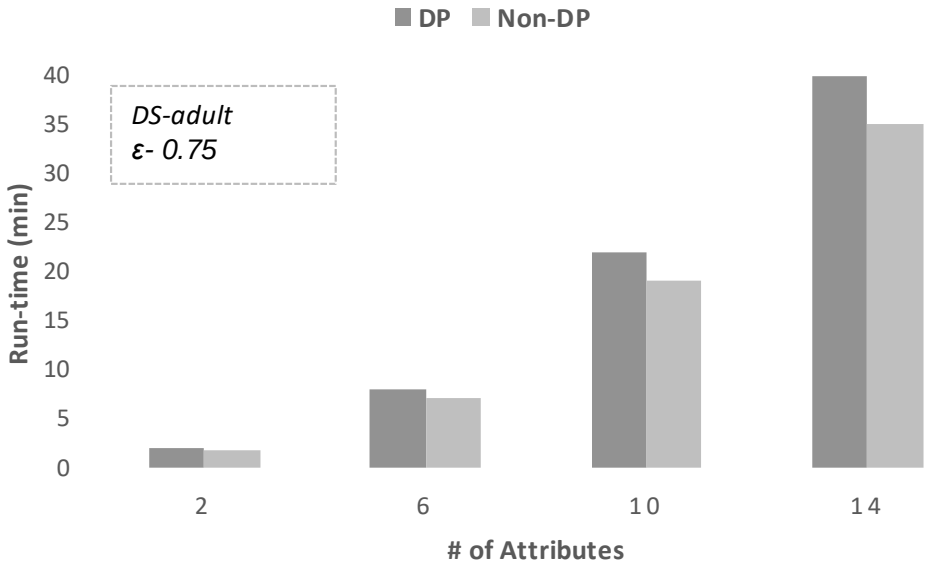

Figure 6.3: Comparative efficiency evaluation with respect to the number of attributes in DP and Non-DP privacy mechanism

the increase in the runtime of our approach is considered irrelevant.

CHAPTER 7

CONCLUSION AND FUTURE WORK

7.1 Summary

In this thesis, we performed extensive research on distributed privacy-preserving data mining to build a classifier for better decision making that maintains the participant party's data securely, achieves good utility, and is scalable and efficient with an increase in datasize. We observed that by using secure multi-party protocol we can perform data mining tasks on distributed data in a differentially-private manner and still achieve high utility. In our experimental performance evaluation of our protocol, we found that 74% utility was achieved with all attributes of the dataset, which is slightly behind the utility achieved by a non-differential privacy mechanism.

In chapter 2, we presented and discussed the building blocks of our thesis that are important to the implementations of our thesis work. We explain the data mining task classification, different techniques to build a decision tree, and finding the best attribute to split i.e Information Gain, Gini Index, Max operator, and Gain Ratio. We also discussed differential privacy and different mechanisms to achieve differential privacy i.e Exponential Mechanism, Laplace Noise addition, and the cryptographic scheme like ElGamal for the encryption of each party dataset.

In chapter 3, we explained the related work that has been done over the years in the field of data privacy-preserving. We did an extensive discussion on privacy-preserving data

mining and privacy-preserving data publishing with respect to various data mining tasks like classification, clustering, pattern mining, etc. with both single party or multi-party environments. We did a comparative evaluation and presented in Table 3.1 the major PPDM and PPDP techniques.

In chapter 4, we propose our solution for building a differentially-private decision tree, explaining the execution of our main protocol and sub-protocols. The proposed solution achieves high utility, is scalable with large datasets and efficient in classification tasks of data mining techniques.

In chapter 5, we conducted the privacy and complexity analysis of our proposed approach. We establish that our approach preserves the privacy of distributed data, is scalable and efficient with large datasets and a large number of attributes in the datasets respectively.

In chapter 6, we analyzed the performance of our approach. We carried out our experiments to establish the utility, scalability, and efficiency of our solution. The experiment results show that our approach achieves high utility, is scalable, and efficient with respect to large datasets and a large number of attributes in the dataset.

In a nutshell, we can say that the main role of this thesis work is to propose an approach to build a classifier- a decision tree over distributed data between two parties using differential privacy which maintains the data security, satisfies the differentially-private output, preserves the privacy, achieves high utility, and is scalable and efficient.

7.2 Future Work

The research work explained and the evaluation of the experiments conducted in this thesis lead to many unanswered questions. In the era of Big Data, working with very large datasets will be quite challenging, as the encryption and distributed decryption takes a good amount

of time and makes the system more complex. It would be intriguing to study how the protocol behaves when the number of attributes in the dataset is quite large and the domain of each attribute is also large. In the case of a binary split of an attribute, finding all possible combinations of the domain of an attribute grows exponentially with the increase in domain size. Furthermore, we also look forward to analyzing how the system will respond in a multi-party set up in a distributed scenario.

REFERENCES

[1] Rakesh Agrawal and Ramakrishnan Srikant. Privacy-preserving data mining. *SIGMOD Rec.*, pages 439–450, 2000.

[2] Mohammad Alaggan, Sébastien Gambs, and Anne-Marie Kermarrec. *BLIP: Non-interactive Differentially-Private Similarity Computation on Bloom filters*, pages 202–216. 2012.

[3] Dima Alhadidi, Noman Mohammed, Benjamin C. M. Fung, and Mourad Debbabi. Secure distributed framework for achieving -differential privacy. In *Proceedings of the 12th International Conference on Privacy Enhancing Technologies*, PETS'12, pages 120–139, 2012.

[4] Kevin Bache and Moshe Lichman. *UCI Machine Learning Repository*. University of California, Irvine, School of Information and Computer Sciences, 2013.

[5] Raghavendran Balu, Teddy Furon, and Sébastien Gambs. *Challenging Differential Privacy:The Case of Non-interactive Mechanisms*, pages 146–164. 2014.

[6] Avrim Blum, Cynthia Dwork, Frank McSherry, and Kobbi Nissim. Practical privacy: the SuLQ framework. In *Proceedings of the ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS)*, pages 128–138, 2005.

[7] Antoine Boutet, Davide Frey, Rachid Guerraoui, Arnaud Jégou, and Anne-Marie Kermarrec. Privacy-preserving distributed collaborative filtering. *Computing*, pages 827–846, 2016.

[8] Felix Brandt. Efficient cryptographic protocol design based on distributed el gamal encryption. In *Proceedings of the International Conference on Information Security and Cryptology*, ICISC'05, pages 32–47, 2006.

[9] Leo Breiman, Jerome Friedman, Richard A Olshen, and Charles J Stone. *Classification and Regression Trees*. Wadsworth and Brooks, Monterey, CA, 1984.

[10] Rui Chen, Benjamin C. Fung, Philip S. Yu, and Bipin C. Desai. Correlated network data publication via differential privacy. *The VLDB Journal*, pages 653–676, 2014.

[11] Chris Clifton and Tamir Tassa. On syntactic anonymity and differential privacy. *Trans. Data Privacy*, pages 161–183, 2013.

[12] Ronald Cramer, Rosario Gennaro, and Berry Schoenmakers. A secure and optimally efficient multi-authority election scheme. In *Proceedings of the annual International Conference on Theory and Application of Cryptographic Techniques (EUROCRYPT)*, pages 103–118, 1997.

[13] Gaby G. Dagher, F. Iqbal, M. Arafati, and Benjamin C. Fung. Fusion: Privacy-preserving distributed protocol for high-dimensional data mashup. In *2015 IEEE International Conference on Parallel and Distributed Systems (ICPADS)*, pages 760–769, 2015.

[14] Wei-Yen Day and Ninghui Li. Differentially private publishing of high-dimensional data using sensitivity control. In *Proceedings of the ACM Symposium on Information, Computer and Communications Security*, ASIA CCS '15, pages 451–462, 2015.

[15] Yvo Desmedt. Society and group oriented cryptography: A new concept. In *Conference on the Theory and Application of Cryptographic Techniques*, pages 120–127. Springer, 1987.

[16] Yvo Desmedt and Yair Frankel. Threshold cryptosystems. In *Conference on the Theory and Application of Cryptology*, pages 307–315. Springer, 1989.

[17] Wenliang Du and Zhijun Zhan. Building decision tree classifier on private data. In *Proceedings of the IEEE International Conference on Privacy, Security and Data Mining - Volume 14*, CRPIT '14, pages 1–8, 2002.

[18] Cynthia Dwork. Differential privacy. In *Proceedings of the International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 1–12, 2006.

[19] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *TCC*, 2006.

[20] Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, 9:211–407, 2014.

[21] Taher El Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In *Proceedings of CRYPTO on Advances in Cryptology*, pages 10–18, 1985.

[22] Fatih Emekci, Divyakant Agrawal, Amr E. Abbadi, and Aziz Gulbeden. Privacy preserving query processing using third parties. In *Proceedings of the International Conference on Data Engineering (ICDE)*, pages 27–36, 2006.

[23] Arik Friedman and Assaf Schuster. Data mining with differential privacy. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '10, pages 493–502, 2010.

[24] Jerome H. Friedman. *Ann. Statist.*, pages 1–67, 1991.

[25] Benjamin C.M. Fung, Thomas Trojer, Patrick C.K. Hung, Li Xiong, Khalil Al-Hussaeni, and Rachida Dssouli. Service-oriented architecture for high-dimensional private data mashup. *IEEE Transactions on Services Computing*, pages 373–386, 2012.

[26] Rosario Gennaro, Stanislaw Jarecki, Hugo Krawczyk, and Tal Rabin. Secure distributed key generation for discrete-log based cryptosystems. *Journal of Cryptology*, 20(1):51–83, 2007.

[27] Oded Goldreich. *Foundations of Cryptography*. Cambridge University Press, 2004.

[28] Markus Jakobsson and Ari Juels. Mix and match: Secure function evaluation via ciphertexts. In *ASIACRYPT*, 2000.

[29] Markus Jakobsson, Ari Juels, and Ronald L. Rivest. Making mix nets robust for electronic voting by randomized partial checking. In *USENIX*, 2002.

[30] Wei Jiang and Chris Clifton. A secure distributed framework for achieving k-anonymity. *The VLDB Journal*, pages 316–333, 2006.

[31] Murat Kantarcioglu and Onur Kardes. Privacy-preserving data mining in the malicious model. *Int. J. Inf. Comput. Secur.*, 2:353–375, 2008.

[32] G . V. Kass. An exploratory technique for investigating large quantities of categorical data. *Appl. Stat.*, pages 119–127, 1980.

[33] Krishnaram Kenthapadi, Aleksandra Korolova, Ilya Mironov, and Nina Mishra. Privacy via the johnson-lindenstrauss transform. *CoRR*, abs/1204.2606, 2012.

[34] Ios Kotsogiannis, Ashwin Machanavajjhala, Michael Hay, and Gerome Miklau. Pythia: Data dependent differentially private algorithm selection. In *Proceedings of the 2017 ACM International Conference on Management of Data*, SIGMOD '17, pages 1323–1337, 2017.

[35] Yehuda Lindell and Benny Pinkas. Privacy preserving data mining. In *Proceedings of the Annual International Cryptology Conference on Advances in Cryptology*, pages 36–54, 2000.

[36] Yehuda Lindell and Benny Pinkas. Secure multiparty computation for privacy-preserving data mining. *Journal of Privacy and Condentiality*, 1(1):59–98, 2009.

[37] Ashwin Machanavajjhala, Daniel Kifer, Johannes Gehrke, and Muthuramakrishnan Venkitasubramaniam. L-diversity: Privacy beyond k-anonymity. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 2007.

[38] Frank McSherry and Kunal Talwar. Mechanism design via differential privacy. In *Proceedings of the Annual IEEE Symposium on Foundations of Computer Science*, FOCS '07, pages 94–103, 2007.

[39] Frank D. McSherry. Privacy integrated queries: An extensible platform for privacy-preserving data analysis. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, SIGMOD '09, pages 19–30, 2009.

[40] Noman Mohammed, Dima Alhadidi, Benjamin C. M. Fung, and Mourad Debbabi. Secure two-party differentially private data release for vertically partitioned data. *IEEE Trans. Dependable Secur. Comput.*, pages 59–71, 2014.

[41] Noman Mohammed, Rui Chen, Benjamin C. M. Fung, and Philip S. Yu. Differentially private data release for data mining. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, pages 493–501, 2011.

[42] Noman Mohammed, Benjamin C. Fung, and Mourad Debbabi. Anonymity meets game theory: Secure data integration with malicious participants. *The VLDB Journal*, pages 567–588, 2011.

[43] Noman Mohammed, Benjamin C.M. Fung, Patrick C.K. Hung, and Cheuk-kwong Lee. Anonymizing healthcare data: A case study on the blood transfusion service. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '09, pages 1285–1294, 2009.

[44] Noman Mohammed, Xiaoqian Jiang, Rui Chen, Benjamin C M Fung, and Lucila Ohno-Machado. Privacy-preserving heterogeneous health data sharing. *Journal of the American Medical Informatics Association*, page 462, 2013.

[45] Prashanth Mohan, Abhradeep Thakurta, Elaine Shi, Dawn Song, and David Culler. Gupt: Privacy preserving data analysis made easy. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, SIGMOD '12, pages 349–360, 2012.

[46] John Ross Quinlan. Induction of decision trees. *Mach. Learn.*, pages 81–106, 1986.

[47] John Ross Quinlan. *C4.5: Programs for Machine Learning*. 1993.

[48] Pierangela Samarati. Protecting respondents' identities in microdata release. *IEEE Trans. on Knowl. and Data Eng.*, pages 1010–1027, 2001.

[49] Dong Su, Jianneng Cao, Ninghui Li, Elisa Bertino, and Hongxia Jin. Differentially private k-means clustering. In *Proceedings of the ACM Conference on Data and Application Security and Privacy*, CODASPY '16, pages 26–37, 2016.

[50] Jaideep Vaidya and Chris Clifton. Privacy-preserving k-means clustering over vertically partitioned data. In *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '03, pages 206–215, 2003.

[51] Jaideep Vaidya, Chris Clifton, Murat Kantarcioglu, and A. Scott Patterson. Privacy-preserving decision trees over vertically partitioned data. *ACM Trans. Knowl. Discov. Data*, 2:14:1–14:27, 2008.

[52] Jaideep Vaidya, Basit Shafiq, Anirban Basu, and Yuan Hong. Differentially private naive bayes classification. In *Proceedings of the IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT) - Volume 01*, WI-IAT '13, pages 571–576, 2013.

[53] Dinusha Vatsalan, Peter Christen, and Vassilios S. Verykios. A taxonomy of privacy-preserving record linkage techniques. *Information Systems*, pages 946–969, 2013.

[54] Rebecca Wright and Zhiqiang Yang. Privacy-preserving bayesian network structure computation on distributed heterogeneous data. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, pages 713–718, 2004.

[55] Xiaokui Xiao and Yufei Tao. Anatomy: simple and effective privacy preservation. In *Proceedings of the International Conference on Very Large Data Bases (PVLDB)*, pages 139–150, 2006.

[56] Jun Zhang, Graham Cormode, Cecilia M. Procopiuc, Divesh Srivastava, and Xiaokui Xiao. Privbayes: Private data release via bayesian networks. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, SIGMOD '14, pages 1423–1434, 2014.

[57] Jun Zhang, Xiaokui Xiao, Yin Yang, Zhenjie Zhang, and Marianne Winslett. Privgene: Differentially private model fitting using genetic algorithms. In *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, SIGMOD '13, pages 665–676, 2013.