UNIVERSITY OF TURIN

DOCTORAL SCHOOL OF SCIENCES AND INNOVATIVE TECHNOLOGIES PHD

PROGRAM IN COMPUTER SCIENCE

XXXIII CYCLE

PhD Dissertation

Faisal Imran

Randomization Protocols for Privacy Preserving Data Mining

Advisors

Rosa Meo

Università degli Studi di Torino, Italy

PhD Coordinator

Marco Grangetto

Academic Year 2019-2020

Without privacy
there was no point
in being an
individual

Author, *Jonathan Franzen*

Dedication

This thesis is dedicated to my parents. My father Ali Hassan did not only raise me but also taxed himself dearly over the years for my education and intellectual development. My mother, Survery Khannam has been a source of motivation and strength during moments of despairs and discouragement. Her motherly care and support have been shown in incredible ways recently. I would also like to dedicate this dissertation to my friends and family members, who always supported me and help me at any time I needed help.

In the last, I would like to dedicate this work to my brilliant and loving wife, Noreen. She has been the inspiration in writing this thesis. Through thick and thin she stood by my side no matter how rough things seemed to be. She truly is my best friend and my confidant. I am truly thankful for having you in my life. I also dedicate this thesis to my son, Muhammad Afnan and our newel arrived daughter, Mirha Faisal.

Acknowledgement

This dissertation would not have been possible without the support of an outstanding supervisor, friends, teachers, and family members. First and foremost, I am deeply grateful for the continuous support, insight, and patience of my supervisor, Dr. Rosa Meo: without her constant trust and, sometimes, gentle prodding, this thesis would not have been completed. I appreciate that she always found ways to support my interests, and I have greatly benefited from her comments and feedback. I am also very grateful for her help with my writing and presentation skills, and I hope I have improved on them over the years.

Many thanks to my coauthors and collaborators. Federico Delussu is an extraordinarily inspiring researcher; thank you Federico for your encouragement throughout the completion of this research work.

Many thanks to the IT department of the University of Torino staff for their extraordinary help in navigating the Torino bureaucracy. Thanks for their constant support in providing me excellent and healthy working environment. I would also like to thank my department for providing me with financial support.

Abstract

Publishing users' sensitive information for research or statistical analysis, in a way that preserves the privacy of an individual, is an active area of research. The current de facto standard of privacy is differential privacy. We focus on the notation of differential privacy. The guarantee of differential privacy is that the outcome of a computation, whichever it is, will not have a significant impact on no individual's information. In this thesis, we study randomization protocols for the aggregation of sensitive information in a differentially private manner. Our proposed solution can be executed on a trusted or non-trusted data curator who, as we will see, is the responsible for data aggregation in case data is collected from many sources.

Our approach utilizes the randomized response technique in a novel manner: it provides privacy guarantees to users during the data collection and at the same time preserves the high-utility of the analysis. The randomised response technique protects the individual's data privacy by injecting a controlled amount of noise in data so that the original data is not released as it is. The advantages of randomised response techniques are that the production of the noisy outcome can be done "locally" where the user's data reside, using local differential privacy. Local differential privacy ensures a privacy guarantee using differential privacy and in addition, it also ensures that users' information is never visible to anyone, neither to the data curator who aggregates data from the sources.

In this thesis, we also explore the possible modifications of our proposed randomization mechanism, since generating all the possible combinations of the attributes will be computationally expensive when the domain size is large. To reduce the computational cost; we use low-degree Bayesian networks to reduce the attributes combinations in the contingency tables by focusing only on the attribute having higher mutual information. In this work, we present a privacy-preserving learning procedure that learns the structure of a Bayesian network from the noisy data where the protection to the users' data is ensured by the addition

of noise to the original data. In the second proposed version of our randomised protocol, we utilize the learnt Bayesian networks to perturb the contingency tables that represent the low-dimensional joint distribution of variables. We approximate the joint distribution of users' data using the product of the set of lower-dimensional marginals. We use our modified version of the randomization protocol to inject noise into each marginal to ensure differential privacy, and then use these noisy marginals along with the Bayes network to construct an approximation of the data distribution. We can also generate synthetic data by sampling tuples from this approximated distribution. In this thesis, we also combine our randomization protocol with another mechanism to insert noise in data, that is useful whenever the data has discrete values, that is called the α geometric mechanism. In this way we create a hybrid randomization protocol which ensures a privacy guarantee while maximizing the utility in the released data.

In this thesis, we also perform experiments on the use of Bayesian networks in practical application domains. To this aim, we exploit the use of a controller area network (CAN-bus) to monitor sensors on the buses of local public transportation in a big European city. The aim is to advise fleet managers and policymakers on how to reduce fuel consumption so that air pollution is controlled and public services are improved. We deploy heuristic algorithms and exhaustive ones to generate Bayesian networks among the monitored variables. The aim is to describe the relevant relationships between the variables, to discover and confirm the possible cause–effect relationships. In addition we predict the fuel consumption dependent on the contextual conditions of traffic, and enable an intervention analysis to be conducted on the variables so that our goals of fuel reduction are achieved. We propose a validation technique of the Bayesian networks based on the principle of Granger causality: it relies upon the observations of the time series formed by the successive values of the variables in time. We use the same method based on Granger causality as well, to rank the multiple Bayesian networks obtained by the adoption of different discovery algorithms based on heuristics, with the purpose to evaluate the alternative Bayesian networks and select the most promising one.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

The data collected from smart devices, the Internet of Things (IoT), and Smart Homes can be used for mining purposes and are potentially beneficial for organizations having a large user base, such as Google, Amazon, and Samsung. Data collected from such devices have become an invaluable asset for research purposes, statistical analysis, product designers, and application developers. Such datasets contain personal and sensitive information about individuals. For example, the dataset may contain medical profile data, genomics, financial transactions, socioeconomic attributes, and geolocation data. When publishing such datasets for statistical analysis, the data curator faces a tradeoff between data utility and data privacy. On one hand, if the data curator publishes such data (or even some statistical information about it), the outcome of the publication can be beneficial for the society and the population like in applications for the early monitoring of emergencies, the support to elderly people from remote and for improving the services of organizations having a large user base. On the other hand, for legal and ethical reasons, the privacy of the individuals in the dataset should be protected. Privacy-preserving data analysis aims to accommodate these objectives by providing a balance between data utility and privacy.

In recent years, data breaches have resulted in the unintended disclosure of millions of users' sensitive information [2, 3, 4, 5]. Data breaches are sometimes immutable: once the dataset is published, there is no repossess. Sometimes these breaches can be reduced. For example, if someone stole my password, I can easily change it, I can easily block and reissue a new credit card if it gets stolen. But, there might be situations where the data breaches result in permanent damage. For example, suppose leaked data contains someone's medical history. There is no way the individual can change his medical diagnosis just because the

data has been exposed in a data breach. Hence, data breaches are permanent, therefore, the published data must be protected by any means.

Privacy-preserving data publishing means publishing sensitive information in such a way that it can fulfill the research goal and at the same time it maintains the privacy of an individual. The data owner collects the data and applies some modification techniques to erase or modify the personally identifiable information to protect the sensitive information of individuals. This modified data is then published to fulfill the research purposes. For extensive acceptance, all data-driven applications must guarantee a high level of privacy to their users and accuracy to their applications. Differential privacy is a current de facto standard of privacy, which imposes a statistical requirement on the output of the published data. A significant amount of work has been conducted in achieving this guarantee while maximizing the utility of the released data, especially in the centralized trusted third-party aggregator [6, 7, 8, 9]. However, in recent years each participant in a centralized trusted aggregator is allowed to ensure the differential privacy guarantee in the release of information in isolation with respect to the other participants. This gives the mechanisms like *Federated Learning* [10] and *Local Differential Privacy* (LDP) [11].

The mechanisms of local differential privacy promises statistical disclosure control using differential privacy with a further assurance to the participants that their data is never visible to anyone and they retain plausible deniability of their private information. LDP solves the privacy in a centralized trusted third-party data aggregator that collects randomized responses from the users. In this thesis, we propose that in the adoption of randomization protocols we do not need and are not willing to rely on a trusted data curator because this one is also a single point of failure in the mechanism of data collection and of machine learning model creation. Randomized response mechanism randomized the outcome of the data at the client machine locally, without compromising the users' sensitive information. According to the solution we propose, the curator can still build reliable prediction models on the collected amount of randomized data. Our solution utilizes the randomized response technique in a novel manner: it provides a privacy guarantee during the data collection and simultaneously maximizes the data utility in statistical analysis.

1.1 Outline and Contributions

We divide the contribution of this thesis into four main parts. In the first part, we present our randomize response block aggregation technique that utilizes randomize response to collect the users' sensitive information. In the second part, we combine our proposed model with a geometric mechanism to build a sophisticated randomization technique that is not only a privacy guarantee but also provides high utility in the released data. In the third part, we characterize the notion of utility, both theoretically and by means of experimental analysis in which we compare our protocol with traditional privacy-preserving mechanisms such as differential privacy with Laplace distributed noise. In the last part, we present Bayesian heuristic algorithms, driven by the BIC score and brute force with the purpose of comparing the ability of the algorithms to converge to the same resulting networks. We evaluated their results with the adoption of Granger causality, a third-party criterion, based on the time series formed in time by the observed variables. Chapter 2 and 3 will cover the necessary background, syntactical privacy preservation and failure, and differential privacy.

Chapter 2: Insecure Techniques: Exploring Approaches with Limited Semantic Security

Chapter 2, We start by providing some of the traditional privacy-preserving models, their drawbacks, and privacy issues. Specifically, we show some of the methods for the privacy-preserving data publication practice of releasing datasets while protecting the privacy of individuals whose data is included. The goal is to provide useful information while minimizing the risk of re-identification or unauthorized access to sensitive data. It is important to note that no method for privacy-preserving data publication is perfect. There is always the possibility that an attacker could use sophisticated techniques to identify individuals in an anonymized dataset. At the end, we provide some of the examples of significant privacy breaches that have occurred in recent years:

Chapter 3:Differential Privacy

In Chapter 3, we discuss about differential privacy, its properties, characteristics, and some well-known mechanism for achieving differential privacy. The central theme behind differential privacy is to add a controlled amount of noise to the output of the query in a way that makes it impossible to tell whether a specific individual's data was included or excluded in the dataset. This is done by adding random noise to the data, which makes it difficult to distinguish between the original data and noisy data. Most of the literature on differential

privacy focuses on the setting of centralized computations: where the server performs data analysis and outputs aggregate information in a privacy-preserving manner. We work beyond this limitation and provide a mechanism in which the server/aggregator does not access any client's true values or even build a survey dataset in a privacy-preserving manner from scratch.

Chapter 4: Randomization Response Block Aggregation Model

In Chapter 4, we proposed a randomization response block aggregation model. In this proposed technique, the natural and more general setting is when each client has multiple attributes, and the server is interested in learning the joint distribution of subsets of these attributes. In our approach we do not wish to harm the data owners in the sample by adopting an approach of local differential privacy (LDP) rather than adopting the weaker global differential privacy (GDP). With GDP there is the presence of one or more aggregators which store the true data and could be single point of failures and be the goal of attacks. LDP is stronger than GDP because it adds the restriction that even adversaries might have access to the personal responses and they still will be unable to learn about the single users' personal data. In the proposed protocol, the private, randomized data of a respondent is generated after the selection of subsets of the attributes. The values of these attribute subsets are communicated in a distributed environment to one or more aggregators. The single limitation is that the individual respondent does not have to communicate the same attribute in multiple subsets but just in one. The aggregator who receives the randomized data values from the subset of attributes has the task of computation of contingency tables with the frequencies of the observed values of the attribute subsets. Thanks to the protocol properties, we demonstrate that it is possible to reconstruct with an acceptable accuracy the true joint probability of the attribute subsets from the possibly noisy values, communicated by the individuals using the randomization protocol. The values do not need to correspond to the true ones for each individual, thanks to the deniability property of the protocol. Moreover, in our randomized protocol, the randomization is local to the individual users, and there is no need for a different, trusted organization to perform the randomizer or the addition of a verified amount of noise.

Chapter 5: Randomized Response, a Modified Version and α-Geometric Mechanism

In Chapter 5, will present the second version of our randomized response block aggregation

protocol discussed in the first part. In this second version, we use Bayesian networks to limit the joint distribution of variables at a lower-dimension to approximate the full-dimension of the data. Hence it will reduce the overall computation time while preserving the utility of the data in the synthetic dataset. Bayesian networks (BN) is employed also to perform an assessment of the observed phenomena and to perform an intervention analysis on the causal variables so that the monitored target can be improved. We construct a private Bayesian network by executing the GreedyBayes algorithm [1] and collecting the randomized response from the clients on the attribute-parent pairs identified by GreedyBayes. This phase will help in limiting the number of attributes pairs in the contingency tables to be considered because the remaining attribute pairs can be eliminated since they are considered as statistically independent by the Bayesian network produced by GreedyBayes. Once the Bayesian structure is constructed, we launch our 2^{nd} version of the randomization protocol to focus on the fixed numbers of attribute-parent pairs to collect randomized responses in the subspace of the attributes that results dependent in the discovered BNs. We created synthetic data from the differentially private Bayes structure, without explicitly materializing the full-dimensional distribution.

Chapter 6: Bayesian Networks for Fuel Prediction and Reduction in Public Transportation

In Chapter 6, we employ machine learning models, specifically, Bayesian networks, to analyze sensor data installed on the buses of a public transport company in a European city. The sensors collect data about the vehicle and its use (acceleration, braking, speed, stop durations with the engine on, etc.) with some contextual information about the vehicle location (such as altitude). An analysis of the sensor data using machine learning algorithms applied using procedures of predictive maintenance can also be used to improve vehicle equipment maintenance, with a reduction in costs due to stop times for faults and repair. The main contribution of this work is to provide a BN on the variables monitored by sensors connected in CAN-bus. These BNs show which variables we should change to control the fuel consumption variable. Furthermore, BN also supports simulation of the behavior of the system. We use the BIC score [12], a derivation of the likelihood of the data under the assumed BN model, as a heuristic to evaluate the alternative networks. We revised them and compared their solutions on the sensor data by providing a brute force alternative. Brute force converges to the global optimum of the BIC score within the search space. The brute force

alternative is possible (provided the number of variables is kept limited to some units) thanks to the opportunity that high-performance computing gives us. It makes the workload efficient by distributing the computation among multiple servers and CPUs, and their execution in parallel. One novelty of our approach and the last, but not least, contribution of our work is Granger causality. Granger causality and its statistical test employ vector auto-regression (VAR) as a tool to predict the target in time with the aid of multiple variables (the variables that are in the pathway from causes to the effect). In its essence, the statistical test in the Granger causality method verifies that the prediction of the target, with the aid of the cause variables, is better with a statistical significance guarantee, than without them. The application of this latter criterion is possible only when the flow of values of these variables is stored in time. Granger causality is commonly judged as a weaker principle than the stricter principle of probabilistic dependency between cause and effect. With Granger causality, the existence of a causality relation between cause and the effect is verified only in time thanks to the ability of the cause to predict and anticipate the effect in time.

Chapter 7:Utility Privacy Trade-off in a Differential Private Mechanism

In the last Chapter 7, we model the notation of utility and privacy suitable for the contingency tables. In the context of local differential privacy, there is a trade-off between utility and privacy. The more randomness in the output of the randomized mechanism, the stronger the individual privacy. On the other hand, this makes more noisy the aggregator's estimations. To characterize this utility and privacy trade-off, we need a confidence interval on the probability of observed responses that measures the utility and privacy in the underlying protocol. Using this confidence interval, we can identify a minimal level of privacy that a protocol provides while maximizing its utility. We also perform differential private hypothesis testing to evaluate the effect of Laplace noise on the presence/absence of an arc in a Bayesian network.

Chapter 8

Chapter 8 concludes this dissertation and discusses a few interesting and non-trivial directions for future research.

Chapter 2

Insecure Techniques: Exploring Approaches with Limited Semantic Security

In recent years privacy-preserving data publishing has been a challenging task. Over the years extensive research has been conducted in this domain. The data publisher aims to release the data without exposing sensitive information about the individuals. The publication of such data set includes medical records, financial transactions, census reports, and so on, to use in medical research and social and economic analysis. Hence the goal of privacy-preserving data publishing is to publish sensitive information in such a way that it can be utilised for the intended research and at the same time maintains the privacy of individuals. Formerly, many privacy models exist in the literature to protect the privacy of individuals. Such models include k-anonymity, l-diversity, t-closeness, and differential privacy [13, 14, 15, 16].

2.1 Introduction

The privacy concept of any database is to limit the ability of an adversary to learn any new information about a particular participant after accessing the database [17, 18]. The idea behind this terminology is that the adversary's knowledge about a particular individual remains the same before and after accessing the data. However, this type of privacy assurance is not practical due to the availability of background knowledge. Dwork [16] introduces a

new concept of privacy preservation; the risk to one's privacy should not significantly increase as a result of participating in a database, for example, denial of one's insurance due to his participation in an insurance survey.

Privacy-preserving data publishing (PPDP) means publishing sensitive information in such a way that it can fulfill the research goal and at the same time it maintains the privacy of an individual. The data publishers (aggregators) collect the data and apply some modification techniques to erase or modify the Personally Identifiable Information (PII) to protect the sensitive information of individuals. This anonymized data is then published for the intended purposes.

Personal Identifiable Information (PII)

Personally identifiable information (PII) or personal data is the information used to identify, distinguish or track an individual's identity in the data. According to the data privacy publishers, personally identifiable information can be categorized into four different classes namely, *Quasi-identifier, Explicit identifier, Sensitive, and Non-Sensitive attributes*. Quasi-identifiers are information that cannot uniquely identify an individual, but they can be linked with other quasi-identifiers to create a unique identifier that can sufficiently distinguish an individual. Information in quasi-identifiers includes postal code, birthday, and gender. Explicit information includes name or social security number which can be used to identify any individual. Sensitive attributes include sensitive information about a person such as his medical history or salary and non-sensitive attributes are all those attributes that are not listed above [17].

PII is a piece of information that can be utilized all alone or can be linked with other information to identify individuals. For example, name, date of birth, and social security number [17]. The data curator, before releasing the data uses a *De-identification* process to protect individual privacy while ensuring data utility. Many privacy-preserving techniques have been proposed earlier: in the next Section, we will discuss them briefly.

2.2 Approaches with Limited Semantic Security

In recent years private data analysis has gained adequate popularity in the research community. Many private data learning and data publication models have been studied in the literature [19, 20, 1, 21]. The main aim of all these models is to publish sensitive data to preserve the privacy of individuals participating in the dataset while maximizing the utility

of the data statistical analysis. It is interesting to highlight these approaches because the issues raised by these models can be solved by differential privacy and some of them are the building blocks for the definition of differential privacy.

In general, there are two main approaches to achieving privacy-preserving mechanisms. The *interactive* or *online* and *non-interactive* or *offline* approach. In the interactive setting, the *data curator* known as a trusted entity provides an interface through which the users query the data and receive a noisy response [22]. The interactive setting is used when information about the queries is not known in advance, as this poses severe challenges to the non-interactive model [23]. If the query information is known in advance, any interactive solution poses a non-interactive solution. In this case, the data curator poses the known queries and then publishes the results [24]. In a non-interactive setting, the curator sanitized the published data by modifying the key identifiers. This modification of key identifiers is known as *anonymization* or *de-identification*. In the next section, we will discuss some of the approaches for data anonymization.

Data anonymization is a technique of protecting individual identity by removing, masking, or encrypting identifiers that link the users whose data are stored in the database. Through the data anonymization process, you can mask the PII such as name, address, and social security numbers, which allows us to keep the data but makes the source anonymous. Data anonymization enables data controllers to publish data among departments within the same company or among different companies while focusing on reducing the risk of unintended data disclosure. Although, data anonymization removes the individuals identifier, attackers can use the *de-anonymization* process to retrace the data anonymization process [2]. Since data usually passes through multiple sources—some available to the public, de-anonymization techniques can cross-reference the sources and reveal personal information. For example the Netflix Prize competition released the users' rates on movies [3] and AOL released anonymized search queries [4].

There are several approaches to achieve data anonymization, *data masking* (use altered values to hide private data), *generalization* (intentionally delete some of the identifiers to make it less identifiable), *Pseudonymization* (replace private with fake identifiers), *data swapping* (shuffling and permutation of private identifiers), *data perturbation* (adding random noise), and *synthetic data* (algorithmically generated information that has no connection to true data).

2.2.1 Privacy Using K-Anonymity

To deal with the inadequacy of simple data anonymization methods, Samarati and Sweeney proposed *K-Anonymity* [13]. Well-known model to protect individual privacy. k-Anonymization is often referred to as the power of **"hiding in the crowd"**. Individuals' sensitive information is pooled in a larger group, meaning information in the group could correspond to any single participant, thus hiding the identity of the individual or individuals in question. The keyword **K** in k-anonymization is referred to the number of times each attribute combination appears in the database. A k-anonymized dataset promises that each individual in the database is similar to at least another k-1 other records on the possible identifiable attribute. K-anonymization protects the data from de-anonymization using linkage attacks.

K-anonymization can be implemented using **Generalization** and **Suppression**. In generalization, we substitute a specific attribute with a broader category. For example age attribute can be generalized into an different age groups (i.e. grouping 'Age: 25', 'Age: 30', and 'Age: 35' into 'Age Group: 25-35'). In suppression, we remove the attribute value from the dataset or mask the attribute value with an '*'. We can also substitute the entire column of the database with an '*' or replace the entire column with other specific values. For example, Table 2.1 shows *3*-anonymous generalization and suppression technique. The adversary knows Alice's zip code, age, and gender (47677, 29, F) as shown in Table 2.1a; after anonymization we obtain Table 2.1b and the adversary does not know which one of the first 3 records corresponds to Alice's record.

Machanavajjhala et al. [14] show that k-anonymity does not guarantee strong privacy. It does not provide privacy if the adversary has some background knowledge or if the sensitive attribute lacks diversity in an equivalence class.

2.2.2 Attacks against K-anonymity

Although *k*-anonymity provides a promising approach to protect the sensitive attributes of an individual using a simple wide array of algorithms, unfortunately, the solution provided by *K*-anonymity can still be vulnerable to attacks if the adversary has some background knowledge. Such attacks include:

- **Background knowledge attack:** In this attack, an advisory links one or more quasi-identifiers with the sensitive attribute to limit the set of possible values for the sensitive

Quasi identifier			Sensitive attribute
Age	Gender	Zipcode	Disease
29	F	47677	Heart disease
22	F	47602	Heart disease
27	M	47678	Heart disease
43	M	47905	Cancer
52	F	47909	Flu
47	M	47906	Heart disease

(a) Microdata

Quasi identifier			Sensitive attribute
Age	Gender	Zipcode	Disease
2*	*	476**	Heart disease
2*	*	476**	Heart disease
2*	*	476**	Heart disease
[43,52]	*	4790*	Cancer
[43,52]	*	4790*	Flu
[43.52]	*	4790*	Heart disease

(b) *K*-Anonymization table

Table 2.1: Example of *k*-Anonymity using generalization and suppression techniques: (a) shows microdata, and (b) shows 3-anonymous data; column *Age* and *Zipcode* anonymized using generalization and *Gender* is anonymized using suppression

attribute. For example, the advisory knowledge about the ratio of heart patients is low in Japan, thus limiting the values of a sensitive attribute of patients' disease.

- **Homogeneity Attack:** The advisory can predict the value of a sensitive attribute among the set of k-records if the diversity of the attribute values is limited. For example, with reference to the example shown in Table 2.1a if the attacker knows Alice's zip code is 47678 and her age is 27, he can determine she is suffering from heart disease.

2.2.3 *l*-Diversity

L-diversity is an extension of *k*-anonymity. The drawbacks of *k*-anonymity are overcome by *l*-diversity hence protecting the sensitive attributes against inference attacks. To achieve data privacy, *l*-diversity performs group-based anonymization by working on the granularity of the data representation. In a table that satisfies the *l*-diversity property, in the group in which a tuple shares similar quasi-identifiers there are at least *l*-diverse well-represented values for the sensitive attributes. The core principle of *l*-diversity stated that: *the l-diversity of an equivalence class holds if the equivalence class contains at least l "well-represented" values for the sensitive attribute*. A data table is said to be an *l*-diversity table if it contains a number

of distinct values for the sensitive attributes equal to the *l*-diversity parameter. Table 2.2 illustrates an example of *2*-diverse table: if we have four types of diseases, then according to *2*-diversity the records in the equivalence class should expose the sensitive value as no less then two of the four possible values so as to avoid the possibility of an homogeneity attack based on the sensitive attribute.

Quasi-identifiers			Sensitive attribute	2-diverse
Age	Gender	Zipcode	Disease	
Under 30	*	476**	Heart disease	QI: Group 1
Under 30	*	476**	Heart disease	
Under 30	*	476**	Flu	
Over 40	*	479**	Heart Disease	QI: Group 2
Over 40	*	479**	Heart Disease	
Over 40	*	479**	Cancer	

Table 2.2: Example of *2*-diverse table

Machanavajjhala et al. [14] also define other variants of *l*-Diversity.

1. **Distinct *l*-diversity:** This variant is the simplest among the other two and defines that there must be at least l-distinct values of the sensitive attributes.

2. **Recursive (*c, l*) diversity:** Recursive diversity ensures that the least frequent values do not appear too rarely and most frequent values do not appear too frequently.

3. **Entropy *l*-diversity:** Formally, the entropy of an equivalent class E for a particular sensitive attribute with domain d can be defined as:

$$H(E) = -\sum_{d \in D} pr(E, d) \cdot log \cdot pr(E, d)$$

where $pr(E, c)$ is the probability of an attribute having the value d in group E. A dataset is entropy *l*-diversity if, for each group E the entropy $H(E) \geq log\ l$.

Attacks against *l*-diversity

Although *l*-diversity resolves the limitations of *k*-anonymity, however, this technique is not immune to disclosure attacks. Li et al. [15] demonstrated two attack models on *l*-diversity.

- **Skewness attack:** *L*-diversity does not consider the overall distribution of sensitive values. For example, QI group 1 from the Table 2.2 has two out of three patients

with heart disease. If an advisory links a particular patient to that group, that individual can be considered to have a 66% probability of having heart disease, rather, it is considerably less likely to identify the same from the entire database. The cost of *l*-diversity in this scenario is information loss, as many of the 'non-sensitive' records will have been removed from the Table to meet the *l*-diversity criterion.

- **Similarity attack:** *L*-diversity does not consider semantic meanings of sensitive values, which means when the equivalence class has different but semantically similar values in the sensitive attribute. For example, if patients in a *3*-diverse dataset where the disease is a sensitive attribute having values (lung cancer, stomach cancer, and liver cancer) an adversary can infer that an individual has cancer by linking him to that group.

2.2.4 t-Closeness

L-diversity overcomes the shortcoming of *k*-anonymity, however, it has its limitations. It is very difficult to achieve *l*-diversity for a large database's size, many equivalent classes will be needed to satisfy *l*-diversity. The *l*-diversity approach is insufficient to prevent sensitive attribute disclosure which led to the proposal of another privacy definition called *t*-Closeness proposed by Li et al. [12]. *t*-Closeness anonymized data by keeping each quasi-identifiers sensitive attribute close to their distribution in the database. If the distance between the distribution of a sensitive attribute within an equivalence class and its distribution in the whole table is not greater then the threshold *t*, then the equivalence class is said to have t-closeness. If all equivalence classes within a table have t-closeness then the table is t-closeness. Li et at. [15] use Earth Mover Distance [25] to calculate the semantic relationships among sensitive attributes.

For example, let P and Q denote the distribution of sensitive attributes and all the attributes in the database respectively. Given a threshold t, the equivalent class satisfies *t*-closeness if the distance between P and Q is less than or equal to t.

2.3 Privacy Issues and Failures

Online service provider like Facebook[1], Google[2], Amazon[3], etc,. collect and store as much data as possible, either because they want to improve the users experience, accuracy of prediction in their recommendation systems, or to materialize the stored data by selling it to third parties. Data collected by online services which is not considered necessary might disintegrate the user's expectation about privacy. Online services can build sophisticated models of users' personality from the collected data without the user consent raises ethical, privacy, and security issues[4],[5]. Cambridge Analytica Ltd (CA) is a political consulting firm that combines user profiles, data brokerage, data mining, and data analysis with the goal to achieve a strategic communication during certain specific events, such as the electoral processes[6]. In March 2018, several news agencies reported news about Cambridge Analytica business practices. It was uncovered that Cambridge Analytica required personal information of Facebook users from the third party, who collected this data for research purposes from Facebook. It was investigated that Cambridge Analytica was exploiting users' private information of users to gain political advancement for some politician.

According to the previous studies [26, 27], the majority of US citizens can be identified by combining information from different databases by joining them using the values of quasi-identifiers, such as gender, date of birth, and zip code, thus disclosing sensitive information in the released data. Adversary's background knowledge makes the privacy preservation models vulnerable. As discussed in the previous section, these privacy techniques do not provide privacy guarantees and are vulnerable because of the adversary background knowledge. We will discuss some of the anonymization fiascos, which led to the identification of individuals.

Latanya Sweeney [13] showed that an individual's information can be easily re-identified by linking the anonymously released data with the publicly available data (e.g., voter regis-

[1]Facebook. URL https://www.Facebook.com

[2]Google Inc URL https://www.Google.com

[3]Amazon URL https://www.amazon.com

[4]Davies. Ted Cruz using firm that harvested data on millions of unwitting Facebook users (11 December 2015). URL https://www.theguardian.com/us-news/2015/dec/11/senator-ted-cruz-president-campaign-facebook-user-data

[5]Brannelly. Trump Campaign Pays Millions to Overseas Big Data Firm (4 November 2016). URL https://www.nbcnews.com/storyline/2016-election-day/trump-campaign-pays-millions-overseas-big-data-firm-n677321

[6]Ingram. Factbox: Who is Cambridge Analytica and what did it do? (20 March 2018) URL https://www.reuters.com/article/us-facebook-cambridge-analytica-factbox/factbox-who-is-cambridge-analytica-and-what-did-it-do-idUSKBN1GW07F

tration list) and background knowledge about the individual. She successfully re-identified the Massachusetts governor William Weld's by associating background knowledge (such as an address, zip code, birth date, and gender) with the anonymized data released by the Massachusetts Group Insurance Commission (GIC). She demonstrated that de-identified data could often be re-identified with the help of other available information, such as publicly available data or data from other sources. This means that de-identification alone may not be sufficient to guarantee anonymity. [26, 2].

American Online (AOL) released 20 million search logs of their users for research purposes: these search logs contain users' search history of three months. AOL anonymized these search logs by replacing the key identifiers such as username and IP address with unique identification numbers so that the researchers can relate the searches to that of the individuals. However, the username, his social security number, and even more sensitive information were disclosed based on all the searches made by that particular user. An article "A Face Is Exposed for AOL Searcher No. 4417749" was published New York Times[7] revealing the identity of one of the users, and subsequently, many other users were identified. Within a week, AOL removed the search logs from their server and apologized, saying the team published it for the benefit of academic research and published without any authorization. But copies of the detailed records continue to circulate online, emphasizing how much information people unintentionally reveal by just using search engines and how risky it can be for search engine companies like Google, Yahoo, Bing, and AOL to publish such data.

Netflix, the world's largest video streaming service provider published an anonymized dataset of 100 million movies rating from 500,000 subscribers. They aimed to improve their recommendation system. They removed the user's personal information and anonymized the user IDs, ratings, and dates on which users rated the movie. Narayanan and Shmatikov [28, 29] demonstrated that users' personal information can be disclosed if the adversary combines data from the Internet Movie Database (IMDB) as a piece of background knowledge.

The above-stated anonymization issues demonstrate that protecting individual private information is a challenging task and it can become more challenging if the adversary has sufficient background information that is even not anticipated by the data publisher. Hence we need a better, accurate, and robust privacy-preserving technique to protect data leakage and disclosure of individuals' private information in the worst-case scenarios where the

[7]Michael Barbaro. A Face Is Exposed for AOL Searcher No. 4417749 (9 August 2006). URL https://www.nytimes.com/2006/08/09/technology/09aol.html

adversary has some background knowledge.

The current state-of-the-art standard for privacy is **Differential privacy** proposed by Dwork [22]. Differential privacy (DP) provides strong mathematical promises on an individual's privacy, interpreted as a statistical property of the output of a query on the database when the individual is included in the database and when he is not. To achieve privacy, random noise is added in the query mechanism that responds to requests from the users/analysts. The privacy guarantee of a randomization technique is quantified by the privacy parameter. This privacy parameter controls how different the probabilities are when the randomization query returns the same result in two different situations: when the individual is included in the query response and when he is not.

Differential privacy provides a privacy guarantee against background knowledge attacks and can neutralize the linkage attacks as discussed before. Therefore, differential privacy has been considered a promising privacy-preserving technique.

Chapter 3

Differential Privacy

3.1 Introduction

In this chapter, we will discuss differential privacy and its notations. In particular, we will explain the notations and some useful properties of the most common mechanisms used to achieve differential privacy. In this chapter, we will cover some of the relevant literature which revolves around differential privacy and it will help us to establish our work.

3.2 Differential Privacy

The acceptance of modern data collection applications depends on the privacy guaranteed to the end-users. As a consequence of the extensive data collection performed by the curators, a massive amount of data is available whose existence triggers sophisticated attacks on the data storage. Any application that hopes to withstand such sophisticated attacks should provide rigorous privacy guarantees. The current de facto standard is differential privacy [30, 31]. Differential privacy (DP) provides firm mathematical promises on an individual's privacy, interpreted as a statistical property of the output of a query on the database when the individual is included in the database and when not. To protect the individual's privacy, noise is added either on the data or in the query mechanism ($\mathcal{M}$) that answers requests from the users/analysts on the data. The privacy guarantee of the randomization mechanism is quantified by the parameter of the privacy budget ϵ that controls how different can be the probabilities that the randomization query mechanism returns the same result in two different situations in which a single individual is included in the database and in which the individual is not

included.

3.2.1 Databases and Queries

Let us consider a database/dataset (we will use these two terms interchangeably) D as being a collection of records from *finite data universe* $\mathcal{X}$. We can also represent a database D by its histograms: $D \in \mathbb{N}^{|\mathcal{X}|}$, in which each entry D_i denotes the number of elements in the database D of type $i \in \mathcal{X}$ (the symbol $\mathbb{N}$ represents the set of all non-negative integers, including zero). The core importance of differential privacy is the notation of *neighboring databases*. The two databases $D_1 \sim D_2$ are neighbors if they differ by one individual row, record, or tuple.

Definition 3.2.1 (**Neighboring databases [30, 31].**). *Two databases D_1, D_2 are neighbors or neighboring if they differ in at most one individual row, i.e, the distance between two neighboring databases D_1, D_2 will be their ℓ_1 distance and it will be equal to 1. The ℓ_1 distance between two databases D_1 and D_2 is denoted by $\|D_1 - D_2\|_1$ which measures how many records D_1 differs from D_2.*

The two databases $D_1 \overset{t}{\sim} D_2$ can also be neighboring datasets if they differ only in the *value* or a tuple t. This type of neighboring dataset is presented in the literature to simplify the task of differential privacy, Dwork et al [23].

A randomization mechanism $\mathcal{M} : \mathbb{N}^{|\mathcal{X}|} \to \mathcal{R}$ takes an input from the database $D \in \mathbb{N}^{|\mathcal{X}|}$ and returns an output from some Range that belongs to $\mathcal{R}$ (the domain of the reals). Informally, this mechanism represents the output of a statistical query on the database. Its output is probabilistic so that if you run the mechanism twice it will not return the same output. The mechanism is designed to be probabilistic so that from its output the user cannot infer with certainty anything about the database content. The mechanism $\mathcal{M}(D)$ is presented as a stochastic method whose output is probabilistic and this probabilistic behavior is the essence of the privacy protection for the individual's data in the database. Informally, this randomization mechanism is said to be differentially private if the probability distribution $\mathcal{M}(D_1)$ on database D_1 produces approximately the same output as from the probability distribution on $\mathcal{M}(D_2)$, for every $D_1 \overset{t}{\sim} D_2$. We can also say that this randomization mechanism should be indifferent to the presence or absence of any one tuple t in the input database, More formally,

Definition 3.2.2. ***(Differential privacy [30, 31])***. *The definition is due to Dwork et al [30, 31]. A randomized mechanism $\mathcal{M}$ with domain $\mathbb{N}^{|\mathcal{X}|}$ is (ϵ, δ)-differentially private if for all $S \subseteq \text{Range}(\mathcal{M})$ and for all $D_1, D_2 \in \mathbb{N}^{|\mathcal{X}|}$ such that $\left\|D_1 - D_2\right\|_1 \leq 1$:*

$$\Pr\left[\mathcal{M}\left(D_1\right) \in S\right] \leq exp\left(\epsilon\right) \Pr\left[\mathcal{M}\left(D_2\right) \in S\right] + \delta \tag{3.1}$$

Where the probability space is over the flip of a coin of the randomization mechanism $\mathcal{M}$. If $\delta = 0$, we say that this randomization $\mathcal{M}$ is ϵ-differentially private. The above equation can be rewritten as,

$$\frac{\left[\Pr(\mathcal{M}(D_1) \in S)\right]}{\left[\Pr(\mathcal{M}(D_2) \in S)\right]} \leqslant \exp(\epsilon) \in \mathcal{R} \tag{3.2}$$

$(\epsilon, 0)-$differential privacy ensures that for every run of the mechanism $\mathcal{M}(D_1)$, the output observed is almost equally likely to be observed on every neighboring database D_2, at the same time. Mathematically, equation 3.2 states that the ratio between the probabilities of the randomized function $\mathcal{M}$ on the database D_1 and the neighboring database D_2 must be bounded by $(-\epsilon, \epsilon)$. The closer ϵ to 0, the more difficult for an attacker to determine an individual's private information. From the definition, ϵ is a privacy budget or privacy parameter that controls the strength of differential privacy in the output. The higher value of epsilon the lower the protection of privacy while smaller ϵ yields stronger protection of privacy. We can also say that this privacy budget ϵ is a knob that controls the privacy and utility in the output database. Higher protection of privacy deteriorates the utility of the data, whereas lower protection of privacy increases the utility but decreases the protection of privacy of the data given in the output of the randomized mechanism $\mathcal{M}$. Typically the value of epsilon are $0.01, 0.1$, or possibly $\ln 2$ or $\ln 3$ as suggested by, Dwork [32]. When epsilon is small, $\exp^{\epsilon} \approx 1 + \epsilon$.

3.3 Achieving Differential Privacy

A real-valued function $f : \mathbb{N}^{|\mathcal{X}|} \rightarrow \mathcal{R}$ will satisfy differential privacy by adding a carefully chosen random noise into the output value. The magnitude of the injected noise can be adjusted based on the *global sensitivity* and *local sensitivity* of a function, or the upper bound of the amount of noise added to any individual tuple returned by a mechanism in response

to a query on that tuple.

3.3.1 Global Sensitivity

The global sensitivity is the maximum difference in the output of a query on two neighboring datasets, formally:

Definition 3.3.1 (Global sensitivity [31])**.** *The global sensitivity of a real-valued function* $f : \mathbb{N}^{|\mathcal{X}|} \to \mathcal{R}^{\supseteq}$ *is the smallest number* Δf *of difference of* f *when* f *is applied to any pairs* $D_1, D_2 \in \mathbb{N}^{|\mathcal{X}|}$ *that differ in at most one single entry. The global sensitivity of function* f *is given by:*

$$\Delta f = \max_{D_1 \sim D_2} \left\| f(D_1) - f(D_2) \right\|_1 \tag{3.3}$$

where: $\|\cdot\|_1$ is the ℓ_1 *norm*.

This sensitivity Δf is a Lipschitz condition[1] on f: if the Hamming metric $d_H(\cdot,\cdot)$ on $\mathbb{N}^{|\mathcal{X}|}$, then for all pairs of databases $D_1, D_2 \in \mathbb{N}^{|\mathcal{X}|}$:

$$\Delta f = \max_{D_1 \sim D_2} \frac{\left\| f(D_1) - f(D_2) \right\|_1}{d_H(\cdot,\cdot)} \tag{3.4}$$

Global sensitivity depends on the property of the function f alone and it is independent of the input databases D_1 and D_2. A real-valued function f can be $\epsilon-$differentially private by injecting random noise drawn from *Laplace* distribution with zero mean and $\frac{\Delta f}{\epsilon}$ scale in the mechanism response, or by application of the exponential distribution function to the response, controlled by a utility function for answering queries with arbitrary utilities [30]. In our thesis, we will always use the first typology of noise addition.

3.3.2 Smooth Sensitivity

The global sensitivity defined in section 3.3 is a worst-case measure of how much of the output of a function f will be changed by changing a single row in the input. Hence global sensitivity is a function of the computation itself and it is independent of actual data. Global

[1]The function Δf is Lipschitz with respect to D on the domain $\mathbb{N}^{|\mathcal{X}|}$ if there is some constant K such that $|f(D_1) - f(D_2)| \leqslant K|D_1 - D_2|$ for every pair of points D_1 and D_2 in $\mathbb{N}^{|\mathcal{X}|}$. The constant K is called the Lipschitz constant for Δf on the domain $\mathbb{N}^{|\mathcal{X}|}$.

sensitivity emphasizes the function output by changing any possible value in the input. Consequently, inputs that are very rare in practice could lead to a higher sensitivity. Hence in case a lower amount of noise was added, it could eventually leak sensitive information. *Smooth sensitivity* proposed in [33], addresses the above cited problem with global sensitivity. The proposed solution suggested using a smooth bound on *local sensitivity*, rather than controlling the noise scale based on global sensitivity. Local sensitivity of a function is represented as $L\Delta f$ and it estimates the maximum difference in the output of the function f when run on true data D_1 and neighboring database D_2.

Definition 3.3.2. *Local sensitivity [33]*

Local sensitivity of a function f at x, with respect to ℓ_1 metric, $f : \mathbb{N}^{|\mathcal{X}|} \rightarrow \mathcal{R}^d$ is given by:

$$L\Delta f = \max_{(x,y):d(x,y)=1} \|f(x) - f(y)\|_1 \tag{3.5}$$

Where x and y belong to $\mathcal{X}^n$ (that are specific tuples values of the database domain). The local sensitivity is identical to the global sensitivity for the query functions `count` and `range`. Global sensitivity from Definition 3.3.1 is observed as:

$$\Delta f = \max_{\text{for any } x} L\Delta f(x) \tag{3.6}$$

3.3.3 Setting of Privacy Parameter ϵ

The core component of any differentially private algorithm is setting the level of privacy parameter ϵ. The amount of privacy in any DP mechanism can be controlled using ϵ. It can also express the privacy loss measured by the DP mechanism. Consequently, the DP mechanism does not declare how much data become "anonymous". Alternatively, ϵ is thought of as an auxiliary risk an individual is exposed to by participating in the data analysis.

Setting an appropriate value of ϵ is an open and unsolved problem. Additionally, what represents a "good" value of ϵ is context-dependent [34]. In the literature, many researchers reasoned on the strategies of the setting of ϵ, such as calculating the advisory advantage [35], using economical approach [36], and calculating the Bayesian posterior belief of the adversary [37]. Dwork [31] suggests setting the value of ϵ as low as possible: typically it ranges between 0.01, 0.1, $\ln 2$ or $\ln 3$. Still, there is no silver bullet for setting ϵ.

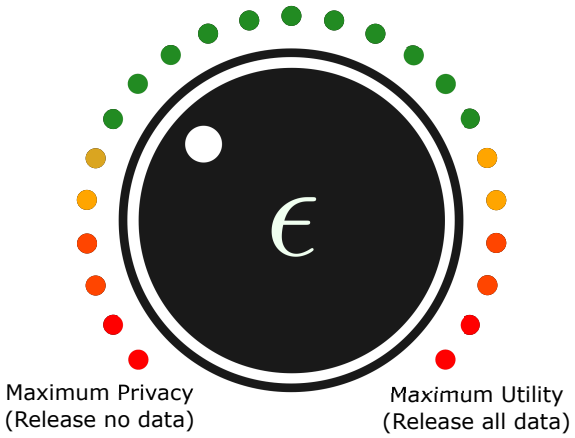


Figure 3.1: Overview of privacy and utility trade-off, where the size and placement of colored dots depend on context and the actor viewing the spectra

Selecting ϵ is a trade-off between privacy and utility between participants and the advisory. The advisory wants to maximize the utility of the data and on the contrary, the individual's main concern is to maximize her privacy. Setting an ideal value of privacy budget ϵ is a challenging task so both of these parties (participants and advisory) should agree upon which ranges are acceptable. Ideally, the green dots illustrated in Figure 3.1. Hence, setting ϵ is mainly dependent on the data case under consideration.

3.3.4 Centralized vs Local Setting

Differential privacy can be achieved using different settings and environments where the differential private algorithm is executed. In this section, we will discuss two major differential private environments: the centralized setting and local differential privacy (LDP).

In a centralized differential privacy setting, originally presented in [31], the data from each source is stored in the centralized server before the differential private algorithm is executed, as shown in Figure 3.2. When the analysts query the data, the server or the curator randomizes the query result with a differentially private algorithm and sends the randomized response to protect individual privacy. The number of responses depends on the privacy budget ϵ allocated to the analysts, and once the budget expires the curator stops all responses to protect privacy strength. This budget allocation is related to the composability property of differential privacy. We will discuss composition techniques in section 3.3.5. The centralized setting is also called *Interactive* differential privacy.

In the local differential privacy (LDP) setting, each participant executes a differentially

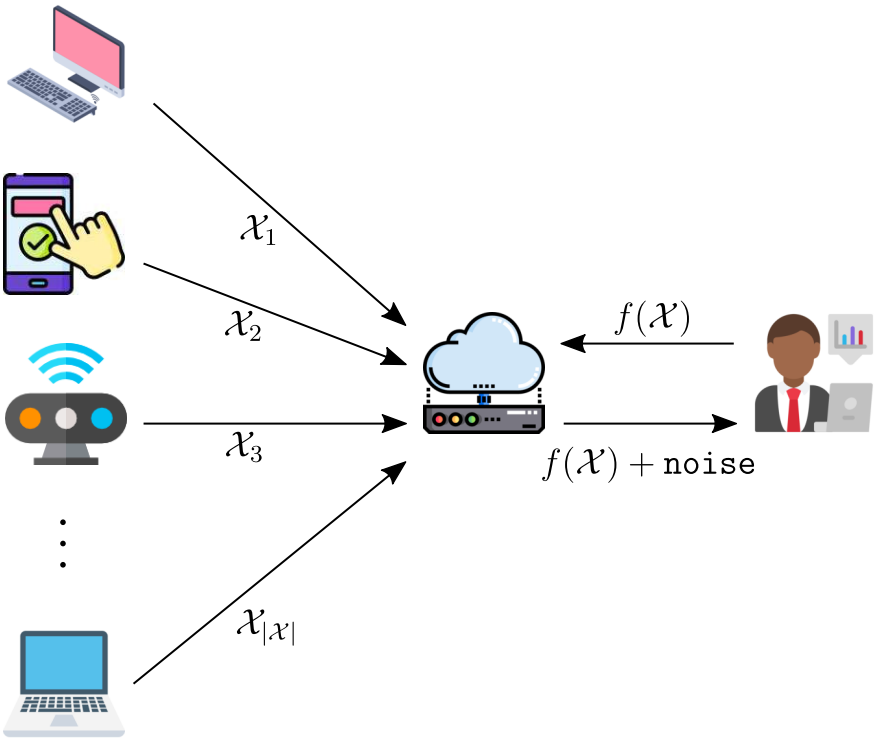


Figure 3.2: Raw data $\mathcal{X}$ from each individual is collected in a centralized server, for each query $f(\mathcal{X})$ on the data is responded under differential privacy ($f(\mathcal{X}) + \text{noise}$)

private algorithm on their data locally before sending the perturbed data to the centralized server. In the LDP model, we do not need to store all the sensitive information at a centralized location. Hence, protecting the data from being potential honeypots for hackers. As such, local models inherently protect against data breaches. The LDP setting is also called *Non-interactive* differential privacy, as illustrated in Figure 3.3.

3.3.5 Composition Techniques

Typically, analysts wish to generate multiple separate statistics on a single dataset. In such a case, each subsequent query might disclose some information about the individuals participating in the dataset. In fact, after multiple responses are returned, the privacy parameter ϵ will necessarily degrade.

Consider computing the same statistic using a randomized mechanism. The average of the responses given by each instance of the mechanism will converge to the true answer. We cannot avoid the fact that the robustness of the privacy guarantee will degrade with the consecutive use of queries. The following theorems (sequential composition, parallel composition, and post-processing) show how the privacy parameter ϵ composes when differentially

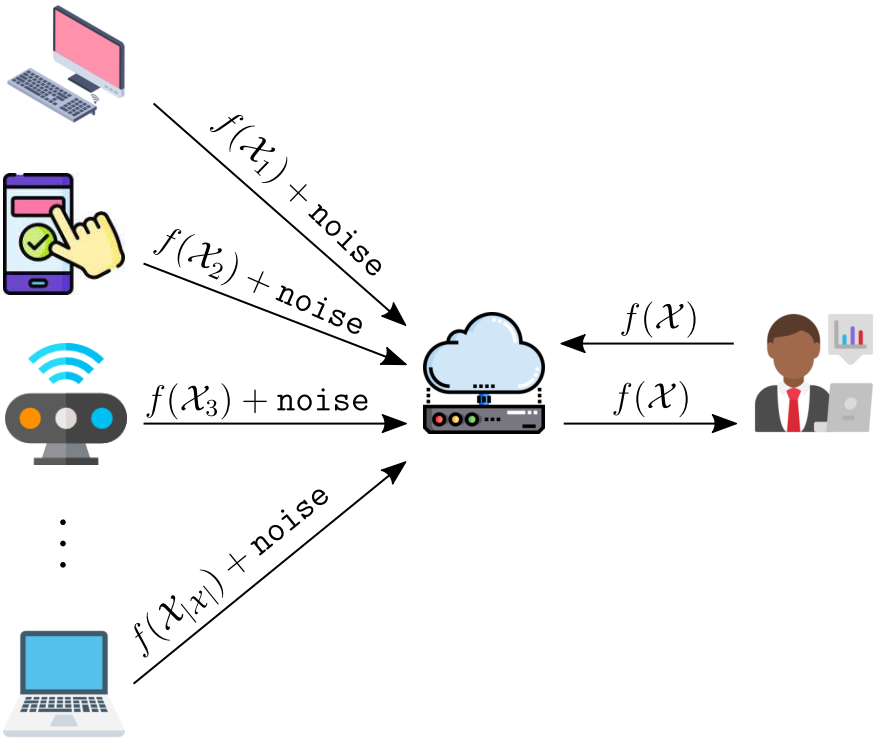


Figure 3.3: Each individual's data $\mathcal{X}$ is randomized under differential privacy algorithm $(f(\mathcal{X}) + \text{noise})$ locally before their data is gathered in a centralized database

private subroutines are combined.

Theorem 3.3.1 (Sequential composition [31])**.** *Let a randomized mechanism $\mathcal{M}_1 : \mathbb{N}^{|\mathcal{X}|} \to \mathcal{R}_1$ be ϵ_1-differentially private, and let $\mathcal{M}_2 : \mathbb{N}^{|\mathcal{X}|} \to \mathcal{R}_2$ be ϵ_2-differentially private algorithm. Then their sequential composition, defined to be $\mathcal{M}_{1,2} : \mathbb{N}^{|\mathcal{X}|} \to \mathcal{R}_1 \times \mathcal{R}_2$ by the mapping: $\mathcal{M}_{1,2} = (\mathcal{M}_1(D), \mathcal{M}_2(D))$ is $(\epsilon_1 + \epsilon_2)$-differentially private.*

The sequential composition can be applied repeatedly to obtain the following lemma:

Lemma 3.3.2. *Let a randomized mechanism $\mathcal{M}_i : \mathbb{N}^{|\mathcal{X}|} \to \mathcal{R}_i$ be ϵ_i-differentially private for $i \in [k]$. Then if $\mathcal{M}_{[k]} : \mathbb{N}^{|\mathcal{X}|} \to \prod_{i=1}^{k} \mathcal{R}_i$ is defined to be $\mathcal{M}_{[k]}(D) = (\mathcal{M}_1(D), \cdots, \mathcal{M}_k(D))$, then $\mathcal{M}_{[k]}$ is $(\sum_{i=1}^{k} \epsilon_i)$-differentially private.*

In situations when the sequence of queries is executed on *non-interactive* databases, we can apply the parallel composition technique. The maximum privacy budget ϵ would provide strong privacy guarantees in this composability. For example, suppose we have d disjoint subsets partitioned from the database D and have d randomization algorithm $\mathcal{M}_d$. Then,

Theorem 3.3.3 (Parallel composition [31])**.** *Let $D \in \mathbb{N}^{|\mathcal{X}|}$ be a dataset and k be a positive integer. For $i \in [k]$, let $\mathcal{X}_i \subset \mathcal{X}$, let a randomization algorithm $\mathcal{M}_i : D \cap \mathcal{X}_i \to R_i$ be a ϵ_i-*

differentially private algorithm executed on $\mathcal{X}_i \subset \mathcal{X}$*, such that* $\forall i, j, \left|\mathcal{X}_i \cap \mathcal{X}_j\right| = \emptyset$ *whenever* $i \neq j$*, and each individual data is contained in exactly one of the* $\mathcal{X}_i's$*. Then the parallel composition of* $\mathcal{M}_1, \cdots, \mathcal{M}_k$ *is* $max\{\epsilon_i : i \in [k]\}$*-differentially private.*

The concept behind parallel composition is to divide the dataset into disjoint chunks and run a differentially private mechanism on each chunk independently. Each individual's data appears in exactly one of the chunks since the chunks are disjoint - so even if there are k disjoint chunks in total, the differentially private mechanism runs exactly once on each individual's data. If $\mathcal{M}(\mathcal{X}_i)$ satisfies $\epsilon-$differential privacy, and we split $\mathcal{X}$ into k disjoint subsets such that $\mathcal{X}_i \cup, \cdots, \cup \mathcal{X}_k = \mathcal{X}$, then the mechanism that releases all the results $\mathcal{M}(\mathcal{X}_i), \cdots, \mathcal{M}(\mathcal{X}_k)$ satisfies $\epsilon-$differential privacy.

3.3.6 Post-Processing

Another useful characteristic of differential privacy is that the output of any differentially private algorithm is also differentially private after "post-processing". The post-processing provided privacy guarantees until and unless it does not "dip" back to the true data.

Theorem 3.3.4 (Post-processing [30])**.** *Let a randomized algorithm* $\mathcal{M} : \mathbb{N}^{|\mathcal{X}|} \to \mathcal{R}$ *be* ϵ*-differentially private, and let* $f : \mathcal{R} \to \mathcal{R}'$ *be an arbitrary deterministic mapping. Then* $f \circ \mathcal{M} : \mathbb{N}^{|\mathcal{X}|} \to \mathcal{R}'$ *is also* ϵ*-differential private.*

3.4 Mechanisms of Differential Privacy

There are various ways to design differentially private algorithms, or techniques can be made differentially private by calibrating them to adhere to definition 3.2.2. It is beyond the scope of this thesis to introduce all differentially private algorithms. Instead, we will introduce three of the most commonly used techniques to achieve differential privacy in this section. First, the technique used in the centralized model includes *Laplace mechanism* and *Exponential mechanism*. Next, *Randomized response* which is typically used in the local setting. Table 3.1 outlines the characteristics of the three most commonly used differential private mechanisms.

Mechanism type	Deployment mode	Response data type
Laplace mechanism	Centralized (Interactive)	Numerical
Randomized response	Local (Non-Interactive)	Categorical
Exponential mechanism	Centralized (Interactive)	Categorical

Table 3.1: Comparison of differentially private mechanisms with their deployment mode and response data type

3.4.1 Laplace Mechanism

The Laplace mechanism is one of the most common and early methods used to achieve differential privacy [31]. A random noise drawn from the Laplace distribution (centered at 0) with scale $\frac{\Delta f}{\epsilon}$ is added to each query response. Each query result is perturbed appropriately using Laplace noise to achieve differential privacy.

Theorem 3.4.1 (Laplace mechanism [31])**.** *Let Δf be the sensitivity of a function $f : \mathbb{N}^{|\mathcal{X}|} \to \mathcal{R}^k$. The Laplace mechanism $\mathcal{M}(D, f(\cdot), \epsilon) = f(D) + (Y_1, \cdots, Y_k)$, where Y_i is i.i.d. random variable drawn from $Lap(\frac{\Delta f}{\epsilon})$.*

3.4.2 Randomized Response

Randomized response proposed by Warner in 1965 in [38] is a data collection technique on sensitive or embarrassing data, where the respondent hesitates to provide a true answer. This technique can be used to inject random noise into the output of a function.

Suppose we are asked to collect a survey: the survey questions might include some embarrassing or illegal behaviors (e.g. do you smoke, or did you file your tax last year). There is a chance that respondents are not truthful and the data curator might end up with the wrong estimations. To solve the problem, the protocol of collection of responses was simply modified: before answering, the respondent is instructed to flip a coin in secret and respond "Yes" if the coin is "Head", otherwise asked to respond truthfully. Randomization response provides deniability for any "Yes", since a "Yes" response might be the outcome of a coin coming up "Head". This strong deniability property can be generalized to both the "Yes" and "No" answers by further modifying the protocol. When the first coin is "Head" the respondent is asked to flip a second coin whose outcome will determine if the answer is "Yes" or "No". In this way, both answers become deniable. Intuitively, the randomized

response permits that *privacy is obtained by the process*. Dwork and colleagues proposed the model of differential privacy [30] in which the mechanism that controls the randomization protocol that produces the output is related to the parameter ϵ: lower the parameter value, better the privacy level. We will show that the above-described randomized response protocol satisfies the properties required by differential privacy and it corresponds to a randomization mechanism that is $(\ln 3, 0)$-differentially private. This level of privacy degrades if the survey is repeated many times by the same respondent. So, in order to maintain a strong privacy guarantee with a high utility we need a better data collection mechanism that preserves privacy as we present in this thesis.

3.4.3 Exponential Mechanism

The exponential mechanism [39] is the natural building block for answering non-numerical queries to preserve differential privacy. In the exponential mechanism, there is a scoring/utility function $u : \mathbb{N}^{|\mathcal{X}|} \times \mathcal{R} \rightarrow \mathcal{R}$ which maps database/output pairs to a utility score. For a database D the user prefers that the exponential mechanism outputs $r \in \mathcal{R}$ with the maximum possible utility score.

Theorem 3.4.2 (The Exponential mechanism [39])**.** *For a fixed database D, choose $r \in \mathcal{R}$ with probability proportional to* $\exp \frac{\epsilon \cdot u(D,r)}{\Delta u}$. *Where $u(D, r)$ is a utility function, ϵ is the privacy parameter, and Δu is the sensitivity of the function given as:*

$$\Delta u = \max_{D_1, D_2 : \|D_1 - D_2\| \leq 1} \max_{r \in \mathcal{R}} \left| u(D_1, r) - u(D_2, r) \right| \tag{3.7}$$

In the exponential mechanism, a randomized algorithm $\mathcal{M}(D, u, \mathcal{R})$ outputs an element $r \in \mathcal{R}$ with probability proportional to:

$$\Pr[\mathcal{M}(D, u, \mathcal{R}) \sim r] = \frac{\exp^{\frac{\epsilon \cdot u(D,r)}{\Delta u}}}{\sum_{r \in \mathcal{R}} \exp^{\frac{\epsilon \cdot u(D,r)}{\Delta u}}} \tag{3.8}$$

Mathematically, from equation 3.8 the probability of outputting r will increase exponentially with the utility function. To ensure ϵ-differential privacy, the maximum change in the utility function u corresponds to the single element change in the sensitivity of the function Δu. In equation 3.8, D is a dataset with discrete attribute values. If D had continuous attribute values, we would have to replace summation with integral.

Complicated privacy-preserving mechanisms have been developed by combining Laplace and Exponential mechanisms. Multiplicative weighted approach for attractively releasing synthetic data [40]. The medium mechanism, which categorizes queries as "easy" or "hard," based on whether a majority of databases consistent with previous answers to hard queries would provide an accurate answer (in which case the user already "knows the answer"). Easy queries are answered by using the corresponding median value, while hard queries are answered in the same way as the Laplace mechanism [41]. The Gaussian mechanism uses Gaussian noise to achieve soft differential privacy, which has also been discussed in the literature besides Laplace and Exponential mechanism [30]. Generalized Gaussian mechanisms for special cases which draw noise from Laplace or Gaussian distributions [42].

3.4.4 Applications of Differential Privacy

In theory, any query response can be made differentially private. However, in practice, some queries are better suited based on the utility and accuracy trade-off. In this section, we will discuss various families of counting queries and differentially private mechanisms to produce the output of these queries.

Counting queries

Counting query is an extremely powerful class of queries. It captures many standard data mining tasks and basic statistics in the database. They are sometimes in the fractional form (fraction of the elements in the database that satisfies property P), sometimes with weights (linear form), and sometimes in more complex form (e.g., apply $l : \mathbb{N}^{|\mathcal{X}|} \rightarrow \{0,1\}$. Formally, a counting query on a tuple from the universe $\mathcal{X} = \{\mathcal{X}_1, \mathcal{X}_2, \cdots, \mathcal{X}_{|\mathcal{X}|}\}$ takes the form $q : \mathcal{X} \rightarrow \{0,1\}$, and applied to a database of tuples returns the `sum` value given by:

$$q(D) = \sum_{\text{tuple } j} q(D_j \cdot \mathcal{X}_i).$$

The sensitivity of the counting query is 1 (adding or removing a single individual in the database will change a count by at most 1). The ϵ-differential privacy can be achieved for counting queries by adding noise scaled to $1/\epsilon$ independent of the size of the database. The noise is drawn from the Laplace distribution $Lap(1/\epsilon)$. The sensitivity of a fixed but arbitrary list of m vector-valued counting queries is m. The ϵ-differential privacy can be

achieved by adding Laplace noise scaled to m/ϵ to the true answer to each query response.

Range/Histogram queries

Histograms provide a complete statistical summary of the database in numerical form. Given a data universe $\mathcal{X} = \{\mathcal{X}_1, \mathcal{X}_2, \cdots, \mathcal{X}_{|\mathcal{X}|}\}$ and $\mathcal{X}_i \in \mathcal{R}$, for each i histogram queries merge neighbouring counts into m bins over the integer domain $[1, |\mathcal{X}|]$. A histogram is valid if and only if all the bins completely cover the domain $[1, |\mathcal{X}|]$ without overlapping. The sensitivity of a histogram query is 1 because the cells in a histogram are disjoint and adding or removing a single individual in the database will change the count by at most 1. The ϵ-differentially private histogram queries can be achieved by adding Laplace noise scaled to $(1/\epsilon)$ to each true cell count.

k-way Marginals

Contingency tables are a particular case of histogram queries. They refer to the projection onto the subset of the attributes in the database. The counts in the table are the marginals; each marginal is associated with a subset of attributes and called k-way marginals when at most $k \leq d$ attributes are used. These tables represent the association between many different and possibly overlapping attributes. The query sensitivity on contingency tables can be either 1 or 2, depending on how you view the removal of a single individual from the dataset. Suppose the individual is removed from the dataset: then the sensitivity is 1 because it will affect only one cell in the contingency table. On the other hand, if the individual changes its attribute value, the maximum difference is 2, one leaving the cell and adding it to another cell.

Our proposed randomization mechanism injects a controlled amount of noise into the contingency tables, which we will discuss in Section 4.4. Contingency tables provide a concise way to represent the conditional dependencies between variables in a Bayesian network and are useful for performing probabilistic inference and updating beliefs based on observed evidence. Additionally, we use the *low-degree* Bayesian network to reduce the combinations in the contingency tables by focusing only on the attribute having higher mutual information, discussed in Section 5.2.3. We also use the contingency tables in the utility-privacy tradeoff of the privacy mechanism and experimentally show the effect of Laplace noise on the dependence structure of the variables in the Bayesian network, covered in Section 7.3

and 7.4.

Categorical queries

Adding numerical noise in query responses containing categorical attributes is not practically possible. In such scenarios, we can use the exponential mechanism to add noise to the query output, as discussed in Section 3.4.3.

Chapter 4

Randomized Response Block Aggregation Protocol

The data collected from smart devices, the Internet of Things (IoT), and Smart Homes can be used for mining purposes and potentially benefit the organization with a large user base, such as Google, Amazon, and Samsung. The data collected from personal devices is intrinsically private and should be collected through a privacy-guaranteed mechanism to ensure privacy breaches. Local differential privacy solves privacy problems by collecting randomized responses from each user, and it does not need to rely on a trusted data aggregator/curator. The curator can still build reliable prediction models on the collected amount of randomized data. Our approach utilizes the randomized response technique in a novel manner: it guarantees privacy to users during the data collection and simultaneously preserves the high utility of the analysis. Our proposed method can be seen as a particular case of synthetic data generation by producing contingency tables (marginals) in a privacy-preserving mechanism. This chapter will describe our randomized response techniques and discuss the motivating applications domains. We will justify why the protocol satisfies the property of differential privacy and utility guarantees theoretically and through experimental analysis.

4.1 Introduction

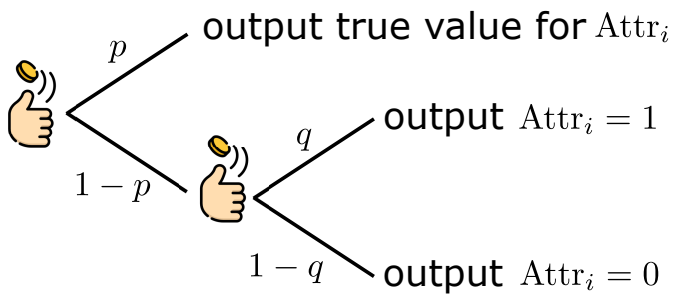
Data collected from smart devices, including mobile phones, home applications, wearables, sensors, and vehicles, have become invaluable assets for product designers and application developers. Companies collect data from end-users and use them to tailor their products

and services, including third-party advertising, developers' advertising or marketing, product personalization, analytics for customer retention, application functionality, and for other multiple purposes. For instance, collecting user data for analytics to assess consumer behavior, such as determining the efficacy of product features, planning new ones, or determining the audience size and characteristics. Data collected from smart devices are used to enable better-informed decisions.

The problem with massive data collection is that collecting sensitive personal data poses a significant security risk to people's privacy rights. To get accurate information from the individuals, the data collection process should enforce robust privacy-preservation mechanisms and consider the collected data's utility. We introduce a novel data collection protocol with randomized responses to achieve data collection with privacy guarantees. The protocol occurs in a non-trusted, third-party data aggregator/curator. Our proposed method provides strong privacy guarantees combined with a high data utility, as this work shows.

Our privacy-preservation randomized response is built on the idea of *randomized response* proposed by Warner in 1965 [38], a data collection technique on sensitive or embarrassing data, where the respondent hesitates to provide a true answer. This technique can be used to inject random noise into the output of a function.

As discussed in Section 3.4.2, surveys generated using randomized responses allow easy computations of correct population statistics while protecting the individual's privacy. The survey respondent is asked to flip two fair coins in secret; if the first coin is "Head", the respondent is asked to flip a second coin whose outcome will determine if the answer is "Yes" or "No". It is simple to see that in a situation where both "Yes" and "No" answers can be denied (flipping two fair coins), the true number of "Yes" answers can be accurately estimated by $2(P - 0.25)$, where P is the proportion of "Yes" responses. This estimation requires absolute adherence to the randomization protocol, which may not hold for human subjects and may even be difficult for algorithmic implementations [43]. A case analysis of the two fair coin flips makes it clear that the respondents will, on average, give the correct response 75% of the time. Importantly, for one-time collection, the aforementioned randomization mechanism provides $ln(3)-$differential privacy guarantee ($ln(0.75/(1-0.75)) = ln(3)$), irrespective of attacker's prior knowledge [31]. This level of privacy degrades if the survey is repeated many times by the same respondent. So, to maintain a strong privacy guarantee with a high utility, we need a better data collection mechanism that preserves privacy,



https://www.overleaf.com/project/5fc2c1ca11bb2d0d9c2ecdc4

Figure 4.1: The flow of the randomized protocol and two flips of the coins, under the assumption that the attribute Att_i is binary.

as we present in this thesis. The contingency table, constructed after the collection of the responses from the population, will be close to the truth if the probability of the successful random event in the protocol (a parameter that we will denote by q) is equal to the probability of the true attribute value associated to that event ($P(\mathrm{Attr}_i = 1)$). This observation motivates the proposal of our protocol. The flow of our randomization protocol is shown in Figure 4.1. Notice that in our proposed protocol, the flow of the operations is slightly different than in the original one proposed by Warner [38]. In our protocol, the true response occurs after the flip of the first coin is "Head" and the randomized one occurs after the flip of the second coin.

In our proposed technique, the natural and more general setting is when each client has multiple attributes, and the server is interested in learning the joint distribution of subsets of these attributes. Knowledge of the joint distribution of subsets opens the way to powerful descriptive and predictive analytical models such as statistical inferential models and Bayesian Networks. They allow the user-analyst to make and test hypotheses on the properties of a population, even on the root causes of observed phenomena, having observed just a sample. In our approach, we do not wish to harm the data owners in the sample, especially in the case they are in a limited number. We adopt an approach of local differential privacy (LDP) rather than the weaker global differential privacy (GDP).

With GDP, one or more aggregators store the actual data and could be a single point of failure and be the goal of attacks. LDP is stronger than GDP because it adds the restriction that even if adversaries had access to the personal responses since the responses are randomized, the adversaries still would be unable to learn about the single users' data. In the proposed protocol, a respondent's private, randomized data is generated after selecting sub-

sets of the attributes. The values of these attribute subsets are communicated in a distributed environment to one or more aggregators. As a final step, the aggregator who receives the randomized data values from the subset of attributes has the task of computation of contingency tables with the frequencies of the observed values of the attribute subsets. The single limitation to generating randomized responses is that the individual respondent does not have to communicate the same attribute in multiple subsets but just in one. If the same attribute were involved in more than one randomized response, the system should remember its emitted value to enforce its consistency among the responses. If that were not the case, the adversaries would quickly determine the true responses from the "fake" ones because the true values would have a higher probability of being observed than the "fake" ones.

Thanks to the protocol properties, we demonstrate that it is possible to reconstruct the true joint probability of the attribute subsets from the possibly noisy values communicated by the individuals using the randomization protocol. The transmitted values do not need to correspond to the true ones for each individual, in virtue of the deniability property of the protocol. Moreover, in our randomized protocol, the randomization is local to the individual users, and there is no need for a different, trusted organization to perform the randomizer or add a verified amount of noise.

Finally, we will show that the data collected at the aggregator provides a high utility value. The tasks of analytics computation and inferential learning require knowledge of the existing dependencies between attributes. Our proposed solution gives guarantees, at certain confidence levels, that the statistical dependencies observed in the reconstructed data correspond to the true ones. The proposed solution relies on a combination of sophisticated machine learning modeling and numerical optimization with hypothesis tests, as we will see in Section 4.6.

Apple uses LDP protocols to collect data on the personal uses of the iOS keyboards [44]. RAPPOR [45] by Google uses a randomized response technique [38] to provide LDP in the Chrome browser. RAPPOR uses Randomized Response as a core functionality to aggregate users' responses, such as yes/no questions, and provides $\epsilon-$LDP guarantees. LDP core idea is to aggregate randomized responses from the users without collecting sensitive personal information. The data collection protocol we propose in this thesis is unlike other differential privacy models [16, 31] that provide robust DP but still collect sensitive information in a data store. Thus they expose data to the risk of attacks and require data protection on single points

of failure. LDP avoids collecting users' sensitive information in the first place, thus ensuring strong privacy guarantees to the users and the aggregator.

4.1.1 The Motivating Application Domain

Our randomized response data aggregation is a general approach for privacy-preserving data collection from distributed devices, which can be used in various contexts. The frequencies of values of the attribute subsets collected by the aggregator might be considered noisy tables, contingency tables, or marginals. These tables are considered as a workhorse for data analysis [46]. The statistics generated by these tables are essential in analyzing the true distribution of data and identifying correlations between different attributes. These tables can also be used for query analysis and answering queries on the data. Many machine learning tasks and inference algorithms must capture accurate correlations from these contingency tables. These algorithms include predictive text modeling [47] and association rule mining for classification, which work on the low-order marginals as a pre-processing step [48]. Direct sampling for multivariate distributions is very costly. These algorithms rely on low-order marginals as a building block and compute accurate approximations by the Maximum Likelihood principle and vine-copulas [49, 50]. Our proposed method generates low-dimensional tables on m attributes (*m-way*). We can generate even lower-dimensional contingency tables from these lower-dimensional ones by further marginalizing the existing ones. We call these further contingency tables higher-level if we show them organized in a lattice, as shown in Figure 4.2. We propose to apply linear programming to the *m-way* contingency tables to make consistent the marginals of the higher level ones (*k-way* contingency tables, with $k<m$). This approach is also followed by [46].

Our proposed method is a perfect candidate for private Bayesian learning. The generated noisy tables can be used to test the validity of the hypothesis of statistical dependence between attributes by subjecting them to statistical hypothesis testing. We can use these tables to answer more complex analyses, such as analyzing the probabilistic relation between various attributes, explaining the causal relationships between variables involved in an observed phenomenon, and modeling the relationships between variables by Bayesian Networks. Bayesian networks solve complex probabilistic dependency problems, whose dimension grows exponentially, exploiting the conditional independence between variables entailed by influence chains. The assumptions of conditional independence and the collec-

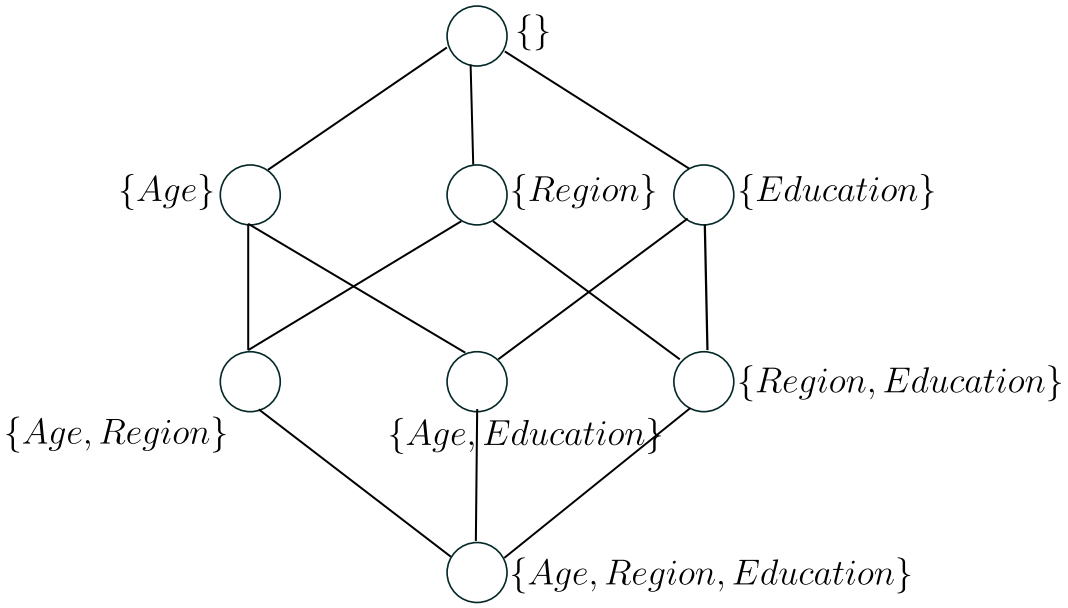


Figure 4.2: Lattice structure of contingency tables on three variables {Age, Region, Education}. As {Age} ⊆ {Region, Education}, contingency table {Age} can be computed either from {Age, Region} or from {Age, Education}

tion of the combinations of values of the sets of variables that are instead dependent permit the probabilistic inference of the interesting variables in an acceptable amount of time. Such probabilistic models are derived from the conditional probability tables (CPTs), which are derived directly from the marginals in contingency tables.

Our proposed technique can also be applied to the private collection of up-to-date crowdsourced statistics. Collecting personal data from clients is considered a violation of end-users privacy. On the other hand, not collecting such data will also be unfavorable to the users. If the operator cannot collect the correct statistic, they cannot improve their products or services and meet the users' expectations. The operator can only collect high-order statistics using privacy-preserving mechanisms to resolve privacy risks. Unfortunately, few existing practical techniques provide, at the same time, both guarantees of maintaining the utility of the data for analysis and protecting the end-users' privacy. Therefore, to reduce the concerns about privacy, these operators rely on pragmatic information processing: for example, removing unique identifiers [13], performing regular deletion in data storage of users' data after a certain time period [51], implementing access-control and auditing policies on data access [52]. However, these approaches have limitations in providing provably-strong privacy guarantees. Our protocol can help such operators to handle these challenges and the potential privacy pitfalls.

4.2 Related Work

Privacy-preserving data statistics are often considered in a centralized setting. The data stored at a central place or the output of a query on the data itself is perturbed by adding a random noise using a random variable from Laplace distribution or applying the Exponential mechanism to the output of the queries. These perturbation techniques reduce the risk for an individual to be identified [30, 53]. However, in the classical approach, with true data in the database, individual privacy is still not guaranteed from external attacks (e.g., by the presence of not secure data centers) or internal adversaries (e.g., eavesdropping). Our approach is based instead on the decentralized setting with local differential privacy. Each client randomizes its own true values using a randomization mechanism (possibly implemented in certified, simple applications for random number generation installed locally). The noisy values are then sent on the network to the aggregator without needing to be protected. Data are then aggregated by the aggregator to produce the desired statistics and thus reducing, even more, the risk of data breaches.

A multitude of approaches exist: they combine randomized response techniques [38] to create sophisticated noise addition mechanisms which are both privacy-guaranteed by DP with an extra assurance that their sensitive information is never visible to data aggregators; still, each user maintains "plausible deniability" of any sensitive information [44, 45, 54, 55, 56]. In randomized response techniques, the users who own a private bit of information flip it with some predefined probabilities to have plausible deniability of their response. Google RAPPOR [45] collects users' data in a private setting, where the responses (i.e., the users' URLs) are mapped to a Bloom filter using a hash function. RAPPOR implements a two-step randomization technique: first, by mapping the user string onto a Bloom filter using a hash function, and second by flipping each bit in the Bloom filter with given probabilities. RAPPOR provides strong privacy against Longitudinal attacks, i.e., if the attacker links several successive answers collected in time from a single user. The privacy of RAPPOR depends on the size of the Bloom filter and on the number of hash functions. This number linearly increases with the expected number of users' strings. For accurate estimations, RAPPOR requires a high sample size and can only be used on the frequency of users' strings whose domain is already known. Besides a simple counting of these strings, a follow-up method proposed in [56] extended the ability of RAPPOR to estimate the frequencies of unknown users' strings that are frequent across the users' distribution. However, these techniques

have some limitations: first, RAPPOR accuracy decreases with the increasing number of attributes; it cannot compute aggregates on numeric attributes (e.g., the average login time of users). PRIVAPPROX [57] uses the same perturbation technique as RAPPOR but requires a smaller sample size and is designed for stream data analytics.

Apple implements privacy in their iOS to collect user statistics through users sketching. This is done to reduce the dimensionality of huge domains [44, 58]. Microsoft collects users' app statistics privately using rounding and memorization techniques [59]. Wang *et al*. [60] proposed an optimization technique to reduce the variance in noisy responses. Their technique utilized the asymmetric randomization response and hashing function to achieve noisy responses. Kairouz *et al*. [61, 62] propose the optimal generalizations of randomized responses to estimate the frequency of a single categorical attribute; they use it to generate a histogram or to calculate the peaks in the inputs (heavy hitters). Their proposed technique can solve the problem of measuring a one-dimensional marginal distribution.

4.3 Preliminaries

We consider a setting where each client owns a set of attributes. The centralized server wishes to collect and store these attributes in a privacy persevering manner to protect the privacy of each client participating in the data collection. The server or curator can then release these privacy-guaranteed attributes as a joint distribution on these attributes. This joint distribution is materialized as a *k-way* contingency table on a subset of k attributes. These joint distributions can be utilized in Bayesian structure learning, statistical analysis, and various machine learning models computation. These privacy-preserving multi-dimensional distributions are frequently implemented using LDP setting in [46, 63, 64, 65] where the server adds noise using Laplace distribution. These tables are then released for further statistical analysis or machine learning model computation.

4.3.1 Notations

We consider a dataset D with d attributes $\mathbf{A} = (A_1, A_2, \cdots, A_d)$. We use a_i to denote a specific value of A_i and $|A_i|$ to denote the attribute cardinality (i.e., distinct values). Each row in the dataset represents a single user or client u (we will use these terms interchangeably)

(a) Dataset consists of 6 attributes (Age, Region, Education, Occupation, Sex, Transportation)

S.No	**A**	**R**	**E**	**O**	**S**	**T**
v^1	adult	big	high	emp	M	car
v^4	adult	big	high	emp	F	train
v^2	adult	small	high	emp	F	other
v^3	adult	small	high	self	F	car
v^5	old	big	uni	emp	F	train
v^6	old	small	uni	self	M	other
v^7	old	small	uni	self	M	train
v^8	old	small	high	self	M	train
v^9	young	big	uni	self	M	car
v^{10}	young	small	high	emp	F	car

(b) Contingency table with 2 attributes T_{AT}

v	$T_{\text{AT}}[v]$
(adult, car)	2
(adult, train)	1
(adult, other)	1
(old, car)	0
(old, train)	3
(old, other)	1
(young, car)	2
(young, train)	0
(young, other)	0

(c) Marginal table for Region

v	$T_{\mathbf{R}}[v]$
(big, $*$)	4
(small,$*$)	6

(d) Marginal table for Age.

v	$T_{\mathbf{A}}[v]$
($*$, adult)	4
($*$,old)	4
($*$,young)	2

Table 4.1: Example of a dataset, contingency table, and the marginals

having one or more values for each attribute [1]. Thus a specific client is represented by a $d-$dimensional vector, denoted by $v = \{v_1, \cdots, v_d\}$ such that $v_i \in Dom(A_i)$ that we denote by $[a_i]$ for each attribute A_i. Given a subset $\mathbb{A} \subseteq \mathbf{A}$ of k attributes, we use $T_{\mathbb{A}}$ to denote a *k-way* contingency table, where $T_{\mathbb{A}} = \{(w_1, w_2, \cdots, w_d) : w_j$ belongs to the cartesian product of $Dom(A_i)$, if $A_i \in \mathbb{A}$, otherwise $w_j = * \}$ to represent all possible combinations of values (denoted as C) of the attributes in $\mathbb{A}$. The symbol $*$ is a convenient notation indicating the corresponding attribute is not present in $\mathbb{A}$.

We can also generate a *k-way* marginal table over $\mathbb{A}$, where each cell in the marginal table is calculated by aggregating over the cell values in $T_{\mathbb{A}}$ that have the same values on the attributes in $\mathbb{A}$. We will also use a symbol $T_{\mathbb{A}}[\cdot]$ to denote every entry in the table, $T_{\mathbb{A}}[w_i]$ for a specific cell value, and $T_{\mathbb{A}}[\theta]$ the sum of all the cells in $T_{\mathbb{A}}$.

Example 4.3.1. *Database D in Table 4.1a has six attributes: Age = {adult, old, young}; Region = {big, small}; Education = {high, uni}; Occupation = {emp, self}; Sex = {M, F}; and Transportation = {car, train, other}, to be aggregated with the* `count` *aggregation function applied to subsets of their values. Table 4.1b shows a contingency table over a set of two attributes. Table 4.1c and Table 4.1d show contingency tables viewed as marginalizations*

[1] In case of more values for the same attribute, the user is represented in the database by multiple rows so that in each row we have atomic values for the attributes.

of a dataset on subsets of the attribute dimensions, creating sets of cells with the associated aggregated measures.

4.4 Randomized Response Block Aggregation

In this section, we will discuss in detail our proposed method **Randomized Response Block Aggregation (RRBA)** for aggregating randomized responses from the clients. This noisy data representation is then published to the server for successive analysis, including hypothesis testing, predictive and probabilistic modeling, and other sophisticated machine learning tasks. Our method is inspired by the randomized response technique [38], which ensures $\epsilon-$LDP.

Before querying the end-users on the clients' devices and collecting their randomized responses, the aggregator selects a set of attributes $\mathbb{A} \subseteq \mathbf{A}$ on which the server wishes to learn some joint distribution. The aggregator generates disjoint subsets of k attributes taken from the original set of d attributes $\mathbb{A}$ to form a certain number of *size-k* tables called *views* $\mathbf{V}$. The subsets $\mathbf{V}$ form separate views on the sample population. The union of these views should be as large as possible, leaving out the smallest set of attributes that cannot be included in any view of cardinality equal to k. On these views, we will query a single client for his/her values by using our randomization protocol.

The aggregator arbitrarily selects a combination of views from the possible ones for querying the single client whose attribute values could be randomized in his/her response: this arbitrary selection that changes for each client provides an extra layer of protection in the randomization protocol. These views privately publish a synopsis of the entire dataset. Successively, we can reconstruct any higher-order marginals from these views. To show how to assign attributes into views, we show an example where $d = 5$ and the attributes are $\{A_1, A_2, \cdots, A_5\}$ with $k = 2$ combinations of distinct attributes per view. This is the list of alternative possibilities for each individual. Each alternative is a view composed of two disjoint sets of attributes of cardinality 2 each.

$$V_1 = \{A_1A_2, A_3A_4\}, V_2 = \{A_2A_3, A_4A_5\}, V_3 = \{A_1A_3, A_2A_5\},$$
$$V_4 = \{A_1A_4, A_3A_5\}, V_5 = \{A_1A_5, A_2A_4\}$$

For the first view V_1, we left out the attribute A_5, for the second one A_1, and so on. Just a single one because it could not be paired with another one without allowing repetition of one attribute in the same view. If the first alternative is selected, the view is formed by the two combinations of attributes $\{A_1A_2\}$ and $\{A_3A_4\}$.

We will use the symbol C to denote a single combination in a view and T_C to denote a contingency table built on the combination C.

Both combinations are considered for the same individual. The attributes in any combination are randomized together, thus keeping intact possible statistical dependencies between them, as shown in Example 4.4.1.

Example 4.4.1. *Consider the survey dataset in Table 4.1a. Suppose the server selects the first view $V_1 = \{AR, EO\}$ for the individual. In that case, the combination of these attributes is represented as C_1 = {(adult, big), (adult, small), (old, big), (old, small), (young, big), (young, small)} and C_2 = {(high, emp), (high, self), (uni, emp), (uni, self)}. The randomized responses are collected on these combinations in the contingency tables T_C using our randomization protocol.*

This step is necessary because the randomization protocol must not generate multiple times randomized values of the same attribute from the same individual. Indeed, if an eavesdropper observed the multiple outcomes of the same attribute, even if in combination with other ones, it would observe with higher probability the true attribute values, thus distinguishing the true from the randomized ones. An alternative solution would be to maintain the value generated for each attribute in the internal memory of the clients' devices. But this solution is not always possible for all devices and would require a large memory size for data sets with many attributes. Another reason this latter solution is not preferable is that the generated values for the joint distributions of the attributes would break the possible statistical dependencies among the attributes. The generated values of the attributes already communicated in other views would be generated in an independent way w.r.t. the attributes in the view, thus spoiling the validity of the reconstructed statistics.

Observe that no attribute in a combination repeats itself in the same view, and any pair of attributes is assigned in at least one view. We can generate any $k-$way contingency table from these views. Since independent noise is added to these views, marginalizing two different contingency tables generated from these views to obtain the same marginals would likely give different results. To make consistent the tables generated from these views, we

perform the constraint optimization technique discussed in Section 4.4.3.

We have two different versions of our protocol. In the first version, the aggregator selects arbitrary combinations from a view $V_i \in \mathbf{V}$. The aggregator sends this combination as a question, such as: "What is your age and Which region do you belong to" or "What is your age, your region, and your gender"). Clients' responses are collected in a randomized mechanism to ensure that either randomly selected responses or true responses are collected by the aggregator. In the second version, we divide the clients into groups called blocks B. We then perform randomized data aggregation in parallel within the blocks. Once all responses are collected, the aggregator moves to the next block until all blocks are executed. Before the next block is processed, the probability distribution used to generate random responses is updated to be closer to the true one. This is done by updating the probability distribution with the responses collected in the previous block.

Example 4.4.2. *Consider the database D in Table 4.1a with six attributes. We show how we assign each attribute into views so that each view covers the maximum number of attribute combinations such that no attribute in a view repeats itself and any pair of attributes is assigned in at least one view.*

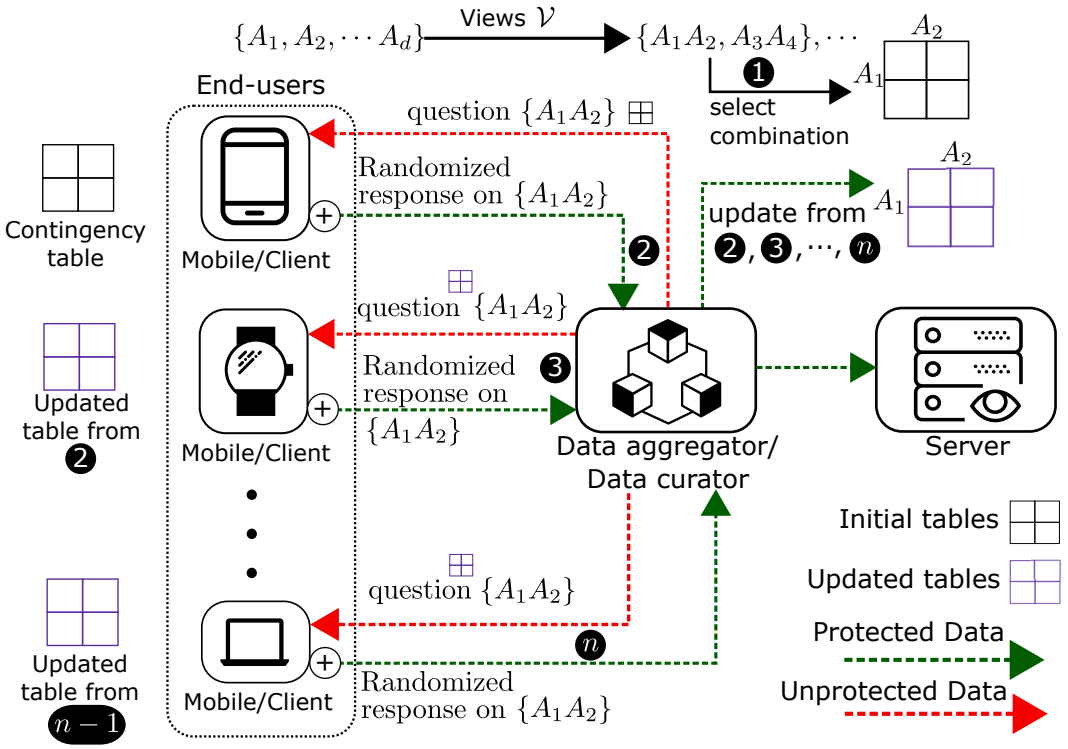
$$V_1 = \{AR, EO, ST\}, V_2 = \{AE, RS, OT\}, V_3 = \{AT, RE, OS\},$$
$$V_4 = \{AO, RT, ES\}, V_5 = \{AS, RO, ET\}$$

Three attributes are included within a single view without repetition. We have a total of five views ($V_1, V_2, \cdots, V_5$) to cover all the possible combinations of attributes.

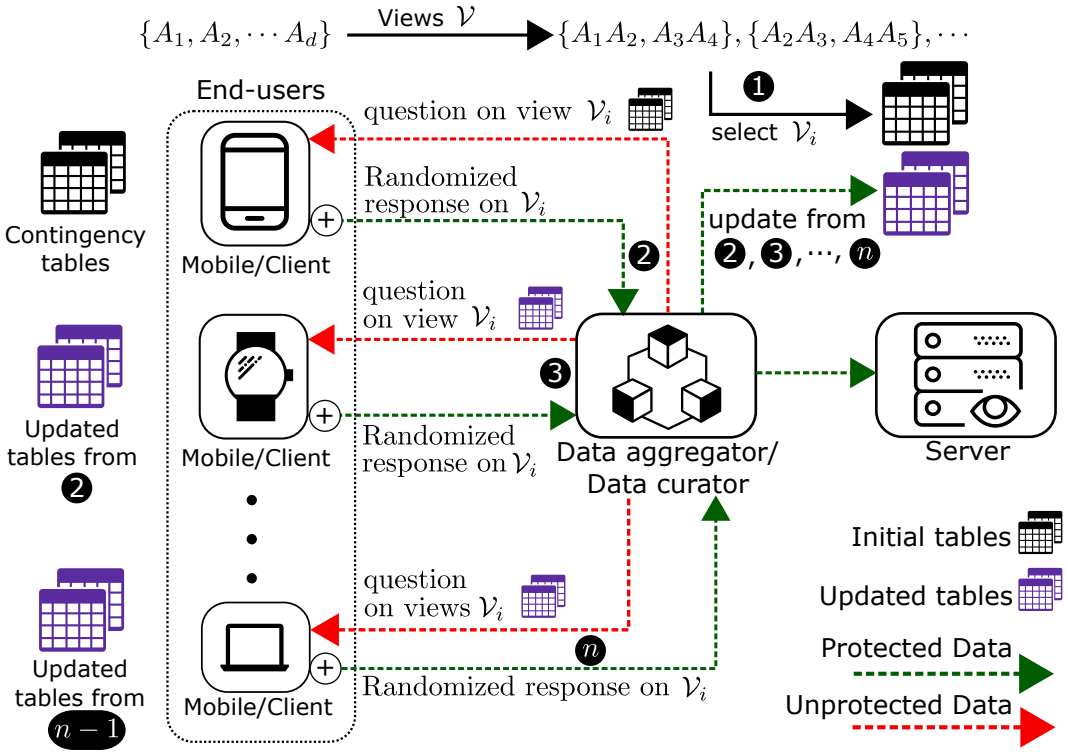
Now suppose we have only five attributes in the abovementioned database D then the formulation of the views is given as:

$$V_1 = \{AR, EO\}, V_2 = \{RE, OS\}, V_3 = \{AE, RS\},$$
$$V_4 = \{AO, ES\}, V_5 = \{AS, RO\}$$

We left out a single attribute in each view because it could not be paired with another without allowing the repetition of one attribute in the same view.



(a) Randomized response on a single view



(b) Randomized response on multiple views

Figure 4.3: Overview of communication between data aggregator and mobile clients to generate noisy contingency table on single or multiple views

4.4.1 Fundamentals of the Randomized Response Block Aggregation Method

Given a set of views V, the aggregator arbitrarily selects a view $V_i \in \mathbf{V}$ comprised of multiple combinations of attributes. On all these combinations of attributes of the selected view, the responses are collected from the client in the $\epsilon-$LDP setting. The aggregator initializes for each combination of attributes in V the joint distribution by a contingency table whose cells values correspond to the uniform distribution, i.e.,

$$T_{C_i} = \frac{1}{|T_{C_i}|} \tag{4.1}$$

To provide an example, for simplicity, suppose we have two attributes such that $C_1 : \{A_1, A_2\} \in V_i$; then $|T_{C_1}|$ is the product of distinct categories in A_1 and A_2, also called respectively the number of rows and columns in a contingency table represented in the classical form of a matrix. The aggregator records this contingency table and sends a copy of T_{C_i} and a query on C_i to the end-users and collects randomized responses. The contingency table T_{C_i} will be sent so that when the user will run the randomized protocol it will be able to generate random values that suitably represent the supposed values distribution of the combination of the attributes.

Example 4.4.3. *Consider database D in Table 4.1a. Suppose V_i contains two combinations of attributes V_i ={AR, EO}. For simplicity we only select one joint combination of attributes, that is C_1 : AR. Then $|C_1| = 3 * 2 = 6$, since attribute $|Dom(A)| = |\{adult, old, young\}| = 3$ and $|Dom(R)| = |\{big, small\}| = 2$. Each cell v_i in the contingency table $T_{C_1}[v_i]$ will be initialized with the uniform distribution, i.e., the value* $1/6 = 0.1666$.

Upon receiving a question from the aggregator on each set of attributes in a view V_i, the client responds according to the outcomes of the random variables, drawn with the predefined probabilities p and q. Probability p is tunable to adjust the privacy and utility of the responses. Probability q is randomly drawn between 0 and 1: it represents the value of the cumulative joint probability function of the attributes values. It makes correspond each combination of attributes values represented in the multivariate contingency table with a probability value that these values are observed. Monte Carlo sampling exploits it to draw first the probability value and then returns the corresponding combination of attribute values.

The random variable p is implemented by drawing a random value, between 0 and 1,

Algorithm 1 Randomized response on single client

Input: Set of attributes **A**, and probability p
Output: Noisy contingency table T_{C_1}

Function `Aggregator(A)`**:**
  make views $V = V_1, V_2, \cdots, V_d$;
  randomly generate the views and check that the combinations of attributes are not repeated in the views ;
  generate uniform distribution in T_{C_i} of all views using equation 4.1 ;
  while *exists a client that has not yet communicated* **do**
    select arbitrary view $V_i \in \mathbf{V}$;
    o $\leftarrow$ $Client$(T$_{C_i}$, query(C_i)) ; `/* Call client procedure */`
    reconstruct T'_{C_i} from o and T_{C_i} using equation 4.3 ;
    update: $T_{C_i} \leftarrow T'_{C_i}$
  end
Function `Client(`T_{C_i}, $query(C_i)$`)`**:**
  Sample a Bernoulli variable **B** ;
  if B = *"Head"* **then**
    Respond true value $w_i \in C_i$
  end
  else
    Respond a fake value using equation 4.2, with a random probability q drawn between [0, 1] ;
  end

uniformly distributed, and comparing it to the threshold p. This random variable controls if the user communicates the true values of the combination of attributes. Instead, if the random value is above p, ”fake” values are communicated to the aggregator, according to the second random variable q, drawn between $[0, 1]$. The outcome of this latter random variable corresponds to one of the cells (denoted by w_i) in the contingency table by their probability. In turn, each cell corresponds to some combinations of the categories of the attributes. The variable q for emitting a ”fake” value is a type of Monte Carlo sampling from the given discrete joint distribution T_{C_i}, such that:

$$T_{C_1}[\theta] = 1$$

and $0 \leq q \leq 1$ then

$$\sum_{i=1}^{l-1} T_{C_i}[w_i] \leq q < \sum_{i=1}^{l} T_{C_i}[w_i] \tag{4.2}$$

This "fake" response is emitted in such a way to disclose a *"controlled"* amount of information about the client's true attribute values. Hence, limiting the aggregator's ability to

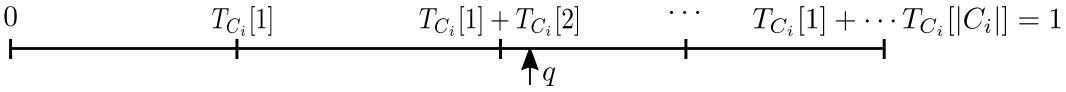


Figure 4.4: The probability intervals of the cumulative probability distribution function that makes correspond each probability interval with a cell w_i of the contingency table T_{C_i}

learn with confidence the true values of the client, Monte Carlo sampling improves the utility of our protocol by emitting combinations of values based on their probability as stored in the contingency table.

Once the aggregator receives a response from the client, it reconstructs a noisy contingency table T'_{C_i} using the contingency table from the previous client T_{C_i} by equation 4.3

$$T'_{C_i}[w_i] = \left(\frac{o_i}{n} - T_{C_i}[w_i] \cdot (1-p)\right) \cdot \frac{1}{p} \tag{4.3}$$

where o_i is the number of clients who communicated those attributes values represented by combination w_i and n is the total number of clients (if the response is from a single client, then $n = 1$). The above equation is justified by the fact that o_i is the number of observed responses corresponding to the same cell i in contingency table $T_{C_i}[w_i]$ and the responses come from the execution of the randomization protocol: they are outcomes of the true probability distribution with probability p (the first coin gives "Head") and are random outcomes controlled by the probability distribution in T_{C_i} with probability $(1-p)$ (the first coin gives "Tail").

The aggregator updates its table $T_{C_i} = T'_{C_i}$ and sends this updated table T_{C_i} to the next client u_{i+1} for next randomized response. Client u_{i+1} now uses the updated probabilities T_{C_i} in the Monte Carlo sampling.

At some point during the protocol execution, it could occur that the aggregator observed a negative cell value in T'_{c_i} due to the application of equation 4.3. This might occur because of the assumption of uniform distribution in initialization, which is too different from the true one. If that situation occurred, the protocol replaces it with a positive, very low probability (representing the event with a negligible probability). It reconstructs the probabilities of the cells w_i of the contingency table T'_{C_i} using Equation 4.4

$$T'_{C_i}[w_i] = \frac{T'_{C_i}[w_i] - \min(T'_{C_i}[\cdot])}{\max(T'_{C_i}[\cdot]) - \min(T'_{C_i}[\cdot])} \tag{4.4}$$

in which $\frac{o_i}{n}$ is the fraction of the total number of responses n in which there is an observed

number o_i of responses with the attribute values corresponding to the single cell w_i of the contingency table T_{C_i} and p is the probability that the client answers with a true response in our randomized response protocol.

Using Equation 4.3, an aggregator reconstructs the unknown probability values of the attribute combination in C_i by accessing the perturbed table T'_{C_i} observed by the responses. In Equation 4.3, we concentrate on a single cell value in the perturbed table. The intuition is that out of the total responses n, the expected number of responses that get perturbed is $T'_{C_i}[w_i](1-p)$. The total number of responses observed in T'_{C_i}, o_i, can be seen as the sum of those responses that were perturbed and those that were unperturbed. Subtracting the $T'_{C_i}[w_i](1-p)$ perturbed responses from o_i, we get an estimate of the number of unperturbed responses. This is scaled up by $1/p$ to get the total number of responses, as only a p fraction of responses were retained. We divided o_i with n in Equation 4.3 to get the empirical probability estimation of the observation of the attribute combination of w_i.

The aggregator aggregates all the responses in T_{C_i} and publishes this noisy table to the server for statistical analysis. Since the aggregator aggregates the noisy reconstructed values T'_{C_i} from Equation 4.3 into T_{C_i}, so this T_{C_i} is also a noisy contingency table. At any point, the aggregator can reconstruct the noisy counts in T_{C_i} using Equation 4.5.

$$T_{C_i}[\cdot] = T_{C_i}[\cdot] \cdot n \tag{4.5}$$

Observe that the aggregator has no access to the client's true values. Thus our mechanism ensures local differential privacy. Algorithm 1 outlines the complete working of our protocol, including both client-side and aggregator procedures.

If the frequency of the attribute values is strongly unbalanced, this first version of the protocol does not always permit the correct reconstruction of the probabilities. The reason is that the uniform probability used to generate "fake data" too strongly impacts the observed data. To resolve this problem, we introduce our second version of the protocol.

The improved version of the protocol

The second improved version of our randomized response data aggregation works similarly to the first version, except now, we divide the number of clients into groups called blocks B. The aggregator now executes the collection of responses from each client in parallel within the blocks. The aggregator aggregates the responses from the blocks and updates the contin-

gency table using equation 4.3, where now n is the block size or the number of clients who responded within that block. When all the responses are collected, the aggregator publishes the noisy contingency table T_{C_i} to the server. The overview of our proposed randomized responses protocol and the communication between the aggregator and its end-users is shown in Figure 4.3, where Figure 4.3a shows how to collect responses on a single combination of attributes $\{A_1, A_2\}$ and Figure 4.3b outlines the collection of responses from clients on multiple combinations within a view V. The block size n is defined by the data aggregator/curator. In Section 4.5, we will demonstrate, with experiments, the selection of optimal block size, which will make converge the estimated probabilities in the contingency tables to true probabilities. The impact of the uniform distribution assumption in the initialization in the second version of the protocol influences only the first block. In contrast, the remaining ones get the contingency table initialization from the result of the work of the previous blocks.

4.4.2 Differential Privacy of Randomized Response Block Aggregation

Our proposed mechanism aims to minimize the risk of disclosure to ensure a strong privacy guarantee while satisfying the strict concept of $\epsilon-$LDP. It promises strong privacy despite the amount of background knowledge of an adversary. Hence, with a substantial amount of auxiliary information, an adversary could not confidently identify the true responses from the clients. Since a single report from the client contributes to the count measure of a single cell w_i in T_{C_i}, the privacy level ϵ is independent of the number of cells in T_{C_i}. Hence, we need to prove the satisfaction of $\epsilon-$differential privacy for only a single contingency table cell.

Theorem 4.4.1. *The proposed randomized response protocol satisfies $\epsilon-$ differential privacy, with:*

$\epsilon \geq ln\left(\frac{1}{1-p}\right)$

where p is the probability that the first coin gives "Head," and the client responds with the true answer.

Proof. Let us consider the contingency tables $T_C^1 \in \mathbb{N}^{|\mathcal{X}|}$ and $T_C^2 \in \mathbb{N}^{|\mathcal{X}|}$ that come respectively from two databases D_1 and D_2 that differ for a single record. Let w_i be the reported combination of attribute values returned by the proposed randomization protocol from the record u_i that differ in the two databases. It corresponds to the cell of the contingency table

$T_C[w_i]$. According to the definition of differential privacy [30] we need to consider when the proposed randomization protocol works as a randomized mechanism and transforms the input databases D_1 and D_2 into the same contingency table T_C, regardless of having in input the database D_1 or D_2. Let us assume that q is the probability that w_i occurs in a database record. According to the usage of the proposed randomized protocol, w_i is the reported attribute value if the first coin draws "Head" and if w_i is the true value: this occurs with probability pq. In addition, the first coin could give instead "Tail", but the value w_i is drawn as a consequence of the second random event: this overall event occurs with probability $(1 - p)q$. On the other database, with a different record u_i', the only possibility that the randomized protocol returns the value w_i is that the first coin gives "Tail" and the second random event returns the value w_i, and this occurs with probability $(1-p)q$. Mathematically, we obtain:

$$\begin{aligned}
\frac{P[\mathcal{M}(D_1) = T_C]}{P[\mathcal{M}(D_2) = T_C]} &\leq \exp^{\epsilon} \\
\frac{P[\mathcal{M}(u_i) = w_i]}{P[\mathcal{M}(u_i') = w_i]} &\leq \exp^{\epsilon} \\
\frac{pq + (1-p)q}{(1-p)q} &\leq \exp^{\epsilon} \\
\frac{q(p+1-p)}{q(1-p)} &\leq \exp^{\epsilon} \\
\frac{1}{(1-p)} &\leq \exp^{\epsilon}
\end{aligned}$$

$$\Rightarrow \epsilon \geq ln\left(\frac{1}{1-p}\right) \tag{4.6}$$

From the opposite side, when D_1 does not contain row u_i but D_2 does, we obtain:

$$\begin{aligned}
\frac{P[\mathcal{M}(D_2) = T_C]}{P[\mathcal{M}(D_1) = T_C]} &\leq \exp^{\epsilon} \\
\frac{P[\mathcal{M}(u_i') = w_i]}{P[\mathcal{M}(u_i) = w_i]} &\leq \exp^{\epsilon} \\
\frac{(1-p)q}{pq + (1-p)q} &\leq \exp^{\epsilon} \\
\frac{q(1-p)}{q(p+1-p)} &\leq \exp^{\epsilon} \\
(1-p) &\leq \exp^{\epsilon} \\
\epsilon &\geq ln\,(1-p)
\end{aligned}$$

that is always satisfied with $0 \geq p \geq 1$. ■

The equation 4.6 shows the relationship between the parameter ϵ (the privacy budget that controls the amount of privacy preserved) and the parameter p of the randomized response protocol (the fraction of times clients respond trustfully). Notice that it does not depend on q, the probability of the emitted value; thus, it is valid regardless of the response.

Decreasing p makes ϵ arbitrarily low, the desired situation since it allows the randomized protocol to make stronger privacy preservation. As a drawback, with low p the convergence of the reconstruction of the true probability distribution from the observed responses becomes slower, as we will see from the experimental results. On the opposite side, as p grows, it grows the risk that true values are emitted too frequently, and ϵ cannot be reduced to small values.

4.4.3 Consistency between Noisy Tables

Given a set of noisy views, the server wishes to release marginals of some attributes with a privacy guarantee. Since independent noise is added in each attribute combination within a view, aggregating marginals from the different views will create inconsistencies in the marginals of the common attributes.

Suppose we have $T_{\mathbb{A}}$, where $\mathbb{A}' \subseteq \mathbb{A} \subseteq \mathbf{A}$ are subsets of the attributes. We use the symbol $\mathbf{T}_{\mathbb{A}' \leftarrow \mathbb{A}}[w]$ to denote the marginal over $\mathbb{A}'$ calculated from $T_{\mathbb{A}}$ by aggregating the corresponding entries.

Consistency between views. We consider the marginal contingency tables $\mathbf{T}^1_{\mathbb{A}}$ and $\mathbf{T}^2_{\mathbb{A}}$ with A coming from two noisy views $A \in V_i$ and $A \in V_j$. The two marginal contingency

tables $\mathbf{T}^1_{\mathbb{A}}$ and $\mathbf{T}^2_{\mathbb{A}}$ are consistent if and only if the marginal table over the common attributes in $V_i \cap V_j$ reconstructed from view V_i is the same as reconstructed from view V_j, that is $\mathbf{T}^1_{[\mathbb{V}_i \cap \mathbb{V}_j] \leftarrow V_i} = \mathbf{T}^2_{[\mathbb{V}_i \cap \mathbb{V}_j] \leftarrow V_j}$.

Given a set of views in $\mathbf{V}$ and set of attributes $\mathbb{A}$, we can compute k-way marginals $\mathbf{T}_{\mathbb{A}}$. When at least one view $V_i \in \mathbf{V}$ includes all the attributes in $\mathbb{A}$, i.e $\mathbb{A} \subseteq V_i$, we can reconstruct $\mathbf{T}_{\mathbb{A}}$ by summing over the corresponding entries of $T_{\mathbb{A}}$ in $T_{\mathbb{V}_i}$, that is using $\mathbf{T}_{\mathbb{A} \leftarrow V_i}$. However, when we have multiple views V_i such that $\mathbb{A} \subseteq V_i$, we need to perform a linear optimization technique to return consistent marginals from all the views V_i that cover all the attributes in $\mathbb{A}$. When $\mathbb{A} \cap V_i$ contains j attributes, then $T_{\mathbb{A}_i}$ provides exactly 2^j constraints on the cells for $T_{\mathbb{A}}$. We can extract all these linear constraints from all the views to generate an under-specified system of equations.

One can utilize the ℓ_1-norm optimization technique discussed in [46] to reconstruct the marginals in $T_{\mathbb{A}}$. This technique does not create a unique solution, and linear programming has no preference among different solutions. So we employ another constraint optimization technique ℓ_2-norm (least square solution). We will follow the quadratic programming approach similar to the work in [66] to solve the under-specified system of equations as a minimizing problem:

$$\min_{v} \quad \sum_{v \in T_{\mathbb{A}}} \mathbf{T}_{\mathbb{A}}[w]^2$$

$$s.t., \quad \forall_{w \in \mathbf{T}_{\mathbb{A}}} \mathbf{T}_{\mathbb{A}[w]} \geqslant 0$$

$$\forall_{V_i \in \mathbf{V}} \; \forall_{w' \in V_i \cap \mathbb{A}} T_{\mathbb{A}}[w'] = \mathbf{T}_{\mathbb{A}}[w']$$

It has been shown that this is a quadratic optimization problem, and we solved it with convex optimization approaches [67].

Example 4.4.4. *We use the example in Table 4.2 to illustrate the consistency procedure between noisy contingency tables* $T^1_{\mathbb{A}} = \{A_1, A_2\}$ *and* $T^2_{\mathbb{A}} = \{A_1, A_3\}$. *After the optimization procedure, the marginal tables* $\mathbf{T}^1_{\mathbb{A}}$ *and* $\mathbf{T}^2_{\mathbb{A}}$ *agree on attribute* A_1 *without changing the marginalization of attributes not involved in the consistency procedure, i.e.,* A_2 *and* A_3.

4.5 Convergence and block size estimation

In this section, we show that the probabilities generated from $T_{\mathbb{A}}[\cdot]$ will converge to the true probabilities after we have used the protocol aggregating the observations sent from the indi-

$T^1_{\mathbb{A}} = \{A_1, A_2\}$ before consistency	0.1	0.4	0.3	0.2
$T^2_{\mathbb{A}} = \{A_1, A_3\}$ before consistency	0.2	0.4	0.3	0.1
$\mathbf{T}^1_{\mathbb{A}}$ marginalized on A_1	0.5		0.5	
$\mathbf{T}^2_{\mathbb{A}}$ marginalized on A_1	0.6		0.4	
$T^1_{\mathbb{A}} = \{A_1, A_2\}$ after consistency procedure	0.175	0.325	0.225	0.275
$T^2_{\mathbb{A}} = \{A_1, A_3\}$ after consistency procedure	0.375	0.125	0.125	0.375
$\mathbf{T}^1_{\mathbb{A}}$ consistent marginal on A_1	0.5		0.5	
$\mathbf{T}^2_{\mathbb{A}}$ consistent marginal on A_1	0.5		0.5	

Table 4.2: Consistency procedure on the marginalization of attribute A_1 using noisy views V_1 and V_3

viduals in a certain number of blocks of size n. The probability $T_{\mathbb{A}}[w_i]^{B_k}$ is the probability of a cell of the contingency table $T_{\mathbb{A}}$ created by running the randomized protocol on the users of block B_k, where we use the superscript B_k to denote the block number. The estimated probability at round k sending outcomes from block B_k is done with $T_{\mathbb{A}}[w_i]^{B_k}$. The estimation of the probabilities, done by the protocol, converges to the true probabilities by oscillating around the true value within a tolerance interval related to the error in observing a Bernoulli variable. The tolerance interval is given by the width of the confidence interval of the Bernoulli variable, with the success probability equal to the true but unknown value v_i and interval width opportunely estimated as follows.

If the approximation of the Bernoulli distribution with the Normal distribution holds (i.e., if $v_i > 5$, with $v_i = T_{\mathbb{A}}[w_i]$ and v_i/n the probability estimation), we can use a symmetrical interval, where the confidence interval size can be estimated by $2z_{1-\alpha/2\sigma}$ with $\sigma = \sqrt{\frac{(v_i/n \cdot (1-v_i/n))}{n}}$ the standard deviation of the Bernoulli distribution. Otherwise, maximum likelihood confidence intervals must be used with the log odds. We set the α confidence level equal to the standard values, e.g., 0.05 or 0.01. This latter means that the estimated probability will remain within the confidence interval with a probability equal to $1 - \alpha$.

The convergence algorithm proceeds as follows:

1. **Initialization with $k = 0$:** $T_{\mathbb{A}}[w_i]^{B_0} = \frac{1}{|T_{\mathbb{A}}|}$
2. **At iteration $k = k + 1$:** run the RRBA protocol and estimate $T_{\mathbb{A}}[w_i]^{B_k}$ from equation 4.3
3. **Repeat:** step 2 until convergence, i.e. $\left|T_{\mathbb{A}}[w_i]^{B_k} - T_{\mathbb{A}}[w_i]^{B_{k-1}}\right| < \delta^*$, for some $\delta^* > 0$

4. **Return:**
   $T_{\mathbb{A}}[w_i] = \frac{T_{\mathbb{A}}[w_i]^{B_k} + T_{\mathbb{A}}[w_i]^{B_{k-1}}}{2}$
   which is the average between the two consecutive observed values in consecutive blocks.

where δ^* is the size of the confidence interval.

4.5.1 Relationship between the privacy budget ϵ and the probability p

This section discusses the relationship between the privacy budget ϵ, and the value of p, i.e., the probability of the first coin is head. We want to evaluate the effect of varying values of p on the convergence of the observed probabilities. In general, the relationship between ϵ and p is that as ϵ increases, p can also increase, as shown in Figure 4.5.

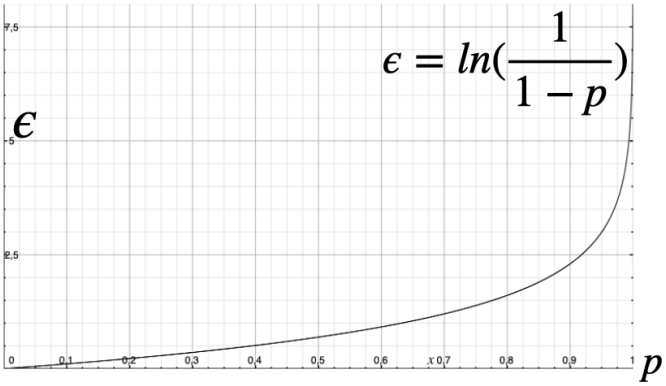


Figure 4.5: Graph of the relationship between the protocol parameter p of the first coin "Head" and the privacy budget ϵ

The above relationship means that as more information can be released (i.e., less privacy is desired and higher values of the privacy budget ϵ), the chance of incorrect data being released decreases (by emitting fake responses with decreasing probability $1-p$). Conversely, as the privacy budget ϵ decreases, p must also decrease, meaning that as less information can be released (i.e., more privacy is desired), the chance of incorrect data being released increases (with probability $1-p$). Unfortunately, as we can observe from Figure 4.5, ϵ as a function of p does not grow linearly with p but exponentially. This means that if we increase p to reconstruct the correct values of the probabilities faster and reach convergence faster, ϵ increases even faster, thus obliging us to add less noise to the output data and reduce privacy.

We drive the values of p using the Equation 4.7 and identify at what value of p we see convergence in the observed probabilities using a given block size in the protocol. In the ex-

periments of Section 4.7.2 we discussed the effect of different values of p on the convergence at different block sizes.

$$e^{\epsilon} \geq \frac{1}{1-p} \Rightarrow \epsilon \geq \ln(\frac{1}{1-p}) \Rightarrow p \leq 1 - e^{-\epsilon} \tag{4.7}$$

4.6 Testing for Association

One of the first questions posed while dealing with two or more categorical attributes is whether they are independent. Two attributes are independent when their joint distribution is equal to the product of their marginals, i.e.

$$P(A_1, A_2) = P(A_1|A_2)P(A_2) = P(A_2|A_1)P(A_1) = P(A_1)P(A_2).$$

In other words, knowing the value of an attribute A_1 provides no valuable information when predicting A_2. The test of independence χ^2 [68] is one of the most common statistical tests when dealing with two or more categorical attributes. We now consider the problem of testing whether two or more categorical attributes in the noisy contingency table $T_{\mathbb{A}}$ are independent of each other. This is important if one wants to employ the reconstructed probabilities in advanced models (e.g., Bayesian networks). Before discussing the test of independence on noisy attributes, we first discuss how independence testing is performed on categorical attributes using χ^2.

We now consider two categorical attributes, A_1, and A_2. They are independent given the null hypothesis $H_0 : A_1 \perp A_2$. One approach to performing a test of independence H_0 is to sample the database (or the users' answers) and obtain n joint outcomes from A_1 and A_2. From the sample, we count the number of observed outcomes as $\mathbb{T}_{A_1,A_2}$ which is the number of times $A_1 = v_i$ and $A_2 = v_j$ occur in the n trials; so we can aggregate all the joint outcomes as a contingency table $\mathbb{T}_{i,j}$. Compute the $\widehat{\chi}^2$ statistic as:

$$\widehat{\chi}^2 = \sum_{i,j} \frac{(\mathbb{T}_{i,j} - \widehat{m}_{i,j})^2}{\widehat{m}_{i,j}} \tag{4.8}$$

Where $\mathbb{T}_{i,j}$ stands for observed cell count and $\widehat{m}_{i,j}$ for the *maximum likelihood estimator* (MLE). Further, we can write MLE $\widehat{m}$ as described in [69] as:

Lemma 4.6.1. *Given a contingency table $\mathbb{T}$ which stores the outcomes of n samples on A_1*

and A_2 if $A_1 \perp A_2$, then the MLE for $\widehat{m}$ is given as:

$$\widehat{m}_{i,j} = \frac{(\mathbb{T}_{i,\cdot})(\mathbb{T}_{\cdot,j})}{\mathbb{T}_{\cdot,\cdot}} \quad \text{for } i \in [r], j \in [c] \tag{4.9}$$

$$\textit{where} \quad \mathbb{T}_{i,\cdot} = \sum_{j=1}^{r} \mathbb{T}_{i,j} \quad \mathbb{T}_{\cdot,j} = \sum_{i=1}^{c} \mathbb{T}_{i,j} \quad \mathbb{T}_{\cdot,\cdot} = \sum_{i,j}^{r,c} T_{i,j}$$

To perform a test of independence, we first calculate **MLE** by $\widehat{m}_{i,j}$ from equation 4.9. Compute $\widehat{\chi}^2$ using equation 4.8 and set $(r-1)(c-1)$ for the degrees of freedom of the test. If $\widehat{\chi}^2 > \chi^2_{(r-1)(c-1),(1-\alpha)}$ (for a $1-\alpha$ significance test) and all entries in $\mathbb{T}$ are greater than 5 we Reject H_0 else Accept H_0.

To perform a similar test of independence for a noisy version of the table, we need to determine an estimation for $\widehat{m}$ where we do not have access to the true cell counts in the contingency table. Now suppose we only have access to the noisy cell values in $T_{\mathbb{A}}$, where noise is added in each cell independently, for instance, using our randomization protocol. To find the best estimates for $\widehat{m}$ given the noisy cells, we consider the total likelihood of the noisy $r \times c$ contingency table. We will perform a two-step MLE calculation similar to the work of [70, 71].

In a two-step MLE procedure, we first find the most likely contingency table $\widehat{T}_{\mathbb{A}}$ given the noisy table $T_{\mathbb{A}}$ and in the second step, we use equation 4.9 to calculate MLE given a table of counts $\widehat{T}_{\mathbb{A}}$. For the first step, we need to minimize $\left\| T_{\mathbb{A}} - \widehat{T}_{\mathbb{A}} \right\|$ subject to $\widehat{T}_{\mathbb{A}}[\theta] = n$ and $\widehat{T}_{\mathbb{A}}[\cdot] \geqslant 0$. Note that if we add independent noise in each cell of a table $T_{\mathbb{A}}$, the above optimization problem gives multiple solutions, and it is not clear which solution to use. The ℓ_1 norm in our objective function in Equation 4.10 is convex but not strongly convex, which means its solution is optimal but may not be unique and sensitive to an initial guess. To overcome this problem, we add a strongly convex function in the objective function:

$$\begin{aligned} \underset{\widehat{T}_{\mathbb{A}}}{\text{minimize}} \quad & \gamma \left\| T_{\mathbb{A}} - \widehat{T}_{\mathbb{A}} \right\|_1 + (1-\gamma) \left\| T_{\mathbb{A}} - \widehat{T}_{\mathbb{A}} \right\|_2^2 \\ \text{subject to} \quad & \widehat{T}_{\mathbb{A}}[\theta] = n, \\ & \widehat{T}_{\mathbb{A}}[\cdot] \geqslant 0. \end{aligned} \tag{4.10}$$

where γ is a mixing parameter in the range $[0, 1]$. The above objective function is in the form of *elastic net regularize* [72] function proposed by [71]. The solution of this objective function will converge to the solution provided by the ℓ_1 norm when γ is sufficiently large.

For the test of independence, in the two-step **MLE** calculation, if any cell value in $\widehat{T}_{\mathbb{A}} < 5$, we follow the commonly chosen rule of thumb to Accept H_0.

4.7 Experiments

For experimental reproducibility, we use three publicly available datasets: *Survey* [73], *Alarm* [74], and *Child* [75]. They vary in the number of instances and attributes as described in the overview of Table 4.3. All attributes are discrete.

Datasets	Records	Attributes	Categories
Survey	500	6	14
Alarm	10,000	37	103
Child	10,000	20	60

Table 4.3: Summary of the selected datasets.

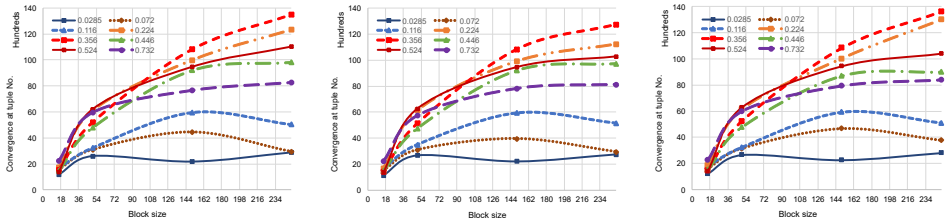
4.7.1 Monte Carlo simulation: Convergence of the randomization protocol

To perform a test of convergence of the second version of the proposed randomized response protocol, we test with any of the values of the attributes v_i whose probability of occurrence is in $\{0.0285, 0.072, 0.116, 0.224, 0.356, 0.446, 0.524, \text{ and } 0.732\}$ and let vary the block size $s = \{18,\ 50,\ 150,\ \text{and}\ 250\}$. We perform 40 trials on 200 blocks on each probability value and block size. We average the number of tuples emitted when the condition holds $\left|T_{\mathbb{A}}[w_i]^{B_k} - T_{\mathbb{A}}[w_i]^{B_{k-1}}\right| < \delta^*$, and remains valid throughout the blocks.

4.7.2 Convergence Results

We perform the test of convergence in the three different datasets (`Survey`, `Alarm`, and `Child`). We plotted the results of these experiments in Figure 4.6, where the x-axis represents the block size, and the y-axis shows the number of tuples emitted when the convergence is reached. The behavior of convergence of our method is almost similar in all three datasets.

From the graph, it is clear that a smaller block size allows more easily to reach early convergence of the proposed technique, both in lower and higher probabilities. Hence, from the experiments, it is sufficient to have a block size equal to the dimension of the contingency table.



(a) Survey dataset (b) Alarm dataset (c) Child dataset

Figure 4.6: Convergence in probabilities $P(A_i = w_i) = \{0.072, 0.116, 0.224, 0.356, 0.446, 0.524, 0.732\}$ and block size $s = \{18, 50, 150, 250\}$ on three different datasets, *Survey, Alarm* and *Child*.

We perform similar experiments on convergence utilizing different values of p (the probability of releasing true data due to the first coin "Head" in the randomization protocol). Due to the computational limitations, we focused on a few probabilities to analyze convergence on the varying value of p. The selected probabilities of the true attribute values $P(A_i = w_i) = \{0.072, 0.116, 0.356, 0.446\}$, block size $s = \{18, 50, 150, 250\}$ and the probability of the first coin "Head" $p = \{0.009, 0.048, 0.095, 0.139, 0.221\}$.

At $p = \{0.009, 0.0480, 0.095\}$ none of the processes of reconstruction of the probabilities $P(A_i = w_i)$ converge at given block sizes s. Instead, at $p = 0.1390$, the reconstruction process of the higher probabilities $P(A_i = w_i)$ set at 0.356 and 0.446 converge with the higher block sizes, i.e., 150 and 250. At $p = 0.221$ the reconstruction process of all the probabilities converges with the higher block sizes, as shown in the graph of Figure 4.7. In the graph, there is no convergence of all the probabilities with block sizes 18 and 50. If we increase the block size, the reconstruction processes of all the probabilities $P(A_i = w_i)$ start to converge. A similar behavior is observed at $p = 0.295$. The results of Figure 4.7 show that if we have a smaller value of p we must select larger block sizes so that the reconstruction process of the probabilities converge; if we select a higher value of p we can see the convergence at smaller block sizes, as shown in Figure 4.6.

4.7.3 Monte Carlo Simulation: Test of Independence

We generate a $k-$way noisy contingency table $T_{\mathbb{A}}$ using our proposed randomization technique. We calculated the parameters $\hat{m}_{i,j}$ using the two-step MLE procedure. Using these estimates, we sample $l > 1/\alpha$ many contingency tables (where α is the significance level, 0.05). We then add noise to these sampled tables using our randomized response protocol. Using the same two-step MLE calculation, we obtain l different $\hat{\chi}^2$ values from these sam-

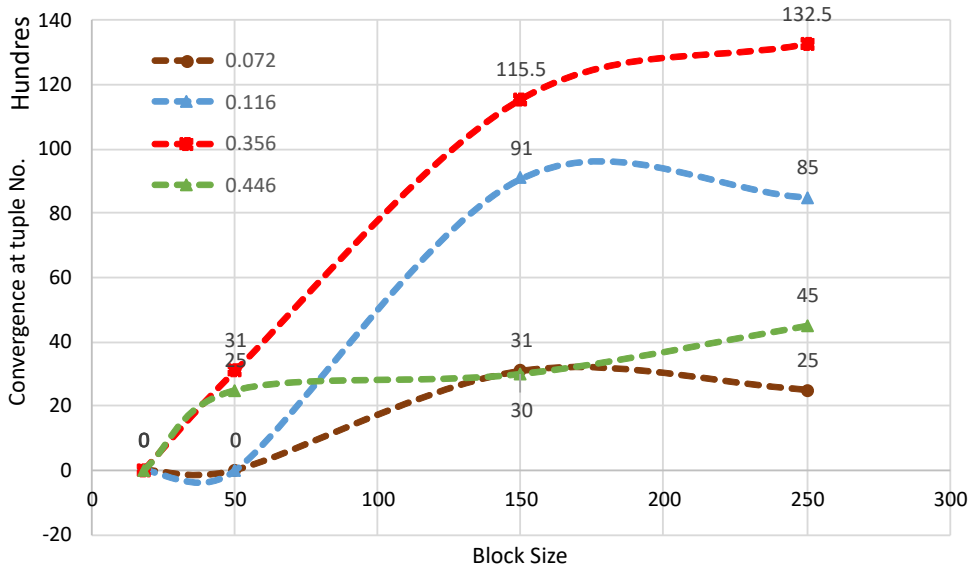


Figure 4.7: Convergence in probabilities $P(A_i = w_i) = \{0.072, 0.116, 0.224, 0.356, 0.446, 0.524, 0.732\}$ and block size $s = \{18, 50, 150, 250\}$ at probability p (first coin is head) $= \{0.009, 0.048, 0.095, 0.139, 0.221\}$

pled noisy tables. We rank these statistics by choosing $\lceil (l+1)(1-\alpha) \rceil$ as threshold $\widehat{\vartheta}^{\alpha}$. If $\widehat{\chi}^2 > \widehat{\vartheta}^{\alpha}$ we Reject H_0 else, we Accept H_0. If at any point the two-step MLE calculation outputs any cell count < 5 then we Accept H_0.

Significance Results

We now show how our tests of Independence perform on real-world data when H_0 is both rejected or accepted. We set $\alpha = 0.05$, level of significance $(1-\alpha) = 0.95$, $\gamma = 0.01$ as the parameter in the two-step MLE, and the privacy budget $\epsilon = 0.25$ in all our tests.

We perform the independence testing on $2-$way, $3-$way, and $4-$way contingency tables with binary attributes. Note that the independence tests can also be performed on arbitrary $r \times c$ noisy contingency tables generated by our proposed method. Notice that as soon as the number of values increases, our protocol is more robust than the others and succeeds in the tests a higher number of times.

In the above experiments with Laplace distribution, since it does not provide critical values, we used the true values of the attributes as the values for the comparison with noisy data: they are known in advance because the selected datasets are publicly available. If this was not possible, one could also find the critical values of simulated data using **R** package "CompQuadForm".

		2-way		3-way		4-way	
		Reject H_0	Accept H_0	Reject H_0	Accept H_0	Reject H_0	Accept H_0
Laplace noise	Reject H_0	68	32	55	45	50	50
	Accept H_0	35	65	41	59	41	59
	Accuracy	66.5%		57%		54.5%	
MCIndep [76]	Reject H_0	94	6	94	6	92	8
	Accept H_0	5	95	7	93	9	91
	Accuracy	94.5%		93.5%		91.5%	
RRBA	Reject H_0	**96**	4	**94**	6	**93**	7
	Accept H_0	3	**97**	6	**94**	6	**94**
	Accuracy	**96.5%**		**94%**		**93.5%**	

Table 4.4: Comparison of independence tests on $k-$way contingency tables ($k = 2, 3,$ and 4) with Laplace noise, MCIndep (performs Monte Carlo independence testing to determine whether a given contingency table should be rejected or not, while ensuring differential privacy), and randomized response block aggregation (RRBA) on 100 trials with $\alpha = 0.05$, $p = 0.5$ (privacy parameter of our protocol), and privacy budget $\epsilon = 0.25$ (privacy parameter for Laplace noise and MCIndep).

In Table 4.4, we compare the performance of our proposed method with state-of-the-art competitors using a confusion matrix with Laplace noise and MCIndep [76]. We perform 100 trials for H_0 rejected and 100 trials for H_0 accepted with various parameters to generate the contingency tables. The accuracy of our method is excellent (96.5%, 94%, and 93.5%) in all $k-$way contingency tables. These results are better than both Laplace and MCIndep methods. Further, our block randomization protocol is robust even in sparse data, where contingency cells often have very low or zero count values. On the contrary, Laplace and MCIndep do not produce valid results in these extreme situations, which can be a killer application.

4.8 Conclusion

In this work, we systematically explore the problem of collecting and analyzing data from smart devices under $\epsilon-$local differential privacy, in which neither the aggregator nor the server are trusted, have access to randomized responses from the users, and reconstruct statistical models based on the perturbed data. The server can compute accurate statistics from these joint distributions as a basis for building more advanced machine learning models. With the experiments, we showed that our protocol achieves high utility in reconstructing the probabilities of attribute values committing a low error bound. In future work, we will

like to use the hash function to store these contingency tables to reduce the computation and communication overheads.

Chapter 5

Randomized Response, a Modified Version and α-Geometric Mechanism

This chapter will present two modified variations of our proposed randomization protocol. In the first part of this chapter, we propose a modified version using a Bayesian network. We utilize Bayesian networks to perturb the low-dimensional distribution of variables in the contingency tables. Our results show that this modified version provides better accuracy and utility than the previous randomization protocol. In the second part of this chapter, we proposed a hybrid approach by combining our proposed randomization protocol with the α-Geometric mechanism. This α-Geometric mechanism is also called the discretized version of the Laplace distribution.

5.1 Introduction

In this section, we present our modified randomization protocol. Our previous version focuses on publishing noisy *k-way* contingency tables. In this modified version, we present a robust solution to the problem of publishing differential private moderate-to-high-dimensional databases, unlike the solution proposed in the previous version. Our previous version focused on aggregating and optimizing the noisy responses from clients for specific workloads, like `count` and `sum` queries, which can be calculated directly from the noisy contingency tables released by the aggregator. Now, we focus on estimating the high-dimensional distribution of the original dataset with a data-dependent set of well-chosen low-dimensional distributions. Our results confidently show that the resulting noisy data will provide high

accuracy for almost any query type (linear function of data values or non-linear).

In any case, we still assume that the noisy dataset generated using our modified randomization protocol will obey the original schema and keep the same format as the true dataset.

In the result section, we provide extensive experimental evidence of the accuracy of our proposed modified randomization protocol. Our experiments show that this proposed version is more accurate than our previous version without implementing any optimization techniques.

We use the well-known Bayesian networks, graphical models widely studied in machine learning,0. and statistical analysis [77]. Bayesian networks combine low-dimensional distributions to estimate the high-dimensional distribution of the dataset. To achieve differential privacy, our proposed randomized response protocol[1] consists of the following steps:

1. (*Differential private structure learning*) In the first step, we construct a differential private Bayesian structure using our previous randomization protocol.

2. (*Differential private distribution learning*) We compute the differentially private joint distributions using our modified randomization version. These joint distributions are generated from the differentially private Bayesian network learned in the previous step.

3. (*Synthesis data publishing*) In the last step, we generate synthetic data from the noisy joint distributions generated in the previous step without explicitly materializing the full-dimensional distribution.

5.2 Preliminaries

This section provides an overview of the Bayesian network and necessary notations to help us establish our work.

5.2.1 Bayesian Network

A Bayesian network $\mathcal{G}$ is a probabilistic graphical model that represents a joint probability distribution on the set of attributes $\mathbf{A}$ by compact conditional dependencies among a certain subset of attributes in $\mathbf{A}$. A Bayesian structure, also known as DAG (Directed Acyclic

[1] Our proposed modified randomization versions are all executed in blocks as we have discussed in Section 4.4.2

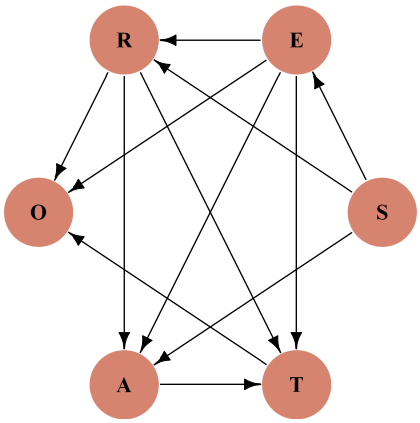


Figure 5.1: A Bayesian network generated on six attributes using a Hill Climbing algorithm

Graph), represents each attribute as a node and the conditional dependencies among the attributes using directed edges. For example, Figure 5.1 illustrates a Bayesian network constructed on the dataset D from Table 4.1a. This dataset comprises six attributes (`Age`, `Region`, `Education`, `Occupation`, `Sex`, and `Transportation`). For any two attributes $A_1, A_2 \in \mathbf{A}$, three possible cases exist between A_1 and A_2.

1. ***Direct dependence***: Suppose there is an edge directing from A_2 to A_1, then A_2 is the *parent* of A_1, and we refer to the set of all parents of A_1 as its *parent set*. This shows that for any tuple in the dataset D, its distribution on A_1 is calculated by its value on A_2. For example, in Figure 5.1, an edge from `Region` to `Transportation` indicates the type of transportation depends on whether the region is big or small.

2. ***Strong conditional independence***: If there is no edge between A_2 and A_1, then there is conditional independence given any parent sets of A_1 and A_2.

3. ***Weak conditional independence***: If there is no edge between A_2 and A_1, but a path exists between A_2 and A_1, then A_1 and A_2 are conditionally independent, given the A_1 parent set. For example, in Figure 5.1, there is no direct edge between `Sex` and `Occupation`, but we can reach `Occupation` from `Sex` through (`Education, Region`). This indicates that her occupation and sex are conditionally independent, given her education and region.

i	A_i	$\prod_i$
1	Sex	$\varnothing$
2	Education	{Sex}
3	Region	{Education, Sex}
4	Age	{Education, Region, Sex}
5	Transportation	{Age, Education, Region}
6	Occupation	{Education, Region, Transportation}

Table 5.1: The attribute-parent pairs in $\mathcal{G}$

A Bayesian network $\mathcal{G}$ extracted on a dataset D with d attributes $\mathbf{A} = (A_1, A_2, \cdots, A_d)$, is defined as a set of d *attribute-parent set (AP)* pairs, as follows:

$\big\{(\Pr[A_1|\Pi_1]), (\Pr[A_2|\Pi_2]), \cdots, (\Pr[A_d|\Pi_d])\big\}$

with A_i an attribute in $\mathbf{A}$. Each Π_i is a subset of attributes $\mathbb{A} \in \mathbf{A}$, which represents the parent set of A_i in the network $\mathcal{G}$.

Nodes are ordered according to an ordering defined as the possible parent sets for each node. For such an ordering, for any node whose position is i in this ordering, the node index is denoted by i_i. The ordering is such that $i_1 \leq i_i < i_j \leq i_d$ and we have $A_{i_j} \notin \Pi_{i_i}$. There is no edge from A_{i_j} to A_{i_i} in the Bayesian network, thus maintaining the acyclic nature of the network. We also define the degree k of the Bayesian network. The degree k defines a node's maximum parent set or the maximum number of incoming edges to a node. Table 5.1 shows the attribute-parent pairs in the Bayesian network $\mathcal{G}$, and Figure 5.1 shows the Bayesian network of degree 3 since the parent set of any node in the network has size at most 3.

Let $\Pr[\mathbf{A}]$ indicate the full distribution of attributes in the database. The attribute-parent set pairs in $\mathcal{G}$ approximate $\Pr[\mathbf{A}]$ with d conditional distributions given as $\Pr[A_1|\Pi_1]$, $\Pr[A_2|\Pi_2], \cdots, \Pr[A_d|\Pi_d]$. According to the DAG characteristics of the Bayesian network, any A_i and $A_j \notin \Pi_i$ are conditionally independent given Π_i. We have:

$$\begin{aligned}\Pr[\mathbf{A}] &= \Pr[A_1, A_2, \cdots, A_d] \\ &= \Pr[A_1] \cdot \Pr[A_2|A_1] \cdot \Pr[A_3|A_1, A_2], \cdots, \Pr[A_1, \cdots, A_{d-1}] \\ &= \prod_i^d \Pr[A_i|\Pi_i]\end{aligned} \tag{5.1}$$

If the Bayesian Network $\mathcal{G}$ captures the conditional dependence among the attributes in $\mathbf{A}$, then $\Pr_{\mathcal{G}}[\mathbf{A}]$ would be a good approximation of $\Pr[\mathbf{A}]$. If k is small, then the computation of $\Pr_{\mathcal{G}}[\mathbf{A}]$ is relatively simple, as it only requires d low-dimensional distributions.

5.2.2 Learning Non-Private Bayesian Structure

The k-degree Bayesian network $\mathcal{G}$ on the attributes $\mathbf{A}$ in the dataset D accurately captures the full distribution of the tuple in the dataset, i.e., $\Pr_{\mathcal{G}}[\mathbf{A}]$ should be close to $\Pr[\mathbf{A}]$. To estimate this approximation, we use the standard notions for information theory. The Entropy[2] of a random variable A over its domain $\mathbf{A}$ is given by

$$H(\mathbf{A}) = -\sum_{A \in Dom(\mathbf{A})} \Pr[A] \log \Pr[A] \tag{5.2}$$

and the mutual information $I(\cdot, \cdot)$ is given by

$$I(\mathbf{A}, \Pi) = \sum_{A,\Pi} \Pr[A, \Pi] \log \left(\frac{\Pr[A, \Pi]}{\Pr[A] \Pr[\Pi]} \right) \tag{5.3}$$

where $\Pr[A, \Pi]$ is the joint distribution of A and its parent set Π, and $\Pr[A]$, $\Pr[\Pi]$ are the marginal distribution of A and Π respectively. We use *Kullback–Leibler divergence* (KL divergence) [78] to calculate the distance between two probabilities distributions $\Pr_{\mathcal{G}}[\mathbf{A}]$ to $\Pr[\mathbf{A}]$, defined as:

$$D_{KL}(\Pr[\mathbf{A}] \,||\, \Pr_{\mathcal{G}}[\mathbf{A}]) = -\sum_{i=1}^{d} I(A_i, \Pi_i) + \sum_{i=1}^{d} H(A_i) - H(\mathbf{A}) \tag{5.4}$$

The term $\sum_{i=1}^{d} H(A_i) - H(\mathbf{A})$ in the above equation is decided by $\Pr[\mathbf{A}]$ and is fixed once the input dataset is given. Hence, the KL-divergence is small if and only if $\sum_{i=1}^{d} I(A_i, \Pi_i)$ is maximized. Therefore, the construction of a Bayesian network $\mathcal{G}$ is seen as an optimization problem, where we focus on selecting a parent set Π_i for each attribute A_i in the dataset to maximize $\sum_{i=1}^{d} I(A_i, \Pi_i)$.

We use the *GreedyBayes* technique, initially discussed in [1], to construct k-degree Bayesian networks. When $k = 1$, Chow et al. [79] show that greedily picking the next edge based on maximum mutual information is optimal. However, if $k > 1$, the optimization problem is NP-hard. For this reason many heuristic algorithms, like *hill-climbing* [80], *genetic algorithms* [81], and *simulated annealing* [82] are often used. Zhang et al. [1] modify the Chow-Liu approach for the greedy construction of a $k > 1$ degree's Bayesian network, as outlined in Algorithm 2.

Algorithm 2, at line 1, initializes the Bayesian network $\mathcal{G}$ and subset $\mathbb{A}$ to the empty set,

[2] All logarithms used in this chapter are of base 2

Algorithm 2 GreedyBayes [1]

Input: set of an attributes $\mathbf{A}$, degree of Bayesian network k
Output: Bayesian network $\mathcal{G}$

1 initialize $\mathcal{G} = \emptyset$ and $\mathbb{A} = \emptyset$;
2 randomly select an attribute $A_i \in \mathbf{A}$; add $(A_i, \emptyset)$ to $\mathcal{G}$; add A_i to $\mathbb{A}$;
3 **for** $i = 2$ *to* d **do**
4 initialize $\Omega = \emptyset$;
5 for each $A \in \mathbf{A} \setminus \mathbb{A}$ for each $\Pi \in \binom{\mathbb{A}}{k}$ add (A, Π) to Ω ;
6 select a pair (A_i, Π_i) from Ω with $I(A_i, \Pi_i)$ maximum mutual information ;
7 add (A_i, Π_i) to $\mathcal{G}$;
8 add A_i to $\mathbb{A}$;
9 **end**
10 **return** $\mathcal{G}$

where $\mathbb{A}$ contains all the attributes whose parent set has been fixed in the partial creation of $\mathcal{G}$. At line 2 the algorithm randomly selects any attribute A_i from $\mathbf{A}$ and sets its parent set to null ($\Pi_i = \emptyset$). From line 4 - 8, the algorithm consists of $d - 1$ iterations: in each iteration, it greedily adds to $\mathcal{G}$ an attribute-parent pair having the maximum mutual information. The attribute-parent pair is selected from the candidate set Ω that contains every attribute-parent pair (A, Π) satisfying the requirements listed in Section 5.2.1. Once the parent sets for each attribute are decided, the algorithm returns the Bayesian network $\mathcal{G}$ at line 10. In each iteration, i, the number of pairs considered is $(d - i)\binom{i}{k}$. Summing over all iterations, the computation cost is bounded by $d \sum_{i=1}^{d} \binom{i}{k} = d\binom{d+1}{k+1}$.

5.2.3 Private Bayesian Network

Observe that in Algorithm 2, the only time we access the true dataset D is when the algorithm at line 6, in each iteration, greedily selects an attribute-parent pair (A_i, Π_i). As a result, all we have to do to make Algorithm 2 differentially private is replace line 6 with a procedure that selects (A_i, Π_i) from Ω privately. Such a procedure can be implemented using our randomized response discussed in Section 4.4. Once, the GreedyBayes Algorithm 2 identified attribute-parent pairs $(A_i, \Pi_i) \in \Omega$, we generate the noisy distribution in $(A_i, \Pi_i) \in \Omega$ using Algorithm 1. In our randomization protocol, we use $T_{\mathbb{A}}$ to denote the contingency table over the joint distribution of a subset of attributes $\mathbb{A}$. So to adapt Algorithm 1 working to this situation, we have to consider the contingency table of the pair $\mathbb{A} = (A_i, \Pi_i)$. In the next Section, we explain the randomization of each candidate attribute-parent pair in Ω. We select the attribute-parent pair with the maximum mutual information $I(A_i, \Pi_i) \in \Omega$ using

Equation 5.3.

In the next Section, we explain how we generated the noisy joint distribution of each attribute-parent pair (identified as said by the GreedyBayes Algorithm 2). From these noisy joint distributions, we generated the conditional distributions that are needed in the Bayesian network at each edge.

5.2.4 Noisy Attribute-parent Pairs

This Section will discuss how we generated differential private conditional distributions of each attribute-parent (AP) pair. These are identified using Algorithm 2. Zhang et al. proposed `PRIVBAYES` [1] to generate noisy conditional distributions using the standard Laplace distribution $Lap(\frac{4(d-k)}{\epsilon})$ and their proposed algorithm can access the true dataset at least once. We do not have access to the original dataset in our LDP setting. We assume an environment where we build a synthetic dataset by randomizing users' responses.

`PRIVBAYES` is a differentially private method for releasing private data via Bayesian networks. The privacy in `PRIVBAYES` is achieved using Laplace noise. `PRIVBAYES` constructs a Bayesian network $\mathcal{G}$ on a given dataset D. The Bayesian network provides a compact model of the correlations between the attributes in the dataset. It enables the approximation of the probability distributions of the data using a set of low-dimensional marginals. After that, to ensure differential privacy, `PRIVBAYES` injects Laplace noise into each marginal. Then, it approximates the dataset probability distributions using the noisy marginals and the Bayesian network. Finally, `PRIVBAYES` constructs and releases a synthetic dataset by sampling tuples from the approximate distribution. The scale of the Laplace noise added to each low-dimensional marginal is set to $Lap(\frac{4(d-k)}{\epsilon})$ (where d is the number of attributes in the dataset, k is the degree of the Bayesian network) which ensures that the generated noisy marginals are differentially private.

We now outline the generation of the noisy conditional distributions of AP pairs in Algorithm 3. Once the GreedyBayes Algorithm 2 generates AP pairs $(A_i, \Pi_i) \in \Omega$ at line 5, we pass these candidates set Ω to the `Aggregator` function in Algorithm 1. We then start collecting randomized user responses at line 4 of the `Noisy Conditionals` Algorithm 3. Once the aggregator has collected all the responses in the joint distribution of each AP pair in Ω, from lines 5 - 6, the Algorithm 3 normalizes these distributions using *MinMix* normalization 4.4 and adds them to Ω'. The set of noisy candidates Ω' is then sent to GreedyBayes

Algorithm 2 for the computation and selection of the pairs having the maximum mutual information. The Algorithm 3 iteratively performs these steps until all the AP pairs are randomized.

Algorithm 3 Noisy Conditionals

Input: candidate set Ω
Output: noisy AP pairs Ω'

initialized $\Omega' = \emptyset$;
while exists a AP pair $\in \Omega$ not yet randomized **do**
    initialize the joint distribution $\Pr[A_i, \Pi_i] \in \Omega$;
    generate differentially private $\Pr'[A_i, \Pi_i]$ using Algorithm 1 ;
    normalize $\Pr'[A_i, \Pi_i]$;
    add $\Pr'[A_i, \Pi_i]$ to Ω'
end
return: Ω'

Using our randomized response algorithm, we generated a differential private Bayesian network by combining GreedyBayes and `Noisy Conditionals`. The only problem with this setting is that each time the `Aggregate` function collects randomized responses from the clients in the candidate set Ω, it might release some information to the adversary. Since each attribute A_i is repeatedly combined with other attributes in Ω, this might leak some information about the attribute values. Furthermore, collecting all the responses within each AP pair is computationally expensive.

We proposed a modified version of our randomization protocol to solve the above problems. We execute our previous version of the randomization protocol only on a few blocks to learn the Bayesian structure in the LDP setting. Once we learned the full dimensional distribution of the attributes $\Pr_{\mathcal{G}}[\mathbf{A}]$, we stopped and executed our modified randomization protocol. At this point, the aggregator uses the noisy data $\Pr'_{\mathcal{G}}[\mathbf{A}]$ generated while learning the Bayesian structure, collects new randomized responses using the proposed modified randomization version, and aggregates them with the already collected noisy data $\Pr'_{\mathcal{G}}[\mathbf{A}]$. As an alternative, the server can directly start with the modified version of the randomized protocol to collect randomized responses based on the probability distribution given by $\Pr_{\mathcal{G}}[\mathbf{A}]$ to generate noisy $\Pr'_{\mathcal{G}}[\mathbf{A}]$.

5.2.5 Modified Randomization Protocol

Suppose we are given a k-degree Bayesian network $\mathcal{G}$, as shown in Figure 5.1. Our modified version utilizes only $(d - k)$ joint distributions to be randomized in a differentially private manner. For example, suppose we have $k = 3$, then the algorithm will select AP pairs that contain only $(k + 1)$ attributes in the joint distributions $\Pr[\mathbf{A}]$ to be randomized. The full-dimensional distribution will be reconstructed from the $(d - k)$ joint distributions. The setting of our modified version is the same as our previous randomization protocol, where we have a server (that performs the aggregator) that collects randomized client responses. These responses are collected in contingency tables. As far as the notation is concerned, in this chapter, we denote contingency tables of AP pairs by using the same notation we used in $T_{\mathbb{A}}$, where the subscript refers to the AP pair, with $\mathbb{A} \in (A_i, \Pi_i)$). The server can send a query for each AP pair and collects randomized responses. Again, we have p, $(1 - p)$, and q probabilities to adjust the privacy and utility of the modified randomized response protocol.

We use the same Ω to denote the set of AP pairs to be randomized. The server randomly selects any AP pair $T_{\mathbb{A}} \in \Omega$ to collect responses from each client. With probability p, the client answers a true value of the attributes in the AP pair. With probability $(1 - p)$, the client replies a "fake" value using Monte Carlo sampling based on probability q. Recall that AP pairs are statistically independent as defined by Bayesian networks. The second version of our randomization protocol independently performs Monte Carlo sampling from each AP pair. This is an enhancement on the first version of our randomized protocol that sampled independently from the disjoint views. This sampling in the views could emit the attribute values independently, thus introducing a possible distortion in the probability distributions of statistically dependent attributes separated in different views.

As said, in the second version of our protocol, the Monte Carlo sampling is performed differently from the previous version. In the first version of our randomized response protocol, the execution of a single Monte Carlo sampling was performed on a single table (or view) using a single probability mass table. In this new version, the schema from which Monte Carlo sampling emits ”fake” values belongs to an AP pair that could be ”spread” across multiple tables (views). Furthermore, differently from the first version of the protocol, the views’ schema might be overlapping. Our randomized protocol starts from an AP pair denoted by $T_{\mathbb{A}}$. The attributes of the AP pair might not be found in a single table (or view V_i), but some are found in one view and others in another. Each view corresponds to

a contingency table $T_{\mathbb{V}_i}$ from which we emit fake values that must be in agreement with the attributes that are in common. In our protocol, this "fake" value emittance occurs with the same probability as above $(1 - p)$ in all the involved views. The presence of an overlapping set of attributes among the different views in which the original schema of an AP pair is divided is an enhancement w.r.t. the previous protocol according to whom it was impossible to share any attributes among the different views.

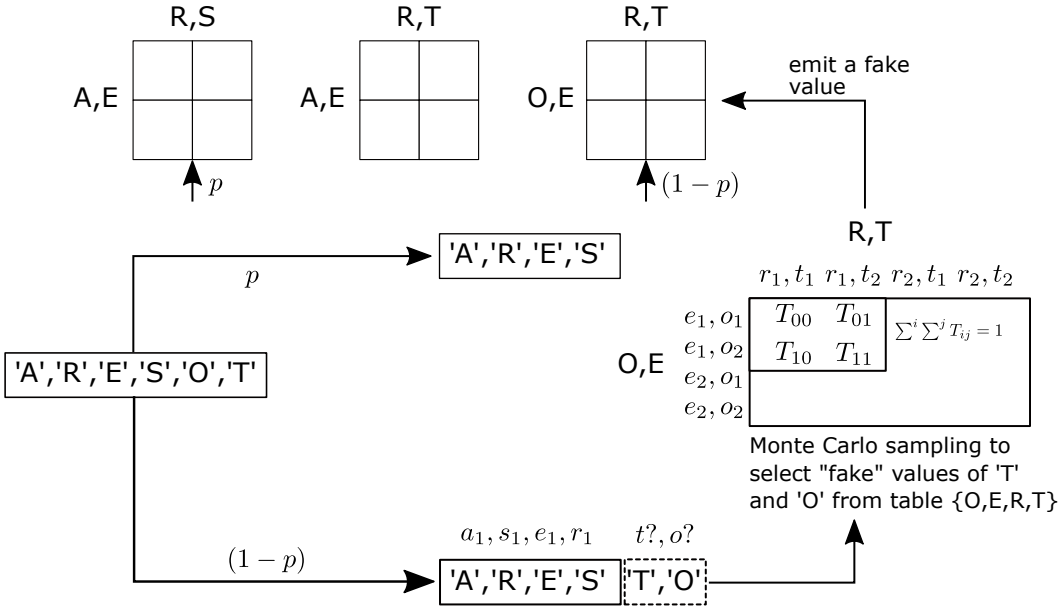
To better understand how we emit a fake value in Monte Carlo sampling, we use Figure 5.2 with Example 5.2.1. In this example, we consider that the aggregator works with the modified version of our protocol and collects randomized responses based on the probability distribution given by $\Pr_{\mathcal{G}}[\mathbf{A}]$ to generate noisy distributions $\Pr'_{\mathcal{G}}[\mathbf{A}]$.

Example 5.2.1. *Let us consider a k-degree Bayesian network as shown in Figure 5.1. The* $(k + 1)$ *AP pairs* $\Omega = \{(A, E, R, S), (A, E, R, T), (O, E, R, T)\}$. *For the sake of simplicity, we assume that these attributes are all binary. We use these AP pairs* Ω *in the modified version of our randomization protocol. As shown in Figure 5.3a, the aggregator arbitrarily selected an AP pair* $T_{\mathbb{A}} = (A, R, E, S)$ *to be randomized. With probability p, the client emits true values of all the attributes in* $T_{\mathbb{A}}$. *With probability* $(1 - p)$, *the client emits fake values. The attributes that are not included in* $T_{\mathbb{A}}$, *i.e., the remaining attributes, do not depend on the attributes in the AP pair just selected* $T_{\mathbb{A}} = (A, R, E, S)$ *and can be emitted independently, i.e., even with fake values by Monte Carlo sampling. In the example, these latter attributes are* T *(*`Transportation`*) and* O *(*`Occupation`*) that can be emitted from the single view (an AP table)* (O, E, R, T) *by Monte Carlo sampling.*

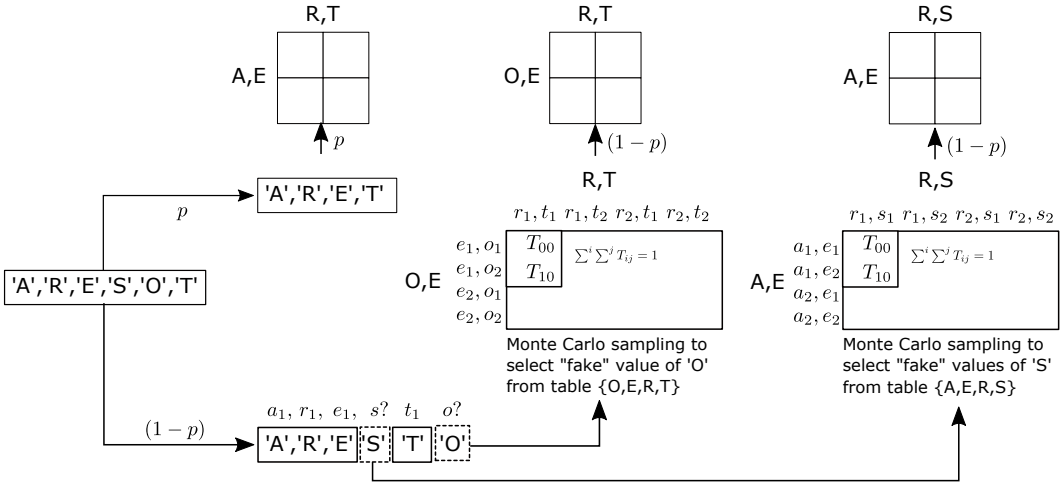
Suppose the true values emitted of the attributes (E, R) *are* e_1 *and* r_1. *If the client emits fake values with probability* $(1 - p)$ *for* (T, O), *these latter need to be combined with the true values* e_1, r_1. *The Monte Carlo sampling is performed using the probability q in Equation 4.2 referred to the contingency table of the second view* $T_{\mathbb{A}} = (O, E, R, T)$ *whose schema contains* (T, O). *Suppose, from Figure 5.3a, the values emitted correspond to the cell with values* (t_1, o_2)[3]. *They are combined with the other attribute values, and the client emits* (e_1, o_2) *and* (r_1, t_1).

Now consider that the aggregator randomly selected the table $T_{\mathbb{A}} = (A, E, R, T)$. *With probability p, the client emits true values, and with* $(1 - p)$, *it emits fake values. It performs similarly also for the remaining attributes, which are* S *(*`Sex`*) and* O *(*`Occupation`*).*

[3]In Figure, we slightly abuse the notation and use T_{10} to indicate the cell in the contingency table T at the second row and first column, i.e. using a binary encoding of the values at rows/columns.



(a) Monte Carlo sampling to emit "fake" value from a single table



(b) Monte Carlo sampling to emit "fake" values using multiple tables

Figure 5.2: Monte Carlo sampling to emit fake value

Algorithm 4 The second version of our randomization protocol with AP pairs and views

Input: The set of $(k+1)$ AP pairs Ω, the set of views V=$\{V_i\}$
Output: The noisy row **r** to be emitted
Init: r=$\emptyset$; `/* output row */`
Init: S=$\emptyset$; `/* portion of the row schema already emitted */`
1 **for** *each AP pair* $T_A \in \Omega$ **do**
2 flip the first coin;
3 **if** *the first coin gives "Head"* **then**
4 $A' \leftarrow$A $\setminus S$;
5 r = r $\cup$ row[A']; `/* row[A'] are the true row attributes known only to the user */`
6 S = S $\cup$ A;
7 **else**
8 draw the second random variable: $q \in [0..1]$;
9 $V' \leftarrow$ set of views $\{V_j\} \mid \bigcup_j$ Schema$(V_j) \supseteq A$;
10 **for** *each* $V_j \in V'$*:* **do**
11 $A_j \leftarrow Schema$(V$_j$)$\cap$A;
12 $A_{js} \leftarrow A_j \cap S$; `/* attribute values already emitted */`
13 $T_{V_j|r[A_{js}]} \leftarrow$ subset of T_{V_j} indexed by $r[A_{js}]$;
14 $T' \leftarrow normalization\_sums\_to\_1$(T$_{V_j|r[A_{js}]}$);
15 $w \leftarrow$ Monte Carlo (T', q); `/* return the attribute values` w `corresponding to` q `*/`
16 r = r $\cup$ w;
17 S = S $\cup A_j$;
18 **end**
19 **end**
20 **end**
Output: row r

Two contingency tables are needed to perform Monte Carlo sampling for emitting the pair of values of (S, O)*:* (A, E, R, S) *and* (O, E, R, T)*, as shown in Figure 5.3b. Suppose the client's true values are* (a_1, r_1, e_1, t_1)*, so with probability p, those values are emitted, and with probability* $(1-p)$*, the client emits a fake value for O using Monte Carlo sampling in the contingency table* (O, E, R, T) *and a fake value for S in the other contingency table* (A, E, R, S).

The algorithm just described in Example 5.2.1 is formalized as Algorithm 4.

The above algorithm repeatedly chooses an AP pair and for its attributes (A) it emits their values (either true or "fake" ones). It flips the first coin (line 2): if "Head" it emits the true values of the private client data (*row*) otherwise it runs Monte Carlo sampling (line 15) on a set V' of contigency tables (T_{V_j}) of views V_j whose overall schema cover the attribute schema A (line 9). Views store the knowledge acquired about the frequencies of the attribute values and Monte Carlo sampling will rely on them to generate the "fake" responses. It draws the random variable q (line 8) to decide the attribute ("fake") values to be communicated in

the output row r. For each view V_j these attributes are denoted by A_j (line 11). Each time it emits an attribute value it progressively extends the schema S of the row (lines 6 and 17) to keep memory of the attribute values already emitted in order to avoid multiple emittances and possible inconsistencies. For view V_j the attributes already emitted are stored in A_{js} and $r[A_{js}]$ denotes the values already emitted. $T_{V_j|r[A_{js}]}$ denotes the portion of the contingency table of view V_j whose cells are indexed with the attribute values already emitted (A_{js}) in the output row r. Monte Carlo sampling will have to restrict its action to these cells because they are the ones that agree with the already emitted attribute values. In case none of the attributes in view V_j has been previously emitted A_{js} is empty and the contingency table $T_{V_j|r[A_{js}]}$ coincides with the original one T_{V_j}. The portion of the contingency table is normalized such that the sum of the probabilities of the cells values gives 1 (at line 14 with `normalization_sums_to_1`). This is necessary so that the probability value q drawn from the second random variable can be used to return the values w by Monte Carlo sampling on the contingency table $T_{V_j|r[A_{js}]}$. The obtained "fake" values w are thus in agreement with the already emitted values and drawn according to the value of q. The values w are added to the ouput row r (line 16) and the schema S is updated with the already emitted attributes (line 17).

5.2.6 Noisy Conditional Distribution in Modified Randomization Protocol

Our modified randomization protocol generates noisy joint distributions of all $\Pr'(A_i, \Pi_i)$ with $i \in [k+1, d]$ AP pairs. Our proposed randomization protocol satisfies $\epsilon-$differential privacy since the randomization is achieved using a similar randomization mechanism of the method presented in Section 4.4.2, which is $\epsilon-$differential private. After obtaining the noisy $\Pr'[A_i, \Pi_i]$ distribution, our algorithm generates noisy conditional distributions of all $\Pr'(A_i|\Pi_i)$ with $i \in [k+1, d]$ AP pairs.

To construct the approximate distribution $\Pr_{\mathcal{G}}[\mathbf{A}]$, we need a conditional distribution $\Pr[A_i|\Pi_i]$ with $i \in [1, d]$, as shown in Equation 5.1. After $\Pr'[A_{k+1}|\Pi_{k+1}], \cdots, \Pr'[A_d|\Pi_d]$ are constructed, we proceed to generate $\Pr'[A_i|\Pi_i]$ with $i \in [1, k]$. However, this generation does not require any additional information from the clients. Instead, the algorithm derives $\Pr'[A_i|\Pi_i]$ with $i \in [1, k]$ directly from $\Pr'[A_{k+1}, \Pi_{k+1}]$ [4]. Such deriving is possible since

[4]Since generating lower degree distributions from the upper level will generate inconsistencies among the values at the lower level, one can apply any linear optimization technique to solve these inconsistencies. We discussed and solved this problem by applying an additional post-processing technique in Section 4.4.3. For simplicity, we omit such optimization from this presentation.

the algorithm for constructing the Bayesian network $\mathcal{G}$ in Section 5.2.1 ensures that $A_i \in \Pi_{k+1}$ and $\Pi_i \subset \Pi_{k+1}$ for any $i \in [1, k]$.

We consider that each $\Pr'[A_i|\Pi_i](i \in [1, k])$ is generated from $\Pr'[A_{k+1}, \Pi_{k+1}]$ without interactions with the clients. Hence, the construction of $\Pr'[A_i|\Pi_i]$ does not incur any privacy cost. We explain this derivation $\Pr'[A_i|\Pi_i](i \in [1, k])$ in Example 5.2.2.

Example 5.2.2. *We consider a Bayesian network $\mathcal{G}$ in Figure 5.1. Our modified randomization protocol generated three noisy joint distributions* $\Pr'[A, E, R, S]$, $\Pr'[T, A, E, R]$, *and* $\Pr'[O, T, R, E]$. *Our algorithm generates noisy conditional distributions* $\Pr'[A|E, R, S]$, $\Pr'[T|A, E, R]$, *and* $\Pr'[O|T, R, E]$ *based on these joint distributions. In addition, the algorithm uses* $\Pr'[A, E, R, S]$ *to derive other conditional distributions* $\Pr'[R|E, S]$, $\Pr'[E|S]$, *and* $\Pr'[S]$. *Given these six conditional distributions, we can approximate the full distribution of the input tuple as*

$$\Pr'_{\mathcal{G}}[A, R, E, S, O, T] = \Pr'[S] \cdot \Pr'[E|S] \cdot \Pr'[R|E, S] \cdot \Pr'[A|E, R, S] \cdot \Pr'[T|A, E, R] \cdot \Pr'[O|T, R, E]$$

5.2.7 Generation of Synthetic Data

One of the advantages of using the Bayesian Network is that it provides a means to approximate the sampling of attributes without materializing $\Pr'_{\mathcal{G}}[\mathbf{A}]$. In the first version of our randomization protocol, the server releases noisy contingency tables, considered a workhorse for data analysis. In this modified version, we can release the whole synthetic data, which can be used to perform many machine learning tasks or more sophisticated statistic modeling. As shown in the Equation 5.1, we can sample each A_i from $\Pr'[A_i|\Pi_i]$ independently, without considering any attribute not in $\Pi_i \cup A_i$. Furthermore, the acyclic characteristics of the Bayesian network ensure that for any $i_j > i_i$, $A_{i_j} \notin \Pi_{i_i}$. As a result, if we sample A_{i_i} with $i_i \in [i_1, i_d]$ in increasing order of i_i, we must have sampled all attributes in i_j by the time we sample A_{i_j} with $i_j \in [i_2, i_d]$, i.e., we will be able to sample A_{i_j} from $\Pr'[A_{i_j}|i_j]$ given the previously sampled attributes. The sampling of A_{i_j} does not require the full distribution $\Pr'_{\mathcal{G}}[\mathbf{A}]$.

Using the above sampling technique, we can produce an arbitrary number of tuples from $\Pr'_{\mathcal{G}}[\mathbf{A}]$ to publish a synthetic dataset D'. We generated the n number of tuples in the synthetic dataset, equivalent to the total number of responses collected from the clients.

5.3 Experiments on Modified Randomization Protocol

We present an empirical evaluation of our modified randomized response protocol on real datasets. For experimental reproducibility, we use three publicly available datasets *Survey* [73], *Alarm* [74], and *Child* [75]. We recall here their characteristics. The `Survey` dataset has 5k records, six attributes, and 14 categories. The `Alarm` dataset has 10k records with 37 attributes and a total of 103 categories, and the `Child` has 10k records with 20 attributes and 60 categories. All attributes in these datasets are discrete.

We performed three experiments to evaluate the accuracy of our proposed modified randomization protocol with Laplace noise and our previous randomization protocol. In our first experiment, Section 5.3.1, we evaluate the accuracy of the generated synthetic dataset D' with the original dataset D by comparing the Bayesian structure generated on both datasets (D', D). Next, we use ℓ_1 distance and *Jensen-Shannon divergence* to evaluate the performance of our proposed modified version.

5.3.1 Performance for Bayesian Networks

We use the heuristic approach "Hill-Climbing" to generate Bayesian networks on the noisy and the original datasets. We use the symbols $\mathcal{G}$ and $\mathcal{G}'$ to denote the original and the noisy networks, respectively. We generated the adjacency matrix of $\mathcal{G}$ and $\mathcal{G}'$ as boolean vectors g and g' so that each vector i-th entry is a boolean indicator of the presence of the link i. We compared two models: the true model g and the noisy model g', which evaluate the presence of each link on a given link collection with size L (in our case, for $d = 6$ attributes, we have $L = 30$ possible links). If we consider the original vector g as the ground truth of this binary classification task, we can compare the binary metrics g' with ground truth g; i.e., the number of links in the collection that are true positive (TP), true negative (TN), false positive (FP), and false negative (FN). Then, we can construct a set of Bayesian networks performance metrics as shown in Table 5.2.

Sensitivity (*sens*) measures the ratio of true positives (TP) over the total positives of the ground truth g (TP + FN), that is, the percentage of g present links that are correctly identified by g'. Specificity (*spec*) measures the ratio of true negatives (TN) over the total negatives of the ground truth g (TN + FP), that is, the percentage of g absent links that are correctly identified by g'. Average recall (*avg_recall*) measures the average between sensitivity (*sens*) and specificity (*spec*). Accuracy (*acc*) measures the ratio of true positives

Binary Metrics	**Bayesian Network Performance Metrics**
$TP = \sum_{i=1}^{L} \mathbf{I}(g_i = 1, g'_i = 1)$	$sens = TP/(TP + FN)$
$TN = \sum_{i=1}^{L} \mathbf{I}(g_i = 0, g'_i = 0)$	$spec = TN/(TN + FP)$
$FP = \sum_{i=1}^{L} \mathbf{I}(g_i = 1, g'_i = 0)$	$avg\_recall = (sens + spec)/2$
$FN = \sum_{i=1}^{L} \mathbf{I}(g_i = 0, g'_i = 1)$	$acc = (TP + TN)/(TP + TN + FP + FN)$

Table 5.2: Binary metrics of Bayesian vector $\mathbf{g}'$ with respect to ground truth $\mathbf{g}$ ($\mathbf{g}$ and $\mathbf{g}'$ are boolean link-indicator vectors of a link collection of size L). From binary metrics, Bayesian performance metrics are evaluated: sensitivity (*sens*), specificity (*spec*), average recall (*avg_recall*), and accuracy (*acc*).

and true negatives (TP + TN) over the link collection size L = TP + TN + FP + FN, which is the percentage of links, whether absent or present for the ground truth g, that are correctly identified by g'.

For the performance evaluation, we execute 500 times our randomization protocol to generate a noisy Bayesian network $\mathcal{G}'$ and each noisy vector g' is compared with the ground truth g. The noisy network $\mathcal{G}'$ is generated using our previous randomization protocol, the modified version, and Laplace noise[5]. We report the average performance in Table 5.3, where the error is computed as the standard deviation of performance. Figure 5.5 shows the distribution of these performance metrics.

	ACC	**SENS**	**SPEC**	**AVG_RECALL**
Modified randomization protocol	**0.81 ± 0.06**	**0.93 ± 0.06**	**0.6 ± 0.07**	**0.76 ± 0.05**
Randomization protocol	0.74 ± 0.06	0.87 ± 0.06	0.47 ± 0.07	0.67 ± 0.05
Laplace noise	0.71 ± 0.06	0.81 ± 0.05	0.52 ± 0.09	0.66 ± 0.05

Table 5.3: Bayesian average performance metrics: sensitivity (SENS), specificity (SPEC), average recall (AVG_RECALL), and accuracy (ACC) evaluated for the Bayesian networks generated using (Laplace noise, previous Randomization protocol, and modified version). The metrics represent the level of accordance between the ground truth Bayesian network and the collection of networks obtained by our randomization protocols and Laplace noise.

Table 5.3 shows that the Bayesian network generated using our modified randomization protocol has the best consistency between the noisy Bayesian network and the ground truth. Our modified version beats the other protocols on all four scales, having a high accuracy, sensitivity, specificity, and average recall.

[5]In these experiments, the Laplace noise is generated with $Lap(\frac{2}{\epsilon})$, where $\epsilon = 0.1$.

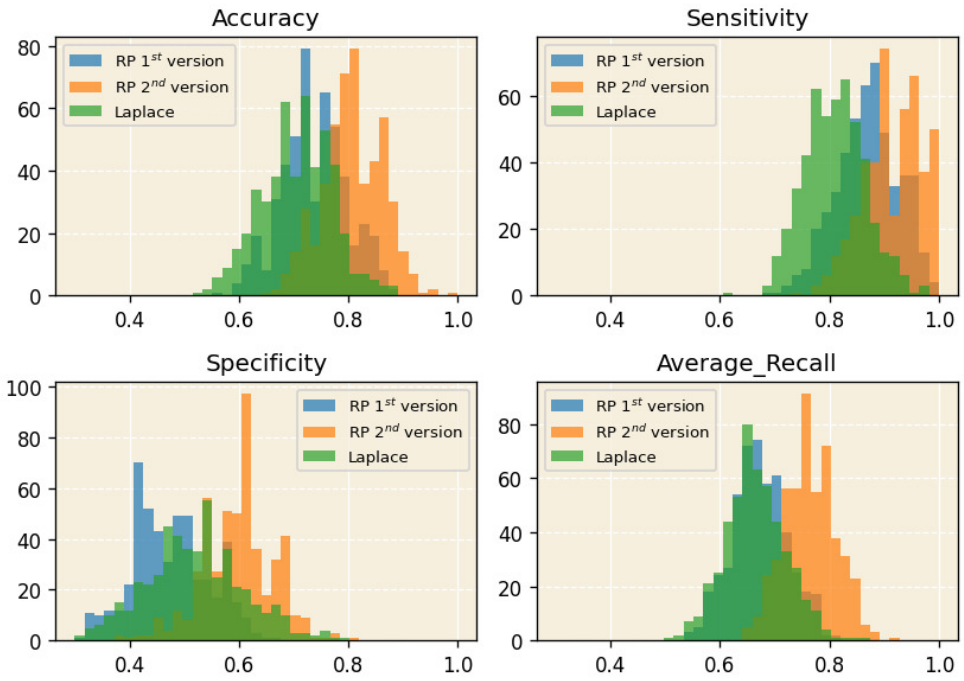


Figure 5.4: Bayesian network performance metric histograms (accuracy on the top left, sensitivity on the top right, specificity on the bottom left, and average recall on the bottom right) over the resulting collection of the noisy Bayesian networks generated using Randomization protocols (RR 1st version is the randomization version, RR 2nd is the modified randomization, and Laplace noise). The average of each performance metric for each Bayes network is reported in Table 5.3

5.3.2 Performance using ℓ_1 norm

In this section, we evaluate the performance of our proposed modified randomization protocol using ℓ_1 distance. For evaluation purposes, we use the noisy joint distribution AP pair. $\Pr'(A_i, \Pi_i)$ with $i \in [k+1, d]$. This noisy table is compared with the ground truth. The ground truth table is calculated directly on the unperturbed data[6]. A ground truth table contains the same attributes as a noisy table, so they have the same dimensionality. The ℓ_1 norm is the error metric referred to as the ℓ_1 distance between the noisy and the original table. Since each table is a probability density function (p.d.f.) table, we transformed these probabilities into real values by multiplying each cell entity with the total number of responses collected by the aggregator. We performed 500 trials on `Survey`, `Child`, and `Alarm` data sets and reported the average performance in Table 5.4. Figure 5.5 shows the distribution of these performance metrics.

[6]In experiments performed using ℓ_1 and Jensen-Shannon distance, the noisy w' and original w tables are generated on $\Pr(A_i, \Pi_i$ with $i \in [k+1, d]$ attributes.

	Survey	Child	Alarm
Modified randomization protocol	**61.38**	**60.3**	**51.79**
Randomization protocol	120.12	120.82	107.96
Laplace noise	124.79	125.06	112.93

Table 5.4: ℓ_1 distance between the true and the noisy AP pair table on `Survey`, `Child`, and `Alarm` dataset. The comparison is performed by evaluating the performance of the modified randomization with the previous randomization version and the Laplace noise with the original table (ground truth).

From Table 5.4, our modified randomization protocol has the lowest average ℓ_1 distance on all the three datasets (`Survey`, `Child`, and `Alarm`).

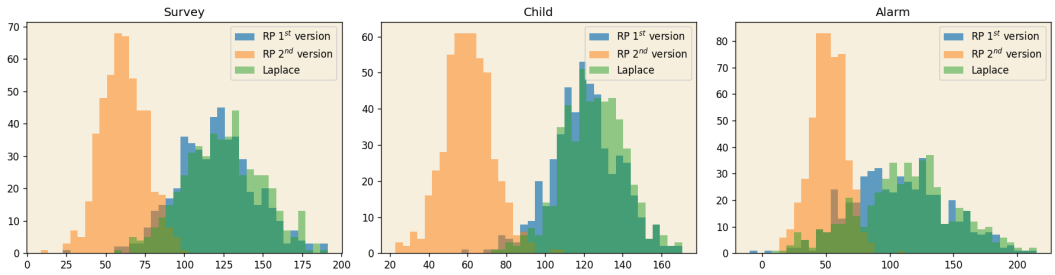


Figure 5.5: Modified randomization protocol, Randomization protocol, and Laplace noise performance metric histograms (`Survey` dataset on the left, `Child` in the middle, and `Alarm` on the right) over the resulting collection of the noisy AP pairs using ℓ_1 norm. The average of each performance metric for each dataset is reported in 5.4

5.3.3 Performance Evaluation using Jensen–Shannon Divergence

Jensen-Shannon divergence (*JSD*) is based on Kullback-Leibler divergence and is used to compute the similarities between two probability distributions. Shannon has some notable and valuable differences as compared with Kullback-Leibler. It always has a finite value, and it is symmetric. The square root of *JSD* is a valid metric distance between distributions [83].

$$JSD\left(T \parallel T'\right) = \frac{1}{2} D_{KL}\left(T \parallel Q\right) + \frac{1}{2} D_{KL}\left(T' \parallel Q\right) \tag{5.5}$$

where:

$$Q = \frac{1}{2}\left(T + T'\right)$$

and D_{KL} is Kullback-Leibler divergence on both T and T' is given as

$$D_{KL}(T \parallel Q) = \sum_{i \in I} \log\left(\frac{T_i}{Q_i}\right); \quad D_{KL}\left(T' \parallel Q\right) = \sum_{i \in I} \log\left(\frac{T'_i}{Q_i}\right)$$

We use the same noisy and original tables we used in the ℓ_1 distance experiments. The negative values in the noisy table are set to zero, and each distribution is normalized, ensuring a total probability mass of 1. Jensen-Shannon distance is calculated using Equation 5.5, where T' is the noisy AP pair table and T is the original table calculated directly on the unperturbed dataset. We perform 500 trials on `Survey`, `Child`, and `Alarm` datasets and report the average performance in Table 5.5. Figure 5.6 shows the distribution of these performance metrics.

	Survey	Child	Alarm
Modified randomization protocol	**0.03**	**0.04**	**0.03**
Randomization protocol	0.07	0.09	0.06
Laplace noise	0.08	0.1	0.083

Table 5.5: Jensen-Shannon distance between the true T and noisy T' contingency tables on `Survey`, `Child`, and `Alarm` dataset. The comparison is performed by evaluating the performance of the modified randomization, the original randomization, and the Laplace noise with the original table T (ground truth).

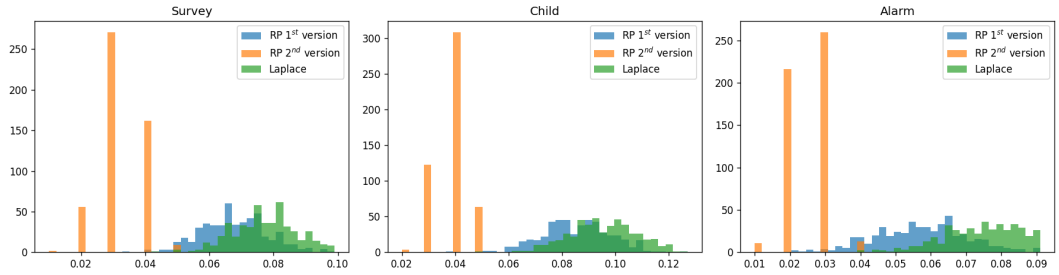


Figure 5.6: Modified randomization, randomization, and Laplace noise performance metric histograms (`Survey` dataset on the left, `Child` in the middle, and `Alarm` on the right) over the resulting collection of the noisy AP pairs tables using Jensen-Shannon distance. The average of each performance metric for each dataset is reported in Table 5.4

From Table 5.5, we can conclude that our proposed modified randomization model has the lowest distance on the Jensen-Shannon distance scale (a lower scale means that the noisy distribution is similar to the ground truth distribution). The runner-up is our previous randomization protocol, and then the Laplace noise. The results from the experiments (per-

formance metric using Bayesian networks comparison, ℓ_1 distance, and Jensen-Shannon divergences) show that our proposed modified model wins over our previous version and Laplace noise. Our proposed privacy protocol maximizes utility in the perturbed dataset while ensuring $\epsilon-$differential privacy.

5.4 Randomized Response using $\alpha-$Geometric Mechanism

This section presents a hybrid local differential private mechanism by combining our randomized response technique discussed in Section 4.4 and $\alpha-$geometric noise to create a more robust randomization model. Our results show that this hybrid model is more robust than the Laplace and the randomized response approach. First, in Section 5.4.1, we will explain differential privacy via $\alpha-$geometric mechanism. In 5.4.2, we will explain our proposed hybrid approach by combining randomized response and alpha-geometric noise. In Section 5.5, we evaluate the performance of our proposed hybrid approach with Laplace and previously discussed randomization protocols.

Geometric noise refers to a specific technique used in differential privacy to add noise to a dataset in order to preserve the privacy of individual users. This technique is based on the exponential mechanism, which is a general approach to differentially private data release.

The exponential mechanism works by selecting a randomized response from a set of possible outputs, with probability proportional to the utility function. In the case of geometric noise addition, the utility function is a matrix of probabilities, and the randomized response is a perturbed version of the original data.

5.4.1 $\alpha-$Geometric Mechanism

We consider a dataset D having n rows, where each row in the dataset represents an individual. Two datasets, D_1 and D_2 are neighbors if they differ in a single row. A `count` query f takes an input dataset D and returns the result $f(D) \in \mathbb{N} = \{0, \cdots, n\}$, that is the number of rows that satisfy a certain predicate on the domain D. Such queries are also called subset-count queries and are extensively studied in their own right [19, 84, 85].

A randomization mechanism $\mathcal{M} : \mathbb{N}^{|\mathcal{X}|} \to r$, with $r \in \mathcal{R}$. We use $\mathcal{M}_{D_r}$ to denote the probability of outputting a response r when the underline dataset is D. For $\alpha \in [0, 1]$, a mechanism $\mathcal{M}$ is α-differentially private if the ratio $\mathcal{M}_{D_1 r}/\mathcal{M}_{D_2 r}$ lies in the interval $[\alpha, 1/\alpha]$

for every possible response $r \in \mathcal{R}$ and neighbouring databases [7]. Intuitively, α is a control level of privacy. It is also the probability of every response of the privacy mechanism, independent of whether a specific user "opts in" or "opts out" in the dataset.

For all responses, $r \in \mathcal{R}$, a randomization mechanism $\mathcal{M}$ is *oblivious* such that $\mathcal{M}_{D_1 r} = \mathcal{M}_{D_2 r}$ whenever $f(D_1) = f(D_2)$, and the output distribution only depends on the query result. We can define an oblivious mechanism using the probability $\mathcal{M}_{ir}$ of outputting a response r for every query $i \in \mathbb{N}^{|D|}$. The α-differentially private is equal to the ratio of the probability $\mathcal{M}_{ir}/\mathcal{M}_{(i+1)r}$ for the query result $i \in \mathbb{N}^{|D|}$ and for every possible output $r \in \mathcal{R}$.

Definition 5.4.1 ($\alpha-$Geometric Mechanism [86])**.** *The $\alpha-$geometric mechanism is oblivious with a range $\mathbb{Z}$, a count query f, and a privacy parameter α, defined as a modification of the result of a true query $f(D)$ by the randomized mechanism that outputs $f(D) + Z$. Z is a random variable with a two-sided geometric distribution:*

$$\Pr[Z = z] = \frac{1-\alpha}{1+\alpha}\alpha^{|z|}. \tag{5.6}$$

The geometric mechanism uses $\alpha : (0 < \alpha \leq 1)$ to indicate $e^{-\epsilon}$, where ϵ is the level of privacy (or privacy budget). For every integer z, the $\alpha-$geometric mechanism is illustrated in Table 5.7a. For simplicity, Ghosh also shows that this oblivious mechanism will always return a true query result from $f(D)$ when $\alpha = 0$ ($0-$geometric mechanism) and always return 0 when $\alpha = 1$ ($1-$geometric mechanism), independent of the input dataset D.

The definition of the $\alpha-$geometric mechanism is equivalent to $\alpha-$differential privacy because the result of a count query $f(D)$ differs by at most one on two neighboring databases D_1 and D_2. For each z, the probabilities $\Pr[Z = z+1]$ and $\Pr[Z = z-1]$ lie between the interval $\alpha \Pr[Z = z]$ and $\Pr[Z = z]/\alpha$. The α-geometric mechanism is a discretized version of a Laplace distribution with density $\epsilon/2 \cdot e^{-\epsilon|z|}$ on $\mathcal{R}$, where $\epsilon = ln\frac{1}{\alpha}$.

The above mentioned $\alpha-$geometric mechanism can generate an infinite output domain. Ghosh considers the *truncated* version of the $\alpha-$geometric mechanism to deal with the infinite output. The idea is to remap the probability mass of every negative value to 0 and the probability mass of every value greater than n to n. The truncated $\alpha-$geometric mechanism is defined as:

[7] The definition of differential privacy 3.2.2 is based on the ratio of all range subsets. With a countable range, the two definitions are equivalent.

(a) $\frac{1}{2}-$geometric mechanism with n = 5

Input/Output	⋯	-1	0	1	2	3	4	5	6	⋯
0	⋯	1/6	1/3	1/6	1/12	1/24	1/48	1/96	1/192	⋯
1	⋯	1/12	1/6	1/3	1/6	1/12	1/24	1/48	1/96	⋯
2	⋯	1/12	1/6	1/3	1/6	1/12	1/12	1/24	1/48	⋯
3	⋯	1/48	1/24	1/12	1/6	1/3	1/6	1/12	1/24	⋯
4	⋯	1/96	1/48	1/24	1/12	1/6	1/3	1/6	1/12	⋯
5	⋯	1/192	1/96	1/48	1/24	1/12	1/6	1/3	1/6	⋯

(b) Truncated $\frac{1}{2}-$geometric mechanism with n = 5

Input/Output	0	1	2	3	4	5
0	2/3	1/6	1/12	1/24	1/48	1/48
1	1/3	1/3	1/6	1/12	1/24	1/24
2	1/6	1/6	1/3	1/6	1/12	1/12
3	1/12	1/12	1/6	1/3	1/6	1/6
4	1/24	1/24	1/12	1/6	1/3	1/3
5	1/48	1/48	1/24	1/12	1/6	2/3

Table 5.6: The probabilities of geometric and truncated geometric mechanisms, for $\alpha = \frac{1}{2}$ and $n = 5$. The columns correspond to the mechanism output $r \in \mathcal{R}$ and the rows correspond to the query result $i \in \mathbb{N}$

Definition 5.4.2 (Truncated geometric mechanism [86])**.** *The output of the $\alpha-$geometric mechanism is "obviously wrong" as $f(D) + Z$ - one less than 0 or greater than n - with non-zero probability. To deal with this problem, Ghosh proposed a truncated version of the $\alpha-$geometric mechanism by "remapping the probability mass of every negative value to 0, and the probability mass of every value greater than n to n. Particularly, the truncated mechanism applies the following distribution of noise Z when the query result is $f(D)$;*

$$\begin{aligned} \Pr[Z < -f(D)] = \Pr[Z > n - f(D)] = 0 \\ \Pr[Z = -f(D)] = \frac{\alpha^{f(D)}}{(1+\alpha)} \\ \Pr[Z = n - f(D)] = \frac{\alpha^{n-f(D)}}{(1+\alpha)} \end{aligned} \tag{5.7}$$

and all the other probabilities are as in the $\alpha-$geometric mechanism in Definition 5.4.1. Table 5.7b illustrates the $\alpha-$truncated mechanism. The truncated mechanism is also $\alpha-$differentially private.

5.4.2 Randomized Response and Geometric Mechanism

To create a more robust mechanism with privacy guarantees, we combine our randomized response block aggregation technique of Section 4.4 with the $\alpha-$geometric mechanism with the goal to create an hybrid approach for collecting the randomized responses from the users' in a differentially private manner. We use the same environment setting on the server side as we did in the first version of our protocol, in which a server aggregates responses from the users in a local differential privacy manner. We only change the setting at the client interface to utilize the geometric mechanism. Once the client receives a query from the aggregator on any attribute subset $\mathbb{A}$, the client responds according to the predefined probabilities p, $(1-p)$, and q. Probability p and $(1-p)$ are tunable to adjust the privacy and utility of the responses, whereas probability q is randomly drawn between $0 \leq q \leq 1$.

With a probability p, the client responds a true value of the attributes in $\mathbb{A}$, and with probability $(1-p)$ the client answers a "fake" value using Monte Carlo sampling controlled by q. However, at this time, the Monte Carlo sample is generated using the truncated geometric matrix, as shown in Table 5.8. For notational simplicity, we use C rather than $\mathcal{M}$ in the oblivious mechanism. The symbol C denotes the $\alpha-$truncated matrix, as shown in Table 5.8. We use C_{ir} to denote the probability that, on the input i, the α-truncated mechanism reports r. Hence, C_{ir} forms a stochastic matrix representing the conditional probability of i given r and C_{ir} is the element at the intersection of the i^{th} row and the r^{th} column, such that $\forall i, r \in [0,1] C_{ir} \geq 0$ and $\forall i \in [0,1] \sum_r C_{ir} \geq 0$.

We show how we perform Monte Carlo sampling for selecting a pair from the probability distributions in Table 5.8. We consider Table 4.1a (*ShortSurvey* dataset) as an example for Monte Carlo sampling. Suppose the server wishes to collect responses on two attributes $E = \{high, uni\}$ and $R = \{big, small\}$. Their joint distribution is given as $\{high, big\}$ $\{uni, big\}\{high, small\}\{uni, small\}$ as shown in the input and output of Table 5.8. Suppose the client's true response is $\{uni, high\}$, so at probability $(1-p)$, our protocol samples the fake value from the probabilities distributions of the second row in Table 5.8. The Monte Carlo sampling from the matrix C is given as:

$$\sum_{r=1}^{l-1} C_{ir} \leq q \leq \sum_{r=1}^{l} C_{ir} \tag{5.8}$$

where i is the true response of the client and r is the "fake" response sampled based on the

Input/Output	{high, big}	{uni, big}	{high, small}	{uni, small}
{high, big}	2/3	1/6	1/12	1/12
{uni, big}	1/3	1/3	1/6	1/6
{high, small}	1/6	1/6	1/3	1/3
{uni, small}	1/12	1/12	1/6	2/3

Table 5.8: Truncated $\frac{1}{2}$-geometric mechanism with $n = 4$

probability q in the Monte Carlo sampling. This fake response is emitted in such a way as to disclose a *"controlled"* amount of information about the client's true value. Hence, limiting the aggregator's ability to learn with confidence the true values of the client, Monte Carlo sampling improves the utility of our protocol by emitting combinations of values based on their probability as stored in the stochastic matrix C.

Once the aggregator receives a response from the client, it reconstructs a noisy contingency table. We will discuss the reconstruction of the original distribution from the collection of noisy data in Section 5.4.5. However, first, we will discuss the utility maximization of matrix C. The idea is that the probabilities in the alpha-truncated matrix can be further enhanced using the optimization technique discussed in [86]. We use the same linear optimization technique to enhance the accuracies of the probability distribution in the $\alpha-$truncated mechanism.

5.4.3 Utility Maximization

The utility model that Ghosh et al. [86] proposed provides strong and general utility guarantees for the truncated mechanisms. We desire a privacy mechanism that ensures maximum utility to every possible user, independent of their side information and preferences. Same as differential privacy, which offers protection against every potential attacker, regardless of their side information and preferences. Ghosh model a client's preferences via a *loss function* $l; l(i, r)$ defined on a matrix C. The loss function l denotes the client's loss when the query result is i, and the output of the $\alpha-$truncated mechanism is r. The loss function l is subject to the nonnegative and nondecreasing properties in $|i - r|$ for each fixed query i. For example, $l; |i - r|$ measures the mean error, or $(i - r)^2$, which essentially measures the variance of the error, or the binary loss function $l_{binary}(i - r)$, which is 0 if $i = r$ and 1 otherwise.

Ghosh models the side information of a client as a prior probability distribution $\hat{p}_i$ on the query result in $i \in \mathbb{N}$. This $\hat{p}_i$ denotes the client's beliefs, which is not introduced to weaken

the definition of differential privacy. The standard definition of differential privacy makes no assumption about the auxiliary information of an attacker. This prior probability is only used to discuss the utility of a private mechanism for a potential client.

Consider a prior $\hat{p}_i$, loss function l, oblivious mechanism C with range $\mathcal{R}$, and an input dataset D with query result $i = f(D)$. Then the user's expected loss is given as $\sum_{r\in\mathcal{R}} C_{ir} \cdot l(i,r)$, where the expectation is over the flip of a coin internal to the mechanism. Following that, the user's prior provides a measure of the mechanism's overall (dis)utility, and is given as:

$$\sum_{i\in\mathbb{N}} \hat{p}_i \sum_{r\in\mathcal{R}} C_{ir} \cdot l(i,r) \tag{5.9}$$

We now introduce the *optimal* $\alpha-$truncated mechanism using linear programming to minimize the user-specific objective function in Equation 5.9.

5.4.4 Linear Programming Formulation for a User-Optimal Mechanism

Could a single privacy mechanism be good enough for all users? Each user can post-process the output of a privacy mechanism, and this post-processing can reduce the user's expected loss. No matter what side knowledge and preferences a potential user has, the geometric mechanism provides as much utility as interacting with a differentially private mechanism that is best fitted to that user. The prior from the utility model has no impact on the definition of privacy. While the geometric method is user-independent (all users see the same distribution of responses), various users remap the responses differently, based on their prior distributions and loss function.

We use linear programming to determine the differentially private mechanism that minimizes the user's expected loss. While the objective function of this linear model is user-specific, the feasible region is not. We thus consider a privacy level $\alpha \in [0,1]$, a loss function l for a fixed user, and a prior probability distribution $\hat{p}$ to formulate a linear model. The privacy constraints in Section 5.4.1 combined with the objective function from Equation 5.9

Input/Output	{high, big}	{uni, big}	{high, small}	{uni, small}
{high, big}	2/3	1/4	1/12	0
{uni, big}	1/3	1/2	1/6	0
{high, small}	1/6	1/2	1/3	0
{uni, small}	1/12	1/4	2/3	0

Table 5.9: Optimization of the $\frac{1}{2}$-geometric mechanism with $n = 4$, for a user with prior $(\frac{1}{2}, \frac{1}{4}, \frac{2}{9}, 0)$, and loss function $l(i-r) = |i-r|^{1.5}$

yield the following linear program whose solution is optimal for the specific user.

$$\text{minimize} \quad \sum_{i\in\mathbb{N}} \hat{p}_i \sum_{r\in\mathbb{N}} C_{ir} \cdot l(i,r) \tag{5.10a}$$

$$C_{ir} - \alpha \cdot C_{(i+1)r} \geq 0 \qquad \forall r \in \mathbb{N}/\{n\}, \forall i \in \mathbb{N} \tag{5.10b}$$

$$\alpha \cdot C_{ir} - C_{(i+1)r} \leq 0 \qquad \forall r \in \mathbb{N}/\{n\}, \forall i \in \mathbb{N} \tag{5.10c}$$

$$\sum_{r\in\mathcal{R}} C_{ir} = 1 \qquad \forall i \in \mathbb{N} \tag{5.10d}$$

$$C_{ir} \geq 0 \qquad \forall i \in \mathbb{N}, \forall r \in \mathbb{N} \tag{5.10e}$$

The objective function 5.10a is the expected loss as in Equation 5.9 incurred by the user. Constraints (5.10b) and (5.10c) enforce $\alpha-$differential privacy, and constraint (5.10d) and (5.10e) ensure that the solution to this linear program can be interpreted as a probabilistic function. Table 5.9 shows an optimal privacy mechanism for a particular user. This table is generated by applying the linear optimization on Table 5.9, where $\alpha = 0.5, \hat{p} = \{\frac{1}{2}, \frac{1}{4}, \frac{2}{9}, 0\}$, and a loss function $l(i,r) = |i-r|^{1.5}$.

5.4.5 Reconstructing the original distribution from a collection of Noisy Data

Once the server receives the noisy data from the clients, it aggregates these noisy responses in the contingency table $T_{\mathbb{A}}$. To this goal, we use the same reconstruction Equation 4.3. But now we have p, $(1-p)$, and the $\alpha-$truncated matrix to reconstruct the original probabilities. We will consider the matrix C from Table 5.8 to design the system of linear equations for the reconstruction of the true probabilities.

First, we design the linear equation for the first query result i (the first row in the matrix C) and then generalize it for the whole matrix C.

Consider $i = 0$; we use x_i to denote the probability that the server observes a value in $\{high, big\}$ and the true query result is also $\{high, big\}$. For all other remaining probabili-

ties, we denote $x_j : [j \neq i, n]$ that are $\{x_2, x_3, x_4\}$ on which the server observes $\{high, big\}$, but the true query result is not $\{high, big\}$. The linear equation is given as:

$$o_0 = p * x_0 + (1-p) * x_0 * \frac{2}{3} + (1-p) * x_1 * \frac{1}{6} + (1-p) * x_2 * \frac{1}{12} + (1-p) * x_3 * \frac{1}{12} \tag{5.11}$$

where o_0 is the observed value for the first query result $i = 0$. The generalized equation is given as

$$o_i = p \cdot x_i + (1-p) \cdot \sum_{i=0}^{n} \cdot x_i \cdot C_{ir}. \tag{5.12}$$

Expanding this generalized Equation 5.12 on the Table 5.8, gives the following system

$$O_0 = p * x_0 + (1-p) * x_0 * \frac{2}{3} + (1-p) * x_1 * \frac{1}{6} + (1-p) * x_2 * \frac{1}{12} + (1-p) * x_3 * \frac{1}{12} \tag{5.13a}$$

$$O_1 = (1-p) * x_0 * \frac{1}{3} + p * x_1 + (1-p) * x_1 * \frac{1}{3} + (1-p) * x_2 * \frac{1}{6} + (1-p) * x_3 * \frac{1}{6} \tag{5.13b}$$

$$O_2 = (1-p) * x_0 * \frac{1}{6} + (1-p) * x_1 * \frac{1}{6} + p * x_2 + (1-p) * x_2 * \frac{1}{3} + (1-p) * x_3 * \frac{1}{3} \tag{5.13c}$$

$$O_3 = (1-p) * x_0 * \frac{1}{12} + (1-p) * x_1 * \frac{1}{12} + (1-p) * x_2 * \frac{1}{6} + p * x_3 + (1-p) * x_3 * \frac{2}{3} \tag{5.13d}$$

We solve the above system of linear equations as $Ax - b = 0$ in a numerical solver to find the value of $x_i \in [0, n]$ as a reconstructed probability from the noisy data. Our experiment shows that the value reconstructed through the numerical solver was not accurate, and the accuracy was also not up to the standard. To the best of our knowledge, this is due to the fact that there exist similar probabilities in two consecutive cells in the matrix C, which makes the numerical solver unable to solve the linear equations accurately.

For reconstructing the original probabilities from the collection of noisy data, we use the *Matrix inversion* and *Iterative Bayesian* technique discussed in [87]. Before discussing how we perform the matrix inversion and the iterative Bayesian technique, we first converted the above linear equations into a transition matrix C'. In other words, this transition matrix combines our randomized response and the $\alpha-$truncated mechanism. The transition matrix

(a) The randomized response combined with with the $\frac{1}{2}-$geometric mechanism with $n = 4$

Input/Output	{high, big}	{uni, big}	{high, small}	{uni, small}
{high, big}	p + (1-p)*2/3	(1-p)*1/6	(1-p)*1/12	(1-p)*1/12
{uni, big}	(1-p)*1/3	p+(1-p)*1/3	(1-p)*1/6	(1-p)*1/6
{high, small}	(1-p)*1/6	(1-p))*/6	p + (1-p) * 1/3	(1-p)*1/3
{uni, small}	(1-p)*1/12	(1-p)*1/12	(1-p)*1/6	p+(1-p)*2/3

(b) Transition matrix C' with $p = 0.5$, $n = 4$, and $\alpha = \frac{1}{2}$.

Input/Output	{high, big}	{uni, big}	{high, small}	{uni, small}
{high, big}	5/6	1/12	1/24	1/24
{uni, big}	1/6	2/3	1/12	1/12
{high, small}	1/12	1/12	2/3	1/6
{uni, small}	1/24	1/24	1/12	5/6

Table 5.10: The probabilities that define the randomized response and the $\frac{1}{2}-$truncated geometric mechanism, for $\alpha = \frac{1}{2}$, $p = 0.5$, and $n = 5$. Columns correspond to the mechanism output $r \in \mathcal{R}$ and rows correspond to the query result $i \in \mathbb{N}$

C' is given in Table 5.11a.

The following property is important and will be used in the matrix inversion and in the iterative Bayesian technique.

Lemma 5.4.1 (The matrix C' is Invertible.)**.**

Proof. Consider the relationship between C' and the $\frac{1}{2}$-truncated geometric mechanism C, and observe that the highest probabilities of C are all on the diagonal. The matrix C' also corresponds to the same principal diagonal as C. Hence, C's rows are linearly independent. Since C' is obtained directly from C by adding the probability p on the diagonals, the rows of C' are still linearly independent (C'_{00} is still the highest probability of the column 0 and C'_{nn} is still the highest probability of column n). ■

Matrix Inversion technique

Suppose the transition matrix C' generates the perturbed probabilities (the observed probabilities o collected by the aggregator). To reconstruct the probabilities x, we can use the inverse $(C')^{-1}$ of the same transition matrix C. We can calculate the reconstructed vector x from the observed vector o as

$$x = o \cdot (C')^{-1} \tag{5.14}$$

Theorem 5.4.2 (Matrix Inversion [87])**.** *The vector x estimated as* $o \cdot (C')^{-1}$ *is the maximum*

likelihood estimator (MLE) of the relaxed a priori distribution ($\sum_i x_i = 1$ and $0 \leq x_i \leq 1$ are the exact constraints. The relaxed constraint only guarantee $\sum_i x_i = 1$) on the states that generated the transaction table C'.

Iterative Bayesian technique

Let us consider a vector x of size 2^n (n is the number of columns in the matrix C'). Let us consider a prior distribution on the states of the original rows, and o (of size 2^n) the posterior distribution on the state of the perturbed rows C'. The prior distribution is given by the random variables $\{X_1, X_2, \cdots, X_n\}$, while the state of the n perturbed rows in C' is given by the random variables $\{Y_1, Y_2, \cdots, Y_n\}$. Then for $0 \leq a, b \leq t = (2^n - 1)$ and $1 \leq i \leq n$, we have $\Pr[Y_i = b] = o_b$, and $\Pr[X_i = a] = x_a$. Also, $\Pr[Y_i = b \mid X_i = a] = C'_{ab}$ is the transition probability from state a to b.

Applying the Bayes rule, we have

$$\begin{aligned} \Pr[X_i = a \mid Y_i = b] &= \frac{\Pr[Y_i = b \mid X_i = a] \Pr[X_i = a]}{\Pr[Y_i = b]} \\ &= \frac{\Pr[Y_i = b \mid X_i = a] \Pr[X_i = a]}{\sum_{r=0}^{t} \Pr[Y_i = b \mid X_i = r] \Pr[X_i = r]} \\ &= \frac{C'_{ab} x_a}{\sum_{r=0}^{t} C'_{ab} x_r} \end{aligned} \tag{5.15}$$

we iteratively update x using

$$\Pr[X_i = a] = \sum_{b=0}^{t} \Pr[Y_i = b] \Pr[X_i = a \mid Y_i = b]$$

The update rule is then given by

$$x_a^{T+1} = \sum_{b=0}^{t} y_b \frac{C'_{ab} x_a^T}{\sum_{r=0}^{t} C'_{rb} x_r^T}$$

where the vector x^T represents the iteration at step T, and the vector $x^{(T+1)}$ denotes the iteration at step $T + 1$. We initialize $x^0 = y$, and iterate until two consecutive x^T, x^{T+1} iterations do not differ much.

The iterative Bayesian technique makes smaller errors than the matrix inversion procedure, especially when several columns are reconnected. This seems unintuitive as the matrix inversion technique was shown to provide the MLE estimator for x, satisfying $\sum_i x_i = 1$.

This can be explained as the Bayesian iterative technique gives the MLE estimator in the constrained space, i.e., for the subspace of $\sum_i x_i = 1$ that satisfies $\forall i, 0 \leq x_i \leq 1$. Since the number of rows is always non-negative, this subspace contains the exact original p.d.f. vector x. When the number of columns in the matrix to be reconnected increases, the error during the randomization and the reconnection increase and the matrix inversion technique may return a point outside the constrained boundary, the reconnection error by the matrix inversion method can grow arbitrarily [87].

Once we have the reconstructed vector x, we assign this vector to the contingency table $T_{\mathbb{A}}$ such that, $T_{\mathbb{A}}[w_i] = x_i$. We then use this reconstructed contingency table $T_{\mathbb{A}}$ as prior probabilities used in the utility maximization as discussed in Section 5.4.3 to update C'. This updated matrix C' is then used in the next block to collect randomized responses.

5.5 Experiments on Randomize $\alpha-$Geometric Mechanism

To evaluate the performance of our proposed randomized response $\alpha-$geometric mechanism, we set the same experimental environment as we did in Section 5.3.1. We only use the ℓ_1 distance to measure the performance of our proposed randomized response $\alpha-$geometric mechanism compared with the modified randomization 5.2.5, the original randomization 4.4, and the Laplace noise. We report the average performance of each randomization protocol in Table 5.12. Figure 5.7 shows the distributions of these performance metrics.

	Survey	Child	Alarm
Modified randomization protocol	**61.38**	**60.3**	**51.79**
Randomization using Geometric mechanism	94.74	102.52	97.81
Randomization protocol	120.12	120.82	107.96
Laplace noise	124.79	125.06	112.93

Table 5.12: Comparison of ℓ_1 distance between the noisy and the original distributions (`Survey`, `Child`, and `Alarm` datasets); noise is added using the three randomization protocols and the Laplace noise.

Table 5.12 shows that our modified randomization protocol has the lowest ℓ_1 distance among all the randomization protocols. The runner-up is the randomization protocol combined with the geometric mechanism. This is because the dimensionality of the contingency table induces more noise in the probability distribution using the $\alpha-$geometric mechanism. In our modified randomization protocol, the Monte Carlo simulation concentrates only on the

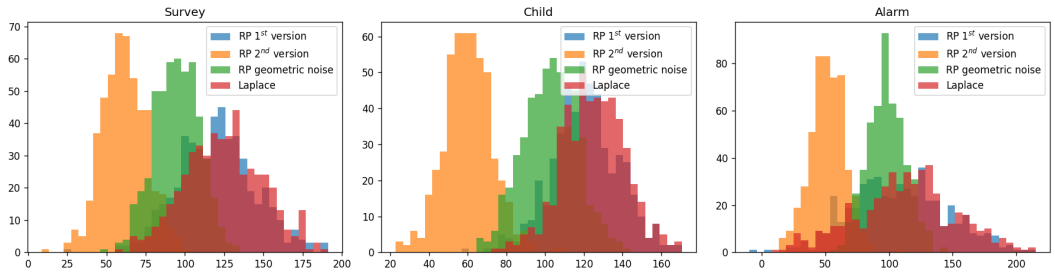


Figure 5.7: Randomized response $\alpha-$geometric mechanism, modified randomized response, randomized response, and Laplace noise performance metric histograms (`Survey` dataset on the left, `Child` in the middle, and `Alarm` on the right) over the resulting collection of the noisy contingency tables. The average of each performance metric for each data set is reported in Table 5.12

remaining attributes to emit fake values, thus maximizing the utility and strong $\epsilon-$differential privacy.

5.6 Conclusion

In this chapter, we proposed two modifications to our proposed randomization protocol. The first one uses the advantages of a Bayesian network to generate the full-dimensional distribution of data from a lower number of dimensions. Using our modified version of the randomization protocol, we use this full-dimensional distribution to generate noise in the attribute-parents pair. The Bayesian network is itself generated in a differentially private manner.

In the second part of this chapter, we propose a hybrid mechanism by combining our randomization protocol with the $\alpha-$geometric mechanism. We use this randomized response geometric mechanism to generate fake values in the Monte Carlo simulation. Both mechanisms present strong privacy guarantees while maximizing the utility of the perturbed datasets.

Chapter 6

Bayesian Networks for Fuel Prediction and Reduction in Public Transportation

In this chapter, We exploit the use of a controller area network (CAN-bus) to monitor sensors on the buses of local public transportation in a big European city. The aim is to advise fleet managers and policymakers on how to reduce fuel consumption so that air pollution is controlled and public services are improved. We deploy heuristic algorithms and exhaustive ones to generate Bayesian networks among the monitored variables. The aim is to describe the relevant relationships between the variables, to discover and confirm the possible cause–effect relationships, to predict the fuel consumption dependent on the contextual conditions of traffic, and to enable an intervention analysis to be conducted on the variables so that our goals are achieved. We propose a validation technique using Bayesian networks based on Granger causality: it relies upon observations of the time series formed by successive values of the variables in time. We use the same method based on Granger causality to rank the Bayesian networks obtained as well. A comparison of the Bayesian networks discovered against the ground truth is proposed in a synthetic data set, specifically generated for this study: the results confirm the validity of the Bayesian networks that agree on most of the existing relationships.

6.1 Introduction

According to the World Health Organization (WHO), air pollution is the second leading cause of noncommunicable diseases, such as stroke, cancer, and heart disease, and pulmonary diseases, such as chronic obstructive pulmonary diseases and lower respiratory infections. Ambient air pollution accounts for an estimated 4.2 million deaths per year [88]. Around 91% of the world's population lives in places where air quality levels exceed WHO limits and the suggested standards for a healthy life [89, 90, 91]. Air pollution is due to particulate matter 2.5 (PM2.5), which refers to tiny particles in the air that are two and one-half microns or less in width. Studies suggest that long-term exposure to fine particulate matter may be associated with increased rates of chronic bronchitis, reduced lung function, and increased mortality from lung cancer and heart disease. Furthermore, nitrogen dioxide (NO_2) is one of the other main air-quality pollutants of concern and is typically associated with vehicle emissions. The annual EU limit for NO_2 was widely exceeded across Europe in 2017. Some 86% of these exceedances were detected at roadside monitoring locations.

The red and violet colors in the map in Figure 6.1 show the areas in which the limits were overcome multiple times in past years in European countries. Similar maps are available for the other main air pollutants. In many countries, diseases can only be significantly reduced by improving air quality. Turning air-pollution-reduction goals into policies to combat noncommunicable diseases leads to multiple benefits for the environment, economy, and health. With this work, we address these concerns by putting data science to use at the service of public policies. According to the European Environment Agency, we can reach the goal of a reduction in air pollution by monitoring and modeling air quality, collecting data using sensors on roads and on vehicles, and maintaining emission inventories. We should employ emission-control strategies to reduce the number of private transport; improve public ones; reduce their emissions; increase the use of renewable energy; and apply contingency measures, new policies, and rules that, for instance, encourage planning of more compact cities.

In this work, we employ machine learning models, specifically, Bayesian networks, to analyze sensor data installed on the buses of a public transport company in a European city. The sensors collect data about the vehicle and its use (acceleration, braking, speed, stop durations with the engine on, etc.) with some contextual information about the vehicle location (such as altitude). An analysis of the sensor data using machine learning algorithms applied

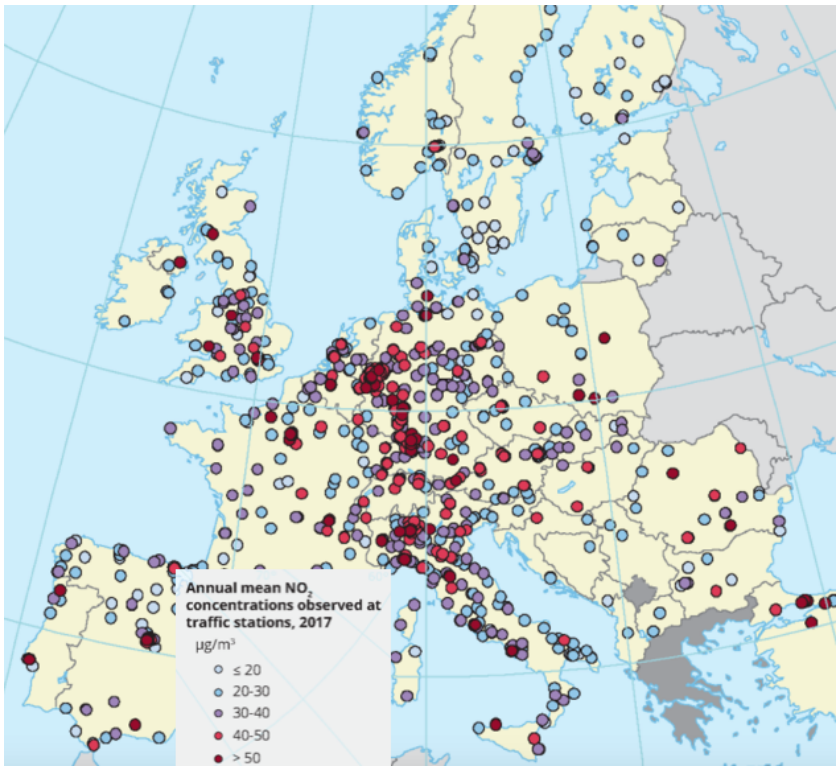


Figure 6.1: Map of locations with NO_2 emissions exceeded over the annual mean limit.

using procedures of predictive maintenance can also be used to improve vehicle equipment maintenance, with a reduction in costs due to stop times for faults and repair. Several related works exist in the literature. The application of Bayesian networks for the purposes of monitoring natural resources and applying policies we proposed in [92]. The majority of the works that monitor fuel consumption in vehicles applied predictive models. Schoen et al. in [93] adopted Artificial Neural Networks (ANN) to predict average fuel consumption in a fleet of heavy vehicles. They adopted a data summarization technique of consumption based on distance rather than time in order to eliminate the conversion of the scale for the prediction of average fuel consumption. We also apply a similar technique in this work because we build models that employ the fuel consumed per kilometer. Perrotta et al. [94] compared multiple machine learning models—support vector regression (SVR), random forest (RF), and artificial neural networks (ANN)—to predict fuel consumption in heavy vehicles. Moradi et al. [95] used multiple models in cascade and confirmed that ANN outperforms the other models. The goals of these works were to reduce costs and to obtain better routing of the fleets, even though they found it difficult to determine an accurate estimation of the fuel level. Yao et al. in [96] used smartphones to collect vehicle mobility data based on

their global positioning system (GPS) combined with data from onboard diagnostics (OBD) terminals to predict fuel consumption based on taxi drivers' driving styles. They compared ANN, SVR, and RF and showed that they all reached satisfactory prediction performances. Random forest achieved superior accuracy. Rimpas et al. in [97] selected some parameters for monitoring vehicles retrieved through the OBD-II diagnostics protocol and related them to vehicle operation and fuel consumption. They collected the proportion of oxygen in exhaust gases using a Lambda Sensor and adjusted the fuel quantity measured by a short-term fuel trim (STFT) sensor related to the immediate change in fuel flow and used it as a proxy of the accelerator pedal pressed by the driver. They collected the airflow as measured by a mass air flow sensor (MAF) as a measure of engine malfunction, a vehicle speed sensor (VSS), and the value of the engine coolant temperature (ECT) sensor where the coolant temperature affects engine overheating and fuel consumption. The authors in [98] quantified the uncertainty in measuring fuel consumption, both in light and heavy vehicles. They show that, in urban conditions, the uncertainty reaches 7%. In [99], the authors considered the prediction of fuel consumption in public buses using a multivariate data set including several explanatory variables. They compared RF, gradient boosting (GB), and ANN. Based on their analysis, RF produces a more accurate prediction compared to both GB and ANN. In [100], the authors included weather variables for the task of fuel prediction and considered them useful for an accurate prediction. Quite often in the above studies, the sample vehicles (in terms of make, model, and age) were comparable so that the type and status of the vehicle do not influence fuel consumption. We made a similar choice in selecting heavy vehicles (buses of the same model, type, mass, length, and age).

In this work, we used sensors with the sole purpose of collecting data about fuel consumption and monitoring the drivers' usage of the bus's resources (fuel, breaks, acceleration, and air conditioning). The goal was to monitor fuel consumption and its contextual conditions with the ultimate objective to provide a descriptive and explainable model of the variables that influence and cause fuel consumption and that ultimately produce air pollution. We employed Bayesian networks that permit us to afford a unique model with multiple tasks: a description with a graph of the dependence relationships between the variables, identification of the variables that are independent of the target, selection of the variables that have an impact on the target, quantification of the amount of impact on the target, prediction of the target, simulation of the variables in a scenario, and intervention in the scenario by changing

some of the variables.

The first contribution of this work is to provide a public data set [101] on sensors installed on board public transports with information about vehicle usage and fuel consumption. Sensors communicate their measures via the controller area network (CAN-bus), a specialized internal communications network that interconnects components inside a vehicle [102]. CAN is a robust vehicle standard designed to allow micro-controllers and devices to communicate with each other's applications without a host computer. It is a message-based protocol, originally designed for multiplex electrical wiring within automobiles, but it can be applied to many other contexts. For each device (sensor and actuator), the data in a frame are transmitted sequentially. Thanks to this, the vehicle turns out to be an advanced, computerized control system available on board and capable of sensor data storage.

Thanks to the collected data, we assessed the sensor outcomes to support decision making. We employed Bayesian networks (BN) as an essential tool that is able to provide descriptive and explainable models of the relationships between the monitored variables, and dependence relations that might also represent the cause-effect relationships [103]. In fact, BN captures the independence and the conditional independence among the variables: in a BN, we represent variables with nodes and dependence relationships with edges. The presence of a path connecting a variable V with a target variable T makes it clear that we should change the values of V in order to modify the values of the target (query) variable T. Instead, a change in variables not connected within a path including the target should not cause any effect on it. The main contribution of this work is to provide a BN on the variables monitored by sensors connected to CAN-bus. These BN show which variables we should change to control the fuel consumption variable. Furthermore, BN also supports the simulation of the behavior of the system.

BN is employed also to perform an assessment of the observed phenomena and to perform an intervention analysis on the causal variables so that the monitored target can be improved. As a result, we can provide the results and suggestions to drivers and policy-makers with the goal of improving air quality and reducing the costs of fuel. This is the third contribution of this work. One of the main results of this intervention analysis is to show that a change in the vehicle paths (longer but with a reduced slope) turns into a decrease in fuel consumption. Other results concern the quantification of the impact on fuel consumption of air conditioning and brake usage.

The main difficulty with BN is that the search space of the possible alternative models increases in a super-exponential way in the number of variables (graph nodes) [104]. Therefore, it is customary to employ approximate algorithms [105, 80, 106] driven by heuristics that are used to rank and evaluate the alternatives. The results are that the algorithms might converge to different and suboptimal solutions but in tractable times. Their results, as we experienced and shown in this work, might differ. In this paper, we deal with some representative algorithms for BN synthesis from data that are popular in the BN community [80, 106]. We use the BIC score [12], a derivation of the likelihood of the data under the assumed BN model, as a heuristic to evaluate the alternative networks. We revised them and compared their solutions on the sensor data by providing a brute-force alternative. Brute force converges to the global optimum of the BIC score within the search space. The brute force alternative is possible (provided the number of variables is kept limited to some units) thanks to the opportunity that high-performance computing gives us. It makes the workload efficient by distributing the computation among multiple servers and CPUs and their execution in parallel. This is the fourth contribution of this work and one of the novelties of our approach: a comparison of the results of different algorithms for BN generation from data that allows us to rank them and evaluate how closely they reach the overall optimum of brute force. This is not so common in the BN community, since BNs are usually initially provided by domain experts and later validated against evidence from data [107, 108]. To overcome the discrepancies among BNs, we compared and ranked them by proposing and adopting an alternative method: Granger causality [109]. This is one novelty of our approach and last, but not least, the contribution of our work. Granger causality and its statistical test employ vector auto-regression (VAR) as a tool to predict the target in time with the aid of multiple variables (the variables that are in the pathway from causes to effect). In its essence, the statistical test in Granger causality method verifies that the prediction of the target, with the aid of the cause variables, is better than without them. Applying this latter criterion is possible only when the flow of values of these variables is stored in time. Granger causality is commonly judged as weaker than the stricter principle of probabilistic dependency between cause and effect. With Granger causality, the existence of a causality relation between the cause and the effect is verified only in time thanks to the ability of the cause to predict and anticipate the effect in time [110, 111].

6.2 Materials and Methods

6.2.1 Sensor Data

The data set was collected by sensors on a fleet of bus vehicles. (The data set contains records for a fleet of 24 vehicles over 43 dates comprising dates between January and August 2019. It is publicly available at [101]). Sensors from the onboard diagnostics (OBD) interface collect kinematic variables such as speed, acceleration, engine speed (RPM), load (mass), and road grade. For each vehicle, the sensors perform measurements during a path from departure to the arrival bus stop; thus, the data are not sampled regularly according to time. We have a collection of multiple path records for each date on which the vehicle is driven. (For each vehicle and date, the number of records generally comprises between 100 and 400 units). The variables measured during the path include the physical properties of the travel (path length, duration, and change of height), time variables (time intervals spent coasting, braking, or in motion), and the fuel consumption of the vehicle during these time intervals. Unfortunately, we could not include the weather condition and the road type of the tracks, even if we assumed that, in the domain of public transportation, the road type is almost always metropolitan.

Our work aims to apply Bayes networks and Granger causality to study the causal association between variables, especially on fuel consumption.

6.2.2 Feature Selection and Construction

We constructed a set of representative features over which we performed our experiments.

- **Original feature collection** can be grouped as follows:
  - Path variables
    - HDIFF (m): difference in altitude between departure and arrival bus-stop
    - DIST (m): distance covered during the travel
    - MASS (kg): mass of vehicle and passengers
  - Time Interval variables (s)
    - TMTOT: total time of the travel
    - TMAIR: time with air-conditioning on

        * TMCOAST: time spent coasting
        * TMBRAKE: time spent using the brakes
        * TMMOTION: time spent with the vehicle in motion

          From this variables we can derive:

          · TMTRACTION: TMMOTION − TMCOAST − TMBRAKE: time spent in traction, that is, pressing the accelerator pedal

          · TMSTOP: TMTOT − TMMOTION: time spent with the stopped vehicle with the engine on

    – Fuel consumption variables (mL)
        * FUELSTOP: fuel consumption in TMSTOP time
        * FUELMOTION: fuel consumption in TMMOTION time

- **New features collection** is constructed as follows:

  - avg_slope (%): HDIFF/DIST
  - mass (ton): MASS/1000
  - brake_usage (%): (TMBRAKE − TMCOAST)/TMTOT
  - air_cond_ptime (%): TMAIR/TMTOT
  - stop_ptime (%): TMSTOP/TMTOT
  - fuel_per_km (l/km): (FUELSTOP + FUELMOTION)/DIST
  - accel (m/s^2): $2 \times$ DIST/(TMMOTION $\times$ TMTRACTION)

Concerning the derived variables, we can state the following:

- In the data set, buses travel at all different lengths and durations. We chose to divide the total fuel consumption by distance to perform a better comparison among buses traveling at different lengths. For the same reason, we decided to normalize all of the time variables involved in the analysis so that they represent a fraction of the total travel time;
- The variable *brake_usage* was created as an indicator of the good practice of choosing coasting instead of braking. This variable is negative when the time spent coasting is greater than the time spent braking, zero when these fractions are equal, and positive otherwise;

- The variable *stop_ptime* includes only the idling time, that is, the time spent with the vehicle not in motion but with the engine on;

- The variable *accel* is obtained as the result of a simplified model of bus travel. We assume that the bus travels starting, at time 0 with $v_0 = 0$. We assume that the velocity increases linearly with a constant acceleration (that is presumed to be an important variable for the prediction of fuel consumption) until the time is equal to TMTRACTION. Then, we assume that the bus velocity starts to decrease linearly for a time equal to the sum of time spent coasting and braking, so that when time is equal to TMMOTION (i.e., TMTRACTION+TMBRAKE+TMCOAST), the final velocity turns out to again be null: $v_f = 0$. The *accel* value can be easily derived in this simplified model by observing that the length of the travel is equal to the area of the velocity graph in a velocity-time diagram or, more formally, by solving

$$\begin{cases} s = \frac{1}{2}a_1{t_1}^2 + v_1(t_2 - t_1) + \frac{1}{2}a_2(t_2 - t_1)^2 \\ v_1 = a_1 t_1 \\ 0 = v_1 + a_2(t_2 - t_1) \end{cases}$$

  where s is the traveled distance, t_1=TMTRACTION, t_2=TMMOTION, v_1 is the velocity at time t_1, a_1 is the positive acceleration we are looking for, and a_2 is a negative acceleration (not involved in fuel consumption).

We show the main statistics (mean, standard deviation, and min-max range) of the new feature collection for the data set on which we perform our experiments in Table 6.1. We can observe that the mass of the vehicle is around 20 tons, that the path is generally on the plain ground (from the mean *avg_slope*), and that the fuel consumption is around 0.6 L per km.

6.3 Algorithms for Bayesian Network Learning

In this section, we outline some of the algorithms to learn causal models from the observed data. Learning a Bayesian network occurs in two steps: *structure learning* and *parameter learning*. Suppose that learning a BN with DAG $\mathcal{G}$ and parameters Θ from a data set $\mathcal{D}$

	Mean	Std	Min	Max
avg_slope (%)	0.00	0.02	−0.30	0.21
mass (ton)	21.19	1.55	17.92	29.85
aircond_ptime (%)	0.0	0.2	0.0	1.0
stop_ptime (%)	0.19	0.15	0.01	0.97
brake_usage (%)	0.20	0.09	−0.06	0.72
accel (m/s^2)	0.36	0.21	0.01	1.80
fuel_per_km (l/km)	0.57	0.20	0.02	3.93

Table 6.1: Statistics of the data set features: mean, std (standard deviation), and minimum and maximum feature values.

having n observations is driven by the following:

$$P(\mathcal{G}, \Theta \mid \mathcal{D}) = P(\mathcal{G} \mid \mathcal{D}) \cdot P(\Theta \mid \mathcal{G}, \mathcal{D})$$

Structure learning is involved in learning $P(\mathcal{G}|\mathcal{D})$: it aims to find the DAG $\mathcal{G}$ that incorporates the dependence structure between the variables of the data $\mathcal{D}$. In contrast, parameter learning is focused on $P(\Theta|\mathcal{G}, \mathcal{D})$ and consists of estimating the parameters Θ given $\mathcal{G}$. Suppose that the parameters are independent in distributions; then, they can be learned in parallel for each node X_i as follows:

$$P(\Theta \mid \mathcal{G}, \mathcal{D}) = \prod_{i=1}^{N} P(\Theta_{X_i} \mid Pa_i, \mathcal{D})$$

where, with Pa_i, we represent the set of parent nodes of X_i (connected with a directed edge, incoming in X_i) and, with Θ_{X_i}, we represent the set of parameters of the conditional distribution of X_i given its parents Pa_i in $\mathcal{G}$. Learning the structure of BN is an NP-hard problem and computationally challenging. Suppose there are N nodes; then, the possible arcs are $N(N-1)/2$, and the number of DAGs grows super-exponentially as the number of nodes N increases. Hence, only a small number of the possible alternative DAGs can be explored in a reasonable time. There are three main possible approaches used in the structure learning of the BN: *score-based, constraint-based, and hybrid*. Each is based on a different statistical criterion.

6.3.1 Learning the Structure: Score-based Methods

Score-based approach is a general class of optimization techniques to learn BN structure.

Each learned BN is assigned a network score based on its *Goodness-of-Fit*; the algorithm then tries to maximize the network score. Assuming a BN structure $\mathcal{G}$ on data set $\mathcal{D}$, its score is

$$Score(\mathcal{G}, \mathcal{D} = P(\mathcal{G}|\mathcal{D}),$$

Score-based methods try to maximize this score. The above computation can be re-written using Bayes' law:

$$P(\mathcal{G}|\mathcal{D}) = \frac{(\mathcal{D}|\mathcal{G})P(\mathcal{G})}{P(\mathcal{D})}$$

To maximize the BN score we only need to maximize $P(\mathcal{D}|\mathcal{G})P(\mathcal{G})$, since $P(\mathcal{D})$ only depends on the data set. The network score can be the *Bayesian Information Criterion* [112] (BIC) or *Akaike Information Criterion* [113] (AIC) for both discrete and continuous data sets. BN with lower BIC are generally preferred, but it highly depends on the development environment: In R programming language, higher BIC is a criterion for model selection. BIC is defined as

$$BIC = k\ln(n) - 2\ln(L)$$

where L is the *likelihood function* which maximize the model $\mathcal{G}$ value, given as $L = P(X|\Theta, \mathcal{G})$, where Θ maximize the *likelihood function*. X is the observed data, n is the sample size, and k is the number of parameters calculated by the model. Score-based approach examples include *simulated annealing, greedy search* [82], *genetic algorithms* [81], and *hill climbing* (HC) [80].

Example 6.3.1. *For example, consider a data set containing 6 attributes* $(\mathtt{A}, \mathtt{S}, \mathtt{E}, \mathtt{O}, \mathtt{R}, \mathtt{T})$ *and the global probabilities' distribution is given as*

$$P(\mathtt{A}, \mathtt{S}, \mathtt{E}, \mathtt{O}, \mathtt{R}, \mathtt{T}) = \mathtt{P(A)P(S)P(E|A,S)P(O|E)P(R|E)P(T|O,R)}$$

Then BIC can be calculated using

$$BIC = \ln P(\mathtt{A}, \mathtt{S}, \mathtt{E}, \mathtt{O}, \mathtt{R}, \mathtt{T}) - \frac{\mathtt{k}}{2}\ln(\mathtt{n}) = \left[\ln \mathtt{P(A)} - \frac{\mathtt{k_A}}{2}\ln(\mathtt{n})\right] +$$
$$\left[\ln P(\mathtt{S}) - \frac{\mathtt{k_S}}{2}\ln(\mathtt{n})\right] + \left[\ln P(\mathtt{E}|\mathtt{A},\mathtt{S}) - \frac{\mathtt{k_E}}{2}\ln(\mathtt{n})\right] + \left[\ln P(\mathtt{O}|\mathtt{E}) - \frac{\mathtt{k_O}}{2}\ln(\mathtt{n})\right] +$$
$$\left[\ln P(\mathtt{R}|\mathtt{E}) - \frac{\mathtt{k_R}}{2}\ln(\mathtt{n})\right] + \left[\ln P(\mathtt{T}|\mathtt{O},\mathtt{R}) - \frac{\mathtt{k_T}}{2}\ln(\mathtt{n})\right]$$

Where k is the number of parameters in the BN, n is the sample size, and $d_{\mathrm{A}}, d_{\mathrm{S}}, d_{\mathrm{E}}, d_{\mathrm{O}}, d_{\mathrm{R}}, d_{\mathrm{S}}, d_{\mathrm{T}}$ *are the number of parameters associated with each node.*

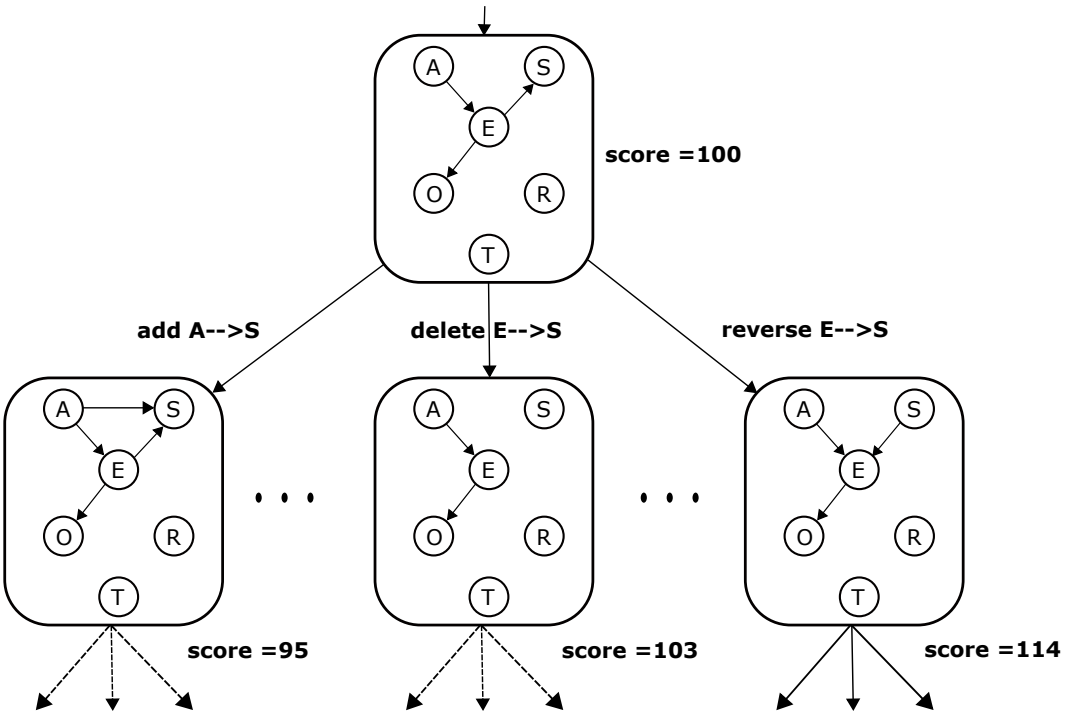


Figure 6.2: Illustration of a Bayesian network hill-climbing search procedure

6.3.2 Learning the Structure: Constraint-based Methods

In constraint-based methods, constraints are imposed on the model based on conditional independence statements. BN structures may also entail non-independence-based constraints where the latent variables exist [114]. Although conditional independence tests are used for statistical tests on data set, these tests can also be used to reconstruct the structure under several assumptions, including *Causal Sufficiency, Faithfulness*, and *Causal Markov*. With these assumptions, one can determine the edge between two nodes, or the direction of that edge.

In constraint-based algorithms, the existence of an edge between two variables X_1 and X_2 is tested using a number of conditional independence tests. Each of these conditions on a distinct subset of $\mathcal{W} - \{X_1, X_2\}$. If there exists an edge $X_1 - X_2$ and faithfulness hold, then all these independences tests should also fail. If no edge exists between X_1 and X_2, then there must exist a d-separating subset. Suppose there is no direct edge between X_1 and X_2 in the true structure, one of the subsets is the set of parents of one of the nodes. The

algorithm attempts all possible subsets of $\mathcal{W} - \{X_1, X_2\}$ to find the existence of an edge between every pair of variables in the domain. After the undirected edges are determined in the structure, the algorithm determines the directionality of these edges. The directionality of edges is performed by examining triples of variables X_1, X_2, and X_3, such that there is an edge between $X_1 - X_3$ and $X_2 - X_3$ but no $X_1 - X_3$. If X_1 and X_2 are not conditional independent given $\mathbf{S}$ such that $X_1 \not\perp\!\!\!\perp X_2 \mid \mathbf{S}, \forall \mathbf{S} = X_3 \cup \mathbf{S}, \mathbf{S}' \subseteq \mathcal{W} - \{X_1, X_2, X_3\}$, then the directionality of $X_1 - X_3$ and $X_2 - X_3$ is $X_1 \rightarrow X_2$ and $X_2 \rightarrow X_3$, respectively. The algorithm repeats the same process for all the triples variables and propagates the edge direction while maintaining acyclicity.

6.3.3 Learning the structure: Hybrid methods

Hybrid approaches are constraint-based and use restriction to reduce the candidate space of DAGs; they are score-based and use *maximize* implementations to find the optimal DAG in the restricted space by implementing any combination of constraint-based and score-based algorithms. Hybrid approaches include *Max-Min Hill Climbing algorithm* (MMHC) [105], *Restricted Maximization* (RSMAX2) [115], and *Hybrid HPC* (H2PC) [116]

6.3.4 Hill Climbing Algorithm

The hill climbing algorithm belongs to the class of greedy search algorithms. Hill climbing (HC) assigns a network score (*Goodness-of-Fit*) to the candidate BNs, and heuristic algorithms strive to maximize the network score since a higher value means a better fit. HC starts from a DAG structure, and then it adds, reverses, and deletes arcs until the network score no longer improves [80]. The network score can be the *Bayesian Information Criterion* [112] (BIC) or *Akaike Information Criterion* explain in section 6.3.1 [113] (AIC) for both discrete and continuous data sets. The graphical representation of the hill-climbing algorithm is shown in figure 6.2

Restrictive Maximization Algorithm

The restrictive maximization algorithm belongs to the class of hybrid approaches. RM achieves faster structure learning by restricting the search space and by implementing a combination of constraint-based and score-based algorithms [115].

Brute Force Algorithm

In this work, we introduce the brute force algorithm to afford the computational complexity of the complete exploration of the search space of the possible BN alternatives. We take advantage of the parallel computing technology provided by HPC4AI (Turin's High-Performance Centre for Artificial. Intelligence[1] The brute force formalization and implementation are one of the original contributions of this work. We split up the search space for model selection and assign each to an independent processor that delivers the best BN of the corresponding subspace. Finally, these results are compared to choose the very best model. Each candidate BN is assigned a network score "*Goodness-of-fit*". The brute force algorithm returns a BN with the maximum score since a higher score means a better fit. We used a score derived from the Bayesian information criterion (BIC) as implemented in the R Library [117]; this network score is suitable for both continuous and discrete data sets.

The idea of the brute force algorithm is to partition the space for all possible Bayesian networks and to allocate each partition to a different processor, such that each processor in parallel executes the task to evaluate the BIC score of all networks in its partition. Each network is represented as a vector—a binary configuration of as many bits as possible arcs in the networks. Each bit in the vector represents whether the corresponding arc is present or absent in the network.

The algorithm starts with an input data set $\mathcal{D}$ containing N variables. $AllArcs$ is a matrix (p$*$2) with p $= N(N-1)/2$ being the number of possible (undirected) arcs. Each row in the matrix represents an arc (from-to): the first column represents the starting node, and the second represents the ending node. Each pair of nodes is identified by a matrix row index from 1 to p.

$k_1 <$ p is the number of arcs that are actively considered by each processor, and the processor is free to vary in any way in combination with the remaining arcs that instead are fixed. The different processors have different configurations in present/absent arcs fixed in the remaining subset of $p-k_1$ arcs. $FixedArcsPresence$ is a vector of length $p-k_1$ containing information related to the arcs for which the presence/absence is fixed for that processor. $p-k_1$ is the prearranged arcs (or pairs of nodes). In total, we have 2^{p-k_1} available processors. Each processor runs the brute force algorithm 5, with $FixedArcsPresence$ as an input argument. $FixedArcsPresence$ is a vector of the ordered list representing the presence/absence

[1] https://hpc4ai.it/

of each of the prearranged $p-k_1$ arcs. Each element in this vector corresponds to a different nodes pair with values 0, 1 such that $FixedArcsPresence[i] = 1$ if the ith pair of nodes is considered by that processor to be connected; otherwise, it is 0. The processors are executed in parallel, where each processor has a different realization of $FixedArcsPresence$. For each total configuration of arcs present or absent, from the fixed part and the variable part, the processor evaluates the BIC score of the corresponding Bayesian network with the goal of finding the one with the maximum value. Regarding the determination of the arcs' directions, defined within the algorithm, it establishes whether each arc is oriented according to the direction taken as the reference in such a matrix or the other way around and evaluates the BIC score for both arc directions. In the end, the maximum score among the scores found by the processors is selected and so is the corresponding Bayesian network.

However, some care should be taken when Bayesian networks are learned from the data. It should be kept in mind that networks learned from observational data may establish some relationships that are hard to explain based on our prior knowledge of the domain. Some relationships may reveal aspects of phenomena that we did not expect, some may be explained by introducing exogenous variables acting as confounders, and the influence of variable X_i on another variable X_j may be mediated by an unobserved variable X_l that is not included either in the model or in the available data. Moreover, we should take into account that the model, due to a lack of flexibility, could be unable to accurately describe the phenomenon. For example, the assumption of Gaussianity may be inadequate for our data, and adapting the variables to multinomial assumptions through discretization may lead to mutual information loss.

Therefore, we cannot expect to find a rational justification for each connection but we can apply critical thinking to extract helpful insights based on what the data supports.

6.4 Granger Causality

We performed a model evaluation of the Bayesian networks employing the statistical concept of Granger causality that applies to the time-series domain [109]. In the following, we provide a formalization of the application of the concept of Granger causality to the evaluation of Bayesian networks. Later, in Section 6.5.2, we apply this method to the task of ranking and comparing Bayesian networks, resulting from the application to the same data of different, approximate, and heuristic-driven algorithms. The provided solution can solve

a data analyst's uncertainty about the choice among them. These concepts, to the best of our knowledge, are original in their application to the validation of Bayesian networks and

in their formalization.

Algorithm 5 Brute force algorithm for learning the best Bayesian network.

Input: $\mathcal{D}$, $AllArcs$, k_1, and $FixedArcsPresence$
Output: bestG, bestscore
Initialization:
    bestG ← the current best DAG, initially empty ;
    bestscore ← score of the current best DAG (initially empty) ;
end
For index($base10_1$) **from 0 up to** ($2^{k_1} - 1$) **do:**
    Initialization:
        $Structure$ ← binary vector of length k_1, initially empty, such that each element represents an undirected arc
        ratio←$base10_1$
    end
    For index i from k_1 **down to 1 do:**
        ▷ build the vector defining whether each arc is present or absent.
        $Structure[i] \leftarrow ratio$ mod 2
        $ratio \leftarrow ratio$ divided by 2 but rounded down (integer divide)
    end
    $Structure$ ← concatenate($Structure$,$FixedArcsPresence$) ▷ append the digits associated with the fixed arcs to the end of $Structure$
    $PresentArcs$ ← the ids of the present arcs (the indexes i such that $structure[i] = 1$)
    ▷ set the arcs in the network according to their default reference direction determined by the matrix $AllArcs$
    $Arcs \leftarrow AllArcs[PresentArcs, \cdot]$
    $k_2 \leftarrow$ length($PresentArcs$)
    If $k_2 > 0$**:**
        For $base10_2$ **from 0 up to** $2^{k_2} - 1$ **do:**
            $ratio \leftarrow base10_2$
            If $k_2 > 1$ **then:**
                $Directions$ ← binary vector of length k_2, initially empty, such that each element represents the direction of the corresponding arc conditioned on its presence
                For i from k_2 **down to 1 do:**
                    ▷ build the vector determining the orientation of each arc
                    $Directions[i] \leftarrow ratio$ mod 2
                    $ratio \leftarrow ratio$ divided by 2 but rounded down (integer divide)
                    If $Directions[i]$**=1 then:**
                        change the orientation by swapping the two elements of $Arcs[i, \cdot]$
                        ▷ else leave the direction as it is
                    end
                end
                If the graph defined jointly by the matrices $Arcs$ **and** $Directions$ **is a DAG then:**
                    build the corresponding network G
                    score ← score(G)
                end
                If score>bestscore:
                    bestdag ← G;
                    bestscore ← score
                end
            end
        end
    end
end
Return bestG, bestscore

Granger Test

Given a stationarized multivariate time series $t^{(l)}$, including variables A and B, we want to establish if $A \overset{(g)}{\Rightarrow} B$. (The time series must satisfy the stationarity condition that is assessed for each feature by the Augmented Dickey Fuller test (ADF) with significance level 0.05. If the S_i feature gives rise to a time series that is not stationary, we iteratively apply first-differencing $S_i(t) \rightarrow [S_i(t) - S_i(t-1)]$ and repeat ADF until we reach stationarity.) Notation $A \overset{(g)}{\Rightarrow} B$ denotes that A Granger-causes B. We performed a Granger test by comparing the two auto-regressive models:

$$B_t = \sum_{l=1}^{q} \beta_l B_{t-l} + \epsilon_t \tag{6.1}$$

$$B_t = \sum_{l=1}^{q} \beta_l B_{t-l} + \sum_{l=1}^{q} \alpha_l A_{t-l} + \epsilon_t \tag{6.2}$$

The Granger test is an F-test with null hypothesis $H_0 := \{\alpha_l = 0; \quad l = 1, \dots, q\}$. The success of the test implies that $A \overset{(g)}{\Rightarrow} B$, that is, A has a predictive power on B since its lag coefficients α_l in the second auto regressive model are significantly different from 0. For our multivariate time series $t^{(l)}$, we conducted a Granger test for each possible ordered distinct pair from feature set $\{S_1, S_2, \dots, S_N\}$. Here, we denote the variables with symbols S_i because we want to highlight that each gives rise in a vehicle to a set of time series (one for each vehicle in time). We stored the result of the $t^{(l)}$ tests in a Granger matrix $G^{(l)}$:

$$G^{(l)}_{i,j} = \begin{cases} 1 : S_i \overset{(g)}{\Rightarrow} S_j \text{ for } t^{(l)} \\ 0 : else \end{cases} \tag{6.3}$$

We fit a vector auto regressive model (VAR) over the time series $t^{(l)}$, and the maximum-lag order q was selected automatically according to the AIC criteria. We then performed a Granger test for each pair (S_i, S_j). The related F-test was performed with a significance level of 0.1. We made this choice because we observed that, with a level of 0.05, we have an average decrease of 5% in the rate of success of the Granger test over the set of time series of the experiments.

We frame our data set as a collection of multivariate time series $\{t^{(l)}; \quad l = 1, \dots, T\}$. We performed the Granger test for each time series $t^{(l)}$ and for each ordered variable pair (S_i, S_j). The whole test results were then stored as a collection of Granger matrices $\{G^{(l)}; \quad l =$

$1, \ldots, T\}$. This collection is successively used for validating the Bayes networks.

6.5 Results

6.5.1 The Discovered Bayes Networks

We illustrate the Bayes networks discovered from the algorithms introduced in Section 6.3 applied on the data set with the constructed features described in Section 6.2.2. We initially conduct an analysis on the found relationships between data set features based on our knowledge about the data set domain: public transportation. We then introduce the diverse applications of Bayes networks such as feature selection for a supervised prediction task and intervention analysis in order to perform decision making.

Bayesian Networks Analysis

We name the discovered Bayesian networks after the algorithms introduced in Section 6.3 as employed for their construction:

- HC: Hill climbing
- RM: Restrictive Maximization
- BF: Brute Force

We group the collection of links found by the networks in Table 6.2 as follows:

- Common Links (CL)

  We have 10 links on which the three networks agree both on presence and direction, identifying reasonable dependencies. Specifically, we have that *brake_usage* and *accel* are both caused by *avg_slope* and *stop_ptime*: this can be interpreted with the fact that a steep path in which we have to slow down or stop, if necessary, implies more frequent use of the brakes or, conversely, in order to ride up a steep path, to use the accelerator pedal. *fuel_per_km* is caused by variables *avg_slope*, since a steeper path causes a greater consumption; *stop_ptime*, since the fuel consumption of a stopped vehicle and the engine still on is higher; *brake_usage*, since more frequent use of the brakes is related to a greater *stop_ptime*; *mass*, since a heavier vehicle (full of passengers) requires greater fuel consumption; and *aircond_ptime*, since air-conditioning

is expensive in terms of fuel. Moreover, we have that *aircond_ptime* is caused by *mass*; this can be explained by the fact that a higher mass implies a greater number of persons, which increases the temperature within the vehicle and requires the use of air-conditioning.

- Common Links with Discordant Direction (CLDD)

  We have three variable pairs on which the three networks agree on the link presence but are discordant on the direction. We discuss the relationship for each of the three related variable pairs.

  For *accel* and *brake_usage*, we think that more frequent use of the brakes implies subsequent use of the accelerator: according to this, *brake_usage* causes *accel*, as stated by HC and RM. BF states the opposite, and it seems reasonable that the use of the accelerator may lead to successive use of the brakes for decreasing the speed. We cannot infer the actual direction of the causal relationship between the two variables without auxiliary information concerning the traffic condition and driving behavior. Unfortunately, the data set does not contain these features. We only have a proxy of these conditions from *stop_ptime* and *brake_usage*.

  For *accel* and *fuel_per_km*, we retain that more frequent use of the accelerator causes a higher consumption, so *accel* causes *fuel_per_km*, as stated by HC and BF (while RM states the opposite).

  For *mass* and *brake_usage*, we retain that a higher mass of the vehicle implies a higher probability that someone on the bus requires leaving the vehicle, which follows the requirement that the bus driver needs to brake and to: so *mass* causes *brake_usage*, as stated by HC and RM (while BF states the opposite).

- Uncommon Links (UL)

  [-30]We have four links for which the three networks do not agree, both on presence and direction.

  For the variable pair *mass* and *accel*, we think that a heavier vehicle requires more frequent use of the accelerator to reach its destination, so *mass* causes *accel*, as stated by HC, while BF states the opposite and RM does not find a relationship.

  The causal relationship between *mass* and *stop_ptime* is found only by HC. It appears

reasonable and in agreement with the fact that *mass* causes *brake_usage*.

The causal relationship between *slope* and *mass* is found only by BF. This link is more difficult to interpret; maybe the dependence between slope and mass may be explained by the fact that a low value of slope may be a proxy for identifying a crowded region of the town, where more people get on the bus and therefore the mass increases.

From our considerations, we observe that the links found by the networks can be explained with arguments concerning the domain of public transportation. Moreover, we notice that, for links with a discordant network direction (CLDD), a feedback link over a variable pair may exist though it is not contemplated by the DAG structure found by the Bayes network algorithms. That is, given a variable pair and a network construction algorithm, we find a directed causal relationship that may not be the only one in the considered domain. For example, we are uncertain on the causal direction for the pair (*brake_usage* and *accel*). Indeed, excessive acceleration may lead to the use of the brakes and the use of brakes ensures the later use of the accelerator during the same bus path. Therefore, the true relationship between this pair may be a feedback link (a cycle). Unfortunately, we know that the network construction algorithm excludes the formation of loops and it will never be found.

	CL	**CLDD**	**UL**
1	avg_slope → brake_usage	accel → brake_usage (BF)	mass → accel (HC)
2	stop_ptime → brake_usage	brake_usage → accel (HC, RM)	mass → stop (HC)
3	avg_slope→ accel	brake_usage → mass (BF)	accel → mass (BF)
4	stop_ptime → accel	mass → brake_usage (HC, RM)	slope → mass (BF)
5	avg_slope → fuel_per_km	fuel_per_km → accel (RM)	
6	stop_ptime → fuel_per_km	accel → fuel_per_km (HC, BF)	
7	brake_usage → fuel_per_km		
8	mass → fuel_per_km		
9	aircond_ptime → fuel_per_km		
10	mass → aircond_ptime		

Table 6.2: Link collection found by the three Bayes networks: hill climbing (HC), restrictive maximization (RM), and brute force (BF). The collection is grouped as common links (CL), common links with discordant direction (CLDD), and uncommon Links (UL). For CLDD and UL, we specify the networks for which the links are present.

These observations highlight the difficulty of Bayes networks in determining the causal direction between variables, which may be chosen after network construction with the aid of the expert knowledge.

Concerning fuel consumption, we observe that all of the networks agree on the causal re-

lationship of the variables (*brake_usage*, *avg_slope*, *air_cond_ptime*, *stop_ptime*, and *mass*) over *fuel_per_km* while the reasonable relationship of *accel* causing *fuel_per_km* is found by HC and BF. RM states the opposite relationship, which we consider inexact.

In Figure 6.3, we show the details of the BN that might have been extracted if the presence of a latent variable on the typology of the location (e.g., downtown) were not left as a latent information but were explicit. As a consequence of the presence of an unobserved variable, which is a common cause of other variables (*mass* and *avg_slope*), we observe the situation on the right, with a possible mutual link between the effects that are not easily explained alone. This common situation is recognized also in the literature [118], and the BN are deemed as equivalent from the viewpoint of the algorithms (but not by the experts).

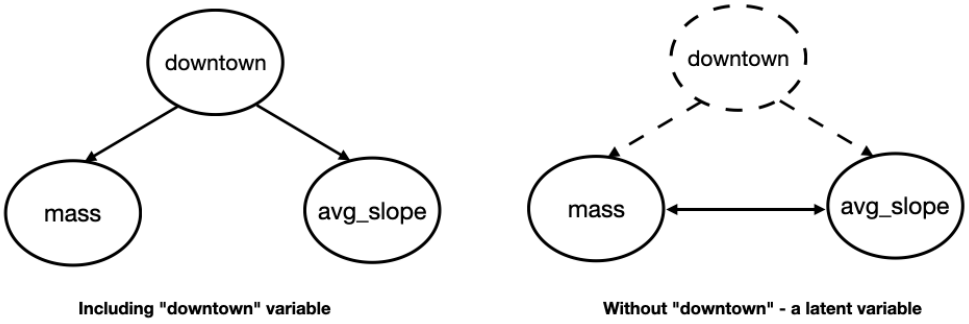


Figure 6.3: Detail of the uncertainty in Bayesian networks due to the presence of a latent variable.

Feature Selection and Target Prediction with Bayesian Networks

The Bayesian networks can be applied to perform feature selection for a given supervised prediction task; we considered multivariate linear regression of a given feature node x_v, where we say x_v is the *target*. Given the feature set $\mathcal{X} = \{x_1, x_2, \dots, x_N\}$, we define $\mathcal{X}_{-v} = \mathcal{X} \setminus \{x_v\}$ and inquire which features of $\mathcal{X}_{-v}$ should be selected to perform a regression on *target* x_v. Given a Bayes network $\mathcal{B}$, we introduce the feature set $\mathcal{P}_v^{(\mathcal{B})}$ as the set of parent nodes of x_v with respect to $\mathcal{B}$ (as an example, from Figure 6.4, we have for the Brute Force network (BF) that the parent set of *brake* node is $\mathcal{P}_{brake}^{(BF)} = \{accel, stop\_ptime, avg\_slope\}$).

We performed feature selection by choosing the features of parent set $\mathcal{P}_v^{(\mathcal{B})}$ for the regression of *target* x_v. We notice that this feature selection is feasible only when $\mathcal{P}_v^{(\mathcal{B})} \neq \emptyset$. For example, we observe from Figure 6.4 that *avg_slope* does not admit a non-null parent set for any of the discovered Bayes networks.

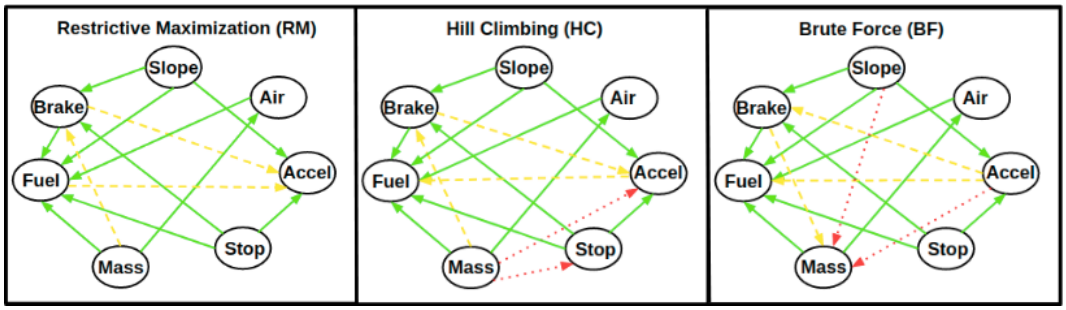


Figure 6.4: Bayes networks discovered by the algorithms (RM, HC, and BF) over the feature set *accel* (Accel), *avg_slope* (Slope), *air_cond_ptime* (Air), *brake_usage* (Brake), *mass* (Mass), *fuel_per_km* (Fuel), and *stop_ptime* (Stop). The set is described in Section 6.2.2. The green continuous arrows are common links (CL) between the networks, the yellow dashed arrows are common links with discordant directions (CLDD), and the red dotted arrows are uncommon links (UL), as presented in Table 6.2

We evaluated the performance of the target prediction using the regression model constructed with the parent feature selection. In the evaluation of the performance, we applied the 10-fold cross-validation score on the root mean squared error (*rmse*) of the regression. We then compared the 10-fold averaged *rmse* with the mean and standard deviation of the *target* feature, which can be obtained from Table 6.1. We report the *rmse* performance for each feature and for each parent set identified by our Bayes network collection in Table 6.3.

	accel	fuel_per_km	brake_usage	stop_ptime	aircond_ptime	mass
CV1	0.18 (RM)	0.13 (HC,BF)	0.07 (BF)	0.15 (HC)	0.13 (HC,BF,RM)	1.6 (BF)
CV2	0.18 (HC)	0.14 (RM)	0.08 (HC, RM)			
CV3	0.21 (BF)					
std	0.21	0.20	0.09	0.15	0.18	1.6
mean	0.37	0.58	0.20	0.19	0.04	21.2

Table 6.3: 10-fold Cross Validation (CV) averaged *rmse* score for each regression of *target* (column) on possible parent sets identified by Bayes network collection {Brute Force (BF), Hill Climbing (HC), Restrictive Maximisation (RM)}. The first three rows list the possible parent set, ordered CV scores together with the Bayes networks that generate the given parent set. Detailed information on parent set can be retrieved from Figure 6.4. Last two rows display *target* mean and standard deviation

We observe that, for each *target* feature, the CV scores tend to be of the same order of magnitude with respect to the *target* standard deviation and are generally smaller. Therefore, we can state that the target regression with respect to the parent set tends to provide reasonably low discrepancy errors. When it is possible, we can employ multiple CV scores in order to compare the Bayes networks in order to assess their ability to perform feature selection by identifying different parent sets. To take an example, if we consider target *fuel_per_km*,

the networks HC and BF have better performances with respect to RM in terms of parent feature selection for regression. In fact, HC and BF identify a parent set made by all of the features $\mathcal{X}_{-fuel\_per\_km}$ while the RM parent set does not include the *accel* feature.

In order to perform a more comprehensive study on feature selection, we compared parent set selection with the variance inflation factor (VIF) technique. VIF is a feature-selection technique [119] that has the goal of reducing multicollinearity in a multivariate data set given the feature set $S = (s_1, s_2, \dots, s_d)$.

We compute the variance inflation factors collection $S^{(VIF)} = (V_1, V_2, \dots, V_d)$ (We evaluate for each feature s_l the quantity R^2_{-l}, that is the R-squared of regression of feature s_l with respect to S_{-l} and the corresponding variance inflation factor $V_l = 1/(1 - R^2_{-l})$). We remove the feature s_m with the highest inflation factor if $V_m > 5$ (For high V_m we have that feature s_m has high collinearity with respect to the other features and has a scarce impact in the regression). We then repeat iteratively the same procedure by recomputing the VIF and removing one feature at each step of the iteration until we reach $V_m < 5$. We apply VIF on $\mathcal{X}_{-v}$ for feature target x_v but we are not able to perform feature removal since the VIF feature values are of order 10^{-2} or less, suggesting that our data set does not exhibit multicollinearity. Therefore we can reasonably use the Bayes networks as an alternative valid instrument to perform feature parent selection.

Intervention Analysis

One interesting feature of Bayesian networks is the possibility to estimate the impact of the intervention on variables using just observational data. This is an advantage because we do not need to perform costly and, in some cases, impossible experiments. We say we perform an intervention on a variable when we treat it as fixed for the whole data set. The goal of this task is to estimate the impact on the target of the action of control and to change the values on one of its causes. This is an original and valuable contribution of our work since this intervention aims to reduce fuel consumption and provides actionable knowledge as a result of sensor data analysis.

To estimate the impact of intervention without using the experimental data, we follow the approach provided in [107, 120]. For a Gaussian BN, the causal effect of X on Y is determined as follows:

- We determine the set of parents of X in the BN graph (we denote it as $Pa(X)$); it is

the set of variables directly connected to X in the graph.

- We perform a linear regression of Y on X and $Pa(X)$; it computes the target as a function of the other variables on which it depends; and
- The coefficient of X provides us with the causal effect of X on Y: each coefficient quantifies the amount of impact of each cause to the target.

[-18]Assuming the BN structure obtained using the brute force algorithm to be true and restricting our attention to `fuel_per_km` as the target variable, we obtain the variables that have an effect on the target. They are (`slope`, `mass`, `air_cond_ptime`, `stop_ptime`, `brake_usage`, and `accel`). Table 6.4 shows them together with the other variables (the adjustment set). In the determination of the contribution of each single cause to the effect, we need to maintain the values of the adjustment set in order to block-out their causal effect on the target and concentrate only on a single cause (adjustment criterion) [121]. All of these variables are included as inputs in the regression for the determination of the target; later, we consider the variation in the target as a function only of a single causal variable for the quantification of its impact on the target.

Variable	**Adjustment Set**	**Causal Effect**
slope	{ }	6.635
mass	{slope,brake_usage,accel}	0.012
air_cond_ptime	{mass}	0.107
stop_ptime	{ }	0.445
brake_usage	{slope,stop_ptime,accel}	0.206
accel	{slope,stop_ptime}	0.189

Table 6.4: Causal effects of variables on target variable `fuel_per_km`.

Table 6.4 summarizes all of the possible impacts that the variables have on the target `fuel_per_km`. This is exactly the added value of Bayesian networks compared to the usual analytical studies based on prediction models: we can forget about the impact on the target of the remaining variables that are not directly connected to the target because they cannot have a direct impact on it. The (causal) variables of the target are directly connected to it and are exactly those ones that can have an effect on it. This effect is precisely quantified by the amount called "causal effect": it measures the increase in the target for any unit of increase in the corresponding causal variable.

Using the values in Table 6.4, we can consider the following about the driving styles that are suitable to reduce fuel consumption:

- If we decrease of one unit `air_cond_ptime`, we can expect a decrease of 0.107 units in `fuel_per_km`;
- If we decrease of one unit `brake_usage`, we can expect a decrease of 0.206 units in `fuel_per_km`; and
- We can obtain similar considerations about `mass`, obtaining a decrease in `fuel_per_km` of 0.012 for each decrease in a ton of mass.

We observe that the causal effect for *avg_slope* is relatively high with respect to the other variables. This can be explained by the fact that, for the considered angle interval $(0°, 10°)$, the corresponding slope, that is the tangent of the angle, ranges in the interval $(0, 0.18)$. Then, the corresponding slope variations are of the order $10^{-1} - 10^{-2}$. Therefore, since the maximum slope variation is 0.18, that is, very small with respect to the unit value, we have that the corresponding maximum *fuel_per_km* variation is comparable to $6.635 \times 0.18 = 1.19$ l/km. This latter variation is compliant with the domain knowledge. Although the `mass` and `avg_slope` variables are not under the control of the driver, this information can still be useful. A decision-maker can use it, for example, to choose whether it is convenient to choose a path that is longer but with a lower slope. Additional considerations on this type of intervention follow.

Case Study: Intervention on Slope From the intervention analysis results, we introduce a simple case study in which we compare two paths that reach the same destination but have a different configuration. The first path has a higher slope and a lower length, while the alternative path has a decrease in slope and therefore a higher length. By the intervention on *avg_slope*, we want to study how the fuel consumption varies and if we have a fuel saving under some configuration of the parameters intervals.

Proof. Path p_i has length l_i and angle α_i; we introduce the slope of the path as the tangent of its angle: $s_i = tan\alpha_i$. ■

We formulated the fuel consumption of path p_i as $F_i = l_i f_i$, where f_i is the *fuel_per_km* consumption related to path p_i. According to the causal effect information from Table 6.4,

we assume that f_i increases linearly with the slope $s_i = tan\alpha_i$. That is, from a positive slope variation $\Delta s_i = \Delta tan\alpha_i$, we have a positive $\Delta f_i = r\Delta(tan\alpha_i)$ with $r = 6.635$. We refer to paths (1) and (2) of Figure 6.5, respectively, as p_1 and p_2. We observed that p_1 and p_2 are two possible paths for reaching the same destination (from Figure 6.5, we observe that path (2) is equivalent to path (1.a), which reaches the same destination H_1 of path (1); model (2) has a straight path to facilitate the computation of the fuel savings). We can model fuel savings as $\mathcal{R}(\alpha_1, \alpha_2) = F_1 - F_2$ between path (1) and path (2). From this formulation, we ask which values of (α_1,α_2), with $\alpha_2 < \alpha_1$, have a positive saving $\mathcal{R}(\alpha_1, \alpha_2) > 0$. Knowing that $f_1 > f_2$, since path (1) has a greater slope than path (2), we have that $f_1 = f_2 + r(tan\alpha_1 - tan\alpha_2)$. Therefore, we have the following:

$$\begin{aligned}\mathcal{R}(\alpha_1, \alpha_2) &= f_1 l_1 - f_2 l_2 \\ &\overset{0}{=} f_1 l_1 - [f_1 - r(tan\alpha_1 - tan\alpha_2)] l_2 \\ &\overset{1}{=} l_2 r(tan\alpha_1 - tan\alpha_2) - f_1(l_2 - l_1) \\ &\overset{2}{=} rh(tan\alpha_1 - tan\alpha_2)/sin\alpha_2 - f_1 h(1/sin\alpha_2 - 1/sin\alpha_1)\end{aligned} \tag{6.4}$$

Passage 2 of Equation (6.4) is found according to $l_i = h/sin\alpha_i$. We computed fuel saving $\mathcal{R}(\alpha_1, \alpha_2)$ by setting the following parameters:

- $\alpha_1 = 5°$: the angle for initial path p_1, which approximately has a 10% inclination, denotes a very steep path (5° is standardized as the maximum slope allowed for roads).
- $h = 0.01$ km: a height of 10 m is reached by paths p_1 and p_2. We have that, for angle α_1, the path p_1 has a length of about 100 m.
- $f_1 \in [0.2, 0.5, 0.8]$ l/km: *fuel_per_km* consumption values for path p_1: we select them according to Table 6.1.
- $\alpha_2 < \alpha_1$: we investigate fuel saving for paths with a lower inclination and consequently a higher length.

From Figure 6.6, we observe that we have a positive fuel saving for $f_1 = 0.2$ l/km and $f_1 = 0.5$ l/km, while for $f_1 = 0.8$ l/km, we waste fuel. By remembering that path p_2 has a lower angle α_2 but a greater length, from the result, we see that, for greater values of f_1, we do not save fuel because the length of the path has a higher impact on the consumption. In

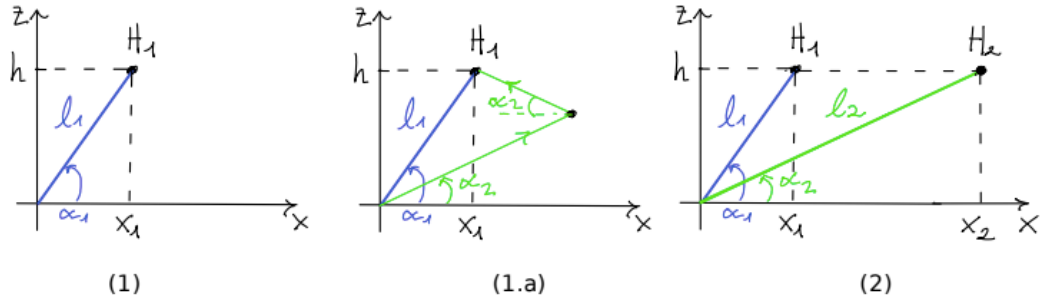


Figure 6.5: Two paths for reaching the height $z = h$: path (**1**) has a length l_1 and an angle α_1; path (**1.a**) reaches the same destination as (**1**) by keeping a lower constant slope with a longer length. Path (**2**) is used in the proof and is equivalent to path (**1.a**) in the angle α_2 and length l_2. The paths are compared in terms of fuel savings under different configurations, as shown in Figure 6.6.

contrast, for lower values of f_1 (starting from a threshold that is approximately 0.55 l/km), we have a positive saving because the decrease in slope has a higher impact on fuel consumption.

6.5.2 Bayes Network Validation

We compare the Bayes networks discovered with the results of the Granger experiments described in Section 6.4 on a set of multivariate time series constructed from the original data set.

Our data set is temporally confined between January and June 2019, and for each month, we generally have five consecutive dates of measurement. We exploit the temporal order of the data set and frame it as a multivariate time series collection. The path records contain information concerning the bus service: `DAT` (date and time of measurement) and `VehicleID` (identifier of the vehicle). We subset the data set into a collection of multivariate time series $\{t^{(l)};\quad l = 1, \dots, T\}$ according to `VehicleID` and the month of `DAT`. Each multivariate time series $t^{(l)} := t_{vm}$ corresponds to the time-ordered collection of the path records measured for a given vehicle v on a given month m (we selected a time series with more than 20 temporal records and obtained a collection of $T = 125$ multivariate time series). By formulating the data set as a time-series collection $\{t^{(l)} : l = 1, \dots, T\}$, we obtained a set of boolean Granger matrices $\{G^{(l)} : l = 1, \dots, T\}$, which represents the results of the Granger experiments.

Each Granger matrix $G^{(l)}$, computed from $t^{(l)}$ according to Equation (6.3), can be interpreted as the adjacency matrix of a graph $\mathcal{G}^{(l)}$. Given a Bayes network $\mathcal{B}$, we compared it with the set of Granger experiment graphs $\{\mathcal{G}^{(l)}\}$ obtained for each time series of $\{t^{(l)}\}$. That is, we compare the Bayes adjacency matrix B with the set of Granger matrices $\{G^{(l)}\}$

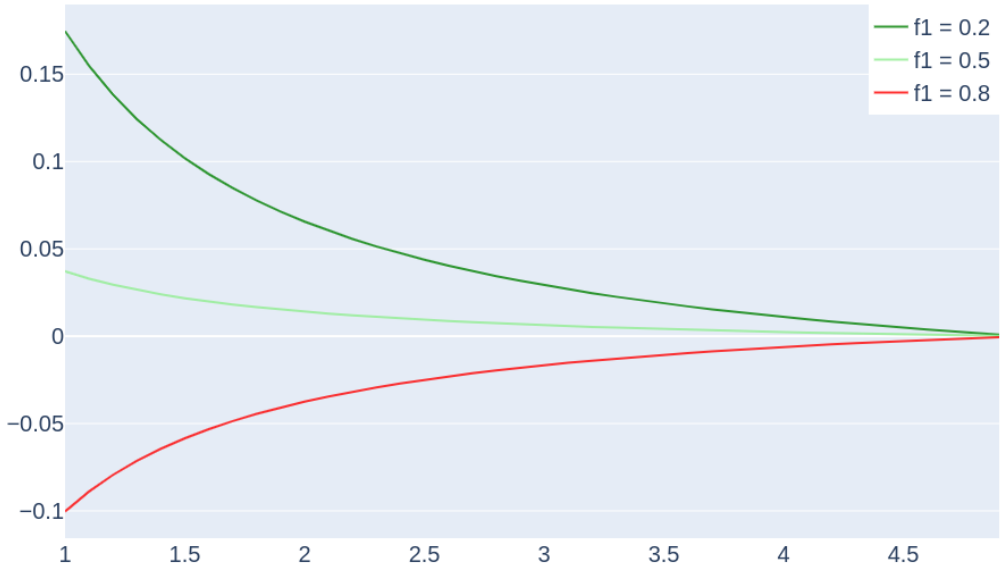


Figure 6.6: Fuel consumption saving (l) vs. slope angle α_2 (°) of path (2). The saving is computed by the difference in the consumption of path (1) (with fixed $\alpha_1 = 5°$) to path (2) of Figure 6.5. The curves in the colors are for different values of $f_1 \in [0.2, 0.5, 0.8]$ (l/km) (*fuel_per_km* consumption in path p_1).

by considering each element of the set as a ground truth.

We define a Bayes performance metric $m(B, G)$, and from it, we define $M(B, \{G^{(l)}\}) = \frac{1}{T}\sum_{l=1}^{T} m(B, G^{(l)})$ as the average Bayes performance over the collection $\{G^{(l)}\}$. We use $M(B, \{G^{(l)}\})$ as a mean of the comparison over our Bayes network collection.

Performance with Respect to Granger Experiments

We frame the adjacency matrices B and G as boolean vectors $\mathbf{b}$ and $\mathbf{g}$ so that each vector's ith entry is a boolean indicator of the presence of link i. We compared two models, Bayes vector $\mathbf{b}$ and Granger vector $\mathbf{g}$, which evaluate the presence for each link on a given link collection with size L (in our case, for $V = 7$ variables, we have $L = 42$ possible links). If we establish the Granger vector $\mathbf{g}$ as the ground truth of this binary classification task, we can evaluate the binary metrics of $\mathbf{b}$ with respect to ground truth $\mathbf{g}$: that is, the number of links in the collection that are true positive (TP), true negative (TN), false positive (FP), and false negative (FN). Then, we can construct a set of Bayes performance metrics as shown in

Table 6.5:

Binary Metrics	Bayes Network Performance Metrics
$TP = \sum_{i=1}^{L} \mathbf{I}(b_i = 1, g_i = 1)$	$sens = TP/(TP + FN)$
$TN = \sum_{i=1}^{L} \mathbf{I}(b_i = 0, g_i = 0)$	$spec = TN/(TN + FP)$
$FP = \sum_{i=1}^{L} \mathbf{I}(b_i = 1, g_i = 0)$	$avg\_recall = (sens + spec)/2$
$FN = \sum_{i=1}^{L} \mathbf{I}(b_i = 0, g_i = 1)$	$acc = (TP + TN)/(TP + TN + FP + FN)$

Table 6.5: Binary metrics of Bayes vector **b** with respect to ground truth **g** (**b** and **g** are boolean link-indicator vectors of a link collection of size L). From binary metrics, Bayes performance metrics are evaluated: sensitivity (*sens*), specificity (*spec*), average recall (*avg_recall*), and accuracy (*acc*).

Sensitivity (*sens*) measures the ratio of true positives (TP) over the total positives of ground truth **g** (TP + FN), that is the percentage of **g** present links that are correctly identified by **b**. Specificity (*spec*) measures the ratio of true negatives (TN) over the total negatives of the ground truth **g** (TN + FP), that is the percentage of **g** absent links that are correctly identified by **b**. Average recall (*avg_recall*) measures the average between sensitivity (*sens*) and specificity (*spec*). Accuracy (*acc*) measures the ratio of true positives and true negatives (TP + TN) over the link collection size L = TP + TN + FP + FN, which is the percentage of links, whether absent or present for ground truth **g**, that are correctly identified by **b**.

For each performance evaluation $m(B, G^{(l)})$ related to a given time series $t^{(l)}$ (m : sens, spec, avg_recall, and acc), we compute the averaged Bayes performance metrics $M(B, \{G^{(l)}\}) = \frac{1}{T}\sum_{l=1}^{T} m(B, G^{(l)})$ over the time series collection $\{t^{(l)}\}$ (M : SENS, SPEC, AVG_RECALL, and ACC) for each discovered Bayes Network. We report the average performances in Table 6.6, where the error is computed as the standard deviation of performance over the Granger matrix collection $\{G^{(l)}\}$.

	SENS	SPEC	AVG_RECALL	ACC
BF	**0.42 ± 0.12**	0.68 ± 0.06	**0.55 ± 0.09**	**0.58 ± 0.08**
HC	0.41 ± 0.09	0.68 ± 0.05	0.54 ± 0.07	0.57 ± 0.06
RM	0.36 ± 0.09	**0.72 ± 0.05**	0.54 ± 0.07	**0.58 ± 0.07**

Table 6.6: Bayes average performance metrics over the Granger experiment set $\{G^{(l)}\}$: sensitivity (SENS), specificity (SPEC), average recall (AVG_RECALL), and accuracy (ACC) evaluated for the Bayes networks (HC, BF, and RM). The metrics represent the level of accordance between the Bayes causality models and the collection of results obtained by the Granger experiments.

From Table 6.6, we observe that the Bayes network performances are below 60% for all

metrics except for specificity. A higher sensitivity is reached by brute force (BF)m which on average identifies 42% of present links with respect to the time-series Granger graphs. A higher specificity is reached by restricted maximization (RM), which identifies 72% of absent links with respect to the time-series Granger graphs. The average recall, which is the mean between sensitivity and specificity, is reasonably similar for all the networks and is around 55%. We have the same for accuracy, which is around 58%.

From Table 6.6, we can state that we have a poor consistency between the discovered Bayes networks and the Granger experiments since we have that, on average, these models have a low percentage of commonly identified causal relationships (e.g., accuracy for all networks is around 58%).

We can motivate these results by the following arguments:

1) Time-series properties

   The time series may not be correlated with time and may have a consistent random component. We can verify this with Ljung–Box test [122] with total number of lags $h = 20$ and significance level $\alpha = 0.05$ for each feature of the time series of our collection $\{t^{(l)}\}$. For each feature, we report the percentage of time series for which we confirm the independence assumption: *accel*, 71%; *avg_slope*, 54%; *air_cond_ptime*, 12%; *brake_usage*, 57%; *mass*, 7%; *fuel_per_km*, 57%; and *stop_ptime*, 87%. We observe that most of the features, especially *stop_ptime* and *accel*, present a high independence frequency over the time-series collection except for *aircond_ptime* and *mass*. This result may suggest that our time-series framing may be the reason for the low consistency between Granger experiments and Bayes networks.

2) Conceptual causality difference

   We may observe that Granger test searches for causality by identifying a past temporal dependence by means of the vector auto regression model, while Bayes networks search for a present causality between features, which are collected on the same temporal level.

   Let us take an example. We consider multivariate time-series variables (A, B) for which the Granger test provides $A \overset{(g)}{\Rightarrow} B$. We have that A has a predictive power in forecasting B, but we may not be sure about the existence of a present causal dependence between A and B, that is the type of causality identified by Bayesian networks.

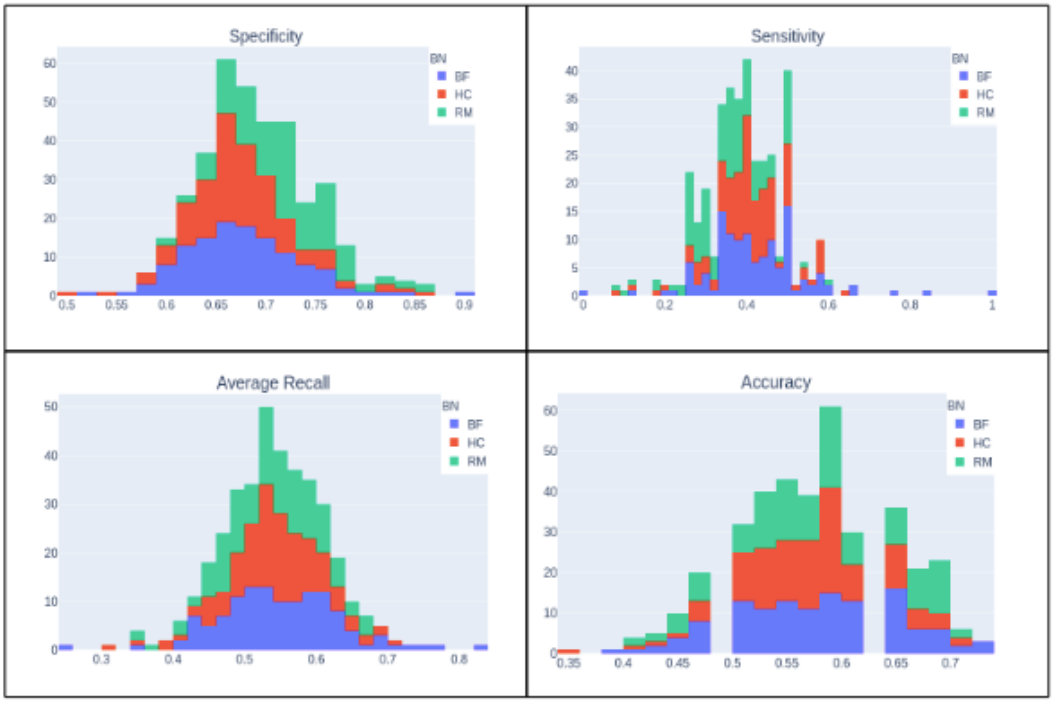


Figure 6.7: Bayes network (HC, BF, and RM) performance metric histograms (specificity on the top left, sensitivity on the top right, average recall on the bottom left, and accuracy on the bottom right) over the resulting collection of the Granger experiments performed on the multivariate time-series data set. The average of each performance metric for each Bayes network is reported in Table 6.6.

6.6 Conclusions

This work presents different contributions with the purpose of analyzing the conditions at which fuel consumption occurs in vehicles and of understanding how to reduce it by intervening in the scenario. We provided a collection of data from sensors installed on buses used as public transport. Thanks to the sensor data analysis, we discovered that, in some contextual conditions (with a fuel consumption per kilometer that does not exceed the value of 0.75 L per kilometer), it is preferable to choose a longer but less steep path than a shorter one. As a consequence of the analysis of cause–effect relationships between the variables and the target, we precisely quantified the impact of all causes on the target: with a decrease of one unit of `air_cond_ptime` (percentage of travel time with air conditioning), we can can expect a decrease of 0.107 units in `fuel_per_km`; with a decrease of one unit of the percentage of time with `brake_usage`, we can expect a decrease of 0.206 units in `fuel_per_km`; and with a decrease of a unit in `stop_ptime` (stop percentage time with engine on), we can expect a decrease of 0.445 units in `fuel_per_km`. In the literature [93], the important effect

of this variable was confirmed.

We tested both approximate algorithms, driven by the BIC score and brute force with the purpose of comparing the ability of the algorithms to converge to the same resulting networks. We evaluated their results with the adoption of Granger causality, a third-party criterion, based on the time series formed in time by the observed variables. This is an original contribution to the scientific community of Bayesian networks that are usually scored by BIC or K2. According to the Granger causality, we are also able to rank the alternatives, even in the case where multiple BNs share the same score. We compared BNs also by using their ability to perform feature selection and to predict the target variable.

We discussed the comparison results. The networks sometimes agree, and other times, they do not. This mismatch perhaps is due to the multiple maxima that sometimes exist in the large search space of the solution. The observed mismatches on the edges might also be a consequence of the heuristics. Heuristics are indeed used to eliminate multiple rankings of the alternatives, in choosing edge directions (choice of the cause and the effect that often requires the experts' advises), and for avoiding cycles in the BN graphs.

In summary, the contributions of our work are as follows:

1. Bayesian networks were applied for the analysis of fuel consumption. Past studies on fuel consumption in vehicles (reported in Section 6.1) applied only machine learning predictive models (based on SVR, ANN, random forest, or gradient boosting). All of them have the sole goal of predicting the target value. None provide machine learning models that are able to also perform the following: (a) describing and discovering the cause–effect relationships between variables and the target (Section 6.5) and
   (b) performing an intervention analysis on the causes, with the goal of achieving a desired impact on the target and quantifying this impact (Section 6.5.1).
   Bayesian networks are powerful and we used them to reach multiple goals: perform feature selection (Section 6.5.1) whose outcomes we compared with another standard method (VIF [119]); perform predictive modeling (target estimation, whose results are shown in Table 6.3), intervention analysis (Section 6.5.1) and counterfactual analysis (what-if analysis).

2. Comparing the results of approximate algorithms (heuristic-driven) for Bayesian networks with a brute force algorithm, an original one, implemented for this work (Algorithm 5) was made possible thanks to the availability of high-performance computing

technology that permits us to afford an extremely high computational load of traversing the huge search space of the possible networks by partitioning it and spreading evaluations of the alternative graphs throughout many servers. The outcome of this comparison (Section 6.5.2) can help analysts with the uncertainty of which Bayesian network to use.

3. The use of the Granger causality concept was introduced and formalized for an evaluation of Bayesian networks (Section 6.4). Granger causality was used as an independent, third party notion to compare, evaluate, and rank the different Bayesian networks, generated from the same data by different algorithms.

4. Bayesian network discovery is customarily used to test the domain knowledge, previously distilled under the form of an already available graph [107, 108, 118, 123]. Differently, in this paper, we did not start from an already available graph but directly started from the collected (sensor) data and provided experts with assumptions about this knowledge (cause–effect relationships) under the form of a Bayesian network.

5. Last but not least, we provided a public data sets to the scientific community [101] with real data from buses, useful for testing Bayesian network algorithms and time series analyses.

Chapter 7

Utility Privacy Trade-off in a Differential Private Mechanism

7.1 Introduction

Local differential privacy aggregates randomized responses from each client to publish sensitive information for statistical analysis while protecting individuals' privacy. This chapter presents a utility privacy trade-off model suitable for contingency tables. We characterize the notion of the utility model through experimental analysis, in which we demonstrate the effect of the Laplace noise on the dependencies structure of two or more random variables in the contingency tables.

In the context of LDP, there is a trade-off between utility and privacy. The more randomness in the output of the randomized mechanism, the stronger the individual privacy. On the other hand, this makes more noisy the aggregator's estimations.

To characterize this utility and privacy trade-off, we need a confidence interval on the probability of observed responses that measures the utility and privacy in the underlying protocol. Using this confidence interval, we can identify a minimal level of privacy that a protocol provides while maximizing the utility.

In theory, LDP provides a privacy guarantee in any situation, and in the worst-case scenario, LDP provides an upper bound on the privacy leakage. However, it can be practically difficult to define an upper bound in the privacy definition [124, 125]. Many relaxed definitions of LDP settings have been proposed in the literature [8, 126, 127]. Therefore, it is very important to understand the behaviour of a privacy mechanism and the privacy guarantee it

provides in the worst-case scenario.

The privacy leakage by a randomized mechanism can be defined using Mutual Information (MI) [128], which models how well an aggregator with access to the released data can refine its belief about the private data [129]. Mutual information is evaluated on the difference between the randomized response received by an aggregator and the users' original data. However, the aggregator infers users' data by linking the knowledge obtained from other data sources. Hence, mutual information obtained from the users' data and the output of the randomization protocol does not correctly quantify information leakage by the mechanism if one does not condition on the data statistics [8].

Several approaches in the literature define the information-theoretic utility measures in LDP [124, 125, 129, 130]. Some of them concentrate on mutual information, while others measure utility via an information leakage by the privacy protocol.

In this chapter, we also use information theory to characterize the notion of utility by means of experimental analysis in the differential private protocol. In this chapter, we present the utility privacy tradeoff suitable for the contingency tables. In the context of LDP, there is a trade-off between utility and privacy. The more randomness in the output of the randomized mechanism, the stronger the individual privacy. On the other hand, this makes more noisy the aggregator's estimations. We use the *log-likelihood* ratio G^2/G-test [131] to build a confidence interval on the utility of the responses on the basis of observed noisy values due to noise addition by the Laplace mechanism for privacy preservation.

7.2 Notations

For the notation simplicity, we only consider a *k-way* contingency table T over k attributes $\mathbf{A} = (A_1, A_2, \cdots, A_k)$. The table can be calculated directly from the original dataset or the attributes-parent pairs in the Bayesian network, as presented in Section 5.2.1. We denote by $T[w_i]$ a a specific value in the contingency table. We will also denote the original contingency table by T and use T' to denote a projection of T after noise addition by a $\epsilon-$differentially private mechanism.

A privacy mechanism $\mathcal{M}$:$T \rightarrow T'$ maps original values in T to noisy values in T', where the noise is drawn from the randomization protocols outlined in Chapter 3. For generalization, we use a standard ϵ-differentially private Laplace distribution to add noise to remap $T \rightarrow T'$.

7.3 Confidence Interval for a Bayesian Network Built over Noisy Data

We assume we have a noisy contingency table T', representing the joint distribution of two binary random attributes A_1 and A_2. Noise is added in the contingency table by addition in the counting of each cell w_i of a random variable x_i, drawn from the Laplace distribution $Lap(0, b)$ with zero mean and a scale that depends on the privacy budget ϵ as $b = \frac{2}{\epsilon}$. For simplicity, we consider a 2-by-2 contingency table with one degree of freedom.

(a) Observed values T'

w_1	w_2	n_1
w_3	w_4	n_3
n_2	n_4	n

(b) Expected counting (T)

$w_1 - x_1$	$w_2 - x_2$
$w_3 - x_3$	$w_4 - x_4$

Table 7.1: Observed and expected counts in a 2-by-2 contingency table

When two variables are judged dependent by application of a statistical test for independence (like *log-likelihood* ratio G^2 or G-test), an edge between the two variables in the Bayesian Network is drawn. The G^2 test is applied by the following formula:

$$G^2 = 2 \sum_i O_i \cdot \ln \frac{O_i}{E_i} \tag{7.1}$$

For the analysis of the noisy table T' the value of G^2 can be expressed in terms of mutual information, given as

$$G^2 = 2 \sum_i \left[w_i \cdot \ln \left(\frac{w_i}{w_i - x_i} \right) \right] \tag{7.2}$$

with w_i the counts in cell i and x_i the amount of random noise added.

Let us assume that on the noisy contingency table, the G^2 test is passed, that means that $G^2 \geq G_c^2$ (where G_c^2 is critical value), with G^2 a constant that depends on the degrees of freedom of the contingency table (it is 3.84 for a 2-by-2 table). In this situation we make the null hypothesis that the variables are independent ($H_0 : A_1 \perp\!\!\!\perp A_2$) and we ask ourselves which is the probability that as a consequence of the addition of noise variables, distributed by Laplace distribution, the outcome of the dependency test changes and the null hypothesis is rejected. The computation of the probability of this event, is the goal of this section. Instead, if the G^2 test failed on the noisy contingency table, we make the opposite assumption: the null hypothesis is that the variables were actually dependent (and on the true contingency

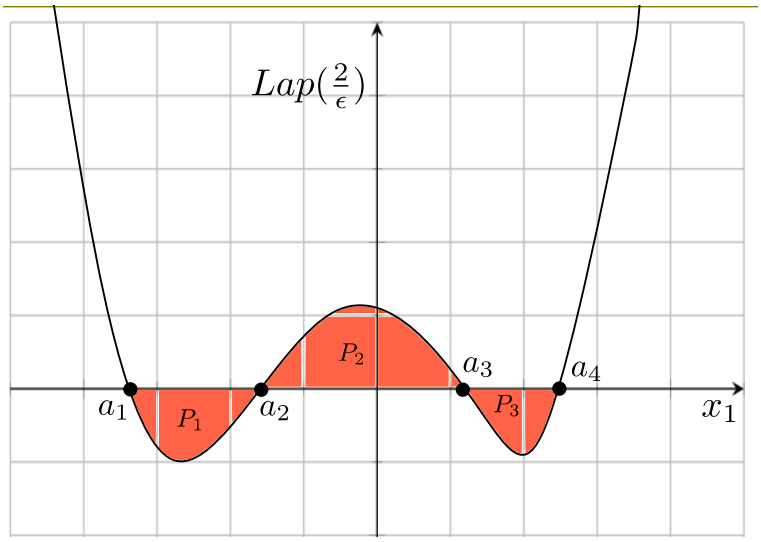


Figure 7.1: Area under the curve of Laplace noisy x_1 and $Lap\frac{2}{\epsilon}$

table the G^2 test is passed). This second case is completely dual of the first one, so for the sake of brevity we will deal only on the first one.

We have to compare G^2 with the critical value G_c^2 below which, the variables per row and column are presumed as independent. Therefore, we need to find the values of w_1, w_2, w_3 and w_4 (which depend on the Laplace distributed variables x_1, x_2, x_3 and x_4 that represent the added noise to the cells) for which the following equation holds:

$$G^2 = 2\sum_i \left[\left(w_i \cdot \ln \frac{w_i}{w_i - x_i}\right) \cdot \left(\frac{1}{\epsilon} \exp^{-\frac{\epsilon \cdot x_i}{2}}\right)\right] \tag{7.3}$$

We are interested in finding when the outcome of the G^2 test will *change* as a result of the noise (x_1, x_2, x_3, x_4), such that:

$$G^2(x_1, x_2, x_3, x_4) - G_c^2 \leq 0 \tag{7.4}$$

We are interested in calculating the areas of the segments contained between the *x-axis* or values of x and the curve $Lap(0, \frac{2}{\epsilon}) = \frac{1}{\epsilon} \exp^{-\frac{\epsilon \cdot x_i}{2}}$. In order to calculate the values of x, it seems reasonable for us to draw a sketch of the curve, as there is no mention of the ordinates, or values of noise x. We know that, the curve crosses the *x-axis* when $Lap(0, \frac{2}{\epsilon}) = 0$, in other words when $x_1 = a_1, x_2 = a_2, x_3 = a_3, x_4 = a_4$. We see that when x is large and positive $Lap(0, \frac{2}{\epsilon})$ is also large and positive, and when x is large and negative, $Lap(0, \frac{2}{\epsilon})$ is also large and negative. Joining all these features we obtain the curve as shown in Figure 7.1.

So we focused on the intervals in which the Equation 7.4 is negative (in the area P_1, that is, between a_1 and a_2, and the area P_2, between a_2 to a_3 and so on). We integrate the Laplace distributed variable x_1 using Equation 7.5 in each of these intervals. This allowed us to obtain P_i that is the probability that the observed value is generated as a consequence of noise addition given an assumed true value.

We also know that all the noisy variables $x_1, \cdots, x_4$ are independently drawn from the Laplace distribution. Summing up them will approximate the total probability that the dependency will change due to the noise addition. This probability will define the utility of the data in the contingency table obtained after the addition of noise due to privacy. On the opposite side it will also help to approximate how much noise is sufficient to add so that the dependency between the random variables remains intact.

$$
\begin{aligned}
Lap(0, \frac{2}{\epsilon}) &= \int \left(\frac{1}{\epsilon} \exp^{-\frac{\epsilon \cdot x_i}{2}} \right) \partial x_i \\
Lap(0, \frac{2}{\epsilon}) &= \left(\frac{1}{2 \cdot \frac{2}{\epsilon}} \right) \cdot \exp \left(\frac{-|x_i|}{\frac{2}{\epsilon}} \right)
\end{aligned}
\tag{7.5}
$$

Lemma 7.3.1. *The extreme values for all the variables* (x_1, x_2, x_3, x_4) *are within the intervals* $\pm 9\sigma = \pm 9\sqrt{2}\frac{2}{\epsilon}$

Proof. We look for the extreme values between the interval $\pm 9\sigma = \pm 9\sqrt{2}\frac{2}{\epsilon}$ for all the variables (x_1, x_2, x_3, x_4) and check the values of $G^2 - G_c^2$ in case it is monotonic: we have 2^x possibilities. To find the maximum probability of each P_i, we use Chebyshev's inequality. According to Chebyshev's inequality:

$$
P(|x_i - \mu| \geq k\sigma) \leq \frac{1}{k^2} = 1 - P_i \tag{7.6}
$$

We use $k = 9$ to achieve the minimum 98.7654% within k-standard deviations of the mean. Since we have 4 independent variables (x_1, x_2, x_3, x_4), multiplying each $P_{x_1} \cdot P_{x_2} \cdot P_{x_3} \cdot P_{x_4} = (0.9876)^4 = 0.9153 = 91.53\%$ confidence level. The extreme boundaries on the Laplace distribution of x_i are given by:

$$
k = 9; \quad P_{x_1} = \frac{1}{81} = 0.01234 = 1 - 0.9876
$$

Figure 7.2 illustrates the maximum probability of variable x_1, the upper $u_{bound} = 9\sigma$ and lower $l_{bound} = 9\sigma$ bound, and the extreme values $1 - P_{x_1} = \frac{1 - P_{x_1}}{2} = 0.0062$. ■

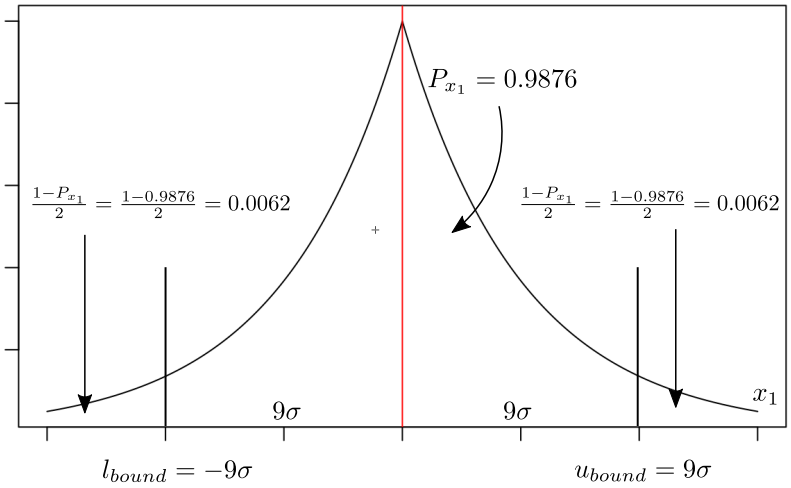


Figure 7.2: Upper and lower bound of Laplace distribution on x_1

7.3.1 Experimental Evaluation: G^2 test

For the experimental evaluation, we select a 2-by-2 noisy contingency table with one degree of freedom. We start with the lowest noisy value in the first cell ($T'[w_1] = 5$ because one of the requirements to pass the G^2 test is that, each cell value in the contingency table should be greater than or equal to 5). We adjust the remaining cell values (w_2, w_3, and w_4) so that the marginals (n_1, n_2, n_3, and n_4) and the total n remain consistent. To calculate the total probabilities ($P_{x_1}, \cdots, P_{x_4}$) for all the four variables ($x_1, \cdots, x_4$) for which G^2 passes, we use the following equation:

$$\int_{lb_{x_i}}^{ub_{x_i}} \prod_i \left(\frac{1}{\left(2 \cdot \frac{2}{\epsilon}\right)} \cdot \exp\left(\frac{-|x_i|}{\frac{2}{\epsilon}} \right) \right) \partial x_i \cdot 2 \sum_i \left(w_i \cdot \ln\left(\frac{w_i}{w_i - x_i} \right) \right) \tag{7.7}$$

The first part of the equation is the integral of the Laplace distribution from Equation 7.5. We iteratively run the above Equation 7.7 in Mathematica[1] by incrementing the first cell value ($w_1 + c$, where c is constant assumed as the original true cell value), until the marginals does not change anymore. We plot the confidence intervals in Figure 7.3 which shows the probabilities that the dependence will not change on the *y-axis* and G^2 test on the *x-axis*. Figure 7.3 shows the confidence intervals at different number of total records n in the contingency table. As shown in the Figure 7.3, it is 99% probable that the dependency between two random variables remains intact after the addition of noise in the contingency tables

[1] https://www.wolfram.com/mathematica/

with higher cardinality than n. Whereas, contingency tables with lower cardinality than n, it is 80 % probable that the dependency will break with the addition of noise drawn from $Lap(\frac{2}{\epsilon})$ where $\epsilon = 0.1$.

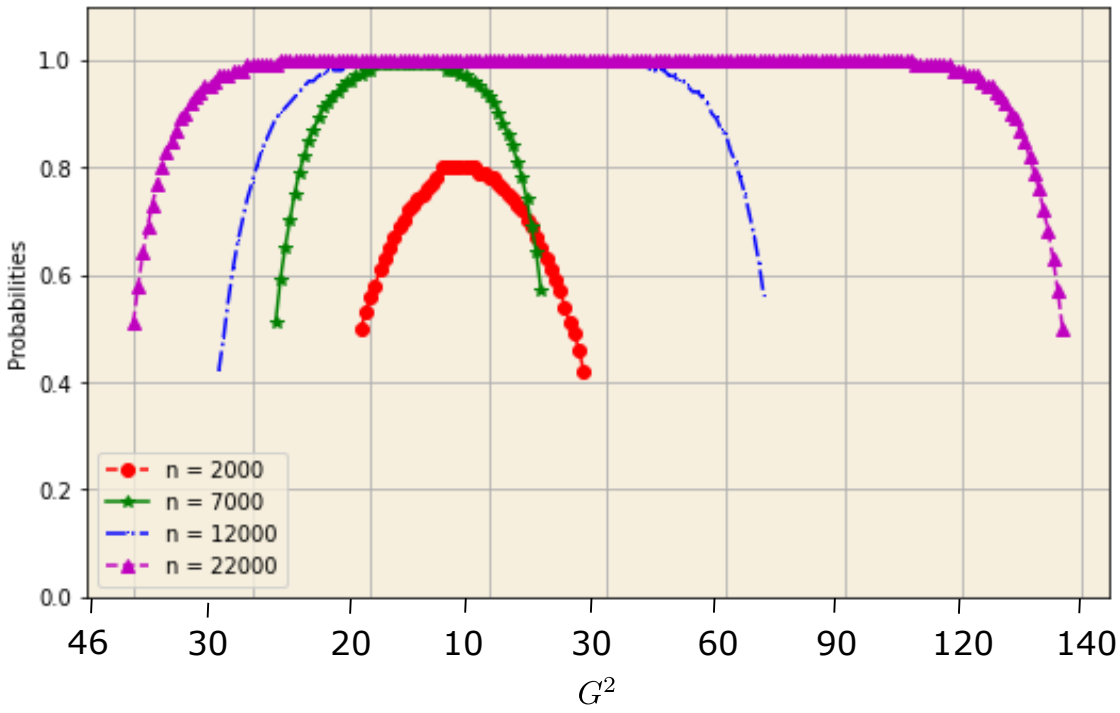


Figure 7.3: Confidence interval using G^2 test on 2-by-2 contingency tables having different records ($n = [2000, 7000, 12000, 22000]$, and $\epsilon = 0.1$). On the *y-axis* we plot probabilities and on *x-axis* we have G^2 values (G^2 values are divided by a factor 100 for the plotting purpose)

7.4 Robustness of Noise on the Dependence Structure Between Variables

In this section, we will discuss the effect of the Laplace noise on the BN structure at various values of the privacy parameter ϵ. We build the confidence interval on the BN dependence structure using Pearson's chi-square test of independence χ^2 (as discussed in Section 6.3.2). Our aim is to evaluate the effect of Laplace noise on the dependencies statements $(A_i|\Pi_i)$. We believe that, adding a large amount of noise (corresponding to low values of ϵ) might break the dependencies between the variables. Conversely, if we add a small amount of noise (with higher values of ϵ), the dependencies should remain intact. To perform χ^2 test, we add Laplace noise in the conditional distribution of the variables. These conditional distributions

are represented as the contingency tables, which are identified by the Markov blanket of the Bayesian structure.

Definition 7.4.1 (**Markov blanket** [132])**.** *The Markov blanket of a node* $N \in \mathbf{A}$ *is the minimal subset* $\mathbb{S}$ *of* $\mathbf{A}$*, conditioned on which other nodes are independent with* N*:*

$$N \perp\!\!\!\perp \mathbf{A} \setminus \mathbb{S} \quad | \mathbb{S} \tag{7.8}$$

It means that the subset $\mathbb{S}$ *contains all the information one needs to infer* N*, where the attributes in* $\mathbf{A} \setminus \mathbb{S}$ *are redundant.*

In any Bayesian network, the Markov blanket of a node N is its parent set Π, its children, and all the other nodes sharing a child with N.

7.4.1 Confidence Interval

To model the confidence interval, we use χ^2 independence testing on a differential private contingency table T'. This noisy table T' is generated by adding Laplace noise $Lap(\frac{2}{\epsilon})$ independently in each cell of the original contingency table T. We can say that this T' is a noisy projection of T. In the following section, we will discuss the test of independence using differentially private hypothesis testing.

Hypothesis Testing Preliminaries

Given a noisy conditional distribution T' on a set of attributes $(A_i|\Pi_i)$, we wish to identify whether this distribution comes from a specific model, which is given as a *null hypothesis* H_0. The hypothesis testing is done using a procedure which takes a distribution T', level of significance $1 - \alpha$, and null hypothesis H_0. This procedure outputs a decision of whether to reject H_0 or accept it. We would like to design our significance test to such a degree that the hypothesis testing algorithm achieves *Type I* error with probability α (that is the error of rejecting the null hypothesis when it is true). More formally, $Pr\left[T', \alpha, H_0\right] : Reject\left(H_0\right) \leq \alpha$ while also achieving a small *Type II* error (risk of accepting the null hypothesis when it is false).

Conditional independence test focus on the presence of an *arc* in the Bayesian network. Each arc contains a probabilistic dependence, and conditional independence tests can be used to model whether the data support this probabilistic dependence. An arc is considered to be

Algorithm 6 Private Hypothesis Testing

Input: contingency table T generated using $(A_i|\Pi_i)$, privacy parameters ϵ

$T' = T + Z$ with $Z \sim Lap(\frac{2}{\epsilon})$

$\widetilde{m} = MLE(T')$

if $\widetilde{m} == \emptyset$ **return:** Accept

else

  $\hat{q} \leftarrow \widehat{\chi}^2$ using T' and $\widetilde{m}$ `// from equation 4.8`

  set $k > 1/\alpha$ and $q = \emptyset$

  for $t \in [k]$ **do**

    generate contingency table $\widehat{T}'$ using $\widetilde{m}$

    $\widetilde{T}' = \widehat{T}' + Z$ with $Z \sim Lap(\frac{2}{\epsilon})$

    $\widetilde{m} = MLE(\widetilde{T}')$

    if $\widetilde{m} == \emptyset$ **return:** Accept

    else compute $\widehat{\chi}^2$ from 4.8, add it to the array q

  end

  $\tilde{\tau}^{\alpha} = \lceil (k+1)(1-\alpha) \rceil$ ranked statistic in q

  $\widetilde{m}$ = MLE with $\widehat{T}'$ using Equation 4.9

  return: $\widetilde{m}$

added in the network if the null hypothesis H_0 (by conditional independence) is rejected. For example, consider a Bayesian network in Figure 5.1. Consider adding an edge from $\mathtt{S} \rightarrow \mathtt{O}$. The $H_0 : O \perp\!\!\!\perp S|\{\mathtt{R}\}$ is that, `Occupation` is probabilistically independent of `Sex` conditioned on its parent (`Region`). The alternative hypothesis is $H_0 : O \not\perp\!\!\!\perp S|\{\mathtt{R}\}$.

7.4.2 Deferentially Private Hypothesis Testing

Differential private independence testing is performed using the same technique which we discussed in Section 4.6. Here, we present an algorithm to perform a Monte Carlo hypothesis testing on attributes-parent pairs $A_i|\Pi_i$ in BN. This differential private hypothesis testing is performed at different privacy levels ($\epsilon \in [0.1, 0.25,$ and $0.35]$) similar to the work of [76].

We use Algorithm 6 to build a confidence interval for accepting or rejecting the dependencies between the attributes parent-pairs in the contingency table $(A_i|\Pi_i)$. This confidence interval shows at what level of Laplace noise (controlled by ϵ) a strong dependency/independency between two random variables in the Bayesian network remains intact. Algorithm 6 works as follows: given a contingency table, we add a random noise drawn from the Laplace distribution $Lap(\frac{2}{\epsilon})$ to generate T' $\epsilon-$ differentially private. We use the function `MLE` to estimate the parameters $\widetilde{m}$ that correspond to the null hypothesis of independence. Using $\widetilde{m}$, we sampled $k > 1/\alpha$ many noisy contingency tables to get k different values for $\widehat{\chi}^2$ (equa-

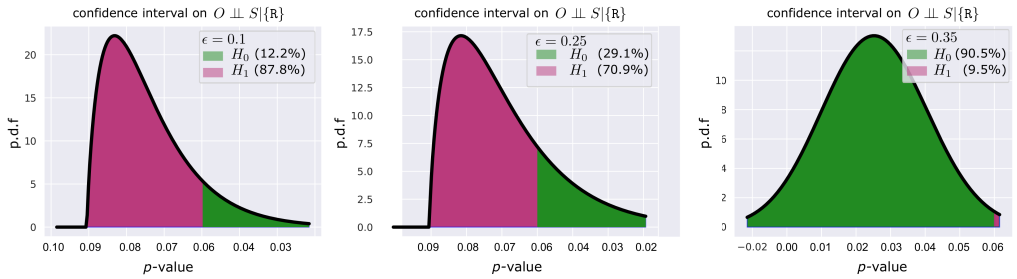


Figure 7.4: Level of confidence on χ^2 probabilistically independent hypothesis test (H_0 : $O \perp\!\!\!\perp S|\{\mathrm{R}\}, \alpha = 0.05$), left ($H_0 = 12.2\%$, $H_1 = 87.8\%$ with $\epsilon = 0.1$), middle ($H_0 = 29.1\%$, $H_1 = 70.9\%$ with $\epsilon = 0.25$), right ($H_0 = 90.5\%$, $H_1 = 9.5\%$ with $\epsilon = 0.35$)

tion 4.8) and choose $\lceil (k+1)(1-\alpha) \rceil$ ranked statistics as our threshold $\tilde{\tau}^{\alpha}$. If at any stage MLE returns $\emptyset$, we select Accept H_0.

7.4.3 Monte Carlo Test: Test of Independence

Monte Carlo simulation is performed to evaluate the robustness of Laplace noise at different privacy budgets ϵ. We want to evaluate the result of noise on the dependencies between the attributes in the BN structure. We believe that if we add a large amount of noise in the contingency tables, it might break the dependencies between the attributes and if we add a lower amount of noise, the dependencies should remain intact. Likewise, we select conditional distribution $(A_i|\Pi_i) \leftarrow (O, S|R)$ from Figure 5.1. Our hypothesis is given as, $H_0 : O \perp\!\!\!\perp S|\{\mathrm{R}\}$ and the alternative hypothesis is $H_1 : O \not\perp\!\!\!\perp S|\{\mathrm{R}\}$. To test the χ^2, we set $\alpha = 0.05$. We repeated our experiments over 1000 trials. Each set of 1000 trials is performed on varying values of noise ($\epsilon \in [0.1, 0.25, 0.35]$).

Figure 7.4 illustrates the confidence interval using the χ^2 on conditional probabilities $(A_i|\Pi_i)$. Our null hypothesis is $O \perp\!\!\!\perp S|\{\mathrm{R}\}$; the alternative hypothesis is $O \not\perp\!\!\!\perp S|\{\mathrm{R}\}$, $\alpha = 0.05$, with 3 degree of freedom. Figure 7.4 shows that if we add a large amount of noise (e.g, with $\epsilon = 0.1$) the null hypothesis is 12.2% of the times accurate; if $\epsilon = 0.25$, then H_0 is accurate 29.1% of the times; if $\epsilon = 0.35$ the null hypothesis H_0 is accurate 90.5% of the times.

Hence, we conclude that adding a large amount of noise, the noise will create false dependencies in the BN structure, thus reducing the utility of the data.

Chapter 8

Conclusions

In this dissertation, we developed a set of randomized response protocols to collect users' sensitive information in a *non-trusted* environment to achieve local differential privacy. We proposed differentially private algorithms which provide a privacy guarantee while maximizing the utility of the released data.

We systematically explored the problem of collecting and analyzing data from users under $\epsilon-$local differential privacy. In our setting, neither the aggregator nor the server is trusted, and they only access randomized responses from the users' to reconstruct statistical models. The server can compute accurate statistics from these joint distributions as contingency tables.

We proved theoretically that the proposed randomized response protocols satisfy the definition of differential privacy. We also analyzed the relationships between the parameters of the proposed protocols and the privacy budget, the main parameter adopted in literature to control the level of privacy. From the experimental viewpoint, the experiments showed that our protocol achieves high utility in reconstructing the probabilities of attribute values while achieving a low error bound.

We also investigated how to adapt our randomization protocol in other applications. We combine our randomization protocol with the knowledge on the variables dependencies contained in a Bayesian network and collect users' sensitive information in a privacy-preserving manner. We use the Bayesian network to approximate the distribution of users' responses using a set of low-dimensional contingency tables. We use randomized responses to ensure these marginals are differentially private. These noisy tables, generated from the Bayesian network, enable us to estimate the full-dimensional data distribution. The differential pri-

vate Bayesian network generated in *PrivBayes* [1] considers a centralized setting where the Laplace noise is added to the dataset. Since the server can access the true values of the clients and can become a single point of failure. Our protocol overcomes this limitation by creating a decentralized setting, where the server can collect randomized responses from each client in both steps. The server has no access to the client's true values, which makes our protocol the best candidate to collect randomized responses to generate privacy preservation surveys.

We also utilized the α-geometric mechanism, a discretized version of Laplace distribution. We combined α-geometric noise with our randomized response algorithm to create a hybrid approach for collecting users' responses in a non-trusted environment. We showed that our proposed hybrid approach satisfies the privacy guarantee and maximizes the utility of the perturbed dataset.

In this work, we also presented different contributions to analyze the conditions at which fuel consumption occurs in vehicles and understand how to reduce it by intervening in the scenario. We provided a collection of data from sensors installed on buses used as public transport. We tested both approximate algorithms, driven by the BIC score and brute force, to compare the algorithms ability to converge to the same resulting networks. We evaluated their results by adopting Granger causality, a third-party criterion, based on the time series formed in time by the observed variables. According to the Granger causality, we can also rank the alternatives, even when multiple Bayesian networks share the same score. We also compared Bayesian networks by using their ability to perform feature selection and predict the target variable.

We show, by means of experiments, the effect of noise on the privacy and utility of a privacy mechanism suitable for the contingency tables. We use information theory to characterize the notion of utility by means of experimental analysis in the differential private protocol. We use the *log-likelihood* ratio G^2/G-test to build a confidence interval on the utility of the responses on the basis of observed noisy values due to noise addition by the Laplace mechanism for privacy preservation.

In future work, we would like to use the hash function to store these contingency tables to reduce the computation and communication overheads. We can execute our protocol using the hash function on low-memory devices. We would also love to work on federated learning, where we can deploy our randomization protocol to many clients (e.g., mobile devices or whole organization) collaboratively to train a machine learning model under the

orchestration of a non-trusted central server (e.g., service provider), while keeping the training data decentralized. This setting enables focused collection and data minimization and uses our randomization mechanism to mitigate many privacy risks and costs resulting from traditional, non-trusted centralized machine learning.

Bibliography

[1] J. Zhang, G. Cormode, C. M. Procopiuc, D. Srivastava, and X. Xiao, "Privbayes: Private data release via bayesian networks," *ACM Transactions on Database Systems (TODS)*, vol. 42, no. 4, pp. 1–41, 2017.

[2] D. Barth-Jones, "The're-identification'of governor william weld's medical information: a critical re-examination of health data identification risks and privacy protections, then and now," *Then and Now (July 2012)*, 2012.

[3] J. Bennett, S. Lanning, *et al.*, "The netflix prize," in *Proceedings of KDD cup and workshop*, vol. 2007, p. 35, New York, NY, USA., 2007.

[4] M. Barbaro, T. Zeller, and S. Hansell, "A face is exposed for aol searcher no. 4417749," *New York Times*, vol. 9, no. 2008, p. 8, 2006.

[5] A. Jungherr, G. Rivero, and D. Gayo-Avello, *Retooling politics: How digital media are shaping democracy*. Cambridge University Press, 2020.

[6] F. du Pin Calmon and N. Fawaz, "Privacy against statistical inference," in *2012 50th annual Allerton conference on communication, control, and computing (Allerton)*, pp. 1401–1408, IEEE, 2012.

[7] M. S. Alvim, M. E. Andrés, K. Chatzikokolakis, P. Degano, and C. Palamidessi, "Differential privacy: on the trade-off between utility and information leakage," in *International Workshop on Formal Aspects in Security and Trust*, pp. 39–54, Springer, 2011.

[8] P. Cuff and L. Yu, "Differential privacy as a mutual information constraint," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pp. 43–54, 2016.

[9] W. Wang, L. Ying, and J. Zhang, "On the relation between identifiability, differential privacy, and mutual-information privacy," *IEEE Transactions on Information Theory*, vol. 62, no. 9, pp. 5018–5029, 2016.

[10] Q. Li, Z. Wen, Z. Wu, S. Hu, N. Wang, Y. Li, X. Liu, and B. He, "A survey on federated learning systems: vision, hype and reality for data privacy and protection," *arXiv preprint arXiv:1907.09693*, 2019.

[11] G. Cormode, S. Jha, T. Kulkarni, N. Li, D. Srivastava, and T. Wang, "Privacy at scale: Local differential privacy in practice," in *Proceedings of the 2018 International Conference on Management of Data*, pp. 1655–1658, 2018.

[12] G. Schwarz, "Estimating the Dimension of a Model," *The Annals of Statistics*, vol. 6, no. 2, pp. 461 – 464, 1978.

[13] L. Sweeney, "Achieving k-anonymity privacy protection using generalization and suppression," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 10, no. 05, pp. 571–588, 2002.

[14] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam, "l-diversity: Privacy beyond k-anonymity," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 1, no. 1, pp. 3–es, 2007.

[15] N. Li, T. Li, and S. Venkatasubramanian, "t-closeness: Privacy beyond k-anonymity and l-diversity," in *2007 IEEE 23rd International Conference on Data Engineering*, pp. 106–115, IEEE, 2007.

[16] C. Dwork and K. Nissim, "Privacy-preserving datamining on vertically partitioned databases," in *Annual International Cryptology Conference*, pp. 528–544, Springer, 2004.

[17] T. Dalenius, "Towards a methodology for statistical disclosure control," *statistik Tidskrift*, vol. 15, no. 429-444, pp. 2–1, 1977.

[18] C. Dwork and M. Naor, "On the difficulties of disclosure prevention in statistical databases or the case for differential privacy," *Journal of Privacy and Confidentiality*, vol. 2, no. 1, 2010.

[19] A. Blum, K. Ligett, and A. Roth, "A learning theory approach to noninteractive database privacy," *Journal of the ACM (JACM)*, vol. 60, no. 2, pp. 1–25, 2013.

[20] S. Chawla, C. Dwork, F. McSherry, and K. Talwar, "On privacy-preserving histograms," *arXiv preprint arXiv:1207.1371*, 2012.

[21] W. Du and Z. Zhan, "Using randomized response techniques for privacy-preserving data mining," in *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 505–510, 2003.

[22] C. Dwork, "Differential privacy, in, automata, languages and, programming, m. bugliesi, et. al., editors," 2006.

[23] C. Dwork and J. Lei, "Differential privacy and robust statistics," in *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pp. 371–380, 2009.

[24] C. Dwork, "Differential privacy: A survey of results," in *International conference on theory and applications of models of computation*, pp. 1–19, Springer, 2008.

[25] E. Levina and P. Bickel, "The earth mover's distance is the mallows distance: Some insights from statistics," in *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, vol. 2, pp. 251–256, IEEE, 2001.

[26] L. Sweeney, "Uniqueness of simple demographics in the us population," *LIDAP-WP4, 2000*, 2000.

[27] P. Golle, "Revisiting the uniqueness of simple demographics in the us population," in *Proceedings of the 5th ACM Workshop on Privacy in Electronic Society*, pp. 77–80, 2006.

[28] A. Narayanan and V. Shmatikov, "How to break anonymity of the netflix prize dataset," *arXiv preprint cs/0610105*, 2006.

[29] A. Narayanan and V. Shmatikov, "Robust de-anonymization of large sparse datasets," in *2008 IEEE Symposium on Security and Privacy (sp 2008)*, pp. 111–125, IEEE, 2008.

[30] C. Dwork, A. Roth, *et al.*, "The algorithmic foundations of differential privacy.," *Foundations and Trends in Theoretical Computer Science*, vol. 9, no. 3-4, pp. 211–407, 2014.

[31] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Theory of cryptography conference*, pp. 265–284, Springer, 2006.

[32] C. Dwork, "A firm foundation for private data analysis," *Communications of the ACM*, vol. 54, no. 1, pp. 86–95, 2011.

[33] K. Nissim, S. Raskhodnikova, and A. Smith, "Smooth sensitivity and sampling in private data analysis," in *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pp. 75–84, 2007.

[34] J. Lee and C. Clifton, "How much is enough? choosing ϵ for differential privacy," in *International Conference on Information Security*, pp. 325–340, Springer, 2011.

[35] P. Laud and A. Pankova, "Interpreting epsilon of differential privacy in terms of advantage in guessing or approximating sensitive attributes," *arXiv preprint arXiv:1911.12777*, 2019.

[36] J. Hsu, M. Gaboardi, A. Haeberlen, S. Khanna, A. Narayan, B. C. Pierce, and A. Roth, "Differential privacy: An economic method for choosing epsilon," in *2014 IEEE 27th Computer Security Foundations Symposium*, pp. 398–410, IEEE, 2014.

[37] D. Bernau, G. Eibl, P. W. Grassal, H. Keller, and F. Kerschbaum, "Quantifying identifiability to choose and audit ϵ in differentially private deep learning," *arXiv preprint arXiv:2103.02913*, 2021.

[38] S. L. Warner, "Randomized response: A survey technique for eliminating evasive answer bias," *Journal of the American Statistical Association*, vol. 60, no. 309, pp. 63–69, 1965.

[39] F. McSherry and K. Talwar, "Mechanism design via differential privacy," in *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07)*, pp. 94–103, IEEE, 2007.

[40] M. Hardt and G. N. Rothblum, "A multiplicative weights mechanism for privacy-preserving data analysis," in *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, pp. 61–70, IEEE, 2010.

[41] A. Roth and T. Roughgarden, "Interactive privacy via the median mechanism," in *Proceedings of the forty-second ACM symposium on Theory of computing*, pp. 765–774, ACM, 2010.

[42] F. Liu, "Generalized gaussian mechanism for differential privacy," *IEEE Transactions on Knowledge and Data Engineering*, vol. 31, no. 4, pp. 747–756, 2018.

[43] I. Mironov, "On significance of the least significant bits for differential privacy," in *Proceedings of the 2012 ACM conference on Computer and communications security*, pp. 650–661, 2012.

[44] D. P. Team, "Learning with privacy at scale differential privacy team," December 2017.

[45] Ú. Erlingsson, V. Pihur, and A. Korolova, "Rappor: Randomized aggregatable privacy-preserving ordinal response," in *Proceedings of the 2014 ACM SIGSAC conference on computer and communications security*, pp. 1054–1067, 2014.

[46] B. Barak, K. Chaudhuri, C. Dwork, S. Kale, F. McSherry, and K. Talwar, "Privacy, accuracy, and consistency too: a holistic solution to contingency table release," in *Proceedings of the twenty-sixth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pp. 273–282, 2007.

[47] D. Jurafsky, *Speech & language processing*. Pearson Education India, 2000.

[48] C. Zhang and S. Zhang, *Association rule mining: models and algorithms*, vol. 2307. Springer, 2003.

[49] T. K. Leen, T. G. Dietterich, and V. Tresp, "Advances in neural information processing systems 13, papers from neural information processing systems (nips) 2000, denver, co," 2001.

[50] T. Bedford and R. M. Cooke, "Probability density decomposition for conditionally dependent random variables modeled by vines," *Annals of Mathematics and Artificial intelligence*, pp. 245–268, 2001.

[51] S. M. Diesburg and A.-I. A. Wang, "A survey of confidential data storage and deletion methods," *ACM Computing Surveys (CSUR)*, vol. 43, no. 1, pp. 1–37, 2010.

[52] P. Samarati and S. C. d. Vimercati, "Access control: Policies, models, and mechanisms," in *International School on Foundations of Security Analysis and Design*, pp. 137–196, Springer, 2000.

[53] L. Fan and H. Jin, "A practical framework for privacy-preserving data analytics," in *Proceedings of the 24th International Conference on World Wide Web*, pp. 311–321, 2015.

[54] B.-C. Lin, S.-H. Wu, Y.-T. Tsou, and Y. Huang, "Ppdca: Privacy-preserving crowdsensing data collection and analysis with randomized response," in *2018 IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1–6, IEEE, 2018.

[55] Z. Qin, Y. Yang, T. Yu, I. Khalil, X. Xiao, and K. Ren, "Heavy hitter estimation over set-valued data with local differential privacy," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pp. 192–203, 2016.

[56] G. Fanti, V. Pihur, and Ú. Erlingsson, "Building a rappor with the unknown: Privacy-preserving learning of associations and data dictionaries," *Proceedings on Privacy Enhancing Technologies*, vol. 2016, no. 3, pp. 41–61, 2016.

[57] M. Beck, P. Bhatotia, R. Chen, C. Fetzer, T. Strufe, *et al.*, "Privapprox: privacy-preserving stream analytics," in *2017 {USENIX} Annual Technical Conference ({USENIX}{ATC} 17)*, pp. 659–672, 2017.

[58] R. Bassily, K. Nissim, U. Stemmer, and A. Thakurta, "Practical locally private heavy hitters," *arXiv preprint arXiv:1707.04982*, 2017.

[59] B. Ding, J. Kulkarni, and S. Yekhanin, "Collecting telemetry data privately," *arXiv preprint arXiv:1712.01524*, 2017.

[60] T. Wang, J. Blocki, N. Li, and S. Jha, "Locally differentially private protocols for frequency estimation," in *26th {USENIX} Security Symposium ({USENIX} Security 17)*, pp. 729–745, 2017.

[61] P. Kairouz, S. Oh, and P. Viswanath, "Differentially private multi-party computation: Optimality of non-interactive randomized response," *arXiv preprint arXiv:1407.1546*, 2014.

[62] P. Kairouz, S. Oh, and P. Viswanath, "Extremal mechanisms for local differential privacy," *arXiv preprint arXiv:1407.1338*, 2014.

[63] G. Cormode, T. Kulkarni, and D. Srivastava, "Marginal release under local differential privacy," in *Proceedings of the 2018 International Conference on Management of Data*, pp. 131–146, 2018.

[64] X. Ren, C.-M. Yu, W. Yu, S. Yang, X. Yang, J. A. McCann, and S. Y. Philip, "High-dimensional crowdsourced data publication with local differential privacy," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 9, pp. 2151–2166, 2018.

[65] Z. Zhang, T. Wang, N. Li, S. He, and J. Chen, "Calm: Consistent adaptive local marginal for marginal release under local differential privacy," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pp. 212–229, 2018.

[66] W. Qardaji, W. Yang, and N. Li, "Priview: practical differentially private release of marginal contingency tables," in *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, pp. 1435–1446, 2014.

[67] S. Boyd, S. P. Boyd, and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[68] A. Agresti, *An introduction to categorical data analysis*. John Wiley & Sons, 2018.

[69] Y. M. Bishop, S. E. Fienberg, and P. W. Holland, *Discrete multivariate analysis: theory and practice*. Springer Science & Business Media, 2007.

[70] V. Karwa, A. Slavković, *et al.*, "Inference using noisy degrees: Differentially private β-model and synthetic graphs," *Annals of statistics*, vol. 44, no. 1, pp. 87–112, 2016.

[71] J. Lee, Y. Wang, and D. Kifer, "Maximum likelihood postprocessing for differential privacy under consistency constraints," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 635–644, 2015.

[72] H. Zou and T. Hastie, "Regularization and variable selection via the elastic net," *Journal of the royal statistical society: series B (statistical methodology)*, vol. 67, no. 2, pp. 301–320, 2005.

[73] M. Scutari and J.-B. Denis, *Bayesian networks: with examples in R*. CRC press, 2014.

[74] I. A. Beinlich, H. J. Suermondt, R. M. Chavez, and G. F. Cooper, "The alarm monitoring system: A case study with two probabilistic inference techniques for belief networks," in *AIME 89*, pp. 247–256, Springer, 1989.

[75] D. J. Spiegelhalter, A. P. Dawid, S. L. Lauritzen, and R. G. Cowell, "Bayesian analysis in expert systems," *Statistical science*, pp. 219–247, 1993.

[76] M. Gaboardi, H. Lim, R. Rogers, and S. Vadhan, "Differentially private chi-squared hypothesis testing: Goodness of fit and independence testing," in *International conference on machine learning*, pp. 2111–2120, PMLR, 2016.

[77] D. Koller and N. Friedman, *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

[78] S. Kullback and R. A. Leibler, "On information and sufficiency," *The annals of mathematical statistics*, vol. 22, no. 1, pp. 79–86, 1951.

[79] C. Chow and C. Liu, "Approximating discrete probability distributions with dependence trees," *IEEE transactions on Information Theory*, vol. 14, no. 3, pp. 462–467, 1968.

[80] S. Russell and P. Norvig, "Artificial intelligence: a modern approach," 2002.

[81] P. Larranaga, B. Sierra, M. J. Gallego, M. J. Michelena, and J. M. Picaza, "Learning bayesian networks by genetic algorithms: a case study in the prediction of survival in malignant skin melanoma," in *Conference on Artificial Intelligence in Medicine in Europe*, pp. 261–272, Springer, 1997.

[82] R. R. Bouckaert, *Bayesian belief networks: from construction to inference*. PhD thesis, Heidelberglaan 8 3584 CS, 1995.

[83] D. M. Endres and J. E. Schindelin, "A new metric for probability distributions," *IEEE Transactions on Information theory*, vol. 49, no. 7, pp. 1858–1860, 2003.

[84] A. Blum, C. Dwork, F. McSherry, and K. Nissim, "Practical privacy: the sulq framework," in *Proceedings of the twenty-fourth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pp. 128–138, 2005.

[85] C. Dwork and K. Nissim, "Privacy-preserving datamining on vertically partitioned databases," in *Annual International Cryptology Conference*, pp. 528–544, Springer, 2004.

[86] A. Ghosh, T. Roughgarden, and M. Sundararajan, "Universally utility-maximizing privacy mechanisms," *SIAM Journal on Computing*, vol. 41, no. 6, pp. 1673–1693, 2012.

[87] R. Agrawal, R. Srikant, and D. Thomas, "Privacy preserving olap," in *Proceedings of the 2005 ACM SIGMOD international conference on Management of data*, pp. 251–262, 2005.

[88] World Health Organization, "The top 10 causes of death." https://www.who.int/news-room/fact-sheets/detail/the-top-10-causes-of-death, 2021. [Online; accessed 30-May-2021].

[89] Directorate-General for Environment European Commission, "Air Quality Standards." https://ec.europa.eu/environment/air/quality/standards.htm, 2021. [Online; accessed 30-May-2021].

[90] European Environment Agency, "Exceedances of air quality limit values due to traffic." https://www.eea.europa.eu/data-and-maps/indicators/exceedances-of-air-quality-objectives-7/assessment-2, 2017. [Online; accessed 30-May-2021].

[91] A. Stoica, "Pollution in eu wreaking havoc on human health." https://energyindustryreview.com/analysis/pollution-in-eu-wreaking-havoc-on-human-health/, 2018. [Online; accessed 30-May-2021].

[92] C. A. Pollino and C. Henderson, "Bayesian networks: A guide for their application in natural resource management and policy," Tech. Rep. 14, Landscape Logic, 2010.

[93] A. Schoen, A. Byerly, B. Hendrix, R. M. Bagwe, E. C. d. Santos, and Z. B. Miled, "A machine learning model for average fuel consumption in heavy vehicles," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 7, pp. 6343–6351, 2019.

[94] F. Perrotta, T. Parry, and L. C. Neves, "Application of machine learning for fuel consumption modelling of trucks," in *2017 IEEE International Conference on Big Data (Big Data)*, pp. 3810–3815, 2017.

[95] E. Moradi and L. Miranda-Moreno, "Vehicular fuel consumption estimation using real-world measures through cascaded machine learning modeling," *Transportation Research Part D: Transport and Environment*, vol. 88, p. 102576, 2020.

[96] Y. Yao, X. Zhao, C. Liu, J. Rong, Y. Zhang, Z. Dong, and Y. Su, "Vehicle fuel consumption prediction method based on driving behavior data collected from smartphones," *Journal of Advanced Transportation*, vol. 2020, March 2020.

[97] D. Rimpas, A. Papadakis, and M. Samarakou, "Obd-ii sensor diagnostics for monitoring vehicle operation and consumption," *Energy Reports*, vol. 6, pp. 55–63, 2020.

[98] J. Pavlovic, G. Fontaras, S. Broekaert, B. Ciuffo, M. Ktistakis, and T. Grigoratos, "How accurately can we measure vehicle fuel consumption in real world operation?," *Transportation Research Part D: Transport and Environment*, vol. 90, p. 102666, 2021.

[99] S. Wickramanayake and H. D. D. Bandara, "Fuel consumption prediction of fleet vehicles using machine learning: A comparative study," *2016 Moratuwa Engineering Research Conference (MERCon)*, pp. 90–95, 2016.

[100] T. Bousonville, M. Dirichs, and T. Krüger, "Estimating truck fuel consumption with machine learning using telematics, topology and weather data," in *2019 International Conference on Industrial Engineering and Systems Management (IESM)*, pp. 1–6, 2019.

[101] Delussu, Federico and Meo, Rosa, "Sensors data about fuel consumption in buses of public transport." https://github.com/rosameo/Sensors-Data-about-Fuel-Consumption-in-Buses, 2021. [Online; accessed 31-May-2021].

[102] Wikipedia, "CAN bus." https://en.wikipedia.org/wiki/CAN_bus, 2021. [Online; accessed 30-May-2021].

[103] J. Pearl, "The seven tools of causal inference, with reflections on machine learning," *Communications of the ACM*, vol. 62, no. 3, pp. 54–60, 2019.

[104] R. W. Robinson, "Counting unlabeled acyclic digraphs," in *Combinatorial Mathematics V* (C. H. C. Little, ed.), (Berlin, Heidelberg), pp. 28–43, Springer Berlin Heidelberg, 1977.

[105] I. Tsamardinos, L. E. Brown, and C. F. Aliferis, "The max-min hill-climbing bayesian network structure learning algorithm," *Machine learning*, vol. 65, no. 1, pp. 31–78, 2006.

[106] D. M. Chickering, "A transformational characterization of equivalent bayesian network structures," *arXiv preprint arXiv:1302.4938*, 2013.

[107] J. Pearl, M. Glymour, and N. P. Jewell, *Causal inference in statistics – A primer*. Wiley, 2016.

[108] D. Heckerman, D. Geiger, and D. M. Chickering, "Learning bayesian networks: The combination of knowledge and statistical data," *Machine Learning*, vol. 20, no. 3, pp. 197–243, 1995.

[109] C. W. Granger, "Investigating causal relations by econometric models and cross-spectral methods," *Econometrica: journal of the Econometric Society*, pp. 424–438, 1969.

[110] J. Breitung and B. Candelon, "Testing for short-and long-run causality: A frequency-domain approach," *Journal of econometrics*, vol. 132, no. 2, pp. 363–378, 2006.

[111] J. E. Contreras-Reyes and C. Hernández-Santoro, "Assessing granger-causality in the southern humboldt current ecosystem using cross-spectral methods," *Entropy*, vol. 22, no. 10, p. 1071, 2020.

[112] E. Wit, E. v. d. Heuvel, and J.-W. Romeijn, "'all models are wrong...': an introduction to model uncertainty," *Statistica Neerlandica*, vol. 66, no. 3, pp. 217–236, 2012.

[113] H. Akaike, "Information theory and an extension of the maximum likelihood principle," in *Selected papers of hirotugu akaike*, pp. 199–213, Springer, 1998.

[114] T. Verma, J. Pearl, *et al.*, "Equivalence and synthesis of causal models," 1991.

[115] N. Friedman, I. Nachman, and D. Pe'er, "Learning bayesian network structure from massive datasets: The" sparse candidate" algorithm," *arXiv preprint arXiv:1301.6696*, 2013.

[116] M. Gasse, A. Aussem, and H. Elghazel, "A hybrid algorithm for bayesian network structure learning with application to multi-label learning," *Expert Systems with Applications*, vol. 41, no. 15, pp. 6755–6772, 2014.

[117] M. Scutari, "bnlearn - an R package for Bayesian network learning and inference." https://www.bnlearn.com/documentation/man/score.html, 2021. [Online; accessed 31-May-2021].

[118] T. S. Verma and J. Pearl, "On the equivalence of causal models," in *Proceedings of the Sixth Conference on Uncertainty in Artificial Intelligence (UAI1990)*, 2013.

[119] D. H. Vu, K. M. Muttaqi, and A. P. Agalgaonkar, "A variance inflation factor and backward elimination based robust regression model for forecasting monthly electricity demand using climatic variables," *Applied Energy*, vol. 140, pp. 385–394, 2015.

[120] "E-Book: Bayesian Networks |& Bayesialab - A Practical Introduction for Researchers." https://library.bayesia.com/articles/#!bayesialab-knowledge-hub/book, October 2015. accessed on 8th of July, 2021.

[121] I. Shpitser, T. VanderWeele, and J. M. Robins, "On the validity of covariate adjustment for estimating causal effects," 2012.

[122] G. M. Ljung and G. E. P. Box, "On a measure of lack of fit in time series models," *Biometrika*, vol. 65, pp. 297–303, 08 1978.

[123] D. Heckerman, *A Tutorial on Learning with Bayesian Networks*, pp. 33–82. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008.

[124] L. Lai, S.-W. Ho, and H. V. Poor, "Privacy–security trade-offs in biometric security systems—part i: Single use case," *IEEE Transactions on Information Forensics and Security*, vol. 6, no. 1, pp. 122–139, 2010.

[125] A. Haeberlen, B. C. Pierce, and A. Narayan, "Differential privacy under fire.," in *USENIX Security Symposium*, vol. 33, 2011.

[126] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor, "Our data, ourselves: Privacy via distributed noise generation," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 486–503, Springer, 2006.

[127] C. Dwork and G. N. Rothblum, "Concentrated differential privacy," *arXiv preprint arXiv:1603.01887*, 2016.

[128] A. Kraskov, H. Stögbauer, and P. Grassberger, "Estimating mutual information," *Physical review E*, vol. 69, no. 6, p. 066138, 2004.

[129] J. Liao, O. Kosut, L. Sankar, and F. P. Calmon, "A general framework for information leakage," 2017.

[130] M. Lopuhaä-Zwakenberg, B. Škorić, and N. Li, "Information-theoretic metrics for local differential privacy protocols," *arXiv preprint arXiv:1910.07826*, 2019.

[131] J. H. McDonald, *Handbook of biological statistics*, vol. 2. sparky house publishing Baltimore, MD, 2009.

[132] J. Pearl, *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan kaufmann, 1988.